# HW 2

Adon Rosen

Date: 2020-10-14

## Problem 1

Using the following dataset where the class attribute is "Treatment Applied" and using the Decision Tree Induction Algorithm 3.1 given on Page 137 in the textbook, answer the following questions:

### a)

```
# First declare the data
horse.surgery <- c(0,1,1,0,0,1,1,0,0,0,1,1)
horse.pulse <- c(92, 88, 64, 48, 76, 76, 88, 48, 92, 48, 64, 64)
adbominal.distension <- c("None", "Severe", "Severe",
                          "Slight", "Slight", "None",
                          "Severe", "Severe", "Severe",
                          "Slight", "Slight", "Slight")
treat.applied <- c(3, 2, 2, 1, 4, 1, 3, 1, 4, 1, 1, 4)
treat.applied <- paste("Level", treat.applied)
horse.data <- data.frame(horse.surgery, horse.pulse, adbominal.distension, treat.applied)

# Now declare a function which will calculate the entropy of a given set of classes
calc.entropy <- function(tableVal){
  ## This function requires a table of the classes to be classified
  freqs = tableVal/sum(tableVal)
  out.val <- -sum(ifelse(freqs > 0, freqs * log(freqs), 0))
  out.val <- out.val / log(2)
  return(out.val)
}

## Now calculate the entire dataset entropy
entropy.treatment <- calc.entropy(table(horse.data$treat.applied))
# Total dataset entropy == 1.887919

### First explore first round cuts
## First calculate surgery here
entropy.treatment.surgery.no <- calc.entropy(table(horse.data$treat.applied[which(horse.data$horse.surg
entropy.treatment.surgery.yes <- calc.entropy(table(horse.data$treat.applied[which(horse.data$horse.sur
## Now calculate the weighted difference
weigthed.entropy.step1.1 <- (6/12 * entropy.treatment.surgery.no) + (6/12 * entropy.treatment.surgery.y
entropy.gain1.1 <- entropy.treatment - weigthed.entropy.step1.1
## Information gain for surgery == .1991966

## Now try information gain for abdominal distension
entropy.treatment.distenssion.none <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbo
```

```r
entropy.treatment.distenssion.slight <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adb
entropy.treatment.distenssion.severe <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adb
## Now calculate the weighted difference
weigthed.entropy.step1.2 <- (2/12 * entropy.treatment.distenssion.none) + (5/12 * entropy.treatment.dist
entropy.gain1.2 <- entropy.treatment - weigthed.entropy.step1.2
## Information gain here ==.5158857


## Now explore HR with various cutoff points
for(i in unique(horse.data$horse.pulse)){
  horse.data$hrBin <- 0
  horse.data$hrBin[horse.data$horse.pulse>i] <- 1
  entropy.treatment.hr.bin.0 <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hrBin==0)])
  entropy.treatment.hr.bin.1 <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hrBin==1)])
  weigthed.entropy.step1.4 <- (sum(horse.data$hrBin==0)/12 * entropy.treatment.hr.bin.0) + (sum(horse.da
  entropy.gain1.4 <- entropy.treatment - weigthed.entropy.step1.4
  #print(paste(i, ":", entropy.gain1.4))
}
# [1] "92 : NA"
# [1] "88 : 0.253781796468065"
# [1] "64 : 0.302956001949977"
# [1] "48 : 0.406715376769688"
# [1] "76 : 0.522055208874201"

## The first step will be to create a binary variable using heartrate with a cutoff of > 76
## If HR is lower than 76 --> treatment group 1

horse.data$hrBin <- 0
horse.data$hrBin[horse.data$horse.pulse>76] <- 1

## First calculate the total entropy of the pulse <= 76 cohort
entropy.treatment.low.hr <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hrBin==0)]))

## Now calculate entropy for surgery
entropy.treatment.low.hr.surgery.no <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hrBi
entropy.treatment.low.hr.surgery.yes <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hrb

## Now calculate the weighted difference
weigthed.entropy.step2.1 <- (4/8 * entropy.treatment.low.hr.surgery.no) + (4/8 * entropy.treatment.low.h
entropy.gain2.1 <- entropy.treatment.low.hr - weigthed.entropy.step2.1
## Information gain for surgery == 0.1431559

## Now try information gain for abdominal distension
entropy.treatment.low.none <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbominal.dis
entropy.treatment.low.slight <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbominal.d
entropy.treatment.low.severe <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbominal.d

## Now calculate the information gain
weigthed.entropy.step2.2 <- (1/8 * entropy.treatment.low.none) + (5/8 * entropy.treatment.low.slight) +
entropy.gain2.2 <- entropy.treatment.low.hr - weigthed.entropy.step2.2
## Information gain for surgery == 0.4419508

## The second step for those horses who have lower heart rate will be to flag for abdominal distension:
```

```
# for those with slight & none --> level 1 treatment
# for those with Severe abdominal distension --> level 2 treatment

## Now run through the same procedure for the high hr observations
## First calculate the total entropy of the pulse > 76 cohort
entropy.treatment.high.hr <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hrBin==1)]))

## Now calculate entropy for surgery
entropy.treatment.high.hr.surgery.no <- calc.entropy(table(horse.data$treat.applied[which(horse.data$hr
entropy.treatment.high.hr.surgery.yes <- calc.entropy(table(horse.data$treat.applied[which(horse.data$h

## Now calculate the weighted difference
weigthed.entropy.step2.1 <- (2/4 * entropy.treatment.high.hr.surgery.no) + (2/4 * entropy.treatment.high
entropy.gain2.1 <- entropy.treatment.low.hr - weigthed.entropy.step2.1
## Information gain for surgery == 0.5

## Now try information gain for abdominal distension
entropy.treatment.high.none <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbominal.d
entropy.treatment.high.slight <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbominal
entropy.treatment.high.severe <- calc.entropy(table(horse.data$treat.applied[which(horse.data$adbominal

## Now calculate the information gain
weigthed.entropy.step2.2 <- (1/4 * entropy.treatment.high.none) +  (3/4 * entropy.treatment.high.severe
entropy.gain2.2 <- entropy.treatment.low.hr - weigthed.entropy.step2.2
## Information gain for distension == 0.3112781

## The second step for those horses who have higher heart rate will be to flag for surgery:
# for those with surgery --> level 2 treatment
# for those without surgery --> level 4 treatment
```

**Final tree for part a:**

Heart rate <=76

| Distension == none : treatment level 1

| Distension == slight: treatment level 1

| Distension == severe: treatment level 2

Heart rate >76

| Surgery == Yes: treatment level 2

| Surgery == No : Treatment level 4

## b)

```
calc.gini <- function(counts){
  x <- c(1,2,3,4)
  x <- rep(x, counts)
  n <- length(x)
  x <- sort(x)
  n <- length(x)
  x <- sort(x)
  res <- 2 * sum(x * 1:n)/(n * sum(x)) - 1 - (1/n)
```

```
  res <- n/(n - 1) * res
  return(pmax(0, res))
}

## Now calculate the entire dataset gini
gini.treatment <- calc.gini(table(horse.data$treat.applied))
# Total dataset gini == 1.887919

### First explore first round cuts
## First calculate surgery here
gini.treatment.surgery.no <- calc.gini(table(horse.data$treat.applied[which(horse.data$horse.surgery==0
gini.treatment.surgery.yes <- calc.gini(table(horse.data$treat.applied[which(horse.data$horse.surgery==
## Now calculate the weighted difference
weigthed.gini.step1.1 <- (6/12 * gini.treatment.surgery.no) + (6/12 * gini.treatment.surgery.yes)
gini.gain1.1 <- gini.treatment - weigthed.gini.step1.1
## Information gain for surgery == -0.02065342

## Now try information gain for abdominal distension
gini.treatment.distenssion.none <- calc.gini(table(horse.data$treat.applied[which(horse.data$adbominal.
gini.treatment.distenssion.slight <- calc.gini(table(horse.data$treat.applied[which(horse.data$adbominal
gini.treatment.distenssion.severe <- calc.gini(table(horse.data$treat.applied[which(horse.data$adbominal
## Now calculate the weighted difference
weigthed.gini.step1.2 <- (2/12 * gini.treatment.distenssion.none) + (5/12 * gini.treatment.distenssion.
gini.gain1.2 <- gini.treatment - weigthed.gini.step1.2
## Information gain here == -0.04871633


## Now explore HR with various cutoff points
for(i in unique(horse.data$horse.pulse)){
  horse.data$hrBin <- 0
  horse.data$hrBin[horse.data$horse.pulse>i] <- 1
  gini.treatment.hr.bin.0 <- calc.gini(table(horse.data$treat.applied[which(horse.data$hrBin==0)]))
  gini.treatment.hr.bin.1 <- calc.gini(table(horse.data$treat.applied[which(horse.data$hrBin==1)]))
  weigthed.gini.step1.4 <- (sum(horse.data$hrBin==0)/12 * gini.treatment.hr.bin.0) + (sum(horse.data$hrB
  gini.gain1.4 <- gini.treatment - weigthed.gini.step1.4
  print(paste(i, ":", gini.gain1.4))
}
```

```
[1] "92 : NaN"
[1] "88 : 0.0157527657527657"
[1] "64 : 0.0230699148346207"
[1] "48 : 0.123474326599326"
[1] "76 : 0.0234247234247234"
```

```
# [1] "92 : NaN"
# [1] "88 : 0.0157527657527657"
# [1] "64 : 0.0230699148346207"
# [1] "48 : 0.123474326599326"
# [1] "76 : 0.0234247234247234"

## The first step will be to create a binary variable using heartrate with a cutoff of > 49
## If HR is lower than 49 --> treatment group 1

horse.data$hrBin <- 0
```

```
horse.data$hrBin[horse.data$horse.pulse>48] <- 1

## Now run through this same classification procedure for the high hr horses
## First calculate the total entropy of the pulse > 76 cohort
gini.treatment.high.hr <- calc.gini(table(horse.data$treat.applied[which(horse.data$hrBin==1)]))

## Now calculate gini for surgery
gini.treatment.high.hr.surgery.no <- calc.gini(table(horse.data$treat.applied[which(horse.data$hrBin==1
gini.treatment.high.hr.surgery.yes <- calc.gini(table(horse.data$treat.applied[which(horse.data$hrBin==

## Now calculate the weighted difference
weigthed.gini.step2.1 <- (3/9 * gini.treatment.high.hr.surgery.no) + (6/9 * gini.treatment.high.hr.surg
gini.gain2.1 <- gini.treatment.high.hr - weigthed.gini.step2.1
## Information gain for surgery == 0.02514569

## Now try information gain for abdominal distension
gini.treatment.high.none <- calc.gini(table(horse.data$treat.applied[which(horse.data$adbominal.distens
gini.treatment.high.slight <- calc.gini(table(horse.data$treat.applied[which(horse.data$adbominal.diste
gini.treatment.high.severe <- calc.gini(table(horse.data$treat.applied[which(horse.data$adbominal.diste

## Now calculate the information gain
weigthed.gini.step2.2 <- (2/9 * gini.treatment.high.none) +  (3/9 * gini.treatment.high.slight) + (4/9 *
gini.gain2.2 <- gini.treatment.high.hr - weigthed.gini.step2.2
## Information gain for surgery == -0.04566498

## The second step for those horses who have higher heart rate will be to flag for surgery:
# for those with surgery --> level 2 treatment
# for those without surgery --> level 4 treatment
```

**Final tree for part b:**

Heart rate <=48 : treatment level 1

Heart rate >76

| Surgery == Yes: treatment level 2

| Surgery == No : Treatment level 4

## c)

$err_{gen}(T) = err(T) + \Omega \times \frac{k}{N}_{train}$

$err(T) = \frac{7}{12}$

$\Omega = .5$

$k = 5$

$N_{train} = 12$

$err_{gen}(T) = .792$

## d)

$Cost(tree, data) = Cost(tree) + Cost(tree|data)$

$Cost(tree) = log2m_{attributes} + log2k_{classes} = log2(3) + log2(4) = 3.6$

**Entropy**

$Cost(tree) = log2m_{attributes} + log2k_{classes} = log2(3) + log2(4) = 3.6$

$Cost(tree|data) = \sum log2n_{errorInstances} = 17.9$

$MDL_{Entropy} = 21.5$

**GINI**

$Cost(tree) = log2m_{attributes} + log2k_{classes} = log2(2) + log2(4) = 3$

$Cost(tree|data) = \sum log2n_{errorInstances} = 17.9$

$MDL_{GINI} = 20.9$

The MDL principle suggests the GINI tree is a more efficient method for approximation. This is because while the error rates are equivalent acorss the two methods (7/12) as well as the number of classes to identify (k=4) the GINI index only requires 2 attributes (k=2), whereas the Entropy method requires 3 (k=3), thus the GINI has a lower MDL of 20.9 to the Entropy's 21.5, a difference of .6 bits.

# Problem 2

## 1)

The Iterative Dichotomiser 3 (ID3) algorithm and also the Classification and Regression Tree (CART) are two tree based approaches that are used to classify and for the latter cases they can also be used for regression. The ID3 approach stems from work performed in the 1980's by J.R. Quinlan whom introduced the technique as a method to classify noisy and incomplete data (1985). The CART approach was contemporaneously developed by Breiman as an alternative to parametric regression techniques which were the predominant analytic technique of the time period for the behavioral sciences (1984).

The ID3 algorithm is an iterative procedure which explores the entirety of a parameter space and builds a decision based on information gain, typically it uses entropy or a similar metric to measure information gain. In the original paper written by Quinlan, he explicitly described the information gain procedure using entropy (1985). The benefits of this algorithm are that it guarantees an solution; however, it is stated that this may not be the most optimal solution possible as the tree building procedure.

The CART approach as described Breiman is similar in practice to the ID3 algorithm in that it is trying to classify observations based on a tree based rule structure, although one clear advantage of this suite of tools is their ability to work with continuous data. The algorithm works in a similar manner wherein the algorithm is attempting to minimize the total Gini impurity within the data. The Gini impurity measures how equally distributed a variable is within a set of observations.

Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81–106

Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

## 2)

**a)**

```
# Load the data
in.data <- read.csv('../Data/wine.csv')
# Now make a plot of the Magnesium; Color Intensity; and Malic Acid
out.plot.standard <- in.data[,c("Magnesium", "Color_intensity", "Malic_acid")] %>%
  mutate(across(where(is.numeric), scale)) %>%
  reshape2::melt(.) %>%
```

```
  ggplot(., aes(x=variable, y=value)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("z-score values boxplot")
```

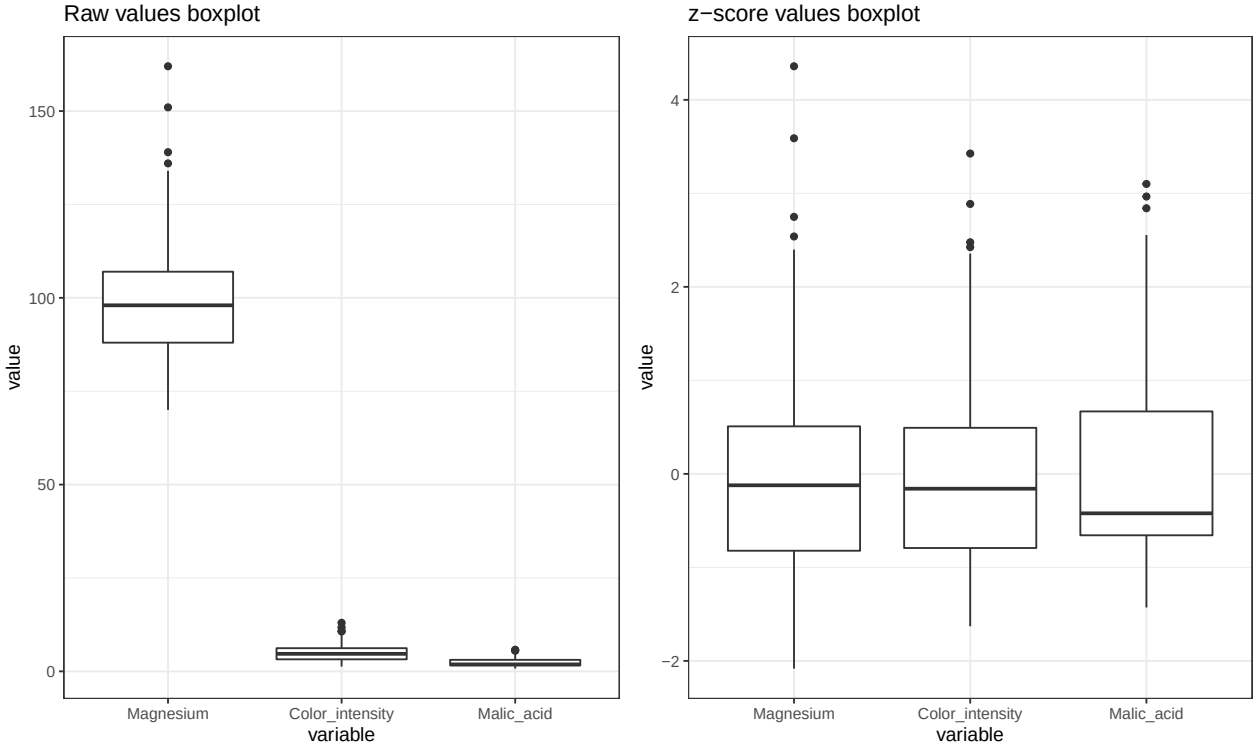
```
No id variables; using all as measure variables
```

```
Warning: attributes are not identical across measure variables; they will be
dropped
```

```
out.plot.raw <- in.data[,c("Magnesium", "Color_intensity", "Malic_acid")] %>%
  #mutate(across(where(is.numeric), scale)) %>%
  reshape2::melt(.) %>%
  ggplot(., aes(x=variable, y=value)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Raw values boxplot")
```

```
No id variables; using all as measure variables
```

```
multiplot(out.plot.raw, out.plot.standard, cols = 2)
```




**b)**

It appears all three of the plotted variables have observations that are greater than 1.5 times the interquartile range. Interestingly they are all in the same direction such that the outliers are larger than the reported means.

```
## Now identify the outliers
out_ind1 <- which(in.data$Magnesium %in% boxplot.stats(in.data$Magnesium)$out)
out_ind2 <- which(in.data$Color_intensity %in% boxplot.stats(in.data$Color_intensity)$out)
out_ind3 <- which(in.data$Malic_acid %in% boxplot.stats(in.data$Malic_acid)$out)
```

```
out_ind <- union(out_ind1, out_ind2)
out_ind <- union(out_ind, out_ind3)
## Remove the outliers from the data
long_dt <- in.data[-out_ind,]
## Now print the data frame
long_dt_print <- long_dt
colnames(long_dt_print) <- substring(names(long_dt_print), 5)
kable(long_dt_print,
  format    = "latex",
  longtable = T,
  digits = 3) %>%
  kable_styling(full_width = FALSE)
```

| | ivar | hol | c_acid | | linity_of_ash | esium | l_phenols | anoids | lavanoid_phenols | nthocyanins | r_i |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 14.23 | 1.71 | 2.43 | 15.6 | 127 | 2.80 | 3.06 | 0.28 | 2.29 | |
| 2 | 1 | 13.20 | 1.78 | 2.14 | 11.2 | 100 | 2.65 | 2.76 | 0.26 | 1.28 | |
| 3 | 1 | 13.16 | 2.36 | 2.67 | 18.6 | 101 | 2.80 | 3.24 | 0.30 | 2.81 | |
| 4 | 1 | 14.37 | 1.95 | 2.50 | 16.8 | 113 | 3.85 | 3.49 | 0.24 | 2.18 | |
| 5 | 1 | 13.24 | 2.59 | 2.87 | 21.0 | 118 | 2.80 | 2.69 | 0.39 | 1.82 | |
| 6 | 1 | 14.20 | 1.76 | 2.45 | 15.2 | 112 | 3.27 | 3.39 | 0.34 | 1.97 | |
| 7 | 1 | 14.39 | 1.87 | 2.45 | 14.6 | 96 | 2.50 | 2.52 | 0.30 | 1.98 | |
| 8 | 1 | 14.06 | 2.15 | 2.61 | 17.6 | 121 | 2.60 | 2.51 | 0.31 | 1.25 | |
| 9 | 1 | 14.83 | 1.64 | 2.17 | 14.0 | 97 | 2.80 | 2.98 | 0.29 | 1.98 | |
| 10 | 1 | 13.86 | 1.35 | 2.27 | 16.0 | 98 | 2.98 | 3.15 | 0.22 | 1.85 | |
| 11 | 1 | 14.10 | 2.16 | 2.30 | 18.0 | 105 | 2.95 | 3.32 | 0.22 | 2.38 | |
| 12 | 1 | 14.12 | 1.48 | 2.32 | 16.8 | 95 | 2.20 | 2.43 | 0.26 | 1.57 | |
| 13 | 1 | 13.75 | 1.73 | 2.41 | 16.0 | 89 | 2.60 | 2.76 | 0.29 | 1.81 | |
| 14 | 1 | 14.75 | 1.73 | 2.39 | 11.4 | 91 | 3.10 | 3.69 | 0.43 | 2.81 | |
| 15 | 1 | 14.38 | 1.87 | 2.38 | 12.0 | 102 | 3.30 | 3.64 | 0.29 | 2.96 | |
| 16 | 1 | 13.63 | 1.81 | 2.70 | 17.2 | 112 | 2.85 | 2.91 | 0.30 | 1.46 | |
| 17 | 1 | 14.30 | 1.92 | 2.72 | 20.0 | 120 | 2.80 | 3.14 | 0.33 | 1.97 | |
| 18 | 1 | 13.83 | 1.57 | 2.62 | 20.0 | 115 | 2.95 | 3.40 | 0.40 | 1.72 | |
| 19 | 1 | 14.19 | 1.59 | 2.48 | 16.5 | 108 | 3.30 | 3.93 | 0.32 | 1.86 | |
| 20 | 1 | 13.64 | 3.10 | 2.56 | 15.2 | 116 | 2.70 | 3.03 | 0.17 | 1.66 | |
| 21 | 1 | 14.06 | 1.63 | 2.28 | 16.0 | 126 | 3.00 | 3.17 | 0.24 | 2.10 | |
| 22 | 1 | 12.93 | 3.80 | 2.65 | 18.6 | 102 | 2.41 | 2.41 | 0.25 | 1.98 | |
| 23 | 1 | 13.71 | 1.86 | 2.36 | 16.6 | 101 | 2.61 | 2.88 | 0.27 | 1.69 | |
| 24 | 1 | 12.85 | 1.60 | 2.52 | 17.8 | 95 | 2.48 | 2.37 | 0.26 | 1.46 | |
| 25 | 1 | 13.50 | 1.81 | 2.61 | 20.0 | 96 | 2.53 | 2.61 | 0.28 | 1.66 | |
| 26 | 1 | 13.05 | 2.05 | 3.22 | 25.0 | 124 | 2.63 | 2.68 | 0.47 | 1.92 | |
| 27 | 1 | 13.39 | 1.77 | 2.62 | 16.1 | 93 | 2.85 | 2.94 | 0.34 | 1.45 | |
| 28 | 1 | 13.30 | 1.72 | 2.14 | 17.0 | 94 | 2.40 | 2.19 | 0.27 | 1.35 | |
| 29 | 1 | 13.87 | 1.90 | 2.80 | 19.4 | 107 | 2.95 | 2.97 | 0.37 | 1.76 | |
| 30 | 1 | 14.02 | 1.68 | 2.21 | 16.0 | 96 | 2.65 | 2.33 | 0.26 | 1.98 | |
| 31 | 1 | 13.73 | 1.50 | 2.70 | 22.5 | 101 | 3.00 | 3.25 | 0.29 | 2.38 | |
| 32 | 1 | 13.58 | 1.66 | 2.36 | 19.1 | 106 | 2.86 | 3.19 | 0.22 | 1.95 | |
| 33 | 1 | 13.68 | 1.83 | 2.36 | 17.2 | 104 | 2.42 | 2.69 | 0.42 | 1.97 | |
| 34 | 1 | 13.76 | 1.53 | 2.70 | 19.5 | 132 | 2.95 | 2.74 | 0.50 | 1.35 | |
| 35 | 1 | 13.51 | 1.80 | 2.65 | 19.0 | 110 | 2.35 | 2.53 | 0.29 | 1.54 | |
| 36 | 1 | 13.48 | 1.81 | 2.41 | 20.5 | 100 | 2.70 | 2.98 | 0.26 | 1.86 | |
| 37 | 1 | 13.28 | 1.64 | 2.84 | 15.5 | 110 | 2.60 | 2.68 | 0.34 | 1.36 | |
| 38 | 1 | 13.05 | 1.65 | 2.55 | 18.0 | 98 | 2.45 | 2.43 | 0.29 | 1.44 | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 39 | 1 | 13.07 | 1.50 | 2.10 | 15.5 | 98 | 2.40 | 2.64 | 0.28 | 1.37 | |
| 40 | 1 | 14.22 | 3.99 | 2.51 | 13.2 | 128 | 3.00 | 3.04 | 0.20 | 2.08 | |
| 41 | 1 | 13.56 | 1.71 | 2.31 | 16.2 | 117 | 3.15 | 3.29 | 0.34 | 2.34 | |
| 42 | 1 | 13.41 | 3.84 | 2.12 | 18.8 | 90 | 2.45 | 2.68 | 0.27 | 1.48 | |
| 43 | 1 | 13.88 | 1.89 | 2.59 | 15.0 | 101 | 3.25 | 3.56 | 0.17 | 1.70 | |
| 44 | 1 | 13.24 | 3.98 | 2.29 | 17.5 | 103 | 2.64 | 2.63 | 0.32 | 1.66 | |
| 45 | 1 | 13.05 | 1.77 | 2.10 | 17.0 | 107 | 3.00 | 3.00 | 0.28 | 2.03 | |
| 46 | 1 | 14.21 | 4.04 | 2.44 | 18.9 | 111 | 2.85 | 2.65 | 0.30 | 1.25 | |
| 47 | 1 | 14.38 | 3.59 | 2.28 | 16.0 | 102 | 3.25 | 3.17 | 0.27 | 2.19 | |
| 48 | 1 | 13.90 | 1.68 | 2.12 | 16.0 | 101 | 3.10 | 3.39 | 0.21 | 2.14 | |
| 49 | 1 | 14.10 | 2.02 | 2.40 | 18.8 | 103 | 2.75 | 2.92 | 0.32 | 2.38 | |
| 50 | 1 | 13.94 | 1.73 | 2.27 | 17.4 | 108 | 2.88 | 3.54 | 0.32 | 2.08 | |
| 51 | 1 | 13.05 | 1.73 | 2.04 | 12.4 | 92 | 2.72 | 3.27 | 0.17 | 2.91 | |
| 52 | 1 | 13.83 | 1.65 | 2.60 | 17.2 | 94 | 2.45 | 2.99 | 0.22 | 2.29 | |
| 53 | 1 | 13.82 | 1.75 | 2.42 | 14.0 | 111 | 3.88 | 3.74 | 0.32 | 1.87 | |
| 54 | 1 | 13.77 | 1.90 | 2.68 | 17.1 | 115 | 3.00 | 2.79 | 0.39 | 1.68 | |
| 55 | 1 | 13.74 | 1.67 | 2.25 | 16.4 | 118 | 2.60 | 2.90 | 0.21 | 1.62 | |
| 56 | 1 | 13.56 | 1.73 | 2.46 | 20.5 | 116 | 2.96 | 2.78 | 0.20 | 2.45 | |
| 57 | 1 | 14.22 | 1.70 | 2.30 | 16.3 | 118 | 3.20 | 3.00 | 0.26 | 2.03 | |
| 58 | 1 | 13.29 | 1.97 | 2.68 | 16.8 | 102 | 3.00 | 3.23 | 0.31 | 1.66 | |
| 59 | 1 | 13.72 | 1.43 | 2.50 | 16.7 | 108 | 3.40 | 3.67 | 0.19 | 2.04 | |
| 60 | 2 | 12.37 | 0.94 | 1.36 | 10.6 | 88 | 1.98 | 0.57 | 0.28 | 0.42 | |
| 61 | 2 | 12.33 | 1.10 | 2.28 | 16.0 | 101 | 2.05 | 1.09 | 0.63 | 0.41 | |
| 62 | 2 | 12.64 | 1.36 | 2.02 | 16.8 | 100 | 2.02 | 1.41 | 0.53 | 0.62 | |
| 63 | 2 | 13.67 | 1.25 | 1.92 | 18.0 | 94 | 2.10 | 1.79 | 0.32 | 0.73 | |
| 64 | 2 | 12.37 | 1.13 | 2.16 | 19.0 | 87 | 3.50 | 3.10 | 0.19 | 1.87 | |
| 65 | 2 | 12.17 | 1.45 | 2.53 | 19.0 | 104 | 1.89 | 1.75 | 0.45 | 1.03 | |
| 66 | 2 | 12.37 | 1.21 | 2.56 | 18.1 | 98 | 2.42 | 2.65 | 0.37 | 2.08 | |
| 67 | 2 | 13.11 | 1.01 | 1.70 | 15.0 | 78 | 2.98 | 3.18 | 0.26 | 2.28 | |
| 68 | 2 | 12.37 | 1.17 | 1.92 | 19.6 | 78 | 2.11 | 2.00 | 0.27 | 1.04 | |
| 69 | 2 | 13.34 | 0.94 | 2.36 | 17.0 | 110 | 2.53 | 1.30 | 0.55 | 0.42 | |
| 71 | 2 | 12.29 | 1.61 | 2.21 | 20.4 | 103 | 1.10 | 1.02 | 0.37 | 1.46 | |
| 72 | 2 | 13.86 | 1.51 | 2.67 | 25.0 | 86 | 2.95 | 2.86 | 0.21 | 1.87 | |
| 73 | 2 | 13.49 | 1.66 | 2.24 | 24.0 | 87 | 1.88 | 1.84 | 0.27 | 1.03 | |
| 75 | 2 | 11.96 | 1.09 | 2.30 | 21.0 | 101 | 3.38 | 2.14 | 0.13 | 1.65 | |
| 76 | 2 | 11.66 | 1.88 | 1.92 | 16.0 | 97 | 1.61 | 1.57 | 0.34 | 1.15 | |
| 77 | 2 | 13.03 | 0.90 | 1.71 | 16.0 | 86 | 1.95 | 2.03 | 0.24 | 1.46 | |
| 78 | 2 | 11.84 | 2.89 | 2.23 | 18.0 | 112 | 1.72 | 1.32 | 0.43 | 0.95 | |
| 80 | 2 | 12.70 | 3.87 | 2.40 | 23.0 | 101 | 2.83 | 2.55 | 0.43 | 1.95 | |
| 81 | 2 | 12.00 | 0.92 | 2.00 | 19.0 | 86 | 2.42 | 2.26 | 0.30 | 1.43 | |
| 82 | 2 | 12.72 | 1.81 | 2.20 | 18.8 | 86 | 2.20 | 2.53 | 0.26 | 1.77 | |
| 83 | 2 | 12.08 | 1.13 | 2.51 | 24.0 | 78 | 2.00 | 1.58 | 0.40 | 1.40 | |
| 84 | 2 | 13.05 | 3.86 | 2.32 | 22.5 | 85 | 1.65 | 1.59 | 0.61 | 1.62 | |
| 85 | 2 | 11.84 | 0.89 | 2.58 | 18.0 | 94 | 2.20 | 2.21 | 0.22 | 2.35 | |
| 86 | 2 | 12.67 | 0.98 | 2.24 | 18.0 | 99 | 2.20 | 1.94 | 0.30 | 1.46 | |
| 87 | 2 | 12.16 | 1.61 | 2.31 | 22.8 | 90 | 1.78 | 1.69 | 0.43 | 1.56 | |
| 88 | 2 | 11.65 | 1.67 | 2.62 | 26.0 | 88 | 1.92 | 1.61 | 0.40 | 1.34 | |
| 89 | 2 | 11.64 | 2.06 | 2.46 | 21.6 | 84 | 1.95 | 1.69 | 0.48 | 1.35 | |
| 90 | 2 | 12.08 | 1.33 | 2.30 | 23.6 | 70 | 2.20 | 1.59 | 0.42 | 1.38 | |
| 91 | 2 | 12.08 | 1.83 | 2.32 | 18.5 | 81 | 1.60 | 1.50 | 0.52 | 1.64 | |
| 92 | 2 | 12.00 | 1.51 | 2.42 | 22.0 | 86 | 1.45 | 1.25 | 0.50 | 1.63 | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 93 | 2 | 12.69 | 1.53 | 2.26 | 20.7 | 80 | 1.38 | 1.46 | 0.58 | 1.62 | |
| 94 | 2 | 12.29 | 2.83 | 2.22 | 18.0 | 88 | 2.45 | 2.25 | 0.25 | 1.99 | |
| 95 | 2 | 11.62 | 1.99 | 2.28 | 18.0 | 98 | 3.02 | 2.26 | 0.17 | 1.35 | |
| 97 | 2 | 11.81 | 2.12 | 2.74 | 21.5 | 134 | 1.60 | 0.99 | 0.14 | 1.56 | |
| 98 | 2 | 12.29 | 1.41 | 1.98 | 16.0 | 85 | 2.55 | 2.50 | 0.29 | 1.77 | |
| 99 | 2 | 12.37 | 1.07 | 2.10 | 18.5 | 88 | 3.52 | 3.75 | 0.24 | 1.95 | |
| 100 | 2 | 12.29 | 3.17 | 2.21 | 18.0 | 88 | 2.85 | 2.99 | 0.45 | 2.81 | |
| 101 | 2 | 12.08 | 2.08 | 1.70 | 17.5 | 97 | 2.23 | 2.17 | 0.26 | 1.40 | |
| 102 | 2 | 12.60 | 1.34 | 1.90 | 18.5 | 88 | 1.45 | 1.36 | 0.29 | 1.35 | |
| 103 | 2 | 12.34 | 2.45 | 2.46 | 21.0 | 98 | 2.56 | 2.11 | 0.34 | 1.31 | |
| 104 | 2 | 11.82 | 1.72 | 1.88 | 19.5 | 86 | 2.50 | 1.64 | 0.37 | 1.42 | |
| 105 | 2 | 12.51 | 1.73 | 1.98 | 20.5 | 85 | 2.20 | 1.92 | 0.32 | 1.48 | |
| 106 | 2 | 12.42 | 2.55 | 2.27 | 22.0 | 90 | 1.68 | 1.84 | 0.66 | 1.42 | |
| 107 | 2 | 12.25 | 1.73 | 2.12 | 19.0 | 80 | 1.65 | 2.03 | 0.37 | 1.63 | |
| 108 | 2 | 12.72 | 1.75 | 2.28 | 22.5 | 84 | 1.38 | 1.76 | 0.48 | 1.63 | |
| 109 | 2 | 12.22 | 1.29 | 1.94 | 19.0 | 92 | 2.36 | 2.04 | 0.39 | 2.08 | |
| 110 | 2 | 11.61 | 1.35 | 2.70 | 20.0 | 94 | 2.74 | 2.92 | 0.29 | 2.49 | |
| 111 | 2 | 11.46 | 3.74 | 1.82 | 19.5 | 107 | 3.18 | 2.58 | 0.24 | 3.58 | |
| 112 | 2 | 12.52 | 2.43 | 2.17 | 21.0 | 88 | 2.55 | 2.27 | 0.26 | 1.22 | |
| 113 | 2 | 11.76 | 2.68 | 2.92 | 20.0 | 103 | 1.75 | 2.03 | 0.60 | 1.05 | |
| 114 | 2 | 11.41 | 0.74 | 2.50 | 21.0 | 88 | 2.48 | 2.01 | 0.42 | 1.44 | |
| 115 | 2 | 12.08 | 1.39 | 2.50 | 22.5 | 84 | 2.56 | 2.29 | 0.43 | 1.04 | |
| 116 | 2 | 11.03 | 1.51 | 2.20 | 21.5 | 85 | 2.46 | 2.17 | 0.52 | 2.01 | |
| 117 | 2 | 11.82 | 1.47 | 1.99 | 20.8 | 86 | 1.98 | 1.60 | 0.30 | 1.53 | |
| 118 | 2 | 12.42 | 1.61 | 2.19 | 22.5 | 108 | 2.00 | 2.09 | 0.34 | 1.61 | |
| 119 | 2 | 12.77 | 3.43 | 1.98 | 16.0 | 80 | 1.63 | 1.25 | 0.43 | 0.83 | |
| 120 | 2 | 12.00 | 3.43 | 2.00 | 19.0 | 87 | 2.00 | 1.64 | 0.37 | 1.87 | |
| 121 | 2 | 11.45 | 2.40 | 2.42 | 20.0 | 96 | 2.90 | 2.79 | 0.32 | 1.83 | |
| 122 | 2 | 11.56 | 2.05 | 3.23 | 28.5 | 119 | 3.18 | 5.08 | 0.47 | 1.87 | |
| 123 | 2 | 12.42 | 4.43 | 2.73 | 26.5 | 102 | 2.20 | 2.13 | 0.43 | 1.71 | |
| 125 | 2 | 11.87 | 4.31 | 2.39 | 21.0 | 82 | 2.86 | 3.03 | 0.21 | 2.91 | |
| 126 | 2 | 12.07 | 2.16 | 2.17 | 21.0 | 85 | 2.60 | 2.65 | 0.37 | 1.35 | |
| 127 | 2 | 12.43 | 1.53 | 2.29 | 21.5 | 86 | 2.74 | 3.15 | 0.39 | 1.77 | |
| 128 | 2 | 11.79 | 2.13 | 2.78 | 28.5 | 92 | 2.13 | 2.24 | 0.58 | 1.76 | |
| 129 | 2 | 12.37 | 1.63 | 2.30 | 24.5 | 88 | 2.22 | 2.45 | 0.40 | 1.90 | |
| 130 | 2 | 12.04 | 4.30 | 2.38 | 22.0 | 80 | 2.10 | 1.75 | 0.42 | 1.35 | |
| 131 | 3 | 12.86 | 1.35 | 2.32 | 18.0 | 122 | 1.51 | 1.25 | 0.21 | 0.94 | |
| 132 | 3 | 12.88 | 2.99 | 2.40 | 20.0 | 104 | 1.30 | 1.22 | 0.24 | 0.83 | |
| 133 | 3 | 12.81 | 2.31 | 2.40 | 24.0 | 98 | 1.15 | 1.09 | 0.27 | 0.83 | |
| 134 | 3 | 12.70 | 3.55 | 2.36 | 21.5 | 106 | 1.70 | 1.20 | 0.17 | 0.84 | |
| 135 | 3 | 12.51 | 1.24 | 2.25 | 17.5 | 85 | 2.00 | 0.58 | 0.60 | 1.25 | |
| 136 | 3 | 12.60 | 2.46 | 2.20 | 18.5 | 94 | 1.62 | 0.66 | 0.63 | 0.94 | |
| 137 | 3 | 12.25 | 4.72 | 2.54 | 21.0 | 89 | 1.38 | 0.47 | 0.53 | 0.80 | |
| 139 | 3 | 13.49 | 3.59 | 2.19 | 19.5 | 88 | 1.62 | 0.48 | 0.58 | 0.88 | |
| 140 | 3 | 12.84 | 2.96 | 2.61 | 24.0 | 101 | 2.32 | 0.60 | 0.53 | 0.81 | |
| 141 | 3 | 12.93 | 2.81 | 2.70 | 21.0 | 96 | 1.54 | 0.50 | 0.53 | 0.75 | |
| 142 | 3 | 13.36 | 2.56 | 2.35 | 20.0 | 89 | 1.40 | 0.50 | 0.37 | 0.64 | |
| 143 | 3 | 13.52 | 3.17 | 2.72 | 23.5 | 97 | 1.55 | 0.52 | 0.50 | 0.55 | |
| 144 | 3 | 13.62 | 4.95 | 2.35 | 20.0 | 92 | 2.00 | 0.80 | 0.47 | 1.02 | |
| 145 | 3 | 12.25 | 3.88 | 2.20 | 18.5 | 112 | 1.38 | 0.78 | 0.29 | 1.14 | |
| 146 | 3 | 13.16 | 3.57 | 2.15 | 21.0 | 102 | 1.50 | 0.55 | 0.43 | 1.30 | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 147 | 3 | 13.88 | 5.04 | 2.23 | 20.0 | 80 | 0.98 | 0.34 | 0.40 | 0.68 | |
| 148 | 3 | 12.87 | 4.61 | 2.48 | 21.5 | 86 | 1.70 | 0.65 | 0.47 | 0.86 | |
| 149 | 3 | 13.32 | 3.24 | 2.38 | 21.5 | 92 | 1.93 | 0.76 | 0.45 | 1.25 | |
| 150 | 3 | 13.08 | 3.90 | 2.36 | 21.5 | 113 | 1.41 | 1.39 | 0.34 | 1.14 | |
| 151 | 3 | 13.50 | 3.12 | 2.62 | 24.0 | 123 | 1.40 | 1.57 | 0.22 | 1.25 | |
| 153 | 3 | 13.11 | 1.90 | 2.75 | 25.5 | 116 | 2.20 | 1.28 | 0.26 | 1.56 | |
| 154 | 3 | 13.23 | 3.30 | 2.28 | 18.5 | 98 | 1.80 | 0.83 | 0.61 | 1.87 | |
| 155 | 3 | 12.58 | 1.29 | 2.10 | 20.0 | 103 | 1.48 | 0.58 | 0.53 | 1.40 | |
| 156 | 3 | 13.17 | 5.19 | 2.32 | 22.0 | 93 | 1.74 | 0.63 | 0.61 | 1.55 | |
| 157 | 3 | 13.84 | 4.12 | 2.38 | 19.5 | 89 | 1.80 | 0.83 | 0.48 | 1.56 | |
| 158 | 3 | 12.45 | 3.03 | 2.64 | 27.0 | 97 | 1.90 | 0.58 | 0.63 | 1.14 | |
| 161 | 3 | 12.36 | 3.83 | 2.38 | 21.0 | 88 | 2.30 | 0.92 | 0.50 | 1.04 | |
| 162 | 3 | 13.69 | 3.26 | 2.54 | 20.0 | 107 | 1.83 | 0.56 | 0.50 | 0.80 | |
| 163 | 3 | 12.85 | 3.27 | 2.58 | 22.0 | 106 | 1.65 | 0.60 | 0.60 | 0.96 | |
| 164 | 3 | 12.96 | 3.45 | 2.35 | 18.5 | 106 | 1.39 | 0.70 | 0.40 | 0.94 | |
| 165 | 3 | 13.78 | 2.76 | 2.30 | 22.0 | 90 | 1.35 | 0.68 | 0.41 | 1.03 | |
| 166 | 3 | 13.73 | 4.36 | 2.26 | 22.5 | 88 | 1.28 | 0.47 | 0.52 | 1.15 | |
| 167 | 3 | 13.45 | 3.70 | 2.60 | 23.0 | 111 | 1.70 | 0.92 | 0.43 | 1.46 | |
| 168 | 3 | 12.82 | 3.37 | 2.30 | 19.5 | 88 | 1.48 | 0.66 | 0.40 | 0.97 | |
| 169 | 3 | 13.58 | 2.58 | 2.69 | 24.5 | 105 | 1.55 | 0.84 | 0.39 | 1.54 | |
| 170 | 3 | 13.40 | 4.60 | 2.86 | 25.0 | 112 | 1.98 | 0.96 | 0.27 | 1.11 | |
| 171 | 3 | 12.20 | 3.03 | 2.32 | 19.0 | 96 | 1.25 | 0.49 | 0.40 | 0.73 | |
| 172 | 3 | 12.77 | 2.39 | 2.28 | 19.5 | 86 | 1.39 | 0.51 | 0.48 | 0.64 | |
| 173 | 3 | 14.16 | 2.51 | 2.48 | 20.0 | 91 | 1.68 | 0.70 | 0.44 | 1.24 | |
| 175 | 3 | 13.40 | 3.91 | 2.48 | 23.0 | 102 | 1.80 | 0.75 | 0.43 | 1.41 | |
| 176 | 3 | 13.27 | 4.28 | 2.26 | 20.0 | 120 | 1.59 | 0.69 | 0.43 | 1.35 | |
| 177 | 3 | 13.17 | 2.59 | 2.37 | 20.0 | 120 | 1.65 | 0.68 | 0.53 | 1.46 | |
| 178 | 3 | 14.13 | 4.10 | 2.74 | 24.5 | 96 | 2.05 | 0.76 | 0.56 | 1.35 | |

**c)**

```
# Now make a plot of the Magnesium; Color Intensity; and Malic Acid
out.plot.standard <- long_dt[,c("Magnesium", "Color_intensity", "Malic_acid")] %>%
  mutate(across(where(is.numeric), scale)) %>%
  reshape2::melt(.) %>%
  ggplot(., aes(x=variable, y=value)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("z-score values boxplot")
```

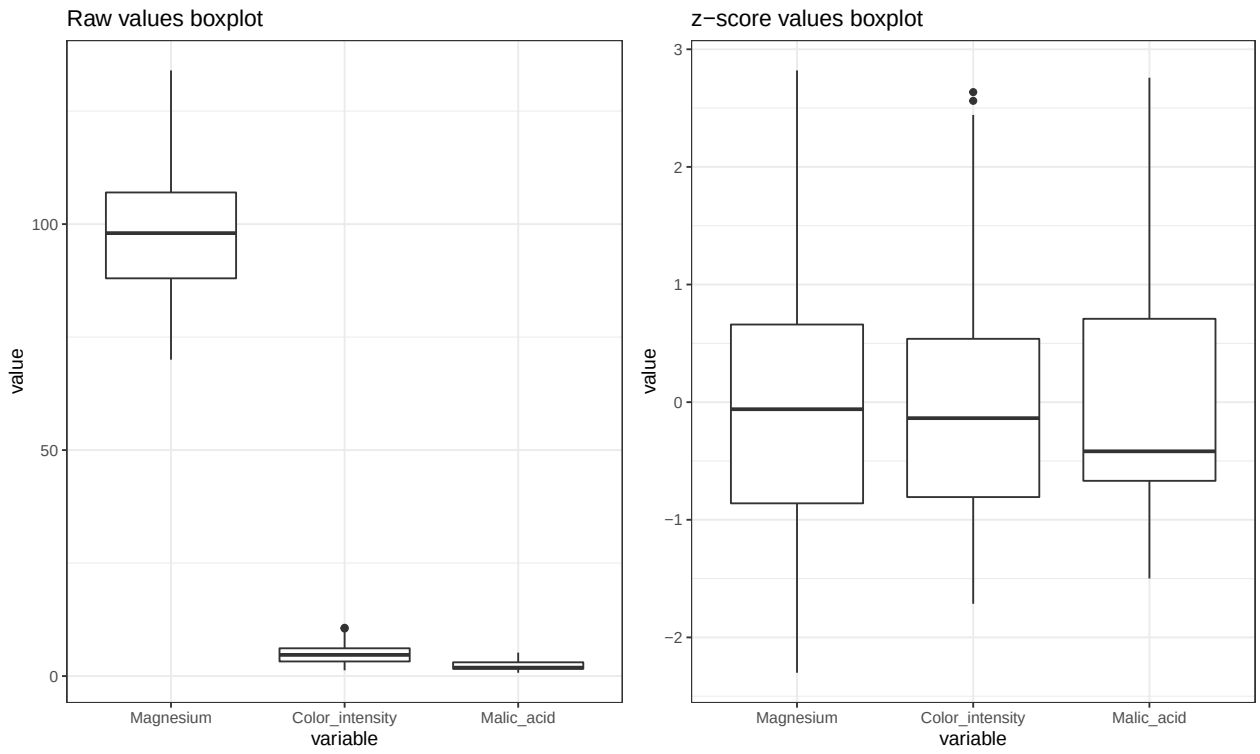
```
No id variables; using all as measure variables
```

```
Warning: attributes are not identical across measure variables; they will be
dropped
```

```
out.plot.raw <- long_dt[,c("Magnesium", "Color_intensity", "Malic_acid")] %>%
  #mutate(across(where(is.numeric), scale)) %>%
  reshape2::melt(.) %>%
  ggplot(., aes(x=variable, y=value)) +
  geom_boxplot() +
  theme_bw() +
  ggtitle("Raw values boxplot")
```

```
No id variables; using all as measure variables
```

```
multiplot(out.plot.raw, out.plot.standard, cols = 2)
```

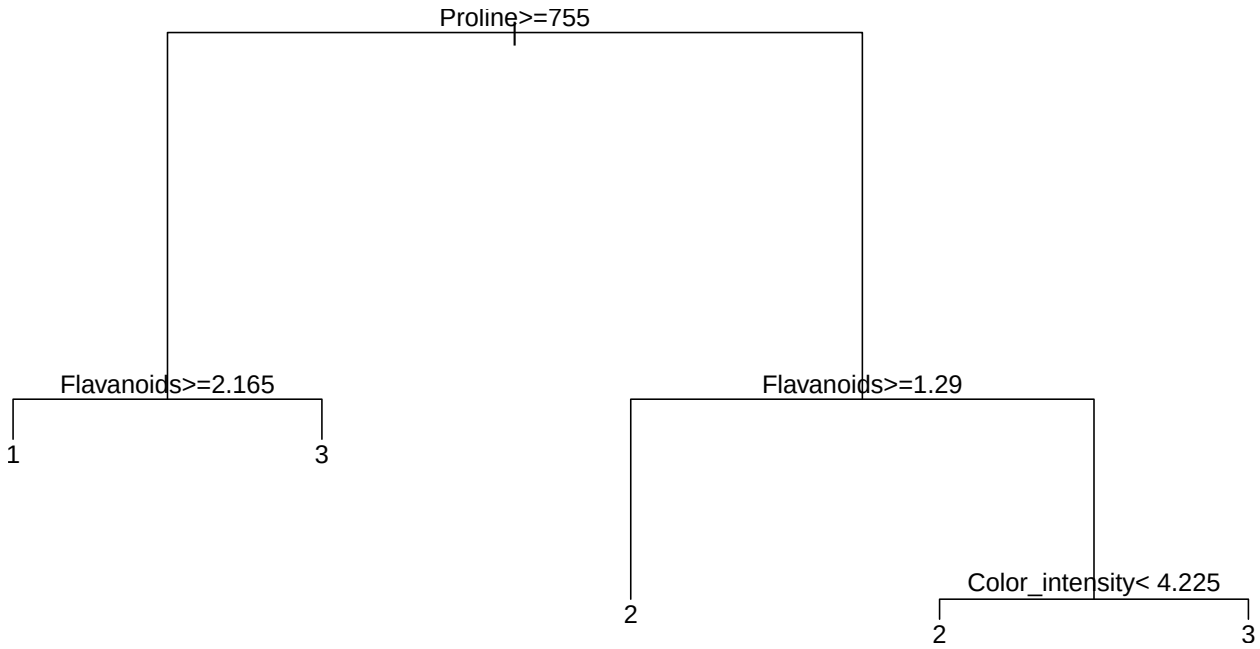



Seen above it would appear the intensity variable still posses minor outliers, although these all do appear to have much more satisfactory distributions. Malic acid does display a little positive skewness; whereas, both magnesium and color intensity appear to be more normally distributed.

**d)**

```
long_dt$Cultivar <- factor(long_dt$Cultivar)
mod.1 <- rpart::rpart(Cultivar ~., data=long_dt)

## Now plot the tree
plot(mod.1)
text(mod.1)
```




```
## Now run through a cross validation using these same data with k=5
# First create the folds
set.seed(16)
folds <- caret::createFolds(long_dt$Cultivar, k = 5)

## Now prepare the output
confusion.matrix.out <- list()
model.error <- list()
## Now run a loop where in each loop a model is trained
## and then tested in the left out sample
for(i in 1:length(folds)){
  ## First train the model
  mod.tmp <- rpart::rpart(Cultivar~., data=long_dt[-folds[[i]],])
  ## Now get the confusion matrix
  pred.vals <- predict(mod.tmp, newdata = long_dt[folds[[i]],], type='class')
  true.vals <- long_dt[folds[[i]],"Cultivar"]
  confusion.matrix <- table(pred.vals, true.vals)
  ## Now get the error
  total.vals <- sum(confusion.matrix)
  correct.vals <- sum(diag(confusion.matrix))
  model.error.tmp <- 1 - correct.vals/total.vals
  ## Now output these values
  print(confusion.matrix)
  confusion.matrix.out[[i]] <- confusion.matrix
  model.error[[i]] <- model.error.tmp
}
```

```
         true.vals
pred.vals  1  2  3
        1 11  0  0
        2  1 11  1
        3  0  2  8
         true.vals
pred.vals  1  2  3
```

```
        1 11  1  0
        2  0 11  3
        3  0  1  6
         true.vals
pred.vals  1  2  3
        1 12  0  0
        2  0 13  1
        3  0  0  8
         true.vals
pred.vals  1  2  3
        1 11  0  0
        2  0 12  0
        3  1  1  8
         true.vals
pred.vals  1  2  3
        1 11  0  0
        2  1 13  2
        3  0  1  6
```

```
model.error1 <- model.error
```

e)

```
ID3 <- make_Weka_classifier("weka/classifiers/trees/Id3")
long_dt$Cultivar <- factor(long_dt$Cultivar)

## ID3 requires nominal attributes so I am going to
## convert all of my values into factors based on quartiles
long_dt_id3 <- long_dt
long_dt_id3[,2:14] <- apply(long_dt_id3[,2:14], 2, function(x)cut(x, breaks = 4))
for(i in 2:14){long_dt_id3[,i] <- factor(long_dt_id3[,i])}
mod.2 <- ID3(`Cultivar` ~ . , data=long_dt_id3)

## Now run through a cross validation using these same data with k=5
# First create the folds
set.seed(16)
folds <- caret::createFolds(long_dt_id3$Cultivar, k = 5)

## Now prepare the output
confusion.matrix.out <- list()
model.error <- list()
## Now run a loop where in each loop a model is trained
## and then tested in the left out sample
for(i in 1:length(folds)){
  ## First train the model
  mod.tmp <- ID3(Cultivar~., data=long_dt_id3[-folds[[i]],])
  ## Now get the confusion matrix
  pred.vals <- predict(mod.tmp, newdata = long_dt_id3[folds[[i]],], type='class')
  true.vals <- long_dt[folds[[i]],"Cultivar"]
  confusion.matrix <- table(pred.vals, true.vals)
  ## Now get the error
  total.vals <- sum(confusion.matrix)
  correct.vals <- sum(diag(confusion.matrix))
  model.error.tmp <- 1 - correct.vals/total.vals
```

```
  ## Now output these values
  print(confusion.matrix)
  confusion.matrix.out[[i]] <- confusion.matrix
  model.error[[i]] <- model.error.tmp
}
```

```
         true.vals
pred.vals  1  2  3
        1 10  0  0
        2  2 13  0
        3  0  0  9
         true.vals
pred.vals  1  2  3
        1 10  0  0
        2  0 13  0
        3  0  0  9
         true.vals
pred.vals  1  2  3
        1 11  0  0
        2  1 13  2
        3  0  0  7
         true.vals
pred.vals  1  2  3
        1 10  0  0
        2  2 12  0
        3  0  0  8
         true.vals
pred.vals  1  2  3
        1 12  1  0
        2  0 11  0
        3  0  1  8
```

```
model.error2 <- model.error
```

f)

```
# First obtain the model mean errors from the 5-fold cross validation
mean.model.error.1 <- mean(unlist(model.error1))
mean.model.error.2 <- mean(unlist(model.error2))


## Now use these errors to estimate the error varaince
model.error.varaince = mean.model.error.1*(1-mean.model.error.1)/dim(long_dt)[1] + mean.model.error.2*(
model.difference = mean.model.error.1 - mean.model.error.2

## Now calculate the confidence interval
upper.bound = model.difference+2.58*sqrt(model.error.varaince)
lower.bound = model.difference-2.58*sqrt(model.error.varaince)


## Now return this value
out.string <- paste("The estimated model difference is ", model.difference, "with a lower bound 99% con
print(out.string)
```

[1] "The estimated model difference is 0.0413324420677362 with a lower bound 99% confidence interval of -0.0324466948293731 and an upper bound 99% confidence interval estimate of 0.115111578964846"

```
out.string2 <- paste("Because this confidence interval includes 0 within it's boundaries it leads us to
print(out.string2)
```

[1] "Because this confidence interval includes 0 within it's boundaries it leads us to conclude that there is no significant difference between the error of these models"

g)

```
## Now return the predicted class from in an input tuple
input.tuple.1  <- c(NA, apply(long_dt[,2:14], 2, function(x) sample(x, 1)))
input.tuple.1 <- as.data.frame(t(input.tuple.1))
## Now print the tuple to kable
input.tuple.1 %>%
  kable(., ,
            format    = "latex",
            longtable = T,
            digits = 3) %>%
  kable_styling(full_width = FALSE)
```

| V1 | Alcohol | Malic_acid | Ash | Alcalinity_of_ash | Magnesium | Total_phenols | Flavanoids | Nonflavanoid_phen |
|---|---|---|---|---|---|---|---|---|
| NA | 12.96 | 1.51 | 2.64 | 21 | 100 | 2.86 | 3.03 | 0 |

```
## Now predict the class for this tuple
pred.class <- predict(mod.1, input.tuple.1, type='class')
## Now print this
print(paste("Predicted class input 1:", pred.class))
```

[1] "Predicted class input 1: 1"

```
input.tuple.2  <- c(NA, apply(long_dt[,2:14], 2, function(x) sample(x, 1)))
input.tuple.2 <- as.data.frame(t(input.tuple.2))
## Now print the tuple to kable
input.tuple.2 %>%
  kable(., ,
        format    = "latex",
        longtable = T,
        digits = 3) %>%
  kable_styling(full_width = FALSE)
```

| V1 | Alcohol | Malic_acid | Ash | Alcalinity_of_ash | Magnesium | Total_phenols | Flavanoids | Nonflavanoid_phen |
|---|---|---|---|---|---|---|---|---|
| NA | 12.43 | 3.59 | 2.25 | 16 | 96 | 3.25 | 2.9 | 0 |

```
## Now predict the class for this tuple
pred.class <- predict(mod.1, input.tuple.2, type='class')
## Now print this
print(paste("Predicted class input 2:", pred.class))
```

[1] "Predicted class input 2: 1"

```
input.tuple.3  <- c(NA, apply(long_dt[,2:14], 2, function(x) sample(x, 1)))
input.tuple.3 <- as.data.frame(t(input.tuple.3))
## Now print the tuple to kable
```

```
input.tuple.3 %>%
  kable(., ,
        format    = "latex",
        longtable = T,
        digits = 3) %>%
  kable_styling(full_width = FALSE)
```

| V1 | Alcohol | Malic_acid | Ash | Alcalinity_of_ash | Magnesium | Total_phenols | Flavanoids | Nonflavanoid_phen |
|---|---|---|---|---|---|---|---|---|
| NA | 13.84 | 1.71 | 2.51 | 24.5 | 96 | 3.2 | 2 | ( |

```
## Now predict the class for this tuple
pred.class <- predict(mod.1, input.tuple.3, type='class')
## Now print this
print(paste("Predicted class input 3:", pred.class))
```

[1] "Predicted class input 3: 3"