Notes and Exercises Chapter 1 – Intro to Data
Math 217
Saidi

Statistics is the science of making decisions by analyzing data, in the face of variability and uncertainty.

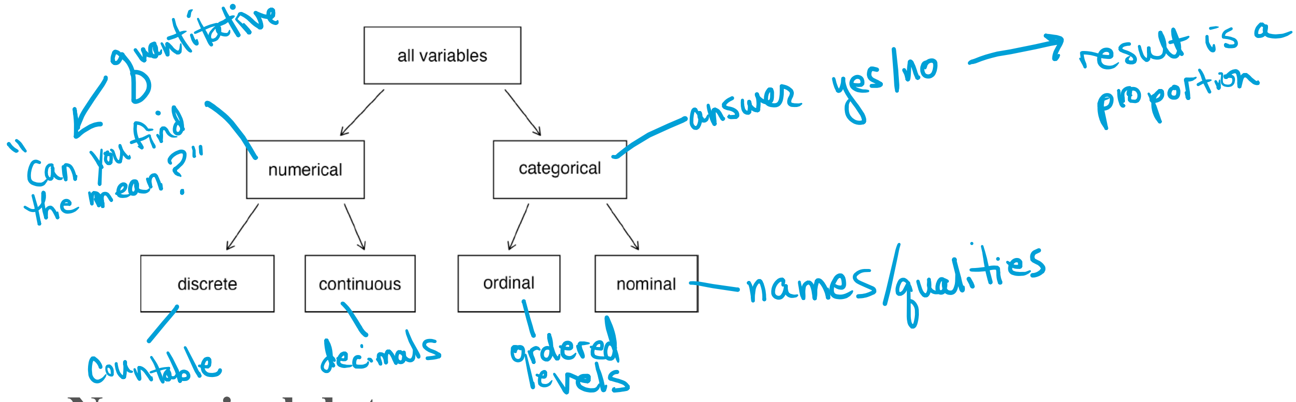
Introduction to data

Scientists seek to answer questions using rigorous methods and careful observations. These observations form the backbone of a statistical investigation and are called data. Statistics is the study of how best to collect, analyze, and draw conclusions from data. It is helpful to put statistics in the context of a general process of investigation:

- **Step 1**: Identify a question or problem.
- **Step 2**: Collect relevant data on the topic.
- **Step 3**: Analyze the data.
- **Step 4**: Form a conclusion.

1.2.2 Types of Variables

- A ***variable*** is any characteristic that is recorded for each case or subject.
- ***Inevitable variability*** – researchers must distinguish between the true "signal" and the "noise".



Numerical data

Numerical variables can be further categorized as **continuous or discrete**.

- **Continuous numerical** variables are usually measured, such as height. These variables can take on an infinite number of values within a given range.

- **Discrete numerical** variables are those that take on one of a specific set of numeric values where we are able to count or enumerate all of the possibilities. One example of a discrete variable is number of pets in a household. In general, count data are an example of discrete variables.

When determining whether a numerical variable is continuous or discrete, it is important to think about the nature of the variable and not just the observed value, as rounding of continuous variables can make them appear to be discrete. For example, height is a continuous variable, however we tend to report our height rounded to the nearest unit of measure, like inches or centimeters.

Categorical data

Categorical variables can be further categorized as **factors or ordinal values.**

- **Factor** variables can take on one of a limited, and usually fixed, number of possible values, assigning each individual or other unit of observation to a particular group or nominal category on the basis of some qualitative property

- **Ordinal** variables have an order to their factor levels.

Packages

Packages are the fundamental units of reproducible R code. They include reusable functions, the documentation that describes how to use them, and sample data. In this lesson we will make use of two packages:

- **`tidyverse`**: Tidyverse is a collection of R packages for data science that adhere to a common philosophy of data and R programming syntax, and are designed to work together naturally. You can learn more about tidyverse here.
- **`openintro`**: The openintro package contains datasets used in openintro resources. You can find out more about the package here.

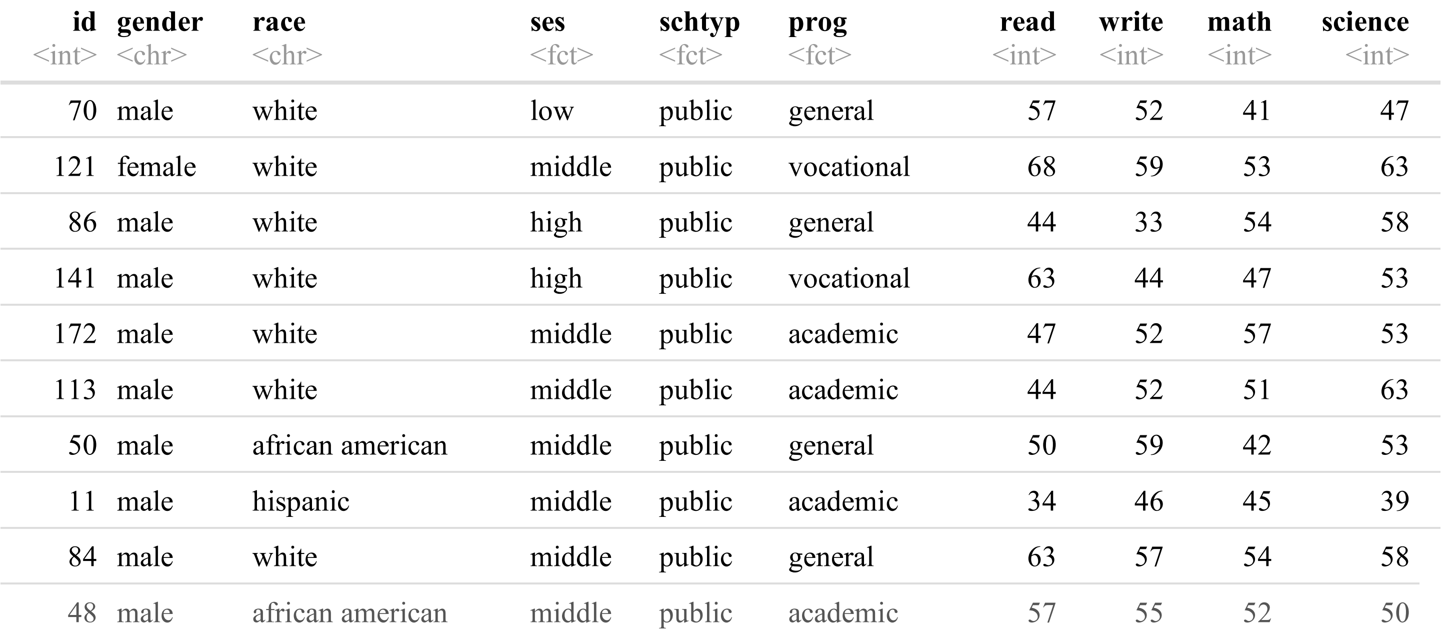
Once we have installed the packages, we use the `library()` function to load packages into R.
Let's load these two packages to be used in the remainder of this lesson.

```
library(tidyverse)
library(openintro)
```

Example1

Explore the **High School and Beyond Survey Data** from the National Center of Education Statistics.

Below is a preview of the dataset. View the variables and the row values in the dataset. You can access the **hsb2** dataset here.



id <int>	gender <chr>	race <chr>	ses <fct>	schtyp <fct>	prog <fct>	read <int>	write <int>	math <int>	science <int>
70	male	white	low	public	general	57	52	41	47
121	female	white	middle	public	vocational	68	59	53	63
86	male	white	high	public	general	44	33	54	58
141	male	white	high	public	vocational	63	44	47	53
172	male	white	middle	public	academic	47	52	57	53
113	male	white	middle	public	academic	44	52	51	63
50	male	african american	middle	public	general	50	59	42	53
11	male	hispanic	middle	public	academic	34	46	45	39
84	male	white	middle	public	general	63	57	54	58
48	male	african american	middle	public	academic	57	55	52	50

Later, you will learn to load a CSV file that has been saved on your computer into R Studio. For now, the dataset is contained in the **`openintro`** library, so you will not need to load the dataset at this time.

Pipes in R

The history of this operator in R starts in 2012 when Hadley Wickham started the dplyr package on GitHub, which is based off of F# (pronounced F Sharp, as in Visual F# Programming Language, which is an open source, cross platform compiler, which can generate JavaScript and graphics processing unit (GPU) code.

The pipe operator: `|>  (also %>%)`

The **pipe operator**, which **is percent greater than percent**, tells R to pass the object that comes before it into the first argument of the function that comes after it. Mathematically, **x pipe f(y)** becomes *f(x, y)*, since x is piped into the first argument of the function f().

```
x |> f(y)
f(x, y)
```

What is a Pipe?

It takes the output of one statement and makes it the input of the next statement. When describing it, you can think of it as a "THEN".

This is one of the most powerful things about the Tidyverse. In fact, having a standardized chain of processing actions is called "a pipeline".

Example2

Here is an example of using piping in the **hsb2** dataset:

- use the `filter()` function to filter the data to only include public school students.

```
hsb2_public <- hsb2 |>
       filter(schtyp == "public")
```

We can read the above code as: “take the hsb2 data frame and **pipe it** into the filter() function. Next, filter() the data **for cases where school type is equal to public**. Then, **assign the resulting data frame** to a new data frame called **hsb2 underscore public**.”

We should take note of two pieces of R syntax: *the double equal sign* (==) and quotations (“ “). In R, == is a logical test for *“is equal to”*. R uses this logical test to search for observations (rows) in the data frame where school type is equal to public, and returns a data frame where this comparison is TRUE for every row.

In R, variables that are categorical use characters (rather than numbers) for values. To indicate to R that you want your logical test to compare the values of a categorical variable to a specific level of that variable, you need to surround the name of the level in quotations (e.g. schtyp == "public"). The quotations tell R that the value of the variable is a character, not a number. If you forget to use quotations, R will give you an error message!

The first variable is id, which is an identifier variable for the student.

```
## Rows: 200

## Columns: 1

## $ id <int> 70, 121, 86, 141, 172, 113, 50, 11, 84, 48, 75, 60, 95, 104, 38, 11…
```

Strictly speaking, this is a categorical variable, though the labeling of this variable is likely not that useful since we would not use this variable in an analysis of relationships between the variables in the dataset. You can think of this variable as being an anonymized version to having the names of the students in the dataset.

The next variable is gender, a categorical variable, with levels "male" and "female". It should be noted that the language of government surveys, such as High School and Beyond, is slow to change. So with these types of data, you will continue to see variables mislabeled as “gender” when they in fact measure the biological sex (male, female) of the participant.

There is no inherent ordering to the levels of this variable, no matter what anyone tells you! So, this is just a categorical variable. The same is true for the race variable, which has levels of "white", "african american", "hispanic", and "asian".

```
## Rows: 200
## Columns: 2
## $ race   <chr> "white", "white", "white", "white", "white", "white", "african …
## $ gender <chr> "male", "female", "male", "male", "male", "male", "male", "male…
```

Socio-economic status, on the other hand, has three levels "low", "middle", and "high" that have an inherent ordering, hence this variable is an *ordinal* categorical variable.

```
## Rows: 200
## Columns: 1
## $ ses <fct> low, middle, high, high, middle, middle, middle, middle, middle, m
…
```

School type and program are also both categorical variables, with no inherent ordering to their levels.

```
## Rows: 200
## Columns: 2
## $ schtyp <fct> public, public, public, public, public, public, public, public,
…
## $ prog   <fct> general, vocational, general, vocational, academic, academic, g
…
```

The remaining variables are scores that these students received in reading, writing, math, science, and social studies tests. Since these scores are all whole numbers, and assuming that it is not possible to obtain a non-whole number score on these tests, these variables are discrete numerical.

Discretize variables

A common way of creating a new variable from an existing variable is discretizing, that is converting a numerical variable to a categorical variable based on certain criteria.

Example3

For example, suppose we are not interested in the actual reading score of students, but instead whether their reading score is below average or at or above average. First, we need to calculate the average reading score with the mean() function. This will give us the mean value, 52.23.

```
# Calculate average reading score and show the value

mean(hsb2$read)
```

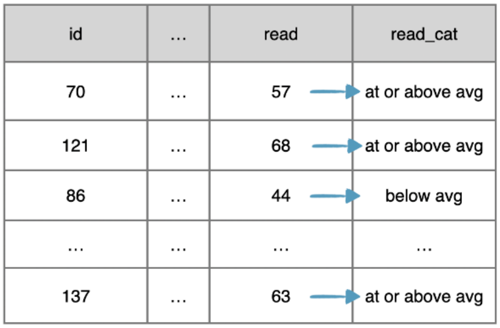
However, in order to be able to refer back to this value later on, we might want to store it as an object that we can refer to by name. So instead of just printing the result, let's save it as a new object called **avg underscore read**.

```
# Calculate average reading score and store as avg_read

avg_read <- mean(hsb2$read)
```

Most often we want to do both; see the value and also store it for later use. The approach we used here, running the `mean()` function twice, is redundant. A less redundant way to accomplish this task is to wrap your assignment code in parentheses so that R will both assign the average value of reading test scores to avg_read, and print out the value assigned to avg_read.

Next, in order to create the two groups of interest, we need to determine whether each student is either (1) below or (2) at or above average. For example, a reading score of 57 is above average, so is 68, but 44 is below. Obviously, going through each record like this would be tedious and error prone, so let's explore another option!

New variable: read_cat



id	…	read	read_cat
70	…	57	at or above avg
121	…	68	at or above avg
86	…	44	below avg
…	…	…	…
137	…	63	at or above avg

Instead we can create a new variable, named read_cat, with the mutate() function and the helpful if_else() function.

```
hsb2_new <- hsb2 %>%
    mutate(read_cat = if_else(read < avg_read, "below average", "at or above
average"))

hsb2
```

First, start with the data frame, hsb2, and pipe it into the mutate() function. We use the mutate() function to create a new variable called read_cat. Note that we are using a new variable name here, so that we do not overwrite the existing reading score variable, called read.

The values of this new variable are simple: if the reading score of the student is below the average reading score, the variable will have the label "below average", otherwise, the label will be "at or above average".

This discretization can be accomplished using the if_else() function in R:

- The first argument of the function is the logical test we wish to perform: read < avg_read.
- The second argument is what we want the function to do if the result of the logical test is TRUE, in other words, if the student's score is below the average score: "below average".
- The third argument is what we want the function to do if the result of the logical test is FALSE, in other words, if the student's score is above the average score: "at or above average".

Example4

TWO WAY CATEGORICAL TABLES

Explore variability in responses with two different experiments

1. A vaccine for anthrax was used on 24 sheep and 24 other sheep were unvaccinated as a control. All sheep were inoculated with the live culture of anthrax bacillus.
Table of response:

	Treatment	
Response	**Vaccinated**	**Not vaccinated**
Died	0	24
Survived	24	0
Total	24	24
Percent survival	100%	0%

Unequivocal positive response to being vaccinated to prevent death due to anthrax.

2. Mice with naturally high incidence of liver tumors were split into two treatment groups – one group of mice were maintained germ-free, and the other group were exposed to intestinal bacteria.
Table of response:

	Treatment	
Response	**E. coli**	**Germ free**
Liver tumors	8	19
No liver tumors	5	30
Total	13	49
Percent tumors	62%	39%

Variable response to being in germ-free environment to prevent liver tumors.

The question we need to ask in Experiment #2 is: is the observed difference in percentages (62% to 39%) due to a true effect (the signal) or chance variation (the noise)?

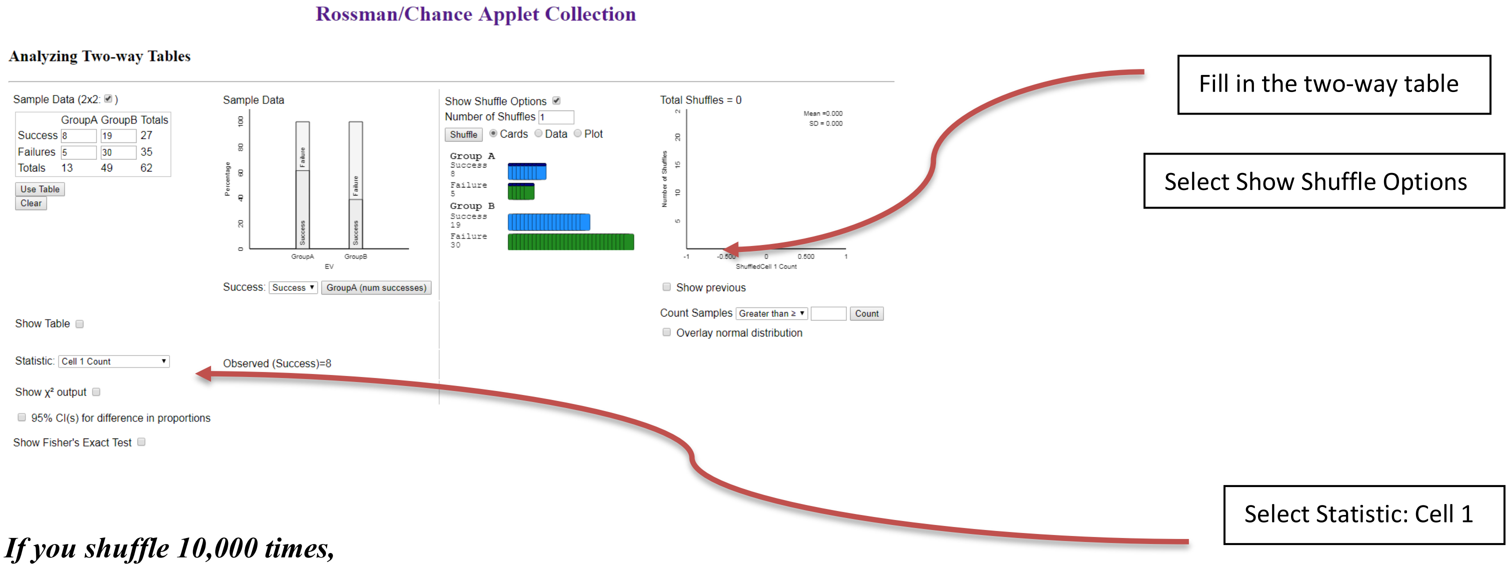
TRY THIS:
We could test to see how likely this response is by creating a "randomization distribution" (repeating a simulation of this situation something like 10,000 times) to see how likely the mice remain tumor-free. See page 2. We will address this more in future chapters.

Randomization Distribution Simulation

Instead of physically shuffling cards, we can simulate this randomization distribution:
http://www.rossmanchance.com/applets/

I have created a video to help demonstrate how to do this:
https://youtu.be/_bP7nraTEhA

Under **“Statistical Inference”**, select **two-way table.** Then fill in the information from example #2 above with liver tumors and germ-free or E. coli environments:



Rossman/Chance Applet Collection

Analyzing Two-way Tables

Sample Data (2x2: ☑)

	GroupA	GroupB	Totals
Success	8	19	27
Failures	5	30	35
Totals	13	49	62

Use Table
Clear

Success: Success ▾ GroupA (num successes)

Show Table □

Statistic: Cell 1 Count ▾

Observed (Success)=8

Show χ² output □

□ 95% CI(s) for difference in proportions

Show Fisher's Exact Test □

Show Shuffle Options ☑
Number of Shuffles 1
Shuffle ● Cards ○ Data ○ Plot

Total Shuffles = 0

Mean =0.000
SD = 0.000

□ Show previous

Count Samples Greater than ≥ ▾ Count

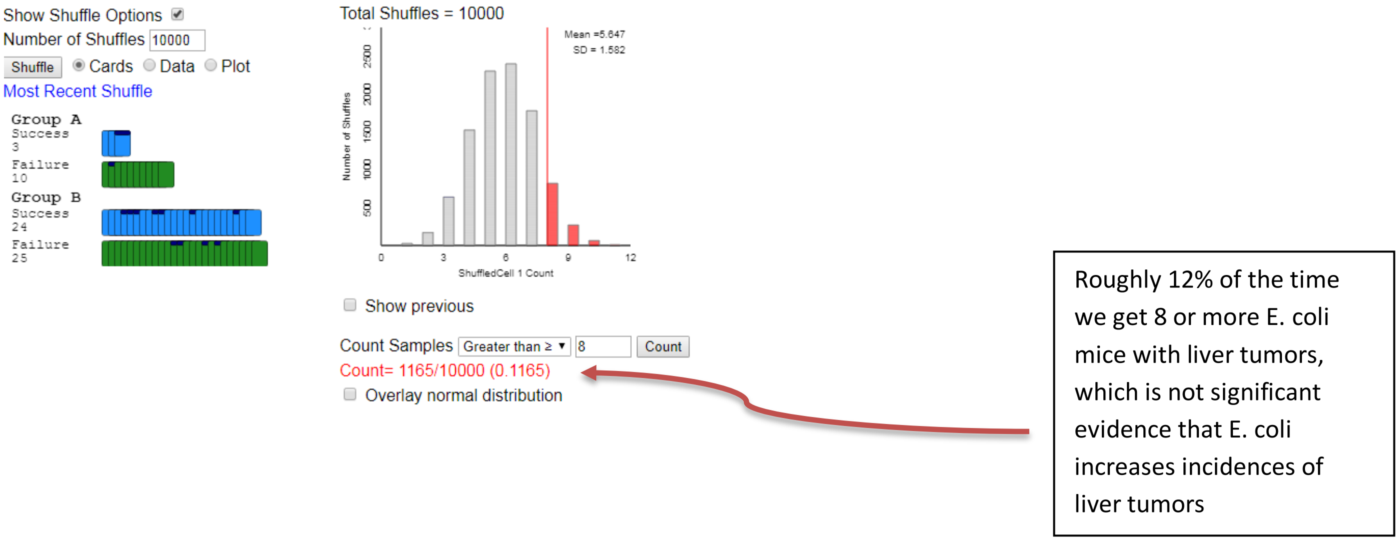
□ Overlay normal distribution

Fill in the two-way table

Select Show Shuffle Options

Select Statistic: Cell 1

If you shuffle 10,000 times,



Show Shuffle Options ☑
Number of Shuffles 10000
Shuffle ● Cards ○ Data ○ Plot
Most Recent Shuffle

Total Shuffles = 10000

Mean =5.647
SD = 1.582

□ Show previous

Count Samples Greater than ≥ ▾ 8 Count
Count= 1165/10000 (0.1165)
□ Overlay normal distribution

Roughly 12% of the time we get 8 or more E. coli mice with liver tumors, which is not significant evidence that E. coli increases incidences of liver tumors

1.2 Data Basics Using loan50 dataset (live coding)

Access the dataset here: https://www.openintro.org/data/index.php?data=loan50

Loan data from Lending Club

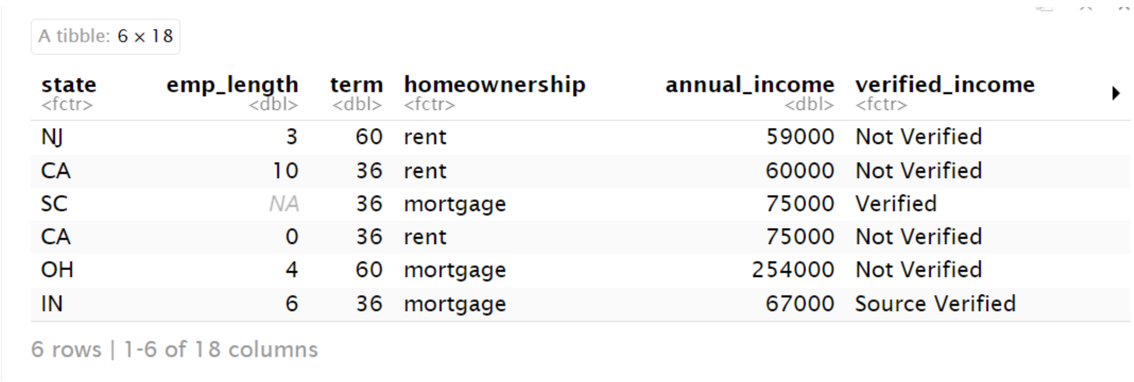
This data set represents 50 loans made through the Lending Club platform, which is a platform that allows individuals to lend to other individuals. Of course, not all loans are created equal. Someone who is a essentially a sure bet to pay back a loan will have an easier time getting a loan with a low interest rate than someone who appears to be riskier. And for people who are very risky? They may not even get a loan offer, or they may not have accepted the loan offer due to a high interest rate. It is important to keep that last part in mind, since this data set only represents loans actually made, i.e. do not mistake this data for loan applications!

Scan the 1st 6 rows of the loan50 dataset

When you want to work with data in R, a good first step is to take a peek at what the data look like. The `head()` function is one good way of doing this.

How many observations and variables are there? ______________________________

```
head(loan50)
```



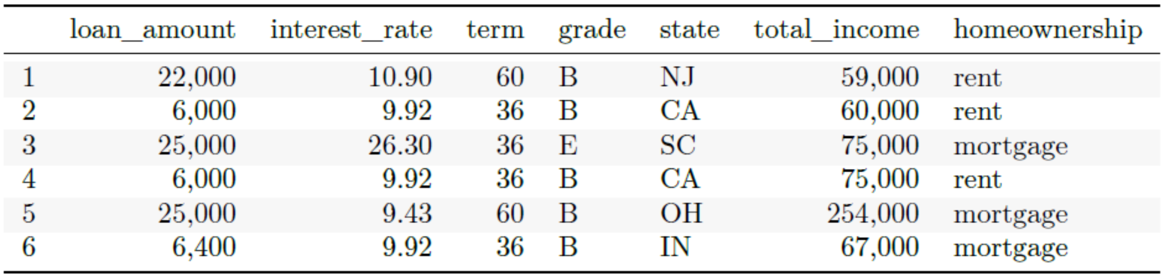
```
A tibble: 6 x 18

state    emp_length  term  homeownership       annual_income  verified_income
<fctr>        <dbl>  <dbl> <fctr>                      <dbl>  <fctr>
NJ                3    60  rent                        59000  Not Verified
CA               10    36  rent                        60000  Not Verified
SC               NA    36  mortgage                    75000  Verified
CA                0    36  rent                        75000  Not Verified
OH                4    60  mortgage                   254000  Not Verified
IN                6    36  mortgage                    67000  Source Verified

6 rows | 1-6 of 18 columns
```

What is a ***tibble****?* ______________________________

Go to live coding: View EACH variable (column header) and try to determine if the variable is numerical or categorical.

Each row in the table represents a single loan. The formal name for a row is a case or observational unit. The columns represent characteristics of each loan, where each column is referred to as a variable. For example, the first row represents a loan of $22,000 with an interest rate of 10.90%, where the borrower is based in New Jersey (NJ) and has an income of $59,000.

Table 1.3: Six observations from the `loan50` dataset



	loan_amount	interest_rate	term	grade	state	total_income	homeownership
1	22,000	10.90	60	B	NJ	59,000	rent
2	6,000	9.92	36	B	CA	60,000	rent
3	25,000	26.30	36	E	SC	75,000	mortgage
4	6,000	9.92	36	B	CA	75,000	rent
5	25,000	9.43	60	B	OH	254,000	mortgage
6	6,400	9.92	36	B	IN	67,000	mortgage

Come up with a few questions about the loan50 dataset you see. Write them down.

Live Coding

load the libraries

```
library(tidyverse)
library(openintro)
```

view high school and beyond data class

```
data("loan50")
```

Scan the first 6 rows of the data

```
head(loan50)

# A tibble: 6 × 18
  state emp_length  term homeownership annual_income verified_income
  <fct>      <dbl> <dbl> <fct>                 <dbl> <fct>
1 NJ             3    60 rent                  59000 Not Verified
2 CA            10    36 rent                  60000 Not Verified
3 SC            NA    36 mortgage              75000 Verified
4 CA             0    36 rent                  75000 Not Verified
5 OH             4    60 mortgage             254000 Not Verified
6 IN             6    36 mortgage              67000 Source Verified
#
```

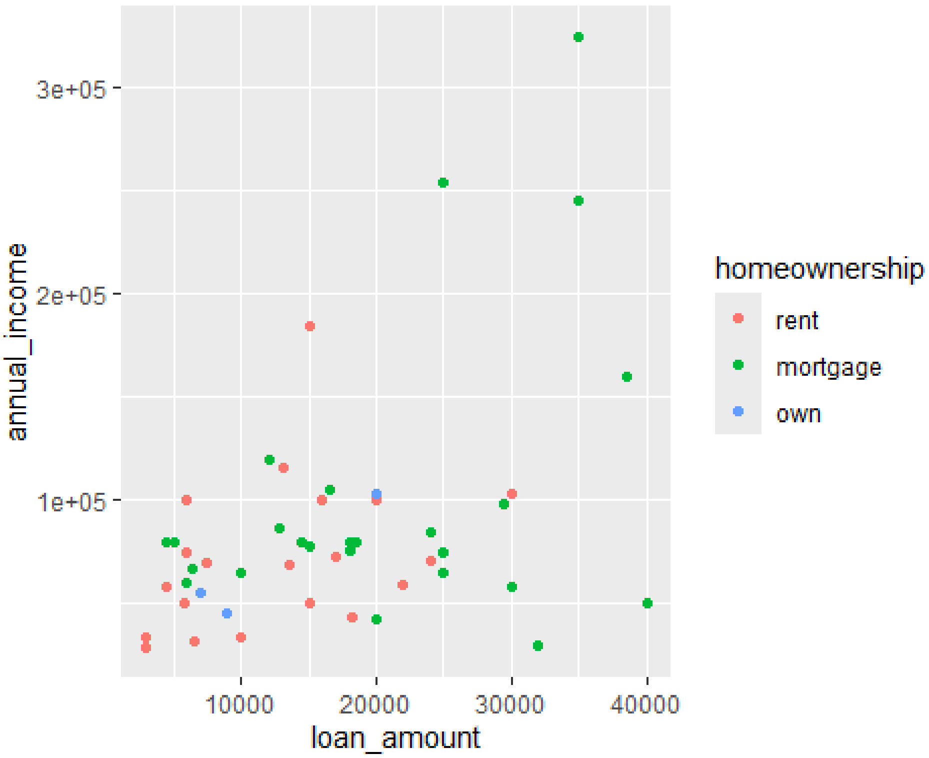
```
# now define the mean loan amount
avg_loan <- mean(loan50$loan_amount)
```

We can see the average loan amount is $17083.

Create a new variable (using mutate)

categorize loan_cat as: “at or above average” and “below average” loan amount

```
loan_new <- loan50 |>
  mutate(loan_cat = if_else(loan_amount < avg_loan, "below average","at or above average"
))
# view the newly created tibble
head(loan_new)

# A tibble: 6 × 19
  state emp_length  term homeownership annual_income verified_income
  <fct>      <dbl> <dbl> <fct>                 <dbl> <fct>
1 NJ             3    60 rent                  59000 Not Verified
2 CA            10    36 rent                  60000 Not Verified
3 SC            NA    36 mortgage              75000 Verified
4 CA             0    36 rent                  75000 Not Verified
5 OH             4    60 mortgage             254000 Not Verified
6 IN             6    36 mortgage              67000 Source Verified
#
```



Now use “group_by” to do the same thing

Notice if you create a new tibble with the backwards arrow, you have to “print” the tibble by naming it or calling head(new_df)

```
loan_df <- loan50 |>
  group_by(homeownership) |>
  count()
loan_df

# A tibble: 3 × 2
# Groups:   homeownership [3]
  homeownership     n
  <fct>         <int>
1 rent             21
2 mortgage         26
3 own               3
```

Calculate proportions of renters between 2 groups under loan_cat

This method (using group_by first and then count) provides marginal row proportions

```
prop1 <- loan_new  |>
  # Group by "group"
  group_by(loan_cat) |>
  count(homeownership) |>
  # Create new variable, prop, using mutate
  mutate(prop = n/sum(n))
prop1
```

```
# A tibble: 6 × 4
# Groups:   loan_cat [2]
  loan_cat              homeownership     n   prop
  <chr>                 <fct>         <int>  <dbl>
1 at or above average rent              5 0.227
2 at or above average mortgage         16 0.727
3 at or above average own               1 0.0455
4 below average       rent             16 0.571
5 below average       mortgage         10 0.357
6 below average       own               2 0.0714
```

Filter further for homeownership == “rent” to compare whether loan_cat had higher proportions of those at or above the mean loan amount.

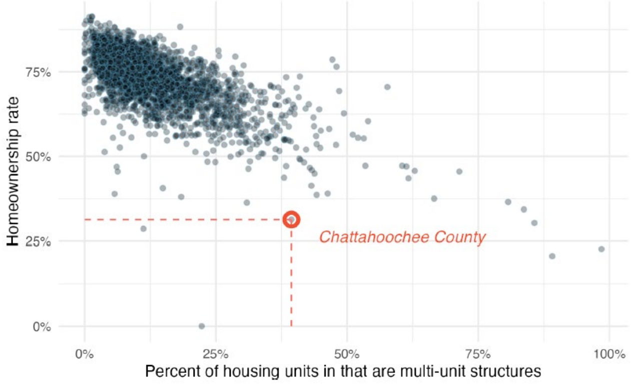
```
prop2 <- loan_new  |>
  # Group by "group"
  group_by(loan_cat) |>
  count(homeownership) |>
  # Create new variable, prop, using mutate
  mutate(prop = n/sum(n)) |>
  filter(homeownership == "rent")
prop2
```

```
# A tibble: 2 × 4
# Groups:   loan_cat [2]
  loan_cat              homeownership     n  prop
  <chr>                 <fct>         <int> <dbl>
1 at or above average rent              5 0.227
2 below average       rent             16 0.571
```

We can see that renters had higher proportions of loans below the average loan amount.

1.2.3 Relationships Between Variables

Scatterplots are one type of graph used to study the relationship between two numerical variables. The plot below displays the relationship between the variables, `homeownership` and `multi_unit`, which is the percent of housing units that are in multi-unit structures (e.g., apartments, condos). Each point on the plot represents a single county. For instance, the highlighted dot corresponds to County 413 in the county dataset: Chattahoochee County, Georgia, which has 39.4% of housing units that are in multi-unit structures and a homeownership rate of 31.3%.



1.2.4 Explanatory and Response Variables

We use the terms **explanatory** and **response** to describe variables where the response might be predicted using the explanatory even if there is no causal relationship.

explanatory variable → *might affect* → response variable

For many pairs of variables, there is no hypothesized relationship, and these labels would not be applied to either variable in such cases.

Homework Chapter 1

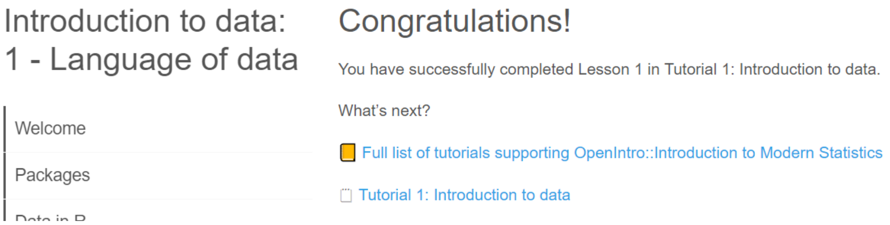
1. Review section 1.3 (the chapter review)
2. Suggested problems from textbook section 1.4 exercises: 1 – 15, odd only
3. Suggested tutorials to learn code: Tutorial 1 – Lessons 1, 2, and 3

○ **Language of data**

○ **Types of studies**

○ **Sampling strategies and experimental design**

It will look something like this:



Congratulations!

You have successfully completed Lesson 1 in Tutorial 1: Introduction to data.

What's next?

Full list of tutorials supporting OpenIntro::Introduction to Modern Statistics

Tutorial 1: Introduction to data