PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PREDIÇÃO DO RISCO DE CRÉDITO EM INSTITUIÇÕES FINANCEIRAS UTILIZANDO *MACHINE LEARNING*

GUILHERME FREIRE MAGALHÃES

GOIÂNIA
2024

GUILHERME FREIRE MAGALHÃES

PREDIÇÃO DO RISCO DE CRÉDITO EM INSTITUIÇÕES FINANCEIRAS UTILIZANDO *MACHINE LEARNING*

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Orientador:

Prof. Me. Aníbal Santos Jukemura

Banca examinadora:

Prof. Mª. Lucília Gomes Ribeiro

Prof. Esp. Vitor Hugo Menezes Machado

GOIÂNIA
2024

GUILHERME FREIRE MAGALHÃES

PREDIÇÃO DO RISCO DE CRÉDITO EM INSTITUIÇÕES FINANCEIRAS UTILIZANDO *MACHINE LEARNING*

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Ciência da Computação, em _____/_____/_____.

_____

Orientador: Prof. Me. Aníbal Santos Jukemura

_____

Prof. Mª. Lucília Gomes Ribeiro

_____

Prof. Esp. Vitor Hugo Menezes Machado

GOIÂNIA
2024

AGRADECIMENTOS

Agradeço primeiramente a Deus por me conceder forças para vencer os desafios enfrentados durante essa jornada acadêmica.

Agradeço à minha família, especialmente aos meus pais, pelo apoio e incentivo constantes durante a minha formação.

Agradeço aos meus colegas de turma, Daniel Xavier e Lucas Augusto, pela companhia e pelos momentos de aprendizado e alegria compartilhados ao longo desta jornada.

Agradeço ao professor Aníbal Santos Jukemura pelo excelente trabalho realizado na orientação deste TCC.

Agradeço a todos os professores e professoras do curso de Ciência da Computação da Escola Politécnica e de Artes por todo aprendizado e orientação oferecidos durante a minha formação acadêmica.

RESUMO

As instituições financeiras têm transformado suas estratégias para concessão de crédito através da análise de dados históricos e do uso de tecnologias de inteligência artificial para apoiar as suas decisões, visando maior precisão nas análises e redução da exposição aos riscos de inadimplência. Este trabalho tem como propósito aplicar e comparar algoritmos de *Machine Learning* para classificação preditiva de clientes com potencial de inadimplência. O objetivo é demonstrar que algoritmos de *Machine Learning* podem ser utilizados como ferramentas preditivas, auxiliando empresas do setor a tomarem decisões mais assertivas. O estudo apresenta uma comparação entre três diferentes algoritmos utilizados em problemas de classificação e predição: *K-Nearest Neighbors*, *Random Forest* e XGBoost. Após a realização dos experimentos e a coleta dos resultados, observou-se que, entre os algoritmos analisados, o XGBoost apresentou o melhor desempenho. Conclui-se que o uso da inteligência artificial através do *Machine Learning* nas análises de risco de crédito pode contribuir significativamente para melhorar as decisões de concessão de crédito, e que a implementação do algoritmo XGBoost pode ser um importante aliado nesse processo.

Palavras-chave: Crédito. Análise de Dados. Inteligência Artificial. *Machine Learning*. *K-Nearest Neighbors*. *Random Forest*. *XGBoost.*

ABSTRACT

Financial institutions have been transforming their credit granting strategies by analyzing historical data and using artificial intelligence technologies to support their decisions, aiming for greater accuracy in analyses and reducing exposure to default risks. This work aims to apply and compare Machine Learning algorithms for predictive classification of customers with potential for default. The objective is to demonstrate that Machine Learning algorithms can be used as predictive tools, helping companies in the sector to make more assertive decisions. The study presents a comparison between three different algorithms used in classification and prediction problems: K-Nearest Neighbors, Random Forest and XGBoost. After conducting the experiments and collecting the results, it was observed that, among the algorithms analyzed, XGBoost presented the best performance. It is concluded that the use of artificial intelligence through Machine Learning in credit risk analysis can significantly contribute to improving credit granting decisions, and that the implementation of the XGBoost algorithm can be an important ally in this process.

Keywords: Credit. Data Analysis. Artificial Intelligence. Machine Learning. K-Nearest Neighbors. Random Forest. XGBoost.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

ADASYN	- *Adaptive Synthetic Sampling Approach for Imbalanced Learning*
ENN	- *Edited Nearest Neighbors*
FN	- Falso Negativo
FP	- Falso Positivo
IA	- Inteligência Artificial
KNN	- *K-Nearest Neighbors*
PCA	- *Principal Component Analysis*
SMOTE	- *Synthetic Minority Oversampling Technique*
SVM	- *Support Vector Machine*
VN	- Verdadeiro Negativo
VP	- Verdadeiro Positivo
XGBoost	- *eXtreme Gradient Boosting*

SUMÁRIO

1 INTRODUÇÃO

O crédito, no contexto financeiro, constitui-se como uma importante fonte de recursos que viabiliza a concretização de objetivos que demandam investimento de capital. Sua base está na relação de confiança entre cliente e credor, estabelecida a partir de uma promessa de pagamento dentro do prazo acordado em contrato no momento da aquisição da dívida. Contudo, essa característica torna as operações de concessão de crédito suscetíveis ao risco de inadimplência. Portanto, é fundamental que o credor realize uma análise da saúde e da capacidade de pagamento do cliente antes de conceder o capital (SANTOS, 2024).

Em décadas anteriores, a análise de crédito era realizada com base em um número muito limitado de variáveis, frequentemente restringindo-se aos dados básicos de identificação do cliente. No entanto, com o aumento da inadimplência, as instituições financeiras passaram a investir na contratação de profissionais especializados e a ampliar a coleta de informações sobre os seus clientes, incluindo os seus históricos financeiros. Esse avanço possibilitou a realização de análises mais detalhadas e uma avaliação mais precisa dos riscos envolvidos nas operações de crédito (SEBBEN, 2020).

Um dos mecanismos adotados para analisar essa vasta quantidade de informações é a Ciência de Dados. Ela é composta de métodos desenvolvidos com o propósito de extrair informações significativas por meio da análise de conjuntos de dados extensos e complexos. A Ciência de Dados combina conhecimentos provenientes de áreas como a Matemática, Estatística e Ciência da Computação (RAUTENBERG; CARMO, 2019).

De acordo com Kalinowski et al. (2023) um projeto Ciência de Dados pode ser dividido em sete etapas: definição do problema, coleta e análise dos dados, pré-processamento, modelagem, pós-processamento, apresentação de resultados e implantação do modelo e geração de valor. Freitas (2023) menciona três tipos de análises realizadas na Ciência de Dados: análise descritiva, análise preditiva e análise prescritiva. A análise descritiva procura elucidar eventos passados, a análise preditiva

visa prever um evento futuro e a análise prescritiva procura sugerir medidas para agir diante de eventos em potencial.

Segundo a Clearsale (2022), empresa especializada em soluções antifraude e score de crédito, os modelos preditivos são muito úteis para a prevenção de riscos. Portanto, adotar a abordagem de análise preditiva se torna relevante, contribuindo para uma avaliação mais precisa do risco de inadimplência. Essa abordagem busca antecipar padrões futuros com base em dados históricos, utilizando técnicas de previsão, correspondência de padrões e modelagem preditiva.

Na análise preditiva estão inseridas ferramentas como o *Machine Learning* ou aprendizado de máquina. *Machine Learning* é uma subárea de conhecimento que pertence ao ramo da Inteligência Artificial (IA), seu objetivo é desenvolver algoritmos que capacitam computadores a aprender padrões de forma autônoma com base no processamento de conjuntos de dados. Esses algoritmos são capazes de construir modelos preditivos que aprendem a partir de dados históricos, permitindo a tomada de decisões e a previsão de resultados (COSTA; PYRES, 2024).

Sendo assim, é relevante estudar este tema devido à crescente evolução e complexidade das operações financeiras, as quais expõem as empresas credoras a diversos riscos nas suas atividades de concessão de crédito. Esses riscos incluem perdas decorrentes da inadimplência dos tomadores de crédito, seja pela falta de pagamento ou incapacidade de cumprir com as obrigações financeiras, além de outros fatores como a deterioração dos contratos de crédito, custos operacionais e complicações na recuperação do crédito (JUNIOR; KLEFENS, 2015). Nesse sentido, torna-se necessário que tais empresas busquem estratégias eficazes para reduzir a sua exposição a esses riscos, a fim de evitar a insuficiência financeira e eventual falência.

Diante desse contexto, este trabalho visa responder à seguinte questão de pesquisa: **Como modelos preditivos baseados em *Machine Learning* podem contribuir na identificação de clientes propensos a se tornarem inadimplentes?**

1.1 Objetivo

- Analisar modelos de *Machine Learning* para previsão do risco de crédito, visando identificar potenciais clientes suscetíveis à inadimplência.

1.1.1 Objetivos Específicos

- Apresentar conceitos de *Machine Learning* relevantes na Ciência de Dados;
- Pesquisar e tratar um conjunto de dados relacionado a ativos de crédito;
- Explorar modelos de *Machine Learning* para classificação;
- Aplicar técnicas de ajuste de hiperparâmetros;
- Realizar testes de predição.

1.2 Estrutura do trabalho

O presente trabalho está organizado da seguinte maneira:

- O Capítulo 2 apresenta uma revisão bibliográfica dos conceitos relacionados à IA e *Machine Learning*, bem como os algoritmos abordados neste estudo.
- O Capítulo 3 oferece um resumo de trabalhos que serviram de referência para o desenvolvimento deste estudo e que estão relacionados à mesma área de estudo.
- O Capítulo 4 descreve a metodologia utilizada no desenvolvimento deste trabalho, além do ambiente, tecnologias e métricas de avaliação adotadas.
- O Capítulo 5 traz uma análise dos resultados obtidos nos experimentos realizados.
- O Capítulo 6 apresenta as conclusões baseadas nos resultados observados, bem como sugestões para futuros trabalhos relacionados à predição com *Machine Learning*.

2 REVISÃO BIBLIOGRÁFICA

2.1 Inteligência Artificial

A IA é uma área de estudo da Ciência da Computação que consiste no desenvolvimento de algoritmos capazes de simular a capacidade humana de resolver problemas. Os estudos sobre a possibilidade de um computador possuir inteligência remontam à década de 1950, quando Alan Turing propôs um questionamento sobre a capacidade das máquinas de pensarem por conta própria. Diante disso, ele desenvolveu um teste que consiste em um jogo de perguntas e respostas, no qual uma pessoa elaborava algumas perguntas e deveria distinguir se as respostas foram fornecidas por outra pessoa ou por uma máquina. Esse teste ficou conhecido como o Teste de Turing. No entanto, o termo “Inteligência Artificial” só foi desenvolvido anos mais tarde, em 1956, por John McCarthy, durante a primeira conferência sobre IA organizada no Dartmouth College (IBM, 2024a).

2.2 Machine Learning

O *Machine Learning* é uma subárea da IA, o seu foco está no desenvolvimento de algoritmos que são capazes de aprender padrões a partir de uma vasta quantidade de dados, auxiliando na tomada de decisões. As formas de aprendizado de um algoritmo de *Machine Learning* podem ser dividias em duas principais categorias: aprendizado supervisionado e não supervisionado (GOOGLE CLOUD, 2024).

2.2.1 Aprendizado supervisionado

O aprendizado supervisionado utiliza como base um conjunto de dados rotulados, onde as entradas possuem os seus valores de saída previamente definidos. Os algoritmos que utilizam essa abordagem ajustam os seus pesos até que o erro entre o resultado previsto e o real seja minimizado. Os problemas que utilizam esse tipo de aprendizado podem ser divididos em duas categorias: classificação e regressão. Na classificação, os algoritmos aprendem a partir das características rotuladas no conjunto de dados e, com base nelas, classificam uma entidade considerando suas características. Na regressão, os algoritmos identificam uma

relação entre as variáveis dependentes e independentes para realizar projeções (IBM, 2024b).

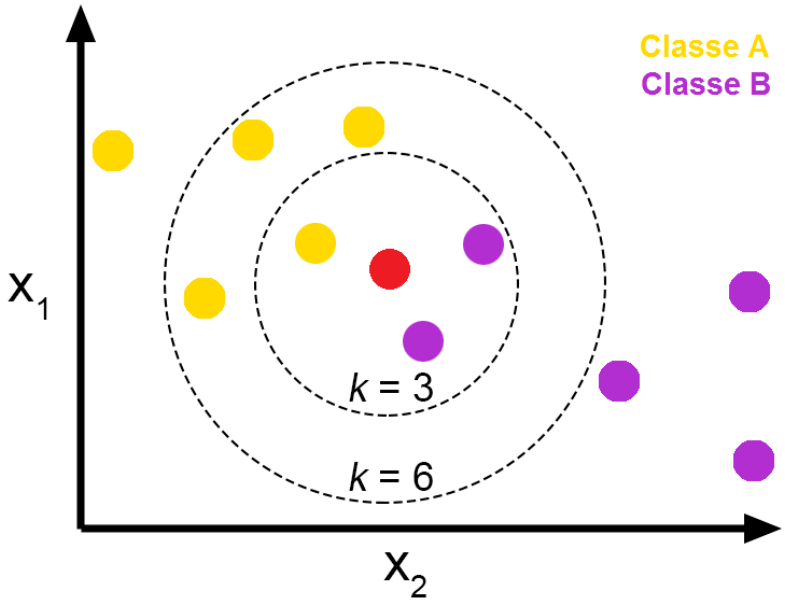
2.2.2 Aprendizado não supervisionado

Diferentemente do aprendizado supervisionado, o não supervisionado utiliza conjuntos de dados não rotulados; ou seja, os dados de entrada não possuem uma saída previamente especificada. Os algoritmos não supervisionados buscam assimilar características dentro do conjunto de dados, criando agrupamentos (*clusters*) entre as entidades que possuem semelhanças entre si, sem a necessidade de intervenção humana (ORACLE, 2024).

2.3 K-Nearest Neighbors

O algoritmo *K-Nearest Neighbors* (KNN), ou algoritmo do vizinho mais próximo, é um dos algoritmos de aprendizado de máquina supervisionado mais simples para aplicações de classificação. Seu conceito baseia-se na ideia de classificar um dado com base em seus k vizinhos mais próximos; ou seja, a classificação se fundamenta na presença de uma classe majoritária observada em uma quantidade determinada de vizinhos próximos (RASCHKA, 2018).

Figura 1 - Gráfico de dispersão para a classificação de um ponto usando KNN



Fonte: José (2018)

A Figura 1 exemplifica o processo de classificação de um ponto observando a presença de uma classe majoritária entre os vizinhos do ponto não classificado (ponto vermelho). Observa-se que, entre os vizinhos mais próximos dentro do limite $k = 3$, a classe predominante é a Classe B (pontos roxos); portanto, o ponto avaliado seria classificado como pertencente à Classe B. Entretanto, se o limite for expandido para $k = 6$, a classe predominante entre os vizinhos mais próximos torna-se a Classe A (pontos amarelos); portanto, neste cenário, o ponto avaliado seria classificado como pertencente à Classe A.

Raschka (2018) expressa que existem diversas métricas para determinar quem são os vizinhos mais próximos de um dado ponto, e a escolha de uma delas depende do problema em questão. Contudo, a distância euclidiana é a mais comumente utilizada, ela mede a distância em linha reta entre o ponto a ser classificado e o ponto de consulta, conforme a equação (1).

$$d(x, y) = \sqrt{\sum_{i=1}^{k} (y_i - x_i)^2} \quad (1)$$

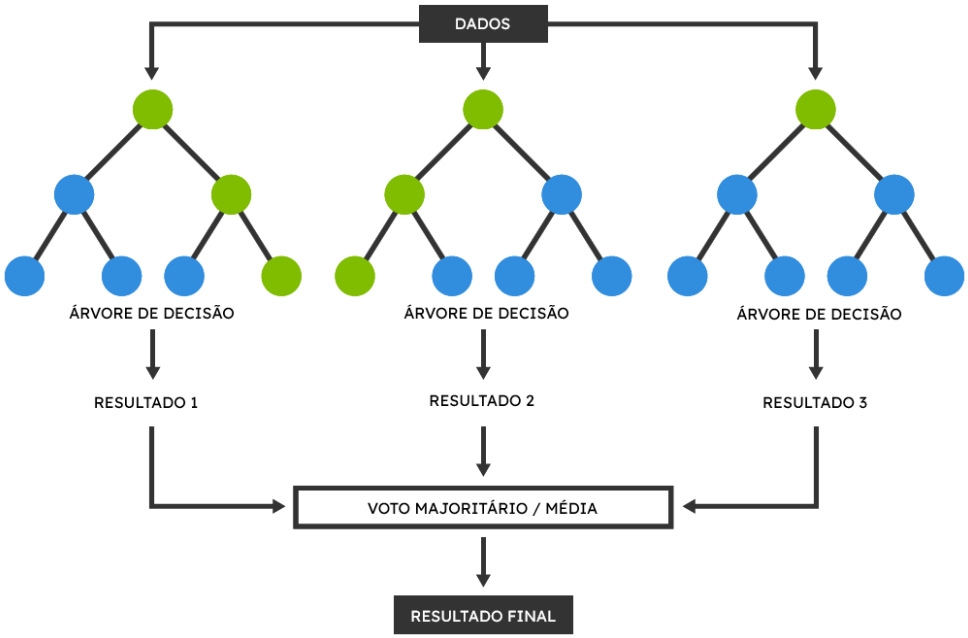
A escolha da quantidade de vizinhos k deve ser equilibrada, levando em consideração o conjunto de dados de entrada, um valor desequilibrado pode resultar em *overfitting* ou *underfitting* do modelo. Um valor baixo para k pode ocasionar alta variância, mas com baixo viés. Já um valor alto para k pode ocasionar o efeito inverso (IBM, 2024c).

Ainda de acordo com IBM (2024c), o KNN não realiza um treinamento explícito; os exemplos de treino são simplesmente armazenados, e o processamento dos dados de entrada é adiado até o momento da previsão. Devido a essa característica, ele é classificado como um algoritmo preguiçoso. O KNN é amplamente aplicado em problemas de pré-processamento de dados, mecanismos de recomendação, assistência médica, reconhecimento de padrões e finanças.

2.4 Random Forest

O algoritmo *Random Forest*, ou floresta aleatória, é um algoritmo de aprendizado supervisionado baseado em modelos de árvore. Um modelo de árvore particiona o conjunto de dados recursivamente em dois subgrupos por meio de um critério de escolha. As árvores de decisão partem de um nó raiz, cada nó possuí uma pergunta cuja resposta gera uma divisão em dois outros nós ou folhas. As divisões se repetem até que a condição de parada estabelecida seja atendida. Os modelos de árvore individualmente não possuem um bom desempenho e são propensos a apresentar *overfitting* e alto viés. Em virtude disso, Leo Breiman propôs o algoritmo denominado *Random Forest*, que consiste na construção de uma coleção de árvores de decisão em conjunto para produzir um resultado melhor, conforme exemplificado na Figura 2 (SCHONLAU; ZOU, 2020).

Figura 2 - Representação do algoritmo *Random Forest* com 3 árvores



Fonte: Adaptado de Gunay (2023)

No modelo proposto por Breiman, as amostras são selecionadas utilizando o método de conjuntos conhecido como *bagging*, ou agregação *bootstrap*. No método de agregação *bootstrap*, as amostras obtidas do conjunto de dados de treinamento são selecionadas utilizando amostragem por substituição. Esse método reduz o

overfitting e proporciona um melhor desempenho para o algoritmo (SCHONLAU; ZOU, 2020).

O algoritmo *Random Forest* pode ser utilizado tanto para problemas de regressão quanto de classificação. Na classificação, o resultado é determinado pelo voto majoritário entre as classes resultantes de cada árvore individualmente. Já na regressão, o resultado é determinado pela média dos resultados individuais das árvores (SCHONLAU; ZOU, 2020). Segundo a IBM (2024d), o algoritmo é aplicado em diversos setores, auxiliando nas decisões de negócio, como finanças, assistência médica e sistemas de recomendação em *e-commerce*.

2.5 XGBoost

O *eXtreme Gradient Boosting* (XGBoost) é um algoritmo de *Machine Learning* que possuí um desempenho significativamente superior quando comparado com outros algoritmos de classificação e regressão, como KNN, Regressão Logística e *Random Forest* (QIN et al., 2021). O XGBoost se baseia em técnicas de *ensemble*, algoritmos de *gradient boosting* e de aprendizado de máquina supervisionado, como árvores de decisão. Assim como o algoritmo *Random Forest*, o XGBoost realiza a construção e a combinação de várias árvores de decisão (NVIDIA, 2024).

O XGBoost utiliza o método de *gradient boosting*, que é uma extensão do método de *boosting*. Esse método consiste em minimizar os erros através da combinação de modelos fracos de árvores de decisão, com o propósito de produzir um modelo coletivamente forte utilizando um algoritmo de descida de gradiente. Os modelos de árvore são treinados a cada iteração, e o erro produzido por um modelo é utilizado para refinar os modelos seguintes (MITCHELL, 2017).

Além disso, o XGBoost realiza o treinamento dos modelos de árvore em paralelo, proporcionando um ganho computacional considerável (NVIDIA, 2024). Segundo Gomes (2019) o algoritmo XGBoost é aplicado em problemas de análise de crédito, diagnóstico médico, marketing segmentado e previsão de demanda no varejo.

3 TRABALHOS RELACIONADOS

Com a evolução das operações financeiras e a crescente complexidade das operações de crédito, as instituições financeiras enfrentam o desafio de mitigar os riscos associados ao não cumprimento das obrigações financeiras por parte de seus clientes. Com a digitalização cada vez mais presente nos processos de análise de risco, uma vasta quantidade de dados sobre os clientes passou a ser armazenada. Como resposta a essa necessidade, modelos foram desenvolvidos para lidar com essa abundância de dados e auxiliar na redução do risco atrelado a essas operações.

Souza (2021), em seu estudo sobre algoritmos de *Machine Learning* para predição de fraudes em cartões de crédito, utilizou um conjunto de dados contendo uma ampla quantidade de informações de transações de cartões de crédito. Para tal, algumas técnicas foram empregadas para lidar com seu conjunto de dados, como a redução da dimensionalidade utilizando *Principal Component Analysis* (PCA), preservando as características de maior relevância. Além do PCA, Souza (2021) utilizou abordagens para o balanceamento das classes da variável de interesse. Souza (2021) abordou alguns algoritmos de *Machine Learning* para realizar previsões sob o conjunto de dados tratado. A fim de obter o melhor desempenho possível, o autor utilizou técnicas para realizar o ajuste dos hiperparâmetros fornecidos aos algoritmos. Ele conclui o seu trabalho realizando um comparativo entre os resultados de desempenho obtidos por cada algoritmo.

Silva (2022), em sua obra sobre modelos de predição de risco de crédito utilizando *Machine Learning* para a identificação de ativos problemáticos em instituições financeiras, realizou uma preparação inicial do conjunto de dados escolhido, aplicando técnicas para lidar com valores ausentes e variáveis categóricas. Silva (2022) também utilizou técnicas para normalização de *features* e balanceamento dos dados. Os algoritmos de *Machine Learning* escolhidos foram validados utilizando técnicas como *Permutation Feature Importance* e *Cross Validation* (validação cruzada). Silva (2022) conclui realizando uma avaliação dos modelos e a exploração dos resultados obtidos.

Já Lopes, Colombi e Mutz (2023) desenvolveram um trabalho comparativo entre diversos algoritmos de aprendizado de máquina para previsão de pontuação de

crédito, aplicados em três bases de dados distintas. Em seu estudo, Lopes, Colombi e Mutz (2023) realizaram o pré-processamento dos conjuntos de dados utilizando técnicas de padronização e redução da dimensionalidade com PCA. Em seguida, aplicaram alguns experimentos com os algoritmos de *Machine Learning* nos três conjuntos de dados, utilizando métricas para apurar o desempenho de cada algoritmo. Lopes, Colombi e Mutz (2023) concluem a sua obra com uma discussão sobre os resultados obtidos pelos algoritmos sob os diferentes conjuntos de dados.

Antes de proceder com o treinamento de um modelo em um conjunto de dados, é essencial realizar uma análise minuciosa da qualidade desses dados. Este processo é conhecido como pré-processamento, durante o qual a base de dados é preparada para garantir a adequação aos modelos, visando obter um desempenho ótimo. Silva (2022) utilizou técnicas para lidar com valores ausentes nas variáveis do conjunto de dados. Essas técnicas incluem o preenchimento dos valores ausentes utilizando métodos estatísticos como média, mediana ou moda. Além disso, há abordagens que optam por remover apenas os registros que possuem variáveis com valores ausentes.

Outra etapa importante nesse processo é o tratamento das variáveis categóricas. Silva (2022) adotou a abordagem de criação de variáveis *dummy*, a qual transforma as classes de uma variável categórica em novas variáveis, indicando a presença ou ausência daquela classe em um determinado registro.

Em *Machine Learning*, é comum a ocorrência de problemas que lidam com conjuntos de dados desbalanceados, onde a discrepância entre as quantidades de observações das diferentes classes é significativa. Em seu trabalho, Souza (2021) explorou duas metodologias para lidar com o balanceamento de dados: sobreamostragem e subamostragem. Este trabalho propõe abordar ambas as técnicas, visando observar e comparar o desempenho dos modelos em diferentes cenários de balanceamento.

Para apurar a capacidade de generalização dos modelos, Lopes, Colombi e Mutz (2023) utilizaram a técnica de validação cruzada. Essa técnica realiza a divisão aleatória do conjunto de dados de treinamento em k subconjuntos de treino e teste, conhecidos como *folds*. Ela permite avaliar a capacidade do modelo em aprender com os dados, identificando se há sobreajuste (*overfitting*), no qual o modelo se especializa

excessivamente em uma característica específica dos dados, ou subajuste (*underfitting*), quando o modelo não consegue se ajustar adequadamente ao conjunto de dados. De semelhante modo, Silva (2022) também utilizou a validação cruzada em sua pesquisa para avaliar a capacidade de generalização dos modelos empregados.

Como maneira de otimizar o desempenho, Silva (2022) utilizou a técnica de ajuste de hiperparâmetros. Essa técnica envolve a realização de uma busca pelas melhores configurações de parâmetros disponíveis, visando aprimorar os resultados de um modelo. De maneira similar, Souza (2021) também utilizou essa técnica para melhorar os resultados dos modelos utilizados em seu trabalho.

Na análise de desempenho realizada por Lopes, Colombi e Mutz (2023) foram empregadas algumas métricas essenciais para avaliar os resultados dos modelos aplicados, utilizando a matriz de confusão como base. As métricas utilizadas por Lopes, Colombi e Mutz (2023) na apuração da eficácia dos modelos foram: acurácia, precisão, revocação e F1-Score. Seguindo uma abordagem semelhante, Souza (2021) e Silva (2022) também adotaram essas métricas em seus trabalhos.

No contexto de previsão do risco de crédito, é particularmente relevante para as empresas a capacidade do modelo em identificar corretamente os possíveis casos de inadimplência. Nesse sentido, Lopes, Colombi e Mutz (2023) ressaltam que a avaliação da métrica de revocação assume uma importância significativa, pois ela mensura a taxa de acerto na detecção dos casos de inadimplência previstos. Portanto, um resultado mais elevado nesta métrica indica um melhor desempenho do modelo para este cenário específico.

Existem diversos algoritmos para lidar com problemas de previsão. No estudo desenvolvido por Lopes, Colombi e Mutz (2023), foi realizada uma análise comparativa entre nove desses algoritmos de aprendizado de máquina, aplicados na previsão de pontuação de crédito. Após a avaliação com validação cruzada, o algoritmo XGBoost alcançou a segunda posição com base na média de acurácia, ficando atrás somente do algoritmo *Stack*. Conforme a Figura 3, o algoritmo também apresentou o melhor resultado de F1-Score, demonstrando um bom equilíbrio entre precisão e revocação.

Figura 3 - Resultados médios obtidos por Lopes, Colombi e Mutz (2023)

Modelo	Acurácia	Precisão	Revocação	*F1 Score*
AdaBoost	76,14%	59,95%	53,63%	55,94%
KNN	78,77%	65,26%	50,62%	56,36%
Regressão Logística	80,48%	70,92%	52,69%	57,66%
MLP	80,15%	68,14%	56,73%	61,14%
Random Forest	80,40%	67,91%	54,92%	59,86%
Stack	**81,41%**	**71,40%**	55,54%	61,31%
SVM	72,47%	51,81%	**56,82%**	54,05%
Árvore de Decisão	75,87%	57,73%	55,25%	56,45%
XGBoost	80,87%	69,51%	56,76%	**61,65%**

Fonte: Lopes, Colombi e Mutz (2023)

Souza (2021), em seu estudo de algoritmos para previsão de fraudes em transações com cartões de crédito, utilizou os algoritmos KNN, SVM e *Random Forest*. Seus resultados mostram que o KNN apresentou uma boa média entre acurácia e F1-Score quando combinado com a técnica de sobreamostragem *Synthetic Minority Oversampling Technique* (SMOTE), utilizando 40% do conjunto de dados para treino, conforme a Figura 4. No entanto, as médias obtidas pelo SVM e *Random Forest* foram superiores às do KNN ao utilizar 60% do conjunto de dados para treino, aplicando a mesma técnica, conforme a Figura 5 e Figura 6 respectivamente.

Figura 4 - Resultados obtidos por Souza (2021) para o algoritmo KNN

MÉDIA – k-NN		
Método de Balanceamento	Método de Seleção de atributos	
	60% do conjunto	40% do conjunto
RU	0.7110	0.8969
SMOTE	0.9981	**0.9988**

Fonte: Souza (2021)

Figura 5 - Resultados obtidos por Souza (2021) para o algoritmo SVM

MÉDIA – SVM		
Método de Balanceamento	Método de Seleção de atributos	
	60% do conjunto	40% do conjunto
RU	0.9960	0.9291
SMOTE	**0.9984**	0.9973

Fonte: Souza (2021)

Figura 6 - Resultados obtidos por Souza (2021) para o algoritmo *Random Forest*

MÉDIA – *Random Forest*		
Método de Balanceamento	Método de Seleção de atributos	
	60% do conjunto	40% do conjunto
RU	0.9862	0.9376
SMOTE	**0.9994**	0.9989

Fonte: Souza (2021)

Silva (2022) em seu estudo sobre modelos de predição de risco de crédito, utilizou os algoritmos de Regressão Logística, Random Forest e Gradient Boosting, sendo o XGBoost uma de suas implementações mais conhecidas. Em seus experimentos, Silva (2022) aplicou diferentes proporções de amostras da base de dados e avaliou os resultados obtidos por cada modelo, conforme apresentado na Figura 7, Figura 8 e Figura 9. Seus resultados mostram que, embora o algoritmo de Regressão Logística tenha demonstrado um desempenho relevante, seus resultados foram inferiores aos obtidos pelos algoritmos Random Forest e XGBoost.

Figura 7 - Resultados de Silva (2022) para as amostras 1/1 e 3/1 respectivamente

Índice	RF	LR	XG
Acurácia	84.81%	78.65%	84.04%
AUC	84.81%	78.65%	84.04%
F1	83.94%	77.62%	83.06%
Recall	79.4%	74.06%	78.27%
Precision	89.04%	81.54%	88.48%
MSE	0.15	0.21	0.16
MAE	0.15	0.21	0.16

Resultados das Validações dos Modelos

Índice	RF	LR	XG
Acurácia	98.27%	93.64%	98.38%
AUC	98.52%	89.06%	98.63%
F1	96.63%	86.26%	96.83%
Recall	99.0%	79.89%	99.13%
Precision	94.37%	93.74%	94.63%
MSE	0.02	0.06	0.02
MAE	0.02	0.06	0.02

Resultados das Validações dos Modelos

Fonte: Silva (2022)

Figura 8 - Resultados de Silva (2022) para as amostras 5/1 e 7/1 respectivamente

Índice	RF	LR	XG
Acurácia	98.4%	94.42%	98.42%
AUC	98.5%	86.0%	98.54%
F1	95.37%	81.41%	95.41%
Recall	98.65%	73.38%	98.74%
Precision	92.31%	91.42%	92.3%
MSE	0.02	0.06	0.02
MAE	0.02	0.06	0.02

Resultados das Validações dos Modelos

Índice	RF	LR	XG
Acurácia	98.35%	95.24%	98.4%
AUC	98.24%	84.28%	98.44%
F1	93.71%	78.53%	93.9%
Recall	98.08%	69.68%	98.49%
Precision	89.71%	89.96%	89.72%
MSE	0.02	0.05	0.02
MAE	0.02	0.05	0.02

Resultados das Validações dos Modelos

Fonte: Silva (2022)

Figura 9 - Resultados de Silva (2022) para as amostras 10/1 e total respectivamente

Índice	RF	LR	XG
Acurácia	98.37%	95.98%	98.39%
AUC	97.44%	82.14%	97.89%
F1	91.47%	74.68%	91.67%
Recall	96.32%	65.21%	97.28%
Precision	87.1%	87.37%	86.68%
MSE	0.02	0.04	0.02
MAE	0.02	0.04	0.02

Resultados das Validações dos Modelos

Índice	RF	LR	XG
Acurácia	99.47%	99.17%	99.45%
AUC	76.26%	68.42%	77.19%
F1	65.29%	45.83%	65.16%
Recall	52.61%	37.09%	54.5%
Precision	86.05%	59.96%	80.99%
MSE	0.01	0.01	0.01
MAE	0.01	0.01	0.01

Resultados das Validações dos Modelos

Fonte: Silva (2022)

Os resultados apresentados por Silva (2022) e por Lopes, Colombi e Mutz (2023) demonstram uma concordância entre si, destacando o algoritmo XGBoost entre os de melhor desempenho. Além disso, é possível notar que, em ambos os casos, o XGBoost demonstra superioridade em relação ao algoritmo *Random Forest*. De maneira semelhante, os resultados apresentados por Silva (2022) também estão alinhados aos de Souza (2021), com relação ao desempenho do algoritmo *Random Forest*, que apresentou resultados expressivos e relevantes. Dessa forma, espera-se que os algoritmos XGBoost e *Random Forest* apresentem resultados promissores.

4 METODOLOGIA

Esta pesquisa segundo a sua natureza é um resumo de assunto, buscando explicar conceitos e definições comuns dentro da área de estudo deste trabalho. Um resumo de assunto busca traçar uma evolução histórica desses conceitos e definições, resultado de uma investigação cuidadosa das informações obtidas, levando ao entendimento das causas e explicações dos fenômenos estudados (WAZLAWICK, 2024).

Segundo os seus objetivos, esta pesquisa é exploratória e descritiva. Na pesquisa exploratória, o autor não parte de uma hipótese ou um objetivo predefinido, no entanto, busca construir uma base de conhecimento por meio de observações para a elaboração de uma pesquisa mais completa. A pesquisa descritiva visa obter dados mais detalhados e consistentes sobre o fenômeno em estudo, sem a necessidade de desenvolver teorias explicativas. Seu objetivo reside em descrever os fatos conforme eles são em sua originalidade (WAZLAWICK, 2024).

Com relação aos seus procedimentos técnicos, esta pesquisa é bibliográfica e experimental. A pesquisa bibliográfica fundamenta-se no material já publicado, explorando livros, periódicos, documentos, textos, imagens, manuscritos, recursos disponíveis na internet, entre outros. Esse material desempenha um papel fundamental em todas as etapas do processo de pesquisa, contribuindo para a formulação de hipóteses, justificação do estudo e a elaboração do relatório final (FONTELLES et al., 2009).

A pesquisa experimental parte da formulação de um questionamento ou problema, ela estabelece uma hipótese traçando uma relação de causa e resultado entre as variáveis. Essas variáveis devem proporcionar o esclarecimento da hipótese levantada através da definição operacional. Nessa abordagem a variável objetivo deve ser claramente estabelecida, juntamente com os procedimentos para mensurar e avaliar os resultados (GIL, 2017).

Ainda de acordo com Gil (2017), na pesquisa experimental devem ser determinados os sujeitos a partir de uma amostra que seja capaz de proporcionar uma generalização dos resultados para a população total. Em seguida, ocorre a definição do ambiente onde serão conduzidos os experimentos, esse ambiente deve fornecer

todos os recursos necessários para a realização dos experimentos. Posteriormente, os resultados serão coletados e interpretados utilizando análise estatística, concluindo com uma discussão baseada no conhecimento obtido na fundamentação teórica.

4.1 Ambiente de desenvolvimento

Este capítulo aborda o ambiente de desenvolvimento, as ferramentas e os recursos utilizados na realização dos experimentos. Entre as ferramentas empregadas, destacam-se o Kaggle e o Google Colaboratory, juntamente com a linguagem Python e suas bibliotecas, que foram fundamentais para a obtenção e o tratamento do conjunto de dados, a visualização e exploração dos dados, bem como para a aplicação dos modelos propostos, testes, métricas e ajustes de hiperparâmetros. O espaço de trabalho utilizado para o desenvolvimento dos experimentos e aplicação dos algoritmos foi hospedado na plataforma GitHub[1].

4.1.1 Ambiente

O conjunto de dados utilizado para o treinamento dos algoritmos de aprendizado de máquina foi obtido através da plataforma Kaggle[2]. O Kaggle é uma plataforma online amplamente utilizada por cientistas de dados e estudantes para obtenção de *datasets* voltados a diversas aplicações, especialmente nas áreas de Ciência de Dados e IA. Além disso, ela facilita a troca de conhecimento entre os usuários, permitindo que os espaços de trabalho, chamados de *notebooks*, sejam compartilhados e utilizados para fins de aprendizagem (CATUNDA, 2022b).

Para trabalhar com o conjunto de dados selecionado e aplicar os algoritmos propostos, utilizou-se o Google Colaboratory. Popularmente conhecido como Colab, o Google Colaboratory é uma plataforma em nuvem desenvolvida e mantida pela Google, com o intuito de incentivar as pesquisas em IA e áreas correlacionadas. Seu objetivo é proporcionar um ambiente de desenvolvimento que permita a estudantes, pesquisadores e profissionais realizar trabalhos que envolvam programação, sem a

[1] Disponível em <https://doi.org/10.5281/zenodo.14401430> Acesso em: 11 dez, 2024.
[2] Disponível em <https://www.kaggle.com/datasets/laotse/credit-risk-dataset> Acesso em: 11 dez, 2024.

necessidade de um *hardware* de alto desempenho ou um ambiente de desenvolvimento local (SANTOS, 2023).

De acordo com Santos (2023) o Colab é utilizado predominantemente com a linguagem de programação Python, a qual foi empregada para lidar com o conjunto de dados e os algoritmos, conforme será detalhado nos capítulos seguintes.

4.1.2 Python

Python é uma linguagem de programação interpretada e multiplataforma, altamente versátil, utilizada para diversas finalidades, como desenvolvimento de *softwares*, *websites*, jogos, automação, entre outros. Além disso, a linguagem é amplamente empregada nas áreas de Ciência de Dados e IA, oferecendo uma vasta gama de bibliotecas tanto para análise e manipulação de dados quanto para implementações de algoritmos de IA (CATUNDA, 2022a). As bibliotecas utilizadas no desenvolvimento deste trabalho serão apresentadas no capítulo seguinte.

4.1.3 Bibliotecas

As bibliotecas de uma linguagem de programação visam proporcionar abstrações de implementações de algoritmos e funcionalidades úteis aos programadores, auxiliando no processo de desenvolvimento. Para a execução dos experimentos deste trabalho, foram utilizadas as seguintes bibliotecas do Python:

- ***Pandas*:** *Pandas* é uma biblioteca amplamente utilizada para lidar com grandes volumes de dados, sendo essencial para aplicações em Ciência de Dados. Ela proporciona mecanismos de manipulação, tratamento, limpeza e análise estatística de dados, além de integrar-se com outras bibliotecas, como *Scikit-learn, Matplotlib* e *Seaborn* (MONUTTI, 2024).
- ***Seaborn*:** *Seaborn* é uma biblioteca muito popular para visualização gráfica de dados. Ela oferece suporte para a geração de uma grande variedade de gráficos, sendo particularmente útil na análise exploratória dos dados (LAGO, 2022). A visualização gráfica permite ao pesquisador obter uma compreensão mais ampla sobre a distribuição dos dados trabalhados.

- ***Matplotlib*:** Assim como *Seaborn, Matplotlib* também é uma biblioteca utilizada para visualização gráfica de dados, oferecendo uma interface simples e flexível para a criação de gráficos (ANDRADE, 2023).
- ***Scikit-learn*:** *Scikit-learn* é uma biblioteca de código aberto que fornece uma interface eficiente para aplicações de aprendizado de máquina, como classificação, regressão, *clustering* e redução de dimensionalidade de dados. Ela disponibiliza um amplo conjunto de algoritmos de aprendizado de máquina, além de ferramentas para pré-processamento de dados, seleção de hiperparâmetros e avaliação de modelos (HABBEMA, 2024).
- ***Imblearn*:** *Imblearn* é uma biblioteca que oferece uma interface para lidar com classes desbalanceadas em *datasets*, utilizando técnicas de amostragem, como *oversampling* e *undersampling* (SANTIAGO, 2023).
- **XGBoost:** XGBoost é uma biblioteca otimizada que fornece uma implementação eficiente do algoritmo XGBoost, utilizado para solução de problemas relacionados a *Machine Learning* (AWARI, 2023).

4.2 Métricas de avaliação

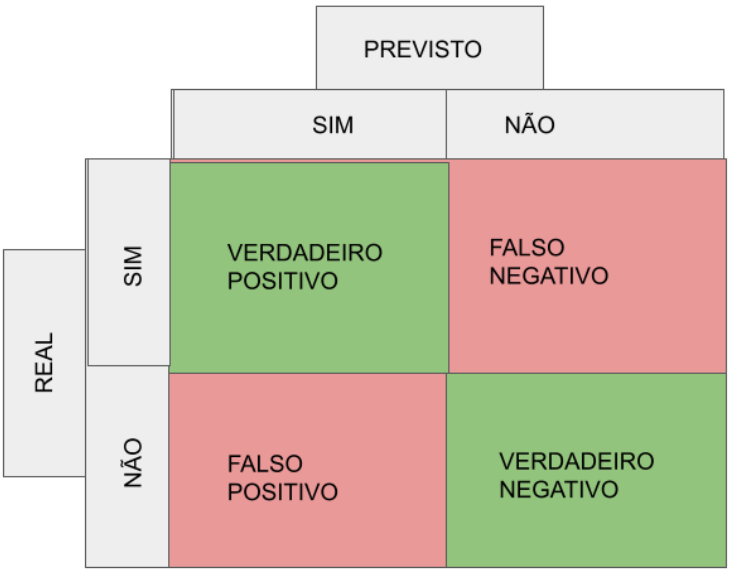
As métricas de avaliação constituem um importante mecanismo para mensurar o desempenho de um algoritmo de *Machine Learning*. Por meio delas, é possível identificar desequilíbrios no aprendizado do modelo, como *overfitting* e *underfitting*, e realizar os ajustes necessários para otimizar o desempenho. As métricas abordadas neste trabalho incluem acurácia, precisão, revocação e F1-Score. Grande parte delas é derivada a partir de uma matriz de confusão, exceto a métrica F1-Score, que considera os resultados de precisão e revocação.

4.2.1 Matriz de confusão

A matriz de confusão é uma forma simples de visualização dos resultados produzidos por um algoritmo de *Machine Learning* utilizado para classificação. Ela representa a quantidade de exemplos classificados pelo algoritmo como Falso Positivo (FP), Falso Negativo (FN), Verdadeiro Positivo (VP) e Verdadeiro Negativo

(VN), permitindo identificar quantos exemplos foram classificados corretamente ou de forma equivocada (KUNUMI, 2020). Algumas das principais métricas aplicadas em *Machine Learning* têm como base os resultados da matriz de confusão. A estrutura dessa matriz pode ser observada na Figura 10.

Figura 10 - Matriz de confusão



Fonte: Sousa (2022)

Um resultado FP indica que o algoritmo classificou erroneamente um indivíduo como pertencente à classe positiva (JUNIOR et al., 2022). No cenário de classificação do risco de crédito em instituições financeiras, um FP seria o equivalente a classificar um cliente adimplente como inadimplente, restringindo possivelmente o seu acesso a crédito na instituição.

Um resultado FN indica que o algoritmo classificou erroneamente um indivíduo como pertencente à classe negativa (JUNIOR et al., 2022). Essa classificação é potencialmente problemática, pois seria o equivalente a classificar um cliente propenso à inadimplência como um possível cliente adimplente. Os resultados VP e VN indicam a quantidade de classificações corretas de um algoritmo de *Machine Learning*, tanto para a classe positiva quanto para a classe negativa, respectivamente.

4.2.2 Acurácia

A acurácia é uma métrica que representa a quantidade de exemplos que foram corretamente classificados. Ela mede o percentual de resultados verdadeiros (VP e VN) em relação ao total de resultados. A acurácia é uma métrica amplamente utilizada; todavia, ela não é capaz de expressar os erros por FP e FN, principalmente em conjuntos de dados desbalanceados (KUNUMI, 2020). A acurácia pode ser calculada através da equação (2).

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \quad (2)$$

4.2.3 Precisão

A precisão, por sua vez, representa o percentual de classificações corretas dentro de todas as classificações positivas (VP e FP), ou seja, mede a assertividade das classificações que são realmente positivas. É uma medida muito útil para mensurar o impacto de FPs nos resultados do algoritmo (JUNIOR et al., 2022). Seu cálculo pode ser realizado por meio da equação (3).

$$Precisão = \frac{VP}{VP + FP} \quad (3)$$

4.2.4 Revocação

A revocação, ou sensibilidade, também conhecida pelo termo em inglês *recall*, representa o percentual de todos os VPs corretamente previstos dentre todos os exemplos positivos. A revocação é uma métrica muito relevante para avaliar os cenários em que os FNs são potencialmente prejudiciais, permitindo observar a capacidade do algoritmo em identificar FNs (JUNIOR et al., 2022). Através da revocação, é possível avaliar se o algoritmo está sendo capaz de identificar corretamente os clientes com potencial para a inadimplência. A revocação pode ser obtida através da equação (4).

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

4.2.5 F1-Score

O F1-score é uma métrica que associa a precisão e a revocação, permitindo avaliar se há um equilíbrio entre essas duas métricas; ou seja, um modelo que possuí um F1-score alto é capaz de acertar as suas previsões e os exemplos da classe positiva, representando uma grande confiabilidade do algoritmo (KUNUMI, 2020). Seu valor pode ser obtido através da média harmônica da precisão e da sensibilidade, conforme a equação (5).

$$F1 = 2 * \left(\frac{precisão * revocação}{precição + revocação}\right) \quad (5)$$

4.3 *Feature Engineering*

Feature Engineering é uma etapa fundamental na construção de modelos de *Machine Learning*, ela é responsável por selecionar as variáveis e melhorar a qualidade dos dados de entrada. Para isso, são utilizadas técnicas de *encoding*, normalização, padronização, entre outras, que podem contribuir para aumentar significativamente o desempenho do modelo. No conjunto de dados utilizado, foram aplicadas algumas dessas técnicas. O *dataset* é composto por 12 variáveis, incluindo variáveis quantitativas e qualitativas, totalizando 32.581 registros. As variáveis presentes no *dataset* estão detalhadas no Quadro 1.

Quadro 1 - Descrição das variáveis do conjunto de dados e seus tipos

Descrição das variáveis do conjunto de dados e seus tipos		
Variável	**Descrição**	**Tipo**
person_age	idade	numérica
person_income	renda anual	numérica
person_home_ownership	tipo de propriedade	categórica
person_emp_length	tempo de emprego	numérica
loan_intent	intenção do empréstimo	categórica
loan_grade	grau de risco do empréstimo	categórica
loan_amnt	valor do empréstimo	numérica
loan_int_rate	taxa de juros	numérica
loan_status	situação do empréstimo	numérica
loan_percent_income	percentual de comprometimento da renda	numérica
cb_person_default_on_file	inadimplência histórica	categórica
cb_person_cred_hist_length	tempo do histórico de crédito	numérica

Fonte: Elaborado pelo autor

Para que os algoritmos possam lidar apenas com dados numéricos, as variáveis categóricas foram transformadas em valores inteiros. Utilizou-se a técnica de *Label Encoding* para as variáveis categóricas binárias, *One-Hot Encoding* para variáveis categóricas não binárias e *Ordinal Encoding* para as variáveis categóricas ordinais. Todas essas transformações foram realizadas utilizando a biblioteca *Scikit-learn*.

Por exemplo, a variável *cb_person_default_on_file* é uma variável categórica nominal binária que representa a ocorrência de inadimplência no histórico do cliente. Ela contém os valores “Y” e “N”, os quais indicam, respectivamente, se o cliente possuí ou não histórico de inadimplência. Esses valores foram convertidos em 1 e 0, representando a presença e a ausência dessa característica, conforme apresentado na Figura 11.

Figura 11 - Transformação da varável *cb_person_default_on_file*

cb_person_default_on_file	cb_person_default_on_file
Y	1
N	0
N	0
N	0
Y	1

Fonte: Elaborado pelo autor

A variável *loan_grade* é uma variável categórica ordinal, ou seja, as categorias possuem uma ordem específica. Ela representa o risco atrelado ao empréstimo, onde os valores têm uma ordem crescente de risco, com a classe “A” indicando baixo risco e a classe “G” indicando um alto risco. Para preservar essa característica, as classes foram convertidas em valores numéricos de forma que a categoria “A” foi representada pelo valor 0, com um incremento de 1 unidade para cada categoria subsequente, até a categoria “G”, conforme apresentado na Figura 12.

Figura 12 - Transformação da variável *loan_grade*

loan_grade	loan_grade
A	0.0
B	1.0
C	2.0
D	3.0
E	4.0
F	5.0
G	6.0

Fonte: Elaborado pelo autor

As variáveis *person_home_ownership* e *loan_intent* são categóricas nominais não binárias. Para representá-las em valores numéricos, as categorias foram transformadas em variáveis *dummy*, cada uma indicando a presença ou ausência de

uma determinada categoria no registro por meio dos valores 0 e 1, o resultado desse procedimento pode ser visualizado na Figura 13.

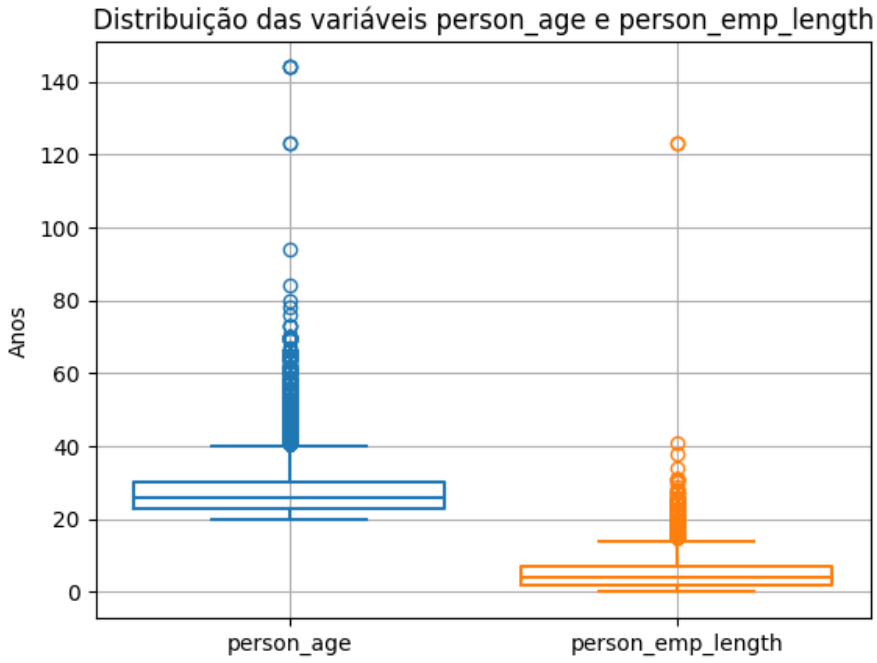
Figura 13 - Transformação da variável *person_home_ownership*

person_home_ownership_MORTGAGE	person_home_ownership_OTHER	person_home_ownership_OWN	person_home_ownership_RENT
0	0	0	1
0	0	1	0
1	0	0	0
0	0	0	1
0	0	0	1

Fonte: Elaborado pelo autor

A Figura 14 apresenta o gráfico boxplot das variáveis *person_age* e *person_emp_length*. Através do gráfico, foi possível identificar a presença de alguns registros com valores altamente discrepantes, conhecidos como *outliers*, que estão distantes dos demais dados e excedem o limite aceitável. Esses *outliers* foram tratados com técnicas estatísticas para minimizar o seu impacto na base de dados. Os valores de idade superior a 100 anos foram substituídos pela média das idades. Similarmente, esse procedimento foi aplicado a variável *person_emp_length*, os valores de tempo de emprego superiores a 50 anos foram substituídos pela média.

Figura 14 - Gráfico da distribuição das variáveis *person_age* e *person_emp_length*



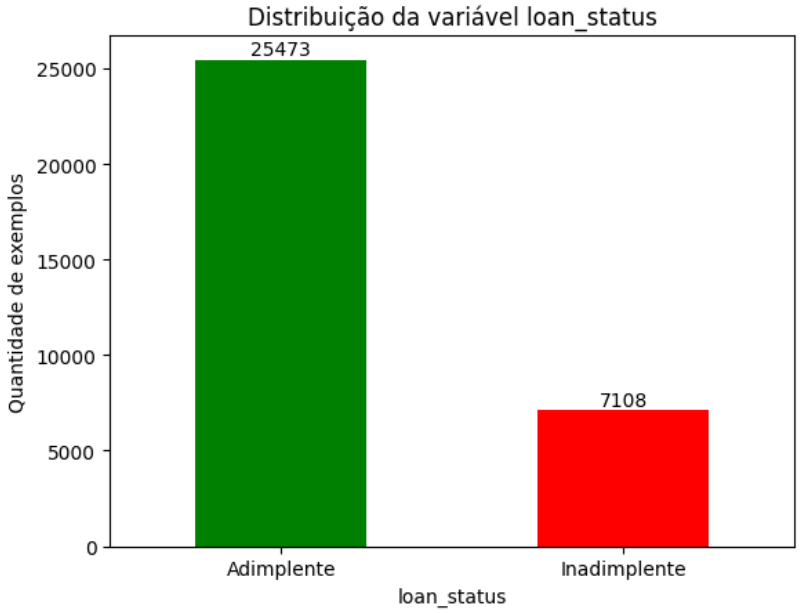
Fonte: Elaborado pelo autor

Ao avaliar as variáveis *person_emp_length* e *loan_int_rate* foi identificada a presença de valores ausentes, correspondendo a aproximadamente 2,75% e 9,56% do total de registros, respectivamente. Para lidar com esses valores ausentes, foram empregadas técnicas estatísticas, preenchendo-os com a média dos valores disponíveis.

4.3.1 Balanceamento

Ao avaliar a variável de interesse *loan_status*, constatou-se a presença de um desbalanceamento entre as classes adimplente e inadimplente, sendo que a classe dos adimplentes apresenta a maior quantidade de exemplos, conforme apresentado na Figura 15. Para melhorar o equilíbrio entre as classes, foram adotadas duas abordagens: sobreamostragem (*oversampling*) e subamostragem (*undersampling*).

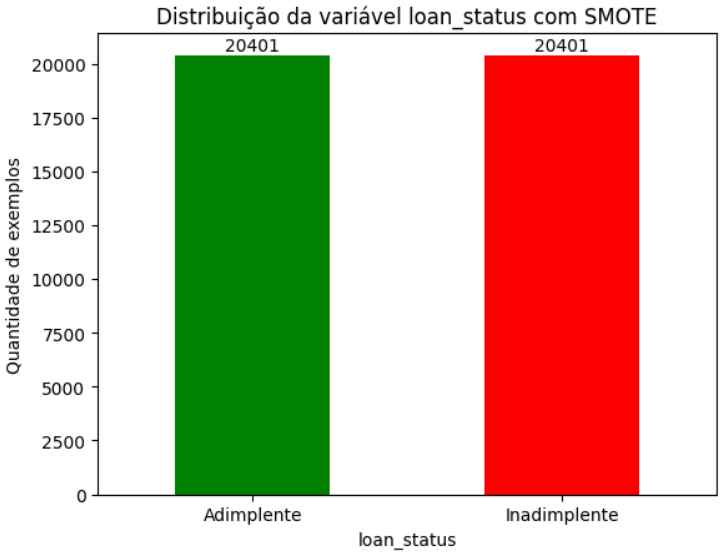
Figura 15 - Gráfico da distribuição das classes da variável *loan_status*



Fonte: Elaborado pelo autor

Para realizar a sobreamostragem, foi utilizada a técnica SMOTE que consiste em gerar novos exemplos sintéticos da classe minoritária, equilibrando a quantidade de exemplos entre as classes. O objetivo dessa técnica é evitar que o algoritmo se especialize excessivamente na classe majoritária, fenômeno conhecido como *overfitting*. Após a aplicação da técnica, os exemplos de ambas as classes foram equilibrados conforme demonstrado no gráfico da Figura 16.

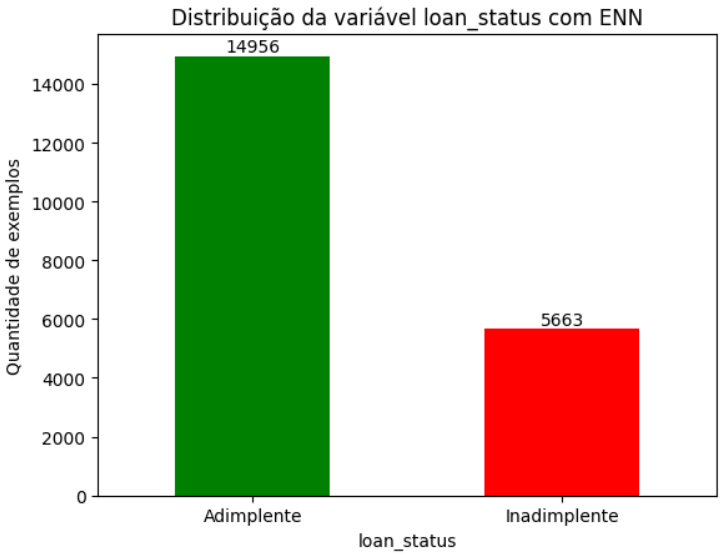
Figura 16 - Gráfico da distribuição da variável *loan_status* com SMOTE



Fonte: Elaborado pelo autor

Para realizar a subamostragem, foi empregada a técnica *Edited Nearest Neighbors* (ENN), que consiste em remover exemplos da classe majoritária que estão distantes da maioria dos exemplos, destacando a distinção entre as duas classes da variável de interesse. Esse método utiliza o algoritmo dos vizinhos mais próximos. Conforme observado no gráfico da Figura 17, o resultado produzido pelo ENN aponta uma redução no número de exemplos da classe majoritária, embora não o suficiente para igualar a quantidade de exemplos da classe minoritária.

Figura 17 - Gráfico da distribuição da variável *loan_status* com ENN



Fonte: Elaborado pelo Autor

4.3.2 Validação Cruzada

A validação cruzada é uma técnica utilizada para testar um modelo com diferentes amostras do conjunto de dados, visando verificar se o modelo não está sofrendo de *overfitting*. As amostras são divididas em dois conjuntos: um para treinamento e outro para o teste e a avaliação do modelo (RABELLO, 2019). Uma das variações dessa técnica é a *K-Fold Cross Validation*.

A técnica *K-Fold Cross Validation* consiste em dividir o conjunto de dados em k subconjuntos, de modo que cada um contenha aproximadamente o mesmo número de exemplos. Em cada iteração, $k - 1$ subconjuntos são utilizados para o treinamento, enquanto o restante é usado para teste e avaliação do modelo. Essa abordagem garante que o modelo seja testado com todos os subconjuntos (RABELLO, 2019).

Brownlee (2023) destaca que os tratamentos aplicados ao conjunto de dados, como o balanceamento de classes e a busca de hiperparâmetros, devem ser realizados separadamente para cada subconjunto utilizado na validação cruzada. Para isso, foram criados *pipelines* com validação cruzada para cada modelo, utilizando as técnicas de balanceamento SMOTE e ENN, conforme apresentado na Figura 18. Os *pipelines* auxiliam na criação de cadeias de transformações e reamostragens, permitido que essas etapas sejam executadas de uma só vez (MARTIN, 2019). Assim, é possível aplicar as técnicas de balanceamento a cada subconjunto da validação cruzada de maneira simplificada.

Figura 18 - Aplicação do *pipeline* para validação cruzada do KNN com SMOTE

```
imba_pipeline_knn_smote = make_pipeline(smote, knn_clf)
results = cross_validate(imba_pipeline_knn_smote, X_train, y_train, cv=kf, scoring=metrics)
results = pd.DataFrame(results).mean()
general_results['knn_smote'] = [
    results['test_accuracy'],
    results['test_recall'],
    results['test_precision'],
    results['test_f1']]
results
```

Fonte: Elaborado pelo autor

4.3.3 Hiperparâmetros

Hiperparâmetros são parâmetros utilizados em algoritmos de aprendizado de máquina, definidos manualmente antes do treinamento do modelo. Eles influenciam diretamente o desempenho e a capacidade de aprendizagem do modelo. Portanto, a escolha dos hiperparâmetros deve ser equilibrada, levando em consideração tanto o desempenho quanto a capacidade de generalização dos resultados, selecionando adequadamente os valores para extrair o melhor desempenho possível do modelo (AWS, 2024).

Os valores ideais de hiperparâmetros variam dependendo do problema e da estrutura dos dados utilizados. Por isso, a escolha dos valores é um processo iterativo que combina diversos valores para alcançar o resultado ideal. Há diversas abordagens para encontrar uma combinação ideal de hiperparâmetros, sendo as mais comuns a pesquisa em grade (*grid search*) e a pesquisa aleatória (*random search*).

A pesquisa em grade se baseia em uma lista de hiperparâmetros a serem analisados e uma métrica de desempenho. Ela realiza uma busca exaustiva, testando todas as possíveis combinações entre os hiperparâmetros fornecidos. No entanto, por explorar todas as combinações, essa abordagem possui um alto custo computacional, que cresce exponencialmente à medida que o número de hiperparâmetros a serem testados aumenta (AWS, 2024).

A pesquisa aleatória, por sua vez, funciona de maneira semelhante à pesquisa em grade, mas seleciona os hiperparâmetros de forma aleatória para determinar o melhor ajuste. A busca aleatória é especialmente útil quando há muitos hiperparâmetros a serem testados (AWS, 2024). No entanto, por sua natureza aleatória, essa técnica pode não identificar a melhor combinação possível de hiperparâmetros.

Em virtude disso, e considerando que a quantidade de hiperparâmetros não é elevada, optou-se por utilizar a pesquisa em grade para obter a melhor combinação para os hiperparâmetros fornecidos. Para isso, foi utilizada a biblioteca *GridSearchCV*, que implementa essa técnica de forma eficiente, conforme demonstrado na Figura 19.

Figura 19 - Aplicação do *GridSearchCV* para o *pipeline* do KNN com SMOTE

```
params = {
    "n_neighbors": [5, 10, 20, 50, 100],
    "weights": ['uniform', 'distance'],
}
new_params = {'kneighborsclassifier__' + key: params[key] for key in params}
```

```
grid_search_knn = GridSearchCV(
    imba_pipeline_knn_smote,
    new_params,
    cv=kf,
    scoring='recall',
    return_train_score=True
)
grid_search_knn.fit(X_train, y_train);
grid_search_knn.best_params_
```

Fonte: Elaborado pelo autor

O parâmetro *scoring* determina a métrica que será utilizada pelo algoritmo para determinar a melhor combinação de hiperparâmetros. Conforme destacado por Lopes, Colombi e Mutz (2023), no contexto de avaliação do risco de crédito, a métrica de revocação é particularmente relevante, pois representa a capacidade do modelo em identificar corretamente os clientes com potencial de inadimplência. Dessa forma, o parâmetro *scoring* foi configurado para utilizar a métrica de *recall* (revocação).

Assim como na etapa de balanceamento das classes da variável alvo, a validação cruzada foi aplicada durante o ajuste de hiperparâmetros, com o auxílio de *pipelines*. Esses *pipelines* permitiram realizar a busca de hiperparâmetros em cada subconjunto da validação cruzada, garantindo a obtenção dos melhores resultados em termos de configuração de hiperparâmetros.

5 DESCRIÇÃO E ANÁLISE DOS RESULTADOS

Para a realização dos experimentos, foi utilizada a proporção de 80% dos dados para o treino e 20% para o teste. Os experimentos foram executados em três etapas sequenciais. Na primeira etapa, cada algoritmo foi submetido à validação cruzada com os conjuntos de dados reamostrados. Para a validação cruzada, foram utilizados 5 subconjuntos. Na segunda etapa, cada algoritmo foi submetido ao processo de pesquisa de hiperparâmetros. Na terceira e última etapa, cada algoritmo foi submetido novamente à validação cruzada, com os hiperparâmetros otimizados.

5.1 KNN

O primeiro algoritmo submetido aos experimentos foi o KNN. Os resultados obtidos após a execução da primeira etapa estão detalhados na Tabela 1.

Tabela 1 - Resultados obtidos com o KNN antes da otimização de hiperparâmetros

Métrica	**SMOTE**	**ENN**
Acurácia	0,771793	0,789595
Revocação	0,662995	0,665315
Precisão	0,481914	0,512150
F1-Score	0,557958	0,578672

Fonte: Elaborado pelo autor

O KNN não obteve resultados expressivos durante a validação cruzada sem otimização de hiperparâmetros. A acurácia resultante ficou abaixo de **80%** para ambos os conjuntos de dados, o que indica que o algoritmo não conseguiu atingir uma boa taxa de acertos em suas previsões. A revocação também foi baixa; ambos os conjuntos apresentaram valores semelhantes, com cerca de **66%** de acerto entre os verdadeiros positivos.

Na segunda etapa, o algoritmo foi submetido à pesquisa em grade. Os parâmetros utilizados foram *n_neighbors* e *weights*. Os valores usados para cada parâmetro na busca estão detalhados na Tabela 2.

Tabela 2 - Parâmetros utilizados na pesquisa em grade para o KNN

Parâmetro	**Valores**
n_neighbors	5, 10, 20, 50, 100
weights	uniform, distance

Fonte: Elaborado pelo autor

O parâmetro *n_neighbors* define a quantidade de vizinhos que o modelo deve analisar para classificar um dado desconhecido. O parâmetro *weights* determina à medida que o algoritmo utilizará para calcular a distância entre os vizinhos. Os valores encontrados para os hiperparâmetros testados estão detalhados na Tabela 3.

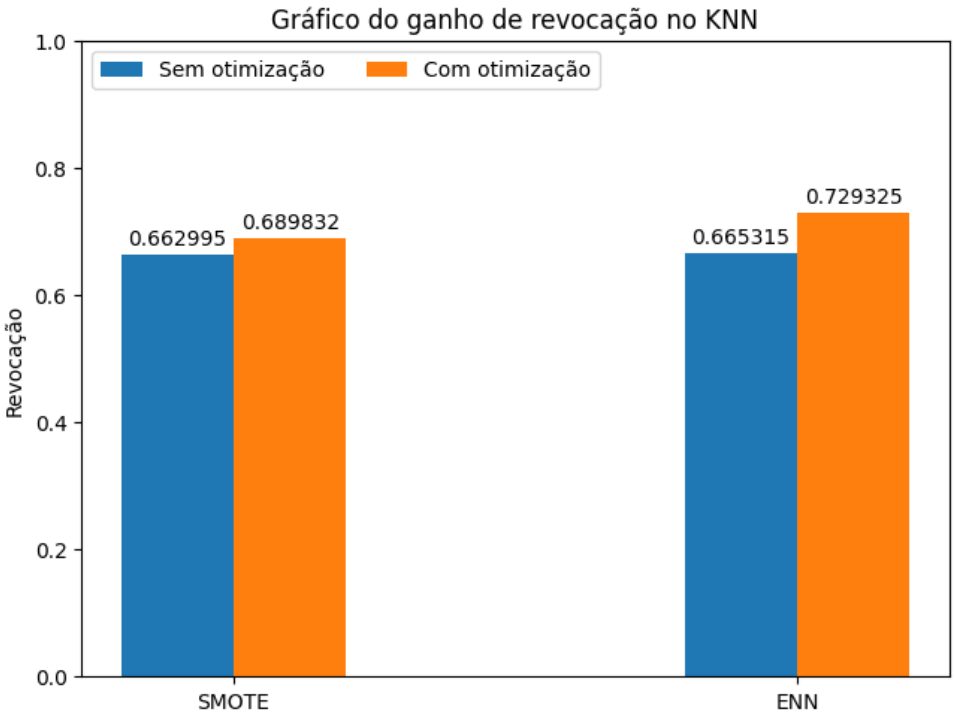
Tabela 3 - Parâmetros resultantes da pesquisa em grade para o KNN

Parâmetro	**SMOTE**	**ENN**
n_neighbors	50	5
weights	uniform	distance

Fonte: Elaborado pelo autor

Na terceira etapa, o algoritmo foi submetido novamente à validação cruzada, com os hiperparâmetros otimizados. Após a otimização dos hiperparâmetros, o algoritmo apresentou um ganho de revocação, conforme ilustrado na Figura 20. Para o conjunto onde foi aplicado o SMOTE, o ganho foi pequeno, cerca de **2,69%**. Já para o conjunto onde foi aplicado o ENN, o ganho foi mais expressivo, aproximadamente **6,4%**.

Figura 20 - Gráfico do ganho de revocação no KNN



Fonte: Elaborado pelo autor

5.2 Random Forest

O segundo algoritmo submetido aos experimentos foi o *Random Forest*. Os resultados obtidos na primeira etapa dos testes estão apresentados na Tabela 4.

Tabela 4 - Resultados obtidos com o *Random Forest* antes da otimização de hiperparâmetros

Métrica	**SMOTE**	**ENN**
Acurácia	0,932435	0,909837
Revocação	0,719926	0,754817
Precisão	0,958102	0,815955
F1-Score	0,822080	0,784192

Fonte: Elaborado pelo autor

Observando os resultados do algoritmo *Random Forest* na Tabela 4 e comparando-os com os resultados do algoritmo KNN na Tabela 1, foi notado um ganho expressivo em todas as métricas utilizadas, indicando a superioridade do *Random Forest* em relação ao KNN para lidar com este problema. O algoritmo obteve uma boa

acurácia em ambos os conjuntos de dados, destacando-se o conjunto onde foi aplicado SMOTE, que apresentou bons resultados de precisão e F1-Score. Isso indica que o algoritmo teve uma boa taxa de acerto entre os verdadeiros positivos (VP) e possui um bom equilíbrio entre precisão e revocação, conforme indicado pela métrica F1-Score.

Na segunda etapa, os parâmetros *n_estimators*, *criterion* e *max_depth* foram submetidos à pesquisa em grade. Os valores utilizados estão detalhados na Tabela 5.

Tabela 5 - Parâmetros utilizados na pesquisa em grade para o *Random Forest*

Parâmetro	**Valores**
n_estimators	100, 200, 400, 800
criterion	gini, entropy

Fonte: Elaborado pelo autor

O parâmetro *n_estimators* indica a quantidade de árvores que o algoritmo utilizará, e o parâmetro *criterion* determina qual será o critério de qualidade adotado pelo algoritmo durante as divisões dos nós. Os resultados obtidos na pesquisa em grade para o algoritmo *Random Forest* estão apresentados na Tabela 6.

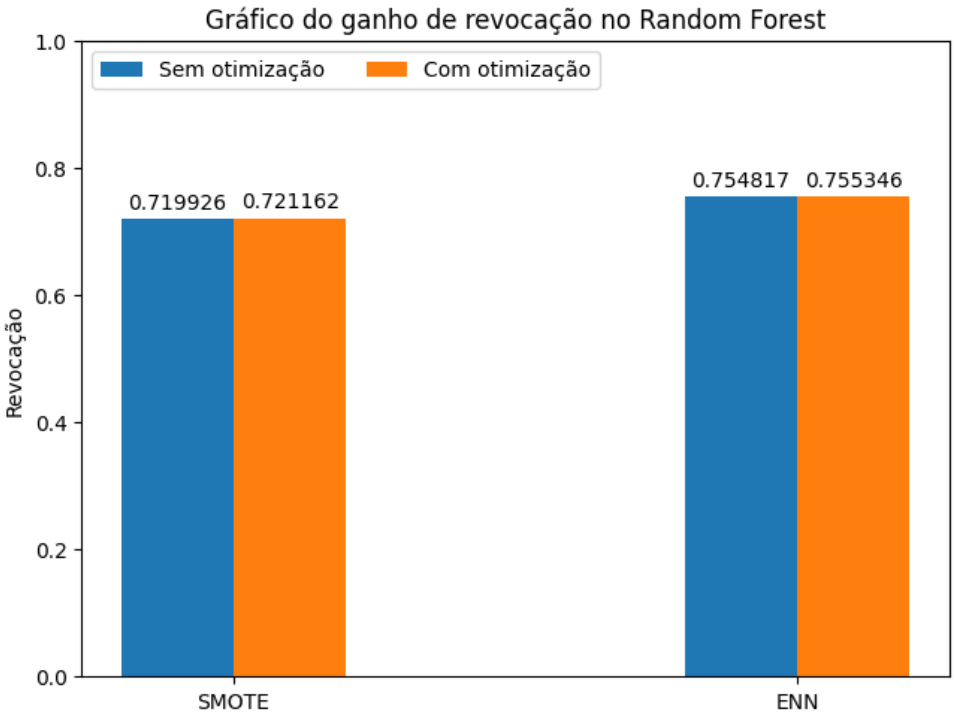
Tabela 6 - Parâmetros resultantes da pesquisa em grade para o *Random Forest*

Parâmetro	**SMOTE**	**ENN**
n_estimators	100	400
criterion	entropy	gini

Fonte: Elaborado pelo autor

Por fim, na terceira etapa, o algoritmo *Random Forest* foi novamente submetido à validação cruzada, utilizando os hiperparâmetros ideais resultantes da pesquisa em grade. Após a aplicação da validação cruzada, foi identificado um ganho sutil de revocação, conforme o gráfico da Figura 21. Para o conjunto onde foi aplicado o SMOTE, o ganho de revocação foi de aproximadamente **0,12%**, e para o conjunto onde foi aplicado o ENN, o ganho foi de cerca de **0,05%**.

Figura 21 - Gráfico do ganho de revocação no *Random Forest*



Fonte: Elaborado pelo autor

5.3 XGBoost

Na primeira etapa, o algoritmo XGBoost foi submetido à validação cruzada. Os resultados obtidos estão apresentados na Tabela 7.

Tabela 7 - Resultados obtidos com o XGBoost antes da otimização de hiperparâmetros

Métrica	**SMOTE**	**ENN**
Acurácia	0,933970	0,906000
Revocação	0,739235	0,801289
Precisão	0,944517	0,773708
F1-Score	0,829277	0,787228

Fonte: Elaborado pelo autor

Ao avaliar os resultados do algoritmo XGBoost, observaram-se valores próximos aos obtidos pelo *Random Forest*, conforme a Tabela 4, com a diferença mais significativa na métrica de revocação. O algoritmo apresentou um bom desempenho de revocação no conjunto de dados onde foi aplicado o ENN, alcançando cerca de

80,13%. Nesse cenário, o desempenho foi superior ao do *Random Forest*, com aproximadamente **4,65%** a mais de revocação. Na segunda etapa, os parâmetros *max_depth*, *subsample* e *num_parallel_tree* foram utilizados na pesquisa em grade. Os resultados obtidos estão apresentados na Tabela 8.

Tabela 8 - Parâmetros utilizados na pesquisa em grade para o XGBoost

Parâmetro	**Valores**
max_depth	3, 6
subsample	0, 0,5, 1
num_parallel_tree	1, 2, 3

Fonte: Elaborado pelo autor

O parâmetro *max_depth* especifica a profundidade máxima das árvores geradas pelo algoritmo, o parâmetro *subsample* determina a proporção de subamostragem dos dados de treinamento, e o parâmetro *num_parallel_tree* define a quantidade de árvores paralelas que o algoritmo utilizará. Os parâmetros ideais obtidos após a pesquisa em grade estão apresentados na Tabela 9.

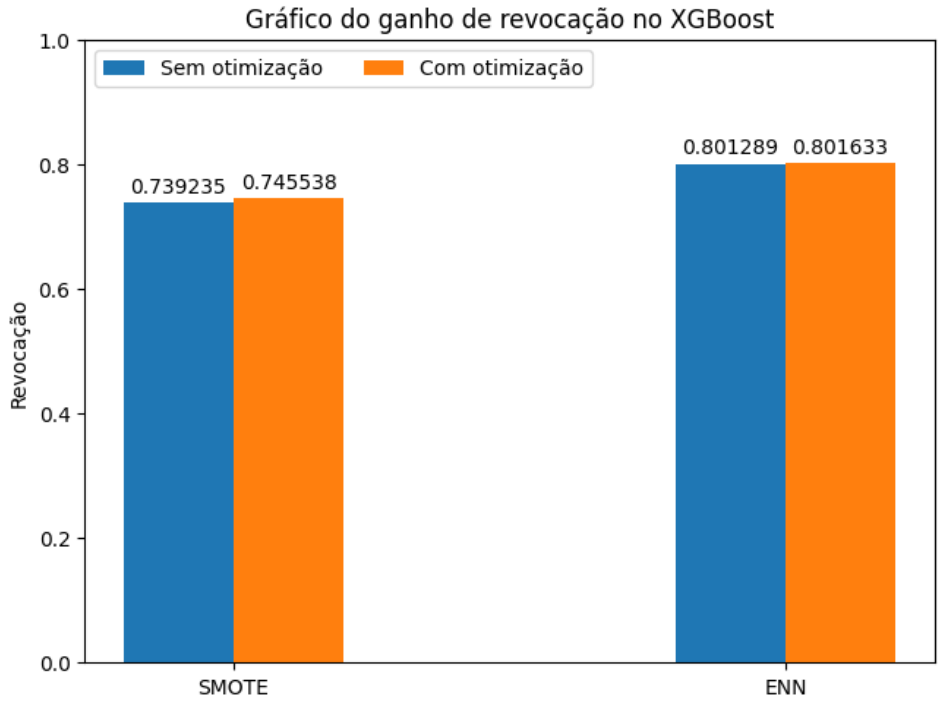
Tabela 9 - Parâmetros resultantes da pesquisa em grade para o XGBoost

Parâmetro	**SMOTE**	**ENN**
max_depth	6	6
subsample	0.5	1
num_parallel_tree	1	2

Fonte: Elaborado pelo autor

Na terceira etapa, o algoritmo XGBoost foi submetido à validação cruzada com os hiperparâmetros resultantes da etapa anterior. Assim como no algoritmo *Random Forest*, os ganhos de revocação após a otimização foram sutis, conforme o gráfico da Figura 22. O algoritmo apresentou um ganho de revocação de cerca de **0,63%** para o conjunto de dados com SMOTE e **0,03%** para o conjunto de dados com ENN.

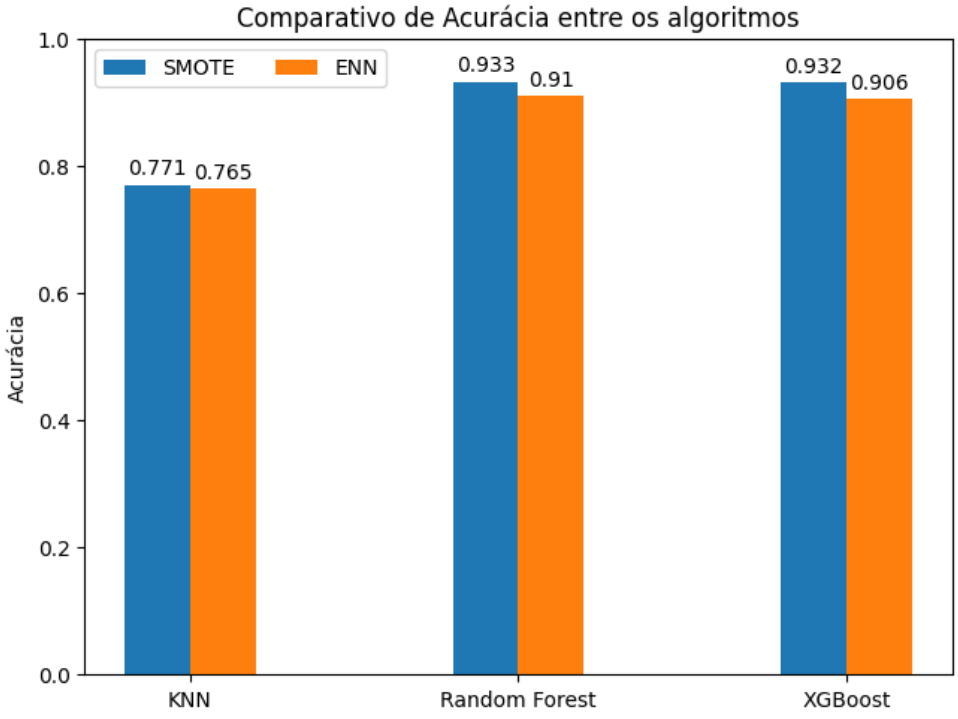
Figura 22 - Gráfico do ganho de revocação no XGBoost



Fonte: Elaborado pelo autor

5.4 Comparativo de resultados

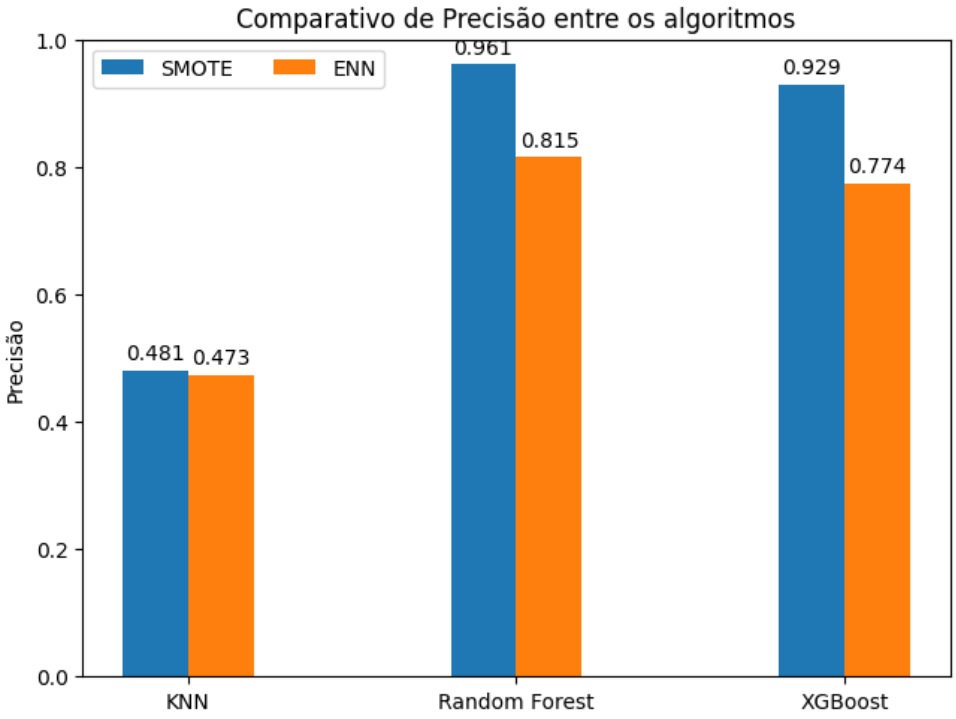
Figura 23 - Gráfico de acurácia entre os algoritmos



Fonte: Elaborado pelo autor

A Figura 23 mostra um comparativo entre os três algoritmos em termos de acurácia. Observa-se que os algoritmos baseados em árvores, como *Random Forest* e XGBoost obtiveram os maiores índices de acurácia, com valores superiores a **90%**, especialmente ao utilizar o conjunto de dados em que foi aplicado SMOTE. Em contraste, o KNN apresentou índices de acurácia inferiores, sendo aproximadamente **77,1%** para o conjunto com SMOTE e **76,5%** para o conjunto com ENN.

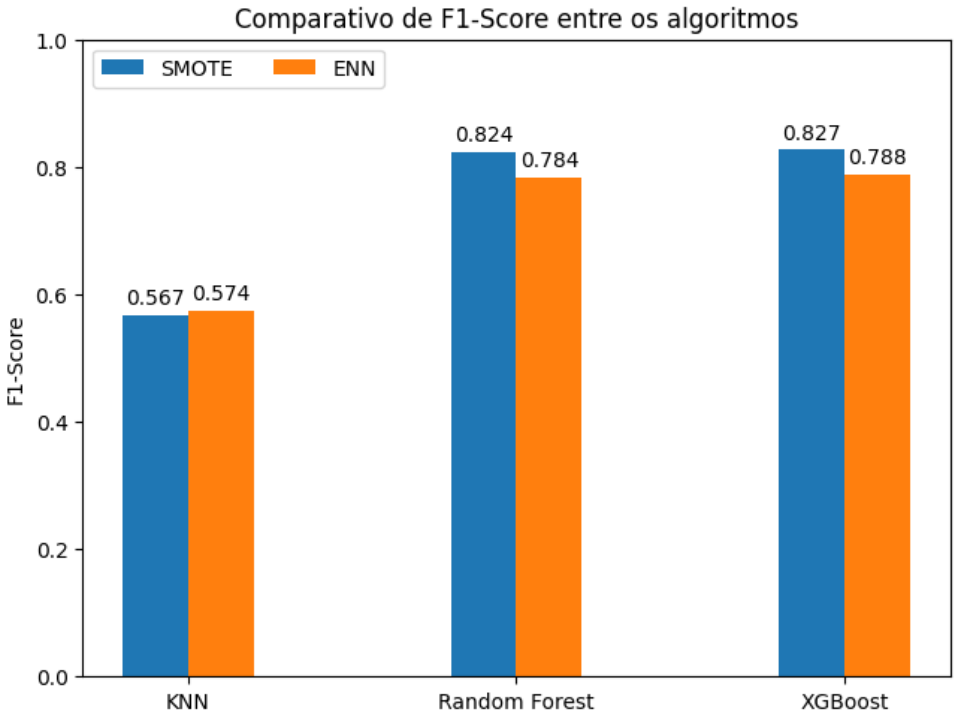
Figura 24 - Gráfico de precisão entre os algoritmos



Fonte: Elaborado pelo autor

A Figura 24 apresenta um comparativo de desempenho com relação à métrica de precisão. Novamente, os algoritmos *Random Forest* e XGBoost se destacaram com relação ao KNN, com valores expressivos de precisão, superiores a **90%** no conjunto de dados onde foi aplicado SMOTE. Neste cenário, o KNN obteve resultados consideravelmente inferiores para ambos os conjuntos de dados, com índices de precisão abaixo de **50%**.

Figura 25 - Gráfico de F1-Score entre os algoritmos



Fonte: Elaborado pelo autor

A Figura 25 apresenta um comparativo do F1-Score entre os algoritmos. Assim como nas métricas anteriores, os algoritmos *Random Forest* e XGBoost se destacaram em relação ao KNN. Ambos obtiveram valores de F1-Score superiores a **80%** para o conjunto com SMOTE, o que indica um bom equilíbrio entre precisão e revocação. Em contraste, o KNN apresentou índices de F1-Score em torno de **56,7%** para o conjunto com SMOTE e **57,4%** para o conjunto com ENN.

5.5 Teste de predição

Para testar as previsões do algoritmo XGBoost, foi extraída da base de dados uma amostra aleatória contendo 10 exemplos, composta por clientes inadimplentes (1) e adimplentes (0), conforme apresentado na Figura 26.

Figura 26 - Amostra aleatória extraída para o teste de previsão

person_age	person_income	person_emp_length	loan_grade	loan_amnt	loan_int_rate	loan_status
27.0	47900	1.0	2.0	7500	13.47	0
34.0	73000	0.0	1.0	10000	11.11	0
32.0	54000	16.0	0.0	10000	7.74	0
23.0	38000	3.0	0.0	19750	7.51	1
24.0	42996	5.0	0.0	5000	7.14	0
23.0	45996	7.0	2.0	10000	13.16	0
29.0	50000	4.0	1.0	21850	12.42	0
22.0	40000	4.0	1.0	14125	11.49	1
21.0	18000	1.0	0.0	3200	8.49	1
21.0	14400	0.0	0.0	2000	5.79	0

Fonte: Elaborado pelo Autor

Em seguida, as amostras foram submetidas ao algoritmo, que foi treinado com as mesmas configurações que apresentaram o melhor desempenho nos experimentos, utilizando o conjunto de dados balanceado com ENN e os hiperparâmetros otimizados. As previsões realizadas pelo algoritmo estão apresentadas na Figura 27.

Figura 27 - Previsões realizadas pelo XGBoost para a amostra aleatória

```
Testando predição para a amostra 1:
valor real: 0
valor previsto: 0
Testando predição para a amostra 2:
valor real: 0
valor previsto: 0
Testando predição para a amostra 3:
valor real: 0
valor previsto: 0
Testando predição para a amostra 4:
valor real: 1
valor previsto: 1
Testando predição para a amostra 5:
valor real: 0
valor previsto: 0
Testando predição para a amostra 6:
valor real: 0
valor previsto: 0
Testando predição para a amostra 7:
valor real: 0
valor previsto: 0
Testando predição para a amostra 8:
valor real: 1
valor previsto: 1
Testando predição para a amostra 9:
valor real: 1
valor previsto: 1
Testando predição para a amostra 10:
valor real: 0
valor previsto: 0
```

Fonte: Elaborado pelo Autor

Conforme a Figura 27, é possível observar que, para a amostra extraída, o algoritmo apresentou uma taxa de acerto de **100%**, o que, neste exemplo, representa um excelente resultado.

6 CONCLUSÃO E TRABALHOS FUTUROS

Em função do aumento da complexidade no processo de avaliação de risco de crédito e da vasta quantidade de dados disponíveis para empresas do setor financeiro, especialmente aquelas atuantes no segmento de crédito, este trabalho apresentou como a IA, por meio do *Machine Learning*, pode contribuir para otimizar e aumentar a confiabilidade no processo de avaliação do risco de crédito. Para isso, foram apresentados três modelos de algoritmo de *Machine Learning* voltados à classificação, aplicáveis para este problema. Os algoritmos apresentados incluem o KNN, *Random Forest* e XGBoost.

Este trabalho abordou conceitos relacionados à área de *Machine Learning*, incluindo métodos de aprendizado supervisionado e não supervisionado. Também são detalhados os algoritmos utilizados, juntamente com a sua fundamentação teórica. Além disso, discute-se o uso de técnicas fundamentais na análise de dados para aprimorar o desempenho dos algoritmos, como *feature engineering*, balanceamento de classes e otimização de hiperparâmetros. Também são apresentadas as métricas utilizadas para avaliar o desempenho dos modelos, sendo elas: acurácia, revocação, precisão e F1-Score. A métrica de revocação é destacada como a mais relevante para o problema tratado neste trabalho, uma vez que representa a taxa de acerto entre os verdadeiros positivos (VP), isto é, a capacidade do algoritmo em identificar corretamente os clientes potencialmente inadimplentes.

Como metodologia, este trabalho propôs submeter cada algoritmo a diversos cenários de teste. Para isso, foi utilizado o ambiente Google Colaboratory, juntamente com a linguagem Python, que é amplamente utilizada em projetos de *Machine Learning*, fornecendo bibliotecas que auxiliam no desenvolvimento.

Cada algoritmo foi submetido à validação cruzada e à otimização de hiperparâmetros para extrair o melhor desempenho. Comparativamente, o KNN obteve resultados de revocação inferiores aos demais algoritmos. O *Random Forest* apresentou resultados satisfatórios com relação à métrica de acurácia e resultados superiores ao KNN com relação à métrica de revocação.

Analisando os resultados obtidos, conclui-se que os algoritmos *Random Forest* e XGBoost possuem um bom desempenho para previsão do risco de crédito. No

entanto, o algoritmo XGBoost destacou-se na métrica de revocação, que possuí relevância considerável para o problema tratado. Portanto, pode-se concluir que o algoritmo XGBoost é o mais indicado para este caso.

Com relação às técnicas de balanceamento utilizadas com o XGBoost, a técnica ENN apresentou o melhor resultado em revocação, com aproximadamente **80,2%**. Nas demais métricas, este conjunto obteve **90,6%** acurácia, **77,4%** de precisão e **78,8%** de F1-Score. Dado que a revocação é uma métrica crucial para este problema, conclui-se que a técnica ENN é a mais adequada quando aplicada ao algoritmo XGBoost.

Os resultados obtidos nos experimentos realizados estão em consonância com os alcançados por Silva (2022), cujos experimentos demonstraram que os algoritmos *Random Forest* e *Gradient Boosting* apresentaram os melhores desempenhos, com o *Gradient Boosting* se destacando nas validações realizadas. De maneira semelhante, nos experimentos conduzidos por Lopes, Colombi e Mutz (2023), o algoritmo XGBoost também apresentou resultados superiores de revocação em comparação com os algoritmos *Random Forest* e KNN.

Para trabalhos futuros, sugere-se um estudo mais aprofundado dos algoritmos utilizados, bem como a exploração de outros algoritmos de classificação em *Machine Learning* como Redes Neurais Artificiais. Além disso, recomenda-se a utilização de um conjunto de dados com maior abrangência histórica e uma diversidade mais ampla de *features*, as quais devem ser exploradas para identificar relações e tendências relevantes. Propõe-se também a investigação de outras técnicas de tratamento de dados como PCA, e balanceamento de classes como a *Adaptive Synthetic Sampling Approach for Imbalanced Learning* (ADASYN). Por fim, sugere-se a exploração de mais combinações de hiperparâmetros, visando otimizar os resultados obtidos pelos algoritmos.

7 REFERÊNCIAS

ANDRADE, E. **Matplotlib: Dando vida aos dados em Python**. Disponível em: <https://www.dio.me/articles/matplotlib-dando-vida-aos-dados-em-python>. Acesso em: 23 ago. 2024.

AWARI. **XGBoost Python: Aprenda a utilizar essa poderosa biblioteca de machine learning**. Disponível em: <https://awari.com.br/xgboost-python-aprenda-a-utilizar-essa-poderosa-biblioteca-de-machine-learning/>. Acesso em: 23 ago. 2024.

AWS. **O que é ajuste de hiperparâmetros?** Disponível em: <https://aws.amazon.com/pt/what-is/hyperparameter-tuning/>. Acesso em: 24 ago. 2024.

BROWNLEE, J. **A Gentle Introduction to k-fold Cross-Validation**. Disponível em: <https://machinelearningmastery.com/k-fold-cross-validation/>. Acesso em: 14 set. 2024.

CATUNDA, H. **O que é Python e por que aprender? Guia para Iniciantes**. Disponível em: <https://www.hashtagtreinamentos.com/o-que-e-python>. Acesso em: 23 ago. 2024a.

CATUNDA, H. **O que é o Kaggle? Entenda e saiba como começar a usá-lo**. Disponível em: <https://www.hashtagtreinamentos.com/kaggle>. Acesso em: 21 ago. 2024b.

CLEARSALE. **Modelo Preditivo: o que é, para que serve e como aplicá-lo?** Disponível em: <https://blogbr.clear.sale/modelo-preditivo-saiba-como-aplica-lo>. Acesso em: 30 abr. 2024.

COSTA, M.; PYRES, L. **Direto ao ponto: O que é Machine Learning com exemplos reais**. Disponível em: <https://www.alura.com.br/artigos/machine-learning>. Acesso em: 27 abr. 2024.

FONTELLES, M. J. et al. Metodologia da Pesquisa Científica: Diretrizes para a Elaboração de um Protocolo de Pesquisa. **Universidade da Amazônia**, 2009.

FREITAS, N. C. A. DE. **Inteligência de negócios e análise de dados**. 1. ed. São Paulo: Editora Senac, 2023.

GIL, A. C. **Como elaborar projetos de pesquisa**. 6. ed. São Paulo: Atlas, 2017.

GOMES, P. C. T. **XGBoost: conheça este algoritmo de machine learning**. Disponível em: <https://www.datageeks.com.br/xgboost/>. Acesso em: 8 jul. 2024.

GOOGLE CLOUD. **O que é machine learning (ML)?** Disponível em: <https://cloud.google.com/learn/what-is-machine-learning?hl=pt-br#section-2>. Acesso em: 2 jul. 2024.

GUNAY, D. **Random Forest**. Disponível em: <https://medium.com/@denizgunay/random-forest-af5bde5d7e1e>. Acesso em: 7 jul. 2024.

HABBEMA, H. **Introdução ao Scikit-Learn**. Disponível em: <https://medium.com/@habbema/introdu%C3%A7%C3%A3o-ao-scikit-learn-f00b7201dbf7>. Acesso em: 23 ago. 2024.

IBM. **O que é inteligência artificial?** Disponível em: <https://www.ibm.com/br-pt/topics/artificial-intelligence>. Acesso em: 2 jul. 2024a.

IBM. **O que é aprendizado supervisionado?** Disponível em: <https://www.ibm.com/br-pt/topics/supervised-learning>. Acesso em: 2 jul. 2024b.

IBM. **What is the k-nearest neighbors (KNN) algorithm?** Disponível em: <https://www.ibm.com/topics/knn>. Acesso em: 4 jul. 2024c.

IBM. **What is random forest?** Disponível em: <https://www.ibm.com/topics/random-forest>. Acesso em: 7 jul. 2024d.

JOSÉ, I. **KNN (K-Nearest Neighbors) Como funciona?** Disponível em: <https://medium.com/brasil-ai/knn-k-nearest-neighbors-1-e140c82e9c4e>. Acesso em: 3 jul. 2024.

JUNIOR, G. B. V. et al. Métricas Utilizadas Para Avaliar A Eficiência De Classificadores Em Algoritmos Inteligentes. **Centro de Pesquisas Avançadas em Qualidade de Vida**, v. 14, n. 2, 2022.

JUNIOR, L. D. C.; KLEFENS, P. C. D. O. **Análise de Crédito, Cobrança e Risco**. Londrina: Editora e Distribuidora Educacional S.A., 2015.

KALINOWSKI, M. et al. **Engenharia de Software para Ciência de Dados: Um guia de boas práticas com ênfase na construção de sistemas de Machine Learning em Python**. São Paulo: Casa do Código, 2023.

KUNUMI. **Métricas de Avaliação em Machine Learning: Classificação**. Disponível em: <https://medium.com/kunumi/m%C3%A9tricas-de-avalia%C3%A7%C3%A3o-em-machine-learning-classifica%C3%A7%C3%A3o-49340dcdb198>. Acesso em: 10 jul. 2024.

LAGO, B. **Gráficos usando Seaborn**. Disponível em: <https://medium.com/@bernardolago/gr%C3%A1ficos-usando-seaborn-61f7d23481cf>. Acesso em: 23 ago. 2024.

LOPES, R. D. S.; COLOMBI, L. R.; MUTZ, F. Comparação de Algoritmos de Aprendizado de Máquina para Predição de Pontuação de Crédito. **XIV Computer on the Beach**, p. 424–431, 30 mar. 2023.

MARTIN, D. **How to do cross-validation when upsampling data**. Disponível em: <https://kiwidamien.github.io/how-to-do-cross-validation-when-upsampling-data.html>. Acesso em: 14 set. 2024.

MITCHELL, R. **Gradient Boosting, Decision Trees and XGBoost with CUDA**. Disponível em: <https://developer.nvidia.com/blog/gradient-boosting-decision-trees-xgboost-cuda/>. Acesso em: 8 jul. 2024.

MONUTTI, D. **Pandas Python – O que é, para que Serve e Como Instalar**. Disponível em: <https://www.hashtagtreinamentos.com/pandas-python>. Acesso em: 23 ago. 2024.

NVIDIA. **XGBoost**. Disponível em: <https://www.nvidia.com/en-us/glossary/xgboost/>. Acesso em: 8 jul. 2024.

ORACLE. **O que é Machine Learning?** Disponível em: <https://www.oracle.com/br/artificial-intelligence/machine-learning/what-is-machine-learning/>. Acesso em: 2 jul. 2024.

QIN, C. et al. XGBoost Optimized by Adaptive Particle Swarm Optimization for Credit Scoring. **Hindawi Mathematical Problems in Engineering**, v. 2021, 2021.

RABELLO, E. B. **Cross Validation: Avaliando seu modelo de Machine Learning**. Disponível em: <https://medium.com/@edubrazrabello/cross-validation-avaliando-seu-modelo-de-machine-learning-1fb70df15b78>. Acesso em: 14 set. 2024.

RASCHKA, S. **Machine Learning Lecture Notes**. Madison: [s.n.]. Disponível em: <https://github.com/rasbt/stat479-machine-learning-fs18/blob/master/02_knn/02_knn_notes.pdf>. Acesso em: 19 nov. 2024.

RAUTENBERG, S.; CARMO, P. R. V. DO. Big Data e Ciência de Dados: Complementariedade Conceitual no Processo de Tomada de Decisão. **Brazilian Journal of Information Studies: Research Trends**, v. 13, n. 1, p. 56–67, 2019.

SANTIAGO, D. **Aprenda a balancear seus dados com Undersampling e Oversampling em Python**. Disponível em: <https://medium.com/@daniele.santiago/aprenda-a-balancear-seus-dados-com-undersampling-e-oversampling-em-python-6fd87095d717>. Acesso em: 23 ago. 2024.

SANTOS, J. O. DOS. **Análise de crédito - empresas, pessoas físicas e agronegócio: guia prático com capítulo dedicado ao uso da inteligência artificial**. 6. ed. São Paulo: Editora Dialética, 2024.

SANTOS, T. G. **Google Colab: o que é, tutorial de como usar e criar códigos**. Disponível em: <https://www.alura.com.br/artigos/google-colab-o-que-e-e-como-usar>. Acesso em: 21 ago. 2024.

SCHONLAU, M.; ZOU, R. Y. The random forest algorithm for statistical learning. **The Stata Journal**, v. 20, n. 1, p. 3–29, 2020.

SEBBEN, R. J. **Análise de Crédito e Cobrança**. 1. ed. São Paulo: Novatec Editora Ltda, 2020.

SILVA, J. S. **Gerenciamento Integrado de Riscos: Modelos de Predição de Risco de Crédito em Machine Learning para a Identificação de Ativos Problemáticos em uma Instituição Financeira - Segmento Habitacional PF**. Brasília: Universidade de Brasília, 2022.

SOUSA, M. H. S. DE. **Regressão Logística e Matriz de Confusão com Sklearn**. Disponível em: <https://www.linkedin.com/pulse/regress%C3%A3o-log%C3%ADstica-e-matriz-de-confus%C3%A3o-com-sklearn-silva-de-sousa/>. Acesso em: 10 jul. 2024.

SOUZA, T. C. L. DE. **Estudo de Algoritmos de Machine Learning para predição de fraudes em cartões de crédito**. Trabalho de Conclusão de Curso—Goiânia: Pontifícia Universidade Católica de Goiás, 2021.

WAZLAWICK, R. S. **Metodologia de pesquisa para ciência da computação**. 2. ed. Rio de Janeiro: Elsevier, 2024.

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
GABINETE DO REITOR

Av. Universitária, 1069 ● Setor Universitário
Caixa Postal 86 ● CEP 74605-010
Goiânia ● Goiás ● Brasil
Fone: (62) 3946.1000
www.pucgoias.edu.br ● reitoria@pucgoias.edu.br

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O(A) estudante GUILHERME FREIRE MAGALHÃES do Curso de CIÊNCIA DA COMPUTAÇÃO, matrícula 2020.1.0028.0126-4, telefone: (62)985203901 e-mail guilhermefreire841@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado PREDIÇÃO DO RISCO DE CRÉDITO EM INSTITUIÇÕES FINANCEIRAS UTILIZANDO MACHINE LEARNING, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto (PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 10 de DEZEMBRO de 2024.

Assinatura do(s) autor(es): Guilherme Freire Magalhães

Nome completo do autor: GUILHERME FREIRE MAGALHÃES

Assinatura do professor-orientador:

Nome completo do professor-orientador: ANÍBAL SANTOS JUKEMURA