ML Workshop

Ben Gelderd

2025-10-23

Here's some maths:

$$\log(x^3) \neq \frac{\exp(3x)}{x}.$$

Align subsection

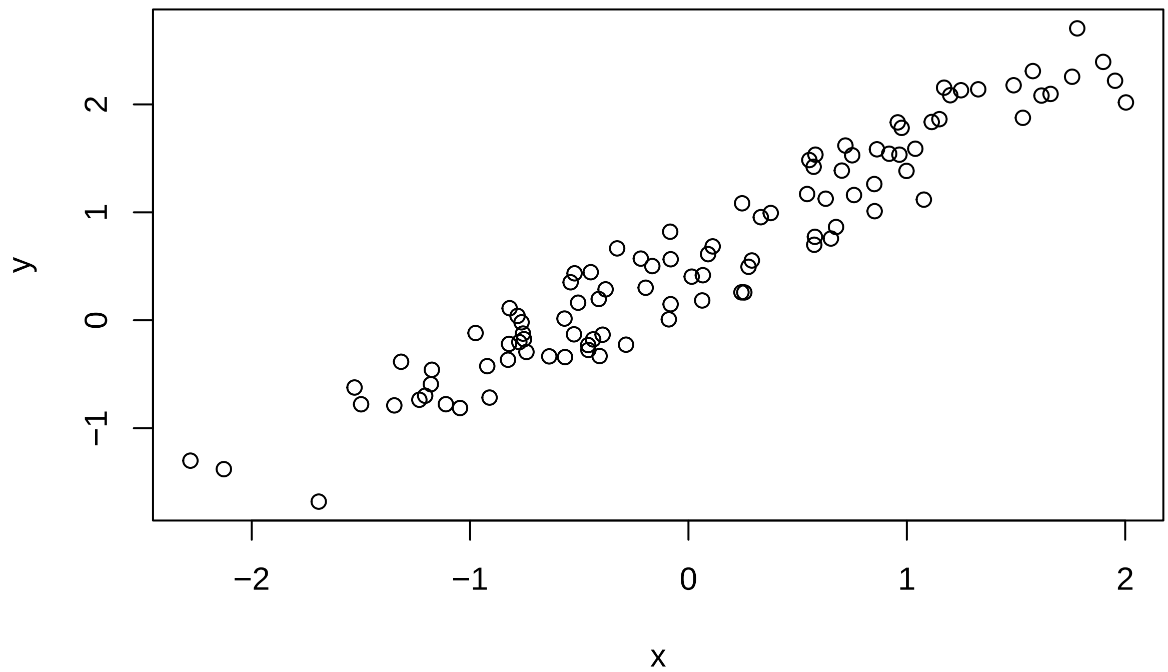
Adding maths in an "align" environment makes it easier to line things up.

$$\begin{aligned}
\mu &\sim \mathrm{N}(0, 1), \\
X_i|\mu &\sim \mathrm{N}(\mu, 1), \quad i = 1, \ldots, n.
\end{aligned}$$

R section

Here's some very simple R code.

```r
x <- rnorm(100)
y <- runif(100) + x
plot(x,y)
```

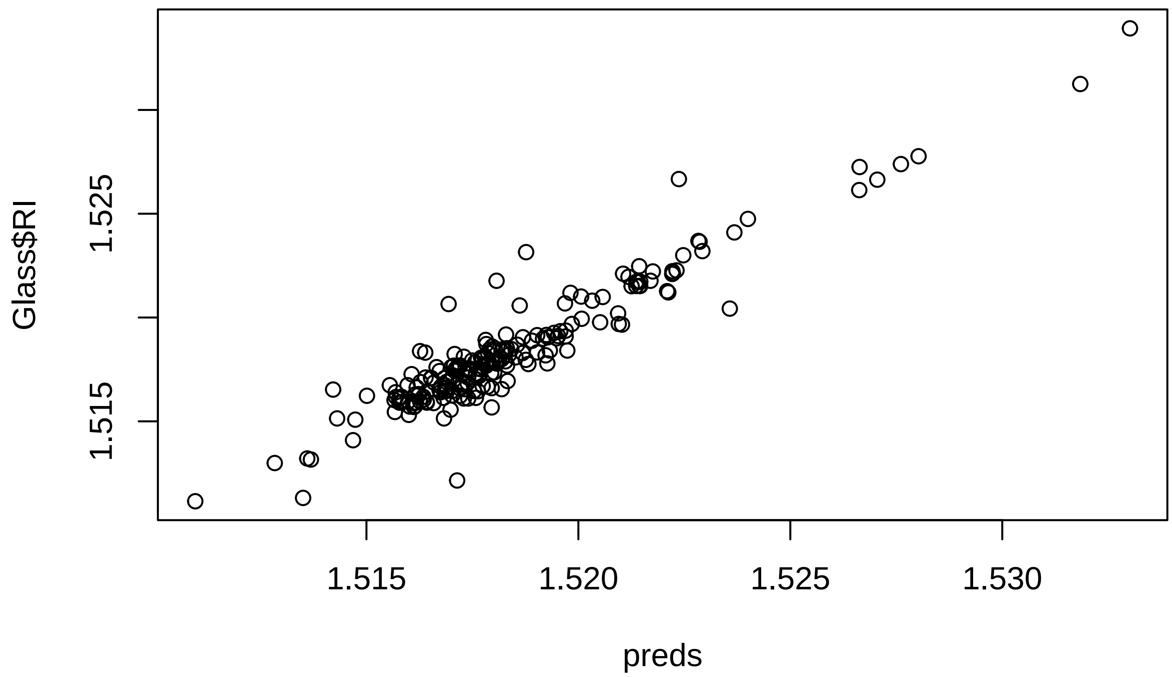



```r
Glass <- read.csv("https://www.maths.dur.ac.uk/users/john.p.gosling/MATH3431_practicals/Glass.csv")
# Fit a linear model
lm1 <- lm(RI ~ . - Type,
          data = Glass)
summary(lm1)
```

```
##
## Call:
## lm(formula = RI ~ . - Type, data = Glass)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0049898 -0.0004273 -0.0000264  0.0004187  0.0043833
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.453e+00  6.704e-02  21.678  < 2e-16 ***
## Na          1.395e-03  6.551e-04   2.130  0.03436 *
## Mg          1.844e-03  6.755e-04   2.730  0.00688 **
## Al          3.262e-05  6.983e-04   0.047  0.96278
## Si          1.685e-04  6.774e-04   0.249  0.80380
## K           1.383e-03  6.900e-04   2.004  0.04636 *
## Ca          3.117e-03  6.684e-04   4.663 5.61e-06 ***
## Ba          2.983e-03  6.760e-04   4.412 1.65e-05 ***
## Fe          4.263e-04  7.787e-04   0.547  0.58468
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.001004 on 205 degrees of freedom
## Multiple R-squared:  0.8948, Adjusted R-squared:  0.8907
## F-statistic: 217.9 on 8 and 205 DF,  p-value: < 2.2e-16
```

```r
preds <- predict(lm1)
plot(preds, Glass$RI)
```




```r
#calculate mean square error and mean absolute error for the model
mean((Glass$RI - preds)**2)
```

```
## [1] 9.657919e-07
```

```r
mean(abs(Glass$RI - preds))
```

```
## [1] 0.0006399008
```

```r
# Fit a linear model
lm2 <- lm(RI ~ Na + Mg + K + Ca + Ba,
          data = Glass)
summary(lm2)
```

```
##
## Call:
## lm(formula = RI ~ Na + Mg + K + Ca + Ba, data = Glass)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0048871 -0.0004520 -0.0000279  0.0004198  0.0043659
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.469e+00  2.270e-03 647.364  < 2e-16 ***
## Na          1.247e-03  1.159e-04  10.758  < 2e-16 ***
## Mg          1.717e-03  8.093e-05  21.221  < 2e-16 ***
## K           1.208e-03  1.360e-04   8.882 3.12e-16 ***
## Ca          2.986e-03  8.102e-05  36.851  < 2e-16 ***
## Ba          2.813e-03  1.803e-04  15.604  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0009985 on 208 degrees of freedom
## Multiple R-squared:  0.8944, Adjusted R-squared:  0.8919
## F-statistic: 352.5 on 5 and 208 DF,  p-value: < 2.2e-16
```

```r
preds2 <- predict(lm2)
mean((Glass$RI - preds2)**2)
```

```
## [1] 9.689946e-07
```

```r
mean(abs(Glass$RI - preds2))
```

```
## [1] 0.0006393474
```

```r
# Fit a linear model
lm3 <- lm(RI ~ Si + Al + Fe,
          data = Glass)
summary(lm3)
```

```
##
## Call:
## lm(formula = RI ~ Si + Al + Fe, data = Glass)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0068121 -0.0011799 -0.0003764  0.0007744  0.0088538
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.6751971  0.0144519 115.915  < 2e-16 ***
## Si          -0.0021111  0.0001986 -10.629  < 2e-16 ***
## Al          -0.0024676  0.0003076  -8.022 7.24e-14 ***
## Fe           0.0019356  0.0015832   1.223    0.223
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002235 on 210 degrees of freedom
## Multiple R-squared:  0.466,  Adjusted R-squared:  0.4584
## F-statistic: 61.08 on 3 and 210 DF,  p-value: < 2.2e-16
```

```
pred3 <- predict(lm3)
mean((Glass$RI - pred3)**2)
```

```
## [1] 4.901921e-06
```

```
mean(abs(Glass$RI-pred3))
```

```
## [1] 0.001525121
```

```
# Fit a linear model without interactions
lm4 <- lm(RI ~ Na + Ba, data = Glass)

# Summarise the model
summary(lm4)
```

```
##
## Call:
## lm(formula = RI ~ Na + Ba, data = Glass)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0072624 -0.0018338 -0.0008541  0.0012016  0.0147547
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.5289942  0.0035380 432.160  < 2e-16 ***
## Na          -0.0007983  0.0002652  -3.010  0.00293 **
## Ba           0.0004258  0.0004356   0.978  0.32941
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002988 on 211 degrees of freedom
## Multiple R-squared:  0.04116,    Adjusted R-squared:  0.03207
## F-statistic: 4.529 on 2 and 211 DF,  p-value: 0.01186
```

```
# Fit a model with the interaction
lm5 <- lm(RI ~ Na*Ba,
          data = Glass)

# Summarise the model
summary(lm5)
```

```
##
## Call:
## lm(formula = RI ~ Na * Ba, data = Glass)
```

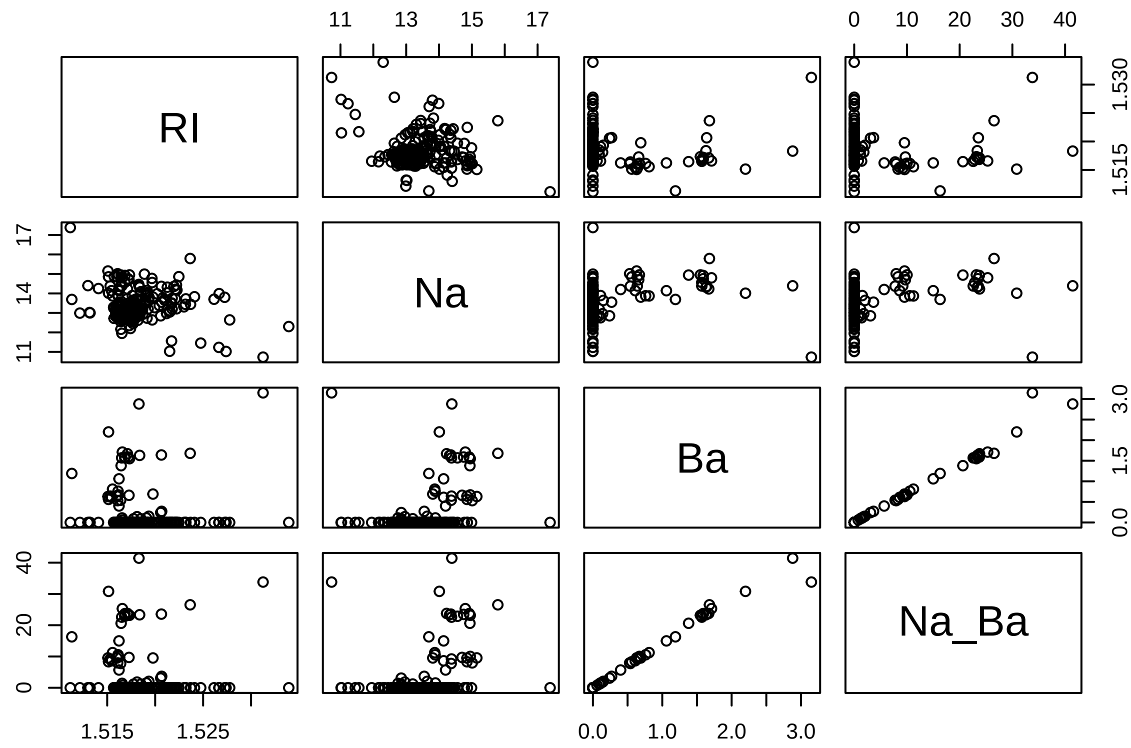
```
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0073689 -0.0018427 -0.0006707  0.0010093  0.0151228
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.5235728  0.0039060 390.061  < 2e-16 ***
## Na          -0.0003874  0.0002935  -1.320  0.18823
## Ba           0.0124291  0.0039871   3.117  0.00208 **
## Na:Ba       -0.0008827  0.0002915  -3.028  0.00277 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002932 on 210 degrees of freedom
## Multiple R-squared:  0.08127,    Adjusted R-squared:  0.06815
## F-statistic: 6.192 on 3 and 210 DF,  p-value: 0.0004739
```

```
'
It is important to consider the affect of one variable on the other ultimately
changing our output for the model thus we must include interactions.
'
```

```
## [1] "\nIt is important to consider the affect of one variable on the other ultimately\nchanging our o
```

```
# Create a proxy for the interaction term
Glass$Na_Ba <- Glass$Na * Glass$Ba

# Look at the relationships between the variables
pairs(Glass[, c("RI", "Na", "Ba", "Na_Ba")])
```




```r
# Load in the data
LetterRecognition <- read.csv("https://www.maths.dur.ac.uk/users/john.p.gosling/MATH3431_practicals/Let

# Look at the first few rows
head(LetterRecognition)
```

```
##   lettr x.box y.box width high onpix x.bar y.bar x2bar y2bar xybar x2ybr xy2br
## 1     T     2     8     3    5     1     8    13     0     6     6    10     8
## 2     I     5    12     3    7     2    10     5     5     4    13     3     9
## 3     D     4    11     6    8     6    10     6     2     6    10     3     7
## 4     N     7    11     6    6     3     5     9     4     6     4     4    10
## 5     G     2     1     3    1     1     8     6     6     6     6     5     9
## 6     S     4    11     5    8     3     8     8     6     9     5     6     6
##   x.ege xegvy y.ege yegvx
## 1     0     8     0     8
## 2     2     8     4    10
## 3     3     7     3     9
## 4     6    10     2     8
## 5     1     7     5    10
## 6     0     8     9     7
```

```r
# Look at the structure of the data
str(LetterRecognition)
```

```
## 'data.frame':    20000 obs. of  17 variables:
```

```
##  $ lettr: chr  "T" "I" "D" "N" ...
##  $ x.box: int  2 5 4 7 2 4 4 1 2 11 ...
##  $ y.box: int  8 12 11 11 1 11 2 1 2 15 ...
##  $ width: int  3 3 6 6 3 5 5 3 4 13 ...
##  $ high : int  5 7 8 6 1 8 4 2 4 9 ...
##  $ onpix: int  1 2 6 3 1 3 4 1 2 7 ...
##  $ x.bar: int  8 10 10 5 8 8 8 8 10 13 ...
##  $ y.bar: int  13 5 6 9 6 8 7 2 6 2 ...
##  $ x2bar: int  0 5 2 4 6 6 6 2 2 6 ...
##  $ y2bar: int  6 4 6 6 6 9 6 2 6 2 ...
##  $ xybar: int  6 13 10 4 6 5 7 8 12 12 ...
##  $ x2ybr: int  10 3 3 4 5 6 6 2 4 1 ...
##  $ xy2br: int  8 9 7 10 9 6 6 8 8 9 ...
##  $ x.ege: int  0 2 3 6 1 0 2 1 1 8 ...
##  $ xegvy: int  8 8 7 10 7 8 8 6 6 1 ...
##  $ y.ege: int  0 4 3 2 5 9 7 2 1 1 ...
##  $ yegvx: int  8 10 9 8 10 7 10 7 7 8 ...
```

```r
# Look at the levels of the letter variable
levels(as.factor(LetterRecognition$lettr))
```

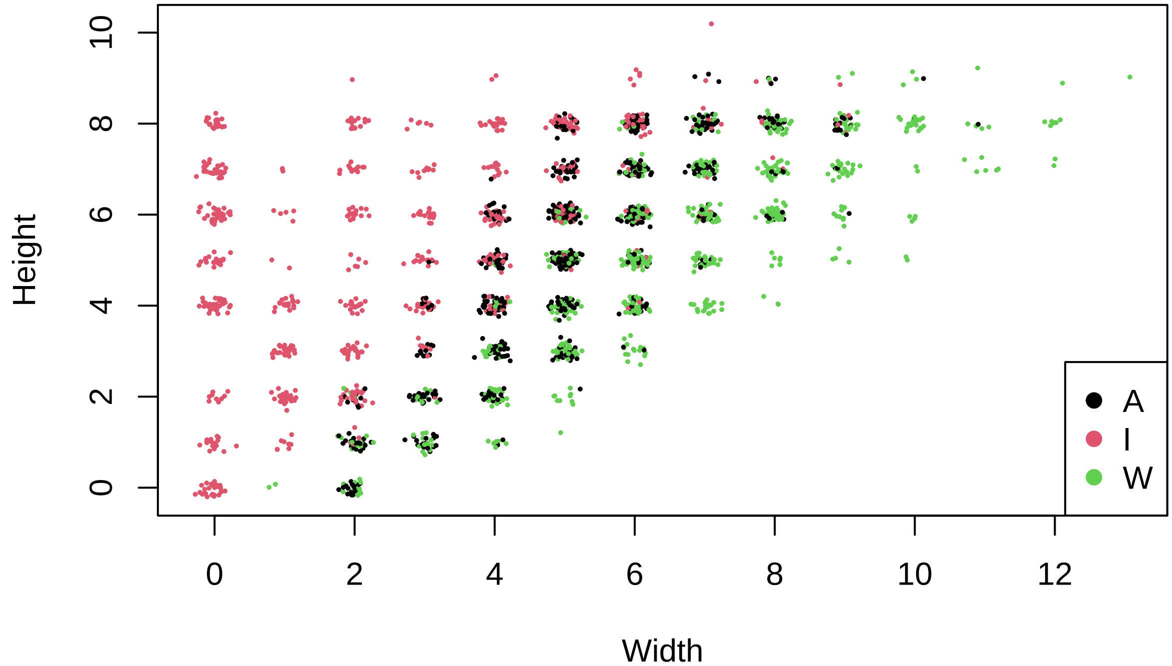
```
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
## [20] "T" "U" "V" "W" "X" "Y" "Z"
```

```r
# Create a subset of the data
ltrs <- subset(LetterRecognition,
               lettr %in% c("I", "A", "W"))

# Reset the levels of the letter variable
ltrs$lettr <- factor(ltrs$lettr)
```

```r
# A jittered scatter plot of the width vs the height with the letters coloured
plot(ltrs$width+rnorm(nrow(ltrs), 0, 0.1),
     ltrs$high+rnorm(nrow(ltrs), 0, 0.1),
     col = as.numeric(ltrs$lettr),
     pch = 19, cex = 0.2,
     xlab = "Width", ylab = "Height")

# Add a legend
legend("bottomright", legend = levels(ltrs$lettr), col = 1:3, pch = 19)
```




```r
# Load the MASS package
library(MASS)

# Fit the LDA model
lda1 <- lda(lettr ~ .,
            data = ltrs)

# Summarise the model
lda1
```

```
## Call:
## lda(lettr ~ ., data = ltrs)
##
## Prior probabilities of groups:
##         A         I         W
## 0.3436411 0.3288328 0.3275261
##
## Group means:
##      x.box    y.box    width     high    onpix    x.bar    y.bar    x2bar
## A 3.337136 6.975919 5.128010 5.178707 2.991128 8.851711 3.631179 2.755387
## I 2.270199 6.980132 2.631788 5.209272 1.825166 7.458278 7.035762 1.940397
## W 5.168883 7.156915 6.486702 5.343085 4.851064 6.078457 9.214096 3.488032
##      y2bar    xybar    x2ybr    xy2br     x.ege     xegvy    y.ege    yegvx
## A 2.043093 7.802281 2.338403 8.465146 2.7718631  6.321926 2.875792 7.468948
## I 5.973510 9.476821 5.797351 7.649007 0.5377483  8.066225 2.141722 7.931126
## W 2.226064 7.574468 8.441489 7.801862 7.5970745 10.375000 1.594415 7.142287
```

```
##
## Coefficients of linear discriminants:
##               LD1          LD2
## x.box -0.07946727 -0.194882075
## y.box -0.03778009  0.012191441
## width  0.15201837  0.526401971
## high  -0.13646197 -0.108057441
## onpix  0.18064087 -0.089799084
## x.bar -0.06167859  0.030193424
## y.bar  0.06560792 -0.057183082
## x2bar -0.03550467 -0.207503443
## y2bar -0.38666661 -0.615744391
## xybar  0.03952814 -0.040605913
## x2ybr  0.19062903 -0.352688732
## xy2br  0.06908132  0.502195140
## x.ege  0.59513534  0.007468394
## xegvy  0.28671786 -0.252754449
## y.ege -0.22168084  0.144032356
## yegvx  0.05901144 -0.456242557
##
## Proportion of trace:
##    LD1    LD2
## 0.6454 0.3546
```

```r
# Make predictions
preds <- predict(lda1)

# Calculate the confusion matrix using the table function
table(ltrs$lettr, preds$class)
```

```
##
##       A   I   W
##   A 754   4  31
##   I  12 743   0
##   W   3   0 749
```

```r
# Fit the LDA model
lda2 <- lda(lettr ~ x.box + y.box + width + high,
            data = ltrs)

# Summarise the model
lda2
```

```
## Call:
## lda(lettr ~ x.box + y.box + width + high, data = ltrs)
##
## Prior probabilities of groups:
##         A         I         W
## 0.3436411 0.3288328 0.3275261
##
## Group means:
##      x.box    y.box    width     high
## A 3.337136 6.975919 5.128010 5.178707
```

```
## I 2.270199 6.980132 2.631788 5.209272
## W 5.168883 7.156915 6.486702 5.343085
##
## Coefficients of linear discriminants:
##               LD1        LD2
## x.box  0.0479466  1.1594755
## y.box -0.1222237 -0.3431877
## width  0.6762307 -0.8194431
## high  -0.2642703  0.3586949
##
## Proportion of trace:
##    LD1    LD2
## 0.8633 0.1367
```

```r
# Make predictions
preds2 <- predict(lda2)

# Calculate the confusion matrix
table(ltrs$lettr, preds2$class)
```

```
##
##       A   I   W
##   A 609  47 133
##   I 151 557  47
##   W 256   5 491
```

```r
# Calculate the overall accuracy
# (hint consider the elements of the confusion matrix)
mean(ltrs$lettr == preds2$class)
```

```
## [1] 0.7216899
```

```r
# Calculate the precision for `A`
precision <- table(ltrs$lettr, preds2$class)[1,1] / sum(table(ltrs$lettr, preds2$class)[,1])
precision
```

```
## [1] 0.5994094
```

```r
# Calculate the recall for `A`
recall <- table(ltrs$lettr, preds2$class)[1,1] / sum(table(ltrs$lettr, preds2$class)[1,])
recall
```

```
## [1] 0.7718631
```

```r
# Fit the LDA model
lda3 <- lda(lettr ~ I(width^0.5) + high,
            data = ltrs)

# Summarise the model
lda3
```

```
## Call:
## lda(lettr ~ I(width^0.5) + high, data = ltrs)
##
## Prior probabilities of groups:
##         A         I         W
## 0.3436411 0.3288328 0.3275261
##
## Group means:
##   I(width^0.5)     high
## A     2.235980 5.178707
## I     1.344486 5.209272
## W     2.512131 5.343085
##
## Coefficients of linear discriminants:
##                     LD1         LD2
## I(width^0.5) -1.9808085 -0.03414421
## high          0.3005129  0.45751432
##
## Proportion of trace:
##    LD1    LD2
## 0.9992 0.0008
```

```r
# Make predictions
preds3 <- predict(lda3)

# Calculate the confusion matrix
table(ltrs$lettr, preds3$class)
```

```
##
##       A   I   W
##   A 634   3 152
##   I 282 450  23
##   W 283   0 469
```

```r
# Calculate the overall accuracy
mean(ltrs$lettr == preds3$class)
```

```
## [1] 0.6763937
```

```r
# Calculate the precision for `A`
table(ltrs$lettr, preds3$class)[1,1] / sum(table(ltrs$lettr, preds3$class)[,1])
```

```
## [1] 0.528774
```

```r
# Calculate the recall for `A`
table(ltrs$lettr, preds3$class)[1,1] / sum(table(ltrs$lettr, preds3$class)[1,])
```

```
## [1] 0.8035488
```