Chapter 4: Classification

The linear model in Ch. 3 assumes the response variable Y is quantitiative. But in many situations, the response is categorical.

e.g. eye color
cancer diagnosis
product purchase.

In this chapter we will look at approaches for predicting categorical responses, a process known as *classification*.

Classification problems occur often, perhaps even more so than regression problems. Some examples include

1. A person arrives in the ER w/ set of symptoms that could possibly be attributed to one of 3 conditions. Which one of these conditions does the person have?

2. An online banking service must be able to determine if a transaction is fraudulent on basis of user's IP address, past transaction history, etc.

3. Something is in the street in front of the self-driving car you are riding in. The car must determine if it is human or another car.

As with regression, in the classification setting we have a set of training observations $(x_1, y_1), \dots, (x_n, y_n)$ that we can use to "build a classifier". We want our classifier to perform well on the training data and also on data not used to fit the model (**test data**).

fit a model

most importantly!

We will use the `Default` data set in the `ISLR` package for illustrative purposes. We are interested in predicting whether a person will default on their credit card payment on the basis of annual income and credit card balance.

yes or no $\Rightarrow$ categorical.

head (Default)

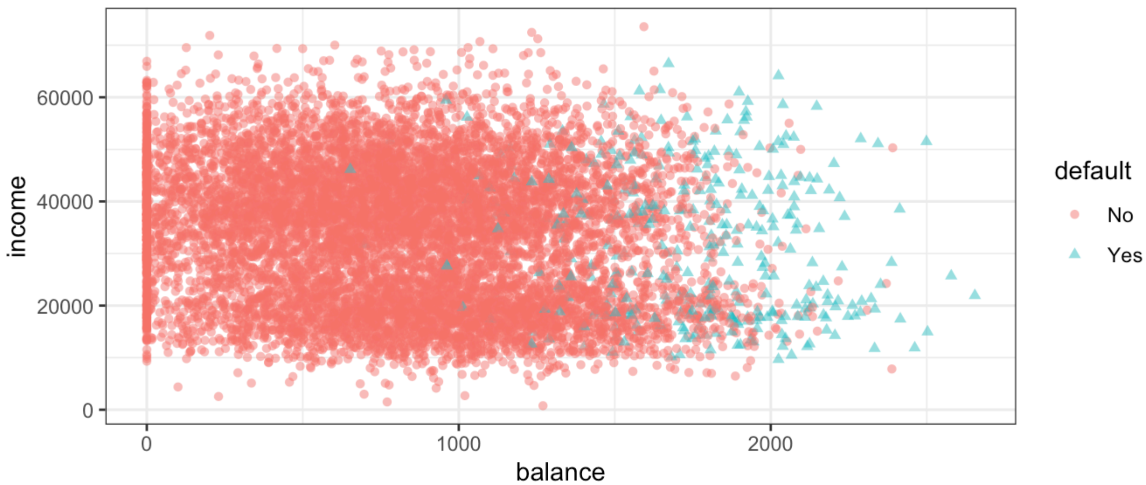
focus on these first.

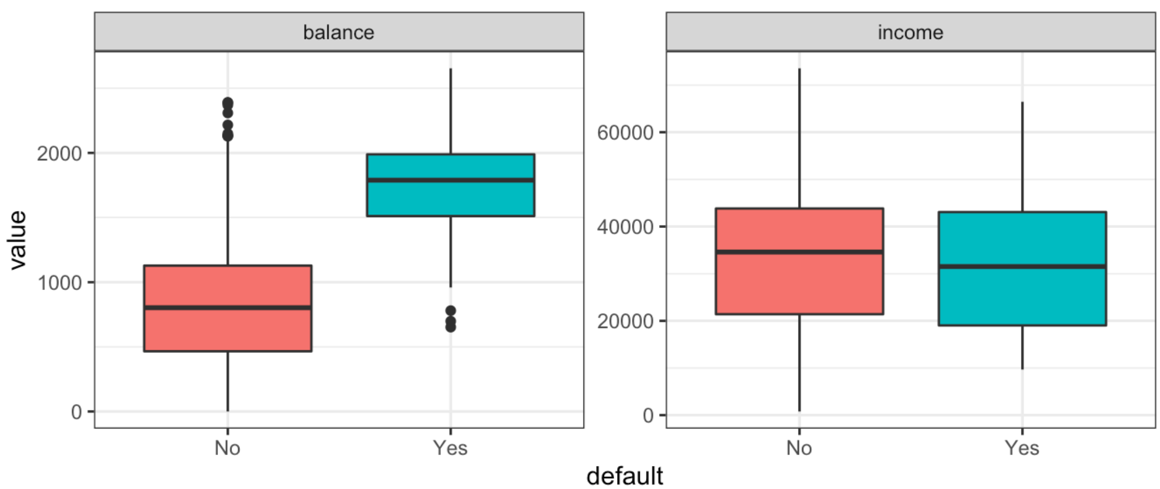
```
##   default student   balance    income
## 1      No      No  729.5265 44361.625
## 2      No     Yes  817.1804 12106.135
## 3      No      No 1073.5492 31767.139
## 4      No      No  529.2506 35704.494
## 5      No      No  785.6559 38463.496
## 6      No     Yes  919.5885  7491.559
```

y

features

some separation





pronounced relationship btw/ balance and default

↳ in most problems, this relationship is not so clear.

1 Why not Linear Regression?

I have said that linear regression is not appropriate in the case of a categorical response. Why not?

Let's try it anyways. We could consider encoding the values of `default` in a quantitative repsonse variable Y

$$Y = \begin{cases} 1 & \text{if } \texttt{default} \\ 0 & \text{otherwise} \end{cases}$$

Using this coding, we could then fit a linear regression model to predict Y on the basis of `income` and `balance`. This implies an ordering on the outcome, not defaulting comes first before defaulting and insists the difference between these two outcomes is 1 unit. In practice, there is no reason for this to be true.

We could let $Y = \begin{cases} 0 & \text{default} \\ 1 & \text{don't default} \end{cases}$

$Y = \begin{cases} 0 & \text{default} \\ 100 & \text{don't} \end{cases}$

there is no natural reason to use 0/1 encoding. but is has an advantage:

Using the 0/1 dummy encoding, we can get a rough estimate of $P(\texttt{default}|X)$, but it is not guaranteed to be scaled correctly.

doesn't need to be between 0 and 1 but will provide an ordering.

Real problem: this cannot easily be extended to more than 2 classes

We will instead use methods specifically formulated for categorical response.

2 Logistic Regression

Let's consider again the `default` variable which takes values `Yes` or `No`. Rather than modeling the response directly, logistic regression models the *probability* that Y belongs to a particular category.

e.g. $P(\text{default} = \text{yes} \mid \text{balance})$

which we can abbreviate $p(\text{balance}) \in [0,1]$.

For any given value of `balance`, a prediction can be made for `default`.

e.g. predict default = yes if $p(\text{balance}) > 0.5$

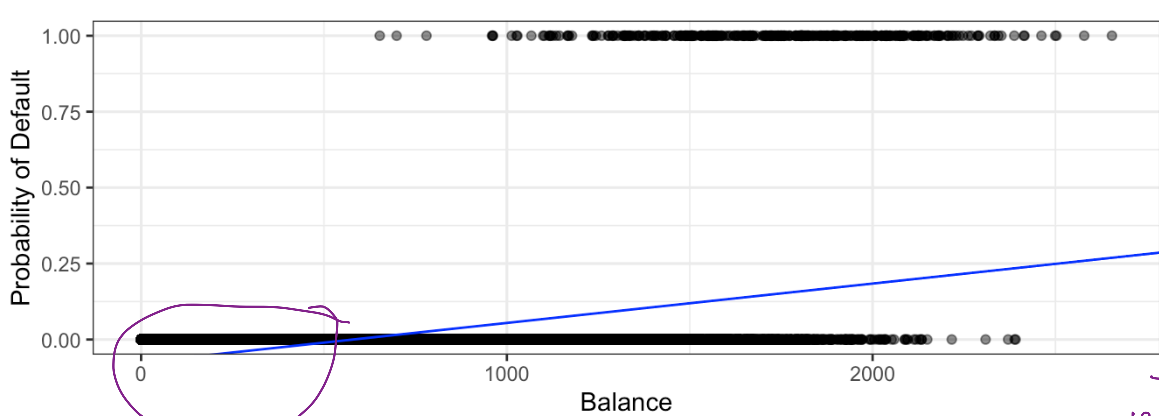
or the company could be more conservative and predict default = yes if $p(\text{balance}) > \underbrace{0.1}_{\text{threshold}}$

2.1 The Model

using 0/1 encoding.

How should we model the relationship between $p(X) = P(Y = 1|X)$ and X? We could use a linear regression model to represent those probabilities

$$p(X) = \beta_0 + \beta_1 X + \varepsilon$$



If we had balances very large would be predicting probability above 1

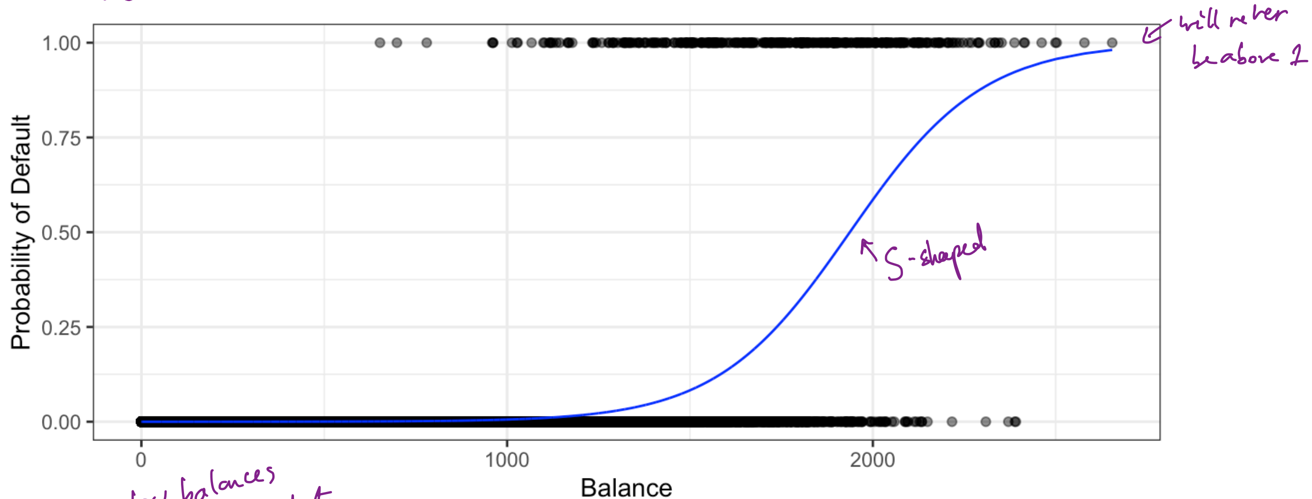
problem: for balances close to 0 we predict negative probability

neither of these things make sense!

To avoid this, we must model $p(X)$ using a function that gives outputs between 0 and 1 for all values of X. Many functions meet this description, but in *logistic* regression, we use the *logistic* function,

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

We will use maximum likelihood estimation to estimate coefficients (β_0, β_1) – later.



low balances are predicted to have probability close to 0, but never below 0.

After a bit of manipulation,

$$\underbrace{\frac{p(x)}{1-p(x)}}_{\text{"odds"}} = e^{\beta_0+\beta_1 X}$$

"odds" → can take any value between 0 and ∞

⇒ low prob of default (0)

⇒ high prob. of default (∞)

e.g. if $p(x) = 0.2$ (1 in 5 ppl default)

$$\Rightarrow \text{odds} = \frac{0.2}{1-0.2} = \frac{1}{4}$$

By taking the logarithm of both sides we see,

$$\underbrace{\log\left(\frac{p(X)}{1-p(X)}\right)}_{\substack{\text{"log-odds"} \\ \text{"logit"}}} = \beta_0 + \beta_1 X$$

log-odds are linear in X.

Recall from Ch. 3 that β_1 gives the "average change in Y associated with a one unit increase in X." In contrast, in a logistic model,

increasing X by one unit changes log-odds by β_1

$\iff$

increasing X by one unit multiplies the odds by e^{β_1}

However, because the relationship between $p(X)$ and X is not linear, β_1 does **not** correspond to the change in $p(X)$ associated with a one unit increase in X. The amount that $p(X)$ changes due to a 1 unit increase in X depends on the current value of X.

Regardless of the value of X,

If β_1 is positive $\Rightarrow$ increasing X increases $p(X)$

If β_1 is negative $\Rightarrow$ increasing X decreases $p(X)$.

2.2 Estimating the Coefficients

The coefficients β_0 and β_1 are unknown and must be estimated based on the available training data. To find estimates, we will use the method of *maximum likelihood*.

The basic intuition is that we seek estimates for β_0 and β_1 such that the predicted probability $\hat{p}(x_i)$ of default for each individual corresponds as closely as possible to the individual's observed default status.

to do this, use the likelihood function $\ell(\beta_0, \beta_1) = \prod_{i: y_i = 1} p(x_i) \prod_{i: y_i = 0} (1 - p(x_i))$

$= \prod_{i: y_i=1} \frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}} \prod_{i: y_i=0} \frac{1}{1+e^{\beta_0+\beta_1 x_i}}$

$\hat{\beta}_0$ and $\hat{\beta}_1$ chosen to maximize $\ell(\beta_0, \beta_1)$.

in fact least squares method is equivalent to maximum likelihood

```
logistic_spec <- logistic_reg()

logistic_fit <- logistic_spec |>
  fit(default ~ balance, family = "binomial", data = Default)

logistic_fit |>
  pluck("fit") |>
  summary()
```

formula as before.

Y takes values in $\{0, 1\}$.

```
##
## Call:
## stats::glm(formula = default ~ balance, family = stats::binomial,
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2697  -0.1465  -0.0589  -0.0221   3.7589
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.065e+01  3.612e-01  -29.49   <2e-16 ***
## balance      5.499e-03  2.204e-04   24.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1596.5  on 9998  degrees of freedom
## AIC: 1600.5
##
## Number of Fisher Scoring iterations: 8
```

$\hat{\beta}_0$ $\hat{\beta}_1$

accuracy of estimates

test statistic $\hat{\beta}_i / \hat{se}(\hat{\beta}_i)$

Hypothesis test $H_0: \beta_i = 0$ $H_a: \beta_i \neq 0$ $i = 0, 1$

for $i = 1$, H_0 implies $p(X) = \frac{e^{\beta_0}}{1+e^{\beta_0}}$ doesn't depend on X (i.e. no relationship between $p(X)$ and X)

there is a significant relationship between default & balance.

$\hat{\beta}_1 = 0.0055 \Rightarrow$ increase in balance of \$1 is associated w/ increase in prob of default

↳ increase in log-odds of default by .0055 units.

↳ multiplicative increase in P(default) by $e^{.0055}$

2.3 Predictions

Once the coefficients have been estimated, it is a simple matter to compute the probability of `default` for any given credit card balance. For example, we predict that the default probability for an individual with `balance` of $1,000 is

$$\hat{p}(x) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x}} = \frac{e^{-10.6513 + 0.0055 \times 1000}}{1 + e^{-10.6513 + 0.0055 \times 1000}} = 0.00575$$

In contrast, the predicted probability of default for an individual with a balance of $2,000 is

$$\hat{p}(x) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x}} = \frac{e^{-10.6513 + 0.0055 \times 2000}}{1 + e^{-10.6513 + 0.0055 \times 2000}} = 0.586$$

$58.6\% > 50\% \Rightarrow$ maybe we would predict default = yes if threshold $= 0.5$

2.4 Multiple Logistic Regression

We now consider the problem of predicting a binary response using multiple predictors. By analogy with the extension from simple to multiple linear regression,

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p$$

$$\Downarrow$$

$$p(x) = \frac{e^{\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p}}$$

Just as before, we can use maximum likelihood to estimate $\beta_0, \beta_1, \dots, \beta_p$.

0/1 response.

```
logistic_fit2 <- logistic_spec |>
  fit(default ~ ., family = "binomial", data = Default)
```

y ~ every other column in data

```
logistic_fit2 |>
  pluck("fit") |>
  summary()
```

```
##
## Call:
## stats::glm(formula = default ~ ., family = stats::binomial, data =
data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4691  -0.1418  -0.0557  -0.0203   3.7383
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
## studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
## balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
## income       3.033e-06  8.203e-06   0.370  0.71152
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1571.5  on 9996  degrees of freedom
## AIC: 1579.5
##
## Number of Fisher Scoring iterations: 8
```

$\hat{\beta}$ $\hat{SE}(\hat{\beta})$ $H_0: \beta_i = 0$, $H_a: \beta_i \neq 0$

dummy variable.

no significant relationship w/ income.

$\hat{\beta}_{\text{student(yes)}} < 0 \Rightarrow$ if you are a student LESS likely to default holding balance and income constant.

Student confounded w/ balance – if you are a student you are more likely to have a higher balance) but if you are a non-student w/ same balance/income more to default.

By substituting estimates for the regression coefficients from the model summary, we can make predictions. For example, a student with a credit card balance of \$1,500 and an income of \$40,000 has an estimated probability of default of

$$\hat{p}(x) = \frac{e^{-10.869 + 0.00574 \times 1500 + 0.000003 \times 40000 + (-0.6468)\cdot 1}}{1 + e^{-10.869 + 0.00574 \times 1500 + 0.000003 \times 40000 + (-0.6468)\cdot 1}}$$

$$= 0.058$$

A non-student with the same balance and income has an estimated probability of default of

$$\hat{p}(x) = \frac{e^{-10.869 + 0.00574 \times 1500 + 0.000003 \times 40000 + (-\cancel{0.6468})\cdot 0}}{1 + e^{-10.869 + 0.00574 \times 1500 + 0.000003 \times 40000 + (-0.64\cancel{68})\cdot 0}}$$

$$= 0.105$$

see augment or predict in R

2.5 Logistic Regression for > 2 Classes

We sometimes which to classify a response variable that has more than two classes. There are multi-class extensions to logistic regression ("multinomial regression"), but there are far more popular methods of performing this.

3 LDA "linear discriminant analysis"

Logistic regression involves direction modeling $P(Y = k|X = x)$ using the logistic function for the case of two response classes. We now consider a less direct approach.

Idea:

Model the distribution of predictors X separately in each of the response classes (given Y) and the use Bayes theorem to flip these probabilities and get estimates for $P(Y=k|X=x)$

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

Why do we need another method when we have logistic regression?

1. When classes are well separated, the parameter estimates for logistic regression are surprisingly unstable.
2. If n is small (and distributions of X approximately match what we assume in LDA -- Normal) LDA is more stable than logistic regression.
3. We might have more than 2 response classes.

3.1 Bayes' Theorem for Classification

Suppose we wish to classify an observation into one of K classes, where $K \geq 2$.

Categorical Y can take on K possible distinct and unordered values.

notation

π_k — overall or "prior" probability that a randomly chosen observation comes from the k^{th} class.

$$f_k(x) = \begin{cases} P(X = x \mid Y = k) & \text{discrete } X \\ \text{prob that } X \text{ falls in a small region around } x \text{ given } Y = k & (\text{continuous } X). \end{cases}$$

"density function" of X for an observation that comes from class k

$$\overset{A}{P(Y = k} | \overset{B}{X = x)} = \frac{\overset{P(Y=k)}{\pi_k} \; \overset{P(X=x \mid Y=k)}{f_k(x)}}{\underbrace{\sum_{\ell=1}^{K} \pi_\ell f_\ell(x)}_{P(X=x)}} \quad \text{(Bayes theorem)}$$

We will use the abbreviation $p_k(x)$ as before to denote $P(Y=k \mid X=x)$ "posterior probability" that an obs $X = x$ comes from class k

In general, estimating π_k is easy if we have a random sample of Y's from the population.

Compute fraction of training observations that come from k^{th} class.

Estimating $f_k(x)$ is more difficult unless we assume some particular forms.

If we can estimate $f_k(x)$ we can develop a classifier that is close to the "best" classifier (more later).

3.2 p = 1

Let's (for now) assume we only have 1 predictor. We would like to obtain an estimate for $f_k(x)$ that we can plug into our formula to estimate $p_k(x)$. We will then classify an observation to the class for which $\hat{p}_k(x)$ is greatest.

$p=1$

$\dfrac{\pi_k f_k(x)}{\sum_{j=1}^{K} \pi_j f_j(x)}$

estimating the Bayes classifier!

assigning to class w/ highest $p_k(x)$ is called the "Bayes classifier" is known to be "optimal" i.e. we can do no better!

Suppose we assume that $f_k(x)$ is normal. In the one-dimensional setting, the normal density takes the form

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

μ_k and σ_k^2 are mean and variance parameters for k^{th} class.

Let's also assume (for now) $\sigma_1^2 = \dots = \sigma_K^2 = \sigma^2$ (shared variance term).

Plugging this into our formula to estimate $p_k(x)$,

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu_k)^2\right)}{\sum_{\ell=1}^{K} \underbrace{\pi_\ell}_{\text{prior probability}} \frac{1}{\sqrt{2\underbrace{\pi}_{3.14159\ldots}\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu_\ell)^2\right)}$$

that obs. falls in k^{th} class.

We then assign an observation $X = x$ to the class which makes $p_k(x)$ tthe largest. This is equivalent to

(log + rearranging)

assign obs to class for which

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

is largest.

Example 3.1 Let $K = 2$ and $\pi_1 = \pi_2$. When does the Bayes classifier assign an observation to class 1?

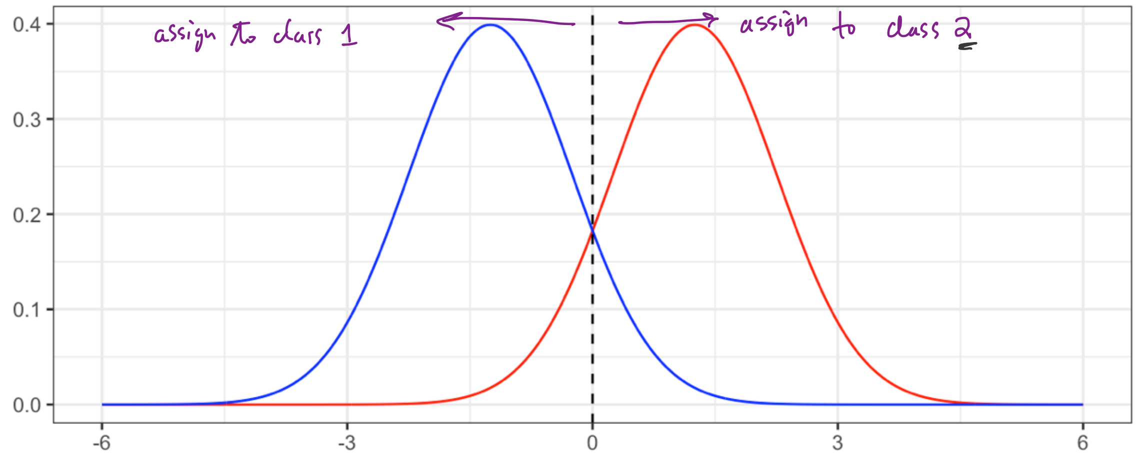
When $\delta_1(x) > \delta_2(x)$

$$\iff x\frac{\mu_1}{\cancel{\sigma^2}} - \frac{\mu_1^2}{2\cancel{\sigma^2}} + \cancel{\log(\pi_1)} > x\frac{\mu_2}{\cancel{\sigma^2}} - \frac{\mu_2^2}{2\cancel{\sigma^2}} + \cancel{\log(\pi_2)}$$

$$\iff 2x(\mu_1 - \mu_2) > \mu_1^2 - \mu_2^2$$

$$\iff x > \frac{\mu_1 + \mu_2}{2}$$

Bayes classifier.



↑ decision boundary at $\frac{-1.25+1.25}{2} = 0$

example where $\pi_1 = \pi_2 = 0.5$

$\mu_1 = -1.25,\ \mu_2 = 1.25,\ \sigma = 1$

In this case, we know $f_k(x) \sim N(\mu_k, \sigma^2)$, π_k $\Rightarrow$ we can create the Bayes classifier!

In practice, even if we are certain of our assumption that X is drawn from a Gaussian distribution within each class, we still have to estimate the parameters $\mu_1, \dots, \mu_K, \pi_1, \dots, \pi_K, \sigma^2$. to estimate the Bayes classifier

The *linear discriminant analysis* (LDA) method approximated the Bayes classifier by plugging estimates in for π_k, μ_k, σ^2.

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i: y_i = k} x_i$$ ← average of training observations in class k

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i: y_i = k} (x_i - \hat{\mu}_k)^2$$ ← weighted average of class variances.

n = total training obs.

n_k = # training obs in class k.

from "science" or directly from problem.

Sometimes we have knowledge of class membership probabilities $\pi_1, \dots, \pi_K$ that can be used directly. If we do not, LDA estimates π_k using the proportion of training observations that belong to the kth class.

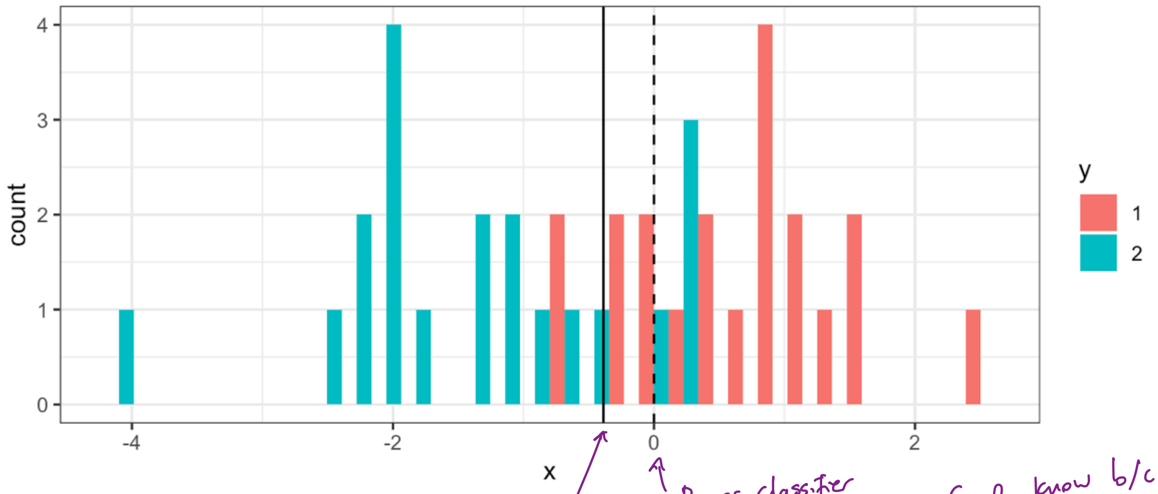
$$\hat{\pi}_k = \frac{n_k}{n}$$

The LDA classifier assigns an observation $X = x$ to the class with the highest value of

$$\hat{\delta}_k(x) = x \frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + \log(\hat{\pi}_k)$$

linear in x $\Rightarrow$ "linear discriminant analysis" histogram of randomly sampled points from class 1 & 2 (prev. plot).

$n_1 = n_2 = 20$



LDA boundary (based on training data)

Bayes classifier decision boundary (only know b/c I simulated the data)

$\frac{\hat{\mu}_1 + \hat{\mu}_2}{2}$

predicted

```
##      pred
## y        1     2
##    1 18966  1034
##    2  3855 16145
```

got wrong

true value

got right

"confusion matrix" on simulated many test points (20k from each class)

The LDA test error rate is approximately 12.22% while the Bayes classifier error rate is approximately 10.52%.

The Bayes error rate is the best we could possibly do in this problem!
(we can only estimate it because this is a simulated example)

The LDA approach did almost as well!

The LDA classifier results from assuming that the observations within each class come from a normal distribution with a class-specific mean vector and a common variance σ^2 and plugging estimates for these parameters into the Bayes classifier.

we will relax this assumption later

3.3 p > 1

We now extend the LDA classifier to the case of multiple predictors. We will assume

$X = (X_1, \dots, X_p)$ drawn from Multivariate Normal dsn w/ class specific mean vector & common covariance

↳ each individual component follows Normal dsn and some covariance between components.

p×1 vector p×p matrix

$$X \sim N_p(\mu, \Sigma) \Rightarrow \begin{array}{l} EX = \mu \\ cov(X) = \Sigma \end{array}$$

=Normal

Formally the multivariate Gaussian density is defined as

"transpose"

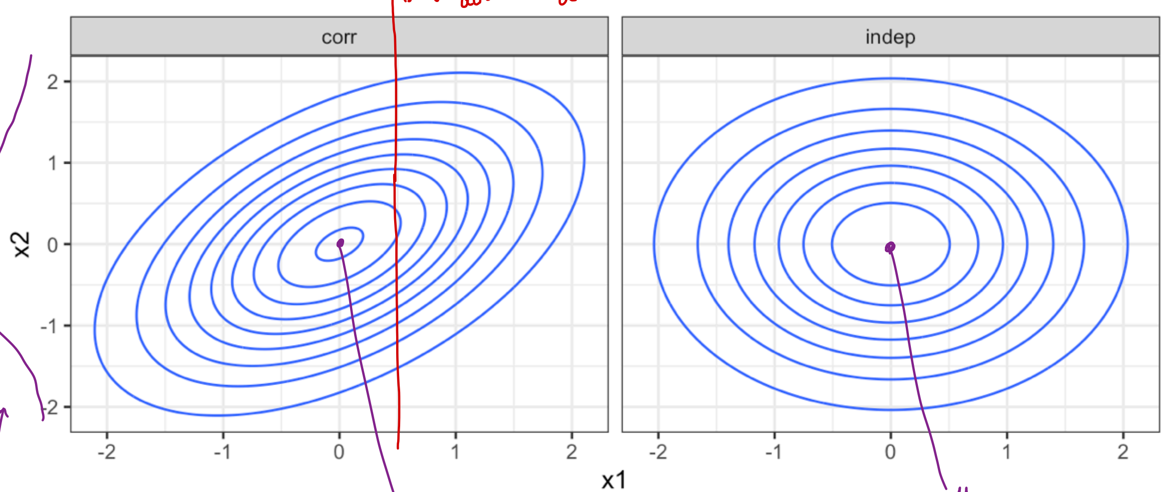
$$f(\underline{x}) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\underline{x}-\underline{\mu})^T \Sigma^{-1} (\underline{x}-\underline{\mu})\right)$$

matrix inverse.

"trace"
↳ sum of diagonal elements

if you sliced across (conditioned on a value of X1 or X2) ⇒ Normal dsn

$cov(X_1, X_2) = \frac{1}{2}$ ⇒ more oval shape



independence ⇒ $cov(X_1, X_2) = 0$ results in "round" Normal dsn.

μ μ

if you marginalize out X_1 (or X_2) ⇒ Normal dsn

p=2 Gaussian density w/ $\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and 2 Σs.

In the case of $p > 1$ predictors, the LDA classifier assumes the observations in the kth class are drawn from a multivariate Gaussian distribution $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$. common covariance.

↑class specific mean

Plugging in the density function for the kth class, results in a Bayes classifier

assign an observation $X = \underline{x}$ to class which maximizes

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

This decision rule is still linear in $\underline{x}$

Once again, we need to estimate the unknown parameters $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K, \pi_1, \dots, \pi_K, \boldsymbol{\Sigma}$.

use similar formulas as for $p=1$ case.

To classify a new value $X = x$, LDA plugs in estimates into $\delta_k(x)$ and chooses the class which maximized this value. $\Rightarrow$ get $\hat{\delta}_k(x)$, choose k which maximizes (i.e. estimating Bayes classifier)

Let's perform LDA on the `Default` data set to predict if an individual will default on their CC payment based on balance and student status.

```
lda_spec <- discrim_linear(engine = "MASS")
            "LDA"                       package actually performing LDA.

lda_fit <- lda_spec |>
  fit(default ~ student + balance, data = Default)

lda_fit |>
  pluck("fit")
```

formula just like linear & logistic regression $Y \sim X$

```
## Call:
## lda(default ~ student + balance, data = data)
##
## Prior probabilities of groups:
##     No    Yes
## 0.9667 0.0333
##
## Group means:
##     studentYes   balance
## No   0.2914037  803.9438
## Yes  0.3813814 1747.8217
##
## Coefficients of linear discriminants:
##                     LD1
## studentYes -0.249059498
## balance     0.002244397
```

$\leftarrow \hat{\pi}_k$ based on training data

$\hat{\underline{\mu}}_1$, $\hat{\underline{\mu}}_2$

linear combinations of student & balance use to form the LDA decision rule.

```r
# training data confusion matrix
lda_fit |>
  augment(new_data = Default) |>
  conf_mat(truth = default, estimate = .pred_class)
```

predict on new-data

confusion matrix

training data

column names

column names

overall training error rate $= 2.75\%$

```
##             Truth
## Prediction     No   Yes
##          No  9644   252
##          Yes   23    81
```

got wrong.

got right

For Default = Yes only got $\frac{81}{252+81} = 24\%$ right!

Why does the LDA classifier do such a poor job of classifying the customers who default?

Only 3.33% of individuals in training data defaulted!

$\Rightarrow$ A simple (but useless) classifier that predicts default = NO get only 3.33% wrong

LDA is trying to approximate Bayes classifier $\Rightarrow$ yield smallest possible overall error rate

A CC company may want to avoid miss classifying default = YES people so can adjust how to select classes.

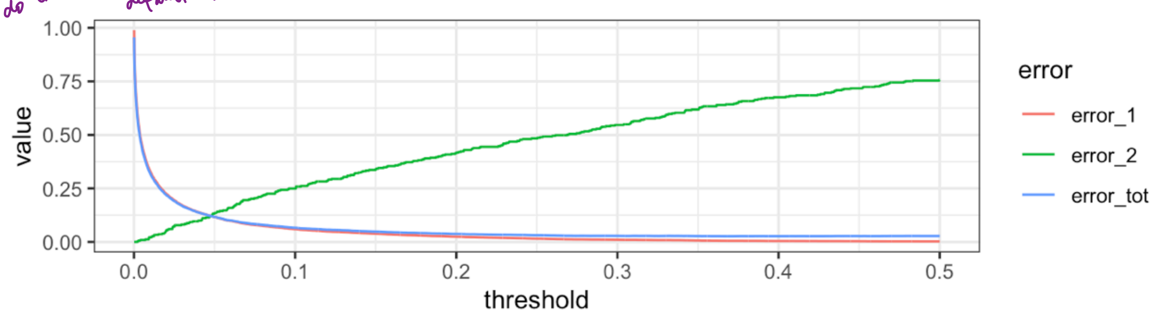
```r
lda_fit |>
  augment(new_data = Default) |>
  mutate(pred_lower_cutoff = factor(ifelse(.pred_Yes > 0.2, "Yes",
         "No"))) |>
  conf_mat(truth = default, estimate = pred_lower_cutoff)
```

can adjust threshold $\Rightarrow$ no longer approximating Bayes classifier.

```
##             Truth
## Prediction     No   Yes
##          No  9432   138
##          Yes  235   195
```

do worse at default = No.

no better at default = Yes



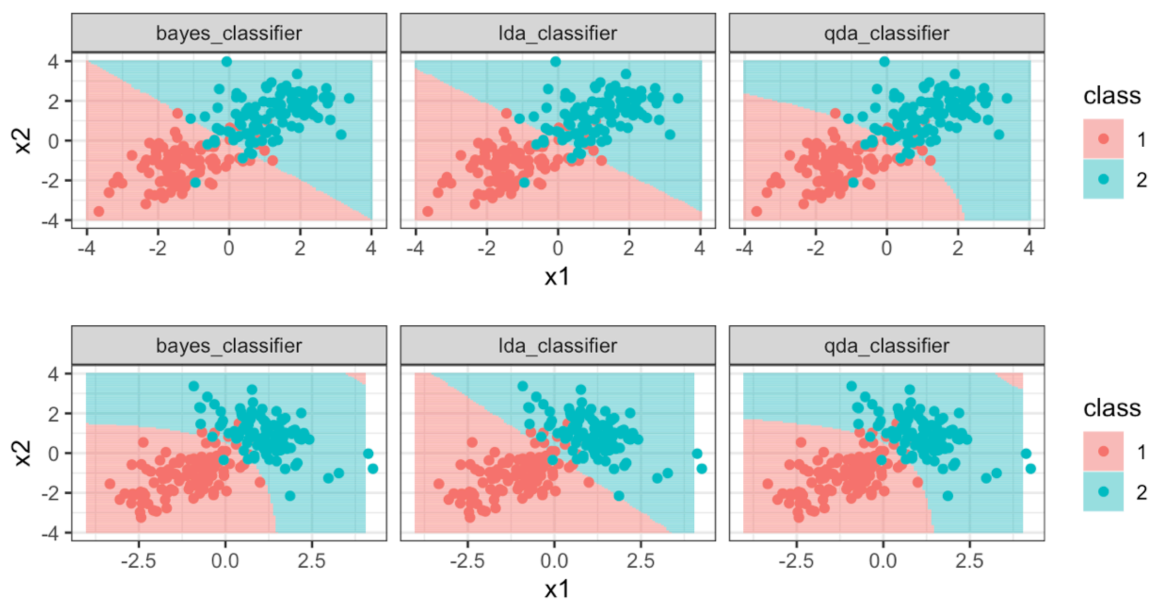
3.4 QDA

LDA assumes that the observations within each class are drawn from a multivariate Gaussian distribution with a class-specific mean vector and a common covariance matrix across all K classes.

Quadratic Discriminant Analysis (QDA) also assumes the observations within each class are drawn from a multivariate Gaussian distribution with a class-specific mean vector but now each class has its own covariance matrix.

Under this assumption, the Bayes classifier assigns observation $X = x$ to class k for whichever k maximizes

When would we prefer QDA over LDA?

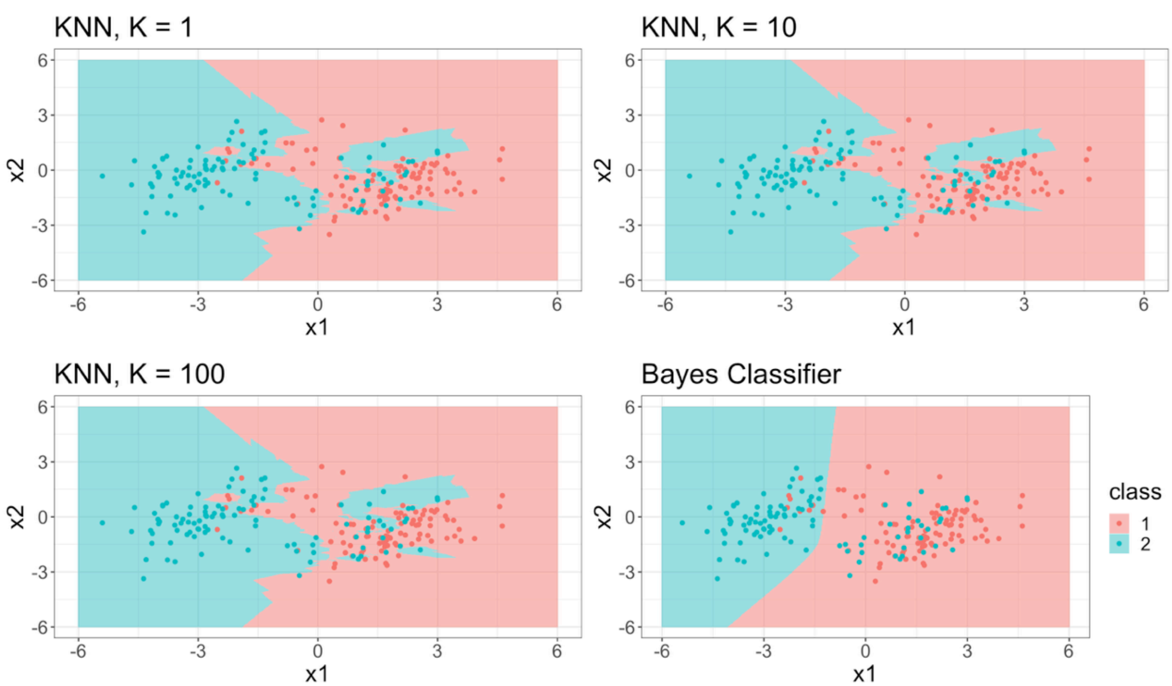


4 KNN

Another method we can use to estimate $P(Y = k|X = x)$ (and thus estimate the Bayes classifier) is through the use of K-nearest neighbors.

The KNN classifier first identifies the K points in the training data that are closest to the test data point $X = x$, called $\mathcal{N}(x)$.

Just as with regression tasks, the choice of K (neighborhood size) has a drastic effect on the KNN classifier obtained.



5 Comparison

LDA vs. Logistic Regression

(LDA & Logistic Regression) vs. KNN

QDA