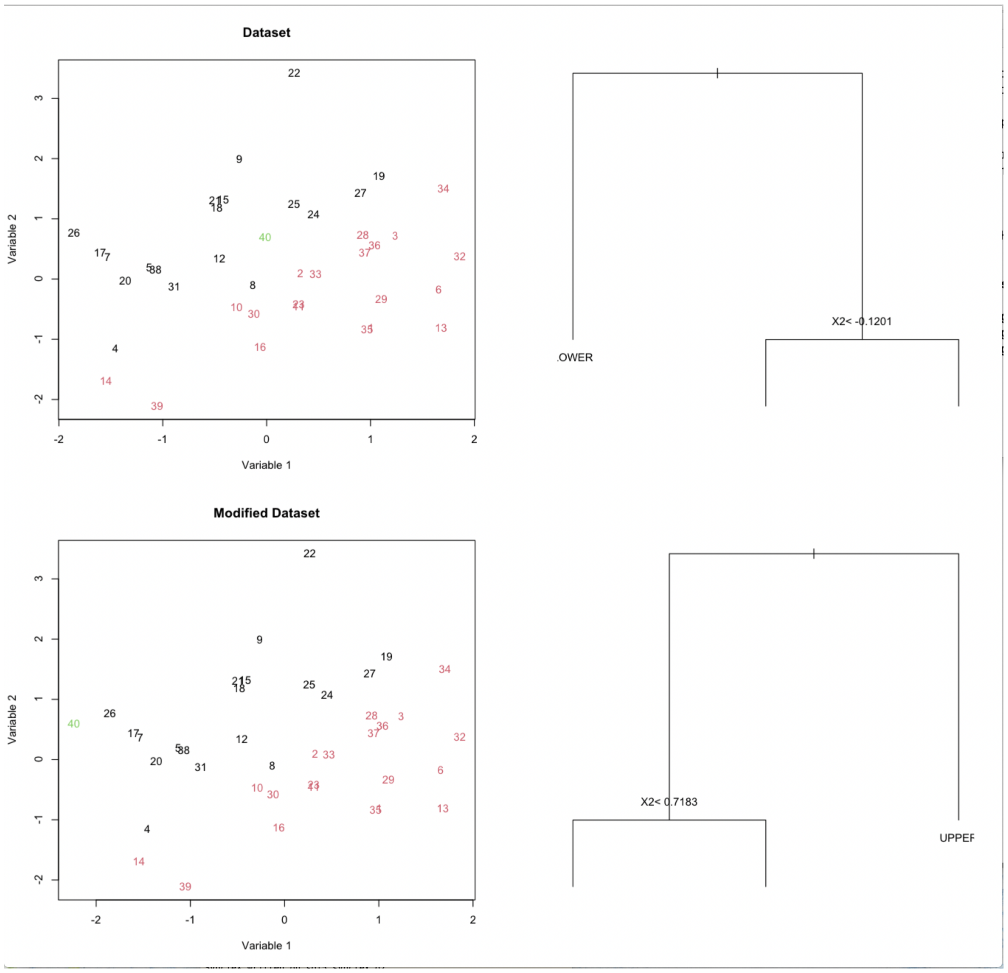
Elements of Statistical Learning
Solution Sheet 5 (rev3: 23rd May 2021)

1. Below is a suggested function. It essentially tries a number of possibilities and, on each iteration, you can compare the movement of the (last) point and the tree that is constructed. Here, the coordinates of the last point change, but not its class membership.

```
fig.q1 <-
function (n=40, pch=16)
{

# Generate same random sequence each time for reproducibility
set.seed(123)

# Arrange plot to have two rows and two cols
oldpar <- par(mfrow=c(2,2))


for(i in 1:10) {

new.seed <- sample(1:10000, 1)

cat("new seed is: ", new.seed, "\n")
set.seed(new.seed)


# Generate two dimensional synthetic data
x <- matrix(rnorm(n*2), nrow=n, ncol=2)

# Rename x variables for data frame
X1 <- x[,1]
X2 <- x[,2]

# Calculate synthetic response variable (linear)
y <- as.numeric(X1 - X2 > 0) + 1

# Create a fictitious factor
yfactor <- factor(c("UPPER", "LOWER"))

# Create new factor the same length as y but replacing 0
# and 1 by the factor levels
ynew <- yfactor[y]

# Create data frame out of X1, X2 and y
the.df <- data.frame(ynew=ynew, X1=X1, X2=X2)
# Fit tree
the.tree1 <- rpart(ynew~X1+X2, data=the.df)

# Plot the data
plot(X1, X2, type="n", xlab="Variable 1", ylab="Variable 2",
    main="Dataset")
text(X1[1:(n-1)], X2[1:(n-1)], lab=1:(n-1), col=y, pch=pch)
text(X1[n], X2[n], col=3, lab=n)

# Plot the tree
plot(the.tree1)
text(the.tree1)

# Now, this is the key part of the question
# Move point 2 down a bit
X1[n] <- X1[n] + rnorm(1, mean=0, sd=2)
X2[n] <- X2[n] + rnorm(1, mean=0, sd=2)

# Recalculate y, ynew and new dataframe
#y2 <- as.numeric(X1 - X2 > 0) + 1
#ynew2 <- yfactor[y2]
the.df2 <- data.frame(ynew=ynew, X1=X1, X2=X2)

# Plot the modified config
plot(X1, X2, type="n", xlab="Variable 1", ylab="Variable 2",
    main="Modified Dataset")
text(X1[1:(n-1)], X2[1:(n-1)], lab=1:(n-1), col=y, pch=pch)
text(X1[n], X2[n], col=3, lab=n)

# Compute and plot the new tree
the.tree2 <- rpart(ynew~X1+X2, data=the.df2)

plot(the.tree2)
text(the.tree2)
# Get user to input character ("Hit key to continue")
scan()
        }
}
```

The plot below shows what happens on the second iteration. Point 40 has moved some way to the left and the resulting trees are different.



Dataset
Variable 1
Variable 2
X2< -0.1201
.OWER
Modified Dataset
Variable 1
Variable 2
X2< 0.7183
UPPEF

2. Here is a function that computes the trimmed mean and generates bootstrap estimates of its standard error.

```
fig.q2 <-
function(x=c(1.2, 3.5, 4.7, 7.3, 8.6, 12.4, 13.8, 18.1), B=25){

nx <- length(x)

my.trim.mean <- function(x) {

sort.x <- sort(x)
trim.x <- sort.x[c(-1, -2, -7, -8)]
return(mean(trim.x))

}

cat("Trimmed mean of sample is: ", my.trim.mean(x), "\n")


Bans <- rep(0, B)

for(i in 1:B) {

BS.sample <- sample(x=x, size=nx, replace = TRUE)

Bans[i] <- my.trim.mean(BS.sample)
}


return(sd(Bans))
}
```

   (a) I make the trimmed mean equal to 8.25.

   (b) The values of my bootstrap estimate of standard error are

B	25	100	200	500	1000	2000
$\hat{\text{se}}_B$	2.742	2.321	2.272	2.576	2.380	2.489

   These are given to three decimal places and, of course, depend on the state of my random number generator. Yours might be similar. IF you plot these, and perhaps for higher, they should start tending towards 2.46ish.

   (c) Here's a short R programme to do this 100 times for each B:

```
> fig.q2a
function(nsims=100){

B <- c(25 , 100 , 200 , 500 , 1000 , 2000)
lB <- length(B)

mans <- matrix(0, nrow=nsims, ncol=lB)
dimnames(mans)[[2]] <- as.character(B)

for(b in 1:lB) {

for(i in 1:nsims)
mans[i, b] <- fig.q2(B=B[b])

}

return(mans)
}
```

If you run `boxplot(fig.q2a(nsims=100))` you get the plot in Figure 1. It's kind of clear that the bootstrap, even for small B is estimating the standard error well on average, but might be a bit variable. It depends what kind of tolerance you are looking for, but $B = 2000$ looks reasonable and within about 10% of the value.

3. Let $x_1, ..., x_n$ be iid Gaussian random variables with $x_i \sim N(\mu, \sigma^2)$. It can be noted in the chapter that for $\bar{x}_i^* := n^{-1}\sum_{j=1}^n x_{i,j}$, where $x_{i,j}$ are randomly drawn with replacement from our Gaussian random variables. We show below that the covariance can be calculated as:

$$\mathrm{cov}(x_{i,j}, x_{\ell,k}) = \sigma^2 \mathbb{P}(x_{i,j}, x_{\ell,k}) = n^{-1}\sigma^2. \tag{1}$$

Furthermore, it is then clear that $\mathrm{cov}(\bar{x}_1^*, \bar{x}_2^*) = \frac{\sigma^2}{n}$. It is also necessary to note from the covariance that

$$\mathrm{var}(\bar{x}_1^*) = \frac{1}{n^2}\sum_{i=1}^n \sum_{j=1}^n \mathrm{cov}(x_{1,i}, x_{1,j}) = \frac{2n-1}{n^2}\sigma^2.$$

It is trivial to show that $\mathrm{var}(\bar{x}_2^*)$ has the same formula. Combining this and noting the general formula for correlation, ρ,

$$\rho_{\bar{x}_1^*, \bar{x}_2^*} = \frac{\mathrm{cov}(\bar{x}_1^*, \bar{x}_2^*)}{\{\mathrm{var}(\bar{x}_1^*)\,\mathrm{var}(\bar{x}_2^*)\}^{1/2}} = \frac{\sigma^2/n}{(2n-1)\sigma^2/n^2} = \frac{n}{2n-1}.$$

Finally, it can be seen that the variance of $\bar{x}_{\text{bag}}$,

$$\mathrm{var}(\bar{x}_{\text{bag}}) = \mathrm{var}\left\{(\bar{x}_1^* + \bar{x}_2^*)/2\right\} = \frac{3n-1}{2n^2}\sigma^2.$$

Covariance between two bootstrap variates from different bootstrap samples. Let us know look in more detail at how we might obtain the covariance in (1).

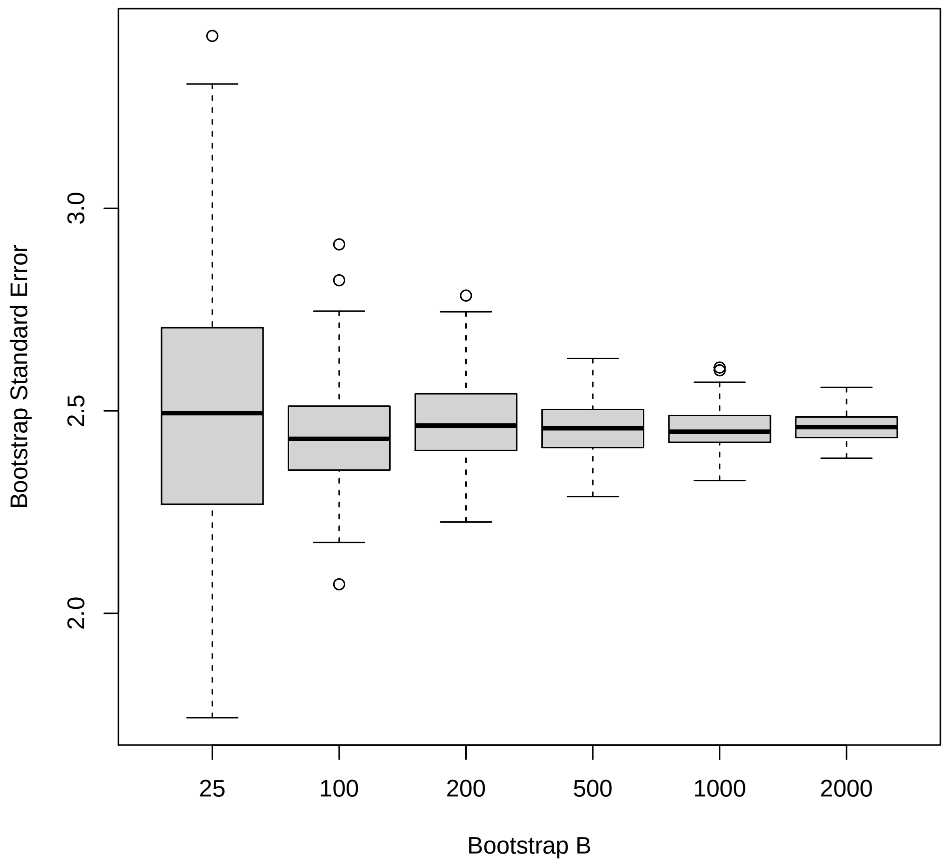


Figure 1: Boxplot of bootstrap estimate of standard error for different bootstrap lengths.

Let us apply the standard formula for covariance:

$$\begin{aligned} \operatorname{cov}(x_{i,j}, x_{\ell,k}) &= \mathbb{E}(x_{i,j}x_{\ell,k}) - \mathbb{E}(x_{i,j})\mathbb{E}(x_{\ell,k}) && (2) \\ &= \mathbb{E}(x_{i,j}x_{\ell,k}) - \mu^2, && (3) \end{aligned}$$

since each individual variable has expectation μ. Let us now consider the expectation of the product $x_{i,j}x_{l,k}$, but pause to understand what these can be.

$$\begin{aligned} x_{i,j} &: \text{the } j\text{th element of the } i\text{th bootstrap sequence.} && (4) \\ x_{\ell,k} &: \text{the } k\text{th element of the } \ell\text{th bootstrap sequence.} && (5) \end{aligned}$$

Bootstrap sample i is a sample of length n with replacement from $x_1, \ldots, x_n$. Bootstrap sample ℓ is another sample of length n with replacement from $x_1, \ldots, x_n$. So, $x_{i,j}$ and $x_{\ell,k}$ *could* be the same actual value x_m from $x_1, \ldots, x_n$ or they could be different.

To analyse this problem further, we need to separate out the cases for when the bootstrap sample $x_{i,j}$ and $x_{\ell,k}$ are the same or different. This is tricky to do 'all in one go', so we break down the problem by introducing a further random variable, which will just be a simple indicator function (or Kronecker delta, if you like) defined to be

$$\delta_{j,k}^{(i,\ell)} = \begin{cases} 1, & x_{i,j} = x_{\ell,k}, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

We will condition on the indicator variable to obtain the required expectation of the product and use the law of total expectation (or the tower property). Hence

$$\begin{aligned} \mathbb{E}(x_{i,j}x_{\ell,k}) =& \mathbb{E}\{\mathbb{E}(x_{i,j}x_{\ell,k}|\delta_{j,k}^{(i,\ell)})\} && (7) \\ =& \mathbb{E}(x_{i,j}x_{\ell,k}|\delta_{j,k}^{(i,\ell)} = 1)\mathbb{P}(\delta_{j,k}^{(i,\ell)} = 1) && (8) \\ &+ \mathbb{E}(x_{i,j}x_{\ell,k}|\delta_{j,k}^{(i,\ell)} = 0)\mathbb{P}(\delta_{j,k}^{(i,\ell)} = 0), && (9) \end{aligned}$$

In the first case, (8), if $\delta_{j,k}^{(i,\ell)} = 1$, then the variates $x_{i,j}$ and $x_{\ell,k}$ are the same variable and so $\mathbb{E}(x_{i,j}x_{\ell,k}|\delta_{j,k}^{(i,\ell)} = 1) = \mathbb{E}(x_j^2) = \operatorname{var}(x_j) + \mathbb{E}(x_j)^2 = \sigma^2 + \mu^2$. The variables are the same with probability n^{-1} (which can be obtained by similar conditioning arguments).

In the second case, (9), where $\delta_{j,k}^{(i,\ell)} = 0$, the variates $x_{i,j}$ and $x_{\ell,k}$ are different random variables, hence independent and so $\mathbb{E}(x_{i,j}x_{\ell,k}|\delta_{j,k}^{(i,\ell)} = 0) = \mathbb{E}(x_{i,j})\mathbb{E}(x_{\ell,k}) = \mu^2$. The variables are different with probability $1 - n^{-1}$. So, putting it altogether

$$\begin{aligned} \mathbb{E}[x_{i,j}x_{l,k}] &= n^{-1}(\sigma^2 + \mu^2) + \mu^2(1 - n^{-1}) && (10) \\ &= n^{-1}\sigma^2 + \mu^2. && (11) \end{aligned}$$

Now using this and (3) gives us

$$\operatorname{cov}(x_{i,j}, x_{\ell,k}) = n^{-1}\sigma^2 + \mu^2 - \mu^2 = n^{-1}\sigma^2, \quad (12)$$

as required.

4. An outline of the solution is as follows:

    - This procedure does not yield the required independence of a bootstrap sample. To see this intuitively notice that for a general dataset it might be impossible to draw the exact same value all N times using this approach, since some values might not be present at every partition. If the draws were truly independent, this should be possible (though very unlikely).
    - Since there are $n \ll N$ unique levels in this dataset we can compress the dataset into a frequency table of how often each unique score occurs. This can be done by splitting the dataset across the computers and at each computer counting the number of times that each unique level occurs. All of these counts can then be sent to the central computer to create the final frequency table. Doing this we go from a dataset which was a vector in N dimensions to a $n \times 2$ array. This will be able to fit on a single computer. We can then use the multinomial distribution to draw samples of the required size as frequency tables.