Comparing Frequency Distributions

No último exercício vimos os tipos de gráfico para cada situação, para cada tipo de dado. Agora vamos ver os gráficos que nos permitem comparar distribuições no mesmo gráfico.

Explorando dados do WNBA

O dataset pode ser acessado através deste link.

E o glossário dos termos neste link.

```
wnba <- read_csv("WNBA_Stats.csv")
```

```
## Rows: 143 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): Name, Team, Pos, Birth_Place, Birthdate, College, Experience
## dbl (25): Height, Weight, BMI, Age, Games Played, MIN, FGM, FGA, FG%, 15:00,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Vamos criar a variável que representa o nível de experiência da jogadora para comparar com a posição em que joga. Como vamos utilizar em gráficos, estamos transformando em factor para poder ordenar uma variável do tipo texto.

```
wnba <- wnba %>%
  mutate(Experience = ifelse(Experience == "R", 0, as.numeric(Experience))) %>%
  mutate(Experience_level =
           case_when(Experience == 0 ~ "Rookie",
                     Experience >= 1 & Experience <= 3  ~ "Little Experience",
                     Experience >= 4 & Experience <= 5  ~ "Experienced",
                     Experience >= 6 & Experience <= 10 ~ "Very Experienced",
                     Experience > 10 ~ "Veteran"
```

```
                    )) %>%
  mutate(Experience_level = factor(Experience_level,
                                   levels=c("Rookie",
                                            "Little Experience",
                                            "Experienced",
                                            "Very Experienced",
                                            "Veteran")))
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `Experience = ifelse(Experience == "R", 0,
##   as.numeric(Experience))`.
## Caused by warning in `ifelse()`:
## ! NAs introduzidos por coerção
```

```
wnba$Experience_level %>% table(useNA = "always") %>% as.data.frame()
```

```
##                    . Freq
## 1             Rookie   23
## 2 Little Experience   42
## 3       Experienced   25
## 4  Very Experienced   37
## 5           Veteran   16
## 6              <NA>    0
```

Uma alternativa é criar a tabela frequência, porém dessa vez informando duas variáveis no agrupamento. Afinal queremos comparar o nível de experiência à posição que a jogadora pertence.

```
exp_by_pos <-
wnba %>%
  group_by(Experience_level,Pos) %>%
  summarise(Freq = n())
```

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

```
exp_by_pos
```

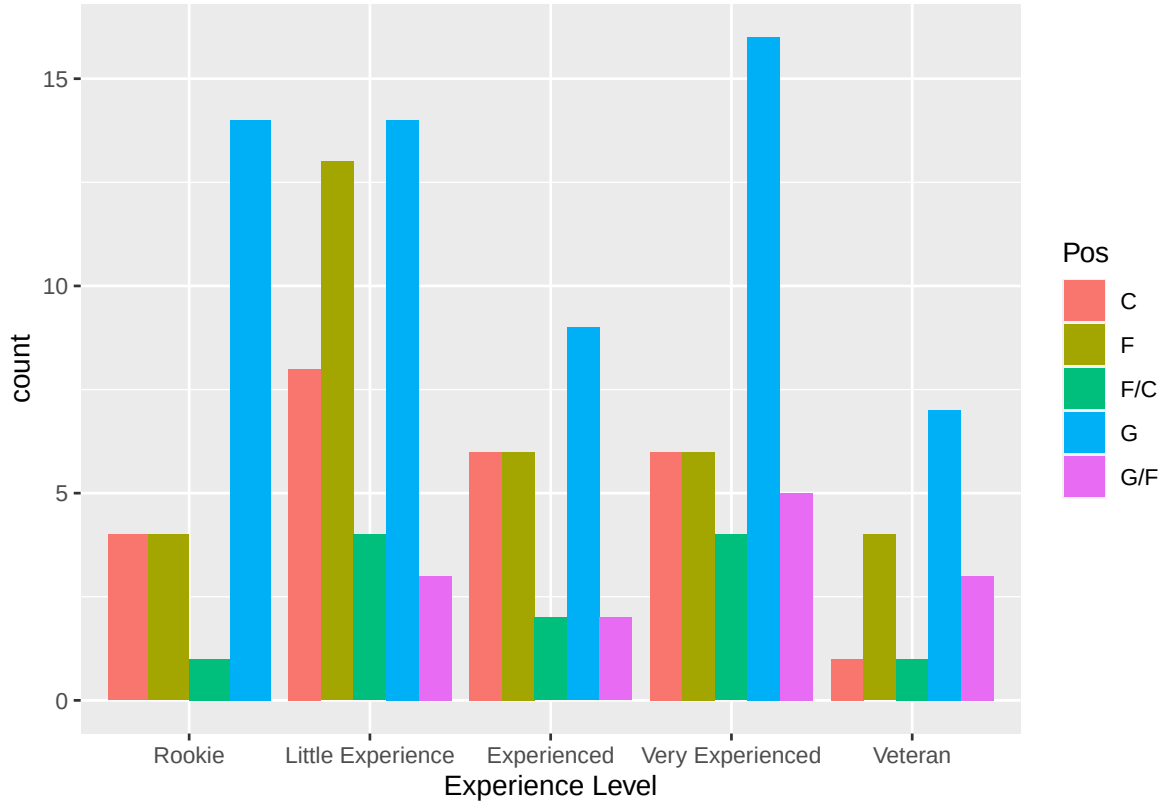
```
## # A tibble: 24 x 3
## # Groups:   Experience_level [5]
##    Experience_level  Pos    Freq
##    <fct>             <chr> <int>
##  1 Rookie            C         4
##  2 Rookie            F         4
##  3 Rookie            F/C       1
##  4 Rookie            G        14
##  5 Little Experience C         8
##  6 Little Experience F        13
##  7 Little Experience F/C       4
##  8 Little Experience G        14
##  9 Little Experience G/F       3
## 10 Experienced       C         6
## # i 14 more rows
```

Gráfico de Barras Agrupado

Podemos de forma mais simples e visual fazer essa comparação. Podemos fazer de duas formas, deixar o próprio gráfico criar a tabela de distribuição e exibir.

position = “dodge” informa que queremos as barras lado a lado e não empilhadas (stacked)

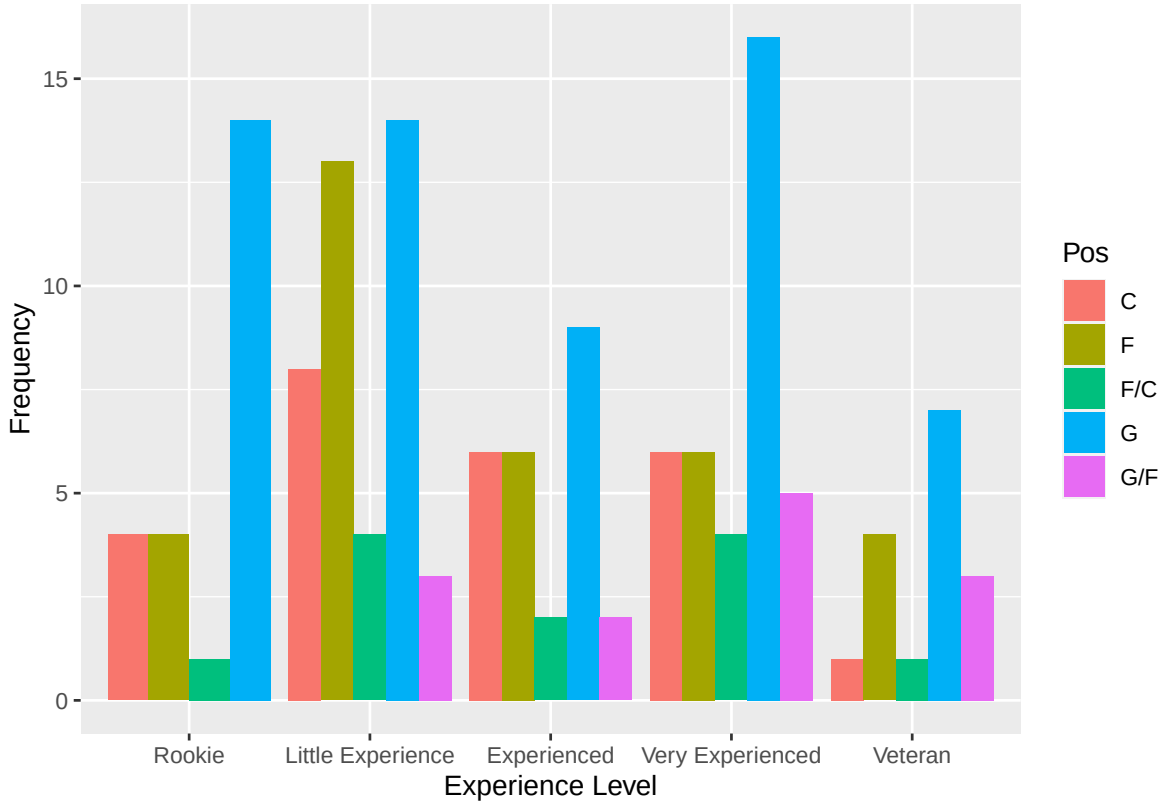
```
wnba %>%
  ggplot(aes(x=Experience_level, fill=Pos)) +
  geom_bar(position = "dodge") +
  labs(x="Experience Level")
```



Ou nós podemos usar a tabela criada previamente e teremos o mesmo resultado.

stat = “identity” informa que o gráfico não deve fazer o “count”, pois já estamos informando um objeto sumarizado.

```
exp_by_pos %>%
  ggplot(aes(x=Experience_level, y=Freq, fill=Pos))+
  geom_bar(position = "dodge", stat = "identity")+
  labs(x="Experience Level",
       y="Frequency")
```



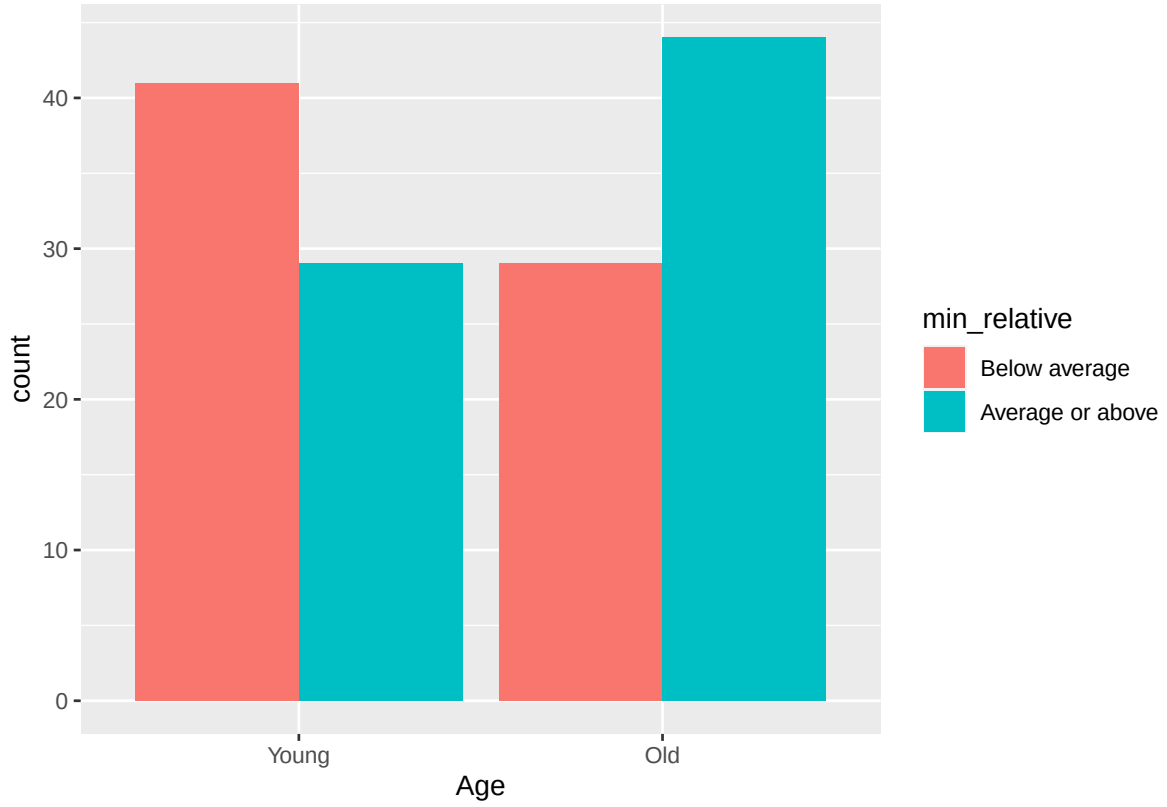
Agora vamos levantar uma hipótese de que jogadoras mais velhas tendem a participar menos dos jogos. Para confirmar essa hipótese podemos gerar um gráfico de barras que separam as jogadoras mais novas das mais velhas e o tempo que jogaram ao longo da temporada.

```
wnba <- wnba %>%
  mutate(age_relative = ifelse(Age >= 27, "Old", "Young"),
         min_relative = ifelse(MIN >= 497, "Average or above", "Below average")) %>%
  mutate(age_relative = factor(age_relative,
                               levels = c( "Young", "Old")),
         min_relative = factor(min_relative,
                               levels = c("Below average","Average or above")),
         )
```

```
wnba %>% select(age_relative,min_relative) %>% table()
```

```
##             min_relative
## age_relative Below average Average or above
##        Young            41                29
##        Old              29                44
```

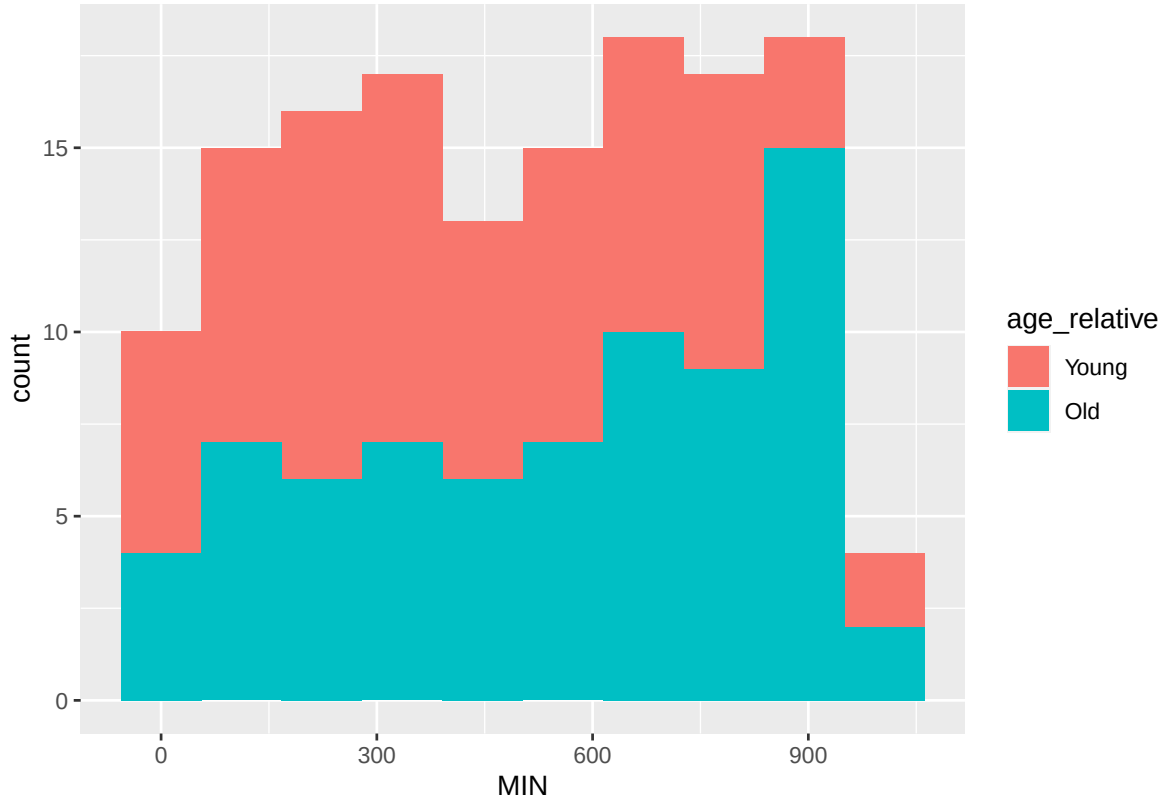
```
wnba %>%
  ggplot(aes(x=age_relative, fill=min_relative)) +
  geom_bar(position = "dodge") +
  labs(x="Age")
```



O único problema do gráfico acima é que perdemos a granularidade da informação, e portanto não conseguimos analisar o quanto as jogadoras mais velhas ficam mais tempo nas partidas, pode ser o caso de a maioria ficar na média, ou então ultrapassar muito a média.

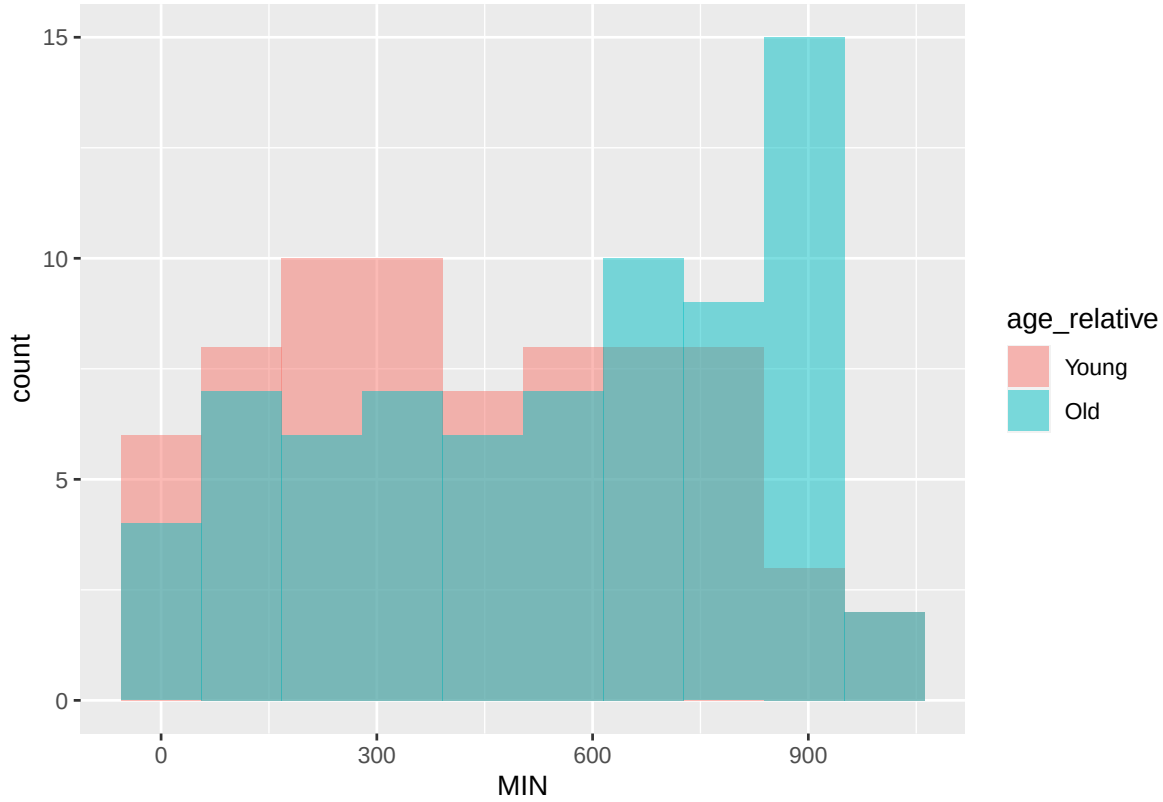
Uma forma de enxergar isso é com um histograma

```
wnba %>%
  ggplot(aes(x=MIN, fill=age_relative))+
  geom_histogram(bins=10)
```



No entanto o histrograma empilhado dificulta o entendimento, então precisamos de um pequeno ajuste. O position = “identity” indica que não deve empilhar, e o alpha = 0.5 mexe na transparência para ser possível enxergar ambas cores devido a sobreposição.

```
wnba %>%
  ggplot(aes(x=MIN, fill=age_relative))+
  geom_histogram(bins=10,
                 position = "identity",
                 alpha = 0.5)
```

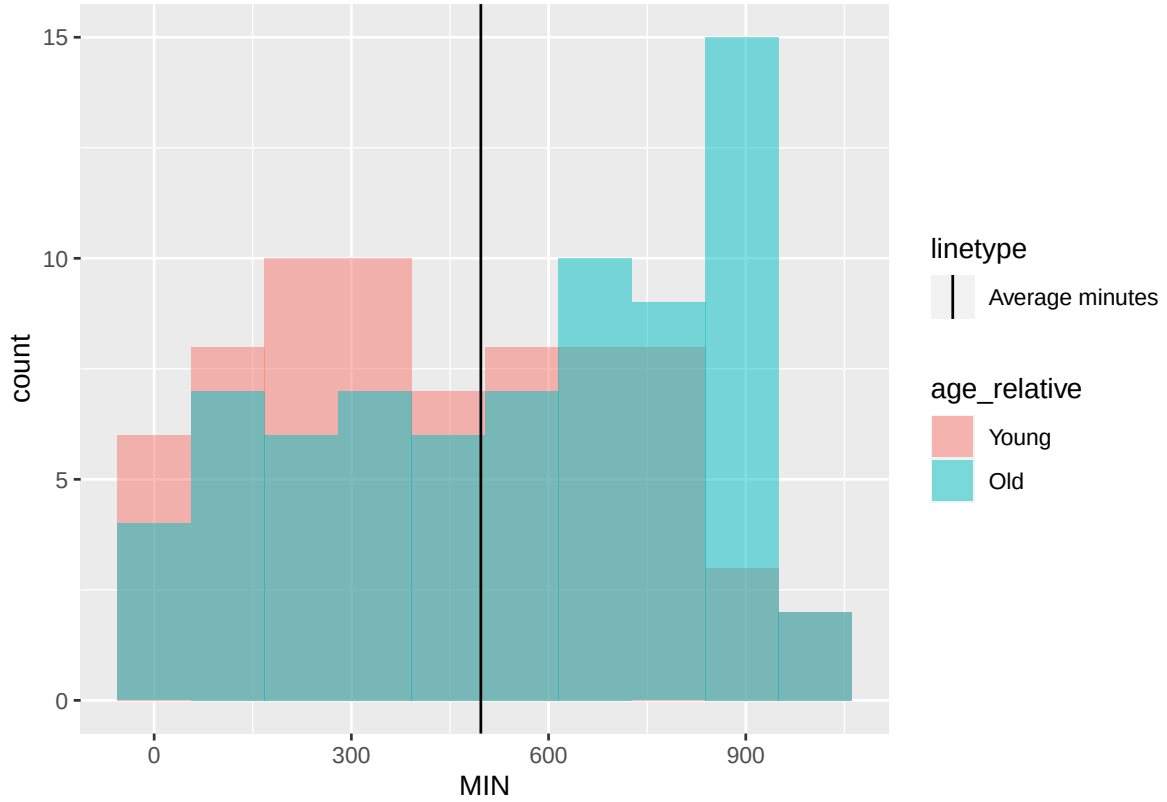


Agora sim podemos constatar que as jogadoras mais velhas que estão na categoria de “Average or above” na verdade estão muito acima da média.

Para enfatizar ainda mais o quanto essas jogadoras estão acima da média podemos adicionar uma barra vertical que represente a média

```
wnba %>%
  ggplot(aes(x=MIN, fill=age_relative))+
  geom_histogram(bins=10,
                 position = "identity",
                 alpha = 0.5)+
  geom_vline(aes(xintercept = mean(wnba$MIN),
                 linetype = "Average minutes"),
             color = "black")
```

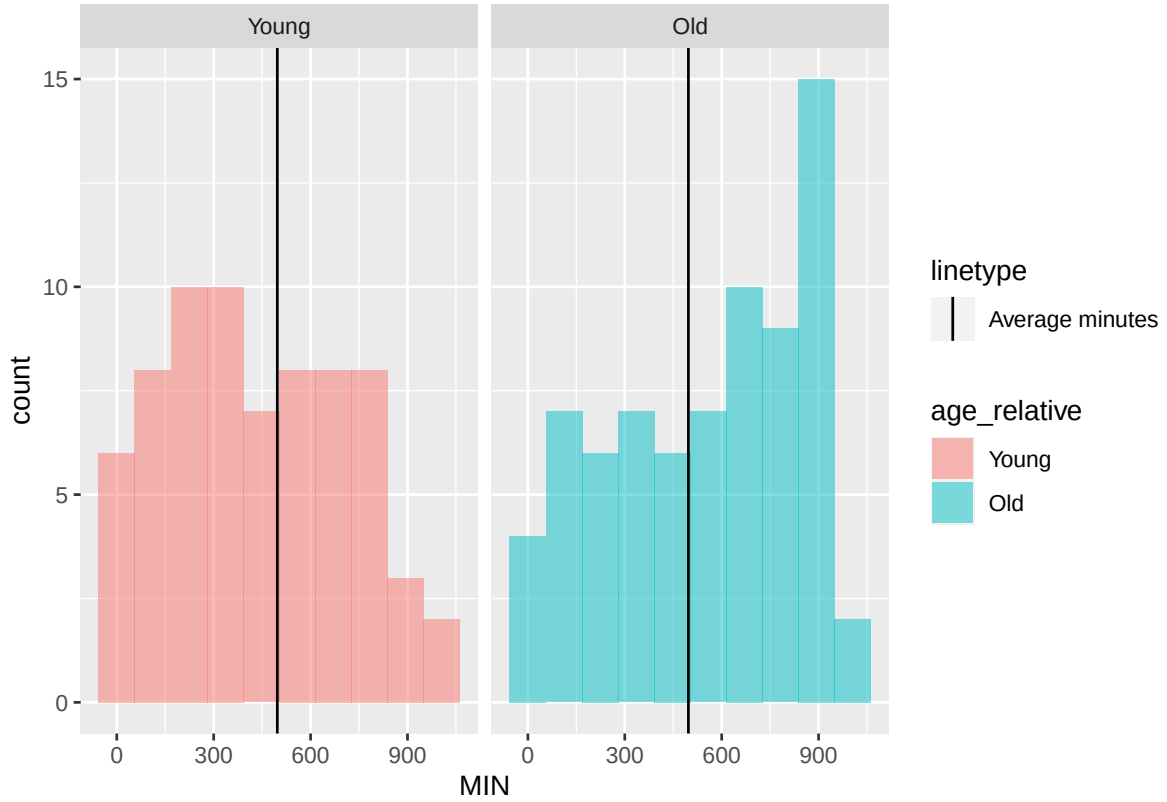
```
## Warning: Use of `wnba$MIN` is discouraged.
## i Use `MIN` instead.
```



Outra alternativa para comparar a distribuição além de usar sobreposição é comparar gráficos lado a lado usando face_wrap.

```
wnba %>%
  ggplot(aes(x=MIN, fill=age_relative))+
  geom_histogram(bins=10,
                 position = "identity",
                 alpha = 0.5)+
  geom_vline(aes(xintercept = mean(wnba$MIN),
                 linetype = "Average minutes"),
             color = "black") +
  facet_wrap(~age_relative)
```

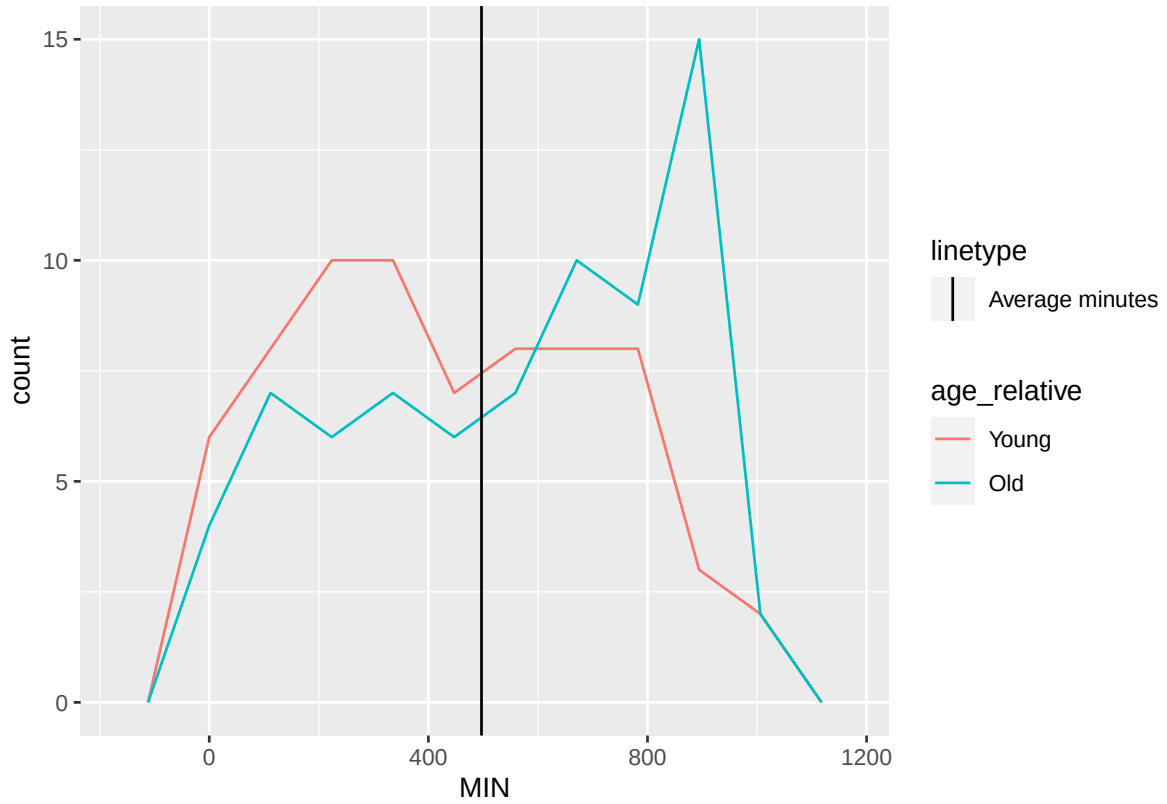
```
## Warning: Use of `wnba$MIN` is discouraged.
## i Use `MIN` instead.
```



Se nossa intenção adicionar mais categorias para comparar, ficaria quase impossível de entender. Pensando nisso podemos converter as barras em linhas que facilitam enxergar esse comparativo.

```
wnba %>%
  ggplot(aes(x=MIN, color=age_relative))+
  geom_freqpoly(bins=10,
                position = "identity")+
  geom_vline(aes(xintercept = mean(wnba$MIN),
                linetype = "Average minutes"),
             color = "black")
```

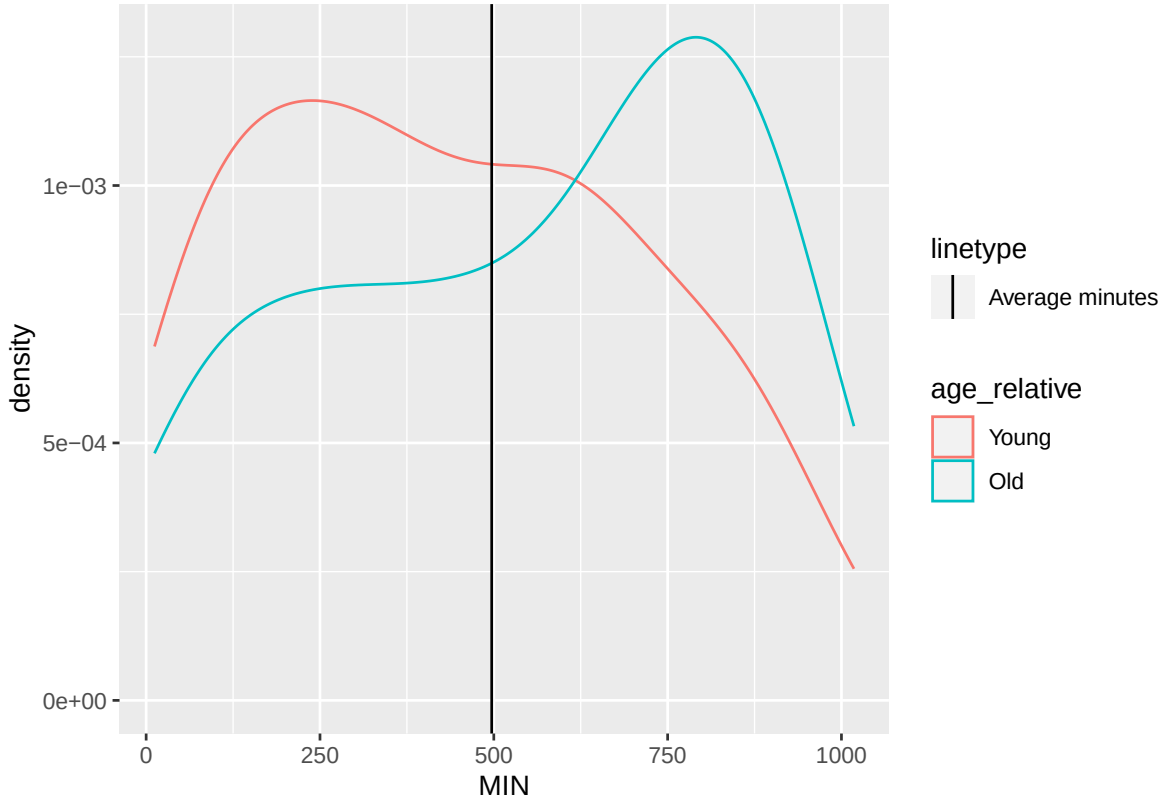
```
## Warning: Use of `wnba$MIN` is discouraged.
## i Use `MIN` instead.
```



o geom_freqpoly deixa as linhas mais retas, para suavizar esse efeito podemos usar a função geom_density. Nessa função o position = “identity” é o padrão e portanto não precisa ser informado

```
wnba %>%
  ggplot(aes(x=MIN, color=age_relative))+
  geom_density()+
  geom_vline(aes(xintercept = mean(wnba$MIN),
                 linetype = "Average minutes"),
             color = "black")
```

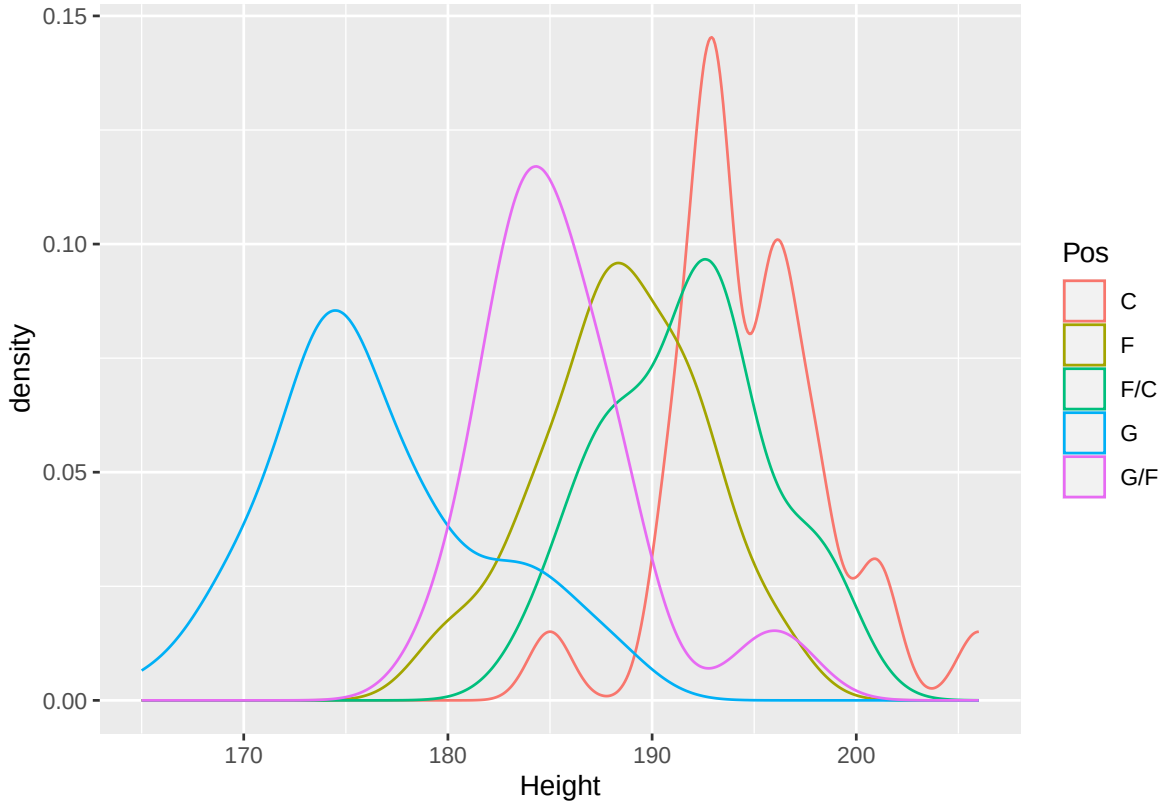
```
## Warning: Use of `wnba$MIN` is discouraged.
## i Use `MIN` instead.
```



A grande diferença desse gráfico é que ele se baseia na densidade e não na frequência dos dados. São valores de probabilidade, e são úteis quando não queremos os valores exatos e sim tem uma ideia da distribuição dos dados.

Até aqui aprendemos que para variáveis nominais e ordinais o ideal é usarmos gráficos de barra, enquanto que para razão e intervalos os gráficos de densidade são os ideais, porém quando temos 5 ou mais variáveis começa a dificultar o entendimento.

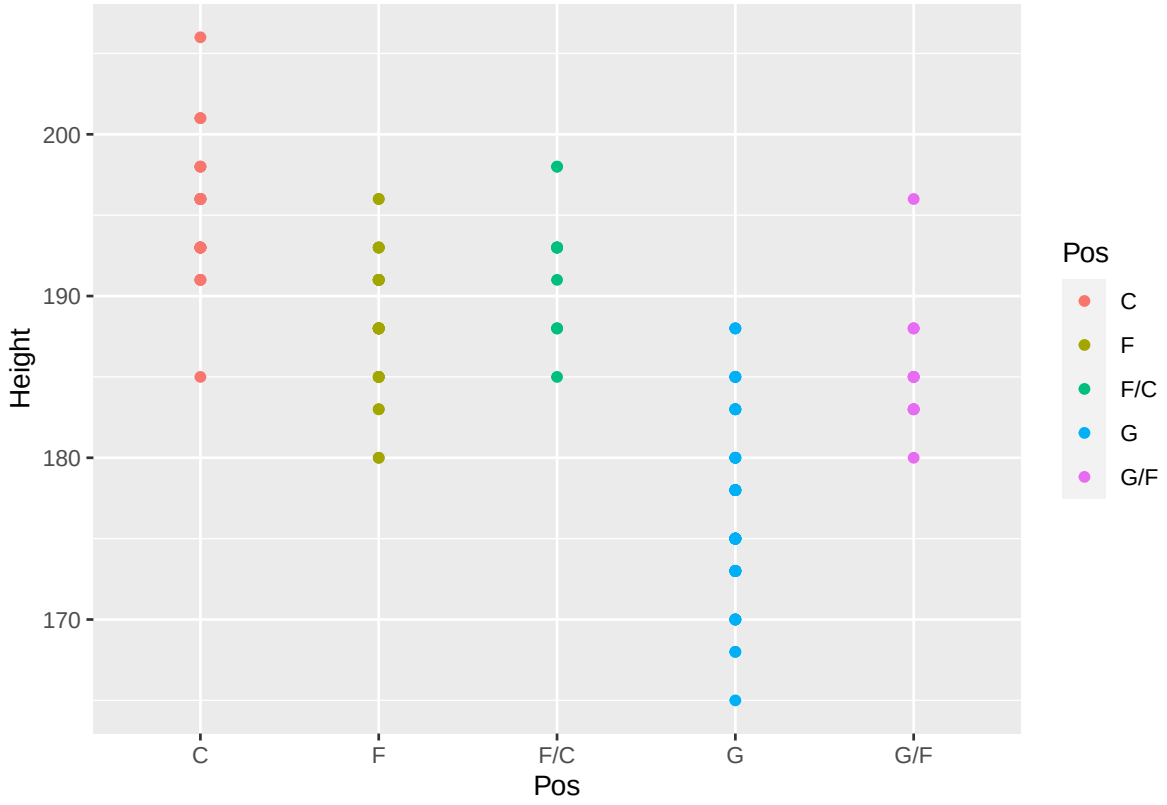
```
wnba %>%
ggplot(aes(x = Height, color = Pos)) +
  geom_density()
```



Para resolver essa necessidade podemos utilizar os scatter plots.

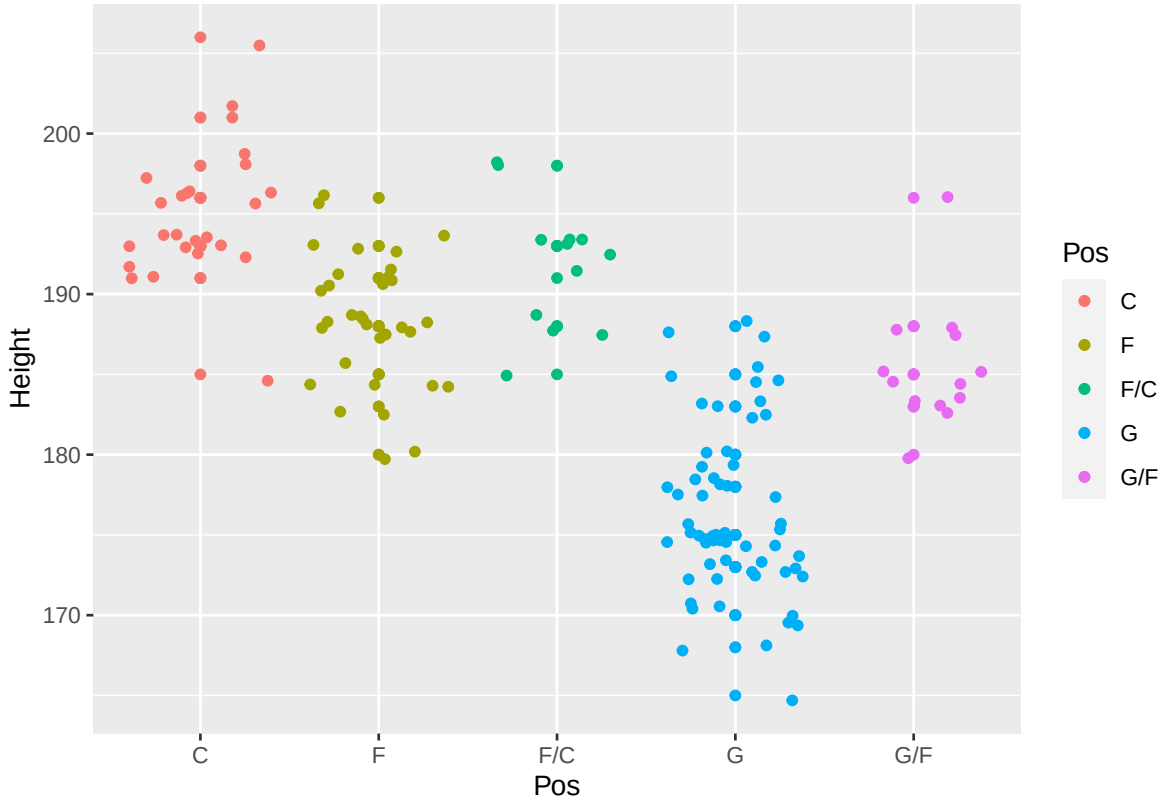
Nesse gráfico usamos a posição das jogadoras para segmentar o gráfico e como as jogadoras só podem ser de uma dos cinco tipos de posições (não há meio termo) cria-se 5 linhas imaginárias e os pontos apenas se movimentam com relação a altura de cada jogadora ao longo do eixo y. Por isso esse tipo de gráfico nessa situação fica conhecido como Strip Plot.

```
wnba %>%
  ggplot(aes(x = Pos, y = Height, color = Pos)) +
  geom_point()
```



Uma desvantagem é que os pontos estão sobrepostos uns dos outros, uma forma de resolver isso é com o jitter que cria um ruído aleatório em cada ponto para mover um pouquinho do lugar e assim conseguimos enxergar.

```
wnba %>%
  ggplot(aes(x = Pos, y = Height, color = Pos)) +
  geom_point() +
  geom_jitter()
```

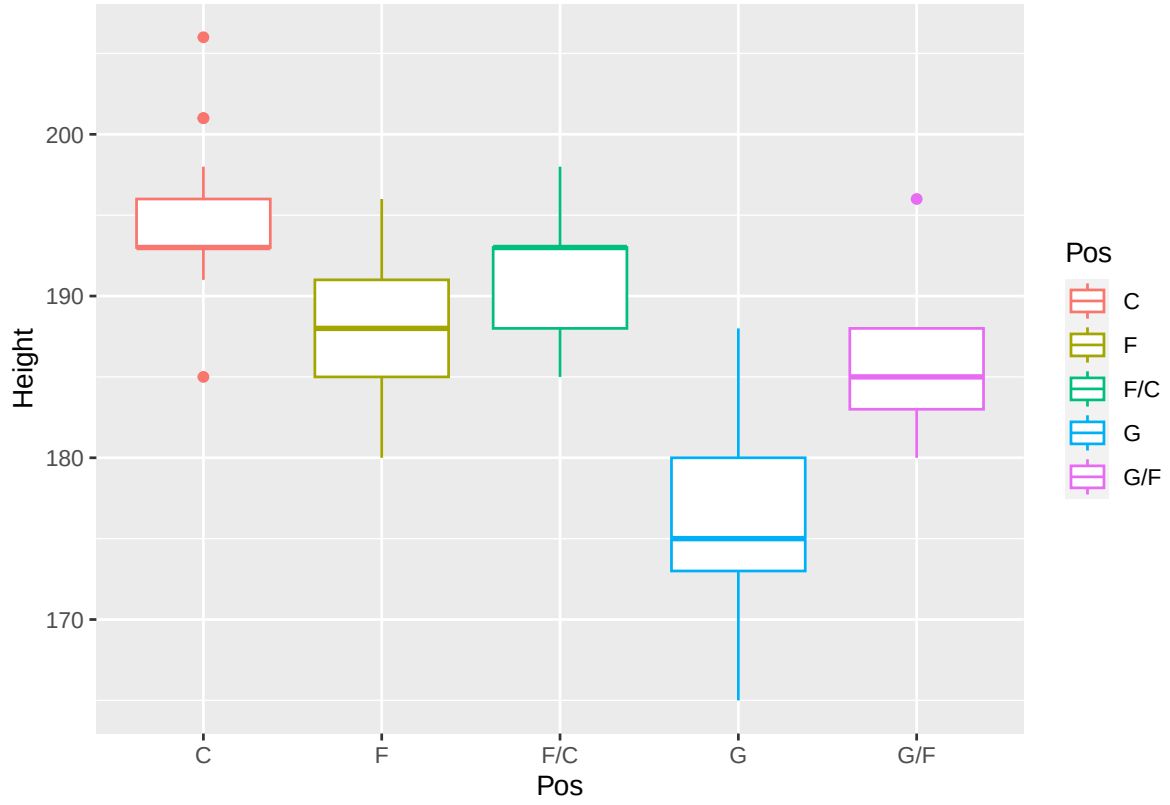


Existe outra alternativa de gráfico para visualizar distribuição de informações contínuas que tem uma visualização fácil de interpretar. Podemos usar os Boxplots.

Ele é mais completo em informações:

- traz os 3 quartis representados pelo retângulo cortado ao meio: a linha que divide o retângulo é a mediana, e o topo e o fundo do retangulo é o primeiro e terceiro quartil.
- outliers: são valores extremos que se diferem muito do restante dos dados representados por bolinhas, estão muito abaixo do mínimo e máximo esperado.
- mínimo e máximo: são representados pelos extremos das linhas abaixo e acima do retangulo, desconsideram os outliers.

```
wnba %>%
  ggplot(aes(x = Pos, y = Height, color = Pos)) +
  geom_boxplot()
```



A regra de como o gráfico classifica um outlier se dá da seguinte forma:

- é feito o cálculo do interquartil (terceiro quartil - primeiro quartil)
- multiplica o resultado por 1.5
- assim somamos esse resultado ao terceiro quartil enquanto que subtraímos do primeiro quartil
- e assim teremos o valor limite que um valor pode ter antes de ser considerado outlier

Vamos a um exemplo, vamos selecionar as jogadoras na posição Centers (C). Usando a função summary, identificamos com maior precisão o primeiro e terceiro quartil.

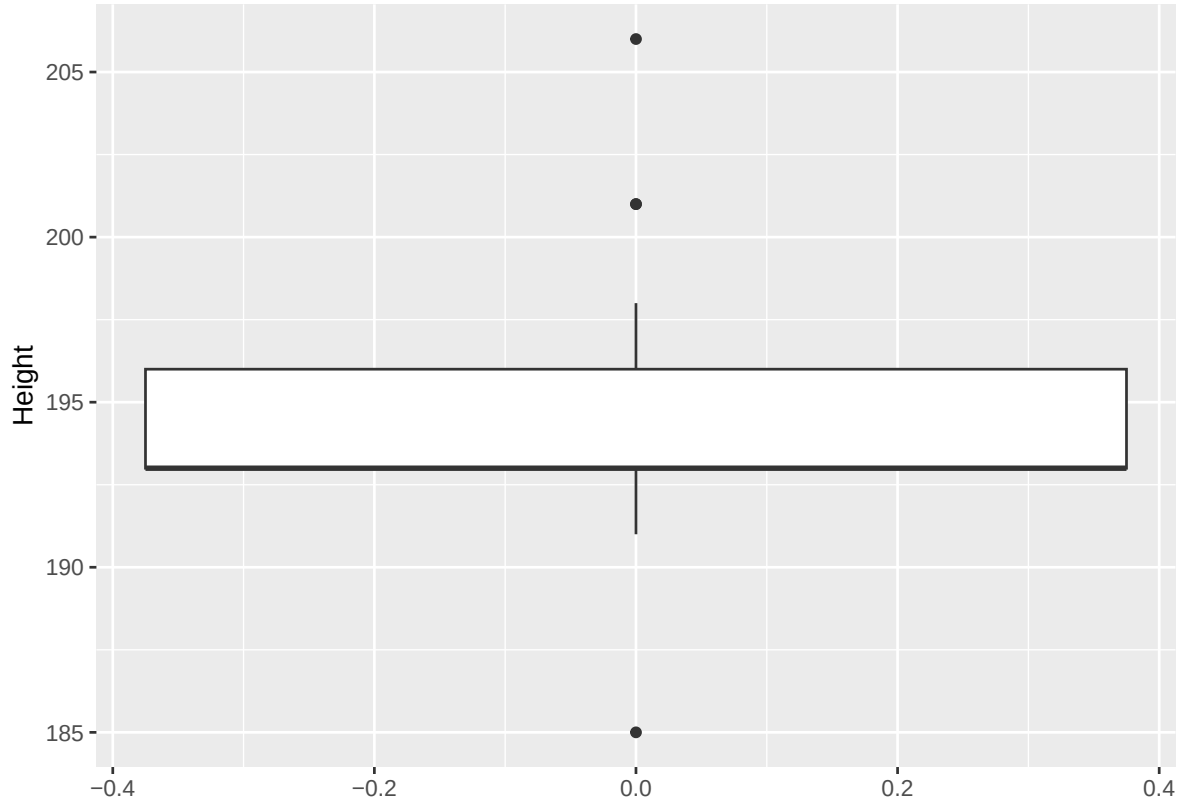
```
centers <-
  wnba %>%
  filter(Pos == "C")

centers$Height %>% summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   185.0   193.0   193.0   194.9   196.0   206.0
```

- Sendo assim o cálculo Interquartil fica 196 - 193 = 3
- O cálculo para considerar o outlier fica 3 x 1.5 = 4.5
- Sendo assim o calculo para o limite no valor máximo fica 196 + 4 = 200.5
- Enquanto que no valor mínimo fica 193 - 4.5 = 188.5

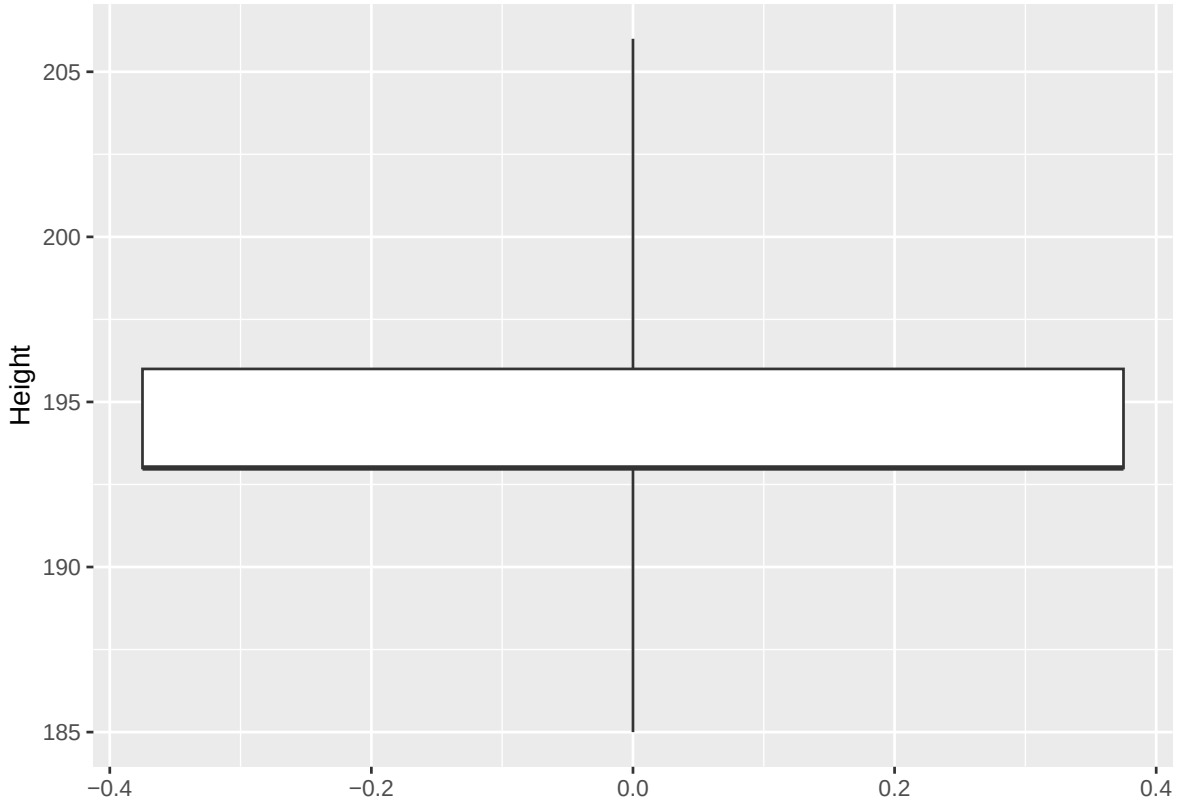
Então os limites mínimo e máximo que o valor pode assumir antes de ser considerado um outlier são 188.5 e 200.5. O que ultrapassar esses valores será representado por bolinhas no gráfico.

```
centers %>%
ggplot(aes(y = Height)) +
  geom_boxplot()
```



Esse coeficiente de 1.5 para calcular os outliers é o valor padrão utilizado na função, no entanto temos a flexibilidade de alterar quando quisermos. Se aumentarmos, provavelmente valores considerados outliers podem deixar de ser.

```
centers %>%
ggplot(aes(y = Height)) +
  geom_boxplot(coef = 4)
```


Height
205
200
195
190
185
−0.4
−0.2
0.0
0.2
0.4