```r
library(ggplot2)

# list datasets
data(package = "ggplot2")

# Data sets in package 'ggplot2':
# diamonds                      Prices of 50,000 round cut diamonds
# economics                     US economic time series
# economics_long                US economic time series
# faithfuld                     2d density estimate of Old Faithful data
# luv_colours                   'colors()' in Luv space
# midwest                       Midwest demographics
# mpg                           Fuel economy data from 1999 and 2008 for 38 popular models
# msleep                        An updated and expanded version of the mammals sleep data
# presidential                  Terms of 11 presidents from Eisenhower to Obama
# seals                         Vector field of seal movements
# txhousing                     Housing sales in TX

head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa...
## 2 audi         a4      1.8  1999     4 manual(m5) f        21    29 p     compa...
## 3 audi         a4      2    2008     4 manual(m6) f        20    31 p     compa...
## 4 audi         a4      2    2008     4 auto(av)   f        21    30 p     compa...
## 5 audi         a4      2.8  1999     6 auto(l5)   f        16    26 p     compa...
## 6 audi         a4      2.8  1999     6 manual(m5) f        18    26 p     compa...
```

```r
# online description
?mpg
dim(mpg)
```

```
## [1] 234  11
```

```r
#
# class levels
unique(mpg$class)
```

```
## [1] "compact"    "midsize"    "suv"        "2seater"    "minivan"
## [6] "pickup"     "subcompact"
```

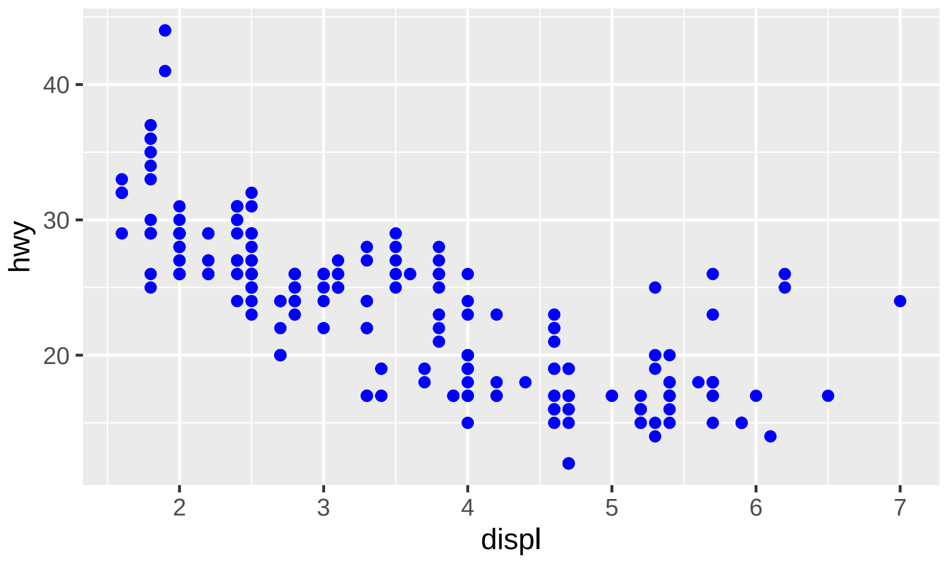
```r
# number of cars in each class level
table(mpg$class)
```

```
##
##    2seater    compact    midsize    minivan     pickup subcompact        suv
##          5         47         41         11         33         35         62
```

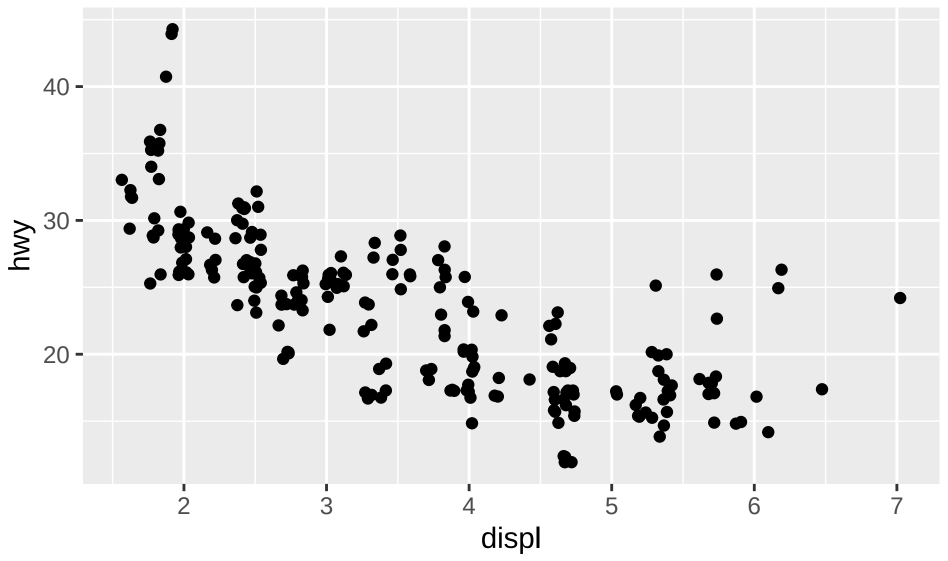
```r
class(table(mpg$class))
```

```
## [1] "table"
```

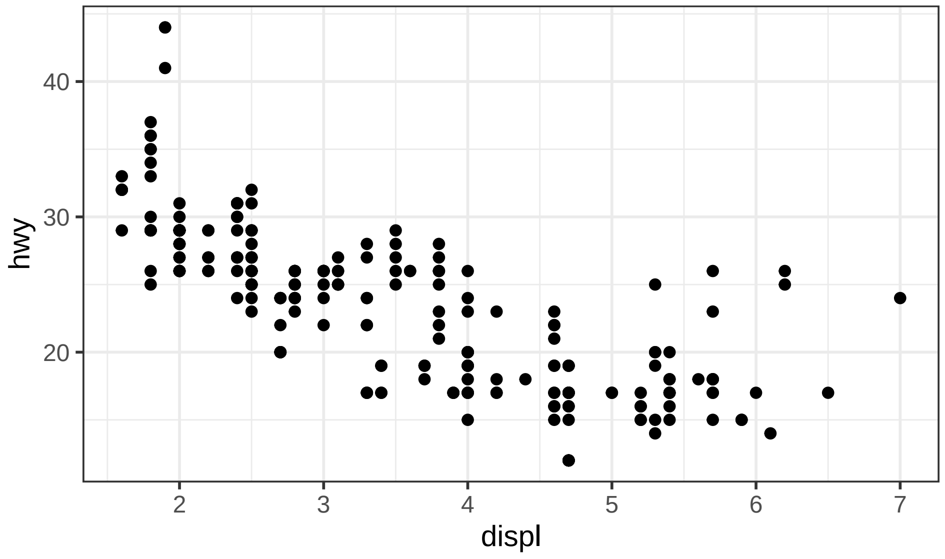
```r
# SCATTERPLOT
#
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy), color = "blue")
```



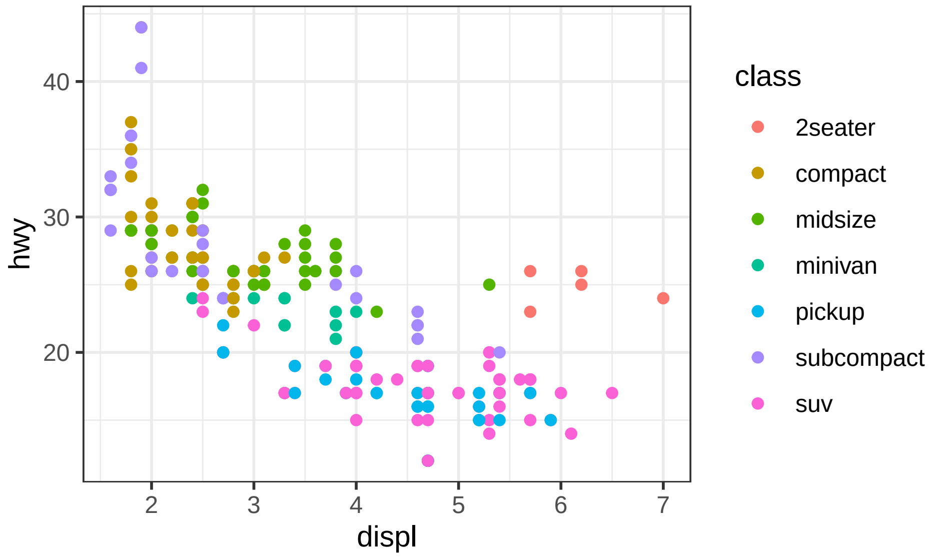
```
# there are points overlapping
#
# spread out the points
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy),position = "jitter")
```



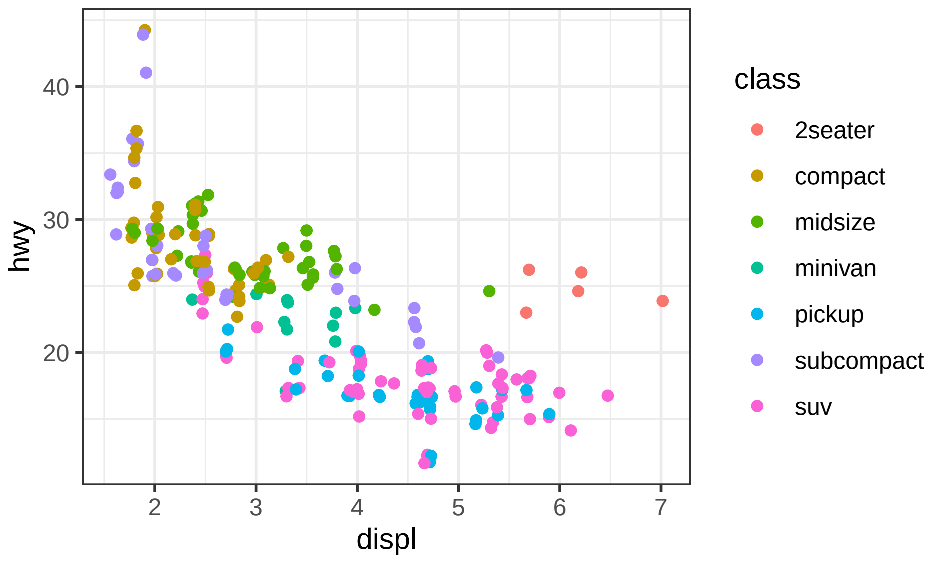
```
# change background -themes
theme_set(theme_bw())
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy))
```



```r
#
# scatterplot with categories by color
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, color = class))
```

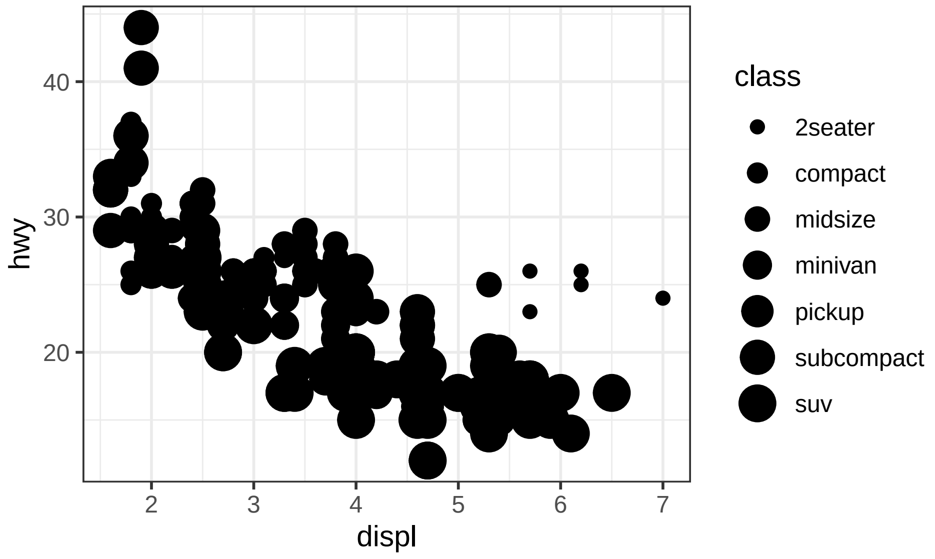


```r
#
# there are cars, from different classes, overlapping
#
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, color = class),
                               position='jitter')
```



```
#
# scatterplot with categories by size
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, size = class))
```

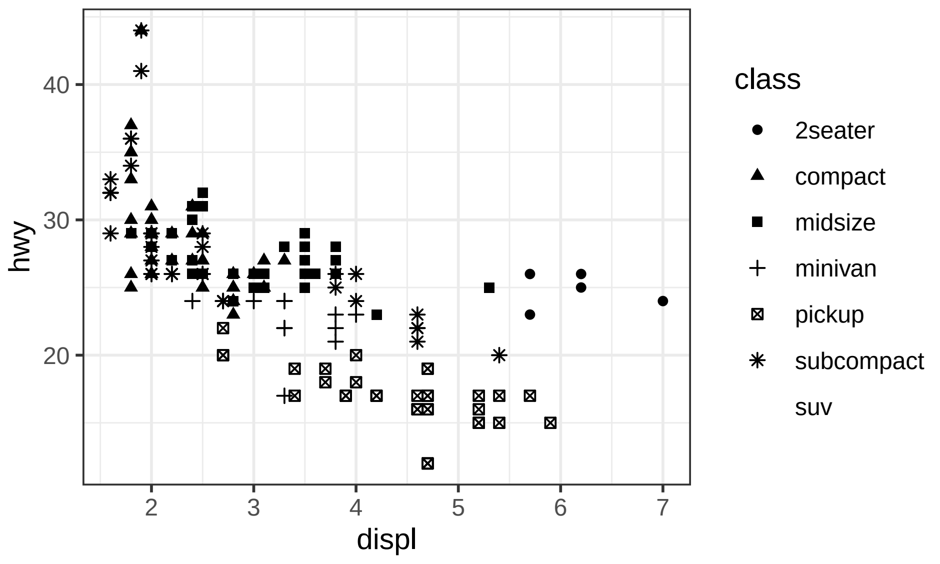
```
## Warning: Using size for a discrete variable is not advised.
```



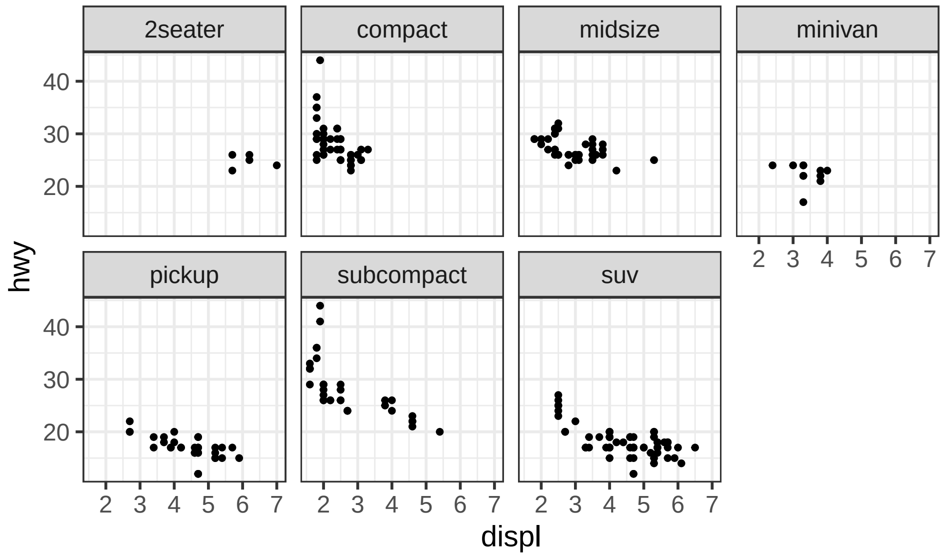
```
#
# different sizes identify different class
#
# scatterplot with categories by shape
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, shape = class))
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 7. Consider
## specifying shapes manually if you must have them.
```

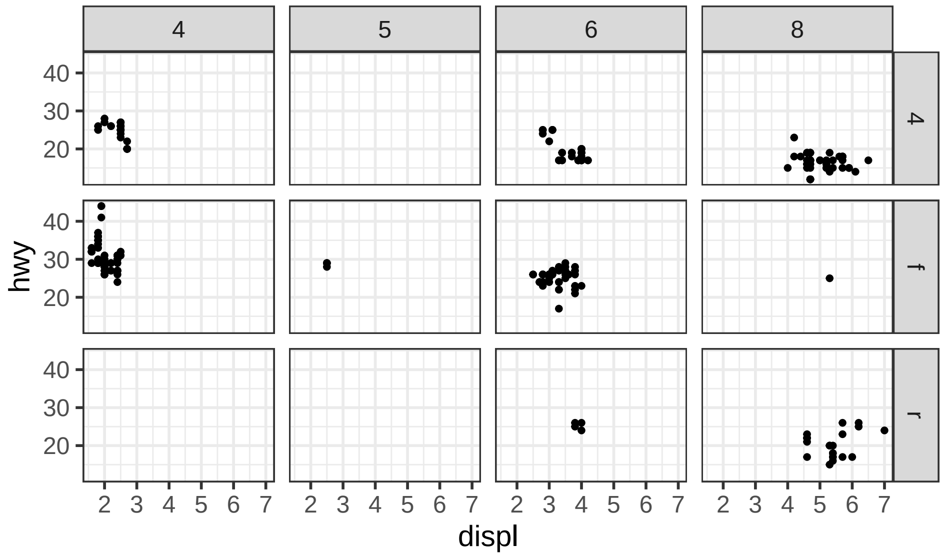
```
## Warning: Removed 62 rows containing missing values (geom_point).
```



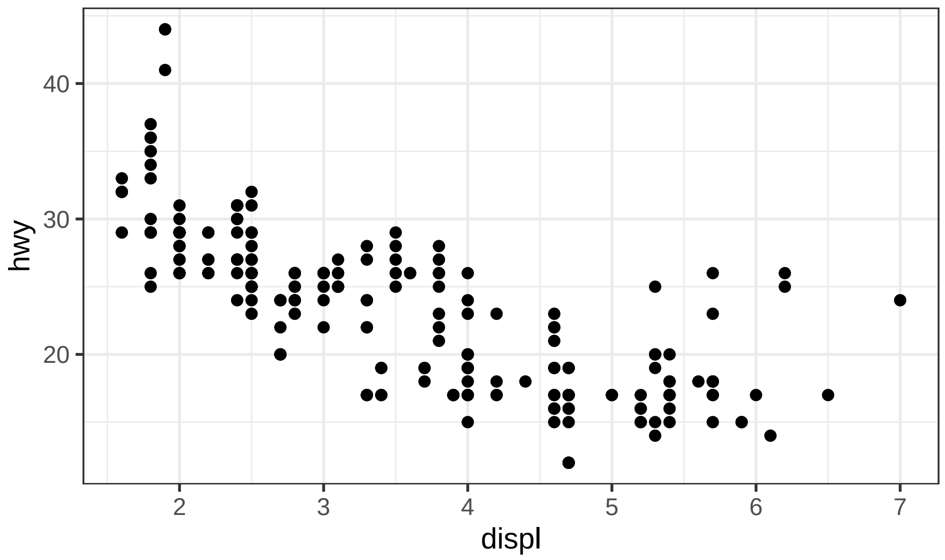
```
#
# SUVs were removed from plot

#
# FACETS
#
# split plot into subplots called facets
#
# scatterplots by one categorical var
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy), size = 0.7) +
  facet_wrap(~ class, nrow = 2)
```



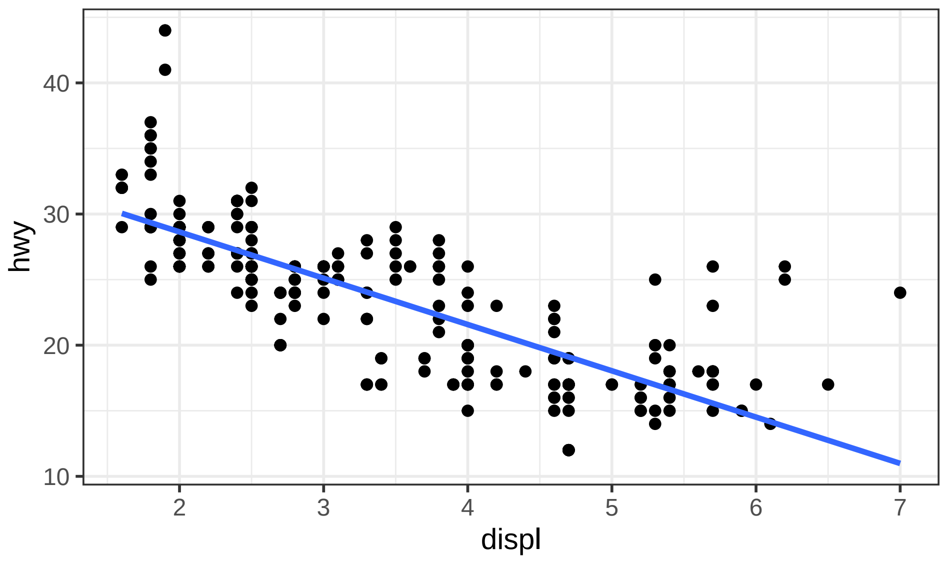
```
#
# scatterplots by two categorical var
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy), size = 0.7) +
  facet_grid(drv ~ cyl)
```



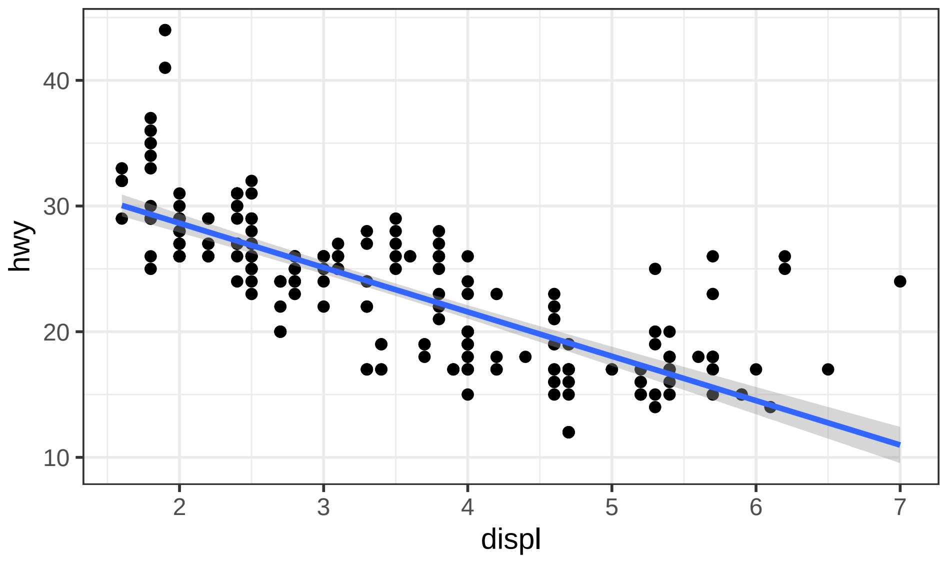
```
# scatterplot
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy))
```



```
#
# scatterplot with fitted line
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  geom_smooth(mapping = aes(x = displ, y = hwy),method = 'lm', se=F)
```

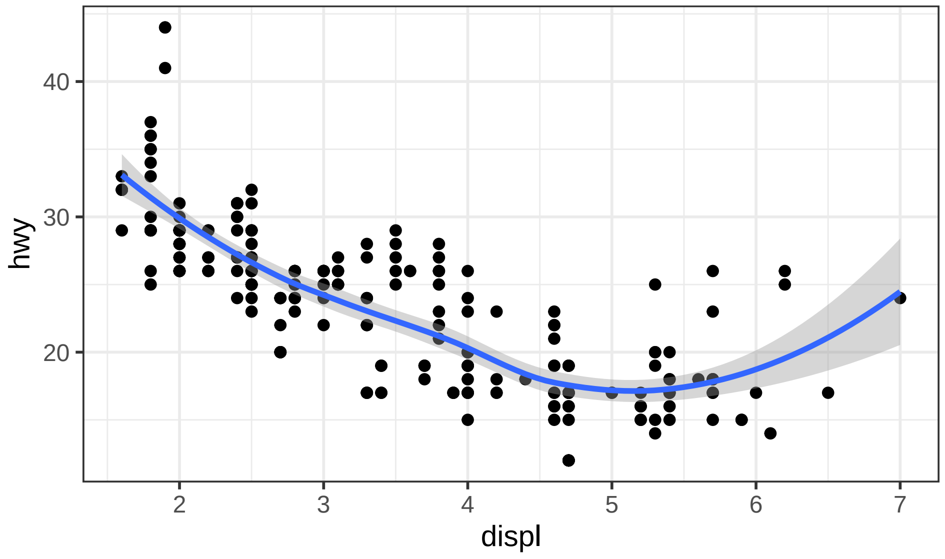


```r
#
# scatterplot with fitted line and 95% CI on the mean
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  geom_smooth(mapping = aes(x = displ, y = hwy),method = 'lm')
```



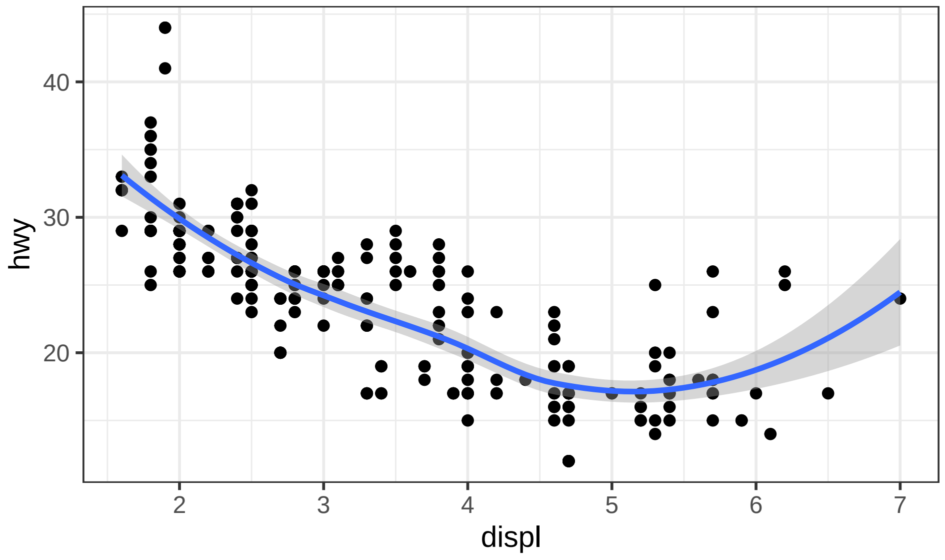
```r
#
# scatterplot with curve fitting
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  geom_smooth(mapping = aes(x = displ, y = hwy))
```

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



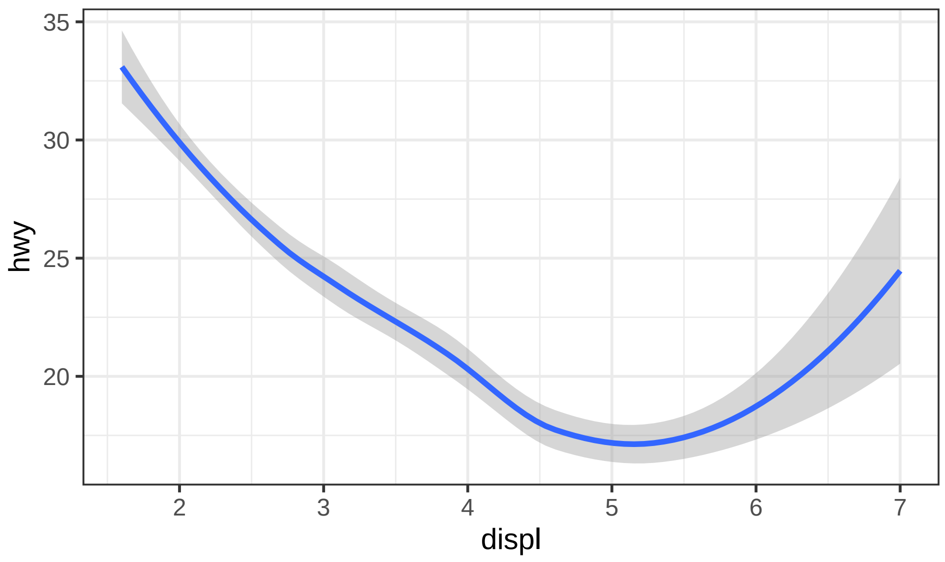
```
#
# or use global mapping
ggplot(data = mpg,mapping = aes(x = displ, y = hwy)) +
  geom_point() +
  geom_smooth()
```

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



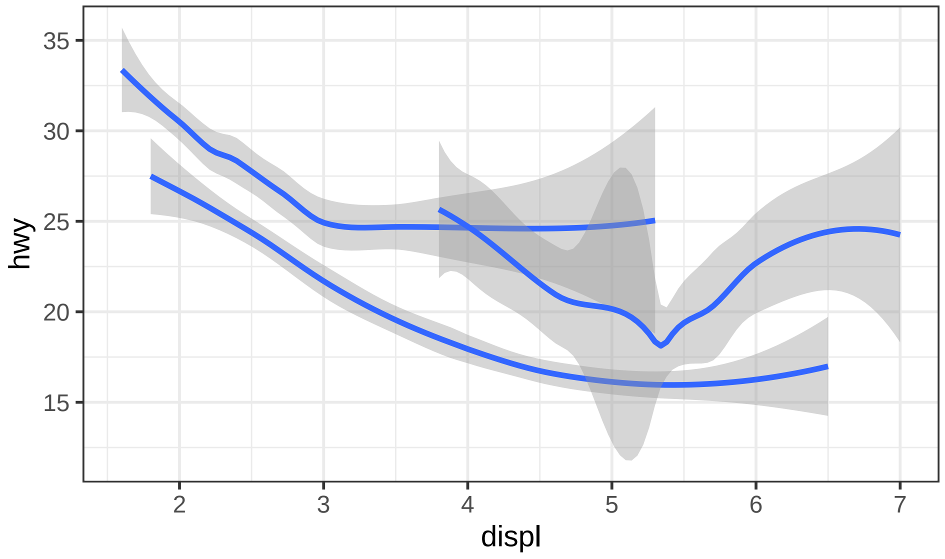
```
#
# placing mapping inside a geom() function makes them local mappings
#
# curve fitting alone
ggplot(data = mpg) + geom_smooth(mapping = aes(x = displ, y = hwy))
```

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



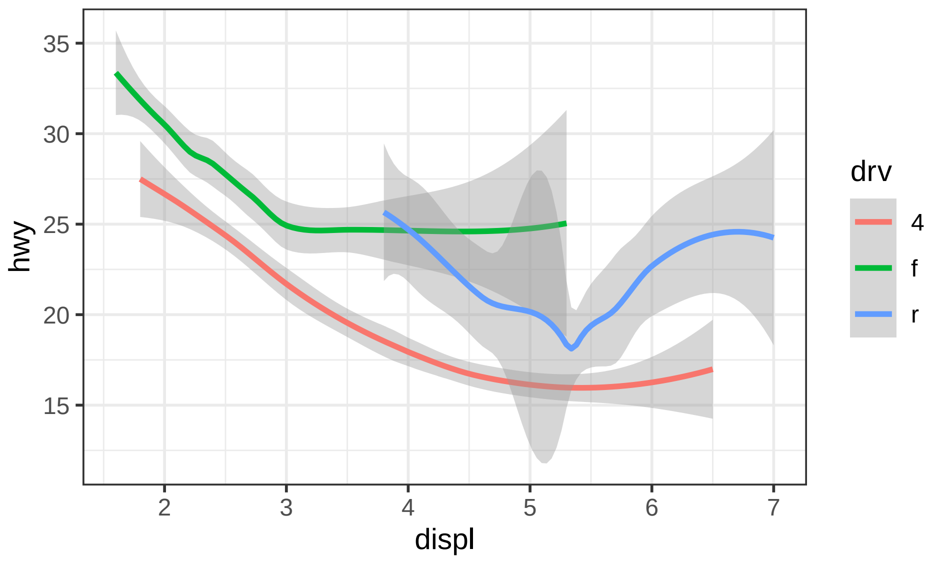
```r
#
# curves by DriveTrain
ggplot(data = mpg) + geom_smooth(mapping = aes(x = displ, y = hwy, group = drv))
```

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



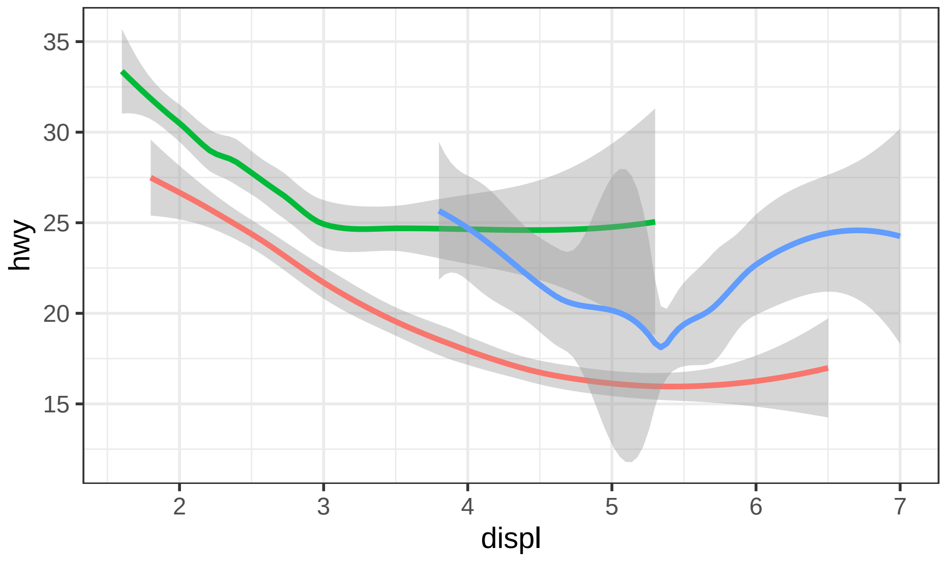
```r
# curves colored by DriveTrain
ggplot(data = mpg) + geom_smooth(mapping = aes(x = displ, y = hwy, color = drv))
```

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



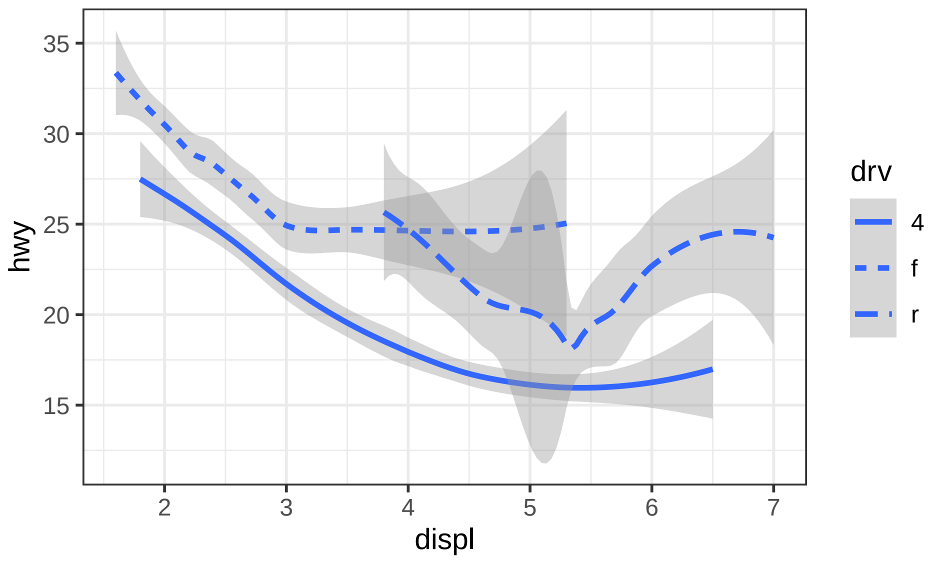
```
#
# color adds a legend
#
# curves colored by DriveTrain -no legend
ggplot(data = mpg) + geom_smooth(mapping = aes(x = displ, y = hwy, color = drv), show.legend=F)
```

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



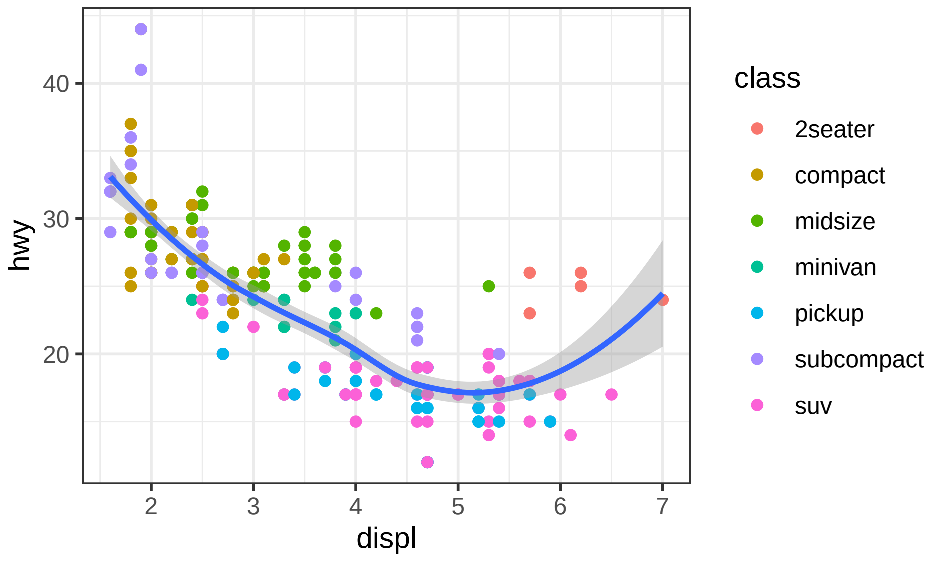
```
# curve line types by DriveTrain
ggplot(data = mpg) + geom_smooth(mapping = aes(x = displ, y = hwy, linetype = drv))
```

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



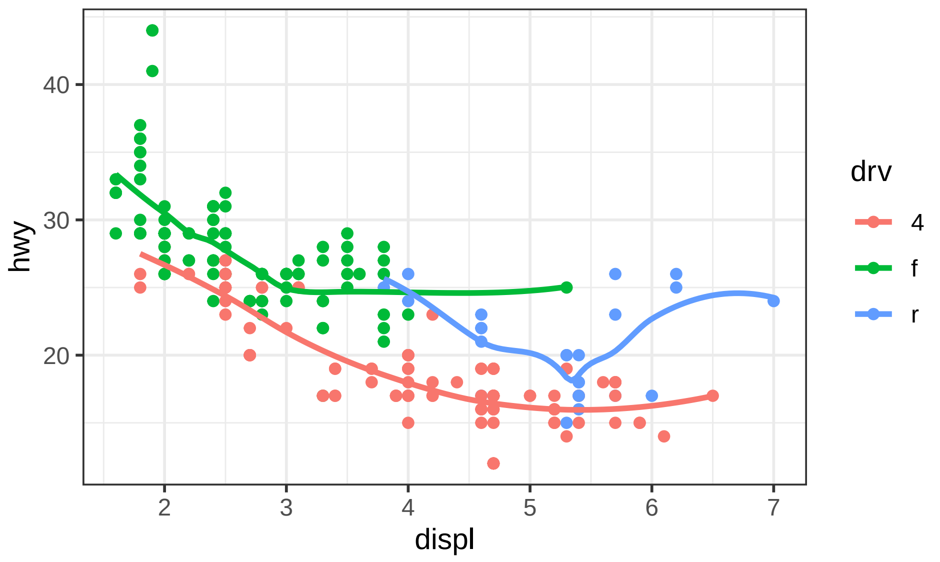
```r
#
# linetype adds a legend
#
# scatterplot by categories + with curve
ggplot(mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point(mapping = aes(color = class)) +
  geom_smooth()
```

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

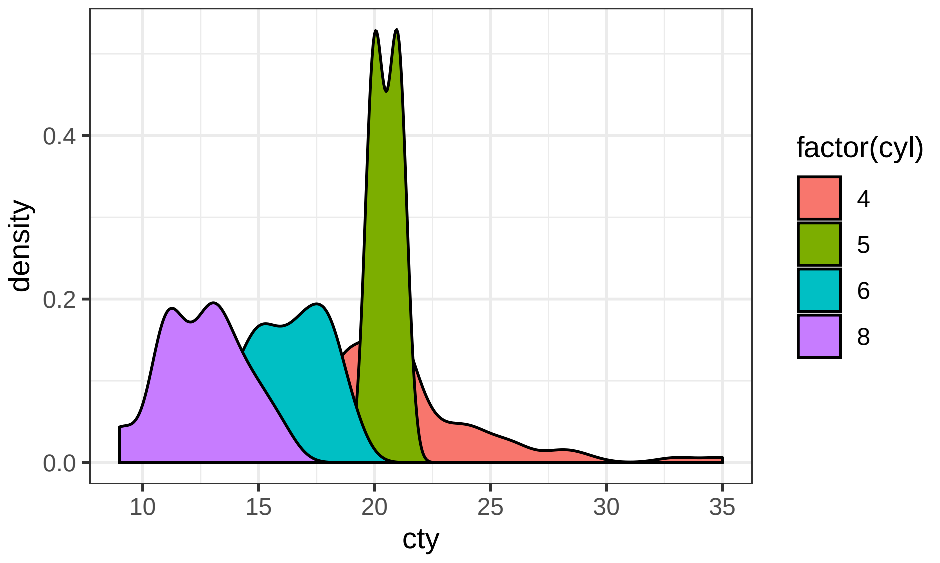


```r
# scatterplot and curves by categories
ggplot(mpg,mapping = aes(x = displ, y = hwy, color = drv))+
  geom_point() + geom_smooth(se = FALSE)
```

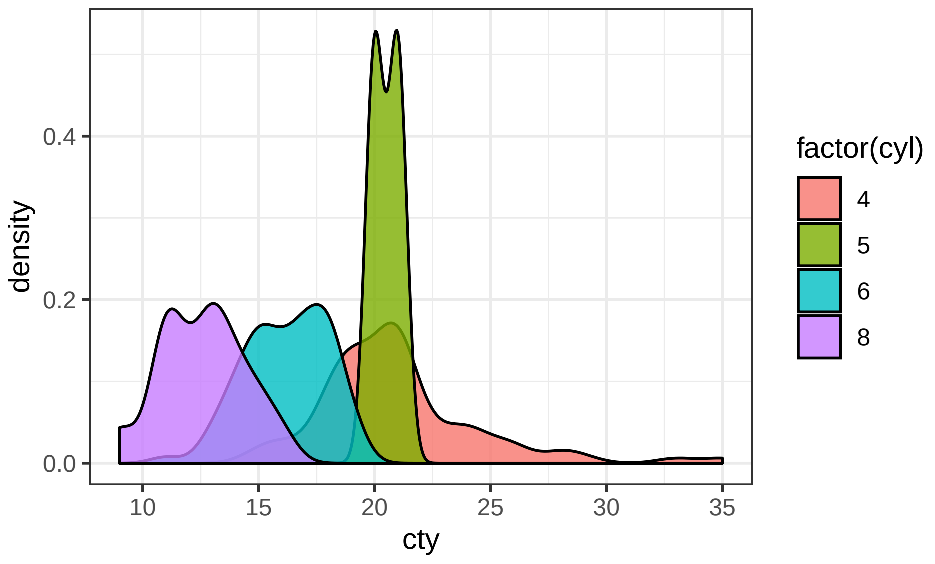
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



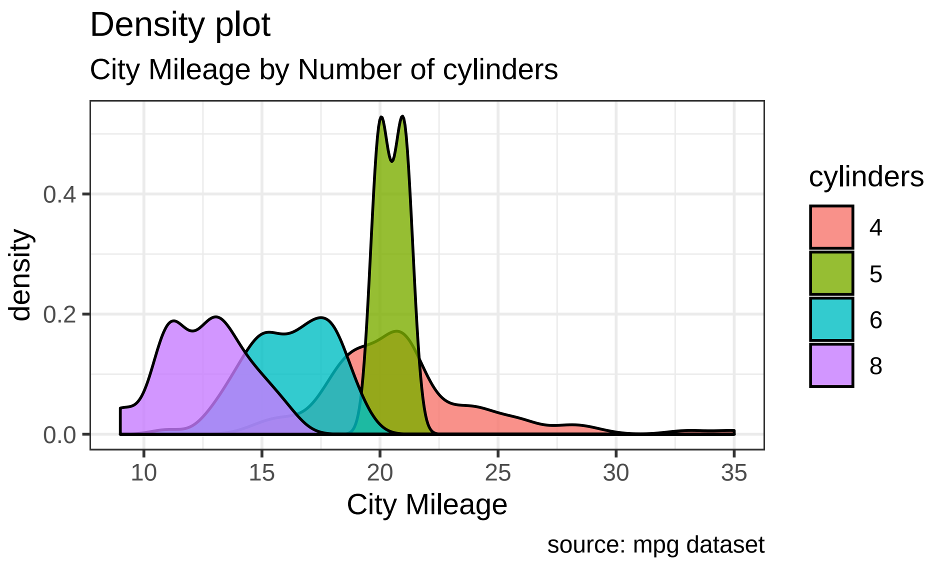
```
#
# Densities plot
ggplot(mpg,aes(cty)) + geom_density(aes(fill=factor(cyl)))
```



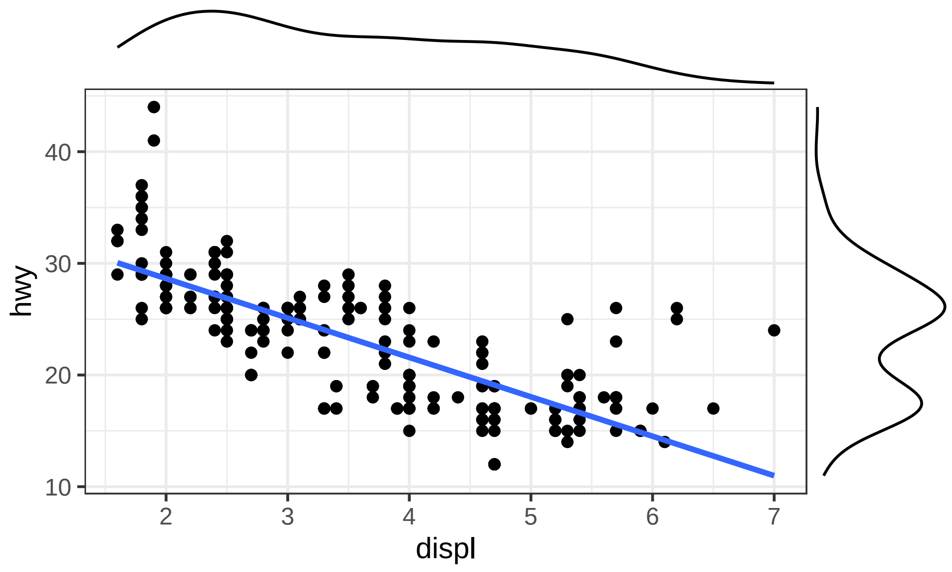
```
# transparent and labeling
g1 = ggplot(mpg,aes(cty)) + geom_density(aes(fill=factor(cyl)),alpha=0.8)
g1
```



```
g1 + labs(title = 'Density plot',
          subtitle = 'City Mileage by Number of cylinders',
          x = 'City Mileage',
          fill = 'cylinders',
          caption = 'source: mpg dataset')
```

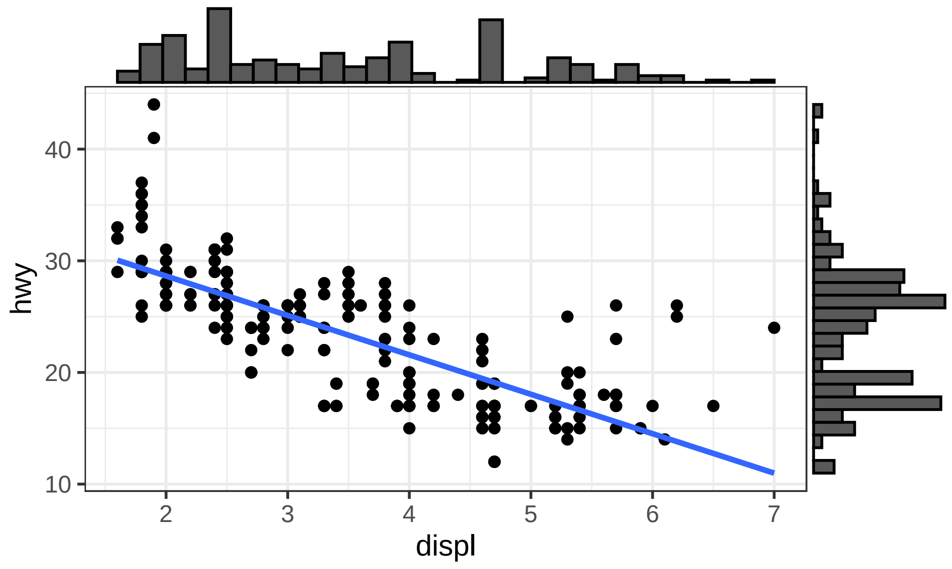


```
# Marginal plots
#
library(ggExtra)
#
g = ggplot(mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point() +
  geom_smooth(method='lm',se=FALSE)
ggMarginal(g,type = 'density')
```

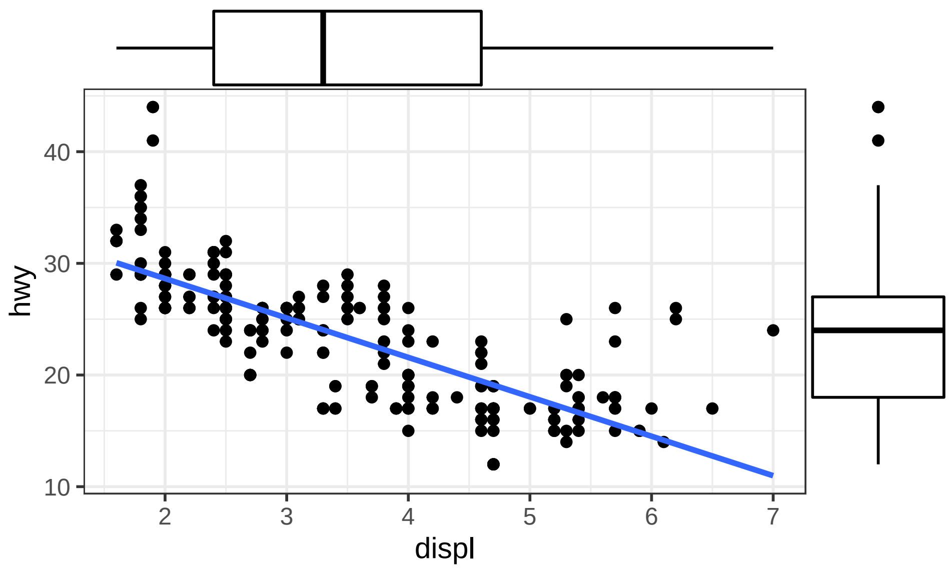


```
#
```

```
#
g = ggplot(mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point() +
  geom_smooth(method='lm',se=FALSE)
ggMarginal(g,type = 'histogram')
```



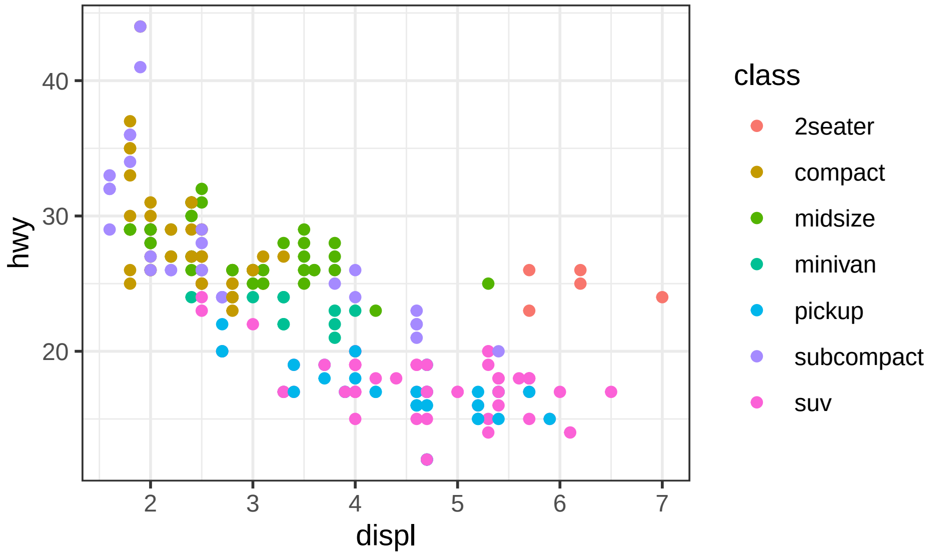
```
#
g = ggplot(mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point() +
  geom_smooth(method='lm',se=FALSE)
ggMarginal(g,type = 'boxplot',fill = 'transparent')
```



```r
#
# Circling points
#
library(ggalt)
```

```
## Registered S3 methods overwritten by 'ggalt':
##   method                  from
##   grid.draw.absoluteGrob  ggplot2
##   grobHeight.absoluteGrob ggplot2
##   grobWidth.absoluteGrob  ggplot2
##   grobX.absoluteGrob      ggplot2
##   grobY.absoluteGrob      ggplot2
```

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



```r
#
# data to circle
library(dplyr)
```

```
##
## 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
```

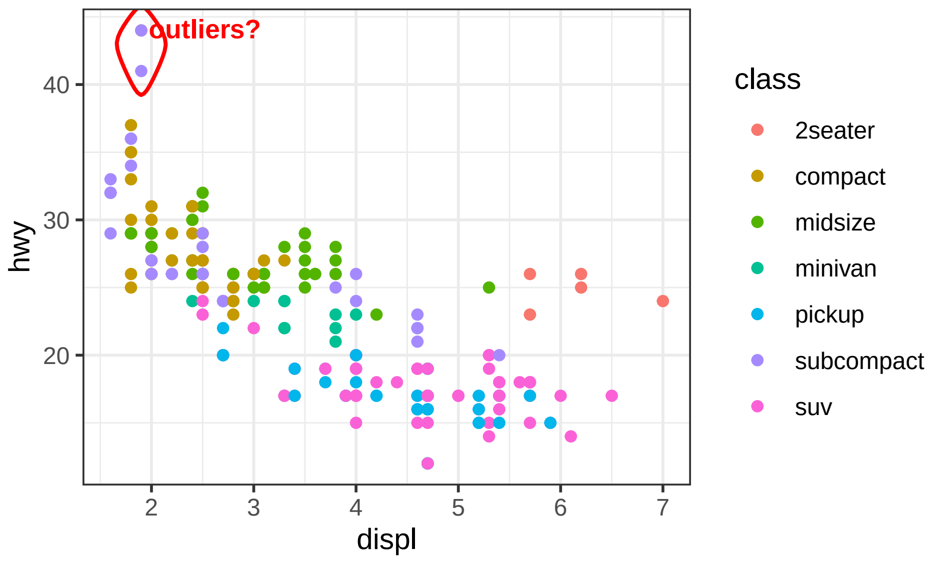
```r
names(mpg)
```

```
##  [1] "manufacturer" "model"        "displ"        "year"         "cyl"
##  [6] "trans"        "drv"          "cty"          "hwy"          "fl"
## [11] "class"
```

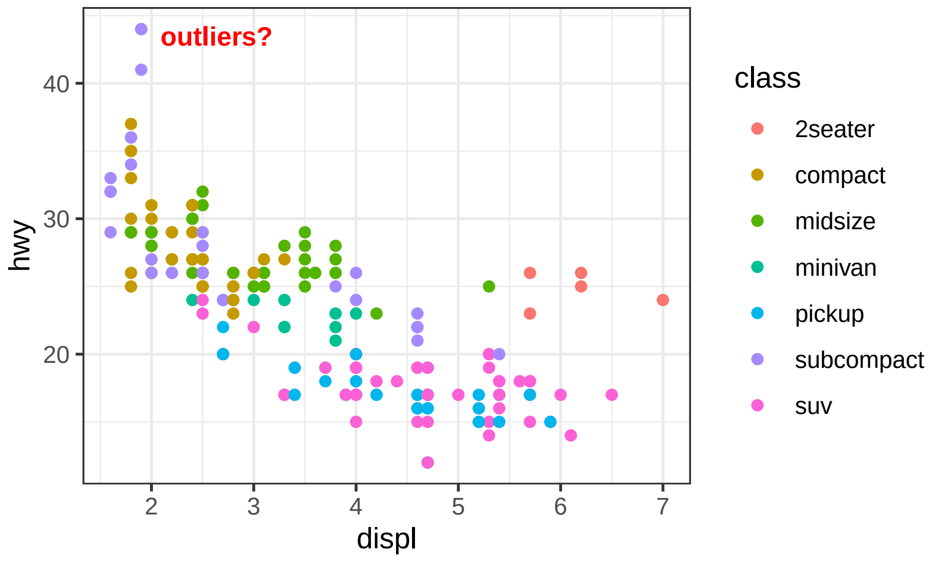
```r
mpg2 = select(mpg,displ,cty,hwy)
head(mpg2)
```

```
## # A tibble: 6 x 3
##   displ   cty   hwy
##   <dbl> <int> <int>
## 1   1.8    18    29
## 2   1.8    21    29
## 3   2      20    31
## 4   2      21    30
## 5   2.8    16    26
## 6   2.8    18    26
```

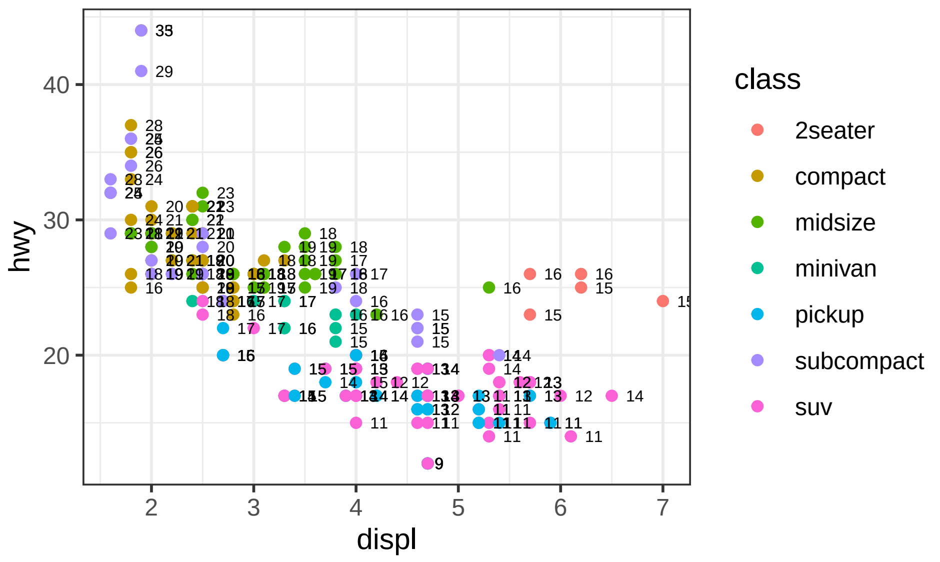
```r
mpg2 = mpg2[mpg2$hwy>40 & mpg2$displ<2,]
# plot
gg = ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, color = class))
gg + geom_encircle(aes(x = displ, y = hwy, color = class),data = mpg2, color='red',
                   size = 2,spread = 0.001)
#
# Annotations
#
library(grid)
#
gg = ggplot(mpg) + geom_point(mapping = aes(x = displ, y = hwy, color = class))
text1 = 'outliers?'
specs1 = grid.text(text1,x = 0.22,y=0.94,
                   gp = gpar(fontsize=10,fontface='bold',col='red'))
```



```
gg + annotation_custom(specs1)
```



```
#
# point labeling
gg = ggplot(mpg) + geom_point(mapping = aes(x = displ, y = hwy, color = class))
gg + geom_text(aes(x = displ, y = hwy,label=cty), size=2.2,hjust=-0.8)
```



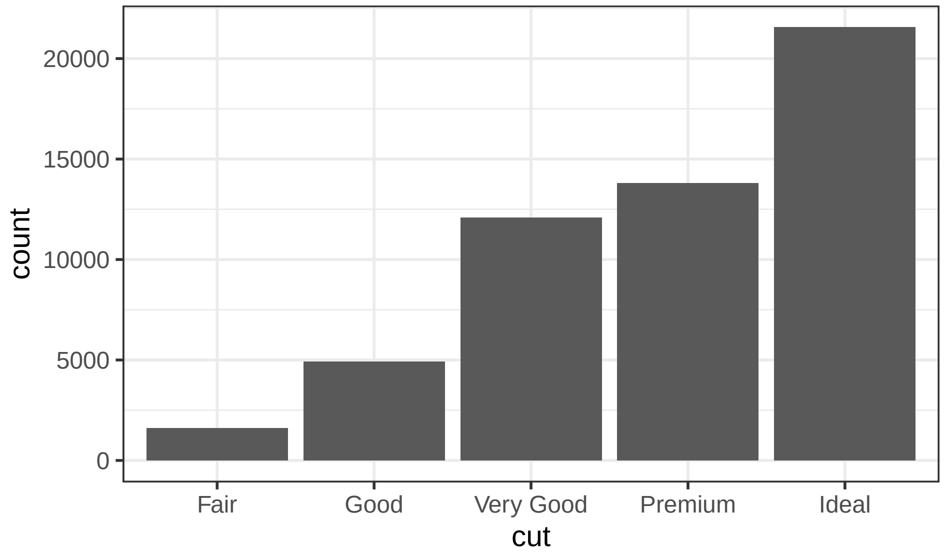
```
#

# BARPLOTS
#
# use ggplot2::diamonds dataset
head(diamonds)
```

```
## # A tibble: 6 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1 0.23  Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
## 2 0.21  Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
## 3 0.23  Good      E     VS1      56.9    65   327  4.05  4.07  2.31
## 4 0.290 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
## 5 0.31  Good      J     SI2      63.3    58   335  4.34  4.35  2.75
## 6 0.24  Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

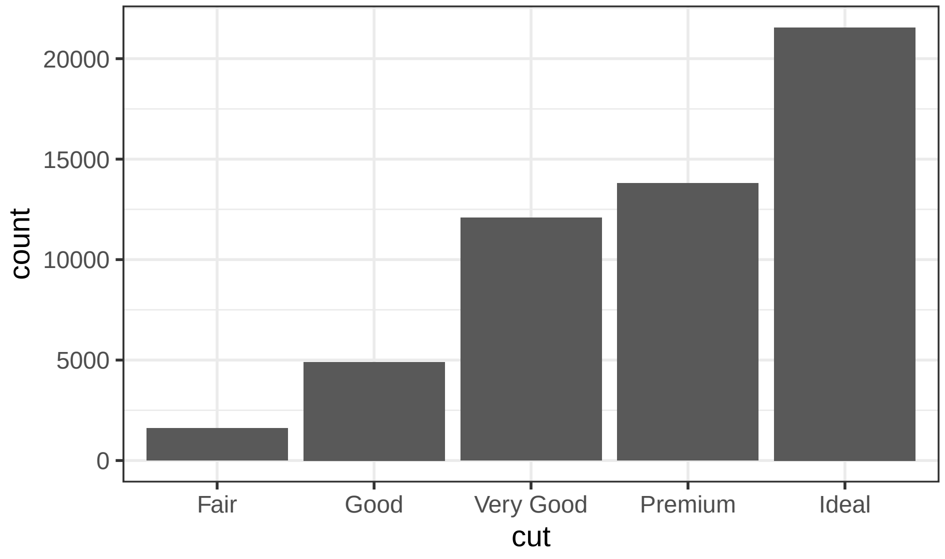
```
# online description
?diamonds

table(diamonds$cut)
```

```
##
##      Fair      Good Very Good   Premium     Ideal
##      1610      4906     12082     13791     21551
```

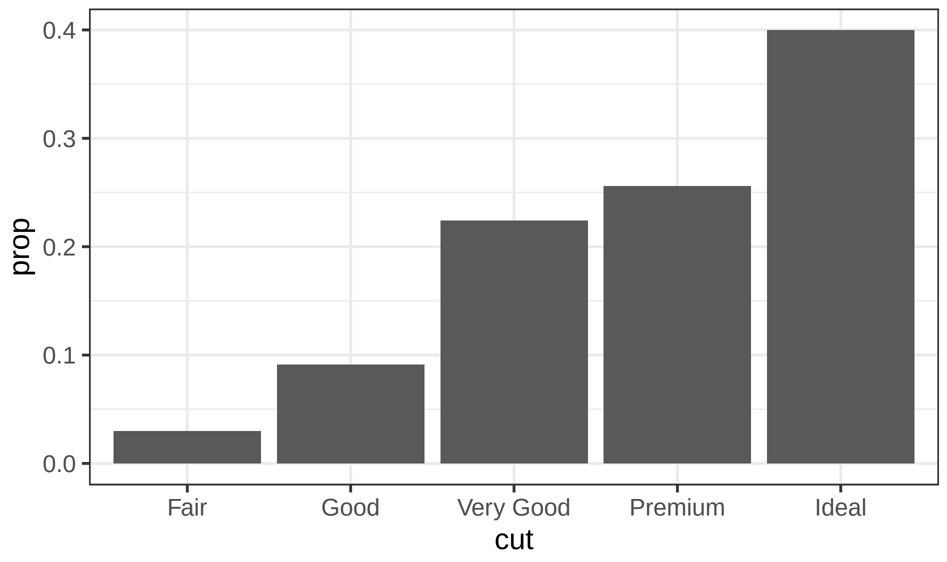
```
# counts
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut))
```



```
#
# or
ggplot(data = diamonds) + stat_count(mapping = aes(x = cut))
```



```
#
# proportions accross all cut categories (group = 1)
ggplot(data = diamonds) +  geom_bar(mapping = aes(x = cut, y = stat(prop), group = 1))
```

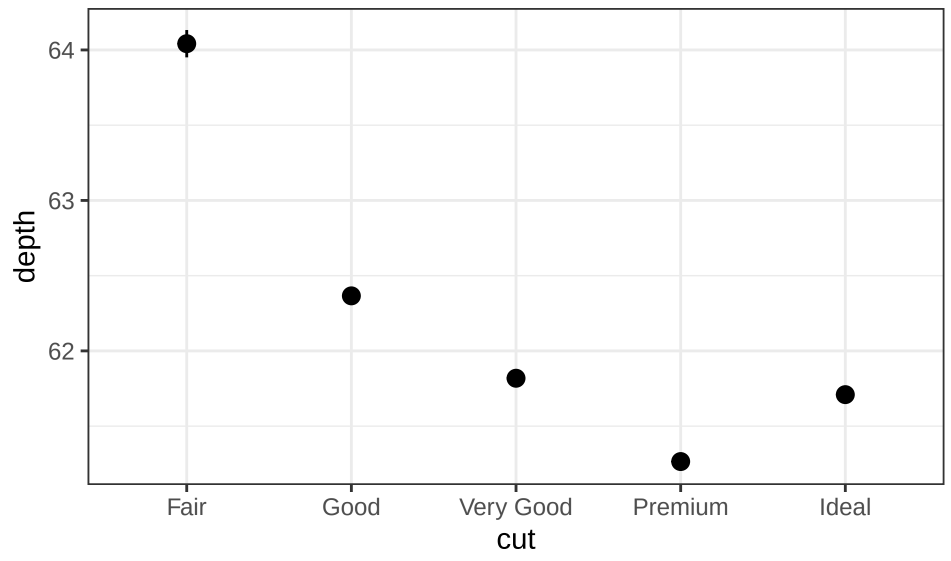


```r
# average depth by cut
tapply(diamonds$depth,diamonds$cut,mean)
```

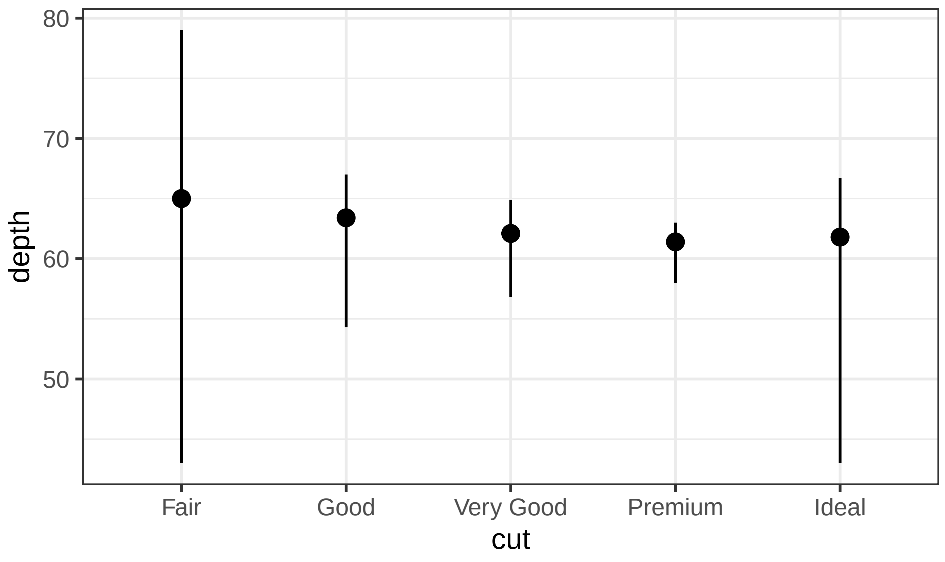
```
##      Fair      Good Very Good   Premium     Ideal
##  64.04168  62.36588  61.81828  61.26467  61.70940
```

```r
# points
ggplot(data = diamonds) +
  stat_summary(mapping = aes(x = cut, y = depth))
```

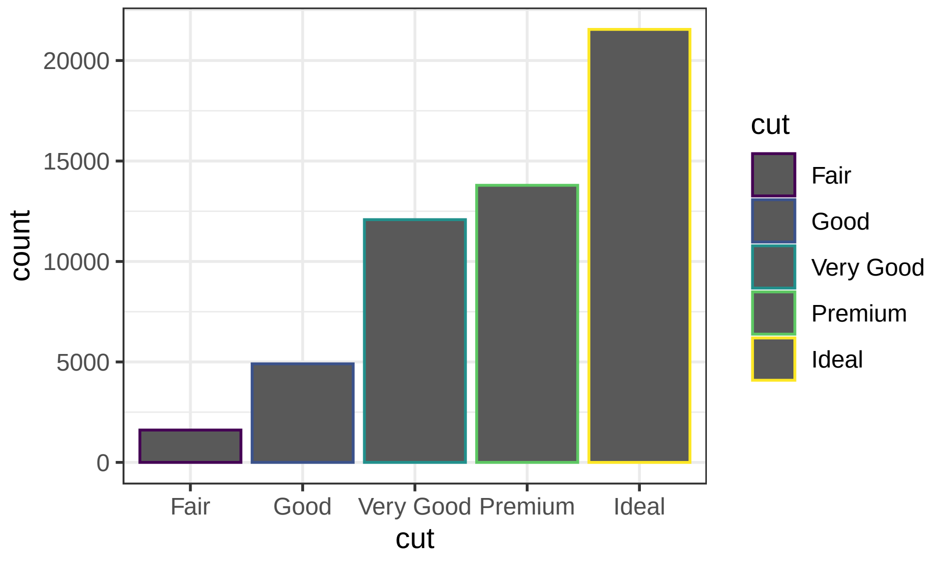
```
## No summary function supplied, defaulting to `mean_se()
```



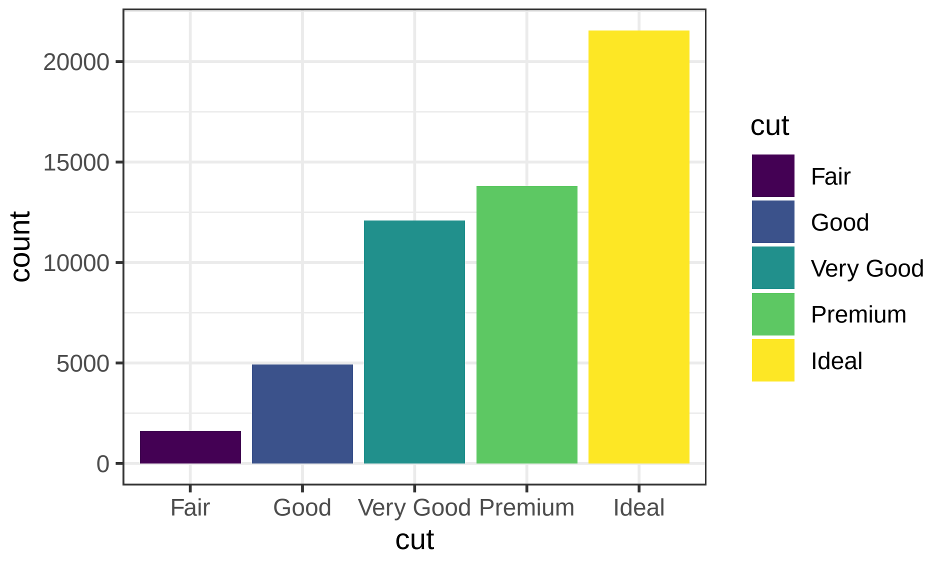
```r
# points + vertical lines
ggplot(data = diamonds) +
  stat_summary(mapping = aes(x = cut, y = depth),
               fun.ymin = min,
               fun.ymax = max,
               fun.y = median)
```



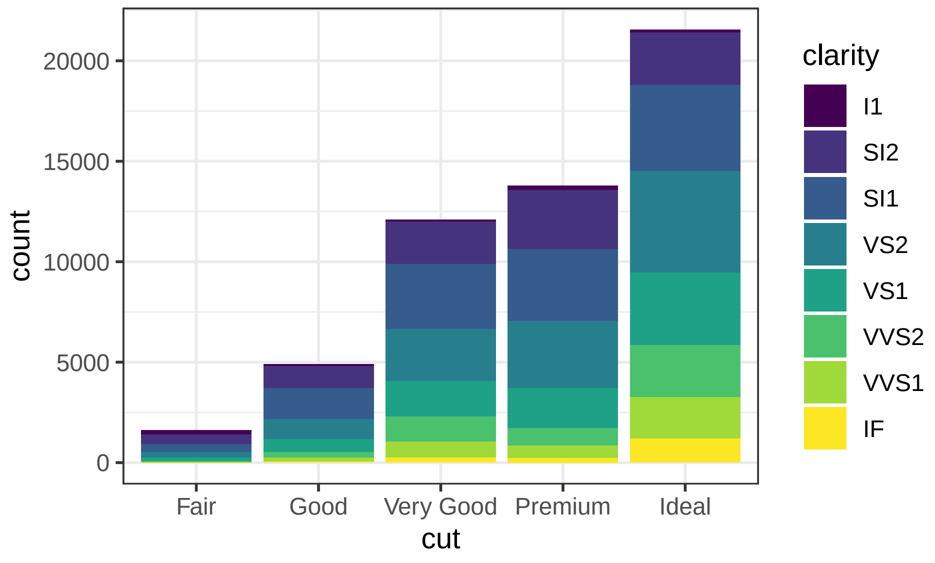
```
# barplots with categories
#
# color given by cut
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut, color = cut))
```



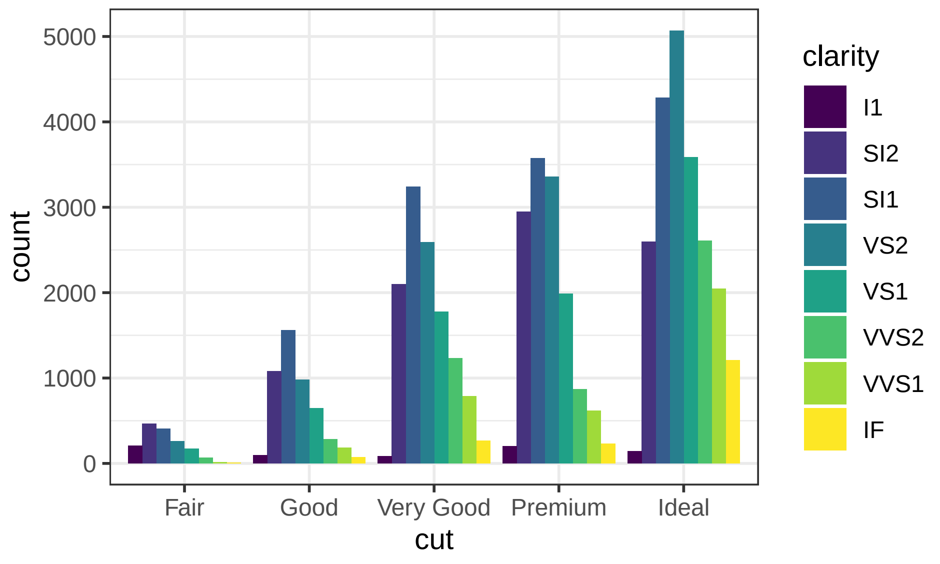
```
#
# not good
#
# use fill
#
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut, fill = cut))
```



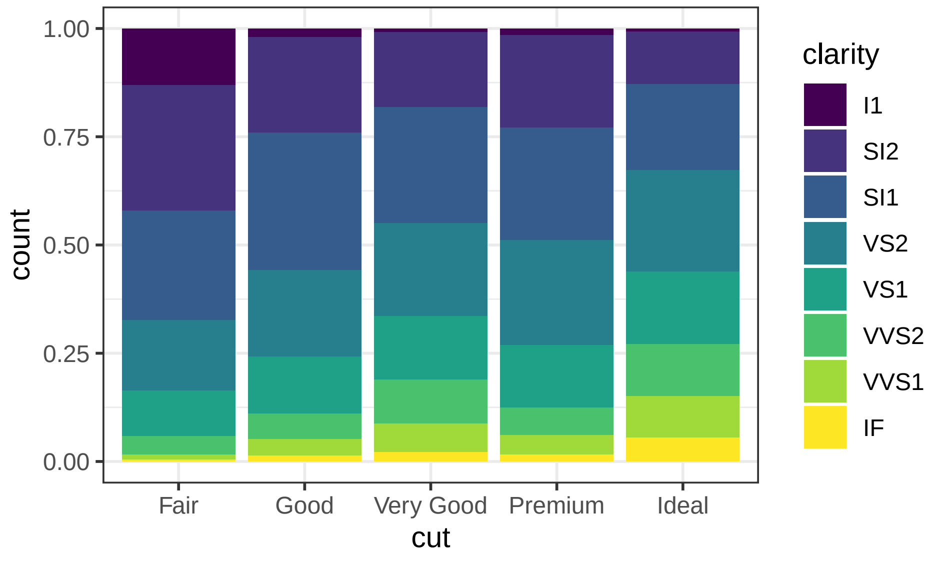
```
#
# legend not needed, actually
#

# second categorical variable clarity (for filling the bars)
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut, fill = clarity))
```



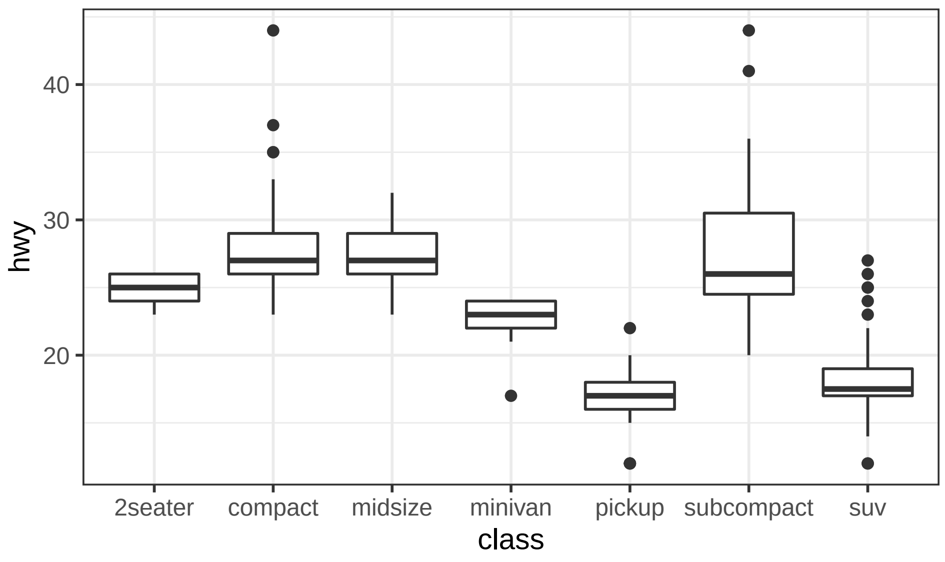
```
#
# counts by clarity category side-by-side
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut, fill = clarity),position = "dodge")
```



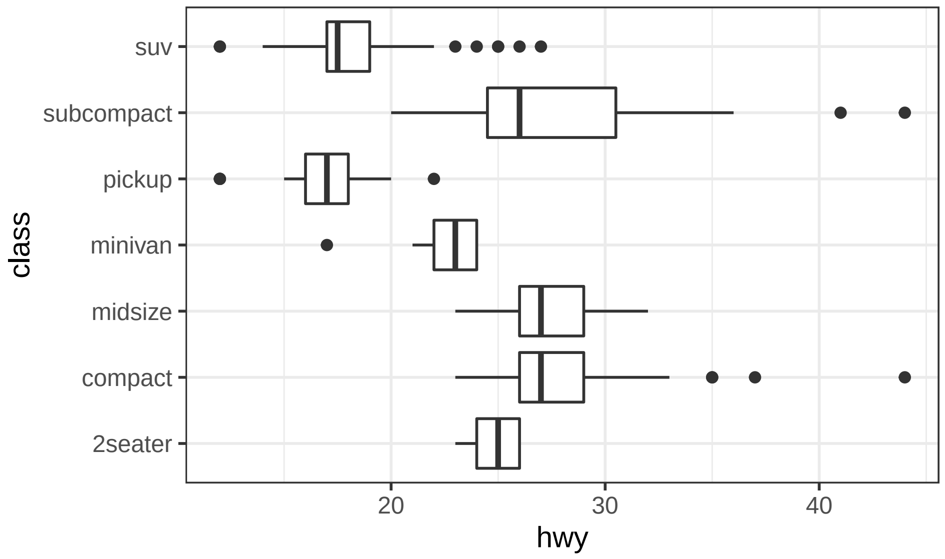
```
#
# proportions across clarity categories
ggplot(data = diamonds) + geom_bar(mapping = aes(x = cut, fill = clarity),position = "fill")
```



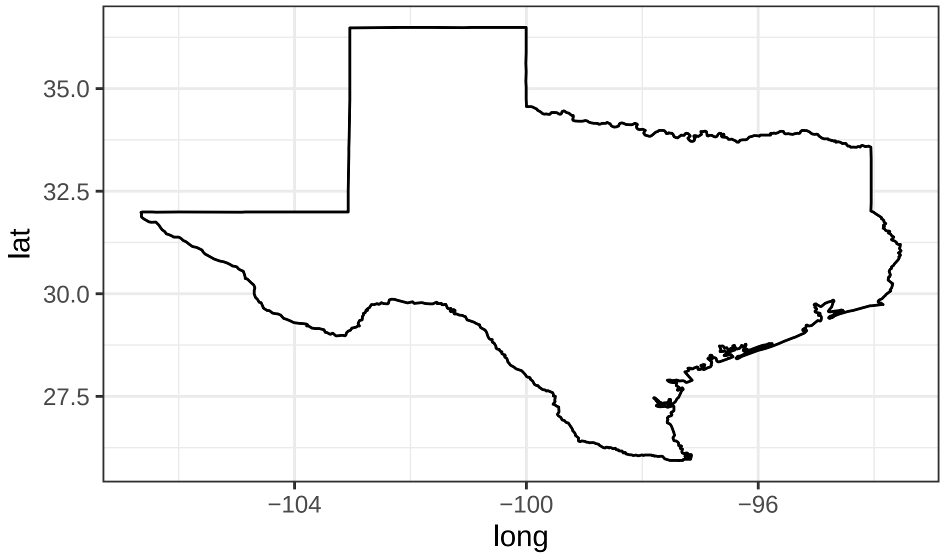
```
#
# COORDINATE SYSTEMS
#
# use mpg data
#
# vertical boxplots
ggplot(mpg, mapping = aes(x = class, y = hwy)) + geom_boxplot()
```



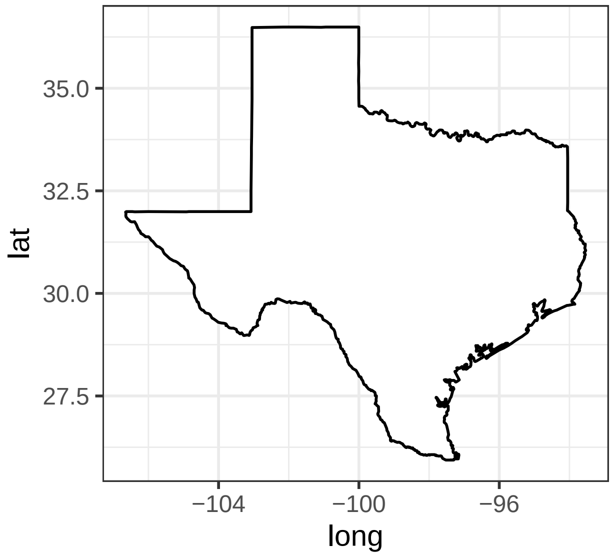
```
#
# coord_flip() switches x and y axes
ggplot(mpg, mapping = aes(x = class, y = hwy)) + geom_boxplot() + coord_flip()
```



```
#
# MAPS
#
# install.packages("maps")
library(maps)
library(help=maps)
#
tx <- map_data("state","texas")
#
ggplot(tx, aes(long, lat, group = group)) + geom_polygon(fill = "white", color = "black")
```



```
#
# set aspect ratio
ggplot(tx, aes(long, lat, group = group)) + geom_polygon(fill = "white", color = "black") +
  coord_quickmap()
```



```
#
# map with no coordinates
tx <- map("state","texas")
```