DSCI 552 Lab 2

Introduction to Confidence Intervals and Hypothesis Testing

Contents

```
BEGIN ASSIGNMENT
requirements: requirements.R
files:
  - data
generate:
    show_stdout: true
    show_hidden: true
environment: environment.yml
export_cell: false
```

Lab Mechanics

rubric={mechanics:5}

- Paste the URL to your GitHub repo here: **INSERT YOUR GITHUB REPO URL HERE**
- Once you finish the assignment, you must **knit** this `R` markdown to create a `.pdf` file and push everything to your GitHub repo using `git push`. You are responsible for ensuring all the figures, texts, and equations in the `.pdf` file are appropriately rendered.
- You must submit this `.Rmd` **and** the rendered `.pdf` files to Gradescope.

  Heads-up: You need to have a minimum of 3 commits.

Code Quality

rubric={quality:3}

The code that you write for this assignment will be given one overall grade for code quality. Check our **code quality rubric** as a guide to what we are looking for. Also, for this course (and other MDS courses that use `R`), we are trying to follow the `tidyverse` code style. There is a guide you can refer too: http://style.tidyverse.org/

Each code question will also be assessed for code accuracy (i.e., does it do what it is supposed to do?).

Writing

rubric={writing:3}

To get the marks for this writing component, you should:

- Use proper English, spelling, and grammar throughout your submission (the non-coding parts).
- Be succinct. **This means being specific about what you want to communicate, without being superfluous.**

Check our **writing rubric** as a guide to what we are looking for.

Setup

If you fail to load any packages, you can install them and try loading the library again.

```
library(infer)
library(palmerpenguins)
library(testthat)
library(digest)
library(tidyverse)
library(knitr)
```

```
library(cowplot)
library(datateachr)
```

Exercise 1: Median Salary in Vancouver

The `data/salary.csv` data set includes remuneration and expenses from $n = 3003$ employees earning over CAD $75,000/year. This data was retrieved from the **Vancouver Open Data Portal**.

Q1.1.

rubric={reasoning:5}

Imagine you are a data scientist working for the Vancouver City Council. **They want to make an inference on the median salary of all employees in Vancouver who earn over $75,000/year.** Population mean is off the table, given its sensitivity to outliers that are likely to appear in such an unequal city.

The city has collected a small sample (stored in `salary.csv`) of size $n = 3003$. Explain to the Vancouver City Council how you would estimate a bootstrapped 90% confidence interval (CI) of the median salary in Vancouver from this sample.

Write down the steps in written English such as:

Step 1:

Step 2:

⋮

Step k:

ANSWER:

The steps can be described as follows:

Step 1: *We will assume that the base sample of size $n = 3003$ is representative enough from the population of interest. Hence, we generate m bootstrap replicates of the base sample (i.e., sampling with replacement).*

Step 2: *For each one of these m bootstrap replicates, we calculate the bootstrap sample statistic (i.e, the bootstrap sample median) that aims to estimate the population parameter of interest (i.e., the population median in this case).*

Step 3: *From the distribution of bootstrap sample statistics that you generated, we find the values (bootstrap sample median estimates) at the 5th and 95th percentiles.*

Step 4: *We report those values as our 90% CI.*

Q1.2.

rubric={autograde:7}

Using the sample `salary`, compute the 90% CI of the median salary of all employees in Vancouver who earn over $75,000/year.

```
salary <- read_csv("data/salary.csv")
```

You can use the `get_confidence_interval()` function from the `infer` package. Note the following:

- Use seed `552` when appropriate for reproducibility.
- Use 3,000 bootstrap samples.

Store the lower bound in the vector `answer1_2_lower_ci` and the upper bound in `answer1_2_upper_ci`.

```
BEGIN QUESTION
name: Q1.2
points:
```

```
 - 3.5
 - 3.5
```

```
answer1_2_lower_ci <- NULL # this should a single value
answer1_2_upper_ci <- NULL # this should a single value

# BEGIN SOLUTION
set.seed(552)

bootstrap_dist <- salary %>%
  select(Remuneration) %>%
  rep_sample_n(reps = 3000, size = nrow(salary), replace = TRUE) %>%
  group_by(replicate) %>%
  summarize(stat = median(Remuneration))

ci <- bootstrap_dist %>%
  get_confidence_interval(level = 0.90, type = "percentile")

answer1_2_lower_ci <- ci$lower_ci
answer1_2_upper_ci <- ci$upper_ci
# END SOLUTION

answer1_2_lower_ci
```

```
## [1] 97836
```

```
answer1_2_upper_ci
```

```
## [1] 99510
```

Q1.3.

rubric={viz:4}

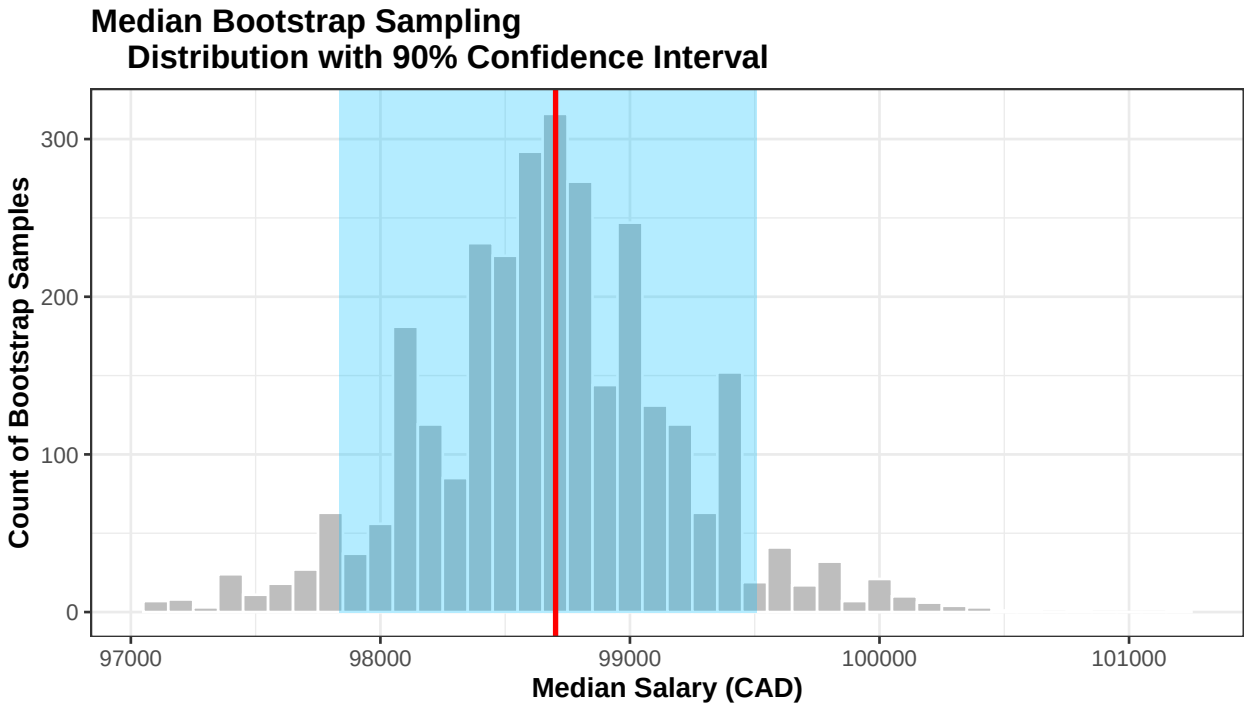
Plot your 3000 bootstrap sample medians as a histogram. Include the 90% bootstrap CI bounds along with the sample median. You could use `ggplot2`'s function `geom_vline()`.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to an object called `boot_CI_median_plot`.

Heads-up: This plot does not have auto grading tests.

```
boot_CI_median_plot <- NULL

# BEGIN SOLUTION
boot_CI_median_plot <- bootstrap_dist %>%
  ggplot(aes(x = stat)) +
  geom_histogram(binwidth = 100, colour = "white", fill = "grey") +
  annotate("rect",
    xmin = ci[[1]], xmax = ci[[2]], ymin = 0, ymax = Inf,
    fill = "deepskyblue",
    alpha = 0.3
  ) +
  geom_vline(
    xintercept = median(salary$Remuneration),
    size = 1,
    colour = "red"
  ) +
```

```
  labs(
    title = "Median Bootstrap Sampling
    Distribution with 90% Confidence Interval",
    x = "Median Salary (CAD)",
    y = "Count of Bootstrap Samples"
  ) +
  theme_bw() +
  theme(
    text = element_text(size = 11),
    plot.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold"),
    legend.title = element_text(face = "bold"),
  )
# END SOLUTION

boot_CI_median_plot
```



Exercise 2: The Canadian 44th Election for Prime Minister

Canada's 44th election for Prime Minister took place last year. Throughout the electoral campaign, we saw many polls carried out to estimate the proportion of votes each party would get **based on a randomly selected sample of eligible voters**. This exercise will use **the September 18th, 2021, poll reported by the Angus Reid Institute (ARI)** that asked $n = 2,042$ people which party they intended to vote for.

> **Note:** The ARI's report states that "*discrepancies in or between totals are due to rounding.*" Hence, the corresponding estimated percentages per party will not add up to 100%.

Let us create a synthetic data set `polls` that resembles the results of the poll by running the code cell below:

```
# Poll's sample size
n <- 2042

# Party vector
party <- c("CPC", "Liberals", "NDP", "Bloc", "PPC", "Green", "Others")

# Setting up sampled voting proportions
# We make an adjustment on Others to 0.0302, instead of 0.01 as in the
# ARI's report, so the other estimated proportions in our computational
# analysis will match.
prop <- c(0.32, 0.3, 0.2, 0.07, 0.05, 0.03, 0.0302)

# Voting counts
votes <- round(n * prop, 0)

# Creating data
polls <- data.frame(party = rep(party, votes))
head(polls)
```

```
##   party
## 1   CPC
## 2   CPC
## 3   CPC
## 4   CPC
## 5   CPC
## 6   CPC
```

```
tail(polls)
```

```
##       party
## 2037 Others
## 2038 Others
## 2039 Others
## 2040 Others
## 2041 Others
## 2042 Others
```

We will be extremely inquisitive with these polling results. Hence, let us check the ARI's methodology:

> *The Angus Reid Institute conducted an online survey from Sept. 15 – 18, 2021 among a representative randomized sample of 2,042 Canadian adults who are members of **Angus** Reid Forum. For comparison purposes only, a probability sample of this size would carry a margin of error of +/- 2.2 percentage points, 19 times out of 20. Discrepancies in or between totals are due to rounding. The survey was self-commissioned and paid for by ARI.*

The summary of this methodology implicates a classical approach to compute the uncertainty of the polling

estimates (i.e., their sampling distribution) using a classical approach via what they call the *margin of error* (**check this helpful resource**). The expression "*19 times out of 20*" implies a 95% confidence (i.e., $\frac{19}{20} \times 100\% = 95\%$).

This classical approach implies **Normal assumptions when** $n \to \infty$ under the frequentist paradigm. Nevertheless, with an online survey of size $n = 2,042$ from a population of millions of voters, **are we entirely sure we can use these classical assumptions?**

Let us try the computational approach!

Q2.1.

rubric={autograde:10}

Your task as an inquisitive Data Scientist in inferential matters is to use the sampled data in `polls` to estimate the voter population proportion parameters per party, p_{party_k}, which is the proportion of Canadian population who intend to vote for each of the k parties in the data set `polls` (**at the time the poll was ran!**). This will yield the following estimate:

$$\hat{p}_{\text{party}_k} = \frac{n_k}{n},$$

where n_k is the number of respondents who expressed their electoral sympathy for the kth party in the sample `polls` and n is the overall sample size.

Furthermore, compute the **95% bootstrap CIs** of the **estimated** voting proportion $\hat{p}_{\text{party}_k}$ of each party, using **15,000 bootstrapped samples**. This will give you the corresponding uncertainty measures **without using margin of errors via Normal approximations**. Use seed `552` when appropriate for reproducibility.

The output data frame `answer2_1` should look like below.

Table 1: Empty `answer2_1`.

party	lower_ci	upper_ci	p_hat
Liberals			
CPC			
NDP			
Bloc			
PPC			
Green			
Others			

Heads-up: Round up all your **final** table results to 2 decimal places.

```
BEGIN QUESTION
name: Q2.1
points:
 - 0
 - 0
 - 2
 - 4
 - 4
```

```
answer2_1 <- NULL

# BEGIN SOLUTION
polls <- polls %>%
```

```
  mutate(party = as.factor(party))

p_hats <- polls %>%
  group_by(party) %>%
  summarise(p_hat = round(n() / nrow(polls), 2))

set.seed(552)

answer2_1 <- polls %>%
  rep_sample_n(size = nrow(polls), reps = 15000, replace = TRUE) %>%
  group_by(replicate, party) %>%
  summarize(stat = n() / nrow(polls)) %>%
  ungroup() %>%
  select(-replicate) %>%
  group_by(party) %>%
  nest() %>%
  summarize(ci = map(data, ~ get_confidence_interval(.,
    level = 0.95,
    type = "percentile"
  ))) %>%
  unnest(ci) %>%
  left_join(p_hats) %>%
  mutate(
    party = fct_reorder(party, p_hat, .desc = TRUE),
    lower_ci = round(lower_ci, 2),
    upper_ci = round(upper_ci, 2)
  ) %>%
  arrange(desc(p_hat))
# END SOLUTION

answer2_1
```

```
## # A tibble: 7 x 4
##   party    lower_ci upper_ci p_hat
##   <fct>       <dbl>    <dbl> <dbl>
## 1 CPC          0.3      0.34  0.32
## 2 Liberals     0.28     0.32  0.3
## 3 NDP          0.18     0.22  0.2
## 4 Bloc         0.06     0.08  0.07
## 5 PPC          0.04     0.06  0.05
## 6 Green        0.02     0.04  0.03
## 7 Others       0.02     0.04  0.03
```

Q2.2.

rubric={viz:6}

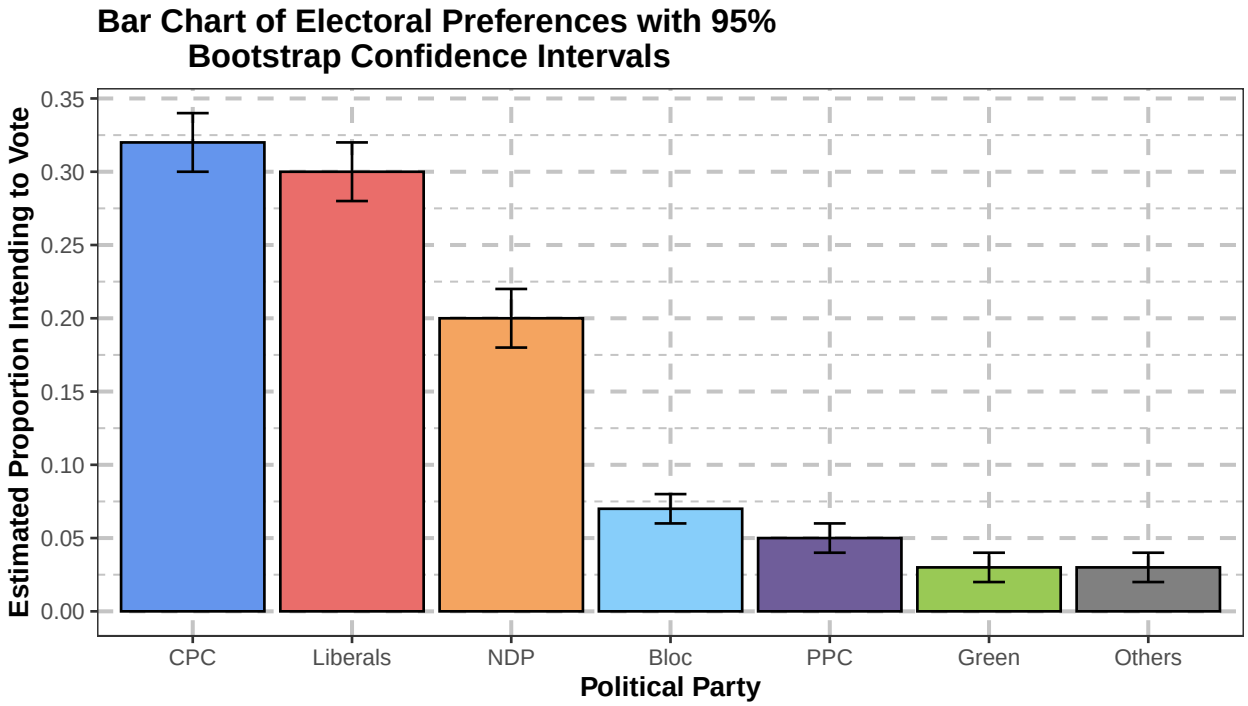
Visualize the estimate, $\hat{p}_{party_k}$, and the 95% bootstrap CIs on a plot, using a bar plot with **error bars**. You should map the k parties to the x-axis and the proportion estimates and CIs as **error bars** to the y-axis. The order of the bars has to be decreasing by $\hat{p}_{\text{party}_k}$ from left to right.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to the variable `party_bar_plot`.

> **Hint:** Use `geom_bar(stat = "identity")` from `ggplot2`.

Heads-up: There are no auto-grading test functions for this question. Furthermore, the error bars represent the bounds of your 95% bootstrap CIs per party.

```
party_bar_plot <- NULL

# BEGIN SOLUTION
party_colors <- c("#6495ED", "#EA6D6A", "#F4A460", "#87CEFA", "#6F5D9A", "#99C955", "#808080")

party_bar_plot <- ggplot(answer2_1, aes(x = party, y = p_hat, fill = party)) +
  geom_bar(
    stat = "identity",
    colour = "black"
  ) +
  geom_errorbar(aes(ymin = lower_ci, ymax = upper_ci),
    size = 0.5, color = "black", width = .2
  ) +
  scale_fill_manual(values = party_colors) +
  scale_y_continuous(breaks = seq(0, 0.35, by = 0.05)) +
  theme_bw() +
  xlab("Political Party") +
  ylab("Estimated Proportion Intending to Vote") +
  ggtitle("Bar Chart of Electoral Preferences with 95%
          Bootstrap Confidence Intervals") +
  theme(
    text = element_text(size = 11),
    plot.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold"),
    panel.grid = element_line(
      color = "gray77",
      size = 0.75, linetype = 2
    ),
    legend.position = "none"
  )
# END SOLUTION

party_bar_plot
```



Q2.3.

rubric={reasoning:5}

In one or two paragraphs, use written English to interpret and report the estimates (and their bootstrap 95% CIs) for each of the k parties.

> **Heads-up:** Note that a survey of electoral preferences is just a **snapshot** of the moment data was collected. Having said that, we must clarify the following: **an electoral survey does not predict a winning candidate/party**.

As an important side note, beginning this statistical course in MDS (along with others like DSCI 562, 553, and 554), we need to communicate our data analysis **in the context of the problem**. Therefore, this matter implicates to avoid writing **generic conclusions** such as:

> *We are 95% confident that* $p_{Liberals}$ *is between...*
>
> *We are 95% confident that* p_{CPC} *is between...*
>
> *We are 95% confident that* p_{NDP} *is between...*
>
> *We are 95% confident that* p_{Bloc} *is between...*
>
> *We are 95% confident that* p_{PPC} *is between...*
>
> *We are 95% confident that* p_{Green} *is between...*
>
> *We are 95% confident that* p_{Others} *is between...*

The above sentences do not make any sense to the readers if they are disconnected, which makes a bad data analysis.

Even though we might question the figures reported by the ARI **here**, the authors provide a well-written summary of their statistical estimations. You can use this summary as a starting example **without going too much further into political events they also depicted**.

ANSWER:

Let us start with the top 2 contestants: Conservative Party of Canada (CPC) versus Liberals. First off, the corresponding percentage estimates of electoral preferences are 32% and 30% for CPC and Liberals respectively. From the perspective of a point estimate, we might be tempted to state that the population of voters favoured CPC at the moment the survey was performed. Nevertheless, the bootstrap 95% CIs are overlapped, which indicates that the precision associated with both proportion estimates is not enough to make such as strong statement (i.e., it might be the case that both population proportions are equal!).

When it comes to the other parties, the NDP has a clear third place with an estimated percentage of 20%. Our bar chart shows that its corresponding bootstrap 95% CI does not overlap with any other party. On the other hand, Bloc Québécois is definitely ahead of the Green Party and Others, given its situation with the corresponding CIs. Nonetheless, we have ***almost overlapped*** *CIs between the Bloc Québécois (estimated 7%) and the People's Party of Canada (PPC) whose estimated percentage is 5%.*

(Challenging) Q2.4.

rubric={viz:3,reasoning:3}

Let us go even deeper into the ARI's results and compare them with our computational outputs and the corresponding 95% CIs (both classical from the ARI and your bootstrap CIs).

The **resource previously shared** on the margin of error in surveys states the following:

> *For results expressed as percentages, it is often possible to calculate a maximum margin of error that applies to all results from the survey (or at least all results based on the full sample). The* ***maximum*** *margin of error can sometimes be calculated directly from the sample size (the number of poll respondents).*

Again, let us check ARI's methodology:

> *The Angus Reid Institute conducted an online survey from Sept. 15 – 18, 2021 among a representative randomized sample of 2,042 Canadian adults who are members of Angus Reid Forum. For comparison purposes only, a probability sample of this size would carry a margin of error of +/- 2.2 percentage points, 19 times out of 20. Discrepancies in or between totals are due to rounding. The survey was self-commissioned and paid for by ARI.*

Moreover, ARI's authors do not provide any specific margins of error per party. Hence, **we will use 0.022 as the maximum margin of error for the ARI's estimates**. This will yield the following 95% classical CI computations:

```
# Note Others has a proportion of 0.01 since this is the estimate
# reported in the summary.
ARI_results <- data.frame(
  party = c(
    "CPC", "Liberals", "NDP", "Bloc",
    "PPC", "Green", "Others"
  ),
  p_hat = c(0.32, 0.3, 0.2, 0.07, 0.05, 0.03, 0.01))
ARI_results <- ARI_results %>%
  mutate(lower_ci = p_hat - 0.022, upper_ci = p_hat + 0.022)
```

The previous chunk of code provides the following ARI's table (we will not use *Others* in this analysis):

Table 2: Angus Reid Institute point and classical interval estimates at 95% confidence.

Party	Proportion Estimate	95% Bootstrap Lower Bound	95% Bootstrap Upper Bound
CPC	0.32	0.298	0.342
Liberals	0.30	0.278	0.322
NDP	0.20	0.178	0.222
Bloc	0.07	0.048	0.092
PPC	0.05	0.028	0.072
Green	0.03	0.008	0.052

Now, this is your computational table from **Q2.1** (we will not use *Others* in this analysis):

Table 3: Your point and bootstap interval estimates at 95% confidence.

Party	Proportion Estimate	95% Bootstrap Lower Bound	95% Bootstrap Upper Bound
CPC	0.32	0.30	0.34
Liberals	0.30	0.28	0.32
NDP	0.20	0.18	0.22
Bloc	0.07	0.06	0.08
PPC	0.05	0.04	0.06
Green	0.03	0.02	0.04

Visualizing both sets of results via **density plots** (either theoretical for ARI or estimated via your **15,000 bootstrapped samples** from **Q2.1.**) would be easier. Therefore, code the following:

- Six **Normal density plots** (one per party) whose mean will be corresponding $\hat{p}_{\text{party}_k}$ (i.e., column **Proportion Estimate** as a vertical **DOTTED** line) along with vertical **SOLID** lines depicting the bounds for the ARI's classical 95% CIs. Name these plots as follows: `ARI_CPC_plot`, `ARI_Liberals_plot`, `ARI_NDP_plot`, `ARI_Bloc_plot`, `ARI_PPC_plot`, and `ARI_Green_plot`. **Function `stat_function()` would be useful.**

  Heads-up: You will also need a **standard error**, which will be the `sd` argument for each Normal plot per party. We can easily obtain this **maximum standard error** manipulating the below expression:

$$\text{Maximum Margin of Error} = \underbrace{1.96}_{\text{0.975-quantile from } \mathcal{N}(0,1)} \times \text{Maximum Standard Error.}$$

- Six **empirical density plots** (one per party) whose mean will be corresponding $\hat{p}_{\text{party}_k}$ (i.e., column **Proportion Estimate** as a vertical **DOTTED** line) along with vertical **SOLID** lines depicting the bounds of the 95% bootstrap CIs you obtained in **Q2.1**. Name these plots as follows: `Bootstrap_CPC_plot`, `Bootstrap_Liberals_plot`, `Bootstrap_NDP_plot`, `Bootstrap_Bloc_plot`, `Bootstrap_PPC_plot`, and `Bootstrap_Green_plot`.

  Heads-up: To make these **empirical** density plots, you will need to plot your 15,000 bootstrap estimates per party as six histograms using `aes(y = ..density..)` via `geom_histogram()`. Then, you will have to use `geom_density()`.

Note both axes have to be properly labelled along with an adequate plot title. **You will also need to adjust the axes limits per party to make further comparisons easier.** Creating plotting functions might be a handy idea. Use `552` as a simulation seed.

Heads-up: There are no auto-grading test functions for this question.

```
ARI_CPC_plot <- NULL
ARI_Liberals_plot <- NULL
ARI_NDP_plot <- NULL
ARI_Bloc_plot <- NULL
ARI_PPC_plot <- NULL
ARI_Green_plot <- NULL
Bootstrap_CPC_plot <- NULL
Bootstrap_Liberals_plot <- NULL
Bootstrap_NDP_plot <- NULL
Bootstrap_Bloc_plot <- NULL
Bootstrap_PPC_plot <- NULL
Bootstrap_Green_plot <- NULL

# BEGIN SOLUTION
# Computing maximum standard error
max_std_error <- 0.022 / 1.96

#' Create ARI's Normal density plot for a single party
#'
#' @param p_hat numeric vector containing estimated proportion
#' for the party
#' @param std_error numeric vector containing the standard deviation
#' to be used in the Normal distribution
#' @param party_colour string defining the colour to be used for the plot lines
#' @param plot_title string containing the plot's title
#' @param party_CI vector containing the lower and upper CI bounds
#' to be used in the plot
#' @return ggplot2 object that is the Normal density plot
#' @export
normal_density <- function(p_hat, std_error,
                           party_colour, plot_title, party_CI)
{
  ggplot() +
  geom_function(
    fun = dnorm, colour = party_colour,
    args = list(mean = p_hat, sd = std_error)
  ) +
  labs(x = "Proportion", y = "Density") +
  ggtitle(plot_title) +
  theme(
    text = element_text(size = 7),
    plot.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold")
  ) +
  geom_vline(
    xintercept = p_hat,
    colour = party_colour,
    linetype = "dotted"
  ) +
  geom_vline(
    xintercept = party_CI$lower_ci,
    colour = party_colour
```

```
  ) +
  geom_vline(
    xintercept = party_CI$upper_ci,
    colour = party_colour
  ) +
  ylim(0, 50)
}

ARI_CPC_plot <- normal_density(
  p_hat = ARI_results$p_hat[1],
  std_error = max_std_error,
  party_colour = "darkblue",
  plot_title = "ARI's 95% Classical CIs for CPC",
  party_CI = filter(ARI_results, party == "CPC")[, c(3, 4)]
) +
  xlim(0.25, 0.4)

ARI_Liberals_plot <- normal_density(
  p_hat = ARI_results$p_hat[2],
  std_error = max_std_error,
  party_colour = "darkred",
  plot_title = "ARI's 95% Classical CIs for Liberals",
  party_CI = filter(ARI_results, party == "Liberals")[, c(3, 4)]
) +
  xlim(0.25, 0.35)

ARI_NDP_plot <- normal_density(
  p_hat = ARI_results$p_hat[3],
  std_error = max_std_error,
  party_colour = "orange",
  plot_title = "ARI's 95% Classical CIs for NDP",
  party_CI = filter(ARI_results, party == "NDP")[, c(3, 4)]
) +
  xlim(0.15, 0.25)

ARI_Bloc_plot <- normal_density(
  p_hat = ARI_results$p_hat[4],
  std_error = max_std_error,
  party_colour = "lightblue",
  plot_title = "ARI's 95% Classical CIs for Bloc Québécois",
  party_CI = filter(ARI_results, party == "Bloc")[, c(3, 4)]
) +
  xlim(0, 0.15) +
  ylim(0, 80)

ARI_PPC_plot <- normal_density(
  p_hat = ARI_results$p_hat[5],
  std_error = max_std_error,
  party_colour = "purple",
  plot_title = "ARI's 95% Classical CIs for PPC",
  party_CI = filter(ARI_results, party == "PPC")[, c(3, 4)]
) +
  xlim(0, 0.1) +
```

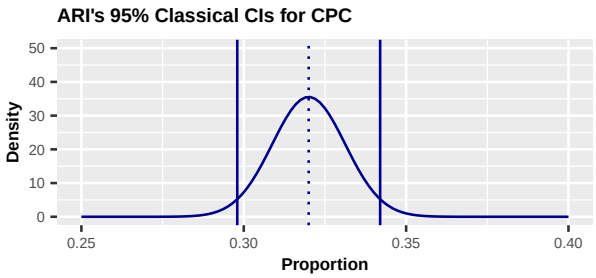
```
  ylim(0, 90)

ARI_Green_plot <- normal_density(
  p_hat = ARI_results$p_hat[6],
  std_error = max_std_error,
  party_colour = "darkgreen",
  plot_title = "ARI's 95% Classical CIs for Green",
  party_CI = filter(ARI_results, party == "Green")[, c(3, 4)]
) +
  xlim(0, 0.075) +
  ylim(0, 110)

#' Create bootstrap empirical density plot for a single party
#'
#' @param p_hat numeric vector containing estimated proportion
#' for the party
#' @param data data frame from which we will bootstrap from
#' #' @param party_name string defining the party name as in data
#' @param party_colour string defining the colour to be used for the plot lines
#' @param plot_title string containing the plot's title
#' @return ggplot2 object that is an empirical density plot
#' @export
boot_empirical_density <- function(p_hat, data, party_name,
                                   party_colour, plot_title) {
  set.seed(552)

  boot_data <- data %>%
    rep_sample_n(size = nrow(data), reps = 15000, replace = TRUE) %>%
    group_by(replicate, party) %>%
    summarize(stat = n() / nrow(data))

  filtered_boot_data <- boot_data %>%
    filter(party == party_name)

  ci_95 <- filtered_boot_data %>%
    get_confidence_interval(level = 0.95, type = "percentile")

  filtered_boot_data %>%
    ggplot(aes(x = stat)) +
    geom_histogram(aes(y = ..density..),
      colour = "white",
      fill = party_colour,
      alpha = 0.3, bins = 100
    ) +
    theme(
      text = element_text(size = 7),
      plot.title = element_text(face = "bold"),
      axis.title = element_text(face = "bold")
    ) +
    geom_density(colour = party_colour) +
    ggtitle(plot_title) +
    geom_vline(
      xintercept = p_hat,
```

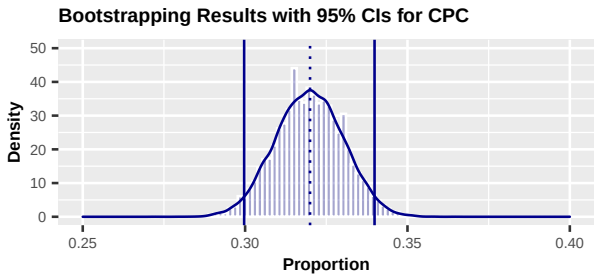
```
      colour = party_colour,
      linetype = "dotted"
    ) +
    geom_vline(
      xintercept = ci_95$lower_ci,
      colour = party_colour
    ) +
    geom_vline(
      xintercept = ci_95$upper_ci,
      colour = party_colour
    ) +
    ylim(0, 50) +
    labs(x = "Proportion", y = "Density")
}

Bootstrap_CPC_plot <- boot_empirical_density(
  p_hat = ARI_results$p_hat[1],
  data = polls,
  party_name = "CPC",
  party_colour = "darkblue",
  plot_title = "Bootstrapping Results with 95% CIs for CPC"
) +
  xlim(0.25, 0.4)

Bootstrap_Liberals_plot <- boot_empirical_density(
  p_hat = ARI_results$p_hat[2],
  data = polls,
  party_name = "Liberals",
  party_colour = "darkred",
  plot_title = "Bootstrapping Results with 95% CIs for Liberals"
) +
  xlim(0.25, 0.35)

Bootstrap_NDP_plot <- boot_empirical_density(
  p_hat = ARI_results$p_hat[3],
  data = polls,
  party_name = "NDP",
  party_colour = "orange",
  plot_title = "Bootstrapping Results with 95% CIs for NDP"
) +
  xlim(0.15, 0.25)

Bootstrap_Bloc_plot <- boot_empirical_density(
  p_hat = ARI_results$p_hat[4],
  data = polls,
  party_name = "Bloc",
  party_colour = "lightblue",
  plot_title = "Bootstrapping Results with 95% CIs for Bloc Québécois"
) +
  xlim(0, 0.15) +
  ylim(0, 80)

Bootstrap_PPC_plot <- boot_empirical_density(
```

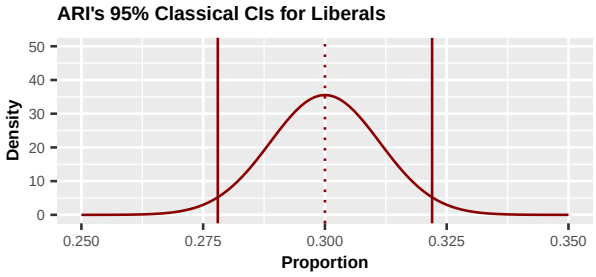
```
  p_hat = ARI_results$p_hat[5],
  data = polls,
  party_name = "PPC",
  party_colour = "purple",
  plot_title = "Bootstrapping Results with 95% CIs for PPC"
) +
  xlim(0, 0.1) +
  ylim(0, 90)

Bootstrap_Green_plot <- boot_empirical_density(
  p_hat = ARI_results$p_hat[6],
  data = polls,
  party_name = "Green",
  party_colour = "darkgreen",
  plot_title = "Bootstrapping Results with 95% CIs for Green"
) +
  xlim(0, 0.075) +
  ylim(0, 110)
# END SOLUTION
```

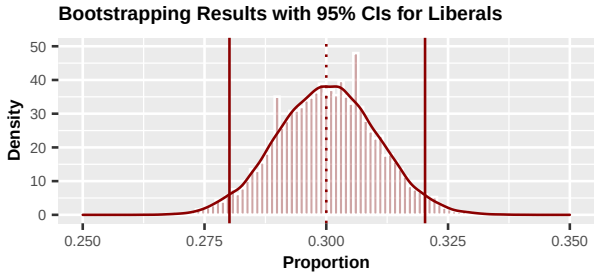
Run the following chunk of code before proceeding:

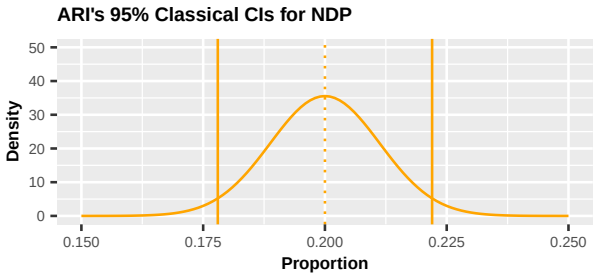
```
plot_grid(ARI_CPC_plot, Bootstrap_CPC_plot,
  ARI_Liberals_plot, Bootstrap_Liberals_plot,
  ARI_NDP_plot, Bootstrap_NDP_plot,
  ARI_Bloc_plot, Bootstrap_Bloc_plot,
  ARI_PPC_plot, Bootstrap_PPC_plot,
  ARI_Green_plot, Bootstrap_Green_plot,
  nrow = 6
)
```

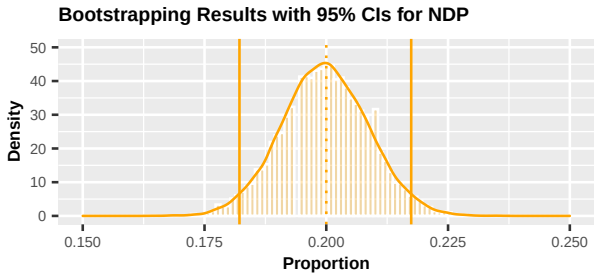

ARI's 95% Classical CIs for CPC
Density
Proportion

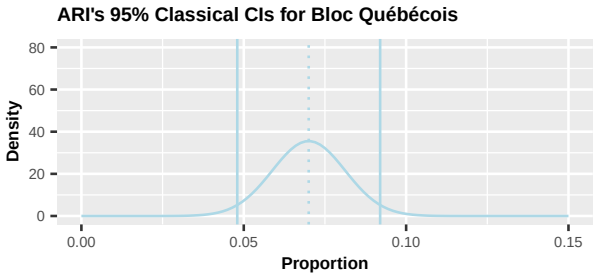

Bootstrapping Results with 95% CIs for CPC
Density
Proportion

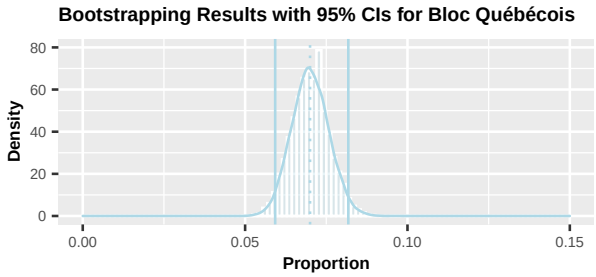

ARI's 95% Classical CIs for Liberals
Density
Proportion

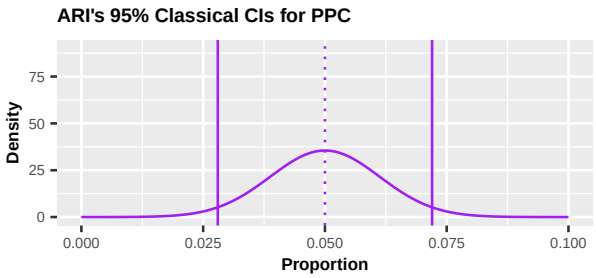

Bootstrapping Results with 95% CIs for Liberals
Density
Proportion

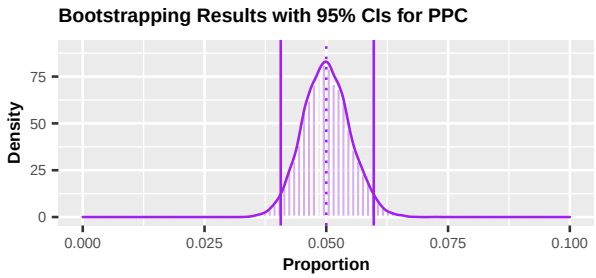

ARI's 95% Classical CIs for NDP
Density
Proportion

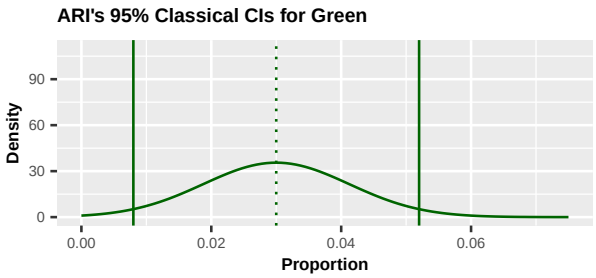

Bootstrapping Results with 95% CIs for NDP
Density
Proportion

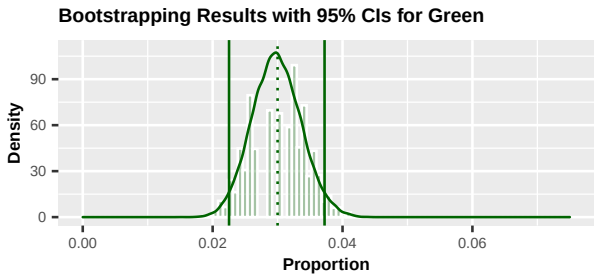

ARI's 95% Classical CIs for Bloc Québécois
Density
Proportion


Bootstrapping Results with 95% CIs for Bloc Québécois
Density
Proportion


ARI's 95% Classical CIs for PPC
Density
Proportion


Bootstrapping Results with 95% CIs for PPC
Density
Proportion


ARI's 95% Classical CIs for Green
Density
Proportion


Bootstrapping Results with 95% CIs for Green
Density
Proportion

The previous plot grid should show you the sampling distributions (either theoretical or empirical via bootstrapping) for each one of the six estimators $\hat{p}_{\text{party}_k}$. Now, **in two paragraphs**, explain the differences between both sets of plots. Which method do you think is more adequate? ARI's classical method or your bootstrapping method? Why?

ANSWER:

> *ARI's classical approach has some severe drawbacks. First off, it assumes the same sampling variability across the six proportion estimates. The Normal density plots corresponding to these estimators have the same standard deviation, which is a strong distributional assumption in surveys of this class. We cannot ensure the six groups of potential voters have the same variability! Furthermore, we can see that these 95% classical CIs (using the so-called* ***maximum margin of error****) are generally wider for all the parties than their bootstrapping counterparts.*
>
> *Now, from an inferential perspective, a better approach is bootstrapping. We can see how these intervals are more precise since they are narrower than their classical counterparts. Moreover, it is quite interesting to see that even though these bootstrapping distributions have bell shapes (****resembling*** *a Normal distribution!), their spreads heavily vary compared to their classical counterparts. This fact tells us that this* ***maximum margin of error*** *practice might not be ideal in this class of surveys.*

(Challenging) Q2.5.

rubric={reasoning:3}

We will get into a suitable sampling scheme depending on your population/system of interest. If we are not familiar with the Canadian political system, we might be surprised that the Prime Minister does not get elected by the popular vote but with the majority of seats in the Parliament (formally known as the **House of Commons**).

Let us check the basics of Canadian Politics 101 (taken from **Elections Canada website**):

> *Canada's political system is based on that of the United Kingdom. It is a constitutional monarchy, composed of the Queen of Canada, who is officially represented by the Governor General (or by a lieutenant-governor at the provincial level), the Senate and the House of Commons.*
>
> *There are 105 seats in the Senate, whose members are appointed by the Governor General on the recommendation of the Prime Minister. The House of Commons has* ***338 seats****, held by members elected by citizens who vote in general elections or by-elections. The Government originates in the elected House of Commons. According to the principle of constitutional monarchy, therefore, the Queen rules but does not govern.*
>
> ...
>
> *After a general election, by convention,* ***the leader of the party with the largest number of elected representatives will normally form the Government****. The Governor General will ask the leader of that party to be the Prime Minister. He or she must be able at all times to maintain the confidence of the House in order to remain in power. The party with the second-largest number of elected representatives is usually the official Opposition. The leader of this party is the Leader of the Opposition.*
>
> *The Prime Minister chooses people (usually members of the House of Commons of his or her party) to serve as the Cabinet ministers heading various government departments. Though not common, the Prime Minister can also appoint senators and others from outside of Parliament to Cabinet.*

Hence, let us check the **"surprising"** election results **here**. We can see that the Tories (the colloquial name of the Conservative Party of Canada, CPC) won the popular vote versus the Liberals (33.7 versus 32.6%). Still, the Liberals had a landslide victory in Parliament seats with 47.30% versus the Tories with 35.20%!

Thus, we might wonder: what is the point of these polls by popular vote if they are not taking the snapshot on a Parliament level?

Therefore, provide an alternative sampling scheme (**in two or three paragraphs**) supposing you are the Chief Data Scientist of a given electoral poll for this class of political system.

> **Heads-up:** There are not entirely right or wrong answers. Any sampling will always have its pros and cons. Make sure you highlight them.

Sampling: Design and Analysis **(Lohr, 2019)** might be a helpful resource to elaborate your answer. You can get a `.pdf` copy at the **UBC library** via your CWL and password.

POSSIBLE ANSWER:

> *As we already highlighted, any sampling scheme will have its pros and cons. Nevertheless, we might be particularly interested in a scheme called multi-stage sampling. This is a variation of cluster sampling, but it has an essential tweak, as we will see further.*
>
> *Standard cluster sampling is a pretty handy solution when we aim to cover a population that is too spread out. In such a big country like Canada, with 338 Electoral Districts (which represent Parliament seats), our sampling resources might be quite limited if we want to cover the whole electoral territory. Essentially, cluster sampling would split the electoral population into clusters (non-homogeneous groups). In this case, the clusters could be the 338 Electoral Districts. Then, we randomly select a subset of clusters (the number of clusters to sample will be a Power Analysis matter!). In these sampled clusters, we will poll all the people eligible to vote. People belonging to those non-sampled clusters will not be surveyed. We can see the following:*
>
> - **Pros:** *We would need fewer resources to travel since we will not have to go to all the Electoral Districts. Moreover, the standard error of our sampling estimates will be more realistic than the ones assumed by the ARI's poll.*
> - **Cons:** *We might lose valuable information from those non-sampled clusters. Furthermore, surveying all the eligible voters in those sampled clusters will be pretty unfeasible.*
>
> *Now, in terms of multi-stage sampling, this scheme also involves identifying clusters such as Electoral Districts. However, we would not sample a subset of clusters. Instead, we would include all 338 clusters in our sampling. Then, we will sample eligible voters from each cluster using a probability sampling method (e.g., simple random sampling by cluster). Again, the number of sampled subjects by cluster would be a Power Analysis matter. Note the following:*
>
> - **Pros:** *We would sample from each Electoral District. This will yield more accurate electoral estimates on a Parliament level. We could estimate how many seats could correspond to each party at the time the poll is run.*
> - **Cons:** *We would need more resources to travel since we will have to go to all the 338 Electoral Districts to sample from. We will also need to determine a secondary probability sampling method by cluster.*

Exercise 3: Bootstrapping and Penguins

Penguin researchers have hired you to help estimate **the population mean body size in grams (g)** of penguins from three different species: *Pygoscelis papua* (Gentoo), *Pygoscelis adeliae* (Adelie) and *Pygoscelis antarcticus* (Chinstrap). The researchers have told you they are defining the mean body size of penguins by the mean body mass for a given penguin species.

To do your inferential analysis, you will use the `penguins` data set from the `palmerpenguins R` package. Run the code chunk below to work with a copy called `penguins_sample` of **sample size** $n = 342$. Note we discard two penguins with missing data via `drop_na()`.

```
penguins_sample <- penguins %>%
  select(species, body_mass_g) %>%
  drop_na()
nrow(penguins_sample)
```

```
## [1] 342
```

Figure 1: Some cute penguins.

Your task is to use this data to estimate the population parameter, the μ_{mass_k}, for body mass in grams for each of the k penguin species in the data set.

Q3.1.

rubric={autograde:10}

Compute the estimated mean body mass per penguin, $\hat{\mu}_{\text{mass}_k}$ for the kth species. The estimator will be

$$\hat{\mu}_{\text{mass}_k} = \frac{\sum_{j=1}^{n_k} X_{k,j}}{n_k},$$

where:

- $X_{k,j}$ is the body mass weight for the jth penguin belonging to the kth species in the random sample.
- n_k is the number of penguins belonging to the kth species in the random sample.

Furthermore, compute the **95% bootstrap CIs** of each **estimated** body mass weight $\hat{\mu}_{\text{mass}_k}$ using **15,000 bootstrapped samples**. This will give you the corresponding uncertainty measures.

The output data frame `answer3_1` should look like below.

species	lower_ci	upper_ci	estimated_mean
Adelie			
Gentoo			
Chinstrap			

Heads-up: Round up all your **final** table results to 2 decimal places.

Code a function called `ci_mean()` which computes the **95% bootstrap CI** by species. It should return a tibble with one row and two columns, one for the lower bound and another for the upper bound. Use seed `552` for reproducibility within this function.

```
BEGIN QUESTION
name: Q3.1
points:
 - 0
 - 0
 - 2
 - 4
 - 4
```

```
answer3_1 <- NULL

# BEGIN SOLUTION
#' Confidence intervals for the mean
#'
#' @param sample tibble or data frame containing the sample data
#' @param var unquoted column name of the column for which the confidence
#' intervals are being calculated
#' @param level numeric vector of length one specifying the confidence level.
#' Default is 0.95.
#' @param type character vector of length one specifying the method to be used
#' for calculating the confidence intervals. Default is "percentile".
#'
#' @return a tibble with one row and two columns, one for the lower confidence
#' bound and one for the upper confidence bound
#' @export
#'
#' @examples
#'library(palmerpenguins)
#'ci_mean(penguins, body_mass_g)
ci_mean <- function(sample, var, level = 0.95, type = "percentile") {
  set.seed(552)
  sample %>%
    rep_sample_n(nrow(sample), replace = TRUE, reps = 15000) %>%
    summarise(stat = mean({{ var }})) %>%
    get_confidence_interval(level = level, type = type)
}

est_mass_means <- penguins_sample %>%
  select(species, body_mass_g) %>%
  drop_na() %>%
  group_by(species) %>%
  summarise(estimated_mean = round(mean(body_mass_g), 2))

raw_answer3_1 <- penguins_sample %>%
```

```
  select(species, body_mass_g) %>%
  drop_na() %>%
  group_by(species) %>%
  nest() %>%
  mutate(ci = map(data, ~ ci_mean(., body_mass_g))) %>%
  unnest(c(data, ci)) %>%
  left_join(est_mass_means) %>%
  nest(body_mass_g)
```

```
## Joining, by = "species"
```

```
answer3_1 <- raw_answer3_1[, c(
  "species",
  "lower_ci", "upper_ci",
  "estimated_mean"
)]

answer3_1 <- answer3_1 %>%
  mutate(lower_ci = round(lower_ci, 2), upper_ci = round(upper_ci, 2))
# END SOLUTION

answer3_1
```

```
## # A tibble: 3 x 4
## # Groups:   species [3]
##   species   lower_ci upper_ci estimated_mean
##   <fct>        <dbl>    <dbl>          <dbl>
## 1 Adelie       3627.    3774.          3701.
## 2 Gentoo       4988.    5164.          5076.
## 3 Chinstrap    3643.    3825.          3733.
```

Q3.2.

rubric={viz:6}

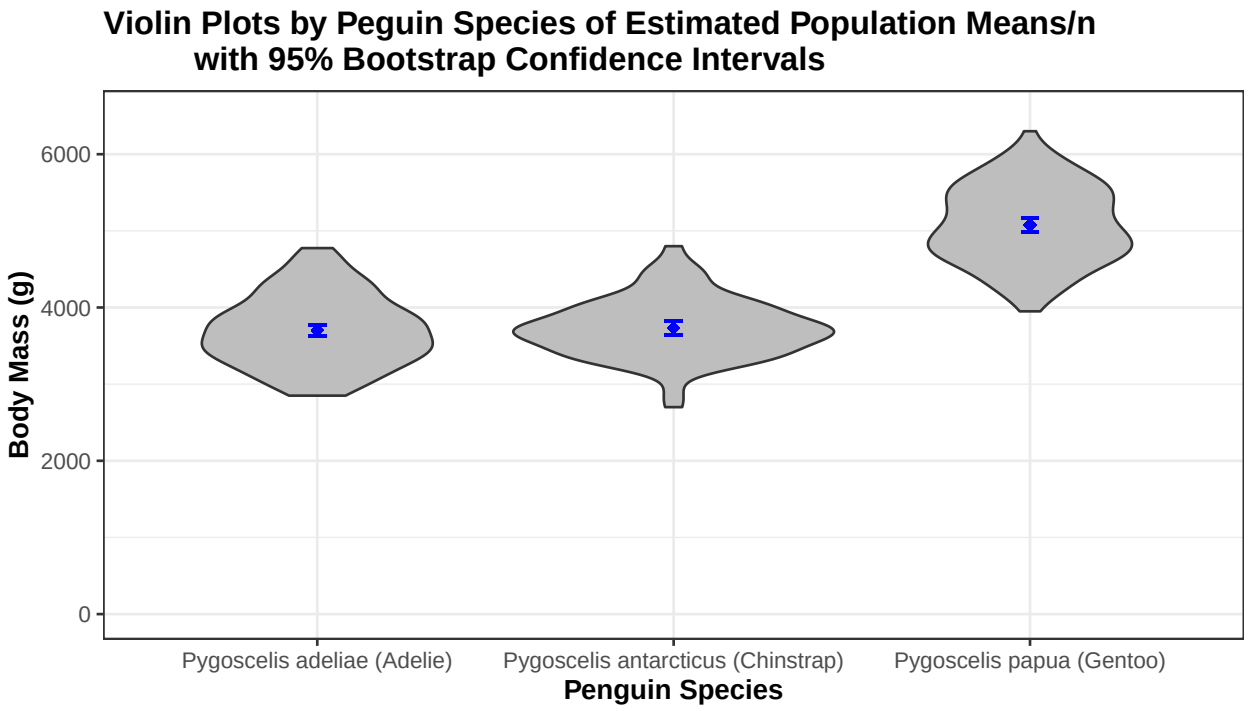
Visualize the estimate $\hat{\mu}_{\text{mass}_k}$, the 95% bootstrap confidence intervals, and the **sample distribution** on a plot, using a jitter plot, violin plot, or other suitable visualization that shows data distribution (**only pick one class of plot**).

You should map the kth penguin species to the x-axis and the estimate and confidence intervals to the y-axis. Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to the variable `penguin_sample_plot`.

Heads-up: There are no auto-grading test functions for this question.

```
penguin_sample_plot <- NULL

# BEGIN SOLUTION
penguin_sample_plot <- ggplot(raw_answer3_1 %>%
  unnest(data), aes(x = species, y = body_mass_g)) +
  geom_violin(fill = "grey") +
  geom_point(aes(x = species, y = estimated_mean),
    shape = 18, size = 2, color = "blue"
  ) +
  geom_errorbar(
    inherit.aes = FALSE,
    mapping = aes(
```

```
      x = species,
      ymin = lower_ci,
      ymax = upper_ci
    ),
    size = 0.5, color = "blue", width = 0.05
  ) +
  theme_bw() +
  xlab("Penguin Species") +
  ylab("Body Mass (g)") +
  ylim(c(0, 6500)) +
  scale_x_discrete(labels = c(
    "Adelie" =
      "Pygoscelis adeliae (Adelie)",
    "Chinstrap" =
      "Pygoscelis antarcticus (Chinstrap)",
    "Gentoo" =
      "Pygoscelis papua (Gentoo)"
  )) +
  ggtitle("Violin Plots by Peguin Species of Estimated Population Means/n
          with 95% Bootstrap Confidence Intervals") +
  theme(
    text = element_text(size = 11),
    plot.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold")
  )
# END SOLUTION

penguin_sample_plot
```



Q3.3.

rubric={reasoning:5}

In one or two paragraphs, use written English to interpret and report the estimates (and their bootstrap 95% CIs) for each of the k penguin species.

ANSWER:

The violin plots by species show that our sample has higher densities of sampled penguins around the estimated means. However, the Chinstrap species shows less spread in their sample distribution with a roughly symmetrical shape. We see a similar situation with the Adelie but with heavier tails. Note the samples means are pretty similar (3701g for Adelie and 3733g for Chinstrap). Moreover, their 95% bootstrap CIs are [3627, 3774] for Adelie and [3643, 3825] for Chinstrap. The CI overlap gives us statistical evidence to state that both body mass population means might be equal with a 95% confidence level.

Now, by checking the results for Gentoo, their violin plot show a bi-modal sample distribution whose estimated mean is 5076g which is larger than the estimated means for Adelie and Chinstrap. Note Gentoo's 95% bootstrap CI is not overlapped with the other two CIs, which gives us statistical evidence to state that Gentoo's body mass population mean is larger than the other two with a 95% confidence level.

Exercise 4: Hypothesis Test for a Difference in Means

Now, we want to know whether the mean body mass (in grams) of *Pygoscelis antarctica* (Chinstrap) is different from the mean body mass (in grams) of *Pygoscelis adeliae* (Adelie). To answer this question, we will use the data set `penguins_sample` again.

Your task is to answer this question by performing a hypothesis test that uses permutation and the `infer` package (see, for example, **this case**). Let us perform this analysis part by part.

Q4.1.

rubric={reasoning:4}

Using mathematical notation, **along with a proper parameter definition**, clearly specify the null, H_0, and alternative, H_A, hypotheses.

ANSWER:

Let $\mu_{mass_{Chinstrap}}$ and $\mu_{mass_{Adelie}}$ be the population means for the Chinstrap and Adelie penguins, respectively.

The hypotheses are the following:

$$H_0 : \mu_{\text{mass}_{Chinstrap}} = \mu_{\text{mass}_{Adelie}}$$
$$H_A : \mu_{\text{mass}_{Chinstrap}} \neq \mu_{\text{mass}_{Adelie}}$$

Q4.2.

rubric={reasoning:2}

Using mathematical notation, **along with a proper estimator definition**, define the test statistic you will use.

ANSWER:

Let

$$\hat{\mu}_{\text{mass}_k} = \frac{\sum_{j=1}^{n_k} X_{k,j}}{n_k},$$

be estimator of the population mean for the kth species, where:

- $X_{k,j}$ *is the body mass weight for the* j*th penguin belonging to the* k*th species in the random sample.*
- n_k *is the number of penguins belonging to the* k*th species in the random sample.*

The test statistic, δ^**, we will use is the difference between sample means:*

$$\delta^* = \hat{\mu}_{\text{mass}_{Chinstrap}} - \hat{\mu}_{\text{mass}_{Adelie}}$$

Q4.3.

rubric={accuracy:10}

Code the hypothesis test using the permutation method provided by the `infer` package with $r = 1000$ permuted data sets. Your code should return your test statistic from **Q4.2** as `chinstrap_adelie_test_stat`, and the p-value `chinstrap_adelie_p_value`. Use `552` as a simulation seed.

Hint: You will need to do some initial data wrangling with `penguins_sample`.

Heads-up: There are no auto-grading test functions for this question.

```
chinstrap_adelie_test_stat <- NULL
chinstrap_adelie_p_value <- NULL

# BEGIN SOLUTION
set.seed(552)

# Filtering out Gentoo
chinstrap_adelie <- penguins_sample %>%
  select(species, body_mass_g) %>%
  filter(species != "Gentoo") %>%
  mutate(species = fct_drop(species))

# Running permutation test
chinstrap_adelie_null_distribution <- chinstrap_adelie %>%
  specify(formula = body_mass_g ~ species) %>%
  hypothesize(null = "independence") %>%
  generate(reps = 1000, type = "permute") %>%
  calculate(
    stat = "diff in means",
    order = c("Chinstrap", "Adelie")
  )

chinstrap_adelie_test_stat <- chinstrap_adelie %>%
  specify(formula = body_mass_g ~ species) %>%
  calculate(
    stat = "diff in means",
    order = c("Chinstrap", "Adelie")
  )

chinstrap_adelie_p_value <- chinstrap_adelie_null_distribution %>%
  get_p_value(chinstrap_adelie_test_stat, direction = "both")
```

```
# END SOLUTION

chinstrap_adelie_test_stat
```

```
## Response: body_mass_g (numeric)
## Explanatory: species (factor)
## # A tibble: 1 x 1
##    stat
##   <dbl>
## 1  32.4
```

```
chinstrap_adelie_p_value
```

```
## # A tibble: 1 x 1
##   p_value
##     <dbl>
## 1   0.588
```

Q4.4.

rubric={viz:6}

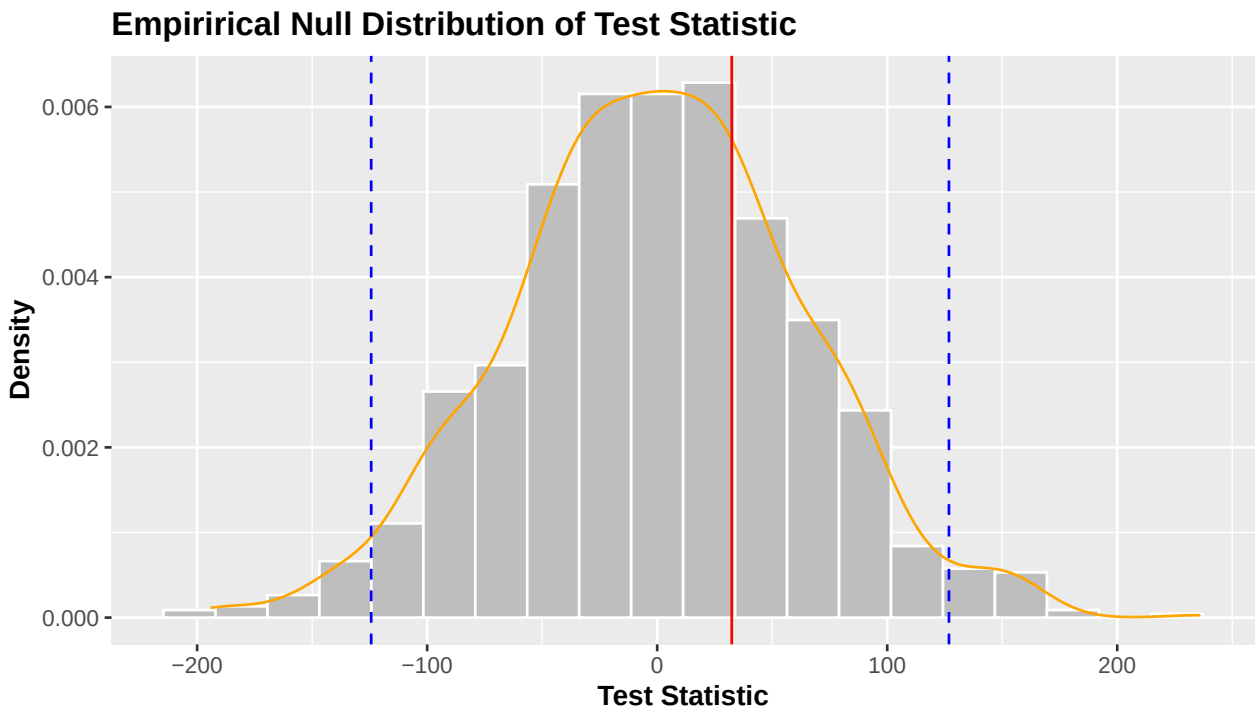
Visualize the **null distribution** you obtained as a histogram along with an estimated density plot (via `geom_density()`). Next, indicate your **OBSERVED test statistic** δ^* as a solid vertical red line and the corresponding empirical quantile thresholds as blue vertical dashed lines.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to the variable `penguin_null_dist_plot`.

Heads-up: There are no auto-grading test functions for this question.

```
penguin_null_dist_plot <- NULL

# BEGIN SOLUTION
alpha_threshold_1 <- quantile(chinstrap_adelie_null_distribution$stat, 0.025)
alpha_threshold_2 <- quantile(chinstrap_adelie_null_distribution$stat, 0.975)

penguin_null_dist_plot <- chinstrap_adelie_null_distribution %>%
  ggplot(aes(x = stat)) +
  geom_histogram(aes(y = ..density..),
    colour = "white", fill = "grey",
    bins = 20
  ) +
  geom_density(color = "orange") +
  ggtitle("Empiririical Null Distribution of Test Statistic") +
  theme(
    text = element_text(size = 11),
    plot.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold")
  ) +
  labs(x = "Test Statistic", y = "Density") +
  geom_vline(
    xintercept = alpha_threshold_1,
    color = "blue", linetype = "dashed"
  ) +
  geom_vline(
    xintercept = alpha_threshold_2,
```

```
    color = "blue", linetype = "dashed"
  ) +
  geom_vline(
    xintercept = chinstrap_adelie_test_stat$stat,
    color = "red"
  )
# END SOLUTION

penguin_null_dist_plot
```



Q4.5.

rubric={reasoning:5}

In **one paragraph**, report the results of your analysis, which should include the **effect size** (i.e., your test statistic δ^*) and the p-value.

ANSWER:

> *We estimate that the difference between the population body mass means of Chinstrap and Adelie species (as measured in grams) is 32. This is a small difference given the sample means for each species was 3733 for Chinstrap (95% bootstrap confidence interval is between 3643 and 3825) and 3701 (95% bootstrap confidence interval is between 3627 and 3774) for Adelie. Carrying out a difference of means permutation test also indicated that we do not have enough statistical evidence to state that there is a difference between the body mass as measured in grams for these species at a population level (p-value = 0.6).*

(Challenging) Exercise 5: Confidence Interval Simulation and Plotting

rubric={viz:3,accuracy:1}

Heads-up: There are no auto-grading test functions for this exercise

Ismay and Kim (2021) illustrate the concept of a confidence interval via a simulation using the percentile method for a population proportion. Let us check a similar simulation scheme with the data `vancouver_trees` from package `datateachr` for the continuous population mean `diameter` of public street trees located in `SUNSET`.

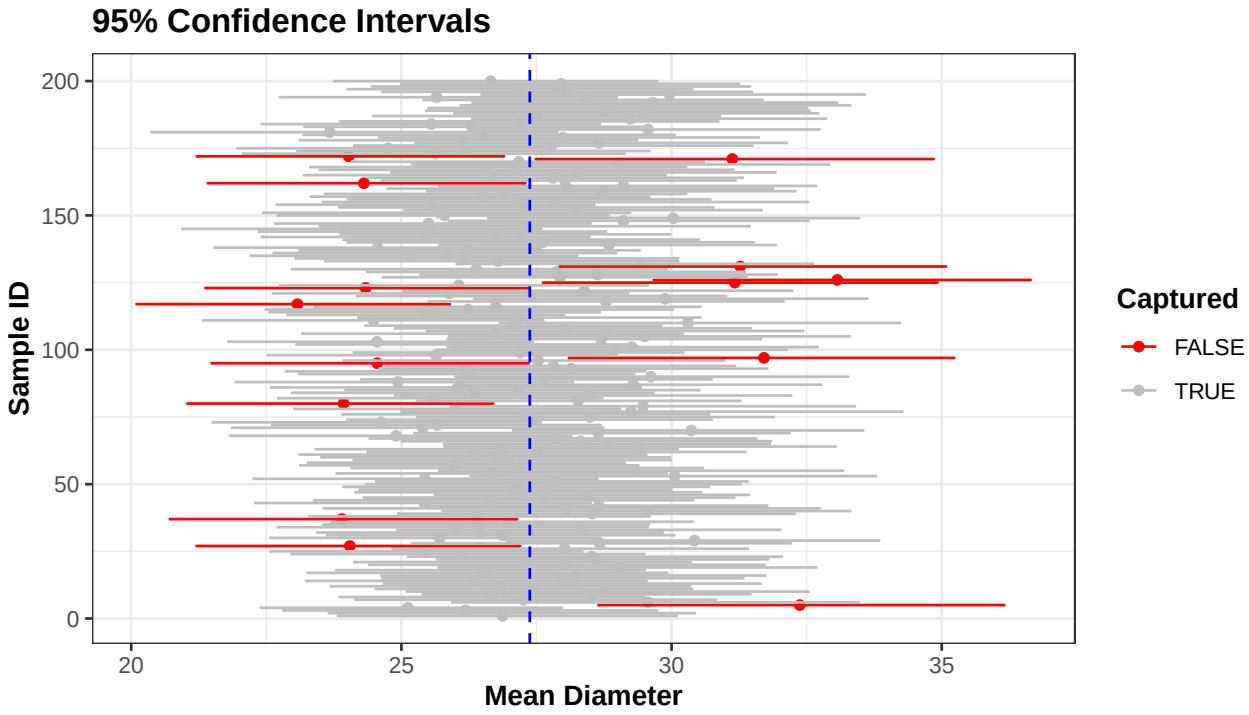
Code the following (**use simulation seeds when necessary**):

1. From `vancouver_trees`, filter those rows belonging to `neighbourhood_name SUNSET`. Call this data frame `sunset_population`
2. In `sunset_population`, select column `diameter`. Make sure to convert to the metric system (**inches to centimetres**).
3. Compute the `SUNSET` population mean diameter as a numeric vector `pop_mean_diameter`.
4. Draw 200 **different samples** of size `150` from `sunset_population`.
5. For each of these 200 different samples, draw 1000 bootstrap samples of the same size `150` to obtain the corresponding 95% CI via the percentile method.
6. Plot each one of these 200 95% bootstrap confidence intervals.
7. **The resulting plot** should show the population mean `pop_mean_diameter` as a dashed vertical blue line.
8. In contrast, each confidence interval associated with these 200 samples should be depicted as a horizontal line with their respective lower and upper bounds (**along with the center showing the original sample mean as a point**). The colours of the horizontal lines indicate whether `pop_mean_diameter` is captured by the confidence interval.

Ensure that your x and y-axes are human-readable. Moreover, include a title. Assign your plot to the variable `bootstrap_CI_plot`.

```
bootstrap_CI_plot <- NULL

# BEGIN SOLUTION
sunset_population <- vancouver_trees %>%
  filter(neighbourhood_name == "SUNSET") %>%
  select(diameter) %>%
  mutate(diameter = diameter * 2.54)

pop_mean_diameter <- mean(sunset_population$diameter)

multiple_samples <- sunset_population %>%
  rep_sample_n(reps = 200, size = 150, replace = FALSE) %>%
  summarise(mean_sample = mean(diameter), std_dev = sd(diameter))

number_samples <- 200

generate_bootstrap_df <- function(seed) {
  set.seed(seed)

  random_sample <- sunset_population %>%
    rep_sample_n(reps = 1, size = 150, replace = FALSE)

  sample_mean <- random_sample %>%
```

```
    ungroup() %>%
    summarize(Mean = mean(diameter))

  stats <- random_sample %>%
    ungroup() %>%
    select(diameter) %>%
    rep_sample_n(reps = 1000, size = 150, replace = TRUE) %>%
    group_by(replicate) %>%
    summarize(stat = mean(diameter)) %>%
    get_confidence_interval(level = 0.95, type = "percentile")

  stats <- cbind(sample_mean, stats)

  return(stats)
}

seeds <- seq(1:number_samples)
bootstrap_dfs <- lapply(seeds, generate_bootstrap_df)
bootstrap_dfs <- do.call(rbind, bootstrap_dfs)
bootstrap_dfs <- bootstrap_dfs %>%
  mutate(
    Captured = ifelse(lower_ci < pop_mean_diameter,
      ifelse(upper_ci > pop_mean_diameter, "TRUE", "FALSE"), "FALSE"
    ),
    Sample = 1:number_samples
  )

bootstrap_CI_plot <- bootstrap_dfs %>% ggplot(aes(x = Sample, y = Mean)) +
  geom_point(aes(color = Captured)) +
  scale_color_manual(values = c("red", "gray")) +
  geom_errorbar(aes(ymin = lower_ci, ymax = upper_ci, color = Captured)) +
  coord_flip() +
  theme_bw() +
  theme(
    text = element_text(size = 11),
    plot.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold"),
    legend.title = element_text(face = "bold"),
  ) +
  ggtitle("95% Confidence Intervals") +
  geom_hline(
    yintercept = pop_mean_diameter,
    colour = "blue",
    linetype = "dashed"
  ) +
  labs(y = "Mean Diameter", x = "Sample ID")
# END SOLUTION

bootstrap_CI_plot
```



Now, how many intervals (out of the 200, as a percentage!) caught the true population parameter? Compute the corresponding proportions from your previous data simulation.

```
# BEGIN SOLUTION
bootstrap_dfs %>%
  group_by(Captured) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum(n), 3))
```

```
## # A tibble: 2 x 3
##   Captured     n  freq
##   <chr>    <int> <dbl>
## 1 FALSE       14  0.07
## 2 TRUE       186  0.93
```

```
# END SOLUTION
```

Your final answer should be around 95%. Nonetheless, it will not be exactly 95% due to random noise.

Submission

Congratulations! You are done the lab!!! Do not forget to:

- Knit the assignment to generate `.pdf` file and push everything to your GitHub repo.
- Submit the `.Rmd` AND the `.pdf` files to Gradescope.