

Análise da Base de Dados Adult

Sobre a base de dados

O conjunto de dados utilizado é chamado “Adult” e foi derivado do banco de dados do censo dos EUA. Foi preparado por Barry Becker a partir do censo de 1994. O propósito desse conjunto de dados é prever se uma pessoa ganha mais ou menos de US\$ 50.000 por ano.

Link do conjunto de dados: <https://archive.ics.uci.edu/ml/datasets/Adult>

Carregando os Dados do arquivo

```
database <- read.csv(file = "Adult.CSV")

colnames(database) <- c("age", "workclass", "fnlwgt", "education", "education_num", "marital_status", "occupation", "relationship", "race", "sex", "capital_gain", "capital_loss", "hours_per_week", "native_country", "income")

#Printar a base de dados
database
```

Table 1: Adult Database - Part 1

age	workclass	fnlwgt	education	education_num	marital_status	occupation
39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical
50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial
38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners
53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners
28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty

Table 2: Adult Database - Part 2

relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	income
Not-in-family	White	Male	2174	0	40	United-States	<=50K
Husband	White	Male	0	0	13	United-States	<=50K
Not-in-family	White	Male	0	0	40	United-States	<=50K
Husband	Black	Male	0	0	40	United-States	<=50K
Wife	Black	Female	0	0	40	Cuba	<=50K

1. Identificação do atributo alvo (saída)

A base de dados possui 32561 exemplos e 15 atributos sendo seu atributo alvo “income”

```
target_attribute <- "income"
```

2. Identificação dos tipos de dados dos atributos de entrada (quantitativo, qualitativo)

Identificar os tipos de dados em uma base de dados é essencial para entender a natureza das informações presentes. Na base de dados analisada, os dados podem ser classificados nas seguintes categorias:

- Categóricos: Representam características com valores qualitativos.
- Numéricos: Representam valores numéricos.

```
#Para classificação dos atributos como int ou char
str(database)
```

```
## 'data.frame':    32562 obs. of  15 variables:
##   $ age          : int  39 50 38 53 28 37 49 52 31 42 ...
##   $ workclass    : chr  "State-gov" "Self-emp-not-inc" "Private" "Private" ...
##   $ fnlwgt       : int  77516 83311 215646 234721 338409 284582 160187 209642 45781 159449 ...
##   $ education     : chr  "Bachelors" "Bachelors" "HS-grad" "11th" ...
##   $ education_num : int  13 13 9 7 13 14 5 9 14 13 ...
##   $ marital_status: chr  "Never-married" "Married-civ-spouse" "Divorced" "Married-civ-spouse" ...
##   $ occupation    : chr  "Adm-clerical" "Exec-managerial" "Handlers-cleaners" "Handlers-cleaners" ...
##   $ relationship   : chr  "Not-in-family" "Husband" "Not-in-family" "Husband" ...
##   $ race          : chr  "White" "White" "White" "Black" ...
##   $ sex           : chr  "Male" "Male" "Male" "Male" ...
##   $ capital_gain  : int  2174 0 0 0 0 0 0 14084 5178 ...
##   $ capital_loss  : int  0 0 0 0 0 0 0 0 0 ...
##   $ hours_per_week: int  40 13 40 40 40 40 16 45 50 40 ...
##   $ native_country: chr  "United-States" "United-States" "United-States" "United-States" ...
##   $ income         : chr  "<=50K" "<=50K" "<=50K" "<=50K" ...
```

```
#classificação manual
data_types <- c(
  age = "quantitativo",
  workclass = "qualitativo",
  fnlwgt = "quantitativo",
  education = "qualitativo",
  education_num = "quantitativo",
  marital_status = "qualitativo",
  occupation = "qualitativo",
  relationship = "qualitativo",
  race = "qualitativo",
  sex = "qualitativo",
  capital_gain = "quantitativo",
  capital_loss = "quantitativo",
  hours_per_week = "quantitativo",
  native_country = "qualitativo",
  income = "qualitativo"
)
data_types
```

Table 3: Tipos de dados dos atributos de entrada

Atributo	Classificação
age	quantitativo
workclass	qualitativo
fnlwgt	quantitativo
education	qualitativo
education_num	quantitativo
marital_status	qualitativo
occupation	qualitativo
relationship	qualitativo
race	qualitativo
sex	qualitativo
capital_gain	quantitativo
capital_loss	quantitativo
hours_per_week	quantitativo
native_country	qualitativo
income	qualitativo

3. Identificação da escala de dados dos atributos de entrada (nominal, ordinal, intervalar, racional)

A identificação da escala de dados é relevante para entender a natureza e a magnitude dos valores presentes em uma base de dados. A escala dos dados pode ser dividida em quatro categorias principais:

- Nominal: Categorizados em diferentes grupos ou categorias, sem qualquer ordem ou hierarquia específica.
- Ordinal: Possuem uma ordem ou hierarquia específica, mas a diferença entre os valores não é necessariamente uniforme.
- Intervalar: Expressos em valores numéricos com uma diferença uniforme entre eles.
- Racional: Representam um valor numérico absoluto.

```
data_scales <- c(
  age = "Racional",
  workclass = "Nominal",
  fnlwgt = "Racional",
  education = "Ordinal",
  education_num = "Ordinal",
  marital_status = "Nominal",
  occupation = "Nominal",
  relationship = "Nominal",
  race = "Nominal",
  sex = "Nominal",
  capital_gain = "Racional",
  capital_loss = "Racional",
  hours_per_week = "Racional",
  native_country = "Nominal",
  income = "Nominal"
)
data_scales
```

Table 4: Identificação da escala de dados

Atributo	Escala
age	Racional
workclass	Nominal
fnlwgt	Racional
education	Ordinal
education_num	Ordinal
marital_status	Nominal
occupation	Nominal
relationship	Nominal
race	Nominal
sex	Nominal
capital_gain	Racional
capital_loss	Racional
hours_per_week	Racional
native_country	Nominal
income	Nominal

4. Exploração dos dados através de medidas de localidade

As medidas de localidade, fornecem informações sobre o valor central dos dados, no qual ajudam a entender onde a maioria dos dados está concentrado. Essas medidas ajudam a entender onde a maioria dos dados está concentrada. Incluem:

- Média
- Moda
- Mediana

```
numeric_columns1 <- sapply(database, is.numeric)
summary(database[, numeric_columns1])
```

```
##      age          fnlwgt      education_num      capital_gain
##  Min.   :17.00    Min.   : 12285    Min.   : 1.00    Min.   : 0
##  1st Qu.:28.00   1st Qu.: 117827   1st Qu.: 9.00    1st Qu.: 0
##  Median :37.00   Median : 178356   Median :10.00    Median : 0
##  Mean   :38.58   Mean   : 189778   Mean   :10.08    Mean   : 1078
##  3rd Qu.:48.00   3rd Qu.: 237051   3rd Qu.:12.00    3rd Qu.: 0
##  Max.   :90.00   Max.   :1484705   Max.   :16.00    Max.   :99999
##  NA's    :1       NA's    :1       NA's    :1       NA's    :1
##      capital_loss      hours_per_week
##  Min.   : 0.0     Min.   : 1.00
##  1st Qu.: 0.0     1st Qu.:40.00
##  Median : 0.0     Median :40.00
##  Mean   : 87.3    Mean   :40.44
##  3rd Qu.: 0.0     3rd Qu.:45.00
##  Max.   :4356.0   Max.   :99.00
##  NA's    :1       NA's    :1
```

```

Modes <- function(x) {
  ux <- unique(x)
  tab <- tabulate(match(x, ux))
  ux[tab == max(tab)]
}

for (i in 1:ncol(database)) {
  mod_val <- Modes(database[,i])
  print(mod_val)
}

```

```

## [1] 36
## [1] " Private"
## [1] 123011 164190 203488
## [1] " HS-grad"
## [1] 9
## [1] " Married-civ-spouse"
## [1] " Prof-specialty"
## [1] " Husband"
## [1] " White"
## [1] " Male"
## [1] 0
## [1] 0
## [1] 40
## [1] " United-States"
## [1] " <=50K"

```

5. Exploração dos dados através de medidas de espalhamento

As medidas de espalhamento são utilizadas para avaliar a dispersão ou variabilidade dos dados em um conjunto de observações. Elas fornecem informações sobre o quanto distantes os valores estão uns dos outros e da medida central dos dados. Incluem

- Variância
- Desvio Padrão
- Amplitude

```

spread_measures <- function(x) {
  c(variance = var(x, na.rm = TRUE), sd = sd(x, na.rm = TRUE), range = range(x, na.rm = TRUE))
}

# Identifica quais colunas são numéricas.
numeric_columns2 <- sapply(database, is.numeric)
# Aplica a função spread_measures a cada coluna numérica para calcular a variância, desvio padrão e amplitude
sapply(database[, numeric_columns2], spread_measures)

```

	age	fnlwgt	education_num	capital_gain	capital_loss
## variance	186.06140	11140797792	6.61889	54542539.178	162376.9378
## sd	13.64043	105550	2.57272	7385.292	402.9602
## range1	17.00000	12285	1.00000	0.000	0.00000
## range2	90.00000	1484705	16.00000	99999.000	4356.0000

```

##           hours_per_week
## variance      152.45900
## sd            12.34743
## range1        1.00000
## range2        99.00000

```

6. Exploração dos dados através de medidas de distribuição

As medidas de distribuição são utilizadas para analisar a forma e a natureza da distribuição dos dados em um conjunto de observações. Essas medidas fornecem informações sobre a simetria, o achatamento e a forma geral da distribuição dos dados. Incluem:

- Assimetria
- Curtose

```

# Funções para calcular assimetria e curtose
calc_skewness <- function(x) {
  n <- length(x)
  mean_x <- mean(x, na.rm = TRUE)
  sd_x <- sd(x, na.rm = TRUE)
  skewness <- sum(((x - mean_x) / sd_x) ^ 3, na.rm = TRUE) / n
  return(skewness)
}

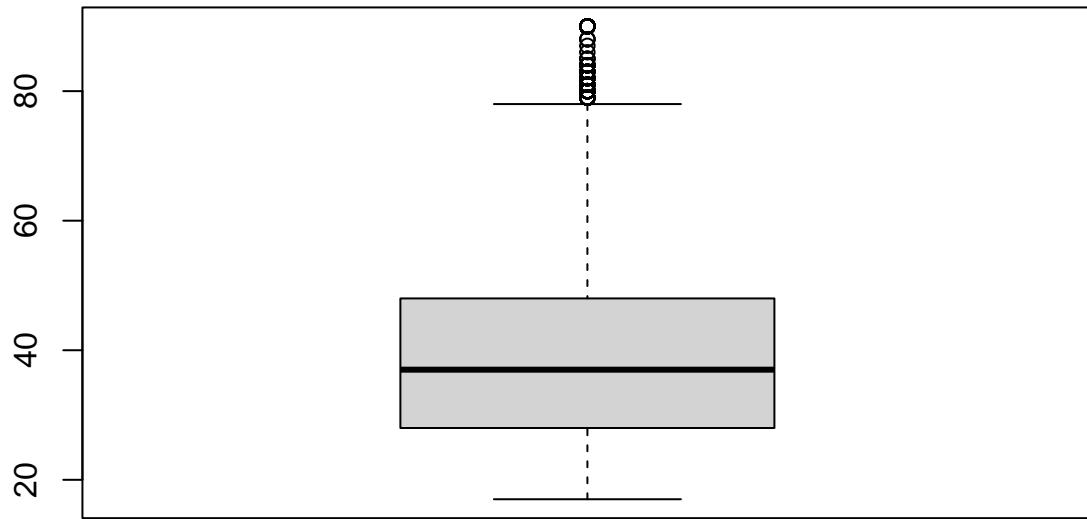
calc_kurtosis <- function(x) {
  n <- length(x)
  mean_x <- mean(x, na.rm = TRUE)
  sd_x <- sd(x, na.rm = TRUE)
  kurtosis <- sum(((x - mean_x) / sd_x) ^ 4, na.rm = TRUE) / n - 3
  return(kurtosis)
}

# Supondo que 'database' é o seu conjunto de dados
numeric_columns3 <- sapply(database, is.numeric)

# Plotando boxplots para cada variável numérica
boxplot(database$age, main = "Age Boxplot")

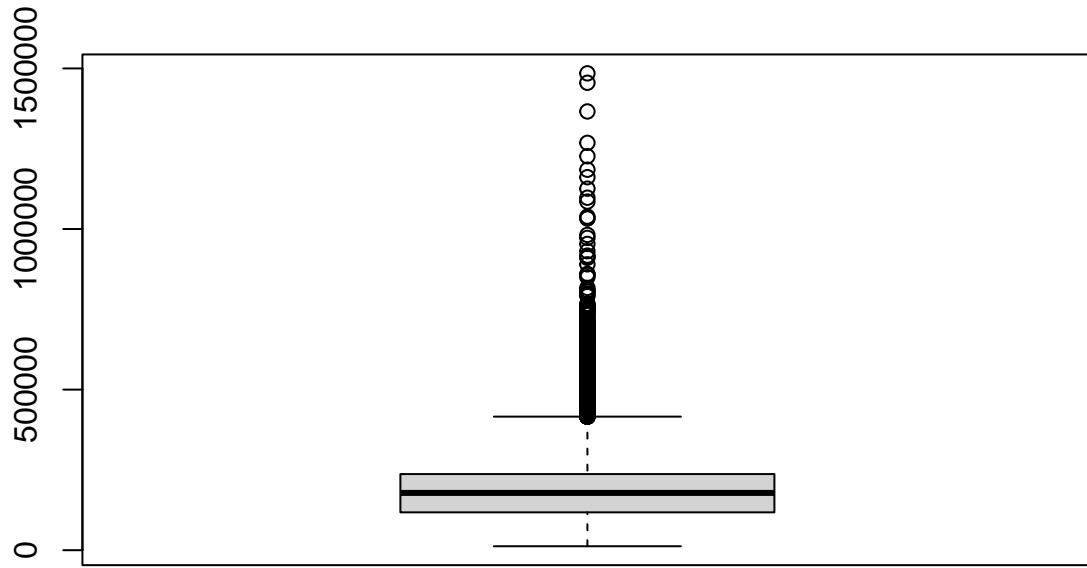
```

Age Boxplot



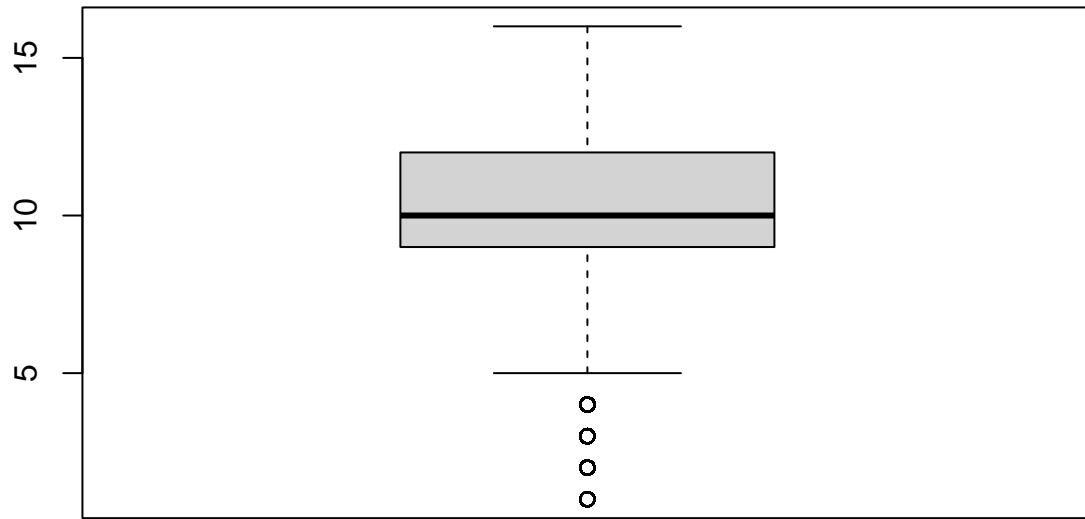
```
boxplot(database$fnlwgt, main = "Fnlwgt Boxplot")
```

Fnlwgt Boxplot



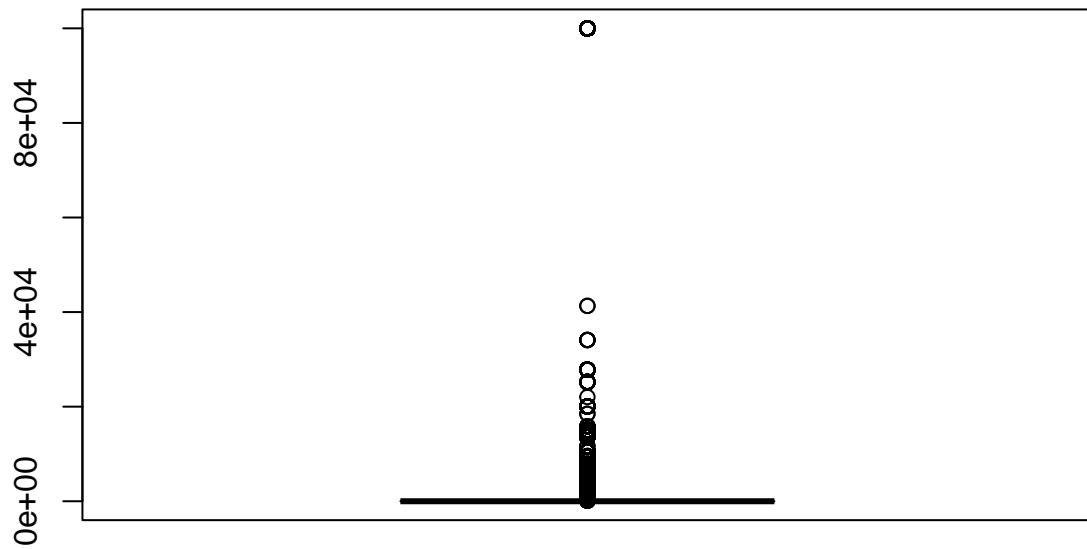
```
boxplot(database$education_num, main = "Education_num Boxplot")
```

Education_num Boxplot



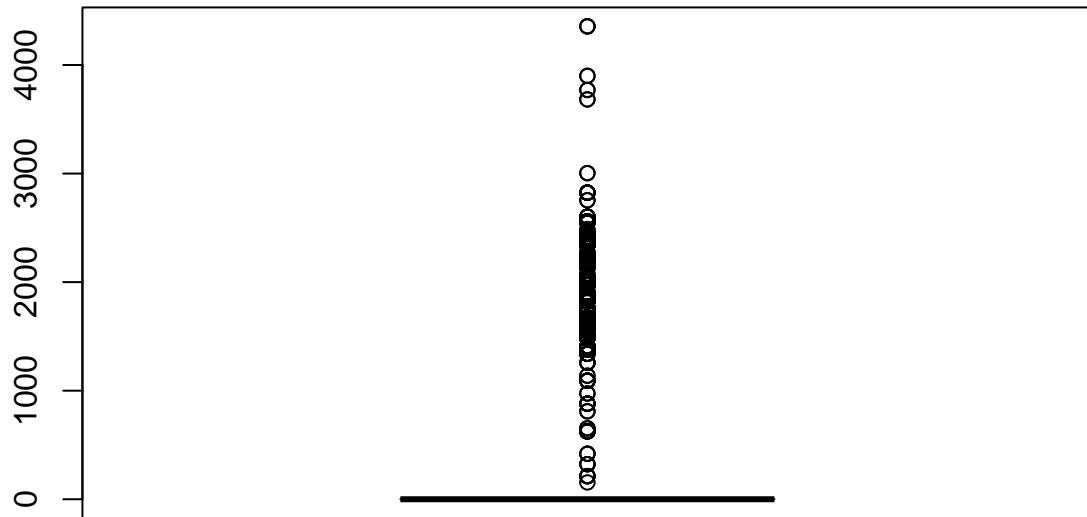
```
boxplot(database$capital_gain, main = "capital_gain Boxplot")
```

capital_gain Boxplot



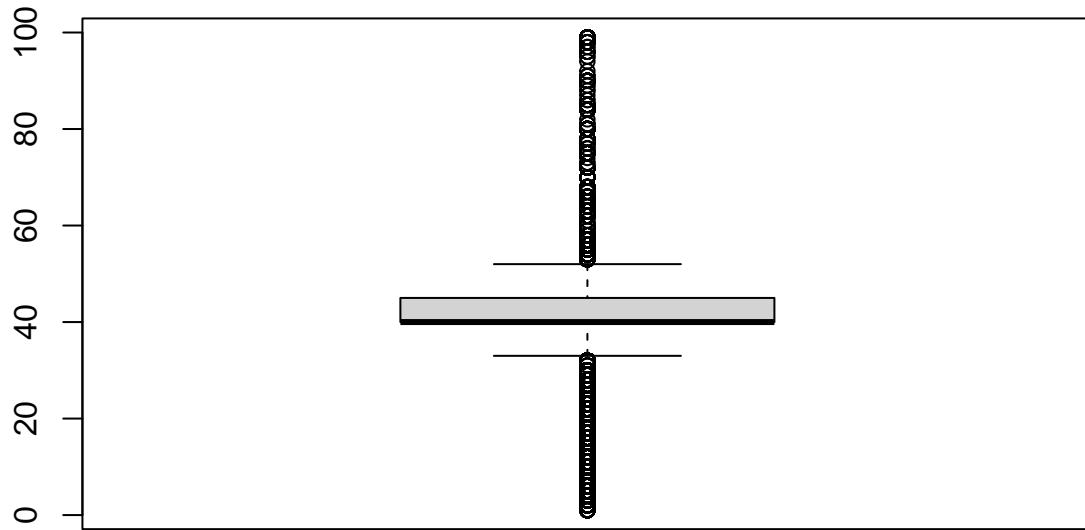
```
boxplot(database$capital_loss, main = "capital_loss Boxplot")
```

capital_loss Boxplot



```
boxplot(database$hours_per_week, main = "hours_per_week Boxplot")
```

hours_per_week Boxplot

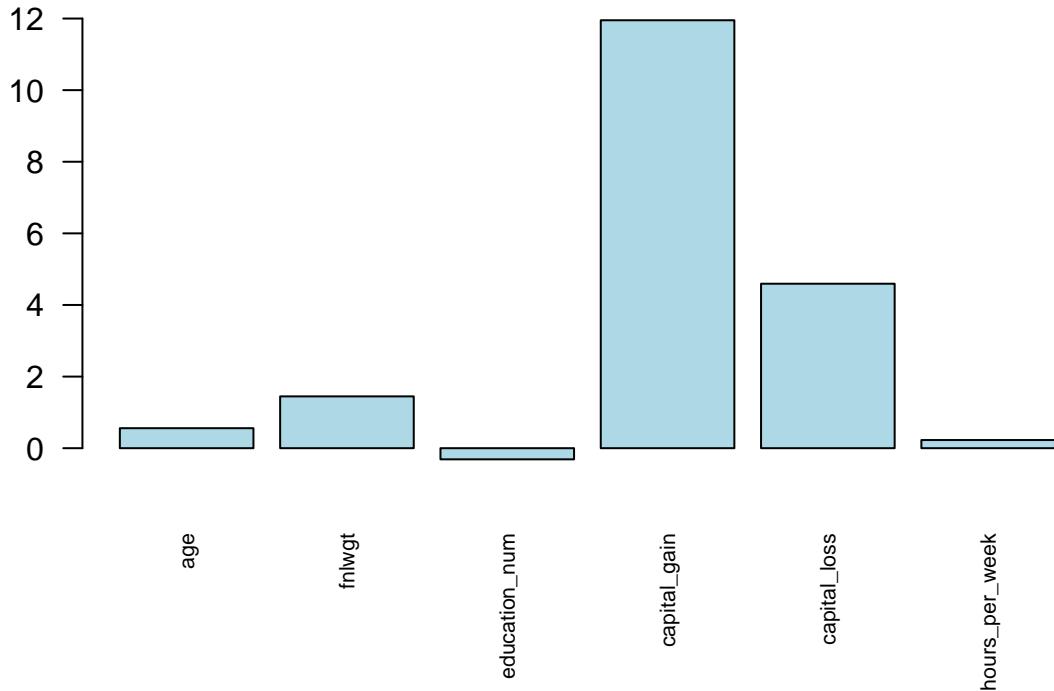


```
# Calculando assimetria e curtose
skewness_values <- sapply(database[, numeric_columns3], calc_skewness)
kurtosis_values <- sapply(database[, numeric_columns3], calc_kurtosis)

# Criando um dataframe para plotagem
metrics_df <- data.frame(
  Variable = names(skewness_values),
  Skewness = skewness_values,
  Kurtosis = kurtosis_values
)

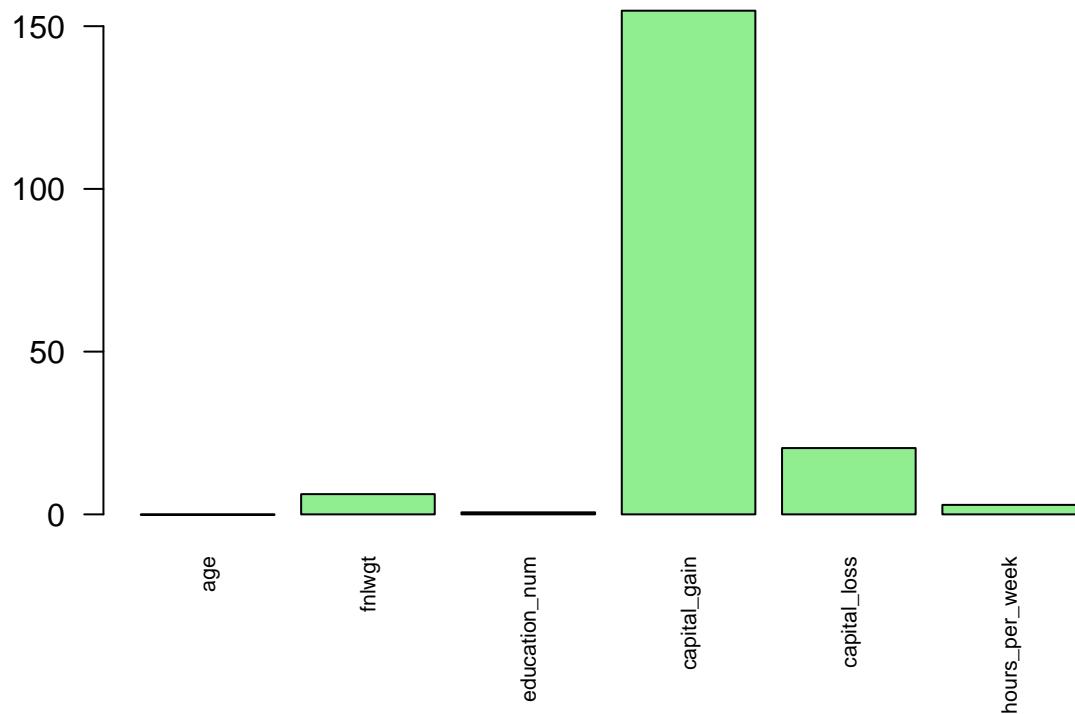
# Gráfico de assimetria
barplot(metrics_df$Skewness, names.arg = metrics_df$Variable,
        main = "Assimetria das Variáveis Numéricas",
        col = "lightblue", ylim = c(min(metrics_df$Skewness) - 1, max(metrics_df$Skewness) + 1),
        cex.names = 0.7, las = 2)
```

Assimetria das Variáveis Numéricas



```
# Gráfico de curtose
barplot(metrics_df$Kurtosis, names.arg = metrics_df$Variable,
        main = "Curtose das Variáveis Numéricas",
        col = "lightgreen", ylim = c(min(metrics_df$Kurtosis) - 1, max(metrics_df$Kurtosis) + 1),
        cex.names = 0.7, las = 2)
```

Curtose das Variáveis Numéricas



7. Identificação e separação do conjunto de teste

A base de dados original possui 32561 exemplos e 15 atributos

A base de dados teste possui 16281 exemplos e 15 atributos

O que implica em uma base de dados com metade dos exemplos.

```
train_data <- read.csv(file = "Adult.CSV")
test_data <- read.csv(file = "adultTest.csv")

colnames(train_data) <- c("age", "workclass", "fnlwgt", "education", "education_num", "marital_status",
                         "occupation", "relationship", "race", "sex", "capital_gain", "capital_loss", "hours_per_week", "income")

colnames(test_data) <- c("age", "workclass", "fnlwgt", "education", "education_num", "marital_status",
                        "occupation", "relationship", "race", "sex", "capital_gain", "capital_loss", "hours_per_week", "income")

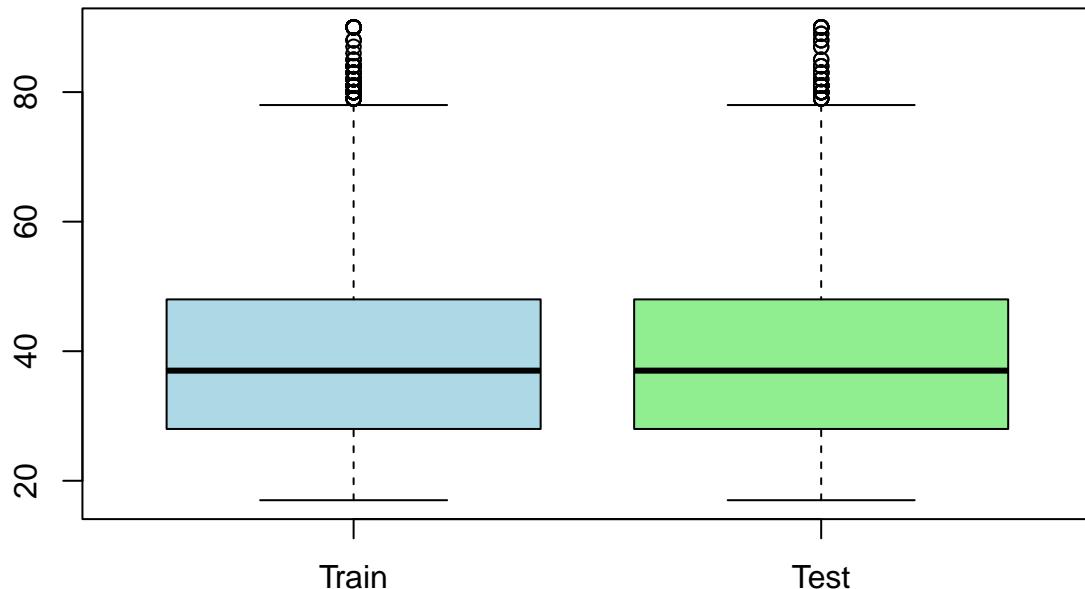
test_data$income <- gsub(">50K.", " >50K", test_data$income)
test_data$income <- gsub("<=50K.", " <=50K", test_data$income)

# Identificando colunas numéricas
numeric_columns <- sapply(train_data, is.numeric)

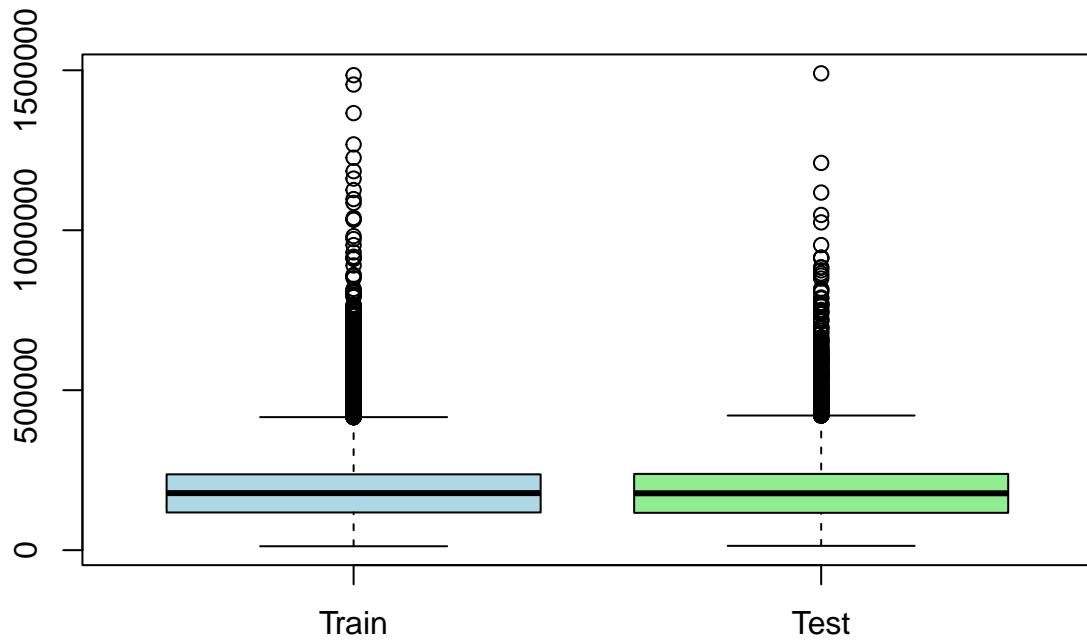
par(mfrow=c(1, 1))
for (col in names(train_data)[numeric_columns]) {
  boxplot(train_data[[col]], test_data[[col]],
          names = c("Train", "Test"),
          main = paste("Boxplot Comparativo para", col),
```

```
    col = c("lightblue", "lightgreen"))
}
```

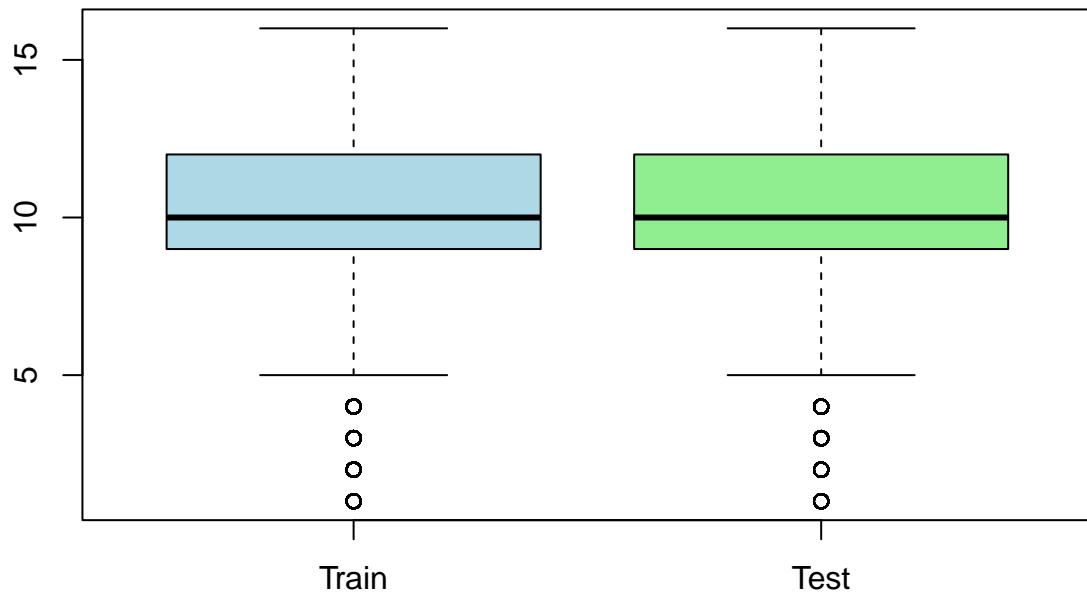
Boxplot Comparativo para age



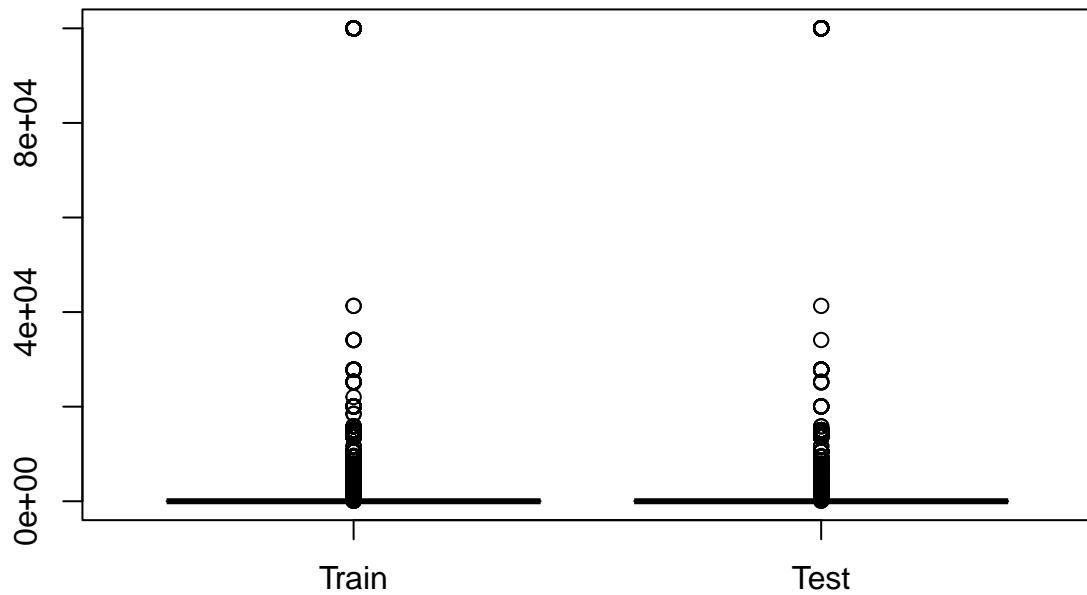
Boxplot Comparativo para fnlwgt



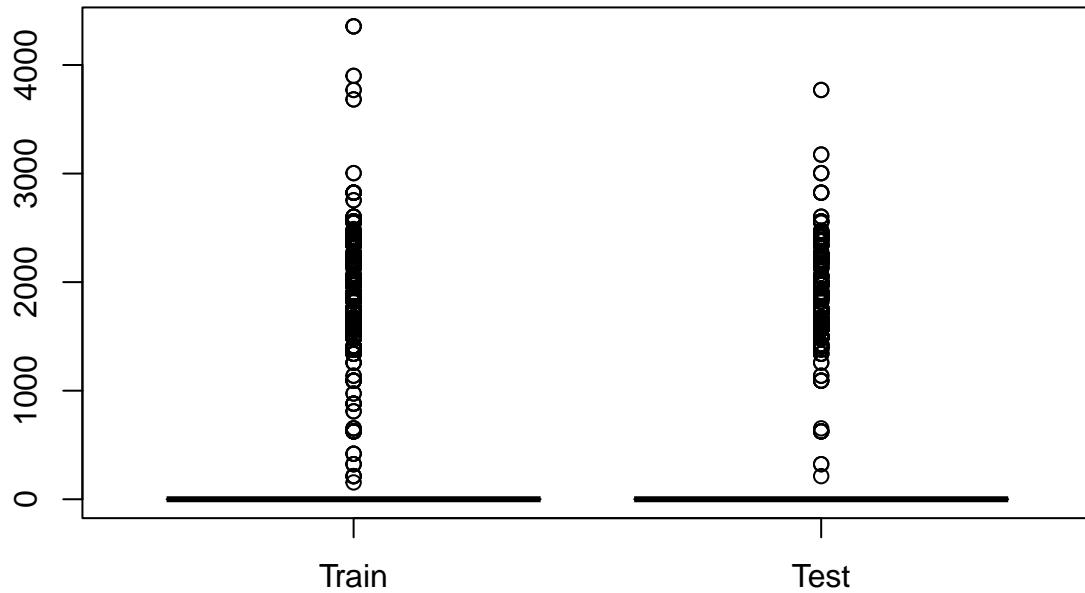
Boxplot Comparativo para education_num



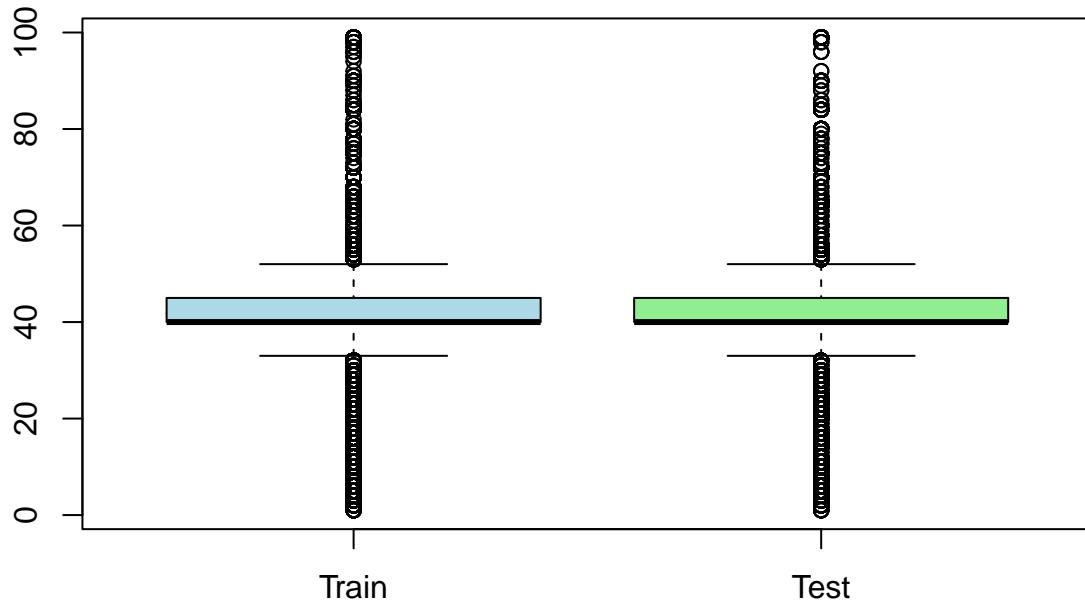
Boxplot Comparativo para capital_gain



Boxplot Comparativo para capital_loss



Boxplot Comparativo para hours_per_week



```
jitter_horizontal <- function(x) {
  jitter(x, factor = 0.1)
}

# Criando um loop para gerar gráficos de dispersão comparativos
for (col in names(train_data)[numeric_columns]) {
  plot(jitter_horizontal(rep(1, length(train_data[[col]]))), train_data[[col]],
       xlim = c(0.5, 2.5), xaxt = "n",
       main = paste("Gráfico de Dispersão Comparativo para", col),
       xlab = "Conjunto de Dados", ylab = col, col = "blue")

  points(jitter_horizontal(rep(2, length(test_data[[col]]))), test_data[[col]], col = "green")

  axis(1, at = 1:2, labels = c("Train", "Test"))
  legend("topright", legend = c("Train", "Test"), col = c("blue", "green"), pch = 1)
}
```

Gráfico de Dispersão Comparativo para age

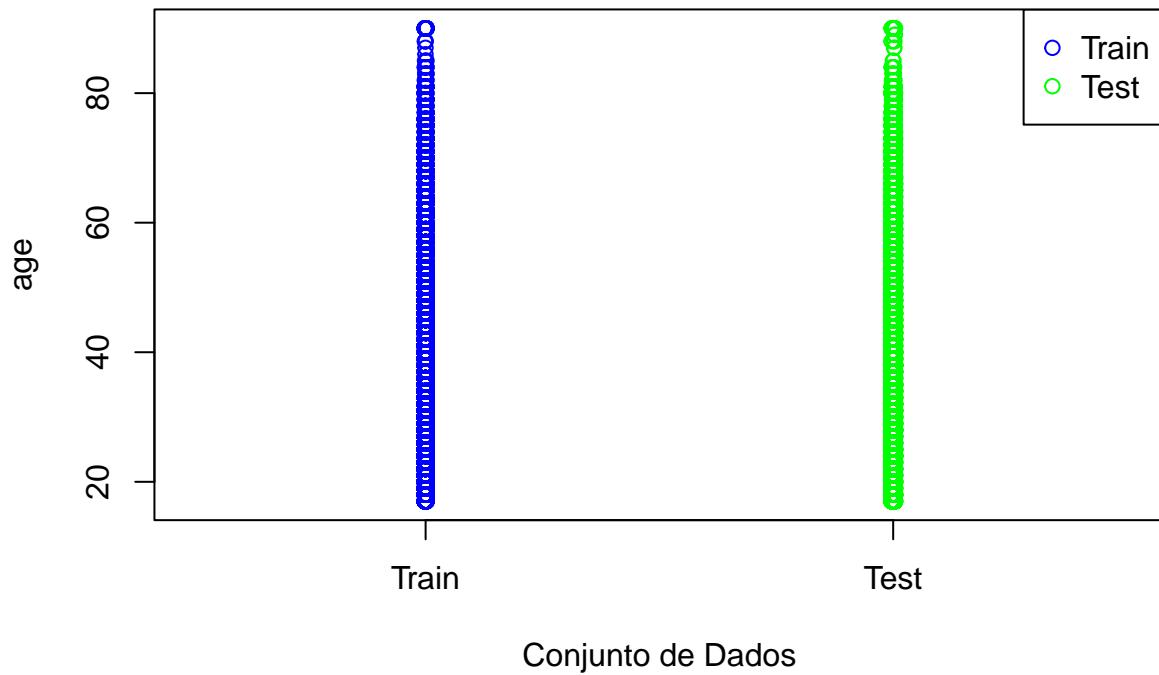


Gráfico de Dispersão Comparativo para fnlwgt

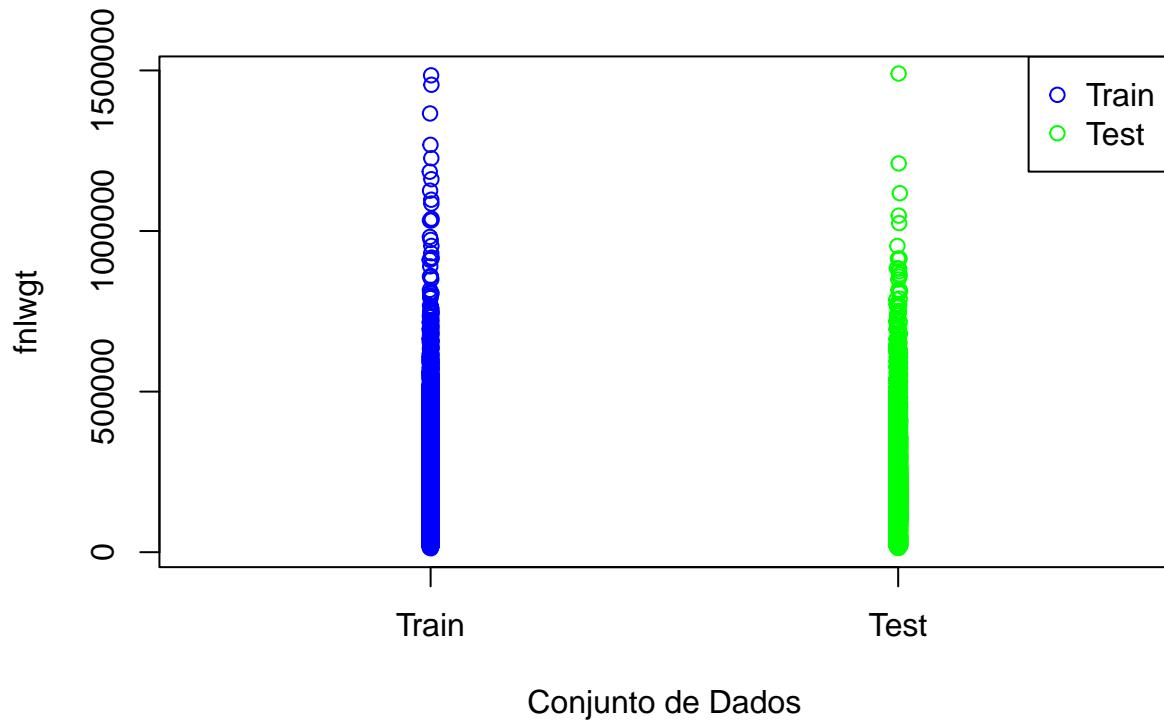


Gráfico de Dispersão Comparativo para education_num

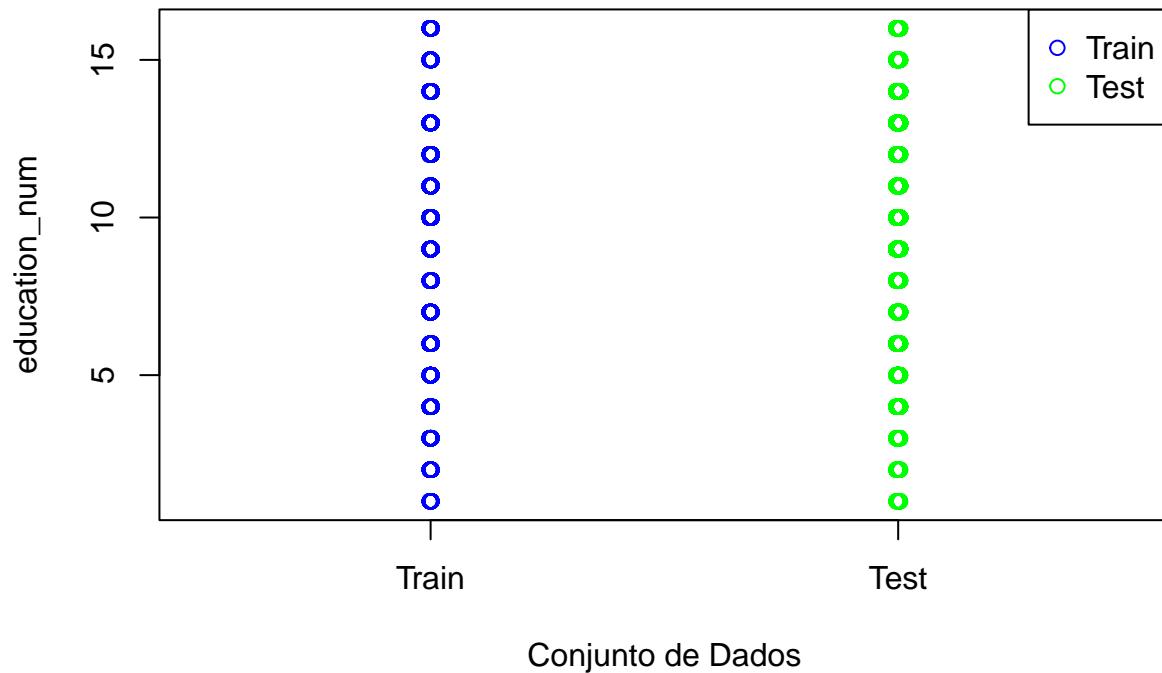


Gráfico de Dispersão Comparativo para capital_gain

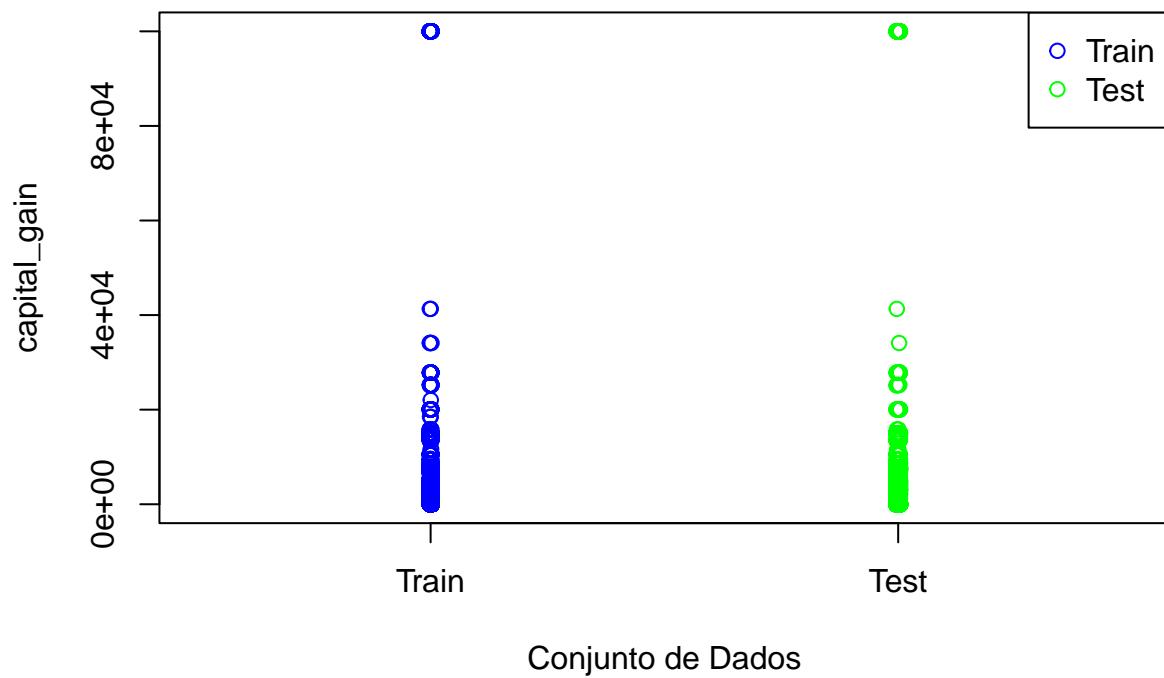


Gráfico de Dispersão Comparativo para capital_loss

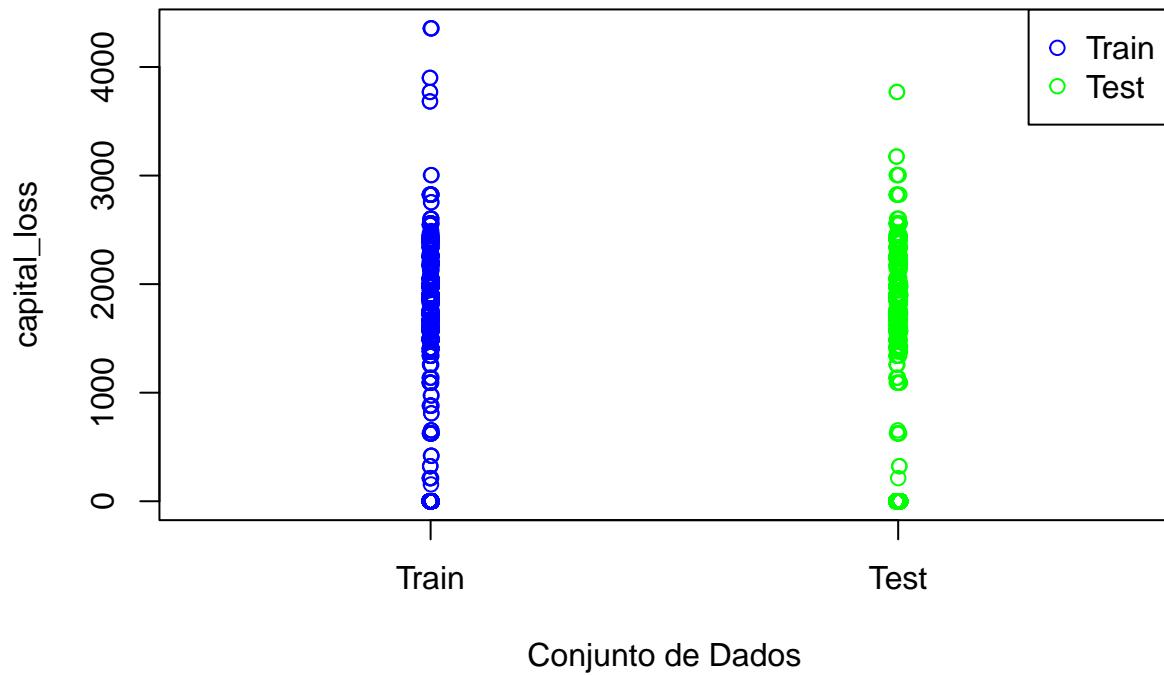
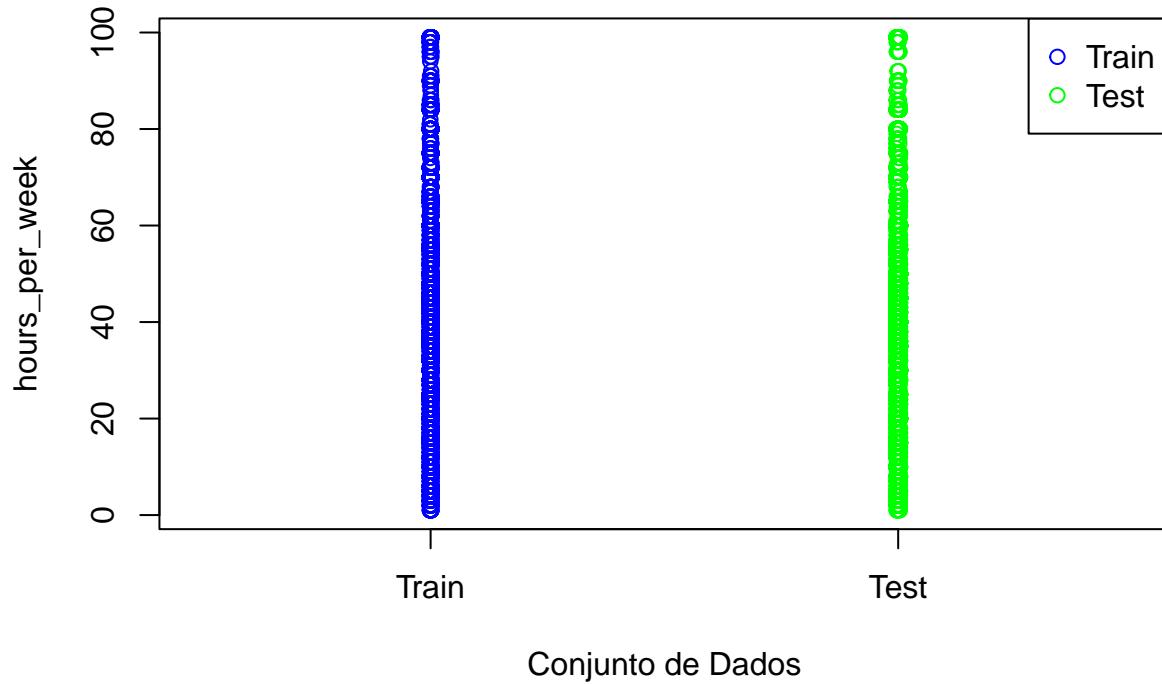


Gráfico de Dispersão Comparativo para hours_per_week



Os boxplots mostram a distribuição dos dados por meio de quartis, mediana, outliers e intervalo interquartil. Ao comparar os boxplots, nota-se que as medidas estatísticas, como quartis e mediana, são próximas entre as bases original e de teste. Isso sugere que os atributos estão sendo preservados de forma consistente na base de teste, com valores semelhantes aos do conjunto original. Os gráficos de dispersão mostram a relação entre dois atributos, permitindo observar a dispersão dos pontos. Ao comparar os gráficos de dispersão, podemos identificar se a relação entre os atributos é preservada na base de teste. Se os pontos seguem uma tendência semelhante e mantêm uma relação coerente, isso indica que a base de teste reflete adequadamente as características do conjunto original. Conclui-se que a base de dados de testes mantém as mesmas características da base de dados original.

8. Identificação e eliminação de atributos não necessários

Foi removido o atributo fnlwgt, pois não é relevante para a tarefa de previsão em questão, e o atributo education-num, pois duplica as informações disponíveis no atributo education.

```
#é a cópia de education
database$education_num <- remove()
#Não é um dado significante para a base de dados
database$fnlwgt <- remove()
```

9. Identificação e eliminação de exemplos não necessários

A única eliminação feita foi em uma linha na base de dados onde todos os atributos se encontram como “N/A”

```
#Achar o único exemplo onde não há nada nele
empty_rows <- !complete.cases(database)
#Printar a linha no qual ele se encontra
print(database[empty_rows, ])

##      age workclass education marital_status occupation relationship race sex
## 32562    NA          NA          NA          NA          NA          NA          NA
##      capital_gain capital_loss hours_per_week native_country income
## 32562          NA            NA           NA          NA          NA

#Remover a linha N/A
database <- na.omit(database)

#Checando se há algum exemplo que está como NA na base de teste
empty_rows <- !complete.cases(test_data)
print(test_data[empty_rows, ])

## [1] age       workclass   fnlwgt     education   education_num
## [6] marital_status occupation relationship race       sex
## [11] capital_gain  capital_loss hours_per_week native_country income
## <0 linhas> (ou row.names de comprimento 0)
```

10 e 11. Análise e aplicação de técnicas de amostragem de dados e desbalanceamento

Para realizar uma análise ou aplicar técnicas de amostragem de dados primeiro é preciso verificar se a base de dados de treinamento e teste estão desbalanceados.

```
# Checar a distribuição da base de treinamento do atributo alvo
table(database$income)
```

```
##
##    <=50K    >50K
##    24720    7841
```

```
# Calcular proporções
prop.table(table(database$income))
```

```
##
##    <=50K    >50K
## 0.7591904 0.2408096
```

```
#Checar a distribuição da base de teste do atributo alvo
table(test_data$income)
```

```

## 
##   <=50K    >50K
##   12434    3846

prop.table(table(test_data$income))

```

```

## 
##   <=50K    >50K
## 0.7637592 0.2362408

```

Como podemos ver ambas as bases estão desbalanceadas, e por isso vamos aplicar técnicas de amostragem de dados e desbalanceamento, a técnica de amostragem utilizada foi amostragem aleatória simples (Sem reposição de exemplos) e para técnicas de desbalanceamento foi utilizado a técnica de undersampling (Onde igualamos os números de casos da classe majoritária com o número de casos da classe minoritária).

```

# Função para realizar amostragem estratificada
amostra_estratificada <- function(data, target_col) {
  # Separar as classes
  class_min <- subset(data, data[[target_col]] == ">50K")
  class_maj <- subset(data, data[[target_col]] == "<=50K")

  # Número de exemplos na classe minoritária
  num_min <- nrow(class_min)

  # Amostragem aleatória da classe majoritária
  set.seed(123) # para reproduzibilidade
  class_maj_sampled <- class_maj[sample(nrow(class_maj), num_min, replace = FALSE), ]

  # Combinar as classes amostradas
  balanced_data <- rbind(class_min, class_maj_sampled)

  # Embaralhar os dados para evitar ordenação por classe
  balanced_data <- balanced_data[sample(nrow(balanced_data)),]

  return(balanced_data)
}

# Aplicar a função nas bases de dados
train_data_balanced <- amostra_estratificada(train_data, "income")
test_data_balanced <- amostra_estratificada(test_data, "income")

# Nova distribuição da base de treinamento
table(train_data_balanced$income)

## 
##   <=50K    >50K
##   7841     7841

prop.table(table(train_data_balanced$income))

```

```

## 
##   <=50K    >50K
##      0.5     0.5

```

```
# Nova distribuição da base de teste

```