DATA-613: Exam 1

Cody Meagher

2023-06-11

```r
# Loading libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(epiDisplay)
```

```
## Loading required package: foreign
## Loading required package: survival
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## Loading required package: nnet
##
## Attaching package: 'epiDisplay'
##
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```r
library(data.table)
```

```
##
## Attaching package: 'data.table'
##
## The following objects are masked from 'package:lubridate':
```

```
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
##
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
##
## The following object is masked from 'package:purrr':
##
##     transpose
```

```r
library(Sleuth3)
```

Problem 1

Read the help page of the gss_cat data frame from the forcast package.

```r
# Creating data frame in environment
gss <- forcats::gss_cat
```

1. What are the variables?

The variables in gss_cat are **year, age, marital, race, rincome, partyid, relig, denom, and tvhours.**

2. What is the class of gss_cat?

```r
# Checking class of the data frame
class(gss)
```

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

gss_cat is a tibble, a subclass of a data frame.

3. What is the class of each variables?

Year, age, and tvhours are numeric. Marital, race, rincome, partyid, relig, and denom are factors (categorical)..

4. Reorder the levels of the "relig" variable so the levels are in alphabetical order. Write code that shows the order has been changed. Change order to descending.

```r
# Reordering relig in alphabetical order
gss$relig <- factor(as.character(gss$relig))

# Checking for alphabetical order
levels(gss$relig)
```

```
##  [1] "Buddhism"                "Catholic"
##  [3] "Christian"               "Don't know"
##  [5] "Hinduism"                "Inter-nondenominational"
##  [7] "Jewish"                  "Moslem/islam"
##  [9] "Native american"         "No answer"
## [11] "None"                    "Orthodox-christian"
## [13] "Other"                   "Other eastern"
## [15] "Protestant"
```

```r
# Changing to descending order
gss$relig <- factor(gss$relig, levels(gss$relig)[15:1])

# Checking for descending order
levels(gss$relig)
```

```
##  [1] "Protestant"              "Other eastern"
##  [3] "Other"                   "Orthodox-christian"
##  [5] "None"                    "No answer"
##  [7] "Native american"         "Moslem/islam"
##  [9] "Jewish"                  "Inter-nondenominational"
## [11] "Hinduism"                "Don't know"
## [13] "Christian"               "Catholic"
## [15] "Buddhism"
```

5. Find the frequency of each category.

```r
# Producing a table showing frequencies of each category of relig
# using EpiDisplay package which produces a cleaner table
# table(gss_cat$relig) would also produce the frequencies
tab1(gss_cat$relig, graph = F)
```

```
## gss_cat$relig :
##                         Frequency Percent Cum. percent
## No answer                      93     0.4          0.4
## Don't know                     15     0.1          0.5
## Inter-nondenominational       109     0.5          1.0
## Native american                23     0.1          1.1
## Christian                     689     3.2          4.3
## Orthodox-christian             95     0.4          4.8
## Moslem/islam                  104     0.5          5.3
## Other eastern                  32     0.1          5.4
## Hinduism                       71     0.3          5.7
## Buddhism                      147     0.7          6.4
## Other                         224     1.0          7.5
## None                         3523    16.4         23.9
## Jewish                        388     1.8         25.7
## Catholic                     5124    23.9         49.5
## Protestant                  10846    50.5        100.0
## Not applicable                  0     0.0        100.0
##   Total                     21483   100.0        100.0
```

6. Put levels in descending order of how frequently each level occurs in the data.

```r
# Ordering levels by frequency
gss$relig <- fct_infreq(gss$relig)

# Checking for descending order
levels(gss$relig)
```

```
##  [1] "Protestant"              "Catholic"
##  [3] "None"                    "Christian"
##  [5] "Jewish"                  "Other"
##  [7] "Buddhism"                "Inter-nondenominational"
##  [9] "Moslem/islam"            "Orthodox-christian"
## [11] "No answer"               "Hinduism"
## [13] "Other eastern"           "Native american"
## [15] "Don't know"
```

7. Modify the factor levels of marital to be abbreviations of their long-names. For example, "Divorced" can just be "D".

```r
# Abbreviating levels of marital variable
levels(gss$marital) <- abbreviate(levels(gss$marital), minlength = 1,
                                  use.classes = F, )

# Checking abbreviations
levels(gss$marital)
```

```
## [1] "Na" "Nm" "S"  "D"  "W"  "M"
```

Problem 2

The first two numbers of the Fibonacci Sequence are 0 and 1. Each succeeding number is the sum of the previous two numbers in the sequence. For example, the third element is $1 = 0 + 1$, while the fourth elements is $2 = 1 + 1$, and the fifth element is $3 = 2 + 1$.

1. Use a for loop to calculate the first 100 Fibonacci Numbers.

```r
# Creating vector to use in for loop
fibo <- numeric(100)
fibo[1] <- 0
fibo[2] <- 1

# For loop telling R to modify the 3rd value in the vector to the last
# by adding the previous two numbers together
for (i in 3:100)
{
  fibo[i] = fibo[i-1]+fibo[i-2]
}

# Printing resulting vector
print(fibo)
```

```
##   [1] 0.000000e+00 1.000000e+00 1.000000e+00 2.000000e+00 3.000000e+00
##   [6] 5.000000e+00 8.000000e+00 1.300000e+01 2.100000e+01 3.400000e+01
##  [11] 5.500000e+01 8.900000e+01 1.440000e+02 2.330000e+02 3.770000e+02
##  [16] 6.100000e+02 9.870000e+02 1.597000e+03 2.584000e+03 4.181000e+03
##  [21] 6.765000e+03 1.094600e+04 1.771100e+04 2.865700e+04 4.636800e+04
##  [26] 7.502500e+04 1.213930e+05 1.964180e+05 3.178110e+05 5.142290e+05
##  [31] 8.320400e+05 1.346269e+06 2.178309e+06 3.524578e+06 5.702887e+06
##  [36] 9.227465e+06 1.493035e+07 2.415782e+07 3.908817e+07 6.324599e+07
##  [41] 1.023342e+08 1.655801e+08 2.679143e+08 4.334944e+08 7.014087e+08
##  [46] 1.134903e+09 1.836312e+09 2.971215e+09 4.807527e+09 7.778742e+09
##  [51] 1.258627e+10 2.036501e+10 3.295128e+10 5.331629e+10 8.626757e+10
##  [56] 1.395839e+11 2.258514e+11 3.654353e+11 5.912867e+11 9.567220e+11
##  [61] 1.548009e+12 2.504731e+12 4.052740e+12 6.557470e+12 1.061021e+13
##  [66] 1.716768e+13 2.777789e+13 4.494557e+13 7.272346e+13 1.176690e+14
##  [71] 1.903925e+14 3.080615e+14 4.984540e+14 8.065155e+14 1.304970e+15
##  [76] 2.111485e+15 3.416455e+15 5.527940e+15 8.944394e+15 1.447233e+16
##  [81] 2.341673e+16 3.788906e+16 6.130579e+16 9.919485e+16 1.605006e+17
##  [86] 2.596955e+17 4.201961e+17 6.798916e+17 1.100088e+18 1.779979e+18
##  [91] 2.880067e+18 4.660047e+18 7.540114e+18 1.220016e+19 1.974027e+19
##  [96] 3.194043e+19 5.168071e+19 8.362114e+19 1.353019e+20 2.189230e+20
```

2. Return the first 15 Fibonacci Numbers

```r
# Retrieving the first 15 through element selection
fibo[1:15]
```

```
##  [1]   0   1   1   2   3   5   8  13  21  34  55  89 144 233 377
```

3. Write a code that finds the nth Fibonacci Number. What is the 30th Fibonacci Number?

```r
# Calculates the Fibonacci number
#
# x: a numeric
#
# returns: the fibonacci number of x
fiboF <- function(x) {
  fibo_v = c(0, 1)

  if (x > 2) {
      for (i in 3:x) {
        fibo_v[i] = fibo_v[i - 1] + fibo_v[i - 2]
      }
    }
    fibo_v[x]
}

fiboF(30)
```

```
## [1] 514229
```

4. Sanity Check: The log_2 of the 100th Fibonacci Number is about 67.57.

```r
# Checking the log2 of the 100th fibonacci number
log2(fiboF(100))
```

```
## [1] 67.56899
```

Problem 3

Load the wmata_ridership data frame into R

1. Save the data in your local machine in your working directory (use write_csv()).
2. Upload it into R (use read_csv() and relative path) and name it wmata.

```r
wmata <- read_csv("wmata_ridership.csv")
```

```
## Rows: 5469 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): Total
## date (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

3. What are the variables?

The variables are date and the amount of public transport ridership in Washington D.C..

4. Separate variable Date to year, month, and day.

```r
# Changing Date to a datetime var
# Creating new variables using lubridate functionality
wmata <- wmata %>%
  mutate(Date = ymd(Date)) %>%
  transmute(year = year(Date),
            month = month(Date),
            day = mday(Date),
            dayofweek = wday(Date),
            total = Total)
# Checking
head(wmata)
```

```
## # A tibble: 6 x 5
##    year month   day dayofweek  total
##   <int> <int> <int>     <int>  <dbl>
## 1  2004     1     1         5 129000
## 2  2004     1     2         6 419000
## 3  2004     1     3         7 222000
## 4  2004     1     4         1 140000
## 5  2004     1     5         2 564000
## 6  2004     1     6         3 609000
```

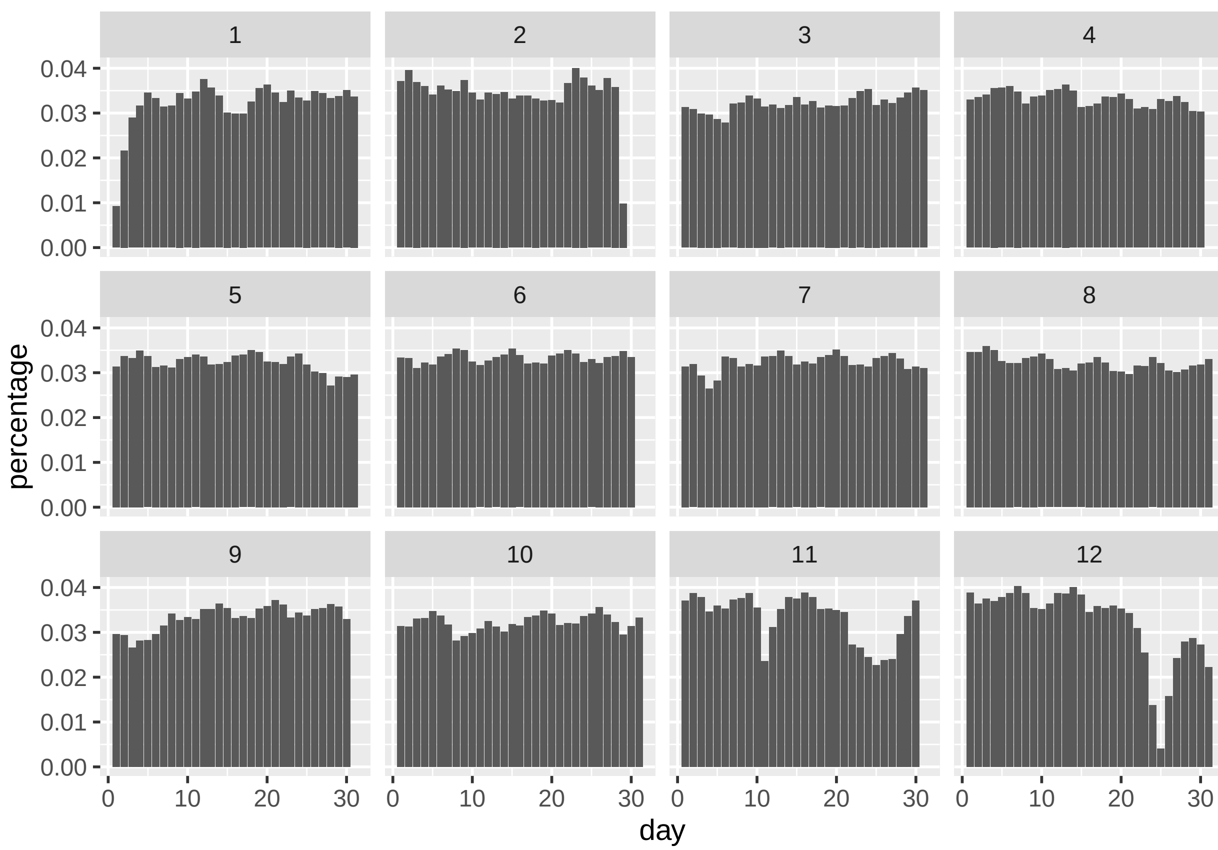
5. For each month, calculate the proportion of rides made on a given day of the month.

```r
# Creating new variable showing the proportion of rides
# on a given day of the month
wmata <- wmata %>%
  group_by(month) %>%
  mutate(percentage = total / sum(total))

head(wmata)
```

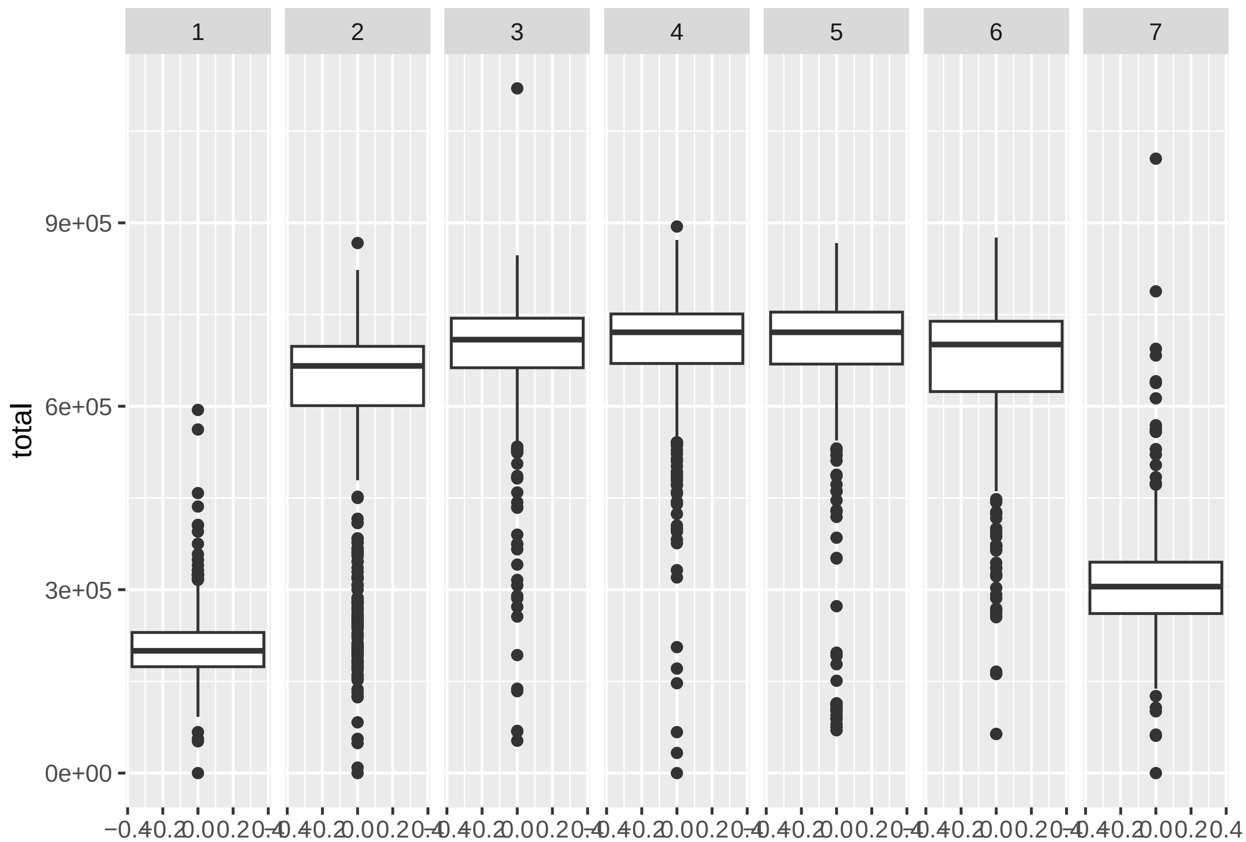
```
## # A tibble: 6 x 6
## # Groups:   month [1]
##    year month   day dayofweek  total percentage
##   <int> <int> <int>     <int>  <dbl>      <dbl>
## 1  2004     1     1         5 129000   0.000554
## 2  2004     1     2         6 419000   0.00180
## 3  2004     1     3         7 222000   0.000954
## 4  2004     1     4         1 140000   0.000601
## 5  2004     1     5         2 564000   0.00242
## 6  2004     1     6         3 609000   0.00262
```

```r
# Visualization
wmata %>%
  ggplot(aes(x = day, y = percentage)) +
  geom_col() +
  facet_wrap(~month)
```



6. Make box plots of the proportions of ridership vs. day of the week. Exclude days from 2004.

```
wmata %>%
  filter(year != 2004) %>%
  ggplot(aes(y = total)) +
  geom_boxplot() +
  facet_grid(~dayofweek)
```



Problem 4

1. Create a new repository in Github. Name it repositoryexam_1
2. Drag and Drop 3 files from your desktop to your new repository (any files that you think is appropriate)
3. Take a screenshot of the created repository showing evidence of the three files uploaded.

!("data614_exam14.png")

4. Now go to the bash terminal and clone the repository back to your Desktop

Problem 5

1. Type your PAT token

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

2. Push the exam_1 file (without solution) to your repositoryexam_1
3. Take screenshot and post the url of your Github page that shows the file being pushed along with the commit message "Add exam 1 problems set"

Problem 6 (data.table package)

```r
# Loading data frames
flights <- data.table(nycflights13::flights)
airlines <- data.table(nycflights13::airlines)
```

1. Add the full airline names to the flights data.table.

```r
# Adding full airline names to flights data.table using a join
flights <- airlines[flights, on = .(carrier)]
```

2. Use data.table to calculate the median air time for each month.

```r
# Calculating median air time by month
# Filtered out NAs with !is.na(air_time)
med_airtime <- flights[!is.na(air_time),
                       .(med_airtime = median(air_time)), by = month]

# Printing Median Air times by Month
med_airtime
```

```
##     month med_airtime
##  1:     1         137
##  2:    10         124
##  3:    11         133
##  4:    12         142
##  5:     2         136
##  6:     3         132
##  7:     4         133
##  8:     5         122
##  9:     6         127
## 10:     7         124
## 11:     8         124
## 12:     9         117
```

3. Use data.table to calculate the number of trips from each airport for the carrier code DL.

```r
# Calculating number of trips from each airport for Delta Airlines
DL_flights <- flights[carrier == "DL",
                      .(flight_no = length(carrier)),
                      by = origin]

# Printing result
DL_flights
```

```
##    origin flight_no
## 1:    LGA     23067
## 2:    JFK     20701
## 3:    EWR      4342
```

4. Calculate the mean departure delay for each origin in the months of January and February.

```r
# Calculating mean departure delay for each origin in January/February
mean_delay <- flights[month %in% c(1, 2),
                      .(mean_delay = mean(dep_delay, na.rm = T)),
                      by = origin]

# Printing result
mean_delay
```

```
##    origin mean_delay
## 1:    EWR   14.03920
## 2:    LGA    6.26982
## 3:    JFK   10.10761
```

Problem 7

The 2010 General Social Survey asked 1,500 US residents: "Do you think the use of marijuana should be made legal, or not?" 35% of the respondents said it should be made legal.

a. Is 35% a sample statistic or a population parameter? Explain.

35% is a sample statistic. A statistic is a summary of the data computed from the sample. In this case, 35% of the 1,500 sample of respondents said marijuana should be made illegal (525/1500 people). A population parameter is a number describing the entire population and is often unknown. We can estimate the population parameters with their corresponding sample statistics though which can give us an idea of the true value amongst the population.

b. Construct a 95% confidence interval for the proportion of US residents who think marijuana should be made legal, and interpret it in the context of the data.

First, I will construct the 95% confidence interval by hand using the formula:

95% Confidence Interval $= .35 \pm 1.96\sqrt{\frac{.35(1-.35)}{1500}}$

95% Confidence Interval $= .35 \pm 0.024 = (0.326, 0.374)$

Now, I will confirm the results using software:

```r
prop.test(525, 1500, correct = F)
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  525 out of 1500, null probability 0.5
## X-squared = 135, df = 1, p-value < 2.2e-16
```

```
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.3262734 0.3744929
## sample estimates:
##    p
## 0.35
```

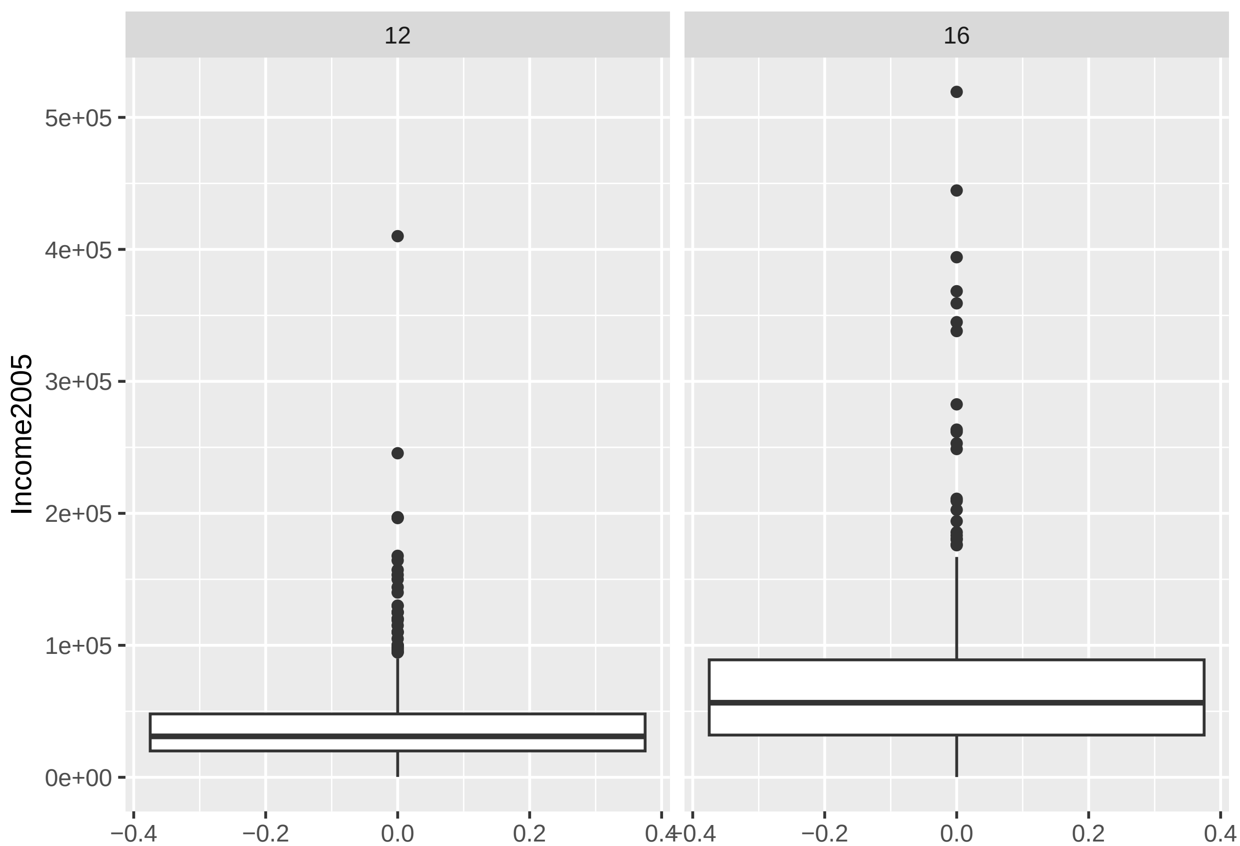
With 95% confidence, the true proportion of US residents who think marijuana should be made legal (in 2010) is between 32.6% and 37.4%.

Problem 8

Read up on the ex0330 dataset from the Sleuth3 R package. Determine if education level is associated with income. Interpret any estimates and confidence intervals you derive.

```r
# Loading dataset
edu_inc <- Sleuth3::ex0330
```

```r
# Visualizing the association between Education Level and Income
edu_inc %>%
  ggplot(aes(y = Income2005)) +
  geom_boxplot() +
  facet_wrap(~Educ)
```



From the box plot visualization, there appears to be an association between a higher education level and a higher income.

```r
# Analyzing association using a regression model
eduinc_lm <- lm(Income2005~Educ, data = edu_inc)

# Model summary
summary(eduinc_lm)
```

```
##
## Call:
## lm(formula = Income2005 ~ Educ, data = edu_inc)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -69797 -21865  -6931  12985 449343
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -62531.3     8235.1  -7.593 5.62e-14 ***
## Educ          8283.0      620.9  13.339  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42330 on 1424 degrees of freedom
## Multiple R-squared:  0.1111, Adjusted R-squared:  0.1105
## F-statistic: 177.9 on 1 and 1424 DF,  p-value: < 2.2e-16
```

Another way to check for an association is through the creation of a regression model. One notable finding from the model summary is that education has a large positive slope of 8283. In the context of the data, this means when education level is "16", there is a 33,132 increase in income.

```r
hs <- edu_inc %>%
  filter(Educ == 12)
college <- edu_inc %>%
  filter(Educ == 16)

t.test(college$Income2005, hs$Income2005)
```

```
##
##  Welch Two Sample t-test
##
## data:  college$Income2005 and hs$Income2005
## t = 9.9827, df = 473.85, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  26610.39 39653.77
## sample estimates:
## mean of x mean of y
##  69996.97  36864.90
```

We can also look at a t.test to see if the difference of means in income are significantly different between the two education groups. From the output, the extremely small p-value indicates a significant difference between the mean income of the groups. The 95% confidence interval is telling us with 95% confidence, the increase in income for the "16" education group from the "12 education group" will be from 26,610.39 to 39,653.77.

There does seem to be a significant association between education level and income in this data set. The box plot visualizations, linear regression model, and t.test all support this finding.