Name: _____

Student Workbook

Workbook v1.5

Brought to you by the Bootstrap team:

- Emmanuel Schanzer
- Kathi Fisler
- Shriram Krishnamurthi
- Dorai Sitaram
- Joe Politz
- Jennifer Poole
- Ed Campos
- Ben Lerner
- Nancy Pfenning
- Flannery Denny

Visual Designer: Colleen Murphy

Introduction to Computational Data Science

Many important questions ("What's the best restaurant in town?", "Is this law good for citizens?", etc.) are answered with *data*.
Data Scientists try and answer these questions by writing *programs that ask questions about data*.
Data of all types can be organized into **Tables**.

- Every Table has a **header row** and some number of **data rows**.
- **Quantitative data** is numeric and measures *an amount*, such as a person's height, a score on a test, distance, etc. A list of quantitative data can be ordered from smallest to largest.
- **Categorical data** is data that specifies *qualities*, such as sex, eye color, country of origin, etc. Categorical data is not subject to the laws of arithmetic — for example, we cannot take the "average" of a list of colors.

Answering questions with data can take many forms. Here are a few types of questions, each requiring a different kind of analysis:

- **Lookup Questions** can be answered just by finding the right row and column of a table. (e.g., "How old is Toggle?")
- **Compute Questions** can be answered by computing over a single row or column. (e.g., "What is the average weight of animals from the shelter?")
- **Relate Questions** require looking for trends across multiple columns. (e.g., "Do cats tend to be adopted sooner than dogs?")

Introduction to Computational Data Science

Students are introduced to the Animals Dataset, learn about Tables, Categorical and Quantitative data, and consider the kinds of questions that can be asked about a dataset.

Prerequisites	None
Relevant Standards K12CS CSTA CC-ELA	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Explain the difference between Categorical and Quantitative data • Identify whether a variable in a dataset is Categorical or Quantitative • Identify the Header Row and Identifier Column of a Table
Student-facing Lesson Goals	• Let's learn about data inside tables.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Each student (or pair of students) should have a Google Account. • Student workbook, and something to write with • Opening questions printed for each student, group of students, or posted around the room. **Note:** these are just ideas to get you started. Use questions that you know will interest *your* students!
Preparation	• Make sure student computers can access the Animals Spreadsheet and the Animals Starter File. • Make sure all materials have been gathered • Decide how students will be grouped in pairs • Decide how the first activity (opening questions) will be run
Supplemental Resources	
Language Table	No language features in this lesson

Glossary

categorical data :: data whose values are qualities that are not subject to the laws of arithmetic.

data science :: the science of collecting, organizing, and drawing general conclusions from data, with the help of computers

programming language :: a set of rules for writing code that a computer can evaluate

quantitative data :: number values for which arithmetic makes sense

---

Introduction

20 minutes

Overview

Students look at opening questions, either at their desks or in a walk around the room. They select a question they are personally interested in, and think about the data required to answer that question. This process draws a direct line between answering questions they care about and the basics of data science.

Launch

- Give students 2 minutes to choose a question that grabs their attention, and group themselves by question. Ideally, no student will be the only one interested in that question.
- Have students spend 2 minutes coming up with a hypothesis about what the answer is, and explaining why. Does every student in a single question-grouping have the same answer?

Investigate

- *What information would you collect to answer this question?* Give students 5 minutes to think about what information they would need to collect, to find the answer.

Possible Misconceptions

Students may lean towards questions about *individuals*, instead of questions about what's true for a *group of individuals* who vary from one to another. For example, instead of wondering what movie gets the highest rating, they should ask what's the typical rating for movies in a list, or how much those ratings tend to vary.

Synthesize

Have students share back the different data they would gather to answer their questions. For each question, students would likely have to gather many different kinds of data. If we wanted to find out if small schools are better than big schools, for example, we might want to gather data on SAT scores, college acceptance, etc. Each of these is a **variable** in our dataset: any two schools we look at could *vary* by each of them.

What's the greatest movie of all time? Is Climate Change real? Who is the best quarterback? Is Stop-and-Frisk racially biased? We can't survey every school in the world, get data on every movie ever made, or every police action - but we can do an analysis for a *sample* of them, and try to infer something about all of them as a whole. These questions quickly turn into a discussion about data—how you assess it, how you interpret the results, and what you can *infer* from those results.

The process of learning from data is called ***Data Science***. Data science techniques are used by scientists, business people, politicians, sports analysts, and hundreds of other different fields to ask and answer questions about data.

We'll use a ***programming language*** to investigate these questions. Just like any human language, programming languages have their own vocabulary and grammar that you will need to learn. The language you'll be learning for data science is called Pyret.

The Animals Dataset

25 minutes

Overview

Students explore the Animals Dataset, sharing observations and familiarizing themselves with the idiosyncrasies and patterns in the data. In the process, they learn about Categorical and Quantitative data.

Notice and Wonder Pedagogy

This pedagogy has a rich grounding in literature, and is used throughout this course. In the "Notice" phase, students are asked to crowd-source their observations. No observation is too small or too silly! Students may notice that the animals table has corners, or that it's printed in black ink. But by listening to other students' observations, students may find themselves taking a closer look at the dataset to begin with. The "Wonder" phase involves students raising questions, but they must also explain the context for those questions. Sharon Hessney (moderator for the NYTimes excellent

What's going on in this Graph? activity) sometimes calls this "what do you wonder...and **why**?". Both of these phases should be done in groups or as a whole class, with time given to each.

Launch

Have students open the Animals Spreadsheet in a browser tab, or turn to The Animals Dataset (Page 2) in their Student Workbooks.

Investigate

This table contains data from an animal shelter, listing animals that have been adopted. We'll be analyzing this table as an example throughout the course, but you'll be applying what you learn to *a dataset you choose* as well.

- Turn to ../../lessons/ds-intro/pages/exploring-animals-dataset.adoc in your Student Workbook. What do you _Notice about this dataset? Write down your observations in the first column.
- Sometimes, looking at data sparks questions. What do you *Wonder* about this dataset, and why? Write down your questions in the second column.
- There's a third column, called "Question Type" — we're going to return to that later, so you can ignore it for now.
- If you look at the bottom of the spreadsheet file, you'll see that this document contains multiple sheets. One is called `"pets"` and the other is called `"README"`. Which sheet are we looking at?
- Each sheet contains a table. For our purposes, we only care about the animals table on the `"pets"` sheet.

Any two animals in our dataset may have different ages, weights, etc. Each of these is called a **variable** in the dataset. Data Scientists work with two broad kinds of data: Categorical Data and Quantitative Data. ***Categorical Data*** is used to *classify*, not measure. Categories aren't subject to the laws of arithmetic. For example, we couldn't ask if "cat is more than lizard", and it doesn't make sense to "find the average ZIP code" in a list of addresses. "Species" is a categorical variable, because we can ask questions like "which species does Mittens belong to?"

What are some other categorical variables you see in this table?

Quantitative Data is used to measure an amount of something, or to compare two pieces of data to see which is *less or more*. If we want to ask "how much" or "which is most", we're talking about Quantitative Data. "Pounds" is a quantitative variable, because we can talk about whether one animal weighs more than another or ask what the average weight of animals in the shelter is.

We use ***Categorical Data*** to answer "what kind?", and ***Quantitative Data*** to answer "how much?".

- Turn to page Categorical or Quantitative? (Page 3), and answer the questions 1-5
- Sometimes it can be tricky to figure out if data is categorical or quantitative, because it depends on *how that data is being used!*
- On ../../lessons/ds-intro/pages/exploring-animals-dataset.adoc in your Student Workbook, fill in the blanks for questions 8-13.

Synthesize

Have students share back their noticings (statements) and wonderings (questions), and write them on the board.
Data Science is all about using a smaller sample of data to make predictions about a larger population. It's important to remember that tables are only a *sample* of a larger population: this table describes some animals, but obviously it isn't every animal in the world! Still, if we took the average age of the animals from this particular shelter, it might tell us something about the average age of animals from other shelters.

Question Types

10 minutes

Overview

Students begin to categorize questions, sorting them into "lookup", "compute", and "relate" questions - as well as questions that simply can't be answered based on the data.

Launch

Once we have a dataset, we can start asking questions! But how do we know what questions to ask? There's an art to asking the right questions, and good Data Scientists think hard about what kind of questions can and can't be answered. Most questions can be broken down into one of four categories:

- **Lookup questions** — These can be answered simply by looking up a single value in the table and reading it out. Once you find the value, you're done! Examples of lookup questions might be "is Sunflower fixed?" or "How many legs does Felix have?"
- **Compute questions** — These can be answered by computing an answer across a single column. Examples of computing questions might be "how much does the heaviest animal weigh?" or "What is the average age of animals from the shelter?"
- **Relate questions** — These ones take the most work, because they require looking for relationships between multiple columns. Examples of analysis questions might be "Do cats tend to be adopted faster than dogs?" or "Are older animals heavier than young ones?"
- **Can't answer** — These are questions that just can't be answered based on the available data. We might ask "are cats or dogs better for elderly owners?", but the Animals Dataset doesn't have information that we can use to answer it.

Investigate

- Come up with examples for each type of question.
- Look back at the Wonders you wrote on ../../lessons/ds-intro/pages/exploring-animals-dataset.adoc. Are any of these Lookup, Compute, or Relate questions? Circle the question type that's appropriate. Can you come up with additional examples for each type of question?

Synthesize

Have students share their questions with the class. Allow time for discussion!

Closing

Debrief with the class, and have students reflect on what they learned by writing on What's on your mind? (Page 5). Some prompts that may be helpful:

- What new vocabulary did you learn?
- What question was exciting to you, and what data would you need to answer it? Is that data Qualitative or Quantitative?
- What do you hope to learn in the next lesson?

Additional Exercises:

- What Questions Can You Answer?

The Animals Dataset

name	species	sex	age	fixed	legs	pounds	weeks
Sasha	cat	female	1	false	4	6.5	3
Snuffles	rabbit	female	3	true	4	3.5	8
Mittens	cat	female	2	true	4	7.4	1
Sunflower	cat	female	5	true	4	8.1	6
Felix	cat	male	16	true	4	9.2	5
Sheba	cat	female	7	true	4	8.4	6
Billie	snail	hermaphrodite	0.5	false	0	0.1	3
Snowcone	cat	female	2	true	4	6.5	5
Wade	cat	male	1	false	4	3.2	1
Hercules	cat	male	3	false	4	13.4	2
Toggle	dog	female	3	true	4	48	1
Boo-boo	dog	male	11	true	4	123	24
Fritz	dog	male	4	true	4	92	3
Midnight	dog	female	5	false	4	112	4
Rex	dog	male	1	false	4	28.9	9
Gir	dog	male	8	false	4	88	5
Max	dog	male	3	false	4	52.8	8
Nori	dog	female	3	true	4	35.3	1
Mr. Peanutbutter	dog	male	10	false	4	161	6
Lucky	dog	male	3	true	3	45.4	9
Kujo	dog	male	8	false	4	172	30
Buddy	lizard	male	2	false	4	0.3	3
Gila	lizard	female	3	true	4	1.2	4
Bo	dog	male	8	true	4	76.1	10
Nibblet	rabbit	male	6	false	4	4.3	2
Snuggles	tarantula	female	2	false	8	0.1	1
Daisy	dog	female	5	true	4	68	8
Ada	dog	female	2	true	4	32	3
Miaulis	cat	male	7	false	4	8.8	4
Heathcliff	cat	male	1	true	4	2.1	2
Tinkles	cat	female	1	true	4	1.7	3
Maple	dog	female	3	true	4	51.6	4

Categorical or Quantitative?

For each piece of data below, circle whether it is **Categorical** or **Quantitative** data.

1	Hair color	categorical	quantitative
2	Age	categorical	quantitative
3	ZIP Code	categorical	quantitative
4	Year	categorical	quantitative
5	Height	categorical	quantitative
6	Sex	categorical	quantitative
7	Street Name	categorical	quantitative

For each question, circle whether it will be answered by **Categorical** or **Quantitative** data.

8	We'd like to find out the average price of cars in a lot.	categorical	quantitative
9	We'd like to find out the most popular color for cars.	categorical	quantitative
10	We'd like to find out which puppy is the youngest.	categorical	quantitative
11	We'd like to find out which cats have been fixed.	categorical	quantitative
12	We want to know which people have a ZIP code of 02907.	categorical	quantitative
13	We'd like to sort a list of phone numbers by area code.	categorical	quantitative

Questions and Column Descriptions

What questions can you ask about the animals dataset? Come up with at least one **Lookup**, **Compute**, **Relate** or **Can't Answer** question, and write them as wonders below. (Note: These question types are defined on Page 1.)

What do you NOTICE about this dataset?	What do you WONDER about this dataset?	*Question Type*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*

1. This dataset is Animals that came from an animal shelter, which contains 31 data rows.

2. Some of the columns are:

   a. species, which contains categorical data. Some example values are: "cat", "dog", and "rabbit".

   b. _____, which contains _____ data. Some example values are: _____.

What's on your mind?

Introduction to Programming in Pyret

Programming languages involve different *datatypes*, such as Numbers, Strings, and Booleans.

- Numbers are values like `1`, `0.4`, `1/3`, and `-8261.003`.
  - Numbers are *usually* used for quantitative data and other values are *usually* used as categorical data.
  - In Pyret, any decimal *must* start with a 0. `0.22` is valid, but `.22` is not.
- Strings are values like `"Emma"`, `"Rosanna"`, `"Jen and Ed"`, or even `"08/28/1980"`.
  - In Pyret, all strings *must* be surrounded in quotation marks.
- Booleans are either `true` or `false`.

Operators (like `+`, `-`, `*`, `<`, etc.) work the same way in Pyret that they do in math.

- Operators are written between values, for example: `4 + 2`.
- In Pyret, operators must always have a space around them. `4 + 2` is valid, but `4+2` is not.
- If an expression has different operators, parentheses must be used to show order of operations. `4 + 2 + 6` and `4 + (2 * 6)` are valid, but `4 + 2 * 6` is not.

Applying Functions also works the way it does in math. The function name is first, followed by a list of **arguments** in parentheses.

- In math this could look like `f(5)` or `f(g(10, 4))`.
- In Pyret this could look like `star(50, "solid", "red")`.
- There are many other Pyret functions, for example `num-sqr`, `num-sqrt` `triangle`, `star`, `string-repeat`, etc.

Functions have **contracts**, which help explain how a function should be used. Every contract has three parts:

- The *Name* of the function - literally, what it's called.
- The *Domain* of the function - what *types of values* the function consumes, and in what order.
- The *Range* of the function - what *type of value* the function produces.

Value Definitions (like `x = 4`, or `y = 9 + 6`) also work the way they do in math. Every value definition starts with a *name*, followed by an equals sign, and then an expression. Once a value is defined, it can be refered to by name.

Starting to Program

Students begin to program in Pyret, learning about basic datatypes, operations, and value definitions.

Prerequisites	None
Relevant Standards CSTA	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Explain the difference between several data types: Numbers, Strings, Images and Booleans • Identify a data type for a given value • Write Numbers, Strings, and Booleans in the Interactions Area • Define values, and evaluate simple expressions that use defined values
Student-facing Lesson Goals	• Let's explore programming in Pyret and learn about data types.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • Make sure student computers can access the Pyret IDE (CPO) • All students will need acces to code.pyret.org, also known as "CPO". They should be able to log in using a Google Classroom, Google, or YouTube login.
Supplemental Resources	
Language Table	Students are not expected to have any familiarity with the Pyret programming for this lesson.

Glossary

data row :: a structured piece of data in a dataset that typically reports all the information gathered about a given individual

definitions area :: the left-most text box in the Editor where definitions for values and functions are written

editor :: software in which you can write and evaluate code

header :: the titles of each column of a table, usually shown at the top

identifier column :: a column of unique values which identify all the individual rows (e.g. - student IDs, SSNs, etc)

interactions area :: the right-most text box in the Editor, where expressions are entered to evaluate

---

Introducing Pyret

10 minutes

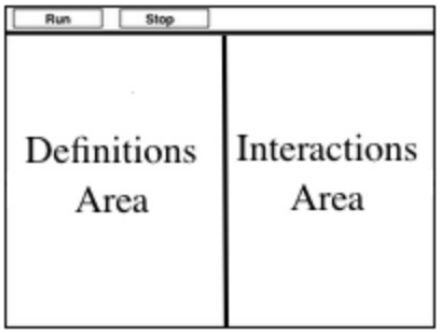
Overview

Students open up the Pyret environment (code.pyret.org, or "CPO") and see how tables look in Pyret.

Students open up the Pyret environment (code.pyret.org, or "CPO") and see how tables look in Pyret.

Launch

Open up the Animals Starter File in a new tab. Click "Connect to Google Drive" to sign into your Google account. This will allow you to save Pyret files into your Google Drive. Next, click the "File" menu and select "Save a Copy". This will save a copy of the file into your own account, so that you can make changes and retrieve them later.



This screen is called the ***Editor***, and it looks something like the diagram you see here. There are a few buttons at the top, but most of the screen is taken up by two large boxes: the ***Definitions Area*** on the left and the ***Interactions Area*** on the right.

The ***Definitions Area*** is where programmers define values and functions that they want to keep, while the ***Interactions Area*** allows them to experiment with those values and functions. This is like writing function definitions on a blackboard, and having students use those functions to compute answers on scrap paper.

For now, we will only be writing programs in the Interactions Area.

The first few lines in the Definitions Area tell Pyret to `import` files from elsewhere, which contain tools we'll want to use for this course. We're importing a file called Bootstrap:Data Science, as well as files for working with Google Sheets, tables, and images:

```
include shared-gdrive("Bootstrap-DataScience-...")
include gdrive-sheets
include tables
include image
```

After that, we see a line of code that *defines* `shelter-sheet` to be a spreadsheet. This table is loaded from Google Drive, so now Pyret can see the same spreadsheet you do. (Notice the funny scramble of letters and numbers in that line of code? If you open up the Google Sheet, you'll find that same scramble in the address bar! That scramble is how the Pyret editor knows which spreadsheet to load.) After that, we see the following code:

```
# load the 'pets' sheet as a table called animals-table
animals-table = load-table: name, species, age, fixed, legs
  source: pets-sheet.sheet-by-name("pets", true)
end
```

The first line (starting with `#`) is called a *Comment*. Comments are notes for humans, which the computer ignores. The next line defines a new table called `animals-table`, which is loaded from the `shelter-sheet` defined above. We also create names for the columns: `name`, `species`, `sex`, `age`, `fixed`, `legs`, `pounds` and `weeks`. We could use any names we want for these columns, but it's always a good idea to pick names that make sense!

> Even if your spreadsheet already has column headers, Pyret requires that you name them in the program itself.

Click "Run", and type `animals-table` into the Interactions Area to see what the table looks like in Pyret. Is it the same table you saw in Google Sheets? What is the same? What is different?

In Data Science, every table is composed of cells, which are arranged in a grid of rows and columns. Most of the cells contain data, but *the first row and first column* are special. The first row is called the ***header row***, which gives a unique name to each variable (or "column") in the table. The first column in the table is the ***identifier column***, which contains a unique ID for each row. Often, this will be the name of each individual in the table, or sometimes just an ID number.

Below is an example of a table with one header row and two data rows:

name	species	sex	age	fixed	legs	pounds	weeks
"Sasha"	"cat"	"female"	1	false	4	6.5	3
"Mittens"	"cat"	"female"	2	true	4	7.4	1

Investigate

- How many variables are listed in the header row for the Animals Dataset? What are they called? What is being used for the identifier column in this dataset?
- Try changing the name of one of the columns, and click "Run". What happens when you print out the table back in the Interactions Area?
- What happens if you remove a column from the list? Or add an extra one?

After the header, Pyret tables can have any number of ***data rows***. Each data row has values for every column variable (nothing can be left empty!). A table can have any number of data rows, including *zero*, as in the table below:

name	species	sex	age	fixed	legs	pounds	weeks

Numbers, Strings and Booleans

25 minutes

Overview

This lesson starts them programming, showing students how to make Pyret do simple math, work with text, and create simple computer graphics. It also draws attention to error messages, which are helpful when diagnosing mistakes.

Launch

Pyret lets us use many different kinds of data. In the animals table, for example, there are Numbers (the number of legs each animal has), Strings (the species of the animal), and Booleans (whether it is true or false that an animal is fixed). Pyret has the usual arithmetic operators: addition (`+`), subtraction (`-`), multiplication (`*`), and division (`/`).

To identify if an animal is male, we need to know if the value in the `sex` column is *equal* to the string `"male"`. To sort the table by age, we need to know if one animal's age is *less than* another's and should come before it. To filter the table to show only young animals, we might want to know if an animal's age is *less than* 2. Pyret has Boolean operators, too: equals (`==`), less-than (`<`), greater-than (`>`), as well as greater-than-or-equal (`>=`) and less-than-or-equal (`<=`).

Investigate

In pairs, students complete Numbers and Strings (Page 7).

Discuss what students have learned about Pyret:

- Numbers and Strings evaluate to themselves.
- Anything in quotes is a String, even something like `"42"`.
- Strings *must* have quotation marks on both sides.
- Operators like `+`, `-`, `*`, and `/` need spaces around them.
- Any time there is more than one operator being used, Pyret requires that you use parentheses.
- Types matter! We can add two Numbers or two Strings to one another, but we can't add the Number `4` to the String `"hello"`.

Error messages are a way for Pyret to explain what went wrong, and are a really helpful way of finding mistakes. Emphasize how useful they can be, and why students should read those messages out loud before asking for help. Have students see the following errors:

- `6 / 0`. In this case, Pyret obeys the same rules as humans, and gives an error.
- A`(2 + 2`. An unclosed quotation mark is a problem, and so is an unmatched parentheses.

In pairs, students complete Booleans (Page 8).

Synthesize

Debrief student answers as a class.

Going Deeper

By using the `and` and `or` operators, we can *combine* boolean tests, as in: `(1 > 2) and ("a" == "b")`. This is handy for more complex programs! For example, we might want to ask if a character in a video game has run out of health points *and* if they have any more lives. We might want to know if someone's ZIP Code puts them in Texas or New Mexico. When you go out to eat at a restaurant, you might ask what items on the menu have meat and cheese. We'll use these Boolean operators in a lot of our Data Science work later on. See "Additional Exercises" if you'd like to have students get some practice with `and` and `or`.

Defining Values

20 minutes

Overview

Students learn how to define values in Pyret (note that these definitions work the way variable substitution does in math, as opposed to variable assignment you may have seen in other programming languages).

Launch

Pyret allows us to define names for values using the `=` sign. In math, you're probably used to seeing definitions like *x = 4*, which defines the name x to be the value 4. Pyret works the same way, and you've already seen two names defined in this file: `shelter-sheet` and `animals-table`. We generally write definitions on the left, in the Definitions Area. You can add your own definitions, for example:

```
my-name = "Maya"
sum = 2 + 2
kittens-are-cute = true
```

With your partner, take turns adding definitions to this file:

- Define a value with name `food`, whose value is a String representing your favorite food
- Define a value with name `year`, whose value is a Number representing the current year
- Define a value with name `likes-cats`, whose value is a Boolean that is `true` if you like cats and `false` if you don't

Synthesize

TODO

Additional Exercises:

- Boolean Operators

Numbers and Strings

Make sure you've loaded the code.pyret.org editor, and clicked "Run".

1. Try typing `42` into the Interactions Area and hitting "Enter". What happens?
2. Try typing in other Numbers. What happens if you try a decimal like `0.5` ? A fraction like `1/3` ? Try really big Numbers, and really small ones.
3. String values are always in quotes. Try typing your name (in quotes!). What happens when you hit Enter?
4. Try typing your name with the opening quote, but *without* the closing quote. What happens? Now try typing it without any quotes.
5. Is *42* the same as `"42"` ? Why or why not? Write your answer below:

Operators

6. Just like math, Pyret has **operators** like `+` , `-` , `*` and `/` . Try typing in `4 + 2` , and then `4+2` (without the spaces). What can you conclude from this? Write your answer below:

7. Type in the following expressions, one at a time: `4 + 2 + 6` , `4 + 2 * 6` , `4 + (2 * 6)` . What do you notice? Write your answer below:

8. Try typing in `4 + "cat"` , and then `"dog" + "cat"` . What can you conclude from this? Write your answer below:

Booleans

Boolean expressions are yes-or-no questions and will always evaluate to either `true` ("yes") or `false` ("no"). What will each of the expressions below evaluate to? Write down the result in the blanks provided, and type them into Pyret if you're not sure.

1) `3 <= 4`		7) `"a" > "b"`	
2) `3 == 2`		8) `"a" < "b"`	
3) `2 < 4`		9) `"a" == "b"`	
4) `3 <> 3`		10) `"a" <> "b"`	
5) `5 >= 5`		11) `"a" <> "a"`	
6) `4 >= 6`		12) `"a" == "a"`	

13) In your own words, desribe what **>** does.

14) In your own words, describe what **<=** does.

15) In your own words, desribe what **<>** does.

16) How many **Numbers** are there in the entire universe?

17) How many **Strings** are there in the entire universe?

18) How many **Images** are there in the entire universe?

19) How many **Booleans** are there in the entire universe?

Applying Functions

Type this line of code into the interactions area and hit "Enter": `triangle(50, "solid", "red")`

1)	What is the name of this function?	_____
2)	What did the expression evaluate to?	_____
3)	How many arguments does `triangle` expect?	_____
4)	What does the `triangle` function produce? (Numbers? Strings? Booleans?)	_____

Catching Bugs

The following lines of code are all BUGGY! Can you spot the mistake? If you have time, type in the buggy code and see if Pyret agrees with you!

5) `triangle(20, "solid" "red")`

Can you spot the mistake? _____

What error message does Pyret return? _____

6) `triangle(20, "solid")`

Can you spot the mistake? _____

What error message does Pyret return? _____

7) `triangle(20, 10, "solid", "red")`

Can you spot the mistake? _____

What error message does Pyret return? _____

8) `triangle (20, "solid", "red")`

Can you spot the mistake? _____

What error message does Pyret return? _____

9) `triangle 20, "solid", "red")`

Can you spot the mistake? _____

What error message does Pyret return? _____

Applying Functions

Students learn how to apply Functions, and how to interpret the information contained in a Contract: Name, Domain and Range. They then use this knowledge to explore more of the Pyret language.

Prerequisites	Starting to Program
Relevant Standards OK CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Apply functions to create Images • Identify the parts of a Contract, and use it to apply a function
Student-facing Lesson Goals	• Let's use different types of input to create images with functions.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • Make sure student computers can access the Pyret IDE (CPO) • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	No language features in this lesson

Glossary

arguments :: the inputs to a function; expressions for arguments follow the name of a function
contract :: a statement of the name, domain, and range of a function
domain :: the type or set of inputs that a function expects
function :: a mathematical object that consumes inputs and produces an output
range :: the type or set of outputs that a function produces

Applying Functions

15 minutes

Overview

Students learn how to apply functions in Pyret, reinforcing concepts from standard Algebra.

Launch

Students know about Numbers, Strings, Booleans and Operators — all of which behave just like they do in math. But what

about ***functions***? They may remember functions from algebra: $f(x) = x^2$.

- What is the name of this function?
- The expression $f(2)$ applies the function f to the number 2. What will it evaluate to?
- What will the expression $f(3)$ evaluate to?
- The values to which we apply a function are called its ***arguments***. How many arguments does f expect?

Arguments (or "inputs") are the values passed into a function. This is different from *variables*, which are the placeholders that get *replaced* with input values! Pyret has lots of built-in functions, which we can use to write more interesting programs.

Have students log into CPO and open the "Animals Starter File". If they don't have the file, they can open a new one. Have students type this line of code into the interactions area and hit Enter: `num-sqrt(16)`.

- What is the name of this function?
- What do we think the expression `num-sqrt(16)` will evaluate to?
- What did the expression `num-sqrt(16)` evaluate to?
- Does the `num-sqrt` function produce Numbers? Strings? Booleans?
- How many ***arguments*** does `num-sqrt` expect?

Have students type this line of code into the interactions area and hit Enter: `num-min(140, 84)`.

- What is the name of this function?
- What does the expression `num-min(16, 99)` evaluate to?
- Does the `num-min` function produce Numbers? Strings? Booleans?
- How many ***arguments*** does `num-min` expect?
- What happens if we forget to include a comma between our numbers?

Just like in math, functions can also be *composed* with one another. For example:

```
# take the minimum of 84 and 99, then take the square root of the result
num-sqrt(num-min(84, 99))
```

Investigation

Have students complete Applying Functions (Page 9).

Synthesize

Debrief the activity with the class. What kind of value was produced by that expression? (An Image! New datatype!) Which error messages were helpful? Which ones weren't?

Contracts

35 minutes

Overview

Students learn about **Contracts**, and how they can be used to figure out new functions or diagnose errors in their code. Then they use this knowledge to explore the contracts pages in their workbooks.

Launch

When students typed `triangle(50, "solid", "red")`, they created an example of a new Datatype, called an *Image*.

- What are the types of the arguments `triangle` was expecting?
- How does this output relate to the inputs?
- Try making different triangles. Change the size and color! Try using `"outline"` for the second argument.

The `triangle` function consumes a Number and two Strings as input, and produces an Image. As you can imagine, there are many other functions for making images, each with a different set of arguments. For each of these functions, we need to keep track of three things:

1. **Name** — the name of the function, which we type in whenever we want to use it
2. **Domain** — the type of data we give to the function (names and Types!), written between parentheses and separated by commas
3. **Range** — the type of data the function produces

Domain and Range are *Types*, not specific values. As a convention, we **capitalize Types and keep names in lowercase**. `triangle` works on many different Numbers, not just the `20` we used in the example above!
These three parts make up a ***contract*** for each function. Let's take a look at the Name, Domain, and Range of the functions we've seen before:

```
# num-sqrt :: (n :: Number) -> Number
# num-min :: (a :: Number, b :: Number) -> Boolean
# triangle :: (side :: Number, mode :: String, color :: String) -> Image
```

The first part of a contract is the function's name. In this example, our functions are named `num-sqrt`, and `triangle`. The second part is the ***Domain***, or the names and types of arguments the function expects. `triangle` has a Number and two Strings as variables, representing the length of each side, the mode, and the color. We write name-type pairs with double-colons, with commas between each one. Finally, after the arrow goes the type of the ***Range***, or the function's output, which in this case is Image.
Contracts tell us a lot about how to use a function. In fact, we can figure out how to use functions we've never seen before, just by looking at the contract! Most of the time, error messages occur when we've accidentally broken a contract.

Investigate

Turn to the back of your workbook, and get some practice reading and using Contracts! Make sure you try out the following functions:

- `text`
- `circle`
- `ellipse`
- `star`
- `string-repeat`

When you've figured out the code for each of these, **write it down in the empty line beneath each contract**. These pages will become your reference for the remainder of the class!
Here's an *example* of another function. Type it into the Interactions Area to see what it does. Can you figure out the contract, based on the example? `string-contains("apples, pears, milk", "pears")`

Possible Misconceptions

Students are *very* likely to randomly experiment, rather than actually using the Contracts page. You should plan to ask lots of direct questions to make sure students are making this connection, such as:

- How many items are in this function's Domain?
- What is the *name* of the 1st item in this function's Domain?
- What is the *type* of the 1st item in this function's Domain?
- What is the *type* of the Range?

Synthesize

You've learned about Numbers, Strings, Booleans, and Images. You've learned about operators and functions, and how they can be used to make shapes, strings, and more!
One of the other skills you'll learn in this class is how to diagnose and fix errors. Some of these errors will be *syntax errors*: a

missing comma, an unclosed string, etc. All the other errors are *contract errors*. If you see an error and you know the syntax is right, ask yourself these two questions:

- What is the function that is generating that error?
- What is the contract for that function?
- Is the function getting what it needs, according to its Domain?

By learning to use values, operations and functions, you are now familiar with the fundamental concepts needed to write simple programs. You will have many opportunities to use these concepts in this course, by writing programs to answer data science questions.
Make sure to save your work, so you can go back to it later!

Additional Exercises:

- Fun with Images
- Reading Contracts
- Matching Expressions and Contracts

Contracts

Consider the following contract:

```
rotate :: (degree :: Number, img :: Image) -> Image
```

What is the **Name** of this function? _____

How many things are in this function's **Domain**? _____

What is the **type** of this function's **first argument**? _____

What is the **name** of this function's **second argument**? _____

What is the **Range** of this function? _____

Circle the expression below that is the correct application of this function, based on its contract.

1. `rotate(45, 90)`
2. `rotate(circle(99, "solid", "green"))`
3. `rotate(25, rectangle(7, 10, "outline", "black"))`
4. `rotate(rectangle(7, 10, "outline", "black"), 25)`

Matching Expressions and Contracts

Match the contract (left) with the expression described by the function being used (right).

Contract			Expression
make-id :: (name :: String, age :: Number) -> Image	1	A	`make-id("Hannah", "Smith")`
phone-bill :: (minutes :: Number, texts :: Number) -> Number	2	B	`make-id("George", 17)`
phone-bill :: (minutes :: Number) -> Number	3	C	`phone-bill(31, 287)`
make-id :: (first :: String, last :: String) -> Image	4	D	`make-id("Jessica", "Jones", 32)`
make-id :: (first :: String, last :: String, age :: Number) -> Image	5	E	`phone-bill(55)`

What's on your mind?

Plotting and Displaying Data

Data Scientists use **displays** to visualize data. You've probably seen some of these charts, graphs and plots yourselves! When it comes to displaying **Categorical Data**, there are two displays that are especially useful.

1. **Bar charts** show the *count or percentage* of rows in each category.
    - Bar charts provide a visual representation of the frequency of values in a categorical column.
    - Bar charts have a bar for every category in a column.
    - The more rows in a category, the taller the bar.
    - Bars in a bar chart can be show in *any order*, without changing the meaning of the chart. However, bars are usually shown in some sensible order (bars for the number of orders for different t-shirt sizes might be presented in order of smallest to largest shirt).
2. **Pie charts** show the *percentage* of rows in each category.
    - Pie charts provide a visual representation of the relative frequency of values in a categorical column.
    - Pie charts have a slice for every category in a column.
    - The more rows in a category, the larger the slice.
    - Slices in a pie chart can be shown in *any order*, without changing the meaning of the chart. However, slices are usually shown in some sensible order (e.g. slices might be shown in alphabetical order or from the smallest to largest slice).

Displaying Categorical Data

Students learn to apply functions to entire Tables, generating pie charts and bar charts. They then explore other plotting and display functions that are part of the Data Science library.

Prerequisites	Applying Functions
Relevant Standards OK K12CS CSTA	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to: • Read pie and bar charts • Explain the difference between pie and bar charts • Generate pie and bar charts (among others) from the Animals Dataset
Student-facing Lesson Goals	• Let's use functions to create graphs from data.
Materials	• Lesson slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay`	

Glossary

bar chart :: a display of categorical data that uses bars positioned over category values; each bar's height reflects the count or percentage of data values in that category

contract :: a statement of the name, domain, and range of a function

domain :: the type or set of inputs that a function expects

pie chart :: a display that uses areas of a circular pie's slices to show percentages in each category

---

Displaying Categorical Variables

10 minutes

Overview

Students extend their understanding of Contracts and function application, learning new functions that consume Tables and produce displays and plots.

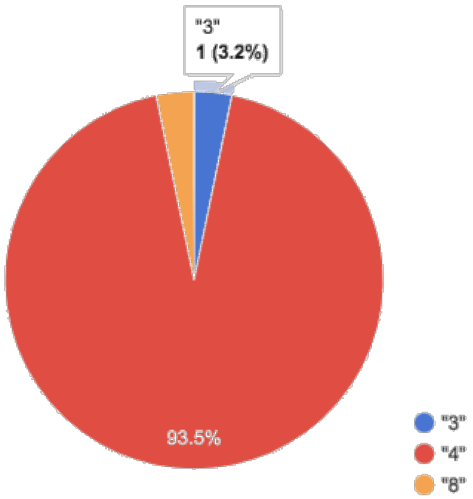
Launch

Have students ever seen any *pictures* created from tables of data? Can they think of a situation when they'd want to consume a *Table*, and use that to produce an image? The library included at the top of the file includes some helper functions that are useful for Data Science, which we will use throughout this course. Here is the ***Contract*** for a function that makes ***pie charts***, and an example of using it:

```
# pie-chart :: (t :: Table, col :: String) -> Image
pie-chart(animals-table, "legs")
```

- What is the Name of this function?
- How many inputs are in its ***Domain***?
- In the Interactions Area, type `pie-chart(animals-table, "legs")` and hit Enter. What happens?

Hovering over a pie slice reveals the label, as well as the count and the percentage of the whole. In this example we see that there is one three-legged animal, representing 3.2% of the population.
We can also resize the window by dragging its borders. This allows us to experiment with the data before closing the window and generating the final, non-interactive image.
The function `pie-chart` consumes a Table of data, along with the *name of a* ***categorical*** *column you want to display*. The computer goes through the column, counting the number of times that each value appears. Then it draws a pie slice for each value, with the size of the slice being the percentage of times it appears. In this example, we used our `animals-table` table as our dataset, and made a pie chart showing the distribution of `legs` across the shelter.



Investigate

Here is the ***Contract*** for another function, which makes ***bar charts***:

```
# bar-chart :: (t :: Table, col :: String) -> Image
```

- Which column of the animals table tells us how many legs an animal has?
- Use `bar-chart` to make a display showing how many animals have each number of legs.
- Experiment with pie and bar charts, passing in different column names. If you get an error message, *read it carefully!*
- What do you think are the rules for what kinds of columns can be used by *bar-chart* and *pie-chart*?
- When would you want to use one chart instead of another?

Possible Misconceptions

Pie charts and bar charts may show counts or percentages (in Pyret, pie charts show percentages and bar charts show counts). Bar charts look a lot like histograms, which are actually quite different because they display quantitative data, not categorical. Also, a pie chart can only display one categorical variable but a bar chart might be used to display two or more categorical variables.

Synthesize

Pie and Bar Charts display what portion of a sample that belongs to each category. If they are based on sample data from a larger population, we use them to *infer* the proportion of a whole population that might belong to each category.

larger population, we use them to *infer* the proportion of a whole population that might belong to each category.

Pie charts and bar charts are mostly used to *display categorical columns* .

While bars in some bar charts should follow some logical order (alphabetical, small-medium-large, etc), the pie slices and bars can technically be placed in *any* order, without changing the meaning of the chart.

Exploring other Displays

30 minutes

Overview

Students freely explore the Data Science display library. In doing so, they experiment with new charts, practice reading ***Contracts*** and error messages, and develop better intuition for the programming constructs they've seen before.

Launch

There are *lots* of other functions, for all different kinds of charts and plots. Even if you don't know what these plots are for yet, see if you can use your knowledge of Contracts to figure out how to use them.

Investigate

Complete Exploring Displays (Page 14) and (More) Exploring Displays (Page 15).

Possible Misconceptions

There are *many* possible misconceptions about displays that students may encounter here. **But that's ok!** Understanding all those other plots is *not* a learning goal for this lesson. Rather, the goal is to have them develop some loose familiarity, and to get more practice reading Contracts.

Synthesize

Today you've added more functions to your toolbox. Functions like `pie-chart` and `bar-chart` can be used to visually display data, and even transform entire tables!

You will have many opportunities to use these concepts in this course, by writing programs to answer data science questions.

Extension Activity

Sometimes we want to summarize a categorical column in a Table, rather than a pie chart. For example, it might be handy to have a table that has a row for dogs, cats, lizards, and rabbits, and then the count of how many of each type there are. Pyret has a function that does exactly this! Try typing this code into the Interactions Area:

```
count(animals-table, "species")
```

What did we get back? `count` is a function that consumes a table and the name of a categorical column, and produces a *new table* with exactly the columns we want: the name of the category and the number of times that category occurs in the dataset. What are the names of the columns in this new table?

- Use the `count` function to make a table showing the number of animals that are `fixed` (or not) from the shelter.
- Use the `count` function to make a table showing the number of animals of each `sex` from the shelter.

Sometimes the dataset we have is *already* summarized in a table like this, and we want to make a chart from *that*. In this situation, we want to base our display on the summary table: the size of the pie slice or bar is taken directly from the count column, and the label is taken directly from the value column. When we want to use summarized data to produce a pie chart, we have another function:

```
# pie-chart-summarized :: (t :: Table, label :: String, data ::
```

```
String) -> Image
pie-chart-summarized(count(animals-table,"species"), "value",
"count")
```

Additional Exercises:

Practice Plotting

Exploring Displays

Using your Contracts page and the Animals Starter File, make each type of display below in pyret. Then sketch the displays and answer the questions. Be sure to add examples of the code you use to your contracts page!

Pie Charts	Bar Charts
Sketch a pie chart here.	Sketch a bar chart here.
Pie charts are constructed from 1 column(s).	Bar charts are constructed from _____ column(s).
They show **categorical** data.	They show _____ data.
What does this display tell us?	What does this display tell us?

Box Plots	Histograms
Sketch a box plot here.	Sketch a histogram here.
Box plots are constructed from _____ column(s).	Histograms are constructed from _____ column(s).
They show _____ data.	They show _____ data.
What does this display tell us?	What does this display tell us?

(More) Exploring Displays

For each type of display, fill in the information below.

Scatter Plots	Linear Regression Plots
Sketch a box plot here.	Sketch a histogram here.
Box plots are constructed from _____ column(s).	Histograms are constructed from _____ column(s).
They show _____ data.	They show _____ data.
What does this display tell us? _____ _____	What does this display tell us? _____ _____

What's on your mind?

Data Displays and Lookups

Data scientists use data visualizations to gain better insights into their data, and to communicate their findings with others. Making a display requires answering three questions:

1. **What data** is being displayed? This could be "a random sample of 2000 people", "every animal from the shelter", or "students' aged 14-17".
2. **What variables** are being explored? Are we looking at the `species` column? The number of `kilograms` that an animal weighs? Searching for a relationship between a person's `income` and their `height`?
3. **What display** is being used, given the variables being explored? If it's a quantitative variable, we might use a histogram or box plot. If it's categorical, we could use a pie or bar chart. If it's two quantitative variables, we probably want a scatter plot.

When **looking up a data Row** from a Table, programmers use the `row-n` method. This method takes a single number as its input, which tells the computer which Row we want. *Note: Rows are numbered starting at zero!*
For example:

```
animals-table.row-n(0)  # access the 1st data row
animals-table.row-n(16) # access the 17th data row
```

When **looking up a column** from a Row, programmers use square brackets and the name of the column they want.
For example:

```
animals-table.row-n(11)["age"]      # look up the age of the animal in the 12st data row
animals-table.row-n(14)["species"]  # look up the species of the animal in the 15th data
row
```

Throughout the rest of the workbook, we will sometimes refer to animalA and animalB.

```
animalA = animals-table.row-n(4)
animalB = animals-table.row-n(13)
```

Data Displays and Lookups

Students continue to practice making different kinds of data displays, this time focusing less on programming and more on using displays to answer questions. They also learn how to extract individual rows from a table, and columns from a row.

Prerequisites	Applying Functions
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Given a human-language request for a data display involving the entire Animals Dataset, break it down into parts and generate the display. • Given a Table, use the `row-n` method to extract any Row from that table • Given a Row, use the column lookups to extract the value of any column in the Row
Student-facing Lesson Goals	• Let's practice making graphs and retrieving information from tables.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count`	

Glossary

categorical data :: data whose values are qualities that are not subject to the laws of arithmetic.

contract :: a statement of the name, domain, and range of a function

method :: a function that is only associated with an instance of a datatype, which consumes inputs and produces an output based on that instance

quantitative data :: number values for which arithmetic makes sense

Displaying Data

20 minutes

Overview

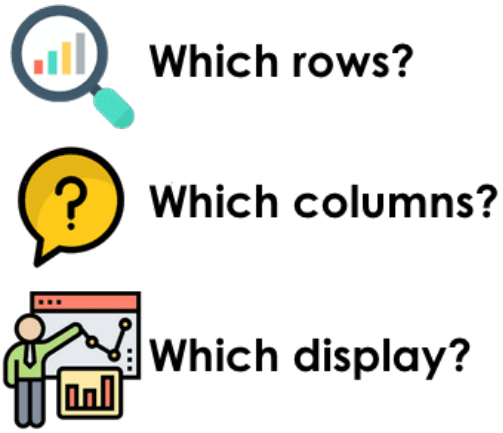
Students get some more practice applying the plotting functions and working with Contracts, and begin to shift the focus from *programming* to *data visualization*. This activity stresses a hard programming skill (reading Contracts) with formal reading comprehension (identifying key portions of the sentence).

Launch

The Contracts page in the back of students' workbooks contains ***contracts*** for many plotting functions.
Suppose we wanted to generate a display showing the ratio of fixed to un-fixed animals from the shelter? How do we go from a simple sentence to working code that makes a data display?

> To make a data display, we ask "Which Rows?", "Which Column(s)?", and "What Display?"

1. We start by asking **which rows** we're talking about. In this case, it's all the animals from the shelter.
2. We also need to know **which column(s)** - or "which variable(s)" - we are displaying. In this case, it's the `fixed` column.
3. Finally, we need to know **which display** we are using. Is it a histogram? Bar chart? Scatter plots are essential for displaying relationships *between* columns, but the other displays only deal with one column. Some displays work for ***categorical data***, and others are for ***quantitative data***.



Once we can answer these questions, all we need to do is find the Contract for that display and fill in the Domain!
To display the categorical data, we can choose between pie and bar charts. Which one of these two is best, and why?

Investigate

Do you know what kind of data is used for each display?

> Turn to What Display Goes with Which Data? (Page 18), and see if you identify what kind of data each display needs!

Let's get some practice going from questions to code, making visualizations.

> Turn to Data Displays (Page 19), and see if you can fill in these three parts for a number of data display requests. When you're finished, try to make the display in Pyret using the appropriate function.

Synthesize

Debrief the activity with students.
Optional: As an extension, have students break into teams and come up with additional Data Display challenges, then race to see which team can complete the other team's challenges first!

Row and Column Lookups

30 minutes

Overview

Students learn how to access individual rows from a table in Pyret, and how to access a particular column from those rows.

Launch

Have students open their saved Animals Starter File (or make a new copy), and click "Run".
Tables have special functions associated with them, called ***Methods***, which allow us to do all sorts of things with those tables. For example, we can get the first data row in a table by using the `.row-n` method: `animals-table.row-n(0)`

tables. For example, we can get the first data row in a table by using the `.row-n` method: `animals-table.row-n(0)`

> Don't forget: data rows start at index *zero*!

For practice, in the Interactions Area, use the `row-n` method to get the second and third data rows.

What is the Domain of .row-n? What is the Range? Find the contract for this method in your contracts table. A table ***method*** is a special kind of function which always operates on a specific table. In our example, we always use .row-n with the animals table, so the number we pass in is always used to grab a particular row from animals-table.

Pyret also has a way for us to get at individual columns of a Row, by using a Row Accessor. Row accessors start with a Row value, followed by square brackets and the name of the column where the value can be found. Here are three examples that use row accessors to get at different columns from the first row in the animals-table:

```
animals-table.row-n(0)["name"]
animals-table.row-n(0)["age"]
animals-table.row-n(0)["fixed"]
```

Investigate

- How would you get the `weeks` column out of the *second* row? The third?
- Complete the exercises on Lookup Questions (Page 20).

We can use the `row-n` method to define entire *animal rows* as values. Type the following lines of code into the Definitions Area and click "Run":

```
animalA = animals-table.row-n(1)
animalB = animals-table.row-n(16)
```

Flip back to page 2 of your workbook and look at The Animals Dataset. Which row is animalA? Label it in the margin next to the dataset. Which row is animalB? Label it in the margin next to the dataset.

Now turn back to your screen. What happens when you evaluate `animalA` in the Interactions Area?

- Define *at least* two additional values to be animals from the `animals-table`, called `animalC` and `animalD`.

Synthesize

Have students share their answers, and see if there are any common questions that arise.

Additional Exercises:

- More Practice with Lookups

What Display Goes with Which Data?

Match the Display with the description of the data being plotted. Some descriptions may go with more than one display!

Pie Charts	**1**	**A**	1 column of Quantitative Data
Bar Charts	**2**		
Histograms	**3**	**B**	2 columns of Quantitative Data
Box Plots	**4**		
Scatter Plots	**5**	**C**	1 column of Categorical Data

Data Displays

Fill in the tables below, then write the Pyret code that will make that display. The first column has been filled in for you.

1) A `pie-chart` showing the `species` of animals from the shelter.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code: _____

2) A `bar-chart` showing the `sex` of animals from the shelter.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code: _____

3) A `histogram` of the number of `pounds` that animals weigh.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code: _____

4) A `box-plot` of the number of `pounds` that animals weigh.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code: _____

5) A `scatter-plot`, using the animals' `species` as the labels, `age` as the x-axis, and `pounds` as the y-axis.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code: _____

6) A `scatterplot`, using the animals' `name` as the labels, `pounds` as the x-axis, and `weeks` as the y-axis.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code: _____

Lookup Questions

The table below represents four pets:

`pets-table`

name	sex	age	pounds
`"Toggle"`	`"female"`	`3`	`48`
`"Fritz"`	`"male"`	`4`	`92`
`"Nori"`	`"female"`	`6`	`35.3`
`"Maple"`	`"female"`	`3`	`51.6`

1) *Match* each Lookup Question (left) to the code that will give the answer (right).

"How much does Maple weigh?"	**1**	**A**	`pets-table.row-n(3)`
"Which is the last row in the table?	**2**	**B**	`pets-table.row-n(2)["name"]`
"What is Fritz's sex?"	**3**	**C**	`pets-table.row-n(1)["sex"]`
"What's the third animal's name?"	**4**	**D**	`pets-table.row-n(3)["age"]`
"How much does Nori weigh?"	**5**	**E**	`pets-table.row-n(3)["pounds"]`
"How old is Maple?"	**6**	**F**	`pets-table.row-n(0)`
"What is Toggle's sex?"	**7**	**G**	`pets-table.row-n(2)["pounds"]`
"What is the first row in the table?"	**8**	**H**	`pets-table.row-n(0)["sex"]`

2) Fill in the blanks (left) with code that will produce the value (right).

a.	pets-table.row-n(3)["name"]	`"Maple"`
b.	_____	`"male"`
c.	_____	`4`
d.	_____	`48`
e.	_____	`"Nori"`

What's on your mind?

Defining Functions

We can **define our own functions** , using a technique called the **Design Recipe** .

- We use the Design Recipe to help us define functions **and think through problems clearly**.
- The first step is to write a **Contract** and **Purpose Statement** for the function, which specify the Name, Domain and Range of the function and give a summary of what it does.
- The second step is to **write at least two examples**, which show how the function should work for specific inputs. These examples help us see patterns, and we express those patterns by **circling and labeling** what changes.
- The final step is to **define the function**, which generalizes our examples.

Defining Functions

Students learn a structured approach to problem solving called the “Design Recipe”. They then use these functions to create images, and learn how to apply them to enhance their scatterplots.

Prerequisites	Applying Functions
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • define one-argument functions that consume a Number and produce an Image • define one-argument functions that consume a String and produce an Image • define one-argument functions that consume a Row and produce an Image • create custom scatter plots, using functions they have defined • define one-argument functions that make Images from Numbers, Strings, and even Rows • create custom scatter plots using those functions
Student-facing Lesson Goals	• Let’s learn how to write our own functions in Pyret.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don’t have the file, they can open a new one
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

design recipe :: a sequence of steps that helps people document, test, and write functions

Defining Functions over Numbers

20 minutes

Defining Functions over Numbers

20 minutes

Overview

Students have learned to define *values* (e.g. - `name = "Maya"` , `x = 5` , etc). Students should have defined `animalA` and `animalB` to be two different rows in the animals table. If they haven't, make sure they do this now.

Launch

Suppose we want to make a solid, green triangle of size 10. What would we type? What if we wanted to make one of size 20? 25? 1000?

```
triangle(10, "solid", "green")
triangle(20, "solid", "green")
triangle(25, "solid", "green")
triangle(1000, "solid", "green")
```

This is a lot of redundant typing, when the only thing changing is the size of the triangle! It would be convenient to define a *shortcut* , which only needs the size. Suppose we call it `gt` for short:

```
gt(10)
gt(20)
gt(25)
gt(1000)
```

We don't need to tell `gt` whether the shape is `"solid"` or `"outline"` , and we don't need to tell it what color to use. We will define our shortcut so it already knows these things, and all it needs is the size. This is a lot like defining values, which we already know how to do. But values don't change, so our triangles would always be the same size. Instead of defining values, we need to define *functions* .

To build our own functions, we'll use a series of steps called the ***Design Recipe***. The Design Recipe is a way to think through the behavior of a function, to make sure we don't make any mistakes with the animals that depend on us! The Design Recipe has three steps, and we'll go through them together for our first function.

Turn to The Design Recipe (Page 23) in your Student Workbook, and read the word problem at the top of the page.

Step 1: Contract and Purpose

The first thing we do is write a Contract for this function. You already know a lot about contracts: they tell us the Name, Domain and Range of the function. Our function is named `gt` , and it consumes a Number. It makes triangles, so the output will be an Image. A Purpose Statement is just a description of what the function does:

```
# gt :: (size :: Number) -> Image
# Consumes a size, and produces a solid green triangle of that size.
```

Since the contract and purpose statement are notes for humans, we add the # symbol at the front of the line to turn them into comments.

Be sure to check students' contracts and purpose statements before having them move on!

Step 2: Write Examples

Examples are a way for us to tell the computer how our function should behave for a specific input. We can write as many examples as we want, but they must all be wrapped in an examples: block and an end statement. Examples start with the name of the function we're writing, followed by an example input. Suppose we write `gt(10)` . What work do we have to do, in order to produce the right shape as a result? What if we write `gt(20)` ?

```
# gt :: (size :: Number) -> Image
# Consumes a size, and produces a solid green triangle of that size.
examples:
```

```
  gt(10) is triangle(10, "solid", "green")
  gt(10) is triangle(10, "solid", "green")
end
```

Step 3: Define the Function

We start with the `fun` keyword (short for "function"), followed by the name of our function and a set of parentheses. This is exactly how all of our examples started, too. But instead of writing `10` or `20`, we'll use the label from our Domain. Then we add a colon (`:`) in place of `is`, and write out the work we did to get the answers for our examples. Finally, we finish with the `end` keyword.

```
# gt :: (size :: Number) -> Image
# Consumes a size, and produces a solid green triangle of that size.
examples:
  gt(10) is triangle(10, "solid", "green")
  gt(10) is triangle(10, "solid", "green")
end
fun gt(size):
  triangle(size, "solid", "green")
end
```

Investigate

Type your function definition into the Definitions Area. Be sure to include the Contract, Purpose Statement, Examples *and* your Definition! Once you have typed everything in, click "Run" and evaluate `gt(10)` in the Interactions Area. What did you get back?

Once we have defined a function, we can use it as our shortcut! This makes it easy to write simpler code, by moving the complexity into a function that can be tested and re-used whenever we like.

- Use the Design Recipe to solve the word problem at the bottom of The Design Recipe (Page 23).
- Type in the Contract, Purpose Statement, Examples and Definition into the Definitions Area.
- Click "Run", and make sure all your examples pass!
- Type bc(20) into the Interactions Area. What happens?

Synthesize

Ask students what happens if they change one of the examples to be incorrect: gt(10) is triangle(99, "solid", "green")

Defining Functions over Other Datatypes 20 minutes

Overview

Students deepen their understanding of function definition and the Design Recipe, by solving different kinds of problems.

Launch

Functions can consume values besides Numbers. For example, we might want to define a function called `sticker` that consumes a *Color*, and draws a star of that color:

```
sticker("blue") is star(50, "solid", "blue")
sticker("yellow") is star(50, "solid", "yellow")
```

Or a function called `nametag` that consumes a *Row* from the animals table, and draws that animal's name in purple letters.

```
nametag(animalA) is text(animalA["name"], 10, "purple")
```

```
nametag(animalB) is text(animalB["name"], 10, "purple")
```

Investigate

Turn to The Design Recipe (Page 24), and use the Design Recipe to write both of these functions.

Custom Scatter Plot Images

15 minutes

Overview

Students discover *functions that consume other functions* , and compose a scatter plot function with one of the functions they've already defined.

Launch

Students have used Pyret functions that use Numbers, Strings, Images, and even Tables and Rows. Now they've written functions of their own that work with these datatypes. However, Pyret functions can even use *other functions* ! Have students at the Contract for `image-scatter-plot` :

```
image-scatter-plot :: (t :: Table, xs :: String, ys :: String, f :: (Row -> Image)) -
> Image
```

This function looks a lot like the regular `scatter-plot` function. It takes in a table, and the names of columns to use for x- and y-values. Take a closer look at the third input...

```
...f :: (Row -> Image)...
```

That looks like the contract for a function! Indeed, the third input to `image-scatter-plot` is named `f` , which itself is a function that consumes Rows and produces Images. In fact, students have just defined a function that does exactly that!

Investigate

- Type `image-scatter-plot(animals-table, "pounds", "weeks", nametag)` into the Interactions Area.
- What did you get?
- What other scatter plots could we create?

Note: the optional lesson If Expressions goes deeper into basic programming constructs, using `image-scatter-plot` to motivate more complex (and exciting!) plots.

Synthesize

Functions are powerful tools, for both mathematics and programming. They allow us to create reusable chunks of logic that can be tested to ensure correctness, and can be used over and over to solve different kinds of problems. A little later on, you'll learn how to combine, or *compose* functions together, in order to handle more complex problems.

Additional Exercises:

- The Design Recipe

The Design Recipe

Directions: Define a function called `gt`, which makes solid green triangles of whatever size we want.

Contract and Purpose Statement

Every contract has three parts…

```
# gt:: (size :: Number) -> Image
```

function name | *domain* | *range*

```
# Consumes a size, and produces a solid green triangle of that size.
```

what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

```
_____ (_____) is _____
```

function name | *input(s)* | *what the function produces*

```
_____ (_____) is _____
```

function name | *input(s)* | *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

```
fun gt(size):
```

function name | *variable(s)*

```
  triangle(size, "solid", "green")
```

what the function does with those variable(s)

end

Directions: Define a function called `bc`, which makes solid blue circles of whatever radius we want.

Contract and Purpose Statement

Every contract has three parts…

```
# _____ :: _____ -> _____
```

function name | *domain* | *range*

```
# _____
```

what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

```
_____ (_____) is _____
```

function name | *input(s)* | *what the function produces*

```
_____ (_____) is _____
```

function name | *input(s)* | *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

```
fun _____(_____):
```

function name | *variable(s)*

```
  _____
```

what the function does with those variable(s)

end

The Design Recipe

Directions: Define a function called `sticker`, which draws 50px stars in whatever color is input.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Directions: Define a function called `nametag`, which consumes a `Row` of the animals table and draws their name in purple, 10px letters. (Assume you have rows `animalA` and `animalB` defined.)

Contract and Purpose Statement

Every contract has three parts…

`nametag` :: `(r :: Row)` -> `Image`
function name *domain* *range*

Consumes an animal, and produces that animal's name in purple, 10px letters.
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

`nametag` (`"animalA"`) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun `nametag` (`r`):
function name *variable(s)*

```
text(r["name"], 10, "purple")
```
what the function does with those variable(s)

end

What's on your mind?

Defining Row Functions & Using Table Methods

Methods are special functions that are attached to pieces of data. We use them to manipulate Tables.

- In this course, the methods we'll be using are
  - `row-n` - consumes an index (starting with zero!) and produces a row from a table
  - `order-by` - consumes the name of a column and a Boolean value to determine if that table should be sorted by that column in ascending order
  - `filter` - consumes a *Boolean-producing function*, and produces a table containing only rows for which the function returns `true`
  - `build-column` - consumes the name of a new column, and a function that produces the values in that column for each Row
- Unlike functions, methods can't be used alone. They have a "secret" argument, which is the data they are attached to. They are written as part of that data, separated by a dot. For example:

```
shapes.row-n(2)
```

- Contracts for methods are different from other functions. They include the type of the data as part of their names. For example:

```
<table>.row-n :: (index :: Number) -> Row
```

Table Methods

Students learn about *table methods*, which allow them to order, filter, and build columns to extend the animals table.

Prerequisites	Defining Functions
Relevant Standards OK CSTA NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • order the Animals Dataset by a number of criteria • filter the Animals Dataset by species, fixed status, and age
Student-facing Lesson Goals	• Let's learn how to start with one table and transform it into another.
Materials	• Lesson Slides (Google Slides) • "Function Purpose cards" which describe simple boolean functions to apply to students • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with • All students should log into CPO and open the Table Methods Starter File • One copy of Function Cards printed and cut.
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay`	

Review Function Definitions

15 minutes

Overview

Students get some practice reading function definitions, and in the process they build knowledge that's needed later on in the lesson.

Launch

Let's see how much you remember about function definitions! Load the Table Methods Starter File, go to the File menu, and click "Save a Copy".

Investigate

Students complete Reading Function Definitions (Page 27) in their student workbooks.

Synthesize

Can students explain what each function does?

Ordering Tables

10 minutes

Overview

Students learn a second table method, which allows them to sort rows in ascending or descending order, according to one column.

Launch

Have students find the contract for `.order-by` in their contracts pages. The `.order-by` method consumes a String (the name of the column by which we want to order) and a Boolean (true for ascending, false for descending). But what does it produce?

Investigate

- Type `animals-table.order-by("name", true)` into the Interactions Area. What do you get?
- Type `animals-table.order-by("age", false)` into the Interactions Area. What do you get?
- Sort the animals table from heaviest-to-lightest.
- Sort the animals table alphabetically by species.
- Sort the animals table by how long it took for each animal to be adopted, in ascending order.

Synthesize

Answer any questions students may have. Class discussion: what do `.order-by` and `.row-n` have in common? How are they different?

Filtering Tables

20 minutes

Overview

Students learn how to *filter* tables, by removing rows.

Launch

Explain to students that you have "Function Cards", which describe the purpose statement of a function that consumes a Row from a table of students, and produces a Boolean (e.g. - "this student is wearing glasses"). Select a volunteer to be the "filter method", and have them *randomly choose* a Function Card, and make sure they read it without showing it to anyone else.

Have ~10 students line up in front of the classroom, and have the filter method go to each student and say "stay" or "sit" depending on whether their function would return true or false for that student. If they say "sit", the student sits down. If they say true, the student stays standing.

Ask the class: based on who sat and who stayed, *what function was on the card?*

> The `.filter` method takes a function, and produces a *new table* containing only rows for which the function returns `true`.

Suppose we want to get a table of only animals that have been fixed? Have students find the contract for `.filter` in their contracts pages. The `.filter` method is taking in a *function*. What is the contract for that function? Where have

we seen functions-taking-functions before?

Investigate

- In the Interactions Area, type `animals-table.filter(is-fixed)`. What did you get?
- What do you expect `animals-table` to produce, and why? Try it out. What happened?
- In the Interactions Area, type `animals-table.filter(is-old)`. What did you get?
- In the Interactions Area, type `animals-table.filter(is-dog)`. What did you get?
- In the Interactions Area, type `animals-table.filter(lookup-name)`. What did you get?

The `.filter` method walks through the table, applying whatever function it was given to each row, and producing a new table containing all the rows for which the function returned `true`. Notice that the Domain for `.filter` says that test must be a function (that's the arrow), which consumes a `Row` and produces a `Boolean`. If it consumes anything besides a single `Row`, or if it produces anything else besides a `Boolean`, we'll get an error.

Possible Misconceptions

Students often think that filtering a table *changes* the table. In Pyret, all table methods produce a *brand new table*. If we want to save that table, we need to define it. For example: `cats = animals-table.filter(is-cat)`.

Synthesize

Debrief with students. Some guiding questions on filtering:

- Suppose we wanted to determine whether cats or dogs get adopted faster. How might using the `.filter` method help?
- If the shelter is purchasing food for older cats, what filter would we write to determine how many cats to buy for?
- Can you think of a situation where filtering fixed animals would be helpful?

Building Columns

10 minutes

Overview

Students learn how to *build columns*, using the `.build-column` table method.

Launch

Suppose we want to *transform* our table, converting `pounds` to `kilograms` or `weeks` to `days`. Or perhaps we want to add a "cute" column that just identifies the puppies and kittens? Have students find the contract for `.build-column` in their contracts pages. The `.build-column` method is taking in a *function* and a *string*. What is the contract for that function?

Investigate

- Try typing `animals-table.build-column("old", is-old)` into the Interactions Area.
- Try typing `animals-table.build-column("sticker", label)` into the Interactions Area.
- What do you get? What do you think is going on?

The `.build-column` method walks through the table, applying whatever function it was given to each row. Whatever the function produces for that row becomes the value of our new column, which is named based on the string it was given. In the first example, we gave it the `is-old` function, so the new table had an extra Boolean column for every animal, indicating whether or not it was young. Notice that the Domain for `.build-column` says that the builder must be a function which consumes a `Row` and produces some other value. If it consumes anything besides a single `Row`, we'll get an error.

Synthesize

Debrief with students. Ask them if they think of a situation where they would want to use this. Some ideas:

- A dataset about school might include columns for how many students are in the school and how many pass the state exam. But when comparing schools of different sizes, what we really want is a column showing what *percentage* passed the exam. We could use `.build-column` to compute that for every row in the table.
- The animals shelter might want to print nametags for every animal. They could build a column using the `text` function to have every animal's name in big, purple letters.
- A dataset from Europe might list everything in metric (centimeters, kilograms, etc), so we could build a column to convert that to imperial units (inches, pounds, etc).

Additional Exercises:

What Table Do We Get?

Reading Function Definitions

Make sure you have the "Table Methods Starter File" open on your computer, and click "Run".

1	How many functions are defined here?	
2	What are their names?	
3	What is the domain of `is-dog` ?	
4	What is the range of `is-old` ?	
5	What is the range of `lookup-name` ?	
6	What does `is-fixed(animalA)` evaluate to?	
7	What does `lookup-name(animalB)` evaluate to?	
8	What does `is-old(animalA)` evaluate to?	
9	What does `is-dog(animalA)` evaluate to?	
10	What does `is-fixed` do?	
11	What does `lookup-name` do?	
12	What does `is-old` do?	

The Design Recipe

For the word problems below, assume `animalA` and `animalB` are defined as the data rows for Felix and Midnight, respectively.

Directions: Define a function called `lookup-fixed`, which looks up whether or not an animal is fixed.

Contract and Purpose Statement

Every contract has three parts...

```
# lookup-fixed:: (r :: Row) -> Boolean
```
function name *domain* *range*

```
# Consumes an animal, and looks up the value in the fixed column.
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

```
(                ) is
```
function name *input(s)* *what the function produces*

```
(                ) is
```
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun lookup-fixed( r ):
```
function name *variable(s)*

```
  r["fixed"]
```
what the function does with those variable(s)

end

Directions: Define a function called `lookup-sex`, which consumes a Row of the animals table and looks up the sex of that animal.

Contract and Purpose Statement

Every contract has three parts...

```
#                :: -> 
```
function name *domain* *range*

```
#
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

```
(                ) is
```
function name *input(s)* *what the function produces*

```
(                ) is
```
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun                ( ):
```
function name *variable(s)*

what the function does with those variable(s)

end

Defining Table Functions

Students continue practicing the Design Recipe, writing helper functions to filter rows and build columns in the Animals Dataset, using Methods.

Prerequisites	Table Methods
Relevant Standards OK K12CS CSTA NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • write custom helper functions to filter the animals table • write custom helper functions to build on the animals table
Student-facing Lesson Goals	• Let's practice writing functions to filter and expand our tables.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with • All students should log into CPO and open the "Table Methods Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Defining Lookup Functions

25 minutes

Overview

Students continue practicing the Design Recipe, by writing functions to answer **Lookup Questions**.

Launch

Take two minutes to find all the fixed animals by hand. Turn to The Animals Dataset, and walk down the table one row at

a time, putting a check next to each animal that is fixed.

To do this activity, what kind of question were you asking of each animal? Was it a **Lookup**, **Compute**, or **Relate** question? You went through the table one row at a time, and for **each row** you did a lookup on the `fixed` column.

Have students type the code that will look up if `animalX` is fixed or not, then do the same with `animalY`. Suppose we wanted to do this for every animal in the table? This seems really repetitive, doesn't it? We would keep typing the same thing over and over, but all that's really changing is the animal. Wouldn't it be great if Pyret had a function called `lookup-fixed`, that would do this for us?

Fortunately, we already know how to define functions using the Design Recipe!

Turn to The Design Recipe (Page 28) in your Student Workbook.

Step 1: Contract and Purpose

The first thing we do is write a Contract for this function. You already know a lot about contracts: they tell us the Name, Domain and Range of the function. Our function is named lookup-fixed, and it consumes a row from the animals table. It looks up the value in the fixed column, which will always be a Boolean. A Purpose Statement is a description of what the function does:

```
# lookup-fixed :: (r :: Row) -> Boolean
# Consumes an animal, and lookup the value in the fixed column
```

Since the contract and purpose statement are notes for humans, we add the # symbol at the front of the line to turn it into a comment. Note that we used "lookup" in the purpose statement!
Be sure to check students' contracts and purpose statements before having them move on.

Step 2: Write Examples

Writing examples for Lookup questions is really simple: all we have to do is look up the correct value in the table, and then write the answer!

```
# lookup-fixed :: (r :: Row) -> Boolean
# Consumes an animal, and looks up the value in the fixed column
examples:
  lookup-fixed(animalX) is true
  lookup-fixed(animalY) is false
end
```

Step 3: Define the Function

When defining the function, we replace the answer with the lookup code.

```
# lookup-fixed :: (animal :: Row) -> Boolean
# Consumes an animal, and looks up the value in the fixed column
examples:
  lookup-fixed(animalX) is true
  lookup-fixed(animalY) is false
end
fun lookup-fixed(r): r["fixed"]
end
```

Investigate

For practice, try using the Design Recipe to define another lookup function.

- Use the Design Recipe to solve the word problem at the bottom of The Design Recipe (Page 28).
- Type in the Contract, Purpose Statement, Examples and Definition into the Definitions Area.

- Click “Run”, and make sure all your examples pass!
- Type `lookup-sex(animalX)` into the Interactions Area.

Defining Compute Functions

25 minutes

Overview

Students define functions that answer **Compute Questions**, again practicing the Design Recipe.

Launch

We’ve only been writing **Lookup Functions**: they consume a Row, look up one column from that row, and produce the result as-is. And as long as that row contains Boolean values, we can use that function with the `.filter` method.
But what if we want to filter by a Boolean expression? For example, what if we want to find out specifically whether or not an animal is a cat, or whether it’s young? Let’s walk through an example of a Compute Function using the Design Recipe, by turning to The Design Recipe (Page 29).
Suppose we want to define a function called `is-cat`, which consumes a row from the `animals-table` and returns true if the animal is a cat.

- Is this a Lookup, Compute or Relate question?
- What is the name of this function? What are its Domain and Range?
- Is Sasha a cat? *What did you do to get that answer?*

To find out if an animal is a cat, we look-up the species column and check to see if that value is *equal* to `"cat"`. Suppose `animalX` is a cat and `animalY` is a dog. What should our examples look like? **Remember: we replace any lookup with the actual value, and check to see if it is equal to `"cat"`.**

```
# is-cat :: (r :: Row) -> Boolean
# Consumes an animal, and compute whether the species is "cat"
examples:
  is-cat(animalX) is "cat" == "cat"
  is-cat(animalY) is "dog" == "cat"
end
```

Write two examples for your defined animals. Make sure one is a cat and one isn’t!

As before, we’ll use the pattern from our examples to come up with our definition.

```
# is-cat :: (r :: Row) -> Boolean
# Consumes an animal, and compute whether the species is "cat"
examples:
  is-cat(animalX) is "cat" == "cat"
  is-cat(animalY) is "dog" == "cat"
end
fun is-cat(r): r["species"] == "cat"
end
```

Don’t forget to include the lookup code in the function definition! We only write the actual value for our examples!

Investigate

- Type this definition—and its examples!—into the Definitions Area, then click “Run” and try using it to filter the `animals-table`.
- For practice, try solving the word problem for `is-young` at the bottom of The Design Recipe (Page 29).

Synthesize

Debrief as a class. Ask students to brainstorm some other functions they could write?

The Design Recipe

For the word problems below, assume `animalA` and `animalB` are defined as the data rows for Felix and Midnight, respectively.

Directions: Define a function called `is-cat`, which consumes a `Row` of the animals table and *computes* whether the animal is a cat.

Contract and Purpose Statement

Every contract has three parts...

```
# is-cat:: (r :: Row) -> Boolean
```
function name *domain* *range*

```
# Consumes an animal, and computes whether the species == "cat"
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

```
is-cat ( "animalA" ) is
```
function name *input(s)* *what the function produces*

```
( ) is
```
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun is-cat( r ):
  r["species"] == "cat"
```
function name *variable(s)* *what the function does with those variable(s)*

end

Directions: Define a function called `is-young`, which consumes a Row of the animals table and *computes* whether it is less than four years old.

Contract and Purpose Statement

Every contract has three parts...

```
# :: ->
```
function name *domain* *range*

```
#
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

```
( ) is
```
function name *input(s)* *what the function produces*

```
( ) is
```
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun ( ):
```
function name *variable(s)*

what the function does with those variable(s)

end

What's on your mind?

Method Chaining

Method chaining allows us to apply multiple methods with less code.

For example, instead of using multiple definitions, like this:

```
with-labels = animals-table.build-column("labels", nametag)
cats = with-labels.filter(is-cat)
cats.order-by("age", true)
```

We can use method-chaining to write it all on one line, like this:

```
animals-table.build-column("labels", nametag).filter(is-cat).order-by("age", true)
```

Order Matters! The methods are applied in the order they appear. For example, trying to order a table by a column that hasn't been built will result in an error.

Method Chaining

Students continue practicing their Design Recipe skills, making lots of simple functions dealing with the Animals Dataset. Then they learn how to chain Methods together, and define more sophisticated subsets.

Prerequisites	Defining Table Functions
Relevant Standards OK K12CS CSTA NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Use method chaining to write more sophisticated analyses using less code • Identify bugs introduced by chaining methods in the wrong order
Student-facing Lesson Goals	• Let's practice writing functions and combining methods together.
Materials	• Lesson slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, order-by, .filter, .build-column`	

Design Recipe Practice

25 minutes

Overview

Students practice more of what they learned in the previous lesson, applying the Design Recipe to simple table functions that operate on rows of the Animals Dataset. The functions they create - in addition to the ones they've already made - set up the method-chaining activity.

Launch

Launch

The Design Recipe is a powerful tool for solving problems by writing functions. It's important for this to be like second nature, so let's get some more practice using it!

Investigate

Define the Compute functions on The Design Recipe (Page 32) and The Design Recipe (Page 33).

Synthesize

Did students find themselves getting faster at using the Design Recipe? Can students share any patterns they noticed, or shortcuts they used?

Chaining Methods

25 minutes

Overview

Students learn how to perform multiple table operations (sorting, filtering, building) in the same line of code.

Launch

Now that we are doing more sophisticated analyses, we might find ourselves writing the following code:

```
# get a table with the nametags of all the fixed animals, ordered by species
fixed = animals-table.filter(is-fixed)
fixed-with-nametags = fixed.build-column("tag", nametag)
result = fixed-with-nametags.order-by("species", true)
```

That's a lot of code, and it also requires us to come up with names for each intermediate step! Pyret allows table methods to be *chained together*, so that we can build, filter *and* order a Table in one shot. For example:

```
# get a table with the nametags of all the fixed animals, ordered by species
result = animals-table.build-column("label", nametag).filter(is-fixed).order-
by("species", true)
```

This code takes the `animals-table`, and builds a new column. According to our Contracts Page, `.build-column` produces a new Table, and that's the Table whose `.filter` method we use. That method produces *yet another Table*, and we call that Table's `order-by` method. The Table that comes back from that is our final result.

Teaching Tip

Use different color markers to draw *nested boxes* around each part of the expression, showing where each Table came from.

It can be difficult to read code that has lots of method calls chained together, so we can add a line-break before each "`.`" to make it more readable. Here's the exact same code, written with each method on its own line:

```
# get a table with the nametags of all the fixed animals, order by species
animals-table
  .build-column("label", nametag)
  .filter(is-fixed)
  .order-by("species", true)
```

Order matters: Build, Filter, Order.

Suppose we want to build a column and then use it to filter our table. If we use the methods in the wrong order (trying to filter by a column that doesn't exist yet), we might wind up crashing the program. Even worse, the program might work, but produce results that are incorrect!

produce results that are incorrect!

Investigate

> When chaining methods, it's important to build first, then filter, and then order.

How well do you know your table methods? Complete Chaining Methods (Page 34) and Chaining Methods 2: Order Matters! (Page 35) in your Student Workbook to find out.

Synthesize

As our analysis gets more complex, method chaining is a great way to keep the code simple. But complex analysis also has more room for mistakes, so it's critical to think carefully when we use it!

The Design Recipe

For the word problems below, assume you have `animalA` and `animalB` defined in your code.

Directions: Define a function called `is-dog`, which consumes a `Row` of the animals table and *computes* whether the animal is a dog.

Contract and Purpose Statement

Every contract has three parts…

```
# is-dog:: (r :: Row) -> Boolean
```
function name *domain* *range*

```
# Consumes an animal, and computes whether the species == "dog"
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

```
examples:
  is-dog ( "animalA" ) is animalA["species"] == "dog"
  is-dog ( "animalB" ) is
end
```
function name *input(s)* *what the function produces*

Definition

Write the definition, giving variable names to all your input values…

```
fun is-dog( r ):
  r["species"] == "dog"
end
```
function name *variable(s)* *what the function does with those variable(s)*

Directions: Define a function called `is-female`, which consumes a Row of the animals table and returns true if the animal is female.

Contract and Purpose Statement

Every contract has three parts…

```
# :: ->
```
function name *domain* *range*

```
#
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

```
examples:
  ( ) is
  ( ) is
end
```
function name *input(s)* *what the function produces*

Definition

Write the definition, giving variable names to all your input values…

```
fun ( ):

end
```
function name *variable(s)* *what the function does with those variable(s)*

The Design Recipe

For the word problems below, assume you have `animalA` and `animalB` defined in your code.

Directions: Define a function called `is-old`, which consumes a Row of the animals table and *computes* whether it is more than 12 years old.

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Directions: Define a function called `name-has-s`, which returns true if an animal's name contains the letter "s"

Contract and Purpose Statement

Every contract has three parts...

`name-has-s` :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun `name-has-s` (`r`):
function name *variable(s)*

```
string-contains(r["name"], "s")
```
what the function does with those variable(s)

end

Chaining Methods

You have the following functions defined below (read them *carefully!*):

```
fun is-fixed(r): r["fixed"]                 end
fun is-young(r): r["age"] < 4               end
fun nametag(r):  text(r["name"], 20, "red") end
```

The table `t` below represents four animals from the shelter:

name	sex	age	fixed	pounds
"Toggle"	"female"	3	true	48
"Fritz"	"male"	4	true	92
"Nori"	"female"	6	true	35.3
"Maple"	"female"	3	true	51.6

Match each Pyret expression (left) to the description of what it does (right).

`t.order-by("age", true)`	**1**	**A**	Produces a table containing only Toggle and Maple
`t.filter(is-fixed)`	**2**	**B**	Produces a table of only young, fixed animals
`t.build-column("sticker", nametag)`	**3**	**C**	Produces a table, sorted youngest-to-oldest
`t.filter(is-young)`	**4**	**D**	Produces a table with an extra column, named "sticker"
`t.filter(is-young)` `  .filter(is-fixed)`	**5**	**E**	Produces a table containing Maple and Toggle, in that order
`t.filter(is-young)` `  .order-by("pounds", false)`	**6**	**F**	Produces a table containing the same four animals
`t.build-column("label", nametag)` `  .order-by("age", true)`	**7**	**G**	Won't run: will produce an error
`t.order-by("agee", false)`	**8**	**H**	Produces a table with an extra "label" column, sorted youngest-to-oldest

Chaining Methods 2: Order Matters!

You have the following functions defined below (read them *carefully!*):

```
fun is-female(r): r["sex"] == "female"  end
fun kilograms(r): r["pounds"] / 2.2     end
fun is-heavy(r):  r["kilos"] > 25       end
```

The table `t` below represents four animals from the shelter:

name	sex	age	fixed	pounds
"Toggle"	"female"	3	true	48
"Fritz"	"male"	4	true	92
"Nori"	"female"	6	true	35.3
"Maple"	"female"	3	true	51.6

Match each Pyret expression (left) to the description of what it does (right). **Note: one description might match multiple expressions!**

```
t.order-by("kilos", true)
```

1

A Produces a table containing Toggle, Nori and Maple, with an extra column showing their weight in kilograms

```
t.filter(is-female)
  .build-column("kilos", kilograms)
```

2

B Produces a table containing Maple, Nori and Toggle (in that order)

```
t.build-column("kilos", kilograms)
  .filter(is-heavy)
```

3

C Produces a table containing only Fritz, with a single extra column called `kilos`

```
t.filter(is-heavy)
  .build-column("kilos", kilograms)
```

4

D Won't run: will produce an error

```
t.build-column("kilos", kilograms)
  .filter(is-heavy)
  .order-by("sex", true)
```

5

E Produces a table containing only Fritz, with two extra columns

```
t.build-column("female", is-female)
  .build-column("kilos", kilograms)
  .filter(is-heavy)
```

6

F Produces a table containing Maple and Fritz

What's on your mind?

Mood Generator

1) Open the Mood Generator starter file, and read through the code you find there. This code contains new programming that you haven't seen yet! Take a moment to list everything you Notice, and then everything you Wonder...

Notice	Wonder

2) Add another line of code to the definition, so that `mood("mad")` produces the *same* emoji as `mood("angry")`.

3) Add **another example** to the `examples:` section for "laughing", using the appropriate emoji. (To bring up the emojis on your computer, type `Cmd-Ctrl-Space` on a Mac, or `Windows-Period` on Windows 10)

4) Come up with some new moods, and add them to the code. Make sure you include `examples:` !

5) In your own words, how do if-expressions work in Pyret? Write your answer below.

6) Write down at least 2 ways you could use if-expressions when analyzing the Animals Dataset.

If-Expressions

Students build on their knowledge of the image-scatter-plot function, motivating the need for if-expressions in their programming toolkit. This drives deeper insight into subgroups within a population, and motivates the need for more advanced analysis.

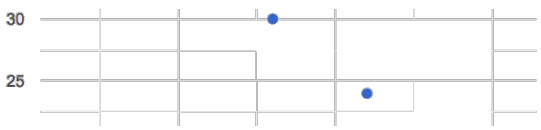
Prerequisites	Defining Functions
Relevant Standards K12CS CSTA NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • use if-then-else expressions in Pyret • explain the behavior of a (specific) higher order function
Student-facing Lesson Goals	• Let's explore functions that behave differently based on the input.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with • The Mood Generator Starter File (CPO) • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	

Language Table

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Warmup

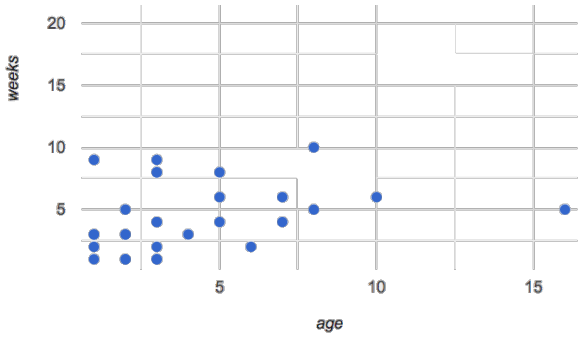
1. Show students this code, which uses `image-url` and `scale` to generate icons of animals.
2. What do they Notice? What do they Wonder? How might



this scatterplot change our analysis?

3. Have students make a scatter plot of animals, using `age` as the x-axis values and `weeks` as the y-axis.

(For now, the scatter plot is *purely* to give students practice with contracts and displays. They are **not** expected to know much about scatter plots at this point.)



If-Expressions

20 minutes

Overview

Students explore a program that makes use of an *if-expression*, develop their own understanding, and modify it.

Launch

So far, all of the functions we know how to write have had a *single rule*. The rule for `gt` was to take a number and make a solid, green triangle of that size. The rule for `bc` was to take a number and make a solid, blue circle of that size. The rule for `nametag` was to take a row and make an image of the animal's name in purple letters.

What if we want to write functions that apply different rules, depending on the input? For example, what if we want to change the color of the nametag depending on the species of the animal?

Investigate

- Open the Mood Generator starter file.
- Complete Mood Generator (Page 37) in your student workbooks.

Synthesize

Have the class share their own explanations for how if-expressions work.

Pyret allows us to write if-expressions, which contain:

1. the keyword `if`, followed by a *condition*.
2. a colon (`:`), followed by a rule for what the function should do if the condition is `true`
3. an `else:`, followed by a rule for what to do if the condition is `false`

We can chain them together to create multiple rules, with the last `else:` being our fallback in case every other condition is `false`.

Better Image Scatter Plots

20 minutes

Overview

Suppose we want to make a scatter plot for the Animals Dataset, but with each dot being a different color depending on the species. This would make it possible to see if different animals are "clustered" in different parts of the plot.

Investigate

Have students open Word Problem: species-color (Page 38). Make sure they all write the Contract and Purpose Statement *first*, and check in with their partner *and* the teacher before proceeding.

Once they've got the Contract and Purpose Statement, have them come up with `examples:` for *each species*. Once again, have them check with a partner *and* the teacher before finishing the page.

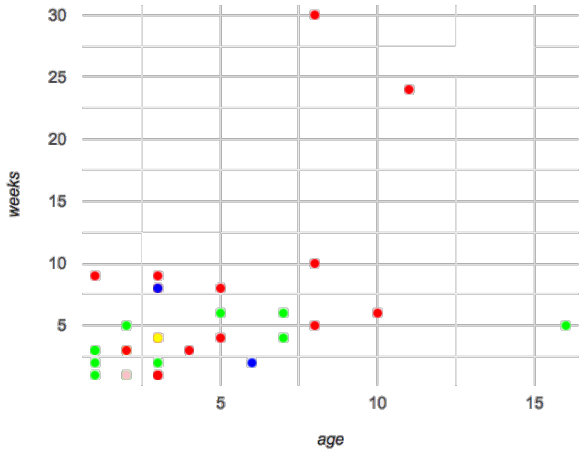
Once another student *and* the teacher has checked their work, have them type this function into their animals starter file, and use it to make an image scatter plot using [illegible] as the x-axis and [illegible] as the y-axis.

files, and use it to make an `image-scatter-plot` using `age` as the x-axis and `weeks` as the y-axis.

Synthesize

1. What do you Notice about this scatter plot?
2. What do you Wonder?

What does this new visualization tell us about the relationship between age and weeks? What other analysis would be helpful here?



Closing

Make sure to direct the conversation *back to Data Science!* Does this scatter plot make us think we should be analyzing animals separately? What other scatter plots might this be useful for?
This scatterplot makes it clear that we may want to analyze each species separately, rather than grouping them all together! In the next lesson, students will learn how to do just that.

Word Problem: species-color

Directions: We want to generate a custom dot for our `image-scatter-plot`, such that every species gets a unique color. Write a function called `species-color`, which takes in a Row from the animals table and returns a solid, 5px circle using a color you've chosen.

Contract and Purpose Statement

Every contract has three parts…

_____ **::** _____ **->** _____

function name *domain* *range*

_____

what does the function do?

Examples

Write some examples, then circle and label what changes…

`examples:`

_____ (_____) **`is`** _____

function name *input(s)* *what the function produces*

_____ (_____) **`is`** _____

function name *input(s)* *what the function produces*

_____ (_____) **`is`** _____

function name *input(s)* *what the function produces*

_____ (_____) **`is`** _____

function name *input(s)* *what the function produces*

_____ (_____) **`is`** _____

function name *input(s)* *what the function produces*

`end`

Definition

Write the definition, giving variable names to all your input values…

`fun` _____ (_____):

function name *variable(s)*

_____

what the function does with those variable(s)

_____

what the function does with those variable(s)

_____

what the function does with those variable(s)

_____

what the function does with those variable(s)

_____

what the function does with those variable(s)

`end`

`end`

Randomness and Sample Size

Computer Scientists may take **samples** that are subsets of a data set. If their sample is well chosen, they can use it to test if their code does what it's supposed to do. However, choosing a good sample can be tricky!

Random Samples are a subset of a population in which each member of the subset has an equal chance of being chosen. A random sample is intended to be a representative subset of the population. The larger the random sample, the more closely it will represent the population and the better our inferences about the population will tend to be.

Grouped Samples are a subset of a population in which each member of the subset was chosen for a specific reason. For example, we might want to look at the difference in trends between two groups ("Is the age of a dog a bigger factor in adoption time v. the age of a cat?"). This would require making grouped samples of *just the dogs* and *just the cats*.

Randomness and Sample Size

Students learn about random samples and statistical inference, as applied to the Animals Dataset. In the process, students get a light introduction to the role of sample size and the importance of statistical inference.

Prerequisites	Defining Table Functions
Relevant Standards OK CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Take random samples from a population • Understand the need for random samples • Understand the role of sample size
Student-facing Lesson Goals	• Let's explore how random sampling can be used with datasets.
Materials	• Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Lesson slides (Google Slides) • Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

statistical inference :: using information from a sample to draw conclusions about the larger population from which the sample was taken

---

Do Now

Students should log into CPO open the Random Samples Starter File, and save a copy.

Flip the Script: Inference v. Probability

30 minutes

Overview

Statistical inference involves looking at a sample and trying to *infer something you don't know* about a larger population. This requires a sort of backwards reasoning, kind of like making a guess about a *cause*, based on the *effect* that we see. To better understand the process of going from the sample back to the population, it helps to understand the more straightforward process of going from the population to a sample. If the sample is random, we call this process Probability! In real life we typically don't know what's true for an entire population. But this probability thought-experiment will start with a larger population with *known* properties (such as the fact that half of the entire population are males). Then we'll see what kind of behavior we tend to see in random samples taken from that population.

Launch

Inference Reasons Backwards; Probability Reasons Forwards

One of the most useful tasks in Data Science is using sample data to *infer* (guess) what's true about the larger population from which the sample was taken. This process, called ***statistical inference***, is used to gain information in practically every field of study you can imagine: medicine, business, politics, history; even art! Early on, statisticians discovered that *random* samples almost always work best.

Suppose we want to make an educated guess about who the next US president will be. We can't ask everyone who they're voting for, so pollsters instead take a *sample* of Americans, and *generalize* the opinion of the sample to estimate how Americans as a whole feel. But choosing a sample can be tricky...

- Would it be problematic to only call voters who are registered Democrats? To only call voters under 25? To only call regular churchgoers? Why or why not?
- How could we choose a representative subset, or *sample* of American voters?
- Would it be problematic to only sample a handful of voters? What do we gain by taking a larger sample?

Before we infer something *unknown* about a population from a sample, we need to know what makes a "good" sample!

Sampling is a complicated issue. The main reason for doing inference is to guess about something that's *unknown* for the whole population. But a useful step along the way is to practice with situations where we happen to *know* what's true for the whole population. As an exercise, we can keep taking random samples from that population and see how close they tend to get us to the truth. Another discovery (besides the value of randomness) that statisticians made early on was something that's perfectly consistent with common sense: Larger samples are better than smaller ones, because they tend to get us closer to the truth about the whole population.

Let's see what happens if we switch from smaller to larger sample sizes, if we're taking a random sample of shelter animals to infer what's true about the larger population...

Investigate

The Animals Dataset we've been using is just one *sample* taken from a very large animal shelter. How much can we infer about the whole population of hundreds of animals, by looking at just this one sample?

- Divide the class into groups of 3-5 students.
- Have students open the Random Samples Starter File, and click "Run".
- Have students complete Sampling and Inference (Page 40), sharing their results and discussing with the group.

Synthesize

Have students share how much better their larger samples are at guessing the truth about the whole population.

Common Misconceptions

Larger populations need to be represented by larger sample sizes. In fact, the formulas that Data Scientists use to assess how good a job the sample does is only based on the *sample size*, not the population size.

how good a job the sample does is only based on the sample size, not the population size.

Going Deeper

If appropriate for your learning goes, this is a great place to include more rigorous statistics content about sample size.

Additional Exercises

- Project: Project: Food Habits
- Project: Project: Time-Use

Sampling and Inference

1) Evaluate the `big-animals-table` in the Interactions Area. This is the *complete* population of animals from the shelter! Below is a true statement about that population:

The population is 47.7% fixed and 52.3% unfixed.

2) How close to these percentages do we get with random samples?
Type each of the following lines into the Interactions Area and hit "Enter".

```
random-rows(big-animals-table, 10)
random-rows(big-animals-table, 40)
```

3) What do you get?

_____

4) What is the contract for `random-rows` ? _____

5) What does the `random-rows` function do?

_____

_____

6) In the Definitions Area, define `tiny-sample` and `small-sample` to be these two random samples.

7) Make a `pie-chart` for the animals in each sample, showing percentages of fixed and unfixed.

- The percentage of fixed animals in the entire populations is **47.7%**.
- The percentage of fixed animals in `tiny-sample` is _____.
- The percentage of fixed animals in `small-sample` is _____.

8) Make a `pie-chart` for the animals in each sample, showing percentages for each species.

- The percentage of tarantulas in the entire population is **roughly 5%**.
- The percentage of tarantulas in `tiny-sample` is _____.
- The percentage of tarantulas in `small-sample` is _____.

9) Click "Run" to direct the computer to generate a different set of random samples of these sizes. Make a new `pie-chart` for each sample, showing percentages for each species.

- The percentage of tarantulas in the entire population is **roughly 5%**.
- The percentage of tarantulas in `tiny-sample` is _____.
- The percentage of tarantulas in `small-sample` is _____.

10) Which repeated sample gave us a more accurate inference about the whole population? Why?

_____

_____

Grouped Samples from the Animals Dataset

Use method chaining to define the **grouped samples** below, using the helper functions that you've already defined: `is-old`, `is-young`, `is-cat`, `is-dog`, `is-female`, `lookup-fixed`, and `has-s-name`. We've given you the solution for the first sample, to get you started.

Subset	The code to define that subset
Kittens	`kittens = animals-table.filter(is-cat).filter(is-young)`
Puppies	`young-dogs = animals-table.` _____
Fixed Cats	`fixed-cats = animals-table.` _____
Cats with "s" in their name	`s-cats = animals-table.` _____
Old Dogs	`old = animals-table.` _____
Fixed Animals	`fixed = animals-table.` _____
Old Female Cats	`old-cats = animals-table.` _____
Fixed Kittens	`young-fixed-cats = animals-table.` _____
Fixed Female Dogs	`fixed-female-dogs = animals-table.` _____
Old Fixed Female Cats	`old-fixed-female-cats = animals-table.` _____

Grouped Samples

Students learn about grouped samples, and practice creating them from the Animals Dataset. In the process, they practice using the Design Recipe to create filter functions, and come up with questions they wish to explore.

Prerequisites	Defining Table Functions
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Make grouped samples from a population
Student-facing Lesson Goals	• Let's combine what we know about sampling and filtering with creating displays.
Materials	• Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Lesson slides (Google Slides) • Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

grouped sample :: a non-random subset of individuals chosen from a larger set, where the individuals belong to a specific group

---

Problems with a Single Population

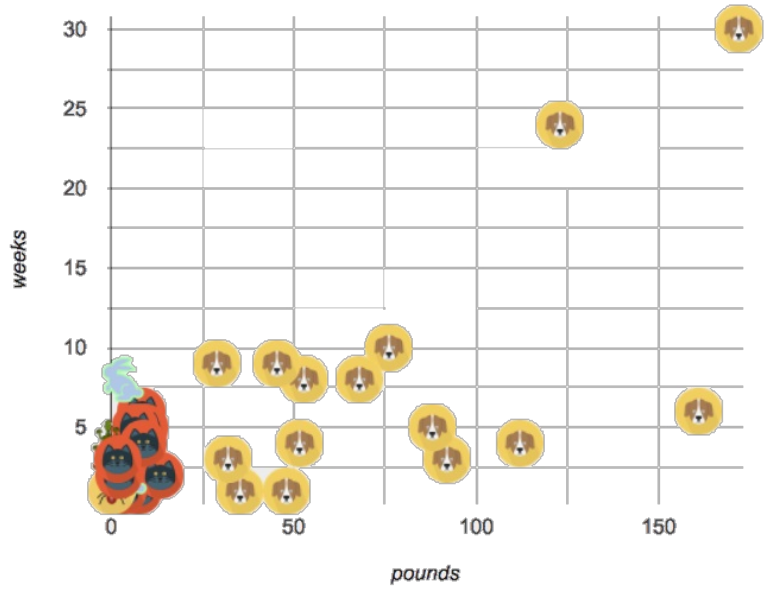
10 minutes

Overview

This activity is all about ***grouped samples***: Students make a bunch of subsets from the Animals Dataset, and see how each subset might answer the same question differently.

Launch

When looking at a scatter plot of our animals, it looks like the amount an animal weighs may have something to do with how long it takes to be adopted.
But if we label the dots by animal (see the image on the right), we notice *every data point* after 25 pounds belongs to a dog from the shelter!



Investigate

Divide the class into groups of 3-4, with one student identified as the "reporter".

- Looking at this scatterplot, does it make sense to analyze all the animals together? Why or why not?
- Are there some questions where it would be important to break up the population into species-specific populations? What are they?
- Are there some questions where it would be important to keep the whole population together? What are they?

Synthesize

Have the reporters share their findings with the class.
Imagine that you've been handed a dataset from a country where half the people are wealthy and have access to amazing medical care, and the other half are poor and have no healthcare. If we took a random sample of the population as a whole, we might think that they are generally middle-income and have average health. But if we ask the same question about the two groups *separately*, we would discover inequality hiding in plain sight!

Grouped Samples

20 minutes

Launch

Ultimately, it might make more sense to certain questions about "just the cats" or "just the dogs". Averaging every animal together will give us an answer, but it may not be a *useful* answer.

Sometimes important facts about samples get *lost* if we mix them with the rest of the population!

Data Scientists make ***grouped samples*** of datasets, breaking them up into sub-groups that may be helpful in their analysis.

Investigate

A "kitten" is an animal who *is a cat* and who *is young*. How would you make a subset of just kittens?

- Turn to Grouped Samples from the Animals Dataset (Page 41), and see what code will compute whether or not an animal is a kitten.
- Can you fill in the code for the other subsets?
- When you're done, type these definitions into the Definitions Area

- When you're done, type these definitions into the Definitions Area.

We already know how to define values, and how to filter a dataset. So let's put those skills together to define one of our subsets:

```
dogs  = animals-table.filter(is-dog)
```

- Define the other subsets, and click "Run".
- Make a pie chart showing the species in the `young` subset, by typing `pie-chart(young, "species")`.
- Make pie charts for every grouped sample. Which one is the most representative of the whole population? Why?

Synthesize

Debrief with students. Thoughtful question: how could we filter *and* sort a table? How can we combine methods?

Displaying Samples

20 minutes

Overview

Students revisit the data display activity, now using the samples they created.

Launch

Making grouped and random samples is a powerful skill to have, which allows us to dig deeper than just making charts or asking questions about a whole dataset. Now that we know how to make subsets, we can make much more sophisticated displays!

Investigate

Complete Displaying Data (Page 42), using what you've learned about samples to make more sophisticated data displays.

Synthesize

Were any of the students' displays interesting or surprising? Given a novel question, can students identify what helper functions they would need to write?

Displaying Data

Fill in the tables below, then use Pyret to make the following displays. Record the code you used.
The first table has been filled in for you.

1) A `bar-chart` showing how many puppies are fixed or not.

What Rows?	Which Column(s)?	What Display?
puppies	*fixed*	*bar-chart*

code: `bar-chart(puppies, "fixed")`

2) A `pie-chart` showing how many heavy dogs are fixed or not.

What Rows?	Which Column(s)?	What Display?

code: _____

3) A `histogram` of the number of `weeks` it takes for a random sample of animals to be adopted.

What Rows?	Which Column(s)?	What Display?

code: _____

4) A `box-plot` of the number of `pounds` that kittens weigh.

What Rows?	Which Column(s)?	What Display?

code: _____

5) A `scatter-plot` of a random sample using `name` as the labels, `age` as the x-axis, and `weeks` as the y-axis.

What Rows?	Which Column(s)?	What Display?

code: _____

6) A `scatter-plot` of fixed cats, using `species` as the labels, `pounds` as the x-axis, and `weeks` as the y-axis.

What Rows?	Which Column(s)?	What Display?

code: _____

What's on your mind?

Choosing Your Dataset

When selecting a dataset to explore, *pick something that matters to you!* You'll be working with this data for a while, so you don't want to pick something at random just to get it done.

When choosing a dataset, it's a good idea to consider a few factors:

1. Is it **interesting**? This should be data you are curious about, that answers questions you'd want to ask. Pick a dataset you're genuinely interested in, so that you can explore questions that matter to you!
2. Is it **relevant**? Does this data impact you in any way? Are there questions you have about the dataset that mean something to you or someone you know? Pick a dataset that deals with something personally relevant to you!
3. Is it **familiar**? You wouldn't be able to make samples of the Animals Dataset properly if you didn't know that some animals are much bigger or longer-lived than others. Pick a dataset you know about, so you can use your expertise to deepen your analysis!

Choosing Your Dataset

Students summarize their dataset by exploring the data and identifying categorical and quantitative columns, datatypes, and more. They also define a few sample rows, random subsets, and logical subsets.

Prerequisites	Grouped Samples
Relevant Standards OK K12CS CSTA NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Explain why they chose their dataset • Describe their dataset • Make subsets from their dataset
Student-facing Lesson Goals	• Let's all choose an interesting dataset to investigate.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column, random-rows`	

The Data Cycle

20 minutes

Overview

Students learn about the *Data Cycle*, which helps them get situated in the process of analyzing the datasets they will select in this lesson. They browse through the library of provided datasets, and choose one they want to work with. NOTE: the selection process can also be done as a homework assignment, if all students have internet access at home.

selection process can also be done as a homework assignment, if all students have internet access at home.

Launch

Zoom out a little and help students reflect on what they've done so far. Students began by exploring the Animals Dataset, formulating questions and exploring them with data displays. This led to further questions, making subsets, and asking more questions.

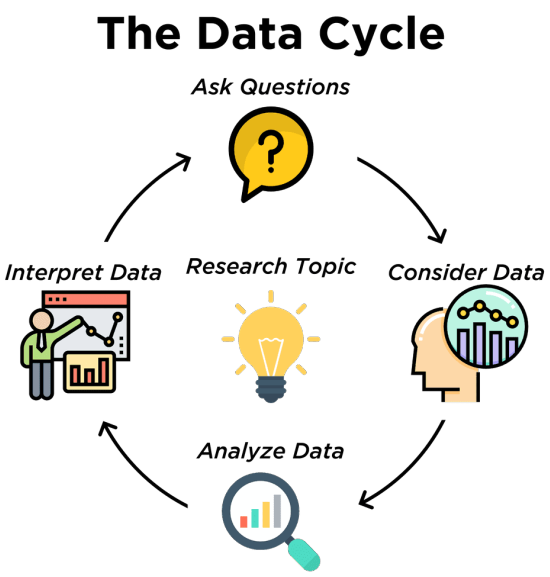
The Data Cycle[*] is a *roadmap*, which helps guide us in the process of data analysis.

(Step 1) We start by **Asking Questions** - statistical questions that can be answered with data.

(Step 2) Then we **Consider Data**. This could be done by conducting a survey, observing and recording data, or finding a dataset that meets our needs.

(Step 3) Then it's on to **Analyzing the Data**, in which we produce data displays and new tables of filtered or transformed data in order to identify patterns and relationships.

(Step 4) Finally, we **Interpret the Data**, in which we answer our questions and summarize the results. As we've already seen from the Animals Dataset, these interpretations often lead to *new* questions....and the cycle begins again.



Explain to students that they will now select a dataset for them to work with for the remainder of the course. Make sure they understand that it genuinely has to be something they are interested in - their engagement with the data is critical to engaging with the class.

Students can also find their own dataset, and use this (Blank Starter file). See this tutorial video for help importing your own data into Pyret.

Students must have at least 2 questions that are both *interesting* and *answerable* using their dataset.

Investigate

Choose a dataset that is interesting to you! You should have at least two questions that the dataset can help you answer. Write these questions down on What's on your mind? (Page 49).

Movies	[Dataset Starter File]
Schools	[Dataset Starter File]
US Income	[Dataset Starter File]
US Presidents	[Dataset Starter File]
Countries of the World	[Dataset Starter File]
Music	[Dataset Starter File]
NYC Restaurant Health Inspections	[Dataset Starter File]
Pokemon Characters	[Dataset Starter File]
IGN Video Game Reviews	[Dataset Starter File]
2016 Presidential Primary Election	[Dataset Starter File]
US Cancer Rates	[Dataset Starter File]
US State Demographics	[Dataset Starter File]
Sodas	[Dataset Starter File]
Cereals	[Dataset Starter File]
Summer Olympic Medals	[Dataset Starter File]
Winter Olympic Medals	[Dataset Starter File]
MLB Hitting Stats	[Dataset Starter File]
Spotify Top Songs	[Dataset Starter File]

Open the Research Paper template, and save a copy.

- Students fill in their first and last name(s), the teacher name on the first page of the Research Paper.
- Students should also copy the link to the dataset (spreadsheet), and paste it into the first page of the Research Paper.
- Students should click "Publish" in their Pyret Starter File, then copy/paste the resulting link into the first page of the Research Paper.

Synthesize

Have students share their datasets and their questions.

For the rest of this course, students will be learning new programming and Data Science skills, practicing them with the Animals Dataset and then applying them to their own data.

Exploring Your Dataset

flexible

Overview

Students apply what they've learned about describing and making subsets from the Animals Dataset to their own dataset.

Note: this activity can be done briefly as a homework assignment, but we recommend giving students an *additional class period* to work on this.

Launch

By now you've already learned what to do when you approach a new dataset. With the Animals Dataset, you first read the data itself, and wrote down your Notice and Wonders. You described the columns in the Animals Dataset, identifying which were categorical and which were quantitative, and whether they were Numbers, Strings, Booleans, etc. Finally, you used the Design Recipe and table methods to make random and logical subsets.

Now, you're doing to do the same thing *with your own dataset*.

Investigate

- Have students look at the spreadsheet for their dataset. What do they **Notice**? What do they **Wonder**? Have them complete My Dataset (Page 45), making sure to have at least two Lookup Questions, two Compute Questions, and two Relate Questions.
- In the Definitions Area, students use `random-rows` to define **at least three** tables of different sizes: `tiny-sample`, `small-sample`, and `medium-sample`.
- In the Definitions Area, students use `.row-n` to define **at least three** values, representing different rows in your table.
- Have students think about subsets that might be useful for their dataset. Name these subsets and write the Pyret code to test an individual row from your dataset on Samples from My Dataset (Page 46).
- Students should fill in My Dataset portion of their Research Paper.
- Students should fill in Categorical Visualizations portion of their Research Paper, by generating pie and bar charts for their dataset and explaining what they show.

Turn to The Design Recipe (Page 47), and use the Design Recipe to write the filter functions that you planned out on Samples from My Dataset (Page 46). When the teacher has checked your work, type them into the Definitions Area and use the `.filter` method to define your new sample tables.

Choose one categorical column from your dataset, and try making a bar or pie-chart for the whole table. Now try making the same display for each of your subsets. Which is most representative of the entire column in the table?

Synthesize

Have students share which subsets they created for their datasets.

[*] From the Mobilizing IDS project and GAISE

My Dataset

I chose to work with the _____ dataset, which contains _____ data rows.

What do you NOTICE?	What do you WONDER?	*Question Type*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*
		Lookup *Compute* *Relate* *Can't answer*

Some of the columns are:

1) _____, which contains _____ data. Some example values from this column are:

_____.

2) _____, which contains _____ data. Some example values from this column are:

_____.

Samples from My Dataset

How can we define grouped samples? For a given row `r`, what function will identify if that row is in the sample?

Subset	A function that returns true if a row `r` is in the subset

	`fun _____(r):`
	
	`end`

	`fun _____(r):`
	
	`end`

	`fun _____(r):`
	
	`end`

	`fun _____(r):`
	
	`end`

	`fun _____(r):`
	
	`end`

	`fun _____(r):`
	
	`end`

The Design Recipe

Write helper functions for **your** dataset, which you can use to define subsets.

Directions: Define a function called _____, which consumes a Row of the _____ table and produces _____.

Contract and Purpose Statement

Every contract has three parts...

_____ :: Row -> Boolean
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Directions: Define a function called _____, which consumes a Row of the _____ table and produces _____.

Contract and Purpose Statement

Every contract has three parts...

_____ :: Row -> Boolean
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

The Design Recipe

Write your own word problems below, and solve them using the Design Recipe.

Directions: Define a function called _____, which consumes a Row of the _____ table and produces _____.

Contract and Purpose Statement

Every contract has three parts...

_____ :: Row -> Boolean

function name *domain* *range*

_____

what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____

function name *input(s)* *what the function produces*

_____ (_____) **is** _____

function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):

function name *variable(s)*

_____

what the function does with those variable(s)

end

Directions: Define a function called _____, which consumes a Row of the _____ table and produces _____.

Contract and Purpose Statement

Every contract has three parts...

_____ :: Row -> Boolean

function name *domain* *range*

_____

what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____

function name *input(s)* *what the function produces*

_____ (_____) **is** _____

function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):

function name *variable(s)*

_____

what the function does with those variable(s)

end

What's on your mind?

Histograms

To best understand histograms, it's helpful to contrast them first with bar charts.

Bar charts show the number of rows belonging to a given category. The more rows in each category, the taller the bar.

- *Bar charts provide a visual representation of the frequency of values in a **categorical** column.*
- There's no strict numerical way to order these bars, but **sometimes there's an order** that makes sense. For example, bars for the sales of different t-shirt sizes might be presented in order of smallest to largest shirt.

Histograms show the number of rows that fall within certain intervals, or "bins", on a horizontal axis. The more rows that fall within a particular "bin", the taller the bar.

- *Histograms provide a visual representation of the frequencies (or relative frequencies) of values in a **quantitative** column.*
- Quantitative data **can always be ordered**, so the bars of a histogram always progress from smallest (on the left) to largest (on the right).
- When dealing with histograms, it's important to select a good **bin size**. If the bins are too small or too large, it is difficult to see the shape of the dataset. Choosing a good bin size can take some trial and error!

The **shape** of a data set tells us which values are more or less common.

- In a **symmetric** data set, values are just as likely to occur a certain distance above the mean as below the mean.
- A data set that is **skewed left** and/or has low outliers has a few values that are unusually low. The histogram for a skewed left dataset has a few data points that are stretched out to the left (lower) end of the x-axis.
- A data set that is **skewed right** and/or high outliers means there are a few values that are unusually high. The histogram for a skewed right dataset has a few data points that are stretched out to the right (higher) end of the x-axis.
- One way to visualize the difference between a histogram of data that is **skewed left** or **skewed right** is to think about the lengths of our toes on our left and right feet. Much like a histogram that is "skewed left", our left feet have smaller toes on the left and a bigger toe on the right. Our right feet have the big toe on the left and smaller toes on the right, more closely resembling the shape of a histogram of "skewed right" data.

Histograms

Students explore new visualizations in Pyret, this time focusing on the distribution in a quantitative dataset. Students are introduced to Histograms by comparing them to bar charts, and learn to construct them by hand and in Pyret.

Prerequisites	Defining Table Functions
Relevant Standards K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • create histograms using the Animals Dataset • visualizations of frequency using their chosen dataset, and write up their findings
Student-facing Lesson Goals	• Let's use histograms to talk about data.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized`	
Table	`count, .row-n, .order-by, .filter, .build-column, random-rows`	

Glossary

bar chart :: a display of categorical data that uses bars positioned over category values; each bar's height reflects the count or percentage of data values in that category

frequency :: how often a particular value appears in a data set

histogram :: a display of quantitative data that uses vertical bars positioned over bins (sub-intervals); each bar's height reflects the count or percentage of data values in that bin.

shape :: The aspect of a dataset that tells which values are more or less common

Review

20 minutes

Have students open their Animals Starter File, and click "Run". (If they do not have this file, or if something has happened to it, they can always make a new copy.)

- Turn to The Design Recipe (Page 51), and write the functions you see there. When you're ready, type the contracts, purpose statements, examples and definitions into the Definitions Area.
- Use the `.build-column` method to add a new column to the animals table, showing the weight of every animal in kilograms.
- Use the `image-scatter-plot` function to plot all of the animals, putting`age` on the x-axis, number of `weeks` in the shelter on the y-axis, and `smart-dot` as our function.

Introducing Histograms

20 minutes

Overview

Students look at a bar chart and a histogram, compare/contrast them, and make observations about what they have in common and how they are different. Then they learn a more formal explanation of histograms.

Launch

Have students complete Summarizing Columns (Page 52).

The display on the left side of that page is a ***Bar chart***.

- The x-axis lists the values of a categorical variable (`species`).
- The y-axis shows the ***frequency*** of categorical values in the dataset.
- This chart happens to show the categorical values in alphabetical order from left to right, but it would be fine to re-order them any way we wish. The bar for "dogs" could have been drawn before the one for "cats", without changing the meaning of the display. *It never makes sense to talk about the "shape" of a categorical data set* , since that shape holds no meaning.

The display on the right side is called a ***histogram***.

- Histograms show the distribution of *quantitative* data.
- Since quantitative data must follow a natural order, these bars *cannot* be re-ordered.
- Histograms allow us to see the shape of a data set.

Investigate

To build a histogram, we start by sorting all of the numbers in our column from smallest to largest, marking our x-axis from the smallest value (or a bit below) to the largest value (or a bit above) and dividing into equally-sized intervals, or "bins". For example, if our values ranged from 3 to 53 we might mark our x-axis from 0 to 60 and divide it into bins of width 10. If they range from 22 to 41 we might mark our x-axis from 20 to 45 and divide it into bins of width 5. Once we have our bins, we put each value in our dataset into the bin where it belongs, and then count how many values fall in each bin. This count determines the height of the bars on our y-axis.

Turn to Making Histograms (Page 53), and try drawing a histogram from a dataset.

Possible Misconceptions

Note that intervals on this display include the left endpoint but not the right. If we included the right endpoint and someone had 0 teeth, we'd have to add on a bar from -5 to 0, which would be awfully strange!

Synthesize

Review: How are histograms and bar charts different?

Choosing the Right Bin Size

15 minutes

Overview

Students make histograms from the animals-dataset, and explore different bin sizes.

Launch

The size of the bins matters a lot! Bins that are too small will hide the shape of the data by breaking it into too many short bars. Bins that are too large will hide the shape by squeezing the data into just a few tall bars. In this workbook exercise, the bins were provided for you. But how do you choose a good bin-size?

Investigate

A display of how long it takes animals to get adopted can make it easier to get an idea of what adoption times were most common, and if there were any unusually long or short times that it took for an animal to be adopted.

Suppose we want to know how long it takes for animals from the shelter to be adopted.

- Find the contract for the `histogram` function.
- Make a histogram for the `"weeks"` column in the `animals-table`, using a bin size of 10.
- How many took between 0 and 10 weeks? Between 10 and 20?
- Try some other bin sizes (be sure to experiment with bigger and smaller bins!) - what shapes emerge? What bin size gives you the best picture of the distribution?

Look at the histogram and count how many animals took between 0 and 5 weeks to be adopted. How many took between 5 and 10 weeks? What else do you Notice? What do you Wonder?

Some observations you can share with the class, to get them started:

- We see most of the histogram's area under the two bars between 0 and 10 weeks, so we can say it was most common for an animal to be adopted in 10 weeks or less.
- We see a small amount of the histogram's area trailing out to unusually high values, so we can say that a couple of animals took an unusually long time to be adopted: one took even more than 30 weeks.
- More than half of the animals (17 out of 31) took just 5 weeks or less to be adopted. But those few unusually long adoption times pulled the average up to 5.8 weeks. We'll talk more about Shape of a histogram in the next lesson, and about its effect on average (the mean) in the lesson after that.

If someone asked what was a typical adoption time, we could say: "Almost all of the animals were adopted in 10 weeks or less, but a couple of animals took an unusually long time to be adopted—even more than 20 or 30 weeks!" Without looking at the histogram's shape, we could not have drawn this conclusion.

What would the histogram look like if most of the animals took more than 20 weeks to be adopted, but a couple of them were adopted in fewer than 5 weeks?

Synthesize

Have students talk about the bin sizes the tried. Encourage open discussion as much as possible here, so that students can make their own meaning about bin sizes before moving on to the next point.

Rule of thumb: a histogram should have between 5–10 bins.

Histograms are a powerful way to display a data set and assess its ***shape***. Choosing the right bin size for a column has a lot to do with how data is distributed between the smallest and largest values in that column! With the right bin size, we can see the *shape* of a quantitative column. But how do we talk about or describe that shape, and what does the shape actually tell is? The next lesson addresses all of these.

The Design Recipe

For the word problems below, assume you have `animalA` and `animalB` defined in your code.

Directions: Define a function called `kilos`, which consumes a Row of the animals table and divides the pounds column by 2.2 to *compute* the animal's weight in kilograms.

Contract and Purpose Statement

Every contract has three parts...

```
# _____ :: (r :: Row) -> _____
```
function name *domain* *range*

```
# _____
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

```
_____ ( _____ ) is _____
```
function name *input(s)* *what the function produces*

```
_____ ( _____ ) is _____
```
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun _____ ( _____ ):
```
function name *variable(s)*

```
  _____
```
what the function does with those variable(s)

end

Directions: Define a function called `smart-dot`, which consumes a Row of the animals table and *computes* the image of a solid red circle using the animal's `pounds` as the radius.

Contract and Purpose Statement

Every contract has three parts...

```
# _____ :: _____ -> Image
```
function name *domain* *range*

```
# Consumes an animal, and computes a solid red circle using the weight in pounds as the radius
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

```
smart-dot ( "animalA" ) is _____
```
function name *input(s)* *what the function produces*

```
_____ ( _____ ) is _____
```
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun _____ ( _____ ):
```
function name *variable(s)*

```
  _____
```
what the function does with those variable(s)

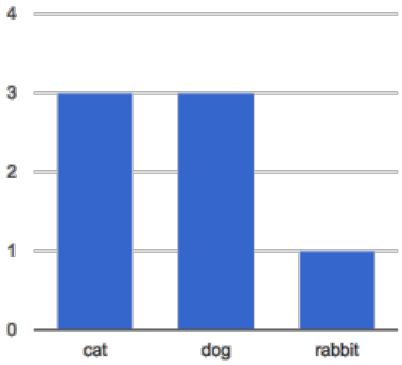
end

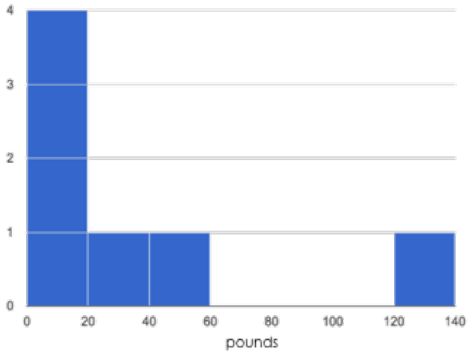
Summarizing Columns

name	species	age	pounds
"Sasha"	"cat"	1	6.5
"Boo-boo"	"dog"	11	123
"Felix"	"cat"	16	9.2
"Nori"	"dog"	6	35.3
"Wade"	"cat"	1	3.2
"Nibblet"	"rabbit"	6	4.3
"Maple"	"dog"	3	51.6

1	How many cats are there in the table above?	
2	How many dogs are there?	
3	How many animals weigh between 0-20 pounds?	
4	How many animals weigh between 20-40 pounds?	
5	Are there more animals weighing 40-60 than 60-140 pounds?	

The charts below are both based on this table. What is similar about them? What is different?





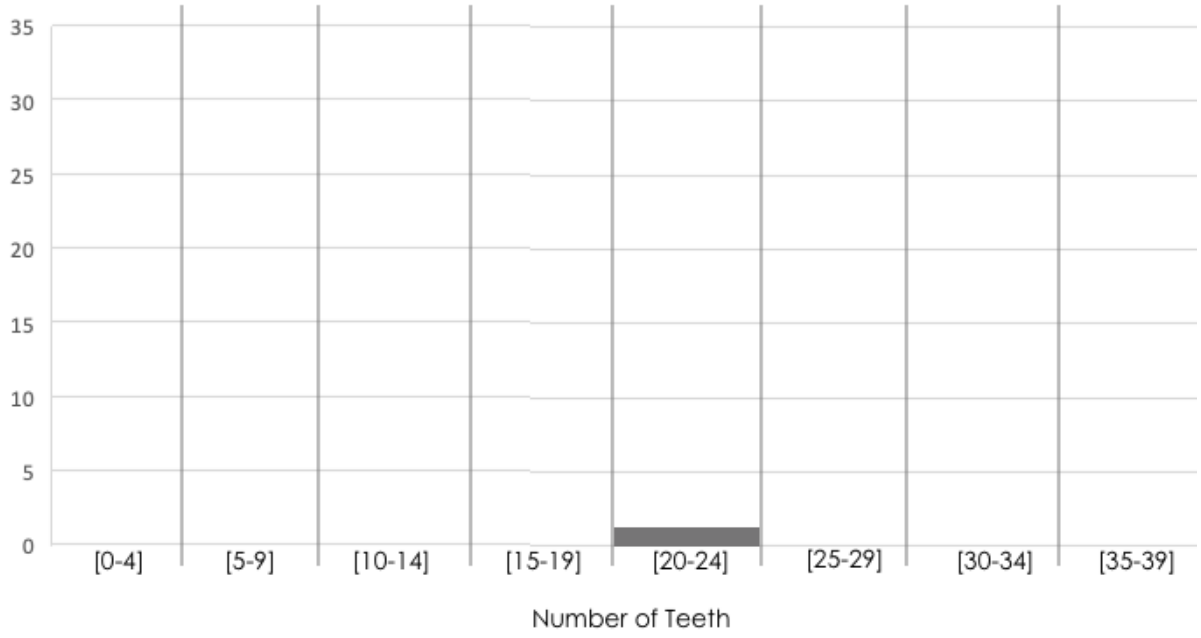
Similarities	Differences

Making Histograms

Suppose we have a data set for a group of 50 adults, showing the number of teeth each person has:

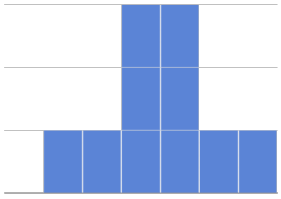
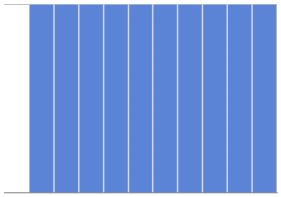
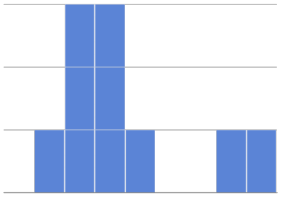
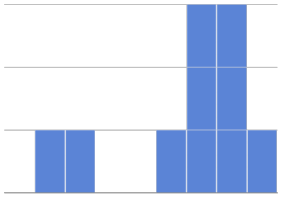
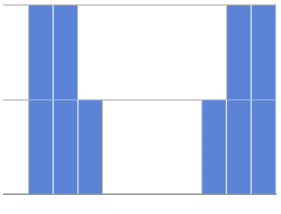
Number of teeth	Count
0	5
22	1
26	1
27	1
28	4
29	3
30	5
31	3
32	27

Draw a histogram for the table in the space below. For each row, find which interval (or "bin") on the x-axis represents the right number of teeth. Then fill in the box so that the height of the box is equal to the *sum of the counts* that fit into that interval. One of the intervals has been completed for you.



Reading Histograms

Students watched 5 videos, and rated them on a scale of 1 to 10. While the **average score** for every video is the same (5.5), the **shapes** of the ratings distributions were very different! *Match* the summary description (left) with the *shape* of the histogram of student ratings (right). For each histogram, **the x-axis is the score, and the y-axis is the number of students who gave it that score**. These axes are intentionally unlabeled - focusing on the *shape* is what matters here!

Most of the students were fine with the video, but a couple of them gave it an unusually low rating.	**1**	**A**	
Most of the students were okay with the video, but a couple students gave it an unusually high rating.	**2**	**B**	
Students tended to give the video an average rating, and they weren't likely to stray far from the average.	**3**	**C**	
Students either really liked or really disliked the video.	**4**	**D**	
Reactions to the video were all over the place: high ratings and low ratings and inbetween ratings were all equally likely.	**5**	**E**	

Visualizing the "Shape" of Data

Students explore the concept of "shape", using histograms to determine whether a dataset has skewness, and what the direction of the skewness means. They apply this knowledge to the Animals Dataset, and then to their own.

Prerequisites	Choosing Your Dataset
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Create histograms for variables in the Animals Dataset • Create visualizations of frequency using their chosen dataset, and write up their findings
Student-facing Lesson Goals	• Let's investigate what the shape of a histogram can tell us about the data.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram`	
Table	`count, .row-n, order-by, .filter, .build-column, random-rows`	

Glossary

shape :: The aspect of a dataset that tells which values are more or less common

skewed left :: A distribution is skewed left if there are a few values that are fairly low compared to the bulk of data values. A display of the data will show a longer tail to the left.

skewed right :: A distribution is skewed right if there are a few values that are fairly high compared to the bulk of data values. A display of the data will show a longer tail to the right.

symmetric :: A symmetric distribution has a balanced shape, showing that it's just as likely for the variable to take lower values as higher values.

Review

15 minutes

Have students turn to Reading Histograms (Page 54), and complete the matching activity there.

Describing Shape

20 minutes

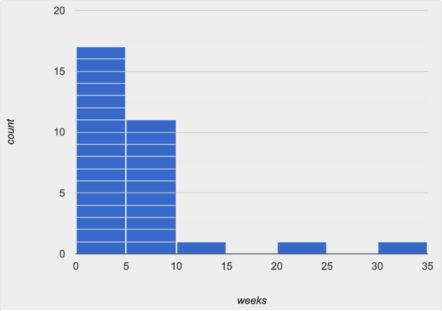
Overview

This activity focuses on *describing shape* based on a histogram. Students learn about "left skewed", "right skewed", and "symmetric" data, and what those descriptions tell us about a dataset.

Launch

Shape is one way to *summarize* information in a dataset, to quickly describe what values are more or less common.

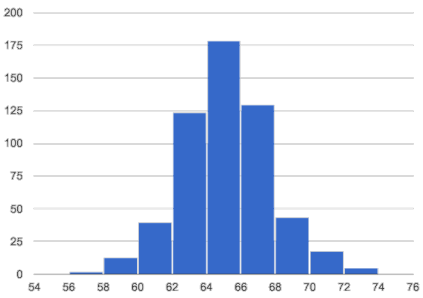
Consider the image on the right: most of the data points are clustered on the left side, and it contains a few unusually high values way off to the right. We might describe this histogram by saying that it is *"skewed right, or has high outliers."*



Here are the most common shapes that we see for real-world data sets:

Symmetric: values are balanced on either side of the middle.

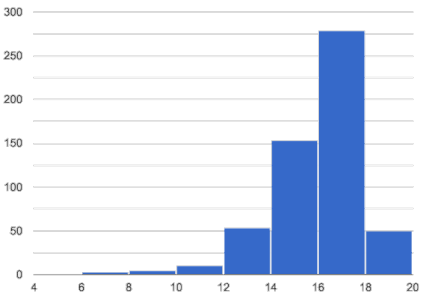
In a ***symmetric*** distribution, it's just as likely for the variable to take a value a certain distance below the middle as it is to take a value that same distance above the middle. Examples:



- Heights of 12-year-olds would have a symmetric shape. It's just as likely for a 12-year-old to be a certain number of inches below average height as it is to be that number of inches above average height.
- In a standardized test, most students score fairly close to what's average. Also, we see just as many students scoring a certain number of points above average as we see scoring that same number of points below average. The shape is symmetric (and bulges in the middle because most students score fairly close to what's average).

Skewed left, or low outliers.

In a distribution that is ***skewed left***, values are clumped around what's typical, but they trail off to the left with a few unusually low values. Examples:

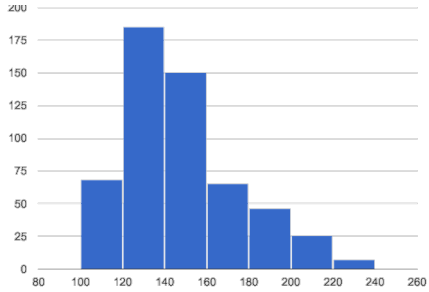


- Number of teeth that adults have in their mouths would be skewed left or have low outliers. Most adults will have close to a full set of 32 teeth, but a few of them with serious dental problems would have a very small number of teeth. We won't get anyone in our data set who has 10 or 20 *extra* teeth in their mouths!
- If most students did pretty well on an exam, but a few students performed very badly, then we'd see a shape that has left skewness and/or low outliers.

Skewed right, or high outliers.

In a distribution that is ***skewed right***, values are clumped around what's typical

In a distribution that is **skewed right**, values are clumped around what's typical, but they trail off to the right with a few unusually high values. We see this shape often in the real world, because there are many variables—like "income" or "time spent on the phone"—for which a few individuals have unusually high values, which aren't balanced out by unusually low values (things like "income" and "phone time" can't be less than zero). Examples:



- Age when a woman in the U.S. gives birth would be skewed right or have high outliers. A few women would be unusually old (40+ years), above the average age of 26 (check the tabloids!), but none of them could be even close to 40 years below average to balance things out!
- A data set of earnings almost always shows right skewness or high outliers, because there are usually a few values that are so far above average, they can't be balanced out by any values that are so far below average. (Earnings can't be negative.)

Investigate

- Make a histogram for the pounds column in the animals table, sorting the animals into 20-pound bins:
- Would you describe the shape of your histogram as being skewed left, skewed right, or symmetric?
- Which one of these statements is justified by the histogram's shape?
  1. A few of the animals were unusually light.
  2. A few of the animals were unusually heavy.
  3. It was just as likely for an animal to be a certain amount below or above average weight.
- Try bins of 1-pound intervals, then 100-pound intervals. Which of these three histograms best satisfies our rule of thumb?
- On Identifying Shape (Page 55), describe the shape of the histograms you see there.
- On The Shape of the Animals Dataset (Page 56), describe the pounds histogram and another one you make yourself. When writing down what you notice, try to use the language Data Scientists use, discussing both skew and outliers.

Challenge Questions: - Compare histograms for the `pounds` column of both cats and dogs in the dataset. Are their shapes different? How much overlap is there? - Compare histograms for the `age` column of both cats and dogs in the dataset. Are their shapes different? How much overlap is there? - Can you explain why the amount of overlap between these two distributions different?

Synthesize

Discuss as a class, making sure students agree on the description of the shape.

Your Analysis

flexible

Overview

Students repeat the previous activity, this time applying it to their own dataset and interpreting their own results. **Note: this activity can be done briefly as a homework assignment, but we recommend giving students an *additional class period* to work on this.**

Launch

Now it's time to try looking at the shape of your own dataset! Pick one quantitative column in your dataset, and hypothesize whether you think it will be skewed right, skewed left, or symmetric. What do you think?

Investigate

- How is your dataset distributed? Choose two quantitative variables and display them with histograms. Explain what you learn by looking at these displays. If you're looking at a particular subset of the data, make sure you write that up in your findings on The Shape of My Dataset (Page 57).

- Students should fill in the Quantitative Visualizations portion of their Research Paper, using histograms they've constructed for their dataset and explaining what they show.

Synthesize

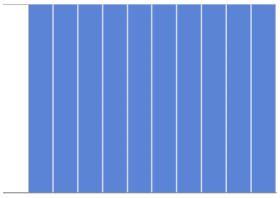
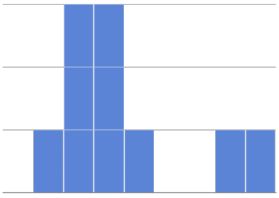
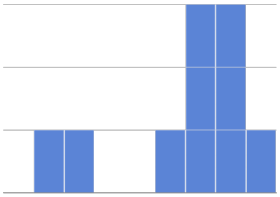
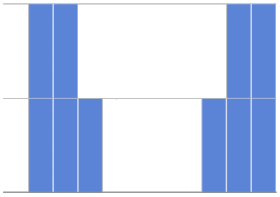
Have students share their findings.

Closing

Histograms are a powerful way to display a data set and see its ***shape***. But shape is just one of three key aspects that tell us what's going on with a quantitative data set. In the next unit, we'll explore the other two: center and spread.

Identifying Shape

Describe the shape of histograms on the left in complete sentences, using vocabulary like "Skewed Left", "Skewed Right", or "Symmetric".

1		
2		
3		
4		
5		

The Shape of the Animals Dataset

Describe two histograms made from columns of the animals dataset.

1) Make a histogram, showing the distribution of pounds (column in your dataset) for animals from the shelter (your subset, e.g., "fixed dogs from the shelter").

2) Make another histogram, showing the distribution of _____ (column in your dataset) for _____ (your subset, e.g., "fixed dogs from the shelter").

3) What do you Notice and Wonder about these two histograms? What shape do they have?

What do you NOTICE?	What do you WONDER?

The Shape of My Dataset

Describe two of the histograms you made from your dataset.

1) I made a histogram, showing the distribution of _____ for
column in your dataset

_____.
your subset, e.g., "fixed dogs from the shelter"

2) I made a histogram, showing the distribution of _____ for
column in your dataset

_____.
your subset, e.g., "fixed dogs from the shelter"

3) In the table below, describe the histograms. **Are they symmetric?** Do they show left skewness and/or low outliers? ** Do they show Right skewness and/or high outliers?

What do you NOTICE about these displays?	What do you WONDER about these displays?

What's on your mind?

Measures of Center and Spread

There are three ways to measure the **center** of a dataset, to summarize a whole column of quantitative data using just one number:

- The **mean** of a dataset is the average of all the numbers.
- The **median** of a dataset is a value that is smaller than half the dataset, and larger than the other half. In an ordered list the median will either be the middle number or the average of the two middle numbers.
- The **mode(s)** of a data set is the value (or values) occurring most often. When all of the values occur equally often, a dataset has no mode.

In a **symmetric** dataset, values are just as likely to occur a certain distance above the mean as below the mean, and the median and mean are usually close together.

When a dataset is asymmetric, the median is a more decriptive measure of center than the median.

- A dataset with **left skew**, and/or low outliers, has a few values that are unusually low, pulling the mean *below* the median.
- A dataset with **right skew**, and/or high outliers, means there are a few values that are unusually high, pulling the mean *above* the median.

When a dataset contains a small number of values, the mode may be the most descriptive measure of center.

Data Scientists can also measure the **spread** of a dataset using a **five-number summary**:

- The **minimum** – the lowest value in the dataset
- The **first, or "lower" quartile (Q1)** – the middle of the lower half of values, which separates the lowest quarter from the next smallest quarter
- The **second quartile (Q2)** – the middle value, which separates the entire dataset into "top" and "bottom" halves
- The **third, or "upper" quartile (Q3)** – the middle of the higher half of values which separates the second highest quarter from the highest quarter
- The **maximum** – the largest value in the dataset

Measures of Center

Students learn different ways to report the center of a quantitative data set: mean, median and mode(s). After applying these concepts to a contrived dataset, they apply them to their own datasets and interpret the results.

Prerequisites	Visualizing the "Shape" of Data
Relevant Standards OK K12CS NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Students explore the concept of center of a distribution, learning how to compute the mean, median and mode(s) of a dataset • Students find the mean, median and mode(s) of various columns in the Animals table
Student-facing Lesson Goals	• Let's use mean, median, and mode to describe our data.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	

Language Table

Types	Functions	Values
Number	`num-sqrt, num-sqr`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

mean :: average, calculated as the sum of values divided by the number of values

median :: the middle element of a quantitative data set

mode :: the most commonly appearing categorical or quantitative value or values in a data set

outlier :: a data point that is unusually far above or below most of the others

skew :: lack of balance in a dataset's shape, arising from more values that are unusually low or high. Such values tend to trail off, rather than be separated by a gap (as with outliers).

Do Now

Open your workbooks to Summarizing Columns in the Animals Dataset (Page 61). We've already filled in the answer to Question 1 for you (`pounds`). In your animals starter file, make a `box-plot` for the `pounds` column, and fill in the 5-number summary for that column in your workbook.

Mean

15 minutes

Overview

Students learn about mean (or "average"), and how it is one way (among others!) to summarize a quantitative column.

Launch

According to the Animal Shelter Bureau, the average pet weighs almost 41 pounds.

Some medicines are dosed by weight: heavier animals need a larger dose. If someone from the shelter needs to give a dose of medicine to the animals, is the "average" the best estimate we can use?

"The average pet weighs 41 pounds" is a statement about the entire dataset, which summarizes a whole column of values with a single number. Summarizing a big dataset means that some information gets lost, so it's important to pick an appropriate summary. Picking the wrong summary can have serious implications! Here are just a few examples of summary data being used for important things. Do you think these summaries are appropriate or not?

- Students are sometimes summarized by two numbers — their GPA and SAT scores — which can impact where they go to college or how much financial aid they get.
- Schools are sometimes summarized by a few numbers — student pass rates and attendance, for example — which can determine whether or not a school gets shut down.
- Adults are often summarized by a single number — like their credit score — which determines their ability to get a job or a home loan.
- When buying uniforms for a sports team, a coach might look for the most common size that the players wear.

Can you think of other examples where someone uses a number or two to summarize something complex?

Every kind of summary has situations in which it does a good job of reporting what's typical, and others where it doesn't really do justice to the data. In fact, the shape of the data can play a huge role in whether or not one kind of summary is appropriate!

One of the ways that Data Scientists summarize quantitative data is by talking about its *center* - literally asking "what is a typical value in this sample?", in the hopes of inferring something about a larger population. But there are many different ways to define "center", and each method has strengths and weaknesses. Let's check the "41 pounds" claim and see if it's an appropriate measure of center. Later on, you'll have a chance to apply what you've learned to your own dataset, to find the best way to provide an overall summary of the data.

Investigate

Open your "Animals Starter File". (If you do not have this file, or if something has happened to it, you can always make a new copy.)

If we plotted all the pounds values as points on a number line, what could we say about the average of those values? Is there a midpoint? Is there a point that shows up most often? Each of these are different ways of "measuring center".

The Animal Shelter Bureau used one method of summary, called the ***mean***, or "average". In general, the mean of a data set is the sum of values divided by the number of values. To take the average of a column, we add all the numbers in that column and divide by the number of rows.

Pyret has a way for us to compute the mean of any quantitative column in a Table. It consumes a Table and the name of the column you want to measure, and produces the mean — or average — of the numbers in that column.

```
# mean :: (t :: Table, col :: String) -> Number
```

What is its name? Domain? Range?

Notice that calculating the mean requires being able to add and divide, so the mean only makes sense for quantitative data. For example, the mean of a list of Presidents doesn't make sense. Same thing for a list of zip codes: even though we can divide a sum of zip codes, the output doesn't correspond to some "center" zip code.

Type `mean(animals-table, "pounds")` . What does this give us? Does this support the Bureau's claims?

Open your workbooks to Summarizing Columns in the Animals Dataset (Page 61). Under the "measures of center" section, fill in the computed mean.

Median

15 minutes

Overview

Students learn a second measure of center: the ***median***. They learn the algorithm and the code to find the median, as well as situations where taking the median is more appropriate than the mean.

Launch

You computed the mean of that column to be almost exactly 41 pounds. That IS the average, but if we scan the dataset we'll quickly see that most of the animals weigh less than 41 pounds! In fact, more than half of the animals weigh less than just 15 pounds. What is throwing off the average so much?

Kujo and Mr. Peanutbutter!

In this case, the mean is being thrown off by a few extreme data points. These extreme points are called ***outliers***, because they fall far outside of the rest of the dataset. Calculating the mean is great when all the points are fairly balanced on either side of the middle, but it distorts things for datasets with extreme outliers. The mean may also be thrown off by the presence of ***skewness***: a lopsided shape due to values trailing off left or right of center.

Make a `histogram` of the `pounds` column, and try different bin sizes. Can you see the skew towards the right, with a huge number of animals clumped to the left?

A different way to measure center is to line up all of the data points—in order—and find a point in the center where half of the values are smaller and the other half are larger. This is the ***median***, or "middle" value of a list.

As an example, consider this list of ACT scores:

```
25, 26, 28, 28, 28, 29, 29, 30, 30, 31, 32
```

Here 29 is the median, because it separates the "bottom half" (5 values below it) from the top half" (5 values above it).

The algorithm for finding the median of a quantitative column is:

1. Sort the numbers (we did this for you in the above example).
2. Cross out the highest number.
3. Cross out the lowest number.
4. Repeat until there is only one number left. If there are two numbers left at the end, take the *mean* of those numbers.

Investigate

- Pyret has a function to compute the median of a list as well. Find the contract in your contracts page.
- Compute the median for the `pounds` column in the Animals Dataset, and add this to Summarizing Columns in the Animals Dataset (Page 61).
- Is it different than the mean?
- What can we conclude when the mean is so much greater than the median?
- For practice, compute the mean and median for the weeks and age columns.

Synthesize

By looking at the histogram, we can develop an intuition for whether it's probably better to use the mean or median. Pronounced left skewness and/or low outliers can pull the mean down below the median, while right skewness and/or high outliers can pull it up. Either way, such shapes distort the mean as a measure of what's typical for the data set. Data scientists generally prefer to use the mean as their measure of center, because it contains information from every single data value. However, if a data set has substantial skewness or outliers, they use median to report the center .

Modes

25 minutes

Overview

Students learn about the mode(s) of a dataset, how to compute the mode, and when it is appropriate to use this as a measure of center.

Launch

The third measure of center is called the ***mode*** of a dataset. The ***mode*** of a data set is the value that appears *most often* . Median and Mean always produce one number, but if two or more values are equally common, there can be more than one mode. If all values are equally common, then there is no mode at all! Often there will be just one mode in the list of most common values: many data sets are what we call "unimodal". But sometimes there are exceptions! Consider the following three datasets:

```
1, 2, 3, 4
1, 2, 2, 3, 4
1, 1, 2, 3, 4, 4
```

- The first dataset has *no mode at all!*
- The mode of the second data set is 2, since 2 appears more than any other number.
- The modes (plural!) of the last data set are 1 and 4, because 1 and 4 both appear more often than any other element, and because they appear equally often.

Mode is rarely used to summarize quantitative data. It is very common as a summary of *categorical* data, telling us which category occurs most often.

In Pyret, the mode(s) are calculated by the modes function, which consumes a Table and the name of the column you want to measure, and produces a *List* of Numbers.

```
# modes :: (t :: Table, col :: String) -> List<Number>
```

Investigate

Compute the `modes` of the `pounds` column, and add it to Summarizing Columns in the Animals Dataset (Page 61). What did you get?

Synthesize

The most common number of pounds an animal weighs is 6.5! That's well below our mean and even our median, which is further evidence of outliers or skewness.

At this point, we have a lot of evidence that suggests the Bureau's use of "mean" to summarize animal weights isn't ideal. Our mean weight agrees with their findings, but we have three reasons to suspect that ***mean*** isn't the best value to use:

- The median is only 13.4 pounds.
- The mode of our dataset is only 6.5 pounds, which suggests a cluster of animals that weigh less than one-sixth the mean.
- When viewed as a histogram, we can see the right skewness and high outliers in the dataset. Mean is sensitive to datasets with skewness and/or outliers.

Closing

The Animal Shelter Bureau started with a fact: the mean weight *is* about 41 pounds. But then they reported a conclusion without checking to see if that was the best summary statistic to look at. As Data Scientists, we had to look deeper into the data to find out whether or not to settle for the Bureau's summary. This is why using tools like histograms that show shape can be so important when deciding on a summary tool.

"In 2003, the average American family earned $43,000 a year—well above the poverty line! Therefore very few Americans were living in poverty."

Do you trust this statement? Why or why not? Consider how many policies or laws are informed by statistics like this! Knowing about measures of center helps us see through misleading statements.

You now have three different ways to measure center in a dataset. But how do you know which one to use? Depending on the shape of the dataset, a measure could be really useful or totally misleading! Here are some guidelines for when to use one measurement over the other:

- If the data is doesn't show much skewness or have outliers, ***mean*** is the best summary because it incorporates information from every value.
- If the data has noticeable outliers or skewness, ***median*** gives a better summary of center than the mean.
- If there are very few possible values, such as AP Scores (1–5), the ***mode*** could be a useful way to summarize the data set.

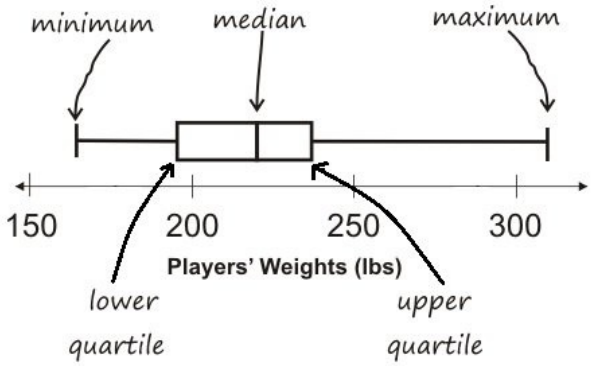
Exercises

Critiquing Written Findings

Measures of Center and Spread (continued)

The **five-number summary** can be used to draw a **box plot**.

- Each of the four sections of the box plot contains 25% of the data. *If the values are distributed evenly across the range, the four sections of the box plot will be equal in width.* Uneven distributions will show up as differently-sized sections of a box plot.
- The left **whisker** extends from the minimum to Q1.
- The **box**, or **interquartile range**, extends from Q1 to Q3. It is divided into 2 parts by the **median**. Each of those parts contains 25% of the data, so the whole box contains the central 50% of the data.
- The right **whisker** extends from Q3 to the maximum.



The box plot above, for example, tells us that:

- The minimum weight is about 165 pounds. The median weight is about 220 pounds. The maximum weight is about 310 pounds.
  - 1/4 of the players weigh roughly between 165 and 195 pounds
  - 1/4 of the players weigh roughly between 195 and 220 pounds
  - 1/4 of the players weigh roughly between 220 and 235 pounds
  - 1/4 of the players weigh roughly between 235 and 310 pounds
  - 50% of the players weigh roughly between 165 and 220 pounds
  - 50% of the players weigh roughly between 195 and 235 pounds
  - 50% of the players weigh roughly between 220 and 310 pounds
- The densest concentration of players' weights is between 220 and 235 pounds.
- Because the widest section of the box plot is between 235 and 310 pounds, we understand that the weights of the heaviest 25% fall across a wider span than the others. 310 may be an outlier, the weights of the players weighing between 235 pounds and 310 pound could be evenly distributed across the range, or all of the players weighing over 235 pounds may weigh around 310 pounds.

Summarizing Columns in the Animals Dataset

Find the measures of center and spread to summarize the pounds column of the Animals Table.

Be sure to add examples to your Contracts page as you work.

Measures of Center

The three measures of center for this column are:

Mean (Average)	Median	Mode(s)

Since the mean is _____ compared to the median, this suggests the shape is

[higher/lower/about equal]

_____.

[skewed right (or high outliers) / skewed left (or low outliers) / symmetric]

Measures of Spread

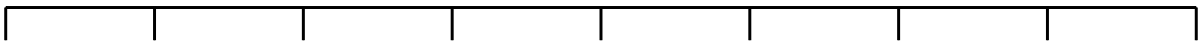
My five-number summary is:

Minimum	Q1	Median	Q3	Maximum

Displaying Center and Spread with a Box Plot

Draw a box plot from this summary on the number line below.

Be sure to label the number line with consistent intervals.



From this summary and box plot, I conclude:

Interpreting Spread

Consider the following dataset, representing the annual income of ten people.

All numbers represent *thousands of dollars* (so `14` means "$14,000"):

```
60, 10, 21, 180, 14, 20, 45, 35, 45, 170
```

1) In the space below, rewrite this dataset in **sorted order**.

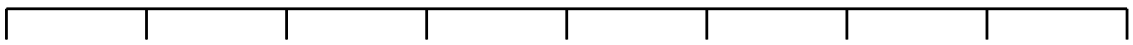
2) In the table below, compute the **measures of center** for this dataset.

Mean (Average)	Median	Mode(s)

3) In the table below, compute the **five number summary** of this dataset.

Minimum	Q1	Q2 (Median)	Q3	Maximum

4) On the number line below, draw a **box plot** for this dataset.



5) The following statements are *correct* ... but misleading. Write down the reason why.

Statement	Why it's misleading
"They're rich! The average person makes more than $70k dollars!"	
"It's a middle-income list: the most common salary is $45k/yr!"	
"This group is really diverse, with people making as little as $14,000 and as much as $280,000!"	

Spread of a Data Set

Students learn how to evaluate the spread of a quantitative column using box plots, and explore how this offers a different perspective on shape from what can be achieved with a histogram. After applying these concepts to a contrived dataset, they apply them to their own datasets and interpret the results.

Prerequisites	Measures of Center
Relevant Standards K12CS CSTA CC-Math NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • apply one approach to measuring and displaying spread of a data set • compare and contrast information displayed in a box plot and a histogram
Student-facing Lesson Goals	• Let's compare different uses for box plots and histograms when talking about data.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

box plot :: the box plot (a.k.a. box-and whisker-plot) is a way of displaying a distribution of data based on the five-number summary: minimum, first quartile, median, third quartile, and maximum

interquartile range :: (IQR) is one possible measure of spread, based on dividing a data set into four parts. The values that divide each part are called the first quartile (Q1), the median, and third quartile (Q3). IQR is calculated as Q3 minus Q1.

median :: the middle element of a quantitative data set

quartiles :: three values that divide a data set into four equal-sized groups

range of a data set :: the distance between minimum and maximum values

shape :: The aspect of a dataset that tells which values are more or less common
spread :: the extent to which values in a data set vary, either from one another or from the center

---

Measures of Spread

30 minutes

Overview

Students are introduced to the notion of *spread* in a dataset. They learn about quartiles, box plots, and how to use them to talk about spread.

Launch

A teacher may report that her students averaged a 75 on a test, but it's important to know how those scores were spread out: did all of them get exactly 75, or did half score 100 and the other half 50? When Data Scientists use the mean of a sample to estimate the mean of a whole population, it's important to know the spread in order to report how good or bad a job that estimate does.

Suppose we lined up all of the values in the pounds column of the animals data set from smallest to largest, and then split the line up into two equal groups by taking the ***median***. We can learn something about the ***spread*** of the data set by taking things further: The middle of the lighter half of animals is called the first ***quartile*** - or "Q1" - and the middle of the heavier half of animals is the third quartile (also called "Q3"). Once we find these numbers, we can say that the middle half of the animals' weights are spread between Q1 and Q3.

The first quartile (Q1) is the value for which 25% of the animals weighed that amount or less. What does the third quartile represent?

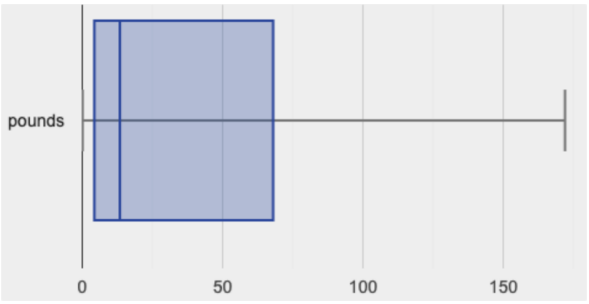
Besides looking at the median as center, and the spread between Q1 and Q3, we also gain valuable information from the spread of the entire data set—that is, the distance between minimum and maximum. This is called the ***range of a data set***. We can use ***box plots*** to visualize all of this information. These plots are constructed using **just five numbers**, which makes them convenient ways to display both center and spread of a data set in a clear and simple way. Below is the contract for `box-plot`, along with an example that will make a box plot for the `pounds` column in the `animals-table`.

```
# box-plot :: (t :: Table, column :: String) -> Image
box-plot(animals-table, "pounds")
```

Box plots divide our sample into equally-sized groups, and show where those groups are spread thin or clumped together.

Type in this expression in the Interactions Area, and see the resulting plot.

This plot shows us the center and spread in our dataset according to those five numbers.



- The **minimum** value in the dataset (at the left of "whisker"). In our dataset, that's just 0.1 pounds.
- The **First Quartile (Q1)** (the left edge of the box), is computed by taking *the median of the lower half of the values*. In the pounds column, that's 4.3 pounds.
- The **Median** value (the line in the middle), which is the middle Quartile of the whole dataset. We already computed this to be 13.4 pounds.
- The **Third Quartile (Q3)** (the right edge of the box), which is computed by taking *the median of the upper half of the values*. That's 68 pounds in our dataset.
- The **maximum** value in the dataset (at the right of the "whisker"). In our dataset, that's 172 pounds.

One way to summarize the spread in the dataset is to measure the distance between the largest value and the smallest value. When we talk about functions having many possible outputs, we use the term "Range" to describe them. (**Note**: the

term "Range" means something different in statistics than it does in algebra and programming!) When we look at the distance between the smallest and largest values in our dataset, we use the same term.

Extension Activity

In statistics, it is not uncommon to use *modified box plots*, which remove extreme datapoints from the box-and-whisker and draw them as dots outside of the blot. The box plot then represents only the "non-extreme" points. Modified box plots are also available in Bootstrap:Data Science, using the following contract:

```
# modified-box-plot :: (t :: Table, column :: String) -> Image
```

Investigate

- Turn to Summarizing Columns in the Animals Dataset (Page 61)
- Fill in the five-number summary for the `pounds` column, and sketch the box plot.
- What conclusions can you draw about the distribution of values in this column?

Data Scientists subtract the 1st quartile from the 3rd quartile to compute the range of the "middle half" of the dataset, also called the ***interquartile range***.

- Find the ***interquartile range*** of this dataset.
- What percentage of animals fall within the interquartile range?
- What percentage of animals fall below the First Quartile? Above the Third Quartile? What percentage fall anywhere between the minimum and the maximum?

Now that you're comfortable creating box plots and looking at measures of spread on the computer, it's time to put your skills to the test!

Turn to Interpreting Spread (Page 62) and complete the questions you see there.

Just as pie and bar charts are ways of visualizing categorical data, box plots and histograms are both ways of visualizing the shape of quantitative data. Box plots make it easy to see the 5-number summary, and compare the Range and Interquartile Range. Histograms make it easier to see skewness and more details of the shape, and offer more granularity when using smaller bins.

Left-skewness is seen as a long tail in a histogram. In a box plot, it's seen as a longer left "whisker" or more spread in the left part of the box. Likewise, right skewness is shown as a longer right "whisker" or more spread in the right part of the box.

Box plots and Histograms can both tell us a lot about the shape of a dataset, but they do so by grouping data quite differently. A box plot is always divided into four parts, which may fall on differently-sized intervals but all contain the same number of points. A histogram, on the other hand, has identically-sized intervals which can contain very different numbers of points.

Turn to Identifying Shape (Page 63) and see if you can describe box plots using what you know about skewness.

Challenge Questions: - Compare the for the `pounds` column of both cats and dogs in the dataset. Are their shapes different? How much overlap is there? - Compare histograms for the `age` column of both cats and dogs in the dataset. Are their shapes different? How much overlap is there? - Can you explain why the amount of overlap between these two distributions is different?

Possible Misconceptions

It is extremely common for students to forget that every quartile *always* includes 25% of the dataset. This will need to be heavily reinforced.

Synthesize

Histograms, box plots, and measures of center and spread are all different ways to get at the ***shape*** of our data. It's important to get comfortable using every tool in the toolbox when discussing shape!

Modified Box Plots

More Statistics- or Math-oriented classes will also be familiar with *modified box plots*. These are similar to traditional box plots, but the box-and-whisker just extends to minimum and maximum non-outliers. To call our attention to outliers, they are drawn as small dots or asterisks at the extreme ends of the graph (watch a video on modified box plots). Pyret also has a `modified-box-plot` function, with the same Domain as `box-plot`.

Comparing Box Plots

15 minutes

Overview

Students assess the degree of visual overlap of two numerical distributions.

Launch

Mutiple box plots are extremely useful for showing us the answer to a particular kind of **Relate Question**, such as "Do dogs take longer to get adopted than cats?" This is really asking us about the interplay between a categorical variable (species) and a quantitative one (weeks to adoption). Instead of creating a whole new display tool, all we have to do is extend our usual box plot display so we can look at how the weeks distributions compare for cats and dogs. This works fine as long as we're sure to use a common scale: Note that both box plots in the display below share the same axis for adoption times, which ranges from about 1 to 10 weeks.

Box plots make it easy to decide if values of a quantitative variable seem to be fairly similar or quite different, depending on which group an individual is in. The trick is to train your eyes to look for whether there's a lot of overlap in the two box plots, or if one is noticeably higher than the other.

Investigate

Have students break into groups of 3-4, and compare the box plot of weeks-to-adoption for cats with the one for dogs.

Note: they can generate the pair of box plots themselves, but we recommend simply giving them this image: cats v. dogs

1. Do the two box plots mainly overlap, or is one noticeably higher than the other?
2. Roughly how do the medians compare?

Next, each group examines the pair of box plots that compare weeks to adoption for fixed versus unfixed animals: fixed v. unfixed. Once again, consider how similar or different the two plots seem.

1. Do the two box plots mainly overlap, or is one noticeably higher than the other?
2. Roughly how do the medians compare?

Students should confirm that the box plots for adoption times of unfixed versus fixed animals have more overlap than the box plots for adoption times of cats versus dogs.

Box plots and histograms give us two different views on the concept of shape. In a histogram, the intervals between the bars are fixed with different numbers of datapoints in each interval. A box plot is the exact opposite: the intervals are variable, with a fixed number of datapoints in each one. In a histogram, we can think of a datapoints that fall into bins, filling them up so we can see how many are in each. A box plot treats the data more like pizza dough, dividing it into four equal quarters and squeezing or stretching it to show where the data is tightly clumped or spread out over a long interval. To compare the two, complete Matching Box-Plots to Histograms (Page 65).

Synthesize

Referring to our first side-by-side box plots, the one for dogs' adoption times was much higher than the one for cats' adoption times; the top half of the dogs' box plot doesn't overlap at all with the cats' box plot. Does this suggest that species *does* or *does not* play a role in how long it takes for an animal to be adopted?

Referring to our second pair of box plots, we saw that adoption times for unfixed and fixed animals overlapped a lot, and the

medians (shown by the lines through the middle of each box) were pretty close: both a bit less than 4. Does this suggest that being fixed or not does or does not play a role in how long it takes for an animal to be adopted?
Which variable seems to have more of an effect on adoption time: species (cat or dog) or whether an animal is fixed or not?
Have students share back their findings.

Your Analysis

flexible

Overview

Students repeat the previous activity, this time applying it to their own dataset and interpreting their own results. **Note: this activity can be done briefly as a homework assignment, but we recommend giving students an *additional class period* to work on this.**

Investigate

- Take 15 minutes to fill out Shape of My Dataset (Page 64) in your Student Workbook. Choose a column to investigate, and write up your findings.
- Students should fill in Measures of Center and Spread portion of their Research Paper, using the means, medians, modes, box plots and five-number summaries they've constructed for their dataset and explaining what they show.

Synthesize

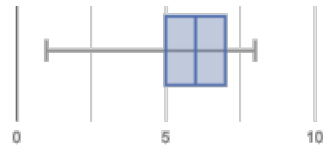
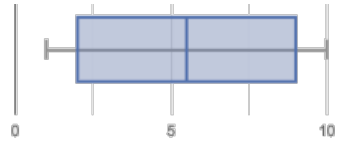
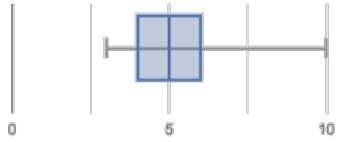
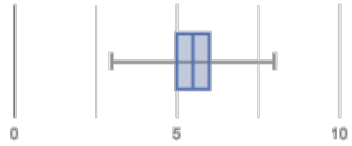
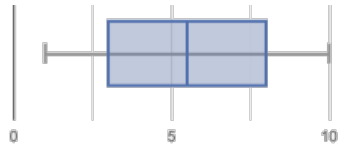
Have students share their findings with one another.

Additional Exercises:

- Project: Project: Stress or Chill? (You will also need the Personality True Colors assessment)

Identifying Shape

Describe the shape of the box plots below in complete sentences, using vocabulary like "Skewed Left", "Skewed Right", or "Symmetric".

1		
2		
3		
4		
5		

Shape of My Dataset

Find the measures of center and spread to summarize a column of your dataset.

The column I chose to summarize is _____.

Measures of Center

The three measures of center for this column are:

Mean (Average)	Median	Mode(s)

Since the mean is _____ compared to the median, this suggests the shape is
[higher/lower/about equal]

_____.
[skewed right (or high outliers) / skewed left (or low outliers) / symmetric]

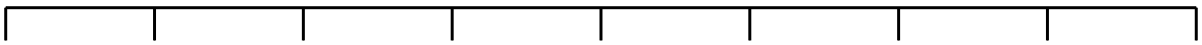
Measures of Spread

My five-number summary is:

Minimum	Q1	Q2 (Median)	Q3	Maximum

Displaying Center and Spread with a Box Plot

Draw a box plot from this summary on the number line below.
Be sure to label the number line with consistent intervals.



From this summary and box plot, I conclude:

_____

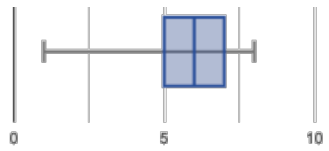
_____

_____

_____

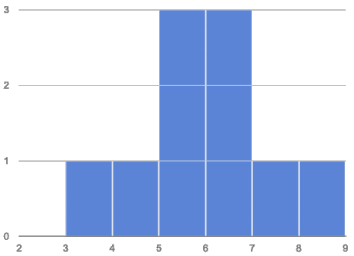
Matching Box-Plots to Histograms

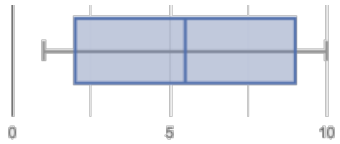
Students watched 5 videos, and rated them on a scale of 1 to 10. For each video, their ratings were used to generate box-plots and histograms. **Match the box-plot to the histogram that displays the same data.**



1

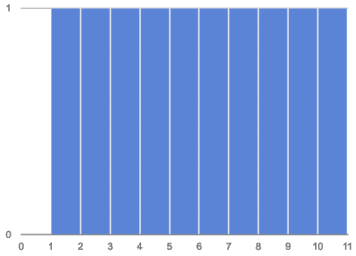
A

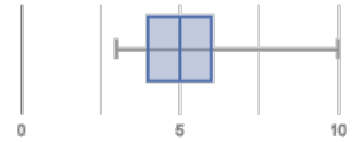




2

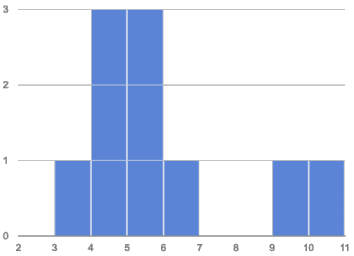
B

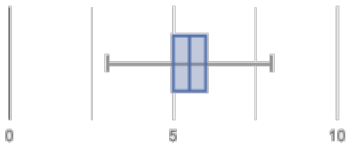




3

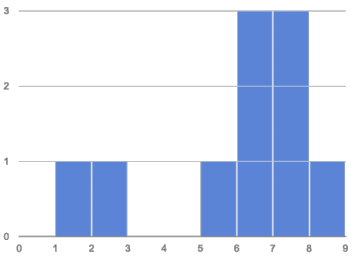
C

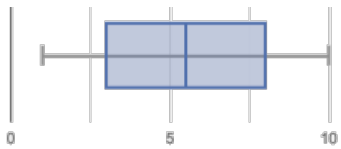




4

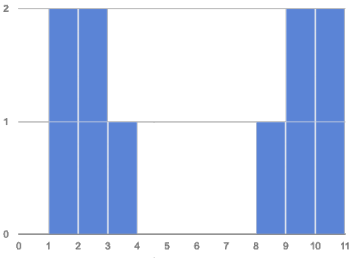
D





5

E



What's on your mind?

"Trust, but verify …"

A "helpful" Data Scientist gives you access to the following functions:

```
# fixed-cats :: (animals :: Table) -> Table
# consumes a table of animals, and produces a table containing only
# cats that have been fixed, sorted from youngest-to-oldest
```

You can use the function, *but you can't see the code for it!* **How do you know if you can trust their code?**

HINT:

- You could make a *verification subset* that contains one of every species, and make sure that the function filters out everything but cats.
- You could make sure this subset has multiple cats not already ordered of youngest-to-oldest, and make sure the function puts them in the right order.

1) What other qualities would this subset need to have?

2) Create your verification subset! In the space below, list the name of each animal in your subset.

Name

Checking Your Work

Students consider the concept of trust and testing — how do we know if a particular analysis is trustworthy?

Prerequisites	None
Relevant Standards OK K12CS CSTA NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... - Create a subset of data to verify that a given transformation works as-advertised, using attributes of the transformation and the dataset.
Student-facing Lesson Goals	• Let's learn how to test the trustworthiness of a data analysis.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • Make sure all students can access the Trust-but-Verify Starter File
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Confirming Analysis

30 minutes

Overview

Students learn how to create a *Testing Table* , which is small enough to reason about and can be used to test whether code does the right thing.

Launch

Samples are taken in Data Science and Computer Programming for two different reasons. One of the main purposes of Data Science is to take a representative sample from a larger population, and use information from the sample to infer what's true about the whole population. In programming, we often extract a smaller Table from a larger one, for the purpose of testing that our code seems to do what it's supposed to. In this lesson, we focus on the tasks of programmers, and consider

best practices for setting up a Testing Table that helps us check our code.

- Uber and Google are making self-driving cars, which use artificial intelligence to interpret sensor data and make predictions about whether a car should speed up, slow down, or slam on the brakes. This AI is trained on a lot of sample data, which it learns from. What might be the problem if the sample data only included roads in California?
- Law enforcement in many towns has started using facial-recognition software to automatically detect whether someone has a warrant out for their arrest. A lot of facial-recognition software, however, has been trained on sample data containing mostly white faces. As a result, it has gotten really good at telling white people apart, but often can't tell the difference between people who aren't white. Why might this be a problem?
- Why might it be a bad thing to only test medicines only on men (or only on women), before prescribing them to the general public?

Testing Matters!

A good Testing Table should be *representative* of the population, and *relevant* to what's being analyzed. A good Testing Table should have...

- *At least* the columns that matter — whether we'll be ordering or filtering by those columns.
- Enough rows to include different circumstances that are relevant to the task at hand. For instance, if our code is supposed to extract certain cats from the animals table, our Testing Table should include at least one animal that's not a cat.
- Rows that aren't already sorted, if our analysis is supposed to sort for us.

Data scientists usually think in terms of samples that best serve the purpose of *performing inference*: Samples should be representative of the entire population, and large enough to get us fairly close to the truth about that population. Computer programmers need to think in terms of *Testing Tables* that best serve the purpose of verifying that their code does what it's supposed to: The Tables should be designed to call attention to any imperfections in the code's instructions.

Investigate

Testing Tables can also be used to *verify* that a certain analysis is correct. A function that is supposed to filter a table and *show only the cats* can't be tested with a Testing Table that only has cats to begin with. How would we know if the function filters out non-cats?

Suppose a function takes in a table of animals and shows *only the kittens*. A Testing Table should have cats and non-cats, as well as old and young cats.

Suppose a function takes in a table of animals and shows only the kittens, sorted in ascending order by weight. Now a Testing Table should have cats and non-cats, as well as old and young cats... *and* have rows that aren't already sorted!

- Turn to "Trust, but verify ..." (Page 67) in your student workbook.
- You've been given a function called `fixed-cats` and a description of what it *claims* to do.
- List the names of the animals that you would use in a Testing Table to verify whether the function works as advertised. When you've finished, open the Trust-but-Verify Starter File. There are three versions of `fixed-cats` here. Are they all correct? If not, which ones are broken?
- Turn to "Trust, but verify..." (Page 68). Using the same Starter File, construct a Testing Table and figure out which (if any) of the functions are correct!

Synthesize

Complex analysis has more room for mistakes, so it's critical to think about a Testing Table that allows us to trust that our code really does what it's supposed to!

"Trust, but verify…"

A "helpful" Data Scientist gives you access to the following functions:

```
# old-dogs-nametags:: (animals :: Table) -> Table
# consumes a table of animals, and produces a table containing only
# dogs 5 years or older, with an extra column showing their name in red
```

You can use the function, *but you can't see the code for it!* **How do you know if you can trust their code?**

1) What qualities would a verification subset need to have?

2) Create your verification subset! In the space below, list the name and index of each animal in your subset.

Name

What's on your mind?

Scatter Plots

Scatter Plots can be used to show a relationship between two quantitative columns. Each row in the dataset is represented by a point, with one column providing the x-value and the other providing the y-value. The resulting "point cloud" makes it possible to look for a relationship between those two columns.

- If the points in a scatter plot appear to follow a straight line, it is possible that a linear relationship exists between those two columns. A number called a **correlation** can be used to summarize this relationship.
- r is the name of the **correlation statistic**. The r-value will always fall between −1 and +1. The sign tells us whether the correlation is positive or negative. Distance from 0 tells us the strength of the correlation.
  - −1 or +1 is really strong.
  - 0 means no correlation.
- The correlation is **positive** if the point cloud slopes up as it goes farther to the right. It is **negative** if it slopes down as it goes farther to the right. If the points are tightly clustered around a line, it is a **strong** correlation. If they are loosely scattered, it is a **weak** correlation.
- Points that are far above or below the cloud of points in a scatter plot are called **outliers**.
- We graphically summarize this relationship by drawing a straight line through the data cloud, so that the vertical distance between the line and each of the points is as small as possible. This line is called the **line of best fit** and allows us to predict y-values based on x-values.

Scatter Plots

Students investigate scatter plots as a method of visualizing the relationship between two quantitative variables.

Prerequisites	Displaying Categorical Data
Relevant Standards OK K12CS CSTA CC-Math NGSS	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • consider explanatory and response roles of variables • make scatter plots by hand, given a list of (x,y) pairs • make scatter plots using Pyret • identify a possible linear relationship by looking at a point cloud • Consider unusual observations in a scatter plot
Student-facing Lesson Goals	• Let's use Pyret to create scatter plots of data.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

explanatory variable :: the variable in a relationship that is presumed to impact the other variable

response variable :: the variable in a relationship that is presumed to be affected by the other variable

scatter plot :: a display of the relationship between two quantitative variables, graphing each explanatory value on the x axis and the accompanying response on the y axis

Relationships *Between* Columns 15 minutes

Overview

Students are finally introduced to *Relate Questions*, which ask about the relationship between one quantitative column and another.

Launch

Can animals' weights help explain why some are adopted quickly while others take a long time? What other factors explain why one pet gets adopted right away, and others wait months?

Theory 1: Smaller animals get adopted faster because they're easier to care for.

How could we test that theory? Bar and pie charts are great for showing us frequencies or percentages in a categorical column. Histograms and box plots are great for showing us the shape, center, and spread of a single quantitative column. But none of these displays will help us see connections between **two** quantitative columns.

Investigate

- Take a few minutes to look through the whole dataset, and see if you agree with Theory 1.
- Could any of our visualizations or summaries provide evidence for or against the theory?
- Write down your hypothesis on (Dis)Proving a Claim (Page 71), as well as a theory about how we could use this dataset to see if you're right.

Synthesize

We've got a lot of tools in our toolkit that help us think about an entire *column* of a dataset:

- We have ways to find measures of center and spread for a given quantitative column.
- We have visualizations that let us see the shape of values in a quantitative column.
- We have visualizations that let us see frequencies or percentages in a categorical column.

What columns is this question asking about?

Making Scatter Plots 20 minutes

Overview

Students are introduced to ***scatter plots***, which are visualizations tailored to Relate Questions about quantitative variables. They learn how to construct scatter plots by hand, and in Pyret.

Launch

This question is asking about *two columns* in our dataset. Specifically, it's asking **if there is a relationship** between `pounds` and `weeks`.

Before we can draw a ***scatter plot***, we have to make an important decision: which variable is ***explanatory*** and which is ***response***? In this case, are we suspecting that an animal's weight can explain how long it takes to be adopted, or that how long it takes to be adopted can explain how much an animal weighs?

The first of these makes sense, and reflects our suspicion that weight plays a role in adoption time. The convention is to use the horizontal axis for our explanatory variable and the vertical axis for the response. Thus, `pounds` will be x and `weeks` will be y.

Investigate

We will produce our scatter plot by graphing each animal's `pounds` and `weeks` values as a point on the x and y axes.

Complete Creating a Scatter Plot (Page 72) in your Student Workbook.

Teaching Tip

Divide the full table up into sub-lists, and have a few students plot 3-4 animals on the board. This can be done collaboratively, resulting in a whole-class scatterplot!

- Open your "Animals Starter File". (If you do not have this file, or if something has happened to it, you can always make a new copy.)
- Make a scatter plot that displays the relationship between weight and adoption time.
- Are there any patterns or trends that you see here?
- Try making a few other scatter plots, looking for relationships between other columns in the `animals-table`.

Synthesize

Have students share their observations. What trends do they see? Are there any points that seem unusual? Why?

Looking for Trends

20 minutes

Overview

Students are asked to identify patterns in their scatter plots. This activity builds towards the idea of *linear associations*, but does not go into depth (as the following lesson does).

Launch

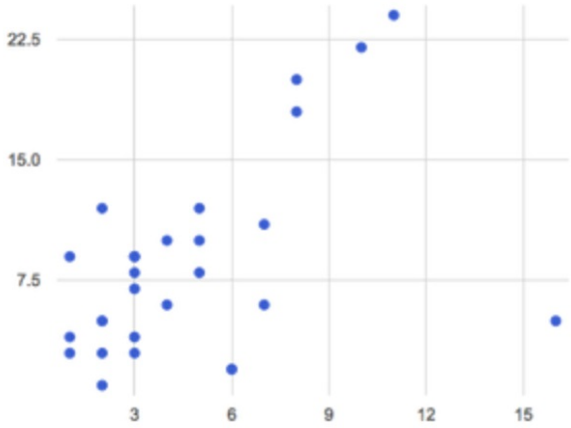
- Can you see a "cloud" around which the points are clustered?
- Does the number of weeks to adoption seem to go up or down as the weight increases?
- Are there any points that "stray from the pack"? Which ones?

Teaching Tip

Project the scatter plot at the front of the room, and have students come up to the plot to point out their patterns.

A straight-line pattern in the cloud of scatter plot points suggests that there is a linear relationship between two columns. If we can pinpoint a line around which the points cluster (as we'll do in a future lesson), it would be useful for making predictions. For example, our line might predict how many `weeks` a new dog would wait to be adopted, if the dog weighs 68 `pounds`.

Do any data points seem unusually far away from the main cloud of points? Which animals are those? These points are called **unusual observations**. Unusual observations in a scatter plot are like outliers in a histogram, but more complicated because it's the *combination* of x and y values that makes them stand apart from the rest of the cloud.



Unusual observations are *always* worth thinking about

- Sometimes they're just random. Felix seems to have been adopted quickly, considering how much he weighs. Maybe he just met the right family early, or maybe we find out he lives nearby, got lost and his family came to get him. In that case, we might need to do some deep thinking about whether or not it's appropriate to remove him from our dataset.
- Sometimes they can give you a deeper insight into your data. Maybe Felix is a special, popular (and heavy!) breed of cat, and we discover that our dataset is missing an important column for breed!

and we discover that our dataset is missing an important column for breed.

- Sometimes unusual observations are the points we are looking for! What if we wanted to know which restaurants are a good value, and which are rip-offs? We could make a scatter plot of restaurant reviews vs. prices, and look for an observation that's high above the rest of the points. That would be a restaurant whose reviews are *unusually good* for the price. An observation way below the cloud would be a really bad deal.

Investigate

For practice, try making scatter plots for each of the following relationships, always expressed as "response variable vs explanatory variable". If you see any **unusual observations**, try to explain them!

- The `pounds` of an animal vs its `age`
- The number of `weeks` for an animal to be adopted vs its number of `legs`
- The number of `legs` vs the `age` of an animal.
- Do you see a linear (straight-line) relationship in any of these, evidenced by a cloud of points that's clearly rising or falling from left to right? Are there any unusual observations?

Synthesize

Debrief, showing the plots on the board. Make sure students see plots for which there is no relationship, like the last one!

Theory 2: Younger animals get adopted faster because they are cuter.

It might be tempting to go straight into making a scatter plot to explore how weeks to adoption may be affected by age. But different animals have very different lifespans! A 5-year-old tarantula is still really young, while a 5-year-old rabbit is fully grown. With differences like this, it doesn't make sense to put them all on the same scatter plot. By mixing them together, we may be *hiding* a real relationship, or creating the illusion of a relationship that isn't really there! So it's probably best to make several displays, one for each species.

(Dis)Proving a Claim

Smaller animals get adopted faster because they're easier to care for."

Do you agree? If so, why?

I hypothesize …

What would you look for in the dataset to see if you are right?

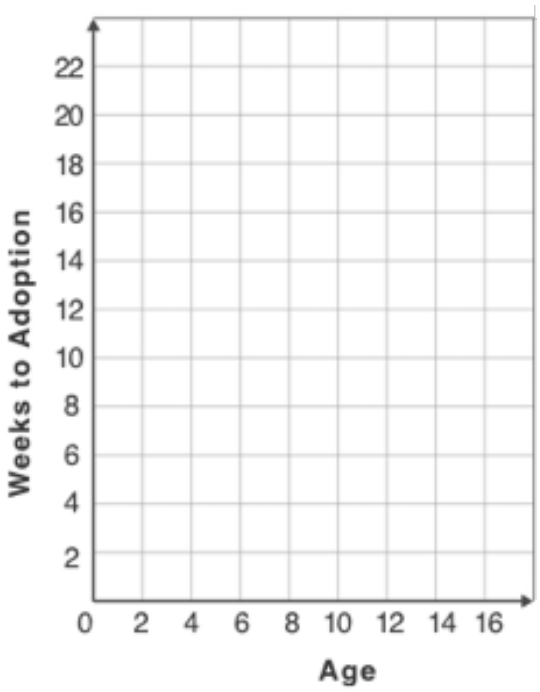
Creating a Scatter Plot

1. **For each row in the Sample Table on the left, add a point to the scatter plot on the right.** Use the values from the age column for the x-axis, and values from the weeks column for the y-axis.
2. Do you see a pattern? Do the points seem to go up or down as age increases to the right?
   - **Draw a line on the scatter plot to show this pattern.**
3. Does the line slope upwards or downwards?

   _____
4. Are the points tightly clustered around the line or loosely scattered?

   _____

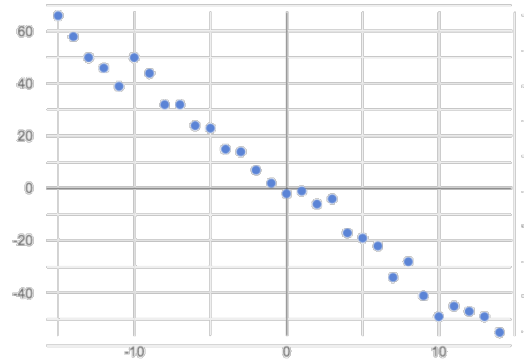
name	species	age	weeks
"Sasha"	"cat"	1	3
"Boo-boo"	"dog"	11	5
"Felix"	"cat"	16	4
"Buddy"	"lizard"	2	24
"Nori"	"dog"	6	9
"Wade"	"cat"	1	2
"Nibblet"	"rabbit"	6	12
"Maple"	"dog"	3	2



Identifying Form, Direction and Strength

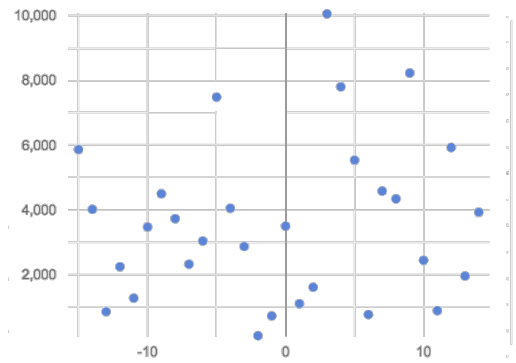
Can you identify the Form, Direction, & Strength of these displays? **Note:** If the form is non-linear, there is no direction!

A



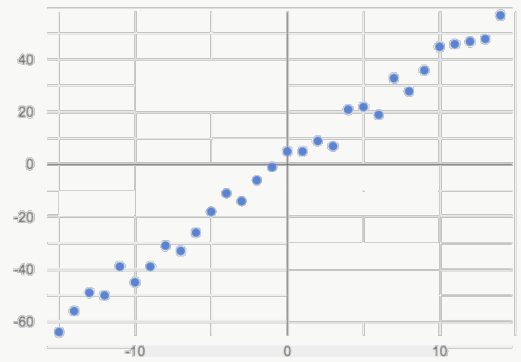
Form :	Linear	Non-Linear	None
Direction :	Positive	Negative	None
Strength :	Strong	Weak	None

B



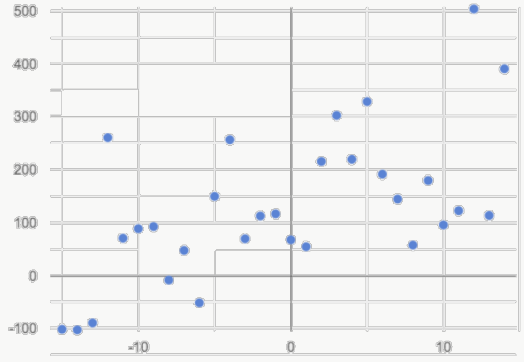
Form :	Linear	Non-Linear	None
Direction :	Positive	Negative	None
Strength :	Strong	Weak	None

C



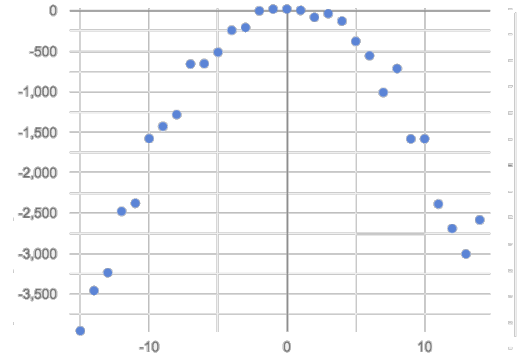
Form :	Linear	Non-Linear	None
Direction :	Positive	Negative	None
Strength :	Strong	Weak	None

D



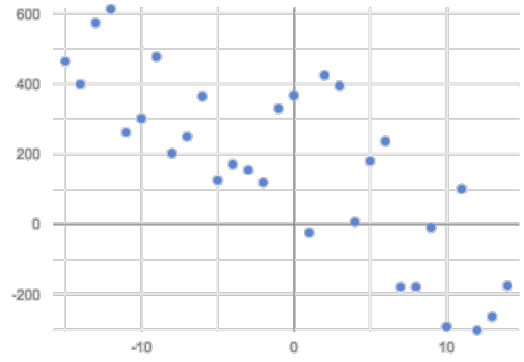
Form :	Linear	Non-Linear	None
Direction :	Positive	Negative	None
Strength :	Strong	Weak	None

E



Form :	Linear	Non-Linear	None
Direction :	Positive	Negative	None
Strength :	Strong	Weak	None

F



Form :	Linear	Non-Linear	None
Direction :	Positive	Negative	None
Strength :	Strong	Weak	None

Correlations

Students continue to interpret scatter plots, and think about direction and strength of linear relationships.

Prerequisites	None
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Confirm if a scatter plot appears linear • Understand how correlation measures direction in a linear relationship • Understand how correlation measures strength in a linear relationship
Student-facing Lesson Goals	• Let's explore scatter plots and what they can tell us about data relationships.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don't have the file, they can open a new one
Supplemental Resources	Spurious Correlations
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram, scatter-plot`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

form :: of a relationship between two quantitative variables: whether the two variables together vary linearly or in some other way

r :: a number between −1 and 1 that measures the direction and strength of a linear relationship between two quantitative variables (also known as correlation value)

Correlations have *Form* 10 minutes

Overview

Students identify and make use of patterns in scatter plots, learning to characterize them as being linear, curved, or showing no clear pattern. This builds intuition for determining if the ***form*** is linear, in which case we can proceed to correlation and linear regression

Launch

By now we have learned ways to summarize a single quantitative variable, like the `age` of an animal in our dataset: report the center, spread, and shape of the distribution. Together, those numbers tell us what age is typical, how much the ages vary, and what kind of age values are usual or unusual. We could do the same for `pounds`, `weeks`, or any other quantitative column.

But those individual summaries tell us nothing about the *relationship* between animals' ages and weights. In order to understand such relationships, we have to expand our view from a single dimension (along one axis) to two dimensions. This goes hand in hand with expanding our display from a one-dimensional histogram to a two-dimensional scatter plot.

Rather than summarizing each distribution in one dimension, we can summarize a *linear relationship* between two quantitative variables. But this only makes sense if the scatter plot follows a *straight-line pattern*, as opposed to being curved. So the very first assessment we have to make is to identify the ***form*** of the relationship as being linear or not.

Form: whether a relationship is linear or not

Investigate

The relationship between two quantitative variables can take many forms - some patterns are *linear*, and appear as a straight line sloping up or down. Some patterns are *non-linear*, and may look like a curve or an arc. And sometimes there is no pattern or relationship at all!

Have students turn to Identifying Form, Direction and Strength (Page 73) in their student workbooks. For each scatter plot, identify whether the relationship is linear, non-linear or if there's no relationship at all.

Synthesize

Data Scientists use their eyes all the time! It doesn't make sense to search for correlations when there's no pattern at all, and only linear relationships make sense if we want to summarize with a correlation.

Going Deeper

In an AP Statistics class or full-year Data Science class, it's appropriate to discuss non-linear relationships here. In a dedicated computer science class, it may also be appropriate to talk about *transforming* the x- or y-axis (using `.build-column`!) via a quadratic, exponential, or logarithmic function and then looking for a linear pattern in the resulting scatter plot. All of these are **extensions** to the materials presented here.

Correlations have *Direction & Strength* 20 minutes

Overview

Once students have learned to identify a possible linear relationship, they can turn their attention to other qualities of that relationship: its *direction* and *strength*. Each of these is expressed in the r-value, which students learn to read.

Launch

Assuming a relationship is linear, data scientists calculate a single number called "correlation" - or r-value - that reports both the direction and strength.

Direction: whether a linear relationship is positive or negative

Direction: whether a linear relationship is positive or negative.

A linear relationship between two quantitative variables is *positive* if, in general, the scatter plot points are sloping up: smaller x values tend to go with smaller y values, and larger x values tend to go with larger y values. The relationship is *negative* if points slope down: smaller x values tend to go with *larger* y values, and larger x values tend to go with *smaller* y values.

- **Positive** directions are by far more common because of natural tendencies for variables to increase in tandem. For example, "the older the animal, the more it tends to weigh". This is usually true for human animals, too!
- **Negative** relationships can also occur. For example, "the older a child gets, the fewer new words he or she learns each day."

Strength: how closely the two variables are correlated.

A relationship between two quantitative variables is strong if the scatter plot points are tightly clustered together. In this case, knowing the x-value of a data point gives us a very good idea of what its y-value will be. In other words, if the relationship is linear and strong, the scatter plot points are clumped together in a thin cloud.

- A **strong** linear relationship means that the points in the scatter plot are all clustered closely around an invisible line. If the cloud point is very tightly packed around the line, the relationship is said to be strong.
- A **weak** linear relationship means that the cloud of points is scattered very loosely around the line.

Investigate

Have students turn to Identifying Form, Direction and Strength (Page 73) in their student workbooks. For each scatter plot, identify whether the relationship is positive or negative, and whether it is strong or weak.

The correlation *r* is a number (between -1 and 1) that tells us the direction and strength of a linear relationship between two variables. r is positive or negative depending on whether the correlation is positive or negative. **The strength of a correlation is the distance from zero**: an r-value of zero means there is no correlation at all, and stronger correlations will be closer to −1 or 1.

An r-value of about ±0.65 or ±0.70 or more is typically considered a strong correlation, and anything between ±0.35 and ±0.65 is "moderately correlated". Anything less than about ±0.25 or ±0.35 may be considered weak. However, these cutoffs are not an exact science! In some contexts an r-value of ±0.50 might be considered impressively strong!

Calculating r from a data set only tells us the direction and strength of the relationship in *that particular sample*. If the correlation between adoption time and age for a representative sample of about 30 shelter animals turns out to be +0.44, the correlation for the larger population of animals will probably be *close* to that, but certainly not the same.

Have students turn to Identifying Form and r-Values (Page 74) in their student workbooks. For each scatter plot, identify whether the relationship linear, and use r to summarize direction and strength.

- In the Interactions Area, create a scatter plot for the Animals Dataset, using `"pounds"` as the xs and `"weeks"` as the ys.
- **Form:** Does the point cloud appear linear or non-linear?
- **Direction:** If it's linear, does it appear to go up or down as you move from left to right?
- **Strength:** Is the point cloud tightly packed, or loosely dispersed?
- Would you predict that the r-value is positive or negative? Will it be closer to zero, closer to ±1, or in between?
- Have Pyret compute the r-value, by typing `r-value(animals-table, "pounds", "weeks")`. Does this match your prediction?
- Repeat this process using `"age"` as the xs. Is this correlation stronger or weaker than the correlation for `"pounds"`? What does that *mean*?

Common Misconceptions

- Students often conflate strength and direction, thinking that a strong correlation *must* be positive and a weak one *must* be negative.
- Students may also falsely believe that there is ALWAYS a correlation between any two variables in their dataset.
- Students often believe that strength and sample size are interchangeable, leading to mistaken assumptions like "any

- Students often believe that strength and sample size are interchangeable, leading to mistaken assumptions like "any correlation found in a million data points *must* be strong!"

Synthesize

It is useful to ask students probing questions, to help address the misconceptions listed above. Some examples:

- What is the difference between a *weak* relationship and a *negative* relationship?
- What is the difference between a *strong* relationship and a *positive* relationship?
- If we find a strong relationship in a sample, can we always infer that relationship holds for the whole population?
- Suppose we have two correlations, one drawn from 10 data points and one drawn from 50. If both correlations are identical in direction and strength, should we trust them equally when making an inference about the larger population?

Correlation does NOT imply causation.

It's easy to be seduced by large r-values, and believe that we're really onto something that will help us make predictions! But Data Scientists know better than that...

Here are some real-life correlations that have absolutely no causal relationship; they come about either by chance or because both of them are tied in with another variable that's (often) lurking in the background.

- "Number of people who drowned after falling out of a fishing boat" v. "Marriage rate in Kentucky" (r = 0.98)
- "Average per-person consumption of chicken" v. "U.S. crude oil imports" (r = 0.95)
- "Marriage rate in Wyoming" v. "Domestic production of cars" (r = 0.99)
- "Number of people who get tangled in their own bedsheets" v. "Amount of cheese consumed that year" (r = 0.95)

All of these correlations come from the Spurious Correlations website. If time allows, have your students explore the site to see more!

Your Analysis

flexible

Overview

Students repeat the previous activity, this time applying it to their own dataset and interpreting their own results. **Note: this activity can be done as a homework assignment, but we recommend giving students an *additional class period* to work on this.**

Launch

What correlations do you think there are in your dataset? Would you like to investigate a subset of your data to find those correlations?

Investigate

- Brainstorm a few possible correlations that you might expect to find in your dataset, and make some scatter plots to investigate.
- Turn to Correlations in My Dataset (Page 75), and list three correlations you'd like to search for.
- Investigate these correlations. If you need blank Design Recipes, you can find them at the back of your workbook, just before the Contracts.
- What correlations did you find?
- Did you need to filter out certain rows in order to get those correlations?

Synthesize

Have students share back their correlations, and why they expect to find them.

After looking at the scatter plot for our animal shelter, do students still agree with the claim on (Dis)Proving a Claim (Page 71)? (Perhaps they need more information, or to see the analysis broken down separately by animal!)

Additional Exercises:

- Identifying Form, Direction and Strength (Matching)

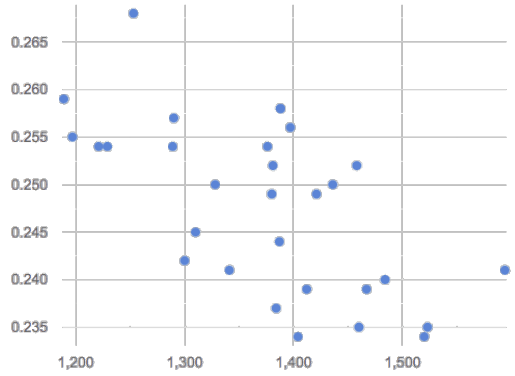
Identifying Form and r-Values

Can you identify the Form, Direction, and Strength of these displays?

If the form is linear, approximate the r-value to express Direction and Strength.

Reminder: An r-value close to -1 is a strong negative relationship, an r-value close to 0 is weak, and an r-value close to +1 is a strong positive!

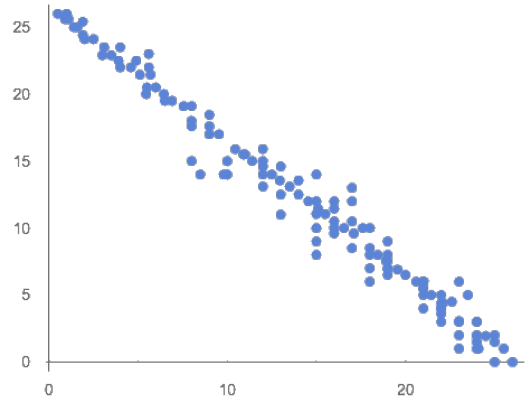
A



Form :

r close to :

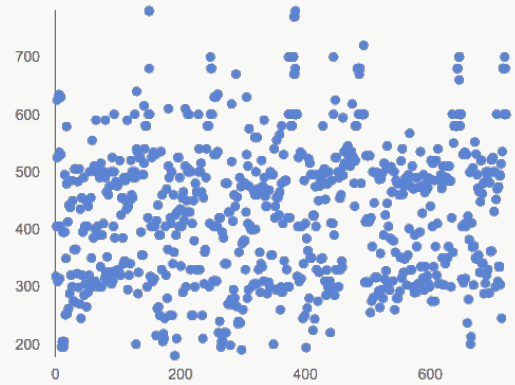
B



Form :

r close to :

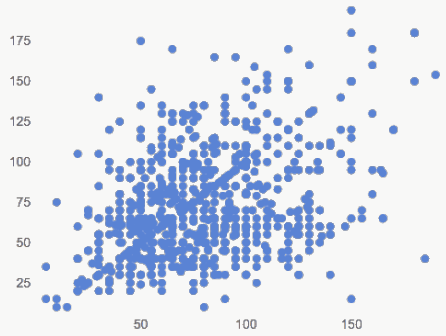
C



Form :

r close to :

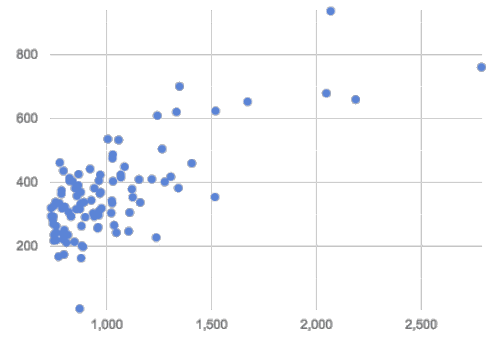
D



Form :

r close to :

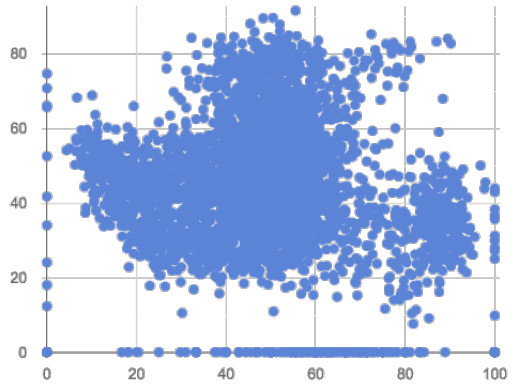
E



Form :

r close to :

F



Form :

r close to :

Correlations in My Dataset

1) There may be a correlation between _____ (column) and _____ (column).

I think it is a _____ (strong/weak), _____ (positive/negative) correlation,

because _____.

It might be stronger if I looked at _____ (a sample or extension of my data).

---

2) There may be a correlation between _____ (column) and _____ (column).

I think it is a _____ (strong/weak), _____ (positive/negative) correlation,

because _____.

It might be stronger if I looked at _____ (a sample or extension of my data).

---

3) There may be a correlation between _____ (column) and _____ (column).

I think it is a _____ (strong/weak), _____ (positive/negative) correlation,

because _____.

It might be stronger if I looked at _____ (a sample or extension of my data).

Computing Relationships

Linear Regression is a way of computing the **line of best fit** , which minimizes the sum of vertical distances of all scatter plot points from the line. Calculating the slope and intercept of this line is a task best left to computing or statistical software.

- **Slope** provides us with the easiest summary to grasp: it's how much we predict the y-variable (response variable) will increase or decrease for each unit that the x-variable (explanatory variable) increases.
- **Correlation is not causation!** Correlation only suggests that two column variables are related, but does not tell us if one causes the other. For example, hot days are correlated with people running their air conditioners, but air conditioners do not cause hot days!
- **Sample size matters!** The number of data values is also relevant. We'd be more convinced of a positive relationship in general between cat age and time to adoption if a correlation of +0.57 were based on 50 cats instead of 5.

Linear Regression

Students compute the “line of best fit” using linear regression, and summarize linear relationships in a dataset.

Prerequisites	None
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • interpret linear regression in the context of the animals table • use linear regression to quantify patterns in their chosen dataset, and write up their findings about ◦ the animal dataset ◦ their chosen dataset
Student-facing Lesson Goals	• Let’s learn how to determine the strength of data relationships.
Materials	• Lesson slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure students can access the Interactive LR Plot • Make sure all materials have been gathered • Decide how students will be grouped in pairs • All students should log into CPO and open the "Animals Starter File" they saved from the prior lesson. If they don’t have the file, they can open a new one
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram, scatter-plot`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

explanatory variable :: the variable in a relationship that is presumed to impact the other variable

line of best fit :: summarizes the relationship (if linear) between two quantitative variables

linear regression :: modeling the relationship between two quantitative variables using a straight line

predictor function :: a function which, given a value from one data set, makes an educated guess at a related value in a

predictor function :: a function which, given a value from one data set, makes an educated guess at a related value in a different data set

response variable :: the variable in a relationship that is presumed to be affected by the other variable

---

Warmup

Have students open their "Animals Dataset" files. (If they do not have this file, or if something has happened to it, they can always make a new copy.)

Make two scatterplots from the `animals-table`, using `age` as the explanatory variable in one plot and `pounds` as the ***explanatory variable*** in the other. In both plots, use `weeks` as your ***response variable*** and `name` for the `labels`. We will refer to the explanatory column as "xs" and the response column as "ys."

Intro to Linear Regression

10 minutes

Overview

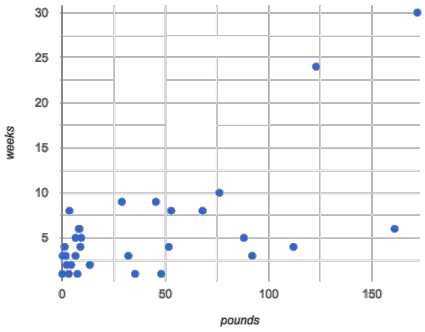
Students are introduced to the *concept* of linear regression, and learn how to interpret the slope and intercept. For teachers who have the need and the bandwidth to go deeper, this is a good opportunity to teach the algorithm behind linear regression.

Launch

"Can we *predict* an animal's adoption time based on its size? Its age?"

Have students write down what they think on What's on your mind? (Page 81), then quickly survey the class.

We are asking if we can use an animal's size or age to predict how long it will take to be adopted. A scatter plot of adoption time versus size does suggest that smaller animals get adopted in a shorter period of time and larger animals take longer. Similarly, younger animals tend to be adopted faster than older ones. Can we be more precise about this, and actually *predict* how long it will take an animal to be adopted, based on these factors? And which one would give us a better prediction?



The mean, median, and mode are three different ways to measure the "center" of a dataset in one dimension. Each represents a different way to collapse a bunch of points on a number line into a single, summary value. If the "center" of points on a *one dimensional* number line is a single point, what is the "center" of points in a *two-dimensional* cloud, which cluster around a line?

What we need to do is find a *line* — called a ***line of best fit***, or a ***regression line*** — that is at the center of this cloud. Each point in our scatter plot "pulls" on the line, with points above the line yanking it up and points below the line dragging it down. Points that are really far away — especially influential observations that are far out in the x direction —- pull on the line with more force. This line can be graphed on top of the scatter plot as a function, called the ***predictor function***.

Given a value on the x-axis, this line allows us to predict what the corresponding value on the y-axis might be. This allows us to make predictions based on our data.

Is there only one "best line"? Based on methods of calculus, data scientists know the answer to this question is yes! That justifies us talking about a single "line of best fit."

Data scientists use a statistical method called ***linear regression*** to pinpoint linear relationships in a dataset. When we draw our ***regression line*** on a scatter plot, we can imagine a rubber bands stretching vertically between the line itself and each point in the plot — every point pulls the line a little "up" or "down". Linear regression is the math behind the line of best fit.

Going Deeper

If you want to teach students the algorithm for linear regression, now is the time!

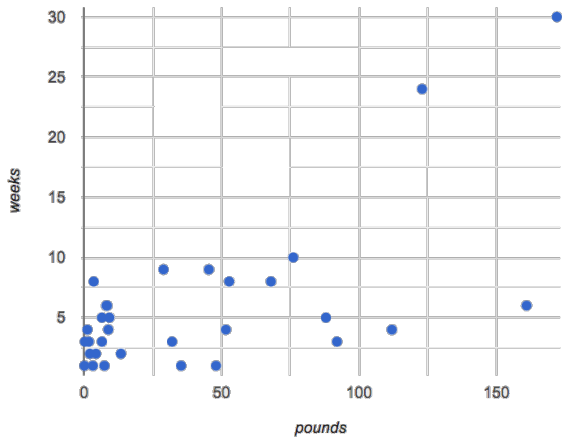
However, this algorithm is *not* a required portion of Bootstrap:Data Science.

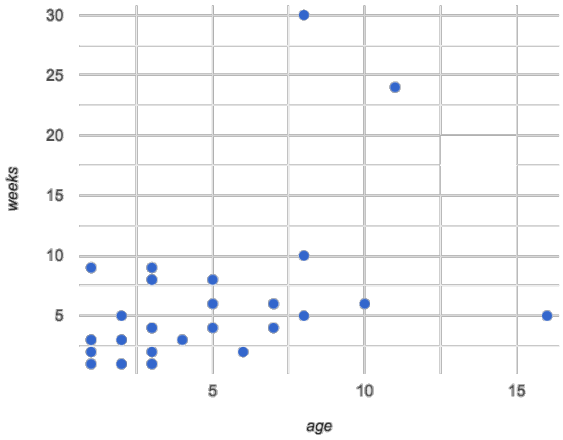
Investigate

Have students open this Interactive LR Plot.

- Try moving the blue point "P", and see what effect it has on the red line.
- Find the number called r. In your own words, explain what this number tells us.
- What's the largest r-value you can get? What do you think that number means?
- Where can you move it so that it is *most* aligned with the other points?
- Where can you move it so that it is *least* aligned with the other points?
- Could the ***regression line*** ever be above or below *all* the points? Why or why not?

Let's explore scatter plots for weeks-v-pounds and weeks-v-age:





After looking at the point clouds, we are left with a few questions:

- Do the relationships appear to be linear for one? Both?
- If a relationship is linear, what line in particular are the scatter plot points clustering around?
- What is the r-value for each relationship?

- Turn to Drawing Predictors (Page 77).
- In the first column, draw a ***line of best fit*** through each of the scatter plots.
- In the second column, circle whether the slope of the line (which is the same as the *direction* of the correlation) is positive or negative.

Synthesize

Give students some time to experiment, then share back observations. Can they come up with rules or suggestions for how to minimize error?

- Would it be possible to have a line that is *below* all the points? (no)
- Would it be possible to have a line that is *above* all the points? (no)
- Would it be possible to have a line with more points on one side than the other? (yes)

Linear Regression in Pyret

20 minutes

Overview

Students are introduced to the `lr-plot` function in Pyret, which performs a linear regression and plots the result.

Launch

Pyret includes a powerful display, which (1) draws a scatterplot, (2) draws the line of best fit, and (3) even displays the

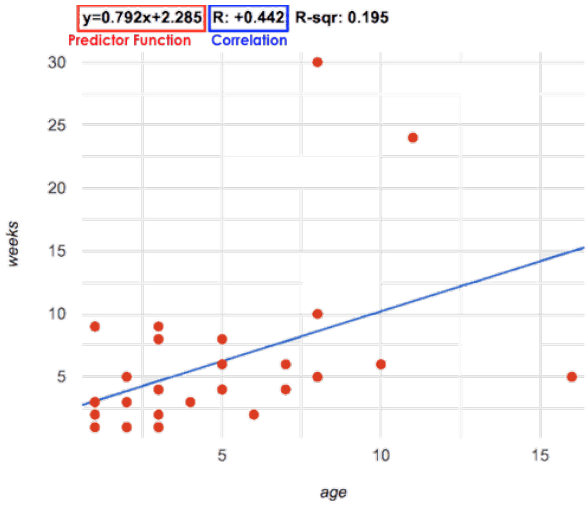
equation for that line:

```
# use linear regression to extract a predictor function
# lr-plot :: (t :: Table, ls :: String, xs :: String, ys :: String) -> Image
lr-plot(animals-table, "name", "age", "weeks")
```

`lr-plot` is a function that takes a Table and the names of **3 columns**:

- `ls` — the name of the column to use for *labels* (e.g. "names of pets")
- `xs` — the name of the column to use for *x-coordinates* (e.g. "age of each pet")
- `ys` — the name of the column to use for *y-coordinates* (e.g. "weeks for each pet to be adopted")

Our goal is to use values of the variable on our x-axis to *predict* values of the variable on our y-axis.



Pedagogical Note

We prefer the words "explanatory" and "response" in our curriculum, because in other contexts the words "dependent" and "independent" refer to whether or not the variables are related at all, as opposed to what role each plays in the relationship.

Have students create an `lr-plot` for our `animals-table`, using `"names"` for the labels, `"age"` for the x-axis and `"weeks"` for the y-axis.

The resulting scatterplot looks like those we've seen before, but it has a few important additions. First, we can see the ***line of best fit*** drawn onto the plot. We can also see the equation for that line (in red), in the form $y = mx + b$. In this plot, we can see that the slope of the line is 0.792, which means that on average, each extra year of age results in an extra 0.792 weeks of waiting to be adopted (about 5 or 6 extra days). By plugging in an animal's age for x, we can make a *prediction* about how many weeks it will take to be adopted. For example, we predict a 5-year-old animal to be adopted in $0.792(5) + 2.285 = 6.245$ weeks. That's the y-value exactly on the line at x=5.

The intercept is `2.285`. This is where the best-fitting line crosses the y-axis. We want to be careful not to interpret this too literally, and say that a newborn animal would be adopted in 2.285 weeks, because none of the animals in our data set was that young. Still, the ***regression line*** (or ***line of best fit***) suggests that a baby animal, whose age is close to 0, would take only about 3 weeks to be adopted.

We also see the r-value is +0.442. The sign is positive, consistent with the fact that the scatter plot point cloud, along with the line of best fit, slopes upward. The fact that the magnitude falls well between 0 and 1 tells us that the strength is moderate. This is consistent with the fact that the scatter plot points are somewhere between being really tightly clustered and really loosely scattered.

Going Deeper

Students may notice another value in the lr-plot, called R^2. This value describes the *percentage of the variation in the y-variable that is explained by least-squares regression on the x variable*. In other words, an R^2 value of 0.20 could mean that "20% of the variation in adoption time is explained by regressing adoption time on the age of the animal". Discussion of R^2 may be appropriate for older students, or in an AP Statistics class.

Investigate

- Make another lr-plot, but this time use the animals' weight as our explanatory variable instead of their age.
- If an animal is 5 years old, how long would our line of best fit predict they would wait to be adopted? What if they were a newborn, just 0 years old?
- If an animal weighs 21 pounds, how long would our line of best fit predict they would wait to be adopted? What if they weighed 0.1 pounds?
- Make another lr-plot, comparing the `age` v. `weeks` columns for *only the cats*.

Synthesize

A predictor only *makes sense within the range of the data that was used to generate it* . Statistical models are just proxies for the real world, drawn from a limited sample of data: they might make a useful prediction in the range of that data, but once we try to extrapolate beyond that data we may quickly get into trouble!

Does the linear regression for our sample of the Animals Dataset allow us to *make inferences* about the behavior of the larger dataset? Why or why not?

Interpreting LR Plots in Pyret

20 minutes

Overview

Students learn how to *write* about the results of a linear regression, using proper statistical terminology and thinking through the many ways this language can be misused.

Launch

How well can you interpret the results of a linear regression analysis? Can you write your own?

- What does it mean when a data point is *above* the line of best fit?
- What does it mean when a data point is *below* the line of best fit?

Investigate

- Turn to Interpreting Regression Lines & r-Values (Page 78), and match the write-up on the left with the line of best fit and r-value on the right.
- Turn to Regression Analysis in the Animals Dataset (Page 79) to see how Data Scientists would write up the finding involving cats' age and adoption time. Write up two other findings from the linear regressions you performed on this dataset.

When looking at a regression for adoption time v. age for just the cats, we saw that the slope of the predictor function was +0.23, meaning that for every year older a cat is, we expect a +0.23-week increase in the time taken to adopt the cat. The r-value was +0.566, confirming that the correlation is positive and indicating moderate strength.

Common Misconceptions

Students often think it doesn't matter which variable is assigned to be x and which is y in a regression. It's true that you'll get the same correlation either way---for example, $r = +0.442$ whether your scatter plot shows `weeks` v. `pounds` or `pounds` v. `weeks` . **However, the regression line *is* different, due to the math involved in minimizing *vertical* distances from the line, not horizontal** .

Synthesize

Have students read their text aloud, to get comfortable with the phrasing.

Your Analysis

flexible

Overview

Overview

Students repeat the previous activity, this time applying it to their own dataset and interpreting their own results. **Note: this activity can be done briefly as a homework assignment, but we recommend giving students an *additional class period* to work on this.**

Launch

Now that you've gotten some practice performing linear regression on the Animals Dataset, it's time to apply that knowledge to your own data!

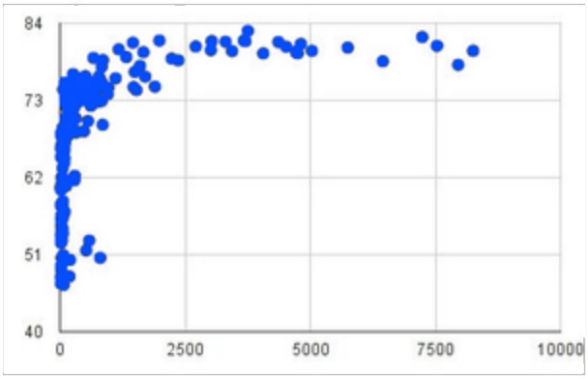
Investigate

- Write up your findings by filling out Regression Analysis in Your Dataset (Page 80).
- Students should fill in the Correlations portion of their Research Paper, using the scatter plots and linear regression plots they've constructed for their dataset and explaining what they show.

Synthesize

Have students share their findings with the class. Get excited about the connections they are making and the conclusions they are drawing! Encourage students to make suggestions to one another about further analysis.

You've learned how linear regression can be used to fit a line to a linear cloud, and how to determine the direction and strength of that relationship. The word "linear" is important here. In the image on the right, there's clearly a pattern, but it doesn't look like a straight line! There are many other kinds of statistical models out there, but all of them work the same way: use a particular kind of mathematical function (linear or otherwise), to figure out how to get the "best fit" for a cloud of data.

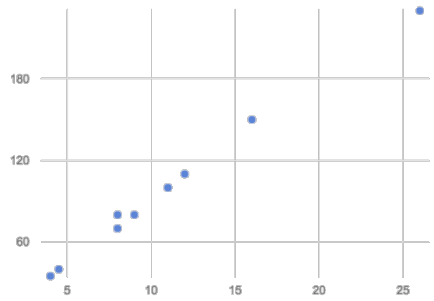
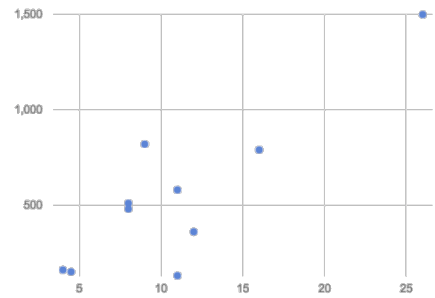
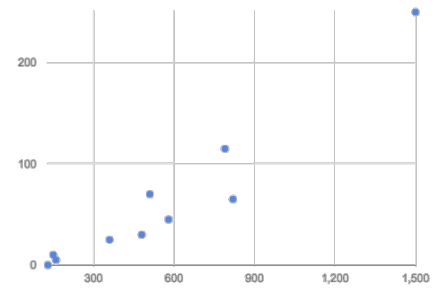
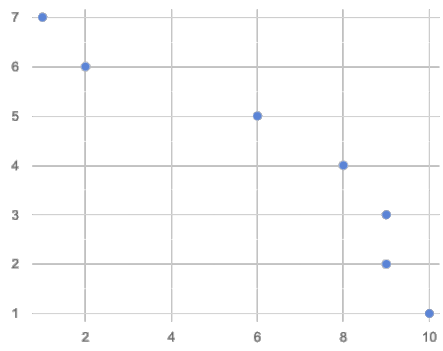


Additional Exercises:

- ../../lessons/ds-linear-regression/pages/describing-relationships-1.pdf
- ../../lessons/ds-linear-regression/pages/describing-relationships-2.pdf
- Project: Olympic Records - A mini-project in which students use linear regression to find correlations in the improvement of records in a sport of their choice.

Drawing Predictors

For each of the scatter plots below, draw a **predictor line** that seems like the best fit. Describe the correlation in terms of Direction and Strength, then estimate the r-value.

A		**Direction :** Positive Negative None **Strength :** Strong Weak **r :** _____
B		**Direction :** Positive Negative None **Strength :** Strong Weak **r :** _____
C		**Direction :** Positive Negative None **Strength :** Strong Weak **r :** _____
D		**Direction :** Positive Negative None **Strength :** Strong Weak **r :** _____

Interpreting Regression Lines & r-Values

Each description on the left is written about the linear regression findings on the right. Fill in the blanks using the information in the line of best fit and the r-value.

1	For every additional Marvel Universe movie released each year, the average person is predicted to consume _____ [amount] _____ [more / fewer] pounds of sugar! This correlation is _____ [strong, moderate, weak, non-existent].	`y = -3.19x + 12` `r = -0.05`
2	Shoe size and height are _____ [strongly, moderately, weakly, not], _____ [positively / negatively] correlated. If person A is one size bigger than person B, we predict that they will be roughly _____ [amount] inches taller than person B as well.	`y = 1.65x + 52` `r = 0.89`
3	There is _____ [a strong, a moderate, a weak, no] relationship found between the number of Uber drivers in a city and the number of babies born each year.	`y = -15.3x + 1150` `r = 0.01`
4	The correlation between weeks-of-school-missed and SAT score is _____ [strong, moderate, weak, non-existent] and _____ [positive / negative]. For every week a student misses, we predict a more than a _____ [amount] point _____ [gain / drop] in their SAT score.	`y = -5.35x - 16` `r = -0.65`
5	There is a _____ [strong, moderate, weak, non-existent], _____ [positive / negative] correlation between the number of streaming video services someone has, and how much they weigh. For each service, we expect them to be roughly _____ [amount] pounds heavier.	`y = 1.6x + 160` `r = 0.12`

Regression Analysis in the Animals Dataset

1) I performed a linear regression on a sample of cats from the shelter (dataset or subset) and found a moderate (r=0.566), positive (a weak/strong/moderate (R=...), positive/negative) correlation between age of the cats (in years) ([x-axis]) and number of weeks to adoption ([y-axis]). I would predict that a 1 year ([x-axis units]) increase in age ([x-axis]) is associated with a 0.23 week ([slope, y-units]) increase ([increase/decrease]) in adoption time ([y-axis]).

---

2) I performed a linear regression on a sample of _____ (dataset or subset) and found _____ (a weak/strong/moderate (R=...), positive/negative) correlation between _____ ([x-axis]) and _____ ([y-axis]).

I would predict that a 1 _____ ([x-axis units]) increase in _____ ([x-axis]) is associated with a _____ ([slope, y-units]) _____ ([increase/decrease]) in _____ ([y-axis]).

---

3) I performed a linear regression on a sample of _____ (dataset or subset) and found _____ (a weak/strong/moderate (R=...), positive/negative) correlation between _____ ([x-axis]) and _____ ([y-axis]).

I would predict that a 1 _____ ([x-axis units]) increase in _____ ([x-axis]) is associated with a _____ ([slope, y-units]) _____ ([increase/decrease]) in _____ ([y-axis]).

Regression Analysis in Your Dataset

My Dataset is _____.

1) I performed a linear regression on _____ and found
dataset or subset

_____ correlation between
a weak/strong/moderate (R=...), positive/negative

_____ and _____.
[x-axis] [y-axis]

I would predict that a 1 _____ increase in _____ is associated with a
[x-axis units] [x-axis]

_____ _____ in _____.
[slope, y-units] [increase/decrease] [y-axis]

---

2) I performed a linear regression on _____ and found
dataset or subset

_____ correlation between
a weak/strong/moderate (R=...), positive/negative

_____ and _____.
[x-axis] [y-axis]

I would predict that a 1 _____ increase in _____ is associated with a
[x-axis units] [x-axis]

_____ _____ in _____.
[slope, y-units] [increase/decrease] [y-axis]

---

3) I performed a linear regression on _____ and found
dataset or subset

_____ correlation between
a weak/strong/moderate (R=...), positive/negative

_____ and _____.
[x-axis] [y-axis]

I would predict that a 1 _____ increase in _____ is associated with a
[x-axis units] [x-axis]

_____ _____ in _____.
[slope, y-units] [increase/decrease] [y-axis]

What's on your mind?

Case Study: Ethics, Privacy, and Bias

My Case Study is _____

1) Read the case study you or your group was assigned, and write your summary here.

2) Is this a good thing or a bad thing? Why?

3) What are the arguments on *each* side?

Data Science used for this purpose is good because…

Data Science used for this purpose is bad because…

Ethics and Privacy

Students consider ethical issues and privacy in the context of data science.

Prerequisites	Introduction to Computational Data Science
Relevant Standards OK NGSS K12CS CSTA	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Describe ethical and privacy considerations when it comes to data science
Student-facing Lesson Goals	• Let's discuss ethical concerns surrounding data science.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	

Language Table

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram, scatter-plot, lr-plot`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Case Studies

40 minutes

Overview

Students break into groups and read one of three case studies, each dealing with a different issue in Data Science. They discuss the implications of each, then share back.

Launch

> "With great power comes great responsibility"

During World War 2, scientists were engaged in a race to develop new weapons, more powerful than anything the world had ever seen. While the immediate goal was "win the war", many of the scientists realized that the weapons they were

had ever seen. While the immediate goal was "win the war", many of the scientists realized that the weapons they were developing could be used for all sorts of things *after the war was over* - and not all of them were good.

With tech companies hiring Data Scientists at a staggering rate and collecting massive datasets on users for those scientists to mine, there's a *new* arms race happening right now. Search engines tailor their results based on what they know about the customer doing the search, and social media networks want to recommend friends based on what they know about all of us. Both of these goals require building profiles on everyone, figuring out what their preferences are and where they tend to spend their time. They might require figuring out whether each of us is male or female, more likely to go to a sports game or a play, or about to buy a dishwasher or a television.

But these datasets and profiles could be used for far more than that. What if the FBI used them to try and figure out who is likely to commit a crime, or a company tries to learn their employees' religion or sexual orientation?

As they build ever-more sophisticated models based on ever-more accurate datasets, Data Scientists need to think about the ethics of what they're doing as well!

Investigate

Divide the class into groups of 3-4, and assign each group a different case study. Have each group choose one person to share back with the class.

- How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did (Forbes)
- Facebook 'likes' can reveal your secrets (CNN)
- Algorithmic Bias in Criminal Sentencing (Propublica)

(**Note:** The third article is quite long, but only the first half is needed for students to complete this activity.)

Have students complete Case Study: Ethics, Privacy, and Bias (Page 82).

Synthesize

Give students time to discuss and share back. Encourage students to share back differing views on the articles.

What are some commonalities and differences among the issues raised by these articles?

OPTIONAL: Can the class come up with a list of "Rules for Ethical Data Science"?

Going Deeper

1) For homework, have students write arguments in support of a randomly-chosen side of each case study. Select twelve students (two for each side of all three case studies), and have them debate in front of the class. Each side gets to make "opening" and "closing" arguments, and they take turns so that the closer for each side can respond to what the other side said. Then have the class vote on who was most convincing.

2) For homework, have students find their own articles about ethical issues in data science and write a one-page essay defending one side of it.

Threats to Validity

Threats to Validity can undermine a conclusion, even if the analysis was done correctly.

Some examples of threats are:

- **Selection bias** - identifying the favorite food of the rabbits won't tell us anything reliable about what all the animals eat.
- **Study bias** - If someone is supposed to assess how much cat food is eaten each day on average, but they only measure how much cat food is put in the bowls (instead of how much is actually consumed), they'll end up with an over-estimate.
- **Poor choice of summary** - Suppose a different shelter that had 10 animals recorded adoption times (in weeks) as 1, 1, 1, 7, 7, 8, 8, 9, 9, 10. Using the mode (1) to report what's typical would make it seem like the animals were adopted much quicker than they really were, since 7 out of 10 animals took at least 7 weeks to be adopted.
- **Confounding variables** - Shelter workers might steer people towards newer animals, because they've become attached to the animals that have been there for a while, making it appear that "staying in the shelter longer" means "less likely to be adopted".

Threats to Validity

Students consider possible threats to the validity of their analysis.

Prerequisites	Linear Regression
Relevant Standards OK K12CS CSTA NGSS CC-Math	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to... • Define several types of Threats to Validity • Identify those threats by reading the description of an analysis • Identify those threats in their own analysis
Student-facing Lesson Goals	• Let's identify issues that could affect our data analysis.
Materials	• Lesson Slides (Google Slides) • Computer for each student (or pair), with access to the internet • Student workbook, and something to write with
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	
Language Table	

Types	Functions	Values
Number	`num-sqrt, num-sqr, mean, median, modes`	`4, -1.2, 2/3`
String	`string-repeat, string-contains`	`"hello", "91"`
Boolean	`==, <, <=, >=, string-equal`	`true, false`
Image	`triangle, circle, star, rectangle, ellipse, square, text, overlay, bar-chart, pie-chart, bar-chart-summarized, pie-chart-summarized, histogram, scatter-plot, lr-plot`	
Table	`count, .row-n, .order-by, .filter, .build-column`	

Glossary

threats to validity :: factors that can undermine the conclusion of a study

---

Threats to Validity

20 minutes

Overview

Students are introduced to the concept of *validity*, and a number of possible threats that might make an analysis invalid.

Launch

Survey says: “People prefer cats to dogs”

As good Data Scientists, the staff at the animal shelter is constantly gathering data about their animals, their volunteers, and the people who come to visit. But just because they have data doesn’t mean the conclusions they draw from it are correct! For example: suppose they surveyed 1,000 cat-owners and found that 95% of them thought cats were the best pet. Could they really claim that people generally prefer cats to dogs?

Have students share back what they think. The issue here is that cat-owners are not a representative sample of the population, so the claim is invalid.

There’s more to data analysis than simply collecting data and crunching numbers. In the example of the cat-owning survey, the claim that “people prefer cats to dogs” is **invalid** because the data itself wasn’t representative of the whole population (of course cat-owners are partial to cats!). This is just one example of what are called ***Threats to Validity***.

There are several major threats to validity you should be on guard against:

1. **Selection bias** - Data was gathered from a biased, non-representative sample of the population. This is the problem with surveying cat owners to find out which animal is most loved. *Remember that, in general, randomness is the key to obtaining unbiased samples!*
2. **Bias in the study design** - Suppose you survey a random sample of pet owners that includes representative numbers of both cat and dog owners. But you ask them a “loaded” question like “Since annual vet care comes to about $300 for dogs and only about half of that for cats, would you say that owning a cat is less of a burden than owning a dog?” This could easily lead to a misrepresentation of people’s true opinions.
3. **Poor choice of summary** - Even if the selection is unbiased, sometimes outliers are so extreme that they shift the results of our analysis (such as the mean) in ways that don’t represent the population as a whole. For example, if the shelter happened to house a 100-year-old tortoise, and summarized its animals’ ages with the mean, this would inflate our perception of what age is typical.
4. **Sample error** - Even if the selection is unbiased and has a large enough sample size, sometimes outliers are so extreme that they shift the results of our analysis in ways that don’t represent the population as a whole.
5. **Confounding variables** - The gathered data does not take into account other factors that might influence a relationship. For example, a study might conclude that cat owners are more environmentally conscious: they’re more likely to use public transportation than dog owners. The confounding variable here could be urban versus rural dwelling: people who live in big cities are more likely to use public transportation and also more likely to own cats.

This is just a small list of different threats to validity. There are plenty more!

Investigate

On Identifying Threats to Validity (Page 84) and Identifying Threats to Validity (Page 85), you’ll find four different claims backed by four different datasets. Each one of those claims suffers from a serious threat to validity. Can you figure out what those threats are?

Synthesize

Give students time to discuss and share back.

Life is messy, and there are *always* threats to validity. Data Science is about doing the best you can to minimize those threats, and to be up front about what they are whenever you publish a finding. When you do your own analysis, make sure you include a discussion of the threats to validity!

Fake News!

20 minutes

Overview

Students are asked to consider the ways in which statistics are misused in popular culture, and become critical consumers of some statistical claims. Finally, they are given the opportunity to misuse their *own* statistics, to better understand how someone might distort data for their own ends.

Launch

You've already seen a number of ways that statistics can be misused:

1. Intentionally using the wrong chart
2. Changing the scale of a chart
3. Using the mean instead of the median with heavily-skewed data
4. Using the wrong language when describing a Linear Regression
5. Using a correlation to imply causation

With all the news being shared through newspapers, television, radio, and social media, it's important to be critical consumers of information!

Investigate

- On Fake News! (Page 86), you'll find some deliberately misleading claims made by slimy Data Scientists. Can you figure out *why these claims should not be trusted* ?
- Once you've finished, consider your own dataset and analysis: what misleading claims could someone make about your work? Turn to Lies, Darned Lies, and Statistics (Page 87), and come up with four misleading claims based on data or displays from your work.
- Trade papers with another group, and see if you can figure out why each other's claims are not to be trusted!

Synthesize

Have students share back their "lies". Was anyone able to stump the other group?

Your Analysis

flexible

Overview

Students repeat the previous activity, this time applying it to their own dataset and interpreting their own results. **Note: this activity can be done briefly as a homework assignment, but we recommend giving students an *additional class period* to work on this.**

Launch

In every analysis, there are always threats to validity. It's important to always be upfront about what those threats are, so that anyone who reads your analysis can make their own decision.

Investigate

- Students should fill in the Findings portion of their Research Paper, discussing threats to validity and drawing conclusions from their linear regression results.

Additional Exercises:

- Identifying Threats to Validity (Part 1)
- Identifying Threats to Validity (Part 2)
- Identifying Threats to Validity (Part 3)
- Project: Project: Threats to Validity

Identifying Threats to Validity

Some volunteers from the animal shelter surveyed a group of pet owners at a local dog park. They found that almost all of the owners were there with their dogs. From this survey, they concluded that dogs are the most popular pet in the state.

What are some possible threats to the validity of this conclusion?

The animal shelter noticed a large increase in pet adoptions between Christmas and Valentine's Day. They conclude that at the current rate, there will be a huge demand for pets this spring.

What are some possible threats to the validity of this conclusion?

Identifying Threats to Validity

The animal shelter wanted to find out what kind of food to buy for their animals. They took a random sample of two animals and the food they eat, and they found that spider and rabbit food was by far the most popular cuisine!

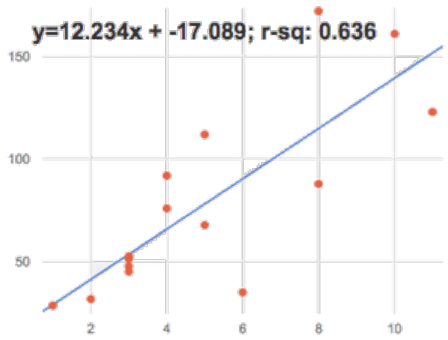
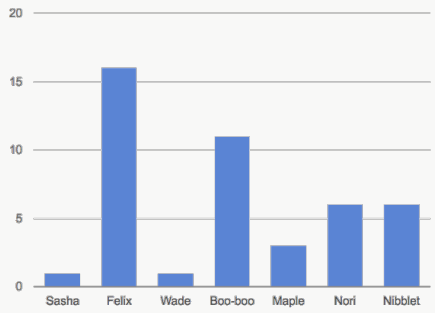
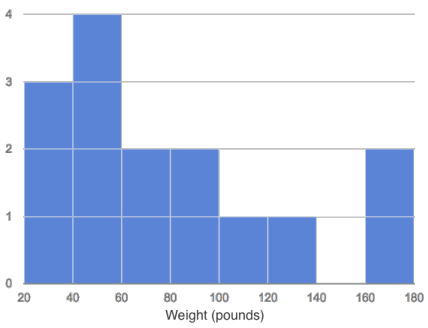
Explain why sampling just two animals can result in unreliable conclusions about what kind of food is needed.

A volunteer opens the shelter in the morning and walks all the dogs. At mid-day, another volunteer feeds all the dogs and walks them again. In the evening, a third volunteer walks the dogs a final time and closes the shelter. The volunteers report that the dogs are much friendlier and more active at mid-day, so the shelter staff assume the second volunteer must be better with animals than the others.

What are some possible threats to the validity of this conclusion?

Fake News!

Every claim below is wrong! Your job is to figure out why by looking at the data.

	Data	Claim	What's Wrong
1	The average player on a basketball team is 6′1″.	*"Most of the players are taller than 6 feet."*	
2	Linear regression found a positive correlation (r=0.18) between people's height and salary.	*"Higher salaries can make people taller!."*	
3		*"According to the predictor function indicated here, the value on the x-axis will predict the value on the y-axis 63.6% of the time."*	
4		*"According to this bar chart, Felix makes up a little more than 15% of the total ages of all the animals in the dataset."*	
5		*"According to this histogram, most animals weigh between 40 and 60 pounds."*	
6	Linear regression found a negative correlation (r= -0.91) between the number of hairs on a person's head and their likelihood of owning a wig.	*"Owning wigs causes people to go bald."*	

Lies, Darned Lies, and Statistics

1) Using real data and displays from your dataset, come up with a misleading claim.

2) Trade papers with someone and figure out why their claims are wrong!

Data	Claim	Why it's wrong
1		
2		
3		
4		

What's on your mind?

Design Recipe

Directions:

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Directions:

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Design Recipe

Directions:

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Directions:

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Design Recipe

Directions:

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Directions:

Contract and Purpose Statement

Every contract has three parts...

_____ :: _____ -> _____
function name *domain* *range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
`triangle`	`::`	`(side-length :: Number, style :: String, color :: String)`	`->`	`Image`
`circle`	`::`	`(radius :: Number, style :: String, color :: String)`	`->`	`Image`
`star`	`::`	`(radius :: Number, style :: String, color :: String)`	`->`	`Image`
`rectangle`	`::`	`(width :: Num, height :: Num, style :: Str, color :: Str)`	`->`	`Image`
`ellipse`	`::`	`(width :: Num, height :: Num, style :: Str, color :: Str)`	`->`	`Image`
`square`	`::`	`(size-length :: Number, style :: String, color :: String)`	`->`	`Image`
`text`	`::`	`(str :: String, size :: Number, color :: String)`	`->`	`Image`
`overlay`	`::`	`(img1 :: Image, img2 :: Image)`	`->`	`Image`
`beside`	`::`	`(img1 :: Image, img2 :: Image)`	`->`	`Image`
`above`	`::`	`(img1 :: Image, img2 :: Image)`	`->`	`Image`
`put-image`	`::`	`(img1 :: Image, x :: Number, y :: Number, img2 :: Image)`	`->`	`Image`
`rotate`	`::`	`(degree :: Number, img :: Image)`	`->`	`Image`
`scale`	`::`	`(factor :: Number, img :: Image)`	`->`	`Image`

Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
`string-repeat`	`::`	`(text :: String, repeat :: Number)`	`->`	`String`
`string-contains`	`::`	`(text :: String, search-for :: String)`	`->`	`Boolean`
`num-sqr`	`::`	`(n :: Number)`	`->`	`Number`
`num-sqrt`	`::`	`(n :: Number)`	`->`	`Number`
`num-min`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`num-max`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`count`	`::`	`(t :: Table, col :: String)`	`->`	`Table`
`mean`	`::`	`(t :: Table, col :: String)`	`->`	`Number`
`median`	`::`	`(t :: Table, col :: String)`	`->`	`Number`
`modes`	`::`	`(t :: Table, col :: String)`	`->`	`List<Number>`
`bar-chart`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`pie-chart`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`histogram`	`::`	`(t :: Table, values :: String, bin-width :: Number)`	`->`	`Image`

Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
box-plot	::	(t :: Table, col :: String)	->	Image
modified-box-plot	::	(t :: Table, col :: String)	->	Image
scatter-plot	::	(t :: Table, labels :: String, xs :: String, ys :: String)	->	Image
image-scatter-plot	::	(t :: Table, xs :: String, ys :: String, f :: (Row -> Image))	->	Image
r-value	::	(t :: Table, xs :: String, ys :: String)	->	Number
lr-plot	::	(t :: Table, labels :: String, xs :: String, ys :: String)	->	Image
random-rows	::	(t :: Table, num-rows :: Number)	->	Table
<Table>.row-n	::	(n :: Number)	->	Row
<Table>.order-by	::	(col :: String, increasing :: Boolean)	->	Table
<Table>.filter	::	(test :: (Row -> Boolean))	->	Table
<Table>.build-column	::	(col :: String, builder :: (Row -> Any))	->	Table
bar-chart-summarized	::	(t :: Table, labels :: String, values :: String)	->	Image
pie-chart-summarized	::	(t :: Table, labels :: String, values :: String)	->	Image