Solução Lista 03

Nome: Vinicius de Oliveira Bezerra
E-mail: v.bezerra@aluno.ufabc.edu.br
Nome: Deyved Kevyn Alves Lima
E-mail: deyved.lima@aluno.ufabc.edu.br

11 March, 2025

Solução Exercício 01

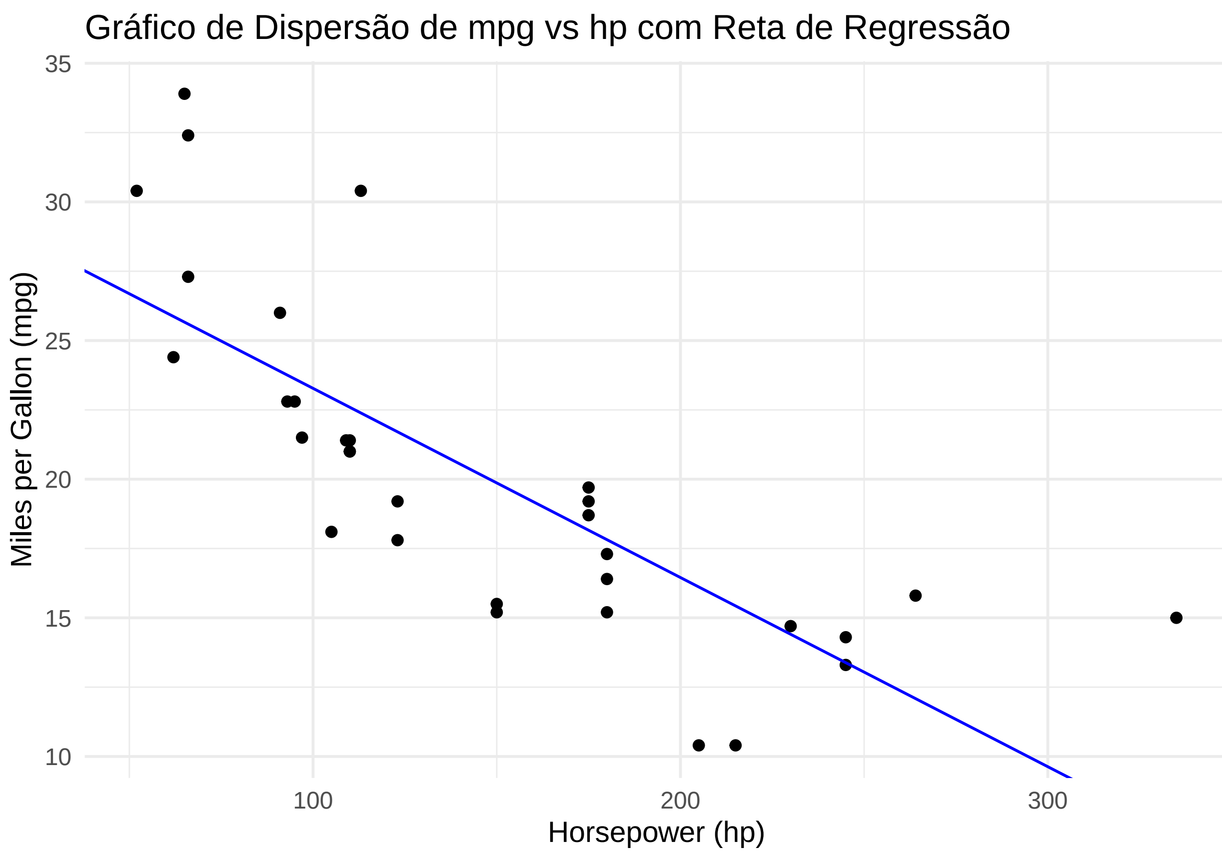
```r
#Importações
library(tidymodels)
library(ggplot2)
library(car)

#Carregar o banco de dados
df = as_tibble(mtcars)

#Regressão linear
lin.model = lm(mpg ~ hp, data = df)
summary(lin.model) #Detalhes do modelo de regressão linear
```

```
##
## Call:
## lm(formula = mpg ~ hp, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.7121 -2.1122 -0.8854  1.5819  8.2360
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.09886    1.63392  18.421  < 2e-16 ***
## hp          -0.06823    0.01012  -6.742 1.79e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.863 on 30 degrees of freedom
## Multiple R-squared:  0.6024, Adjusted R-squared:  0.5892
## F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07
```

O modelo de regressão linear usando hp como preditor para mpg mostrou-se estatisticamente significativo, com um valor-p muito baixo (1.79e-07) para o preditor hp. O coeficiente de -0.06823 indica que, para cada aumento de uma unidade em hp, o valor de mpg diminui em aproximadamente 0.06823. O R^2 de

0.6024 sugere que aproximadamente 60% da variabilidade em mpg é explicada por hp, indicando um ajuste razoavelmente bom. O erro padrão dos resíduos é 3.863, e a distribuição dos resíduos parece simétrica. Em resumo, hp é um preditor importante para mpg, mas outros fatores podem ser considerados para melhorar o modelo.

```r
#Gerar o gráfico de dispersão
intercept = coef(lin.model)[1] #Interceptor
slope = coef(lin.model)[2] #Inclinação

ggplot(df, aes(x = hp, y = mpg)) +
  geom_point() +  # Adicionar os pontos de dispersão
  geom_abline(intercept = intercept, slope = slope, color = "blue") +  # Adicionar a reta de regressão
  labs(title = "Gráfico de Dispersão de mpg vs hp com Reta de Regressão",
       x = "Horsepower (hp)",
       y = "Miles per Gallon (mpg)") +
  theme_minimal()
```




```r
#Novo modelo de regressão linear
lin.model.new = lm(mpg ~ ., data = df)

#Verificar o resumo do modelo
summary(lin.model.new)
```

```
##
## Call:
```

```
## lm(formula = mpg ~ ., data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4506 -1.6044 -0.1196  1.2193  4.6271
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12.30337   18.71788   0.657   0.5181
## cyl         -0.11144    1.04502  -0.107   0.9161
## disp         0.01334    0.01786   0.747   0.4635
## hp          -0.02148    0.02177  -0.987   0.3350
## drat         0.78711    1.63537   0.481   0.6353
## wt          -3.71530    1.89441  -1.961   0.0633 .
## qsec         0.82104    0.73084   1.123   0.2739
## vs           0.31776    2.10451   0.151   0.8814
## am           2.52023    2.05665   1.225   0.2340
## gear         0.65541    1.49326   0.439   0.6652
## carb        -0.19942    0.82875  -0.241   0.8122
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.65 on 21 degrees of freedom
## Multiple R-squared:  0.869,  Adjusted R-squared:  0.8066
## F-statistic: 13.93 on 10 and 21 DF,  p-value: 3.793e-07
```

```r
#Calcular o fator de inflação de variância (VIF)
vif(lin.model.new)
```

```
##       cyl      disp        hp      drat        wt      qsec        vs        am
## 15.373833 21.620241  9.832037  3.374620 15.164887  7.527958  4.965873  4.648487
##      gear      carb
##  5.357452  7.908747
```

A análise do modelo de regressão linear múltipla usando todos os preditores do banco de dados mtcars para prever mpg revelou alguns insights importantes. No modelo anterior, onde apenas hp era usado como preditor, hp era altamente significativo e explicava cerca de 60% da variabilidade em mpg. No entanto, ao incluir todos os preditores no modelo múltiplo, a importância de hp diminuiu significativamente, com um valor-p de 0,335, indicando que ele não é mais estatisticamente significativo. Isso ocorre porque outros preditores, como cyl, disp e wt, estão capturando parte da variabilidade que hp explicava anteriormente, devido à colinearidade entre as variáveis.

A qualidade geral do modelo é boa, com um R^2 de 0,869, indicando que 86,9% da variabilidade em mpg é explicada pelos preditores incluídos. No entanto, o R^2 ajustado de 0,8066 sugere que alguns preditores podem não estar contribuindo significativamente para o modelo. A análise dos fatores de inflação de variância (VIF) mostrou que há colinearidade entre os preditores, especialmente para cyl, disp e wt, que têm VIFs altos. Isso inflaciona os erros padrão dos coeficientes e reduz a significância estatística dos preditores.

Solução Exercício 02

```r
library(tidyverse)
library(car)
library(stringr)

# Carregar os Dados
file_url <- "https://drive.google.com/uc?export=download&id=1jiWcGsl_tbqK5F0ryUTq48kcDTKWTTuk"
df_orign <- read.csv(file_url) %>% as_tibble()

# Visualizar primeiras linhas
glimpse(df_orign)
```

```
## Rows: 17,981
## Columns: 75
## $ X                    <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, ~
## $ Name                 <chr> "Cristiano Ronaldo", "L. Messi", "Neymar", "L. Suá~
## $ Age                  <int> 32, 30, 25, 30, 31, 28, 26, 26, 27, 29, 31, 26, 25~
## $ Photo                <chr> "https://cdn.sofifa.org/48/18/players/20801.png", ~
## $ Nationality          <chr> "Portugal", "Argentina", "Brazil", "Uruguay", "Ger~
## $ Flag                 <chr> "https://cdn.sofifa.org/flags/38.png", "https://cd~
## $ Overall              <int> 94, 93, 92, 92, 92, 91, 90, 90, 90, 90, 90, 89, 89~
## $ Potential            <int> 94, 93, 94, 92, 92, 91, 92, 91, 90, 90, 90, 92, 92~
## $ Club                 <chr> "Real Madrid CF", "FC Barcelona", "Paris Saint-Ger~
## $ Club.Logo            <chr> "https://cdn.sofifa.org/24/18/teams/243.png", "htt~
## $ Value                <chr> "€95.5M", "€105M", "€123M", "€97M", "€61M", "€92M"~
## $ Wage                 <chr> "€565K", "€565K", "€280K", "€510K", "€230K", "€355~
## $ Special              <int> 2228, 2154, 2100, 2291, 1493, 2143, 1458, 2096, 21~
## $ Acceleration         <chr> "89", "92", "94", "88", "58", "79", "57", "93", "6~
## $ Aggression           <chr> "63", "48", "56", "78", "29", "80", "38", "54", "6~
## $ Agility              <chr> "89", "90", "96", "86", "52", "78", "60", "93", "7~
## $ Balance              <chr> "63", "95", "82", "60", "35", "80", "43", "91", "6~
## $ Ball.control         <chr> "93", "95", "95", "91", "48", "89", "42", "92", "8~
## $ Composure            <chr> "95", "96", "92", "83", "70", "87", "64", "87", "8~
## $ Crossing             <chr> "85", "77", "75", "77", "15", "62", "17", "80", "8~
## $ Curve                <chr> "81", "89", "81", "86", "14", "77", "21", "82", "8~
## $ Dribbling            <chr> "91", "97", "96", "86", "30", "85", "18", "93", "7~
## $ Finishing            <chr> "94", "95", "89", "94", "13", "91", "13", "83", "7~
## $ Free.kick.accuracy   <chr> "76", "90", "84", "84", "11", "84", "19", "79", "8~
## $ GK.diving            <chr> "7", "6", "9", "27", "91", "15", "90", "11", "10",~
## $ GK.handling          <chr> "11", "11", "9", "25", "90", "6", "85", "12", "11"~
## $ GK.kicking           <chr> "15", "15", "15", "31", "95", "12", "87", "6", "13~
## $ GK.positioning       <chr> "14", "14", "15", "33", "91", "8", "86", "8", "7",~
## $ GK.reflexes          <chr> "11", "8", "11", "37", "89", "10", "90", "8", "10"~
## $ Heading.accuracy     <chr> "88", "71", "62", "77", "25", "85", "21", "57", "5~
## $ Interceptions        <chr> "29", "22", "36", "41", "30", "39", "30", "41", "8~
## $ Jumping              <chr> "95", "68", "61", "69", "78", "84", "67", "59", "3~
## $ Long.passing         <chr> "77", "87", "75", "64", "59", "65", "51", "81", "9~
## $ Long.shots           <chr> "92", "88", "77", "86", "16", "83", "12", "82", "9~
## $ Marking              <chr> "22", "13", "21", "30", "10", "25", "13", "25", "6~
## $ Penalties            <chr> "85", "74", "81", "85", "47", "81", "40", "86", "7~
## $ Positioning          <chr> "95", "93", "90", "92", "12", "91", "12", "85", "7~
## $ Reactions            <chr> "96", "95", "88", "93", "85", "91", "88", "85", "8~
## $ Short.passing        <chr> "83", "88", "81", "83", "55", "83", "50", "86", "9~
## $ Shot.power           <chr> "94", "85", "80", "87", "25", "88", "31", "79", "8~
```

```
## $ Sliding.tackle      <chr> "23", "26", "33", "38", "11", "19", "13", "22", "6~
## $ Sprint.speed        <chr> "91", "87", "90", "77", "61", "83", "58", "87", "5~
## $ Stamina             <chr> "92", "73", "78", "89", "44", "79", "40", "79", "7~
## $ Standing.tackle     <chr> "31", "28", "24", "45", "10", "42", "21", "27", "8~
## $ Strength            <chr> "80", "59", "53", "80", "83", "84", "64", "65", "7~
## $ Vision              <chr> "85", "90", "80", "84", "70", "78", "68", "86", "8~
## $ Volleys             <chr> "88", "85", "83", "88", "11", "87", "13", "79", "8~
## $ CAM                 <dbl> 89, 92, 88, 87, NA, 84, NA, 88, 83, 81, 70, 86, NA~
## $ CB                  <dbl> 53, 45, 46, 58, NA, 57, NA, 47, 72, 46, 87, 57, NA~
## $ CDM                 <dbl> 62, 59, 59, 65, NA, 62, NA, 61, 82, 52, 83, 70, NA~
## $ CF                  <dbl> 91, 92, 88, 88, NA, 87, NA, 87, 81, 84, 70, 85, NA~
## $ CM                  <dbl> 82, 84, 79, 80, NA, 78, NA, 81, 87, 71, 74, 84, NA~
## $ ID                  <int> 20801, 158023, 190871, 176580, 167495, 188545, 193~
## $ LAM                 <dbl> 89, 92, 88, 87, NA, 84, NA, 88, 83, 81, 70, 86, NA~
## $ LB                  <dbl> 61, 57, 59, 64, NA, 58, NA, 59, 76, 51, 84, 66, NA~
## $ LCB                 <dbl> 53, 45, 46, 58, NA, 57, NA, 47, 72, 46, 87, 57, NA~
## $ LCM                 <dbl> 82, 84, 79, 80, NA, 78, NA, 81, 87, 71, 74, 84, NA~
## $ LDM                 <dbl> 62, 59, 59, 65, NA, 62, NA, 61, 82, 52, 83, 70, NA~
## $ LF                  <dbl> 91, 92, 88, 88, NA, 87, NA, 87, 81, 84, 70, 85, NA~
## $ LM                  <dbl> 89, 90, 87, 85, NA, 82, NA, 87, 81, 79, 71, 85, NA~
## $ LS                  <dbl> 92, 88, 84, 88, NA, 88, NA, 82, 77, 87, 72, 81, NA~
## $ LW                  <dbl> 91, 91, 89, 87, NA, 84, NA, 88, 80, 82, 69, 85, NA~
## $ LWB                 <dbl> 66, 62, 64, 68, NA, 61, NA, 64, 78, 55, 81, 71, NA~
## $ Preferred.Positions <chr> "ST LW ", "RW ", "LW ", "ST ", "GK ", "ST ", "GK "~
## $ RAM                 <dbl> 89, 92, 88, 87, NA, 84, NA, 88, 83, 81, 70, 86, NA~
## $ RB                  <dbl> 61, 57, 59, 64, NA, 58, NA, 59, 76, 51, 84, 66, NA~
## $ RCB                 <dbl> 53, 45, 46, 58, NA, 57, NA, 47, 72, 46, 87, 57, NA~
## $ RCM                 <dbl> 82, 84, 79, 80, NA, 78, NA, 81, 87, 71, 74, 84, NA~
## $ RDM                 <dbl> 62, 59, 59, 65, NA, 62, NA, 61, 82, 52, 83, 70, NA~
## $ RF                  <dbl> 91, 92, 88, 88, NA, 87, NA, 87, 81, 84, 70, 85, NA~
## $ RM                  <dbl> 89, 90, 87, 85, NA, 82, NA, 87, 81, 79, 71, 85, NA~
## $ RS                  <dbl> 92, 88, 84, 88, NA, 88, NA, 82, 77, 87, 72, 81, NA~
## $ RW                  <dbl> 91, 91, 89, 87, NA, 84, NA, 88, 80, 82, 69, 85, NA~
## $ RWB                 <dbl> 66, 62, 64, 68, NA, 61, NA, 64, 78, 55, 81, 71, NA~
## $ ST                  <dbl> 92, 88, 84, 88, NA, 88, NA, 82, 77, 87, 72, 81, NA~
```

```r
# Seleção e Limpeza dos Dados
df <- df_orign %>%
  select(Age, Overall, Potential, Wage, Special,
         Acceleration, Aggression, Agility, Balance, Ball.control,
         Composure, Crossing, Curve, Dribbling, Finishing, Positioning,
         Stamina, Interceptions, Strength, Vision, Volleys, Jumping, Penalties,
         Shot.power, Sprint.speed, Heading.accuracy, Long.passing, Short.passing) %>%

  # Extrair apenas números da coluna Wage
  mutate(Wage = as.integer(str_extract(Wage, "[0-9]+"))) %>%

  # Converter colunas de texto para número
  mutate_if(is.character, as.integer) %>%

  # Remover entradas com NA
  na.omit()

glimpse(df)
```

```
## Rows: 17,401
## Columns: 28
## $ Age              <int> 32, 30, 25, 30, 31, 28, 26, 26, 27, 29, 31, 26, 25, 2~
## $ Overall          <int> 94, 93, 92, 92, 92, 91, 90, 90, 90, 90, 90, 89, 89, 8~
## $ Potential        <int> 94, 93, 94, 92, 92, 91, 92, 91, 90, 90, 90, 92, 92, 8~
## $ Wage             <int> 565, 565, 280, 510, 230, 355, 215, 295, 340, 275, 310~
## $ Special          <int> 2228, 2154, 2100, 2291, 1493, 2143, 1458, 2096, 2165,~
## $ Acceleration     <int> 89, 92, 94, 88, 58, 79, 57, 93, 60, 78, 75, 76, 46, 8~
## $ Aggression       <int> 63, 48, 56, 78, 29, 80, 38, 54, 60, 50, 84, 68, 23, 8~
## $ Agility          <int> 89, 90, 96, 86, 52, 78, 60, 93, 71, 75, 79, 80, 61, 9~
## $ Balance          <int> 63, 95, 82, 60, 35, 80, 43, 91, 69, 69, 60, 75, 45, 8~
## $ Ball.control     <int> 93, 95, 95, 91, 48, 89, 42, 92, 89, 85, 84, 87, 23, 8~
## $ Composure        <int> 95, 96, 92, 83, 70, 87, 64, 87, 85, 86, 80, 84, 52, 8~
## $ Crossing         <int> 85, 77, 75, 77, 15, 62, 17, 80, 85, 68, 66, 90, 14, 8~
## $ Curve            <int> 81, 89, 81, 86, 14, 77, 21, 82, 85, 74, 73, 83, 19, 7~
## $ Dribbling        <int> 91, 97, 96, 86, 30, 85, 18, 93, 79, 84, 61, 85, 13, 9~
## $ Finishing        <int> 94, 95, 89, 94, 13, 91, 13, 83, 76, 91, 60, 83, 14, 8~
## $ Positioning      <int> 95, 93, 90, 92, 12, 91, 12, 85, 79, 92, 52, 84, 13, 8~
## $ Stamina          <int> 92, 73, 78, 89, 44, 79, 40, 79, 77, 72, 84, 87, 38, 8~
## $ Interceptions    <int> 29, 22, 36, 41, 30, 39, 30, 41, 85, 20, 88, 56, 15, 4~
## $ Strength         <int> 80, 59, 53, 80, 83, 84, 64, 65, 74, 85, 81, 73, 70, 7~
## $ Vision           <int> 85, 90, 80, 84, 70, 78, 68, 86, 88, 70, 63, 90, 44, 8~
## $ Volleys          <int> 88, 85, 83, 88, 11, 87, 13, 79, 82, 88, 66, 82, 12, 8~
## $ Jumping          <int> 95, 68, 61, 69, 78, 84, 67, 59, 32, 79, 93, 65, 68, 8~
## $ Penalties        <int> 85, 74, 81, 85, 47, 81, 40, 86, 73, 70, 68, 77, 27, 7~
## $ Shot.power       <int> 94, 85, 80, 87, 25, 88, 31, 79, 87, 88, 79, 85, 36, 8~
## $ Sprint.speed     <int> 91, 87, 90, 77, 61, 83, 58, 87, 52, 80, 77, 75, 52, 8~
## $ Heading.accuracy <int> 88, 71, 62, 77, 25, 85, 21, 57, 54, 86, 91, 53, 13, 7~
## $ Long.passing     <int> 77, 87, 75, 64, 59, 65, 51, 81, 93, 59, 72, 84, 31, 7~
## $ Short.passing    <int> 83, 88, 81, 83, 55, 83, 50, 86, 90, 75, 78, 90, 32, 8~
```

```r
# Criar Modelo de Regressão Linear
model <- lm(Overall ~ ., data = df)
summary(model)
```

```
##
## Call:
## lm(formula = Overall ~ ., data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.8183  -1.2656   0.1601   1.4196   8.1059
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -1.730e+01  3.202e-01 -54.016  < 2e-16 ***
## Age               6.674e-01  5.481e-03 121.768  < 2e-16 ***
## Potential         6.423e-01  4.353e-03 147.540  < 2e-16 ***
## Wage              2.311e-02  8.615e-04  26.822  < 2e-16 ***
## Special           2.950e-02  4.823e-04  61.155  < 2e-16 ***
## Acceleration     -9.554e-04  3.085e-03  -0.310   0.7568
## Aggression       -3.784e-02  1.806e-03 -20.959  < 2e-16 ***
## Agility          -1.910e-02  2.368e-03  -8.065 7.77e-16 ***
## Balance          -5.443e-02  2.144e-03 -25.383  < 2e-16 ***
```

```
## Ball.control      3.138e-02  3.832e-03   8.188 2.85e-16 ***
## Composure         4.108e-02  2.396e-03  17.141  < 2e-16 ***
## Crossing         -2.725e-02  2.236e-03 -12.188  < 2e-16 ***
## Curve            -4.148e-02  2.195e-03 -18.897  < 2e-16 ***
## Dribbling        -4.094e-02  3.206e-03 -12.768  < 2e-16 ***
## Finishing        -1.222e-02  2.504e-03  -4.881 1.06e-06 ***
## Positioning      -3.697e-02  2.475e-03 -14.936  < 2e-16 ***
## Stamina          -8.615e-03  1.974e-03  -4.365 1.28e-05 ***
## Interceptions    -8.232e-02  2.101e-03 -39.182  < 2e-16 ***
## Strength          3.148e-03  2.072e-03   1.519   0.1288
## Vision           -2.873e-02  2.327e-03 -12.346  < 2e-16 ***
## Volleys          -2.534e-02  2.338e-03 -10.836  < 2e-16 ***
## Jumping          -2.068e-02  1.771e-03 -11.674  < 2e-16 ***
## Penalties        -4.729e-02  2.169e-03 -21.797  < 2e-16 ***
## Shot.power       -3.712e-02  2.204e-03 -16.843  < 2e-16 ***
## Sprint.speed     -6.745e-03  2.888e-03  -2.335   0.0195 *
## Heading.accuracy -4.445e-03  1.950e-03  -2.280   0.0226 *
## Long.passing     -5.715e-02  2.860e-03 -19.987  < 2e-16 ***
## Short.passing     9.438e-03  3.702e-03   2.549   0.0108 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.111 on 17373 degrees of freedom
## Multiple R-squared:  0.9091, Adjusted R-squared:  0.9089
## F-statistic:  6433 on 27 and 17373 DF,  p-value: < 2.2e-16
```

```r
# Análise de Colinearidade (VIF)
vif_values <- vif(model)
print(vif_values)
```

```
##              Age        Potential             Wage          Special
##         2.509186         2.745870         1.554025        67.805324
##     Acceleration       Aggression          Agility          Balance
##         8.330612         3.911186         4.820937         3.595871
##     Ball.control        Composure         Crossing            Curve
##        16.392557         3.774959         6.693749         6.436049
##        Dribbling        Finishing      Positioning          Stamina
##        14.632799         9.319317         9.092076         3.902737
##    Interceptions         Strength           Vision          Volleys
##         7.394824         2.675883         4.393156         6.733236
##          Jumping        Penalties       Shot.power     Sprint.speed
##         1.738479         4.629194         5.769647         7.027140
## Heading.accuracy     Long.passing    Short.passing
##         4.545899         7.753732        12.053466
```

```r
# Removendo Variáveis com Alta Colinearidade
df_reduced <- df %>%
  select(-Potential, -Short.passing)  # Exemplo de remoção

model_reduced <- lm(Overall ~ ., data = df_reduced)
summary(model_reduced)
```

```
##
```

```
## Call:
## lm(formula = Overall ~ ., data = df_reduced)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -30.3608  -2.1028   0.0212   2.0653  12.5871
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      17.0580752  0.3303574  51.635  < 2e-16 ***
## Age               0.1928030  0.0066542  28.974  < 2e-16 ***
## Wage              0.0672892  0.0012130  55.474  < 2e-16 ***
## Special           0.0656489  0.0006225 105.467  < 2e-16 ***
## Acceleration     -0.0276101  0.0046250  -5.970 2.42e-09 ***
## Aggression       -0.0810838  0.0026753 -30.309  < 2e-16 ***
## Agility          -0.0512702  0.0035393 -14.486  < 2e-16 ***
## Balance          -0.1022399  0.0031797 -32.154  < 2e-16 ***
## Ball.control      0.0961452  0.0054181  17.745  < 2e-16 ***
## Composure         0.1270856  0.0034871  36.444  < 2e-16 ***
## Crossing         -0.0872292  0.0033011 -26.425  < 2e-16 ***
## Curve            -0.0902044  0.0032542 -27.719  < 2e-16 ***
## Dribbling        -0.0721986  0.0048037 -15.030  < 2e-16 ***
## Finishing        -0.0394204  0.0037490 -10.515  < 2e-16 ***
## Positioning      -0.0891764  0.0036765 -24.255  < 2e-16 ***
## Stamina          -0.0702638  0.0028956 -24.266  < 2e-16 ***
## Interceptions    -0.1746208  0.0030120 -57.976  < 2e-16 ***
## Strength         -0.0136529  0.0031046  -4.398 1.10e-05 ***
## Vision           -0.0522858  0.0034834 -15.010  < 2e-16 ***
## Volleys          -0.0652117  0.0034878 -18.697  < 2e-16 ***
## Jumping          -0.0440471  0.0026454 -16.650  < 2e-16 ***
## Penalties        -0.0867901  0.0032327 -26.847  < 2e-16 ***
## Shot.power       -0.0865521  0.0032702 -26.467  < 2e-16 ***
## Sprint.speed     -0.0220336  0.0043324  -5.086 3.70e-07 ***
## Heading.accuracy -0.0079990  0.0029002  -2.758  0.00582 **
## Long.passing     -0.1205038  0.0035895 -33.572  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.169 on 17375 degrees of freedom
## Multiple R-squared:  0.7949, Adjusted R-squared:  0.7946
## F-statistic:  2694 on 25 and 17375 DF,  p-value: < 2.2e-16
```

```r
vif(model_reduced)
```

```
##              Age             Wage          Special     Acceleration
##         1.640322         1.366349        50.080254         8.301361
##       Aggression          Agility          Balance     Ball.control
##         3.808282         4.776944         3.506193        14.531173
##        Composure         Crossing            Curve        Dribbling
##         3.544956         6.472335         6.273429        14.568017
##        Finishing      Positioning          Stamina    Interceptions
##         9.265839         8.896207         3.725004         6.739991
##         Strength           Vision          Volleys          Jumping
##         2.663833         4.364564         6.643032         1.720227
```

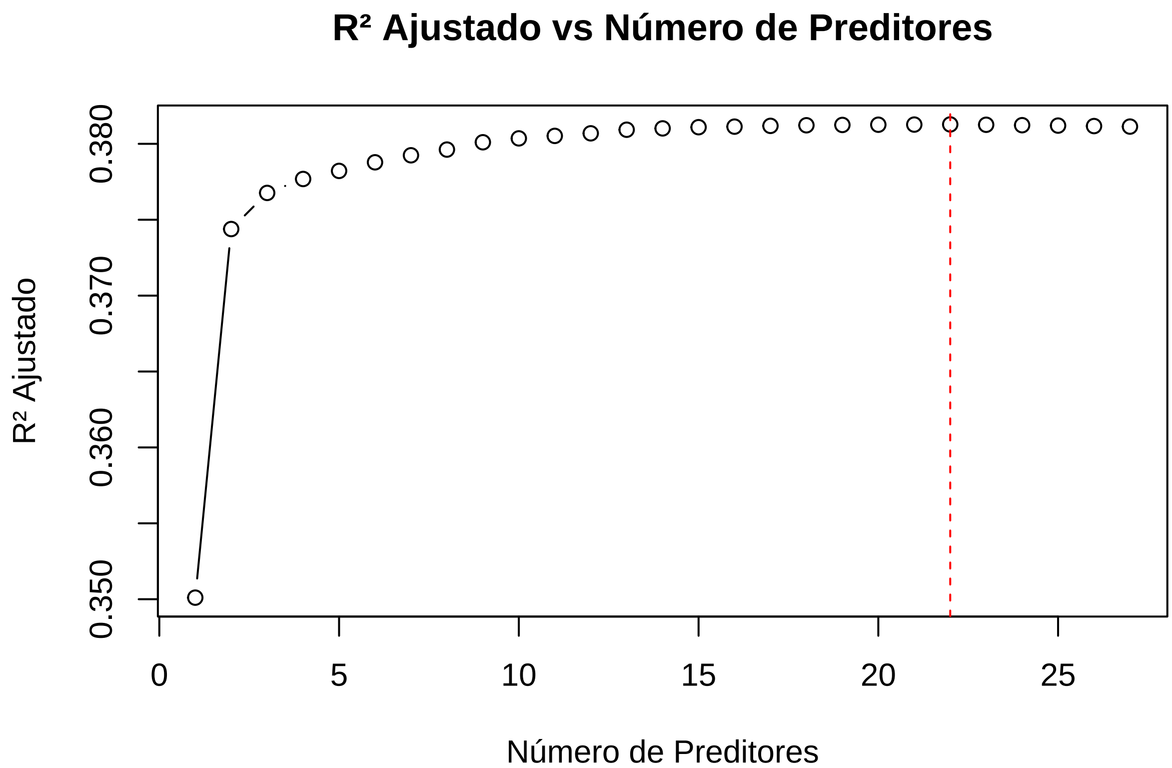
```
##        Penalties        Shot.power      Sprint.speed Heading.accuracy
##         4.558097          5.635464          7.011517         4.461762
##     Long.passing
##         5.418254
```

Solução Exercicio 03

```r
# Carregar bibliotecas
library(leaps)
library(dplyr)
library(broom)

# Executar o Best Subset Selection
regfit.full <- regsubsets(Wage ~ ., data = df, method = "exhaustive", nvmax = nrow(df) - 1)

# Visualizar os resultados
tidy(regfit.full) %>% View()

# Extrair o resumo e encontrar o melhor modelo
regfit.summary <- tidy(regfit.full)
best_model_index <- which.max(regfit.summary$adj.r.squared)
best_model_index
```

```
## [1] 22
```

```r
# Criar o gráfico do R² ajustado
plot(regfit.summary$adj.r.squared, type = "b", xlab = "Número de Preditores", ylab = "R² Ajustado", mai
abline(v = best_model_index, col = "red", lty = 2)
```

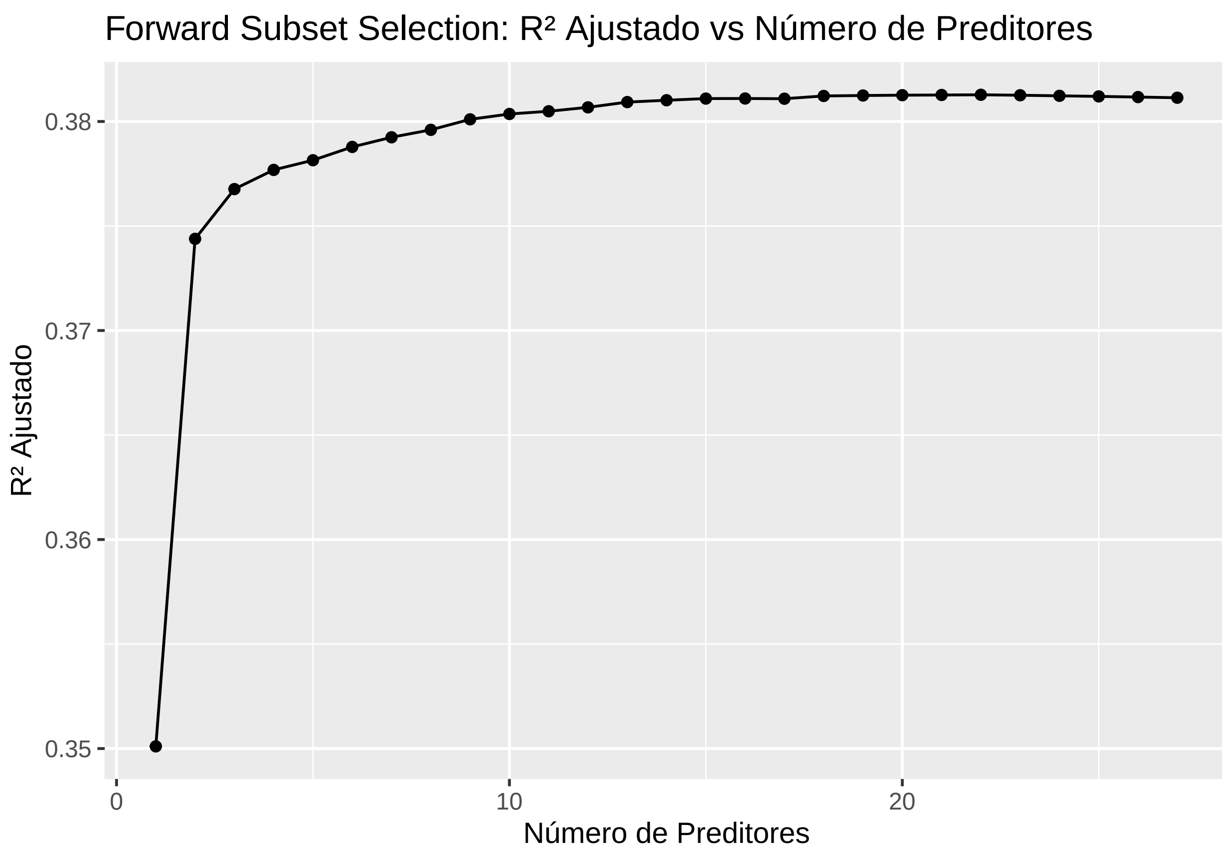


Solução Exercício 04

```
library(leaps)

regfit.forward <- regsubsets(Wage ~ ., df, method = "forward", nvmax = ncol(df)-1)
regfit.summary <- tidy(regfit.forward)

# Encontrar o melhor modelo com maior R² ajustado
best_model <- which.max(regfit.summary$adj.r.squared)
print(paste("Melhor modelo encontrado com", best_model, "preditores."))
```

```
## [1] "Melhor modelo encontrado com 22 preditores."
```

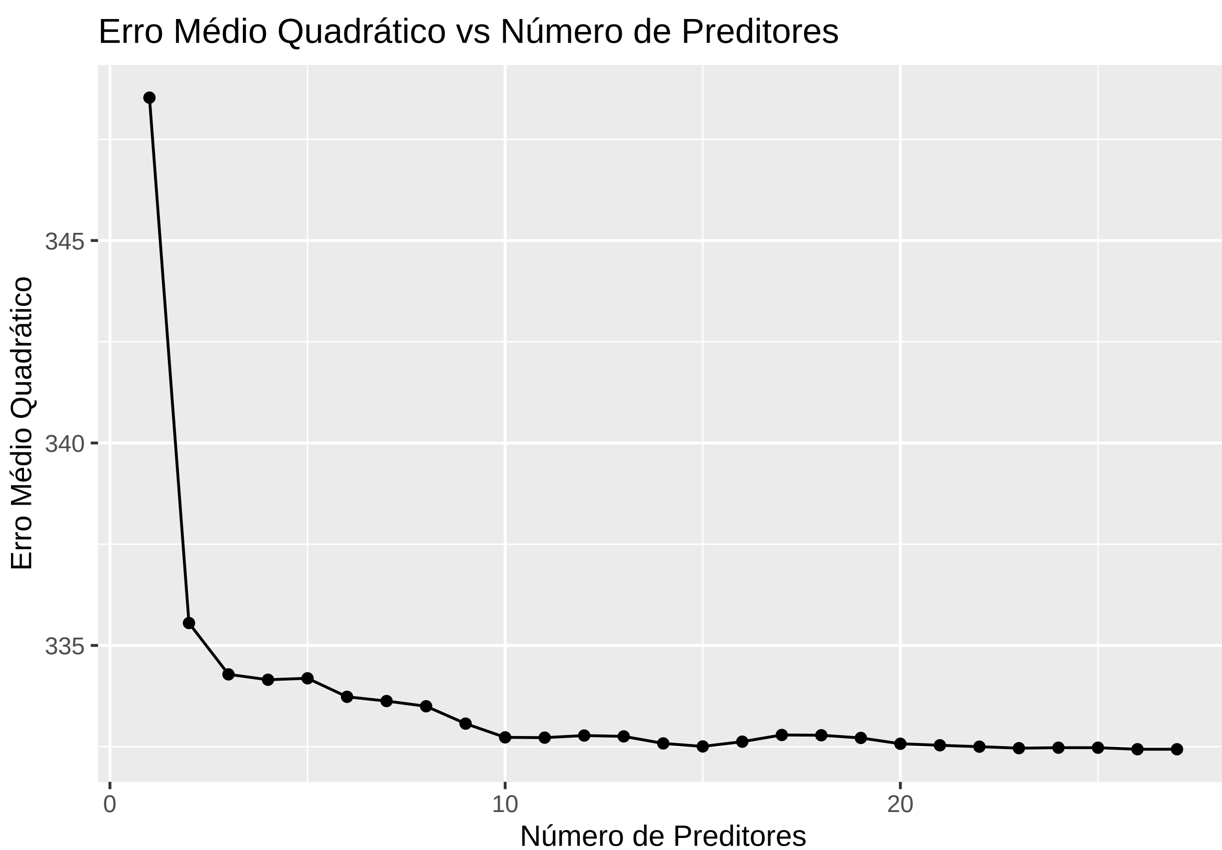
```
# Gráfico do R² ajustado
plot.df <- tibble(Preditores = 1:nrow(regfit.summary), R2_Ajustado = regfit.summary$adj.r.squared)

ggplot(plot.df, aes(x = Preditores, y = R2_Ajustado)) +
  geom_line() +
  geom_point() +
  labs(title = "Forward Subset Selection: R² Ajustado vs Número de Preditores",
       x = "Número de Preditores",
       y = "R² Ajustado")
```



Solução Exercício 5

```r
library(rsample)

cv.split <- vfold_cv(df, v = 10)

# Criar matriz para armazenar os resultados
results <- matrix(0, nrow = length(cv.split$splits), ncol = ncol(df) - 1)

for (i in 1:length(cv.split$splits)) {
  s <- cv.split$splits[[i]]
  train <- analysis(s)
  test <- assessment(s)

  rss.fit <- regsubsets(Wage ~ ., train, method = "forward", nvmax = ncol(df)-1)
  rss.td <- tidy(rss.fit)

  for (j in 1:nrow(rss.td)) {
    coefs <- coef(rss.fit, id = j)
    v.names <- names(coefs)
    test.mat <- model.matrix(Wage ~ ., data = test)
    pred <- test.mat[, v.names] %*% coefs
    MSS <- mean((test$Wage - pred)^2)
```

```r
    results[i, j] = MSS
  }
}

# Criar dataframe com resultados médios
ggplot(tibble(Preditores = 1:ncol(results), MSS = colMeans(results)), aes(x = Preditores, y = MSS)) +
  geom_line() +
  geom_point() +
  labs(title = "Erro Médio Quadrático vs Número de Preditores",
       x = "Número de Preditores",
       y = "Erro Médio Quadrático")
```




```r
# Melhor modelo baseado no menor erro médio quadrático
best_model_cv <- which.min(colMeans(results))
print(paste("Melhor modelo encontrado com", best_model_cv, "preditores baseado na validação cruzada."))
```

```
## [1] "Melhor modelo encontrado com 27 preditores baseado na validação cruzada."
```