Aplicação de Modelos de Aprendizado de Máquina para Detecção de Fraudes em Transações de Cartões de Crédito

João Paulo P. Dantas[1], **Ana Régia de M. Neves**[2]

[1]Eixo de Informação e Comunicação – Instituto Federal de Brasília (IFB)
Brasília – DF – Brasil

joaod3v@gmail.com[1], ana.neves@ifb.edu.br[2]

Abstract. *Fraud in credit card transactions in Brazil represents an estimated annual loss of 7 billion dollars. Fraud detection systems must check each transaction accurately and precisely. Current systems based only on rules are no longer able to face this problem. The application of machine learning techniques can learn transaction patterns and detect when one is legitimate or fraudulent with high precision and accuracy, contributing to reduce the scenario of high losses with credit cards. Thus, the objective of this research is, based on the results of the systematic review, to identify which machine learning technique can have the best possible performance in detecting fraud in credit card transactions.*

Resumo. *A fraude em transações de cartões de crédito no Brasil representa perdas anuais estimadas em 7 bilhões de dólares, pois os atuais sistemas de detecção de fraudes, baseados apenas em regras, não apresentam precisão e acurácia suficientes para a identificação das mesmas. As técnicas de aprendizado de máquina aplicadas nos sistemas de detecção de fraudes permitem que as máquinas assimilem padrões das transações e detectem sua legitimidade com precisão e acurácia e neste trabalho será explorado se essas técnicas são capazes de contribuir para redução dos elevados prejuízos enfrentados pelas instituições que administram os cartões de crédito no Brasil. Assim, o objetivo desta pesquisa é de, com base nos resultados de uma revisão sistemática, identificar qual melhor técnica de aprendizado de máquina poderá ter maior desempenho na detecção de fraudes nesse contexto.*

1. Introdução

A expansão da Internet possibilitou a migração de diversos modelos de negócios para o ambiente virtual transformando o cartão de crédito em um dos principais meios de pagamento em transações comerciais adotados por consumidores no mundo todo. Porém, a facilidade em sua utilização gerou uma janela de oportunidade para que criminosos explorassem as fragilidades desse meio de pagamento praticando diversos tipos de fraudes.

Fraudes em cartões de crédito ocorrem quando o cartão é utilizado por terceiros sem a autorização do titular. Segundo [Can et al. 2020], destacam-se cinco tipos de fraudes, que são: (i) roubo simples, ocorre quando o cartão físico é roubado; (ii) fraude na aquisição, ocorre quando um cartão é adquirido com base em documentos falsos; (iii) fraude de insolvência, ocorre quando o titular continua utilizando o cartão mesmo sem

condições de honrar o dispêndio realizado; (iv) fraude interna, ocorre quando empregados da instituição roubam informações do cartão e o utilizam indevidamente; e (v) fraude de comportamento, ocorre de maneira não presencial quando os fraudadores obtêm os dados do cartão por *phishing* ou aquisição de dados do titular e o utilizam para realizar compras sem autorização do mesmo.

De acordo com [Tingfei et al. 2020], a fraude global em cartões de crédito saltou de 9,84 bilhões de dólares em 2011 para 27,69 bilhões de dólares em 2017, um crescimento de mais de 180% no período. No Brasil, a estimativa de prejuízo das instituições é de 7 bilhões de dólares por ano [Paul et al. 2020].

Identificar uma transação fraudulenta em meio à milhares de transações legítimas é um processo que demanda grandes esforços quando realizada manualmente. Segundo [Can et al. 2020], as soluções tradicionais de detecção de fraudes são baseadas em sistemas de regras, em que são estipulados parâmetros de monitoramento de variáveis pré-selecionadas e é emitido um alerta para análise humana quando estes parâmetros são atingidos.

[Bagga et al. 2020] e [Misra et al. 2020] afirmam que devido ao elevado volume das transações eletrônicas, a maneira tradicional de detecção de fraudes demanda muito tempo para análise, não é escalável e possui baixa acurácia. Segundo [Dornadula and Geetha 2019], mesmo com o incremento de novos artifícios tecnológicos, tais como os cartões com chip, os fraudadores adaptam-se e mudam o perfil da fraude.

Segundo [Paul et al. 2020], novos métodos para validação das transações são necessários para garantir mais segurança, com isso, os métodos estatísticos e de aprendizado de máquina destacam-se no reconhecimento de fraudes de maneira mais precisa, pois processam de forma eficiente elevados volumes de dados. Tais modelos utilizam dados do usuário, da transação e dos metadados do equipamento como insumos para recomendar a aprovação ou não de uma transação. Desta forma, sistemas de detecção de fraude baseados em aprendizado de máquina contribuem para que estas transações possam ocorrer com o menor risco possível, evitando prejuízos para clientes e instituições.

Neste contexto, o objetivo deste trabalho é de identificar qual técnica de aprendizado de máquina poderá ter melhor resultado na detecção de fraudes em cartões de crédito.

As demais seções deste trabalho estão organizadas como segue: a Seção 2 apresenta a revisão sistemática da literatura dos conteúdos de interesse deste trabalho; a Seção 3 apresenta as principais características da base de dados escolhida, análise exploratória dos dados e construção de modelos; a Seção 4 são apresentados os resultados obtidos; a Seção 5 apresenta conclusões desta pesquisa e encaminhamentos para trabalhos futuros.

2. Revisão Sistemática da Literatura

Esta pesquisa é fundamentado de acordo com a revisão sistemática da literatura e direcionado pela seguinte questão: “Qual técnica de aprendizado de máquina poderá ter melhor desempenho na detecção de fraudes em transações de cartões de créditos?”. Com base nessa pesquisa, será possível avaliar a possibilidade de implementação de um sistema real de monitoramento de fraudes com base em aprendizado de máquina em uma instituição financeira.

As buscas foram baseadas no título e resumo dos trabalhos, e ocorreram no período de abril até maio de 2021. Os critérios de inclusão e exclusão são descritos na Tabela 1. As fontes onde as buscas foram realizadas e o conjunto de *strings* utilizadas são apresentados na Tabela 2.

Tabela 1. Critérios definidos para a revisão sistemática

Critérios de Inclusão	Critérios de Exclusão
Detecção de fraudes em transações de cartões de crédito com aplicação de técnicas de aprendizagem de máquina	Detecção de fraudes em transações de cartões de crédito sem aplicação de técnicas de aprendizagem de máquina

Tabela 2. Fontes e *Strings* de busca utilizadas

Base de dados	Palavra-chave	Resultados
IEEE XPLORE	(Machine Learning) AND (Credit Card) AND (Fraud Detection) - Open Access	11
Science Direct	(Machine Learning) AND (Credit Card) AND (Fraud Detection) - Open Access	76
Google Scholar	(Aprendizado de Máquina) AND (Detecção de Fraude) AND (Cartão de Crédito)	27

As palavras de busca utilizadas nas bases *IEEE XPLORE* e *Science Direct* não retornaram resultados quando consultadas em português por isso não foram compiladas na Tabela 2. Também por essa mesma razão, utilizou-se palavra-chave exclusivamente em português na busca do *Google Scholar*.

2.1. Resultados da Revisão Sistemática da Literatura

Na organização da pesquisa sistemática foi utilizado o aplicativo *Rayyan* [1] e os resultados da revisão são apresentados no Diagrama Prisma, conforme Figura 1. No total foram recuperados cento e quatorze trabalhos das bases *IEEE Xplore*, *Science Direct* e *Google Scholar*. Destes, nove foram considerados para síntese final e três foram considerados como trabalhos correlatos, Seção 2.2, pois demonstraram todo o processo de aplicação das técnicas de aprendizado de máquina.

Dentre os trabalhos analisados cerca de 63% utilizaram as seguintes técnicas de aprendizado de máquina:

- *Support Vector Machine* (SVM), tem por objetivo encontrar o melhor hiperplano que separa de maneira otimizada dois conjuntos de pontos em classes distintas [Makki et al. 2019]. Segundo [Rtayli and Enneya 2020], nesta técnica os atributos de um conjunto de dados são transformados em vetores em um hiperplano,

[1]Disponível em https://rayyan.ai. Os resultados obtidos das três bases de dados da revisão sistemática no formato *BibTex* foram organizados nesta ferramenta que possibilita a verificação de trabalhos duplicados, categorização e leitura rápida dos metadados de uma grande quantidade de artigos.

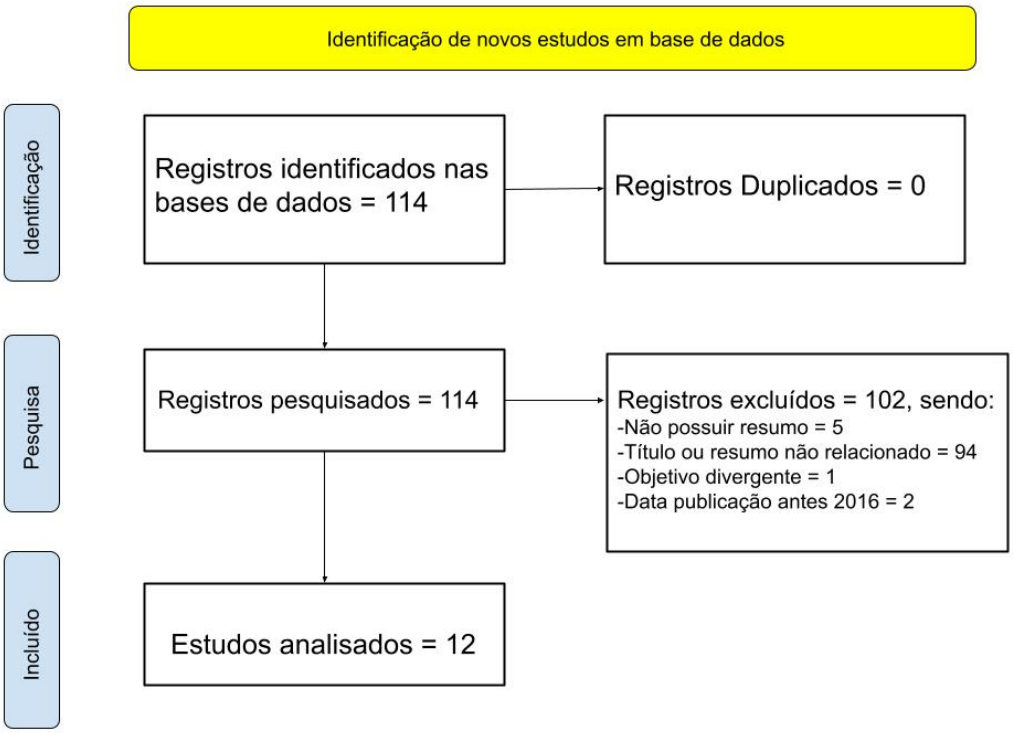

Figura 1. Diagrama Prisma da Revisão Sistemática da Literatura

assim, cada vetor é representado como um ponto em planos de coordenadas de k dimensões. Desta forma, o SVM será capaz de executar uma função que busca minimizar a distância entre os pontos comuns a fim de maximizar a distância entre os planos que contêm cada tipo de classificação;

- *Decision Tree* usa da analogia de uma árvore composta por galhos e folhas para demonstrar um conjunto de regras e seus resultados [Makki et al. 2019]. Nesta técnica, o galho representa um nó em que um atributo é testado por uma regra. O resultado da aplicação da regra é utilizado como insumo pelo novo nó e, assim, sucessivamente, até que esgote todas as possibilidades de regras gerando um resultado final ou, segundo a analogia, uma folha;
- *K-Nearest Neighbors* (KNN), segundo [Makki et al. 2019], cria uma função classificadora por meio de um sistema de votação baseado nos pontos mais próximos de um determinado centro. Suponha que o parâmetro k seja cinco para o número de vizinhos e a forma de calcular a distância entre eles seja a euclidiana. O classificador do KNN encontrará as cinco amostras mais próximas de uma transação. Dessa forma, será possível classificar se uma transação pertence a uma classe ou outra com base na menor distância possível entre seus vizinhos;
- *Logistic Regression* é um modelo de generalização linear, sendo o objetivo encontrar a probabilidade de uma transação ser de uma determinada classe. Cada atributo será multiplicado por um coeficiente e o resultado desses produtos serão somados a probabilidade de erro e a um coeficiente constate, assim será possível estimar o risco da fraude;
- *Multi-Layer Perceptron* é uma estrutura formada por três ou mais camadas com-

postas por neurônios totalmente interconectados [Can et al. 2020], os quais utilizam de uma função de ativação de dados para realizar testes gerando resultados que são reinseridos no sistema, formando ciclos de retro-propagação, até a realização do aprendizado pela máquina;
- *Naive Bayes*, nela cada atributo é condicionalmente independente dos demais [Makki et al. 2019], o que gera uma dificuldade em definir classes em dados desbalanceados, uma vez que, ela tendencia classes majoritárias;
- e, *Random Forest* cria um conjunto aleatório de árvores de decisão, a partir do número de árvores escolhido pelo usuário [Randhawa et al. 2018]. Posteriormente utiliza de um sistema de votação para determinar qual das árvores do conjunto gerado representa a classificação de uma determinada transação.

[Taha and Malebary 2020] apresentaram em sua revisão da literatura que as técnicas de aprendizado de máquina na detecção de fraudes em cartões de crédito mais populares são as supervisionadas e que as transações devem ser divididas entre, no mínimo, duas classes já previamente rotuladas. O aprendizado supervisionado de máquina necessita que o conjunto de dados seja organizado em um conjunto de treino e em conjunto de teste.

Após a assimilação das características das transações legítimas e das transações fraudulentas no conjunto de testes, será possível a técnica classificar, em níveis satisfatórios, transações nunca vistas do conjunto de testes em classe negativa ou legítima e classe positiva ou fraude. Entretanto, para 75% dos autores, a aplicação de qualquer uma das técnicas sem um tratamento prévio nos dados não será capaz de classificar corretamente uma transação da classe positiva já que o volume de transações dela é muito pequeno, ou seja, o desbalanceamento entre a classe positiva e classe negativa é considerado o maior problema na detecção de fraudes de cartões de crédito.

Para [Dornadula and Geetha 2019], [Can et al. 2020], [Misra et al. 2020], [Paul et al. 2020], [Bagga et al. 2020], [Rtayli and Enneya 2020], [Taha and Malebary 2020], [RB and KR 2021] e [Tingfei et al. 2020] as técnicas mais usadas no pré-processamento são: *Synthetic Minority Oversampling Technique* (SMOTE) e a redução de dimensionalidade de atributos. Segundo [Tingfei et al. 2020], o SMOTE usa o KNN para gerar dados sintéticos a partir dos dados da classe minoritária assimilando seu padrão. Segundo [Rtayli and Enneya 2020], quanto maior o número de atributos não redundantes e que sejam relevantes, melhor será o desempenho do sistema de classificação de fraudes.

Por fim, a última etapa de um processo de aprendizado de máquina é a mensuração dos resultados obtidos por cada uma das técnica utilizadas, de modo que seja possível comparar o desempenho entre elas e identificar suas vantagens e desvantagens. Quase 70% dos trabalhados recuperados utilizaram as seguintes métricas de avaliação: acurácia, sensibilidade ou revocação, precisão e *F-score*.

Para as métricas serão utilizadas a seguintes siglas a definir: verdadeiro positivo (VP), que é a predição correta da fraude; falso positivo (FP), que é a predição incorreta da fraude; verdadeiro negativo (VN), que é a predição correta da não fraude; e, por fim, falso negativo (FN), que é a predição incorreta da não fraude.

Para [Paul et al. 2020], a precisão é a "razão das transações fraudulentas em rela-

ção ao total de transações que o modelo previu como fraudulentas" ou ainda

$$\frac{VP}{VP + FP}$$

A sensibilidade "é a proporção de casos previstos como positivos em relação ao total de casos positivos" ou ainda

$$\frac{VP}{VP + FN}$$

E, o *F1-score* é "a média harmônica da precisão e sensibilidade obtidas" ou ainda

$$2 * \frac{\text{sensibilidade} * \text{precisão}}{\text{sensibilidade} + \text{precisão}}$$

Para [Bagga et al. 2020], a acurácia é a razão entre número das predições corretas dividido pelo total de predições ou ainda

$$\frac{VP * VN}{TP + TN + FP + FN}$$

2.2. Trabalhos Correlatos

Segundo [Paul et al. 2020] compararam o *KNN*, *Random Forest* e *Gradient Boosting* após aplicação de técnica de pré-processamento t-SNE e obtiveram que a maior precisão foi no KNN, porém com baixa sensibilidade e o *Random Forest* apresentou o maior *F-score*.

Já [Bagga et al. 2020] aplicaram uma técnica de *oversampling* conhecida como Adaptativa Synthetic (ADASYN) e propõem a utilização do *Pipelining* e *Esemble Learning* para obter melhores resultados na classificação de fraudes comparados a KNN, *Logistic Regression*, *Random Forest*, *Naive Bayes*, *Multilayer Perceptron*, *Adaboost* e *Quadrant Discriminative Analysis*. Os métodos *Pipelining* e *Esemble Learning* obtiveram melhores resultados quando comparados as demais técnicas de aprendizado de máquina, sendo a acurácia a métrica de maior destaque.

Por fim, [RB and KR 2021] comparam o uso de KNN, *Artificial Neural Network* (ANN) e *Support Vector Machine* (SVM), sendo utilizada como técnica de pré-processamento a normalização e a redução de escala nos dados após aplicação de *undersampling* na classe negativa. Os resultados demonstraram que o método ANN e KNN obtiveram 99% de acurácia, porém, ao custo de perda de sensibilidade e precisão. O SVM é o classificador que se destaca no experimento por seu desempenho melhor em precisão e sensibilidade, sendo a acurácia menor que o demais, mas ainda em nível aceitável de 93%.

3. Material e Métodos

Esta pesquisa é aplicada e utiliza a mesma estrutura dos trabalhos correlatos, organizada em cinco fases, a saber:

1. análise exploratória dos dados;
2. preparação dos dados para construção dos modelos
3. aplicação de técnicas de pré-processamento;
4. construção dos modelos com base nas técnicas selecionadas; e,
5. avaliação e comparação de resultados.

3.1. Características da base de dados

A base de dados foi coletada em setembro de 2021 no site de competições de Aprendizado de Máquina *Kaggle*[2] e foi disponibilizada pela *Université Libre de Bruxelles* (ULB) contendo 2 dias de transações de cartões de créditos europeus que realizaram compras em 2013.

A base possui 284.807 transações e 31 atributos, quais sejam: (i) V1; (ii) V2; (iii) V3; (iv) V4; (v) V5; (vi) V6; (vii) V7; (viii) V8; (ix) V9; (x) V10; (xi) V11; (xii) V12; (xiii) V13; (xiv) V14; (xv) V15; (xvi) V16; (xvii) V17; (xviii) V18; (xix) V19; (xx) V20; (xxi) V21; (xxii) V22; (xxiii) V23; (xxiv) V24; (xxv) V25; (xxvi) V26; (xxvii) V27; (xxviii) V28; (xxix) *Time*; (xxx) *Amount*; e, (xxxi) *Class* .

Os atributos V1 até V28 foram transformados pela técnica Análise de Componentes Principais(ACP) de modo a garantir confidencialidade dos titulares e não foram explicadas pela *ULB* o que representavam. O atributo *Time* representa o tempo decorrido entre cada transação desde a primeira transação da base de dados. O Atributo *Amount* representa o valor efetivamente gasto na compra. Por fim, o atributo *Class* representa a variável de resposta sendo: "1"fraude (classe positiva) e "0"não-fraude (classe negativa).

Todos os atributos são do tipo *float64*, com exceção do atributo *Class* que é do tipo *int64*. A base não possui dados nulos ou vazios.

3.2. Análise Exploratória dos Dados

A exploração dos dados, o pré-processamento, a construção dos modelos preditivos e a análise de resultados foi realizada na ferramenta *Google Colaboratory*[3] (Colab). Nesse ambiente foi criado um *notebook* na linguagem de programação *Python* e foram utilizadas as seguintes bibliotecas para importação e análise exploratórias dos dados, a saber: *pandas*[4], *matplotlib*[5], *seaborn*[6] e numpy[7].

Na Figura 2, obseva-se um resumo de *boxplots* para os atributos da base, exceto para *Class*. Os *outliers* foram ocultados por distorcerem a proporção do gráfico. O uso dos *boxplots* justifica-se nesta etapa devido a necessidade de visualização da dispersão dos dados, ou seja, identificar a variabilidade nas amostras dos dados por meio das medidas de posição, como mediana e quartis. O *boxplot* apresenta visualmente essas informações, simplifica o processo de comparação e análise da dispersão para cada uma das variáveis. Por exemplo, na Figura 2, observa-se que a mediana dos atributos de V1 até V28 tendem a 0 (zero), porém, a análise dos valores máximos e mínimos e valores interquartis de cada atributo mostram grande dispersão dos dados para cada um dos atributos.

---

[2] `https://www.kaggle.com/mlg-ulb/creditcardfraud`

[3] `https://colab.research.google.com/`

[4] Biblioteca utilizada para analisar e manipular dados, disponível em `https://pandas.pydata.org/`

[5] Biblioteca utilizada para analisar e visualizar, disponível em `https://matplotlib.org/`

[6] Biblioteca derivada da *matplotlib* utilizada para customizar a visualização de dados, disponível em `https://seaborn.pydata.org/`

[7] Biblioteca utilizada para manipulação de dados `https://numpy.org/`

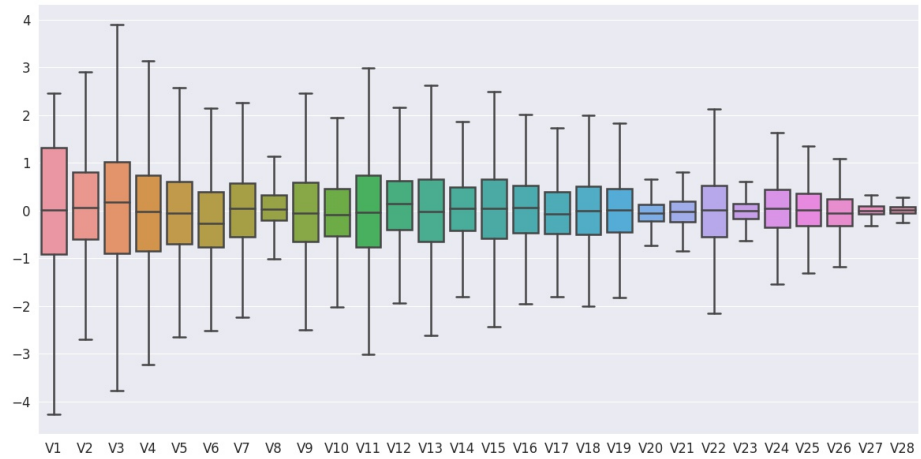

Figura 2. *Boxplot* dos atributos transformados pelo ACP

Na Figura 3, observa-se que cada variável tem características únicas, conforme análise a seguir: o atributo *Time* possui dados apenas positivos, o que é esperado uma vez que o tempo em segundos é acumulativo entre cada transação desde a primeira transação; e, o atributo *Amount*, observa-se que, além de possuir grande variação dos dados, a metade das compras realizadas são de valores pequenos (menor que 25 unidades monetárias).

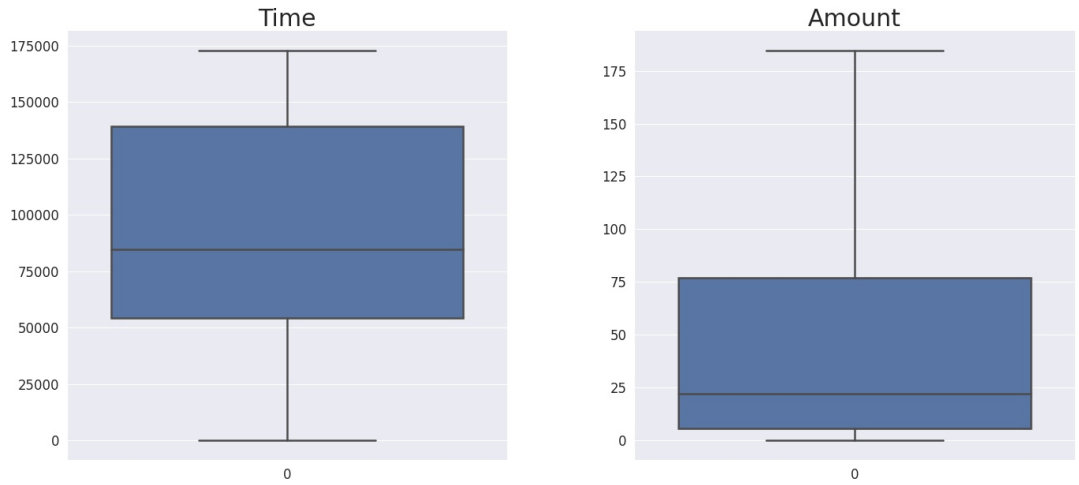

Figura 3. *Boxplot* dos atributos Time e Amount

Na Figura 4, é apresentado o histograma do atributo *Class* no qual observa-se elevado desbalanceamento entre as classes positiva e negativa. Sendo que as transações fraudulentas, classe positiva, representam apenas 0,17% de todas as transações.

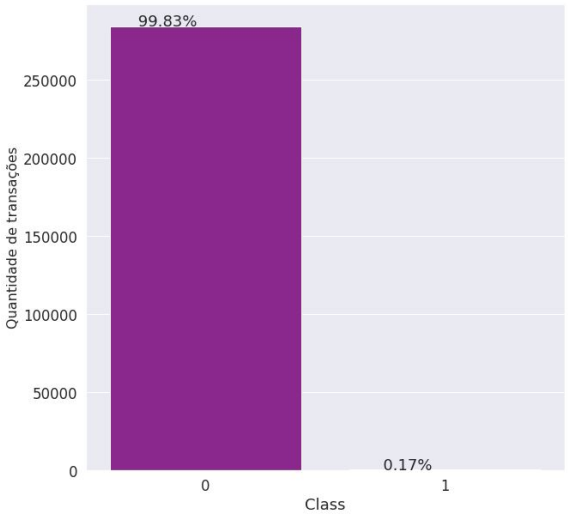

Figura 4. Histograma do atributo *Class*

3.3. Pré-processamento

Como descrito anteriormente, das 284.807 observações da base original apenas 0,17% correspondem a transações fraudulentas (classe positiva). Para tarefas de classificação, esse tipo de problema gera modelos tendenciosos para a previsão da classe majoritária. Desse modo, antes da etapa de treinamento, deve-se reduzir o desbalanceamento entre as classes. Uma abordagem que permite a criação de dados artificiais para a classe minoritária é denominada de *oversampling* segundo [Dornadula and Geetha 2019].

Neste trabalho, foi aplicada a técnica de *oversampling Synthetic Minority Oversampling Techinique* (SMOTE) da biblioteca *imblearn.over_sampling* [8]. Como resultado, obteve-se uma nova base de dados com 568.630 registros organizados nos mesmos 31 atributos, porém, nessa nova base de dados metade das transações são fraudulentas e a outra metade são de transações legítimas.

Dois hiperparâmetros foram utilizados na modelagem do SMOTE, a saber: (i) *random_state*, para controlar a aleatoriedade do algoritmo; e (ii) *neighbors*, o qual define a quantidade de amostras que será coletada para a criação dos dados artificiais da classe fraude. Os valores configurados foram 7 e 3, respectivamente.

3.4. Construção dos modelos

O próximo passo, após o tratamento do dados, é a construção dos modelos e foram escolhidos os principais modelos identificados nos trabalhos correlatos. A seguir, são apresentados os modelos e as bibliotecas correspondentes disponíveis no *Scikit-learn*[9]: (i) modelo *Logistic Regression*, biblioteca *linear_model*; (ii) modelo *Decision Tree*, biblioteca *tree*; (iii) modelo *Random Forest*, biblioteca *ensemble*; (iv) modelo KNN, biblioteca *neighbors*; (v) modelo Naive Bayes, biblioteca *naive_bayes*; (vi) modelo *Support Vector Machine*, biblioteca *svm*; (vii) modelo *Neural Network*, biblioteca *neural_network*.

[8]Biblioteca derivada da *Scikit-learn* utilizada para tratamento de bases desbalanceadas, disponível em `https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html`

[9]Conjunto de ferramentas utilizadas para predição de dados, disponível em `https://scikit-learn.org/`

De modo a controlar a aleatoriedade e permitir sua reprodução em trabalhos futuros, estabeleceu-se como *random_state*, ou semente padrão, o valor 7. Nesta etapa, estabeleceu-se que a semente padrão de todos os modelos será igual a 7. Desenvolveu-se um algoritmo de laço de repetição que faz modelagem, predição e avaliação dos resultados para cada tipo de técnica de aprendizado de máquina. Esse algoritmo foi utilizado para criar 4 agrupamentos de modelos, a saber: (i) modelos sem alteração dos hiperparâmetros padrão e sem utilização do SMOTE; (ii) modelos com hiperparâmetros alterados e sem utilização do SMOTE; (iii) modelos sem alteração dos hiperparâmetros e com utilização do SMOTE, e; (iv) modelos com hiperparâmetros alterados e com utilização do SMOTE.

Esses quatro tipos de modelagem permitem que os dados possam ser comparados e avaliados de modo a identificar se a técnica *SMOTE* tem algum tipo de impacto nos resultados obtidos. Em todos os modelos foi utiliza a proporção de 70% dos dados para aprendizado e 30% para testes, mesmo para a base sintética gerado pela aplicação da técnica de pré-processamento.

A Tabela 3 sintetiza os hiperparâmetros utilizados para cada tipo de técnica.

Tabela 3. Hiperparâmetros utilizados nas Técnicas de Aprendizado de Máquina

Técnica	Parâmetro 1	Parâmetro 2	Parâmetro 3
Logistic Regression	*solver* = saga	-	-
KNN	*n_neigbors* = 3	-	-
Decision Tree	*criterion = entropy*	*max_features = sqrt*	-
Random Forest	*bootstrap = False*	*criterion = entropy*	*max_features* = 3
Naive Bayes	*var_smoothing* = 0,10987546321	-	-
Support Vector Machine	*kernel = linear*	*max_iter* = 3	-
Multi-Layer Perceptron	*hidden_layer_sizes* = 3	*max_iter* = 50	-

No modelo *Logistic Regression*, utilizou-se o hiperparâmetros *solver* com valor igual a *saga* que é o mais indicado pela própria documentação para base de dados grandes tal como a utilizado nesta pesquisa.

No modelo *KNN* sem aplicação *SMOTE*, utilizou-se o hiperparâmetro *n_neigbors* com valor igual 3, uma vez que existem poucas amostras de transações fraudulentas. Para fins de comparação, na base com aplicação de *SMOTE* também utilizou-se o mesmo valor de hiperparâmetro.

No modelo *Decision Tree*, utilizou-se os seguintes hiperparâmetros: *criterion* com valor igual a *entropy* uma vez que o padrão da biblioteca utiliza o valor *gini*; e, *max_features* com valor igual a *sqrt* que busca otimizar a aprendizagem do modelo ao reduzir o número de atributos utilizados na modelagem.

No modelo *Random Forest*, utilizou-se os seguintes hiperparâmetros: *bootstrap* como valor igual *False* para forçar a utilização de todos os atributos na modelagem; *criterion* com valor igual *entropy* para verificar outra opção fora do padrão; e, *max_features* com valor igual a 3 uma vez que existem poucas amostras de transações fraudulentas.

No modelo *Naive Bayes*, utilizou-se o hiperparâmetro *var_smoothing* com valor igual 0,10987546321 pois o valor padrão da biblioteca é de 1e-09 e observa-se da base de dados da *ULB* que a mesma possui uma média de variância próxima ao valor escolhido.

No modelo *Support Vector Machine*, utilizou-se os seguintes hiperparâmetros: *kernel* como valor igual *linear* como uma tentativa de alterar o valor; e, *max_iter* com valor igual 3 como uma tentativa de reduzir o tempo de execução do processo de aprendizado do modelo.

No modelo Multi-Layer Perceptron, utilizou-se os seguintes hiperparâmetros: *hidden_layer_sizes* como valor igual 3 e o parâmetro *max_iter* com valor igual 50, ambos foram alterados como uma tentativa de reduzir o tempo de execução do processo de aprendizado do modelo.

4. Resultados e Discussão

Nesta etapa da pesquisa, apresentam-se os resultados obtidos referentes aos experimentos realizados e que estão organizados nas Tabelas 4, 5, 6, 7.

Em cada tabela, as técnicas foram comparadas entre si tendo por base o *F1-score* e destacadas em cores diferentes conforme resultado, a saber: cor verde para a técnica com melhor resultado e cor vermelha para a técnica com pior resultado. Na Tabela 6 foi utilizada a cor cinza para destacar o caso de um modelo que não foi possível obter resultados devido seu tempo de execução inviabilizar a modelagem.

Tabela 4. Métricas dos modelos criados sem alteração dos hiperparâmetros e sem SMOTE

Modelo	Acurácia	Precisão	Sensibilidade	*F1-score*
Logistic Regression	0,999040	0,729032	0,738562	0,733766
KNN	0,998268	0,051613	0,888889	0,097561
Decision Tree	0,999239	0,812903	0,777778	0,794953
Random Forest	0,999567	0,825806	0,927536	0,873720
Naive Bayes	0,993083	0,677419	0,162539	0,262172
Support Vector Machine	0,998186	0,000000	0,000000	0,000000
Multi-Layer Perceptron	0,998631	0,858065	0,583333	0,694517

Tabela 5. Métricas dos modelos criados com alteração dos hiperparâmetros e sem SMOTE

Modelo	Acurácia	Precisão	Sensibilidade	*F1-score*
Logistic Regression	0,998186	0,000000	0,000000	0,000000
KNN	0,998268	0,096774	0,652174	0,168539
Decision Tree	0,999228	0,838710	0,760234	0,797546
Random Forest	0,999625	0,845161	0,942446	0,891156
Naive Bayes	0,998186	0,000000	0,000000	0,000000
Support Vector Machine	0,348302	0,548387	0,001526	0,003044
Multi-Layer Perceptron	0,998186	0,000000	0,000000	0,000000

Tabela 6. Métricas dos modelos criados sem alteração dos hiperparâmetros e com SMOTE

Modelo	Acurácia	Precisão	Sensibilidade	*F1-score*
Logistic Regression	0,979811	0,922581	0,077047	0,142218
KNN	0,966843	0,993548	0,051574	0,098058
Decision Tree	0,999064	0,993548	0,660944	0,793814
Random Forest	0,999918	1,000000	0,956790	0,977918
Naive Bayes	0,992709	0,800000	0,173184	0,284730
Support Vector Machine	-	-	-	-
Multi-Layer Perceptron	0,977587	0,929032	0,070312	0,130731

Tabela 7. Métricas dos modelos criados com alteração dos hiperparâmetros e com SMOTE

Modelo	Acurácia	Precisão	Sensibilidade	*F1-score*
Logistic Regression	0,950564	0,593548	0,021632	0,041742
KNN	0,978360	1,000000	0,077345	0,143585
Decision Tree	0,999134	0,993548	0,678414	0,806283
Random Forest	0,999941	1,000000	0,968750	0,984127
Naive Bayes	0,493030	0,541935	0,001939	0,003863
Support Vector Machine	0,956369	0,903226	0,036335	0,069860
Multi-Layer Perceptron	0,989221	0,922581	0,135932	0,236951

A técnica *Logistic Regression* obteve o melhor resultado na versão sem alteração dos hiperparâmetros e sem *SMOTE*, Tabela 4, com um *F1-score* de 73%. A alteração nos hiperparâmetros ou aplicação do pré-processamento não contribuiu na melhoria dos resultados, conforme apresentado nas Tabelas 5, 6 e 7.

A técnica *KNN* obteve o melhor resultado na versão com alteração dos hiperparâmetros e sem *SMOTE*, Tabela 5, com um *F1-score* de 16%. A alteração nos hiperparâmetros melhorou os resultados em ambas versões com e sem *SMOTE*, Tabelas 5 e 7, porém, mesmo com a pequena melhoria essa técnica não seria apropriada para um sistema de detecção de fraudes dado sua baixa capacidade de identificar fraudes (sensibilidade) e sua baixa capacidade de detectar transações fraudes corretamente (precisão).

A técnica *Decision Tree* obteve o melhor resultado na versão com alteração dos hiperparâmetros e com *SMOTE*, Tabela 7, com um *F1-score* de 80,62%. Observa-se que a alteração dos parâmetros e a utilização da técnica de pré-processamento incrementaram os resultados apresentados. Em geral, todos os modelos apresentaram bom desempenho para um sistema de detecção de fraudes uma vez que o modelo faria predições de fraude e não fraude corretamento em 99,91% dos casos (acurácia), detectaria transações fraudulentas em 99,35% dos casos (precisão) e detectaria 67,84% do total de casos de fraude (sensibilidade).

A técnica *Random Forest* obteve o melhor resultado na versão com alteração dos hiperparâmetros e com *SMOTE*, Tabela 7, com um *F1-score* de 98,41%. Observa-se que a alteração dos parâmetros e a utilização da técnica de pré-processamento incrementaram os resultados apresentados. Em geral, todos os modelos apresentaram bom desempenho para um sistema de detecção de fraudes uma vez que o modelo faria predições de fraude e não fraude corretamento em 99,99% dos casos (acurácia), detectaria transações fraudulentas em 100% dos casos (precisão) e detectaria 96,87% do total de casos de fraude (sensibilidade).

A técnica *Naive Bayes Forest* obteve o melhor resultado na versão sem alteração dos hiperparâmetros e com *SMOTE*, Tabela 6, com um *F1-score* de 28,47%. Observase que a alteração dos parâmetros piorou consideravelmente os resultados e a utilização da técnica de pré-processamento pouco contribuiu para melhoria do *F1-score*. Em geral, todos os modelos apresentaram baixo desempenho para um sistema de detecção de fraudes uma vez que o modelo faria poucas detecções 17,31% do total de casos de fraude (sensibilidade).

A técnica *Support Vector Machine* obteve o melhor resultado na versão com alteração dos hiperparâmetros e com *SMOTE*, Tabela 7, com um *F1-score* de 6,98%. Conforme demonstrado na Tabela 6, o experimento sem alteração de hiperparâmetros e com *SMOTE* não foi possível obter resultado uma vez que o tempo de execução gastou mais de 2 horas, situação na qual perdia-se a conexão e não permitiu a aferição do resultado.

Por fim, a técnica *Multi-Layer Perceptron* obteve o melhor resultado na versão sem alteração dos hiperparâmetros e sem *SMOTE*, Tabela 4, com um *F1-score* de 69,45%. Os demais modelos obtiveram desempenho insuficiente para serem cogitados sua aplicação em uma sistema de detecção de fraudes.

Resumidamente, as técnicas a seguir obtiveram melhoria com alterações dos hiperparâmetros e uso do *SMOTE*: *Decision Tree*, *Random Forest* e *Support Vector Ma-*

chine. Já as seguintes técnicas não melhoraram seus resultados alterando os hiperparâmetros ou aplicando *SMOTE*: *Logistic Regression* e *Multi-Layer Perceptron*. Por fim, a técnica *KNN* melhorou alterando apenas seu hiperparâmetro e a técnica *Naive Bayes* melhorou apenas aplicando o SMOTE.

5. Considerações Finais e Trabalhos futuros

Esta pesquisa justificou-se devido ao alto volume de fraudes em cartões de crédito no mundo e no Brasil, sendo uma possibilidade da causa desse cenário a utilização de sistemas de detecção de fraudes baseados em modelos de regras tratados por pessoas. Com isso, este trabalhou apresentou uma possível alternativa para aplicação em sistemas de detecção de fraudes baseados em aprendizado de máquina, pois essa abordagem poderia contribuir para identificar mais fraudes de modo mais preciso.

O presente trabalho fundamentou-se na revisão sistemática da literatura e por meio de sua metodologia de pesquisa buscou-se atingir o objetivo a partir questão norteadora. Nesta etapa, foi realizada pesquisa em três bases de dados por trabalhos utilizando os critérios estabelecidos que permitissem identificar estudos que utilizassem modelos de aprendizado de máquina na detecção de fraudes em transações de cartões de crédito.

Devido dificuldades em localizar outras pesquisas em língua portuguesa e bases de dados reais com todos os atributos, foram utilizadas, principalmente, pesquisas em língua inglesa e fonte de dado anonimizada. Apesar dessa situação, foi possível aplicar todo o *pipeline* de aprendizado de máquina em uma base de dados desbalanceada e observar seus resultados.

Foram utilizadas 7 técnicas de aprendizado de máquina para criação de modelos classificadores de fraudes e buscou-se compará-los a fim de verificar quais deles apresentariam melhores resultados. Tal como apresentado nos trabalhos correlatos, foi aplicado a técnica *SMOTE* na base de dados de modo a tratar o desbalanceamento entre transações fraudulentas e legítimas. Observou-se que aplicação dessa técnica associada a alteração de hiperparâmetros contribuiu na melhoria dos resultados de 3 das 7 técnicas, sendo que entre todas as técnicas, destacou-se a técnica *Random Forest* com resultado de *F1-score* de 98%.

A aplicação de ajustes finos em hiperparâmetros nas técnicas de aprendizado de máquina podem ou não resultar na melhoria de resultados. A pesquisa demonstrou as duas situações e, para trabalhos futuros, sugere-se o estudo aprofundado de cada hiperparâmetro de cada técnica. Essa etapa será fundamental para melhorar os resultados das técnicas. Como também, sugere-se a utilização de técnicas de automatização de testes que realizem várias combinações de hiperparâmetros de modo a identificar o melhor conjunto de hiperparâmetros a ser utilizado para determinada técnica.

A fraude em cartões de créditos representa apenas uma pequena fração em uma base de dados e esse desbalanceamento necessita de tratamento, sendo que a aplicação de uma ou um conjunto de técnicas de pré-processamento é fundamental para contribuir com modelos mais precisos. Nesse sentindo, sugere-se a aplicação de técnicas de pré-processamento que utilizem com base em *undersampling*. A aplicação dessa técnica poderá ocorrer exclusivamente ou em conjunto com outras técnicas de *oversampling*.

Referências

Bagga, S., Goyal, A., Gupta, N., and Goyal, A. (2020). Credit card fraud detection using pipeling and ensemble learning. *International Conference on Smart Sustainable Intelligent Computing and Applications under ICITETM2020*, 173:104–112.

Can, B., Yavuz, A. G., Karsligil, E. M., and Guvensan, M. A. (2020). A closer look into the characteristics of fraudulent card transactions. *IEEE Access*, 8:166095–166109.

Dornadula, V. N. and Geetha, S. (2019). Credit card fraud detection using machine learning algorithms. *2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019*, 165:631–641.

Makki, S., Assaghir, Z., Taher, Y., Haque, R., Hacid, M., and Zeineddine, H. (2019). An experimental study with imbalanced classification approaches for credit card fraud detection. *IEEE Access*, 7:93010–93022.

Misra, S., Thakur, S., Ghosh, M., and Saha, S. K. (2020). An autoencoder based model for detecting fraudulent credit card transaction. *International Conference on Computational Intelligence and Data Science*, 167:254–262.

Paul, A. C. B., Alves, M. A. Z., and de Oliveira, L. E. S. (2020). Avaliação de métodos de machine learning na detecção de fraude em dados transacionais de cartão de crédito.

Randhawa, K., Loo, C. K., Seera, M., Lim, C. P., and Nandi, A. K. (2018). Credit card fraud detection using adaboost and majority voting. *IEEE Access*, 6:14277–14284.

RB, A. and KR, S. K. (2021). Credit card fraud detection using artificial neural network. *1st International Conference on Advances in Information, Computing and Trends in Data Engineering (AICDE - 2020)*, 2(1):35–41.

Rtayli, N. and Enneya, N. (2020). Selection features and support vector machine for credit card risk identification. *13th International Conference Interdisciplinarity in Engineering, INTER-ENG 2019, 3–4 October 2019, Targu Mures, Romania*, 46:941–948.

Taha, A. A. and Malebary, S. J. (2020). An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE Access*, 8:25579–25587.

Tingfei, H., Guangquan, C., and Kuihua, H. (2020). Using variational auto encoding in credit card fraud detection. *IEEE Access*, 8:149841–149853.