STA234 HW8

Derin Gezgin

2025-05-06

100/100

Importing the Necessary Libraries

```
library(tidyverse)
library(ggmap)
library(gt)
library(pROC)
library(gtExtras)
library(nycflights13)
```

Problem 1 [25 points] *Must Use Loops*

1.(a)

For the `diamonds` data in `ggplot2`, calculate the average price for each cut using a `for` loop and assign the values to a dataframe called `average_prices`.

```
# Saving the cut levels
cut_levels = levels(diamonds$cut)

# Creating an empty dataframe to store
# the average prices
average_prices = data.frame(
    cut = character(),
    avg_price = numeric()
    )

# For each cut level
for (this_cut in cut_levels) {
    # Calculate the mean price
    mean_price = mean(diamonds$price[diamonds$cut == this_cut])

    # Add this mean to the existing dataframe
    average_prices = rbind(
        average_prices,
        data.frame(cut = this_cut,
                   avg_price = mean_price))
}

average_prices
```

```
##         cut avg_price
## 1      Fair  4358.758
## 2      Good  3928.864
## 3 Very Good  3981.760
## 4   Premium  4584.258
## 5     Ideal  3457.542
```

1.(b)

Use a `while` loop to find the count of diamonds with a `carat` greater than a specified `threshold` (e.g. 2 carats) until `count` exceeds 100.

```
threshold = 2  # Threshold value
current_index = 1  # Point in the dataframe
count = 0  # Variable to keep track of count

while (current_index <= nrow(diamonds) && count <= 100) {
    # If the carat size is larger than the threshold, increment the count
    if (diamonds$carat[current_index] > threshold) { count = count + 1 }
    # Increment the index to pass to the next row
    current_index = current_index + 1
}

print(paste("Diamonds with a carat larger than",
            threshold,
            "exceeded 100 in row",
            current_index))
```

```
## [1] "Diamonds with a carat larger than 2 exceeded 100 in row 22147"
```

1.(c)

Use a `repeat` loop to determine the total price of diamonds until you reach a certain number of diamonds or a specific budget.

```
# Stopping conditions
max_diamonds = 200  # For diamond count
budget_limit = 50000  # For budget

# Starting parameters
current_index = 1
total_price = 0
count = 0

repeat {
    # If we arrive to the end of the dataframe, end the loop
    if (current_index > nrow(diamonds)) break

    # Update count and price
    total_price = total_price + diamonds$price[current_index]
```

```
    count = count + 1

    # If any of the stop conditions are met, end the loop
    if (count >= max_diamonds || total_price >= budget_limit) break
    current_index = current_index + 1
}

print(paste("Total price:", total_price))
```

```
## [1] "Total price: 50063"
```

```
print(paste("Diamonds used:", count))
```

```
## [1] "Diamonds used: 94"
```

Problem 2 [25 points] *Function Only*

Using the flights data from package `nycflights13`, write a function to group data by a given `variable1` (input) and show minimum, maximum, and average of another `variable2` (input). Set default for `variable1` as `carrier` and `variable2` as `distance`. Call function without any inputs and show output. Call function with `variable1` as `origin` and `variable2` as `dep_delay` and show the output.

```
flightSummary = function(variable1 = "carrier",
                         variable2 = "distance") {
    flights %>%
        group_by(.data[[variable1]]) %>%
        summarise(
            min = min(.data[[variable2]], na.rm = TRUE),
            max = max(.data[[variable2]], na.rm = TRUE),
            avg = mean(.data[[variable2]], na.rm = TRUE),
        )
}
```

Calling the functions without any inputs

```
flightSummary()
```

```
## # A tibble: 16 × 4
##    carrier   min   max   avg
##    <chr>   <dbl> <dbl> <dbl>
##  1 9E         94  1587  530.
##  2 AA        187  2586 1340.
##  3 AS       2402  2402 2402
##  4 B6        173  2586 1069.
##  5 DL         94  2586 1237.
##  6 EV         80  1389  563.
##  7 F9       1620  1620 1620
##  8 FL        397   762  665.
```

```
##  9 HA       4983  4983 4983
## 10 MQ        184  1147  570.
## 11 OO        229  1008  501.
## 12 UA        116  4963 1529.
## 13 US         17  2153  553.
## 14 VX       2248  2586 2499.
## 15 WN        169  2133  996.
## 16 YV         96   544  375.
```

Calling the function with specific inputs.

```
flightSummary("origin", "dep_delay")
```

```
## # A tibble: 3 × 4
##   origin   min   max   avg
##   <chr>  <dbl> <dbl> <dbl>
## 1 EWR      -25  1126  15.1
## 2 JFK      -43  1301  12.1
## 3 LGA      -33   911  10.3
```

Problem 3 [25 points]

Using the code (either yours or mine) from hw-6, where you created the function called `madness_check` and save it in a separate R script as `madnessfunction.R`, do the following:

3.(a)

Source the function.

```
source("madnessfunction.R")
```

3.(b)

Call the function for all values of `surveydata` (from hw-6) using functionals to do the following:

suppose we surveyed 100 people and we get `surveydata` (same as hw-6). Create a dataframe called `surveyresult` which includes two columns: first column named as `surveydata`, is the original survey data; second column named as `madness`, is the madness result using the scales from question a.

```
set.seed(100)
surveydata = sample(1:11, 100, replace = T)

surveyresult = data.frame(surveydata = surveydata,
                          madness = sapply(surveydata, madness_check))
```

```
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Very mad"
## [1] "Light mad"
## [1] "Light mad"
## [1] "Light mad"
## [1] "Very mad"
## [1] "Light mad"
## [1] "Very mad"
## [1] "Light mad"
## [1] "Light mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Very mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Very mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Very mad"
```

```
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Very mad"
## [1] "Light mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Light mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Moderate mad"
## [1] "Light mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Moderate mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Light mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Very mad"
## [1] "Moderate mad"
## [1] "Moderate mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Light mad"
## [1] "Very mad"
## [1] "Rate your madness using a number from 0 to 10!"
## [1] "Very mad"
```

Show the first six and last six rows of your data `surveyresult`.

```
head(surveyresult)
```

```
##   surveydata      madness
## 1         10     Very mad
## 2          7 Moderate mad
## 3          6 Moderate mad
## 4          3    Light mad
## 5          9     Very mad
## 6         10     Very mad
```

```
tail(surveyresult)
```

```
##     surveydata                                        madness
## 95           5                                  Moderate mad
## 96          11 Rate your madness using a number from 0 to 10!
## 97           2                                     Light mad
## 98           8                                      Very mad
## 99          11 Rate your madness using a number from 0 to 10!
## 100          8                                      Very mad
```

3.(c)

Show how would you be able to repeat generating the same survey data every time we run the code?

In the current version of the code I set the random seed to a specific number, which means that the random generator in the computer has a fixed seed.

Problem 4 [25 points]

Get the built-in data set `state.x77` as:

```
library(MASS)
data(state)
```

This data consists of eight columns describing the 50 U.S. states in 1977, and the data sources are U.S. Department of Commerce, Bureau of the Census (1977) Statistical Abstract of the United States; U.S. Department of Commerce, Bureau of the Census (1977) County and City Data Book. Read more details about the data set named as `state.x77` which you need for the following parts:

Make sure `state.x77` is a dataframe, if not set it to a dataframe format.

```
class(state.x77)
```

```
## [1] "matrix" "array"
```

We can see that it is not in the dataframe format

```
state_df = as.data.frame(state.x77)
```

We can convert it to a dataframe and check it again

```
class(state_df)
```

```
## [1] "data.frame"
```

4.(a) [2 points]

Use `apply()` function to `state.x77` data to find the median of each column.

```
medians = apply(X = state_df,
                MARGIN = 2,
                FUN = median)
medians
```

```
## Population     Income Illiteracy   Life Exp     Murder    HS Grad
Frost
##   2838.500   4519.000      0.950     70.675      6.850     53.250
114.500
##       Area
##  54277.000
```

4.(b) [8 points]

Use the appropriate functional to get a data.frame named as `summary_inf`, which includes the median, minimum and maximum value of each column. And the column names for `summary_inf` are Median, Min, Max respectively.

```
summary_inf = data.frame(
    Median = sapply(state_df, median),
    Min = sapply(state_df, min),
    Max = sapply(state_df, max)
    )

summary_inf
```

```
##                Median     Min      Max
## Population  2838.500  365.00  21198.0
## Income      4519.000 3098.00   6315.0
## Illiteracy     0.950    0.50      2.8
## Life Exp      70.675   67.96     73.6
## Murder         6.850    1.40     15.1
## HS Grad       53.250   37.80     67.3
## Frost        114.500    0.00    188.0
## Area       54277.000 1049.00 566432.0
```

4.(c) [10 points]

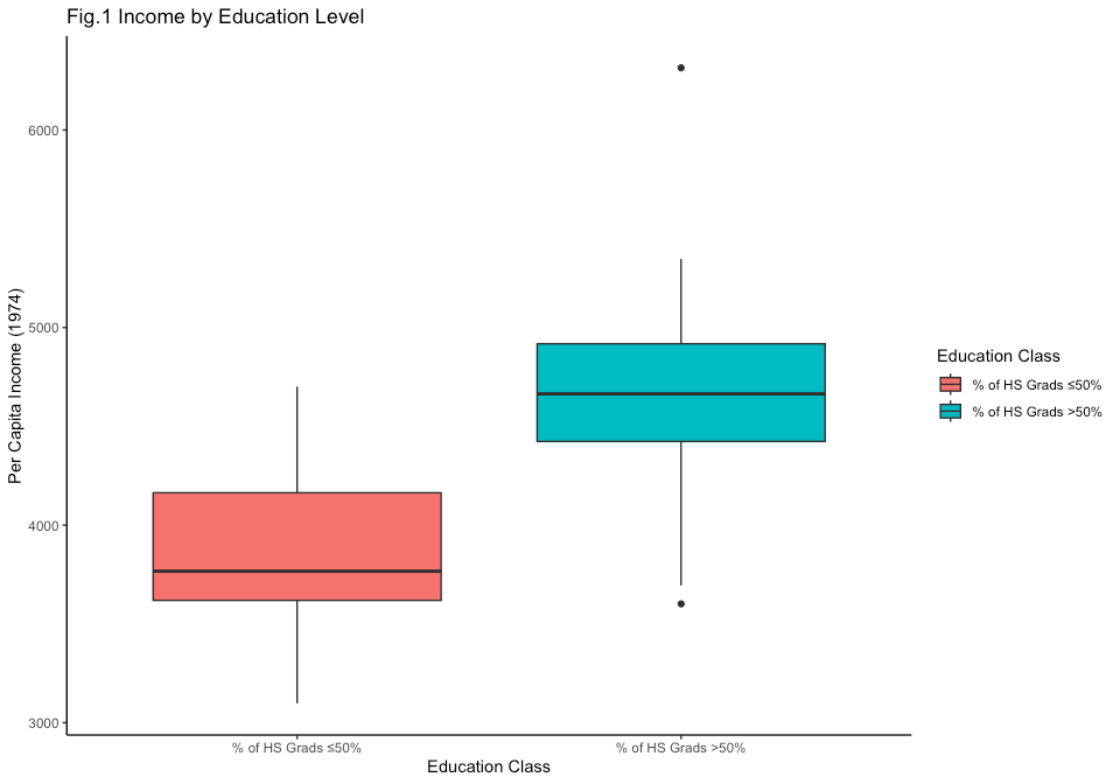
Define a dummy variable named as `education` based on `HS Grad`: If a state has more than 50% of high-school graduates, then `education=1`, if not, `education=0`.

```
state_df = state_df %>%
    mutate(education = ifelse(test = `HS Grad` > 50,
                              yes = 1,
                              no = 0))

# Converting to a factor for better visualization
state_df$education = factor(state_df$education,
                            levels = c(0, 1),
                            labels = c("% of HS Grads ≤50%",
                                       "% of HS Grads >50%"))
```

Then use boxplots to show the following:

1. if there are big differences in `Income` for two groups of education data;

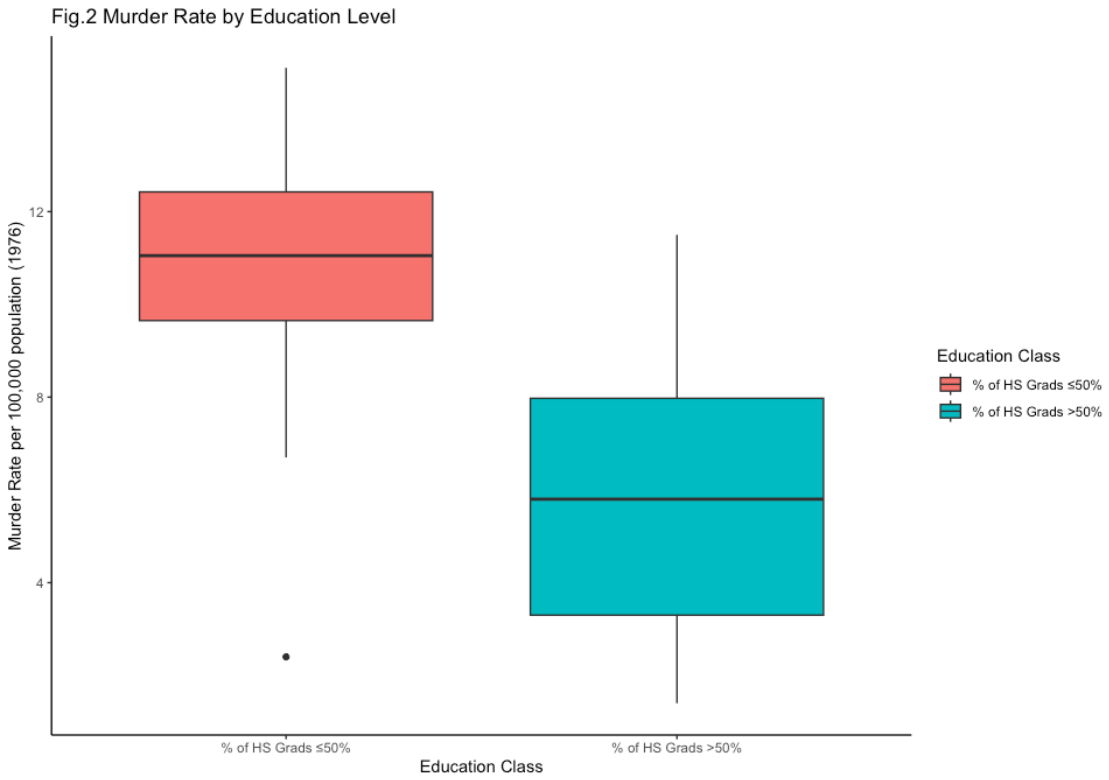
```
incomeEducationPlot = ggplot(data = state_df,
                             aes(x = education,
                                 y = Income)) +
    geom_boxplot(aes(fill = state_df$education)) +
    labs(title = "Fig.1 Income by Education Level",
         x = "Education Class",
         y = "Per Capita Income (1974)",
         fill = "Education Class") +
    theme_classic()

incomeEducationPlot
```



From this box-plot, we can see that there is a significant difference between the per capita income for the two education groups. The group that has the states with less than 50% high school graduates in the population has a significantly lower per capita income compared to the group that has more than 50% high school graduates.

If there are big differences of Murder for two groups of education data.

```
murderEducationPlot = ggplot(data = state_df,
                             aes(x = factor(education),
                                 y = Murder)) +
    geom_boxplot(aes(fill = state_df$education)) +
    labs(title = "Fig.2 Murder Rate by Education Level",
         y = "Murder Rate per 100,000 population (1976)",
         x = "Education Class",
         fill = "Education Class") +
    theme_classic()

murderEducationPlot
```

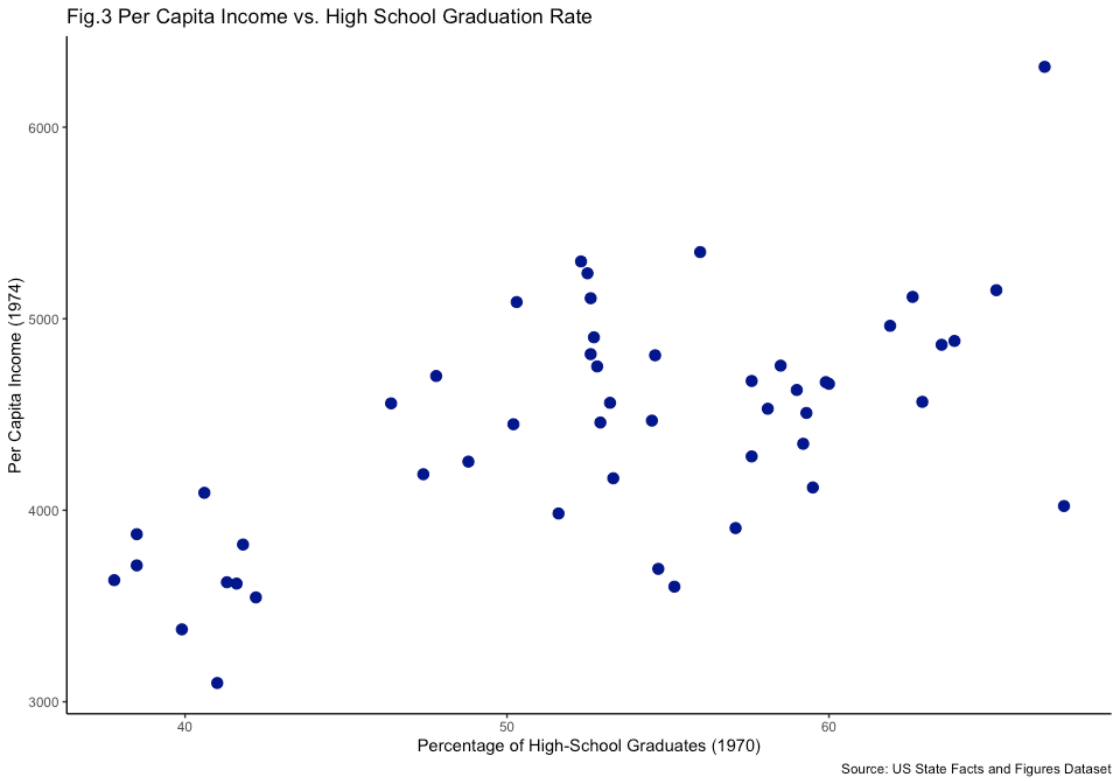


We can see a reverse trend here where the group that has the states with less than 50% high school graduation rate has a significantly higher murder rate compared to the states with more than 50% high school graduation rate. At the same time, the >50% group has a significantly higher variance compared to the ≤50% group.

4.(d) [5 points]

Create a scatter plot of `Income` against `HS Grad`. Describe your findings from the scatter plot.

```
incomeHSGradPlot = ggplot(data = state_df,
                          aes(x = `HS Grad`,
                              y = Income)) +
    geom_point(color = "darkblue",
               size = 3) +
    labs(title = "Fig.3 Per Capita Income vs. High School Graduation Rate",
         x = "Percentage of High-School Graduates (1970)",
         y = "Per Capita Income (1974)",
         caption = "Source: US State Facts and Figures Dataset") +
    theme_classic()

incomeHSGradPlot
```



In this scatterplot, we can definitely see an upward trend. As the percentage of high school graduates increase, the per capita income also increases. While a line fit might not have a very good R-squared value due to the spread-out nature of the data, we can still see the positive upward trend in the data.

Project Problem [100 points]

This counts for presentation

P.(a) [25 points]

Explain what your project is about, share your research questions and justify why are these research questions are valid using existing research in the field? Share research papers as a list of references (See writing examples from assignments 6 and 7)

Traffic stops are a regular part of our lives, in fact, more than 20 million Americans are stopped each year in the traffic (Pierson et al., 2020). Traffic stops are one of the most common ways of public-police interaction. As police officers conduct these stops, the decision-making process comes down to human judgment, which certainly comes with a certain type of bias. There have been many research projects that focuses on possible bias factors in traffic stops. Most of these studies found that the race of the driver is an important factor influencing the likelihood of being stopped and the outcome of the stop.