Detecção de fraudes em transações de cartão de crédito: análise de modelos de machine learning

Mariana Caetano Vidal [1]*; Anna Carolina Martins[2]

[1] Bacharel em Economia, cientista de dados. Avenida Mario Lopes Leão, 916, Santo Amaro, 04754-010, São Paulo, SP, Brasil
[2] PECEGE. Doutora em Economia, professora orientadora. Rua Alexandre Herculano 120, 13418-445, Piracicaba, SP, Brasil.
*autor correspondente: maricvidal@hotmail.com

Detecção de fraudes em transações de cartão de crédito: análise de modelos de machine learning

Resumo

O crescimento do uso de cartões de crédito aumentou a incidência de fraudes financeiras, tornando essencial o desenvolvimento de soluções eficazes para sua detecção. Este trabalho teve como objetivo desenvolver e avaliar modelos de aprendizado de máquina para identificar transações fraudulentas com maior eficiência e precisão. Para isso, foram utilizados os algoritmos K-Nearest Neighbors (KNN), Random Forest e Gradient Boosting, treinados e comparados por meio das métricas precisão, recall e F-score. A metodologia envolveu a seleção das variáveis preditoras mais relevantes, o balanceamento da base de dados para lidar com a desproporção entre transações fraudulentas e não fraudulentas, além da análise do impacto dessas estratégias no desempenho dos modelos. Os resultados demonstraram que os algoritmos Random Forest e KNN apresentaram um melhor equilíbrio entre precisão e recall, mostrando-se mais adequados para a detecção de fraudes. Conclui-se que abordagens baseadas em aprendizado de máquina podem aprimorar a segurança dos sistemas financeiros, reduzindo perdas e aumentando a eficiência na identificação de transações suspeitas.
Palavras-chave: segurança digital; análise preditiva; anomalias; inteligência artificial; risco financeiro.

Introdução

O cartão de crédito é um instrumento financeiro que permite aos consumidores realizar transações de maneira prática, utilizando um limite de crédito pré-aprovado pela instituição emissora, com a possibilidade de adiar o pagamento. De acordo com Evans e Schmalensee (2005), o funcionamento do cartão de crédito é baseado em um sistema de crédito rotativo, oferecem flexibilidade no pagamento e acesso facilitado ao consumo. Segundo dados do Nilson Report (2023), o uso de cartões de crédito tem crescido significativamente ao longo das décadas, especialmente devido à conveniência e à ampla aceitação em redes comerciais globais. Em 2022, por exemplo, o volume global de transações com cartões de crédito foi um dos mais altos da história, consolidando sua importância no sistema financeiro.

Além de facilitar as transações comerciais, os cartões de crédito desempenham um papel importante na economia global, promovendo a inclusão financeira e fomentando o acesso ao consumo em diversas camadas sociais (Evans e Schmalensee, 2005; Nilson Report, 2023).

Atualmente, os cartões de crédito não apenas representam um meio de pagamento, mas também constituem uma ferramenta essencial para a gestão financeira pessoal e empresarial. No âmbito pessoal, permitem maior controle dos gastos, facilitam o planejamento financeiro e oferecem benefícios como programas de recompensas. Para empresas, auxiliam na gestão do fluxo de caixa, na separação de despesas corporativas e na otimização de pagamentos recorrentes. No entanto, o aumento significativo no uso dessa forma de

pagamento também trouxe desafios, como o crescimento exponencial dos casos de fraudes, exigindo estratégias cada vez mais sofisticadas para detecção e prevenção desses eventos.

Fraudes com cartões de crédito englobam uma ampla gama de atividades ilícitas, incluindo roubo de informações de cartão, clonagem, compras não autorizadas e uso de dados comprometidos em transações digitais. Segundo Nilson Report (2023), as perdas globais causadas por fraudes com cartões de crédito alcançaram cerca de US$ 35 bilhões em 2022, representando um desafio significativo para instituições financeiras e consumidores. Estudos apontam que métodos de fraude têm se tornado mais sofisticados com o avanço das tecnologias digitais, explorando vulnerabilidades em sistemas de pagamento e plataformas de e-commerce (Anderson et al., 2020). Entre os tipos mais comuns de fraude estão o "phishing", onde criminosos enganam usuários para obter dados sensíveis, e ataques baseados em inteligência artificial para simular comportamentos legítimos (Kardefelt-Winther, 2021).

Diante desse cenário, a detecção de fraudes em tempo hábil é essencial para reduzir prejuízos financeiros e proteger a integridade dos sistemas de pagamento. Soluções baseadas em machine learning têm se destacado como uma abordagem eficiente, permitindo a identificação de padrões anômalos em grandes volumes de dados transacionais. Essas técnicas oferecem vantagens significativas em relação a métodos tradicionais, como regras estáticas ou análises manuais, por sua capacidade de adaptação a novas formas de fraude e pela precisão na detecção (Zhang et al., 2019).

Nesse contexto, o presente trabalho teve como objetivo identificar o modelo mais adequado para a detecção de fraudes, avaliando diferentes técnicas para selecionar o modelo mais eficaz. Para tanto, isto é, a detecção dos eventos de fraude, foram utilizados três métodos distintos: K-Nearest Neighbors (KNN), Random Forest e Gradient Boosting. A proposta envolve o treinamento e a comparação desses modelos, utilizando métricas de avaliação como precisão, revocação e F-score. Além disso, busca-se explorar como técnicas de manipulação de dados, como balanceamento de classes e engenharia de atributos, podem influenciar o desempenho dos modelos.

Material e Métodos

A base de dados utilizada neste estudo foi coletada pela Universidade Livre de Bruxelas (ULB) e contém transações com cartão de crédito realizadas por usuários europeus em setembro de 2013. O conjunto de dados é composto por 284.807 transações, das quais apenas 492 são fraudes, representando apenas 0,172% do total, o que torna o problema altamente desbalanceado.

A base de dados é composta por 31 colunas, das quais 28 são variáveis numéricas transformadas por Análise de Componentes Principais (PCA). Essa transformação foi aplicada para garantir a anonimização das informações, preservando a confidencialidade dos dados. As variáveis Time (tempo, em segundos, desde a primeira transação registrada) e Amount (valor da transação) não passaram pelo processo de PCA, mantendo seus valores originais. A variável Class representa a variável alvo do modelo, sendo que o valor 1 indica uma transação fraudulenta e 0 indica uma transação legítima. A Tabela 1 apresenta a descrição dessas variáveis.

Tabela 1. Variáveis utilizadas neste estudo

Variável	Tipo	Descrição	Periodicidade
Class	Categórica	Indica se a transação é fraudulenta (1) ou legítima (0).	Transação a transação
Amount	Numérica	Valor monetário da transação.	Transação a transação
Time	Numérica	Tempo decorrido desde a primeira transação registrada no conjunto de dados.	Transação a transação
V1 a V28	Numérica	Variáveis obtidas a partir de uma transformação PCA (Principal Component Analysis) para preservar a privacidade dos dados.	Transação a transação

Fonte: Elaboração própria a partir dos dados disponibilizados por Machine Learning Group – Universidade Livre de Bruxelas (ULB, 2025)

A Análise de Componentes Principais (PCA) é uma técnica estatística utilizada para a redução de dimensionalidade, que transforma um conjunto de variáveis correlacionadas em um novo conjunto de variáveis não correlacionadas, chamadas de componentes principais (Jolliffe & Cadima, 2016). Essa transformação busca reter a maior parte da variabilidade dos dados originais enquanto reduz a complexidade do modelo, facilitando a análise e melhorando o desempenho de algoritmos de aprendizado de máquina (Abdi & Williams, 2010). No caso da base de dados utilizada neste estudo, a aplicação do PCA permitiu a anonimização das informações sensíveis das transações, garantindo a privacidade dos usuários sem comprometer a estrutura dos dados necessária para a detecção de fraudes.

As variáveis Time e Amount, que não foram transformadas por Análise de Componentes Principais (PCA), apresentaram escalas distintas e elevada dispersão, conforme evidenciado na análise descritiva dos dados. Como algoritmos de aprendizado de

máquina, especialmente aqueles baseados em distância, como o K-Nearest Neighbors (KNN), são sensíveis à escala das variáveis, tornou-se necessário aplicar uma técnica de normalização a essas duas colunas (Han, Kamber e Pei, 2011; Géron, 2019).

Antes de aplicar a técnica de escalonamento, foi realizada uma análise exploratória utilizando gráficos de caixa (boxplots), com o objetivo de verificar a presença de outliers nas distribuições de Time e Amount. Os gráficos foram gerados com a biblioteca Seaborn v. 0.12.2 e Matplotlib v. 3.7.1, ambas implementadas em Python. Essa abordagem permite identificar a existência de valores extremos, sendo uma das formas mais tradicionais e eficazes de detecção visual de outliers em dados univariados (Tukey, 1977; Aggarwal, 2017). O boxplot destaca a mediana, os quartis e os pontos discrepantes fora do intervalo interquartílico, sendo especialmente útil em conjuntos de dados financeiros, que comumente apresentam variabilidade elevada e valores extremos (Han, Kamber e Pei, 2011).

Dentre os métodos disponíveis, destacam-se o StandardScaler e o RobustScaler, ambos pertencentes à biblioteca Scikit-learn. O primeiro realiza a padronização dos dados com base na média e no desvio padrão, sendo mais indicado quando os dados seguem uma distribuição aproximadamente normal e não contêm outliers significativos. Já o segundo utiliza a mediana e a amplitude interquartílica (IQR), sendo, portanto, mais adequado em contextos onde há presença de valores extremos, pois reduz a influência desses pontos na transformação dos dados (Pedregosa et al., 2011).

Os dados apresentados na Tabela 1, disponibilizados pela ULB (2025), estão desbalanceados, ou seja, a distribuição entre as classes não é uniforme, havendo uma quantidade significativamente maior de transações legítimas em comparação às transações fraudulentas. Esse desbalanceamento pode comprometer o desempenho dos modelos de machine learning, tornando-os tendenciosos para a classe majoritária. Para garantir maior robustez na análise dos resultados, foi necessário realizar o balanceamento das classes.

O balanceamento das classes foi realizado utilizando a técnica de Random Under-Sampling, amplamente discutida por Batista et al. (2004) e He e Garcia (2009). Essa abordagem consiste em reduzir o número de instâncias da classe majoritária até que se iguale ao número de instâncias da classe minoritária, criando um conjunto de dados balanceado (Batista et al., 2004). Para isso, foi identificado o número de transações fraudulentas (classe minoritária) e, em seguida, selecionadas aleatoriamente a mesma quantidade de instâncias da classe não fraudulenta (classe majoritária) (He; Garcia, 2009). Os dados foram embaralhados para garantir que o treinamento do modelo ocorresse de forma uniforme e livre de viés (Batista et al., 2004). Essa técnica foi utilizada para reduzir os impactos do desbalanceamento no treinamento dos modelos, uma prática recomendada em problemas de aprendizado supervisionado com classes desbalanceadas (He; Garcia, 2009).

A seleção das variáveis para o modelo de classificação foi realizada com base na análise de correlação entre as variáveis preditoras e a variável alvo. Inicialmente, foi aplicado o teste de normalidade de Shapiro-Wilk (Shapiro; Wilk, 1965) e o Kolmogorov-Smirnov (Massart et al., 1983) para verificar a distribuição dos dados. O teste de Shapiro-Wilk é amplamente utilizado para avaliar se uma amostra segue uma distribuição normal, sendo recomendado para amostras menores (inferiores a 2.000 observações). Este teste calcula uma estatística de teste W, que compara a distribuição dos dados com a distribuição normal. A hipótese nula (H_0) do teste é que os dados seguem uma distribuição normal, enquanto a hipótese alternativa (H_1) é que os dados não seguem uma distribuição normal. O valor de W é calculado com base na comparação entre a distribuição observada e a distribuição normal esperada. Para analisar os resultados, se o valor-p obtido for menor do que o nível de significância (comumente 0,05), rejeita-se a hipótese nula, indicando que os dados não seguem uma distribuição normal.

O teste de Kolmogorov-Smirnov é utilizado para comparar a distribuição empírica dos dados com uma distribuição teórica específica. Esse teste é adequado para amostras maiores e calcula a estatística D, que é a maior diferença absoluta entre a distribuição empírica acumulada dos dados e a distribuição acumulada teórica. A hipótese nula do teste é que os dados seguem a distribuição teórica especificada (normal, por exemplo), e a hipótese alternativa é que os dados não seguem essa distribuição. Assim como no teste de Shapiro-Wilk, se o valor-p for menor que 0,05, a hipótese nula é rejeitada, sugerindo que os dados não seguem a distribuição teórica. Este teste é particularmente útil quando a normalidade não pode ser assumida de forma explícita, como em grandes amostras, e permite a comparação de dados com diferentes distribuições teóricas.

Com base nos resultados desses testes, foi definida a métrica de correlação a ser utilizada. Para variáveis que apresentaram distribuição normal, foi aplicada a correlação de Pearson (Pearson, 1895), que mede a relação linear entre as variáveis, sendo amplamente utilizada quando os dados seguem uma distribuição normal (Mukaka, 2012). A Equação 1, refere-se a métrica do coeficiente de correlação de Pearson.

$$r = \frac{\sum_{i=1}^{n}(X_i-\overline{X})(Y_i-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\overline{X})^2\sum_{i=1}^{n}(Y_i-\overline{Y})^2}} \quad (1)$$

onde r é o coeficiente de correlação de Pearson, X_i e Y_i são os valores das variáveis X e Y, e $\overline{X}$ e $\overline{Y}$ são as médias das variáveis X e Y, respectivamente.

Para as variáveis que não apresentaram normalidade, utilizou-se a correlação de Spearman (Spearman, 1904), que avalia a relação monotônica entre as variáveis, sendo mais robusta para distribuições não normais e relações não lineares (Schmidt; Hunter, 1996). A Equação 2, refere-se a métrica do coeficiente de correlação de Spearman.

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2-1)} \quad (2)$$

onde ρ é o coeficiente de correlação de Spearman e d_i é a diferença entre os postos das observações nas duas variáveis, e n é o número total de observações.

Para a seleção das variáveis preditoras, foram consideradas apenas aquelas que apresentaram correlação moderada a forte com a variável alvo, ou seja, coeficientes superiores a 0.4 em valor absoluto, conforme a classificação proposta por Guilford (1956).

Após o balanceamento e a seleção de variáveis, a próxima etapa foi dividir o conjunto de dados em dois subconjuntos: treino e teste. A separação do conjunto de dados em dois subconjuntos, treino e teste, é uma etapa importante no desenvolvimento de modelos preditivos. O conjunto de treinamento é utilizado para ajustar os parâmetros do modelo, enquanto o conjunto de teste serve para avaliar o desempenho do modelo em dados novos e não vistos durante o treinamento. Esta divisão permite simular a aplicação do modelo em um ambiente real, onde ele precisa fazer previsões sobre dados desconhecidos, ajudando a avaliar sua capacidade de generalização. Sem essa separação, o modelo pode simplesmente memorizar os dados de treinamento (fenômeno conhecido como “overfitting”), resultando em um bom desempenho no treinamento, mas com baixo poder de generalização, ou seja, com baixo desempenho em dados novos (Hastie, Tibshirani & Friedman, 2009).

No contexto deste estudo, foi adotada a divisão de 70% dos dados para treinamento e 30% para teste, uma prática comum e amplamente utilizada (Kuhn & Johnson, 2013). Essa proporção representa um equilíbrio entre fornecer ao modelo dados suficientes para aprender e manter uma quantidade razoável de dados para avaliação. Com 30% dos dados reservados para teste, é possível obter uma avaliação robusta do desempenho do modelo em dados não vistos. Em algumas situações, podem ser adotadas outras proporções, como 80/20 ou 60/40, dependendo da quantidade de dados disponíveis.

O conjunto de treino é destinado ao ajuste dos parâmetros do modelo ou simplesmente para o modelo aprender, enquanto o conjunto de teste serve para validar seu desempenho ou verificar se ele generaliza. Essa abordagem assegura uma avaliação justa, identificando potenciais problemas de “overfitting”, onde o modelo se ajusta demais aos dados de treino, comprometendo sua eficácia em novos dados. A separação adequada entre os conjuntos é, portanto, um passo crítico para garantir a robustez e a confiabilidade das previsões do modelo (James et al., 2013).

Para realizar a previsão, serão implementados três modelos de aprendizado de máquina: K-Nearest Neighbors (KNN), Random Forest e Gradient Boosting. Esses modelos foram escolhidos por suas diferentes abordagens e características que podem influenciar a eficácia na detecção de fraudes em transações de cartão de crédito. A análise comparativa

dos desempenhos permite identificar o modelo mais adequado ao conjunto de dados, considerando as métricas de avaliação selecionadas.

O K-Nearest Neighbors (KNN) é um método de aprendizado supervisionado baseado na ideia de que instâncias semelhantes tendem a pertencer à mesma classe. Para classificar uma nova instância, o algoritmo calcula a distância entre essa instância e todas as demais do conjunto de treinamento, utilizando métricas como a distância euclidiana, Manhattan ou Minkowski, dependendo do contexto da aplicação (Duda, Hart & Stork, 2001). Em seguida, ele seleciona os k vizinhos mais próximos e atribui à nova instância a classe predominante entre esses vizinhos, seguindo o princípio do voto majoritário (Cover & Hart, 1967). Dessa forma, o desempenho do KNN depende diretamente da escolha do parâmetro k, que define a quantidade de vizinhos considerados na decisão. Valores muito baixos podem tornar o modelo sensível a ruídos e outliers, enquanto valores muito altos podem levar a uma suavização excessiva da fronteira de decisão, reduzindo sua capacidade de capturar padrões mais complexos nos dados (García et al., 2008).

Além disso, o algoritmo é sensível à escala das variáveis, pois atributos com valores numéricos mais elevados podem dominar a medida de distância, tornando essencial a aplicação de técnicas de normalização ou padronização (Han, Kamber & Pei, 2011). Outro fator a ser considerado é o impacto de conjuntos de dados desbalanceados, nos quais as classes majoritárias podem influenciar desproporcionalmente a classificação, comprometendo a precisão para as classes minoritárias (Weiss & Provost, 2003).

A implementação do KNN neste estudo foi realizada utilizando a biblioteca Scikit-learn por meio da classe KNeighborsClassifier, que oferece suporte a diferentes métricas de distância e otimizações computacionais, possibilitando ajustes que melhoram o desempenho do modelo conforme as características do conjunto de dados analisado.

O Random Forest é um algoritmo de aprendizado de máquina usado para classificação, ou seja, para atribuir uma classe ou categoria a uma nova observação com base nos dados de treinamento. Ele cria várias Árvores de Decisão a partir de diferentes subconjuntos dos dados de treinamento e, ao fazer a previsão, cada árvore "vota" na classe mais provável. A classe mais votada entre todas as árvores é a que o modelo final escolhe. Esse processo de combinar várias árvores ajuda o Random Forest a ser mais preciso e robusto em comparação a uma única árvore de decisão, pois ele diminui o risco de erros causados por variabilidades nos dados de treinamento (Breiman, 2001).

Uma das principais vantagens do Random Forest é que ele pode lidar bem com conjuntos de dados complexos, com muitas variáveis e interações entre elas, além de ser menos sensível a ruídos ou outliers presentes nos dados (Cutler et al., 2007). Com a combinação de múltiplas árvores, ele também é capaz de fornecer uma medida de

importância das variáveis, o que ajuda a identificar quais fatores são mais relevantes para a tomada de decisão (Liaw & Wiener, 2002). Neste estudo, a implementação do Random Forest foi realizada utilizando a biblioteca Scikit-learn, por meio da classe RandomForestClassifier.

O Gradient Boosting é um algoritmo de aprendizado de máquina baseado em ensemble que constrói modelos de forma sequencial, ou seja, ele cria várias Árvores de Decisão em etapas. Cada árvore construída tenta corrigir os erros das árvores anteriores, ajustando-se aos casos que as árvores anteriores classificaram incorretamente. Esse processo de "aprendizado com correção" permite que o modelo melhore seu desempenho gradualmente, focando nos erros que ainda precisam ser corrigidos, o que torna o Gradient Boosting muito eficaz para problemas complexos e difíceis de resolver (Friedman, 2001). Neste estudo, a implementação do Gradient Boosting foi realizada utilizando a classe GradientBoostingClassifier da biblioteca Scikit-learn.

Para a avaliação do desempenho dos modelos de classificação implementados, foram selecionados três métricas amplamente utilizadas em problemas de classificação binária: Precision, Recall e F-Score. As bibliotecas do Python Scikit-learn foram empregadas para essa análise, utilizando as funções classification_report para calcular as métricas e visualizar o desempenho dos modelos. O classification_report fornece as métricas de avaliação como precisão, recall e F-score para cada classe. Essa função é implementada diretamente na biblioteca Scikit-learn, que pode ser acessada na documentação oficial (Scikit-learn, 2025).

De acordo com Powers (2011) a métrica Precision (ou Precisão) mede a proporção de verdadeiros positivos em relação ao total de positivos preditos pelo modelo. A Equação 3, refere-se ao cálculo da métrica de Precisão.

$$precision = \frac{TP}{TP + FP} \quad (3)$$

onde TP (True Positives) representa os verdadeiros positivos, ou seja, o número de transações fraudulentas corretamente identificadas pelo modelo, e FP (False Positives) refere-se aos falsos positivos, ou seja, o número de transações legítimas incorretamente classificadas como fraudulentas. A Precision avalia a qualidade das previsões positivas feitas pelo modelo, indicando o quão confiável ele é quando identifica uma transação como fraudulenta. Essa métrica é especialmente relevante em contextos onde o custo de falsos positivos é elevado, como em sistemas de monitoramento de fraudes, onde bloquear ou investigar transações legítimas pode gerar custos operacionais adicionais e prejudicar a experiência do usuário.

A interpretação da Precision pode ser feita da seguinte forma: quando a Precision é alta, significa que, ao classificar uma transação como fraudulenta, é muito provável que ela realmente seja uma fraude. Isso é desejável, pois minimiza o número de transações legítimas que são classificadas erroneamente como fraudulentas. Por outro lado, quando a Precision é

baixa, isso indica que o modelo está classificando muitas transações legítimas como fraudulentas, o que pode resultar em custos adicionais e frustração para os usuários, já que transações legítimas podem ser bloqueadas ou necessitar de verificação adicional. No contexto da detecção de fraudes, uma alta Precision é essencial para garantir a confiabilidade do modelo, minimizando os impactos negativos dos falsos positivos e garantindo que as transações fraudulentas sejam identificadas de maneira eficiente (Powers, 2011).

O Recall, também conhecido como sensibilidade, calcula a proporção de verdadeiros positivos em relação ao total de positivos reais. A Equação 4, refere-se ao cálculo da métrica de Recall.

$$recall = \frac{TP}{TP + FP} \tag{4}$$

onde TP (True Positives) representa os verdadeiros positivos, ou seja, o número de transações fraudulentas corretamente identificadas pelo modelo, e FN (False Negatives) refere-se aos falsos negativos, ou seja, o número de transações fraudulentas que o modelo não conseguiu identificar. O Recall é uma métrica importante para identificar transações fraudulentas corretamente, minimizando o número de falsos negativos, ou seja, transações fraudulentas que passam despercebidas pelo modelo. Essa métrica é particularmente importante em cenários onde o custo de falsos negativos é elevado, como em sistemas de detecção de fraudes, onde a não identificação de uma transação fraudulenta pode resultar em prejuízos financeiros significativos.

Quando o Recall é alto, significa que o modelo está identificando uma grande proporção de transações fraudulentas corretamente, minimizando o risco de permitir que transações fraudulentas passem despercebidas. Por outro lado, quando o Recall é baixo, isso indica que o modelo está deixando muitas transações fraudulentas sem detecção, o que pode ser problemático, especialmente em contextos em que a detecção de fraudes é crítica para a segurança financeira. Portanto, o Recall busca garantir que o maior número possível de fraudes seja identificado, mesmo que isso resulte em alguns falsos positivos. No contexto da detecção de fraudes, um alto Recall é desejável, pois assegura que a maioria das transações fraudulentas seja capturada pelo modelo (Powers, 2011).

O F-Score, também conhecido como F1-Score, é a média harmônica entre precisão e recall. A Equação 5, refere-se ao cálculo da métrica de F-Score.

$$FScore = 2 \times \frac{precisison \times recall}{precision + recall} \tag{5}$$

O F-Score fornece uma métrica balanceada que considera tanto os falsos positivos quanto os falsos negativos, sendo particularmente útil quando há um trade-off entre precisão e recall. Esse trade-off ocorre porque, em algumas situações, um modelo pode ser otimizado para aumentar a precisão em detrimento do recall, ou vice-versa. O F-Score resolve esse

dilema, equilibrando ambos os aspectos e proporcionando uma visão mais equilibrada do desempenho do modelo. Essa métrica é ideal para problemas de classificação com desbalanceamento de classes, como é o caso da detecção de fraudes, pois ela reflete o desempenho geral do modelo em identificar fraudes, sem priorizar uma métrica em detrimento da outra. Em contextos onde as classes são desbalanceadas, como no caso da detecção de fraudes em transações financeiras, o F-Score ajuda a garantir que o modelo não se concentre apenas em identificar transações legítimas (o que poderia aumentar a precisão) ou apenas em identificar fraudes (o que poderia aumentar o recall), mas que busque um equilíbrio entre essas duas métricas para alcançar um desempenho geral eficaz (Sokolova & Lapalme, 2009).

Além disso, os gráficos foram gerados em Python 3.12 (Python Software Foundation, 2025), utilizando as bibliotecas Seaborn (Waskom, 2021) e Matplotlib (Hunter, 2007), em ambiente Jupyter Notebook (Kluyver et al., 2016). A modelagem foi realizada com o pacote Scikit-learn (Pedregosa et al., 2011).

Resultados e Discussão

Para compreender a distribuição das variáveis e suas características estatísticas, foi realizada uma análise descritiva dos dados. A Tabela 2 apresenta estatísticas como média, desvio padrão, valores mínimo e máximo, além dos quartis para cada variável do conjunto de dados. Essa análise é fundamental para identificar padrões, possíveis outliers e a necessidade de pré-processamento antes da aplicação dos métodos de machine learning.

Tabela 2. Resultados da análise descritiva

(continua)

Variáveis	Média	Desvio Padrão	Mínimo	25%	50%	75%	Máximo
Time	94813,8596	47488,1460	0,0000	54201,5000	84692,0000	139320,5000	172792,0000
V1	0,0000	1,9587	-56,4075	-0,9204	0,0181	1,3156	2,4549
V2	0,0000	1,6513	-72,7157	-0,5985	0,0655	0,8037	22,0577
V3	0,0000	1,5163	-48,3256	-0,8904	0,1798	1,0272	9,3826
V4	0,0000	1,4159	-5,6832	-0,8486	-0,0198	0,7433	16,8753
V5	0,0000	1,3802	-113,7433	-0,6916	-0,0543	0,6119	34,8017
V6	0,0000	1,3323	-26,1605	-0,7683	-0,2742	0,3986	73,3016
V7	0,0000	1,2371	-43,5572	-0,5541	0,0401	0,5704	120,5895
V8	0,0000	1,1944	-73,2167	-0,2086	0,0224	0,3273	20,0072
V9	0,0000	1,0986	-13,4341	-0,6431	-0,0514	0,5971	15,5950
V10	0,0000	1,0888	-24,5883	-0,5354	-0,0929	0,4539	23,7451
V11	0,0000	1,0207	-4,7975	-0,7625	-0,0328	0,7396	12,0189
V12	0,0000	0,9992	-18,6837	-0,4056	0,1400	0,6182	7,8484
V13	0,0000	0,9953	-5,7919	-0,6485	-0,0136	0,6625	7,1269

Tabela 2. Resultados da análise descritiva

(conclusão)

Variáveis	Média	Desvio Padrão	Mínimo	25%	50%	75%	Máximo
V14	0,0000	0,9586	-19,2143	-0,4256	0,0506	0,4931	10,5268
V15	0,0000	0,9153	-4,4989	-0,5829	0,0481	0,6488	8,8777
V16	0,0000	0,8763	-14,1299	-0,4680	0,0664	0,5233	17,3151
V17	0,0000	0,8493	-25,1628	-0,4837	-0,0657	0,3997	9,2535
V18	0,0000	0,8382	-9,4987	-0,4988	-0,0036	0,5008	5,0411
V19	0,0000	0,8140	-7,2135	-0,4563	0,0037	0,4589	5,5920
V20	0,0000	0,7709	-54,4977	-0,2117	-0,0625	0,1330	39,4209
V21	0,0000	0,7345	-34,8304	-0,2284	-0,0295	0,1864	27,2028
V22	0,0000	0,7257	-10,9331	-0,5424	0,0068	0,5286	10,5031
V23	0,0000	0,6245	-44,8077	-0,1618	-0,0112	0,1476	22,5284
V24	0,0000	0,6056	-2,8366	-0,3546	0,0410	0,4395	4,5845
V25	0,0000	0,5213	-10,2954	-0,3171	0,0166	0,3507	7,5196
V26	0,0000	0,4822	-2,6046	-0,3270	-0,0521	0,2410	3,5173
V27	0,0000	0,4036	-22,5657	-0,0708	0,0013	0,0910	31,6122
V28	0,0000	0,3301	-15,4301	-0,0530	0,0112	0,0783	33,8478
Amount	88,3496	250,1201	0,0000	5,6000	22,0000	77,1650	25691,1600

Fonte: Resultados originais da pesquisa

A análise descritiva das variáveis revela a distribuição dos dados em termos de medidas estatísticas centrais e de dispersão. A variável Time apresenta média de aproximadamente 94.814 e desvio padrão de 47.488, indicando a variação temporal das transações registradas. A variável Amount, que representa o valor da transação, possui uma média de 88,35 e um desvio padrão elevado de 250,12, evidenciando uma ampla dispersão nos valores das transações financeiras.

As variáveis V1 a V28 possuem média próxima de zero e diferentes magnitudes de desvio padrão, o que indica que passaram por um processo de transformação que centralizou os valores. Os valores mínimos e máximos dessas variáveis demonstram a presença de uma ampla variação entre os dados, e os quartis revelam a distribuição dessas variações dentro do conjunto analisado. Como as variáveis V1 a V28 não possuem uma descrição explícita de seu significado original, apenas suas distribuições estatísticas podem ser observadas.

Além disso, como as variáveis Time e Amount não passaram pela transformação via PCA, foi necessário avaliar suas distribuições de forma específica. A seguir, apresenta-se a análise gráfica e o procedimento de normalização adotado.

Com base na análise exploratória realizada por meio de gráficos de caixa (boxplots), observou-se a presença de diversos outliers nas variáveis Time e, especialmente, Amount, o que pode comprometer o desempenho de algoritmos sensíveis à escala dos dados. A Figura 1 ilustra essa distribuição, evidenciando a assimetria e a existência de valores extremos expressivos, principalmente nos valores monetários das transações.

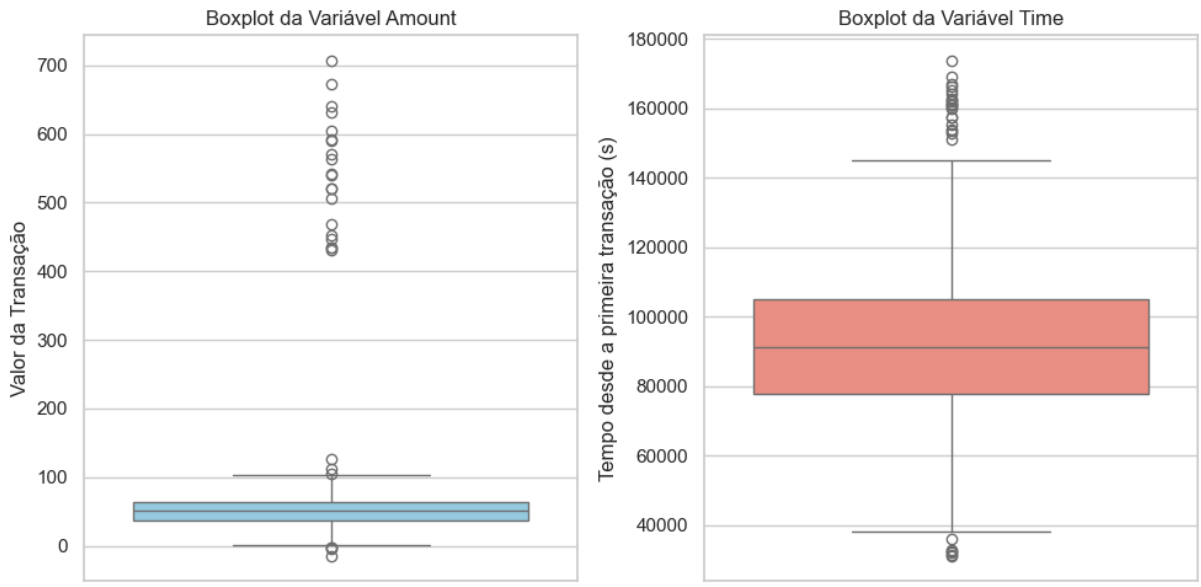

Figura 1 – Distribuição das variáveis Amount e Time antes do escalonamento (boxplots)
Fonte: Elaboração própria com base nos dados do Machine Learning Group – Universidade Livre de Bruxelas (ULB, 2025)
Nota: Figura gerada em Python com biblioteca Seaborn

Diante desse cenário, foi adotado o método RobustScaler, da biblioteca Scikit-learn, como técnica de escalonamento das variáveis Time e Amount. O RobustScaler realiza a normalização com base na mediana e na amplitude interquartílica (IQR), sendo mais adequado para conjuntos de dados que apresentam valores extremos, por reduzir o impacto de outliers na transformação das variáveis (Pedregosa et al., 2011; Scikit-learn, 2025).

Após a aplicação do escalonamento, as variáveis originais foram removidas do conjunto de dados e substituídas pelas colunas scaled_time e scaled_amount. Essa transformação garantiu que todas as variáveis estivessem em escalas compatíveis, o que contribui diretamente para o desempenho dos métodos de classificação e para a estabilidade das métricas de avaliação, como demonstrado nas análises subsequentes (Hastie, Tibshirani e Friedman, 2009).

A seleção das variáveis para o modelo de classificação foi realizada com base na análise de correlação entre as variáveis preditoras e a variável alvo Class. Inicialmente, foi aplicado o teste de normalidade de Shapiro-Wilk (Shapiro; Wilk, 1965) e o teste de Kolmogorov-Smirnov (Massart et al., 1983) para verificar a distribuição dos dados. A variável V13 foi a única a apresentar distribuição normal, com p-valores de 0,09775 no teste de Shapiro-Wilk e 0,40678 no teste de Kolmogorov-Smirnov. No entanto, a correlação dessa variável com a variável alvo não foi suficiente para sua inclusão no modelo.

Em razão da predominância de variáveis com distribuição não normal, foi utilizada a correlação de Spearman (Spearman, 1904) para avaliar a relação monotônica entre as variáveis. A Figura 2 apresenta a matriz de correlação gerada para todas as variáveis do

conjunto de dados, destacando visualmente as associações entre os atributos, inclusive em relação à variável alvo Class. A matriz foi gerada com a biblioteca Seaborn v. 0.12.2 e Matplotlib v. 3.7.1, ambas implementadas em Python.

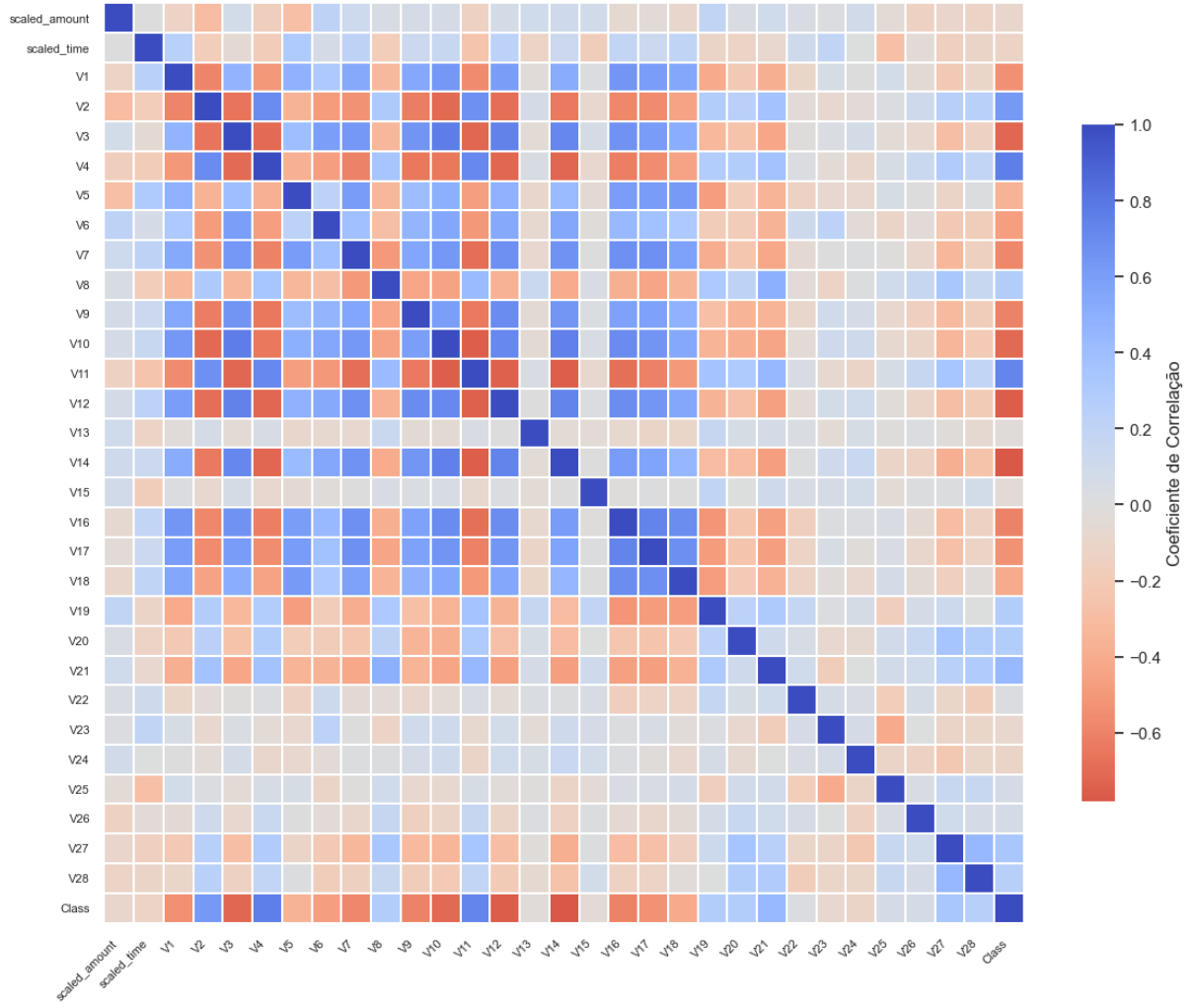

Figura 2. Matriz de correlação entre variáveis
Fonte: Elaboração própria com base nos dados do Machine Learning Group – Universidade Livre de Bruxelas (ULB, 2025)
Nota: Figura gerada em Python com biblioteca Seaborn

Foram selecionadas as variáveis que apresentaram correlação acima de 0,5 ou abaixo de -0,5 com a variável alvo, conforme os resultados apresentados na Tabela 3.

Tabela 3. Resultado da correlação de Spearman

(continua)

Variável	Correlação com 'Class'
V4	0,7643
V11	0,7325
V2	0,6073

V17	-0,5382
V7	-0,5829
V9	-0,5874
V16	-0,5988

Tabela 3. Resultado da correlação de Spearman

(conclusão)

Variável	Correlação com 'Class'
V3	-0,7043
V10	-0,7158
V12	-0,7587
V14	-0,7730

Fonte: Resultados originais da pesquisa

Essas variáveis foram selecionadas por apresentarem uma forte relação com a variável alvo, o que facilita a discriminação entre transações fraudulentas e não fraudulentas. Com base nesse conjunto de variáveis, os modelos de classificação foram treinados e avaliados.

Com isso, os modelos de classificação foram treinados utilizando uma divisão de 70% para treino e 30% para teste, conforme a metodologia descrita. A seguir, apresentam-se os resultados obtidos para cada modelo, com base nas métricas de Precision, Recall e F-Score.

Tabela 4. Resultados dos Modelos de Classificação

Modelo	Precision	Recall	F-Score
Random Forest	0.9423	0.9462	0.9424
Gradient Boosting	0.9450	0.9487	0.9457
K-Nearest Neighbors (KNN)	0.9467	0.9506	0.9459

Fonte: Resultados originais da pesquisa

Os resultados obtidos na avaliação dos métodos Random Forest, Gradient Boosting e K-Nearest Neighbors (KNN), com base nas métricas de precision, recall e F1-Score, indicaram que o modelo KNN apresentou o melhor desempenho geral. Esse modelo obteve uma precisão de 0,9467, recall de 0,9506 e F1-Score de 0,9459, demonstrando sua capacidade de identificar corretamente as transações fraudulentas e, ao mesmo tempo, manter uma taxa reduzida de falsos positivos.

O modelo Gradient Boosting apresentou resultados muito próximos, com F1-Score de 0,9457, seguido do Random Forest, com F1-Score de 0,9424. Apesar da diferença numérica ser pequena, o desempenho ligeiramente superior do KNN indica um equilíbrio mais eficiente entre precision e recall, características essenciais em sistemas de detecção de fraude.

Esses resultados demonstram que todos os métodos utilizados são viáveis para a tarefa, mas o KNN se destacou pelo equilíbrio entre suas métricas. O Gradient Boosting também mostrou desempenho consistente, o que reforça sua utilidade. Já o Random Forest, embora tenha apresentado a menor pontuação entre os três, ainda obteve métricas elevadas, mantendo-se como uma alternativa robusta, especialmente em cenários com dados mais complexos ou sujeitos a ruído.

Considerando o contexto de transações financeiras, no qual tanto os falsos positivos quanto os falsos negativos podem gerar impactos significativos, a escolha do modelo ideal deve priorizar o equilíbrio entre identificar corretamente fraudes reais e evitar alertas falsos. Nesse sentido, o KNN apresentou a combinação mais eficiente, sendo o mais adequado para este estudo.

Conclusão

Este trabalho teve como objetivo avaliar a eficácia de diferentes algoritmos de aprendizado de máquina na detecção de fraudes em transações com cartão de crédito. Para isso, foram comparados os desempenhos dos métodos K-Nearest Neighbors (KNN), Random Forest e Gradient Boosting, utilizando as métricas de precision, recall e F1-Score.

A abordagem metodológica incluiu o balanceamento das classes com o método de Random Under-Sampling e a seleção de variáveis com base na correlação com a variável alvo. Essa estratégia contribuiu para reduzir o impacto do desbalanceamento e permitiu a construção de modelos mais eficientes e consistentes.

Os resultados obtidos demonstraram que todos os modelos foram eficazes na detecção de fraudes. No entanto, o KNN apresentou o melhor desempenho global, seguido de perto pelo Gradient Boosting e, por fim, pelo Random Forest. A escolha do modelo ideal deve considerar o equilíbrio entre precision e recall, especialmente em cenários sensíveis como o financeiro.

Concluiu-se que os modelos de aprendizado de máquina são ferramentas promissoras para a prevenção de fraudes em sistemas de pagamento digital. Sua aplicação pode contribuir para a redução de perdas financeiras e o aumento da confiança dos usuários. Estudos futuros podem explorar novos algoritmos, técnicas de balanceamento alternativas e bases de dados mais amplas ou atualizadas para aprimorar os resultados obtidos.

Referências

Abdi, H.; Williams, L. J. 2010. Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics 2(4): 433-459.

Aggarwal, C. C. 2017. Outlier analysis. 2. ed. Cham: Springer.

Anderson, R.; Barton, C.; Böhme, R.; Clayton, R.; van Eeten, M. J. G.; Levi, M.; Moore, T.; Savage, S. 2020. Measuring the cost of cybercrime. *In:* The economics of information security and privacy. Cham, Springer. p. 265–300.

Batista, G. E. A. P. A.; Prati, R. C.; Monard, M. C. 2004. A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explorations Newsletter, 6(1): 20-29.

Breiman, L. 2001. Random forests. Machine Learning 45(1): 5-32.

Chawla, N.V.; Kegelmeyer, W.P.; Bowyer, K.W.; Hall, L.O.; Philip, S.Y. 2002. SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research 16( ): 321-357.

Cover, T.; Hart, P. 1967. Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13(1): 21-27.

Cutler, D.R.; Edwards Jr, T.C.; Beard, K.H.; et al. 2007. Random forests for classification in ecology. Ecology 88(11): 2783-2792.

Duda, R.O.; Hart, P.E.; Stork, D.G. 2001. Pattern classification. 2ed. Wiley, New York, EUA

Evans, D. S.; Schmalensee, R. 2005. Paying with plastic: the digital revolution in buying and borrowing. 2ed. Cambridge, MIT Press.

Friedman, J.H. 2001. Greedy function approximation: a gradient boosting machine. Annals of Statistics 29(5): 1189-1232.

García, V.; Sánchez, J.S.; Márquez, G. 2008. On the effectiveness of preprocessing methods when dealing with different levels of class imbalance in classification problems: Knowledge-Based Systems 25(1): 13-21

Géron, A. 2019. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. 2. ed. O'Reilly Media, Sebastopol, EUA.

Guilford, J.P. 1956. Fundamental statistics in psychology and education. McGraw-Hill, New York, EUA.

Han, J.; Kamber, M.; Pei, J. 2011. Data mining: concepts and techniques. 3ed. Morgan Kaufmann, Waltham.

Hastie, T.; Tibshirani, R.; Friedman, J. 2009. The elements of statistical learning: data mining, inference, and prediction. 2ed. Springer, New York, NY, EUA.

He, H.; Garcia, E.A. 2009. Learning from imbalanced data: IEEE Transactions on Knowledge and Data Engineering 21(9): 1263-1284.

Hunter, J. D. 2007. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3): 90–95.

James, G.; Witten, D.; Hastie, T.; Tibshirani, R. 2013. An introduction to statistical learning: with applications in R. New York, Springer.

Jolliffe, I.T.; Cadima, J. 2016. Principal component analysis: a review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 374(2065): 20150202.

Kardefelt-Winther, D. Artificial intelligence and child rights. Florença, UNICEF Office of Global Insight and Policy, 2021. Disponível em: https://www.unicef.org/globalinsight/reports/artificial-intelligence-and-child-rights. Acesso em: 08 abr. 2025.

Kluyver, T.; et al. 2016. Jupyter Notebooks – a publishing format for reproducible computational workflows. Amsterdam: IOS Press.

Kuhn, M.; Johnson, K. 2013. Applied predictive modeling. Springer, New York, NY, EUA

Liaw, A.; Wiener, M. 2002. Classification and regression by randomForest. R News 2(3): 18-22.

Machine Learning Group - Universidade Livre de Bruxelas [ULB]. Credit card fraud detection. Disponível em: [https://www.kaggle.com/mlg-ulb/creditcardfraud. Acesso em: 20 janeiro 2025.

Massart, D.L.; Kaufman, L.; Rousseeuw, P.J.; Lejeune, M. 1983. Least median of squares: a robust method for outlier and model error detection in regression and calibration. Analytica Chimica Acta 150: 227-236.

Mukaka, M.M. 2012. A guide to appropriate use of correlation coefficient in medical research. Malawi Medical Journal 24(3): 69-71.

Nilson Report. Global cards: growth, market share and industry data. Nilson Report, issue 1243, abr. 2023. Disponível em: https://nilsonreport.com. Acesso em: 08 abr. 2025.

Pearson, K. 1895. Mathematical contributions to the theory of evolution. III. Regression, heredity, and panmixia. Philosophical Transactions of the Royal Society of London 187: 253-318.

Pedregosa, F.; Varoquaux, G.; Gramfort, A. 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, v. 12, p. 2825–2830.

Powers, D.M.W. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation. Journal of Machine Learning Technologies 2(1): 37-63.

Python Software Foundation. 2025. Python: a programming language (versão 3.12). Disponível em: https://www.python.org/. Acesso em: 18 ago. 2025.

Razali, N.M.; Wah, Y.B. 2011. Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. Journal of Statistical Modeling and Analytics 2(1): 21-33.

Riedman, J.H. 2001. Greedy function approximation: a gradient boosting machine. Annals of Statistics 29(5): 1189-1232.

Schmidt, F.L.; Hunter, J.E. 1996. Measurement error in psychological research: lessons from 26 research scenarios. Psychological Methods 1(2): 199-223.

Scikit-learn developers. 2020. Scikit-learn: Machine Learning in Python (v0.24). Disponível em: https://scikit-learn.org. Acesso em: 20 jan. 2025.

Shapiro, S.S.; Wilk, M.B. 1965. An analysis of variance test for normality (complete samples). Biometrika 52(3/4): 591-611.

Sokolova, M.; Lapalme, G. 2009. A systematic analysis of performance measures for classification tasks: Information Processing & Management 45(4): 427-437.

Spearman, C. 1904. The proof and measurement of association between two things: The American Journal of Psychology 15(1): 72-101.

Tukey, J. W. 1977. Exploratory data analysis. Reading, MA: Addison-Wesley.

Waskom, M. L. 2021. Seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60): 3021.

Weiss, G. M.; Provost, F. 2003. The effect of class distribution on classifier learning: an empirical study: Technical Report ML-TR-43, Department of Computer Science, Rutgers University.

Zhang, Y.; Zhou, J.; Wang, J.; Gong, Z.; Wang, Y. Fraud detection in online financial transactions using machine learning algorithms. Information, 10(7): 232, 2019. Disponível em: https://doi.org/10.3390/info10070232. Acesso em: 08 abr. 2025.