Desafio A3Data

Felipe A. Petri

2023-07-27

```
library(tidyverse)
library(readxl)
library(caret)
library(ROSE)
library(randomForest)
library(kableExtra)
```

Introdução das análises

```
churn=read_xlsx("Customer-Churn.xlsx")
head(churn)
```

```
## # A tibble: 6 x 21
##   customerID gender SeniorCitizen Partner Dependents tenure PhoneService
##   <chr>      <chr>          <dbl> <chr>   <chr>       <dbl> <chr>
## 1 7569-NMZYQ Female             0 Yes     Yes            72 Yes
## 2 8984-HPEMB Female             0 No      No             71 Yes
## 3 5989-AXPUC Female             0 Yes     No             68 Yes
## 4 5734-EJKXG Female             0 No      No             61 Yes
## 5 8199-ZLLSA Male               0 No      No             67 Yes
## 6 9924-JPRMC Male               0 No      No             72 Yes
## # i 14 more variables: MultipleLines <chr>, InternetService <chr>,
## #   OnlineSecurity <chr>, OnlineBackup <chr>, DeviceProtection <chr>,
## #   TechSupport <chr>, StreamingTV <chr>, StreamingMovies <chr>,
## #   Contract <chr>, PaperlessBilling <chr>, PaymentMethod <chr>,
## #   MonthlyCharges <dbl>, TotalCharges <dbl>, Churn <chr>
```

```
# Transformando variáveis categóricas em 0s e 1s (fica mais fácil para futura análise)
churn=churn %>%
  mutate_at(c(4,5,7,8,10:15,17,21),~ifelse(.=="Yes",1,0))
# Alterando tipos de colunas
churn=churn %>%
  mutate_at(c(2:5,7:18,21),~as.factor(.))
```

Introdução aos dados

```
churn[,c(1:10)] %>%
  summary() %>%
  kable(format="latex",escape=FALSE) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_options
  row_spec(0, bold = TRUE, background = "#D3D3D3")
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity
	Length:7043	Female:3488	0:5901	0:3641	0:4933	Min. : 0.00	0: 682	0:4072	DSL :2421	0:5024
	Class :character	Male :3555	1:1142	1:3402	1:2110	1st Qu.: 9.00	1:6361	1:2971	Fiber optic:3096	1:2019
	Mode :character	NA	NA	NA	NA	Median :29.00	NA	NA	No :1526	NA
	NA	NA	NA	NA	NA	Mean :32.37	NA	NA	NA	NA
	NA	NA	NA	NA	NA	3rd Qu.:55.00	NA	NA	NA	NA
	NA	NA	NA	NA	NA	Max. :72.00	NA	NA	NA	NA

```
churn[,c(11:21)] %>%
  summary() %>%
  kable(format="latex",escape=FALSE) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_options
  row_spec(0, bold = TRUE, background = "#D3D3D3")
```

	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
	0:4614	0:4621	0:4999	0:4336	0:4311	Month-to-month:3875	0:2872	Bank transfer (automatic):1544	Min. : 18.25	Min. : 18.8	0:5174
	1:2429	1:2422	1:2044	1:2707	1:2732	One year :1473	1:4171	Credit card (automatic) :1522	1st Qu.: 35.50	1st Qu.: 401.4	1:1869
	NA	NA	NA	NA	NA	Two year :1695	NA	Electronic check :2365	Median : 70.35	Median :1397.5	NA
	NA	NA	NA	NA	NA	NA	NA	Mailed check :1612	Mean : 64.76	Mean :2283.3	NA
	NA	NA	NA	NA	NA	NA	NA	NA	3rd Qu.: 89.85	3rd Qu.:3794.7	NA
	NA	NA	NA	NA	NA	NA	NA	NA	Max. :118.75	Max. :8684.8	NA
	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA's :11	NA

Valores Faltantes

```
summary(churn)
```

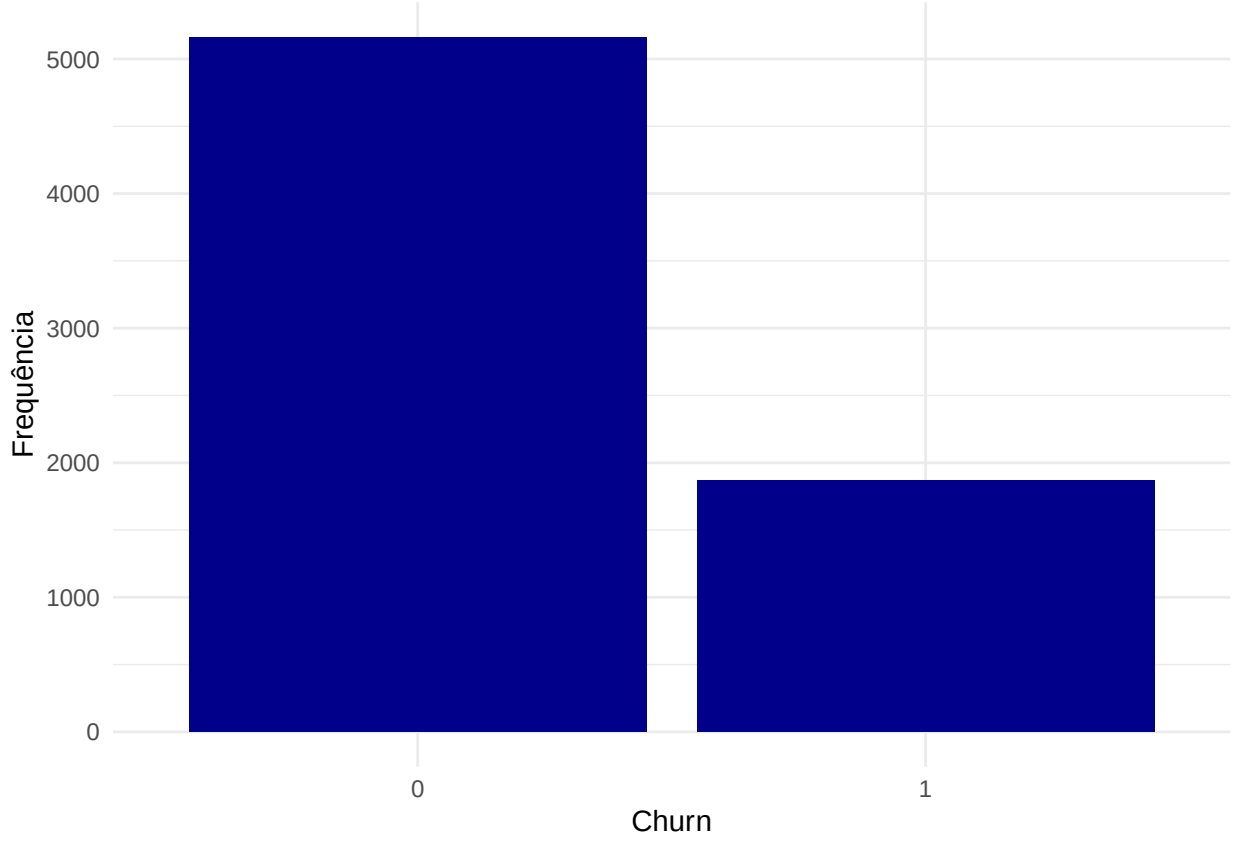
```
##   customerID          gender     SeniorCitizen Partner  Dependents
##  Length:7043        Female:3488   0:5901        0:3641   0:4933
##  Class :character   Male  :3555   1:1142        1:3402   1:2110
##  Mode  :character
##
##
##
##
##      tenure      PhoneService MultipleLines    InternetService OnlineSecurity
##  Min.   : 0.00   0: 682       0:4072         DSL        :2421   0:5024
##  1st Qu.: 9.00   1:6361       1:2971         Fiber optic:3096   1:2019
##  Median :29.00                               No         :1526
##  Mean   :32.37
##  3rd Qu.:55.00
##  Max.   :72.00
##
##  OnlineBackup DeviceProtection TechSupport StreamingTV StreamingMovies
##  0:4614       0:4621           0:4999      0:4336      0:4311
##  1:2429       1:2422           1:2044      1:2707      1:2732
##
##
##
##
##
##            Contract    PaperlessBilling                   PaymentMethod
##  Month-to-month:3875   0:2872           Bank transfer (automatic):1544
##  One year      :1473   1:4171           Credit card (automatic)  :1522
##  Two year      :1695                    Electronic check         :2365
##                                         Mailed check             :1612
##
##
##
##  MonthlyCharges    TotalCharges    Churn
```

```
##  Min.   : 18.25   Min.   :  18.8   0:5174
##  1st Qu.: 35.50   1st Qu.: 401.4   1:1869
##  Median : 70.35   Median :1397.5
##  Mean   : 64.76   Mean   :2283.3
##  3rd Qu.: 89.85   3rd Qu.:3794.7
##  Max.   :118.75   Max.   :8684.8
##                   NA's   :11
```

```
# Como temos apenas uma coluna com 11 dados faltantes podemos remover todos os dados faltantes
churn=churn %>%
  drop_na()
```

Verificando Balanceamento de Dados

```
churn %>%
  ggplot(aes(Churn))+
  geom_bar(fill="darkblue")+
  labs(x="Churn",y="Frequência")+
  theme_minimal()
```



Nossos dados não são balanceados

Treino e Teste

```
# Separando dados em treino e teste
set.seed(1)
```

```r
ind_treino <- sample(nrow(churn), 0.7 * nrow(churn))
treino <- churn[ind_treino, ]
teste <- churn[-ind_treino, ]
```

Análise descritiva

Tabelas de contingência

```r
contingencia=function(x){
  table(treino$Churn,treino[,x][[1]])
}
categoricas=setdiff(names(treino)[sapply(treino, is.factor)], "Churn")
tabelas_contingencia=lapply(categoricas,contingencia)
names(tabelas_contingencia) <- paste("Churn", categoricas, sep = " x ")

# Função para deixar as tabelas bonitas:
beautiful_tabelas=function(tabela,nome){
  tabela=addmargins(tabela)
  rownames(tabela)[nrow(tabela)]="Total"
  colnames(tabela)[ncol(tabela)]="Total"
  tabela=kable(tabela,caption=paste("Tabela de Contingência de",nome), format = "latex") %>%
    kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_opti
  row_spec(0, bold = TRUE, background = "#D3D3D3") %>%
  column_spec(1, bold = TRUE, background = "#D3D3D3")
  return(tabela)
}

# Plotando as tabelas bonitas:
kables_contingencia=imap(tabelas_contingencia,beautiful_tabelas)
for (i in 1:length(kables_contingencia)) {
  cat(kables_contingencia[[i]], "\n\n")
}
```

Table 1: Tabela de Contingência de Churn x gender

	Female	Male	Total
0	1767	1822	3589
1	664	669	1333
Total	2431	2491	4922

Table 2: Tabela de Contingência de Churn x SeniorCitizen

	0	1	Total
0	3127	462	3589
1	1001	332	1333
Total	4128	794	4922

Table 3: Tabela de Contingência de Churn x Partner

	0	1	Total
0	1703	1886	3589
1	840	493	1333
Total	2543	2379	4922

Table 4: Tabela de Contingência de Churn x Dependents

	0	1	Total
0	2374	1215	3589
1	1109	224	1333
Total	3483	1439	4922

Table 5: Tabela de Contingência de Churn x PhoneService

	0	1	Total
0	345	3244	3589
1	121	1212	1333
Total	466	4456	4922

Table 6: Tabela de Contingência de Churn x MultipleLines

	0	1	Total
0	2097	1492	3589
1	725	608	1333
Total	2822	2100	4922

Table 7: Tabela de Contingência de Churn x InternetService

	DSL	Fiber optic	No	Total
0	1328	1263	998	3589
1	318	933	82	1333
Total	1646	2196	1080	4922

Table 8: Tabela de Contingência de Churn x OnlineSecurity

	0	1	Total
0	2398	1191	3589
1	1122	211	1333
Total	3520	1402	4922

Table 9: Tabela de Contingência de Churn x OnlineBackup

	0	1	Total
0	2276	1313	3589
1	944	389	1333
Total	3220	1702	4922

Table 10: Tabela de Contingência de Churn x DeviceProtection

	0	1	Total
0	2292	1297	3589
1	953	380	1333
Total	3245	1677	4922

Table 11: Tabela de Contingência de Churn x TechSupport

	0	1	Total
0	2387	1202	3589
1	1122	211	1333
Total	3509	1413	4922

Table 12: Tabela de Contingência de Churn x StreamingTV

	0	1	Total
0	2269	1320	3589
1	746	587	1333
Total	3015	1907	4922

Table 13: Tabela de Contingência de Churn x StreamingMovies

	0	1	Total
0	2258	1331	3589
1	733	600	1333
Total	2991	1931	4922

Table 14: Tabela de Contingência de Churn x Contract

	Month-to-month	One year	Two year	Total
0	1543	877	1169	3589
1	1176	122	35	1333
Total	2719	999	1204	4922

Table 15: Tabela de Contingência de Churn x PaperlessBilling

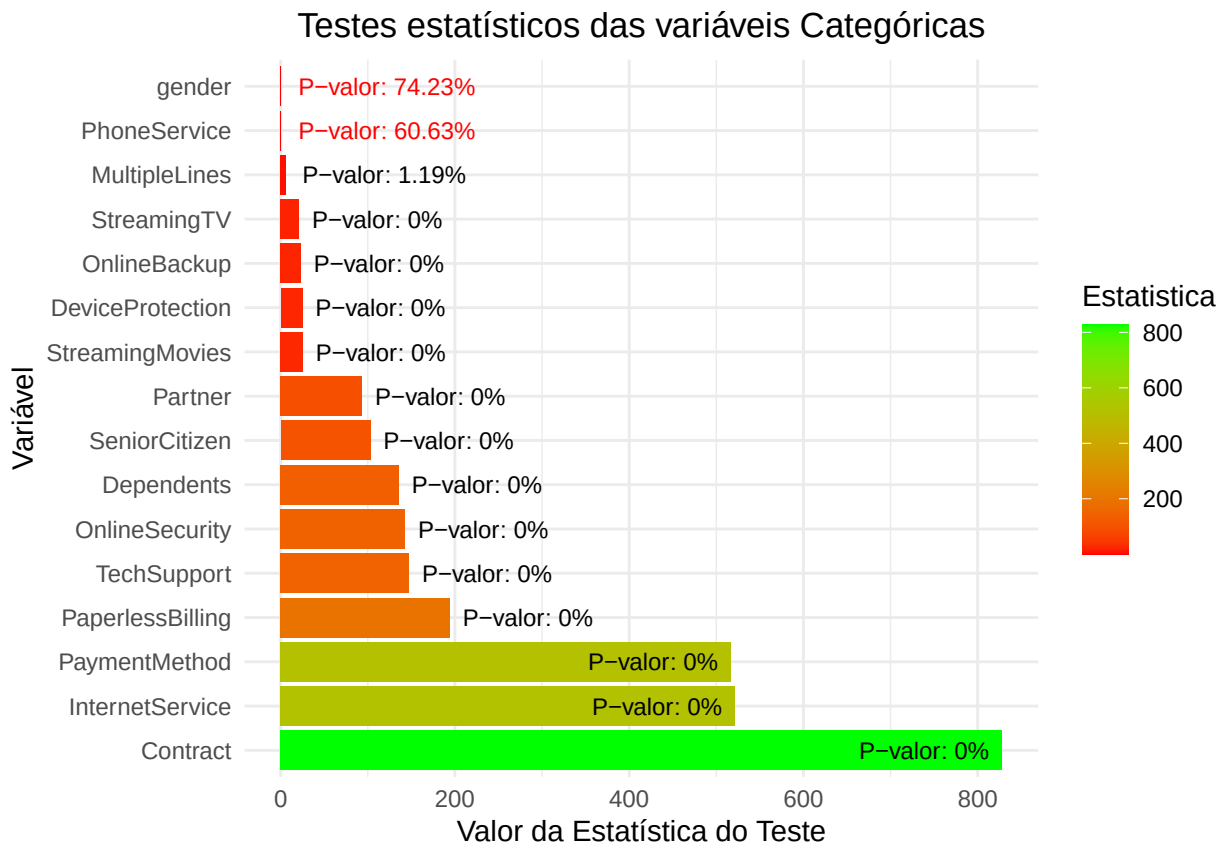
	0	1	Total
0	1676	1913	3589
1	329	1004	1333
Total	2005	2917	4922

Table 16: Tabela de Contingência de Churn x PaymentMethod

	Bank transfer (automatic)	Credit card (automatic)	Electronic check	Mailed check	Total
0	898	906	910	875	3589
1	177	147	797	212	1333
Total	1075	1053	1707	1087	4922

```
# Conduzindo teste estatístico para ver se as variáveis categóricas tem correlação com nossa variável d
testes=lapply(tabelas_contingencia,chisq.test)
testes_df <- tibble(
  Variaveis = names(testes),
  Estatistica = map_dbl(testes, ~ .x$statistic),
  P_valor = map_dbl(testes, ~ .x$p.value)
)
testes_df=testes_df %>%
  mutate(Var1=str_replace(Variaveis, "\\Qx \\E.*", ""),
         Var2=str_replace(Variaveis, ".*?\\Qx \\E", "")) %>%
  select(-Variaveis)

# Plotando os resultados dos testes estatísticos
testes_df %>%
  ggplot(aes(x = Estatistica, y = reorder(Var2,desc(Estatistica)), fill = Estatistica)) +
  geom_bar(stat="identity") +
  geom_text(aes(label=paste0("P-valor: ",round(P_valor*100,2),"%"),
                hjust=ifelse(Estatistica>500,1.1,-.1),
                color=ifelse(P_valor>=.05,"red","black")),
            size=3)+
  scale_color_identity()+
  scale_fill_gradient(low = "red", high = "green") +
  labs(title="Testes estatísticos das variáveis Categóricas",
       x = "Valor da Estatística do Teste",
       y = "Variável") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



Podemos observar que as colunas gender e PhoneService não são correlacionadas com churn

Análise de variância

```
numericas=names(treino)[sapply(treino, is.numeric)]
teste_aov=aov(
  eval(parse(text=paste0("cbind(",paste(numericas,collapse = ","),")")))~Churn,
  data=treino
)
summary(teste_aov)
```

```
##  Response tenure :
##               Df  Sum Sq Mean Sq F value    Pr(>F)
## Churn          1  393430  393430  753.43 < 2.2e-16 ***
## Residuals   4920 2569140     522
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response MonthlyCharges :
##               Df  Sum Sq Mean Sq F value    Pr(>F)
## Churn          1  174706  174706  198.74 < 2.2e-16 ***
## Residuals   4920 4325101     879
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response TotalCharges :
```

```
##                Df     Sum Sq    Mean Sq F value    Pr(>F)
## Churn           1 1.0555e+09 1055484885  214.96 < 2.2e-16 ***
## Residuals    4920 2.4158e+10    4910211
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Todas as variáveis numéricas são correlacionadas com o churn

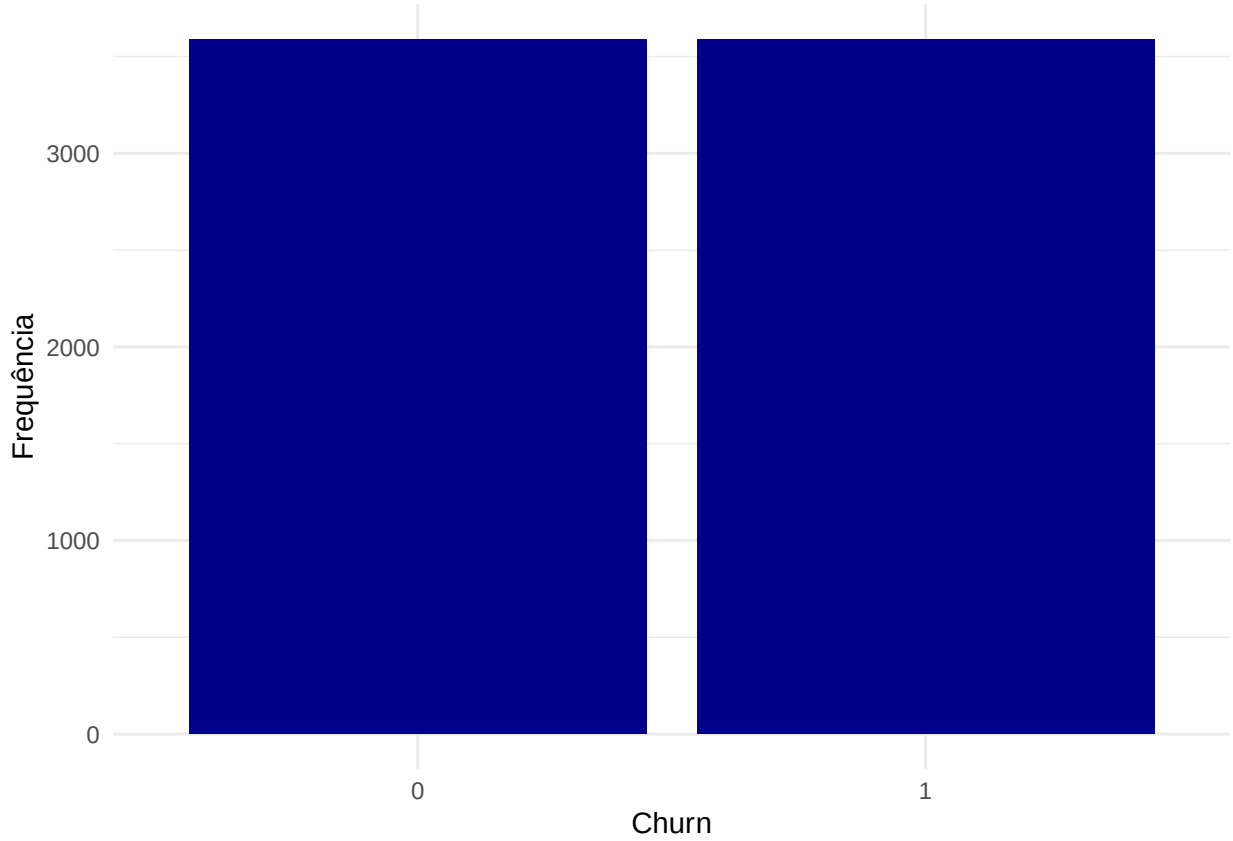
Modelo de Random Forest

Balanceando dados

```
# Como temos dados não balanceados, vamos fazer um oversample no treino:
freqs=summary(as.factor(treino$Churn))
valor_oversample=freqs[1]-freqs[2]

treino_over <- ovun.sample(Churn ~ ., data = treino, method = "over", N = nrow(treino)+valor_oversample)

# Checando se oversample funcionou:
treino_over %>%
  ggplot(aes(Churn))+
  geom_bar(fill="darkblue")+
  labs(x="Churn",y="Frequência")+
  theme_minimal()
```



Ajustando o modelo

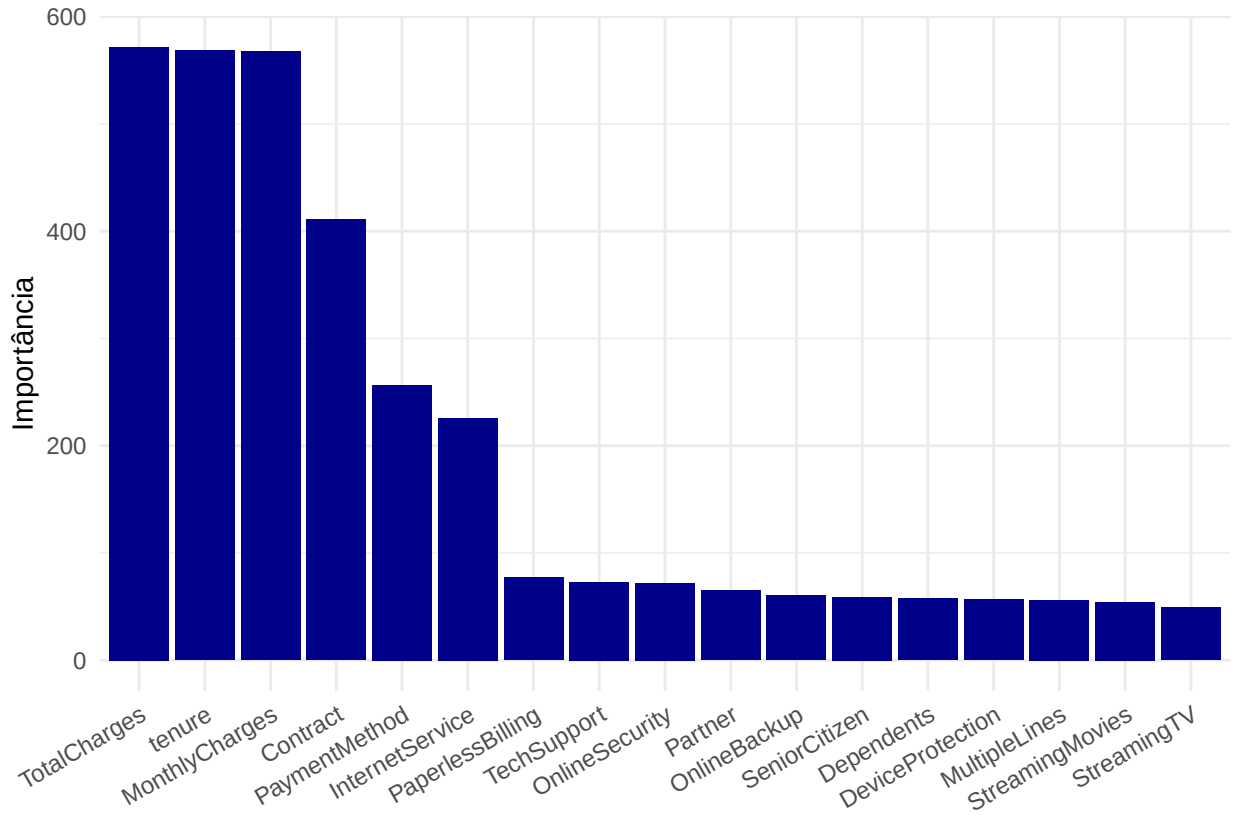
```
# Retirando variáveis que não iremos utilizar (sem correlação significativa)
treino_over=treino_over %>% select(-c(customerID,gender,PhoneService))
teste=teste %>% select(-c(customerID,gender,PhoneService))

# Fazendo o modelo
modelo_rf=randomForest(Churn~.,data=treino_over)

pred <- predict(modelo_rf, newdata = teste)

# Feature importance
importancia <- modelo_rf$importance

tibble(Variavel=rownames(importancia),Importancia=importancia) %>%
  ggplot(aes(reorder(Variavel,desc(Importancia)),Importancia))+
  geom_bar(stat="identity",fill="darkblue")+
  labs(x="",y="Importância")+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 30, hjust = 1))
```



Verificando precisão do Random Forest

```
confusao=table("Predito"=pred,"Observado"=teste$Churn)
confusao=addmargins(confusao)
```

```
rownames(confusao)[3]="Total"
colnames(confusao)[3]="Total"
kable(cbind("",confusao),caption="Matriz de Confusão") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_options
  row_spec(0, bold = TRUE, background = "#D3D3D3") %>%
  column_spec(1, bold = TRUE, background = "#D3D3D3") %>%
  add_header_above(header = c("Predito","", "Observado"=2,""))
```

Table 17: Matriz de Confusão

Predito		Observado		
		0	**1**	**Total**
0		1318	228	1546
1		256	308	564
Total		1574	536	2110

Com esse modelo nós seríamos capazes de identificar 57.46 % dos novos churns, sendo que consideraríamos 26.73 % dos clientes como churn, porém 12.13 % dos clientes seriam identificados como churn erroneamente.

```
TP <- confusao[1, 1]
TN <- confusao[2, 2]
FP <- confusao[2, 1]
FN <- confusao[1, 2]

# Calculando as métricas
accuracy <- (TP + TN) / sum(confusao)
precision <- TP / (TP + FP)
recall <- TP / (TP + FN)
specificity <- TN / (TN + FP)
f1_score <- 2 * (precision * recall) / (precision + recall)

tabela_rf=tibble("Métrica"=c("Acurácia","Precisão","Recall","Especificidade","F1-Score"),
       "Resultado"=paste(round(c(accuracy,precision,recall,specificity,f1_score)*100,2),"%"))
```

Outros Modelos

XGBoost

```
library(xgboost)
treino_over=mutate_at(treino_over,c(1:3,5,7:12,14,18),~as.numeric(as.character(.)))
teste=mutate_at(teste,c(1:3,5,7:12,14,18),~as.numeric(as.character(.)))

treino_dummy <- model.matrix(~ . - 1, data = treino_over)
treino_dummy=as.tibble(treino_dummy)
teste_dummy <- model.matrix(~ . - 1, data = teste)
teste_dummy=as.tibble(teste_dummy)

treino_boost <- xgb.DMatrix(data = as.matrix(select(treino_dummy,-Churn)), label = treino_over$Churn)

params <- list(
  objective = "binary:logistic",
  booster = "gbtree",
```

```
  eval_metric = "logloss",
  eta = 0.1,
  max_depth = 50,
  nrounds = 250
)

modelo_xgb <- xgboost(params = params, data = treino_boost, nrounds = params$nrounds,verbose = 0)
```

```
## [19:16:21] WARNING: src/learner.cc:767:
## Parameters: { "nrounds" } are not used.
```

```
teste_boost <- xgb.DMatrix(data = as.matrix(select(teste_dummy,-Churn)))
pred_boost <- predict(modelo_xgb, teste_boost)

library(pROC)
auc_roc <- auc(roc(response = teste$Churn, predictor = pred_boost))
auc_roc
```

```
## Area under the curve: 0.7998
```

```
# Considerando threshold=0.5
aprox_boost=ifelse(pred_boost>0.5,1,0)


confusao_boost=table("Predito"=aprox_boost,"Observado"=teste$Churn)

confusao_boost=addmargins(confusao_boost)
rownames(confusao_boost)[3]="Total"
colnames(confusao_boost)[3]="Total"
kable(cbind("",confusao_boost),caption="Matriz de Confusão") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_options
  row_spec(0, bold = TRUE, background = "#D3D3D3") %>%
  column_spec(1, bold = TRUE, background = "#D3D3D3") %>%
  add_header_above(header = c("Predito","", "Observado"=2,""))
```

Table 18: Matriz de Confusão

Predito		Observado		
		0	**1**	**Total**
0		1353	258	1611
1		221	278	499
Total		1574	536	2110

Com esse modelo nós seríamos capazes de identificar 57.46 % dos novos churns, sendo que consideraríamos 26.73 % dos clientes como churn, porém 12.13 % dos clientes seriam identificados como churn erroneamente.

```
TP <- confusao_boost[1, 1]
TN <- confusao_boost[2, 2]
FP <- confusao_boost[2, 1]
FN <- confusao_boost[1, 2]

# Calculando as métricas
accuracy <- (TP + TN) / sum(confusao_boost)
precision <- TP / (TP + FP)
```

```
recall <- TP / (TP + FN)
specificity <- TN / (TN + FP)
f1_score <- 2 * (precision * recall) / (precision + recall)

tabela_boost=tibble("Métrica"=c("Acurácia","Precisão","Recall","Especificidade","F1-Score"),
       "Resultado"=paste(round(c(accuracy,precision,recall,specificity,f1_score)*100,2),"%"))
```

Comparando performances

```
tabela_rf %>%
  kable(caption="Métricas de performance do Random Forest") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_options
  row_spec(0, bold = TRUE, background = "#D3D3D3") %>%
  column_spec(1, bold = TRUE, background = "#D3D3D3")
```

Table 19: Métricas de performance do Random Forest

Métrica	Resultado
Acurácia	19.27 %
Precisão	83.74 %
Recall	85.25 %
Especificidade	54.61 %
F1-Score	84.49 %

```
tabela_boost %>%
  kable(caption="Métricas de performance do XGBoost") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE,latex_options
  row_spec(0, bold = TRUE, background = "#D3D3D3") %>%
  column_spec(1, bold = TRUE, background = "#D3D3D3")
```

Table 20: Métricas de performance do XGBoost

Métrica	Resultado
Acurácia	19.32 %
Precisão	85.96 %
Recall	83.99 %
Especificidade	55.71 %
F1-Score	84.96 %