E-Content on Data Manipulation and Visualization through Statistical Software R (Open Source Software)

Dr Gurpreet Singh Tuteja and Dr Dhiraj Kumar Singh
Department of Mathematics
Zakir Husain Delhi College
(University of Delhi)
Jawaharlal Nehru Marg, Delhi - 110002

Day 4: July 30, 2020

1 Session 4

1.1 ggplot2 Package

Let begin this session of ggplot2 also known as grammar of visualization with the installation. The ggplot2 can be installed in two ways either directly by installing ggplot2 or installing tidyverse as we did in dplyr, *install.packages("ggplot2") install.packages("tidyverse").* After installation, we load the package, so that we can use all built-in funtions or commands of it *library(ggplot2)*

The ggplot2 has three key components:

1. data
2. aesthetic mapping and visual properties :aes
3. render each observation visually : geom

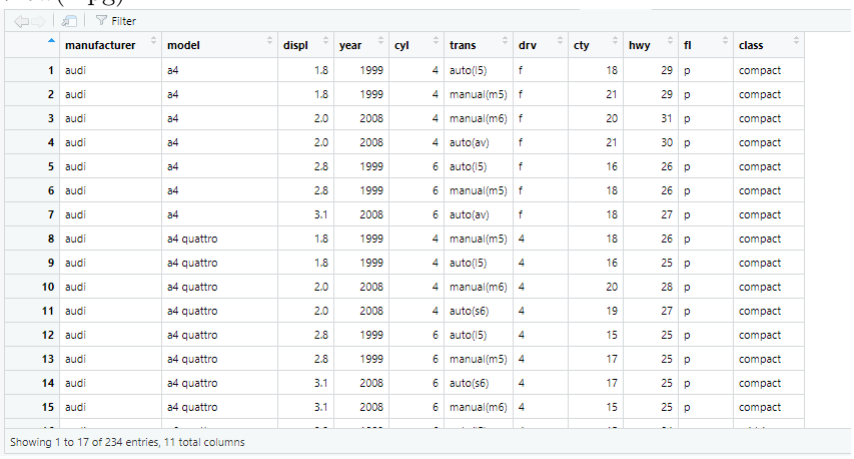
Consider a built in database known as 'mpg': *summary(mpg)*

```
> summary(mpg)
 manufacturer          model               displ           year           cyl           trans
 Length:234         Length:234         Min.   :1.600   Min.   :1999   Min.   :4.000   Length:234
 Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999   1st Qu.:4.000   Class :character
 Mode  :character   Mode  :character   Median :3.300   Median :2004   Median :6.000   Mode  :character
                                       Mean   :3.472   Mean   :2004   Mean   :5.889
                                       3rd Qu.:4.600   3rd Qu.:2008   3rd Qu.:8.000
                                       Max.   :7.000   Max.   :2008   Max.   :8.000
     drv                 cty              hwy             fl               class
 Length:234         Min.   : 9.00   Min.   :12.00   Length:234         Length:234
 Class :character   1st Qu.:14.00   1st Qu.:18.00   Class :character   Class :character
 Mode  :character   Median :17.00   Median :24.00   Mode  :character   Mode  :character
                    Mean   :16.86   Mean   :23.44
                    3rd Qu.:19.00   3rd Qu.:27.00
                    Max.   :35.00   Max.   :44.00
```

str(mpg)

```
> str(mpg)
tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
 $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
 $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
 $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
 $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
 $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
 $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
 $ drv         : chr [1:234] "f" "f" "f" "f" ...
 $ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
 $ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
 $ fl          : chr [1:234] "p" "p" "p" "p" ...
 $ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
>
```

View(mpg)



Filter

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
1	audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
2	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
3	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
4	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact
5	audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact
6	audi	a4	2.8	1999	6	manual(m5)	f	18	26	p	compact
7	audi	a4	3.1	2008	6	auto(av)	f	18	27	p	compact
8	audi	a4 quattro	1.8	1999	4	manual(m5)	4	18	26	p	compact
9	audi	a4 quattro	1.8	1999	4	auto(l5)	4	16	25	p	compact
10	audi	a4 quattro	2.0	2008	4	manual(m6)	4	20	28	p	compact
11	audi	a4 quattro	2.0	2008	4	auto(s6)	4	19	27	p	compact
12	audi	a4 quattro	2.8	1999	6	auto(l5)	4	15	25	p	compact
13	audi	a4 quattro	2.8	1999	6	manual(m5)	4	17	25	p	compact
14	audi	a4 quattro	3.1	2008	6	auto(s6)	4	17	25	p	compact
15	audi	a4 quattro	3.1	2008	6	manual(m6)	4	15	25	p	compact

Showing 1 to 17 of 234 entries, 11 total columns

we can observe the unique values of variables or fields after attaching names of the fields and using :

unique(manufacturer)

[1] "audi" "chevrolet" "dodge" "ford" "honda" "hyundai" "jeep" "land rover"
[9] "lincoln" "mercury" "nissan" "pontiac" "subaru" "toyota" "volkswagen"

unique(drv)

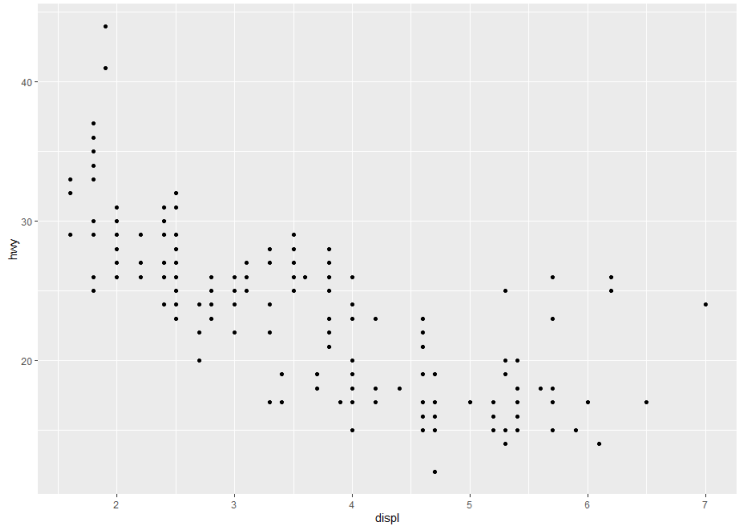
[1] "f" "4" "r"

unique(cyl)

[1] 4 6 8 5

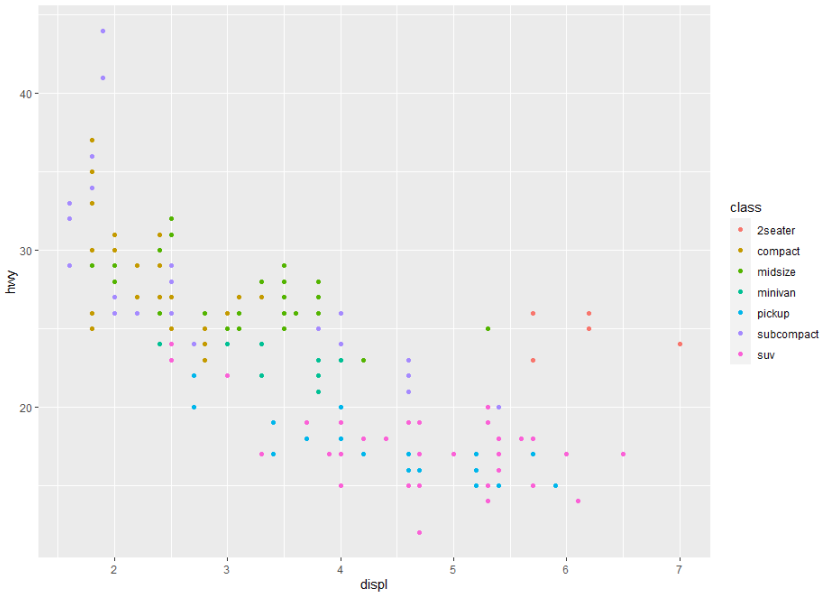
1.2 geom_point

Now, we begin with our first plot:
ggplot(mpg,aes(x=displ,y=hwy))+
geom_point()
It has three componenets viz. mpg as data, aesthetics includes displ on x-axis while hwy on y-axis. The graph plot is geometric point.

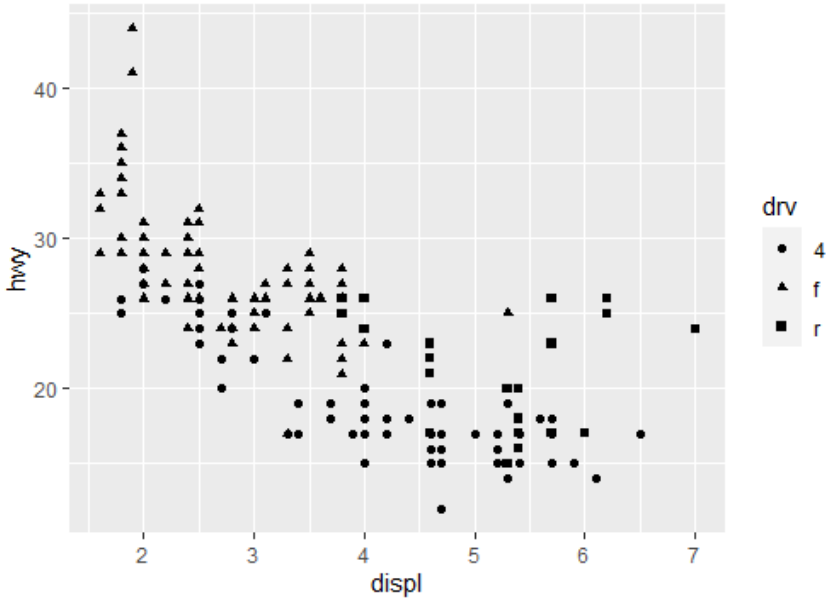


Now, we increase the dimension of the plot by adding color, shape or size to the points representing the class of the vehicle. It may be noted that class of the vehicle is character(discrete) data type.

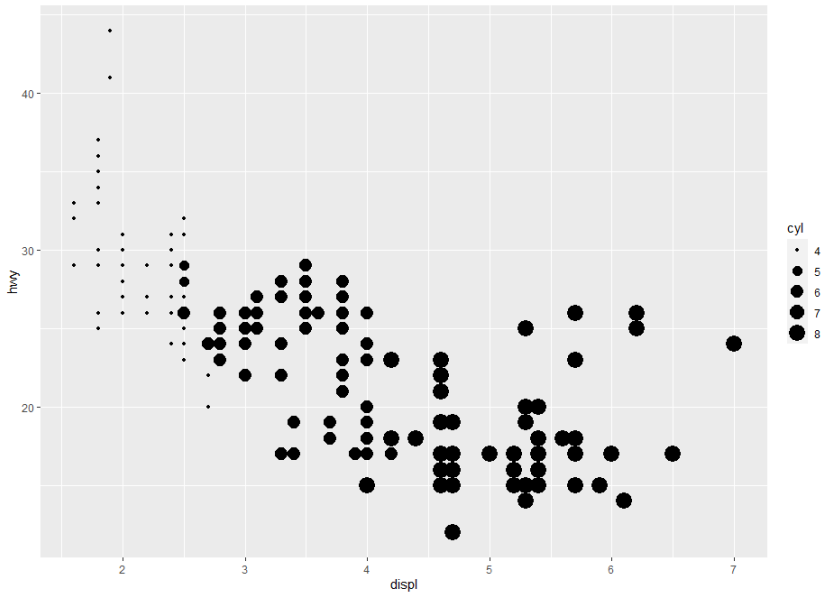
ggplot(mpg,aes(displ,hwy,color=class))+
geom_point()



ggplot(mpg,aes(displ,hwy,shape=drv))+
geom_point()

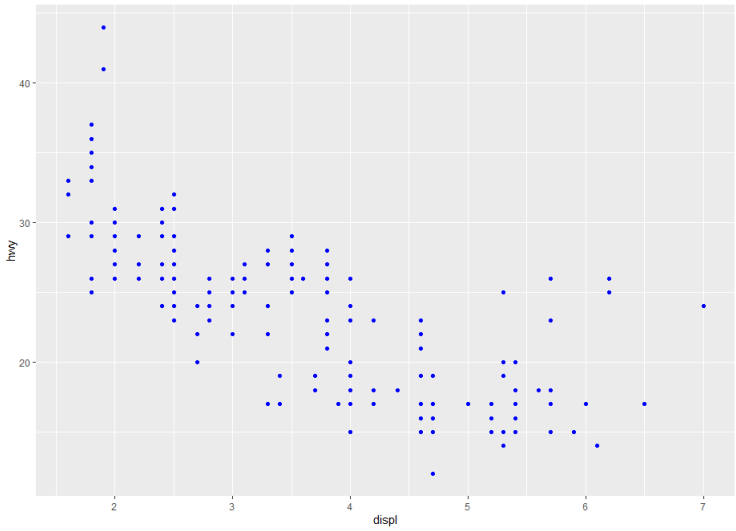


ggplot(mpg,aes(displ,hwy,size=cyl))+
geom_point()



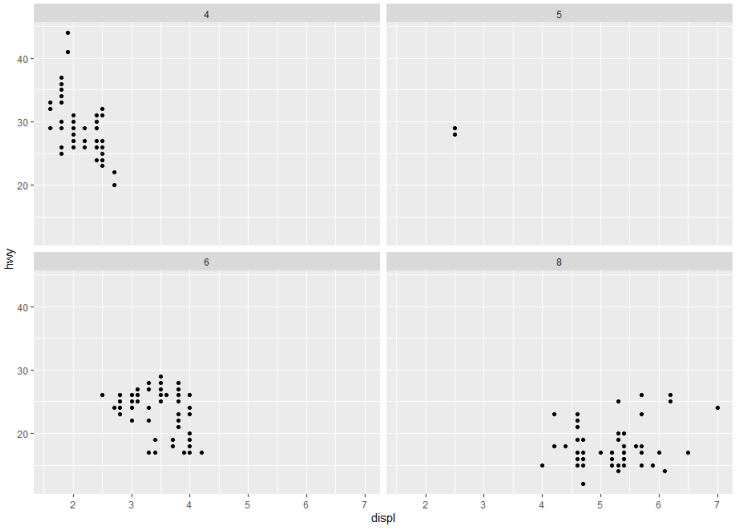
Another, representation of the aes is:
ggplot(mpg,aes(displ,hwy))+ geom_point()
It is not necessary to mention the x-axis and y-axis explicitly.

The color of the geom points can be changed to blue :
ggplot(mpg,aes(displ,hwy))+ geom_point(color="blue")



The following command is incorrect and should be avoided.
ggplot(mpg,aes(displ,hwy))+ geom_point(aes(color="blue"))

You can create multi-panel plots using a single function. You can split a single plot into many related plots using facet_wrap ().In the following facet wrap, displ vs hwy is plotted for number of cylinders
ggplot(mpg,aes(displ,hwy))+
geom_point()+ facet_wrap(cyl)

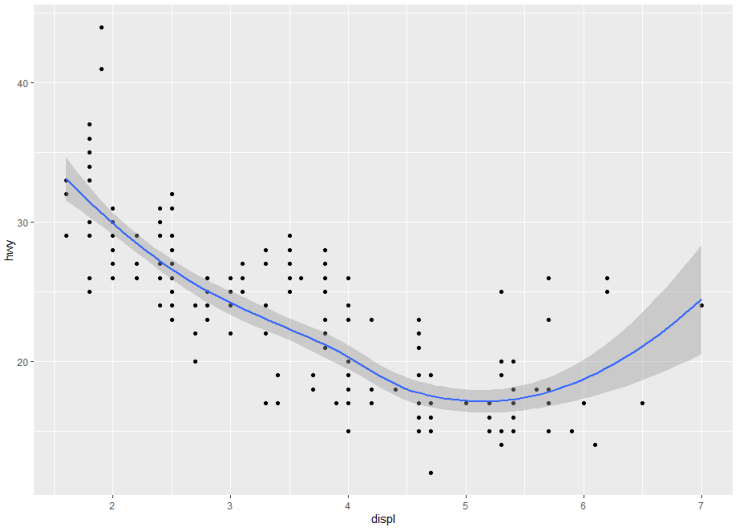


1.3 R function geom_smooth()

Key R function: geom_smooth() for adding smoothed conditional means / regression line. Key arguments:

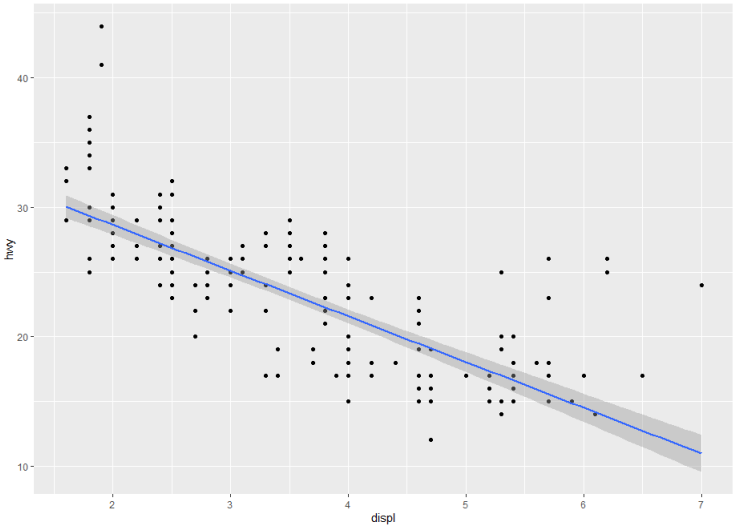
1. color, size and linetype: Change the line color, size and type.
2. fill: Change the fill color of the confidence region.

ggplot(mpg,aes(displ,hwy))+
geom_point()+
geom_smooth()



1.3.1 Regression Line

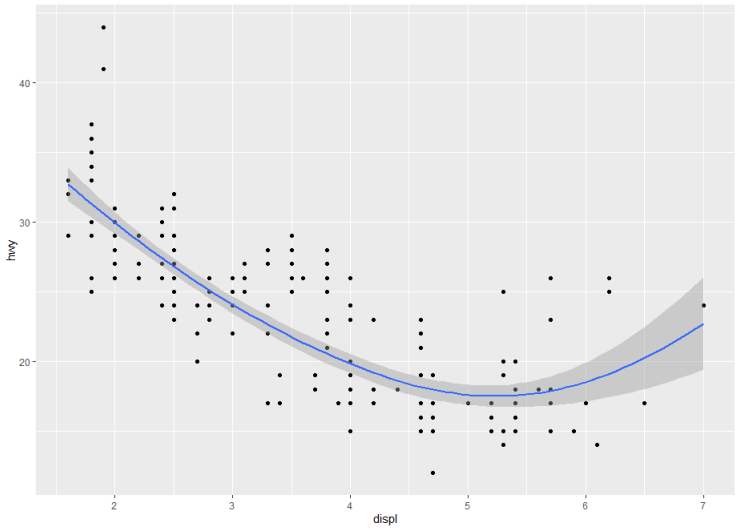
To add a regression line on a scatter plot, the function geom_smooth() is used in combination with the argument method = lm. lm stands for linear model.
ggplot(mpg,aes(displ,hwy))+
geom_point()+
geom_smooth(method=lm)



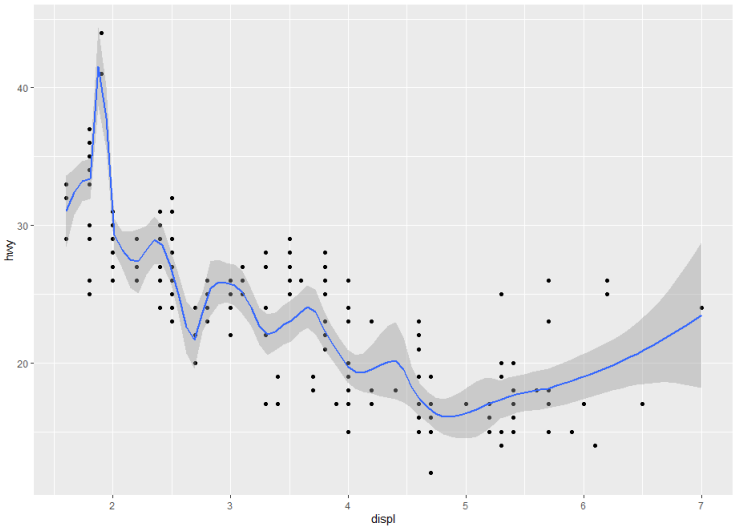
You can try method=loess also.

1.3.2 Span in geom_smooth

Span controls the amount of smoothing for the default loess smoother. Smaller numbers produce wigglier lines, larger numbers produce smoother lines.
ggplot(mpg,aes(displ,hwy))+
geom_point()+
geom_smooth(span=1)

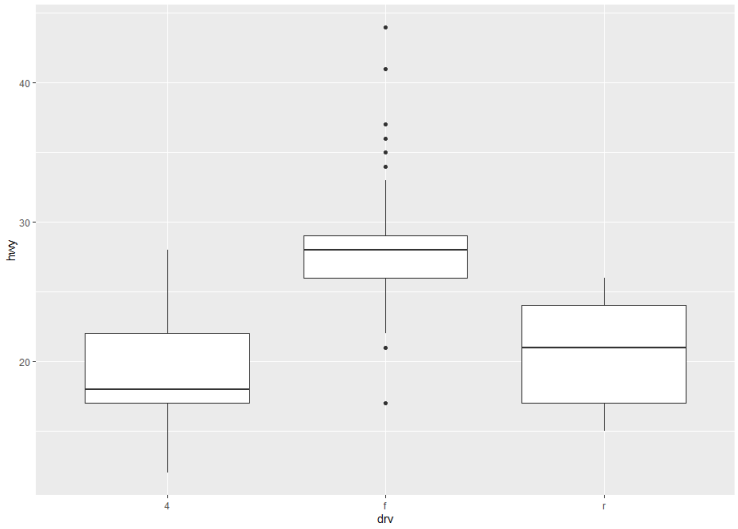


ggplot(mpg,aes(displ,hwy))+
geom_point()+
geom_smooth(span=0.2)



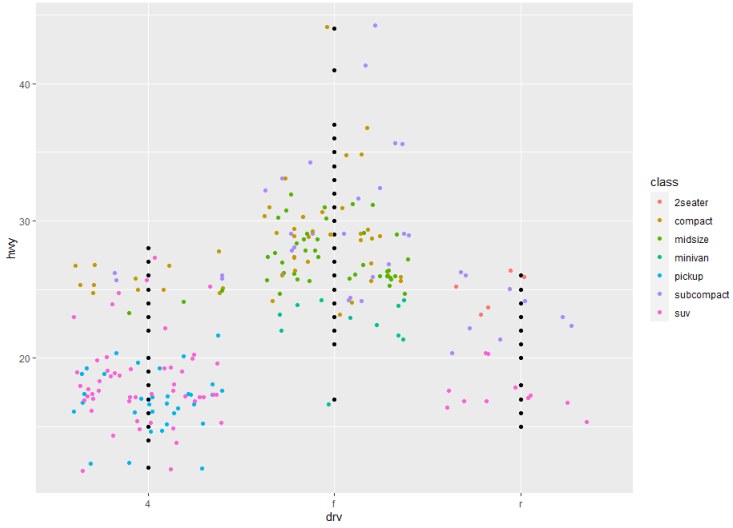
1.4 Boxplot

The boxplot compactly displays the distribution of a continuous variable. It visualises five summary statistics (the median, two hinges and two whiskers), and all "outlying" points individually.A boxplot is a method for graphically depicting groups of numerical data through their quartiles. It may also have lines extending from the boxes (whiskers) indicating variability outside the upper and lower quartiles, hence the terms box-and-whisker plot and box-and-whisker diagram.
ggplot(mpg,aes(drv,hwy))+
geom_boxplot()



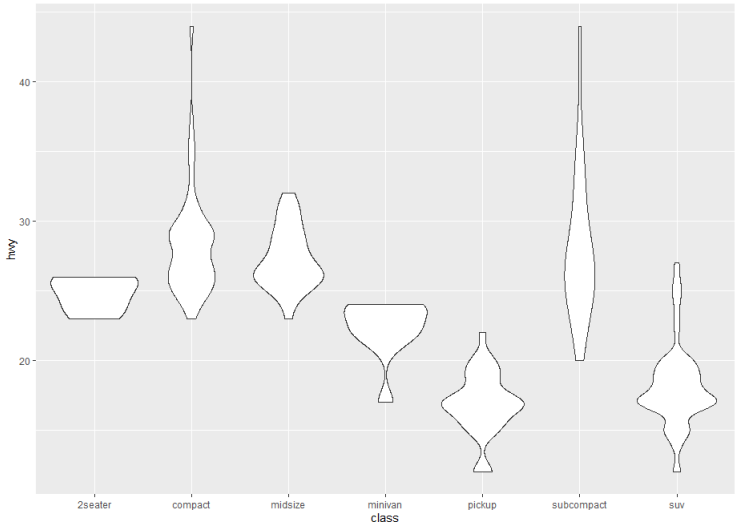
1.5 Jitter

Jitter plots include special effects with which scattered plots can be depicted. Jitter is nothing but a random value that is assigned to dots to separate them:
ggplot(mpg,aes(drv,hwy))+
geom_jitter()



1.6 Violin Plot

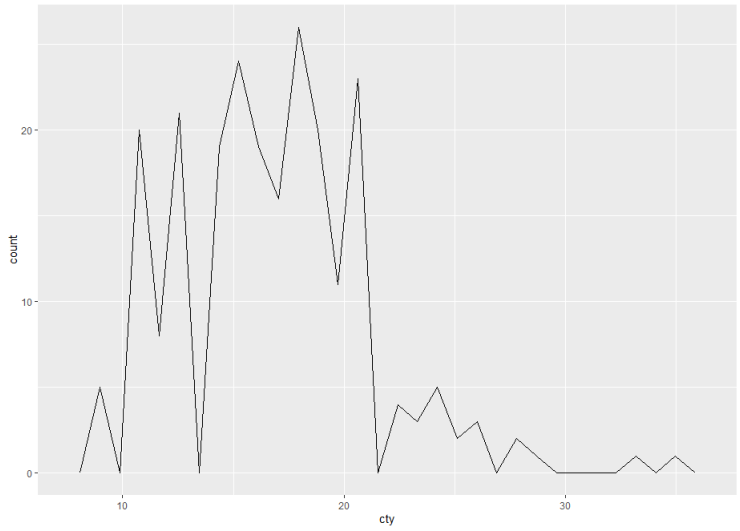
Violin plots are similar to box plots, except that they also show the probability density of the data at different values. These plots include a marker for the median of the data and a box indicating the interquartile range, as in the standard box plots. Overlaid on this box plot is a kernel density estimation. Like box plots, violin plots are used to represent comparison of a variable distribution (or sample distribution) across different "categories".
A violin plot is more informative than a plain box plot. In fact while a box plot only shows summary statistics such as mean/median and interquartile ranges, the violin plot shows the full distribution of the data.
ggplot(mpg,aes(class,hwy))+
geom_violin()



1.7 Histograms and Frequency Polygon

Frequency polygons are the graphs of the values to understand the shape of the distribution of the values. They are useful in comparing different data sets and visualising cumulative frequency distribution of the data sets. In base R, we can use polygon function to create the frequency polygon but first we should create a line plot for the two variables under consideration.
ggplot(mpg,aes(cty))+

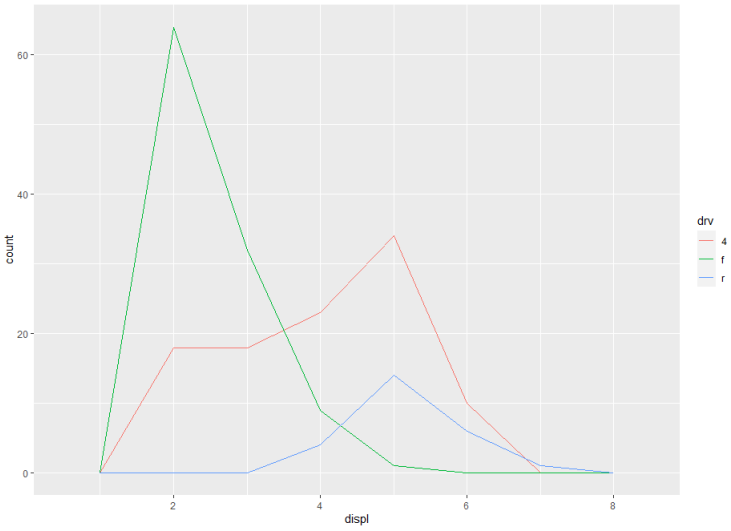
geom_freqpoly()



1.7.1 Binwidth

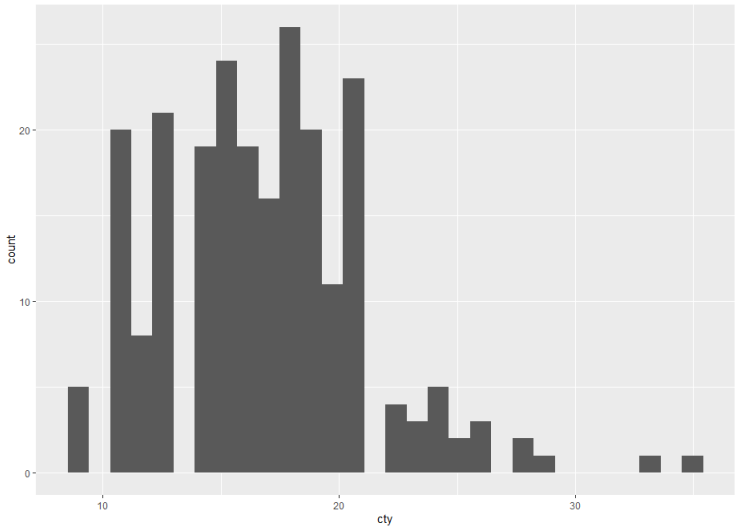
The binwidth determines how smooth your distribution will appear: the smaller the binwidth, the more jagged your distribution becomes. It's good practice to consider several binwidths in order to detect different types of structure in your data.

```
ggplot(mpg,aes(displ,color=drv))+
geom_freqpoly(binwidth=1)
```

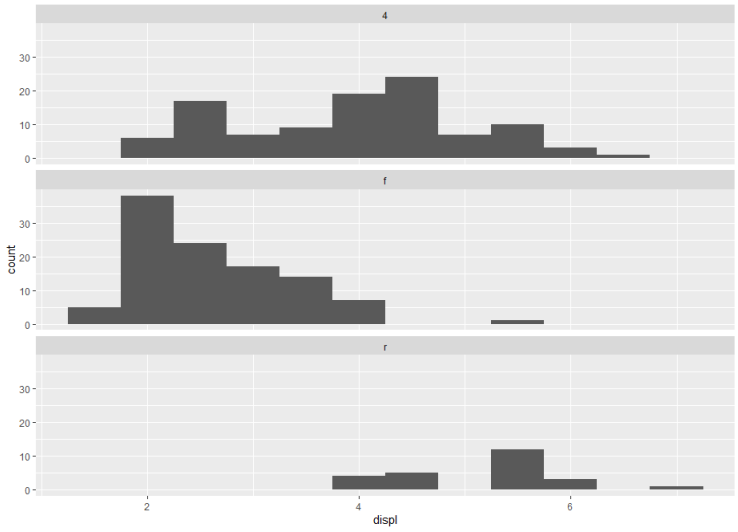


1.8 Histogram

A histogram represents the frequencies of values of a variable bucketed into ranges. Histogram is similar to bar chat but the difference is it groups the values into continuous ranges. Each bar in histogram represents the height of the number of values present in that range.
ggplot(mpg,aes(cty))+
geom_histogram()

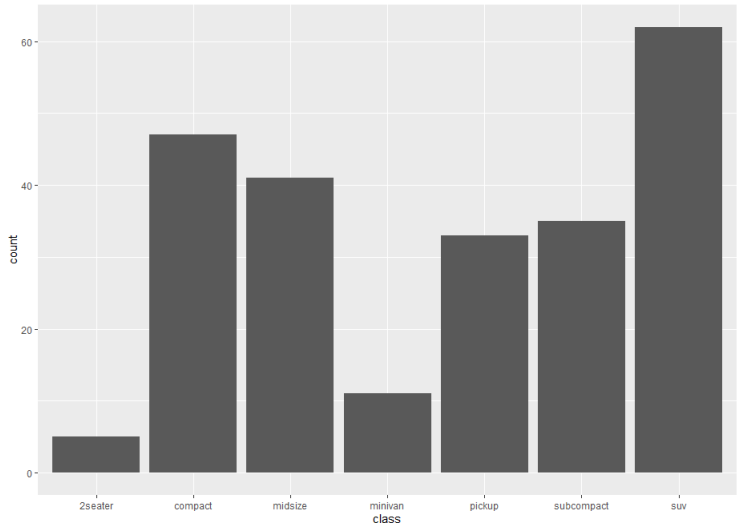



```
ggplot(mpg,aes(displ,fill=hwy))+
geom_histogram(binwidth=.5)+
facet_wrap( drv,ncol=1)
```

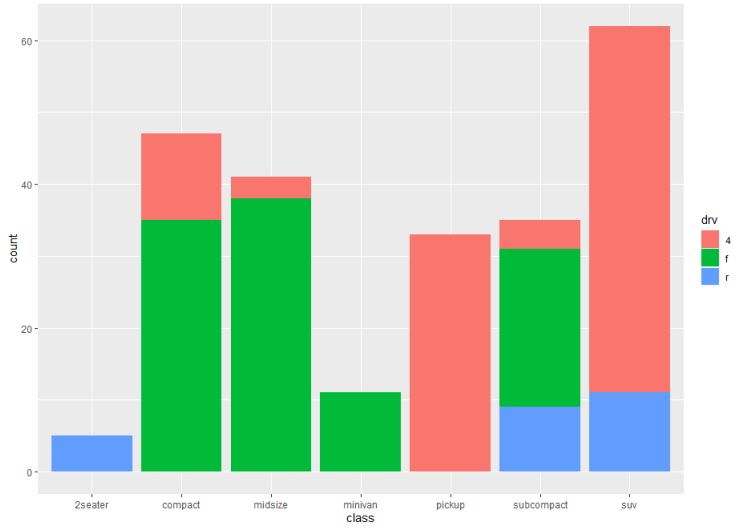


1.9 Bar Chart

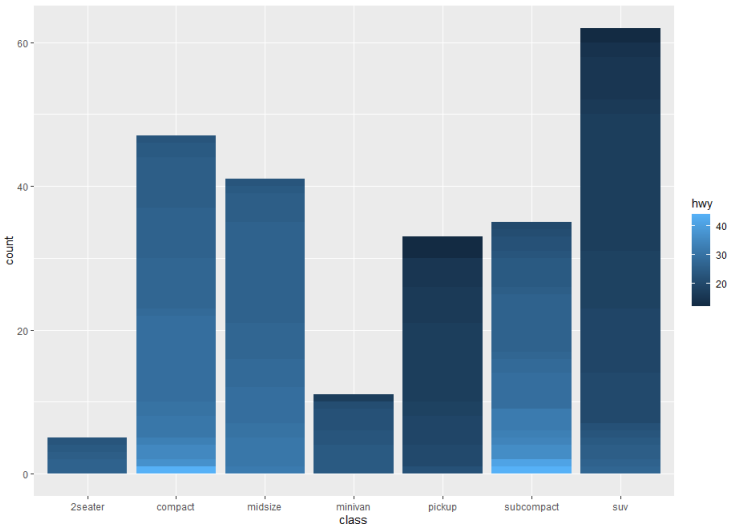
A bar chart represents data in rectangular bars with length of the bar proportional to the value of the variable. R uses the function barplot() to create bar charts. R can draw both vertical and Horizontal bars in the bar chart. In bar chart each of the bars can be given different colors.
ggplot(mpg,aes(class))+
geom_bar()



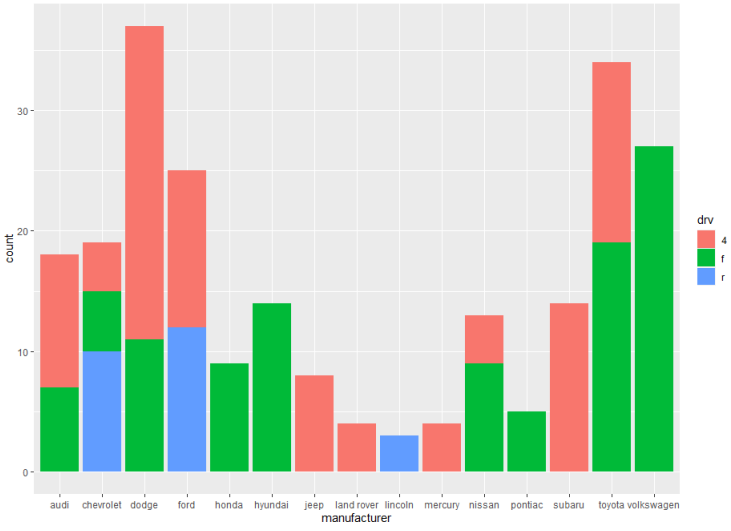
ggplot(mpg,aes(class,fill=drv))+
geom_bar()



ggplot(mpg,aes(class,fill=hwy,group=hwy))+
geom_bar()

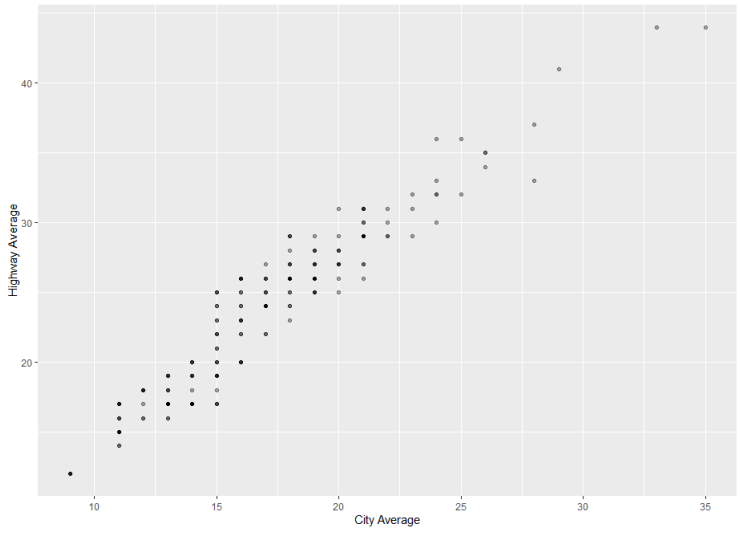


ggplot(mpg,aes(manufacturer,fill=drv))+
geom_bar()



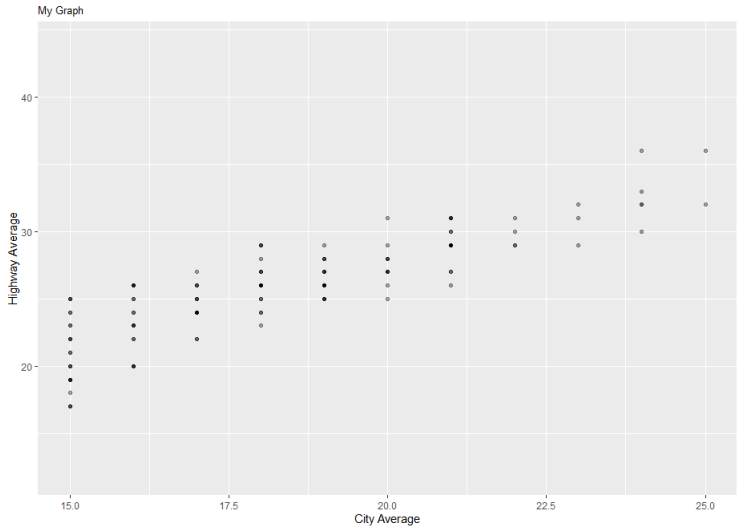
1.10 Labelling Axes

```
ggplot(mpg,aes(cty,hwy))+
geom_point(alpha=1/3)+
xlab("City Average")+
ylab("Highway Average")
```



1.11 Theme

```
ggplot(mpg,aes(cty,hwy))+
geom_point(alpha=1/3)+
xlab("City Average")+
ylab("Highway Average")+
theme(plot.title = element_text(size=10))+
ggtitle("My Graph")+
xlim(15,25)
```



Here ends session 4 and in next session we will be using the concepts of dplyr and ggplot2 to wrangle the live data of COVID19. You are requested to appear for quiz.