Midterm GR5206 October 21, 2016

Name (UNI):

Section:

This is the written portion of the midterm. It is expected that this portion should take you 60 minutes to complete, and you will be given 90 minutes to complete this portion. We will collect the written portion at 4:10 PM at which point we will begin the computing portion of the exam. **You may not use computers, cell phones, or any other electronic devices during the written portion of the exam, and if we see you doing this, you will receive a 0 grade.**

Some of the following questions ask you to write code with the goal of producing a certain value. Your code can contain multiple lines (meaning multiple statements), but the last statement of the code should produce the desired result.

Question 1: Resampling

Imagine we have a function `moment.estimates`, which takes in a data vector and returns a vector of the parameter estimates $\hat{\mu}$ and $\hat{\sigma}$ (by calculating the mean and standard deviation of the data vector) and some other data summaries.

```
varCI <- function(x, B = 100, alpha = 0.05) {

  data.fit         <- moment.estimates(x)
  data.params      <- data.fit$params

  boot.ests  <- rep(NA, B)
  for (b in 1:B) {
    boot.sample <- rep(NA, length(x))
    for (i in 1:length(x)) {
      boot.sample[i] <- sample(1:length(x), 1)
    }
    boot.ests[b] <- moment.estimates(x[boot.sample])$params[2]
  }
  diff_ests <- boot.ests - data.params[2]

  CU <- data.params[2] + quantile(diff_ests, 1-alpha/2)
  CL <- data.params[2] + quantile(diff_ests, alpha/2)
  return(c(CL, CU))
}
```

(a) Explain in one sentence what this function calculates.

Solution:

Bootstrap confidence interval Lab 3

(b) Replace the inner loop with a single line of code by replacing the blank line in the following.

Solution:

```
normal.var.95CI <- function(x, B = 100) {

  data.fit          <- normal.estimates(x)
  data.params       <- data.fit$params

  boot.ests  <- rep(NA, B)
  for (b in 1:B) {
    boot.sample <- ____________________________________________
    boot.ests[b] <- normal.estimates(x[boot.sample])$params[2]
  }
  diff_ests <- boot.ests - data.params[2]

  CU <- data.params[2] + quantile(diff_ests, 0.975)
  CL <- data.params[2] + quantile(diff_ests, 0.025)
  return(c(CL, CU))
}
```

(c) We have the following information:

```
> x
[1]  0.32 -0.73  0.14  0.02 -0.43 -0.49
> boot.sample
[1] 6 6 3 2 6 4
```

For the above values of `x` and `boot.sample`, describe the calculation that takes place with the line of code

```
normal.estimates(x[boot.sample])$params[2]
```

by filling in the sentence written in the solution line. I expect a sentence like, 'the code calculates the `minimum` of the values `(1, 2, 3, 4)`' (though this is obviously not the answer).

Solution:

The code `normal.estimates(x[boot.sample])$params[2]` calculates the

____

of the values

____.

Question 2: Regression

Recall the `iris` dataset we studied in class with the following variables:

- `Sepal.Length`: The iris sepal length in centimeters.
- `Sepal.Width`: The iris sepal width centimeters.
- `Petal.Length`: The iris petal length centimeters.
- `Petal.Width`: The iris petal width centimeters.
- `Setosa`: A dummy variable equal to 1 if the iris species is Setosa and 0 otherwise.

A student uses linear regression to predict iris sepal width with iris sepal length and whether or not the iris is of the setosa species as predictors. The output of the regression model is the following:

```
> lm0
Call:
lm(formula = iris$Sepal.Length ~ iris$Sepal.Width + iris$Setosa)

Coefficients:
     (Intercept)  iris$Sepal.Width       iris$Setosa
          3.6           0.5            -1.8
```

(a) What is the estimated model? Write the form of the estimated model using the above output.

Solution:

(b) What sepal length does the model predict for an iris of the setosa species with a sepal width of 4.0 centimeters?

Solution:

(c) The student calls the above model ‘lm0’, and prints out summaries of the squared residuals, the residuals, and fitted values of the model as follows:

```
> summary(residuals(lm0)^2)
    Min.   1st Qu.   Median     Mean    3rd Qu.   Max.
  0.000     0.030     0.090     0.224    0.227   2.870

> summary(residuals(lm0))
    Min.   1st Qu.   Median     Mean    3rd Qu.   Max.
  -1.012    -0.300   -0.051     0.000    0.255    1.694

> summary(fitted(lm0))
    Min.   1st Qu.   Median     Mean    3rd Qu.   Max.
  3.944     5.262     6.100     5.843    6.383    7.136
```

Recall that the fitted values (or predicted values) are the values predicted by the model for each set of predictors in the dataset and the residuals are the differences between the actual values and the fitted values. Using this output, report the training error and the test error of the model. If the output doesn’t provide these values, explain in a single sentence why not.

Solution:

Question 3: Common Errors

Below you will find chunks of R code and some of them contain bugs (errors). Use your best judgement to determine which code statements won't work, meaning they won't do what the author probably wanted. For the lines with errors, write a corrected version of the code. For the ones without errors, write CORRECT.

Assume we have a data frame called HomeworkData which has two variables (columns) labeled Grade and Name. The data frame holds information on student grades from the first homework assignment.

(a).

```
HomeworkGrades  <- HomeworkData["Grade"]
AverageGrade    <- mean(HomeworkGrades)
```

Solution:

(b).

```
sort(HomeworkData$Grade, decreasing = TRUE)
HighestGrade <- HomeworkData$Grade[1]
```

Solution:

(c).

```
HighGrade    <- 90
HighGrades   <- HomeworkData["Grade" >= HighGrade, ]
TopStudents  <- HighGrades[, "Name"]
```

Solution:

(d).

```
HighGrade   <- 90
HighScorers <- HomeworkData$Grade >= HighGrade

HomeworkData$Message <- rep(NA, nrow(HomeworkData))

HomeworkData$Message[HighScorers]  <- "Good Work!"
HomeworkData$Message[!HighScorers] <- "Ask for help on Piazza!"
```

Solution:

Question 4: Writing Functions

(a) Write a function `is.short` which takes as input a single argument, a vector named `the.vector`, and returns a logical value: `TRUE` if the vector has fewer than 10 elements and `FALSE` if the vector has ten or more elements.

Solution:

(b) Even if you didn't complete part (a), assume that we have a function `is.short` as described above. Suppose we have two vectors, `vector.A` and `vector.B`. Write `R` code that returns `TRUE` if the concatenation (the combination) of `vector.A` and `vector.B` is short (meaning it has fewer than 10 elements) and `FALSE` otherwise.

Solution:

Question 5: Manipulating Data Frames, Control Statements + Vectorized Functions

Suppose I have a data frame called `StudentData` which has n rows and a five columns labeled `Name`, `HW1`, `HW2`, `HW3`, and `HW4`, containing each student's name and scores on four homework assignments.

(a) We would like to add a new column to the data frame containing the lowest score each student has scored in their homework assignments, which we'll call `LowestScore`. Assume that the data frame `StudentData` already exists, that its columns are named `Name`, `HW1`, `HW2`, `HW3`, and `HW4`, and none of the homework scores are missing (no NA values). Write `R` code to create the new column `LowestScore`. **To answer this question, you may use either a `for` loop or the `apply()` function, but not both.**

HINT: When I look at `R` help for `apply()` family functions, I see that take the following input:

```
apply(X, MARGIN, FUNCTION)
lapply(X, FUNCTION)
sapply(X, FUNCTION)
tapply(X, INDEX, FUNCTION)
```

Solution:

(b) Write code that does the same thing as in part (a), but do it the way you didn't do it last time – use a `for` loop if you used `apply()` and vice versa. If you used neither previously, write code that uses either a `for` loop or `apply()`, but not both.

Solution: