Projeto09.R

rodrigolima82

2019-09-05

```
# 01 - Iniciando o script de Machine Learning
```

```
# Carregando os Pacotes
library(rhdf5)
```

```
# Carregando o Dataset
data <- h5read('data/train.h5','train')
colnames = data[['axis0']]
data = cbind(t(data[['block0_values']]),t(data[['block1_values']]))
df = as.data.frame(data)
colnames(df) = colnames
rm(colnames)
rm(data)

# Visualizando os dados
head(df)
```

```
##   id timestamp   derived_0    derived_1   derived_2   derived_3
## 1 10         0  0.37032622 -0.006316399  0.22283109 -0.21303013
## 2 11         0  0.01476468 -0.038064219 -0.01742488  0.32065201
## 3 12         0 -0.01062180 -0.050577067  3.37957478 -0.15752506
## 4 25         0         NaN          NaN         NaN         NaN
## 5 26         0  0.17669280 -0.025284184 -0.05767995  0.01509975
## 6 27         0  0.34685576  0.166239306 -6.08070087 -0.99224919
##     derived_4 fundamental_0 fundamental_1 fundamental_2 fundamental_3
## 1  0.72927678  -0.335633248     0.1132921   1.621238232   -0.17940393
## 2 -0.03413433   0.004412613     0.1142851  -0.210184768    0.21628062
## 3 -0.06855001  -0.155936614     1.2194387  -0.764515758           NaN
## 4         NaN   0.178494513           NaN  -0.007262451   -0.09790272
## 5  0.18089357   0.139444515    -0.1256869  -0.018707044    0.19639103
## 6 -0.12591551   0.345812112           NaN  -0.584239423           NaN
##   fundamental_5 fundamental_6 fundamental_7 fundamental_8 fundamental_9
## 1           NaN   -0.07210785    0.24918664    0.02440143   -0.12794249
## 2    0.09674974    0.08204235   -0.22438331   -0.08552912    0.02476283
## 3           NaN   -0.05141819   -0.25832987   -0.12213951   -0.12104109
## 4           NaN           NaN   -0.09367702   -0.02695086           NaN
## 5           NaN   -0.16387995   -0.01984391   -0.03568115    0.11291958
## 6           NaN   -0.79234421    0.04182036    0.12252308   -0.89469582
##   fundamental_10 fundamental_11 fundamental_12 fundamental_13
## 1            NaN     1.41274226   -0.029575348     1.26524603
## 2    -0.06233732    -0.20224664    1.746691585    -0.18750525
## 3    -0.05828731    -0.89951503   -0.022131024    -0.07900227
## 4    -0.04988065     0.01986567    0.009356987     0.22988893
## 5     0.10423920    -0.16781068   -0.025993237    -0.22842701
## 6     5.24008036    -0.73875082    0.075777955    -0.12490548
##   fundamental_14 fundamental_15 fundamental_16 fundamental_17
## 1    -0.05574706     1.59225595     -0.2852753   -0.212887898
## 2    -0.03466408    -0.13517660      0.3055622    0.027594831
```

```
## 3    -0.03111553    -0.10060307     -0.3414741   -0.007521908
## 4     0.01816135     0.05057177      0.1689191   -0.444930524
## 5     0.13959038    -0.07559659     -0.1416952   -0.033660695
## 6     0.22623423    -0.04075276     -0.3833538   -6.435785294
##   fundamental_18 fundamental_19 fundamental_20 fundamental_21
## 1     0.40418023      0.1169028     0.19758974    -0.19745456
## 2    -0.20874569      0.1185199    -0.12351113     0.11381815
## 3     0.07797654     -0.1495810     0.03694647     0.16618180
## 4    -0.31024513      0.5003964    -0.03373413     0.05041546
## 5     0.02183271      3.2118070     0.32805243    -0.12994799
## 6    -0.97555882     -0.2866374    -0.08543911     0.87818182
##   fundamental_22 fundamental_23 fundamental_24 fundamental_25
## 1     -0.1950233   -0.059886154    -0.02119897    -0.01355474
## 2            NaN   -0.115836978     0.02828172     0.07656525
## 3            NaN    0.172750562     0.96109074     1.02820516
## 4      0.3829628   -0.098141208            NaN    -0.21680045
## 5     -0.1323387    0.037320133     0.06436387    -0.14015268
## 6      0.0405086    0.003751554     0.36500034            NaN
##   fundamental_26 fundamental_27 fundamental_28 fundamental_29
## 1     -0.2363706    -0.25335732     0.57556158     0.28394711
## 2     -0.2173460     0.10005385     0.35880750     0.05750468
## 3     -0.1653695     0.05600566            NaN     0.24952987
## 4            NaN            NaN     0.09381972     0.20181367
## 5     -0.1468391    -0.11660887    -0.21563920     2.44870591
## 6     -0.2281016            NaN            NaN    -0.30810022
##   fundamental_30 fundamental_31 fundamental_32 fundamental_33
## 1    -0.00620764      0.6163509   -0.034578145     0.73265231
## 2     0.02571917     -0.1413388    2.293475628     0.02494543
## 3    -0.15607783            NaN    0.019191509     0.05336369
## 4    -0.28183049      0.1024029    0.000134846    -0.11441991
## 5    -0.22797129     -0.2267765   -0.048719175     3.83264589
## 6     0.37192032            NaN   -0.022952750     0.17189245
##   fundamental_34 fundamental_35 fundamental_36 fundamental_37
## 1   -0.002432259    -0.09250765    -0.07218485      0.4472957
## 2    0.388208181    -0.24092659    -0.12623174     -0.1316190
## 3   -0.020246787            NaN     0.06402827     -0.2491056
## 4   -0.074282780    -0.02560604    -0.09164249      0.1116431
## 5   -0.019201363     0.89227074    -0.16825312      0.1758619
## 6    0.062413301     0.05673797    -0.16559221     -0.1290620
##   fundamental_38 fundamental_39 fundamental_40 fundamental_41
## 1     -0.1943178    0.264373422     0.01876296            NaN
## 2      0.2322677   -0.001455282    -0.48986280     0.01731654
## 3     -0.1062164   -0.085508503     0.38358936            NaN
## 4            NaN    0.044684645    -0.06250231     0.01721823
## 5     -0.4974731   -0.079920575     0.31286350     0.19995309
## 6     -1.1107478    0.177814752    -0.15592350    -1.09568095
##   fundamental_42 fundamental_43 fundamental_44 fundamental_45
## 1    -0.14527814    -0.10854690      0.1481892    -0.22694202
## 2     0.02732062     0.33418259      0.1135128     0.11153645
## 3    -0.07402094    -0.05429483     -0.7460732    -0.05760493
## 4     0.02641706    -0.01198676      0.0489116     0.01690083
## 5     0.19592221    -0.06268372      0.5598638    -0.36446521
## 6    -0.87539679     0.27140915     -0.1547605     0.07140402
##   fundamental_46 fundamental_47 fundamental_48 fundamental_49
```

```
## 1      0.2628636   -0.252933115    -0.11326484     0.18055937
## 2     -0.1793791    0.221287444    -0.09188633            NaN
## 3     -0.1262382    0.462523431     0.06090220            NaN
## 4      0.1787913   -0.252577633    -0.07975223    -0.05074152
## 5     -0.1026950   -0.008203458    -0.20312336     0.08170344
## 6      0.1522563            NaN    -0.15281743     0.02987363
##   fundamental_50 fundamental_51 fundamental_52 fundamental_53
## 1    -0.03381164     0.04295074    -0.06803131     0.20840208
## 2     0.10798185     0.54868716    -0.16007870    -0.28136835
## 3    -0.26289243    -0.05939471     0.56297481    -0.02639207
## 4     0.10440151     0.01033698    -0.04325306     0.08583502
## 5     0.11980287    -0.02408651     0.34460610     0.08251095
## 6    -0.20487507    -0.05336672    -0.06268062    -0.40910393
##   fundamental_54 fundamental_55 fundamental_56 fundamental_57
## 1    -0.08274297    1.061058998     1.12080145    -0.22820024
## 2     0.06393260   -0.235643029    -0.20508254    -0.19434905
## 3     0.01343005   -0.627149761    -0.47036201    -0.10186490
## 4            NaN    0.004252009    -0.01389190            NaN
## 5     0.10433068   -0.189730763    -0.03869983    -0.06889307
## 6    -0.09925562   -0.168522671    -0.15034601    -0.21348691
##   fundamental_58 fundamental_59 fundamental_60 fundamental_61
## 1    -0.11998748    -0.13192914    -0.14598490    -0.15598874
## 2    -0.36518562     0.04192833    -0.04490655    -0.03907863
## 3     0.43679401    -0.05704880    -0.12085094    -0.06194123
## 4    -0.02921824     0.05176127    -0.02311574            NaN
## 5     0.28059679    -0.09730822    -0.06551986    -0.02344253
## 6    -0.11472964     0.18792215     0.28341451            NaN
##   fundamental_62 fundamental_63 technical_0 technical_1 technical_2
## 1            NaN     0.03768013         NaN         NaN          -2
## 2     -0.0750000    -0.28041780         NaN         NaN          -2
## 3     -0.6000193     0.13835683         NaN         NaN          -2
## 4      0.1339363            NaN         NaN         NaN           0
## 5      0.2571468     0.45949975         NaN         NaN          -2
## 6     -0.1162348            NaN         NaN         NaN          -2
##   technical_3 technical_5 technical_6 technical_7 technical_9 technical_10
## 1         NaN         NaN          -2  -0.2739574         NaN           -2
## 2         NaN         NaN          -2  -0.1594319         NaN            0
## 3         NaN         NaN          -2  -0.2278113         NaN           -2
## 4         NaN         NaN          -2  -0.1063338         NaN          NaN
## 5         NaN         NaN           0   0.0000000         NaN            0
## 6         NaN         NaN          -2  -0.2736721         NaN           -2
##   technical_11 technical_12 technical_13 technical_14 technical_16
## 1           -2          NaN  0.001651551           -2          NaN
## 2           -2          NaN  0.004316619            0          NaN
## 3           -2          NaN  0.000000000           -2          NaN
## 4           -2          NaN  0.000000000           -2          NaN
## 5           -2          NaN  0.000000000           -2          NaN
## 6           -2          NaN  0.000000000           -2          NaN
##   technical_17 technical_18 technical_19 technical_20 technical_21
## 1           -2          NaN   0.65298057  0.000000000   -0.1424528
## 2           -2          NaN  -0.39952040  0.000000000   -0.3093565
## 3           -2          NaN  -0.49628371  0.006941625    0.1228407
## 4            0          NaN   0.99062681  0.006765784    0.8126783
## 5           -2          NaN  -0.08668493  0.006236139   -0.4111451
```

```
## 6           -2          NaN   0.49715295  0.009999788    0.9452949
##   technical_22 technical_24 technical_25 technical_27 technical_28
## 1          0.0          NaN          NaN    1.4274690          NaN
## 2         -0.5          NaN          NaN    0.1546130          NaN
## 3         -0.5          NaN          NaN    0.3783325          NaN
## 4          0.0          NaN          NaN    1.0237128          NaN
## 5         -0.5          NaN          NaN    0.7415445          NaN
## 6          0.5          NaN          NaN   -0.2268579          NaN
##   technical_29 technical_30 technical_31 technical_32 technical_33
## 1           -2            0          NaN          NaN          NaN
## 2            0            0          NaN          NaN          NaN
## 3           -2            0          NaN          NaN          NaN
## 4           -2            0          NaN          NaN          NaN
## 5            0            0          NaN          NaN          NaN
## 6           -2            0          NaN          NaN          NaN
##   technical_34 technical_35 technical_36 technical_37 technical_38
## 1          0.0    0.9378802   0.77520812          NaN          NaN
## 2          0.5    0.2321541   0.02559014          NaN          NaN
## 3         -0.5    0.3726879   0.15188117          NaN          NaN
## 4         -0.5    0.7510211   1.03593647          NaN          NaN
## 5          0.5    0.5952055   0.63023198          NaN          NaN
## 6         -0.5   -0.3997368  -0.37936625          NaN          NaN
##   technical_39 technical_40 technical_41 technical_42 technical_43
## 1          NaN -0.414775848          NaN          NaN           -2
## 2          NaN -0.273607433          NaN          NaN           -2
## 3          NaN -0.175710350          NaN          NaN           -2
## 4          NaN -0.211506337          NaN          NaN           -2
## 5          NaN -0.001956611          NaN          NaN            0
## 6          NaN -0.001956611          NaN          NaN           -2
##   technical_44           y
## 1          NaN -0.011753449
## 2          NaN -0.001240137
## 3          NaN -0.020939544
## 4          NaN -0.015959399
## 5          NaN -0.007337788
## 6          NaN  0.031425107
```

```r
View(df)

# Visualizando os nomes das colunas
names(df)
```

```
##   [1] "id"             "timestamp"      "derived_0"      "derived_1"
##   [5] "derived_2"      "derived_3"      "derived_4"      "fundamental_0"
##   [9] "fundamental_1"  "fundamental_2"  "fundamental_3"  "fundamental_5"
##  [13] "fundamental_6"  "fundamental_7"  "fundamental_8"  "fundamental_9"
##  [17] "fundamental_10" "fundamental_11" "fundamental_12" "fundamental_13"
##  [21] "fundamental_14" "fundamental_15" "fundamental_16" "fundamental_17"
##  [25] "fundamental_18" "fundamental_19" "fundamental_20" "fundamental_21"
##  [29] "fundamental_22" "fundamental_23" "fundamental_24" "fundamental_25"
##  [33] "fundamental_26" "fundamental_27" "fundamental_28" "fundamental_29"
##  [37] "fundamental_30" "fundamental_31" "fundamental_32" "fundamental_33"
##  [41] "fundamental_34" "fundamental_35" "fundamental_36" "fundamental_37"
##  [45] "fundamental_38" "fundamental_39" "fundamental_40" "fundamental_41"
##  [49] "fundamental_42" "fundamental_43" "fundamental_44" "fundamental_45"
```

```
##  [53] "fundamental_46" "fundamental_47" "fundamental_48" "fundamental_49"
##  [57] "fundamental_50" "fundamental_51" "fundamental_52" "fundamental_53"
##  [61] "fundamental_54" "fundamental_55" "fundamental_56" "fundamental_57"
##  [65] "fundamental_58" "fundamental_59" "fundamental_60" "fundamental_61"
##  [69] "fundamental_62" "fundamental_63" "technical_0"    "technical_1"
##  [73] "technical_2"    "technical_3"    "technical_5"    "technical_6"
##  [77] "technical_7"    "technical_9"    "technical_10"   "technical_11"
##  [81] "technical_12"   "technical_13"   "technical_14"   "technical_16"
##  [85] "technical_17"   "technical_18"   "technical_19"   "technical_20"
##  [89] "technical_21"   "technical_22"   "technical_24"   "technical_25"
##  [93] "technical_27"   "technical_28"   "technical_29"   "technical_30"
##  [97] "technical_31"   "technical_32"   "technical_33"   "technical_34"
## [101] "technical_35"   "technical_36"   "technical_37"   "technical_38"
## [105] "technical_39"   "technical_40"   "technical_41"   "technical_42"
## [109] "technical_43"   "technical_44"   "y"
```

```
# Observacoes Iniciais
```

```
# O dataset contém 1.710.756 observacoes e 111 atributos
# 01 coluna 'id'
# 01 coluna 'timestamp'
# 05 colunas 'derived_X', onde X é uma sequencia numerica iniciando de 0
# 63 colunas 'fundamental_X', onde X é uma sequencia numerica iniciando de 0 (exceto 'fundamental_4' ??
# 40 colunas 'technical_X', onde X é uma sequencia numerica iniciando de 0 (exceto _4, _8, _15, _23 e _
# 01 coluna "y" que é o alvo
```

```
# 02 - Analise Exploratoria de Dados
```

```
# Carregando os Pacotes
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(ggplot2)
library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
```

```
##
##     format.pval, units
```

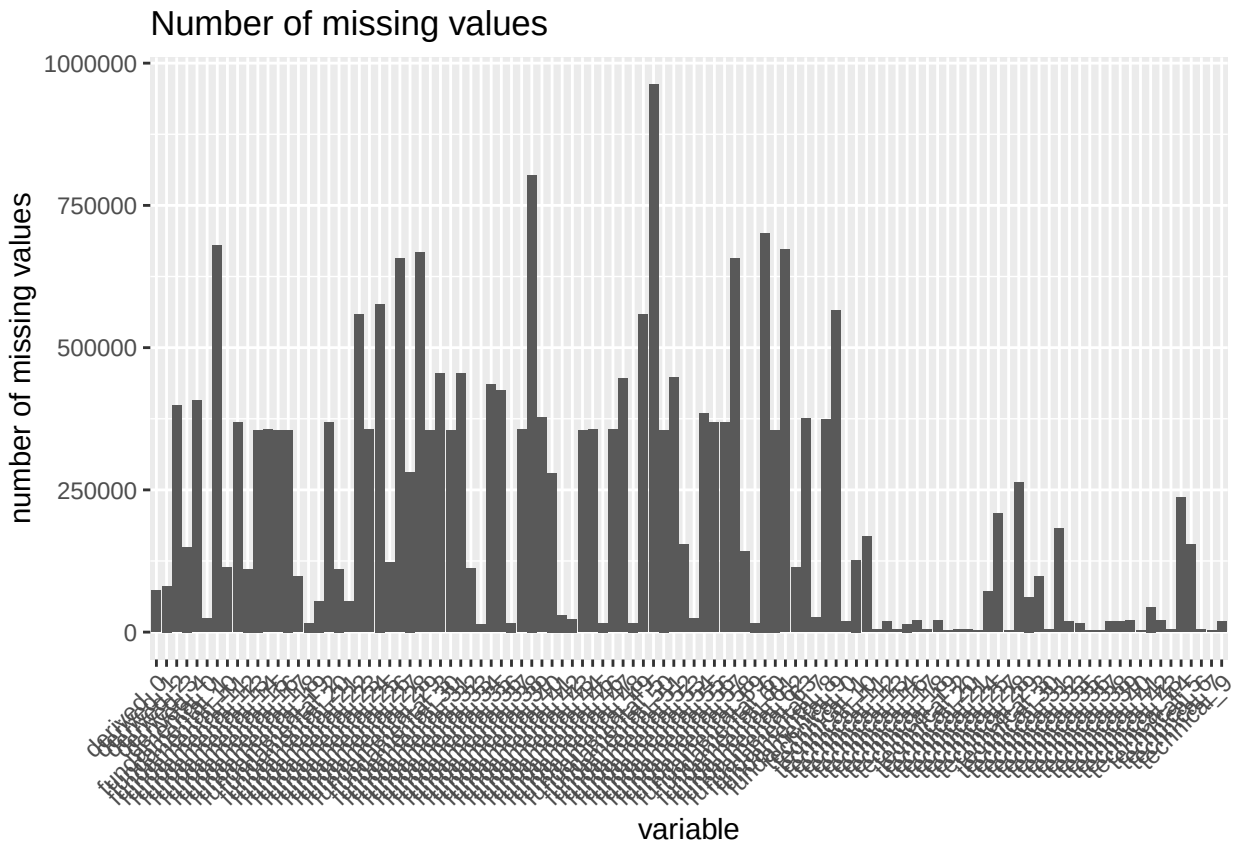
```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
# Verificar se existem valores ausentes (missing) em cada coluna
# Valor missing encontrado
any(is.na(df))
```

```
## [1] TRUE
```

```
# Visualizando valores missing no dataset
# Gerando um novo df agrupando os dados missing
missing.values <- df %>%
  gather(key = "key", value = "val") %>%
  mutate(is.missing = is.na(val)) %>%
  group_by(key, is.missing) %>%
  summarise(num.missing = n()) %>%
  filter(is.missing==T) %>%
  select(-is.missing) %>%
  arrange(desc(num.missing))

# Visualizando usando ggplot
missing.values %>%
  ggplot() +
  geom_bar(aes(x=key, y=num.missing), stat = 'identity') +
  labs(x='variable', y="number of missing values", title='Number of missing values') +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
head(missing.values)
```

```
## # A tibble: 6 x 2
## # Groups:   key [6]
##   key            num.missing
##   <chr>                <int>
## 1 fundamental_5       962020
## 2 fundamental_38      803489
## 3 fundamental_6       701625
## 4 fundamental_1       679070
## 5 fundamental_61      671801
## 6 fundamental_28      667331
```

```
# Observacoes sobre Missing Values
```

```
# Coluna 3 a 111 sao features independentes
# Existem muitas colunas com valores missing
# Variaveis com maior % de valores missing:
# fundamental_5, fundamental_38, fundamental_6, fundamental_1, fundamental_61 e fundamental_28

# Tratamento dos valores missing
# Aplicando o valor médio das colunas
```

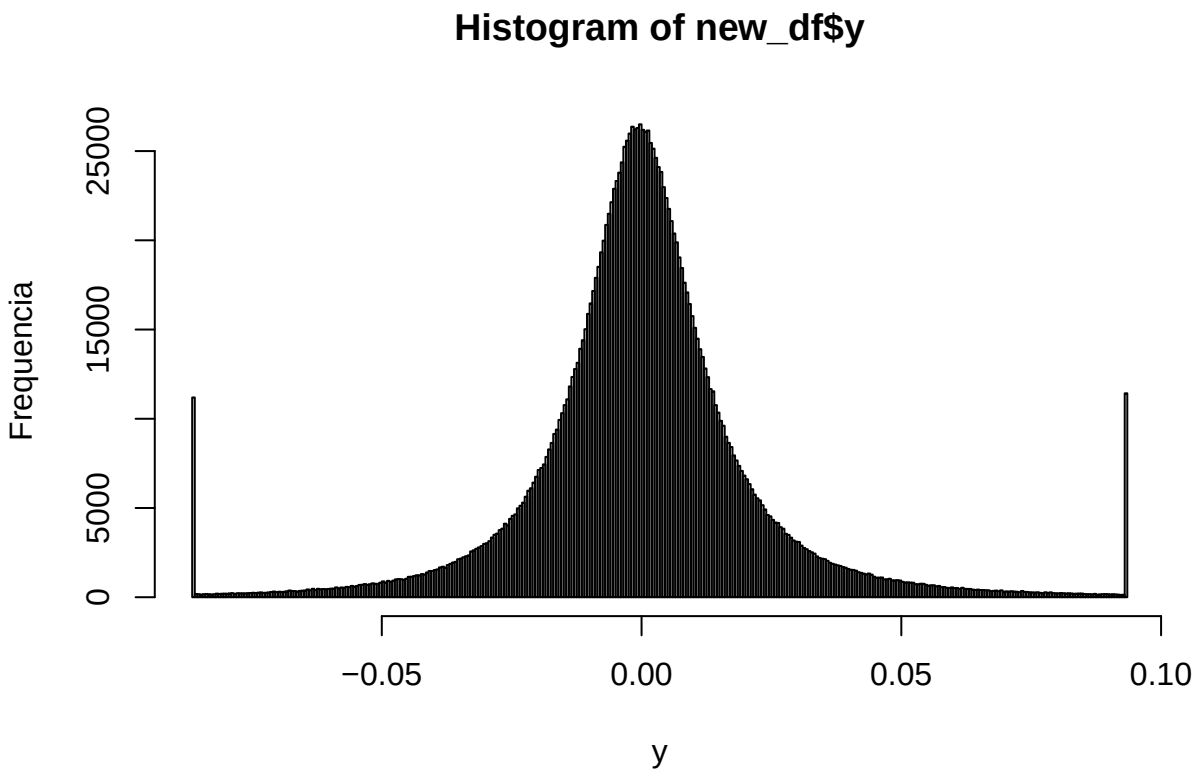
```
# Criando uma copia do dataset original
new_df <- cbind(df)

for(i in 1:ncol(new_df)){
  new_df[is.na(new_df[,i]), i] <- mean(new_df[,i], na.rm = TRUE)
```

```
}

# Verificar se existem valores ausentes (missing) em cada coluna no novo dataset
any(is.na(new_df))
```

```
## [1] FALSE
```

```
# Avaliando a variavel target "y" (considerando o novo dataset sem valores missing)
```

```
hist(new_df$y, "FD", xlab="y", ylab="Frequencia")
```




```
# Analise estatistica da variavel target 'y'
describe(new_df$y)
```

```
## new_df$y
##          n    missing   distinct       Info       Mean        Gmd
##    1710756          0    1672946          1  0.0002217    0.02279
##        .05        .10        .25        .50        .75        .90
## -0.0334915 -0.0220302 -0.0095614 -0.0001571  0.0095210  0.0227159
##        .95
##  0.0351158
##
## lowest : -0.08609413 -0.08609316 -0.08609165 -0.08609056 -0.08608633
## highest:  0.09347494  0.09348199  0.09348705  0.09349189  0.09349781
```

```
# Verificar os outliers
outliers <- distinct(filter(new_df, new_df$y >= 0.0934 | new_df$y <= -0.0860))
describe(outliers$id)
```

```
## outliers$id
##        n  missing distinct     Info     Mean      Gmd      .05      .10
```

```
##    22504        0     1284        1     1094    721.8      115      221
##      .25      .50      .75      .90      .95
##      556     1118     1637     1966     2054
##
## lowest :    0    7   10   11   12, highest: 2152 2154 2155 2156 2158
```

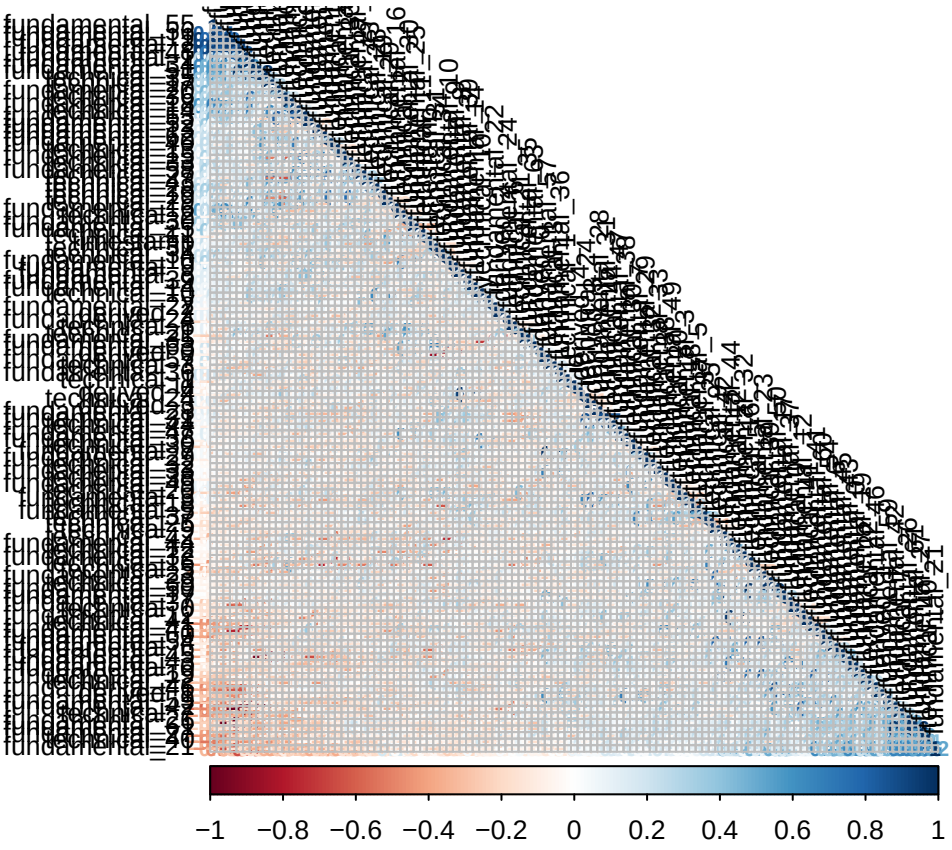
```
# Observacoes da variavel target
```

```
# A variavel y mostra uma distribuicao normal, exceto na extremidade da cauda
# Olhando os dados da cauda, verificamos que o outlier gira em torno de 0,0934 e -0,086
# Da a "impressao" que houve corte nos dados
# Verificando os outliers dos IDs, percebe que a maioria sao valores extremos
#   ex: 1284 id distintos onde 95% em 2054
```

```
# 03 - Feature Engineering
```

```
# Carregando os Pacotes
library(scales)

# Gerando uma copia do dataset
dfTrain <- new_df

# Normalizando as variáveis numericas
scale.features <- function(df, variables){
  for (variable in variables){
    df[[variable]] <- scale(df[[variable]], center=T, scale=T)
  }
  return(df)
}

# Normalizando todas as variaveis numericas
numeric.vars <- unlist(lapply(dfTrain, is.numeric))
dfTrain <- scale.features(dfTrain, numeric.vars)

# Analise de Correlacao
```

```
# Separando as colunas numericas para correlacao
numeric.vars <- unlist(lapply(dfTrain, is.numeric))
data_cor <- cor(dfTrain[,numeric.vars])

# Visualizando a correlacao de uma amostra do dataset
rrow <- sample(1:dim(dfTrain)[1],50)
corGraph <- cor(dfTrain[rrow, ])
corrplot(corGraph, order = "FPC", method = "number", type = "lower",
         tl.cex = 0.8, tl.col = rgb(0, 0, 0),number.cex = 0.7, number.digits = 2)
```




```
# Observacoes da correlacao
```

```
# De acordo com o grafico de correlacao, as features abaixo já estao correlacionadas cmo outras feature
# Por isso, vou remove-las no momento certo, na avaliacao de alguns modelos
# fundamental_61,fundamental_11,fundamental_56,fundamental_26,fundamental_10,fundamental_15,
# fundamental_57,fundamental_41,fundamental_30,fundamental_53,fundamental_42,fundamental_26,
# fundamental_10,fundamental_60,fundamental_48,fundamental_55,fundamental_11,fundamental_45,
# fundamental_16,fundamental_34,fundamental_12,fundamental_51,fundamental_43,fundamental_1,
# fundamental_42,fundamental_30,fundamental_53
```

```
# 05 - Criando alguns modelos de ML para comparacoes
```

```
# Observacoes
```

```
# Como a variavel target 'y' é uma variável contínua, o modelo mais fácil de construir é o modelo de re
# Vou criar três modelos de regressão: linear regression, ridge regression and lasso regression.
# A diferença entre ridge e lasso está na função de penalidade.
# Também vou avaliar a performance usando o modelo Generalized Boosted Regression Modeling (GBM) e
# eXtreme Gradient Boosting (XGBoost)

# Carregando os Pacotes
library(caret)
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:survival':
##
##     cluster
```

```r
# Gerando dados de treino e de teste
```

```r
splits <- createDataPartition(dfTrain$y, p=0.8, list=FALSE)

# Separando os dados de treino e teste
dados_treino <- dfTrain[ splits,]
dados_teste <- dfTrain[-splits,]

# Definindo a formula para os modelos
# Usando as colunas com maiores correlacoes
#formula <- "y ~ ."
formula <- "y ~ fundamental_5 + fundamental_7 + fundamental_14 +
                fundamental_17 + fundamental_19 + fundamental_20 + fundamental_21 +
                fundamental_31 + fundamental_33 + fundamental_36 + fundamental_40 +
                fundamental_44 + fundamental_46 + fundamental_47 + fundamental_49 +
                fundamental_63 + technical_0 + technical_7 + technical_11 +
                technical_16 + technical_20 + technical_21 + technical_22 +
                technical_27 + technical_30 + technical_32 + technical_36 +
                technical_37 + technical_44"
formula <- as.formula(formula)


# Funcao para calcular o valor R (que sera a forma de avaliacao dos modelos)
# Usei este metodo pra ficar de acordo com a solicitacao do desafio no Kaggle
r_value <- function(R_sq){
  R_val <- sign(R_sq)*sqrt(abs(R_sq))
  return(R_val)
}

# Construindo um modelo Linear Regression (LM)
```

```r
set.seed(1234)
controlLM <- trainControl(method="cv", number=5)
modeloLM <- train(formula, data = dados_treino, method = "lm", trControl=controlLM)

# Fazendo previsoes para avaliar a performance do modelo nos dados de teste
predLM <- modeloLM %>% predict(dados_teste)

# Avaliando a performance do modelo
linearR = r_value(R2(predLM, dados_teste$y))
print(linearR)
```

```
## [1] 0.02129646
```

```r
# Construindo um modelo Ridge Regression
```

```r
# Criando o modelo
set.seed(1234)
controlRidge <- trainControl(method="cv", number=5)
modeloRidge <- train(formula, data = dados_treino, method = "glmnet",
                     trControl = controlRidge,
                     tuneGrid = expand.grid(alpha = 0, lambda = 0))

# Fazendo previsoes para avaliar a performance do modelo
predRigde <- modeloRidge %>% predict(dados_teste)
```

```
# Avaliando a performance do modelo
ridgeR = r_value(R2(predRigde, dados_teste$y))
print(ridgeR)
```

```
## [1] 0.02127378
```

```
# Construindo um modelo Lasso Regression
```

```
# Criando o modelo
set.seed(1234)
controlLasso <- trainControl(method="cv", number=5)
modeloLasso <- train(formula, data = dados_treino, method = "glmnet",
                     trControl = controlLasso,
                     tuneGrid = expand.grid(alpha = 1, lambda = 0))

# Fazendo previsoes para avaliar a performance do modelo
predLasso <- modeloLasso %>% predict(dados_teste)

# Avaliando a performance do modelo
lassoR = r_value(R2(predLasso, dados_teste$y))
print(lassoR)
```

```
## [1] 0.02126376
```

```
# Construindo um modelo Generalized Boosted Regression Modeling (GBM)
```

```
# Criando o modelo
set.seed(1234)
controlGBM <- trainControl(method="cv", number=2)
modeloGBM <- train(formula, data=dados_treino, method="gbm",  verbose=FALSE, trControl=controlGBM)

# Fazendo previsoes para avaliar a performance do modelo
predGBM <- modeloGBM %>% predict(dados_teste)

# Avaliando a performance do modelo
GBM_R = r_value(R2(predGBM, dados_teste$y))
print(GBM_R)
```

```
## [1] 0.03092463
```

```
# Comparando a performance dos modelos
```

```
p <- data.frame(linearR, ridgeR, lassoR, GBM_R)
colnames(p) = c('Linear', 'Ridge', 'Lasso', 'GBM')
m <- as.matrix(p)
m
```

```
##          Linear      Ridge      Lasso        GBM
## [1,] 0.02129646 0.02127378 0.02126376 0.03092463
```

```
# Observacoes da Performance dos modelos (dados de teste)
```

```
# Podemos verificar que usando todas as variaveis o modelo Generalized Boosted Regression Modeling teve
# Vou usar esse modelo para otimizacao dos hyperparametros
```

```
# 06 - Otimizando o modelo Generalized Boosted Regression Modeling
```

```r
# Carregando os Pacotes
library(gbm)
```

```
## Loaded gbm 2.1.5
```

```r
# Criando grid de hyperparameter
hyper_grid <- expand.grid(
  shrinkage = c(.01, .1),
  interaction.depth = c(1, 3),
  n.minobsinnode = c(5, 10),
  bag.fraction = c(.65, .8),
  optimal_trees = 0,
  min_RMSE = 0
)

# Numero total de combinacoes
nrow(hyper_grid)
```

```
## [1] 16
```

```r
# Grid Search
for(i in 1:nrow(hyper_grid)) {

  # Criando o modelo
  set.seed(1234)

  gbm.tune <- gbm(
    formula = formula,
    distribution = "gaussian",
    data = dados_treino,
    n.trees = 100,
    interaction.depth = hyper_grid$interaction.depth[i],
    shrinkage = hyper_grid$shrinkage[i],
    n.minobsinnode = hyper_grid$n.minobsinnode[i],
    bag.fraction = hyper_grid$bag.fraction[i],
    train.fraction = .75,
    n.cores = NULL,
    verbose = FALSE
  )

  # Verificando os erros do treinamento
  hyper_grid$optimal_trees[i] <- which.min(gbm.tune$valid.error)
  hyper_grid$min_RMSE[i] <- sqrt(min(gbm.tune$valid.error))
}

hyper_grid %>%
  dplyr::arrange(min_RMSE) %>%
  head(10)
```

```
##    shrinkage interaction.depth n.minobsinnode bag.fraction optimal_trees
## 1       0.10                 3              5         0.80            54
## 2       0.10                 3             10         0.80            66
## 3       0.10                 3              5         0.65            77
## 4       0.10                 3             10         0.65            67
## 5       0.10                 1             10         0.65           100
```

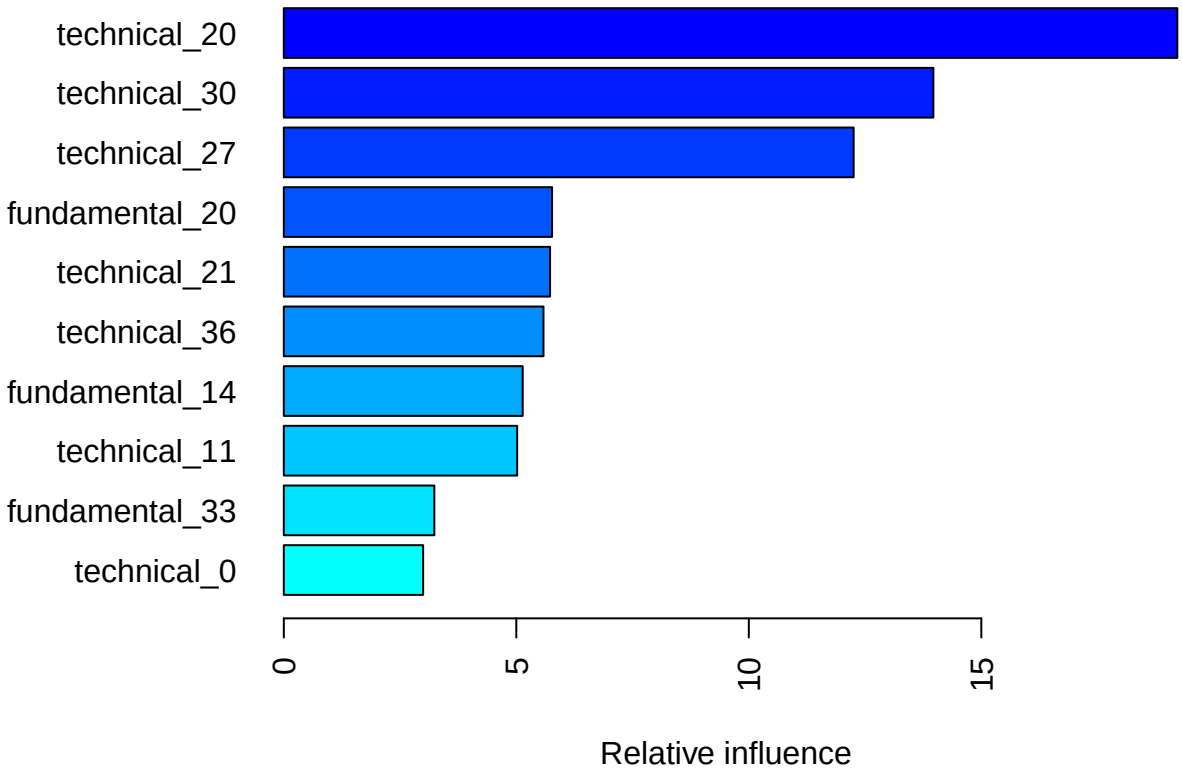
```
## 6       0.10                   1              5         0.65            91
## 7       0.10                   1             10         0.80           100
## 8       0.10                   1              5         0.80            81
## 9       0.01                   3             10         0.80           100
## 10      0.01                   3              5         0.80           100
##      min_RMSE
## 1  0.02835203
## 2  0.02835206
## 3  0.02835277
## 4  0.02835307
## 5  0.02835429
## 6  0.02835508
## 7  0.02835569
## 8  0.02835573
## 9  0.02835954
## 10 0.02835980
```

```
# print results
print(gbm.tune)
```

```
## gbm(formula = formula, distribution = "gaussian", data = dados_treino,
##     n.trees = 100, interaction.depth = hyper_grid$interaction.depth[i],
##     n.minobsinnode = hyper_grid$n.minobsinnode[i], shrinkage = hyper_grid$shrinkage[i],
##     bag.fraction = hyper_grid$bag.fraction[i], train.fraction = 0.75,
##     verbose = FALSE, n.cores = NULL)
## A gradient boosted model with gaussian loss function.
## 100 iterations were performed.
## The best test-set iteration was 66.
## There were 29 predictors of which 26 had non-zero influence.
```

```
# 07 - Avaliando o modelo otimizado nos dados de teste
```

```
# train GBM model
modeloGBM_otm <- gbm(
  formula = formula,
  distribution = "gaussian",
  data = dados_treino,
  n.trees = 80,
  interaction.depth = 3,
  shrinkage = 0.1,
  n.minobsinnode = 10,
  bag.fraction = 0.8,
  train.fraction = 1,
  n.cores = NULL,
  verbose = FALSE
)

par(mar = c(5, 8, 1, 1))
summary(
  modeloGBM_otm,
  cBars = 10,
  method = relative.influence, # also can use permutation.test.gbm
  las = 2
)
```




```
##                          var    rel.inf
## technical_20     technical_20 19.2152609
## technical_30     technical_30 13.9693849
## technical_27     technical_27 12.2545858
## fundamental_20 fundamental_20  5.7690702
## technical_21     technical_21  5.7257133
## technical_36     technical_36  5.5844788
## fundamental_14 fundamental_14  5.1373370
## technical_11     technical_11  5.0179007
## fundamental_33 fundamental_33  3.2374344
## technical_0       technical_0  2.9958794
## fundamental_36 fundamental_36  2.8695571
## fundamental_44 fundamental_44  2.5346257
## technical_7       technical_7  2.3745878
## fundamental_47 fundamental_47  1.9903502
## technical_16     technical_16  1.7940911
## fundamental_21 fundamental_21  1.5466017
## fundamental_63 fundamental_63  1.4950254
## fundamental_17 fundamental_17  1.4506363
## technical_44     technical_44  1.2735725
## fundamental_19 fundamental_19  1.1457353
## fundamental_7   fundamental_7  0.8414167
## fundamental_31 fundamental_31  0.5800277
## technical_37     technical_37  0.4857313
## fundamental_49 fundamental_49  0.3910426
## fundamental_46 fundamental_46  0.3199531
## fundamental_5   fundamental_5  0.0000000
## fundamental_40 fundamental_40  0.0000000
## technical_22     technical_22  0.0000000
```

```
## technical_32     technical_32  0.0000000
```

```
# predict values for test data
predGBM_otm <- predict(modeloGBM_otm, n.trees = modeloGBM_otm$n.trees, dados_teste)

# Avaliando a performance do modelo
GBM_otm = r_value(R2(predGBM_otm, dados_teste$y))
print(GBM_otm)
```

```
## [1] 0.03662339
```

```
# Comparando a performance dos modelos finais
```

```
p <- data.frame(linearR, ridgeR, lassoR, GBM_R, GBM_otm)
colnames(p) = c('Linear', 'Ridge', 'Lasso', 'GBM', 'GBM_otm')
m <- as.matrix(p)
m
```

```
##          Linear      Ridge      Lasso        GBM    GBM_otm
## [1,] 0.02129646 0.02127378 0.02126376 0.03092463 0.03662339
```

```
# 09 - Conclusao Final
```

```
# De acordo com o Kaggle, foi descrito para não desanimar com baixos valores de R
# Em finanças, dada a alta proporção de sinal-ruído, até um pequeno R pode oferecer um valor significat
```

```
# O melhor algoritmo para esse dataset é foi o Generalized Boosted Regression Modeling
# O modelo GBM considerando a selecao das variaveis do dataset teve uma performance de 0.03263341
# Otimizando este modelo, obtive uma performance de 0.03726622
# Comparando com o leaderboard do Kaggle, foi um resultado muito bom (apesar de nao submeter)
# O ideal seria validar esse modelo gerando o submission, mas não é mais possivel pois a competição foi
```