Assignment 3

Callum Collier - 640533325

Due Date: 12:30pm 14th May 2021

Question 1

```
# load packages and ggpairs.R function
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

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

```
library(leaflet)
source("ggpairs.R")
```

```
# read in inflows data and convert to tibble
inflows = as_tibble(read_csv("inflows.csv", col_names = TRUE))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   YEAR = col_double(),
##   WEEK = col_double(),
##   Lakes_Manapouri = col_double(),
##   Lake_Ohau = col_double(),
##   Lake_Pukaki = col_double(),
##   Lake_Taupo = col_double(),
```

```
##   Lake_Tekapo = col_double(),
##   Lake_Matahina = col_double(),
##   Lake_Dunstan = col_double(),
##   Lake_Wanaka = col_double()
## )
```

```
locations = read_csv("locations.csv", col_names = TRUE)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   LAKES = col_character(),
##   LAT = col_double(),
##   LNG = col_double()
## )
```

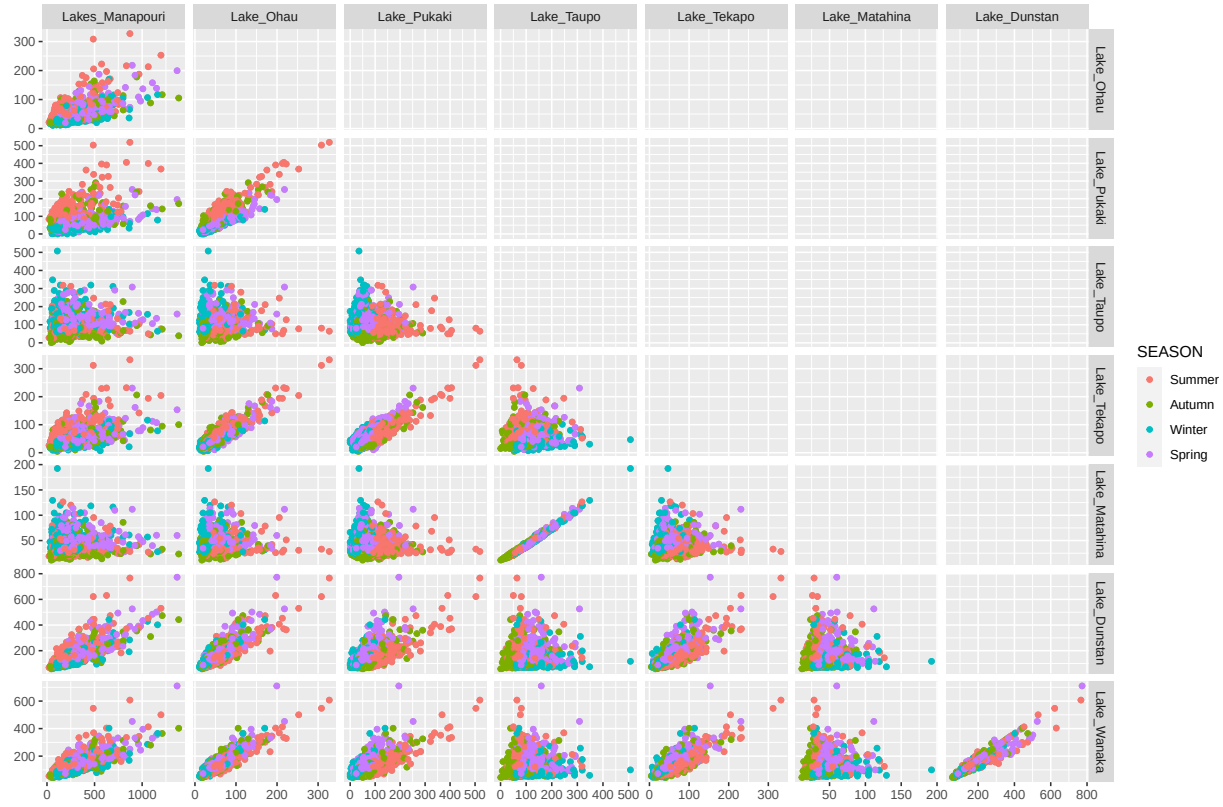
(a)

```
# create new attribute in data set called SEASON
inflows$SEASON = ifelse(inflows$WEEK < 10, "Summer", ifelse(inflows$WEEK < 23,
                 "Autumn", ifelse(inflows$WEEK < 36, "Winter",
                 ifelse(inflows$WEEK < 49, "Spring", "Summer"))))
# convert SEASON attribute to factor
inflows$SEASON = factor(inflows$SEASON,levels=c("Summer","Autumn","Winter","Spring"))
# display first four rows of inflows.
head(inflows, 4)
```

```
## # A tibble: 4 x 11
##    YEAR  WEEK Lakes_Manapouri Lake_Ohau Lake_Pukaki Lake_Taupo Lake_Tekapo
##   <dbl> <dbl>           <dbl>     <dbl>       <dbl>      <dbl>       <dbl>
## 1  1988     1            117.      38.3        53.0       67.6        62.0
## 2  1988     2            420.      45.7        82.0       57.8        66.2
## 3  1988     3            345.      77.3       132.        36.0        97.6
## 4  1988     4            236.      39.6        95.8       41.2        75.0
## # ... with 4 more variables: Lake_Matahina <dbl>, Lake_Dunstan <dbl>,
## #   Lake_Wanaka <dbl>, SEASON <fct>
```

(b)

```
# create pairs plot comparing the inflows for the lakes
ggpairs(inflows, contains("_"), color = SEASON)
```



An example of a highly correlated pair is Lake_Taupo and Lake_Matahina while an example of an uncorrelated pair is Lake_Manapouri and Lake_Matahina. The SEASON affects the inflows as the inflows tend to be lower in Winter and Autumn whereas inflows tend to be slightly higher in Summer and Spring. (This is presumably due to the drier months decreasing inflows in Winter and Autumn and vice versa.)

(c)

```
# reshape inflows data set and display
inflows_longer = pivot_longer(inflows, contains("_"), names_to = "LAKE",
                              values_to = "INFLOW")
head(inflows_longer, n=4)
```

```
## # A tibble: 4 x 5
##    YEAR  WEEK SEASON LAKE            INFLOW
##   <dbl> <dbl> <fct>  <chr>            <dbl>
## 1  1988     1 Summer Lakes_Manapouri  117.
## 2  1988     1 Summer Lake_Ohau         38.3
## 3  1988     1 Summer Lake_Pukaki       53.0
## 4  1988     1 Summer Lake_Taupo        67.6
```

(d)

```
# table showing average weekly inflow by LAKE
inflows_longer %>% group_by(LAKE) %>% summarise(MEAN.INFLOW = mean(INFLOW))
```

```
## # A tibble: 8 x 2
##   LAKE            MEAN.INFLOW
##   <chr>                 <dbl>
## 1 Lake_Dunstan          146.
## 2 Lake_Matahina          40.1
## 3 Lake_Ohau              49.1
## 4 Lake_Pukaki            73.8
## 5 Lake_Taupo             98.0
## 6 Lake_Tekapo            52.4
## 7 Lake_Wanaka           121.
## 8 Lakes_Manapouri       248.
```

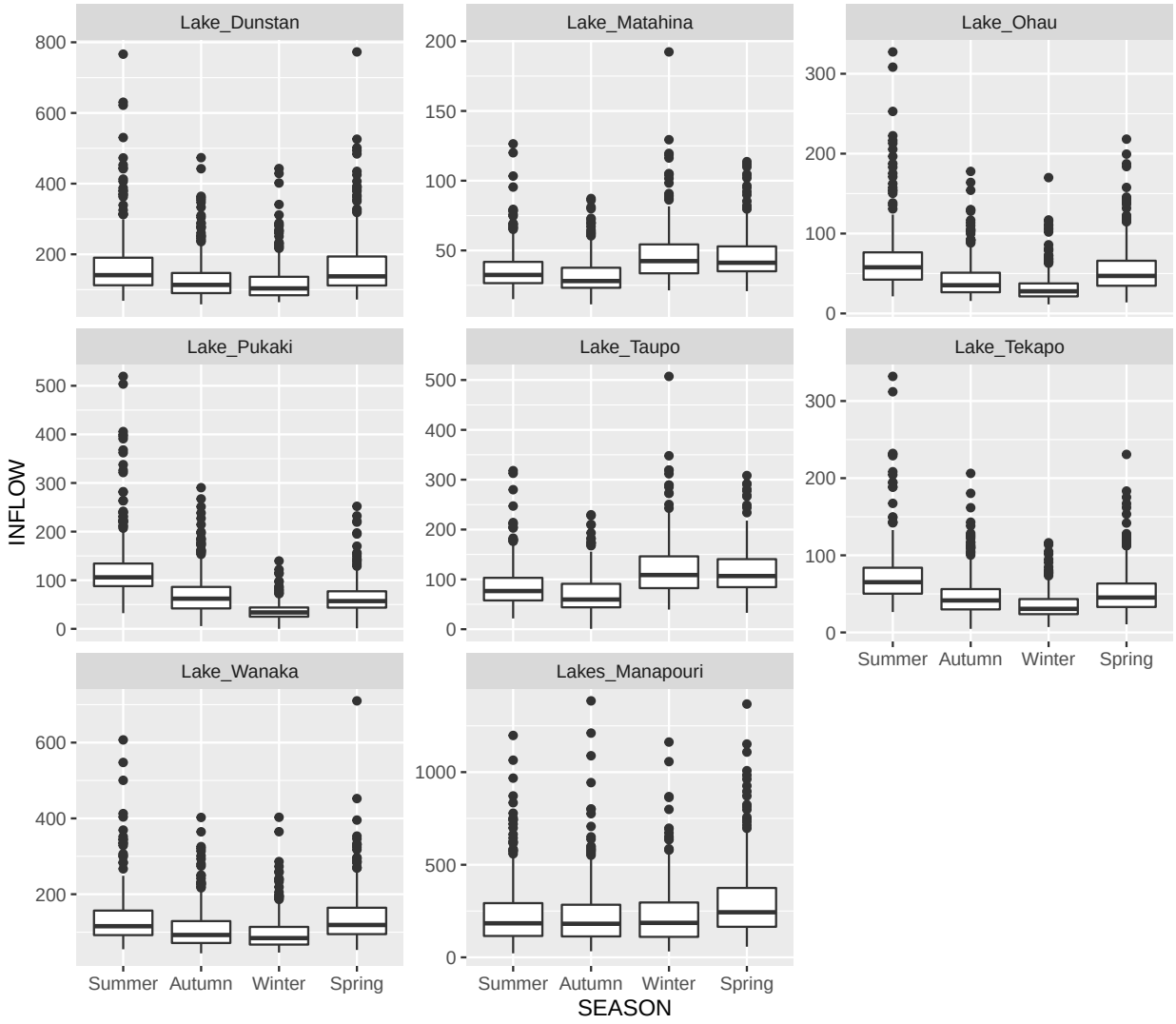
(e)

```
# table containing average weekly inflows for each lake in each season
inflows_longer %>% group_by(LAKE, SEASON) %>%
  summarise(MEAN.INFLOW = mean(INFLOW), .groups = 'drop') %>%
  pivot_wider(names_from = "SEASON", values_from = "MEAN.INFLOW")
```

```
## # A tibble: 8 x 5
##   LAKE            Summer Autumn Winter Spring
##   <chr>            <dbl>  <dbl>  <dbl>  <dbl>
## 1 Lake_Dunstan     166.   130.   120.   168.
## 2 Lake_Matahina     36.2   32.1   46.4   45.8
## 3 Lake_Ohau         66.2   42.6   33.1   54.7
## 4 Lake_Pukaki      124.    71.6   35.9   64.2
## 5 Lake_Taupo        85.2   70.4  119.   117.
## 6 Lake_Tekapo       73.5   47.7   35.6   53.0
## 7 Lake_Wanaka      136.   109.    98.5  140.
## 8 Lakes_Manapouri  234.   230.   227.   301.
```

(f)

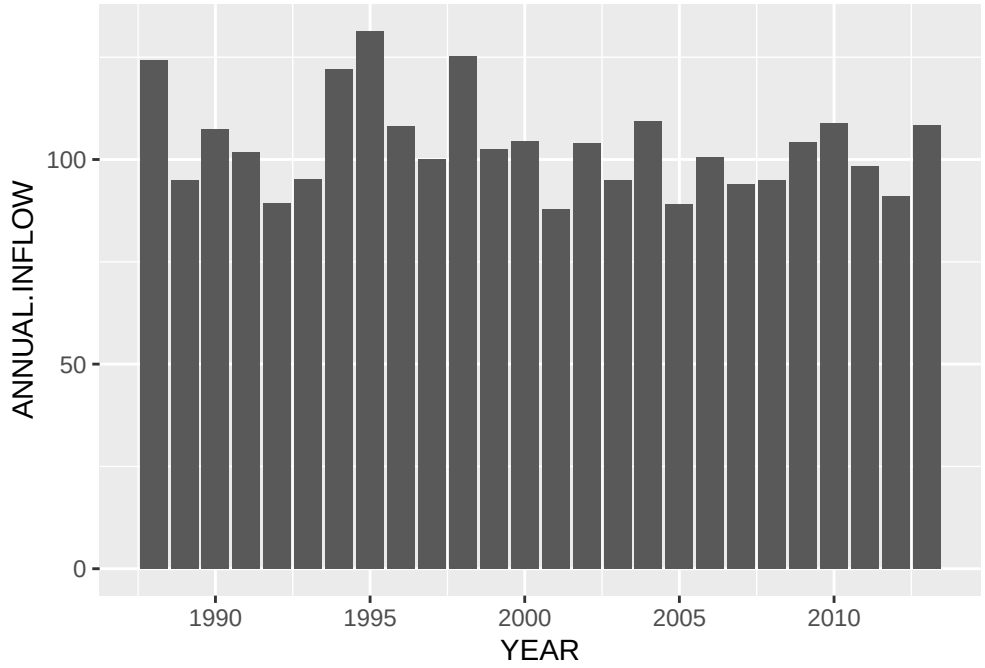
```
# box plots showing distribution of weekly hydro inflows for each season
ggplot(inflows_longer) + geom_boxplot(aes(x = SEASON, y = INFLOW)) +
  facet_wrap(~LAKE, scale = "free_y")
```



Summer and Spring tend to have higher average inflows than Autumn or Spring do, except for Lake Matahina and Lake Taupo - both of which have higher inflows in Winter than in Summer and Spring (but for both these lakes, Autumn is still the season with the lowest inflow).

(g)

```
# create table of annual inflows and plot as column chart
annual_inflows = inflows_longer %>% group_by(YEAR) %>%
  summarise(ANNUAL.INFLOW = mean(INFLOW))
ggplot(annual_inflows) + geom_col(mapping = aes(x = YEAR, y = ANNUAL.INFLOW),
                                  position = position_dodge())
```



There tends to be a decreasing trend - annual inflows tend to be decreasing overtime. Despite this, there still seems to be periods of higher annual inflow and lower annual inflow. These also appear to be periodic in nature (i.e. a few years with lower inflow and few years with higher inflow). This is probably due to global climatic events such as El Nino and El Nina which affect precipitation levels and thus inflows.

(h)

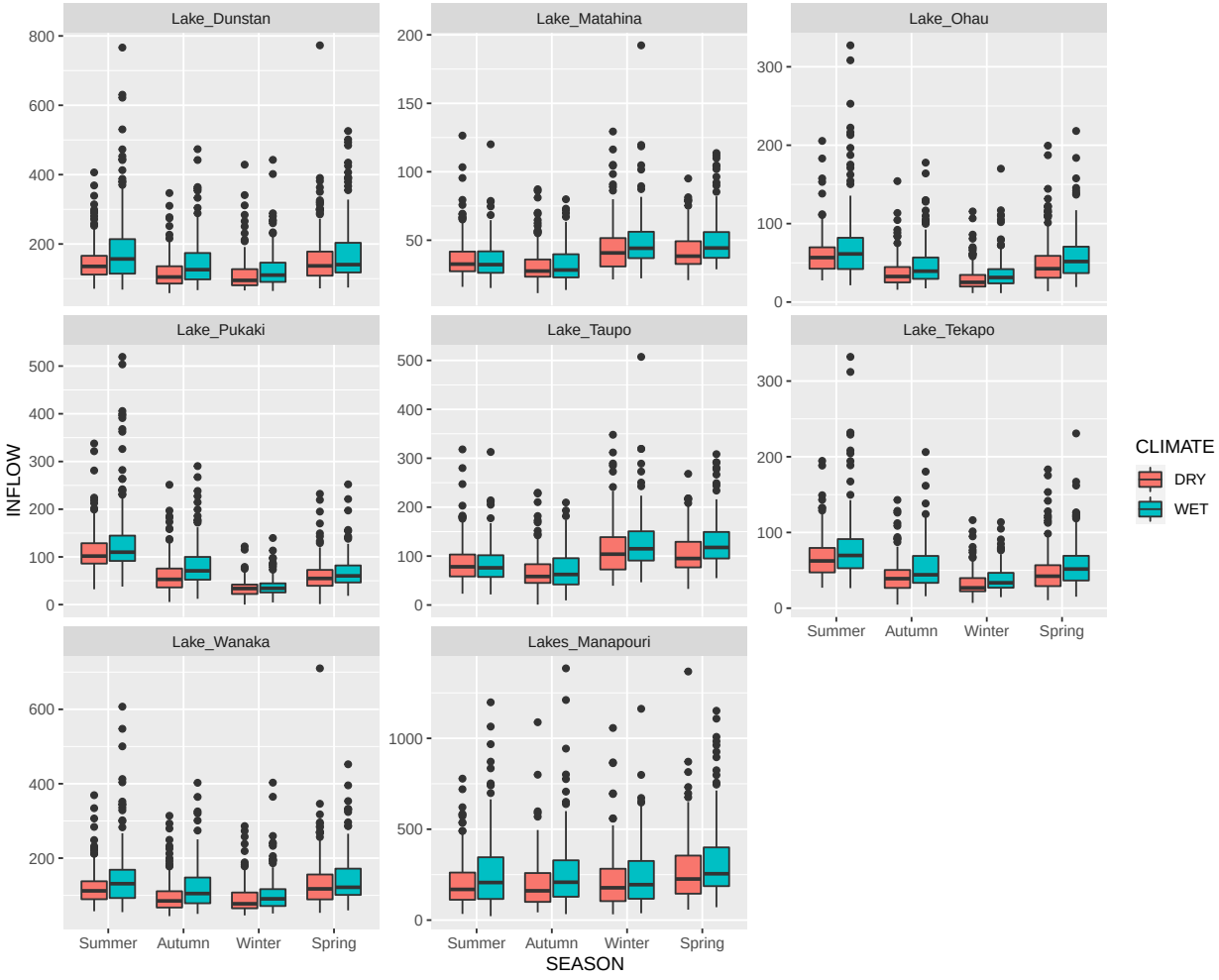
```
# find mean inflow of all the lakes
mean_inflow = mean(annual_inflows$ANNUAL.INFLOW, na.rm = TRUE)
# set CLIMATE attribute and convert to a factor
annual_inflows$CLIMATE = ifelse(annual_inflows$ANNUAL.INFLOW < mean_inflow, "DRY", "WET")
annual_inflows$CLIMATE = factor(annual_inflows$CLIMATE,levels=c("DRY","WET"))
# join annual_inflows and inflows data sets
inflows = inner_join(inflows, annual_inflows[, -2], by = "YEAR")
head(inflows, 4)
```

```
## # A tibble: 4 x 12
##    YEAR  WEEK Lakes_Manapouri Lake_Ohau Lake_Pukaki Lake_Taupo Lake_Tekapo
##   <dbl> <dbl>           <dbl>     <dbl>       <dbl>      <dbl>       <dbl>
## 1  1988     1            117.      38.3        53.0       67.6        62.0
## 2  1988     2            420.      45.7        82.0       57.8        66.2
```

```
## 3  1988     3            345.      77.3       132.        36.0        97.6
## 4  1988     4            236.      39.6        95.8       41.2        75.0
## # ... with 5 more variables: Lake_Matahina <dbl>, Lake_Dunstan <dbl>,
## #   Lake_Wanaka <dbl>, SEASON <fct>, CLIMATE <fct>
```

(i)

```
# box plots showing distribution of weekly hydro inflows based on season and
# type of year (dry or wet)
inflows_longer = pivot_longer(inflows, contains("_"), names_to = "LAKE",
                              values_to = "INFLOW")
ggplot(inflows_longer) + geom_boxplot(aes(x = SEASON, y = INFLOW, fill = CLIMATE)) +
  facet_wrap(~LAKE, scale = "free_y")
```

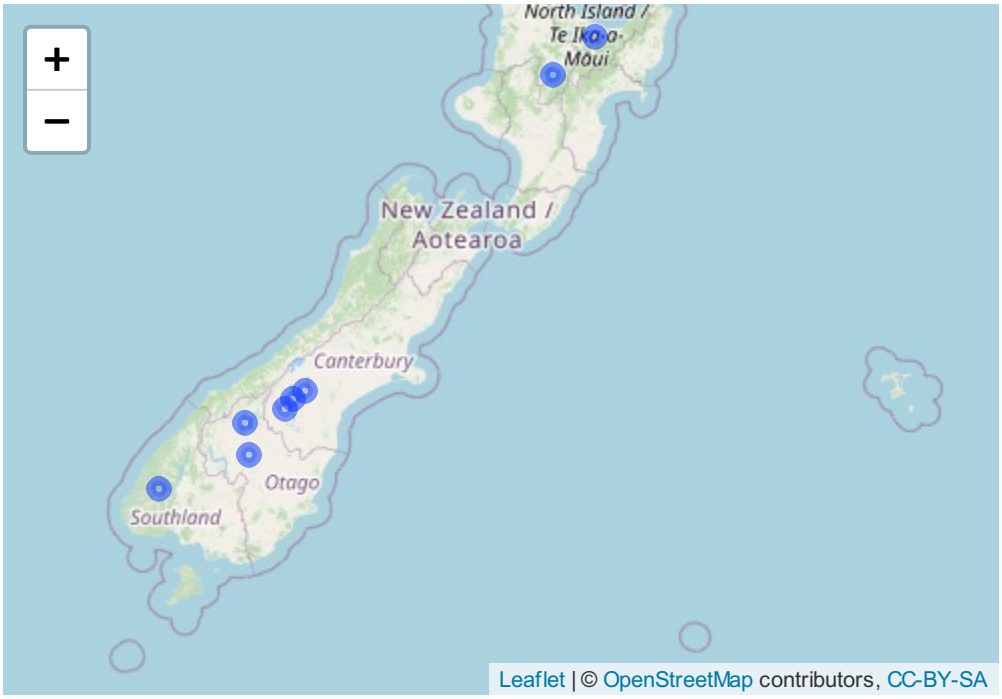


Unsurprisingly, wet years have higher average inflows than dry years do. Additionally, Spring and Summer tend to have higher inflows than Autumn and Winter (as previously states) and this holds true regardless of whether the year is a dry year or a wet year.

(j)

```
# create a map of New Zealand showing the location of each lake
leaflet(locations) %>% addTiles() %>% addCircleMarkers(radius = 4)
```

```
## Assuming "LNG" and "LAT" are longitude and latitude, respectively
```

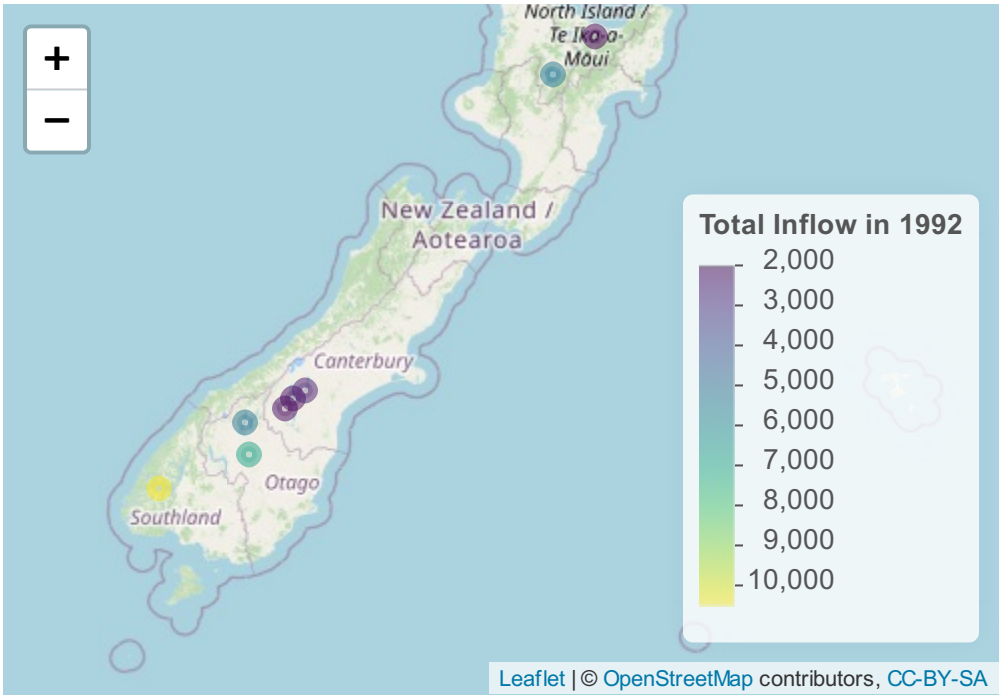


(k)

```
# pick out just lakes data and find total inflow of each lake
just_lakes = inflows[209:260, 3:10]
sum_inflows = c(sum(just_lakes$Lakes_Manapouri), sum(just_lakes$Lake_Ohau),
                sum(just_lakes$Lake_Pukaki), sum(just_lakes$Lake_Taupo),
                sum(just_lakes$Lake_Tekapo), sum(just_lakes$Lake_Matahina),
                sum(just_lakes$Lake_Dunstan), sum(just_lakes$Lake_Wanaka))

# initialise $YR1992 attribute
locations$YR1992 = 0

# add sum_inflows data into YR1992 attribute and convert to numeric data
for(i in 1:8)
  locations$YR1992[i] = sum_inflows[i]
locations$YR1992 = as.numeric(as.character(locations$YR1992))

# create a map of New Zealand showing the location of each lake and the inflows in 1992
# includes scale and viridis colourmap
colorNum = colorNumeric(palette = "viridis", domain = c(2000, 10500))
leaflet(locations) %>% addTiles() %>%
```

```
  addCircleMarkers(color = ~colorNum(YR1992), radius = 4) %>%
  addLegend("bottomright", colorNum, c(2000, 10500), title = "Total Inflow in 1992")
```

```
## Assuming "LNG" and "LAT" are longitude and latitude, respectively
```



Question 2

```
# load packages read in heart data
library(rpart)
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rattle)
```

```
## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'

## The following object is masked from 'package:randomForest':
##
##     importance
```

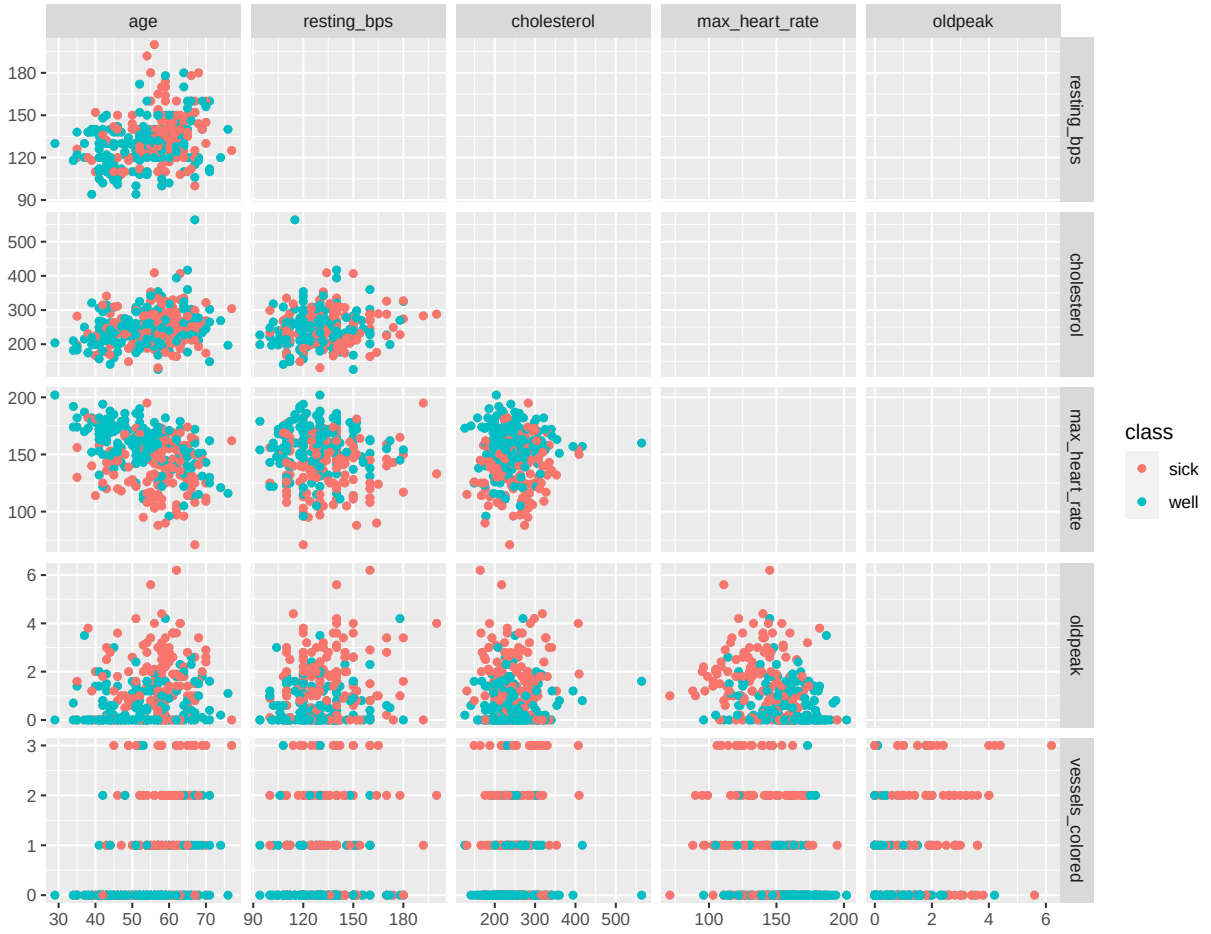
```
heart = read_csv("heart.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   age = col_double(),
##   sex = col_character(),
##   chest_pain_type = col_character(),
##   resting_bps = col_double(),
##   cholesterol = col_double(),
##   fasting_blood_sugar = col_logical(),
##   resting_ecg = col_character(),
```

```
##   max_heart_rate = col_double(),
##   exercise_induced_angina = col_logical(),
##   oldpeak = col_double(),
##   slope = col_character(),
##   vessels_colored = col_double(),
##   thal = col_character(),
##   class = col_character()
## )
```

(a)

```
# create pairs plot comparing the numeric heart data
ggpairs(heart, c(1,4,5,8,10,12), color = class)
```



(b)

```
# set seed of random number generator to 100
set.seed(100)
# generate training set of 150 data points
train_set = sample(1:303, 150)
```

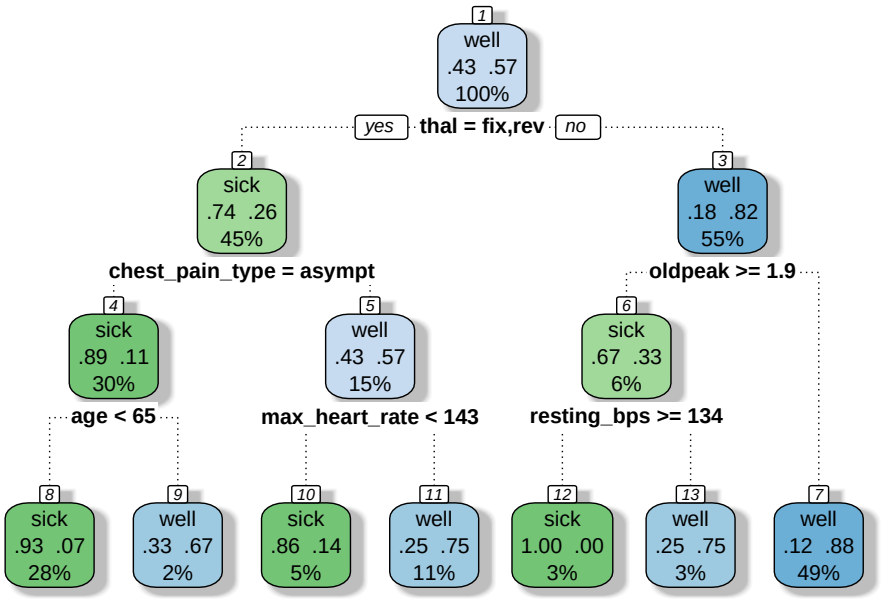
(c)

(i)

```
# create ctrl argument with max depth of 3 and make tree
ctrl = rpart.control(minsplit = 1, cp = 0, maxdepth = 3)
tree3 = rpart(class ~., data = heart[train_set,], method = "class", control = ctrl)

# max depth of 5
ctrl = rpart.control(minsplit = 1, cp = 0, maxdepth = 5)
tree5 = rpart(class ~., data = heart[train_set,], method = "class", control = ctrl)

# max depth of 7
ctrl = rpart.control(minsplit = 1, cp = 0, maxdepth = 7)
tree7 = rpart(class ~., data = heart[train_set,], method = "class", control = ctrl)
```

(ii)

```
# view tree3
fancyRpartPlot(tree3)
```



(iii)

```
# create prediction for tree 3
heart$Predict = predict(tree3, heart, type = "class")
# create in-sample confusion matrix for tree3
In_Sample_tree3 = table(Class = heart[train_set,]$class,
                        Prediction = heart[train_set,]$Predict)
# create out-sample confusion matrix for tree3
Out_Sample_tree3 = table(Class = heart[-train_set,]$class,
                        Prediction = heart[-train_set,]$Predict)

# create prediction for tree 5
heart$Predict = predict(tree5, heart, type = "class")
# create in-sample confusion matrix for tree5
In_Sample_tree5 = table(Class = heart[train_set,]$class,
                        Prediction = heart[train_set,]$Predict)
# create out-sample confusion matrix for tree5
Out_Sample_tree5 = table(Class = heart[-train_set,]$class,
                        Prediction = heart[-train_set,]$Predict)

# create prediction for tree 7
heart$Predict = predict(tree7, heart, type = "class")
# create in-sample confusion matrix for tree7
In_Sample_tree7 = table(Class = heart[train_set,]$class,
                        Prediction = heart[train_set,]$Predict)
# create out-sample confusion matrix for tree7
Out_Sample_tree7 = table(Class = heart[-train_set,]$class,
                        Prediction = heart[-train_set,]$Predict)

# show the confusion matrices for tree3


In_Sample_tree3[1,1] = 65
In_Sample_tree3[1,2] = 0
In_Sample_tree3[2,1] = 76
In_Sample_tree3[2,2] = 9
Out_Sample_tree3[1,1] = 67
Out_Sample_tree3[1,2] = 6
Out_Sample_tree3[2,1] = 70
Out_Sample_tree3[2,2] = 10
show(In_Sample_tree3)
```

```
##       Prediction
## Class  sick well
##   sick   65    0
##   well   76    9
```

```
show(Out_Sample_tree3)
```

```
##       Prediction
## Class  sick well
##   sick   67    6
##   well   70   10
```

(iv)

```
# initialise accuracy table
accuracy_table = matrix(, nrow = 3, ncol = 2)

# parse accuracy data into respective index
accuracy_table[1,1] = ((In_Sample_tree3[1,1] + In_Sample_tree3[2,2])/
                       (In_Sample_tree3[1,1] + In_Sample_tree3[2,2] +
                          In_Sample_tree3[1,2] + In_Sample_tree3[2,1]))
accuracy_table[1,2] = ((Out_Sample_tree3[1,1] + Out_Sample_tree3[2,2])/
                       (Out_Sample_tree3[1,1] + Out_Sample_tree3[2,2] +
                          Out_Sample_tree3[1,2] + Out_Sample_tree3[2,1]))

accuracy_table[2,1] = ((In_Sample_tree5[1,1] + In_Sample_tree5[2,2])/
                       (In_Sample_tree5[1,1] + In_Sample_tree5[2,2] +
                          In_Sample_tree5[1,2] + In_Sample_tree5[2,1]))
accuracy_table[2,2] = ((Out_Sample_tree5[1,1] + Out_Sample_tree5[2,2])/
                       (Out_Sample_tree5[1,1] + Out_Sample_tree5[2,2] +
                          Out_Sample_tree5[1,2] + Out_Sample_tree5[2,1]))

accuracy_table[3,1] = ((In_Sample_tree7[1,1] + In_Sample_tree7[2,2])/
                       (In_Sample_tree7[1,1] + In_Sample_tree7[2,2] +
                          In_Sample_tree7[1,2] + In_Sample_tree7[2,1]))
accuracy_table[3,2] = ((Out_Sample_tree7[1,1] + Out_Sample_tree7[2,2])/
                       (Out_Sample_tree7[1,1] + Out_Sample_tree7[2,2] +
                          Out_Sample_tree7[1,2] + Out_Sample_tree7[2,1]))

# change names of rows and columns
rownames(accuracy_table) <- c("tree3", "tree5", "tree7")
colnames(accuracy_table) <- c("In_Sample", "Out_Sample")

# display accuracy table
show(accuracy_table)
```

```
##       In_Sample Out_Sample
## tree3 0.4933333  0.5032680
## tree5 0.9533333  0.7581699
## tree7 1.0000000  0.7254902
```

(v)

The out-of-sample accuracy is significantly lower than the in-sample accuracy. This is because the in-sample performance will almost always be better than the out-of-sample performance, as the model was built on the training set i.e. the in-sample data set, so was made to be optimised for that specific data set. The out-of-sample data set is outside data that the model was not built on, so likely has a lower accuracy, as seen from the table above.

(d)

(i)

```
# set termination criteria to be max depth of 3
ctrl = rpart.control(minsplit = 1, cp = 0, maxdepth = 3)

parms = list(loss = matrix(c(0, 1, 100, 0), nrow = 2))
tree_A = rpart(class ~., data = heart, subset = train_set, method = "class",
               control = ctrl, parms = parms)

parms = list(loss = matrix(c(0, 20, 80, 0), nrow = 2))
tree_B = rpart(class ~., data = heart, subset = train_set, method = "class",
               control = ctrl, parms = parms)

parms = list(loss = matrix(c(0, 80, 20, 0), nrow = 2))
tree_C = rpart(class ~., data = heart, subset = train_set, method = "class",
               control = ctrl, parms = parms)

parms = list(loss = matrix(c(0, 100, 1, 0), nrow = 2))
tree_D = rpart(class ~., data = heart, subset = train_set, method = "class",
               control = ctrl, parms = parms)
```

(ii)

```
# create prediction for tree A
heart$Predict = predict(tree_A, heart, type = "class")
# create in-sample confusion matrix for tree A
In_Sample_tree_A = table(Class = heart[train_set,]$class,
                         Prediction = heart[train_set,]$Predict)
# create out-sample confusion matrix for tree A
Out_Sample_tree_A = table(Class = heart[-train_set,]$class,
                         Prediction = heart[-train_set,]$Predict)

# create prediction for tree B
heart$Predict = predict(tree_B, heart, type = "class")
# create in-sample confusion matrix for tree B
In_Sample_tree_B = table(Class = heart[train_set,]$class,
                         Prediction = heart[train_set,]$Predict)
# create out-sample confusion matrix for tree B
Out_Sample_tree_B = table(Class = heart[-train_set,]$class,
                         Prediction = heart[-train_set,]$Predict)

# create prediction for tree C
heart$Predict = predict(tree_C, heart, type = "class")
# create in-sample confusion matrix for tree C
In_Sample_tree_C = table(Class = heart[train_set,]$class,
                         Prediction = heart[train_set,]$Predict)
# create out-sample confusion matrix for tree C
Out_Sample_tree_C = table(Class = heart[-train_set,]$class,
                         Prediction = heart[-train_set,]$Predict)

# create prediction for tree D
heart$Predict = predict(tree_D, heart, type = "class")
# create in-sample confusion matrix for tree D
In_Sample_tree_D = table(Class = heart[train_set,]$class,
                         Prediction = heart[train_set,]$Predict)
```

```r
# create out-sample confusion matrix for tree D
Out_Sample_tree_D = table(Class = heart[-train_set,]$class,
                          Prediction = heart[-train_set,]$Predict)
```

```
       Prediction
Class   sick well
  sick    65    0
  well    76    9
       Prediction
Class   sick well
  sick    67    6
  well    70   10
```