# Comprehension Check: Caret Package

These exercises take you through an analysis using the **`tissue_gene_expression`** dataset.

## Q1

1/1 point (graded)

Use the `rpart` function to fit a classification tree to the `tissue_gene_expression` dataset. Use the `train` function to estimate the accuracy. Try out `cp` values of `seq(0, 0.1, 0.01)`. Plot the accuracies to report the results of the best model. Set the seed to 1991.

Which value of `cp` gives the highest accuracy?

0.00 ✓ **Answer:** 0

0.00

**Explanation**

The following code can be used to do generate the plot and get the value of `cp`:

```
library(caret)
library(dslabs)
set.seed(1991)
data("tissue_gene_expression")

fit <- with(tissue_gene_expression,
                train(x, y, method = "rpart",
                      tuneGrid = data.frame(cp = seq(0, 0.1, 0.01))))

ggplot(fit)
```

Submit   You have used 1 of 10 attempts

Answers are displayed within the problem

## Q2

1/1 point (graded)

Note that there are only 6 placentas in the dataset. By default, `rpart` requires 20 observations before splitting a node. That means that it is difficult to have a node in which placentas are the majority. Rerun the analysis you did in the exercise in Q1, but this time, allow `rpart` to split any node by using the argument `control = rpart.control(minsplit = 0)`. Look at the confusion matrix again to determine whether the accuracy increases. Again, set the seed to 1991.

What is the accuracy now?

0.9141 ✓ **Answer:** 0.9141

0.9141

**Explanation**

The following code can be used to re-run the analysis and view the confusion matrix:

```
set.seed(1991)
data("tissue_gene_expression")

fit_rpart <- with(tissue_gene_expression,
                  train(x, y, method = "rpart",
                        tuneGrid = data.frame(cp = seq(0, 0.10, 0.01)),
                        control = rpart.control(minsplit = 0)))
ggplot(fit_rpart)
confusionMatrix(fit_rpart)
```

Submit

You have used 1 of 10 attempts

Answers are displayed within the problem

## Q3

1/1 point (graded)

Plot the tree from the best fitting model of the analysis you ran in Q2.

Which gene is at the first split?

- ○ B3GNT4
- ○ CAPN3
- ○ CES2

CFHR4

CLIP3

[✓ GPA33]

HRH1

✓

**Explanation**
The first split is at GPA33 >= 8.794. The following code will give the tree:

```
plot(fit_rpart$finalModel)
text(fit_rpart$finalModel)
```

Submit You have used 1 of 2 attempts

Answers are displayed within the problem

## Q4

1/1 point (graded)
We can see that with just seven genes, we are able to predict the tissue type. Now let's see if we can predict the tissue type with even fewer genes using a Random Forest. Use the `train` function and the `rf` method to train a Random Forest model and save it to an object called `fit`. Try out values of `mtry` ranging from `seq(50, 200, 25)` (you can also explore other values on your own). What `mtry` value maximizes accuracy? To permit small `nodesize` to grow as we did with the classification trees, use the following argument: `nodesize = 1`.

Note: This exercise will take some time to run. If you want to test out your code first, try using smaller values with `ntree`. Set the seed to 1991 again.

What value of `mtry` maximizes accuracy?

100 ✓ **Answer:** 100

100

**Explanation**
The following code can be used to do the analysis:

```
set.seed(1991)
library(randomForest)
fit <- with(tissue_gene_expression,
                train(x, y, method = "rf",
                      nodesize = 1,
                      tuneGrid = data.frame(mtry = seq(50, 200, 25))))

ggplot(fit)
```

Submit You have used 1 of 10 attempts

Answers are displayed within the problem

## Q5

1/1 point (graded)

Use the function `varImp` on the output of `train` and save it to an object called `imp`.

```
imp <- #BLANK
imp
```

What should replace #BLANK in the code above?

Do not include spaces in your answer.

varImp(fit) ✓ **Answer:** varImp(fit)

Submit You have used 1 of 10 attempts

Answers are displayed within the problem

## Q6

1/1 point (graded)

The `rpart` model we ran above produced a tree that used just seven predictors. Extracting the predictor names is not straightforward, but can be done. If the output of the call to train was `fit_rpart`, we can extract the names like this:

```
tree_terms <- as.character(unique(fit_rpart$finalModel$frame$var[!
(fit_rpart$finalModel$frame$var == "<leaf>")]))
tree_terms
```

Calculate the variable importance in the Random Forest call for these seven predictors and examine where they rank.

What is the importance of the CFHR4 gene in the Random Forest call?
Enter a number.

35 ✓ **Answer:** 35.0

35

What is the rank of the CFHR4 gene in the Random Forest call?
Enter a number.

7 ✓ **Answer:** 7

7

**Explanation**
The following code can be used to calculate the rank and importance in the Random Forest call for the predictors from the `rpart` model:

```
data_frame(term = rownames(imp$importance),
                        importance = imp$importance$Overall) %>%
        mutate(rank = rank(-importance)) %>% arrange(desc(importance)) %>%
        filter(term %in% tree_terms)
```

Submit You have used 2 of 10 attempts

Answers are displayed within the problem

Ask your questions or make your comments about Tuning Parameters with Caret here! **Remember, one of the best ways to reinforce your own learning is by explaining something to someone else, so we encourage you to answer each other's questions (without giving away the answers, of course).**

Some reminders:

- Search the discussion board before posting to see if someone else has asked the same thing before asking a new question.
- Please be specific in the title and body of your post regarding which question you're asking about to facilitate answering your question.
- Posting snippets of code is okay, but posting full code solutions is not.
- If you do post snippets of code, please format it as code for readability. If you're not sure how to do this, there are instructions in a pinned post in the "general" discussion forum.

## Discussion: Tuning Parameters with Caret

Show Discussion

**Topic:** Section 5: Classification with more than two classes and the caret package / 5.2: Tuning Parameters with Caret