Applied Multivariate and Longitudinal Data Analysis

Discriminant analysis and classification

Ana-Maria Staicu
SAS Hall 5220; 919-515-0644; astaicu@ncsu.edu

Consider the examples:

- An online banking service collects the IP address, past transaction history, and so forth of users and based on the acquired information must be able to determine whether or not a transaction being performed on the site is fraudulent or not.
- An emergency room system collects standard medical information on the admitted patients, such as blood pressure, heart rate, temperature, age etc and based on this it must be able to assign patients to one of three categories: "extremely urgent care", "very urgent" and "urgent care".

The problem of separating two or more groups is sometimes called discrimination or "supervised" classification.

Discrimination: finding the features that *separate* known groups in a multivariate sample.

Classification: developing a rule to *allocate* a new object into one of a number of known groups.

Wine data contains information on three varieties of wine cultivars ('wines' in the data folder). For each of the 178 wines examined information on 14 variables is recorded. Here are the 14 variables.

Variable name	Description
Class	Which of three cultivars of wine grapes
Alcohol	Alcohol content
Malic	Malic acid: provides a sour taste
Ash	Ash content
Alcal	Alcalinity of ash
Mg	Magnesium content
Phenol	Total phenols: compounds found in drugs and plastics
Flav	Flavanoids: compounds found widely in plants
Nonf	Nonflavanoid phenols
Proan	Proanthocyanins: tannins which affect color and aging
Color	
Hue	
Abs	Ratio of light absorption at two different frequencies
Proline	Proline, an amino acid

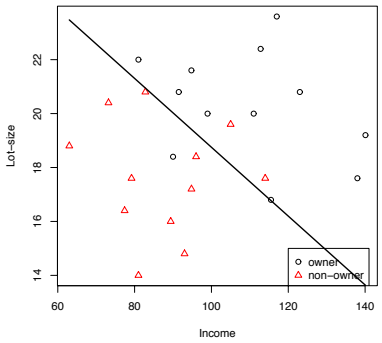
Question: Identify the distinguishing characteristics between the different groups.

A classification rule is based on the features that separate the groups, so the goals overlap. Making mistakes is inevitable: 1) try to make as few mistakes as possible; and 2) quantify the cost of misclassification.

In this chapter we will discuss discriminant and classification analysis for two groups (logistic regression, linear discriminant analysis, quadratic discriminant analysis, Fisher's discriminant analysis) for more than two groups and possibly on modern classification methods (k-nearest neighbor, classification and regression tree, support vector machines).
Both discrimination and classification depend on multivariate observation $\mathbf{X} \in \mathbb{R}^p$.

1 Logistic Regression

A. Two groups/classes



Riding lawnmower example.
The data includes information on 24 families, of various income and lot-size, out of which 12 own a riding lawnmower and 12 are currently non-owners. The data is in the file "T11-1.txt".

Goal: Classify families based on their income and lot size as prospective owners or non-owners.

Some initial observations: Looking at the plot we see that

- riding mowers owners tend to have higher income and bigger lot-size
- there is some overlap between two groups
- income seems to be a better discriminator than lot-size

Setup: For each family i in the observed sample, we have

- $Y_i = 1$ if the family is a riding lawnmower owner and $Y_i = 0$ if non-owner
- $\mathbf{X}_i = (\text{Income}_i, \text{Lot-size}_i)^\top$
- Goal: Develop a *classification rule* that tells us the class membership of a new family based of their income level and lot-size.

[*Logistic regression*:] Directly model the probability that the observation belongs to class 1, given the predictor value for that observation. *Logistic regression* models the probability of the class membership (Y) given the predictor ($\mathbf{X}$) using the logistic function (eg probability of the family being a lawnmower owner given the income and lot size).

$$\begin{aligned} p(\mathbf{X}) &:= \Pr(Y = 1|\mathbf{X}) \\ &= \frac{\exp(\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p)}{1 + \exp(\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p)} = \frac{\exp(\beta_0 + \beta^T \mathbf{X})}{1 + \exp(\beta_0 + \beta^T \mathbf{X})} \end{aligned}$$

for some coefficients $\beta_0, \beta_1, \ldots, \beta_p$.

From how $p(\mathbf{X})$ is defined, it follows that $p(\mathbf{X}) \in [0,1]$. The idea is to predict category $Y=1$ is $p(\mathbf{X})$ is large (ie closer to 1) and predict category $Y=0$ if $p(\mathbf{X})$ is closer to 0.
Work with your partner to simplify the following log ratio

$$\log \frac{p(\mathbf{X})}{1-p(\mathbf{X})} = \log \frac{Pr(Y=1|\mathbf{X})}{Pr(Y=0|\mathbf{X})}$$

This quantity is called *log odds ratio* and can take any value in $\mathbb{R}^1$; we see that the log-odds ratio is linear in $\mathbf{X}$. Values that are close to $-\infty$ indicate very low probability while values close to ∞ indicate very high probability.

Interpretation: a covariate X_j for which the corresponding coefficient β_j is positive, is associated with an increase in the log-odds ratio, or equivalently an increase in $p(\mathbf{X})$. In contrast, covariate X_j for which the corresponding coefficient β_j is negative, is associated with a decrease in the log-odds ratio, or equivalently decrease in $p(\mathbf{X})$.

1.1 Estimating the model parameters

Model assumption: Conditional on $\mathbf{X}=\mathbf{x}$, we have a logistic regression model

$$Y|\mathbf{X}=\mathbf{x} \sim Bernoulli(p(\mathbf{x})), \qquad p(\mathbf{x}) = \frac{\exp(\beta_0+\beta_1 x_1+\cdots+\beta_p x_p)}{1+\exp(\beta_0+\beta_1 x_1+\cdots+\beta_p x_p)}$$

Remark: the definition does not require any assumptions for the distribution of covariates (such as multivariate normality).

The model parameters $\beta_0, \beta_1, \ldots, \beta_p$ are directly estimated by maximizing the likelihood function. Specifically, the likelihood function corresponding to a sample $\{y_i, \mathbf{x}_i : \ i=1,\ldots,n\}$ is

$$L(\beta_0,\beta_1,\ldots,\beta_p) = \prod_{i=1}^{n} p(\mathbf{x}_i)^{y_i}\{1-p(\mathbf{x}_i)\}^{1-y_i}; \tag{1}$$

the maximizer does not have an analytical expression. Instead it is obtained numerically by *iteratively reweighted least squares*. It follows that $\mathrm{P}(Y=1|\mathbf{X})$ can be estimated by

$$\frac{\exp\left(\hat{\beta}_0+\hat{\beta}_1 X_1+\cdots+\hat{\beta}_p X_p\right)}{1+\exp\left(\hat{\beta}_0+\hat{\beta}_1 X_1+\cdots+\hat{\beta}_p X_p\right)}.$$

Classifier: We can predict the class membership (Y) for a new data object $\mathbf{x}_{new}$ using the estimated probability that $Y=1$ conditional on $\mathbf{X}=\mathbf{x}$ or equivalently using $\hat{\beta}_0+\hat{\beta}^T\mathbf{x}_{new}$ as follows:

- $Y_{pred}=1$ if $\widehat{P}(Y=1|\mathbf{X}=\mathbf{x}_{new}) \geq 0.5$ or equivalently if $\hat{\beta}_0+\hat{\beta}^T\mathbf{x}_{new} \geq 0$
- $Y_{pred}=0$ if $\widehat{P}(Y=1|\mathbf{X}=\mathbf{x}_{new}) < 0.5$or equivalently if $\hat{\beta}_0+\hat{\beta}^T\mathbf{x}_{new} < 0$.

2 Assessing predictive ability of classifier: confusion matrix and ROC curve

In practice, a binary classifier such as this one can make two types of errors: it can incorrectly assign a family who owns a riding mower to be in the "non-owner" category or it can incorrectly assign a non-owner to be in the "owner" category. A *confusion matrix* is a table that displays this information; and thus is used to describe the performance of a classifier.

	True	
Classify as:	owner	non-owner
owner	11	2
non-owner	1	10

The confusion matrix compares the classifier predictions with the true class membership. Elements on the diagonal of the matrix represent individuals whose default statuses were correctly predicted, while off-diagonal elements represent individuals that were misclassified.

Consider the classiffier that has the confusion matrix shown above. The *accuracy* of a classifier is defined as the proportion of correct identification of positive and negatives, in our case $(11 + 10)/(12+12) = 87.5\%$. Thus while the overall missclassification rate is relatively low $(1+2)/(12+12) = 12.5\%\%$, the error rate among the non-owner families is quite high $(2/12 = 16.67\%)$.

Class-specific performance is also important in medicine and biology, where the terms *sensitivity* and *specificity* characterize the performance of sensitivity specificity a classifier or screening test. In this case the *sensitivity* (true positive rate, TPR=TP/P) is the percentage of true owners that are identified, $11/12 = 91.67\%$ in this case. The specificity (true negative rate, TNR=TN/N) is the percentage of non-owners that are correctly identified, here $10/12 = 83.33\%$. One minus specificity is the proportion of negatives incorrectly identified, also known as false positive rate (FPR=FP/N); here FPR $=2/12 = 16.67\%$. In terms of hypothesis testing: FPR or one minus speficifity corresponds to the Type I error, the false negative rate (FNR=1-TPR) to the Type II error, and TPR to the power.

	True	
Classify as:	Positive	Negative
$+$	True Positive (TP)	False Positive(FP)
$-$	False Negative (FN)	True Negative
Total	Positives (P)	Negatives (N)

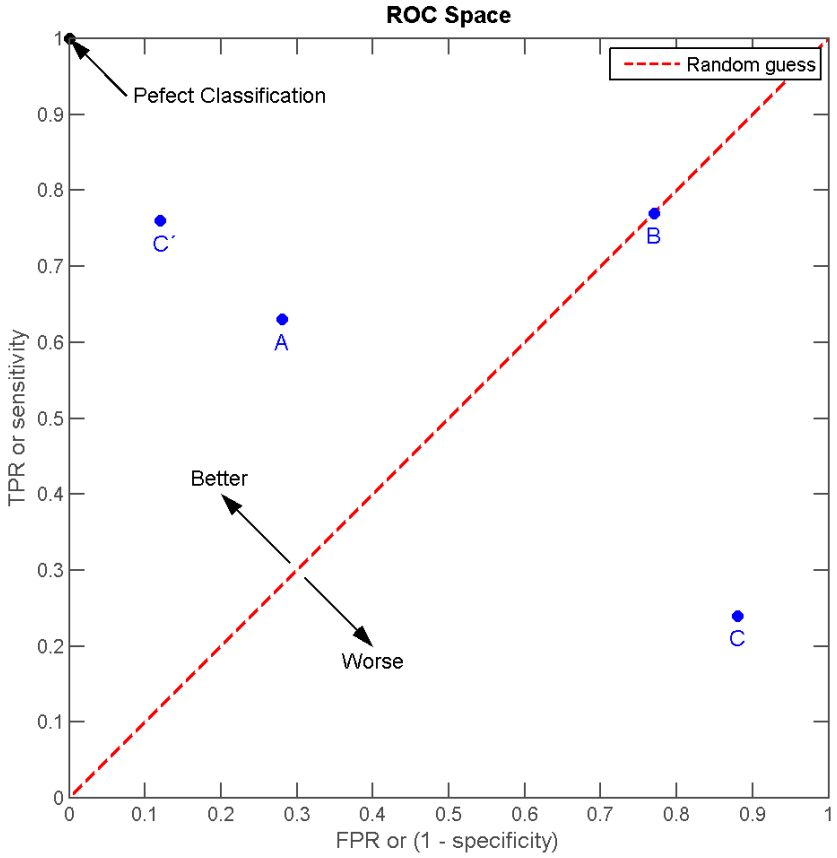
The confusion matrix compares the classifier predictions with the true class membership. Elements on the diagonal of the matrix represent individuals whose default statuses were correctly predicted, while off-diagonal elements represent individuals that were misclassified.

Assessing the accuracy of a classifier on the data that was used to determine the classifier, would yield to an overly optimistic misclassification error (lower misclassification error than true). In general it is recommended to split the data into a training set and a test set; use the training set to construct the classifier, and then use the test set to evaluate its accuracy.

The Receiver Operating Characteristic (in short ROC) curve is a popular graphic for simultaneously displaying the two types of errors for all possible thresholds.

- It plots sensitivity (y-axis) versus 1 - specificity (x-axis).

The larger the area under the ROC curve (AUC) the better. Out of multiple classifiers, select the one with the largest AUC.



B. More than two groups/classes

Consider that the response can take multiple categories (eg. Y=1,2,3 in the wine example). Let k index the classes $\{1, 2, \ldots, K\}$, we posit the model

$$Pr(Y = k|\mathbf{X} = \mathbf{x}) = \frac{\exp(\beta_{k0} + \boldsymbol{\beta}_k^{\mathsf{T}}\mathbf{x})}{\sum_{\ell=1}^{K} \exp(\beta_{\ell 0} + \boldsymbol{\beta}_\ell^{\mathsf{T}}\mathbf{x})},$$

for unknown intercepts β_{k0} and unknown regression coefficients $\boldsymbol{\beta}_k$.

Since $Pr(Y|\mathbf{X})$ completely specifies the conditional distribution, the multinomial distribution is appropriate. The model is fitted through maximum likelihood estimation, in a similar way to the case of two classes.

Class prediction: Assign a new observation to the *most likely class* given its predictor value. More formally, classify a new observation with the predictor value $\mathbf{x}_{new}$, as the class that maximizes $\{\widehat{Pr}(Y = \ell|\mathbf{X} = \mathbf{x}_{new}) : \ell = 1, \ldots, K\}$, where the "hat" notation means that we are using the previously described model with the unknown parameters β's substituted by their maximum likelihood estimates:

$$Y_{pred} = k \quad \text{if} \quad \widehat{P}(Y = k|\mathbf{X} = \mathbf{x}_{new}) = \max_{\ell} \widehat{P}(Y = \ell|\mathbf{X} = \mathbf{x}_{new}).$$

The last part is essentially equivalent to:

$$Y_{pred} = k \quad \text{if} \quad \hat{\beta}_{k0} + \hat{\boldsymbol{\beta}}_k^{\mathsf{T}}\mathbf{x}_{new} = \max_{\ell=1,\ldots,K} \{\hat{\beta}_{\ell 0} + \hat{\boldsymbol{\beta}}_\ell^{\mathsf{T}}\mathbf{x}_{new}\}.$$

Final remarks on the logistic regression model:

- When the classes are well-separated, the parameter estimates for the logistic regression model are surprisingly unstable.
- If the sample size n is small and the distribution of the predictors $\mathbf{X}$ is approximately normal in each of the classes, again the estimation in the logistic regression model is unstable.

3 Discriminant Analysis

Intuition: Logistic regression involves directly , $Pr(Y = k|\mathbf{X} = \mathbf{x})$ using the logistic function. We now consider an alternative and less direct approach to estimating these probabilities using the logic:

$$Pr(Y = k|\mathbf{X} = \mathbf{x}) = \frac{Pr(Y = k)P(\mathbf{x}|Y = k)}{P(\mathbf{x})}.$$

This entails

- describing the *prior class probabilities* - the overall probability that a random observation is in the kth class, $\pi_k = Pr(Y = k)$,
- modeling the distribution of the predictors $\mathbf{X}$ for an observation that comes from the kth class; $f_k(\mathbf{x})$ is the density of the $\mathbf{X}$ in the kth group/class;

application of the Bayes theorem provides the *posterior* probability that an observation for which $\mathbf{X} = \mathbf{x}$ belongs to class k:

$$Pr(Y = k|\mathbf{x}) = \frac{\pi_k f_k(\mathbf{x})}{\sum_{k=1}^{K} \pi_k f_k(\mathbf{x})}. \tag{2}$$

When the underlying distributions are normal with same covariance, it turns out that the final model is very similar to the logistic regression. However, unlike the logistic regression model which shows instability in the case of perfect separation or small sample size, linear/quadratic discriminant analysis does not suffer from this problem.

Class prediction: The classifier described by (2) that has the lowest possible (total) error rate out of all classifiers is *Bayes classifier*, which will classify an observation to the most likely class, given its predictor values, that is the class for which $Pr(Y = k|\mathbf{x})$ is the largest.

Of course in practice we do not know the prior class probabilities π_k, nor the distribution of the predictors in class k, for $k = 1, \ldots, L$; so we need to estimate them.

- Estimate π_k by $\hat{\pi}_k = n_k/n$ which is the fraction of the observations belonging to class k. Here n_k is the number of observations in class k and n is the total number of observations.
- Find an approach to estimate $f_k(\mathbf{x})$; this allows to develop a classifier that approximates the Bayes classifier.

LDA assumes that the distribution of the predictors in each class is multivariate normal $N_p(\boldsymbol{\mu}_k, \Sigma)$ with class-specific mean vectors $\boldsymbol{\mu}_k$ and same covariance across the classes $\Sigma_k = \Sigma$.

3.1 LDA for $p = 2$

Consider $p = 2$ and **assume** further that the common covariance matrix is diagonal, $\Sigma = \mathrm{diag}(\sigma_1^2, \sigma_2^2)$. Recall the density of a bivariate normal distribution:

$$f_k(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2} \exp\left\{ -\frac{(x_1 - \mu_{k1})^2}{2\sigma_1^2} - \frac{(x_2 - \mu_{k2})^2}{2\sigma_2^2} \right\}$$

where $\boldsymbol{\mu}_k = (\mu_{k1}, \mu_{k2})^T$ is the class k mean vector.

$f_k(\mathbf{x})$ is the multivariate density for the

In class work with partner:

- Plug in the density above to calculate the posterior probability that an observation with $\mathbf{X} = \mathbf{x}$ is in class k:

- Based on this can you find a simpler way to describe when an observation with $\mathbf{X} = \mathbf{x}$ is in class k. For illustration, consider further that there are only $K = 2$ classes:

The function

$$\delta_k(\mathbf{x}) = x_1 \frac{\mu_{k1}}{\sigma_1^2} + x_2 \frac{\mu_{k2}}{\sigma_2^2} - \frac{\mu_{k1}^2}{2\sigma_1^2} - \frac{\mu_{k2}^2}{2\sigma_1^2} + \log \pi_k$$

is called *discriminant function*. The word *linear* in the name of the method (linear discriminant analysis) comes from the fact that the discriminant function $\delta_k(\mathbf{x})$ is linear in in $\mathbf{x}$.

In practice this still requires estimation of the group mean vectors $\boldsymbol{\mu}_k$ and of the shared variance matrices Σ_k. Denote by $\widehat{\boldsymbol{\mu}}_k$ and $\widehat{\Sigma}$ their corresponding estimates; $\widehat{\boldsymbol{\mu}}$ is the group mean vector and $\widehat{\Sigma}$ is the pooled covariances, which for illustration is assumed diagonal here).

The Bayesian decision boundaries for two different classes k and l are defined by the points for which $\delta_k(\mathbf{x}) = \delta_l(\mathbf{x})$; formally $\{\mathbf{x} \in R^2 : \delta_k(\mathbf{x}) = \delta_l(\mathbf{x})\}$.

3.2 LDA for $p \geq 2$

Assume that $f_k(\mathbf{x})$ is the multivariate normal density $N_p(\boldsymbol{\mu}_k, \Sigma)$, which recall, has the expression

$$f_k(\mathbf{x}) = (2\pi)^{-p/2} |\Sigma|^{-1/2} \exp\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)\}.$$

It implies that the Bayes classifier, assigns an observation with $\mathbf{x}$ to class k for which the value of the discrimination function

$$\delta_k(\mathbf{x}) = \mathbf{x}^T\Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k\Sigma^{-2}\boldsymbol{\mu}_k + \log\pi_k$$

is the largest.

Also the decision boundaries for two different classes k and l are given by

$$\{\mathbf{x} \in R^p : \mathbf{x}^T\Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k\Sigma^{-2}\boldsymbol{\mu}_k + \log\pi_k = \mathbf{x}^T\Sigma^{-1}\boldsymbol{\mu}_l - \frac{1}{2}\boldsymbol{\mu}_l\Sigma^{-2}\boldsymbol{\mu}_l + \log\pi_l\};$$

notice linear bundaries (again related to the "linear" part of the method name).

In practice, like for the bivariate case, the class specific mean vectors are estimated by the class means, $\widehat{\boldsymbol{\mu}}_k = \frac{1}{n_k}\sum_{i:\mathbf{x}_i \in \text{ class } k}\mathbf{x}_i$ and $\widehat{\Sigma}$ is estimated by the pooled covariance; see previous notes on how to calculate it.

Linear discriminant analysis is very popular for more than two classes. When the multivariate normality assumption holds and furthermore the class-specific covariances are the same across $k = 1, \ldots, K$ then LDA works really well. However when the common variance assumption is far from valid, then LDA can be improved.

4 Quadratic discriminant analysis (QDA)

When the normality assumption holds, but the commn variance assumption does not hold, then another approach, quadratic discriminant analysis offers an improved classifiers.

In class: What does the word *quadratic* seem to imply in terms of the discriminant function? or boundaries ?

Model assumption: $f_k(\mathbf{x})$ is the multivariate normal density $N_p(\boldsymbol{\mu}_k, \Sigma_k)$,

$$f_k(\mathbf{x}) = (2\pi)^{-p/2}|\Sigma_k|^{-1/2}\exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\Sigma_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\}.$$

Under this assumption, the Bayes classifier assigns an observation with $\mathbf{x}$ to class k for which

$$\begin{aligned}\delta_k(\mathbf{x}) &= -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\Sigma_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k) - \frac{1}{2}\log|\Sigma_k| + \log\pi_k \\ &= -\frac{1}{2}\mathbf{x}^T\Sigma_k^{-1}\mathbf{x} + \mathbf{x}^T\Sigma_k^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^T\Sigma_k^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\log|\Sigma_k| + \log\pi_k\end{aligned}$$

is the largest. Unlike LDA, this quantity is quadratic in $\mathbf{x}$, hence the name "quadratic" in the title.

Final remarks:

- Both LDA and QDA are parametric models: they assume the distribution of the predictors in each class is normal. Validity of this assumption must be ensured in order to apply the methods.
- The difference between LDA and QDA is similar to the bias-variance trade off. LDA uses fewer parameters, $p(p+1)$ covariance parameters, while QDA uses much more parameters $Kp(p+1)$; thus QDA is more flexible. As a result LDA has less variance than QDA. General rule of thumb: QDA is recommended if the training set is very large and the variance of the classifier is not a concern, or if the common covariance assumption is clearly violated.
- Logistic regression does not require any assumptions about the distribution of the predictors.
- When the true decision boundaries are linear LDA and logistic regression tend to perform well.
- When the true decision boundaries are non-linear then QDA may give better results
- With the logistic regression we can add quadratic terms $X_1^2, X_2^2, \ldots$ or cross products $X_1X_2, \ldots$ of the predictor $\mathbf{X} = (X_1, \ldots X_p)^T$. Thus the form of the model could yield similar decision boundaries as QDA.

5 K-Nearest-Neighbors

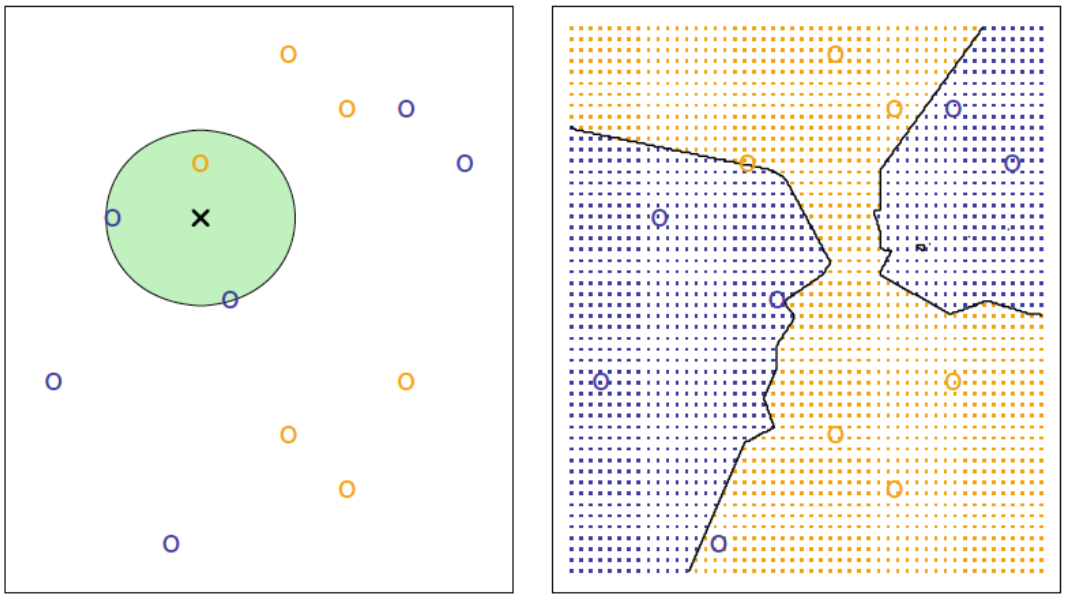
In theory we would always like to predict a class-based response using the Bayes classifier. But for real data this requires knowledge of the conditional distribution of the response given the predictors which we so far modeled using parametric assumptions. In the following two methods we discuss approaches that relax these assumptions.

One such method is the K-nearest neighbors (KNN) classifier. Given a positive integer K and a test observation $\mathbf{x}_0$, the KNN classifier first identifies the neighbors K points in the training data that are closest to $\mathbf{x}_0$, represented by $\mathcal{N}_0$. It then estimates the conditional probability for class j as the fraction of points in $\mathcal{N}_0$ whose response values equal j. Formally we write this as

$$Pr(Y = j|\mathbf{X} = \mathbf{x}_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j), \tag{3}$$

where $\mathcal{N}_0 = \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_0\|$ is among the k smallest distances$\}$

Finally, KNN applies Bayes rule and classifies the test observation $\mathbf{x}_0$to the class with the largest probability. The figure below illustrates the procedure for a toy example (James, Witten, Hastie and Tibshirani, 2013) and considers $K = 3$. Left: a test observation at which a predicted class label is desired is shown as a black cross. The three closest points to the test observation are identified, and it is predicted that the test observation belongs to the most commonly-occurring class, in this case blue. Right: The KNN decision boundary for this example is shown in black. The blue grid indicates the region in which a test observation will be assigned to the blue class, and the orange grid indicates the region in which it will be assigned to the orange class.

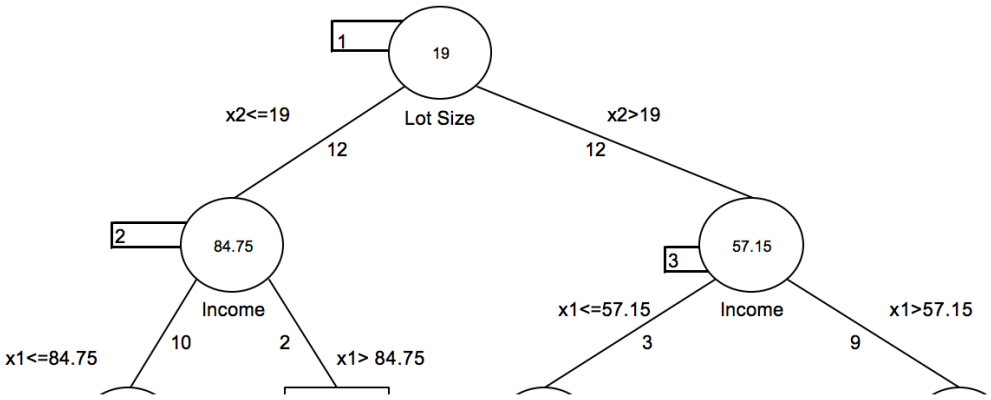


The choice of K has a drastic effect on the KNN classifier obtained. When K is very small, the decision boundary is overly flexible and finds patterns in the data that dont correspond to the Bayes decision boundary. The resulted classifier has low bias but very high variance. On the other hand, if K is too large, the method becomes less flexible and produces a decision boundary that is close to linear. This corresponds to a low-variance but high-bias classifier.

Choosing the correct level of K and thus of flexibility is critical to the success of any statistical learning method. In this sense, splitting the data into a train set and test set and choosing K as the level that minimizes the test set error tends to produce good KNN classifiers.

6 Classification Trees

Tree-based methods for classification involve segmenting the predictor space into a number of simple regions (see the Figure below for the toy example with the riding lawnmowers). The approach is easily extended to accommodate continuous outcome and the decision trees are called "regression trees". Brieman, Friedman, Olshen and Stone (1984) proposed this methodology and developed R program to implement the techniques called CART (classification and regression trees).



As before Y denotes the categorical response and $\mathbf{X} = (X_1, \ldots, X_p)^T$ denoting the predictor. In classification trees:

1. we divide the predictor space - the set of possible values for $X_1, X_2, \ldots, X_p$ - into J distinct and non-overlapping regions $R_1, R_2, \ldots, R_J$.

2. for every observation $\mathbf{x}_{new}$ that falls into the region R_m we predict its class as the most commonly occurring class for the training observations in region R_m.

A tree is "grown" by using recursive binary splitting into two subgroups of the type $X_j < c$ and $X_j \geq c$. The criterion to make the binary splits is based on the *classification error rate*, defined as the fraction of the training observations in that region that do not belong to the most common class. (Notice this classification error rate is based on the training set). However in practice it turns out that the classification error rate is not sufficiently sensitive for growing tress.

Notation $\hat{p}_{mk}$ - represents the proportion of training observations in region R_m that are from the class k. Other more common alternatives are:

- *Gini index*: $G_m = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$. This gives a measure of the total variance across the K classes. Gini index takes a low value for a node (region) for which $\hat{p}_{mk}$ are close to 0 or 1; such type of node with small Gini index is called "pure node" and it indicates that the observations are predominantly from one single class.

- *Entropy*: $D_m = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$. Since all the proportions are between 0 and 1 it follows that the entropy is non-negative. Like Gini index, the entropy will take a small value if the node is pure.

When building a classification tree, either the Gini index or the entropy are typically used to assess the quality of a particular split. In either case one repeats steps [1.] and [2.] until the accuracy criterion used is below a certain threshold.

The algorithm described above may produce good predictions on the train set, but it likely overfits the data and thus may lead to poor test set performance. This is because the tree may be too complex. In this case, smaller tree with fewer splits may be better that is leading to lower variance and better interpretation at the cost of little bias. To bypass this one

3. *prunes* it back in order to obtain a subtree. Typically you select the subtree that yields the lowest misclassification error rate. In this regard the classification error rate is typically preferred; though any of the three evaluation methods could be used.

At step [3.] K-fold cross validation may be used, where the training set is divided into K folds: $(K-1)$ folds are used to grow the tree and the Kth one is used to evaluate the prediction error. The tree that results in the smallest prediction error is selected.

Some advantages and disadvantages of trees

+ Trees are very easy to explain to people. In fact, they are even easier to explain than linear regression!

+ Some believe that decision trees more closely mirror human decision-making than do any other classification techniques studied

+ Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).

+ Trees can easily handle qualitative predictors without the need to create dummy variables.

– Trees generally do not have the same level of predictive accuracy as some of the other classification techniques

– The trees can be very non-robust. In other words, a small change in the data can cause a large change in the final estimated tree.

The classification trees can be improved. Classification trees-based approaches

- Bagging. Create multiple copies of the original training set using the bootstrap; then fit a decision tree to each copy; then combine all of the trees to create a single predictive model. This approach lowers the variance of the classification method, at the expense of interpretation.

- Random forests. Build B decision trees on bootstrap training samples, but for each one use a subset of predictors determined by random selection (typically $m \approx \sqrt{p}$). This reduces the variability even more.

- Boosting/ This works similarly to bagging, except the trees are grown sequentially by using information from the previously grown trees.

We do not cover these methods in detail.

7 Support Vector Machines

Separating hyperplane:

- $Y = \pm 1$ (two-class).
- A hyperplane:

$$\{\mathbf{x} : f(\mathbf{x}) = \mathbf{x}^{\mathsf{T}}\beta + \beta_0 = 0\}$$

- Classification rule:

$$\text{sign}[f(\mathbf{x})] = \text{sign}[\mathbf{x}^{\mathsf{T}}\beta + \beta_0]$$

or equivalently $Yf(\mathbf{x}) > 0$.

Data $\{y_i, \mathbf{x}_i : i = 1, \ldots, n\}$. **When the two classes are linearly separable:**

- More than one hyperplane can separate the training points perfectly.
- Find a hyperplane that achieves biggest margin between the training points for $+1$ and -1.
- Thus we maximize C (margin width) such that

$$y_i(\mathbf{x}_i^{\mathsf{T}}\beta + \beta_0) \geq C, \quad i = 1, \ldots, n.$$

When the two classes overlap:

- allow for some points to be on the wrong side of the margin
- Now we maximize C (margin width) such that

$$y_i(\mathbf{x}_i^{\mathsf{T}}\beta + \beta_0) \geq C(1 - \xi_i), \quad i = 1, \ldots, n;$$

$$\xi_i \geq 0, \quad \sum_{i=1}^{n} \xi_i \leq \text{ constant}$$

When the two classes are not linearly separable:

- $f(\mathbf{x}) = h(\mathbf{x})^{\mathsf{T}}\beta + \beta_0$, where $h(\mathbf{x})$ is a basis expansion of $\mathbf{x}$.
- The quantities $h(\mathbf{x})$ are called "features" (typically unknown)
- SVM utilizes these features and builds a classification rule based on them

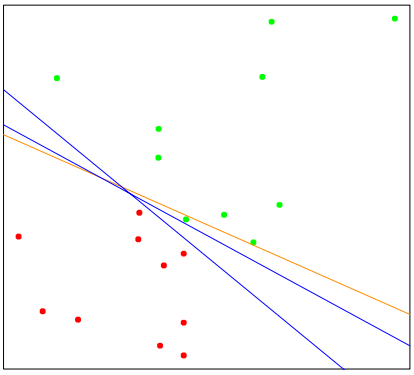


Figure 4.13: *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the* perceptron learning algorithm *with different random starts.*

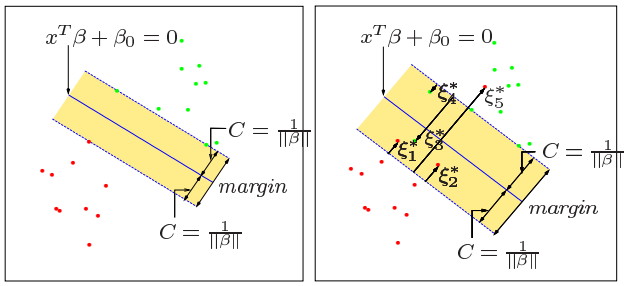


Figure 12.1: *Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width $2C = 2/\|\beta\|$. The right panel shows the nonseparable (overlap) case. The points labeled ξ_j^* are on the wrong side of their margin by an amount $\xi_j^* = C\xi_j$; points on the correct side have $\xi_j^* = 0$. The margin is maximized subject to a total budget $\sum \xi_i \leq$ constant. Hence $\sum \xi_j^*$ is the total distance of points on the wrong side of their margin.*

Classification methods at a glance

LDA	Normal distribution for covariates same covariance matrix
QDA	Normal distribution for covariates different covariance matrix
Logistic regression	No distributional assumption on covariates regression setup
K-NN	No distributional assumption on covariates Nonparametric
Classification trees	No distributional assumption on covariates Nonparametric
SVM	No distributional assumption on covariates Nonparametric