Trabalho de Conclusão de Curso apresentado para obtenção do título
de especialista em Data Science e Analytics – 2024

.

# Previsão de churn de assinantes em serviço de Marketplace

Fabricio Rodrigues Zillig[1]*; Rodrigo Rosalis Da Silva[2]

[1] Designer pós-graduado em Novos Negócios e Transformação Digital. São Paulo, São Paulo, Brasil.
[2] Doutor em Ensino de Ciências e Matemática pela Universidade Estadual de Campinas (Unicamp), Mestre em Ensino de Ciências e Matemática pela Unicamp. Graduado em Matemática pela Universidade Federal de São Carlos (UFSCar). São Paulo, São Paulo, Brasil.
*autor correspondente: fabricio.zillig@gmail.com

# Previsão de Churn de assinantes em serviço de Marketplace

## Resumo

No dinâmico e competitivo ambiente empresarial contemporâneo, a capacidade de manter clientes ativos e engajados é crucial. Nesse contexto, o churn, que representa a desistência ou o cancelamento de assinaturas de serviços por parte dos consumidores, emerge como um dos principais desafios a serem superados. A previsão de churn, portanto, transforma-se em uma ferramenta indispensável, habilitando empresas a identificar e reagir proativamente aos sinais de insatisfação ou desengajamento de seus clientes antes que estes optem por abandonar o serviço. O presente estudo desenvolveu um modelo preditivo visando identificar usuários com maior probabilidade de cessar suas vendas na plataforma nos próximos três meses, e, por tanto, ocorrer o churn, utilizando dados reais fornecidos pela empresa e disponibilizados publicamente. A metodologia adotada para a realização desta pesquisa foi o CRISP-DM, uma abordagem consolidada e altamente reconhecida no campo da ciência de dados, envolvendo etapas de compreensão do negócio, dados, preparação dos dados, modelagem, avaliação e implementação. As fases iniciais se concentraram na manipulação e elaboração de features que foram usadas na etapa de modelagem, concentrando-se no uso do algoritmo Random Forest. Os resultados demonstraram que o modelo treinado com base nas features criadas se mostrou viável para uso em um contexto real, demonstrando um bom resultado geral.

Palavras-chave: Churn; Random Forest; CRISP-DM; Modelagem; Machine Learning.

# Churn Prediction of Subscribers in a Marketplace Service

## Abstract

In the dynamic and competitive contemporary business environment, the ability to keep customers active and engaged is crucial. In this context, churn, which represents the discontinuation or cancellation of service subscriptions by consumers, emerges as one of the main challenges to be overcome. Therefore, churn prediction becomes an indispensable tool, enabling companies to identify and proactively respond to signs of dissatisfaction or disengagement from their customers before they choose to abandon the service. This study developed a predictive model with the aim of identifying users with a higher probability of ceasing their sales on the platform in the next three months, and thus occurring churn, using for this real data provided by the company and made publicly available. The methodology adopted for this research was CRISP-DM, a consolidated and highly recognized approach in the field of data science, involving stages of business understanding, data understanding, data preparation, modeling, evaluation, and implementation. The initial phases focused on the manipulation and elaboration of features that were used in the modeling stage, focusing on the use of the Random Forest algorithm. The results showed that the model trained based on the created features proved viable for use in a real context, demonstrating a good result.

Keywords: Churn; Random Forest; CRISP-DM; Modeling; Machine Learning.

## Introdução

O cenário empresarial atual é caracterizado por intensa competição e busca incessante por expandir a base de clientes. Nesse contexto, em que os clientes têm à disposição diversas opções concorrentes, a manutenção dessa base ativa é extremamente importante para os negócios.

Os desafios inerentes à manutenção de uma base sólida de consumidores são especialmente evidentes em serviços digitais, onde a dinâmica de oferta e demanda, assim como as expectativas dos consumidores, podem evoluir rapidamente.

De forma conjunta à competição por clientes, nos últimos 15 anos, tem sido observado grandes investimentos em infraestrutura de negócios, visando melhorias na capacidade de coleta de dados de diversos setores, permitindo, assim, a exploração desse insumo para obtenção de vantagem competitiva (Provost e Fawcett, 2016).

Essa abundância de dados permitiu o surgimento de uma prática chamada Decisões Orientadas a Dados [DOD]. Provost e Fawcett (2016) explicam essa técnica como basear em análise de dados as decisões de negócio, em vez de utilizar apenas a intuição. Estudos comprovam que o uso dessa prática está altamente associado a uma maior produtividade e aumento de lucratividade das empresas (Brynjolfsson et al., 2011).

Provost e Fawcett (2016) mostram que em um cenário de competição acirrada, é mais caro atrair novos clientes do que manter os existentes. Nessa realidade, de abundância de dados e necessidade de retenção de clientes, uma métrica importante a ser observada é o Customer Churn, representando a quantidade de consumidores que deixam de se relacionar com um produto ou serviço ao longo de um determinado período (Nettleton, 2014).

A capacidade de antecipar quais clientes estão mais propensos a encerrar sua relação com uma empresa, com base em dados de comportamento e consumo, possibilita a adoção de estratégias proativas para evitar esse desfecho. Por esse motivo, a análise e previsão de churn se tornam tão relevantes e importantes atualmente.

Este trabalho se propõe a abordar o problema de churn em um cenário específico: um serviço de Marketplace que oferece, na forma de assinatura, uma ferramenta que auxilia nas vendas online. Com o acesso a uma base de dados, disponibilizada de forma gratuita pela empresa, esta pesquisa busca desenvolver um modelo capaz de prever o churn dos assinantes (vendedores) dessa empresa.

O objetivo deste trabalho é criar um modelo preditivo que, com base nas características dos clientes e no seu comportamento de venda dentro da plataforma ao longo do tempo, seja capaz de identificar quais usuários deixam de vender na plataforma nos próximos 3 meses e, por consequência, têm uma maior probabilidade de encerrar seu relacionamento com a plataforma (churn).

## Material e Métodos

O trabalho de pesquisa foi baseado no processo CRISP-DM, um modelo para projetos de Ciência de Dados não proprietário, de uso livre e comum na indústria. Essa metodologia divide o processo em seis fases: Compreensão do Negócio, Compreensão dos Dados, Preparação dos Dados, Modelagem, Avaliação e Implementação (Shearer, 2000).

### Compreensão do Negócio

No atual contexto do comércio eletrônico, que se caracteriza pela sua natureza dinâmica e competitiva, pequenos empreendedores enfrentam desafios no gerenciamento eficaz de múltiplas plataformas de marketplace.

O serviço oferecido pela empresa usada neste projeto procura endereçar essa problemática ao centralizar a exibição de produtos de diversos comerciantes em um único local. Esta estratégia facilita a gestão de presenças online em múltiplos marketplaces, maximizando a visibilidade dos itens ofertados e simplificando operações comerciais.

Seus clientes contratam esse serviço através de pagamentos mensais, por isso a importância de prever usuários que deixarão de vender nos próximos meses, e com isso, aumentarem as chances de cancelar a assinatura, resultando em churn.

Para os objetivos do trabalho, considerando que não temos o acesso aos dados de assinantes da plataforma, vamos determinar que os clientes que não vendem no período de 3 meses e cancelam a assinatura, logo, prever quem não venderá nos próximos períodos é uma forma de prevermos churn.

Nesse contexto, também vamos definir como base ativa (conjunto de clientes que se relacionam de alguma maneira com a empresa) todos os usuários que fizeram alguma venda nos últimos seis meses.

Em situações reais, essas definições seriam extraídas através das áreas de negócios da empresa em reuniões de alinhamento, antes do início do projeto.

## Compreensão dos Dados

O conjunto de dados usado neste estudo é proveniente da própria empresa e foi divulgado publicamente no site Kaggle. O dataset consiste em um conjunto de arquivos no formato CSV com informações sobre clientes e produtos de 100 mil pedidos realizados entre os anos de 2016 a 2018 em vários marketplaces no Brasil (Olist, 2018).

Os arquivos incluem informações anonimizadas, como o status dos pedidos, os preços, as formas de pagamento, o frete, a localização do pedido, os atributos do produto e as avaliações feitas pelos clientes.

Os dados foram importados dos arquivos CSV para um banco de dados relacional SQLite. Essa transformação permitiu uma melhor organização e manipulação das informações durante as etapas subsequentes.

Cada arquivo CSV foi transformado em tabelas dentro desse banco de dados e os registros delas se relacionam através de chaves de valores únicos, como é possível verificar na Figura 1. Existem tabelas que contém informações transacionais e outras apenas com informações cadastrais (não transacionais).

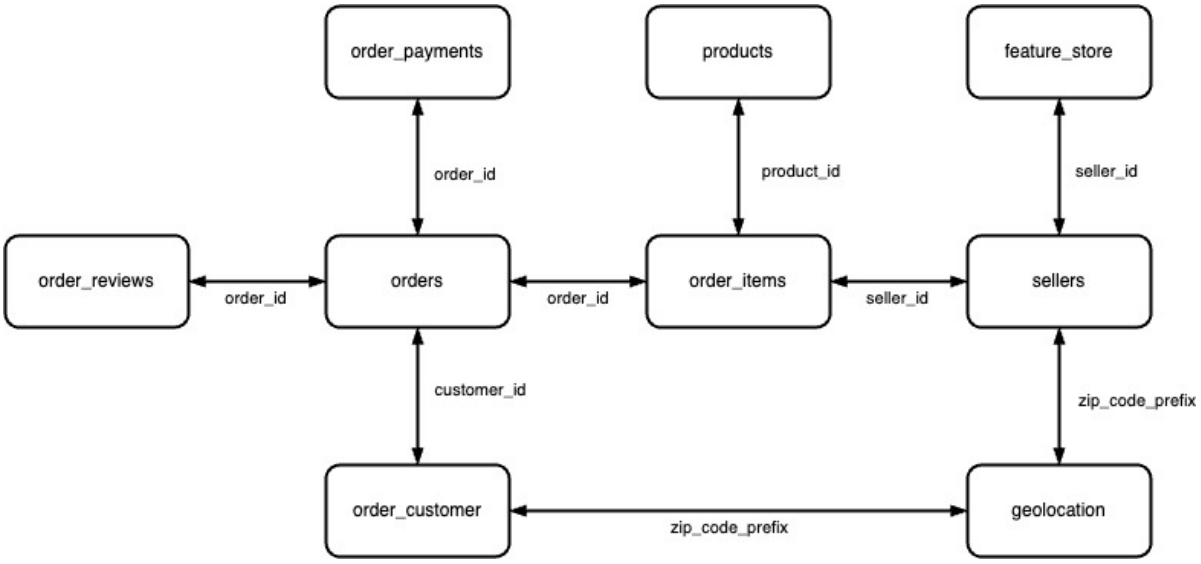


Figura 1. Esquema de relação das tabelas no banco de dados

Para alcançar o objetivo definido usamos como ponto de partida a tabela "orders", que contém as informações de pedidos realizados na plataforma entre os dias 15-09-2016 e

03-09-2018. Apenas para facilitar as manipulações e divisões dos dados, consideramos os pedidos compreendidos entre 01-10-2016 até 01-09-2018 e que foram entregues aos compradores.

**Preparação dos Dados**

Na etapa de Preparação dos Dados foi feito o processo de Feature Engineering, definido por Zheng, A. (2018) como a formulação das mais apropriadas variáveis, tendo em vista o conjunto de dados, o modelo a ser desenvolvido e o objetivo a ser alcançado.

Essa é uma das etapas mais importantes e trabalhosas no processo de criação de um modelo preditivo. É nesse momento que a criatividade é crucial para transformar os dados brutos em features significativas, que alimentarão o modelo e influenciarão diretamente sua capacidade de fazer previsões precisas e úteis.

As features criadas (Tabela 1) incluíram características do cliente, estatísticas relacionadas às vendas, avaliações dos compradores, entre outros. Elas foram armazenadas em uma tabela no banco de dados que usamos como Feature Store.

Tabela 1. Variáveis criadas e armazenadas na Feature Store

| Nome da Feature | Descrição |
|---|---|
| data_ref | Data de referência da Safra |
| seller_id | Identificador único do vendedor |
| primeira_venda_dias | Número de dias desde a primeira venda (idade na base) |
| ultima_venda_dias | Número de dias desde a última venda (última ativação) |
| receita_total | Valor total de vendas realizadas no período (últimos 6 meses) |
| qtd_vendas | Quantidade de vendas realizadas no período |
| avg_receita_vendas | Valor médio dos pedidos realizados no período |
| avg_receita_vendas_mes | Valor médio mensal vendido no período |
| avg_receita_vendas_mes_ativado | Valor médio vendido nos meses que houve alguma venda |
| qtd_mes_ativado | Quantidade de meses que o usuário realizou alguma venda |
| prop_meses_ativado | Proporção de meses que o usuário realizou alguma venda |
| qtd_produtos | Quantidade de produtos vendidos no período |
| qtd_dist_produtos | Quantidade de produtos distintos vendidos no período |
| avg_produtos_vendas | Quantidade média de itens vendidos por pedido |
| avg_valor_produtos | Valor médio dos produtos vendidos |
| avg_avaliacao_pedido | Nota média dos pedidos feitos no período |
| prop_pedidos_atrasados | Proporção dos pedidos entregues após o prazo estimado |
| avg_estimativa_entrega | Quantidade média de dias estimado para entrega |
| prop_product_[categoria] | Proporção de itens vendidos de determinada categoria |

Segundo Kumar (2022) uma Feature Store é um sistema para gerenciar e entregar features aos modelos em produção. No projeto, para fins de estudo e diminuição da complexidade, utilizamos uma tabela SQL como Feature Store. Ela foi responsável por armazenar as variáveis (features) criadas e que usamos para treinamento do modelo de churn. Isso permitiu que usássemos uma abordagem mais estruturada, facilitando a etapa de modelagem e aproximando o resultado da realidade.

Dado a característica do problema e o contexto de negócio encontrado, as características de cada usuário, salvas na Feature Store, foram criadas a partir do que foi definido como base ativa, ou seja, para cada data de referência (sempre o primeiro dia de um mês) foram extraídas features baseadas nas informações de vendas dos usuários de 6 meses anteriores àquela data. Cada conjunto desses foi chamado safra.

A criação das features foram realizadas usando a linguagem SQL e a ingestão das safras na tabela de feature store foi feito através de um script Python.

A partir da Feature Store e de todas as safras disponíveis construímos a tabela analítica [ABT], através de um script python executando código SQL. Segundo Rajan (2015), a ABT consiste em uma única tabela contendo features pré-definidas e a variável alvo. As covariáveis, ou features, já estavam construídas e coletadas na feature store, já variável alvo, ou target, construímos através de uma consulta SQL específica.

Através de uma query SQL, dado um usuário em cada uma das safras, foi coletada a informação de venda nos três meses subsequentes, o resultado foi uma variável binária contendo 0 se houve alguma venda, e, portanto, não aconteceu churn, e 1 caso não tenha ocorrido nenhuma venda, e, portanto, ocorreu o churn.

Dada a necessidade de uma janela de três meses de vendas para sabermos se um usuário cometeu churn ou não, olhamos apenas para as safras de 01-10-2016 até 01-06-2018, o período subsequente, até 01-09-2018, foi utilizado apenas para criar a feature alvo.

O resultado da consulta SQL, que retornou a ABT, foi salvo em um arquivo CSV para facilitar o consumo pelo modelo a ser treinado e poder realizar uma análise exploratória dos dados. O objetivo dessa análise foi entender o comportamento e as características dos registros presentes na tabela.

**Modelagem**

Na fase seguinte, a Modelagem, realizamos o treinamento de um modelo de aprendizado de máquina para problemas de classificação chamado Floresta de Árvores,

utilizando Python e a biblioteca scikit-learn. Este modelo, usado através da classe RandomForestClassifier no pacote ensemble do scikit-learn, como explicado por Negri (2021), é um caso particular de “Bagging”, onde são combinadas diferentes árvores de decisão, treinadas em uma amostragem “bootstrap” dos dados, e a tomada de decisão, ou seja, a classificação, é feita através de votação por maioria. Essa é uma boa forma de contornar o baixo poder preditivo das árvores de decisão isoladas (Izbicki e Santos, 2020).

Esse tipo de modelo foi escolhido pelas características do problema que temos: classificar observações com inúmeras features com variâncias e escalas diversas. Além disso, Morettin e Singer (2022) afirmam que a Floresta de Árvores costuma apresentar uma boa precisão comparada com Árvores únicas.

**Divisão dos dados**

Antes de realizar o treinamento do modelo com os dados, é sempre importante saber como o modelo se comportará na realidade, segundo Silva et al. (2016) não é adequado medir sua performance através dos dados já usados no treinamento, caso seja feito, corre-se o risco do modelo se apresentar sobre ajustado (overffiting).

O sobre ajuste do modelo acontece quando o modelo é gerado de forma a representar o fenômeno com uma fidelidade maior que a necessária, ou seja, não possui bom poder de generalização, tornando-o pouco útil para o uso real (Silva et al., 2016).

Para evitar esse cenário, a ABT foi dividida em dois dataframes, utilizando a biblioteca Pandas, onde um desses subconjuntos é usado para o treinamento e o outro, chamado Out of Time, utilizado para a avaliação final do modelo. Para Klosterman (2019), por termos dados se estendendo ao longo do tempo, por meio de safras, é importante fazer essa separação para calcular as métricas de erro.

A base OOT (Out of Time) contém a última safra gerada (mais recente), referente à 01-06-2019 e será usada para avaliar o desempenho do modelo em um cenário próximo do real. Já os dados de treinamento são todas as safras anteriores à 01-06-2019. A quantidade de registros resultantes da divisão pode ser vistos na Tabela 2.

Tabela 2. Número de observações em cada dataset

| Dataset | Número de Registros |
|---|---|
| df_train | 17.015 |
| df_oot | 1.858 |
| **Total** | **18.873** |

**Definição de métricas de avaliação**

Um passo importante ao treinar um modelo é definir como ele será avaliado para determinar seu desempenho e se ele é melhor que um modelo baseline, por tanto, útil para o uso no mundo real.

Como modelo baseline, aplicado na comparação com outros modelos treinados, foi feito uso do modelo DummyClassifier da biblioteca scikit-learn. Esse modelo classifica todos os resultados como 1, ou seja, como tendo acontecido churn.

Além de um modelo baseline, é importante definir a métrica usada para dizer se um modelo é bom ou não. Um método muito comum é o emprego da matriz de confusão e as métricas derivadas dela. Segundo Silva et al. (2016) uma matriz de confusão tem dimensões C x C onde C é o número de classes possíveis a serem preditos pelo modelo. Cada linha é referenciada pelas classes esperadas enquanto as colunas referenciam as classes preditas pelo modelo. Em uma classificação binária, onde se assume uma classe positiva e outra negativa, a matriz de confusão assume a forma da Figura 2.

| | | $f(x)$ | |
|---|---|---|---|
| | | *Classe Predita Positiva* | *Classe Predita Negativa* |
| $y$ | Classe Real Positiva | Verdadeiro Positivo (**VP**) | Falso Negativo (**FN**) |
| | Classe Real Negativa | Falso Positivo (**FP**) | Verdadeiro Negativo (**VN**) |

Figura 2. Matriz de Confusão

Analisando os resultados por uma matriz de confusão, podemos identificar certos problemas com o modelo dado o contexto do projeto. (Silva et al., 2016). Quando meu modelo prevê corretamente uma classe igual à classe esperada f(X)=y, temos um Verdadeiro Positivo ou um Verdadeiro Negativo. Já quando a previsão não é f(X)=y, temos um Falso Negativo (FN) ou um Falso Positivo (FP).

A partir da matriz, podemos extrair algumas das principais métricas analisadas em modelos de classificação.

A Acurácia (eq. 1), também chamada eficiência global do modelo, corresponde ao percentual de acerto da classificação feita pelo modelo.

$$\text{acurácia} = \frac{VP + VN}{VP + VN + FP + FN} \quad (1)$$

A Precisão (eq. 2) corresponde a quantidade de predições positivas corretas sobre a soma de todos os valores classificados como positivos; ou seja, dos classificados como positivo, qual a porcentagem dos que realmente são positivos.

$$\text{precisão} = \frac{VP}{VP + FP} \quad (2)$$

A Especificidade (eq. 3), por sua vez, corresponde a quantidade de predições negativas corretas sobre a soma de todos os valores verdadeiramente negativo; ou seja, do universo de negativos, qual percentual o modelo conseguiu classificar corretamente.

$$\text{especificidade} = \frac{VN}{VN + FP} \quad (3)$$

A Sensibilidade (eq. 4), também conhecido como recall, é semelhante à especificidade, porém, para casos positivos, é a quantidade de predições positivas corretas sobre a soma de todos os valores verdadeiramente positivos; ou seja, do universo de positivos, qual percentual o modelo conseguiu classificar corretamente.

$$\text{sensibilidade} = \frac{VP}{VP + FN} \quad (4)$$

Uma métrica bastante usada por combinar a sensibilidade e precisão em um único indicador é o F-1 Score (eq. 5), que demonstra um resultado mais equilibrado do desempenho de um modelo, principalmente em situações de classes desbalanceadas, onde tenho uma maior quantidade de observações de uma classe do que de outra.

$$\text{F-1 score} = 2 * \frac{precisão * sensibilidade}{precisão + sensibilidade} \quad (5)$$

Em trabalhos acadêmicos e relatórios gerenciais é comum usarmos um gráfico conhecido como curva ROC (Receiver Operating Characteristic) que apresenta a variação da sensibilidade em função de (1 - especificidade). Esse gráfico facilita a comparação entre modelos já que quanto maior a área embaixo da curva maior a eficiência global do modelo. (Fávero e Belfiore, 2024).

Uma forma comum de avaliar os modelos é através de uma medida única extraída da curva ROC, a métrica AUC (Area under the ROC curve) produz valores entre 0 e 1, onde valores próximos de 1 são considerados melhores que valores próximos de 0,5, que é o equivalente a termos um modelo com classificações aleatórias (Faceli et al., 2021).

**Validação Cruzada e Hiperparâmetros**

Antes do treinamento propriamente dito, é comum configurar certos parâmetros do modelo, tais parâmetros, chamados hiperparâmetros, não são determinados durante o treino e, sim, anteriormente pelo Cientista de Dados que, caso feita de forma errada, pode impactar negativamente a performance do modelo (Negri, 2021).

Para encontrar os melhores hiper parâmetros existem diversas técnicas, entre elas há a busca em grade (Grid Search), que avalia uma métrica de performance, como a acurácia ou a AUC, para diferentes combinações de parâmetros. Os valores de hiper parâmetros escolhidos serão aqueles que maximizam o desempenho do modelo (Negri, 2021).

É comum, juntamente com o Grid Search, o uso da Validação Cruzada (k-fold Cross Validation), que Faceli et al. (2021) define como uma técnica onde o conjunto de dados é dividido em k subconjuntos de tamanho aproximadamente igual. Os dados de k-1 partições são usados para treinamento e a partição restante é aplicada para validação. Esse processo é repetido k vezes, sendo usado em cada ciclo uma partição diferente para validação do desempenho do modelo. Isso permite avaliar que determinada performance não é específica de uma divisão particular do dataset.

A técnica de k-fold Cross Validation com Grid Search para encontrar os melhores hiper parâmetros estão implementadas na biblioteca scikit-learn através do pacote model_selection pela classe GridSearchCV.

Após o treinamento e a avaliação dos modelos com diferentes hiper parâmetros, o modelo que apresentar o melhor desempenho na predição do churn foi selecionado para a validação final na base OOT.

**Resultados e Discussão**

Antes do treinamento, foi realizada uma análise exploratória para entendimento dos comportamentos dos dados usados a fim de treinamento e validação, etapa importante para as decisões tomadas em seguida.

Por conta da característica do problema, foi escolhido como métrica de desempenho a AUC, que se comporta bem nos casos de datasets desbalanceados, que era o caso. Como complemento, foi analisado a F1-score, por ser uma métrica que se comporta bem com desbalanceamento, e a Sensibilidade, já que para o problema proposto é importante identificarmos a maior quantidade possível de usuários que cometerão churn e, assim, poder agir de forma preventiva.

Foi realizado o treino e a avaliação de três modelos distintos: um modelo baseline, que sempre classifica como churn os usuários; uma Floresta de Árvores com hiper parâmetros de valores padrão; e uma Floresta de Árvore com os hiper parâmetros retornados pelo “Grid Search”.

**Análise exploratória dos dados presentes na ABT**

A primeira análise realizada foi verificar a presença de dados nulos ou inválidos e a extração de estatísticas básicas das variáveis. Não foram encontrados valores nulos ou inválidos nos dados. Já ao analisar os quartis, foi identificado um comportamento estranho em relação à distribuição em algumas variáveis.

Para visualizar melhor o que estava acontecendo, foi criado um histograma da variável qtd_vendas (quantidade de vendas realizadas pelos usuários em determinada safra). Como podemos ver na Figura 3, temos uma quantidade enorme de vendedores que vendem poucas vezes, uma mediana de 5 vendas, enquanto existem alguns que realizam muitas vendas, criando essa cauda longa no histograma.

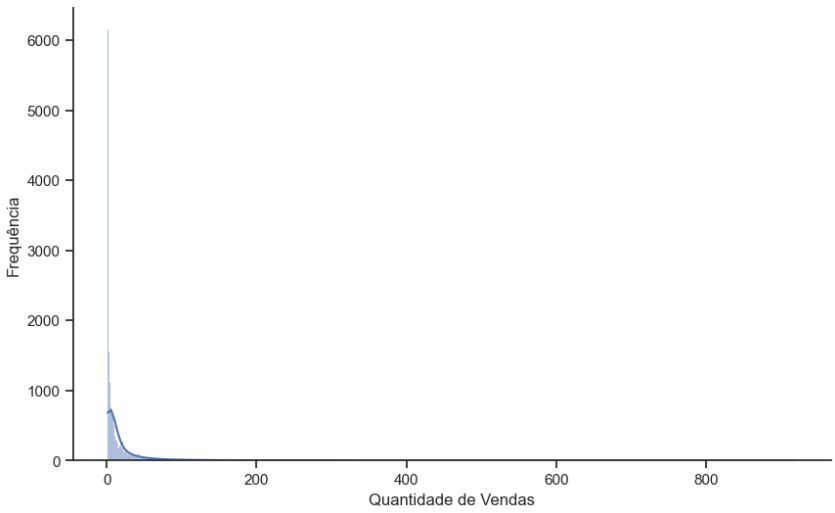


Figura 3. Histograma da variável qtd_vendas

Outra variável explorada foi a ultima_venda_dias (no momento da safra, quantos dias se passaram desde a última venda). Através do Histograma na Figura 4, é possível notar que a base de clientes é bem ativa, com mais da metade vendendo pelo menos um produto por mês.

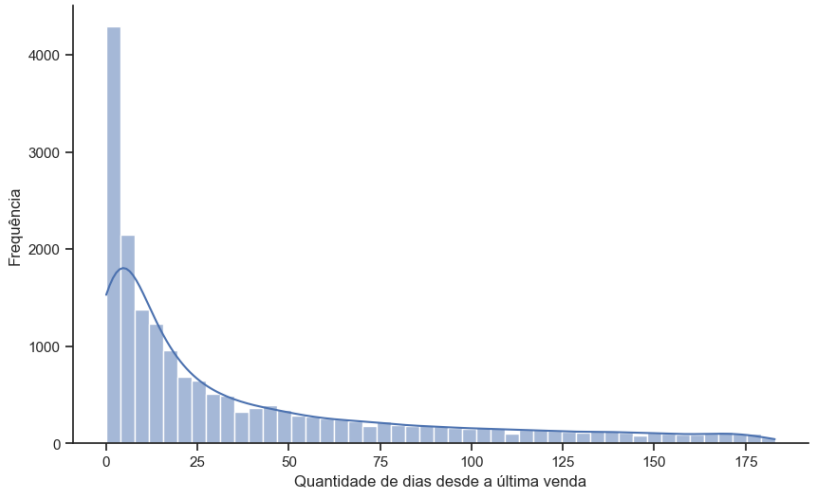


Figura 4. Histograma da variável ultima_venda_dias

Através do Boxplot da Figura 5, podemos verificar que 50% dos usuários nas safras realizam alguma venda em até 17 dias (mediana) e 75% levam no máximo 56 dias (3º quartil)

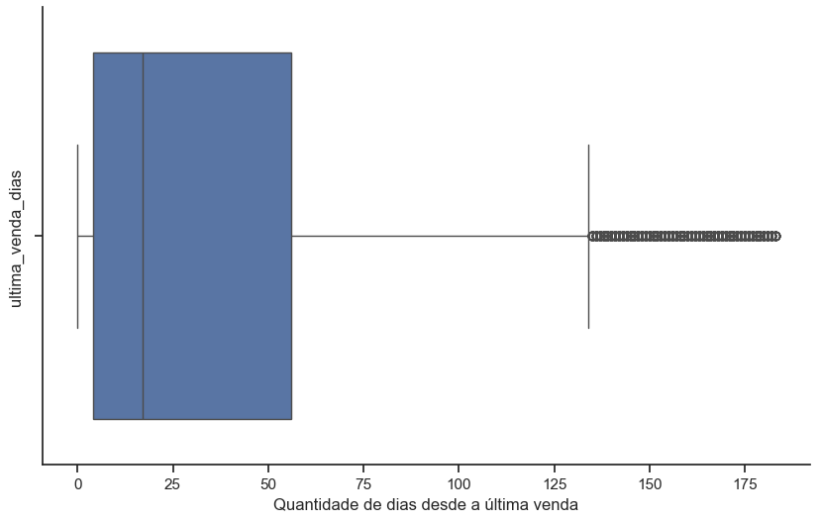


Figura 5. Boxplot da variável ultima_venda_dias

Ao verificar a correlação entre as variáveis na Figura 6, é possível notar a existência de features extremamente correlacionadas, como qtd_vendas (quantidade de vendas realizadas pelos usuários em determinada safra) e qtd_produtos (quantidade de produtos vendidos pelos usuários em determinada safra).

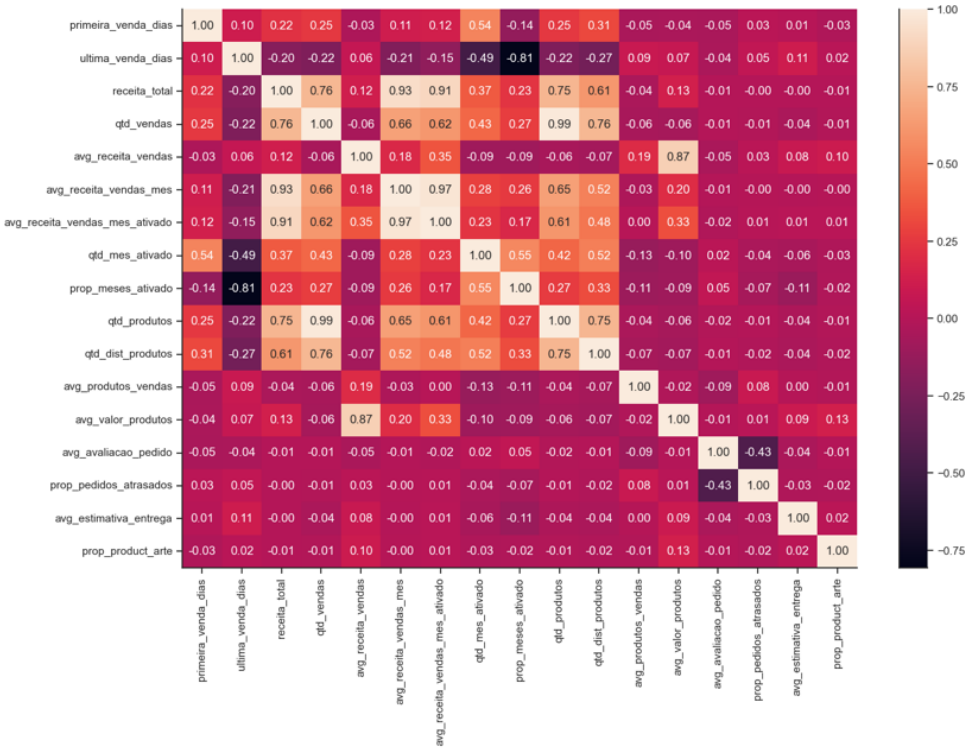


Figura 6. Mapa de calor da matriz de correlação

Essa correlação alta é esperada, já que podemos constatar na Figura 7 que mais de 95% das vendas são de, em média, apenas uma única unidade.

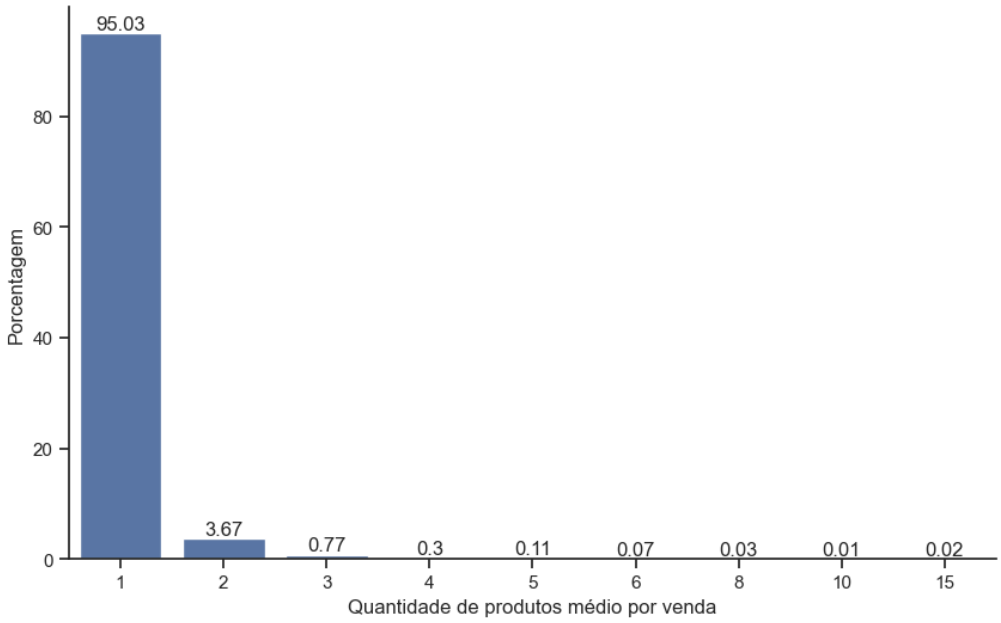


Figura 7. Gráfico de barra da quantidade média de produtos vendidos por pedido

Algumas outras correlações fortes podem ser observadas, como prop_meses_ativado (proporção de meses da safra que ocorreu alguma venda) e ultima_venda_dias (No momento da safra, quantos dias se passaram desde a última venda), que tem uma alta correlação negativa, ou seja, quanto menor o número de dias desde a última venda, maior tende a ser a proporção de meses que algo foi vendido.

Outra relação interessante de ser observada é a prop_pedidos_atrasados (proporção dos pedidos que foram entregues atrasados) e avg_avaliacao_pedido (nota média dado ao pedido). Quanto maior a proporção de atraso nos pedidos, menor é a nota média dos pedidos do vendedor.

Quando concentramos a atenção à variável alvo, podemos ver na Figura 8 que há um desbalanceamento de classes, sendo 68,3% dos usuários na ABT que não cometem churn e 31,7% resultam em churn.

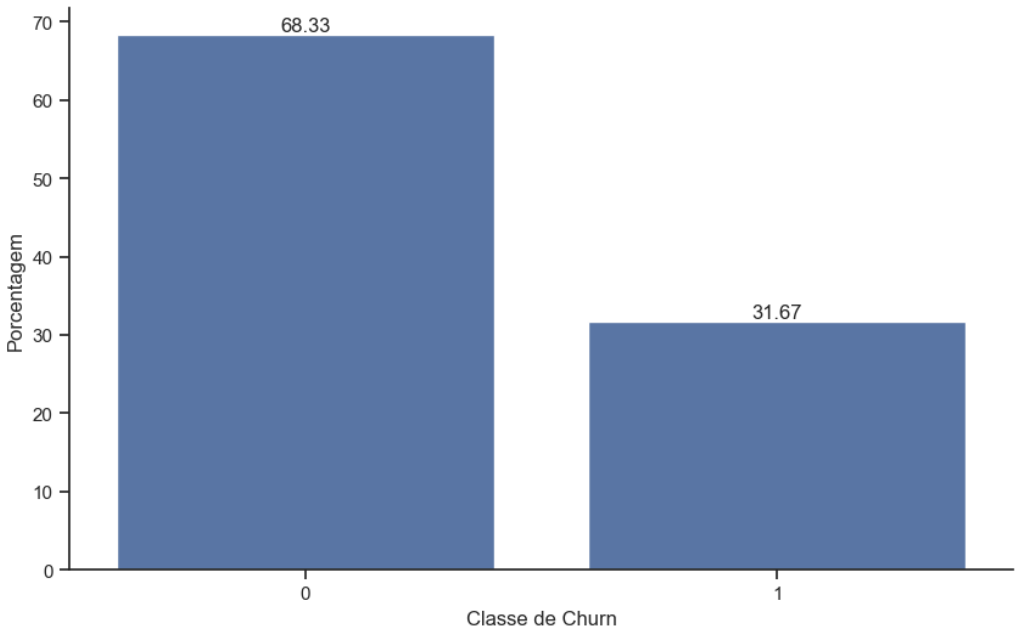


Figura 8. Gráfico de barra da distribuição da classe de churn
Nota: valor 0 da classe representa um não evento (não houve churn) enquanto o valor 1 representa um evento (houve churn)

Ao analisar a correlação das features com a variável resposta, flag_churn, vemos na Figura 9 que as features de proporção de vendas por categoria de produto aparentam ser pouco correlacionadas com churn, por tanto, não devem ser valiosas para prever churn. Já a variável ultima_venda_dias e prop_meses_ativado aparentam ser variáveis interessantes por apresentarem uma correlação significativa.

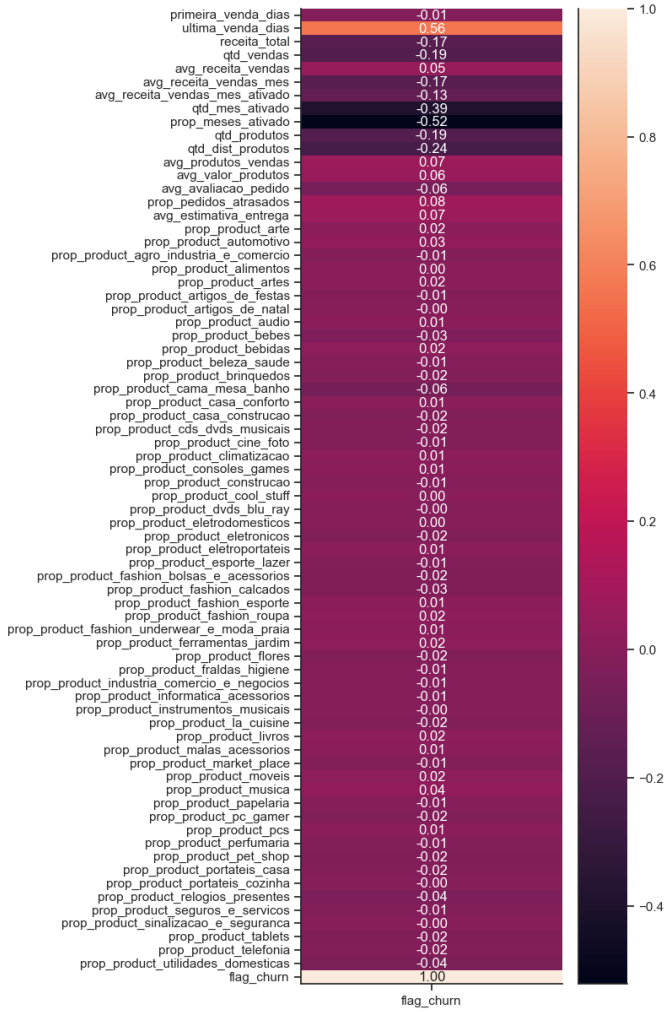


Figura 9. Mapa de calor das correlações entre as covariáveis e variável alvo

**Preparação dos dataframes para o treinamento**

Antes de iniciar o treinamento dos modelos, foi necessário a exclusão de algumas variáveis e separação em dataframes diferentes, as variáveis independentes X, aquelas que usaremos para fazer a classificação, e a variável dependente (y), aquela que queremos classificar.

Como variável independente, foi utilizado todas com exceção da data_ref, seller_id, estado_seller, cidade_seller e flag_churn. Já como variável dependente, separamos a variável flag_churn.

### Treinamento e avaliação do classificador Baseline

O primeiro modelo a ser treinado e avaliado, usado como modelo baseline, classifica todos os registros como churn (classe positiva = 1). Para isso, foi usada a classe DummyClassifier, do módulo dummy da biblioteca scikit-learn.

Como podemos ver na Tabela 3, tanto na base de Treinamento quanto na base de validação (OOT), obtivemos um AUC de 0.5, resultando em uma linha diagonal no gráfico da curva ROC na Figura 10. Esse comportamento é esperado, já que se trata de um modelo baseline.

Já a Sensibilidade (Recall), obtivemos uma métrica igual a 1, o que também é esperado dado que todos os registros foram classificados como churn, logo, de todos os churns possíveis, identificamos 100% dos casos.

Tabela 3. Métricas do modelo Baseline

| | Base de Treino | | | Base de Validação (OOT) | | |
|---|---|---|---|---|---|---|
| **Modelo** | AUC | F1 | Recall | AUC | F1 | Recall |
| Baseline | 0.500 | 0.480 | 1.000 | 0.500 | 0.494 | 1.000 |

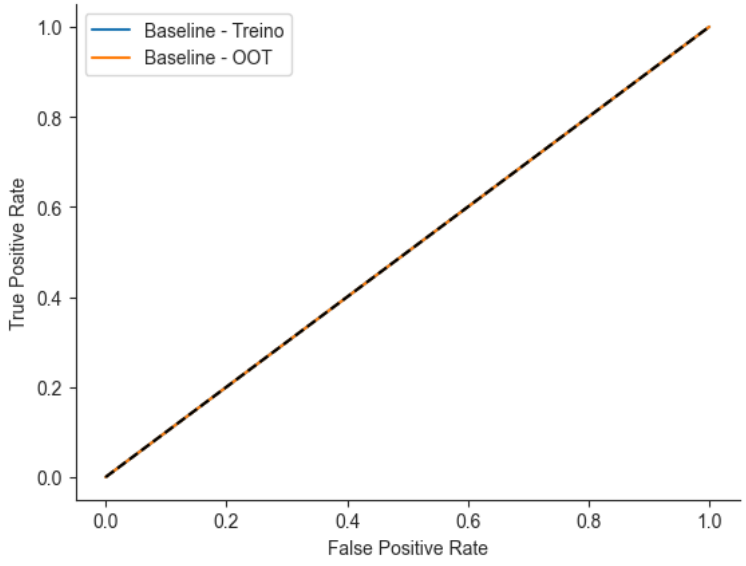


Figura 10. Curva ROC do modelo Baseline

### Treinamento e avaliação dos classificadores Random Forest

Com o modelo Baseline feito e suas métricas avaliadas, iniciamos o treinamento do modelo Random Forest. Para fins de comparação, treinamos um modelo com os hiper parâmetros padrão definidos pela biblioteca scikit-learn.

O treino aconteceu com a base de treino e, em seguida, a performance do modelo foi avaliada, tanto na própria base de treino quanto na base de validação (Out of Time). O resultado pode ser visto na Tabela 4 e na Figura 11.

Tabela 4. Métricas do modelo Random Forest com hiper parâmetros default

| | Base de Treino | | | Base de Validação (OOT) | | |
|---|---|---|---|---|---|---|
| **Modelo** | AUC | F1 | Recall | AUC | F1 | Recall |
| Random Forest (Default) | 1.000 | 1.0000 | 1.000 | 0.864 | 0.681 | 0.633 |

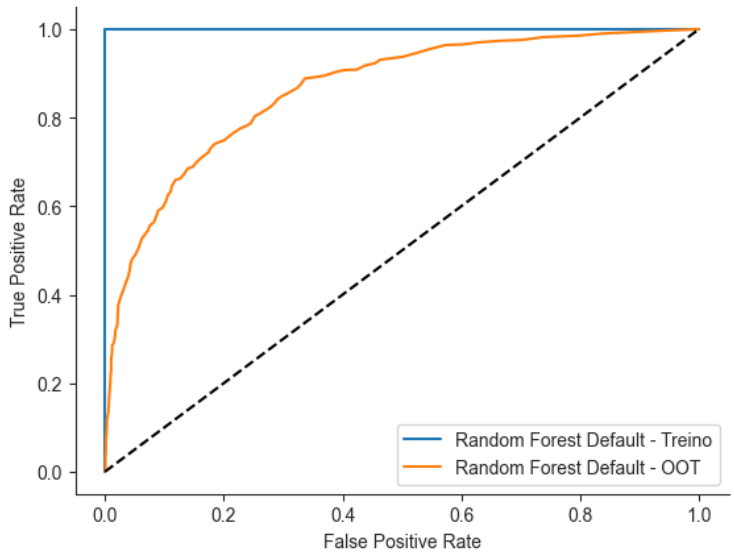


Figura 11. Curva ROC do modelo Random Forest sem alterar hiperparâmetros

Para tentar melhorar o desempenho do modelo de Floresta de Árvores, fizemos um Grid Search com um Cross Validation de 5 “folds” no conjunto de treino. Os hiper parâmetros que buscamos para maximizar o AUC do modelo foram n_estimator (número de árvores), max_depth (profundidade máxima das árvores), min_sample_split (número mínimo de elementos para que uma divisão de nó aconteça) e min_samples_leaf (número mínimo

de elementos em uma folha). No total, haviam 270 combinações de parâmetros possíveis a serem testadas em 5 divisões diferentes do dataset de treino, resultando em 1350 treinamentos.

Como podemos ver na Tabela 5, nos top 5 maiores AUC, obtivemos um AUC médio de mais de 87% em diversas combinações de parâmetros. Optamos por seguir com aqueles que resultaram em menor tempo de treinamento.

Tabela 5. Top 5 resultados do Grid Search com Cross Validation

| # | Parâmetros | AUC Médio | Tempo Médio Treino |
|---|---|---|---|
| 113 | max_depth: 20;<br>min_samples_leaf: 1;<br>min_samples_split: 2;<br>n_estimators: 800 | 0.877 | 16.97 seg |
| 221 | max_depth: None;<br>min_samples_leaf: 1;<br>min_samples_split: 2;<br>n_estimators: 800 | 0.877 | 23.23 seg |
| 112 | max_depth: 20;<br>min_samples_leaf: 1;<br>min_samples_split: 2;<br>n_estimators: 500 | 0.877 | 10.48 seg |
| 220 | max_depth: None;<br>min_samples_leaf: 1;<br>min_samples_split: 2;<br>n_estimators: 500 | 0.877 | 13.23 seg |
| 167 | max_depth: 40;<br>min_samples_leaf: 1;<br>min_samples_split: 2;<br>n_estimators: 800 | 0.877 | 18.12 seg |

Ao usar os parâmetros de max_depth igual a 20, min_samples_leaf igual a 1, min_samples_split igual a 2 e n_estimator igual a 500, foi realizado o treinamento de mais uma Random Forest. Pelos resultados na Tabela 6 e na Figura 12 podemos ver que conseguimos aumentar 0,3% nossa AUC usando esses hiper parâmetros.

Tabela 6. Métricas do modelo Random Forest após processo de fine-tunning

| | Base de Treino | | | Base de Validação (OOT) | | |
|---|---|---|---|---|---|---|
| **Modelo** | AUC | F1 | Recall | AUC | F1 | Recall |
| Random Forest (Tunned) | 0.999 | 0.970 | 0.956 | 0.867 | 0.687 | 0.638 |

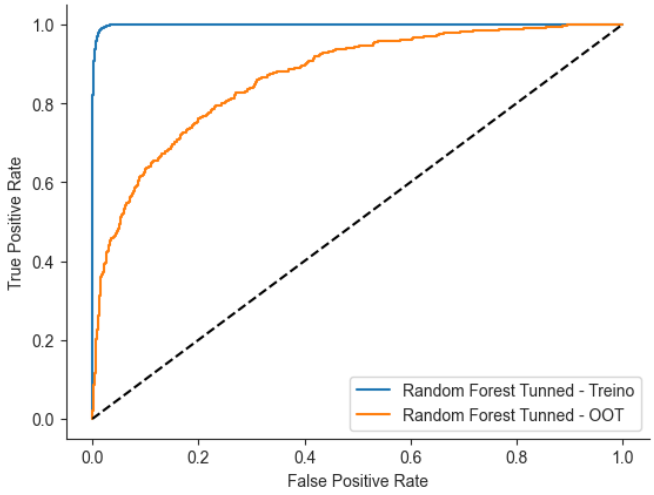


Figura 11. Curva ROC do modelo Random Forest com Hiper parâmetros ajustados

Outra informação interessante que podemos extrair do modelo de Floresta de Árvores é a importância das features usadas no treinamento. A importância das variáveis é determinada observando-se quão efetivo cada feature é na redução do erro nas decisões das árvores individuais. Essa medida é calculada para cada variável em todas as árvores, sendo depois agregada e normalizada.

Features com valores de importância mais altos são consideradas mais influentes para o modelo, indicando que elas contribuem significativamente para a precisão das previsões. Podemos ver na Tabela 7 as dez features que mais contribuíram na divisão dos nós das árvores, por tanto, mais contribuíram para o resultado.

Tabela 7. Features Importance retornada pelo modelo Random Forest Tunned

| Feature | Feature Importance |
|---|---|
| ultima_venda_dias | 0.182 |
| prop_meses_ativado | 0.093 |
| avg_receita_venda_mes | 0.080 |
| primeira_venda_dias | 0.061 |
| receita_total | 0.057 |
| avg_receita_vendas_mes_ativado | 0.052 |
| qtd_vendas | 0.047 |
| avg_receita_vendas | 0.046 |
| avg_valor_produtos | 0.046 |
| qtd_produtos | 0.042 |

Podemos reparar que as duas features com maior valor de importância para o modelo são as mesmas destacadas na análise exploratória como as com maior correlação à nossa variável alvo flag_churn.

Esse resultado é compreensível dado que olhar para quantos dias fazem desde a última venda é um ótimo indicador de um possível churn, da mesma forma a proporção de meses ativados nos indica o quão ativo nas vendas está aquele usuário.

**Comparação final dos resultados**

Ao final do treinamento e da validação dos três modelos, podemos verificar na Tabela 8 que o melhor modelo foi a Random Forest com um AUC de 0,87%. Obtivemos com esse modelo o maior F1 Score e Sensibilidade (Recall) sendo 0,69 e 0,64 respectivamente.

Tabela 8. Métricas do modelo Random Forest com hiper parâmetros default

| | Base de Treino | | | Base de Validação (OOT) | | |
|---|---|---|---|---|---|---|
| **Modelo** | AUC | F1 | Recall | AUC | F1 | Recall |
| Baseline | 0.500 | 0.480 | 1.000 | 0.500 | 0.494 | 1.000 |
| Random Forest (Default) | 1.000 | 1.0000 | 1.000 | 0.864 | 0.681 | 0.633 |
| Random Forest (Tunned) | 0.999 | 0.970 | 0.956 | 0.867 | 0.687 | 0.638 |

Com nosso modelo, conseguimos identificar 64% de todos os usuários que, provavelmente, abandonarão nosso serviço (churn) nos próximos 3 meses. Com essa informação, a empresa poderá tomar ações antecipadas para mitigar ou pelo menos reduzir as chances de abandono desses clientes.

## Considerações Finais

Desenvolver um projeto que abrange desde a coleta e o tratamento de dados até a criação de features pertinentes para a resolução do problema de churn em um marketplace representou um desafio interessante e extremamente valioso para o aprendizado sobre problemas reais. O uso da metodologia CRISP-DM mostrou um caminho claro a ser seguido, além de proporcionar uma imersão profunda nos dados, facilitando a compreensão

das nuances que caracterizam situações de abandono de clientes, e destacando a importância desse tipo de trabalho para a tomada de decisão.

A opção por um algoritmo baseado em árvores de decisão se provou acertada e promissora, destacando-se pela sua simplicidade de compreensão, treinamento e implementação. Esta escolha se revelou eficaz para o problema abordado, mas interessante também para uma variedade de problemas similares.

A solução desenvolvida se mostrou madura e viável de ser utilizada, mas também apresenta espaço para evoluções futuras. A exploração de outros algoritmos de machine learning e a criação de novas features representam oportunidades de refinamento e aumento da acurácia das predições.

**Agradecimento**

Agradeço à minha família e ao meu namorado Alex por todo o apoio e paciência durante o desenvolvimento desse projeto, assim como o professor Rodrigo Rosalis Da Silva, por ter me orientado nesse trabalho e a todos os professores do MBA USP/Esalq pelo carinho e conhecimento compartilhado durante essa jornada.

## Referências

Brynjolfsson, E.; Hitt, L.M.; Kim, H.H. 2011. Strength in Numbers: How Does Data-Driven Decisionmaking Affect Firm Performance?. SSRN Electronic Journal

Faceli, K.; Lorena, A.C.; Gama, J.; Almeida, T.A.; Carvalho, A.C.P.L.F. 2021. Inteligência Artificial: uma abordagem de Aprendizado de Máquina. 2ed. LTC, Rio de Janeiro, RJ, Brasil.

Fávero, Luiz Paulo Lopes e Belfiore, Patrícia Prado. 2024. Manual de análise de dados: estatística e modelagem multivariada com excel, SPSS, stata, R e Python. 2ed. Elsevier, Rio de Janeiro, RJ, Brasil

Izbicki, Rafael e Santos, Tiago Mendonça dos. 2020. Aprendizado de Máquina: uma abordagem estatística. São Carlos, SP, Brasil.

Klosterman, S. 2019. Projetos de Ciência de Dados com Python. Novatec Editora Ltda, São Paulo, SP, Brasil.

Kumar, J. 2022. Feature Store for Machine Learning. Packt Publishing, Birmingham, Inglaterra.

Negri, Rogério Galante. 2021. Reconhecimento de padrões: um estudo dirigido. Blucher, São Paulo, SP, Brasil

Nettleton, D. 2014. Commercial data mining: processing, analysis and modeling for predictive analytics projects. Morgan Kaufmann, Waltham, Massachusetts, USA.

Olist, 2018. Brazilian E-Commerce Public Dataset by Olist. Disponível em: <https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce> Acesso em 20 ago. 2023.

Provost, F.; Fawcett, T. 2016. Data Science para negócios. Alta Books, Rio de Janeiro, RJ, Brasil.

Rajan, M.; B, T. 2015. Credit Scoring Process using Banking Detailed Data Store. International Journal of Applied Information Systems 8 (6): 13-20

Shearer, C. 2000. The CRISP-DM Model: The New Blueprint for Data Mining. Journal of Data Warehousing 5 (4): 13-22

Silva, L.A.; Peres, S.M.; Boscarioli, C. 2016. Introdução à Mineração de Dados com aplicações em R. Elsevier, Rio de Janeiro, RJ, Brasil.

Zheng, A.; Casari, A. 2018. Feature engineering for machine learning principles and techniques for data scientists. O'Reilly Media, Inc., Sebastopol, CA, Estados Unidos.