Module_2_3

Keaton Wilson

Using simulations

Today, we're going to be talking about simulating data. Not in a 'making up data' unethical framework, but using our assumptions and knowledge about to build simulated data, and then comparing that to data we've collected. This has a number of benefits over just comparing to summary statistics of data…

We'll talk about all of this in a second.

Scenario - Research is costly

So here is the basic scenario. We found a correlation between eating fish and people getting sick, but that isn't really the root of the problem. We don't actually know if rates of disease are present above average levels in the tanks, and we definitely don't know what kinds of factors are contributing to this problem (if there is one!).

Our first challenge is that we can't sample the disease rate in all tanks - it's too expensive and it takes too long. So we're going to task our aqauaculture scientists to take a sub-sample (50 tanks). Our aquaculture scientists have shared this data with us, and we can access it here:
https://tinyurl.com/yf3pv3am.

Let's scope it out:

```
library(tidyverse)
```

```
## -- Attaching packages -------------------------------------------------------------------- tidyverse
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ----------------------------------------------------------------------- tidyverse_confl
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
sick_fish = read_csv("https://tinyurl.com/yf3pv3am")
```

```
## Parsed with column specification:
## cols(
##   tank_id = col_double(),
##   species = col_character(),
##   avg_daily_temp = col_double(),
##   num_fish = col_double(),
##   day_length = col_double(),
##   tank_volume = col_double(),
##   num_sick = col_double(),
##   size_day_30 = col_double()
## )
```

```
glimpse(sick_fish)
```

```
## Observations: 50
## Variables: 8
## $ tank_id        <dbl> 331, 580, 925, 996, 472, 171, 903, 777, 279, 734, 91...
## $ species        <chr> "tilapia", "tilapia", "trout", "trout", "tilapia", "...
## $ avg_daily_temp <dbl> 24.82100, 23.62557, 14.01980, 14.69082, 23.82299, 23...
## $ num_fish       <dbl> 99, 98, 89, 99, 104, 97, 75, 84, 94, 99, 102, 100, 1...
## $ day_length     <dbl> 9, 10, 12, 11, 10, 10, 14, 12, 8, 10, 12, 11, 11, 10...
## $ tank_volume    <dbl> 399.7492, 399.4314, 400.6617, 399.9064, 400.3092, 39...
## $ num_sick       <dbl> 0, 7, 14, 19, 1, 2, 18, 12, 8, 13, 19, 18, 3, 16, 4,...
## $ size_day_30    <dbl> 2779.6720, 2780.1393, 147.5595, 149.5450, 2789.2763,...
```

Group challenge

What is the mean number of sick fish for each species (both in terms of numbers and percentage of the entire tank?

```
sick_fish %>%
  group_by(species) %>%
  summarize(mean_sick = mean(num_sick),
            mean_perc = mean(num_sick/num_fish),
            n = n())
```

```
## # A tibble: 2 x 4
##   species mean_sick mean_perc     n
##   <chr>       <dbl>     <dbl> <int>
## 1 tilapia       6      0.0597    33
## 2 trout        12.8    0.146     17
```

Justifying a simulation

This seems high, at least for trout, but it could be that it's high because of a biased sample, right? Maybe we just happened to pick tanks (17 out of 250), with higher numbers. Can we simulate some draws based on an acceptable disease percentage and see what we end up getting and then compare this to what our aquaculture scientists provided us?

```
# We know our expected outcomes for disease rates in tanks:
# Tilapia = 4%
# Trout = 9%

# Let's generate some data for tilapia to start
# How many fish do we have in each tank? Well, it varies, but on average, it's...
fish_tank_data = read_csv("https://tinyurl.com/tbyskxl")
```

```
## Parsed with column specification:
## cols(
##   tank_id = col_double(),
##   species = col_character(),
##   avg_daily_temp = col_double(),
##   num_fish = col_double(),
##   day_length = col_double(),
##   tank_volume = col_double(),
##   size_day_30 = col_double()
## )
```

```r
fish_tank_data %>%
  group_by(species) %>%
  summarize(mean_num_fish = mean(num_fish))
```

```
## # A tibble: 2 x 2
##   species mean_num_fish
##   <chr>           <dbl>
## 1 tilapia          99.8
## 2 trout            74.2
```

```r
# About 100 for tilapia and 74 for trout.
# So 4% of 100 is 4, we expect about 4 sick for each tank, but this is on
# average...
```

The normal distribution

So, we want to build a simulation that is 4 fish sick on average from a tank, but we want to realistically simulate natural variation across fish tanks. There aren't going to be exactly 4 fish sick in every tank if we have our expectations. We can simulate this using something called the normal distribution, which is just a bell curve.

There is a lot going on here that you don't need to worry about. We have standard deviations across the x-axis, and the percentages of where the data fall across the top. The mean/median of the data falls at the center of the peak. The big takeaway here is that we can use this distribution to model our data… most of the data will fall close to the mean we set, with a fewer random points further outside.

```r
# We can use the function rnorm to randomly draw values from a normal
# distribution that we set up ahead of time.

# Let's just plot a normal distribution
x = seq(-10, 10, length = 500)
y = dnorm(x, sd = 1)
y2 = dnorm(x, sd = 4)

plot(x, y, type = "l")
lines(x, y2, type = "l", col = "blue")
```

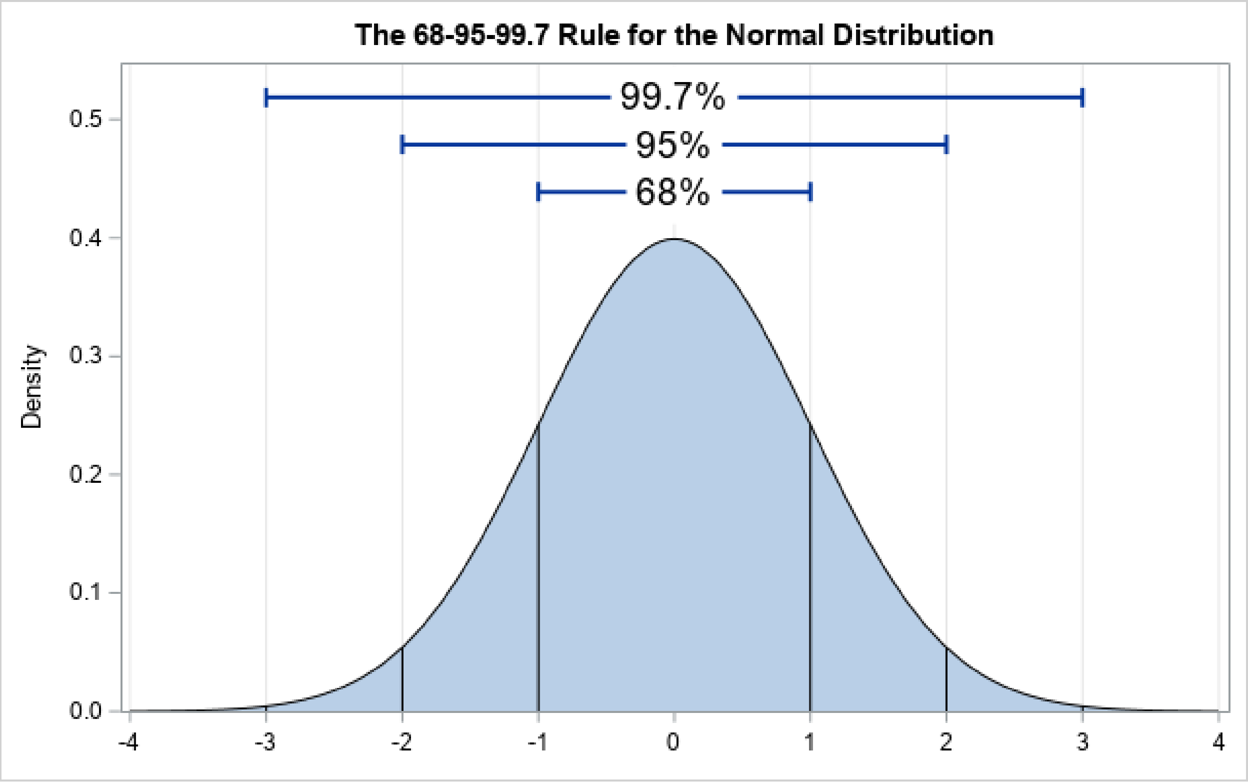
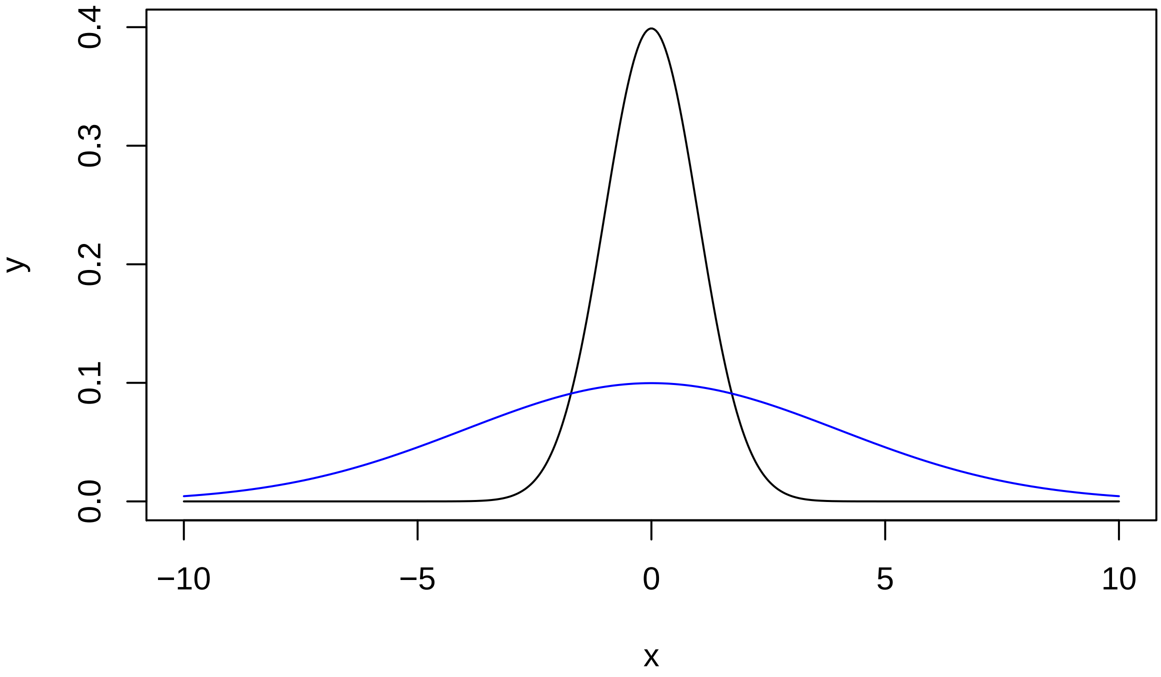


Figure 1: Normal Distribution



```
#Discussion of what this means when we're simulating data
rnorm(n = 1, mean = 4, sd = 1)
```

```
## [1] 4.016895
```

```
sick_tilapia_sim = rnorm(n = 33, mean = 4, sd = 1)
sick_tilapia_sim = round(rnorm(n = 33, mean = 4, sd = 2))
sick_tilapia_sim
```

```
##  [1] 5 7 5 1 6 4 5 4 2 5 5 2 6 2 2 5 6 2 4 4 4 3 4 5 2 2 5 4 2 7 6 6 3
```

```
mean(sick_tilapia_sim)/100
```

```
## [1] 0.04090909
```

Group Challenge

Perform the same thing above, except:
1. Do it for trout instead of tilapia
2. Make a function instead of just ad lib like I did
3. Test your function to make sure it works

```
fish_sick_simulator = function(mean = NULL, species = "trout", sd = NULL){
  if(species == 'trout'){
    sick_fish_vec = round(rnorm(n=17, mean = mean, sd = sd))
    return(mean(sick_fish_vec)/74)
  } else {
    sick_fish_vec = round(rnorm(n=33, mean = mean, sd = sd))
    return(mean(sick_fish_vec)/100)
  }
}

# 9% of 74 for trout...
74*0.09
```

```
## [1] 6.66
```

```
fish_sick_simulator(mean = 6.66, species = "trout", sd = 2)
```

```
## [1] 0.08903021
```

Followup questions:

1. Why is the number different every time we run the simulation?
2. Given that this is the case, what could we do to make our simulation more realistic?