

[Get started](#)[Open in app](#)

# Paulo Victor

4 Followers   Lists   About

[Follow](#)

Faça login em Medium com o Google



Alexandre Magno Galieta Oliveira  
alexandre.galieta@gmail.com



Moris Salvan  
ms1010tm@gmail.com

[X](#)

Mais 3 contas

## Churn Prediction (Taxa de Rotatividade)



Paulo Victor 1 day ago · 14 min read



**Churn rate**, ou simplesmente **churn**, representa a taxa de rotatividade ou evasão da sua base de clientes, ou seja, **indica o número de clientes que cancelam em determinado período de tempo**, em serviços como Spotify ou Netflix, ela representaria a taxa de cancelamento de assinaturas. Entender por que seus clientes abandonam o seu produto ou serviço é vital para conquistar um crescimento sustentável, como o Churn tem um efeito negativo na receita de uma empresa, entender o que é esse indicador e como trabalhar para mitigar essa métrica é algo crítico para o sucesso de muitos negócios.

**Qual a taxa ideal de Churn?**

[Get started](#)[Open in app](#)

Silício e gerencia mais de \$ 4 bilhões em investimentos ao redor do mundo, indica que uma **taxa de cancelamento aceitável deve ficar entre 5% e 7%**. Muitos negócios costumam ter essa média como base.



## Por que fazer Churn Prediction?

Uma das soluções mais eficientes oferecidas pelo Big Data é o Churn Prediction, ou seja, a previsão do abandono de um serviço pelo cliente. Essa aplicação é realizada por meio de machine learning (método de análise de dados que faz uso automatizado de algoritmos que aprendem interativamente e encontram insights), a partir da análise da rotatividade da base de usuários propensos a desistir do que você oferece ao mercado.

Com isso, ao colocar em prática o conceito de Churn Prediction em sua empresa, será possível criar ações ou campanhas específicas para reter esses clientes e evitar, dessa forma, uma queda no número de conversões — e, consequentemente, no faturamento da empresa.

Outro ponto positivo da aplicação do Churn Prediction é a possibilidade da **redução de**

[Get started](#)[Open in app](#)

antever suas necessidades e oferecer soluções que atendam essas demandas de forma assertiva.

Uma coisa que todo empresário sabe bem é que os custos para conquistar um novo cliente podem ser até **15 vezes maiores** do que para reter um cliente atual. Daí a importância de fazer uso do Big Data Analytics e de sua solução de Churn Prediction. **Com isso, o objetivo dessa publicação é mostrar um modelo de machine learning criado com python na qual faço o Churn prediction de uma empresa de telecomunicações.**

Caso queira ver o notebook do projeto [click aqui](#).

## Aquisição dos Dados

Os dados utilizados neste projeto foram originalmente disponibilizados na [plataforma de ensino da IBM Developer](#), e tratam de um problema típico de uma companhia de telecomunicações e de qualquer outra empresa: cancelamento de serviço. O *dataset* completo pode ser encontrado [neste link](#).

Logo, vamos mostrar os resultados da análise exploratória a qual servirá como objeto para futuras estratégias comerciais da empresa de telecomunicações e após isso, iremos mostrar o nosso modelo de machine learning para churn prediction, como o seu passo a passo.

## Dicionário das variáveis

- **customerID** — ID\_Usuario
- **gender** — Gênero
- **SeniorCitizen** — Idoso
- **Partner** — Parceiro
- **Dependents** — Dependentes
- **tenure** — Meses
- **PhoneService** — Serviço de telefone
- **MultipleLines** — Multi linhas

[Get started](#)[Open in app](#)

## — Security — Segurança —

- **OnlineBackup** — Backup Online
- **DeviceProtection** — Proteção de dispositivo
- **TechSupport** — Suporte técnico
- **StreamingTV** — Transmissão de TV
- **StreamingMovies** — Streaming de filmes
- **Contract** — Contrato
- **PaperlessBilling** — Fatura sem papel
- **PaymentMethod** — Forma de pagamento
- **MonthlyCharges** — Cobranças Mensais
- **TotalCharges** — Custos totais
- **Churn** — Cancelamento

O Dataset é dividido em 3 grandes grupos são eles as características do cliente, o tipo de serviço oferecido a ele e por último questões relacionada a contrato e pagamento.

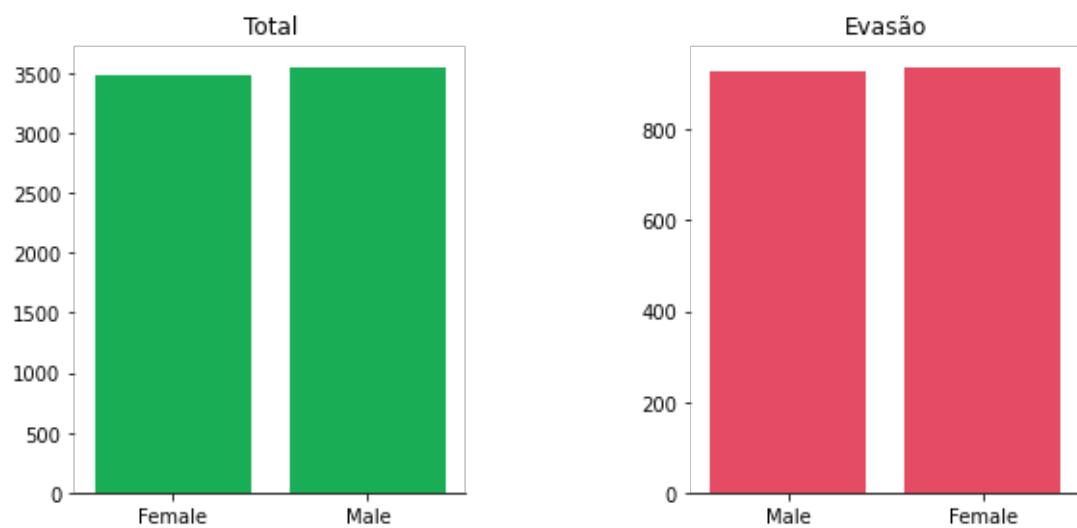
A variável alvo está na coluna **Churn**, indicando os potenciais cancelamentos;

## Análise Exploratória das Variáveis

### Quantidade de Clientes por Gênero

[Get started](#)[Open in app](#)

## Gênero (Total & Evasão)



### Perfil dos clientes:

- Não há diferença entre o número de clientes por gênero, ou seja, a quantidade de homens e mulheres são próximas.

### Perfil dos clientes que cancelam:

- O gênero não exerce influência no cancelamento de serviços.

### Quantidade Total de Clientes Idosos



### Perfil dos clientes:

- Clientes não idosos são maiores em quantidade em relação aos clientes idosos

[Get started](#)[Open in app](#)

- Clientes não idosos tem 3x mais chances de cancelar do que clientes não idosos

## Quantidade de Clientes com Parceiros



### Perfil dos clientes:

- A quantidade de clientes com e sem parceiros são relativamente próximas, porém a categoria sem parceiros se sobressai em relação aos clientes com parceiros

### Perfil dos clientes que cancelam:

- Clientes sem parceiros têm 2x mais chances de cancelar do que clientes com parceiros

## Quantidade de Clientes com Dependentes

[Get started](#)[Open in app](#)

### **Perfil dos clientes:**

- O número de clientes sem dependentes é maior do que os consumidores com dependentes

### **Perfil dos clientes que cancelam:**

- Clientes sem dependentes têm 5x mais chances de cancelar do que clientes com dependentes

### **Quantidade de Clientes por forma de pagamento e tempo de contrato**



### **Perfil dos clientes:**

[Get started](#)[Open in app](#)

check(cheque eletrônico)

- A maioria dos clientes possuem contrato mensal

#### **Perfil dos clientes que cancelam:**

- 57,3% dos consumidores pagavam suas cobranças via electronic check(cheque eletrônico)
- 88,5% dos consumidores possuíam contrato mensal

#### **Quantidade de Clientes por mês de fidelidade**



#### **Perfil dos clientes:**

- A empresa possui mais consumidores no primeiro mês, porém muitos clientes já possuem 72 meses de fidelidade
- Entre os meses 6–69 a empresa possui uma constância na quantidade de clientes

#### **Perfil dos clientes que cancelam:**

- O primeiro mês representa a maior taxa de cancelamento
- Os cancelamentos vao diminuindo progressivamente ate o sexto mes, a partir dai seguesse uma constancia, mas com reduções da taxa de cancelamento com o passar dos meses.

#### **Informações adicionais**



## Preparação dos Dados

Já feita uma análise exploratória, vamos partir para a fase de preparação dos dados. A seguir, vamos preparar o dataset para o modelo.

Indicações, nesse sentido, o objetivo é garantir que os dados sejam limpos e adequados para o modelo. A seguir, a partir disso, vamos preparar o dataset.

- Exclusão de outliers
- Feature Engineering
- Separação dos dados

Vale ressaltar que o **Schilling Standard Survival** é um valor considerado alto já que a empresa Bessemer Venture Partners, indica que uma taxa de cancelamento aceitável deve ficar entre 5% e 7%.

Caso queira entender o motivo da preparação, como a explicação e função de cada Perfil dos clientes que cancelam, método ou como foram divididos os dados, recomendo a leitura completa do [notebook](#) do projeto a qual possui todas essas informações detalhadas.

- Metade dos consumidores que cancelaram fizeram após 10 meses de serviço
- 25% dos consumidores pagavam cerca de R\$56 por mês

**Exclusão de coluna customerID**: 25% dos clientes que cancelaram fizeram após 2 meses de serviço

Para deixar o nosso modelo de previsão mais leve e assertivo, vamos excluir a coluna customerID, pois a permanência dela no nosso dataset influenciará a tomada de decisão do modelo, gerando previsão pouco assertivas.

## Feature Engineering

Agora para melhorar nosso modelo transformaremos as variáveis categóricas em numéricas usando os métodos **label encoder** e **get dummies**. A transformação vai permitir que o modelo aprenda de forma mais coerente e que possa nos dar uma

[Get started](#)[Open in app](#)

- **Método Label Encoding** Essa abordagem é muito simples e envolve a conversão de cada valor categórico em uma coluna de valor binário 0 ou 1. Como exemplo na variável Dependents os valores são NÃO e SIM e usando o método label encoder vamos transformá-las em 0 e 1.
- **Método Get Dummies** Uma Dummy Variable assume um valor 0 ou 1 para indicar a ausência ou presença de determinada variável. Diferente do Label Encoding, onde cada categoria assume um valor numérico, aqui criamos uma espécie de matriz esparça, onde cada categoria ganha uma coluna, com valores 0 indicando ausência, e 1 presença.

Separamos os dados da seguinte maneira:



Avaliar o modelo com os mesmos dados usados no treinamento não é útil, pois isso acaba ocorrendo recompensando os modelos que conseguem “memorizar” os dados de treinamento, invés de fazer a generalização a partir deles, por isso decidimos implementar mais uma etapa (simulação), além da validação do modelo para analisar como o modelo reage a dados novos e menores em relação ao treino.

## Feature Scaling

Por que padronizar os dados?

- O objetivo da padronização é alterar os valores das colunas numéricas no conjunto

[Get started](#)[Open in app](#)

bastante o resultado e não necessariamente porque ela é mais importante como um preditor.

Para Padronizar o dataset de treino usaremos o método:

- **StandardScaler**

Usar o método StandardScaler irá padronizar as variáveis que resultarão em uma média igual a 0 e um desvio padrão igual a 1.

## Balanceamento dos dados

Por que balancear os dados da variável alvo (Churn)?

- Se você está construindo um modelo de machine learning para classificação será necessário balancear os dados, pois, a consequência do desequilíbrio da variável alvo é que o modelo terá uma tendência a dar muitos “alarmes falsos”. Ou seja, na prática ele irá responder muito bem às entradas para as classes majoritárias, mas terá um desempenho inferior para as minoritárias.

Por fim vamos balancear os dados, usando técnicas de:



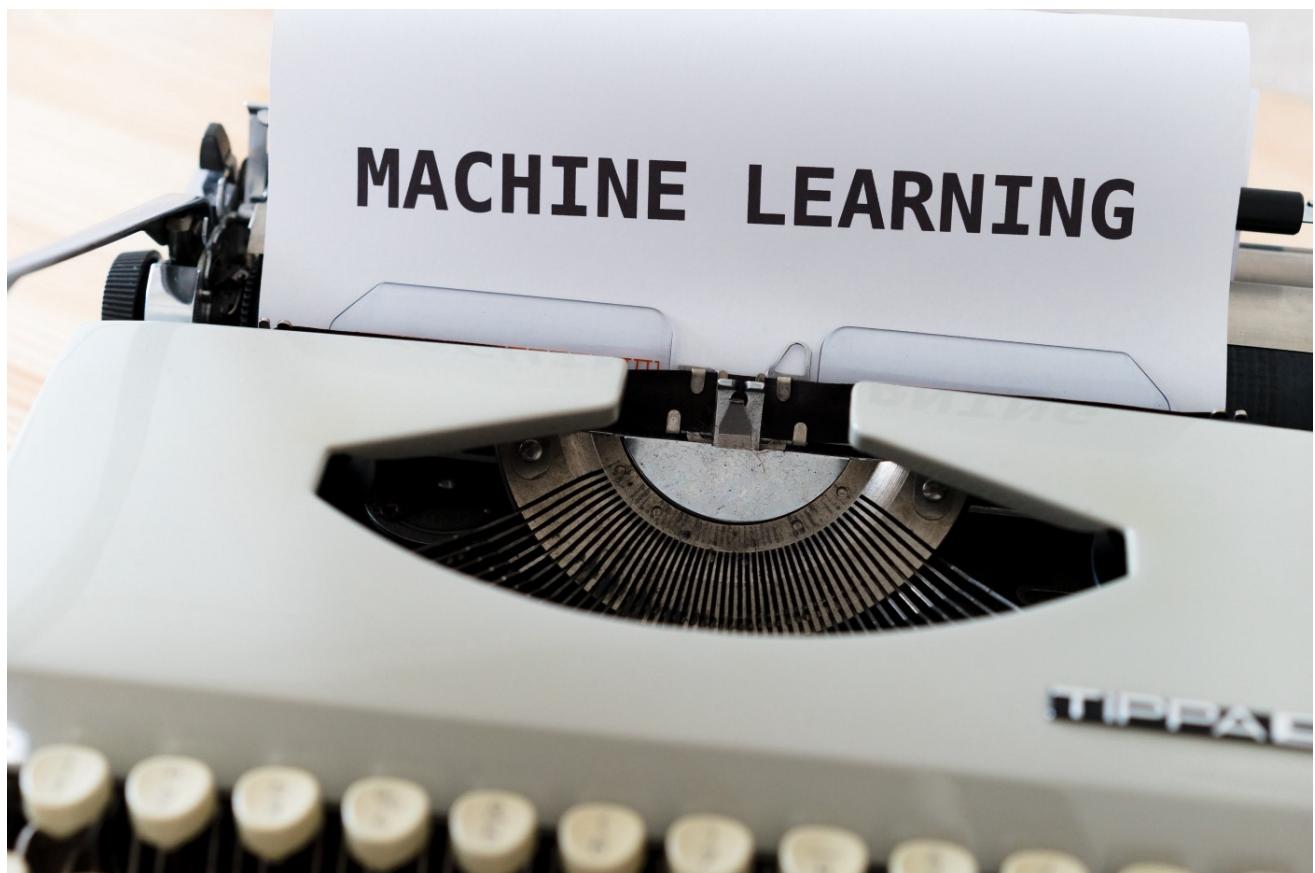
- **UnderSampling**

Irá extrair um subconjunto aleatório da classe majoritária, preservando as características da classe minoritária, sendo ideal para situações onde você tem grandes volumes de dados. Apesar de reduzir o tempo computacional e de armazenamento, esta técnica descarta informações da classe majoritária, o que pode levar a uma performance inferior nas previsões dela.

[Get started](#)[Open in app](#)

Consiste em gerar dados sintéticos (não duplicados) da classe minoritária a partir de vizinhos. Porém o custo computacional será elevado e você irá deteriorar a performance do algoritmo para as classes minoritárias.

## Modelo de Machine Learning



Com o processamento realizado na etapa anterior, os dados já podem ser usados nos modelos de machine learning.

Os Modelos que serão testados na validação cruzada são de Classificação para Aprendizado supervisionado, ou seja, irão analisar as variáveis independentes para prever a variável dependente (Churn), além disso na validação cruzada desses modelos usaremos dados balanceados com SMOTE e Under Sampling e definiremos qual teve melhor resultado.

Selecionamos 4 Modelos para comparação, são eles:

- **DecisionTree**



variável de entrada ( $x$ ) e um ponto de divisão nessa variável (assumindo que a variável seja numérica). Os nós das folhas da árvore contêm uma variável de saída ( $y$ ) que é usada para fazer uma previsão. As previsões são feitas percorrendo as divisões da árvore até chegar a uma folha e gerar o valor da classe nessa folha.

- **RandomForest**

Os algoritmos Random Forest são criados por várias árvores de decisão, geralmente treinados com o método de bagging, cuja ideia principal é que a combinação de modelos aumenta o resultado final.

- **XGBoost**

Modelo de Machine Learning que usa a técnica de Gradient Boosting, capaz de combinar resultados de diversos classificadores “fracos” (tipicamente árvores de decisão) que são combinados mediante um comitê forte de decisão.

- **Logistic Regression**

A regressão logística é o método usado para problemas de classificação binária (problemas com dois valores de classe), utilizando conceitos de estatística e probabilidade. É um algoritmo que lida com questões e problemas de classificação, analisando diferentes aspectos ou variáveis de um objeto para depois determinar uma classe na qual ele se encaixa melhor.

## **Validação Cruzada**

A validação cruzada é uma técnica para avaliar como a análise estatística se generaliza para um conjunto de dados independente. É uma técnica para avaliar modelos de aprendizado de máquina treinando vários modelos em subconjuntos dos dados de entrada disponíveis e avaliando-os no subconjunto complementar dos dados. Usando validação cruzada, há grandes chances de que possamos detectar o ajuste excessivo com facilidade.

O nosso primeiro recall irá servir de parâmetro para os nossos modelos, essa primeira validação cruzada refere-se a um modelo de Random Forest, usando dados de treino, ou seja, dados não balanceados.

[Get started](#)[Open in app](#)

Recall Da nossa primeira validação cruzada: 0.50 (+/- 0.02)



- **Por que utilizamos o recall como forma de avaliar o modelo?**

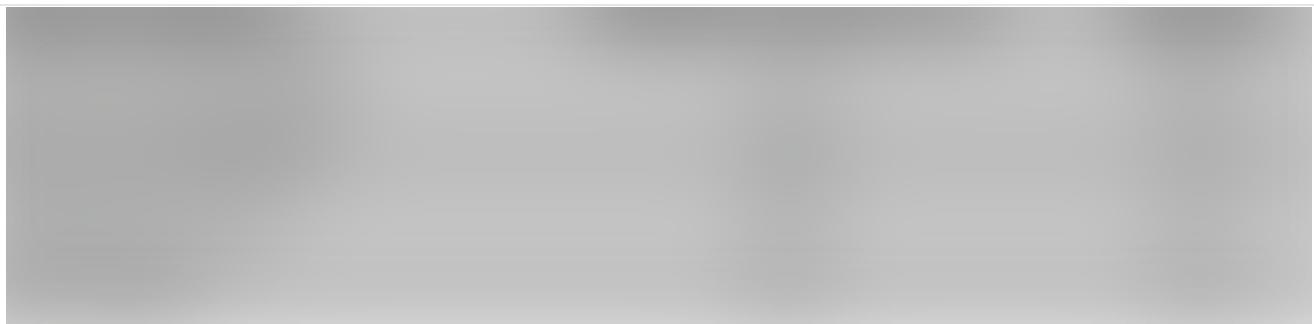
O recall pode ser usado em situações em que os Falsos Negativos são considerados mais prejudiciais que os Falsos Positivos. Por exemplo, o modelo deve de qualquer maneira encontrar todos os consumidores que irão cancelar, mesmo que classifique alguns usuários fiéis como possíveis canceladores do serviço (situação de Falso Positivo) no processo. Ou seja, o modelo deve ter alto recall, pois classificar consumidores que têm alto potencial de cancelamento como clientes fiéis pode ser uma tragédia.

- **Por que usamos Random Forest como o primeiro modelo e quais foram seus resultados?**

A Random Forest, como o próprio nome indica, consiste em um grande número de Decision Trees que operam como um conjunto. Essa combinação de modelos, torna ele um algoritmo muito poderoso, por isso decidimos usá-lo como primeiro modelo da nossa validação cruzada e teremos um recall de comparação com os próximos modelos a serem avaliados com dados balanceados.

Vale ressaltar que o modelo treinado com os dados de treino apresentou um recall de 50%, ou seja, ele consegue prever 50% dos clientes com potencial de cancelamento.

## **Validação cruzada dos Modelos**

[Get started](#)[Open in app](#)

O modelo com o melhor resultado (Recall mais alto) para os dados balanceados com Under Sampling foi o XGBoost, já o modelo com o melhor resultado (Recall mais alto) com os dados balanceados com SMOTE foi o Random Forest.

## Otimização de hiperparâmetros

Sabe-se que ao balancear os dados com SMOTE, perdemos a capacidade preditiva do modelo de prever as classes minoritárias, ou seja, algo que não queremos, porém vamos



Para evitar um tipo de situação chamada de overfitting: quando o nosso modelo fica “viciado” no treino e ruim para os testes, iremos otimizar nossos hiperparâmetros para tentar evitar o overfitting do modelo, vamos testar o modelo XGBoost com Undersampling e SMOTE e de acordo com o resultado iremos, replicar seus hiperparâmetros para o modelo de simulação real.



- Grid Search
- Randomized Search

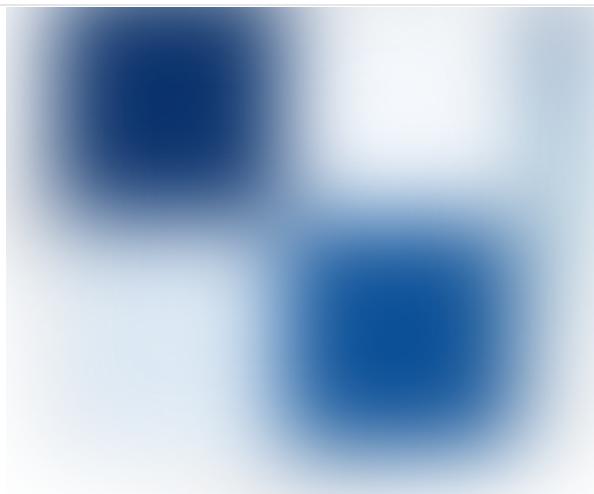
Para o nosso modelo usaremos o Random Search, pois, ele é um ótimo otimizador de hiperparâmetros, que funciona de forma semelhante ao grid search, entretanto, ao invés de testar todas as combinações com a vizinhança, o random search, testa todas as combinações aleatórias dos hiperparâmetros, conforme um número de amostras especificado definido pelo usuário.

Sua principal vantagem em relação ao Grid Search é que ao definirmos um número específico de amostras para cada hiperparâmetro do nosso modelo, conseguimos diminuir o tempo de processamento do algoritmo.

Os hiperparâmetros que vamos melhorar são:

- **N\_estimators:** Indica o número de árvores a serem geradas
- **Max\_depth:** Indica a altura máxima das árvores usadas pelo modelo, ou seja: a quantidade máxima de Nodes que podem haver da raiz até uma folha.
- **Min\_child\_weight:** Indica a soma mínima do peso da instância (hessian) necessária em uma child.
- **Gamma:** Indica a redução de perda mínima necessária para fazer uma partição adicional em um nó folha da árvore. Quanto maior gamma for, mais conservador será o algoritmo.
- **Learning rate:** Indica a taxa de aprendizado é usada para dimensionar a magnitude das atualizações de parâmetros durante a descida do gradiente.

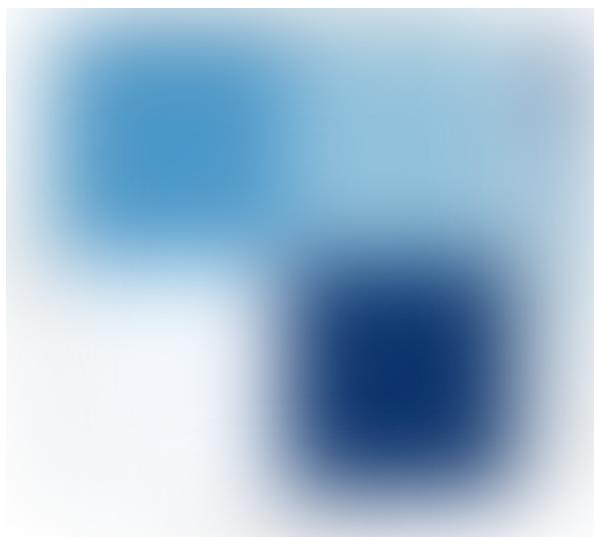
## Validação dos Hiperparâmetros do modelo XGBoost com SMOTE

[Get started](#)[Open in app](#)

Recall — 73%

Podemos ver que a adoção de hiperparâmetros não resultou no recall acima do encontrado com a validação cruzada, mas, em um desempenho menor, com isso, fica evidente que o modelo sofreu com overfitting, ou seja, estava tendo resultados bons para os dados de treino mas ao fazer-lo com os dados de teste com apresentou déficit na sua capacidade preditiva, isso também, reforça o nosso ponto que ao balancear os dados com SMOTE a capacidade preditiva também diminui para as classes minoritárias.

## Validação dos Hiperparâmetros do modelo XGBoost com UnderSampling



### Importância de cada variável no modelo Vencedor — ( XGBoost com UnderSampling)

Diferente do nosso modelo balanceado com SMOTE, este modelo atingiu um recall de quase 90% ao ser balanceado com UnderSampling, ou seja, ele não sofreu overfitting. RandomizedSearchCV para definição dos melhores hiperparâmetros, as variáveis mais importantes para o modelo de previsão são:

[Get started](#)[Open in app](#)

- **OnlineSecurity\_No**
- **InternetService\_Fiber optic**
- **tenure**
- **TechSupport\_No**

## Modelo vencedor com dados de Simulação real

Chegamos no modelo final do nosso projeto de previsão da taxa de rotatividade (Churn) para uma empresa de telecomunicações, mas antes de mostrar seus resultados e capacidade preditiva, vamos mostrar o que significa cada métrica avaliada do modelo.

As formas de desempenho avaliadas nos modelos serão principalmente:

- **Matriz Confusão:**



## Tipos de erros

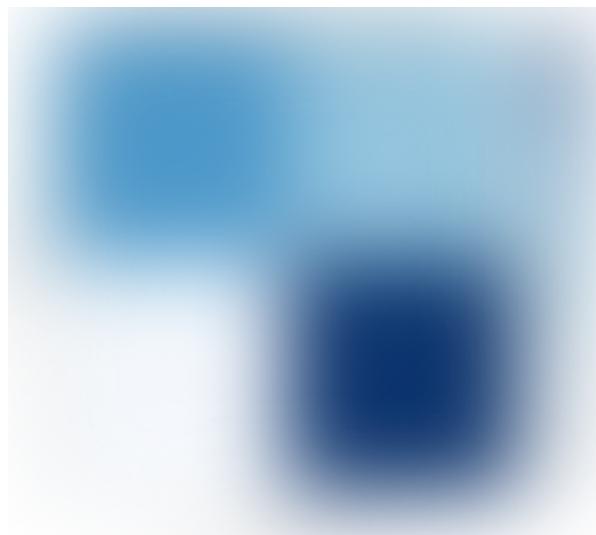
- **Verdadeiro positivo (*true positive* — TP):** Por exemplo, quando o cliente vai cancelar e o modelo classifica como potencial cancelador.
- **Falso positivo (*false positive* — FP):** Por exemplo, quando o cliente não vai cancelar e o modelo classifica como potencial cancelador.
- **Verdadeiro negativo (*true negative* — TN)** Por exemplo, quando o cliente não vai cancelar e o modelo classifica como potencial renovação.

[Get started](#)[Open in app](#)

- **Acurácia:** A acurácia mostra diretamente a porcentagem de acertos do nosso modelo.
- **Recall:** Mostra a proporção de positivos encontrados corretamente, ou seja, o quanto o modelo é capaz de prever potenciais canceladores.
- **AUC:** A curva AUC é derivada da curva ROC, então vamos inicialmente entender a curva ROC. A curva ROC mostra o quanto bom o modelo criado pode distinguir entre duas coisas, a AUC (“area under the ROC curve”) nada mais é que uma maneira de resumir a curva ROC em um único valor, agregando todos os limiares da ROC, calculando a “área sob a curva”.

Agora que tudo já foi explicado, vamos a conclusão do projeto.

## Conclusão



O Modelo obteve as métricas de desempenho:

- **Acurácia:** 64.30% — Porcentagem de acertos do nosso modelo.
- **Recall:** 85.28% — Capacidade de prever potenciais canceladores.
- **Auc:** 71.28% — Precisão das classificações entre renovação e cancelamento.

[Get started](#)[Open in app](#)

objetivo era um alto recall, ou seja, ter qualidade de prever somente as pessoas que irão cancelar como potenciais canceladores.

Com isso, uma **capacidade preditiva de quase 86%**, para os dados de simulação real é uma métrica bastante satisfatória, porém, ainda existe margem para melhorar o modelo como técnicas de feature engineering, feature selection e otimização de hiperparâmetros.

Por fim, através desta análise/modelo é possível traçar o perfil dos consumidores satisfeitos com os serviços prestados e aumentar a retenção dos clientes, visto que este custo é mais baixo do que o de aquisição de novos usuários, para ver mais detalhes do meu projeto recomendo a leitura do [colab](#).

Vale ressaltar que este projeto pertence ao curso Data Science na Prática do professor Carlos Melo.

[Linkedin](#) | [GitHub](#) | [Medium](#)

Machine Learning    Churn    Python    Analytics    Bussiness

About   Write   Help   Legal

Get the Medium app

