Statistical Methods for Machine Learning

Derek Li

Contents

1 Introduction

1.1 Terminology

$X = (X_1, \cdots, X_p)$ are p predictor variables and suppose there are n observations, $X_p = (X_{p1}, \cdots, X_{pn})$. Output varible typically denoted by Y.

1.2 Supervised and Unsupervised

Supervised statistical learning involves building a statistical model for predicting or estimating an output based on one or more inputs. For unsupervised statistical learning, there are inputs but no supervising output.

Alternatively we could say for supervised learning, all data is labeled and the algorithms learn to predict the output from the input data; for unsupervised learning, all data is unlabeled and the algorithms learn to inherent structure from the input data.

1.2.1 Supervised Learning

In the regression problem, Y is quantitative, but Xs could be quantitative or qualitative. The goal is to predict a new observation not in the training data set - what is the expected value of y_0, given x_0.

In the classification problem, Y takes values in a finite, unordered set (e.g., binary or dichotomous, categorical), but Xs could be quantitative or qualitative.The goal is to predict a new observation not in the training data set - what is the class probability of y_0, given x_0.

1.2.2 Unsupervised Learning

$X = (X_1, \cdots, X_p)$ are measured variables on n observations and usually p is large. Note that there is no response (Y) to supervise the algorithm.

Two common methods for unsupervised learning:

- Clustering: Understand the relationships between the ***observations.*** Cluster the observations on the basis of the variables measured and identify the group to which each observation belongs (number of groups is unknown).
- Principal components analysis (PCA): Understand the relationships between the ***variables*** and reduce the number of variables to a smaller number.

1.3 Simple Model

Suppose that we observe a quantitative response Y and p different predictors, $X_1, \cdots, X_p$. Assume a general form of a relationship by a model

$$Y = f(X) + \varepsilon,$$

where f is fixed and unknown and ε is a random error term, which is independent of X and has mean zero. ε captures measurement errors and other discrepancies such as omitted predictors.

There are two main reasons that we estimate f : prediction and inference. With a more accurate f we can make predictions of Y at new points $X = x$, understand which components of $X = (X_1, X_2, \cdots, X_p)$ are important in explaining Y, and depending on the complexity of f, understand how each component X_i of X affects Y.

1.3.1 Prediction

We can predict Y using $\widehat{Y} = \widehat{f}(X)$. The accuracy of $\widehat{Y}$ depends on:

- Reducible error: an error introduced by because of $\widehat{f}$ not being the perfect estimate of f but by using the most appropriate algorithm, this error can be reduced.
- Irreducible error: Y is a function of ε that cannot be predicted using X, and we cannot reduce the error introduced by ε.

Theorem 1.1. Consider a given estimate $\widehat{f}$ and a set of predictors X that yields the prediction $\widehat{Y} = \widehat{f}(X)$. Assume $\widehat{f}$ and X are fixed. Then

$$\mathbb{E}\left[\left(Y - \widehat{Y}\right)^2\right] = \underbrace{\left[f(X) - \widehat{f}(X)\right]^2}_{\text{Reducible}} + \underbrace{\mathrm{Var}[\varepsilon]}_{\text{Irreducible}} .$$

Proof. We have

$$\begin{aligned}\mathbb{E}\left[\left(Y - \widehat{Y}\right)^2\right] &= \mathbb{E}\left[\left(f(X) + \varepsilon - \widehat{f}(X)\right)^2\right] \\ &= \mathbb{E}\left[\left(f(X) - \widehat{f}(X)\right)^2\right] + \mathbb{E}[\varepsilon^2] + \mathbb{E}\left[2\varepsilon\left(f(X) - \widehat{f}(X)\right)\right].\end{aligned}$$

Since $\mathbb{E}[\varepsilon] = 0$ and $\varepsilon \perp X$, we have

$$\mathbb{E}\left[2\varepsilon\left(f(X) - \widehat{f}(X)\right)\right] = 0.$$

Besides,

$$\mathbb{E}[\varepsilon^2] = \mathbb{E}[\varepsilon^2] - (\mathbb{E}[\varepsilon])^2 = \mathrm{Var}[\varepsilon].$$

Therefore,

$$\mathbb{E}\left[\left(Y - \widehat{Y}\right)^2\right] = \left[f(X) - \widehat{f}(X)\right]^2 + \mathrm{Var}[\varepsilon].$$

□

1.3.2 Inference

Inference is the goal of classical statistical methods: we want to understanding the relationship between Y and Xs and how strong is the relationship, etc.

1.4 Estimation Methods

We can use linear and non-linear approaches to estimate f. For these methods, we observe a set of n different data points (training data) and train the specific method to find an estimate $\widehat{f}$ for f.

1.4.1 Parametric Methods

Parametric methods involve a two-step model-based approach.

- Assume the functional form of f.
- Use the training data to train the model.

The problem of estimating f reduces to one of estimating a set of parameters without fitting an entirely arbitrary function f so that we call this method parametric.

Assuming a parametric form for f simplifies the problem of estimating f because it is generally much easier to estimate a set of parameters (in the linear model).

Nevertheless, the model we choose will usually not match the true unknown form of f, so that our estimate will be poor. We can try flexible model but it requires estimating a greater number of parameters, which is more complex and can lead to overfitting (following the errors or noise too closely).

1.4.2 Non-Parametric Methods

Non-parametric methods do not make explicit assumptions about the functional form of f, but seek an estimate of f that gets as close to the data points as possible.

These methods can fit a wider range of possible shapes for f but since these methods do not reduce the problem of estimating f to a small number of parameters, a very large number of observations is required in order to obtain an accurate estimate for f.

1.5 Flexibility and Interpretability

Restrictive models (e.g., linear regression) are more interpretable, but flexible models (e.g., splines, boosting) are less interpretable and they can have complicated estimates of f.

Generally, as flexibility increases, interpretability decreases. Also, as flexibility increases, prediction accuracy may decrease due to over-fitting.

1.6 Assessing Model Accuracy

1.6.1 Measuring the Quality of Fit

In the regression setting, the most commonly-used measure is the mean squared error (MSE):

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{f}(\mathbf{x}_i))^2,$$

where $\widehat{f}(\mathbf{x}_i)$ is the prediction that $\widehat{f}$ gives for the ith observation.

Since the MSE above is computed using the training data that was used to fit the model, so we call it training MSE. Nevertheless, we are interested in the accuracy of the predictions that we obtain when we apply our model to previously unseen test data not used to train the model.

Suppose we have new observations $(\mathbf{x}_0, y_0)$.We want to choose the model that gives the lowest test MSE

$$\text{Ave}(y_0 - \widehat{f}(\mathbf{x}_0))^2,$$

as opposed to the lowest training MSE.

Recall that $\text{Var}[\varepsilon]$ is the irreducible error, and it corresponds to the lowest achievable test MSE among all possible methods.

Here is a fundamental property of statistical learning that holds regardless of the data set and the model being used.

Property 1.1. As the flexibility of the model increases, there is a monotone decrease in the training MSE and a U-shape in the test MSE.

Definition 1.1 (Overfitting)**.** When a given model yields a small training MSE but a large test MSE, we are said to be overfitting the data.

Overfitting happens because the model may pick up some patterns that are just caused by random chance rather than by true properties of the unknown function f. The test MSE will be large because the patterns do not exist in the test data.

Note that we almost always expect the training MSE to be smaller than the test MSE since most models seek to minimize the training MSE.

In practice, test observations usually are not available, we still cannot select a model that minimizes the training MSE, because there is no guarantee that the method with the lowest training MSE will also have the lowest test MSE. To estimate test MSE, we may use cross-validation.

1.6.2 The Bias-Variance Trade-Off

Suppose we have fit a model $\widehat{f}(x)$ to some training data and let (x_0, y_0) be a test observation drawn from the population. Suppose the true model is $Y = f(X) + \varepsilon$ and $f(x) = \mathbb{E}[Y|X = x]$, then for given x_0, test MSE is

$$\mathbb{E}\left[\left(y_0 - \widehat{f}(x_0)\right)^2\right] = \text{Var}\left[\widehat{f}(x_0)\right] + \left[\text{Bias}\left(\widehat{f}(x_0)\right)\right]^2 + \text{Var}[\varepsilon].$$

Proof. We have

$$\begin{aligned}
\mathbb{E}\left[\left(y_0 - \widehat{f}(x_0)\right)^2\right] &= \mathbb{E}\left[\left(y_0 - \mathbb{E}\left[\widehat{f}(x_0)\right] + \mathbb{E}\left[\widehat{f}(x_0)\right] - \widehat{f}(x_0)\right)^2\right] \\
&= \mathbb{E}\left[\left(\mathbb{E}\left[\widehat{f}(x_0)\right] - y_0\right)^2\right] + \mathbb{E}\left[\left(\widehat{f}(x_0) - \mathbb{E}\left[\widehat{f}(x_0)\right]\right)^2\right] \\
&= \mathbb{E}\left[\left(\mathbb{E}\left[\widehat{f}(x_0)\right] - f(x_0) - \varepsilon\right)^2\right] + \operatorname{Var}\left[\widehat{f}(x_0)\right] \\
&= \mathbb{E}\left[\left(\mathbb{E}\left[\widehat{f}(x_0)\right] - f(x_0)\right)^2\right] + \mathbb{E}[\varepsilon^2] - (\mathbb{E}[\varepsilon])^2 + \operatorname{Var}\left[\widehat{f}(x_0)\right] \\
&= \left[\operatorname{Bias}\left(\widehat{f}(x_0)\right)\right]^2 + \operatorname{Var}[\varepsilon] + \operatorname{Var}\left[\widehat{f}(x_0)\right].
\end{aligned}$$

□

Here are some comments:

- Expected test MSE can never lie below $\operatorname{Var}(\varepsilon)$, the irreducible error.
- To minimize the expected test error, we need to select a model that simultaneously achieves low variance and low bias.
- Variance refers to the amount by which $\widehat{f}$ would change if we estimated it using a different training data set. Ideally, the estimate for f should not vary too much between training sets. More flexible models have higher variance.
- Bias refers to the error introduced by approximating a real life problem, which may be extremely complicated, by a much simpler model. Generally, more flexible models result in less bias.
- As flexibility increases, the variance will increase and the bias will decrease. The relative rate of change in variance and bias determines whether the test MSE increases or decreases.
- As we increase the flexibility, the bias tends to initially decrease faster than the variance increases so that the expected test MSE declines. At some point increasing flexibility has little impact on the bias but starts to significantly increase the variance so that the test MSE increases.

2 Linear Regression

Linear regression is a simple model to supervised learning. It assumes that the dependence of Y on $X_1, \cdots, X_p$ is linear.

2.1 Simple Linear Regression

We assume a model

$$Y = \beta_0 + \beta_1 X + \varepsilon,$$

where β_0 and β_1 are two unknown parameters represents the intercept and slope respectively, and ε is the error term. ε is to catch what we miss with the model. Assume ε is independent of X and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

We have

$$\mathbb{E}[Y|X] = \mathbb{E}[\beta_0 + \beta_1 X + \varepsilon|X] = \beta_0 + \beta_1 X$$

and

$$\mathrm{Var}[Y|X] = \mathrm{Var}[\beta_0 + \beta_1 X + \varepsilon|X] = \mathrm{Var}[\varepsilon|X] = \sigma^2.$$

Given some estimates $\widehat{\beta}_0$ and $\widehat{\beta}_1$ for the model coefficients, we can predict Y given X by

$$\widehat{y} = \widehat{\beta}_0 + \widehat{\beta}_1 x,$$

where $\widehat{y}$ indicates a prediction of Y given $X = x$.

2.1.1 Interpretation

β_0 is the intercept term, i.e., the expected value of Y when $X = 0$. β_1 is the slope, i.e., the average increase in Y associated with a one-unit increase in X.

2.1.2 Estimation of the Parameters by Least Squares

Let $\widehat{y}_i = \widehat{\beta}_0 + \widehat{\beta}_1 x_i$ and $e_i = y_i - \widehat{y}_i$, where $i = 1, \cdots, n$, and n is the sample size and e_i is the ith residual of observation i.

We define residual sum of squares

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2,$$

and by the least squares method, we choose $\widehat{\beta}_0$ and $\widehat{\beta}_1$ to minimize RSS:

$$\begin{aligned}
\widehat{\beta}_0 &= \overline{y} - \widehat{\beta}_1 \overline{x}, \\
\widehat{\beta}_1 &= \frac{\sum\limits_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sum\limits_{i=1}^{n} (x_i - \overline{x})^2}.
\end{aligned}$$

We can prove that $\mathbb{E}\left[\widehat{\beta}_0\right] = \beta_0$ and $\mathbb{E}\left[\widehat{\beta}_1\right] = \beta_1$, i.e., $\widehat{\beta}_0$ and $\widehat{\beta}_1$ are unbiased estimators for β_0 and β_1 respectively.

The unbiased means that if we average the values of $\widehat{\beta}_0$ or $\widehat{\beta}_1$ obtained over a huge number of data sets, the average would exactly equal to β_0 or β_1 and an unbiased estimator does not systematically over or under estimate the true parameter.

2.1.3 Assessing the Accuracy of the Coefficient Estimates

A standard error of a statistic is the estimated standard deviation of the statistic and it reflects how it varies under repeated sampling:

$$\mathrm{SE}\left(\widehat{\beta}_0\right)^2 = \sigma^2\left[\frac{1}{n} + \frac{\overline{x}^2}{\sum\limits_{i=1}^{n}(x_i - \overline{x})^2}\right], \mathrm{SE}\left(\widehat{\beta}_1\right)^2 = \frac{\sigma^2}{\sum\limits_{i=1}^{n}(x_i - \overline{x})^2},$$

where $\sigma^2 = \mathrm{Var}[\varepsilon]$, provided ε_i are independent of each other.

When the standard error increases, i.e., values of the estimators are more spread out, gives an inaccurate representation of the true population parameters.

Here are some comments for SE:

- When the x_i are more spread out, $\mathrm{SE}\left(\widehat{\beta}_1\right)$ is smaller.
- When $\overline{x} = 0, \mathrm{SE}\left(\widehat{\beta}_0\right)^2 = \mathrm{Var}\left[\widehat{\beta}_0\right] = \frac{\sigma^2}{n}$.
- $\widehat{\beta}_0 = \overline{y} - \widehat{\beta}_1\overline{x}$ and when $\overline{x} = 0, \widehat{\beta}_0 = \overline{y}, \mathrm{Var}\left[\overline{y}\right] = \frac{\sigma^2}{n}$.

SE can be used to compute confidence intervals. A 95% confidence interval is defined as a range of values s.t. with 95% probability, the range will contain the true unknown value of the parameter. For linear regression, the 95% confidence interval for $\beta_i, i = 0$ or 1, is

$$\left[\widehat{\beta}_i - 1.96 \cdot \mathrm{SE}\left(\widehat{\beta}_i\right), \widehat{\beta}_i + 1.96 \cdot \mathrm{SE}\left(\widehat{\beta}_i\right)\right],$$

i.e., there is approximately a 95% chance that the interval will contain the true value of β_i.

SE can be used to perform hypothesis tests on the coefficients. The most common hypothesis test involves testing if there is some relationship between X and Y:

$$H_0 : \beta_1 = 0$$

versus

$$H_A : \beta_1 \neq 0.$$

To test the null hypothesis, we compute a t-statistic given by

$$t = \frac{\widehat{\beta}_1 - 0}{\mathrm{SE}(\widehat{\beta}_1)},$$

which will have a t distribution with $n-2$ degrees of freedom, assuming $\beta_1 = 0$. We can compute the probability of observing any value equal to $|t|$ or larger under H_0 and we call the probability the p-value.

2.1.4 Assessing the Accuracy of the Model

The residual standard error (RSE) is an estimate of σ, the standard deviation of ε, given by

$$\text{RSE} = \sqrt{\frac{1}{n-2}\text{RSS}}.$$

RSE is an absolute measure of the lack of fit of the model to the data and is measured in the units of Y. RSE $= a$ means actual Y deviate from the true regression line by a units on average. It is not always clear what constitutes a good RSE. The R^2 statistic provides an alternative measure of fit:

$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}},$$

where TSS $= \sum_{i=1}^{n}(y_i - \overline{y})^2$ is the total sum of squares.

R^2 statistic provides an alternative measure of fit and it measures the proportion of variability in Y that can be explained using X. $R^2 = b, b \in [0,1]$ indicates that $b\%$ of the variability in the response is explained by the regression. $R^2 \approx 0$ occurs when the linear model is wrong, or the inherent error σ^2 is high, or both.

In simple linear regression, $R^2 = r^2$, where r is the correlation between X and Y :

$$r = \text{Cor}(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}.$$

In SLR, r^2 can be used to assess the fit.

2.2 Multiple Linear Regression

We assume a model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p + X_p + \varepsilon,$$

where β_j is the average effect on Y of a one unit increase in X_j, holding all other predictors fixed. But predictors usually change together. Also note that claims of causality should be avoided for observational data.

2.2.1 Estimation of Parameters

Given estimates $\widehat{\beta}_0, \widehat{\beta}_1, \cdots, \widehat{\beta}_p$, we can make predictions by

$$\widehat{y} = \widehat{\beta}_0 + \widehat{\beta}_1 x_1 + \cdots + \widehat{\beta}_p x_p.$$

We estimate β_i as the values

$$\min \text{RSS} = \min \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 = \min \sum_{i=1}^{n}\left(y_i - \widehat{\beta}_0 - \widehat{\beta}_1 x_{i1} - \cdots - \widehat{\beta}_p x_{ip}\right)^2.$$

2.2.2 Testing the Relationship

We test

$$H_0 : \beta_1 = \cdots = \beta_p = 0$$

against

$$H_A : \text{at least one } \beta_j \text{ is non} - \text{zero}.$$

We use F-statistic

$$F = \frac{(\text{TSS} - \text{RSS})/p}{\text{RSS}/(n-p-1)} \sim F_{p(n-p-1)}.$$

2.2.3 Deciding on Important Variables

The most direct method is called all subsets or best subsets regression: we compute the least squares fit for all possible subsets and then choose between them based on some criterion that balances training error with model size. There are 2^p models and thus if p is large, we need an automated approach.

- Forward selection. Begin with the null model that contains an intercept but no predictors. Fit p SLRs and add to the null model the variable that results in the lowest RSS. Add to that model the variable that results in the lowest RSS for the new two-variable model. This approach is continued until some stopping rule is satisfied.
- Backward selection. Start with all variables in the model, and remove the variable with the largest p-value, i.e., the variable is the least statistically significant. The new $(p-1)$-variable model is fit and remove the variable with the largest p-value. This procedure continues until a stopping rule is reached. For example, we may stop when all remaining variables have a significant p-value defined by some significance threshold.

2.2.4 Qualitative Predictors

Example 2.1. For the ethnicity (Caucasian, African American or Asian) variable, we create two dummy variables. The first could be

$$x_{i1} = \begin{cases} 1 & \text{if } i\text{th person is Asian} \\ 0 & \text{if } i\text{th person is not Asian} \end{cases},$$

and the second could be

$$x_{i2} = \begin{cases} 1 & \text{if } i\text{th person is Caucasian} \\ 0 & \text{if } i\text{th person is not Caucasian} \end{cases}.$$

We have

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \varepsilon_i = \begin{cases} \beta_0 + \beta_1 + \varepsilon_i & \text{if } i\text{th person is Asian} \\ \beta_0 + \beta_2 + \varepsilon_i & \text{if } i\text{th person is Caucasian} \\ \beta_0 + \varepsilon_i & \text{if } i\text{th person is AA} \end{cases}.$$

The level with no dummy variable, AA in this example, is known as the baseline.

2.2.5 Extensions of the Linear Model

In practice, there is a synergy effect and in statistics it is referred to as an interaction effect and thus we will introduce interaction term in the model.

Sometimes it is the case that an interaction term has a very small p-value, but the associated main effects do not.

We follow the hierarchy principle: If we include an interaction in a model, we should also include the main effects, even if the p-values associated with their coefficients are not significant. The rationale for the principle is that interactions are hard to interpret without main effects and their meaning is changed.

Here is an example for interactions between qualitative and quantitative variables.

Example 2.2. Without an interaction term, the model takes the form

$$y_i = \beta_0 + \beta_1 x_i + \begin{cases} \beta_2 \\ 0 \end{cases} = \beta_1 x_i + \begin{cases} \beta_0 + \beta_2 \\ \beta_0 \end{cases} .$$

With an interaction, we have

$$y_i = \beta_0 + \beta_1 x_i + \begin{cases} \beta_2 + \beta_3 x_i \\ 0 \end{cases} = \begin{cases} (\beta_0 + \beta_2) + (\beta_1 + \beta_3) x_i \\ \beta_0 + \beta_1 x_i \end{cases} .$$

2.2.6 Non-Linear Relationships

We can extend the linear model to accommodate non-linear relationships, using polynomial regression.

3 Classification

In classification model, the response variable Y is qualitative. Our goals are to

- Build a classifier $C(X)$ that assigns a class label from C to a future unlabeled observation or assess the uncertainty in each classification.
- Understand the roles of the different predictors among $X = (X_1, \cdots, X_p)$.

3.1 Bayes Optimal Classifier

Suppose the M labels in C are numbered $1, \cdots, M$. Let

$$p_m(\mathbf{x}) = P(Y = m|X = \mathbf{x}), m = 1, \cdots, M.$$

The Bayes optimal classifier at $\mathbf{x}$ is

$$C(\mathbf{x}) = j \text{ if } p_j(\mathbf{x}) = \max\{p_1(\mathbf{x}), \cdots, p_M(\mathbf{x})\},$$

i.e., to choose the label with highest probability and thus minimizing the probability that is makes an error. Hence, Bayes classifier (using the true $p_m(\mathbf{x})$) has smallest error.

3.1.1 Model Accuracy

We measure the performance by training error rate, which is the proportion of mistakes, given by

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \widehat{C}(\mathbf{x}_i)),$$

where I is an indicator function.

Test error rate is associated with a set of test observations of the form $(\mathbf{x}_0, y_0)$ is given by

$$\text{Test Error Rate} = \text{Ave}(I(y_0 \neq \widehat{C}(\mathbf{x}_0))).$$

A good classifier is one where the test error is smallest and actually the Bayes classifier $C(\mathbf{x})$ produces the lowest possible test error rate, called the Bayes error rate and thus we call it optimal Bayes classifier.

3.1.2 Bayes Decision Boundary

Definition 3.1 (Bayes Decision Boundary)**.** The line that represents the points where the probability is exactly 50% is called the Bayes decision boundary.

3.1.3 Limitation

In practice, we do not know the true population probabilities and thus we need to define other classifiers that can approximate $p_m(\mathbf{x})$.

3.2 K-Nearest Neighbors

To estimate the conditional distribution, we can use K-nearest neighbors (KNN) classifier.

Given a $K \in \mathbb{N}$ and a test observation $\mathbf{x}_0$, the KNN classifier first identifies the K points in the training data that are closest to $\mathbf{x}_0$, represented by $\mathcal{N}_0$. It then estimates the conditional probability

$$P(Y = j|X = \mathbf{x}_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j).$$

Finally, KNN applies Bayes rule and classifies $\mathbf{x}_0$ to the class with te largest probability.

The flexibility is $\frac{1}{K}$. With $K = 1$, the KNN training error rate is 0, but the test error rate may be quite high.

3.3 Logistic Regression

Logistic regression uses the logistic (sigmoid-S shaped) function:

$$p(X) = P(Y = 1|X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}.$$

Wherefore,

$$\text{logit} = \ln\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X.$$

The monotone transformation is called the log odds or logit transformation of $p(X)$.

3.3.1 Estimation

We use maximum likelihood to estimate the parameters:

$$L(\beta_0, \beta) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i)),$$

where gives the joint probability of the observed zeros and ones in the data.

In R, we use the `glm` function.

3.3.2 Interpretation

A one-unit increase in X is associated with an increase in the log odds of Y by $\widehat{\beta}_1$ units. $\widehat{\beta}_0$ is typically not of interest.

3.3.3 Maximum Likelihood Estimation

Definition 3.2 (Fisher's Information Matrix)**.** Let l be the log likelihood function. The Fisher's information matrix is defined as

$$\mathcal{I}(\theta) := \mathbb{E}\left[-\frac{\partial^2}{\partial \theta^2} l(X|\theta)\right] = \mathbb{E}[-H],$$

where H is Hessian matrix.

3.4 Multiple Logistic Regression

We use

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}} \Rightarrow \ln\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p,$$

where $X = (X_1, \cdots, X_p)$ are p predictors. We use the maximum likelihood method to estimate β's.

Definition 3.3 (Confounding Variable)**.** A confounding variable is a factor associated with both the predictor and response.

Definition 3.4 (Interaction)**.** Interaction among variables, also known as effect modification, exists when the effect of one predictor variable on the response depends on the particular level or value of another predictor variable.

3.5 Multi-Classes Logistic Regression

We could use R package glmnet.

3.6 Discriminant Analysis

When we have more than two response classes, when the classes are well-separated (the parameter estimates for the logistic regression model are unstable), or if n is small and the distribution of the predictors X is approximately normal in each of the classes (the linear discriminant model is mode stable), we will use discriminant analysis rather than logistic regression.

Theorem 3.1 (Bayes Theorem)**.** $P(Y = k|X = \mathbf{x}) = \frac{P(X=\mathbf{x}|Y=k)\cdot P(Y=k)}{P(X=\mathbf{x})}$

Let $\pi_k = P(Y = k)$ be the marginal or prior probability that a randomly chosen observation comes from the kth class. Let $f_k(X) = P(X = \mathbf{x}|Y = k)$ be the density for X in class k, then

$$p_k(X) = P(Y = k|X = \mathbf{x}) = \frac{\pi_k f_k(\mathbf{x})}{\sum\limits_{i=1}^{K} \pi_i f_i(\mathbf{x})}.$$

Assume predictors X are continuous variables and conditional class densities of $X = \mathbf{x}$ are multivariate normal distributions for each class k, i.e.,

$$f_k(\mathbf{x}) \sim \mathcal{N}(\mu_k, \Sigma_k), k = 1, \cdots, K.$$

We classify an observation to the class k for which $p_k(\mathbf{x})$ is largest.

3.6.1 Linear Discriminant Analysis

(1) **Case:** $p = 1$. The Gaussian density has the form

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma_k}\right)^2}.$$

Assume $\sigma_k = \sigma$, we have

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma}\right)^2}}{\sum\limits_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu_l}{\sigma}\right)^2}}.$$

We can consider the term regardless of k as a constant, and have

$$p_k(x) = C\pi_k e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma}\right)^2} \Rightarrow \ln(p_k(x)) = \ln C + \ln \pi_k - \frac{(x-\mu_k)^2}{2\sigma^2},$$

and have

$$\ln(p_k(x)) = C' + \frac{x\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \ln \pi_k.$$

So the objection function, or linear discriminant function is

$$\delta_k(x) = \ln \pi_k - \frac{\mu_k^2}{2\sigma^2} + \frac{x\mu_k}{\sigma^2} x.$$

The Bayes classifier is equivalent to assigning x to the class with the largest discriminant score $\delta_k(x)$. Notice that $\delta_k(x)$ is linear because $\sigma_k^2 = \sigma^2$ for all k.

Example 3.1. Assume $K = 2$ and $\pi_1 = \pi_2 = 0.5$, then the Bayes classifier assigns an observation to class 1 if $\delta_1(x) > \delta_2(x)$. Besides, the decision boundary is at $x = \frac{\mu_1+\mu_2}{2}$, i.e., when $\delta_1(x) = \delta_2(x)$.

In practice, we are not able to calculate the Bayes classifier and LDA approximates the Bayes classifier when we substitute estimated parameters using the training observations.

$$\widehat{\pi}_k = \frac{n_k}{n},\ \widehat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i,\ \widehat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \widehat{\mu}_k)^2 = \sum_{k=1}^{K} \frac{n_k - 1}{n-K} \cdot \widehat{\sigma}_k^2,$$

where $\widehat{\sigma}_k^2 = \frac{1}{n_k-1} \sum\limits_{i:y_i=l} (x_i - \widehat{\mu}_k)^2$. Hence,

$$\widehat{\delta}_k(x) = \ln \widehat{\pi}_k - \frac{\widehat{\mu}_k^2}{2\widehat{\sigma}^2} + \frac{\widehat{\mu}_k}{\widehat{\sigma}^2} x.$$

Note that, we need to assume the observations within each class come from a normal distribution with a class-specific mean vector and a common variance σ^2. With the test data, we can compute

$$\text{Bayes Test Error Rate} = \text{Ave}(I(y_0 \neq C(x_0))),$$

and

$$\text{LDA Test Error Rate} = \text{Ave}(I(y_0 \neq \widehat{C}(x_0))).$$

(2) **Case:** $p > 1$. We assume $X = (X_1, \cdots, X_p)$ is multivariate Gaussian distribution with a class-specific mean vector and a common covariance matrix, so that each individual predictor follows a one-dimensional normal distribution. We write

$$X \sim \mathcal{N}(\mu, \Sigma),$$

where μ is a $p \times 1$ mean vector and Σ is a $p \times p$ covariance matrix. The multivariate normal density function of X is

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)}.$$

Similarly, we have the discriminant function

$$\delta_k(\mathbf{x}) = \mathbf{x}^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \ln \pi_k = c_{k0} + c_{k1}x_1 + \cdots + c_{kp}x_p,$$

provided Σ is common to all K classes and $\mathbf{x} = (x_1, \cdots, x_p)$. Bayes classifier assigns an observation $X = \mathbf{x}$ to the class for which $\delta_k(\mathbf{x})$ is largest.

3.6.2 Confusion Matrix

Suppose we have a binary classifier $C(\mathbf{x})$ and $C(\mathbf{x})$ can make two types of errors: $C(\mathbf{x}) = G_1$ when in fact $\mathbf{x}$ is from G_2 and $C(\mathbf{x}) = G_2$ when in fact $\mathbf{x}$ is from G_1. A confusion matrix can display the error information.

Notice that errors that are very lopsided could be a problem. Hence we also consider:

- False positive rate: The fraction of negative example that are classified as positive.
- False negative rate: The fraction of positive example that are classified as negative.
- Sensitivity/True positive rate: The fraction of positive example that are classified as positive.
- Specificity/True negative rate: The fraction of negative example that are classified as negative.

We may modify the threshold value but there is a trade-off between these rates.

3.6.3 ROC and AUC

ROC curve, an acronym for receiver operating characteristics, is for simultaneously displaying the two types of errors for all possible thresholds.

The overall performance of a classifier, summarized over all possible thresholds, is given by the area under the ROC curve, called AUC. Higher AUC is better.

3.6.4 Quadratic Discriminant Analysis

With Gaussian conditional class probabilities but different Σ_k in each class, we get quadratic discriminant analysis (QDA). The discriminant function is

$$\delta_k(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_k)^T\Sigma_k^{-1}(\mathbf{x} - \mu_k) + \ln \pi_k.$$

4 Resampling Methods

Resampling methods involve repeatedly drawing samples from a training set and refitting a model of interest on each sample in order to obtain additional information about the fitted model.

Here are the brief introductions of two resampling methods:

- Cross-Validation: to estimate the test error associated with a given learning method to evaluate its performance; to select the appropriate level of flexibility.
- Bootstrap: to provide a measure of accuracy (bias) or variability (precision) of a parameter estimate or a given learning method.

4.1 Prediction Error Estimates

Usually, training error rate underestimates the test error rate. The best solution is having a large test dataset and we can apply our fitted model to test set to estimate the test error, but a test dataset often is not available.

Besides, we can make a mathematical adjustment to the training error rate in order to estimate the test error rate, such as C_p, AIC, and BIC.

Or, we can use cross-validation method that holds out a subset of the training observations as the test data.

4.2 Cross-Validation

4.2.1 Validation Set Approach

We randomly divide the available set of samples into two parts: a training set and a validation set. The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set. The resulting validation set error provides an estimate of the test error.

Algorithm 1: Validation Set Approach

```
Input: training set $S = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$, set of parameters values $\Theta$ and
       learning algorithm $A$
partition $S$ equally into $S_{\text{training}}$ and $S_{\text{validation}}$;
for each $\theta \in \Theta$ do
    $\text{model}_\theta = A(S_{\text{training}}; \theta)$;
    $\text{error}(\theta) = L_{\text{validation}}(\text{model}_\theta)$;
end
Output: $\theta^* = \arg\min_\theta[\text{error}(\theta)]$ and $\text{model}_{\theta*} = A(S; \theta^*)$
```

Note that test error is typically assessed using (1) MSE when the response is quantitative; (2)

misclassification rate when the response is qualitative.

Here are some drawbacks:

- The validation estimate of the test error can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set.
- Only a subset of the observations are used to fit the model. Since learning methods tend to perform worse when trained on fewer observations, the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set.

4.2.2 K-Fold Cross-Validation

The estimates can be used to select best model and give an idea of the test error of the final chosen model.

We randomly divide the data into K equal-sized parts, leave out part k, fit the model to the other $K-1$ parts, and obtain predictions for the left-out kth part, for all $k = 1, \cdots, K$.

Let C_k denotes the indices of the observations in part k and suppose there are n_k observations in part k. If n is a multiple of K then $n_k = \frac{n}{K}$, and we have

$$\mathrm{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{MSE}_k = \frac{1}{K} \sum_{k=1}^{K} \sum_{i \in C_k} \frac{(y_i - \widehat{y}_i)^2}{n_k} = \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in C_k} (y_i - \widehat{y}_i)^2 ,$$

where $\widehat{y}_i$ is the fit for observation i, obtained from the data with part k removed.

Algorithm 2: K-Fold Cross-Validation

```
Input: training set $S = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$, set of parameters values $\Theta$, learning
       algorithm $A$ and integer $K$
partition $S$ equally into $S_1, \cdots, S_K$;
for each $\theta \in \Theta$ do
    for $i = 1, \cdots, K$ do
        $\mathrm{model}_{i,\theta} = A(S \backslash S_i; \theta)$;
        $\mathrm{error}_i(\theta) = L_{S_i}(\mathrm{model}_{i,\theta})$;
    end
    $\mathrm{error}(\theta) = \frac{1}{K} \sum_{i=1}^{K} L_{S_i}(\mathrm{model}_{i,\theta})$;
end
Output: $\theta^* = \arg\min_\theta[\mathrm{error}(\theta)]$ and $\mathrm{model}_{\theta*} = A(S; \theta^*)$
```

4.2.3 Leave-One Out Cross-Validation (LOOCV)

LOOCV is special case of K-fold cross-validation when $K = n$.

With least squares linear or polynomial regression, we have

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \widehat{y}_i}{1-h_i}\right)^2,$$

where $\widehat{y}_i$ is the ith fitted value from original least squares fit and h_i is the leverage.

LOOCV is sometimes useful but typically does not shake up the data enough. The estimates from each fold are highly correlated and hence their average can have high variance.

4.2.4 Bias-Variance Trade-Off

Since each training set is only $\frac{K-1}{K}$ as big as the original training set, the estimates or prediction error will typically be biased upward and the bias is minimized when $K = n$ (LOOCV) but with high variance.

$K = 5$ or 10 provides a good compromise for this bias-variance trade-off.

4.2.5 Cross-Validation for Classification

We have

$$\mathrm{CV}_{(K)} = \sum_{k=1}^{K}\frac{n_k}{n}\mathrm{Err}_k = \frac{1}{n}\sum_{k=1}^{K}\sum_{i\in C_k} I\left(y_i \neq \widehat{y}_i\right).$$

The estimated standard deviation of $\mathrm{CV}_{(K)}$ is

$$\mathrm{SE}(\mathrm{CV}_{(K)}) = \sqrt{\frac{\sum\limits_{k=1}^{K}\left(\mathrm{Err}_k - \overline{\mathrm{Err}_k}\right)^2}{K-1}}.$$

4.3 Bootstrap

The bootstrap is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method. For example, it can provide an estimate of the standard error of a coefficient, or a confidence interval for the coefficient.

Rather than repeatedly obtaining independent data sets from the population, we instead obtain bootstrap data sets by repeatedly sampling observations from the original data set with replacement and the size is the same as the original dataset. Each bootstrap data set is used to obtain an estimate, say α.

Let the bootstrap data set be Z^{*r} and use Z^{*r} to produce a new bootstrap estimate for α, called $\widehat{\alpha}^{*r}$, where $r = 1, \cdots, B$ and this procedure is repeated B times of some large value of B. Besides, the standard error of these bootstrap estimates is

$$\mathrm{SE}\left(\widehat{\alpha}\right) = \sqrt{\frac{1}{B-1}\sum_{r=1}^{B}\left(\widehat{\alpha}^{*r} - \overline{\widehat{\alpha}^{*}}\right)^2}.$$

5 Linear Model Selection and Regularization

It is very important to reduce the number of features or choose the most informative ones. This is not just to obtain a solution, but to avoid overfitting. Three methods to improve the linear model by reducing the number of features: subset selection, shrinkage (regularization) and dimension reduction. Note that these methods can apply to logistic regression or/and other types of models.

5.1 Subset Selection

The method finds among the p predictors the ones that are the most related to the response.

5.1.1 Best Subsets

Algorithm 3: Best Subset Selection

```
let $M_0$ denote the null model, which contains no predictors, and the model simply
  predicts the sample mean for each observation;
for $k = 1, \cdots, p$ do
    fit all $\binom{p}{k}$ models that contain exactly $k$ predictors;
    pick the best among these $\binom{p}{k}$ models, and call it $M_k$ using the smallest RSS or
      largest $R^2$;
end
select a single best model from among $M_0, \cdots, M_p$ using cross-validated prediction
  error, $C_p$, AIC, BIC, or adjusted $R^2$.
```

For a broader class of models (with different link functions), we instead use the deviance $-2l$, where l is the maximized log-likelihood function.

5.1.2 Forward Stepwise

For computational reasons, best subset selection cannot be applied with large p. It may also suffer from statistical problems when p is large: an enormous search space can lead to overfitting and high variance of the coefficient estimates. Therefore, stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

Forward stepwise selection (considering $\frac{p^2+p+2}{2}$ possible models) is a computationally efficient alternative to best subset selection (considering 2^p possible models). But forward stepwise selection is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors.

Algorithm 4: Forward Stepwise Selection

```
let $M_0$ denote the null model, which contains no predictors;
for $k = 0, \cdots, p-1$ do
    consider all $p-k$ models that augment the predictors in $M_k$ with one additional predictor;
    choose the best among these $p-k$ models, and call it $M_{k+1}$ using the smallest RSS or highest $R^2$;
end
select a single best model from among $M_0, \cdots, M_p$ using cross-validated prediction error, $C_p$, AIC, BIC, or adjusted $R^2$.
```

5.1.3 Backward Stepwise

Backward stepwise selection provides an efficient alternative to best subset selection.

Algorithm 5: Backward Stepwise Selection

```
let $M_p$ denote the full model, which contains all $p$ predictors;
for $k = p, \cdots, 1$ do
    consider all $k$ models that contain all but one of the predictors in $M_k$, for a total of $k-1$ predictors;
    choose the best among these $k$ models, and call it $M_{k-1}$ using the smallest RSS or highest $R^2$;
end
select a single best model from among $M_0, \cdots, M_p$ using cross-validated prediction error, $C_p$, AIC, BIC, or adjusted $R^2$.
```

Backward selection requires that $n > p$ so that the full model can be fit. In contrast, forward stepwise can be used even when $n < p$ and so is the only viable subset method when p is very large.

5.1.4 Choosing the Optimal Model

We want to choose a model with a low test error and there are two approaches:

- Indirectly estimate test error by making an adjustment to the training error to account for the bias due to overfitting.

  ∗ Mallow's C_p :

$$C_p = \frac{1}{n}(\text{RSS} + 2d\hat{\sigma}^2),$$

  where d is the total number of parameters in the current model, $\hat{\sigma}^2$ is an estimate of the variance of the error and is obtained using the full model with p parameters. C_p statistic adds a penalty of $2d\hat{\sigma}^2$ to the training RSS. If $\hat{\sigma}^2$ is an unbiased estimate of σ^2, then C_p is an unbiased estimate of test MSE. Thus We select the model that has the lowest C_p value.

$*$ Akaike information criterion (AIC) is defined for a large class of models fit by maximum likelihood:

$$\text{AIC} = -2\ln L + 2d,$$

where L is the maximized value of the likelihood function for the estimated model. In the model with Gaussian errors, maximum likelihood and least squares are same and

$$\text{AIC} = \frac{1}{n\widehat{\sigma}^2}(\text{RSS} + 2d\widehat{\sigma}^2).$$

$*$ Bayesian information criterion (BIC):

$$\text{BIC} = \frac{1}{n}(\text{RSS} + \ln(n)d\widehat{\sigma}^2),$$

where n is the number of observations. We select the model that has the lowest BIC value. Since $\ln(n) > 2$ for any $n > 7$, the BIC statistic places a heavier penalty on models with many variables and hence results in the selection of smaller models than C_p.

$*$ Adjusted R^2 : for a least squares model with d variables,

$$\text{Adjusted } R^2 = 1 - \frac{\text{RSS}/(n-d-1)}{\text{TSS}/(n-1)}.$$

We select the model that has the largest adjusted R^2. Notice that RSS always decreases as the number of variables in the model increases but $\frac{\text{RSS}}{n-d-1}$ may increase or decrease due to d. Adjusted R^2 statistic pays a price for the inclusion of unnecessary variables in the model.

- Directly estimate the test error using either the validation set or the cross-validation approach. Each of the subset selection procedures returns a sequence of models M_k. We then compute the validation set or the cross-validation error for each M_k and select the k for which the resulting estimated test error is smallest.

Note that the the validation set or the cross-validation approach has an advantage relative to C_p, AIC, BIC and adjusted R^2, in that it provides a direct estimate of the test error and does not require an estimate of the error variance σ^2 and be used in a wider range of model selection tasks.

If n-variable models are roughly equivalent in terms of their test errors, we can select a model using the one-standard-error rule: first calculate the standard error of the estimated test MSE for each model size, and then select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve. The rationale is that if a set of models appear to be more or less equally good, we might choose the simplest model.

5.2 Shrinkage Methods

The subset selection methods use least squares to fit a linear model that contains a subset of the predictors. As an alternative, we can fit a model containing all p predictors using a technique that constrains or regularizes the parameter estimates, or equivalently, that shrinks the parameter estimates towards zero. It may not be immediately obvious why such a constraint should improve the fit, but it turns out that shrinking the parameter estimates can significantly reduce their variance.

5.2.1 Ridge Regression

The ridge regression coefficient estimates $\widehat{\beta}^R$ are the values that minimize

$$\sum_{i=1}^{n}\left(y_i - \widehat{\beta}_0 - \sum_{j=1}^{p}\widehat{\beta}_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\widehat{\beta}_j^2 = \text{RSS} + \lambda\sum_{j=1}^{p}\widehat{\beta}_j^2,$$

where $\lambda \geqslant 0$ is a tuning parameter and has to determine separately.

Ridge regression seeks coefficient estimates that fit the data well by making the RSS small, while $\lambda \sum\limits_{j=1}^{p} \widehat{\beta}_j^2$, called a shrinkage penalty, is small when $\widehat{\beta}_j$'s are close to zero and thus it has the effect of shrinking the estimates.

λ serves to control the relative impact of RSS and shrinkage penalty on the regression coefficient estimates so that selecting a good value for λ is critical. We use cross-validation to find λ.

Note that

$$\sqrt{\sum_{j=1}^{p}\widehat{\beta}_j^2} = \|\widehat{\beta}\|_2,$$

i.e., the l_2 norm of a vector.

Since $X_j\widehat{\beta}^R_{j,\lambda}$ will depend not only on λ, but also on the scaling of X_j and even on the scaling of the other predictors, it is best to apply ridge regression after standardizing the predictors using

$$\widetilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum\limits_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}.$$

5.2.2 Lasso

Ridge regression has one obvious disadvantage: it will include all p predictors in the final model without selection and the lasso overcomes this disadvantage.

The lasso coefficients $\widehat{\beta}^L$ are the values that minimize

$$\sum_{i=1}^{n}\left(y_i - \widehat{\beta}_0 - \sum_{j=1}^{p}\widehat{\beta}_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\widehat{\beta}_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\widehat{\beta}_j|,$$

where $\sum\limits_{j=1}^{p}|\widehat{\beta}_j| = \|\widehat{\beta}\|_1$.

The lasso also shrinks the coefficient estimates towards zero, and the l_1 penalty in lasso has the effect of forcing some of the coefficient estimates to be exactly equal to zero when λ is sufficiently large. Hence, like best subset selection, the lasso performs variable selection. We say the lasso yields sparse model, i.e., the model involves only a subset of the variables.

Selecting a good value for λ is critical and we use cross-validation to find λ.

Note that the lasso and ridge regression coefficient estimates solve the problems

$$\min_{\widehat{\beta}} \sum_{i=1}^{n} \left(y_i - \widehat{\beta}_0 - \sum_{j=1}^{p} \widehat{\beta}_j x_{ij} \right)^2 \quad \text{s.t.} \quad \sum_{j=1}^{p} |\widehat{\beta}_j| \leqslant s$$

and

$$\min_{\widehat{\beta}} \sum_{i=1}^{n} \left(y_i - \widehat{\beta}_0 - \sum_{j=1}^{p} \widehat{\beta}_j x_{ij} \right)^2 \quad \text{s.t.} \quad \sum_{j=1}^{p} \widehat{\beta}_j^2 \leqslant s.$$

Neither ridge regression nor the lasso will universally dominate the other. A technique such as cross-validation can be used in order to determine which approach is better on a particular data set.

5.2.3 Selecting the Tuning Parameter

Cross-validation provides a simple way: we choose a grid of λ and compute the cross-validation error rate for each value of λ, and select the λ for which the cross-validation error is smallest. Finally, the model is re-fit using all of the available observations and the selected value of the tuning parameter.

5.3 Dimension Reduction Methods

Dimension reduction methods is a class of approaches that transform the predictors and then fit a least squares model using the transformed variables.

Let $Z_1, \cdots, Z_M$ represent $M < p$ linear combinations of original p predictors:

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$$

for some constants $\phi_{m1}, \cdots, \phi_{mp}$. We can then fit the linear regression model

$$y_i = \sum_{m=1}^{M} \theta_m z_{im} + \varepsilon_i, i = 1, \cdots, n$$

using ordinary least squares.

6 Beyond Linearity

The linear models have some advantages: (1) simple to describe and implement; (2) easy interpretation and can make inferences; (3) low bias if linearity assumption hold; (4) low variance when the model is restrictive ($p \ll n$). However, they have high variance when the model is not restrictive ($p \gg n$) and high bias when linearity assumption does not hold.

Since the truth may not be linear, we relax the linearity assumption and thus we will be able to improve the bias.

6.1 Polynomial Regression

We assume a function

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d + \varepsilon_i,$$

where ε_i is the error term.

We can calculate the pointwise 95% CI with $\mathrm{Var}\left[\widehat{f}(x_0)\right]$ for each fitted function values at any value x_0. Since $\widehat{f}(x_0)$ is a linear function of the $\widehat{\beta}_j$'s, we can get a simple expression for pointwise variances $\mathrm{Var}\left[\widehat{f}(x_0)\right]$ at any value x_0 using covariance matrix of $\widehat{\beta}_j$'s.

We can also write the function as

$$P(y_i|x_i) = \frac{e^{\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d}}{1 + e^{\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d}}.$$

To get confidence intervals, we compute upper and lower bounds on the logit scale, and invert to get on probability scale.

We either fix the degree d at some reasonably low value or use cross-validation to choose d.

6.2 Step Functions

We create cut points at $c_1, \cdots, c_K$ in the range of X and then construct $K+1$ new variables as

$$\begin{aligned} C_0(X) &= I(X < c_1), \\ C_1(X) &= I(c_1 \leqslant X < c_2), \\ &\vdots \\ C_{K-1}(X) &= I(c_{K-1} \leqslant X < c_K), \\ C_K(X) &= I(c_K \leqslant X), \end{aligned}$$

where $I(\cdot)$ is an indicator function. Then we use least squares to fit a linear model (piecewise-constant regression model)

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \cdots + \beta_K C_K(x_i) + \varepsilon_i.$$

When $X < c_1$, all of the predictors are zero and β_0 can be interpreted as the mean value of Y for $X < c_1$. Besides, the model predicts a response of $\beta_0 + \beta_j$ for $c_j \leqslant X < c_{j+1}$, so β_j represents the average increase in the response for X in $c_j \leqslant X < c_{j+1}$ relative to $X < c_1$.

However, unless there are natural breakpoints in the predictors, piecewise constant functions can miss the action.

6.3 Basis Functions

Suppose $b_1(X), \cdots, b_K(X)$ is a family of functions or transformations that can be applied to a variable X, we fit the model

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \cdots + \beta_K b_K(x_i) + \varepsilon_i.$$

Note that the basis functions are fixed and known. Polynomial and piecewise-constant regression models are in fact special cases of a basis function approach ($b_j(x_i) = x_i^j$ and $b_j(x_i) = I(c_j \leqslant x_i < c_{j+1})$, respectively).

6.4 Regression Splines

6.4.1 Piecewise Polynomials

Instead of a single polynomial in X over its whole domain, we can rather use different polynomials in regions defined by knots. For instance,

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \varepsilon, & \text{if } x_i < c \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \varepsilon, & \text{if } x_i \geqslant c \end{cases}.$$

6.4.2 Constraints and Splines

We can fit a piecewise polynomial under the constraint that the fitted curve must be continuous or even more smooth (continuity, continuity of the first derivative, and continuity of the second derivative).

6.4.3 Linear Splines

A linear spline with knots at $\xi_k, k = 1, \cdots, K$ is a piecewise linear polynomial continuous at each knot. We can represent the model as

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \cdots + \beta_{K+1} b_{K+1}(x_i) + \varepsilon_i,$$

where b_k's are basis functions: $b_1(x_i) = x_i, b_{k+1}(x_i) = (x_i - \xi_k)_+, k = 1, \cdots, K$ and

$$(x_i - \xi_k)_+ = \begin{cases} x_i - \xi_k, & \text{if } x_i > \xi_k \\ 0, & \text{otherwise} \end{cases}.$$

6.4.4 Cubic Splines

A cubic spline with knots at $\xi_k, k = 1, \cdots, K$ is a piecewise cubic polynomial with continuous derivatives up to order 2 at each knot. We can represent the model as

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \cdots + \beta_{K+3} b_{K+3}(x_i) + \varepsilon_i,$$

where $b_1(x_i) = x_i, b_2(x_i) = x_i^2, b_3(x_i) = x_i^3, b_{k+3}(x_i) = (x_i - \xi_k)_+^3, k = 1, \cdots, K$ and

$$(x_i - \xi_k)_+^3 = \begin{cases} (x_i - \xi_k)^3, & \text{if } x_i > \xi_k \\ 0, & \text{otherwise} \end{cases}.$$

In general, *df* of cubic splines with K knots is $K + 4 = 4 \times (K + 1) - 3K$.

6.4.5 Natural Cubic Splines

A natural spline is a regression spline with additional boundary constraints: the function is required to be linear at the boundary (in the region where X is smaller than the smallest knot or larger than the largest knot) to produce more stable estimates at the boundaries. *df* of natural cubic splines with K knots is K.

6.4.6 Choosing the Number and Locations of the Knots

One option is to place more knots in places where the function might vary most rapidly and to place fewer knots where it seems more stable. In practice, it is common to place knots in a uniform fashion: specify the desired degrees of freedom and place the corresponding number of knots at uniform quantiles of the data. In R, we use `bs(x, ...)` for any degree splines and `ns(x, ...)` for natural cubic splines in package `splines`.

We then use cross validation to choose K. We remove a portion of the data, fit a spline with a certain number of knots to the remaining data, use the spline to make predictions for the held-out portion, repeat this process multiple times until each observation has been left out once, and compute the overall cross-validated RSS. We then repeat above procedure for different numbers of knots K, and choose the K that gives the smallest RSS.

6.4.7 Comparison to Polynomial Regression

Regression splines often give superior results to polynomial regression. This is because splines introduce flexibility by increasing the number of knots but keeping the degree fixed and produce more stable estimates; while polynomials must use a high degree to produce flexible fits and extra flexibility in the polynomial produces undesirable results at the boundaries.

6.5 Smoothing Splines

An approach to get a smooth function that solve

$$\min_{g \in \mathcal{S}} \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int g''(t)^2 \mathrm{d}t,$$

where λ is a nonnegative tuning parameter and called roughness penalty, $g(x)$ is known as a smoothing spline. The first term is RSS and tries to make $g(x)$ match the data at each x_i and the second term is a roughness penalty and controls how wiggly $g(x)$ is. The smaller λ, the more wiggly the function, eventually interpolating each y_i when $\lambda = 0$. As $\lambda \to \infty, g(x)$ becomes linear.

A smoothing spline is simply a natural cubic spline with knots at every unique value of x_i but it is a shrunken version of such a natural cubic spline, where the value of the tuning parameter λ controls the level of shrinkage.

6.5.1 Choosing the Smoothing Parameter λ

Smoothing splines avoid the knot-selection issue, leaving a single λ to be chosen. We use effective degrees of freedom df_λ to measure the flexibility of the smoothing spline:

$$df_\lambda = \sum_{i=1}^{n} \{\mathbf{S}_\lambda\}_{ii},$$

and

$$\widehat{\mathbf{g}}_\lambda = \mathbf{S}_\lambda \mathbf{y},$$

where $\widehat{\mathbf{g}}$ is the solution of smooth function for a particular choice of λ, that is, it is a n-vector containing the fitted values of the smoothing spline at the training points $x_1, \cdots, x_n, \mathbf{S}_\lambda$ is an $n \times n$ matrix determined by x_i and λ. In R, `smooth.spline()` fits a smoothing spline. Note that we can specify *df* rather than λ : `smooth.spline(age, wage, df = 10)`.

We choose λ using cross validation. The leave-one-out cross validation (LOOCV) error is

$$\text{RSS}_{\text{CV}}(\lambda) = \sum_{i=1}^{n} \left(y_i - \widehat{g}_\lambda^{(-i)}(x_i) \right)^2 = \sum_{i=1}^{n} \left[\frac{y_i - \widehat{g}_\lambda(x_i)}{1 - \{\mathbf{S}_\lambda\}_{ii}} \right]^2 .$$

We calculate $\text{RSS}_{\text{CV}}(\lambda)$ for a range of λ and pick the value that minimizes $\text{RSS}_{\text{CV}}(\lambda)$.

6.6 Local Regression

Local regression involves computing the fit at a target point x_0 using only the nearby training observations. With a sliding weight function, we fit separate linear fits over the range of x by weighted least squares. In R, we use `loess()`.

7 Tree-Based Methods

Since the set of splitting rules used to segment the predictor space can be summarized in a tree, we call those approaches as decision-tree methods. Tree-based methods are simple and useful for interpretation. However, they typically are not competitive with the best supervised learning methods in terms of prediction accuracy. Hence, we also discuss bagging, random forests and boosting. These methods grow multiple tress which are then combined to yield a single consensus prediction. Combing a large number of trees can often result in dramatic improvements in prediction accuracy, at the expense of some loss interpretation.

7.1 The Basics of Decision Trees

7.1.1 Regression Trees

Decision trees are typically drawn upside down, in the sense that the terminal nodes or leaves are at the bottom of the tree. The points along the tree where the predictor space is split are internal nodes.

We divide the predictor space, i.e., the set of possible values for $X_1, \cdots$ into J distinct and non-overlapping regions, $R_1, \cdots, R_J$, For every observation that falls into R_j, we make the same prediction, which is the mean of response values for the training observations in R_j.

In theory, the regions could have any shape. But we choose to divide the space into high-dimensional rectangles or boxes for simplicity and ease of interpretation of the resulting predictive model. The goal is to find $R_1, \cdots, R_J$ that minimize the RSS given by

$$\sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \widehat{y}_{R_j}\right)^2,$$

where $\widehat{y}_{R_j}$ is the mean response for the training observations within the jth box.

It is computationally infeasible to consider every possible partition of the feature space into J boxes and thus we take a top-down, greedy approach that is known as recursive binary splitting:

- Top-Down: it begins at the top of the tree and then successively splits the predictor space; each split is indicated via two new branches further down on the tree.
- Greedy: at each step of the tree-building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.

We first select the predictor X_i and the cut point s s.t. splitting the predictor space into two regions $\{X|X_j < s\}$ and $\{X|X_j \geqslant s\}$ leads to the greatest possible reduction in RSS given by

$$\sum_{i: x_i \in R_1(j,s)} \left(y_i - \widehat{y}_{R_1}\right)^2 + \sum_{i: x_i \in R_2(j,s)} \left(y_i - \widehat{y}_{R_2}\right)^2.$$

Then we repeat the precess, looking for the best predictor and best cut point in order to split the data further so as to minimize RSS within each of the resulting regions. However, instead of splitting the entire predictor space, we split one of the two previously identified regions and have three regions. Again, we look to split one of these three regions to minimize RSS and the process continues until a stopping criterion is reached.

We predict the response for a given test observation using the mean of the training observations in the region to which that observation belongs.

The process is likely to overfit the data. A smaller tree with fewer splits (regions) might lead to lower variance and better interpretation at the cost of a little bias. One possible alternative to the process is to grow the tree only so long as the decrease in RSS due to each split exceeds some (high) threshold. This strategy will result in smaller trees, but is too short-sighted: a seemingly worthless split early on in the tree might be followed by a very good split.

Another strategy is to grow a very large tree T_0, and then prune it back in order to obtain a subtree. Cost complexity pruning (weakest link pruning) is used. We consider a sequence of trees indexed by a nonnegative tuning parameter α. For each α there corresponds a subtree $T \subset T_0$ s.t.

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha|T|$$

is as small as possible. $|T|$ indicates the number of terminal nodes of the tree T, R_m is the rectangle corresponding to the mth terminal node, $\widehat{y}_{R_m}$ is the mean of the training observations in R_m, and the tuning parameter α controls a trade-off between the subtree's complexity and its fit to the training data.

Algorithm 6: Regression Tree

```
use recursive binary splitting to grow a large tree on the training data, stopping only
  when each terminal node has fewer than some minimum number of observations;
apply cost complexity pruning to the large tree in order to obtain a sequence of best
  subtrees, as a function of $\alpha$;
divide the training observations into $K$ folds;
for $k = 1, \cdots, K$ do
    repeat Step 1 and 2 on all but the $k$th fold of the training data;
    evaluate the mean squared prediction error on the data in the left-out $k$th fold as a
      function of $\alpha$, average the results for each value of $\alpha$, and pick $\alpha$ to minimize the
      average error;
end
return the subtree from Step 2 that corresponds to the chosen value of $\alpha$.
```

7.1.2 Classification Trees

For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs. We use recursive binary splitting to grow a classification tree and may use classification error rate (rather than

RSS) as a criterion:

$$E = 1 - \max_k (\widehat{p}_{mk}),$$

where $\widehat{p}_{mk}$ is the proportion of training observations in the mth region that are from the kth class. But E is not sufficiently sensitive for tree-growing and in practice we prefer Gini index or cross-entropy.

The Gini index (purity index) is

$$G = \sum_{k=1}^{K} \widehat{p}_{mk}(1 - \widehat{p}_{mk}).$$

Gini index is a measure of total variance across K classes and takes on a small value if all of the $\widehat{p}_{mk}$'s are all near zero or one. Hence Gini index is referred to as a measure of node purity - a small value indicates that a node contains predominantly observations from a single class.

The entropy (deviance) is

$$D = -\sum_{k=1}^{K} \widehat{p}_{mk} \ln \widehat{p}_{mk}.$$

Since $0 \leqslant \widehat{p}_{mk} \leqslant 1$, it follows that $0 \leqslant -\widehat{p}_{mk} \ln \widehat{p}_{mk}$. The entropy will take on a value near zero if the $\widehat{p}_{mk}$'s are all near zero or one. Hence like Gini index, the entropy will take on a small value if the mth node is pure. In fact, the Gini index and the entropy are very similar numerically.

7.1.3 Advantages and Disadvantages of Trees

- Trees are very easy to explain to people (even easier to explain than linear regression).
- Decision trees more closely mirror human decision-making than do the regression and classification methods.
- Trees can be displayed graphically and are easily interpreted even by a non-expert.
- Trees can easily handle qualitative predictors without the need to create dummy variables.
- Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification methods. However, by aggregating many decision trees, the predictive performance of trees can be substantially improved.

7.2 Bagging

Bootstrap aggregation or bagging is a general-purpose procedure for reducing the variance of a statistical learning method.

Recall that given a set of n independent observations $Z_1, \cdots, Z_n$, with variance σ^2,

$$\text{Var}\left[\overline{Z}\right] = \frac{\sigma^2}{n}.$$

Hence, averaging a set of observations reduces variance but it is not practical since we generally do not have access to multiple training sets. Instead, we can bootstrap, by taking repeated samples from the training data set.

We generate B different bootstrapped training data sets, train our method on the bth bootstrapped training set in order to get $\widehat{f}^{*b}(x)$, the prediction at a point x, and average all the predictions to obtain

$$\widehat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \widehat{f}^{*b}(x),$$

which is called bagging.

Bagging can be applied to regression and classification trees. For classification problems, we display the simplest approaches: for a given test observation, we record the class predicted by each of the B trees, and take a majority vote - the overall prediction is the most commonly occurring class among the B predictions.

7.2.1 Out-of-Bag Error Estimation

We can show that on average, each bagged tree makes use of around two-thirds of the observations.The remaining one-third of the observations not used to fit a given bagged tree are referred to as the out-of-bag (OOB) observations.

We can predict the response for the ith observation using each of the trees in which that observation is OOB. This will yield around $\frac{B}{3}$ predictions for the ith observation. To obtain a single prediction for the ith observation, we can average these predicted responses or take a majority vote and then we can compute the OOB MSE or classification error. The resulting OOB error is a valid estimate of the test error for the bagged model, since the response for each observation is predicted using only the trees that were not fit using that observation. This estimate is essentially the LOOCV error for bagging if B is large.

7.3 Random Forests

Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the tress, which reduces the variance when we average the trees.

As in bagging, we build a number of decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random selection of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. A fresh selection of m predictors is taken at each split, and typically we choose $m \approx \sqrt{p}$.

7.4 Boosting

Recall that bagging involves creating multiple copies of the original training data set using the bootstrap, fitting a separate decision tree to each copy, and then combining all of the trees in order to create a single predictive model. Notably, each tree is built on a bootstrap data set, independent of the other trees.

Boosting works in a similar way, except that the trees are grown sequentially: each tree is grown using information from previously grown trees. Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set.

Unlike fitting a single large decision tree to the data, which amounts to fitting the data hard and potentially overfitting, the boosting approach instead learns slowly. Given the current model, we fit a decision tree to the residuals from the model and add this new decision tree into the fitted function in order to update the residuals.

Algorithm 7: Boosting for Regression Trees

```
set $\widehat{f}(x) = 0$ and $r_i = y_i$ for all $i$ in the training set;
for $b = 1, \cdots, B$ do
    fit a tree $\widehat{f}^b$ with $d$ splits ($d+1$ terminal nodes) to the training data $(X, r)$;
    update $\widehat{f}$ by adding in a shrunken version of the new tree: $\widehat{f}(x) \leftarrow \widehat{f}(x) + \lambda \widehat{f}^b(x)$;
    update the residuals: $r_i \leftarrow r_i - \lambda \widehat{f}^b(x_i)$;
end
Output: $\widehat{f}(x) = \sum_{b=1}^{B} \lambda \widehat{f}^b(x)$
```

Each of these trees can be rather small, with just a few terminal nodes, determined by the parameter d in the algorithm. By fitting small trees to the residuals, we slowly improve $\widehat{f}$ in areas where it does not perform well. The shrinkage parameter λ slows the process down even further, allowing more and different shaped trees to attack the residuals.

Boosting for classification is similar in spirit to boosting for regression. The R package `gbm` (gradient boosted models) handles a variety of regression and classification problems.

7.4.1 Tuning Parameters

- The number of trees B : Unlike bagging and random forests, boosting can overfit if B is too large, although this overfitting tends to occur slowly if at all. We use cross-validation to select B.

- The shrinkage parameter $\lambda > 0$: It controls the rate at which boosting learns and typical values are 0.01 or 0.001, and the right choice can depend on the problem. Very small λ can require using a very large value of B in order to achieve good performance.

- The number of splits d in each tree: It controls the complexity of the boosted ensemble. Often $d = 1$ works well, in which case each tree is a stump, consisting of a single split and resulting in an additive model. More generally d is the interaction depth, and controls the interaction order of the boosted model, since d splits can involve at most d variables.

7.5 Variable Importance

For bagged/random forests regression trees, we record the total amount that the RSS is decreased due to splits over a given predictor, averaged over all B trees. A large value indicates an important predictor. Similarly, for bagged/random forests classification trees, we add up the total amount that the Gini index is decreased by splits over a given predictor, averaged over all B trees.

7.6 Summary

Decision trees are simple and interpretable models for regression and classification, but they are often not competitive with other methods in terms of prediction accuracy. Bagging, random forests and boosting are good methods for improving the prediction accuracy of trees. They work by growing many trees on the training data and then combing the predictions of the resulting ensemble of trees. Random forests and boosting are among the state-of-the-art methods for supervised learning, but their results can be difficult to interpret.