# Lab03: Classification and Regression Prediction Models

**Handed out:** Wednesday, March 8, 2023

**Return date:** Friday, March 24, 2023, at the ELEARNING link **`Lab03Submit`** in the **`Lab03`** folder.

**Objectives:** Comparison of different classification as well as regression tree models and the evaluation of their predictive properties.

**Grades:** This lab counts 9 % towards your final grade

**Format of answer:** Your answers (statistical figures and verbal description) should be submitted electronically as Word document. Add a running title with the following information: Lab03, your name and page numbers. Use this document as template: add your answers for each subtask, i.e., 1 (a) etc., in a red color as well as any requested statistical figures. Trial and error answers will lead to a deduction of points. You are expected to hand in professionally formatted answers: use a fixed pitch font, like **`Courier New`**, for any R code and output.

## Part 1: Classification trees [6 points]

You will be using for this part the dataset **`mushrooms.csv`** and split it into a stratified ***training*** data-frame with 2/3 of the observations and ***test*** data-frame with 1/3 observations. The dependent variable is **`type`**. Remove the variable **`veil_type`** with **`mushrooms$veil_type <- NULL`** because it is constant over all observations.

**Task 1:** Build a classification tree, properly prune the tree, and interpret the pruned tree. Show both the pruned and unpruned trees. Use the ***training*** data-frame. [1 points]

**Task 2:** Build a predictive model using the **`randomForest`** function with bagging based on the parameter **`ntree`**. Evaluate the relevance of the features. Use the ***training*** data-frame. [1 point]

**Task 3:** Build a predictive random forest model using the **`randomForest`** function and find the optimal hyper-parameter **`mtry`** for the number of features explored at each steps. Use the ***training*** data-frame. [1 point]

**Task 4:** Build a predictive boosted tree model using the function **`gbm`** and find the optimal depth hyper-parameter **`interaction.depth`**. Use the ***training*** data-frame. [1 point]

**Task 5:** Compare the models from tasks 1 to 4 for the ***test*** data-frame by using their **ROC** curves, the **`auc`** statistic, their prediction error rate and their overall node purity. Justify which model you would use to avoid mushroom poisoning. [2 points]

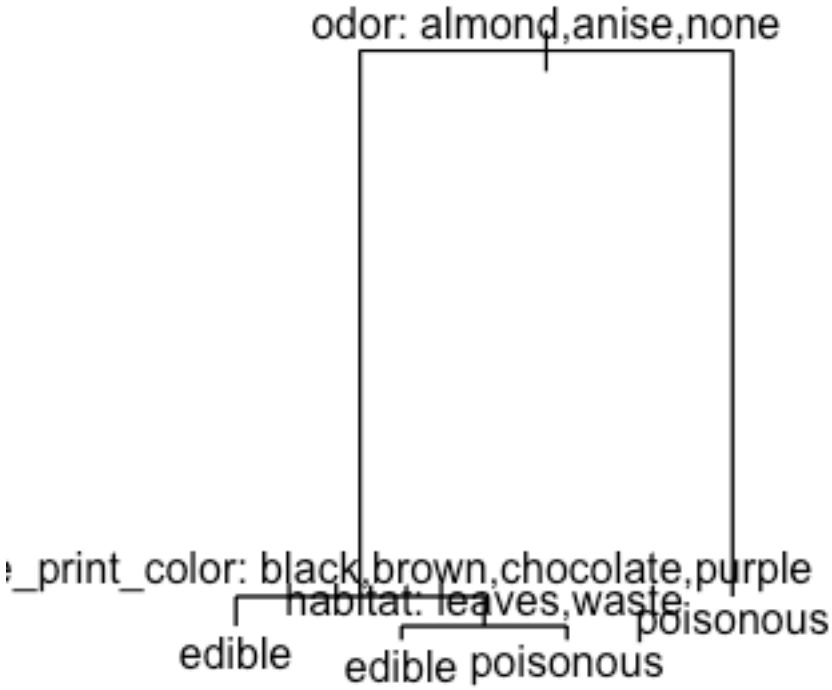
```
rm(list=ls())                          # Clear environment
oldpar <- par()                        # save default graphical
parameters
```

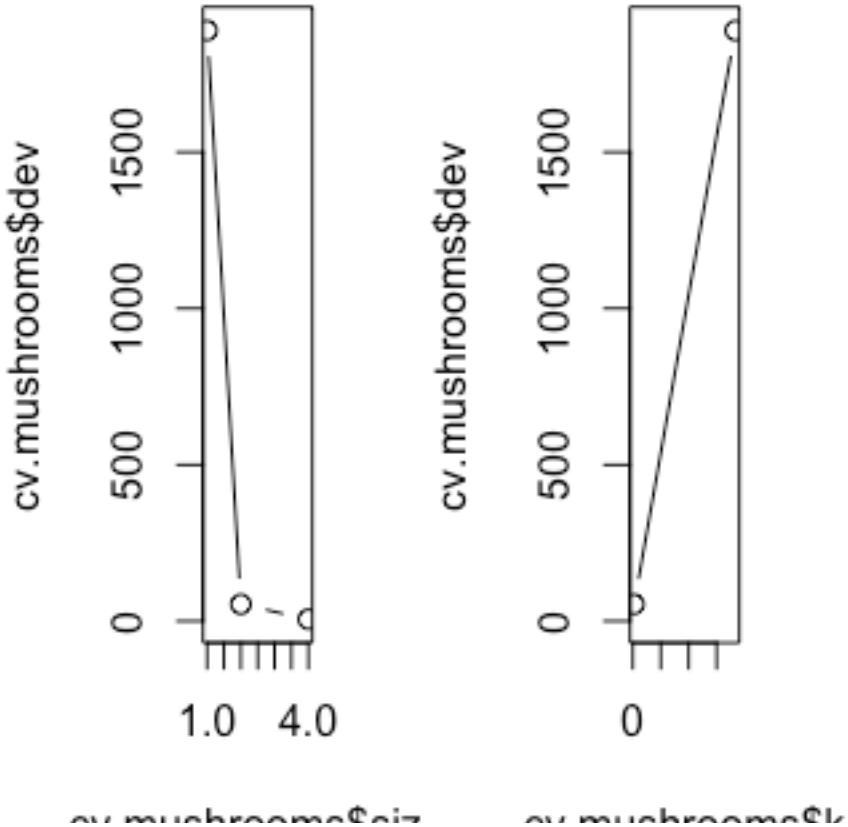
```
if (!is.null(dev.list()["RStudioGD"])) # Clear plot window
  dev.off(dev.list()["RStudioGD"])
cat("\014")                            # Clear the Console

library(tree)
library(ISLR2)
library(caret)
library(randomForest)
library(gbm)
library(caret)



# Load the dataset
mushrooms <- read.csv("/Users/jimpan/Documents/EPPS 6326/lab/lab3/
mushrooms.csv")

# Remove the variable veil_type
mushrooms$veil_type <- NULL

# Convert non-numeric columns to factors
non_numeric_cols <- sapply(mushrooms, is.character)
mushrooms[, non_numeric_cols] <- lapply(mushrooms[, non_numeric_cols],
factor)

table(is.na(mushrooms))
table(mushrooms$type)
# Remove rows with missing values
mushrooms <- na.omit(mushrooms)

# Split the dataset into training and test sets
set.seed(1)
train <- mushrooms[1:round(2/3*nrow(mushrooms)),]
```

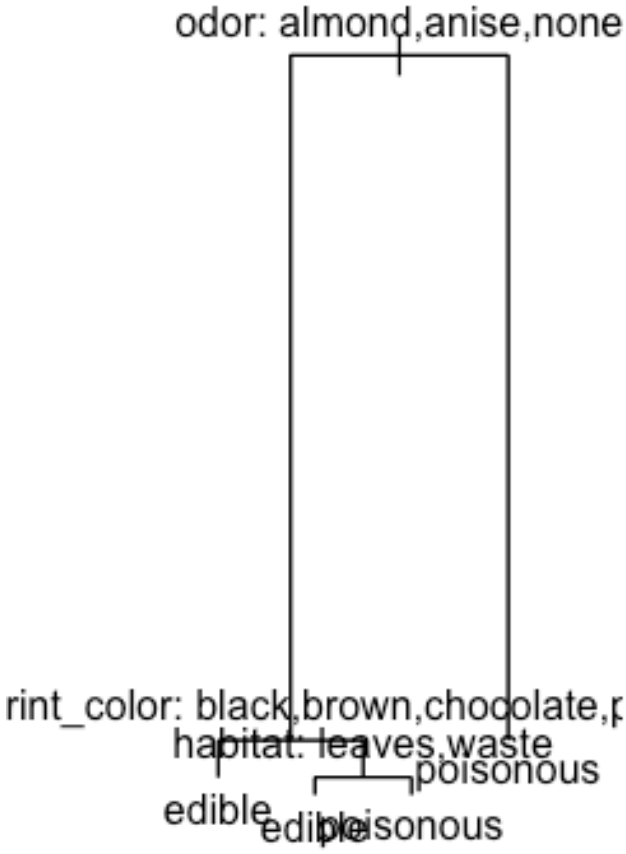
```
test <- mushrooms[-(1:round(2/3*nrow(mushrooms))),]

summary(train)

train <- na.omit(train)


#Q1
# Build an unpruned tree using the training data
tree_mushrooms <- tree(type ~ ., data = train)
summary(tree_mushrooms)


plot(tree_mushrooms)
text(tree_mushrooms, pretty = 0)
## Deviance: -2 * sum_Classes(sum_Nodes(obs_CN * log(phat_CN)))
tree_mushrooms

## Pruning
set.seed(1)

help("cv.tree")
cv.mushrooms <- cv.tree(tree_mushrooms, FUN = prune.misclass, K=10)
str(cv.mushrooms)
cv.mushrooms

par(mfrow = c(1, 2))
plot(cv.mushrooms$size, cv.mushrooms$dev, type = "b")
plot(cv.mushrooms$k, cv.mushrooms$dev, type = "b")
par(oldpar)
```

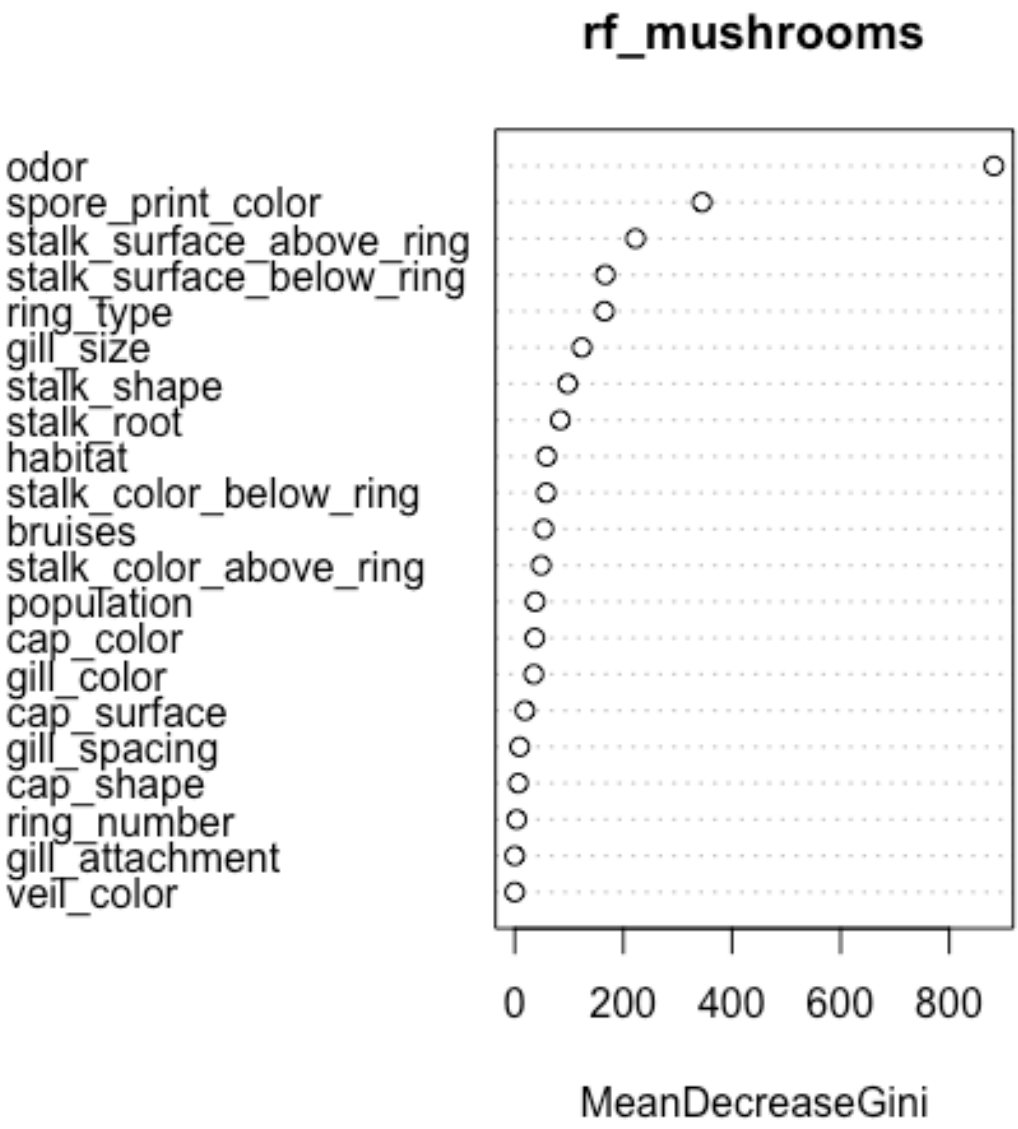
```
## Evaluate pruned tree
prune.mushrooms <- prune.misclass(tree_mushrooms, best = 4)
plot(prune.mushrooms)
text(prune.mushrooms, pretty = 0)



# Q2
# Build a random forest model using the training data
rf_mushrooms <- randomForest(type ~ ., data = train, ntree = 500)

## Importance plot
varImpPlot(rf_mushrooms)

#Q3
table(is.na(train))
train <- na.omit(train)

# Find the optimal mtry
tune_mushrooms <- tuneRF(train[, -1], train[, 1], ntree = 500)
tune_mushrooms$mtry

#mtry = 4  OOB error = 0%
#Searching left ...
#mtry = 2  OOB error = 0%
#NaN 0.05
#Error in if (Improve > improve) { : missing value where TRUE/FALSE
needed




#Q4
```

```
set.seed(1)


## Find the optimal interaction depth

depths <- 1:10

errors <- rep(0, length(depths))


for (i in 1:length(depths)) {

  boost_mushrooms <- gbm(type ~ ., data = train, distribution =
"multinomial",

                         n.trees = 5000, interaction.depth =
depths[i])

  errors[i] <- boost_mushrooms$cv.error[boost_mushrooms$n.trees ==
5000]

}

## Plot the cross-validation error rates

plot(depths, errors, type = "b", xlab = "Interaction Depth", ylab =
"CV Error Rate")



#Q5
```


odor: almond,anise,none
_print_color: black,brown,chocolate,purple
habitat: leaves,waste
edible
edible
poisonous
poisonous


cv.mushrooms$dev
0
500
1000
1500
1.0
4.0
cv.mushrooms$dev
0
500
1000
1500
0


odor: almond,anise,none
rint_color: black,brown,chocolate,p
habitat: leaves,waste
poisonous
edible
edible
poisonous


rf_mushrooms
odor
spore_print_color
stalk_surface_above_ring
stalk_surface_below_ring
ring_type
gill_size
stalk_shape
stalk_root
habitat
stalk_color_below_ring
bruises
stalk_color_above_ring
population
cap_color
gill_color
cap_surface
gill_spacing
cap_shape
ring_number
gill_attachment
veil_color
0
200
400
600
800
MeanDecreaseGini

## Part 2: Regression trees [3 point]

You will be using for this part the dataset **`redwines.csv`** and split it stratified with just 3 matching strata into 2/3 ***training*** data and 1/3 ***test*** data. The dependent variable is **`quality`**.

**Task 6:** Build a pruned regression tree with all feature variables and interpret the pruned tree. Show the pruned and unpruned trees. For model calibration use the ***training*** data-frame. Calculate the model fit for the ***test*** data-frame. [1 point]

**Task 7:** Calibrate for the ***training*** data-frame with all feature variables a random forest model and identify its optimal hyper-parameters **`ntree`** and **`mtry`**. Evaluate the variable importance. Calculate the model fit for the ***test*** data-frame. [1 point]

**Task 8:** Calibrate for the ***training*** data-frame with all feature variables a boosted model and identify its optimal depth hyper-parameter **`interaction.depth`**. Calculate the model fit for the ***test*** data-frame. [1 point]