UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE SÃO JOÃO DA BOA VISTA

EDUARDO YUKIO MIAMOTO

Comparação de modelos de aprendizado de máquina para a predição de churn

São João da Boa Vista
2024

Eduardo Yukio Miamoto

Comparação de modelos de aprendizado de máquina para a predição de churn

Trabalho de Graduação apresentado ao Conselho de Curso de Graduação em Engenharia Eletrônica e de Telecomunicações do Campus de São João da Boa Vista, Universidade Estatual Paulista, como parte dos requisitos para obtenção do diploma de Graduação em Engenharia Eletrônica e de Telecomunicações .

Orientador: Profº Dr. Ivan Aritz Aldaya Garde

São João da Boa Vista
2024

M618c

Miamoto, Eduardo Yukio
Comparação de modelos de aprendizado de máquina para a predição de churn / Eduardo Yukio Miamoto. -- São João da Boa Vista, 2024
43 p. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Engenharia de Telecomunicações) - Universidade Estadual Paulista (UNESP), Faculdade de Engenharia, São João da Boa Vista
Orientador: Ivan Aritz Aldaya Garde

1. Inteligência artificial. 2. Aprendizado do computador. 3. Árvores de decisão. 4. Comércio eletrônico. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE ENGENHARIA - CÂMPUS DE SÃO JOÃO DA BOA VISTA
GRADUAÇÃO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

TRABALHO DE CONCLUSÃO DE CURSO

COMPARAÇÃO DE MODELOS DE APRENDIZADO DE MÁQUINA PARA A PREDIÇÃO DE CHURN

Aluno: Eduardo Yukio Miamoto
Orientador: Prof. Dr. Ivan Aritz Aldaya Garde

Banca Examinadora:

- Ivan Aritz Aldaya Garde (Orientador)
- Marlon Garcia Rodrigues (Examinador)
- Rita de Cassia Domingos (Examinadora)

Os formulários de avaliação e a ata da defesa, na qual consta a aprovação do trabalho, devidamente assinados pela banca encontram-se no prontuário eletrônico do aluno.

São João da Boa Vista, 03 de dezembro de 2024

Dedico este trabalho à minha família, a meus pais Ana e Milton, e meus irmãos Cristina e Henrique, que sempre foram minha base, meu alicerce e minha maior inspiração.

AGRADECIMENTOS

A conclusão desta etapa tão significativa em minha vida não seria possível sem o apoio e a contribuição de muitas pessoas que, de diferentes formas, marcaram minha trajetória e fizeram parte desta jornada.

Agradeço primeiramente a todos os servidores e docentes da UNESP que, com dedicação, paciência e excelência, foram fundamentais em minha formação acadêmica e pessoal. Cada aula, conselho e ensinamento contribuíram para moldar minha visão de mundo e consolidar minha paixão pelo aprendizado.

Quero expressar minha profunda gratidão aos meus colegas, que compartilharam comigo os desafios, as alegrias e as descobertas ao longo desta caminhada. A convivência com vocês tornou essa experiência ainda mais rica e inesquecível.

Em especial, dedico meus agradecimentos aos meus amigos Erik Liberato, Fernando Camargo, Matheus Damacena, Renan Cato, Sthephanie Liberato, Eduardo de Oliveira e Lucas Moreira. Obrigado por sempre estarem ao meu lado, por me incentivarem nos momentos difíceis e celebrarem comigo cada conquista. A amizade e o apoio de vocês foram a força que me impulsionou a seguir em frente e superar cada obstáculo.

A todos, meu mais sincero e profundo agradecimento.

“Não é o mais forte que sobrevive,
nem o mais inteligente, mas o que
melhor se adapta às mudanças.“
(Charles Darwin)

RESUMO

Este trabalho analisa e compara três algoritmos de aprendizado de máquina — árvore de decisão, floresta aleatória e *LightGBM* — com o objetivo de prever o churn de vendedores em um *marketplace* de *e-commerce*. O estudo utilizou um conjunto de dados real de transações para avaliar a capacidade preditiva dos modelos, considerando o desafio do desbalanceamento das classes. Para garantir a robustez dos resultados, a métrica AUC (Área Sob a Curva) foi adotada, evitando distorções comuns em dados desbalanceados. A pesquisa explorou a influência de diferentes volumes de dados na performance dos algoritmos, com testes realizados tanto na base de dados inicialmente analisada quanto em uma base expandida. Os melhores modelos foram selecionados com base em suas configurações de hiperparâmetros e avaliados em três amostras: treino, teste e fora do tempo (*out of time* OOT). O estudo conclui que a base expandida melhora o desempenho do *LightGBM*, enquanto o aumento de dados não altera significativamente os resultados dos outros modelos. Este trabalho fornece insights relevantes para estratégias de retenção em plataformas de *e-commerce*.

PALAVRAS-CHAVE: aprendizado de máquina; predição de churn; *e-commerce*; árvore de decisão; floresta aleatória; *lightgbm*; AUC.

ABSTRACT

This study analyzes and compares three machine learning algorithms—decision tree, random forest, and LightGBM—with the objective of predicting vendor churn in an e-commerce marketplace. The research utilized real transaction data to evaluate the predictive capabilities of these models, considering the challenges posed by class imbalance. To ensure robust results, the AUC (Area Under the Curve) metric was adopted, avoiding common distortions in imbalanced data. The study explored the influence of different data volumes on algorithm performance, with tests conducted on both the initially analyzed and an expanded dataset. The top-performing models were selected based on their hyperparameter configurations and evaluated across three samples: training, testing, and out-of-time (OOT). The study concludes that the expanded dataset significantly enhances the LightGBM's performance, while data increase does not substantially alter the results of the other models. This research offers valuable insights for retention strategies in e-commerce platforms.

KEYWORDS: machine learning; churn prediction; e-commerce; decision tree; random forest; lightGBM; AUC.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

UNESP	Universidade Estadual Paulista
OOT	Out of Time
AUC	*Area Under Curve*
LitghtGBM	*Light Gradient Boosting Machine*

SUMÁRIO

1 INTRODUÇÃO

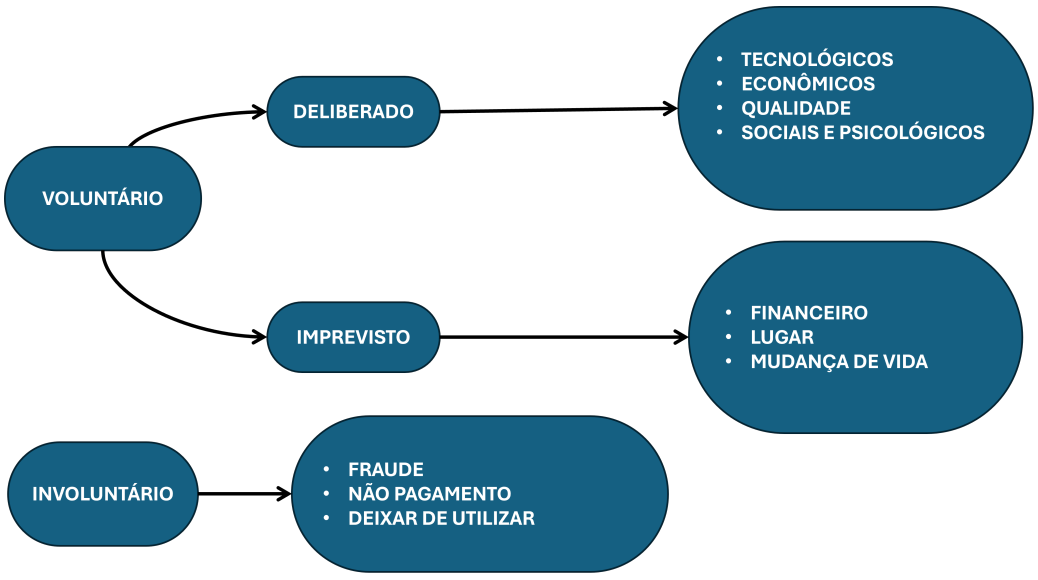
1.1 O CHURN

O termo churn refere-se à rotatividade de clientes, que é uma medida importante da fidelidade dos consumidores a uma empresa. De acordo com Strouse (1999) e Ferreira (2005), o churn representa a perda de clientes que migram para a concorrência, evidenciando a volatilidade das relações entre consumidores e empresas. Essa volatilidade está alinhada ao significado do verbo "*to churn*" na língua inglesa, que remete à agitação e à troca frequente de clientes entre diferentes fornecedores, o que leva as empresas a buscarem novas estratégias para manter sua base de clientes e conquistar os consumidores da concorrência (STROUSE, 1999; FERREIRA, 2005)

1.1.1 Tipos de Churn

Existem dois tipos de churn o involuntário e voluntário como ilustrado na Figura 1. O churn involuntário ocorre quando um cliente não consegue cumprir suas obrigações financeiras, levando a empresa a interromper a prestação de serviços. Já o churn voluntário acontece quando o cliente decide encerrar o relacionamento por razões que podem ser acidentais, como mudança de endereço, ou deliberadas, como a busca por melhores ofertas na concorrência. A previsão de churn envolve a identificação dos clientes com maior probabilidade de abandonar a empresa, permitindo ações preventivas Mattison (2005) enfati dos clientes, incluindo suas motivações iniciais e razões para mudança. Embora o preço seja um fator crucial para a perda de clientes, o processo de mudança é complexo e envolve vários aspectos, como qualidade do serviço e fatores tecnológicos. za que, ao tentar entender os motivos do churn, muitas vezes as empresas não analisam adequadamente o perfil

Figura 1 – Tipos de churn



Fonte: Elaborado pelo autor

1.2 CONTEXTUALIZAÇÃO

Com a crescente digitalização dos mercados, a análise de churn tem se mostrado uma estratégia essencial para a retenção de clientes, especialmente em ambientes altamente competitivos, como o *e-commerce*. Inicialmente mais comum em setores de telecomunicações e serviços financeiros, o conceito de churn, que reflete a taxa de cancelamento ou abandono de clientes, tem se expandido para o comércio eletrônico, onde a perda de clientes pode significar uma significativa diminuição da receita e da base de usuários. Assim, identificar e prever comportamentos que indiquem a probabilidade de churn tornou-se uma prática crucial para empresas que buscam otimizar a experiência e a fidelidade de seus clientes (LEMMENS; CROUX, 2006).

O crescimento exponencial do *e-commerce* no Brasil evidencia a relevância dessas estratégias. De acordo com o Ministério do Desenvolvimento, Indústria, Comércio e Serviços (MDIC), o setor movimentou R$ 196,1 bilhões em 2023, representando um aumento de 4,8% em relação ao ano anterior. Esse crescimento reflete a consolidação do comércio eletrônico como um dos principais canais de consumo no país. Desde 2016, o setor mais que quintuplicou em volume de negócios, demonstrando sua capacidade de transformação e adaptação ao comportamento do consumidor digital (DESENVOLVIMENTO INDÚSTRIA, 2024).

Além disso, conforme destaca a Universidade *Marketplaces*, o setor passou de R$ 35 bilhões em 2016 para impressionantes R$ 196,1 bilhões em 2023, demonstrando um aumento contínuo e consistente ao longo dos anos. Esse desempenho foi impulsionado por fatores como maior acesso à internet, confiança nas plataformas digitais e a diversificação de produtos e serviços. Nesse cenário de crescimento acelerado, entender os padrões de comportamento dos clientes, incluindo o risco de abandono, tornou-se não apenas uma vantagem competitiva, mas uma necessidade para a sustentabilidade das empresas no mercado digital.

Nesse contexto, o avanço do aprendizado de máquina tem desempenhado um papel crucial, trazendo novas abordagens para a previsão de churn, com modelos como árvore de decisão, floresta aleatória e *LightGBM* ganhando destaque por sua capacidade de detectar padrões comportamentais em grandes volumes de dados(BREIMAN, 2001; FRIEDMAN, 2001). Esses algoritmos se destacam por oferecer previsões precisas e adaptáveis, sendo eficazes tanto para dados balanceados quanto para conjuntos de dados onde a classe de churn é desbalanceada. Modelos como o *LightGBM*, por exemplo, têm sido preferidos em cenários de grande escala devido à sua capacidade de sintonização e rapidez, especialmente em comparação com algoritmos tradicionais de aumento de gradiente(KE et al., 2017).

A avaliação de modelos preditivos em churn demanda métricas robustas que sejam capazes de diferenciar corretamente entre clientes com alta e baixa probabilidade de abandono. Nesse contexto, a métrica de AUC (Área Sob a Curva ROC) tem se mostrado particularmente útil, já que oferece uma visão mais equilibrada do desempenho do modelo em cenários de desbalanceamento, comuns na predição de churn em *e-commerce* (HANLEY; MCNEIL, 1982). A utilização de métricas como a AUC, além de permitir uma análise mais acurada, ajuda a evitar distorções que podem surgir com o uso de medidas de acurácia simples, tornando a avaliação mais confiável em aplicações práticas (JENI; COHN; TORRE, 2013).

A adaptação de modelos de aprendizado de máquina à previsão de churn também exige técnicas

de ajuste de hiperparâmetros, uma etapa essencial para maximizar o desempenho dos algoritmos. Abordagens como Random Search e otimização bayesiana têm sido amplamente empregadas para essa tarefa, proporcionando uma exploração mais eficiente do espaço de hiperparâmetros em comparação com métodos tradicionais como o Grid Search (BERGSTRA; BENGIO, 2012). Esses métodos de ajuste permitem que os modelos de aprendizado de máquina atinjam um desempenho otimizado em condições reais, favorecendo a generalização dos resultados em novos dados, o que é crítico para prever comportamentos de churn em ambientes dinâmicos como o *e-commerce* (PROBST; BOULESTEIX; BISCHL, 2019).

A justificativa para este trabalho fundamenta-se na importância estratégica da retenção de clientes em ambientes de *e-commerce*, um setor em que a competitividade é crescente e a fidelidade dos clientes e vendedores desempenha papel crucial para o sucesso das plataformas. A capacidade de prever o churn, isto é, a probabilidade de abandono, tornou-se um diferencial valioso para empresas que buscam otimizar suas estratégias de retenção. Além de impactar diretamente na receita, a perda de clientes e vendedores também afeta a reputação e o posicionamento de mercado, uma vez que indica potenciais falhas no atendimento ou nas vantagens oferecidas pela plataforma.

Esse cenário motiva a aplicação de técnicas de aprendizado de máquina na análise de churn, que permite processar e interpretar grandes volumes de dados de forma eficaz e precisa. Os algoritmos de aprendizado supervisionado, como árvore de decisão, floresta aleatória e *LightGBM*, são especialmente relevantes para essa tarefa, pois conseguem identificar padrões de comportamento que indicam a propensão ao abandono. Esses modelos não apenas facilitam a identificação de possíveis causas do churn, mas também possibilitam a criação de intervenções personalizadas para aumentar a retenção de forma proativa.

O *e-commerce*, em particular, beneficia-se dessas análises preditivas devido à sua capacidade de coleta de dados detalhados sobre o comportamento dos usuários, incluindo histórico de compras, padrões de navegação e interações com a plataforma. No entanto, o desafio do desbalanceamento dos dados, onde casos de churn são geralmente menos frequentes, exige o uso de métricas robustas, como a AUC, para avaliar a eficácia dos modelos. A escolha cuidadosa de algoritmos e métricas adequadas é fundamental para assegurar que as previsões sejam confiáveis e aplicáveis na prática, maximizando assim o valor das estratégias de retenção.

Dessa forma, o presente trabalho contribui tanto para o avanço das técnicas de análise de churn quanto para a prática empresarial no mercado digital, onde a retenção é uma vantagem competitiva essencial.

1.3 OBJETIVOS

Este trabalho tem como objetivo comparar três algoritmos de aprendizado de máquina para prever o churn de vendedores assinantes de um mercado virtual: árvore de decisão, floresta aleatória e *LightGBM*. A análise será conduzida utilizando a métrica de área sob a curva (AUC) para avaliar o desempenho dos modelos na classificação e previsão de churn. O estudo busca identificar o modelo mais eficaz e fornecer ideas que possam contribuir para a melhoria das estratégias de retenção de

vendedores, além de ampliar o entendimento sobre a aplicação de técnicas de aprendizado de máquina em mercados virtuais.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está estruturado em seis capítulos. O primeiro capítulo introduz o tema e estabelece o contexto da pesquisa. O segundo capítulo apresenta uma revisão literária abrangente sobre o uso de aprendizado de máquina na previsão de churn, destacando os algoritmos que serão comparados. No terceiro capítulo, a metodologia adotada para conduzir a pesquisa é detalhada, especificando os conjuntos de dados utilizados, as métricas de avaliação, e o processo de treinamento dos modelos. O quarto capítulo apresenta os resultados da pesquisa, seguidos pelo quinto capítulo, que discute as implicações desses resultados e suas aplicações práticas. Finalmente, o sexto capítulo conclui o trabalho

2 CONCEITOS TEÓRICOS

2.1 APRENDIZADO DE MÁQUINA

O aprendizado de máquina é uma área da inteligência artificial focada em algoritmos que permitem aos sistemas aprender e melhorar a partir de dados, sem a necessidade de serem explicitamente programados para cada tarefa específica. Os algoritmos de aprendizado de máquina são geralmente classificados em duas categorias principais: aprendizado supervisionado e não supervisionado. No aprendizado supervisionado, o modelo é treinado com dados rotulados, onde cada exemplo no conjunto de dados possui uma resposta conhecida, ou rótulo, permitindo ao algoritmo aprender as relações entre variáveis de entrada e saída. Este tipo de aprendizado é amplamente utilizado em problemas de classificação e regressão, como a predição de churn (ou evasão de clientes), onde o objetivo é prever se um cliente está propenso a cancelar um serviço (MURPHY, 2012).

Já o aprendizado não supervisionado lida com dados sem rótulos, focando na identificação de padrões ou estruturas ocultas dentro dos dados, como agrupamentos de comportamento, o que é valioso para segmentação de clientes em diferentes perfis (GARETH et al., 2013). Embora o churn seja, em sua essência, um problema de classificação binária (churn ou não churn) e geralmente seja abordado por algoritmos supervisionados, algumas abordagens híbridas utilizam métodos não supervisionados para entender características ou perfis de clientes que, em seguida, podem ajudar na predição de churn (GOODFELLOW, 2016).

Como este trabalho está focado na predisão de churn, será abordada apenas a árvore de decisão aplicada à classificação

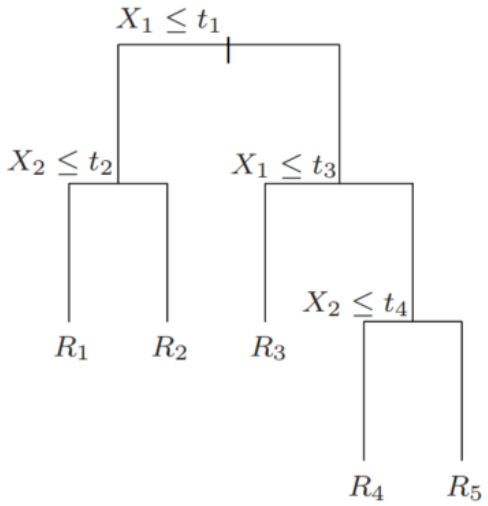
Com o crescimento dos dados no setor de *e-commerce*, o uso de técnicas de aprendizado de máquina supervisionadas tem sido essencial para prever com precisão a probabilidade de churn, identificando comportamentos críticos antes da ocorrência real, permitindo que as empresas atuem de forma proativa para reter clientes (KELLEHER; NAMEE; D'ARCY, 2020). O aprendizado de máquina continua evoluindo, expandindo-se para incluir modelos complexos e abordagens híbridas que combinam aprendizado supervisionado e não supervisionado, aprimorando a eficácia de sistemas preditivos (ALPAYDIN, 2020).

2.2 ALGORITMOS DE APRENDIZADO DE MÁQUINA UTILIZADOS

2.2.1 Árvore de Decisão

As árvores de decisão (*Decision Trees*, DTs) são métodos de aprendizado supervisionado não paramétricos usados para classificação e regressão. O objetivo é criar um modelo que preveja o valor de uma variável alvo por meio de regras de decisão simples inferidas das características dos dados. Esses métodos segmentam o espaço dos preditores em várias regiões simples e, para fazer uma previsão para uma determinada observação, normalmente utilizam a média ou o modo das observações de treinamento na região correspondente.

Figura 2 – Árvore de decisão



Fonte: Retirada de (HASTIE; TIBSHIRANI; FRIEDMAN, 2009)

Os métodos baseados em árvores particionam o espaço das características em um conjunto de retângulos e, em seguida, ajustam um modelo simples (como uma constante) em cada um. Apesar de serem conceitualmente simples, eles são bastante poderosos. O método CART (*Classification and Regression Trees*) é amplamente utilizado tanto para regressão quanto para classificação.

Para ilustrar, considere um problema de regressão com uma resposta contínua e entradas e , cada uma variando entre 0 e 1. Uma abordagem para simplificar a modelagem é restringir a atenção a particionamentos binários recursivos. Primeiramente, o espaço é dividido em duas regiões e a resposta é modelada pela média de em cada região. Escolhe-se a variável e o ponto de divisão para obter o melhor ajuste. Esse processo continua até que um critério de parada seja alcançado assim como ilustrado na Figura 2 (HASTIE; TIBSHIRANI; FRIEDMAN, 2009; GARETH et al., 2013; SCIKIT-LEARN. . . ,).

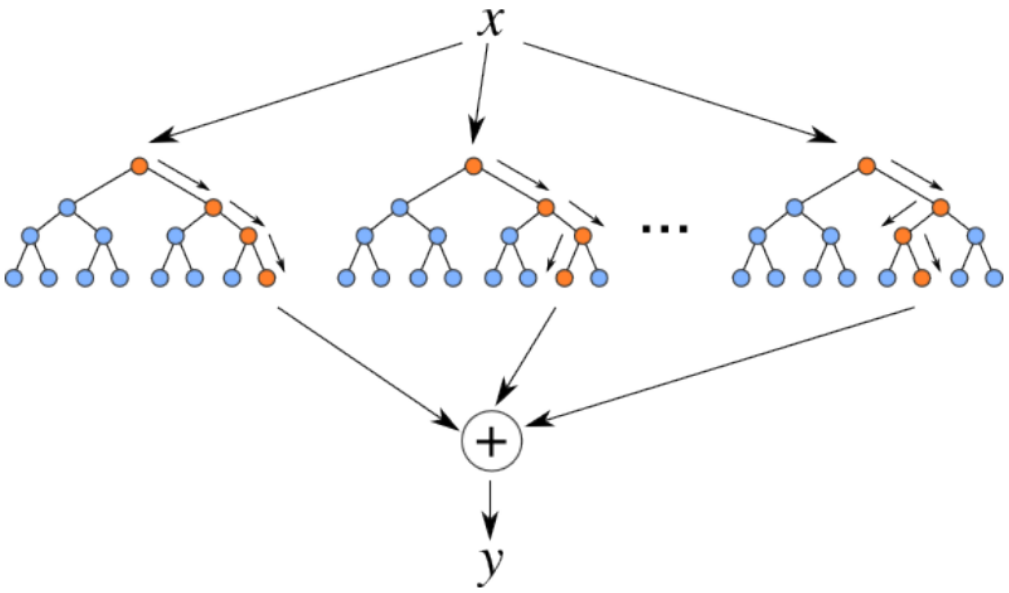
2.2.1.1 *Árvore de Classificação*

Uma árvore de classificação é um método de aprendizado supervisionado utilizado para prever respostas qualitativas. Para uma observação, a previsão é baseada na classe mais frequente das observações de treinamento que pertencem ao mesmo nó terminal. Além da classe prevista, também é importante considerar as proporções de cada classe entre as observações de treinamento que caem em uma determinada região do nó terminal. O processo de construção de uma árvore de classificação utiliza divisões binárias recursivas para crescer a árvore. Diferente de outras abordagens, a taxa de erro de classificação, o índice de Gini e a entropia são utilizadas como critérios de divisão (GARETH et al., 2013).

2.2.2 Floresta Aleatória

O segundo algoritmo escolhido foi a Floresta Aleatória, um poderoso método de aprendizado de máquina do tipo *ensemble*, que combina múltiplos modelos para criar previsões mais robustas. A Floresta Aleatória trabalha criando múltiplas Árvores de Decisão a partir de amostras aleatórias dos

Figura 3 – Ilustração do algoritmo floresta aleatória



Fonte:Retirada de (TELOKEN et al., 2016)

dados de treinamento. Cada árvore é construída de forma independente e é ligeiramente diferente das outras devido à aleatoriedade introduzida tanto na seleção dos dados quanto na escolha dos atributos que serão utilizados em cada nó da árvore. Essa aleatoriedade é essencial para reduzir a correlação entre as árvores e, consequentemente, melhorar a precisão do modelo final (MÜLLER; GUIDO, 2016; MOENARDY; ARIFIN; KUMADJI, 2016).

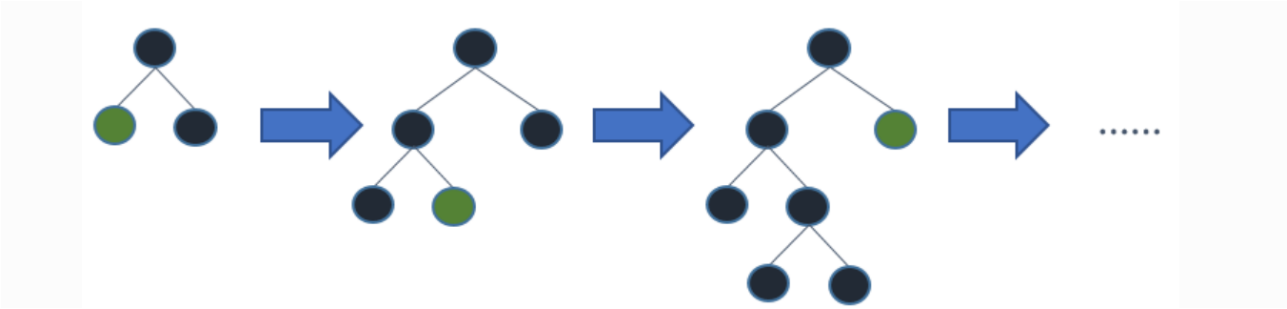
O processo de construção de uma Floresta Aleatória pode ser resumido em quatro etapas principais: primeiramente, uma amostra aleatória dos dados de treinamento é selecionada com reposição, um método conhecido como bootstrapping. Em seguida, uma Árvore de Decisão é criada a partir dessa amostra, onde, em cada nó, é escolhida aleatoriamente uma característica do conjunto de dados para fazer a divisão. Esse processo é repetido várias vezes para gerar múltiplas árvores (Figura 3). Por fim, a previsão final do modelo é determinada pela média ou pelo resultado mais frequente entre todas as árvores (TELOKEN et al., 2016; PARK; KIM, 2019).

2.2.3 LightGBM

O terceiro algoritmo escolhido foi o *LightGBM* (*Light Gradient Boosting Machine*), uma variante do *Gradient Boosting* desenvolvida pela Microsoft. O *LightGBM* foi projetado para lidar com grandes conjuntos de dados, oferecendo alta velocidade de treinamento e precisão preditiva. Conforme descrito por Yan et al. (2021), o *LightGBM* destaca-se por sua eficiência computacional, precisão e estabilidade, atributos que o tornam ideal para tarefas que exigem processamento rápido e eficiente de grandes volumes de dados.

Uma das principais características do *LightGBM* é o seu método de crescimento de árvores, que difere dos outros algoritmos. Enquanto a maioria dos algoritmos de árvore de decisão cresce horizontalmente, ou seja, em termos de nível (*level-wise tree growth*), o *LightGBM* adota um crescimento

Figura 4 – *Light gradient boosting machine*



Fonte:Retirada de (Microsoft Corporation, 2021)

vertical, em termos de folhas (*leaf-wise tree growth*). Isso significa que o *LightGBM* seleciona a folha com maior perda para continuar o crescimento, o que permite uma redução mais eficaz das perdas do que o crescimento horizontal de árvores. Essa técnica de crescimento, como ilustra a Figura 4, permite que o *LightGBM* atinja um desempenho superior, especialmente em grandes conjuntos de dados (KE et al., 2017; Microsoft Corporation, 2021).

2.3 PREDIÇÃO DE CHURN

A predição de churn, ou taxa de evasão, refere-se ao processo de identificar quais clientes têm maior probabilidade de interromper o uso de um serviço ou produto, ajudando as empresas a desenvolver estratégias eficazes de retenção. Inicialmente aplicada em setores como telecomunicações e bancos, a análise de churn tornou-se essencial no *e-commerce* devido ao aumento da concorrência e à necessidade de manter a fidelização dos clientes (HADDEN et al., 2007).

No contexto digital, a predição de churn permite que empresas reconheçam padrões comportamentais de insatisfação e engajamento reduzido que podem levar ao abandono da plataforma. Estes padrões incluem, por exemplo, uma diminuição na frequência de compras, quedas no valor médio de transações e respostas mais demoradas aos estímulos de marketing (GLADY; BAESENS; CROUX, 2009). Ao prever quais clientes estão em risco de churn, as empresas podem direcionar recursos para intervenções eficazes, priorizando a retenção daqueles com maior probabilidade de evasão e aumentando o valor de vida do cliente (*Customer Lifetime Value* - CLV) (BUCKINX; POEL, 2005).

Com a utilização de algoritmos de aprendizado de máquina, a predição de churn vai além dos métodos estatísticos tradicionais, possibilitando uma análise de grandes volumes de dados para identificação de padrões complexos. A implementação de modelos preditivos para churn permite que empresas não apenas identifiquem quais clientes estão propensos a abandonar, mas também entendam melhor os fatores que impulsionam a retenção e o engajamento (TSAI; LU, 2009). Essas informações são particularmente valiosas para o *e-commerce*, onde a competitividade exige estratégias de fidelização robustas e eficazes para maximizar o valor do cliente (VERHOEF; LEMON, 2013).

3 METODOLOGIA

3.1 FERRAMENTAS UTILIZADAS

Para o desenvolvimento deste trabalho, foram adotadas tecnologias que garantem robustez e eficiência, aspectos essenciais para a análise e predição de churn. Na manipulação e análise de dados com Python, foi utilizada a biblioteca pandas, amplamente reconhecida pela sua capacidade de lidar com grandes volumes de informações e realizar operações estatísticas de maneira eficiente. A criação e aplicação de modelos de aprendizado de máquina contaram com scikit-learn, que possibilitou a implementação dos algoritmos de árvore de decisão e floresta aleatória. O treinamento e otimização do *LightGBM* foram realizados com a biblioteca específica lightgbm, projetada para garantir desempenho elevado nesse tipo de tarefa. Para a visualização dos resultados, matplotlib foi a biblioteca escolhida, oferecendo ferramentas robustas para a criação de gráficos informativos e bem elaborados.

No que diz respeito à gestão dos dados, SQLite desempenhou um papel central no processo de ETL (Extração, Transformação e Carga), sendo escolhido por sua leveza e eficiência, características que o tornam ideal para o armazenamento e manipulação de dados de forma ágil. Para viabilizar a integração com o ambiente de desenvolvimento em Python, SQLAlchemy foi utilizado, garantindo uma interação fluida e consistente entre os bancos de dados e as ferramentas do projeto. Além disso, SQL teve um papel indispensável na execução de consultas e manipulações de dados, permitindo o acesso rápido e organizado às informações necessárias durante todo o desenvolvimento do trabalho.

3.1.1 Hiperparâmetros

Os hiperparâmetros são elementos fundamentais em modelos de aprendizado de máquina, representando configurações que não podem ser ajustadas diretamente pelos dados de treinamento. Em vez disso, eles precisam ser definidos antes do processo de aprendizado, pois desempenham um papel crucial na performance do modelo. Exemplos de hiperparâmetros incluem o número de iterações, o termo de regularização e a profundidade de uma árvore de decisão. O ajuste desses parâmetros, conhecido como "*tuning*", é essencial para sintonizar a capacidade preditiva do modelo (PROBST; BOULESTEIX; BISCHL, 2019; YANG; SHAMI, 2020);

Há duas abordagens principais para a escolha dos hiperparâmetros: a manual e a automática. A escolha manual exige um entendimento profundo sobre como os hiperparâmetros influenciam a generalização do modelo, frequentemente exigindo múltiplas iterações de treinamento e validação para identificar as melhores configurações. Por outro lado, a seleção automática de hiperparâmetros, embora simplifique o processo ao eliminar a necessidade de um conhecimento profundo sobre o modelo, demanda maior poder computacional para explorar as diferentes combinações possíveis (MANTOVANI et al., 2024).

A tarefa de tunar os hiperparâmetros é frequentemente tratada como um problema de sintonização de caixa-preta, onde o objetivo é encontrar a configuração de hiperparâmetros que maximize a performance do modelo em um dado conjunto de dados. Diversas técnicas foram desenvolvidas para

esse fim, cada uma buscando sintonizar o processo de ajuste de forma eficiente e eficaz (YANG; SHAMI, 2020; HUTTER; KOTTHOFF; VANSCHOREN, 2019).

Essas técnicas variam desde abordagens simples, como a busca em grade, até métodos mais sofisticados, como sintonização bayesiana e algoritmos evolutivos. Independentemente da abordagem, o sucesso do ajuste de hiperparâmetros depende da escolha correta da técnica, bem como da compreensão do impacto que cada hiperparâmetro tem no modelo em questão.

Os hiperparâmetros sintonizados foram:

- **min_samples_split**: Número mínimo de amostras necessárias para dividir um nó interno. Esse hiperparâmetro pode ser ajustado nos modelos de árvore de decisão e floresta aleatória. A variável aceita é do tipo **int** ou **float**.

- **max_depth**: Profundidade máxima da árvore. Se definido como **None**, os nós serão expandidos até que todas as folhas sejam puras ou contenham menos que **min_samples_split** amostras. Este hiperparâmetro pode ser configurado em árvore de decisão, floresta aleatória e *LightGBM*. A variável utilizada é do tipo **int**.

- **splitter**: Estratégia usada para selecionar a divisão em cada nó. As opções incluem "melhor"para escolher a melhor divisão e "aleatória"para selecionar uma divisão aleatoriamente. Aplicável a árvore de decisão. A variável aceita é do tipo **string**.

- **min_samples_leaf**: Número mínimo de amostras que deve estar presente em um nó folha. Configurável em árvore de decisão, floresta aleatória e *LightGBM*. A variável utilizada é do tipo **int** ou **float**.

- **max_features**: Número de features a serem consideradas ao procurar a melhor divisão. Aplicável em árvore de decisão, floresta aleatória e *LightGBM*. A variável aceita é do tipo **int**, **float**, ou ainda outras opções como **"auto"**, **"sqrt"** e **"log2"**.

- **random_state**: Controla a aleatoriedade dos resultados, permitindo a reprodutibilidade. Utilizado em árvore de decisão, floresta aleatória e *LightGBM*. A variável utilizada é do tipo **int**.

- **class_weight**: Pesos associados às classes no formato (**class_label: weight**). Se não fornecido, todas as classes têm peso igual. Aplicável em árvore de decisão e floresta aleatória. A variável aceita é um **dicionário**, uma **lista de dicionário**, **"balanced"** ou **None**.

- **learning_rate**: Taxa de aprendizado (η), responsável por controlar a influência de cada nova iteração no aprendizado do modelo. Este hiperparâmetro é específico para o *LightGBM*. A variável utilizada é do tipo **float**.

- **num_leaves**: Controla o número máximo de folhas a crescer em cada iteração, impactando diretamente na complexidade do modelo. Específico para o *LightGBM*. A variável utilizada é do tipo **int**.

- **n_estimators**: Número de estimadores ou iterações no processo de aumento de gradiente. Controla quantas árvores são cultivadas durante o treinamento. Aplicável em floresta aleatória e *LightGBM*. A variável utilizada é do tipo **int**.

- **min_child_samples:** Este parâmetro define o número mínimo de amostras que cada folha deve ter. Valores mais altos previnem o modelo de aprender padrões muito específicos (*overfitting*). Esse hiperparâmetro é utilizado no Ligthgbm. A variável aceita é do tipo **float**.

- **colsample_bytree:** define a fração de colunas (features) a serem usadas em cada árvore. É útil para aumentar a variabilidade do modelo e evitar overfitting.). Esse hiperparâmetro é utilizado no Ligthgbm. A variável aceita é do tipo **float**.

3.1.2 Sintonização dos hiperparâmetros

A sintonização de hiperparâmetros desempenha um papel fundamental no desempenho de modelos de aprendizado de máquina, uma vez que os valores corretos dos parâmetros podem melhorar significativamente a precisão e a generalização do modelo. Existem diversas abordagens para essa sintonização, e neste trabalho, a técnica de Random Search foi adotada devido à sua eficiência em termos de tempo e recursos computacionais.

O *Grid Search* é um método tradicional para a busca de hiperparâmetros. Nele, uma grade com valores pré-determinados para cada hiperparâmetro é criada e todas as combinações possíveis desses valores são testadas. Essa abordagem pode ser eficiente em espaços de baixa dimensionalidade, mas conforme o número de hiperparâmetros e os valores potenciais aumentam, o número de combinações cresce exponencialmente, tornando o Grid Search impraticável em termos de tempo e custo computacional (BERGSTRA; BENGIO, 2012).

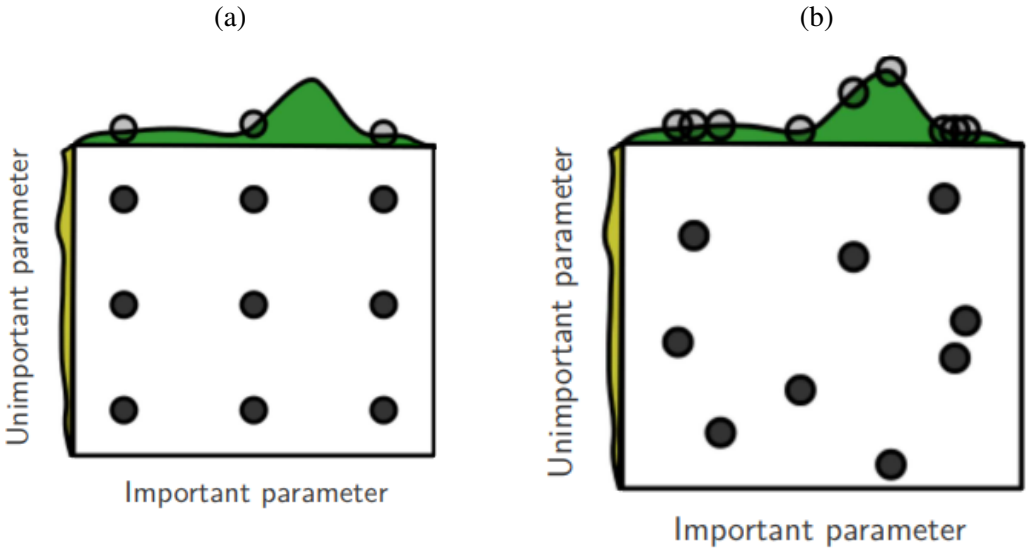
Devido a essas limitações e ao tempo reduzido para a realização dos experimentos, optou-se por utilizar o *Random Search* como estratégia de sintonização. O *Random Search*, conforme descrito por Bergstra e Bengio (2012), consiste em amostrar aleatoriamente combinações de hiperparâmetros dentro de um intervalo predefinido. Ao contrário do Grid Search, ele não testa todas as combinações possíveis, mas explora o espaço de hiperparâmetros de maneira mais ampla e eficiente.

Além das diferenças teóricas, a Figura 5 ilustra visualmente as estratégias de busca adotadas pelo *Grid Search* e pelo *Random Search*:

Na Figura 5a, que representa o *Grid Search*, as combinações de hiperparâmetros são distribuídas de forma regular e uniforme pelo espaço de busca, sem levar em consideração a importância dos hiperparâmetros. Esse método pode resultar em um uso ineficiente dos recursos, pois ele explora áreas do espaço de busca onde a variação dos parâmetros tem pouco ou nenhum impacto no desempenho do modelo.

Já na Figura 5b, que representa o *Random Search*, mostra uma distribuição mais dispersa e aleatória dos pontos. Embora possa parecer menos estruturado, o Random Search se beneficia da aleatoriedade para explorar de maneira mais eficaz diferentes regiões do espaço de hiperparâmetros.

Figura 5 – Comparação dos métodos. (a) *Grid Layout*. (b) *Random Layout*



Fonte: Retirada de (BERGSTRA; BENGIO, 2012)

3.2 DESCRIÇÃO DOS DADOS

Para a realização deste trabalho, foi utilizada a base de dados *Brazilian E-Commerce Public Dataset by Olist*, disponível no Kaggle. Essa base oferece uma visão abrangente do funcionamento de um mercado virtual. Ela é composta por diversas tabelas inter-relacionadas, ilustradas na Figura 6. Entre essas tabelas estão: geolocalização, itens, pagamentos, avaliações, pedidos, produtos e vendedores, que serão explicadas posteriormente. No total, o conjunto de dados abrange informações sobre cem mil pedidos realizados entre 2016 e 2018 (OLIST, 2018).

3.2.1 Tabelas

- Itens: Armazena detalhes sobre os produtos incluídos em cada pedido, como o ID do produto, preço, quantidade e a posição do item no pedido. Esta tabela é crucial para entender o comportamento de compra, como a diversidade de produtos adquiridos por pedido.
- Pagamentos: Contém informações sobre os pagamentos realizados pelos clientes, incluindo o método de pagamento (cartão de crédito, débito, boleto ou *voucher*), número de parcelas e o valor total pago. Isso permite análises sobre as preferências de pagamento e a relação entre métodos de pagamento e o valor do pedido.
- Avaliações: Registra as avaliações dadas pelos clientes após a conclusão dos pedidos, com notas que variam de 1 a 5. Também inclui comentários dos clientes. Esta tabela é fundamental para estudos de satisfação do cliente e para identificar possíveis problemas na experiência de compra.
- Pedidos: É a tabela central que conecta todas as outras. Contém informações gerais sobre cada pedido, como a data de compra, data de entrega estimada e data de entrega real. A partir desta

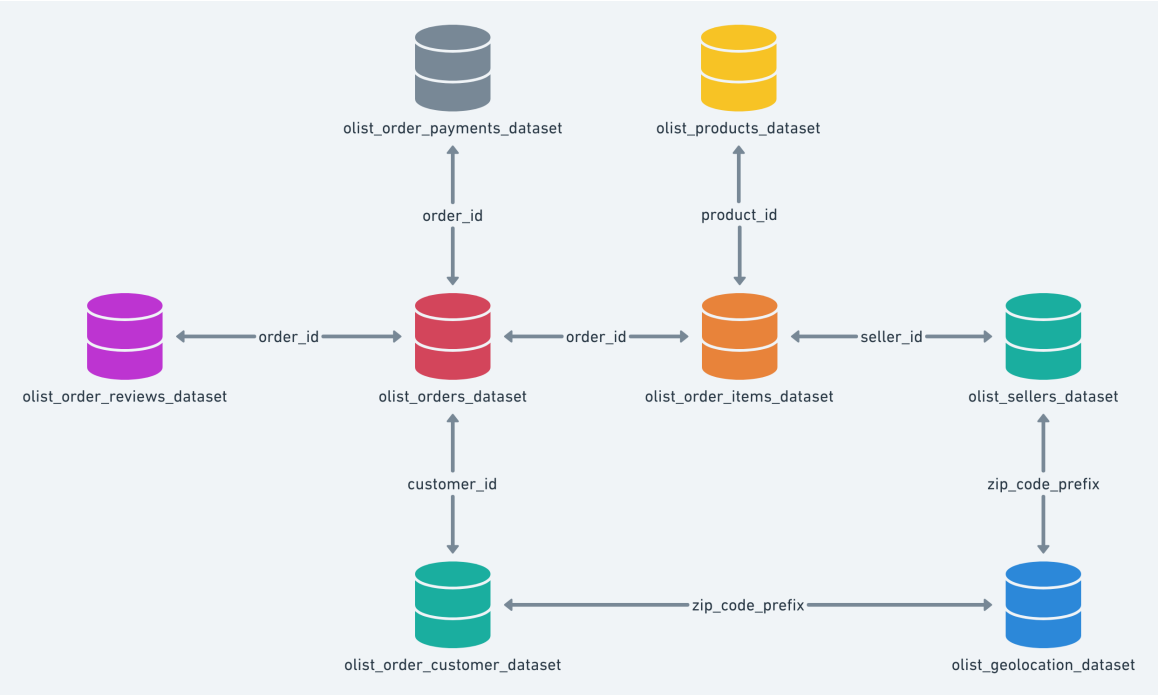
tabela, é possível analisar o ciclo de vida de um pedido, desde a realização da compra até a entrega.

- Produtos: Contém dados sobre os produtos vendidos, incluindo o nome, categoria, peso e dimensões. Esses dados são importantes para análises de catálogo, como a popularidade de diferentes categorias de produtos e a relação entre as características dos produtos e as vendas.
- Vendedores: Armazena informações sobre os vendedores que oferecem produtos na plataforma, incluindo o ID do vendedor e a localização. Essa tabela é útil para entender a distribuição dos vendedores e como isso afeta a logística e o desempenho de vendas.
- Clientes: Fornece informações detalhadas sobre os clientes, como o ID do cliente, sua localização e características demográficas. Esses dados são essenciais para segmentação de mercado, análise de comportamento de compra e para entender a distribuição dos clientes em diferentes regiões.

3.3 MANIPULAÇÃO DOS DADOS

As manipulações realizadas no trabalho, detalhadas na Seção 3.3.1, foram desenvolvidas por Téo, que desempenhou um papel essencial na organização e transformação dos dados. Ele implementou tabelas intermediárias e consultas SQL que permitiram extrair, combinar e agregar informações de várias fontes, resultando em uma tabela final robusta e abrangente, com características (features) relevantes para a análise de churn. Sua habilidade técnica garantiu que os dados fossem processados de maneira eficiente e organizada, estabelecendo uma base sólida para todas as etapas subsequentes do estudo.

Figura 6 – Base de dados, tabelas inter-relacionadas



Fonte: Retirada de (OLIST, 2018)

Embora o critério de churn e o período de análise tenham sido definidos por mim, Téo teve uma participação crucial na implementação das estratégias que possibilitaram sua aplicação. O critério de churn foi ajustado para 35 dias consecutivos sem vendas, em contraste com os 45 dias do repositório original, permitindo identificar inatividade de maneira mais rápida e eficaz. O período de análise, de 19 de novembro de 2016 a 19 de março de 2017, foi escolhido para incluir períodos de alta movimentação, como a Black Friday e as festas de fim de ano, além de momentos de menor atividade, oferecendo uma visão abrangente do comportamento dos vendedores.

A escolha dos algoritmos de aprendizado de máquina utilizados no estudo foi feita por Téo, que indicou modelos que equilibram eficiência e simplicidade. Com base nessa escolha, optei por manter os algoritmos propostos, dada sua capacidade de apresentar bons resultados com baixo custo computacional, alinhando-se aos objetivos do trabalho. Téo não apenas contribuiu com sua expertise técnica, mas também agregou valor estratégico ao projeto, garantindo que as soluções adotadas fossem eficazes e compatíveis com as metas estabelecidas.

3.3.1 Consultas Realizadas

3.3.1.1 *Relação vendedor avaliação*

A consulta *fs_vendedor_avaliação.sql* utilizou dados das tabelas de pedidos, avaliações e vendedores para calcular métricas associadas ao desempenho dos vendedores em termos de *feedback* recebido dos clientes. Foram consideradas a média das avaliações, a menor e maior nota registradas, bem como a proporção de pedidos que receberam avaliações. Esse conjunto de métricas permite uma análise aprofundada da qualidade do serviço prestado pelos vendedores, fornecendo uma visão clara sobre a satisfação do cliente.

3.3.1.2 *Relação Vendedor Cliente*

Na consulta *fs_vendedor_cliente.sql*, os dados das tabelas de clientes, pedidos e vendedores foram empregados para identificar padrões de distribuição geográfica dos consumidores. Essa análise abrangeu a contagem dos estados de origem dos pedidos e o cálculo da proporção de vendas realizadas em cada um dos 26 estados brasileiros. A partir dessas informações, foi possível compreender melhor a distribuição regional da demanda e a abrangência do mercado dos vendedores.

3.3.1.3 *Relação Vendedor Entrega*

A consulta *fs_vendedor_entrega.sql* focou na análise dos prazos e condições de entrega. Utilizando informações das tabelas de pedidos, produtos, vendedores e clientes, essa etapa avaliou o percentual de pedidos entregues fora do prazo estimado, a frequência de cancelamentos e os custos relacionados ao frete, identificando o valor médio, máximo e mínimo. Além disso, foram analisados o tempo médio entre a aprovação do pedido e a entrega, o tempo total desde a realização até a entrega e a discrepância média entre a data prevista e a entrega efetiva. Essa análise fornece *insights* fundamentais sobre a eficiência logística e a capacidade de atender às expectativas dos consumidores.

3.3.1.4 *Relação Vendedor Pagamentos*

Na consulta *fs_vendedor_pagamentos.sql*, foram analisadas informações das tabelas de pedidos, pagamentos, produtos e vendedores para compreender o comportamento dos consumidores em relação aos métodos de pagamento. Foram contabilizados os pedidos realizados por cartão de crédito, débito, boleto ou voucher, e o valor total gasto em cada modalidade. Além disso, foi calculada a proporção de cada método no total de pedidos e no valor das vendas, e, no caso de pagamentos com cartão de crédito, foi identificado o número de parcelas utilizadas. Essas métricas permitem uma análise aprofundada das preferências de pagamento e das estratégias de venda adotadas.

3.3.1.5 *Relação Vendedor Produto*

A consulta *fs_vendedor_produto.sql* combinou dados das tabelas de produtos, pedidos e vendedores para avaliar o portfólio de produtos oferecidos. As métricas extraídas incluíram o número médio de fotos associadas aos produtos, o volume médio calculado a partir das dimensões dos itens, e a distribuição percentual de produtos em diferentes categorias, como cama, mesa e banho, beleza, saúde, esporte e lazer. Esses indicadores fornecem uma visão detalhada sobre a diversificação e a competitividade do portfólio de cada vendedor.

3.3.1.6 *Relação Vendedor Vendas*

Por fim, a consulta *fs_vendedor_vendas.sql* analisou o desempenho comercial dos vendedores a partir das tabelas de pedidos e vendedores. Essa análise considerou o número de dias em que houve vendas, o total de itens vendidos, o tempo desde o último pedido (indicador de recência) e os valores médios por pedido e por produto. Também foram identificados os valores máximo e mínimo dos produtos vendidos, além da média de itens por pedido, oferecendo uma visão abrangente sobre a dinâmica de vendas e o desempenho dos vendedores ao longo do período avaliado.

3.3.2 Tabela Final

Após a aplicação das consultas nas tabelas mencionadas na subseção 3.3.1, foi obtida a tabela final que serviu como base para a análise de churn dos vendedores. Essa tabela foi criada a partir da integração de seis subconjuntos de dados principais, cada um contendo características específicas que representam diferentes aspectos do desempenho e comportamento dos vendedores na plataforma.

A tabela final reúne um total de 95 características, abrangendo métricas de vendas, avaliações, perfil dos clientes, dados de entrega, métodos de pagamento e detalhes dos produtos. Essas características foram criteriosamente selecionadas para capturar uma visão ampla e detalhada das interações dos vendedores com a plataforma, o que é essencial para uma predição eficaz de churn. As variáveis incluem informações como número de pedidos, intervalo entre vendas, notas de avaliação, distribuição geográfica dos pedidos, porcentagem de pedidos atrasados ou cancelados, preferências de métodos de pagamento dos clientes, e diversidade de produtos oferecidos.

Para definir o churn, este estudo considerou a ausência de novos pedidos por parte de um vendedor dentro de um período de 35 dias a partir da data de referência. Esse critério foi implementado por meio

de uma estrutura de junção que verifica a presença de pedidos dentro desse intervalo. Caso nenhum novo pedido seja detectado nesse período, o vendedor é marcado com um flag de churn (flChurn = 1), indicando que ele está inativo. Esse indicador permite uma detecção rápida de inatividade, fornecendo uma base sólida para ações preventivas de retenção e para a criação de modelos preditivos mais precisos e direcionados.

3.4 PREPARAÇÃO PARA OS ALGORITMOS

A análise foi cuidadosamente estruturada em três partes principais: treino, teste e fora do tempo (*out of time OOT*). A base fora do tempo foi definida como os dados a partir de 02 de Fevereiro de 2017. Este cojunto foi reservado exclusivamente para avaliar o desempenho do modelo em dados nunca vistos durante o processo de treinamento e validação. Essa estratégia é crucial para simular o comportamento do modelo em produção, garantindo que ele mantenha a capacidade de generalização necessária para prever corretamente situações futuras.

A base de treino foi composta por dados até 02 de Fevereiro de 2017, essa base foi dividida, utilizando-se 80% para o treinamento do modelo e 20% para a validação interna. Essa separação permitiu uma avaliação precisa do desempenho do modelo em dados não vistos, assegurando que ele fosse capaz de aprender padrões relevantes sem perder a capacidade de generalização.

Durante a preparação dos dados, identificou-se a presença de valores ausentes em várias variáveis importantes. Para lidar com esse desafio e garantir a consistência do conjunto de dados, foram aplicadas duas estratégias de imputação: uma substituindo os valores ausentes por 0 e outra substituindo-os por -100.

A estratégia de imputação com 0 foi utilizada para variáveis em que a ausência de valor pode ser interpretada como a inexistência de um evento ou característica específica. Por exemplo, em variáveis que indicam a quantidade de parcelas de pagamento ou o percentual de pedidos com atraso, a ausência de informação pode ser tratada como zero, sugerindo que o evento simplesmente não ocorreu. Essa abordagem preserva a coerência sem distorcer as distribuições dos dados, permitindo que o modelo compreenda a ausência de maneira neutra.

Por outro lado, a estratégia de imputação com -100 foi aplicada em variáveis onde a ausência de valor não pode ser interpretada como inexistência, mas sim como uma informação relevante por si só. Utilizar um número fora do intervalo típico da variável como -100 destaca a ausência de forma explícita, sinalizando ao modelo que essa falta de informação pode ser significativa. Essa estratégia permite ao modelo identificar padrões em situações em que dados críticos estão ausentes, oferecendo uma solução que evita excluir registros importantes durante o treinamento.

Essa preparação criteriosa dos dados, combinando estratégias de imputação específicas para cada contexto, garantiu que o modelo fosse capaz de lidar com dados imperfeitos de maneira eficaz. Além disso, a divisão dos dados em treino, teste e fora do tempo assegurou que o modelo fosse treinado e avaliado de forma justa, minimizando a possibilidade de viés e maximizando a capacidade de previsão.

A separação da base fora do tempo foi essencial para testar o desempenho do modelo em condições que simulam o uso real em produção. Dessa forma, foi possível verificar não apenas o desempenho do modelo nos dados internos, mas também sua capacidade de generalização. Esse processo foi essencial

para garantir que o modelo final pudesse identificar padrões de comportamento dos vendedores de maneira precisa, mesmo diante de mudanças no comportamento ao longo do tempo.

Por fim, a combinação das estratégias de imputação e divisão de dados ofereceu uma preparação robusta, aumentando a resiliência do modelo em situações adversas. Ao permitir que o modelo compreendesse tanto a ausência quanto a presença de informações em diferentes contextos, foi possível melhorar a qualidade das previsões e proporcionar uma análise mais precisa do churn.

3.5 IMPLEMENTAÇÃO DOS ALGORITMOS

A implementação dos algoritmos de árvore de decisão, floresta aleatória e *lightGBM* seguiu um processo padronizado que incluiu etapas de pré-processamento, ajuste de hiperparâmetros e avaliação de desempenho. O objetivo principal foi configurar e sintonizar cada modelo de forma a maximizar a métrica de AUC (Área Sob a Curva ROC), proporcionando uma análise comparativa robusta dos algoritmos em estudo.

Inicialmente, os dados foram tratados por meio de pipelines, que aplicavam estratégias específicas para lidar com valores ausentes, assegurando que os modelos fossem treinados sem problemas decorrentes de dados faltantes. Em seguida, cada algoritmo foi incorporado a um pipeline de classificação, facilitando a aplicação uniforme das transformações e dos parâmetros ao longo do processo de ajuste.

A seleção e configuração dos hiperparâmetros ajustados para cada algoritmo foram baseadas nas características que mais impactam o desempenho e a capacidade de generalização de modelos de aprendizado supervisionado. No caso da árvore de decisão e da floresta aleatória, foram ajustados hiperparâmetros como profundidade máxima, número mínimo de amostras para dividir um nó e número mínimo de amostras por folha, buscando encontrar um equilíbrio entre complexidade e generalização. Esses parâmetros.

Para o modelo de floresta aleatória, foi também ajustado o número de árvores, um hiperparâmetro essencial para aumentar a robustez do modelo, uma vez que um número adequado de árvores permite maior consenso entre previsões, reduzindo variações e melhorando a estabilidade. Na floresta aleatória e no *LightGBM*, a proporção de recursos considerados em cada divisão foi configurada para minimizar correlações entre árvores, favorecendo a capacidade preditiva do modelo.

No caso do *LightGBM*, o foco esteve em parâmetros que afetam a estrutura das árvores e a velocidade de aprendizado. Parâmetros como *learning rate* foram ajustados para controlar a taxa de aprendizado, promovendo uma abordagem que prioriza ajustes graduais, a fim de evitar sobreajustes. Da mesma forma, o número de folhas nas árvores foi configurado para balancear a complexidade do modelo com a necessidade de manter uma estrutura eficiente e de alta precisão.

O ajuste dos hiperparâmetros foi realizado com *RandomizedSearchCV*, que seleciona aleatoriamente combinações de valores pré-definidos e otimiza a métrica de AUC. Esse método permitiu explorar o espaço de hiperparâmetros (Tabela 1) de maneira eficiente, evitando o tempo elevado que seria necessário em uma busca exaustiva, e identificando combinações promissoras.

Foram realizadas várias iterações, utilizando validação cruzada para verificar o desempenho em diferentes subconjuntos dos dados de treino, e os resultados obtidos foram salvos para possibilitar uma análise comparativa posterior.

Tabela 1 – Intervalo de valores dos hiperparâmetros utilizados

hiperparâmetro	Intervalo	Algoritmo
n_estimators	100 a 100 passo de 100	Floresta Aleatória
min_samples_split	2 a 100	Árvore de decisão e Floresta Aleatória
max_depth	None, 1 a 20 com um passo de 1 e de 30 a 110 com um passo de 10	Árvore de decisão, Floresta Aleatória e *LightGBM*
min_samples_leaf	10 a 100	Árvore de decisão e Floresta Aleatória
max_features	sqrt, log2	Árvore de decisão e Floresta Aleatória
class_weight	None, balanced	Árvore de decisão e Floresta Aleatória
criterion	gini e entropy	Floresta Aleatória
num_leaves	30 a 90 com passo de 10	*LightGBM*
n_estimators	Valores aleatórios de 100 a 1000	*LightGBM*
learning_rate	valores aleatórios de 0.001 a 1	*LightGBM*
colsample_bytree	0.3, 0.475, 0.65, 0.825, 1	*LightGBM*

Fonte: Elaborado pelo autor

Após a sintonização, foram selecionados os melhores modelos, e suas performances foram reavaliadas com base na Área Sob a Curva. Essa avaliação foi conduzida em três conjuntos de dados distintos: treino, teste e um conjunto de dados fora do tempo (*out of time*), com o objetivo de garantir a capacidade de generalização dos modelos, além de identificar potenciais problemas de ajuste excessivo ou insuficiente.

Para auxiliar na interpretação visual das taxas de acerto, foram gerados gráficos das curvas ROC para cada algoritmo, comparando os desempenhos nos diferentes conjuntos de dados. Esses gráficos permitiram avaliar visualmente a sensibilidade e a especificidade dos modelos, tornando mais clara a interpretação das taxas de acerto.

3.6 MÉTRICA DE AVALIAÇÃO

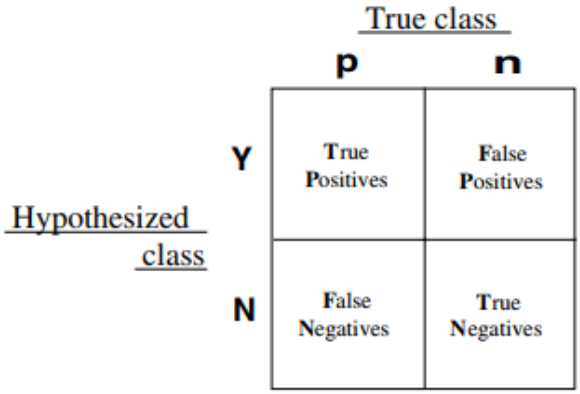
Para compreender a métrica escolhida para avaliar o modelo, é essencial primeiro entender o que é a Curva ROC. No entanto, para compreender a Curva ROC, é necessário entender as métricas derivadas da matriz de confusão, que formam a base para essa curva.

3.6.1 Matriz de Confusão e Métricas Derivadas

A matriz de confusão é uma ferramenta fundamental para a avaliação de modelos de classificação, oferecendo uma visão detalhada dos acertos e erros cometidos pelo modelo. Ela é composta por quatro elementos principais: verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos (Figura 7). A partir desses elementos, podem-se calcular métricas importantes como precisão, acurácia, sensibilidade (taxa de verdadeiros positivos) e especificidade (taxa de verdadeiros negativos) (FAWCETT, 2006).

- **Sensibilidade:** representa a capacidade do modelo de identificar corretamente os casos positivos (churn), sendo crucial em cenários onde a classe positiva é de interesse particular, como na retenção de clientes. A equação que representa a sensibilidade está descrita em (1)

Figura 7 – Matriz de confusão



Fonte: Retirada de (FAWCETT, 2006)

$$\text{Sensibilidade} = \frac{TP}{P} \tag{1}$$

- **Especificidade:** avalia a capacidade do modelo de identificar corretamente os casos negativos, evitando falsos alarmes. A equação que representa a especificidade está descrita em (2)

$$\text{Especifidade} = \frac{TN}{N} \tag{2}$$

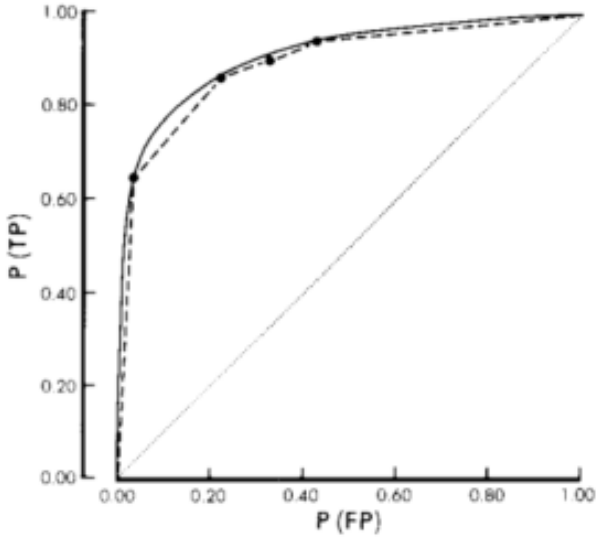
3.6.2 Curva ROC

A Curva ROC (*Receiver Operating Characteristic*) é um gráfico que demonstra a relação entre a sensibilidade (taxa de verdadeiros positivos) e 1 - especificidade (taxa de falsos positivos) para diferentes limiares de decisão. Esse gráfico é especialmente útil em problemas de classificação binária e ajuda a visualizar o desempenho do modelo na distinção entre as classes positiva e negativa (HANLEY; MCNEIL, 1982). Quanto mais próxima a Curva ROC estiver do canto superior esquerdo (Figura 8), melhor o desempenho do modelo, pois indica uma alta taxa de verdadeiros positivos e uma baixa taxa de falsos positivos (FAWCETT, 2006)

3.6.3 Área Sob a Curva

A métrica da Área Sob a Curva ROC (AUC - Area Under the Curve) foi escolhida para avaliar o desempenho do modelo de predição de churn neste estudo, devido à natureza desbalanceada dos dados. Em problemas de classificação desbalanceada, uma das classes (neste caso, a classe de churn) é menos frequente em comparação com a outra. Em uma situação de desbalanceamento, métricas como a acurácia podem ser enganosas, pois o modelo poderia obter uma alta acurácia simplesmente classificando a maioria dos casos na classe majoritária, ignorando a classe minoritária. A AUC, ao contrário, é mais robusta, pois mede a capacidade do modelo de distinguir entre as classes positiva (churn) e negativa (não churn) independentemente de um limiar específico, o que proporciona uma visão mais confiável do desempenho do modelo (BRADLEY, 1997).

Figura 8 – Curva ROC



Fonte: Retirada de (HANLEY; MCNEIL, 1982)

A Área Sob a Curva ROC (AUC - Area Under the Curve) quantifica a Curva ROC em um único valor, variando de 0 a 1, que representa a probabilidade do modelo classificar corretamente uma observação positiva em comparação com uma negativa. Uma AUC de 0,5 indica um desempenho equivalente ao acaso, enquanto valores próximos a 1 indicam um alto poder discriminativo do modelo (BRADLEY, 1997). Segundo Hanley Mcneil (1982), a AUC é uma métrica robusta para avaliar modelos de classificação, pois considera o desempenho do modelo em todos os possíveis limiares de decisão, fornecendo uma visão geral da capacidade de separação do modelo (HANLEY; MCNEIL, 1982).

4 RESULTADOS E DISCUSSÕES

Para assegurar uma análise robusta dos modelos, primeiramente foi realizada uma expansão da base de dados. A análise inicial foi conduzida com uma quantidade menor de dados, com o objetivo de avaliar a performance dos algoritmos em um cenário com dados limitados. Em seguida, ampliou-se o conjunto de dados, abrangendo um período maior, de 2016/11/19 a 2017/09/19, totalizando aproximadamente 34 milhões de registros. Esse aumento permitiu uma comparação direta entre o desempenho dos algoritmos com a base inicial e com a base expandida, proporcionando uma visão mais completa sobre a capacidade de generalização dos modelos.

A Tabela 2 mostra a distribuição das classes nas duas bases, tanto na base inicial que foi de Novembro de 2016 até Março de 2017 quanto a base expandida que foi de Novembro de 2016 até Setembro de 2017, evidenciando o desbalanceamento entre as classes, o que justifica o uso da área sob a curva como principal métrica de avaliação.

Tabela 2 – Classes desbalanceadas

Base inicial			**Base Expandida**		
Classe	**Quantidade**	**Percentual**	**Classe**	**Quantidade**	**Percentual**
0 (não churn)	13.499	68,87 %	0 (não churn)	305.654	64,90 %
1 (churn)	6.102	31,13 %	1 (churn)	165.372	35,10 %

Fonte: Elaborado pelo autor

4.1 ÁRVORE DE DECISÃO

Os resultados obtidos para o algoritmo de Árvore de Decisão,estão apresentados na Tabela 3. Nessa tabela, foram exibidos o desempenho nas duas bases de dados analisadas: a inicial e a expandida, com os três melhores modelos selecionados para cada base.

Tabela 3 – Área sob a curva para o algoritmo árvore de decisão

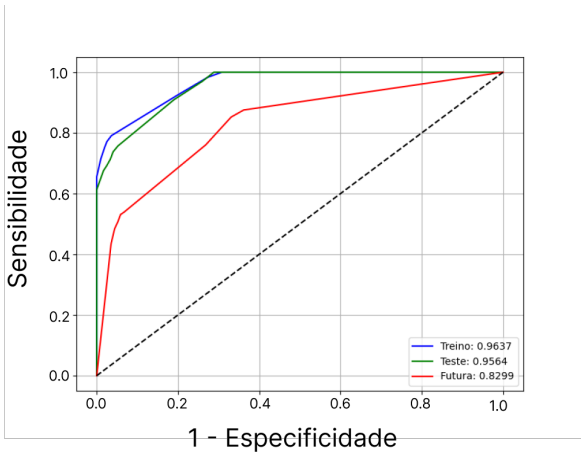
Base inicial			**Base Expandida**		
Treino	**Teste**	**OOT**	**Treino**	**Teste**	**OOT**
0,9537	0,9564	0,8299	0,9527	0,9523	0,7363
0,9430	0,9420	0,8216	0,9115	0,9107	0,7172
0,9358	0,9299	0,8184	0,8743	0,8741	0,6994

Fonte: Elaborado pelo autor

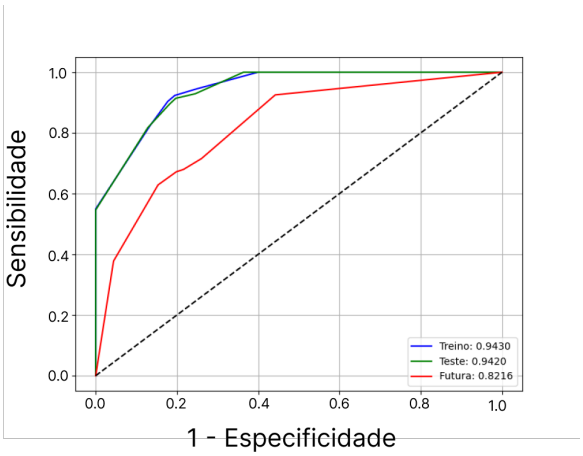
Os valores de AUC foram coletados para os conjuntos de treino, teste e fora do tempo (OOT), proporcionando uma visão ampla do desempenho do modelo em diferentes amostras e permitindo uma comparação entre as bases de dados. Observa-se que, para a base expandida, o desempenho AUC do modelo no conjunto fora do tempo teve uma ligeira queda em comparação com a base inicial, o que pode ser atribuído à simplicidade do modelo de árvore de decisão, que tem uma capacidade limitada para capturar relações complexas em bases de dados maiores.

Figura 9 – Curva ROC e AUC do algoritmo árvore de decisão para base inicial. (a) Melhor resultado. (b) 2º melhor resultado. (c) 3º melhor resultado.

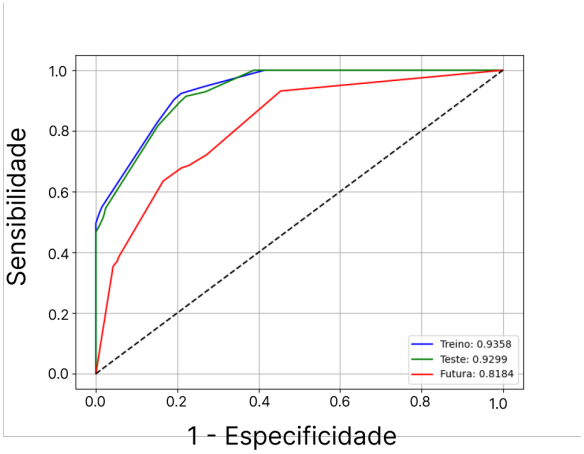
(a)



(b)



(c)



Fonte: Elaborado pelo autor

A Figura 9 apresenta a curva ROC e a métrica de AUC para os resultados obtidos utilizando a base de dados inicial, enquanto a Figura 10 exibe a curva ROC e a AUC para a base de dados expandida. Esse comparativo revela que o modelo de árvore de decisão manteve um desempenho satisfatório, mas não conseguiu aproveitar completamente o aumento de dados para melhorar a capacidade de generalização, sugerindo que algoritmos mais simples podem ter limitações em cenários com grandes volumes de dados.

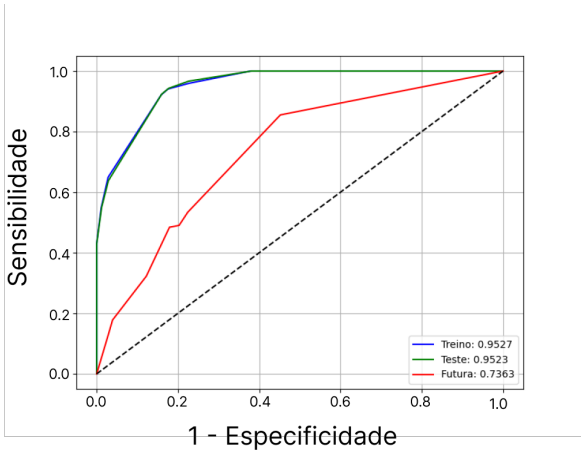
Em termos de hiperparâmetros, documentados na Tabela 4, foram utilizadas condições específicas para cada conjunto de dados. Embora esses valores tenham resultado em um bom desempenho, não foi possível identificar uma influência isolada de cada hiperparâmetro, dada a complexidade das interações entre eles. A tabela serve, portanto, como um registro informativo dos parâmetros que contribuíram para os melhores desempenhos observados.

4.2 FLORESTA ALEATÓRIA

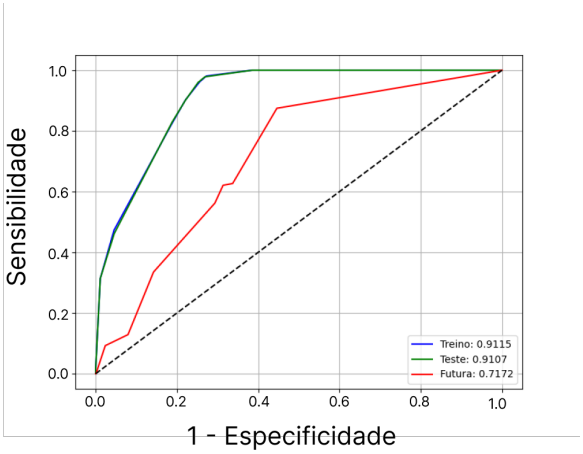
Para o modelo de Floresta Aleatória, os resultados estão descritos na Tabela 5, que exibe o desempenho tanto na base inicial quanto na expandida, com os três melhores modelos para cada

Figura 10 – Curva ROC e AUC do algoritmo árvore de decisão base expandida. (a) Melhor resultado. (b) 2º melhor resultado. (c) 3º melhor resultado.

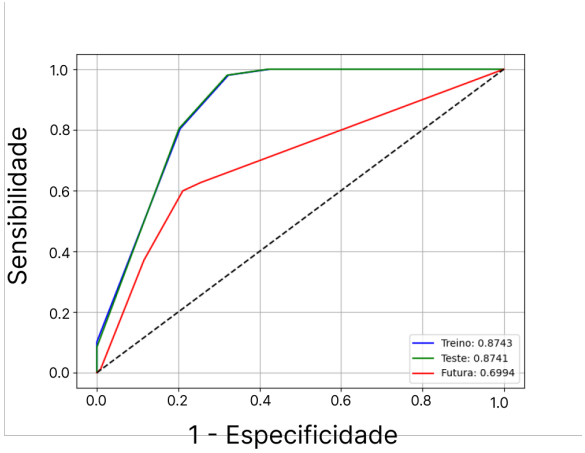
(a)



(b)



(c)



Fonte: Elaborado pelo autor

Tabela 4 – Hiperparâmetros utilizados para o algoritmo árvore de decisão

	Base Inicial			Base Expandida		
Hiperparâmetros	**Melhor**	**2º Melhor**	**3º Melhor**	**Melhor**	**2º Melhor**	**3º melhor**
splitter	best	best	best	best	best	random
min_samples_split	9	97	72	72	31	75
min_samples_leaf	20	29	44	44	66	23
max_features	log2	null	null	null	null	null
max_depth	7	4	4	4	4	6
class_weigth	null	null	null	null	balanced	balanced

Fonte: Elaborado pelo autor

conjunto. A Floresta Aleatória apresentou uma alta AUC em ambos os cenários, mantendo-se estável entre as bases de dados inicial e expandida. Isso indica que o modelo conseguiu generalizar bem em ambos os conjuntos de dados, beneficiando-se parcialmente do aumento de dados sem perda significativa de performance no conjunto OOT.

As curvas ROC para a base inicial (Figura 11) e para a base expandida (Figura 12) mostram que o

Tabela 5 – Área sob a curva para o algoritmo floresta aleatória

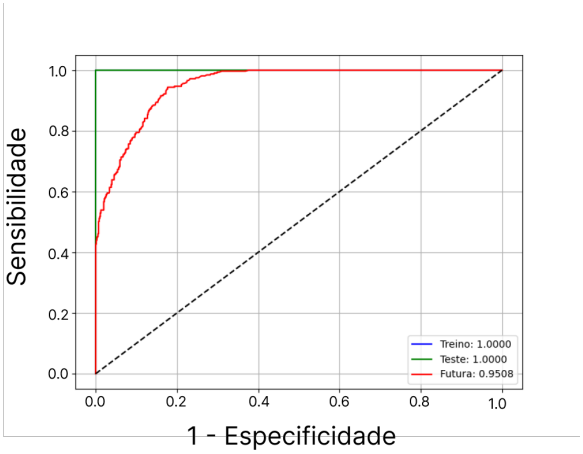
Base inicial			Base Expandida		
Treino	Teste	OOT	Treino	Teste	OOT
1,00	1,00	0,9508	1,00	1,00	0,8250
1,00	1,00	0,9507	1,00	1,00	0,8241
1,00	1,00	0,9506	1,00	1,00	0,8238

Fonte: Elaborado pelo autor

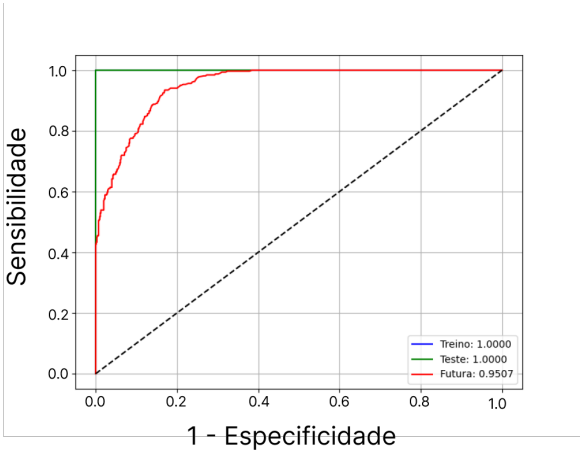
modelo manteve uma 1-especificidade e sensibilidade elevadas, independentemente da quantidade de dados.

Figura 11 – Curva ROC e AUC do algoritmo floresta aleatória para a base inicial.(a) Melhor resultado. (b) 2º melhor resultado. (c) 3º melhor resultado.

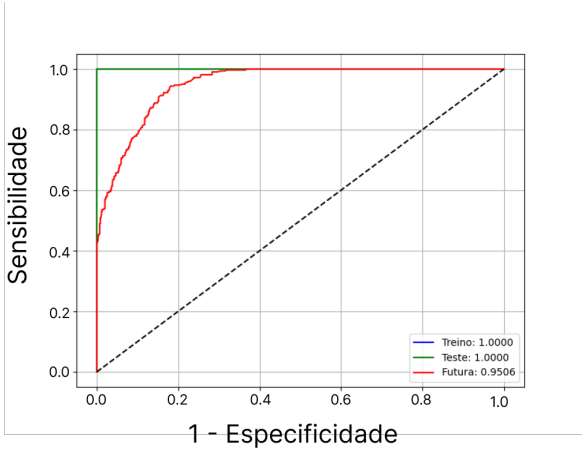
(a)



(b)



(c)

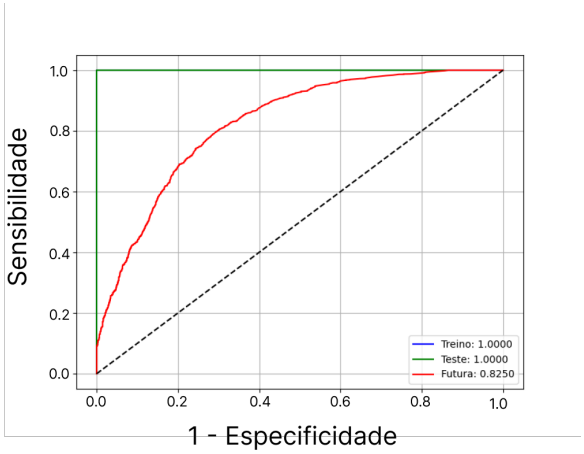


Fonte: Elaborado pelo autor

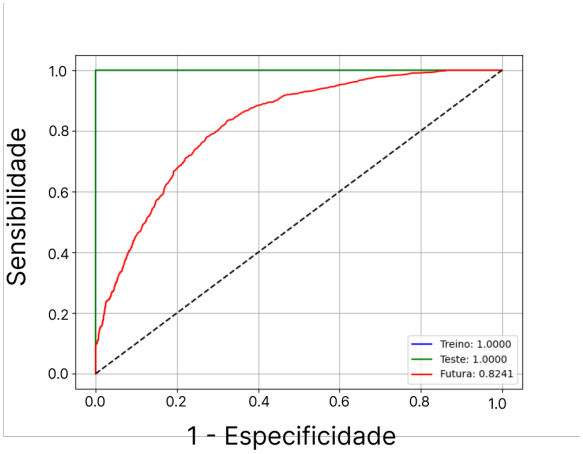
Os hiperparâmetros utilizados, mostrados na Tabela 6, incluem o número de estimadores e a profundidade máxima, que foram ajustados para cada conjunto de dados. A escolha desses parâmetros permitiu um desempenho consistente, mas, assim como com a árvore de decisão, as interações entre os parâmetros tornam difícil atribuir um impacto isolado a cada um deles. Dessa forma, a tabela é utilizada para registrar os valores que resultaram nos melhores desempenhos, sem conclusões isoladas sobre o impacto de cada hiperparâmetro.

Figura 12 – Curva ROC e AUC do algoritmo floresta aleatória para a base expandida. (a) Melhor resultado. (b) 2º melhor resultado. (c) 3º melhor resultado.

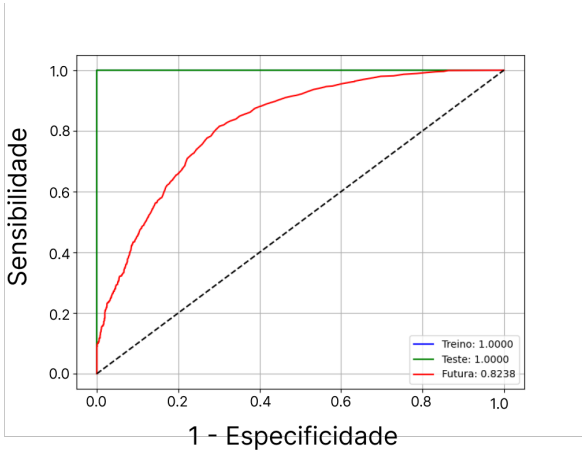
(a)



(b)



(c)



Fonte: Elaborado pelo autor

Tabela 6 – Hiperparâmetros utilizados para o algoritmo floresta aleatória

	Base inicial			**Base Expandida**		
Hiperparâmetros	**Melhor**	**2º Melhor**	**3º Melhor**	**Melhor**	**2º Melhor**	**3º melhor**
n_estimators	700	500	800	500	700	300
min_samples_split	31	33	2	43	79	46
min_samples_leaf	15	10	11	26	73	25
max_features	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt
max_depth	50	70	90	16	19	17
criterion	entropy	entropy	entropy	gini	gini	gini
class_weigth	null	null	null	null	null	null

Fonte: Elaborado pelo autor

4.3 LIGHTGBM

O algoritmo *LightGBM* apresentou valores de AUC para os três melhores modelos, avaliados nas bases inicial e expandida, conforme mostrado na Tabela 7. Esse modelo destacou-se com o melhor desempenho entre os três algoritmos, especialmente na base expandida, onde apresentou um AUC OOT mais elevado em comparação com as outras abordagens. O *LightGBM* parece ter se beneficiado

mais do aumento de dados, revelando-se altamente eficaz para modelos com muitos dados.

Tabela 7 – Área sob a curva para o algoritmo *LightGBM*

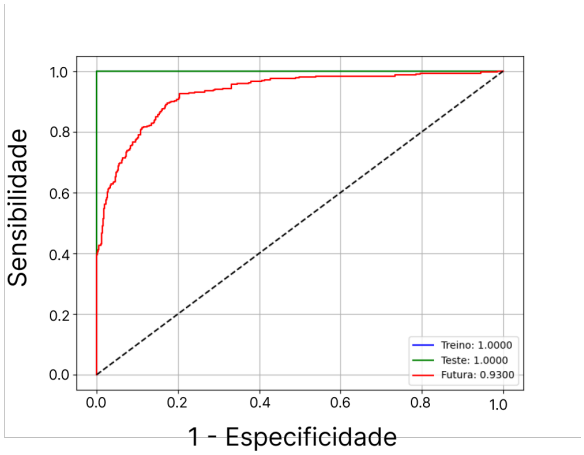
Base inicial			Base Expandida		
Treino	Teste	OOT	Treino	Teste	OOT
1,00	1,00	0,9300	1,00	1,00	0,9851
1,00	1,00	0,9270	1,00	0,9999	0,9788
1,00	1,00	0,9192	0,9997	0,9997	0,9573

Fonte: Elaborado pelo autor

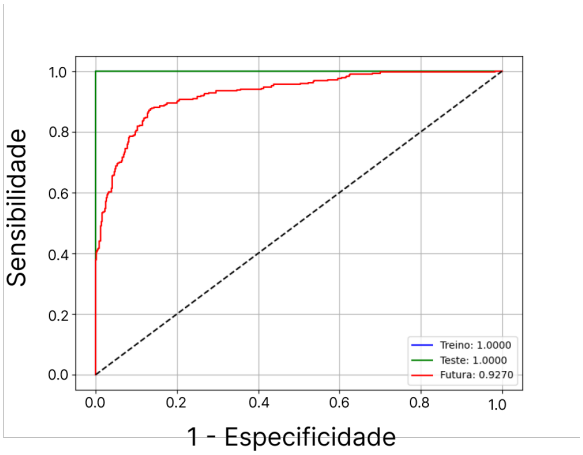
A análise das curvas ROC para a base inicial (Figura 13) e para a base expandida (Figura 14) confirma que o *LightGBM* conseguiu manter uma alta capacidade de discriminação entre classes, ajustando-se melhor ao maior volume de dados e superando as outras abordagens em termos de AUC. Esse desempenho pode ser atribuído à forma como o *LightGBM* aprende com o erro acumulado, sendo mais adequado para cenários com uma maior complexidade e quantidade de dados.

Figura 13 – Curva ROC e AUC para o algoritmo *LightGBM* base de dados inicial. (a) Melhor resultado. (b) 2º melhor resultado. (c) 3º melhor resultado.

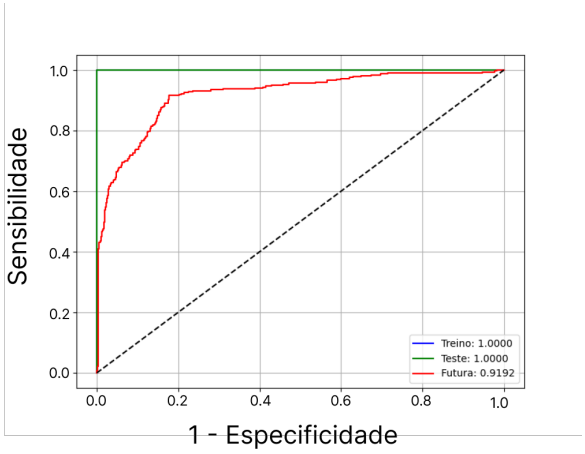
(a)



(b)



(c)

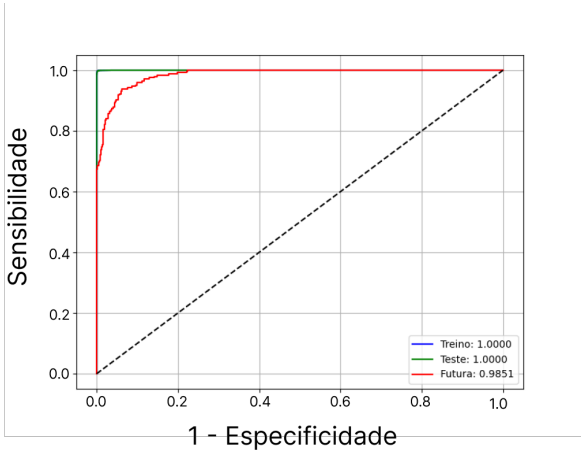


Fonte: Elaborado pelo autor

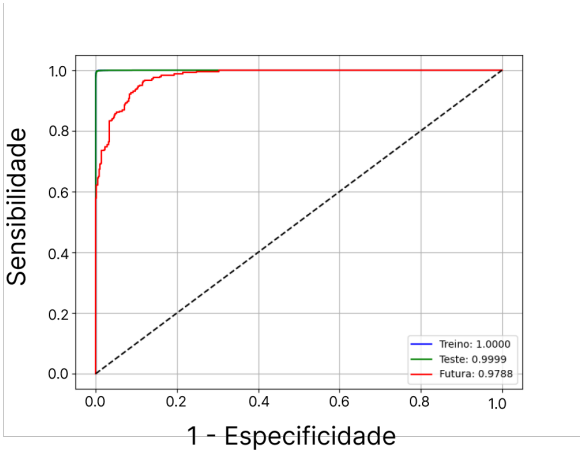
Os hiperparâmetros utilizados para o *LightGBM*, documentados na Tabela 8, foram ajustados conforme os critérios de desempenho. Parâmetros como número de folhas e taxa de aprendizado foram

Figura 14 – Curva ROC e AUC para o algoritmo *LightGBM* base de dados expandida.(a) Melhor resultado. (b) 2º melhor resultado. (c) 3º melhor resultado.

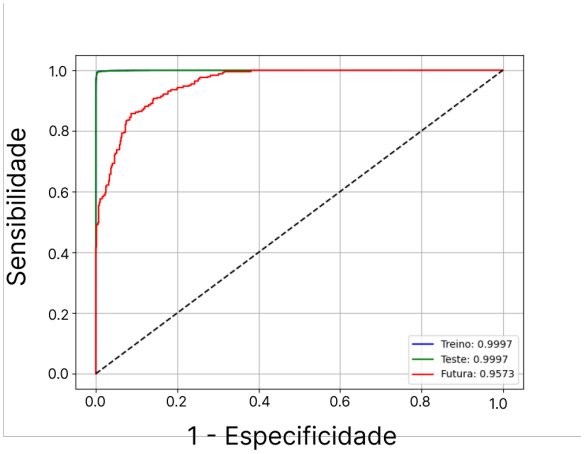
(a)



(b)



(c)



Fonte: Elaborado pelo autor

escolhidos para maximizar a AUC, especialmente na base expandida. No entanto, como nos outros algoritmos, a complexidade das interações entre os parâmetros impede uma análise conclusiva sobre a importância isolada de cada hiperparâmetro. A tabela, portanto, serve para documentar as condições específicas que resultaram nos melhores desempenhos para o *LightGBM*.

Tabela 8 – Hiperparâmetros utilizados para o algoritmo *LightGBM*

	Base inicial			**Base Expandida**		
Hiperparâmetros	**Melhor**	**2º Melhor**	**3º Melhor**	**Melhor**	**2º Melhor**	**3º melhor**
num_leaves	50	50	80	70	60	70
n_estimators	318	181	154	481	190	118
min_child_samples	30	90	70	60	80	100
max_depth	13	9	8	11	17	14
learning_rate	0.089	0.045	0,1	0.023	0.056	0.067
colsample_bytree	0,475	0,475	1	0,475	0.825	0.475

Fonte: Elaborado pelo autor

4.3.1 Comparação dos Algoritmos

Ao comparar os resultados numéricos dos algoritmos de Árvore de Decisão, Floresta Aleatória e *LightGBM*, é possível observar como cada modelo respondeu de forma distinta ao aumento de volume de dados e ao desbalanceamento das classes, com AUCs variando em níveis que destacam tanto a precisão quanto a capacidade de generalização dos modelos.

O *LightGBM* mostrou-se o mais beneficiado com o aumento de dados, com AUCs muito próximas de 1,0 na base expandida, especialmente no conjunto fora do tempo (OOT). Esse valor de AUC indica uma excelente capacidade do modelo para distinguir entre as classes positivas e negativas. Por exemplo, uma AUC de 1,0, como observada em alguns conjuntos de treino e teste, representa um desempenho perfeito, onde o modelo classifica corretamente todos os exemplos. Embora o AUC de 1,0 em treino e teste possa indicar um possível sobreajuste, o desempenho consistente na base expandida sugere que o modelo consegue generalizar bem para novos dados. Esse resultado reflete a estrutura de boosting do *LightGBM*, que se ajusta aos erros de previsões anteriores para aumentar a precisão.

Com AUCs consistentes em torno de 0,95 ou mais nos conjuntos de treino e teste, a floresta aleatória também mostrou um desempenho sólido tanto na base inicial quanto na expandida. A AUC de 0,95 indica uma capacidade alta do modelo de discriminar entre as classes, sendo ideal em cenários onde se busca um equilíbrio entre precisão e generalização. No conjunto OOT da base expandida, o AUC caiu ligeiramente para aproximadamente 0,82, o que representa uma boa performance, mas com menos confiança na discriminação entre classes em novos dados. Esses valores sugerem que a Floresta Aleatória consegue capturar bem a estrutura dos dados, mas talvez não se beneficie tanto do aumento de dados quanto o *LightGBM*.

Os resultados para o modelo de Árvore de Decisão apresentam AUCs na faixa de $0,93$ a $0,95$ nos conjuntos de treino e teste da base inicial, mas caem para a faixa de $0,87$ na base expandida, particularmente no conjunto OOT, onde a AUC chega a $0,69$. Um AUC de $0,69$ indica que o modelo ainda discrimina melhor que o acaso $(0,5)$, mas com uma capacidade limitada de separar corretamente as classes. Essa diminuição na base expandida aponta que, diferentemente dos outros algoritmos, a árvore de decisão pode ter dificuldades em se ajustar a grandes volumes de dados, possivelmente por sua simplicidade, que impede de capturar relações complexas quando há um aumento expressivo de dados.

5 CONCLUSÕES

Este trabalho apresentou uma análise comparativa de três modelos de aprendizado de máquina: árvore de decisão, floresta aleatória e *LightGBM*, aplicados na predição de churn em um *e-commerce*. Os resultados indicam que todos os algoritmos podem ser eficazes nessa tarefa, mas cada um possui características que o tornam mais ou menos adequado para diferentes volumes de dados. O *LightGBM* mostrou-se particularmente eficiente na base expandida, sugerindo que é o modelo mais adequado para contextos com grandes quantidades de dados disponíveis. A floresta aleatória manteve um desempenho sólido e consistente, sendo uma escolha robusta independentemente do tamanho da base. A árvore de decisão, por sua vez, apresentou limitações na base expandida, refletindo sua simplicidade estrutural.

O uso da métrica AUC foi validado pela natureza desbalanceada dos dados, permitindo uma avaliação mais realista da capacidade dos modelos de distinguir entre as classes. A análise dos hiperparâmetros, embora informativa, destacou a complexidade das interações entre eles, e o trabalho sugere como extensão futura a análise isolada dos efeitos de cada hiperparâmetro, com o intuito de aprimorar o entendimento dos processos de ajuste dos modelos.

Este estudo contribui para o entendimento de como diferentes algoritmos podem ser aplicados à predição de churn em e-commerce, fornecendo insights sobre a importância do ajuste de hiperparâmetros e do aumento de dados para aprimorar a capacidade preditiva. Como próximas etapas, sugere-se explorar o impacto isolado dos hiperparâmetros, a inclusão de novas variáveis para enriquecer o modelo e a aplicação de técnicas de aprendizado profundo, visando obter ainda mais precisão e insights sobre o comportamento de clientes. Assim, o trabalho abre caminho para futuras pesquisas e melhorias práticas em modelos de predição de churn, contribuindo para estratégias de retenção mais eficazes.

REFERÊNCIAS

ALPAYDIN, E. **Introduction to machine learning**. [S.l.]: MIT press, 2020.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. **Journal of Machine Learning Research**, v. 13, n. 2, 2012.

BRADLEY, A. P. The use of the area under the ROC curve in the evaluation of machine learning algorithms. **Pattern Recognition**, Elsevier, v. 30, n. 7, 1997.

BREIMAN, L. Random forests. **Machine learning**, Springer, v. 45, 2001.

BUCKINX, W.; POEL, D. Van den. Customer base analysis: partial defection of behaviourally loyal clients in a non-contractual fmcg retail setting. **European Journal of Operational Research**, Elsevier, v. 164, n. 1, 2005.

DESENVOLVIMENTO INDÚSTRIA, C. e. S. M. Ministério do. **E-commerce no Brasil cresce 4,8% e alcança R$ 196 bi em 2023**. 2024. <https://www.gov.br/mdic/pt-br/assuntos/noticias/2024/setembro/e-commerce-no-brasil-cresce-4-e-alcanca-r-196-bi-em-2023>. Acesso em: 20 out. 2024.

FAWCETT, T. An introduction to roc analysis. **Pattern Recognition Letters**, Elsevier, v. 27, n. 8, 2006.

FERREIRA, J. B. Mineração de dados na retenção de clientes em telefonia celular. **Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro–PUC**, 2005.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. **Annals of Statistics**, JSTOR, 2001.

GARETH, J. et al. **An introduction to statistical learning: with applications in R**. [S.l.]: Spinger, 2013.

GLADY, N.; BAESENS, B.; CROUX, C. Modeling churn using customer lifetime value. **European Journal of Operational Research**, Elsevier, v. 197, n. 1, p. 402–411, 2009.

GOODFELLOW, I. **Deep learning**. [S.l.]: MIT press, 2016.

HADDEN, J. et al. Computer assisted customer churn management: State-of-the-art and future trends. **Computers & Operations Research**, Elsevier, v. 34, n. 10, 2007.

HANLEY, J.; MCNEIL, B. The meaning and use of the area under a receiver operating characteristic (roc) curve. **Radiology**, v. 143, 05 1982.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2. ed. [S.l.]: Springer, 2009.

HUTTER, F.; KOTTHOFF, L.; VANSCHOREN, J. **Automated machine learning: methods, systems, challenges**. [S.l.]: Springer Nature, 2019.

JENI, L. A.; COHN, J. F.; TORRE, F. D. L. Facing imbalanced data–recommendations for the use of performance metrics. In: IEEE. **2013 Humaine association conference on affective computing and intelligent interaction**. [S.l.], 2013.

KE, G. et al. Lightgbm: A highly efficient gradient boosting decision tree. **Advances In Neural Information Processing Systems**, v. 30, 2017.

KELLEHER, J. D.; NAMEE, B. M.; D'ARCY, A. **Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies**. [S.l.]: MIT press, 2020.

LEMMENS, A.; CROUX, C. Bagging and boosting classification trees to predict churn. **Journal of Marketing Research**, SAGE Publications Sage CA: Los Angeles, CA, v. 43, n. 2, p. 276–286, 2006.

MANTOVANI, R. G. et al. Better trees: an empirical study on hyperparameter tuning of classification decision tree induction algorithms. **Data Mining and Knowledge Discovery**, Springer, p. 1–53, 2024.

Microsoft Corporation. **LightGBM Documentation**. 2021. Acesso em: 13/08/2024. Disponível em: <https://lightgbm.readthedocs.io/en/latest/>.

MOENARDY, K. K.; ARIFIN, S.; KUMADJI, S. The effect of service quality and relationship marketing to customer value, customer satisfaction, switching cost, and customer retention: A case study on the customers of bank ntt at east nusa tenggara province. **International Journal of Management and Administrative Sciences**, v. 3, n. 4, 2016.

MÜLLER, A. C.; GUIDO, S. **Introduction to machine learning with Python: a guide for data scientists**. [S.l.]: "O'Reilly Media, Inc.", 2016.

MURPHY, K. P. **Machine learning: a probabilistic perspective**. [S.l.]: MIT press, 2012.

OLIST. **Brazilian E-Commerce Public Dataset by Olist**. 2018. Disponível no Kaggle. Disponível em: <https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce>.

PARK, S.; KIM, J. Landslide susceptibility mapping based on random forest and boosted regression tree models, and a comparison of their performance. **Applied Sciences**, MDPI, v. 9, n. 5, 2019.

PROBST, P.; BOULESTEIX, A.-L.; BISCHL, B. Tunability: Importance of hyperparameters of machine learning algorithms. **Journal of Machine Learning Research**, v. 20, n. 53, p. 1–4, 2019.

SCIKIT-LEARN: Decision Trees. <https://scikit-learn.org/stable/modules/tree.html>. Accessed: 2024-08-12.

STROUSE, K. G. Marketing telecommunications services: new approaches for a changing environment. **(No Title)**, 1999.

TELOKEN, A. et al. Estudo comparativo entre os algoritmos de mineração de dados random forest e j48 na tomada de decisão. **Simpósio de Pesquisa e Desenvolvimento em Computação**, v. 2, n. 1, 2016.

TSAI, C.-F.; LU, Y.-H. Customer churn prediction by hybrid neural networks. **Expert Systems with Applications**, Elsevier, v. 36, n. 10, 2009.

VERHOEF, P. C.; LEMON, K. N. Successful customer value management: Key lessons and emerging trends. **European Management Journal**, Elsevier, v. 31, n. 1, p. 1–15, 2013.

YANG, L.; SHAMI, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. **Neurocomputing**, Elsevier, v. 415, p. 295–316, 2020.