STATISTICAL LEARNING AND HIGH-DIMENSIONAL DATA

DONGLIN ZENG
Department of Biostatistics
University of North Carolina at Chapel Hill

December 20, 2010
This set of notes are used for teaching class “Statistical Learning and High-dimensional Data”. The goal is to give a comprehensive review on statistical learning methods and the methods for handling high-dimensional data. Currently, these appear to be hot areas in statistical fields so I hope that through this course, students can have a sense of what are their foundations and what are systematic ways of handling statistical problems in these areas. In these notes, I try to cover a list of learning methods (supervised and unsupervised, parametric and nonparametric), some theoretical foundations for learning theory, and recent methods namely developed for high-dimensional data. For the first part, the most relevant reference is “The Elements of Statistical Learning: Data, Mining, Inference, and Prediction”, by Hastie et al, which can be downloaded from their website. The part on learning theory comes from “A Probabilistic Theory of Pattern Recognition”, by Devroye et al. Unfortunately, for the last subject, it is difficult to systematically review those methods so I provide a list of references which can be used for further reading. Even with the best effort, I am sure there are numerous typos, grammar errors, and incorrect statements and I take all the responsibilities for them.

Donglin Zeng
December, 2010
Contents

1 INTRODUCTION 7

I SUPERVISED AND UNSUPERVISED LEARNING 9

2 STATISTICAL DECISION THEORY IN SUPERVISED LEARNING 11

3 DIRECT LEARNING: PARAMETRIC APPROACHES 15
   3.1 Linear regression and shrinkage methods 15
      3.1.1 Subset selection 16
      3.1.2 Ridge regression 17
      3.1.3 Least Absolute Shrinkage and Selection Operator (LASSO) 18
      3.1.4 Other shrinkage methods 19
   3.2 Logistic regression and discriminant analysis 20
   3.3 Generalized discriminant analysis 22

4 DIRECT LEARNING: SEMI-NONPARAMETRIC APPROACHES 25
   4.1 Neural networks 25
   4.2 Generalized additive models 27
   4.3 Projection pursuit regression 28

5 DIRECT LEARNING: NONPARAMETRIC APPROACHES 31
   5.1 Nearest neighbor methods 31
CONTENTS

5.2 Kernel methods .............................................. 32
5.3 Sieve methods ............................................... 35
5.4 Tree-based methods ....................................... 37
5.5 Multivariate adaptive regression splines .................. 38

6 INDIRECT LEARNING ........................................... 41
6.1 Separate hyperplane ........................................ 41
6.2 Support vector machine ..................................... 43
   6.2.1 Support vector classifier ............................. 43
   6.2.2 Support vector machine ............................... 45
   6.2.3 Casting SVM into a penalized learning ............ 46
6.3 Function estimation via regularization ................... 46

7 AGGREGATED SUPERVISED LEARNING ....................... 49

8 MODEL SELECTION IN SUPERVISED LEARNING ............... 53
8.1 Akaike and Bayesian information criteria .................. 53
8.2 Model selection based on VC-complexity .................. 54
8.3 Cross-validation ........................................... 57

9 UNSUPERVISED LEARNING .................................. 59
9.1 Principal component analysis .............................. 59
9.2 Latent component analysis ................................ 60
9.3 Multidimensional scaling .................................. 61
9.4 Cluster analysis .......................................... 62

II LEARNING THEORY .......................................... 67

10 BAYES ERROR ................................................. 69
10.1 Bayes error .................................................. 69
10.2 Consistency based on Bayes error .......................... 73

11 CONSISTENCY OF DIRECT LEARNING METHODS 75
11.1 Histogram estimator ........................................ 75
11.2 $k$-nearest neighborhood estimator ....................... 78
   11.2.1 Asymptotic error for fixed $k$ ......................... 79
   11.2.2 Consistency of the nearest neighborhood rule with increasing $k$ 80
11.3 Kernel rules .................................................. 82
11.4 Classification rules based on sieve estimation .......... 83

12 CONSISTENCY OF INDIRECT LEARNING METHODS 87
12.1 The maximum likelihood principle ......................... 87
12.2 Empirical risk minimization ................................ 89
12.3 Empirical risk minimization with complexity regularization 90

13 CONVERGENCE RATES 93

14 CLASSIFICATION ERROR ESTIMATION 95
14.1 Resubstitution estimate .................................... 95
14.2 Cross-validation estimate .................................. 96
14.3 Other error estimates ...................................... 98

15 CONCENTRATION INEQUALITIES 101
15.1 Concentration inequalities for random variables ......... 101
15.2 Concentration inequalities for random processes .......... 103

III HIGH-DIMENSIONAL DATA 105

16 DIMENSION REDUCTION 107
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>16.1</td>
<td>Unsupervised dimension reduction</td>
<td>107</td>
</tr>
<tr>
<td>16.2</td>
<td>Supervised dimension reduction</td>
<td>109</td>
</tr>
<tr>
<td>17</td>
<td>VARIABLE SELECTION VIA REGULARIZATION</td>
<td>111</td>
</tr>
<tr>
<td>17.1</td>
<td>Fixed feature dimension</td>
<td>111</td>
</tr>
<tr>
<td>17.2</td>
<td>Increasing feature dimension</td>
<td>112</td>
</tr>
<tr>
<td>17.3</td>
<td>Computing algorithms</td>
<td>113</td>
</tr>
<tr>
<td>18</td>
<td>HIGH-DIMENSIONAL MULTIPLE TESTING</td>
<td>115</td>
</tr>
<tr>
<td>19</td>
<td>APPLICATIONS</td>
<td>117</td>
</tr>
<tr>
<td>19.1</td>
<td>Graph models</td>
<td>117</td>
</tr>
<tr>
<td>19.2</td>
<td>Spatial-temporal data</td>
<td>117</td>
</tr>
<tr>
<td>19.3</td>
<td>Individual tailored treatment</td>
<td>118</td>
</tr>
<tr>
<td>19.4</td>
<td>Population stratification</td>
<td>118</td>
</tr>
</tbody>
</table>
In these set of lecture notes, we concentrate on two major topics: one is called “statistical learning” and the other is named “statistical methods with high-dimensional data”. The former is focused on extracting important features from data based on certain loss functions; while the latter specifically concerns about estimation and inference when data variables are high-dimensional. Even though we divide the topics in this way, the two topics may not necessarily be mutually exclusive, as many methods in “statistical learning” also deals with high-dimensional data. In some context, these two topics are unified into the framework of the so-called “data mining”.

Different from machine learning which is popular in computer science, “statistical learning” refers to the situations when such learning methods are developed under some probabilistic framework. It usually consists of “supervised learning” and “unsupervised learning” (as you guess, there also exists some methods called “semi-supervised learning” but we will not study them in this book). By saying “supervised learning”, we aim to learn an outcome measurement (either quantitative or qualitative and sometimes called labels if it categorical) based on a set of features. To perform supervised learning, we should have a training set of data, which contains a set of feature variables and a column of outcome variable. Based on this training data set, we then develop a learning method/decision rule which enable us to use given feature variables to predict the outcome. A good learning method is the one that accurately predicts the outcome for any future observation. On the other hand, by saying “unsupervised learning”, we only observe the features but not outcomes. In this framework, the goal is to extract most important structures within observed feature data.

Compared to traditional statistical modelling, supervised learning is most similar to fitting
a regression model, where one is interested in finding the relationship between an outcome variable and a number of regressors; while unsupervised learning is most close to density estimation, where the focus is to find out how data present themselves in distributional sense. However, the key difference between traditional statistical modelling and statistical learning lies in their goals. The former aims to find the best model explaining the probabilistic behavior of data; thus, the maximum likelihood principle is usually adopted for estimation. Moreover, the former is specially concerned about the inference of model parameters so the efficiency of estimation method is often an important issue. Comparatively, statistical learning concentrates on prediction accuracy so developed learning methods are not necessary to maximize likelihood function but may minimize prediction errors as defined by certain loss functions. The inference itself is not of main interest in statistical learning (partially due to its own difficulty). Thus, because of the important role of loss functions in statistical learning, the theoretical foundation for statistical learning is based on statistical decision theory and the primary theoretical interest is often on estimation of prediction inaccuracy (sometimes called risk).

Compared to statistical learning methods, statistical methods dealing with high-dimensional data, although they experience explosive developments in the past ten years, appear to be less unified and are usually specific to relevant applications. Typical applications in high-dimensional data include the problems of variable selection in regression problems and statistical learning, multiple testing in high-dimensional tests and etc. Thus, it is rather difficult to provide a unified framework for this topic. Instead, we try to classify statistical methods for high-dimensional data into “two-stage methods” and “regularization methods”: the former refers to the methods where one performs dimension reduction first then proceeds to make inference in a much lower-dimensional space; while the latter refers to the methods where the dimensionality reduction and inference are carried out simultaneously via certain regularization methods.

The whole lecture notes are divided into three parts: statistical learning methods, learning theory and high-dimensional data. Throughout the book, we provide numerical examples to demonstrate the performance of each method. A lot of examples and contents are borrowed from HTF and DGL books. We acknowledge that there exist many topics related to the themes here which are not covered, for example, statistical learning methods for ordinal outcomes, semi-supervised learning, statistical learning in manifolds, learning censored outcomes, and some theories for large p small n etc. Finally, there are many places at which we should have cited references but we did not. All the errors are ours.
Part I

SUPERVISED AND UNSUPERVISED LEARNING
In this chapter, we formalize supervised learning based on statistical decision theory. Throughout, we use $X$ to denote the $p$-dimensional feature variables and use $Y$ to denote the outcome variable. We assume $(X, Y)$ from a joint distribution in some measure space. In supervised learning, one aims to find a map $f$ from the feature space to the space of the outcome such that the expectation of some loss function $L(Y, f(X))$ is minimized. That is, the target map $f = \text{arg\,min}_f E[L(Y, f(X))]$.

One important issue is the choice of the loss function, $L(y, x)$. Usually such a choice depends on data attributes and prediction purposes. For example, when $Y$ is continuous, a natural choice is the square loss with $L(y, x) = (y - x)^2$; when $Y$ is categorical, the most useful choice is called the zero-one loss by letting $L(y, x) = I(y \neq x)$. Of course, other choices of loss functions can be useful in some specific context, such as the $L_1$ loss function with $L(y, x) = |y - x|$ or the preference loss $L(y_1, y_2, x_1, x_2) = I(y_1 < y_2, x_1 < x_2)$ when $Y$ is ordinal. The plots of some loss functions are given in the following figure.

For some loss functions, the target map $f(X)$ can be explicitly obtained in terms of $(Y, X)$’s distribution. For example, in the square loss, $f(X) = E[Y|X]$ and in the $L_1$ loss, $f(X) = \text{med}(Y|X)$. For the zero-one loss with categorical $Y$, since

$$E[I(Y \neq f(X))] = \int \sum_{k=1}^{K} P(Y = y_k | X = x) I(f(x) \neq y_k) dP(x)$$

where $\{y_1, ..., y_K\}$ are the distinct nominal values of $Y$ and $P(x)$ is the marginal distribution,
Figure 2.1: Plot of loss functions: square loss, absolute loss, zero-one loss and Huber loss
we can obtain that the best $f(x)$ should be the one minimizing the integrand

$$
\sum_{k=1}^{K} P(Y = y_k|X = x)I(f(x) \neq y_k) = 1 - P(Y = f(x)|X = x)
$$

so $f(x) = \arg\max_k P(Y = y_k|X = x)$. The best $f$ is called the Bayes classifier and the minimal error is called the Bayes error. Particularly, if $Y$ is binary with value 0 or 1, then $f(x)$ chooses the category which has the conditional probability larger than $1/2$ and the Bayesian error is given by

$$
E[\min(\eta(X), 1 - \eta(X))] = \frac{1}{2} - \frac{1}{2}E[|2\eta(X) - 1|] = 1 - \sum_{k=0}^{1} I(f(x) = k)P(Y = k|X = x),
$$

where $\eta(X) = E[Y = 1|X]$. We remark that for many loss functions, $f(x)$ does not have an explicit solution.

Another important issue is how to estimate the best $f(x)$ using training data $(X_i, Y_i), i = 1, ..., n$. There are two commonly used methods for obtaining $f(x)$. The first approach is to directly estimate $f(x)$ if we know its explicit solution. We call this approach “direct learning”. For example, in the square loss, since $f(x) = E[Y|X = x]$, we can fit regression models to estimate this conditional mean; in the zero-one loss with dichotomous outcome, since $f(x) = E[Y = 1|X = x]$, a logistic regression model can be used to estimate $f(x)$. Most of the learning methods we will discuss in these lectures take this direct learning approach. The second approach, which we call “indirect learning”, is based on minimizing an empirical version of the expected loss given as

$$
L_n(f) = \sum_{i=1}^{n} L(Y_i, f(X_i)).
$$

Some literature call these methods as “empirical risk minimization” or “M-estimation”. Obviously, the indirect learning is universally applicable to any loss functions and it does not depend on whether or not the best $f(X)$ has an explicit solution.

In either direct learning or indirect learning, the choices of the candidates for $f(x)$ are often restricted to some functional spaces. There are two main reasons why this is needed. First, the dimension of the feature space $X$ is often high in practice. This high dimensionality makes the observed data a very sparse sample. For example, suppose we have $N$ data points uniformly distributed in a $p$-dimensional unit ball centered around the origin. It can be shown that the median distance from the origin to the closest data point is $(1 - 2^{-1/N})^{1/p}$. Thus, for $N = 5000$ and $p = 10$, such median distance is about $0.52$, more than half way to the boundary. This implies that most data points are closer to the boundary, which makes an
accurate estimation at the origin almost impossible. Such a phenomenon is well known as the
curse of dimensionality. To read more, see page 22-27 of HTF book. Since the data are sparse,
more extrapolation is needed for prediction but that requires that the candidates for \( f(x) \) cannot
be fully nonparametric so they must possess some restrictive structures. The second reason for
restricting the choices for \( f(x) \) is to avoid overfitting. For example, in indirect learning, if
\( L(y, x) \) is the square loss, one best solution is obtained by setting \( f(X_i) = Y_i \) and it gives a
perfect fit in the training data. However, such an \( f \) ignores the randomness in generating \( Y_i \)
and thus will inevitably cause large bias in future prediction. This is called overfitting which
should be avoided in practice.

There are two common ways to determine candidates for \( f(x) \) in learning literature. One
way is to restrict \( f \) to some candidate function space \( \mathcal{F}_n \), for instance, linear functions, the
spaces of splines or wavelets, additive functional spaces and etc. Such a function space \( \mathcal{F}_n \)
often increases with \( n \) and is called sieve space. Moreover, although the best \( f(x) \) may not
lie in \( \mathcal{F}_n \), we expect that the limit space of \( \mathcal{F}_n \) will eventually contain \( f(x) \). The other way
is that in estimating \( f(x) \) or minimizing the empirical risk, we impose some penalty term to
prevent those candidates from overfitting. The example of penalties include roughness penalty
in smoothing splines, the number of leaves in classification trees and etc. Penalties can also be
constructed for assessing learning methods using different function spaces for \( f \).
Chapter 3

DIRECT LEARNING: PARAMETRIC APPROACHES

In this chapter, we focus on parametric learning methods where \( f(x) \) is assumed to be a linear function of feature variables. The results can be generalized to more flexible cases when \( f(x) \) is assumed to be a linear combination of given basis functions, i.e., \( f(x) = \sum_{k=1}^{K} \beta_k h_k(x) \), where \( h_k(x) \) is the \( k \)th basis function such as mono-polynomials, B-splines, trigonometric functions and etc.

3.1 Linear regression and shrinkage methods

We assume that the outcome variable \( Y \) is a continuous quantity and the loss function is the square loss function. From the previous decision theory, we know that the target map is \( f(x) = E[Y|X = x] \). Further, we assume \( f(x) = x^T \beta \) (we include \( x \) the intercept term). Then \( f(x) \) can be easily estimated by the usual linear regression so obtain

\[
\hat{f}(x) = x^T(X^TX)^{-1}X^TY,
\]

where \( X \) is the matrix of all feature observations and \( Y \) is the column of all outcome observations. The theoretical properties of such an estimator are well known under Gaussian assumptions. See Section 3.2 and 3.3 in HTF book.

What we really want to discuss here is a variety of shrinkage methods in such a simple regression problem. There are two reasons why shrinkage is useful. The first one is that via shrinking some coefficients to zeros, we sacrifice a bit bias in prediction but gain in reducing the variability of the predicted values. The second reason is more for high-dimensional feature
space, in which one often believes only a small subset of the features really present strong effects. Thus, shrinkage methods can help to determine those important features. There exist many shrinkage methods in the linear regression problem, among which most of them are via penalty terms in terms of the model complexity. We only list the commonly used ones in the following sections.

### 3.1.1 Subset selection

This method aims to determine the best subset of given $k$ feature variables which gives the smallest residual sum of squares (RSS). In other words, one goes through all possible $k$ feature variables by fitting linear regression models, from which the best subset is selected if it yields the smallest RSS. An efficient algorithm—the leaps and bounds procedure (Furnival and Wilson, 1974)—is feasible for carrying out this process when the dimension of the whole feature space is below 40 but the procedure becomes infeasible if the dimension is much larger than 40. Once we determine the subsets for all $k$'s, the best $k$ will be further chosen based on some model assessment criteria. One particular criterion is based on the prediction error $E[(Y - \hat{f}_k(x_0))^2 | X = x_0]$, where $\hat{f}_k$ is the estimated function from the $k$ best feature variables. Under the assumption that $\text{Var}(Y - f(X)) = \sigma^2$, this prediction error is equivalent to

$$
\sigma^2 + (f(x_0) - E[\hat{f}_k(x_0)])^2 + \text{Var}(\hat{f}_k(x_0)),
$$

which thus consists of the irreducible noise error, the square of the bias, and the variance of $\hat{f}_k(x_0)$. Plugging $\hat{f}_k(x_0)$ into the above expression and taking the average over the feature points in the training data, we have that prediction error is

$$
\sigma^2 + \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - E[\hat{f}_k(X_i)])^2 + \frac{\sigma^2}{n} \text{Trace}(X_k^T (X_k^T X_k)^{-1} X_k),
$$

where $X_k$ is the feature matrix for the best subset of size $k$. On the other hand, we observe that the in-sample error, which is given by

$$
\frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{f}_k(X_i))^2 = \frac{1}{n} \sum_{i=1}^{n} Y_i^2 - \frac{1}{n} \sum_{i=1}^{n} \hat{f}_k(X_i)^2,
$$

has an expectation equal to

$$
\sigma^2 + \frac{1}{n} \sum_{i=1}^{n} \left\{ f(X_i)^2 - E[\hat{f}_k(X_i)]^2 \right\} - \frac{1}{n} \sum_{i=1}^{n} \text{Var}(\hat{f}_k(X_i)).
$$

Additionally, note that

$$
\frac{1}{n} \sum_{i=1}^{n} \left\{ f(X_i)^2 - E[\hat{f}_k(X_i)]^2 \right\} = \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - E[\hat{f}_k(X_i)])^2.
$$
We thus conclude that the expectation of the prediction error is equal to the expectation of the in-sample error plus \(2\sigma^2 n^{-1} \text{Trace}(X_k^T(X_k^T X_k)^{-1} X_k) = 2\sigma^2 k/n\). Therefore, the best \(k\) can be chosen as the one minimizing

\[
\frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{f}_k(X_i))^2 + 2\hat{\sigma}^2 k/n,
\]

where \(k = 1, \ldots, p\) and \(\hat{\sigma}^2\) is an estimator for \(\sigma^2\) using the whole feature space. This turns out to be the Mallow’s CP criterion function for model selection. There are other methods of finding the best, such as the AIC, BIC, and we will discuss them in later sections.

Alternatively, instead of searching through all possible combinations, we can search through a good path using either the forward, backward or stepwise selection strategy, where at each step, one either adds or deletes one feature variable and tests for its significance via F-statistic. One remark is that these strategies only control the best selection conditional on existing subsets so they may not find the best model at the end.

### 3.1.2 Ridge regression

Ridge regression is a method of obtaining the estimator for \(\beta\) while shrinking the regression coefficients by imposing a penalty on their sizes. Specifically, the estimator for \(\beta\) minimizes the following penalized summed residual squares:

\[
\sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2,
\]

where \(\lambda\) is a positive penalty parameter that controls the shrinkages, and the intercept term, \(\beta_0\), is left out from the second term. Clearly, the larger \(\lambda\) is, the more shrinkage the estimator will be. Numerically, such a minimization problem is equivalent to the following optimization problem:

\[
\min \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s,
\]

where there exists a one-to-one map between \(\lambda\) and \(s\) (in fact, we can set \(s = \sum_{j=1}^{p} \beta_{\lambda,j}^2\), where \((\beta_{\lambda,0}, \ldots, \beta_{\lambda,p})\) is the optimal solution to the first minimization problem). The ridge regression can also be understood as deriving the mean or mode of the posterior distribution for \(\beta\) when assuming that \(\beta\) has a prior distribution \(N(0, \tau^2)\) where \(\tau^2 = \sigma^2 / \lambda\). Thus, it is clear when \(\lambda\) is large, the prior distribution dominates so the posterior mean or mode shrinks to zeros.

The solution to the ridge regression gives

\[
\hat{\beta} = (X^T X + \lambda I)^{-1} X Y,
\]
where $I$ is the $p \times p$ identity matrix. Obviously, when we have no penalty ($\lambda = 0$), this is the usual least square estimator; when we increase the penalty constant, the coefficients in $\beta$ will shrink towards zeros. As in the usual least square regression, the trace of the project matrix $X^T(X^TX + \lambda I)^{-1}X$ is called the effective degrees of freedom.

### 3.1.3 Least Absolute Shrinkage and Selection Operator (LASSO)

LASSO is another shrinkage method similar to the ridge regression by replacing the square penalty by the absolute value penalty (sometimes we say replacing $L_2$-penalty by $L_1$-penalty). Particularly, we estimate $\beta$ by minimizing

$$
\sum_{i=1}^{n}(Y_i - X_i^T \beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j|.
$$

Or equivalently, we solve the following optimization problem:

$$
\min \sum_{i=1}^{n}(Y_i - X_i\beta)^2 \text{ subject to } \sum_{j=1}^{p} |\beta_j| \leq s,
$$

where $\lambda$ and $s$ are the penalty constants and there exists one-to-one map relationship between $\lambda$ and $s$ in these two equivalent problems. From the Bayesian point of view, the above estimation is equivalent to finding the posterior mode of $\beta$ after we impose the double exponential prior distribution for each component of $\beta$. Usually, the quadratic programming or the coordinate descent algorithm is used to obtain the solution.

To compare LASSO versus the previous shrinkage methods, let us examine one special case when the columns of $X$ are orthonormal variables. In this case, the least square estimator for the $j$th component of $\beta$ is given by $\hat{\beta}_{lse}^j = \sum_{i=1}^{n} X_{ij}Y_i$. In the best subset selection of size $k$, we only retain those $\hat{\beta}_{lse}^j$ when it is among the top $k$ absolute coefficients; in other words, we shrink those $(p - k)$ small coefficients to zeros. In the ridge regression, it is easy to see that $\hat{\beta}_{rs}^j = \hat{\beta}_{lse}^j / (1 + \lambda)$; therefore, we shrink all the coefficients proportionally. For LASSO, if $\hat{\beta}_{LASSO}^j \neq 0$, it solves equation that if $\beta_j \neq 0$,

$$
- \sum_{i=1}^{n} 2X_{ij}(Y_i - X_i^T \beta) + \lambda \text{sign}(\beta_j) = 0,
$$

so by the orthogonality of $X$, it solves

$$
2(\hat{\beta}_{lse}^j - \beta_j) = \lambda \text{sign}(\beta_j).
$$
Therefore, if the solution $\beta_j$ is positive, then $\beta_j = \hat{\beta}_j^{lse} - \lambda/2$; if the solution $\beta_j$ is negative, then $\beta_j = \hat{\beta}_j^{lse} + \lambda/2$. If $\hat{\beta}_j^{LASSO} = 0$, then the left-derivative of the objective function at $\beta_j = 0$ is negative but the right-derivative at $\beta_j = 0$ is positive. That is,

$$-2\hat{\beta}_j^{lse} - \lambda \leq 0, \quad -2\hat{\beta}_j^{lse} + \lambda \geq 0;$$

equivalently, $|\hat{\beta}_j^{lse}| \leq \lambda/2$. Combining these result, we obtain

$$\hat{\beta}_j^{LASSO} = \text{sign}(\hat{\beta}_j^{lse})(|\hat{\beta}_j^{lse}| - \lambda/2)^+. $$

This demonstrates the nonlinear shrinkage of the LASSO estimator: that is, for larger coefficients, their least square estimators are shrunk by the same constant $\lambda/2$ towards zero; for small coefficients, their least square estimator are shrunk to zeros. One remark we want to make here is that such selective behavior in the LASSO estimation is only true for the orthonormal feature variables; when the variables are correlated, this may not be true but the non-uniform shrinkage still exists.

From this simple example, we can see that both the best subset selection and LASSO estimation are useful for selecting important feature variables, but the ridge estimation is not. Since the best subset selection is computationally intensive or even infeasible when the feature space is large, the LASSO estimation becomes most attractive when one is interested in selecting important variables.

### 3.1.4 Other shrinkage methods

In addition to the LASSO estimation, there are many other shrinkage methods in literature. They can be categorized into two groups. The first group includes all the threshold methods, either hard threshold or soft threshold, where the former methods set those estimated coefficients to zeros once they are below some threshold and the latter methods only shrink these coefficient estimators towards zero. The threshold methods have been widely used in denoising signals via wavelets. The second group includes all the penalized methods such as the LASSO estimation. The difference among these methods lies in the choice of the penalty term in the minimization. One of such methods is to generalize LASSO to the following minimization problem:

$$\min \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q,$$

where $q$ is some negative number. Particularly, setting $q$ in (1,2) can gain partial advantage from both the LASSO estimation (selectivity) and the ridge estimation (good prediction perfor-
Another generalization of the LASSO estimation is to give flexible weights for penalizing different components of \( \beta \). That is, the estimator is obtained by solving the following problem

\[
\min \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda \sum_{j=1}^{p} w_j |\beta_j|,
\]

where \( w_j, j = 1, ..., p \) are some weights and could depend on data. One particular choice of the weights is to set \( w_j = |\hat{\beta}_{jSE}|^{-q} \) for some non-negative number \( q \). This becomes the so-called adaptive LASSO estimation (aLASSO). Some literature also consider the mixture \( L_2 \) and \( L_1 \) penalty in estimation:

\[
\min \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} |\beta_j|^2.
\]

There has also been some interest on obtaining the oracle property of selection: if the true \( \beta_j \) is known to be zero, the estimator for \( \beta_j \) is also zero with probability tending to one. Such an oracle property can be obtained if one uses some careful choices of the penalty term. One example is the Shrinkage Clipped Absolute Deviation (SCAD) penalty where the optimization problem becomes

\[
\min \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \sum_{j=1}^{p} J_\lambda(\beta_j),
\]

where

\[
J_\lambda(x) = \lambda \left\{ I(x \leq \lambda) + \frac{a \lambda - x}{(a - 1) \lambda} I(x > \lambda) \right\},
\]

where \( a \) is a constant larger than 2 (often \( a = 3.7 \) is used). The following figure shows how this penalty differs from the other ones discussed above. Since this optimization is not a convex problem, the computation is difficult.

Using LASSO and other shrinkage methods is not just restricted to linear regression; they have been applied to a variety of other regression problems. We will give more details in Part III of this book.

### 3.2 Logistic regression and discriminant analysis

In this section, we start to review parametric approaches for directly learning \( f(x) \) when \( Y \) is categorical. Such a problem is called a classification problem in order to make it different from
3.2. LOGISTIC REGRESSION AND DISCRIMINANT ANALYSIS

Figure 3.1: Plot of penalty functions with $\lambda=2$ for (a) the hard threshold; (b) aLASSO with $\alpha=3$; (c) SCAD

The regression problem in the previous section. From the decision theory, the ideal learning rule is to classify a future subject with feature $x$ into the category with label $k$, $k = 1, \ldots, K$, when $P(Y = k | X = x)$ is the largest. Thus, in direct learning, everything ends up with estimating $P(Y = k | X = x)$ using empirical observations.

A natural way of estimating $P(Y = k | X = x)$ is via a logistic model (if $Y$ is binary) or a log-odds model (if $Y$ has more than two categories). Particularly, we assume

$$P(Y = k | X) = \frac{\exp\{\beta_{k0} + X^T \beta_k\}}{1 + \sum_{l=1}^{K-1} \exp\{\beta_{l0} + X^T \beta_l\}}, \quad k = 1, \ldots, K - 1.$$

To estimate $\beta$, an iterative weighted least square algorithm is used to maximize the observe likelihood function. The resulting decision rule is then

$$f(x) = \arg\max_{k=1,\ldots,K} \left\{ \hat{\beta}_{k0} + x^T \hat{\beta}_k \right\},$$

where we set $\beta_{K0} = 0$ and $\beta_K = 0$.

Another commonly used method is called linear discriminant analysis. In this method, instead of modelling the conditional distribution of $Y$ given $X$, we model the distribution of the feature variables $X$ within each category of $Y$. Particularly, we assume that given $Y = k$, $k = 1, \ldots, K$, the distribution of $X$ is a multivariate normal distribution with mean $\mu_k$ and covariance matrix $\Sigma_k$; that is,

$$p_k(X) = \frac{1}{(2\pi)^{p/2} |\Sigma_k|^{1/2}} \exp\{-(X - \mu_k)^T \Sigma_k^{-1} (X - \mu_k)/2\}.$$
One can then maximize the observed likelihood function to estimate all the parameters,

\[ \hat{\mu}_k = \frac{\sum_{i=1}^{n} X_i I(Y_i = k)}{n_k}, \quad \hat{\Sigma}_k = \frac{\sum_{i=1}^{n} (X_i - \hat{\mu}_k)^T (X_i - \hat{\mu}_k)}{n_k}, \]

where \( n_k \) is the number of subjects in category \( k \). Under such an assumption, it is easy to see by the Bayesian rule,

\[ P(Y = k|X) = \frac{\pi_k p_k(X)}{\sum_{l=1}^{K} \pi_l p_l(X)}, \]

where \( \pi_k \) denotes the prior probability of \( Y = k \), and \( \sum_{l=1}^{K} \pi_l = 1 \). Therefore, the decision rule is that we classify one subject with feature value \( x \) into category \( k \) if \( p_k(x) \pi_k \) is the largest.

Under the special case when \( K = 2 \), this is equivalent to examine the sign of

\[ \log \frac{\pi_2}{\pi_1} - \frac{1}{2} (x - \hat{\mu}_2)^T \hat{\Sigma}_2^{-1} (x - \hat{\mu}_2) + \frac{1}{2} (x - \hat{\mu}_1)^T \hat{\Sigma}_1^{-1} (x - \hat{\mu}_1), \]

which is a quadratic function of \( x \). Such a rule is called quadratic discriminant analysis. If we further assume \( \Sigma_1 = \Sigma_2 = \Sigma \), then \( \hat{\mu}_k \) is the same as before but

\[ \hat{\Sigma} = \frac{\sum_{i=1}^{n} \sum_{k=1}^{2} (X_i - \hat{\mu}_k)^T (X_i - \hat{\mu}_k)}{n}. \]

The decision rule can be simplified as checking the sign of

\[ \log \frac{\pi_2}{\pi_1} - \frac{1}{2} \hat{\mu}_2^T \hat{\Sigma}^{-1} \hat{\mu}_2 + \frac{1}{2} \hat{\mu}_1^T \hat{\Sigma}^{-1} \hat{\mu}_1 + x^T \hat{\Sigma}^{-1} (\hat{\mu}_2 - \hat{\mu}_1). \]

This is called linear discriminant analysis as the rule is based on a linear function of \( x \). Some literature suggest to use \( \alpha \hat{\Sigma}_k + (1 - \alpha) \hat{\Sigma} \) to replace \( \hat{\Sigma}_k \) in the quadratic discriminant analysis, which is a compromise between the linear discriminant analysis and the quadratic discriminant analysis.

Comparing the logistic regression and discriminant analysis, it is not difficult to see that the former only models the distribution of \( Y \) given \( X \) so it can handle qualitative feature variables; the latter models the distribution of \( X \) given \( Y \) via normality assumption so it requires \( X \)'s being Gaussian. The former will be less efficient if the true distribution of \( X \) in each category is Gaussian; however, the latter is not robust to gross outliers. Generally, it is felt that the logistic regression is a safer and more robust procedure than the discriminant analysis, although a lot of numerical experiences do not really show that one performs better than the other.

### 3.3 Generalized discriminant analysis

There are some generalizations of the discriminant analysis methods we have discussed. One generalization is to replace feature variables by some basis functions of feature values. In this
Another generalization is to assume that the distribution of $X$ given each $Y$-category is a mixture normal distribution, i.e.,

$$ P(X|Y = k) = \sum_{r=1}^{R_k} \pi_{kr} N(\mu_{kr}, \Sigma), $$

where $\pi_{kr}$ is the mixing proportion. The estimators for the parameters can be obtained by maximizing the observed likelihood function, for which the expectation-maximization (EM) algorithm is often used.
Chapter 4

DIRECT LEARNING: SEMI-NONPARAMETRIC APPROACHES

In this chapter, we describe some semi-nonparametric approaches in direct learning. By saying semi-nonparametric, we mean that the model for estimating \( f(X) \) is assumed to be close to but not fully nonparametric. A list of such methods include neural networks, slice inverse regression, generalized additive models and multivariate adaptive regression splines.

4.1 Neural networks

Neural networks are prediction models for outcome \( Y \) (either quantitative or qualitative) based on input \( X \). These models are some directed networks with one or multiple hidden layers (see Figure 11.2 of HTF book). For description, we focus on the neural network with one single hidden layer (called vanilla neural) as shown in this figure. Suppose \( Z_1, ..., Z_m \) are the intermediate variable in the hidden layer. The first set of models are to link input \( X \) to \( Z_1, ..., Z_m \) via

\[
Z_k = \sigma_k(X^T \alpha_k), \quad k = 1, ..., m.
\]

The second set of models are to link \( Z_1, ..., Z_m \) to output \( Y \) by assuming

\[
E[Y|X] = g(\beta_1 Z_1 + ... + \beta_m Z_m + \beta_0) \equiv f(X).
\]

Here, the link functions \( \sigma_1(\cdot), ..., \sigma_m(\cdot) \) and \( g(\cdot) \) are usually from one of the following classes \( 1/(1 + e^{-x}), x, I(x > 0) \). Under the neural network models, the target function \( f(X) \) is then
estimated as
\[ g(\hat{\beta}_1 \sigma_1 (X^T \hat{\alpha}_1) + ... + \hat{\beta}_m \sigma_m (X^T \hat{\alpha}_m)), \]
where \( \hat{\beta} \)'s and \( \hat{\alpha} \)'s are the estimates for \( \beta \)'s and \( \alpha \)'s respectively. Since each single direct link is modelled parametrically, the neural networks appear to be parametric models. However, due to the arbitrary choices of the number of hidden variables \( Z \), such models are very flexible and one can even show that such networks will approximate any function of \( E[Y|X] \).

The advantage of the neural networks is also its computational simplicity due to simple parametric model in any direct link. An algorithm called back-propagation is used to estimate all the parameters (sometimes called weights). Specifically, we aim to minimize the following loss function
\[
\sum_{i=1}^{n} \left( Y_i - g(\beta_1 \sigma_1 (X_i^T \alpha_1) + ... + \beta_m \sigma_m (X_i^T \alpha_m)) \right)^2
\]
if \( Y \) is continuous, or
\[
- \sum_{i=1}^{n} Y_i \log g(\beta_1 \sigma_1 (X_i^T \alpha_1) + ... + \beta_m \sigma_m (X_i^T \alpha_m))
\]
if \( Y \) is binary. The back-propagation algorithm is a gradient decent algorithm, where at \((r+1)\)st iteration,
\[
\beta_k^{(r+1)} = \beta_k^{(r)} - \gamma_r \sum_{i=1}^{n} \delta_i Z_{ki},
\]
\[
\alpha_{kl}^{(r+1)} = \alpha_{kl}^{(r)} - \gamma_r \sum_{i=1}^{n} s_{ik} X_{il},
\]
where \( \gamma_r \) is the step size in the decent algorithm (called learning rate) and
\[
s_{ik} = \sigma_k'(X_i^T \alpha_k) \beta_k \delta_i;
\]
and
\[
\delta_i = -2(Y_i - f(X_i))f'(\beta_1 Z_{i1} + ... + \beta_m Z_{im} + \beta_0)
\]
for the continuous \( Y \) and
\[
\delta_i = -Y_i/f(X_i)f'(\beta_1 Z_{i1} + ... + \beta_m Z_{im} + \beta_0)
\]
for the binary \( Y \). Thus, the update for the parameters can be carried in two-pass algorithm. In the forward pass, we use the current parameters to estimate \( f(\cdot) \); in the backward pass, we compute \( \delta_i \) then \( s_{ik} \). Because the computation components are local, that is, each hidden unit passes and receives information only to and from units that share a connection, this algorithm can be implemented efficiently on a parallel architecture computer.
Finally, the learning rate $\gamma_r$ is usually taken to be a constant but can be optimized by a line search that minimizes the error at each update. See examples in Section 11.6 of HTF book.

4.2 Generalized additive models

Generalized additive models are one class of flexible models for directly estimating $f(x)$ (either $E[Y|X = x]$ or $P(Y = 1|X = x)$). For continuous $Y$, such models take form

$$E[Y|X_1, \ldots, X_p] = \alpha + \sum_{k=1}^{p} f_k(X_{(k)}),$$

where $f_1, \ldots, f_p$ are unknown smooth functions and $X_{(k)}$ denotes the $k$th component of $X$. For dichotomous $Y$, such models take form

$$\logit P(Y = 1|X) = \alpha + \sum_{k=1}^{p} f_k(X_{(k)}).$$

Clearly, the generalized additive model include the linear model as special cases and allow a fully nonparametric relationship between each component of $X$ and $Y$, but not a fully nonparametric relationship between the whole $X$ and $Y$. That is why we include this method as one of semi-nonparametric methods.

We first focus on continuous $Y$. The estimation of all $f$’s is based on minimizing a regularized loss function

$$\sum_{i=1}^{n} \{Y_i - \alpha - \sum_{j=1}^{p} f_k(X_{i(j)})\}^2 + \sum_{j=1}^{p} \lambda_j \int f''_j(t_j)^2 dt_j,$$

where $X_{i(j)}$ denotes the $j$th component of $X_i$. The second term is a penalty to penalize non-smoothness of $f_j$ and will result in fitting $f_j$ via cubic smoothing splines with knots at the observed $X_{i(j)}$’s. Other penalties can be used, as will be seen in next chapter. For identifiability, we assume $\sum_{i=1}^{n} f_i(X_{i(j)}) = 0$ so $\alpha$ is the average of $Y_i$’s. To minimize this objective function, there exists a simple algorithm called “backfitting” which can be used to estimate all $f_k$’s. This algorithm is described below:

1. Initialization: set $\hat{\alpha} = n^{-1} \sum_{i=1}^{n} Y_i$ and $\hat{f}_j = 0$, $j = 1, \ldots, p$.
2. Iterate from $j = 1, \ldots, p$. At $j$th iteration, we set

$$\hat{Y}_i = Y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(X_{i(k)}).$$

We fit smoothing splines by regressing $\hat{Y}_i$ on $X_{i(j)}$ to estimate $\hat{f}_j$. Cycle this iterations till the convergence of $\hat{f}$’s.
CHAPTER 4. DIRECT LEARNING: SEMI-NONPARAMETRIC APPROACHES

For qualitative outcome $Y$, the same backfitting algorithm can be applied: at $j$th iteration, we fix other $f$’s at the current value but maximize the likelihood function to estimate $f_j$. Such estimation can be particularly incorporated in the iteratively reweighted least squares algorithm. For example, in the case when $Y$ is a dichotomous outcome, the backfitting algorithm works as follows:

1. Set $\hat{\alpha} = \log[\bar{Y}/(1 - \bar{Y})]$ and $\hat{f}_j = 0$.
2. Define $\hat{n}_i = \hat{\alpha} + \sum_{j=1}^{m} \hat{f}_j(X_{i(j)})$ and $\hat{p}_i = 1/(1 + \exp{-\hat{n}_i})$. Let
   $Z_i = \hat{n}_i + (Y_i - \hat{p}_i)/(\hat{p}_i(1 - \hat{p}_i))$.
   and $w_i = \hat{p}_i(1 - \hat{p}_i)$. Repeat the second step in the previous backfitting algorithm by minimizing the following weighted least square
   $$\sum_{i=1}^{n} w_i(Z_i - \sum_{j=1}^{p} f_j(X_{i(j)}))^2.$$ Cycle till convergence.

Generalized additive models provide flexible modelling for obtaining the decision function $\hat{f}(X)$. However, it does not account for the interactions among $X$’s and the computation may not be feasible when the number of $X$’s is large.

4.3 Projection pursuit regression

In project pursuit regression, we model $f(X)$ using form
   $$f(x) = \sum_{k=1}^{m} g_k(\beta_k^T x),$$
where both $g_k$ and $\beta_k$ are unknown. For identifiability, we require $\|\beta_k\| = 1$. When $\beta_k^T X = X_k$, this model becomes the generalized additive model. However, the project pursuit regression allows the interactions among feature variables and in fact, if $m$ is large enough, such an expression can be used to approximate any continuous function. When $m$ is 1, this becomes the single index model which is commonly used in econometrics.

Model fitting in project pursuit regression is carried out in a forward step-wise way. We start $m = 1$ to first fit model $f(X) = g_1(\beta_1^T X)$. To do this, the backfitting procedure can be applied by iteratively estimating $\beta_1$ then $g_1$. Particularly, given $g_1$, we approximate $g_1(\beta_1^T X)$ by
   $$g_1(\{\beta_1^{old}\}^T X) + g'_1(\{\beta_1^{old}\}^T X)(\beta_1 - \beta_1^{old})^T X$$
then minimize
\[ \sum_{i=1}^{n} \left( Y_i - \left\{ g_1(\{\beta_1^{\text{old}}\}^T X_i) + g'_1(\{\beta_1^{\text{old}}\}^T X_i) (\beta_1 - \beta_1^{\text{old}})^T X_i \} \right\}^2 \]
to obtain \( \beta_1 \). Then fixing \( \beta_1 \), we estimate \( g_1 \) by regressing \( Y_i \) on \( \beta_1^T X_i \) via smoothing splines or other smoothing nonparametric regression methods. We cycle till the convergence of the estimators for \( \beta_1 \) and \( g_1 \). We then move to the model with an additional term \( g_2(\beta_2^T X) \). This can be done similarly by replacing \( Y_i \) with \( Y_i - g_1(\beta_1^T X_i) \). Such a procedure can be carried out by adding more additive components but stops when the next added term does not appreciably improve the prediction performance of the model.

The projection pursuit is not restricted to regression model. Its applications also include density estimation and are reflected in the neural networks discussed before. In different context, a close and similar area to the projection pursuit is called central subspace, which is defined as a linear space containing some linear combinations of \( X \) explaining the dependence between \( Y \) and \( X \), for instance, \( \{\beta_1^T X, \ldots, \beta_m^T X\} \) in the current models. There has been a lot of work on identifying central subspaces but the earliest one is the so-called slice inverse regression as introduced by Duan and Li (1991).
Chapter 5

DIRECT LEARNING: NONPARAMETRIC APPROACHES

5.1 Nearest neighbor methods

One of the most prototype methods for classification is the nearest neighborhood method. Suppose $Y$ denotes the class label. To predict the class label for a given feature value $x$, we simply search within the observations $(X_1, Y_1), ..., (X_n, Y_n)$ and locate a number of ones whose feature values are closest to $x$. The majority of the corresponding $Y_i$ for these neighbors is set to be the predicted value for $x$. The number of neighborhood is often fixed at some positive integer $k$; so this method is called the $k$-nearest neighborhood method.

Although this method is simple, it has been successful in many applications including handwritten digits, satellite image scenes and EKG patterns, where the decision boundary is very irregular. When $k$ decreases, the training error is close to zero but the variance becomes high. However, a famous result of Cover and Hart (1967) shows that asymptotically the error rate of the 1-nearest neighborhood is never more than twice the Bayes error rate.

One essential issue in this method is how to define distances between any two points in the feature space. Normally one will use the Euclidean distance for continuous feature variables and use Hamming distance for categorical one. Some other metrics can also be used, especially when feature variables lie on some manifold.
5.2 Kernel methods

Kernel methods belong to direct learning methods where one uses smoothing techniques to estimate target $f(x)$. Particularly, such smoothing is a way of local smoothing; that is, to estimate the value of $f(x)$ at some point $x = x_0$, most likely, the local observations where $X_i$ is close to $x_0$ are used for interpolate $f(x_0)$, where the localization is determined by some kernel weighting function. In some sense, the kernel methods are similar to the nearest neighborhood method described previously, except that the neighborhood is defined more softly and smoothly in the kernel methods.

In a regression setting, to estimate $f(x_0) = E[Y|X = x_0]$, a typical kernel estimator is the so-called Nadaraya-Watson kernel estimator:

$$\hat{f}(x_0) = \frac{n^{-1} \sum_{i=1}^{n} K_h(X_i, x_0)Y_i}{n^{-1} \sum_{i=1}^{n} K_h(X_i, x_0)},$$

where $K_h(x)$ is a kernel function with bandwidth $h$ (it can be a vector $(h_1, ..., h_p)$). Sometimes, we choose $K_h(x) = (h_1 h_2 \cdots h_p)^{-1} K_1(|x_1|/h_1) \times \cdots \times K_p(|x_p|/h_p)$ with $K_1(\cdot), ..., K_p(\cdot)$ being possibly different kernel function (positive and integrable) in one-dimensional space; but usually, we let $K_h(x) = h_1^{p} K_1(\|x\|/h_1)$ where $\|\cdot\|$ is some norm defined in $R^p$ space. In most of practice, $K_1, ..., K_p$ are chosen to be either the Gaussian kernel $(2\pi)^{-1/2} \exp\{-x^2/2\}$ or the Epanechnikov kernel $0.75I(|x| \leq x)(1- x^2)$. The choice of the bandwidths $h$ can be adaptive to $x_0$. Generally, large bandwidths result in lower variances but higher bias. When $x_0$ is on the boundary of $X$’s domain, the above kernel estimation can be large biased due to the fact that the local neighborhood contains less points.

There have been a large number of theoretical results developed for the kernel estimation in the past literature. Here, we focus on the issue of variance and bias trade-off in the kernel estimation. Consider the case that $X$ is one-dimensional and for simplicity, we only examine the numerator in the definition of $\hat{f}(x_0)$, i.e.,

$$\hat{g}(x_0) = n^{-1} \sum_{i=1}^{n} K_h(|X_i - x_0|/h)Y_i.$$

Assume $Var(Y_i|X_i) = \sigma^2$. Note that

$$E[\hat{g}(x_0)] = E[K_h(|X_1 - x_0|/h)f(X_1)]$$

and its variance

$$Var[\hat{g}(x_0)] = n^{-1} Var(K_h(|X_1 - x_0|/h)Y_1).$$
5.2. KERNEL METHODS

\begin{equation}
E[K_h(|X_1 - x_0|/h)] = \int_x h^{-1} K_1((x - x_0)/h) g(x)p(x) dx,
\end{equation}

where \( p(x) \) is the smooth density of \( X_1 \). After transforming \( x_1 = x_0 + hz \) and the Taylor expansion, we obtain

\begin{equation}
E[K_h(|X_1 - x_0|/h)g(X_1)] = \int_z K_1(z) g(hz + x_0)p(hz + x_0) dz
= \int_z K_1(z) \{ g(x_0)p(x_0) + h(gp)'|_{x=x_0} + h^2(gp)''|_{x=x_0}/2 + \ldots \} dz.
\end{equation}

Since the kernel function is symmetric, we thus have that the above term is equal to \( g(x_0)p(x_0) + O(h^2) \). Similarly, we can show

\begin{equation}
E[K_h(|X_1 - x_0|/h)^2 g(X_1)] = h^{-1}\{ g(x_0)p(x_0) + O(h^2) \}.
\end{equation}

Following these result, we conclude that

\begin{equation}
E[\hat{g}(x_0)] = f(x_0)p(x_0) + O(h^2)
\end{equation}

and

\begin{equation}
Var(\hat{g}(x_0)) = (nh)^{-1}\sigma^2(p(x_0) + O(h^2)) + (nh)^{-1}(f(x_0)^2p(x_0) + O(h^2)) - n^{-1}\{ f(x_0)p(x_0) + O(h^2) \}^2
= O((nh)^{-1}).
\end{equation}

Actually, for the Nadaraya-Watson estimator, we obtain similar results:

\begin{equation}
E[\hat{f}(x_0)] = f(x_0) + O(h^2), \ Var(\hat{f}(x_0)) = O((nh)^{-1}).
\end{equation}

This confirms that when smaller bandwidth is used, the kernel estimator has smaller bias but larger variance. Finally, the bias-variance trade-off can be quantified using the mean square error given as

\begin{equation}
\{E[\hat{f}(x_0)] - f(x_0)\}^2 + Var(\hat{f}(x_0)) = O(h^4) + O((nh)^{-1}).
\end{equation}

Thus, the optimal bandwidth in terms of minimizing this quantity is in the order \( n^{-1/5} \), which is the optimal bandwidth in one-dimensional kernel estimation. For general feature space in \( \mathbb{R}^p \), this optimal bandwidth is given by \( n^{-1/(4+p)} \).
As mentioned before, the above kernel estimator, which relies on the local average, has large bias when \(x_0\) is close to the boundary. To solve this issue, an alternative estimator is called the local linear estimator, which fits a weighted linear regression locally. To see this, we first notice that the previous kernel estimator is essentially minimizing the following weighted least square problem:

\[
\sum_{i=1}^{n} K_h(|X_i - x_0|) \{Y_i - \alpha(x_0)\}^2,
\]

where \(\alpha(x_0)\) is a constant parameter. Essentially, we fit a locally constant line to data. Then a local linear estimator is to minimizing

\[
\sum_{i=1}^{n} K_h(|X_i - x_0|) \{Y_i - \alpha(x_0) - \beta(x_0)(X_i - x_0)\}^2;
\]

that is, instead of fitting a constant locally, we fit a linear line locally. The obtained \(\hat{\alpha}(x_0)\) is the local linear estimator for \(f(x_0)\) and \(\hat{\beta}(x_0)\) is actually a kernel estimator for the first derivative of \(f(x_0)\). Because of the approximation using the linear estimator locally, it is easy to see that the local linear estimator corrects bias up to the first order. A further generalization of the local linear estimator is the following local polynomial regression, which minimizing

\[
\sum_{i=1}^{n} K_h(|X_i - x_0|) \left\{Y_i - \beta_0(x_0) - \beta_1(x_0)(X_i - x_0) - ... - \beta_k(x_0)(X_i - x_0)^k/k! \right\}^2.
\]

Thus, the derived estimator \(\hat{\beta}_0(x_0)\) for \(f(x_0)\) corrects bias up to the \(k\)th order. Of course, there is price paid for such bias reduction and that is increased variance.

There has been a great amount of work on the latter kernel estimators. Most of theory rely on the delicate and tedious Taylor expansion. Some helpful conclusions for practical use include: local linear estimators help bias reduction dramatically at the boundaries while local quadratic fits do little at the boundaries for bias but increase the variance a lot. The local polynomials of odd degree dominates those of even degrees. Interesting readers can consult the reference by Fan and Gijbes (1996).

The above methods can be easily generalized to regression problem in multiple dimension feature spaces. However, when the dimension becomes high, local regression becomes less useful due to the curse of dimensionality. Moreover, boundary effects become a much bigger problem in two or higher dimensional space since the fraction of points on the boundary is large. Finally, the visualization of \(\hat{f}(x)\) is also difficulty in higher dimension.

So far, we only consider estimating \(E[Y|X]\), mainly based on the locally weighted least square. In some situations, when \(Y\) is nominal or ordinal, \(f(x)\) is related to conditional density
of \( Y \) given \( X \). Using the least square method may not be efficient. In this case, we can estimate \( f(x) \) via the following local likelihood approach. In this method, the main idea is to maximize the observed log-likelihood locally. For example, suppose that the density of \( Y \) given \( X \) is given by \( g(Y, f(X)) \). Then a local log-likelihood function is defined as

\[
\sum_{i=1}^{n} K_h(|X_i - x_0|) \log g(Y_i, f(x_0)).
\]

We can maximize the above function to estimate \( f(x_0) \). Similarly, we can generalize this estimation to polynomial approximation by replacing \( g(Y_i, f(x_0)) \) in the above expression with \( g(Y_i, \beta_0(x_0) + \beta_1(x_0)(X_i - x_0) + ... + \beta_k(x_0)(X_i - x_0)^k/k!) \).

The local likelihood function has been applied to many non-continuous or non-regular settings, for example, censored data.

There are other local methods based on kernel approximation, including local median, local polynomial in least absolute deviations and etc.

5.3 Sieve methods

Different from the previous local estimation approaches, sieve estimation is a way of directly learning \( f(x) \) in a global sense. To be explicit, this method estimates \( f(x) \) via a linear approximation of basis functions,

\[
\sum_{k=1}^{m} \beta_k h_k(x),
\]

where \( h_1(x), ..., h_m(x) \) are basis functions. That is, we approximate the target function globally using a series of simple approximations. The choices of basis functions include trigometric functions, polynomials, splines, and wavelets etc. Particularly, the last two basis functions are most popular in learning literature, which we will discuss below. Again, we start with as simple case assuming \( X \) from one-dimensional feature space.

Splines are essentially piece-wise polynomials which require some smoothness at joint points. To be more specific, suppose \( X \in [0, 1] \) and we call joint points as knots and denote as \( 0 < t_1 < t_2 < ... < t_s < 1 \). Then a spline function is some polynomial in \([0, t_1], [t_1, t_2], ..., \) but this function is assumed to be continuous or even have higher continuous derivatives at \( t_1, t_2, ..., t_s \).

When the knots are fixed, such splines are sometimes called regression splines. It turns out that another way to represent these splines can be constructed through \( x^{k-1} \) or \((x - t_l)^{k-1} \) for
a set of \( k \)'s and \( l = 1, \ldots, s \). However, these expressions, although mathematically simple, may not be useful for practical computation. A more computationally useful spline representation is called B-spline basis, which is computed using the following iterative equation:

\[
B_{i,k}(x) = \frac{x - t_i}{t_{i+k-1} - t_i} B_{i,k-1} + \frac{t_{i+k} - x}{t_{i+k} - t_{i+1}} B_{i+1,k-1}(x)
\]

and \( B_{i,1}(x) = I(t_i \leq x < t_{i+1}) \). Actually, in the B-spline approximation, we can allow the knots to be duplicated (more duplication results in less smoothness at the knots). In theory, the B-splines can be used to approximate any function with sufficient smoothness, such as the weakly-differentiable functions in Sobolev spaces.

Wavelets smoothing is another sieve approximation, which receives extensive applications in signal processing and compression. This method relies on constructing a series of wavelet basis functions, which can capture signals in both time and frequency domain (note that traditional Fourier analysis only approximates functions in frequency domain). Its mathematical definition is as follows. Let \( \phi(x) \) be a mother wavelet such like the Haar basis \( I(x \in [0,1]) \) or the Daubiechius wavelets or symmlet wavelets. Let \( \phi_{j,k}(x) = 2^{j/2}\phi(2^j x - k) \) and let \( V_j \) be the space spanned by \( \{\phi_{j,k} : k = \ldots, -1, 0, 1, \ldots\} \). Due to the choice of \( \phi \), \( V_0 \subset V_1 \subset V_2 \subset \ldots \) and the limit space is \( L_2 \)-space. We can understand that the projection of any function \( f(x) \) on \( V_j \) as the signal in \( f(x) \) up to \( j \)th level resolution. Furthermore, if we decompose \( V_{j+1} \) into the direct summation of \( V_j \) and \( W_j \), then

\[
V_j = V_0 \oplus W_0 \oplus W_1 \oplus \ldots \oplus W_j.
\]

Thus, the projection of \( f(x) \) on \( W_k \) can be treated as the details seen at the \( k \)th level resolution. In other words, the wavelet approximation is equivalent to decompose the raw function (signal) into the details at a series of increasing resolution levels, an analysis called a multiresolution analysis.

The details at high resolution levels are very likely due to high-frequency noises in the signals so should be discarded (called denoising process). This is equivalent to shrinking the wavelet coefficients associated with the projection a high resolution levels towards zeros. A popular method for such a shrinkage is called SURE shrinkage (Stein Unbiased Risk Estimation) which adds a \( L_1 \)-penalty to the wavelet coefficients:

\[
\min_{\theta} \|Y - W\theta\|^2 + \lambda \sum |\theta_j|,
\]

where \( W \) is the wavelet transformation matrix. Since \( W \) is orthonormal, this leas to

\[
\hat{\theta}_j = \text{sign}(Y_j^*) \left( |Y_j^*| - \lambda \right)^+,
\]
where $Y_j^*$ is the $j$th component of $W^{-1}Y$. The choice of $\lambda$ is often set to $\sigma \sqrt{2 \log N}$, where $\sigma$ is an estimate of the standard deviation of the noise, and $N$ is the number of data points. The inverse of $W$ can be calculated using a clever pyramidal scheme, which is even faster than the fast Fourier transform.

5.4 Tree-based methods

Tree-based methods can be considered as another type of sieve approximation for estimating $f(x)$. In these methods, $f(x)$ is approximated by a linear combination of high-order interactions of dichotomized functions $I(x_{(j)} < t_k)$ or $I(x_{(j)} > t_k)$ where $x_{(j)}$ is the $j$th component of $x$ and $t_k$ is the dichotomization point. However, such an approximation is performed in a sequential order. For a regression tree in estimating $f(x) = E[Y|X = x]$, we provide details in the following.

Starting with all the data, we consider partition along the $j$th component $X_{(j)}$ and determine the split point $s$ to minimizing

$$
\min_{c_1} \sum_{i=1}^{n} (Y_i - c_1)^2 I(X_{i(j)} \leq s) + \min_{c_2} \sum_{i=1}^{n} (Y_i - c_2)^2 I(X_{i(j)} > s).
$$

Then we perform a greed search for $j$ and $s$ so that the above function attains minimal. In other words, we look for the optimal component and the optimal dichotomization so that the total mean square errors are minimized. Now suppose that this optimal partition is obtained. Next, within each partitioned rectangle $\{x : I(x_{(j)} \leq s)\}$ and $\{x : I(x_{(j)} > s)\}$, we now search for another component and split point in order to minimize the total mean square errors within each rectangle. We continue such partitions for $m$ steps.

Obviously, this tree can grow to the largest tree when each branch contains only one observation. However, such largest tree is not desirable as it causes overfitting the data. Therefore, there should be some way to determine when the tree growth should stop. An effective strategy in pruning a tree is based on cost-complexity trade-off. For a given tree, suppose that it has $m$ nodes at the end (in other words, each node represents the partitioned rectangle at the end). We let $V_k$ denote the within rectangle variability and $N_k$ be the number of observations in this rectangle. Then a cost-complexity can be defined as

$$
\sum_{k=1}^{m} N_k V_k + \alpha m.
$$
In other words, when a tree grows, the first term is decreasing but the second term increases so as to penalize a complex tree. The constant $\alpha$ balances the trade-off between these quantities.

The same partition idea can be carried out for dichotomous outcome, which results in the so-called classification tree. The difference is that choosing partition is based on minimizing some different loss function in the classification tree. Such loss function can be the misclassification error (the proportion of the observations which are labelled different from the majority class in the partitioned rectangle), the Gini index, $\sum_{k=1}^{K} \hat{p}_k(1 - \hat{p}_k)$, where $\hat{p}_k$ is the proportion of the observations labelled as class $k$, and the cross-entropy or deviance, $\sum_{k=1}^{K} \hat{p}_k \log \hat{p}_k$.

Recently, another effective classification method has been developed based on classification tree and it is termed as random forest. Random forest is an ensemble method which uses recursive partitioning to generate many trees and then aggregate the results, where each tree is independently generated using a bootstrap sample of the data. Because of such randomness and aggregation, this method is robust against over-fitting and missing observations and can handle large numbers of input feature variables. The method is easy to parallelize as the forest is created using the observations not selected in each bootstrap sample. However, it is computationally slow and may use lots of memory because a large number of trees are stored.

The algorithm in random forest can be briefly described below. Suppose we want to grow $N$ trees. We randomly draw $N$ bootstrap samples from the original data. For each of the bootstrap samples, grow an unpruned classification or regression tree with the following modification: at each node of the tree, we randomly sample $s (s << p)$ of the feature variables and choose the best split from these variables. Finally, we predict new data by aggregating the predictions of $N$ trees.

Using a random forest, we can also calculate the misclassification error rates in the data not in the bootstrap sample and this estimate is quite accurate for the true error rate when enough trees are grown. Additionally, a random forest can also be used to assess variable importance and proximity measure between any two observations (the fraction of trees in which two elements are in the same terminal node).

5.5 Multivariate adaptive regression splines

Multivariate adaptive regression splines, abbreviated as MARS, is an adaptive procedure for regression and is useful for high-dimensional problem. This method uses expansions in piecewise
5.5. MULTIVARIATE ADAPTIVE REGRESSION SPLINES

linear basis functions of form \((X_{(j)} - t)_+\) and \((t - X_{(j)})_+\), where \(t\) takes values of the observed \(X_{(j)}\)'s for \(j = 1, ..., p\). Using these basis functions, we build regression models via a forward stepwise linear regression:

\[
f(X) = \beta_0 + \sum_{k=1}^{m} \beta_k h_k(X),
\]

where \(h_k(X)\) is in form of \((X_i(j) - t)_+\) and \((t - X_{(j)})_+\) and the coefficients are estimated using the least square regression. At each stage, we add to the model the best term in a form of \(h_k(X)(X_{(j)} - t)_+\) and \(h_k(X)(t - X_{(j)})_+\) which gives the largest decrease in training error. We continue till the preset maximal number of terms in the model is reached.

The final model typically overfits data so a backward deletion procedure is applied. In the backward procedure, a term whose removal causes the smallest increase in residual square errors is deleted, producing the best model for each model size. The best model size is then selected via some general cross-validation, which we will introduce later.

The reason of using these piecewise linear basis functions is due to their local approximation property, similar to wavelets. This is seen in the product of these functions where only a small part around observed data is non-zero. The second important advantage of using these basis functions is about computation. This is said in more detail in HFT book.
Chapter 6

INDIRECT LEARNING

In this chapter, we introduce indirect learning methods, which estimate \( f(x) \) by minimizing some sensible loss function instead of estimating \( f(x) \) directly. This is often useful when the true \( f(x) \) associated with given loss functions is not explicit in terms of the joint distribution of \((Y, X)\). In this chapter, we focus on the classification problem where \( Y \) has two categories (value -1 and 1).

6.1 Separate hyperplane

A separate hyperplane is equivalent to finding a linear function \((x^T \beta + \beta_0)\) with constraint \( \|\beta\| = 1 \) of feature variables which can separate two classes well in some sense. We will describe two separate hyperplane methods: Rosenblatt’s perceptron learning algorithm and optimal separating hyperplane.

The perceptron learning algorithm aims to find a separating hyperplane which minimizes the distance of misclassified points to the decision boundary. Suppose that the decision rule is that we classify subject into 1 if \( x^T \beta + \beta_0 > 0 \) and −1 otherwise. Then any misclassified points are those subject \( i \) from \( 1, \ldots, n \) such that \( Y_i(x_i^T \beta + \beta_0) < 0 \). Then the summed distances from these points to the decision boundary are

\[
D(\beta, \beta_0) = \sum_{i=1}^{n} \left\{ Y_i(x_i^T \beta + \beta_0) \right\}^{-}
\]

where \( x^- = \max(0, -x) \). A stochastic gradient descent algorithm is used to minimize this function, where the gradients are give as

\[
\frac{\partial D(\beta, \beta_0)}{\partial \beta} = \sum_{i=1}^{n} Y_i x_i I \left\{ Y_i(x_i^T \beta + \beta_0) < 0 \right\}, \quad \frac{\partial D(\beta, \beta_0)}{\partial \beta_0} = \sum_{i=1}^{n} Y_i I \left\{ Y_i(x_i^T \beta + \beta_0) < 0 \right\}.
\]
In this algorithm one updates \((\beta, \beta_0)\) after visiting each misclassified subjects using \((\beta, \beta_0) + \rho(Y_i X_i, Y_i)\) where \(\rho\) is a step size (called learning rate). It can be shown that the algorithm converges to a separating hyperplane in finite steps if such a separating hyperplane does exist. However, there are a number of problems with this algorithm as well: first, when data are separable, there are many solutions depending on start values; convergence can be slow; the algorithm will not converge if data are not separable.

To obtain a unique separating hyperplane, a method has also been developed to find the optimal separating hyperplane (Vapnik, 1996). This method aims to maximize the signed distance from the decision boundary to the closest point from either class. If we let \(C\) denotes such distance, then such an optimization problem is

\[
\max_{\beta, \beta_0, ||\beta|| = 1} C \quad \text{subject to} \quad Y_i (X_i^T \beta + \beta_0) \geq C, \ i = 1, \ldots, n.
\]

Note that setting \(||\beta|| = 1\) in this optimization problem is arbitrary so we can constrain this norm to any positive constant, say \(1/C\). After reparameterizing \(\beta_0\) and \(\beta_0 ||\beta||\), the problem is equivalent to

\[
\min_{\beta, \beta_0} ||\beta|| \quad \text{subject to} \quad Y_i (X_i^T \beta + \beta_0) \geq 1, \ i = 1, \ldots, n,
\]

or equivalently,

\[
\min_{\beta, \beta_0} \frac{1}{2} ||\beta||^2 \quad \text{subject to} \quad Y_i (X_i^T \beta + \beta_0) \geq 1, \ i = 1, \ldots, n.
\]

This is a quadratic criterion with linear inequality constraints so is a convex optimization problem. The corresponding Lagrange function is

\[
\frac{1}{2} ||\beta||^2 + \sum_{i=1}^{n} \alpha_i \left\{ Y_i (X_i^T \beta + \beta_0) - 1 \right\}
\]

subject to constraints \(\alpha_i \geq 0, \ i = 1, \ldots, n\). Setting the derivatives to zeros, we obtain

\[
\sum_{i=1}^{n} \alpha_i Y_i X_i = \beta, \quad \sum_{i=1}^{n} \alpha_i Y_i = 0.
\]

After plugging it back to the Lagrange function, we obtain the so-called Wolfe dual

\[
\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j Y_i Y_j X_i^T X_j \quad \text{subject to} \quad \alpha_i \geq 0.
\]

This is a simple convex optimization problem which standard softwares can solve. Furthermore, by the Kusush-Kuhn-Tucker conditions, the solution also satisfies

\[
\alpha_i \left\{ Y_i (X_i^T \beta + \beta_0) - 1 \right\} = 0, \quad i = 1, \ldots, n.
\]
(Read reference on Convex Optimization.) Therefore, if $\alpha_i > 0$, then $Y_i(X_i^T \beta + \beta_0) = 1$ so subject $i$ is on the boundary of a slab closest to the separate hyperplane; otherwise, $\alpha_i = 0$ and $Y_i(X_i^T \beta + \beta_0) > 1$ so subject $i$ is away from the boundary. Additionally, the previous derivation shows $\beta = \sum_{i=1}^{n} \alpha_i Y_i X_i = \sum_{\alpha_i > 0} Y_i X_i$; thus, $\beta$ is determined by the points on the boundary of the slab, which are called the support points. Once the separate hyperplane is obtained, the classification rule is simply $\text{sign}\{x^T \hat{\beta} + \hat{\beta}_0\}$.

The optimal separating hyperplane is unique if the data are truly separable. Since the hyperplane only depends on a few support points, it is more robust to model misspecification or outliers. This is one advantage of this method over discriminant analysis. However, when the data are not separable, there will be no feasible solution and an alternative method is needed. Such a method is known as the support vector machine, which allows for overlap and will be introduced next.

### 6.2 Support vector machine

Support vector machine is one of the most popularly used learning method in practice. The advantages of this method include allowing nonseparable data, computational simplicity and good prediction performance. We consider two types of this method: in first type, the input is just the feature space and the method is called support vector classifier; in the second type, the input is some basis functions associated with each data point and the method is called support vector machine.

#### 6.2.1 Support vector classifier

Recall that in the method of finding the optimal separating hyperplane, we try to find a hyperplane separating the data in two classes so that their distances from the hyperplane is at least some constant $C$. In other words, the two classes of data points are well separated and lie out of a band which centers around the hyperplane and the band width (called margin) is $2C$. We choose the optimal plane so that this margin is the largest. However, when the data points are not separable, this is impossible and we should allow some points on the wrong side of the hyperplane. To realize it mathematically, we relax the strict constraint $Y_i(X_i^T \beta + \beta_0) \geq C$ by changing it to

$$Y_i(X_i^T \beta + \beta_0) \geq C(1 - \xi_i),$$
where $\xi_i \geq 0, i = 1, \ldots, n$, are called slack variables.

We note that $\xi_i$ also represents the proportion amount by which the prediction $X_i^T \beta + \beta_0$ is on the wrong side of the margin of the band. Therefore, one possibility is to set a bound for the total proportion amount, $\sum_{i=1}^{n} \xi_i$. Under such a bound, we then look for the band with the large margin. In other words, we search for a hyperplane with largest margin separating data points so that the proportion amount of prediction on the wrong sides of the margins is controlled under some bound, that is, the largest separation by allowing some proportion of misclassification rates.

Rigourously writing, this becomes the following optimization problem

$$
\max_{\beta, \beta_0, \|\beta\|=1} C \quad \text{subject to} \quad Y_i(X_i^T \beta + \beta) \geq C(1 - \xi_i), \quad \xi_i \geq 0, \quad \sum_{i=1}^{n} \xi_i \leq \text{constant}.
$$

Using the same transformation as in the previous section, we obtain an equivalent problem

$$
\min \frac{1}{2} \|\beta\|^2 \quad \text{subject to} \quad Y_i(X_i^T \beta + \beta) \geq (1 - \xi_i), \quad \xi_i \geq 0, \quad \sum_{i=1}^{n} \xi_i \leq \text{constant}.
$$

Again, this is a convex optimization problem with linear constraints. An equivalent problem is

$$
\min \frac{1}{2} \|\beta\|^2 + \gamma \sum_{i=1}^{n} \xi_i \quad \text{subject to} \quad Y_i(X_i^T \beta + \beta) \geq (1 - \xi_i), \quad \xi_i \geq 0,
$$

where $\gamma$ replaces the constant before. The separate case corresponds to $\gamma = \infty$.

The Lagrange function is

$$
\frac{1}{2} \|\beta\|^2 + \gamma \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \left\{Y_i(X_i^T \beta + \beta_0) - (1 - \xi_i)\right\} - \sum_{i=1}^{n} \mu_i \xi_i,
$$

with constraints $\alpha_i \geq 0, \mu_i \geq 0$. Its derivatives with respect to $(\beta, \beta_0)$ and $\xi_i$ yield

$$
\beta = \sum_{i=1}^{n} \alpha_i Y_i X_i, \quad 0 = \sum_{i=1}^{n} \alpha_i Y_i, \quad \alpha_i = \gamma - \mu_i.
$$

After substituting back to the Lagrange function, we obtain the dual problem

$$
\max_{\alpha, \mu} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j Y_i Y_j X_i^T X_j,
$$

subject to constraints

$$
0 \leq \alpha_i \leq \gamma, i = 1, \ldots, n, \quad \sum_{i=1}^{n} \alpha_i Y_i = 0.
$$

This can be solved using standard softwares for convex optimization (Murray et al, 1981).
From the Karush-Kuhn-Tucker conditions, we obtain

\[ \alpha_i \left\{ Y_i (X_i^T \hat{\beta} + \hat{\beta}_0) - (1 - \hat{\xi}_i) \right\} = 0, \quad \hat{\mu}_i \hat{\xi}_i = 0, \quad Y_i (X_i^T \hat{\beta} + \hat{\beta}_0) - (1 - \hat{\xi}_i) \geq 0. \]

We thus conclude that if \( \hat{\alpha}_i \in (0, \gamma) \), then \( Y_i (X_i^T \hat{\beta} + \hat{\beta}_0) = 1 - \hat{\xi}_i \); but under this case, \( \hat{\mu}_i > 0 \) so \( \hat{\xi}_i = 0 \); therefore, \( Y_i (X_i^T \hat{\beta} + \hat{\beta}_0) = 1 \) so such data points lie on the margins of the band; for those points inside the band, \( \hat{\xi}_i > 0 \) and \( \hat{\alpha}_i = \gamma \). Now, since

\[ \hat{\beta} = \sum_{\hat{\alpha}_i > 0} \hat{\alpha}_i Y_i X_i, \]

we conclude that \( \hat{\beta} \) is determined by the points within or on the boundary of the band (these points are called support vectors). Furthermore, \( \hat{\beta}_0 \) can also be determined using the first equation from the Karush-Kuhn-Tucker conditions.

### 6.2.2 Support vector machine

So far, the support vector machine targets a linear boundary of feature spaces, which may not be practically useful if the separation is actually nonlinear. However, the above approach can be easily generalized to obtain nonlinear boundaries if we replace feature space \( X_i \) by some basis functions evaluated at \( X_i \). The procedure is the same as before. Suppose that we choose basis functions \( h(x) = (h_1(x), ..., h_m(x))^T \) then the classification boundary is given by

\[ f(x) = h(x)^T \beta + \beta_0. \]

Following the previous derivation, the dual problem becomes maximizes

\[ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j Y_i Y_j < h(X_i), h(X_j) >, \]

subject to constraints

\[ 0 \leq \alpha_i \leq \gamma, \quad \sum_{i=1}^{n} \alpha_i Y_i = 0, \]

where \( < x, y > = x^T y \). Then the classification boundary is given by

\[ f(x) = \sum_{i=1}^{n} \alpha_i Y_i < h(x), h(x_i) > + \beta_0. \]

Let \( K(x, x') = < h(x), h(x') >, \) which is called a kernel function. The above calculation and classification rule only depend on the kernel function. Therefore, in this support vector machine method, one only needs to specify the kernel function for calculation. Some popular choices
of the kernel function in the support vector machine literature include the polynomial kernel, $K(x, x') = (1+ <x, x'>)^d$, the radial basis, $K(x, x') = \exp\{-\|x - x\|^2/c\}$, and the neural network, $K(x, x') = \tanh(\theta_1 <x, x'> + \theta_2)$.

The constant $\gamma$ in the support vector machine governs the smoothness of the boundary. A large value of $\gamma$ gives a wiggly boundary so could overfit training data.

Another extension as observed here is that we can even allow feature space belongs to some Hilbert space, for example, $X_i$ represents subject’s profile over time. The above procedure still applies if we replace $<x, x'>$ by the inner product in the Hilbert space. In other words, the SVM method applies to the case that one uses profile information to classify subjects.

6.2.3 Casting SVM into a penalized learning

The way we introduced the SVM method is more based on intuitive thinking that one tries to separate two classes in some maximal sense. In fact, the SVM can be translated into an empirical risk minimization problem as discussed in Chapter 2.

Specifically, we define a loss function $L(y, x) = (1 - yx)^+$. We aim to minimize the empirical loss but subject to a constraint $\|\beta\|$ bounded by some constant. Equivalently, we minimize

$$\sum_{i=1}^{n} \{1 - Y_if(X_i)\}^+ + \lambda\|\beta\|^2/2,$$

where $\lambda$ is a constant. By setting $\xi_i = \{1 - Y_if(X_i)\}^+$ and letting $\gamma = 1/\lambda$, we can easily show that this minimization is equivalent to maximizing the objective function in the previous section. In this way, we cast the SVM as a regularized empirical risk minimization.

Following this framework, we can also obtain the SMV for other problems, including multiclass problems and regression problems. The former essentially solves many two-class SVM problems. For the latter, the basic idea is to replace the loss function $\{1 - yf(x)\}^+$ by a different loss $V(y - f(x))$, where $V(t) = (|t| - \epsilon)I(|t| \geq \epsilon)$ for some small constant $\epsilon$ which allows some small prediction errors. Note that the loss function uses the linear contribution of the absolute residuals so the fit is less sensitive to outliers (the same advantage in Huber estimation).

6.3 Function estimation via regularization

Regularization methods aim to estimate $f(x)$ by simultaneously regularizing the complexity allowed in estimation through imposing large penalty for those undesired estimators. In a
6.3. FUNCTION ESTIMATION VIA REGULARIZATION

simple regression problem, to estimate $f(x)$, we consider minimizing the following penalized residual sum of squares:

$$\sum_{i=1}^{n} (Y_i - f(X_i))^2 + \lambda \int [f''(x)]^2 dx,$$

where $\lambda$ is a fixed smoothing parameter. In this objective function, the first term measures the fit performance of $f(x)$; while the second term penalizes curvatures in this function. These two terms are balanced through $\lambda$; otherwise, when $\lambda = 0$, the estimator is any function such that $f(X_i) = Y_i$ resulting in overfitting, when $\lambda = \infty$, the estimator is a linear function which may produce large bias. It can be shown that there exists a unique minimizer which is actually a natural cubic spline with knots at the unique values of the observed $X_1, ..., X_n$. Furthermore, the estimation is equivalent to a ridge regression with these cubic splines being regressors. When $Y$ is not continuous, the same regularization can be applied to the likelihood function by replacing the above least square with the negative log-likelihood function form observed data.

Generally, we can write any regularization methods as

$$\min_{f \in \mathcal{H}} \left[ \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda J(f) \right],$$

where $\mathcal{H}$ is a functional space (usually a Hilbert space) which $f$ is chosen from, $L(y, x)$ is a loss function, and $J(f)$ is a penalty functional for $f$. A general penalty given by Girosi et al. (1995) takes form

$$J(f) = \int \left| \tilde{f}(s) \right|^2 \tilde{G}(s) ds,$$

where $\tilde{f}(s)$ is the Fourier transform of $f$ and $\tilde{G}(s)$ is some positive function that falls off to zero as $\|s\| \to \infty$. In other words, we penalty high-frequency component of $f$. They show that the solutions have form

$$\sum_{k=1}^{K} \alpha_k \phi_k(x) + \sum_{i=1}^{n} \theta_k G(x - X_i),$$

where $\{\phi_k\}$ spans the null space of $J$-operator and $G$ is the inverse Fourier transformation of $\tilde{G}$.

Another important application of the above regularization method is to set $J(f) = \|f\|_{\mathcal{H}_K}$, where $\mathcal{H}_K$ is a reproducing kernel Hilbert space (RKHS) defined based on a positive definite kernel function $K(x, y)$. Specifically, an RKHS is a Hilbert space in which all the point evaluations are bounded linear functionals (unlike $L_2$-space). If we use $\langle, \rangle$ to denote the inner product in this space, then there exists some function $\eta_t$ in this space, such that for any $f$ in this pace,

$$\langle \eta_t, f \rangle = f(t).$$
Then let $K(t, x) = \eta_t(x)$ so it is a positive definite function and called the reproducing kernel in the space for the reason that $< K(t, \cdot), K(s, \cdot) > = K(s, t)$. On the other hand, the Moore-Aronszajn theorem states that for every positive definite function $K(t, s)$, there exists a unique RKHS associated with $K(t, s)$. Since such a kernel function possesses an eigen-expansion

$$K(x, y) = \sum_{i=1}^{\infty} \gamma_i \phi_i(x) \phi_i(y)$$

with $\gamma_i \geq 0$, $\sum \gamma_i^2 < \infty$ and $\phi_1, \phi_2, \ldots$ are the orthonormal basis functions in $\mathcal{H}_K$. Thus, for any function $f \in \mathcal{H}_K$,

$$f(x) = \sum_{i=1}^{\infty} c_i \phi_i(x).$$

The minimization problem is equivalent to minimizing

$$\sum_{i=1}^{n} L(Y_i, \sum_{j=1}^{\infty} \phi_j(X_i)) + \lambda \sum_{j=1}^{\infty} c_j^2 / \gamma_j.$$ 

It can also be shown that the solution is finite dimensional and has form

$$\hat{f}(x) = \sum_{i=1}^{n} \hat{\alpha}_i K(x, X_i),$$

where $\hat{\alpha}$’s minimizes

$$\sum_{i=1}^{n} L(Y_i, \sum_{j=1}^{n} \hat{\alpha}_n K(X_j, X_i)) + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} K(X_i, X_j) \alpha_i \alpha_j.$$ 

Such an expression is a linear of combinations of $K(x, X_i)$, known as the representer of evaluation at $X_i$ in $\mathcal{H}_K$.

The choice of the kernel functions includes $(< x, y > +1)^d$, the Gaussian kernel and etc. We have already seen using such kernel functions in the support vector machine.
Chapter 7

AGGREGATED SUPERVISED LEARNING

Aggregated learning is essentially to combine different learning methods to obtain better prediction rules. A simplest way is to try different learning methods then average their predictions. For example, in classification problem, we may use logistic discriminant, nearest neighborhood, SVM, or classification tree. When a new subject enters, the predicted class of this subjects will be the majority of the predictions from all these methods. This idea is equivalent to model averaging in Bayesian framework.

Another way of aggregating different learning methods is called stacking. We consider squared error loss. Let $\hat{f}_1^{(-i)}, \ldots, \hat{f}_m^{(-i)}$ be the predicted values for subject $i$ using learning methods 1, 2, ..., $m$ based on the data excluding subject $i$. The stacking method is then to find the optimal linear combinations of these predictions to minimize

$$\sum_{i=1}^n \left(Y_i - \sum_{k=1}^m \omega_k \hat{f}_k^{(-i)}(X_i)\right)^2.$$

The final prediction rule is given by

$$\sum_{k=1}^m \hat{\omega}_k f_k(x),$$

where $\hat{\omega}_k$ is the minimizer. This method aggregates all the learning methods based on their cross-validation errors, which will be discussed later and which are good assessment of the prediction performance from each learning method.

A more powerful aggregation way from multiple learning methods is called boosting, which is an iterative procedure to combine the outputs of weak learning methods to produce a powerful committee. Here, a weak learning method means that the error rate is only slightly better than
random guessing. We first look at one binary classification problem \((Y = -1, 1)\). The final output from the boosting method is a prediction rule given as
\[
\text{sign} \left( \sum_{k=1}^{m} \alpha_k \hat{f}_k(x) \right),
\]
where \(\hat{f}_1, ..., \hat{f}_m\) are the estimators from \(m\) learning methods and \(\alpha_1, ..., \alpha_m\) are their corresponding weights. The sequential procedure in the boosting method is a sequential way of updating these weights. The detail of this algorithm (called AdaBoost) is below:
1. We assign each subject \(i\) equal weight \(w_i = 1/n\).
2. From learning method \(k = 1\) to \(m\),
   (a) we apply learning method \(k\) to data using weights \((w_1, ..., w_n)\) to obtain \(\hat{f}_k\),
   (b) compute the error rate as
   \[
   \text{err}_k = \frac{\sum_{i=1}^{n} w_i I(Y_i \neq \hat{f}_k(X_i))}{\sum_{i=1}^{n} w_i}
   \]
   then
   \[
   \alpha_k = \log[(1 - \text{err}_k)/\text{err}_k],
   \]
   (c) recalculate each individual weight as proportional to \(w_i \exp\{\alpha_k I(Y_i \neq \hat{f}_k(X_i))\}\) and send to next classifier.
3. Finally output \(\text{sign} \left( \sum_{k=1}^{m} \alpha_k \hat{f}_k(x) \right)\).

The idea in the above algorithm is that if for \(k\)th classifier, subject \(i\) is misclassified, we then increase this subject’s weight by a factor \(\exp\{\alpha_k\}\) in the \((k + 1)\)th classifier. In other words, we use a new classifier to make up for the misclassification in the current classifier. The AdaBoost procedure sometimes can dramatically increase the performance of even a very weak classifier. Clearly, if we let all the learning methods to be the same (for example, all are classification trees), then every iteration in this procedure is to keep training classification tree to correct misclassified subjects. This may be the reason why we call it boosting. Interesting, such a boosting algorithm is equivalent to minimize an exponential loss \(L(Y, f(X)) = \exp\{-Y f(X)\}\) using forward stagewise additive models, i.e., at \(k\)th stage, we minimize
\[
\sum_{i=1}^{n} \exp\{-Y_i(\hat{f}_{k-1}(X_i) + \beta g(X_i))\}
\]
over \(\beta\) and \(g(x)\) is a function belonging to feasible sets in \(k\)th learning method. The equivalence can be bound in Section 10.4 of HFT book. Moreover, because of this recursive nature and the
forward stagewise learning in the boosting algorithm, this method can be naturally incorporated into classification tree, which is also a recursive learning procedure. The resulting method is called boosting tree.
Chapter 8

MODEL SELECTION IN SUPERVISED LEARNING

In all the learning methods, there are some parameters controlling the complexity of learning methods in order to avoid overfitting. These parameters can be model size in parametric learning and semi-nonparametric learning, the number of observations in nearest neighborhood method, the bandwidth in kernel learning, the number of basis functions in sieve estimation, tree size, and penalty parameters in SVM and regularization methods. However, we discussed very little about the choices of these parameters till now. Specifically, we will discuss a few commonly used approaches to assess learning methods, including Bayesian information criteria, minimum description length and cross-validation. Obviously, there exists many other methods out there to assess learning methods but since they are in the same spirit to balance the prediction accuracy and complexity, we will not review them in this chapter.

With no doubt, assessing learning methods is extremely important in guiding practical use of learning methods and quantifying the performance of final models. A good method for assessing learning performance should result in a parsimonious model with accurate prediction in any external testing data.

8.1 Akaike and Bayesian information criteria

Both AIC and BIC are applicable methods when the learning methods are carried out by maximizing some log-likelihood function and the complexity of methods is reflected in the number of parameters used in the methods. Specially, the AIC is defined as

\[ -2 \log\text{-likelihood} + 2d/n, \]

53
and the BIC is
\[ -2 \log\text{-likelihood} + 2d \log n, \]
where \( d \) is the number of the parameters and \( n \) is the size of data. The former is derived based on the following asymptotic relationship:
\[ -2E[\log P(Y; \hat{\theta})] \approx -\frac{2}{n} E \left[ \sum_{i=1}^{n} \log P(Y_i; \hat{\theta}) \right] + \frac{2d}{n}, \]
where \( P(y; \theta) \) is the working distribution for \( Y \) indexed by parameter \( \theta \) and \( d \) is the number of \( \theta \). Instead, the BIC is motivated by the Bayesian approach for model selection: when a uniform prior is assumed for all the candidate models, the model with the largest posterior probability should have largest conditional probability of the observed data given this model; however, the latter, by a Laplace approximation at the maximum likelihood estimator, is approximated by log-likelihood at \( \hat{\theta} \) subtracting \( d \log n \). We note that the BIC tends to penalize complex models more heavily, giving preference to simpler models in selection. In practice, there is no clear choice between AIC and BIC, since AIC usually chooses models which are too complex when \( n \) goes to infinity while BIC chooses models which are too simple for finite sample. As a note, the BIC method is also equivalent to the minimum description length approach, which was motivated from optimal coding theory.

8.2 Model selection based on VC-complexity

As seen before, the AIC and BIC are only applicable when the loss function is equivalent to the negative log-likelihood function and the complexity of learning models is represented by the number of parameter in consideration. A more general extension is model selection approach based on VC-complexity, which essentially applies to any loss function and any classes of learning model with finite VC-dimensionality. We remark that for parametric models, the VC-dimensionality is equal to the number of independent parameters.

To illustrate idea, we introduce some general notations. We use \( \gamma_n(f) \) to denote
\[ n^{-1} \sum_{i=1}^{n} L(Y_i, f(X_i)). \]
Let \( \mathcal{M}_n \) be a class of models in consideration for estimating \( f(X) \). For any model \( \Omega \) from \( \mathcal{M}_n \), we let \( \hat{f}_\Omega \) be the estimated \( f(x) \) based on this model (the estimation procedure can be either minimizing \( \gamma_n(f) \) over the parameters in model \( \Omega \) or using direct learning method as before).
8.2. MODEL SELECTION BASED ON VC-COMPLEXITY

For example, in parametric learning, \( \Omega \) can be linear regression model with fixed model size; in sieve learning, \( \Omega \) can be a model consisting of smoothing functions with a fixed number of basis functions.

Suppose \( f_0 \) is the minimizer minimizing \( E[\gamma_n(f_0)] \) and we define a natural loss

\[
l(f_0, f) = E[\gamma_n(f)] - E[\gamma_n(f_0)].
\]

For each given model \( \Omega \in \mathcal{M}_n \), we define \( f^*_\Omega \) as the one minimize \( l(f_0, f) \) for \( f \) over \( \Omega \) and this is called an oracle estimator by Donoho and Johnstone. The ideal way of choosing best model \( \Omega \) is to minimize \( l(f_0, f^*_\Omega) \). However, since the true expectation is not calculable in real data, we may consider minimizing an empirical version of \( l(f_0, f) \), which is equivalent to minimizing \( \gamma_n(\hat{f}_\Omega) \).

Unfortunately, the best model minimizing \( \gamma_n(\hat{f}_\Omega) \) may not necessarily minimize \( E[\gamma_n(\hat{f}_\Omega)] \) due to stochastic errors

\[
\gamma_n(\hat{f}_\Omega) - E[\gamma_n(\hat{f}_\Omega)].
\]

To account for such errors, one commonly used method is that instead of minimizing \( \gamma_n(\hat{f}_\Omega) \), we aim to minimize a penalized version

\[
\gamma_n(\hat{f}_\Omega) + \text{pen}_n(\Omega),
\]

where \( \text{pen}_n(\Omega) \) is a penalty function imposed for model \( \Omega \).

Now the question becomes what penalty function, \( \text{pen}_n(\Omega) \), is appropriate. To see this, suppose that the \( \hat{\Omega} \) is the minimizer for the above function. We note that for any \( \Omega \),

\[
l(f_0, \hat{f}_\Omega) = E[\gamma_n(\hat{f}_\Omega)] - E[\gamma_n(f_0)] = -(\gamma_n(\hat{f}_\Omega) - E[\gamma_n(\hat{f}_\Omega)]) + \gamma_n(\hat{f}_\Omega) - E[\gamma_n(f_0)].
\]

Since

\[
\gamma_n(\hat{f}_\Omega) + \text{pen}_n(\hat{\Omega}) \leq \gamma_n(\hat{f}_\Omega) + \text{pen}_n(\Omega) \leq \gamma_n(f^*_\Omega) + \text{pen}_n(\Omega),
\]

we obtain

\[
l(f_0, \hat{f}_\Omega) \leq -(\gamma_n(\hat{f}_\Omega) - E[\gamma_n(\hat{f}_\Omega)]) + \gamma_n(f^*_\Omega) - \text{pen}_n(\hat{\Omega}) + \text{pen}_n(\Omega) - E[\gamma_n(f^*_\Omega)] + l(f_0, f^*_\Omega) \leq |(\gamma_n(f^*_\Omega) - E[\gamma_n(f^*_\Omega)]) - (\gamma_n(\hat{f}_\Omega) - E[\gamma_n(\hat{f}_\Omega)])| - \{\text{pen}_n(\hat{\Omega}) - \text{pen}_n(\Omega)\} + l(f_0, f^*_\Omega).
\]

Therefore, if we can choose a penalty function such that in probability,

\[
|(\gamma_n(f^*_\Omega) - E[\gamma_n(f^*_\Omega)]) - (\gamma_n(\hat{f}_\Omega) - E[\gamma_n(\hat{f}_\Omega)])| \leq \text{pen}_n(\hat{\Omega}),
\]
then it yields
\[ l(f_0, \hat{f}_\Omega) \leq l(f_0, \hat{f}_\Omega^0) + \text{pen}_n(\Omega). \]
Consequently, if we further let \( \text{pen}_n(\Omega) \) uniformly diminishes as data size increases, it is concluded that the model based on the penalized minimization will result in an estimator whose asymptotic loss is equivalent to the best oracle estimator.

The key condition for the penalty function is
\[ |(\gamma_n(f_\Omega^*) - E[\gamma_n(f_\Omega^*)]) - (\gamma_n(\hat{f}_\Omega) - E[\gamma_n(\hat{f}_\Omega)])| \leq \text{pen}_n(\hat{\Omega}), \]
which is equivalent to saying that the penalty dominates the stochastic fluctuation of \( \gamma_n(\cdot) \). However, since \( \hat{\Omega} \) and \( \hat{f}_\Omega \) is unknown, we may wish to study the uniform behavior of
\[ \sup_{\Omega \in \mathcal{M}_n} \sup_{f \in \hat{\Omega}} |(\gamma_n(f_\Omega^*) - E[\gamma_n(f_\Omega^*)]) - (\gamma_n(f) - E[\gamma_n(f)])| - \text{pen}_n(\hat{\Omega}). \]
This is closely related to the stochastic behavior of the empirical process
\[ \sup_{f \in \hat{\Omega}} \{\gamma_n(f) - E[\gamma_n(f)] : f \in \hat{\Omega}\} \]
so concentration inequalities play essential roles. The latter will be described with more details in the second part of this book but we focus on one special case (in fact, the most common situation in statistical learning), where the complexity of models in \( \mathcal{M}_n \) can be characterized by the so-called Vapnick-Chernovenkis (VC) dimension.

The former definition of the VC dimension for a model \( \Omega \), which consists of finite or infinitely many functions for \( f(x) \), is the largest number of points that can be shattered by the subgraphs of these functions. In some sense, the VC dimension characterizes the compactness of the functions in \( \Omega \). A simple example, if the functions \( \Omega \) belong to a linear space with \( q \)-dimension, then the VC dimension is \( q + 1 \). For the VC class, one important result from the empirical process theory is that in probability,
\[ \sup_f |\gamma_n(f) - E[\gamma_n(f)]| \leq a(\text{VC dimension}) \log \frac{n}{\sqrt{n}}, \]
where \( a(\cdot) \) is a deterministic function independent of \( n \). Therefore, from the previous derivation, we can choose the penalty function as
\[ \text{pen}_n(\Omega) = n^{-1/2} a(\text{VC dimension of } \Omega) \log n. \]
In other words, the way to select the best model based on the VC complexity is to minimize
\[ \gamma_n(\hat{f}_\Omega) + n^{-1/2} a(\text{VC dimension of } \Omega) \log n. \]
We note that in parametric models, the VC dimension is equal to one plus the number of parameters, so the above way of model selection is closely related to the BIC method described in the previous section. Using the VC complexity, Vapnik suggested a structural risk minimization for learning. Essentially, one fits a nested sequence of models of increasing VC dimensions and then chooses the model with the smallest value of the above objective function.

### 8.3 Cross-validation

Although the model selection based on VC-complexity is applicable to any types of loss functions, one limitation is that one has to theoretically evaluate the VC dimensionality of each model. Moreover, the penalty function depends on an upper bound controlling the stochastic error of the empirical process, which may not be a sharp bound so may result in over simple models for prediction.

Recall that the goal of model selection in assessing learning methods is to produce a model which has the smallest prediction error when applied to any external data. Because of this goal, the simplest and most widely used method for estimating prediction error is the method of cross-validation. The idea of this method is straightforward. We randomly partition the observed data into two sets of data with one set called training set and the other called testing set. We apply the candidate method/model to the training set to obtain \( \hat{f} \) then evaluate the prediction error in the testing set. We repeat this process a number of times and use the average of all the prediction errors as a criterion to assess the performance of learning methods/models. Such an average is named the cross-validation error. Therefore, the best learning methods/models are chosen to be the ones with the smallest cross-validation error.

There are different ways of partitioning observed data. The simplest way is called the leave-one-out cross validation. In this method, only one subject is in the test set while we use the rest \((n - 1)\)-subjects in the training set. If let \( \hat{f}^{(-i)} \) denote the final estimator for \( f \) based the training set with subject \( i \), then the cross-validation error is given as

\[
\frac{1}{n} \sum_{i=1}^{n} L(Y_i, \hat{f}^{(-i)}(X_i)).
\]

Other ways of partitioning data include \( k \)-fold cross-validation, where \( 1/k \) proportion of the data are randomly selected into the test set. Normally, the larger size in the test set, the larger bias in terms of how accurate the cross-validation error is for the true prediction error; but the lower variance it gives. Usually, five- or ten-fold cross-validation are recommended in practice.
For the leave-one-out cross validation, the cross-validation error can sometimes be approximated by simple expression when the loss is squared error loss and the predicted values for all the subjects are written as $\Sigma Y$, where $\Sigma$ is a $n$ by $n$ matrix. Such an approximation, often called generalized cross-validation, is given as

$$n^{-1} \sum_{i=1}^{n} \left[ \frac{Y_i - \hat{f}(X_i)}{1 - \text{trace}(\Sigma)/n} \right]^2.$$ 

The trace of $\Sigma$ is called the effective number of the parameters. The advantage of the generalized cross-validation is its computational convenience, as only one learning procedure is needed to evaluate the leave-one-out cross validation error.

An alternative way of the cross-validation is to use the bootstrapped sample for learning then average over all the bootstrapped samples. We will not review this method here but refer interested readers to Section 7.11 of HFT book.
Chapter 9

UNSUPERVISED LEARNING

9.1 Principal component analysis

Principal component analysis is one of the most important methods in unsupervised learning, where data contain only feature variables but no outcomes and the goal is to identify the intrinsic distributional structures in given data. The principal component analysis is to identify the so-called principal directions so that the data variability along these directions represents most of the total variability in the data.

Specifically, let $X_1, ..., X_n$ be the observed feature values in $R^p$ from $n$ subjects. We aim to find a matrix $V_{p \times q} = (V_1, ..., V_q)$ where $q$ is the rank of $(X_1^T, ..., X_n^T)^T$ such that $V_1, ..., V_q$ are orthogonal unit vectors and

$$\sum_{i=1}^n \|X_i - \bar{X}_n - VV^T(X_i - \bar{X}_n)\|^2$$

is minimized. Here, $\bar{X}_n$ is the sample mean of $X_1, ..., X_n$. To understand the above expression, we note that $V^T(X_i - \bar{X}_n)$ is the projection of the centered feature $(X_i - \bar{X}_n)$ on the space spanned by the columns of $V$. Therefore, the above minimization is equivalent to finding a space of dimension $q$ so that the projection of the observed feature (after centralization) absorb the maximal variability in the original data.

The solution for the optimal $V$ can be obtained via the singular value decomposition. Particularly, let $X$ be a $n$ by $p$ matrix with each row being $X_1 - \bar{X}_n, ..., X_n - \bar{X}_n$. A singular value decomposition gives

$$X = UDV^T,$$

where $U$ is $n$ by $p$ orthonormal matrix, $V$ is a $p$ by $p$ orthonormal matrix, and $D$ is a diagonal matrix so that its diagonal elements satisfy $d_1 \geq d_2 \geq ... \geq d_p \geq 0$. Then the optimal $V$ is
given as the first \( q \) columns of \( \mathbf{V} \). The first \( q \) columns of \( \mathbf{UD}^T \) are the projection of \( \mathbf{X} \) on these \( q \) principal directions so are called principal components.

From the above singular decomposition, it is easy to show that \( X_iV_1 \) has the highest variance among all the linear combination of the feature variables; \( X_iV_2 \) has the highest variance among all the linear combinations which are orthogonal to \( V_1 \) and so on. Actually, this is the original intuition for conducting principal component analysis.

The choice of the number of principal components is subjective. One often chooses the first \( q \) principal components if their explained variation is above some threshold \( c \) (for example, \( c = 70\% \)) or even more of the total variability in the data; that is,

\[
\frac{d_1^2 + ... + d_q^2}{d_1^2 + ... + d_p^2} \geq c.
\]

When \( q \) is much smaller than \( p \), the first \( q \) principal components are said to sufficiently represent the whole feature variables so can be used in downstream analysis. Thus, the principal component analysis is a useful tool for dimension reduction.

### 9.2 Latent component analysis

Latent component analysis assumes that the data of feature variables are simply multiple indirect measurements of a few latent sources. Therefore, if we can capture the latent sources, we then characterize the most important structure within the data. Moreover, when the number of latent sources is small, they can be used to represent the whole data so we achieve another way of dimension reduction.

Two most important methods in latent component analysis are factor analysis and independent component analysis. In factor analysis, the Gaussian distribution plays an essential role; however, independent component analysis relies on the non-Gaussian nature of the underlying sources.

In factor analysis, we assume that there exists \( q \) (\( q < p \)) latent variables, \( S_1, ..., S_q \), such that

\[
X_{(k)} = a_{k1}S_1 + ... + a_{kq}S_q + \epsilon_k,
\]

where \( a_{k1}, ..., a_{kq} \) are constants and \( \epsilon_k \) is independent noise not explained by latent sources \( S_1, ..., S_q \). We further assume \( S_1, ..., S_q \) are from Gaussian distributions and uncorrelated. As
the result, if denote $A = (a_{kj})_{k=1,...,p; j=1,...,q}$, then it follows

$$\Sigma = AA^T + \text{diag}(\text{var}(\epsilon_1), ..., \text{var}(\epsilon_p)),$$

where $\Sigma$ is the sample covariance of $(X_1, ..., X_p)$. We remark that there is an unidentifiability associated with $A$ as $AO$ satisfies the same model for any orthonormal matrix. In other words, one has to restrict $A$ to obtain a unique solution. Obtaining an estimator for $A$ is often carried out using the singular value decomposition or the maximum likelihood method.

Comparatively, independent component analysis uses the same latent models structure; however, it requires that $S_1, ..., S_q$ be independent but not necessarily Gaussian. Such a restriction imposes more stringent higher moment conditions than uncorrelated relationship in factor analysis. Thus, it makes the estimation of $A$ unique and allows the non-Gaussian distribution of $S_1, ..., S_q$. The solution to the independent component analysis is obtained by minimize some entropy or we can start from factor analysis then look for some rotation that leads to independent components.

### 9.3 Multidimensional scaling

Both principal component analysis and latent component analysis map the original data points to some low-dimensional manifold, where such a low-dimensional manifold can be explicitly expressed in terms of principal components or latent components. Multidimensional scaling has a similar goal but the obtained low-dimensional manifold may not be so explicit due to its different motivation.

The multidimensional scaling method only uses the dissimilarity between any two observations, which is defined as some distance between these two observations. Let $d_{ij}$ denote the dissimilarity between data $X_i$ and $X_j$. Then the multidimensional scaling seeks the corresponding values $Z_1, ..., Z_n$ for all the subjects in a low dimensional space $R^q$ so that the dissimilarity among subjects is retained as maximally as possible; that is,

$$\left[ \sum_{i \neq j} (d_{ij} - \|Z_i - Z_j\|)^2 \right]^{1/2}$$

is minimized. This is also known as least squares or Kruskal-Shephard scaling. A gradient descent algorithm is used to find the minimum.
Some variation of the criterion can be used, including Sammon mapping which minimizes
\[
\sum_{i \neq j} \left( d_{ij} - \frac{\|Z_i - Z_j\|}{d_{ij}} \right)^2.
\]
The latter emphasizes more on preserving smaller pairwise distances. Another way, called Shephard-Kruskal nonmetric scaling only relies on the ranks of the dissimilarities by minimizing
\[
\sum_{i \neq j} (d_{ij} - g(\|Z_i - Z_j\|))^2,
\]
where \(g\) is an increasing function also in the minimization.

Because multidimensional scaling only gives the projections of the original data on a low-dimensional manifold so does not give a parameterization of the manifold, it only reveals the intrinsic structures in the existing data so may not be convenient to be applied to new data. In this sense, multidimensional scaling is more useful for visualizing data in some low-dimensional manifolds.

### 9.4 Cluster analysis

Different from the previous unsupervised learning methods, cluster analysis, also called data segmentation, does not aim for a low-dimensional representation of data; instead, it seeks some collections of subjects (clusters) such that subjects within clusters are more similar than between clusters in terms of feature values. Because of this, the central quantity in the cluster analysis is similar to multidimensional scaling, that is, the degree of similarity (dissimilarity) between subjects. The real quantity used in the cluster analysis is the so-called proximity matrix, which is a \(n\) by \(n\) matrix with \((i, j)\) element being the similarity (or dissimilarity) between subject \(i\) and subject \(j\).

Since both multidimensional scaling and cluster analysis use dissimilarity, we may discuss a bit more on how to define such a measure. For quantitative features, it may be simply defined as \(l(|X_i - X_j|)\), where \(l(\cdot)\) is a non-negative loss function, for instance, the Euclidean distance. For the feature with ordinal values, one way is to assign scores to each ordinal value then treat the assigned scores as quantitative feature. The most common distance for the categorical feature is the Hamming distance, which is calculated as the number of mismatched categories between any two subjects. Therefore, when the feature from each individual consists some or all these types of values, a weighted summation of the distances from each coordinate can be
used to define the distance between these two observations. The choice of the weights is a subject matter.

With dissimilarity matrix, the first algorithm for cluster analysis is called combinatorial algorithm. We suppose that the whole data consist of $K$ clusters and we label them as $1, 2, \ldots, K$. Then the goal of cluster analysis is to identify a map $C$ which maps each subject id to one of these $K$ labels. Since cluster analysis wants to have subjects within the same cluster more similar to the subjects between clusters, a natural way is to define the within-cluster loss as

$$
\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{K} I(C(i) = C(j) = k)d(X_i, X_j)
$$

while define the between-cluster loss as

$$
\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{K} I(C(i) = k, C(j) \neq k)d(X_i, X_j).
$$

Hence, we want to either minimize the within-cluster loss or maximize the between-cluster loss. These two optimizations are equivalent since the summation of these two losses is a constant. Unfortunately, such an optimization is almost infeasible due to larger number of maps to be calculated.

Some strategies based on iterative greedy decent are feasible, although they may end up with suboptimal maps. Among them, one of the most popular algorithms is called the $K$-means algorithm, which applies to the situation when all the feature values are quantitative and the distance is the squared Euclidean distance. Under this case, this algorithm follows from the observation that the within-cluster loss is equal to

$$
\sum_{i=1}^{n} \sum_{k=1}^{K} I(C(i) = k)\|X_i - m_k\|^2,
$$

where $m_k$ is the mean of the $k$th cluster. Thus, the $K$-means algorithm can be described as follows: given $C$, we find $m_1, \ldots, m_K$ to minimize the above function; next, given $m_1, \ldots, m_K$, for each subject $i$, we determine $C(i)$ as

$$
\arg\min_{k=1, \ldots, K} \|X_i - m_k\|^2;
$$

we iterate till no change of cluster assignment. Clearly, the $K$-means algorithm is easy to be implemented. However, it may go to some local minimum so it is often suggested to start from many different random choices of $m_1, \ldots, m_K$ then choose the solution having the smallest value of the within-cluster loss. As a final note, the $K$-means algorithm is closely related to the EM
algorithm in estimating a Gaussian mixture model, where in each iteration, the M-step updates the means of the latent normal components and the E-step imputes the membership of each observation.

For general feature values and general proximity matrix, the \( K \)-means algorithm is not applicable. To handle this issue, one develops the \( K \)-medoids algorithm. This algorithm is very similar to the \( K \)-means algorithm, except that in the first step, instead of identifying the mean, we identify cluster medoids as the observation in the cluster which minimizes the total distances to all other points in the same cluster:

\[
\arg\min_{i \in C^{-1}(k)} \sum_{C(j)=k} d(X_j, X_i);
\]

the second step is the same but we replace the Euclidean distance by \( d(X_j, X_i) \).

Both \( K \)-means and \( K \)-medoids require a pre-specified number of the clusters. There is another clustering algorithm called hierarchical clustering, which does not specify the number of the clusters but lets data automatically form clusters. Eventually, users can decide how many clusters are appropriate. Strategies for hierarchical clustering divide into two basic approaches: agglomerative (bottom-up) and divisive (top-down). Agglomerative approaches starts at the bottom, where each subject is treated as a single cluster, and recursively merge a selected pair of clusters into a single cluster. The pair chosen for merging consist of the two groups with the smallest intergroup dissimilarity. Eventually, all the clusters will be merged into one largest cluster containing all the subjects. Instead, divisive approach starts from a single cluster consisting of all the subjects and recursively split of one the existing clusters into two clusters, where the split is chosen to produce two new groups with the largest between-group dissimilarity. Eventually, the last level at the bottom contains \( n \) clusters where each cluster contains one single subject. Thus, in both methods, there are a total \((n - 1)\) levels in the hierarchy.

Recursive binary splitting/agglomeration can be represented by a rooted binary tree, where the nodes of the trees at \( k \)the level represents the \( k \)th level clusters. Along the tree, the dissimilarity between merged clusters is monotone increasing so that we can used the height of each node to be proportional to the value of the intergroup dissimilarity between its two descendent clusters. This tree graph is called a dendrogram.

In hierarchy clustering, it is necessary to define the dissimilarity between any two clusters.
There are different ways for this definition. One definition is called the single linkage is to define
\[
d(C_1, C_2) = \min_{i \in C_1, j \in C_2} d(X_i, X_j).
\]
A second definition is called complete linkage by defining
\[
d(C_1, C_2) = \max_{i \in C_1, j \in C_2} d(X_i, X_j).
\]
Additionally, a third definition of group average is
\[
d(C_1, C_2) = \frac{1}{n_1 n_2} \sum_{i \in C_1, j \in C_2} d(X_i, X_j).
\]
One general observation is that if the data dissimilarity indicate a strong clustering tendency, with each of the clusters being compact and well separated from others, then all these definitions for group dissimilarity produce similar results. However, because of the nature in these definitions, the single linkage can produce clusters with very large diameter (the maximal distance within the cluster) and the complete linkage is oppositive; while the group average is a compromise between the two extremes.

Finally, another unsupervised learning method is called self-organizing maps. This method can be viewed a constrained version of $K$-means clustering, where the prototypes are encouraged to lie in a one- or two-dimensional manifold in the feature space. The resulting manifold is called a constrained topological map. The detail of the algorithm can be found in Section 14.4 of HFT book.
Part II

LEARNING THEORY
Chapter 10

BAYES ERROR

This part studies theoretical foundations behind supervised learning. We only focus on classification problem with a binary outcome. The central role in learning theory is the Bayesian error discussed in Chapter 2. Thus, this part will include the estimation of Bayesian error and the consistency and convergence rates of a variety of classifiers in term of Bayesian errors. In most of situations, proving consistency and convergence rates require the use of concentration inequalities we have seen before, so the last section in this chapter gives a comprehensive review of these types of inequalities.

10.1 Bayes error

As before, we use \((Y_1, X_1), \ldots, (Y_n, X_n)\) to denote the class label \((Y_i = 0, 1)\) and the feature values for independent subjects \(i = 1, \ldots, n\). As shown in Chapter 2, the most desirable classifier is

\[
g^*(x) = I(\eta(x) > 1/2), \quad \text{where} \quad \eta(x) = E[Y|X = x] = P(Y = 1|X = x).
\]

The theoretical error resulting from such a classifier is called the Bayes error and is given by

\[
L(g^*) \equiv P(g^*(X) \neq Y) = E[\min(\eta(X), 1 - \eta(X))] = \frac{1}{2} - \frac{1}{2} E[|1 - 2\eta(X)|].
\]

Equivalently, if \(X\) given \(Y = y\) has a density \(f_y(x)\) with respect to some dominating measure \(\mu\) and \(P(Y = y) = p_y\), then the Bayes error is equivalent to

\[
\int \min(p_0 f_0(x), p_1 f_1(x)) d\mu(x).
\]

Under a special case when \(f_0\) and \(f_1\) are nonoverlapping and \(p_0 = 1/2\), this error is

\[
\frac{1}{2} - \frac{1}{4} \int |f_1(x) - f_0(x)| d\mu(x).
\]
Recall that in Chapter 2, we have shown that $L(g^*) \leq L(g)$ for any other classifier $g$. In other words, the Bayes error is the best prediction error we can achieve in theory.

In statistical learning, other than the Bayes error, there are many other types of error definitions (distance measures) to characterize the performance of classifiers. One of such definitions is called the Komogorov variational distance, which is the total variation distance between distributions:

$$\frac{1}{2} E \{ |P(Y = 1|X) - P(Y = 0|X)| \}.$$ 

Therefore, the larger the distance is, the better classification will be. It turns out that the the negative value of this distance is equivalent to the Bayes error since the above expression is equal to $E[|2\eta(X) - 1|]/2$.

One error based on $\eta(X)$ is called the nearest neighborhood error defined as

$$L_{NN} = E[2\eta(X)(1 - \eta(X))],$$

which corresponds to the asymptotic error of the nearest neighbor rule. Since $2\eta(X)(1-\eta(X)) \geq \min(\eta(X), 1 - \eta(X))$, $L_{NN}^* \geq L(g^*)$ where $L_{NN}^*$ is the smallest nearest neighborhood error. To obtain an upper bound for $L_{NN}^*$, we need the following association inequalities.

**Lemma 10.1.** (Association inequalities) For any random variable $X$ and let $Q_1$ and $Q_2$ be monotone nondecreasing functions. Then

$$E[Q_1(X)Q_2(X)] \geq E[Q_1(X)]E[Q_2(X)]$$

provided all expectations exit and are finite. If $Q_1$ is monotone increasing and $Q_2$ is monotone decreasing, then

$$E[Q_1(X)Q_2(X)] \leq E[Q_1(X)]E[Q_2(X)].$$

**Proof.** Let $Y$ be another random variable independent of $X$ and have the same distribution as $X$. To prove the first half, since

$$\{Q_1(X) - Q_1(Y)\}\{Q_2(X) - Q_2(Y)\} \geq 0.$$ 

We take the expectation and obtain the result. The second half is based on that the above inequality is in the opposite direction.

Therefore, using Lemma 10.1, we have

$$L_{NN} = 2E[\min(\eta(X), 1 - \eta(X))\{1 - \min(\eta(X), 1 - \eta(X))\}]$$
10.1. BAYES ERROR

\[
\leq 2E[\min(\eta(X), 1 - \eta(X))]E[\{1 - \min(\eta(X), 1 - \eta(x))\}] = 2L(g)(1 - L(g)) \leq 2L(g).
\]

So \(L_{NN}^* \leq 2L(g^*)\). We then conclude that the nearest neighborhood error has some information about the Bayes error.

The Bhattacharyya measure of affinity is defined as

\[
-\log E \left\{ \sqrt{\eta(X)(1 - \eta(X))} \right\},
\]

where the expectation part, denote by \(\rho\), is referred as the Matushita error. This \(\rho\) takes values between 0 and 1/2, corresponding to the zero Bayes error and the random classification respectively. Thus, the smaller the measure is, the better classification it will has. The relationship between this distance and the Bayes error is nonlinear. First,

\[
\left( E \left\{ \sqrt{\eta(X)(1 - \eta(X))} \right\} \right)^2 \leq E[\eta(X)(1 - \eta(X))] = L_{NN}/2 \leq L(g)(1 - L(g)).
\]

On the other hand,

\[
\sqrt{\eta(X)(1 - \eta(X))} \geq 2\eta(X)(1 - \eta(X)).
\]

We conclude

\[
\frac{1}{2} - \frac{1}{2}\sqrt{1 - 4\rho^*^2} \leq L(g^*) \leq \rho^*,
\]

where \(\rho^*\) is the smallest value of \(\rho\). This inequality provides information about \(\rho\) related to the Bayes error. A generalization of Bhattacharyya’s measure of affinity is

\[
-\log \left\{ E[\eta(X)^\alpha(1 - \eta(X))^{1-\alpha}] \right\},
\]

where \(\alpha\) is a constant in (0, 1). But this generalization has no practical interpretation if \(\alpha\) is different from 1/2.

The Shannon entropy is the key quantity in information theory, which can be used to characterize how different a distribution is from complete randomness. Thus, a definition of classification error can be defined as the expectation of the conditional entropy of \(Y\) given \(X\):

\[
\mathcal{E} \equiv -E [\eta(X) \log \eta(X) + (1 - \eta(X)) \log(1 - \eta(X))].
\]

The relationship between \(\mathcal{E}\) and \(L(g)\) is given as follows. First,

\[
\mathcal{E} = -E [A(X) \log A(X) + (1 - A(X)) \log(1 - A(X))],
\]

where \(A(X) = \min(\eta(X), 1 - \eta(X))\). By the Jensen’s inequality using the convexity of \(-x \log x\),

\[
\mathcal{E} \leq -E[A(X)] \log E[A(X)] - (1 - E[A(X)]) \log(1 - E[A(X)])
\]
\(-L(g) \log L(g) - (1 - L(g)) \log (1 - L(g)).\)

On the other hand, by the Taylor expansion,

\[-A(X) \log A(X) - (1 - A(X)) \log (1 - A(X)) \leq \log 2 - \frac{1}{2} (2A(X) - 1)^2,\]

we have

\[\mathcal{E} \leq \log 2 - \frac{1}{2} E[1 - 2L_{NN}] \leq \log 2 - \frac{1}{2} + 2L(g)(1 - L(g)) = \log 2 - \frac{1}{2} (1 - L(g))^2.\]

We then have

\[\mathcal{E}^* \leq \min \left\{ -L(g^*) \log L(g^*) - (1 - L(g^*)) \log (1 - L(g^*)), \log 2 - \frac{1}{2} (1 - L(g^*))^2 \right\}.\]

Additionally, we note

\[\mathcal{E} \geq -E[\log(A(X)^2 + (1 - A(X))^2)] \geq -\log E[1 - 2A(X)(1 - A(X))] = -\log (1 - L_{NN}) \geq -\log (1 - L(g)).\]

Thus,

\[\mathcal{E}^* \geq -\log (1 - L(g^*)).\]

An additional way of defining distance is based on the Kullback-Leibler divergence and it is defined as

\[E \left[ (2\eta(X) - 1) \log \frac{\eta(X)}{1 - \eta(X)} \right].\]

This divergence can be infinite so it is impossible to bound it from above by \(L(g^*)\). While, it can be bounded from below in terms of \(L(g^*)\). To see this, we note

\[(2\eta(X) - 1) \log \frac{\eta(X)}{1 - \eta(X)} = [2\eta(X) - 1] \log \frac{1 + |2\eta(X) - 1|}{1 - |2\eta(X) - 1|}.\]

Using the convexity of \(x \log(1 + x)/(1 - x)\) and the Jensen’s inequality, we obtain

\[E \left[ (2\eta(X) - 1) \log \frac{\eta(X)}{1 - \eta(X)} \right] \geq (1 - 2L(g)) \log \frac{1 + (1 - 2L(g))}{1 - (1 - 2L(g))} = (1 - 2L(g)) \log \frac{1 - L(g)}{L(g)} \geq 2(1 - L(g))^2.\]

Finally, we summarize all the above errors in form of \(E[F(\eta(X))]\), where \(F\) is a concave function. Different classification errors correspond to different choices of \(F\). Clearly, the closer
10.2 Consistency based on Bayes error

Let $g_n$ be a classifier based on training data $D_n((X_1, Y_1), \ldots, (X_n, Y_n))$. Then the error resulting from this classifier is given by

$$L_n = L(g_n) = P(g_n(X) \neq Y|D_n).$$

When we say $g_n$ is a consistency classifier, we expect that this error gets arbitrarily close to the Bayes error $L(g^*)$. Formally, the definition is as follows.

**Consistency definition** A classification rule is consistent (or asymptotically Bayes-risk efficient) for a certain distribution of $(X, Y)$ if

$$E[L_n] = P(g_n(X) \neq Y) \to L(g^*)$$

and strongly consistent if $L_n$ converges to $L(g^*)$ almost surely. If the above converges holds for any distribution of $(X, Y)$, we call the rule universally (strongly) consistent.

The consistency of a classification rule is related to certain consistency of the estimator, $\eta_n(x)$. Let $g_n$ be the classification rule corresponding to $\eta_n(x)$, i.e., $g_n(x) = I(\eta_n(x) > 1/2)$. Then we note

$$L_n(g_n) - L(g^*) = E[\{I(g_n(X) \neq Y) - I(g^*(X) \neq Y)\}|D_n]$$

$$= E[\eta(X)\{I(g_n(X) = 0) - I(g^*(X) = 0)\} + (1 - \eta(X))\{I(g_n(X) = 1) - I(g^*(X) = 1)\}]$$

$$= E[(1 - 2\eta(X))\{I(g_n(X) = 1) - I(g^*(X) = 1)\}|D_n] = E[2\eta(X) - 1|I(g_n(X) \neq g^*(X))|D_n].$$

Hence, since if $g_n(x) \neq g^*(x)$, then $\eta_n(x) > 1/2 \geq \eta(x)$ or $\eta_n(x) \leq 1/2 < \eta(x)$, we obtain

$$L_n(g_n) - L(g^*) \leq 2E[|\eta(X) - \eta_n(X)||D_n] \leq 2E[(\eta(X) - \eta_n(x))^2|D_n]^{1/2}.$$

In other words, the consistency in terms of the Bayes error is implied if we have the $L_1$ or $L_2$-consistency of $\eta_n$. Using this result, we can obtain the consistency of many classification rules in direct learning because $\eta_n$ usually has some explicit expression. For convenience, from now on, we use $E_n[-]$ to denote the conditional expectation of $E[-|D_n]$. 
Chapter 11

CONSISTENCY OF DIRECT LEARNING METHODS

In direct learning method, classification rules are obtained by plugging one estimator for \( \eta(X) \) or some other functions used in the decision. Therefore, according to the previous result, the consistency for these method is usually obtained by examining the \( L_1 \)- or \( L_2 \)-consistency of the plug-in estimators.

11.1 Histogram estimator

A histogram estimator for estimating \( \eta(x) \) takes form

\[
\eta_n(x) = \sum_{k=1}^{M} I(x \in A_{nk}) \hat{Y}_k,
\]

where \( A_{n1}, ..., A_{nM} \) are a partition of the feature space into cubes with the same size, and \( \hat{Y}_k \) is the majority of the observed classes in the cube \( A_{nk} \), i.e.,

\[
\hat{Y}_k = \frac{\sum_{i=1}^{n} Y_i I(X_i \in A_{nk})}{\sum_{i=1}^{n} I(X_i \in A_{nk})}.
\]

The histogram estimator can also be written as

\[
\frac{1}{N(x)} \sum_{i=1}^{n} Y_i I(X_i \in A(x)),
\]

where \( N(x) \) is the denominator in defining \( \hat{Y}_k \), and \( A(x) \) denotes the cube which \( x \) belongs to.

To prove the consistency of the histogram estimator, we let \( \bar{\eta}(x) = E[\eta(X)|X \in A(x)] \). Then

\[
E_n[|\eta_n(x) - \eta(X)|] \leq E_n[|\eta_n(x) - \bar{\eta}(X)|] + E[|\bar{\eta}(X) - \eta(X)|].
\]
Clearly,

\[ E_n[|\eta_n(X) - \bar{\eta}(X)||X] \leq E_n[I(N(X) > 0)|\sum_{i=1}^n Y_i I(X_i \in A(X)) \over N(X)} - \bar{\eta}(X)|X] + I(N(X) = 0). \]

Since conditional on \( X \), \( \sum_{i=1}^n Y_i I(X_i \in A(X)) \) follows Binomial distribution \( (N(X), p(X)) \) where \( p(X) = \bar{\eta}(X) \), the first term in the right-hand side of above inequality is bounded by

\[ I(N(X) > 0)\sqrt{\bar{\eta}(X)(1 - \bar{\eta}(X))}/N(X) \leq I(N(X) > 0)/(2\sqrt{N(X)}). \]

We then obtain

\[ E_n[|\eta_n(X) - \bar{\eta}(X)|] \leq {1 \over 2} P(N(X) \leq k|D_n) + {1 \over 2\sqrt{k}} + P(N(X) = 0|D_n), \]

where \( k \) is any positive constant. Therefore, if we can show \( P(N(X) \leq k|D_n) \to 0 \) for any fixed \( k \), then by first choosing \( k \) large than \( n \) tending to infinity, it gives

\[ E_n[|\eta_n(X) - \bar{\eta}(X)|] \to 0. \]

which gives the universal consistency of the histogram rule.

On the other hand, for any \( \epsilon > 0 \), we find a continuous function with compact support, \( \eta^\epsilon(x) \), such that

\[ E[|\eta^\epsilon(X) - \eta(X)|] < \epsilon. \]

Then

\[ E[|\bar{\eta}(X) - \eta(X)|] \leq E[|\bar{\eta}(X) - \bar{\eta}^\epsilon(X)|] + E[|\bar{\eta}^\epsilon(X) - \eta^\epsilon(X)|] + \epsilon, \]

where \( \bar{\eta}^\epsilon(x) = E[\eta^\epsilon(X)|X \in A(x)] \). Since

\[ E[|\bar{\eta}(X) - \bar{\eta}^\epsilon(X)|] \leq E[|\eta(X) - \eta^\epsilon(X)|] \leq \epsilon, \]

we have

\[ E[|\bar{\eta}(X) - \eta(X)|] \leq E[|\bar{\eta}^\epsilon(X) - \eta^\epsilon(X)|] + 2\epsilon. \]

Additionally, since \( \eta^\epsilon \) is uniformly continuous, the first term in the right-hand vanishes as \( n \) goes to \( \infty \). Thus, we first let \( n \) go to \( \infty \) then \( \epsilon \) go to 0, we obtain

\[ E[|\bar{\eta}(X) - \eta(X)|] \to 0. \]

Combining these results, we have proved

\[ E_n[|\eta_n(X) - \eta(X)|] \to 0, \]
11.1. HISTOGRAM ESTIMATOR

which gives the universal consistency of the histogram rule.

It remains to give some sufficient conditions so that \( P(N(X) \leq k|D_n) \to 0 \) as \( n \) goes to infinity. One sufficient condition is given by \( h_n \to 0 \) and \( nh_n^d \to \infty \), where \( h_n \) is the edge length of each cube in the partition. The detail can be found in Section 6.4 of DGL book.

We have only shown the weak universal consistency of the histogram rule. Actually, we can further obtain the strong universal consistency.

**Strong universal consistency of histogram rule** Suppose \( h_n \to 0 \) and \( nh_n^d \to \infty \). Then for any distribution of \((X, Y)\) and for any \( \epsilon > 0 \), there is an integer \( n_0 \) such that for \( n > n_0 \),

\[
P(L_n - L(g^*) > \epsilon) \leq 2e^{-n\epsilon^2/32}.
\]

Following the Borel-Cantelli lemma, we immediately obtain \( L_n - L(g^*) \to 0 \) almost surely hence the strong universal consistency.

**Proof.** First, the histogram rule is equivalent to the rule based on

\[
\eta_n^*(x) = \sum_{i=1}^n Y_i I(X_i \in A(x)) / nP(X \in A(x)).
\]

Using the relationship between \( L_n - L(g^*) \) and the \( L_1 \)-distance of \( \eta_n^* - \eta \), it is sufficient to show

\[
P(E_n[|\eta_n^*(X) - \eta(X)|] > \epsilon/2) \leq e^{-n\epsilon^2/32}.
\]

We write

\[
E_n[|\eta_n^*(X) - \eta(X)|] = E_n[|\eta_n^*(X) - \eta(X)|] - E[|\eta_n^*(X) - \eta(X)|] + E[|\eta_n^*(X) - \eta(X)|],
\]

where \( E[\cdot] \) denotes the full expectation. According to the weak universal consistency of the the histogram rule, the last term is smaller than \( \epsilon/4 \) when \( n \) is large (details can also be found in Section 9.2 of DGL book). Thus, it suffices to show the exponential bound holds for the tail probability of

\[
E_n[|\eta_n^*(X) - \eta(X)|] - E[|\eta_n^*(X) - \eta(X)|].
\]

For this purpose, we use the bounded difference concentration inequality given in the last section of this chapter. Particularly, if we replace one of the training data \((X_i, Y_i)\) by \((\tilde{X}_i, \tilde{Y}_i)\), the difference in changing \( E_n[|\eta_n^*(X) - \eta(X)|] \) only happens when \( X \in A_n(X_i) \) and \( A_n(\tilde{X}_i) \) so this change is bounded by

\[
\frac{1}{nP(X \in A_n(X_i))}P(X \in A_n(X_i)) + \frac{1}{nP(X \in A_n(\tilde{X}_i))}P(X \in A_n(\tilde{X}_i)) = \frac{2}{n}.
\]
Thus, following the bounded difference inequality, we obtain
\[ P(E_n[|\eta^*_n(X) - \eta(X)| - E[|\eta^*_n(X) - \eta(X)|] > \epsilon/4]) \leq e^{-n\epsilon^2/32}. \]

The proof is completed.

### 11.2 $k$-nearest neighborhood estimator

In a $k$-nearest neighborhood rule, the predicted class for feature value $x$ is given by

\[ g_n(x) = I(\sum_{i=1}^{n} \omega_{ni} I(Y_i = 1 > \sum_{i=1}^{n} \omega_{ni} I(Y_i = 0)), \]

where $\omega_{ni} = 1/k$ if $X_i$ is among the $k$ nearest neighbors of $x$ and 0 otherwise. In this section, we study the consistency of the $k$-nearest neighborhood rule for either fixed $k$ or $k$ increasing with $n$.

Fix $x$ and we order $(X_1, Y_1), ..., (X_n, Y_n)$ according to ascending values of $\|X_i - x\|$. The obtained sequence is denoted by

\[ (X_{(1)}(x), Y_{(1)}(x)), ..., (X_{(n)}(x), Y_{(n)}(x)). \]

The first fact is that for any $x$ in the support of $X_1$, $\|X_{(k)}(x) - x\| \to 0$ with probability one when $k/n \to 0$. This is because

\[ \|X_{(k)}(x) - x\| > \epsilon \text{ is equivalent to } \sum_{i=1}^{n} I(\|X_i - x\| \leq \epsilon) < k. \]

Furthermore, for any i.i.d variable $X$, by the dominated convergence theorem,

\[ P(\|X_{(k)}(X) - X\| > \epsilon) \to 0. \]

Since $\|X_{(k)}(X) - X\|$ is decreasing for fixed $k$, it must decrease to zero with probability one. If $k$ grows with $n$, we consider the sequence of monotone decreasing sequence

\[ \sup_{m \geq n} \|X_{(k)}(X) - X\| \]

and obtain the same conclusion. In summary, we have $\|X_{(k)}(x) - x\| \to 0$ and $\|X_{(k)}(X) - X\| \to 0$ with probability one.
11.2. K-NEAREST NEIGHBORHOOD ESTIMATOR

11.2.1 Asymptotic error for fixed $k$

We first establish the expression of the asymptotic error of the nearest neighborhood rule for fixed $k$. The key idea is to obtain the equivalence between the nearest neighborhood rule and another classification rule based on i.i.d Bernoulli random variables. Specifically, we define another classification rule based on i.i.d observations $U_1, ..., U_n$ from the uniform distribution in $[0, 1]$. This rule is given by

$$\tilde{g}_n(x) = I(\sum_{i=1}^k \tilde{Y}_i(x) > k/2),$$

where $\tilde{Y}_i(x) = I(U_i \leq \eta(x))$. Note that the original rule is

$$g_n(x) = I(\sum_{i=1}^k Y_i(x) > k/2).$$

Therefore, we conclude that the difference between these two rules is

$$P(\tilde{g}_n(x) \neq g_n(x)) \leq P(\sum_{i=1}^k \tilde{Y}_i(x) \neq \sum_{i=1}^k Y_i(x)) \leq \sum_{i=1}^k P(\tilde{Y}_i(x) \neq Y_i(x))$$

$$\leq \sum_{i=1}^k P(\eta(X_i(x)) \leq U(i)) \leq \eta(x) + \sum_{i=1}^k P(\eta(x) \leq U(i)) \leq \eta(X_i(x)))$$

$$\leq \sum_{i=1}^k E[|\eta(x) - \eta(X_i(x))|].$$

Hence,

$$E[|L(g_n) - L(\tilde{g}_n)|] \leq P(\tilde{g}_n(X) \neq \tilde{g}_n'(X)) \leq \sum_{i=1}^k E[|\eta(X) - \eta(X_i(X))|].$$

We now show that the right-hand side converges to zero. Clearly, if $\eta$ is the uniformly continuous and has a bound support, for any $\epsilon$, this term is bounded by

$$k\|\eta\|_{\infty}P(\|X - X_{(k)}(X)\| > \delta) + k\epsilon,$$

where $\delta$ depends on $\epsilon$. Then it converges to zero based on the fact $\|X - X_{(k)}(X)\|$ converges to zero. For general $\eta$, we find a uniform continuous function with a bounded support, denoted by $\tilde{\eta}$ such that

$$\|\eta(X) - \tilde{\eta}(X)\| < \epsilon.$$

Then the right-hand side is bounded by

$$\sum_{i=1}^k E[|\eta(X) - \tilde{\eta}(X)|] + \sum_{i=1}^k E[|\eta(X_{(i)}(X)) - \tilde{\eta}(X_{(i)}(X))|] + \sum_{i=1}^k E[|\tilde{\eta}(X) - \tilde{\eta}(X_{(i)}(X))|].$$
The first term is no larger than \( k\epsilon \); the second term, following Stone’s lemma (the proof is given in Section 5.3 of DGL book), is bounded by \( k[(1 + 2/\sqrt{2} - \sqrt{3})^p - 1] \sum_{i=1}^k E[|\eta(X) - \tilde{\eta}(X)|] \) so is also small; the third term vanishes with \( n \) because \( \tilde{\eta} \) is uniformly continuous and has a bounded support. In conclusion, we have shown

\[
E[|L(g_n) - L(\tilde{g}_n)|] \to 0.
\]

It remains to find the limit of \( E[L(\tilde{g}_n)] \). One interesting observation in the new rule \( \tilde{g}_n \) is that \( \tilde{Y}_{(1)}, \ldots, \tilde{Y}_{(i)} \) are i.i.d from a Bernoulli distribution with probability \( \eta(x) \). Thus, it is easy to calculate that when \( k \) is odd,

\[
E[L(\tilde{g}_n)] = E \left\{ \sum_{j=0}^{k} \binom{k}{j} \eta(X)^j (1 - \eta(X))^{k-j} (\eta(X)I(j < k/2) + (1 - \eta(X))I(j > k/2)) \right\},
\]

which is denoted as \( L_{kNN} \). Hence, we obtain the following result.

**Universal consistency of \( k \)-nearest neighborhood rule (fixed \( k \))**

Let \( k \) be odd and fixed. Then,

\[
\lim_{n} E[L(g_n)] = L_{kNN}.
\]

Additionally, for even \( k \), \( L_{kNN} = L_{(k-1)NN} \).

The last part can be seen in Section 5.6 of DGL book. From this result, we notice that the error from the \( k \)-nearest neighborhood rule is necessary the same as the Bayes rule. Actually, for \( k = 1 \), Section 10.1 has shown that \( L(g^*) \leq L_{NN} \leq 2L(g^*) \). For general \( k \), there exists some similar results to examine how far \( L_{kNN} \) from \( L(g^*) \). Particularly, it is shown that for odd \( k \),

\[
L_{kNN} \leq L(g^*) + \sqrt{\frac{2L_{NN}}{k}}, \quad L_{kNN} \leq L(g^*)(1 + \frac{\gamma}{\sqrt{k}}(1 + O(k^{-1/6}))) \text{if } k \geq 3,
\]

where \( \gamma = \sup_{r>0} 2r(1 - \Phi(r)) = 0.3399434... \) The proofs can be found in Section 5.7 of DGL book.

**11.2.2 Consistency of the nearest neighborhood rule with increasing \( k \)**

The previous results show that the nearest neighborhood rule is not consistent for fixed \( k \). However, in this part, we show that if we allow \( k \) increasing but slower than \( n \), the \( k \)-nearest neighborhood is then strongly universal consistent.
11.2. **K-NEAREST NEIGHBORHOOD ESTIMATOR**

**Strongly universal consistency of k-nearest neighborhood rule** Assume $X$ is absolutely continuous with respect to Lebesgue measure. If $k \to \infty$ and $k/n \to 0$, then for any $\epsilon > 0$, there is an $n_0$ such that for $n > n_0$,

$$P(L_n - L(g^*) > \epsilon) \leq 2e^{-n\epsilon^2/(72\gamma_p^2)},$$

where $\gamma_p$ is the minimal number of cones centered at the origin of angle $\pi/6$ that cover $R^p$.

**Proof.** The estimator for $\eta(x)$ associated with the $k$-nearest neighborhood rule is

$$\eta_n(x) = \frac{1}{k} \sum_{i=1}^{n} Y_i(x).$$

Thus, from Section 10.2, it suffices to show

$$P(E_n[|\eta(X) - \eta_n(X)| > \epsilon/2] \leq 2e^{-n\epsilon^2/(72\gamma_p^2)}).$$

Let $\rho_n(x)$ be the radius of a ball centered at $x$ such that the probability of $X$ belonging to this ball is equal to $k/n$. This value exists due to the absolute continuity of $X$-induce measure. We define

$$\tilde{\eta}_n(x) = \frac{1}{k} \sum_{j=1}^{n} Y_j I(\|X_j - x\| < \rho_n(x)).$$

Then

$$|\eta(x) - \eta_n(x)| \leq |\eta(x) - \tilde{\eta}_n(x)| + |\tilde{\eta}_n(x) - \eta_n(x)|.$$

Note

$$|\tilde{\eta}_n(x) - \eta_n(x)| = \left| \frac{1}{k} \sum_{i=1}^{n} Y_i I(X_i \in \rho_n(x)) - \frac{1}{k} \sum_{i=1}^{n} Y_i I(X_i \in r_n(x)) \right|$$

$$\leq \frac{1}{k} \sum_{i=1}^{n} |I(X_i \in \rho_n(x)) - I(X_i \in r_n(x))| = \frac{1}{k} \sum_{i=1}^{n} |I(X_i \in \rho_n(x)) - 1|,$$

where $r_n(x) = \|X_{(k)}(x) - x\|$. It gives

$$E_n[|\tilde{\eta}_n(X) - \eta_n(X)|] = \int E[|\tilde{\eta}_n(x) - \eta_n(x)|]P(dx) \leq \int E[|\tilde{\eta}_n(x) - \eta_n(x)|^2]P(dx)$$

$$\leq \int \sqrt{\frac{1}{k^2} Var(I(\|X - x\| < \rho_n(x))}P(dx) \leq \frac{1}{\sqrt{k}},$$

which converges to zero. Second, following Theorem 6.3 and 6.4 in DGL book,

$$E[\int |\eta(x) - \eta_n(x)|P(dx)] \to 0.$$
Therefore, when $n$ is large enough,

$$P(E_n[|\eta(X) - \eta_n(X)|] > \epsilon/2) \leq P(E_n[|\eta(X) - \tilde{\eta}_n(X)|] > \epsilon/4) + P(E_n[|\tilde{\eta}_n(X) - \eta_n(X)|] > \epsilon/4)$$

$$\leq P(E_n[|\eta(X) - \tilde{\eta}_n(X)|] - E[|\eta(X) - \tilde{\eta}_n(X)|] > \epsilon/6)$$

$$+ P(E_n[|\tilde{\eta}_n(X) - \eta_n(X)|] - E[|\tilde{\eta}_n(X) - \eta_n(X)|] > \epsilon/4).$$

For either term in the right-hand side, we apply the bounded difference inequality. For example, consider

$$A(X_1, Y_1, ..., X_n, Y_n) = E_n[|\eta(X) - \tilde{\eta}_n(X)|] - E[|\eta(X) - \tilde{\eta}_n(X)|].$$

If we replace $(X_i, Y_i)$ by $(X'_i, Y'_i)$ and denote the resulted function as $A'$ with corresponding function as $\tilde{\eta}'_n$, then

$$|A - A'| \leq E_n[|\tilde{\eta}_n(X) - \tilde{\eta}'_n(X)|].$$

Note that the difference between $\tilde{\eta}_n$ and $\tilde{\eta}'_n$ is only when $\|X - X_i\| \leq \rho_n(X_i)$ or $\|X - X'_i\| \leq \rho_n(X'_i)$ and the difference is bounded by $2/k$. Therefore,

$$|A - A'| \leq \sup_i \frac{2}{k} \int_{x : \|x - X_i\| \leq \rho_n(x) \leq \rho_n(x)} P(dx).$$

The latter is bounded by $2\gamma_p/n$ following Lemma 11.1 of DGL book. We have justified the conditions for the bounded difference inequality. Hence, we conclude

$$P(E_n[|\eta(X) - \eta_n(X)|] > \epsilon/2) \leq 2n^{-n\epsilon^2/(72\gamma_p^2)}.$$

### 11.3 Kernel rules

The kernel rule for the classification is given as

$$g_n(x) = I(\sum_{i=1}^n Y_i K_h(X_i - x) > \sum_{i=1}^n (1 - Y_i) K(X_i - x)),$$

where $K_h(x) = h^{-p} K(x/h)$ for a kernel function $K(x)$ and a bandwith $h$. The corresponding $\eta_n(x)$ so that $g_n(x) = I(\eta_n(x) > 1/2)$ can be defined as

$$\eta_n(x) = \frac{\sum_{i=1}^n Y_i K_h(X_i - x)}{n E[K_h(X - x)]}.$$

Note that $\eta_n(x)$ has a different denominator from the usual kernel estimator. However, both result in the same rule. For the kernel rule, the following consistency result holds.
11.4 Classification rules based on sieve estimation

Strong universal consistency of kernel rules

Assume that $X$ has a density with respect to the Lebesgue measure, $K$ is a regular kernel and $h \to 0$ and $nh^d_n \to \infty$. Then for any distribution of $(X, Y)$ and $\epsilon > 0$,

$$P(L_n - L(g^*) > \epsilon) \leq 2e^{-n\epsilon^2/(32\rho^2)}$$

when $n$ is large enough. Here, $\rho$ is a constant only depending on $K$ and $p$.

A regular kernel function $K(x)$ satisfies that $K(x) \geq 0$ and there exists a $r > 0$ and $b > 0$ such that $K(x) \geq bI(\|x\| \leq r)$ and $\int \sup_{\|y-x\| \leq r} K(y)dx < \infty$.

The proof to the above result is similar to the one for the $k$-nearest neighborhood estimator. First, using the relationship between the errors and the estimators, we only need to show

$$P(\int |\eta_n(x) - \eta(x)|dP(dx) > \epsilon/2) \leq 2e^{-n\epsilon^2/(32\rho^2)}.$$

We decompose

$$\int |\eta_n(x) - \eta(x)|dP(dx) = \int |\eta_n(x) - \eta(x)|dP(dx) - E[\int |\eta_n(x) - \eta(x)|dP(dx)] + E[\int |\eta_n(x) - \eta(x)|dP(dx)].$$

Using the kernel estimator property, we can show that the last term vanishes with $n$ increasing. We then apply the bounded difference inequality to

$$A = \int |\eta_n(x) - \eta(x)|dP(dx) - E[\int |\eta_n(x) - \eta(x)|dP(dx)].$$

After replacing $(X_i, Y_i)$ by $(X'_i, Y'_i)$ which results in $A'$, we find that the difference between $A$ and $A'$ is bounded by

$$\int |\eta_n(x) - \eta'_n(x)|P(dx) \leq \sup_{y \in \mathbb{R}^d} \int \frac{2K_h(x-y)}{nE[K_h(X-x)]}P(dx) \leq \frac{2\rho}{n}.$$

The consistency result thus follows from the bounded difference inequality. The details can be founded in Chapter 10 of DGL book.

11.4 Classification rules based on sieve estimation

Such rule relies on estimating $\eta(x)$ via sieve estimation, such like the standard trigonometric bases, the Legendre polynomials, the Hermite functions, the Laguerre bases, the Harr bases
and etc. If we denote \( \psi_0, \psi_1, \ldots \) to be a complete orthonormal basis in \( \mathbb{R}^p \), then a classification rule based on sieve approximation is given as

\[
g_n(x) = I(\sum_{j=0}^{K_n} \alpha_{n,j} \psi_j(x) > 0),
\]

where \( \alpha_{n,j} = n^{-1} \sum_{i=1}^{n} (2Y_i - 1) \psi_j(X_i) \). Here, \( \alpha_n(x) = \sum_{j=0}^{K_n} \alpha_{n,j} \psi_j(x) \) is considered to be an estimator for

\[
\alpha(x) = (2\eta(x) - 1)f(x) = P(Y = 1, X = x) - P(Y = 0, X = 1).
\]

These classification rules are not necessary to be universally consistent but they are consistent for general classes of the distribution for \((X, Y)\).

**Strong consistency of classification rules based on sieve estimation** Suppose \( |\psi_i(x)| \leq B \) for a constant \( B \) and any \( i = 0, 1, 2, \ldots \). Moreover, \( X \) has conditional densities in \( L_2(\mathbb{R}^p) \) given \( Y = 0 \) or \( 1 \). If \( K_n \to \infty \) and \( K_n/n \to 0 \), the above rule is weakly consistent. If we further assume \( K_n \log n/n \to 0 \), then it is also strongly consistent.

**Proof.** We define \( \alpha_j = \int \alpha(x) \psi_j(x) dx \) then in \( L_2(\mathbb{R}^p) \) space, we have

\[
\alpha(x) = \sum_{j=0}^{\infty} \alpha_j \psi_j(x).
\]

Using the orthonormality of the basis functions, we obtain

\[
\int (\alpha(x) - \alpha_n(x))^2 dx = \sum_{j=1}^{K_n} (\alpha_n,j - \alpha_j)^2 + \sum_{j=K_n+1}^{\infty} \alpha_j^2.
\]

Since \( E[\alpha_{n,j}] = \alpha_j \),

\[
E[\int (\alpha(x) - \alpha_n(x))^2 dx] = \sum_{j=1}^{K_n} Var(\alpha_{n,j}) + \sum_{j=K_n+1}^{\infty} \alpha_j^2.
\]

On the other hand,

\[
Var(\alpha_{n,j}) = n^{-1} Var(\psi_j(X)(2\eta(X) - 1)) \leq n^{-1} B^2.
\]

Therefore, it is clear to see if \( K_n \to \infty \) and \( K_n/n \to 0 \),

\[
E[\int (\alpha(x) - \alpha_n(x))^2 dx] \to 0,
\]

which gives the weak consistency.
To prove the strong consistency, for any $\epsilon > 0$, when $n$ is large enough, we can have

$$\sum_{j=K_n+1}^{\infty} \alpha_j^2 < \epsilon/2.$$ 

Then

$$P(\int (\alpha(x) - \alpha_n(x))^2 dx > \epsilon) \leq P(\sum_{j=1}^{K_n} (\alpha_{n,j} - \alpha_j)^2 > \epsilon/2) \leq \sum_{j=1}^{K_n} P(|\alpha_{n,j} - \alpha_j| > \sqrt{\epsilon/(2K_n)}).$$

We apply the Hoeffding’s inequality to the tail probability of

$$\alpha_{n,j} - \alpha_j = \frac{1}{n} \sum_{i=1}^{n}((2Y_i - 1)\psi_j(X_i) - E[(2Y_i - 1)\psi_j(X_i)]),$$

and obtain the right-hand side is bounded by $2K_ne^{-n\epsilon/(2K_nB^2)}$. Because $K_n \log n/n \to 0$, the summation of this upper bound over $n$ is finite. Thus, the strong consistency follows from the Borel-Cantelli lemma.
Chapter 12

CONSISTENCY OF INDIRECT LEARNING METHODS

For indirect learning methods, classification rules do not have explicit expressions. The estimators for $\eta(X)$ are usually obtained by maximizing/minimizing some criteria. Therefore, for these methods, deriving the consistency of classification rules relies on working on the criteria functions directly. In this section, we focus on the maximum likelihood principle and the empirical risk minimization.

12.1 The maximum likelihood principle

In the maximum likelihood principle, the estimator for $\eta(x)$ is obtained by maximizing the likelihood function

$$\prod_{i=1}^{n} \eta(X_i)^{Y_i}(1 - \eta(X_i))^{1-Y_i}$$

over a family $F$. If denote $\eta_n(x)$ as the maximizer, the associate Bayes error is then

$$L_n = P(Y \neq I(\eta_n(X) > 1/2) | D_n).$$

Obviously, to obtain the consistency, the true function $\eta(x)$ should be assumed to belong to $F$. Second, we can allow $F$ to be rich enough including most of parametric models or even semiparametric models for $\eta(x)$. The richness (in fact tightness as well) of $F$ can be determined by the bracket covering number in $L_1(P_X)$, which is denoted as $N(\epsilon, F, L_1(P))$ and it defined as the minimal number of brackets $\{[\eta^1_L, \eta^1_U], ..., [\eta^m_L, \eta^m_U]\}$ such that $E[|\eta^k_L(X) - \eta^k_U(X)|] < \epsilon$

and for any $\eta'(x) \in F$, there exists some $k$ so that $\eta^k_L(x) < \eta(x) < \eta^k_U(x)$. Finally, since $\eta_n$ asymptotically maximizes the Kullback-Leibler distance, the consistency in terms Bayes error...
relied on how the Kullback-Leibler distance relates the Bayes error. We summarize the main result below.

**Strong consistency of the maximum likelihood principle** Let $F$ be a class of regression functions with values in $[0, 1]$ and for any $\epsilon > 0$, $N(\epsilon, F, L_1(P)) < \infty$. Then the maximum likelihood choice $\eta$ satisfies

$$L_n \equiv L(g_n) \rightarrow L(g^*)$$

with probability 1.

**Proof.** Let $F_\epsilon$ be those functions $\tilde{\eta}$ in $F$ such that $L(\tilde{\eta}) > L(g^*) + \epsilon$ where $\tilde{\eta}(x) = I(\tilde{\eta}(x) > 1/2)$. Then

$$P(L_n - L(g^*) > \epsilon) = P(\eta_n \in F_\epsilon) \leq P(\sup_{\tilde{\eta} \in F_\epsilon} (l_n(\tilde{\eta}) - l_n(\eta)) > 0),$$

where $l_n(\eta)$ denotes the log-likelihood function divided by $n$.

We note that $E[l_n(\eta)] - E[l_n(\tilde{\eta})]$ is exactly the Kullback-Leibler distance

$$E[\eta(X) \log \frac{\eta(X)}{\tilde{\eta}(X)} + (1 - \eta(X)) \log \frac{1 - \eta(X)}{1 - \tilde{\eta}(X)}].$$

By the relationship between the Kullback-Leibler distance and the Hellinger distance, the last expression is bounded from below by

$$E[\eta(X)(\sqrt{\eta(X)} - \sqrt{\tilde{\eta}(X)})^2 + (1 - \eta(X))(\sqrt{1 - \eta(X)} - \sqrt{1 - \tilde{\eta}(X)})^2],$$

which is further larger than

$$E[\eta(X)^2/2 + (1 - \eta(X))(\eta(X) - \tilde{\eta}(X))^2] = E[(\eta(X) - \tilde{\eta}(X))^2]$$

since for $\tilde{\eta} \in F_\epsilon$,

$$\epsilon \leq L(\tilde{\eta}) - L(g^*) \leq 2E[(\eta(X) - \tilde{\eta}(X))^2],$$

we conclude

$$\inf_{\tilde{\eta} \in F_\epsilon} \{E[l_n(\tilde{\eta})] - E[l_n(\eta)]\} \leq -\epsilon/2.$$

Hence,

$$P(L_n - L(g^*) > \epsilon) \leq P(\sup_{\tilde{\eta} \in F_\epsilon} (l_n(\tilde{\eta}) - l_n(\eta) - \{E[l_n(\tilde{\eta})] - E[l_n(\eta)]\}) > \epsilon/2).$$

Notice

$$(l_n(\tilde{\eta}) - l_n(\eta) - \{E[l_n(\tilde{\eta})] - E[l_n(\eta)]\})$$

is an empirical process indexed by $\eta$ in $F$, which has finite bracket number under $L_1(P)$ (also $L_2(P)$). We apply the concentration inequality for empirical process to obtain the above probability is bounded by $n^c_0 e^{-n\epsilon^2/c_1}$ for two constants $c_0$ and $c_1$. The strong consistency then holds by applying the Borel-Cantelli lemma.
12.2 Empirical risk minimization

This approach directly minimizes an empirical version the Bayes error, which is defined as

$$\hat{L}(g) = \frac{1}{n} \sum_{i=1}^{n} I(Y_i \neq g(X_i)),$$

where $g$ belongs to a class $C$. Suppose $g_n$ is the one minimizing this empirical risk. Then the consistency question becomes whether $L(g_n)$ converges to $L(g^*)$.

Clearly, if $g^*$ is not in $C$, we would expect $L(g_n)$ converges to $\inf_{g \in C} L(g)$ which may not be the same as $L(g^*)$. Similar to the previous section, the class $C$ can not be arbitrary. The most important class for $C$ in studying consistency is called VC class: a class of $g: \mathbb{R}^p \mapsto \{0, 1\}$ is a VC class with VC dimension $v$ if the collection of set

$$\{x : g(x) = 1\} \times \{0\}, \ \{x : g(x) = 0\} \times \{1\}$$

shatters at most $v$ points. We use $s(C, n)$ to denote the number of subsets of $n$ points picked out by the above collection. The main consistency result for the VC class is follows.

**Strong consistency for empirical risk minimization** If $C$ has a finite VC dimension $v$, then

$$P(\sup_{g \in C} |\hat{L}_n(g) - L(g)| > \epsilon) \leq 8s(C, n)e^{-n\epsilon^2/32}.$$

Therefore,

$$P(L(g_n) - \inf_{g \in C} L(g) > \epsilon) \leq 8s(C, n)e^{-n\epsilon^2/32}.$$

The strong consistency holds if $g^* \in C$.

The proof is long and uses combinatorics, Hoeffding inequality, and symmetrization techniques as well as Rademacher series. We refer interested readers to Chapter 12 and 13 of DGL book. It can be shown $s(C, n) \leq (en/v)^v$, where $v$ is the VC-dimension of $C$.

One application is to consider linear classifiers based on sieve estimation which we studied before. In this classification, $g(x) = I(\sum_{k=0}^{K_n} \alpha_{n,k}\psi_k(x) > 0)$. The empirical risk minimization is then minimizing

$$\sum_{i=1}^{n} I(Y_i \neq g(X_i)).$$
For this class of $g$, the VC dimension is no more than $K_n$. Therefore, the above result implies

$$P(L(g_n) - \int_{g \in C(K_n)} L(g) > \epsilon) \leq 8e^{nH(K_n/n)}e^{-n\epsilon^2/128},$$

where $H(x) = x \log x + (1 - x) \log(1 - x)$. The upper bound is $e^{-n\epsilon^2/128+o(n)}$ if $K_n = o(n)$. Furthermore, if $K_n$ increases, the projection of the true $\alpha(x)$ in $C(K_n)$ eventually converges to $\alpha(x)$, then

$$\int_{g \in C(K_n)} L(g) \rightarrow L(g^*).$$

We thus conclude the strong consistency of the linear classifiers based on sieve approximation. Another application of the above result is to show the consistency of classifiers based on data-dependent partitions, such like classification trees, data-dependent histogram estimator, automatical kernel rule and neural networks etc. To use the result, many proofs rely on the evaluation of the VC-dimension of interested classes.

### 12.3 Empirical risk minimization with complexity regularization

The requirement of a finite VC dimension for $C$ in the previous empirical risk minimization makes $C$ very small. Thus, the limit of the best rule in this class may be far from the Bayes risk, as $\inf_{g \in C} L(g) - L(g^*)$ can be very large. One way to solve this is to consider an increasing sequence of VC classes

$$C^{(1)} \subset C^{(2)} \subset ...$$

so that

$$\lim_{k \to \infty} \int_{g \in C^{(k)}} L(g) = L(g^*).$$

However, as the VC dimension increases, the stochastic error of

$$\hat{L}(g^{(k)}_{n}) - L(g^{(k)}_{n})$$

increases, where $g^{(k)}_{n}$ is the best classifier from $C^{(k)}$ minimizing $\hat{L}(g)$. The latter is because with large probability,

$$\hat{L}(g^{(k)}_{n}) \geq L(g^{(k)}_{n}) + R(n, k),$$

where $R(n, k) = \sqrt{32v_k \log(en)/n}$ and $v_k$ is the VC-dimension of $C^{(k)}$. Therefore, among all the best rules $\{g^{(k)}_{n}\}$, it is necessary to find the one balance the deviation from the Bayes error.
and the stochastic error. One way to do this is to choose from \( \{g_n^{(k)}\} \) the one minimizing the empirical risk with complexity regularization:

\[
\tilde{L}(g) = \hat{L}(g) + R(k,n).
\]

We remark that such a method has already been discussed in the model selection section, in the same spirit as the AIC, BIC and etc. The following theorem shows such an empirical risk minimization with complexity regularization (also called structural risk minimization) avoids overfitting the data.

**Strong universal consistency for structural risk minimization** Suppose \( \lim_k \int_{g \in C(k)} L(g) = L(g^*) \) and the VC-dimensions satisfy

\[
\sum_{k=1}^{\infty} e^{-v_k} < \infty.
\]

Then the classification rule based on structural risk minimization is strongly universally consistent.

**Proof.** Let \( \tilde{g} \) be the selected rule. We write

\[
L(\tilde{g}) - L(g^*) = (L(\tilde{g}) - \inf_{k \geq 1} \hat{L}(g_n^{(k)})) + (\inf_{k \geq 1} \hat{L}(g_n^{(k)}) - L(g^*)).
\]

For the first term in the right-hand side, we have

\[
P(L(\tilde{g}) - \inf_{k \geq 1} \hat{L}(g_n^{(k)}) > \epsilon) = P(L(\tilde{g}) - \hat{L}(g_n^{(k)}) > \epsilon)
\]

\[
\leq P(\sup_{k \geq 1} (L(g_n^{(k)}) - \hat{L}(g_n^{(k)})) > \epsilon) = \sum_{k \geq 1} P(|L(g_n^{(k)}) - \hat{L}(g_n^{(k)})| > \epsilon + R(n,k))
\]

\[
\leq \sum_{k \geq 1} 8n^{v_k} e^{-n(\epsilon + R(n,k))^2/32} \leq 8e^{-n\epsilon^2/32} \sum_k e^{-v_k}.
\]

Therefore, the first term converges to zero with probability one.

For the second term in the right-hand side, we first choose \( m \) large enough such that

\[
\int_{g \in C(m)} L(g) - L(g^*) < \epsilon/4
\]

and \( R(n,m) < \epsilon/4 \). Then,

\[
P(\inf_{k \geq 1} \hat{L}(g_n^{(k)}) - L(g^*) > \epsilon) \leq P(\inf_{k \geq 1} \hat{L}(g_n^{(k)}) - \int_{g \in C(m)} L(g) > 3\epsilon/4)
\]

\[
\leq P(\hat{L}(g_n^{(m)}) - \int_{g \in C(m)} L(g) > 3\epsilon/4) \leq P(\hat{L}(g_n^{(m)}) - \int_{g \in C(m)} L(g) > \epsilon/2)
\]
≤ 8n^{v_m} e^{-n\epsilon^2/128}.

Since $v_m < \infty$, we conclude that the second term converges to zero with probability one. Thus, $L(\tilde{g}) \rightarrow L(g^*)$ almost surely.

Actually, from the above proof, if the Bayes rule belongs to one of $C^{(k^*)}$, then we obtain

$$P(L(\tilde{g}) - L(g^*) > \epsilon) \leq e^{-n\epsilon^2/128} \sum_k e^{-v_k} + 8n^{v_k^*} e^{-n\epsilon^2/512}$$

for any $(n, \epsilon)$ satisfying $v_k \log(en) \leq n\epsilon^2/512$. Furthermore, from the above proof, we obtain

$$E[L(\hat{g})] - L(g^*) \leq \int_k \left( c_0 \sqrt{\frac{v_k \log n}{n}} + \left( \int_{g \in C^{(k)}} L(g) - L(g^*) \right) \right).$$

Thus, it is more clear that the structural risk minimization chooses a nearly optimal rule balance the bias and the stochastic errors based one a single class.
Chapter 13

CONVERGENCE RATES

We have established consistency or universal consistency for a number of classifiers. The next question is whether a universal rate of convergence to $L(g^*)$ is achievable for some classifier. For example, whether $E[L(g_n)] - L < c/n^a$ for all distributions.

Unfortunately, this is impossible unless the true distribution is known to lie in some restrictive spaces. Simply put, the Bayes error is too difficult a target to shoot at. Particularly, one result is the following.

No universal convergence rate Let $\{a_n\}$ be a sequence of positive numbers decreasing to zero and $a_1 \leq 1/16$. For every sequence of classifiers $\{g_n\}$, there exists a distribution of $(X, Y)$ such that $L(g^*) = 0$ but $E[L(g_n)] \geq a_n$.

The construction is given in Section 7 of DGL book. This shows that the universal rate does not exits. One has to give restrictive assumptions to the distributions $(X, Y)$ to obtain meaningful convergence rates so further considers the so-called optimal convergence rates.
Chapter 14

CLASSIFICATION ERROR ESTIMATION

One important problem is to know what actual performance a classifier $g_n$ is. Thus, it is useful to obtain a consistent estimator of the Bayes error associated with $g_n$, i.e., $P(g_n(X) \neq Y|D_n)$. Moreover, it will be desirable to find such estimator without any condition on the distribution of $(X,Y)$. This motivates a number of methods to search for distribution-free performance bound for error estimation.

14.1 Resubstitution estimate

One estimate is called the resubstitution estimate given by

$$\frac{1}{n} \sum_{i=1}^{n} I(g_n(X_i) \neq Y_i),$$

sometimes called the apparent error rate. However, this estimate is usually strongly biased since $g_n$ is tuned by the same data so $g_n$ behaves better on $D_n$ than any independent data. Consider 1-nearest neighborhood estimate. This error is zero. Now suppose we have a test sequence $(X_{n+1}, Y_{n+1}), \ldots, (X_{n+m}, Y_{n+m})$ independent of training data $D_n$. Then an estimator for $L(g_n)$ is

$$\hat{L}(g_n) = \frac{1}{n} \sum_{j=1}^{m} I(g_n(X_{n+j}) \neq Y_{n+j}).$$

Clearly, $E[\hat{L}(g_n)|D_n] = L(g_n)$ shows that this is an unbiased estimator for $L(g_n)$. Moreover,

$$Var(\hat{L}(g_n)|D_n) = \frac{L(g_n)(1-L(g_n))}{m} \leq \frac{1}{4m}.$$
Finally, by the Hoeffding inequality, we obtain that for any $\epsilon$,

$$P(|\hat{L}(g_n) - L(g_n)| > \epsilon|D_n) \leq 2e^{-2m\epsilon^2},$$

which gives a distribution-free performance bound for this error estimate.

### 14.2 Cross-validation estimate

However, in most of practice, we may not have a large number of test data to evaluate the error performance. Under this situation, one popular estimator for $L(g_n)$ is called cross-validation, leave-one-out, or U-method estimator. This idea is to delete one observation from training data and construct the classifier using the remaining $(n - 1)$ observations. This classifier is then used to predict the class label for the one deleted. The average predicted error will be treated as an estimator for the classification error. Formally, we denote $D_{n,-i}$ as the data after deleting observation $i$ and use $g_{n,-i}$ to denote the classifier obtained using the data without $i$th observation, then the cross-validation estimator is given by

$$\hat{L}_{CV} = \frac{1}{n} \sum_{i=1}^{n} I(g_{n,-i}(X_i) \neq Y_i).$$

Actually, $E[\hat{L}_{CV}] = E[L(g_{n-1})]$. This estimator should be treated as an estimator of $L(g_{n-1})$ instead of $L(g_n)$, but the difference between the latter two is negligible when $n$ is large. A general upper bound is available for the difference between $\hat{L}_{CV}$ and $L(g_n)$.

**Upper bound for the cross-validation performance** Assume that $g_n$ is a symmetric classifier; that is, the classifier remains the same if we permute data $D_n$. Then

$$E \left\{ (\hat{L}_{CV} - L(g_n))^2 \right\} \leq \frac{1}{n} + 6P(g_n(X) \neq g_{n-1}(X)).$$

**Proof.** We express

$$E \left\{ (\hat{L}_{CV} - L(g_n))^2 \right\} = E[\hat{L}_{CV}^2] - 2E[\hat{L}_{CV}L(g_n)] + E[L(g_n)^2].$$

Using the symmetry of $g_n$,

$$E[\hat{L}_{CV}^2] = E\left[\frac{1}{n^2} \sum_{i=1}^{n} I(g_{n,-i}(X_i) \neq Y_i)^2\right]$$

$$= \frac{1}{n} E[I(g_{n,-1}(X_1) \neq Y_1)] + \frac{n-1}{n} P(g_{n,-1}(X_1) \neq Y_1, g_{n,-2}(X_2) \neq Y_2)$$
\[ \leq \frac{1}{n} + P(g_{n,-1}(X_1) \neq Y_1, g_{n,-2}(X_2) \neq Y_2). \]

Additionally,
\[ E[\hat{L}_{CV} L(g_n)] = \frac{1}{n} \sum_{i=1}^{n} E[L(g_n)I(g_{n,-i}(X_i) \neq Y_i)] = P(g_n(X) \neq Y, g_{n,-1}(X_1) \neq Y_1). \]

We also have
\[ E[L_n^2] = P(g_n(X) \neq Y, g_n(X') \neq Y'). \]

Here, \((X, Y)\) and \((X', Y')\) are both i.i.d. copy of \((X_1, Y_1)\).

On the other hand, if we write \(g_{n,-i}(X_i)\) as \(g_{n,-1}(X_i; D_{n,-i})\) to indicate what training data are used can \(D_{n,-\{1,2\}}\) as \(\hat{D}_n\), additionally, we define
\[ A_{k,i,j} = \{g_n(X_k; X_i, Y_i, X_j, Y_j, \hat{D}_n) \neq Y_k\} \]

and
\[ B_{k,i} = \{g_{n-1}(X_k, X_i, Y_i, \hat{D}_n) \neq Y_k\}, \]

then
\[ E[(\hat{L}_{CV} - L(g_n))^2] \leq \frac{1}{n} + P(B_{1,2}B_{2,1}) - P(A_{n+1,1,2}B_{2,1}) \]
\[ + P(A_{n+1,1,2}A_{n+2,1,2}) - P(A_{n+1,1,2}B_{1,2}). \]

Note that
\[ P(B_{1,2}B_{2,1}) - P(A_{n+1,1,2}B_{2,1}) \]
\[ = P(B_{n+1,n+2}B_{n+2,n+1}) - P(A_{n+1,n+2}B_{n+2,2}) \]
\[ = P(B_{n+1,n+2}B_{n+2,n+1}) - P(A_{n+1,n+2}B_{n+2,n+1}) \]
\[ + P(A_{n+1,n+2}B_{n+2,n+1}) - P(A_{n+1,n+2}B_{n+2,2}) \]
\[ \leq P(B_{n+1,n+2}A_{n+1,n+2}) + P(B_{n+2,n+1}B_{n+2,2}) \]
\[ \leq P(B_{n+1,n+2}A_{n+1,n+2}) + P(B_{n+2,n+1}B_{n+2,2}) + P(A_{n+2,n+1}A_{n+2,n+2}B_{n+2,2}) \]
\[ = 3P(A_{n+2,n+1,2}B_{n+2,2}). \]

Similarly,
\[ P(A_{n+1,1,2}A_{n+2,1,2}) - P(A_{n+1,1,2}B_{1,2}) \]
\[ = P(A_{n+1,1,2}A_{n+2,1,2}) - P(A_{n+1,n+2}A_{n+2,1,2}) + P(A_{n+1,n+2}B_{n+2,1,2}) - P(A_{n+2,1,2}B_{n+1,2}) \]
\[ \leq P(A_{n+1,1,2}A_{n+1,n+2}) + P(A_{n+1,n+2}B_{n+1,2}) \]
\[ \leq P(A_{n+1,1,2}B_{n+1,2}) + P(B_{n+1,2}A_{n+1,n+2}) + P(A_{n+1,n+2}B_{n+1,2}) \]
\[ = 3P(A_{n+1,1,2}B_{n+1,2}). \]
Combining all these results and noticing that
\[ P(A_{n+1,1,2} \Delta B_{n+1,2}) = P(A_{n+2,1,2} \Delta B_{n+2,2}) = P(g_n(X) \neq g_{n-1}(X)), \]
we obtain the upper bound.

Using this result, we can justify the validity of the cross-validation for many classification rules. For instance, for \( k \)-nearest neighborhood, \( g_n(X) \neq g_{n-1}(X) \) only when \( X_n \) is among the \( k \)-nearest neighborhood but this probability is just \( k/n \) due to the symmetry. We hence obtain that
\[ E[(\hat{L}_{CV} - L(g_n))^2] \leq \frac{6k+1}{n}. \]

For the kernel rule when the kernel has a bounded support in the unit sphere and \( K(x) \geq \beta \) for \( \|x\| \leq \rho \), since
\[ P(g_n(X) \neq g_{n-1}(X)) \leq P(\sum_{i=1}^{n-1}(2Y_i - 1)K_h(X - X_i) \leq K_h(X - X_n), K_h(X - X_n) > 0) \]
\[ \leq P(\sum_{i=1}^{n}(2Y_i - 1)K_h(X - X_i) \leq h^{-p}, \|X_n - X\| \leq h), \]
which, according to Petrov’s result, is further bounded by
\[ E \left[ I(\|X_n - X\| \leq h), \frac{2C}{2\beta \sqrt{n}} \right] \]
\[ \leq \frac{C}{\beta \sqrt{n}} \int \int |y - x| < h \frac{P(dy)}{P(\|X - x\| < h)} P(dx) \]
\[ \leq \frac{C_p}{\beta \sqrt{n}} (1 + \frac{1}{\rho})^{p/2}. \]

Hence, we conclude
\[ E(\hat{L}_{CV} - L(g_n))^2 \leq \frac{1}{n} + \frac{6C_p}{\beta \sqrt{n}} (1 + \frac{1}{\rho})^{p/2}. \]

Here, \( C \) and \( C_p \) are constants with the latter depending on \( p \).

14.3 Other error estimates

There are many other estimates for estimating classification errors. We list a few here but will not discuss them in details (the details can be found in Chap 31 of DLG book). One estimate is called smooth error count defined by
\[ \frac{1}{n} \sum_{i=1}^{n} (I(Y_i = 0)r(\eta_n(X_i)) + I(Y_i = 1)(1 - r(\eta_n(X_i)))) , \]
where $\eta_n(x)$ is the estimator for $P(Y = 1|X)$ and $r(x)$ is a monotone increasing function satisfying $r(1/2 - u) = 1 - r(1/2 + u)$. For example, $r(u) = u, 1/(1 + e^{1/2 - cu})$. Such an estimator has smaller variability compared to the resubstitution estimate, which corresponds to $r(u) = I(u \geq 1/2)$. A second estimate called posterior probability estimate arises from the observation

$$L(g_n) = E[I(\eta_n(X) \leq 1/2)\eta(X) + I(\eta_n(X) > 1/2)(1 - \eta(X))|D_n].$$

Its definition is given by

$$\frac{1}{n} \sum_{i=1}^{n} I(\eta_n(X_i) \leq 1/2)\eta_n(X_i) + I(\eta_n(X_i) > 1/2)(1 - \eta_n(X_i)).$$

A further version is the cross-validation of such an estimate:

$$\frac{1}{n} \sum_{i=1}^{n} I(\eta_{n,-i}(X_i) \leq 1/2)\eta_{n,-i}(X_i) + I(\eta_{n,-i}(X_i) > 1/2)(1 - \eta_{n,-i}(X_i)),$$

where $\eta_{n,-i}$ is the estimator for $\eta$ based on the data without $i$th observation. A third estimate is essentially the same as the cross-validation; but instead leaving-one-out, we divide the data into $m$ blocks and leave one block each time for training classifiers. The error is the average of the predicted errors over $m$ blocks. The last estimate we list here is the bootstrap estimate. Specifically, we take a random sample with replacement observations and construct the classifier based on the bootstrap sample. We then evaluate the error on the observations not selected in this sample. We repeat this process many times and use the average error as the estimate for $L(g_n)$. 

14.3. OTHER ERROR ESTIMATES
Chapter 15

CONCENTRATION INEQUALITIES

We list a number of useful concentration inequalities, which are usually used for obtaining error tail behavior so are very important when studying asymptotic learning theory. We do not provide the proof for all these results. Interesting readers can find the proofs in relevant references.

15.1 Concentration inequalities for random variables

Concentration inequalities for random variables are essentially inequalities for the tail probability of the deviation of a random variable from its mean (or median). It often takes form as

\[ P(\xi(X_1, ..., X_n) - E[\xi(X_1, ..., X_n)] \geq x) \leq \exp\{-\frac{x^2}{2\nu}\}. \]

One of the most classical one is

\[ P(f(Z) - E[f(Z)] \geq x) \leq \exp\{-\frac{x^2}{2L}\}, \]

where \( Z \) follows \( N(0, I_d) \) and \( f \) is Lipschitz continuous with Lipschitz constant \( L \). In the above inequality, \( E[f(Z)] \) can be replaced by the median. Since then, there have been may techniques developed to obtain more general inequalities. One commonly used technique is based on the Cramer-Chernoff method (Chernoff inequality):

\[ P(Z \geq x) \leq \exp\{-\psi_Z^*(x)\}, \]

where \( \psi_Z^*(x) = \sup_\lambda \{\lambda x - \log E[e^{\lambda Z}]\} \) and it is called the Cramer transformation. Using this method, one can obtain the well-known Hoeffding’s inequality and Bennett inequality from bounded (half-bounded) random variables:
Hoeffding’s inequality: \( X_1, ..., X_n \) are independent and \( X_i \in [a_i, b_i] \),

\[
P(\sum_i X_i - \sum_i E[X_i] \geq x) \leq \exp\{ -\frac{2x^2}{\sum_i (b_i - a_i)^2} \}.
\]

Bennett’s (Bernstein’s) inequality: \( X_1, ..., X_n \) are independent and square integrable with \( X_i \leq b \) for some constant \( b > 0 \),

\[
P(\sum_i X_i - \sum_i E[X_i] \geq x) \leq \exp\{ -\frac{\nu b^2}{2(\nu + bx/3)} \} \leq \exp\{ -\frac{x^2}{2(v + bx/3)} \},
\]

where \( \nu = \sum_i E[X_i^2] \) and \( h(y) = (1 + y) \log(1 + y) - y \).

Similar inequality holds for unbounded variables but with additional assumptions for moments:

Bernstein’s inequality for unbounded variables: \( X_1, ..., X_n \) are independent and \( \nu = \sum_i E[X_i^2] \). If for \( k \geq 3 \), \( E[(X_i)^k] \leq k!\nu c^{k-2}/2 \), then

\[
P(\sum_i X_i - \sum_i E[X_i] \geq \sqrt{2\nu x + cx}) \leq \exp\{-x\}.
\]

Some concentration inequalities are bounded using information theory, where one basic technique relies on the following result

Information inequalities (Han’s inequality) Suppose \( Y = (Y_1, ..., Y_n) \) takes finite values in a product measure space and write \( Y^{(i)} = (Y_1, ..., Y_{i-1}, Y_{i+1}, ..., Y_n) \). Then

\[
h_s(Y) \leq \frac{1}{n-1} \sum_{i=1}^n h_s(Y^{(i)})
\]

where \( h_s(Y) = -\sum_y P(Y = y) \log P(Y = y) \) (Shannon entropy).

In this direction, one of the most useful results is the bounded difference inequality.

Bounded difference inequality: If \( |\xi(x_1, ..., x_i, ..., x_n) - \xi(x_1, ..., y_i, ..., x_n)| \leq c_i \), then

\[
P(\xi(X_1, ..., X_n) - E[\xi(X_1, ..., X_n)] \geq x) \leq \exp\{ -\frac{2x^2}{\sum_i c_i} \}.
\]

Many other techniques have been implemented in the discovery of the inequalities, including \( \phi \)-entropy method, isoperimetric method, contraction inequality and Rademacher sequence and etc.
15.2 Concentration inequalities for random processes

In many situations, we are concerned about the tail behavior of the suprema of stochastic processes (sometimes we call maximal inequalities). There are some concentration inequalities developed for random processes and they essentially use concentration inequalities for random variables. However, the complexity/size of the index (VC-class, bounded entropy with bracket or other) for random processes also characterize the upper bound.

First, it is easy to generate the concentration inequality for the bounded difference from random variables to random processes.

**Bounded difference process:** If \( a_{it} \leq X_i(t) \leq b_{it} \) and let \( L_2^2 = \sum_i \sup_{t \in T} (b_{it} - a_{it})^2 \), then

\[
P(\sup_{t \in T} \sum_i X_i(t) - E[\sup_{t \in T} \sum_i X_i(t)] \geq x) \leq \exp\left\{-\frac{2x^2}{2L^2}\right\}.
\]

Another type of the inequalities for such suprema are the following.

**Talagrand’s inequality** Let \( \mathcal{F} \) be some countable family of measurable functions such that \( \|f\|_{\infty} \leq b < \infty \). Let

\[
Z = \sup_{f \in \mathcal{F}} \left| \sum_{i=1}^{n} (f(X_i) - E[f(X_i)]) \right| \quad \text{and} \quad \sigma^2 = \sup_{f \in \mathcal{F}} \left[ \sum_{i=1}^{n} \text{Var}(f(X_i)) \right].
\]

Then for any \( \epsilon, x > 0 \),

\[
P(Z \geq (1 + \epsilon)E[Z] + \sigma \sqrt{2\nu x} + \nu(\epsilon)bx) \leq e^{-x}
\]

with \( \nu = 4 \) and \( \nu(\epsilon) = 2.5 + 32\epsilon^{-1} \) and

\[
P(Z \geq (1 - \epsilon)E[Z] - \sigma \sqrt{2\nu'x} - \nu'(\epsilon)bx) \leq e^{-x}
\]

with \( \nu' = 5.4 \) and \( \nu'(\epsilon) = 2.5 + 43.2\epsilon^{-1} \).

**Maximal inequality based on packing number** Let \( \psi \) be convex, nondecreasing, nonzero and \( \psi(0) = 0 \) and \( \limsup_n \psi(x)\psi(y)/\psi(cxy) < \infty \). Then

\[
\| \max_{1 \leq i \leq m} X_i \|_{\psi} \leq K \psi^{-1}(m) \max_i \|X_i\|_{\psi}.
\]

When \( X(\cdot) \) is a random process indexed by \( t \), suppose

\[
\|X(s) - X(t)\|_{\psi} \leq Cd(s, t),
\]
then
\[ \| \sup_{d(s,t) \leq \delta} |X(s) - X(t)| \|_{\psi} \leq K \left[ \int_0^{\eta} \psi^{-1}(D(\epsilon, d)) d\epsilon + \delta \psi^{-1}(D^2(\eta, d)) \right]. \]

The direct application of the above result gives:

**Maximal inequality for sub-Gaussian processes** If \( X(t) \) is sub-Gaussian with
\[ P(|X(s) - X(t)| > x) \leq 2e^{-x^2/2\sigma^2(s,t)}, \]
then
\[ E[ \sup_{d(s,t) \leq \delta} |X(s) - X(t)|] \leq K \int_0^{\delta} \sqrt{\log D(\epsilon, d)} d\epsilon. \]

When random processes are empirical processes, the concentration inequalities can be determined using the covering number or the bracket covering number. The derivation either uses the Radamacher sequence or chain arguments.

**Concentration inequality empirical processes**: If \( \mathcal{F} \) is bounded by 1 and it satisfies
\[ \sup_{Q} N(\epsilon, \mathcal{F}, L_2(Q)) \leq (K/\epsilon)^V \]
or
\[ N(\epsilon, \mathcal{F}, L_2(P)) \leq (K/\epsilon)^V, \]
then for any \( 1 \geq \sigma^2 \geq \sup_{f \in \mathcal{F}} \text{Var}(f(X)) \) and \( \delta > 0, \)
\[ P^*(\|G_n\|_{\mathcal{F}} > t) \leq C \left( \frac{1}{\sigma} \right)^{2V} \left( 1 + \frac{t}{\sigma} \right)^{3V+\delta} e^{-\frac{1}{2} \frac{\sigma^2 + (3+\delta)/\sqrt{n}}{1 + \frac{t}{\sigma}}}. \]

In summary, concentration inequalities (and maximal inequalities) play essential role in the theory for statistical learning, although so far, they are only used in proving “consistency” in these problems. What inequalities to be used are often specific to problems. In most cases, we can write the estimators in terms of empirical processes, then the results from the last inequalities are useful. However, when data are not identically distributed or there is no clear expression in terms of empirical process, one may borrow from other inequalities, including Hoeffding’s inequality, Bernstein inequality, or bounded difference inequality.
Part III

HIGH-DIMENSIONAL DATA
Chapter 16

DIMENSION REDUCTION

In many applications, the dimension of feature variables is very high and sometime much larger than the number of observations. For example, the number of SNPs in genomewide association study, the number of genes in microarray data, the number of voxels in image analysis, the number of confounders in large field study are often in thousands or as many as millions. When the number of feature variables is so high, classical statistical inference and learning is no longer applicable. However, in practice, one usually believes that the feature information related to outcome should be a small number of dimension which are hidden among these thousands of collected feature variables. The question is how to dig out these informative feature/combined features. One way is to apply all different kinds of dimension reduction techniques to capture the essential low-dimensional structures within data. We will focus on these dimension reduction approaches (sometimes called data mining approaches) in this chapter. Another approaches are to select only informative feature variable via regularization, which usually couple with some regression models. The latter will be described in next chapter.

16.1 Unsupervised dimension reduction

The first group of dimension techniques are called unsupervised dimension reduction, which means that we only explore feature data to find out a low-dimensional representation of the whole data.

Because of the similar nature of unsupervised dimension reduction to the one of unsupervised learning, many techniques discussed before for the latter can be used to achieve dimension reduction. They include principal component analysis, latent component analysis (factor analysis), independent component analysis and multidimensional scaling.
In principal component analysis, one decompose the centered feature matrix $X_{n \times p}$ via singular value decomposition into

$$X_{n \times p} = U_{n \times n} S_{n \times p} V_{p \times p},$$

where the left-upper $n \times n$ block in $S_{n \times p}$ is a diagonal matrix containing all the singular values and the rest in $S_{n \times p}$ is zero. Then the columns of $U_{n \times n}$, called principal components, are the projection of the feature matrix on the space of eigenvectors. If the first $k \ (k << \min(n, p))$ principal components explain the large proportion of the variation in $X_{n \times p}$, they can be used to represent all the features so we achieve dimension reduction. This calculation is applicable even if $p$ is of millions, which has been applied in determining population substructure in genomewide association studies. One difficulty of using the principal component is interpretation, as they are all linear combinations of the feature variables so they may not really correspond to meaningful physical quantities. An alternative way to reduce the dimension is to use the information (loading factors) in the principal components to determine important variables in the original data.

Factor analysis assumes that feature variable satisfies

$$X_{p \times 1} = \Lambda_{p \times k} f_k + \epsilon_{p \times 1},$$

where $\epsilon$ is uncorrelated errors and each component may have difference variance, and $E[f] = 0$ and $Var(f) = I$. Moreover, $f$, the latent factors are assumed to be independent of $\epsilon$. Therefore, these latent factors with dimension $k$ can be treated as a low-dimensional representation of $X$ when $k$ is much smaller than $p$. The calculation of $\Lambda$ and $f$ also rely on the singular value decomposition of the feature value so it can handle sufficiently large $p$. Comparatively, independent component analysis look for latent component which are independent. However, the obtained components have not obvious order and it is unclear that how large dimension this method can handle.

The purpose of multidimensional scaling method is exactly a dimension reduction since it projects the whole data into a low-dimensional space. The usefulness of the multidimensional scaling method in high-dimension feature space is not clear. There are many other unsupervised dimension reduction techniques, including genetic and evolutionary algorithm, vector quantization (in image analysis), nonlinear principal component, density networks, principal curves and random projection (in text mining). Interesting readers can read this paper and its cited references

16.2 Supervised dimension reduction

Compared to unsupervised dimension reduction, supervised dimension reduction uses outcome variables to determine a low-dimension information from feature variables. The most popular methods include projection pursuit regression and sliced inverse regression. The former have been already discussed before so we focus on the second one.

Let $Y$ denote outcome variable. A supervised dimension reduction is to find a low dimensional representation of feature variable $X$ so that this low-dimensional information is sufficient (under certain sense) to explain the relationship between $X$ and $Y$. The slice inverse regression is one of practically useful approaches. Particularly, it looks for some linear combinations of $X$ to achieve goal. Specifically, we look for a matrix $B_{p \times k}$ such that $Y$ is independent of $X$ given $B^T X$; in other words, all the information between $Y$ and $X$ is contained in between $Y$ and $B^T X$, which is a low-dimensional representation of $X$ if $k$ is much smaller than $p$. We note that any single index models for $Y$ given $X$ satisfy this condition.

To estimate $B$, we first standardize $X$ so that each column has mean zero and different columns are correlated, still denoted by $X$. We then discretize $Y$ into a categorized variable, denoted by $\tilde{Y}$. Within each category (slice) of $\tilde{Y} = h$, we compute the average of the standardized $X$, denoted as $\hat{E}_h$. We then obtain

$$\hat{M} = \sum_h \frac{n_h}{n} \hat{E}_h \hat{E}_h^T,$$

where $n_h$ is the number of the observations in slice $\tilde{Y} = h$. Let $\mu_1, ..., \mu_p$ be the eigenvectors corresponding the descending eigenvalues of $\hat{M}$. The first $k$ eigenvectors become the columns of $B$ matrix. The choice of $k$ can be obtained by performing a series of chi-square test. Theoretically, it can be shown the above estimators are consistent if $X$ given $Y$ is elliptically distributed.

Another approach in the same spirit as the sliced inverse regression is called the sliced average variance estimation. The only difference is that $\hat{M}$ in this approach is defined as

$$\sum_h \frac{n_h}{n} (I - \hat{V}_h)^2,$$

where $\hat{V}_h$ is the sample covariance of $X$ within slice $\tilde{Y} = h$. It is claimed that this approach can capture the largest proportion of the central subspace, a concept specifically developed for this dimension reduction technique. Interesting reader can read this paper and its cited references

In addition to these approaches, it is clear that many models we have discussed in supervised learning can be used to obtain low-dimension reduction of $X$, including generalized additive models, multivariate adaptive regression splines, tree regression, neural networks and so on. All these methods can not handle very large number of feature variables as seen in the examples of genetics and bioinformatics.

When the dimension of feature variable is ultra-high (thousands or even millions), any model-based dimension reduction is infeasible. One simplest approach is to screen each feature variable first and choose the top ones with high correlation to outcome variable. Although this simple approach was widely used years ago, the justification for this method was not given till a recent paper by Fan and Lv (2008), where they showed that for linear regression, the selection will contain the true important predictions with probability one. Some further results were later obtained for generalized models, additive models and proportional hazards models. However, one essential assumption in these justifications is that the significant marginal effects in screening stage must imply a significant effect in multivariate regressions, which may not be practically true especially when feature variable are highly dependent. Additionally, it is unclear how well these approaches perform with moderate number of observations. Interestingly reader can read this selective review paper

Chapter 17

VARIABLE SELECTION VIA REGULARIZATION

In this chapter, we focus on selecting important feature variables in regression problems. We discuss this issue in general regression models, from linear regression to generalized linear regression. Nevertheless, there are many more problems on variable selection in current literature which cannot be covered in this single chapter.

Most methods for variable selection in regression models can be summarized into the following optimization:

$$n^{-1}l_n(\beta) - \sum_{j=1}^{p} p_\lambda(\|\beta_j\|),$$

where $l_n(\beta)$ is often the log-likelihood function or the summed squared errors based on the regression model, and $\beta$ are the coefficients associated with $X$. The regularization term is given by the penalty $p_\lambda(\|\beta_j\|)$. The different methods use different penalty. Some of these methods have been revised before, including LASSO, adaptive LASSO and SCAD. The general penalty condition can be found in Johnson et al (2008).

17.1 Fixed feature dimension

This section only considers the situation when $p$ is fixed. Since all these regularization approaches are similar and can be treated as solving some estimating equations, we present the details in this paper

17.2 Increasing feature dimension

The validity of these methods can be also justified for high-dimensional settings when $p$ increases with $n$ and could be much larger than $n$. We provide a list of papers for readers. There are many other papers with some modified penalties or applications to regression models for censored data, partial linear regression models, single index regression models and robust regression etc. One thing we want to emphasize here is that when $p$ increases with $n$, many asymptotic results heavily rely on concentration inequalities or maximal inequalities which were reviewed in Part II.

For LASSO in linear model,


For bridge estimator in linear model,


For generalized linear model and LASSO,


For variable selection via multiple testing,


For linear regression with concave penalty,


For generalized additive model,

17.3 Computing algorithms

One important issue in variable selection with high-dimensional feature variables is how to develop an efficient algorithm for computing estimates. There have been a number of algorithms developed so far.

In the LASSO estimation, the objective function to be maximized is a concave function but non-differentiable along with a larger number of parameters. One early version of solving this problem is called the least angle regression (LARS, Efron et al 2004).

Instead of giving a vector of estimates, the LARS solution consists of a curve denoting the solution for each value of the penalty parameter. Essentially, the algorithm is similar to forward stepwise regression, but instead of including variables at each step, the estimated parameters are increased in a direction equiangular to each one’s correlations with the residual. This algorithm is as fast as forward selection and gives a full piecewise linear solution path, which is useful in cross-validation or similar attempts to tune the model. The algorithm is stable and effective even if \( p \) is much larger than \( n \). However, with large noisy data and high multicollinearity, the selected variables may not be the actual underlying causal variables.

Another simple, fast and stable algorithm is coordinate decent algorithm (Friedman et al. 2007; Wu and Lange 2008). The idea is straightforward, we cycle through all the parameters and update one at each iteration. The slight difference in Wu and Lange (2008) is to update the parameter which has the most negative value of gradient.

When the objective in not concave, such like the SCAD penalty, a local quadratic approximation (LQA) is suggested for optimization. The idea is to locally approximate the objective function by a quadratic function. Specifically, the penalty term can be approximated by

\[
p_\lambda(||\beta_j^*||) + \frac{1}{2} \frac{p_\lambda'(||\beta_j^*||)}{||\beta_j^*||} (\beta_j - \beta_j^2).
\]

To avoid numerical problem, one sets the updated \( \beta_j \) to be zero if \( \beta_j^* \) is very close to zero. A better approximation is to use the local linear approximation by

\[
p_\lambda(||\beta_j^*||) + p_\lambda'(||\beta_j^*||)(||\beta_j|| - ||\beta_j^*||).
\]
Both algorithms can be treated as some special cases of the minorization-maximization (MM) algorithm (Hunter and Lange, 2000).
Chapter 18

HIGH-DIMENSIONAL MULTIPLE TESTING

The challenge of high-dimensionality in statistical theory not only appear in the number of feature variables but also shows up when one conducts a large number of tests. Some applications are seen in genomewide association study when one tests the association between phenotype and each single nucleotide polymorphism (SNP) or genotype or haplotype, and in image analysis when statistical tests are conducted for each voxel. The usual Bonferroni correction is too conservative when the number of tests is large. On the other hand, in most of high-dimensional test problems, one may be more interested in discovering significant tests than just controlling type I errors. Thus, different error rates to be controlled should be used.

The first group of error rates are called family wise error rates, which is the probability of rejecting one or more of the hypotheses erroneously. Bonferroni procedure applies to this case by setting the threshold for p-value to be $\gamma/m$, where $\gamma$ is a pre-specified significance level and $m$ the number of hypotheses. This procedure is valid when tests are assumed to be independent but even so, this test tends to be conservative. To relax that, another similar rate is called $k$-family wise error rates, which is the proportion of rejecting $k$ or more of hypotheses erroneously.

A more popular error rate used in high-dimensional multiple testing is is called false discovery rate, because one may be more interested in knowing the error among those tests identified to be significant. This concept was introduced in Benjamini et al. (1995) and is defined as the expected proportion of the number of falsely rejected hypotheses among total number of rejected hypotheses. So far there are several implementations of FDR under different assumptions. In Benjamini et al. (1995), one assumes independence among test statistics and denotes the p-
values from \(m\) tests are \(p_1, p_2, \ldots, p_m\). We order them from the smallest to the largest, denoted by \(p_{(1)}, p_{(2)}, \ldots, p_{(m)}\). Then the so-called Benjamini-Hochberg (BH) procedure finds maximal \(k\) such that \(p_{(k)} \leq \gamma(k/m)\), where \(\gamma\) is a pre-specified level of significance. It is known to control FDR at level \(m_0/n\gamma\) under independence assumption of test statistics. Later, Bejamini and Yekutieli (2001) extends the BH procedure to control FDR at level \(\gamma\) under positive dependence conditions. It uses the linear step-up procedure, and it finds maximal \(k\) such that \(p_{(k)} \leq \gamma k/m(\sum_{i=1}^{m} i^{-1})^{-1}\). Note that these two approaches do not estimate the number of the null hypotheses \(m_0\) but the procedures could be improved if they know \(m_0\). To estimate \(m_0\), Storey et al. (2001) uses the fact that independent null p-values are distributed uniformly on \([0, 1]\) and then plug the estimator into the FDR-estimator. Benjamini et al. (2006) estimate \(m_0\) in a two-stage adaptive control of FDR. Simultaneously, Storey and Hastie (2001) proposes the so-called \(q\)-value for multiple testing based on pFDR, which is defined as the expected proportion of the number of false rejections over the number of rejections given the number of rejections is at least one. The \(q\)-value is the minimum pFDR where the statistic is declared significant.

Finally, there are other works using mixture distributions (Cai and Sun 2009) or hyper criticism approaches (Donoho et al). We give a number of papers below:


Chapter 19

APPLICATIONS

In this chapter, we provide a list of papers involving high-dimensional data complexity in current literature related to some specific applications, including spatial-temporal data, image data, dynamic treatment regimes, and population stratifications etc. The papers are listed in chronical order.

19.1 Graph models

A list of paper readings are:


19.2 Spatial-temporal data

A list of paper readings are:

Chen X. et al. (2010). Learning spatial-temporal varying graphs with applications to climate analysis.

19.3 Individual tailored treatment

A list of paper readings are:


19.4 Population stratification

A list of paper readings are:

