MAE 5905: Introdução à Ciência de Dados

Lista 2. Primeiro Semestre de 2023. Entregar 12/05/2023.

Alunos: Leonardo Makoto - 7180679 Leonardo Lima

Preliminares

```
library(ISLR2) #pacote dos dados do livro ISLR
library(tidyverse) #pacote de manipulação de dados
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(leaps)
library(glmnet)
```

```
## Carregando pacotes exigidos: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-7
```

```
set.seed(9845)
```

Exerício 1

item 1a

(a) Use a função rnorm() (simula valores de uma distribuiçãao normal) do R para gerar um preditor X com n = 100 observações, bem como um erro também de comprimento 100.

```r
# criando a variável x
x <- rnorm(100)

cat("variável x:", "\n")
```

```
## variável x:
```

```r
x
```

```
##   [1] -1.278203636  0.286892860 -0.216949783 -0.421134147  1.043960670
##   [6] -0.734620988 -0.390856275 -1.829554205 -1.446438083 -0.987232824
##  [11] -1.386305006 -0.486963149 -0.465549597  0.406320108  0.481055065
##  [16]  1.288844354 -1.224094750  0.873712252 -1.107984379  1.123315106
##  [21] -0.900842949  1.501863285  0.504898845 -0.347773112 -0.714499393
##  [26]  0.309550669  1.547256491  0.486789333 -0.212875398 -0.118627210
##  [31]  0.774214749 -0.547478966  1.871261964  1.028319496 -1.287307594
##  [36]  0.246030006  1.294643360 -0.030815527  1.290097735 -1.783342295
##  [41]  0.971181214  0.155039227 -1.635324726 -0.814569914 -0.525552674
##  [46]  0.330473098  0.719466453 -0.556728179  1.019973897 -1.332373539
##  [51] -0.346760229 -1.439807234 -0.812576313  0.514104699  0.266023656
##  [56] -0.129620571  0.505046818  0.604663110 -0.216911480 -0.892006665
##  [61]  1.026016274 -0.256815654  0.123516628 -1.388996725 -1.163719264
##  [66] -0.522207690  1.481489665 -0.998660547  0.821010077  1.064413490
##  [71] -0.715376813 -0.354330938 -0.306926144  0.967646250 -0.414352653
##  [76]  0.607431712 -1.090953572 -0.148759004  0.138230885  0.002197272
##  [81]  0.568857932 -0.096385594 -1.275033003 -0.531277781  0.003730824
##  [86]  2.737175369  0.239455355  0.136095118  1.779326754  0.874270379
##  [91] -1.046446783  0.040144746 -0.150703549 -0.103491397  0.002927862
##  [96]  0.493142840  1.080788415 -0.634675772 -0.027025665 -1.230964310
```

```r
# criando o termo de erro
e <- rnorm(100)
cat("o termo de erro e:", "\n")
```

```
## o termo de erro e:
```

```r
e
```

```
##   [1]  0.590194054 -0.799633152  0.912490454  2.214096141 -0.165503864
##   [6] -1.285979770 -0.579976527  1.215028845 -0.023632125 -0.226572865
##  [11]  0.030437962 -2.668859556  1.575776255 -0.749287686  0.879224633
##  [16] -0.118735584 -0.454950387 -2.040809168 -0.752959030  0.787572403
##  [21] -0.400499365 -1.226618994  1.477633102  0.465401368  0.752827893
##  [26] -0.377349708  0.095027998  0.147591623  0.588256955  0.384172551
##  [31] -1.093724898 -0.217057105  1.154311078  0.067850514  0.178732780
##  [36] -0.320676479 -0.544486353 -0.499337817  0.999454288 -0.228275123
##  [41] -0.430494198  0.309016418  1.018728786 -0.154010602  0.849362067
##  [46]  0.856936741 -1.651118898  1.014647084 -0.898437539  0.456994152
##  [51] -0.234184964 -1.703710860 -1.052452578  0.679292290  0.599205366
##  [56]  0.965718601 -0.320030764  0.671098933 -0.462065222  0.245028278
##  [61] -1.836747873  2.259733701 -0.692729381  0.412920090  0.735586447
##  [66] -0.329073480 -1.265359069 -0.055535321 -0.886635226 -0.400490029
```

```
##  [71]  0.050453667 -0.785413865  0.294226920  1.049358428 -1.564589302
##  [76]  0.003820687  0.346836102  0.059911753  1.895881822  0.899610485
##  [81] -1.284080078 -1.946526831 -1.382680248 -0.745324459 -0.099464005
##  [86] -0.492861280 -0.531193539  0.268388388  2.818215963  0.426146738
##  [91] -1.359994462  0.189427340  0.735353226 -0.121820353  0.134641509
##  [96] -1.163358520  0.211480221  0.231614721 -0.558379131 -0.562674496
```

item 1b

(b) Simule um vetor de resposta Y, de comprimento n = 100, de acordo com o modelo $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \varepsilon$, em que os parâmetros β_i são constantes de sua escolha.

```
# criando Y
y <- 5 + 3*x - 0.5*x^2 + x^3 + e
```

item 1c

(c) Considere o modelo de (b), agora com os β_i e ε desconhecidos, X como em (a) e Y como em (b). Qual seria o melhor modelo usando R^2 ajustado e BIC?

```
# criando o modelo simples
mod_simples <- lm(y ~ x)
summary(mod_simples)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3832 -1.1313 -0.0206  0.8813 10.6708
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.7215     0.1807   26.14   <2e-16 ***
## x             5.1468     0.1970   26.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.805 on 98 degrees of freedom
## Multiple R-squared:  0.8744, Adjusted R-squared:  0.8732
## F-statistic: 682.5 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
# criando o modelo quadrático
mod_quad <- lm(y ~ poly(x,2))
summary(mod_quad)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 2))
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3394 -1.1069  0.0262  0.9420  8.4284
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.5442     0.1771  25.657   <2e-16 ***
## poly(x, 2)1  47.1627     1.7711  26.629   <2e-16 ***
## poly(x, 2)2   3.8879     1.7711   2.195   0.0305 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.771 on 97 degrees of freedom
## Multiple R-squared:  0.8804, Adjusted R-squared:  0.8779
## F-statistic:   357 on 2 and 97 DF,  p-value: < 2.2e-16
```

```
# criando o modelo cúbico
mod_cubico <- lm(y ~ poly(x,3))
summary(mod_cubico)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 3))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.61314 -0.55728 -0.02967  0.65723  2.83134
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.54419    0.09821  46.269  < 2e-16 ***
## poly(x, 3)1 47.16272    0.98212  48.021  < 2e-16 ***
## poly(x, 3)2  3.88785    0.98212   3.959 0.000145 ***
## poly(x, 3)3 14.54891    0.98212  14.814  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9821 on 96 degrees of freedom
## Multiple R-squared:  0.9636, Adjusted R-squared:  0.9625
## F-statistic: 847.1 on 3 and 96 DF,  p-value: < 2.2e-16
```

```
# modelo de ordem 4
mod_4 <- lm(y ~ poly(x,4))
summary(mod_4)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 4))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.60426 -0.55789 -0.01526  0.64380  2.89771
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.54419    0.09868  46.049  < 2e-16 ***
## poly(x, 4)1 47.16272    0.98682  47.793  < 2e-16 ***
## poly(x, 4)2  3.88785    0.98682   3.940 0.000156 ***
## poly(x, 4)3 14.54891    0.98682  14.743  < 2e-16 ***
## poly(x, 4)4  0.29332    0.98682   0.297 0.766932
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9868 on 95 degrees of freedom
## Multiple R-squared:  0.9636, Adjusted R-squared:  0.9621
## F-statistic: 629.3 on 4 and 95 DF,  p-value: < 2.2e-16
```

```
# comparando o R2 de cada modelo
summary(mod_simples)$adj.r.squared # 0.8731602
```

```
## [1] 0.8731602
```

```
summary(mod_quad)$adj.r.squared # 0.8779173
```

```
## [1] 0.8779173
```

```
summary(mod_cubico)$adj.r.squared # 0.9624596
```

```
## [1] 0.9624596
```

```
summary(mod_4)$adj.r.squared # 0.9621
```

```
## [1] 0.9620997
```

Com base nas métricas R^2 e BIC, o melhor modelo considerando polinômios de ordem até 4 é o modelo cúbico, dado que o R^2 ajustado é o maior e BIC é o menor, como pode ser observado abaixo.

```
## BIC do modelo simples: 413.7257
```

```
## BIC do modelo quadrático: 413.4825
```

```
## BIC do modelo cubico: 299.1233
```

```
## BIC do modelo de ordem 4: 303.6355
```

item 1d

(d) Para o modelo como em (c), obtenha os estimadores ridge e lasso. Use VC para selecionar o valor ótimo de λ.

```r
# vamos criar a base de x e y
base <- data.frame(x,y)

# Vamos criar a nossa matriz X de explicativas
X <- model.matrix(y ~ poly(x,3), base)[,-1]
  # removemos o coeficiente pq ele cria automaticamente.

# Ridge
reg_ridge <- cv.glmnet(X,y, alpha = 0)
```

```
## Coeficientes do Ridge:

## 4 x 1 sparse Matrix of class "dgCMatrix"
##                     s1
## (Intercept)  4.544191
## poly(x, 3)1 43.129599
## poly(x, 3)2  3.555384
## poly(x, 3)3 13.304760

## Valor ótimo de lambda para o Ridge:  0.4716272
##

## Coeficientes do Lasso:

## 4 x 1 sparse Matrix of class "dgCMatrix"
##                     s1
## (Intercept)  4.544191
## poly(x, 3)1 46.752518
## poly(x, 3)2  3.477656
## poly(x, 3)3 14.138711

## Valor ótimo de lambda para o Lasso:  0.04101972
```

Exercício 2

2. Considere o conjunto de dados Weekly do pacote ISLR, contendo 1.089 retornos semanais de ações de 1990 a 2010.

```r
# removendo os dados do exercício anterior
rm(list = ls())

# carrengando o pacote sugerido da questão:
library(astsa)


# carrengando a base de dados do exercício
data(Weekly)
```

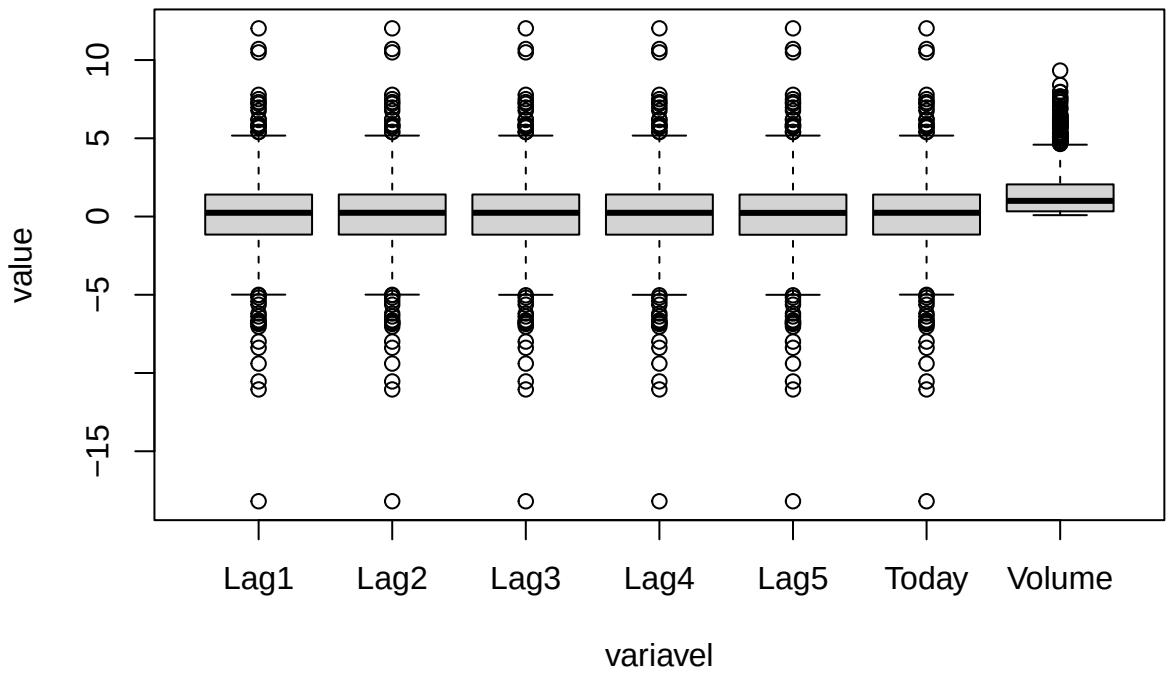
item 2a

(a) Calcule algumas medidas numéricas dos dados, como média, variância, quantis etc. Faça alguns gráficos para sumarizar os dados (use, por exemplo, o pacote astsa).

```
## criando as estatísticas descritivas ----

descritivas <- Weekly %>%
  select(-Direction) %>%
  pivot_longer(cols = 1:ncol(.), names_to = 'variavel') %>%
  group_by(variavel) %>%
  summarise(media = mean(value),
            variancia = var(value),
            desvio_p = sd(value),
            mediana = median(value),
            prim_quartil = quantile(value, probs = 0.25),
            terc_quartil = quantile(value,probs = 0.75),
            minimo = min(value),
            maximo = max(value))

descritivas
```

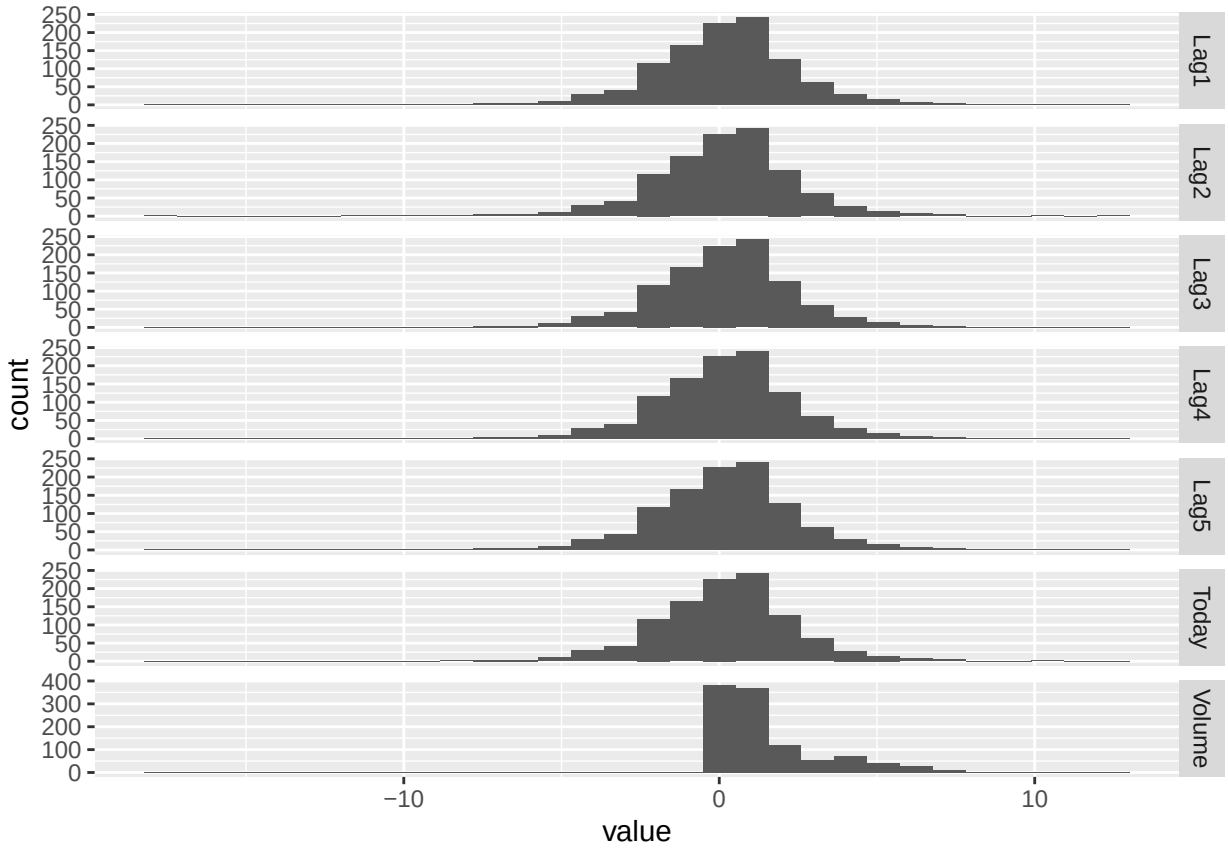
```
## # A tibble: 8 x 9
##   variavel   media variancia desvio_p mediana prim_quartil terc_quartil   minimo
##   <chr>      <dbl>     <dbl>    <dbl>   <dbl>        <dbl>        <dbl>    <dbl>
## 1 Lag1     1.51e-1      5.56     2.36 2.41e-1       -1.15          1.40 -1.82e+1
## 2 Lag2     1.51e-1      5.56     2.36 2.41e-1       -1.15          1.41 -1.82e+1
## 3 Lag3     1.47e-1      5.57     2.36 2.41e-1       -1.16          1.41 -1.82e+1
## 4 Lag4     1.46e-1      5.57     2.36 2.38e-1       -1.16          1.41 -1.82e+1
## 5 Lag5     1.40e-1      5.58     2.36 2.34e-1       -1.17          1.40 -1.82e+1
## 6 Today    1.50e-1      5.56     2.36 2.41e-1       -1.15          1.40 -1.82e+1
## 7 Volume   1.57e+0      2.84     1.69 1.00e+0        0.332         2.05  8.75e-2
## 8 Year     2.00e+3     36.4      6.03 2   e+3     1995          2005     1.99e+3
## # ... with 1 more variable: maximo <dbl>
```

```
## criando as estatísticas descritivas ----


## gráficos das estatísticas ----

week_long <- Weekly %>%
  select(-Direction) %>%
  pivot_longer(cols = 1:ncol(.), names_to = 'variavel')

# boxplot

boxplot(value ~ variavel, data = week_long %>%
          filter(variavel != "Year"))
```




```
# histograma
ggplot(week_long %>%
         filter(variavel != "Year"), aes(x = value)) +
  geom_histogram() +
  facet_grid(variavel ~ ., scales = "free")
```

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



item 2b

(b) Use o conjunto todo de dados e ajuste uma regressão logística, com Direction (up and down) como variável resposta e variável defasada Lag1 como preditora. Comente os resultados.

```
# regressão logística
reg_log_1 <- glm(Direction ~ Lag1, data = Weekly, family = binomial) # 1 para Up e 0 para down

# sumário dos resultados
summary(reg_log_1)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1, family = binomial, data = Weekly)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.456  -1.263   1.041   1.087   1.277
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.23024    0.06124   3.760  0.00017 ***
## Lag1        -0.04313    0.02622  -1.645  0.10001
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1493.5  on 1087  degrees of freedom
## AIC: 1497.5
##
## Number of Fisher Scoring iterations: 4
```

Resultados mostram que Lag1 não é significativo para explicar a direção dos retornos das ações.

item 2c

(c) repita (b), agora tendo como preditores Lag1 e Lag2. Comente.

```
## c ----

# regressão logística para 2 lags
reg_log_2 <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial) # 1 para Up e 0 para down

# sumário dos resultados
summary(reg_log_2)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.623  -1.261   1.001   1.083   1.506
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.22122    0.06147   3.599 0.000319 ***
## Lag1        -0.03872    0.02622  -1.477 0.139672
## Lag2         0.06025    0.02655   2.270 0.023232 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1488.2  on 1086  degrees of freedom
## AIC: 1494.2
##
## Number of Fisher Scoring iterations: 4
```

No caso com 2 Lags, a estatística Lag2 é significativa para explicar a direção dos retornos das ações na semana. O coeficiente demonstra que há uma relação positiva entre o percentual de retorno das duas semanas anteriores com a semana atual. Mais especificamente,cada percentual a mais de 2 semanas atrás aumenta em 1,06 a chance da direção das ações ser Up esta semana.

item 2d

(d) Ajuste uma regressão logística usando como período de treinamento os dados de 1990 a 2008, com Lag2 como preditor. Obtenha a matriz de confus~ao e a taxa de erro de classificaç~ao para o período de teste, 2009-201.

```
# separando a base de treino
Weekly_train <- Weekly %>%
  filter(Year<=2008)

# separando a base de teste
Weekly_teste <- Weekly %>%
  filter(Year > 2008)

# calibrando a função com base na amostra de treino
fit.log <- glm(Direction ~ Lag2, family = binomial, data = Weekly_train)

# vamos fazer a previsão com a amostra de teste
log.probs <- predict(fit.log, Weekly_teste, type = "response")
  # response é para retornar as probs, não log.

# vamos supor que se prob > 0.5, classificamos como Up
log.previsao <- rep("Down", 104) # 104 porque há 104 observações na amostra de teste

log.previsao[log.probs > 0.5] <- "Up"
```

```
## [1] "Criando a tabela de confusão:"
```

```
##
## log.previsao Down Up
##         Down    9  5
##         Up     34 56
```

```
## [1] "A taxa de erro de classificação será a soma das classificações erradas sobre total:"
```

item 2e

(e) repita (d) usando KNN, com K=1.

```
## e ----

# carregando o pacote class para realizar KNN
library(class)

# realizando a previsão

# como os dados precisam ser imputados em matriz, precisarei converter para matrix as variáveis
knn.previsao <- knn(as.matrix(Weekly_train$Lag2), as.matrix(Weekly_teste$Lag2), Weekly_train$Direction,
```

```
## [1] "Matriz de confusão para KNN:"
```

```
##
## knn.previsao Down Up
##         Down   21 30
##         Up     22 31
```

```
## [1] "calculando a taxa de erro para knn:"
```

```
## [1] 0.5
```

item 2f

(f) Qual método fornece os melhores resultados?

Considerando a taxa de erro de classificação como métrica de seleção, a regressão logística fornece os resultados mais precisos.

Exercício 3.

3. Considere o conjunto de dados Auto do pacote ISLR.

```
library(ISLR)
```

```
##
## Attaching package: 'ISLR'
```

```
## The following objects are masked from 'package:ISLR2':
##
##     Auto, Credit
```

```
library(tidyverse)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(ggplot2)
library(cowplot)
library(class)
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## The following object is masked from 'package:ISLR2':
##
##     Boston
```

```
set.seed(123)
```

item 3a

(a) Crie uma variável binária, mpg1, que é igual a 1 se mpg for maior que sua mediana, e mpg1 igual a zero, se mpg for menor que sua mediana. (Use a funç~ao data.frame () para criar um conjunto de dados contendo mpg1 e as outras variáveis do conjunto Auto).

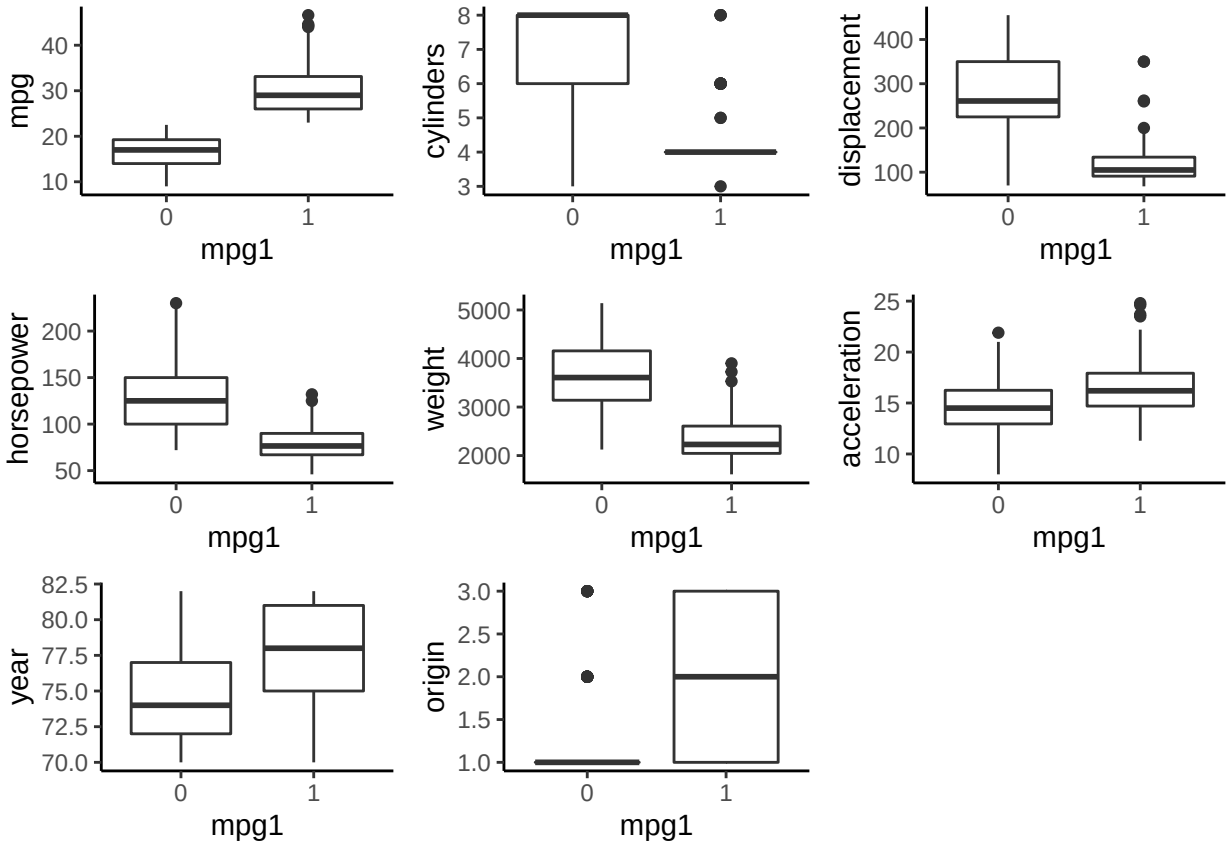
```
db_1 <- Auto
db_2 <- db_1 %>%
  mutate(
    mpg1 = case_when(
      mpg > median(mpg) ~ 1,
      mpg < median(mpg) ~ 0
      )
    )

# Visualizando o conjunto de dados resultante
head(db_2)
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
##                       name mpg1
## 1 chevrolet chevelle malibu    0
## 2         buick skylark 320    0
## 3        plymouth satellite    0
## 4             amc rebel sst    0
## 5               ford torino    0
## 6          ford galaxie 500    0
```

item 3b

(b) Faça gráficos para investigar a associaç~ao entre mpg1 e as outras variáveis (e.g., draftsman display, boxplots). Divida os dados em conjunto de treinamento e de teste.

```r
plots <- list()


# lista com os nomes das variáveis
var_names <- c("mpg", "cylinders",
               "displacement", "horsepower", "weight",
               "acceleration", "year", "origin")

# inicialize uma lista vazia para armazenar os plots
plots <- list()

# loop através dos nomes das variáveis
for (i in 1:length(var_names)) {
  # criar o plot usando ggplot
  plot <- ggplot(db_2, aes(x = as.factor(mpg1), y = .data[[var_names[i]]])) +
    geom_boxplot() +
    xlab("mpg1") +
    ylab(var_names[i]) +
    theme_classic()

  # adicionar o plot à lista
  plots[[i]] <- plot
}

# exibir os plots
# for (i in 1:length(plots)) {
#   print(plots[[i]])
# }

# exibir todos os gráficos no formato draftsman display
plot_grid(plots[[1]], plots[[2]], plots[[3]],
          plots[[4]], plots[[5]], plots[[6]],
          plots[[7]], plots[[8]],
          ncol = 3, align = "h")
```




```
sample <- sample(c(TRUE, FALSE), nrow(db_2), replace=TRUE, prob=c(0.7,0.3))
train  <- db_2[sample, ]
test   <- db_2[!sample, ]
```

item 3c

(c) Use análise discriminante linear de Fisher para prever mpg1 usando os preditores que você acha que sejam mais associadas com ela, usando o item (b). Qual a taxa de erros do conjunto teste?

Para prever mpg1 com os preditores, vamos calcular o discriminante para todas combinações dos preditores numéricos: cylinders, displacement, horsepower, weight, acceleration, year.

Não usamos a variável mpg, nem origin, nem name: mpg explica mpg1 já que uma é função da outra; origin não se mostrou muito correlacionada com mpg1 no boxplot; e name é uma variável categórica.

```
# Lista de variáveis independentes
preditores_possiveis <- names(db_2)[2:7]

# Todas as combinações possíveis de variáveis
preditores_combinacoes <- unlist(
  lapply(
    seq_along(
      preditores_possiveis),
    function(x) combn(preditores_possiveis, x, simplify = FALSE)),
  recursive = FALSE
```

```
  )

# Função para ajustar o modelo e calcular o erro de classificação
fit_lda <- function(x) {
  formula <- as.formula(paste("mpg1 ~", paste(x, collapse = "+")))
  lda_fit <- lda(formula, data = train)
  lda_pred <- predict(lda_fit, newdata = test)
  lda_error <- mean(lda_pred$class != test$mpg1)
  return(list(x = x, lda_fit = lda_fit, lda_pred = lda_pred, lda_error = lda_error))
}

# Aplicar a função em todas as combinações possíveis de variáveis
lda_results <- lapply(preditores_combinacoes, fit_lda)

# Selecionar o modelo com o menor erro de classificação
best_lda <- lda_results[[which.min(sapply(lda_results, function(x) x$lda_error))]]

# Imprimir o modelo selecionado e a matriz de confusão
print(best_lda$lda_fit)
```

```
## Call:
## lda(formula, data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.5107143 0.4892857
##
## Group means:
##   cylinders     year
## 0  6.713287 74.38462
## 1  4.160584 77.58394
##
## Coefficients of linear discriminants:
##                   LD1
## cylinders -0.8381493
## year       0.1079226
```

```
matriz_confusao <- table(best_lda$lda_pred$class, test$mpg1)


# calculando taxa de erro
taxa_erro <- sum(matriz_confusao[row(matriz_confusao) != col(matriz_confusao)]) / sum(matriz_confusao)
```

Variáveis preditoras selecionadas: cylinders, year.

Taxa de erro: 0.0714286.

item 3d

(d) Use KNN, com vários valores de K, e determine a taxa de erros do conjunto teste. Qual valor de K é melhor nesse caso?

```r
# função para ajustar o modelo e calcular o erro de classificação
fit_knn <- function(x, k) {
  knn_pred <- knn(
    as.data.frame(train[, unlist(x)]),
    as.data.frame(test[, unlist(x)]),
                  train$mpg1, k = k
    )
  knn_error <- mean(knn_pred != test$mpg1)
  return(list(x = x, k = k, knn_pred = knn_pred, knn_error = knn_error))
}

# aplicar a função em todas as combinações possíveis de variáveis e valores de k


knn_results <- lapply((preditores_combinacoes), function(x) {
  lapply(seq(1, 11,1), function(k) {
    fit_knn(x, k)

  })
})

# selecionar o modelo com o menor erro de classificação


k_menores_erros <- lapply(knn_results, function(x) which.min(sapply(x, function(y) y$knn_error)))
previsores_menor_erro <- lapply(knn_results, function(x) min(sapply(x, function(y) y$knn_error))) %>%
  which.min()
k_menor_erro <- k_menores_erros[[previsores_menor_erro]]

best_knn_0 <- min(unlist(lapply(knn_results, function(x) min(sapply(x, function(y) y$knn_error)))))

best_knn <- knn_results[[previsores_menor_erro]][[k_menor_erro]]$knn_error




knn_pred_selecionadas <- knn_results[[previsores_menor_erro]][[k_menor_erro]]$x

fischer_pred_selecionadas <- best_lda$x
```

item 3e

(e) Qual classificador você julga que é melhor?

A Taxa de erro do melhor modelo de knn: 0.0625, com o valor de k selecionado: 1 e com as variáveis preditoras selecionadas: cylinders .

A Taxa de erro do melhor modelo de an´alise discriminante linear de Fisher: 0.0714286, com as seguintes variáveis preditoras selecionadas: cylinders, year.

O melhor modelo é o knn, com $k = 1$ usando a variável cylinder.