BIOS 545 - Homework #4

Due 11:59 PM on May 1, 2020

Instructions

Submit your answer sheet in a file named using the convention of `BIOS545_LastName_FirstName_HW4.R` (or `BIOS545_LastName_FirstName_HW4.Rmd`) to Canvas. If you submit a .Rmd file, consider to submit the html file at the same time. You should use RStudio to create the file.

- Due by 11:59 PM on 05/1/20. A penalty of 15% per day late applies to submissions that arrive after the due date and time. After three days, late submissions will no longer be accepted and a score of 0 will be rendered.
- In submitting your homework you are indicating that all work is your own. Please do not develop your solutions collaboratively or share/accept code with other students (current or former).
- Putting comments in your code is helpful. At least we can understand what you were trying or intending to do which makes it easier for us to award partial credit even if the code isn't working as intended.

1) Palindromes (20 points)

Let's consider some special types of words called "palindromes". An example would be a word like "civic" which is spelled the same forwards as it is backwards. We would also consider the word "noel" to be a palindrome of "leon". The same could be said of the words "saw" and "was".

Write a function called *has.palindrome* that, given a word and a vector, indicates whether the vector contains a palindrome for the word.

```
words <- c("leon", "yam", "store", "cat", "saw")
```

Here is a shell:

```
has.palindrome <- function(inp, words) {

# Function to determine if a word has palindromes in a vector of words
#
# INPUT: inp - a single word in quotes (e.g. "cat")
#        words - a characer vector of words
#
# OUTPUT: retval - a logical value - TRUE or FALSE that indicates
#                  if a word has a palindrome in the input vector

  return(retval)

}
```

To do this figure out a way to compare the input word to the words in the vector. We went over some functions that will help you do this.

```
has.palindrome("dog", words)     # no palindrome
```

```
## [1] FALSE
```

```
has.palindrome("noel", words)    # is a palindrome for "leon"
```

```
## [1] TRUE
```

```
has.palindrome("was", words)     # is a palindrome for "deal"
```

```
## [1] TRUE
```

2) Summarize simulation results (20 points)

Statisticians often run simulations to test the performance of a new statistical method or compare several methods against each other. The beauty of simulations is that you know what the "truth" is: the true slope for a linear regression, the true difference in group means for a t-test, and so on. Simulations are great for assessing validity and efficiency of statistical methods.

Write a function called *summarize_sim* that takes the following inputs:

Input	Default	Description
`point.estimates`	(none)	Numeric matrix where each column contains point estimates for a particular method.
`truth`	(none)	Numeric value specifying true value of parameter being estimated.
`decimals`	3	Numeric value specifying number of decimals to print.

And returns a numeric matrix with 3 columns:

- `Mean bias` = Mean of the point estimates minus the truth.
- `SD` = Standard deviation of point estimates.
- `MSE` = Mean squared error, defined as $\frac{1}{n}\sum_{i=1}^{n}(x_i - \text{truth})^2$.

For example, suppose we wish to estimate the mean of a normally distributed variable, and we are comparing two estimators: the sample mean, and the sample median. Both should be unbiased for μ, but the mean should be more efficient.

```
# 1,000 trials of 30 draws from N(0, 1)
set.seed(123)
estimates <- matrix(NA, ncol = 2, nrow = 1000,
             dimnames = list(NULL, c("Sample mean", "Sample median")))
for (ii in 1: 1000) {
  x30 <- rnorm(n = 30)
  estimates[ii, ] <- c(mean(x30), median(x30))
}
```

```
summarize_sim(estimates, truth = 0)
```

```
##                Mean bias    SD   MSE
## Sample mean       -0.006 0.178 0.032
## Sample median     -0.009 0.218 0.048
```

To get full credit, add error checking to ensure that `point.estimates` is a matrix with numeric values, `truth` is numeric, and `decimals` is a non-negative integer. The function should work for any number of estimators (not just 2), with row names specifying the name of each estimator.

3) Create a Package - 30 Points and Bonus 5 Points

With respect to the *has.palindrome* function from Question 1 - Create an R package called **wordtools** that contains the *has.palindrome* function. Apply the methods described in the package lecture and lab to create a package that we can install on our systems. Your work will result in a file named something like:

wordtools_1.0.tar.gz or wordtools_1.0.zip (version number might vary on different systems). The package must:

- Be installable and loadable in R. (10 points)
- Contain a function called *has.palindrome* that works as described in Question 1 (10 points)
- Include a function help file for the *has.plaindrome* function, with the following sections: name, title, usage, arguments, value, description, details, examples, references, seealso, author, value, and example. (10 points)
- Create a html/pdf vignette to your package, with the following information: name, title, descriptions, an example to *has.plaindrome*. You vignette should be able to appear if vignette("wordtools") or browseVignettes("wordtools") is called. (Bonus 5 points)

4) Statistical Analysis - 30 Points

In this problem, you will use data from the National Health and Nutrition Examination Survey (NHANES) to explore a research question of epidemiological interest. You can load the data into R by running:

```
load(url("https://github.com/vandomed/vandomed.github.io/raw/master/nhanes.rda"))
```

We expect older adults to become less active and more sedentary as they age. In the NHANES dataset, there is a variable called `sed_min` that has the average number of minutes per day spent sedentary for each subject, measured over a 1-week period. We can use that variable to estimate how much extra sedentary time is associated with each 1-year increase in age. As you can see, we already expect it will be a positive association. The older people get, the more time they tend to spend sedentary.

We also want to see if there are any differences by race/ethnicity, which we will analyze as 3 groups: non-Hispanic white (eth = 1), non-Hispanic black (eth = 2), and other (eth = 3 or 4 or 5).

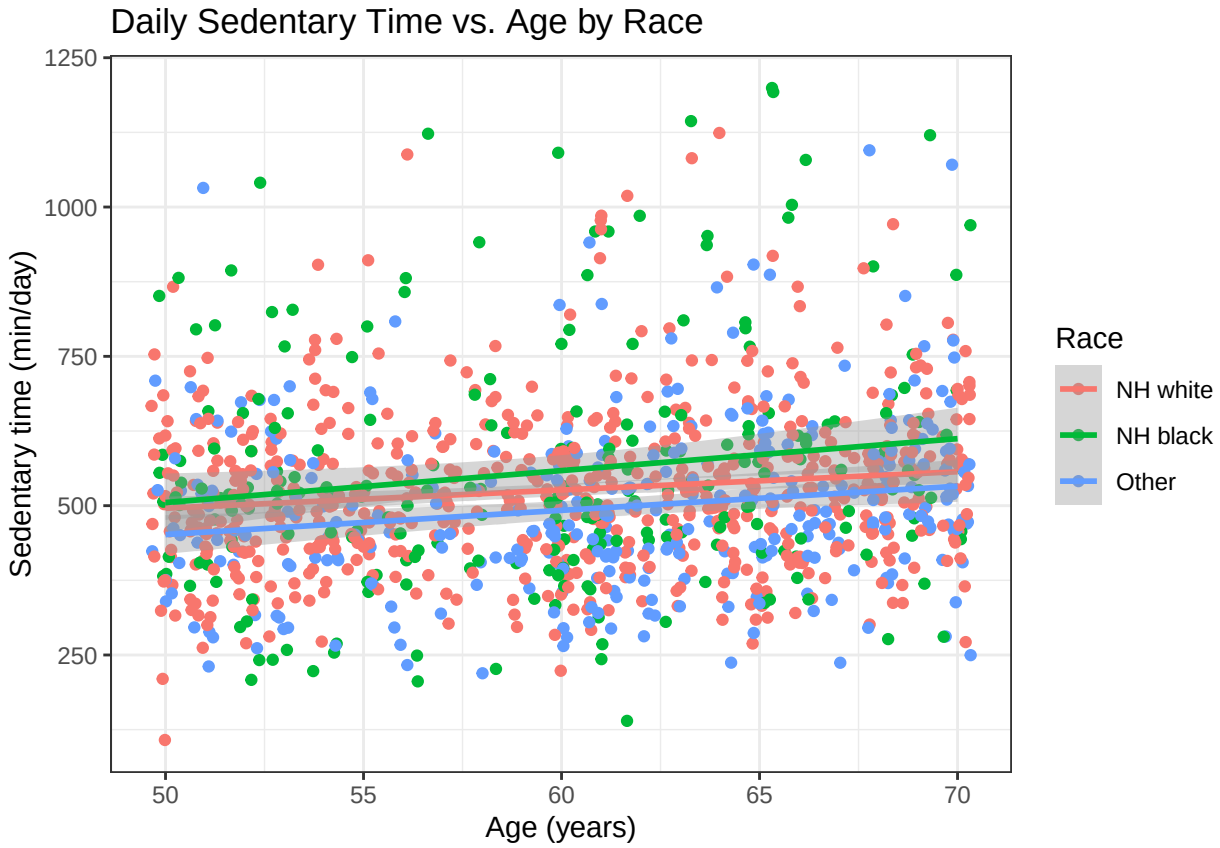
Please complete the following:

- Subset the `nhanes` data frame for the age range of interest, let's say 50-70.
- Create a scatterplot of `sed_min` vs. `age`, with data points color-coded according to race/ethnicity, and with color-coded regression lines for each race/ethnicity (see Figure below; I used *geom_jitter* to make the ages plot a little nicer).
- Fit a regression model for `sed_min` vs. `age` and your 3-level race/ethnicity variable, allowing for the relationship between `sed_min` and `age` to differ for each race/ethnicity.
- Calculate the estimated slopes for each race/ethnicity according to the fitted model.

You can use any approach you like, but personally I would suggest this 3-step process:

1. Use **dplyr** pipes to prepare your dataset (subset data, calculate variables).
2. Use **ggplot2** to create the scatterplot.
3. Fit the regression model and calculate the slopes.

Don't worry about writing a summary of your results or anything like that. All you need to do is subset the data, generate the plot, fit the regression model, and calculate the 3 slopes.




```
## [1] "Estimated slope of sed_min versus age for Race = NH white"
```

```
##      age
## 3.060662
```

```
## [1] "Estimated slope of sed_min versus age for Race = NH black"
```

```
##      age
## 5.346282
```

```
## [1] "Estimated slope of sed_min versus age for Race = Other"
```

```
##    age
## 4.0533
```