

WhyR

M_Raess

November 13, 2016

1. Why R - packages to be used

```
library(dplyr)
library(ggplot2)
library(scales)
library(ggiraph)
library(RColorBrewer)
library(RCurl)
library(findviews)
library(yarrrr)
```

2. How R structures data - the data frame

Working with built-in data sets

```
data("mtcars")
```

```
head(mtcars) # gives us the first six rows alt. head(mtcars, 10)
```

```
##           mpg   cyl  disp    hp  drat    wt   qsec    vs  am  gear  carb
## Mazda RX4      21.0   6  160  110  3.90  2.620  16.46   0   1    4    4
## Mazda RX4 Wag  21.0   6  160  110  3.90  2.875  17.02   0   1    4    4
## Datsun 710      22.8   4  108   93  3.85  2.320  18.61   1   1    4    1
## Hornet 4 Drive  21.4   6  258  110  3.08  3.215  19.44   1   0    3    1
## Hornet Sportabout 18.7   8  360  175  3.15  3.440  17.02   0   0    3    2
## Valiant        18.1   6  225  105  2.76  3.460  20.22   1   0    3    1
```

```
summary(mtcars) # summary stats
```

```
##           mpg           cyl           disp           hp
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
##           drat           wt           qsec           vs
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000
##           am           gear           carb
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000
```

```
## 1st Qu.:0.0000    1st Qu.:3.000    1st Qu.:2.000
## Median :0.0000    Median :4.000    Median :2.000
## Mean   :0.4062    Mean   :3.688    Mean   :2.812
## 3rd Qu.:1.0000    3rd Qu.:4.000    3rd Qu.:4.000
## Max.   :1.0000    Max.   :5.000    Max.   :8.000
```

```
str(mtcars) # structure of your data set
```

```
## 'data.frame':    32 obs. of  11 variables:
## $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : num   6  6  4  6  8  6  8  4  4  6 ...
## $ disp: num  160 160 108 258 360 ...
## $ hp  : num  110 110  93 110 175 105 245  62  95 123 ...
## $ drat: num   3.9  3.9  3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
## $ wt  : num   2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num  16.5 17 18.6 19.4 17 ...
## $ vs  : num   0  0  1  1  0  1  0  1  1  1 ...
## $ am  : num   1  1  1  0  0  0  0  0  0  0 ...
## $ gear: num   4  4  4  3  3  3  3  4  4  4 ...
## $ carb: num   4  4  1  1  2  1  4  2  2  4 ...
```

```
#findviews(mtcars) # gives you a great overview of categorical and continous variables
```

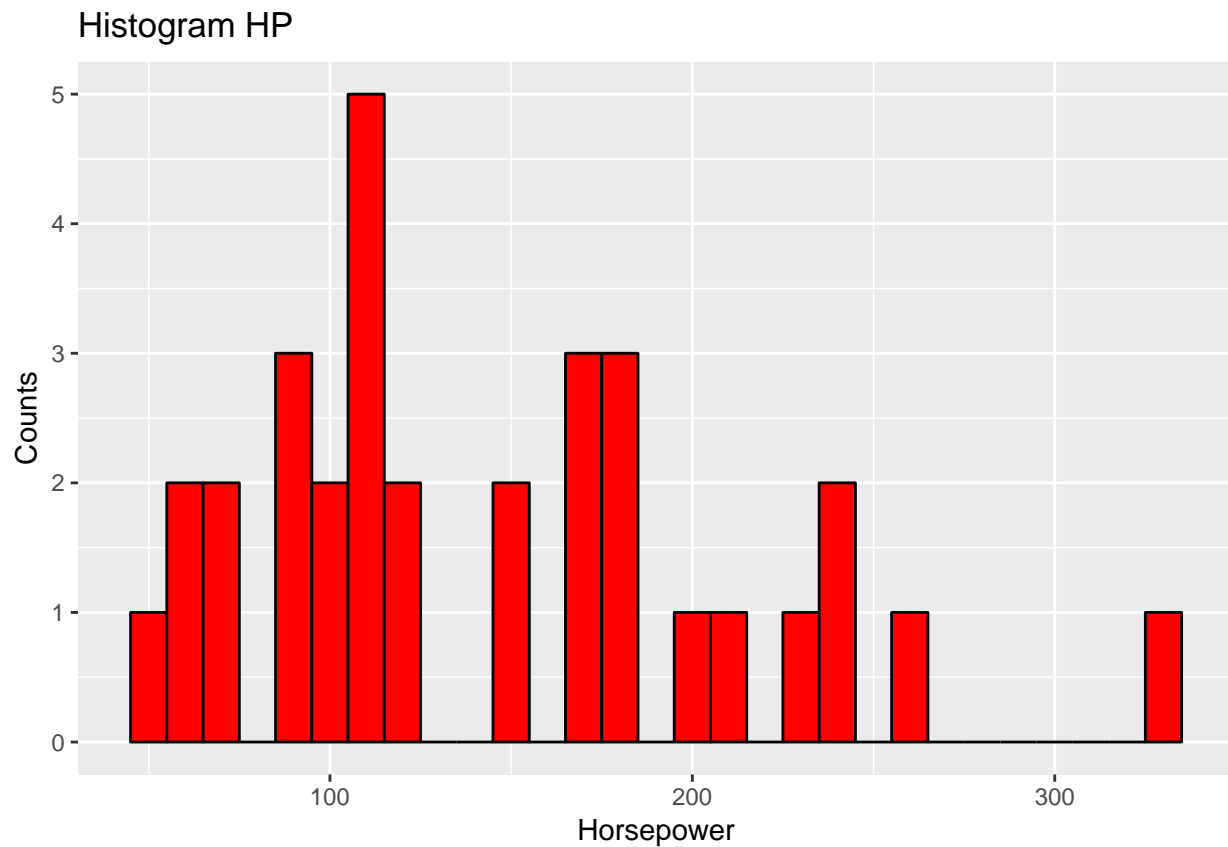
```
#View(mtcars) # brings up the data set
```

Now, that we have a general idea of how our data is structured, lets do some exploratory data analysis

This is what we are using the ggplot2 package for

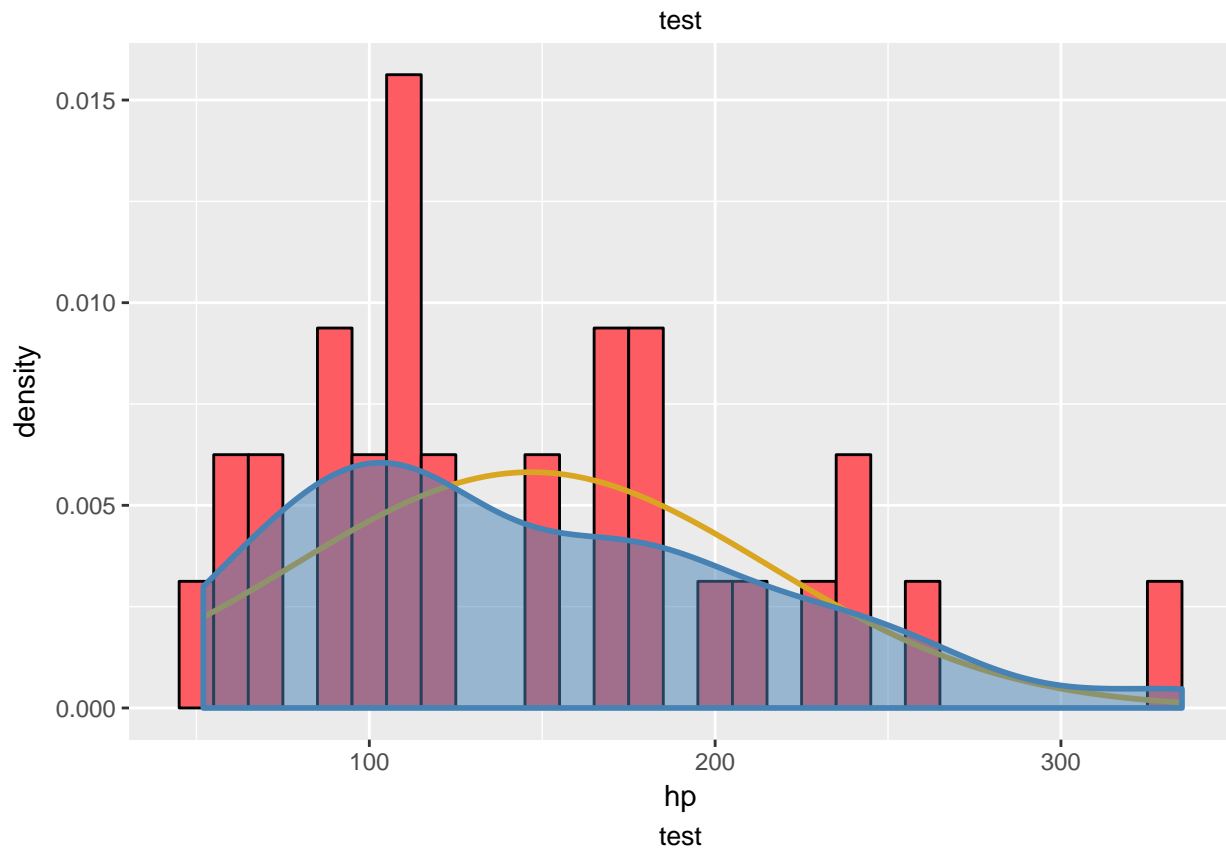
3. Histogram for quantitative variables

```
ggplot(mtcars, aes(hp)) +
  geom_histogram(binwidth = 10, fill = "red", col = "black") +
  ggtitle("Histogram HP") +
  xlab("Horsepower") +
  ylab("Counts")
```



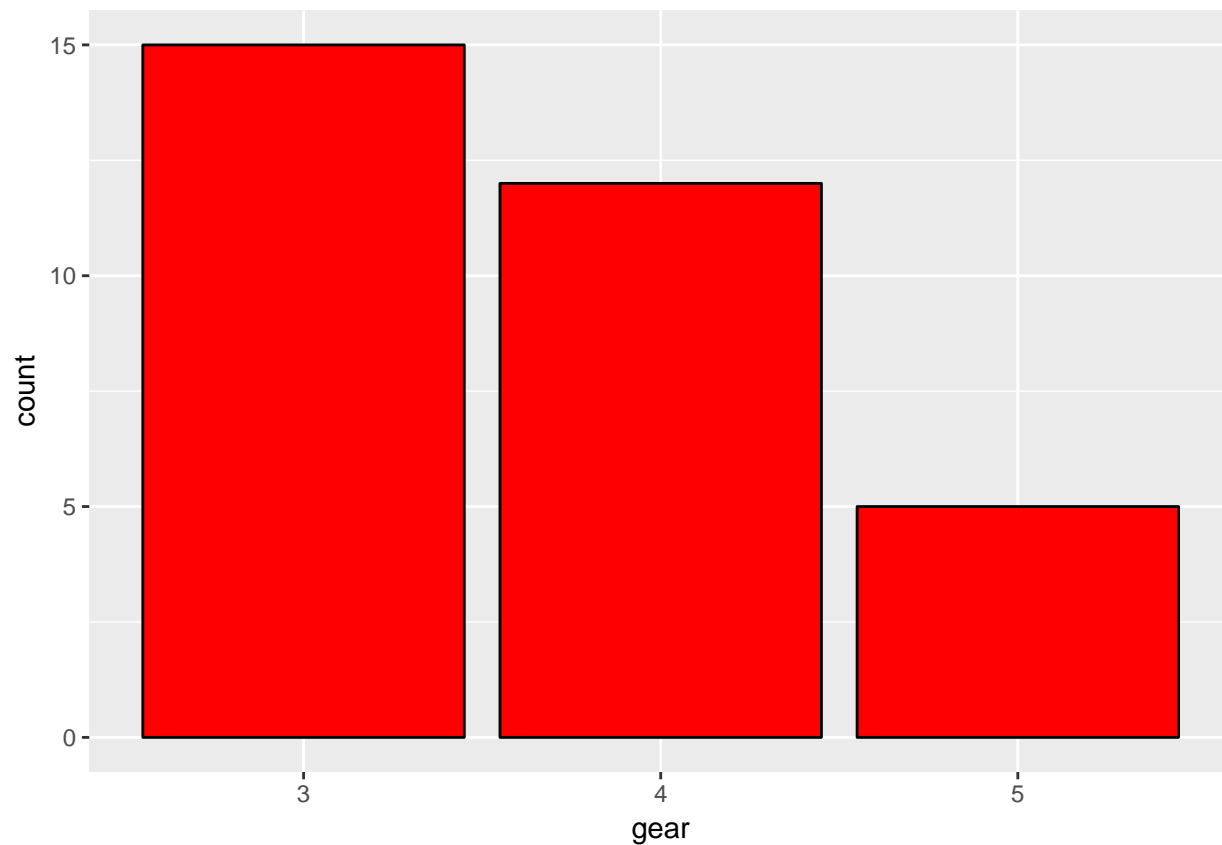
Alternatively (or additionally), we can overlay the density curve and the normal distribution

```
ggplot(mtcars, aes(hp)) +  
  geom_histogram(aes(y = ..density..), fill = "#fd5c63", binwidth = 10, col = "black") +  
  stat_function(fun = dnorm, color = "goldenrod", lwd = 1,  
               args=list(mean = mean(mtcars$hp), sd = sd(mtcars$hp))) +  
  geom_density(color = "steelblue", lwd = 1, fill = "steelblue", alpha = .5) +  
  labs(subtitle = "test", caption = 'test') +  
  theme(plot.caption = element_text(hjust=0.5), plot.subtitle = element_text(hjust = 0.5))
```



4. Barplot for categorical variables

```
mtcars2 <- mtcars  
mtcars2 <- mutate(mtcars, gear = as.factor(gear)) # we have to change gear from num to fact  
  
ggplot(mtcars2, aes(gear)) +  
  geom_bar(fill = "red", col = "black")
```

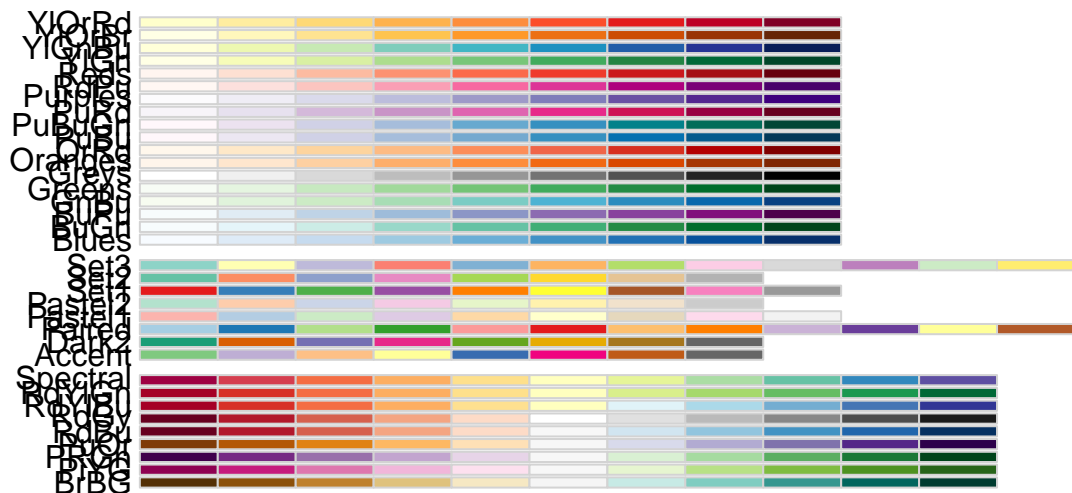


5. Working with colors - get brandcolors - woohoo - brandcolors.net (hex-code)

```
myColors <- c("#1da1f2", "#fd5c63", "#003a70")  
names(myColors) <- levels(mtcars2$gear)  
names(myColors) <- c("4", "3", "5") # change the level-colors according to order
```

Colors with RColorBrewer and yarr

```
display.brewer.all()
```

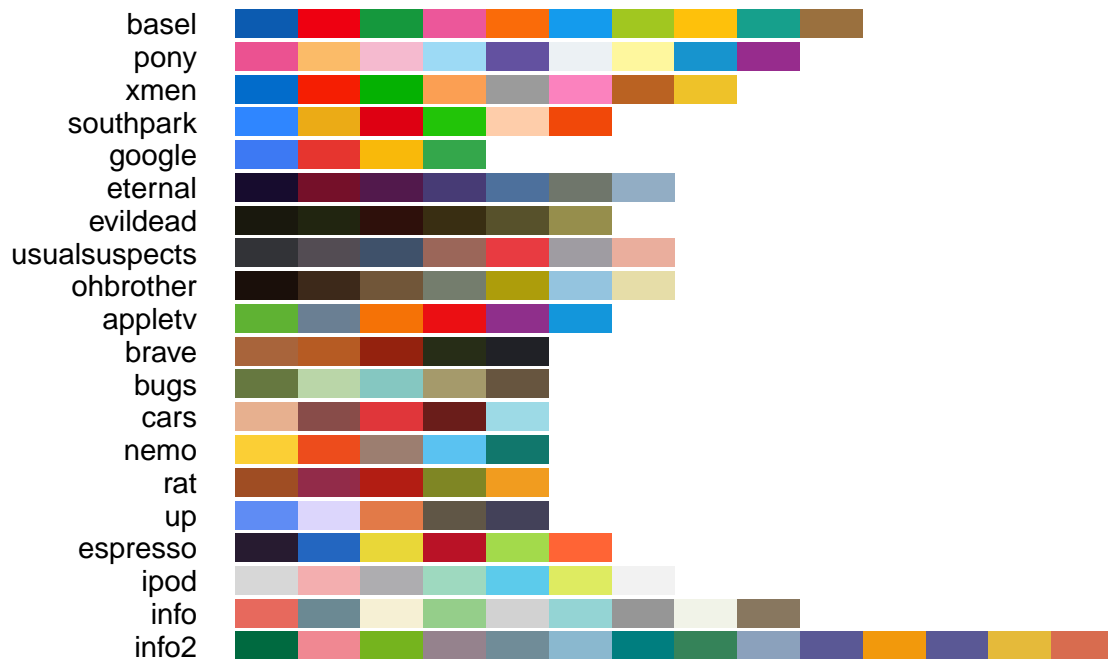


```
myColors2 <- brewer.pal(3, "Pastel1")
```

```
piratepal(palette = "all") # yarrrr
```

Here are all of the pirate palettes

Transparency is set to 0



```
myColors3 <- unname(piratepal(palette = "google"))
```

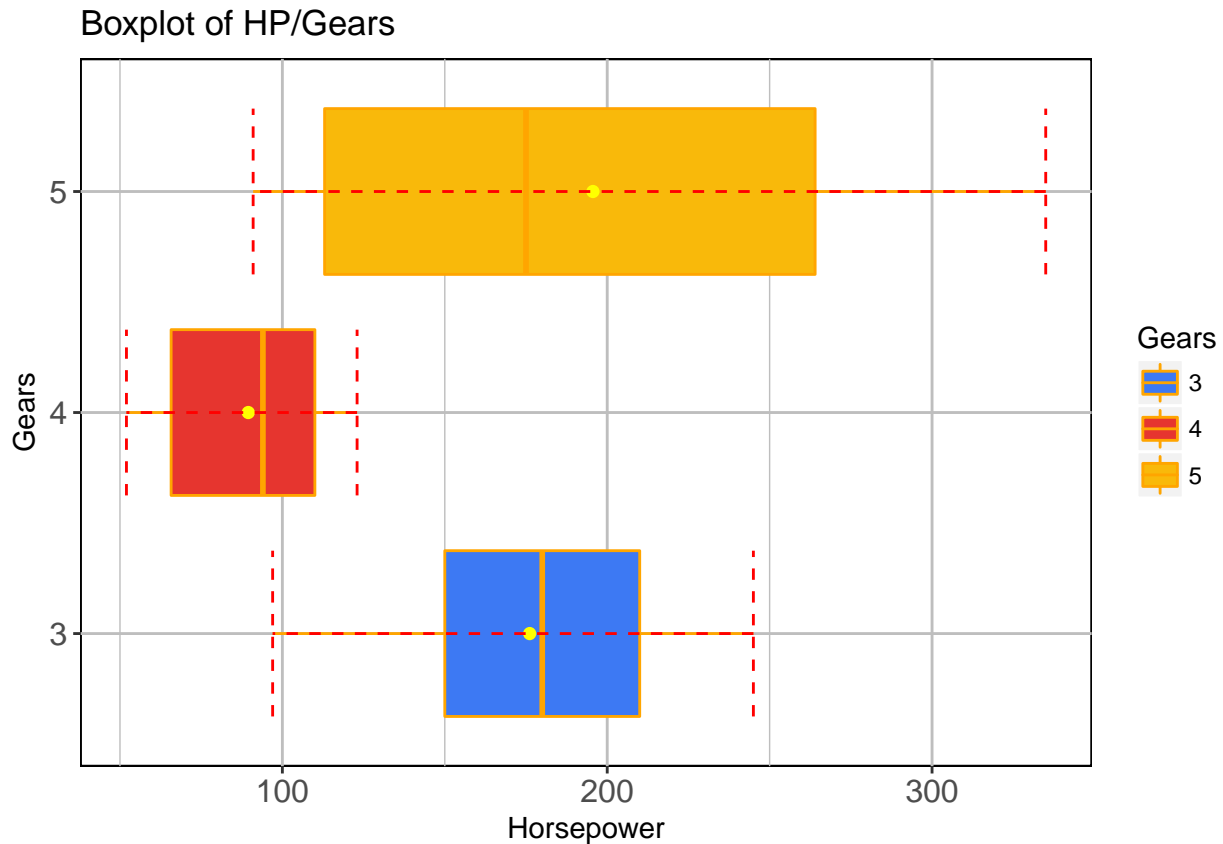
6. Boxplots for quantitative variables

```
ggplot(mtcars2, aes(gear, hp)) +
  geom_boxplot(aes(fill = gear), col = "orange", show.legend = TRUE) +
  scale_fill_manual(name = "Gears", values = myColors3) +
  coord_flip() +
```

```

stat_summary(fun.y=mean, geom="point", shape=16, size=2, col = "yellow") +
stat_boxplot(geom = "errorbar", col = "red", lty = 2) +
theme(axis.text = element_text(size = 12),
      panel.grid.major = element_line(color = "grey"),
      panel.grid.minor = element_line(color = "grey"), # element_blank() gets rid of minor grid
      panel.background = element_rect(fill = "white", color = "black"))+
ggtitle("Boxplot of HP/Gears") +
xlab("Gears") +
ylab("Horsepower")

```



7. It is getting more intense - we are going to plot the meanHP/cylinder

Some data-processing to get mean hp (we create a new dataset with meanHP and cyl from mtcars)

```

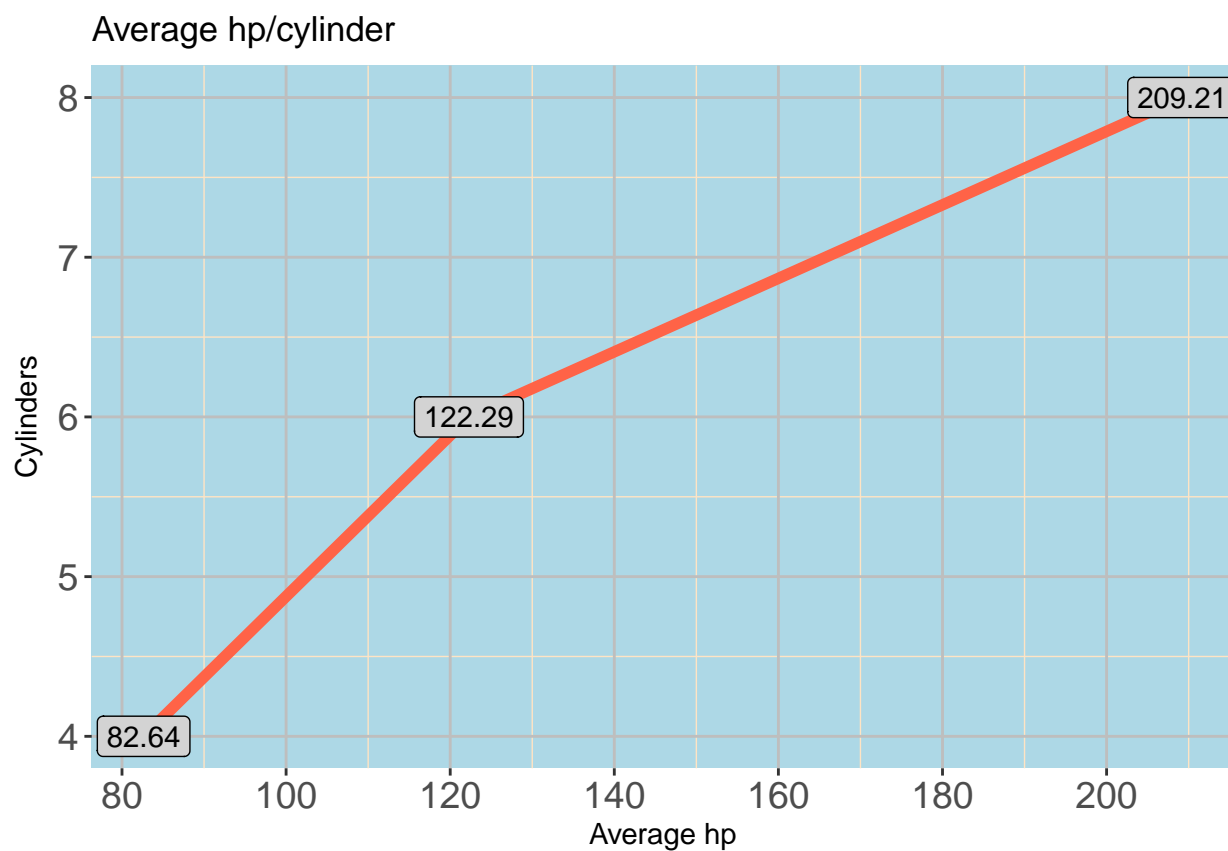
meanHp <- mtcars %>%
  group_by(cyl) %>%
  filter(!is.na(hp)) %>%
  summarize(avg_hp = mean(hp, na.rm=TRUE))

```

Actual plot

```
p1 <- ggplot(meanHp, aes(avg_hp, cyl, label = round(avg_hp, 2))) +
  geom_line(lwd = 2, col = "tomato") +
  geom_text(check_overlap = TRUE) + # the text and label add-ons overwrite the points!
  geom_label(color = "black", bg = "lightgrey") +
  ggtitle("Average hp/cylinder") +
  ylab("Cylinders") +
  xlab("Average hp") +
  scale_x_continuous(breaks=seq(0,220,20)) + # set tick marks manually
  theme(
    axis.text = element_text(size = 14),
    panel.grid.major = element_line(color = "grey"),
    panel.grid.minor = element_line(color = "bisque"), # element_blank() gets rid of minor
    panel.background = element_rect(fill = "lightblue"))
```

p1



Saving the plot as a .png file

```
ggsave(p1, filename = "AvgHP.png")
```


8. Now for the fun part, working with real-life online data (RCurl package) - crime statistics

Getting the data

```
onlineData <- getURL("http://www.onthelambda.com/wp-content/uploads/2014/07/CrimeStatebyState.csv",
                     ssl.verifyhost=FALSE, ssl.verifypeer=FALSE)

class(onlineData) # needs to be put into a dataframe

## [1] "character"

crimeData <- read.csv(textConnection(onlineData), header = TRUE)
rm(onlineData) # removes onlineData so we can keep everything nice and clean
```

9. Plot crime counts per year

```
head(crimeData)

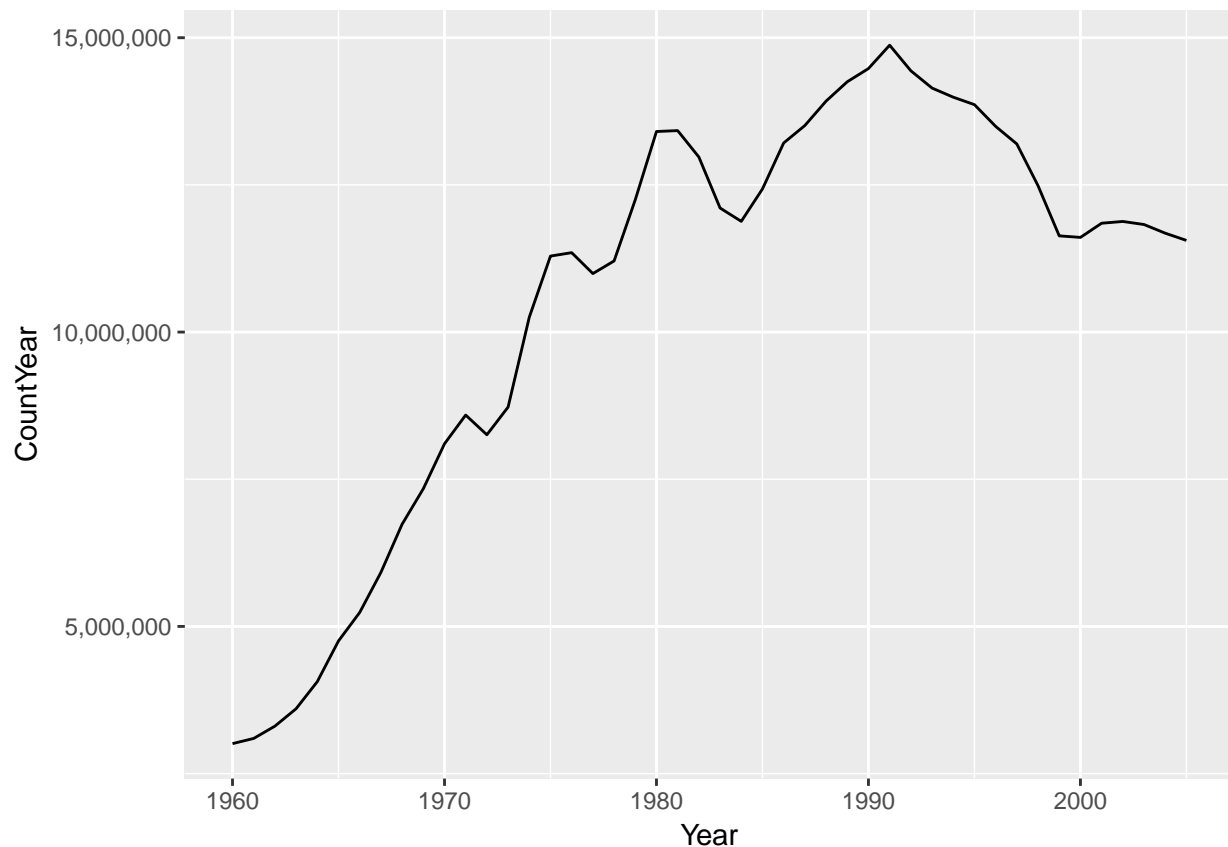
##      State Type.of.Crime      Crime Year Count
## 1 Alabama Violent Crime Murder and nonnegligent Manslaughter 1960    406
## 2 Alabama Violent Crime Murder and nonnegligent Manslaughter 1961    427
## 3 Alabama Violent Crime Murder and nonnegligent Manslaughter 1962    316
## 4 Alabama Violent Crime Murder and nonnegligent Manslaughter 1963    340
## 5 Alabama Violent Crime Murder and nonnegligent Manslaughter 1964    316
## 6 Alabama Violent Crime Murder and nonnegligent Manslaughter 1965    395

str(crimeData)

## 'data.frame':    16422 obs. of  5 variables:
##  $ State      : Factor w/ 51 levels "Alabama","Alaska",...: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Type.of.Crime: Factor w/ 2 levels "Property Crime",...: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Crime       : Factor w/ 7 levels "Aggravated assault",...: 6 6 6 6 6 6 6 6 6 6 ...
##  $ Year        : int  1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 ...
##  $ Count       : int  406 427 316 340 316 395 384 415 421 485 ...

crimeCountYear <- crimeData %>%
  group_by(Year) %>%
  summarize(CountYear = sum(Count))

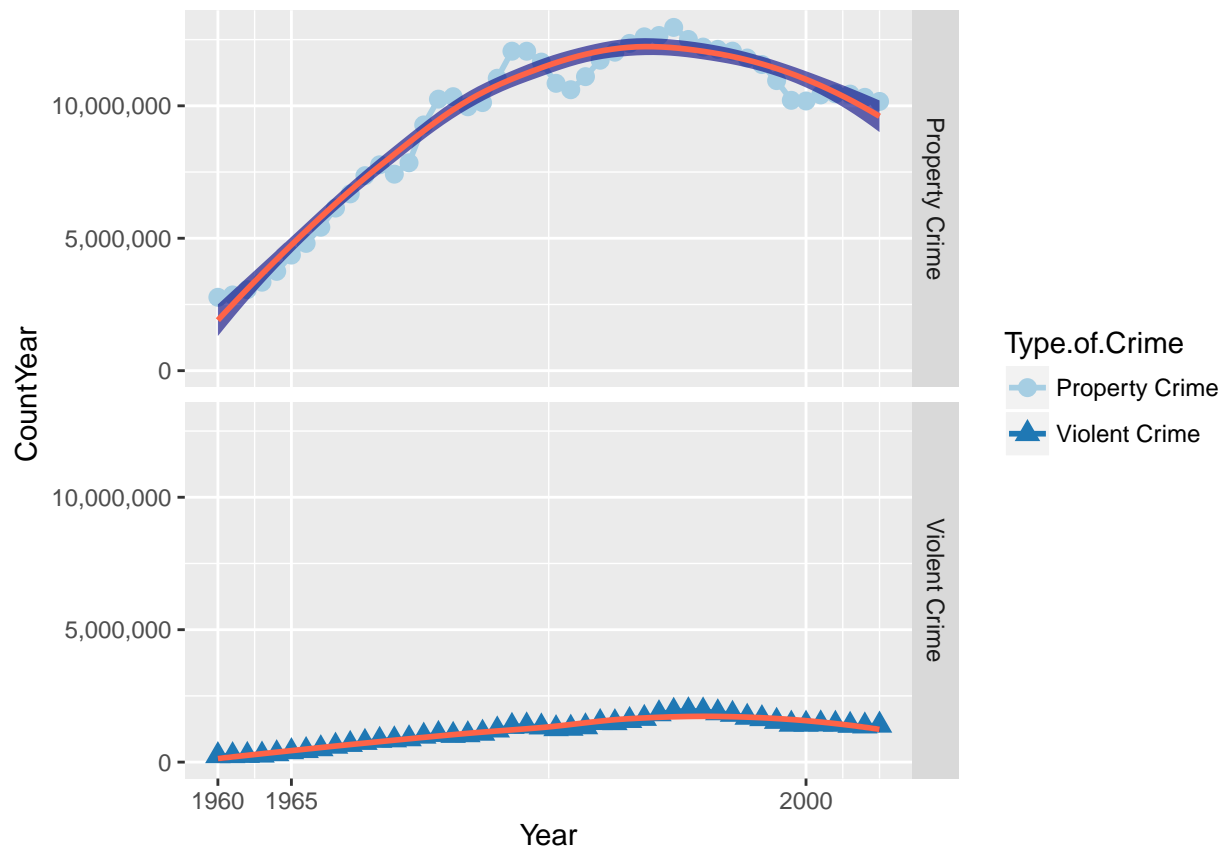
ggplot(crimeCountYear, aes(Year, CountYear)) +
  geom_line() +
  scale_y_continuous(labels = comma)
```



```
crimeCountYear2 <- crimeData %>%
  group_by(Year, Type.of.Crime) %>%
  summarize(CountYear = sum(Count))

ggplot(crimeCountYear2, aes(Year, CountYear, col = Type.of.Crime, shape = Type.of.Crime)) +
  geom_line(lwd = 1) +
  geom_point(size = 3) +
  scale_color_manual(values = brewer.pal(2, "Paired")) +
  geom_smooth(method = "loess", na.rm = TRUE, se=TRUE, fullrange = TRUE, fill = "navy", color = 'black') +
  facet_grid(Type.of.Crime ~.) +
  scale_x_continuous(breaks = c(1960, 1965, 2000)) +
  scale_y_continuous(labels = comma)
```

```
## Warning in brewer.pal(2, "Paired"): minimal value for n is 3, returning requested palette with 3 dif
```

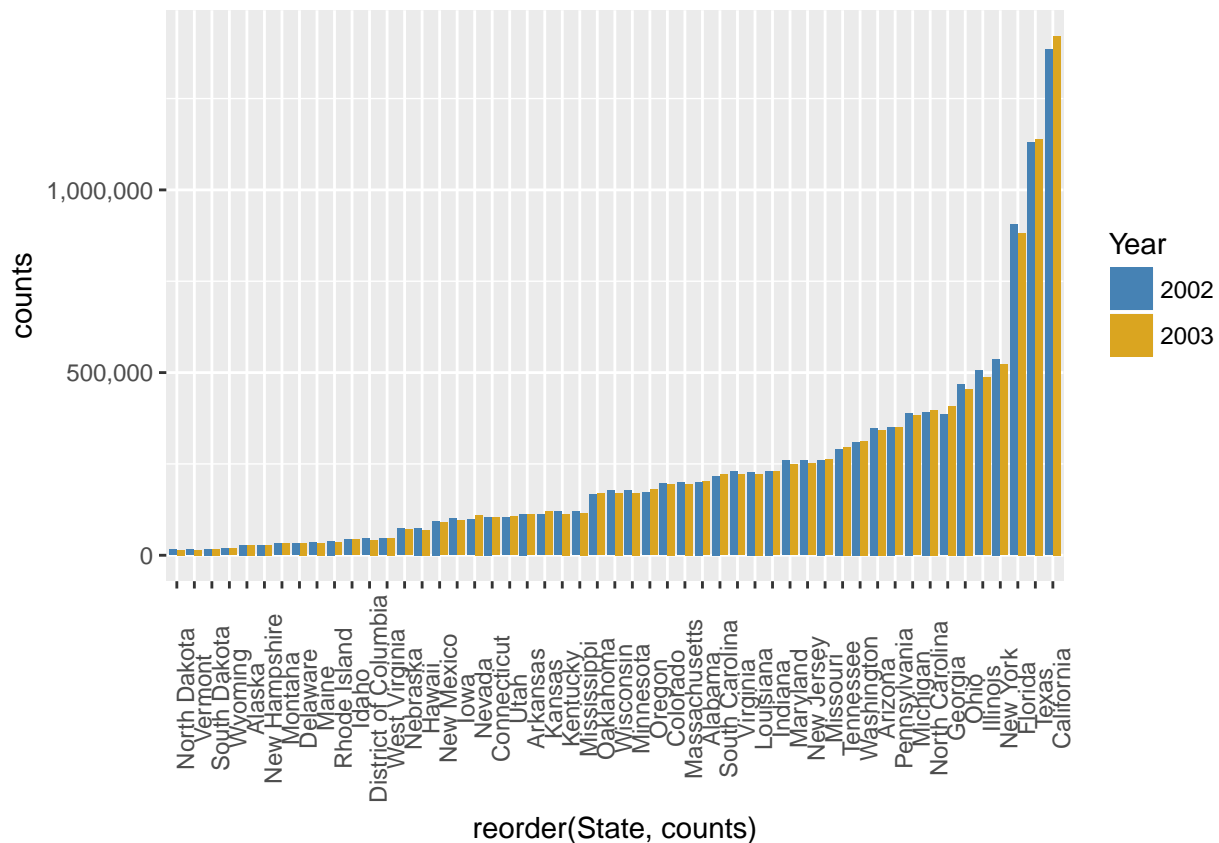


10. We are getting more selective (only years 2002 and 2003)

```
crimeSel <- crimeData %>%
  filter(Year == 2002:2003 & !is.na(Year)) %>%
  group_by(State, Year) %>%
  summarize(counts = sum(Count)) %>%
  arrange(desc(counts))

crimeSel$Year <- as.factor(crimeSel$Year)

ggplot(crimeSel, aes(reorder(State, counts), counts, fill = Year)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_fill_manual(values = c("steelblue", "goldenrod")) +
  theme(axis.text.x = element_text(angle = 90)) +
  scale_y_continuous(labels = comma)
```



11. Finally, we are going to plot the top ten of year 2000

```
crime2000 <- crimeData %>%
  filter(Year == 2002) %>%
  group_by(State) %>%
  summarize(counts = sum(Count)) %>%
  arrange(desc(counts)) %>%
  filter(counts > 350445)

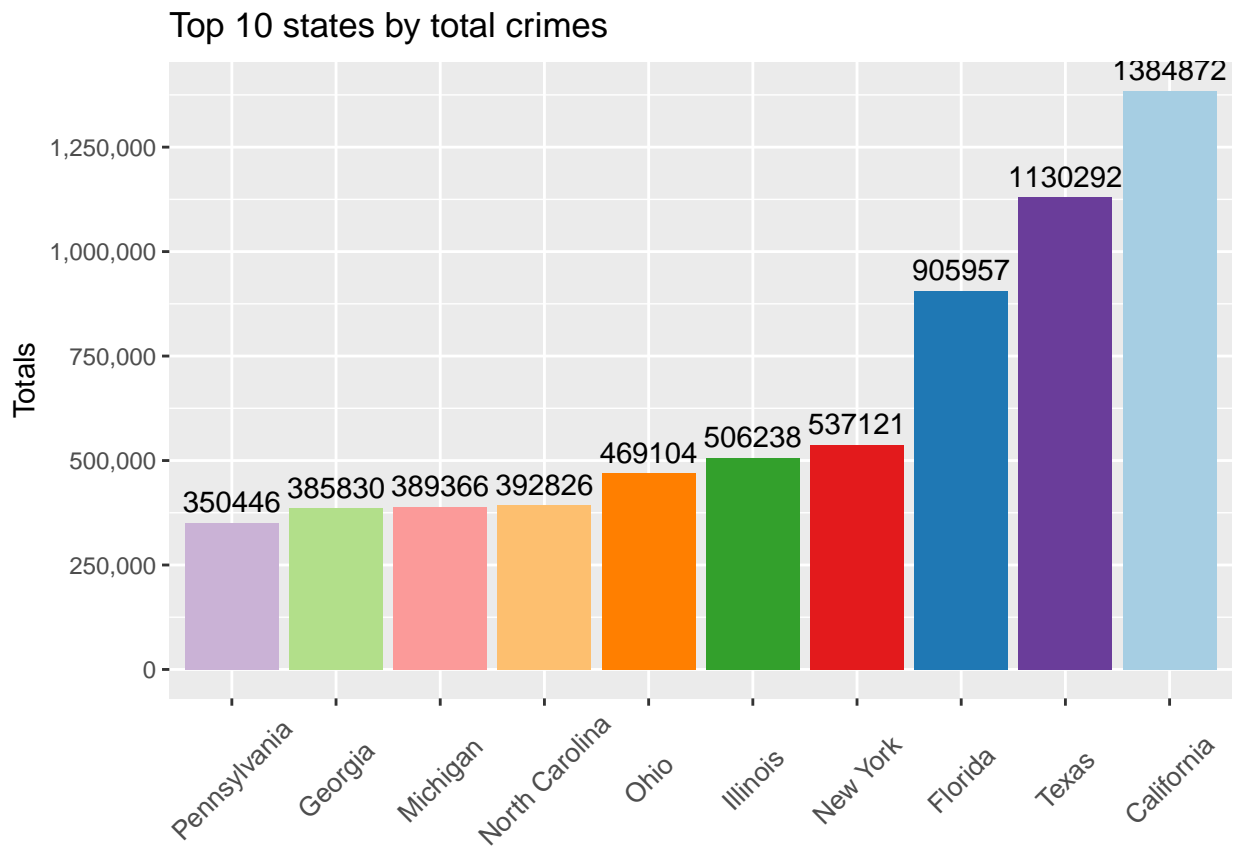
head(crime2000,10)
```

```
## # A tibble: 10 × 2
##   State counts
##   <fctr> <int>
## 1 California 1384872
## 2 Texas 1130292
## 3 Florida 905957
## 4 New York 537121
## 5 Illinois 506238
## 6 Ohio 469104
## 7 North Carolina 392826
## 8 Michigan 389366
## 9 Georgia 385830
## 10 Pennsylvania 350446
```

```
library(scales) # for label formatting
library(RColorBrewer)

crime <- ggplot(crime2000, aes(reorder(State, counts), counts)) +
  geom_bar(stat = "identity", aes(fill = State)) +
  scale_fill_manual(values = brewer.pal(10, "Paired"), guide = FALSE) +
  theme(axis.text.x = element_text(angle=45, vjust=0.5, size=10), axis.title.x = element_blank()) +
  scale_y_continuous(labels = comma, breaks = c(0, 250000, 500000, 750000, 1000000, 1250000)) + #
  geom_text(aes(State, counts, label = counts), vjust = -0.5) +
  ggtitle("Top 10 states by total crimes") +
  ylab("Totals")

crime
```



Saving the plot

```
ggsave(crime, filename = "crimePlot.png")
```

12. Gimmick - plotting with emojis

```
devtools::install_github("dill/emoGG")

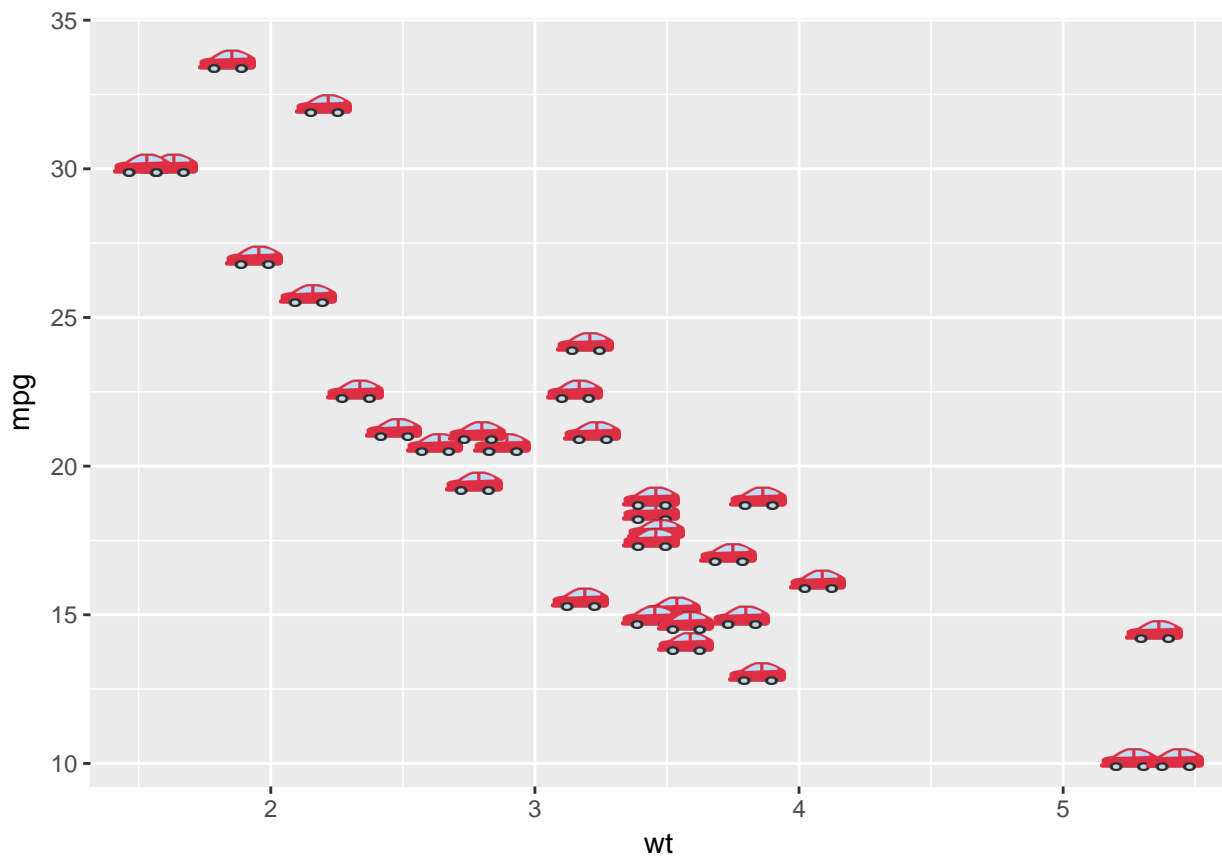
library(ggplot2)
library(emoGG)
```

```
## Find an emoji you want to use
```

```
emoji_search("car")
```

##	emoji	code	keyword
## 1753	bus	1f68c	car
## 1762	minibus	1f690	car
## 1797	car	1f697	red
## 1798	car	1f697	transportation
## 1799	car	1f697	vehicle
## 1800	oncoming_automobile	1f698	car
## 1812	tractor	1f69c	car
## 2733	a	1f170	red-square
## 2734	a	1f170	alphabet
## 2735	a	1f170	letter

```
ggplot(mtcars, aes(wt, mpg)) +  
  geom_emoji(emoji="1f697")
```



```
emoji_search('smiley')
```

##	emoji	code	keyword
## 2	grinning	1f600	smile
## 7	grin	1f601	smile
## 15	smiley	1f603	face
## 16	smiley	1f603	happy
## 17	smiley	1f603	joy
## 18	smiley	1f603	haha

```
## 19          smile 1f604      face
## 20          smile 1f604      happy
## 21          smile 1f604      joy
## 22          smile 1f604      funny
## 23          smile 1f604      haha
## 24          smile 1f604      laugh
## 25          smile 1f604      like
## 51          blush 1f60a      smile
## 65          yum 1f60b        smile
## 83          sunglasses 1f60e  smile
## 87          smirk 1f60f       smile
## 149         stuck_out_tongue 1f61b smile
## 155 stuck_out_tongue_winking_eye 1f61c smile
## 160 stuck_out_tongue_closed_eyes 1f61d smile
## 3128         m 24c2      alphabet
## 3129         m 24c2 blue-circle
## 3130         m 24c2      letter
```

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_emoji(emoji="1f604")
```

