Detecção de fraudes bancárias: uma análise de performance do algoritmo TabPFN

Joel Antônio Godoy de Moraes (ESALQ/USP) *j.godoy.moraes@gmail.com*
Jailson dos Santos Silva (UFSC) *engjailsonsantos@outlook.com*

Resumo

As fraudes são uma grande preocupação para as instituições bancárias, uma vez que impactam não apenas a receita, mas também a imagem delas como instituições seguras para os clientes. Desse modo, a capacidade de identificar e remediar transações fraudulentas constitui um objetivo relevante para essas organizações. No entanto, os analistas se deparam com o problema de desbalanceamento de classes, comumente observado neste contexto, que afeta significativamente o desempenho dos modelos de classificação. Assim, o objetivo desta pesquisa é analisar o desempenho do modelo TabPFN em relação a modelos tradicionalmente empregados no mercado para este fim, tais como a Regressão Logística e o *XGBoost*. Para tanto, foram realizados nove experimentos, utilizando-se dois bancos de dados, um deles contendo informações sobre transações bancárias em geral e outro sobre transações com o cartão de crédito, classificadas como fraudulentas ou não. Para contornar a problemática do desbalanceamento de classes, foram adotadas técnicas de *oversampling* e *undersampling*. Como métricas de avaliação, foram adotadas a acurácia média, a área sob a curva ROC, o F1-*score*, a precisão, o *recall* e os tempos de treino e teste. Os resultados mostraram que o TabPFN ainda não é capaz de substituir os algoritmos tradicionalmente utilizados no contexto real de detecção de fraudes. Todavia, os achados desta pesquisa representam um norte para novas pesquisas e para futuras aplicações do modelo analisado.

Palavras-Chaves: *Desbalanceamento de classes, Redes Neurais, Prior-Data Fitted Networks, Gradient Boosting Decision Trees.*

Abstract

Fraud constitutes a major concern for banking institutions, as it affects not only their revenues but also their image as secure organizations for customers. Consequently, the ability to identify and remediate fraudulent transactions represents a relevant objective for these organizations. However, analysts face the problem of class imbalance, which is commonly observed in this

context and significantly affects the performance of classification models. Thus, the objective of this study is to analyze the performance of the TabPFN model relative to models traditionally used for this purpose, such as Logistic Regression and XGBoost. To this end, nine experiments were conducted using two datasets: one containing information on general banking transactions and another comprising credit card transactions, classified as fraudulent or non-fraudulent. To address the class imbalance, oversampling and undersampling techniques were used. The evaluation metrics included mean accuracy, the area under the ROC curve (AUC), F1-score, precision, recall, and training and testing times. The results indicate that TabPFN is not yet able to replace the algorithms traditionally used in real-world fraud detection contexts. Nevertheless, the findings of this research provide guidance for future studies and potential applications of the analyzed model.

Keywords: *Class imbalance; Neural Networks; Prior-Data Fitted Networks; Gradient Boosting Decision Tree.*

1. Introdução

O problema de detecção de fraudes é uma grande preocupação para as instituições financeiras, uma vez que as fraudes bancárias impactam não apenas a receita dessas, mas também a sua imagem enquanto instituições seguras para os clientes (Tosta; Dias, 2025).

Com a crescente bancarização, o número de transações bancárias de diferentes tipos também aumentou, em particular com a introdução de meios digitais, que hoje são responsáveis pela maior parte das transações bancárias, segundo o Banco Central do Brasil (BACEN, 2024).

Atualmente, é comum que as instituições financeiras lidem com milhões de transações diárias, o que torna imperativa a necessidade de desenvolver métodos computacionais autônomos que garantam a integridade dessas transações, sem a interferência humana. Nesse contexto, o desenvolvimento de modelos de detecção de fraudes torna-se uma das prioridades para lidar com os milhões de transações diárias, prevenindo fraudes (Tosta; Dias, 2025).

Pode-se definir fraude como um ato deliberado e ilícito de ludibriar ou manipular pessoas, sistemas ou documentos com o intuito de obter vantagens pessoais. No caso das fraudes do sistema financeiro, são atitudes que visam deturpar a integridade das operações dos sistemas bancários e financeiros, de modo a produzir uma vantagem ou ganho econômico para o fraudador (Lima, 2025).

Do ponto de vista da Ciência de Dados, o problema de detecção de fraudes apresenta desafios interessantes, a começar pelo fato de que tipicamente se lida com bases de dados largamente desbalanceadas, uma vez que um número bastante reduzido das transações, em relação ao total, constitui transações fraudulentas (Martins; Galegale, 2022). Desta forma, é necessária uma avaliação cautelosa dos modelos para discriminar se um classificador que indica transações fraudulentas está desempenhando bem seu papel ou não, dado que a acurácia, por si só, pode oferecer uma falsa percepção de bom desempenho (Fávero; Belfiore, 2024).

Uma forma de atenuar o problema das bases de dados largamente desbalanceadas é a criação de dados sintéticos da classe minoritária (*oversampling*) ou a remoção seletiva de amostras da classe majoritária (*undersampling*), o que pode facilitar aos modelos de aprendizado de máquina aprender a discriminar melhor as amostras de ambas as classes (Rubaidi; Ammar; Aouicha, 2022).

Outra dificuldade das instituições financeiras é que o problema de detecção de fraudes demanda um ciclo *Extract, Transform, Load (ETL) muito rápido, exigindo, por conseguinte, um novo processo de treinamento dos modelos com* os novos dados. Isso ocorre porque os fraudadores estão constantemente criando formas de burlar os dispositivos de segurança dessas instituições (Martins; Galegale, 2022).

Atualmente, o cenário dos modelos para este fim, bem como dos demais cenários de problemas baseados em dados tabulares, é dominado pelos Gradient Boosting Decision Trees (GBDT), que tradicionalmente desempenham melhor nas tarefas de classificação de dados tabulares do que outras técnicas, apresentando tempo de treinamento reduzido e elevada robustez (Shwartz-Ziv; Armon, 2022). Contudo, recentemente foi desenvolvido o modelo *Tabular Prior-Data Fitted Network* (TabPFN), baseado em redes neurais, que oferece uma alternativa a esses modelos tradicionais consolidados (Hollmann et al., 2022).

De acordo com os autores do método, Hollmann et al. (2022), essa solução apresenta desempenho similar ou superior ao dos métodos GBDT tradicionais na classificação e é muito mais rápida, pois executa as etapas de treino e teste sobre um único conjunto de dados e não sofre modificação nos seus pesos sinápticos após o pré-treino *off-line*.

Diante deste novo cenário, o objetivo deste artigo é analisar o desempenho do modelo TabPFN em comparação com modelos tradicionalmente empregados no mercado para a detecção de fraudes, tais como a Regressão Logística e o *XGBoost*.

2. Referencial Teórico

2.1. Regressão Logística

O modelo de Regressão Logística consiste essencialmente em um modelo de aprendizado de máquina supervisionado, linear nos parâmetros, que busca classificar as amostras por meio da estimação da probabilidade de que uma amostra pertença a uma classe específica, obtida pela eq. (1), conforme discute Fávero e Belfiore (2024).

$$p = \frac{1}{1 + e^{-Z_i}} = \frac{1}{1 + e^{-(\alpha + \beta_1 X_{1i} + \cdots + \beta_k X_{1k})}} \quad (1)$$

onde, p é a probabilidade de pertencimento a classe definida como positiva, variando entre 0 e 1, α e β_j, com j=1...k são os parâmetros do modelo logístico a serem ajustados no treinamento e X_{ij} se trata das variáveis de cada amostra i do banco de dados.

O processo de ajuste dos parâmetros é realizado por meio do método da Máxima Verossimilhança, visando maximizar o acerto das classes das amostras rotuladas (Fávero; Belfiore, 2024).

2.2. *XGBoost*

O modelo *Extreme Gradient Boosting* (*XGBoost*) consiste em uma implementação que visa construir uma diversidade de árvores de decisão de forma sequencial. O objetivo é treinar várias árvores de decisão, testando diversas combinações de hiperparâmetros para selecionar a melhor árvore, porém criando-as em sequência, de modo a reduzir progressivamente os erros cometidos pelos modelos anteriores (Géron, 2022). Neste artigo, foi utilizado o método *Grid Search* para realizar uma busca metódica sobre o espaço de valores possíveis para os hiperparâmetros desse modelo, utilizando como critério de seleção para o melhor modelo o valor da área sob a curva ROC.

2.3. TabPFN

O modelo TabPFN consiste em uma rede neural de arquitetura *Transformer* do tipo *Prior-Data Fitted Network* (PFN). A rede passa por um processo de pré-treino *off-line*, realizado sobre conjuntos de dados sintéticos, gerados por redes neurais bayesianas e modelos causais estruturados, e é avaliada na tarefa de previsão de conjuntos de dados rotulados, em vez de prever os rótulos de amostras individuais desses conjuntos de dados, realizando o aprendizado em contexto (Hollmann et al., 2022).

A previsão dos rótulos em contexto ocorre na medida que as redes neurais do tipo PFN buscam aproximar as distribuições de probabilidade dos rótulos em função das amostras, ao invés de tentar prever diretamente o rótulo a partir da amostra de dado, de modo que a escolha do método de geração dos dados sintéticos utilizados no pré-treino é fundamental para que a rede neural possa aprender a aproximar uma ampla gama de funções de distribuição de probabilidade, adquirindo robustez suficiente para ser utilizada sobre qualquer conjunto de dados (Müller et al., 2022).

Uma vez concluído o pré-treino, a rede está pronta para uso em um conjunto de dados real. O conjunto de dados rotulado de treino é passado para a rede que realiza um processo de treinamento muito mais rápido e diferenciado que consiste no ajuste da função de distribuição de probabilidades dos rótulos para as amostras de treinamento, chamada de *Predictive Probabilty Distribution* (PPD) (Nagler, 2023). As amostras de teste não rotuladas, são passadas na sequência e o rótulo das amostras é então obtido através da PPD aproximada no treinamento, de modo que todos os rótulos de todas as amostras do conjunto de teste são aproximados de uma única vez, o que também oferece um ganho significativo no tempo necessário para realizar a tarefa de classificação.

2.4. SMOTE

O método *Synthetic Minority Over-sampling Technique* (SMOTE) é um método de balanceamento de conjuntos de dados com classes desbalanceadas que gera amostras sintéticas de dados até que a proporção entre as amostras de cada classe seja atingida. O algoritmo identifica a classe minoritária nas amostras do conjunto de dados fornecido e, em seguida, realiza a identificação de um número k de vizinhos mais próximos, onde k é um parâmetro passado para o algoritmo. Após a identificação de seus vizinhos, as amostras sintéticas podem ser geradas realizando a interpolação linear entre uma amostra e seus vizinhos e escolhendo-se um ponto com distância intermediária entre eles, introduzindo uma perturbação aleatória com valor entre 0 e 1 no processo de geração da amostra. O processo é repetido até que a proporção das amostras de cada classe seja atingida (Chawla et al., 2002).

2.5. *Near Miss*

O algoritmo *Near Miss*, versão 3, consiste em um método de balanceamento de conjuntos de dados com classes desbalanceadas que utiliza subamostragem ou *undersampling*. O processo ocorre em duas etapas, sendo a primeira a identificação de cada instância da classe minoritária seguida da seleção de um determinado k de amostras mais próximas, da classe majoritária, de cada uma dessas instâncias. Após essa seleção, a versão 3 do algoritmo, identifica um valor N de vizinhos mais próximos da instância de referência que pertencem à classe minoritária e calcula a distância média dos k vizinhos a essas amostras. Por fim, apenas o subconjunto de amostras da classe majoritária das k originais é mantido, sendo aqueles cuja

distância média até os N vizinhos é maior. O processo é repetido até que a proporção desejada entre as amostras de cada classe seja atingida (Tanimoto et al., 2022).

3. Metodologia

Para avaliar o desempenho do TabPFN, foram realizados nove testes, comparando-o com outros dois modelos de aprendizado de máquina supervisionado: a Regressão Logística e o *XGBoost*. Foram utilizados dois bancos de dados públicos provenientes da plataforma *Kaggle*, um com dados de transações de diversos meios de pagamento e outro especificamente com dados de transações de cartão de crédito. O primeiro banco de dados possui um total de 116 fraudes em 101.613 amostras de transações (0,11%), enquanto o segundo possui 492 fraudes em 284.315 amostras (0,17%).

Os nove testes de comparação entre os modelos foram realizados a partir do seguinte fluxo: para os quatro primeiros, foi realizada a amostragem de 20.000 amostras de ambos os bancos de dados a serem estratificadas em treinamento e teste (50% para cada), respeitando a proporção entre as amostras das classes positiva (fraude) e negativa (não-fraude). Nos primeiros dois experimentos não se realizou balanceamento das amostras selecionadas.

Nos quatro experimentos seguintes, em que as amostras foram balanceadas, estratificaram-se os bancos de dados em treino e teste antes de aplicar os procedimentos de balanceamento, realizados somente nas amostras reservadas para treinamento dos modelos. Para o balanceamento foram usadas técnicas de *oversampling*, através do método SMOTE, e de *undersampling*, através do método *Near Miss*.

Por fim, os últimos três testes realizados envolveram a utilização dos bancos de dados completos, sendo o último teste realizado com o banco de dados de fraudes em transações bancárias balanceado.

Esses testes exigiram realizar procedimentos de adaptação do TabPFN, utilizando diferentes técnicas de pré-processamento para contornar seu limite de pré-treino de 10.000 amostras de treino e de teste (Hollmann et al., 2022). As técnicas utilizadas para adaptação do modelo foram: *subsampling*, *post-hoc ensemble* com *Random Forest* e extração de *embeddings* utilizando uma árvore de decisão.

Na técnica de *subsampling*, já programada no pacote no qual o modelo foi disponibilizado, o TabPFN selecionou aleatoriamente um subconjunto do conjunto de

treinamento e o utilizou para realizar o aprendizado, tentando generalizar para o conjunto de teste a distribuição de probabilidade das classes aprendida no processo de treinamento.

Na técnica de *post-hoc ensemble*, foi usado o método *Random Forest* para gerar diversas árvores de decisão, as quais, por sua vez, particionam os dados em subconjuntos, de forma que o número de amostras em cada subconjunto esteja dentro dos limites do número de amostras suportadas pelo TabPFN (Hollmann et al., 2022). Em seguida, o TabPFN é executado em cada folha das árvores de decisão e por fim, realizou-se uma média das classificações das árvores, utilizando *Hard* ou *Soft voting* (método padrão), obtendo assim a classe das amostras.

Por fim, na extração de *embeddings*, utilizou-se o método *Random Forest* para extrair *embeddings* das amostras de treinamento, treinando as árvores de decisão geradas nesse conjunto de dados e o particionando, de modo que cada amostra é alocada em um nó folha. O caminho percorrido pelas amostras em cada árvore é usado para criar um *embedding* no qual cada nó da árvore é codificado como um elemento indicando sua presença ou ausência. Após atribuir os *embeddings* a cada amostra, realizou-se o agrupamento do conjunto de treinamento em 10.000 clusters, utilizando o método *K-Means*, técnica de agrupamento não hierárquica, e selecionou-se de cada *cluster* uma amostra representativa e aleatória para treinar o TabPFN. No teste, deve-se realizar a classificação em *batches* de 10.000 amostras e depois concatenar os resultados de cada classificação.

Para garantir a aleatoriedade no processo de seleção, as amostras originais dos bancos de dados foram sorteadas utilizando cinco diferentes sementes aleatórias antes de realizar a divisão das amostras entre treino e teste, a qual utilizou a mesma semente aleatória para a estratificação. O procedimento de amostragem, inevitavelmente, causa perda de informação, de modo que a seleção aleatória, realizada conforme o descrito, é fundamental para garantir que os modelos sejam submetidos as amostras tão representativas do conjunto original quanto possível.

A divisão das amostras entre treino e teste foi feita separando 50% das amostras para treino e 50% para teste. Ainda, cada método passou por um processo de treinamento distinto, adequando-se às mesmas condições de comparação entre métodos, conforme discutido em Hollman et al. (2022).

O TabPFN, simplesmente realizou o procedimento de treino e teste (eventualmente incrementado com as técnicas de pré-processamento quando necessário para suportar mais amostras do que o estabelecido no limite de pré-treino), somente contando com a estratificação

das amostras, já que ele não se beneficia, devido às configurações de pré-treino, de procedimentos como o *cross-validation*, de modo que se limitou ao *holdout*; a Regressão Logística foi treinada utilizando o procedimento de *cross-validation* sobre o conjunto de treinamento e depois testada no conjunto de teste; por fim, no caso do *XGBoost*, realizou-se o procedimento de *Grid Search* para otimização de hiperparâmetros com *cross-validation* realizado sobre o conjunto separado para o treinamento, antes de avaliar seu desempenho no conjunto de teste e compará-lo com os demais métodos.

Para avaliar os modelos, foram empregadas as seguintes métricas de desempenho: a acurácia média, a área sob a curva ROC (ROC AUC), o F1-*score*, a precisão, a sensibilidade ou *recall* e os tempos de treino e teste. Todas as medidas tiveram seus valores médios extraídos, isto é, retirou-se a média dos resultados dos testes executados para cada configuração obtida por uma semente aleatória, para um determinado banco de dados.

4. Resultados e Discussão

Os primeiros testes de comparação entre os modelos de aprendizado de máquina em questão, limitados a 10.000 amostras de treino e 10.000 amostras de teste, com classes estratificadas e sem o balanceamento das bases de dados, tiveram um resultado insatisfatório para o TabPFN. No caso, para o banco de dados de transações bancárias, o modelo mostrou-se incapaz de aprender a discriminar as transações fraudulentas das não fraudulentas, como se vê na Tabela 1.

Tabela 1 - Resultado para o banco de dados sem balanceamento

Métricas	**Regressão Logística**	***XGBoost***	**TabPFN**
Acurácia	0,9995	0,9991	0,9990
ROC AUC	0,8098	0,9718	0,9863
F1-*score*	0,8247	0,2000	0
Precisão	0,8048	0,6000	0
Recall	0,8881	0,1200	0
Tempo (s)	1,976	104,9	1,080

Fonte: Os autores

Vê-se também que, para esse mesmo banco de dados, o *XGBoost* também não apresentou um bom desempenho, mostrando um baixo aprendizado, aparentemente, devido ao baixo número de amostras de fraudes, contudo, é preciso ressaltar que nem o *XGBoost* e tampouco a Regressão Logística são limitados pelo limite de pré-treino de 10.000 amostras, de

modo que, se esse limite for ignorado, ambos os algoritmos podem vir a apresentar um bom desempenho para o problema analisado.

Já os testes com o segundo banco de dados, contendo fraudes em transações no cartão de crédito, sem balanceamento da base de dados, revelaram um resultado bastante positivo para o TabPFN, no qual ele se saiu melhor do que seus dois competidores, com destaque para seu F1-*score* de 88,97%, muito superior aos resultados obtidos pelo *XGBoost* e pela Regressão Logística, indicando que nesse cenário o TabPFN foi capaz de minimizar tanto os falsos positivos quanto os falsos negativos com eficácia conforme se nota na Tabela 2.

Tabela 2 - Resultado para o banco de dados de fraudes no cartão de crédito sem balanceamento

Métricas	**Regressão Logística**	***XGBoost***	**TabPFN**
Acurácia	0,9989	0,9992	0,9996
ROC AUC	0,9350	0,9769	0,9876
F1-*score*	0,6622	0,7160	0,8897
Precisão	0,8154	0,8724	0,9328
Recall	0,6380	0,6861	0,8570
Tempo (s)	2,359	104,3	1,698

Fonte: Os autores

Neste segundo cenário, o baixo número de amostras de fraude disponíveis não foi um obstáculo para o aprendizado do algoritmo, tampouco para os outros modelos, mas o TabPFN mostrou possuir o melhor *trade-off* entre acurácia e tempo de execução, tal como seria esperado de acordo com seus autores (Hollmann et al., 2022).

Nos testes seguintes, foram selecionadas as 20.000 amostras dos bancos de dados e balanceados os dados de treinamento com o método SMOTE. No caso dos testes realizados no banco de dados de fraudes bancárias, o *XGBoost* se saiu melhor do que os outros dois modelos, obtendo resultados um pouco melhores do que a Regressão Logística obteve para o mesmo banco de dados, sem balanceamento. Por outro lado, a Regressão Logística e o TabPFN tiveram desempenhos muito ruins na tarefa de classificação nessas condições, conforme pode ser visto na Tabela 3.

Tabela 3 - Resultados para o banco de dados balanceado com SMOTE

Métricas	Regressão Logística	*XGBoost*	TabPFN
Acurácia	0,9394	0,9930	0,9970
ROC AUC	0,9637	0,9998	0,9999
F1-*score*	0,9406	0,9930	0,9970
Precisão	0,9221	0,9885	0,9966
Recall	0,9598	0,9975	0,9974
Tempo (s)	0,4322	163,2	0,8488

Fonte: Os autores

No segundo banco de dados, o TabPFN se saiu melhor que os demais na discriminação das fraudes, porém, seu desempenho foi pior do que o observado no mesmo banco de dados sem balanceamento. Em particular, depreende-se da queda na precisão que o número de falsos positivos aumentou significativamente, ainda que o desempenho na classe negativa permaneça bom, de modo que se pode afirmar que, ao passo que o *oversampling* permite que ao menos um dos modelos aprenda a discriminar a classe positiva no cenário do banco de dados de transações bancárias, no segundo banco de dados ele prejudica o aprendizado da mesma classe em todos os modelos, conforme pode ser observado na Tabela 4.

Tabela 4 - Resultados para o banco de dados de fraudes no cartão de crédito balanceado com oversampling

Métricas	Regressão Logística	*XGBoost*	TabPFN
Acurácia	0,9821	0,9963	0,9988
ROC AUC	0,9716	0,9897	0,9757
F1-*score*	0,1526	0,4637	0,7143
Precisão	0,0831	0,3077	0,6000
Recall	0,9411	0,9412	0,8824
Tempo (s)	1,2269	154,6	4,5683

Fonte: Os autores

Nos testes seguintes, selecionando as 20.000 amostras dos bancos de dados e balanceando os dados de treinamento com o método *Near Miss*, novamente observou-se que, para o banco de dados de fraudes em transações, os modelos têm uma grande dificuldade de aprender as classes devido à redução do número de amostras causada pelo processo de *undersampling*.

No cenário experimental envolvendo o banco de dados de fraudes em transações bancárias, observa-se que todos os modelos têm dificuldade com a classe positiva, possuindo um número muito elevado de falsos positivos, o que diminui a precisão de todos eles para próximo de zero, apesar de que o *XGBoost* e, principalmente, o TabPFN, apresentam uma boa diferenciação entre falsos e verdadeiros negativos, conforme observado na Tabela 5, indicando

que o número de amostras da classe positiva (fraudes) é insuficiente para o aprendizado dos modelos.

Tabela 5 - Resultado para o banco de dados balanceado com *undersampling*

Métricas	Regressão Logística	*XGBoost*	TabPFN
Acurácia	0,9816	0,5230	0,9292
ROC AUC	0,8960	0,7708	0,9347
F1-*score*	0,1052	0,0033	0,0160
Precisão	0,0579	0,0019	0,0080
Recall	0,7622	0,8068	0,8680
Tempo (s)	0,7036	129,6	1,982

Fonte: Os autores

No experimento realizado com a base de dados de fraudes no cartão de crédito, observa-se que o desempenho dos modelos foi bem pior do que no caso de *oversampling*, apresentado na Tabela 4. Mais uma vez, os modelos apresentaram baixíssima precisão, não tendo a capacidade de aprender a classe positiva, ainda que consigam discriminar bem verdadeiros negativos de falsos negativos, como se observa na Tabela 6.

Tabela 6 - Resultado para o banco de dados de fraudes de cartão de crédito balanceado com *undersampling*

Métricas	Regressão Logística	*XGBoost*	TabPFN
Acurácia	0,9628	0,9857	0,9862
ROC AUC	0,9569	0,9747	0,9775
F1-*score*	0,0785	0,0772	0,1000
Precisão	0,0411	0,0404	0,0531
Recall	0,8962	0,9057	0,9063
Tempo (s)	1,121	134,7	3,209

Fonte: Os autores

Esses resultados, podem ser explicados com certa facilidade, já que o método de *undersampling* reduz o número de amostras disponíveis para treinamento, e como visto no primeiro experimento, cujos dados são disponibilizados na Tabela 1, o baixo número de amostras disponíveis para treinamento inviabiliza o aprendizado da classe positiva pelos modelos.

Em seguida, compararam-se os resultados dos testes realizados com os bancos de dados completos, sem amostragem, comparando a Regressão Logística, o *XGBoost* e o TabPFN com as modificações necessárias para acomodar todas as amostras dos bancos de dados: *subsampling*, *post-hoc ensemble* com *Random Forest* e extração de *embeddings*, os quais são apresentados nas Tabelas 7 e 8.

Tabela 7 - Resultado para o banco de dados completo

Métricas	**Regressão Logística**	***XGBoost***	**TabPFN *subsampling***	**TabPFN RF**	**TabPFN *embedded***
Acurácia	0,9991	0,9994	0,9988	0,9993	0,9993
ROC AUC	0,9499	0,9920	0,9731	0,9557	0,9541
F1-*score*	0,4927	0,6678	0,0000	0,5680	0,5535
Precisão	0,7837	0,9050	0,0000	0,9043	0,9727
Recall	0,4502	0,5308	0,0000	0,4172	0,3966
Tempo (s)	1,330	157,8	114,9	759,1	178,2

Fonte: Os autores

Tabela 8 - Resultado para o banco de dados de fraudes de cartão de crédito completo

Métricas	**Regressão Logística**	***XGBoost***	**TabPFN *subsample***	**TabPFN RF**	**TabPFN *embedded***
Acurácia	0,9990	0,9995	0,9983	0,9993	0,9994
ROC AUC	0,9007	0,9753	0,9569	0,9280	0,9443
F1-*score*	0,7071	0,8566	0,0000	0,6231	0,8284
Precisão	0,7075	0,9426	0,0000	0,9100	0,8657
Recall	0,7069	0,7854	0,0000	0,4759	0,7951
Tempo (s)	2,328	156,7	178,4	3363	109,7

Fonte: Os autores

Os experimentos conduzidos com os bancos de dados completos evidenciam limitações significativas dos modelos avaliados na identificação de fraudes em transações bancárias. Nesse contexto, o *XGBoost* apresentou desempenho superior em relação aos demais algoritmos, enquanto o TabPFN com *subsampling* demonstrou o pior resultado, não evidenciando capacidade efetiva de aprendizado. Ainda assim, observa-se que todos os modelos enfrentaram dificuldades substanciais na aprendizagem da classe positiva, refletidas em valores de *recall* relativamente baixos, mesmo nos casos em que a precisão foi elevada, como no *XGBoost* e nas variações do TabPFN com *post-hoc ensemble* e extração de *embeddings*. Como consequência, verificou-se uma elevada taxa de falsos negativos, com aproximadamente metade das transações fraudulentas sendo classificadas como não fraudulentas. Embora a Regressão Logística e o *XGBoost* tenham apresentado baixos índices de falsos positivos, sugerindo um aprendizado mais consistente da classe negativa, tal comportamento indica um viés sistemático em favor da classe majoritária. Assim, nesse cenário experimental, nenhum dos modelos pode ser considerado adequado para a identificação eficaz de fraudes.

Em contraste, no banco de dados de fraudes em cartões de crédito, observou-se uma melhora no desempenho geral dos modelos, com exceção do TabPFN com *subsampling*, que não apresentou capacidade de aprendizado. O *XGBoost* obteve o melhor desempenho, seguido pelo TabPFN com extração de *embeddings*. Ambos alcançaram elevada precisão e valores de

recall próximos de 80%, indicando uma capacidade efetiva de discriminar a classe positiva, diferentemente do que foi observado no primeiro conjunto de dados.

Quanto ao custo computacional, destaca-se que, embora o TabPFN com extração de *embeddings* permaneça mais rápido do que o *XGBoost*, este último mostrou-se aproximadamente 1,5 vezes mais lento. Em contrapartida, em cenários sem balanceamento e com amostragem, o *XGBoost* apresentou um tempo de processamento até 60 vezes superior. Esses resultados indicam que a aplicação de técnicas de pré-processamento para permitir que o TabPFN lide com volumes de dados superiores ao seu limite de pré-treino acarreta degradações relevantes no desempenho computacional, especialmente no tempo de processamento.

Por fim, a aplicação de técnicas de *oversampling* aos bancos de dados completos permitiu concluir os experimentos apenas no conjunto de dados de fraudes em transações bancárias. No caso do banco de dados de fraudes em cartões de crédito, o TabPFN não conseguiu acomodar o aumento no número de amostras geradas, mesmo com pré-processamento, o que resultou em falhas de execução devido a restrições de recursos computacionais. Assim, apenas os resultados do primeiro banco de dados puderam ser reportados.

Tabela 9 - Resultado para o banco de dados completo balanceado com *oversampling*

Métricas	**Regressão Logística**	***XGBoost***	**TabPFN *subsampling***	**TabPFN RF**	**TabPFN *embedded***
Acurácia	0,9274	0,9445	0,9934	0,9975	0,9982
ROC AUC	0,9257	0,9445	0,8517	0,9602	0,9541
F1-*score*	0,0285	0,2685	0,2085	0,3692	0,4665
Precisão	0,0145	0,1694	0,1222	0,2687	0,3679
Recall	0,8710	0,6452	0,7097	0,5935	0,6387
Tempo (s)	0,9947	163,9	113,8	652,2	61,63

Fonte: Os autores

Nota-se que o *oversampling* não permitiu que os modelos melhorassem seus respectivos desempenhos, inclusive, gerando a piora do desempenho em geral, o que ocorreu pela diminuição da precisão, mostrando que os modelos passaram a gerar mais falsos positivos do que no caso em que o banco de dados não foi balanceado, sugerindo que esse processo de balanceamento, para esse problema, atrapalha o processo de aprendizado dos modelos, que tendem a classificar mais amostras na classe positiva (fraude), que passou a se fazer mais frequente no treinamento devido ao *oversampling*, contudo, essa classificação está, na maioria das vezes, incorreta.

5. Considerações Finais

A presente pesquisa objetivou analisar o desempenho do modelo TabPFN em relação a modelos tradicionalmente empregados no mercado para a detecção de fraudes bancárias. Os resultados mostraram que, apesar do desbalanceamento de classes ser o principal problema para os modelos, os métodos de balanceamento empregados não foram suficientes para resolver essa problemática, podendo, inclusive, piorar o desempenho dos algoritmos. Ao avaliar o desempenho do TabPFN, conclui-se que ele é um método promissor, já que, de fato, foi capaz de competir com o principal algoritmo utilizado no mercado para a classificação de dados tabulares, o *XGBoost*. Porém, sua atual limitação, o limite de amostras estabelecido no pré-treino, cria uma limitação nos cenários nos quais ele pode ser efetivamente utilizado, inclusive os cenários de detecção de fraude, nos quais os modelos devem acomodar no treinamento milhões de amostras, correspondentes ao número de transações observadas pelos bancos diariamente. Deste modo, pesquisas futuras devem levar em consideração a expansão do seu limite de pré-treino.

REFERÊNCIAS

Banco Central do Brasil (BACEN). 2024. **O brasileiro e sua relação com o dinheiro**. Disponível em: <https://www.bcb.gov.br/content/cedulasemoedas/pesquisabrasileirodinheiro/Apresentacao_brasileiro_relacao_dinheiro_2024.pdf >. Acesso em: 17 jun. 2025.

CHAWLA, Nitesh V. et al. **SMOTE: synthetic minority over-sampling technique.** Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

FÁVERO, Luiz Paulo; BELFIORE, Patrícia. **Manual de análise de dados: Estatística e machine learning com Excel®, SPSS®, Stata®, R® e Python®**. 2ed.GEN, LTC, Rio de Janeiro, 2024.

GÉRON, A. **Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems**. 3ed. O'Reilly Media, Inc., 2022.

HOLLMANN, N.; MÜLLER, S.; EGGENSPERGER, K.; HUTTER, F. **TabPFN: A transformer that solves small tabular classification problems in a second**. arXiv preprint arXiv:2207.01848, 2022.

LIMA, Francisco Hércules de Holanda. **Fraude bancária: um estudo sobre a responsabilidade objetiva à luz do direito do consumidor**. 2025.

MARTINS, Emerson; GALEGALE, Napoleão Verardi. **Detecção de fraudes no segmento de crédito financeiro utilizando aprendizado de máquina: uma revisão da literatura**. Revista e-TECH: Tecnologias para Competitividade Industrial, v. 15, n. 3, 2022.

MÜLLER, S.; HOLLMANN, N.; ARANGO, S. P.; GRABOCKA, J.; HUTTER, F. **Transformers can do bayesian inference**. arXiv preprint arXiv:2112.10510, 2021.

NAGLER, T. **Statistical foundations of prior-data fitted networks**. In: International Conference on Machine Learning, 2023, Honolulu, Hawaii, USA. Anais… p. 25660-25676.

RUBAIDI, Zainab Saad; AMMAR, Boulbaba Ben; AOUICHA, Mohamed Ben. **Fraud detection using large-scale imbalance dataset.** International Journal on Artificial Intelligence Tools, v. 31, n. 08, p. 2250037, 2022.

SHWARTZ-ZIV, Ravid; ARMON, Amitai. **Tabular data: Deep learning is not all you need**. Information Fusion, v. 81, p. 84-90, 2022.

TANIMOTO, Akira et al. **Improving imbalanced classification using near-miss instances**. Expert Systems with Applications, v. 201, p. 117130, 2022.

TOSTA, Pedro Lucas Maranini; DIAS, Jonatas Cerqueira. **Detecção de fraudes em transações bancárias utilizando inteligência artificial**. Revista Processando o Saber, v. 17, p. 21-37, 2025.