ISLR Chapter 4 Notes

Ryan Heslin

11/12/2020

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(rlang)
```

```
##
## Attaching package: 'rlang'

## The following objects are masked from 'package:purrr':
##
##     %@%, as_function, flatten, flatten_chr, flatten_dbl, flatten_int,
##     flatten_lgl, flatten_raw, invoke, list_along, modify, prepend,
##     splice
```

Classification

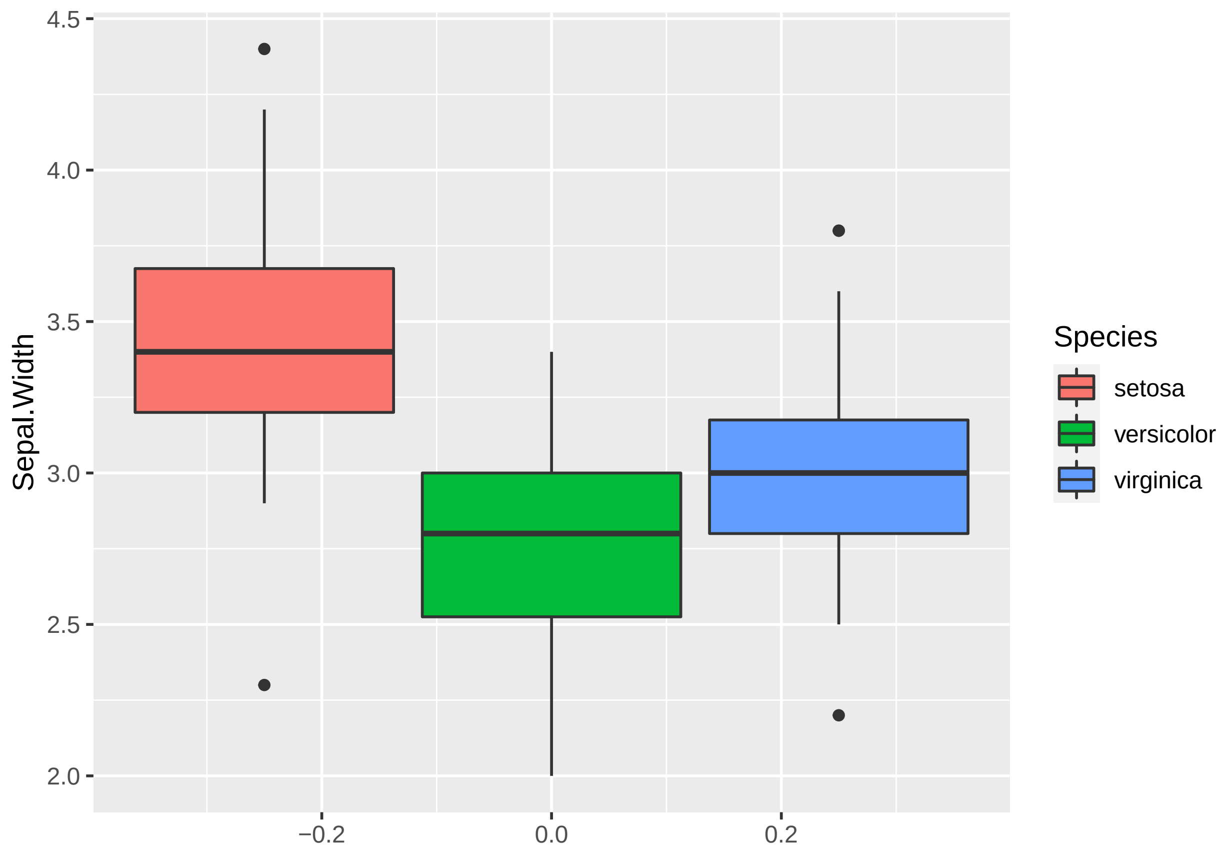
Overview

The general approach to classification is to find probabilites of each class and make a classification.

As with prediction, a classifier should perform well on test as well as training data.

An example where classes are well distinguished

```r
iris %>% ggplot(aes(x = Sepal.Width, fill = Species)) +
  geom_boxplot() +
  coord_flip()
```



Why Not Linear Regression?

Linear regression is a poor choice for classification. Numerically coding the response variable creates an implicit order that may bnot logically exist:

Regression may be used where the outcome is binary and may be coded 0 or 1. However, OLS is inappropriate becuase it will predict probabiliteis outside [0,1].

Logistic Regression

These give the probability of the outcome:

$$P(\text{default} = \text{yes}|\text{balance})$$

The prediction is made at a given level of probability, which is chosen depending on whether Type I or Type II errors are more serious

The Logistic Model

Again, OLS will predict impossible probabilities in most cases. Logistic models take the form:

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

This is fitted using the maxium likelihood function. It always yields an S-shaped curve resembling the normal distriubtion's CDF, where most probability change occurs in the middle of the interval. Adding 1 to the denominator ensures the probability is never greater than 1.

This can be rewritten as:

$$\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$$

The LHS is called the odds - the ratio of class probability to its complement. How many times likelier is the event than the alternative? It is undefined for $p(X) = 0$ and infinity for $p(X) = 1$, hence the asymptotes on the graph.

For instance:

```r
p <- .95
p/ (1-p)
```

```
## [1] 19
```

Logging both sides of the odds equation gives:

$$log(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X$$

So the regression equation gives the natural log of the odds - the logit. A one-unit increase in X increases the log odds by β_1 and multiplies the odds by e_1^{β}. This is a nonlinear relationship: the impact on the odds depends on the value of the predictor. The log odds are simply odds as multiples of e.

```r
b1 <- 4
b2 <- .5
```

Logits are fitted using the maximum likelihood function, which is more general than the partial derivative optimaizer used in OLS. The goal is to ensure each predicted probability is as close as possible to the observed class - 1 for yes and 0 for no (assuming that coding scheme).

$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$

The values of β_0 and β_1 for which this product of summed products is highest are chose.

Once coefficients are chosen, predictions may be made for any X:

```r
b0 <- -10.65
b1 <- .0055
b2 <- .405
X = 1000
X2 <- 1
exp(b0 + b1*X)/ (1 + exp(b0 + b1*X))
```

```
## [1] 0.005765966
```

This gives a probability of under 1%.

Dummy variables can be added and are treated in the usual way, multiplying the coeffieicnt by the numeric code

```
exp(b0 + b1*X + b2*X2)/ (1 + exp(b0 + b1*X +b2*X2))
```

```
## [1] 0.00862011
```

Multiple logistic regression can introduce a subtlety. A predictor's beta may change its sign as more predictors are added. This occurs because the beta indicates the average impact on Y holding all other X constant. When predictors are correlated, they are both assoicated iwth the outcome, but the relationship may change when their effect one each other is controlled for. For example, students have a higher overall defualt rate, but the coefficient of student in a multivariate model is negative because students have lower default rates than nonstudents *with the same balance*. This is an example of confounding. In other words, the students' class mean is higher, but its classification curve gives a lower likelihood of default.

Logistic regression is not recommended for multiple classes.

Linear Discriminant Analysis

LDA extends logistic regression to multiple classes by modeling the predictors separately for each class of Y and use Bayes' rule to derive estiamtes of class probability:

$$Pr(Y = k|X = x)$$

In addition to doing well with multiple response classes, LDA is more stable with samll n.

Bayes' theorem is:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

or the ratio of events B conditioned on A weighted by A's probability to all events B

So if a test is 90% sensitive and 80% specific, and the prevalence is 5%:

```
pA <- .05
pBA <- .9
pB_notA <- .2
pB <- (pBA*pA) + (pB_notA * (1 - pA))

(pBA * pA) / pB
```

```
## [1] 0.1914894
```

$$\frac{.9 \times .05}{.9 \times .05 + .2 \times (1 - .05)}$$

The odds of a positive test (pB) are the sum of sensitivity times prevalence (true positive) and specificity times the complement of prevalence (false positive). In this form, the equation essentially asks what fraction of positives are true. Even if sensitivity is perfect, imperfect specififty means many false positives. From the Bayesian perspective, accuracy is meaningless without knowledge of the prior distribution.

We have k classes. π_k represents the prior probability (i.e., outside our model) of a random observation falling in each k. The dneisty function of an obsevation for class k is:

$$f_k(x) = Pr(X = x|Y = k)$$

The larger $f_k(x)$ for a given x, the higher the odds an observation of class k has x. By Bayes' theorem:

$$Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

Remember to substiute in $f_k(x)$! The odds of belonging to class k for a given value of x are the odds of class k given that predictor's value divided by the odds of the predictor taking that value. This is the same as dividing the likelihood of true positives by the likelihood of *any* positives. This is the posterior probability.

π_k is easy to estiamte from sample proportions of the class, but $f_k(x)$ is more challenging. Recall that we use the bayes classifier, which chooses the class for which $p_k(X)$ is highest.

To start, we assumed $f_k(x)$ has normal density. For a single predictor, this is:

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} exp(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2)$$

with μ_k and σ_k^2 are the mean and variance of class k. Thus the function differs for each class.

By substituting this formula for $f_k(x)$ into Bayes' rule above, we get:

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} exp(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2)}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma_k} exp(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2)}$$

By logging both sides, we can find the formula for the Bayes classifier, which assigns whichever class for which this valueis highest:

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

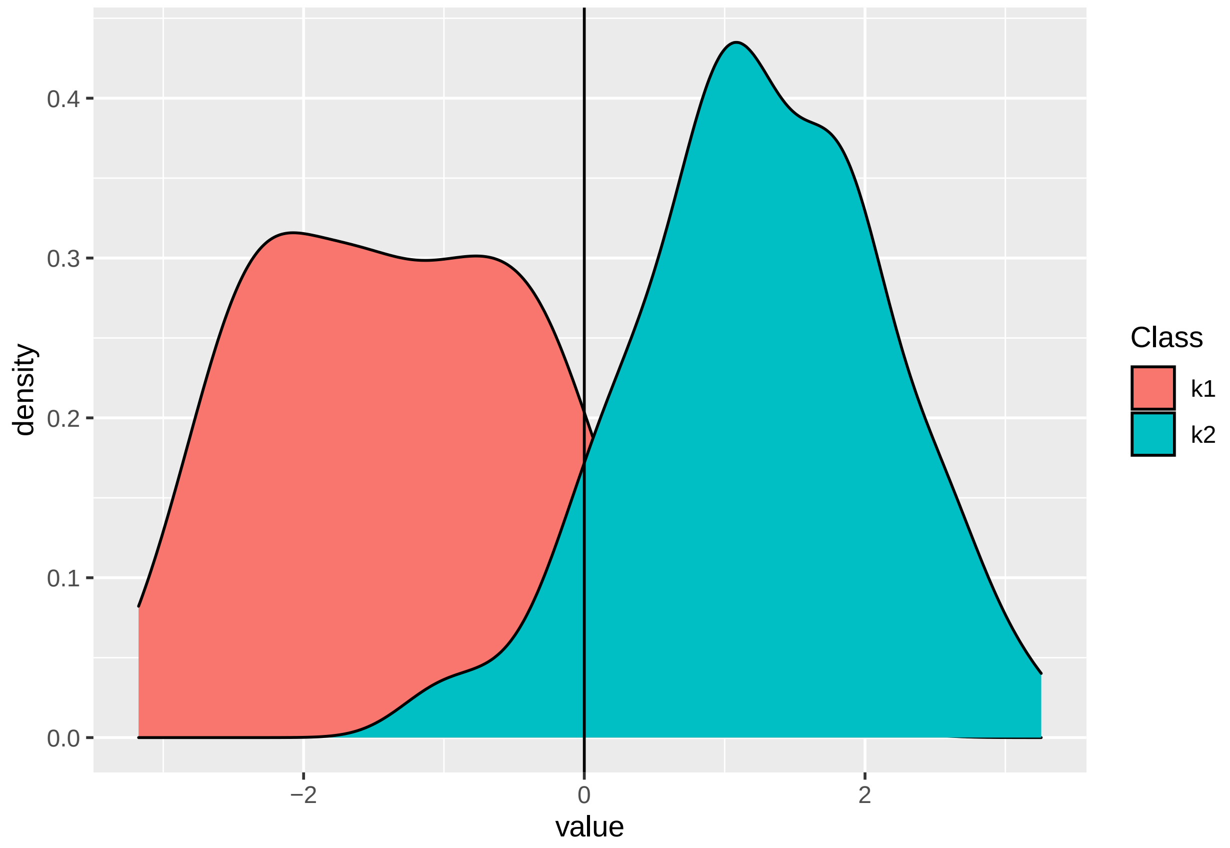
Holding π_k constant, x for a single class becomes:

$$\delta_k(x) = \frac{\mu_k}{2}$$

So if there are two classes with equal prior probabilites, the Bayesian decision boundary is simply the midpoint between the means.

$$x = \frac{\mu_1^2 - \mu_2^2}{2(\mu_1 - \mu_2)} = \frac{\mu_1 + \mu_2}{2}$$

```r
tibble(k1 = rnorm(100, mean = -1.25), k2 = rnorm(100, mean = 1.25)) %>%
  pivot_longer(everything(), names_to = "Class") %>%
  ggplot(aes(value, fill =Class)) +
    geom_density() +
    geom_vline(xintercept = 0)
```



In practice, we still need to estimate the means and variances for the class distributions, as well as the prior probability.

$$\mu_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\sigma^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \mu_k)^2$$

so mean is simply the class mean, and variance is obtained by summing variances for each class and dividing by degrees of freedom. Prior probabilities are simply $\pi_k = \frac{n_k}{n}$

LDA for $\mathbf{p > 1}$

For this case, we assume that all predictors come from a multivariate normal distribution with distinct means for each predictor (the mean vector) but a common covariance matrix. x here is vector of order p.

If predictors are uncorrelated, the multidimensional distributionr resembles the familiar bell curve. If they are correlated, the shape is more elliptical. To indicate that X has a multivariate normal distribution, we write $X\ N(\mu, \Sigma)$ Covariance has replaced variance as the parameter. The density is:

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

The Σ^{-1} term is precision - the inverse of the covariance matrix Remmember that μ is a vector of p predictor means.. The optimizer equation is is simply a vectorized version of the one-predictor case that finds the most probable class for the observation vector.:

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + log \pi_k$$

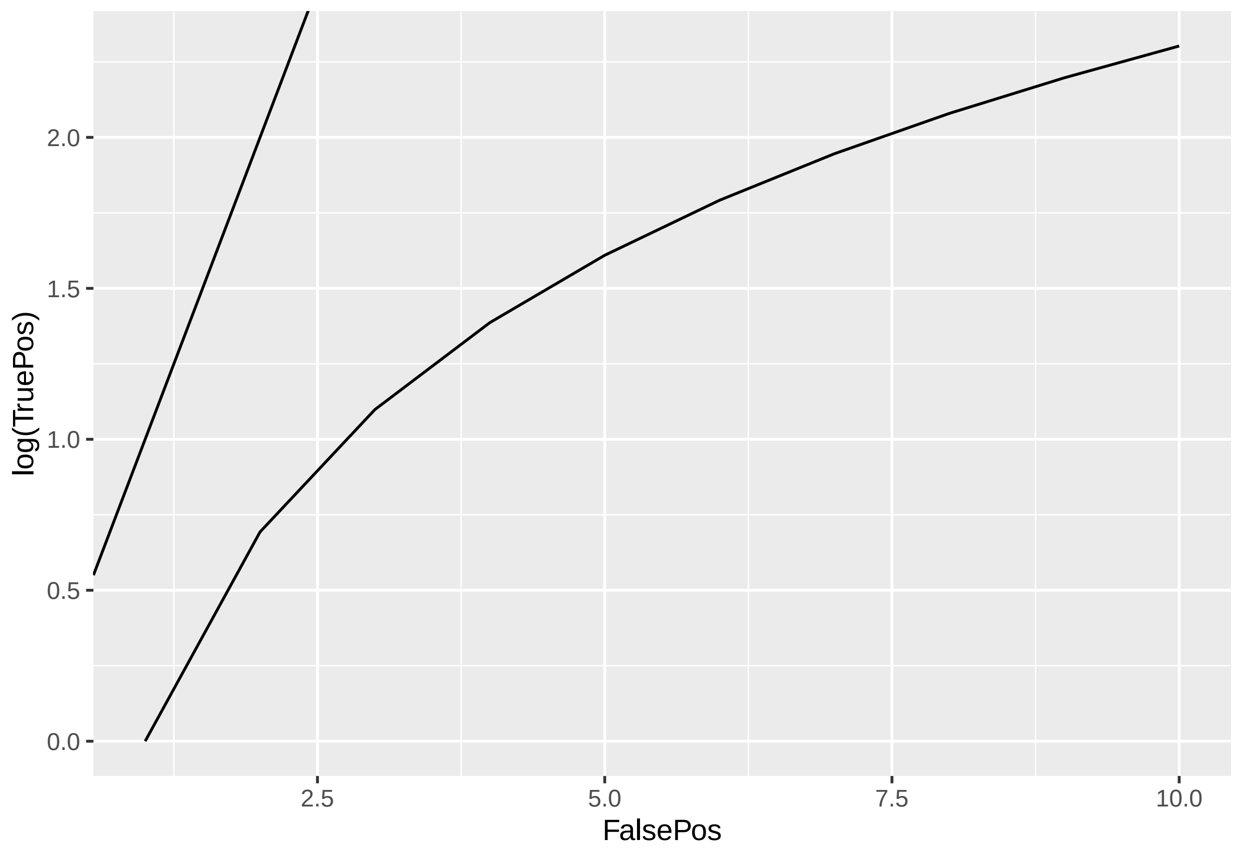
A decision boundary now exists for each pair of classes, so there are six in a k = 3 model.

Errors and Caveats

If one class is rare, the null classifier can do very well by always excluding it. Recall that sensitivity is true positives and specificty true negatives. The Bayes classifier optimizes *overall* accuracy (class for which posterior probability is greatest), so it is inappropriate if one type of error is much more consequential than the other. In a 2-class case, the classifer threshold could be set to 20% to increase sensitivity.

The ROC curve displays performance across different thresholds. The area under the curve represents total performance; 1 would indicate perfection

```r
data.frame(FalsePos = 1:10, TruePos = 1:10) %>%
  ggplot(aes(FalsePos, log(TruePos))) +
  geom_line() +
  geom_abline()
```



Quadratic Discriminant Analysis

QDA assumes each class has itw own covariance matrix with the other classes; that is, relationships among predictors vary by class. This adds $Kp(p+1)/2$ additional parameters to estimate (from the free parameters of each covariance matrix), greatly increasing flexibility at the cost of variance. It is warranted if there are very many training observations, or if it is known there is no common covariance matrix.

Conclusion

LDA and log regresson are fundamentally the same. 1-predictor LDA reduces to:

$$log\left(\frac{p_1(x)}{p_2(x)}\right) = c_0 + c_{1x}$$

Where the RHS derives from the model parameters. This is almost the same as the linear regression form, the only differnce being the estimation method.

Overall, logistic regression and LDA are fairly close. KNN is far more flexible than any regression method but does not provide interpretable coefficitns. QDA falls somewhere between LDA and QDA.

The correct decision depends on the shape of the theoretical Bayes decisions boundary - the less linear, the more flexible the appropriate model.