- Sparse Machine Learning - Regularised Classification of DNA Microarrays

Daniel Arif Murphy

Supervised by Dr. Jochen Einbeck

A thesis submitted in partial fulfilment of the requirements for the degree of

MSc Mathematical Sciences

Department of Mathematical Sciences
Durham University, UK

September, 2020

Abstract

Provided is a discussion of statistical learning for binary classification and its algorithmic approaches, with ultimate focus on the context of high-dimensional medical data. Both logistic regression (LR) and the support vector machine (SVM) are examined as objective function minimisers, and regularisation terms are appended to the two base models to induce sparse solutions. We note their remarkable similarities as joint members of a general family of classifiers. Relevant methods for error estimation are laid out, and comparative data analysis is conducted with the aim of accurately predicting the presence of certain cancers from patient tissue samples. The successes of the elastic-net and the smoothly clipped absolute deviation (SCAD) penalties, in particular, are empirically justified. Finally, some shortcomings of the utilised models are considered, and promising theoretical extensions are discussed for future applications of binary classification methods as medical diagnostic tools.

Acknowledgements

I would like to express my greatest thanks to my supervisor, Dr. Jochen Einbeck, for his continued guidance in both writing an academic report and on technical subject matter, as well as for his patience and support in light of the coronavirus pandemic; his time has been invaluable.

Declaration of Authorship

This piece of work is a result of my own work except where it forms an assessment based on group project work. In the case of a group project, the work has been prepared in collaboration with other members of the group. Material from the work of others not involved in the project has been acknowledged and quotations and paraphrases suitably indicated.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

Statistical learning, as a discipline, is forced to be both rigorous and adaptive in facing the evolving challenges presented to it by scientific developments. With the emergence of the so-coined "fourth industrial revolution", statistical questions have multiplied in scope and intricacy. The notable advent of "bioinformatics" arose from the intersection between computational biology and statistics. We are left entrusted with the task of parsing through the noise that is characteristic of such volumes of data.

According to [22], the process of reducing mass data to its fundamentals requires the hope that the world is less complex that it may seem. The processes that induce malignant tumour development in humans, for instance, we hope not to be directly caused by *all* of the $30,000$ or so genes that the Human Genome Project estimates we possess [7]. This hope for simplicity can come in the form of *sparsity*, where only a select few predictor variables prove explanatorily significant.

More generally, "machine learning" can be considered a hypernym that constitutes three paradigmatic learning approaches: supervised, unsupervised, and reinforcement learning. For the purposes of this thesis, we focus on the first. Supervised learning can be interpreted as featuring a learning agent with a set of labelled data at her disposal, courtesy of some extrinsic, intelligent supervisor. Using the knowledge acquired from this training set, the agent's goal is to extrapolate responses appropriately to cases not initially encountered; that is, she must learn to generalise well in uncharted territory. *Classification* involves a supervised learning problem where the response is some qualitative variable. This may be the presence of a certain medical condition, for instance. Indeed, as concisely observed by Rutherford D. Rogers [21]:

> *"We are drowning in information and starving for knowledge."*

In this thesis, we examine the ability of two popular binary classifiers to satisfy this need. In particular, *logistic regression* (LR) and the *support vector machine* (SVM) are

theoretically dissected, and their classification performances empirically contrasted on a selection of medical datasets.

1.2 Organisation of Thesis

This thesis is split into six chapters. Succeeding this introduction, *Chapter 2* formally introduces the binary classification task. Both LR and the SVM, as well as their regularised modifications, are individually examined, with the latter part of the chapter drawing a clear relationship between the classifiers. *Chapter 3* elucidates the importance of error estimation in performance evaluation. *Chapter 4* deals with application to real medical datasets and contrasts the performance results with what we would expect given the theory in the preceding chapters. Namely, we tackle high-dimensional feature spaces; that is, we traverse the "$p \gg N$" problem. Some additional theoretical augmentations that appear in the literature are briefly discussed in *Chapter 5*, and *Chapter 6* concludes the thesis. The R code to produce the examples in this thesis can be found at the relevant GitHub repository, here.

Chapter 2

A Move to Classification

This chapter chiefly relates to the exposition provided in [21] and [25], unless otherwise stated. We begin with a short narrative of how linear regression fails to act as a robust classification tool, and introduce some common alternative approaches. Namely, the classification models presented are logistic regression (LR) [27] and the support vector machine (SVM) [8]. Modifications are then appended to the base models to induce sparse solutions.

2.1 Introduction to Statistical Learning

In statistical learning, suppose a response variable Y exhibits some relationship with a set of p predictor variables, given by $\mathbf{X} = (X_1, X_2, \ldots, X_p)^T$. We can then write, as an approximation to reality, that our data was derived from a model of the form

$$Y = \phi(\mathbf{X}) + \epsilon, \tag{2.1}$$

where we make the assumption that the random noise is such that $\mathrm{E}(\epsilon) = 0$. The error ϵ is further assumed to capture all the true behaviour between $\mathbf{X}$ and Y that depart from a deterministic relationship. The goal, then, is to estimate the underlying mapping ϕ. In this thesis we assess *parametric* methods, which reduce the task to that of approximating a parameter set, to estimate ϕ through the estimated model $\hat{\phi}$. We also see *non-parametric* methods, which avoid pre-determining the true form of ϕ and instead construct it via the provided data, to estimate the error associated with $\hat{\phi}$. Generally, we can make predictions via

$$\hat{Y} = g(\mathbf{X}), \tag{2.2}$$

where g is a *decision function* that encapsulates our approximation to ϕ, and $\hat{Y}$ gives the corresponding *fitted values* for Y. In a more practical sense, not only are we concerned with raw prediction power, but also the inferences we can draw from the manner in which Y is influenced by $X_1, \ldots, X_p$. In particular, we hope to discern which of the p predictors are the predominant drivers of Y and to place lesser importance on those

that are not; that is, we aim to create *sparse* models. With respect to computational complexity, the *bet on sparsity* principle is informally coined in [21], which states that one ought to "use a procedure that does well in sparse problems, since no procedure does well in dense problems." We consider methods used to achieve this in sections to come.

To estimate ϕ, consider a *training set*, or learning sample, $\tau = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)\}$, which consists of observations $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$ and their related response variables y_i, where $i = 1, \ldots, N$. For problems involving binary classification, where Y is discrete and takes on one of two possible *classes*, which we suitably dummy code through the values $\{0, 1\}$, we wish to be able to classify Y for a set of novel observations. A *classifier* can be considered a decision function g which, given τ, allocates a class to data $\boldsymbol{x} \in \mathbb{R}^p$. More helpfully, some classification tools such as LR, which we will dissect in the following section, output a *probability* of belonging to each of the respective classes.

But if we have access to the explanatory variables, why do standard linear regression techniques not suffice here? Similarly to [25], we provide an illustrative example with the `Default` dataset accessible through the `ISLR` package in R. Consider the simple linear regression model

$$Y = \boldsymbol{X}\boldsymbol{\beta} + \epsilon, \tag{2.3}$$

where Y is the n-dimensional vector of responses, $\boldsymbol{X}$ is the $n \times p$ design matrix, the (i, j) entries of which give the jth feature of the ith observation, $\boldsymbol{\beta}^T = (\beta_1, \ldots, \beta_p)$ is the p-dimensional vector of parameters, and $\epsilon^T = (\epsilon_1, \ldots, \epsilon_n)$ is the vector of errors. Here, to construct our decision function g, one estimates the parameters via $\hat{\boldsymbol{\beta}}^T = (\hat{\beta}_1, \ldots, \hat{\beta}_p)$, which is found by minimising the sum of the squares of the residuals

$$\begin{aligned} R(\boldsymbol{\beta}) &= \sum_{i=1}^{N} \epsilon_i^2 = \epsilon^T \epsilon \\ &= (Y - \boldsymbol{X}\boldsymbol{\beta})^T (Y - \boldsymbol{X}\boldsymbol{\beta}) \\ &= Y^T Y - Y^T \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\beta}^T \boldsymbol{X}^T Y + \boldsymbol{\beta}^T \boldsymbol{X}^T \boldsymbol{X}\boldsymbol{\beta}. \end{aligned} \tag{2.4}$$

Setting the derivative to zero

$$\frac{\partial}{\partial \boldsymbol{\beta}} R(\boldsymbol{\beta}) = -\boldsymbol{X}^T Y - \boldsymbol{X}^T Y + 2(\boldsymbol{X}^T \boldsymbol{X})\boldsymbol{\beta} = 0$$

and solving gives us

$$\boldsymbol{X}^T \boldsymbol{X} \hat{\boldsymbol{\beta}} = \boldsymbol{X}^T Y. \tag{2.5}$$

That is, $\hat{\boldsymbol{\beta}}$ is yielded as the solution to p *normal equations*.

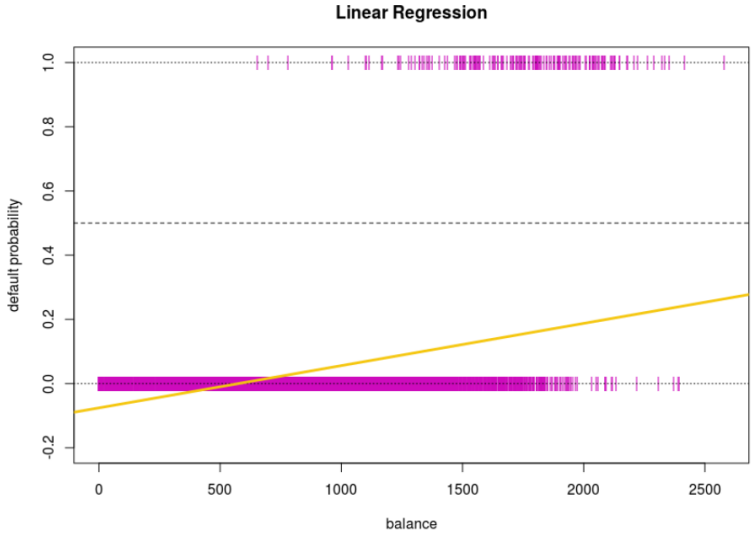


Figure 2.1: Linear regression probability estimates of `default`, coded for 0/1 values representing `No/Yes`, respectively. Some predicted probabilities are negative!

Unfortunately, Figure 2.1 highlights how fitting the standard linear regression line can result in probability estimates that lie outside of the interval $[0, 1]$, causing interpretability to become somewhat confusing.

Even though we could interpret the output of the linear regression model as a sort of crude probability measure since one can deduce an ordinal ranking, this becomes an issue for more complex problems. To cater for this, we introduce some methods more naturally suited to the specific task of binary qualitative classification.

2.2 Logistic Regression (LR)

The following theoretical discussion is principally drawn from the work of [1]. Let the response variable Y be binary and follow a Bernoulli distribution

$$Y \sim \mathcal{B}(1, p(\boldsymbol{x})). \tag{2.6}$$

That is, we have conditional probabilities such that

$$p(\boldsymbol{x}) = \mathbb{P}(Y = 1|\boldsymbol{X} = \boldsymbol{x}) = 1 - \mathbb{P}(Y = 0|\boldsymbol{X} = \boldsymbol{x}), \tag{2.7}$$

where $\boldsymbol{x} \in \mathbb{R}^p$ is a vector of p predictors, and we have the probability function of the response $p : \mathbb{R}^p \to [0, 1]$. One such function that maps outputs within this range is that of the *logistic function*. Then the *multiple logistic regression* model is given by

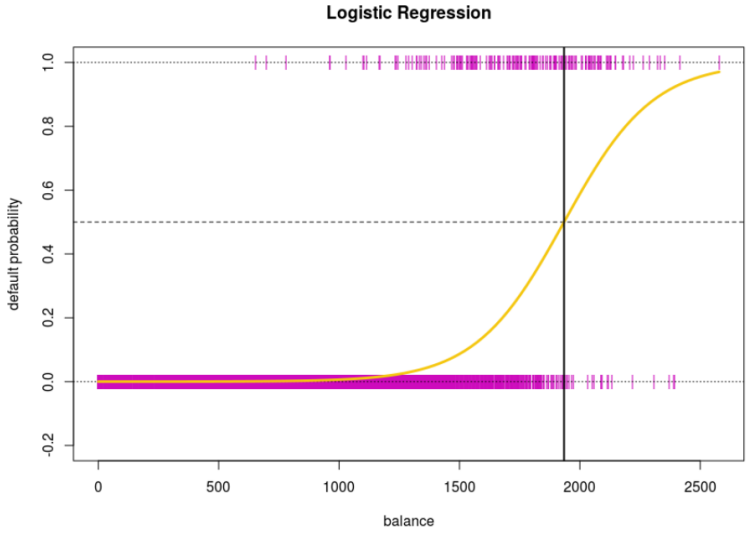


Figure 2.2: LR probability estimates of `default`.

$$p(\boldsymbol{x}) = \frac{e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}}}{1+e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}}} = \frac{\exp(\beta_0+\beta_1 x_1+\ldots+\beta_p x_p)}{1+\exp(\beta_0+\beta_1 x_1+\ldots+\beta_p x_p)}. \tag{2.8}$$

With some algebraic manipulation, we arrive at the *odds*

$$\frac{p(\boldsymbol{x})}{1-p(\boldsymbol{x})} = e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}} \tag{2.9}$$

and, equivalently, we obtain the log-odds, or *logit*, by taking logarithms. The model can hence be described in terms of some linear combination of the predictor variables

$$\log\left(\frac{p(\boldsymbol{x})}{1-p(\boldsymbol{x})}\right) = \text{logit}\Big(p(\boldsymbol{x})\Big) = \beta_0+\boldsymbol{\beta}^T\boldsymbol{x}. \tag{2.10}$$

It can be seen that the LR model takes a predictor vector $\boldsymbol{x} = (x_1,\ldots,x_p)^T$ and generates the posterior probabilities that the response Y will lie in each class. Figure 2.2 depicts the logistic curve applied to the `Default` dataset. In contrast to the fitted linear regression line, the S-shaped form permits probability estimates in the desired range, which can be visualised easily for $p = 1$. The obvious question that follows concerns how the coefficient parameters β_0 and $\boldsymbol{\beta}$ are estimated. A *maximum likelihood* approach is taken in the case of LR [23]. The following is adapted from [21] for the binary case with some extra work. Considering the training set $\tau = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)\}$, the likelihood function is given by

$$
\begin{aligned}
l(\beta_0, \boldsymbol{\beta}) &= \prod_{i=1}^{N} p(\boldsymbol{x}_i)^{y_i} \Big(1 - p(\boldsymbol{x}_i)\Big)^{1-y_i} \\
&= \prod_{i=1}^{N} \Big(1 - p(\boldsymbol{x}_i)\Big) \prod_{i=1}^{N} \left(\frac{p(\boldsymbol{x}_i)}{1 - p(\boldsymbol{x}_i)}\right)^{y_i}. \qquad (2.11)
\end{aligned}
$$

It becomes computationally simpler to use the logarithm of (2.11), the *log-likelihood*

$$
\begin{aligned}
\mathcal{L}(\beta_0, \boldsymbol{\beta}) &= \log\Big[l(\beta_0, \boldsymbol{\beta})\Big] \\
&= -\sum_{i=1}^{N} \log\Big[1 + \exp\left(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i\right)\Big] + \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i\right), \qquad (2.12)
\end{aligned}
$$

where the equality follows from (2.8) and (2.10). By maximising (2.12), we obtain the parameter estimates $\hat{\beta}_0$ and $\hat{\boldsymbol{\beta}}^T = (\hat{\beta}_1, \ldots, \hat{\beta}_p)$. Allowing the partial derivatives to vanish results in the $p + 1$ *score equations*:

$$
\frac{\partial}{\partial \beta_0} \mathcal{L}(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{N} \left\{ y_i - \frac{e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i}}{1 + e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i}} \right\} = 0; \qquad (2.13)
$$

$$
\frac{\partial}{\partial \boldsymbol{\beta}} \mathcal{L}(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{N} \left\{ y_i \boldsymbol{x}_i - \frac{e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i}}{1 + e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i}} \right\} = 0. \qquad (2.14)
$$

Whilst reminiscent of its least squares cousin, the system given by (2.13) and (2.14) is *nonlinear* in $(\beta_0, \boldsymbol{\beta}^T)$, and hence must be evaluated iteratively [21]. One such method is *Newton-Raphson* which updates through successive approximations and requires the Hessian. The details of this can be found in [33].

LR, in its raw form, outputs a set of estimated probabilities. This gives it an advantage over other classification methods when it comes to statistical inference. In our case, as we are primarily concerned with explicit classification performance, the LR model must be modified slightly. This can be done by assigning a threshold r such that we allocate a novel observation $\boldsymbol{z} = (z_1, z_2, \ldots, z_p)$ to class 1 if $\mathbb{P}(Y = 1 | \boldsymbol{X} = \boldsymbol{z}) \geq r$, and to class 0 otherwise. For our data analysis in Chapter 4, we set $r = 0.5$; that is, observations are assigned to the class for which the probability estimate is highest. It should still be noted that the value for r leading to optimal classification performance is often data-dependent, as we will see later when applying the models.

2.2.1 Shrinkage Methods and Sparsity

Model selection attempts to attain a sub-model that exhibits improved prediction, the accuracy of which typically only performs well for problems with $n > p$ [25]. Indeed, prudently shrinking the magnitude of the parameter estimates can, for a marginal hike in *bias*, significantly improve *variance*, a statistical trade-off we visit in Chapter 3. Roughly, bias is a measure of the error arising from using a comparatively simpler model to approximate a (very) complex problem. Conversely, the variance is a measure of how the estimated model $\hat{\phi}$ would change upon a change in the training set τ. Intuitively, then, it is desirable for a model to exhibit both low bias and variance simultaneously. A further advantage of model selection is interpretability: sacrificing predictors with the weakest effects - that is, performing *feature selection* - lessens unnecessary complexity. We visit penalisation methods that conduct this automatically.

There are three major approaches to model selection, two of which are mentioned here in passing: *subset selection*, which applies least squares estimation to a reduced set of the p predictors, and *dimensionality reduction*, which projects all explanatory variables to a d-dimensional subspace such that $d < p$, are outlined in depth in [25]. For our intents and purposes, we limit our discussion to techniques involving *regularisation*, or shrinkage, where coefficient estimates are reduced toward zero. In contrast to the discrete processes involved in subset selection and dimensionality reduction, regularisation is more continuous and hence achieves lesser variance [21].

Ridge regression [11] attempts to tackle the instabilities that may result from predictor collinearity, and circumvents the issue of overfitting to the training set. As alluded to, it achieves this by controlling the magnitude of the variable parameters. In ridge LR, rather than simply maximising (2.12), we first append a regularisation penalty term to the log-likelihood as

$$-\sum_{i=1}^{N}\log\Big[1+\exp\left(\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i\right)\Big]+\sum_{i=1}^{N}y_i\left(\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i\right)-\lambda\|\boldsymbol{\beta}\|_2^2, \tag{2.15}$$

where $\lambda \geq 0$ is the complexity parameter, to be tuned separately, which dictates the level of shrinkage. Practically, the value of λ is usually chosen to minimise the prediction error according to *cross-validation*, a resampling method for model selection we visit in Chapter 3. The penalty in (2.15) involves a member of a family of l_q-norms defined by

$$\|\boldsymbol{w}\|_q=\left(\sum_{j=1}^{n}|w_j|^q\right)^{1/q} \tag{2.16}$$

for some vector $\boldsymbol{w}=(w_1,\ldots,w_n)$. Typically, the task of estimating the ridge parameters is equivalently formulated as minimising the negation of the quantity in (2.15). That is, estimates of true coefficients $(\beta_0^r,\boldsymbol{\beta}^{rT})$ are such that

$$\left(\hat{\beta}_0^r, \hat{\boldsymbol{\beta}}^{rT}\right) = \underset{\beta_0,\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \log\left(1 + e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i}\right) - \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i\right) + \lambda\|\boldsymbol{\beta}\|_2^2 \right\}. \quad (2.17)$$

One should note that β_0 is absent from the penalty term. Although improved stability is obtained by imposing the ridge penalty, all predictors remain included in the model since the coefficients are never shrunk fully to zero. To improve inference for problems where p is large, we want a hybrid of shrinkage and feature selection; we want a penalty that introduces sparsity.

The least absolute shrinkage and selection operator, or *lasso*, is a relatively modern alternative to ridge, courtesy of Tibshirani [36], which achieves precisely this. As applied to LR, estimates of the parameters $(\beta_0^l, \boldsymbol{\beta}^{lT})$ are calculated via

$$\left(\hat{\beta}_0^l, \hat{\boldsymbol{\beta}}^{lT}\right) = \underset{\beta_0,\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \log\left(1 + e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i}\right) - \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i\right) + \lambda\|\boldsymbol{\beta}\|_1 \right\}. \quad (2.18)$$

The resemblance this bears to (2.17) is notable, with the lasso l_1 penalty substituting in for the ridge l_2 penalty. Here, the act of minimising the regularisation term $\lambda\|\boldsymbol{\beta}\|_1 = \lambda\sum_{k=1}^{p}|\beta_k|$ induces automatic feature selection, since the shrinkage of some coefficients to exactly zero is permitted. They hence face total exclusion from the LR model. The mechanistic difference between (2.17) and (2.18) becomes more intuitive upon reformulation to a constrained optimisation problem. Explicitly, we have

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \log\left(1 + e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i}\right) - \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i\right) \right\} \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_2^2 \leq t \quad (2.19)$$

and

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \log\left(1 + e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i}\right) - \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i\right) \right\} \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_1 \leq t \quad (2.20)$$

for the ridge and lasso problems, respectively. For a problem with p variables, the corresponding budget sets can be seen to be a p-dimensional hypersphere for the ridge constraint, and a p-dimensional polytope for the lasso constraint [25]. This is visualised for the $p = 2$ case in Figure 2.3 for some generic objective functions. The budget sets are given by $\beta_1^2 + \beta_2^2 \leq t$ and $|\beta_1| + |\beta_2| \leq t$ for ridge and lasso accordingly. The *red* contours satisfy (2.19), whilst the *green* contours satisfy (2.20). With one exception, the intersections with the lasso constraint occur at a corner.

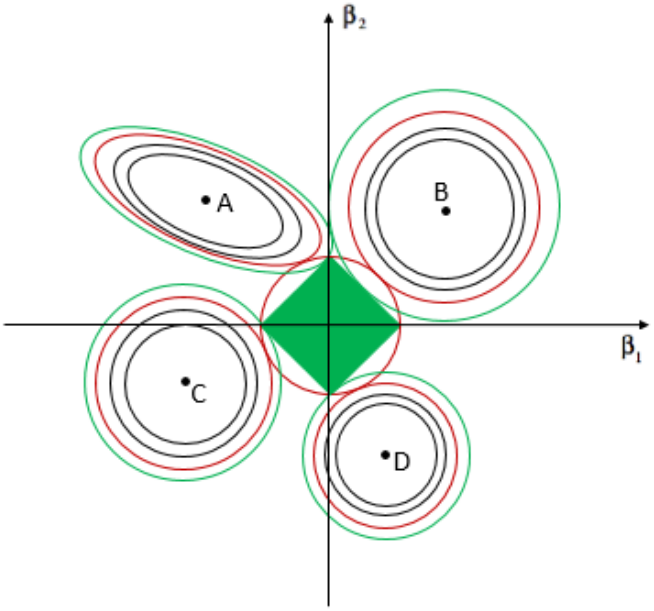


Figure 2.3: Objective function contours with budget sets for the l_1 penalty (green diamond) and l_2 penalty (red disk).

Geometrically, then, due to the sharpness of the polytope budget, the intersections readily occur on the axes of the parameter space. Generally, if the intersection occurs on an axis orthogonal to the kth dimension, the kth component of estimate $\hat{\boldsymbol{\beta}}^T = (\hat{\beta}_1, \ldots, \hat{\beta}_p)$ is shrunk to zero. That is, the lasso penalty conducts variable selection, and yields sparse solutions as we desire. Referring again to Figure 2.3, we see this in action. In all cases, the ridge constraint results in some non-zero estimates for *both* β_1 and β_2. In the instance of the lasso constraint, conversely, objective functions A and D select non-zero estimates for β_2 only, and C does so only for β_1. Indeed, it is only for case B that neither coefficients hit zero. In more high-dimensional settings with $p \gg N$ the difference becomes more stark, since the standard lasso can only select for a maximum of N features. This follows from a property of the optimisation problem since the rank of the design matrix $\boldsymbol{X}$ never exceeds N, and hence at most N unique solutions can be obtained.

2.2.2 Generalisations Beyond the Lasso

As we will see in Chapter 4, the performance of the lasso penalty can sometimes be contingent on how correlated the selected features are with those that do not get selected. This is an allusion to what we'll eventually see to be a failure to satisfy the *irrepresentable condition* [41]. One possible solution to this, as well as to the issue mentioned at the end of the previous subsection, is a hybrid of the ridge and lasso: the *elastic-net*. First proposed in 2005 by [46], this creates a convex problem involving a linear combination

of the l_1 and l_2 penalties, with the preference for each extreme case being governed by some constant, α. We here adapt the rationale outlined in [46] for our case of binary LR. Formally, coefficients of the true $(\beta_0^e, \boldsymbol{\beta}^{eT})$ are solutions to

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \log\left(1 + e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i}\right) - \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i\right) + \lambda \left[(1-\alpha)\|\boldsymbol{\beta}\|_2^2 + \alpha\|\boldsymbol{\beta}\|_1 \right] \right\}, \tag{2.21}$$

or, equivalently

$$\begin{aligned} &\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \log\left(1 + e^{\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i}\right) - \sum_{i=1}^{N} y_i \left(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i\right) \right\} \\ &\text{subject to} \quad (1-\alpha)\|\boldsymbol{\beta}\|_2^2 + \alpha\|\boldsymbol{\beta}\|_1 \leq t. \end{aligned} \tag{2.22}$$

It should be noted that [22] include in the elastic-net penalty term a factor of $1/2$ in the quadratic component, but we omit this in line with [46]. It is clear that (2.22) condenses to the the ridge problem (2.19) for $\alpha = 0$, and condenses to the lasso problem (2.20) for $\alpha = 1$. Figure 2.4 depicts how the parameter α dictates the strength of convexity of the elastic-net budget region. Notice at the vertices of the elastic-net curve there are singularities, similarly to the extreme lasso case. For $\alpha \in (0, 1)$, elastic-net selects for more features than the lasso, but the coefficients of each feature have a reduced magnitude [21]. This contrast can be seen in Figure 2.5, where we have plotted the coefficient paths of the lasso and elastic-net LR models for the `Sonar` dataset from the `datasets` package.

What we see in all of the above instances is that we can always express the problem as minimising a general form of the regularised log-likelihood

$$\mathcal{L}_{\mathcal{R}}(\lambda; \beta_0, \boldsymbol{\beta}) = -\mathcal{L}(\beta_0, \boldsymbol{\beta}) + \lambda \mathcal{R}_\alpha(\boldsymbol{\beta}). \tag{2.23}$$

where, to remain compact, the negative log-likelihood function is expressed in short-hand as defined in (2.12). Similarly to the non-regularised instance, Newton-Raphson can be iteratively applied to (2.23) to derive the coefficient estimates [18]. But the derived estimates are heavily contingent on λ. The magnitude of λ dictates the influence the regularisation term $\mathcal{R}_\alpha(\boldsymbol{\beta})$ has on the minimisation problem. This naturally begs the question concerning suitable methods for selecting optimal values for λ and, in the case of the elastic-net, α. Typical procedures include, amongst others, theoretical *information criteria* and empirically-based methods such as cross-validation. The latter is perhaps the most popular of these techniques and provides a focus for discussion in Chapter 3.

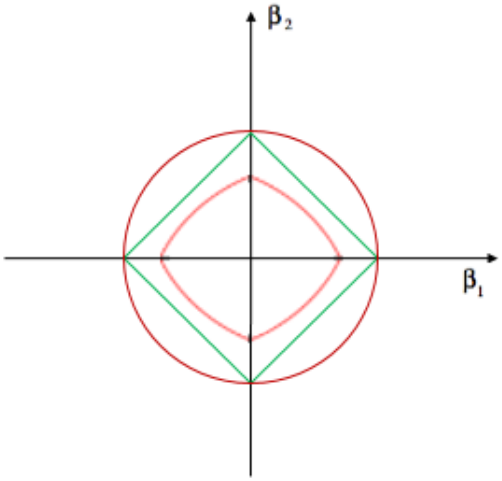


Figure 2.4: Budget sets for the l_1 penalty (green diamond), l_2 penalty (dark red disk), and elastic-net penalty for $\alpha = 0.5$ (light red curve).

2.2.3 Sparse Group Lasso

This section illustrates some theory regarding the *sparse group lasso*. This is based on the exposition provided in [22] after the introduction of the *group lasso* in [39], which we begin with, but adapt in this case to incorporate the LR model.

When the covariates can be logically divided into groups, and the group structure is known, it can be advantageous to achieve sparsity not only via shrinkage of the individual coefficients $(\beta_0, \boldsymbol{\beta}^T)$, but to eliminate entire groups of related variables at once. That is, to obtain *inter*-group sparsity. Take an LR model where the p features are split into G groups such that $\boldsymbol{X}_k \in \mathbb{R}^{p_k}$ holds the covariates of the kth group, where $k = 1, \ldots, G$. Then the coefficient estimates of the group lasso are the solutions to

$$\min_{\beta_0, \boldsymbol{\beta}} \left\{ -\mathcal{L}(\beta_0, \boldsymbol{\beta}) + \lambda \sum_{k=1}^{G} \|\boldsymbol{\beta}_{(k)}\|_2 \right\}, \tag{2.24}$$

where the whole parameter vector is defined by β_0 and $\boldsymbol{\beta}^T = (\beta_1, \ldots, \beta_p) = (\boldsymbol{\beta}_{(1)}^T, \ldots, \boldsymbol{\beta}_{(G)}^T)$. In this extension of the lasso, either *all* components of the estimate vector $\hat{\boldsymbol{\beta}}_{(k)}$ are shrunk to exactly zero, or all remain non-zero. It can be further seen that (2.24) reduces to the ordinary lasso case in (2.18) when $p_k = 1$ for all $k = 1, \ldots, G$. That is, when all groups contain a single covariate, and $G = N$. This comes from the fact that, for a singleton $\beta_{(k)}$, the non-squared l_2 norm outputs the quantity $\sqrt{\beta_{(k)}^2} = |\beta_{(k)}|$, which is precisely what we get with the standard lasso.

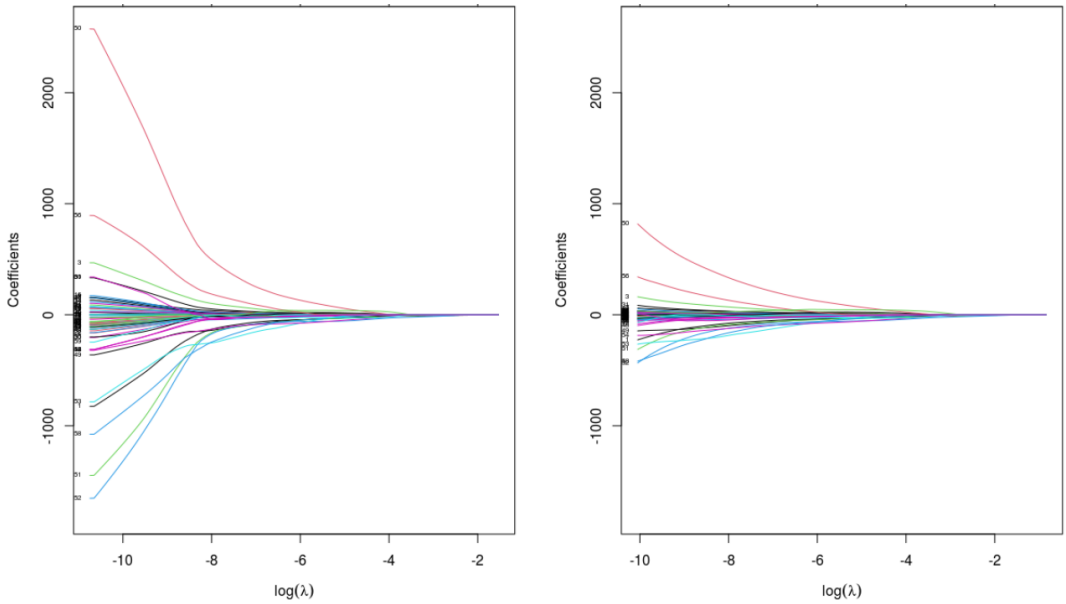


Figure 2.5: Regularised LR coefficient paths for the `Sonar` data: the lasso path (left) and the elastic-net path (right) where $\alpha = 0.5$.

What the group lasso fails to do, as a result of the l_2 penalty, is generate *intra*-group sparsity. It can be desirable to have sparsity both at the level of the group, and with respect to individual coefficients within each group [21]. A more general penalty that achieves this makes additional use of the l_1 norm. The coefficient estimates are now the solutions to the convex problem

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ -\mathcal{L}(\beta_0, \boldsymbol{\beta}) + \lambda \left[\alpha \|\boldsymbol{\beta}\|_1 + (1-\alpha) \sum_{k=1}^{G} \|\boldsymbol{\beta}_{(k)}\|_2 \right] \right\}, \tag{2.25}$$

where $\alpha \in [0, 1]$. Reminiscent of the elastic-net, the choice of α determines the linear combination of the extreme cases: $\alpha = 1$ corresponds to the original lasso (2.18), and $\alpha = 0$ corresponds to the group lasso (2.24). Figure 2.6 compares, for three features, the budget sets for the group lasso and the sparse group lasso. Observe the likeness between the budget set in the (β_1, β_2)-space here with that of the elastic-net in Figure 2.4.

It should be noted that sometimes the negative log-likelihood as in (2.12) can be explicitly rewritten with respect to the group structure of the variables, similarly to [28], as

$$\mathcal{L}(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{N} y_i \left(\beta_0 + \sum_{k=1}^{G} \boldsymbol{\beta}_{(k)}^T \boldsymbol{x}_{i,(k)} \right) - \sum_{i=1}^{N} \log \left[1 + \exp \left(\beta_0 + \sum_{k=1}^{G} \boldsymbol{\beta}_{(k)}^T \boldsymbol{x}_{i,(k)} \right) \right], \tag{2.26}$$

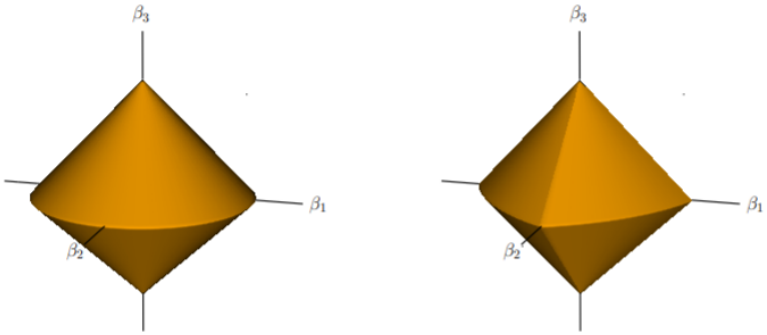


Figure 2.6: The group lasso budget (left) and the sparse group lasso budget (right) where $\alpha = 0.5$ [21]. There are two groups with parameters $\boldsymbol{\beta}_{(1)} = (\beta_1, \beta_2)^T \in \mathbb{R}^2$, and $\boldsymbol{\beta}_{(2)} = \beta_3 \in \mathbb{R}$.

where the $\boldsymbol{x}_{i,(k)}$ are defined such that $\boldsymbol{x}_i^T = (x_{i1}, \ldots, x_{ip}) = (\boldsymbol{x}_{i,(1)}^T, \ldots, \boldsymbol{x}_{i,(G)}^T)$, and $\boldsymbol{x}_i$ is the entire feature vector of the ith observation, rewritten in an analogous fashion to $\boldsymbol{\beta}^T$ above. Since the elements in these linear summations are commutative, this rephrasing does not influence the calculation of the parameter estimates. It is only through the penalty terms in (2.24) and (2.25) that group structures come into play.

More recently, a group-regularised elastic-net LR has been proposed, though this is not discussed here. We point the interested reader to the work of [31], and to the associated `gren` package in R.

2.3 Support Vector Machines (SVMs)

The theory of this section is largely inspired by the treatment as per [21] and [25], and introduces the next of the two base classifiers dealt with in this thesis: the support vector machine (SVM) [8]. Though the discussion to follow assumes a general p-dimensional space, the visualisations are restricted such that $p = 2$ for illustrative purposes. We initially limit the discussion to that concerning data of the *linearly separable* variety, until kernels are introduced for the non-separable case.

2.3.1 Hard-Margin Classifier

Consider again a set of p predictor variables $\boldsymbol{X} = (X_1, \ldots, X_p)^T$, with a binary response variable Y. Similarly to LR, Y can belong to one of two classes which, in the case of SVMs, are typically chosen to be $+1$ and -1. Hence, we have $Y \in \{-1, 1\}$. The classifying mechanism for SVMs is based on the notion of a *separating hyperplane*. Consider a $(p-1)$-dimensional hyperplane that is able to perfectly segregate the two classes in a p-dimensional feature space. Figure 2.7 gives an instance of this, where for $p = 2$ the hyperplane corresponds to a straight line. Now, take a hyperplane defined by

$$\Big\{\boldsymbol{x} \,:\, \phi(\boldsymbol{x}) = \beta_0 + \boldsymbol{\beta}^T\boldsymbol{x} = 0\Big\}. \tag{2.27}$$

Considering the training set $\tau = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)\}$, a separating hyperplane is such that the coefficient vector $(\beta_0, \boldsymbol{\beta}^T)$ satisfies

$$\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i > 0 \quad \text{if} \quad y_i = 1$$

and

$$\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i < 0 \quad \text{if} \quad y_i = -1$$

simultaneously. These together are equivalent to satisfying

$$y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i) > 0 \tag{2.28}$$

for all $i = 1, \ldots, N$. The classifier is hence given by the decision function g, where

$$g(\boldsymbol{x}) = \operatorname{sgn}\Big[\phi(\boldsymbol{x})\Big] = \operatorname{sgn}\Big[\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}\Big]. \tag{2.29}$$

Specifically, some novel observation $\boldsymbol{z}$ is allocated to class 1 if $\phi(\boldsymbol{z}) = \beta_0 + \boldsymbol{\beta}^T\boldsymbol{z}$ is positive, and to class -1 if negative. What we now hope to now is some notion of an "optimal" separating hyperplane. The hard-margin classifier, or *maximal margin classifier*, is one such intuitive choice. The maximal margin hyperplane is the one that maximises the minimal distance between itself and the observations in the training set, also referred to as the *margin*. This is depicted in Figure 2.7 with some simulated data for $p = 2$. The proximal data points are the p-dimensional *support vectors*, which form a subset of the observations that entirely define the maximal margin hyperplane. That is, insofar as the other data points remain either side of the margin, they would be free to move around without influencing the maximal margin hyperplane.

With this goal in mind, in a fashion inspired by [32], the optimisation problem for deriving the maximal margin classifier is formulated as

$$\begin{aligned} &\max_{\beta_0, \boldsymbol{\beta}} \quad D \\ &\text{subject to} \quad \frac{y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)}{\|\boldsymbol{\beta}\|_2} \geq D, \; i = 1, \ldots, N \end{aligned} \tag{2.30}$$

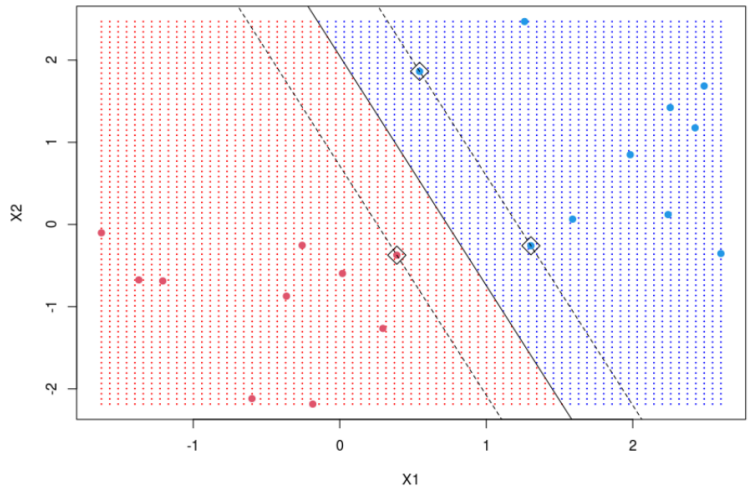


Figure 2.7: A linearly separable problem for $p = 2$. The squares highlight the support vectors. The solid line is the optimal hyperplane. The optimal margin is the distance between the two dotted lines, and solves the problem (2.30).

for a margin of length $2D$. We wish to impose a further constraint on $\|\boldsymbol{\beta}\|_2$ to act as a condition for uniqueness, since otherwise any scalar multiple of the coefficients would satisfy (2.30), yielding infinite possible solutions. The choice of constraint is chosen to ease computation: the conditions in (2.30) can be condensed by noticing that

$$\frac{y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)}{\|\boldsymbol{\beta}\|_2} \geq D \implies y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq D\|\boldsymbol{\beta}\|_2. \tag{2.31}$$

As mentioned, it can be seen that, for β_0 and $\boldsymbol{\beta}$ that satisfy (2.31), there is a family of infinitely many scaled solutions. This hence works when we choose to set $\|\boldsymbol{\beta}\|_2 = 1/D$, which can help to simplify things. It directly follows that (2.31) reduces to $y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq 1, i = 1, \ldots, N$. It then emerges that the problem in (2.30) is equivalent to

$$\begin{aligned} \min_{\beta_0, \boldsymbol{\beta}} \quad & \frac{\|\boldsymbol{\beta}\|_2^2}{2} \\ \text{subject to} \quad & y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq 1, \ i = 1, \ldots, N, \end{aligned} \tag{2.32}$$

where the factor of $1/2$ is for computational convenience as will be seen below. This minimisation problem is convex and can be optimised by defining the Lagrangian primal function, and eventually solving its simpler dual. With respect to β_0 and $\boldsymbol{\beta}$, the primal function to be minimised is given by

$$L(\lambda;\beta_0,\boldsymbol{\beta}) = \frac{\|\boldsymbol{\beta}\|_2^2}{2} - \sum_{i=1}^{N}\lambda_i\Big[y_i(\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i)-1\Big], \tag{2.33}$$

where the introduced λ_i are the N Lagrange multipliers induced by the conditions in (2.32). Allowing the partial derivatives to vanish yields

$$\begin{cases} \dfrac{\partial}{\partial\beta_0}L = -\displaystyle\sum_{i=1}^{N}\lambda_i y_i = 0 \\ \dfrac{\partial}{\partial\boldsymbol{\beta}}L = \boldsymbol{\beta} - \displaystyle\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i = 0 \end{cases} \Longrightarrow \begin{cases} 0 = \displaystyle\sum_{i=1}^{N}\lambda_i y_i \\ \boldsymbol{\beta} = \displaystyle\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i, \end{cases} \tag{2.34}$$

and substituting these into (2.33) gives us

$$\begin{aligned} L(\lambda;\beta_0) &= \frac{1}{2}\left(\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i^T \sum_{i=1}^{N}\lambda_j y_j \boldsymbol{x}_j\right) - \sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i^T \sum_{i=1}^{N}\lambda_j y_j \boldsymbol{x}_j - \beta_0 \underbrace{\sum_{i=1}^{N}\lambda_i y_i}_{=0} + \sum_{i=1}^{N}\lambda_i \\ &= -\frac{1}{2}\left(\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i^T \sum_{i=1}^{N}\lambda_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\lambda_i \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j + \sum_{i=1}^{N}\lambda_i. \end{aligned} \tag{2.35}$$

We have derived the *Wolfe dual* [38] to (2.33). The utility of the factor of $1/2$, then, is to avoid a cancellation of terms in the expansion of $L(\lambda;\beta_0)$. Now the primal problem can be equivalently solved by maximising (2.35) under the Karush-Kuhn-Tucker constraints. That is, subject to (2.34) and

$$\lambda_i\Big[y_i(\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i)-1\Big] = 0 \;\text{ and }\; \lambda_i \geq 0 \;\; \forall i. \tag{2.36}$$

It is immediate from (2.36) that $y_i(\beta_0+\boldsymbol{\beta}^T\boldsymbol{x}_i)-1=0$ must hold if $\lambda_i > 0$. This is the case for $\boldsymbol{x}_i$ on the boundary of the margin - that is, for the support vectors. The parameter estimates $\hat{\boldsymbol{\beta}} = \sum_{i=1}^{N}\hat{\lambda}_i y_i \boldsymbol{x}_i$ are hence determined entirely by the support

vectors. Similarly, the scalar $\hat{\beta}_0$ is found by solving (2.36) such that $\lambda_i > 0$ strictly for all i. As a result, for the set

$$V = \{i : \lambda_i > 0\}, \tag{2.37}$$

we find the maximal margin hyperplane to be

$$\begin{aligned}\hat{\phi}(\boldsymbol{x}) = \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T\boldsymbol{x} = \hat{\beta}_0 + \boldsymbol{x}^T\sum_{i=1}^{N}\hat{\lambda}_i y_i \boldsymbol{x}_i \\ = \hat{\beta}_0 + \sum_{i=1}^{N}\hat{\lambda}_i y_i \boldsymbol{x}^T\boldsymbol{x}_i = \hat{\beta}_0 + \sum_{i\in V}\hat{\lambda}_i y_i \boldsymbol{x}^T\boldsymbol{x}_i.\end{aligned} \tag{2.38}$$

Then, the class of some novel observation $\boldsymbol{z}$ is dictated by the sign of $\hat{\phi}(\boldsymbol{z})$. That is, given $(\hat{\lambda}_i; \hat{\beta}_0, \hat{\boldsymbol{\beta}}^T)$, the classifier is defined by the decision function $\hat{g}(\boldsymbol{z}) = \operatorname{sgn}\big[\hat{\phi}(\boldsymbol{z})\big]$. More specifically, the confidence that a particular observation has been classified correctly can be indicated by the magnitude of $|\hat{\phi}(\boldsymbol{z})|$, as this corresponds to $\boldsymbol{z}$ being as distant from the margin as possible.

2.3.2 Soft-Margin Classifier

We have so far only examined the case of linearly separable data, where the maximal margin classifier performs well. In more typical problems, the classes overlap in the feature space and hence extensions to the maximal margin classifier are required. Where perfect separation cannot be attained, the *support vector classifier*, sometimes referred to the *soft margin classifier*, is one such natural extension. Roughly, the soft margin permits a number of misclassified observations up to a specified constant. The constant can be seen as a budget for the amount of violation of the margin that is deemed permissible.

Take $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_N)^T$ to be slack variables, where ζ_i characterises the distance from the misclassified ith observation to the margin border on the *correct* side. This is visualised in Figure 2.8 with some simulated data for $p = 2$. This gives us the location of the ith location relative both the margin and the hyperplane [25]. A correctly classified observation corresponds to $\zeta_i = 0$. The ith observation violates the margin if $0 < \zeta_i \leq 1$. Finally, the misclassification distance exceeds the width of the margin if $\zeta_i > 1$, with the ith observation lying on the incorrect side of the hyperplane. By constraining the sum of the slack variables, we also constrain the total proportional distance that observations fall on the incorrect side of *their* margin by. This can be formulated in a similar fashion to (2.32) as

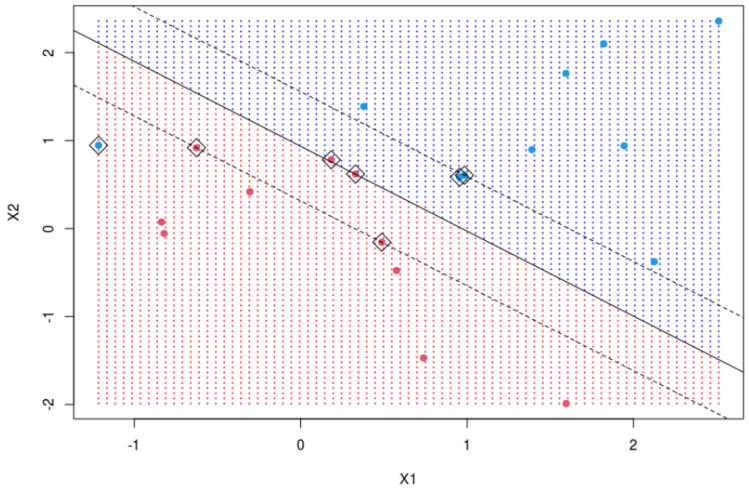


Figure 2.8: A non-linearly separable problem for $p = 2$. The grey squares highlight the support vectors. The soft margin solves the problem (2.39).

$$\begin{aligned} \min_{\beta_0,\boldsymbol{\beta}} \quad & \frac{\|\boldsymbol{\beta}\|_2^2}{2} \\ \text{subject to} \quad & y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq 1 - \zeta_i, \\ & \zeta_i \geq 0, \\ & \sum_{i=1}^{N} \zeta_i \leq A, \; i = 1, \ldots, N, \end{aligned} \tag{2.39}$$

where the constant A serves as a budget for the permissible magnitude of the slack variables. This becomes clearer upon reformulation to

$$\begin{aligned} \min_{\beta_0,\boldsymbol{\beta}} \quad & \frac{\|\boldsymbol{\beta}\|_2^2}{2} + K \sum_{i=1}^{N} \zeta_i \\ \text{subject to} \quad & y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \geq 1 - \zeta_i, \\ & \zeta_i \geq 0, \; i = 1, \ldots, N, \end{aligned} \tag{2.40}$$

where the "cost" K is a regularisation parameter, and replaces the function of constant A in (2.39). Suitable values for K are typically determined by cross-validation. Generally, a greater K demands more penalisation of the slack variables and hence gives rise to a "harder" margin. Indeed, (2.40) approaches the maximal margin problem for $K \to \infty$. As in the maximal margin case, this is again a convex problem that can be optimised by defining the Lagrangian primal function, and eventually solved via its simpler dual function. With respect to β_0, $\boldsymbol{\beta}$ and ζ_i, the primal function to be minimised is given by

$$L(\lambda, \eta; \beta_0, \boldsymbol{\beta}, \zeta) = \frac{\|\boldsymbol{\beta}\|_2^2}{2} + K\sum_{i=1}^{N}\zeta_i - \sum_{i=1}^{N}\lambda_i\Big[y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i) - (1-\zeta_i)\Big] - \sum_{i=1}^{N}\eta_i\zeta_i, \tag{2.41}$$

where the introduced λ_i and η_i are the Lagrange multipliers induced by the conditions in (2.40). Allowing the partial derivatives to vanish yields

$$\begin{cases} \dfrac{\partial}{\partial\beta_0}L = -\displaystyle\sum_{i=1}^{N}\lambda_i y_i = 0 \\ \dfrac{\partial}{\partial\boldsymbol{\beta}}L = \boldsymbol{\beta} - \displaystyle\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i = 0 \\ \dfrac{\partial}{\partial\zeta_i}L = K - \lambda_i - \eta_i = 0 \end{cases} \implies \begin{cases} 0 = \displaystyle\sum_{i=1}^{N}\lambda_i y_i \\ \boldsymbol{\beta} = \displaystyle\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i \\ \lambda_i = K - \eta_i, \forall i. \end{cases} \tag{2.42}$$

Substituting these into (2.41) gives us

$$\begin{aligned} L(\lambda, \eta; \beta_0, \zeta) &= \frac{1}{2}\left(\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i^T \sum_{i=1}^{N}\lambda_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\zeta_i(\lambda_i + \eta_i) - \sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i^T \sum_{i=1}^{N}\lambda_j y_j \boldsymbol{x}_j \\ &\qquad - \beta_0 \underbrace{\sum_{i=1}^{N}\lambda_i y_i}_{=0} + \sum_{i=1}^{N}(1-\zeta_i)\lambda_i - \sum_{i=1}^{N}\zeta_i\eta_i \\ &= -\frac{1}{2}\left(\sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i^T \sum_{i=1}^{N}\lambda_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\lambda_i \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j + \sum_{i=1}^{N}\lambda_i. \end{aligned} \tag{2.43}$$

Analogously to the maximal margin case, we have derived the Wolfe dual to (2.41). Now the primal problem can be equivalently solved by maximising (2.43) under the corresponding Karush-Kuhn-Tucker constraints. That is, subject to (2.42) and

$$\begin{aligned} \lambda_i\Big[y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i) - (1-\zeta_i)\Big] &= 0, \\ \eta_i\zeta_i &= 0, \\ y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i) - (1-\zeta_i) &\geq 0, \\ 0 \leq \lambda_i &\leq K, \\ \text{and } \ \lambda_i, \eta_i &\geq 0, \ \ i = 1, \ldots, N. \end{aligned} \tag{2.44}$$

Similarly to the previous case, it follows directly from the joint constraints (2.44) that $y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) - (1 - \zeta_i) = 0$ must hold if $\lambda_i > 0$. Again, these points correspond to the support vectors. Here, this is the case for $\boldsymbol{x}_i$ on the boundary of the margin, $\boldsymbol{x}_i$ that violate the margin, and $\boldsymbol{x}_i$ that breach the hyperplane entirely. The parameter estimates take the familiar form $\hat{\boldsymbol{\beta}} = \sum_{i=1}^{N} \hat{\lambda}_i y_i \boldsymbol{x}_i$, and $\hat{\beta}_0$ is found in an analogous fashion to that of the maximal margin classifier. The solutions to $\lambda_i > 0$ are all that are required to fully describe the support vector classifier, which is yielded through

$$\begin{aligned} \hat{\phi}(\boldsymbol{x}) = \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T \boldsymbol{x} = \hat{\beta}_0 + \boldsymbol{x}^T \sum_{i=1}^{N} \hat{\lambda}_i y_i \boldsymbol{x}_i \\ = \hat{\beta}_0 + \sum_{i=1}^{N} \hat{\lambda}_i y_i \boldsymbol{x}^T \boldsymbol{x}_i = \hat{\beta}_0 + \sum_{i \in V} \hat{\lambda}_i y_i \boldsymbol{x}^T \boldsymbol{x}_i. \end{aligned} \tag{2.45}$$

where V is the set of indices corresponding to the support vectors as defined in (2.37). Then, the class of some novel observation $\boldsymbol{z}$ is dictated by the sign of $\hat{\phi}(\boldsymbol{z})$. That is, given $(\hat{\lambda}_i; \hat{\beta}_0, \hat{\boldsymbol{\beta}}^T)$, the support vector classifier is defined by the decision function $\hat{g}(\boldsymbol{z}) = \operatorname{sgn}\big[\hat{\phi}(\boldsymbol{z})\big]$.

2.3.3 SVM Kernels and Hilbert Spaces

In most applications, attempting to separate classes with some linear boundary does not lead to a classifier of optimal performance. We want to obtain a decision boundary that can identify classes split in a nonlinear fashion in the original p-dimensional feature space. Roughly speaking, projecting the data to some higher-dimensional feature space should give rise to arbitrary data separation. This leads to a nonlinear decision boundary in the original feature space upon the inverse transformation. This idea originates from *Cover's theorem* [9], which roughly states that linear separation can be obtained via some nonlinear transformation with high probability.

We provide an illustrative example in Figure 2.9: the binary data in the original feature space clearly cannot be separated by a straight line (one-dimensional hyperplane). Upon addition of another variable defined by $X_3 = X_1^2 + X_2^2$, perfect separation can be achieved via a linear decision boundary; that is, by a flat plane (two-dimensional hyperplane). Reducing both the data and the boundary back to the original feature space yields a nonlinear decision boundary that achieves perfect separation of the two classes.

In the first instance, basis functions are utilised to project the original predictor variables $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^T$ to the higher feature space. The basis functions characterise a vector of transformed predictors $\boldsymbol{k}(\boldsymbol{x}_i) = (k_1(\boldsymbol{x}_i), \ldots, k_s(\boldsymbol{x}_i))^T$. These represent the original predictors mapped to a new s-dimensional space such that $s \geq p$. The procedure of obtaining the classifier is then identical to before; the parameter estimates and the optimal hyperplane are computed in a similar fashion to (2.45), with

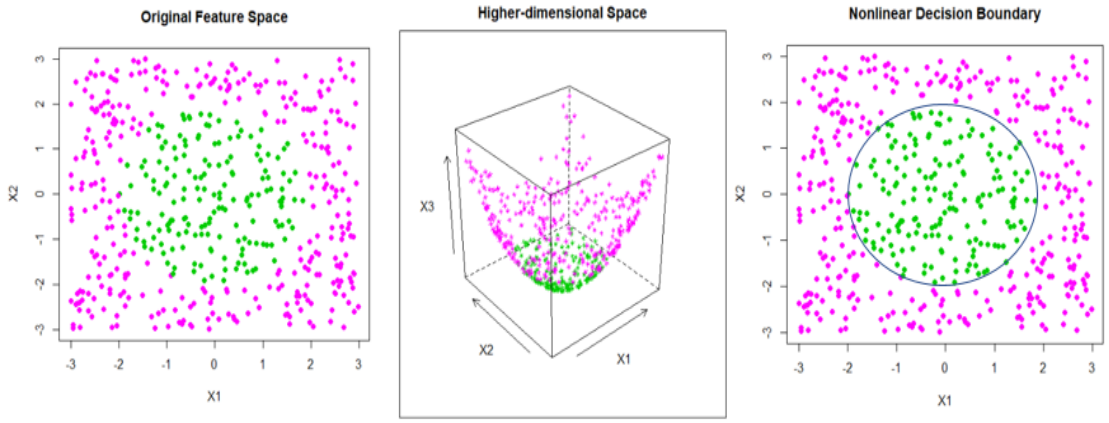


Figure 2.9: *Left:* Binary data in the initial two-dimensional space. *Middle:* Binary data in the expanded three-dimensional space. *Right:* Decision boundary from expanded space reduced down to the initial space.

$$\hat{\boldsymbol{\beta}} = \sum_{i=1}^{N} \hat{\lambda}_i y_i \boldsymbol{k}(\boldsymbol{x}_i) \tag{2.46}$$

and

$$\hat{\phi}(\boldsymbol{x}) = \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T \boldsymbol{k}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^{N} \hat{\lambda}_i y_i \boldsymbol{k}(\boldsymbol{x})^T \boldsymbol{k}(\boldsymbol{x}_i). \tag{2.47}$$

Then, for some novel observation $\boldsymbol{z}$, the classifier is defined by the decision function $\hat{g}(\boldsymbol{z}) = \operatorname{sgn}\big[\hat{\phi}(\boldsymbol{z})\big] = \operatorname{sgn}\big[\hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T \boldsymbol{k}(\boldsymbol{z})\big]$.

More generally, as there exists no upper limit to the dimension of the enlarged feature space, calculations can quickly face intractability. The computational complexity of arbitrarily searching for a suitable nonlinear transformation can be streamlined via the use of *kernels*. Indeed, extending the classifier above to incorporate kernel technology is precisely how we arrive at the flexible *support vector machine* (SVM). Note that the $\boldsymbol{k}(\boldsymbol{x})^T \boldsymbol{k}(\boldsymbol{x}_i)$ term in (2.47) is simply alternate notation for the dot product. The idea of kernel functions uses the *Hilbert-Schmidt theorem*, or eigenfunction expansion theorem, to generalise the notion of the dot product [6]. Formally, a kernel function $\mathcal{K}(\boldsymbol{s}, \boldsymbol{t}) = \big\langle \boldsymbol{k}(\boldsymbol{s}), \boldsymbol{k}(\boldsymbol{t}) \big\rangle$ must be symmetric positive (semi-) definite, and fulfil *Mercer's condition*

$$\int \int \psi(\boldsymbol{s}) \mathcal{K}(\boldsymbol{s}, \boldsymbol{t}) \psi(\boldsymbol{t}) \, d\boldsymbol{s} \, d\boldsymbol{t} \geq 0 \tag{2.48}$$

for all square-integrable ψ [35]. The space of functions $\mathcal{H}_{\mathcal{K}}$ generated by the kernel $\mathcal{K}$ is coined a *reproducing kernel Hilbert space* (RKHS). Further details of the relation between the RKHS and Mercer's theorem are provided in [6]. Though also beyond the scope of this thesis, for a discussion of the eigenfunction expansion of kernels, and for an interesting allusion to a yet wider class of regularisation problems, we point the reader to [21].

With reference to the Lagrange problem in (2.43), the generalised Wolfe dual now looks like

$$\begin{aligned} L &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \left\langle \boldsymbol{k}(\boldsymbol{x}_i), \boldsymbol{k}(\boldsymbol{x}_j)\right\rangle + \sum_{i=1}^{N}\lambda_i \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \, \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_{i=1}^{N}\lambda_i, \end{aligned} \tag{2.49}$$

with solution

$$\hat{\phi}(\boldsymbol{x}) = \beta_0 + \hat{\boldsymbol{\beta}}^T \boldsymbol{k}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^{N}\hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i). \tag{2.50}$$

where $\hat{\beta}_0$ and $\hat{\lambda}_i$ are found in the usual way. Specifically, one should notice that both (2.49) and (2.50) depend on the transformed variables $\boldsymbol{k}$ only through the kernel function $\mathcal{K}$. This is what gives kernels superior computational efficiency over arbitrarily expanding the feature space, since explicit transformations need not be calculated. One only requires knowledge of $\mathcal{K}$ - that is, the inner products in transformed space - hence computations remain tractable. Typical forms of $\mathcal{K}$ in practice include

$$\begin{aligned} \text{Linear:} \quad & \mathcal{K}(\boldsymbol{s}, \boldsymbol{t}) = \boldsymbol{s}^T\boldsymbol{t}, \\ d\text{th-Degree polynomial:} \quad & \mathcal{K}(\boldsymbol{s}, \boldsymbol{t} = \left(1 + \left\langle \boldsymbol{s}, \boldsymbol{t}\right\rangle\right)^d, \\ \text{Radial basis:} \quad & \mathcal{K}(\boldsymbol{s}, \boldsymbol{t}) = \exp\left(-\gamma\|\boldsymbol{s} - \boldsymbol{t}\|\right)^2. \end{aligned}$$

It can be seen from comparison of (2.50) to (2.45) that the SVM reduces to the support vector classifier when assigned a linear kernel. In fact, it also turns out that the the introduction of a new variable in Figure 2.9 is equivalent to employing a polynomial kernel of degree $d = 2$. In Figure 2.10, we have visualised how the SVM with different kernel functions attempts to classify the binary `mixture` dataset, which can be found in the `ElemStatLearn` package. The SVM decision boundaries are represented by the black curves, and should be compared to the blue curve, or *Bayes decision boundary*. The corresponding Bayes classifier gives the exemplar discriminatory standard, since it uses knowledge of the Y given $\boldsymbol{X}$ conditional distribution, which is unavailable in more

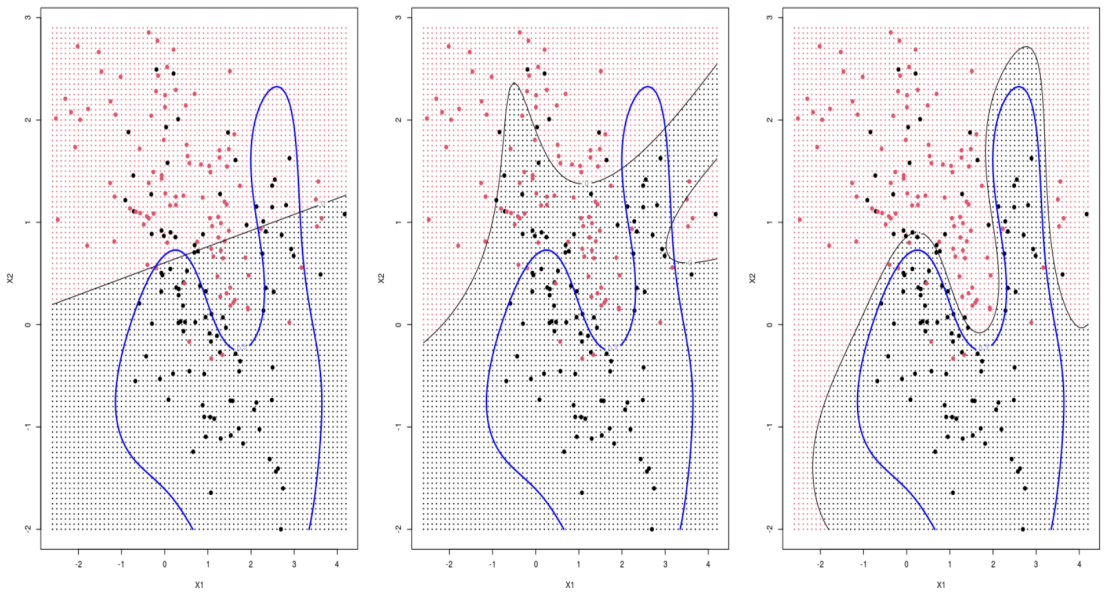


Figure 2.10: Decision boundaries given by SVM classifiers for `mixture` data, with cost parameter $K = 5$. *Left:* Linear kernel. *Middle:* Polynomial kernel with $d = 5$. *Right:* Radial basis kernel. The blue curve is the Bayes decision boundary.

general problems. In this instance, it is clear that the radial basis kernel exhibits superior performance; that is, it performs most closely to the Bayes optimal case. In our analysis of real datasets in Chapter 4, we limit our base SVM model candidates to those with linear kernels and radial basis kernels.

2.3.4 The SVM as a Penalisation Method

The next two subsections see the SVM reformulated as a regularisation problem, an insight which, as alluded to in the LR case, may significantly improve variance for a marginal hike in bias. We transiently set aside the addition of kernel functions for expository convenience, but one should note that this technology can be easily made present through ϕ. For now, take $\phi(\boldsymbol{x}) = \beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}$. According to [25], the support vector classifier criteria given in (2.39), which can take transformed variables as inputs, can be rewritten in an equivalent form as

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \Big[1 - y_i\phi(\boldsymbol{x}_i)\Big]_+ + \lambda\|\boldsymbol{\beta}\|_2^2 \right\}, \tag{2.51}$$

where $[1 - y_i\phi(\boldsymbol{x}_i)]_+ = \max\{0, 1 - y_i\phi(\boldsymbol{x}_i)\}$ is the *hinge* loss function, and $\lambda > 0$ is the familiar tuning parameter that dictates the influence of the regularisation term on the optimisation problem. Practically, reducing λ has an impact comparable to increasing

the cost parameter K. Indeed, a small λ corresponds to lesser weight being placed on the minimisation of $\boldsymbol{\beta}$, and observations that violate the margin are treated more stringently; that is, a "harder" margin results.

2.3.5 Sparsity-inducing Norms for SVMs

Courtesy of the computational complexity of SVMs sitting at $\mathcal{O}(N^2 v)$, where v is the cardinality of the set V in (2.37) - that is, the number of support vectors - SVMs are popular tools in the literature for classification problems involving high-dimensional data ($p \gg N$) [22]. Notably, though, in their current form they are unable to achieve sparsity in the variables by shrinking some entirely to zero. Substituting in an l_1 lasso penalty for the l_2 ridge penalty in (2.51) yields

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \Big[1 - y_i \phi(\boldsymbol{x}_i)\Big]_+ + \lambda \|\boldsymbol{\beta}\|_1 \right\}, \tag{2.52}$$

which is the l_1-regularised SVM, as originally introduced in [45]. Though the l_1 SVM provides a prudent extension to the l_2 alternative when feature selection is desirable, the coefficient estimates $\hat{\boldsymbol{\beta}}$ are subject to a kind of penalisation bias. Specifically, larger estimates are regularised more harshly. In an attempt to overcome this excessive regularisation, the *smoothly clipped absolute deviation* (SCAD) penalty was proposed by [14], and subsequently appended to SVMs in [40]. The objective function corresponding to the SCAD SVM is given by

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \Big[1 - y_i \phi(\boldsymbol{x}_i)\Big]_+ + \sum_{j=1}^{p} P_\lambda\big(|\beta_j|\big) \right\}, \tag{2.53}$$

where

$$P_\lambda\big(|\beta_j|\big) = \begin{cases} \lambda|\beta_j| & \text{if } |\beta_j| \leq \lambda, \\ -\dfrac{\big(|\beta_j|^2 - 2b\lambda|\beta_j| + \lambda^2\big)}{2(b-1)} & \text{if } \lambda < |\beta_j| \lambda, \\ \dfrac{(b+1)\lambda^2}{2} & \text{if } |\beta_j| > b\lambda, \end{cases} \tag{2.54}$$

with tuning parameters $b > 2$ and $\lambda > 0$, typically obtained via cross-validation. Figure 2.11 provides a plot comparison of the l_1, l_2 and SCAD penalties, which can aid in understanding their respective behaviour. A necessary criterion for a regularisation

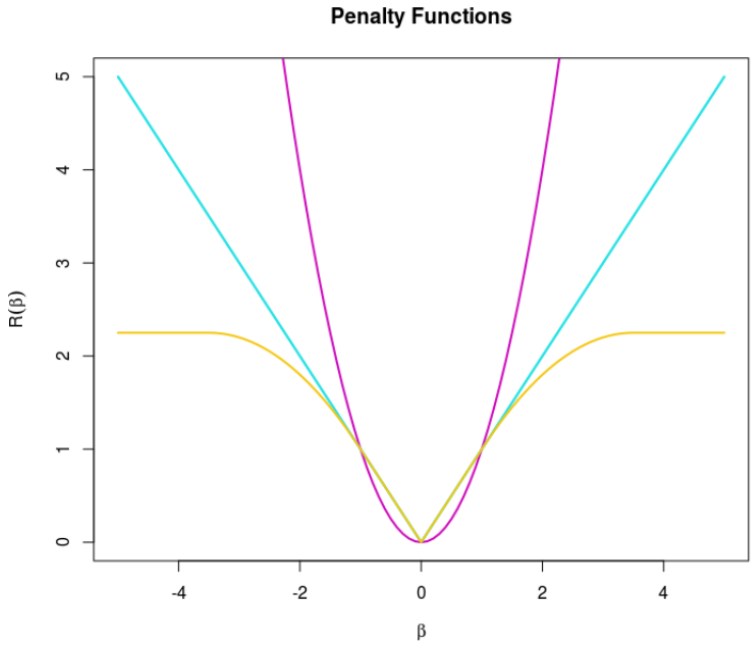


Figure 2.11: Penalty functions for simulated data: l_1 (blue), l_2 (purple) and SCAD with $b = 3.5$ and $\lambda = 1$ (orange).

function to yield sparsity is for it to be non-differentiable at the origin [40]. This rationalises why the base SVM problem in (2.51) fails to select for specific features. By contrast, both the l_1 and SCAD functions satisfy this condition and hence take alike forms in a neighbourhood of the origin. In particular, whereas the l_1 function grows linearly as the β_j increase, the SCAD imposes a constant penalty for large enough β_j. Roughly, this characteristic is what protects SCAD from generating biases in favour of larger parameter estimates.

Further, [22] advise against the use of the l_1-regularised SVM for feature selection, since the related lasso LR model in (2.18) exhibits comparable performance, as will be seen in the following section, whilst possessing superior algorithmic stability. As a result, we focus on the SCAD penalty for the sparse SVM model to be analysed in Chapter 4.

Somewhat surprisingly, despite their obvious lesser flexibility, linear classifiers can frequently offer superior performance to their nonlinear cousins in cases where $p \gg N$ [21]. This follows from some results in [19] about the asymptotic structure of data where $p \gg N$. As $p \to \infty$, the geometry of the data are such that each vector can be described deterministically as a vertex of a regular N-dimensional simplex. In short, any stochastic properties of the data can be described as a random simplex rotation. Then, in cases where the data is binary, linear decision boundaries perform well in separating two such simplices [2]. As our ultimate aim is to classify low-sample, high-dimensional data, the SCAD-penalised SVM used in this thesis is limited to a linear kernel. That is, $s = p$ and

$\boldsymbol{k}(\boldsymbol{x}) = \boldsymbol{x}$. The linear SCAD SVM is one of our candidate models in Chapter 4, and the expectation is for it to perform well in cases where $p \gg N$.

2.4 Relating the Classifiers

Although originally thought to boast a novel underlying framework upon their inception, SVMs have since been shown to possess deep ties with more traditional statistical tools, such as the penalised LR we have discussed in this thesis. This section briefly outlines a general form that encompasses both SVMs and LR, hence exposing their similarities as classification methods.

2.4.1 Comparison of the Loss Functions

Recalling all discussed variations of regularised SVM problems in (2.51), (2.52) and (2.53), it should be clear that they each take a "Loss + Regularisation" form that has appeared frequently in this thesis. Explicitly, they are members of a family of generalised regularisation problems with the optimisation problem

$$\min_{\beta_0,\boldsymbol{\beta}} \Big\{ \mathcal{J}(y, \phi(\boldsymbol{x})) + \lambda \mathcal{R}(\boldsymbol{\beta}) \Big\}, \tag{2.55}$$

where the SVM employs the hinge loss function

$$\mathcal{J}(y, \phi(\boldsymbol{x})) = \sum_{i=1}^{N} [1 - y_i \, \phi(\boldsymbol{x}_i)]_+. \tag{2.56}$$

Crucially, upon inspection of the regularised log-likelihood function in (2.23), which compactly encompassed all our versions of penalised LR, we see that this again is just a specific instance of (2.55) where the loss function is the binomial log-likelihood, or *binomial deviance*

$$\mathcal{J}(y, \phi(\boldsymbol{x})) = \sum_{i=1}^{N} \Big\{ \log\Big(1 + e^{\phi(\boldsymbol{x}_i)}\Big) - y_i \phi(\boldsymbol{x}_i) \Big\}. \tag{2.57}$$

It appears, then, that these two classification methods have much more in common than it may first seem. As it turns out, even though they each enjoy their own respective loss functions, the binomial deviance and the hinge loss share numerous similarities. They take on similar shapes, as depicted in Figure 2.12, which intuitively explains the performance similarity often cited between LR and SVM models [25]. For instance, this claim is supported by [22], who overlay the coefficient paths yielded by lasso LR and l_1-regularised SVM for both narrow data ($p < N$) and wide data ($p \gg N$), and report the similarity to be both "strong" and "striking".

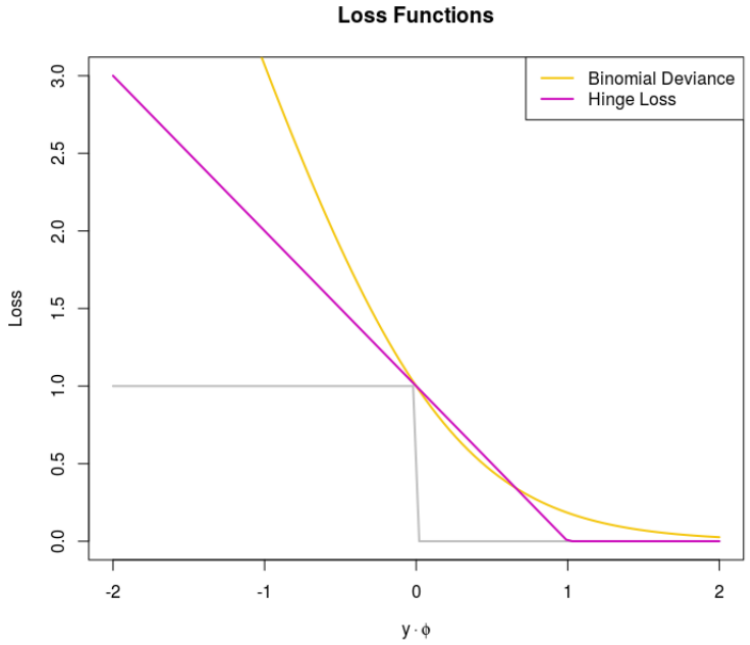


Figure 2.12: Binary classification loss functions, scaled to pass through $(0, 1)$, with response $y \in \{-1, 1\}$ and class prediction $\text{sgn}(\phi)$. The misclassification loss $\mathbb{1}\{\text{sgn}(\phi) \neq y\}$ is in grey, alongside the LR binomial deviance and SVM hinge loss.

Interestingly, the elastic-net LR problem in (2.21) was rigorously proved to reduce to the instance of the linear SVM by [42] in 2015. In short, it was proved that a separating hyperplane problem with binary classes could be artificially formulated for all occurrences of the elastic-net, including in LR. The elastic-net LR coefficient estimates then coincide with the linear SVM hyperplane solution, up to some scalar multiple. This followed a similar equivalence being established for the lasso by [24], where the latter has been seen to be a particular instance of the former. Explicit details of the derivations exceed the scope of this thesis, but we encourage the curious reader to visit the respective cited papers from which they originate.

2.4.2 Kernelised Logistic Regression (KLR)

It also transpires that SVMs do not have a monopoly on the incorporation of kernel functions, and that kernel technology can be appended to LR models, further bridging the technical differences between the classification methods. As mentioned earlier, though the addition of kernel functions may not seem notationally explicit, one should note that this technology is still present through ϕ. This time, take $\phi(\boldsymbol{x}) = \beta_0 + \boldsymbol{\beta}^T \boldsymbol{k}(\boldsymbol{x})$, where the $\boldsymbol{k}(\boldsymbol{x}) = (k_1(\boldsymbol{x}), \ldots, k_s(\boldsymbol{x}))^T$ are the basis functions of the RKHS $\mathcal{H}_\mathcal{K}$ induced by the desired kernel $\mathcal{K}(\cdot, \cdot)$ that allow for the enlarged feature space. Then, naturally modifying the notation in [43] to its more general form gives the optimisation problem

$$\min_{\phi \in \mathcal{H}_\mathcal{K}} \Big\{ \mathcal{J}(y, \phi(\boldsymbol{x})) + \lambda \|\phi\|^2_{\mathcal{H}_\mathcal{K}} \Big\}, \tag{2.58}$$

where using the hinge loss (2.56) fits a kernel SVM, and using the binomial deviance (2.57) fits a *kernel logistic regression* (KLR) model. One distinct advantage of the KLR is that, similarly to its non-kernelised sister, it is able to output explicit class probability estimates of

$$p(\boldsymbol{x}) = \mathbb{P}(Y = 1 | \boldsymbol{X} = \boldsymbol{x}) = \frac{e^{\phi(\boldsymbol{x})}}{1 + e^{\phi(\boldsymbol{x})}} = \frac{e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{k}(\boldsymbol{x})}}{1 + e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{k}(\boldsymbol{x})}} \tag{2.59}$$

similarly to (2.8). Conversely, the SVM acts merely a classification rule through $\text{sgn}[p(\boldsymbol{x}) - 1/2]$. The probability estimates can be intuitively arrived at as follows: we know the fitted solution to a kernel problem takes the form (2.50), which we restate as

$$\hat{\phi}(\boldsymbol{x}) = \beta_0 + \hat{\boldsymbol{\beta}}^T \boldsymbol{k}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^{N} \hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i). \tag{2.60}$$

Then, substituting (2.60) into (2.59) yields the estimates

$$\begin{aligned} \hat{p}(\boldsymbol{x}) = \hat{\mathbb{P}}(Y = 1 | \boldsymbol{X} = \boldsymbol{x}) &= \frac{\exp\Big\{\hat{\beta}_0 + \sum_{i=1}^{N} \hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i)\Big\}}{1 + \exp\Big\{\hat{\beta}_0 + \sum_{i=1}^{N} \hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i)\Big\}} \\ &= \frac{1}{1 + \exp\Big\{ - \hat{\beta}_0 - \sum_{i=1}^{N} \hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i)\Big\}}. \end{aligned} \tag{2.61}$$

Given this inferential advantage, why is KLR not frequently employed in practise? One reason may be that the computational complexity of the KLR sits at $\mathcal{O}(N^3)$, in contrast to the less expensive $\mathcal{O}(N^2 v)$ for the SVM, where v is the number of support vectors [44]. Another may be that, for reasons already established, the SVM classifier is fully determined by these support vectors. This is courtesy of the hinge loss function; re-examination of Figure 2.12 shows how the support vectors (observations such that $y_i \, \phi(\boldsymbol{x}_i) \geq 1$) incur a loss of exactly zero. By contrast, the binomial deviance shrinks for observations far from the decision boundary, but is nowhere exactly zero [25]. That is, all the λ_i's in (2.60) are nonzero. This means it cannot exploit data compression for algorithmic efficiency the same way SVMs are able. Further, it could also simply be that the SVM boasts greater popularity due to its appeal as a more modern tool.

For these reasons, this subsection provides just an allusion to how general the theoretical LR model can be made to be. The natural question that now arises concerns that of empirical performance. Unfortunately, the KLR is notably absent from our analysis in Chapter 4 due to issues getting the `CVST` package to co-operate.

Chapter 3

Model Assessment and Statistical Inference

While the previous chapter examines the inner workings of the two main models that provide our focus, we must look to how their relative classification power can be explicitly assessed. Inevitably, the estimated function $\hat{\phi}$ will conflict with the true model ϕ. In particular, we consider empirical resampling methods to best measure this discrepancy. Courtesy of [10], we see how they sit within a unified bias-variance decomposition framework.

3.1 Error Estimation

Recall that the underlying true model we wish to estimate takes the form $Y = \phi(\mathbf{X}) + \epsilon$, upon the assumptions that the random noise is such that $\mathrm{Var}(\epsilon) = \sigma_\epsilon^2$ and, as before, $\mathrm{E}(\epsilon) = 0$. In the general setting, initially following the ideas from [21], the *training error*, with respect to a suitable loss function $\mathcal{J}(\cdot, \cdot)$, is given by

$$\bar{e} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{J}(y_i, \hat{\phi}(\boldsymbol{x}_i)) \tag{3.1}$$

for a training set $\tau = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$. This gives the average loss specifically over the training set from which the model was trained. It is intuitive that this typically underestimates the error that would be incurred when the classifier is applied to novel data. According to [21], the training error drops directly with model complexity, and potentially down to zero. Yet, this is indicative of a model clearly overfit to the training set, and hence one that will not generalise well to novel data.

A more appropriate measure is given by the generalisation error, or *test error*, which gives the error in prediction over some test set. In practise, the test set is often obtained by splitting the original τ into a reduced training set $\tau_1 = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}$, on

which the model is trained, and the test set $\tau_2 = \{(\boldsymbol{x}_{l+1}, y_{l+1}), \ldots, (\boldsymbol{x}_N, y_N)\}$, on which the test error is calculated. The test error is then obtainable via the quantity

$$\mathcal{E}_\tau = \mathrm{E}\Big[\mathcal{J}(Y_{\tau_2}, \hat{\phi}(\boldsymbol{X}_{\tau_2}))\Big|\tau_1\Big], \tag{3.2}$$

where the τ_2 subscripts represent the entries of the original vector Y and matrix $\boldsymbol{X}$ that correspond to the test set τ_2. It is important to note that, since τ_1 is fixed, the test error is specifically with respect to this particular training set. More generally, averaging over all stochastic components of (3.2) yields the *expected test error*

$$\mathcal{E} = \mathrm{E}[\mathcal{E}_\tau] = \mathrm{E}\Big[\mathcal{J}(Y_{\tau_2}, \hat{\phi}(\boldsymbol{X}_{\tau_2}))\Big]. \tag{3.3}$$

Although a good measure of error for a particular test set is one that approximates (3.2) well for the estimated model $\hat{\phi}$, we note that the *resampling methods* we visit shortly are better suited to estimating the more fundamental quantity (3.3).

In our classification setting, we are interested in quantifying the error explicitly in terms of the estimated decision function $\hat{g} = \mathrm{sgn}[\hat{\phi}]$. In this case, the appropriate loss function for error estimation is typically chosen to be the misclassification loss, or *zero-one loss*, as given by

$$\mathcal{J}(Y, \hat{g}(\boldsymbol{X})) = \mathbb{1}\big\{\hat{g}(\boldsymbol{X}) \neq Y\big\}, \tag{3.4}$$

which was also plotted in Figure 2.12 in grey. This gives rise to the training error rate

$$\bar{e} = \frac{1}{l}\sum_{i=1}^{l} \mathbb{1}\big\{\hat{g}(\boldsymbol{x}_i) \neq y_i\big\} \tag{3.5}$$

for $(\boldsymbol{x}_i, y_i) \in \tau_1$, and the test error rate

$$\mathcal{E}_\tau = \frac{1}{N-l}\sum_{i=l+1}^{N} \mathbb{1}\big\{\hat{g}(\boldsymbol{x}_i) \neq y_i\big\} \tag{3.6}$$

for $(\boldsymbol{x}_i, y_i) \in \tau_2$.

Our aim is to estimate $\mathcal{E} = \mathrm{E}[\mathcal{E}_\tau]$, the expected test error of $\hat{g}$. With greater complexity, the estimated model is more flexibly able to describe intricate behaviours in the training set. Hence, with model complexity comes an associated hike in variance and drop in bias. It turns out that the interplay between these two competing properties can be represented explicitly.

3.1.1 The Bias-Variance Dichotomy

The decomposition of the expected test error into its constituent bias and variance terms is well known in the statistics literature, especially with regards to the square loss function, for which it was originally developed. Unfortunately, for classification problems where the zero-one loss is employed, the decomposition does not extend automatically [37]. This is because the variance and the bias are not strictly additive quantities. Since our concern is with binary classification, we outline a "unified bias-variance decomposition" as introduced in [10].

In order to do this, two terms must first be defined, upon which the bias and variance are dependent. Namely, the *optimal prediction* and the *main prediction*. We represent this generally for response variable Y and novel test data $\boldsymbol{z}$. The *optimal prediction* is given by

$$g^*(\boldsymbol{z}) = \underset{\hat{g}}{\operatorname{argmin}} \, \mathrm{E}_Y \Big[\mathcal{J}(Y, \hat{g}(\boldsymbol{z})) \Big], \tag{3.7}$$

where the expectation is performed over each value of Y, weighted by their respective probabilities given $\boldsymbol{z}$. The *optimal model* is then that which yields the optimal prediction for every $\boldsymbol{z}$; that is, $\hat{g}(\boldsymbol{z}) = g^*(\boldsymbol{z}) \; \forall \boldsymbol{z}$. For the zero-one loss function, this coincides with the Bayes classifier that was mentioned briefly in Section 2.3.3. Since the expected test error averages over different fixed training sets, take T to be a set of training sets. The *main prediction* is then

$$Y_m = \underset{Y'}{\operatorname{argmin}} \, \mathrm{E}_T \Big[\mathcal{J}(Y', \hat{g}(\boldsymbol{z})) \Big], \tag{3.8}$$

where the expectation is now performed over each of the predictions $\hat{g}(\boldsymbol{z})$ yielded when each member of T is used to train the model. The problem (3.8) gives the Y' that corresponds to the minimum average loss with respect to each possible prediction $\hat{g}(\boldsymbol{z})$ from each training set, and is also known as the "systemic prediction". In the case of the zero-one loss function, this is the most frequent prediction class (the prediction mode). Now, given these two definitions, it is possible to break down the expected test error into bias, variance, and noise. The *bias*

$$B(\boldsymbol{z}) = \mathcal{J}(g^*(\boldsymbol{z}), Y_m) \tag{3.9}$$

is the loss from the main prediction with respect to the optimal prediction, and is always 0 or 1 for the zero-one loss function. The *variance*

$$V(\boldsymbol{z}) = \mathrm{E}_T \Big[\mathcal{J}(Y_m, \hat{g}(\boldsymbol{z})) \Big] \tag{3.10}$$

is the average loss of the predictions with respect to the main prediction. It characterises how susceptible the predictions $\hat{g}(\boldsymbol{z})$ are to a change in T. The *noise*

$$N(\boldsymbol{z}) = \mathrm{E}_Y \Big[\mathcal{J}(Y, g^*(\boldsymbol{z})) \Big] \tag{3.11}$$

captures the irreducible error that not even the optimal prediction can avoid; that is, the error that is present independently of the model trained. Given these quantities, it was proved in [10] that, for a general class of loss functions, the expected test error can be then decomposed as

$$\begin{aligned}\mathcal{E} = \mathrm{E}\Big[\mathcal{J}(Y, \hat{g}(\boldsymbol{z}))\Big] &= \mathcal{J}(g^*(\boldsymbol{z}), Y_m) + a_1 \mathrm{E}_T\Big[\mathcal{J}(Y_m, \hat{g}(\boldsymbol{z}))\Big] + a_2 \mathrm{E}_Y\Big[\mathcal{J}(Y, g^*(\boldsymbol{z}))\Big] \\ &= B(\boldsymbol{z}) + a_1 V(\boldsymbol{z}) + a_2 N(\boldsymbol{z}), \end{aligned} \tag{3.12}$$

with coefficients a_1 and a_2 taking on values relative to the specific loss function chosen by the modeller. In particular, in the case of binary classification with the zero-one loss, (3.12) is valid for multiplicative factors $a_1 = +1$ if $B(\boldsymbol{z}) = 0$ and -1 if $B(\boldsymbol{z}) = 1$, and $a_2 = 2\mathrm{P}_T\big[\hat{g}(\boldsymbol{z}) = g^*(\boldsymbol{z})\big] - 1$.

The bias-variance trade-off proves a very useful concept when considering shrinkage methods and model sparsity; shrinking coefficients towards zero incurs an extra estimation bias, courtesy of the model being less fit to the training set. In order to avoid overfitting to the training set and to promote generalisability, it is generally deemed desirable to continue to do this until the hike in bias exceeds the associated decrease in variance. One should also note that the trade-off in classification problems behaves in a somewhat peculiar manner. By nature of the zero-one loss function, the test error is not hurt by cases with considerable estimation errors but that still lead to correct classification. Simply put, there is room for manoeuvre insofar as points lie on the correct side of the decision boundary.

Having said this, the problem of tuning model parameters accordingly remains similar. As alluded to throughout Chapter 2, the value of parameters such as the λ in charge of regularisation or the SVM cost K, should be chosen to minimise the expected test error $\mathcal{E}$. This can be directly measured empirically through so-called resampling methods.

3.2 Resampling Methods and Parameter Tuning

Resampling methods make up a family of statistical tools that are able to provide further information about a proposed model. Generally, they entail drawing repeated samples from the training set τ_1, and refitting the proposed model on each of these samples [25]. Resampling methods are utilised in this case to directly and empirically estimate the expected test error, $\mathcal{E}$. Despite being computationally intensive, they prove indispensable in their theoretical simplicity; they generalise better than their traditional *information criteria* cousins in the literature, and also demand fewer assumptions. The most commonly employed of such methods are *cross-validation* and *the bootstrap*, the former of which we discuss here due to its use in Chapter 4.

3.2.1 Cross-Validation

As mentioned, cross-validation is conducted on the training set. Specifically, τ_1 is split repeatedly into a *training subset* τ_1^t, on which the model is trained, and a corresponding *validation subset* τ_1^v, with respect to which the error rate is then estimated. The general idea of cross-validation is to compute the *cross-validation error*, which takes the average of the error estimates, for each candidate value for the parameter in question. The parameter is typically chosen to take the value that yielded the lowest cross-validation error, and is said to have been "tuned".

One form of this procedure is k-fold cross-validation. To conduct this, τ_1 is split into k similarly-sized groups, or *folds*, denoted here by $H_1, \ldots, H_k$. The algorithm for computing the k-fold cross-validation error, in the case of binary classification, can be performed as follows:

(1) Take $\tau_1^t = \{H_1, \ldots, H_{j-1}, H_{j+1}, \ldots, H_k\}$ (jth fold left out) to train the model.

(2) For all $\boldsymbol{x} \in \tau_1^v = H_j$, predict class $\hat{g}(\boldsymbol{x})$.

(3) Compute the test error on this set:

$$\mathcal{E}_{\tau,j}(\hat{g}) = \frac{1}{m_j} \sum_{i=1}^{m_j} \mathbb{1}\{\hat{g}(\boldsymbol{x}_i) \neq y_i\}, \tag{3.13}$$

where $m_j = |H_j|$ is the cardinality of jth fold.

(4) For all $j = 1, \ldots, k$, repeat steps (1)-(3).

(5) Compute the k-fold cross-validation error

$$CV(\hat{g})_{(k)} = \frac{1}{k} \sum_{j=1}^{k} \mathcal{E}_{\tau,j}(\hat{g}). \tag{3.14}$$

(6) Terminate.

The model selection methodology for the classification models in this thesis require multiple parameters to be tuned. Generally, the value of some parameter δ is chosen such that the estimated test error

$$CV(\hat{g}, \delta)_{(k)} = \frac{1}{k} \sum_{j=1}^{k} \mathcal{E}_{\tau,j}(\hat{g}, \delta), \tag{3.15}$$

contingent on δ, is minimised. The tuned value $\hat{\delta}$ that achieves this is then employed in the training of the estimated final model, $\hat{g}(\boldsymbol{x}, \hat{\delta})$.

The choice of k must be made with respect to the bias-variance trade-off: a lower k is associated with lower variance but greater bias [25]. Empirically, using $k = 5$ or $k = 10$ has been found to strike a good compromise, since the test error rates will not be hurt by excessive variance nor excessive bias. In the following chapter, the model parameters are tuned via 5-fold cross-validation.

3.3 Model Performance Measures

The evaluation of a classification model requires a set of metrics against which different models can be contrasted. Already discussed is the misclassification rate (test error rate) given in (3.6). A well-performing classifier should exhibit a low misclassification rate; that is, have a high rate of *accuracy*. For our binary case, it is often meaningful to have a more granular breakdown of the prediction errors with respect to each specific class, especially in instances where the dataset is unbalanced. To see this, consider a dataset for which we wish to classify 90 boys (B) and 10 girls (G). The overall training error for a blind classifier that attributes every observation to class B would sit at 10%, which looks quite promising. Upon closer inspection, however, the error with respect to class G sits at 100%, suggesting poorer discriminatory power than perhaps originally thought.

More generally, take a binary classification problem where $Y \in \{0, 1\}$. Table 3.1 gives the corresponding 2-dimensional *confusion matrix*. Observations that belong to class 1 and are assigned correctly are called *true positive*, and those that are assigned incorrectly are *false negative*. Further, observations that belong to class 0 are *true negative* if they are classified as such, and *false positive* otherwise [15].

		Predicted class		
		1	0	Total
True class	1	True positive (T)	False negative (F')	$T + F'$
	0	False positive (F)	True negative (T')	$F + T'$
	Total	$T + F$	$F' + T'$	N

Table 3.1: Confusion matrix for a binary classifier.

From Table 3.1, the *specificity* is defined as $T'/(T'+F)$, and the *sensitivity* as $T/(T+F')$. To relate this to the terminology commonly found in statistical hypothesis testing, the specificity is equivalent to $1-$*Type I error*, where the Type I error is the proportion of observations in true class 0 being misclassified. The sensitivity is equivalent to $1-$*Type II error*, where the Type II error is the proportion of observations in true class 1 being misclassified.

The *receiver operating characteristic* (ROC) curve provides a more comprehensive performance metric than accuracy alone. It gives a graphical representation of the posterior

probabilities associated with identifying a true signal (sensitivity) against that of a false signal (1-specificity). This is done for a range of threshold values r, where observations are assigned to class 1 if $\mathbb{P}(Y = 1|\boldsymbol{X} = \boldsymbol{z}) \geq r$, and to class 0 otherwise, as discussed in Chapter 2. A model's discriminatory ability can then be assessed as a scalar quantity between 0 and 1 via the *area under the ROC curve* (AUC). Generally speaking, a model exhibiting a higher AUC can be interpreted as one which better balances the trade-off between sensitivity and specificity, and an AUC of 0.5 reflects a performance comparable to that of a completely random classifier. Finally, it should be noted that, in most cases, perfect classification is unattainable. Often, more useful is the search for models which are straightforwardly interpretable, frugal, and serve as a reasonable estimation to reality.

Chapter 4

Modelling and Data Analysis

The hardware at our disposal featured a Dual-Core Intel Core i3-380M processor with 4GB of RAM. The modelling was conducted entirely in R 3.6.3, and made extensive use of the following packages: `glmnet` [16], `caret` [26], `e1071` [30], and `penalizedSVM` [4].

4.1 Application to Real Datasets

The datasets employed in this chapter are all suitable for the task of binary classification, and specifically concern the task of predicting the presence of certain cancers. We begin with a low-dimensional case and increase the number of predictor variables with each instance of a new dataset, with the latter two being such that $p \gg N$. This is done to compare the performance of the classifiers over varying situations, and to further contrast these with theoretical claims, with the ultimate goal being to find models that perform in a superior manner on high-dimensional microarray data. Indeed, we suspect there will be no free lunch; that is, no model will prove strictly dominant in all circumstances. The candidate models selected for comparison are as follows:

(1) LR with $p = \min\{p, 75\}$: problem (2.12).

(2) Regularised LR with the lasso penalty: problem (2.18).

(3) Regularised LR with the elastic-net penalty: problem (2.21).

(4) SVM with a linear kernel: problem (2.49).

(5) SVM with a radial basis kernel: problem (2.49).

(6) Regularised SVM with the SCAD penalty: problem (2.53).

In all cases, to calculate the test error rate given by (3.6), we choose l to be equal to $N/2$; that is, τ_1 and τ_2 are of equal size. As mentioned previously, all model parameters are tuned via k-fold cross-validation with $k = 5$. More specifically, for each dataset,

all models were trained with multiple seed values to assess consistency. This means the parameters were tuned on different occasions with varying arbitrary cross-validation data splits. The results remained concordant. The results presented here are those for which the seed value was set to 42.

4.1.1 Breast Cancer Diagnostics Dataset

The first dataset is from a study by Street *et al.* [34], and was retrieved for our analysis from the UCI Machine Learning Repository [12]. The primary aim of the study was to improve objectivity in the diagnosis of breast tumours according to their malignancy. Finite needle aspirations (FNAs) extract fluid from a breast mass, bypassing the traditional requirement of conducting a full, invasive biopsy. A combination of geometric imaging techniques were utilised to ascertain the nucleus boundary of each cell, and to establish numerical quantities for characteristics such as the texture, concavity, and fractal dimension. The dataset is such that $N = 682$ and $p = 9$, with 238 observations corresponding to malignant tumours, and with 444 being benign. We assign a value of 1 to observations whose true class is Malignant (M), and 0 to those whose true class is Benign (B). A pairs plot of the covariates is plotted in Figure 4.1. If, for a particular variable, the red labels are well separated from the black labels, then this would indicate a relevant variable to distinguish between groups M and B. Upon initial inspection, most variables seem relevant.

Figures 4.2a and 4.3a shed some light on the parameter tuning process for the lasso and elastic-net LR models, respectively. With respect to lasso LR, Figure 4.2a graphs the cross-validation error incurred by a range of values for the regularisation parameter λ. The left-most dashed line represents the value of λ that corresponds to the minimum cross-validation error. In the literature, although less common, λ is sometimes chosen such that it corresponds to the minimum cross-validation error plus one standard deviation (given by the right-most dashed line) in order to avoid overfitting to the training data. Having said this, we select the optimal λ as per the former definition, and indeed do so for all tuning parameters. Regarding the elastic-net LR model, Figure 4.3a plots the cross-validation accuracy (1 minus cross-validation error) with respect to α, for varying given values of λ. It can be seen that the cross-validation accuracy is maximised, and hence the cross-validation error is minimised, at the left-hand side of the graph for each λ. This corresponds to an elastic-net LR "mixing" parameter of $\alpha = 0.1$.

Figures 4.2b and 4.3b give the coefficient paths of the lasso and elastic-net LR models, respectively, with the dashed lines highlighting the optimal value of λ according to 5-fold cross-validation in each case. Consistent with the reasoning of [16], the coefficient profile present in the former tend to be more erratic and show a less condensed, more dispersed distribution. The smoother aesthetic of the elastic-net LR coefficient profile is courtesy of the additional l_2 penalty giving rise to superior coefficient path stability. For each of the lasso and the elastic-net LR models, the optimal λ can be easily read off the relevant

graphs. We present these explicitly in Table 4.1, alongside the tuned parameters for the remaining models, which are not here visualised but are obtained in analogous fashion.

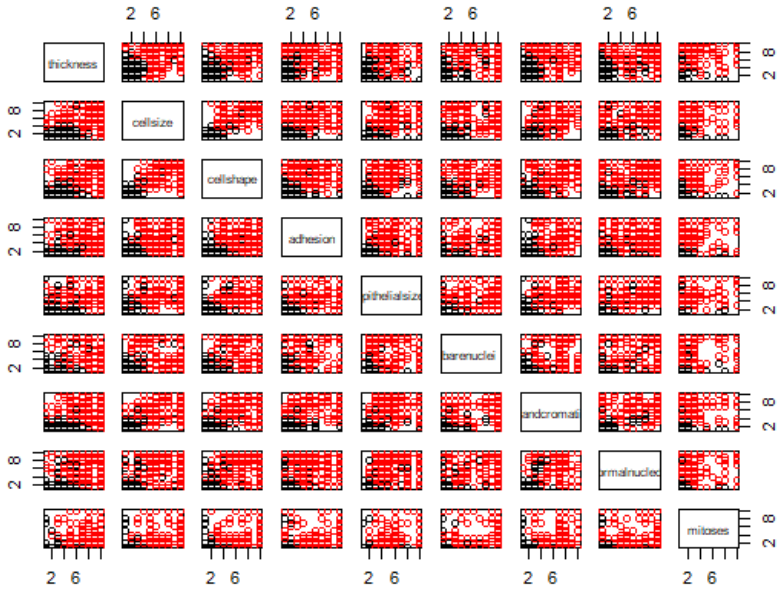
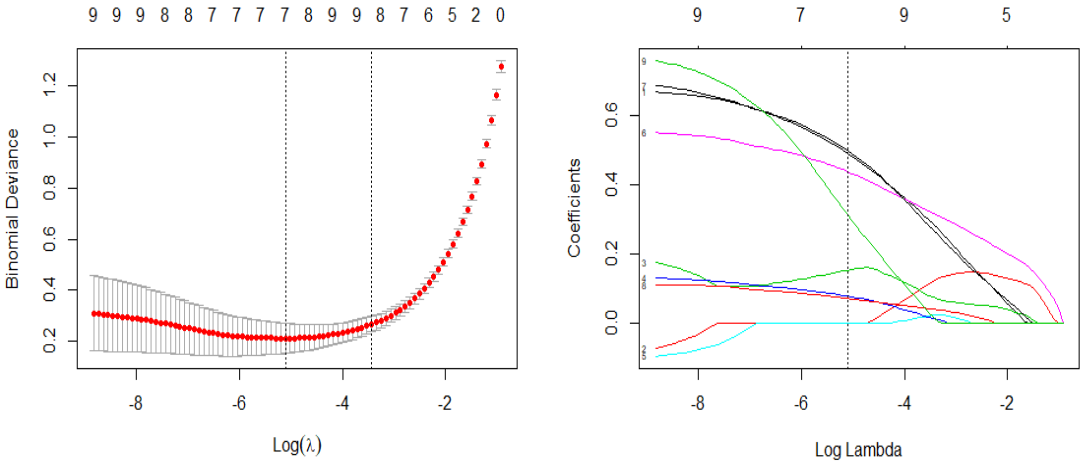


Figure 4.1: Pairs plot for all features in the breast cancer (Street) dataset.

Model	λ	α	K	γ
Lasso LR	0.00481275	1	-	-
Elastic-net LR	0.06187351	0.1	-	-
Linear SVM	-	-	1	-
Radial SVM	-	-	1	0.1111111
SCAD SVM	0.00080958	-	-	-

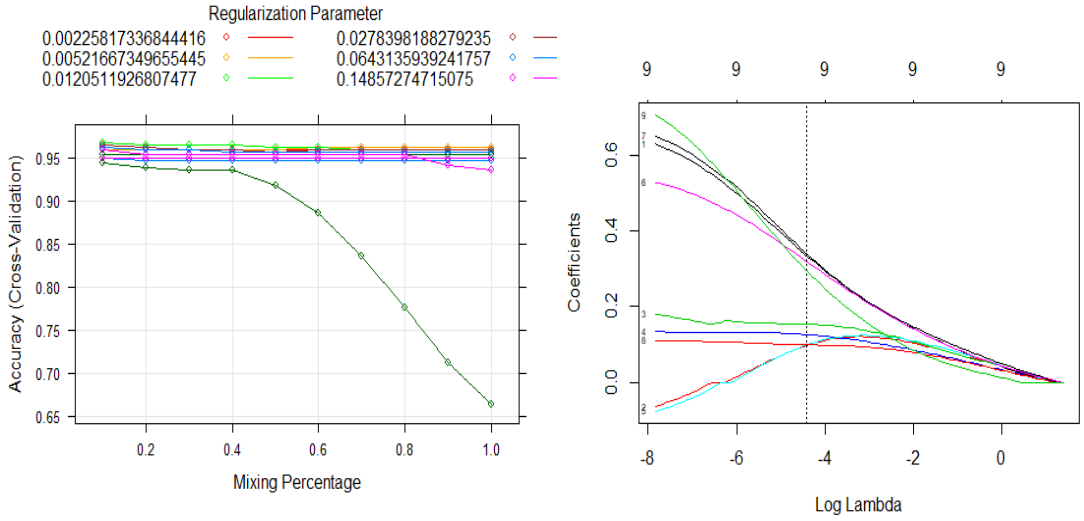
Table 4.1: Tuned parameter values for the breast cancer (Street) dataset.



(a) Cross-validation error for a range of λ.

(b) Coefficent path profile.

Figure 4.2: Parameter tuning of the lasso LR model for the breast cancer (Street) dataset training data.



(a) Cross-validation accuracy for a range of α, for given values of λ.

(b) Coefficient path profile.

Figure 4.3: Parameter tuning of the elastic-net LR model for the breast cancer (Street) dataset training data.

4.1.2 Colon Cancer Microarray Dataset

The next two datasets are each composed of DNA *microarray* data. Microarray technology simultaneously monitors thousands of gene expression levels, and significantly surpasses the volume of data obtained from more traditional genome sequencing techniques [13]. Molecules of DNA are fixed to glass slides in "spots", typically synthesised via a procedure known as photolithography. These molecules may take the form of either explicit genomic DNA, or of oligonucleotide sequences that uniquely match a single gene.

This second dataset comes from a study by Alon *et al.* [3]. The original aim of the paper was to analyse gene expression patterns across tumorous and normal colon tissue. Specimens of colon tissue were collected and placed in liquid nitrogen for snap-freezing. The samples were then treated with affymetrix oligonucleotide arrays, and the associated image processing carried out. In line with the original study, the 2000 genes with the highest minimum estimated spot intensity are used; that is, across all imaged samples, those with the greatest minimum volume of pixels within a precisely-defined spot boundary. The dataset, then, is such that $N = 62$ and $p = 2000$, with 40 observations corresponding to tumorous tissue, and with 22 being normal, healthy tissue. We assign a value of 1 to observations whose true class is Tumorous (T), and 0 to those whose true class is Normal (N). A pairs plot of the first 10 covariates is plotted in Figure 4.4. At first glance, it is not clear which variables may or not be relevant, and this certainly doesn't give a full representation of the relationship between all the present features.

Figure 4.5a graphs the cross-validation error for the lasso LR regularisation parameter λ, yielding an optimal value such that $\log(\lambda) = -1.948$. Figure 4.6a plots the cross-validation accuracy. Similarly to the Street dataset, this is maximised toward the left-hand side, for varying given values of λ, yielding an optimal mixing parameter of $\alpha = 0.1$. It can also be seen that Figures 4.5b and 4.6b again exhibit the expected traits; the lasso LR coefficient path is characteristically "wild" by comparison. Further, it is clear that, as per the theory, whilst the elastic-net LR model selects for more features than the lasso LR, the coefficients of each feature are outputted at a reduced absolute value. This can be seen by noting that the coefficient scale for the elastic-net LR is more granular by an order of magnitude.

Table 4.2 gives the values of the tuned parameters for each model with respect to the Alon dataset. Interestingly, it seems that the parameters of the radial basis SVM have been tuned in a way that work in extreme opposition to each other, in terms of the bias-variance trade-off. The cost K seems abnormally high, which leads to a very "hard" margin and hence a model of *high* variance and *low* bias. That is, insofar as fewer observations will be considered support vectors, each local segment of the decision boundary becomes more "tailored", or less "averaged out". Conversely, the very small value for γ corresponds to a wide Gaussian, implying a decision boundary influenced more heavily by observations a greater distance away. This leads to each local segment of the decision boundary being determined by the weighted average of many observations, rather

than just those close by, and hence a model of *low* variance and *high* bias. It will be interesting to see how these parameters equalise, though the extreme tuning values may be indicative of the idea that complex kernels can prove counterproductive in cases of high-dimensional data.

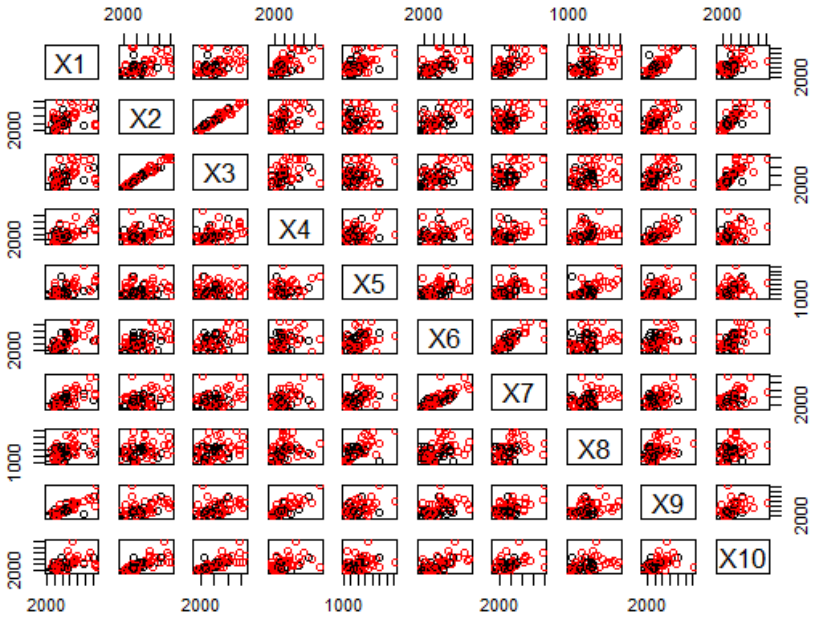
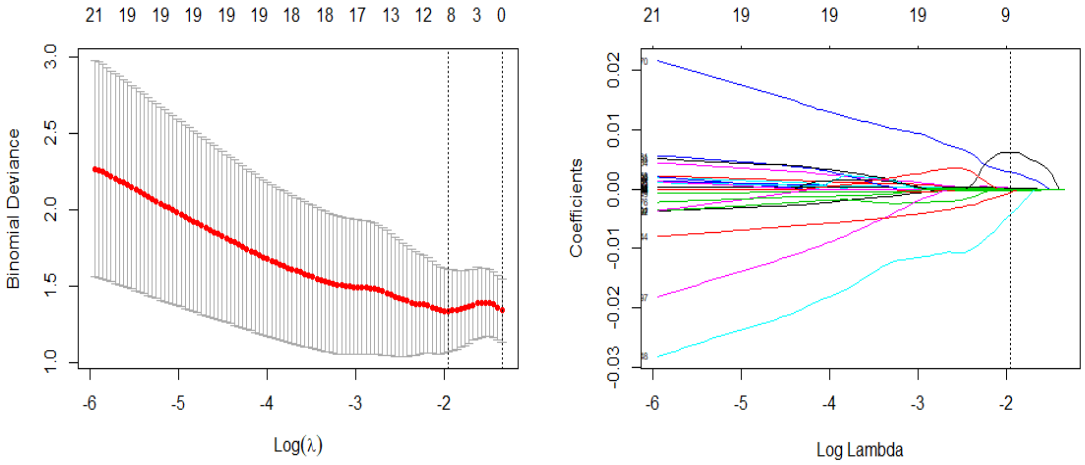


Figure 4.4: Pairs plot for the first 10 features in the colon cancer (Alon) dataset.

Model	λ	α	K	γ
Lasso LR	0.1424995	1	-	-
Elastic-net LR	0.3432823	0.1	-	-
Linear SVM	-	-	1	-
Radial SVM	-	-	4	0.000661314
SCAD SVM	0.233	-	-	-

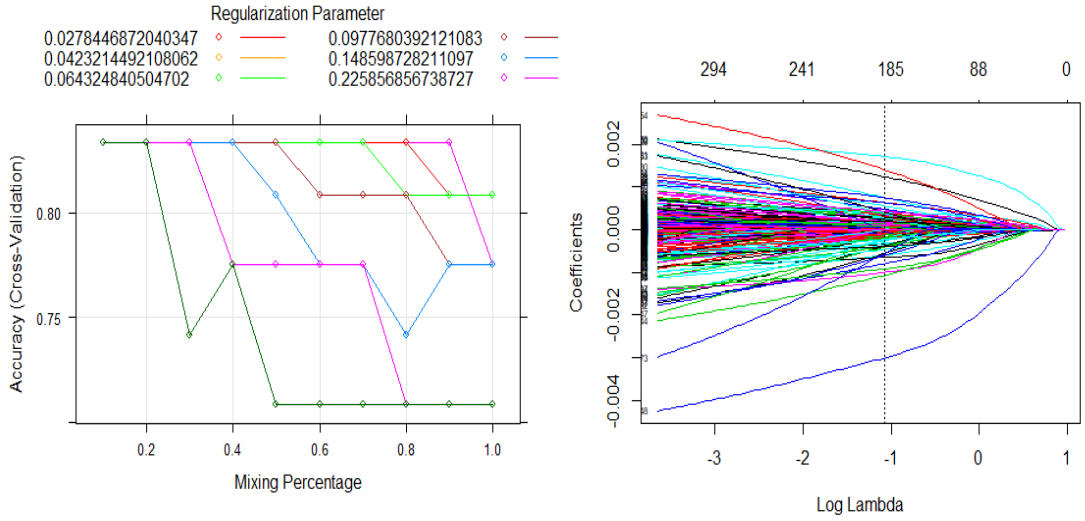
Table 4.2: Tuned parameter values for the colon cancer (Alon) dataset.



(a) Cross-validation error for a range of λ.

(b) Coefficient path profile.

Figure 4.5: Parameter tuning of the lasso LR model for the colon cancer (Alon) dataset training data.



(a) Cross-validation accuracy for a range of α, for given values of λ.

(b) Coefficient path profile.

Figure 4.6: Parameter tuning of the elastic-net LR model for the colon cancer (Alon) dataset training data.

4.1.3 Lung Cancer Microarray Dataset

The third and final dataset is that used in a study by Gordon *et al.* [17]. Primarily, the aim of the study was to discriminate instances of two different lung cancers: malignant pleural mesothelioma (MPM) and adenocarcinoma (ADCA). Since the symptomatic physiology of the two cancers resemble each other remarkably closely, and yet the respective treatments differ vastly, differential diagnoses are critical. Whilst more traditional methods prove clumsy and complicated in discerning the two cancers, statistical analysis of microarray samples provide an opportunity for a relatively inexpensive and accurate mode of diagnosis. Surgical specimens were taken from patients and snap-frozen. The MPM samples consisted of purely tumorous tissue. The lung ADCA samples were made up of a combination of primary malignancies (those that had originated directly in the lung) and those of colon and breast origin that had metastasised to the lung site. The samples were then hybridised to oligonucleotide arrays, and processed for imaging.

As per the primary study, some of the samples that possess artefacts from the scanning process are suitably omitted. The dataset is consequently such that $N = 180$ and $p = 12,533$, with 31 observations corresponding to the presence of MPM, and with 149 being ADCA samples. The unbalanced nature of this dataset should be kept in mind throughout the analysis, and has a bearing on the results. We assign a value of 1 to observations whose true class is MPM, and 0 to those whose true class is ADCA. A pairs plot of the first 10 covariates is plotted in Figure 4.7. Initial thoughts are that, at least with respect to the Alon dataset, there is less overlap between red and black labels. We may hence expect relatively fewer variables to prove relevant to discrimination between the two classes in this instance. Again, though, this may not representative of the relationship between all pairwise combinations of the present features.

Figure 4.8a graphs the cross-validation error for the lasso LR regularisation parameter λ, yielding an optimal value such that $\log(\lambda) = 5.728$. Notably, Figure 4.9a highlights behaviour very similar to the previous two datasets, with small α proving optimal. Strangely, with respect to the associated coefficient paths, those yielded by the lasso LR model in Figure 4.8b exhibit the peculiar trait that all coefficients are non-negative. Since the features themselves take on non-negative values, this would seem to suggest *all* the selected features contribute positively to classifying an observation as MPM. This is visibly not the case for the elastic-net LR in Figure 4.9b.

The values of the tuned parameters for each model trained on the Gordon dataset are given in Table 4.2. Of obvious note here is the value of the regularisation parameter λ for the SCAD SVM model, which sits at a remarkably large value of 10. In this case, we hence have the expectation of heavy penalisation and of strongly sparse solutions. Generally speaking, we will make the finding that the best performing models in this case are those that flexibly deal with issues that arise with the lasso penalty and yet remain sparse; the Gordon dataset cross-validation outputs suggest greater magnitudes

for the regularisation parameter λ are desirable for both elastic-net LR and the SCAD SVM, relative to the outputs for the previous datasets.

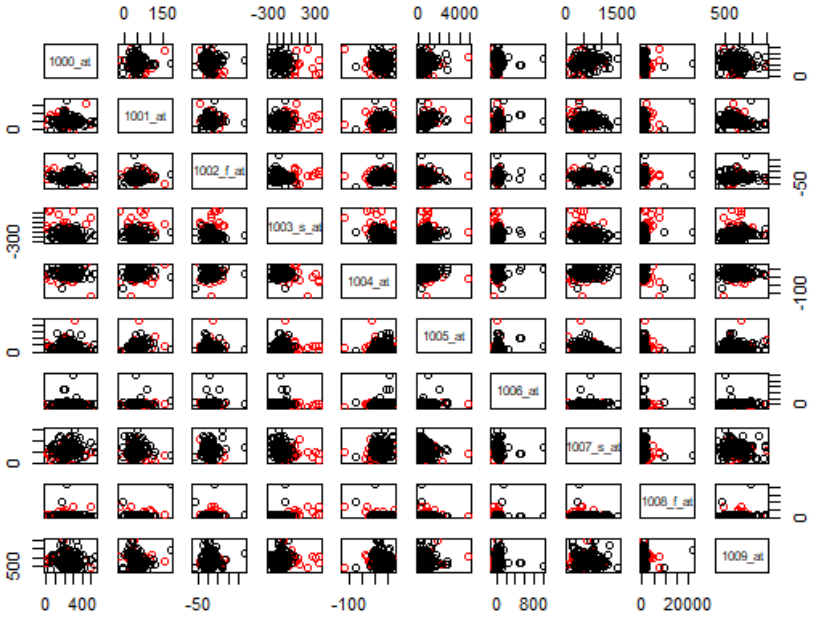
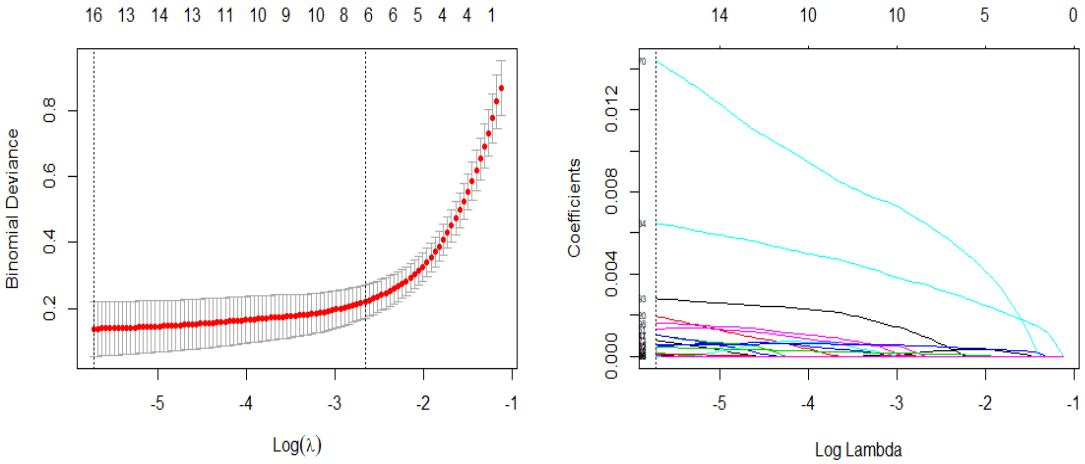


Figure 4.7: Pairs plot for the first 10 features in the lung cancer (Gordon) dataset.

Model	λ	α	K	γ
Lasso LR	0.0032538	1	-	-
Elastic-net LR	0.4281532	0.1	-	-
Linear SVM	-	-	1	-
Radial SVM	-	-	1	0.0000797894
SCAD SVM	10	-	-	-

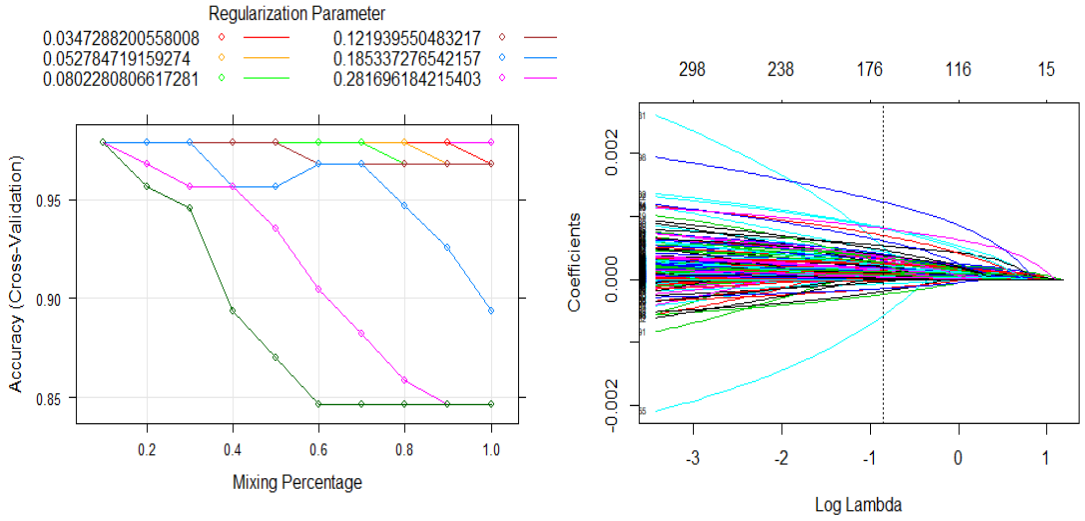
Table 4.3: Tuned parameter values for the lung cancer (Gordon) dataset.



(a) Cross-validation error for a range of λ.

(b) Coefficient path profile.

Figure 4.8: Parameter tuning of the lasso LR model for the lung cancer (Gordon) dataset training data.



(a) Cross-validation accuracy for a range of α, for given values of λ.

(b) Coefficient path profile.

Figure 4.9: Parameter tuning of the elastic-net LR model for the lung cancer (Gordon) dataset training data.

4.1.4 Model Performance

After computing the optimised tuned parameters for each of the candidate models on the training sets, their performances as predictive binary classifiers were gauged by application to the respective test sets. The ROC curves for the first five models are plotted in Figure 4.10, with curves that more closely approach the upper left corner corresponding to models of greater discriminatory power. Since the ROC curves may prove slightly difficult to interpret, the AUC values usefully make this more clear. These are presented in Tables 4.4, 4.5 and 4.6 for each of their associated datasets, alongside the other performance metrics discussed in Section 3.3. Sadly, we had issues obtaining corresponding outputs for the SCAD SVM in the `penalizedSVM` package.

Model	Accuracy	Sensitivity	Specificity	# Features	AUC
LR	0.9619	0.9431	0.9817	9	0.9949
Lasso LR	0.9589	0.9431	0.9862	8	0.9963
Elastic-net LR	0.9707	0.9512	0.9862	9	0.9977
Linear SVM	0.9677	0.9431	0.9771	9	0.9971
Radial SVM	0.9736	0.9756	0.9862	9	0.9944
SCAD SVM	0.9707	0.9756	0.9679	9	-

Table 4.4: Breast cancer (Street) dataset results.

Model	Accuracy	Sensitivity	Specificity	# Features	AUC
LR ($p = 75$)	0.6452	0.6667	0.6154	75	0.6047
Lasso LR	0.5806	1	0	8	0.8675
Elastic-net LR	0.7419	0.9444	0.4615	186	0.812
Linear SVM	0.6667	0.8571	0.4615	2000	0.6319
Radial SVM	0.5806	0.8889	0.1538	2000	0.7692
SCAD SVM	0.6774	0.9444	0.3077	49	-

Table 4.5: Colon cancer (Alon) dataset results.

Model	Accuracy	Sensitivity	Specificity	# Features	AUC
LR ($p = 75$)	0.7333	0.6471	0.7534	75	0.7002
Lasso LR	0.9444	0.7059	1	16	0.8529
Elastic-net LR	0.9778	0.8824	1	169	0.9412
Linear SVM	0.9889	0.9412	1	12,533	0.9706
Radial SVM	0.8889	0.41176	1	12,533	0.7059
SCAD SVM	1	1	1	5	1

Table 4.6: Lung cancer (Gordon) dataset results.

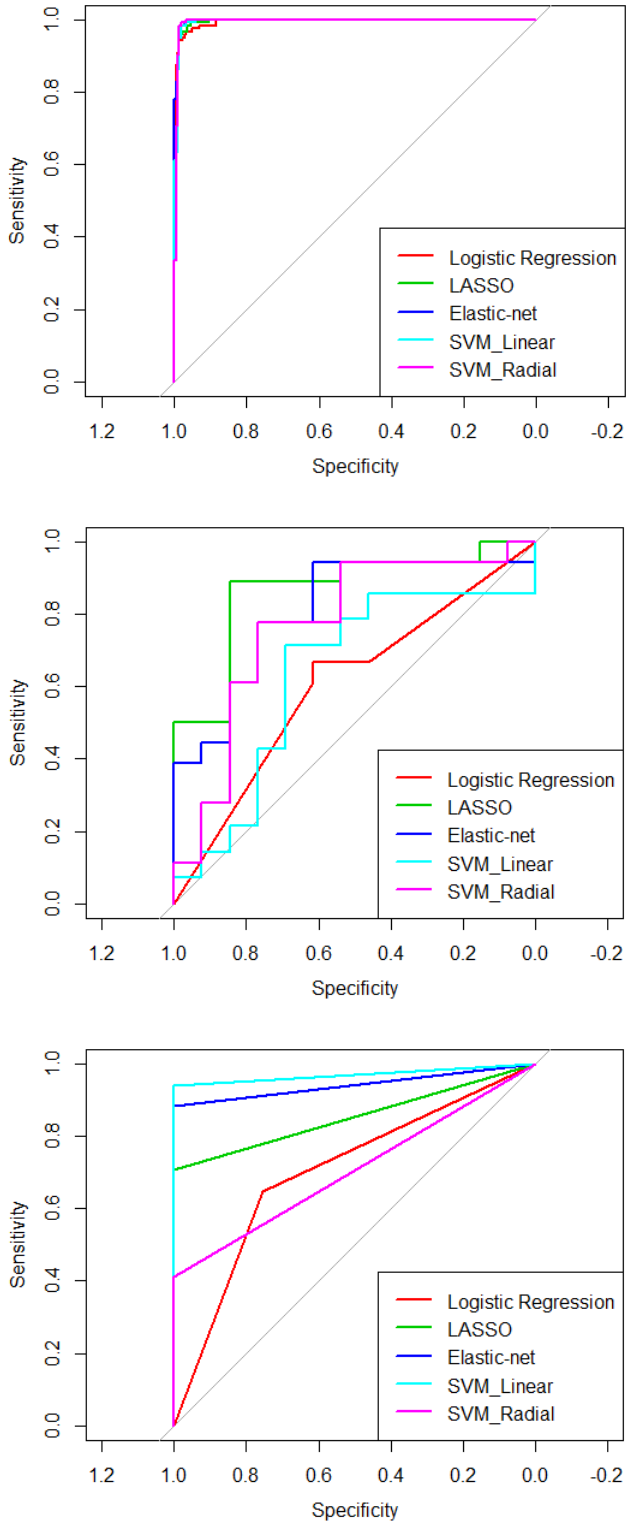


Figure 4.10: ROC plots for each model, minus the SCAD SVM. *Top:* Breast cancer (Street). *Middle:* Colon cancer (Alon). *Bottom:* Lung cancer (Gordon).

4.2 Discussion and Comparison to Theoretical Claims

Breast Cancer Diagnostics Dataset

The notable feature from the first dataset is that the lasso LR model is the only classifier that shrinks any features to zero at all. This vindicates our expectation that the lasso will generally exhibit the most shrinkage, being the extreme of the convex elastic-net problem in (2.21). The fact that all the models seem reluctant to remove any features concurs with our initial inspection of Figure 4.1, where the red labels were well separated from the black labels in all pairwise plots of the features.

The elastic-net LR model boasts the greatest AUC value, implying a strong ability to be able to parse true signals from false signals. The radial basis SVM exhibits the highest accuracy, though; this is testament to the kernel complexity being able to tease out the nuanced behaviours in the training set, and hence supports the claims made in [25]. Overall, however, for this low-dimensional dataset, all the models perform respectably and there is not much to tell them apart.

Colon Cancer Microarray Dataset

Conversely to the previous instance, the radial basis SVM here misclassifies more readily than its linear counterpart. This coincides with the ideas of [2], who posit that exotic kernels can not only prove non-beneficial, but actively counterproductive, if already operating in high-dimensional space where linear separability is more easily obtained. In this case, the more involved kernels have a tendency to overfit the training data, a problem which is especially amplified with our very few (31) training instances.

With this dataset, the lasso LR model seemed to struggle. In fact, it classified all test observations as belonging to class T, hence the extreme values for the sensitivity and specificity. Notably, though, the lasso LR's AUC value remains remarkably high. Indeed, altering the threshold from $r = 0.5$ to $r = 0.8$, for instance, yields a far more respectable accuracy of 0.8065. This is not a characteristic unique to the lasso LR model for this dataset, however, with the trend also holding true for all other candidate models. Unfortunately, in this context, the solution to this issue is not as simple as hiking the threshold value. We know that the threshold represents the relative costs associated with a false negative and a false positive observation. In the context of cancer detection where the malignant class is assigned the value 1, it is generally advisable to have a relatively low threshold. This is equivalent to saying that high sensitivity is preferable to high specificity; intuitively, it is considered less desirable to misclassify a patient as healthy when they in fact require immediate treatment, than vice-versa where a patient may go transiently untreated but be diagnosed at a later date.

Further, it seems that lasso LR too readily attempts to remove features. For test sets with strongly correlated pairwise features, it is well-known in the literature that the lasso penalty can resort to dropping one somewhat arbitrarily. Indeed, looking back at

Figure 4.4, some of the features exhibit some clear association. This presents an issue when the features are not correlated at the population level. Interestingly, the source of the lasso's relative shortcomings can be seen to be a failure to satisfy the *irrepresentable condition*, which is almost necessary and sufficient for the lasso's sparsity profile and the true sparsity profile are asymptotically identical [29]. Given our design matrix $\boldsymbol{X}$, the symmetric covariance matrix $\Sigma = N^{-1}\boldsymbol{X}^T\boldsymbol{X}$ can be rewritten as

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22}, \end{pmatrix} \tag{4.1}$$

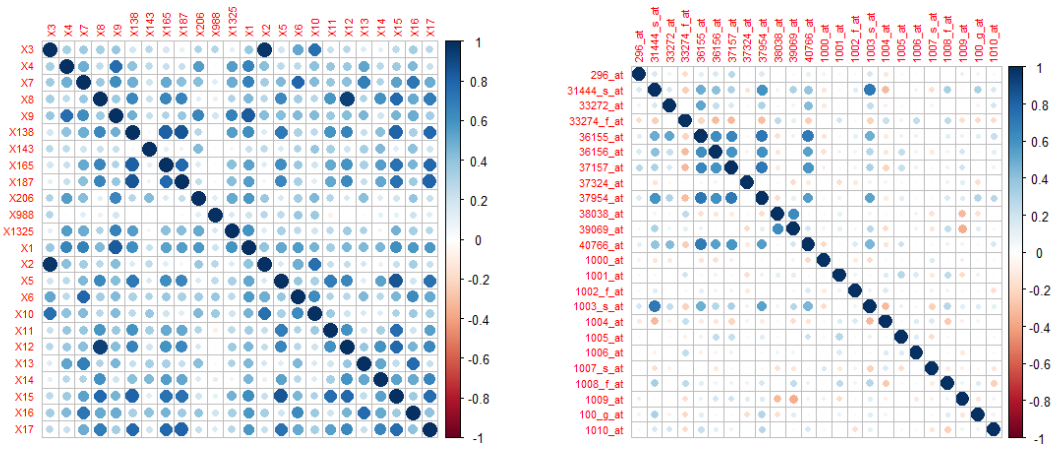
where Σ_{11} is the covariance matrix of the selected features, Σ_{22} is the covariance matrix of the features whose coefficients get shrunk to zero, and Σ_{12} and Σ_{21} give the covariance matrices between the selected and dropped features. The irrepresentable condition then takes the form

$$\left|\Sigma_{21}\Sigma_{11}^{-1}\ \mathrm{sgn}(\boldsymbol{\beta}_{(1)})\right| < 1, \tag{4.2}$$

where $\boldsymbol{\beta}_{(1)}$ is the vector of non-zero coefficients, and (4.2) must hold element-wise. Since the populational covariances are generally unavailable, correlation plots are utilised as a proxy diagnostic measure, as plotted in Figure 4.11. Figure 4.11a and Figure 4.11b depict the strength of the correlations between the first twelve selected features and the first twelve dropped features (in this order) for the Alon and Gordon datasets, respectively. The former is such that the selected features have non-trivial correlation with the dropped ones, hence violating the irrepresentable condition. By contrast, the latter is such that the selected features have little correlation with the dropped ones (neither the top right nor bottom left are densely coloured). We can correspondingly see in Table 4.6 that the lasso LR model acts as intended on the Gordon dataset.

More specifically, we can compare how the lasso LR model operates on the two datasets by contrasting their coefficient regularisation path profiles. Re-examination of Figure 4.5b shows how the coefficients of some of the irrelevant features (the black line is most obvious) begin to increase again towards the end of the path, before dipping back down to zero. Since these features are chosen along the whole coefficient path, it becomes very difficult to obtain the true set of features without using varying penalties for each feature. Again, this sits in contrast to the more well-behaved profile in Figure 4.8b, where the irrelevant features hit zero quickly as λ increases, whilst the relevant features converge more slowly. More of the rigorous specifics of the so-called irrepresentable condition can be found in [29] and [41].

Of the better performing models on the Alon dataset, the SCAD SVM manages to achieve a performance second only to elastic-net LR in terms of both accuracy and sensitivity, but does so more parsimoniously (only 49 features are selected for). This looks a promising trend as we move to more high-dimensional data. Indeed, this in line with the claims of [14] and [40] discussed in Section 2.3.5, who note that the SCAD penalty is protected from generating regularisation biases in favour of larger parameter



(a) Colon cancer (Alon) dataset.

(b) Lung cancer (Gordon) dataset.

Figure 4.11: Correlation plots for the first twelve selected (relevant) features and the first twelve dropped (irrelevant) features.

estimates, an issue the l_1 penalty is guilty of, and one which often arises more readily in high-dimensional data with many features.

Lung Cancer Microarray Dataset

For reasons previously stated, a low classification threshold r may be deemed desirable since sensitivity is a more useful measure for disease diagnosis than specificity. In the context of the Gordon dataset, though, this general rule may not be as applicable since the aim here is to differentiate between two distinct, but both active, cancers. In fact, the consistently high specificity values here suggest a tendency to assign an observation to class ADCA. Further, when interpreting the results on the Gordon dataset, it is crucial to remember that this dataset is highly unbalanced. As a result, all of the models have a propensity to overfit the majority class. Intuitively speaking, the classifiers would perform well according to their loss functions by crudely assigning to the majority class; that is, they would be guilty of the so-called "accuracy paradox".

Even with this in mind, it is still evident that the performance of the SCAD SVM here is impressive. It is perhaps the case that this is an "easy" dataset as far as classification is concerned, courtesy of its unbalanced nature. It can be true, for instance, that the training and test sets shared many underlying characteristics that are not present at the population level. But the relative performance of the SCAD SVM remains remarkable; it simultaneously proves to be the most parsimonious in feature selection, whilst being the only classifier to demonstrate perfect class separation on the test set.

On top of this finding, we also have further confirmation of the asymptotic tendency for the $l = N/2$ observations to form an l-dimensional simplex, based on the theoretical results in [19], for which linear discriminators are effective choice; the linear SVM exhibits

classification performance that matches or surpasses its radial basis kernel counterpart under all of the used performance metrics.

As forecast, the regularised feature-selecting classifiers yield fewer non-zero coefficients here compared to the same models on the Alon dataset, despite the pool of potential features being considerably greater. Specifically, this is congruous with assessment of the respective pairs plots, with the separation between red and black levels being clearly more distinct for the Gordon dataset. Generally, we have promising results across the board for the Gordon dataset, which is encouraging considering the historical difficulty of traditional diagnostic tools in distinguishing between MPM and ADCA.

General Discussion

Some of the limitations of the popular lasso penalty, as appended here to LR, are evident from our data analysis. Further, as intended, elastic-net LR is indeed less strict in feature selection than lasso LR, but the extra coefficient stability proves to be a superior characteristic in the case of *all* of the datasets here examined. As earlier alluded to, we have made the general finding that the best performing models have been those that flexibly deal with issues that arise with the lasso penalty and yet remain sparse. Specifically, in the most high-dimensional case, the cross-validation outputs suggested greater magnitudes for the regularisation parameter λ were desirable for both elastic-net LR and the SCAD SVM.

More generally, given their joint membership to the "Loss + Regularisation" family of problems given in (2.55), and the similarities of the hinge loss and binomial deviance, we expected penalised LR methods to perform broadly similar to SVM methods. Unfortunately, computational limitations prevented us from comparing ridge-regularised LR with the linear SVM, in which case all elements of the minimisation problem would be controlled apart from the respective loss functions. What we do see, though, is that the flexible regularised versions of each of the two main methods perform comparably. The elastic-net LR model and the SCAD SVM are consistently among the best classifiers, in terms of both accuracy and sparsity. We can also infer from the specificity and sensitivity values yielded by the SCAD SVM that it would have a comparably high AUC in each case. More specifically, elastic-net LR proves superior to the SCAD SVM in the first two cases (greater sensitivity in first instance), whereas the SCAD SVM pulls away in relative terms in the most high-dimensional case. They are only both bested in the case of low-dimensional data by the radial basis SVM, as expected. And the elastic-net LR model is bested by the linear SVM in very high dimensions; again, as expected, according to the theoretical claims. But these non-regularised SVM models require all features to be present, and hence lack the interpretability provided by the elastic-net LR model and the SCAD SVM.

The study also provides evidence that, in cases where $p \gg N$, expensive kernel functions can be done away with at no detriment to classification power, and that this becomes more true as p gets increasingly large.

Finally, it is a useful exercise to consider the drawbacks of the ROC curve and the AUC score, namely in our context of disease diagnostics. As discussed earlier, there are asymmetric misclassification impacts associated with false positives and false negatives when detecting disease, something which a raw AUC score does not take into account. In the medical literature, it is firmly held that diagnostic tools are best understood with reference to the advantages and disadvantages to a particular patient individual [20]. AUC is hence poor in terms of medical intepretability since this is not considered. Further, medical professionals may be unconcerned with a performance metric that is averaged over a range of cut-off points, r. Instead, it is typical to concentrate on cut-offs that are medically pertinent. Since the AUC evaluates the discriminatory ability of a classifier across all cut-off points, it is comprised of clinically illogical thresholds alongside those that are applicable. But as long as these caveats are recognised, and what the AUC actually evaluates - the capacity to rank observations as per class membership probabilities - is clear, the AUC remains a popular diagnostic tool. This holds especially true when the task is to compare the performance of multiple classifiers relative to each other, as has been the case here.

Chapter 5

Further Theoretical Extensions

For completeness, this chapter highlights some proposed augmentations in the literature to methods seen up to this point. They strive to combine some of the benefits of the different discussed classifiers featured in this thesis. Namely, they are the import vector machine (IVM) [43] and the elastic-SCAD SVM [5]. The discussion remains limited to the case of binary classification.

5.1 The Import Vector Machine (IVM)

This section is largely adapted from the work in [43]. Consider again a set of p predictor variables $\boldsymbol{X} = (X_1, \ldots, X_p)^T$, with a binary response variable $Y \in \{0, 1\}$, in line with LR tradition. Recall the general minimisation problem (2.58), which we restate here for convenience as

$$\min_{\phi \in \mathcal{H}_\mathcal{K}} \Big\{ \mathcal{J}(y, \phi(\boldsymbol{x})) + \lambda \|\phi\|^2_{\mathcal{H}_\mathcal{K}} \Big\}, \tag{5.1}$$

where $\mathcal{H}_\mathcal{K}$ is the RKHS induced by the kernel $\mathcal{K}(\cdot, \cdot)$, and (5.1) has solution

$$\hat{\phi}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^{N} \hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i). \tag{5.2}$$

We know that the KLR model is created by employing a binomial deviance loss function, and that it has interpretability advantages over its SVM counterpart as discussed in Section 2.4.2, since it outputs explicit probability estimates. What the *import vector machine* (IVM) strives to do is to keep this benefit whilst exploiting computational efficiencies in a similar vein to how SVMs use support vectors (points for which $\lambda_i = 0$) to fully characterise their decision boundary. As distinct from the original KLR model, the IVM generates a sub-model of (5.2) such that

$$\hat{\phi}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{\boldsymbol{x}_i \in \mathcal{V}} \hat{\lambda}_i y_i \, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i), \tag{5.3}$$

where $\mathcal{V} \subset \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ is the set of *import vectors*, and (5.3) should approximate (5.2) without compromising on classification performance. The algorithm for computing $\mathcal{V}$, in the case of binary classification, can be performed as follows:

(1) Let $\mathcal{V} = \emptyset,\ \mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\},\ j = 1.$

(2) For all $\boldsymbol{x}_l \in \mathcal{X}$, let

$$\phi_l(\boldsymbol{x}) = \sum_{\boldsymbol{x}_i \in \mathcal{V} \cup \{\boldsymbol{x}_l\}} \lambda_i \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i). \tag{5.4}$$

Find dual variables λ_i to minimise

$$\mathcal{I}(\boldsymbol{x}_l) = \sum_{i=1}^{N} \left\{ \log\left(1 + e^{\phi_l(\boldsymbol{x}_i)}\right) - y_i \phi_l(\boldsymbol{x}_i) \right\} + \lambda \|\phi_l(\boldsymbol{x})\|^2_{\mathcal{H}_\mathcal{K}}. \tag{5.5}$$

(3) Let

$$\boldsymbol{x}_{l^*} = \underset{\boldsymbol{x}_l \in \mathcal{X}}{\operatorname{argmin}}\, \mathcal{I}(\boldsymbol{x}_l) \tag{5.6}$$

and let $\mathcal{V} = \mathcal{V} \cup \{\boldsymbol{x}_l\},\ \mathcal{X} = \mathcal{X} \backslash \{\boldsymbol{x}_l\},\ \mathcal{I}_j = \mathcal{I}(\boldsymbol{x}_l),\ j = j + 1.$

(4) Repeat steps (2) and (3) until $\mathcal{I}_j$ converges.

(5) Terminate.

The notion of convergence in step (4) is defined by the stopping rule $\frac{|\mathcal{I}_j - \mathcal{I}_{j-\Delta j}|}{|\mathcal{I}_j|} < \epsilon$, where $\Delta j \in \mathbb{N}, \epsilon > 0$ are pre-selected by the modeller.

We know that the SVM support vectors correspond to observations nearest to the decision boundary, and that they fully define the SVM classifier. The IVM import vectors, by contrast, correspond to observations that minimise the binomial deviance most heavily, and may be near to or distant from the decision boundary. This distinction makes intuitive sense since although distant observations do not in any way govern the decision boundary position, they could well play a significant part in estimating class probabilities. Explicitly, the IVM is concerned with probability estimates of $p(\boldsymbol{x})$, whilst the SVM concentrates only on the quantity $\operatorname{sgn}[p(\boldsymbol{x}) - 1/2]$; that is, class membership.

This property of the the binomial deviance makes it generally preferable to the hinge loss in terms of interpretability. According to the data analysis conducted in [43], despite the "support vector" attribute being absent from the base KLR model, the IVM provides an appealing compromise, exhibiting classification power that rivals SVMs.

5.2 The Elastic-SCAD

This section is inspired by the findings in [5]. This follows from the discussion in Section 2.3.5 on sparsity-inducing norms, and the results in the previous chapter suggesting that the SCAD regularisation term, in low-dimensional settings that are non-sparse, can be subject to sparsity limitations. To tackle precisely this issue, [5] outlines the *elastic-SCAD* penalty function, which attempts to quench the drawbacks of each when employed solely. The combined penalty is appended to an SVM model, which we describe here as

$$\min_{\beta_0,\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \Big[1 - y_i\phi(\boldsymbol{x}_i)\Big]_+ + \lambda_r\|\boldsymbol{\beta}\|_2^2 + \sum_{j=1}^{p} P_{\lambda_s}\big(|\beta_j|\big) \right\}, \tag{5.7}$$

where λ_r and λ_s are regularisation parameters to be tuned. The penalty looks different to the elastic-net penalty form we are used to in (2.21), but this is simply an aesthetic choice due to the more complex nature of the SCAD term. Explicitly, (2.21) could be rewritten such that we denote $\lambda_r = \lambda(1-\alpha)$ and $\lambda_s = \lambda\alpha$.

According to [5], the combined regularisation term exhibits continuity, sparsity, and asymptotic normality, characteristics that give rise to the so-called *oracle property*. This is defined technically in [14]. Roughly, a model satisfying the oracle property is one whose performance equals that if the underlying, true model was known beforehand. Further, data analysis was conducted on both simulated data and real microarray datasets. In contrast to the elastic-net and SCAD SVMs, the elastic-SCAD SVM proved the one classifier that performed robustly in instances of both non-sparse and highly sparse data.

Finally, we hope at this point that it is clear that an elastic-SCAD LR analogue could be obtained by substituting the hinge loss in (5.7) for the binomial deviance. It is hence not an infeasible idea that the IVM algorithm in the preceeding section could be incorporated with a sparsity-inducing elastic-SCAD penalty. This could yield a kernel-based classification model that we expect would work well with both low-dimensional and high-dimensional datasets, whilst simultaneously maintaining both the probability estimate interpretability of LR and the computational efficiencies of SVMs. We think this offers motivation for further simulation studies.

Chapter 6

Conclusion

A conclusion is the place where you get tired of thinking.

— Arthur Bloch, *Murphy's Law*

In the present work, within the architecture of binary classification, we discussed logistic regression (LR) and the support vector machine (SVM) as belonging to a general family of regularised objective function minimisers. It was seen that kernel technology can be affixed in analogous fashion to either classifier. Additionally, any desired combination of a variety of penalty terms and kernel functions can be enjoyed, with the modeller's decision ultimately resting on their preference for either the binomial deviance, which explicitly estimates class probabilities, or the hinge loss function, for which computational shortcuts can be exploited.

Our results praise elastic-net and SCAD penalties in overcoming the limitations of the lasso, and are testament to their generalisation performance and simultaneous parsimony. In the most high-dimensional setting, the SCAD SVM was scathing in labelling nearly all features as irrelevant in distinguishing malignant pleural mesothelioma from adenocarcinoma at the lung site, a historically burdensome challenge. Additionally, for narrow data, kernel complexity proved able to tease out the nuanced behaviours in the training set, justifying their relevance for non-linearly separable data. The study also vindicates the claim that, for $p \gg N$, overly exotic kernel functions can be discarded at little to no cost to prediction power.

Finally, the elastic-SCAD term can be employed to bridge the gap between the two favourable penalties for use with both sparse and non-sparse data, and the import vector machine (IVM) imparts an attractive compromise between LR interpretability and SVM efficiency. Their capacities for extension beyond the binary case to multi-class problems in the wider literature further establishes these classifiers as parsimonious, understandable tools, both for medical diagnostics and in more varied contexts.

In response to Rutherford D. Rogers, then, though we may well be drowning in information, it is *learning* that leads to knowledge.

Bibliography

[1] Agresti, A. 2007. *An Introduction to Categorical Data Analysis.* Wiley Publishing.

[2] Ahn, J, Marron, J, Muller, K, & Chi, Y. 2007. The High-Dimension, Low-Sample-Size Geometric Representation Holds Under Mild Conditions. *Biometrika*, **94**(3), 760–766.

[3] Alon, U, Barkai, N, Notterman, D, Gish, K, Ybarra, S, Mack, D, & Levine, A. 1999. Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays. *Proceedings of the National Academy of Sciences*, **96**(12), 6745–6750.

[4] Becker, N, Werft, W, Toedt, G, Lichter, P, & Benner, A. 2009. penalizedSVM: A R-package for Feature Selection SVM Classification. *Bioinformatics*, **25**(13), 1711–1712.

[5] Becker, N, Toedt, G, Lichter, P, & Benner, A. 2011. Elastic SCAD as a Novel Penalization Method for SVM Classification Tasks in High-Dimensional Data. *BMC bioinformatics*, **12**(1), 138.

[6] Carmeli, C, De Vito, E, & Toigo, A. 2006. Vector Valued Reproducing Kernel Hilbert Spaces of Integrable Functions and Mercer Theorem. *Analysis and Applications*, **4**(04), 377–408.

[7] Collins, F, Morgan, M, & Patrinos, A. 2003. The Human Genome Project: Lessons From Large-Scale Biology. *Science*, **300**(5617), 286–290.

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

[9] Cover, T. 1965. Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. *IEEE Transactions on Electronic Computers*, 326–334.

[10] Domingos, P. 2000. A unified Bias-Variance Decomposition. *Pages 231–238 of: Proceedings of 17th International Conference on Machine Learning.*

[11] Draper, N.R, & Smith, H. 1998. *Applied Regression Analysis.* Wiley Publishing.

[12] Dua, Dheeru, & Graff, Casey. 2017. *UCI Machine Learning Repository.*

[13] Dubitzky, W, Granzow, M, Downes, C, & Berrar, D. 2003. Introduction to Microarray Data Analysis. *Pages 1–46 of: A Practical Approach to Microarray Data Analysis*. Springer.

[14] Fan, J, & Li, R. 2001. Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties. *Journal of the American statistical Association*, **96**(456), 1348–1360.

[15] Fawcett, T. 2006. An Introduction to ROC Analysis. *Pattern Recognition Letters*, **27**(8), 861–874.

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

[17] Gordon, G, Jensen, R, Hsiao, L, Gullans, S, Blumenstock, J, Ramaswamy, S, Richards, W, Sugarbaker, D, & Bueno, R. 2002. Translation of Microarray Data into Clinically Relevant Cancer Diagnostic Tests Using Gene Expression Ratios in Lung Cancer and Mesothelioma. *Cancer Research*, **62**(17), 4963–4967.

[18] Grosswindhager, S. 2009. *Using Penalized Logistic Regression Models for Predicting the Effects of Advertising Material*. Ph.D. thesis, Vienna University of Technology.

[19] Hall, P, Marron, J, & Neeman, A. 2005. Geometric Representation of High Dimension, Low Sample Size Data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **67**(3), 427–444.

[20] Halligan, S, Altman, D, & Mallett, S. 2015. Disadvantages of Using the Area Under the Receiver Operating Characteristic Curve to Assess Imaging Tests: A Discussion and Proposal for an Alternative Approach. *European Radiology*, **25**(4), 932–939.

[21] Hastie, T, Tibshirani, R, & Friedman, J. 2008. *The Elements of Statistical Learning*. Springer.

[22] Hastie, T, Tibshirani, R, & Wainwright, M. 2013. *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press.

[23] Hosmer, D.W, & Lemeshow, S. 2013. *Applied Logistic Regression*. Wiley Publishing.

[24] Jaggi, M. 2014. *An Equivalence Between the Lasso and Support Vector Machines*. Tech. rept. Chapman and Hall/CRC New York.

[25] James, G, Witten, D, Hastie, T, & Tibshirani, R. 2013. *An Introduction to Statistical Learning with Applications in R*. Springer.

[26] Kuhn, M, Wing, J, Weston, S, Williams, A, Keefer, C, Engelhardt, A, Cooper, T, Mayer, Z, Kenkel, B, Team, R Core, *et al.* 2020. Package 'caret'.

[27] McFadden, D, *et al.* 1973. Conditional Logit Analysis of Qualitative Choice Behavior. *Institute of Urban and Regional Development, University of California*.

[28] Meier, L, Van De Geer, S, & Bühlmann, P. 2008. The Group Lasso for Logistic Regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **70**(1), 53–71.

[29] Meinshausen, N, Yu, B, *et al.* 2009. Lasso-Type Recovery of Sparse Representations for High-Dimensional Data. *The Annals of Statistics*, **37**(1), 246–270.

[30] Meyer, D, Dimitriadou, E, Hornik, K, Weingessel, A, Leisch, F, Chang, C, Lin, C, & Meyer, M. 2019. Package 'e1071'.

[31] Münch, M, Peeters, C, Van Der Vaart, A, & Van De Wiel, M. 2018. Adaptive Group-Regularized Logistic Elastic Net Regression. *Biostatistics*, 1–15.

[32] Ohemeng, Matthew A. 2017. *A Comparative Study on Support Vector Machines.* Ph.D. thesis, University of Nevada, Reno.

[33] Pregibon, D. 1981. Logistic Regression Diagnostics. *The Annals of Statistics*, **9**(4), 705–724.

[34] Street, W N, Wolberg, W, & Mangasarian, O. 1993. Nuclear Feature Extraction for Breast Tumor Diagnosis. *Pages 861–870 of: Biomedical Image Processing and Biomedical Visualization*, vol. 1905. International Society for Optics and Photonics.

[35] Suykens, J. 2001. Nonlinear Modelling and Support Vector Machines. *Pages 287–294 of: IMTC 2001. Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference. Rediscovering Measurement in the Age of Informatics (Cat. No. 01CH 37188)*, vol. 1. IEEE.

[36] Tibshirani, R. 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society*, **58**(1), 267–288.

[37] Valentini, G, & Dietterich, T. 2004. Bias-Variance Analysis of Support Vector Machines for the Development of SVM-based Ensemble Methods. *Journal of Machine Learning Research*, **5**(Jul), 725–775.

[38] Wolfe, P. 1961. A Duality Theorem for Non-Linear Programming. *Quarterly of Applied Mathematics*, **19**(3), 239–244.

[39] Yuan, M, & Lin, Y. 2006. Model Selection and Estimation in Regression with Grouped Variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **68**(1), 49–67.

[40] Zhang, H, Ahn, J, Lin, X, & Park, C. 2006. Gene Selection Using Support Vector Machines with Non-Convex Penalty. *Bioinformatics*, **22**(1), 88–95.

[41] Zhao, P, & Yu, B. 2006. On Model Selection Consistency of Lasso. *Journal of Machine Learning Research*, **7**(Nov), 2541–2563s.

[42] Zhou, Q, Chen, W, Song, S, Gardner, Weinberger, K, & Chen, Y. 2015. A Reduction of the Elastic Net to Support Vector Machines with an Application to GPU Computing. *In: Twenty-Ninth AAAI Conference on Artificial Intelligence.*

[43] Zhu, J, & Hastie, T. 2002a. Kernel Logistic Regression and the Import Vector Machine. *Pages 1081–1088 of: Advances in Neural Information Processing Systems.*

[44] Zhu, J, & Hastie, T. 2002b. Support Vector Machines, Kernel Logistic Regression and Boosting. *Pages 16–26 of: International Workshop on Multiple Classifier Systems.* Springer.

[45] Zhu, J, Rosset, S, Tibshirani, R, & Hastie, T. 2004. 1-Norm Support Vector Machines. *Pages 49–56 of: Advances in Neural Information Processing Systems.*

[46] Zou, H, & Hastie, T. 2005. Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **67**(2), 301–320.

Appendix A

Algorithms

Appendix A provides an overview of certain algorithms and methods alluded to in the main text, but does not contain any claim to completeness. For further, more detailed rigour, please check the reference pages.

A.1 Newton-Raphson

Beginning with an argument of initial guess for the solution, the Newton method uses the first terms of the Taylor polynomial expansion, evaluated at said initial guess, to find another estimate closer to the solution. This process is iterated until convergence is reached. Assuming that the first n derivatives of f at x_0 all exist, recall that the Taylor polynomial of degree n for a function f at the point x_0 may be written as

$$\sum_{i=0}^{n} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i. \tag{A.1}$$

The first step in Newton's method is to take the first degree Taylor polynomial as an approximation for f, and set it equal to zero:

$$f(x_0) + f'(x_0)(x - x_0) = 0. \tag{A.2}$$

Solving for x, we get

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}. \tag{A.3}$$

Now let $x_1 = x$ and continue in an analogous manner to obtain $x_2, x_3, \ldots$ until successive approximations converge, and the algorithm terminates.

A.2 Elastic-SCAD SVM

In solving equation (5.7), the hinge loss is non-differentiable at zero, and the SCAD penalty is not convex in $\boldsymbol{\beta}$. The objective function can be locally approximated by a

quadratic function, and the minimisation problem can be solved iteratively. With the loss function averaged over the number of observations for mathematical convenience, denote the penalised objective function in (5.7) by

$$A(\beta_0, \boldsymbol{\beta}) := \frac{1}{N}\sum_{i=1}^{N}\Big[1 - y_i\phi(\boldsymbol{x}_i)\Big]_+ = +\lambda_r\|\boldsymbol{\beta}\|_2^2 + \sum_{j=1}^{p} P_{\lambda_s}\big(|\beta_j|\big). \tag{A.4}$$

For each i, the loss term can be split according to

$$\Big[1 - y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)\Big]_+ = \frac{1 - y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)}{2} + \frac{|y_i - (\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)|}{2}. \tag{A.5}$$

Given an initial value $(b_0, \boldsymbol{b}^T)$ close to the minimum of $A(\beta_0, \boldsymbol{\beta})$, consider the following local approximations:

$$|y_i - (\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)| \approx \frac{1}{2}\frac{\big(y_i - (\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)\big)^2}{|y_i - (b_0 + \boldsymbol{b}^T\boldsymbol{x}_i)|} + \frac{1}{2}|y_i - (b_0 + \boldsymbol{b}^T\boldsymbol{x}_i)|. \tag{A.6}$$

When b_j is close to zero, set $\hat{\beta}_j = 0$. Otherwise, use the approximation for the SCAD penalty

$$P_\lambda\big(|\beta_i|\big) \approx P_\lambda\big(|b_j|\big) + \frac{1}{2}\frac{P'_\lambda\big(|b_j|\big)}{|b_j|}(\beta_j^2 - b_j^2), \tag{A.7}$$

where, due to the symmetrical nature of the SCAD penalty, $|\beta_j|$ is used instead of β_j. It can be shown that both approximations and their original functions have the same gradient at the point $(b_0, \boldsymbol{b}^T)$. Therefore, the solution of the local quadratic function corresponds approximately to the solution of the original problem. The local quadratic approximation of $A(\beta_0, \boldsymbol{\beta})$ has the form

$$\begin{aligned} A(\beta_0, \boldsymbol{\beta}) \approx \frac{1}{2} &- \frac{1}{2N}\sum_{i=1}^{N} y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i) \\ &+ \frac{1}{4N}\sum_{i=1}^{N}|y_i - (b + \boldsymbol{b}^T\boldsymbol{x}_i)| \\ &+ \frac{1}{4N}\sum_{i=1}^{N}\frac{\big(y_i - (\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x}_i)\big)^2}{|y_i - (b_0 + \boldsymbol{b}^T\boldsymbol{x}_i)|} \\ &+ \sum_{j=1}^{p} P_{\lambda_s}\big(|b_j|\big) + \sum_{j=1}^{p}\lambda_r\beta_j^2 \\ &+ \sum_{j=1}^{p}\frac{P'_{\lambda_s}\big(|b_j|\big)}{2|b_j|}(\beta_j^2 - b_j^2). \end{aligned} \tag{A.8}$$

By minimisation of $A(\beta_0, \boldsymbol{\beta})$ with respect to β_0 and $\boldsymbol{\beta}$, terms without optimisation parameters β_0 and $\boldsymbol{\beta}$ can be dropped entirely, due to vanishing derivatives of constants. This leaves us with

$$\begin{aligned} A(\beta_0, \boldsymbol{\beta}) \approx & -\frac{1}{2N}\sum_{i=1}^{N} y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i) \\ & + \frac{1}{2N}\sum_{i=1}^{N} \frac{y_i \cdot (\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)}{|y_i - (b_0 + \boldsymbol{b}^T \boldsymbol{x}_i)|} \\ & + \frac{1}{4N}\sum_{i=1}^{N} \frac{(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}_i)^2}{|y_i - (b_0 + \boldsymbol{b}^T \boldsymbol{x}_i)|} \\ & + \sum_{j=1}^{p} \frac{P'_{\lambda_s}(|b_j|)}{2|\boldsymbol{b}_j|} \cdot \beta_j^2 + \sum_{j=1}^{p} \lambda_r \beta_j^2. \end{aligned} \tag{A.9}$$

To write the equations in matrix form, define

$$\begin{aligned} X &= \Big[\mathbf{1}, \boldsymbol{x}_1, \dots \boldsymbol{x}_p\Big], \\ \boldsymbol{r} &= \left[\frac{y_1}{|\epsilon_1|}, \dots, \frac{y_N}{|\epsilon_N|}\right]^T, \\ \mathcal{D}_0 &= \frac{1}{2N}\, diag\left[\frac{1}{|\epsilon_1|}, \dots, \frac{1}{|\epsilon_N|}\right], \\ \mathcal{Q}_1 &= diag\left[0, \frac{P'_\lambda(|b_1|)}{|\boldsymbol{b}_1|}, \dots, \frac{P'_\lambda(|b_d|)}{|\boldsymbol{b}_d|}\right], \\ \mathcal{Q}_2 &= diag\Big[0, 2\lambda_r, \dots, 2\lambda_r\Big], \\ \mathcal{P} &= \frac{1}{2N}(Y + \boldsymbol{r})^T X, \\ \mathcal{Q} &= X^T \mathcal{D}_0 X + \mathcal{Q}_1 + \mathcal{Q}_2. \end{aligned}$$

Then minimising $A(\beta_0, \boldsymbol{\beta})$ is equivalent to minimising the quadratic function given by

$$\tilde{A}(\beta_0, \boldsymbol{\beta}) = \frac{1}{2}\begin{pmatrix}\beta_0 \\ \boldsymbol{\beta}\end{pmatrix}^T \mathcal{Q}\begin{pmatrix}\beta_0 \\ \boldsymbol{\beta}\end{pmatrix} - \mathcal{P}\begin{pmatrix}\beta_0 \\ \boldsymbol{\beta}\end{pmatrix}. \tag{A.10}$$

The solution to (A.10) satisfies the linear system

$$\mathcal{Q}\begin{pmatrix}\hat{\beta}_0 \\ \hat{\boldsymbol{\beta}}\end{pmatrix} = \mathcal{P}. \tag{A.11}$$

The elastic-SCAD SVM can be implemented by the following iterative algorithmic steps:

(1) Set $k = 1$ and specify the initial value $\left(\beta_0^{(1)}, \boldsymbol{\beta}^{(1)^T}\right)$ according to the standard l_2 SVM, as per [40].

(2) Store the solution of the kth iteration by setting $\left(b_0, \boldsymbol{b}^T\right) = \left(\beta_0^{(k)}, \boldsymbol{\beta}^{(k)^T}\right)$.

(3) Minimise $\tilde{A}(\beta_0, \boldsymbol{\beta})$ by solving (A.11), and denote the solution as $\left(\beta_0^{(k+1)}, \boldsymbol{\beta}^{(k+1)^T}\right)$.

(4) Let $k = k + 1$. Repeat steps (2) and (3) until convergence is reached.

(5) Terminate.

If some $\beta_j^{(k)}$ is very close to zero - that is, when $|\beta_j| < \epsilon$ for some pre-selected small thresholding value $\epsilon > 0$ - then the jth variable is regarded as redundant. The jth column is removed from the matrix X, and the coefficient matrix in (A.11) is adjusted accordingly, with the iteration continuing as a reduced optimisation problem. Termination occurs upon inertia of $\left(\beta_0^{(k)}, \boldsymbol{\beta}^{(k)^T}\right)$.