Intro to R for Decision Modeling

Plotting

SickKids and DARTH

11/2/2020

Change `eval` to `TRUE` if you want to knit this document.

This goal of this worksheet is to introduce and provide guidance on visualizing data using `ggplot2` and base R. It is separated into the following sections:

1. Plotting graphs in `R` using `ggplot2`
2. Plotting graphs in `R` using base `R` capabilities

Throughout the course, we will demonstrate code and leave some empty *code chunks* for you to fill in. We will also provide solutions after the session.

Feel free to modify this document with your comments and clarifications.

1.0 Load data into `R`

Before we begin this session, we need to load The Framingham dataset into `R` as we did in the previous session. The Framingham dataset we have provided has 3,263 rows. To demonstrate plotting, we will use a the first 1000 entries from this dataset.

```
data <- read.csv('framingham.csv', header = TRUE)
# Selecting the first 1000 Ids
data_small <- data[1:1000, ]
```

1. Plotting graphs using `ggplot2`

The package `ggplot2` is a powerful visualization tool within `R`. We can use ggplot to create high-quality publication-ready graphs.

All ggplot graphs contain four key components:

1. A `ggplot()` function which outlines what data will be plotted
2. An `aes()` function inside `ggplot()` that determines what parts of the graph will depend on the data.
3. A `geom_` function indicating the type of graph we are creating.
4. Graph options not related to the data (placed outside the `aes()` function).

1.1 Starting off with Scatterplots

Before we begin plotting, it is important that the `ggplot2` package is loaded in `R`.

```
library(ggplot2)
```

Let's start by creating a plot with Age (`AGE`) on the x-axis and Systolic Blood Pressure (`SYSBP`) on the y-axis.

```
ggplot(data = data_small,
       aes(x = AGE,
           y = SYSBP))
```

If we do not specify a `geom_` function we get an empty plot. We can think of `ggplot()` function as creating a base layer which we will be adding geometry to. We add `geom_` to the original `ggplot()` using the `+` sign.

Lets add a scatter plot to the graph we created above.

```
ggplot(data = data_small,
       aes(x = AGE,y = SYSBP))  +
  geom_point()
```

EXERCISE 1 Create a scatter plot of `SYSBP` against `BMI`.

```
# Your turn
```

There are several asthetic options to customize scatter plots.

- **color**: Point colour (a variable name if inside `aes()` or the name of a colour "purple" or HEX codes "#800080")
- **shape**: shape of point (a variable name if inside `aes()` or a number between 1-25)
- **alpha**: transparency of each point (a variable name if inside `aes()`or a number between 0-1 with 0 being completely transparent and 1 being solid)

You can type `?geom_point()` to see the help file for `geom_point()` in the bottom right-hand corner. This will show the complete list of the asthetics assocaited with `geom_point`.

The code below returns a scatter plot of blue points that are 50% transparent.

```
ggplot(data_small,
       aes(x = BMI, y = SYSBP)) +
  geom_point( colour = 'blue',
              alpha = 0.5)
```

Note that if the asthetics should vary depending on the data, i.e. the colour changes due to the subgroup, then they should be included within the `aes()` function. For example, we can vary the colour of the points by age.

```
ggplot(data_small,
       aes(x = BMI, y = SYSBP, color = AGE)) +
  geom_point( alpha = 0.75)
```

EXERCISE 2 Make the colour vary in the graph of `SYSBP` against `BMI` based on Prevalent Hypertensive (PREVHYP).

```
# Your turn
```

In the plot above, you can see that there are two distinct categories for `PREVHYP` but `ggplot` has used a colour scale from 0 to 1. This is because `ggplot` treats numbers as numerical variables rather than understanding that the numbers code two distinct categories (`0` means free from hypertension and `1` means the participant has hypertension). Therefore, to improve the plot, we need to use the `factor()` function to turn `PREVHYP` into a categorical variable:

```
data_small$PREVHYP.fct <-  factor(data_small$PREVHYP,
                                  levels = 0:1,
                                  labels = c('Free of disease',
                                             'Prevalent disease'))
```

EXERCISE 3 Plot a scatter plot with two variables from the Framingham dataset while making colour vary based on `PREVHYP.fct`.

```
# Your turn
```

1.2 Labeling plots & Themes

The `ggplot2` package contains several functions to change titles and labels of plot elements. - `xlab()` & `ylab()` change axis labels. - `ggtitle()` adds a title and a subtitle

```
ggplot(data_small,
       aes(x = AGE, y = SYSBP, color = factor(PREVHYP.fct))) +
  geom_point(alpha = 0.75) +
  xlab(label = "Age, Years") +
  ylab(label = "Systolic Blood Pressure") +
  ggtitle(label = "Figure 1: Age and Systolic Blood Pressure",
          subtitle = "Stratified by prevalence of hypertension")
```

We can also use functions of the form scale_*asthetic*_*type* to change the legend for arguments passed in `aes()`. These different functions are named so the *aesthetic* is the aesthetic that we want to name, e.g. color, fill shape, alpha, and *type* is the format of the data that is plotted with that aesthetic e.g. continuous, discrete, date.

For example, to change the name of the legend associated with the prevalence of hypertension, we use `scale_color_discrete()`.

```
ggplot(data_small,
       aes(x = AGE, y = SYSBP, color = factor(PREVHYP.fct))) +
  geom_point(alpha = 0.75) +
  xlab(label = "Age, Years") +
  ylab(label = "Systolic Blood Pressure") +
  ggtitle(label = "Figure 1: Age and Systolic Blood Pressure",
          subtitle = "Stratified by prevalence of hypertension") +
  scale_color_discrete(name = "Prevalence of hypertension" )
```

We can also change the background of a plot using the `theme_` functions. For example, the default colour for a ggplot is grey (as we see above) but the following graphic has a white background with a black box around using the `theme_bw()` function. You can find more ggplot2 themes in the `ggthemes` package.

```
ggplot(data_small,
       aes(x = AGE, y = SYSBP, color = factor(PREVHYP.fct))) +
  geom_point(alpha = 0.75) +
  xlab(label = "Age, Years") +
  ylab(label = "Systolic Blood Pressure") +
  ggtitle(label = "Figure 1: Age and Systolic Blood Pressure",
          subtitle = "Stratified by prevalence of hypertension") +
  scale_color_discrete(name = "Prevalence of hypertension" ) +
  theme_bw()
```

We can also plot two different types of the same data using two alternative `geom_` functions with the same asthetics. For example, in the graphic below, we produce a scatterplot of the data using `geom_point()` and then add the fitted line from a linear model using the `geom_smooth()` plot type. For example:

```
ggplot(data_small,
       aes(x = AGE, y = SYSBP, color = factor(PREVHYP.fct))) +
  geom_point(alpha = 0.75) +
  geom_smooth(method = "lm") +
  xlab(label = "Age, Years") +
  ylab(label = "Systolic Blood Pressure") +
  ggtitle(label = "Figure 1: Age and Systolic Blood Pressure",
          subtitle = "Stratified by prevalence of hypertension") +
  scale_color_discrete(name = "Prevalence of hypertension" ) +
  theme_bw()
```

Axis limits

You can set alternative axis limits using `ylim()` and `xlim()`. This is done by providing the function a vector of length two indicating where the axis begin and end. Notice that each time we add an additional element to the plot, we add with a `+`.

```
ggplot(data_small,
       aes(x = AGE, y = SYSBP, color = factor(PREVHYP.fct))) +
  geom_point(alpha = 0.75) +
  geom_smooth(method = "lm") +
  xlab(label = "Age, Years") +
  ylab(label = "Systolic Blood Pressure") +
  ggtitle(label = "Figure 1: Age and Systolic Blood Pressure",
          subtitle = "Stratified by prevalence of hypertension") +
  scale_color_discrete(name = "Prevalence of hypertension" ) +
  ylim(c(0,300))+
  theme_bw()
```

Other graphs

EXERCISE 4: Create a box plot with `PREVSTRK.fct` (Prevalent stroke) on the x-axis and `SYSBP` (Systolic Blood Pressure) on the y-axis.

Note: `?geom_boxplot` provides the documentation for the boxplotting function.

```
# Your turn
```

EXERCISE 5: Create a histogram of Age stratified by Prevalent hypertension (`PREVHYP.fct`).

- `?geom_histogram` provides the documentation for the boxplotting function.
- `fill` allows you to change the colour of the histogram (more widely it changes the colour of geometry).

```
# Your turn
```

EXERCISE 6 Plot the density of age in the Framingham dataset.

```
# Your turn
```

2. Plotting graphs using base `R`

While `ggplot2` offers extensive customizability and publication-standard graphics, there are some basic plotting capabilities in `R` that are sufficient, and often preferred, for exploratory analysis.

Scatter plot

Suppose you want to create a scatter plot of two variables you can simply use the `plot()` function. The following command plots heart rate and Systolic blood pressure

```
plot(x = data_small$BMI,y =  data_small$SYSBP,
     col = 'blue',
     main = 'Scatter plot of systolic blood pressure against BMI',
     xlab = 'BMI', ylab = 'systolic blood pressure')
```

EXERCISE 7 Re-create the scatter plot you made in EXERCISE 3 using the base plotting function. Make appropriate titles and labels.

```
# Your turn
```

Box plot

The following code creates a box plot of `SYSBP` stratified by `PREVSTRK`.

```
boxplot(SYSBP ~ PREVSTRK ,data=data_small,
        main="Box plot of systolic blood pressure stratified
        by prevalent stroke",
        names = c('free of disease', 'prevalent disease'),
        col = c('red', 'blue'),
        xlab = "prevalent stroke", ylab="systolic blood pressure")
```

The `~` symbol is frequently used in `R` to define formulas (which we will see in the Regression session), but here indicates that you want to plot the continuous variable `SYSBP` across the two categories of `PREVSTRK`.

EXERCISE 8 Create a box plot of two variables in the Framingham database with the appropriate title and labels.

```
# Your turn
```

Histogram

The following command creates a histogram showing the distribution of `AGE` with appropriate labelling.

```
hist(data_small$BMI, main = 'Histogram of age', xlab = 'age')
```

EXERCISE 9 Create a histogram of BMI with smaller sized bins (hint: explore `?hist()` and the `breaks` option).

```
# Your turn
```

Other types of graphs

Several R functions have output which the `plot()` function understands intunitively how to visualize. For example the `density()` funciton calculates kernel density estimates of a vector. When we call the `plot()` function to the results of `density()` we get a meaninful plot.

Not all output from `R` functions can be plotted.

```
# Plotting output of (density)
den.age <- density(data_small$AGE)
den.age
plot(den.age, main = 'Density plot of age')
```