Inference for numerical data

Complete all **Exercises**

Getting Started

Load packages

In this lab we will explore the data using the `dplyr` package and visualize it using the `ggplot2` package for data visualization. The data can be found in the companion package for this course, `statsr`.

Let's load the packages.

```
library(statsr)
library(dplyr)
library(ggplot2)
```

The data

In 2004, the state of North Carolina released a large data set containing information on births recorded in this state. This data set is useful to researchers studying the relation between habits and practices of expectant mothers and the birth of their children. We will work with a random sample of observations from this data set.

Load the `nc` data set into our workspace.

```
data(nc)
```

We have observations on 13 different variables, some categorical and some numerical. The meaning of each variable is as follows.

variable	description
`fage`	father's age in years.
`mage`	mother's age in years.
`mature`	maturity status of mother.
`weeks`	length of pregnancy in weeks.
`premie`	whether the birth was classified as premature (premie) or full-term.
`visits`	number of hospital visits during pregnancy.
`marital`	whether mother is `married` or `not married` at birth.
`gained`	weight gained by mother during pregnancy in pounds.
`weight`	weight of the baby at birth in pounds.

variable	description
`lowbirthweight`	whether baby was classified as low birthweight (`low`) or not (`not low`).
`gender`	gender of the baby, `female` or `male` .
`habit`	status of the mother as a `nonsmoker` or a `smoker` .
`whitemom`	whether mom is `white` or `not white` .

1. There are 1,000 cases in this data set, what do the cases represent?
    1. The hospitals where the births took place
    2. The fathers of the children
    3. The days of the births
    4. **The births**

As a first step in the analysis, we should take a look at the variables in the dataset. This can be done using the `str` command:

```
str(nc)
```

```
## tibble [1,000 × 13] (S3: tbl_df/tbl/data.frame)
##  $ fage          : int [1:1000] NA NA 19 21 NA NA 18 17 NA 20 ...
##  $ mage          : int [1:1000] 13 14 15 15 15 15 15 15 16 16 ...
##  $ mature        : Factor w/ 2 levels "mature mom","younger mom": 2 2 2 2 2 2 2 2 2 2 ...
##  $ weeks         : int [1:1000] 39 42 37 41 39 38 37 35 38 37 ...
##  $ premie        : Factor w/ 2 levels "full term","premie": 1 1 1 1 1 1 1 2 1 1 ...
##  $ visits        : int [1:1000] 10 15 11 6 9 19 12 5 9 13 ...
##  $ marital       : Factor w/ 2 levels "married","not married": 1 1 1 1 1 1 1 1 1 1 ...
##  $ gained        : int [1:1000] 38 20 38 34 27 22 76 15 NA 52 ...
##  $ weight        : num [1:1000] 7.63 7.88 6.63 8 6.38 5.38 8.44 4.69 8.81 6.94 ...
##  $ lowbirthweight: Factor w/ 2 levels "low","not low": 2 2 2 2 2 1 2 1 2 2 ...
##  $ gender        : Factor w/ 2 levels "female","male": 2 2 1 2 1 2 2 2 2 1 ...
##  $ habit         : Factor w/ 2 levels "nonsmoker","smoker": 1 1 1 1 1 1 1 1 1 1 ...
##  $ whitemom      : Factor w/ 2 levels "not white","white": 1 1 2 2 1 1 1 1 2 2 ...
```

As you review the variable summaries, consider which variables are categorical and which are numerical. For numerical variables, are there outliers? If you aren't sure or want to take a closer look at the data, make a graph.

Exploratory data analysis

We will first start with analyzing the weight gained by mothers throughout the pregnancy: `gained` .

Using visualization and summary statistics, describe the distribution of weight gained by mothers during pregnancy. The `summary` function can also be useful.

```
summary(nc$gained)
```

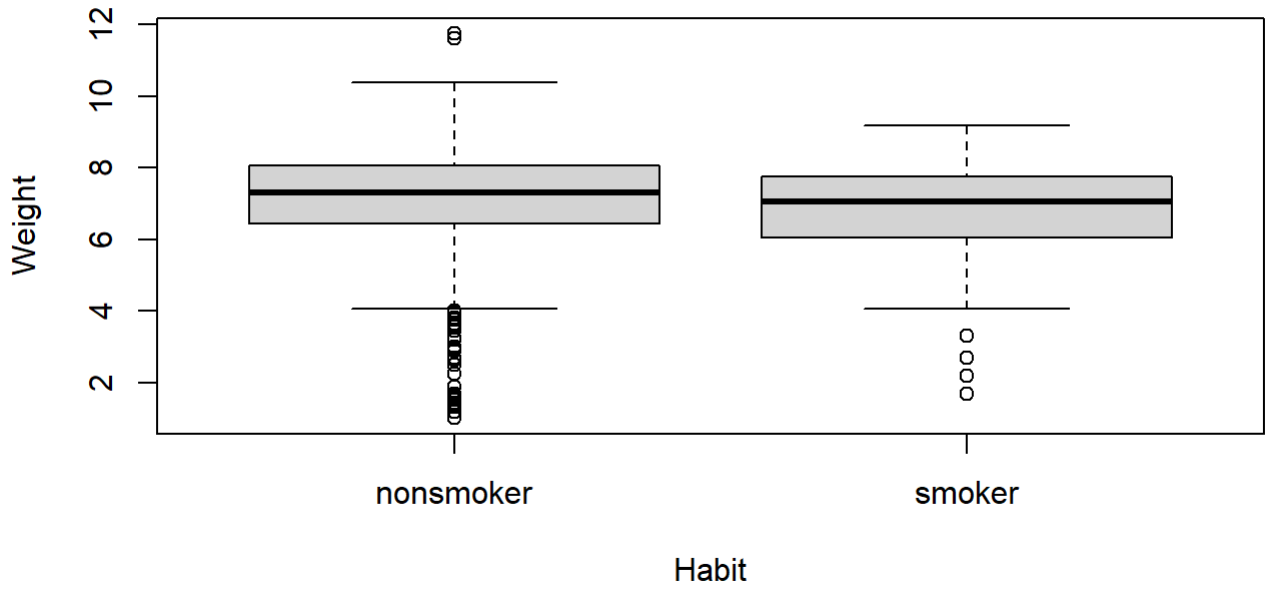
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.00   20.00   30.00   30.33   38.00   85.00      27
```

2. How many mothers are we missing weight gain data from?
    1. 0
    2. 13
    3. **27**
    4. 31

Next, consider the possible relationship between a mother's smoking habit and the weight of her baby. Plotting the data is a useful first step because it helps us quickly visualize trends, identify strong associations, and develop research questions.

3. Make side-by-side boxplots of `habit` and `weight`. Which of the following is false about the relationship between habit and weight?
    1. Median birth weight of babies born to non-smoker mothers is slightly higher than that of babies born to smoker mothers.
    2. **Range of birth weights of babies born to non-smoker mothers is greater than that of babies born to smoker mothers.**
    3. Both distributions are extremely right skewed.
    4. The IQRs of the distributions are roughly equal.

```
# type your code for the Question 3 here, and Knit

boxplot(weight ~ habit, data = nc, xlab = "Habit", ylab = "Weight")
```



The box plots show how the medians of the two distributions compare, but we can also compare the means of the distributions using the following to first group the data by the `habit` variable, and then calculate the mean `weight` in these groups using the `mean` function.

```
nc %>%
  group_by(habit) %>%
  summarise(mean_weight = mean(weight))
```

```
## # A tibble: 3 × 2
##   habit     mean_weight
##   <fct>           <dbl>
## 1 nonsmoker        7.14
## 2 smoker           6.83
## 3 <NA>             3.63
```

There is an observed difference, but is this difference statistically significant? In order to answer this question we will conduct a hypothesis test.

Inference

Exercise: Are all conditions necessary for inference satisfied? Comment on each. You can compute the group sizes using the same `by` command above but replacing `mean(weight)` with `n()`.

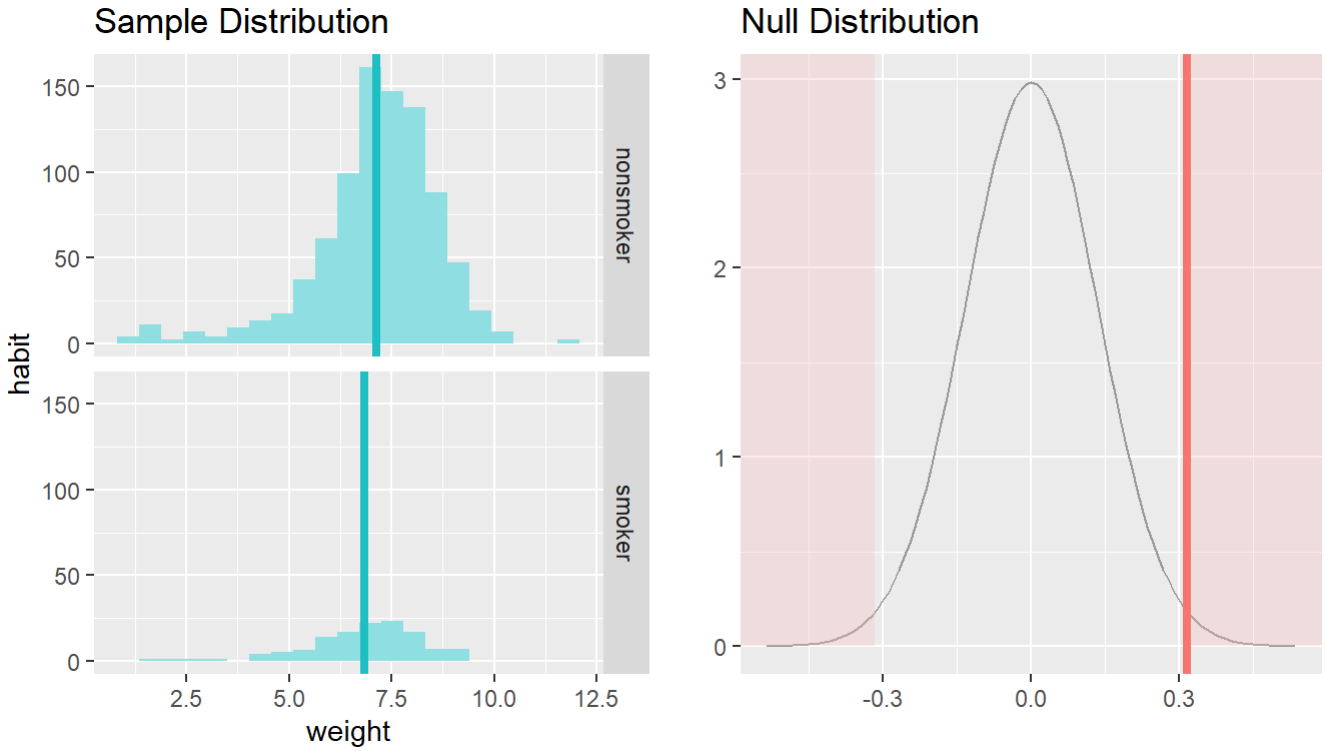
4. What are the hypotheses for testing if the average weights of babies born to smoking and non-smoking mothers are different?
    1. $H_0 : \mu_{smoking} = \mu_{non-smoking}$; $H_A : \mu_{smoking} > \mu_{non-smoking}$
    2. **$H_0: \mu_{smoking} = \mu_{non-smoking}$; $H_A : \mu_{smoking} \neq \mu_{non-smoking}$**
    3. $H_0 : \bar{x}_{smoking} = \bar{x}_{non-smoking}$; $H_A : \bar{x}_{smoking} > \bar{x}_{non-smoking}$
    4. $H_0 : \bar{x}_{smoking} = \bar{x}_{non-smoking}$; $H_A : \bar{x}_{smoking} > \bar{x}_{non-smoking}$
    5. $H_0 : \mu_{smoking} \neq \mu_{non-smoking}$; $H_A : \mu_{smoking} = \mu_{non-smoking}$

Next, we introduce a new function, `inference`, that we will use for conducting hypothesis tests and constructing confidence intervals.

Then, run the following:

```
inference(y = weight, x = habit, data = nc, statistic = "mean", type = "ht", null = 0,
          alternative = "twosided", method = "theoretical")
```

```
## Response variable: numerical
## Explanatory variable: categorical (2 levels)
## n_nonsmoker = 873, y_bar_nonsmoker = 7.1443, s_nonsmoker = 1.5187
## n_smoker = 126, y_bar_smoker = 6.8287, s_smoker = 1.3862
## H0: mu_nonsmoker =  mu_smoker
## HA: mu_nonsmoker != mu_smoker
## t = 2.359, df = 125
## p_value = 0.0199
```



Let's pause for a moment to go through the arguments of this custom function. The first argument is `y`, which is the response variable that we are interested in: `weight`. The second argument is the explanatory variable, `x`, which is the variable that splits the data into two groups, smokers and non-smokers: `habit`. The third argument, `data`, is the data frame these variables are stored in. Next is `statistic`, which is the sample statistic we're using, or similarly, the population parameter we're estimating. In future labs we can also work with "median" and "proportion". Next we decide on the `type` of inference we want: a hypothesis test (`"ht"`) or a confidence interval (`"ci"`). When performing a hypothesis test, we also need to supply the `null` value, which in this case is `0`, since the null hypothesis sets the two population means equal to each other. The `alternative` hypothesis can be `"less"`, `"greater"`, or `"twosided"`. Lastly, the `method` of inference can be `"theoretical"` or `"simulation"` based.

For more information on the inference function see the help file with `?inference`.

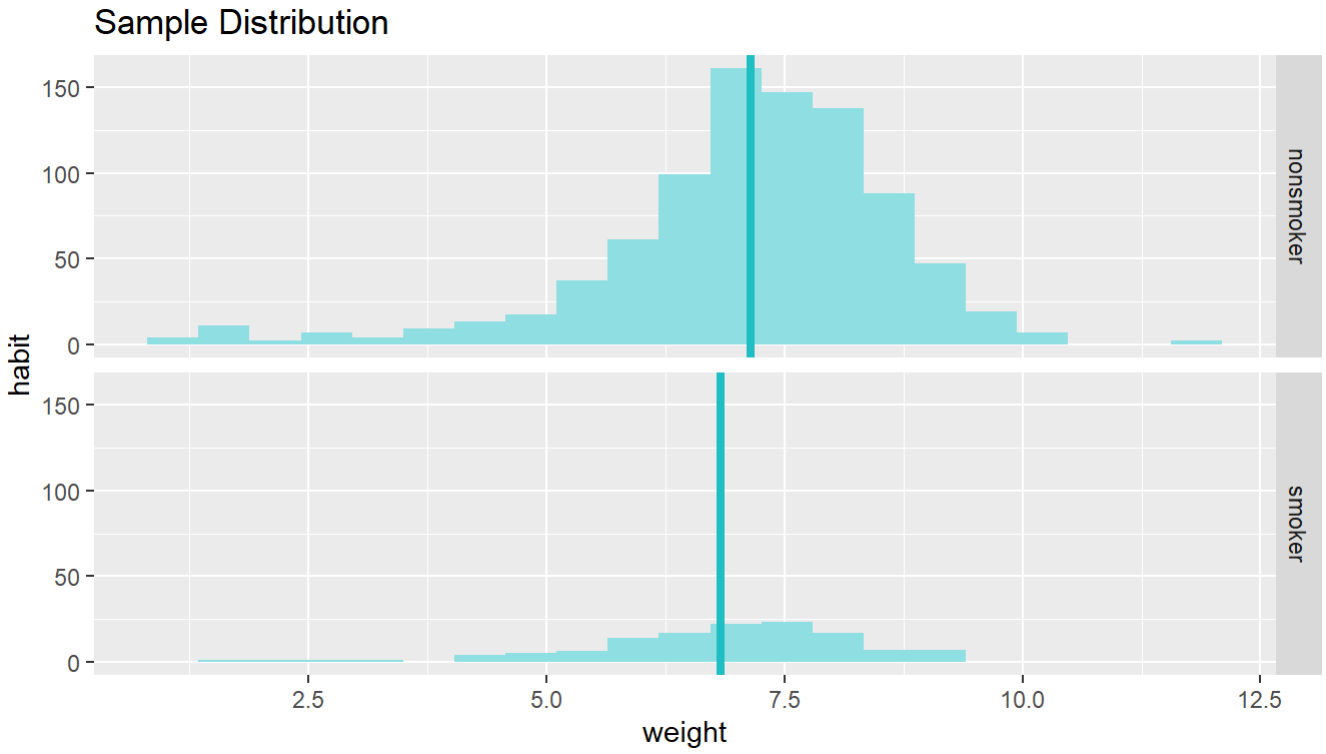
Exercise: What is the conclusion of the hypothesis test?

5. Change the `type` argument to `"ci"` to construct and record a confidence interval for the difference between the weights of babies born to nonsmoking and smoking mothers, and interpret this interval in context of the data. Note that by default you'll get a 95% confidence interval. If you want to change the confidence level, add a new argument (`conf_level`) which takes on a value between 0 and 1. Also note that when doing a confidence interval arguments like `null` and `alternative` are not useful, so make sure to remove them.

1. We are 95% confident that babies born to nonsmoker mothers are on average 0.05 to 0.58 pounds lighter at birth than babies born to smoker mothers.
2. We are 95% confident that the difference in average weights of babies whose moms are smokers and nonsmokers is between 0.05 to 0.58 pounds.
3. We are 95% confident that the difference in average weights of babies in this sample whose moms are smokers and nonsmokers is between 0.05 to 0.58 pounds.
4. **We are 95% confident that babies born to nonsmoker mothers are on average 0.05 to 0.58 pounds heavier at birth than babies born to smoker mothers.**

```
# type your code for the Question 5 here, and Knit

inference(y = weight, x = habit, data = nc, statistic = "mean", type = "ci", method = "theoretic
         al", conf_level = 0.95, order = c("smoker","nonsmoker"))
```

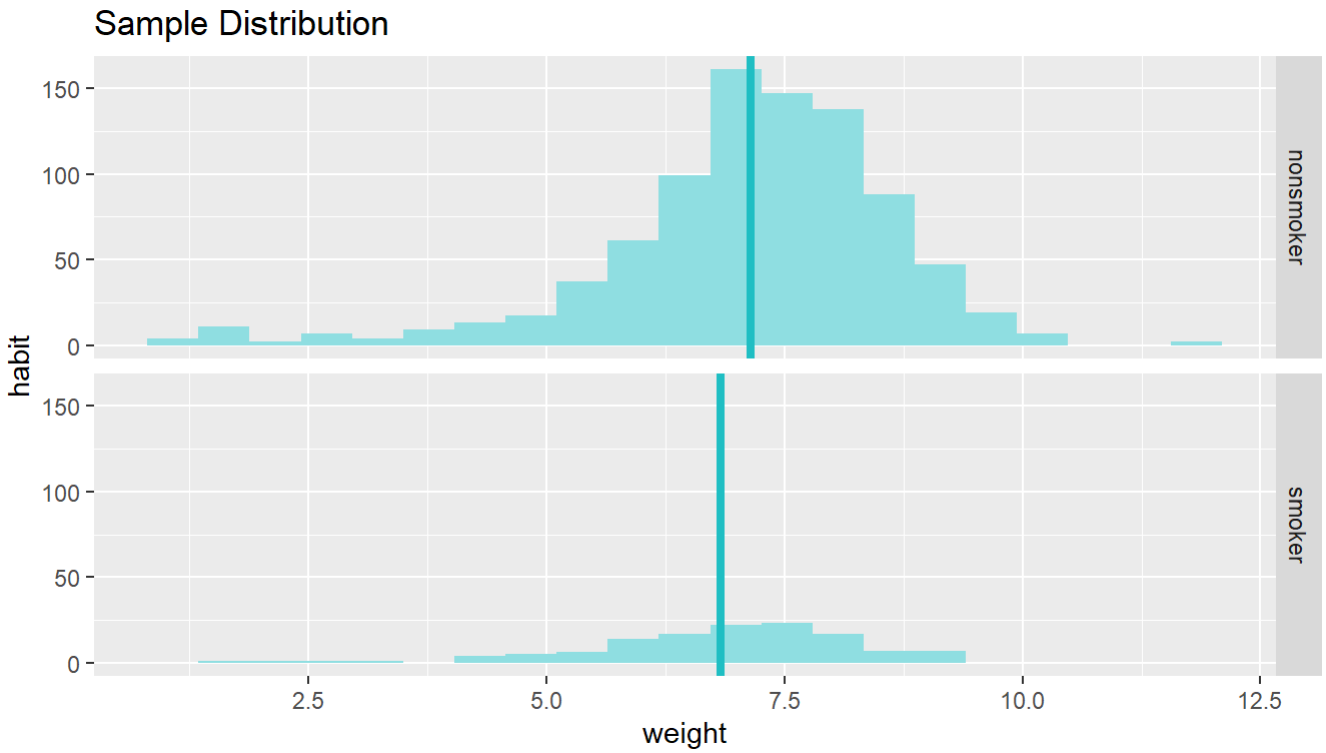
```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_smoker = 126, y_bar_smoker = 6.8287, s_smoker = 1.3862
## n_nonsmoker = 873, y_bar_nonsmoker = 7.1443, s_nonsmoker = 1.5187
## 95% CI (smoker - nonsmoker): (-0.5803 , -0.0508)
```



By default the function reports an interval for ($\mu_{nonsmoker} - \mu_{smoker}$) . We can easily change this order by using the `order` argument:

```
inference(y = weight, x = habit, data = nc, statistic = "mean", type = "ci",
          method = "theoretical", order = c("smoker","nonsmoker"))
```

```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_smoker = 126, y_bar_smoker = 6.8287, s_smoker = 1.3862
## n_nonsmoker = 873, y_bar_nonsmoker = 7.1443, s_nonsmoker = 1.5187
## 95% CI (smoker - nonsmoker): (-0.5803 , -0.0508)
```

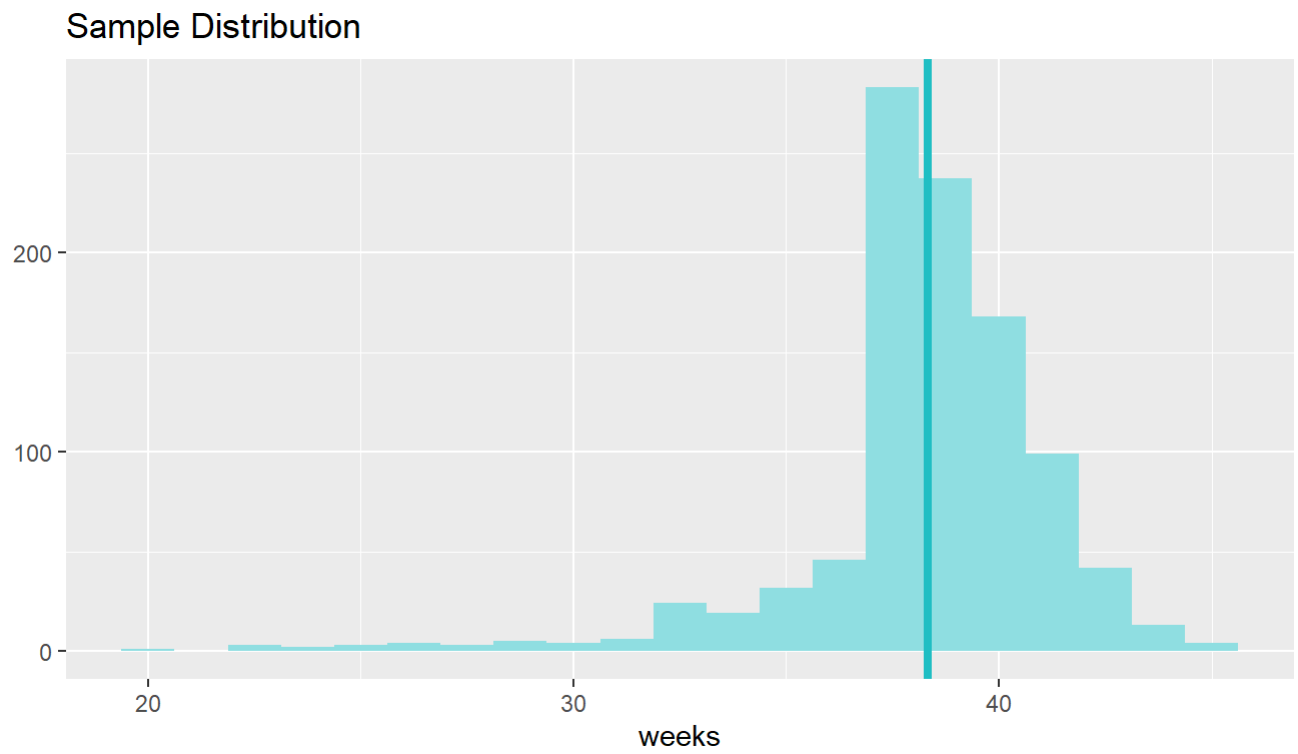


6. Calculate a 99% confidence interval for the average length of pregnancies (`weeks`). Note that since you're doing inference on a single population parameter, there is no explanatory variable, so you can omit the `x` variable from the function. Which of the following is the correct interpretation of this interval?

    1. (38.1526 , 38.5168)
    2. (38.0892 , 38.5661)
    3. (6.9779 , 7.2241)
    4. **(38.0952 , 38.5742)**

```
# type your code for Question 6 here, and Knit

inference(y=weeks, data=nc, type = "ci", statistic = "mean", order = NULL, method = "theoretica
        l", null = 0, alternative = "twosided", conf_level = 0.99)
```

```
## Single numerical variable
## n = 998, y-bar = 38.3347, s = 2.9316
## 99% CI: (38.0952 , 38.5742)
```

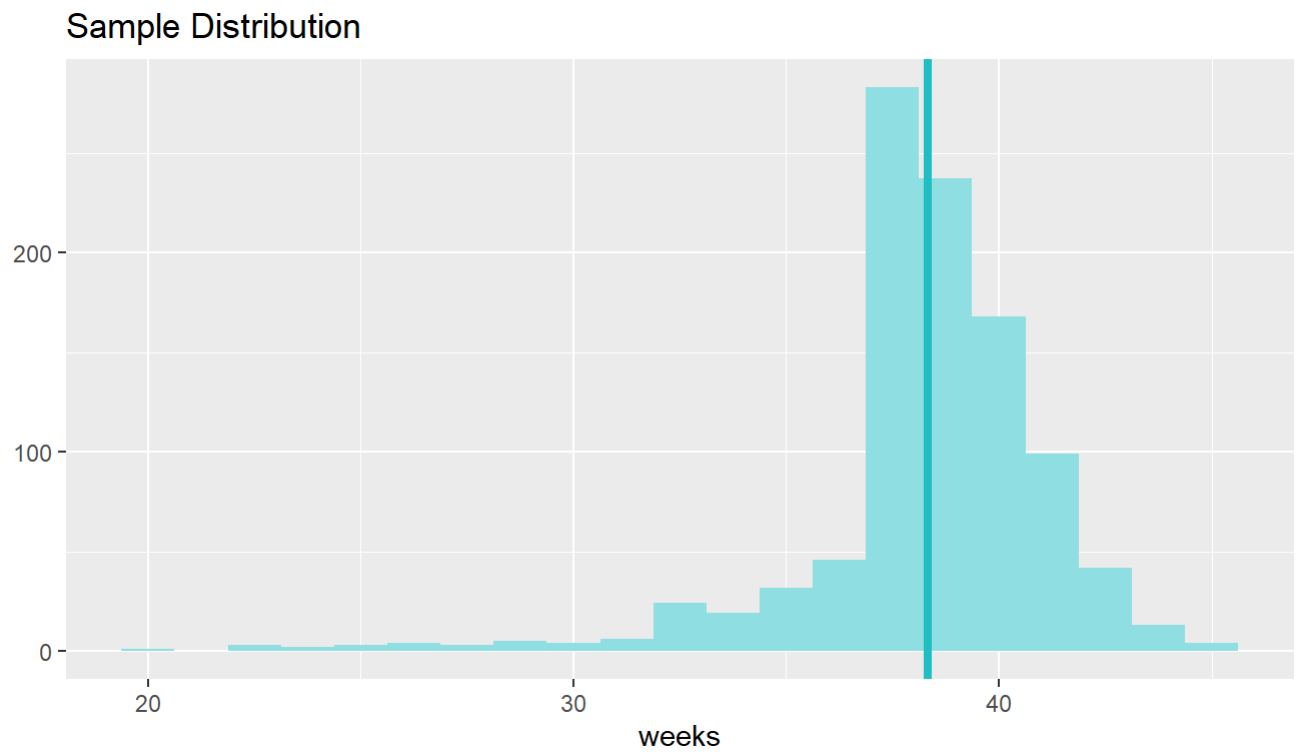


Based on result we can say on 99% confident level average length of pregnancies in between 38 weeks and 38.6 weeks.

Exercise: Calculate a new confidence interval for the same parameter at the 90% confidence level. Comment on the width of this interval versus the one obtained in the the previous exercise.

```
# type your code for the Exercise here, and Knit

inference(y=weeks, data=nc, type = "ci", statistic = "mean", order = NULL, method = "theoretica
         l", null = 0, alternative = "twosided", conf_level = 0.90)
```

```
## Single numerical variable
## n = 998, y-bar = 38.3347, s = 2.9316
## 90% CI: (38.1819 , 38.4874)
```

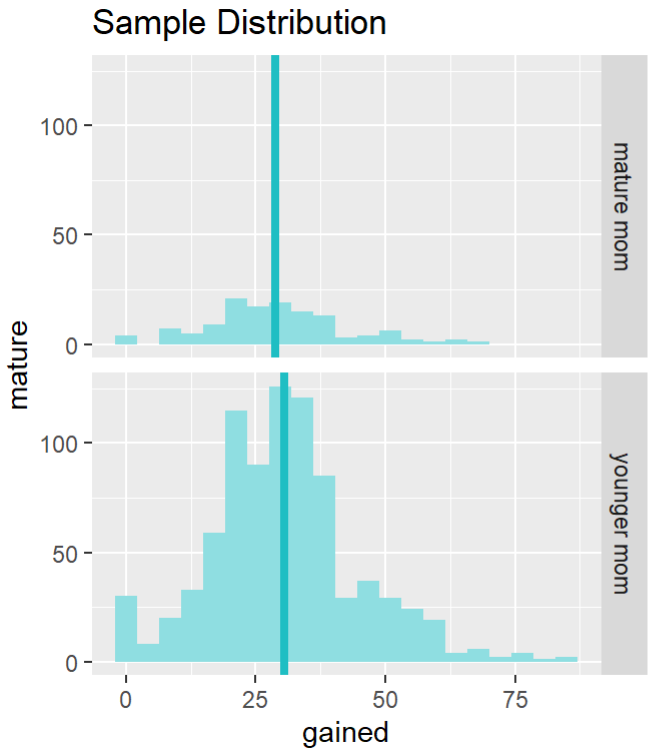


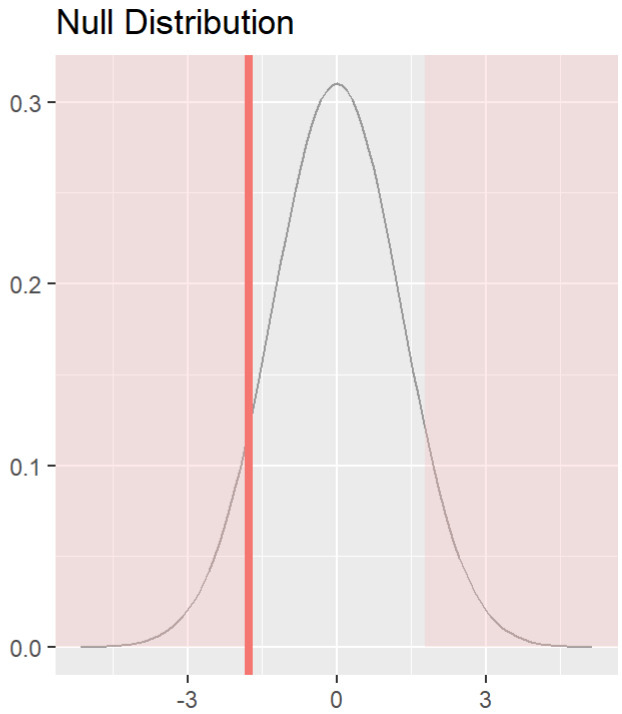
Based on result we can say on 90% confident level average length of pregnancies in between 38.2 weeks and 38.5 weeks. So, we can say it's narrower then previous one.

Exercise: Conduct a hypothesis test evaluating whether the average weight gained by younger mothers is different than the average weight gained by mature mothers.

```
# type your code for the Exercise here, and Knit

inference(y = gained, x = mature, data = nc, statistic = "mean", type = "ht", method = "theoreti
         cal", null=0, alternative = "twosided")
```

```
## Response variable: numerical
## Explanatory variable: categorical (2 levels)
## n_mature mom = 129, y_bar_mature mom = 28.7907, s_mature mom = 13.4824
## n_younger mom = 844, y_bar_younger mom = 30.5604, s_younger mom = 14.3469
## H0: mu_mature mom =  mu_younger mom
## HA: mu_mature mom != mu_younger mom
## t = -1.3765, df = 128
## p_value = 0.1711
```

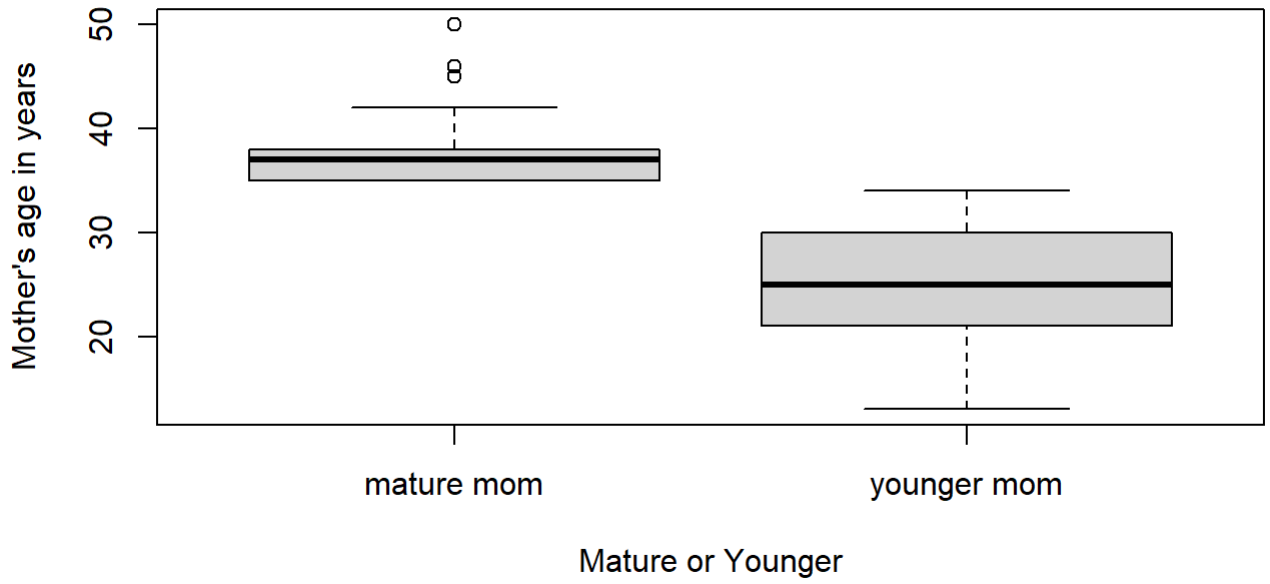




Here we can see the p-value is greater than the significance level which is 0.05. Now we fail to reject the null hypothesis. As a result the average weight gained by younger mothers is equal to the average weight gained by mature mothers.

7. Now, a non-inference task: Determine the age cutoff for younger and mature mothers. Use a method of your choice, and explain how your method works.

```
# type your code for Question 7 here, and Knit

boxplot(mage ~ mature, data = nc, xlab = "Mature or Younger",  ylab = "Mother's age in years")
```




```
by(nc$mage, nc$mature, summary)
```

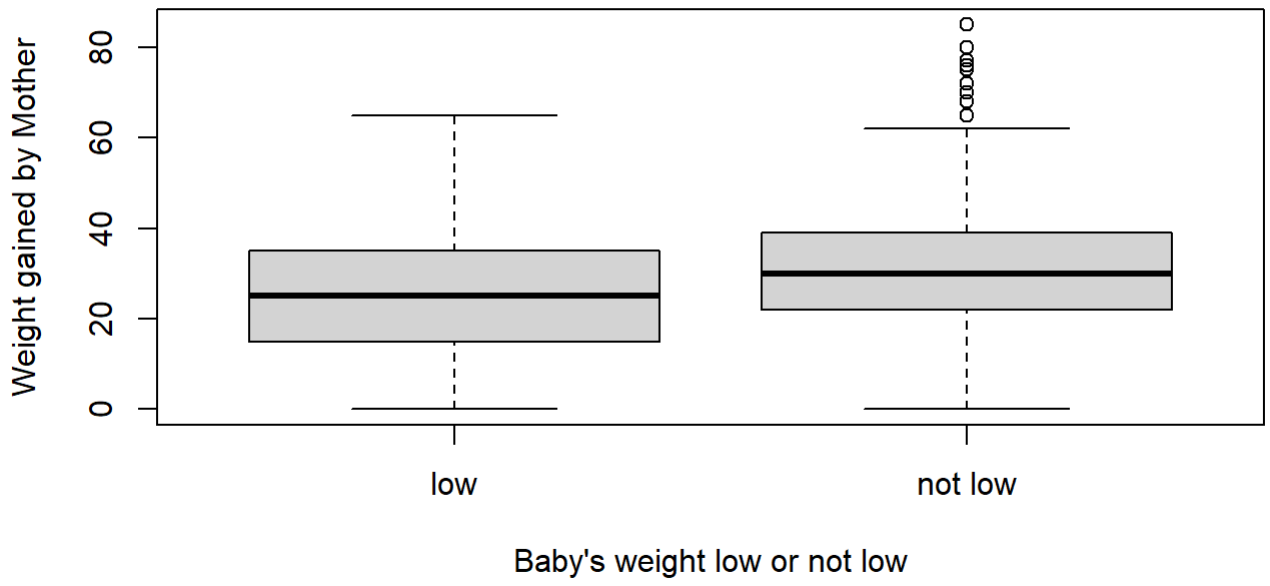
```
## nc$mature: mature mom
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   35.00   35.00   37.00   37.18   38.00   50.00
## ------------------------------------------------------------
## nc$mature: younger mom
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.00   21.00   25.00   25.44   30.00   34.00
```

Here i used by() function, which makes subsets. and used summery to measure the groups. From the result we can see maximum age of younger moms is 34 and minimum age of mature moms is 35.

Exercise: Pick a pair of variables: one numerical (response) and one categorical (explanatory). Come up with a research question evaluating the relationship between these variables. Formulate the question in a way that it can be answered using a hypothesis test and/or a confidence interval. Answer your question using the `inference` function, report the statistical results, and also provide an explanation in plain language. Be sure to check all assumptions,state your α level, and conclude in context. (Note: Picking your own variables, coming up with a research question, and analyzing the data to answer this question is basically what you'll need to do for your project as well.)

I am choosing the variables `gained` (response) and `lowbirthweight` (explanatory). I want to know whether the data set provides statistical significant evidence that weight gained by mother during pregnancy is, on average, is depend on whether baby was classified as low birth weight (`low`) or not (`not low`).

```
# type your code for the Exercise here, and Knit

boxplot(gained ~ lowbirthweight, data = nc, xlab = "Baby's weight low or not low", ylab = "Weigh
        t gained by Mother")
```

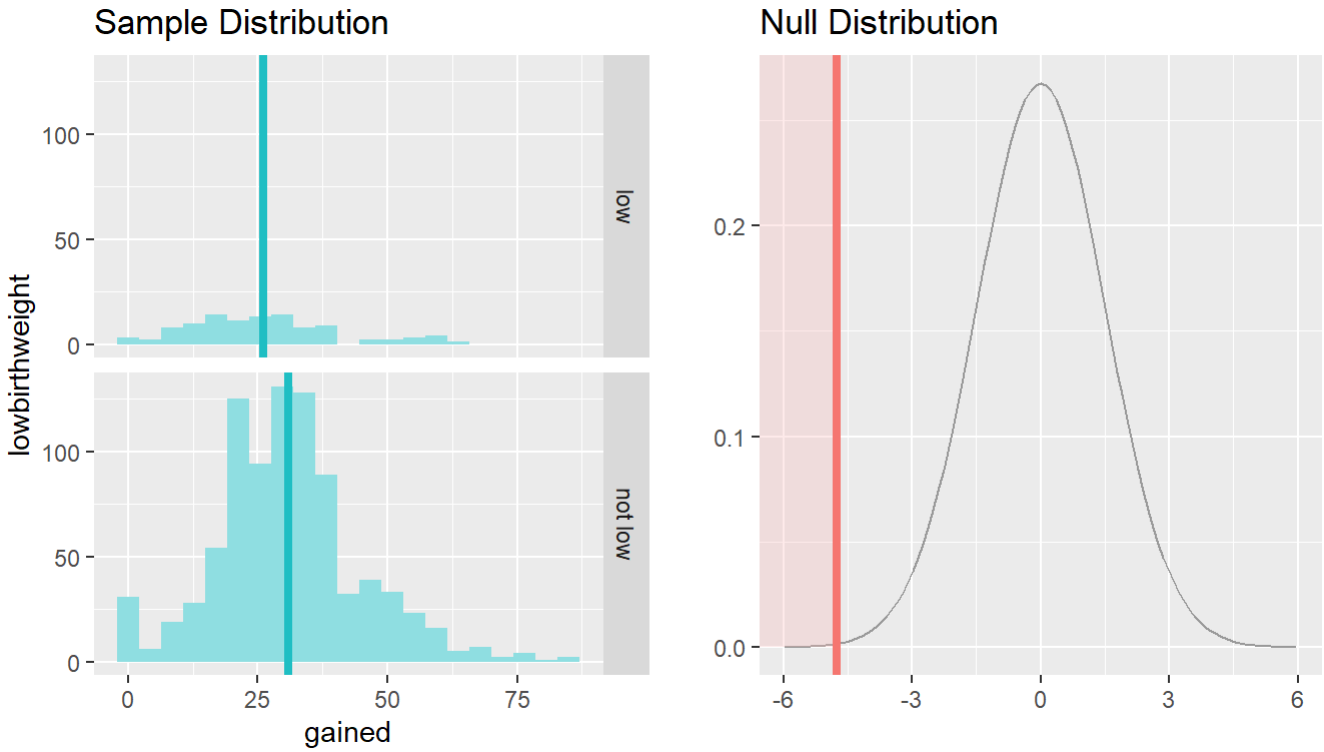


We will conduct the following hypothesis test:

$$H_0 : \mu_{low} = \mu_{notlow}$$
$$H_A : \mu_{low} < \mu_{notlow}$$

```
inference(y = gained, x = lowbirthweight, data = nc, statistic = "mean", type = "ht", null = 0,
          alternative = "less", method = "theoretical")
```

```
## Response variable: numerical
## Explanatory variable: categorical (2 levels)
## n_low = 104, y_bar_low = 26.0769, s_low = 14.4065
## n_not low = 869, y_bar_not low = 30.8343, s_not low = 14.1444
## H0: mu_low =  mu_not low
## HA: mu_low < mu_not low
## t = -3.1887, df = 103
## p_value = 9e-04
```



Here we can see the p-value is less than the significance level which is 0.05. Now we can reject the null hypothesis. And say that baby's weight depends on mother's average weight gaining.

Foundations for inference - Confidence intervals

Complete all **Exercises**

If you have access to data on an entire population, say the size of every house in Ames, Iowa, it's straight forward to answer questions like, "How big is the typical house in Ames?" and "How much variation is there in sizes of houses?". If you have access to only a sample of the population, as is often the case, the task becomes more complicated. What is your best guess for the typical size if you only know the sizes of several dozen houses? This sort of situation requires that you use your sample to make inference on what your population looks like.

Setting a seed: We will take some random samples and calculate confidence based on these samples in this lab, which means you should set a seed on top of your lab. If this concept is new to you, review the previous lab and ask your TA.

Setting a seed will cause R to sample the same sample each time you knit your document. This will make sure your results don't change each time you knit, and it will also ensure reproducibility of your work (by setting the same seed it will be possible to reproduce your results). You can set a seed like this:

```
set.seed(07291995)                # make sure to change the seed
```

The number above is completely arbitraty. If you need inspiration, you can use your ID, birthday, or just a random string of numbers. The important thing is that you use each seed only once. You only need to do this once in your R Markdown document, but make sure it comes before sampling.

Getting Started

Load packages

In this lab we will explore the data using the `dplyr` package and visualize it using the `ggplot2` package for data visualization. The data can be found in the companion package for this course, `statsr`.

Let's load the packages.

```
library(statsr)
library(dplyr)
library(ggplot2)
```

The data

We consider real estate data from the city of Ames, Iowa. This is the same dataset used in the previous lab. The details of every real estate transaction in Ames is recorded by the City Assessor's office. Our particular focus for this lab will be all residential home sales in Ames between 2006 and 2010. This collection represents our population of interest. In this lab we would like to learn about these home sales by taking smaller samples from the full population. Let's load the data.

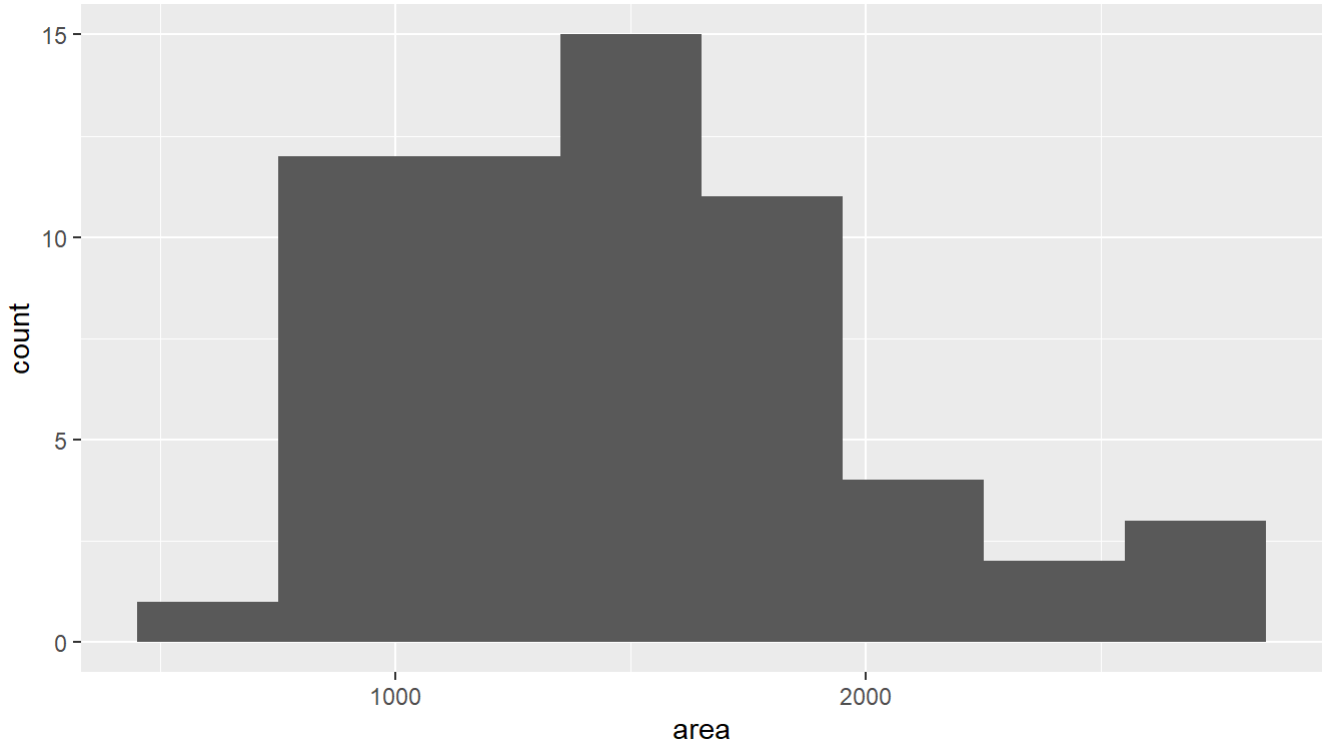
```
data(ames)
```

In this lab we'll start with a simple random sample of size 60 from the population. Specifically, this is a simple random sample of size 60. Note that the data set has information on many housing variables, but for the first portion of the lab we'll focus on the size of the house, represented by the variable `area`.

```
n <- 60
samp <- sample_n(ames, n)
```

Exercise: Describe the distribution of homes in your sample. What would you say is the "typical" size within your sample? Also state precisely what you interpreted "typical" to mean.

```
# type your code for the Exercise here, and Knit

ggplot(data = samp, aes(x=area))+geom_histogram(binwidth = 300)
```



1. True or False: My distribution should be similar to others' distributions who also collect random samples from this population, but it is likely not exactly the same since it's a random sample.
    1. **True.**
    2. False.

Confidence intervals

Return for a moment to the question that first motivated this lab: based on this sample, what can we infer about the population? Based only on this single sample, the best estimate of the average living area of houses sold in Ames would be the sample mean, usually denoted as *[Math Processing Error]* (here we're calling it `x_bar`). That serves

as a good **point estimate** but it would be useful to also communicate how uncertain we are of that estimate. This uncertainty can be quantified using a **confidence interval**.

A confidence interval for a population mean is of the following form *[Math Processing Error]*

You should by now be comfortable with calculating the mean and standard deviation of a sample in R. And we know that the sample size is 60. So the only remaining building block is finding the appropriate critical value for a given confidence level. We can use the `qnorm` function for this task, which will give the critical value associated with a given percentile under the normal distribution. Remember that confidence levels and percentiles are not equivalent. For example, a 95% confidence level refers to the middle 95% of the distribution, and the critical value associated with this area will correspond to the 97.5th percentile.

We can find the critical value for a 95% confidence interal using

```
z_star_95 <- qnorm(0.975)
z_star_95
```

```
## [1] 1.959964
```

which is roughly equal to the value critical value 1.96 that you're likely familiar with by now.

Let's finally calculate the confidence interval:

```
samp %>%
  summarise(lower = mean(area) - z_star_95 * (sd(area) / sqrt(n)),
            upper = mean(area) + z_star_95 * (sd(area) / sqrt(n)))
```

```
## # A tibble: 1 × 2
##   lower upper
##   <dbl> <dbl>
## 1 1344. 1595.
```

To recap: even though we don't know what the full population looks like, we're 95% confident that the true average size of houses in Ames lies between the values *lower* and *upper*. There are a few conditions that must be met for this interval to be valid.

2. For the confidence interval to be valid, the sample mean must be normally distributed and have standard error *[Math Processing Error]*. Which of the following is not a condition needed for this to be true?

    1. The sample is random.
    2. The sample size, 60, is less than 10% of all houses.
    3. **The sample distribution must be nearly normal.**

Confidence levels

3. What does "95% confidence" mean?

    1. 95% of the time the true average area of houses in Ames, Iowa, will be in this interval.

2. **95% of random samples of size 60 will yield confidence intervals that contain the true average area of houses in Ames, Iowa.**
3. 95% of the houses in Ames have an area in this interval.
4. 95% confident that the sample mean is in this interval.

In this case we have the rare luxury of knowing the true population mean since we have data on the entire population. Let's calculate this value so that we can determine if our confidence intervals actually capture it. We'll store it in a data frame called `params` (short for population parameters), and name it `mu`.

```
params <- ames %>%
  summarise(mu = mean(area))
```

Exercise: Does your confidence interval capture the true average size of houses in Ames?

```
# type your code for the Exercise here, and Knit
params
```

```
## # A tibble: 1 × 1
##      mu
##   <dbl>
## 1 1500.
```

4. What proportion of 95% confidence intervals would you expect to capture the true population mean?
    1. 1%
    2. 5%
    3. 95%
    4. **99%**

Using R, we're going to collect many samples to learn more about how sample means and confidence intervals vary from one sample to another.

Here is the rough outline:

- Obtain a random sample.
- Calculate the sample's mean and standard deviation, and use these to calculate and store the lower and upper bounds of the confidence intervals.
- Repeat these steps 50 times.

We can accomplish this using the `rep_sample_n` function. The following lines of code takes 50 random samples of size `n` from population (and remember we defined *[Math Processing Error]* earlier), and computes the upper and lower bounds of the confidence intervals based on these samples.

```
ci <- ames %>%
        rep_sample_n(size = n, reps = 50, replace = TRUE) %>%
        summarise(lower = mean(area) - z_star_95 * (sd(area) / sqrt(n)),
                  upper = mean(area) + z_star_95 * (sd(area) / sqrt(n)))
```

Let's view the first five intervals:

```
ci %>%
  slice(1:5)
```

```
## # A tibble: 5 × 3
##   replicate lower upper
##       <int> <dbl> <dbl>
## 1         1 1464. 1687.
## 2         2 1376. 1602.
## 3         3 1271. 1539.
## 4         4 1351. 1567.
## 5         5 1384. 1616.
```

Next we'll create a plot similar to Figure 4.8 on page 175 of OpenIntro Statistics, 3rd Edition (https://www.openintro.org/os). First step will be to create a new variable in the `ci` data frame that indicates whether the interval does or does not capture the true population mean. Note that capturing this value would mean the lower bound of the confidence interval is below the value and upper bound of the confidence interval is above the value. Remember that we create new variables using the `mutate` function.

```
ci <- ci %>%
  mutate(capture_mu = ifelse(lower < params$mu & upper > params$mu, "yes", "no"))
```

The `ifelse` function is new. It takes three arguments: first is a logical statement, second is the value we want if the logical statement yields a true result, and the third is the value we want if the logical statement yields a false result.

We now have all the information we need to create the plot, but we need to re-organize our data a bit for easy plotting. Specifically, we need to organize the data in a new data frame where each row represents one bound, as opposed to one interval. So this

```
     lower    upper capture_mu
1 1350.540 1544.360        yes
2 1333.441 1584.425        yes
3 1412.133 1663.801        yes
...
```

should instead look like

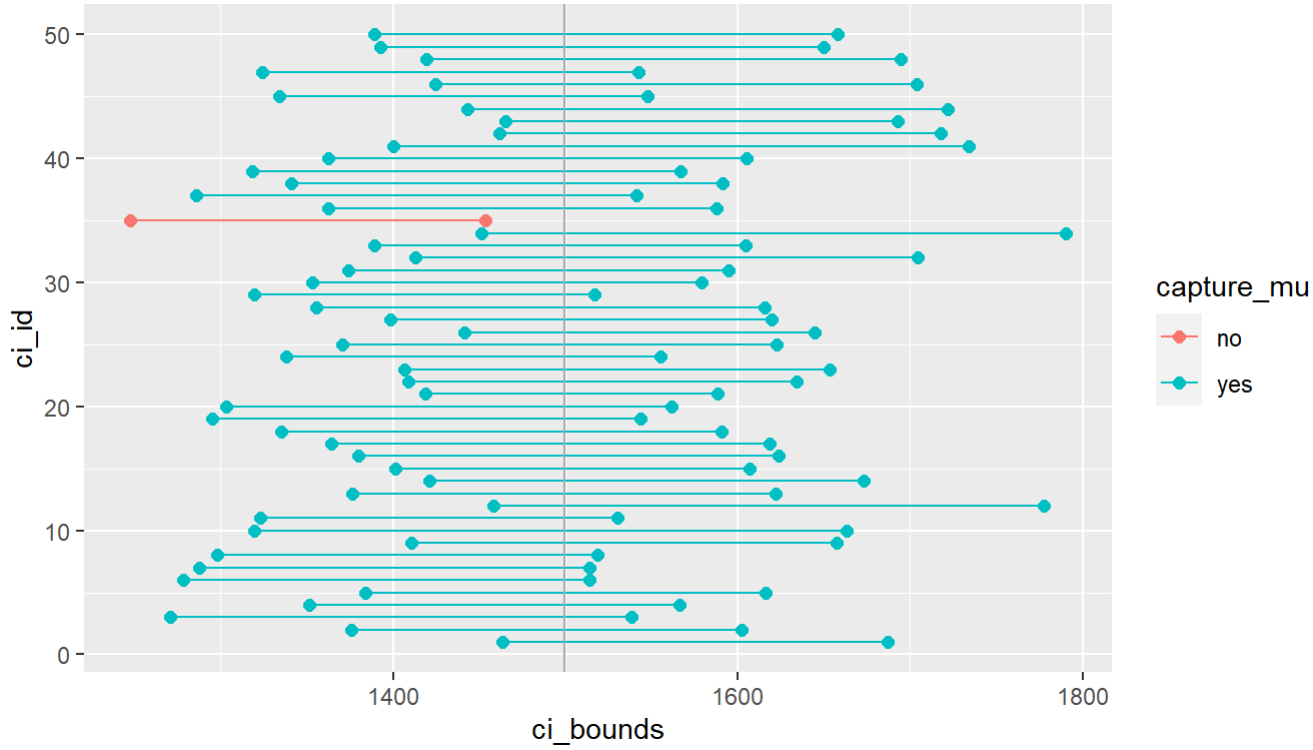
```
  ci_id ci_bounds capture_mu
1     1  1350.540        yes
2     2  1333.441        yes
3     3  1412.133        yes
4     1  1544.360        yes
5     2  1584.425        yes
6     3  1663.801        yes
...
```

We can accomplish this using the following:

```
ci_data <- data.frame(ci_id = c(1:50, 1:50),
                      ci_bounds = c(ci$lower, ci$upper),
                      capture_mu = c(ci$capture_mu, ci$capture_mu))
```

And finally we can create the plot using the following:

```
ggplot(data = ci_data, aes(x = ci_bounds, y = ci_id,
                           group = ci_id, color = capture_mu)) +
  geom_point(size = 2) +  # add points at the ends, size = 2
  geom_line() +           # connect with lines
  geom_vline(xintercept = params$mu, color = "darkgray") # draw vertical line
```



Exercise: What proportion of your confidence intervals include the true population mean? Is this proportion exactly equal to the confidence level? If not, explain why.

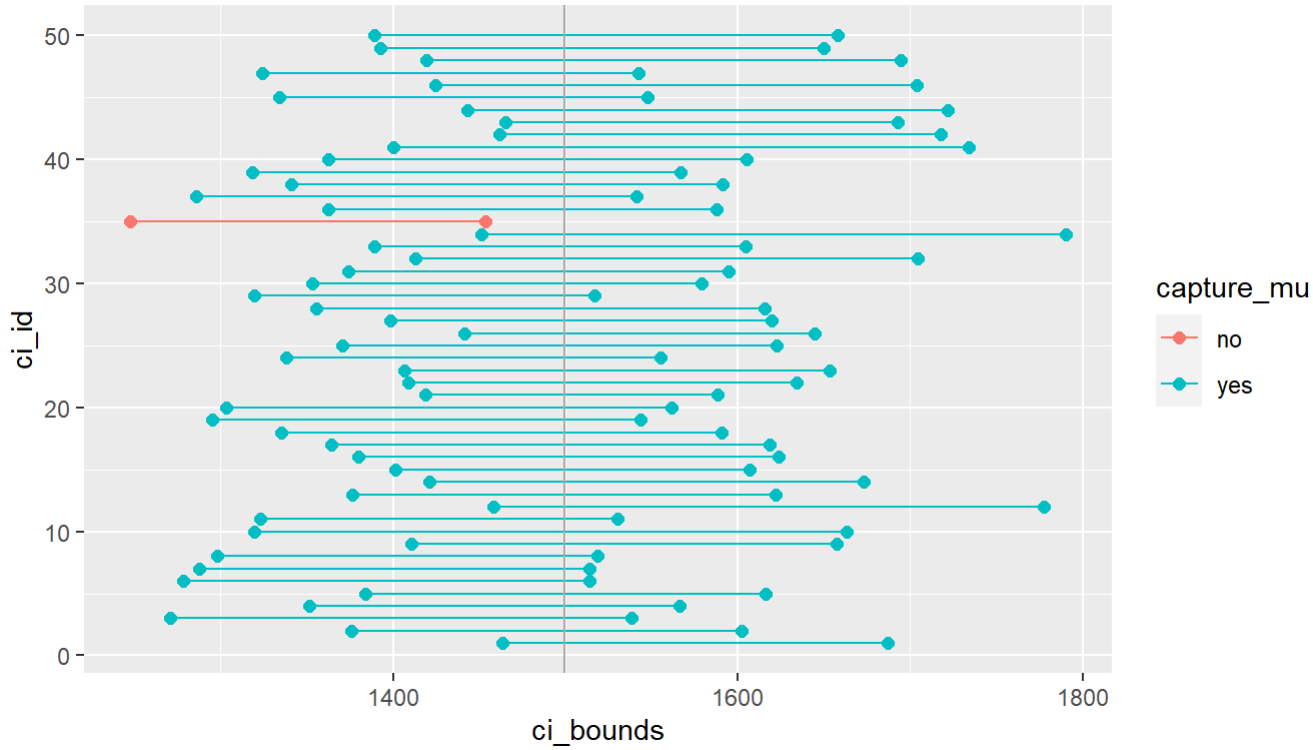
5. What is the appropriate critical value for a 99% confidence level?
    1. 0.01
    2. 0.99
    3. 1.96
    4. 2.33
    5. **2.58**

```
# type your code for the Question 5 here, and Knit
-qnorm(0.5-0.99/2)
```

```
## [1] 2.575829
```

Exercise: Calculate 50 confidence intervals at the 99% confidence level. You do not need to obtain new samples, simply calculate new intervals based on the 95% confidence interval endpoints you had already collected. Plot all intervals and calculate the proportion of intervals that include the true population mean.

```
# type your code for the Exercise here, and Knit

ci <- ames %>% rep_sample_n(size = n, reps = 50, replace = TRUE) %>%
  summarise(lower = mean(area) - 2.58 * (sd(area) / sqrt(n)), upper = mean(area) + 2.58 * (sd(ar
         ea) / sqrt(n)))


ggplot(data = ci_data, aes(x = ci_bounds, y = ci_id, group = ci_id, color = capture_mu)) + geom_
         point(size = 2) + geom_line() + geom_vline(xintercept = params$mu, color = "darkgray")
```



6. We would expect 99% of the intervals to contain the true population mean.

    1. **True**

    2. False

Foundations for inference - Sampling distributions

Complete all **Exercises**

Getting Started

Load packages

In this lab we will explore the data using the `dplyr` package and visualize it using the `ggplot2` package for data visualization. The data can be found in the companion package for this course, `statsr`.

Let's load the packages.

```
library(statsr)
library(dplyr)
library(shiny)
library(ggplot2)
```

The data

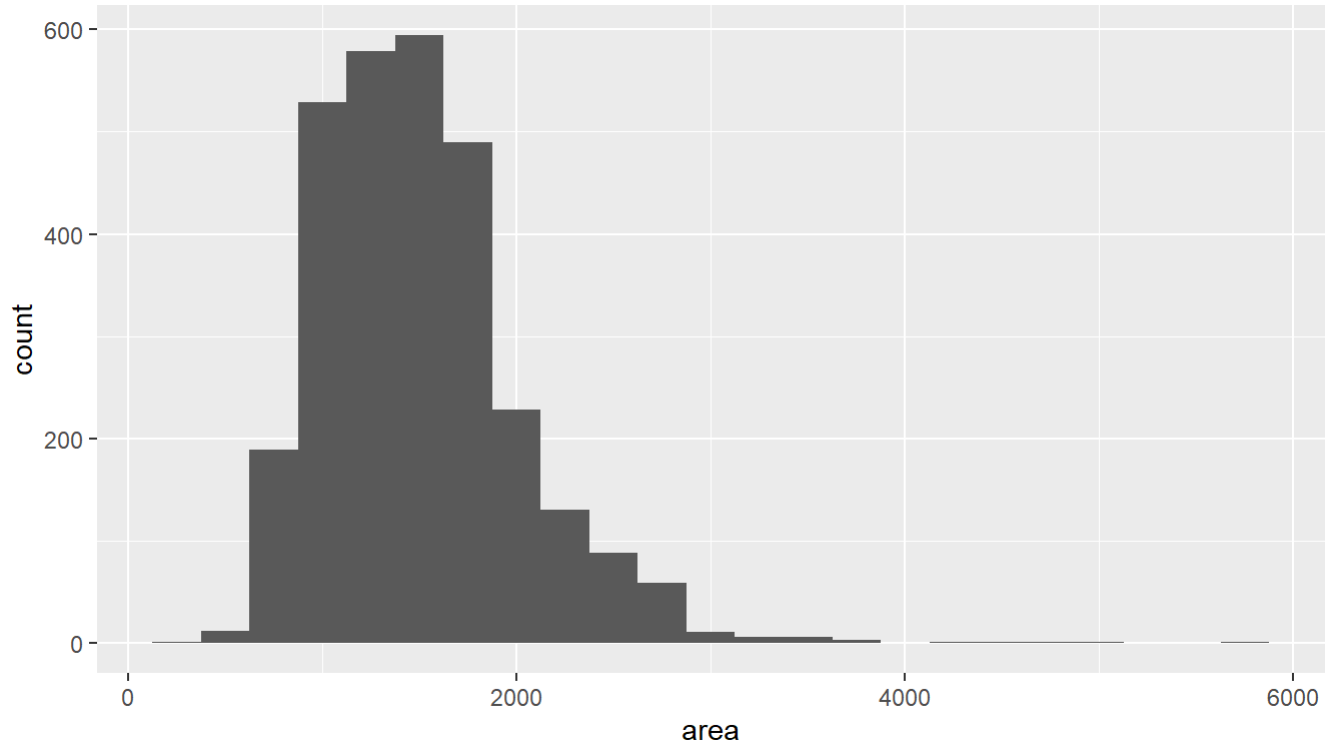
We consider real estate data from the city of Ames, Iowa. The details of every real estate transaction in Ames is recorded by the City Assessor's office. Our particular focus for this lab will be all residential home sales in Ames between 2006 and 2010. This collection represents our population of interest. In this lab we would like to learn about these home sales by taking smaller samples from the full population. Let's load the data.

```
data(ames)
```

We see that there are quite a few variables in the data set, enough to do a very in-depth analysis. For this lab, we'll restrict our attention to just two of the variables: the above ground living area of the house in square feet (`area`) and the sale price (`price`).

We can explore the distribution of areas of homes in the population of home sales visually and with summary statistics. Let's first create a visualization, a histogram:

```
ggplot(data = ames, aes(x = area)) +
  geom_histogram(binwidth = 250)
```



Let's also obtain some summary statistics. Note that we can do this using the `summarise` function. We can calculate as many statistics as we want using this function, and just string along the results. Some of the functions below should be self explanatory (like `mean`, `median`, `sd`, `IQR`, `min`, and `max`). A new function here is the `quantile` function which we can use to calculate values corresponding to specific percentile cutoffs in the distribution. For example `quantile(x, 0.25)` will yield the cutoff value for the 25th percentile (Q1) in the distribution of x. Finding these values are useful for describing the distribution, as we can use them for descriptions like *"the middle 50% of the homes have areas between such and such square feet"*.

```
ames %>%
  summarise(mu = mean(area), pop_med = median(area),
            sigma = sd(area), pop_iqr = IQR(area),
            pop_min = min(area), pop_max = max(area),
            pop_q1 = quantile(area, 0.25),  # first quartile, 25th percentile
            pop_q3 = quantile(area, 0.75))  # third quartile, 75th percentile
```

```
## # A tibble: 1 × 8
##      mu pop_med sigma pop_iqr pop_min pop_max pop_q1 pop_q3
##   <dbl>   <dbl> <dbl>   <dbl>   <int>   <int>  <dbl>  <dbl>
## 1 1500.    1442  506.    617.     334    5642   1126  1743.
```

1. Which of the following is **false**?
    1. The distribution of areas of houses in Ames is unimodal and right-skewed.
    2. **50% of houses in Ames are smaller than 1,499.69 square feet.**
    3. The middle 50% of the houses range between approximately 1,126 square feet and 1,742.7 square feet.
    4. The IQR is approximately 616.7 square feet.

5. The smallest house is 334 square feet and the largest is 5,642 square feet.

The unknown sampling distribution

In this lab we have access to the entire population, but this is rarely the case in real life. Gathering information on an entire population is often extremely costly or impossible. Because of this, we often take a sample of the population and use that to understand the properties of the population.

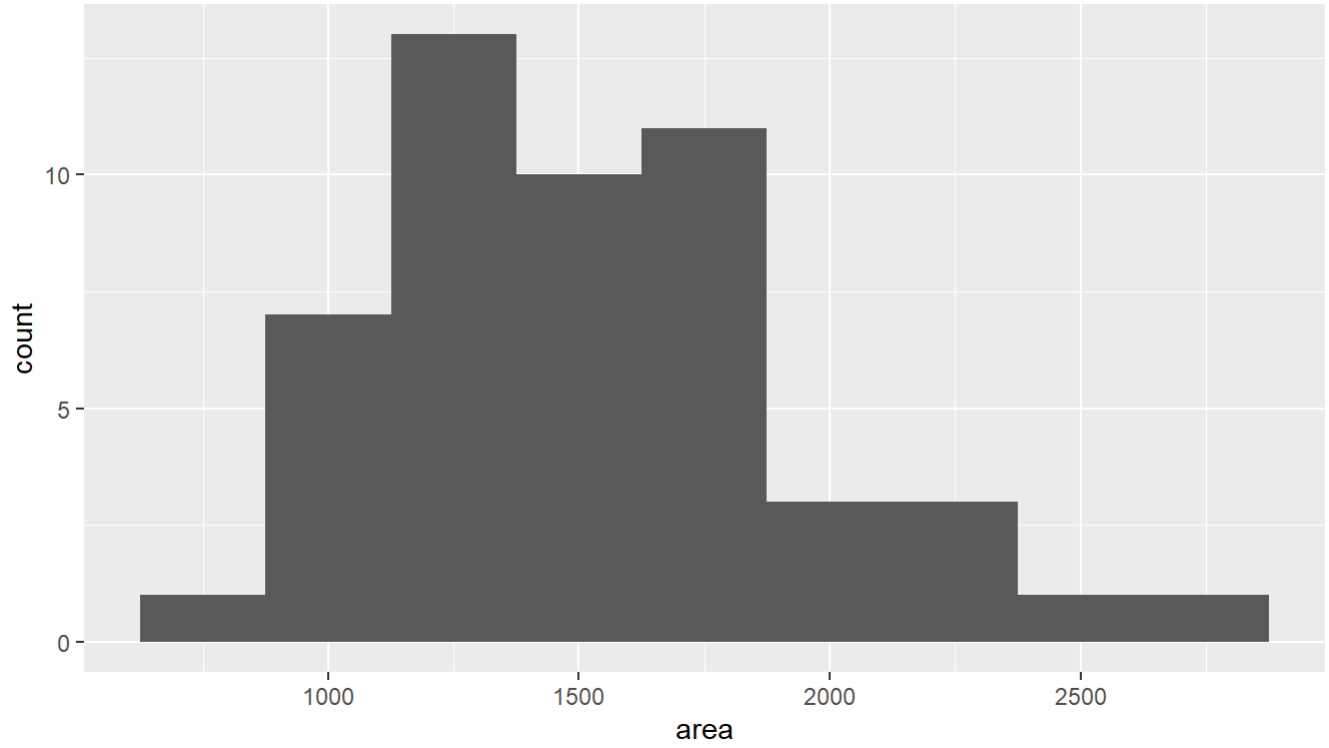
If we were interested in estimating the mean living area in Ames based on a sample, we can use the following command to survey the population.

```
samp1 <- ames %>%
  sample_n(size = 50)
```

This command collects a simple random sample of `size` 50 from the `ames` dataset, which is assigned to `samp1`. This is like going into the City Assessor's database and pulling up the files on 50 random home sales. Working with these 50 files would be considerably simpler than working with all 2930 home sales.

Exercise: Describe the distribution of this sample? How does it compare to the distribution of the population? **Hint:** `sample_n` function takes a random sample of observations (i.e. rows) from the dataset, you can still refer to the variables in the dataset with the same names. Code you used in the previous exercise will also be helpful for visualizing and summarizing the sample, however be careful to not label values `mu` and `sigma` anymore since these are sample statistics, not population parameters. You can customize the labels of any of the statistics to indicate that these come from the sample.

```
# type your code for the Exercise here, and Run Document

ggplot(data = samp1, aes(x = area)) +
  geom_histogram(binwidth = 250)
```



```
samp1 %>%
  summarise(xbar = mean(area), pop_med = median(area), s = sd(area), pop_iqr = IQR(area), pop_mi
        n = min(area), pop_max = max(area), pop_q1 = quantile(area, 0.25), pop_q3 = quantile(ar
        ea, 0.75))
```

```
## # A tibble: 1 × 8
##    xbar pop_med     s pop_iqr pop_min pop_max pop_q1 pop_q3
##   <dbl>   <dbl> <dbl>   <dbl>   <int>   <int>  <dbl>  <dbl>
## 1 1511.   1422.  425.    476.     816    2650   1219  1696.
```

If we're interested in estimating the average living area in homes in Ames using the sample, our best single guess is the sample mean.

```
samp1 %>%
  summarise(x_bar = mean(area))
```

```
## # A tibble: 1 × 1
##   x_bar
##   <dbl>
## 1 1511.
```

Depending on which 50 homes you selected, your estimate could be a bit above or a bit below the true population mean of 1,499.69 square feet. In general, though, the sample mean turns out to be a pretty good estimate of the average living area, and we were able to get it by sampling less than 3% of the population.

2. Suppose we took two more samples, one of size 100 and one of size 1000. Which would you think would provide a more accurate estimate of the population mean?

    1. Sample size of 50.
    2. Sample size of 100.
    3. **Sample size of 1000.**

Let's take one more sample of size 50, and view the mean area in this sample:

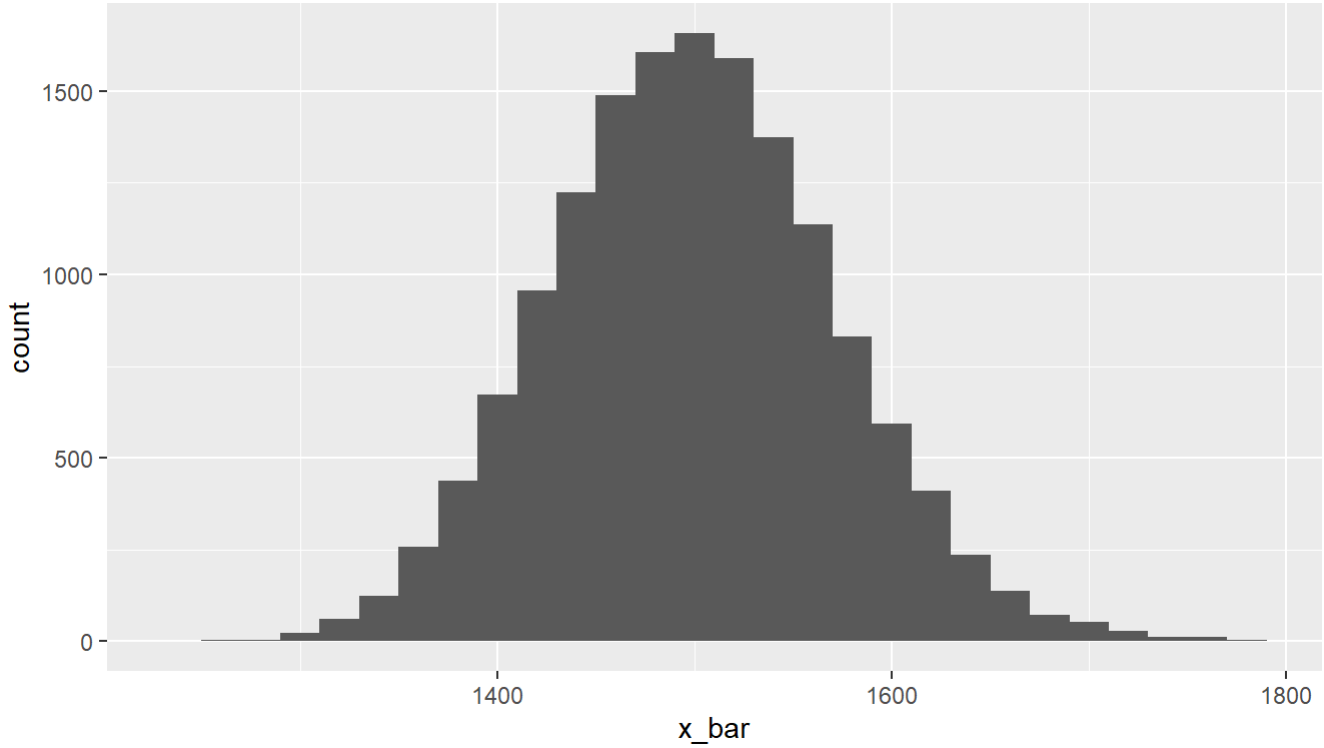
```
ames %>%
  sample_n(size = 50) %>%
  summarise(x_bar = mean(area))
```

```
## # A tibble: 1 × 1
##   x_bar
##   <dbl>
## 1 1492.
```

Not surprisingly, every time we take another random sample, we get a different sample mean. It's useful to get a sense of just how much variability we should expect when estimating the population mean this way. The distribution of sample means, called the *sampling distribution*, can help us understand this variability. In this lab, because we have access to the population, we can build up the sampling distribution for the sample mean by

repeating the above steps many times. Here we will generate 15,000 samples and compute the sample mean of each. Note that we are sampling with replacement, `replace = TRUE` since sampling distributions are constructed with sampling with replacement.

```
sample_means50 <- ames %>%
                    rep_sample_n(size = 50, reps = 15000, replace = TRUE) %>%
                    summarise(x_bar = mean(area))

ggplot(data = sample_means50, aes(x = x_bar)) +
  geom_histogram(binwidth = 20)
```



Here we use R to take 15,000 samples of size 50 from the population, calculate the mean of each sample, and store each result in a vector called `sample_means50` . Next, we review how this set of code works.

Exercise: How many elements are there in `sample_means50` ? Describe the sampling distribution, and be sure to specifically note its center. Make sure to include a plot of the distribution in your answer.

```
# type your code for the Exercise here, and Run Document

nrow(sample_means50)
```

```
## [1] 15000
```

Interlude: Sampling distributions

The idea behind the `rep_sample_n` function is *repetition*. Earlier we took a single sample of size `n` (50) from the population of all houses in Ames. With this new function we are able to repeat this sampling procedure `rep` times in order to build a distribution of a series of sample statistics, which is called the **sampling distribution**.

Note that in practice one rarely gets to build sampling distributions, because we rarely have access to data from the entire population.

Without the `rep_sample_n` function, this would be painful. We would have to manually run the following code 15,000 times

```
ames %>%
  sample_n(size = 50) %>%
  summarise(x_bar = mean(area))
```

as well as store the resulting sample means each time in a separate vector.

Note that for each of the 15,000 times we computed a mean, we did so from a **different** sample!

Exercise: To make sure you understand how sampling distributions are built, and exactly what the `sample_n` and `do` function do, try modifying the code to create a sampling distribution of **25 sample means** from **samples of size 10**, and put them in a data frame named `sample_means_small`. Print the output. How many observations are there in this object called `sample_means_small`? What does each observation represent?

```
# type your code for the Exercise here, and Run Document

sample_means_small <- ames %>% rep_sample_n(size = 10, reps = 25, replace = TRUE) %>%
summarise(x_bar = mean(area))

head(sample_means_small)
```

```
## # A tibble: 6 × 2
##   replicate x_bar
##       <int> <dbl>
## 1         1 1186.
## 2         2 1457.
## 3         3 1466.
## 4         4 1561.
## 5         5 1492.
## 6         6 1693.
```

3. How many elements are there in this object called `sample_means_small`?
    1. 0
    2. 3
    3. **25**
    4. 100
    5. 5,000

```
# type your code for Question 3 here, and Run Document

nrow(sample_means_small)
```

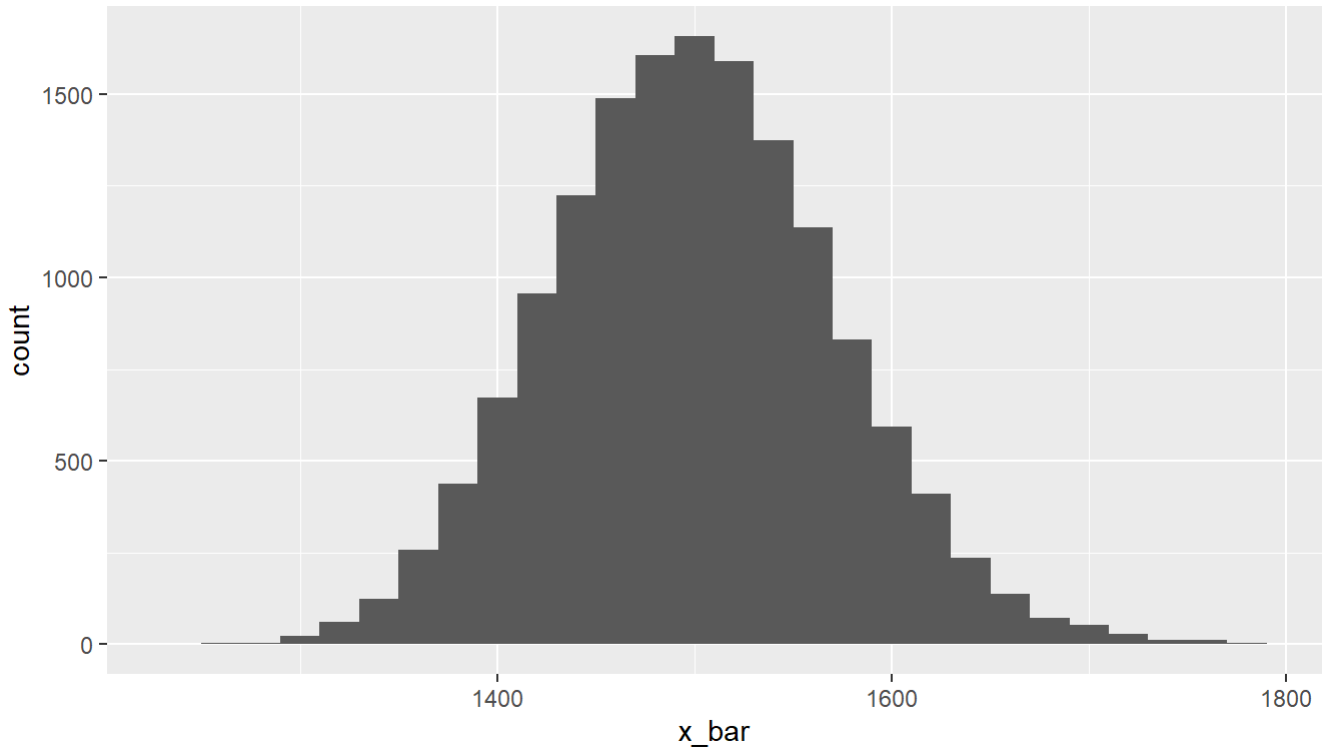
```
## [1] 25
```

4. Which of the following is **true** about the elements in the sampling distributions you created?
    1. **Each element represents a mean square footage from a simple random sample of 10 houses.**
    2. Each element represents the square footage of a house.
    3. Each element represents the true population mean of square footage of houses.

Sample size and the sampling distribution

Mechanics aside, let's return to the reason we used the `rep_sample_n` function: to compute a sampling distribution, specifically, this one.

```
ggplot(data = sample_means50, aes(x = x_bar)) +
  geom_histogram(binwidth = 20)
```



The sampling distribution that we computed tells us much about estimating the average living area in homes in Ames. Because the sample mean is an unbiased estimator, the sampling distribution is centered at the true average living area of the population, and the spread of the distribution indicates how much variability is induced by sampling only 50 home sales.

In the remainder of this section we will work on getting a sense of the effect that sample size has on our sampling distribution.

Exercise: Use the app below to create sampling distributions of means of `area` s from samples of size 10, 50, and 100. Use 5,000 simulations. What does each observation in the sampling distribution represent? How does the mean, standard error, and shape of the sampling distribution change as the sample size increases? How (if at all) do these values change if you increase the number of simulations?

Variable:

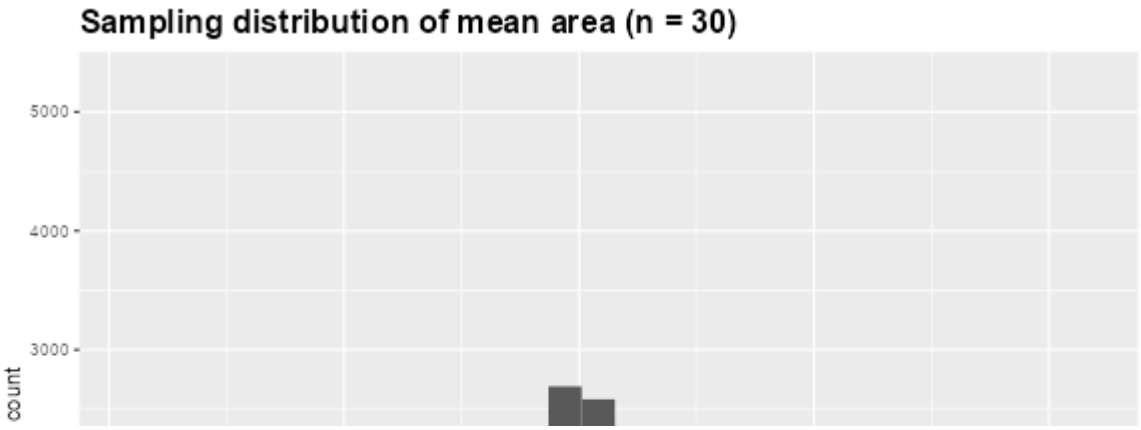
area

Sample size:

30

Number of samples:

15000



5. It makes intuitive sense that as the sample size increases, the center of the sampling distribution becomes a more reliable estimate for the true population mean. Also as the sample size increases, the variability of the sampling distribution _____.
    1. **decreases**
    2. increases
    3. stays the same

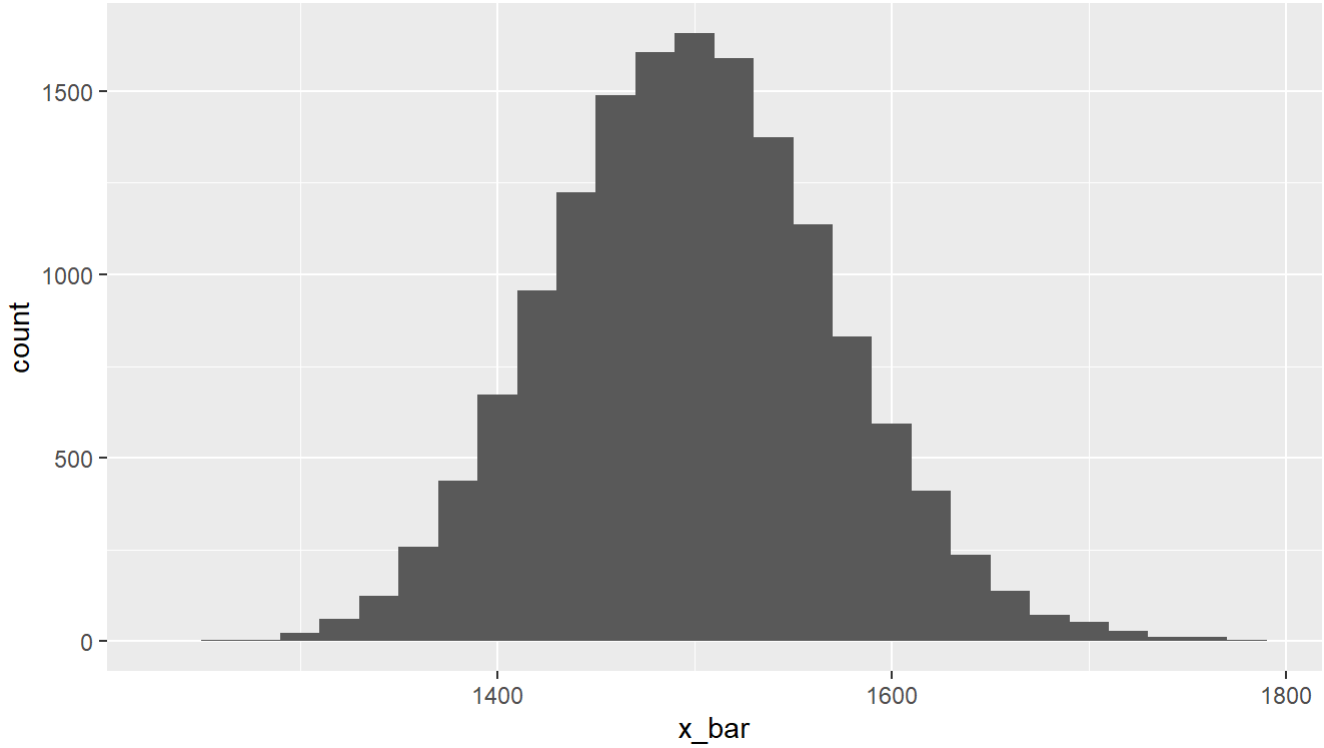
Exercise: Take a random sample of size 50 from `price`. Using this sample, what is your best point estimate of the population mean?

```
# type your code for this Exercise here, and Run Document
ames %>%
  sample_n(size = 50) %>%
  summarise(x_bar = mean(price))
```

```
## # A tibble: 1 × 1
##     x_bar
##     <dbl>
## 1 185057.
```

Exercise: Since you have access to the population, simulate the sampling distribution for $\bar{x}_{price}$ by taking 5000 samples from the population of size 50 and computing 5000 sample means. Store these means in a vector called `sample_means50`. Plot the data, then describe the shape of this sampling distribution. Based on this sampling distribution, what would you guess the mean home price of the population to be?

```
# type your code for this Exercise here, and Run Document
sample_m <- ames %>% rep_sample_n(size = 50, reps = 15000, replace = TRUE) %>%
                    summarise(x_bar = mean(price))

ggplot(data = sample_means50, aes(x = x_bar)) +
  geom_histogram(binwidth = 20)
```

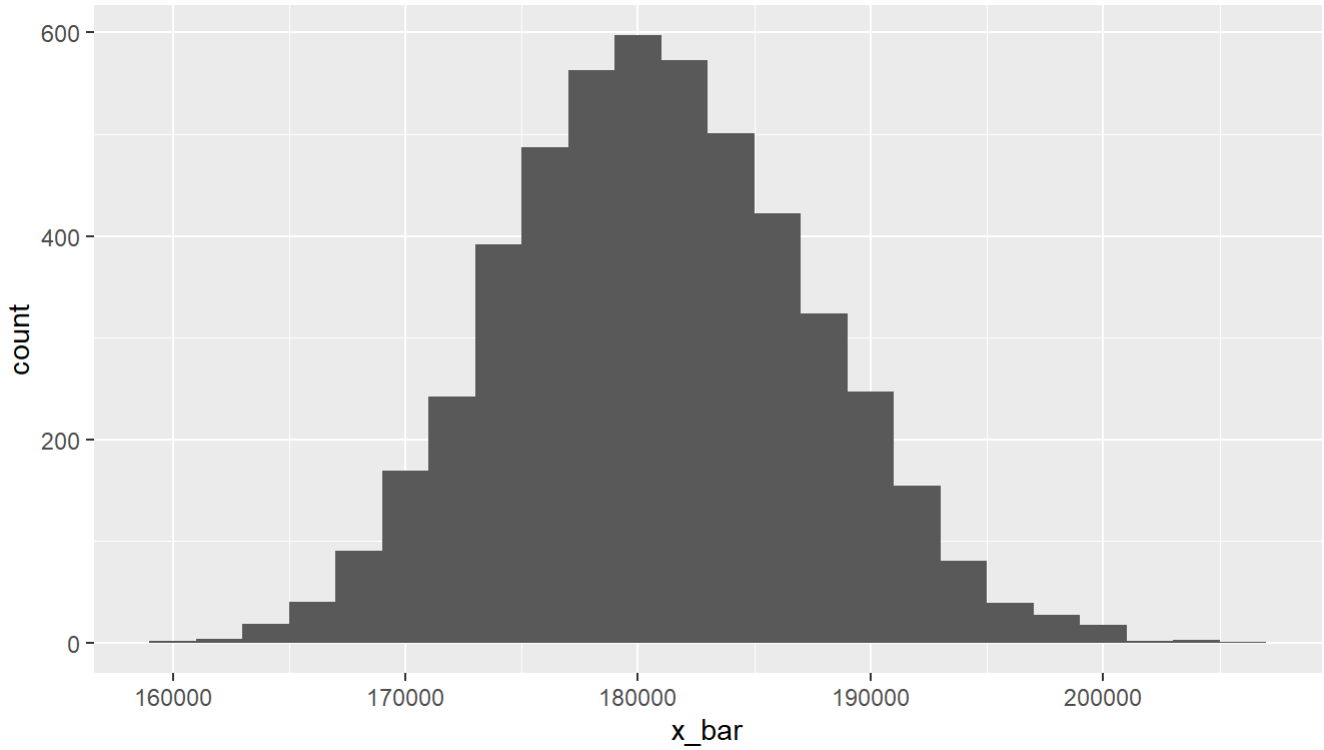



```
sample_means50 %>%
  summarise(mean_sampling_50 = mean(x_bar),
            standard_error_50 = sd(x_bar))
```

```
## # A tibble: 1 × 2
##   mean_sampling_50 standard_error_50
##              <dbl>             <dbl>
## 1            1499.              71.7
```

Exercise: Change your sample size from 50 to 150, then compute the sampling distribution using the same method as above, and store these means in a new vector called `sample_means150`. Describe the shape of this sampling distribution, and compare it to the sampling distribution for a sample size of 50. Based on this sampling distribution, what would you guess to be the mean sale price of homes in Ames?

```
# type your code for this Exercise here, and Run Document
sample_means150 <- ames %>%
                    rep_sample_n(size = 150, reps = 5000, replace = TRUE) %>%
                    summarise(x_bar = mean(price))

ggplot(data = sample_means150, aes(x = x_bar)) +
  geom_histogram(binwidth = 2000)
```




```
sample_means150 %>%
  summarise(mean_sampling_150 = mean(x_bar),
            standard_error_150 = sd(x_bar))
```

```
## # A tibble: 1 × 2
##   mean_sampling_150 standard_error_150
##               <dbl>              <dbl>
## 1           180855.              6602.
```

So far, we have only focused on estimating the mean living area in homes in Ames. Now you'll try to estimate the mean home price.

Note that while you might be able to answer some of these questions using the app you are expected to write the required code and produce the necessary plots and summary statistics. You are welcomed to use the app for exploration.

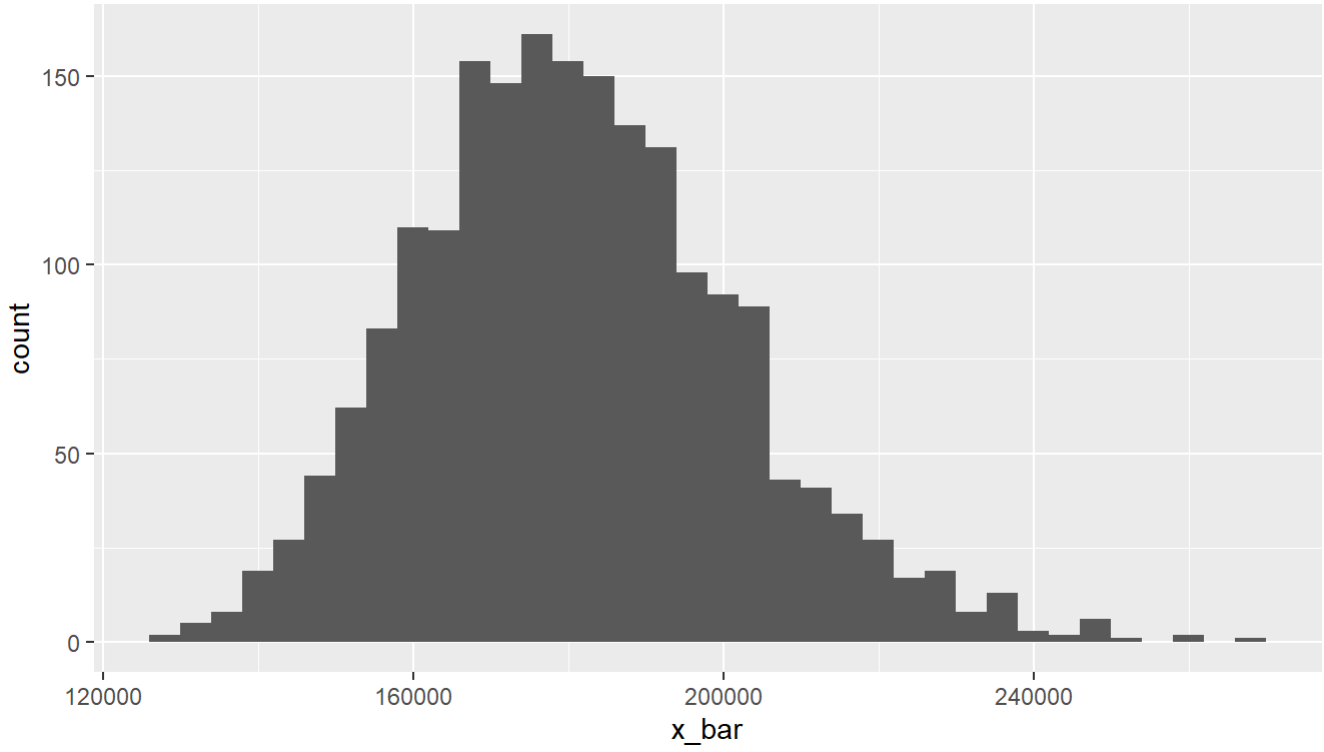
Exercise: Take a sample of size 15 from the population and calculate the mean `price` of the homes in this sample. Using this sample, what is your best point estimate of the population mean of prices of homes?

```
# type your code for this Exercise here, and Run Document
ames %>%
  sample_n(size = 15) %>%
  summarise(x_bar = mean(price))
```

```
## # A tibble: 1 × 1
##    x_bar
##    <dbl>
## 1 171060
```

Exercise: Since you have access to the population, simulate the sampling distribution for $\bar{x}_{price}$ by taking 2000 samples from the population of size 15 and computing 2000 sample means. Store these means in a vector called `sample_means15`. Plot the data, then describe the shape of this sampling distribution. Based on this sampling distribution, what would you guess the mean home price of the population to be? Finally, calculate and report the population mean.

```
# type your code for this Exercise here, and Run Document
sample_means15 <- ames %>%
                    rep_sample_n(size = 15, reps = 2000, replace = TRUE) %>%
                    summarise(x_bar = mean(price))

ggplot(data = sample_means15, aes(x = x_bar)) +
  geom_histogram(binwidth = 4000)
```

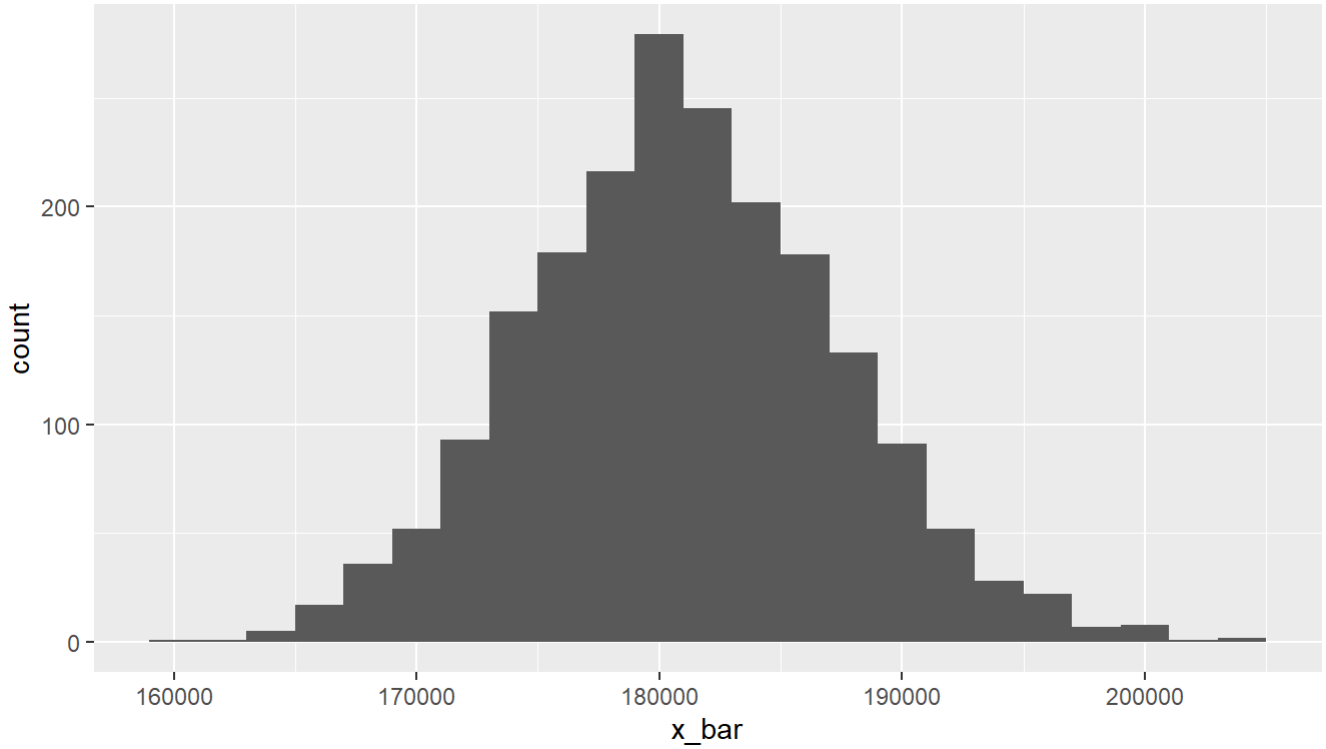



```
sample_means15 %>%
  summarise(mean_sampling_15 = mean(x_bar),
            standard_error_15 = sd(x_bar))
```

```
## # A tibble: 1 × 2
##   mean_sampling_15 standard_error_15
##              <dbl>             <dbl>
## 1          181143.            20681.
```

Exercise: Change your sample size from 15 to 150, then compute the sampling distribution using the same method as above, and store these means in a new vector called `sample_means150`. Describe the shape of this sampling distribution, and compare it to the sampling distribution for a sample size of 15. Based on this sampling distribution, what would you guess to be the mean sale price of homes in Ames?

```
# type your code for this Exercise here, and Run Document
sample_means150 <- ames %>%
                   rep_sample_n(size = 150, reps = 2000, replace = TRUE) %>%
                   summarise(x_bar = mean(price))

ggplot(data = sample_means150, aes(x = x_bar)) +
  geom_histogram(binwidth = 2000)
```




```
sample_means150 %>%
  summarise(mean_sampling_150 = mean(x_bar),
            standard_error_150 = sd(x_bar))
```

```
## # A tibble: 1 × 2
##   mean_sampling_150 standard_error_150
##               <dbl>              <dbl>
## 1           180976.              6424.
```

6. Which of the following is false?

1. **The variability of the sampling distribution with the smaller sample size (`sample_means50`) is smaller than the variability of the sampling distribution with the larger sample size (`sample_means150`).**
2. The means for the two sampling distribtuions are roughly similar.
3. Both sampling distributions are symmetric.

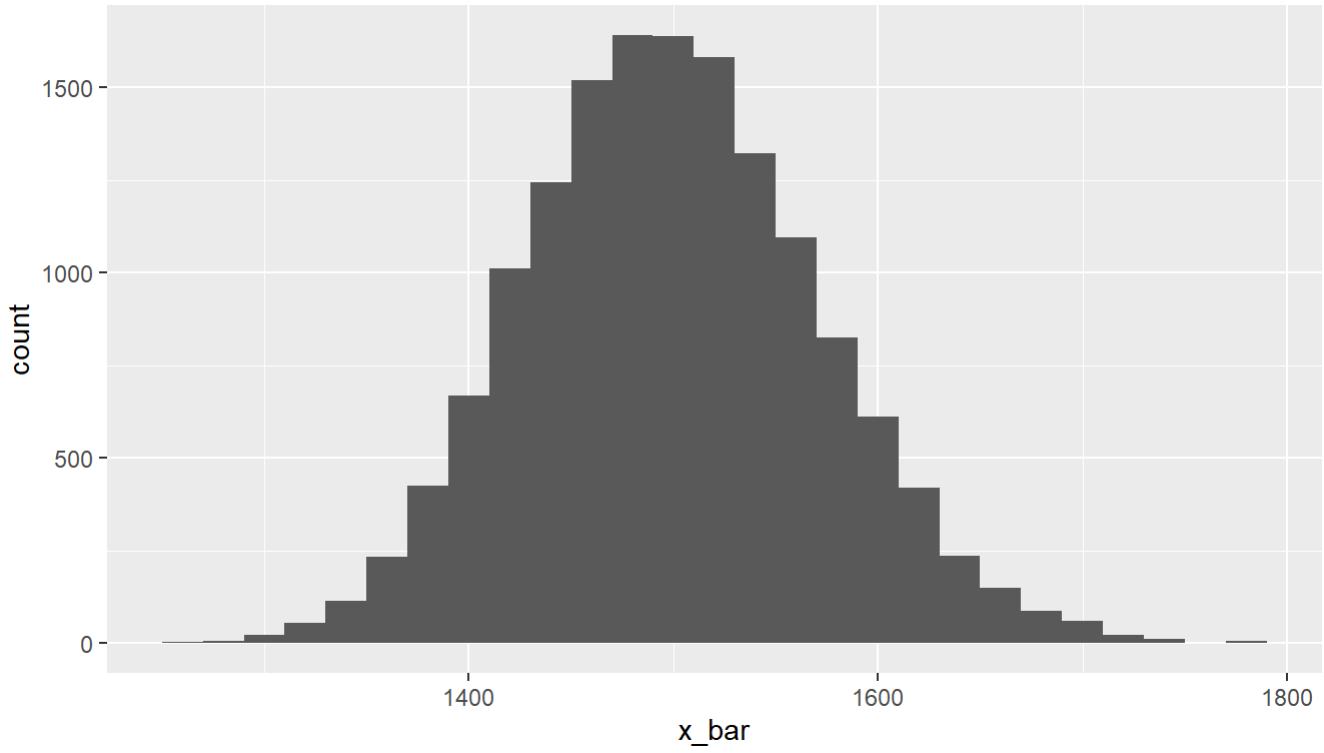
```
# type your code for Question 6 here, and Run Document
sample_means50 <- ames %>%
     rep_sample_n(size = 50, reps = 15000, replace = TRUE) %>%
     summarise(x_bar = mean(area))

sample_means150 <- ames %>%
     rep_sample_n(size = 150, reps = 15000, replace = TRUE) %>%
     summarise(x_bar = mean(area))

sd(sample_means50$x_bar)
```

```
## [1] 71.56386
```

```
mean(sample_means50$x_bar)
```

```
## [1] 1499.175
```

```
ggplot(data = sample_means50, aes(x = x_bar)) +
  geom_histogram(binwidth = 20)
```



```
ggplot(data = sample_means150, aes(x = x_bar)) +
  geom_histogram(binwidth = 10)
```