Códigos - Modelos de Regressão Linear (FIFA 22)

Contents

Inicialmente carreguemos todas as bibliotecas a serem utilizadas nessa análise:

```
library(readr)
library(ggplot2)
library(readr)
library(stringr)
library(MASS)
library(dplyr)
library(tidyr)
library(dplyr)
library(goftest)
library(nortest)
library(faraway)
library(ggridges)
library(Hmisc)
library(GGally)
library(glmnet)
library(car)
library(lmtest)
library(leaps)
```

e o conjunto de dados, que por estar dividido em dois arquivos, precisa ser agrupado:

```
basic_info <- read_csv("basic_info.csv", show_col_types = FALSE)
detailed_info <- read_csv("detailed_info.csv",  show_col_types = FALSE)
dados <- basic_info %>% left_join(detailed_info, by = "ID")
```

Vamos observar todas as primeiras linhas de cada co-váriavel:

```
glimpse(dados)
```

```
## Rows: 19,825
## Columns: 86
## $ ...1.x                     <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1~
## $ ID                         <dbl> 236988, 225863, 241721, 224371, 200104, 238~
## $ Name                       <chr> "Eddie Nketiah", "Olivier Boscagli", "Rafae~
## $ Age                        <dbl> 22, 23, 22, 24, 28, 21, 23, 21, 16, 22, 25,~
## $ Nationality                <chr> "England", "France", "Portugal", "England",~
## $ Overall                    <dbl> 72, 77, 82, 79, 89, 78, 81, 81, 77, 78, 77,~
## $ Potential                  <dbl> 79, 82, 90, 82, 89, 83, 87, 88, 89, 86, 79,~
## $ Club                       <chr> "Arsenal", "PSV", "AC Milan", "West Ham Uni~
## $ Contract                   <chr> "2016 ~ 2022", "2019 ~ 2025", "2019 ~ 2024"~
## $ Value                      <chr> "€4.8M", "€14.5M", "€68.5M", "€24M", "€104M~
## $ Wage                       <chr> "€45K", "€15K", "€52K", "€63K", "€240K", "€~
## $ `Total stat`               <dbl> 1698, 1961, 1959, 1966, 2141, 1955, 1730, 2~
## $ ...1.y                     <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1~
## $ LS                         <dbl> 74, 66, 82, 77, 88, 67, 59, 77, 69, 80, 56,~
## $ ST                         <dbl> 74, 66, 82, 77, 88, 67, 59, 77, 69, 80, 56,~
## $ RS                         <dbl> 74, 66, 82, 77, 88, 67, 59, 77, 69, 80, 56,~
## $ LW                         <dbl> 70, 69, 82, 78, 87, 70, 55, 76, 76, 76, 57,~
## $ LF                         <dbl> 72, 68, 82, 78, 87, 68, 57, 77, 74, 77, 57,~
## $ CF                         <dbl> 72, 68, 82, 78, 87, 68, 57, 77, 74, 77, 57,~
## $ RF                         <dbl> 72, 68, 82, 78, 87, 68, 57, 77, 74, 77, 57,~
## $ RW                         <dbl> 70, 69, 82, 78, 87, 70, 55, 76, 76, 76, 57,~
## $ LAM                        <dbl> 71, 73, 82, 79, 89, 70, 61, 80, 79, 77, 61,~
## $ CAM                        <dbl> 71, 73, 82, 79, 89, 70, 61, 80, 79, 77, 61,~
## $ RAM                        <dbl> 71, 73, 82, 79, 89, 70, 61, 80, 79, 77, 61,~
## $ LM                         <dbl> 70, 74, 83, 80, 89, 73, 61, 79, 78, 78, 62,~
## $ LCM                        <dbl> 62, 76, 74, 74, 83, 70, 68, 83, 79, 71, 67,~
## $ CM                         <dbl> 62, 76, 74, 74, 83, 70, 68, 83, 79, 71, 67,~
## $ RCM                        <dbl> 62, 76, 74, 74, 83, 70, 68, 83, 79, 71, 67,~
## $ RM                         <dbl> 70, 74, 83, 80, 89, 73, 61, 79, 78, 78, 62,~
## $ LWB                        <dbl> 50, 78, 60, 65, 71, 78, 72, 81, 73, 64, 72,~
## $ LDM                        <dbl> 47, 79, 56, 62, 67, 74, 79, 84, 74, 61, 76,~
## $ CDM                        <dbl> 47, 79, 56, 62, 67, 74, 79, 84, 74, 61, 76,~
## $ RDM                        <dbl> 47, 79, 56, 62, 67, 74, 79, 84, 74, 61, 76,~
## $ RWB                        <dbl> 50, 78, 60, 65, 71, 78, 72, 81, 73, 64, 72,~
## $ LB                         <dbl> 48, 78, 56, 62, 67, 78, 75, 81, 71, 62, 73,~
## $ LCB                        <dbl> 42, 79, 48, 55, 57, 75, 83, 82, 66, 58, 79,~
## $ CB                         <dbl> 42, 79, 48, 55, 57, 75, 83, 82, 66, 58, 79,~
## $ RCB                        <dbl> 42, 79, 48, 55, 57, 75, 83, 82, 66, 58, 79,~
## $ RB                         <dbl> 48, 78, 56, 62, 67, 78, 75, 81, 71, 62, 73,~
## $ GK                         <dbl> 18, 19, 21, 19, 22, 18, 19, 19, 19, 21, 22,~
## $ Height                     <dbl> 180, 181, 188, 175, 183, 169, 190, 187, 173~
## $ Weight                     <dbl> 73, 68, 81, 70, 78, 67, 87, 81, 68, 81, 75,~
## $ DOB                        <chr> "May 30, 1999", "Nov 18, 1997", "Jun 10, 19~
## $ `Preferred foot`           <chr> "Right", "Left", "Right", "Left", "Right", ~
## $ `Weak foot`                <dbl> 3, 4, 4, 3, 5, 3, 2, 3, 3, 3, 3, 3, 3, 4, 5~
## $ `Skill move`               <dbl> 3, 3, 4, 4, 4, 3, 2, 3, 3, 3, 2, 2, 4, 4, 4~
## $ `International reputation` <dbl> 1, 1, 1, 1, 4, 1, 2, 2, 1, 1, 1, 1, 2, 1, 3~
## $ `Work rate`                <chr> "High/Medium", "Medium/Medium", "Medium/Low~
```

```
## $ `Body type`                <chr> "Lean (170-185)", "Lean (170-185)", "Lean (~
## $ `Real face`                <chr> "Yes", "Yes", "No", "Yes", "Yes", "No", "No~
## $ `Release clause`           <chr> "€10.1M", "€21.4M", "€130.2M", "€47.4M", "€~
## $ Crossing                   <dbl> 43, 73, 69, 79, 83, 75, 38, 68, 64, 67, 50,~
## $ Finishing                  <dbl> 73, 33, 80, 80, 88, 50, 31, 67, 64, 81, 30,~
## $ `Heading Accuracy`         <dbl> 66, 74, 63, 64, 68, 58, 85, 73, 42, 73, 72,~
## $ `Short Passing`            <dbl> 66, 78, 76, 74, 84, 75, 75, 84, 81, 73, 75,~
## $ Volleys                    <dbl> 69, 29, 74, 58, 79, 59, 28, 50, 49, 67, 27,~
## $ Dribbling                  <dbl> 74, 73, 89, 82, 87, 75, 52, 78, 82, 73, 64,~
## $ Curve                      <dbl> 54, 52, 80, 71, 85, 36, 47, 76, 74, 67, 33,~
## $ `FK Accuracy`              <dbl> 38, 56, 60, 70, 74, 42, 27, 53, 47, 61, 30,~
## $ `Long Passing`             <dbl> 45, 78, 62, 67, 74, 56, 72, 83, 76, 52, 66,~
## $ `Ball Control`             <dbl> 73, 77, 85, 79, 84, 74, 65, 80, 81, 75, 69,~
## $ Acceleration               <dbl> 85, 71, 90, 78, 85, 84, 62, 70, 78, 88, 55,~
## $ `Sprint Speed`             <dbl> 82, 75, 92, 76, 90, 85, 73, 73, 75, 86, 58,~
## $ Agility                    <dbl> 80, 72, 82, 79, 86, 82, 51, 75, 86, 77, 54,~
## $ Reactions                  <dbl> 74, 77, 83, 77, 91, 73, 81, 81, 76, 80, 78,~
## $ Balance                    <dbl> 72, 70, 80, 83, 78, 89, 37, 71, 89, 54, 61,~
## $ `Shot Power`               <dbl> 72, 73, 80, 72, 88, 65, 51, 78, 56, 83, 48,~
## $ Jumping                    <dbl> 73, 82, 62, 64, 60, 85, 70, 74, 70, 73, 76,~
## $ Stamina                    <dbl> 66, 76, 75, 80, 88, 80, 67, 86, 77, 87, 68,~
## $ Strength                   <dbl> 59, 75, 80, 62, 64, 67, 87, 82, 56, 86, 78,~
## $ `Long Shots`               <dbl> 60, 64, 72, 76, 89, 62, 30, 72, 59, 71, 27,~
## $ Aggression                 <dbl> 52, 71, 60, 57, 62, 77, 81, 82, 82, 78, 75,~
## $ Interceptions              <dbl> 19, 80, 24, 37, 39, 74, 80, 84, 65, 41, 78,~
## $ Positioning                <dbl> 74, 62, 80, 79, 91, 60, 44, 75, 73, 77, 40,~
## $ Vision                     <dbl> 58, 74, 74, 74, 84, 52, 59, 82, 81, 74, 57,~
## $ Penalties                  <dbl> 68, 36, 63, 70, 73, 46, 40, 55, 51, 76, 38,~
## $ Composure                  <dbl> 68, 77, 81, 75, 89, 66, 74, 82, 74, 82, 74,~
## $ `Defensive Awareness`      <dbl> 22, 76, 22, 49, 47, 70, 83, 80, 59, 36, 77,~
## $ `Standing Tackle`          <dbl> 19, 78, 24, 40, 34, 80, 84, 83, 66, 39, 81,~
## $ `Sliding Tackle`           <dbl> 15, 77, 21, 36, 33, 78, 80, 78, 62, 33, 78,~
## $ `GK Diving`                <dbl> 12, 8, 10, 14, 11, 9, 12, 9, 8, 14, 14, 8, ~
## $ `GK Handling`              <dbl> 10, 7, 12, 8, 13, 9, 8, 8, 10, 10, 11, 11, ~
## $ `GK Kicking`               <dbl> 11, 14, 15, 14, 13, 12, 13, 12, 11, 12, 10,~
## $ `GK Positioning`           <dbl> 9, 8, 11, 10, 6, 8, 10, 12, 7, 10, 14, 5, 1~
## $ `GK Reflexes`              <dbl> 5, 12, 9, 7, 10, 8, 7, 7, 13, 13, 13, 9, 5,~
## $ Traits                     <chr> "Chip Shot (AI)", "Dives Into Tackles (AI),~
```

Ao todo temos 19825 linhas e 86 colunas.

Retirando colunas

Vamos retirar algumas colunas que julgamos ser não relevantes para o propósito desse trabalho.

As colunas abaixo vamos retirar devido ao fato de não estarem bem documentadas. Isto é, não apresentam uma descrição clara.

```
cols_rm <- c("LS", "ST", "RS", "LW", "LF", "CF", "RF", "RW", "LAM", "CAM",
             "RAM", "LM", "LCM", "CM", "RCM", "RM", "LWB", "LDM", "CDM",
             "RDM", "RWB", "LB", "LCB", "CB", "RCB", "RB", "GK",
             "GK Diving", "GK Handling", "GK Kicking", "GK Positioning",
             "GK Reflexes", "Weak foot", "International reputation",
```

```
            "Real face", "Total stat") # 36 colunas
dados1 <- dados[, -which(colnames(dados) %in% cols_rm)]
```

As colunas "...1.x", "ID", "Name", "...1.y", "DOB" também serão retiradas

```
dados1 <- as.data.frame(dados1)
rownames(dados1) <- paste(dados1$Name, dados1$ID, sep = "_")
dados2 <- dados1 %>% dplyr::select(-`...1.x`, -`ID`, -`Name`, -`...1.y`, -`DOB`)
glimpse(dados2)
```

```
## Rows: 19,825
## Columns: 45
## $ Age                 <dbl> 22, 23, 22, 24, 28, 21, 23, 21, 16, 22, 25, 22, ~
## $ Nationality         <chr> "England", "France", "Portugal", "England", "Kor~
## $ Overall             <dbl> 72, 77, 82, 79, 89, 78, 81, 81, 77, 78, 77, 83, ~
## $ Potential           <dbl> 79, 82, 90, 82, 89, 83, 87, 88, 89, 86, 79, 88, ~
## $ Club                <chr> "Arsenal", "PSV", "AC Milan", "West Ham United",~
## $ Contract            <chr> "2016 ~ 2022", "2019 ~ 2025", "2019 ~ 2024", "20~
## $ Value               <chr> "€4.8M", "€14.5M", "€68.5M", "€24M", "€104M", "€~
## $ Wage                <chr> "€45K", "€15K", "€52K", "€63K", "€240K", "€9K", ~
## $ Height              <dbl> 180, 181, 188, 175, 183, 169, 190, 187, 173, 187~
## $ Weight              <dbl> 73, 68, 81, 70, 78, 67, 87, 81, 68, 81, 75, 80, ~
## $ `Preferred foot`    <chr> "Right", "Left", "Right", "Left", "Right", "Left~
## $ `Skill move`        <dbl> 3, 3, 4, 4, 4, 3, 2, 3, 3, 3, 2, 2, 4, 4, 4, 3, ~
## $ `Work rate`         <chr> "High/Medium", "Medium/Medium", "Medium/Low", "H~
## $ `Body type`         <chr> "Lean (170-185)", "Lean (170-185)", "Lean (185+)~
## $ `Release clause`    <chr> "€10.1M", "€21.4M", "€130.2M", "€47.4M", "€197.6~
## $ Crossing            <dbl> 43, 73, 69, 79, 83, 75, 38, 68, 64, 67, 50, 55, ~
## $ Finishing           <dbl> 73, 33, 80, 80, 88, 50, 31, 67, 64, 81, 30, 61, ~
## $ `Heading Accuracy`  <dbl> 66, 74, 63, 64, 68, 58, 85, 73, 42, 73, 72, 72, ~
## $ `Short Passing`     <dbl> 66, 78, 76, 74, 84, 75, 75, 84, 81, 73, 75, 83, ~
## $ Volleys             <dbl> 69, 29, 74, 58, 79, 59, 28, 50, 49, 67, 27, 50, ~
## $ Dribbling           <dbl> 74, 73, 89, 82, 87, 75, 52, 78, 82, 73, 64, 78, ~
## $ Curve               <dbl> 54, 52, 80, 71, 85, 36, 47, 76, 74, 67, 33, 58, ~
## $ `FK Accuracy`       <dbl> 38, 56, 60, 70, 74, 42, 27, 53, 47, 61, 30, 58, ~
## $ `Long Passing`      <dbl> 45, 78, 62, 67, 74, 56, 72, 83, 76, 52, 66, 82, ~
## $ `Ball Control`      <dbl> 73, 77, 85, 79, 84, 74, 65, 80, 81, 75, 69, 79, ~
## $ Acceleration        <dbl> 85, 71, 90, 78, 85, 84, 62, 70, 78, 88, 55, 70, ~
## $ `Sprint Speed`      <dbl> 82, 75, 92, 76, 90, 85, 73, 73, 75, 86, 58, 74, ~
## $ Agility             <dbl> 80, 72, 82, 79, 86, 82, 51, 75, 86, 77, 54, 55, ~
## $ Reactions           <dbl> 74, 77, 83, 77, 91, 73, 81, 81, 76, 80, 78, 80, ~
## $ Balance             <dbl> 72, 70, 80, 83, 78, 89, 37, 71, 89, 54, 61, 67, ~
## $ `Shot Power`        <dbl> 72, 73, 80, 72, 88, 65, 51, 78, 56, 83, 48, 74, ~
## $ Jumping             <dbl> 73, 82, 62, 64, 60, 85, 70, 74, 70, 73, 76, 74, ~
## $ Stamina             <dbl> 66, 76, 75, 80, 88, 80, 67, 86, 77, 87, 68, 90, ~
## $ Strength            <dbl> 59, 75, 80, 62, 64, 67, 87, 82, 56, 86, 78, 78, ~
## $ `Long Shots`        <dbl> 60, 64, 72, 76, 89, 62, 30, 72, 59, 71, 27, 65, ~
## $ Aggression          <dbl> 52, 71, 60, 57, 62, 77, 81, 82, 82, 78, 75, 84, ~
## $ Interceptions       <dbl> 19, 80, 24, 37, 39, 74, 80, 84, 65, 41, 78, 84, ~
## $ Positioning         <dbl> 74, 62, 80, 79, 91, 60, 44, 75, 73, 77, 40, 58, ~
## $ Vision              <dbl> 58, 74, 74, 74, 84, 52, 59, 82, 81, 74, 57, 79, ~
## $ Penalties           <dbl> 68, 36, 63, 70, 73, 46, 40, 55, 51, 76, 38, 71, ~
## $ Composure           <dbl> 68, 77, 81, 75, 89, 66, 74, 82, 74, 82, 74, 83, ~
```

```
## $ ‘Defensive Awareness‘ <dbl> 22, 76, 22, 49, 47, 70, 83, 80, 59, 36, 77, 82, ~
## $ ‘Standing Tackle‘     <dbl> 19, 78, 24, 40, 34, 80, 84, 83, 66, 39, 81, 84, ~
## $ ‘Sliding Tackle‘      <dbl> 15, 77, 21, 36, 33, 78, 80, 78, 62, 33, 78, 80, ~
## $ Traits                <chr> "Chip Shot (AI)", "Dives Into Tackles (AI), Long~
```

Algumas variáveis categóricas também iremos retirar devido a grande quantidade de valores únicos.

```r
print(paste0("Nº de valores unicos Nationality: ", length(unique(dados2$Nationality))))
```

```
## [1] "Nº de valores unicos Nationality: 163"
```

```r
print(paste0("Nº de valores unicos Club: ", length(unique(dados2$Club))))
```

```
## [1] "Nº de valores unicos Club: 933"
```

```r
print(paste0("Nº de valores unicos Contract: ", length(unique(dados2$Contract))))
```

```
## [1] "Nº de valores unicos Contract: 239"
```

```r
print(paste0("Nº de valores unicos Work rate: ", length(unique(dados2$`Work rate`))))
```

```
## [1] "Nº de valores unicos Work rate: 10"
```

```r
print(paste0("Nº de valores unicos Traits: ", length(unique(dados2$Traits))))
```

```
## [1] "Nº de valores unicos Traits: 845"
```

```r
dados3 <- dados2[,
            -which(colnames(dados2) %in% c("Nationality", "Club", "Contract", "Work rate", "Traits"))]
glimpse(dados3)
```

```
## Rows: 19,825
## Columns: 40
## $ Age                   <dbl> 22, 23, 22, 24, 28, 21, 23, 21, 16, 22, 25, 22, ~
## $ Overall               <dbl> 72, 77, 82, 79, 89, 78, 81, 81, 77, 78, 77, 83, ~
## $ Potential             <dbl> 79, 82, 90, 82, 89, 83, 87, 88, 89, 86, 79, 88, ~
## $ Value                 <chr> "€4.8M", "€14.5M", "€68.5M", "€24M", "€104M", "€~
## $ Wage                  <chr> "€45K", "€15K", "€52K", "€63K", "€240K", "€9K", ~
## $ Height                <dbl> 180, 181, 188, 175, 183, 169, 190, 187, 173, 187~
## $ Weight                <dbl> 73, 68, 81, 70, 78, 67, 87, 81, 68, 81, 75, 80, ~
## $ ‘Preferred foot‘      <chr> "Right", "Left", "Right", "Left", "Right", "Left~
## $ ‘Skill move‘          <dbl> 3, 3, 4, 4, 4, 3, 2, 3, 3, 3, 2, 2, 4, 4, 4, 3, ~
## $ ‘Body type‘           <chr> "Lean (170-185)", "Lean (170-185)", "Lean (185+)~
## $ ‘Release clause‘      <chr> "€10.1M", "€21.4M", "€130.2M", "€47.4M", "€197.6~
## $ Crossing              <dbl> 43, 73, 69, 79, 83, 75, 38, 68, 64, 67, 50, 55, ~
## $ Finishing             <dbl> 73, 33, 80, 80, 88, 50, 31, 67, 64, 81, 30, 61, ~
## $ ‘Heading Accuracy‘    <dbl> 66, 74, 63, 64, 68, 58, 85, 73, 42, 73, 72, 72, ~
## $ ‘Short Passing‘       <dbl> 66, 78, 76, 74, 84, 75, 75, 84, 81, 73, 75, 83, ~
## $ Volleys               <dbl> 69, 29, 74, 58, 79, 59, 28, 50, 49, 67, 27, 50, ~
```

```
## $ Dribbling           <dbl> 74, 73, 89, 82, 87, 75, 52, 78, 82, 73, 64, 78, ~
## $ Curve               <dbl> 54, 52, 80, 71, 85, 36, 47, 76, 74, 67, 33, 58, ~
## $ `FK Accuracy`       <dbl> 38, 56, 60, 70, 74, 42, 27, 53, 47, 61, 30, 58, ~
## $ `Long Passing`      <dbl> 45, 78, 62, 67, 74, 56, 72, 83, 76, 52, 66, 82, ~
## $ `Ball Control`      <dbl> 73, 77, 85, 79, 84, 74, 65, 80, 81, 75, 69, 79, ~
## $ Acceleration        <dbl> 85, 71, 90, 78, 85, 84, 62, 70, 78, 88, 55, 70, ~
## $ `Sprint Speed`      <dbl> 82, 75, 92, 76, 90, 85, 73, 73, 75, 86, 58, 74, ~
## $ Agility             <dbl> 80, 72, 82, 79, 86, 82, 51, 75, 86, 77, 54, 55, ~
## $ Reactions           <dbl> 74, 77, 83, 77, 91, 73, 81, 81, 76, 80, 78, 80, ~
## $ Balance             <dbl> 72, 70, 80, 83, 78, 89, 37, 71, 89, 54, 61, 67, ~
## $ `Shot Power`        <dbl> 72, 73, 80, 72, 88, 65, 51, 78, 56, 83, 48, 74, ~
## $ Jumping             <dbl> 73, 82, 62, 64, 60, 85, 70, 74, 70, 73, 76, 74, ~
## $ Stamina             <dbl> 66, 76, 75, 80, 88, 80, 67, 86, 77, 87, 68, 90, ~
## $ Strength            <dbl> 59, 75, 80, 62, 64, 67, 87, 82, 56, 86, 78, 78, ~
## $ `Long Shots`        <dbl> 60, 64, 72, 76, 89, 62, 30, 72, 59, 71, 27, 65, ~
## $ Aggression          <dbl> 52, 71, 60, 57, 62, 77, 81, 82, 82, 78, 75, 84, ~
## $ Interceptions       <dbl> 19, 80, 24, 37, 39, 74, 80, 84, 65, 41, 78, 84, ~
## $ Positioning         <dbl> 74, 62, 80, 79, 91, 60, 44, 75, 73, 77, 40, 58, ~
## $ Vision              <dbl> 58, 74, 74, 74, 84, 52, 59, 82, 81, 74, 57, 79, ~
## $ Penalties           <dbl> 68, 36, 63, 70, 73, 46, 40, 55, 51, 76, 38, 71, ~
## $ Composure           <dbl> 68, 77, 81, 75, 89, 66, 74, 82, 74, 82, 74, 83, ~
## $ `Defensive Awareness` <dbl> 22, 76, 22, 49, 47, 70, 83, 80, 59, 36, 77, 82, ~
## $ `Standing Tackle`   <dbl> 19, 78, 24, 40, 34, 80, 84, 83, 66, 39, 81, 84, ~
## $ `Sliding Tackle`    <dbl> 15, 77, 21, 36, 33, 78, 80, 78, 62, 33, 78, 80, ~
```

Vamos agora tratar as variáveis categóricas que restaram.

```
dados3 %>% dplyr::select(where(is.character)) %>% glimpse()
```

```
## Rows: 19,825
## Columns: 5
## $ Value            <chr> "€4.8M", "€14.5M", "€68.5M", "€24M", "€104M", "€20M",~
## $ Wage             <chr> "€45K", "€15K", "€52K", "€63K", "€240K", "€9K", "€74K~
## $ `Preferred foot` <chr> "Right", "Left", "Right", "Left", "Right", "Left", "L~
## $ `Body type`      <chr> "Lean (170-185)", "Lean (170-185)", "Lean (185+)", "N~
## $ `Release clause` <chr> "€10.1M", "€21.4M", "€130.2M", "€47.4M", "€197.6M", "~
```

```
dados3 <- dados3 %>% filter(`Release clause` != "</label>")
```

```
dados4 <- dados3 %>%
            mutate(
              Value = str_trim(Value),
              Wage = str_trim(Wage),
              `Release clause` = str_trim(`Release clause`)) %>%
            mutate(
              l_value = str_sub(Value, -1),
              l_wage = str_sub(Wage, -1),
              l_rc = str_sub(`Release clause`, -1)
            )
print(unique(dados4$l_value))
```

```
## [1] "M" "K"
```

```r
print(unique(dados4$l_wage))
```

```
## [1] "K" "0"
```

```r
print(unique(dados4$l_rc))
```

```
## [1] "M" "K"
```

Então Value está em Milhões e em Mil enquanto Wage está em Mil Vamos transformar O que está em milhão do Value em mil

```r
dados5 <- dados4 %>%
           mutate(
             Value = parse_number(Value),
             Wage = parse_number(Wage),
             `Release clause` = parse_number(`Release clause`)
           )
dados5[dados5$l_value == "M", "Value"] <- 1000 * dados5[dados5$l_value == "M", "Value"]
dados5[dados5$l_value == "0", "Value"] <- 0

dados5[dados5$l_wage == "M", "Wage"] <- 1000 * dados5[dados5$l_wage == "M", "Wage"]
dados5[dados5$l_wage == "0", "Wage"] <- 0

dados5[dados5$l_rc == "M", "Release clause"] <- 1000 * dados5[dados5$l_rc == "M", "Release clause"]
dados5[dados5$l_rc == "0", "Release clause"] <- 0

dados5 <- dados5 %>% dplyr::select(-l_value, -l_wage, -l_rc)
glimpse(dados5)
```

```
## Rows: 13,539
## Columns: 40
## $ Age                  <dbl> 22, 23, 22, 24, 28, 21, 23, 21, 16, 22, 25, 22, ~
## $ Overall              <dbl> 72, 77, 82, 79, 89, 78, 81, 81, 77, 78, 77, 83, ~
## $ Potential            <dbl> 79, 82, 90, 82, 89, 83, 87, 88, 89, 86, 79, 88, ~
## $ Value                <dbl> 4800, 14500, 68500, 24000, 104000, 20000, 37000,~
## $ Wage                 <dbl> 45, 15, 52, 63, 240, 9, 74, 46, 14, 15, 62, 76, ~
## $ Height               <dbl> 180, 181, 188, 175, 183, 169, 190, 187, 173, 187~
## $ Weight               <dbl> 73, 68, 81, 70, 78, 67, 87, 81, 68, 81, 75, 80, ~
## $ `Preferred foot`     <chr> "Right", "Left", "Right", "Left", "Right", "Left~
## $ `Skill move`         <dbl> 3, 3, 4, 4, 4, 3, 2, 3, 3, 3, 2, 2, 4, 4, 3, 4, ~
## $ `Body type`          <chr> "Lean (170-185)", "Lean (170-185)", "Lean (185+)~
## $ `Release clause`     <dbl> 10100, 21400, 130200, 47400, 197600, 29500, 7770~
## $ Crossing             <dbl> 43, 73, 69, 79, 83, 75, 38, 68, 64, 67, 50, 55, ~
## $ Finishing            <dbl> 73, 33, 80, 80, 88, 50, 31, 67, 64, 81, 30, 61, ~
## $ `Heading Accuracy`   <dbl> 66, 74, 63, 64, 68, 58, 85, 73, 42, 73, 72, 72, ~
## $ `Short Passing`      <dbl> 66, 78, 76, 74, 84, 75, 75, 84, 81, 73, 75, 83, ~
## $ Volleys              <dbl> 69, 29, 74, 58, 79, 59, 28, 50, 49, 67, 27, 50, ~
## $ Dribbling            <dbl> 74, 73, 89, 82, 87, 75, 52, 78, 82, 73, 64, 78, ~
## $ Curve                <dbl> 54, 52, 80, 71, 85, 36, 47, 76, 74, 67, 33, 58, ~
## $ `FK Accuracy`        <dbl> 38, 56, 60, 70, 74, 42, 27, 53, 47, 61, 30, 58, ~
## $ `Long Passing`       <dbl> 45, 78, 62, 67, 74, 56, 72, 83, 76, 52, 66, 82, ~
```

```
## $ ‘Ball Control‘        <dbl> 73, 77, 85, 79, 84, 74, 65, 80, 81, 75, 69, 79, ~
## $ Acceleration          <dbl> 85, 71, 90, 78, 85, 84, 62, 70, 78, 88, 55, 70, ~
## $ ‘Sprint Speed‘        <dbl> 82, 75, 92, 76, 90, 85, 73, 73, 75, 86, 58, 74, ~
## $ Agility               <dbl> 80, 72, 82, 79, 86, 82, 51, 75, 86, 77, 54, 55, ~
## $ Reactions             <dbl> 74, 77, 83, 77, 91, 73, 81, 81, 76, 80, 78, 80, ~
## $ Balance               <dbl> 72, 70, 80, 83, 78, 89, 37, 71, 89, 54, 61, 67, ~
## $ ‘Shot Power‘          <dbl> 72, 73, 80, 72, 88, 65, 51, 78, 56, 83, 48, 74, ~
## $ Jumping               <dbl> 73, 82, 62, 64, 60, 85, 70, 74, 70, 73, 76, 74, ~
## $ Stamina               <dbl> 66, 76, 75, 80, 88, 80, 67, 86, 77, 87, 68, 90, ~
## $ Strength              <dbl> 59, 75, 80, 62, 64, 67, 87, 82, 56, 86, 78, 78, ~
## $ ‘Long Shots‘          <dbl> 60, 64, 72, 76, 89, 62, 30, 72, 59, 71, 27, 65, ~
## $ Aggression            <dbl> 52, 71, 60, 57, 62, 77, 81, 82, 82, 78, 75, 84, ~
## $ Interceptions         <dbl> 19, 80, 24, 37, 39, 74, 80, 84, 65, 41, 78, 84, ~
## $ Positioning           <dbl> 74, 62, 80, 79, 91, 60, 44, 75, 73, 77, 40, 58, ~
## $ Vision                <dbl> 58, 74, 74, 74, 84, 52, 59, 82, 81, 74, 57, 79, ~
## $ Penalties             <dbl> 68, 36, 63, 70, 73, 46, 40, 55, 51, 76, 38, 71, ~
## $ Composure             <dbl> 68, 77, 81, 75, 89, 66, 74, 82, 74, 82, 74, 83, ~
## $ ‘Defensive Awareness‘ <dbl> 22, 76, 22, 49, 47, 70, 83, 80, 59, 36, 77, 82, ~
## $ ‘Standing Tackle‘     <dbl> 19, 78, 24, 40, 34, 80, 84, 83, 66, 39, 81, 84, ~
## $ ‘Sliding Tackle‘      <dbl> 15, 77, 21, 36, 33, 78, 80, 78, 62, 33, 78, 80, ~
```

Vamos agora as variáveis “Preferred foot” e “Body type”.

```
dados6 <- dados5 %>%
            mutate(
              body_type = as.factor(word(`Body type`, 1)),
              preferred_foot = as.factor(`Preferred foot`)
            ) %>%
            select(-`Body type`, -`Preferred foot`)
print("Body type")
```

```
## [1] "Body type"
```

```
print(table(dados6$body_type))
```

```
##
##   Lean Normal Stocky Unique
##   4885   7949    576    129
```

```
print("Preferred foot")
```

```
## [1] "Preferred foot"
```

```
print(table(dados6$preferred_foot))
```

```
##
##  Left Right
##  3207 10332
```

```r
dados7 <- dados6 %>% drop_na()
glimpse(dados7)
```

```
## Rows: 13,536
## Columns: 40
## $ Age                    <dbl> 22, 23, 22, 24, 28, 21, 23, 21, 16, 22, 25, 22, ~
## $ Overall                <dbl> 72, 77, 82, 79, 89, 78, 81, 81, 77, 78, 77, 83, ~
## $ Potential              <dbl> 79, 82, 90, 82, 89, 83, 87, 88, 89, 86, 79, 88, ~
## $ Value                  <dbl> 4800, 14500, 68500, 24000, 104000, 20000, 37000,~
## $ Wage                   <dbl> 45, 15, 52, 63, 240, 9, 74, 46, 14, 15, 62, 76, ~
## $ Height                 <dbl> 180, 181, 188, 175, 183, 169, 190, 187, 173, 187~
## $ Weight                 <dbl> 73, 68, 81, 70, 78, 67, 87, 81, 68, 81, 75, 80, ~
## $ `Skill move`           <dbl> 3, 3, 4, 4, 4, 3, 2, 3, 3, 3, 2, 2, 4, 4, 3, 4, ~
## $ `Release clause`       <dbl> 10100, 21400, 130200, 47400, 197600, 29500, 7770~
## $ Crossing               <dbl> 43, 73, 69, 79, 83, 75, 38, 68, 64, 67, 50, 55, ~
## $ Finishing              <dbl> 73, 33, 80, 80, 88, 50, 31, 67, 64, 81, 30, 61, ~
## $ `Heading Accuracy`     <dbl> 66, 74, 63, 64, 68, 58, 85, 73, 42, 73, 72, 72, ~
## $ `Short Passing`        <dbl> 66, 78, 76, 74, 84, 75, 75, 84, 81, 73, 75, 83, ~
## $ Volleys                <dbl> 69, 29, 74, 58, 79, 59, 28, 50, 49, 67, 27, 50, ~
## $ Dribbling              <dbl> 74, 73, 89, 82, 87, 75, 52, 78, 82, 73, 64, 78, ~
## $ Curve                  <dbl> 54, 52, 80, 71, 85, 36, 47, 76, 74, 67, 33, 58, ~
## $ `FK Accuracy`          <dbl> 38, 56, 60, 70, 74, 42, 27, 53, 47, 61, 30, 58, ~
## $ `Long Passing`         <dbl> 45, 78, 62, 67, 74, 56, 72, 83, 76, 52, 66, 82, ~
## $ `Ball Control`         <dbl> 73, 77, 85, 79, 84, 74, 65, 80, 81, 75, 69, 79, ~
## $ Acceleration           <dbl> 85, 71, 90, 78, 85, 84, 62, 70, 78, 88, 55, 70, ~
## $ `Sprint Speed`         <dbl> 82, 75, 92, 76, 90, 85, 73, 73, 75, 86, 58, 74, ~
## $ Agility                <dbl> 80, 72, 82, 79, 86, 82, 51, 75, 86, 77, 54, 55, ~
## $ Reactions              <dbl> 74, 77, 83, 77, 91, 73, 81, 81, 76, 80, 78, 80, ~
## $ Balance                <dbl> 72, 70, 80, 83, 78, 89, 37, 71, 89, 54, 61, 67, ~
## $ `Shot Power`           <dbl> 72, 73, 80, 72, 88, 65, 51, 78, 56, 83, 48, 74, ~
## $ Jumping                <dbl> 73, 82, 62, 64, 60, 85, 70, 74, 70, 73, 76, 74, ~
## $ Stamina                <dbl> 66, 76, 75, 80, 88, 80, 67, 86, 77, 87, 68, 90, ~
## $ Strength               <dbl> 59, 75, 80, 62, 64, 67, 87, 82, 56, 86, 78, 78, ~
## $ `Long Shots`           <dbl> 60, 64, 72, 76, 89, 62, 30, 72, 59, 71, 27, 65, ~
## $ Aggression             <dbl> 52, 71, 60, 57, 62, 77, 81, 82, 82, 78, 75, 84, ~
## $ Interceptions          <dbl> 19, 80, 24, 37, 39, 74, 80, 84, 65, 41, 78, 84, ~
## $ Positioning            <dbl> 74, 62, 80, 79, 91, 60, 44, 75, 73, 77, 40, 58, ~
## $ Vision                 <dbl> 58, 74, 74, 74, 84, 52, 59, 82, 81, 74, 57, 79, ~
## $ Penalties              <dbl> 68, 36, 63, 70, 73, 46, 40, 55, 51, 76, 38, 71, ~
## $ Composure              <dbl> 68, 77, 81, 75, 89, 66, 74, 82, 74, 82, 74, 83, ~
## $ `Defensive Awareness`  <dbl> 22, 76, 22, 49, 47, 70, 83, 80, 59, 36, 77, 82, ~
## $ `Standing Tackle`      <dbl> 19, 78, 24, 40, 34, 80, 84, 83, 66, 39, 81, 84, ~
## $ `Sliding Tackle`       <dbl> 15, 77, 21, 36, 33, 78, 80, 78, 62, 33, 78, 80, ~
## $ body_type              <fct> Lean, Lean, Lean, Normal, Unique, Lean, Normal, ~
## $ preferred_foot         <fct> Right, Left, Right, Left, Right, Left, Left, Rig~
```

Ao fim desse pré-processamento de dados ficamos com 40 colunas, sendo:

- 1 variável resposta: Potential;
- 39 variáveis independentes, sendo duas destas variáveis categóricas.

Agora, iremos selecionar cerca de 500 valores desses para realizar a modelagem. Isso ocorrerá de forma aleatória, chamada de Amostragem Aleatória Simples sem reposição.

Isso se justifica devido ao fato de que os dados obtidos são de todos os jogadores do jogo. Ou seja, temos aqui o conjunto universo, sendo assim, vamos analisar uma amostra desse universo. No processo de amostragem iremos setar uma semente, a saber: 22.

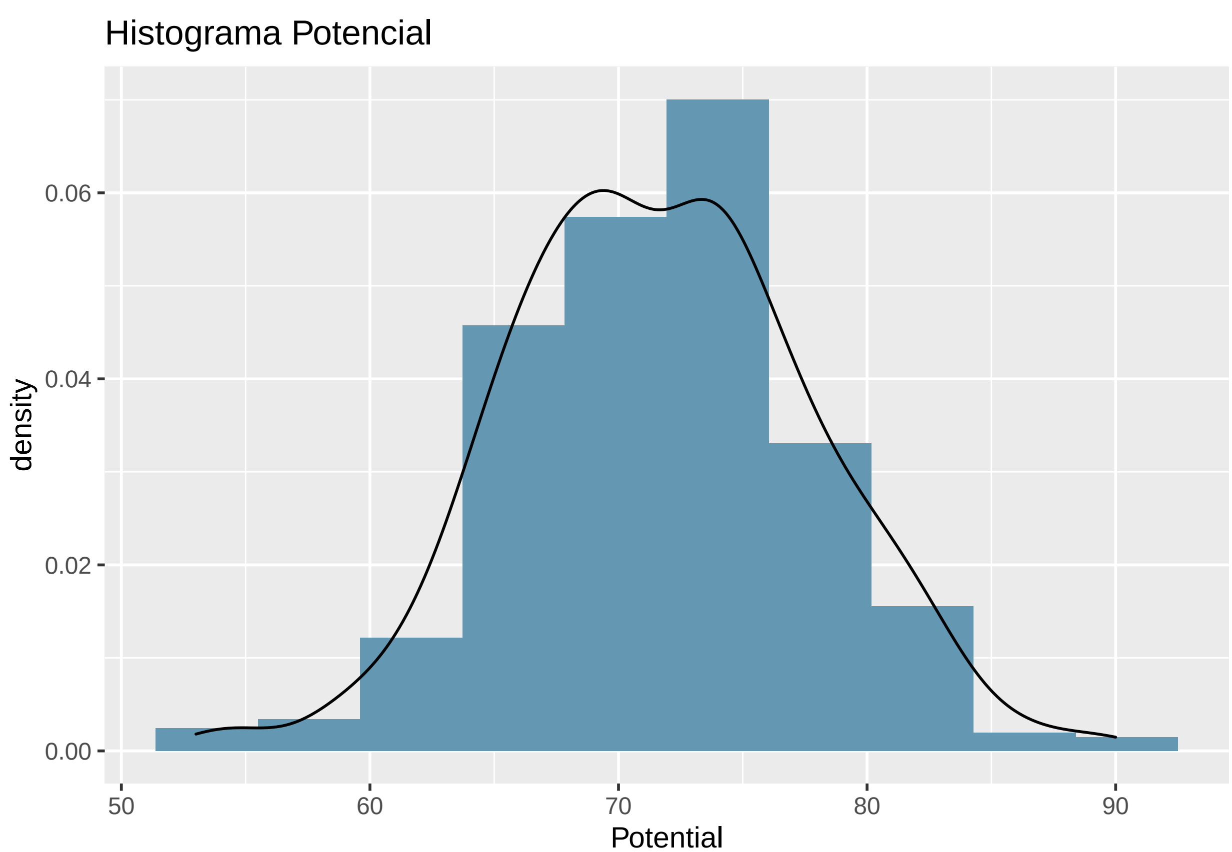
```
set.seed(22)
dados_m <- dados7[sample(nrow(dados7), 500, replace = FALSE), ]
```

Análise descritiva

Vamos primeiro a análise da variável resposta: Potential

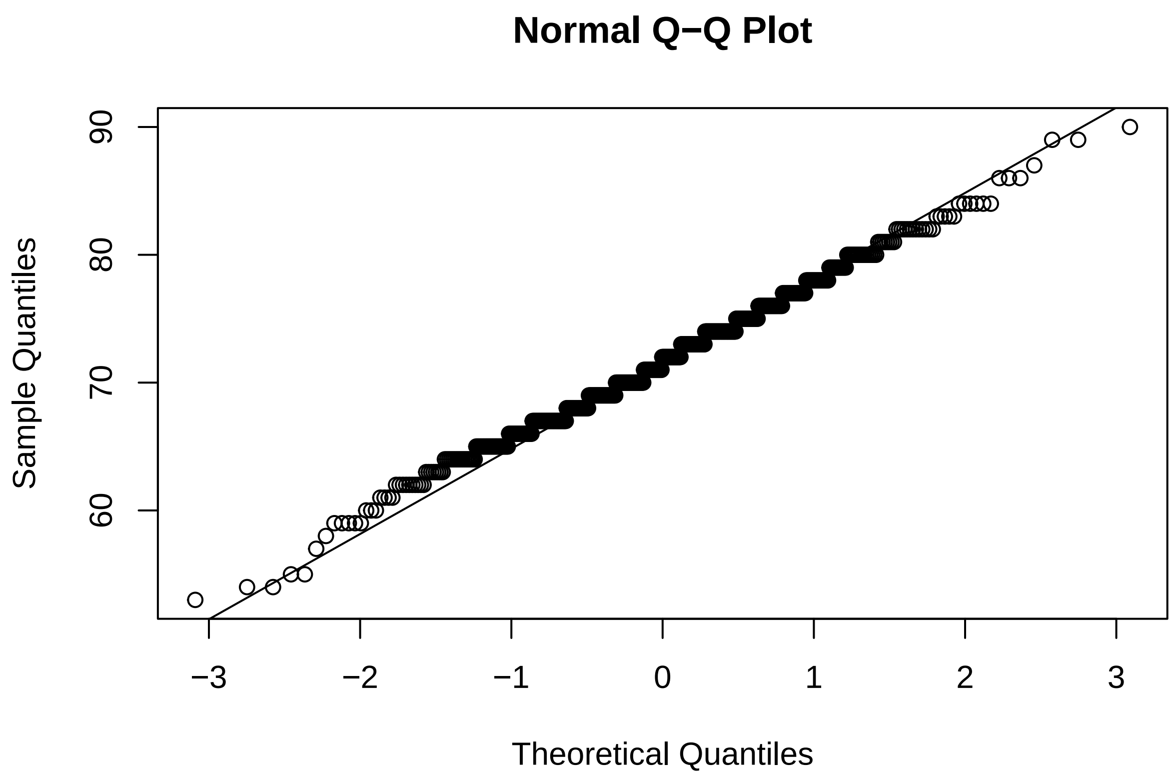
```
sturges_rule <- function(x) return(ceiling(log(length(x), 2)) + 1)

chart_hist_y <- ggplot(dados_m, aes(x = Potential)) +
                  geom_histogram(
                    aes(y = ..density..),
                    bins = sturges_rule(dados_m$Potential),
                    fill = "#6497b1"
                  ) +
                  geom_density() +
                  labs(title = "Histograma Potencial")
chart_hist_y
```



Pelo histograma acima podemos observar uma certa simetria, a qual pode ser um indicativo de que a resposta apresenta normalidade.

```r
qqnorm(dados_m$Potential)
qqline(dados_m$Potential)
```



Pelo QQ-Plot acima podemos observar que os valores amostrais figuram-se próximos dos valores teóricos. O que corrobora a normalidade.

Vamos aplicar testes de hipóteses sobre a variável resposta.

```r
testeNorm <- function(dados) {
  xb <- mean(dados)
  sx <- sd(dados)
  t1 <- ks.test(dados, "pnorm", xb, sx) # KS
  t2 <- lillie.test(dados) # Lilliefors
  t3 <- cvm.test(dados) # Cramér-von Mises
  if(length(dados) <= 5000) {
    t4 <- shapiro.test(dados) # Shapiro-Wilk
  } else {
    t4 <- list(method = "Shapiro-Wilk", statistic = NA, p.value = NA)
  }
  if(length(dados) <= 5000) {
    t5 <- sf.test(dados) # Shapiro-Francia
  } else {
    t5 <- list(method = "Shapiro-Francia", statistic = NA, p.value = NA)
  }
  t6 <- ad.test(dados) # Anderson-Darling
```

```
  # Tabela de resultados
  testes <- c(t1$method, t2$method, t3$method, t4$method, t5$method, t6$method)
  estt <- as.numeric(c(t1$statistic, t2$statistic, t3$statistic,
                       t4$statistic, t5$statistic, t6$statistic))
  valorp <- c(t1$p.value, t2$p.value, t3$p.value, t4$p.value, t5$p.value,
              t6$p.value)
  resultados <- cbind(estt, valorp)
  rownames(resultados) <- testes
  colnames(resultados) <- c("Estatística", "Valor - p")
  return(resultados)
}

testeNorm(dados_m$Potential)
```

```
## Warning in ks.test.default(dados, "pnorm", xb, sx): ties should not be present
## for the Kolmogorov-Smirnov test
```

```
##                                                Estatística    Valor - p
## Asymptotic one-sample Kolmogorov-Smirnov test   0.05733674 0.0746901798
## Lilliefors (Kolmogorov-Smirnov) normality test  0.05733674 0.0004769383
## Cramer-von Mises normality test                 0.15768727 0.0187940061
## Shapiro-Wilk normality test                     0.99503470 0.1087582735
## Shapiro-Francia normality test                  0.99520714 0.1145746293
## Anderson-Darling normality test                 0.90156961 0.0213658050
```

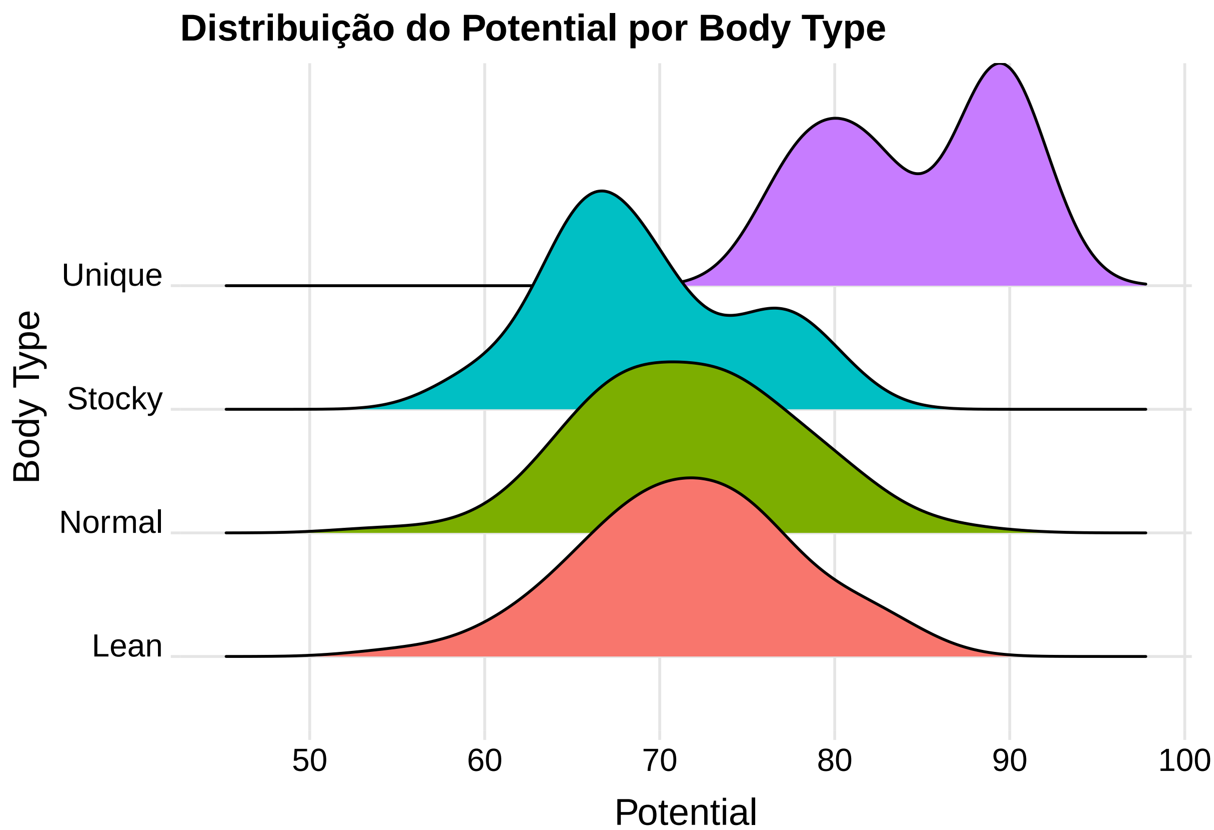
Pelos testes de Kolmogorov-Smirnov, Cramer-von Mises, Shapiro-Wilk, Shapiro-Francia e Anderson-Darling não rejeitamos a hipótese nula de normalidade à um nível de significancia de 1%.

Vamos a análise dos atributos body_type e preferred_foot

body_type

```
chart_bt <- ggplot(dados_m, aes(x = Potential, y = body_type, fill = body_type)) +
              geom_density_ridges() +
              theme_ridges(center_axis_labels = TRUE) +
              theme(legend.position = "none") +
              labs(title = "Distribuição do Potential por Body Type",
                   y = "Body Type", )
chart_bt
```

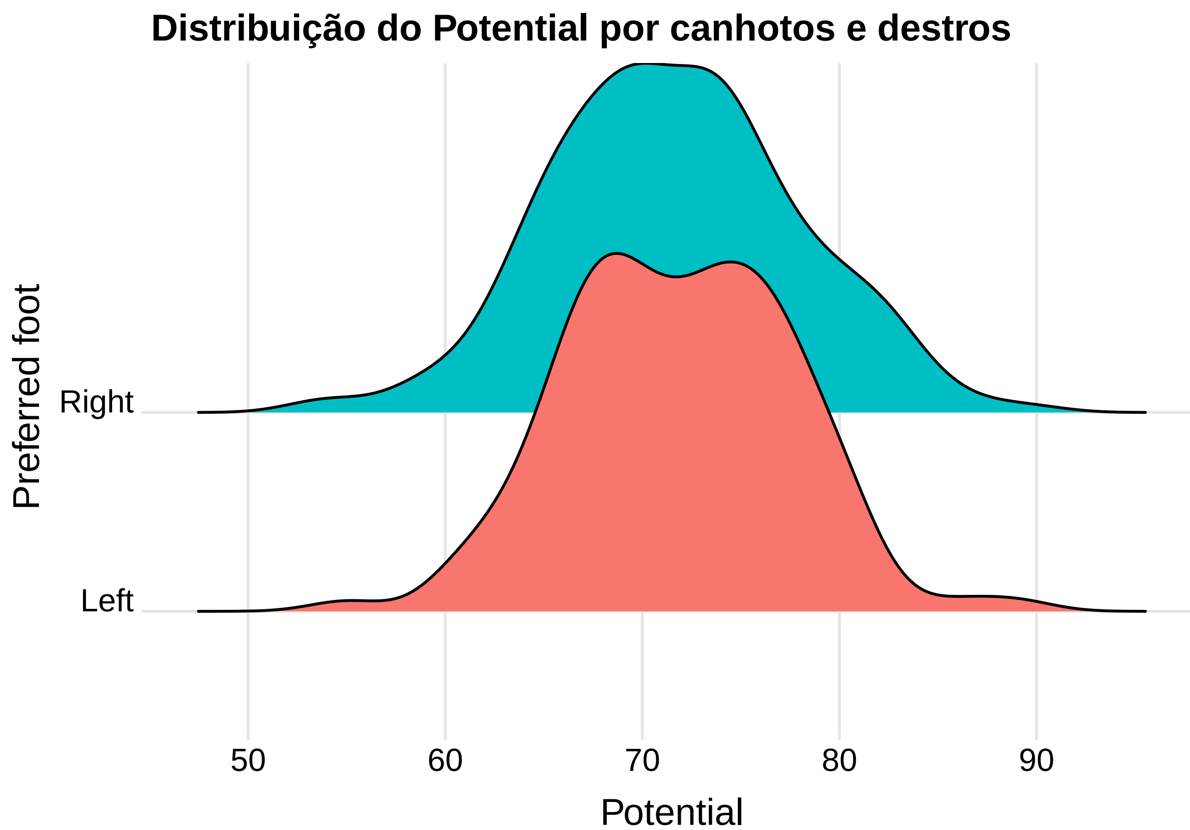
```
## Picking joint bandwidth of 2.59
```



preferred_foot

```
chart_pf <- ggplot(dados_m, aes(x = Potential, y = preferred_foot, fill = preferred_foot)) +
              geom_density_ridges() +
              theme_ridges(center_axis_labels = TRUE) +
              theme(legend.position = "none") +
              labs(title = "Distribuição do Potential por canhotos e destros",
                   y = "Preferred foot")
chart_pf
```

```
## Picking joint bandwidth of 1.84
```



Vamos agora, analisar o numero de outliers em cada variável. Para isso iremos utilizar o método do boxplot. AInda, vamos tomar algumas estatísticas descritivas.

```
numbers_outliers__ <- function(x) {
  pri <- quantile(x, 0.25)
  seg <- quantile(x, 0.5)
  ter <- quantile(x, 0.75)
  iqr <- abs(ter - pri)
  bounds <- c(pri - 1.5 * iqr, ter + 1.5 * iqr)
  num <- sum(x < bounds[1]) + sum(x > bounds[2])
  return(num)
}

dados_m_n <- dados_m %>% dplyr::select(where(is.numeric))

descdf <- t(apply(dados_m_n, 2, summary)) %>% as.data.frame()
sd_df <- apply(dados_m_n, 2, sd)
apresult <- apply(dados_m_n, 2, numbers_outliers__)

descdf <- descdf %>%
            mutate(SD  = sd_df, OutliersNumber = apresult)

descdf %>% arrange(-OutliersNumber)
```

```
##                        Min. 1st Qu. Median     Mean 3rd Qu.   Max.           SD
## Value                    45   425.00   825.0 2905.450 2100.00  59500 6.110043e+03
```

```
## Release clause        56  799.25 1600.0 5519.390 3925.00 105600 1.150775e+04
## Ball Control          -4   52.00   61.0   56.516   68.00     86 1.838798e+01
## Dribbling             -1   48.75   60.0   54.520   68.00     84 1.994606e+01
## Wage                   0    0.00    2.0    8.390    8.00    125 1.715221e+01
## Sprint Speed          -1   60.00   67.0   64.494   74.00     94 1.575800e+01
## Acceleration          -1   60.00   68.0   65.110   75.00     94 1.569811e+01
## Stamina               -1   54.00   64.0   61.024   72.25     92 1.682682e+01
## Short Passing          1   50.75   60.0   57.366   68.00     85 1.548974e+01
## Heading Accuracy      10   42.00   52.0   49.834   62.00     86 1.735306e+01
## Balance               -1   56.75   67.0   63.644   73.25     91 1.472069e+01
## Agility                1   54.00   65.0   62.402   74.00     92 1.561371e+01
## Age                   16   20.00   22.0   22.892   26.00     38 4.436180e+00
## Weight                57   70.00   75.0   74.598   79.00     95 6.742947e+00
## Reactions             25   53.00   60.0   59.608   67.00     87 1.033441e+01
## Jumping               -1   56.00   64.0   63.100   71.00     90 1.151378e+01
## Strength               7   55.00   63.0   62.306   71.00     90 1.250765e+01
## Composure              2   47.00   56.0   55.446   65.00     86 1.364711e+01
## Potential             53   67.00   72.0   71.672   76.00     90 6.138385e+00
## Overall               48   59.00   64.0   64.342   70.00     90 7.834526e+00
## Aggression             4   43.00   55.0   53.334   66.00     90 1.718433e+01
## Height               163  176.00  181.0  181.076  186.00    196 6.782199e+00
## Skill move             1    2.00    2.0    2.304    3.00      4 7.776848e-01
## Crossing              10   36.00   52.0   48.424   62.00     84 1.858944e+01
## Finishing             -4   29.00   49.5   44.756   60.00     82 1.995648e+01
## Volleys                5   29.00   42.0   40.734   54.00     82 1.769161e+01
## Curve                  4   34.75   47.0   46.098   60.00     90 1.835402e+01
## FK Accuracy            9   30.00   40.0   41.364   54.00     87 1.727612e+01
## Long Passing          12   41.00   55.0   51.968   64.00     84 1.579639e+01
## Shot Power            21   45.75   56.0   56.030   66.00     89 1.352623e+01
## Long Shots             1   30.00   48.0   44.940   61.00     87 2.012646e+01
## Interceptions          3   24.00   51.0   45.612   63.00     85 2.135785e+01
## Positioning            4   37.00   54.0   48.444   63.00     84 2.018896e+01
## Vision                20   43.00   54.0   53.218   64.00     86 1.387370e+01
## Penalties              8   38.00   48.0   46.636   59.00     80 1.611816e+01
## Defensive Awareness    2   26.00   50.0   45.228   63.00     85 2.107720e+01
## Standing Tackle        4   27.00   54.0   47.444   65.00     85 2.143786e+01
## Sliding Tackle         4   27.00   52.0   45.822   62.00     84 2.067000e+01
##                     OutliersNumber
## Value                           66
## Release clause                  66
## Ball Control                    60
## Dribbling                       59
## Wage                            58
## Sprint Speed                    47
## Acceleration                    41
## Stamina                         28
## Short Passing                   27
## Heading Accuracy                19
## Balance                         13
## Agility                          8
## Age                              3
## Weight                           3
## Reactions                        3
## Jumping                          3
```

```
## Strength                         3
## Composure                        3
## Potential                        2
## Overall                          1
## Aggression                       1
## Height                           0
## Skill move                       0
## Crossing                         0
## Finishing                        0
## Volleys                          0
## Curve                            0
## FK Accuracy                      0
## Long Passing                     0
## Shot Power                       0
## Long Shots                       0
## Interceptions                    0
## Positioning                      0
## Vision                           0
## Penalties                        0
## Defensive Awareness              0
## Standing Tackle                  0
## Sliding Tackle                   0
```

Vamos à análise da correlação dos dados.

Como temos 38 variáveis numéricas fica inviável a visualização da matriz de correlação. Sendo assim, vamos mostrar abaixo os pares de covariáveis que apresentam correlação maior do 0.8.

```r
# ++++++++++++++++++++++++++++
# flattenCorrMatrix
# ++++++++++++++++++++++++++++
# cormat : matrix of the correlation coefficients
# pmat : matrix of the correlation p-values
flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
  )
}

my_data <- dados_m %>% select(where(is.numeric), -Potential)
res2 <- rcorr(as.matrix(my_data))
correl <- flattenCorrMatrix(res2$r, res2$P)
correl__ <- correl %>% filter(cor > 0.8) %>% arrange(desc(cor))
correl__
```

```
##                   row              column       cor p
## 1               Value      Release clause 0.9927480 0
## 2      Standing Tackle      Sliding Tackle 0.9810794 0
## 3        Interceptions Defensive Awareness 0.9632444 0
## 4        Interceptions     Standing Tackle 0.9616005 0
```

```
## 5  Defensive Awareness     Standing Tackle 0.9550506 0
## 6  Defensive Awareness      Sliding Tackle 0.9473180 0
## 7        Interceptions      Sliding Tackle 0.9467415 0
## 8            Dribbling        Ball Control 0.9297863 0
## 9         Acceleration        Sprint Speed 0.9295093 0
## 10       Short Passing        Ball Control 0.9123597 0
## 11           Finishing          Long Shots 0.9119305 0
## 12           Dribbling         Positioning 0.9071881 0
## 13          Long Shots         Positioning 0.9004560 0
## 14       Short Passing        Long Passing 0.8989205 0
## 15           Finishing         Positioning 0.8957301 0
## 16           Finishing             Volleys 0.8813272 0
## 17           Dribbling          Long Shots 0.8789251 0
## 18             Volleys          Long Shots 0.8782424 0
## 19             Overall           Reactions 0.8767231 0
## 20               Curve         FK Accuracy 0.8704833 0
## 21            Crossing           Dribbling 0.8689752 0
## 22               Curve          Long Shots 0.8679582 0
## 23           Finishing           Penalties 0.8606834 0
## 24         FK Accuracy          Long Shots 0.8589270 0
## 25        Ball Control         Positioning 0.8588285 0
## 26             Volleys           Penalties 0.8549163 0
## 27       Short Passing           Dribbling 0.8547921 0
## 28           Dribbling               Curve 0.8543150 0
## 29           Finishing           Dribbling 0.8465172 0
## 30            Crossing               Curve 0.8451135 0
## 31             Volleys         Positioning 0.8441545 0
## 32             Volleys               Curve 0.8414419 0
## 33       Short Passing           Composure 0.8412524 0
## 34               Curve         Positioning 0.8410173 0
## 35        Ball Control          Long Shots 0.8354276 0
## 36            Crossing        Ball Control 0.8272858 0
## 37          Long Shots           Penalties 0.8259270 0
## 38            Crossing         Positioning 0.8256068 0
## 39           Finishing        Ball Control 0.8252017 0
## 40               Curve        Ball Control 0.8199830 0
## 41             Volleys         FK Accuracy 0.8145750 0
## 42          Aggression Defensive Awareness 0.8125247 0
## 43       Short Passing          Long Shots 0.8087612 0
## 44          Aggression       Interceptions 0.8081483 0
## 45       Short Passing         Positioning 0.8077109 0
## 46         Positioning           Penalties 0.8065623 0
## 47        Long Passing        Ball Control 0.8063204 0
## 48           Finishing               Curve 0.8055989 0
## 49            Crossing       Short Passing 0.8052653 0
## 50             Volleys           Dribbling 0.8005808 0
```

Sendo assim, segundo o critério da correlação vamos retirar dos dados as covariáveis abaixo.

```r
atri <- unique(correl__$column)
atri
```

```
##  [1] "Release clause"      "Sliding Tackle"      "Defensive Awareness"
##  [4] "Standing Tackle"     "Ball Control"        "Sprint Speed"
```

```
##  [7] "Long Shots"          "Positioning"         "Long Passing"
## [10] "Volleys"             "Reactions"           "FK Accuracy"
## [13] "Dribbling"           "Penalties"           "Curve"
## [16] "Composure"           "Interceptions"       "Short Passing"
```

```
dados_m1 <- dados_m[, -which(colnames(dados_m) %in% atri)]
# dados_m1n <- dados_m1 %>% dplyr::select(where(is.numeric))
# ggpairs(dados_m1n,
#         title = "Correlogram chart")
```

Seleção de variáveis:

Temos as seguintes variáveis consideradas.

```
glimpse(dados_m1)
```

```
## Rows: 500
## Columns: 22
## $ Age                <dbl> 29, 32, 32, 16, 24, 17, 17, 20, 23, 25, 20, 20, 18,~
## $ Overall            <dbl> 75, 80, 74, 55, 69, 54, 60, 52, 66, 69, 62, 59, 59,~
## $ Potential          <dbl> 75, 80, 74, 73, 72, 74, 71, 65, 71, 72, 69, 70, 72,~
## $ Value              <dbl> 5500, 16500, 2100, 325, 1900, 300, 475, 180, 1300, ~
## $ Wage               <dbl> 30, 58, 8, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0, 2, 0, 17, ~
## $ Height             <dbl> 176, 181, 180, 186, 169, 170, 188, 183, 170, 186, 1~
## $ Weight             <dbl> 69, 74, 76, 70, 65, 60, 75, 75, 74, 75, 71, 63, 82,~
## $ `Skill move`       <dbl> 4, 3, 1, 2, 4, 2, 2, 2, 3, 2, 2, 3, 2, 3, 2, 3, 3, ~
## $ Crossing           <dbl> 65, 79, 20, 55, 65, 41, 42, 28, 62, 72, 58, 61, 44,~
## $ Finishing          <dbl> 67, 76, 12, 39, 68, 46, 35, 21, 61, 22, 26, 46, 32,~
## $ `Heading Accuracy` <dbl> 48, 69, 13, 33, 50, 35, 59, 49, 59, 66, 55, 21, 55,~
## $ Acceleration       <dbl> 74, 73, 63, 68, 71, 75, 61, 57, 70, 44, 70, 89, 60,~
## $ Agility            <dbl> 77, 73, 69, 58, 75, 82, 66, 40, 66, 57, 63, 84, 48,~
## $ Balance            <dbl> 77, 71, 67, 60, 70, 83, 54, 59, 61, 58, 74, 77, 55,~
## $ `Shot Power`       <dbl> 71, 85, 51, 56, 65, 48, 60, 32, 54, 45, 31, 38, 36,~
## $ Jumping            <dbl> 73, 68, 78, 48, 60, 51, 58, 74, 56, 67, 59, 38, 70,~
## $ Stamina            <dbl> 55, 87, 40, 54, 69, 53, 53, 55, 58, 69, 60, 31, 60,~
## $ Strength           <dbl> 67, 73, 54, 54, 51, 34, 58, 61, 59, 75, 56, 29, 62,~
## $ Aggression         <dbl> 45, 74, 21, 32, 51, 42, 70, 45, 45, 73, 49, 26, 55,~
## $ Vision             <dbl> 77, 78, 60, 54, 73, 50, 40, 31, 66, 56, 41, 61, 40,~
## $ body_type          <fct> Lean, Normal, Lean, Lean, Lean, Normal, Lean, Lean,~
## $ preferred_foot     <fct> Right, Right, Right, Left, Right, Right, Right, Rig~
```

Dessas vamos utilizar métodos de seleção de modelos para tentar reduzir a dimensionalidade. Para isso, faremos uso das seguintes técnicas:

```
model0 <- lm(Potential ~ ., data = dados_m1)
```

```
vif_info <- car::vif(model0)
vif_df <- data.frame(
  Vif = vif_info[, "GVIF"]
)
vif_df %>% arrange(-Vif)
```

```
##                          Vif
## Finishing          6.042969
## Crossing           4.958085
## Overall            4.684914
## `Heading Accuracy` 4.418405
## Vision             4.084858
## Balance            4.073040
## Agility            4.069966
## Stamina            3.898837
## Acceleration       3.798938
## Height             3.538009
## Aggression         3.460813
## Value              3.377304
## `Shot Power`       3.372694
## Weight             3.324367
## Wage               3.270158
## Strength           3.230801
## `Skill move`       3.017702
## Age                2.162020
## Jumping            1.734118
## body_type          1.680911
## preferred_foot     1.113538
```

Atributo Finishing apresenta valor de VIF maior do que 5 então será retirado do modelo.

Agora vamos utilizar o stepAIC com direção both para escolher as cováriaveis

```
dados_m2 <- dados_m1 %>% dplyr::select(-Finishing)
model1 <- lm(Potential ~ ., data = dados_m2)
stepAIC(model1, direction = "both")
```

```
## Start:  AIC=977.6
## Potential ~ Age + Overall + Value + Wage + Height + Weight +
##     `Skill move` + Crossing + `Heading Accuracy` + Acceleration +
##     Agility + Balance + `Shot Power` + Jumping + Stamina + Strength +
##     Aggression + Vision + body_type + preferred_foot
##
##                      Df Sum of Sq    RSS    AIC
## - Balance             1       0.0 3222.2 975.61
## - `Shot Power`        1       0.0 3222.2 975.61
## - Value               1       0.8 3223.0 975.72
## - preferred_foot      1       1.3 3223.4 975.80
## - Vision              1       3.3 3225.5 976.11
## - Wage                1       4.5 3226.6 976.29
## - Agility             1       4.8 3227.0 976.35
## - Jumping             1       5.9 3228.1 976.52
## - Acceleration        1       7.9 3230.1 976.83
## <none>                            3222.2 977.60
## - Height              1      16.9 3239.1 978.22
## - `Heading Accuracy`  1      17.3 3239.4 978.27
## - Weight              1      19.4 3241.6 978.60
## - `Skill move`        1      20.1 3242.3 978.72
## - Strength            1      26.5 3248.7 979.70
## - Aggression          1      27.6 3249.8 979.87
```

```
## - body_type           3      63.8 3286.0  981.40
## - Stamina             1      39.4 3261.6  981.68
## - Crossing            1      71.6 3293.7  986.59
## - Age                 1    4696.3 7918.5 1425.18
## - Overall             1    5640.2 8862.4 1481.48
##
## Step:  AIC=975.61
## Potential ~ Age + Overall + Value + Wage + Height + Weight +
##     `Skill move` + Crossing + `Heading Accuracy` + Acceleration +
##     Agility + `Shot Power` + Jumping + Stamina + Strength + Aggression +
##     Vision + body_type + preferred_foot
##
##                      Df Sum of Sq    RSS     AIC
## - `Shot Power`        1       0.0 3222.2  973.61
## - Value               1       0.8 3223.0  973.72
## - preferred_foot      1       1.3 3223.5  973.80
## - Vision              1       3.3 3225.5  974.12
## - Wage                1       4.5 3226.7  974.30
## - Agility             1       5.8 3228.0  974.51
## - Jumping             1       5.9 3228.1  974.52
## - Acceleration        1       7.9 3230.1  974.83
## <none>                            3222.2  975.61
## - `Heading Accuracy`  1      17.4 3239.6  976.29
## - Height              1      18.5 3240.7  976.47
## - Weight              1      19.5 3241.7  976.62
## - `Skill move`        1      20.3 3242.5  976.74
## + Balance             1       0.0 3222.2  977.60
## - Strength            1      27.5 3249.7  977.86
## - Aggression          1      28.9 3251.1  978.07
## - body_type           3      63.8 3286.0  979.41
## - Stamina             1      39.4 3261.6  979.69
## - Crossing            1      71.6 3293.8  984.59
## - Age                 1    4711.1 7933.3 1424.11
## - Overall             1    5758.6 8980.8 1486.12
##
## Step:  AIC=973.61
## Potential ~ Age + Overall + Value + Wage + Height + Weight +
##     `Skill move` + Crossing + `Heading Accuracy` + Acceleration +
##     Agility + Jumping + Stamina + Strength + Aggression + Vision +
##     body_type + preferred_foot
##
##                      Df Sum of Sq    RSS     AIC
## - Value               1       0.8 3223.0  971.73
## - preferred_foot      1       1.3 3223.5  971.81
## - Vision              1       4.2 3226.5  972.27
## - Wage                1       4.6 3226.8  972.32
## - Agility             1       5.8 3228.1  972.52
## - Jumping             1       6.0 3228.3  972.55
## - Acceleration        1       7.9 3230.1  972.84
## <none>                            3222.2  973.61
## - `Heading Accuracy`  1      17.9 3240.1  974.38
## - Height              1      18.5 3240.8  974.48
## - Weight              1      19.5 3241.7  974.62
## - `Skill move`        1      20.7 3243.0  974.82
```

```
## + `Shot Power`        1       0.0 3222.2  975.61
## + Balance             1       0.0 3222.2  975.61
## - Strength            1      27.5 3249.8  975.87
## - Aggression          1      29.0 3251.2  976.09
## - body_type           3      63.8 3286.0  977.41
## - Stamina             1      39.7 3261.9  977.73
## - Crossing            1      71.6 3293.8  982.59
## - Age                 1    4723.7 7945.9 1422.90
## - Overall             1    5820.0 9042.3 1487.53
##
## Step:  AIC=971.73
## Potential ~ Age + Overall + Wage + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Acceleration + Agility +
##     Jumping + Stamina + Strength + Aggression + Vision + body_type +
##     preferred_foot
##
##                      Df Sum of Sq    RSS     AIC
## - preferred_foot      1       1.5 3224.5  969.95
## - Wage                1       3.9 3226.9  970.34
## - Vision              1       4.3 3227.3  970.40
## - Agility             1       5.7 3228.7  970.62
## - Jumping             1       6.5 3229.5  970.74
## - Acceleration        1       7.6 3230.6  970.90
## <none>                            3223.0  971.73
## - Height              1      18.6 3241.6  972.61
## - `Heading Accuracy`  1      18.7 3241.7  972.63
## - Weight              1      19.5 3242.5  972.74
## - `Skill move`        1      20.1 3243.1  972.84
## + Value               1       0.8 3222.2  973.61
## + `Shot Power`        1       0.0 3223.0  973.72
## + Balance             1       0.0 3223.0  973.73
## - Strength            1      28.1 3251.1  974.07
## - Aggression          1      29.5 3252.5  974.28
## - body_type           3      63.1 3286.1  975.42
## - Stamina             1      40.7 3263.6  976.00
## - Crossing            1      72.6 3295.6  980.87
## - Age                 1    5221.6 8444.6 1451.34
## - Overall             1    6443.1 9666.1 1518.89
##
## Step:  AIC=969.95
## Potential ~ Age + Overall + Wage + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Acceleration + Agility +
##     Jumping + Stamina + Strength + Aggression + Vision + body_type
##
##                      Df Sum of Sq    RSS     AIC
## - Wage                1       3.7 3228.1  968.52
## - Vision              1       4.0 3228.4  968.57
## - Agility             1       5.4 3229.8  968.79
## - Jumping             1       6.1 3230.6  968.90
## - Acceleration        1       7.4 3231.8  969.10
## <none>                            3224.5  969.95
## - `Heading Accuracy`  1      18.3 3242.8  970.79
## - Weight              1      19.1 3243.5  970.90
## - Height              1      19.1 3243.5  970.90
```

```
## - `Skill move`        1      20.3 3244.8  971.10
## + preferred_foot      1       1.5 3223.0  971.73
## + Value               1       1.0 3223.5  971.81
## + Balance             1       0.0 3224.4  971.95
## + `Shot Power`        1       0.0 3224.4  971.95
## - Aggression          1      28.9 3253.3  972.42
## - Strength            1      29.4 3253.8  972.49
## - body_type           3      64.5 3288.9  973.85
## - Stamina             1      39.9 3264.4  974.11
## - Crossing            1      71.3 3295.7  978.89
## - Age                 1    5226.2 8450.6 1449.70
## - Overall             1    6442.5 9666.9 1516.93
##
## Step:  AIC=968.52
## Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Acceleration + Agility +
##     Jumping + Stamina + Strength + Aggression + Vision + body_type
##
##                      Df Sum of Sq     RSS     AIC
## - Vision              1       4.6  3232.7  967.23
## - Agility             1       4.6  3232.7  967.23
## - Jumping             1       5.1  3233.3  967.32
## - Acceleration        1       7.3  3235.4  967.65
## <none>                             3228.1  968.52
## - `Heading Accuracy`  1      18.2  3246.3  969.33
## - Weight              1      18.2  3246.3  969.33
## - Height              1      19.9  3248.1  969.60
## - `Skill move`        1      20.7  3248.8  969.71
## + Wage                1       3.7  3224.5  969.95
## + preferred_foot      1       1.2  3226.9  970.34
## + Balance             1       0.1  3228.0  970.50
## + `Shot Power`        1       0.1  3228.0  970.51
## + Value               1       0.1  3228.1  970.52
## - Strength            1      30.1  3258.2  971.16
## - Aggression          1      31.2  3259.3  971.33
## - Stamina             1      44.1  3272.2  973.31
## - body_type           3      74.7  3302.9  973.97
## - Crossing            1      68.4  3296.5  977.00
## - Age                 1    5272.7  8500.8 1450.65
## - Overall             1    8041.5 11269.6 1591.63
##
## Step:  AIC=967.23
## Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Acceleration + Agility +
##     Jumping + Stamina + Strength + Aggression + body_type
##
##                      Df Sum of Sq     RSS     AIC
## - Agility             1       5.8  3238.5  966.14
## - Jumping             1       6.1  3238.8  966.18
## - Acceleration        1       8.9  3241.6  966.60
## <none>                             3232.7  967.23
## - `Heading Accuracy`  1      16.5  3249.2  967.77
## - Weight              1      17.2  3249.9  967.89
## - Height              1      18.6  3251.3  968.10
```

```
## + Vision              1       4.6  3228.1  968.52
## + Wage                1       4.3  3228.4  968.57
## + `Shot Power`        1       1.2  3231.5  969.04
## + preferred_foot      1       0.9  3231.8  969.10
## + Balance             1       0.1  3232.6  969.21
## + Value               1       0.1  3232.6  969.22
## - `Skill move`        1      29.9  3262.6  969.84
## - Aggression          1      30.6  3263.3  969.94
## - Strength            1      30.9  3263.6  969.99
## - Stamina             1      42.2  3274.9  971.72
## - body_type           3      77.9  3310.6  973.13
## - Crossing            1      63.8  3296.5  975.01
## - Age                 1    5272.6  8505.3 1448.92
## - Overall             1    9716.7 12949.4 1659.10
##
## Step:  AIC=966.14
## Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Acceleration + Jumping +
##     Stamina + Strength + Aggression + body_type
##
##                     Df Sum of Sq     RSS     AIC
## - Acceleration       1       4.6  3243.1  964.84
## - Jumping            1       5.1  3243.6  964.92
## <none>                            3238.5  966.14
## - `Heading Accuracy`  1      14.0  3252.6  966.30
## - Height             1      15.3  3253.8  966.49
## - Weight             1      19.0  3257.5  967.05
## + Agility            1       5.8  3232.7  967.23
## + Vision             1       5.8  3232.7  967.23
## + Wage               1       3.4  3235.2  967.61
## + `Shot Power`        1       1.6  3236.9  967.89
## + Balance            1       1.5  3237.0  967.90
## + preferred_foot     1       0.6  3238.0  968.05
## + Value              1       0.1  3238.5  968.13
## - Strength           1      28.7  3267.3  968.55
## - Aggression         1      30.9  3269.4  968.88
## - `Skill move`        1      35.0  3273.5  969.51
## - Stamina            1      40.3  3278.8  970.32
## - body_type          3      82.2  3320.8  972.67
## - Crossing           1      60.2  3298.8  973.35
## - Age                1    5276.5  8515.0 1447.49
## - Overall            1    9839.3 13077.8 1662.03
##
## Step:  AIC=964.84
## Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Jumping + Stamina + Strength +
##     Aggression + body_type
##
##                     Df Sum of Sq     RSS     AIC
## - Jumping            1       5.7  3248.8  963.72
## - `Heading Accuracy`  1      12.7  3255.9  964.80
## <none>                            3243.1  964.84
## - Height             1      17.6  3260.7  965.55
## - Weight             1      18.3  3261.4  965.65
```

```
## + Vision             1       6.6  3236.5  965.82
## + Acceleration       1       4.6  3238.5  966.14
## + Wage               1       3.7  3239.4  966.27
## + Agility            1       1.6  3241.6  966.60
## + `Shot Power`       1       1.6  3241.6  966.60
## + preferred_foot     1       0.5  3242.6  966.76
## + Balance            1       0.5  3242.6  966.76
## + Value              1       0.2  3243.0  966.82
## - Strength           1      27.9  3271.0  967.12
## - `Skill move`       1      32.2  3275.3  967.78
## - Aggression         1      34.3  3277.5  968.11
## - body_type          3      80.1  3323.2  971.04
## - Stamina            1      55.9  3299.1  971.39
## - Crossing           1      76.6  3319.8  974.52
## - Age                1    5369.6  8612.7 1451.20
## - Overall            1    9839.8 13083.0 1660.23
##
## Step:  AIC=963.72
## Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + `Heading Accuracy` + Stamina + Strength + Aggression +
##     body_type
##
##                      Df Sum of Sq     RSS     AIC
## - `Heading Accuracy`  1       9.9  3258.7  963.23
## <none>                             3248.8  963.72
## - Height             1      17.5  3266.3  964.41
## - Weight             1      18.0  3266.8  964.48
## + Vision             1       7.6  3241.2  964.54
## + Jumping            1       5.7  3243.1  964.84
## + Acceleration       1       5.2  3243.6  964.92
## + Wage               1       2.8  3246.0  965.28
## + `Shot Power`       1       2.3  3246.6  965.37
## + Agility            1       1.0  3247.8  965.57
## + preferred_foot     1       0.3  3248.5  965.67
## + Balance            1       0.1  3248.7  965.70
## + Value              1       0.0  3248.8  965.72
## - Strength           1      28.5  3277.3  966.08
## - Aggression         1      32.2  3281.0  966.64
## - `Skill move`       1      34.4  3283.2  966.99
## - body_type          3      81.0  3329.8  970.03
## - Stamina            1      67.1  3316.0  971.95
## - Crossing           1      71.0  3319.8  972.53
## - Age                1    5396.0  8644.8 1451.05
## - Overall            1   10022.1 13270.9 1665.36
##
## Step:  AIC=963.23
## Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + Stamina + Strength + Aggression + body_type
##
##                      Df Sum of Sq     RSS     AIC
## <none>                             3258.7  963.23
## + `Heading Accuracy`  1       9.9  3248.8  963.72
## - Weight             1      17.7  3276.4  963.94
## - Height             1      19.5  3278.2  964.22
```

```
## + Vision              1       5.1  3253.6  964.45
## - Strength            1      21.9  3280.6  964.59
## + Acceleration        1       3.8  3254.9  964.65
## + Wage                1       3.0  3255.7  964.78
## + Jumping             1       2.8  3255.9  964.80
## + `Shot Power`        1       2.7  3255.9  964.81
## + Agility             1       0.6  3258.1  965.15
## + preferred_foot      1       0.3  3258.4  965.19
## + Balance             1       0.3  3258.4  965.19
## + Value               1       0.0  3258.7  965.23
## - `Skill move`        1      43.8  3302.5  967.92
## - Aggression          1      53.2  3311.9  969.33
## - body_type           3      83.2  3341.8  969.84
## - Stamina             1      57.9  3316.6  970.04
## - Crossing            1      65.9  3324.5  971.24
## - Age                 1    5483.2  8741.9 1454.64
## - Overall             1   10047.3 13306.0 1664.68
```

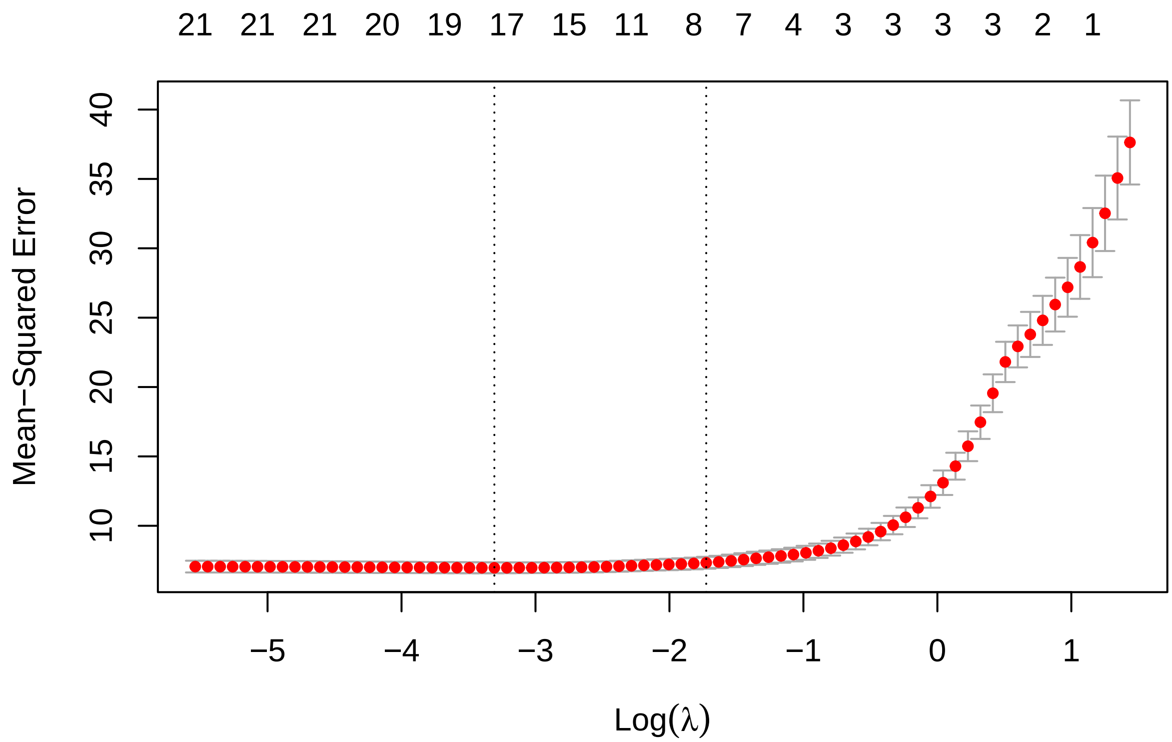
```
##
## Call:
## lm(formula = Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + Stamina + Strength + Aggression + body_type, data = dados_m2)
##
## Coefficients:
##     (Intercept)              Age          Overall           Height
##        32.87701         -1.01250          0.92636          0.05003
##          Weight     `Skill move`         Crossing          Stamina
##        -0.05042          0.57305         -0.03732         -0.03446
##        Strength       Aggression  body_typeNormal  body_typeStocky
##        -0.02827          0.03066         -0.46486          0.79779
## body_typeUnique
##         2.98276
```

Agora vamos utilizar a seleção de variáveis via Regressão Lasso.

```r
X_matrix <- model.matrix(model1)[, -1]
y_obser <- unlist(dados_m1[, "Potential"])

cv_lasso <- cv.glmnet(X_matrix, y_obser, alpha = 1,
                      nfolds = 10, type.measure = "mse")

plot(cv_lasso)
```




```
melhor_lambda <- cv_lasso$lambda.min
print(paste0("Melhor lambda: ", melhor_lambda))
```

```
## [1] "Melhor lambda: 0.0366273477613564"
```

```
best_lasso <- glmnet(X_matrix, y_obser, alpha = 1, lambda = melhor_lambda)
# Coeficiente obtidos
coef(best_lasso)
```

```
## 23 x 1 sparse Matrix of class "dgCMatrix"
##                               s0
## (Intercept)         37.603299828
## Age                 -0.991987888
## Overall              0.897361281
## Value                .
## Wage                 0.006900930
## Height               0.021709932
## Weight              -0.029208001
## `Skill move`         0.361237385
## Crossing            -0.029136286
## `Heading Accuracy`   0.009644934
## Acceleration        -0.006371706
## Agility              .
## Balance              .
## `Shot Power`         .
```

```
## Jumping              -0.003995910
## Stamina              -0.027146893
## Strength             -0.021391855
## Aggression            0.016538512
## Vision                0.006603336
## body_typeNormal      -0.483406670
## body_typeStocky       0.243544946
## body_typeUnique       2.279282251
## preferred_footRight  .
```

```
names(best_lasso$beta[, 1])[best_lasso$beta[, 1] != 0]
```

```
##  [1] "Age"                 "Overall"             "Wage"
##  [4] "Height"              "Weight"              "‘Skill move‘"
##  [7] "Crossing"            "‘Heading Accuracy‘" "Acceleration"
## [10] "Jumping"             "Stamina"             "Strength"
## [13] "Aggression"          "Vision"              "body_typeNormal"
## [16] "body_typeStocky"     "body_typeUnique"
```

O modelo que a intersecção dos métodos supracitados é com as seguintes covariáveis: Age, Overall, Height, Weight, Skill move, Crossing, Stamina, Strength, Aggression e body_type.

```
model3 <- lm(Potential ~ Age + Overall + Height + Weight + `Skill move` +
             Crossing + Stamina + Strength + Aggression + body_type, data = dados_m2)
```

Vamos agora ao cálculo do modelo com maior R2 ajutado.

```
X_ <- model.matrix(model3)[, -1]
models__ <- leaps(X_, y_obser, method= "adjr2")

modelos <- leaps(X_, y_obser)
saida <- modelos$which

colnames(saida) <- colnames(X_)
resp <- as.matrix(apply(saida, 1, create_models <- function(param) {
    texto = ""
    for (i in 1:length(param)) {
        if (param[i] != FALSE) {
            if (texto == "") {
              texto = paste(names(param)[i], sep="")
            } else {
              texto = paste(texto, " + ", names(param)[i], sep="")
            }
        }
    }
    return (texto)
}))

dfR2 <- data.frame(Model = resp, R2Ajs = models__$adjr2)
row.names(dfR2) <- NULL
print(paste0("Melho modelo: ", dfR2[which.max(dfR2$R2Ajs), "Model"]))
```

```
## [1] "Melho modelo: Age + Overall + Height + Weight + ‘Skill move‘ + Crossing + Stamina + Strength + A
```

```
dfR2 %>% arrange(-R2Ajs) %>% head(5)
```

```
##
## 1 Age + Overall + Height + Weight + ‘Skill move‘ + Crossing + Stamina + Strength + Aggression + body_
## 2                   Age + Overall + Height + Weight + ‘Skill move‘ + Crossing + Stamina + Strength +
## 3                                    Age + Overall + ‘Skill move‘ + Crossing + Stamina + Strength +
## 4                          Age + Overall + Height + ‘Skill move‘ + Crossing + Stamina + Strength +
## 5                          Age + Overall + Weight + ‘Skill move‘ + Crossing + Stamina + Strength +
##       R2Ajs
## 1 0.8224158
## 2 0.8222705
## 3 0.8220090
## 4 0.8219334
## 5 0.8218631
```

Modelo escolhido

De acordo com todos os métodos supramencionados temos o modelo final.

```
modelf0 <- lm(Potential ~ Age + Overall + Height + Weight + `Skill move` +
             Crossing + Stamina + Strength + Aggression + body_type,
             data = dados_m2)
summary(modelf0)
```

```
##
## Call:
## lm(formula = Potential ~ Age + Overall + Height + Weight + ‘Skill move‘ +
##     Crossing + Stamina + Strength + Aggression + body_type, data = dados_m2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8464 -1.7024 -0.3015  1.3900  7.9356
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     32.87701    4.43026   7.421 5.22e-13 ***
## Age             -1.01250    0.03537 -28.626  < 2e-16 ***
## Overall          0.92636    0.02391  38.750  < 2e-16 ***
## Height           0.05003    0.02930   1.707  0.08842 .
## Weight          -0.05042    0.03102  -1.625  0.10478
## ‘Skill move‘     0.57305    0.22389   2.560  0.01078 *
## Crossing        -0.03732    0.01189  -3.138  0.00181 **
## Stamina         -0.03446    0.01172  -2.941  0.00343 **
## Strength        -0.02827    0.01561  -1.811  0.07082 .
## Aggression       0.03066    0.01087   2.820  0.00500 **
## body_typeNormal -0.46486    0.26006  -1.788  0.07447 .
## body_typeStocky  0.79779    0.67446   1.183  0.23745
## body_typeUnique  2.98276    1.35527   2.201  0.02821 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
```

```
## Residual standard error: 2.587 on 487 degrees of freedom
## Multiple R-squared:  0.8267, Adjusted R-squared:  0.8224
## F-statistic: 193.6 on 12 and 487 DF,  p-value: < 2.2e-16
```

Note que a soma das estimativas de Height e Weight são muito próximas de zero. Sendo assim, vamos testar se essa soma de fato é igual a zero.

```
linearHypothesis(modelf0, c("Height + Weight = 0"))
```

```
## Linear hypothesis test
##
## Hypothesis:
## Height  + Weight = 0
##
## Model 1: restricted model
## Model 2: Potential ~ Age + Overall + Height + Weight + `Skill move` +
##     Crossing + Stamina + Strength + Aggression + body_type
##
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1    488 3258.7
## 2    487 3258.7  1 0.0011506 2e-04 0.9895
```

Com um nível de significância de 1% não rejeitamos a hipótese nula e concluímos a favor da soma é igual a zero. Sendo assim, vamos retirar essas duas colunas.

```
modelf2 <- lm(Potential ~ Age + Overall + `Skill move` +
             Crossing + Stamina + Strength + Aggression + body_type,
             data = dados_m2)
summary(modelf2)
```

```
##
## Call:
## lm(formula = Potential ~ Age + Overall + `Skill move` + Crossing +
##     Stamina + Strength + Aggression + body_type, data = dados_m2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.0564 -1.6796 -0.2969  1.5241  8.0950
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     38.40785    1.04099  36.896  < 2e-16 ***
## Age             -1.01971    0.03502 -29.120  < 2e-16 ***
## Overall          0.92597    0.02339  39.580  < 2e-16 ***
## `Skill move`     0.56985    0.22246   2.562  0.01072 *
## Crossing        -0.03842    0.01161  -3.309  0.00101 **
## Stamina         -0.03400    0.01149  -2.960  0.00323 **
## Strength        -0.02744    0.01239  -2.214  0.02732 *
## Aggression       0.03117    0.01078   2.893  0.00399 **
## body_typeNormal -0.59444    0.25037  -2.374  0.01797 *
## body_typeStocky  0.43967    0.64906   0.677  0.49848
## body_typeUnique  2.80699    1.35324   2.074  0.03858 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.591 on 489 degrees of freedom
## Multiple R-squared:  0.8254, Adjusted R-squared:  0.8218
## F-statistic: 231.1 on 10 and 489 DF,  p-value: < 2.2e-16
```

Observamos a mesma relação supracitada com as variáveis Aggression e Crossing. Vamos ao teste.

```r
linearHypothesis(modelf2, c("Aggression + Crossing = 0"))
```

```
## Linear hypothesis test
##
## Hypothesis:
## Crossing  + Aggression = 0
##
## Model 1: restricted model
## Model 2: Potential ~ Age + Overall + `Skill move` + Crossing + Stamina +
##     Strength + Aggression + body_type
##
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    490 3285.0
## 2    489 3283.2  1    1.8158 0.2705 0.6033
```

Com um nível de significância de 1% não rejeitamos a hipótese nula e concluímos a favor da soma é igual a zero. Sendo assim, vamos retirar essas duas colunas.

```r
modelf3 <- lm(Potential ~ Age + Overall + `Skill move` + Stamina + Strength +
                body_type,
              data = dados_m2)
summary(modelf3)
```

```
##
## Call:
## lm(formula = Potential ~ Age + Overall + `Skill move` + Stamina +
##     Strength + body_type, data = dados_m2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1010 -1.7621 -0.4499  1.5400  8.5784
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     38.342168   1.014408  37.798   <2e-16 ***
## Age             -1.017813   0.035489 -28.679   <2e-16 ***
## Overall          0.917279   0.022845  40.153   <2e-16 ***
## `Skill move`     0.250321   0.196588   1.273   0.2035
## Stamina         -0.035573   0.009068  -3.923   0.0001 ***
## Strength        -0.007733   0.011510  -0.672   0.5020
## body_typeNormal -0.600483   0.253848  -2.366   0.0184 *
## body_typeStocky  0.258818   0.656479   0.394   0.6936
## body_typeUnique  2.596246   1.367599   1.898   0.0582 .
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.627 on 491 degrees of freedom
## Multiple R-squared:  0.8197, Adjusted R-squared:  0.8168
## F-statistic: 279.1 on 8 and 491 DF,  p-value: < 2.2e-16
```

Pode ser que Age e Overall também somem 0. Vamos testar isso.

```
linearHypothesis(modelf3, c("Age + Overall = 0"))
```

```
## Linear hypothesis test
##
## Hypothesis:
## Age  + Overall = 0
##
## Model 1: restricted model
## Model 2: Potential ~ Age + Overall + `Skill move` + Stamina + Strength +
##     body_type
##
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    492 3463.7
## 2    491 3389.2  1    74.566 10.803 0.001086 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Com um nível de significância de 1% rejeitamos a hipótese nula e concluímos a favor da soma não é igual a zero.

Em razão disso temos o modelo

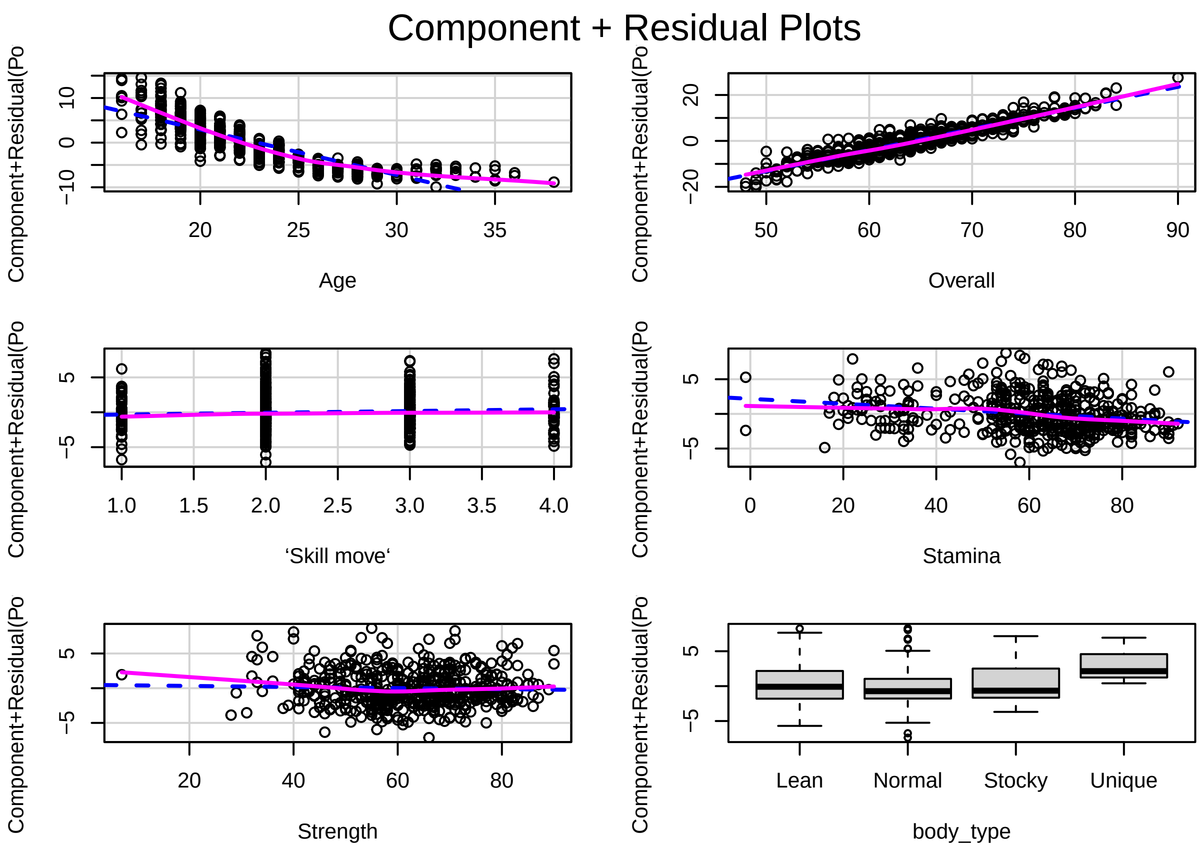
```
modelf <- lm(Potential ~ Age + Overall + `Skill move` + Stamina + Strength +
                         body_type,
             data = dados_m2)
summary(modelf)
```

```
##
## Call:
## lm(formula = Potential ~ Age + Overall + `Skill move` + Stamina +
##     Strength + body_type, data = dados_m2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1010 -1.7621 -0.4499  1.5400  8.5784
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     38.342168   1.014408  37.798   <2e-16 ***
## Age             -1.017813   0.035489 -28.679   <2e-16 ***
## Overall          0.917279   0.022845  40.153   <2e-16 ***
## `Skill move`     0.250321   0.196588   1.273   0.2035
## Stamina         -0.035573   0.009068  -3.923   0.0001 ***
## Strength        -0.007733   0.011510  -0.672   0.5020
```

```
## body_typeNormal -0.600483   0.253848  -2.366   0.0184 *
## body_typeStocky  0.258818   0.656479   0.394   0.6936
## body_typeUnique  2.596246   1.367599   1.898   0.0582 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.627 on 491 degrees of freedom
## Multiple R-squared:  0.8197, Adjusted R-squared:  0.8168
## F-statistic: 279.1 on 8 and 491 DF,  p-value: < 2.2e-16
```

Análise da linearidade

```
crPlots(modelf)
```



A linha azul mostra os residuos esperados se a relação entre o preditor e a resposta for linear. A linha rosa mostra os residuos ajustados. As variáveis apresentam essas duas retas próximas umas das outras.

```
testeNorm(modelf$residuals)
```

```
##                                              Estatística    Valor - p
## Asymptotic one-sample Kolmogorov-Smirnov test   0.07612398 6.086308e-03
## Lilliefors (Kolmogorov-Smirnov) normality test  0.07612398 2.653365e-07
## Cramer-von Mises normality test                 0.58220241 5.112142e-07
## Shapiro-Wilk normality test                     0.97992527 2.202474e-06
## Shapiro-Francia normality test                  0.98006259 7.467180e-06
## Anderson-Darling normality test                 3.13780729 7.080133e-08
```

Ainda, o teste de normalidade dos residuos é rejeitado a um nível de significância de 1% todos os testes.

Percebemos que ajustar um modelo de sem a variável "Age" faz com que os testes de normalidade dos residuos do modelo não seja rejeitado. Todavia a retirada dessa variável provoca uma queda relevante no R2 ajustado, como visto abaixo.

```
modelf1 <- lm(Potential ~  Overall + Height + Weight + `Skill move` +
             Crossing + Stamina + Strength + Aggression + body_type,
             data = dados_m2)
summary(modelf1)
```

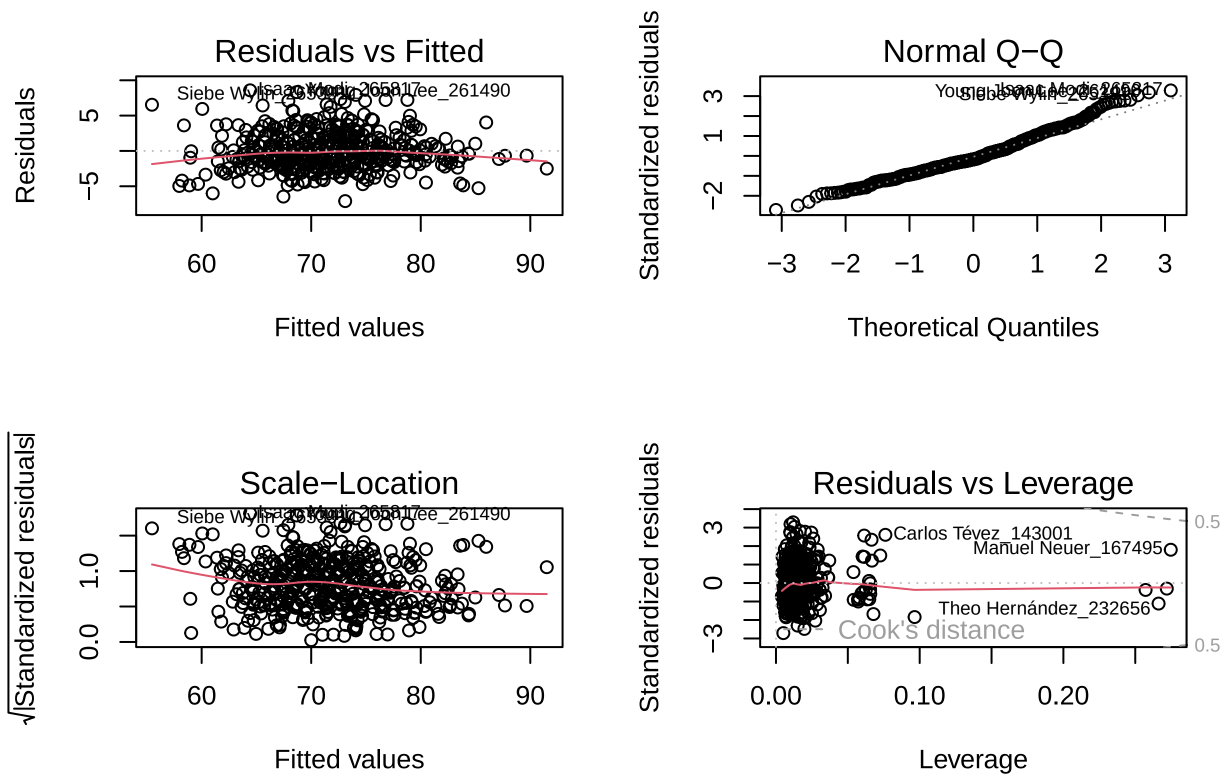
```
##
## Call:
## lm(formula = Potential ~ Overall + Height + Weight + `Skill move` +
##     Crossing + Stamina + Strength + Aggression + body_type, data = dados_m2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.5754  -2.8319  -0.2657   3.0984  11.5216
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     14.87991    7.17543   2.074 0.038628 *
## Overall          0.57403    0.03353  17.118  < 2e-16 ***
## Height           0.17094    0.04745   3.603 0.000347 ***
## Weight          -0.07956    0.05073  -1.568 0.117485
## `Skill move`     1.17237    0.36472   3.214 0.001394 **
## Crossing        -0.01891    0.01943  -0.973 0.330923
## Stamina         -0.02837    0.01917  -1.480 0.139393
## Strength        -0.11340    0.02508  -4.522 7.69e-06 ***
## Aggression       0.03720    0.01778   2.092 0.036963 *
## body_typeNormal -0.14267    0.42511  -0.336 0.737317
## body_typeStocky -1.83353    1.09326  -1.677 0.094158 .
## body_typeUnique  2.44176    2.21728   1.101 0.271333
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.232 on 488 degrees of freedom
## Multiple R-squared:  0.5351, Adjusted R-squared:  0.5246
## F-statistic: 51.05 on 11 and 488 DF,  p-value: < 2.2e-16
```

```
testeNorm(modelf1$residuals)
```

```
##                                                 Estatística Valor - p
## Asymptotic one-sample Kolmogorov-Smirnov test   0.02948182 0.7775738
## Lilliefors (Kolmogorov-Smirnov) normality test  0.02948182 0.3631109
## Cramer-von Mises normality test                 0.07910189 0.2123790
## Shapiro-Wilk normality test                     0.99690737 0.4621337
## Shapiro-Francia normality test                  0.99759777 0.6227527
## Anderson-Darling normality test                 0.43742825 0.2947194
```

Sendo assim, mesmo que a normalidade seja rejeitada nos testes de hipóteses optaremos por permanecer com o modelo contendo a variável "Age".

```
par(mfrow = c(2, 2))
plot(modelf)
```



- Resíduos vs Ajustados: tal gráfico é utilizado para verificar a premissa de linearidade. Caso os resíduos estejam distribuídos em torno de uma linha horizontal sem padrões distintos, tem-se um indicativo de haver uma relação linear. É o que ocorre no gráfico acima
- QQ normal: é utilizado para verificar a suposição de normalidade dos resíduos. Caso os pontos estejam dispostos próximos à reta tracejada, tem-se um indicativo de normalidade. Pelo resultdo acima temos que alguns pontos da reta superior figuram-se pouco acima da reta tracejada.
- Escala-Locação: é útil para verificar a homocedasticidade dos resíduos. Se os resíduos forem espalhados aleatoriamente sem comportamento atípico, a suposição está satisfeita. É o que ocorre nesse caso.
- Resíduos vs Leverage: tal gráfico é usado para identificar pontos influentes. Valores influentes são valores extremos que podem influenciar nos resultados da regressão quando incluídos ou excluídos da análise.

Testes de Homocedasticidade

```
print(bptest(modelf))
```

```
##
```

```
##  studentized Breusch-Pagan test
##
## data:  modelf
## BP = 20.654, df = 8, p-value = 0.008127
```

```
print(gqtest(modelf, fraction = 0.2, alternative = 'two.sided'))
```

```
##
##  Goldfeld-Quandt test
##
## data:  modelf
## GQ = 0.74731, df1 = 191, df2 = 191, p-value = 0.0448
## alternative hypothesis: variance changes from segment 1 to 2
```

Pelo teste de Goldfeld-Quandt nao rejeitamos a Homocedasticidade a 1% de significancia.

Agora, vamos observar os pontos de alavanca.

```
leveragePoints <- function(model) {
  hii <- hatvalues(model)
  p <- length(model$coefficients)
  n <- length(hii)
  dfhii <- data.frame(hii = hii) %>% dplyr::arrange(-hii)
  out <- dfhii %>% dplyr::filter(hii > 2 * (p/n))
  return(out)
}
leverage_result <- leveragePoints(modelf)
leverage_result
```

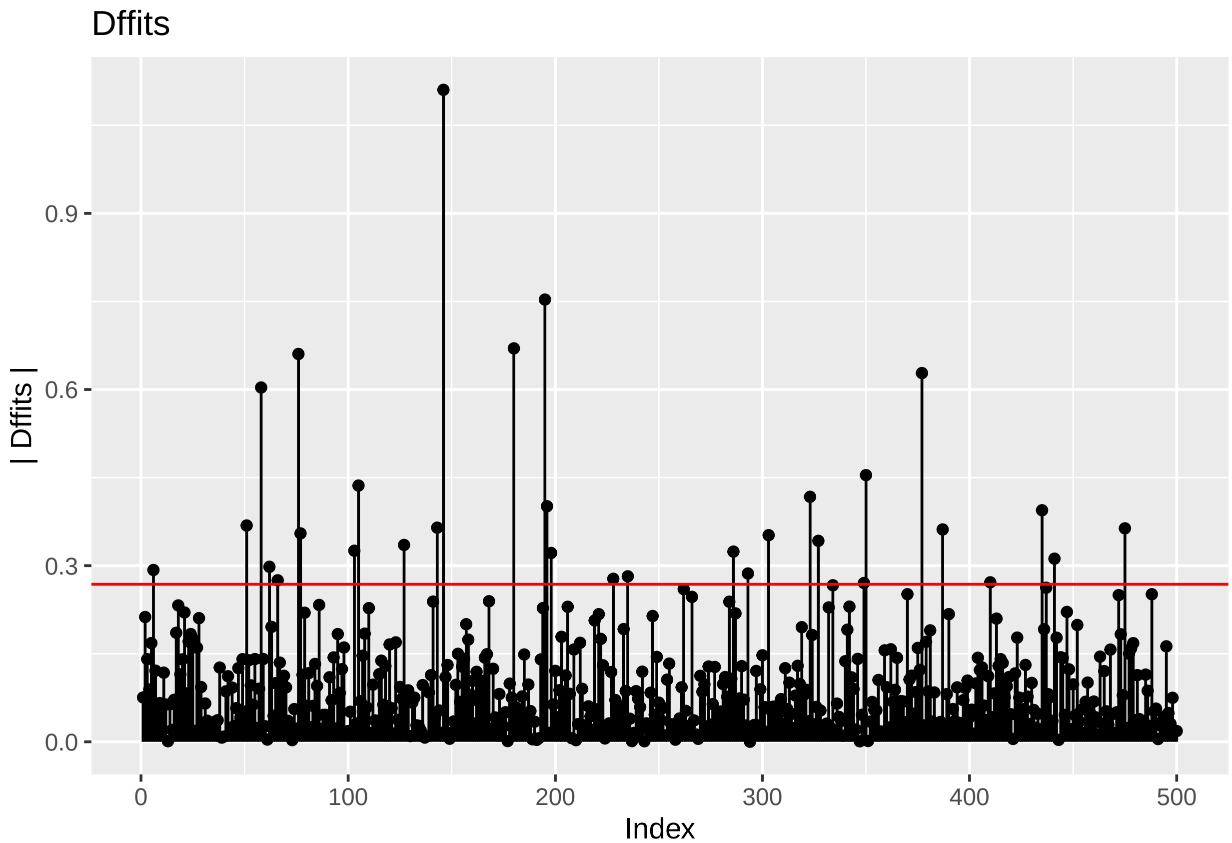
```
##                                                 hii
## Manuel Neuer_167495                      0.27471142
## Mesut Özil_176635                        0.27194784
## Theo Hernández_232656                    0.26623227
## Georginio Wijnaldum_181291               0.25719480
## Andreas Hugo Hoelgebaum Pereira_208450 0.09648214
## Carlos Tévez_143001                      0.07611817
## Scott Carson_157804                      0.07258420
## Lucas Margueron_257142                   0.06783424
## Imam Jagne_259290                        0.06677838
## Yuber Mosquera_253353                    0.06650672
## Anthony Ralston_234072                   0.06564449
## Suk Young Cho_255551                     0.06563450
## Miguel Angel Navarro_255457              0.06552063
## Benjamin Mendy_204884                    0.06495493
## Neil Etheridge_193186                    0.06474427
## Theo Gunnar Martens_263069               0.06340127
## Gabriel Florentín_253135                 0.06257937
## Dionicio Pérez_254954                    0.06141225
## Masahiro Okamoto_266276                  0.06138262
## Henry Martín_224151                      0.06063666
## Lucas Gómez_253920                       0.06031363
## Francisco Parra_254726                   0.05953242
## Pierce Sweeney_207646                    0.05790429
```

```
## Jhon Jairo Rodríguez_266234          0.05726481
## Gil Burón_216394                     0.05708679
## Milan Makarić_263493                 0.05410948
## Hubert Turski_252511                 0.05391578
## Tim Krul_170597                      0.03708529
## Marco Silvestri_190745               0.03640709
## Pedro Ortíz_250797                   0.03622554
```

Vamos a visualização dos pontos influentes com a medida DFFIT

```r
dffitAnalysis <- function(model, y_values) {
  dffit_vec <- dffits(model)
  dfdffit <- data.frame(
    Index = seq(1, length(dffit_vec)),
    Dffits = abs(dffit_vec)
  )
  p <- length(model$coefficients)
  n <- nrow(dfdffit)
  corte <- 2 * sqrt(p/n)
  chart <- ggplot(dfdffit, aes(x = Index, y = Dffits)) +
             geom_point() +
             geom_segment(aes(x = Index, xend = Index, y = 0, yend = Dffits)) +
             geom_hline(yintercept = corte, color = "red") +
             ggtitle("Dffits") + ylab("| Dffits |")
  dfdffit1 <- dfdffit %>% filter(Dffits > corte)
  out <- dfdffit1 %>% mutate(y = y_values[dfdffit1$Index]) %>% arrange(-Dffits)
  return(list(threshold = corte, chartDffits = chart, table = out))
}
dffit_result <- dffitAnalysis(modelf, dados_m[, "Potential"])
dffit_result
```

```
## $threshold
## [1] 0.2683282
##
## $chartDffits
```



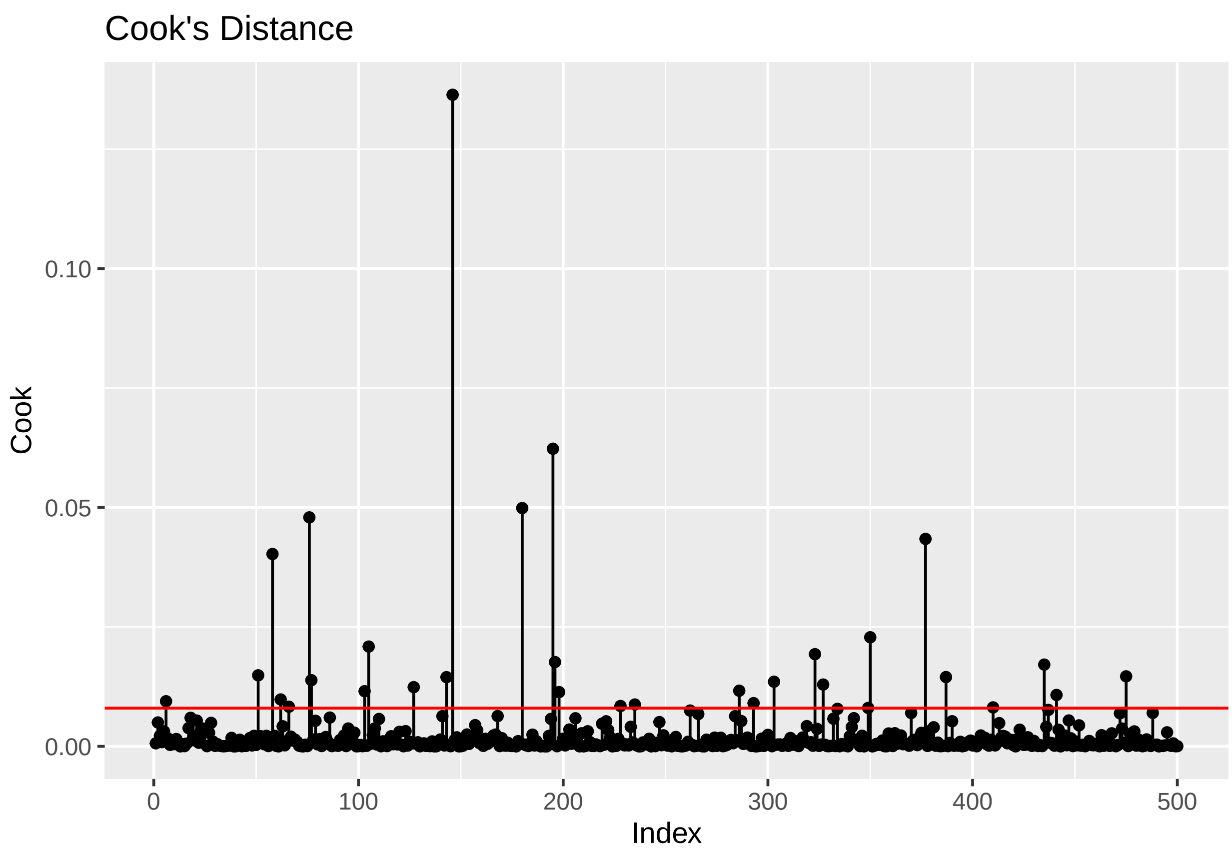

```
##
## $table
##                                         Index    Dffits  y
## Manuel Neuer_167495                       146 1.1104411 90
## Carlos Tévez_143001                       195 0.7532506 78
## Theo Hernández_232656                     180 0.6701330 89
## Masahiro Okamoto_266276                    76 0.6605472 62
## Yuber Mosquera_253353                     377 0.6280714 66
## Andreas Hugo Hoelgebaum Pereira_208450     58 0.6034306 79
## Lucas Margueron_257142                    350 0.4541720 65
## Diego Chará_203067                        105 0.4364383 75
## Scott Carson_157804                       323 0.4172724 66
## Cole Palmer_257534                        196 0.4011890 86
## Diego Godín_182493                        435 0.3943667 78
## Enrique Herrero García_263104              51 0.3684575 80
## Isaac Modi_265817                         143 0.3647089 73
## Lucas Gómez_253920                        475 0.3635673 67
## Dionicio Pérez_254954                     387 0.3616745 65
## Tommy Jackson_264636                       77 0.3549370 61
## Siebe Wylin_265391                        303 0.3519515 82
## Adil Aouchiche_253102                     327 0.3422896 80
## Graziano Pellè_164376                     127 0.3353005 74
## Young Joon Lee_261490                     103 0.3254226 77
## Imam Jagne_259290                         286 0.3239072 75
## Leandro Soria_265512                      198 0.3216288 80
## Christian Gallardo_265495                 441 0.3118844 78
```

```
## Fernando Reges Mouta_184134              62 0.2980641 84
## Tyler Goodrham_263861                     6 0.2926340 74
## Xiangshuo Zhang_261907                  293 0.2866541 55
## Ardon Jashari_257186                    235 0.2818113 74
## Hannibal Mejbri_258171                  228 0.2776399 84
## Luke Mbete_261621                        66 0.2750781 82
## Huapeng Wang_257689                     410 0.2716468 54
## Bendik Brevik_265692                    349 0.2704657 72
```

Como podemos ver temos cerca de 31 valores influentes segundo o DFFIT.

Agora, vamos então observar pontos influentes utilizando a distância Dcook

```r
dcookAnalysis <- function(model, y_values) {
  cook_vec <- cooks.distance(model)
  dfcook <- data.frame(
    Index = seq(1, length(cook_vec)),
    Cook = cook_vec
  )
  corte <- c(4/nrow(dfcook))
  chart <- ggplot(dfcook, aes(x = Index, y = Cook)) +
           geom_point() +
           geom_segment(aes(x = Index, xend = Index, y = 0, yend = Cook)) +
           geom_hline(yintercept = corte, color = "red") +
           ggtitle("Cook's Distance")
  dfcook1 <- dfcook %>% filter(Cook > corte)
  out <- dfcook1 %>% mutate(y = y_values[dfcook1$Index]) %>% arrange(-Cook)
  return(list(threshold = corte, chartCook = chart, table = out))
}
dcook_result <- dcookAnalysis(modelf, dados_m[, "Potential"])
dcook_result
## $threshold
## [1] 0.008
##
## $chartCook
```




```
##
## $table
##                                        Index        Cook  y
## Manuel Neuer_167495                      146 0.136382310 90
## Carlos Tévez_143001                      195 0.062296067 78
## Theo Hernández_232656                    180 0.049873440 89
## Masahiro Okamoto_266276                   76 0.047926648 62
## Yuber Mosquera_253353                    377 0.043429119 66
## Andreas Hugo Hoelgebaum Pereira_208450    58 0.040261109 79
## Lucas Margueron_257142                   350 0.022833819 65
## Diego Chará_203067                       105 0.020885387 75
## Scott Carson_157804                      323 0.019298116 66
## Cole Palmer_257534                       196 0.017638882 86
## Diego Godín_182493                       435 0.017111928 78
## Enrique Herrero García_263104             51 0.014871429 80
## Lucas Gómez_253920                       475 0.014655177 67
## Dionicio Pérez_254954                    387 0.014504751 65
## Isaac Modi_265817                        143 0.014483897 73
## Tommy Jackson_264636                      77 0.013850023 61
## Siebe Wylin_265391                       303 0.013532013 82
## Adil Aouchiche_253102                    327 0.012934630 80
## Graziano Pellè_164376                    127 0.012397313 74
## Imam Jagne_259290                        286 0.011646263 75
## Young Joon Lee_261490                    103 0.011547203 77
## Leandro Soria_265512                     198 0.011351862 80
## Christian Gallardo_265495                441 0.010758580 78
```

```
## Fernando Reges Mouta_184134                62 0.009817186 84
## Tyler Goodrham_263861                       6 0.009442132 74
## Xiangshuo Zhang_261907                    293 0.009049760 55
## Ardon Jashari_257186                      235 0.008755423 74
## Hannibal Mejbri_258171                    228 0.008449470 84
## Luke Mbete_261621                          66 0.008298735 82
## Huapeng Wang_257689                       410 0.008156809 54
## Bendik Brevik_265692                      349 0.008043906 72
```

Temos, tambem, cerca de 31 valores influentes segundo a distância de Cook.

```
obser_r <- unique(
  c(rownames(leverage_result),
    rownames(dffit_result[["table"]]),
    rownames(dcook_result)[["table"]])
)
print(length(obser_r))
```

```
## [1] 50
```

Modelos sem dados influentes

```
dados_sem <- dados_m[!(row.names(dados_m) %in% obser_r),]
dim(dados_sem)
```

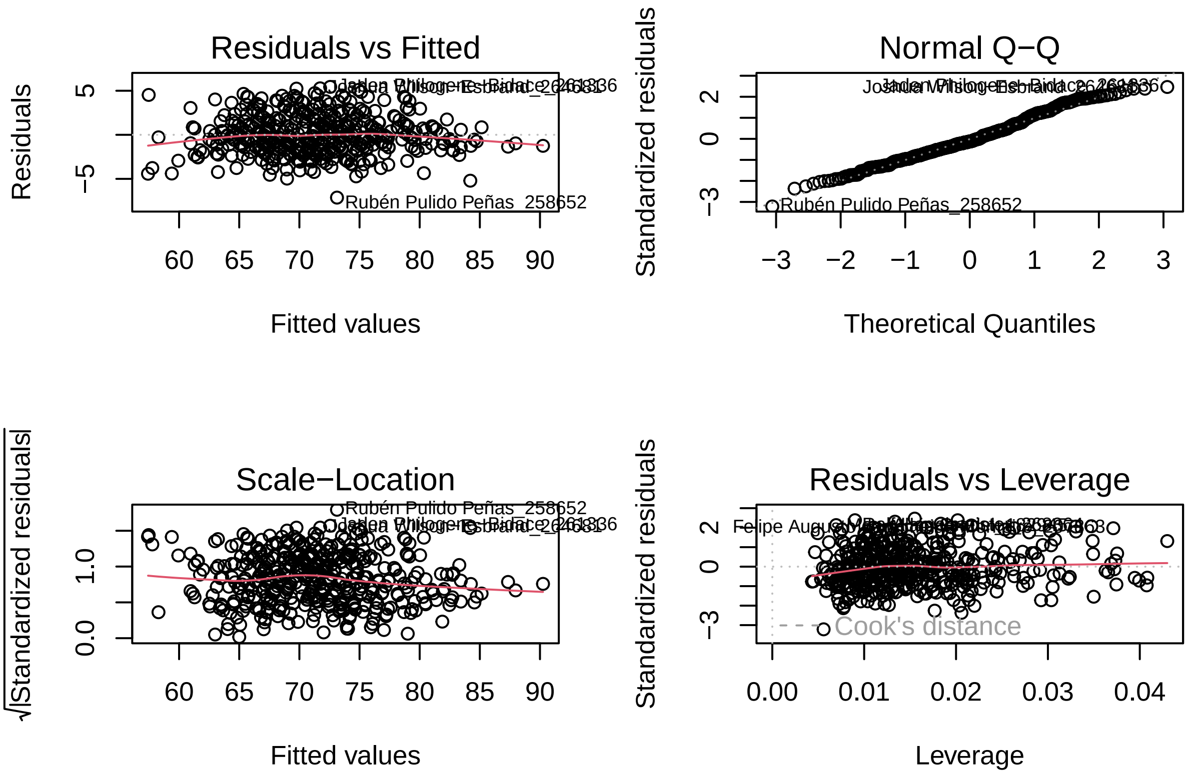
```
## [1] 450  40
```

```
model_sem <- lm(Potential ~ Age + Overall + `Skill move` + Stamina + Strength +
                            body_type,
                data = dados_sem)
summary(model_sem)
```

```
##
## Call:
## lm(formula = Potential ~ Age + Overall + `Skill move` + Stamina +
##     Strength + body_type, data = dados_sem)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1277 -1.5056 -0.2779  1.4523  5.4377
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     38.404036   0.932655  41.177  < 2e-16 ***
## Age             -1.094396   0.033918 -32.266  < 2e-16 ***
## Overall          0.933468   0.020912  44.638  < 2e-16 ***
## `Skill move`     0.210554   0.184983   1.138 0.255638
## Stamina         -0.030856   0.008865  -3.481 0.000549 ***
## Strength        -0.005403   0.010557  -0.512 0.609063
## body_typeNormal -0.471797   0.224032  -2.106 0.035772 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.223 on 443 degrees of freedom
## Multiple R-squared:  0.8568, Adjusted R-squared:  0.8549
## F-statistic: 441.8 on 6 and 443 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2,2))
plot(model_sem)
```




```r
testeNorm(model_sem$residuals)
```

```
##                                                Estatística    Valor - p
## Asymptotic one-sample Kolmogorov-Smirnov test   0.05759675 0.1010006442
## Lilliefors (Kolmogorov-Smirnov) normality test  0.05759675 0.0011215451
## Cramer-von Mises normality test                 0.28575106 0.0004835044
## Shapiro-Wilk normality test                     0.98858629 0.0013867156
## Shapiro-Francia normality test                  0.98916159 0.0028021104
## Anderson-Darling normality test                 1.72261497 0.0002030964
```

```r
print(bptest(modelf))
```

```
##
##  studentized Breusch-Pagan test
##
```

```
## data:  modelf
## BP = 20.654, df = 8, p-value = 0.008127
```

```
print(gqtest(modelf, fraction = 0.2, alternative = 'two.sided'))
```

```
##
##  Goldfeld-Quandt test
##
## data:  modelf
## GQ = 0.74731, df1 = 191, df2 = 191, p-value = 0.0448
## alternative hypothesis: variance changes from segment 1 to 2
```

A análise dos residuos acima é similar à analise feita antes de retirar os pontos. Isto é, a retirada daqueles pontos de uma vez só, pouco influenciou nos resíduos do novo modelo.

A única diferença é que agora, o teste KS não rejeita a normalidade dos resíduas com um nível de significância de 5%.

Pontos de alavanca

```
leveragePoints(model_sem)
```

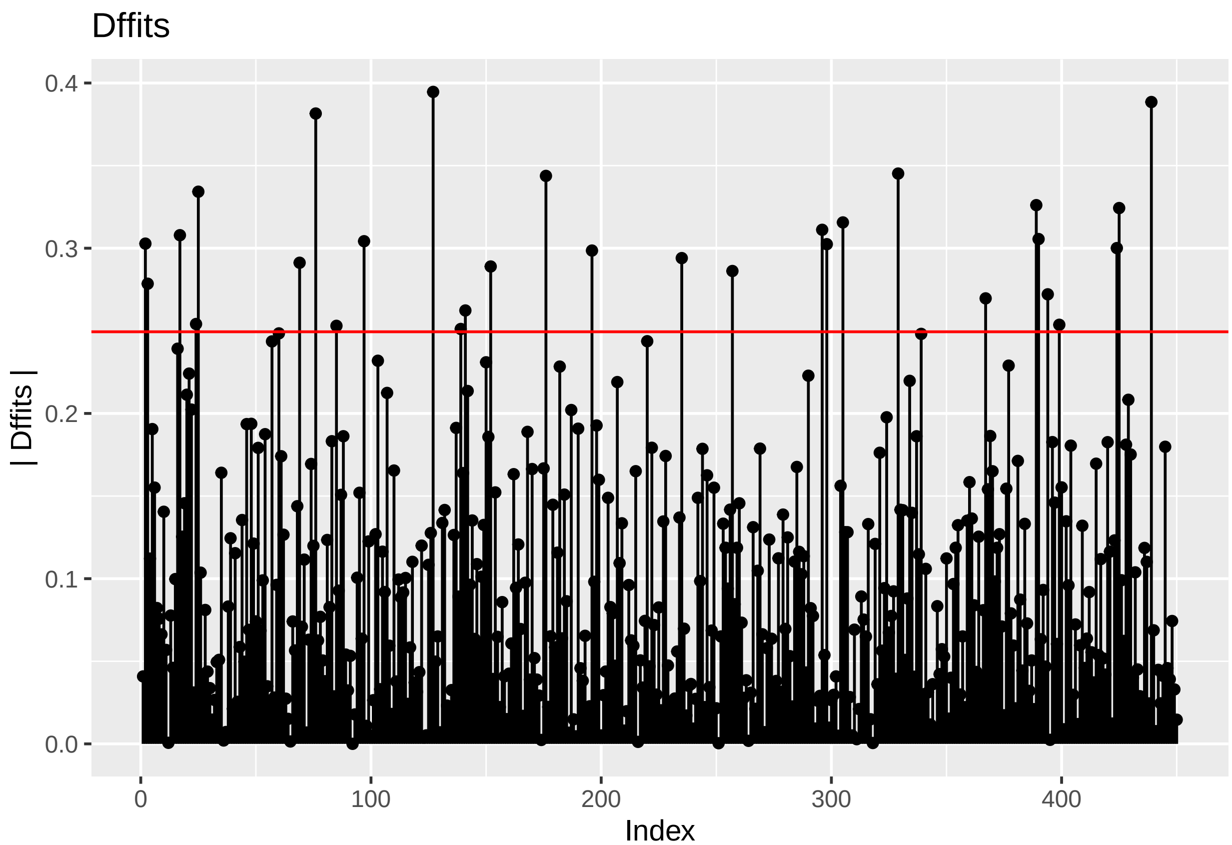
```
##                                                    hii
## Jordi Masip López_199575                    0.04299135
## Iñigo Orozco Andonegi_262496                0.04081942
## Ryonosuke Kabayama_256231                   0.04075239
## Nikola Vasilj_262956                        0.03993569
## Pablo Martín Páez Gavira_264240             0.03945132
## Carlo Pinsoglio_189342                      0.03753661
## Živko Kostadinović_257958                   0.03745424
## Moustapha Zeghba_251511                     0.03734205
## Paulo Gazzaniga_205186                      0.03723642
## Felipe Augusto de Almeida Monteiro_207863 0.03711872
## Maxime Dupé_204513                          0.03687812
## Yuta Matsumura_255439                       0.03655849
## Filip Đuričić_193881                        0.03628276
## Barrie McKay_209729                         0.03498571
## Genki Haraguchi_217648                      0.03491699
## Sandro R. G. Cordeiro_190782                0.03486866
## Ralf Hagelmeister_268364                    0.03473061
## Carlos Vela_169416                          0.03305678
## Max-Alain Gradel_182945                     0.03302499
## Angus Gunn_216325                           0.03238355
## Callum Hudson-Odoi_240740                   0.03221476
## Luis Andrés González_260090                 0.03127463
## Magnus Wolff Eikrem_153048                  0.03124522
```

Note que retirando os pontos de alavanca do modelo anterior surgem novos pontos de alavanca.

Vamos a visualização dos pontos influentes com a medida DFFIT.

```
dffitAnalysis(model_sem, dados_m[, "Potential"])
```

```
## $threshold
## [1] 0.2494438
##
## $chartDffits
```



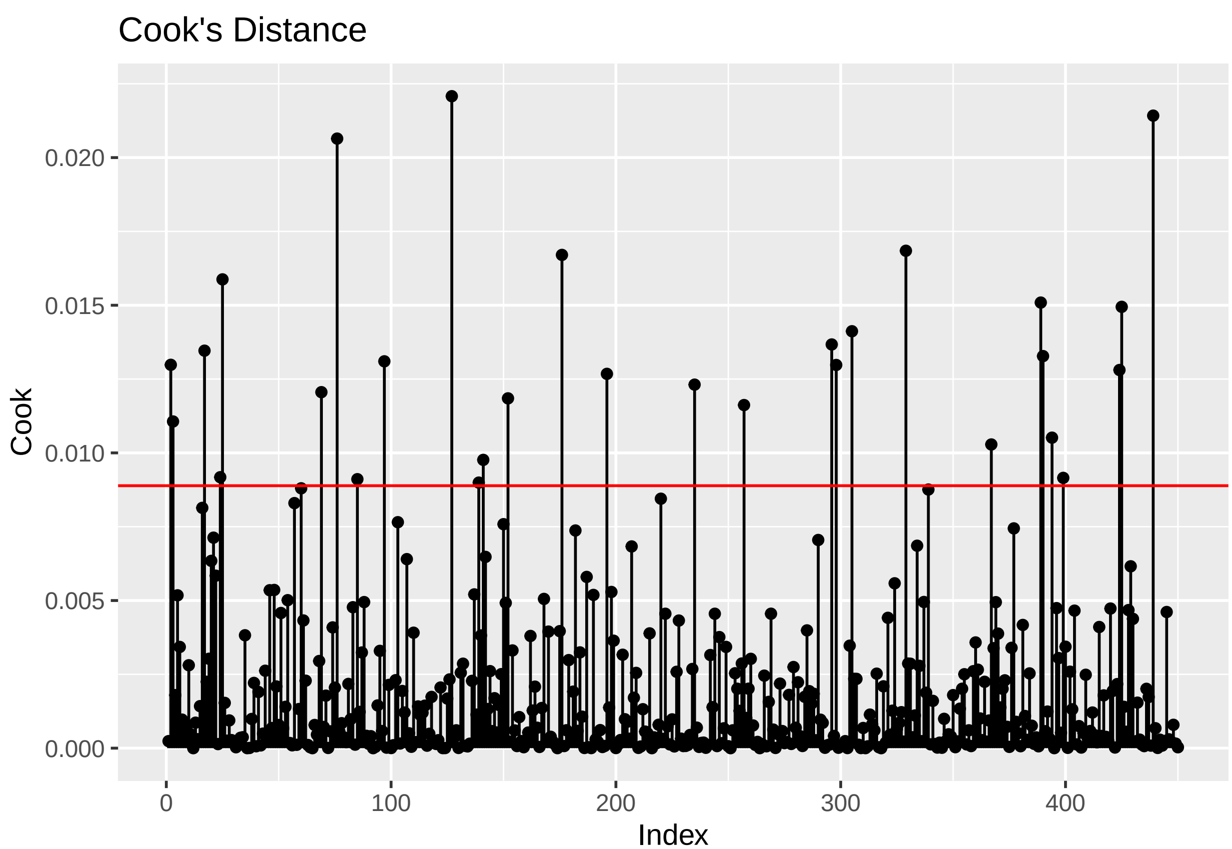

```
##
## $table
##                                              Index    Dffits  y
## Ralf Hagelmeister_268364                       127 0.3946058 74
## Felipe Augusto de Almeida Monteiro_207863      439 0.3884968 69
## Max-Alain Gradel_182945                         76 0.3815297 62
## Luis Henrique Tomaz de Lima_256632             329 0.3451908 74
## James McArthur_171972                          176 0.3437761 62
## Carlos Vela_169416                              25 0.3342126 71
## Juan Villar Vázquez_192490                     389 0.3260846 66
## Edgar Benítez_190523                           425 0.3243212 72
## Michał Pazdan_185010                           305 0.3156180 69
## Jaden Philogene-Bidace_261336                  296 0.3111516 67
## Matías Palacios_246107                          17 0.3078816 78
## Richard Salinas_254770                         390 0.3055443 70
## Emmanuel Ilesanmi_265605                        97 0.3042389 76
## Daniel Wass_172522                               2 0.3027850 80
## Zhen Ma_250978                                 298 0.3024428 68
## Noah Joel Sarenren Bazee_233472                424 0.3000807 69
## César Azpilicueta Tanco_184432                 196 0.2985846 86
```

```
## Barrie McKay_209729                         235 0.2940278 74
## Tae Suk Lee_261063                            69 0.2912205 67
## Dilyimit Tudi_261917                         152 0.2889481 72
## Martin Niøten Palumbo_257859                 257 0.2862095 75
## Jordi Masip López_199575                       3 0.2785196 74
## Jesús Soraire_251139                         394 0.2720990 67
## Williot Swedberg_263227                      367 0.2696451 70
## Jakub Antczak_267614                         141 0.2623180 62
## Andreas Samaris_208230                        24 0.2541340 76
## Jeremy Silva_255774                          399 0.2536788 69
## Danila Bokov_257996                           85 0.2530267 70
## Sandro R. G. Cordeiro_190782                 139 0.2511162 74
```

Note que retirando os pontos influentes segundo DFFIT do modelo anterior surgem novos pontos influentes.

Agora, vamos então observar pontos influentes utilizando a distância Dcook

```
dcookAnalysis(model_sem, dados_m[, "Potential"])
## $threshold
## [1] 0.008888889
##
## $chartCook
```




```
##
## $table
##                                              Index        Cook  y
```

```
## Ralf Hagelmeister_268364                    127 0.022078966 74
## Felipe Augusto de Almeida Monteiro_207863   439 0.021420439 69
## Max-Alain Gradel_182945                      76 0.020642980 62
## Luis Henrique Tomaz de Lima_256632          329 0.016844967 74
## James McArthur_171972                       176 0.016704518 62
## Carlos Vela_169416                           25 0.015875616 71
## Juan Villar Vázquez_192490                  389 0.015090230 66
## Edgar Benítez_190523                        425 0.014946234 72
## Michał Pazdan_185010                        305 0.014119209 69
## Jaden Philogene-Bidace_261336               296 0.013671880 67
## Matías Palacios_246107                       17 0.013461029 78
## Richard Salinas_254770                      390 0.013277397 70
## Emmanuel Ilesanmi_265605                     97 0.013100515 76
## Daniel Wass_172522                            2 0.012982435 80
## Zhen Ma_250978                              298 0.012977558 68
## Noah Joel Sarenren Bazee_233472             424 0.012806596 69
## César Azpilicueta Tanco_184432              196 0.012676051 86
## Barrie McKay_209729                         235 0.012311852 74
## Tae Suk Lee_261063                           69 0.012055696 67
## Dilyimit Tudi_261917                        152 0.011846652 72
## Martin Niøten Palumbo_257859                257 0.011620149 75
## Jordi Masip Lőpez_199575                      3 0.011063726 74
## Jesús Soraire_251139                        394 0.010515713 67
## Williot Swedberg_263227                     367 0.010285374 70
## Jakub Antczak_267614                        141 0.009762814 62
## Andreas Samaris_208230                       24 0.009175859 76
## Jeremy Silva_255774                         399 0.009153132 69
## Danila Bokov_257996                          85 0.009109511 70
## Sandro R. G. Cordeiro_190782                139 0.008993346 74
```

Note que retirando os pontos influentes segundo DCOOK do modelo anterior surgem novos pontos influentes.

Em razão disso, optamos pelo modelo que considera as seguintes covariáveis: "Age", "Overall", "Skill move", "Stamina", "Strength" e "body_type". E também contém todas as 500 observações.

Sendo assim o modelo ajustado foi:

$$\widehat{Potential} = 38.342 - 1.018 * Age + 0.917 * Overall + 0.250 * `Skillmove` + -0.036 * Stamina - 0.008 * Strength - 0.600 * body_typeNor$$

Interpretação da varial dummie.

Se o body type for "Lean" temos:

$$\widehat{Potential} = 38.342 - 1.018 * Age + 0.917 * Overall + 0.250 * `Skillmove` + -0.036 * Stamina - 0.008 * Strength$$

Se o body type for "Normal" temos:

$$\widehat{Potential} = 37.742 - 1.018 * Age + 0.917 * Overall + 0.250 * `Skillmove` + -0.036 * Stamina - 0.008 * Strength$$

Se o body type for "Stocky" temos:

$$\widehat{Potential} = 38.601 - 1.018 * Age + 0.917 * Overall + 0.250 * `Skillmove` + -0.036 * Stamina - 0.008 * Strength$$

Se o body type for "Unique" temos:

$$\widehat{Potential} = 40.938 - 1.018 * Age + 0.917 * Overall + 0.250 * `Skillmove` + -0.036 * Stamina - 0.008 * Strength$$

Ou seja:

- body type "Normal": provoca uma diminuição no intercepto;
- body type "Stocky": provoca um aumento no intercepto;
- body type "Unique": provoca um aumento no intercepto.