MANIPAL ACADEMY *of* HIGHER EDUCATION
(Institution of Eminence Deemed to be University)

MDS5103/ MSBA5104 Segment 02

GRAPHING USING GGPLOT2 – PART II TUTORIAL

Table of Contents

1. Geometric Elements in Ggplot2 3
2. Initialisation 3
3. Scatter Plot 4
4. Smooth Line Plot 5
    4.1 Aesthetic Attributes of Smooth Plot 7
    4.2 Layering of Plots 9
5. Bar Chart 10
6. Box Plot 11
    6.1 Aesthetic Attributes of Box Plot 12
7. Histogram 13
    7.1 Creating a Sample Dataset for Histogram 14
    7.2 Basic Histogram 15
    7.3 Changing the Binwidth 16
    7.4 Aesthetic Attributes of Histogram Plot 17
        7.4.1 Colour-Based on Another Feature 18
        7.4.2 Overlaying Two Histograms 18

1. Geometric Elements in Ggplot2

In ggplot2, geometric objects are graphical elements that are used to visualise data. They can be lines, bars, points, and so on. Data visualisation involves different graphical representations of data in the form of charts, graphs, and maps which makes the analysis of complex data easier. Data visualisation is helpful in data comparison, data analysis, understanding the data distribution over a period, and visualisation of a geographical dataset. In this topic, we will look at various geometric elements in detail.

2. Initialisation

In this topic, the “mpg” dataset will be used to explain the various geometric elements. “mpg” dataset contains a subset of the fuel economy data. The dataset contains the mileage information of car models that are released every year between 1999 and 2008. To begin with, let us load the dataset and display the first five rows. In R Studio, anything typed between the delimiters ```{r} and ``` is considered a piece of R code. The content without the delimiter is treated as text. Please note, the code segments in this document are not enclosed within the delimiters and the learner will have to add them during execution.

```r
# Load the mpg dataset
data('mpg')
mpgData = mpg
# Print the first five rows (or samples) in the data frame
head(mpgData, 5)
```

Output

```
      manufacturer   model   displ   year   cyl   trans /
      audi           a4      1.8     1999   4     auto(l5)
      audi           a4      1.8     1999   4     manual(m5)
      audi           a4      2       2008   4     manual(m6)
      audi           a4      2       2008   4     auto(av)
      audi           a4      2.8     1999   6     auto(l5)
```

Output

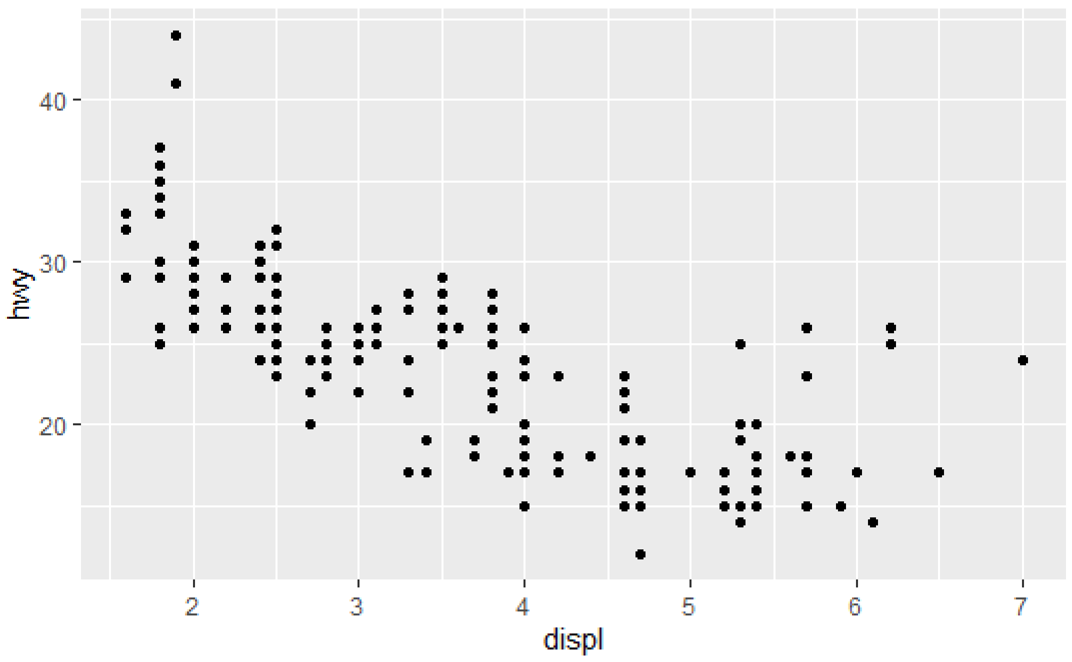
```
      manufacturer   drv   cty   hwy   fl   class
      audi           f     18    29    p    compact
      audi           f     21    29    p    compact
      audi           f     20    31    p    compact
      audi           f     21    30    p    compact
      audi           f     16    26    p    compact
```

3. Scatter Plot

As discussed previously, scatter plot is used to represent the relationship between two continuous features. For example, let us display the same to map the mileage of a car v/s its displacement.

```
# Plot a scatter plot of mileage w.r.t. displacement
p1 = ggplot(data = mpgData) +
  geom_point(mapping = aes(x = displ, y = hwy))
p1
```

Output



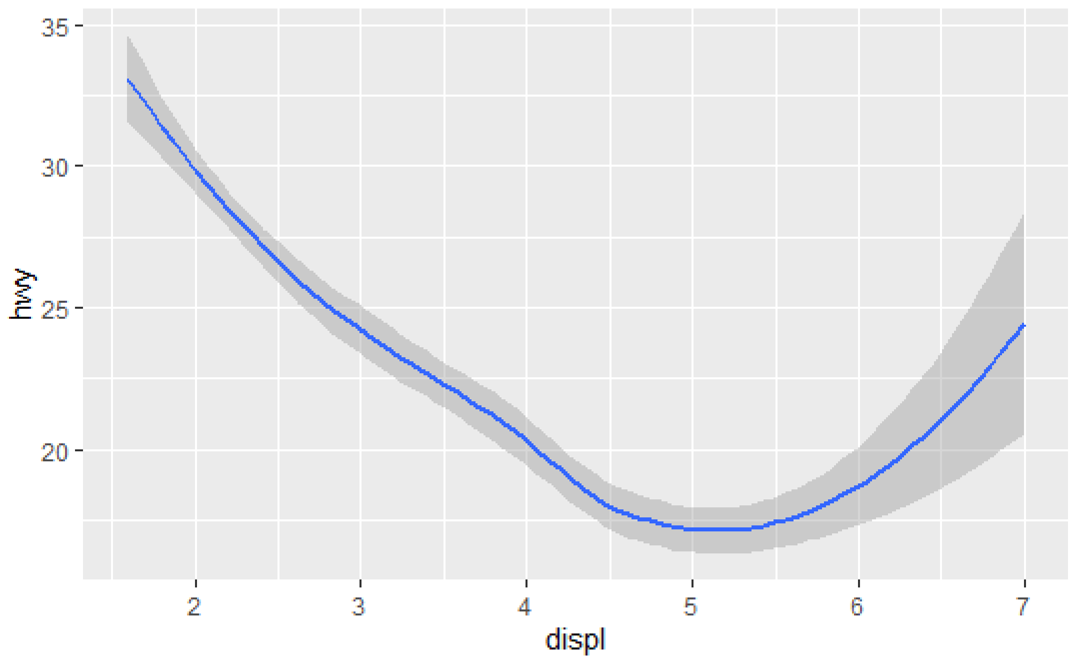
4. Smooth Line Plot

The next form of plotting data is to show a smooth curve as opposed to scattered points; we can use a smooth line plot. To do so, the syntax will change from "geom_point" to "geom_smooth".

*p2= ggplot(data = mpgData) + **geom_smooth**(mapping = aes(x = displ,y = hwy))*

```
# Plot a bar chart w.r.t. number of cylinders
p2= ggplot(data = mpgData) +
 geom_smooth(mapping = aes(x = displ,y = hwy))
p2
```

Output

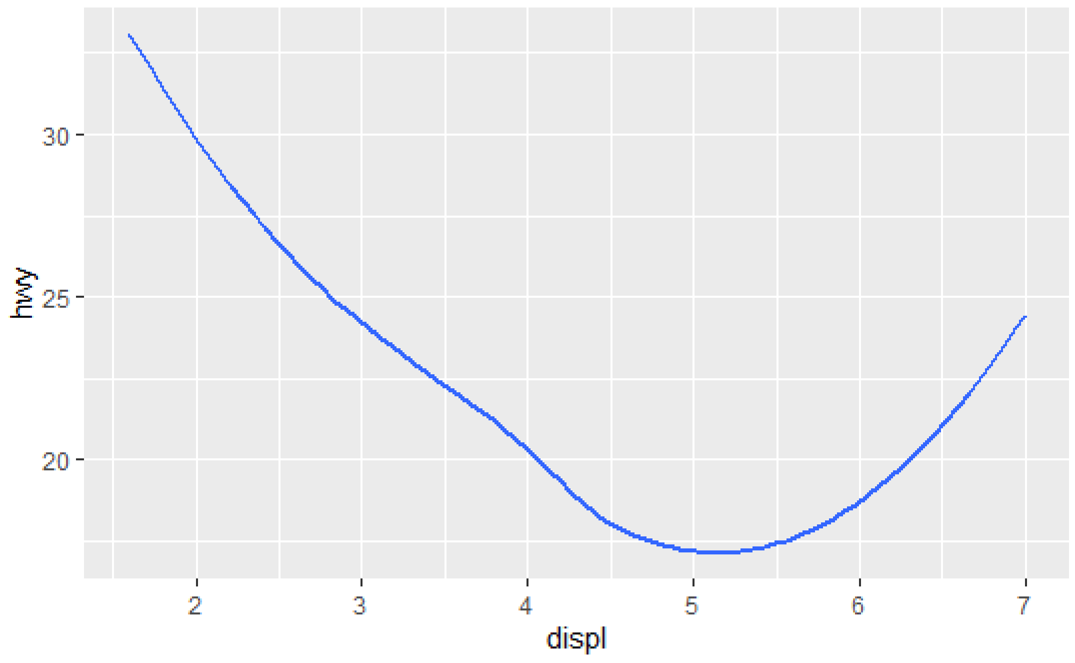


The scatter plot and smooth plot give us similar outputs; the difference is that the values are averaged to get a smooth line in the smooth line plot.

A blue line and a shadow can be observed in the graph. The shadow portrays the **standard error** in the dataset. To avoid displaying the shadow or the standard error (SE), it must be turned off in geom_smooth() function, as shown in the code below.

```
# Plot a bar chart w.r.t. number of cylinders
p2= ggplot(data = mpgData) +
 geom_smooth(mapping = aes(x = displ,y = hwy), se = FALSE)
p2
```

Output:



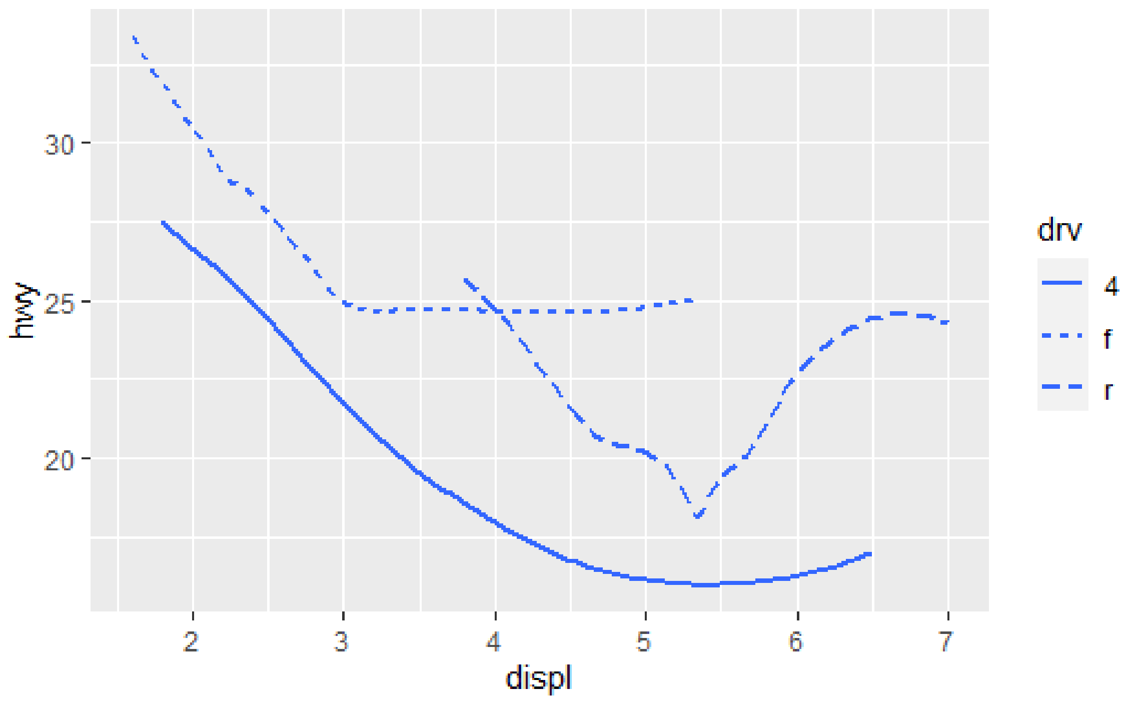
4.1 Aesthetic Attributes of Smooth Plot

Each geometric function has its aesthetics: for the point (scatter plot), there is an aesthetic option of the size which helps change the size of the points displayed on the plot.

Let us now consider the aesthetics of the line plot. We cannot use a different shape for a line/curve plot. However, we can use different line types—dotted line, straight line, etc. For the current dataset, let us use the "linetype" aesthetic to display the mileage-to-displacement plot for different drive types.

```
# Plot a smooth line plot of mileage w.r.t displacement for each drv type
p3 = ggplot(data = mpgData) +
  geom_smooth(mapping = aes(x = displ, y = hwy, linetype = drv), se = FALSE)
p3
```

Output



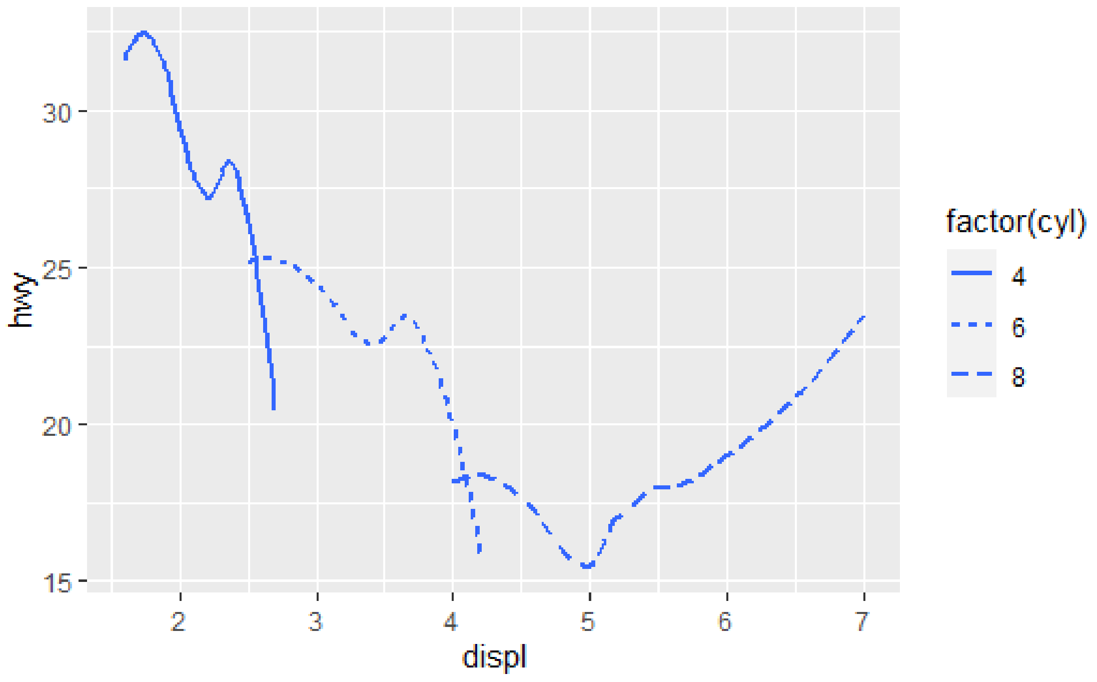
Based on the output of the graphs, there are three different lines:

- Solid line is for 4-wheel drive
- Dotted line is for forward-wheel drive
- Dashed line is for rear-wheel drive

The same type of plot can be plotted based on the cylinder type:

```
#  Plot a smooth line plot of mileage w.r.t displacement for each cylinder type
p3 = ggplot(data = mpgData) +
  geom_smooth(mapping = aes(x = displ, y = hwy, linetype = factor(cyl)), se = FALSE)
p3
```

Output



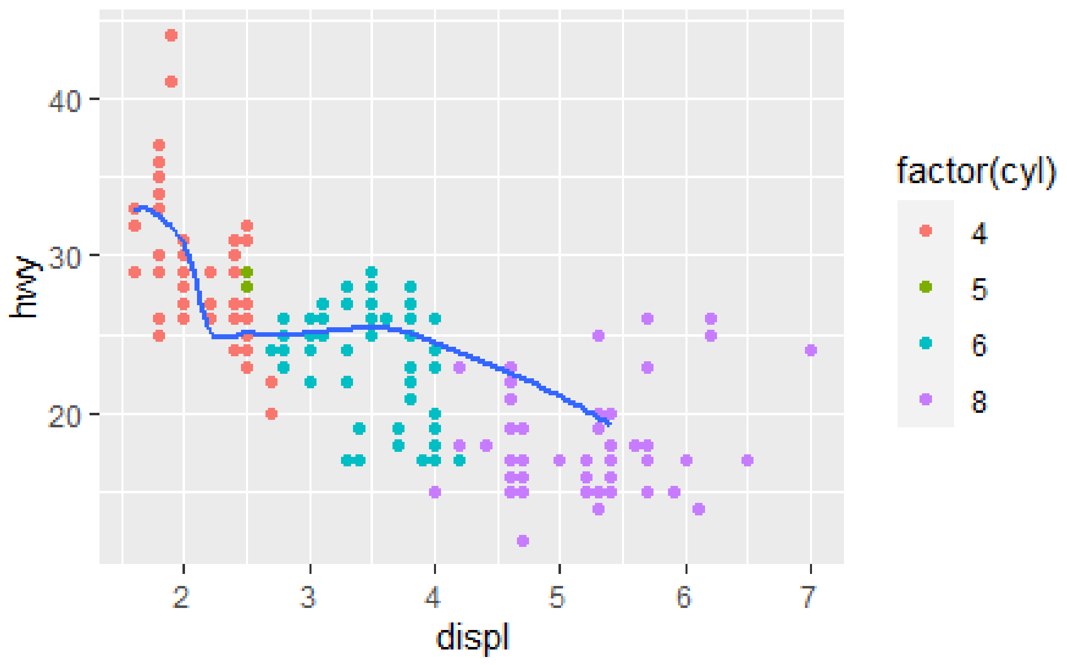
In the above output, the solid line represents the line for a 4-cylinder car, the dotted line for a 6-cylinder car, and so on.

4.2 Layering of Plots

As the data for line type can be defined locally, we can use local data for different layers (plot type, colour, etc.). In this example, we are plotting the mileage vs. displacement of the cars, but for the type of cylinders, point (scatter) plot is used. Colours are applied for differentiation and another layer of plot is added to indicate from where the data has to be taken; we specify this using filter function.

```
# Specify data for layers individually
# Plot mileage w.r.t displacement for all cars but add a smooth
# line only for subcompact cars by filtering
p4 = ggplot(data = mpgData, mapping = aes(x= displ, y = hwy)) +  geom_point(mapping =
aes(color = factor(cyl))) + geom_smooth(data = filter(mpgData, class == 'subcompact'), se
= FALSE)
p4
```

Output



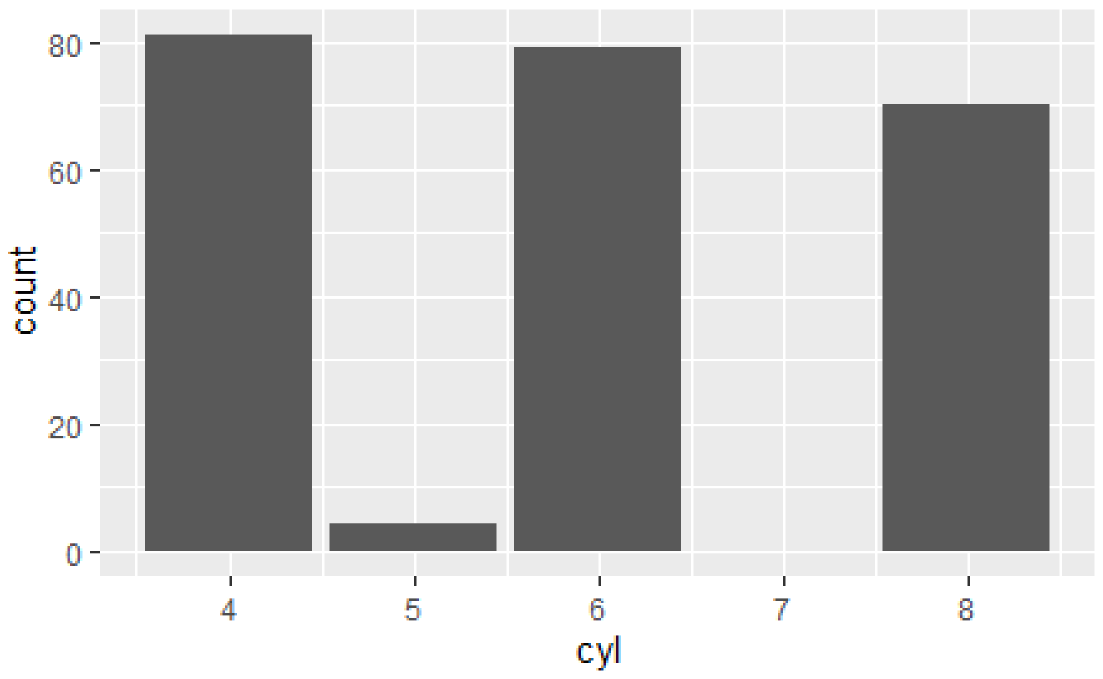
In the above output, the scatter plot represents the relationship between 'hwy' and 'displ'. The points are colour coded as per the type of the cylinder of the car. The blue line indicates the set of points that belong to the 'subcompact' class of cars. Thus, a lot of information can be captured in a single plot.

5. Bar Chart

Now, let us check how to plot bar charts. For plotting bar charts, **geom_bar()** function is applied here.

```
# Plot a bar chart w.r.t. number of cylinders
p5 = ggplot(data = mpgData) +
  geom_bar(mapping = aes(x = cyl))
p5
```

Output



6. Box Plot

A box plot is used to display the relationship between two features. A box plot is constructed based on a 5-number summary which includes minimum, first quartile (q1 | 25th percentile to median), median, third quartile (q3 | median to 75th percentile), and maximum. In *Figure 1*, you can see a different section of a box plot.

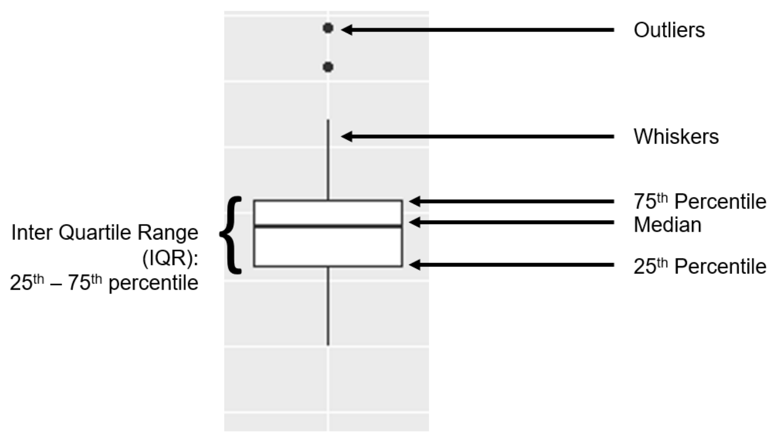
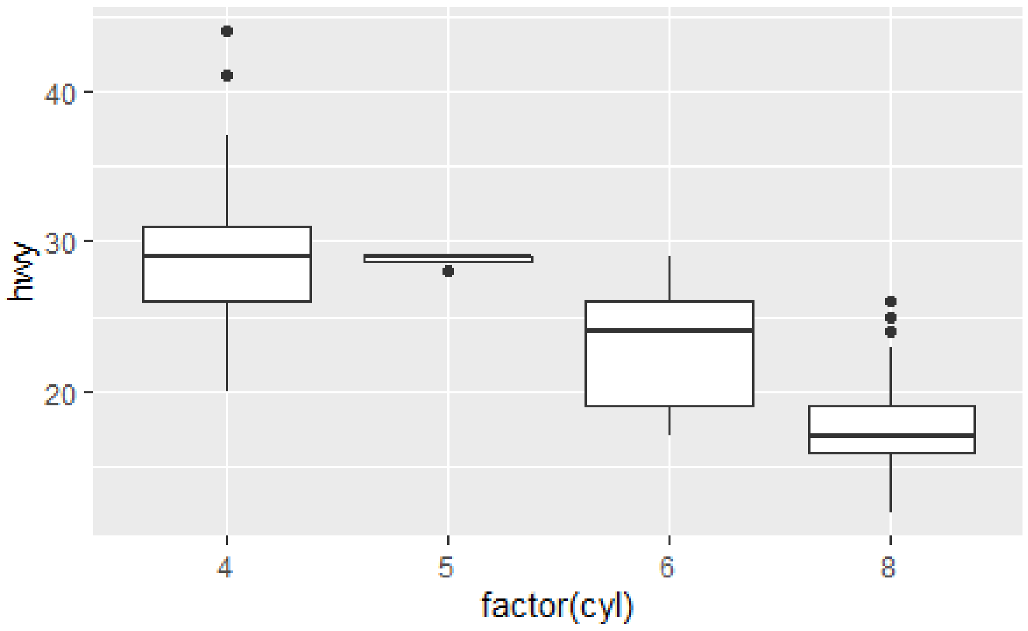


Figure 1: Box Plot

The code below demonstrates how the mileage of cars differs based on the number of cylinders. geom_boxplot() function is used here to plot a box plot.

```
# change outlier color, shape and size
p6 = ggplot(data = mpgData) +
 geom_boxplot(mapping = aes(x = factor(cyl), y = hwy))
p6
```

Output



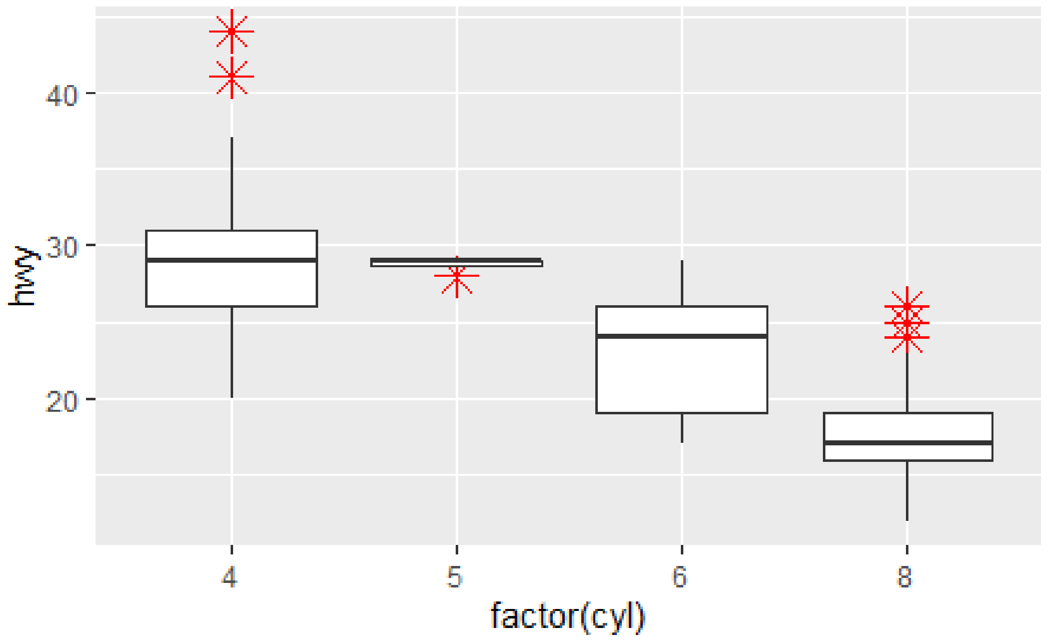
The box plot shows how mileage spread across the mean value based on the number of cylinders.

6.1 Aesthetic Attributes of Box Plot

As in the previous plots, aesthetic options are available for box plots as well; these include the options to change the colours, shapes, and sizes of the outliers. Let us examine how to apply these changes. In the code below, we set the outliers to red colour. We also change the shape and the size of the outliers.

```
# change outlier color, shape and size
p6 = ggplot(data = mpgData) +
  geom_boxplot(mapping = aes(x = factor(cyl), y = hwy))
p6
#change outlier color, shape and size
p6 = ggplot(data = mpgData) +
  geom_boxplot(aes(x=factor(cyl), y = hwy), outlier.colour = "red", outlier.shape = 8, outlier.size
= 4)
p6
```

Output



7. Histogram

The next plot is a histogram. A statistical plot that graphically represents the distribution of data, a histogram is represented by a set of rectangles adjacent to each other. In these sets, every bar represents a type of data from the dataset. Like the previous plots, the histogram is also a geometric object of ggplot2 library.

7.1 Creating a Sample Dataset for Histogram

To demonstrate a histogram using ggplot2, we will create sample data. A normal distribution of heights of males and females is considered here to build a histogram. For this, let us create two columns inside the dataframe object; one for gender and the other for height. In total, 400 samples of heights, which include data of an equal number of male and female samples, are inserted. To populate the gender column, the following command is used inside the dataframe object:

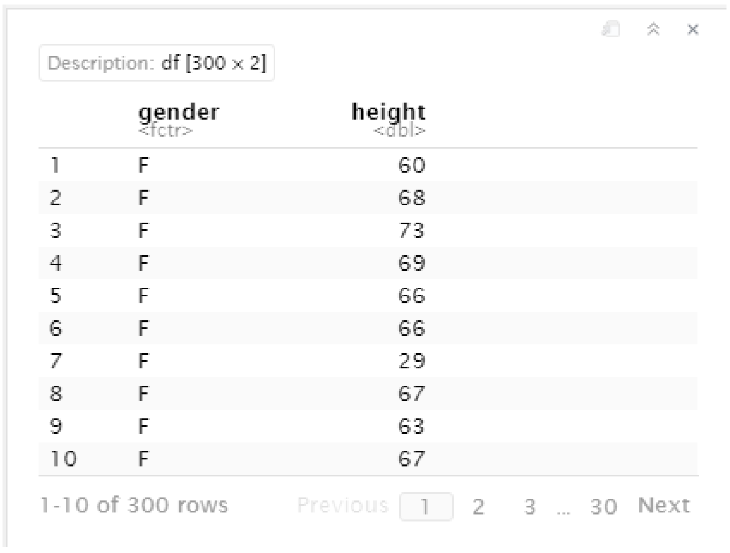
gender = factor(rep(c("F", "M"), each = 200))

And, to populate the height column of the data frame, the following command is used inside the dataframe object with the previous command:

height = round(c(rnorm(200, mean = 60, sd = 10), rnorm(200, mean = 70, sd = 6)))

```
#simulate a data set and store in a data frame
df = data.frame(
  gender = factor(rep(c("F", "M"), each = 200)),
  height = round(c(rnorm(200, mean = 60, sd = 10), rnorm(200, mean = 70, sd = 6)))
)
head(df, 300)
```

Output



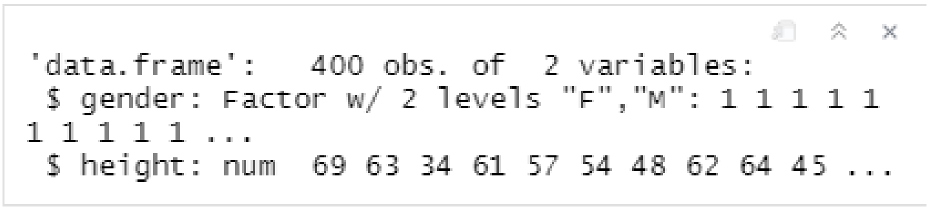
```
Description: df [300 x 2]

     gender     height
     <fctr>      <dbl>
1    F              60
2    F              68
3    F              73
4    F              69
5    F              66
6    F              66
7    F              29
8    F              67
9    F              63
10   F              67
1-10 of 300 rows    Previous 1 2 3 ... 30 Next
```

The dataframe developed is a simulated one for our convenience. We can investigate the structure of the variable to check the structure or the way the variables are developed. To do so, we can use the following command after the previous lines of code:

```
str(df)
```

Output

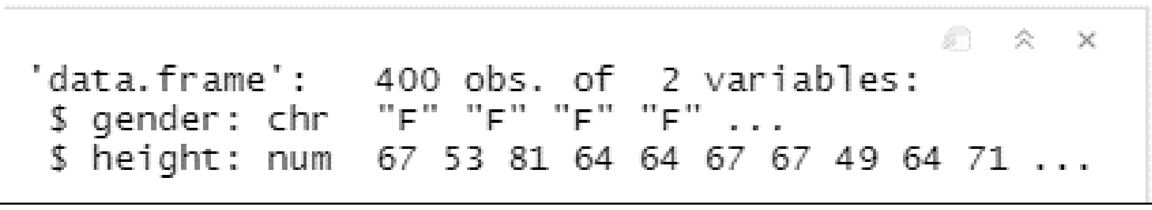


```
'data.frame':   400 obs. of  2 variables:
 $ gender: Factor w/ 2 levels "F","M": 1 1 1 1 1
1 1 1 1 1 ...
 $ height: num  69 63 34 61 57 54 48 62 64 45 ...
```

If we remove the factor from the dataframe, the gender variable/column will be treated as a character array, which will not help us in developing the plot.

```
#simulate a data set and store in a data frame
df = data.frame(
 gender = (rep(c("F", "M"), each = 200)),
 height = round(c(rnorm(200, mean = 60, sd = 10), rnorm(200, mean = 70, sd = 6)))
)
str(df)
```

Output



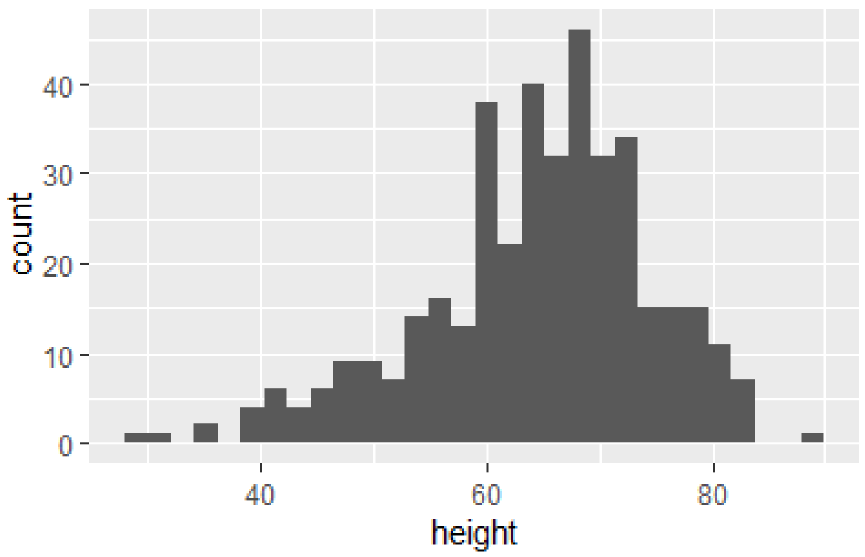
```
'data.frame':   400 obs. of  2 variables:
 $ gender: chr  "F" "F" "F" "F" ...
 $ height: num  67 53 81 64 64 67 67 49 64 71 ...
```

7.2 Basic Histogram

Now that the dataframe is in place, let us plot a basic histogram. According to the definition of a histogram, the plot here will be displaying the rectangles/bins. The height of these rectangles will depend upon the number of people who falls under a particular height. For instance, if there are four people with a height of 51 inches, the rectangle denoting 51 inches will have a height according to the number four. Let us see how it is done:

```
# Plot a basic histogram
ggplot(data = df) +
  geom_histogram(mapping = aes(x = height))
```

Output

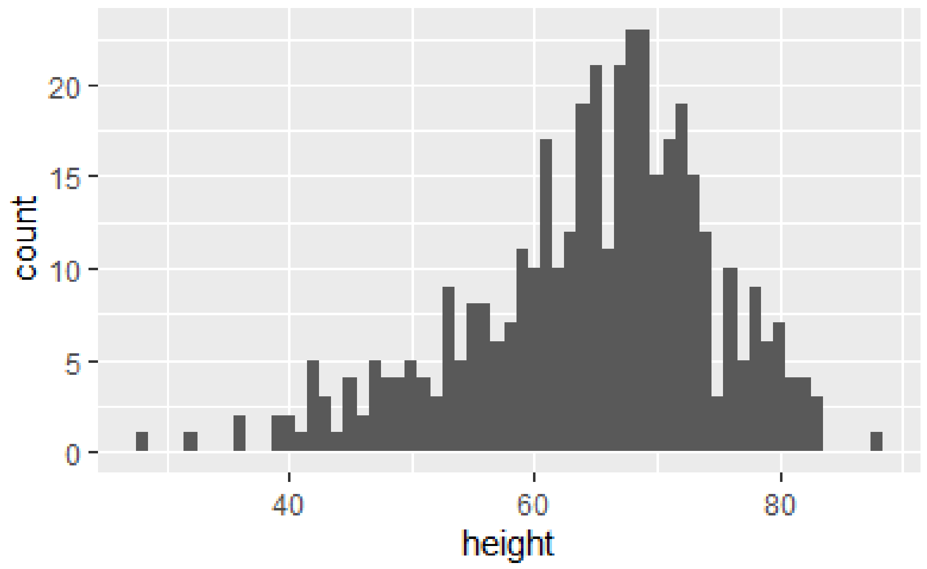


7.3 Changing the Binwidth

The bin/rectangle width is taken by default. If the width of the rectangle is to be changed, the aesthetic function “binwidth” can be used to change the width of the bin.

```
#change the width of bins
ggplot(df, aes(x = height)) +
  geom_histogram(binwidth = 1)
```

Output

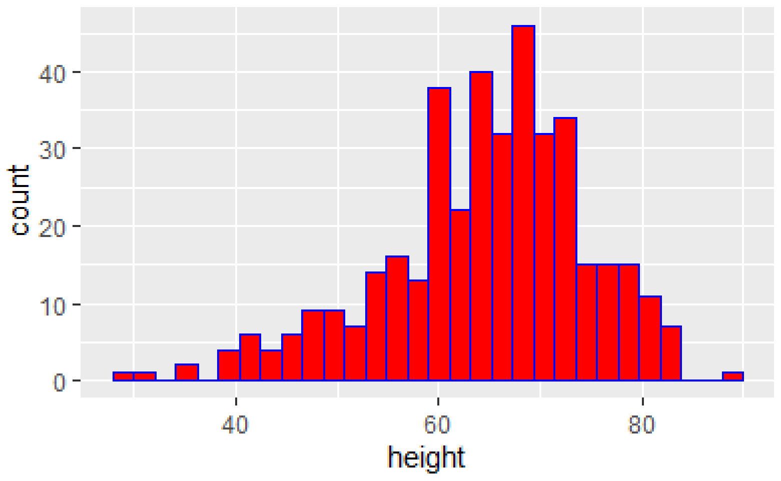


7.4 Aesthetic Attributes of Histogram Plot

Similarly, we can also change its colour by using the colour object. For this, two objects are needed—**colour** and **fill**; The former will change the colour of the boundaries and the latter will fill the bins with the colour specified:

```
#change the color of histogram
ggplot(df,aes(x = height))+
  geom_histogram(color = 'blue', fill = 'red')
```

Output

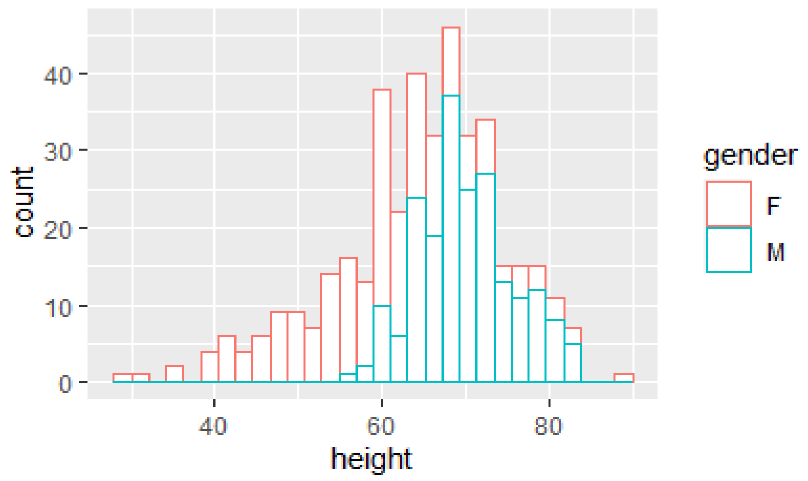


7.4.1 Colour-Based on Another Feature

The histogram above shows us the total number of people within a height. To display the number of males and females under a given height, we can specify using the colour feature of aes object:

```
#change histogram plot line colors by gender
ggplot(df, aes(x = height, color = gender)) +
 geom_histogram(fill = 'white')
```

Output

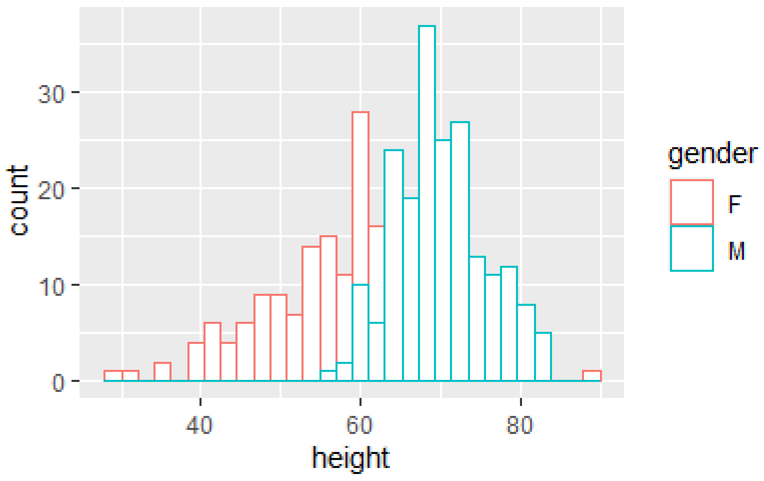


7.4.2 Overlaying Two Histograms

To see the spread of the graph of both male and female heights, we use the **alpha** and **position** features of the aesthetic object. The alpha object is responsible for the transparency of histogram

```
#overlay histograms for both genders
ggplot(df, aes(x = height, color = gender)) +
 geom_histogram(fill = 'white', alpha = 0.3, position = 'identity')
```

Output


count
height
gender
F
M