Statuatory Warning

Will Speak too fast and mumble words

Please ask Siva to repeat if you do not understand

R

- R is an open-source compute programming language and runs on Linux, Windows, and Mac-OS.
- R is FREE.
- The R project web page
  http://www.r-project.org

R

- R is modeled after S and S-Plus. The S language was developed in the late 1980s at AT & T labs.

- The R project was started by Robert Gentleman and Ross Ihaka of the Statistics Department of the University of Auckland in 1995. [*Journal of Computational and Graphical Statistics,5:3, pp. 299-314. 1996.*]

- R is now a collaborative project with many contributors and is maintained by the R core-development team.

Installing R

- To download `R` visit https://cloud.r-project.org

- Rstudio is an Integrated Development Environment, or IDE for R. To download `Rstudio` visit http://www.rstudio.com/download

Getting Started on R– as a calculator

You can try the following commands:

```
> 9 / 44
> 0.6 * 0.4 + 0.3 * 0.6
> log(0.6 * 0.4 + 0.3 * 0.6)
```

Getting Started on R– as a calculator

Your output should look like:

```
> 9 / 44
[1] 0.2045455
> 0.6 * 0.4 + 0.3 * 0.6
[1] 0.42
> log(0.6 * 0.4 + 0.3 * 0.6)
[1] -0.8675006
```

`[1]` at the beginning of each answer is there for a good reason.

Any data is stored in R as a *vector*. `[1]` represents the position of that element in the vector.

R- c function

Suppose we wish to enter Scores in Handicraft class of 10 students.

40, 39, 15, 6, 18, 22, 30, 21, 15, 23

```
> Scores= c(40, 39, 15, 6, 18, 22, 30, 21, 15, 23)
```

R- c function

The output should be

```
> Scores = c(40, 39, 15, 6, 18, 22, 30, 21, 15, 23)
```

In the above, we have assigned values to a variable called `Scores`.

The assignment operator is `=`.

The values do not get displayed automatically unless we call it with `Scores` as below.

```
> Scores
 [1] 40 39 15  6 18 22 30 21 15 23
```

R-inbuilt functions

R has many in-built functions

```
> meancomputation = (40+ 39+ 15+ 6+ 18+ 22+ 30+ 21+ 15+ 23)/10
> meancomputation
[1] 22.9
> meaninbuilt = mean(Scores)
> meaninbuilt
[1] 22.9
```

R-Slicing

Changing one element of Scores: Suppose we want to change the entry of student 4 from 6 to 16.

```
> Scores
 [1] 40 39 15  6 18 22 30 21 15 23
> Scores2 = Scores # create a copy of Scores
> Scores2[4] = 16
> Scores2
 [1] 40 39 15 16 18 22 30 21 15 23
> Scores[c(1,3,5)]
[1] 40 15 18
```

R- Logical Operators ==, <=, >=, <, >

Selecting few elements of Scores: Suppose we want to see which students Scores are equal to 30 marks

```
> Scores
 [1] 40 39 15  6 18 22 30 21 15 23
> y = which(Scores == 30)
> y
[1] 7
```

or those with Scores lesser than or equal to 20.

```
> z = which(Scores <= 20)
> z
[1] 3 4 5 9
```

R- Creating sequence of vectors

```
> x = 1:100
> x
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
 [91]  91  92  93  94  95  96  97  98  99 100
> x[x < 10 | x > 90]
 [1]   1   2   3   4   5   6   7   8   9  91  92  93  94  95  96  97  98  99 100
```

Data

- Data and its analysis has a rich and wide literature.
- Three kinds of Data:
  - Categorical Data
  - Discrete Numeric Data
  - Continuous Numeric Data
- *On the Theory of Scales of Measurement* By S. S. Stevens Science 07 Jun 1946: Vol. 103, Issue 2684, pp. 677-680, gave a broad classification of data from measurements into 9 categories.

Discrete Numerical Data

- Many data are described in terms of numbers.
- Many variables naturally take on only discrete values.
- Boxplot and Histograms are used to visualise such data.

Discrete Numerical Data: Key features

- Center
- Spread
- Shape

Discrete Numerical Data: Key features

- Center Widely used measure of centre is the `mean` or the average of the data set. Other measures include the `median` and the `mode`
- Spread Understanding variabiity of the given data is very important. If one were to understand `mean` as specifying the center then the range of the data set around it is determined by its variability or spread. It is often measured by the variance(`var`) or standard deviation (`sd`) or the inter-quartile range (`IQR`).
- Shape To understand various distributional aspects of the dataset one needs to understand its "shape". For e.g. if it is symmetric or skewed around its mean. Other aspects include among the data points which are more likely than others.

Datasets in R

R has a lot inbuilt Datasets that one can use. The command :

```
> data()
```

will list currently installed data sets.

Datasets in R

R has a lot inbuilt Datasets that one can use. The command :

```
> data()
```

will list currently installed data sets.

Datasets in R

- R stores many datasets as data frame (often).
- A data frame is a rectangular collection of variables (in the columns) and observations (in the rows).

airquality in R

Let us learn about real data stored as data frame.

```
> ?airquality
```

`airquality` in R

Let us learn about `airquality` dataset a bit more.

- we could print the entire data set on the screen

  `>airquality`

  but this is too much information.

- Let us try the `head()` function

  `> head(airquality)`

  This provides the first six rows.

`airquality` in R

Let us learn about `airquality` dataset a bit more.

- Let us try the `tail()` function

  ```
  > tail(airquality)
  ```

  This provides the last six rows.

airquality in R

- Below provides the first ten rows.

```
> head(airquality, n = 10)
```

- Data can be called using row and column number

```
> airquality[148,4]
[1] 63
```

- We can use the variable name for the given column and call it by its position.

```
> airquality$Temp[148]
[1] 63
```

`airquality` in R

- Provides an entire row

  ```
  > airquality[148,]
  ```

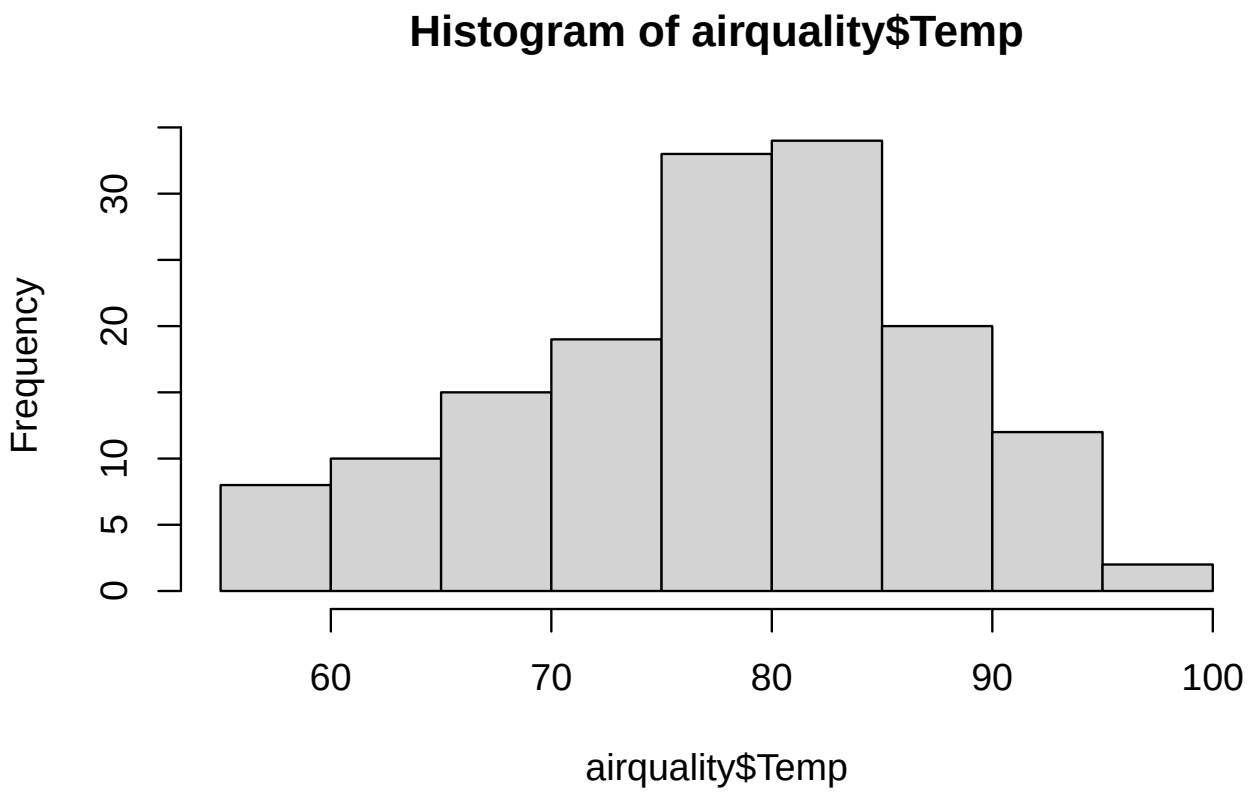
- Provides `Ozone` `Temp` columns

  ```
  > airquality[,c(1,4)]
  ```

  using `c()` function we can form any vector and that will enable display of the respective columns. We did not specify the row, so all rows will be displayed.

Five Number Summary and Histograms

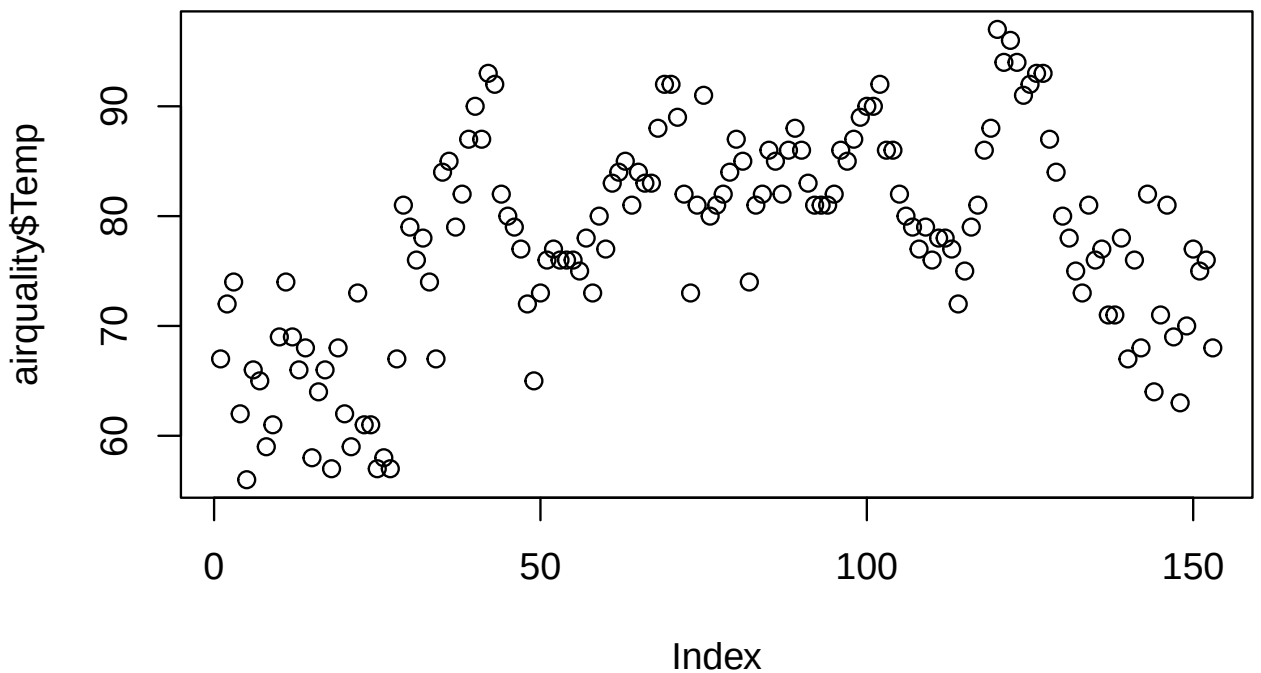
```
> summary(airquality$Temp)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  56.00   72.00   79.00   77.88   85.00   97.00
> hist(airquality$Temp)
```



Plot

We can use the `plot` function to just plot.

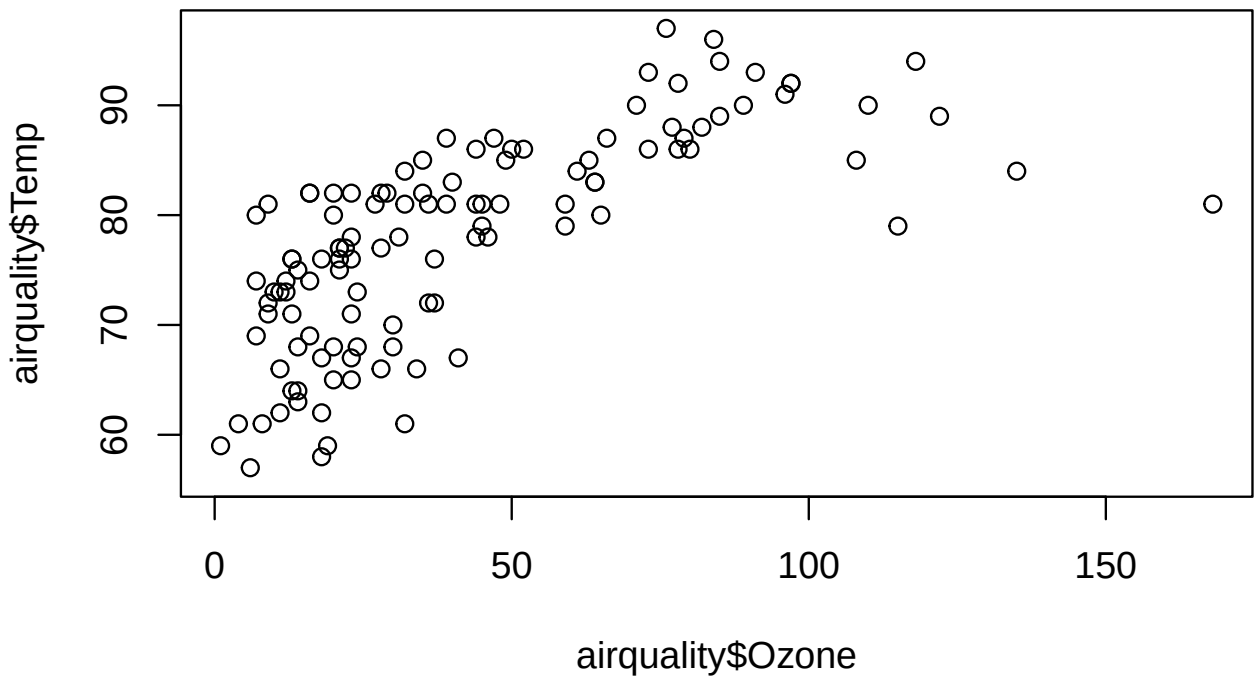
```
> plot(airquality$Temp)
```



Scatter Plot

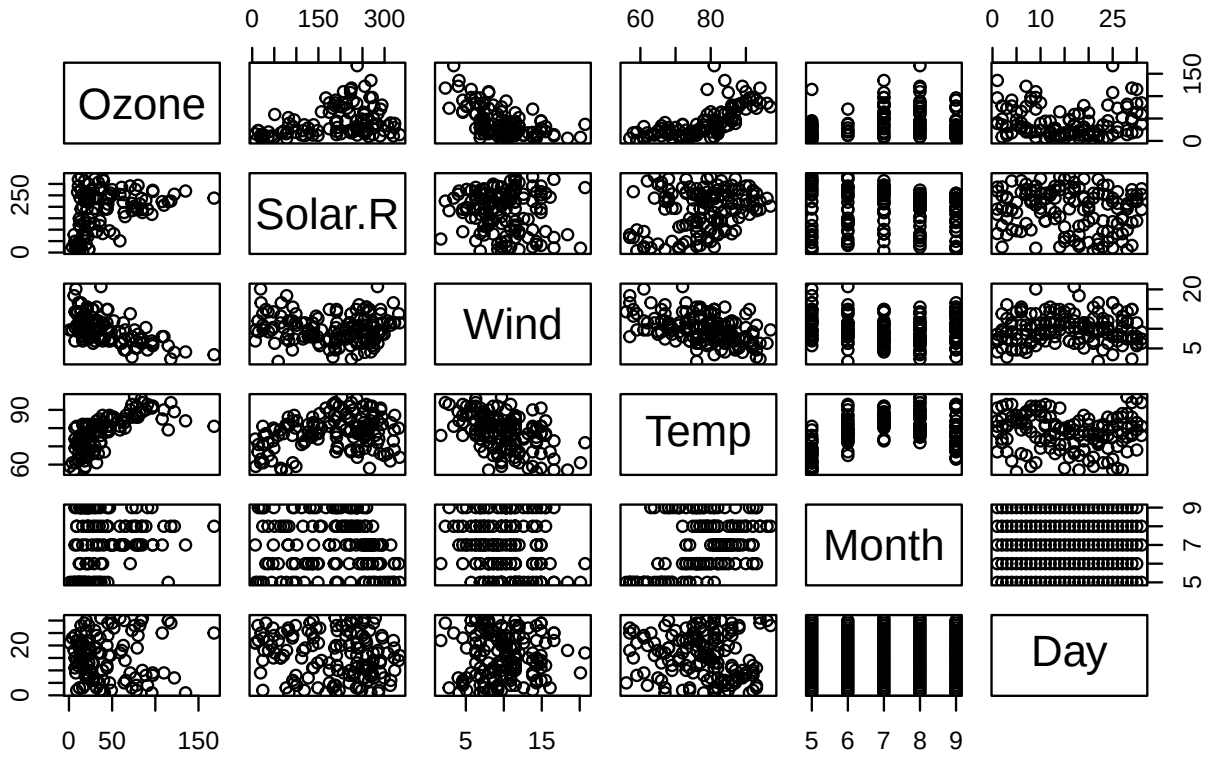
We can use the `plot` function to get a Scatter plot.

```
> plot(airquality$Ozone, airquality$Temp)
```



Plot!

```
> plot(airquality)
```



External Packages in R

R has can be enhanced with a lot of external packages that are available. The package `UsingR` has many datasets loaded in it.

```
> install.packages("UsingR")
```

Once installed then to add to current workspace

```
> library("UsingR")
```

ggplot2- Data Visualisation

ggplot2 implements grammar of graphics

```
> install.packages("tidyverse")
```

Once installed then to add to current workspace

```
> library("tidyverse")
> library("viridis")
```

ggplot2- Data Visualisation

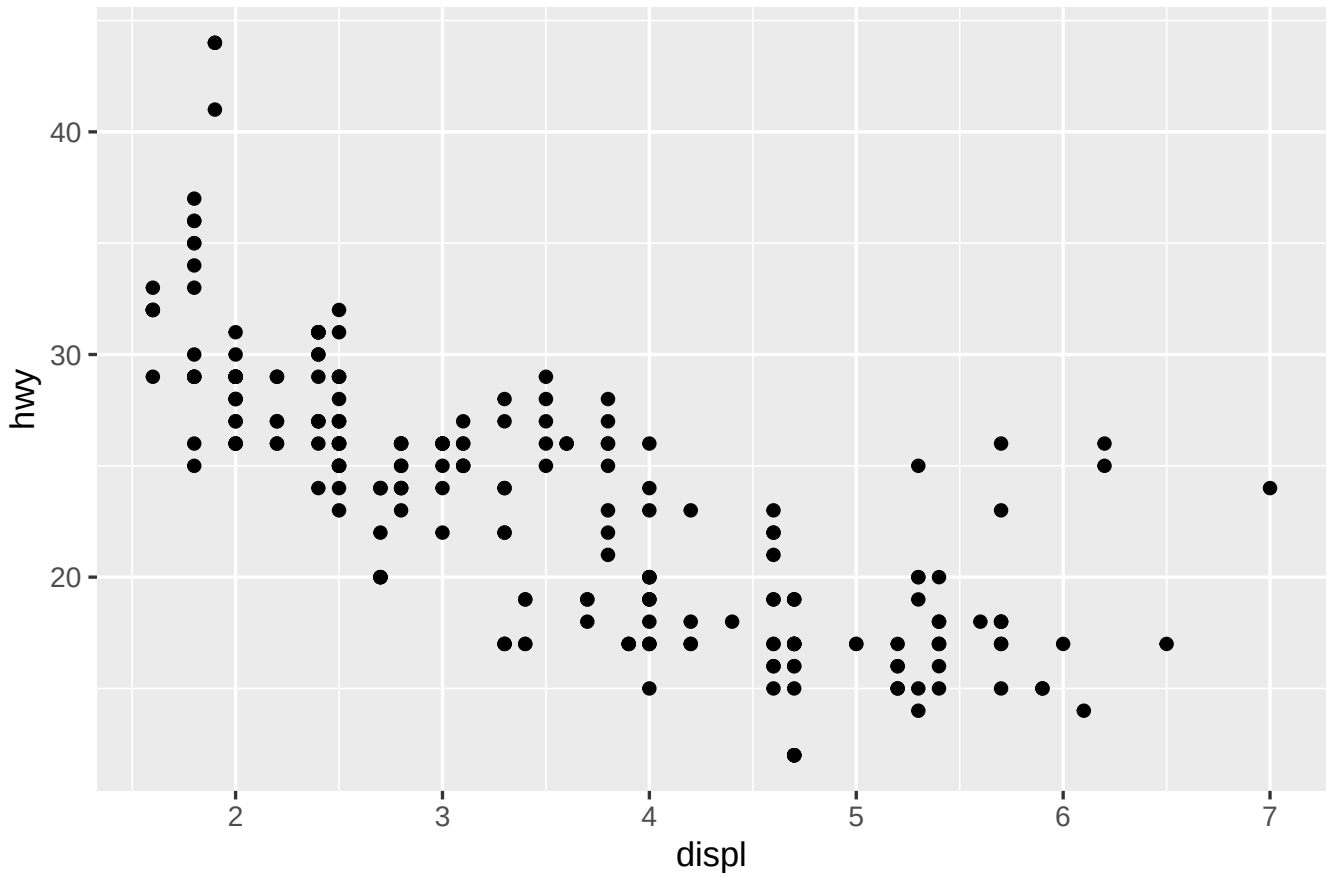
Dataset in `tidyverse`

```
> mpg
```

Observations collected by US Environment Protection Agency on 38 models of cars.

ggplot2- Data Visualisation

```
> ggplot(data=mpg) +
+     geom_point(mapping=aes(x=displ, y=hwy))
```



The plots negative relationship between Engine Size and Fuel Effiency.

ggplot()

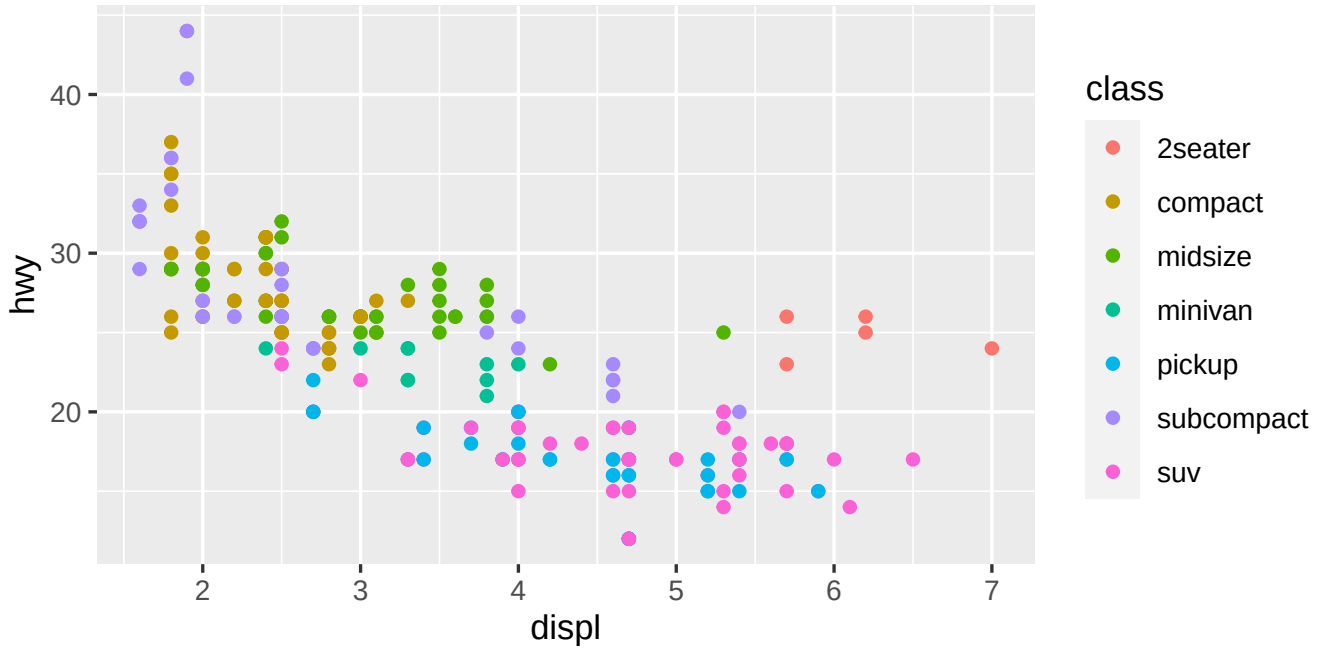
- Begins with a function `ggplot()`- creates a coordinate system that you can add addlayers to. The first arugment is the data set to use `ggplot(data=mpg)` creates an empty graph.
- Add layers to `ggplot()`- the function `geom_point()` adds a layer of points to your plot
- Each geom function takes a `mapping` argument. The `mapping` argument is always paired with `aes()`
- `ggplot(data= <DATA>)+`
  `GEOM-FUNCTION(mapping=aes(<MAPPINGS>))`
- We will learn how to complete and extend this basic template.

ggplot2- Aesthetics Mappings

- `disp` a car's engine size, in litres.
- `hwy`, a car's fuel efficiency on the highway, in miles per gallon (mpg).

ggplot2- Aesthetics Mappings

```
> ggplot(data=mpg) +
+     geom_point(mapping=aes(x=displ, y=hwy, colour=class))
```



Added a third variable called `class` to a 2−D scatter plot by mapping it to an *a*esthetic.

ggplot2- Scaling

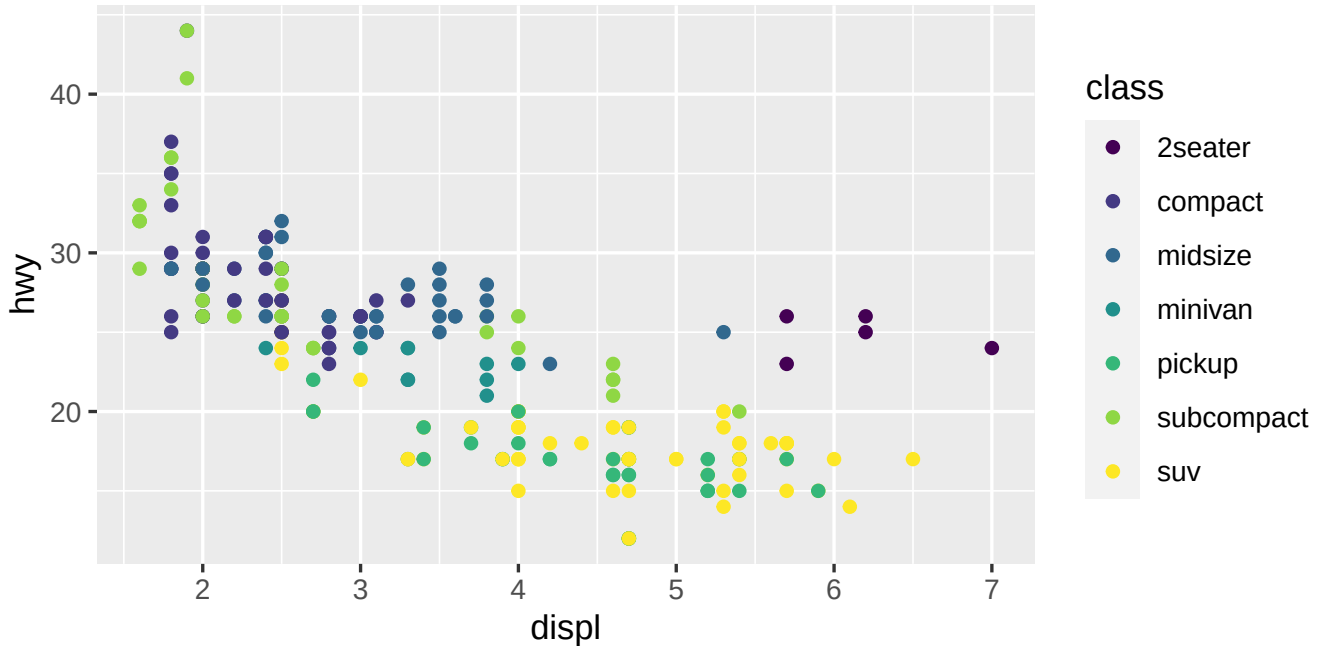
- Map an aesthetic to a variable
- Associate the `name` of the aesthetic to the name of the `variable.`
- Above example Name=colour and Variable=class.
- Scaling: `ggplot2()` will assign a unique level of the aesthetic `colour` to a unique level to the variable `class`.
- `ggplot2` will also add a legend explaining the levels
- Other aesthetics include : shape and size.

ggplot()-viridis options

- The viridis scales provide colour maps that are perceptually uniform in both colour and black-and-white.
- They are also designed to be perceived by viewers with common forms of colour blindness.
- See also https://bids.github.io/colormap/.

ggplot()-viridis options

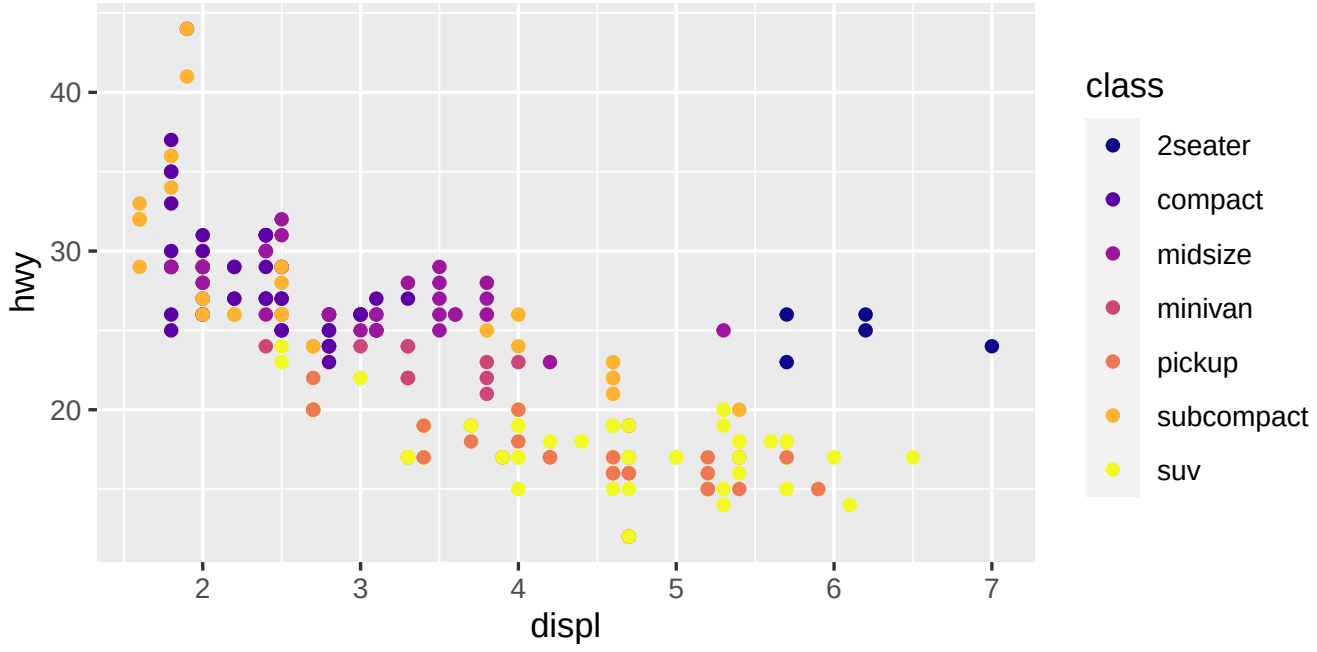
```
> ggplot(data=mpg) +
+     geom_point(mapping=aes(x=displ, y=hwy, colour=class))+
+     scale_colour_viridis_d()
```



Using colour palette from viridis package (colour blind colours).

ggplot()-viridis options

```
> ggplot(data=mpg) +
+     geom_point(mapping=aes(x=displ, y=hwy, colour=class))+
+    scale_colour_viridis_d(option = "plasma")
```



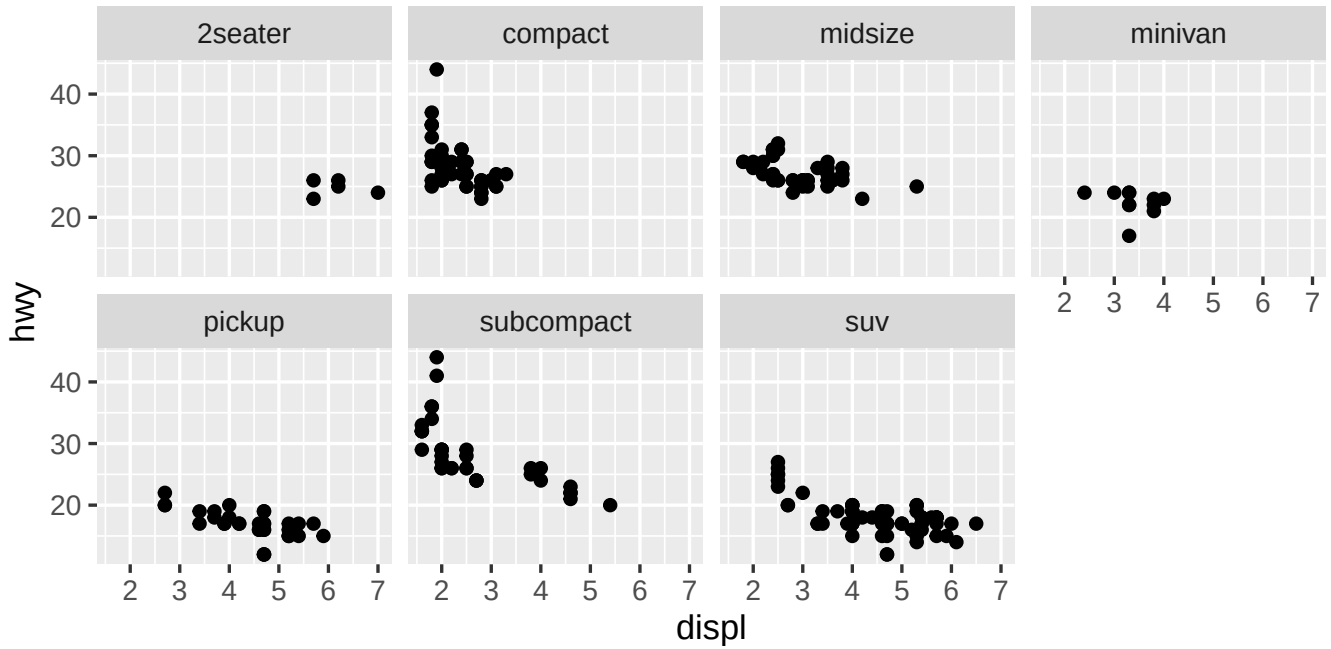
Using colour palette from viridis package (colour blind colours).

ggplot()-facets

- As `aesthetics` was used to add an additional variable to the plot, another way is to add facets (useful for categorical variables).
- `facet_wrap` splits plot by a single variable into subplots that each display one subset of the data.

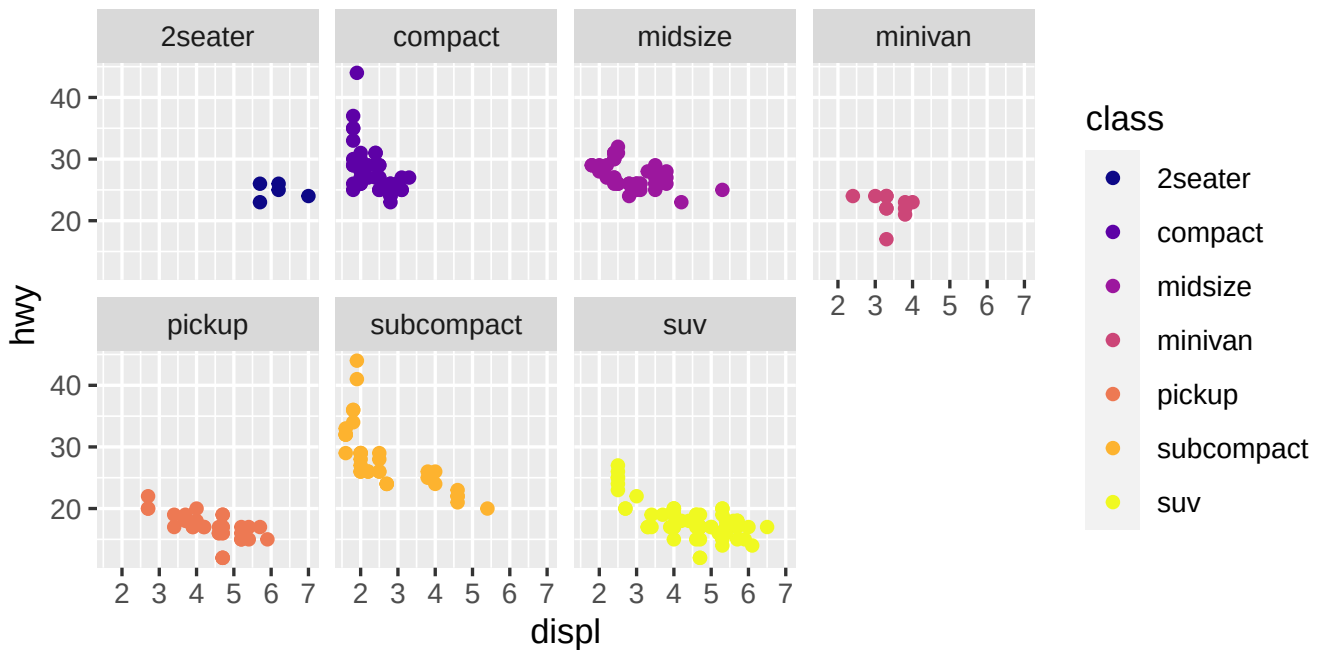
ggplot()-facets

```
> ggplot(data=mpg) +
+     geom_point(mapping=aes(x=displ, y=hwy))+
+                    facet_wrap(~ class, nrow=2)
```



ggplot()-facets and Viridis

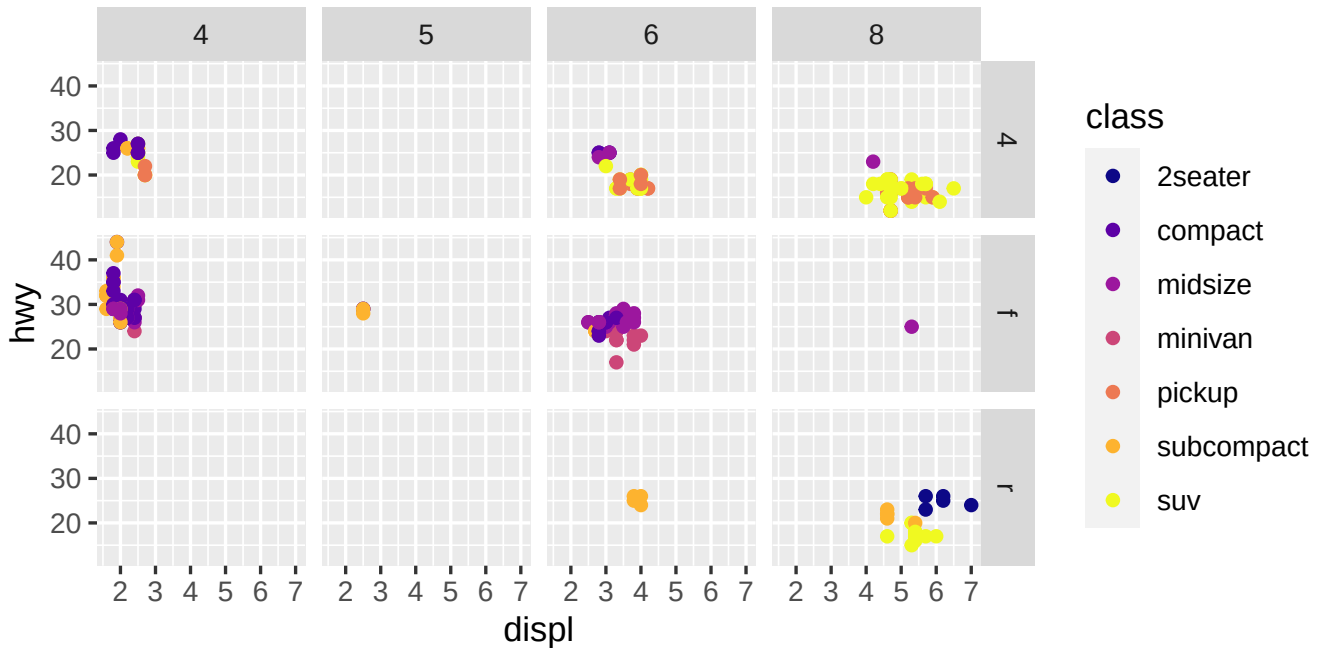
```
> ggplot(data=mpg) +
+   geom_point(mapping=aes(x=displ, y=hwy, colour=class))+
+   scale_colour_viridis_d(option = "plasma")+
+   facet_wrap(~class, nrow=2)
```



`facet_wrap` splits plot by a single variable into subplots that each display one subset of the data.

ggplot()-facets

```
> ggplot(data=mpg) +
+     geom_point(mapping=aes(x=displ, y=hwy, colour=class))+
+    scale_colour_viridis_d(option = "plasma")+
+     facet_grid(drv~cyl)
```

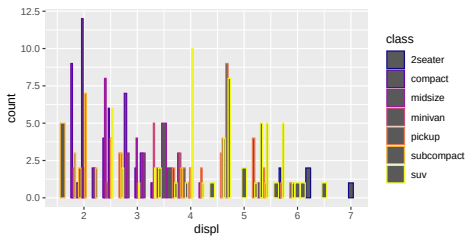


`facet_grid` splits plot by a combination of two variables into subplots that each display one subset of the data.

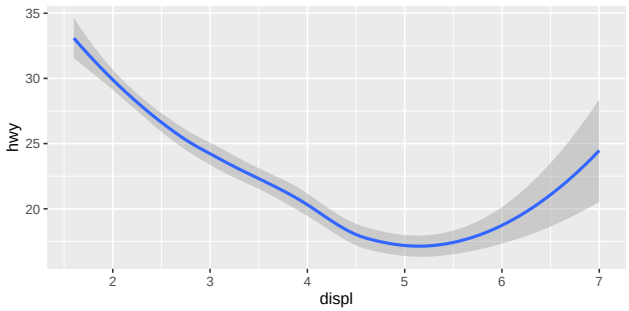
geom-

- geometrical object that a plot uses to represent data.
- can do various plots:

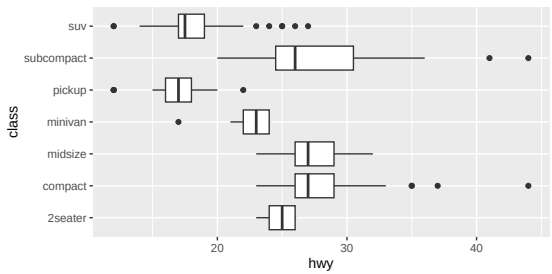
bar charts: `geom_bar`,



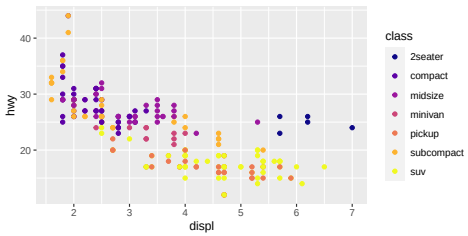
line-charts:`geom_smooth`,



box-plot:`geom_boxplot`,



Scatter:`geom_point`,



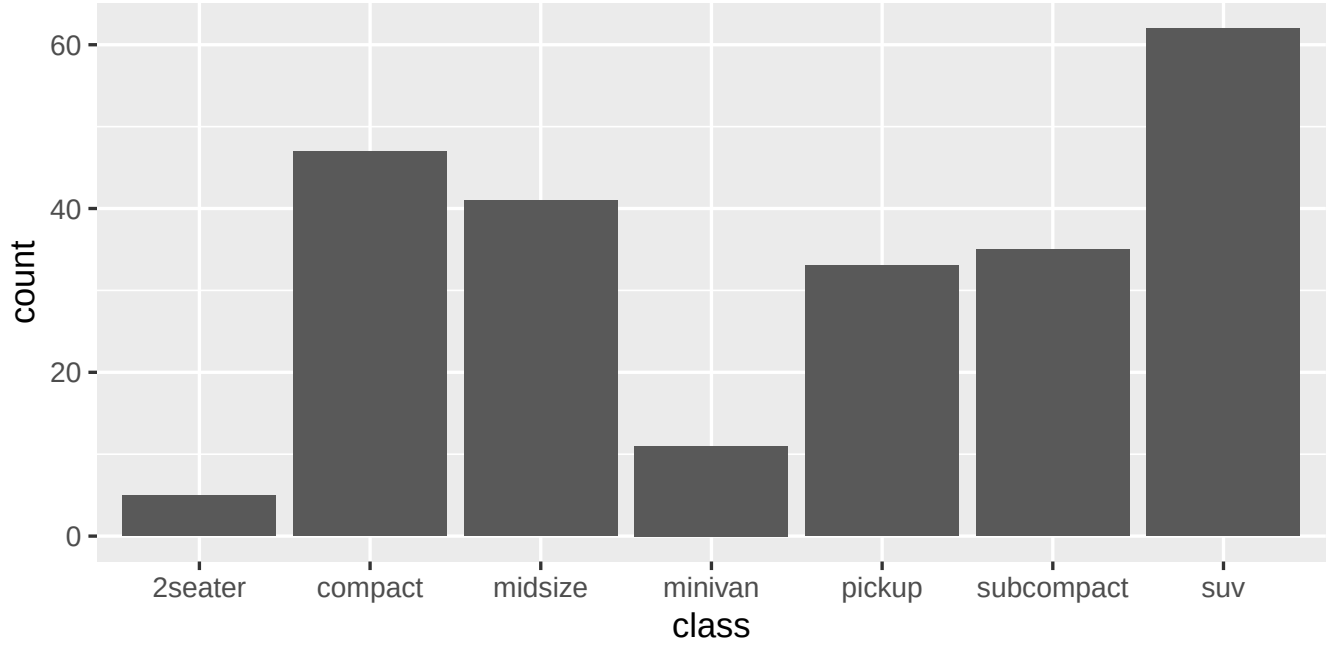
Bar Charts for Categorical Data

- A bar chart is a graph where for each category a bar with a height proportional to the count in the respective category is drawn.

- Along x-axis the categories (or levels) are displayed.

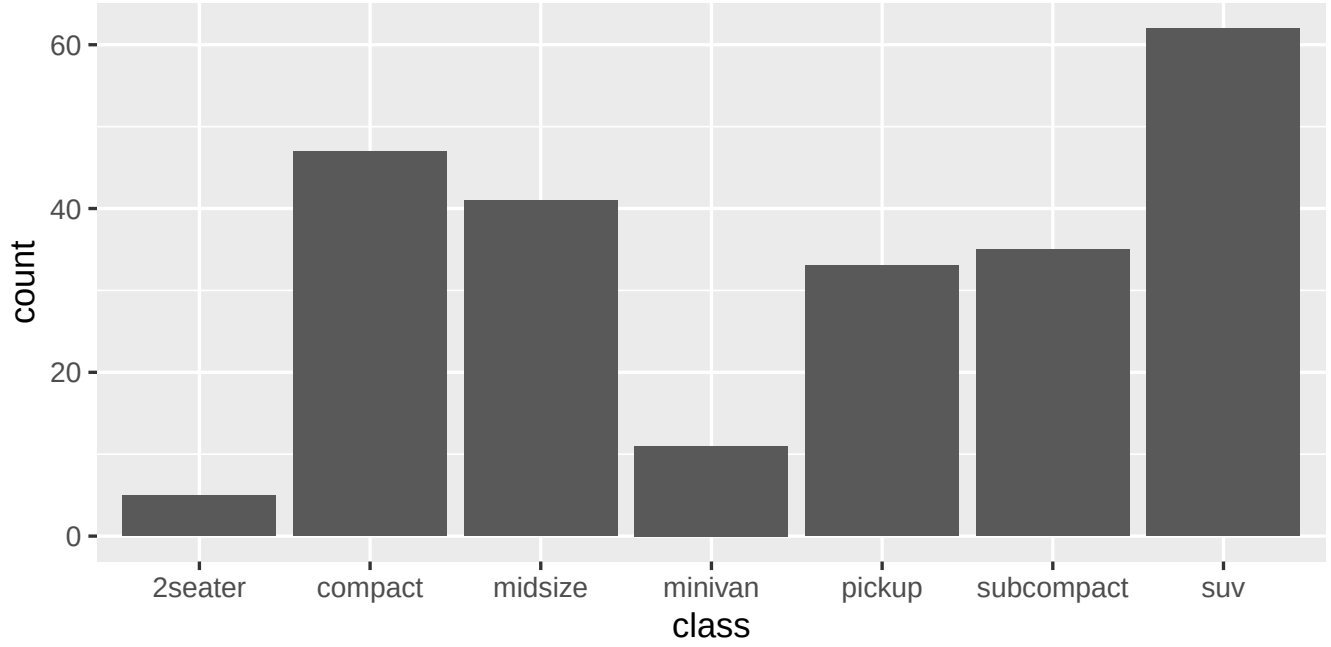
Stat-count – Bar Charts

```
> ggplot(data = mpg) +
+   geom_bar(mapping = aes(x = class))
```



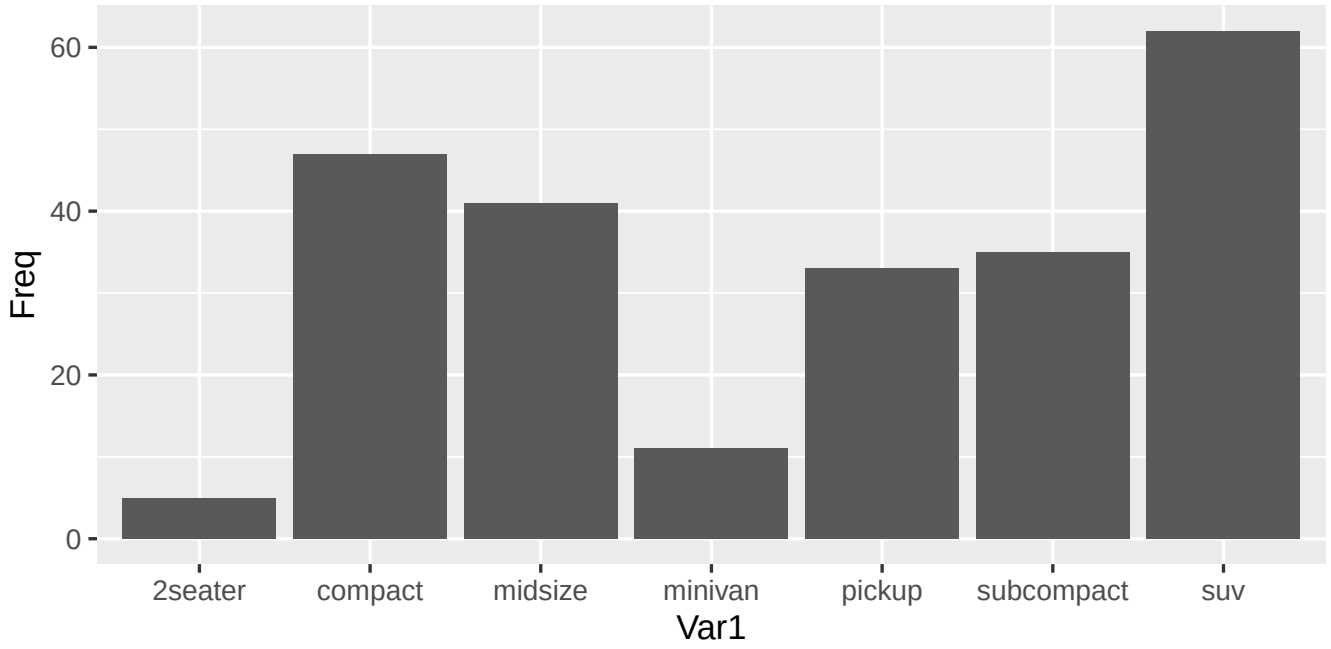
Stat-count – Bar Charts

```
> ggplot(data = mpg) +
+   stat_count(mapping = aes(x = class))
```



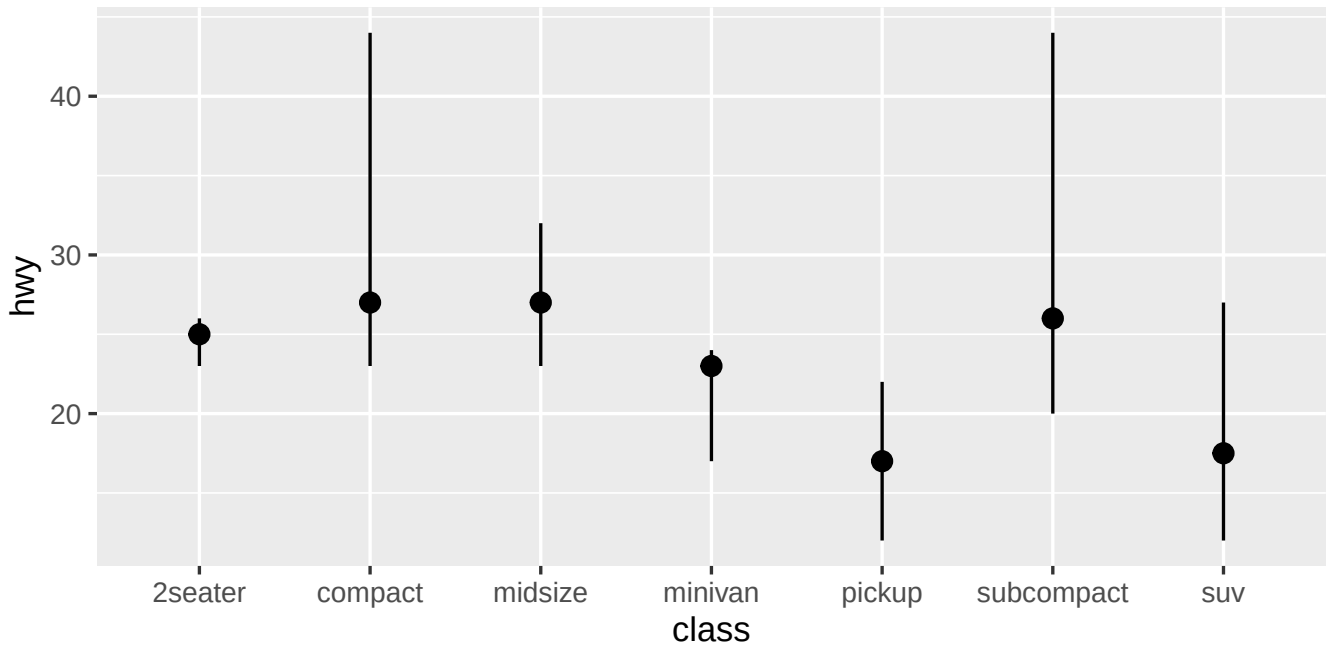
Stat-count – Bar Charts

```
> table= as.data.frame(table(mpg$class))
> ggplot(data = table) +
+ geom_bar(mapping = aes(x = Var1, y=Freq),
+   stat="identity")
```



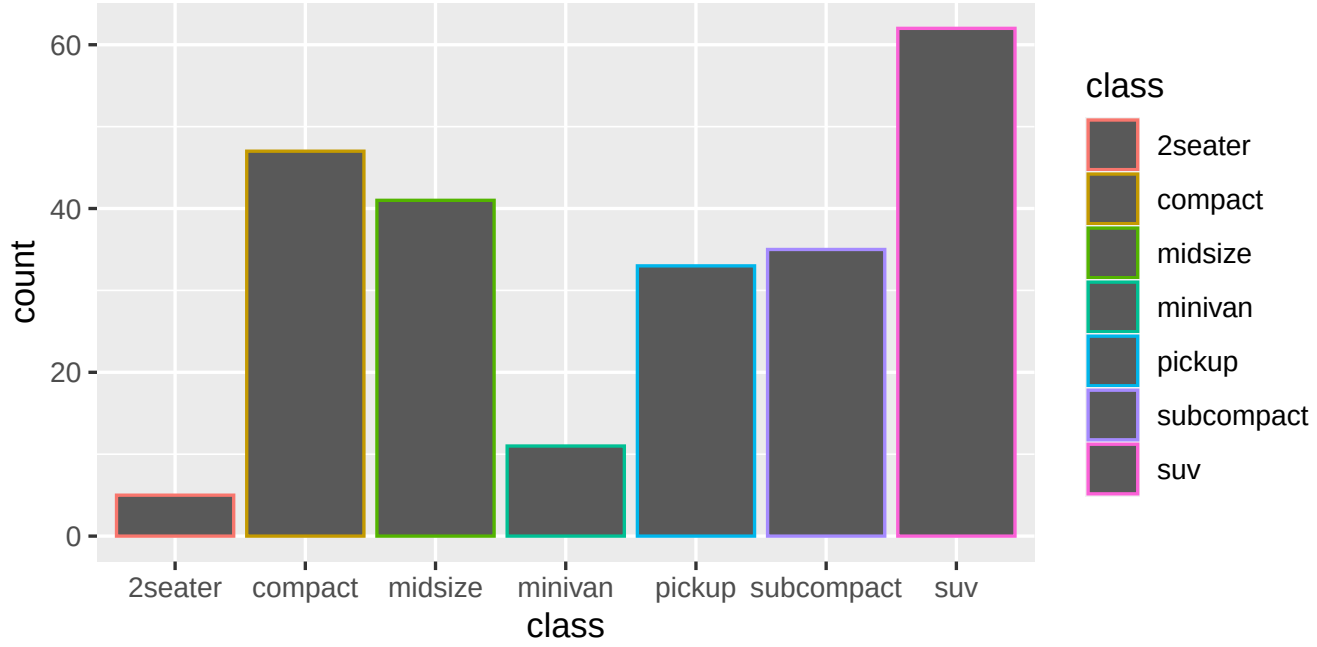
Summary

```
> ggplot(data = mpg) +
+   stat_summary(
+     mapping = aes(x = class, y = hwy),
+     fun.min = min,
+     fun.max = max,
+     fun = median
+   )
```



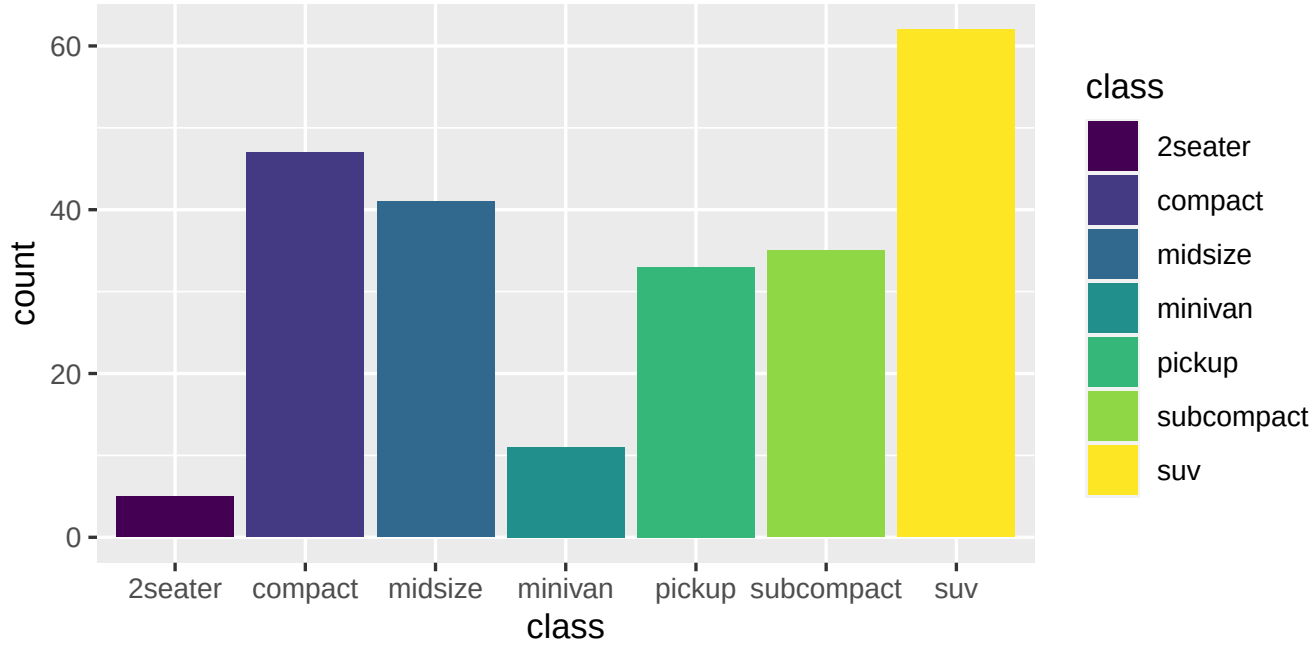
Colours

```
> ggplot(data = mpg) +
+ geom_bar(mapping = aes(x = class, colour = class))
```



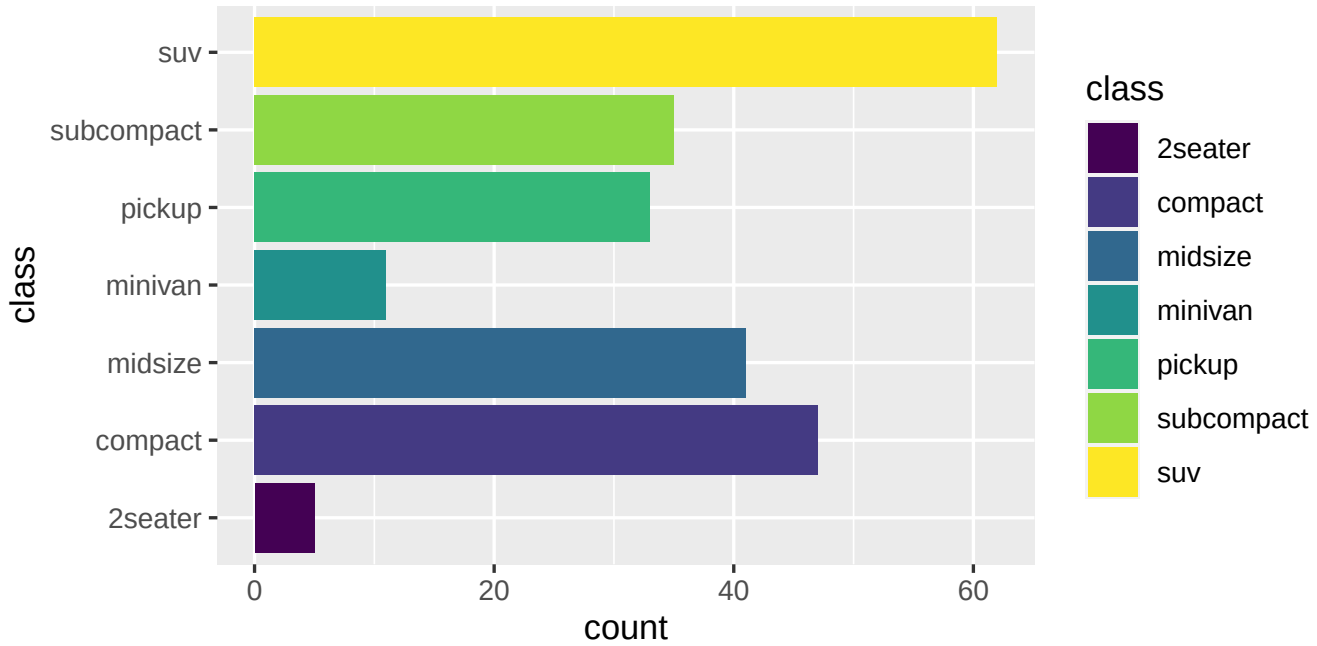
Colours

```
> ggplot(data = mpg) +
+   geom_bar(mapping = aes(x = class, fill = class)) +
+   scale_fill_viridis_d()
```



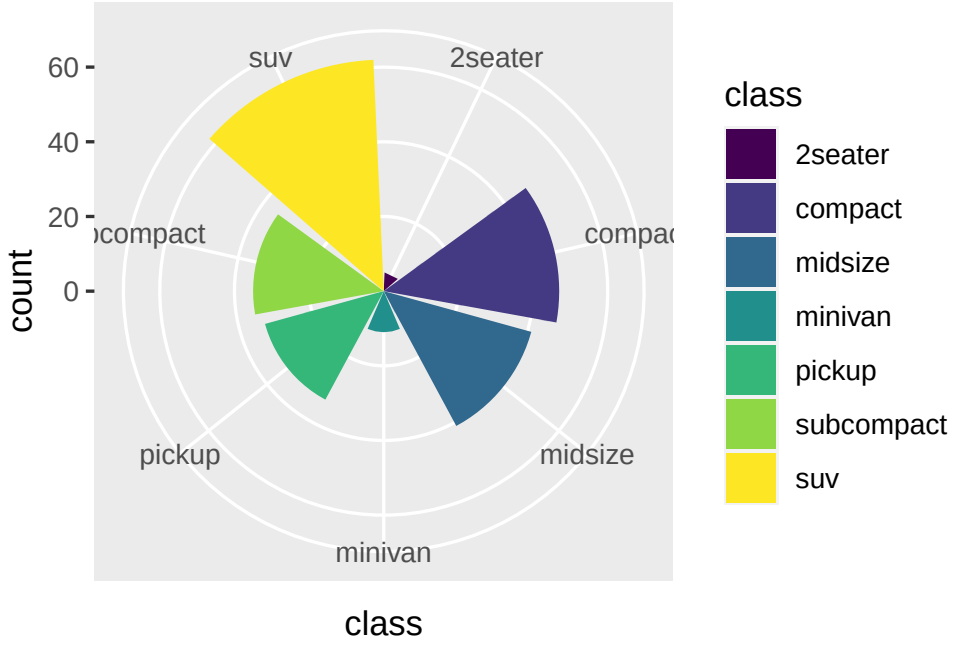
Colours

```
> ggplot(data = mpg) +
+   geom_bar(mapping = aes(x = class, fill = class)) +
+   scale_fill_viridis_d() + coord_flip()
```



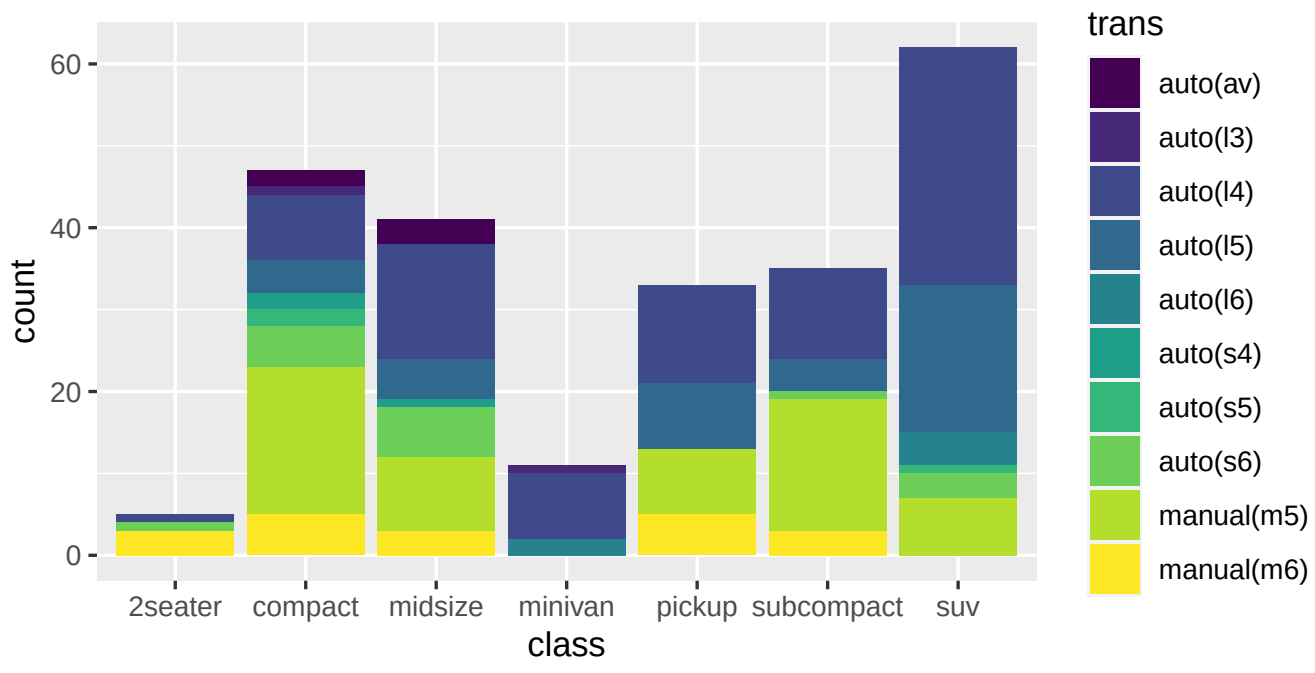
Colours

```
> ggplot(data = mpg) +
+   geom_bar(mapping = aes(x = class, fill = class)) +
+   scale_fill_viridis_d() + coord_polar()
```



Stacked

```
> ggplot(data = mpg) +
+   geom_bar(mapping = aes(x = class, fill = trans)) +
+   scale_fill_viridis_d()
```



Stacked

```
> ggplot(data = mpg) +
+   geom_bar(mapping = aes(x = class, fill = trans),
+            position="dodge") +
+   scale_fill_viridis_d()
```

