PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

José Hamilton de Oliveira Neto

MODELO PREDITIVO PARA RETENÇÃO DE CLIENTES

Belo Horizonte
2021

José Hamilton de Oliveira Neto

MODELO PREDITIVO PARA RETENÇÃO DE CLIENTES

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Belo Horizonte
2021

SUMÁRIO

1. Introdução

1.1. Contextualização

As empresas que oferecem serviços de telecomunicações, hoje enfrentam uma grande concorrência entre elas. Os clientes estão sempre procurando melhores preços e tarifas.

As empresas tentam agregar o máximo de serviços em seus pacotes, com preços promocionais para atrair novos clientes. Algumas vezes, para os clientes, é mais interessante trocar de fornecedor, uma vez que sempre são ofertados generosos descontos para novos contratos.

Com isso existe uma rotatividade de clientes entre as empresas de telecomunicações.

1.2. O problema proposto

As empresas que oferecem serviços de telecomunicações, como telefonia e internet, hoje enfrentam uma grande concorrência entre elas. Os clientes estão sempre à procura melhores preços e tarifas.

Essas empresas estão sempre buscando novos clientes para manter seu crescimento. Essas empresas investem em diferentes promoções, descontos e facilidade de pagamento para atrair novos clientes. Entretanto, também é importante para elas que consigam reter os clientes atuais.

Os dados analisados nesse trabalho é uma pequena amostrar de uma base de clientes de uma empresa de telecomunicações.

O objetivo desse trabalho é analisar os perfis dos clientes dessa empresa, verificando o perfil dos clientes que realizaram cancelamento. Com isso, esperamos conseguir identificar clientes com potencial de cancelamento. Assim a empresa poderá realizar algum tipo de ação para tentar reter esse cliente.

2. Coleta de Dados

O dataset foi baixado do site Kaggle (https://www.kaggle.com/radmirzosimov/telecom-users-dataset) no dia 29/03/2021.

Abaixo a descrição do dataset:

Nome da coluna/campo	Descrição	Tipo
CustomerID	Código do cliente	Texto
gender	Gênero do cliente	Texto
SeniorCitizen	Indica se o cliente já aposentado	Inteiro (0 ou 1)
Partner	Indica se o cliente é casado	Texto (Yes ou No)
tenure	Número de meses que o cliente está com a empresa	Inteiro
PhoneService	Indica se o cliente possui o serivço telefônico contratado	Texto (Yes ou No)
MultipleLines	Indica se o cliente possui múltiplas linhas telefônicas contratadas	Texto (Yes, No ou No phone service)
InternetService	Tipo de serviço de internet contratado	Texto (DSL, Fiber optic, No)
OnlineSecurity	Indica se o serviço de backup online está ativo	Texto (Yes, No, No internet service)

DeviceProtection	Indica se o cliente possui algum equipamento de internet segurado	Texto (Yes, No, No internet service)
TechSupport	Indica se o cliente possui acesso ao suporte técnico da internet	Texto (Yes, No, No internet service)
StreamingTV	Indica se o cliente possui streaming de TV	Texto (Yes, No, No internet service)
StreamingMovies	Indica se o cliente possui o serviço de streaming de filmes	Texto (Yes, No, No internet service)
Contract	Tipo do contrato	Texto (Month-to-month, One year, Two year)
PaperlessBilling	Indica se o cliente optou por não receber a fatura impressa	Texto (Yes ou No)
PaymentMethod	Tipo de pagamento	Texto (Electronic check, Mailed check, Bank transfer (automatic), Credit card (automatic))
MonthlyCharges	Valor Mensal	Número
TotalCharges	Total já pago pelo cliente	Número
Churn	Indica se houve rotatividade	Texto (Yes ou No)

3. Processamento/Tratamento de Dados

3.1. Ferramentas utilizadas

Utilizaremos o Python para manipulação do dataset. Python é uma linguagem de programação amplamente utilizada em ciência de dados. Juntamente com Python iremos utilizar a biblioteca Pandas para facilitar a manipulação dos dados.

Para facilitar a exploração dos dados utilizaremos o Jupyter Notebook. Ele simplifica os testes sobre o dataset.

Após o carregamento do dataset, realizamos um comando para descrever o básico sobre os dados contido nele.

3.2. Tratamento dos dados

Inicialmente executamos o comando para verificar a quantidade de registros que o dataset possui. Conforme imagem abaixo, o dataset possui 5986 registros.

```
In [21]: df.shape
Out[21]: (5986, 22)
```

Também é importante verificar se o dataset possui valores nulo. Essa verificação é importante, pois dependendo do que iremos analisar, teremos que realizar algum tipo de tratamento. Após executar o código abaixo, foi verificado que o dataset não possui nenhum valor nulo.

```
In [16]: df.isnull().values.any()
Out[16]: False
```

Apensar do dataset não apresentar nenhum valor nulo. Em uma análise campo a campo, identificamos que um valor numérico está vazio.

```
for col in df[num_columns].columns:
    print('\n ')
    print('Coluna: ',col, 'Valores: ', df[col].min(), df[col].max())
```

```
Coluna:  TotalCharges Valores:    999.9

Coluna:  MonthlyCharges Valores:  18.25 118.75

Coluna:  tenure Valores:  0 72
```

Como vemos na imagem acima, a coluna *TotalCharges* possui valores vazios. Inicialmente iremos substituir esses valores vazios por 0. Conforme comandos abaixo.

```
df['TotalCharges'] =  df['TotalCharges'].replace(' ', 0)
df['TotalCharges'] =  df['TotalCharges'].astype(float)
```

```
for col in df[num_columns].columns:
    print('\n ')
    print('Coluna: ',col, 'Valores: ', df[col].min(), df[col].max())
```

```
Coluna:  TotalCharges Valores:  0.0 8684.8

Coluna:  MonthlyCharges Valores:  18.25 118.75

Coluna:  tenure Valores:  0 72
```

Agora que convertemos a coluna para o tipo *Float,* podemos calcular as médias para substituir os valores "0", pelas médias para diminuir a distorção.

Calculando médias.

```
print('TotalCharges média:', df['TotalCharges'].mean())
print('tenure média:', df['tenure'].mean())
```

```
TotalCharges média: 2294.2215586368193
tenure média: 32.46876044102907
```

Atualizando campos.

```
df['TotalCharges'] =  df['TotalCharges'].replace(0, 2298.06)
df['tenure'] =  df['tenure'].replace(0,29)
```

Como o objetivo desse trabalho é tentar prever se um cliente irá cancelar ou não o contrato, para facilitar o trabalho de classificação, iremos criar uma nova coluna para normalizar a coluna *Churn.* Iremos classificar o *yes* como 1 e *no* como 0. Iremos executar os seguintes comandos para discretizar esse dado.

```
churns = df['Churn'] == 'Yes'

def churn_0_or_1(value):
    if (value == 'Yes'):
        return 1
    else:
        return 0


df['churn_i'] = df['Churn'].apply(churn_0_or_1)

df.head()
```

InternetService	...	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn	churn_i
No	...	No internet service	No internet service	No internet service	Two year	No	Credit card (automatic)	24.10	1734.65	No	0
Fiber optic	...	No	Yes	No	Month-to-month	Yes	Credit card (automatic)	88.15	3973.2	No	0
Fiber optic	...	No	No	No	Month-to-month	Yes	Bank transfer (automatic)	74.95	2869.85	Yes	1
DSL	...	No	No	Yes	Month-to-month	Yes	Electronic check	55.90	238.5	No	0
DSL	...	No	No	No	Month-to-month	No	Electronic check	53.45	119.5	No	0

4. Análise e Exploração dos Dados

Como citado a seção anterior, utilizamos o Jupyter Notebook para exploração dos dados. Como objetivo é tentar prever se um cliente irá ou não trocar de empresa e não possuímos dados relacionados a satisfação ou qualidade dos serviços, resolvemos analisar dados relacionados aos valores e pagamentos.

4.1. Analisando tempo de contrato

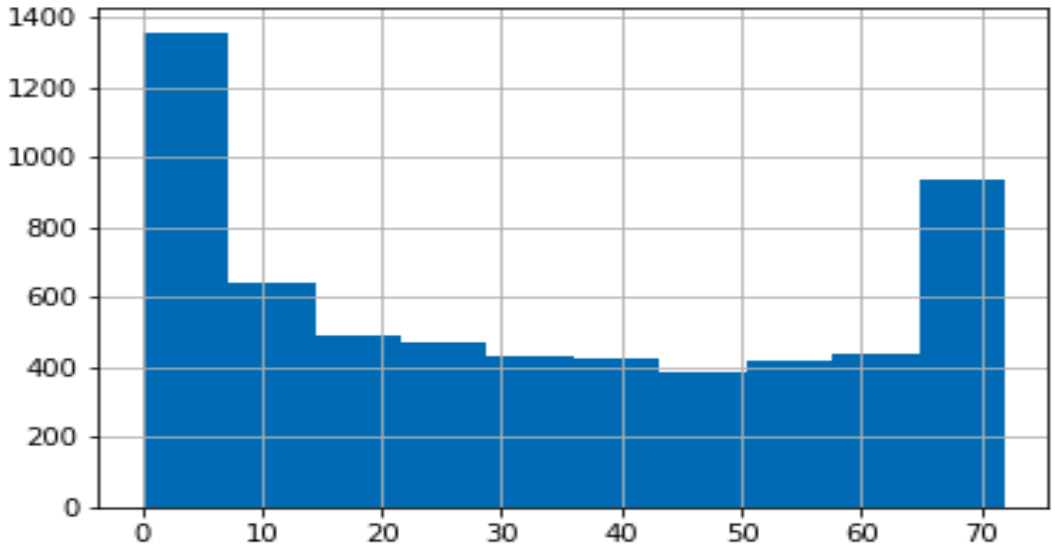
Nessa análise nosso objetivo é entender se existe correlação entre o tempo de contrato ou a quanto tempo o cliente está com a empresa com a probabilidade desse cliente trocar de empresa ou não.

Analisando inicial o campo ternure, esse campo conforme descrição dos dados, informa a quantidade de meses que o cliente está com empresa.

```
In [5]: df['tenure'].describe()

Out[5]: count    5986.000000
        mean       32.468760
        std        24.516391
        min         0.000000
        25%         9.000000
        50%        29.000000
        75%        56.000000
        max        72.000000
        Name: tenure, dtype: float64
```

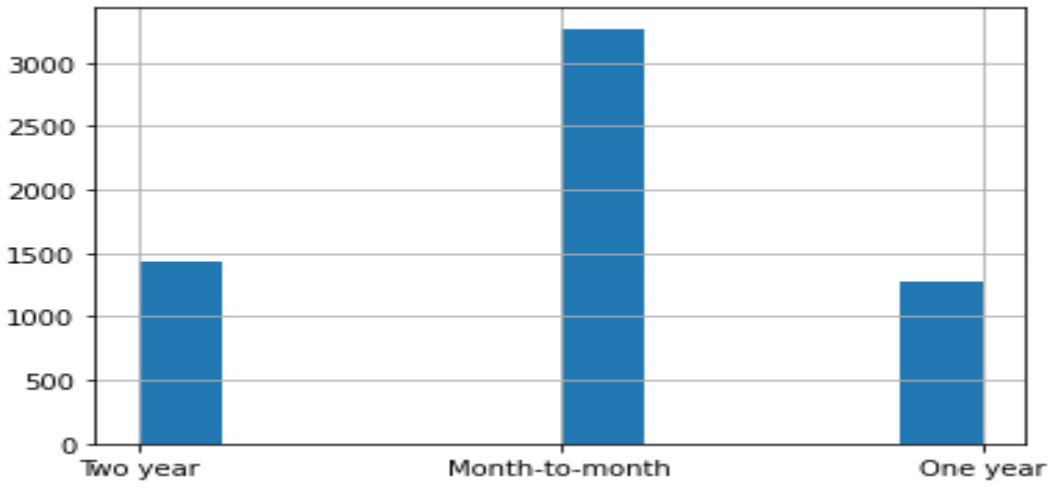
Conforme a imagem acima, vemos que a média de tempo que cliente permanece na empresa é de aproximadamente 32 meses. Os clientes mais antigos estão com a empresa a 72 meses.

O histograma da imagem abaixo nos ajuda a entender melhor a distribuição dos clientes pelo tempo que já são cliente.



Analisando o histograma, vemos uma quantidade maior de clientes nos extremos. Temo muitos clientes com menos de 10 meses e uma boa quantidade com pouco mais de 60 meses. Isso por indicar, por exemplo, que a empresa deve ter algum tipo de contato para retenção do cliente após 12 meses de contrato.

Agora iremos analisar o campo Contract, esse campo informa quanto tempo de contrato o cliente possui junto a empresa.



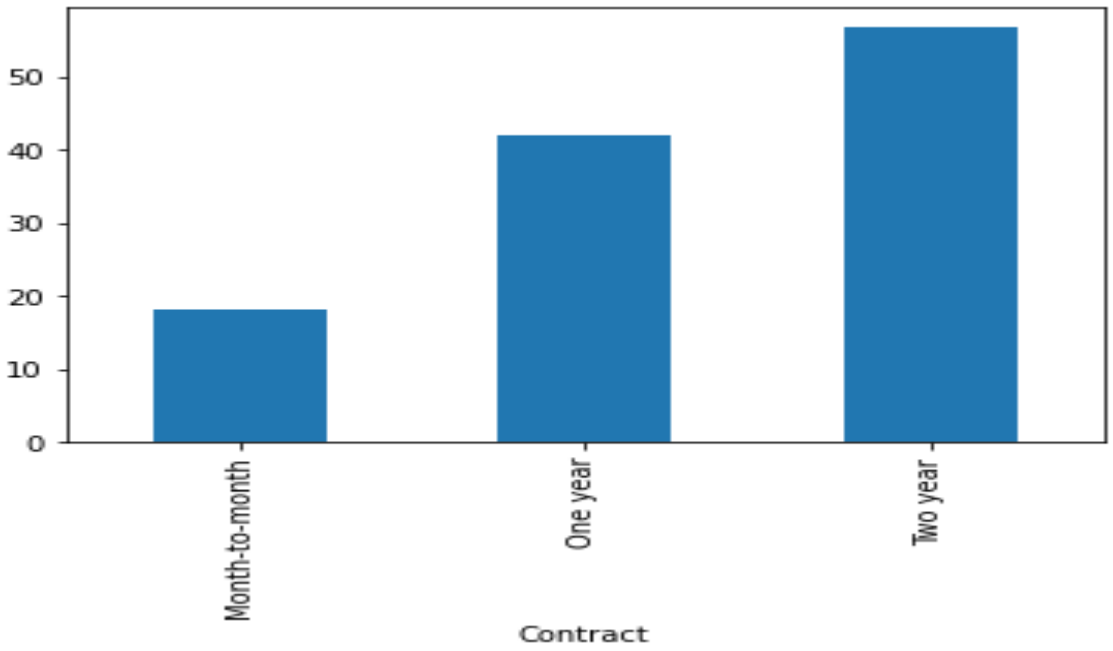
Analisando o histograma acima, verificamos que a maior parte dos contratos são do tipo mensal, ou seja, podem ser cancelados a qualquer momento. Talvez esse tipo de cliente necessite de um acompanhamento melhor a fim de evitar o cancelamento do contrato.

Para complementar a análise dos dados relacionados ao tempo de contrato e a quanto tempo o cliente está com empresa, resolvemos verificar a relação entre o tempo de contrato e o tempo que o cliente está ativo.

Executando o comando abaixo, iremos agrupar o dataset pelo tempo de contrato e mostraremos a média de tempo que cliente permanece com a empresa.

```
df.groupby(["Contract"])["tenure"].mean().plot.bar()
```

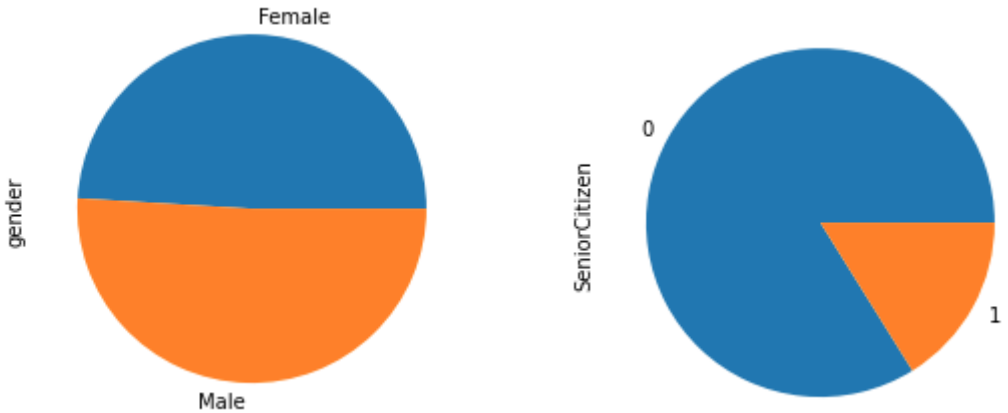
Obtivemos o seguinte gráfico:

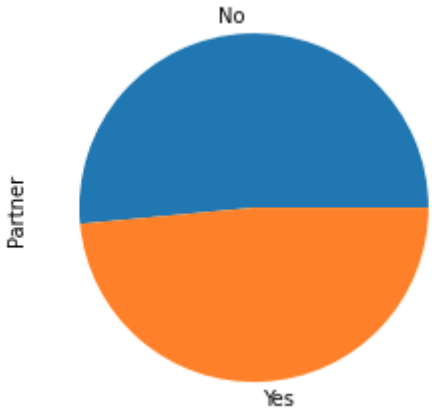


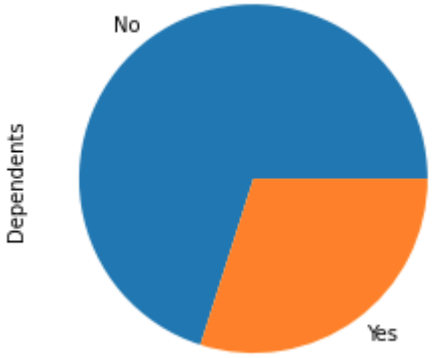
Como podemos verificar no gráfico acima, clientes que possuem contratos mensais tem uma tendência maior de ficar menos tempo fidelizado.

4.2. Analisando do perfil clientes

O objetivo dessa análise e identificar o perfil de cliente da empresa. Serão utlizados para essa análise os campos *gender* (gênero), *SeniorCitizen* (se é ou não aposentado), *Partner* (se é ou não casado), *Dependents* (indica se tem dependentes). Inicialmente vamos apenas fazer um agrupamento de totalização desses campos.







Analisando os gráficos gerados, verificamos que os campos *gender* e *Partner* estão praticamente divididos na metade. Já para o campo *SeniorCitizen* a empresa possui muito mais clientes não aposentados do que aposentados. Com relação aos dependentes a empresa possui muito mais clientes sem dependentes do que com dependente.

Com os resultados apresentados acima, o principal tipo de cliente da empresa são pessoas que não estão aposentadas e que não possuem dependentes.

4.3. Analisando produtos contratados

Agora analisaremos quais são os principais serviços contratos pelos clientes junto a empresa. A empresa possui dois tipos principais de produtos, telefonia e internet. Cada produto desse pode possuir subprodutos associados. Por exemplo, quando cliente possui o serviço de internet contratado, ele pode ter também o serviço de *backup online.*

Executamos os comandos abaixo para obter os resultados.

```
df[df['PhoneService']=='Yes']['PhoneService'].count()
```

```
5396
```

```
df[df['InternetService']!='No']['InternetService'].count()
```

```
4695
```

```
df[(df['PhoneService']=='Yes') & (df['InternetService']!='No')]['PhoneService'].count()
```

```
4105
```

Com isso temos um total de 5396 clientes que possuem pelo menos o serviço telefônico, 4695 que possuem pelo menos o serviço de internet e 4105 que possuem ambos os serviços.

4.4. Analisando dados sobre pagamento

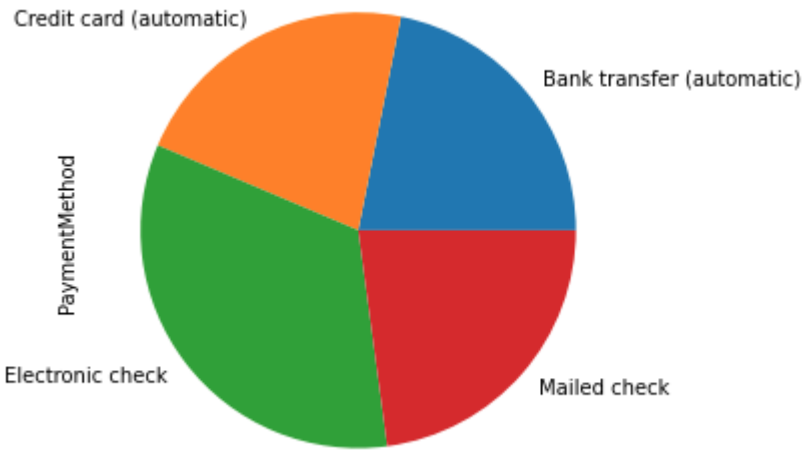
Primeiro vamos saber qual é o ticket médio dos clientes, ou seja, qual o valor médio mensal de cada contrato. Executando o comando abaixo:

```
df['MonthlyCharges'].mean()
```

```
64.80221349816237
```

Como vemos, o valor médio das mensalidades é de aproximadamente 64 dólares. Vamos verificar também os valores pagamos mensalmente com a informação de o cliente cancelou o não o contrato.

Outra informação importante é a distribuição de como essa mensalidade é paga. O gráfico abaixo mostra esse comportamento.



4.5. Correlacionando dados

O principal objetivo da análise desse dataset é tentar identificar se um cliente irá ou não cancelar o contrato junto a empresa. Antes criar o modelo de Machine Learning, é importante tentarmos identificar relação entre os dados. Isso irá nos ajudar

a identificar campos que estejam mais fortemente relacionados ao cancelamento ou não dos contratos.

Analisando inicialmente os clientes que cancelaram os contratos, escolhemos dois campos númericos *(ternure* e *MonthlyCharges*). Temos.

	count	mean	std	min	25%	50%	75%	max
tenure	1587.0	18.246377	19.667262	1.00	2.000	10.0	30.0	72.00
MonthlyCharges	1587.0	74.164871	24.965002	18.85	55.675	79.5	94.4	118.35

E agora os clientes que não cancelaram.

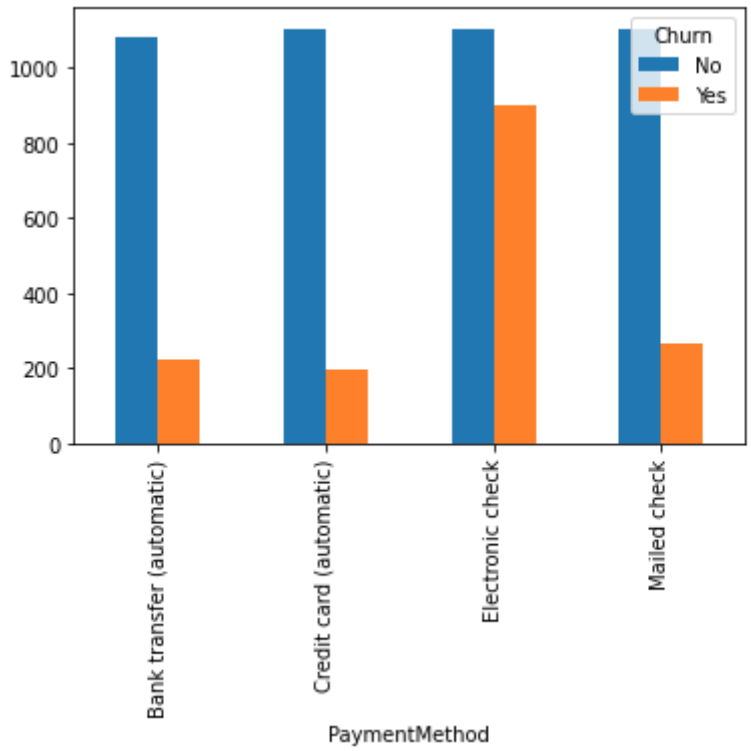
	count	mean	std	min	25%	50%	75%	max
tenure	4399.0	37.599682	24.065131	0.00	15.000	37.00	61.0	72.00
MonthlyCharges	4399.0	61.424506	31.086101	18.25	25.125	64.75	88.7	118.75

Em uma rápida análise verificamos que o tanto o valor médio como a mediana são maiores para os clientes que cancelaram o contrato do que os que não cancelaram.

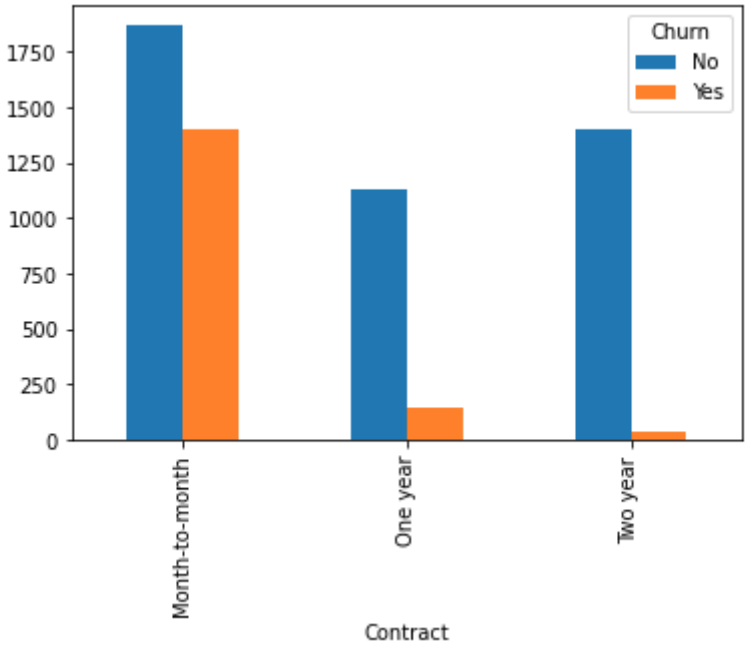
Outra informação que é relevante para nossa análise é a correlação entre as formas de pagamento (*PaymentMethod*) e o cancelamento de contratos. A tabela abaixo mostra que existe uma distribuição uniforme entre as diferentes formas de pagamentos ofericidos pela empresa. Entretanto clientes que optaram por pagar pelo *Eletronic check* tem uma tendência maior de cancelar os contratos do que os clientes que optaram por outros meios.

Churn	No	Yes
PaymentMethod		
Bank transfer (automatic)	1084	224
Credit card (automatic)	1105	198
Electronic check	1104	902
Mailed check	1106	263

O gráfico abaixo mostra melhor essa tendência.



Já analisando os dados do tipo de contrato assinado pelo cliente com o cancelamento do mesmo. Foi verificado que apesar de os clientes que optaram pelo contrato mensal apresentar o maior número de cancelamento, eles também representam a maior parte dos contratos, conforme gráfico abaixo:



4.6. Analisando variável de resposta

Nossa variável de reposta é flag yes/no (sim ou não), sendo assim iremos trabalhar com uma classificação binária. Em nossos dados, consideramos que as amostras (clientes) que irão cancelar o contrato (Churn 'yes') pertencem à classe positiva, enquanto os clientes que não irão cancelar ficarão pertencentes à classe negativa.

Quando trabalhamos como um problema de classificação binária, temos qua analisar qual é a proporção da classe positiva? Na imagem abaixo temos essa resposta.

```
df.groupby('Churn')['customerID'].count()
```

```
Churn
No     4399
Yes    1587
Name: customerID, dtype: int64
```

Para facilitar a análise e como estamos trabalhando com um problema de classificação binária, durante o processo de tratamento dos dados, realizamos a discretizarão da coluna *Churn.* Agora iremos analisar pela colina *churn_i.*

```
df['churn_i'].mean()
```

```
0.2651186100902105
```

Sendo a variável alvo é 1 ou 0, a obtenção dessa média indica que aproximadamente 26% dos clientes cancelaram o contrato, dentro do conjunto analisado. Essa proporção da classe positiva dentro da amostra, também chamada de fração de classe, é uma importante estatística.

Quando trabalhamos com classificação binária, podemos classificar os dataset como balanceados ou desbalanceados. Em um cenário ideal o dataset deveria ter uma divisão 50/50, na prática isso raramente acontece.

Embora nosso dataset não esteja balanceado, fração de classe positov de 26% não necessariamente é um desbalanceamento. Por tomar como exemplo, a área de detecção de fraudes, elas trabalhando com frações de classe positivas na orderm de 1% ou menos.

5. Criação de Modelos de Machine Learning

Agora iremos testar alguns modelos de Machine Learning, para encontrar aquele que melhor se adapta ao nosso problema.

Para a criação do nosso modelo de Machine Learning, iremos usar o scikit-learn. Esse é um dos principais pacotes de Machine Learning para Python. O scikit-learn possui uma ampla variedade de abordagens para classificação e regressão, e aprendizado não supervisionado.

5.1. Classificação

A classificação é um método de aprendizagem supervisionada. A aprendizagem supervisionada implica que temos rótulos para classificação ou números para regressão, os quais o algoritmo deve aprender.

Os dados de entrada podem ser divididos em dois grupos: X, com os atributos a serem utilizados na determinação da classe de saída e Y, que representa a classe de saída (o atributo para o qual se deseja fazer a predição do valor da classe), sendo que em problemas de Classificação, o Y é sempre categórico.

O fluxo basicamente seria, a partir de uma base de dados rotulada, criamos dois subconjuntos: base de treino (cerca 70% dos dados originais) e a base de teste (contento o restante dos dados). A base de treino é submetida ao modelo (classificador) para calibração. Em seguida, ocorre a fase de predição utilizando a base de testes.

Temos algumas medidas para estimar o desempenho de um classificador. A acurácia é uma das mais utilizadas, que representa o percentual de acerto do classificador. Outra métrica bastante utilizada para problemas de classificação é a matriz de confusão, que oferece um detalhamento do desempenho do modelo de classificação, mostrando, para cada classe, o número de classificações corretas em relação ao número de classificações preditas pelo modelo.

5.2. Regressão

A regressão é um método de aprendizagem supervisionada. É semelhante à classificação, mas, em vez de prever um rótulo, tentamos prever um valor contínuo.

Assim como na classificação, a regressão consiste em realizar aprendizado supervisionado a partir de dados históricos. Além do tipo do resultado de saída do modelo, os dois problemas também se diferem quanto às métricas utilizadas para a avaliação de saída.

O coeficiente de determinação é uma métrica de regressão comum. Em geral, esse valor está entre 0 e 1 e representa o percentual da variância do alvo com o qual os atributos contribuem. Valores maiores são melhores, entretanto é difícil avaliar o modelo exclusivamente a partir dessa métrica.

O erro médio absoluto expressa o erro de previsão médio absoluto do modelo. Um modelo perfeito teria um valor igual a 0, mas essa métrica não tem limites superiores, diferente do coeficiente de determinação.

5.3. Algoritmos

5.3.1. Árvore de Decisão

Uma árvore de decisão usa uma estrutura de árvore para representar um número de possíveis caminhos de decisão e um resultado para cada caminho. As árvores de decisão possuem muitas recomendações. Elas são muito fáceis de entender e interpretar, e o processo por onde chegam numa previsão é completamente transparente.

Ao mesmo tempo, encontrar a árvore de decisão perfeita para um conjunto de dados em treinamento é um problema difícil. Temos que tomar para não construirmos árvores que são sobre ajustadas aos dados de treinamento e que não generalizem bem para dados desconhecidos.

Uma forma de evitar esse ajuste quase perfeito aos dados de treinamento é a utilização das chamadas florestas aleatórios.

5.3.2. Regressão Logística

Apesar do nome, é um algoritmo utilizado exclusivamente para problemas de Classificação, mas seu funcionamento lembra muito o funcionamento do algoritmo de Regressão Linear.

A Regressão Logística é usada para estimar valores discretos de classes binárias (valores como 0/1, sim/não, verdadeiro/falso) com base em um conjunto de variáveis independentes. Internamente, a Regressão Logística calcula a probabilidade de ocorrência de um evento, ajustando os dados a uma função logit, uma função que mapeia a saída em valores entre 0 e 1.

A Regressão Logística modela a probabilidade da classe padrão do problema.

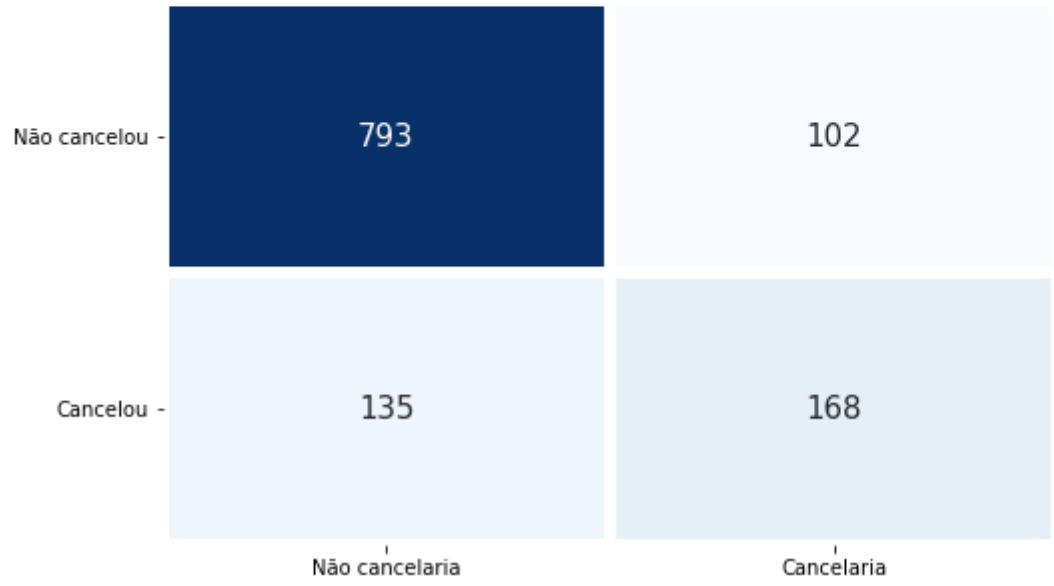
5.4. Aplicação dos modelos

Agora iremos aplicar os modelos descritos anteriormente para analisarmos os resultados.

Primeiro iremos aplicar a regressão logística.

```
# REGRESSÃO LOGISTICA
lg = LogisticRegression(random_state = 42)
lg.fit(X_train, y_train)
y_pred = lg.predict(X_test)
y_prob = lg.predict_proba(X_test)[:,1]

# Metricas
print(classification_report(y_test, y_pred))
print("Acuracia", metrics.accuracy_score(y_test, y_pred))

# Matriz de confusão
lg_cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize = (8, 5))
sns.heatmap(lg_cm, cmap = 'Blues', annot = True, fmt = 'd', linewidths = 5, cbar = False,
            annot_kws = {'fontsize': 15}, yticklabels = ['Não cancelou', 'Cancelou'],
            xticklabels = ['Não cancelaria', 'Cancelaria'])
plt.yticks(rotation = 0)
plt.show()
```

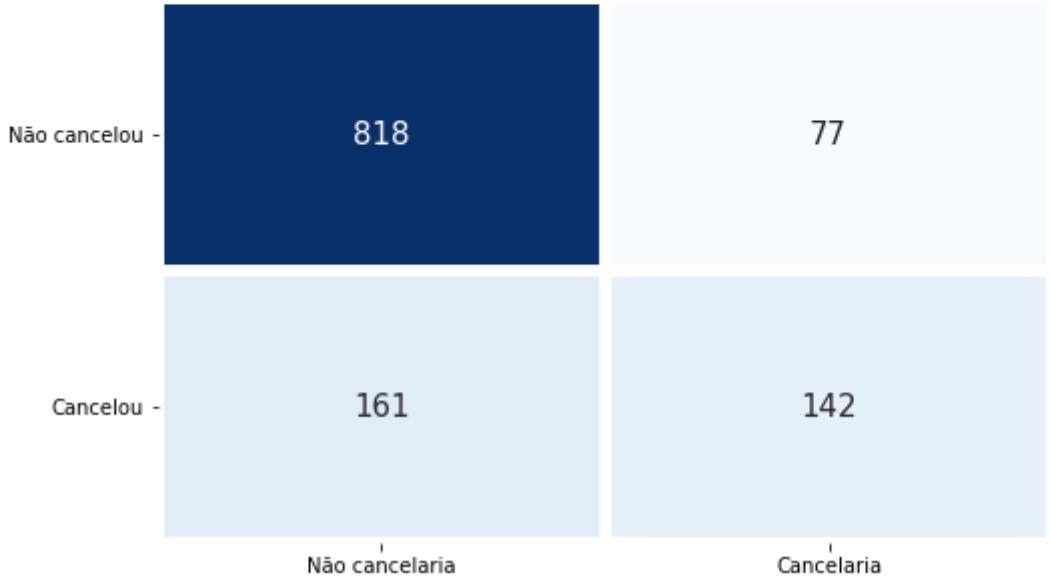
```
              precision    recall  f1-score   support

           0       0.85      0.89      0.87       895
           1       0.62      0.55      0.59       303

    accuracy                           0.80      1198
   macro avg       0.74      0.72      0.73      1198
weighted avg       0.80      0.80      0.80      1198

Acuracia 0.8021702838063439
```



Agora vamos executar a árvore aletóaria.

```
# Floresta aletória

rf = RandomForestClassifier(random_state = 22, max_depth = 5)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)
y_prob = rf.predict_proba(X_test)[:,1]

# Metricas
print(classification_report(y_test, y_pred))
print("Acuracia", metrics.accuracy_score(y_test, y_pred))

plt.figure(figsize = (8, 5))
sns.heatmap(rf_cm, cmap = 'Blues', annot = True, fmt = 'd', linewidths = 5, cbar = False,
            annot_kws = {'fontsize': 15}, yticklabels = ['Não cancelou', 'Cancelou'],
            xticklabels = ['Não cancelaria', 'Cancelaria'])
plt.yticks(rotation = 0)
plt.show()
```

```
              precision    recall  f1-score   support

           0       0.84      0.92      0.87       895
           1       0.65      0.47      0.55       303

    accuracy                           0.80      1198
   macro avg       0.74      0.69      0.71      1198
weighted avg       0.79      0.80      0.79      1198

Acuracia 0.8030050083472454
```



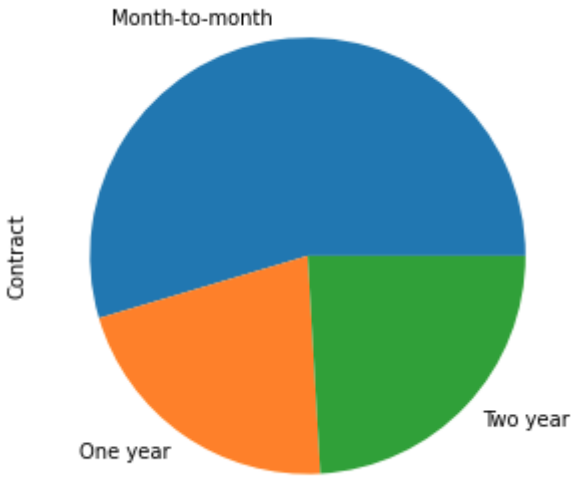
Percebemos que a floresta aleatória possui uma acurácia um pouco melhor do que a regressão logística.

6. Apresentação dos Resultados

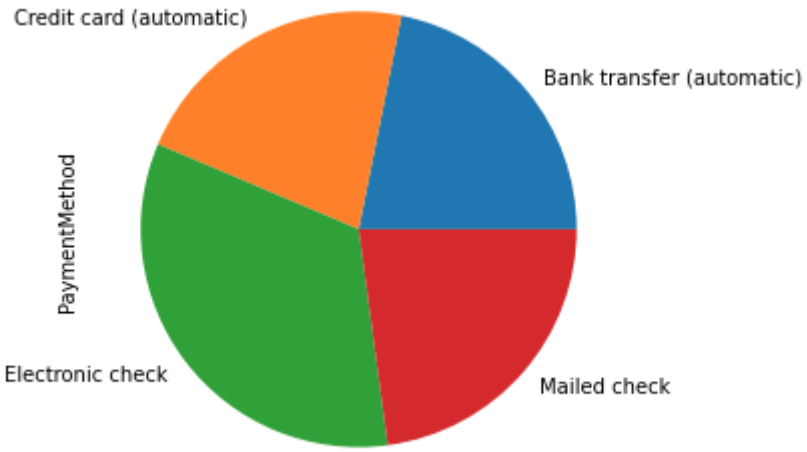
O principal objetivo desse trabalho é conseguir prever se um cliente irá ou não cancelar o contrato junto a empresa.

Iniciamos tentando entender os dados que iremos trabalhar. Nessa seção você deve apresentar os resultados obtidos. Durante o processo de análise e exploração dos dados identificamos algumas informações interessantes.

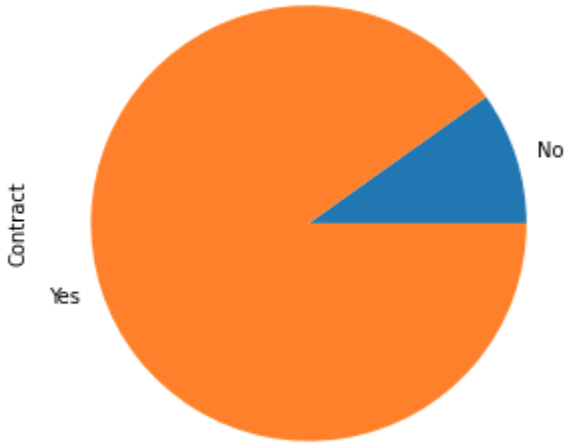
Distribuição dos clientes pelo tempo de contrato.



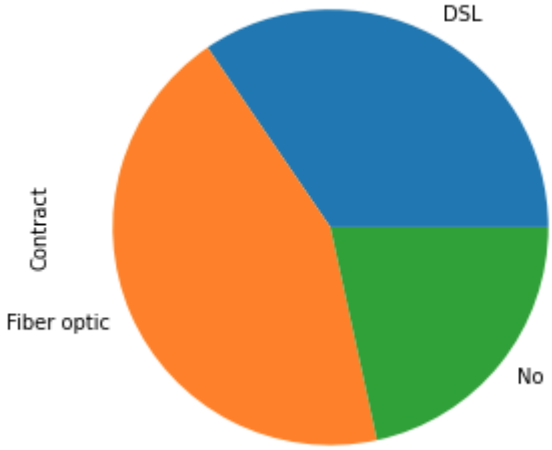
Distribuição dos clientes pela forma de pagamento.



Clientes que possuem serviço de telefonia.

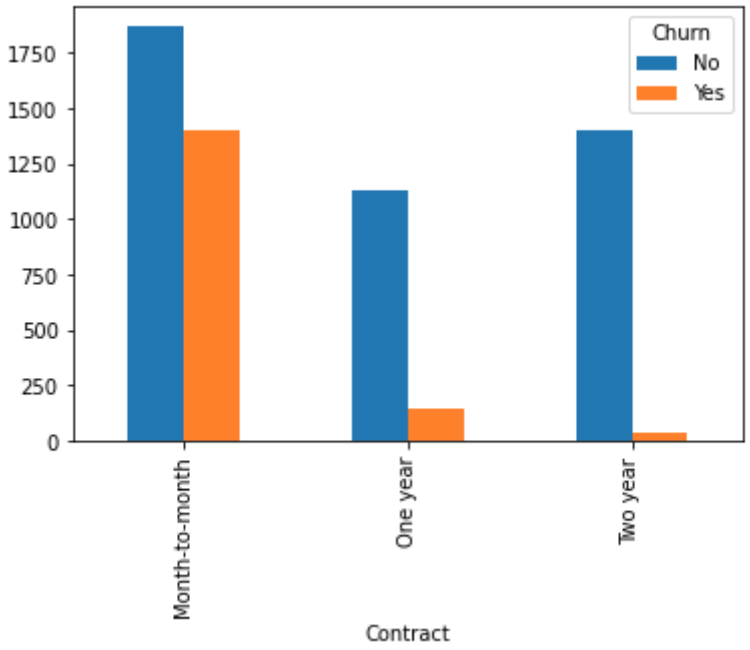


Clientes que possuem serviço de internet.



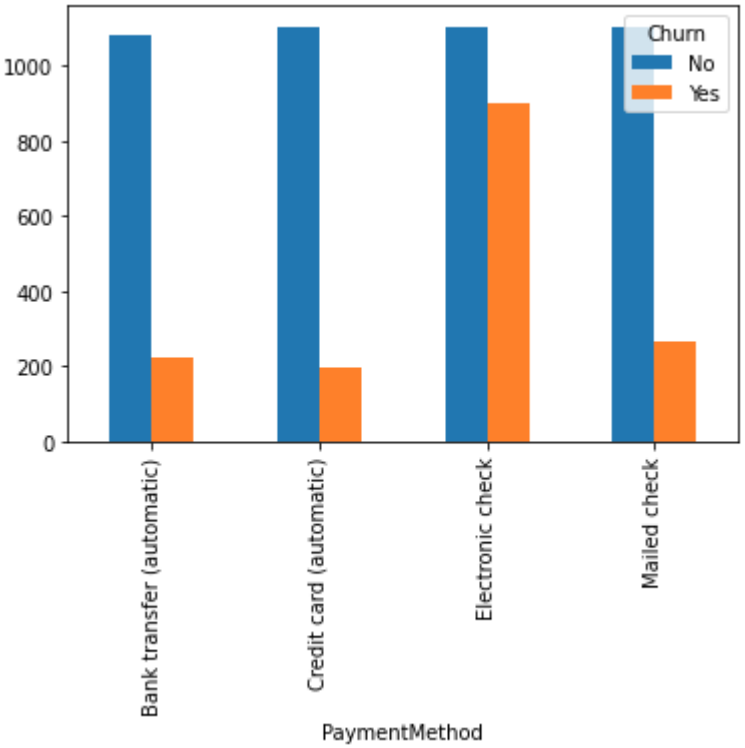
Agora que temos algumas informações sobre a distribuição dos clientes por algumas informações, iremos analisar a relação como cancelamento ou não dos contratos.

Primeiro iremos analisar o cancelamento com o tempo de contrato assinado pelo cliente.



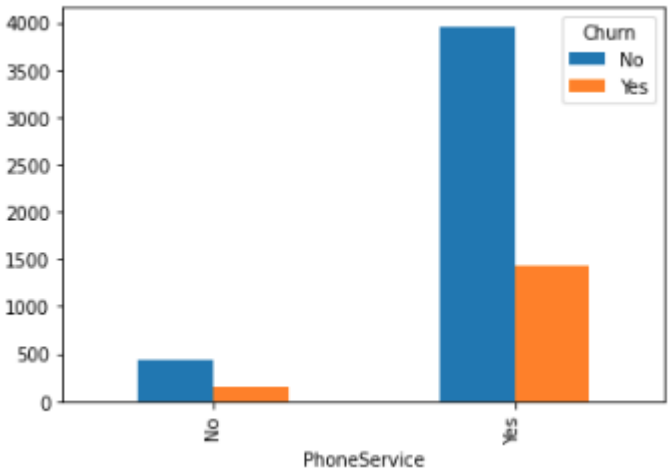
De acordo com o gráfico acima apresentado, vemos que cliente com contrato mensal (*month-to-month*), possuem mais cancelamentos do que as outros tempos de contrato. De certa forma isso faz sentido, pois devem existir algum tipo de penalidade para cliente com contratos mais longo. Isso deve dificularo o cancelamento dos mesmos.

Agora iremos fazer a mesma comparação utilizando a forma de pagamento selecionada pelo cliente.



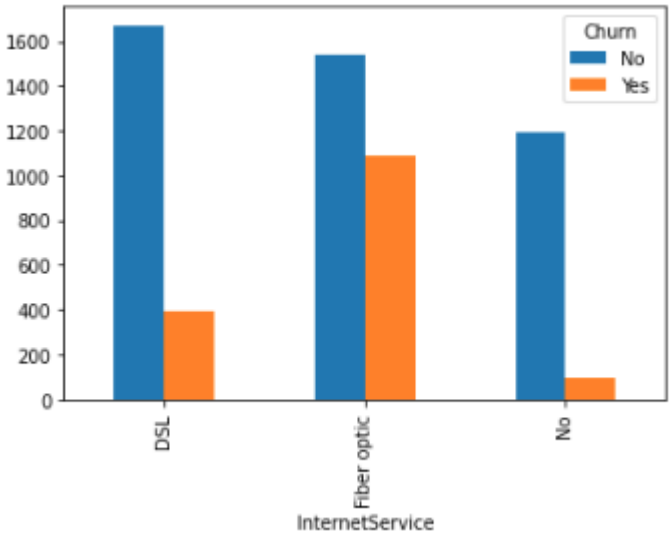
Pelo gráfico acima apresentado, podemos ver que clientes que optam pelo *electronic check* apresentam um maior número de cancelamentos. Temos duas formas de pagamento que são automáticas (*Bank transfer* e *Credit card*), isso de alguma maneira pode interferir de forma positiva para a empresa, uma vez que o cliente não é "lembrado" de realizar o pagamento.

Analisaremos agora o cancelamento em relação aos clientes que possuem pelo menos o serviço de telefonia contratado.



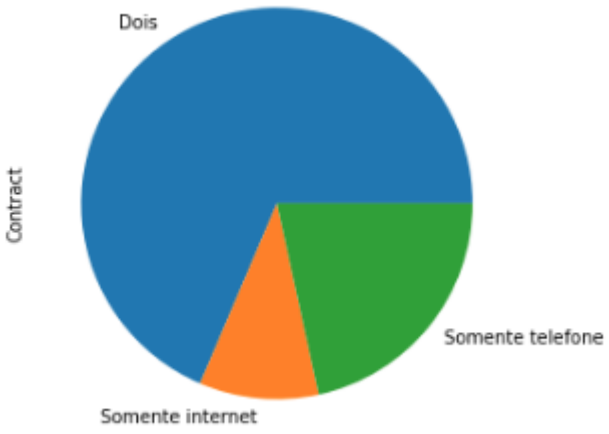
O gráfico não indica muita coisa, uma vez que temos muito mais clientes que possuem o serviço, por consequência termos muito mais cancelamentos.

Agora em relação ao serviço de internet.

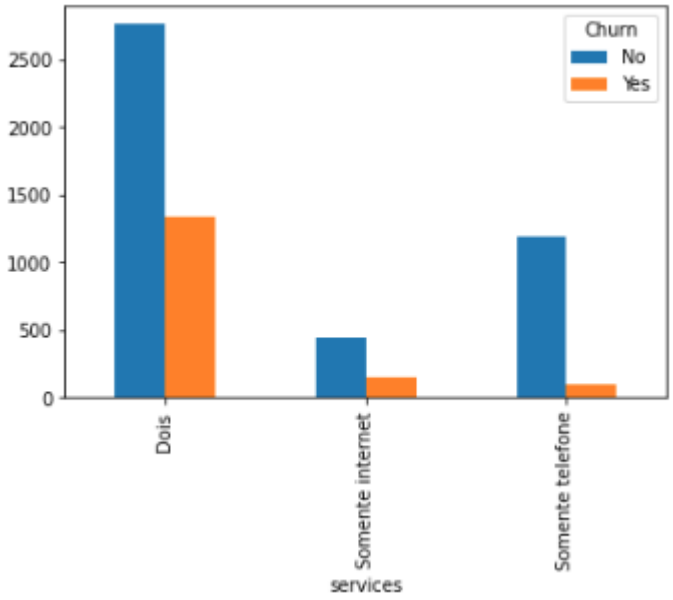


Pelo gráfico apresentado acima, destacaria a diferença de cancelamentos entre cliente que possuem internet DSL e fibra ótica (*fiber optic*). Isso talvez indique que o serviço de fibra ótica da empresa possui um preço maior que a dos concorrentes ou um serviço de qualidade inferior.

Outra análise que é interessante, seria analisar clientes que possuem os dois serviços e não somente um deles. Isso indicaria se caso o cliente possua mais de um serviço contrato, a continuidade do mesmo junto a empresa seria mais certa.



O gráfico mostra que a maior parte dos clientes possuem os dois serviços contratados. Agora vamos tentar associar essa informação ao cancelamento.



Como a maioria dos clientes possuem os dois serviços contratados, esse grupo também pela maioria dos cancelamentos. Destaco somente que o número de cancelamentos de clientes que possuem somente o serviço de internet contrato é maior que o número de cancelamento de clientes que possuem somente o serviço de telefone. Isso pode ocorrer devido à falta de qualidade ou ao preço mais elevado em relação aos concorrentes do serviço de internet. O que pode ocorrer também, que que clientes não querem perder seu número de telefone ou passar pelo processo de migração de operadora. Então, talvez, fosse interessante para empresa fidelizar os clientes através do serviço de telefonia.

7. Links

Link para o vídeo: https://youtu.be/z9s30HsLF80

Link para o repositório: https://github.com/jhoneto/pucminas

REFERÊNCIAS

GRUS, Joel; traduzido por Wellington Nascimento. **Data Science do Zero.** Rio de Janeiro: Alta Books, 2016.

HARRISON, Matt; traduzido por Lúcia A. Kinoshita. **Machine Learning Guia de Referência Rápida.** São Paulo: O'Reilly, 2020.

CORRÊA, Eduardo. **Pandas Python.** São Paulo: Casa do Código, 2019.

APÊNDICE

Programação/Scripts

Exploração

```
#!/usr/bin/env python
# coding: utf-8

import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report, confusion_ma-
trix, roc_curve, auc
from sklearn.metrics import roc_auc_score, precision_score, re-
call_score, f1_score
from sklearn.ensemble import RandomForestClassifier

df = pd.read_csv('./telcom/telecom_users.csv')
df.head()
df.shape
df.isnull().values.any()
cat_columns = ['gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'PhoneService', 'MultipleLines', 'InternetService', 'Online-
Security',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'Streamin-
gTV',
       'StreamingMovies', 'Contract', 'PaperlessBil-
ling', 'PaymentMethod',
       'Churn']
num_columns = ['TotalCharges','MonthlyCharges','tenure']

for col in df[cat_columns].columns:
    print('\n ')
    print('Coluna: ',col, 'Valores: ', df[col].unique())

for col in df[num_columns].columns:
```

```
    print('\n ')
    print('Coluna: ',col, 'Valores: ', df[col].min(), df[col].max())

df['TotalCharges'] =  df['TotalCharges'].replace(' ', 0)
df['TotalCharges'] =  df['TotalCharges'].astype(float)

for col in df[num_columns].columns:
    print('\n ')
    print('Coluna: ',col, 'Valores: ', df[col].min(), df[col].max())

print('TotalCharges média:', df['TotalCharges'].mean())
print('tenure média:', df['tenure'].mean())

df['TotalCharges'] =  df['TotalCharges'].replace(0, 2298.06)
df['tenure'] =  df['tenure'].replace(0,29)

df['MonthlyCharges'].hist()
df['tenure'].describe()
df['tenure'].hist()
df.describe()
df['Contract'].describe()
df['Contract'].hist()
df.groupby(["Contract"])["tenure"].mean().plot.bar()
df['Contract'].hist()
df.groupby(['gender'])['gender'].count().plot.pie()
df.groupby(['SeniorCitizen'])['SeniorCitizen'].count().plot.pie()
df.groupby(['PhoneService'])['PhoneService'].count().plot.bar()
df.groupby(['Partner'])['Partner'].count().plot.pie()
df.groupby(['Dependents'])['Dependents'].count().plot.pie()
df['MonthlyCharges'].mean()
df[df['PhoneService']=='Yes']['PhoneService'].count()
df[df['InternetService']!='No']['InternetService'].count()
df[(df['PhoneService']=='Yes') & (df['InternetServi-
ce']!='No')]['PhoneService'].count()
df.groupby(['InternetService'])['InternetServi-
ce'].count().plot.bar()

df_contracts = df.groupby(['Contract']).Contract.count()
df_contracts.plot.pie(y='Contract', figsize=(5, 5))

df_payments = df.groupby(['PaymentMethod']).PaymentMethod.count()
df_payments.plot.pie(y='PaymentMethod', figsize=(5, 5))

df.groupby(["gender", "SeniorCitizen", "Partner", "Depen-
dents", "Churn"])["Churn"].count().unstack(['gender', "SeniorCiti-
zen", "Partner", "Dependents"])
```

```
df.groupby(["Contract", "Churn"])["Churn"].count().unstack('Con-
tract')
df.groupby(["Contract", "Churn"])["Churn"].count().uns-
tack('Churn').plot.bar()

df.groupby(["PaymentMethod", "Churn"])["Churn"].count().uns-
tack('Churn')

df.groupby(["PaymentMethod", "Churn"])["Churn"].count().uns-
tack('Churn').plot.bar()

df.groupby(['PhoneService'])['PhoneService'].count()

df.groupby('Churn')['customerID'].count()

churns = df['Churn'] == 'Yes'

def churn_0_or_1(value):
    if (value == 'Yes'):
        return 1
    else:
        return 0


df['churn_i'] = df['Churn'].apply(churn_0_or_1)

df.head()

df['churn_i'].mean()

df[df['Churn']=='Yes'][['tenure','MonthlyCharges']].describe().T

df[df['Churn']=='No'][['tenure','MonthlyCharges']].describe().T


df_contracts = df.groupby(['PhoneService']).Contract.count()
df_contracts.plot.pie(y='PhoneService', figsize=(5, 5))

df_contracts = df.groupby(['InternetService']).Contract.count()
df_contracts.plot.pie(y='InternetService', figsize=(5, 5))

df.groupby(["PhoneService", "Churn"])["Churn"].count().uns-
tack('Churn').plot.bar()

df.groupby(["InternetService", "Churn"])["Churn"].count().uns-
tack('Churn').plot.bar()

df[['PhoneService', 'InternetService']].head()
```

```
def services(value):
    #print(value)
    if ((value['PhoneService'] == 'Yes') and (value['InternetServi-
ce'] != 'No')):
        return 'Dois'
    elif (value['PhoneService'] == 'Yes'):
        return 'Somente telefone'
    else:
        return 'Somente internet'
df['services'] = df[['PhoneService', 'InternetService']].apply(ser-
vices, axis=1)
df.head()

df_contracts = df.groupby(['services']).Contract.count()
df_contracts.plot.pie(y='services', figsize=(5, 5))

df.groupby(["services", "Churn"])["Churn"].count().uns-
tack('Churn').plot.bar()
```

MachineLearning

```
#!/usr/bin/env python
# coding: utf-8

import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix,
roc_curve, auc
from sklearn.metrics import roc_auc_score, precision_score, re-
call_score, f1_score
from sklearn.ensemble import RandomForestClassifier
```

```
df = pd.read_csv('./telcom/telecom_users.csv')
df.head()


df = df.drop(['Unnamed: 0', 'customerID'], axis = 1)


df['TotalCharges'] =  df['TotalCharges'].replace(' ', 2298.06)
df['tenure'] =  df['tenure'].replace(0,29)
df['TotalCharges'] =  df['TotalCharges'].astype(float)


ch = {'Yes': 1, 'No': 0}
df['Churn'] = df['Churn'].map(ch)

df.head()


X = df.drop('Churn', axis = 1)
y = df['Churn']

num_cols = X.select_dtypes(include = ['int64', 'float64']).co-
lumns.to_list()
cat_cols = X.select_dtypes(include = ['object']).columns.to_list()

def label_encoder(df):
    for i in cat_cols:
        le = LabelEncoder()
        df[i] = le.fit_transform(df[i])
    return df

sc = StandardScaler()
X[num_cols] = sc.fit_transform(X[num_cols])

# Label encoding
X = label_encoder(X)

X.head()


print(y)


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
= 0.2, random_state = 42)


df.head()
```

```
# REGRESSÃO LOGISTICA
lg = LogisticRegression(random_state = 42)
lg.fit(X_train, y_train)
y_pred = lg.predict(X_test)
y_prob = lg.predict_proba(X_test)[:,1]

# Metricas
print(classification_report(y_test, y_pred))
print("Acuracia", metrics.accuracy_score(y_test, y_pred))

# Matriz de confusão
lg_cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize = (8, 5))
sns.heatmap(lg_cm, cmap = 'Blues', annot = True, fmt = 'd', line-
widths = 5, cbar = False,
            annot_kws = {'fontsize': 15}, yticklabels = ['Não cance-
lou', 'Cancelou'],
            xticklabels = ['Não cancelaria', 'Cancelaria'])
plt.yticks(rotation = 0)
plt.show()


# Floresta aletória

rf = RandomForestClassifier(random_state = 22, max_depth = 5)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)
y_prob = rf.predict_proba(X_test)[:,1]

# Metricas
print(classification_report(y_test, y_pred))
print("Acuracia", metrics.accuracy_score(y_test, y_pred))

plt.figure(figsize = (8, 5))
sns.heatmap(rf_cm, cmap = 'Blues', annot = True, fmt = 'd', line-
widths = 5, cbar = False,
            annot_kws = {'fontsize': 15}, yticklabels = ['Não cance-
lou', 'Cancelou'],
            xticklabels = ['Não cancelaria', 'Cancelaria'])
plt.yticks(rotation = 0)
plt.show()
```

Gráficos

Tabelas

	Unnamed: 0	SeniorCitizen	tenure	MonthlyCharges	TotalCharges
count	5986.000000	5986.000000	5986.000000	5986.000000	5986.000000
mean	3533.561310	0.161377	32.517207	64.802213	2298.060616
std	2035.705666	0.367909	24.480803	30.114702	2272.226516
min	0.000000	0.000000	1.000000	18.250000	18.800000
25%	1777.250000	0.000000	9.000000	35.650000	406.275000
50%	3546.500000	0.000000	29.000000	70.400000	1414.550000
75%	5291.750000	0.000000	56.000000	89.900000	3841.500000
max	7042.000000	1.000000	72.000000	118.750000	8684.800000

gender	**Female**								**Male**							
SeniorCitizen	**0**				**1**				**0**				**1**			
Partner	**No**		**Yes**		**No**		**Yes**		**No**		**Yes**		**No**		**Yes**	
Dependents	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**
Churn																
No	758	94	414	603	131	2	117	26	831	141	377	617	110	4	151	23
Yes	353	28	99	105	133	1	65	7	360	36	110	94	99	1	87	9

Churn	**No**	**Yes**
PaymentMethod		
Bank transfer (automatic)	1084	224
Credit card (automatic)	1105	198
Electronic check	1104	902
Mailed check	1106	263