Session 3: Data Visualization in R

Objectives:

By the end of this session, participants will be able to perform following tasks:

- Use of tapply() for aggregation
- Make bar plot from matrix
- Make changes in the graph
- Make barplot with plus minus error bars
- Make scatterplot by two methods

In this session, we will be understanding how to visualize data in R. The first step is to read data available in csv file.

```
records <- read.csv("breastcancer.csv")
```

Apply tapply() on BMI column to calculate the length, mean and sd for Cancer and non-Cancer patients, as we did for Age column.

```
n.BMI <- tapply(records$BMI, list(records$Classification), length)

mean.BMI <- tapply(records$BMI, list(records$Classification), mean)

sd.BMI <- tapply(records$BMI, list(records$Classification), sd)
```

3.1 Barplot

Now make the barplot of n.BMI, mean.BMI, and sd.BMI.

```
barplot(n.BMI)

barplot(mean.BMI)
```

Now add the axis labels in the above functions for elaboration

```
barplot(n.BMI,

        xlab = "Cancer Classification (1/2)",

        ylab = "Count of Patient")

barplot(mean.BMI,

        xlab = "Cancer Classification (1/2)",

        ylab = "BMI")
```

Now add the vertical axis limit according to the need of the graph.

```
barplot(n.BMI,
        xlab = "Cancer Classification (1/2)",
        ylab = "Count of Patient",
        ylim = c(0,?))
barplot(mean.BMI,
        xlab = "Cancer Classification (1/2)",
        ylab = "BMI",
        ylim = c(0,?))
```

Exercise 3.1

a) Make barplot of mean of Glucose for cancer and non-cancer patients.

3.1.1 Barplot with plus minus error bars

Plot mean ± standard deviation of BMI for cancer and non-cancer patients. We can make a barplot with standard deviation error bars, and to be fancy, let's place as text inside each bar the sample size used to calculate the mean and standard deviation.

First, we need to calculate the means, the standard deviations, and the sample sizes. We have already done it. Secondly, make the barplot for mean which we also have already done above. Now, let's add error bars. The barplot function does not have an automatic set of tools to do this. Here are other packages that do offer this facility. They include Gregory Warnes ***gplots***, Frank Harrell's excellent Hmisc package, Jim Lemmon et al.'s ***plotrix*** package, and Hadley Wickham's ***ggplot2*** package.

The function we are going to use to put error bars onto a barplot is arrows. Here is how it works. The arrow (error bar) needs to go from the top of the error bar down to the bottom. The top of the error bar will be at the mean plus the error, and the bottom at the mean minus the error. So these are the y-coordinates of the error bar arrow: mean + error and mean - error. What are the x-coordinates of the error bar? The x-coordinates of any error bar are the location of the middle of the bars on the x-axis. So all we need to do is get the mid points of the treatments (bars) on the x-axis.

Here is how to do it. We assign the value returned by barplot to an object called mids.

```
mids <- barplot(mean.BMI,
                xlab = "Cancer Classification (1/2)",
                ylab = "BMI",
                ylim = c(0,50))
```

Take a moment to explore the values returned by mids (type it in the console after running it). You'll notice that the object actually contains the x-axis midpoints of the bars—the location of the bars on the x-axis.

Now, check out the help file for *arrows()*. This function requires, as arguments, x0, y0, x1, and y1. x0 and y0 are the x–y coordinates of the start of the arrow and x1 and y1 are the x–y coordinates of the endpoints of the arrow. The keen practitioner will recognize that x0 and x1 are the same for an error bar—the x-coordinates for a vertical line are the same. And they are the midpoints of the bar. Thus, we can use mids as x0 and x1.

But what do we use for y0 and y1? As mentioned before, let's start our error bars below the mean and finish above the mean. The point below the mean is the mean minus the standard deviation: mean.BMI - sd.BMI. The point above the mean is the mean plus the standard deviation: mean.BMI + sd.BMI.

Try typing these (e.g. mean.BMI - sd.BMI) into the console—notice that R performs the calculations for both columns simultaneously and automatically.

In this example, we have two treatments. The matrix of means has two columns and one row, and the matrix of standard deviations is the same dimension. Subtracting the standard deviations from the means results in a matrix of the same dimension and returns two values—the y0 and y1 values for each bar.

```
arrows(mids, mean.BMI - sd.BMI, mids, mean.BMI + sd.BMI,

       code = 3, angle = 90, length = 0.1)
```

There are three arguments to explain in *arrows()*. code = 3 tells arrows to draw "arrows" at both ends of the line. angle = 90 says make the arrow heads 90 degrees—flat. length = 0.1 makes them short (0.1 mm).

Our final objective was to put some text in the bars to indicate the sample size. Again, not surprisingly, the function to add text to a graph is text. Like arrows, it takes as an argument a location (x, y) and a text string. As with arrows, it is important to recognize that an easy location to place the text is centred in the bars—the mids.

```
text(mids, 2, paste("n=", n.BMI))
```

Thus the final code looks like the following:

```
# Make a barplot using the means returned from tapply

n.BMI <- tapply(records$BMI, list(records$Classification), length)

mean.BMI <- tapply(records$BMI, list(records$Classification), mean)

sd.BMI <- tapply(records$BMI, list(records$Classification), sd)


mids <- barplot(mean.BMI,

                xlab = "Cancer Classification (1/2)",

                ylab = "BMI",

                ylim = c(0,50))

# Use Arrows to put error bars on the plot
```

```
arrows(mids, mean.BMI –
       sd.BMI, mids, mean.BMI + sd.BMI,
       code = 3, angle = 90, length = 0.1)
# Add text at the midpoints (x) and at height 2 on the y-axis
# paste the words n=followed by the n.BMI values
text(mids, 2, paste("n=", n.BMI))
```

3.2 Scatterplots

Now make scatterplot of Glucose (x-axis) and Insulin (y-axis) by Method I.

```
plot(records$Glucose, records$Insulin)
```

Now make scatterplot of Glucose (x-axis) and Insulin (y-axis) by Method II.

```
plot(Insulin ~ Glucose, data = records)
```

Exercise 3.2

Make scatterplot of Leptin (x-axis) and Adiponectin (y-axis) by both methods.