# R Assignment 2 SOLUTIONS

Saving values in R

Schwab

2024-12-26

## Introduction/ Goal

In this R Assignment you will learn how to do some computation and save the output of that computation. This process mimics the work you might do on paper and pencil when working on homework, but saves it so you can work with the values you computed later. It has several benefits over traditional paper and pencil work: we will not round too soon, we can reuse code when doing a similar problem later, and we can show someone our steps easily if asked.

## The environment pane

Find and click on your environment pane. It is probably empty or only has a few things in it. Either way click the little broom to clear it.

The environment pane tells you which values are saved and how to reference them.

For example if we want to calculate the mean of the age variable from yrbss (we did this in R Assignment 1) and save it as yrbss_age_mean we can do that with the code below.

```r
# This code calculates the mean
mean(yrbss$age, na.rm=TRUE)
```

```
## [1] 16.15704
```

```r
# This code saves the mean as yrbss_age_mean.
yrbss_age_mean = mean(yrbss$age, na.rm=TRUE)
```

Notice that in the environment the value of yrbss_age_mean appeared. We can use this number by typing yrbss_age_mean in a chunk and running it.

```r
yrbss_age_mean
```

```
## [1] 16.15704
```

We can even do math with it. Let's multiply it by 2 and subtract 1 from it.

```r
yrbss_age_mean*2-1
```

```
## [1] 31.31408
```

## Making variable names

In general we want a few things for a variable name:

1. The name should be clear, so we know what it represents.
2. It should be short, (I will make my variable name longer than normal in this class for clarity).
3. It should not be the name of other functions used in R.

The last one is important. If we saved the mean() of the age variable as the word “mean”. We would not be able to use the mean function without a warning until we clear our environment. Sometimes this will break the function and cause an error.

### Exercise:

In this exercise you will find the lower and upper fences for outliers from data. Specifically the Penelope data. This data has people’s guesses for the weight of a cow named Penelope at a fair. While calculating these fences you will save values, so they do not need to be retyped. You will save your variables as you go. And when calculating outlier fences in the future you can come back to this document and reuse the code.

Recall from chapter 5.5 that the upper and lower bounds (or fences) are 1.5xIQR above Quartile 3 and below Quartile 1. They can be found using

Lower_Fence = Q1 - 1.5xIQR and

Upper_Fence = Q3 + 1.5xIQR

1. But first you should get antiquated with the penelope data.

```r
# Use the ? operator to bring up the documentation for penelope.
?penelope
```

2. We’ll need the first and third quartiles of the data. I’ll calculate the first quartile, you calculate the third. Save it as Q3. We can calculate specific quantiles with the quantile function.

```r
#In the line below I am finding the first quartile of the weight variable and saving it as Q1
Q1 = quantile(x = penelope$weight, probs = 0.25, names = FALSE)

#You can use the code above to calculate the third quartile and save it as Q3. You only need to change
Q3 = quantile(x = penelope$weight, probs = 0.75, names = FALSE)
```

3. We need to calculate the IQR also. We have two choices: we can use the built in IQR() function, or simply subtract Q1 from Q3, since we’ve calculated those values already. Either way we should save the result as IQR_penelope.

```r
# calculate and save the IQR. Save the number as IQR_penelope.
IQR_penelope = IQR(penelope$weight, type = 7)
# or
IQR_penelope = Q3-Q1
```

4. Now we have to put it all together to find the upper and lower bounds. We’ll use the * for multiplication. We do not need to save this value so we wont. We will print out the words “lower bound” and “upper bound” so the answer is easier to read.

You calculate the upper bound.

```r
# Lower bound
print("lower bound")
```

```
## [1] "lower bound"
```

```r
Q1 - 1.5*IQR_penelope
```

```
## [1] -44.25
```

```r
# Upper bound
print("upper bound")
```

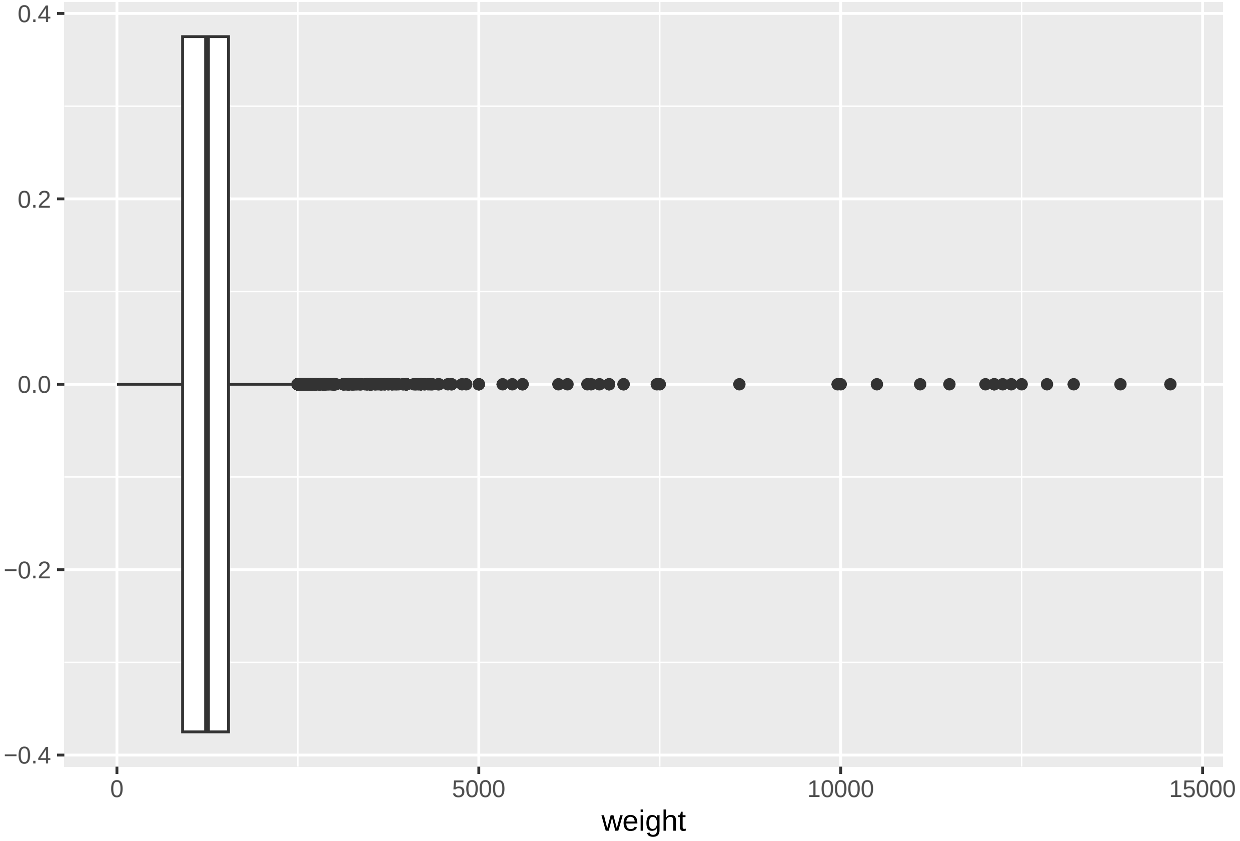
```
## [1] "upper bound"
```

```
Q3 + 1.5*IQR_penelope
```

```
## [1] 2493.75
```

That is it, you've saved values as variables and calculated the bounds for the outliers of the data set. We can also visualize the outliers with a boxplot. The boxplot will show outliers as little dots. Run the chunk below.

```
ggplot(data = penelope)+
  geom_boxplot(aes(x = weight))
```


weight