# Day4 exercise solutions

Ali Movasati

Oct. 7th, 2024

```
# Set global code chunk options
knitr::opts_chunk$set(warning = FALSE)
```

```
# load required libraries
library(ggplot2)
library(magrittr)
library(dplyr)
library(tibble)
library(maps)
library(fields)


# define functions
`%notin%` <- Negate(`%in%`)
```

## Problem 1

```
# read in the data

ec <- read.table(file = "/Users/alimos313/Documents/studies/phd/university/courses/stat-modelling/StatM
```

### 1.A)

```
# inspect the data and variable
str(ec)
```

```
## 'data.frame':    48 obs. of  5 variables:
##  $ City     : chr  "Amsterdam" "Athens" "Bogota" "Bombay" ...
##  $ Work     : int  1714 1792 2152 2052 1708 1971 NA 2041 1924 1717 ...
##  $ Price    : num  65.6 53.8 37.9 30.3 73.8 56.1 37.1 61 73.9 91.3 ...
##  $ Salary   : num  49 30.4 11.5 5.3 50.5 12.5 NA 10.9 61.9 62.9 ...
##  $ SalaryCat: chr  "(35.1,67.6]" "(2.6,35.1]" "(2.6,35.1]" "(2.6,35.1]" ...
```

```
ec <- ec[!is.na(ec$SalaryCat),]
```

**« Comments »**

This dataset contains economic information for 48 cities. The three variables work, price and salary are numeric values. Two of the observations did not have salary info available in the dataset and were removed for the downstream analyses. The salary has been categorized into 3 groups. Below is a summary of each variable:

```
summary(ec)
```

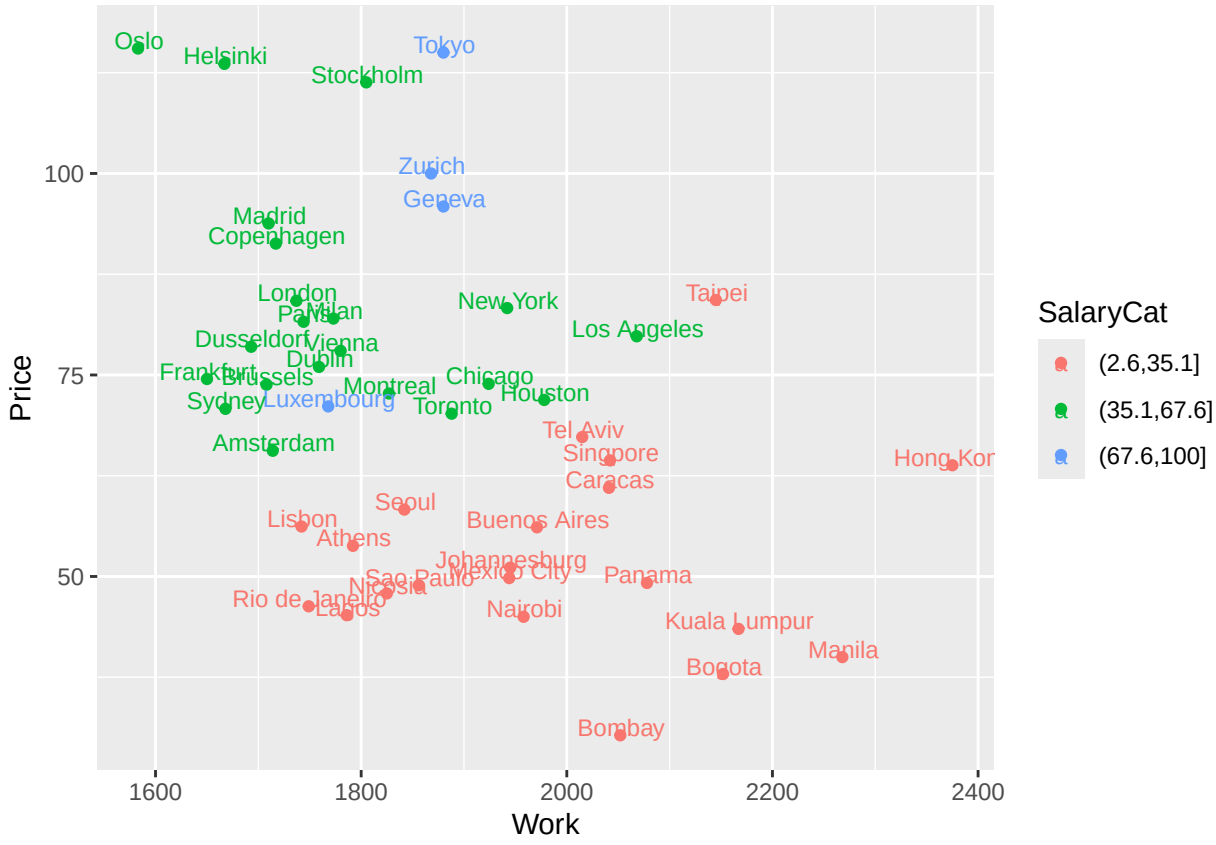
```
##      City                Work          Price            Salary        SalaryCat
##  Length:46          Min.   :1583   Min.   : 30.30   Min.   :  2.70   Length:46
##  Class :character   1st Qu.:1745   1st Qu.: 51.77   1st Qu.: 14.38   Class :character
##  Mode  :character   Median :1849   Median : 70.95   Median : 43.65   Mode  :character
##                     Mean   :1880   Mean   : 70.10   Mean   : 39.55
##                     3rd Qu.:1976   3rd Qu.: 81.90   3rd Qu.: 59.70
##                     Max.   :2375   Max.   :115.50   Max.   :100.00
```

```
table(ec$SalaryCat, useNA = "always")
```

```
##
##  (2.6,35.1] (35.1,67.6]  (67.6,100]         <NA>
##          21           21           4            0
```

We visualize the data based on work and price variables:

```
# visualize the variables
ec %>% ggplot(aes(x = Work, y = Price, col = SalaryCat)) +
       geom_point() +
       geom_text(aes(label = City), nudge_x = 1, nudge_y = 1, size = 3)
```



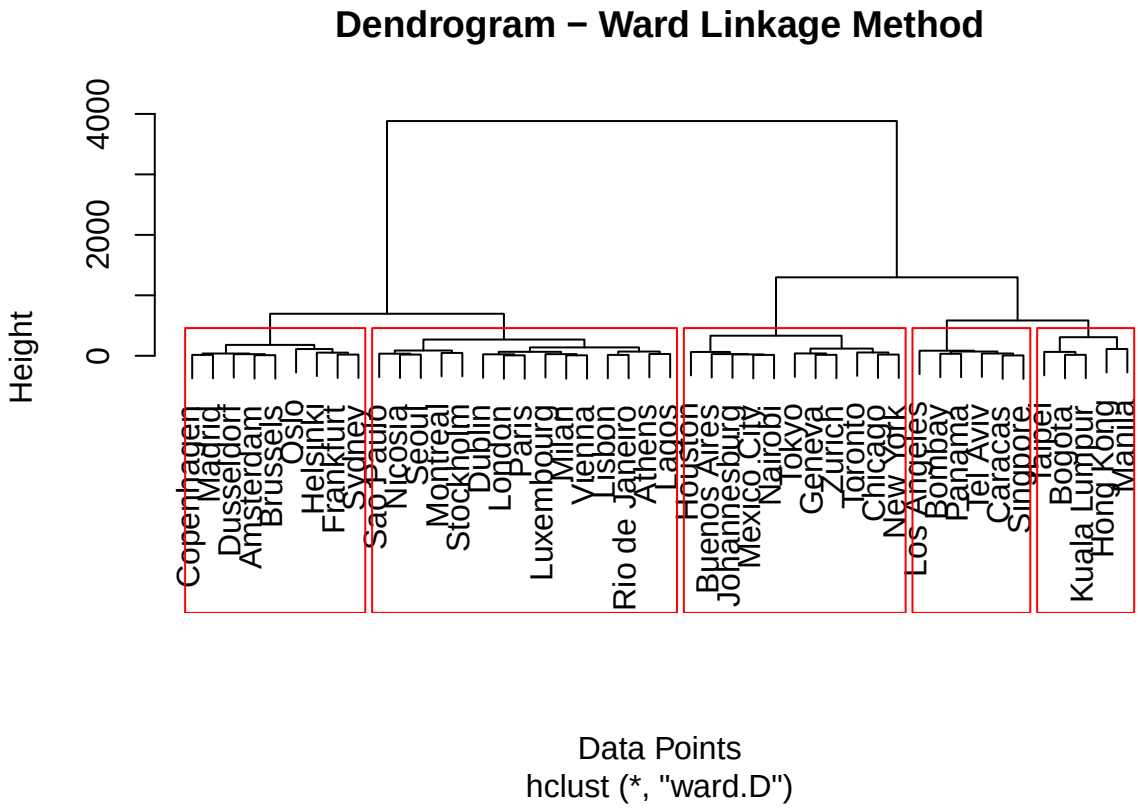

**« Comments »**

We see a noticeable separation in the data based on the salary categories. However, since one of the salary categories has only 4 samples, it might be a bit too small to build a classifier that can accurately and robustly predict observations belonging to this group.

## 1.B)

```
# prepare data for clustering
ec_clust <- ec
rownames(ec_clust) <- ec_clust$City
ec_clust <- ec_clust[,-c(1,5)]
```

```
# perform clustering


distance_matrix <- dist(ec_clust, method = "euclidean")
hc_ward <- hclust(distance_matrix, method = "ward.D")
plot(hc_ward, main = paste0("Dendrogram - Ward Linkage Method"), xlab = "Data Points", ylab = "Height")
rect.hclust(hc_ward, k = 5, border = "red")
```




**« Comments »**

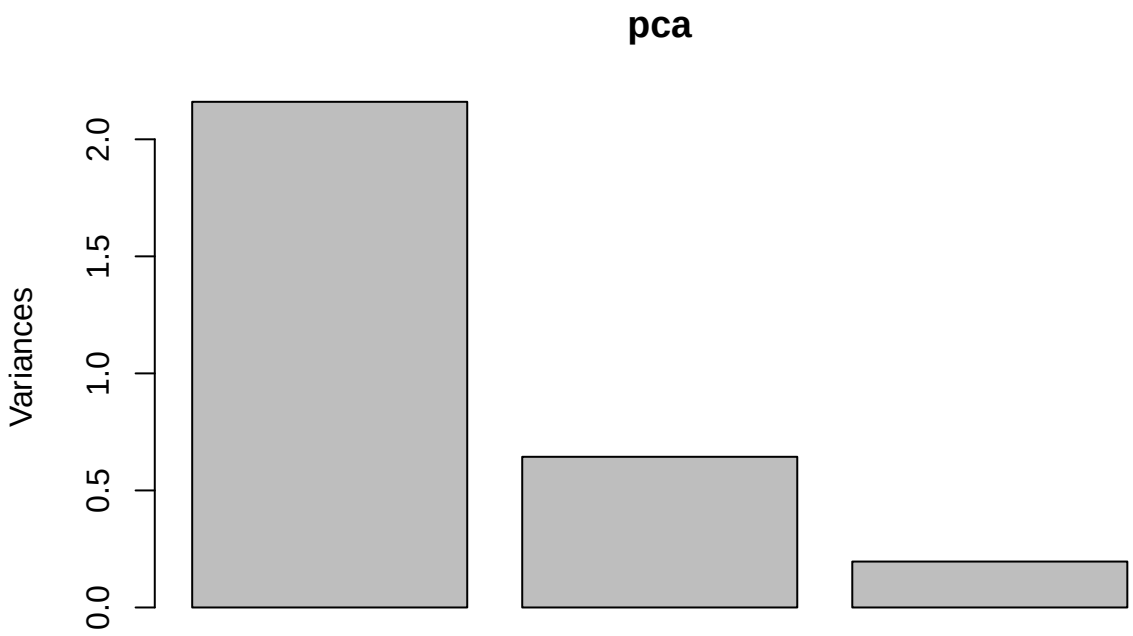
We clustered the data using the Ward linkage method. Based on the topology of dendogram, we can group them into 5 clusters (which is an arbitrary number).

## 1.C)

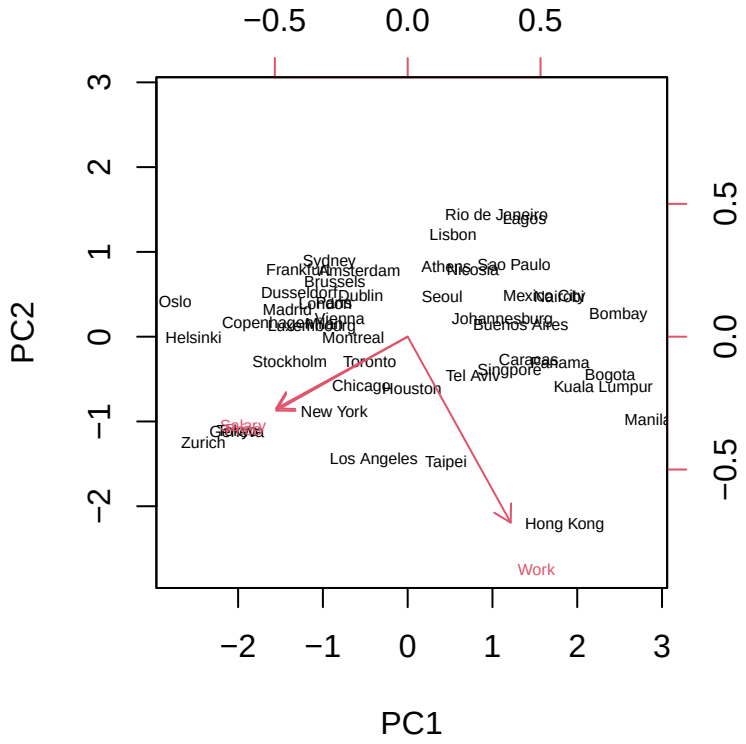
```
# prepare data for pca
ec_pca <- ec
rownames(ec_pca) <- ec_pca$City
ec_pca <- ec_pca[,-c(1,5)]
```

```
## perform PCA
pca <- prcomp(ec_pca, scale=TRUE)
```

```
## plot the PCs and the amount of variation how much of them can explain in the data
plot(pca)
```




```
## plot biplot

biplot(pca, scale = 0, cex = 0.5)
```

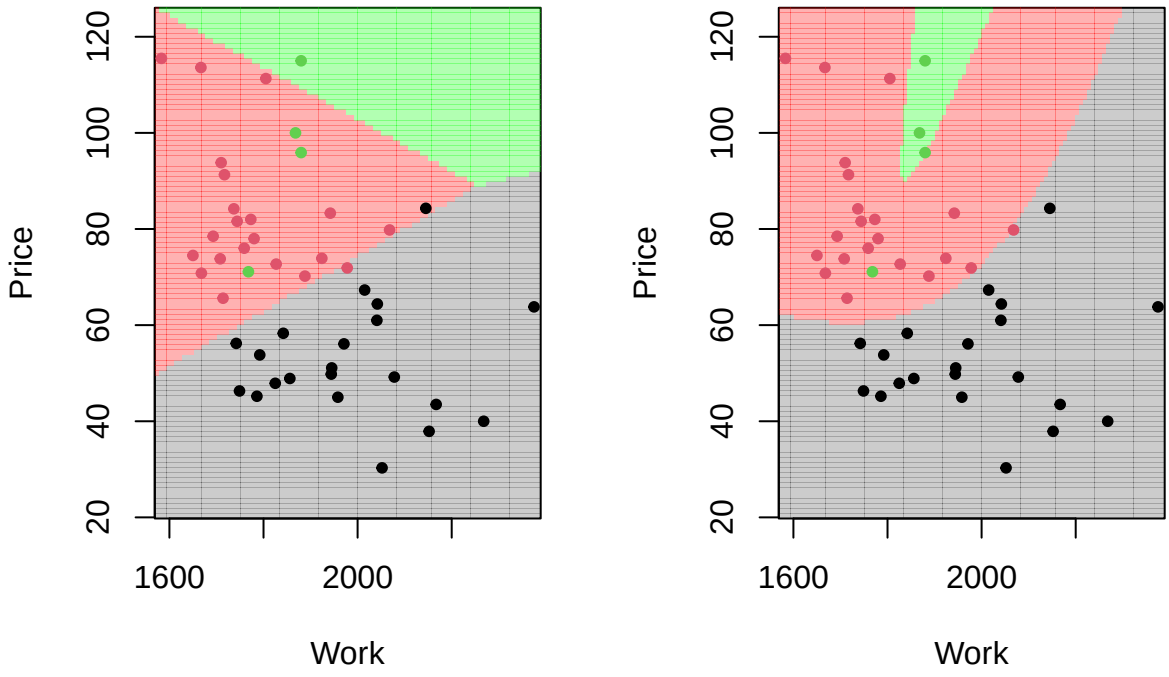


**« Comments »**

Most variability is seen in the variable Work. The two variable price and salary seem to co-vary with each other.

## 1.D)

```
par(mfcol=c(1,2))

# prepare data for discriminant analyses
ec_da <- ec
rownames(ec_da) <- ec_da$City
ec_da <- ec_da[order(ec_da$SalaryCat),]
ec_sal_gp <- factor(ec_da$SalaryCat)


ec_da <- ec_da[,c(2,3)]



x1 <- seq(min(ec_da$Work)-10, to=max(ec_da$Work)+10, length=99)
x2 <- seq(min(ec_da$Price)-10, to=max(ec_da$Price)+10, length=100)

grid <- expand.grid( Work=x1, Price=x2)



## Linear Discriminant Analysis

tmp_linear <- lda(ec_da, grouping=ec_sal_gp)

cll <- predict(tmp_linear, grid)$class


image(x1, x2, matrix(as.numeric(cll), 99, 100),  # partition of space
      col=c(rgb(0,0,0,0.2), rgb(1,0,0,.3), rgb(0,1,0,.3)), xlab = "Work", ylab = "Price")

points(ec_da, col=ec_sal_gp, pch=20)



## Quadratic Discriminant Analysis

tmp_quadratic <- qda(ec_da, grouping=ec_sal_gp)

cll <- predict(tmp_quadratic, grid)$class


image(x1, x2, matrix(as.numeric(cll), 99, 100),  # partition of space
      col=c(rgb(0,0,0,0.2), rgb(1,0,0,.3), rgb(0,1,0,.3)), xlab = "Work", ylab = "Price")

points(ec_da, col=ec_sal_gp, pch=20)
```




**« Comments »**

Obviously, **QDA performs better in classifying salray categories compared to LDA**. The reason lies in the fact that the **variance of the two variables are not equal**. While the variance in Work variable is $3.0395326 \times 10^4$, the variance in Price variable is 457.4968889. Therefore, we cannot make an assumption on the equality of the variances and use a linear discriminant analysis.

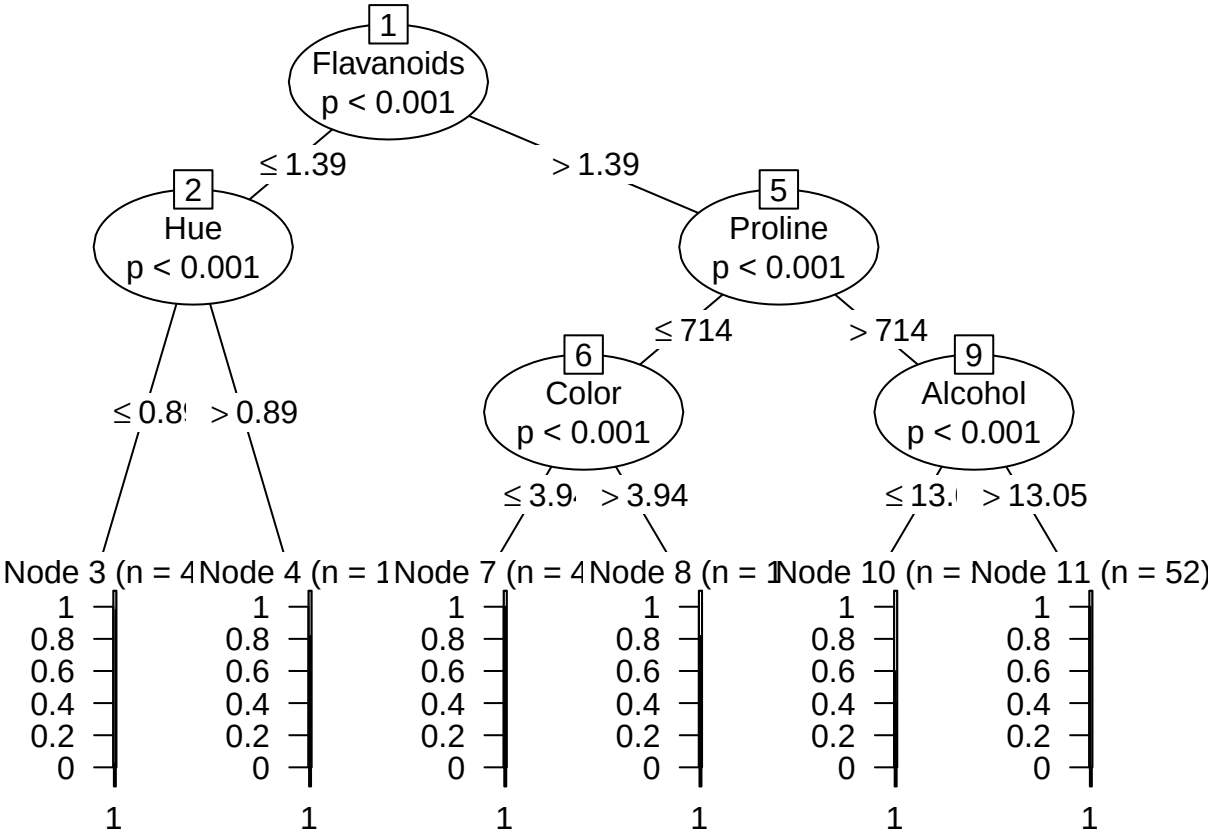
### 1.E)

## Problem 2

```
## load libraries

require("partykit")
require("adabag")
require("randomForest")

## load the data

load("/Users/alimos313/Documents/studies/phd/university/courses/stat-modelling/StatModelEx/day4/data/wi
```

### 2.A)

```
## generate a classification tree
c_tree <- ctree(Type ~ .,  data=wine,
                    control = ctree_control(maxdepth = 3))

## visualize the tree
plot(c_tree)
```




```
## assess performance
predicted_wines <- predict(c_tree, type ="response")
confusion_matrix <- table(Actual = wine$Type, Predicted = predicted_wines)
print(confusion_matrix)
```

```
##       Predicted
## Actual  1  2  3
##      1 58  1  0
##      2  4 66  1
##      3  0  3 45
```

```
## model random forest

wine.rf <- randomForest(Type~., data=wine, ntree=100, prox=TRUE)

## print model specifications

wine.rf
```

```
##
## Call:
##  randomForest(formula = Type ~ ., data = wine, ntree = 100, prox = TRUE)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 1.69%
## Confusion matrix:
##    1  2  3 class.error
```

```
## 1 59  0  0  0.00000000
## 2  1 68  2  0.04225352
## 3  0  0 48  0.00000000
```

```
## confusion matrix
print(confusion_matrix <- wine.rf$confusion[,-4])
```

```
##    1  2  3
## 1 59  0  0
## 2  1 68  2
## 3  0  0 48
```

```
sum( diag(confusion_matrix)) / sum(confusion_matrix)      # accuracy  (overall measure)
```

```
## [1] 0.9831461
```

```
diag(confusion_matrix) / rowSums(confusion_matrix)        # precision for each class
```

```
##         1         2         3
## 1.0000000 0.9577465 1.0000000
```

```
diag(confusion_matrix) / colSums(confusion_matrix)        # sensitivity for each class
```

```
##         1         2         3
## 0.9833333 1.0000000 0.9600000
```

### 2.B)

```
## which variables were most informative
varImpPlot(wine.rf)
```

## 2.C)

```
# Define the cross-validation function
cross_validate_rf <- function(data, target_variable, n_folds = 10, model = "rf") {
  # Convert target variable to a factor
  data[[target_variable]] <- as.factor(data[[target_variable]])

  # Create a vector to store accuracy for each fold
  accuracies <- numeric(n_folds)

  # Create folds
  set.seed(123)  # For reproducibility
  fold_indices <- sample(1:n_folds, nrow(data), replace = TRUE)

  # Cross-validation loop
  for (fold in 1:n_folds) {
    # Split data into training and testing sets
    test_data <- data[fold_indices == fold, ]
    train_data <- data[fold_indices != fold, ]

    if (model == "rf"){
        # Train the Random Forest model
        classification_model <- randomForest(as.formula(paste(target_variable, "~ .")),
                                data = train_data,
                                importance = TRUE)
    } else if (model == "ctree"){
        classification_model <- ctree(Type ~ .,  data=wine,
                                      control = ctree_control(maxdepth = 5))
    }

    # Predict on the test data
    predictions <- predict(classification_model, test_data)

    # Calculate accuracy
    accuracies[fold] <- mean(predictions == test_data[[target_variable]])
  }

  # Calculate the average accuracy
  average_accuracy <- mean(accuracies)

  # Print results
  cat("Average accuracy from cross-validation: ", average_accuracy * 100, "%\n")

  # Return the average accuracy
  return(average_accuracy)
}


## run the function for randomforest model
cross_validate_rf(wine, "Type", 10, "rf")
```

```
## Average accuracy from cross-validation:  98.10714 %
```

```
## [1] 0.9810714
```

```r
## run the function for classification tree model
cross_validate_rf(wine, "Type", 10, "ctree")
```

```
## Average accuracy from cross-validation:  94.96768 %
```

```
## [1] 0.9496768
```

## 2.D)

**« Comments »**

We obtain higher accuracy with random forest model compared to decision tree. This is because random forest tends to be more robust by adding more variability to the resulted trees by not only randomly selecting observations (bootstraping), but also randomly including the variables in the decision tress.