Advanced Learning Methodologies for Biomedical Applications

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Han-Tai Shiao

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Vladimir S. Cherkassky, Advisor

October 2017

Acknowledgments

First of all, I would like to express my deepest thanks to my advisor, Professor Vladimir Cherkassky, who guided me through my graduate study and research work.

He has always skillfully moved between letting me figure things out myself, and pointing me in the right direction. I sincerely appreciate his advice and consistent encouragement.

I would also like to thank my Ph.D. committee members, Professor Jarvis Haupt, Professor Mingyi Hong, and Professor Wei Pan, for reviewing my thesis and giving valuable feedback on my work.

I thank my former and current colleagues in our lab, Dr. Feng Cai, Dr. Sauptik Dhar, Jieun Lee, Thomas Vacek, Adarsh Sivasankaran, Brandon Veber, Hsiang-Han Chen, Zhong Zhuang, and many others for their help and friendship.

Special thanks to Adarsh who generously offered his time for proofreading my draft. Special thanks to Dr. Dan Drake, an alumnus of University of Minnesota. He gave me the style files and skeleton of his thesis so that I can typeset my thesis in $\LaTeX$ easily.

Finally, I shall express my appreciation to my friends and families, for their assistance and encouragements over the years. Particularly, I appreciate the supports from my parents and parents-in-law.

Dedication

To my wife Ya-Yin.

Abstract

There has been a dramatic increase in application of statistical and machine learning methods for predictive data-analytic modeling of biomedical data. Most existing work in this area involves application of standard supervised learning techniques. Typical methods include standard classification or regression techniques, where the goal is to estimate an indicator function (classification decision rule) or real-valued function of input variables, from finite training sample. However, real-world data often contain additional information besides labeled training samples. Incorporating this additional information into learning (model estimation) leads to nonstandard/advanced learning formalizations that represent extensions of standard supervised learning. Recent examples of such advanced methodologies include semi-supervised learning (or transduction) and learning through contradiction (or Universum learning).

This thesis investigates two new advanced learning methodologies along with their biomedical applications. The first one is motivated by modeling complex survival data which can incorporate future, censored, or unknown data, in addition to (traditional) labeled training data. Here we propose original formalization for predictive modeling of survival data, under the framework of Learning Using Privileged Information (LUPI) proposed by Vapnik [1, 2]. Survival data represents a collection of time observations about events. Our modeling goal is to predict the state (alive/dead) of a subject at a pre-determined future time point. We explore modeling of survival data as binary classification problem that incorporates additional information (such as time of death, censored/uncensored status, etc.) under LUPI framework. Then we propose two advanced constructive Support Vector Machine (SVM)-based formulations: SVM+ and Loss-Order SVM (LO-SVM). Empirical results using simulated and real-life survival data indicate that the proposed LUPI-based methods are very effective (versus classical Cox regression) when the survival time does not follow classical probabilistic assumptions.

Second advanced methodology investigates a new learning paradigm for classification called Group Learning. This approach is motivated by modeling high-dimensional data when the number of input features is much larger than the number of training samples. There are two main approaches to solving such ill-posed problems: (a) selecting a small number of informative features via feature selection; (b) using all features but imposing additional complexity constraints, e.g., ridge regression, SVM, LASSO, etc. The proposed Group Learning method takes a different approach, by splitting all features into many (t) groups, and then estimating a classifier in reduced space (of dimensionality d/t). This approach effectively uses all features, but implements training in a lower-dimensional input space. Note that the formation of groups reflects application-domain knowledge. For example, in classifying of two-dimensional images represented as a set of pixels (original high-dimensional input space), appropriate groups can be formed by grouping adjacent pixels or "local patches" because adjacent pixels are known to be highly correlated. We provide empirical validation of this new methodology for two real-life applications: (a) handwritten digit recognition, and (b) predictive classification of univariate signals, e.g., prediction of epileptic seizures from intracranial electroencephalogram (iEEG) signal. Prediction of epileptic seizures is particularly challenging, due to highly unbalanced data (just 4–5 observed seizures) and patient-specific modeling. In a joint project with Mayo Clinic, we have incorporated the Group Learning approach into an SVM-based system for seizure prediction. This system performs subject-specific modeling and achieves robust prediction performance.

Contents

Acknowledgments i

Dedication ii

Abstract iii

List of Figures vii

List of Tables ix

Chapter 1. General Introduction 1

1.1. Motivations 1

1.2. Technical Contributions 2

1.3. Structure of the Thesis 3

Chapter 2. Background 5

2.1. Objective of Predictive Learning 5

2.2. Classification 6

2.3. SVM for Classification 6

2.4. Nonlinear SVM 11

Part 1. Extensions of LUPI 13

Chapter 3. Learning Using Privileged Information 14

3.1. Introduction 14

3.2. SVM+ 17

3.3. Loss-Order SVM 30

3.4. SVM+ versus Loss-Order SVM 34

3.5. Summary 34

Chapter 4. Modeling of Survival Data 36
4.1. Introduction 36
4.2. Survival Analysis 39
4.3. Problem Formalization 47
4.4. Modeling of Survival Data 48
4.5. Empirical Comparisons 54
4.6. Summary 65

Part 2. Group Learning 67

Chapter 5. Overview 68

Chapter 6. Group Learning for Images 73
6.1. Introduction 73
6.2. Problem Formalization 74
6.3. Qualitative Analysis 76
6.4. Empirical Results 78
6.5. Summary 81

Chapter 7. Group Learning for One-Dimensional Signals 83
7.1. Introduction 83
7.2. Problem Formalization 85
7.3. Feature Engineering 88
7.4. SVM System and Experimental Design 91
7.5. Postprocessing 96
7.6. Empirical Evaluation 99
7.7. Summary 103

Bibliography 105

List of Figures

2.1 Binary classification problem. 8

2.2 Nonseparable case for binary classification. 10

3.1 SVM+ maps the training data into different spaces. 18

3.2 An arbitrary linear correcting function would violates $\xi(x^*) \geq 0$. 21

3.3 The only two valid linear correcting functions if x^* is bounded. 22

3.4 Nonlinear correcting functions. 23

3.5 Estimates of computational time of SVM+ implemented in CVX. 25

3.6 Illustration of the enforcing of ordering in LO-SVM. 33

4.1 Example of survival data in a calendar-time scale. 40

4.2 Example of survival data in a study-time scale. 41

4.3 Estimated survival functions based on Cox model. 46

4.4 Example of survival data under the predictive problem setting. 48

4.5 Using Cox model in the predictive setting 50

4.6 The certainty measures for exact and censored observations. 51

4.7 The univariate privileged information x^* for LO-SVM. 52

4.8 Histograms of projections for training data with different noise levels. 56

5.1 MNIST digits 5 and 8 of size 28×28 pixels. 70

5.2 Partition digit 5 into four nonoverlapping patches of size 14×14 pixels. 71

5.3 Partition digit 8 into four nonoverlapping patches of size 14×14 pixels. 71

6.1 Illustration of Standard machine learning approach. 75

6.2 Illustration of Group Learning approach. 75

6.3 Training errors for standard SVM and Group Learning (patches 14-by-14). 79

6.4 Test errors for standard SVM and Group Learning (patches 14-by-14). 80

6.5 MNIST digit 5 and 8 of size 10×10 pixels. 81

6.6 Training errors for standard SVM and Group Learning (patches 10-by-10). 81

6.7 Test errors for standard SVM and Group Learning (patches 10-by-10). 82

7.1 Three feature encodings for iEEG data: BFB, FFT, and XCORR. 90

7.2 Proposed system design for training stage. 91

7.3 Proposed system design for prediction stage. 91

7.4 Predictive modeling in three time scales: 20 s, 1 hr, and 4 hr. 93

7.5 Histograms of Projections of SVM model for Dog-L4 training data. 96

7.6 Histograms of Projections of SVM model for Dog-L4 test data. 98

7.7 Univariate histogram of projections for test samples (a). 99

7.8 Univariate histogram of projections for test samples (b). 100

List of Tables

4.1 Test errors as a function of training sample size 57

4.2 Test errors as a function of censoring rate 58

4.3 Test errors as a function of noise level in survival time 59

4.4 Test errors as a function of training sample size without censored data 60

4.5 Summary of the statistical properties of survival datasets 62

4.6 Experimental results for survival datasets: linear models 63

4.7 Experimental results for survival datasets: nonlinear models 64

7.1 The number of interictal/preictal 1 hr segments for each dog 89

7.2 Experimental design for Dog-L4 under the unbalanced Setting 94

7.3 Predictions for 1 hr and 4 hr segments via different feature encodings 101

7.4 Summary of prediction performances on FP and FN error rates (%) 102

7.5 Summary of prediction performances on SS (%) and FPR per day 102

Chapter 1

General Introduction

1.1. Motivations

Learning is the process of estimating an unknown (input, output) dependency from a limited number of observations [3, 4]. Various applications in engineering, statistics, computer science, health sciences, and social sciences are concerned with estimating "good" predictive models from the available historical data (or training data), in order to use this model for predicting future samples (or test data). Predictive learning is of particular interest because it can objectively define the "usefulness" of the estimated model by predictive (generalization) capabilities.

Most existing work on predictive data analytics involves development and application of standard supervised learning techniques. Typical methods include standard classification or regression techniques, where the goal is to estimate an indicator function (classification decision rule) or real-valued function of input variables, from finite training sample. However, real-world data often contains additional information (besides labeled training samples). Incorporating this additional information into learning (model estimation) leads to non-standard/advanced learning formalizations that represent extensions of standard supervised learning. Recent examples of such advanced methodologies include semi-supervised learning (or transduction) and learning through contradiction (or Universum learning). This thesis investigates two new advanced learning methodologies along with their biomedical applications.

The first contribution is a new mathematical formalization for modeling survival data which can incorporate future, censored, or unknown data, in addition to (traditional) labeled training data. Here we propose original formalization for predictive modeling of survival data, under the framework of Learning Using Privileged Information (LUPI) proposed by Vapnik [1, 2]. Survival data represents a collection of time observations about events. Our modeling goal is to predict the state (alive/dead) of a subject at a pre-determined future time point. We explore modeling of survival data as binary classification problem that incorporates additional information (such as time of death, censored/uncensored status, etc.) under LUPI framework. Then we propose two advanced constructive Support Vector Machine (SVM)-based formulations: SVM+ and Loss-Order SVM (LO-SVM). Empirical results using simulated and real-life survival data indicate that the proposed LUPI-based methods are very effective (versus classical Cox regression) when the survival time does not follow classical probabilistic assumptions.

Our second technical contribution is a new learning method for classification called *Group Learning*. This method is motivated by modeling high-dimensional data when the number of input features is much larger than the number of training samples. There are two main approaches to solving such ill-posed problems: (a) selecting a small number of informative features via feature selection; (b) using all features but imposing additional complexity constraints, e.g., ridge regression, SVM, LASSO, etc. The proposed Group Learning method takes a different approach, by splitting all features into several (t) groups, and then estimating a classifier in reduced space (of dimensionality d/t). This approach effectively incorporates all input features, but implements training in a lower-dimensional input space. Note that the formation of groups reflects application-domain knowledge. For example, for classification of two-dimensional (2-D) images represented as a set of pixels (original high-dimensional input space), appropriate groups can be formed by grouping adjacent pixels or "local patches" because adjacent pixels are known to be highly correlated. We provide empirical validation of this new methodology for two real-life applications (a) handwritten digit recognition, and (b) predictive classification of univariate signals, e.g., prediction of epileptic seizures from intracranial electroencephalogram (iEEG) signal.

1.2. Technical Contributions

Next, we outline several technical contributions presented in this thesis.

- For modeling survival data:
  1. (1) We demonstrated application of SVM+, a LUPI-based learning method, for modeling survival data under binary classification setting. This approach incorporates the survival time information into learning and can also handle censored observations.
  2. (2) We proposed LO-SVM algorithm under LUPI framework. This LO-SVM can encode the survival time information effectively via a new ordering mechanism. We provided empirical comparisons between SVM+, LO-SVM, standard SVM, and statistical model for survival data.
- For Group Learning method:
  1. (1) We developed a Group Learning framework for modeling high-dimensional data under classification setting. The proposed learning methodology has been shown empirically its effectiveness in application of 2-D image recognition problems.
  2. (2) We provided an application of Group Learning to seizure prediction problems formalized as classification of univariate signals (i.e., iEEG recordings of brain activity). Empirical results show that the proposed seizure prediction system achieves high sensitivity and low false-positive error rate.

1.3. Structure of the Thesis

This thesis includes three major parts:

(1) Background description of predictive learning and SVM (in Chapter 2).

(2) Extensions of LUPI for modeling survival data (Chapters 3 and 4).

(3) Group Learning with application to prediction of epileptic seizures (Chapters 5, 6, and 7).

Part I on modeling survival data is based on the following publications and manuscript:

- H.-T. **Shiao** and V. Cherkassky, "Learning Using Privileged Information (LUPI) for modeling survival data," in *Neural Networks (IJCNN), 2014 International Joint Conference on*, Jul. 2014.

- H.-T. **Shiao**, T. Vacek, and V. Cherkassky, "LUPI-based approaches for modeling survival data," in *Proceedings of the International Workshop on Human is More Than a Labeler (BeyondLabeler), co-located with the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, Jul. 2016.
- H.-T. **Shiao**, T. Vacek, and V. Cherkassky, "Predictive modeling of survival data," submitted to IEEE Trans. Neural Netw. Learn. Syst., under review.

Part II on Group Learning is based on the following publications and manuscript:

- V. Cherkassky, B. Veber, J. Lee, H.-T. **Shiao**, E. Patterson, G. A. Worrell, and B. H. Brinkmann, "Reliable seizure prediction from EEG data," in *Neural Networks (IJCNN), 2015 International Joint Conference on*, Jul. 2015.
- H.-T. **Shiao**, V. Cherkassky, J. Lee, B. Veber, E. Patterson, B. H. Brinkmann, and G. A. Worrell, "SVM-based system for prediction of epileptic seizures from iEEG signal," *IEEE Trans. Biomed. Eng.*, vol. 64, pp. 1011–1022, May 2017.
- H.-T. **Shiao**, V. Cherkassky, "Group Learning: shallow deep learning," in preparation.
- H.-H. Chen, H.-T. **Shiao**, V. Cherkassky, "Online prediction system for epileptic seizures using iEEG signal," in preparation.

Chapter 2

Background

This chapter reviews the basic concepts of predictive learning and Support Vector Machine (SVM). The content of this chapter mainly follows [3, 4, 5].

2.1. Objective of Predictive Learning

The process of learning is about estimating an unknown dependency or structure between the input and output of a system, based on a limited number of observations. The finite training set is denoted as independent identically distributed pairs

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n), \tag{2.1}$$

generated from a fixed but unknown probability measure $P(\mathbf{x}, y)$. Suppose $f(\mathbf{x}, \omega)$ denote the set of functions estimated by the learning machine and indexed by ω. The quality of an approximation produced by the learning machine is measured by the loss $L(f(\mathbf{x}, \omega), y)$, or the discrepancy between the output y produced by the original system and the output $f(\mathbf{x}, \omega)$ produced by learning machine for a given input $\mathbf{x}$.

The goal of learning is to estimate the function $f(\mathbf{x}, \omega_0)$ that guarantees the smallest loss. That is, the goal is to find the function which minimizes the *risk functional*

$$R(\omega) = \iint L(f(\mathbf{x}, \omega), y) P(\mathbf{x}, y)\, \mathrm{d}\mathbf{x}\mathrm{d}y \tag{2.2}$$

over the set of functions supported by the learning machine using only training data (2.1). With finite data we cannot expect to find $f(\mathbf{x}, \omega_0)$ exactly. Instead, we can obtain $f(\mathbf{x}, \omega^*)$ which represents the estimate of the optimal solution obtained with finite training data using one learning procedure. The estimated function $f(\mathbf{x}, \omega^*)$ is expected to have good prediction accuracy for future (test) data. High prediction accuracy is normally called good *generalization* in the field of machine learning.

In practice, in order to evaluate a machine learning algorithm or compare different algorithms, we use the so called test error as the criteria. Suppose we are given a test set $(\mathbf{x}_j, y_j)$, $j = 1, \ldots, m$, the test error is defined as

$$\mathrm{E}_{\text{test}} = \sum_{j=1}^{m} L(f(\mathbf{x}_j, \omega^*), y_j). \tag{2.3}$$

2.2. Classification

For the generic learning problems, classification and regression are two common learning tasks. The differences between classification and regression are the outputs and loss functions. We will mainly deal with binary classification in this thesis.

The output of a binary classification system takes on only two values $y \in \{+1, -1\}$ corresponding to two classes. Hence, in learning machine, $f(\mathbf{x}, \omega)$, $\omega \in \Omega$, are a set of indicator functions. The most commonly used loss function for binary classification problem is the misclassification error defined as

$$L(f(\mathbf{x}, \omega), y) = \begin{cases} 0, & \text{if } y = f(\mathbf{x}, \omega), \\ 1, & \text{if } y \neq f(\mathbf{x}, \omega). \end{cases}$$

For completeness, we also give a brief description about regression here. The output of the system in a regression problem takes on real values: $y \in \mathbf{R}$. In a learning machine, $f(\mathbf{x}, \omega)$, $\omega \in \Omega$, are a set of functions with real values. A common loss function for regression problem is the squared error defined as

$$L(f(\mathbf{x}, \omega), y) = (y - f(\mathbf{x}, \omega))^2.$$

2.3. SVM for Classification

This section describes a family of learning algorithms known as Support Vector Machine and provides the fundamental mathematical formulation of SVM. SVM methodology was developed in Statistical Learning Theory [6], and later was adopted by researchers in machine learning, statistics, and signal processing.

According to Vapnik-Chervonenkis (VC) theory, the generalization bound for learning with finite samples is as follows,

$$R(\omega) \leq R_{\text{emp}}(\omega) + \Phi\left(\frac{h}{n}, \frac{\log \eta}{n}\right). \tag{2.4}$$

Detailed analysis suggests that the second term Φ, called the *confidence interval*, depends mainly on the VC-dimension (or the ratio h/n), whereas the first term (empirical risk) depends on parameter ω. SVM provides a special way to achieve small empirical risk (first term) using low complexity (second term) parameterizations. Therefore, SVM provides good generalization for future (test) data.

Consider a binary classification setting where we are given finite training data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$, with $\mathbf{x} \in \mathbf{R}^d$ and $y \in \{+1, -1\}$. The goal of SVM is to find the optimal decision function

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}, \mathbf{x}) + b$$

with good generalization performance.

Assuming that training data is linearly separable, there are many separating hyperplanes $(\mathbf{w}, \mathbf{x}) + b$ satisfying the constraints

$$y_i((\mathbf{w}, \mathbf{x}_i) + b) \geq 1,$$

for $i = 1, \ldots, n$. SVM method considers an optimal separating hyperplane, for which the margin (i.e., the distance between the closest data points to the hyperplane) is maximized [6, 7]. The VC dimension (model complexity) of an optimal separating hyperplane is

$$h \leq \min\left(\frac{r^2}{\Delta^2}, d\right) + 1, \tag{2.5}$$

where r is the radius of the smallest sphere that contains the training input vectors $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, and Δ is the margin. SVM implements structural risk minimization (SRM) inductive principle by keeping the value of empirical risk fixed (zero for the linearly separable case) and minimizing the confidence interval (by maximizing margin). The concept of margin is illustrated in Figure 2.1.

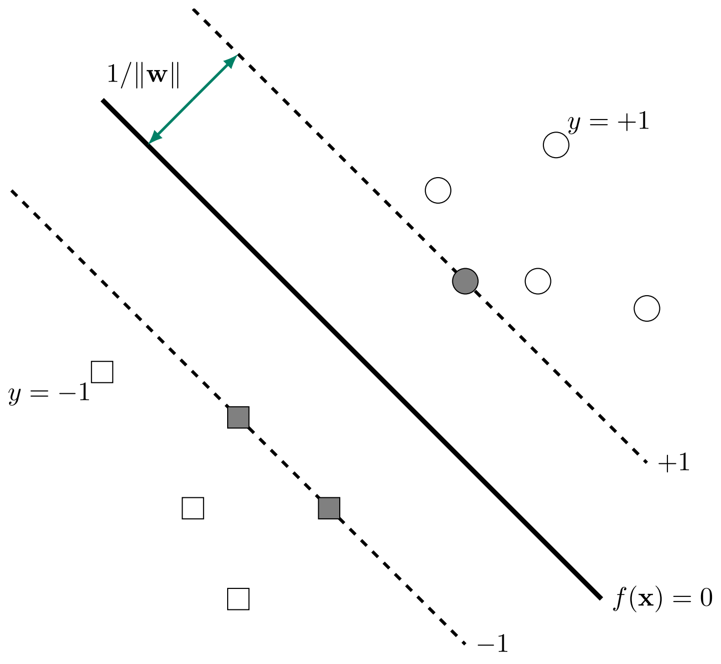


FIGURE 2.1. Binary classification problem, where circles denote samples from positive class and squares denote samples from negative class. The margin $1/\|\mathbf{w}\|$ is the distance between the closest data points to the hyperplane. The shaded circle(s) and square(s) represent the support vectors.

Maximization of margin is equivalent to minimization of $\|\mathbf{w}\|$. To this end, SVM solves the following optimization problem:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to} \quad y_i((\mathbf{w}, \mathbf{x}_i) + b) \geq 1, \quad i = 1, \ldots, n, \end{aligned} \tag{2.6}$$

with $\mathbf{w} \in \mathbf{R}^d$ and $b \in \mathbf{R}$ as the variables. That is, the problem of finding an optimal large-margin separating hyperplane for linearly separable data is reduced to a quadratic programming (QP) problem (2.6). It is common to solve (2.6) (the primal problem) in its dual form. According to convex optimization, an optimization problem has a dual form if the objective function and constraints are strictly convex [8]. In this case, solving the dual problem is equivalent to solving the original. The dual form of (2.6) is

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \\ & \boldsymbol{\alpha} \succeq 0, \end{aligned} \tag{2.7}$$

with $\boldsymbol{\alpha} \in \mathbf{R}_+^n$ as the variables. The optimal hyperplane decision function has the following form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i^* y_i (\mathbf{x}, \mathbf{x}_i) + \hat{b}, \tag{2.8}$$

where α_i^*, $i = 1, \ldots, n$, are the solutions of (2.7), and $\hat{b}$ is called the bias term.

The parameters α_i are generally known as Lagrange multipliers. Note that $\alpha_i \geq 0$ correspond to the constraints $y_i((\mathbf{w}, \mathbf{x}_i) + b) \geq 1$ in (2.6). That is, data points that satisfy the constraints with *inequality* have their multipliers in (2.8) equal zero. The data points that satisfy the constraints with *equality* are support vectors, and they have nonzero values of α_i^*. Therefore, the solution (2.8) depends only on a subset of data points, i.e., support vectors. The support vectors are illustrated in Figure 2.1 with shaded circle(s) and square(s). Once the Lagrange multipliers have been estimated, the bias term $\hat{b}$ can be calculated using any of the support vectors $(\mathbf{x}_s, y_s)$:

$$\hat{b} = y_s - \sum_{i=1}^{n} \alpha_i^* y_i (\mathbf{x}_s, \mathbf{x}_i). \tag{2.9}$$

When training data are not linearly separable, the empirical risk $R_{\text{emp}}(\mathbf{w})$ is no longer zero. That is, a few training samples are allowed to fall inside the margin (so called soft margin). One can introduce the nonnegative slack variables

$$\xi_i = \max(1 - y_i f(\mathbf{x}_i, \mathbf{w}), 0), \tag{2.10}$$

for $i = 1, \ldots, n$, to represent the deviations from margin borders, as illustrated in Figure 2.2. The empirical risk is then defined as

$$R_{\text{emp}}(\mathbf{w}) = \sum_{i=1}^{n} \xi_i.$$

In this case, SVM attempts to strike a balance between the goal of empirical risk minimization and margin maximization by solving the following optimization problem:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{subject to} && y_i((\mathbf{w},\mathbf{x}_i)+b) \geq 1-\xi_i, \quad i=1,\dots,n, \\ & && \boldsymbol{\xi} \succeq 0, \end{aligned} \tag{2.11}$$

with $\mathbf{w} \in \mathbf{R}^d$, $b \in \mathbf{R}$, and $\boldsymbol{\xi} \in \mathbf{R}_+^n$ as the variables.

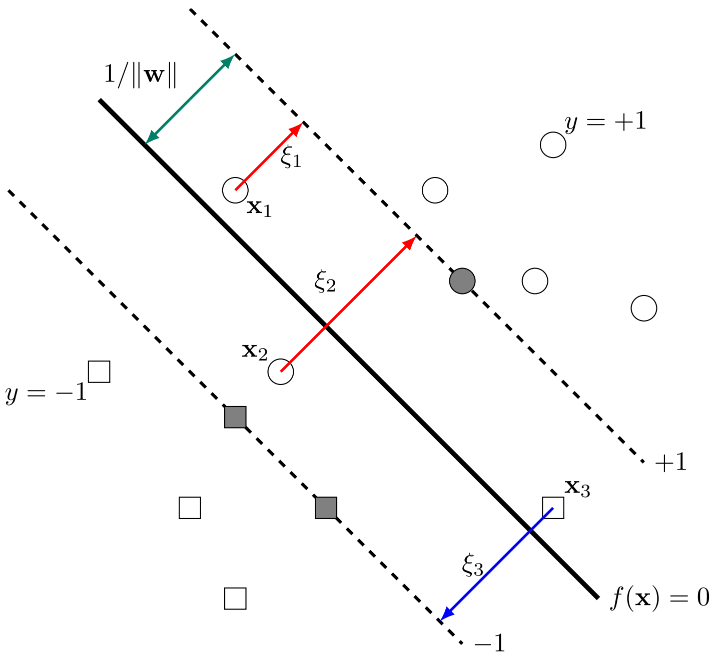


FIGURE 2.2. Nonseparable case for binary classification. Slack variables $\xi_i = 1 - y_i f(\mathbf{x}_i)$ correspond to the deviation from the margin borders. Three data points $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ are nonseparable, since they are within the margin. Data points $\mathbf{x}_2$ and $\mathbf{x}_3$ are misclassified, since they are on the wrong side of the decision boundary. Data point $\mathbf{x}_1$ is nonseparable, but classified correctly.

The $\|\mathbf{w}\|$ term in the objective function of (2.11) controls the size of margin. The ξ_i is the slack variable which indicates the deviation from the margin borders for sample

$(\mathbf{x}_i, y_i)$. The parameter C, selected by users, controls the trade-off between the complexity and proportion of nonseparable samples. The selection of C is known as *model selection*, and it is usually done via cross validation.

Problem (2.11) is also solvable in its dual form:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \\ & 0 \preceq \boldsymbol{\alpha} \preceq C, \end{aligned} \tag{2.12}$$

with $\boldsymbol{\alpha} \in \mathbf{R}_{+}^{n}$ as the variables. This optimization problem differs from that for the separable case (2.7) only with the inclusion of a maximum limit C in constraints. So (2.7), for linearly separable data, can be regarded as a special case of (2.12) with a very large value of parameter C. The optimal hyperplane decision function has the same form as for the separable case (2.8):

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i^* y_i (\mathbf{x}, \mathbf{x}_i) + \hat{b},$$

where α_i^*, $i = 1, \ldots, n$, are the solutions of (2.12) and the bias $\hat{b}$ is given in (2.9).

2.4. Nonlinear SVM

This section discuss how to construct large margin nonlinear decision boundary. The general approach is to map the input vector $\mathbf{x}$ into a high-dimensional feature space, and then construct an optimal hyperplane in this feature space [4, 6]. Obviously, hyperplanes in this feature space will correspond to nonlinear decision boundaries in the input space.

The conceptual motivation for the nonlinear mapping can be also provided by the VC generalization bound (2.4). Note that the empirical risk term in (2.4) can be always reduced to zero or a small value following the nonlinear mapping $\Psi(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{z}$ to a high-dimensional feature space. So a good generalization may be guaranteed, if the confidence interval Φ is small, which can be achieved by controlling the VC dimension, i.e., using large margin hyperplanes in the feature space.

To extend the linear SVM to a nonlinear one, we do not need to specify the nonlinear basis function Ψ explicitly. It is sufficient to provide a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ in the input space and substitute the dot product $(\mathbf{x}_i, \mathbf{x}_j)$ in (2.7), (2.8), (2.9), and (2.12) with $K(\mathbf{x}_i, \mathbf{x}_j)$. Hence, the nonlinear version of (2.12) is

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \\ & 0 \preceq \boldsymbol{\alpha} \preceq C, \end{aligned} \tag{2.13}$$

and the nonlinear SVM decision function is

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i) + \hat{b}, \tag{2.14}$$

with the bias term

$$\hat{b} = y_s - \sum_{i=1}^{n} \alpha_i^* y_i K(\mathbf{x}_s, \mathbf{x}_i). \tag{2.15}$$

The kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ effectively defines the (nonlinear) similarity metric between any two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ in the input space. Commonly used kernel functions include:

(1) Polynomial kernel of degree m (where m is a positive integer)

$$K(\mathbf{x}_i, \mathbf{x}_j) = ((\mathbf{x}_i, \mathbf{x}_j) + 1)^m.$$

(2) Radial basis function (RBF) kernel with width parameter σ

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right),$$

or in alternative form

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2 \right),$$

with $\gamma > 0$.

Part 1

Extensions of LUPI

Chapter 3

Learning Using Privileged Information

This chapter describes a learning paradigm called Learning Using Privileged Information (LUPI), a general methodology for utilizing additional information about training data. Two mathematical formulations under the LUPI framework, namely SVM+ and Loss-Order SVM, are introduced in this chapter as well.

3.1. Introduction

In a data-rich world, there often exists additional information about training data, which is not reflected in the labeled training samples $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$. This additional information is not known during prediction, so the goal of learning is to estimate a model $\hat{y} = f(\mathbf{x})$, as in standard inductive learning [4].

We use the following examples to illustrate this possibility.

(1) *Handwritten digit recognition*, whereas the training data are known to be generated by several persons. In this case, each training sample has an additional person (group) label, and this information can be used to improve generalization. However, this group label is not available during the test stage, e.g., a classifier uses only pixels of an image for making prediction.

(2) *Medical diagnosis*, whereas the goal is to estimate a predictive classifier for diagnosing a disease based on the clinical tests, or input features $\mathbf{x}$, known at the time of initial examination. However, the available data may include additional patient history obtained later, such as pathological reports and advanced tests. This additional information, denoted as feature vector $\mathbf{x}^*$, can be used for training a classifier. However, features $\mathbf{x}^*$ are not known during testing (or prediction stage of a classifier). Hence, the goal of learning is to estimate a

decision rule $\hat{y} = f(\mathbf{x})$, a function of input features only known at the time of initial examination.

(3) *Time series prediction*, whereas the goal is to predict a future event based on past (known) values of a time series. For instance, given historical information about the values of a stock market index up to moment t, we would like to predict if this index at moment $t + \delta t$ will be higher or lower (roughly speaking to make a decision to sell or buy). Clearly, the historical data (used for training) also includes the values of the index *around* $t + \delta t$, e.g., two days before and after $t + \delta t$. This additional information can be incorporated into training, and it may improve generalization.

All these examples include potentially useful extra information about the training inputs and/or training outputs. Further, this additional information is ubiquitous: it usually exists for almost any machine learning problem. Formally, the training data are provided in the following form

$$(\mathbf{x}_1, \mathbf{x}_1^*, y_1), \ldots, (\mathbf{x}_n, \mathbf{x}_n^*, y_n), \tag{3.1}$$

where $\mathbf{x}_i \in X$, $\mathbf{x}_i^* \in X^*$, and $y_i \in \{+1, -1\}$. These data are generated according to a fixed but unknown $P(\mathbf{x}, \mathbf{x}^*, y)$, where the $(\mathbf{x}, y)$ is the *usual* labeled training data and $(\mathbf{x}^*)$ denotes the *additional* information.

Since the additional information is available only for the training set and *is not* available for the test set, it is called *privileged information*, and the new machine learning paradigm is called Learning Using Privileged Information [**1**, **2**, **9**, **10**].

In the LUPI paradigm, the goal is to find among a given set of indicator functions $f(\mathbf{x}, \omega)$, $\omega \in \Omega$, the function $f(\mathbf{x}, \omega^*)$ that guarantees the smallest probability of incorrect classification. Here we have exactly the same goal of minimizing (2.2) as in the classical paradigm described in Section 2.1, i.e., to find the best classification function in the admissible set. However, during the training stage, we have more information, i.e., we have triplets $(\mathbf{x}, \mathbf{x}^*, y)$ instead of pairs $(\mathbf{x}, y)$ as in the classical paradigm. The privileged information $\mathbf{x}^*$ belongs to space X^*, which is, generally speaking, different from X. Note that the function representation $f(\mathbf{x}, \omega^*)$ does not depend on $\mathbf{x}^*$; however, $\mathbf{x}^*$ is involved in the estimation of $f(\mathbf{x}, \omega^*)$.

The privileged information has two common properties:

(1) it is available only for training samples, and not known for test samples;

(2) it should have an informative value for estimating a predictive model $\hat{y} = f(\mathbf{x})$.

These two properties suggest another useful interpretation of the privileged information: it can be viewed as additional feedback from an expert teacher, provided during learning [1]. We give two examples of privileged information that could be generated by an *intelligent* teacher.

Example 1. Suppose that our goal is to find a rule $\hat{y} = f(\mathbf{x})$ that classifies biopsy images $\mathbf{x}$ into two categories y: cancer ($y = +1$) and noncancer ($y = -1$). Here images are in a pixel space X, and the classification rule has to be in the same space. However, the standard diagnostic procedure also includes a pathologist's report $\mathbf{x}^*$ that describes his/her impression about the image in a high-level holistic language X^* (for example, "aggressive proliferation of cells of type A among cells of type B," etc.). The problem is to use the pathologist's reports $\mathbf{x}^*$ as additional information (along with images $\mathbf{x}$) in order to make a better classification rule for images $\mathbf{x}$ just in pixel space X. Classification by a pathologist is a time-consuming procedure, so fast decisions during surgery should be made without consulting him or her.

Example 2. Let our goal be finding a rule that predicts the outcome y of a surgery in three weeks after it, based on information $\mathbf{x}$ available before the surgery. In order to find the rule in the classical paradigm, we use pairs $(\mathbf{x}, y)$ from previous patients. However, for previous patients, there is also additional information $\mathbf{x}^*$ about procedures and complications during surgery, development of symptoms in one or two weeks after surgery, and so on. Although this information is not available *before* surgery, it does exist in historical data and thus can be used as extra information in order to construct a rule that is better than the one obtained without using that information. The issue is how large an improvement can be achieved.

According to VC theory, LUPI is a general methodology for utilizing privileged information about training data, and it constructs a new SRM structure on the training set (3.1) [4, 1, 2]. This task may appear similar to the development of new structures for nonstandard learning formulations, where the new structures incorporate additional constraints, such as a large margin for test samples for transduction, or a large number of contradictions for Universum SVM [3, 4, 1]. The difference is that in earlier nonstandard SVM-based formulations the appropriate structures have been defined in the same feature space X. In contrast, under LUPI setting, additional privileged information is specified in a different feature space X^*, but this information is related to errors in the input space X.

This chapter is organized as follows. Section 3.2 describes the SVM+ technique under LUPI framework. Section 3.3 explains the Loss-Order SVM (LO-SVM) technique under LUPI framework, specializing for univariate privileged information. Section 3.4 provides a comparison between SVM+ and LO-SVM. Finally, a summary is presented in Section 3.5.

3.2. SVM+

SVM+ is a method for function estimation extended from SVM, and it allows one to solve machine learning problems under the LUPI paradigm [1, 2]. The SVM+ method performs learning in two different spaces:

(1) *decision space* $\mathcal{Z}$ (via the mapping $\Psi(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{z}$)

This is the space where the decision function needs to be estimated, and it is the same feature space as used in standard SVM.

(2) *correcting space* $\mathcal{Z}^*$ (via the mapping $\Psi^*(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{z}^*$)

This is the space where the correcting function, reflecting the privileged information about the training data, is defined. This privileged information is encoded in the form of additional constraints on the training errors (e.g., slack variables) in the decision space.

The mappings of inputs $\mathbf{x}$ and $\mathbf{x}^*$ in SVM+ are illustrated in Figure 3.1. This figure shows linear models $\psi(\mathbf{x}, \omega)$ and $\psi^*(\mathbf{x}^*, \omega^*)$ in the decision space and correcting space. However, the mapping $\Psi(\mathbf{x})$ and $\Psi^*(\mathbf{x})$ themselves may be nonlinear, and both the decision and correcting spaces can use different kernels. The considerations of kernel for correcting space are discussed later in Section 3.2.1. It should be noted that the final performance of SVM+ models would depend on the quality of the privileged information.

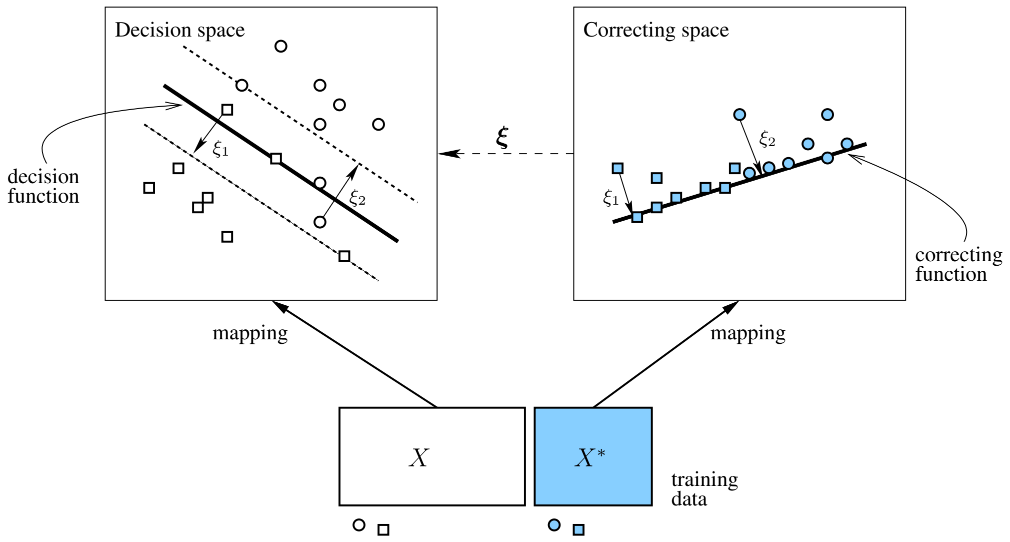


FIGURE 3.1. SVM+ maps the training data simultaneously into the decision space and the correcting space. Slack variables in the decision space are represented by the correcting functions in the correcting space.

Mathematically, SVM+ estimates the decision function $f(\mathbf{z}) = (\mathbf{w}, \mathbf{z}) + b$ from the training data (3.1) by solving the following optimization problem:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\|\mathbf{w}^*\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} & y_i((\mathbf{w}, \mathbf{z}_i) + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n, \\ & \xi_i = (\mathbf{w}^*, \mathbf{z}_i^*) + b^*, \quad i = 1, \ldots, n, \\ & \boldsymbol{\xi} \succeq 0, \end{array} \tag{3.2}$$

with $\mathbf{w} \in \mathbf{R}^d$, $b \in \mathbf{R}$, $\mathbf{w}^* \in \mathbf{R}^k$, $b^* \in \mathbf{R}$, and $\boldsymbol{\xi} \in \mathbf{R}^n_+$ as the variables. In (3.2), $C > 0$ and $\gamma > 0$ are two hyperparameters, whereas $\mathbf{z}$ and $\mathbf{z}^*$ are the feature mappings of $\mathbf{x}$ and $\mathbf{x}^*$. Furthermore, the correcting function $\xi_i = (\mathbf{w}^*, \mathbf{z}_i^*) + b^* = (\mathbf{w}^*, \psi^*(\mathbf{x}_i^*)) + b^*$ in (3.2) represents a linear model for the slack. These correcting functions provide additional constraints on the slack variables (errors) in the decision space. The correcting functions are nonnegative because they correspond to slack variables.

The SVM+ replaces the slack variables in standard SVM with a slack function defined in the correcting space. Through the slack function, the additional privileged information is used to model the loss function, which guides the hyperplane learning in the decision space [11]. In contrast, the slack variables in standard SVM are only constrained to nonnegative values, which is often less effective than the slack function in SVM+.

Relative to standard SVM formulation (2.11), this new formulation (3.2) includes:

(1) additional term $\|\mathbf{w}^*\|$ restricting the capacity (or VC dimension) of the correcting function;

(2) extra constraints reflecting the influence of the privileged information on training errors.

Hyperparameters C and γ control the trade-off between the capacity of decision function (i.e., margin size), the capacity of correcting function, and the number of training errors. Setting γ to zero yields the standard SVM formulation (2.11).

Assuming nonlinear kernels for both decision and correcting spaces, SVM+ has four tuning parameters (two kernel parameters along with C and γ). Model selection with four tuning parameters is quite challenging, and using resampling approaches with finite data often results in very unstable estimated models [4].

Problem (3.2) is commonly solved in its dual form:

$$\begin{aligned}
\text{maximize}\quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(\mathbf{z}_i,\mathbf{z}_j) \\
& -\frac{1}{2\gamma}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i+\beta_i-C)(\alpha_j+\beta_j-C)(\mathbf{z}_i^*,\mathbf{z}_j^*) \\
\text{subject to}\quad & \sum_{i=1}^{n}\alpha_i y_i = 0, \\
& \sum_{i=1}^{n}(\alpha_i+\beta_i-C) = 0, \\
& \boldsymbol{\alpha}\succeq 0, \\
& \boldsymbol{\beta}\succeq 0,
\end{aligned} \tag{3.3}$$

with $\boldsymbol{\alpha}\in\mathbf{R}_+^n$ and $\boldsymbol{\beta}\in\mathbf{R}_+^n$ as the variables.

The solution to SVM+ includes a decision function

$$\begin{aligned} f(\mathbf{x}) = (\mathbf{w}, \mathbf{z}) + \hat{b} = (\mathbf{w}, \psi(\mathbf{x})) + \hat{b} \\ = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + \hat{b}, \end{aligned} \tag{3.4}$$

and a correcting function

$$\begin{aligned} \xi(\mathbf{x}^*) = (\mathbf{w}^*, \mathbf{z}^*) + \hat{b}^* = (\mathbf{w}^*, \psi^*(\mathbf{x}^*)) + \hat{b}^* \\ = \frac{1}{\gamma} \sum_{i=1}^{n} (\alpha_i + \beta_i - C) K^*(\mathbf{x}^*, \mathbf{x}_i^*) + \hat{b}^*, \end{aligned} \tag{3.5}$$

where α_i and β_i, $i = 1, \ldots, n$, are the solutions of (3.3), and K^* is a kernel function in the correcting space. However, only (3.4), the model estimated in the decision space $\mathcal{Z}$, is used for prediction.

Comparing between the solutions of SVM and SVM+, we observe that the SVM solution in (2.8) depends only on the values of pairwise similarities between training vectors defined by the Gram matrix K of elements $K(\mathbf{x}_i, \mathbf{x}_j)$ (which defines similarity between vectors $\mathbf{x}_i$ and $\mathbf{x}_j$). However, the SVM+ solution in (3.4) and (3.5) uses two expressions of similarities between training vectors: one ($K(\mathbf{x}_i, \mathbf{x}_j)$ for $\mathbf{x}_i$ and $\mathbf{x}_j$) that comes from space X, and another one ($K^*(\mathbf{x}_i^*, \mathbf{x}_j^*)$ for $\mathbf{x}_i^*$ and $\mathbf{x}_j^*$) that comes from space of privileged information X^* [10].

Additionally, one can show that $\mathbf{w}$ and $\mathbf{w}^*$ can be expressed in terms of training samples:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{z}_i,$$

$$\mathbf{w}^* = \frac{1}{\gamma} \sum_{i=1}^{n} (\alpha_i + \beta_i - C)\, \mathbf{z}_i^*,$$

based on the Karush-Kuhn-Tucker (KKT) conditions.

3.2.1. Kernels in Correcting Space

In this section, we explore several properties of the correcting function, and argue that a (nonlinear) kernel should be used in the correcting space. The correcting function (in the correcting space) represents a unique way that SVM+ handles the additional privileged information. That is, the SVM+ approach assumes the decision function (3.4) interact differently with training data according to the privileged information via the correcting function (3.5). In SVM+, the correcting function is a real-valued function with its characteristics:

(1) Since the correcting function models the slack variables (in the decision space), as shown in Figure 3.1, the function values have to be nonnegative, $\xi(\mathbf{x}_i^*) \geq 0$, for $i = 1, \ldots, n$. Graphically, training samples in the correcting space have to lie on one side of the corresponding correcting function.

(2) According to the definition of slack variables, $\xi(\mathbf{x}_i) = \max(1 - y_i f(\mathbf{x}_i, \mathbf{w}), 0)$, we know $\xi(\mathbf{x}_i)$ is strictly greater than zero if the data point $\mathbf{x}_i$ falls within the margin, such as data points $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ in Figure 2.2.

(3) A correcting function has to pass through points with zero slack variables. Those points include the support vectors (such as the shaded circle(s) and square(s) in Figure 2.2) and the separable data points.

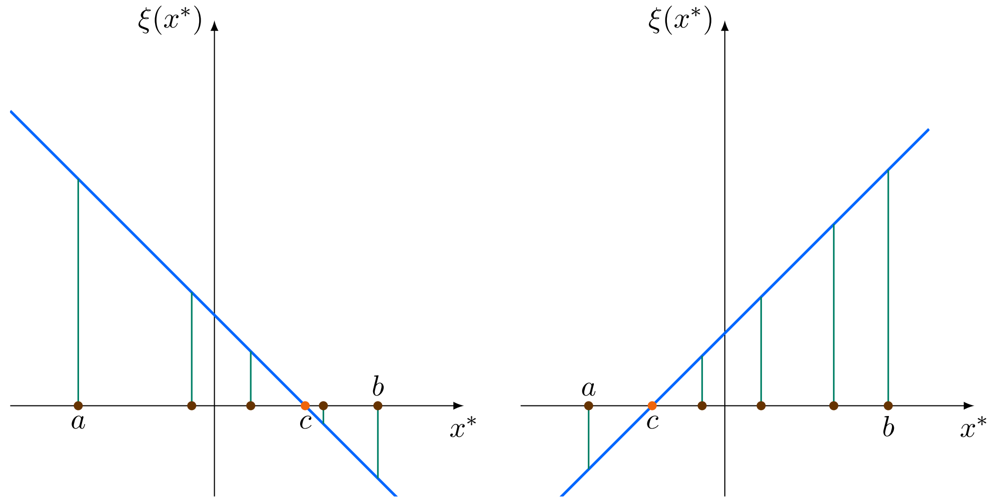


FIGURE 3.2. An arbitrary linear correcting function would violates $\xi(x^*) \geq 0$ at $x^* = b$ (left panel) or $x^* = a$ (right panel).

Now suppose the privileged information is univariate, i.e., $x^* \in \mathbf{R}$ or the dimensionality of the privileged information space X^* is one. It should be obvious that the correcting function $\xi(x^*)$ cannot be any linear function in the correcting space. As shown in Figure 3.2, an arbitrary linear function $\xi(x^*)$ cannot guarantee the properties of a correcting function. First, a few univariate privileged information would violate $\xi(x^*) \geq 0$, for instance, $\xi(b) < 0$ (left panel) and $\xi(a) < 0$ (right panel) in Figure 3.2. Consequently, not all points (privileged information) lie on one side of $\xi(x^*)$. Second, this linear function only pass through $x^* = c$ with $\xi(c) = 0$.

Meanwhile, if we assume that the univariate privileged information is bounded, $x^* \in [a, b]$, the only linear function satisfies the properties is the one that passes through $x^* = b$ with negative slope or through $x^* = a$ with positive slope. Both cases are shown in Figure 3.3. However, linear correcting function with either $\xi(b) = 0$ or $\xi(a) = 0$ is not ideal and realistic. They both implicitly allow only one support vector. In summary, linear correcting function is not going to be useful, and it might not even work in most situations.

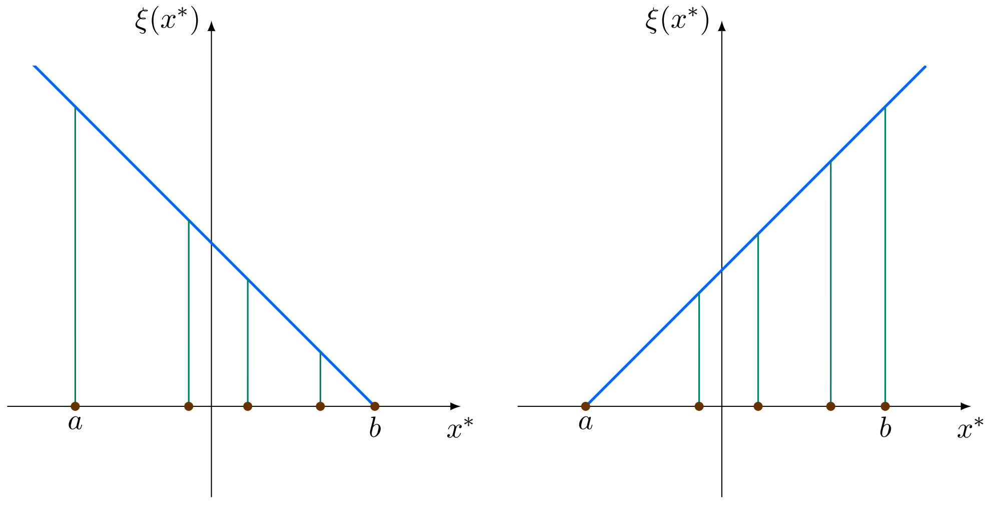


FIGURE 3.3. The only two valid linear correcting functions if $x^* \in [a, b]$.

In order to satisfy all three properties of a correcting function, a nonlinear correcting function would be a proper choice. Figure 3.4 shows three possible choices of nonlinear correcting functions, namely quadratic, exponential, and sigmoid functions. All three functions have all data points on one side of the function. The quadratic function passes through data points around the vertex with $\xi(x^*) = 0$ or close to zero. The exponential and sigmoid functions can pass through a wide range of data points. A valid nonlinear correcting function would still need to ensure the kernel matrix K^* satisfy the Mercer's conditions (symmetric and positive semi-definite). In practice, a nonlinear correcting function can be obtained by specifying a nonlinear mapping $\Psi^*(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{z}^*$. Further, using the "kernel trick" described in Section 2.4, it should suffice to specific a kernel function $K^*(\mathbf{x}_i^*, \mathbf{x}_j^*)$ for constructing a nonlinear function (hyperplane) in the correcting space.

To sum up, we use the univariate privileged information and properties of a correcting function to argue that a suitable correcting function should be nonlinear, and a kernel should be used in the correcting space. This argument can be extended to high-dimensional privileged information as well.

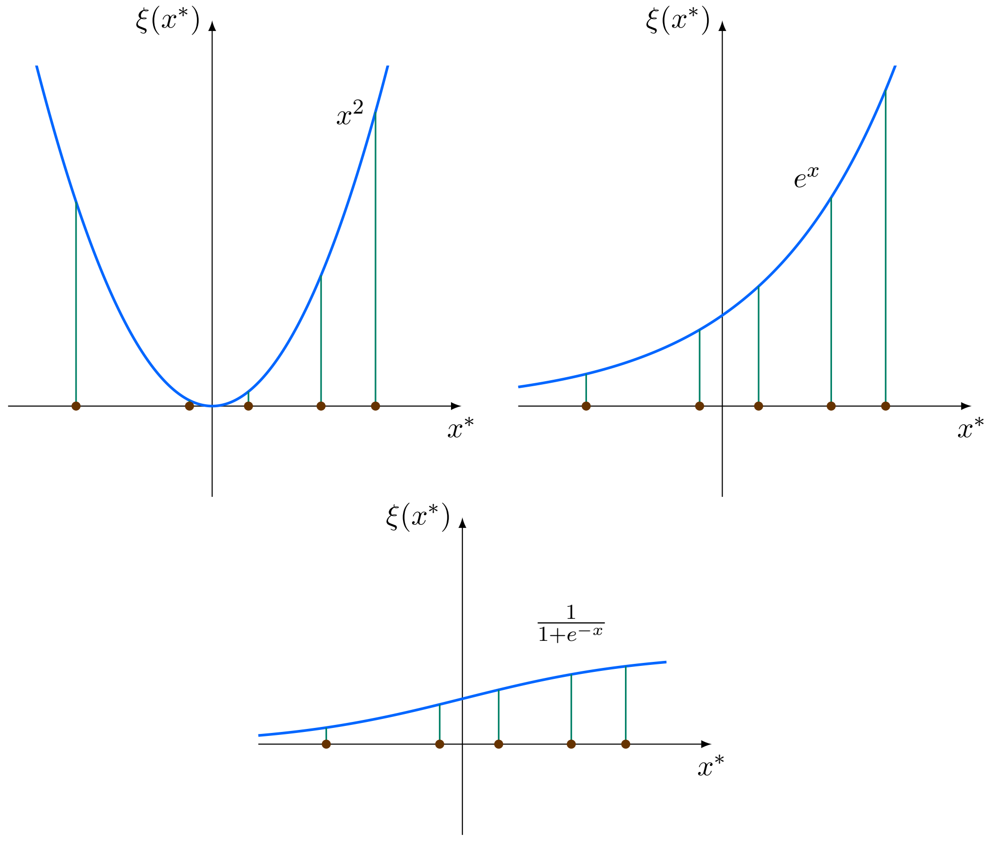


Figure 3.4. Using nonlinear function (quadratic, exponential, sigmoid function) as the correcting function.

3.2.2. Implementation

SVM+ model selection is very difficult due to the fact that the kernelized version of SVM+ binary classifier has four tuning parameters. Hence, the computationally efficient solution of SVM+ optimization becomes critical.

The process of training of standard SVM (or SVM+) involves solving a large Quadratic Programming (QP) problem as introduced in Section 2.3 and 3.2. The computational complexity of solving the QP problem (2.12) in SVM training grows as $\mathcal{O}(n^3)$, where n is the sample size [12]. This is slow when n is large. One of the widely used fast algorithms to train SVM is the Sequential Minimal Optimization (SMO) algorithm proposed by John Platt [13]. This algorithm solves the dual problem of SVM (2.12) in an iterative way. Specifically, SMO breaks a large QP problem into a series of sub-QP problems of the smallest possible size. Each sub-QP problem has only two variables and the analytical solution can be found for this small QP problem, which makes the training much faster. The decision about which pair of variables should be optimized at the current iteration is done by rules of working set selection [13, 14]. This approach was implemented in the LIBSVM package [15] for standard SVM and made this package popular in the machine learning community.

Our initial implementations of SVM+ used a general-purpose convex optimization package CVX [16]. However, the scalability is an issue of using CVX as the solver of the QP problem (3.3). Figure 3.5 shows the empirical estimates of computational time for SVM+ implemented in CVX, as a function of training sample size. Clearly, it takes more than 4 minutes to find a solution for the QP problem (3.3) when the training size exceeds 1000 samples. Thus, current model selection strategies become impractical and infeasible, especially the one using exhaustive grid search. Notably, most recent academic papers seem to use general-purpose optimization for their LUPI implementations, as they have shown empirical comparisons only for small training sets (about 200 to 400 samples), and they did not address/describe the challenging issues of model selection. A typical quote from [17]: "On the data set of this size (a few thousand) we found it infeasible to run experiments using SVM+."

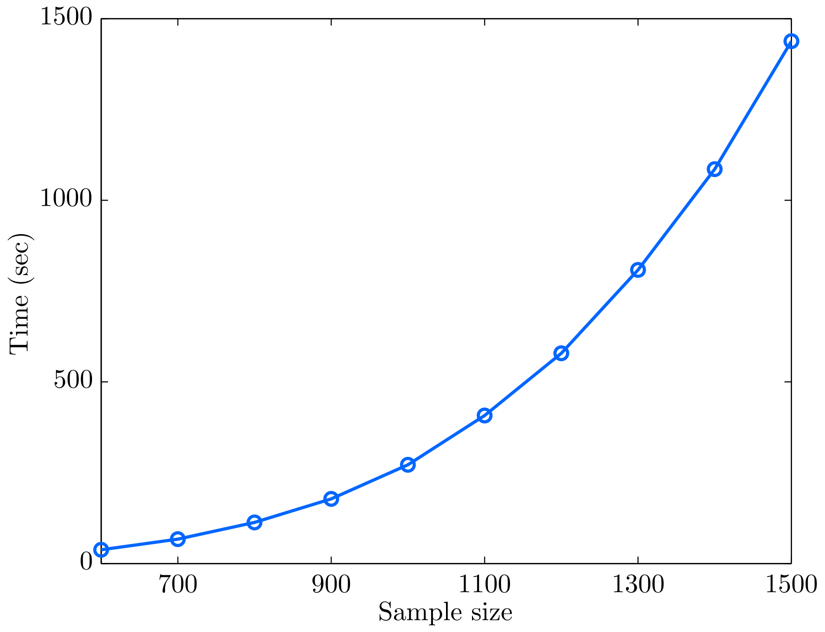


FIGURE 3.5. Empirical estimate of computational time of SVM+ implemented in CVX as a function of training sample size.

Alternatively, we chose to implement SVM+ using the *quadprog* package in Matlab Optimization Toolbox. The *quadprog* package was designed specifically for solving the QP problems, rather than general convex optimization problems. Our implementation involves the selection of the optimization option and also the stopping criterion (tolerance) optimally tuned for the SVM+ algorithm. Our experiments suggest that the *quadprog* implementation of SVM+ is capable of handling training data sets of size 1K-5K samples. That is, solving SVM+ optimization problem (for 1K-5K training samples) takes 2-12 seconds on a typical PC.

Next, we describe details of transforming SVM+ dual problem (3.3) into the canonical QP form for *quadprog*. The objective function of (3.3), reproduced below,

$$-\sum_{i=1}^{n}\alpha_i+\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(\mathbf{z}_i,\mathbf{z}_j)+\frac{1}{2\gamma}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i+\beta_i-C)(\alpha_j+\beta_j-C)(\mathbf{z}_i^*,\mathbf{z}_j^*)$$

can be translated into the following,

$$\frac{1}{2}\begin{bmatrix}\alpha_1\\ \vdots\\ \alpha_n\\ \alpha_1+\beta_1-C\\ \vdots\\ \alpha_n+\beta_n-C\end{bmatrix}^T \begin{bmatrix}H_1 & \\ & H_2\end{bmatrix}\begin{bmatrix}\alpha_1\\ \vdots\\ \alpha_n\\ \alpha_1+\beta_1-C\\ \vdots\\ \alpha_n+\beta_n-C\end{bmatrix} + \begin{bmatrix}-1\\ \vdots\\ -1\\ 0\\ \vdots\\ 0\end{bmatrix}^T\begin{bmatrix}\alpha_1\\ \vdots\\ \alpha_n\\ \alpha_1+\beta_1-C\\ \vdots\\ \alpha_n+\beta_n-C\end{bmatrix}, \tag{3.6}$$

where

$$H_1 = \begin{bmatrix} y_1y_1(\mathbf{z}_1,\mathbf{z}_1) & \cdots & y_1y_n(\mathbf{z}_1,\mathbf{z}_n)\\ \vdots & \ddots & \vdots\\ y_ny_1(\mathbf{z}_n,\mathbf{z}_1) & \cdots & y_ny_n(\mathbf{z}_n,\mathbf{z}_n)\end{bmatrix}, \quad H_2 = \frac{1}{\gamma}\begin{bmatrix}(\mathbf{z}_1^*,\mathbf{z}_1^*) & \cdots & (\mathbf{z}_1^*,\mathbf{z}_n^*)\\ \vdots & \ddots & \vdots\\ (\mathbf{z}_n^*,\mathbf{z}_1^*) & \cdots & (\mathbf{z}_n^*,\mathbf{z}_n^*)\end{bmatrix}.$$

The equality constraints in (3.3), $\sum \alpha_i y_i = 0$ and $\sum(\alpha_i+\beta_i-C)=0$, can be combined into the matrix form below,

$$\begin{bmatrix} y_1 & \cdots & y_n & 0 & & \cdots & & 0\\ 0 & \cdots & 0 & 1 & \cdots & 1 & 0 & \cdots & 0\end{bmatrix}\begin{bmatrix}\alpha_1\\ \vdots\\ \alpha_n\\ \alpha_1+\beta_1-C\\ \vdots\\ \alpha_n+\beta_n-C\end{bmatrix} = \begin{bmatrix}0\\ 0\end{bmatrix}. \tag{3.7}$$

Further, the inequality constraints $\boldsymbol{\alpha} \succeq 0$ and $\boldsymbol{\beta} \succeq 0$ in (3.3) are equivalent to

$$\begin{bmatrix} 1 & & -1 & \\ & \ddots & & \ddots \\ & 1 & & -1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \\ \alpha_1 + \beta_1 - C \\ \vdots \\ \alpha_n + \beta_n - C \end{bmatrix} \preceq \begin{bmatrix} C \\ \vdots \\ C \end{bmatrix}, \tag{3.8}$$

and

$$\begin{bmatrix} 0 \\ \vdots \\ 0 \\ -\infty \\ \vdots \\ -\infty \end{bmatrix} \preceq \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \\ \alpha_1 + \beta_1 - C \\ \vdots \\ \alpha_n + \beta_n - C \end{bmatrix} \preceq \begin{bmatrix} \infty \\ \vdots \\ \infty \\ \infty \\ \vdots \\ \infty \end{bmatrix}. \tag{3.9}$$

Finally, problem (3.3) can be rewritten in a concise QP form,

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2} x^T H x + f^T x \\ \text{subject to} \quad & A_{\text{eq}}\, x = b_{\text{eq}}, \\ & Ax \preceq b, \\ & L_B \preceq x \preceq U_B. \end{aligned} \tag{3.10}$$

The objective function $\frac{1}{2}x^T Hx + f^T x$ is given in (3.6) with x, H, and f defined as

$$x = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \\ \alpha_1 + \beta_1 - C \\ \vdots \\ \alpha_n + \beta_n - C \end{bmatrix}, \quad H = \begin{bmatrix} H_1 & \\ & H_2 \end{bmatrix}, \quad f = \begin{bmatrix} -1 \\ \vdots \\ -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

respectively. The equality constraint $A_{\text{eq}}\, x = b_{\text{eq}}$ is described in (3.7) with

$$A_{\text{eq}} = \begin{bmatrix} y_1 & \cdots & y_n & 0 & & \cdots & & & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 1 & 0 & \cdots & 0 \end{bmatrix}, \quad b_{\text{eq}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The inequality constraints $Ax \preceq b$ and $L_B \preceq x \preceq U_B$ are given in (3.8) and (3.9) with

$$A = \begin{bmatrix} 1 & & & -1 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & -1 \end{bmatrix}, \quad b = \begin{bmatrix} C \\ \vdots \\ C \end{bmatrix}, \quad L_B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -\infty \\ \vdots \\ -\infty \end{bmatrix}, \quad U_B = \begin{bmatrix} \infty \\ \vdots \\ \infty \\ \infty \\ \vdots \\ \infty \end{bmatrix}.$$

Although SVM+ can be transformed into a canonical QP form (3.10) and solved using any efficient QP solver, it is still not an ideal approach. One reason is that the number of dual variables (Lagrange multipliers) are doubled. Instead of solving n dual variables in (2.12), we need to solve $2n$ dual variables in (3.3) or (3.10). This could be an issue when the training sample size is significantly large.

While SVM+ can also be solved by an SMO-style algorithm [18, 19], the working set selection method is complicated, and the algorithm is also slow in practice. Moreover, it is unclear how to apply it to linear SVM+ without calculating the kernel matrix, which is becoming more crucial, due to rapidly increasing data in real-world applications [11]. Li *et al.* proposed two fast algorithms for solving linear and kernel SVM+ in [11]. We summarize both algorithms and comment on their applicabilities below.

(1) *Solution for linear SVM+*

By absorbing the bias terms into the weight vectors in both decision and correcting spaces, the optimization problem (3.2) (or its dual form (3.3)) can be rewritten as a special form of linear SVM introduced in [20]. Such linear SVM can be solved by using a dual coordinate descent method, and this approach has been implemented in the software package LIBLINEAR [21] which specializes in large linear classification problems. Conveniently, the LIBLINEAR package can be used for solving the linear SVM+ as well. Although using LIBLINEAR to solve the linear SVM+ can significantly enhance the scalability, a nonlinear mapping cannot be used in the correcting space. Utilizing the linear mapping in the correcting space could potentially diminish the advantages of SVM+ based on our arguments in Section 3.2.1.

(2) *Solution for kernel SVM+*

Instead of solving (3.2), Li *et al.* considered the l_2 loss SVM+ formulation

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\|\mathbf{w}^*\|^2 + C\sum_{i=1}^{n}\xi_i^2 \\ \text{subject to} \quad & y_i((\mathbf{w},\mathbf{z}_i)+b) \geq 1-\xi_i, \quad i=1,\ldots,n, \\ & \xi_i = (\mathbf{w}^*,\mathbf{z}_i^*)+b^*, \quad i=1,\ldots,n, \\ & \boldsymbol{\xi} \succeq 0, \end{aligned} \tag{3.11}$$

by employing the *squared* hinge loss. Problem (3.11) is almost identical to (3.2), except the ξ_i^2 term in the objective function. Then (3.11) can be rewritten into the l_2 loss ρ-SVM+ formulation based on the l_2 loss ρ-SVM introduced in [22]. The dual form of l_2 loss ρ-SVM+ shares a similar form with one-class SVM implemented in the software package LIBSVM [15]. Therefore, the LIBSVM package (with one-class option) can be readily used for solving the l_2 loss ρ-SVM+ (or equivalently l_2 loss SVM+).

The benefits of this strategy are twofold. First, both linear and nonlinear mappings can be applied to the decision and correcting spaces, in contrast to the limitation in linear SVM+. Second, since the SMO algorithm is the core implementation of LIBSVM, this strategy effectively takes advantage of SMO's efficiency for solving l_2 loss SVM+ (either linear or kernel). This strategy is likely to be faster than the SVM+ solver [19] in practice.

3.3. Loss-Order SVM

In this section, we introduce another LUPI-based formulation which utilizes univariate privileged information (i.e., the dimensionality of privileged information space X^* is one or $\mathbf{x}^* \in \mathbf{R}$).

This univariate information can be an *order oracle* [2, 23], which gives ordering of the training examples. For example, the posterior probability $P(y \mid \mathbf{x})$ defines a total ordering. Utilizing the ordering information via a special type of privileged information can result in improved generalization. For survival data described later in Chapter 4, the survival time information can be naturally considered as a good indicator for ordering.

Suppose the privileged information is related to (unknown) posterior probability $P(y \mid \mathbf{x})$, for instance,

$$P(y \mid \mathbf{x}) = g(x^*), \tag{3.12}$$

or

$$x^* = g^{-1}\big[P(y \mid \mathbf{x})\big], \tag{3.13}$$

where g is *any* nonnegative and nondecreasing function. That is, the actual probability values are not important, as long as their correct orderings are preserved. Conceptually, the privileged information x^*, or $g(x^*)$, can be viewed as a *confidence measure*, and the ordering of x^*-values can improve learning (generalization). Further, we consider the ordering provided by privileged information separately for each class, i.e.,

$$P(y = +1 \mid \mathbf{x}) = g_+(x^*) \tag{3.14}$$

for positive class, and

$$P(y = -1 \mid \mathbf{x}) = g_-(x^*) \tag{3.15}$$

for negative class.

The Loss-Order SVM (LO-SVM) has been proposed as a new formulation under LUPI setting utilizing univariate privileged information [2, 23]. Mathematically, the LO-SVM algorithm solves the optimization problem below:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|\mathbf{w}\|^2 + C_1 \sum_{i=1}^{n} \xi_i + C_2 \sum_{i=1}^{n} \zeta_i \\ &\text{subject to} && \boldsymbol{\xi} \succeq 0, \\ &&& M(\boldsymbol{\xi} + \boldsymbol{\zeta}) \succeq 0, \\ &&& y_i((\mathbf{w}, \mathbf{x}_i) + b) \geq 1 - (\xi_i + \zeta_i), \quad i = 1, \dots, n, \end{aligned} \tag{3.16}$$

with $\mathbf{w} \in \mathbf{R}^d$, $b \in \mathbf{R}$, $\boldsymbol{\xi} \in \mathbf{R}^n_+$, and $\boldsymbol{\zeta} \in \mathbf{R}^n_+$ as the variables. Here, M is an order-enforcing matrix that requires $\xi_i + \zeta_i \leq \xi_j + \zeta_j$ if $g(x_i^*) > g(x_j^*)$ for nonzero ξ_i and ξ_j. If $C_2 \gg C_1$, then $\zeta_i = 0$ for all i, and (3.16) is reduced to standard SVM (2.11). In practice, the tuning parameters C_1 and C_2 should be set so that $C_1 < C_2$. Otherwise, the solution of (3.16) is dominated by the ordering term.

The corresponding dual form of (3.16) is

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i,\mathbf{x}_j) \\
\text{subject to} \quad & \sum_{i=1}^{n}\alpha_i y_i = 0, \\
& \sum_{i=1}^{n}(\alpha_i+\beta_i-y_i\kappa_i-y_i\lambda_i-C_1)=0, \\
& \sum_{i=1}^{n}(\alpha_i+\mu_i-C_2)=0, \\
& \boldsymbol{\alpha}\succeq 0, \\
& \boldsymbol{\beta}\succeq 0, \\
& \boldsymbol{\kappa}\succeq 0, \\
& \boldsymbol{\lambda}\succeq 0, \\
& \boldsymbol{\mu}\succeq 0,
\end{aligned} \tag{3.17}$$

or, equivalently,

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i,\mathbf{x}_j) \\
\text{subject to} \quad & \sum_{i=1}^{n}\alpha_i y_i = 0, \\
& \sum_{i=1}^{n}(\alpha_i-y_i\kappa_i-y_i\lambda_i-C_1)\le 0, \\
& \sum_{i=1}^{n}(\alpha_i-C_2)\le 0, \\
& \boldsymbol{\alpha}\succeq 0, \\
& \boldsymbol{\kappa}\succeq 0, \\
& \boldsymbol{\lambda}\succeq 0,
\end{aligned} \tag{3.18}$$

with $\boldsymbol{\alpha}\in\mathbf{R}^n_+$, $\boldsymbol{\kappa}\in\mathbf{R}^n_+$, and $\boldsymbol{\lambda}\in\mathbf{R}^n_+$ as the variables.

This formulation tries to maintain correct orderings for nonseparable samples from the same class (e.g., training samples with nonzero slack variables) using the privileged information as a confidence measure. Hence, $g(x_i^*) > g(x_j^*)$ implies that we have higher confidence in the class label for training input $\mathbf{x}_i$. For SVM classification, the confidence ordering provided by the privileged information $g(x_i^*) > g(x_j^*)$ is shown in Figure 3.6. That is, training sample $\mathbf{x}_i$ should be closer to the margin border (or further away from the decision boundary), when compared with $\mathbf{x}_j$. Then ζ_i and ζ_j are the amounts of "movement" we need for enabling the ordering between $\mathbf{x}_i$ and $\mathbf{x}_j$, as illustrated in Figure 3.6.

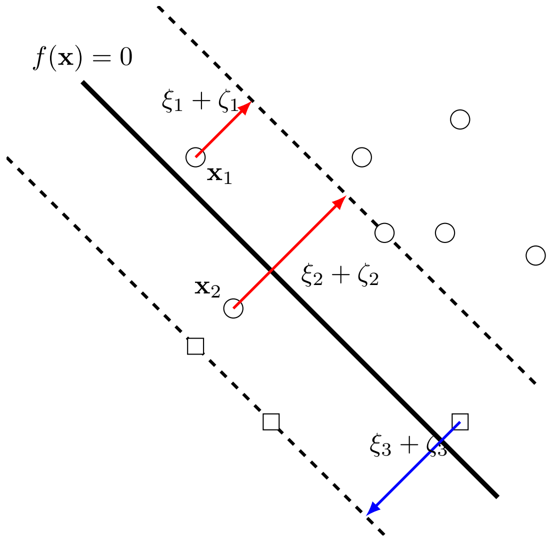


FIGURE 3.6. Given $g(x_1^*) > g(x_2^*)$, the ζ_1 and ζ_2 enforce the ordering $\xi_1 + \zeta_1 < \xi_2 + \zeta_2$. The ordering should assure that $\mathbf{x}_1$ is closer to the margin border, compared with $\mathbf{x}_2$.

The LO-SVM algorithm also involves the specification of a nondecreasing function g to ensure $g(x_i^*) > g(x_j^*)$ if $x_i^* > x_j^*$. In fact, the analytic form of g does not matter, as long as the ordering holds. In our implementation of LO-SVM, we use the identity function for g.

3.4. SVM+ versus Loss-Order SVM

Although both SVM+ and LO-SVM learning approaches incorporate the privileged information, there are several differences between them. We highlight those differences in this section.

The SVM+ uses the privileged information for modeling (or shaping) the slack variables directly, expecting an improvement in the hyperplane's separability and leading to better generalization. On the other hand, the LO-SVM considers the privileged information as a confidence measure (through an appropriate transformation g), which provides proper ordering for training samples.

The LO-SVM can be considered as a special case of SVM+ in two aspects:

(1) the privileged information is utilized in an one-dimensional correcting space;

(2) the correcting function is a nonlinear one composing with a series of inequalities which ensures the ordering of samples.

The SVM+ potentially has wider range of applications since there is no limitation in the dimensionality of the privileged information. Still, LO-SVM can be useful for training data with univariate privileged information. The domain of applications and performances for SVM+ and LO-SVM are discussed later in Chapter 4.

In terms of the computational implementation, the dual form of SVM+ (3.3) contains $2n$ Lagrange multipliers. In contrast, the dual problem of the LO-SVM (3.18) requires finding $3n$ Lagrange multipliers. Solving (3.18) can become difficult for large n, especially during the process of ordering.

Note that both SVM+ and LO-SVM formulations (3.2) and (3.16) are presented only for linear parameterization; they can be readily extended to nonlinear case using kernels [2].

3.5. Summary

The situation with existence of privileged information is very common. In fact, for almost all machine learning problems there exists some sort of privileged information especially in the era of Big Data. This information cannot be utilized by most standard supervised learning methods developed in statistics and machine learning. Nonetheless, effectively utilizing this privileged information during training usually results in improved generalization [2].

LUPI is a learning paradigm and a general methodology for utilizing privileged information about training data. The SVM+ and LO-SVM are two formulations under the LUPI framework. Technically, the SVM+ approach utilizes the privileged information into modeling the training errors (or slack variables) via the correcting function, imposing additional constraints on slack variables. The LO-SVM considers the privileged information as a confidence measure (through an appropriate transformation g), which provides proper ordering for training samples.

Chapter 4

Modeling of Survival Data

Privileged information often appears in modern complex clinical datasets. It could be a patient's survival time or a patient's medical history after diagnosis or medical procedure. The most common type of data that include the privileged information is the survival data. In this chapter, we demonstrate the modeling of survival data using LUPI-based learning methods.

4.1. Introduction

A significant proportion of the medical data is a collection of time-to-event observations. Classical examples are the time from birth to cancer diagnosis, from disease onset to death, and from patient's entry to a study to relapse. All these times are generally known as the *survival time*, even when the endpoint is something different from death. Methods for survival analysis developed in classical statistics have been used to model such data. Survival analysis focuses on the time elapsed from an initiating event to an event, or endpoint, of interest [24, 25]. The models of classical survival analysis describe the occurrence of events by means of survival curves and hazard rates, and analyze their dependence on covariates by means of regression [24]. One classical approach for survival curve estimation is the Cox regression based on the proportional hazards model assumption.

This modeling approach is generally known as the probabilistic (or descriptive) modeling. The probabilistic modeling assumes that

(1) the training and future data $(\mathbf{x}, y)$ are sampled independently from the same distribution $P(\mathbf{x}, y)$;
(2) the distribution $P(\mathbf{x}, y)$ is unknown but can be accurately estimated from data.

Classical statistics further makes specific assumptions about the parametric form of a distribution, and uses the training data to estimate its parameters. Clearly, the probabilistic approach may produce a poor predictive model if the parametric model is specified incorrectly or if the number of training samples is too small.

On the other hand, machine learning methods focus on estimating (learning) a good predictive model from available data. Specifically, this predictive modeling approach first specifies a wide set of admissible models $f(\mathbf{x}, \omega)$ indexed by abstract set of parameters ω, and then estimates the best predictive model from the training data. Further, predictive modeling requires proper specification of the problem setting (formalization) including the loss function used for measuring the quality of prediction, and specification of training and test data [3].

Learning is the process of estimating an unknown dependency between system's inputs and its output, based on a limited number of observations. When the observations are the survival data, the learning task becomes challenging. The main reason is that there are three types of information in survival data. The first type of information is the input variables, corresponding to ordinary inputs for most machine learning algorithms. Additionally, the survival time and the censoring are two other types of information included in survival data.

The survival time is the duration from the beginning of a study to the occurrence of an event. However, the survival time could be censored and it normally happens when an event is not observed at the end of a study. In such case, the observed survival time is only a lower bound of the true survival time. Both the survival time and censoring information are different from the ordinary input variables as they are only available in the training data.

Therefore, machine learning techniques have not been widely used for survival analysis for two main reasons [26]:

(1) First, the survival time is not necessarily observed in all samples. For instance, subjects might not experience the occurrence of event (death or relapse) during a study, or they dropped out before the end of study. In either case, the survival time is incomplete and only known "up-to-a-point." This is termed *censoring*

in biostatistics, which is different from the notion of "missing data" in machine learning.

(2) The second reason is methodological. Machine learning techniques are usually developed and applied under predictive setting, where the main goal is good prediction accuracy for future (or test) samples. In contrast, classical methods in statistics aim at estimating the true probabilistic model underlying available data. So the prediction accuracy is just one of several performance indices. The methodological assumption is that if an estimated model is "correct," then it should yield good predictions. Therefore, statistical methods often do not clearly differentiate between training (model estimation) and prediction (test) stages.

Several recent studies formalize the problem of survival analysis under the regression setting. For example, the Support Vector Machine (SVM) regression is used to estimate a predictive model for survival time [**27**, **28**, **29**]. However, the formalization using regression setting is intrinsically more difficult than classification [**4**].

Our approach for modeling survival data uses classification setting [**26**]. We also propose to incorporate the survival time and censoring information into learning by considering them as the *privileged information*. Learning Using Privileged Information (LUPI) is an advanced learning paradigm, where privileged information about training examples is provided during training stage [**2**]. This chapter investigates two LUPI-based formulations, SVM+ and LO-SVM, for modeling survival data. The SVM+ formulation considers both the survival time and censoring information. The LO-SVM formulation utilizes only the survival time information.

The chapter is organized as follows. Section 4.2 introduces necessary background on survival data and survival analysis. Section 4.3 defines the formalization for a survival prediction problem. Section 4.4 describes the proposed LUPI-based methods for modeling the survival data. The empirical comparisons for several synthetic and real-life datasets are presented in Section 4.5. Finally, a summary is included in Section 4.6.

4.2. Survival Analysis

This section provides general description of survival data. We also consider the basic parameters used in modeling survival data in this section. We shall define these quantities and show how they are interrelated. Common nonparametric and parametric models for survival analysis are introduced as well.

4.2.1. Survival Data

The survival data (also known as failure time data) are obtained by observing subjects (patients) from a certain initial time to either the occurrence of a predefined event or the end of the study. The predefined event is often the death of a subject, remission of disease, or relapse of disease, etc. The major difference between survival data and other types of medical data is that the time-to-event occurrences are not necessarily observed in all subjects [**24**].

A common feature of survival data is the possibility of *censored* observations. An illustration of how censored observations (or censored survival times) may arise is given in Figure 4.1. This figure illustrates a hypothetical clinical study where six subjects are observed over a time period to see whether a predefined event occurs. Figure 4.1 shows the observations as they occur in calendar time. Six subjects enter the study at different times and are then followed until the event occurs or until the closure of the study after t^*.

For subjects 2 and 6, the event was observed within the period of the study, and we have complete observation of their survival times. Subjects 1, 3, and 5 had not yet experienced the event when the study closed, so their survival times are censored. Note that the censoring present in these data cuts off intervals on the right-hand side. This censoring scheme is commonly known as *right censoring.*

In practice, a study may be terminated due to time or cost constraints, and its results are reported before events are observed for all subjects. However, the closure of the study is not the only reason for censoring. Right-censored observations will also

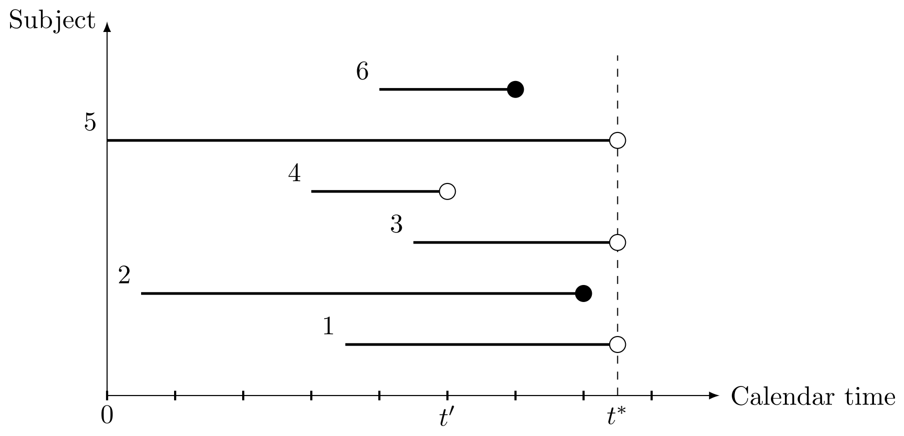


FIGURE 4.1. Example of survival data in calendar-time scale. Six subjects are followed in a hypothetical clinical study. The exact observations are indicated by solid dots, and the censored observations by hollow dots. The dashed vertical line indicates the closing date of the study t^*. Time t' marks the drop out time for subject 4.

occur when an individual withdraws from the study or when a subject is lost to follow-up. For example, subject 4 was initially included in study, but dropped out after t' (before the end of study). Hence, the survival times for subjects 1, 3, 4, and 5 are all right-censored observations. For these censored observations, we can only conclude that the true survival times are at least longer than the observed ones.

For statistical and survival analysis, one often focuses on the time from entry to the event of interest. Each individual will then have his or her own starting point, with time zero being the time of entrance into the study. Figure 4.2 shows the observations of the same hypothetical clinical study in its study-time scale.

There are other types of censoring scheme, such as *left censoring* or *interval censoring*. In left censoring scheme, we only know a subject has experienced the event of interest prior to the start of the study, but the exact event time is unknown. For instance, in a study to determine the distribution of the time until first marijuana use among high school boys [25], the question was asked, “When did you you first use marijuana?” One of the responses was “I have used it but cannot recall just when the first time was.” A boy who chose this response is indicating that the event had occurred prior to the boy's age at interview but the exact age at which he started using marijuana is unknown.

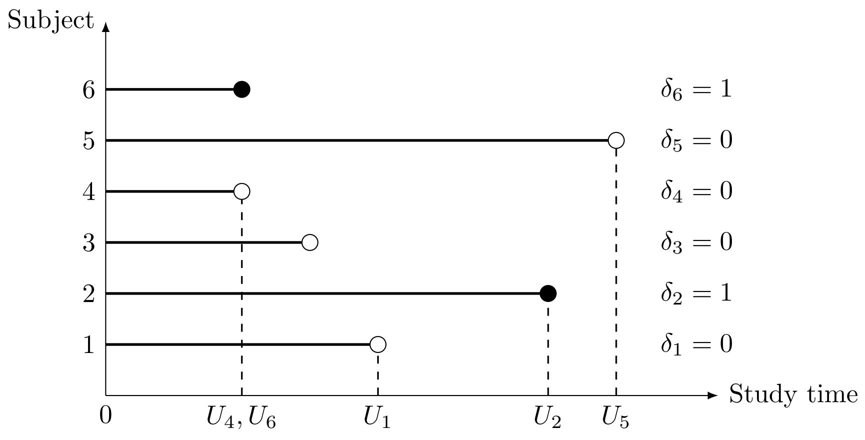


FIGURE 4.2. Illustrating the same observations in Figure 4.1 with the study-time scale. The time "0" for each subject is its entry to the study. The exact observations are indicated by solid dots, and the censored observations by hollow dots.

The interval censoring is a more general type of censoring when the event is only known to occur within an interval. Such interval censoring occurs when subjects in a clinical trial or longitudinal study have periodic follow-up and the subject's event time is only known to fall in an interval $(t_L, t_R]$, where t_L and t_R are the left endpoint and right endpoint of the censoring interval [25]. This type of censoring may also occur in industrial experiments where there is periodic inspection for proper functioning of equipment items.

We assume the right censoring scheme in this chapter.

4.2.2. Definitions

Suppose T denote the event time, such as death or lifetime; C denote the censoring time, e.g., the end of study. For right censoring scheme, the censoring time C is known and its value is always smaller than the event time: $C < T$. Hence, the survival outcome can be represented by a pair of random variables (U, δ). The event indicator δ represents whether the observed survival time U corresponds to an exact observation ($\delta = 1$) or a censored observation ($\delta = 0$). The value of U is equal to T if the event is observed, and to C if it is censored. Mathematically, U and δ are defined as

$$U = \min(T, C), \tag{4.1}$$

and

$$\delta = I(T \leq C) = \begin{cases} 0, & \text{for censored observation,} \\ 1, & \text{for exact observation,} \end{cases} \tag{4.2}$$

where $I(\cdot)$ is an indicator function. Using example of survival data in Figure 4.2, subjects 4 and 6 have the same observed survival time ($U_4 = U_6$), but their event indicators are different ($\delta_4 = 0$, and $\delta_6 = 1$).

In summary, survival data can be represented by a triplet

$$(\mathbf{x}_1, U_1, \delta_1), \ldots, (\mathbf{x}_n, U_n, \delta_n), \tag{4.3}$$

where $\mathbf{x}_i \in \mathbf{R}^d$, $U_i \in \mathbf{R}_+$, and $\delta_i \in \{0, 1\}$. Here $\mathbf{x}$ represents input features or covariates (such as a patient's clinical and demographic variables), U is time-to-event value, and δ is event indicator (also known as censoring variable).

Next, we introduce a few basic parameters commonly used in survival analysis and show they are interrelated. All functions introduced in this section are defined in domain $[0, t]$.

- Suppose $f(t)$[1] is the probability density function (PDF) of random variable T, then the survival function is defined as the probability of surviving beyond time t (or being event-free at t),

$$S(t) = \Pr(T > t) = \int_t^\infty f(u)\,\mathrm{d}u. \tag{4.4}$$

- Assuming that only one incidence could cause the death, the cumulative incidence function (CIF) is the probability of death before time t,

$$F(t) = \Pr(T \leq t), \tag{4.5}$$

  and $F(t)$ is the complement of the survival function $S(t)$,

$$F(t) = 1 - \Pr(T > t) = 1 - S(t). \tag{4.6}$$

[1] We use f to denote admissible functions previously, but overload the notation here. It should be distinguishable from the context.

- Suppose the first derivative of $F(t)$ exists, then the PDF of T can be found by

$$f(t) = \frac{\mathrm{d}F(t)}{\mathrm{d}t} = -\frac{\mathrm{d}S(t)}{\mathrm{d}t}, \tag{4.7}$$

which satisfies

$$F(t) = \int_0^t f(u)\,\mathrm{d}u. \tag{4.8}$$

- The hazard function $h(t)$ is the conditional probability of death occurring between $(t, t+\Delta t)$ given survival at least to time t,

$$h(t) = \lim_{\Delta t \to 0} \frac{\Pr(t \le T < t + \Delta \mid T \ge t)}{\Delta t} = \frac{f(t)}{S(t)}. \tag{4.9}$$

The hazard function is the instantaneous death rate for an individual who has survived to time t.

4.2.3. Remarks

- Hazard function and survival function

  It can be shown that the hazard function completely describes the survival function,

$$h(t) = \frac{f(t)}{S(t)} = \frac{1}{S(t)} \lim_{\Delta t \to 0} \frac{S(t) - S(t+\Delta t)}{\Delta t} = -\frac{\mathrm{d}}{\mathrm{d}t} \log S(t). \tag{4.10}$$

Then integrating both sides of (4.10) leads to

$$S(t) = \exp\left(-\int_0^t h(u)\,\mathrm{d}u\right). \tag{4.11}$$

- CIF, hazard function, and survival function

  From (4.9) we have $f(t) = h(t)S(t)$. Therefore,

$$F(t) = \int_0^t f(u)\,\mathrm{d}u = \int_0^t h(u)S(u)\,\mathrm{d}u. \tag{4.12}$$

Combining (4.11) with (4.12) gives

$$F(t) = \int_0^t h(u) \exp\left(-\int_0^u h(v)\,\mathrm{d}v\right)\mathrm{d}u. \tag{4.13}$$

- Cause-specific hazard function and CIF

If there are more than one cause of death D, we define a cause-specific hazard function as the instantaneous risk of dying of cause k,

$$h_k(t) = \lim_{\Delta t \to 0} \frac{\Pr(t \leq T < t + \Delta t, D = k \mid T \geq t)}{\Delta t}. \tag{4.14}$$

We also have

$$h(t) = \sum_{k=1}^{m} h_k(t), \tag{4.15}$$

as the death must be due to one (and only one) of the m causes. Similarly, the cause-specific CIF $F_k(t)$ is defined as

$$F_k(t) = \int_0^t h_k(u) S(u) \, \mathrm{d}u, \tag{4.16}$$

and the sum of all cause-specific CIF is the complement of the survival function,

$$1 - S(t) = \sum_{k=1}^{m} F_k(t). \tag{4.17}$$

4.2.4. Classical Survival Analysis

We briefly introduce the nonparametric approach and the Cox proportional hazards model in this section, as both are frequently used in survival analysis. One of the non-parametric approaches to estimate the survival function $S(t)$ from a sample of censored survival data is the Kaplan-Meier estimator [25, 30, 24].

Let $N(t)$ count the number of occurrences of the event in $[0, t]$, whereas $Y(t)$ is the number of individuals at risk "just before" time t. Without loss of generality, we write $T_1 < T_2 < \cdots < T_n$ for the *ordered* times when an occurrence of the event is observed, that is, for the jump times of N.

Suppose we partition the time interval $[0, t]$ into a number of small intervals $0 < t_1 < \cdots < t_K = t$, the survival function $S(t)$ can be expressed as

$$S(t) = \prod_{k=1}^{K} S\left(t_k \mid t_{k-1}\right), \tag{4.18}$$

using the multiplication rule for conditional probabilities. Here $S(v \mid u)$, for $v > u$, is the conditional probability that the event will occur later than time v given that it has not yet occurred by time u.

If no event is observed in $(t_{k-1}, t_k]$, we estimate $S(t_k \mid t_{k-1})$ by 1, whereas if an event is observed at time $T_j \in (t_{k-1}, t_k]$, a natural estimate of $S(t_k \mid t_{k-1})$ is

$$1 - \frac{1}{Y(t_{k-1})} = 1 - \frac{1}{Y(T_j)}. \tag{4.19}$$

Then we obtain

$$\hat{S}(t) = \prod_{j:T_j \leq t} \left[1 - \frac{1}{Y(T_j)}\right], \tag{4.20}$$

which is the Kaplan-Meier estimator.

The Cox proportional hazards model investigates the relationship between the death and possible explanatory variables, $\mathbf{x} = (x_1, \ldots, x_d)$. These variables are called the covariates in the survival analysis and also known as the features in the predictive learning.

The Cox model is based on two key assumptions [25, 30, 24]:

(1) the effect of covariates is assumed to be additive and linear on a log-hazard scale;

(2) the ratio of the hazards of two individuals is assumed to be the same at all times.

The hazard function at time t for an individual with covariates $\mathbf{x}$ is assumed to be

$$h(t, \mathbf{x}) = h_0(t)\, e^{\boldsymbol{\beta}'\mathbf{x}}, \tag{4.21}$$

where $h_0(t)$ is the unspecified (nonnegative) baseline hazard function and $\boldsymbol{\beta}$ is a vector of regression coefficients. The coefficients vector $\boldsymbol{\beta}$ is estimated by maximizing the partial likelihood

$$L(\boldsymbol{\beta}) = \prod_{j=1}^{K} \left[\frac{\exp(\boldsymbol{\beta}'\mathbf{x}_j)}{\sum_{l \in \mathcal{R}_j} \exp(\boldsymbol{\beta}'\mathbf{x}_l)}\right], \tag{4.22}$$

or the partial log likelihood

$$l(\boldsymbol{\beta}) = \sum_{j=1}^{K} \log \left[\frac{\exp(\boldsymbol{\beta}'\mathbf{x}_j)}{\sum_{l \in \mathcal{R}_j} \exp(\boldsymbol{\beta}'\mathbf{x}_l)}\right], \tag{4.23}$$

where $\mathcal{R}_j$ represents all subjects at risk at the jth failure time and $K \leq n$ is the number of distinct failure times.

The details for estimating $h_0(t)$ are described in [24, 25]. However, we provide a simple interpretation here. The baseline hazard function $h_0(t)$ can be considered as the proportion of at risk subjects that fail at t_i. Precisely, we have

$$h_0(t_i) \propto \frac{d_i}{r_i}, \tag{4.24}$$

where d_i is the number of subjects fail at t_i and r_i is the number of subjects at risk. Hence, the estimates of $h_0(t)$ will be limited to $h_0(T_1), h_0(T_2), \ldots, h_0(T_K)$, and we normally assume $h_0(t)$ remain unchanged for $T_i < t < T_{i+1}$.

Combining (4.11) and (4.21), we have the following survival function based on the Cox model,

$$S(t, \mathbf{x}) = \exp\left(-\int_0^t h(u)\,\mathrm{d}u\right) = \exp\left(-\int_0^t h_0(u)\,e^{\boldsymbol{\beta}'\mathbf{x}}\,\mathrm{d}u\right). \tag{4.25}$$

Figure 4.3 shows the examples of estimated survival function. Specifically, the survival functions for six individuals, $S(t, \mathbf{x}_1), \ldots, S(t, \mathbf{x}_6)$, are plotted from day 0 to day 280.

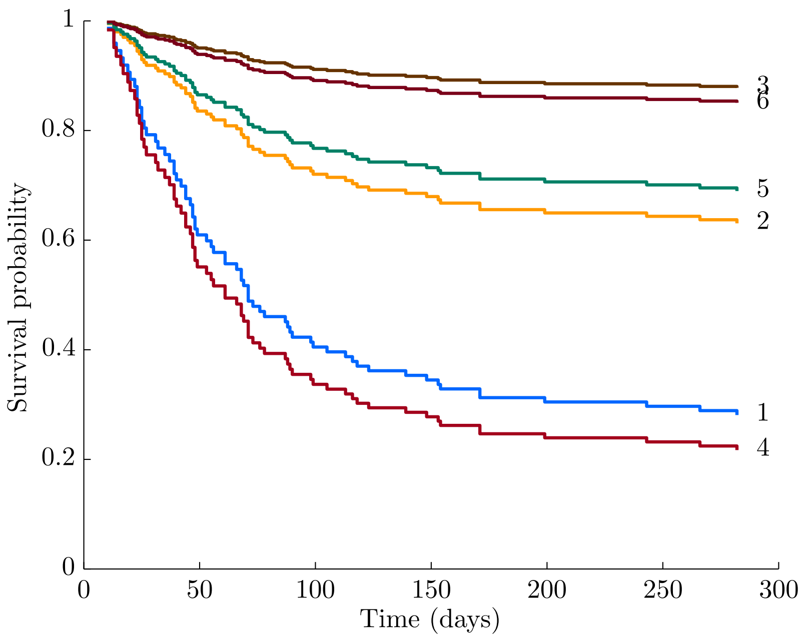


FIGURE 4.3. Estimated survival functions based on Cox model.

It should be noted that the classical survival analysis is not designed to predict events. It is primarily used to estimate a survival curve which measures the probability that an event has not occurred by time t, taking all previous events and censored observations into account.

4.3. Problem Formalization

In many applications, the goal is to predict survival at a pre-specified time point, denoted as τ. This time point is called the *prediction time* in this chapter. The prediction time normally varies from one study to another. For example, in a study about acute cancer, prediction time τ could be three or six months after initial diagnosis. Alternatively, for the study of a chronic disease, the value of τ can be five years. For the purpose of this chapter, we assume that the value of prediction time is specified and known based on medical knowledge. Next, we describe a possible formalization of this survival prediction problem, leading to a binary classification formulation.

Our goal is to estimate a classifier $f(\mathbf{x})$ that predicts a subject's status at prediction time τ based on the input (or covariates) $\mathbf{x}$, given training (past) survival data

$$(\mathbf{x}_1, U_1, \delta_1, y_1), \ldots, (\mathbf{x}_n, U_n, \delta_n, y_n), \tag{4.26}$$

where $\mathbf{x}_i \in \mathbf{R}^d$, $U_i \in \mathbf{R}_+$, $\delta_i \in \{0, 1\}$, and $y_i \in \{+1, -1\}$. Note that the observed survival time U_i and event indicator δ_i will only be available for training, but not for prediction (or testing stage). The training data in (4.26) include the class labels (subjects' statuses) y_i which do not appear in the survival data triplet (4.3).

The status of a subject at prediction time τ can be encoded as a binary class label via the following:

$$y = \begin{cases} +1, & \text{if } U < \tau, \\ -1, & \text{if } U \geq \tau, \end{cases} \tag{4.27}$$

This encoding scheme is illustrated in Figure 4.4 using the observations of a hypothetical study in Figure 4.2. This encoding naturally assigns class labels to exact observations ($\delta = 1$, such as subjects 2 and 6 in Figure 4.4) without ambiguity. As for the censored observations ($\delta = 0$), if $U > \tau$ such as subject 5, we can still safely assign the class label

$y_5 = -1$. Nonetheless, the class labels for subjects 1, 3, and 4 are not well-defined as the true survival times can be longer than τ. Clearly, this uncertainty in class labels for censored training samples does not appear in standard machine learning classification problems. So the challenge is to develop new classification formulations that incorporate the survival time information and the uncertain nature of censored data.

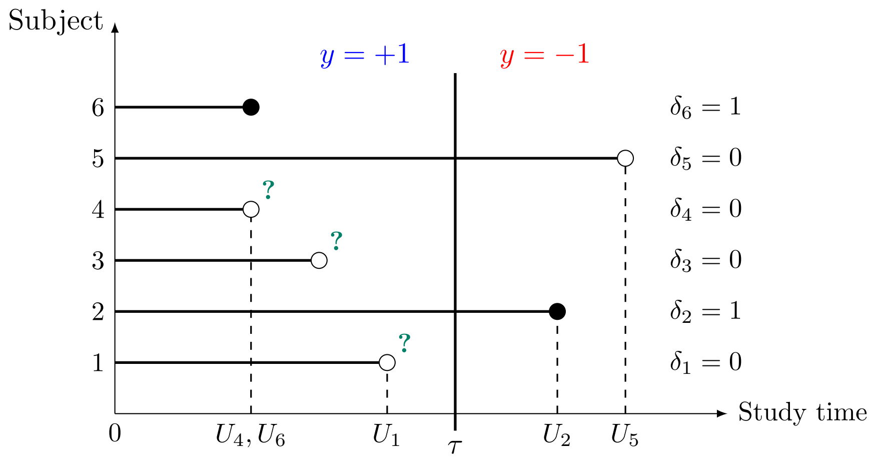


FIGURE 4.4. Example of survival data under the predictive problem setting. The goal is to find a model $f(\mathbf{x})$ that predicts a subject's status at pre-specified prediction time τ. The "?" marks indicate the corresponding class labels are not well-defined.

4.4. Modeling of Survival Data

The goal of modeling survival data is to estimate a classifier that predicts a subject's status at some pre-specified prediction time τ. One classical approach in statistics estimates a survival function which provides the probability of survival as a function of time. Then a simple thresholding rule can be used to determine the binary status at time τ. As for machine learning approaches, there are three strategies for modeling survival data $(\mathbf{x}, U, \delta, y)$ under the classification setting. The first one learns with $(\mathbf{x}, y)$ only. The second incorporates both U and δ as privileged information into learning, whereas the third utilizes only U as privileged information.

4.4.1. Classical Approaches

Classical approach in statistics for modeling survival data aims at estimating a survival function $S(t)$, which is the probability that the time of death is greater than a certain time t, or $\Pr(T > t)$. More generally, the goal is to estimate $S(t, \mathbf{x})$, a survival function conditioned on subject's characteristics, where the characteristics are denoted as feature vector $\mathbf{x}$.

Assuming that the probabilistic model $S(t, \mathbf{x})$ is known, or can be accurately estimated from available data, this model provides a complete statistical characterization of the data. In particular, it can be used for prediction and for explanation (e.g., identifying input features that are strongly associated with an outcome, such as death).

The Cox modeling approach has been widely used to construct the survival function $S(t, \mathbf{x})$, based on the proportional hazards model. We propose using Cox model under the predictive setting via a simple thresholding rule at time $t = \tau$:

$$\hat{y}_i = \begin{cases} +1, & \text{if } S(\tau, \mathbf{x}_i) < r, \\ -1, & \text{if } S(\tau, \mathbf{x}_i) \geq r, \end{cases} \tag{4.28}$$

where the threshold value r should reflect the misclassification costs given *a priori*. We assume *equal* misclassification costs in this chapter, so the threshold level is set to $r = 0.5$.

This thresholding method is demonstrated in Figure 4.5 using the examples of survival function in Figure 4.3. Setting the threshold level to $r = 0.5$ gives the predictions $\hat{y}_1 = \hat{y}_4 = +1$, and $\hat{y}_2 = \hat{y}_3 = \hat{y}_5 = \hat{y}_6 = -1$.

4.4.2. SVM-Based Approaches

There are three possible strategies for modeling survival data $(\mathbf{x}, U, \delta, y)$ under the classification setting. The first one is learning using $(\mathbf{x}, y)$ while ignoring survival time U and event indicator δ. The second strategy incorporates both U and δ (known for training samples) as privileged information into learning. The third is similar to the second, but utilizes only U as privileged information. Detailed descriptions of these strategies are provided next.

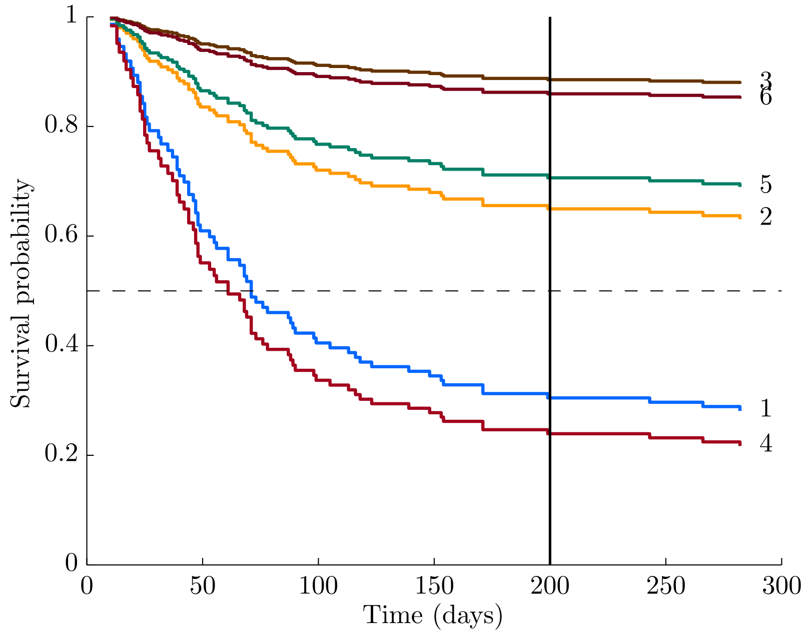


FIGURE 4.5. Using Cox model in the predictive setting. Thresholding each survival function (probability) at day 200 gives the predictions for statuses. The threshold level is indicated with a dashed line at $r = 0.5$.

(1) The simplest way to model the survival data $(\mathbf{x}, U, \delta, y)$ under standard classification setting is to ignore both the survival time U and event indicator δ. Then the *simplified* survival data, in the form of $(\mathbf{x}, y)$, can be modeled via standard SVM with binary classification. This approach is used in our empirical comparisons presented later in Section 4.5 (under the name SVM linear or SVM rbf). It may yield sub-optimal models, as we ignore the information about survival time and event indicator. This SVM approach along with Cox modeling will be used in Sections 4.5 as baseline performance indices.

(2) The second strategy incorporates both the observed survival time and event indicator as privileged information into a classification formulation, which leads to the SVM+ classifier. Before applying SVM+ to survival data, we need to assign a *certainty measure*, denoted as c, to reflect and quantify the uncertain nature of censored data (i.e., training samples with $\delta = 0$). One rule is to set the certainty of a subject being alive/dead at prediction time τ to be inversely proportional to the (known) survival time. That is, certainty measure is defined

as

$$c = \frac{\tau - U}{\tau} = 1 - \frac{U}{\tau}. \tag{4.29}$$

Thus, if U is small, it is more likely this subject will *not* survive at time τ, and c is close to 1. On the other hand, if U is very close to τ, this subject will be either alive or dead at time τ with low certainty, as quantified by lower c-value. We reproduce Figure 4.4 and indicate the certainty measure for each subject in Figure 4.6. Note that we have $c_5 = 1$ for the censored subject 5 with $U_5 > \tau$. Even though subject 5 is censored, there is no ambiguity in its status at time τ. Naturally, we also have $c = 1$ for all exact observations, such as subjects 2 and 6 in Figure 4.6.

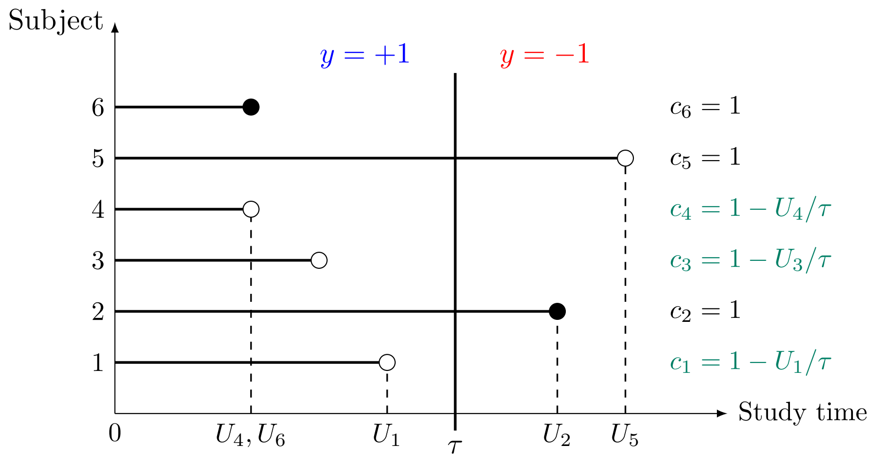


FIGURE 4.6. The certainty measures for exact and censored observations. For censored observations with $U < \tau$, the certainty measure is $c = 1 - U/\tau$. The certainty measure is equal to 1 for exact observations and censored observations with $U > \tau$.

Therefore, the survival data $(\mathbf{x}_i, U_i, \delta_i, y_i)$, $i = 1, \ldots, n$, will be represented as

$$(\mathbf{x}_i, \tau - U_1, c_1, y_1), \ldots, (\mathbf{x}_n, \tau - U_n, c_n, y_n), \tag{4.30}$$

for $i = 1, \ldots, n$. In addition, the survival time and certainty measure can be regarded as the privileged information under the SVM+/LUPI paradigm introduced in Section 3.2. Specifically, the available survival data $(\mathbf{x}, \tau - U, c, y)$ can be considered as $(\mathbf{x}, \mathbf{x}^*, y)$, where $\mathbf{x}^* = (\tau - U, c)$ is the privileged information.

Hence the problem of modeling survival data can be formalized and modeled using the SVM+ approach.

(3) The third strategy is to utilize only the survival time U, while ignoring the event indicator δ. Then we consider

$$|c| = \frac{|\tau - U|}{\tau} \tag{4.31}$$

as a *confidence measure* for a subject's status at time τ, as shown in Figure 4.7.

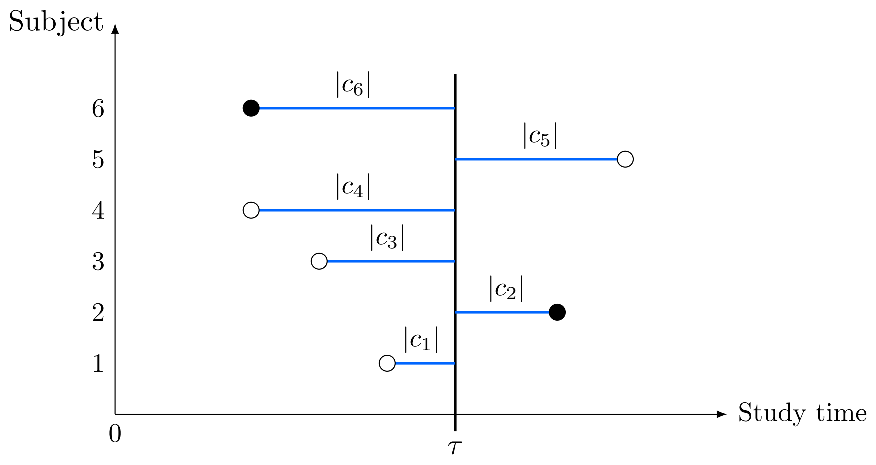


FIGURE 4.7. The univariate privileged information x^* for LO-SVM is defined as $|c| = |\tau - U|/\tau$ for modeling the survival data. While ignoring the censoring information, $|c|$ is considered as a confidence measure.

For all exact observations, the interpretation of $|c|$ is straightforward. However, for a censored observation, such as subject 3, $|c_3|$ should be viewed as an upper bound on the confidence measure. In other words, our confidence level in the status of subject 3 is *at most* $|c_3|$. Similarly, $|c_5|$ is a lower bound for subject 5, and our confidence level in the status of subject 5 is *at least* $|c_5|$.

By ignoring the event indicator δ, we can translate the survival data $(\mathbf{x}, U, y)$ into $(\mathbf{x}, |c|, y)$. Then $x^* = |c|$ is a univariate privileged information for the LO-SVM introduced in Section 3.3. Particularly, the LO-SVM exploits the ordering of training samples near the prediction time τ, e.g., samples with small $|c|$ values. On the other hand, training samples with large $|c|$ values have only minor effect on the solution of LO-SVM.

4.4.3. Remarks

The differences between three proposed SVM-based approaches for modeling survival data are summarized next:

(1) All three modeling approaches (namely, SVM, SVM+, and LO-SVM) estimate a binary classifier $f(\mathbf{x})$, and all these approaches use equal misclassification costs (for uniform comparison in Section 4.5).

(2) For SVM, the observed survival time U and even indicator δ are not used. Still, according to (4.27), the survival time information is partially reflected in the class labels for training data.

(3) The definitions of certainty measure in SVM+ and confidence measure in LO-SVM are similar. The certainty measure is defined for censored observations with observed survival time shorter than prediction time, i.e., $\delta = 0$ and $U < \tau$, such as subjects 1, 3, and 4 in Figure 4.4 or 4.6. However, the confidence measure is defined for *all* subjects as illustrated in Figure 4.7.

(4) The LO-SVM effectively utilizes only the time information in estimating $f(\mathbf{x})$.

(5) The SVM+ approach can incorporate *different types* of privileged information. That is, it can use *only* the survival time (just like the LO-SVM), or use *both* survival time and certainty measure as the privileged information. The two ways of utilizing the privileged information with SVM+ are included in our empirical comparisons presented later in Section 4.5, under the names SVM+1 and SVM+2. Precisely, we have $x^* = \tau - U$ for SVM+1 and $\mathbf{x}^* = (\tau - U, c)$ for SVM+2.

(6) In fact, the SVM+ approach can be readily extended to incorporate any additional privileged information besides the censoring and survival time. For example, additional privileged information may include future clinical information (after initial diagnosis). Also refer to the examples introduced in Section 3.1.

4.5. Empirical Comparisons

This section presents empirical comparisons using classical Cox regression, standard SVM, and the proposed LUPI-based approaches for modeling survival data. All comparisons adopt linear parameterization for standard SVM, LO-SVM, and SVM+, because our synthetic data will be generated using a linear model, and also because the Cox regression assumes linear parameterization. Meanwhile, the radial basis function (RBF) kernel is used for the correcting space of SVM+.

4.5.1. Synthetic Datasets

The purpose of empirical comparisons using synthetic data is to understand the relative strengths and limitations of the LUPI-based methods. The synthetic datasets are designed to include various statistical features, such as the number of training samples, the proportion of censoring, and the noise level in the observed survival time. In practice, the prediction time τ should be specified by the application domain experts, e.g., physicians. For simplicity, our comparisons assume equal misclassification costs and balanced data. Hence, the median of the survival times is chosen as the prediction time τ.

The synthetic datasets are generated following standard procedures used in [31, 32]:

(1) Set the dimension of input features d to 30, and generate $\mathbf{x} \in \mathbf{R}^d$. Every element in $\mathbf{x}$ is a random number drawn from a uniform distribution in $[-1, 1]$.

(2) Use the coefficient vector

$$\boldsymbol{\beta} = [1,1,2,3,3,1,1,1,1,1,0,2,0,2,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0]$$

to generate the event time T via exponential distribution $\mathrm{Exp}((\boldsymbol{\beta}, \mathbf{x}) + 2)$, and add the Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the event time T. Additionally, generate the censoring time C via exponential distribution $\mathrm{Exp}(\lambda)$.

(3) Obtain the observed survival time U and event indicator δ based on the definitions in (4.1) and (4.2). The proportion of censoring in the training data is controlled by the rate λ of the exponential distribution for censoring time C.

(4) Use rule (4.27) to assign a class label to each input vector $\mathbf{x}$. The median of the observed survival times is selected as the prediction time τ, such that the prior probability for each class is roughly the same.

(5) Generate 50, 50, and 2000 samples for training, validation, and testing, respectively.

The training data are used for model estimation, validation data for model selection (or parameter tuning), and test data for estimating the prediction error. Each experiment is repeated ten times with different random realizations of (training, validation, test) data, and the averaged test error and its standard deviation are reported.

The synthetic datasets conform to the probability hypothesis (i.e., exponential distribution) of the statistical modeling method. Therefore, the Cox modeling method should be competitive for such synthetic datasets.

We should point out that modeling survival data using the proposed classification settings (as describe in Section 4.3) typically results in highly nonseparable classification problems. This intrinsic nonseparability can be conveniently demonstrated by using the *histogram-of-projections* technique for visual representation of the estimated SVM model [**3, 4, 33**]. This technique displays the empirical distribution of distances between (high-dimensional) samples and the SVM decision boundary (of a trained SVM classifier).

A typical histograms of projections for synthetic data are shown in Figure 4.8 for different additive noise levels. The empirical distribution is represented in the form of a univariate histogram of distances for training samples (or test samples), along with SVM decision boundary (marked as "0" distance on x-axis) and SVM margin borders for negative and positive classes (marked as "$-1/+1$"). Further, the x-axis of a histogram represents a scaled distance between a high-dimensional feature vector and SVM decision boundary. The y-axis represents the fraction of samples. The distance to the decision boundary is scaled so that the margin borders always have values -1 or $+1$.

Both histograms show a significant portion of overlap between the positive (blue) and negative (red) classes, implying that high test error rates can be expected, especially for

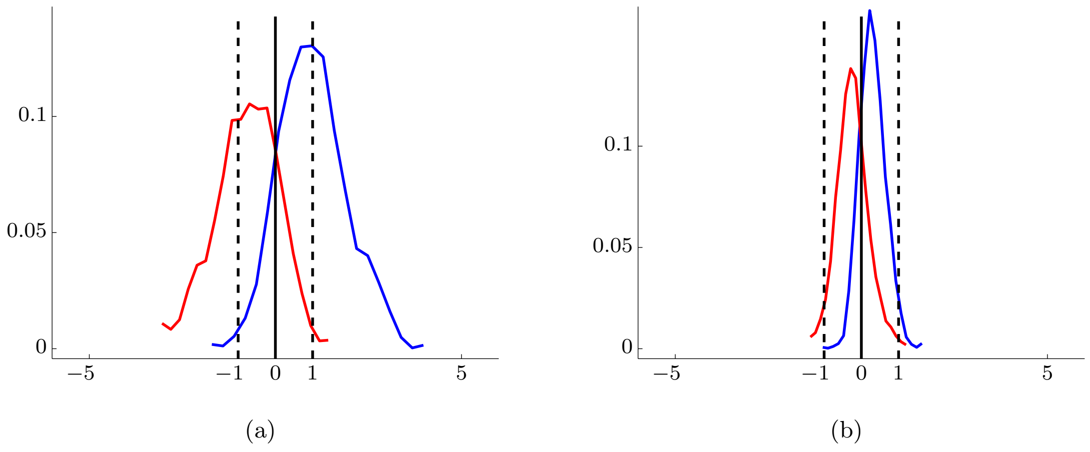


Figure 4.8. Histograms of projections for training data with (a) $\sigma = 0$, and (b) $\sigma = 0.1$. The positive class is shown in blue and negative class in red. Margin borders correspond to $-1/+1$ (marked by dashed vertical lines). The x-axis is the distance (scaled by margin size) and y-axis represents the fraction of samples.

$\sigma = 0.1$. Empirical comparisons presented later in this section attempt to illustrate the effect of a single design parameter on methods' prediction capability.

- *Number of Training Samples*

  To examine how the training sample size affects the prediction capability (e.g., test error), we gradually increase the number of training samples from 50 to 500, and increase the size of validation set accordingly. The proportion of censored observations is controlled around 16%, and the standard deviation of noise σ is set to 0.1. The relative performance of the methods are summarized in Tables 4.1, as a function of sample size. The empirical results shown in Tables 4.1 include the averaged test error and its standard deviation, based on ten random realizations of (training, validation, test) data.

  As expected, the test error decreases for all methods when the number of training samples is increased. Considering the baseline methods, the Cox model and standard SVM have similar performance when the number of training samples is less than 200. But, for large sample sizes, standard SVM performs better than Cox.

The LO-SVM outperforms the Cox model for all sample sizes. It also performs better than the standard SVM for training sample size less than 250. The LO-SVM and standard SVM have similar test errors for sizes larger than 250. This could be explained by noting that for large samples sizes, the relative effect of using privileged information decreases.

Table 4.1. Test errors as a function of training sample size

(a) Training sample size less than 125

Sample size	50	75	100	125
Censoring %	17.80	17.20	16.20	16.16
Cox	31.9 ± 3.1	27.7 ± 3.0	26.4 ± 2.5	26.0 ± 2.7
SVM linear	30.5 ± 2.3	28.6 ± 3.0	26.9 ± 1.9	26.4 ± 2.4
LO-SVM	29.5 ± 2.8	26.4 ± 2.0	25.1 ± 1.6	24.9 ± 1.7
SVM+1	31.8 ± 4.4	30.6 ± 4.5	27.7 ± 2.2	26.4 ± 2.0
SVM+2	31.2 ± 3.7	29.3 ± 4.1	27.1 ± 3.4	25.8 ± 2.0

(b) Training sample size greater than 200

Sample size	200	250	400	500
Censoring %	15.70	15.80	16.65	16.22
Cox	23.9 ± 1.8	22.4 ± 2.0	21.9 ± 1.6	22.0 ± 1.8
SVM linear	22.3 ± 1.3	20.8 ± 1.5	19.9 ± 1.5	19.1 ± 1.3
LO-SVM	20.0 ± 0.8	20.8 ± 2.0	19.6 ± 1.4	19.1 ± 1.3
SVM+1	22.7 ± 2.4	21.5 ± 2.0	20.1 ± 0.9	19.8 ± 1.4
SVM+2	21.1 ± 1.9	19.7 ± 1.4	18.8 ± 1.3	18.2 ± 1.1

Further, LO-SVM has lower test error than SVM+1 regardless of sample size. These results clearly demonstrate the advantage of using univariate privileged information via ordering scheme, rather than mapping it onto an one-dimensional (1-D) correcting space.

However, when 2-D privileged information is used, then SVM+2 has the best performance for training samples larger than 250. This can be explained by the fact that SVM+2 has more tuning parameters than LO-SVM. Thus, a larger amount of training data is required for SVM+2 in order to estimate a good classifier.

It worth noting that the standard deviations of test errors for SVM+1 and SVM+2 are larger than that for LO-SVM when the training sample size is less than 125. This hints that the model selection task for SVM+ is quite challenging, and the estimated models are unstable.

- *Proportion of Censoring*

In order to investigate the effect of censoring on prediction capability, we adjust the proportion of censoring in the training data, ranging from 0% to 39%. The proportion of censoring (or censoring rate) is controlled via the λ parameter of a exponential distribution. The training and validation sample sizes are 200, and the standard deviation of noise is 0.1 for all experiments. Table 4.2 summarizes the experimental results as a function of censoring rate.

TABLE 4.2. Test errors as a function of censoring rate

λ	0	0.05	0.2	0.6	1.2
Censoring %	0.0	11.1	19.8	29.5	38.7
Cox	22.5 ± 1.4	23.0 ± 1.9	25.0 ± 1.4	28.3 ± 2.4	32.3 ± 1.1
SVM linear	21.9 ± 2.5	22.2 ± 2.2	23.6 ± 2.4	26.7 ± 1.7	30.3 ± 1.2
LO-SVM	20.0 ± 1.9	21.0 ± 1.8	22.1 ± 1.4	26.8 ± 2.6	30.9 ± 1.8
SVM+1	22.3 ± 1.9	22.9 ± 1.8	24.3 ± 1.8	28.1 ± 2.0	31.1 ± 1.4
SVM+2	21.9 ± 2.1	21.3 ± 1.5	22.3 ± 1.4	25.9 ± 1.5	29.0 ± 0.9

Comparing the two baseline methods, standard SVM performs better than the Cox model in all censoring rate. Performance comparison between the LO-SVM and the two baseline methods shows that the LO-SVM has lower test error than the Cox model in all censoring rates, and standard SVM for censoring less than 20% (or $\lambda < 0.2$). Note that, for small or zero censoring rate (i.e., nearly all samples are exact observations), the survival time offers highly reliable privileged information. The LO-SVM effectively utilizes this information for samples close to decision boundary yielding superior generalization.

On the contrary, for high censoring rates, the survival times tend to be unreliable and inaccurate. Under such scenario, the ordering task for the LO-SVM becomes rather difficult, and its (relative) performance deteriorates.

As for the SVM+ methods, SVM+1 performs better than the Cox model, but cannot match standard SVM. However, SVM+2 works better than both standard SVM and the Cox model. Especially for high censoring rates, SVM+2 method shows superior performance due to its ability to learn from the privileged information in a more flexible correcting space.

- *Noise Level in Survival Time*

We adjust the standard deviation of noise to the survival time ranging from 0 to 0.5, and explore the effect of noise on the prediction capability. The training and validation sample sizes are 200, and the proportion of censored observations is controlled around 16%. Table 4.3 summarizes the test errors for all methods as a function of noise level.

TABLE 4.3. Test errors as a function of noise level in survival time

Noise level σ	0	0.05	0.2	0.4	0.5
Censoring %	15.1	16.8	15.3	16.8	18.4
Cox	11.0 ± 0.8	17.9 ± 1.5	28.8 ± 1.5	34.4 ± 0.8	35.8 ± 1.4
SVM linear	14.7 ± 0.7	17.4 ± 1.5	27.5 ± 1.9	33.7 ± 1.3	34.2 ± 2.2
LO-SVM	13.1 ± 1.4	16.0 ± 1.4	27.5 ± 2.3	33.4 ± 1.5	34.7 ± 1.4
SVM+1	14.5 ± 1.1	17.5 ± 1.7	28.7 ± 2.0	33.9 ± 1.9	34.6 ± 1.8
SVM+2	15.2 ± 1.3	18.0 ± 3.2	25.7 ± 2.0	31.6 ± 1.5	32.4 ± 1.8

Obviously, when the noise level is reduced, the test error of all methods will be reduced. With little or no noise in the survival time, the data samples are generated from a distribution that matches the Cox modeling assumptions exactly. Unsurprisingly, the Cox model yields the lowest test error in the case of zero noise. However, for datasets with large additive noise, all SVM-based methods are superior to the Cox model.

Meanwhile, the LO-SVM performs better than the standard SVM for low noise levels, corresponding to highly reliable privileged information. On the other hand, the survival times with large noise cannot be reliably explained by an ordering scheme, resulting in poor generalization.

Both SVM+1 and SVM+2 cannot match the two baseline methods and LO-SVM for $\sigma < 0.1$. Still, under large noise settings ($\sigma \geq 0.2$), SVM+2 method shows the best performance due to its ability to incorporate privileged information in a 2-D correcting space. Note that a similar pattern of (relative) performance between LO-SVM and SVM+2 has also been observed in Table 4.2.

- *No Censoring*

This experiment examines a special case of survival data, which contain only exact observations. In this case, there is no censored observations and the event indicator becomes noninformative. Effectively, the survival time is the only privileged information in the training data. For this reason, only SVM+1 method is used but not SVM+2. Empirical results are summarized in Table 4.4 showing the test errors as a function of training sample size.

Table 4.4. Test errors as a function of training sample size without censored data

(a) Training sample size less than 125

Sample size	50	75	100	125
Cox	29.2 ± 2.3	26.1 ± 2.4	22.7 ± 3.1	23.2 ± 2.9
SVM linear	29.8 ± 2.5	26.9 ± 2.8	23.7 ± 3.5	22.3 ± 1.6
LO-SVM	27.9 ± 2.7	25.1 ± 2.7	20.5 ± 3.3	20.5 ± 1.1
SVM+1	30.2 ± 3.6	27.7 ± 4.5	25.3 ± 4.1	23.1 ± 1.7

(b) Training sample size greater than 200

Sample size	200	250	400	500
Cox	17.9 ± 1.4	18.9 ± 2.5	18.9 ± 1.1	17.1 ± 2.0
SVM linear	19.0 ± 1.4	18.2 ± 1.7	17.0 ± 0.8	16.5 ± 1.2
LO-SVM	17.1 ± 1.4	16.5 ± 1.2	16.0 ± 1.0	15.4 ± 1.2
SVM+1	18.2 ± 1.5	18.2 ± 1.7	17.4 ± 0.7	16.4 ± 1.1

Considering the results in Table 4.1 and 4.4 for the two baseline methods, the Cox model is only better than standard SVM for training data without censored observations and size no more than 100.

These results also show that the LO-SVM outperforms either standard SVM or the Cox model regardless of training sample size. A survival dataset without censored data is considered as the best scenario for the LO-SVM, mainly because the observed survival times are the true survival times and they can be translated into highly accurate confidence measures for class labels. Therefore, using these reliable confidence measures for ordering would achieve good generalization.

As for SVM+1 method, its performance is not as good as the LO-SVM, and even no better than the two baseline methods. The performance difference between SVM+1 and LO-SVM also indicates that the training errors (or slack variables) cannot be adequately modeled with univariate privileged information with nonlinear mapping in the correcting space. Hence, the ordering scheme in LO-SVM is a better choice for utilizing the univariate privileged information.

4.5.2. Real-Life Datasets

This section describes the empirical comparisons for three real-life datasets in the *Survival* package of R [**34**]. In all comparisons, both SVM+1 (1-D privileged information) and SVM+2 (2-D privileged information) use linear kernel for the decision space and RBF kernel for the correcting space. For standard SVM and LO-SVM, both linear and RBF kernels are studied. For all experiments, the median of the observed survival times is chosen as the prediction time τ such that the classification dataset is balanced.

Parameters for SVM-based methods are tuned via a resampling technique. Our experimental setup employs a double resampling procedure [**3**]. The estimation of the test error for a learning method is performed in the first level of resampling, whereas the tuning of model parameters takes place in the second level. Within each level of the double resampling procedure, a five-fold cross-validation is used [**3**].

Since there is no well-defined class labels for censored observations with $U < \tau$ as illustrated in Figure 4.4, the test errors are reported for samples with well-defined labels, which include the exact observations and censored observations with $U \geq \tau$. In addition, tuning parameters are chosen based on the validation error for samples with well-defined labels.

Next, we briefly introduce the real-life datasets used for comparisons.

- The *Veteran* dataset was collected in the Veterans' Administration Lung Cancer Study for a randomized trial about two treatment options [34]. There are 137 instances (observations) in this dataset, each with 10 attributes. Less than 7% of the instances are censored. Among the nine censored instances, an observed survival time is less than the prediction time. In other words, only one instance in this dataset corresponds to indeterminate class label.
- The *Lung* dataset studied the survival and daily activities of patients with advanced lung cancer, based on a study by the North Central Cancer Treatment Group (NCCTG) [34]. There are 167 instances in this dataset, each with 8 attributes. Approximately 28% of the instances are censored, and 21 censored instances are associated with indeterminate class labels.
- The *PBC* dataset was collected between 1974 and 1984 by the Mayo Clinic in a trial about primary biliary cirrhosis (PBC) of the liver [34]. There are 258 instances in this dataset, each with 22 attributes. More than 50% of the instances are censored, of which 54 are not associated with well-defined labels.

TABLE 4.5. Summary of the statistical properties of survival datasets

Dataset	Veteran	Lung	PBC
Size	137	167	258
Attributes	10	8	22
τ (days)	80	269	1831
Censored %	6.57	28.14	56.98
Uncertain %	0.7	12.6	20.9

The statistical properties of the three datasets are summarized in Table 4.5. The prediction time τ (in days) is the median of the observed survival times. The proportions of censored observations are shown in the row with "Censored %" label. The percentage of instances with uncertain class labels are listed in the row with "Uncertain %" label.

Table 4.6. Experimental results for survival datasets: linear models

Dataset	Veteran	Lung	PBC
Cox	23.4 ± 4.6	43.3 ± 5.6	34.3 ± 7.1
SVM linear	27.3 ± 5.7	40.8 ± 8.2	32.2 ± 6.4
LO-SVM linear	39.2 ± 10.4	46.7 ± 8.0	23.4 ± 6.3
SVM+1	32.7 ± 5.4	43.3 ± 4.8	21.7 ± 6.9
SVM+2	31.2 ± 9.5	37.5 ± 8.3	19.0 ± 8.3

- The first part of comparisons is focused on methods using the linear kernel, and the experimental results are summarized in Table 4.6, which shows the average and standard deviation of test error estimated via five-fold cross-validation.

  Prediction performance of the baseline methods (Cox and standard SVM) varies across different datasets. While the Cox model has the lowest test error for the *Veteran* dataset, the standard SVM is slight better than the Cox model for two other datasets. This is expected due to the low censoring rate in the *Veteran* dataset.

  Interestingly, the LO-SVM is only better than the standard SVM for the *PBC* dataset, and its performance is poor in a low-censoring scenario. This is not consistent with the conclusions in Sections 4.5.1. Our interpretation is as follows.

  By examining the distribution of the survival times in the *Veteran* dataset, we found out that the majority of the survival times is close to the prediction time $\tau = 80$. Consequently, most of the confidence measure (univariate time privileged information) values $|c|$ are small, making the ordering task for LO-SVM difficult. Hence, a single error in the ordering may potentially lead to multiple errors. The failure of the ordering task in LO-SVM is also reflected in the standard deviation of the test error, which is the largest among all methods.

Further, the LO-SVM is outperformed by SVM+1 and SVM+2, showing its limitation for the three datasets we examined. On the contrary, the SVM+2 shows the best test error for two datasets, suggesting that it is best suitable for survival data with high censoring rate. Additionally, the SVM+2 shows its advantage in utilizing 2-D privileged information for high-dimensional data, as observed in the *PBC* dataset. These conclusions for SVM+2 are consistent with those for the synthetic data in Section 4.5.1.

- In the second part of comparisons, we focus on methods using nonlinear modeling (via RBF kernel). The experimental results are summarized in Table 4.7.

  Both standard SVM and LO-SVM with RBF kernels perform better than their linear counterparts. Moreover, the LO-SVM with RBF kernel has the lowest test error for the *Lung* dataset, which suggests nonlinear nature of the data. Arguably, the performances of SVM+1 and SVM+2 can be improved by using the RBF kernel, but at the expense of additional parameter tuning.

  Finally, for the *Veteran* dataset, standard SVM with linear or RBF kernel performs better than LO-SVM, SVM+1, and SVM+2. This can be explained by the fact that simpler models have better generalization performance even if they do not have the survival time information. In fact, based on our encoding of class labels in (4.27), the survival time information is largely incorporated in the class labels already.

TABLE 4.7. Experimental results for survival datasets: nonlinear models

Dataset	Veteran	Lung	PBC
SVM rbf	25.8 ± 8.8	37.7 ± 7.4	33.0 ± 4.9
LO-SVM rbf	28.1 ± 7.3	34.2 ± 6.2	24.3 ± 5.1
SVM+1	32.7 ± 5.4	43.3 ± 4.8	21.7 ± 6.9
SVM+2	31.2 ± 9.5	37.5 ± 8.3	19.0 ± 8.3

4.6. Summary

This chapter introduces predictive modeling of survival data as a binary classification problem. This approach may be useful when the goal of modeling is to predict patients' condition (alive/dead) at a pre-determined prediction time. For example, the prediction time can be three months after surgery, six months after initial diagnosis, or two years after transplantation. Our modeling approach may help clinicians (and patients) to promote personalized care and strengthen treatment/recovery post diagnosis and surgery.

From the machine learning perspective, our modeling approach implements new formalization for modeling complex biomedical data which may incorporate future, censored, or unknown data, in addition to (traditional) labeled training data. The chapter describes predictive modeling of survival data using the LUPI paradigm. We also present two particular implementations of LUPI for modeling survival data. The first approach called SVM+ incorporates information about survival time and censoring in order to estimate an SVM classifier. The second approach, LO-SVM, utilizes only the survival time information.

In addition, the chapter describes the advantages and limitations of these modeling approaches by using empirical comparisons of several synthetic and real-life datasets. In general, the empirical results show that the proposed LUPI-based methods seem to be very effective (as opposed to classical Cox regression model) when the survival time does not meet the classical probability hypothesis, e.g., the exponential distribution. Since the LUPI-based methods contain privileged information during the training phase, making use of this additional information properly is the key to outperform standard SVM. However, compared to LUPI-based methods, standard SVM is a simpler approach, and it may have better generalization performance for some datasets.

Further, relative performance between two LUPI-based methods is strongly affected by the statistical characteristics of survival data, such as the amount of censored data and additive noise in survival time. In particular, the LO-SVM is very effective when the proportion of censoring in the training data is small, or when the additive noise in the survival time is small. Both situations enable reliable ordering of training samples using survival time information, under this formalization, resulting in better classification models.

Finally, SVM+ formalization has better (relative) performance for survival data with high censoring rate, or with very noisy survival time, due to the fact that privileged information is modeled in the correcting space via nonlinear kernel.

Part 2

Group Learning

Chapter 5

Overview

Sparse high-dimensional data are common in modern machine learning problems where the dimensionality of data samples d is much larger than the training sample size n. Estimation of predictive classification models from high-dimensional data is becoming increasingly important in various applications such as gene micro-array analysis, image based object recognition, functional magnetic resonance imaging (fMRI), etc.

In micro-array data analysis, technologies have been designed to measure the gene expression levels of tens of thousands of genes in a single experiment. However, the sample size in each dataset is typically small ranging from tens to low hundreds due to the high cost of measurements. Similarly, in brain imaging (or fMRI) studies the dimensionality of the input data vector is very high (the number of voxels $d \sim 10000$), but only a few hundred of two-dimensional (2-D) or 3-D images ($n \sim 100$) are available. Such sparse high-dimensional data present new challenges for classification learning methods.

Most approaches for learning with high-dimensional data focus on improving existing inductive methods that try to incorporate a priori knowledge about the optimal model [3, 35, 36]. Common examples include:

(1) clever preprocessing and feature extraction techniques that incorporate application-domain knowledge into the selection of a small number of informative features;
(2) selection of good kernels in SVM methods;
(3) specification of the prior distributions in Bayesian methods.

These techniques have been successfully used in many real-life applications [37].

Among all these approaches, using feature selection techniques to reduce the dimensionality of data is an effective way for solving a problem. There are many potential benefits of feature selection, such as facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and utilization times, defying the curse of dimensionality to improve prediction performance [38].

Feature selection is a broad area of research. There are two main strategies for feature selection using finite training data [4, 39]. The first strategy performs feature ranking/selection prior to learning (model estimation). Such methods are known as *filter* methods. Filter methods select a subset of "informative" features independently of the classifier. Typically, filter methods explore the relation between each feature and the output, rank all the features by a metric (or a statistical score), and select a predefined number of top features. Popular metrics used for feature ranking are Fisher score [40], information gain [38], or conditional mutual information [41], etc.

The second strategy performs feature ranking/selection as a part of learning and provides optimal prediction performance for a given learning method. Hence, optimal feature selection is a part of learning method itself. In this case, feature selection becomes a part of model selection (complexity control) for a given dataset. Such methods are known as *wrapper* methods. For wrapper methods, feature selection can be viewed as an additional complexity parameter of a learning method. This complexity parameter can be tuned using independent validation data, via resampling techniques, or via analytic methods. Most wrapper methods try to find a good subset of features under some measure by searching the space of feature subsets [39]. One simple greedy algorithm, called *backward elimination*, starts with the full set of features, and greedily removes the one that most improves performance, or degrades performance slightly. A similar algorithm, called *forward selection*, starts with an empty set of features, and greedily adds features.

Alternatively, we propose learning with high-dimensional data *without* feature selection. Our approach is to split all features into several groups, then a high-dimensional feature vector $\mathbf{x} \in X$ can be transformed into several lower dimensional ones $\mathbf{x}' \in X'$, where $\mathbf{x} \in \mathbf{R}^d$, $\mathbf{x}' \in \mathbf{R}^{d'}$, and $d' < d$. The process of learning, including model estimating and prediction, will take place in a lower dimensional space X', rather than the original space X.

In order to illustrate the proposed strategy, consider the task of handwritten digit recognition for digits 5 versus 8 in MNIST dataset. Each digit (5 or 8) is a 28×28 pixels image as shown in Figure 5.1, and it can be represented as a real-valued vector of size $28 \times 28 = 784$, i.e., $\mathbf{x} \in \mathbf{R}^{784}$. Within this vector, each of the 784 components (features) represents the pixel intensity in gray scale by a 8-bit integer. This is standard binary classification problem, where digits 5 and 8 are labeled as negative and positive class, respectively.

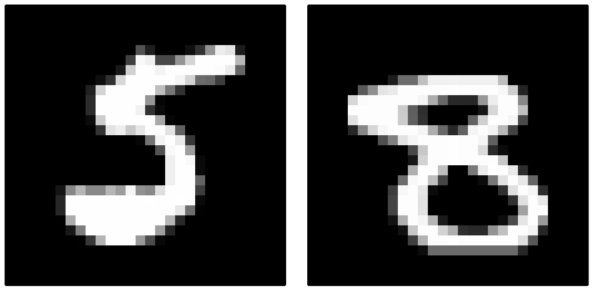


FIGURE 5.1. MNIST digits 5 (negative class) and 8 (positive class) of size 28×28 pixels.

We can partition an image into four nonoverlapping patches, each with size 14×14 pixel, as illustrated in Figures 5.2 and 5.3. Specifically, $\mathbf{x}$ (a training image) is transformed into $\{\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3, \mathbf{x}'_4\}$, where $\mathbf{x}'_j \in \mathbf{R}^{196}$, $j = 1, \ldots, 4$. Then we estimate *one* classifier using feature vectors in space X'. Note that pixels in space X' are more highly correlated than pixels in the original space X as they are close to one another.

This approach allows the reduction in the dimension of feature vector without losing information due to feature selection. Further, the number of training samples can be "increased" without utilizing external training data. The goal of this learning approach is not a good prediction for each $\mathbf{x}'_j$ individually, but an accurate prediction for $\mathbf{x}$. The initial predictions are made for patches $\{\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3, \mathbf{x}'_4\}$, and a final prediction for an (test) image $\mathbf{x}$ is computed. Hence, we need to consider the predictions for patches jointly (as a group), and use a postprocessing procedure to reconcile the four predictions.

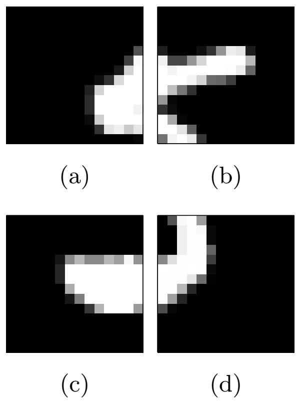


FIGURE 5.2. Partition digit 5 into four nonoverlapping patches of size 14×14 pixels.

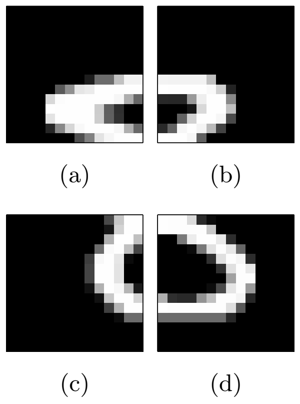


FIGURE 5.3. Partition digit 8 into four nonoverlapping patches of size 14×14 pixels.

We refer to this approach as the *Group Learning* method. The Group Learning method introduced in this thesis is mainly for classification problems, and we will focus on improving the prediction performance.

Further, the proposed Group Learning method is equivalent (or similar) to a Convolutional Neural Network (CNN) [42, 43]. Precisely, the transformation of high-dimensional data corresponds to a *convolutional* layer and the postprocessing procedure is equal to a *pooling* layer of a CNN. Certainly, we only incorporate the two layers *once* in our strategy, whereas the two layers are introduced multiple times in a CNN. Two apparent conveniences for the proposed approach are the lower requirement of computational power and training data.

In summary, we provide a general description of a novel learning paradigm for classification: Group Learning for high-dimensional data. Next, we will present two applications of Group Learning in Chapters 6 and 7. The first one is applying Group Learning method for 2-D images, and the second is for 1-D signals.

Chapter 6

Group Learning for Images

6.1. Introduction

A theoretical framework for predictive learning based on the risk minimization approach is provided by VC-theory. VC-theory makes a strong argument that for finite sample estimation problems one should always use the most appropriate *direct formulation* of the learning problem. This principle can be also applied on the methodological level of formalizing application-domain requirements [**1**, **44**, **3**, **4**]. That is, for a given application, one should first formulate an appropriate learning problem reflecting application domain requirements, and only then develop (or select) learning algorithms for this learning formulation.

In Chapter 5, we introduced a novel learning paradigm for classification called Group Learning for high-dimensional data. This Group Learning approach allows us to reduce the dimensionality of data without losing information due to feature selection, and increase the number of training samples without utilizing external training data simultaneously.

In this chapter, we describe how the problem formalization is closely related to the application domain. Specifically, we demonstrate how Group Learning is formalized and developed for 2-D image problems. Empirical results show that Group Learning method can help improve generalization for high-dimensional data with small training sample size without feature selection.

The remainder of this chapter is organized as follows. Section 6.2 defines the formalization of the Group Learning paradigm. Section 6.3 provides the qualitative analysis on the Group Learning method. The empirical results are presented in Section 6.4 to illustrate the effectiveness of the proposed Group Learning method. Finally, a summary is included in Section 6.5.

6.2. Problem Formalization

In this section, we present the formalization of Group Learning for high-dimensional data (with limited training samples). Suppose every d-dimensional training sample is represented as t samples of dimensionality d/t. Specifically, the training sample $(\mathbf{x}, y)$, where $\mathbf{x} \in X$ and $y \in \{+1, -1\}$, can be considered as

$$(\mathbf{x}'_1, y), (\mathbf{x}'_2, y), \ldots, (\mathbf{x}'_t, y), \tag{6.1}$$

with $\mathbf{x}'_j \in X'$, $j = 1, \ldots, t$. Here, we have $\mathbf{x}'_j \in \mathbf{R}^{d/t}$, $\mathbf{x} \in \mathbf{R}^d$, and $d/t < d$.

The transformation above is equivalent to splitting d features into t subsets. Technically, it can be done by first ordering the features based on application domain knowledge (a *priori*). Let us denote the "ordered" features in vector $\mathbf{x}$ as $x_{(1)}, x_{(2)}, \ldots, x_{(d)}$. Then vector $\mathbf{x}'_j$ can be formed by taking each t-th feature. That is,

$$\begin{aligned}
\mathbf{x}'_1 &= (x_{(1)}, x_{(t+1)}, \ldots), \\
\mathbf{x}'_2 &= (x_{(2)}, x_{(t+2)}, \ldots), \\
&\vdots \\
\mathbf{x}'_t &= (x_{(t)}, x_{(2t)}, \ldots).
\end{aligned}$$

Hence, a training dataset with size n can be transformed into a set of tn samples. The goal of Group Learning is to estimate a *single* classifier using tn training samples, each with dimensionality d/t. In contrast, standard machine learning approach (e.g., SVM) attempts to estimate a classifier using n training samples, each with dimensionality d.

The training and test tasks for standard machine learning approach are illustrated in Figure 6.1. Under the classification setting, the goal of learning is to estimate a classifier $f(\mathbf{x})$ using labeled training samples. This classifier should give good prediction for future (test) inputs. Both model estimation and prediction (training and test tasks) are performed in space X.

Figure 6.2 illustrates the Group Learning approach for the case of $t = 3$. In this approach, a labeled training sample $\mathbf{x}$ is first transformed into three lower dimensional ones, i.e., $\mathbf{x}'_1$, $\mathbf{x}'_2$, and $\mathbf{x}'_3$. Then the Group Learning approach estimates a classifier $f(\mathbf{x}')$ using all labeled training samples $\mathbf{x}'$ ($3n$ in total). In prediction (test) stage, an unlabeled test sample $\mathbf{x}$ is transformed from space X to space X' as well. Therefore, applying the estimated classifier to all "shorten" test inputs $\mathbf{x}'_j$, $j = 1, \ldots, 3$, will result in three predictions $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$. In order to obtain a prediction for the test sample $\mathbf{x}$, a postprocessing procedure for reconciling $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$ into $\hat{y}$ is required. One common postprocessing procedure is the *majority voting* scheme. This scheme will be employed in our experiments presented later in Section 6.4.

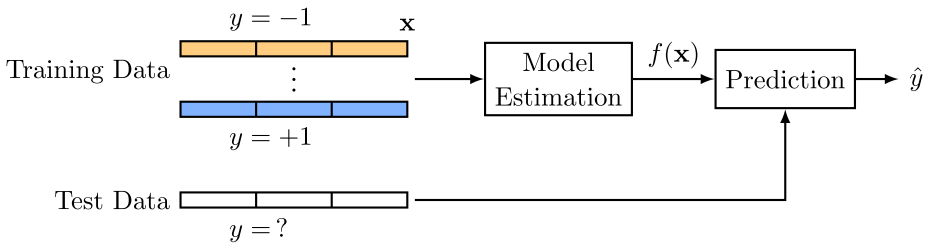


FIGURE 6.1. Standard machine learning approach estimates a classifier using training samples in space X, and makes prediction for test samples in space X.

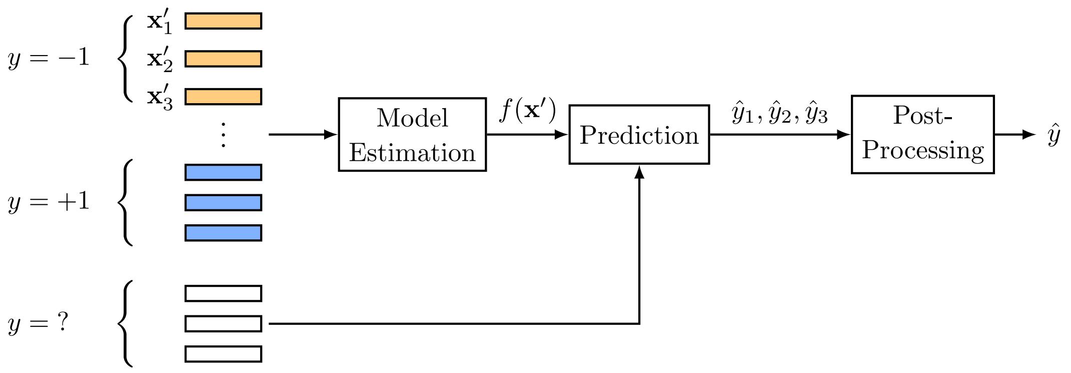


FIGURE 6.2. Group Learning approach first transforms the data from space X to space X'. Both model estimation and prediction are performed in space X', and the predictions need to be reconciled with a postprocessing rule. The illustration is for $t = 3$.

The Group Learning approach is different from standard machine learning approach in two aspects:

(1) First, training and prediction (testing) are performed for data in space X' which has lower dimensionality than space X.

(2) Second, the objective of Group Learning is for accurate prediction for test inputs in space X, rather than the predictions for those in space X' individually. Hence, the role of the postprocessing is to treat the predictions for $\mathbf{x}'_j$ as a group and combine them into one prediction $\hat{y}$.

We motivate the Group Learning approach using the handwritten digit recognition problem and develop the framework. The standard binary classification problem is to differentiate between images of digit 5 and digit 8 (in space X). To reduce the dimensionality of an image, it makes intuitive sense to partition this image into several patches. We can partition an image of size 28×28 pixels into four nonoverlapping patches of size 14×14 pixels, as shown in Figures 5.2 and 5.3. As a result, an image represented by $\mathbf{x} \in \mathbf{R}^{784}$ is transformed into $\{\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3, \mathbf{x}'_4\}$, where $\mathbf{x}'_j \in \mathbf{R}^{196}$, $j = 1, \ldots, 4$.

We observe in Figures 5.2 and 5.3 that none of the patches $\mathbf{x}'_j$ is visually representative to its corresponding digit. Yet, the goal is to learn with patches and make good predictions for *full* images. Therefore, we need to ensure that each patch contains sufficient information to be distinguished between the two digits. Our predictions for $\{\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3, \mathbf{x}'_4\}$ do not have to be perfect. Suppose the majority voting is used as the postprocessing rule, then the final prediction will be based on the "votes" from the *four* predictions for $\{\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3, \mathbf{x}'_4\}$. We can still make a correct prediction for an image as long as there is no more than one prediction error for the four patches. If there were a tie in the votes, the final prediction could be either digit 5 or 8 with *equal* probabilities.

6.3. Qualitative Analysis

In this section, we provide a qualitative analysis on the proposed Group Learning method under the formalization introduced in Section 6.2. This formalization would allow a tighter VC-bound. The VC-bound is a generalization bound for learning with finite samples, and it has been introduced in Section 2.3. We reproduce a complete description here.

According to VC theory, for a binary classification problem, the following bound for generalization ability of a learning method holds with probability of at least $1 - \eta$ for all admissible function $f(\mathbf{x}, \omega)$, including the function $f(\mathbf{x}, \omega^*)$ that minimizes empirical risk:

$$R(\omega) \leq R_{\text{emp}}(\omega) + \Phi\left(\frac{h}{n}, \frac{\log \eta}{n}\right), \tag{6.2}$$

where

$$\Phi\left(\frac{h}{n}, \frac{\log \eta}{n}\right) = \sqrt{\frac{h\left(\log \frac{2n}{h} + 1\right) - \log \frac{\eta}{4}}{n}}. \tag{6.3}$$

Here, h denotes the VC-dimension, n the training sample size. The term Φ is called the confidence interval, since it estimates the difference between the training error $R_{\text{emp}}(\omega)$ and the true test error $R(\omega)$ of a classifier.

Consider the behavior of Φ as a function of sample size n, with all other parameters fixed. Equation (6.3) shows strong dependency of the confidence interval Φ on n/h, the ratio of the number of training samples to the VC-dimension [4]. Thus, we can distinguish two main regimes:

(1) For large sample size, i.e., the ratio is large, the value of the confidence interval Φ becomes small, and the empirical risk can be safely used as a measure of prediction risk. This also implies that a classifier with good generalization is possible.

(2) For small (or finite) sample size, i.e., when the ratio n/h is small, the value of the confidence interval cannot be ignored. In other words, a classifier with small training error $R_{\text{emp}}(\omega)$ does not necessarily lead to good generalization (or small test error).

Most high-dimensional datasets would fall in the second regime. That is, large value of h with small sample size n yield small ratio n/h. Here, we assume h is proportional to d. The precise relation between h and d for SVM classification is given in (2.5). The proposed Group Learning method can increase the sample size while reducing the dimension, and ultimately transform the data modeling problem into the first regime.

The benefits of this strategy are twofold. First, the ordered features are highly correlated, so we are able to reduce the dimension from d to d/t without losing important information. Second, the training sample size "increases" from n to tn, and d decreases to d/t. The ratio of the number of training samples to the VC-dimension is increased from n/d to t^2n/d.

We provide *qualitative* explanation here because the VC-bound assumes independent identically distributed (IID) inputs, but in our setting patches during testing are not really IID.

6.4. Empirical Results

This section presents empirical results to illustrate the effectiveness of the proposed Group Learning method. We use images of handwritten digits 5 and 8 with size 28×28 pixels and partition each image into 4 patches as illustrated in Figures 5.2 and 5.3. The experimental design follows the structures introduced in Figures 6.1 and 6.2 with $t = 4$, and the models are estimated using linear SVM for both learning strategies. Hence, standard linear SVM is the baseline model.

For this experiment, the following settings are used:

(1) image of digit 5: negative class (-1); digit 8: positive class ($+1$);
(2) number of training images: 10 (5 per class);
(3) number of validation images: 20 (10 per class, twice of the training images);
(4) number of test images: 500 (250 per class).

The effective training and validation sample sizes are 40 and 80, respectively. The independent validation set is used for model selection. Specifically, the chosen parameter should yield the lowest error rate for 80 samples in the validation set (or $4m$ patches if there were m validation images). The standard linear SVM classifier will be estimated using 10 training images without partitioning.

Each experiment is repeated 50 times for different random selection of the images, and the average test error is reported. To gain better understanding about the errors, false-positive (FP) and false-negative (FN) error rates for test data are reported separately:

(1) an FP error is defined as incorrect prediction for test digit 5;

(2) an FN error is defined as incorrect prediction for test digit 8.

We also gradually increase the number of training samples from 10 to 50, in order to investigate the prediction capability under various sample sizes. The size of the validation set is adjusted accordingly.

Next we provide experimental results for standard SVM and our Group Learning method using linear kernel. Figure 6.3 shows the training FP and FN error rates as a function of training sample size (per class). Both Group Learning and standard SVM achieve relatively low training errors.

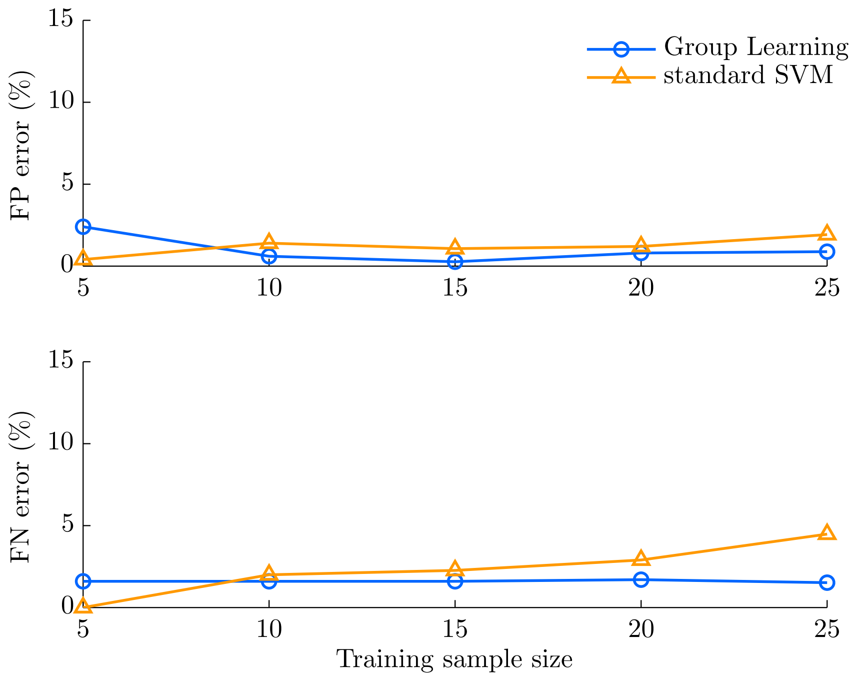


Figure 6.3. The training FP and FN errors as a function of training sample size (per class). The Group Learning method uses 4-patch ($t = 4$) setting and the patch size is 14×14 pixels.

The test errors are for both methods are shown in Figure 6.4. The Group Learning method achieves much lower test errors compared with standard SVM, especially for small training sample size. Interestingly, the training error for Group Learning is higher than that for standard SVM when the training sample size is 5 images per class. However, Group Learning method has significantly lower test error. The results suggest the effectiveness and capability of our Group Learning method in prediction.

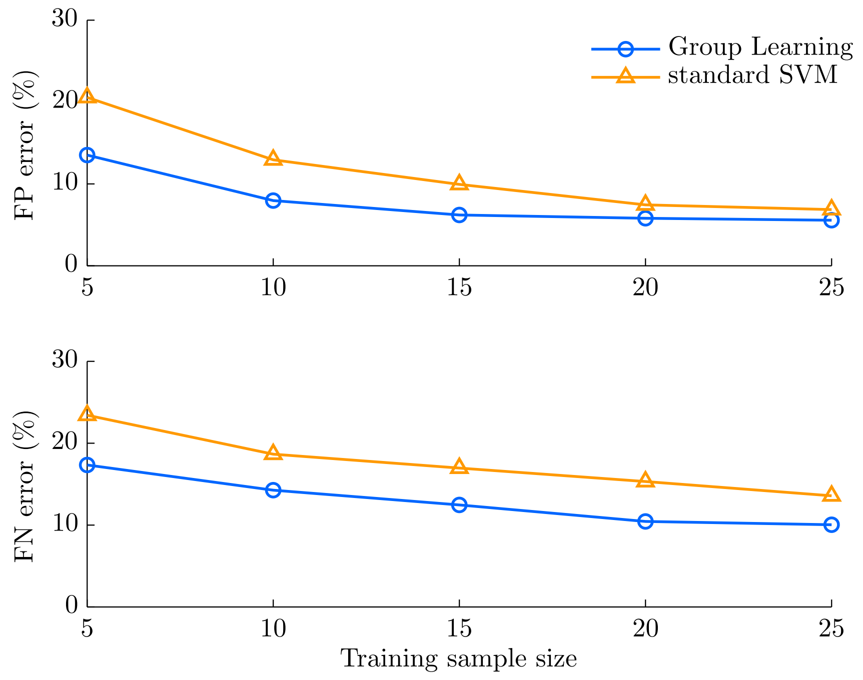


FIGURE 6.4. The test FP and FN errors as a function of training sample size (per class). The Group Learning method uses 4-patch ($t = 4$) setting and the patch size is 14×14 pixels.

To make the experiments more challenging, we down-sample the images to size 10×10 pixel (see Figure 6.5 for exemplary images). Then the experiments described earlier are repeated for the resized digits 5 and 8. The training and test FP/FN error rates for different training sample sizes are shown in Figures 6.6 and 6.7. Similar to the results for 28×28 pixels (high resolution) images, Group Learning method applied to 10×10 pixels (low resolution) images yields higher training error but achieves lower test error when the training sample size is less than 10 images per class. Overall, Group Learning method usually has lower test error than standard SVM regardless the performance during the training stage.

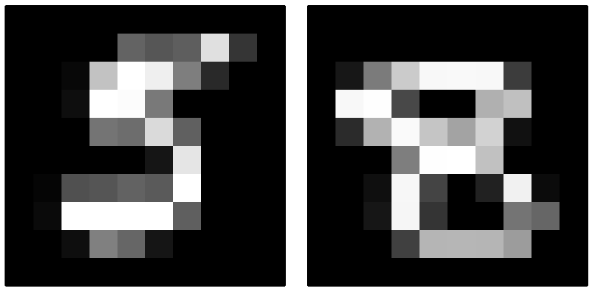


FIGURE 6.5. MNIST digit 5 and 8 of size 10×10 pixels.

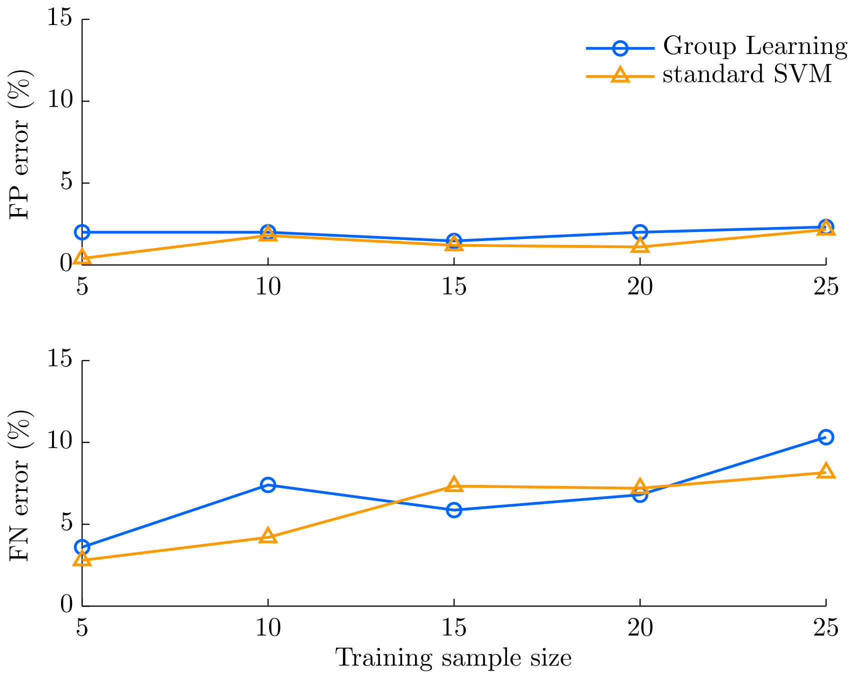


FIGURE 6.6. The training FP and FN errors as a function of training sample size (per class). The Group Learning method uses 4-patch ($t = 4$) setting and the patch size is 10×10 pixels.

6.5. Summary

This chapter introduces and investigates a new learning paradigm: Group Learning for high-dimensional data. Empirical results show improvement in test accuracy for MNIST dataset.

Our Group Learning method first transforms each d-dimensional training/test sample into t training/test samples of dimensionality d/t. The advantages of this step is that

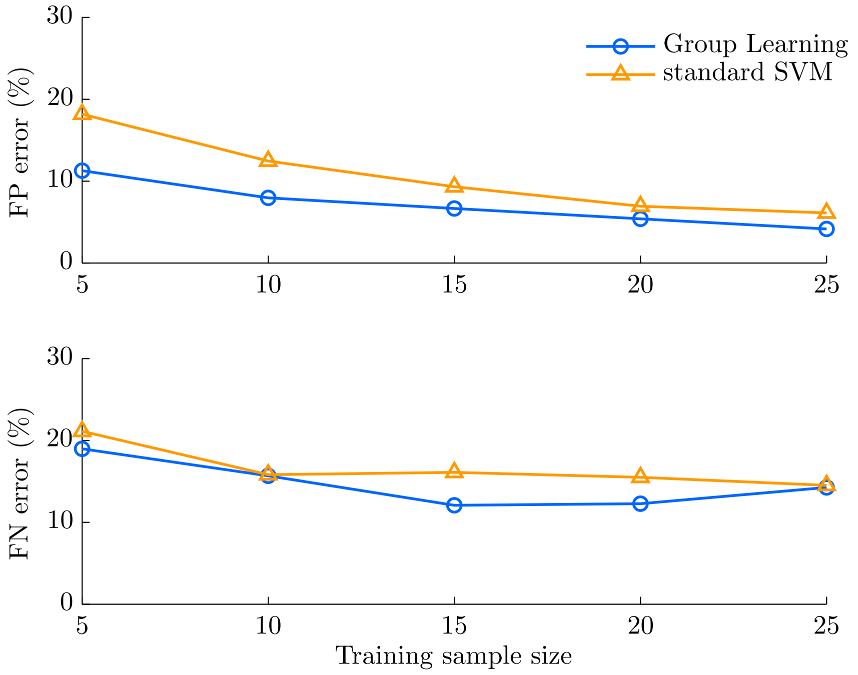


Figure 6.7. The test FP and FN errors as a function of training sample size (per class). The Group Learning method uses 4-patch ($t = 4$) setting and the patch size is 10×10 pixels.

it reduces the dimension of features and increases the number of training samples. The Group Learning method estimates one classifier using all training samples (in lower dimensional space X'). Upon testing, t predictions for t test samples are considered as a group, and combined with a postprocessing rule (majority voting) to obtain a final prediction.

Empirical results in this chapter illustrate the usefulness of the proposed Group Learning method, comparing with standard SVM. In general, the performance of Group Learning method is affected by the level of correlation between features, and the number of groups which contain the subsets of features.

Chapter 7

Group Learning for One-Dimensional Signals

This chapter introduces an application of Group Learning for one-dimensional (1-D) signals. A long duration of 1-D signal can be represented as a group of short nonoverlapping windows, and the task of learning from consecutive windows fits into this framework. Common examples include prediction of epileptic seizures from intracranial electroencephalogram (iEEG), prediction of sudden down turns in the stock market, etc. In this chapter, we will use prediction of epileptic seizures in heavily unbalanced classification setting to demonstrate the formalization of Group Learning and evaluate the performance of this new learning approach.

7.1. Introduction

There is a growing interest in data-analytic modeling for detection and prediction of epileptic seizures from iEEG recording of brain activity [45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55]. Seizure prediction has the potential to transform the management of patients with epilepsy by administering preemptive clinical therapies (such as neuromodulation, drugs) and patient warnings [56].

It is commonly accepted that statistical characteristics of iEEG signal change prior to seizures. However, robust seizure prediction remains a challenging problem, due to the absence of long-term iEEG data recordings containing adequate seizures for training and testing [55] and patient-specific nature of seizure prediction models [45]. Here, we propose a Support Vector Machine (SVM)-based system for seizure prediction, where the design choices and performance metrics are carefully chosen for clinical objectives.

This chapter describes a data-analytic modeling approach for seizure prediction from canine iEEG recordings (dogs with epilepsy). Using canine data are important due to the biological similarity of canine and human seizures, and the availability of high-quality canine iEEG data [46, 57, 58, 59]. Previous research strongly suggests that a successful seizure forecasting should be subject specific [60, 52, 53, 61, 62]. That is, a separate data-analytic model should be estimated for each dog (or for each human subject), using only that dog's past iEEG recordings as training data. The subject-specific or patient-specific nature of data-analytic modeling implies the need for long recordings of iEEG used as labeled training data.

Seizure prediction studies assume that there are three distinct "states" of brain activity in subjects with epilepsy: *interictal*, *preictal*, and *ictal*. The task of seizure forecasting (or prediction) requires discrimination between interictal versus preictal states. This clinical hypothesis can be empirically validated using previously recorded iEEG segments classified (or labeled by a human expert) as interictal or preictal. Using these past labeled data (aka training data), we estimate a data-analytic model for discriminating between interictal and preictal iEEG segments, in order to predict on future inputs (or test inputs). Then, accurate prediction of test inputs (aka out-of-sample data) may be used as the evidence for preictal state.

The task of discriminating between preictal and interictal states is called *seizure prediction* or *seizure forecasting* (from iEEG signal). Thus, we adopt a binary classification setting, where a classifier is estimated using training data, and then its prediction performance is evaluated using out-of-sample test data. The training data represent 1 hr segments obtained from a continuous stream of iEEG recording, and these 1 hr segments are labeled as either preictal or interictal by human experts. That is, *preictal* segments correspond to lead seizures (defined as seizures preceded by a minimum of 4-hour period with no seizures), and *interictal* segments were chosen randomly from iEEG stream (but restricted to be at least one week away from any seizure).

In many studies, the problem of seizure prediction has been formalized as classification of short moving windows of iEEG signal (typically, 20 s long) [46, 47, 48, 49, 50]. This formalization is adopted mainly due to data-analytic reasons, since a single 1 hr segment contains 180 samples (corresponding to 20 s windows). Such a significant increase in the number of training samples makes the classifier estimation/training possible. Still, the goal of prediction is targeted at the 1 hr segments, not the 20 s windows independently. Hence, prediction of 1 hr segments can be formalized via Group Learning introduced in Chapter 5. In contrast to 2-D inputs for Group Learning method presented in Chapter 6, the inputs in this chapter are 1-D.

Additionally, since seizures are very rare events (for most patients and canines), there is clearly an overabundance of available interictal data, but very few preictal data. Therefore, it is common to preselect a ratio of interictal to preictal training data. This asymmetric nature of seizure data is known as an *unbalanced setting* or *unbalanced classification* in data-analytic studies [**3**, 4]. The imbalance ratios used in our study typically range from 8:1 to 10:1 (for different dog-specific models). Note that application of Group Learning presented in Chapter 6 follows a *balanced* setting.

Unbalanced data modeling affects both training and testing stages, as well as the choice of proper performance metrics. For example, wide availability of interictal data implies that classification of interictal segments is intrinsically easier than classification of preictal segments. This consideration may motivate certain modifications of SVM classifiers and may also suggest using appropriate metrics for prediction performance.

The chapter is organized as follows. Section 7.2 presents proper formalization of seizure prediction under predictive classification setting. Section 7.3 describes various design choices for the proposed seizure prediction system, including data representation and feature engineering. Section 7.4 describes the proposed SVM system and the experimental design. The postprocessing steps which are critical for robust prediction performance are presented in Section 7.5. Section 7.6 presents empirical performance evaluation using several canine datasets. Finally, a summary is presented in Section 7.7.

7.2. Problem Formalization

This section presents a statistical formalization for the problem of seizure prediction. The same formalization can be also applied to similar problems involving prediction of rare events (also known as “abnormal” events), such as recurring neurological conditions, prediction of stock market crashes, etc.

The main underlying assumption is that observations of 1-D signals *preceding* the event of interest may contain informative value for prediction. For example, the preictal state could be valuable for seizure prediction. It is natural to view such problems as binary classification where the classifier tries to predict whether a rare event will happen (or not happen) in a given future time period. This time period is known as *prediction horizon* in seizure prediction problems. A formal definition of prediction horizon will be given later in this section.

Formally, each 1 hr training segment can be denoted as $(\mathbf{x}, y)$, where $\mathbf{x} \in X$ and $y \in \{+1, -1\}$. Using the concept of short (20 s) moving windows and following the notation introduced in Section 6.2, $(\mathbf{x}, y)$ can be viewed as $(\mathbf{x}'_1, y), (\mathbf{x}'_2, y), \ldots, (\mathbf{x}'_t, y)$, with $t = 180$. Thus, n training segments can be transformed into a group of $180n$ samples. Here, negative label $y = -1$ corresponds to *normal* (interictal) class, and positive label $y = +1$ corresponds to *abnormal* (preictal) class. Furthermore, the goal is to predict/classify future (unlabeled) segments. We require a prediction for test segment $\mathbf{x}$, but not 180 predictions for $\mathbf{x}'_j$, $j = 1, \ldots, 180$.

The challenging aspects of such “rare-event” predictions are due to several factors:

(1) The number of observed rare events in available data is very small. In other words, for a preictal segment, only a few $\mathbf{x}'_j$ contain warning information about seizure. Almost all $\mathbf{x}'_j$ in an interictal segment do not carry such information.
(2) The unknown distributions for $\mathbf{x}'_j$ from normal and abnormal classes are very similar and highly overlapping.
(3) There also exists high variability in both normal and abnormal class distributions.

Since the training data are highly imbalanced and the class distributions are overlapping, there is always a trade-off between the false-positive (FP) and false-negative (FN) error rates. Therefore, for most applications, the goal is to achieve high or 100% prediction of positive examples (rare events) while keeping the FP rate at a minimal/small/acceptable level.

Although the classification of short moving windows of iEEG signal has been utilized in many studies, it is not clear how accurate prediction of short windows is relevant to the clinical objective of predicting 1 hr segments. In particular, during the operation or test stage, a prediction is usually made for each new moving window. This results in a large number of isolated mispredictions for 20 s windows. Typically, these mispredictions adversely affect the prediction accuracy.

In order to address this problem, several previous studies adopted simple postprocessing, such as three-out-of-five majority voting (over five consecutive predictions for 20 s windows), or a Kalman filter to smooth out the classifier outputs during testing [48]. In the proposed system, we differentiate between the time scales for SVM classification (20 s windows) versus clinical prediction (1 hr segments). Hence, iEEG data are represented in two time scales:

(1) Feature vectors $\mathbf{x}'$ extracted from 20 s windows are used as inputs to SVM classifier.
(2) One hour segments $\mathbf{x}$ (180 consecutive 20 s windows) are used for prediction (or testing stage).

Thus, the prediction of 1 hr segments involves some extensive postprocessing, or majority voting over 180 consecutive predictions for 20 s windows. These postprocessing rules should reflect statistical properties of iEEG signals and also reflect the understanding of SVM classifiers (for unbalanced data), as explained in Section 7.5.

Two additional design considerations important for seizure prediction include the following:

(1) *Preictal period* (PP), or preictal zone, preceding a seizure. The duration of PP is clinically unknown; however, it is implicitly defined by the duration/size of labeled segments in the training data.
(2) *Prediction horizon* (PH) defined as time interval after a seizure prediction/warning is made, within which a leading seizure is expected to occur. The PH is also unknown but it cannot be shorter than the PP. Also note that it is much easier to make predictions with very long PH. For example, one can predict reliably

that the next seizure will occur sometime within the next year, but it is much harder to predict that a seizure will occur within the next 2 hr.

These two design parameters, PP and PH, are clearly important for a successful seizure prediction. Since the inherent variability of seizure prediction should be captured via subject-specific modeling, we select fixed values of PP and PH for all patients/dogs. Specifically, we use 1 hr PP—which effectively assumes that there is a "warning signal" somewhere within 1 hr before a lead seizure. With regard to PH, our modeling approach uses two possibilities (1 hr and 4 hr) during testing (or seizure prediction), reflecting the intrinsic statistical variability of seizure data.

7.3. Feature Engineering

This section describes the available data and our choices of feature representations used in this study [63].

7.3.1. Summary of Available Data

The available data for each dog are continuously recorded from 16 channels of raw iEEG data sampled at 400 Hz. After preprocessing to remove discontinuities and large artifacts, each 1 hr segment of iEEG data is labeled as "interictal" or "preictal" by human experts. Here, a typical canine dataset may contain several leading seizures (1 hr preictal segments) and about eight times more interictal segments.

Table 7.1 summarizes the number of interictal and preictal segments for the six dogs in our analysis. All dogs recorded at least seven seizure episodes (preictal segments corresponding to leading seizures), except for Dog-M3 (having just three leading seizures).

Note that our modeling is performed at several time scales. That is, 20 s windows of iEEG signal are used for classifier training (SVM model estimation), whereas prediction/testing is performed for 1 hr segments (represented as a group of 180 windows). Optionally, SVM system's predictions for four consecutive 1 hr segments may be aggregated to form a prediction for each 4 hr segment.

Table 7.1. The number of interictal/preictal 1 hr segments for each dog

Dog	Interictal	Preictal
L2	152	19
L7	88	11
M3	15	3
P2	64	8
L4	56	7
P1	232	29

7.3.2. Feature Encoding

For data-analytic modeling, each moving window is represented as a set of input features. One common choice is a set of spectral features calculated from the iEEG signals. Standard Delta (0–4 Hz), Theta (4–8 Hz), Alpha (8–12 Hz), Beta (12–30 Hz), and Gamma (30–100 Hz) spectral bands are the most common frequency ranges used, with some studies splitting the Gamma band into 3–4 subbands [46, 48]. Some studies also use additional features such as autoregressive errors, decorrelation time, wavelet coefficients, etc. [47]. These studies have not had the same level of classification accuracy as studies that used only spectral features. Calculation of spectral features requires a predefined time window, with each window resulting in one data sample representing spectral features (for this window). The time window sizes vary from study to study, and the common window size is 20 s (also used in our system). Note that using 20 s windows as training samples for model estimation is also clinically plausible, since seizure warning signals are often manifested as auras that last just a few seconds.

We represent each 1 hr iEEG segment as a group of 20 s nonoverlapping windows. Further, we utilize three approaches to extract features from 20 s windows, as illustrated in Figure 7.1:

(1) The iEEG signal within a 20 s window is first passed through six Butterworth bandpass filters corresponding to six standard Berger frequency bands (0.1–4 Hz, 4–8 Hz, 8–12 Hz, 12–30 Hz, 30–80 Hz, and 80–180 Hz). Then, the output signals from the filters are squared to obtain the estimates of power in six

bands. This procedure is repeated for 16 iEEG channels and yields a feature vector with 96 elements, or $\mathbf{x}' \in \mathbf{R}^{96}$. This feature encoding will be referred to as BFB throughout this chapter. The BFB encoding will be used to explain postprocessing later in Section 7.5.

(2) The frequency spectrum of the iEEG signal is obtained by applying fast Fourier transform (FFT) to each 20 s window. Next, the power in each Berger frequency band is approximated by summing up the magnitudes of the spectrum in the corresponding band. This procedure, indicated as FFT in this chapter, also encodes the spectral contents in 16 iEEG channels as a feature vector $\mathbf{x}' \in \mathbf{R}^{96}$. Note that both BFB and FFT methods perform signal encoding through power estimation. But the former method utilizes signal representation in time domain, whereas the latter in frequency domain.

(3) XCORR calculates the cross-channel correlation of signals from two different channels in order to measure their similarity. Given 16 iEEG channels, there are 120 different pairs. Calculating the cross-channel correlations for all 120 pairs results in a feature vector $\mathbf{x}' \in \mathbf{R}^{120}$.

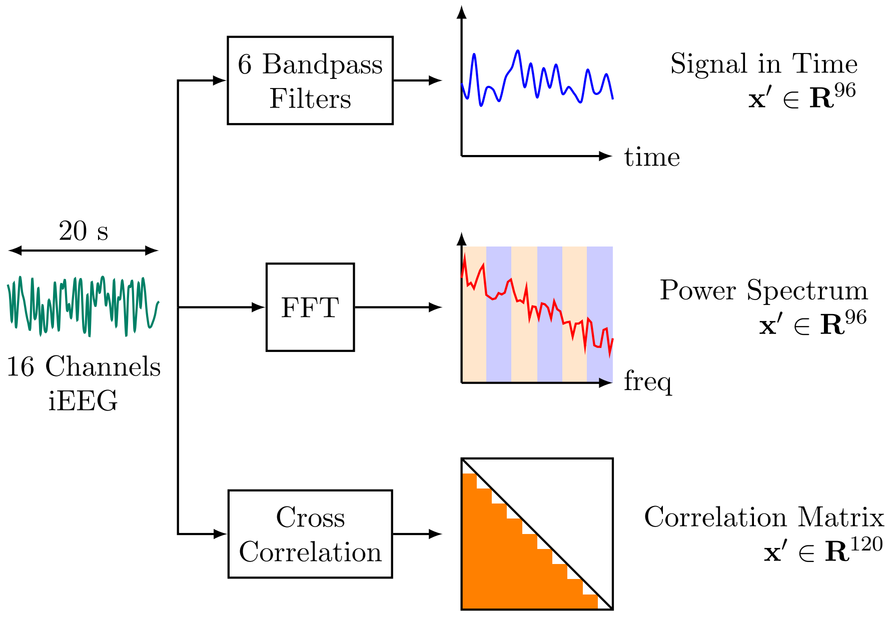

20 s
16 Channels iEEG
6 Bandpass Filters
FFT
Cross Correlation
time
freq
Signal in Time $\mathbf{x}' \in \mathbf{R}^{96}$
Power Spectrum $\mathbf{x}' \in \mathbf{R}^{96}$
Correlation Matrix $\mathbf{x}' \in \mathbf{R}^{120}$

FIGURE 7.1. Three feature encodings for iEEG data: BFB, FFT, and XCORR.

7.4. SVM System and Experimental Design

We describe an SVM-based system for seizure prediction and our experimental design in this section. The available iEEG data include preprocessed 1 hr segments labeled as interictal or preictal. The data-analytic model should predict future (out-of-sample) 1 hr test segments that were never used for model estimation. The proposed system assumes 1 hr PP and 4 hr PH [64].

7.4.1. Proposed System

In our system, an SVM classifier is trained using 20 s labeled windows and then used to predict 1 hr unlabeled test segments, as shown in Figure 7.2 and 7.3.

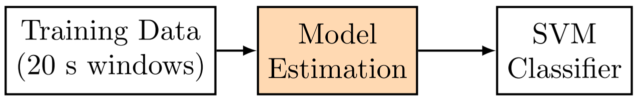


FIGURE 7.2. Proposed system design for training stage.

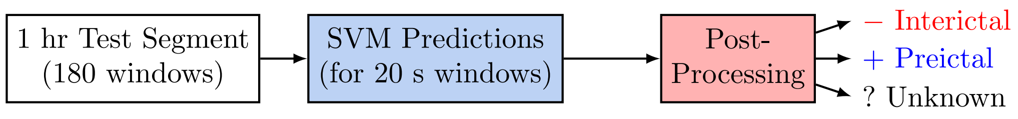

1 hr Test Segment (180 windows)
SVM Predictions (for 20 s windows)
Post-Processing
− Interictal
+ Preictal
? Unknown

FIGURE 7.3. Proposed system design for prediction/operation stage.

Many earlier SVM-based prediction systems used the same implementation for the training stage, i.e., training an SVM classifier using labeled samples corresponding to features extracted from short moving windows [46, 48, 50]. However, all these earlier efforts aimed at achieving good prediction for 20 s windows, according to standard classification setting adopted in machine learning [3, 4].

In contrast, our system aims to make predictions for 1 hr test segments. Hence, during the operation stage shown in Figure 7.3, the system should assign the same class label to all 20 s windows of an 1 hr test segment. This corresponds to the Group Learning approach depicted in Figure 6.2, and it also involves postprocessing explained later in Section 7.5.

The design of our system and the utilization of Group Learning are driven mainly by scarcity and poor quality of preictal data. That is, scarcity refers to very limited amount of preictal data (about 3–11 seizure episodes), and "poor quality" denotes the fact that a "seizure warning signal" may occur somewhere within the 1 hr training segment labeled as "preictal."

The limited amount and poor quality of preictal data contribute to the difficulty of reliable seizure prediction. In our system, these negative factors are partially alleviated by [64]:

(1) Large amount of interictal data, leading to highly imbalanced ratio of interictal versus preictal data (typically, 8:1 to 10:1) during model estimation or training stage.

(2) Proper specification of training (model estimation) and "successful prediction" (or testing). This includes using different time scales for training and operation stages (shown in Figure 7.2 and 7.3), and also additional postprocessing steps critical for robust prediction, as discussed next.

From the clinical perspective, the problem of seizure prediction can be formalized as predictive classification of 1 hr iEEG segments assuming 4 hr PH. Consequently, the training data for model estimation include 1 hr segments labeled as preictal or interictal. The system is designed to predict/classify continuous 1 hr test segments (as preictal or interictal), signaling that a seizure will or would not occur in the next 4-hour period [64]. Hence, the test data consist of 4 hr test segments that should be classified (predicted) as preictal or interictal. The system makes actual predictions for 1 hr test segments, and then combines four predictions in order to predict a 4 hr segment in the following manner:

(1) preictal, if *at least one* of the four consecutive 1 hr test segments is classified as preictal; or

(2) interictal, if *all* four 1 hr test segments are classified as interictals.

Our system's predictions are made in three time scales (20 s, 1 hr, and 4 hr) as shown in Figure 7.4. The system makes predictions for 20 s windows, which are then aggregated into predictions for 1 hr segments. Finally, predictions for four consecutive, nonoverlapping 1 hr test segments are combined into predictions for 4 hr segments.

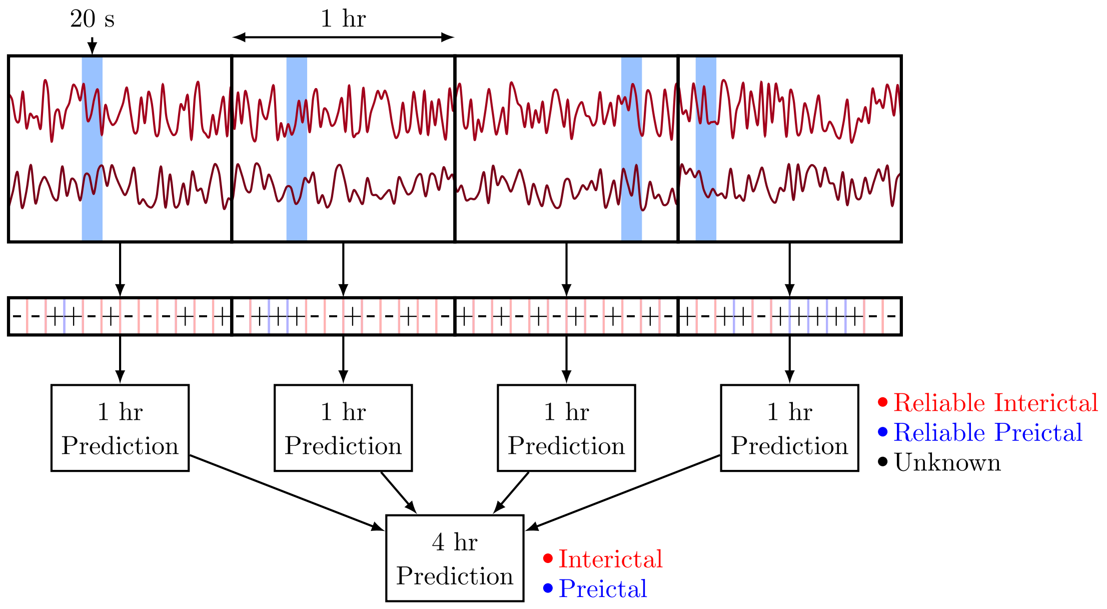


FIGURE 7.4. Predictive modeling in three time scales: 20 s, 1 hr, and 4 hr.

7.4.2. Prediction Performance Indices

The most common prediction performance metrics in machine learning are FP and FN error rates. An FP error corresponds to incorrect prediction for a preictal segment. An FN error is made when a system mispredicts a preictal test segment as interictal.

It is important to note that all performance metrics for seizure prediction are contingent upon the predefined length of PP and PH. Many earlier studies report FP/FN error rates without clearly defined PP and/or PH. Empirical results for our system's prediction performance (shown in Section 7.6) present FP and FN error rates for:

(1) 1 hr test segments, and
(2) 4 hr test segments (formed by combining predictions for four 1 hr segments).

Note that reporting FP and FN error rates is identical to reporting errors for interictal test segments (FP) and for preictal test segments (FN). Further, we also report sensitivity

(SS) and FP rate (FPR) per day, as both are commonly used in seizure prediction research. The two sets of performance indices are in fact equivalent.

7.4.3. Experimental Design

This section describes experimental settings the system shown in Figure 7.2 and 7.3), using the Dog-L4 dataset as an example. This dataset has seven 1 hr preictal segments (correspond to seven recorded leading seizures), and about eight times more interictal segments. The experimental design reflects both the clinical objectives (prediction of 1 hr test segments) and data-analytic constraints (very small number of preictal segments in the training data).

Based on these considerations, we adopted an unbalanced setting for training data (over a balanced one). Under unbalanced setting, the amount of interictal (negative) segments is about eight times more than that of preictal (positive) data. Since Dog-L4 dataset has seven seizures, 6 preictal along with 55 interictal 1 hr segments are used for training (model estimation), and two unlabeled 1 hr segments are used for testing. Under this experimental setting, testing is always performed using out-of-sample data. This unbalanced modeling setup is summarized in Table 7.2, which shows the labels of iEEG segments used for training and testing in each modeling experiment.

Table 7.2. Experimental design for Dog-L4 under the unbalanced setting (The decimal labels encode 1 hr segments)

Experiment	Training set		Test set	
	Interictal	Preictal	Interictal	Preictal
1	2–56	2, 3, 4, 5, 6, 7	1	1
2	1, 3–56	1, 3, 4, 5, 6, 7	2	2
3	1, 2, 4–56	1, 2, 4, 5, 6, 7	3	3
4	1–3, 5–56	1, 2, 3, 5, 6, 7	4	4
5	1–4, 6–56	1, 2, 3, 4, 6, 7	5	5
6	1–5, 7–56	1, 2, 3, 4, 5, 7	6	6
7	1–6, 8–56	1, 2, 3, 4, 5, 6	7	7

According to this experimental setting, we estimate seven different models and each model is tested on its own test set. The final performance index is the prediction accuracy, i.e., the number (or fraction) of accurately predicted test segments in all seven experiments. Reporting prediction accuracy *separately* for interictal and preictal test segments reflects a requirement that a good system should classify each iEEG segment well, rather than many segments over a long observation period. This is because seizures occur very infrequently, so a trivial decision rule "*label every segment as interictal*" will yield quite high prediction accuracy (over long observation period), but it is clinically useless [64].

Further, we discuss details of training the SVM model shown in Figure 7.2. The SVM complexity parameter C is estimated via six-fold cross validation on the training set [3, 4, 64], so that balanced validation data always include samples from one interictal and one preictal segment. A six-fold cross validation is used because Dog-L4 training data have six preictal segments. For other datasets, M-fold cross validation is used if the training data contain M preictal segments. All SVM training and cross validation are performed using *equal* misclassification costs.

There are three important points related to SVM modeling under unbalanced setting [64]:

(1) Linear SVM parameterization is adopted, even though available training data may not be linearly separable. Yet, introducing nonlinear kernels is avoided, as it may result in overfitting, due to high variability of (very limited) preictal training data.

(2) Balanced validation dataset was used for model selection (e.g., tuning C parameter). The decision to use balanced validation data reflects the clinical objective that the system should accurately predict each test segment.

(3) Although SVM training is performed using *equal* misclassification costs, the combination of using unbalanced training data and balanced validation data is formally equivalent to using *unequal* misclassification costs [3].

7.5. Postprocessing

During the test stage shown in Figure 7.3 (or Figure 6.2), the prediction of a 1 hr test segment involves some kind of *postprocessing* or *majority voting* over all 180 windows (comprising this 1 hr test segment). This postprocessing should be related to the properties of binary SVM classifiers, conveniently represented using the *histogram-of-projections* technique for visual representation of the trained SVM model [**3**, **4**, **33**, **64**]. This technique has been introduced in Section 4.5.

A typical histogram of projections of the SVM model estimated using Dog-L4 training data is shown in Figure 7.5. The training data correspond to high-dimensional feature vectors for 20 s windows. As shown in Tables 7.1 and 7.2, the training data include 55 interictal and 6 preictal 1 hr segments, so it is very imbalanced. Note that a small portion of the training interictal segment (in red) falls on the wrong side of the decision boundary, indicating very small error rate (for 20 s windows). A larger portion of the training preictal data (in blue) falls on the wrong side of the SVM model, suggesting higher FN error rate. However, the histogram in Figure 7.5 indicates that interictal (red, negative) and preictal (blue, positive) training samples are generally well separated by the SVM model.

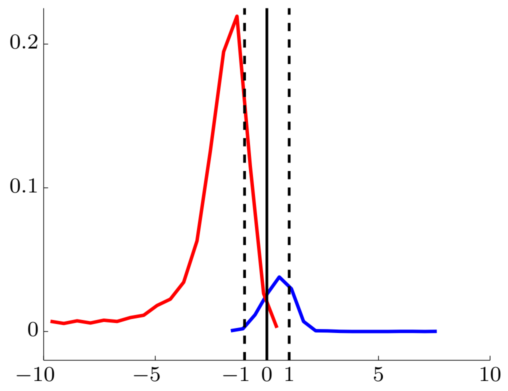


FIGURE 7.5. SVM modeling for Dog-L4 dataset using BFB feature encoding. Histograms of Projections for training data. The preictal data are shown in blue and interictal data in red. Margin borders correspond to $-1/+1$ (marked by dashed vertical lines). The x-axis is the scaled distance and y-axis the fraction of samples.

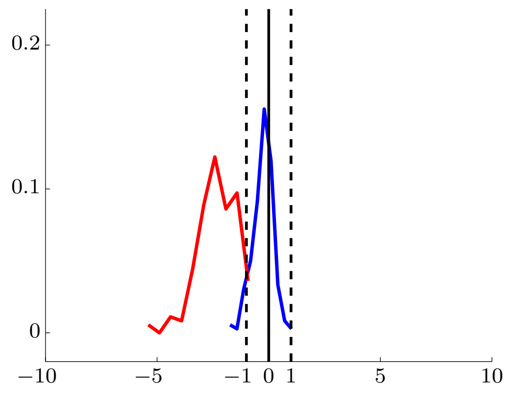
The test data consist of one preictal and one interictal segment, and histograms for such balanced test data are shown in Figure 7.6. The majority of samples for interictal test segment falls on the correct side of the decision boundary "0." However, the histogram for preictal test samples is very unstable, i.e., it can be right skewed or, left skewed with respect to decision boundary, or even may fall within the margin borders, as shown in Figure 7.6a, 7.6b, and 7.6c, respectively. These observations can be used to implement meaningful postprocessing rules for classifying 1 hr test segments, e.g., majority voting over 180 predictions for all 20 s windows comprising 1 hr test segments.

In our system, we adopted the 70% majority threshold [64]. That is, if at least 70% of all SVM predictions for a given 1 hr test segment fall on one side of SVM decision boundary, this segment is classified as *Reliable Interictal* or *Reliable Preictal*; otherwise, it is classified as *Unknown*. As shown in Figure 7.3, our system can make three different predictions. For example, the histograms of the preictal test segments (blue) in Figure 7.6a, 7.6b, and 7.6c will be classified as *reliable interictal* (an error), *reliable preictal*, and *unknown*, respectively. On the other hand, all three interictal test segments (red) in Figure 7.6 will be correctly predicted as *reliable interictal*.

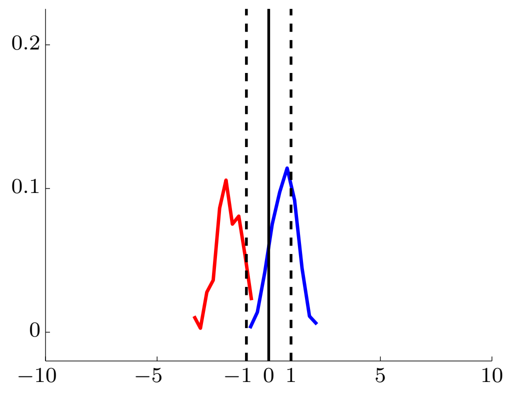
The notion of reliable predictions for 1 hr test segments in our system is quantified as the percentage of test inputs (20 s windows) falling on one side of the decision boundary, as illustrated in Figure 7.7. Three important points about "reliable" predictions should be highlighted:

(1) The reliability of interictal predictions is expected to be higher than that of preictal predictions, since the histograms of projections for training interictal samples are much more stable than those for preictal samples.

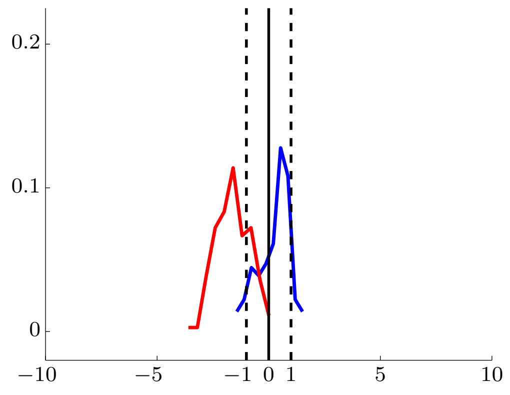
(2) Due to high confidence in interictal predictions (and low confidence in preictal predictions), segments that cannot be predicted reliably as interictal should be regarded as preictal. That is, 1 hr test segments classified as "unknown" in our system (see Figure 7.3) will be always regarded as "preictal," such as the segment (in blue) shown in Figure 7.6c. Hence, the postprocessing decision rules for test segments can be summarized as follows:



(a)



(b)



(c)

FIGURE 7.6. SVM modeling for Dog-L4 dataset using BFB feature encoding. Histograms of Projections for test data. The preictal test segments (blue) will be classified as (a) reliable interictal (an error), (b) reliable preictal, and (c) unknown. All three interictal test segments (red) will be correctly predicted as reliable interictal.

An 1 hr test segment is classified as "interictal" if at least 70% of its 20 s windows are predicted as interictal; otherwise, this segment is classified as "preictal."

(3) The confidence of predictions can be also controlled by the threshold for making prediction decision. In particular, instead of using SVM decision boundary (marked as "0") for classification decision, we can use the margin borders "−1/+1," as illustrated in Figure 7.8. That is, reliable predictions correspond to test input samples falling on the *correct* side of SVM margin borders, whereas predictions falling between the margin borders are regarded as "unreliable." These choices for threshold level have been discussed in [63, 64].

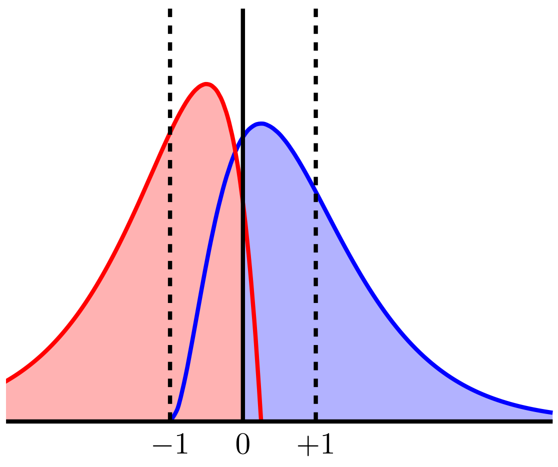


FIGURE 7.7. Univariate histogram of projections for test samples whereas the decision threshold for majority voting is taken relative to decision boundary "0".

7.6. Empirical Evaluation

This section describes prediction performance results for the proposed system using experimental setup outlined in Section 7.4.3. These results illustrate the effect of system's design choices on its prediction performance. These design choices include both preprocessing (e.g., three feature representations) and postprocessing (e.g., making predictions for 1 hr versus 4 hr test segments). As noted earlier in Section 7.4.1, seizure

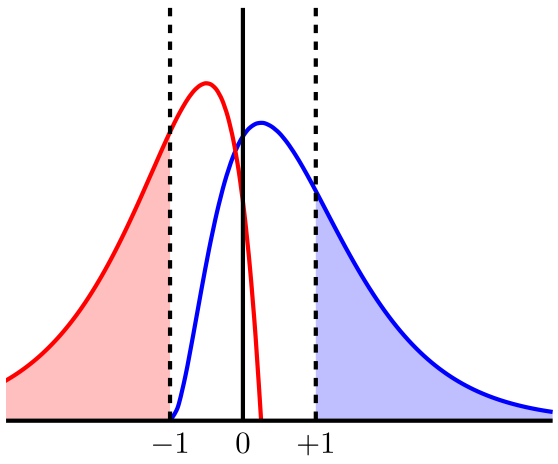


FIGURE 7.8. Univariate histogram of projections for test samples whereas the decision threshold for majority voting is taken relative to margin borders ("−1" or "+1").

prediction using 4 hr PH can be technically implemented by combining SVM predictions for four consecutive 1 hr test segments. That is, a 4 hr PH is modeled via 4 hr test segment, which is classified as preictal only if *at least one* of the four consecutive 1 hr test segments is predicted as preictal.

7.6.1. One-Hour Versus Four-Hour Test Segment

Prediction results for Dog-L4 (using the experimental design shown in Table 7.2) are summarized in Table 7.3. Specifically, Table 7.3 shows the prediction results for test segments under three different feature representations. This table presents predictions for four consecutive 1 hr test segments, treated independently, under "1 hr" column. Combining these 1 hr predictions into a single prediction is shown under "4 hr" column. Symbols `-`, `+`, and `?` denote "reliable interictal," "reliable preictal," and "unknown" predictions, respectively.

These results indicate very good (stable) predictions for interictal test segments, and rather unstable performance for preictal segments. In particular, the patterns of 1 hr predictions for preictal segments vary significantly under three feature encodings. However, most preictal test segments are correctly classified when four 1 hr predictions are combined together. For example, results in Table 7.3 under the FFT feature encoding indicate 100% prediction accuracy for 4 hr preictal segments. Further, the prediction

TABLE 7.3. Predictions for 1 hr and 4 hr segments via different feature encodings for Dog-L4 (Symbols -, +, and ? denote reliable interictal, reliable preictal, and unknown, respectively)

Features	BFB				FFT				XCORR			
Segments	Interictal		Preictal		Interictal		Preictal		Interictal		Preictal	
Exp	1 hr	4 hr	1 hr	4 hr	1 hr	4 hr	1 hr	4 hr	1 hr	4 hr	1 hr	4 hr
1	----	-	-?+?	+	----	-	-+++	+	----	-	-++?	+
2	----	-	-+??	+	--?-	+	++++	+	----	-	?+??	+
3	----	-	??++	+	----	-	?+++	+	----	-	-???	+
4	----	-	++++	+	----	-	++++	+	---?	+	?+++	+
5	----	-	+?+?	+	----	-	???-	+	----	-	-?-?	+
6	----	-	--??	+	----	-	-?++	+	----	-	----	-
7	----	-	----	-	----	-	-++?	+	----	-	-?--	+
Error %	0	0	29	14	4	14	14	0	4	14	39	14

errors for 4 hr preictal segments are smaller than those for 1 hr segments, for all feature representations. This observation underscores the significance of "4 hr PH" aspect in our system, discussed in Section 7.4.1.

Next, we present prediction performance results for several representative datasets, under three feature encodings. All modeling results follow the same methodology as presented in Section 7.4.3 for Dog-L4 dataset. That is, for each dataset we estimate several SVM models, so that the number of experiments equals the number of seizures in the available data. Tables 7.4 and 7.5 summarize the prediction performance results. These results show error rates for 4 hr test segments, obtained by combining predictions for four consecutive 1 hr segments made by the system.

Due to asymmetric nature of the data, we report the FP and FN error rates separately, where FP and FN errors correspond to interictal and preictal errors, respectively. Empirical results in Tables 7.3, 7.4, and 7.5 suggest that no single feature encoding is consistently superior to others. As expected, results in Table 7.4 indicate high FN error

Table 7.4. Summary of prediction performances on FP and FN error rates (%) for 4 hr test segments

Dog	BFB		FFT		XCORR		Combo	
	FP	FN	FP	FN	FP	FN	FP	FN
L2	5	21	10	21	11	42	5	21
L7	9	9	0	27	9	9	0	9
M3	0	33	33	33	0	33	0	33
P2	0	0	0	0	0	25	0	0
L4	0	14	14	0	14	14	0	0
P1	7	17	14	21	7	31	3	10

Table 7.5. Summary of prediction performances on SS (%) and FPR per day for 4 hr test segments

Dog	BFB		FFT		XCORR		Combo	
	SS	FPR	SS	FPR	SS	FPR	SS	FPR
L2	79	0.32	79	0.63	58	0.63	79	0.32
L7	91	0.55	73	0.00	91	0.55	91	0.00
M3	67	0.00	67	2.00	67	0.00	67	0.00
P2	100	0.00	100	0.00	75	0.00	100	0.00
L4	86	0.00	100	0.86	86	0.86	100	0.00
P1	83	0.41	79	0.83	69	0.41	90	0.21

rate and much lower FP error rate. This is due to scarcity and poor quality of preictal training data, as noted in Section 7.4.1.

7.6.2. Combining Predictions

Comparing the predictions for 1 hr test segments under three different feature encodings in Table 7.3 suggests that some errors can be eliminated if the three predictions were combined. For example, an 1 hr interictal segment in Experiment 2 of Dog-L4 is classified as unknown "?" under FFT, but is reliably predicted as interictal under BFB and XCORR encodings, as shown in Table 7.3. Similarly, the last 1 hr interictal segment in Experiment 4 is predicted as "unknown" under XCORR, but is classified correctly under BFB or FFT.

Therefore, we suggest combining the 1 hr segment predictions under BFB, FFT, and XCORR encodings before making the decisions for the 4 hr segments. The combining rule is a simple majority voting (two-out-of-three). The corresponding error rates are shown in Table 7.4 under the "Combo" column. Using this combining rule, both FP and FN error rates are reduced relative to error rates achieved by each feature representation. Equivalently, this combining rule results in improved sensitivity and reduced FPR per day for all datasets, as shown in Table 7.5. Note that using such rule, system's predictions could be better (but never worse) than predictions obtained by each component classifier (using its own feature encoding).

7.7. Summary

This chapter presents the application of Group Learning for seizure prediction using iEEG signal (1-D signal). Modeling results presented in this chapter suggest that reliable seizure prediction from iEEG signal is indeed possible. The proposed SVM-based system for seizure prediction under Group Learning framework can achieve robust prediction of preictal and interictal iEEG segments from dogs with epilepsy.

Two important properties of our seizure prediction system are subject-specific modeling and using heavily unbalanced training data. The proposed seizure prediction system (shown in Figures 7.2 and 7.3) has several novel data-analytic interpretations and improvements:

(1) During *training* stage, a binary classifier is estimated from labeled training samples (20 s windows), as under standard classification setting. Further, we use *unbalanced* training dataset, that includes 20 s samples from *many* interictal segments, in addition to *few* available preictal segments. However, we use *balanced* validation dataset (for model selection), to reflect the clinical requirement that the goal is to classify each 1 hr test segment (as interictal or preictal).

(2) During *testing* stage, the goal is to predict a group of 180 unlabeled test samples (20 s windows), under the assumption that *all* test samples (in this group) belong to the same class. This is clearly different from standard inductive

setting. Further, the system can make three possible predictions for each 1 hr test segment (e.g., *reliable interictal*, *reliable preictal* and *unknown*).

(3) Additional *postprocessing* during testing stage is applied to "unknown" predictions which are all regarded as preictals. This postprocessing scheme assumes that a) the seizure prediction system can predict interictal 1 hr test segments very reliably, and b) the system can predict preictal test segments *either* correctly *or* as "unreliable." This reflects clinical knowledge that interictal segments are inherently much easier to predict (than preictal).

Bibliography

[1] V. Vapnik, *Estimation of Dependences Based on Empirical Data, Empirical Inference Science: Afterword of 2006*. Springer, 2006. iii, 2, 15, 16, 17, 73

[2] V. Vapnik and A. Vashist, "A new learning paradigm: Learning using privileged information," *Neural Networks*, vol. 22, no. 5-6, pp. 544–557, July-August 2009. iii, 2, 15, 17, 30, 31, 34, 35, 38

[3] V. Cherkassky and F. Mulier, *Learning from data: concepts, theory, and methods*, 2nd ed. Hoboken, New Jersey: John Wiley & Sons, 2007. 1, 5, 17, 37, 55, 61, 68, 73, 85, 91, 95, 96

[4] V. Cherkassky, *Predictive Learning*. VCtextbook.com, 2013. 1, 5, 11, 14, 17, 19, 38, 55, 69, 73, 77, 85, 91, 95, 96

[5] F. Cai, "Advanced learning approaches based on SVM+ methodology," Ph.D. dissertation, University of Minnesota, Minneapolis, MN 55455, Jul. 2011. 5

[6] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995. 6, 7, 11

[7] V. Vapnik, *Statistical learning theory*, 1st ed., ser. Adaptive and learning systems for signal processing, communications, and control. Wiley, Sep. 1998. 7

[8] S. Boyd and L. Vandenberghe, *Convex Optimization*, ser. Berichte über verteilte messysteme. Cambridge University Press, 2004. 8

[9] V. Vapnik and R. Izmailov, "Learning with intelligent teacher: Similarity control and knowledge transfer," in *Statistical Learning and Data Sciences*, ser. Lecture Notes in Computer Science, A. Gammerman, V. Vovk, and H. Papadopoulos, Eds. Springer International Publishing, 2015, vol. 9047, pp. 3–32. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-17091-6_1 15

[10] ——, "Learning using privileged information: Similarity control and knowledge transfer," *Journal of Machine Learning Research*, vol. 16, pp. 2023–2049, 2015. [Online]. Available: http://jmlr.org/papers/v16/vapnik15b.html 15, 20

[11] W. Li, D. Dai, M. Tan, D. Xu, and L. Van Gool, "Fast algorithms for linear and kernel SVM+," in *Computer Vision and Pattern Recognition (CVPR)*, 2016. 19, 29

[12] I. W. Tsang, J. T. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVM training on very large data sets," *Journal of Machine Learning Research*, vol. 6, pp. 363–392, Dec. 2005. 24

[13] J. Platt, "Sequential Minimal Optimization: A fast algorithm for training Support Vector Machines," Microsoft Research, Seattle, WA, TechReport MSR-TR-98-14, Apr. 1998. 24

[14] R.-E. Fan, P.-H. Chen, and C.-J. Lin, “Working set selection using second order information for training Support Vector Machines,” *Journal of Machine Learning Research*, vol. 6, pp. 1889–1918, 2005. 24

[15] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 24, 30

[16] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 1.22,” Aug. 2012. [Online]. Available: http://cvxr.com/cvx/ 24

[17] S. Fouad, P. Tino, S. Raychaudhury, and P. Schneider, “Learning Using Privileged Information in prototype based models,” in *Artificial Neural Networks and Machine Learning - ICANN 2012*, ser. Lecture Notes in Computer Science, A. Villa, W. Duch, P. Érdi, F. Masulli, and G. Palm, Eds. Springer Berlin Heidelberg, 2012, vol. 7553, pp. 322–329. 24

[18] D. Pechyony, R. Izmailov, A. Vashist, and V. Vapnik, “SMO-style algorithms for Learning Using Privileged Information,” in *Proceedings of the 2010 International Conference on Data Mining (DMIN’10)*, 2010. 29

[19] D. Pechyony and V. Vapnik, “Fast optimization algorithms for solving SVM+,” *Statistical Learning and Data Science*, 2011. 29, 30

[20] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan, “A dual coordinate descent method for large-scale linear SVM,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML ’08. New York, NY, USA: ACM, 2008, pp. 408–415. [Online]. Available: http://doi.acm.org/10.1145/1390156.1390208 29

[21] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “Liblinear: A library for large linear classification,” *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442794 29

[22] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, “New support vector algorithms,” *Neural Computation*, vol. 12, no. 5, pp. 1207–1245, May 2000. [Online]. Available: http://dx.doi.org/10.1162/089976600300015565 30

[23] H.-T. Shiao, T. Vacek, and V. Cherkassky, “LUPI-based approaches for modeling survival data,” in *Proceedings of the International Workshop on Human is More Than a Labeler (BeyondLabeler), co-located with the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016), New York City, USA, July 10, 2016.*, 2016. 30, 31

[24] O. Aalen, O. Borgan, and H. Gjessing, *Survival and Event History Analysis: A Process Point of View*, ser. Statistics for Biology and Health. Springer New York, 2008. [Online]. Available: https://books.google.com/books?id=2toprArSUMAC 36, 39, 44, 45, 46

[25] J. P. Klein and M. L. Moeschberger, *Survival Analysis: Techniques for Censored and Truncated Data*, ser. SBH Series. Springer, 1997. 36, 40, 41, 44, 45, 46

[26] H.-T. Shiao and V. Cherkassky, "Learning Using Privileged Information (LUPI) for modeling survival data," in *Neural Networks (IJCNN), 2014 International Joint Conference on*, Jul. 2014, pp. 1042–1049. 37, 38

[27] F. M. Khan and V. B. Zubek, "Support Vector Regression for censored data (SVRc): A novel tool for survival analysis," in *Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on*, Dec. 2008, pp. 863–868. 38

[28] J. Shim and C. Hwang, "Support vector censored quantile regression under random censoring," *Comput. Stat. Data Anal.*, vol. 53, no. 4, pp. 912–919, Feb. 2009. 38

[29] P. Shivaswamy, W. Chu, and M. Jansche, "A support vector approach to censored targets," in *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, ser. ICDM '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 655–660. 38

[30] T. Therneau and P. Grambsch, *Modeling Survival Data: Extending the Cox Model*, ser. Statistics for Biology and Health. Springer, 2000. [Online]. Available: https://books.google.com/books?id=9kY4XRuUMUsC 44, 45

[31] M. Zhou. (2015) Use software R to do survival analysis and simulation. a tutorial. [Online]. Available: http://www.ms.uky.edu/~mai/Rsurv.pdf 54

[32] L. Liang, F. Cai, and V. Cherkassky, "Predictive learning with structured (grouped) data," *Neural Networks*, vol. 22, no. 5-6, pp. 766–773, 2009. 54

[33] V. Cherkassky, S. Dhar, and W. Dai, "Practical conditions for effectiveness of the Universum learning," *IEEE Trans. Neural Netw.*, vol. 22, no. 8, pp. 1241–1255, 2011. 55, 96

[34] T. M. Therneau, *A Package for Survival Analysis in R*, 2013, r package version 2.37-4. [Online]. Available: http://CRAN.R-project.org/package=survival 61, 62

[35] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, ser. Springer series in statistics. Springer, 2001. [Online]. Available: https://books.google.com/books?id=VRzITwgNV2UC 68

[36] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, regularization, optimization, and beyond*. MIT press, 2002. 68

[37] G. Camps-Valls, J. Rojo-Álvarez, and M. Martínez-Ramón, *Kernel Methods in Bioengineering, Signal and Image Processing*. Idea Group Publishing, 2007. [Online]. Available: https://books.google.com/books?id=HJJRAAAAMAAJ 68

[38] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, Mar. 2003. 69

[39] G. H. John, R. Kohavi, K. Pfleger *et al.*, "Irrelevant features and the subset selection problem," in *Machine Learning: Proceedings of the Eleventh International Conference*, 1994, pp. 121–129. 69

[40] Q. Gu, Z. Li, and J. Han, "Generalized fisher score for feature selection," in *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, ser. UAI'11.

Arlington, Virginia, United States: AUAI Press, 2011, pp. 266–273. [Online]. Available: http://dl.acm.org/citation.cfm?id=3020548.3020580 69

[41] F. Fleuret, "Fast binary feature selection with conditional mutual information," *Journal of Machine Learning Research*, vol. 5, pp. 1531–1555, 2004. [Online]. Available: http://fleuret.org/papers/fleuret-jmlr2004.pdf 69

[42] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998. 71

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. 71

[44] V. Cherkassky, "New formulations for predictive learning," in *International Joint Conference on Neural Networks (IJCNN)*, 2005. 73

[45] W. Stacey, M. L. V. Quyen, F. Mormann, and A. Schulze-Bonhage, "What is the present-day EEG evidence for a preictal state?" *Epilepsy Research*, vol. 97, no. 3, pp. 243–251, 2011, special Issue on Epilepsy Research UK Workshop 2010 on "Preictal Phenomena". 83

[46] J. J. Howbert, E. E. Patterson, S. M. Stead, B. Brinkmann, V. Vasoli, D. Crepeau, C. H. Vite, B. Sturges, V. Ruedebusch, J. Mavoori, K. Leyde, W. D. Sheffield, B. Litt, and G. A. Worrell, "Forecasting seizures in dogs with naturally occurring epilepsy," *PLoS ONE*, vol. 9, no. 1, pp. 1–8, Jan. 2014. 83, 84, 89, 91

[47] J. Rasekhi, M. R. K. Mollaei, M. Bandarabadi, C. A. Teixeira, and A. Dourado, "Preprocessing effects of 22 linear univariate features on the performance of seizure prediction methods," *Journal of Neuroscience Methods*, vol. 217, no. 1–2, pp. 9–16, 2013. 83, 84, 89

[48] Y. Park, L. Luo, K. K. Parhi, and T. Netoff, "Seizure prediction with spectral power of EEG using cost-sensitive Support Vector Machines," *Epilepsia*, vol. 52, no. 10, pp. 1761–1770, 2011. 83, 84, 87, 89, 91

[49] J. R. Williamson, D. W. Bliss, D. W. Browne, and J. T. Narayanan, "Seizure prediction using EEG spatiotemporal correlation structure," *Epilepsy & Behavior*, vol. 25, no. 2, pp. 230–238, 2012. 83, 84

[50] L. Chisci, A. Mavino, G. Perferi, M. Sciandrone, C. Anile, G. Colicchio, and F. Fuggetta, "Real-time epileptic seizure prediction using AR models and Support Vector Machines," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 5, pp. 1124–1132, May 2010. 83, 84, 91

[51] L. Iasemidis, "Epileptic seizure prediction and control," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 5, pp. 549–558, May 2003. 83

[52] B. Litt, R. Esteller, J. Echauz, M. D'Alessandro, R. Shor, T. Henry, P. Pennell, C. Epstein, R. Bakay, M. Dichter, and G. Vachtsevanos, "Epileptic seizures may begin hours in advance of clinical onset: A report of five patients," *Neuron*, vol. 30, no. 1, pp. 51–64, 2001. 83, 84

[53] B. Litt and K. Lehnertz, “Seizure prediction and the preseizure period,” *Curr. Opin. Neurol.*, vol. 15, no. 2, pp. 173–177, Apr. 2002. 83, 84

[54] K. Lehnertz, F. Mormann, H. Osterhage, A. Mller, J. Prusseit, A. Chernihovskyi, M. Staniek, D. Krug, S. Bialonski, and C. E. Elger, “State-of-the-art of seizure prediction,” *Journal of Clinical Neurophysiology*, vol. 24, no. 2, pp. 147–153, Apr. 2007. 83

[55] F. Mormann, R. G. Andrzejak, C. E. Elger, and K. Lehnertz, “Seizure prediction: the long and winding road,” *Brain*, vol. 130, no. 2, pp. 314–333, 2007. 83

[56] C. E. Elger and F. Mormann, “Seizure prediction and documentation—two important problems,” *The Lancet Neurology*, vol. 12, no. 6, pp. 531–532, 2013. 83

[57] E. E. Patterson, “Canine epilepsy: An underutilized model,” *ILAR Journal*, vol. 55, no. 1, pp. 182–186, 2014. 84

[58] B. H. Brinkmann, E. E. Patterson, C. Vite, V. M. Vasoli, D. Crepeau, M. Stead, J. J. Howbert, V. Cherkassky, J. B. Wagenaar, B. Litt, and G. A. Worrell, “Forecasting seizures using intracranial eeg measures and svm in naturally occurring canine epilepsy,” *PLoS ONE*, vol. 10, no. 8, pp. 1–12, Aug. 2015. 84

[59] B. H. Brinkmann, J. Wagenaar, D. Abbot, P. Adkins, S. C. Bosshard, M. Chen, Q. M. Tieng, J. He, F. J. Muñoz-Almaraz, P. Botella-Rocamora, J. Pardo, F. Zamora-Martinez, M. Hills, W. Wu, I. Korshunova, W. Cukierski, C. Vite, E. E. Patterson, B. Litt, and G. A. Worrell, “Crowdsourcing reproducible seizure forecasting in human and canine epilepsy,” *Brain*, vol. 139, no. 6, pp. 1713–1722, 2016. 84

[60] D. E. Snyder, J. Echauz, D. B. Grimes, and B. Litt, “The statistics of a practical seizure warning system,” *Journal of Neural Engineering*, vol. 5, no. 4, pp. 392–401, 2008. 84

[61] M. Bandarabadi, J. Rasekhi, C. A. Teixeira, M. R. Karami, and A. Dourado, “On the proper selection of preictal period for seizure prediction,” *Epilepsy & Behavior*, vol. 46, pp. 158–166, 2015. 84

[62] M. Bandarabadi, C. A. Teixeira, J. Rasekhi, and A. Dourado, “Epileptic seizure prediction using relative spectral power features,” *Clinical Neurophysiology*, vol. 126, no. 2, pp. 237–248, 2015. 84

[63] H.-T. Shiao, V. Cherkassky, J. Lee, B. Veber, E. Patterson, B. Brinkmann, and G. Worrell, “SVM-based system for prediction of epileptic seizures from iEEG signal,” *IEEE Trans. Biomed. Eng.*, vol. 64, pp. 1011–1022, May 2017. 88, 99

[64] V. Cherkassky, B. Veber, J. Lee, H.-T. Shiao, E. Patterson, G. A. Worrell, and B. H. Brinkmann, “Reliable seizure prediction from EEG data,” in *Neural Networks (IJCNN), 2015 International Joint Conference on*, Jul. 2015. 91, 92, 95, 96, 97, 99