ggplot2

Pedro de Brito Neto

23/08/2021

```
library(tidyverse)
```

Position

- a explicação/motivação contida aqui irá depender do post da Flávia, se ela falou sobre gráficos com grupos.

quando falamos em agrupos as informações nos gráficos, podemos pensar em 3 argumentos principais para o `position`: `position = "stack"`, `position = "fill"` e `position = "dodge"`. Basicamente cada um desses irá agrupar suas informações de formas diferentes. Para mostrar isso vamos utilizar a base de dados *Cars93* do pacote `MASS`, caso deseje utilizar essa base de dados e não tenha o pacote instalado no seu computador, basta rodar o código `install.packages("MASS")`.

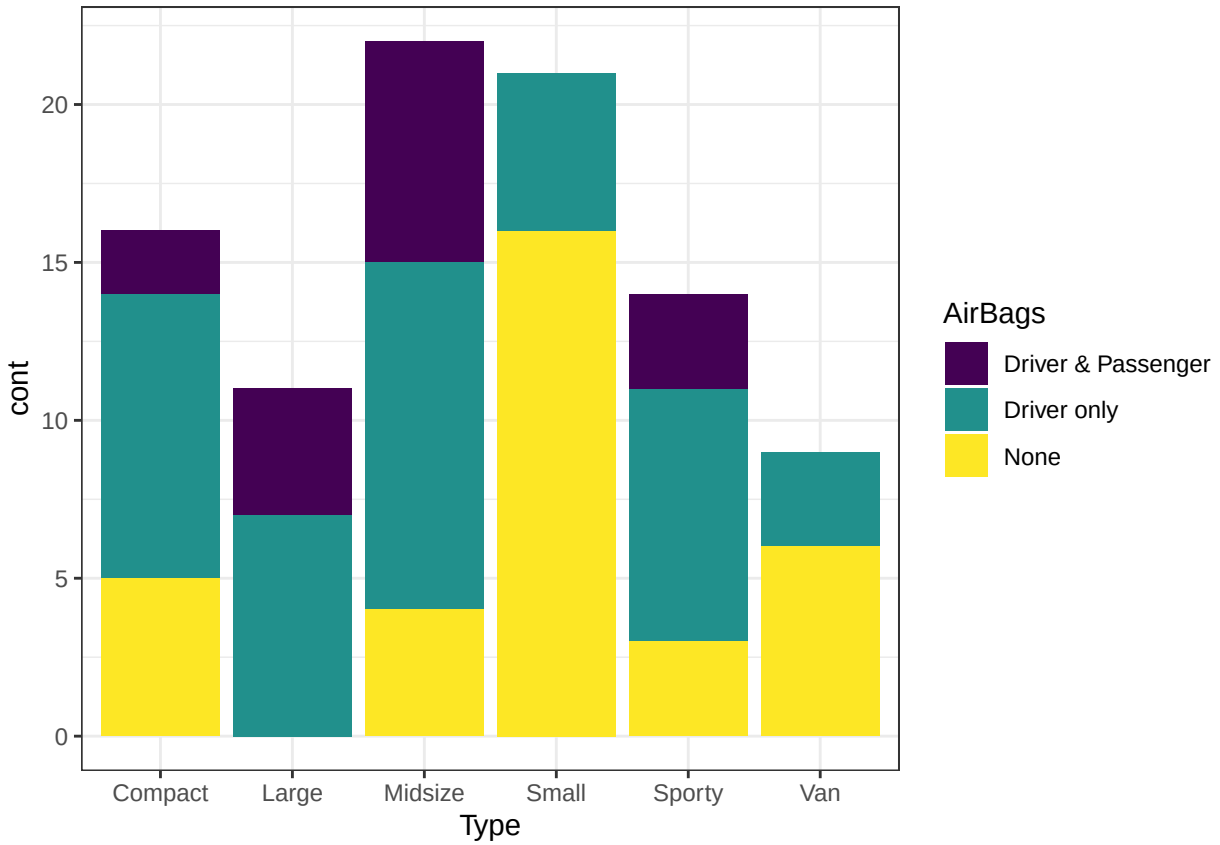
- dar exemplos de situações onde é interessante usar cada caso

```
library(MASS)
```

```
dados_resumo <- Cars93 %>%
  group_by(Type, AirBags) %>%
  summarise(cont = n())
```

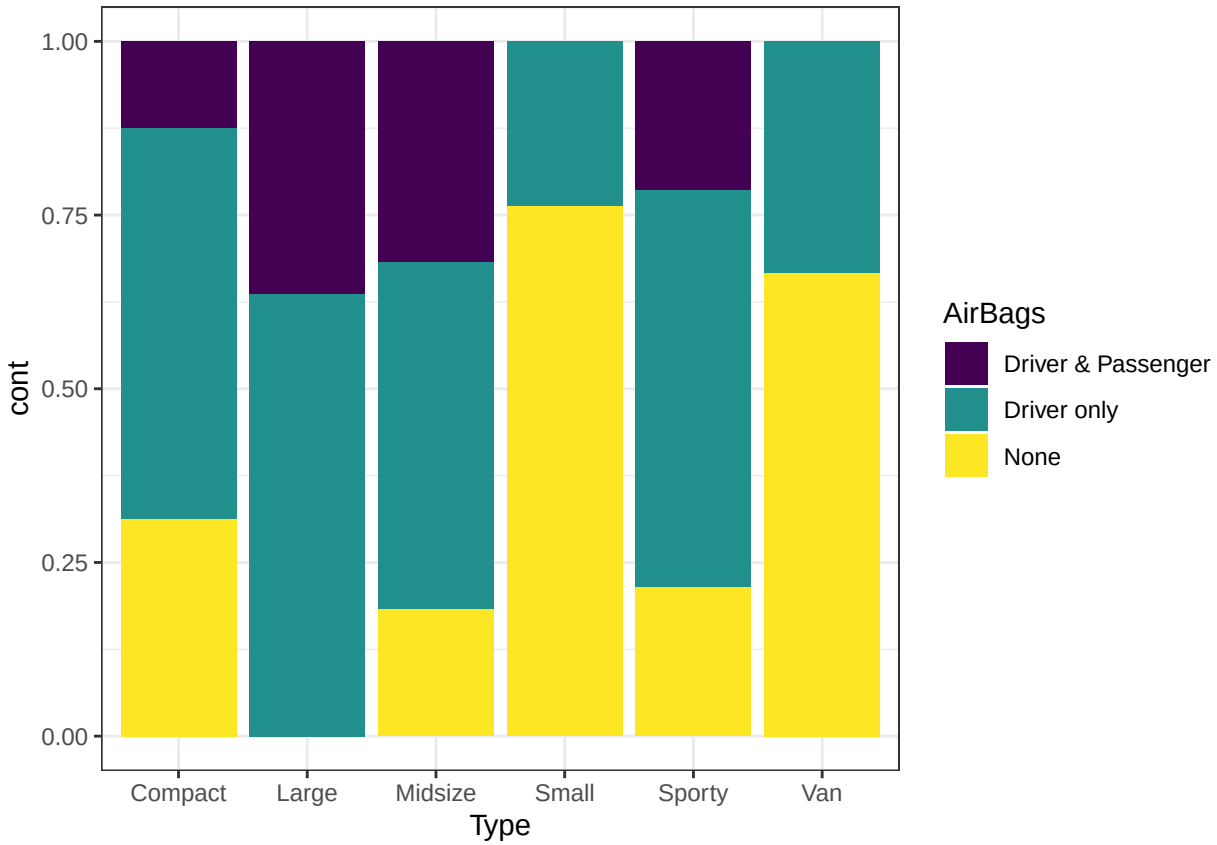
```
## `summarise()` has grouped output by 'Type'. You can override using the `.groups` argument.
```

```
dados_resumo %>%
  ggplot() +
  geom_bar(aes(x = Type, y = cont, group = AirBags, fill = AirBags),
           stat = "identity", position = "stack") +
  scale_fill_viridis_d() +
  theme_bw()
```



No gráfico acima utilizamos o `position = "stack"`, ele fornece gráfico de barras empilhado. Os subgrupos são exibidos apenas uns sobre os outros.

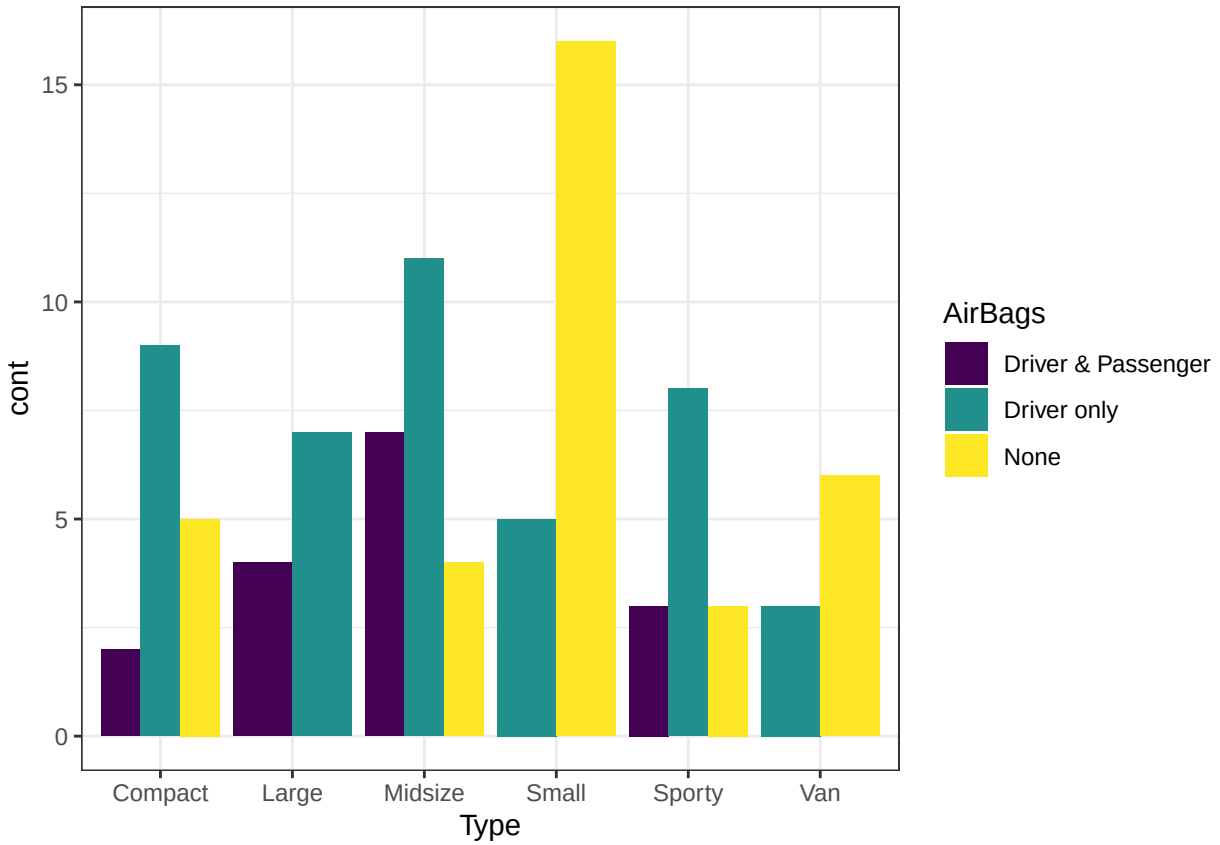
```
dados_resumo %>%
  ggplot() +
  geom_bar(aes(x = Type, y = cont, group = AirBags, fill = AirBags),
           stat = "identity", position = "fill") +
  scale_fill_viridis_d() +
  theme_bw()
```



`position = "fill"` empilha barras e padroniza cada pilha para ter altura constante. Agora, a porcentagem de cada subgrupo está representada, permitindo estudar a evolução de sua proporção no todo.

```
#position = position_dodge2(width = 0.5, preserve = "single", padding = -0.5)

dados_resumo %>%
  ggplot() +
  geom_bar(aes(x = Type, y = cont, group = AirBags, fill = AirBags),
           stat = "identity", position = "dodge") +
  scale_fill_viridis_d() +
  theme_bw()
```



`position = "dodge"` pode-se dizer que é um dos mais utilizado, ele irá colocar as barras lado a lado. Normalmente, informações desse tipo ficam mais fáceis de serem visualizadas, a não ser que exitam muitas informações podendo ficar um pouco confuso. Caso isso acontece pode ser interessante fazer algumas modificações nos gráficos.

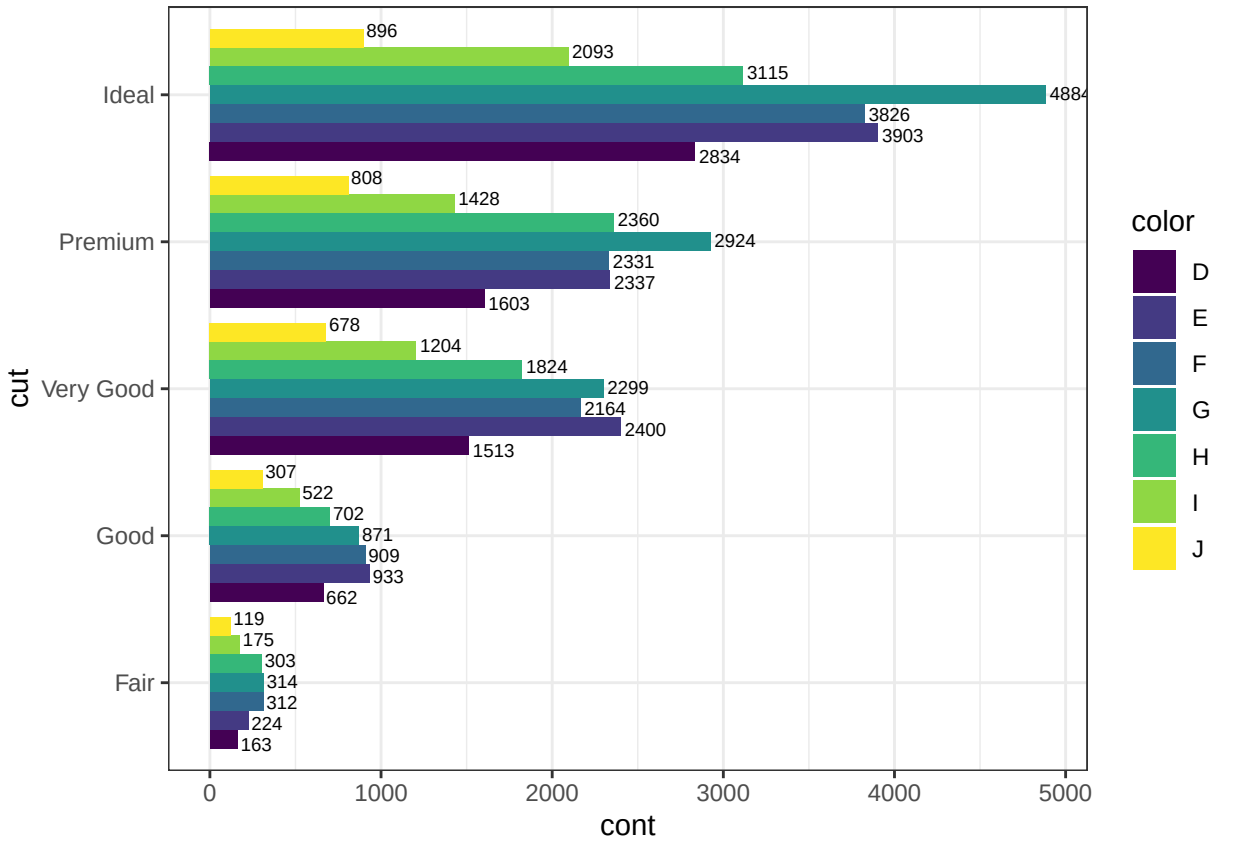
Adicionando algumas informações

Frequência Como dito acima, as vezes é interessante adicionar algumas informações aos gráficos para facilitar a visualização. Para o próximo vamos utilizar a base de dados *diamonds* contida no R.

```
dados_resumo2 <- diamonds %>%
  group_by(cut, color) %>%
  summarise(cont = n())
```

```
## `summarise()` has grouped output by 'cut'. You can override using the `.groups` argument.
```

```
dados_resumo2 %>%
  ggplot() +
  geom_bar(aes(x = cut, y = cont, group = color, fill = color),
           stat = "identity", position = "dodge") +
  geom_text(aes(x = cut, y = cont, label = cont, group = color),
            position =  position_dodge(width = 1), size = 2.5,
            vjust = 0.4, hjust = -0.1, angle = 0) +
  coord_flip() +
  theme_bw()
```



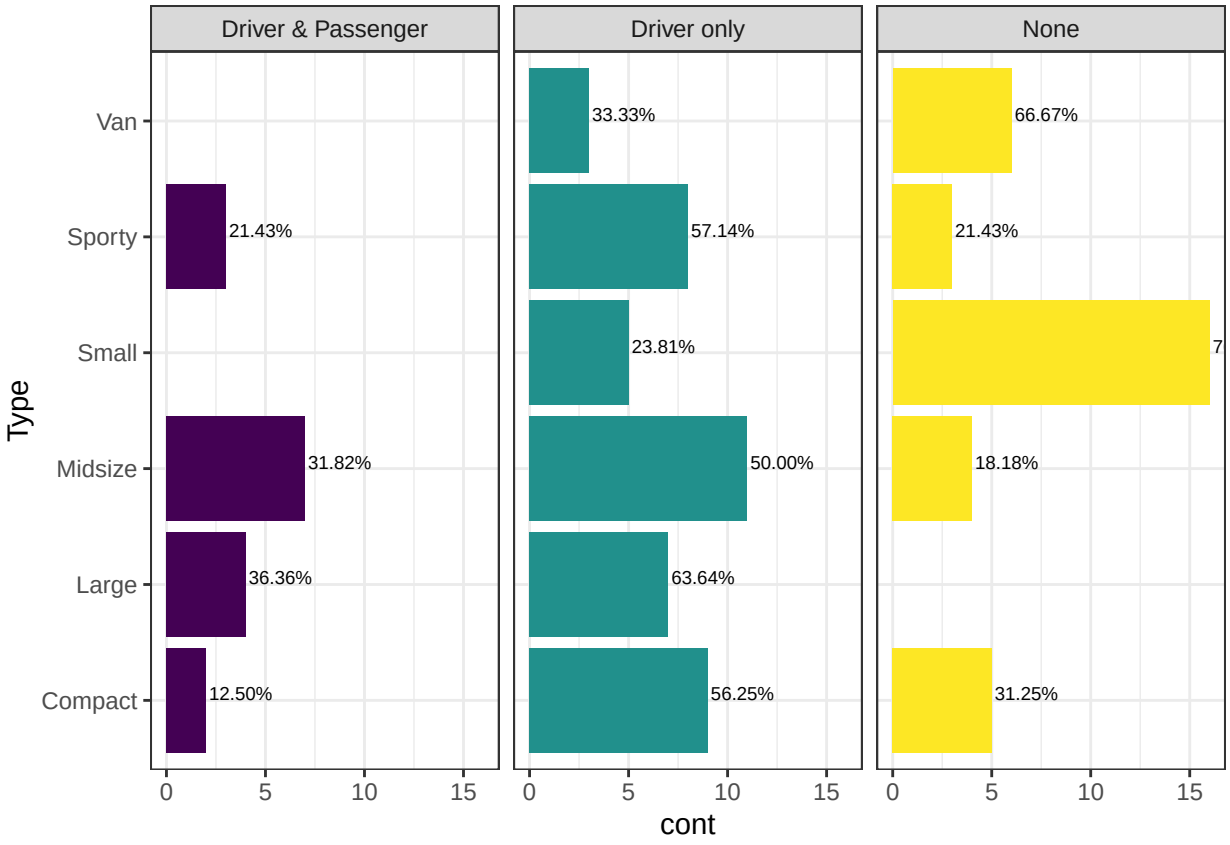
No gráfico acima adicionamos a frequência respectiva acima de cada barra, fizemos isso com a ajuda do `geom_text`, note que utilizamos o `position = position_dodge(width = 1)`. diferente do `position = "dodge"` usado anteriormente, essa outra maneira nos permite configuar algumas coisas como o `width`. Nesse caso foi necessário usar para "configuar" a posição da legenda. Também podemos "brincar" com outros ajustes como o `vjust` e `hjust` que servem para mover a legenda verticalmente e horizontalmente. Também temos o `angle`, que rotacina as legendas quando você fornece o ângulo desejado.

```
dados_resumo <- dados_resumo %>%
   mutate(pct = prop.table(cont))


dados_resumo %>%
  ggplot(aes(label = scales::percent(pct))) +
  geom_bar(aes(x = Type, y = cont, group = AirBags, fill = AirBags),
           stat = "identity", position = "dodge") +
  geom_text(aes(x = Type, y = cont, group = AirBags, fill = AirBags), position = position_dodge(width =
            size = 2.3,
            hjust = - 0.05) +
  scale_fill_viridis_d() +
  theme_bw() +
  facet_wrap(~AirBags) +
  theme(legend.position="none") +
  coord_flip()
```

Porcentagem

```
## Warning: Ignoring unknown aesthetics: fill
```



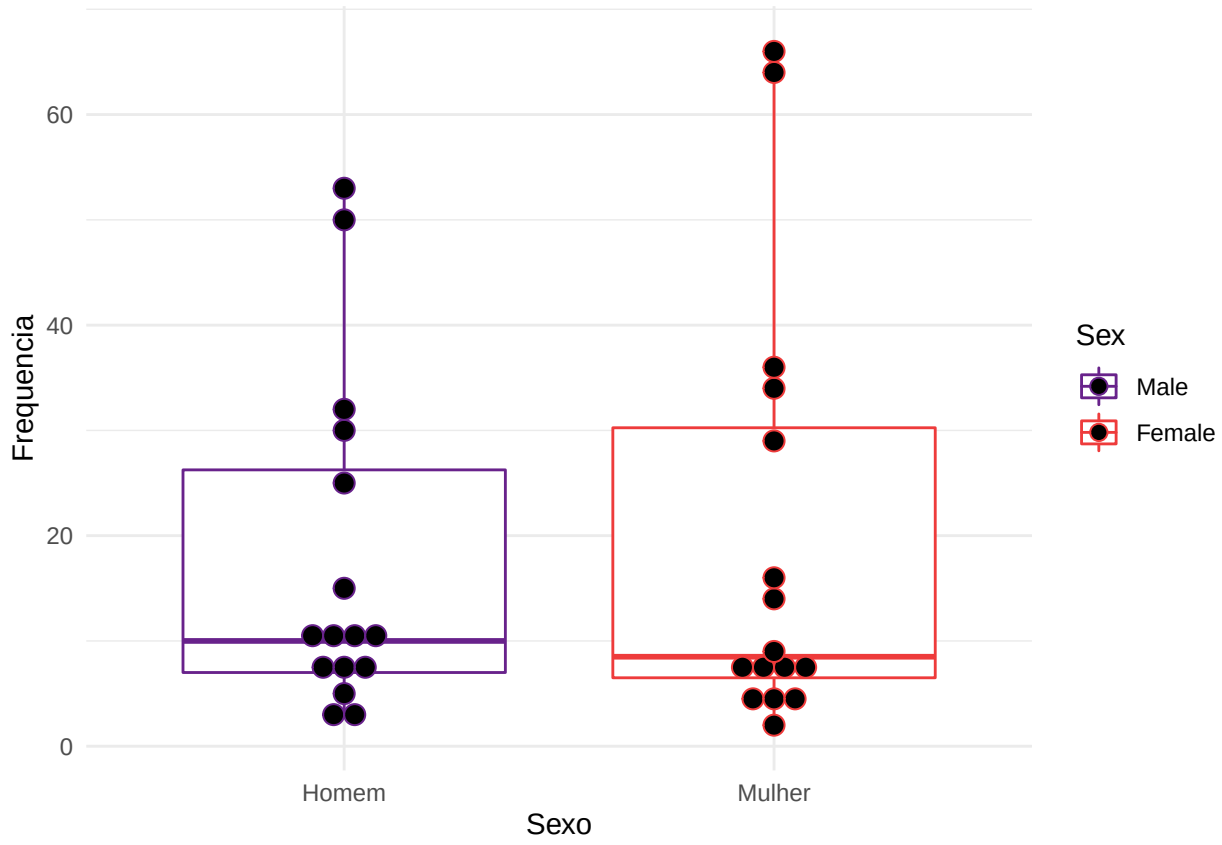
Também pode ser interessante em alguns caso adicionar a porcentagem relativa em cada barra, uma maneira de fazer e criar uma coluna no seu banco de dados que irá conter a porcentagem de cada observação em relação ao total, usamos o `prop.table()`. Agora é só fazer parecido com o que fizemos anteriormente, passando as coordenadas para o `geom_text()` e ajustando ao seu gosto pessoal. Aqui precisamos adicionar também um *label* e ara isso utilizamos a função `percent()` do pacote `scales`, lembrando que caso não tenha o pacote no seu computador será necessário fazer a instalação.

Encrementando um Boxplot

Um gráfico comumente utilizado para visualizar dados é o Boxplot. medidas de estatísticas descritivas como o mínimo, máximo, primeiro quartil, segundo quartil ou mediana e o terceiro quartil formam o boxplot.. Também nos permite visualizar a distribuição e valores discrepantes (outliers) dos dados, fornecendo assim um meio complementar para desenvolver uma perspectiva sobre o caráter dos dados. Podemos montar desde gráficos simples até alguns mais elaborados e esteticamente mais bonitos.

```
dfhair <- data.frame(HairEyeColor)
ggplot(dfhair, aes(x = Sex, y = Freq,
                   color = Sex)) +
  theme_minimal() +
  geom_boxplot() +
  geom_dotplot(
```

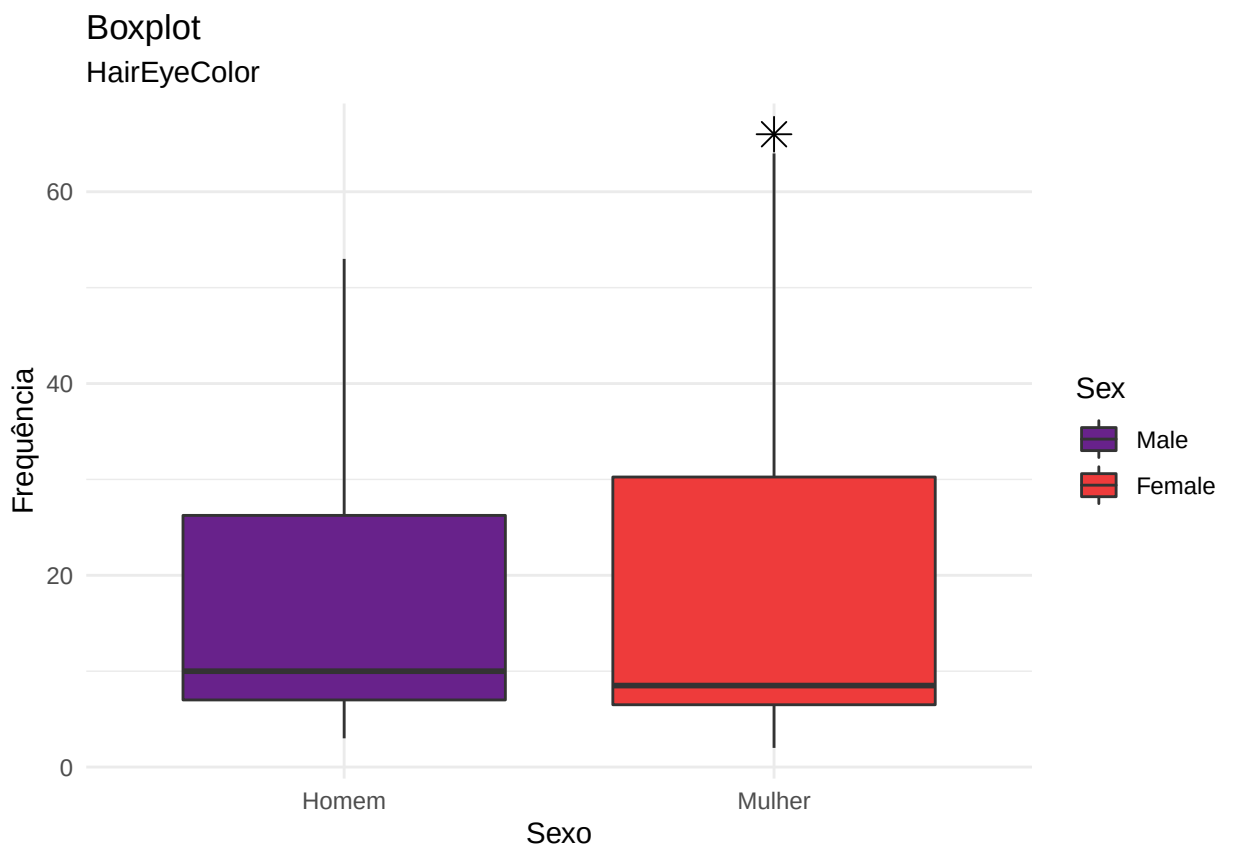
```
    binaxis = 'y',
    stackdir = 'center',
    dotsize = 1,
    binwidth = 2
  ) +
  scale_x_discrete(labels = c("Homem", "Mulher")) +
  xlab("Sexo") +
  ylab("Frequencia") +
  scale_color_manual(values =
                       c("darkorchid4", "brown2"))
```



Aqui plotamos dois gráficos um ao lado do outro. Note que para fazer um boxplot bem elaborado não precisa de muitas camadas. O que pode ser novo aqui são alguns argumentos utilizados dentro do `geom_dotplot()`. Essa camada permite adicionar gráficos de pontos que quando combinado com o boxplot, nos permite uma visualização mais completa dos dados. Você pode dar uma olhada mais aprofundada em todos os argumentos do `geom_dotplot()` clickando aqui.

```
ggplot(dfhair, aes(x = Sex, y = Freq,
                   fill = Sex)) +
  theme_minimal() +
  geom_boxplot(
    outlier.colour = "black",
    outlier.shape = 8,
    outlier.size = 4
  ) +
  scale_x_discrete(labels = c("Homem", "Mulher")) +
```

```
  labs(
    title = "Boxplot",
    x = "Sexo",
    y = "Frequência",
    subtitle = "HairEyeColor"
  ) +
  scale_fill_manual(values =
                      c("darkorchid4", "brown2"))
```



Caso prefira preencher os gráficos com cores da sua escolhar, basta alterar o `color` para `fill` dentro do `aes` e escolher a sua paleta de cores preferida ou até mesmo utilizar `scale_fill_manual` para escolher as cores de sua preferência.

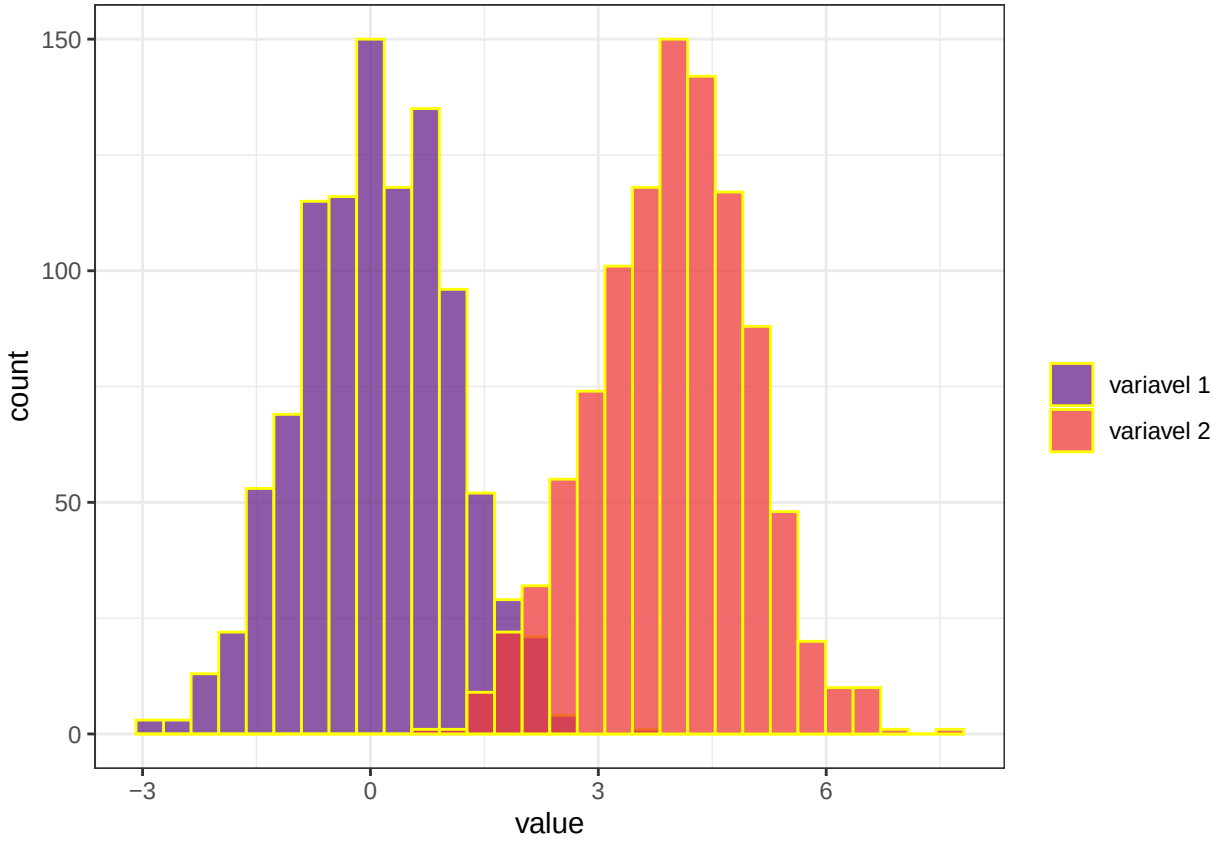
Histogramas um pouco mais elaborados

Fazer histogramas com mais de uma variável para comparação pode ser interessantes em alguns casos. Para o exemplo abaixo vamos criar um data frame simples com duas variáveis geradas a partir da distribuição normal.

```
data <- data.frame(
  type = c( rep("variavel 1", 1000), rep("variavel 2", 1000) ),
  value = c( rnorm(1000), rnorm(1000, mean=4) )
)
```

```r
# Represent it
data %>%
  ggplot( aes(x=value, fill=type)) +
    geom_histogram( color="yellow", alpha=0.75, position = 'identity') +
    scale_fill_manual(values=c("darkorchid4", "brown2")) +
    theme_bw() +
    labs(fill="")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



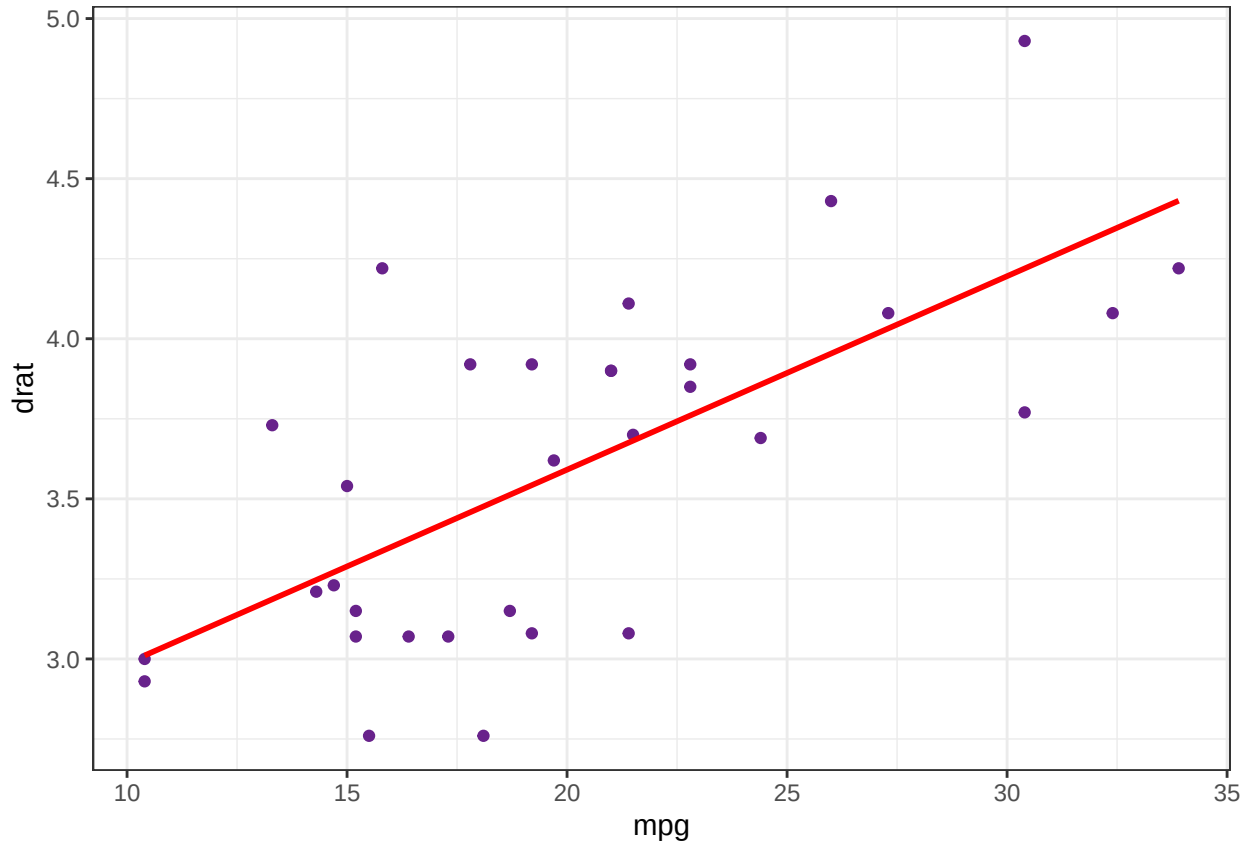
Um argumento legal capaz de nos ajudar na visualização dos gráficos em alguns casos é o `alpha`. O `alpha` se refere à opacidade de um geom. Os valores alphavariam de 0 a 1, com valores mais baixos correspondendo a cores mais transparentes. `color` muda o contorno das barras dos gráficos, também pode nos ajudar na diferenciação dos gráficos quando eles se cruzam.

- *sugestão de ideias*

gráficos de disperção

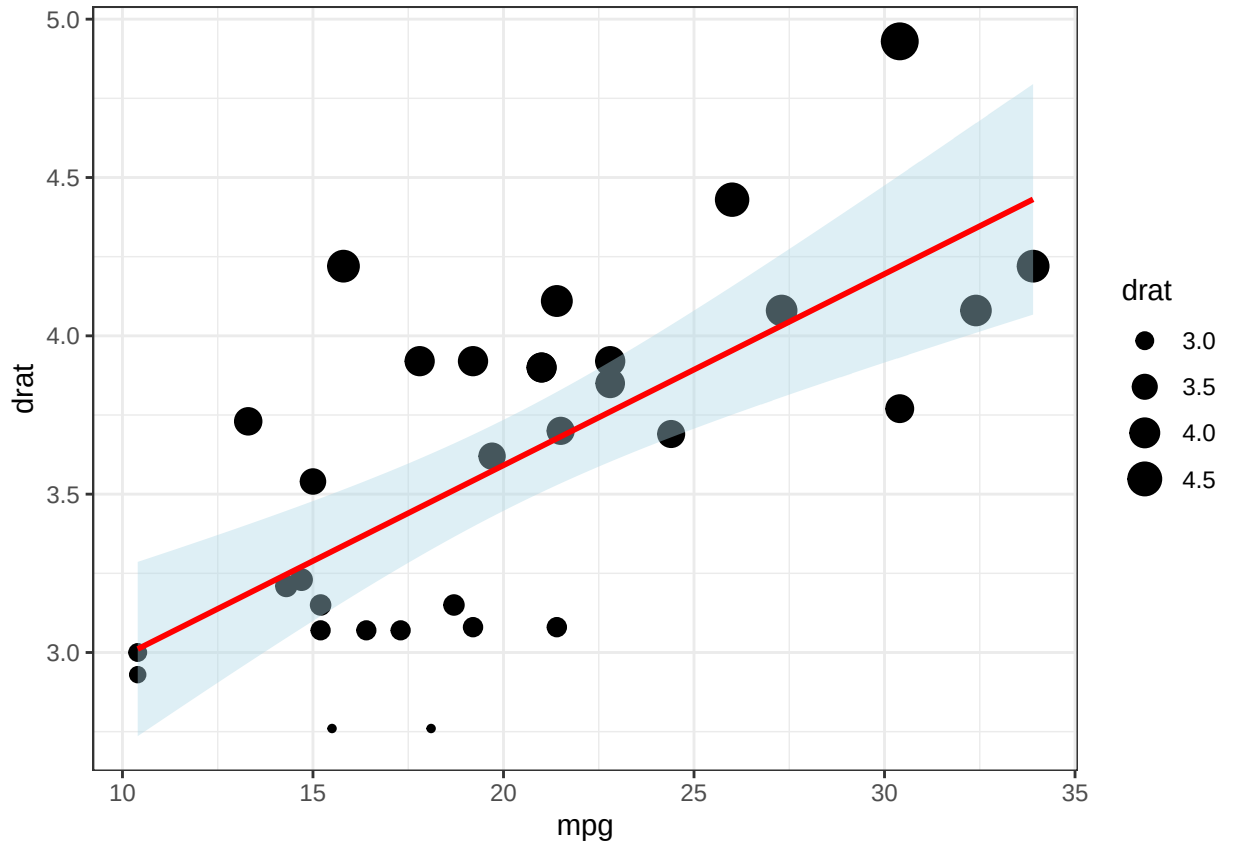
```r
ggplot(mtcars, aes(mpg, drat)) +
  geom_point(color = "darkorchid4") +
  theme_bw() +
  geom_smooth(method = lm, color = "red", se = FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



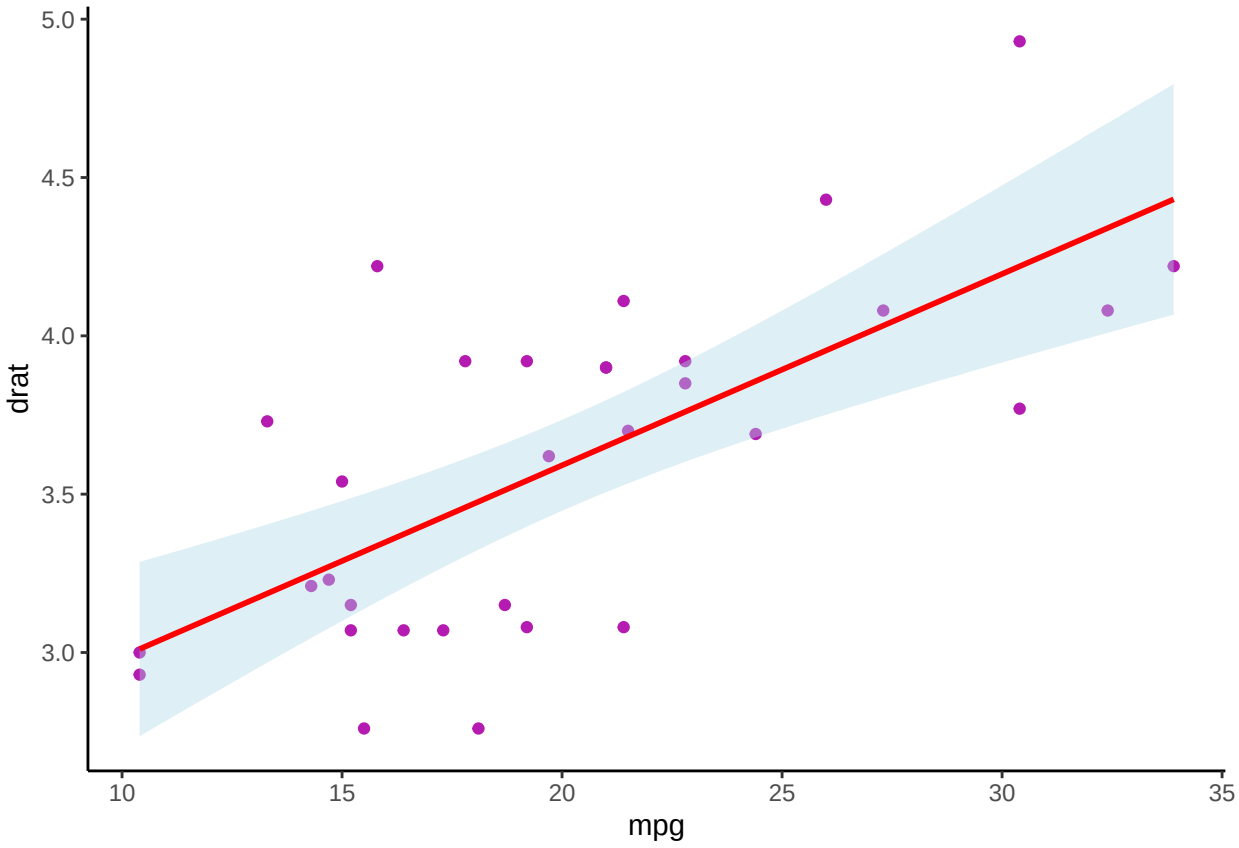

```
ggplot(mtcars, aes(mpg, drat)) +
  geom_point(aes(size = drat)) +
  theme_bw() +
  geom_smooth(method = lm, color="red", se = TRUE, fill = "lightblue")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(mtcars, aes(mpg, drat)) +
  geom_point(color = "#b51bb0") +
  theme_classic() +
  geom_smooth(method = lm, color = "red", se = TRUE, fill = "lightblue")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Outras opções de pacotes para visualização de dados

Referências

https://operdata.com.br/blog/como-interpretar-um-boxplot/

https://www.r-graph-gallery.com/index.html