**Modelos de Machine Learning na Previsão de Churn em E-commerce - Uma Análise Comparativa**

Jorge Prates Nogueira¹\*;Fernando Freire Vasconcelos2

1 Engenheiro Mecânico - Universidade Federal de Minas Gerais, discente do Programa de Pós-Graduação em Data Science e Analytics - USP - Esalq. Rua Marechal Hermes, n. 801 –Gutierrez; 30441-110 Belo Horizonte, Minas Gerais, Brasil.

2 Doutorando em Administração - Universidade de São Paulo – FEA/USP. Endereço completo (pessoal ou profissional) – Bairro; 00000-000 Cidade, Estado, País

\*autor correspondente: jorgepratesn@hotmail.com

**Modelos de Machine Learning na Previsão de Churn em E-commerce - Uma Análise Comparativa**

**Resumo**

O presente Trabalho de Conclusão de Curso (TCC) tem como objetivo principal a identificação do modelo mais eficaz de machine learning para prever o *churn* de clientes em um ambiente de e-commerce. O estudo abrange uma exploração e comparação entre os métodos de regressão logística binária, Random Forest e XGBoost. Adicionalmente, uma análise preliminar dos dados é conduzida, destacando-se a investigação das variáveis mais influentes na construção dos modelos.

Este trabalho busca oferecer contribuições para a otimização das estratégias de retenção de clientes, visando aprimorar a tomada de decisões. Ao fornecer insights sobre os modelos de machine learning mais eficazes para a previsão de *churn*, pretende-se auxiliar gestores e profissionais a desenvolverem abordagens mais assertivas e personalizadas para retenção de clientes.

**Palavras-chave:** Modelo de Regressão, *Gradient Boosting*, *Random Florest*, Predição de *Churn*.

**Introdução**

Manter um cliente pode ser até 25 vezes mais barato do que adquirir um novo (Gallo,2014). Dessa forma, é importante para empresas manter seus clientes e diminuir o *churn,* que édefinido como a propensão dos clientes em interromper negócios com uma empresa em um determinado período de tempo (Lazarov e Capota, 2007). Essa tarefa é um dos principais desafios de um time de *Customer Relationship Management* (CRM), particularmente em ambientes online, nos quais há diversas opções disponíveis e a mudança de empresa não incorre em custos (Buckinx e Van den Poel, 2005).

Segundo Matuszelański e Kopczewska (2022), existem duas formas que companhias podem lidar com *churn* de clientes. A primeira seria aumentando a qualidade de seus produtos e depender de uma publicidade em massa para aumentar a retenção de clientes. A segunda forma, seria focar o marketing em clientes com maior chance de deixar a base, o que é mais barato para empresa. Ainda de acordo com Matuszelański e Kopczewska (2022), cada empresa pode usar uma definição de quando considera que o cliente saiu da base. Isso varia de acordo com o setor, por exemplo, no Varejo Alimentício o tempo entre compras de um cliente é muito menor que no setor de Transporte E Logística.

Dessa forma, a criação de um modelo preciso de previsão de *churn* é essencial para que a empresa possa focar seus esforços de forma assertiva. Segundo Shaaban et al. (2017), modelagem preditiva está principalmente relacionada à previsão do comportamento futuro do cliente por meio da análise de seu comportamento passado. Um exemplo de modelagem preditiva são os modelos de previsão de *churn.*

Os modelos de previsão de *churn* são normalmente formulados como classificação binária que tem como objetivo classificar clientes em duas classes: Y = {*Churner*; *Non Churner*} com base em suas características observadas (Stripling et al., 2017). Além da classificação binaria é importante também que o modelo preveja a probabilidade de o cliente deixar a base. Assim, é possível planejar ações escalonadas sendo as mais agressivas para os clientes que tem mais chance de deixar a base (Matuszelański e Kopczewska, 2022).

Em Patil et al. (2017) algumas técnicas de classificação binaria são introduzidas. Elas incluem: *Random Forests*, *Gradient Boosting,* Regressão Logística e *Neural Networks*. *Random Forests* e *Gradient Boosting* são métodos baseados em arvores de decisão e assim como a Regressão Logística podem ser classificados como técnicas tradicionais. Já o método *Neural Networks* é classificado como s*oft computing techniques (*Shaaban et al. 2017). No próximo tópico essas técnicas serão exploradas em detalhe.

Nesse trabalho serão desenvolvidos modelos utilizando as técnicas de *Random Forests*, de *Gradient Boosting e* de Regressão Logística. Após à criação dos modelos é necessário que sejam selecionados critérios para a avaliação e comparação. Segundo Campos (2018), para modelos de classificação binária, os critérios mais comuns incluem a acurácia do modelo e a área sob a curva ROC. Ambos serão utilizados neste trabalho como métricas fundamentais para avaliação e comparação dos modelos. Os critérios também serão abordados na próxima seção com maior profundidade.

Por fim, é necessário ressaltar que ao se desenvolver modelos de previsão é necessário que haja um cuidado especial para evitar *overfitting*, que acontece quando o modelo é ajustado excessivamente a base de dados que ele é treinado, o que o deixa impreciso para outras bases de dados. Para evitar o *overfitting* será utilizada a técnica de *split validation* ou *Train-ValidationSplit*. De acordo com (Bai, et al., 2021), os dados disponíveis são divididos aleatoriamente em *training split* ou *training set e validation split* ou *validation set*. Dessa forma, o modelo proposto é desenvolvido utilizando o *training set* e testado utilizando o *validation set*.

O objetivo desse trabalho, portanto, é determinar para uma base de dados especifica qual a técnica mais eficaz para prever o *churn.* Para atingir esse propósito serãodesenvolvidos os modelos de classificação mencionados acima.A análise minuciosa desses modelos visa determinar qual deles oferece o desempenho mais robusto e preciso na previsão do *churn* e para isso serão utilizados os dois critérios também mencionados acima.Porém, é necessário ressaltar que a melhor técnica pode variar de acordo com as variáveis disponíveis e para cada caso será necessário um novo estudo.

**Material e Métodos**

**Base de dados**

Para esse trabalho foi utilizada uma base de dados obtida no site *www.kaggle.com.* Ela se chama “E*commerce Customer Churn Analysis and Prediction*” e pertence a uma companhia de *ecommerce* líder de mercado. Não há informação de quando a base foi construída.Ela possui as variáveis listadas na Tabela 1. de 5631 clientes diferentes de um *Ecommerce*. Com base nessa base de dados, foram criados os modelos para previsão de *churn*.

Tabela 1. Base de dados utilizada

|  |  |
| --- | --- |
| Variável | Descrição |
| *CustomerID* | Identificação única do cliente |
| *Churn* | Indicador de *Churn*: 0 sendo *Non Churner* e 1 sendo *Churner* |
| *Tenure* | Tempo de permanência do cliente na organização |
| *PreferredLoginDevice* | Dispositivo de login preferido do cliente |
| *CityTier* | Nível da cidade |
| *WarehouseToHome* | Distância entre o armazém e a casa do cliente |
| *PreferredPaymentMode* | Método de pagamento preferido do cliente |
| *Gender* | Gênero do cliente |
| *HourSpendOnApp* | Número de horas gasto no aplicativo móvel ou site |
| *NumberOfDeviceRegistered* | Número total de dispositivos registrados |
| *PreferedOrderCat* | Categoria de pedido preferida do cliente no último mês |
| *SatisfactionScore* | Pontuação de satisfação do cliente com o serviço |
| *MaritalStatus* | Estado civil do cliente |
| *NumberOfAddress* | Número total de endereços adicionados |
| *Complain* | Se alguma reclamação foi registrada no último mês |
| *OrderAmountHikeFromlastYear* | Percentual de aumento no pedido em relação ao ano anterior |
| *CouponUsed* | Número total de cupons usados no último mês |
| *OrderCount* | Número total de pedidos feitos no último mês |
| *DaySinceLastOrder* | Dias desde o último pedido feito pelo cliente |
| *CashbackAmount* | *Cashback* médio no último mês |

Fonte: O Autor

**Análise Preliminar**

O primeiro passo na construção de um modelo de previsão é a limpeza da base de dados. Segundo Corrales et al. (2018), pode-se afirmar que existem três problemas de qualidade dos dados na construção de um modelo de regressão: valores ausentes, valores atípicos [*outliers*]e redundância (refere-se a instâncias duplicadas). Ao fazer uma análise da base de dados utilizada nesse trabalho é possível identificar que não existe redundância de informação, no entanto, existem valores ausentes e *outliers.*

Segundo Grzymala-Busse e Hu (2001), existem várias formas de se tratar valores ausentes. O método escolhido para esse trabalho foi o de Imputação Baseada em Atributo Ausente que consiste em atribuir um valor representativo a um valor ausente com base em medidas de tendência central, no caso desse estudo foi utilizada a mediana. As variáveis qualitativas não possuem valores nulos na base trabalhada então não foi necessário tratá-las nessa etapa.

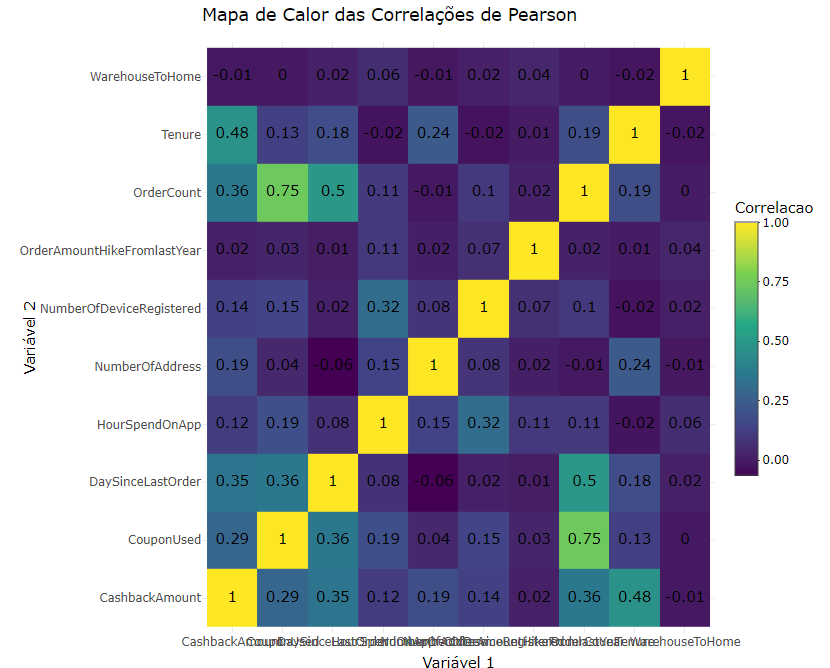
Para lidar com a identificação de valores atípicos, a decisão foi remover os clientes que possuem pelo menos uma variável considerada um *outlier* extremo, conforme sugerido por Corrales et al. (2018). A identificação de outliers extremos é realizada usando a Amplitude Interquartil (AIQ), que representa a distância entre o primeiro quartil [Q1] e o terceiro quartil [Q3]. São considerados *outliers* extremos valores menores que , conforme eq. (1) e maiores que conforme eq. (2) (Fávero e Belfiore,2017).

|  |  |
| --- | --- |
|  | (1) |
|  | (2) |

Na etapa subsequente do processo, realiza-se a análise exploratória dos dados, conforme preconizado por Campos (2018). Este estágio visa caracterizar as variáveis mais relevantes, obtendo insights preliminares. Inicialmente, são calculadas estatísticas descritivas para as variáveis quantitativas, distinguindo entre clientes *churn* e *Non Churner,* proporcionando uma compreensão mais aprofundada de seus comportamentos. São plotados, também, histogramas para as variáveis quantitativas. Em seguida, para variáveis qualitativas, destaca-se o cálculo da representatividade de cada valor, explorando sua influência no modelo a ser construído. Este enfoque abrangente na análise exploratória permite uma compreensão mais sólida e informada dos dados, preparando terreno para as fases subsequentes do processo analítico.

Após esse segundo passo, foi gerada uma matriz de correlação de Pearson, uma tabela na qual as variáveis são apresentadas nas linhas e nas colunas. Os valores nas células representam a correlação entre as variáveis, oferecendo uma visão abrangente das relações existentes no conjunto de dados. (Bruce e Bruce 2019). Esse estudo é importante para identificar a possibilidade de multicolinearidade entre as variáveis independentes. Segundo Daoud (2017), multicolinearidade ocorre quando duas ou mais variáveis independentes no modelo de regressão estão altamente correlacionadas. Um pequeno grau de multicolinearidade, por vezes, pode causar problemas significativos, mas quando é moderado ou alto, torna-se um problema a ser resolvido.

Dessa forma, caso haja uma correlação forte entre as variáveis independentes, é necessário aplicar o método de análise de componentes principais [PCA] (Fridrich e Dostál 2022). Segundo Bruce e Bruce (2019), a ideia na PCA é a combinação de múltiplas variáveis preditoras numéricas em um conjunto menor de variáveis. Para esse trabalho, no entanto, não foi necessário fazer um PCA, uma vez que a correlação entre as variáveis é baixa, conforme Figura 1.

Figura 1. Matriz de correlação entre variáveis independentes numéricas

Fonte: Resultados originais da pesquisa

O coeficiente de correlação é uma métrica que mede o nível em que as variáveis numéricas estão associadas umas às outras. O coeficiente pode variar de –1 a +1, sendo –1 uma correlação linear negativa perfeita, +1 uma correlação linear positiva perfeita e 0 a falta de correlação. Para calcular o coeficiente de correlação de *Pearson* (r), multiplicamos os desvios da média da variável 1 pelos da variável 2, e dividimos pelo produto do desvio-padrão - eq. (3) (Bruce e Bruce 2019).

|  |  |
| --- | --- |
|  | (3) |

Onde:

N é a quantidade de graus de liberdade (nesse caso é a quantidade de observações;

e são os valores das variáveis x e y, respectivamente;

e são os valores médios das variáveis x e y, respectivamente;

e são os desvios padrão de x e y respectivamente.

Após a construção da matriz de correlação é necessário padronizar as variáveis. De acordo com Campos (2018), padronização é o processo de colocar diferentes variáveis na mesma escala, subtraindo cada variável pela sua média e dividindo-a pelo seu desvio padrão. Isso é feito para que as variáveis tenham a mesma influência no modelo, independentemente de seu valor absoluto.

A próxima etapa foi a tratativa das variáveis qualitativas. Segundo Fávero e Belfiore (2017) é necessário transformá-las em variáveis binárias [*Dummies*] que assumem valores iguais a 1 ou 0, conforme exemplo da Tabela 2. Nesse exemplo, a variável Avaliação, que pode assumir três valores (Boa, Média e Ruim), é transformada em 2 variáveis e . Dessa forma, quando a avaliação é “Boa” as duas variáveis irão assumir o valor 0, por exemplo.

Tabela 2. Criação de variáveis binárias (*dummies*) para a variável avaliação

|  |  |  |
| --- | --- | --- |
|  | **Variáveis Binárias (*Dummies*)** | |
| **Avaliação** |  |  |
| Boa | 0 | 0 |
| Média | 1 | 0 |
| Ruim | 0 | 1 |

Fonte: Fávero e Belfiore (2017)

Por fim, foi feito o *split validation. C*onforme explicado na introdução essa técnica separa a base em duas. Uma para treino do modelo, no caso desse estudo 75% da base e outra para teste conforme sugerido por Campos (2018).

O código, feito na linguagem R, para todas as etapas da análise preliminar pode ser encontrado no seguinte link: https://github.com/Jorgepratesn/TCC.

**Regressão Logística**

Conforme introduzido na introdução desse trabalho, um modelo de classificação binária tem como propósito classificar um evento em duas categorias (Y=1 quando o evento ocorre e Y=0 quando o evento não ocorre). Uma das técnicas disponíveis para isso é a Regressão Logística Binária. De acordo com Harrell e Harrell (2015), esse modelo foi desenvolvido principalmente por Cox, Walker e Duncan. Nele os parâmetros de regressão são estimados pelo método da máxima verossimilhança, que será explicado a seguir. Esses parâmetros são essenciais para o cálculo do logito [Z], conforme eq. (4), que por sua vez é necessário para o cálculo da probabilidade de ocorrência do evento [] – eq. (5). A resposta na fórmula de regressão logística é a chance de um resultado binário de 1 (Bruce e Bruce 2019).

|  |  |
| --- | --- |
|  | (4) |
|  | (5) |

Variando o Z de –4 a 4 obtém-se o gráfico da Figura 2. Embora, Z possa assumir valores de a a figura ilustra bem que pode assumir valores de 0 a 1.

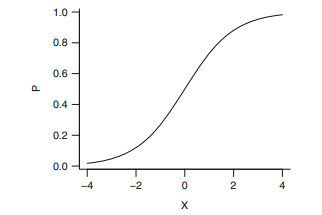


Figura 2. Função Logística

Fonte: Harrell e Harrell (2015)

O método da máxima verossimilhança é utilizado para estimar os parâmetros , conforme mencionado anteriormente. Esse método, segundo King (2018) consiste em iterar a função *log likelihood function* [LL] - eq. (6) - até achar seu valor máximo. A desvantagem de utilizar um método iterativo, e, portanto, o modelo de regressão linear, é a possibilidade que a função não irá convergir na melhor estimativa.

|  |  |
| --- | --- |
|  | (6) |

Onde é a probabilidade observada, podendo ser igual a 1 ou 0.

Um passo importante ao se desenvolver um modelo de regressão logística binária é verificar se cada um dos parâmetros utilizados é estatisticamente significante para o modelo. Nesse trabalho foi utilizada a técnica do *stepwise* para excluir variáveis que não se mostraram estatisticamente significante no modelo, conforme sugerido por. O método *stepwise* é uma técnica interativa em que variáveis são adicionadas ou removidas do modelo. A cada passo, verifica-se a significância estatística das variáveis testadas naquele momento. É necessário definir o nível de significância dos testes que serão feitos, no caso desse trabalho foi adotado um valor de 0,05. As iterações continuam até que seja encontrado um conjunto de variáveis em que todas sejam estatisticamente significativas no modelo. O problema no procedimento *stepwise* é a possibilidade da exclusão de uma variável que é significativa ao modelo. Dessa forma, é necessário comparar o modelo antes e depois do procedimento para avaliar se a aplicação do *stepwise* realmente melhorou o modelo (Fávero e Belfiore,2017).

O código, feito na linguagem R, para a criação do modelo de regressão pode ser encontrado no seguinte link: https://github.com/Jorgepratesn/TCC.

**Árvores de decisão**

As árvores de decisão [DT] são outra abordagem frequentemente empregada em modelagem de classificação binária. Segundo Kotsiantis (2013), as árvores de decisão são modelos sequenciais que combinam logicamente uma sequência de testes simples; cada teste compara um atributo numérico em relação a um valor de limiar ou um atributo nominal em relação a um conjunto de valores possíveis.

A Figura 3, ilustra um modelo simples de árvore de decisão que inclui uma única variável alvo binária Y (0 ou 1) e duas variáveis contínuas, x1 e x2, que variam de 0 a 1. As árvores são compostas por nós e ramos. Cada nó representa características em uma categoria a ser classificada, e cada subconjunto define um valor que pode ser assumido pelo nó (Charbuty e Abdulazeez, 2021). No exemplo da Figura 3, o primeiro nó da árvore de decisão divide a base de dados em duas partes: uma com leituras de x1 menores que 0,5 e outra com leituras de x1 maiores ou iguais a 0,5. Se o valor da leitura de x1 for menor que 0,5, o algoritmo segue para o segundo nó. Este nó, por sua vez, divide a base de dados em duas partes: uma com leituras de x2 menores que 0,3 e outra com leituras de x2 maiores ou iguais a 0,3. Se o valor da leitura de x2 for menor que 0,3, o algoritmo classifica o valor Y como R1. Nada impede que o valor de R1 seja igual ao valor de R5, por exemplo.

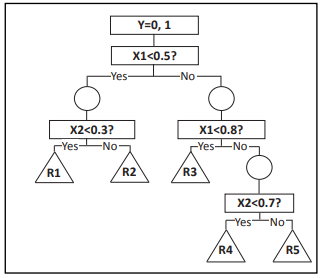


Figura 3. Árvore de decisão baseada na variável alvo binária Y

Fonte: Song e Ying (2015)

Segundo Song e Ying (2015), apenas variáveis relacionadas à variável-alvo [Y] são usadas para dividir os nós, podendo ser discretas ou contínuas. As variáveis mais importantes são identificadas, de forma que cada divisão resulte em um nó mais puro, que conterá o número máximo possível de instâncias em cada subconjunto pertença a uma única classe. É essencial estabelecer um critério para a seleção de uma característica que será um nó na árvore. Tipicamente, isso é feito calculando o Ganho de Informação ou a Entropia das características de cada instância no subconjunto. Esse processo continua até atender a critérios de parada. Nem todas as variáveis potenciais são usadas, e uma mesma variável pode aparecer em diferentes níveis da árvore.

Para evitar *overfitting*, são aplicados os critérios de parada, como o número mínimo de registros em um ramo, o número mínimo de registros em um nó antes da divisão e a profundidade máxima da arvore. Esses hiper parâmetros devem ser escolhidos com base nos objetivos da análise e nas características do conjunto de dados. Recomenda-se manter a proporção de registros em uma folha entre 0,25% e 1,00% do conjunto de dados de treinamento completo (Song e Ying 2015). Para selecionar os melhores hiperparâmetros, pode-se realizar um *Grid Search*, uma abordagem iterativa que busca encontrar os valores de hiperparâmetros que maximizam a acurácia do modelo.

Outra maneira de evitar o *overfitting* é realizar uma "pós-poda". Essa técnica envolve a construção da árvore completa e, em seguida, a remoção de determinadas subárvores, substituindo-as por um nó. Para isso, é calculada a estimativa de erro de cada nó que é pai de uma subárvore. Também é calculado o erro da subárvore podada, contendo apenas um nó. Se a estimativa de erro da subárvore podada for menor, ela será removida. (Carvalho, 2014).

As duas abordagens explicadas nos parágrafos acima para evitar o *overfitting* são mutuamente exclusivas, ou seja, uma não pode ser utilizada se a outra estiver em uso. No entanto, ambas podem ser implementadas em conjunto com a técnica de *k-fold cross*-*validation* ou k-partições validação cruzada. Segundo Carvalho (2014), essa técnica consiste em dividir a base de treino em k partições de forma randômica. São realizadas k iterações, em cada uma das quais uma das partições é usada como conjunto de teste, enquanto as outras são utilizadas como conjunto de treino. Após todas as partições terem sido utilizadas para teste, a margem de erro de cada iteração é somada, e a média das k iterações torna-se a margem de erro do modelo. De acordo com Witten e Frank (2000), foi identificado por meio de testes que o valor ótimo para k é 10.

Nesse trabalho foram empregadas as duas abordagens mais conhecidas baseadas em árvores de decisão para desenvolver um modelo de classificação: *Random Forest* e *XGBoost*. Essas duas técnicas são definidas como modelos e*nsemble*. SegundoDietterich (2000), uma das áreas mais ativas de pesquisa em aprendizado de máquina tem sido o estudo de métodos para construir bons modelos e*nsemble*. A principal descoberta é que, frequentemente, esses modelossão muito mais precisos do que os classificadores individuais que os compõem. Modelos e*nsemble* são um conjunto de modelos cujas decisões individuais combinadas de alguma forma para gerar novos exemplos.

O algoritmo de *Random Forest*, proposto por L. Breiman em 2001, tem sido extremamente bem-sucedido como um método de classificação e regressão. Essa abordagem combina várias árvores de decisão randomizadas e agrega suas previsões por meio de médias (Biau e Scornet, 2016). Segundo Rigatti (2017), a randomização é implementada de duas maneiras: primeiro, o conjunto de dados é amostrado com reposição. Em seguida, ocorre a randomização nos nós de decisão, onde um certo número de preditores é escolhido em cada nó. Este processo é repetido várias vezes para criar a *Random Forest*. Ou seja, cada árvore receberá uma parte diferente da base de dados, e cada árvore terá hiperparâmetros diferentes.

Neste projeto, foi refinada a performance da *Random Forest* através de um *Grid Search*, ajustando hiperparâmetros cruciais, como *mtry* (número de variáveis aleatórias em cada nó), *ntree* (quantidade de árvores) e *nsplit* (máximo de divisões por variável). Foram adotadas estratégias convencionais, como a escolha da raiz quadrada da quantidade de variáveis (entre 4 e 5) explicativas para *mtry* e a exploração de 100 a 1000 árvores para *ntree*. Para *nsplit*, foi estabelecida uma faixa entre 10 e 100 (Rigatti 2017). Para garantir que não haverá *overfitting* foi utilizada a técnica do *k-fold cross*-*validation,* com k igual a 10 para o treinamento da *Random Forest.*

Após o refinamento *Random Forest* é importante que seja feito uma verificação da importância das variáveis. Essa verificação ajuda a identificar as variáveis mais influentes na previsão e a priorizar esforços de coleta de dados e modelagem. Além disso, a análise das variáveis mais importantes pode fornecer insights sobre os fatores que mais afetam a variável alvo, ajudando a entender melhor o problema em questão. Segundo Menze et al. (2009) essa verificação pode ser feita pela importância de Gini. Essa pontuação de importância é, tecnicamente, um subproduto no treinamento do classificador da *Random Forest*: em cada nó τ dentro das árvores binárias da *Random Forest*, busca-se a divisão ideal usando a impureza de Gini, uma aproximação da entropia. Essa métrica mede o quão bem uma divisão está separando as amostras das duas classes em um nó específico.

Já o conceito fundamental do *XGBoost* ou *Extreme Gradient Boosting*, é adicionar novos modelos ao conjunto sequencialmente. No entanto, ao contrário dos métodos como o Random Forest, onde as árvores são cultivadas em paralelo, os métodos de *Boosting* treinam modelos um após o outro, sendo que cada nova árvore ajuda a corrigir erros cometidos pela árvore treinada anteriormente (Wohlwend 2023). O treinamento sequencial do modelo usando o *gradient boosting* diminuirá gradualmente uma função de perda. No *gradient boosting*, após a construção dos *weak learners* (árvores de decisão com profundidade limitada), as previsões são comparadas com os valores reais. A diferença entre a previsão e os valores reais representa a taxa de erro do modelo. A taxa de erro do modelo pode ser usada para calcular o gradiente, essencialmente a derivada parcial da função de perda. O gradiente é usado para encontrar a direção que os parâmetros do modelo teriam que mudar para reduzir o erro na próxima rodada de treinamento (Belyadi e Haghighat, 2021).

Da mesma forma que no Random Forest, o desempenho do *XGBoost* foi aprimorado por meio de um *Grid Search*. Os seguintes hiperparâmetros foram ajustados: *nrounds* (número máximo de iterações) foi definido como 100 e o *early\_stopping\_rounds* (quantidade de iterações necessárias para interromper o modelo em caso de falta de melhorias) foi configurado para 10. Para a taxa de aprendizado [*eta*], foram utilizados valores entre 0.01 e 0.3, e para a profundidade máxima das árvores [*max\_depth]*, os valores foram variados entre 1 e 10 (Ogunleye e Wang, 2019). Para garantir a ausência de *overfitting*, foi aplicada a técnica de k-fold cross-validation, com k igual a 10 durante o treinamento do *XGBoost*.

Assim como no modelo de *Random Forest,* na modelagem de *XGBoost* é importante verificar a importância de cada variável. Após a construção de um modelo *XGBoost* éproduzida uma matriz de importância que além do nome das variáveis contém o ganho, a *coverage* e a frequência. O ganho, que é a métrica mais importante das três indica quão significativa é uma variável na construção do modelo e na realização de previsões. Variáveis com valores de ganho mais altos exercem maior influência sobre as previsões do modelo. Já a *coverage* indica a extensão com que uma variável é utilizada na construção das árvores do modelo e, consequentemente, na determinação das previsões. Por fim, a frequência indica a frequência com que uma variável é utilizada para a tomada de decisões dentro das árvores do modelo, fornecendo um panorama de sua importância geral para a previsão (Abu-Rmileh, 2019).

O código, feito na linguagem R, para a criação do modelo *Random Forest* e *XGBoost* pode ser encontrado no seguinte link: https://github.com/Jorgepratesn/TCC.

**Avaliação dos Modelos**

Ao concluir o processo de modelagem, a avaliação torna-se crucial. Essa análise é conduzida tanto durante a execução de modelos com *Grid Search* quanto após o desenvolvimento completo, permitindo comparações entre eles. Neste estudo, a eficiência global do modelo [EGM] ou acurácia e a área abaixo da curva ROC [*AUC*] foram adotadas como métricas de avaliação, conforme explicado anteriormente. É importante ressaltar, também, que as métricas utilizadas para comparação entre modelos foram construídas no *test set. O*u seja, após o treino dos modelos no *training set* foram feitas predições na base de testes e essas são comparadas com o resultado real.

Além disso, na discussão dos resultados será considerado o tempo de processamento de cada modelo, uma vez que modelos que demandem mais tempo levam desvantagem em relação a modelos mais rápidos.

Segundo Belyadi e Haghighat (2021), o cálculo da acurácia, conforme a equação (7), pode ser sensível a problemas de desbalanceamento de classes em conjuntos de dados. Essa sensibilidade se manifesta especialmente quando há uma distribuição assimétrica, como o exemplo citado em que 90% das instâncias pertencem à classe 0 e apenas 10% à classe 1, como observado neste trabalho. Nesse contexto, os autores recomendam a utilização da matriz de confusão como uma ferramenta adicional para fornecer insights mais detalhados sobre o desempenho do modelo em relação à acurácia.

|  |  |
| --- | --- |
|  | (7) |

Uma representação visual da matriz de confusão pode ser observada na Figura 4. Essa matriz, com dimensões 4 x 4, apresenta os seguintes elementos, segundo Lalwani et al. (2022):

* Verdadeiros Positivos (Tp): Representa o número de clientes classificados corretamente como *churners* pelo modelo.
* Falsos Positivos (Fp): Indica o número de clientes erroneamente classificados como *churners* quando, na verdade, pertencem à categoria de *non-churners*.
* Falsos Negativos (Fn): Refere-se ao número de clientes que são, na verdade, *churners*, mas foram erroneamente classificados como *non-churners* pelo modelo.
* Verdadeiros Negativos (Tn): Representa o número de clientes corretamente classificados como *non-churners* pelo modelo.

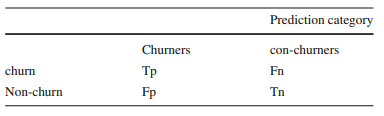


Figura 4. Matriz de confusão

Fonte: Lalwani et al. (2022)

Os modelos de classificação, contudo, não determinam diretamente se um cliente específico será classificado como evento ou não evento (no caso desse trabalho *churner* ou *non-churner)*. Em vez disso, eles fornecem a probabilidade de um cliente pertencer a cada uma das classes. Nesse contexto, é necessário estabelecer um ponto de corte, ou *cutoff*, que define a fronteira de decisão com base nas probabilidades calculadas pelo modelo. Por exemplo, ao definir um *cutoff* de 0.5, todos os clientes com probabilidade acima desse valor seriam classificados como evento (Fávero e Belfiore, 2017). Neste trabalho, os *cutoffs* foram selecionados utilizando a técnica de *Grid Search*, visando identificar aquele que resultasse na melhor acurácia do modelo. Após a definição do *cutoff*, é possível elaborar a matriz de confusão para uma avaliação mais detalhada.

Ainda com base na matriz de confusão, é possível derivar métricas importantes como a sensibilidade (percentual de acertos, para um determinado *cutoff*, apenas nas observações que são realmente eventos) e a especificidade (percentual de acertos, para um determinado *cutoff*, apenas nas observações que realmente não são eventos) (Mitrani, A. 2019). O segundo critério utilizado neste trabalho para comparar os modelos desenvolvidos é a área sob a curva ROC, que se utiliza desses conceitos.

Segundo Hoo et al. (2017), a construção de uma curva de *Receiver Operating Characteristic* (ROC) envolve o plot da taxa de sensibilidade no eixo y em relação à taxa de especificidade no eixo x. A curva ROC, portanto, oferece uma visualização do *trade-off* entre sensibilidade e especificidade. Uma curva ROC que segue a diagonal x=y indicaria que falsos negativos ocorrem com a mesma frequência que falsos positivos. Assim, espera-se que a curva esteja acima dessa diagonal imaginária, conforme representado pela linha azul na Figura 5. Quanto maior a área sob a curva ROC, com valores variando de 0 a 1, melhor é o desempenho do modelo uma vez que esse modelo apresentara maior eficiência geral de previsão, combinadas todas as possibilidades de *cutoff*.

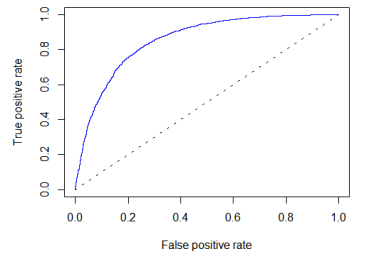


Figura 5. Curva ROC

Fonte: Campos (2018)

O código, feito na linguagem R, para a avaliação dos modelos pode ser encontrado no seguinte link: https://github.com/Jorgepratesn/TCC.

**Resultados Preliminares**

**Análise Preliminar**

Conforme descrito anteriormente, a fase inicial na construção de um modelo de previsão envolve a análise da base de dados. Essa análise começa com a limpeza da base, abordando inicialmente os valores nulos, seguido pela remoção de linhas contendo outliers. A Tabela 3. apresenta uma visão detalhada da quantidade de linhas processadas em cada etapa do processo de limpeza de dados.

Tabela 3. Quantidade de linhas tratadas por variável.

|  |  |  |
| --- | --- | --- |
| Coluna | Quantidade de valores nulos | Quantidade de *outliers* |
| *Tenure* | 264 | 2 |
| *PreferredLoginDevice* | 0 | 0 |
| *CityTier* | 0 | 0 |
| *WarehouseToHome* | 251 | 2 |
| *PreferredPaymentMode* | 0 | 0 |
| *Gender* | 0 | 0 |
| *HourSpendOnApp* | 255 | 0 |
| *NumberOfDeviceRegistered* | 0 | 0 |
| *PreferedOrderCat* | 0 | 0 |
| *SatisfactionScore* | 0 | 0 |
| *MaritalStatus* | 0 | 0 |
| *NumberOfAddress* | 0 | 4 |
| *Complain* | 0 | 0 |
| *OrderAmountHikeFromlastYear* | 265 | 0 |
| *CouponUsed* | 256 | 303 |
| *OrderCount* | 258 | 263 |
| *DaySinceLastOrder* | 307 | 3 |
| *CashbackAmount* | 0 | 0 |

Fonte: O Autor

Após a conclusão da limpeza dos dados, iniciou-se a análise preliminar, inicialmente focada nas variáveis quantitativas. Foram calculadas estatísticas descritivas para a base completa e, posteriormente, para as linhas de clientes *non-churn*. Essa abordagem permitiu identificar variáveis potencialmente mais relevantes para a construção dos modelos. Destacam-se três variáveis - *Tenure, DaySinceLastOrder* e *CashbackAmount* - que apresentaram discrepâncias significativas entre a média e a mediana da base geral em comparação com a base *non-churn*, conforme Tabela 4. Essas diferenças sugerem a hipótese de que essas variáveis terão importância significativa na criação dos modelos.

Tabela 4: Média e Mediana de três variáveis da base limpa

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variável | Média Geral | Média *non-churn* | Mediana Geral | Mediana *non-churn* |
| *Tenure* | 9,775 | 3,605 | 9,0 | 1,0 |
| *DaySinceLastOrder* | 4,148 | 2,823 | 3,0 | 2,0 |
| *CashbackAmount* | 173,8 | 157,2 | 160,6 | 149,0 |

Fonte: O Autor

Por fim foram plotados os gráficos de distribuição. Os gráficos das variáveis *Tenure* e *CashbackAmount* podem ser observados na Figura 6. Esses gráficos ajudam a visualizar algumas estatísticas como média e mediana das variáveis. Por exemplo, como exemplificado na Tabela 4. quanto menor o valor da variável *Tenure,* menor a chance de o cliente ser *churner*. Dessa forma, analisando o gráfico a esquerda da figura, é possível concluir que a empresa está com um bom desempenho em relação a essa variável. Para a variável *CashbackAmount,* por outro há um espaço maior de melhora, para diminuir os valores dados de *cashback* aos clientes, uma vez que clientes *non-churner* ganham menos cashback que a média.

|  |  |
| --- | --- |
|  |  |

Figura 6. Distribuição das Variáveis *Tenure e CashbackAmount*

Fonte: O Autor

No contexto das variáveis qualitativas, foi analisada a representatividade total de cada possível valor, juntamente com o percentual de *churn* associado a cada valor. Essa abordagem assemelha-se à análise realizada para as variáveis quantitativas, visando identificar aquelas que podem ter maior relevância na construção dos modelos. Duas variáveis qualitativas se destacam nessa análise - *PreferedOrderCat* e *Complain*.

Conforme evidenciado na Tabelas 5 a variável *Complain* apresenta variações significativas nos percentuais de clientes *churn*, dependendo do valor assumido por ela. Essas observações sugerem que essas variáveis podem ser relevantes na predição de *churn*.

Tabela 5: Percentual *Churn Complain*

|  |  |  |
| --- | --- | --- |
| Variável | Representatividade Total | Percentual de *Churn* |
| Não | 71% | 11% |
| Sim | 29% | 32% |

Fonte: O Autor

Assim como a variável *Complain,* a variável *PreferedOrderCat* também apresenta variações significativas nos percentuais de clientes *churn*, conforme Tabela 6.

Tabela 6: Percentual *Churn PreferedOrderCat*

|  |  |  |
| --- | --- | --- |
| Variável | Representatividade Total | Percentual de *Churn* |
| *Fashion* | 14% | 16% |
| *Grocery* | 6% | 3% |
| *Laptop & Accessory* | 37% | 10% |
| *Mobile* | 15% | 27% |
| *Mobile Phone* | 24% | 27% |
| *Others* | 4% | 6% |

Fonte: O Autor

Após a fase de análise preliminar, conforme descrito anteriormente, foi traçado a matriz de correlação (Figura 1), os dados foram padronizados, as variáveis qualitativas foram dummizadas e a base foi separada em *train* e *test* para a criação dos modelos.

**Regressão Logística**

O primeiro modelo desenvolvido foi o de regressão logística. Inicialmente, o modelo foi criado usando todas as variáveis, sem a aplicação da técnica *stepwise*. Posteriormente, verificou-se a significância estatística de cada variável no modelo por meio do teste z de Wald. Conforme Fávero e Belfiore (2017), esse teste envolve o cálculo da estatística z de Wald, conforme a equação (8), seguido pela avaliação do nível de confiança (valor-p da estatística z de Wald). Neste trabalho, adotou-se um nível de confiança de 95%. Assim, se o valor-p da estatística z de Wald for inferior a 0,05, pode-se concluir que a hipótese nula é rejeitada, indicando que a variável foi significativa para a construção do modelo.

|  |  |
| --- | --- |
|  | (7) |

Onde s.e significa o erro padrão do parâmetro em análise.

Após essa análise inicial, foi desenvolvido o modelo utilizando a técnica *stepwise.* Nesse modelo, somente variáveis que possuam significância estatística para o modelo foram consideradas no modelo final, as outras foram retiradas de forma iterativa, conforme explicado na seção anterior. Foram retiradas 9 variáveis ao todo do modelo durante a execução do *stepwise* foram elas*: MaritalStatus\_Married, SatisfactionScore\_2, PreferedOrderCat\_Mobile, PreferedOrderCat\_Grocery, PreferredPaymentMode\_E\_wallet, PreferredPaymentMode\_COD, CouponUsed, OrderAmountHikeFromlastYear, HourSpendOnApp.*

A avaliação da qualidade dos modelos será abordada mais adiante nesta seção. Contudo, no caso da técnica de regressão linear, onde foram gerados dois modelos, é essencial determinar qual será utilizado para comparação com outras técnicas. Nesse contexto, o parâmetro de área sob a curva ROC (AUC) para a base de teste foi empregado. No entanto, neste trabalho, não foi observada diferença significativa entre as AUCs de ambos os modelos. Por conseguinte, optou-se pelo uso do modelo completo devido à sua eficiência computacional, sendo mais rápido e exigindo menos recursos de memória. A falta de melhoria no desempenho do modelo *stepwise* em relação ao completo pode ser atribuída à possível exclusão de variáveis estatisticamente significativas, neutralizando os benefícios da remoção de variáveis sem tal significância. Mais adiante nessa seção também serão comparadas a importância de cada variável nos modelos construídos.

**Árvores de decisão**

Após a construção do modelo de regressão foram desenvolvidos os modelos baseados em arvores de decisão. O primeiro modelo desse tipo construído foi o *Random Forest*. Para a construção deste o primeiro passo foi fazer validação cruzada (*grid search cross-validation*) para identificar os hiperparametros ideias para o modelo em questão. Os valores testados foram detalhados na seção anterior. A combinação de hiperparametros a ser usada no modelo final foi escolhida com base na maior acurácia (conforme descrito na seção anterior, na parte de avaliação dos modelos) das iterações.

Os hiperparametros definidos para o modelo após o *grid search* foram: ntree = 500 ,*nsplit* = 10 e *mtry*=4.Os hiperparametros que não tiveram valores incluídos no *grid search* estão detalhados na seção anterior.

O segundo modelo baseado em arvores de decisão desenvolvido foi o *XGBoost.* Seguindo o mesmo raciocínio da *Random Forest,* foi feito uma validação cruzada para definição dos hiperparametros ideias. Os hiperparametros que não tiveram valores incluídos no *grid search* estão detalhados na seção anterior. Já o *eta* e o *max\_depth,* que foram incluídos na validação tiveram os valores definidos como 0,26 e 9, respectivamente. No caso do *XGBoost* a combinação de hiperparametros é escolhida com base na taxa de erro das iterações, conforme explicado na seção anterior.

**Avaliação dos Modelos**

Após a construção dos modelos é necessário avaliá-los conforme métricas discutidas na seção anterior. A Tabela 7, mostra por modelo o valor da área sob a curva ROC (AUC) e a acurácia dos modelos construídos. Importante ressaltar que em ambos os casos, quanto maior o valor das métricas melhor. Conforme explicitado na seção anterior, as métricas foram calculadas utilizando a base de teste.

Tabela 7: Avaliação dos Modelos

|  |  |  |
| --- | --- | --- |
| Modelo | AUC | Acurácia |
| Regressão | 0,9075 | 0,8979 |
| *Random Forest* | 0,9893 | 0,9673 |
| *XGBoost* | 0,9900 | 0,9810 |

Fonte: O Autor

Para complementar os resultados das métricas apresentados acima, é importante visualizar as matrizes de confusão dos modelos, conforme Tabela 8.

Tabelo 8: Matriz de Confusão

|  |  |  |
| --- | --- | --- |
| Regressão | *Random Forest* | *XGBoost* |
| |  |  |  | | --- | --- | --- | |  | *churn* | *non-churn* | | *churn* | *1056* | *108* | | *non-churn* | *26* | *123* | | |  |  |  | | --- | --- | --- | |  | *churn* | *non-churn* | | *churn* | *1063* | *24* | | *non-churn* | *19* | *207* | | |  |  |  | | --- | --- | --- | |  | *churn* | *non-churn* | | *churn* | *1071* | *14* | | *non-churn* | *11* | *217* | |

Fonte: O Autor

Analisando, por fim, os resultados é possível concluir que o modelo *XGBoost* é o mais adequado para a previsão de *churn* de clientes. Ele possui uma acurácia e uma AUC maior que os modelos de regressão e *Random Forest,* chegando muito próximo de 1 em ambos os casos, sendo que 1 é o valor obtido em um modelo perfeito. Além disso, por meio da analisa da matriz de confusão dos modelos pode-se concluir que o *XGBoost* tem mais clientes classificados como Verdadeiros Positivos e Verdadeiros Negativos que os outros modelos. Isso significa que o ele é melhor em todos os casos, tanto para prever o *churn* quanto para prever o *non-churn*.

O resultado encontrado nesse trabalho, no qual o modelo *XGBoost* tem um melhor desempenho está em linha com outros trabalhos publicados que comparam modelos de classificação. De acordo com Chen e Guestrin (2016), em uma determinada competição de machine learning das 29 edições, 17 foram ganhas por modelos *XGBoost,* o segundo lugar, ganhou apenas 11. Esses resultados, aplicados em uma vasta gama de problemas, mostram que o *XGBoost* é flexível e é o estado da arte de métodos de classificação, estando a frente de soluções mais modernas como *neural net*. Em um trabalho para comparar os melhores modelos para diagnostico de doença crônica no fígado, por exemplo, Ogunleye e Wang (2019), também chegam à conclusão de que o melhor modelo é o *XGBoost.*

**Importância das variáveis**

Para consolidar as análises de resultados, foi elaborada a Tabela 9, que destaca as variáveis mais importantes em cada modelo. Como o *XGBoost* foi identificado como o modelo mais adequado, a tabela está ordenada, apresentando a variável com o maior métrica *gain d*o modelo *XGBoost* no topo. Vale relembrar que cada modelo avalia a importância das variáveis de maneira distinta: a regressão utiliza a estatística Z, o *Random Forest* considera a importância de Gini, e o *XGBoost* utiliza o conceito de ganho. No caso da regressão, valores menores indicam maior significância na construção do modelo

Tabela 9: Importância das variáveis nos modelos.

|  |  |  |  |
| --- | --- | --- | --- |
| Variável | Regressão | *Random Forest* | *XGBoost* |
| *Tenure* | 2,32E-56 | 192,10 | 0,27 |
| *WarehouseToHome* | 2,63E-08 | 74,86 | 0,09 |
| *CashbackAmount* | 8,68E-07 | 97,93 | 0,09 |
| *DaySinceLastOrder* | 1,96E-10 | 64,15 | 0,07 |
| *Complain\_1* | 5,87E-60 | 66,00 | 0,07 |
| *NumberOfAddress* | 1,64E-26 | 61,70 | 0,07 |
| *OrderAmountHikeFromlastYear* | 7,79E-01 | 60,07 | 0,06 |
| *NumberOfDeviceRegistered* | 1,46E-09 | 39,31 | 0,03 |
| *MaritalStatus\_Single* | 1,47E-04 | 23,63 | 0,03 |
| *CityTier\_3* | 6,46E-10 | 21,86 | 0,02 |
| *SatisfactionScore\_5* | 3,61E-14 | 21,17 | 0,02 |
| *PreferedOrderCat\_Laptop\_\_Accessory* | 7,39E-13 | 16,28 | 0,02 |
| *OrderCount* | 5,50E-08 | 30,81 | 0,02 |
| *SatisfactionScore\_3* | 2,50E-04 | 16,30 | 0,02 |
| *Gender\_Male* | 4,66E-02 | 18,25 | 0,02 |
| *HourSpendOnApp* | 7,83E-01 | 20,81 | 0,01 |
| *SatisfactionScore\_4* | 1,04E-03 | 13,30 | 0,01 |
| *PreferredPaymentMode\_COD* | 5,06E-01 | 11,38 | 0,01 |
| *PreferredLoginDevice\_Mobile\_Phone* | 7,32E-03 | 14,41 | 0,01 |
| *PreferredPaymentMode\_E\_wallet* | 6,17E-01 | 12,47 | 0,01 |
| *PreferredPaymentMode\_Credit\_Card* | 5,05E-02 | 13,46 | 0,01 |
| *CouponUsed* | 6,39E-01 | 26,48 | 0,01 |
| *PreferredLoginDevice\_Phone* | 2,84E-04 | 12,52 | 0,01 |
| *PreferredPaymentMode\_Debit\_Card* | 1,02E-01 | 14,69 | 0,01 |
| *MaritalStatus\_Married* | 4,69E-02 | 14,27 | 0,01 |
| *PreferedOrderCat\_Mobile\_Phone* | 1,79E-02 | 16,73 | 0,00 |
| *PreferredPaymentMode\_UPI* | 4,00E-02 | 7,51 | 0,00 |
| *CityTier\_2* | 1,78E-04 | 5,67 | 0,00 |
| *PreferredPaymentMode\_CC* | 6,23E-02 | 5,11 | 0,00 |
| *PreferedOrderCat\_Mobile* | 1,70E-01 | 9,79 | 0,00 |
| *SatisfactionScore\_2* | 3,85E-01 | 6,12 | 0,00 |
| *PreferedOrderCat\_Grocery* | 3,75E-01 | 2,97 | 0,00 |

Fonte: O Autor

Analisando a Tabela 9 é possível concluir que as variáveis, tirando poucas exceções influenciam os modelos de maneira similar, sendo que nos 3 modelos *Tenure* foi a variável que mais influenciou a construção dos mesmos.

Além disso, conforme previsto na fase de análise preliminar as variáveis *Tenure*, *CashbackAmount*, *Complain e DaySinceLastOrder* foram altamente influentes na construção dos modelos.

**Referências**

Abu-Rmileh, A. 2019. The Multiple faces of ‘Feature importance’ in XGBoost. Disponível em: <https://towardsdatascience.com/be-careful-when-interpreting-your-features-importance-in-xgboost-6e16132588e7 >. Acesso em: 21 dez. 2023

Bai, Y.; Chen, M.; Zhou, P.; Zhao, T.; Lee, J.; Kakade, S.Xiong, C. 2021. How important is the train-validation split in meta-learning?. In: International Conference on Machine Learning. p. 543-553. PMLR.

Belyadi, H.; Haghighat, A. 2021. Machine Learning Guide for Oil and Gas Using Python: a step-by-step breakdown with data, algorithms, codes, and applications. Gulf Professional Publishing.

Biau, G.; Scornet, E. 2016. A random forest guided tour. Test, 25, 197-227.

Bruce, A.; Bruce, P. 2019. Estatística prática para cientistas de dadosb. Alta Books.

Buckinx, W.; Van den Poel, D. 2005. Customer base analysis: partial defection of behaviourally loyal clients in a non-contractual fmcg retail setting. European Journal of Operational Research, 164(1):252–268.

Campos, A. C. C. 2018. Understanding Customer Churn in Online Grocery Retail: an Inventory Management perspective.

Carvalho, H. M. 2014. Aprendizado de máquina voltado para mineração de dados: árvores de decisão. Monografia - Engenharia de Software. Universidade de Brasília. Brasilia, DF, Brasil.

Charbuty, B.; Abdulazeez, A. 2021. Classification based on decision tree algorithm for machine learning. Journal of Applied Science and Technology Trends, 2(01), 20-28.

Corrales, D. C.; Corrales, J. C.; Ledezma, A. 2018. How to address the data quality issues in regression models: A guided process for data cleaning. Symmetry, 10(4), 99.

Daoud, J. I. 2017. Multicollinearity and regression analysis. In Journal of Physics: Conference Series (Vol. 949, No. 1, p. 012009). IOP Publishing.

Dietterich, T. G. 2000. Ensemble methods in machine learning. In International workshop on multiple classifier systems (pp. 1-15). Berlin, Heidelberg: Springer Berlin Heidelberg.

Fávero, L. P.; Belfiore, P. 2017. Manual de análise de dados: estatística e modelagem multivariada com Excel®, SPSS® e Stata®. Elsevier Brasil.

Fridrich, M.; Dostál, P. 2022. User Churn Model in E-Commerce Retail. Scientific Papers of the University of Pardubice, Series D: Faculty of Economics and Administration, 30(1).

Gallo, A. 2014. The Value of Keeping the Right Customers. Disponível em: < https://hbr.org/2014/10/the-value-of-keeping-the-right-customers>. Acesso em: 07 dez. 2023

Grzymala-Busse, J. W.; Hu, M. 2001. A comparison of several approaches to missing attribute values in data mining. In Rough Sets and Current Trends in Computing: Second International Conference, RSCTC 2000 Banff, Canada, October 16–19, 2000 Revised Papers 2 (pp. 378-385). Springer Berlin Heidelberg.

Harrell, Jr, F. E.; Harrell, F. E. 2015. Binary logistic regression. Regression modeling strategies: With applications to linear models, logistic and ordinal regression, and survival analysis, 219-274.

Hoo, Z. H.; Candlish, J.; Teare, D. 2017. What is an ROC curve?. Emergency Medicine Journal, 34(6), 357-359.

King, J. E. 2008. Binary logistic regression. Best practices in quantitative methods, 358-384.

Kotsiantis, S. B. 2013. Decision trees: a recent overview. Artificial Intelligence Review, 39, 261-283.

Matuszelański, K.; Kopczewska, K. 2022. Customer Churn in Retail E-Commerce Business: Spatial and Machine Learning Approach. J. Theor. Appl. Electron. Commer. Res. 2022, 17, 165-198.

Menze, B. H.; Kelm, B. M.; Masuch, R.; Himmelreich, U.; Bachert, P.; Petrich, W.; Hamprecht, F. A. 2009. A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. BMC bioinformatics, 10, 1-16.

Mitrani, A. 2019. Evaluating Categorical Models II: Sensitivity and Specificity. Disponível em: <https://towardsdatascience.com/evaluating-categorical-models-ii-sensitivity-and-specificity-e181e573cff8#:~:text=Sensitivity%20is%20the%20metric%20that,apply%20to%20any%20categorical%20model. >. Acesso em: 19 jan. 2024

Lalwani, P.; Mishra, M. K.; Chadha, J. S.; Sethi, P. 2022. Customer churn prediction system: a machine learning approach. Computing, 1-24.

Lazarov, V.; Capota, M. 2007. Churn prediction. Bus. Anal. Course. TUM Comput. Sci, 33, 34.

Ogunleye, A.; Wang, Q. G. 2019. XGBoost model for chronic kidney disease diagnosis. IEEE/ACM transactions on computational biology and bioinformatics, 17(6), 2131-2140.

Patil, A. P.; Deepshika, M. P.; Mittal, S.; Shetty, S.; Hiremath, S. S.; Patil, Y. E. 2017. Customer churn prediction for retail business. In 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) (pp. 845-851). IEEE.

Rigatti, S. J. 2017. Random forest. Journal of Insurance Medicine, 47(1), 31-39.

Shaaban, E.; Helmy, Y.; Khedr, A.; Nasr, M. 2012. A proposed churn prediction model. International Journal of Engineering Research and Applications, 2(4), 693-697.

Song, Y. Y.; Ying, L. U. 2015. Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 27(2), 130.

Stripling, E.; vanden Broucke, S.; Antonio, K.; Baesens, B.; Snoeck, M. 2018. Profit maximizing logistic model for customer churn prediction using genetic algorithms. Swarm and Evolutionary Computation, 40, 116-130.

Witten, I. H.; Frank, E. 2002. Data mining: practical machine learning tools and techniques with Java implementations. Acm Sigmod Record, 31(1), 76-77.

Wohlwend, B. 2023. Decision Tree, Random Forest, and XGBoost: An Exploration into the Heart of Machine Learning. Disponível em: <https://medium.com/@brandon93.w/decision-tree-random-forest-and-xgboost-an-exploration-into-the-heart-of-machine-learning-90dc212f4948>. Acesso em: 19 dez. 2023