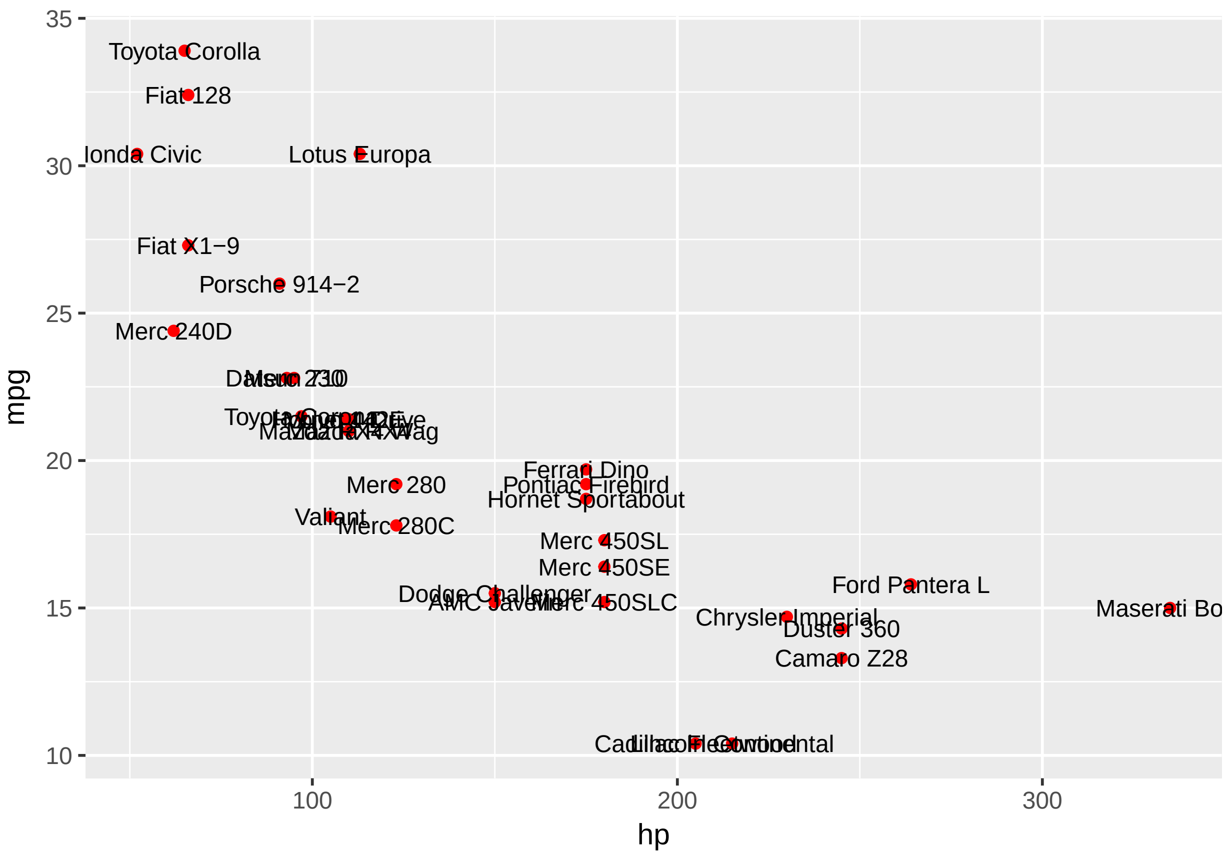
Introduce Visualization Techniques

Hanh Nguyen

Libraries used: ggplot2, dplyr, ggrepel, ggmap

Labels

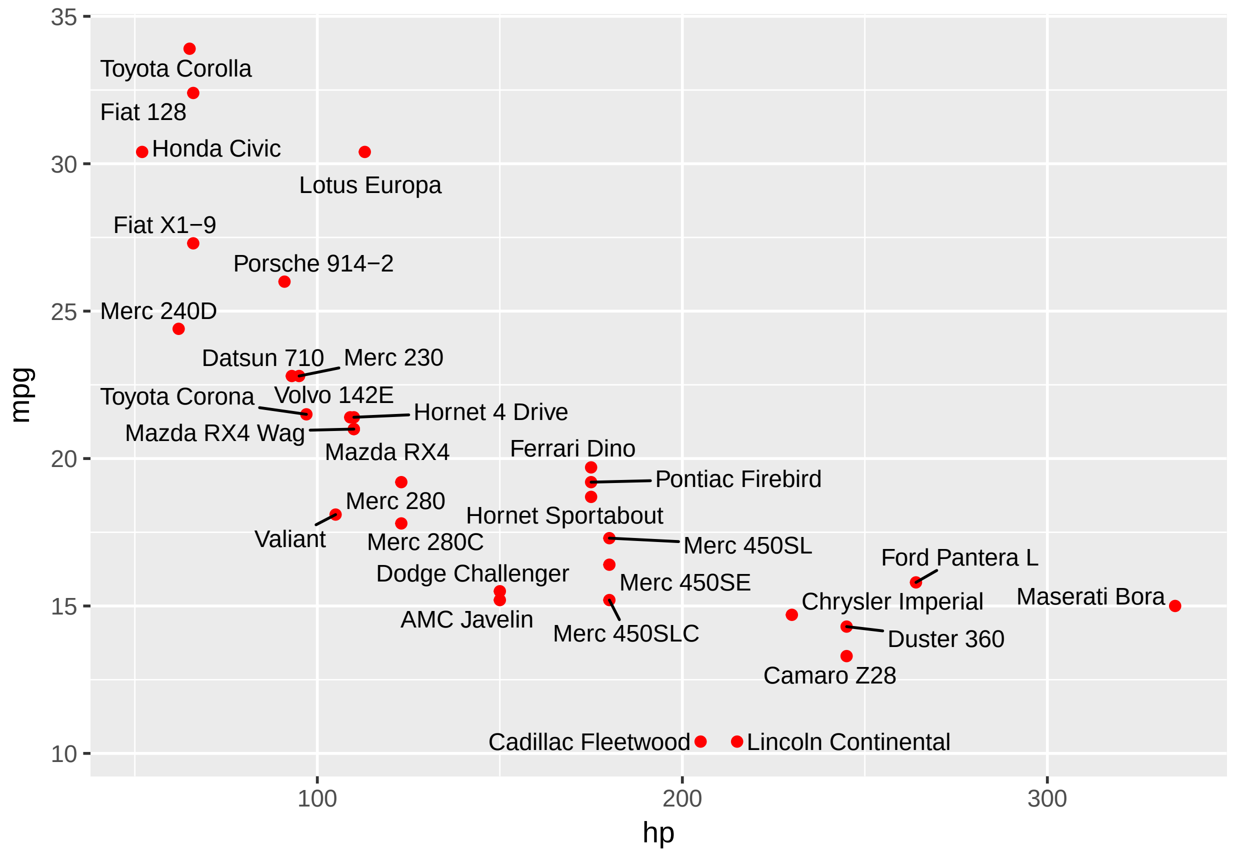
```
mtcars %>%
  ggplot(aes(x = hp, y = mpg)) +
  geom_point(color = 'red') +
  geom_text(aes(hp, mpg, label = rownames(mtcars)),size=3)
```



Labels using ggrepel

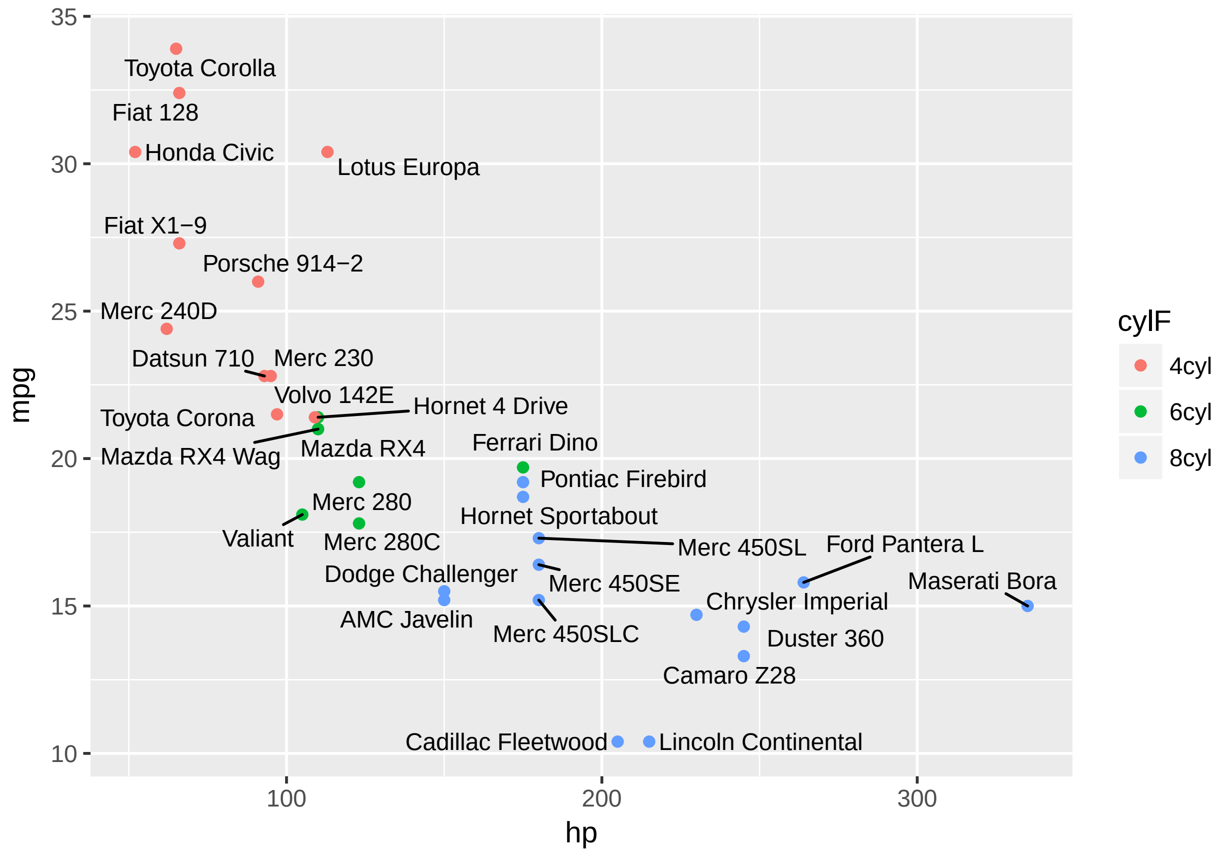
ggrepel implements functions to repel overlapping text labels away from each other and away from the data points that they label.

```
mtcars %>%
  ggplot(aes(x = hp, y = mpg)) +
  geom_point(color = 'red') +
  geom_text_repel(aes(hp, mpg, label = rownames(mtcars)),size=3)
```



Two (2) numeric variables, one (1) categorical variable and data labels

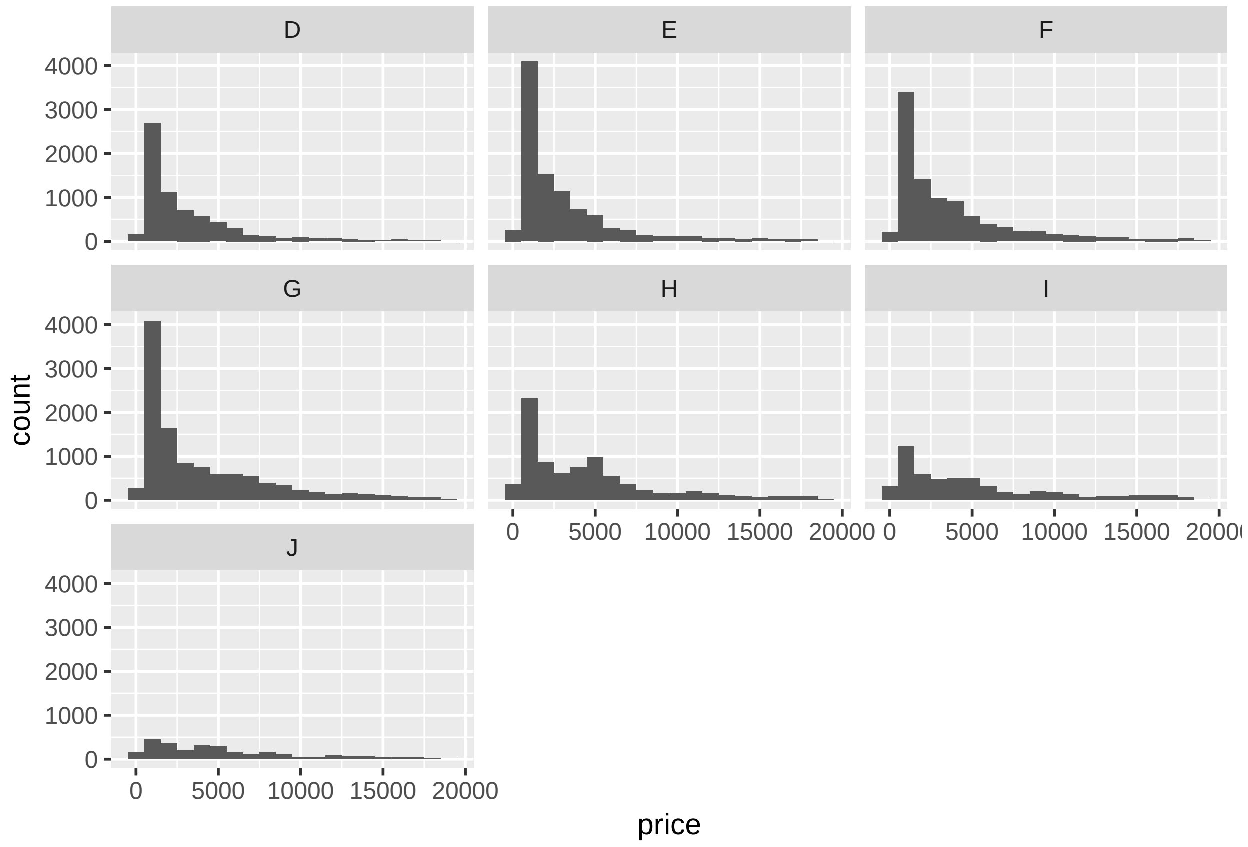
```
mtcars$cylF = factor(mtcars$cyl,
                    levels = c(4,6,8),
                    labels = c("4cyl","6cyl","8cyl"))

mtcars %>%
  ggplot(aes(x = hp, y = mpg)) +
  geom_point(aes(color = cylF)) +
  geom_text_repel(aes(hp, mpg, label = rownames(mtcars)),size=3)
```



Facets

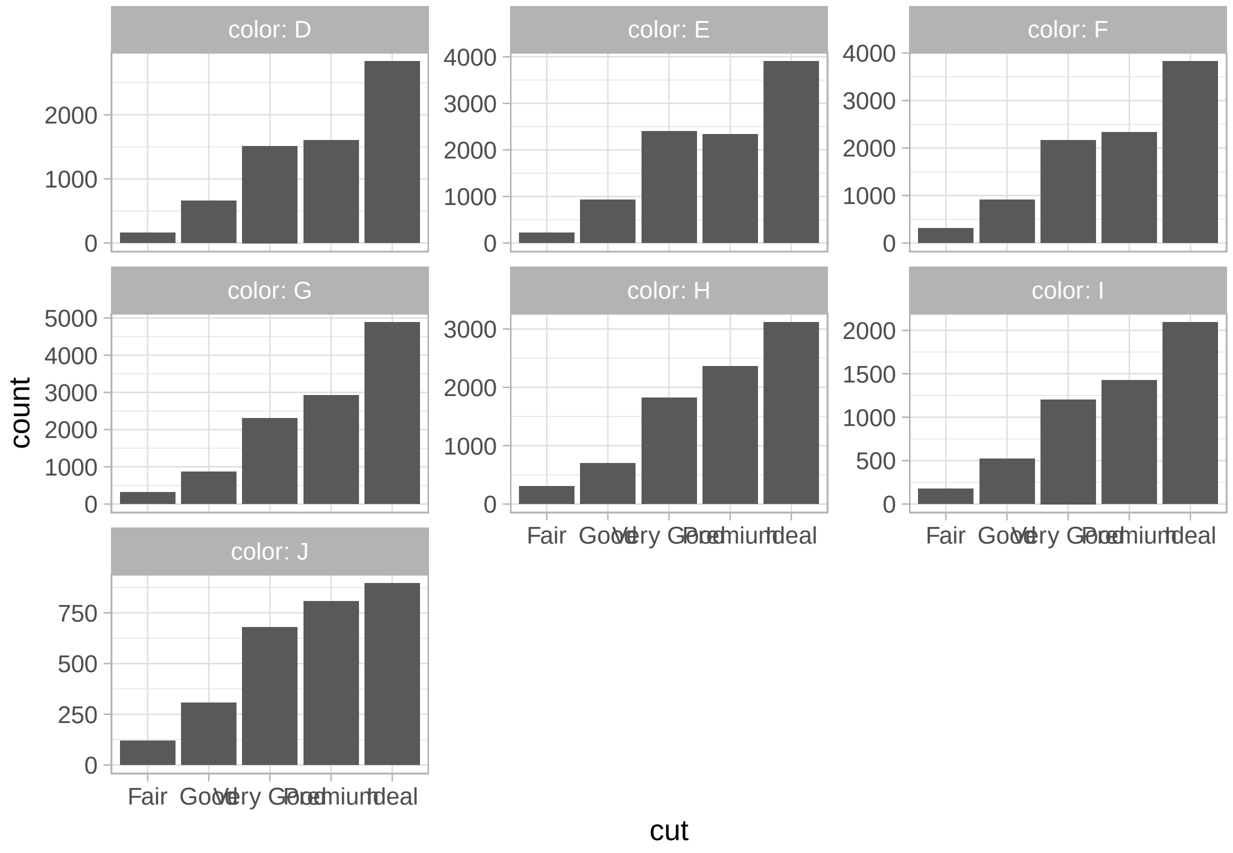
facet_wrap() defines subsets as the levels of a single grouping variable and splits the plot across categories. facet_grid() defines subsets as the crossing of two grouping variables, creates different grids and then plots each plot in the grids instead of creating different plots.

```
diamonds %>%
  ggplot(aes(x=price)) +
  geom_histogram(binwidth=1000) +
  facet_wrap(~ color)
```



The above example keeps the scales constant across all panels, but we can choose to the x scale, the y scale, or both.

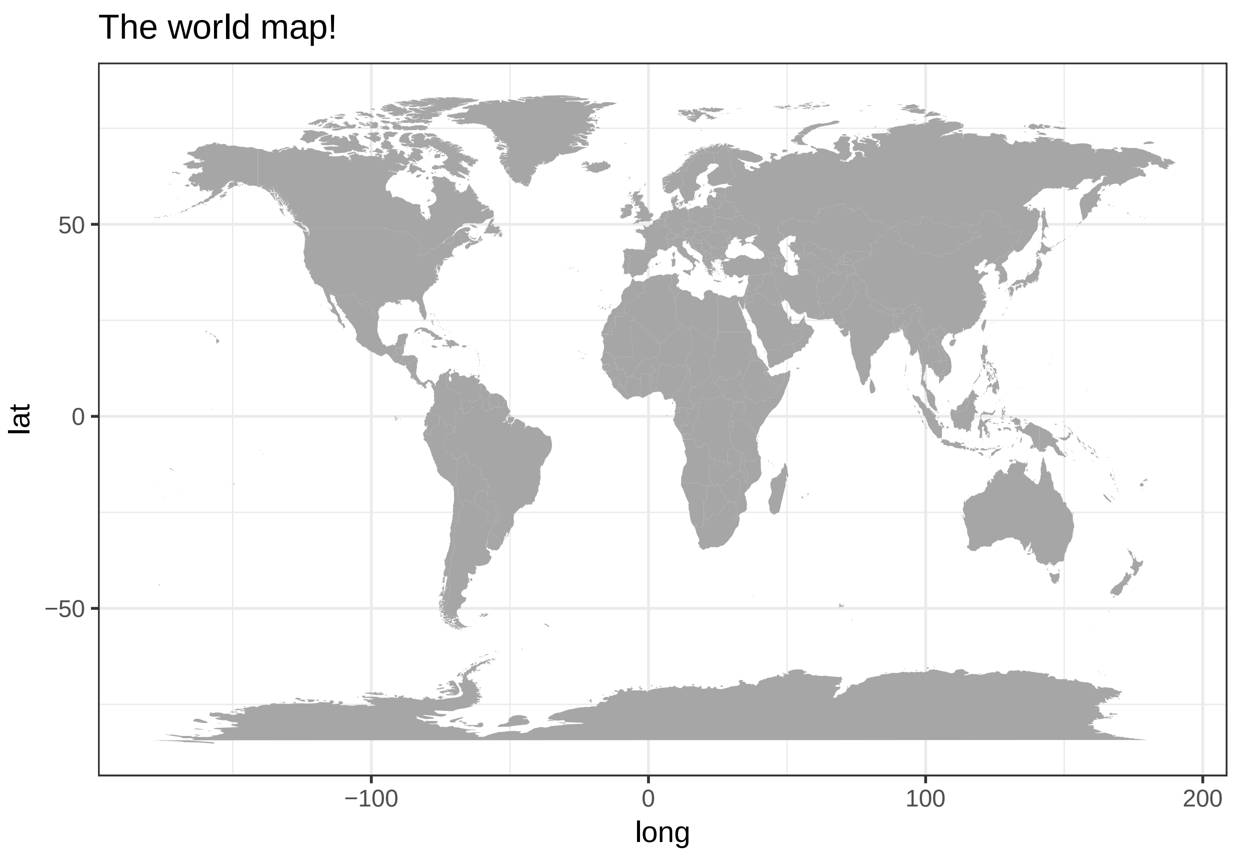
```
diamonds %>%
  ggplot(aes(x=cut)) +
  geom_bar() +
  facet_wrap(~ color, labeller=label_both, scale="free_y") +
  theme_light()
```



Choropleth map

Choropleth maps are helpful for visualizing geographic data.

```
dfworldmap = map_data("world")
dfworldmap %>%
  ggplot()+
  geom_polygon(aes(long,lat, group=group), fill="grey65") +
  theme_bw() +
  ggtitle("The world map!")
```



We demonstrate a country's data with its map. This example will show the number of Murder arrests in each state of the US using USArrests dataset and then put the statistics into the US map.

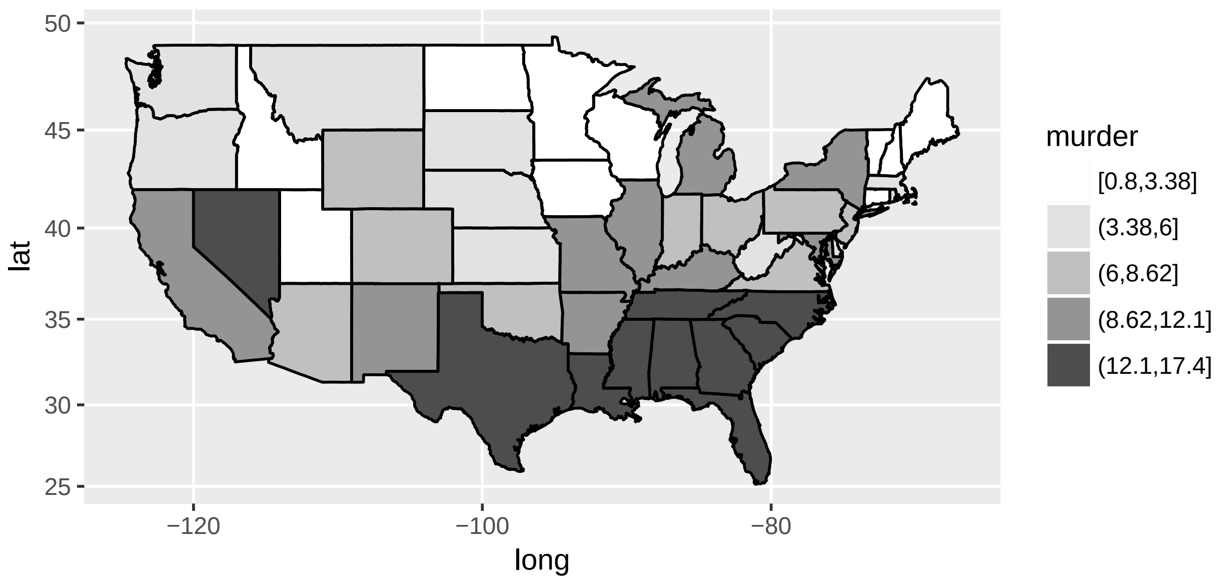
Step 1: transform data

```
arrest <- cbind(region = rownames(USArrests), USArrests)
arrest$region=tolower(arrest$region)
arrest = transform(arrest, murder = cut_number(Murder, 5))
```

Step 2: create the map

map_data() turns data from the maps package into a data frame suitable for plotting with ggplot2. coord_map() projects a portion of the earth, which is approximately spherical, onto a flat 2D plane. scale_fill_grey() changes the colors from the default unordered multiple colors to an ordered and print-friendly black and white (see also scale_fill_brewer).

```
us_map = map_data("state")
map_data = merge(arrest, us_map, by="region")
map_data = map_data[order(map_data$order),]
map_data %>%
  ggplot(aes(x=long, y=lat, group=group)) +
  geom_polygon(aes(fill = murder)) +
  scale_fill_grey(start=1, end =.3) +
  geom_path(colour='black') +
  coord_map()
```



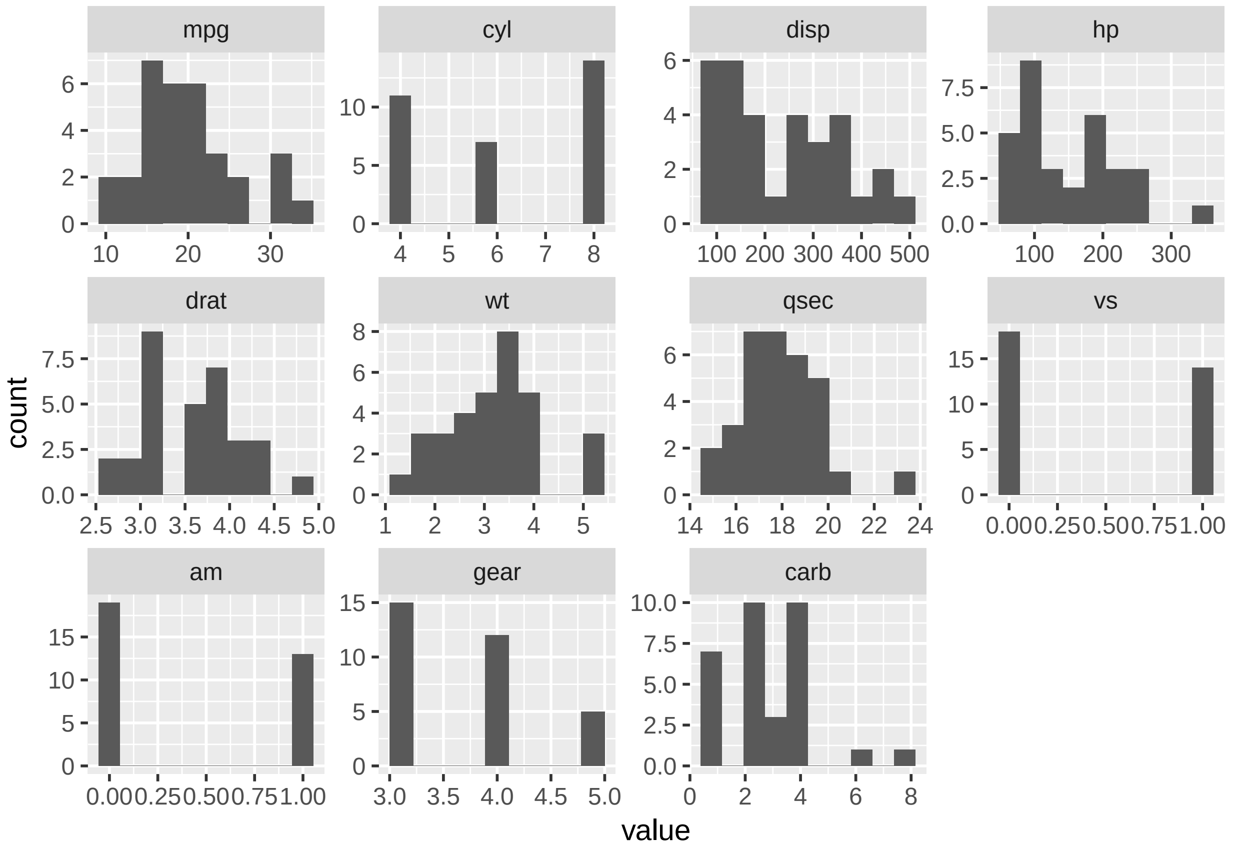
Plotting all variables of a dataset in a plot

The goal is to see the distributions of each variable at once, without having to view one at a time and keep clicking back and forth through our plot pane.

To do that, we need to stack columns into two groups only: identifier and measured variables using melt() function from reshape2 package. For further explanation, please refer to DataTidying.Rmd

```r
melt(mtcars) %>%
  ggplot(aes(x=value)) +
  geom_histogram(bins = 10) +
  facet_wrap(~variable, scales = 'free')
```

```
## Using cylF as id variables
```



Sources

https://cran.r-project.org/web/packages/ggrepel/vignettes/ggrepel.html
http://docs.ggplot2.org/0.9.3.1/facet_wrap.html
http://web.stanford.edu/~imalone/VAM/ggmap.html
https://nhorton.people.amherst.edu/sasr2/choropleth.pdf