MB5370 Module 04. Workshop 2 - Using ggplot2 for communication

Shreyash G Bhandary

2024-05-15

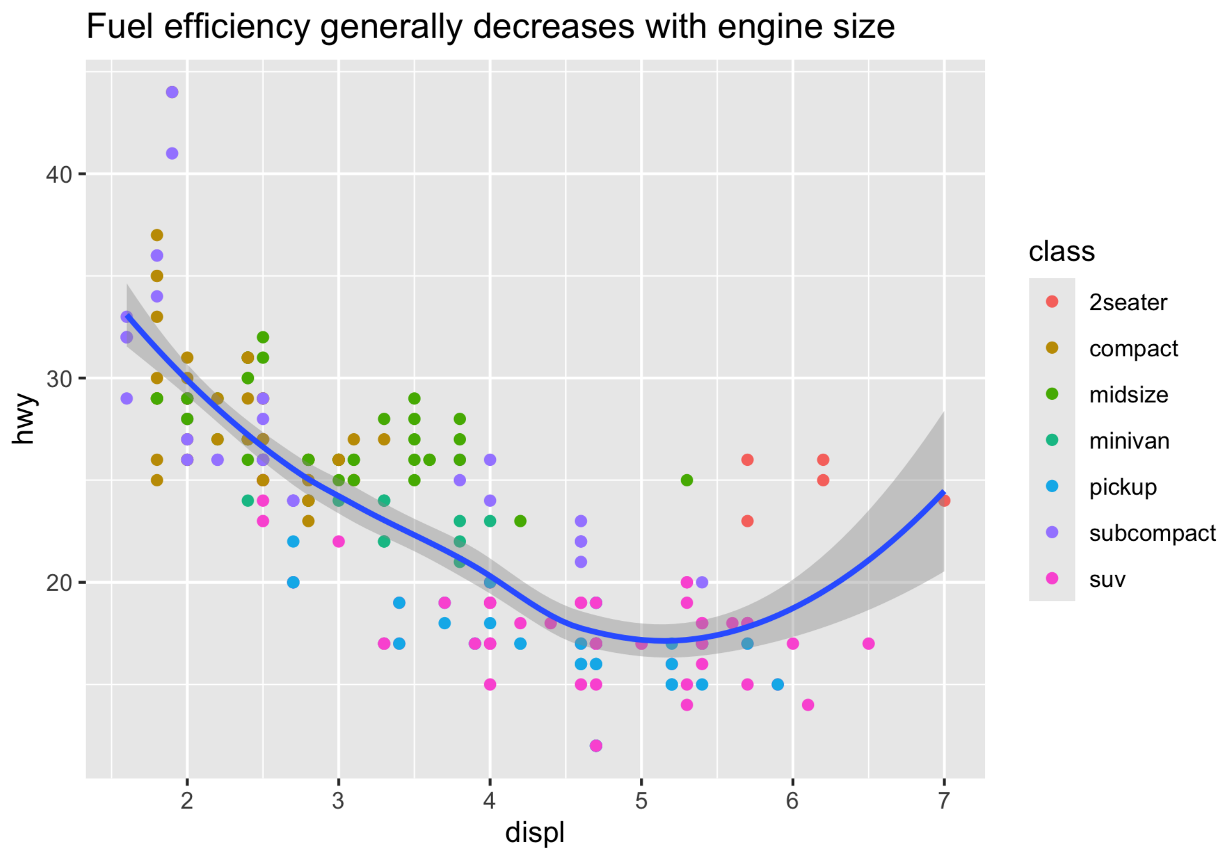
```r
# Load ggplot2 package
library(ggplot2)
```

#Adding a title for your ggplot

```r
ggplot(mpg, aes(displ,hwy)) +
  geom_point(aes(colour = class)) +
  geom_smooth(se.e = FALSE) +
  labs(title = "Fuel efficiency generally decreases with engine size")
```

```
## Warning in geom_smooth(se.e = FALSE): Ignoring unknown parameters: ‘se.e‘
```

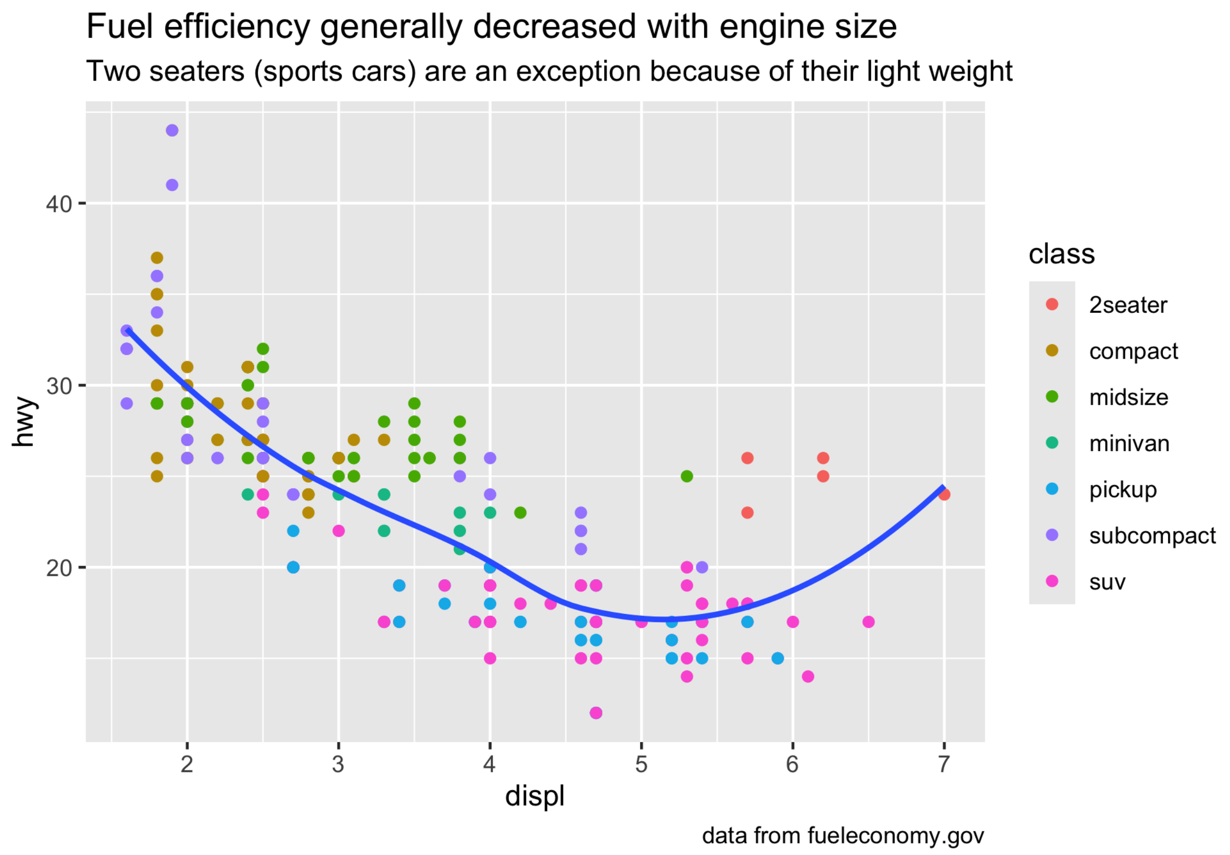
```
## ‘geom_smooth()‘ using method = ’loess’ and formula = ’y ~ x’
```



#Adding a sub-title and a caption

```
ggplot(mpg, aes(displ,hwy)) +
  geom_point(aes(colour = class)) +
  geom_smooth(se = FALSE) +
  labs(
    title = "Fuel efficiency generally decreased with engine size",
    subtitle = "Two seaters (sports cars) are an exception because of their light weight",
    caption = "data from fueleconomy.gov"
  )
```

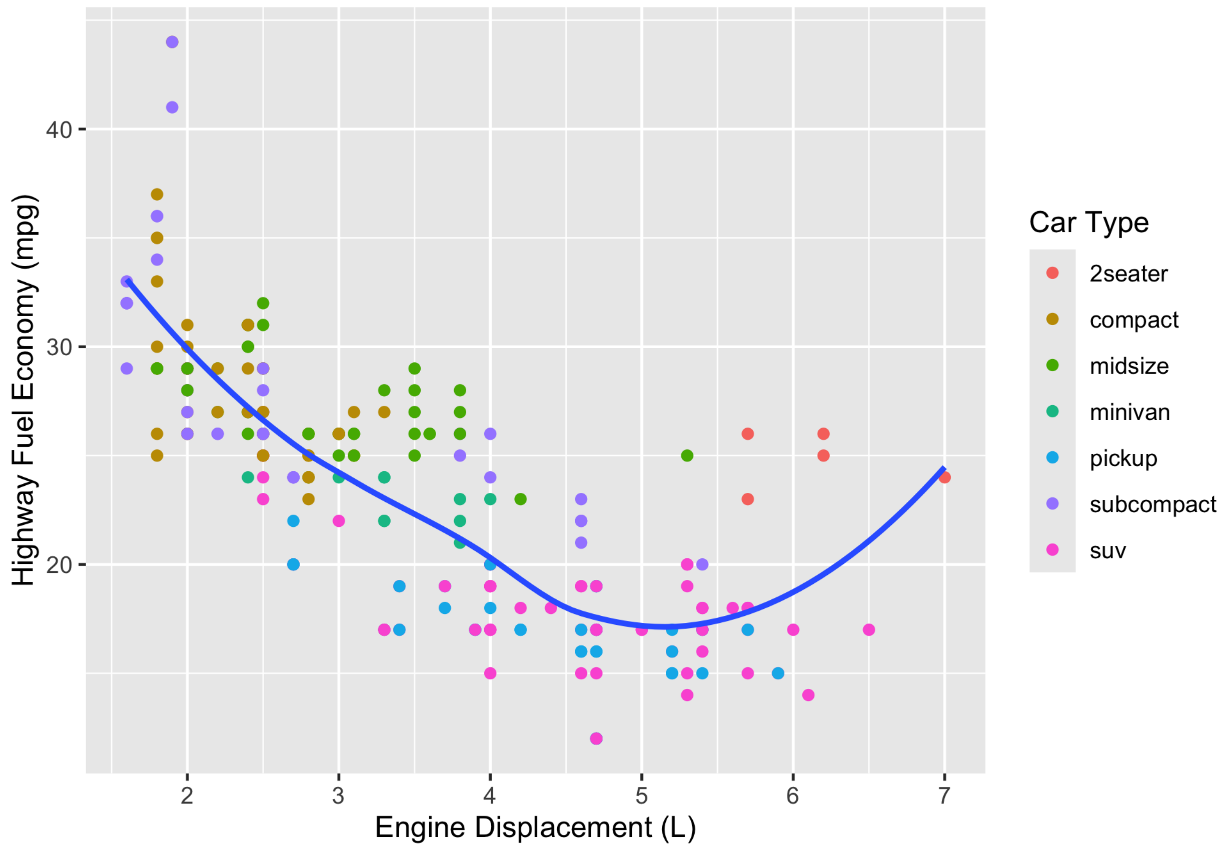
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



#labs() is used to replace the axis and legend titles

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(colour = class)) +
  geom_smooth(se = FALSE) +
  labs(
    x = "Engine Displacement (L)",
    y = "Highway Fuel Economy (mpg)",
    colour = "Car Type"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



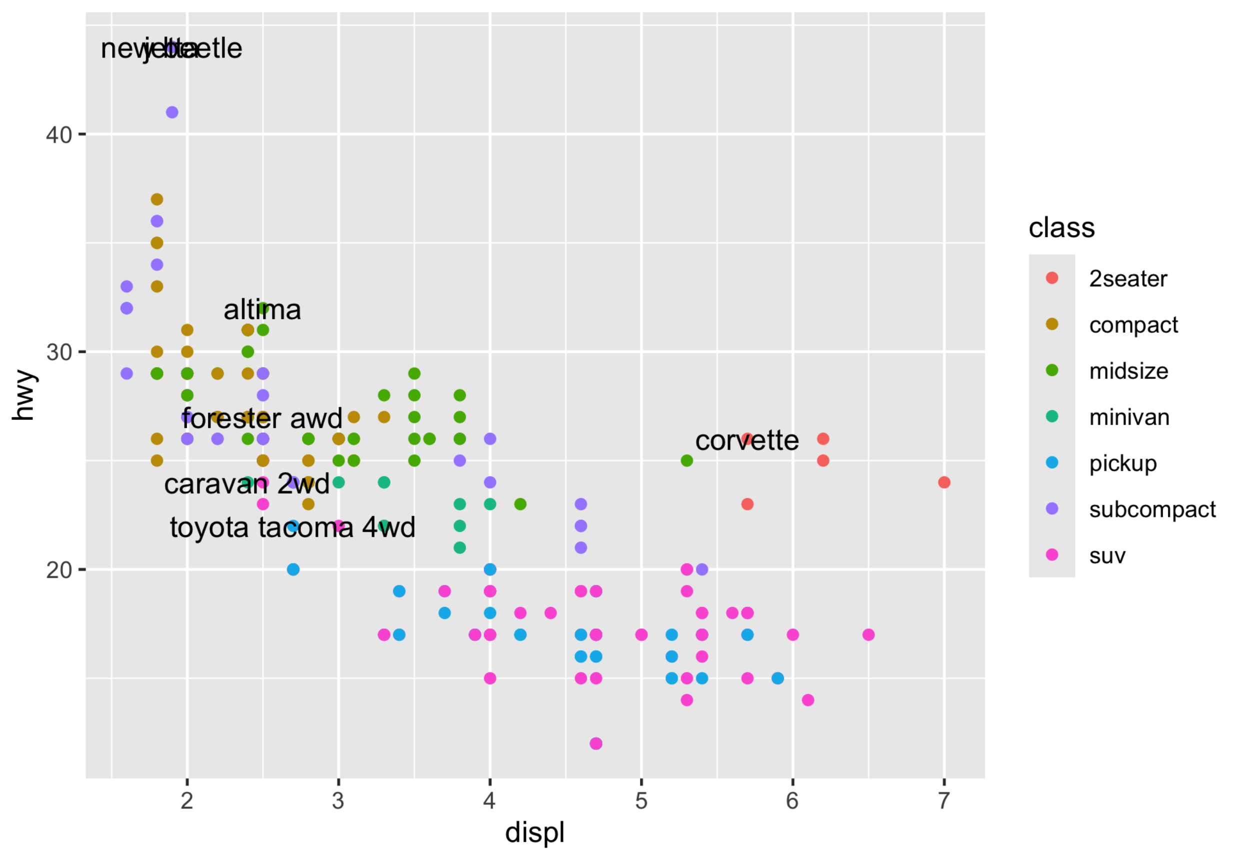
```
library(dplyr) # for data manipulation functions
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(magrittr) # for the %>%
```

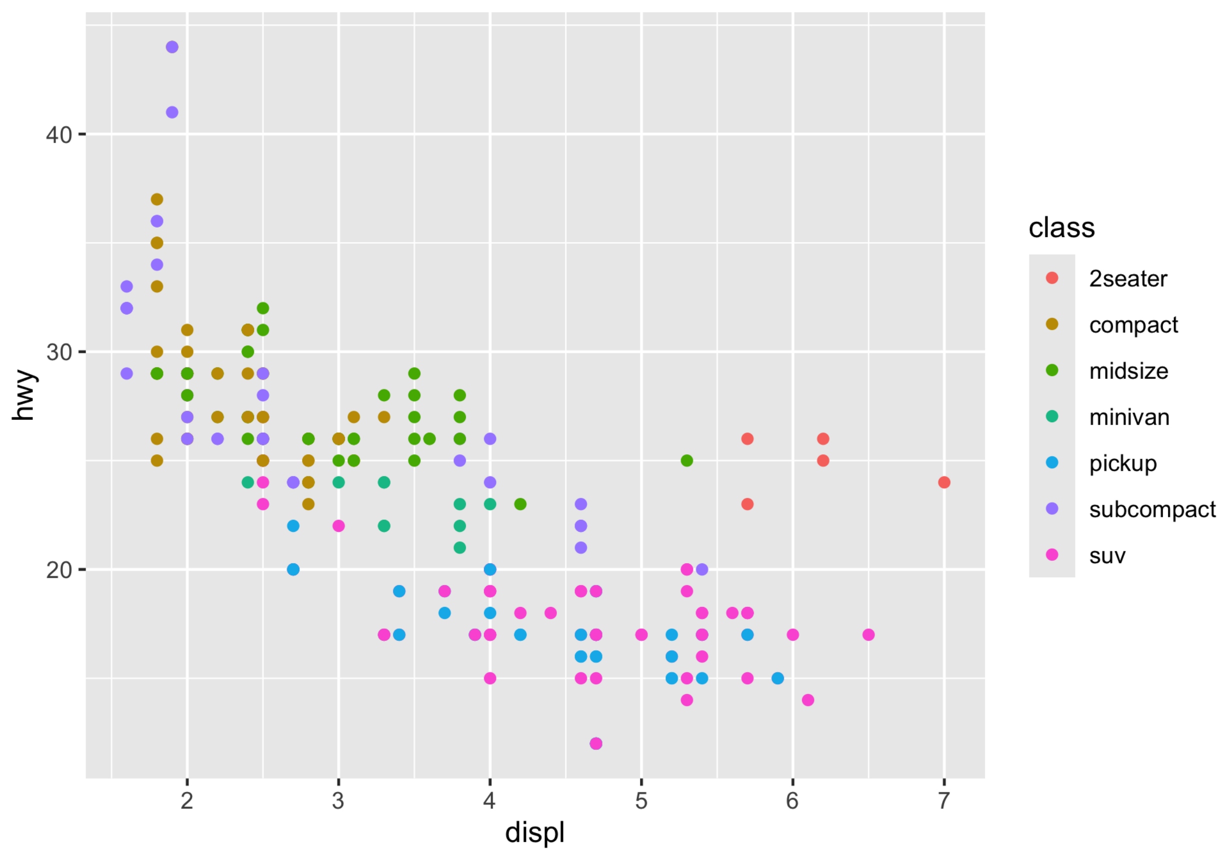
#Adding annotations in your plot using the geom_text()

```
best_in_class <- mpg %>%
  group_by(class) %>%
  filter(row_number(desc(hwy)) == 1)

ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(colour = class)) +
  geom_text(aes(label = model), data = best_in_class)
```



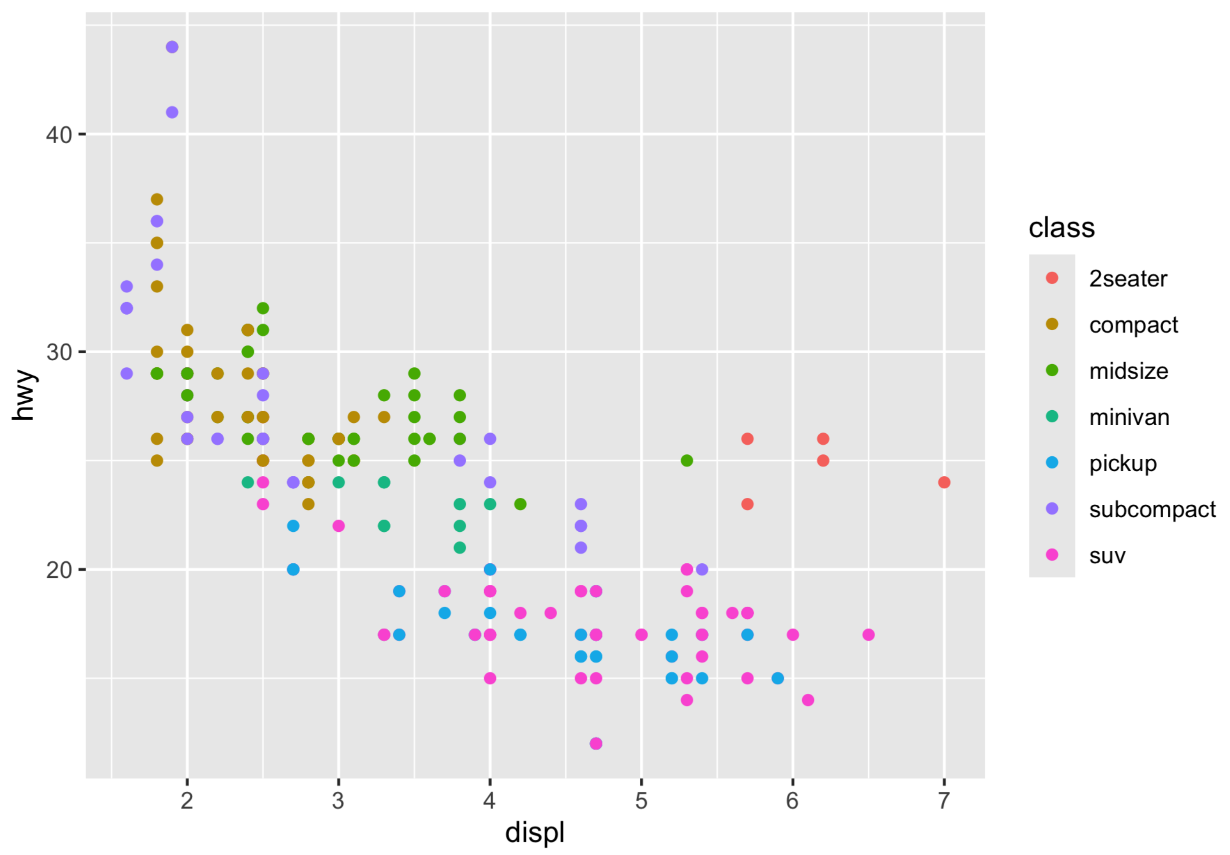
##Scales #R automatically adds the scale when you code the ggplot.

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



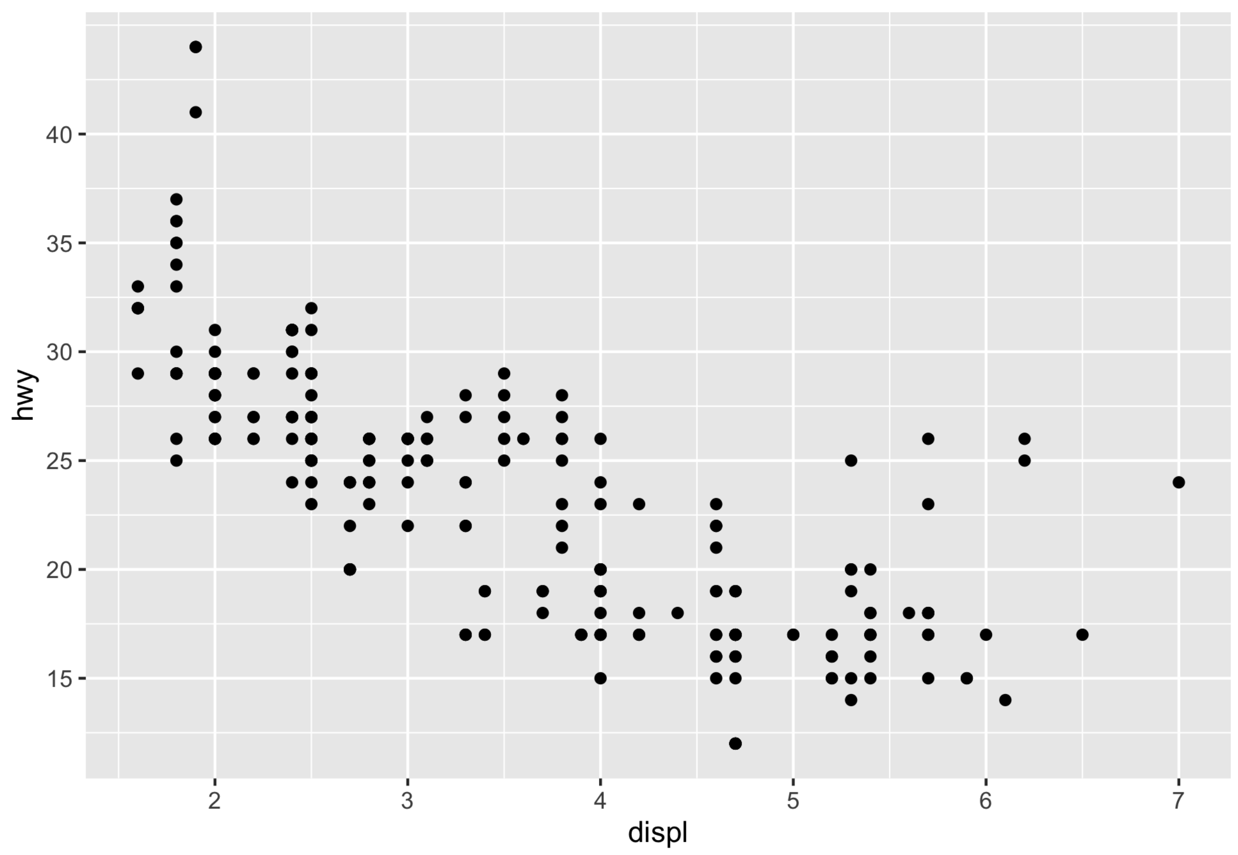
#Tweaking the scales.

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(colour = class)) +
  scale_x_continuous() +
  scale_y_continuous() +
  scale_colour_discrete()
```



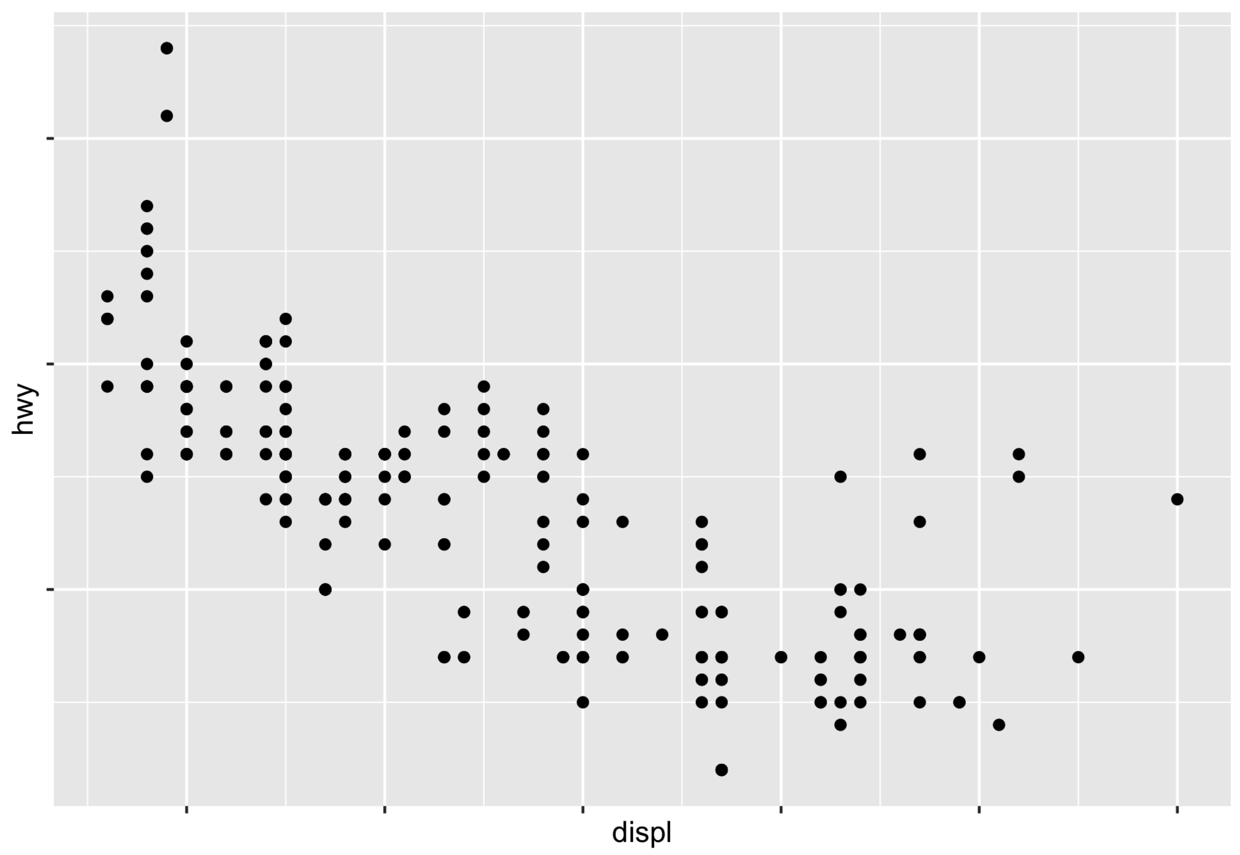
#Axis Ticks

```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point() +
  scale_y_continuous(breaks = seq(15, 40, by = 5))
```



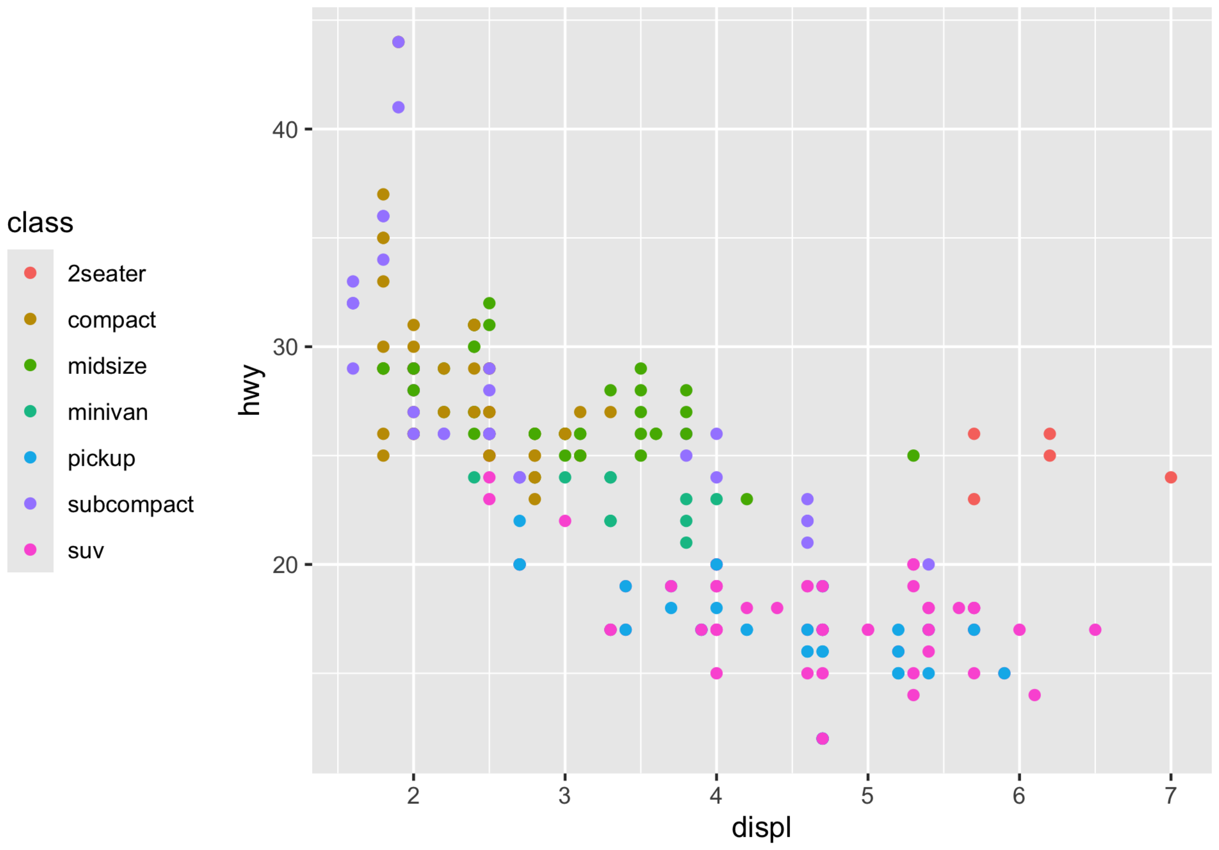
```r
  # seq(lower limit, upper limit, by = interval)
```

```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point() +
  scale_x_continuous(labels = NULL) +
  scale_y_continuous(labels = NULL)
```

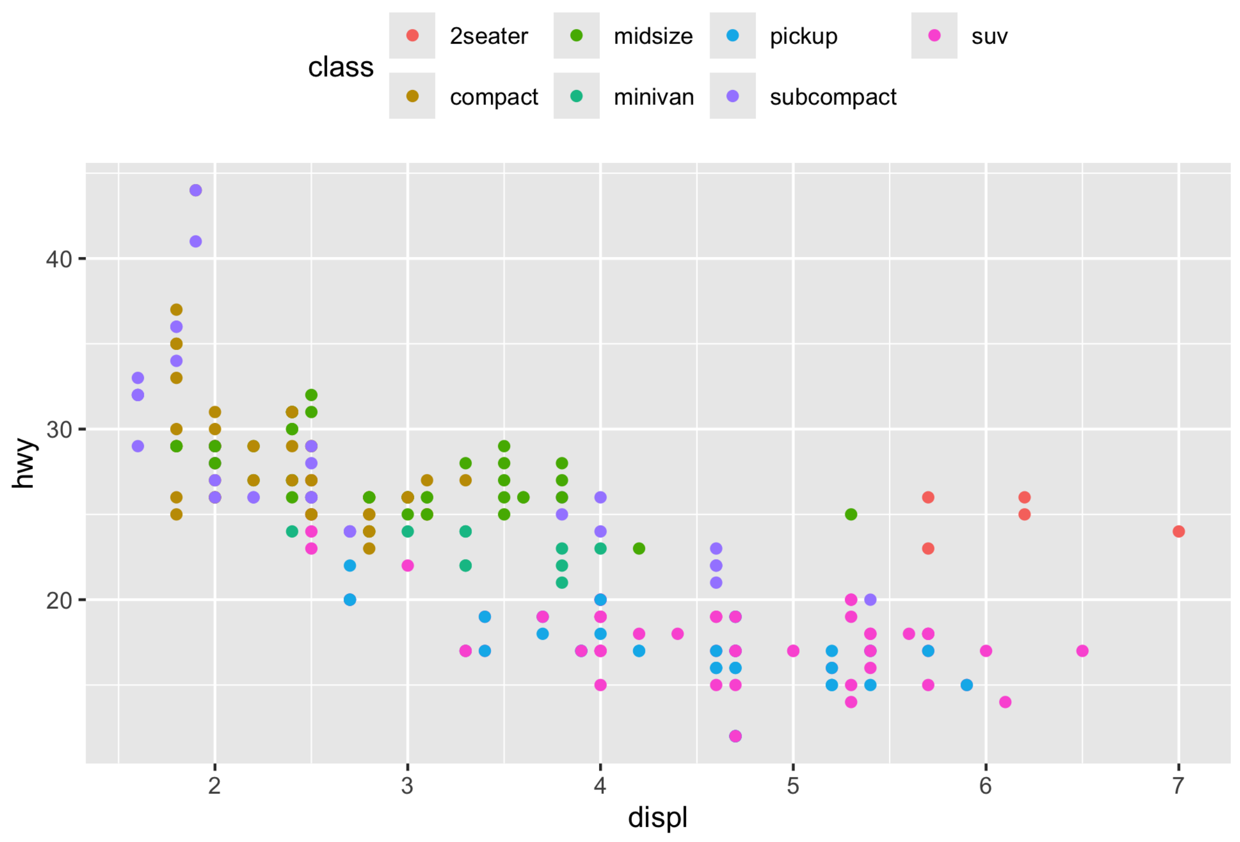


#Legends and colour schemes

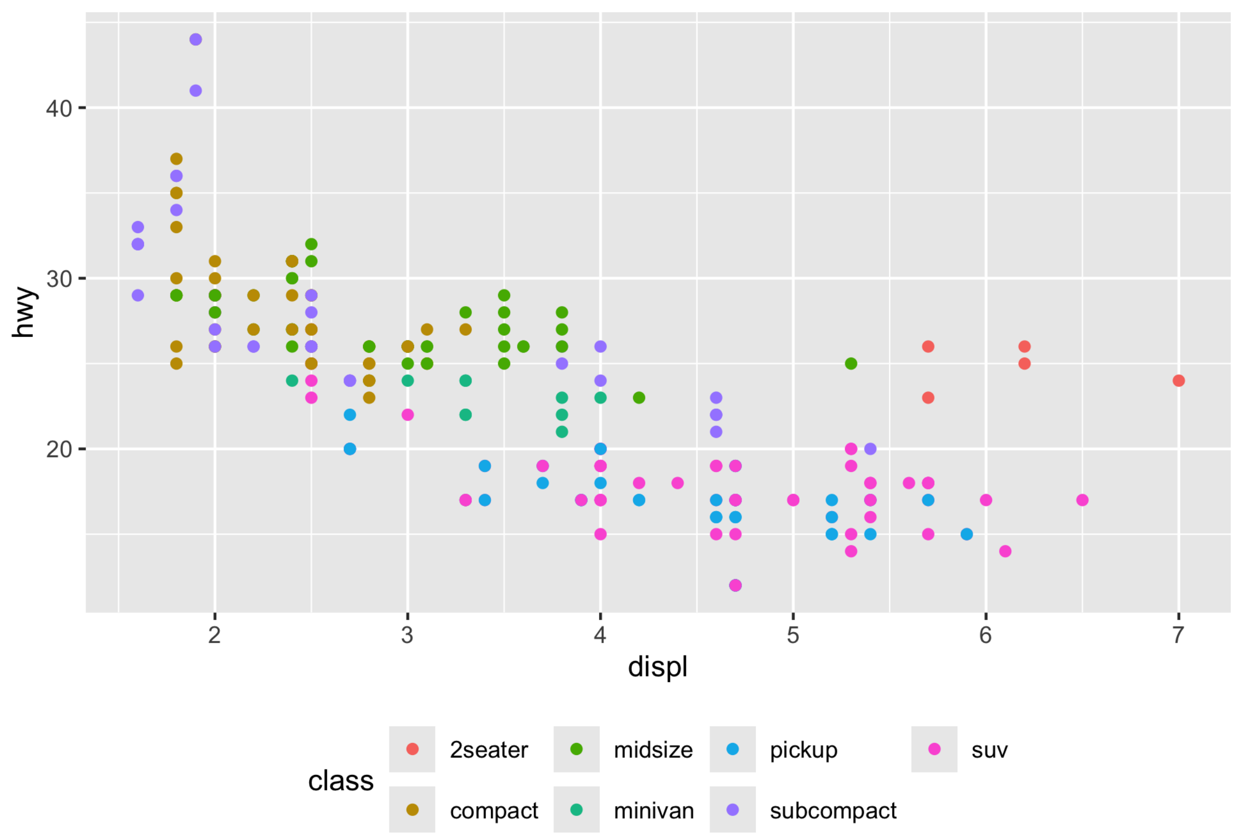
```
base <- ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(colour = class))

base + theme(legend.position = "left")
```



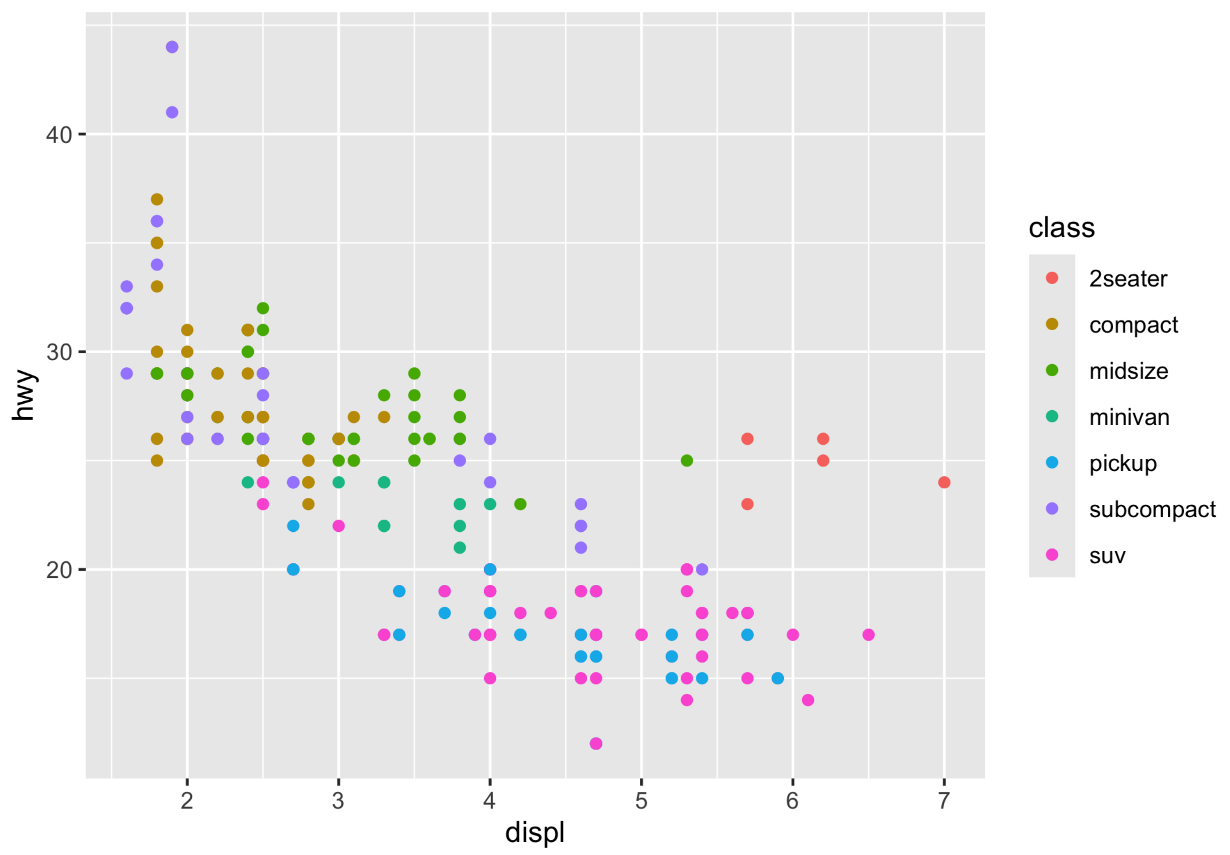
```
base + theme(legend.position = "top")
```



```
base + theme(legend.position = "bottom")
```

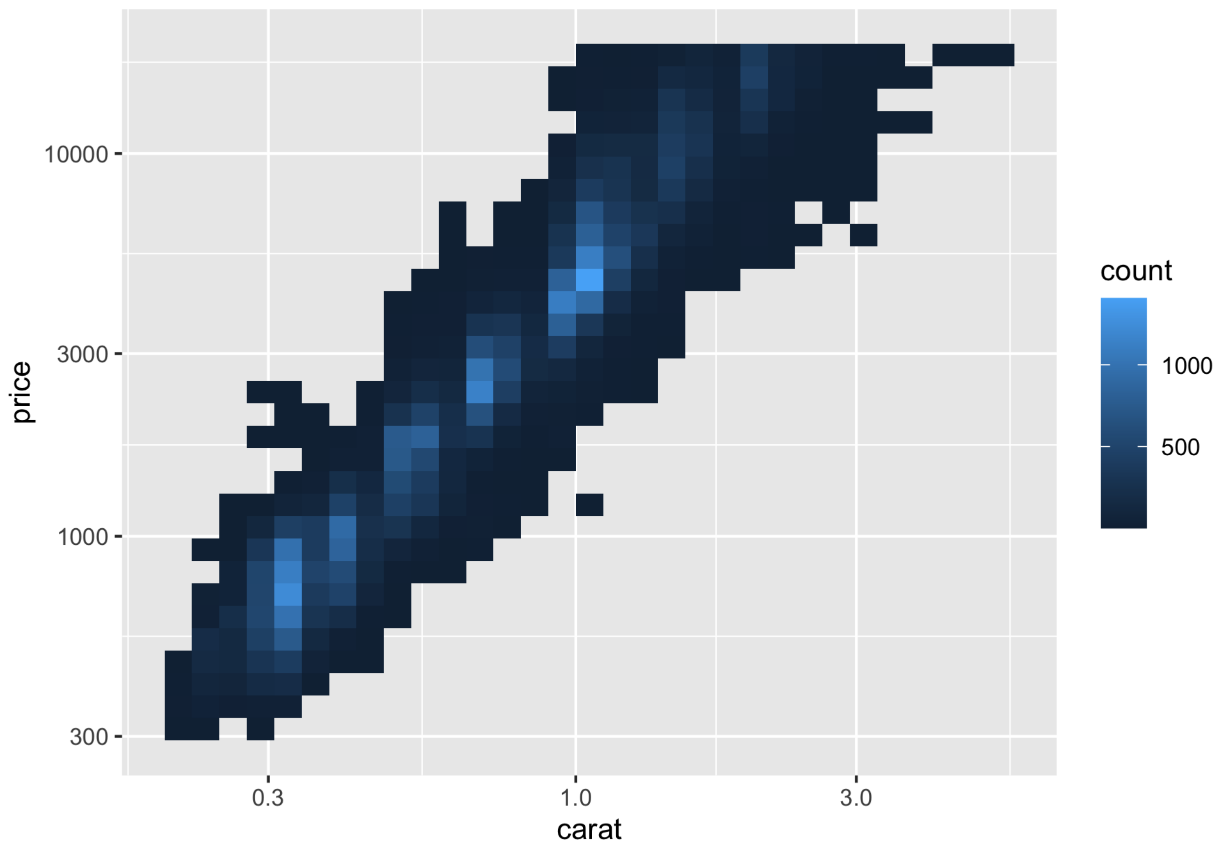


```r
base + theme(legend.position = "right")
```



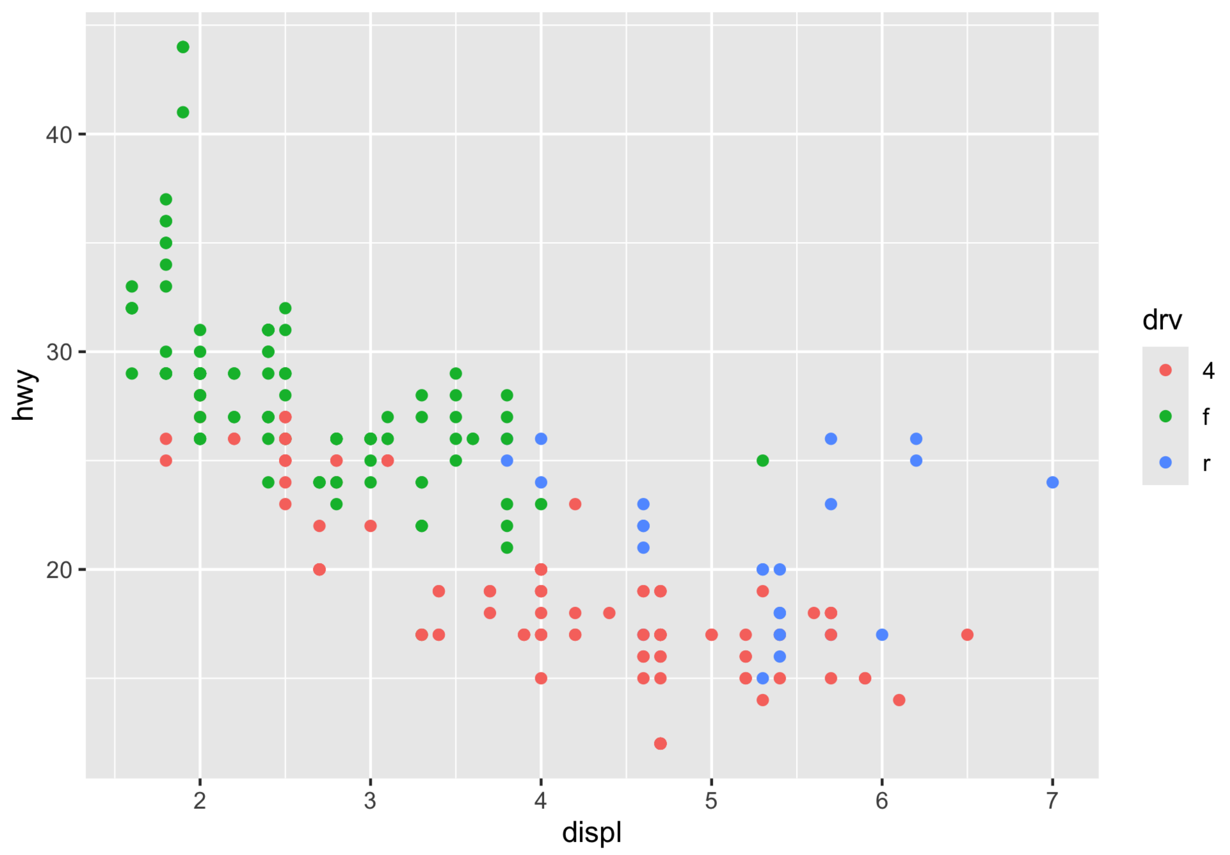
#Replacing a scale

```
ggplot(diamonds, aes(carat, price)) +
  geom_bin2d() +
  scale_x_log10() +
  scale_y_log10()
```

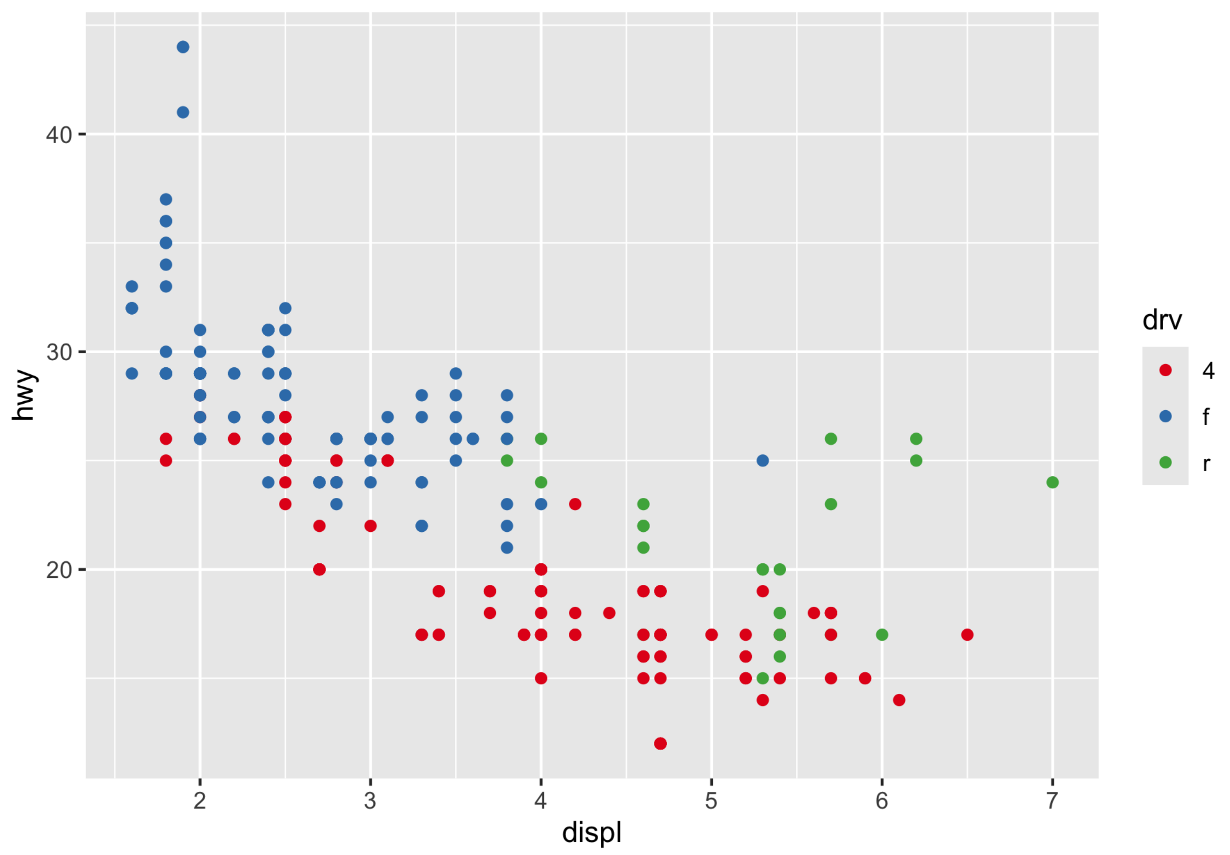


#Customizing the colour scale for better visualisation.

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

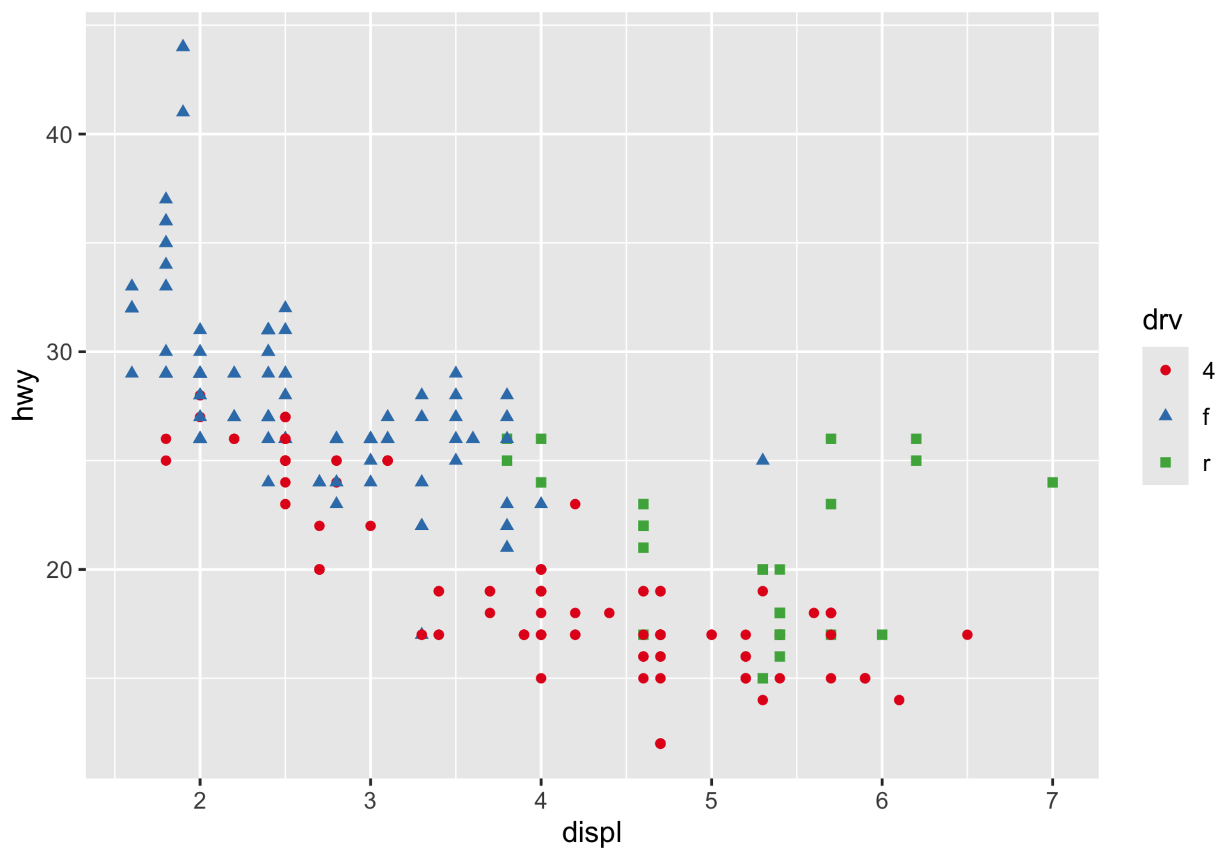


```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = drv)) +
  scale_colour_brewer(palette = "Set1")
```



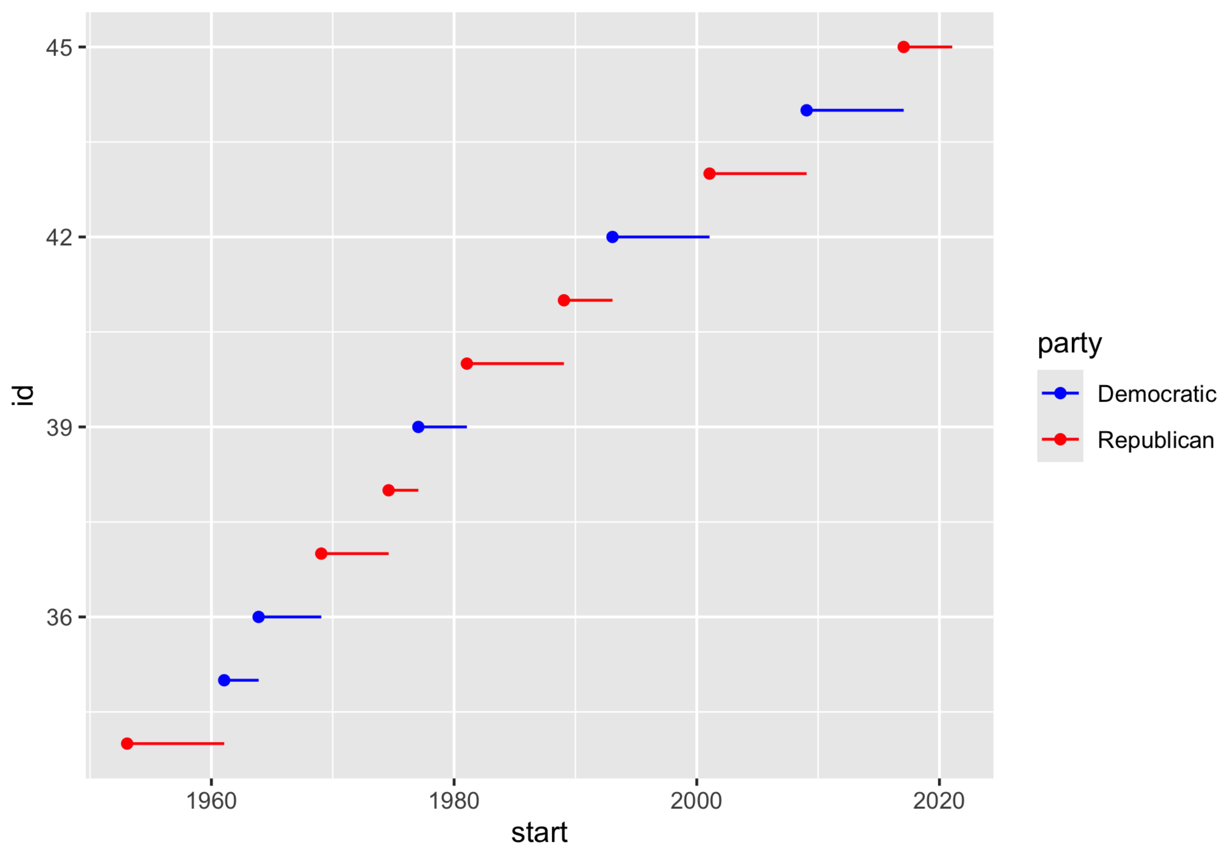
#Redundant shape mapping

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = drv, shape = drv)) +
  scale_colour_brewer(palette = "Set1")
```



#Setting a colour palatte of your own using a set of pre-defined colours.

```
presidential %>%
  mutate(id = 33 + row_number()) %>%
  ggplot(aes(start, id, colour = party)) +
    geom_point() +
    geom_segment(aes(xend = end, yend = id)) +
    scale_colour_manual(values = c(Republican = "red", Democratic = "blue"))
```



#Installing the Viridis & Hexbin packages.

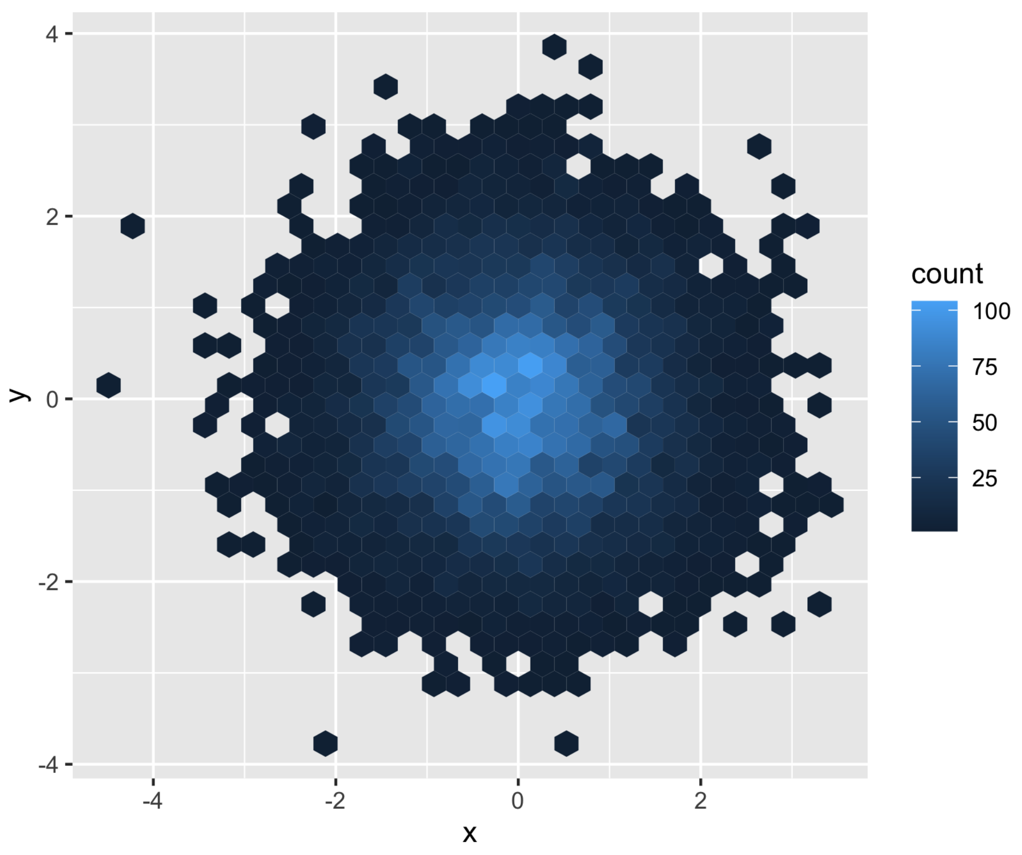
```
#install.packages('viridis')
#install.packages('hexbin')
library(viridis)
```

```
## Loading required package: viridisLite
```

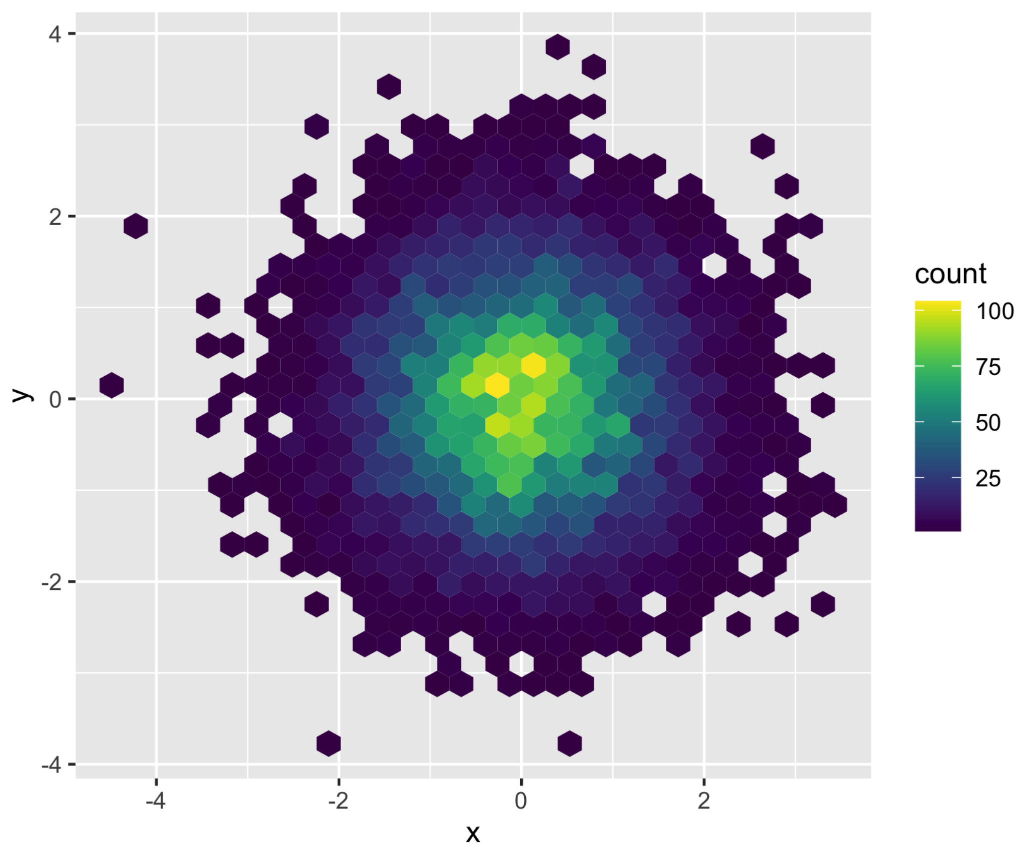
```
library(hexbin)
```

#Trying out the new viridis packages.

```
df <- tibble( # note we're just making a fake dataset so we can plot it
  x = rnorm(10000),
  y = rnorm(10000)
)
ggplot(df, aes(x, y)) +
  geom_hex() + # a new geom!
  coord_fixed()
```



```
ggplot(df, aes(x, y)) +
  geom_hex() +
  viridis::scale_fill_viridis() +
  coord_fixed()
```

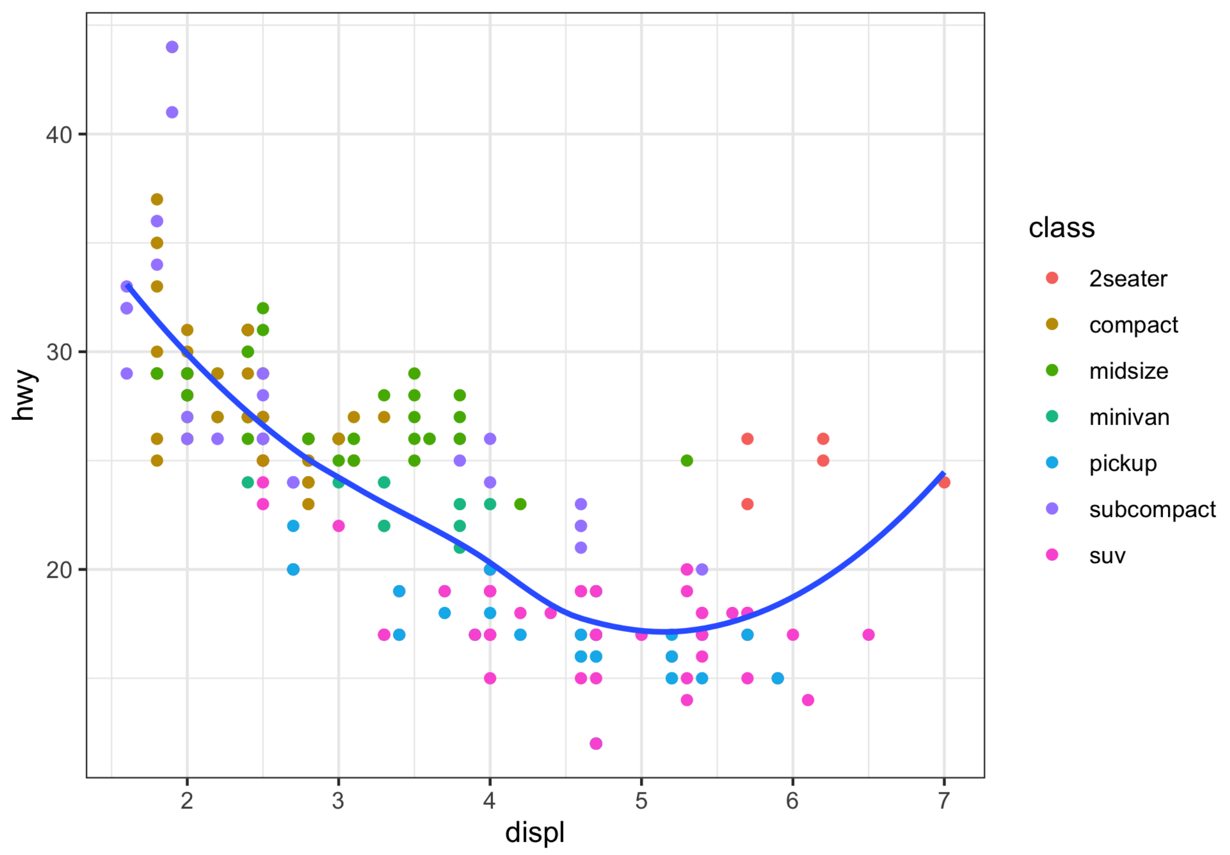


##Themes

#Using the default themes

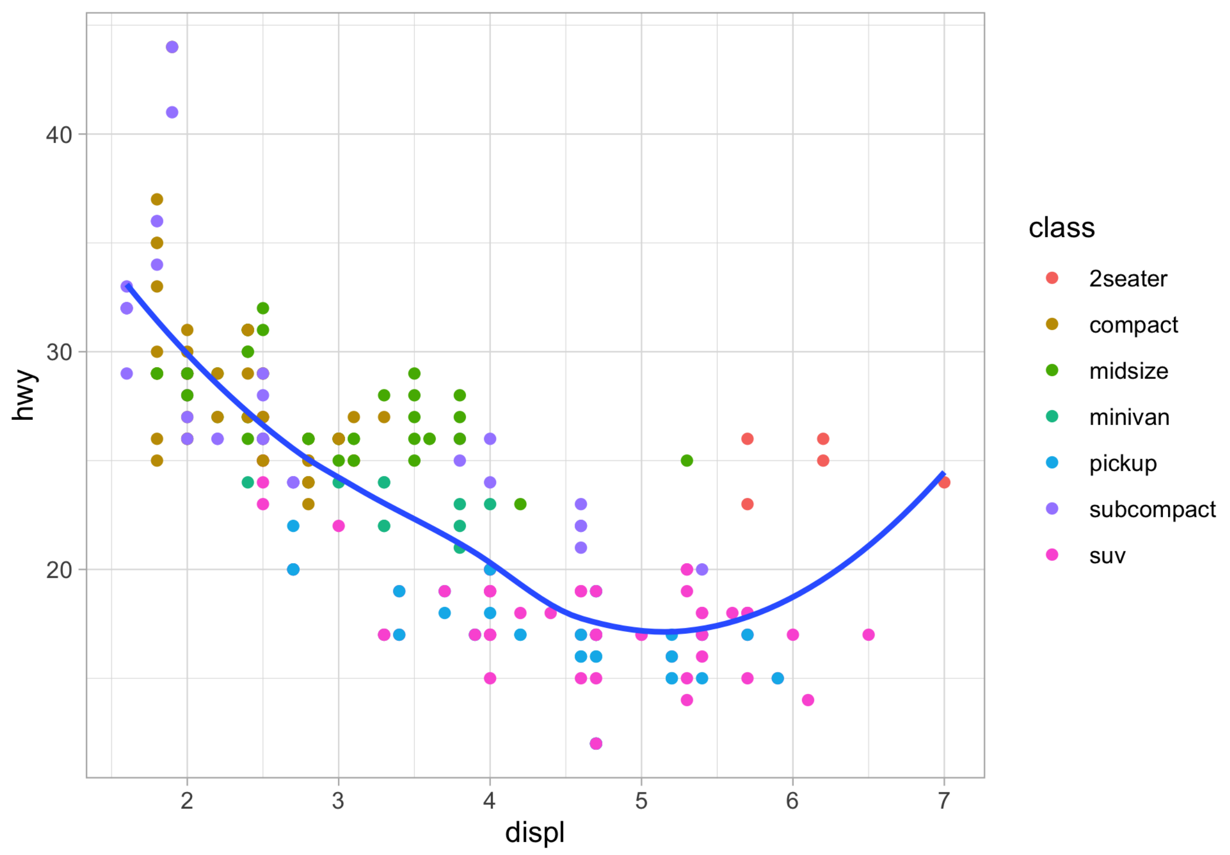
```
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme_bw()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



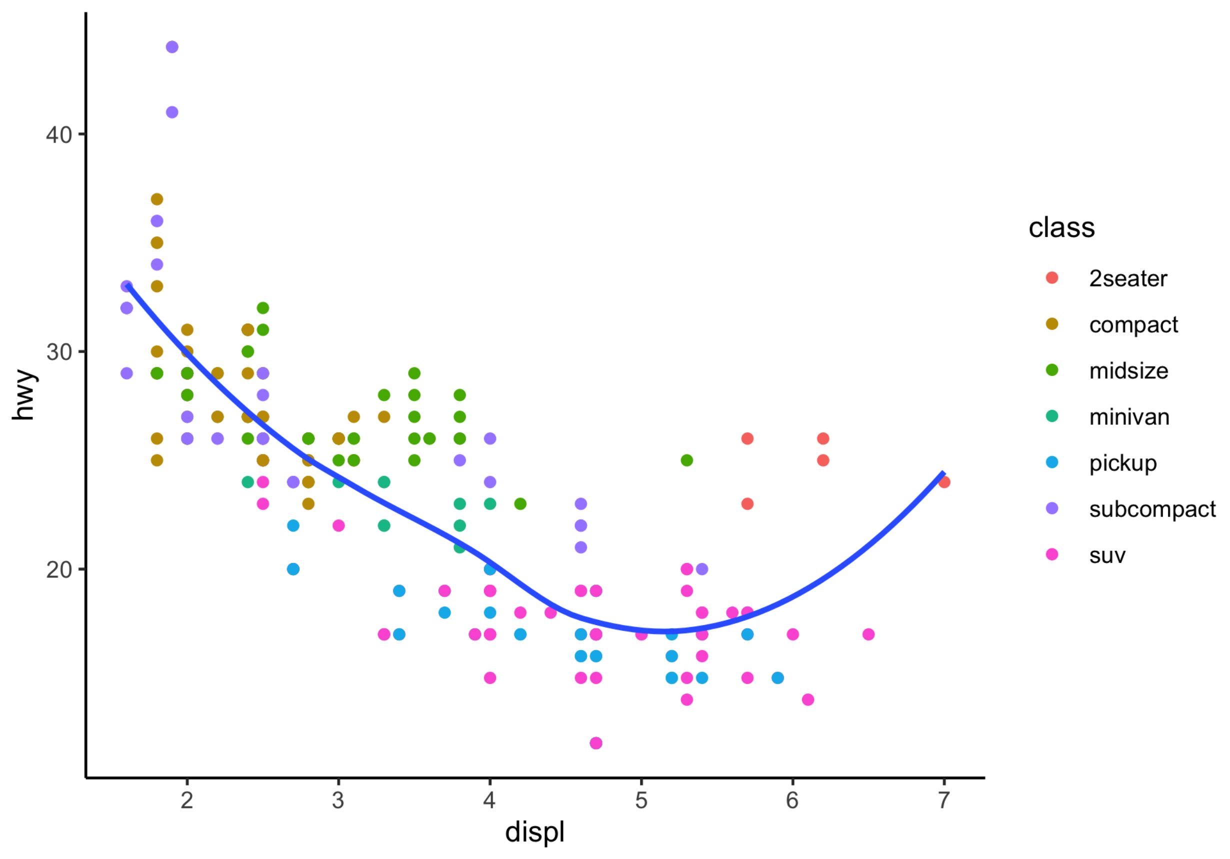
```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme_light()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



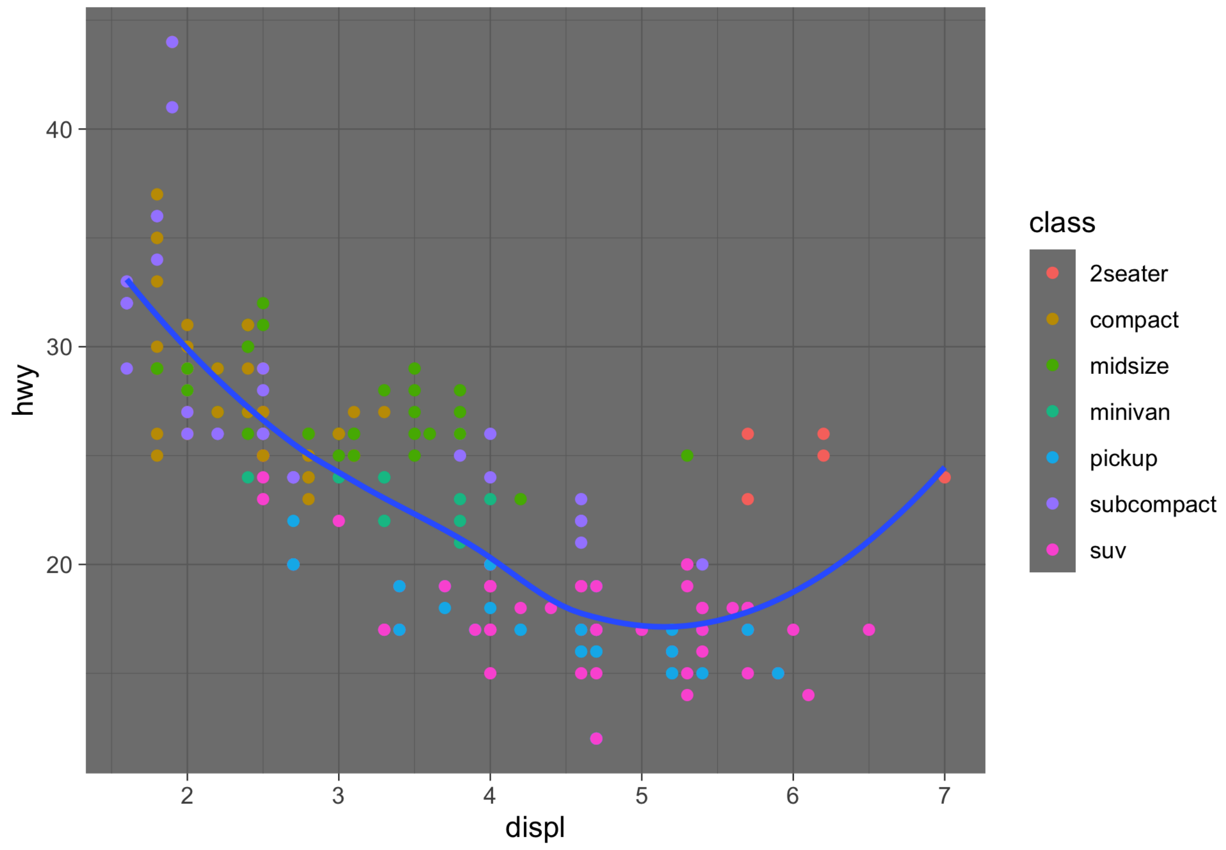
```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme_classic()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme_dark()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



#Ben's Theme

```
theme (panel.border = element_blank(),
       panel.grid.minor.x = element_blank(),
       panel.grid.minor.y = element_blank(),
       legend.position="bottom",
       legend.title=element_blank(),
       legend.text=element_text(size=8),
       panel.grid.major = element_blank(),
       legend.key = element_blank(),
       legend.background = element_blank(),
       axis.text.y=element_text(colour="black"),
       axis.text.x=element_text(colour="black"),
       text=element_text(family="Arial"))
```

```
## List of 12
##  $ text                  :List of 11
##   ..$ family        : chr "Arial"
##   ..$ face          : NULL
##   ..$ colour        : NULL
##   ..$ size          : NULL
##   ..$ hjust         : NULL
##   ..$ vjust         : NULL
##   ..$ angle         : NULL
##   ..$ lineheight    : NULL
##   ..$ margin        : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x       :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "black"
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y       :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "black"
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.background : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key        : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.text       :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : num 8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.position   : chr "bottom"
##  $ panel.border      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.grid.major  : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.grid.minor.x: list()
##    ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.grid.minor.y: list()
##    ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

#Theme from jrnold in GitHub

```
#install.packages("devtools")
#library("devtools")
#(c("hadley/ggplot2", "jrnold/ggthemes"))
```

Setting up a knitr to save all the graph outputs