Unit 1: From Data to Conclusions

Mhairi McNeill and Robert Reside

17 June 2016

What is data?

What isn't data? Any piece of information about the world is potentially data. The prices on a menu is data; the gender of your 5 closest friends is data; a book is data; a picture of your cat is data; the goals scored by your favourite team is data; a conversation is data.

Now, clearly, some of these sources of data are more useful than others. Data becomes useful when we can use it make conclusions about the world. To be useful data needs to fulfil two criteria. First, we need to be able to analyse it. For that reason we will be focusing numeric data (menu prices, number of goals) and category data (gender of friends). These are the types of data we can analyse using statistics.

The second criteria for usefulness is the quality of data. I have left quality intentionally vague here because there is a lot ways data can be low quality, ways which I am sure you have an intuitive understanding of. I want to know if a neighbourhood has good restaurants. Compare the two following sources of data: firstly, I know didn't like the last restaurant I went to there. Secondly I have a dataset of 20,000 detailed online restaurant reviews from the last 5 years written by 1,000s of people. Which data source do you think is better quality for answering my question? The second data has far more information, about far more restaurants and from more people than just me.

Don't confuse quantity of data for quality though. Think of the following headline: 'Tory economic policies deeply unpopular says survey of 10,000 Labour voters'. How useful is that survey for understanding general public opinion? Wouldn't a survey of 500 people chosen completely at random be better?

Be careful with data quality, and match the strength of your conclusions to the strength of your data. We very rarely have data that is perfect for the question we want to answer. For example, the online restaurant review data, while probably better than my single opinion, only comes from people who have actually taken the time to review the restaurant. This probably represents the strongest opinions of all restaurant goers - you'd be much more likely to leave a review if you really loved the restaurant or really hated it. Also, people who regularly leave online reviews are probably younger on average, and probably eat out more often than average. Don't give in if you don't have perfect data (if the 20,000 reviews are all negative then the neighbourhood probably does have remarkably terrible restaurant). But for good answers to your questions quality data is far more important than any of the techniques we are about to discuss below. Always keep quality in mind.

Types of Data

Earlier I said that we'd be looking at numeric data and categorical data. Numeric data is anything that can be measured by a number. That number can be a whole number. For example: how old is this child? Or it can be a decimal number. For example: what is the height of this child?

Categorical data describes the category something is in. Each child is in a year at school, and has a gender. The year a child is in is a special kind of categorical data known as ordinal data. This is because the categories have a natural ordering: year 1 is less than year 2 which is less than year 3 etc. While gender has no natural ordering; this is sometimes called nominal data.

The type of data matters because it effects what type of analysis will work best on the data. Some data has an ambiguous type. You could argue that the year a child is in is also numeric data (1, 2, 3 etc.). That's fine: it just means more than one type of analysis will work on that data.

You'll notice that we have lots of bits of information we can measure about each child. We'll be working with this type of data a lot. In this example where we have surveyed school children each child makes up an *observation*; each measurement (age, height, year, gender) is an *variable*. Thinking about observations and variables is a very useful way of looking at data, which we will see more of later.

Exercises

1. What data do you imagine capturing in your future? This could be for work or for a hobby. Give three examples of numeric data and three examples of categorical data.
2. Of the categorical data which was ordinal and which was nominal?
3. What might be the potential quality issues with:

i) Doing medical trials to test the efficacy of your drug.
ii) Evaluating your personality by asking all your friends.
iii) A internet survey to find out current attitudes to gay marriage. How would the quality issues change if you paid people to fill out the survey?

4. Bonus Question: In the 2015 UK General Election the polls were predicting a vote much closer than the final result. What do you think might be potential quality issues with doing election polls? Can you find out why the poll-makers think they got the result wrong?

Answers

1. Some examples:

- Customer service
  - Numeric: number of customers, satisfaction scores, number of complaints.
  - Categorical: complaint types (nominal), dispute resolved - yes or no (ordinal potentially - yes > no), gender of customers (nominal)
- A runner
  - Numeric: heartrate, miles run, distance run
  - Categorical: surface type (nominal), month ran in (ordinal), gear worn (nominal).
- Social media
  - Numeric: Monthly visits, bounce rate, number of engagements
  - Categorical: browser type (nominal), keywords used (nominal), type of engagement (possibly ordinal - is a share > a like?)

3.

i) This is a huge question. There's so many answers, here's some examples of good answers.
   - Placebo effect. We tend to find any intervention effective (need to have a control group treated with nothing).
   - Are the patents in your trial the population as will be actually using your drug.
   - Can you find very rare adverse effects with your sample size?
   - How are you measuring effectiveness. If it's subjective will the people measuring this be biased and want the drug to succeed. (This is normally dealt with by doing double blind trials - where neither the patients of the doctors know which drugs are real and which are placebo.)

ii) - Biased sample. Your friends probably like you or they wouldn't be your friend.

- They won't want to tell you how they actually feel because it would hurt your feelings.

iii)
- Internet surveys tend to be quite biased in the type of people who respond. Only a certain type of person willingly fills in surveys.
- Often with this type of survey you'll get a very keen group of people who care about the results coming out one way or the other and then try to encourage everyone they know who thinks like them to answer the survey.
- Paying money will probably get a wider range of people to take part (still probably not a representative sample of people). However, people become more likely to just put in random answers to get their money, without thinking about the questions.

4. Poll makers spend a lot of time trying to get a representative sample of the population to answer their questions. But it is difficult. In the last election they think that they under-sampled people who were at work during the time the phone polls and door to door polls were done.

Working with R

To analyse numeric and categorical data we are going to use a programming language called R. You might have used a point-and-click interface for analysing data before like Excel, SPSS or Minitab. Using R is somewhat similar to these programmes but also somewhat different. The biggest difference is that you'll be typing commands rather than clicking on boxes and menus. At first this is more difficult but will almost certainly become quicker and easier in the long run.

There's a bit of debate about what software you should use to analyse data. Every way of doing it has advantages and disadvantages. We'll be using R in this course because:

1. When you type everything you have a written record of everything you've done to a piece of data. This makes it easier to check your logic and the preserve the original data.
2. You will often need to do the same piece of analysis over and over on different datasets. Having the code to do the analysis written out makes it easy to repeat.
3. R is free and open source.
4. R has a lot of excellent libraries (downloadable extensions to R) which make generating plots, doing specialised techniques, making reports and even producing interactive dashboards easy.

If you've learnt to programme before, that's great. I'm sure you will find learning R straightforward. There are a few idiosyncratic features of R that can trip up experienced programmers. You might find this guide useful: http://www.johndcook.com/blog/r_language_for_programmers/

If you've never done programming before that is also great. R is a good language for a beginner and once you've learnt R it will be much easier to pick up other languages, which opens up a huge range of things you can do with computers!

Installing R

We are going to install R the programming language. R is programme that actually does the analysis - you interact with this programme by typing R commands. For example, if you type `2 + 2`, R will do the maths and return the result to you.

Then we are going to install an IDE (interactive development environment) called RStudio. RStudio is a very popular way of interacting with R. While you don't need it to write R code RStudio makes it easy to do things like: 1. Write longer pieces of code and get R to run it piece by piece. 2. Keep track of what data R knows about. 3. Organise all your R code files. 4. See plots and tables you've created in R. 5. Make special types of R files e.g. reports, dashboards etc. 6. Access R's in-built help 7. Manage which R packages (extensions to R) you have installed.

R is free and open source. It is written by volunteers and all the all packages you'll use were also written by volunteers.

RStudio is also free and open source, but is made by a profit making company. They make their money by selling a professional version of RStudio that runs on a sever and has support.

[Videos here]

Getting started.

Once you have RStudio installed and opened you should see four panels. We'll ignore the two right-hand-side panels for just now.

The bottom-left panel is the R interpreter. We can type R commands into the interpreter and R will do some calculations and return an answer. Let's start with one of the most basic commands possible. Let's get R to add 2 and 2. Type `2 + 2` into the interpreter and press enter.

You should see something like this:

```r
2 + 2
```

```
## [1] 4
```

After you hit enter R understood the command and found the answer, returning it almost instantly to you. (The little [1] in front of the answer just means that there's only one element to the answer).

Now, type the same thing in the top left panel. When you press enter here nothing will happen. The top left panel is basically just a very simple text editor like Notepad on windows or TextEdit on a Mac. You can absolutely write R code in a separate text editor and many people do. A big advantage of writing code inside RStudio is that it's very easy to transfer code from the editor to the R interpreter. Just move your cursor to the line with `2 + 2` and press ctrl and enter at the same time. Pressing cmd+enter also works on a Mac. The code you have written will now appear in the interpreter along with the answer.

If you have a longer piece of code that goes across multiple lines you will need to highlight the lines and then press ctrl+enter. Try this just now after typing this in the top left editor.

```r
1 + 2 +
4
```

```
## [1] 7
```

You can save the code you have written in the text editor and come back to it at any time. Just go to **File** and then **Save As...**. When going through this unit keep everything you've written in the text editor saved.

Vectors

A very important thing to understand about R is that it works with *vectors*. Vectors are a list of items. Often this will be a list of numbers, but we will also work with vectors of strings (in programming languages anything made up of letters is called a string) and vectors made up of True and False values. In R we call a vector of strings a character vector. A vector made up of only True and False is a logical vector.

There are other types of vector in R, but for just now we will only be working with numeric, character and logical vectors. To make a vector in R we surround the elements with `c(` and `)`. For example

```
c(1, 4, 6, 3)
```

```
## [1] 1 4 6 3
```

Below is an example of the three vectors types we will learn about.

Numeric | `c(2.2, 0.1, 5)` Character | `c('apple', 'pear', 'e')` Logical | `c(TRUE, FALSE, FALSE)`

Exercise

1. Create a numeric vector containing this data: 10, 11, 11, 14. (Type in the text editor and use ctrl+enter to send to the interpreter)
2. Create a character vector containing this data: 'small', 'small', 'small', 'large'.

Assignment

Creating a vector isn't very useful if we can't store that vector somewhere. In a programming language if we want to store some data, or the results of a calculation we use assignment. Assignment is basically giving a name to a piece of data. If we want to use that data we can just find it by using its name, rather than typing out the data again.

In R we can assign using an arrow. Here's an example of assigning the name x to a vector:

```
x <- c(1, 4, 6, 3)
```

Now we've assigned a vector to x we can see what x is at any point by just typing x into the interpreter.

```
x
```

```
## [1] 1 4 6 3
```

Here we have been giving names to vectors but we can give a name to any type of R object. Here's how we would assign a name to a number:

```
days <- 5
```

We can also give names to more complicated objects which will will see later.

Names in R can only contain letters, numbers, '.' and '_'. They can't start with a number or an underline. Try to make names descriptive, so that when you go back to a piece of analysis you should be able to understand what everything is. Since you can't have spaces in names you can use the underscore character as a space:

```
days_until_launch <- 5
```

Exercise

1. Give the numeric vector we created earlier the name `height`.
2. Give the character vector the name `height_category`.

The difference between 'apple' and apple

A very common point of confusion for new programmers is the difference between a string like `'apple'` and a name like `apple`.

First first `'apple'` with quotes is simply a piece of data. If we type this into the interpreter R sees a new string we have created.

```
'apple'
```

```
## [1] "apple"
```

The second is a name given to an object. Since we haven't called anything `apple` yet R doesn't know anything about `apple`.

```
apple
```

```
## Error in eval(expr, envir, enclos): object 'apple' not found
```

Another way of putting it is that `'apple'` is something that gets given a name:

```
fruit <- 'apple'
```

And `apple` is a name that you can give to data:

```
apple <- TRUE
```

Now you've understood the basics of R, we can begin analysing some data. Starting with some very basic, small data we'll discover ways the data can visualised and summarised which will help us understand it better and answer questions we have about the data.

Visualising Data

Barplots

Here's an example of a piece of data. You asked 15 people how much they liked spinach on a scale from -3 to 3. Where 3 means they love spinach, 0 means they feel completely neutral on spinach and -3 means they hate spinach. Here are the results:

0, 1, -3, 0, -2, 0, -3, 1, 0, -3, -3, 3, 3, 0, 2

Create a vector in R containing the data above. Assign the name `spinach_rating` to it.

Now, if I want to understand what the general opinion is of spinach I can simply read though the the list of scores given and get a general impression. I can see quite a few 0s and -3s. It's possible to understand data just by reading it when we have a small dataset like this. However, imagine we had 100 responses or 1000 responses. We need to do something to understand what our data is actually saying.

The first thing to do is the make a summary table of the data. How many people gave each of the available scores. R makes this very easy:

```r
table(spinach_rating)
```

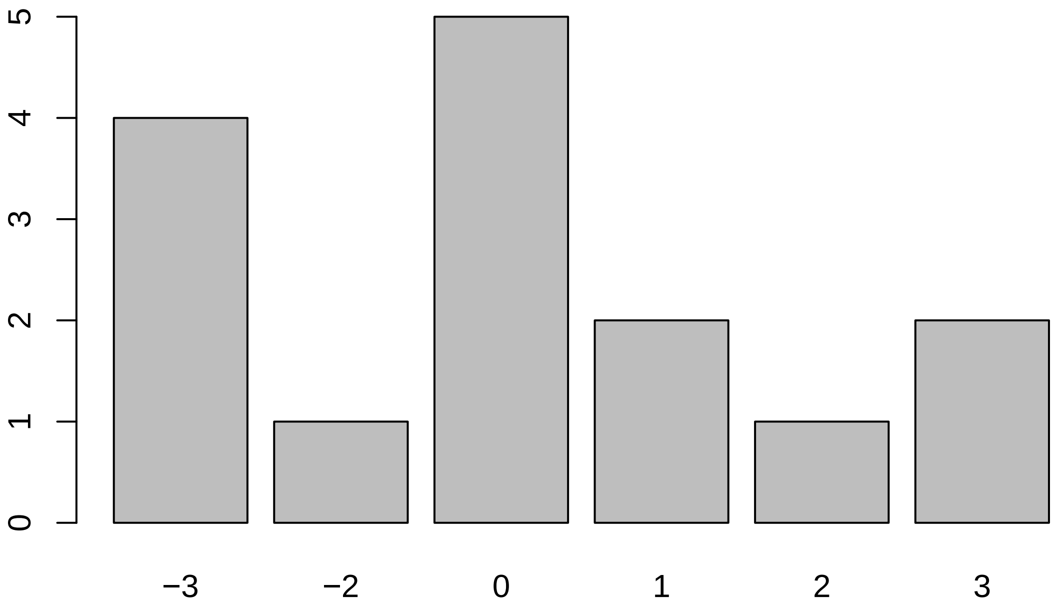
```
## spinach_rating
## -3 -2  0  1  2  3
##  4  1  5  2  1  2
```

We can see that the most popular score to give was a zero, with a minus three the second most popular. We can do even better than that though. A plot will make the results even more clear. Assign the table you have just created to a variable called `spinach_table`.

```r
spinach_table <- table(spinach_rating)
```

And then use the `barplot` to create a plot.

```r
barplot(spinach_table)
```



It's very easy to go from a raw list of numbers to a picture that gives us a very clear overview of the numbers. We can now make some conclusions about the the data: of our 15 respondents most people were neutral on spinach or hated it, although 4 out of the 15 gave spinach an overall positive rating.

Functions

In the code above we used two functions from R: `table` and `barplot`. A function is anything that takes an input and returns an output.

The `table` function took our vector of data as an input and transformed it into an table.

The `barplot` function took in a table and transformed it into plot.

We'll be using a lot of functions, and later we'll even be writing our own functions. They will all follow the same basic pattern of taking in an input and returning an output. We also always write them like this, with the input going in brackets after the name of the function:

```r
some_function(some_input)
```

The output is normally a new R object, like the table we created above and assigned to `spinach_table`. But we can also have other types of output - like the plot we saw.

Histograms

You also asked the same 15 people how tall they were. Here are the results, all in centimetres:

133, 110, 224, 134, 135, 136, 125, 137, 104, 132, 114, 130, 129, 237, 131

Now, just looking at those raw numbers it's a little tricky to say much about how tall the people we surveyed are.

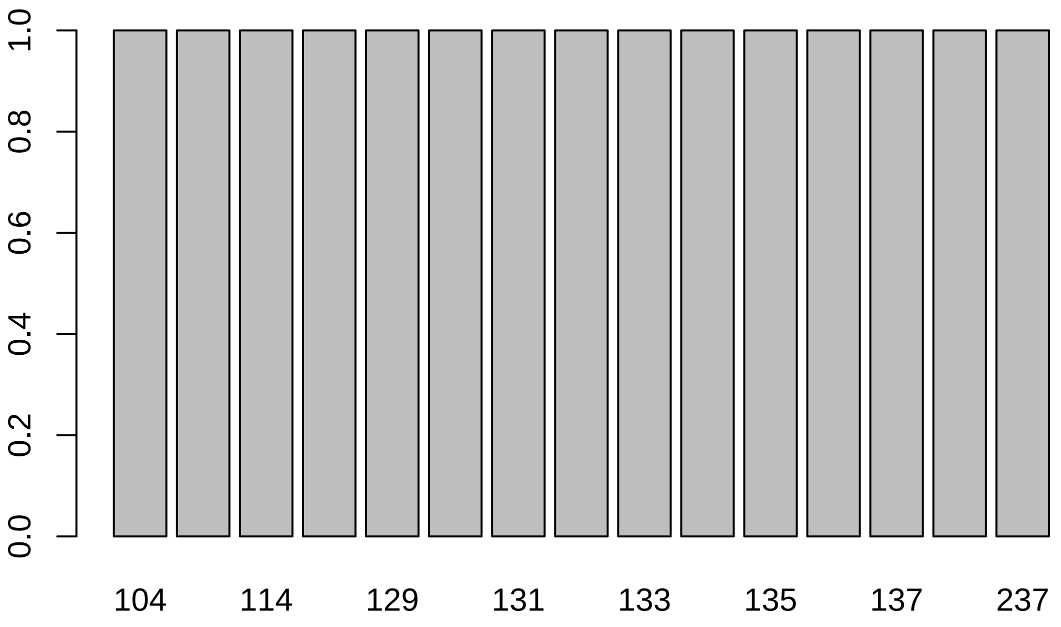
Let's try the technique we learnt above and translate those numbers into a table and a plot:

```r
height <- c(133, 110, 224, 134, 135, 136, 125, 137, 104, 132, 114, 130,
129, 237, 131)
height_table <- table(height)

height_table
```

```
## height
## 104 110 114 125 129 130 131 132 133 134 135 136 137 224 237
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

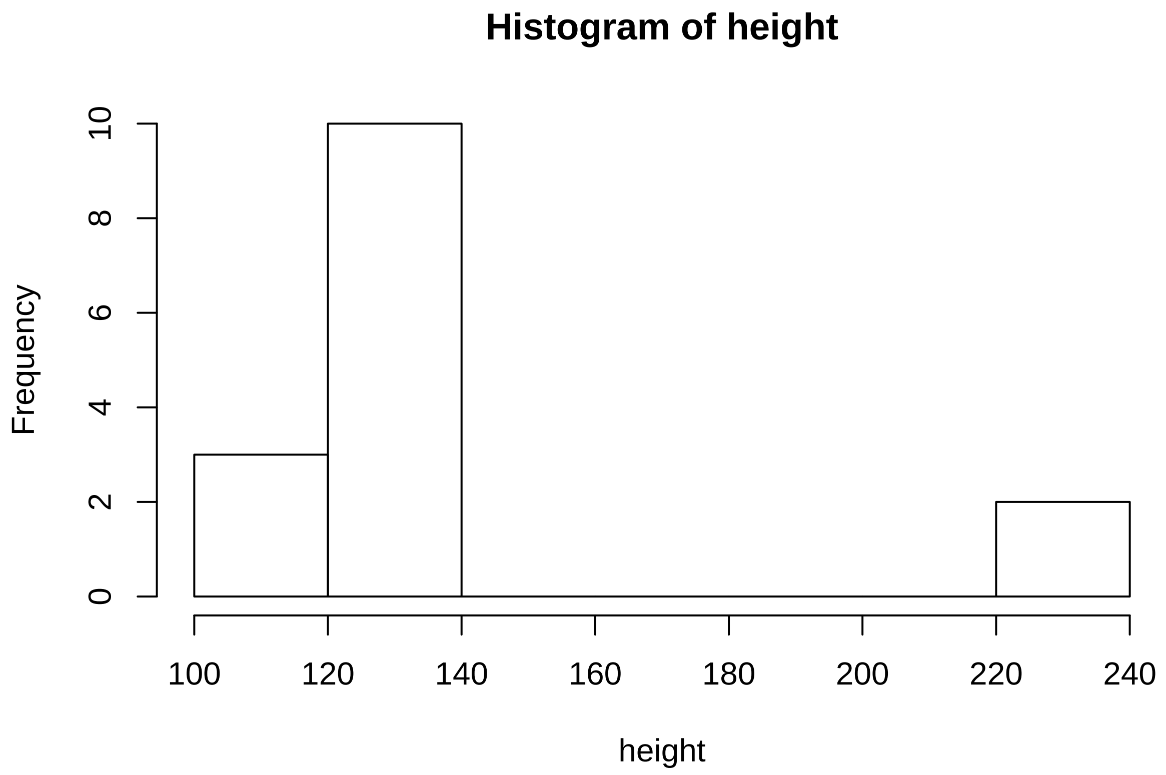
```r
barplot(height_table)
```



That was not that helpful. Since every person in our sample had a different height each count in our table as just one. We need a different way of summarising this data.

Let's start with a plot for this data. We're going to use a histogram, a type of visualisation that looks very similar to a barplot except that each bar counts a range of values, instead of just one single value. Here's how we do this in R:

```r
hist(height)
```

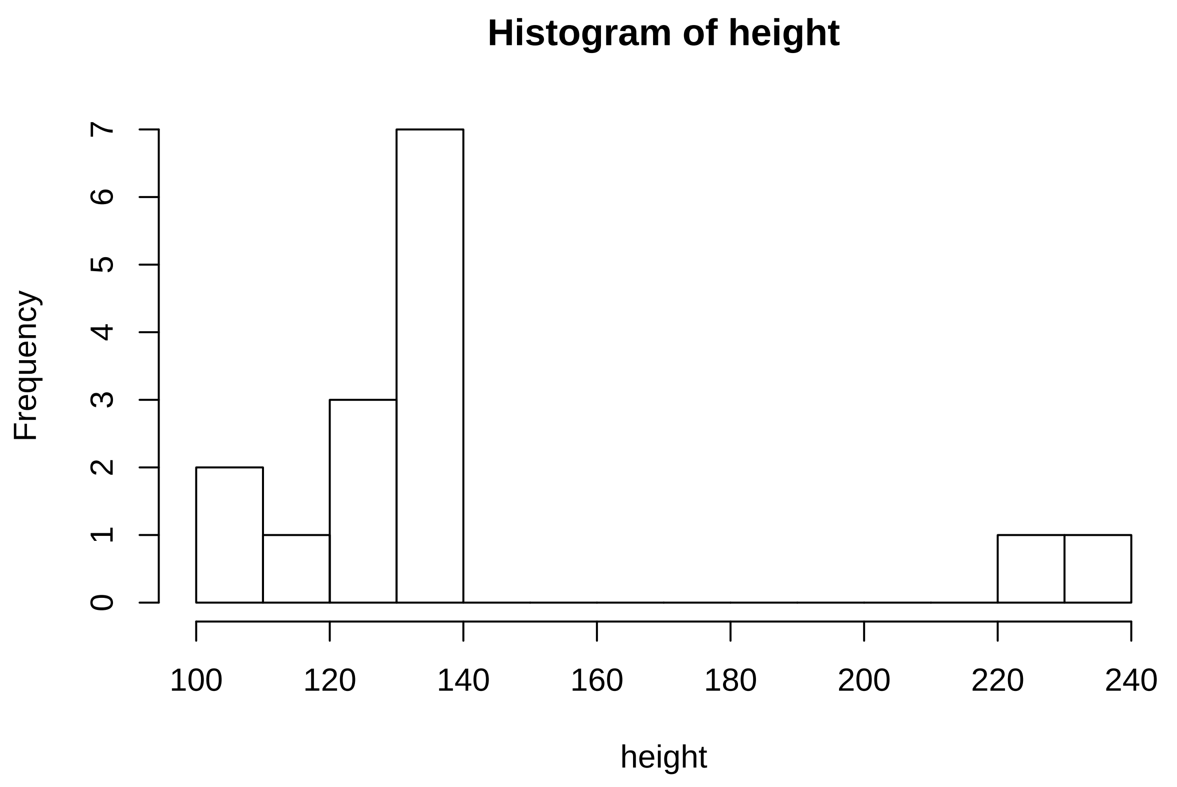


Now we can see that most of the heights were between 100cm and 140cm, with a couple of very tall people who were over 220cm. Perhaps most of the people you surveyed were children. Note that the bars in the histogram are directly next to each other - while the bars in a barplot have spaces between them. That is how you can tell at a glance which type of plot you are looking at.

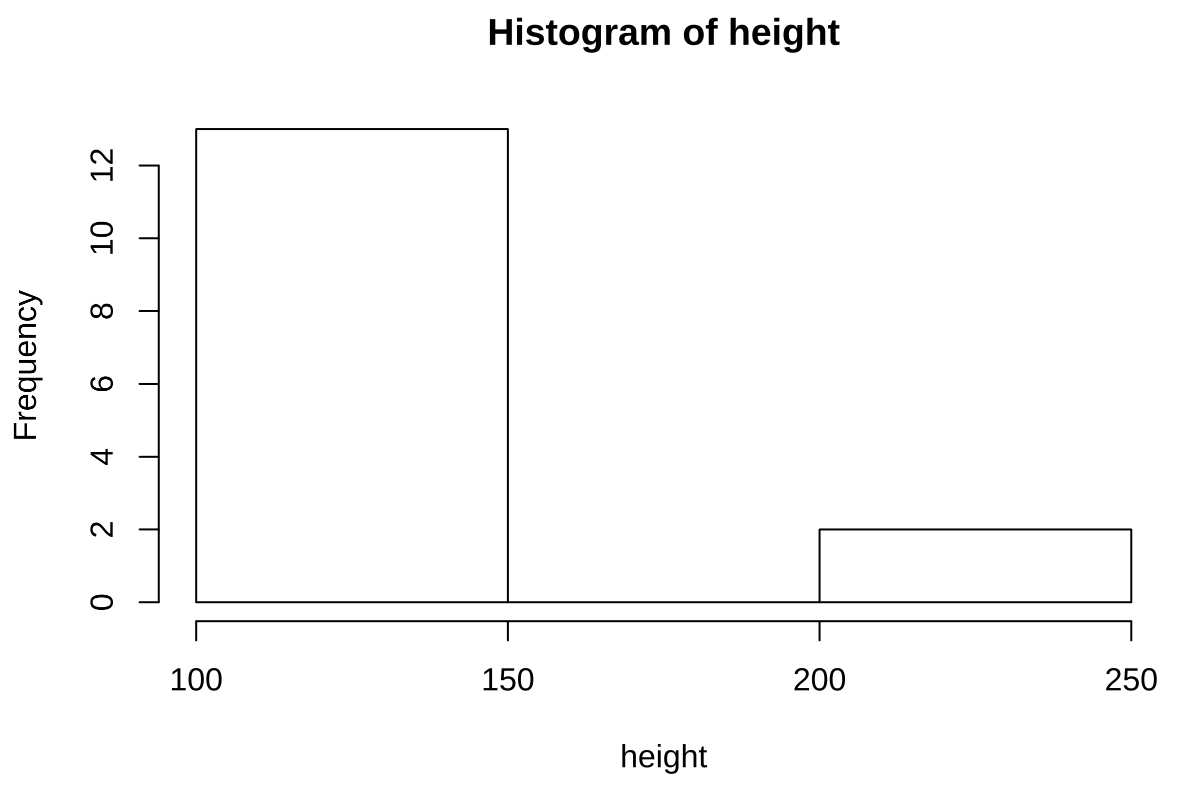
With histograms, the size of the groups can change the shape of plot significantly. If we have more groups we can get a more accurate picture of the data, but too many groups then the plot becomes meaningless - like the barplot of heights we created earlier.

We can set the number of groups in our histogram by changing the breaks argument in the function. Compare the two plots below to the one before.

```r
hist(height, breaks = 10)
```



```r
hist(height, breaks = 3)
```



This is why it is important to try a range of plot sizes; try to use histograms which describe the data well, and don't hide important features.

What if you want to summarise the height data into a table? You can use the `cut` function. This function turns numeric data into a set of groups, which we can then use `table` to summarise.

```
height_groups <- cut(height, 3)
table(height_groups)
```

```
## height_groups
## (104,148] (148,193] (193,237]
##        13         0         2
```

Describing data

Let's go back to our spinach opinion data. You will often find that you want just one number which can describe the dataset overall; we call that number an average. This lets us know what the typical respondent thinks of spinach. The most common type of average is a mean, which is very easy to calculate in R:

```
mean(spinach_rating)
```

```
## [1] -0.2666667
```

So, on average our survey respondents gave spinach a rating of -0.3 i.e. they were very slightly negative on spinach overall. Note, that I rounded the answer given by R. It's rarely helpful to know the exact value given by R; it's just hard for your audience to read. Also, giving an answer to 10 decimal places implies that you can measure spinach opinion to 10 decimal places, which of course you can't.

The maths

The maths behind how R does this is very simple. To calculate a mean you just need to add up all the values in your dataset and divide by the number of values you just added up. Written down mathematically:

$$\bar{x} = \frac{\sum x}{n} \tag{1}$$

Here x is all the values in your dataset and $\sum x$, means take the sum of them all. $\bar{x}$ is a symbol we use for the mean of x values and n is just the size of the dataset.

We can see that this gives results that make sense. What do you think the typical value in these three numbers would be: 3, 4, 5? I think 4 would be a reasonable answer, it's in the middle. Let's check what the mean is:

$$\bar{x} = \frac{3+4+5}{3} = \frac{12}{3} = 4 \tag{2}$$

Exercise

What do you think the average should be for these values? Calculate the mean by hand and compare:

1. 2, 2, 2
2. 4, 6
3. 2, 2, 5

Median

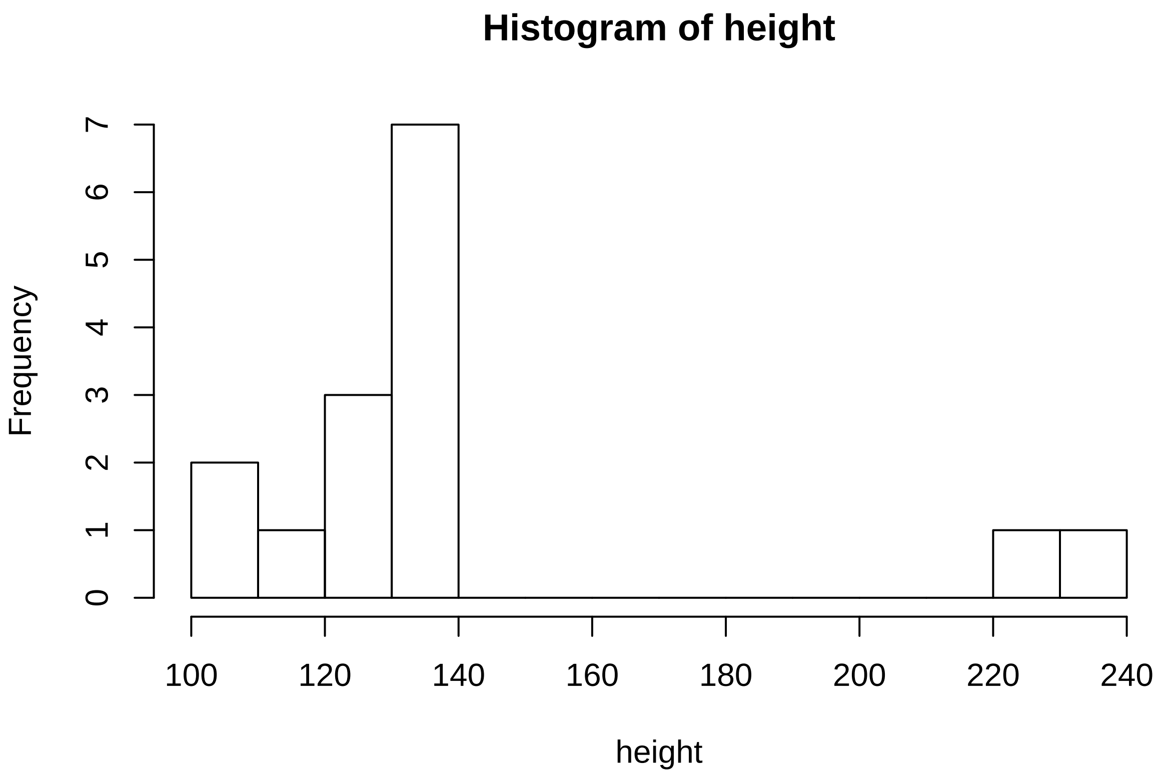
Let's go back to the heights dataset. What's the mean height of our respondents?

```
mean(height)
```

```
## [1] 140.7333
```

On average our respondents were about 140cm. Fair enough, right? Let's have a look at our histogram again

```
hist(height, breaks = 10)
```



To me that doesn't seem quite right, because most of the respondents were less 140cm tall. The mean is getting pulled upwards by those two very tall people.

This type of data is known as skewed data. Data is skewed when a small about of it is much higher or lower than the general trend. When you have skewed data it is often better to use a different measure of average known as the median.

```
median(height)
```

```
## [1] 132
```

I would argue that 132cm is a fairer measure of of the typical height for these respondents. When you have skewed data the mean and the median will often be very different. It's often a question of judgement which is the better measure. This is why it is important to see plots of your data; they will allow you to understand the true shape of the data and summarise it fairly.

Exercise

A company has three regular employees and a CEO. The CEO gets paid £120,000 per year. The three other employees all make £24,000 per year.

1. What is the mean salary of all workers at this company? What is the median salary?
2. Which average salary would it be fair to put on an advert for a new employee?

The maths

How does R calculate a median? This is also very simple. The median is just the middle value when we write out all the values in order. So, to find the median for 2, 4, 6, 2, 5, first write the values out in order:

2, 2, 4, 5, 6

And find the middle value.

2, 2, **4**, 5, 6

So here the median would be 4.

If the middle values is between two numbers, just take the mean of the numbers on either side. So, for this example:

2, 2, **4**, **5**, 5, 6

The median is $\frac{4+5}{2} = 4.5$

Exercise

Calculate the median, by hand, for the three sets of numbers below:

1. 2, 2, 2
2. 4, 6
3. 2, 2, 5

Which sets of numbers have the same mean and median?

Standard deviation

I'm trying to decide what film to watch tonight. I've got two options and they've both been rated 6 times.

> First film: 2 stars, 3 stars, 3 stars, 3 stars, 3 stars, 4 stars
>
> Second film: 1 star, 1 star, 1 star, 5 stars, 5 stars, 5 stars

The mean rating for both films is 3 stars. But, there's clearly something different about these two films. The first film has 6 people agreeing that it is okay. The second film is very divisive. Half the reviewers hated it and half the reviewers loved it.

There's more to a dataset than just the mean; we need some type of statistic that captures the amount of agreement in a dataset. We often call the amount of agreement/disagreement the variability of a dataset. And like the average there's a few options for how to measure this. Your first option is the standard deviation. Like all statistics we have seen so far, R makes this very easy to calculate.

```r
film_1 <- c(2, 3, 3, 3, 3, 4)
film_2 <- c(1, 1, 1, 5, 5, 5)

sd(film_1)
```

```
## [1] 0.6324555
```

```r
sd(film_2)
```

```
## [1] 2.19089
```

But, like the mean, the standard deviation doesn't work so well on skewed datasets. We have the equivalent of the median for measuring variability and it is known as the interquartile range. Below is how to use it in R:

```r
IQR(film_1)
```

```
## [1] 0
```

```r
IQR(film_2)
```

```
## [1] 4
```

The maths

The standard deviation is basically the average difference from the mean, across all values in your dataset. Since we don't care if the difference from the mean is positive or negative, we square all the differences from the mean. To counteract every value being squared we then take the square root of the final answer.

$$sd = \sqrt{\frac{\sum(x - \bar{x})^2}{n}} \tag{3}$$

We don't need to take the final square root. If we leave it out, we have another measure of variability called the variance. This is also very easy to calculate in R.

```r
var(film_1)
```

```
## [1] 0.4
```

You will often see the standard deviation and variance formulas calculated with $n - 1$ on the bottom of the fraction. Indeed, that is the formula R is actually using to calculate the standard deviation and the variance. We will discuss why this is later in the section on samples and populations.

The interquartile range is calculated in a similar way to the median. We put all our values in order:

1, 1, 1, 5, 5, 5

First we find the median:

1, 1, **1**, **5**, 5, 5

Here it's three. Now, for each half of the data on each side of the median, we find another median:

1, **1**, 1, 5, **5**, 5

The difference between those two medians (1 and 5) is the interquartile range (4).

Exercise

A. Look at the following 4 ways of describing a dataset. What are the advantages and disadvantages of each?

1. Raw data.
2. A frequency table.
3. Descriptive summary.
4. A plot.

B. Going back to our spinach opinion and height data, we also asked for opinions on chocolate. We used the same scale where -3 hate, 0 is neutral and 3 is love. Here are the raw data:

3, 3, 0, 3, -3, 3, 0, 2, 2, 2, 3, 3, 2, 3, 1

Create a frequency table, a descriptive summary and a plot for this data. Write a couple of sentences about your conclusions on chocolate opinion.

Answers

A. 1. No information lost. Not very easy to understand.

2. Again, no information lost. Can still be difficult to understand - particularly if very few of the values are the same.
3. Very compact. Answers questions we have about the data. Important information about the shape and spread of the data is lost.
4. Easy to understand overview of the data. Can often show us the shape of the data well. But information may be lost and it can be difficult to get raw numbers out.

Statistical Inference P values One page Confidence Intervals Comparing to a target – one sample t test Comparing two groups Comparing several groups One way ANOVA two pages Association Chi square one page Correlation two pages Regression – simple linear – upto 4 independent variables three pages Interrogating a data set three pages