MBA Executivo em Business Analytics e Big Data

Grupo 1

Junho de 2019

Modelagem Preditiva Avançada - Trabalho Final

A multinacional de varejo Waldata está querendo expandir a sua presença na américa latina e por isso decide firmar uma parceria com a FGV para desenvolver um modelo preditivo do valor de vendas. Além disso a companhia decide apostar em um segundo modelo de 'target ads' tornando mais efetiva as campanhas de marketing. Assim a rede varejista pretende melhorar suas projeções de fluxo de caixa e otimizar a distribuição de seus produtos por departamentos.

Exploração e limpeza dos dados:

```
#Carregando as bibliotecas
library(caret)
library(mlbench)
library(ggplot2)
library(datasets)
library(pROC)
library(ROCR)
library(h2o)
library(DataExplorer)
library(grid)
library(broom)
library(tidyr)
library(dplyr)
library(scales)
library(ggplot2)
library(ggthemes)
library(mlbench)
library(foreach)
library(doParallel)
library(olsrr)
library(lubridate)
library(plotly)

#No R podemos trabalhar com a computação em paralelo por meio de dois pacotes, a saber: foreach e doParallel.
#Com isso aumentamos a velocidade de execução

#Checa quantos núcleos existem
ncl<-detectCores()

#Registra os clusters a serem utilizados
cl <- makeCluster(ncl-1)
registerDoParallel(cl)
```

Passo 1) Importando os datasets RETAIL e MARKETING

```
#Importando os datasets

dataRetail <- read.csv2("RETAIL_1.csv",dec = ".",header = TRUE);
dataMkt <- read.csv2("Marketing.csv",dec = ".",header = TRUE);
```

Passo 2) Exploração dos dados (análise de distribuições, valores faltantes, etc...).

a) Base de Dados RETAIL

```r
#Transformando o campo DATE
dataRetail$DATE <- as.Date(dataRetail$DATE,"%d/%m/%Y")

#Transformando o valor de Store em fator
dataRetail$STORE <- as.factor(dataRetail$STORE)

head(dataRetail)
```

```
##   STORE       DATE TEMPERATURE FUEL_PRICE MARKDOWN1 MARKDOWN2 MARKDOWN3
## 1     1 2010-02-05       42.31      2.572        NA        NA        NA
## 2     1 2010-02-12       38.51      2.548        NA        NA        NA
## 3     1 2010-02-19       39.93      2.514        NA        NA        NA
## 4     1 2010-02-26       46.63      2.561        NA        NA        NA
## 5     1 2010-03-05       46.50      2.625        NA        NA        NA
## 6     1 2010-03-12       57.79      2.667        NA        NA        NA
##   MARKDOWN4 MARKDOWN5           CPI UNEMPLOYMENT ISHOLIDAY WEEKLY_SALES
## 1        NA        NA 2.110.963.582        8.106     FALSE     24924.50
## 2        NA        NA 2.112.421.698        8.106      TRUE     46039.49
## 3        NA        NA 2.112.891.429        8.106     FALSE     41595.55
## 4        NA        NA 2.113.196.429        8.106     FALSE     19403.54
## 5        NA        NA 2.113.501.429        8.106     FALSE     21827.90
## 6        NA        NA 2.113.806.429        8.106     FALSE     21043.39
```

```r
summary(dataRetail)
```

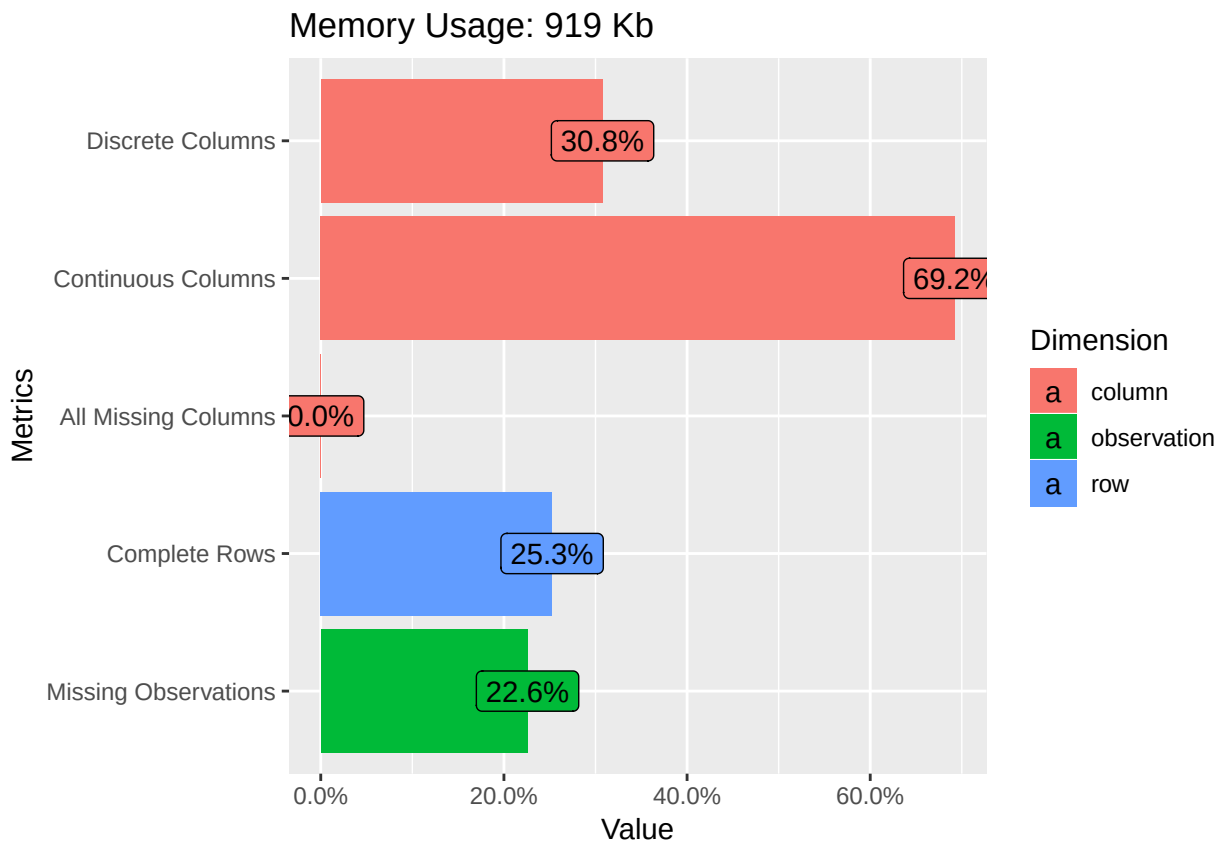
```
##      STORE           DATE              TEMPERATURE       FUEL_PRICE
##  1      : 182   Min.   :2010-02-05   Min.   : -7.29   Min.   :2.472
##  2      : 182   1st Qu.:2010-12-17   1st Qu.: 45.90   1st Qu.:3.041
##  3      : 182   Median :2011-10-31   Median : 60.71   Median :3.513
##  4      : 182   Mean   :2011-10-31   Mean   : 59.36   Mean   :3.406
##  5      : 182   3rd Qu.:2012-09-14   3rd Qu.: 73.88   3rd Qu.:3.743
##  6      : 182   Max.   :2013-07-26   Max.   :101.95   Max.   :4.468
##  (Other):7098
##    MARKDOWN1         MARKDOWN2           MARKDOWN3
##  Min.   : -2781   Min.   :  -265.76   Min.   :  -179.26
##  1st Qu.:  1578   1st Qu.:    68.88   1st Qu.:     6.60
##  Median :  4744   Median :   364.57   Median :    36.26
##  Mean   :  7032   Mean   :  3384.18   Mean   :  1760.10
##  3rd Qu.:  8923   3rd Qu.:  2153.35   3rd Qu.:   163.15
##  Max.   :103185   Max.   :104519.54   Max.   :149483.31
##  NA's   :4158     NA's   :5269        NA's   :4577
##    MARKDOWN4           MARKDOWN5                   CPI
##  Min.   :    0.22   Min.   :  -185.2   1.327.160.968:  33
##  1st Qu.:  304.69   1st Qu.:  1440.8   1.391.226.129:  24
##  Median : 1176.42   Median :  2727.1   2.010.705.712:  12
##  Mean   : 3292.94   Mean   :  4132.2   2.248.025.314:  12
##  3rd Qu.: 3310.01   3rd Qu.:  4832.6   1.260.766.452:  11
##  Max.   :67474.85   Max.   :771448.1   (Other)      :7513
##  NA's   :4726       NA's   :4140       NA's         : 585
##   UNEMPLOYMENT    ISHOLIDAY        WEEKLY_SALES
##  Min.   : 3.684   Mode :logical   Min.   :  -863
##  1st Qu.: 6.634   FALSE:7605      1st Qu.:  2726
##  Median : 7.806   TRUE :585       Median :  7948
##  Mean   : 7.827                   Mean   : 14513
##  3rd Qu.: 8.567                   3rd Qu.: 19408
##  Max.   :14.313                   Max.   :203670
##  NA's   :585
```

Podemos visualizar algumas informações sobre a distribuição dos dados que podem indicar nexcessidade de ajustes nos mesmos:

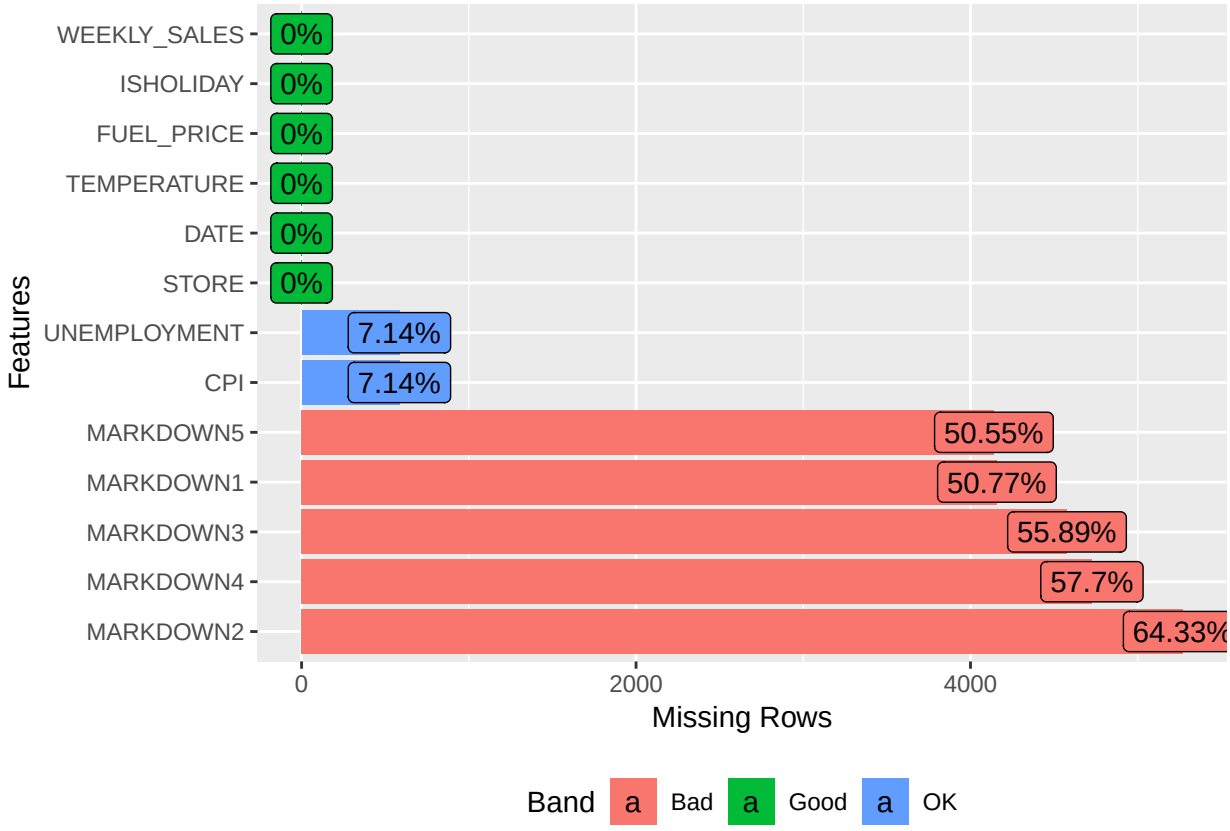
```
## Visualização de dados básicos
introduce(dataRetail)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 8190      13                4                  9                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                24040          2069              106470       941056
```

```
plot_intro(dataRetail)
```

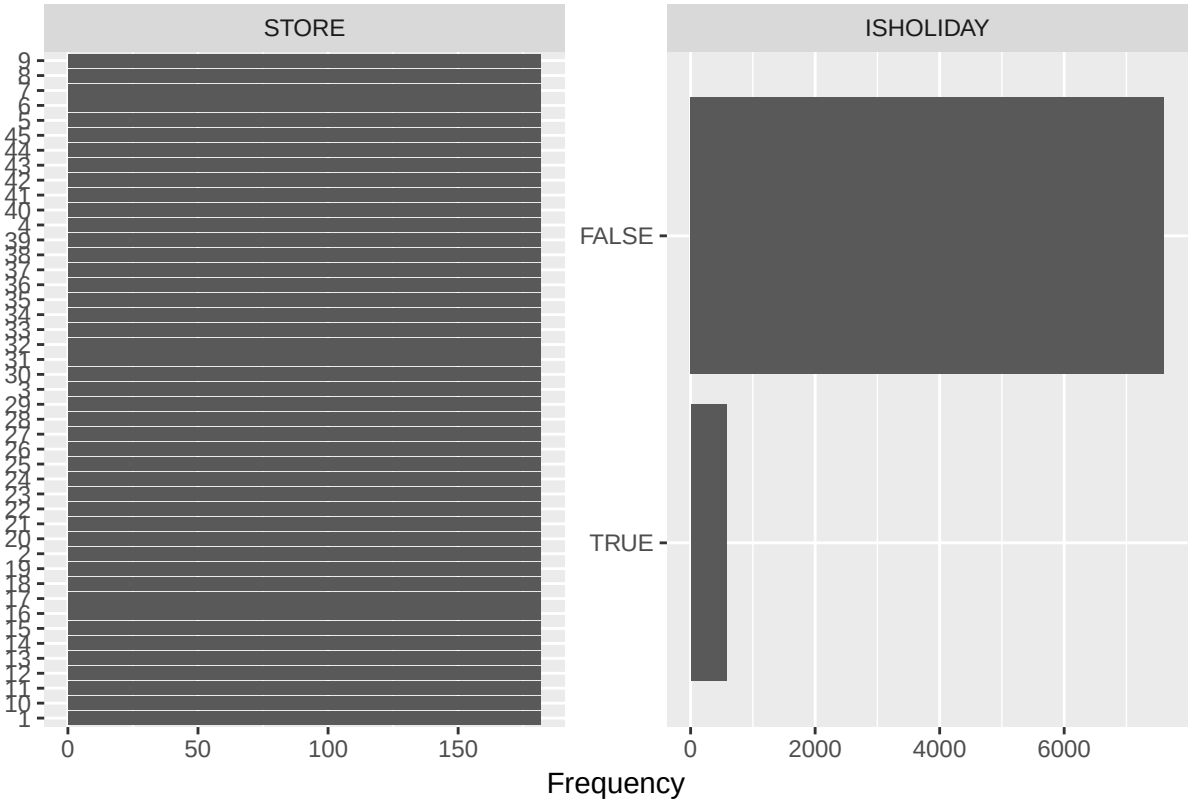



```
## Visualização da distribuição dos dados faltantes
plot_missing(dataRetail)
```

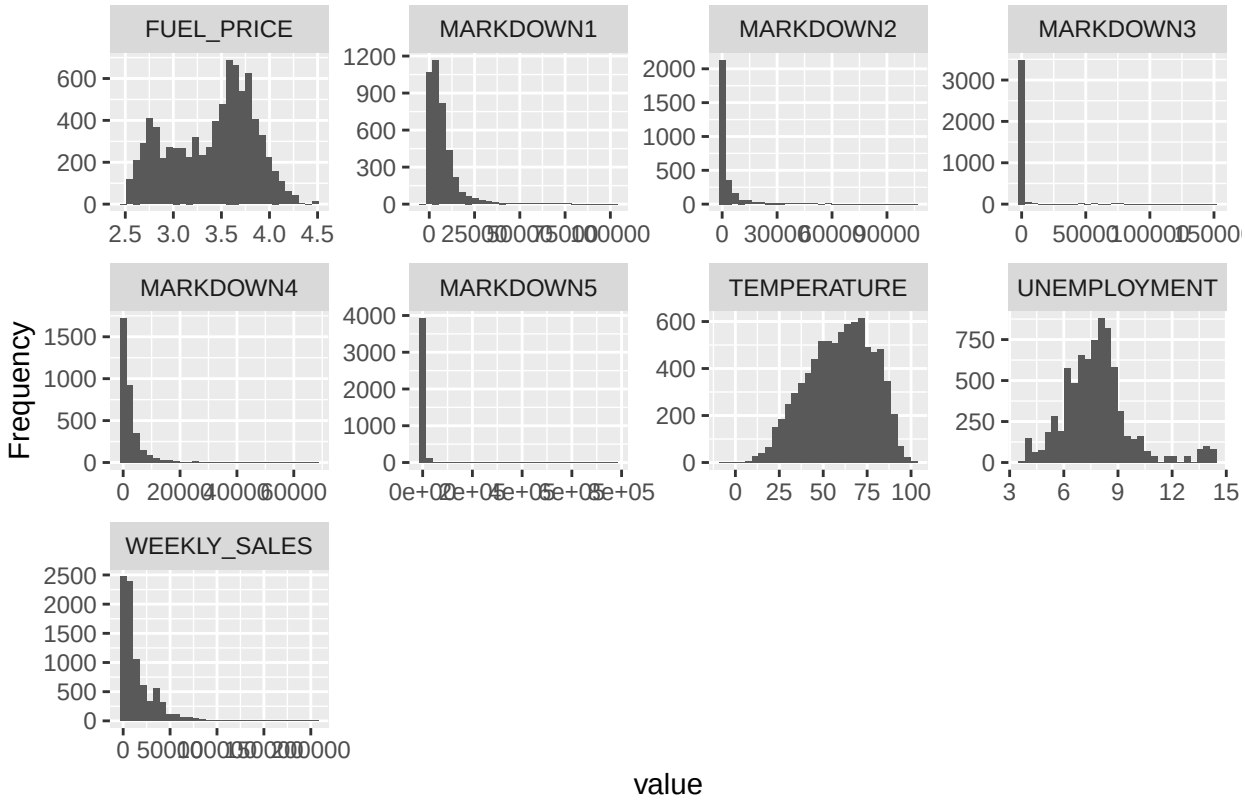



```
## Distribuição de frequênca de todas as variáveis discretas
plot_bar(dataRetail[,which(colnames(dataRetail)!="DATE")])
```

```
## 1 columns ignored with more than 50 categories.
## CPI: 2506 categories
```




```
## Histogramas de todas variáveis continuas
plot_histogram(dataRetail)
```



Verifica-se que os dados faltantes estão concentrados nos campos relacionados aos descontos fornecidos para os produtos e

pode-se sem perda de interpretação assumir com o valor 0 nestes registros.

```
#Ajusta para o valor 0 todos os NS das colunas "MARKDOWN*"
dataRetail[,which(startsWith(colnames(dataRetail),"MARKDOWN"))]<-apply(dataRetail[,which(startsWith(colnames(
```

A coluna "CPI" apresenta os dados com formatação errada e é necessário fazer uma normalização:

```
#Executando as correções
dataRetail$CPI <- substr(gsub(pattern = "[.]",replacement = "",x = dataRetail$CPI),1,5)
dataRetail$CPI <- as.double(dataRetail$CPI)/1E2
```

No caso das colunas "UNEMPLOYMENT" e "CPI", e sendo uma série temporal, iremos considerar a última taxa válida anterior ao dado faltante para cada loja.

Para isso utilizaremos uma função para normalização dos valores :

```
na.lomf <- function(x) {

    na.lomf.0 <- function(x) {
        non.na.idx <- which(!is.na(x))
        if (is.na(x[1L])) {
            non.na.idx <- c(1L, non.na.idx)
        }
        rep.int(x[non.na.idx], diff(c(non.na.idx, length(x) + 1L)))
    }

    dim.len <- length(dim(x))

    if (dim.len == 0L) {
        na.lomf.0(x)
    } else {
        apply(x, dim.len, na.lomf.0)
    }
}

dataRetail$UNEMPLOYMENT <- na.lomf(dataRetail$UNEMPLOYMENT)
dataRetail$CPI <- na.lomf(dataRetail$CPI)

summary(dataRetail)
```

```
##      STORE           DATE              TEMPERATURE       FUEL_PRICE
##  1      : 182   Min.   :2010-02-05   Min.   : -7.29   Min.   :2.472
##  2      : 182   1st Qu.:2010-12-17   1st Qu.: 45.90   1st Qu.:3.041
##  3      : 182   Median :2011-10-31   Median : 60.71   Median :3.513
##  4      : 182   Mean   :2011-10-31   Mean   : 59.36   Mean   :3.406
##  5      : 182   3rd Qu.:2012-09-14   3rd Qu.: 73.88   3rd Qu.:3.743
##  6      : 182   Max.   :2013-07-26   Max.   :101.95   Max.   :4.468
##  (Other):7098
##    MARKDOWN1          MARKDOWN2           MARKDOWN3
##  Min.   :  -185.2   Min.   :   -23.97   Min.   :  -563.9
##  1st Qu.:     0.0   1st Qu.:     0.00   1st Qu.:     0.0
##  Median :     0.0   Median :     0.00   Median :     0.0
##  Mean   :  1703.7   Mean   :  2164.52   Mean   :  2342.6
##  3rd Qu.:   989.1   3rd Qu.:  1457.41   3rd Qu.:  1863.1
##  Max.   :112255.7   Max.   :149483.31   Max.   :139621.5
##
##    MARKDOWN4          MARKDOWN5             CPI         UNEMPLOYMENT
##  Min.   : -2781.4   Min.   :   -20.0   Min.   :126.1   Min.   : 3.684
##  1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:132.7   1st Qu.: 6.565
##  Median :     0.0   Median :     0.0   Median :182.8   Median : 7.742
##  Mean   :  1522.1   Mean   :  1148.8   Mean   :172.9   Mean   : 7.748
##  3rd Qu.:   486.2   3rd Qu.:   229.9   3rd Qu.:214.4   3rd Qu.: 8.549
```

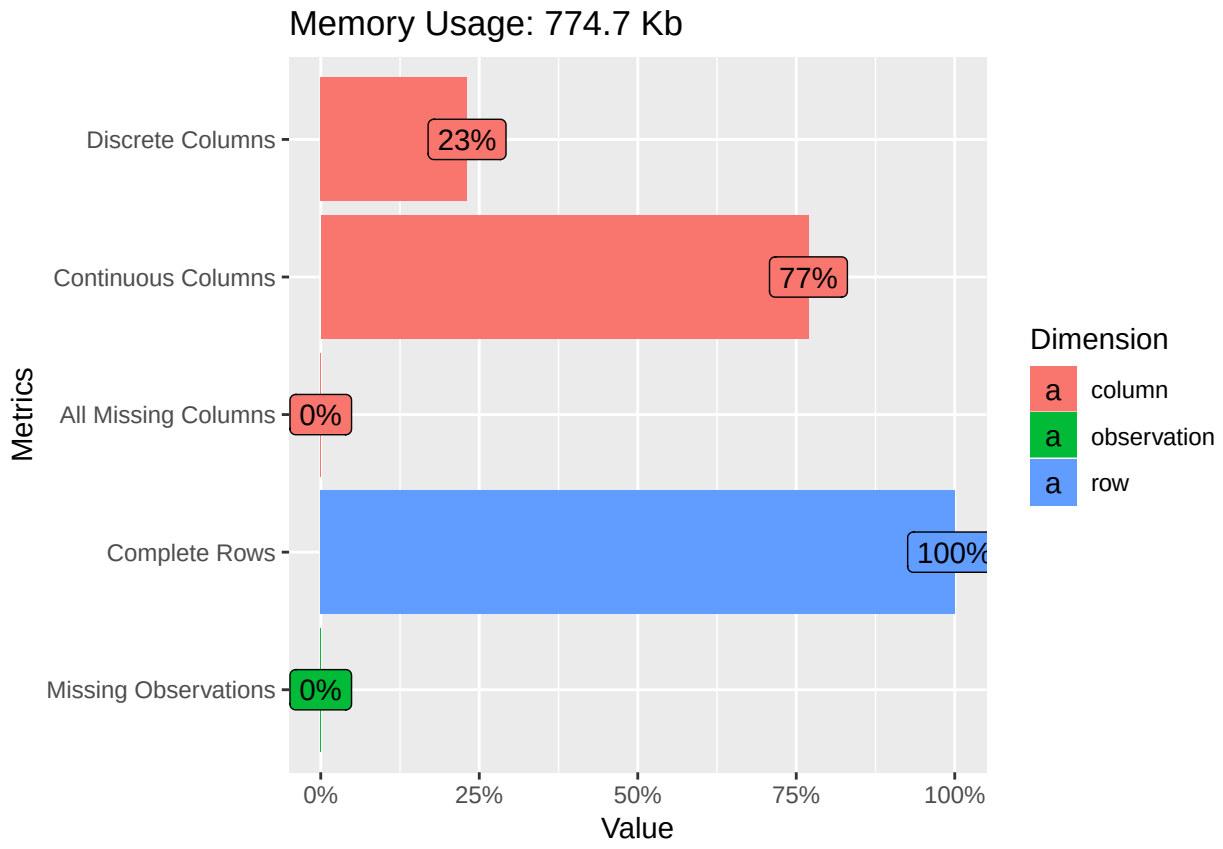
```
##  Max.   :771448.1   Max.   :109976.1   Max.   :229.0   Max.   :14.313
##
##  ISHOLIDAY        WEEKLY_SALES
##  Mode :logical   Min.   :  -863
##  FALSE:7605      1st Qu.:  2726
##  TRUE :585       Median :  7948
##                  Mean   : 14513
##                  3rd Qu.: 19408
##                  Max.   :203670
##
```

Após os ajustes feitos podemos verificar novamente as informações dos dados da base:

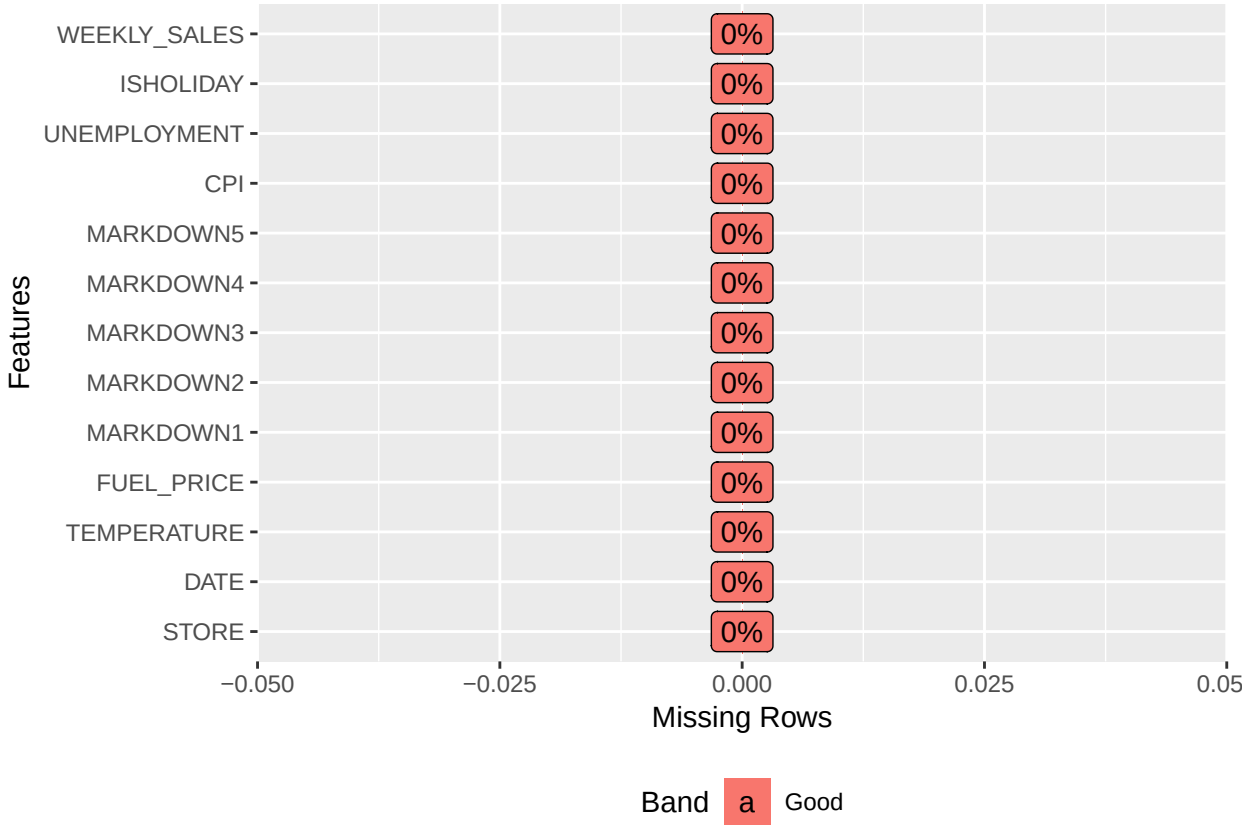
```
## Visualização de dados básicos
introduce(dataRetail)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 8190      13                3                  10                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                    0          8190             106470       793320
```

```
plot_intro(dataRetail)
```

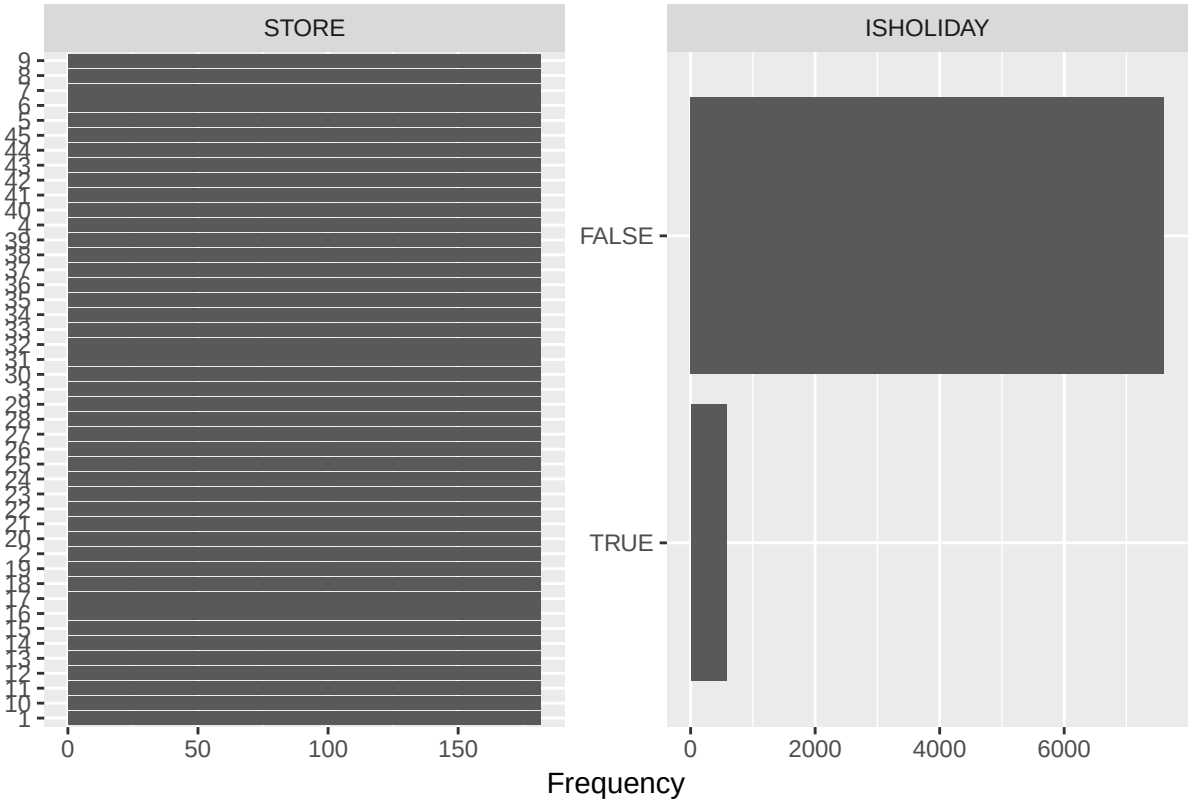



```
## Visualização da distribuição dos dados faltantes
plot_missing(dataRetail)
```

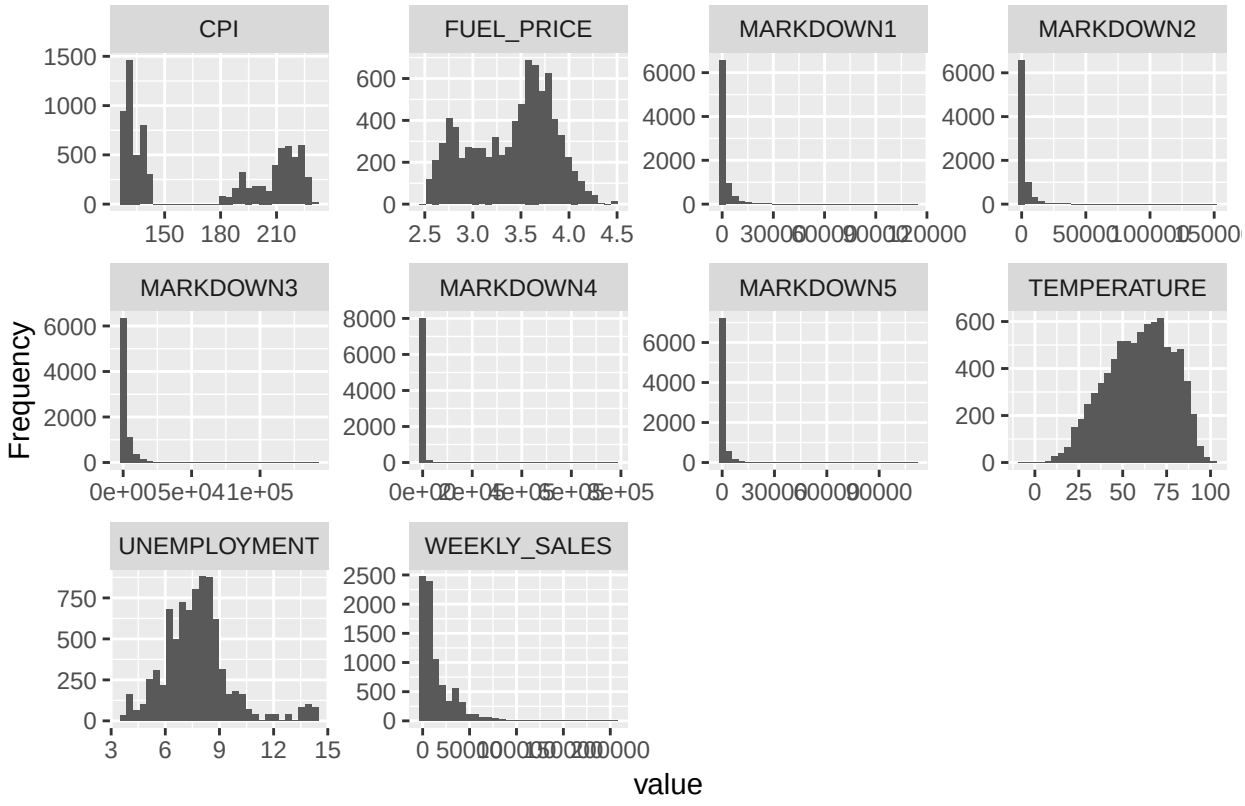



```
## Distribuição de frequênca de todas as variáveis discretas
plot_bar(dataRetail)
```

```
## 1 columns ignored with more than 50 categories.
## DATE: 182 categories
```




```
## Histogramas de todas variáveis continuas
plot_histogram(dataRetail)
```



Temos agora uma base com todas as informações completas:

```
head(dataRetail)
```

```
##   STORE       DATE TEMPERATURE FUEL_PRICE MARKDOWN1 MARKDOWN2 MARKDOWN3
## 1     1 2010-02-05       42.31      2.572         0         0         0
## 2     1 2010-02-12       38.51      2.548         0         0         0
## 3     1 2010-02-19       39.93      2.514         0         0         0
## 4     1 2010-02-26       46.63      2.561         0         0         0
## 5     1 2010-03-05       46.50      2.625         0         0         0
## 6     1 2010-03-12       57.79      2.667         0         0         0
##   MARKDOWN4 MARKDOWN5    CPI UNEMPLOYMENT ISHOLIDAY WEEKLY_SALES
## 1         0         0 211.09        8.106     FALSE     24924.50
## 2         0         0 211.24        8.106      TRUE     46039.49
## 3         0         0 211.28        8.106     FALSE     41595.55
## 4         0         0 211.31        8.106     FALSE     19403.54
## 5         0         0 211.35        8.106     FALSE     21827.90
## 6         0         0 211.38        8.106     FALSE     21043.39
```

b) Base de Dados MARKETING

Verifica-se que alguns valores númericos possuem o caracter "_" no lugar da pontuação decimal:

```
summary(dataMkt)
```

```
##       AGE                 JOB         MARITAL_STATUS
##  Min.   :17.00   admin      :10422   divorced: 4612
##  1st Qu.:32.00   blue-collar: 9254   married :24928
##  Median :38.00   technician : 6743   single  :11568
##  Mean   :40.02   services   : 3969   unknown :   80
##  3rd Qu.:47.00   management : 2924
##  Max.   :98.00   retired    : 1720
##                  (Other)    : 6156
##                EDUCATION        DEFAULT         HOUSING
##  university_degree  :12168   no     :32588   no     :18622
##  high_school        : 9515   unknown: 8597   unknown:  990
##  basic_9y           : 6045   yes    :    3   yes    :21576
##  professional_course: 5243
##  basic_4y           : 4176
##  basic_6y           : 2292
##  (Other)            : 1749
##       LOAN            CONTACT          MONTH       DAY_OF_WEEK
##  no     :33950   cellular :26144   may    :13769   fri:7827
##  unknown:  990   telephone:15044   jul    : 7174   mon:8514
##  yes    : 6248                     aug    : 6178   thu:8623
##                                    jun    : 5318   tue:8090
##                                    nov    : 4101   wed:8134
##                                    apr    : 2632
##                                    (Other): 2016
##     DURATION         CAMPAIGN          PDAYS           PREVIOUS
##  Min.   :   0.0   Min.   : 1.000   Min.   :-1.000   Min.   :0.000
##  1st Qu.: 102.0   1st Qu.: 1.000   1st Qu.:-1.000   1st Qu.:0.000
##  Median : 180.0   Median : 2.000   Median :-1.000   Median :0.000
##  Mean   : 258.3   Mean   : 2.568   Mean   :-0.742   Mean   :0.173
##  3rd Qu.: 319.0   3rd Qu.: 3.000   3rd Qu.:-1.000   3rd Qu.:0.000
##  Max.   :4918.0   Max.   :56.000   Max.   :27.000   Max.   :7.000
##
##         POUTCOME      EMP_VAR_RATE   CONS_PRICE_IDX CONS_CONF_IDX
##  failure    : 4252   1_4    :16234   93.994 :7763   -36_4  :7763
##  nonexistent:35563   -1_8   : 9184   93.918 :6685   -42_7  :6685
##  success    : 1373   1_1    : 7763   92.893 :5794   -46_2  :5794
##                      -0_1   : 3683   93.444 :5175   -36_1  :5175
```

```
##                     -2_9   : 1663   94.465 :4374   -41_8  :4374
##                     -3_4   : 1071   93_2   :3616   -42    :3616
##                     (Other): 1590   (Other):7781   (Other):7781
##  SUBSCRIBED
##  no :36548
##  yes: 4640
##
##
##
##
##
```

```
#Executando as correções nestes campos
dataMkt$CONS_CONF_IDX <- as.double(gsub(pattern = "_",replacement = ".",x = dataMkt$CONS_CONF_IDX))
dataMkt$CONS_PRICE_IDX <- as.double(gsub(pattern = "_",replacement = ".",x = dataMkt$CONS_PRICE_IDX))
dataMkt$EMP_VAR_RATE <- as.double(gsub(pattern = "_",replacement = ".",x = dataMkt$EMP_VAR_RATE))

#Transformando o valor da campanha de Marketing em fator
dataMkt$CAMPAIGN <- as.factor(dataMkt$CAMPAIGN)

head(dataMkt)
```

```
##   AGE       JOB MARITAL_STATUS   EDUCATION DEFAULT HOUSING LOAN   CONTACT
## 1  56 housemaid        married    basic_4y      no      no   no telephone
## 2  57  services        married high_school unknown      no   no telephone
## 3  37  services        married high_school      no     yes   no telephone
## 4  40     admin        married    basic_6y      no      no   no telephone
## 5  56  services        married high_school      no      no  yes telephone
## 6  45  services        married    basic_9y unknown      no   no telephone
##   MONTH DAY_OF_WEEK DURATION CAMPAIGN PDAYS PREVIOUS    POUTCOME
## 1   may         mon      261        1    -1        0 nonexistent
## 2   may         mon      149        1    -1        0 nonexistent
## 3   may         mon      226        1    -1        0 nonexistent
## 4   may         mon      151        1    -1        0 nonexistent
## 5   may         mon      307        1    -1        0 nonexistent
## 6   may         mon      198        1    -1        0 nonexistent
##   EMP_VAR_RATE CONS_PRICE_IDX CONS_CONF_IDX SUBSCRIBED
## 1          1.1         93.994         -36.4         no
## 2          1.1         93.994         -36.4         no
## 3          1.1         93.994         -36.4         no
## 4          1.1         93.994         -36.4         no
## 5          1.1         93.994         -36.4         no
## 6          1.1         93.994         -36.4         no
```

```
summary(dataMkt)
```

```
##       AGE                 JOB         MARITAL_STATUS
##  Min.   :17.00   admin      :10422   divorced: 4612
##  1st Qu.:32.00   blue-collar: 9254   married :24928
##  Median :38.00   technician : 6743   single  :11568
##  Mean   :40.02   services   : 3969   unknown :   80
##  3rd Qu.:47.00   management : 2924
##  Max.   :98.00   retired    : 1720
##                  (Other)    : 6156
##                EDUCATION        DEFAULT         HOUSING
##  university_degree  :12168   no     :32588   no     :18622
##  high_school        : 9515   unknown: 8597   unknown:  990
##  basic_9y           : 6045   yes    :    3   yes    :21576
##  professional_course: 5243
##  basic_4y           : 4176
```

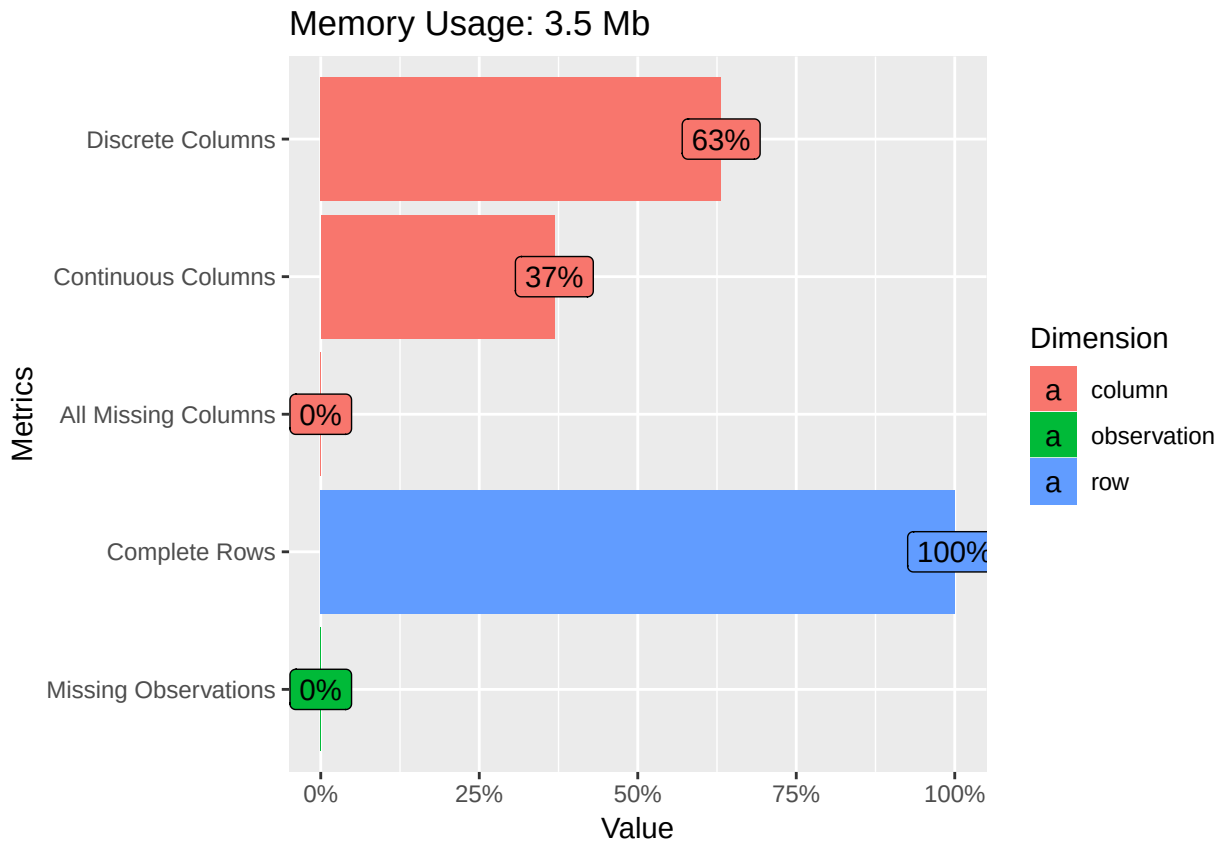
```
##  basic_6y           : 2292
##  (Other)            : 1749
##       LOAN            CONTACT          MONTH       DAY_OF_WEEK
##  no     :33950   cellular :26144   may    :13769   fri:7827
##  unknown:  990   telephone:15044   jul    : 7174   mon:8514
##  yes    : 6248                     aug    : 6178   thu:8623
##                                    jun    : 5318   tue:8090
##                                    nov    : 4101   wed:8134
##                                    apr    : 2632
##                                    (Other): 2016
##     DURATION         CAMPAIGN         PDAYS           PREVIOUS
##  Min.   :   0.0   1      :17642   Min.   :-1.000   Min.   :0.000
##  1st Qu.: 102.0   2      :10570   1st Qu.:-1.000   1st Qu.:0.000
##  Median : 180.0   3      : 5341   Median :-1.000   Median :0.000
##  Mean   : 258.3   4      : 2651   Mean   :-0.742   Mean   :0.173
##  3rd Qu.: 319.0   5      : 1599   3rd Qu.:-1.000   3rd Qu.:0.000
##  Max.   :4918.0   6      :  979   Max.   :27.000   Max.   :7.000
##                   (Other): 2406
##         POUTCOME      EMP_VAR_RATE      CONS_PRICE_IDX  CONS_CONF_IDX
##  failure    : 4252   Min.   :-3.40000   Min.   :92.20   Min.   :-50.8
##  nonexistent:35563   1st Qu.:-1.80000   1st Qu.:93.08   1st Qu.:-42.7
##  success    : 1373   Median : 1.10000   Median :93.75   Median :-41.8
##                      Mean   : 0.08189   Mean   :93.58   Mean   :-40.5
##                      3rd Qu.: 1.40000   3rd Qu.:93.99   3rd Qu.:-36.4
##                      Max.   : 1.40000   Max.   :94.77   Max.   :-26.9
##
##  SUBSCRIBED
##  no :36548
##  yes: 4640
##
##
##
##
##
```

Podemos visualizar algumas informações sobre a distribuição dos dados que podem indicar nexcessidade de ajustes nos mesmos:

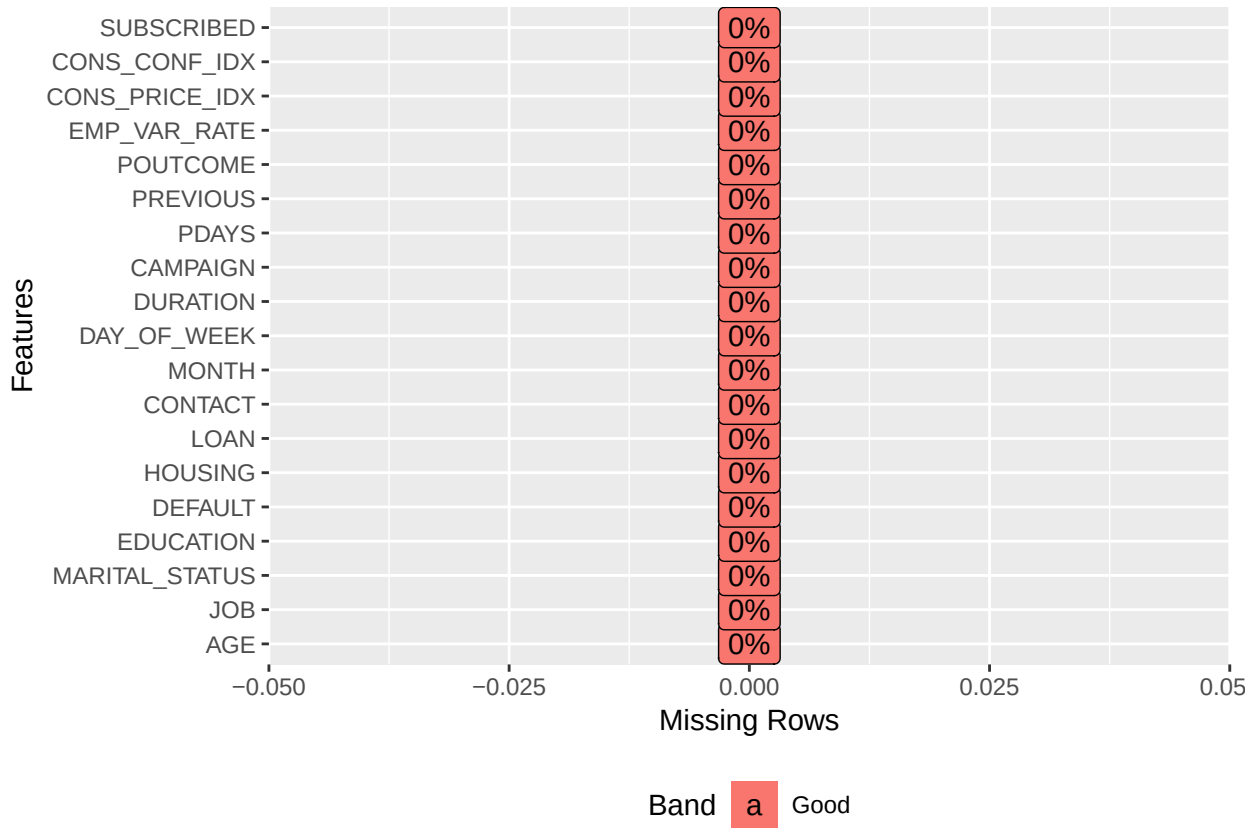
```
## Visualização de dados básicos
introduce(dataMkt)
```

```
##    rows columns discrete_columns continuous_columns all_missing_columns
## 1 41188      19               12                 7                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                    0         41188             782572      3640304
```

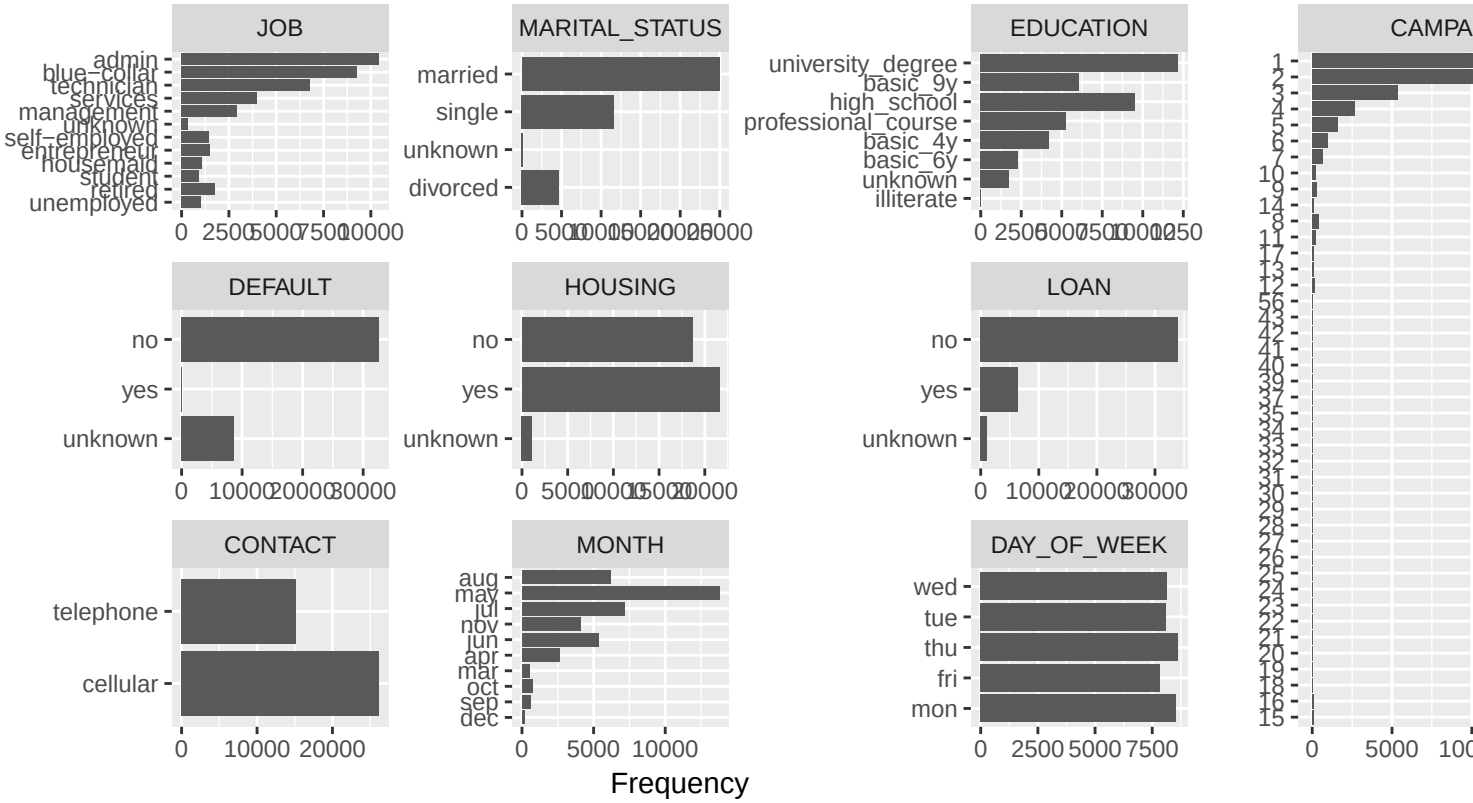
```
plot_intro(dataMkt)
```



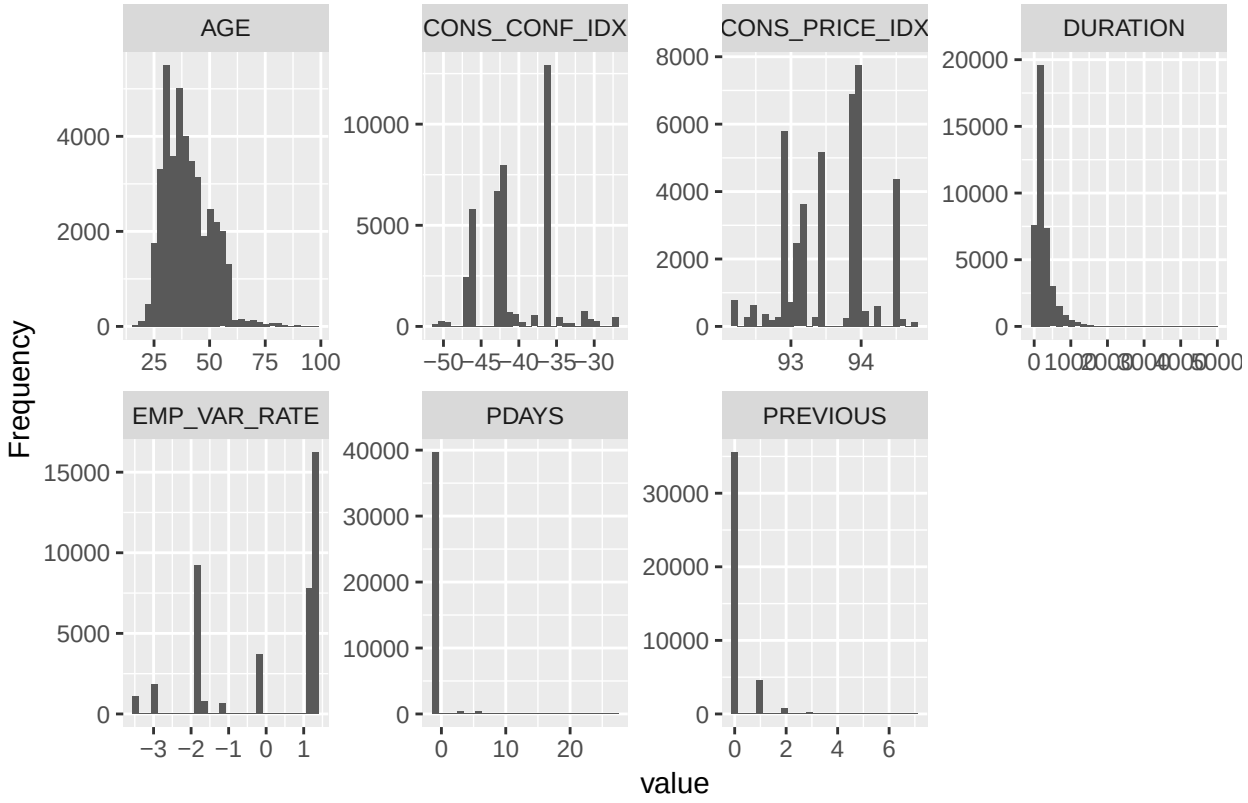

```
## Visualização da distrin=buição dos dados faltantes
plot_missing(dataMkt)
```



```
## Distribuição de frequênca de todas as variáveis discretas
plot_bar(dataMkt)
```




```
## Histogramas de todas variáveis continuas
plot_histogram(dataMkt)
```

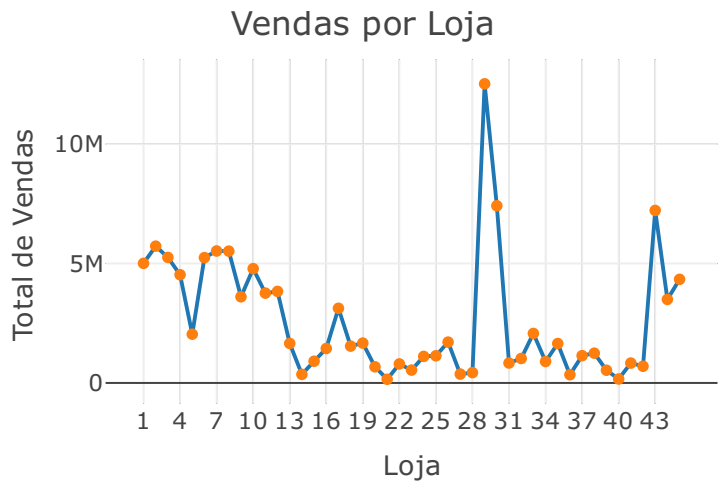


Podemos extrair algumas análises interessantes de forma a esclarecer o entedimento do problema.

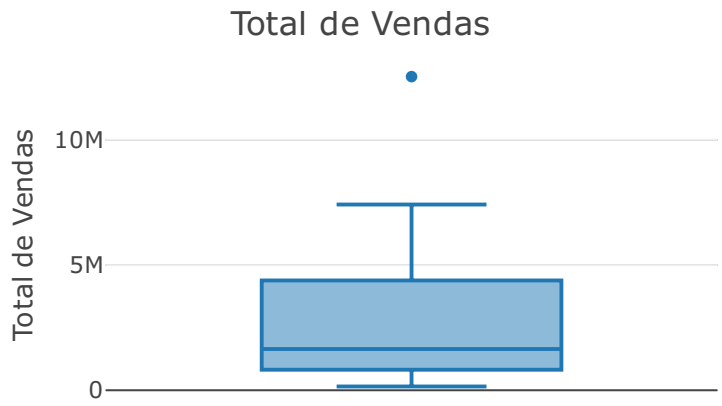
Por exemplo, verificar qual a contribuição de cada loja para o total de vendas:

```
dataRetail %>% group_by(STORE) %>% summarize(Total_Vendas = sum(WEEKLY_SALES)) %>%
            arrange(desc(Total_Vendas)) %>%
            plot_ly(x = ~STORE, y = ~Total_Vendas) %>%
            add_lines() %>%
            add_trace(x = ~STORE, y = ~Total_Vendas, mode = 'markers',type = 'scatter') %>%
            layout(title = "Vendas por Loja",
                   xaxis = list(title = "Loja"),
                   yaxis = list(title = "Total de Vendas"),showlegend = FALSE)
```

```
## Warning: package 'bindrcpp' was built under R version 3.5.2
```



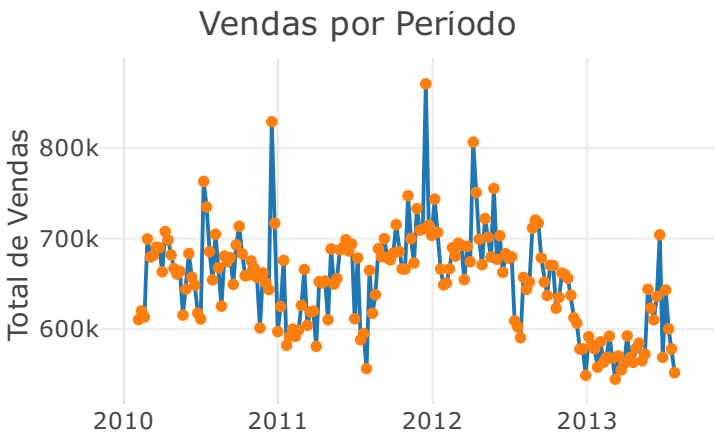

```
dataRetail %>% group_by(STORE) %>% summarize(Total_Vendas = sum(WEEKLY_SALES)) %>%
             arrange(desc(Total_Vendas)) %>%
             plot_ly(y = ~Total_Vendas,type = "box") %>%
             layout(title = "Total de Vendas",
                    xaxis = list(title = "",showticklabels = FALSE),
                    yaxis = list(title = "Total de Vendas"),showlegend = FALSE)
```



Observa-se que a loja 29 obteve um volume de venda total bem mais elevado que as demais lojas e poderia ser tratado como um outlier dentro da base de informações.

Outra análise interessante é o total de vendas por período:

```
dataRetail %>% group_by(DATE) %>% summarize(Total_Vendas = sum(WEEKLY_SALES)) %>%
               arrange(desc(Total_Vendas)) %>%
               plot_ly(x = ~DATE, y = ~Total_Vendas) %>%
               add_lines() %>%
               add_trace(x = ~DATE, y = ~Total_Vendas, mode = 'markers',type = 'scatter') %>%
               layout(title = "Vendas por Período",
                  xaxis = list(title = ""),
                  yaxis = list(title = "Total de Vendas"),showlegend = FALSE)
```



Após os procedimentos de entedimento e ajustes dos dados pode-se passar para a fase de modelagem.

Passo 3) Dividir as bases em 70% para treino e 30% para teste do modelo. (Utilize sempre seed(314))

Dividimos a base de Vendas criando os grupos de treino e teste Para isso definimos "p=0.7", isto é 70% da base será escolhida aleatóriamente para treino e 30% para teste do modelo setando o seed para 314, para garantir que ao replicarmos essa partição em outro computador por exemplo, os mesmos dados irão respectivamente prar treino e teste.

```
set.seed(314)
trainIndex_Retail <- createDataPartition(dataRetail$WEEKLY_SALES, p = .7, list = FALSE)

dfTrain_Retail <- dataRetail[trainIndex_Retail,]
dfTest_Retail  <- dataRetail[-trainIndex_Retail,]

#Dividimos a base de Marketing criando os grupos de treino e teste
set.seed(314)
trainIndex_Mkt <- createDataPartition(dataMkt$SUBSCRIBED, p = .7, list = FALSE)

dfTrain_Mkt <- dataMkt[trainIndex_Mkt,]
dfTest_Mkt  <- dataMkt[-trainIndex_Mkt,]
```

Passo 4) Testar modelos de classificação para as campanhas de Marketing:

a) Regressão Logística

```
set.seed(314)

if (file.exists("modelMkt_GLM.rdata")) {
```

```
  load("modelMkt_GLM.rdata")
} else {
  cv <- trainControl(method = "repeatedcv", number = 10, savePredictions = TRUE,
                     summaryFunction=twoClassSummary, classProbs = TRUE)

  modelMkt_GLM <- train(SUBSCRIBED ~ DURATION + EMP_VAR_RATE + CONTACT + PREVIOUS + CONS_PRICE_IDX + PDAYS +
                          metric="ROC",trControl = cv, control = list(maxit = 50))

}

modelMkt_GLM
```

```
## Generalized Linear Model
##
## 28832 samples
##    16 predictor
##     2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 25949, 25948, 25949, 25949, 25949, 25949, ...
## Resampling results:
##
##   ROC        Sens       Spec
##   0.9321541  0.9720916  0.4104236
```

Quando as colunas de uma matriz forem combinações lineares umas das outras, dizemos que a matriz é rank deficient (ou posto incompleto, em português). O problema é que matrizes assim não são invertíveis. Portanto, não dá para estimar os parâmetros da regressão. Possíveis causas: 1) Uma das variáveis preditoras é combinação linear das demais. Ou seja, alguma variável no modelo é redundante. 2) Talvez a amostra não seja grande o suficiente para o modelo a ser ajustado. 3) O modelo pode ter parâmetros demais e tamanho amostral de menos.

A regra geral é ter pelo menos uma quantidade de pontos igual ao número de parâmetros a serem ajustado no modelo. Assim se garante que a matriz não será rank-deficient.

Usando a base de teste para verificação do modelo:

```
#Usando a base de teste para verificação do modelo
dataMktPred <- predict(modelMkt_GLM, newdata=dfTest_Mkt)
```

Gerando Matriz de Confusão:

```
#Verificando o resultado através da Matriz de Confusão
cmMkt_GLM <- confusionMatrix(data=dataMktPred, dfTest_Mkt$SUBSCRIBED)
cmMkt_GLM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##        no  10678   793
##        yes   286   599
##
##                Accuracy : 0.9127
##                  95% CI : (0.9076, 0.9176)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4806
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
```

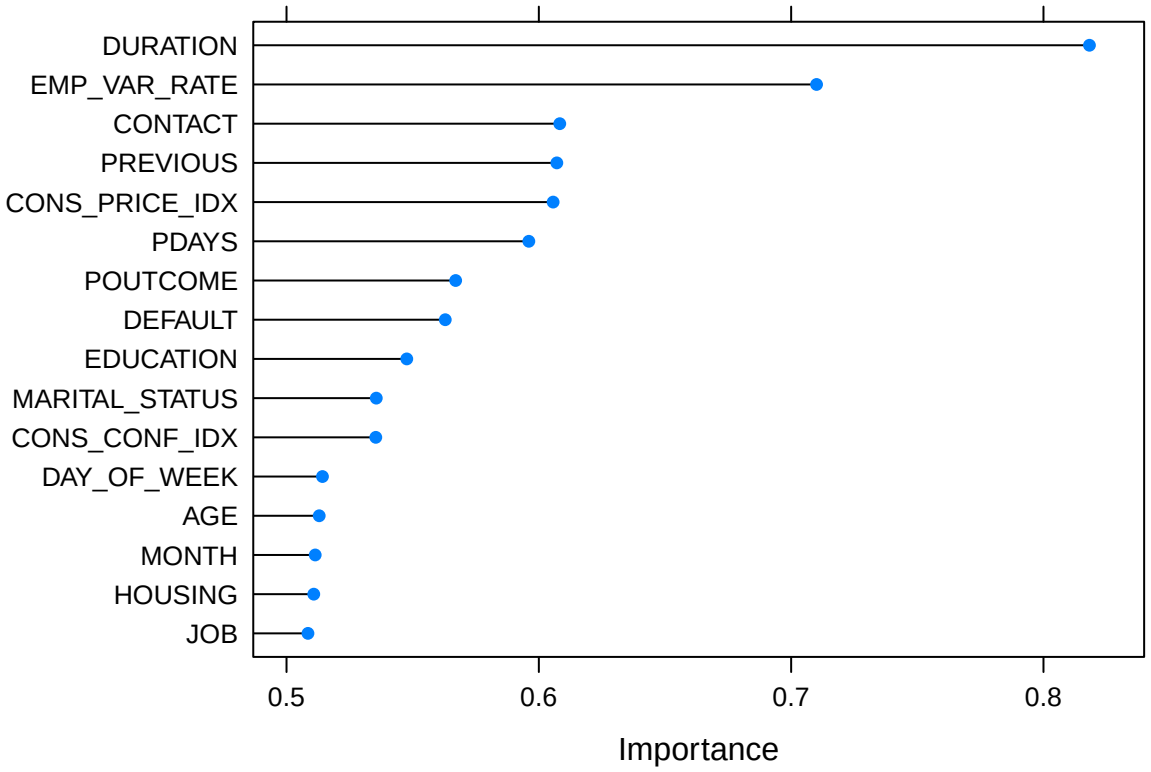
```
##             Sensitivity : 0.9739
##             Specificity : 0.4303
##          Pos Pred Value : 0.9309
##          Neg Pred Value : 0.6768
##              Prevalence : 0.8873
##          Detection Rate : 0.8642
##    Detection Prevalence : 0.9284
##       Balanced Accuracy : 0.7021
##
##        'Positive' Class : no
##
```

Importância das Variáveis Preditoras:

```
imp <- varImp(modelMkt_GLM, useModel=FALSE, scale=FALSE)
imp
```

```
## ROC curve variable importance
##
##                Importance
## DURATION           0.8182
## EMP_VAR_RATE       0.7101
## CONTACT            0.6083
## PREVIOUS           0.6071
## CONS_PRICE_IDX     0.6057
## PDAYS              0.5960
## POUTCOME           0.5670
## DEFAULT            0.5629
## EDUCATION          0.5477
## MARITAL_STATUS     0.5356
## CONS_CONF_IDX      0.5354
## DAY_OF_WEEK        0.5142
## AGE                0.5130
## MONTH              0.5114
## HOUSING            0.5108
## JOB                0.5085
```

```
plot(imp)
```



Gerando a curva ROC:

```
dfProbs <- predict(modelMkt_GLM, newdata=dfTest_Mkt, type="prob")
head(dfProbs)
```

```
##           no         yes
## 1  0.9872625 0.012737459
## 2  0.9951136 0.004886442
## 4  0.9908011 0.009198859
## 7  0.9927951 0.007204881
## 10 0.9952302 0.004769841
## 12 0.9894477 0.010552340
```

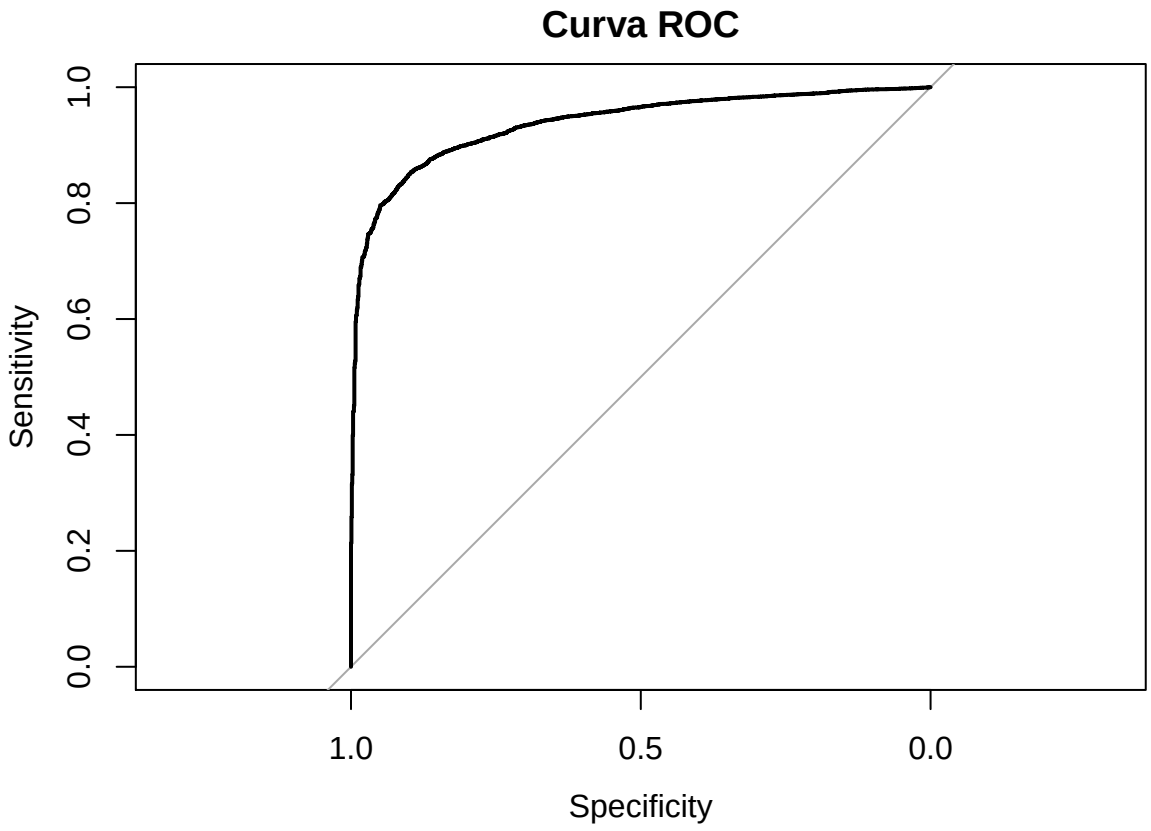
```
gbm.ROC <- roc(predictor=dfProbs$yes,
               response=dfTest_Mkt$SUBSCRIBED,
               levels=rev(levels(dfTest_Mkt$SUBSCRIBED)))
```

```
## Setting direction: controls > cases
```

```
gbm.ROC$auc
```

```
## Area under the curve: 0.9375
```

```
plot(gbm.ROC,main="Curva ROC")
```



Salvando o modelo:

```
save(modelMkt_GLM,file="modelMkt_GLM.rdata")
```

b) Árvores de Decisão

Modelo Random Forest:

```
# Definindo Parâmetros do Cross Validation
set.seed(314)

if (file.exists("modelMkt_RF.rdata")) {
  load("modelMkt_RF.rdata")
} else {
  cv <- trainControl(method = "repeatedcv", number = 10, savePredictions = TRUE, classProbs=TRUE)

# Treinando o modelo
  modelMkt_RF <- train(SUBSCRIBED~., data = dfTrain_Mkt, method = "rf",trControl = cv)
}

modelMkt_RF
```

```
## Random Forest
##
## 28832 samples
##    18 predictor
##     2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 25949, 25948, 25949, 25949, 25949, 25949, ...
## Resampling results across tuning parameters:
##
```

```
##   mtry  Accuracy   Kappa
##    2    0.8889429  0.02751189
##   46    0.9120422  0.52642179
##   91    0.9115565  0.52567461
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 46.
```

Usando a base de teste para verificação do modelo:

```
pred_Mkt_rf <- predict(modelMkt_RF ,newdata=dfTest_Mkt)
```

Gerando Matriz de Confusão:

```
cmMkt_RF <- confusionMatrix(data=pred_Mkt_rf, dfTest_Mkt$SUBSCRIBED)
cmMkt_RF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##        no  10514   630
##        yes   450   762
##
##                Accuracy : 0.9126
##                  95% CI : (0.9075, 0.9175)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5367
##
##  Mcnemar's Test P-Value : 5.129e-08
##
##             Sensitivity : 0.9590
##             Specificity : 0.5474
##          Pos Pred Value : 0.9435
##          Neg Pred Value : 0.6287
##              Prevalence : 0.8873
##          Detection Rate : 0.8509
##    Detection Prevalence : 0.9019
##       Balanced Accuracy : 0.7532
##
##        'Positive' Class : no
##
```

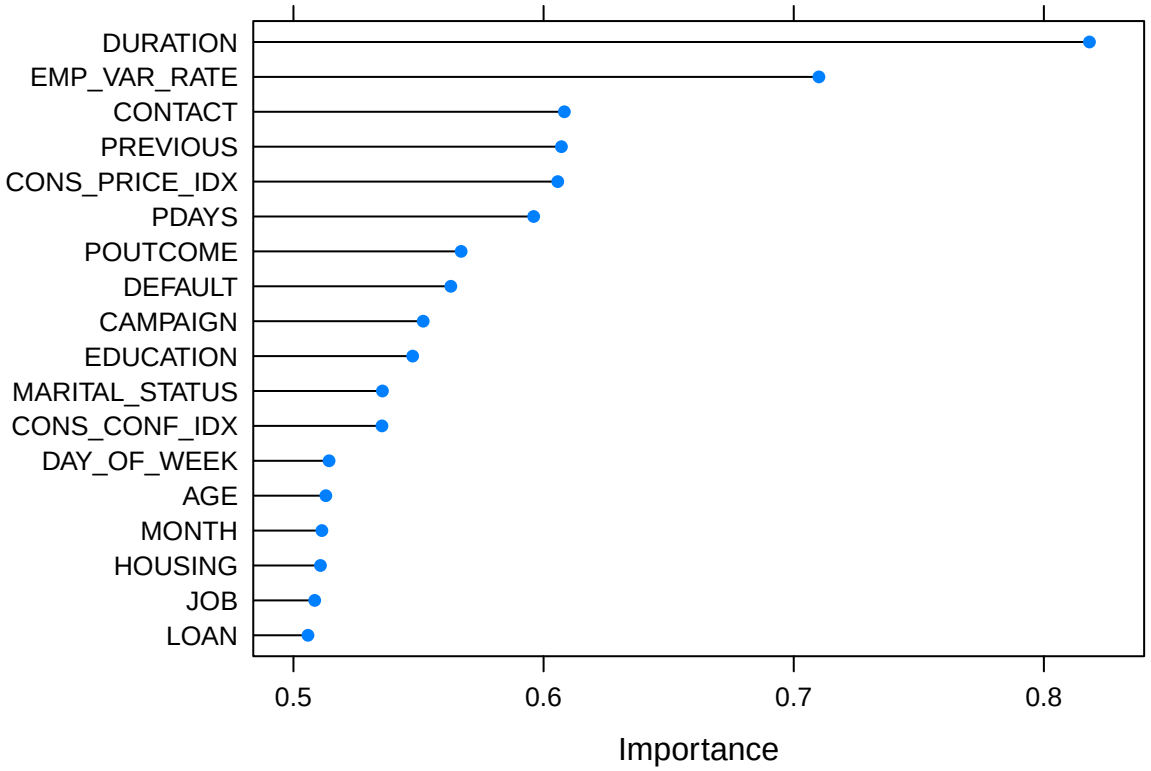
Importância das Variáveis Preditoras:

```
#
imp <- varImp(modelMkt_RF, useModel=FALSE, scale=FALSE)
imp
```

```
## ROC curve variable importance
##
##                Importance
## DURATION           0.8182
## EMP_VAR_RATE       0.7101
## CONTACT            0.6083
## PREVIOUS           0.6071
## CONS_PRICE_IDX     0.6057
## PDAYS              0.5960
## POUTCOME           0.5670
## DEFAULT            0.5629
```

```
## CAMPAIGN           0.5519
## EDUCATION          0.5477
## MARITAL_STATUS     0.5356
## CONS_CONF_IDX      0.5354
## DAY_OF_WEEK        0.5142
## AGE                0.5130
## MONTH              0.5114
## HOUSING            0.5108
## JOB                0.5085
## LOAN               0.5058
```

```
plot(imp)
```



Salvando o modelo:

```
save(modelMkt_RF,file="modelMkt_RF.rdata")
```

c) SVM (Support Vector Machines)

```
options(warn=-1)
set.seed(314)

if (file.exists("modelMkt_SVM.rdata")) {
  load("modelMkt_SVM.rdata")
} else {
  cv <- trainControl(method = "repeatedcv", number = 10)

  modelMkt_SVM <- train(SUBSCRIBED~., data = dfTrain_Mkt, method = "svmLinear", trControl = cv, preProcess =
}

modelMkt_SVM
```

```
## Support Vector Machines with Linear Kernel
```

```
##
## 28832 samples
##    18 predictor
##     2 classes: 'no', 'yes'
##
## Pre-processing: centered (91), scaled (91)
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 25949, 25948, 25949, 25949, 25949, 25949, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9044117  0.3793564
##
## Tuning parameter 'C' was held constant at a value of 1
```

Usando a base de teste para verificação do modelo:

```
dataMktPred <- predict(modelMkt_SVM, newdata=dfTest_Mkt)
```

Gerando Matriz de Confusão:

```
cmMkt_SVM <- confusionMatrix(data=dataMktPred, dfTest_Mkt$SUBSCRIBED)
cmMkt_SVM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##        no  10746   960
##        yes   218   432
##
##                Accuracy : 0.9047
##                  95% CI : (0.8993, 0.9098)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : 2.542e-10
##
##                   Kappa : 0.3785
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9801
##             Specificity : 0.3103
##          Pos Pred Value : 0.9180
##          Neg Pred Value : 0.6646
##              Prevalence : 0.8873
##          Detection Rate : 0.8697
##    Detection Prevalence : 0.9474
##       Balanced Accuracy : 0.6452
##
##        'Positive' Class : no
##
```

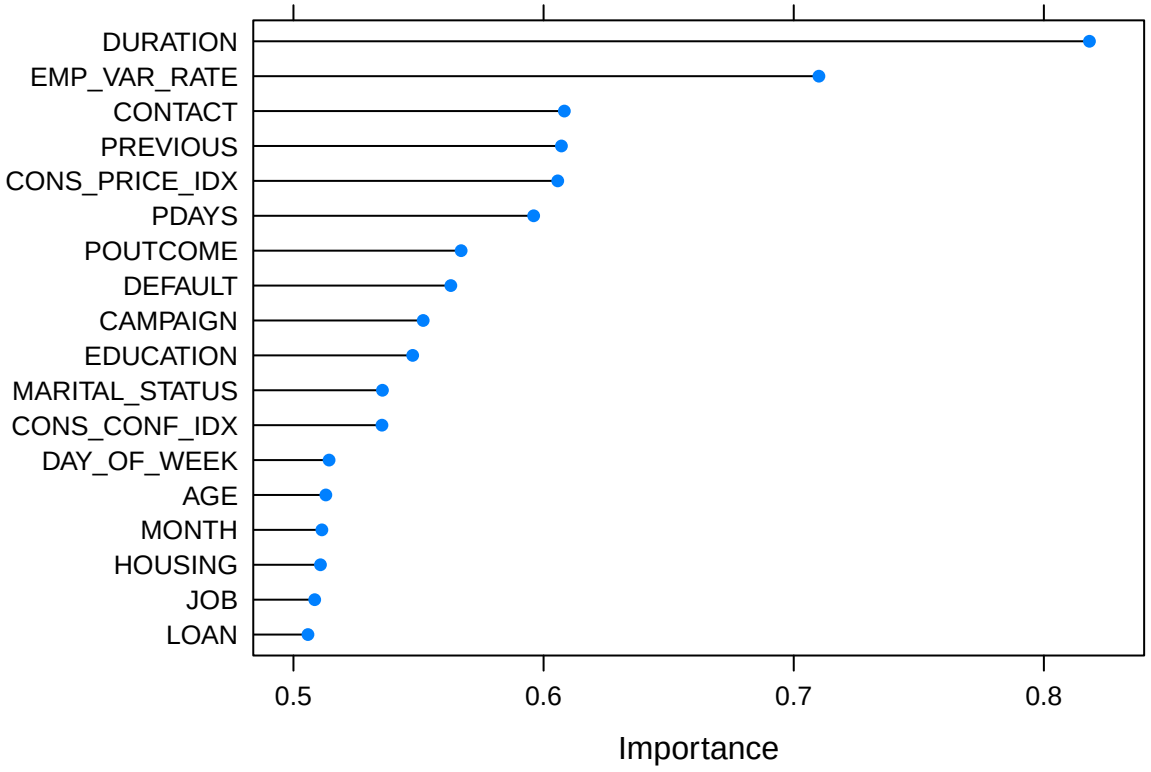
Importância das Variáveis Preditoras:

```
imp <- varImp(modelMkt_SVM, useModel=FALSE, scale=FALSE)
imp
```

```
## ROC curve variable importance
##
##                Importance
## DURATION           0.8182
## EMP_VAR_RATE       0.7101
```

```
## CONTACT            0.6083
## PREVIOUS           0.6071
## CONS_PRICE_IDX     0.6057
## PDAYS              0.5960
## POUTCOME           0.5670
## DEFAULT            0.5629
## CAMPAIGN           0.5519
## EDUCATION          0.5477
## MARITAL_STATUS     0.5356
## CONS_CONF_IDX      0.5354
## DAY_OF_WEEK        0.5142
## AGE                0.5130
## MONTH              0.5114
## HOUSING            0.5108
## JOB                0.5085
## LOAN               0.5058
```

```
plot(imp)
```



Salvando o modelo:

```
save(modelMkt_SVM,file="modelMkt_SVM.rdata")
```

d) Redes Neurais

```
options(warn=-1)

set.seed(314)

if (file.exists("modelMkt_RN.rdata")) {
  load("modelMkt_RN.rdata")
} else {
  modelMkt_RN <- train(SUBSCRIBED~., data = dfTrain_Mkt, method='nnet', trace = FALSE, preProc = c("center",
```

```
}

modelMkt_RN
```

```
## Neural Network
##
## 28832 samples
##    18 predictor
##     2 classes: 'no', 'yes'
##
## Pre-processing: centered (91), scaled (91)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 28832, 28832, 28832, 28832, 28832, 28832, ...
## Resampling results across tuning parameters:
##
##   size  decay  Accuracy   Kappa
##   1     0e+00  0.8927249  0.2264516
##   1     1e-04  0.8939668  0.2346662
##   1     1e-01  0.8928008  0.2527808
##   3     0e+00  0.8960573  0.4608805
##   3     1e-04  0.8951037  0.4607795
##   3     1e-01  0.9004783  0.5045524
##   5     0e+00  0.9027900  0.4786326
##   5     1e-04  0.9011746  0.4875981
##   5     1e-01  0.9024922  0.4936859
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were size = 5 and decay = 0.
```

Scorando o modelo base de teste:

```
#scorando o modelo base de teste
dataMktPred <- predict(modelMkt_RN, newdata=dfTest_Mkt)
```

Gerando Matriz de Confusão:

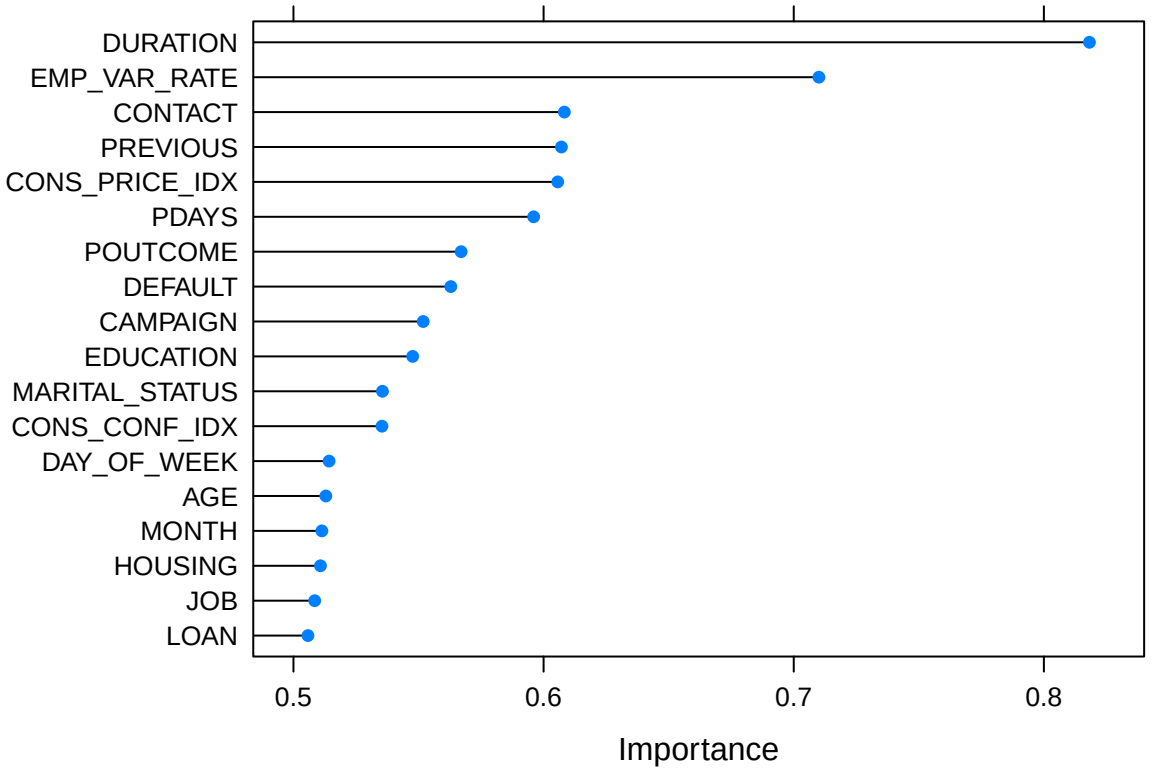
```
cmMkt_RN <- confusionMatrix(data=dataMktPred, dfTest_Mkt$SUBSCRIBED)
cmMkt_RN
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##        no  10423   563
##        yes   541   829
##
##                Accuracy : 0.9107
##                  95% CI : (0.9055, 0.9156)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.55
##
##  Mcnemar's Test P-Value : 0.5274
##
##             Sensitivity : 0.9507
##             Specificity : 0.5955
##          Pos Pred Value : 0.9488
##          Neg Pred Value : 0.6051
##              Prevalence : 0.8873
##          Detection Rate : 0.8436
```

```
##    Detection Prevalence : 0.8891
##       Balanced Accuracy : 0.7731
##
##        'Positive' Class : no
##
```

Importância das Variáveis Preditoras:

```
imp <- varImp(modelMkt_RN, useModel=FALSE, scale=FALSE)
plot(imp)
```



Salvando o modelo:

```
save(modelMkt_RN,file="modelMkt_RN.rdata")
```

Passo 5) Testar modelos de regressão para o valor de vendas das lojas

1) Regressão Linear

```
set.seed(314)

if (file.exists("modelRetail_RL.rdata")) {
  load("modelRetail_RL.rdata")
} else {

  #Desconsiderando a váriável DATE (não considerando como uma série temporal)
  modelRetail_RL <- lm(WEEKLY_SALES ~ . - DATE, data = dfTrain_Retail)
  k <- ols_step_backward_aic(modelRetail_RL)

  #Retirando as variáveis indicadas
  modelRetail_RL <- train(WEEKLY_SALES ~ STORE + FUEL_PRICE + MARKDOWN1 + MARKDOWN2 + CPI, data = dfTrain_Ret

}
```

```
modelRetail_RL
```

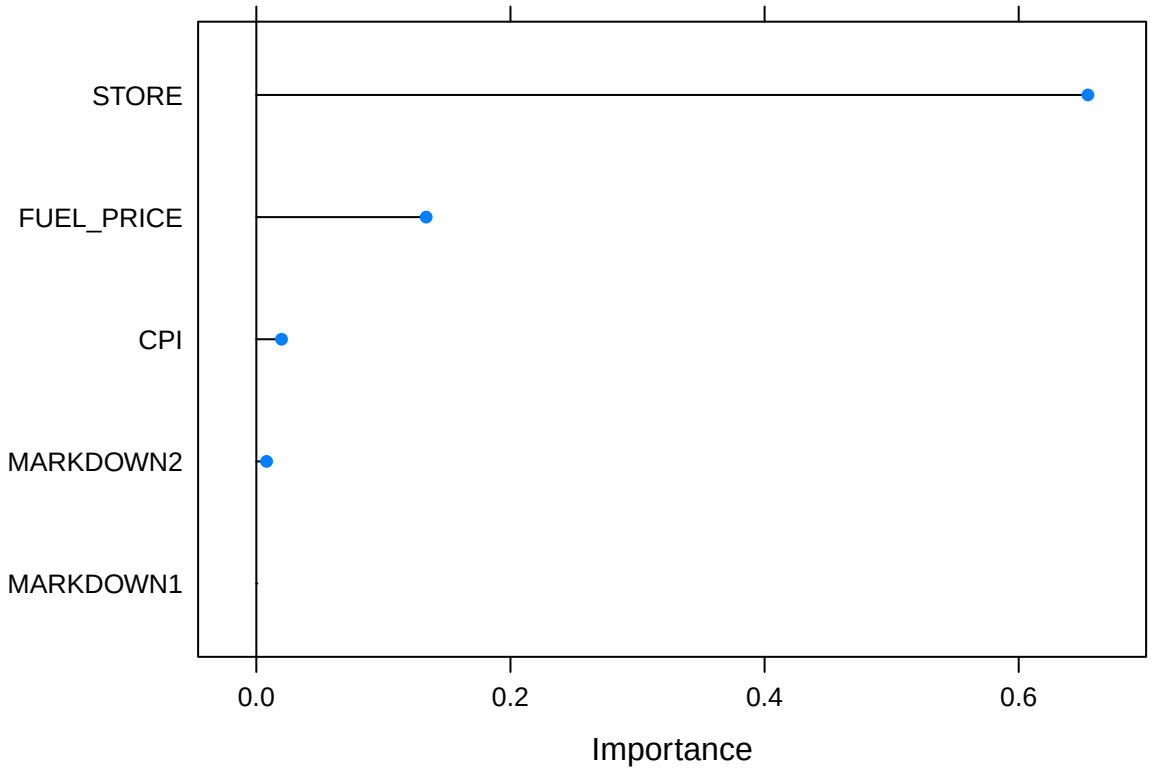
```
## Linear Regression
##
## 5734 samples
##    5 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 5734, 5734, 5734, 5734, 5734, 5734, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   10235.91  0.6451901  6231.525
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Importância das Variáveis Preditoras:

```
imp <- varImp(modelRetail_RL ,useModel=FALSE, scale=FALSE)
imp
```

```
## loess r-squared variable importance
##
##             Overall
## STORE      0.654476
## FUEL_PRICE 0.133666
## CPI        0.019835
## MARKDOWN2  0.008059
## MARKDOWN1  0.000000
```

```
plot(imp)
```



Avaliando o modelo com a base de teste:

```
predict_Model <- predict(modelRetail_RL, dfTest_Retail)

#Retornando as métricas do modelo com a base de teste
metric_modelRetail_RL <- postResample(pred = predict_Model, obs = dfTest_Retail$WEEKLY_SALES)
metric_modelRetail_RL
```

```
##         RMSE     Rsquared          MAE
## 1.101246e+04 6.071794e-01 6.452532e+03
```

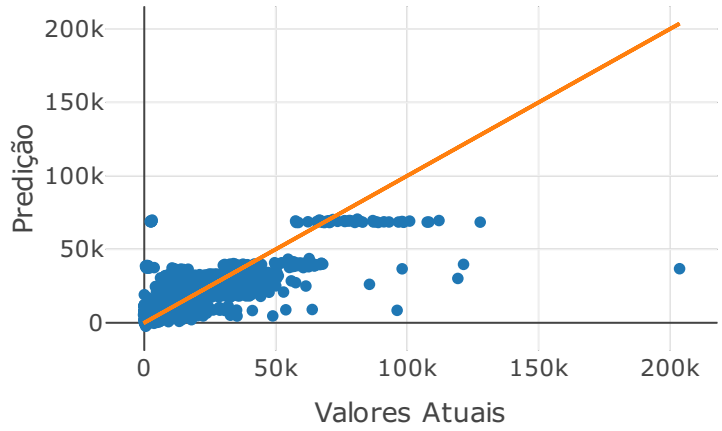
```
dfTest_Retail$Model <- predict_Model

head(dfTest_Retail)
```

```
##    STORE       DATE TEMPERATURE FUEL_PRICE MARKDOWN1 MARKDOWN2 MARKDOWN3
## 3      1 2010-02-19       39.93      2.514         0         0         0
## 5      1 2010-03-05       46.50      2.625         0         0         0
## 10     1 2010-04-09       65.86      2.770         0         0         0
## 12     1 2010-04-23       64.84      2.795         0         0         0
## 13     1 2010-04-30       67.41      2.780         0         0         0
## 17     1 2010-05-28       80.44      2.759         0         0         0
##    MARKDOWN4 MARKDOWN5    CPI UNEMPLOYMENT ISHOLIDAY WEEKLY_SALES    Model
## 3          0         0 211.28        8.106     FALSE     41595.55 27814.66
## 5          0         0 211.35        8.106     FALSE     21827.90 28011.13
## 10         0         0 210.62        7.808     FALSE     42960.91 28530.05
## 12         0         0 210.43        7.808     FALSE     16145.35 28639.99
## 13         0         0 210.38        7.808     FALSE     16555.11 28626.38
## 17         0         0 210.89        7.808     FALSE     15580.43 28422.16
```

```
df_diag <- data.frame(actual = dfTest_Retail$WEEKLY_SALES,
                      fitted = dfTest_Retail$Model)

df_diag %>% plot_ly(x = ~actual, y = ~fitted) %>%
               add_trace(x = ~actual, y = ~fitted, mode = 'markers',type = 'scatter') %>%
               add_trace(x = ~actual, y= ~actual , type="scatter", mode="lines", name='abline') %>%
               layout(title = "",
                  xaxis = list(title = "Valores Atuais"),
                  yaxis = list(title = "Predição"),showlegend = FALSE)
```



Salvando o modelo:

```
save(modelRetail_RL,file="modelRetail_RL.rdata")
```

2) Árvore de Decisão

Modelo com Boosting (XgBoost):

```
set.seed(314)

# Treinando o modelo
if (file.exists("modelRetail_Boosting.rdata")) {
  load("modelRetail_Boosting.rdata")
} else {
  cv <- trainControl(method = "repeatedcv", number = 10, savePredictions = TRUE)
  modelRetail_Boosting <- train(WEEKLY_SALES~. , data = dfTrain_Retail, method = "xgbTree",trControl = cv)
}

modelRetail_Boosting
```

```
## eXtreme Gradient Boosting
##
## 5734 samples
##   12 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 5161, 5160, 5161, 5161, 5162, 5161, ...
```

```
## Resampling results across tuning parameters:
##
##   eta  max_depth  colsample_bytree  subsample  nrounds  RMSE
##   0.3  1          0.6               0.50        50      11662.411
##   0.3  1          0.6               0.50       100      10742.303
##   0.3  1          0.6               0.50       150      10466.877
##   0.3  1          0.6               0.75        50      11749.834
##   0.3  1          0.6               0.75       100      10806.615
##   0.3  1          0.6               0.75       150      10465.747
##   0.3  1          0.6               1.00        50      11820.420
##   0.3  1          0.6               1.00       100      10871.345
##   0.3  1          0.6               1.00       150      10498.132
##   0.3  1          0.8               0.50        50      11665.381
##   0.3  1          0.8               0.50       100      10753.376
##   0.3  1          0.8               0.50       150      10509.213
##   0.3  1          0.8               0.75        50      11721.006
##   0.3  1          0.8               0.75       100      10817.135
##   0.3  1          0.8               0.75       150      10482.607
##   0.3  1          0.8               1.00        50      11799.393
##   0.3  1          0.8               1.00       100      10872.083
##   0.3  1          0.8               1.00       150      10500.019
##   0.3  2          0.6               0.50        50       8631.294
##   0.3  2          0.6               0.50       100       7562.384
##   0.3  2          0.6               0.50       150       7057.032
##   0.3  2          0.6               0.75        50       8511.241
##   0.3  2          0.6               0.75       100       7347.159
##   0.3  2          0.6               0.75       150       6838.627
##   0.3  2          0.6               1.00        50       8634.126
##   0.3  2          0.6               1.00       100       7542.668
##   0.3  2          0.6               1.00       150       6986.769
##   0.3  2          0.8               0.50        50       8493.321
##   0.3  2          0.8               0.50       100       7373.237
##   0.3  2          0.8               0.50       150       6965.094
##   0.3  2          0.8               0.75        50       8414.890
##   0.3  2          0.8               0.75       100       7250.529
##   0.3  2          0.8               0.75       150       6783.783
##   0.3  2          0.8               1.00        50       8699.336
##   0.3  2          0.8               1.00       100       7578.065
##   0.3  2          0.8               1.00       150       7045.071
##   0.3  3          0.6               0.50        50       7425.706
##   0.3  3          0.6               0.50       100       6594.415
##   0.3  3          0.6               0.50       150       6365.257
##   0.3  3          0.6               0.75        50       7250.625
##   0.3  3          0.6               0.75       100       6428.788
##   0.3  3          0.6               0.75       150       6104.467
##   0.3  3          0.6               1.00        50       7415.266
##   0.3  3          0.6               1.00       100       6445.550
##   0.3  3          0.6               1.00       150       6023.405
##   0.3  3          0.8               0.50        50       7202.892
##   0.3  3          0.8               0.50       100       6447.738
##   0.3  3          0.8               0.50       150       6221.864
##   0.3  3          0.8               0.75        50       7141.640
##   0.3  3          0.8               0.75       100       6329.738
##   0.3  3          0.8               0.75       150       6026.314
##   0.3  3          0.8               1.00        50       7370.844
##   0.3  3          0.8               1.00       100       6423.861
##   0.3  3          0.8               1.00       150       6044.624
##   0.4  1          0.6               0.50        50      11209.018
##   0.4  1          0.6               0.50       100      10560.247
```

```
##   0.4  1          0.6              0.50       150      10277.357
##   0.4  1          0.6              0.75        50      11256.787
##   0.4  1          0.6              0.75       100      10513.905
##   0.4  1          0.6              0.75       150      10246.552
##   0.4  1          0.6              1.00        50      11297.486
##   0.4  1          0.6              1.00       100      10557.033
##   0.4  1          0.6              1.00       150      10266.645
##   0.4  1          0.8              0.50        50      11214.004
##   0.4  1          0.8              0.50       100      10539.513
##   0.4  1          0.8              0.50       150      10274.658
##   0.4  1          0.8              0.75        50      11225.926
##   0.4  1          0.8              0.75       100      10508.114
##   0.4  1          0.8              0.75       150      10248.796
##   0.4  1          0.8              1.00        50      11310.561
##   0.4  1          0.8              1.00       100      10554.564
##   0.4  1          0.8              1.00       150      10271.405
##   0.4  2          0.6              0.50        50       8005.001
##   0.4  2          0.6              0.50       100       7176.823
##   0.4  2          0.6              0.50       150       6792.689
##   0.4  2          0.6              0.75        50       7901.577
##   0.4  2          0.6              0.75       100       6951.020
##   0.4  2          0.6              0.75       150       6554.077
##   0.4  2          0.6              1.00        50       8120.865
##   0.4  2          0.6              1.00       100       7067.054
##   0.4  2          0.6              1.00       150       6564.404
##   0.4  2          0.8              0.50        50       7858.207
##   0.4  2          0.8              0.50       100       7086.247
##   0.4  2          0.8              0.50       150       6793.976
##   0.4  2          0.8              0.75        50       7831.713
##   0.4  2          0.8              0.75       100       6961.156
##   0.4  2          0.8              0.75       150       6547.003
##   0.4  2          0.8              1.00        50       8130.830
##   0.4  2          0.8              1.00       100       7068.452
##   0.4  2          0.8              1.00       150       6517.398
##   0.4  3          0.6              0.50        50       7168.457
##   0.4  3          0.6              0.50       100       6677.008
##   0.4  3          0.6              0.50       150       6466.998
##   0.4  3          0.6              0.75        50       6871.088
##   0.4  3          0.6              0.75       100       6127.276
##   0.4  3          0.6              0.75       150       5987.825
##   0.4  3          0.6              1.00        50       6933.161
##   0.4  3          0.6              1.00       100       6281.309
##   0.4  3          0.6              1.00       150       6039.220
##   0.4  3          0.8              0.50        50       6925.145
##   0.4  3          0.8              0.50       100       6531.179
##   0.4  3          0.8              0.50       150       6377.005
##   0.4  3          0.8              0.75        50       6819.745
##   0.4  3          0.8              0.75       100       6276.431
##   0.4  3          0.8              0.75       150       6103.741
##   0.4  3          0.8              1.00        50       6803.539
##   0.4  3          0.8              1.00       100       6032.869
##   0.4  3          0.8              1.00       150       5795.786
##    Rsquared   MAE
##   0.5681204  8435.306
##   0.6153877  7260.491
##   0.6314056  6914.966
##   0.5608036  8505.169
##   0.6146110  7336.996
##   0.6328467  6874.274
```

```
##   0.5613950  8585.274
##   0.6175036  7450.088
##   0.6361115  6926.694
##   0.5623398  8386.770
##   0.6140122  7256.878
##   0.6281935  6925.069
##   0.5626942  8468.054
##   0.6130347  7337.212
##   0.6314739  6882.981
##   0.5636205  8576.012
##   0.6172027  7446.458
##   0.6359002  6930.245
##   0.7653384  5937.121
##   0.8081193  4733.359
##   0.8301383  4242.593
##   0.7788495  5908.406
##   0.8227099  4623.770
##   0.8427745  4100.423
##   0.7733879  6003.623
##   0.8140119  4740.363
##   0.8367401  4130.000
##   0.7724133  5861.993
##   0.8185336  4605.367
##   0.8357045  4152.880
##   0.7810008  5823.324
##   0.8257596  4507.531
##   0.8438374  3980.301
##   0.7690911  6034.333
##   0.8130356  4755.166
##   0.8336454  4154.896
##   0.8191266  4752.266
##   0.8518897  3785.473
##   0.8604485  3483.714
##   0.8297669  4675.328
##   0.8589442  3676.332
##   0.8710552  3320.261
##   0.8243452  4773.381
##   0.8593697  3675.013
##   0.8754989  3245.592
##   0.8290409  4601.607
##   0.8574406  3660.709
##   0.8669324  3380.130
##   0.8345850  4542.593
##   0.8623223  3518.260
##   0.8735058  3165.865
##   0.8264832  4714.685
##   0.8602889  3625.625
##   0.8748392  3204.059
##   0.5898864  7859.730
##   0.6245830  6932.667
##   0.6425375  6706.036
##   0.5902577  7920.970
##   0.6297804  6907.662
##   0.6460039  6628.258
##   0.5932652  7964.947
##   0.6318381  6978.920
##   0.6477330  6629.498
##   0.5886459  7866.232
##   0.6260264  6955.508
```

```
##   0.6428860  6708.534
##   0.5926392  7889.718
##   0.6303833  6907.653
##   0.6457718  6621.155
##   0.5946902  7964.536
##   0.6322856  6981.239
##   0.6473482  6628.762
##   0.7900322  5265.431
##   0.8248232  4314.592
##   0.8415474  4011.838
##   0.7997974  5195.608
##   0.8370289  4132.704
##   0.8533124  3772.196
##   0.7900994  5392.195
##   0.8322929  4249.043
##   0.8525280  3756.480
##   0.7983744  5114.539
##   0.8297508  4251.932
##   0.8419766  3987.718
##   0.8023179  5120.750
##   0.8359813  4103.637
##   0.8532750  3726.349
##   0.7904514  5340.858
##   0.8344101  4181.217
##   0.8563388  3676.658
##   0.8275724  4248.858
##   0.8481292  3655.571
##   0.8571144  3489.690
##   0.8414377  4149.168
##   0.8714550  3354.979
##   0.8765669  3172.849
##   0.8400611  4185.353
##   0.8638803  3323.249
##   0.8735289  3070.367
##   0.8374398  4193.096
##   0.8533337  3601.150
##   0.8609798  3441.486
##   0.8446791  4029.108
##   0.8640801  3336.729
##   0.8709708  3136.387
##   0.8478564  4108.457
##   0.8748736  3214.947
##   0.8834655  2969.321
##
## Tuning parameter 'gamma' was held constant at a value of 0
##
## Tuning parameter 'min_child_weight' was held constant at a value of 1
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were nrounds = 150, max_depth = 3,
##  eta = 0.4, gamma = 0, colsample_bytree = 0.8, min_child_weight = 1
##  and subsample = 1.
```

Avaliando o modelo com a base de teste:

```
pred_boosting <- predict(modelRetail_Boosting ,newdata=dfTest_Retail)

#Retornando as métricas do modelo com a base de teste
metric_modelRetail_Boosting <- postResample(pred = pred_boosting, obs = dfTest_Retail$WEEKLY_SALES)
metric_modelRetail_Boosting
```

```
##         RMSE     Rsquared          MAE
## 6561.0698257    0.8615889 2963.6912254
```

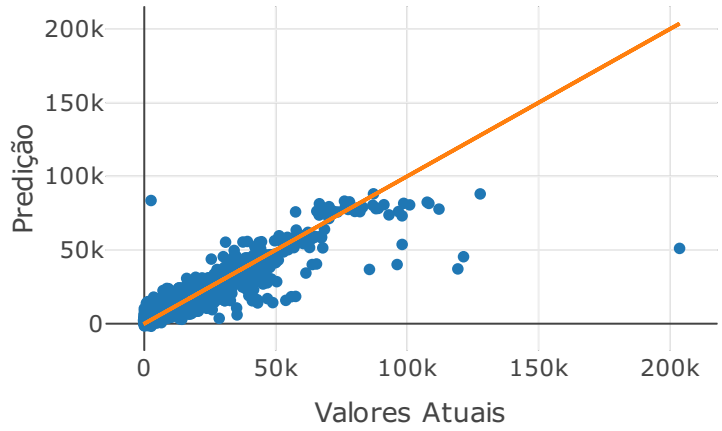
```
dfTest_Retail$Model <- pred_boosting

head(dfTest_Retail)
```

```
##    STORE       DATE TEMPERATURE FUEL_PRICE MARKDOWN1 MARKDOWN2 MARKDOWN3
## 3      1 2010-02-19       39.93      2.514         0         0         0
## 5      1 2010-03-05       46.50      2.625         0         0         0
## 10     1 2010-04-09       65.86      2.770         0         0         0
## 12     1 2010-04-23       64.84      2.795         0         0         0
## 13     1 2010-04-30       67.41      2.780         0         0         0
## 17     1 2010-05-28       80.44      2.759         0         0         0
##    MARKDOWN4 MARKDOWN5    CPI UNEMPLOYMENT ISHOLIDAY WEEKLY_SALES    Model
## 3          0         0 211.28        8.106     FALSE     41595.55 22965.67
## 5          0         0 211.35        8.106     FALSE     21827.90 20877.67
## 10         0         0 210.62        7.808     FALSE     42960.91 16258.98
## 12         0         0 210.43        7.808     FALSE     16145.35 15983.64
## 13         0         0 210.38        7.808     FALSE     16555.11 15874.88
## 17         0         0 210.89        7.808     FALSE     15580.43 19329.27
```

```
df_diag <- data.frame(actual = dfTest_Retail$WEEKLY_SALES,
                      fitted = dfTest_Retail$Model)

df_diag %>% plot_ly(x = ~actual, y = ~fitted) %>%
               add_trace(x = ~actual, y = ~fitted, mode = 'markers',type = 'scatter') %>%
               add_trace(x = ~actual, y= ~actual , type="scatter", mode="lines", name='abline') %>%
               layout(title = "",
                  xaxis = list(title = "Valores Atuais"),
                  yaxis = list(title = "Predição"),showlegend = FALSE)
```

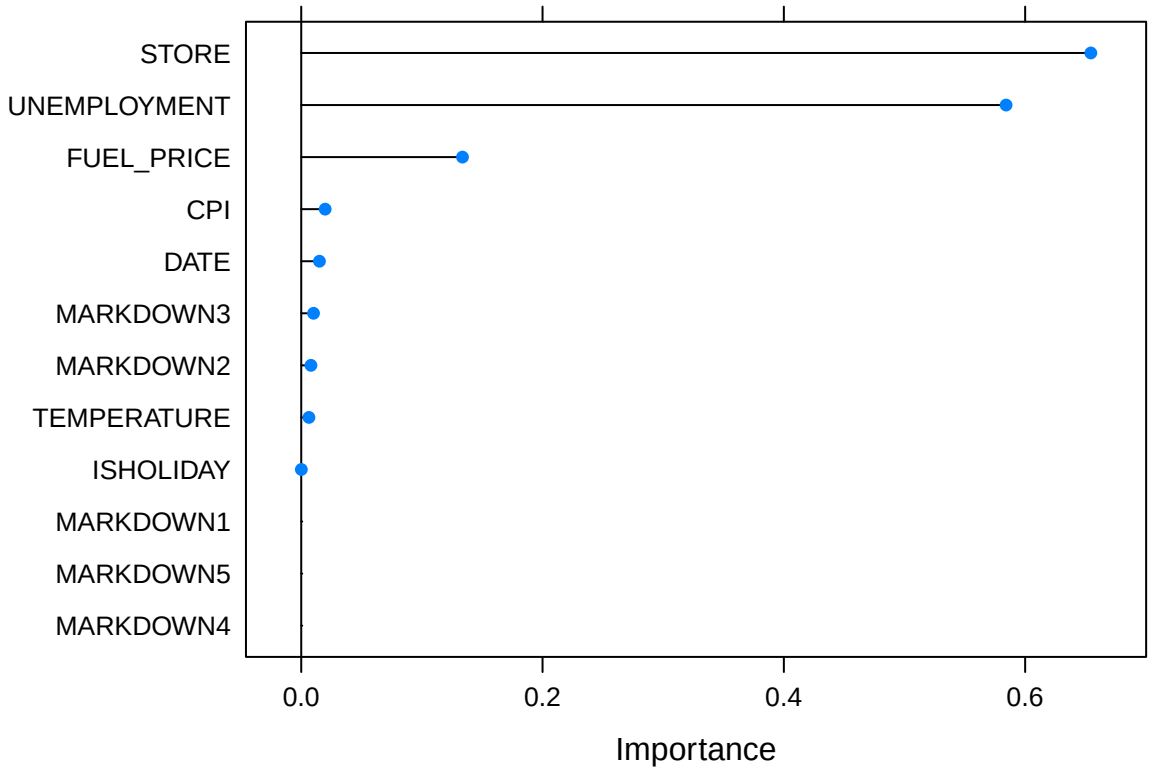


Importância das Variáveis Preditoras:

```
imp <- varImp(modelRetail_Boosting, useModel=FALSE, scale=FALSE)
imp
```

```
## loess r-squared variable importance
##
##                Overall
## STORE        6.545e-01
## UNEMPLOYMENT 5.842e-01
## FUEL_PRICE   1.337e-01
## CPI          1.984e-02
## DATE         1.510e-02
## MARKDOWN3    1.019e-02
## MARKDOWN2    8.059e-03
## TEMPERATURE  6.386e-03
## ISHOLIDAY    9.245e-05
## MARKDOWN4    0.000e+00
## MARKDOWN1    0.000e+00
## MARKDOWN5    0.000e+00
```

```
plot(imp)
```



Salvando o modelo:

```
save(modelRetail_Boosting,file="modelRetail_Boosting.rdata")
```

Modelo Random Forest:

```
#Treinamento o modelo
set.seed(314)

if (file.exists("modelRetail_RF.rdata")) {
  load("modelRetail_RF.rdata")
} else {
  modelRetail_RF <- train(WEEKLY_SALES~., data = dfTrain_Retail, method = "rf", trControl = cv)
}

modelRetail_RF
```

```
## Random Forest
##
## 5734 samples
##   12 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 5161, 5160, 5161, 5161, 5162, 5161, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
##    2    12621.092  0.6832040  9168.274
##   28     5192.653  0.9052310  2393.189
##   55     5276.347  0.9023141  2374.394
##
## RMSE was used to select the optimal model using the smallest value.
```

```
## The final value used for the model was mtry = 28.
```

Avaliando o modelo com a base de teste:

```
pred_RF <- predict(modelRetail_RF ,newdata=dfTest_Retail)

#Retornando as métricas do modelo com a base de teste
metric_modelRetail_RF <- postResample(pred = pred_RF, obs = dfTest_Retail$WEEKLY_SALES)
metric_modelRetail_RF
```

```
##         RMSE     Rsquared          MAE
## 5715.2742158    0.8950248 2328.7048207
```

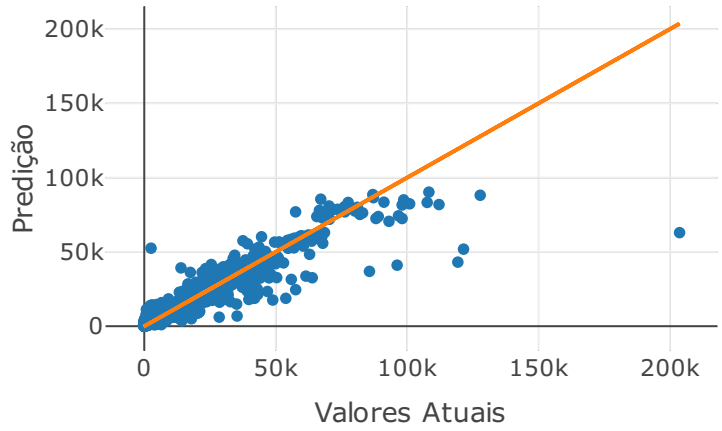
```
dfTest_Retail$Model <- pred_RF

head(dfTest_Retail)
```

```
##    STORE       DATE TEMPERATURE FUEL_PRICE MARKDOWN1 MARKDOWN2 MARKDOWN3
## 3      1 2010-02-19       39.93      2.514         0         0         0
## 5      1 2010-03-05       46.50      2.625         0         0         0
## 10     1 2010-04-09       65.86      2.770         0         0         0
## 12     1 2010-04-23       64.84      2.795         0         0         0
## 13     1 2010-04-30       67.41      2.780         0         0         0
## 17     1 2010-05-28       80.44      2.759         0         0         0
##    MARKDOWN4 MARKDOWN5    CPI UNEMPLOYMENT ISHOLIDAY WEEKLY_SALES    Model
## 3          0         0 211.28        8.106     FALSE     41595.55 28024.57
## 5          0         0 211.35        8.106     FALSE     21827.90 21856.50
## 10         0         0 210.62        7.808     FALSE     42960.91 25166.08
## 12         0         0 210.43        7.808     FALSE     16145.35 17772.59
## 13         0         0 210.38        7.808     FALSE     16555.11 16988.83
## 17         0         0 210.89        7.808     FALSE     15580.43 19618.71
```

```
df_diag <- data.frame(actual = dfTest_Retail$WEEKLY_SALES,
                      fitted = dfTest_Retail$Model)

df_diag %>% plot_ly(x = ~actual, y = ~fitted) %>%
               add_trace(x = ~actual, y = ~fitted, mode = 'markers',type = 'scatter') %>%
               add_trace(x = ~actual, y= ~actual , type="scatter", mode="lines", name='abline') %>%
               layout(title = "",
                  xaxis = list(title = "Valores Atuais"),
                  yaxis = list(title = "Predição"),showlegend = FALSE)
```

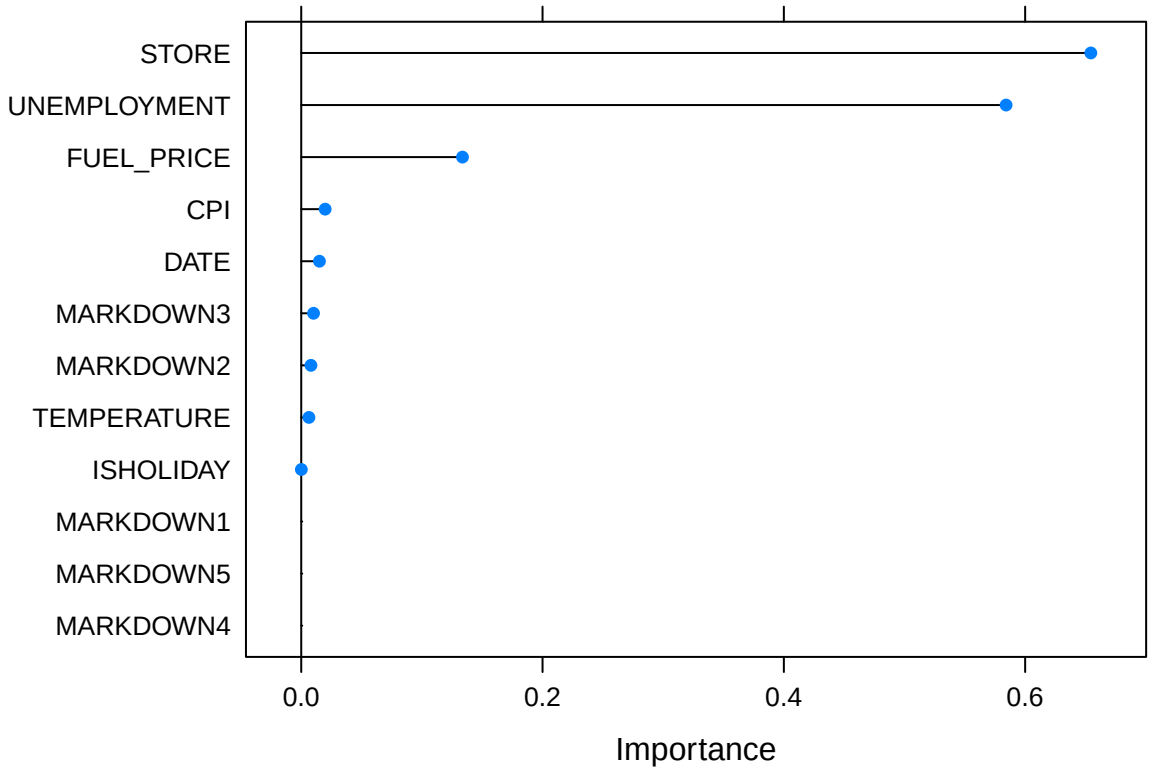


Importância das Variáveis Preditoras:

```
imp <- varImp(modelRetail_RF, useModel=FALSE, scale=FALSE)
imp
```

```
## loess r-squared variable importance
##
##                Overall
## STORE        6.545e-01
## UNEMPLOYMENT 5.842e-01
## FUEL_PRICE   1.337e-01
## CPI          1.984e-02
## DATE         1.510e-02
## MARKDOWN3    1.019e-02
## MARKDOWN2    8.059e-03
## TEMPERATURE  6.386e-03
## ISHOLIDAY    9.245e-05
## MARKDOWN4    0.000e+00
## MARKDOWN1    0.000e+00
## MARKDOWN5    0.000e+00
```

```
plot(imp)
```



Salvando o modelo:

```
save(modelRetail_RF,file="modelRetail_RF.rdata")
```

3) Redes Neurais

```
options(warn=-1)
set.seed(314)

if (file.exists("modelRetail_NNET.rdata")) {
  load("modelRetail_NNET.rdata")
} else {
  # Definindo Parâmetros do Cross Validation
  cv <- trainControl(method = "repeatedcv", number = 10, savePredictions = TRUE)
  modelRetail_NNET <- train(WEEKLY_SALES~ ., data = dfTrain_Retail, method = "nnet",trControl = cv, maxit=100
}

modelRetail_NNET
```

```
## Neural Network
##
## 5734 samples
##   12 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 5161, 5160, 5161, 5161, 5162, 5161, ...
## Resampling results across tuning parameters:
##
##   size  decay  RMSE      Rsquared      MAE
##   1     0e+00  17184.32  0.0016126902  12761.47
##   1     1e-04  17202.13  0.0002116679  12765.10
##   1     1e-01  17180.41  0.0029806597  12759.33
```

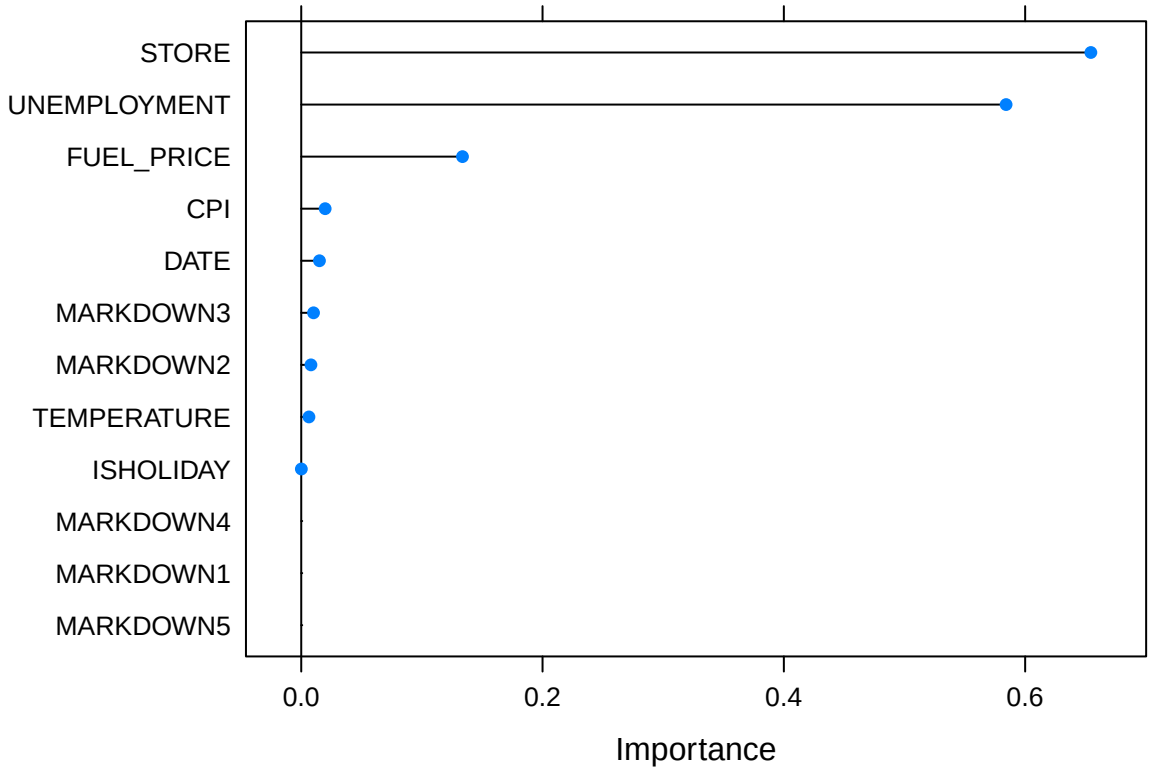
```
##   3     0e+00  17062.30  0.0217330495  12681.37
##   3     1e-04  17164.74  0.0039172021  12755.99
##   3     1e-01  16745.99  0.0575299420  12347.29
##   5     0e+00  17145.61  0.0054903414  12723.18
##   5     1e-04  17110.55  0.0146046906  12724.12
##   5     1e-01  16885.18  0.0531855456  12425.33
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were size = 3 and decay = 0.1.
```

Importância das Variáveis Preditoras:

```
imp <- varImp(modelRetail_NNET, useModel=FALSE, scale=FALSE)
imp
```

```
## loess r-squared variable importance
##
##                Overall
## STORE        6.545e-01
## UNEMPLOYMENT 5.842e-01
## FUEL_PRICE   1.337e-01
## CPI          1.984e-02
## DATE         1.510e-02
## MARKDOWN3    1.019e-02
## MARKDOWN2    8.059e-03
## TEMPERATURE  6.386e-03
## ISHOLIDAY    9.245e-05
## MARKDOWN1    0.000e+00
## MARKDOWN4    0.000e+00
## MARKDOWN5    0.000e+00
```

```
plot(imp)
```



Avaliando o modelo com a base de teste:

```
pred_NNET <- predict(modelRetail_NNET ,newdata=dfTest_Retail)

#Retornando as métricas do modelo com a base de teste
metric_modelRetail_NNET <- postResample(pred = pred_NNET, obs = dfTest_Retail$WEEKLY_SALES)
metric_modelRetail_NNET
```

```
##         RMSE     Rsquared          MAE
## 1.271506e+04 4.768282e-01 7.614559e+03
```

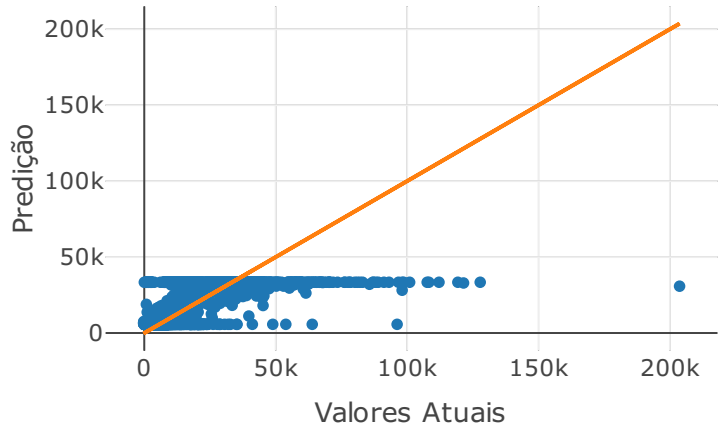
```
dfTest_Retail$Model <- pred_NNET

head(dfTest_Retail)
```

```
##    STORE       DATE TEMPERATURE FUEL_PRICE MARKDOWN1 MARKDOWN2 MARKDOWN3
## 3      1 2010-02-19       39.93      2.514         0         0         0
## 5      1 2010-03-05       46.50      2.625         0         0         0
## 10     1 2010-04-09       65.86      2.770         0         0         0
## 12     1 2010-04-23       64.84      2.795         0         0         0
## 13     1 2010-04-30       67.41      2.780         0         0         0
## 17     1 2010-05-28       80.44      2.759         0         0         0
##    MARKDOWN4 MARKDOWN5    CPI UNEMPLOYMENT ISHOLIDAY WEEKLY_SALES    Model
## 3          0         0 211.28        8.106     FALSE     41595.55 33420.35
## 5          0         0 211.35        8.106     FALSE     21827.90 33420.35
## 10         0         0 210.62        7.808     FALSE     42960.91 33420.35
## 12         0         0 210.43        7.808     FALSE     16145.35 33420.35
## 13         0         0 210.38        7.808     FALSE     16555.11 33420.35
## 17         0         0 210.89        7.808     FALSE     15580.43 33420.35
```

```
df_diag <- data.frame(actual = dfTest_Retail$WEEKLY_SALES,
                      fitted = dfTest_Retail$Model)

df_diag %>% plot_ly(x = ~actual, y = ~fitted) %>%
               add_trace(x = ~actual, y = ~fitted, mode = 'markers',type = 'scatter') %>%
               add_trace(x = ~actual, y= ~actual , type="scatter", mode="lines", name='abline') %>%
               layout(title = "",
                  xaxis = list(title = "Valores Atuais"),
                  yaxis = list(title = "Predição"),showlegend = FALSE)
```



Salvando o modelo:

```
save(modelRetail_NNET,file="modelRetail_NNET.rdata")
```

Redes Neurais usando o pacote h2o:

```
library(h2o)
h2o.init()
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         4 hours 7 minutes
##     H2O cluster timezone:       America/Sao_Paulo
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.22.1.1
##     H2O cluster version age:    5 months and 29 days !!!
##     H2O cluster name:           H2O_started_from_R_CTVIDAL_gjo731
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   2.02 GB
##     H2O cluster total cores:    4
##     H2O cluster allowed cores:  4
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
```

```
##     H2O API Extensions:         Algos, AutoML, Core V3, Core V4
##     R Version:                  R version 3.5.0 (2018-04-23)
```

```
if (file.exists("modelRetail_H2O.rdata")) {
  load("modelRetail_H2O.rdata")
} else {

  y <- "WEEKLY_SALES"
  x <- setdiff(names(dfTrain_Retail), y)

  #Criando uma grid de execução para varrer diversos modelos
  hyper_params <- list(
    activation=c("Rectifier","Tanh","Maxout","RectifierWithDropout","TanhWithDropout","MaxoutWithDropout"),
    hidden=list(c(25,25,25,25),c(50,50,25,10),c(50,50,25,10,5),c(50,50,50,25,10)),
    input_dropout_ratio=c(0,0.05,0.2),
    l1=seq(0,1e-4,1e-6),
    l2=seq(0,1e-4,1e-6)
  )

  #Definindo parâmetros e critérios de parada
  search_criteria = list(strategy = "RandomDiscrete", max_models = 2000, seed=341,max_runtime_secs = 500, sto
  dl_random_grid <- h2o.grid(
    algorithm="deeplearning",
    training_frame=as.h2o(dfTrain_Retail),
    x=x,
    y=y,
    epochs=1000,
    nfolds = 10,
    stopping_metric="AUTO",
    stopping_tolerance=1e-2,
    hyper_params = hyper_params,
    search_criteria = search_criteria
  )
  grid <- h2o.getGrid(dl_random_grid@grid_id,sort_by="r2",decreasing=TRUE)
  grid

  grid@summary_table[1,]
  modelRetail_H2O <- h2o.getModel(grid@model_ids[[1]]) ## Modelo com o maior R2

}

modelRetail_H2O
```

```
## Model Details:
## ==============
##
## H2ORegressionModel: deeplearning
## Model ID:  Grid_DeepLearning_dfTrain_Retail_sid_b7e0_25_model_R_1561605970983_2_model_1
## Status of Neuron Layers: predicting WEEKLY_SALES, regression, gaussian distribution, Quadratic loss, 14.18
##   layer units   type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    59  Input  5.00 %       NA       NA        NA       NA       NA
## 2     2    50 Maxout  0.00 % 0.000048 0.000021  0.044778 0.190949 0.000000
## 3     3    50 Maxout  0.00 % 0.000048 0.000021  0.012433 0.024537 0.000000
## 4     4    25 Maxout  0.00 % 0.000048 0.000021  0.022190 0.050351 0.000000
## 5     5    10 Maxout  0.00 % 0.000048 0.000021  0.098995 0.236253 0.000000
## 6     6     1 Linear      NA 0.000048 0.000021  0.024874 0.055597 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1          NA         NA        NA       NA
## 2   -0.004615   0.189334  0.103453 0.192756
## 3   -0.032263   0.135813  0.668062 0.177408
```

```
## 4   -0.025486   0.150788  0.702122 0.247921
## 5   -0.022075   0.215274  0.646757 0.431579
## 6    0.043850   0.306536  0.295645 0.000000
##
##
## H2ORegressionMetrics: deeplearning
## ** Reported on training data. **
## ** Metrics reported on full training frame **
##
## MSE:  81624237
## RMSE:  9034.613
## MAE:  4240.384
## RMSLE:  NaN
## Mean Residual Deviance :  81624237
##
##
##
## H2ORegressionMetrics: deeplearning
## ** Reported on cross-validation data. **
## ** 10-fold cross-validation on training data (Metrics computed for combined holdout predictions) **
##
## MSE:  86617054
## RMSE:  9306.828
## MAE:  5244.8
## RMSLE:  NaN
## Mean Residual Deviance :  86617054
##
##
## Cross-Validation Metrics Summary:
##                              mean          sd cv_1_valid cv_2_valid
## mae                     5171.5596   368.50864   4991.889  5722.7583
## mean_residual_deviance 8.7106752E7 1.0491326E7 7.619316E7 8.510884E7
## mse                    8.7106752E7 1.0491326E7 7.619316E7 8.510884E7
## r2                      0.70553803 0.021313956  0.7159598  0.6737612
## residual_deviance      8.7106752E7 1.0491326E7 7.619316E7 8.510884E7
## rmse                      9298.529   567.49603   8728.869   9225.445
## rmsle                    1.2980915 0.053460784        NaN  1.3544441
##                         cv_3_valid  cv_4_valid  cv_5_valid  cv_6_valid
## mae                       4769.766     5118.42    4701.216   6254.9653
## mean_residual_deviance 9.1487736E7 6.9342528E7 9.8843152E7 1.0377208E8
## mse                    9.1487736E7 6.9342528E7 9.8843152E7 1.0377208E8
## r2                       0.7045732   0.7559948  0.69668424   0.6692413
## residual_deviance      9.1487736E7 6.9342528E7 9.8843152E7 1.0377208E8
## rmse                      9564.922    8327.216    9941.989   10186.858
## rmsle                          NaN         NaN         NaN         NaN
##                          cv_7_valid   cv_8_valid cv_9_valid cv_10_valid
## mae                        4745.786     5459.525  4485.9214   5465.3477
## mean_residual_deviance 1.07610496E8 1.01392928E8 6.527936E7 7.2037256E7
## mse                    1.07610496E8 1.01392928E8 6.527936E7 7.2037256E7
## r2                        0.6630486   0.72604626  0.7478217   0.7022492
## residual_deviance      1.07610496E8 1.01392928E8 6.527936E7 7.2037256E7
## rmse                      10373.548    10069.405  8079.5645    8487.477
## rmsle                           NaN    1.2417389        NaN         NaN
```

Avaliando o modelo:

```
h2o.r2(modelRetail_H20)
```

```
## [1] 0.7245832
```

```
dfTest_Retail <- dfTest_Retail[,which(colnames(dfTest_Retail)!="Model")]

perf <- h2o.performance(modelRetail_H20,as.h2o(dfTest_Retail))
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
h2o.hit_ratio_table(perf)
```

```
## NULL
```

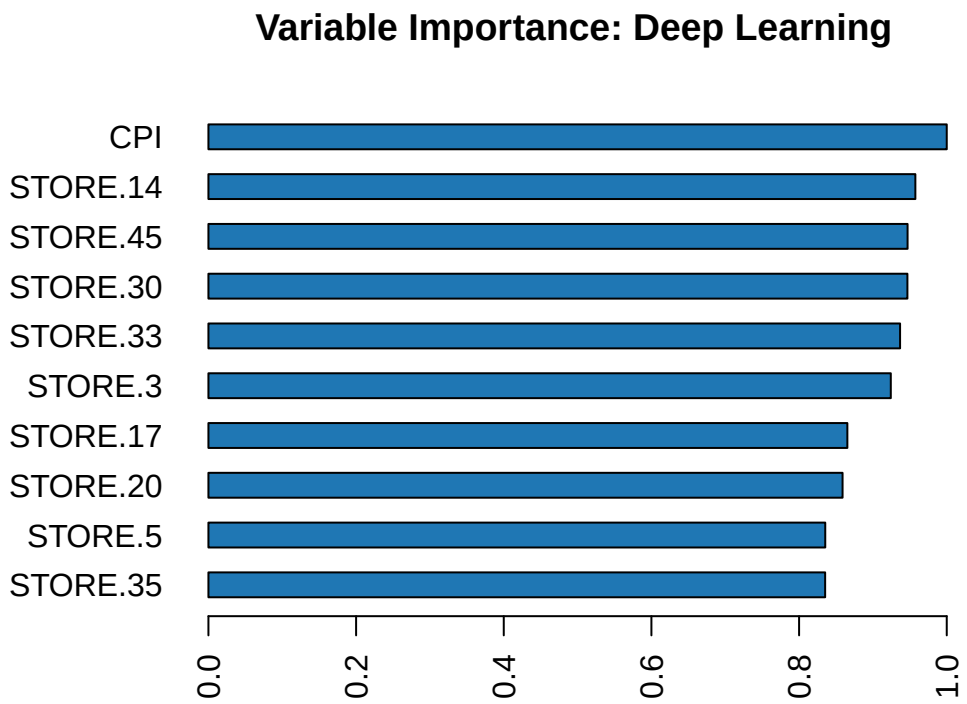
```
#Retornando as métricas do modelo com a base de teste
metric_modelRetail_H20 <- h2o.r2(perf)
metric_modelRetail_H20
```

```
## [1] 0.6426361
```

```
head(h2o.varimp(modelRetail_H20))
```

```
## Variable Importances:
##   variable relative_importance scaled_importance percentage
## 1      CPI            1.000000          1.000000   0.023931
## 2 STORE.14            0.957498          0.957498   0.022914
## 3 STORE.45            0.946941          0.946941   0.022661
## 4 STORE.30            0.946753          0.946753   0.022657
## 5 STORE.33            0.936792          0.936792   0.022418
## 6  STORE.3            0.924226          0.924226   0.022117
```

```
h2o.varimp_plot(modelRetail_H20)
```



Salvando o modelo:

```
save(modelRetail_H20,file="modelRetail_H20.rdata")
```

Passo 6) Avaliação final da performance dos modelos

1) Modelos da base de Marketing

```
comparativo_Mkt <- data.frame(GLM=cmMkt_GLM$overall['Accuracy'],RANDOM_FOREST=cmMkt_RF$overall['Accuracy'],
                              SVM=cmMkt_SVM$overall['Accuracy'],REDES_NEURAIS=cmMkt_RN$overall['Accuracy'])

comparativo_Mkt
```

```
##                GLM RANDOM_FOREST       SVM REDES_NEURAIS
## Accuracy 0.912674     0.9125931 0.9046617     0.9106507
```

2) Modelos da base de vendas

```
comparativo_RT_Train <- data.frame(REGRESSAO_LOG=max(modelRetail_RL$results$Rsquared),TREE_BOOSTING=max(model

comparativo_RT_Test <- data.frame(REGRESSAO_LOG=metric_modelRetail_RL[2],TREE_BOOSTING=metric_modelRetail_Boo

comparativo_RT <- rbind("R2 TRAIN"=comparativo_RT_Train,"R2 TEST"=comparativo_RT_Test)

comparativo_RT
```

```
##          REGRESSAO_LOG TREE_BOOSTING RANDOM_FOREST REDES_NEURAIS_NNET
## R2 TRAIN     0.6451901     0.8834655     0.9052310         0.05752994
## R2 TEST      0.6071794     0.8615889     0.8950248         0.47682820
##          REDES_NEURAIS_H20
## R2 TRAIN         0.7245832
## R2 TEST          0.6426361
```

Passo 7) Conclusão

Para a base de Marketing verifica-se que o modelo que apresenta maior acurácia foi o GLM (Accuracy=0.912674), enquanto que para a base de vendas o modelo que apresentou a melhor perfomance na base de testes foi a Random Forest (R2=0.8950248).