Centro de Informática
UFPE

Clodes Fernando de Morais Silva

Aplicação do PageRank na detecção de fraude em transações com cartão de crédito

Universidade Federal de Pernambuco

Recife
2022

Clodes Fernando de Morais Silva

Aplicação do PageRank na detecção de fraude em transações com cartão de crédito

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

Área de Concentração: *Inteligência Computational*
Orientador: *Tsang Ing Ren*
Co-Orientador: *Hector Natan Batista Pinheiro*

Recife
2022

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Silva, Clodes Fernando de Morais.
Aplicação do PageRank na detecção de fraude em transações com cartão de crédito / Clodes Fernando de Morais Silva. - Recife, 2023.
34 p. : il., tab.

Orientador(a): Tsang Ing Ren
Cooorientador(a): Hector Natan Batista Pinheiro
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2023.

1. Cartão de crédito. 2. Detecção de fraude. 3. PageRank. 4. Inteligência artificial. I. Ren, Tsang Ing. (Orientação). II. Pinheiro, Hector Natan Batista. (Coorientação). III. Título.

000 CDD (22.ed.)

AGRADECIMENTOS

Agradeço, primeiramente, a minha família que me apoiou e me ajudou em toda a minha vida. Agradeço também aos meus amigos, sem os quais minha jornada seria extremamente difícil.

Agradeço a ajuda incondicional e fundamental do professor Hector e professor Tsang, sem os quais este trabalho não seria possível. Obrigado, mestres.

Agradeço à Incognia, empresa que nos forneceu a base de dados anonimizada utilizada neste trabalho, que foi fundamental para a realização dos experimentos. Muito obrigado.

ABSTRACT

Nowadays making payments using credit card is routine, billions of such transactions are made annually and billions of dollars are lost because of fraudulent transactions. This huge monetary loss shows that the application of fraud detection methods to financial transactions is critical. In this paper, we developed a fraud detection system for credit card transactions, which can be divided into two modules: a feature extractor and a binary classifier. A PageRank based algorithm was used as the feature extractor, and a decision tree based model called LightGBM was used as the binary classifier. The proposed system was evaluated using a real database of a food delivery application. The false positive rate (FPR) and true positive rate (TPR) were used for model performance evaluation. When used at an operating point with a FPR of 1%, the model achieved a TPR of 19.3%. Thus, the proposed approach can detect a reasonable amount of fraud even when operating at an operating point with a low FPR, validating its practical use.

Keywords: Credit Card, Fraud Detection, PageRank, Artificial Intelligence

RESUMO

Atualmente realizar transações com cartão de crédito é algo rotineiro, bilhões de transações dessa natureza são feitas anualmente e bilhões de dólares são perdidos por causa de transações fraudulentas. Por conta do impacto de fraudes em transações com cartão de crédito, a aplicação de métodos de detecção de fraude em transações financeiras é importantíssima. Foi desenvolvido neste trabalho um sistema de detecção de fraudes em transações com cartão de crédito, o qual pode ser dividido em dois módulos: um extrator de características e um classificador binário. Um algoritmo baseado no PageRank foi utilizado como extrator de características, e o modelo de aprendizagem de máquina baseado em árvores de decisão chamado LightGBM foi utilizado como classificador binário. O sistema proposto foi avaliado utilizando uma base de dados reais de um aplicativo de entrega de alimentos. As taxas de falsos positivos (TFP) e verdadeiros positivos (TVP) foram utilizadas para a avaliação de desempenho do modelo. Quando utilizado em um ponto de operação com TFP inferior a 1%, o modelo alcançou um TVP de 19.3%. Assim, a abordagem proposta consegue detectar uma quantidade razoável de fraudes mesmo operando em um ponto de operação com baixo TFP, validando a sua utilização prática.

Palavras-chave: Cartão de crédito, Detecção de fraude, PageRank, Inteligência artificial

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

Realizar transações utilizando cartão de crédito é algo rotineiro. Em 2016, mais de 34 bilhões de transações foram feitas, e nesse mesmo ano mais de 2 bilhões de dólares foram perdidos por conta de transações fraudulentas [1]. Assim, por conta desta grande relevância das fraudes em transações com cartão de crédito, a aplicação de métodos de detecção de fraude em transações financeiras é importantíssima.

Um sistema de detecção de fraudes em transações com cartão de crédito pode ser visto como um classificador binário, onde uma transação é classificada como fraudulenta ou confiável. Existem muitos desafios para a criação de um sistema desse tipo, como o grande volume de dados, a possibilidade de mudança no comportamento dos fraudadores e a necessidade de obter a classificação em tempo real [1]. Além disso, outro desafio é o desbalanceamento do dado, já que a proporção de transações fraudulentas por transações confiáveis é muito baixa, o que pode ser problemático para alguns tipos de classificadores, como por exemplo modelos de aprendizagem de máquina.

Inúmeras estratégias para detectar fraudes de cartão de crédito já foram descritas na literatura, várias utilizando modelos de aprendizagem de máquina [2]. Algoritmos baseados em grafos também podem ser utilizados como mostrado em [3], que usa um algoritmo de inferência coletiva para espalhar a influência das fraudes através do grafo, utilizando um conjunto de transações que foram confirmadas como fraudulentas. Um algoritmo bastante conhecido na mineração de dados em grafos é o PageRank [4], que foi criado para ranquear páginas web, mas também pode ser usado mais genericamente como uma forma de indicar a importância de um determinado vértice no grafo, além de também poder ser utilizado para espalhar através do grafo a influência de um subconjunto de vértices, o que é uma estratégia similar a [3].

1.1 OBJETIVOS

Este trabalho tem como objetivo a criação de um sistema de detecção de fraudes de cartão de crédito em pagamentos *online*, o qual será formado por dois componentes: um extrator de características e um classificador binário. Como extrator de características, será utilizado um algoritmo baseado no PageRank em grafos construídos a partir do histórico de transações,

espalhando através da rede a influência de um conjunto de transações fraudulentas, similarmente a [3]. Para a construção desses grafos, atributos atrelados às transações serão usados, os quais são dependentes do cenário no qual a transação está inserida. Em pagamentos *online* em sites de *e-commerce*, por exemplo, as transações estão atreladas a atributos como: conta utilizada para realizar a compra, dispositivo (por exemplo um celular ou computador), endereço de entrega, entre outros. Como classificador binário, será utilizado o modelo de aprendizagem de máquina LightGBM [5], pois é um modelo performático em relação ao tempo de treinamento, e funciona bem na prática para dados tabulares, especialmente quando o volume de dados disponíveis é grande. Os experimentos serão realizados num conjunto de dados referentes a transações reais de um aplicativo de entrega de alimentos, onde cada transação contém três atributos: conta, dispositivo e endereço de entrega.

1.2 ESTRUTURA DA MONOGRAFIA

O capítulo 2 apresentará uma revisão sobre sistemas de detecção de fraude em transações com cartão de crédito, além de abordar um algoritmo essencial para este trabalho que é o PageRank. No capítulo 3 o modelo proposto será apresentado, consistindo na definição de seu extrator de características e seu classificador. O capítulo 4 abordará o conjunto de dados utilizado e os resultados dos experimentos realizados. Finalmente, o capítulo 5 terá as considerações finais e possíveis trabalhos futuros relacionados.

2

REFERENCIAL TEÓRICO

Neste capítulo é feita uma breve revisão sobre alguns conceitos básicos, além de também apresentar alguns modelos de detecção de fraude em transações com cartão de crédito presentes na literatura, assim como métricas utilizadas para a avaliação de desempenho desses modelos.

2.1 DETECÇÃO DE FRAUDES EM TRANSAÇÕES COM CARTÃO DE CRÉDITO UTILIZANDO APRENDIZAGEM DE MÁQUINA

Detectar fraudes em transações com cartão de crédito pode ser visto como um problema de classificação binária, já que o objetivo é classificar uma transação como fraudulenta ou confiável. Aprendizagem de máquina é uma das abordagens que podem ser utilizadas para atacar um problema dessa natureza.

Um dos desafios da detecção de fraudes de cartão de crédito é o desbalanceamento do conjunto de dados, já que o número de transações confiáveis é muito maior do que o número de transações fraudulentas. Por exemplo, na base de dados de transações reais de um aplicativo de entrega de alimentos que foi utilizada neste trabalho, apenas 0.13% das transações são fraudulentas. Em termos gerais, lidar com um severo nível de desbalanceamento entre classes é um desafio para modelos de classificação em aprendizagem de máquina [6]. Uma das abordagens para combater o desbalanceamento dos dados é aplicar uma amostragem [7], podendo ser um *undersampling*, onde as classes são balanceadas por meio da diminuição de amostras da classe majoritária, ou um *oversampling*, onde é gerado um aumento de amostras da classe minoritária de forma a deixar o conjunto de dados mais balanceado.

Vários modelos clássicos já foram utilizados na detecção de fraudes de cartão de crédito, como podemos ver em [8], que faz um estudo comparativo sobre árvore de decisão, kNN (*K-Nearest Neighbors*), regressão logística, floresta aleatória e naive bayes. [9] mostra uma aplicação de Floresta Aleatória utilizando o SMOTE [10] para combater o desbalanceamento do conjunto de dados, que é uma técnica de *oversampling* da classe minoritária por meio da criação de amostras sintéticas.

Combinações de modelos também podem ser utilizadas, como mostrado em [11], o qual propõe uma abordagem composta por dois módulos: um detector de anomalias e um interpretador

de anomalias. O modelo não supervisionado *isolation forest* foi utilizado como detector de anomalias, onde uma pontuação é calculada para cada transação indicando o quão anômala a transação é. Uma floresta aleatória, modelo supervisionado, foi utilizada como segundo módulo chamado de interpretador de anomalias, e é responsável por identificar o tipo de anomalia das transações dadas como anômalas pelo módulo anterior, podendo assim encontrar as fraudes.

2.1.1 Métodos baseados em grafos

Métodos baseados em grafos também podem ser utilizados para detectar fraudes de cartão de crédito, como por exemplo o APATE [3], onde um conjunto de transações que foram confirmadas como fraude são utilizadas para espalhar a influência dessas fraudes através do grafo. Algumas melhorias ao APATE foram propostas posteriore em [12].

Em relação ao APATE [3], o grafo é definido sobre três entidades: transação, titular do cartão, e por fim, comerciante. Inicialmente, o conjunto de vértices do grafo é composto pelos titulares dos cartões e de comerciantes, e uma transação representa uma aresta no grafo entre o titular do cartão utilizado nessa transação e o comerciante. A fim de abordar o caráter dinâmico das fraudes, e inspirado no conceito de meia-vida dos átomos, o APATE insere o tempo no grafo modificando o peso das arestas (transações), assim o peso de uma aresta decai exponencialmente em relação ao tempo que passou desde que a transação aconteceu. Consequentemente, transações mais recentes possuem um peso maior. Após a construção do grafo, um algoritmo de propagação é utilizado para espalhar a influência das transações fraudulentas, obtendo assim, para todos os três tipos de entidades, *features* derivadas do grafo. O APATE também utiliza *features* intrínsecas que são independentes do grafo criado. Essas *features* dependem apenas do histórico de transações do titular do cartão, como por exemplo a soma dos valores transacionados nos últimos 7 dias.

Em [12], várias melhorias ao APATE são propostas. A primeira melhoria é o tratamento de vértices que são adjacentes a muitos outros. A existência desses vértices sem o devido tratamento pode ser prejudicial ao modelo. A segunda e terceira melhorias dizem respeito a uma abordagem para atacar o atraso nos rótulos das transações. O APATE utiliza transações recentes e seus rótulos para construir o modelo, mas em transações financeiras geralmente existe um atraso relacionado a confirmação de que uma fraude de fato ocorreu em uma transação (rótulo), já que muitas vezes isso depende de intervenção humana. Como apenas algumas transações tem seu rótulo disponível, a abordagem para atacar o atraso nesses rótulos é a utilização de um aprendizado semi-supervisionado para construir o modelo. A quarta melhoria é a remoção da pontuação dos comerciantes. Após uma investigação dos autores, foi descoberto que transações novas envolvendo comerciantes novos são prejudiciais ao modelo.

2.2 PAGERANK

O PageRank [4] é um algoritmo originalmente criado pelo Google com o objetivo de ranquear páginas web e foi o primeiro algoritmo usado pela empresa para ordernar os resultados de sua ferramenta de busca [13]. O PageRank calcula, para cada página, utilizando as citações dela, uma pontuação que pode ser interpretada como sua "importância". A pontuação de uma página p pode ser chamada de PageRank de p e definida como $PR(p)$. Se uma página q cita outra página p significa que existe um *link* de p em q, ou seja, é possível acessar p através de q por um clique.

Em [4], é definido um "surfista aleatório", que é alguém que começa a nagevar em uma página aleatória, e após chegar em qualquer página ele clica aleatoriamente em um dos *links* presentes nessa página, indo assim para outra página. Esse "surfista aleatório" pode ficar entediado e parar de seguir os *links*, após isso ele recomeçaria sua navegação numa página aleatória. A probabilidade do "surfista aleatório" seguir por um dos *links* da sua página atual é p_a e pode ser chamada de fator de amortecimento. O PageRank de uma página p pode ser visto como a probabilidade do "surfista aleatório" atingir a página p em sua navegação.

Seja N a quantidade de páginas, p uma página, $in(p)$ o conjunto de páginas que citam p e $out(p)$ o conjunto de páginas que são citadas por p, o PageRank de p pode ser definido como mostra a Equação 2.1 [4, 13]

$$PR(p) = \frac{1-p_a}{N} + p_a \sum_{q \in in(p)} \frac{PR(q)}{|out(q)|} \tag{2.1}$$

O cálculo do PageRank pode ser feito iterativamente, onde inicialmente o valor do PageRank das páginas é definido seguindo uma distribuição de probabilidade e, em cada iteração, seus valores são atualizados através da Equação 2.1. O valor do PageRank de uma determinada página depois da iteração i é denotado por $PR_i(p)$. Usualmente, assume-se uma distribuição uniforme, onde $PR_0(p)$ é $\frac{1}{N}$ para cada uma das páginas, mas outras inicializações também são factíveis.

Dado que P é o conjunto de todas as páginas, o Algoritmo 1 descreve o procedimento descrito para uma quantidade M de iterações.

A saída desse algoritmo iterativo do PageRank é uma distribuição de probabilidade que representa a chance do "surfista aleatório" chegar em uma determinada página após uma certa quantidade de passos [13]. Como o PageRank deve definir uma distribuição de probabilidade, então a Equação 2.2 deve ser obedecida após cada iteração.

$$\sum_{p \in P} PR_i(q) = 1 \tag{2.2}$$

Mas, observando a equação 2.1, é possível perceber que páginas que não contém citações, ou seja, páginas p em que $out(p) = \emptyset$, podem fazer com que a equação 2.2 não seja obedecida

Algoritmo 1: PageRank

Inicialize o vetor PR_0 com alguma distribuição de probabilidade
para $i \leftarrow 1$ ***até*** M
{
  para todo $p \in P$
  {
    $PR_i(p) = (1 - p_a) \ / \ N$
    para todo $q \in in(p)$
    {
      $PR_i(p) = PR_i(p) + p_a \ * \ PR_{i-1}(q) \ / \ |out(q)|$
    }
  }
}

após um número suficientemente grande de iterações. Essas páginas podem ser chamadas de *sinks*, e uma possível solução é considerar que para um *sink* o fator de amortecimento é 0 [13]. Sendo S o conjunto de todos os *sinks*, o PageRank de uma página p pode ser então definido como mostra a Equação 2.3 e o Algoritmo 1 pode ser modificado correspondentemente.

$$PR(p) = \frac{1 - p_a}{N} + p_a \sum_{q \in in(p)} \frac{PR(q)}{|out(q)|} + p_a \sum_{q \in S} \frac{PR(q)}{N} \tag{2.3}$$

2.2.1 PageRank personalizado

Em [14], é apresentado uma nova versão do método, que é nomeado como "PageRank personalizado". No PageRank personalizado, existe um vetor $\vec{s}$ onde s_i denota a probabilidade do "surfista aleatório" ir para a página de índice i caso ele fique entediado, ou seja, caso ele decida parar de seguir os *links*, recomeçando sua navegação em outra página. Seja $id(p)$ o índice da página p, o PageRank personalizado de uma página p pode ser calculado como mostra a equação 2.4. Em [15] é apresentado uma implementação do PageRank personalizado, onde existe um conjunto de páginas especiais S e o "surfista aleatório"começa aleatoriamente em alguma página de S.

$$PR(p) = (1 - p_a) s_{id(p)} + p_a \sum_{q \in in(p)} \frac{PR(q)}{|out(q)|} \tag{2.4}$$

2.3 CLASSIFICADORES BASEADOS EM ÁRVORE

Dado um conjunto de amostras e um conjunto de classes, a tarefa de classificação corresponde a identificar para cada amostra qual classe ela pertence [16]. Um classificador é o responsável por realizar a classificação, predizendo para cada amostra a sua classe. Não é garantido que o classificador vai conseguir predizer corretamente a classe de todas as amostras,

assim erros de classificação podem ocorrer e métricas de desempenho podem ser utilizadas para analisar a performance do classificador. Se apenas duas classes existirem, o problema pode ser chamado de classificação binária e o responsável pela classificação é chamado de classificador binário.

Um grupo de classificadores muito utilizados na literatura, e amplamente utilizados na indústria, são os classificadores baseados em árvore. O mais simples desse grupo, a árvore de decisão, é um modelo de aprendizagem de máquina que constrói uma árvore a partir das amostras aplicando em cada nó uma regra. Dessa maneira, em cada vértice da árvore, uma regra é gerada, separando os dados em dois grupos, aqueles que satisfazem a comparação ou não. Todas as amostras que obedecem a regra vão para um dos filho desse nó, e todas as amostras que não obedecem a regra vão para o outro filho desse nó. Incialmente todas as amostras estão na raiz da árvore, e para cada folha da árvore uma classe é atribuída (a mesma classe pode ser atribuída a mais de uma folha). Assim, ao percorrer a árvore, cada amostra pertencerá a uma folha, e a predição para essa amostra é a classe que foi atribuida à folha, que geralmente é baseada na quantidade de amostras de cada classe. O caminho da raiz até uma folha define a regra que foi utilizada na predição.

Outro modelo de aprendizagem de máquina baseado em árvore muito utilizado é a floresta aleatória, que é um comitê de árvores de decisão, onde cada árvore geralmente é construída utilizando subconjuntos diferentes de *features* e possivelmente uma amostragem do conjunto de dados de treinamento. Para realizar a predição de uma amostra, as predições de todas as árvores podem ser combinadas por abordagens como voto majoritário [17]. Outro tipo de comitê muito utilizado de árvores de decisão é o GBDT (*Gradient Boosting Decision Tree*) [18], o qual utiliza o gradiente da função de perda para treinar iterativamente árvores de decisão de forma a melhorar a predição do modelo em amostras que recebiam predições incorretas anteriormente. Diferentes versões e implementações do GBDT podem ser encontradas, como o XGBoost e o pGBRT. Um problema de muitas implementações do GBDT é a eficiência e escalabilidade [5], e o modelo LightGBM ataca esse problema, conseguindo uma acurácia similar enquanto diminui consideravelmente o tempo de treinamento [5]. O LightGBM pode ser uma escolha interessante para problemas com muitos dados devido a sua eficiência e seu poder preditivo.

2.4 MÉTRICAS DE PERFORMANCE

Diferentes métricas podem ser utilizadas para analisar a performance de modelos de detecção de fraudes em transações com cartão de créditos, como pode ser visto em [1, 8, 19]. Sendo a classe positiva o conjunto de transações fraudulentas e a classe negativa o conjunto de transações confiáveis, as principais métricas utilizadas na literatura para a análise desses modelos estão definidas a seguir.

- Taxa de verdadeiros positivos

$$\text{TVP} = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{2.5}$$

- Taxa de falsos positivos

$$\text{TFP} = \frac{\text{FP}}{\text{FP} + \text{VN}} \tag{2.6}$$

- Taxa de verdadeiros negativos

$$\text{TVN} = \frac{\text{VN}}{\text{FP} + \text{VN}} \tag{2.7}$$

- Taxa de falsos negativos

$$\text{TFN} = \frac{\text{FN}}{\text{VP} + \text{FN}} \tag{2.8}$$

- Precisão

$$\text{P} = \frac{\text{VP}}{\text{VP} + \text{FP}} \tag{2.9}$$

Onde VP, FP, VN e FN são as quantidades de verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos, respectivamente.

É importante notar que algumas métricas clássicas não são tão utilizadas em problemas de fraude em transações com cartão de crédito principalmente por conta do desbalanceamento do conjunto de dados, como por exemplo a acurácia (Equação 2.10), que é um exemplo de métrica que é enviesada pela classe majoritária.

$$\text{Acurácia} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{FN} + \text{VN} + \text{FP}} \tag{2.10}$$

3

MÉTODO PROPOSTO

Este trabalho tem como objetivo a criação de um método de detecção em tempo real de fraudes em transações com cartão de crédito. O método proposto pode ser dividido em dois módulos: o primeiro é um extrator de características responsável por mapear a transação para um vetor de características, e o segundo é um classificador binário responsável por classificar a transação como fraude ou confiável. Um diagrama simples mostrando o processo de predição pode ser visto na Figura 1.

Figura 1: Diagrama que mostra o processo de predição.

Para o treinamento do extrator de características é necessário uma base de transações e seus rótulos (fraude ou confiável), a partir dessa base grafos não direcionados são construídos e o PageRank personalizado é executado. O processo de construção do extrator de características depende de duas etapas fundamentais: criação dos grafos a partir dos atributos das transações, e execução do PageRank personalizado. Considere as definições presentes na Tabela 1.

Tabela 1: Definições utilizadas no método proposto.

Símbolo	**Definição**
N	Número de grafos construídos
G_i	i-ésimo grafo
V_i	Conjunto de vértices do i-ésimo grafo
E_i	Conjunto de arestas do i-ésimo grafo
T	Conjunto de transações
F	Conjunto de transações confirmadas como fraude
t_i	i-ésimo atributo de uma transação
A_i	Conjunto dos índices dos atributos usados na criação do grafo i
$adj(v)$	Conjunto de vértices adjacentes ao vértice v em um determinado grafo

3.1 CONSTRUÇÃO DOS GRAFOS

Um grafo não direcionado é um par (V,E), onde V é um conjunto não vazio de objetos chamados de vértices, e E é um conjunto de pares não ordenados de vértices, chamados de arestas.

Transações contém atributos que podem variar dependendo do contexto. Um conjunto de transações de um *website* de *e-commerce*, por exemplo, pode conter atributos como o valor da transação, uma cadeia de caracteres representando a conta que realizou o pagamento, e uma latitude e longitude representando o endereço de entrega. Já num aplicativo de entrega de alimentos, a base de transações também pode conter outra cadeia de caracteres representando o restaurante que vai preparar o pedido. Vários outros atributos também podem estar disponíveis para uso, dependendo do contexto e da base de dados utilizada.

Para cada grafo, existem dois tipos de vértices: o primeiro tipo representa as transações, e o segundo tipo representa os valores dos atributos utilizados nesse grafo. O conjunto V_i pode ser definido como mostra a Equação 3.1. Perceba que o conjunto A_i para diferentes grafos pode diferir, ou seja, grafos diferentes podem usar atributos diferentes, mas todos eles contêm um vértice para cada transação além dos vértices dos atributos.

$$V_i = \{t \in T\} \cup \{t_j \mid t \in T, j \in A_i\} \tag{3.1}$$

Nesse modelo, as arestas são ligações bidirecionais entre os atributos e os vértices que representam a transação, como pode ser observado na Equação 3.2.

$$E_i = \left\{ (t,t_j) \mid t \in T, j \in A_i \right\} \tag{3.2}$$

Os grafos construídos neste processo podem ser entendidos como uma maneira de possibilitar a utilização de transações fraudulentas para encontrar atributos que possivelmente estarão relacionados a fraudes futuras. Tomando como exemplo a base de transações presentes na Tabela 2, cada transação contém dois atributos: conta e dispositivo, e o grafo gerado por esses dois atributos pode ser visto na Figura 2. Considere que nesse cenário a transação **t1** é fraudulenta, e o autor de **t1** trocou de dispositivo antes de realizar a transação **t2**. Nesse contexto, o dispositivo **d2** não está diretamente relacionado a uma fraude, mas pertence a um fraudador, e assim pode ser considerado como uma potencial fonte de fraudes futuras. Assim, o grafo é uma forma de conectar transações fraudulentas a atributos que são potenciais fontes de fraudes futuras, e o PageRank pode ser entendido como a forma de encontrar *features* que indicam o potencial de cada vértice de estar relacionado a uma fraude no futuro. Mais detalhes sobre o PageRank estão presentes na próxima seção.

Tabela 2: Exemplo de uma base de transações contendo um atributo.

Transação	**Conta**	**Dispositivo**
t1	c1	d1
t2	c1	d2

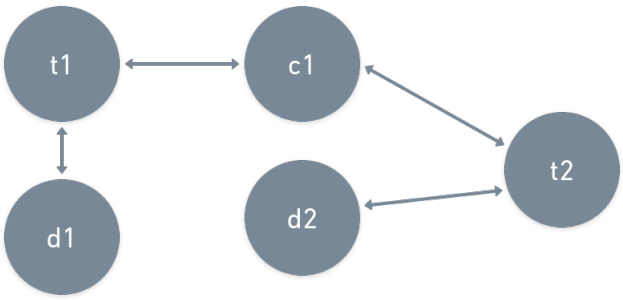

Figura 2: Examplo de um grafo construído a partir da base de transações presente na Tabela 2. A transação **t1** foi realizada por um fraudador, e esse mesmo fraudador trocou de dispositivo antes de realizar a transação **t2**.

É importante notar que para a construção dos grafos pode ser benéfico ao modelo realizar um pré-processamento nos atributos, de forma a criar novos atributos a partir dos existentes na base de dados original. Tomando como exemplo o endereço de entrega, para utilizar a informação da proximidade geográfica de dois endereços diferentes uma das abordagens é mapear endereços próximos para um mesmo valor, de forma que um único vértice represente esses dois endereços próximos. Uma das formas de mapear endereços dessa maneira é utilizando *geohashes* [20] e mapeando cada endereço de entrega para uma cadeia de caracteres, de maneira que endereços próximos sejam caracterizados pela mesma cadeia de caracteres. Também é possível utilizar algoritmos de clusterização e mapear cada endereço para um *cluster*.

Como será apresentado nos experimentos, em especial, o uso de atributos geográficos se mostrou bastante benéfico para o modelo proposto, uma vez que ele gera conexões entre transações mesmo nos casos onde o usuário troca de dispositivo e conta, o que é uma prática comum entre os fraudadores.

3.2 EXECUÇÃO DO PAGERANK

No contexto do PageRank, as páginas podem ser vistas como vértices de um grafo, e os *links* como arestas. Os vértices que são transações que foram confirmadas como fraude são denotados de vértices fraudulentos, e fazem parte do conjunto F. O modelo proposto faz uso do PageRank personalizado [14] tomando como base a implementação de [15], ou seja, o "surfista aleatório"começa aleatoriamente num vértice fraudulento, caso fique entediado ele salta aleatoriamente para algum vértice fraudulento, e caso não fique entendiado ele segue por uma aresta aleatória do vértice que ele está. O PageRank é executado independentemente em cada grafo construído. Conforme a Equação 2.4, o PageRank de um vértice v no i-ésimo grafo pode

ser definido pela Equação 3.4.

$$g(v) = \begin{cases} \frac{1}{|F|}, & v \in F \\ 0, & v \notin F \end{cases} \tag{3.3}$$

$$PR_i(v) = g(v)(1 - p_a) + p_a \sum_{u \in adj(v)} \frac{PR(u)}{|adj(u)|} \tag{3.4}$$

Para o cálculo do PageRank é utilizado o método iterativo como pode ser observado no Algoritmo 2, para uma quantidade M de iterações. $PR_{i,j}(v)$ é o PageRank de v no grafo G_i após a iteração j. Um diagrama indicando o processo completo de construção do extrator de características está indicado na Figura 3.

Algoritmo 2: PageRank personalizado

```
para i ← 1 até N
{
  para todo v ∈ V_i
  {
    PR_{i,0}(v) = 0
    se v ∈ F
    {
      PR_{i,0}(v) = 1/|F|
    }
  }
  para j ← 1 até M
  {
    para todo v ∈ V_i
    {
      PR_{i,j}(v) = 0
      se v ∈ F
      {
        PR_{i,j}(v) = PR_{i,j}(v) + (1 − p_a) / |F|
      }
      para todo u ∈ adj(v)
      {
        PR_{i,j}(v) = PR_{i,j}(v) + p_a * PR_{i,j−1}(u) / |adj(u)|
      }
    }
  }
}
```

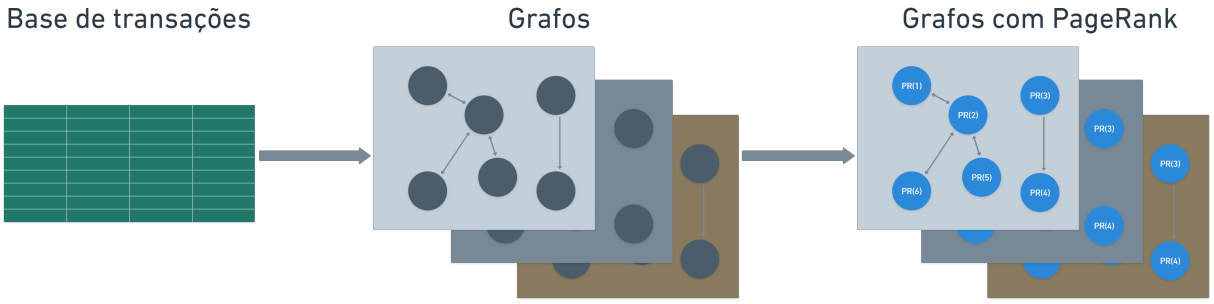

Figura 3: Processo de construção do extrator de características. Num primeiro momento a base de transações é recebida, a partir dela vários grafos são modelados, e após isso, em cada um desses grafos é aplicado o PageRank, encontrando assim para cada grafo o valor do PageRank para todos os vértices.

3.3 EXTRAÇÃO DO VETOR DE CARACTERÍSTICAS

Para a realização da detecção de fraudes em tempo real, o extrator de características deve gerar um vetor de características para uma nova transação através de seus atributos, o qual será posteriormente utilizado na etapa de classificação. O extrator proposto utiliza os atributos da transação e gera características utilizando o PageRank dos atributos de cada um dos grafos construídos anteriormente.

Seja $a_{i,j}$ o j-ésimo atributo utilizado no grafo i, o vetor de características $\vec{v}_i$ proveniente do grafo i pode ser definido como mostrado na Equação 3.5. O vetor de características resultante é a concatenação de todos os $\vec{v}_i$.

$$\vec{v}_i = (PR_i(a_{i,1}),\ PR_i(a_{i,2}),\ ...,\ PR_i(a_{i,|A_i|})) \tag{3.5}$$

3.4 CLASSIFICADOR

Após a obtenção do vetor de características por meio do extrator de características, o modelo proposto faz a classificação da transação como fraude ou confiável por meio da utilização de um classificador binário. A priori, classificadores supervisionados, não supervisionados ou semi-supervisionados podem ser utilizados, como por exemplo alguns classificadores citados na Seção 2.1 e Seção 2.3.

4

EXPERIMENTOS E RESULTADOS

Nesta seção será apresentada uma visão geral da base de dados utilizada, bem como os experimentos realizados e os resultados alcançados.

4.1 BASE DE DADOS

A base de dados utilizada para a validação do modelo proposto por este trabalho consiste de transações reais realizadas através de um dispositivo móvel em um aplicativo de entrega de alimentos durante um período de 67 dias. A base de dados foi disponibilizada pela empresa Incognia [21] e, por motivos de segurança e privacidade, todas as transações passaram por um processo de anonimização e o nome do aplicativo não será divulgado nesse documento. Esse conjunto de dados contém 138,001,049 transações, onde a frequência de cada classe está na Tabela 3. Apenas 0.13% das transações são fraudulentas, mostrando que a base é bastante desbalanceada.

Tabela 3: Frequência de cada classe na base de dados de transações reais de um aplicativo de entrega de alimentos.

Classe	**Transações**	**Transações (%)**
fraude	176,061	0.13
confiável	137,824,988	99.87

Cada transação contém três atributos: uma cadeia de caracteres indicando a conta, outra cadeia de caracteres indicando o dispositivo, e um par de coordenadas geográficas (latitude e longitude) do endereço de entrega do pedido correspondente à transação. Algumas métricas sobre o atributos de conta e dispositivo estão apresentadas na Tabela 4.

4.2 METODOLOGIA

Para a realização dos experimentos, a base de transações foi separada em três conjuntos: o primeiro foi utilizado para o treinamento do extrator de características, o segundo foi utilizado para o treinamento do classificador, e o terceiro foi utilizado para o teste do classificador.

Tabela 4: Análise inicial da base de dados.

Métrica	**Conta**	**Dispositivo**
Valores distintos	20,950,928	22,703,012
Média da quantidade de transações	7.11	6.56
Desvio padrão da quantidade de transações	13.26	12.7
Máxima quantidade de transações	12,669	10,459

Os primeiros 45 dias de transações foram utilizados para o treinamento do extrator de características. Assim, essas transações foram usadas para a geração do grafo e execução do PageRank. Os próximos 15 dias foram utilizados para gerar a base de treinamento do classificador. Os vetores de características para essas transações foram gerados a partir do extrator de características gerado anteriormente, obtendo assim o conjunto que foi utilizado para treinamento do modelo. Finalmente, as transações dos últimos 7 dias foram utilizadas para gerar o conjunto de teste do classificador. Novamente, para essas transações os vetores de características foram gerados utilizando o mesmo extrator desenvolvido na primeira etapa. A Figura 4 mostra a divisão temporal da base de dados. A distribuição da quantidade de transações por rótulo, para cada conjunto de transação mencionado, está na Tabela 5.

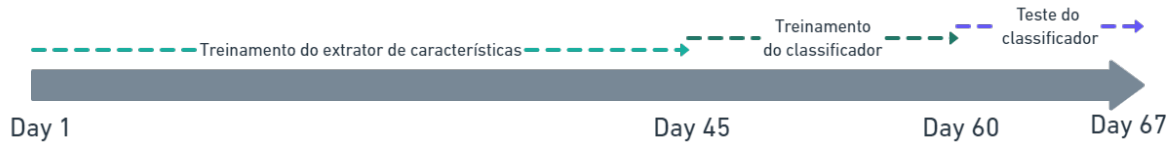

Figura 4: Intervalo temporal da base de transações utilizada para treinar o extrator de características, treinar o classificador e testar o classificador.

Tabela 5: Frequência de cada classe para cada conjunto de transações.

Finalidade do conjunto de transações	**Classe**	**Transações**	**Transações** (%)
Construção do extrator de características	fraude	113,587	0.11
Construção do extrator de características	confiável	103,526,310	99.89
Treinamento do classificador binário	fraude	43,096	0.19
Treinamento do classificador binário	confiável	22,770,565	99.81
Teste do classificador binário	fraude	19,378	0.17
Teste do classificador binário	confiável	11,528,113	99.83

Para os experimentos seguintes, o PageRank personalizado (Algoritmo 2) foi utilizado no extrator de características. O número de iterações considerado foi 10, esse valor é limitado por conta do grande volume de transações e do custo computacional para executar cada iteração do PageRank. O valor do fator de amortecimento considerado foi 0.85, que é o valor comumente utilizado [4]. Um LightGBM com no máximo 200 árvores foi utilizado como classificador binário. No treinamento do LightGBM, a busca de hiper-parâmetros foi realizada via Optuna

[22], utilizando validação cruzada *3-fold*, e a função de *Focal Loss* [23] foi utilizada para abordar o desbalanceamento das classes.

4.3 BASELINE

Uma abordagem bastante popular, e amplamente utilizada na indústria para detecção de fraudes, consiste no bloqueio das transações de dispositivos que foram identificados anteriormente como fraudulentos. Nesse cenário, toda transação realizada por um dispositivo com histórico de fraude, isto é, que realizou no passado alguma transação com rótulo de fraude, é bloqueada. Esse modelo de detecção de fraudes foi utilizado como *baseline* neste trabalho. Seu desempenho será apresentado e comparado com aqueles alcançados pela abordagem proposta.

4.4 EXPERIMENTO INICIAL

O experimento inicial foi realizado utilizando dois atributos das transações: conta e dispositivo. Nesse experimento apenas um grafo é gerado. O vetor de características retornado pelo extrator de características tem dois campos, como mostrado na Tabela 6.

Tabela 6: Campos do vetor de características retornado pelo extrator de características do experimento inicial.

Feature	**Significado**
PR_grafo_1_conta	PageRank do vértice da conta no grafo 1
PR_grafo_1_dispositivo	PageRank do vértice do dispositivo no grafo 1

A Tabela 7 mostra um exemplo de transações realizadas por um mesmo fraudador, onde t_1 aconteceu antes de t_2. Esse cenário é possível já que o fraudador pode ter trocado de dispositivo e criado uma nova conta logo antes de realizar a transação t_2. O grafo gerado pode ser visto na Figura 5.

Tabela 7: Exemplo de uma base de transações contendo conta e dispositivo.

Transação	**Conta**	**Dispositivo**
t_1	c_1	d_1
t_2	c_2	d_2

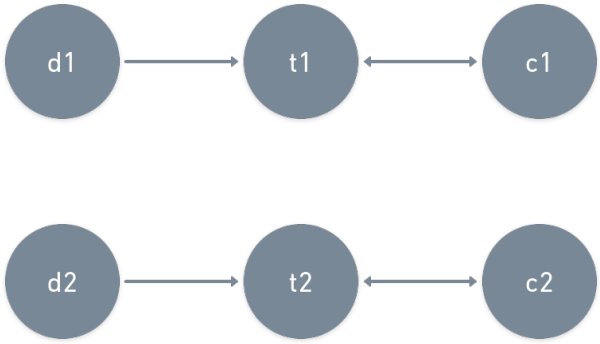

Figura 5: Grafo construído a partir da transação t_1 presente na Tabela 7.

4.5 EXPERIMENTO UTILIZANDO INFORMAÇÃO GEOGRÁFICA

Uma limitação do experimento anterior consiste na possibilidade do fraudador trocar de dispositivo e criar uma nova conta para cometer outra fraude. Nesse cenário, não seria possível associar suas fraudes passadas na detecção de fraudes futuras em tempo real, dado que o modelo proposto utiliza um extrator de características previamente treinado para encontrar o vetor de características de novas transações. O endereço de entrega pode ser utilizado para atenuar essa limitação, já que além de modificar a conta e dispositivo, o fraudador também teria que trocar o endereço de entrega para que não fosse possível possível relacionar suas fraudes passadas com transações futuras.

Assim, nesse experimento, além dos atributos de conta e dispositivo, o endereço de entrega das transações também foi usado. Na base de dados utilizada, o endereço de entrega é representado pelas coordenadas geográficas, indicando assim sua geolocalização. Devido ao fato das coordenadas não serem estáveis e constantes sempre, sua utilização direta para a geração de vértices nos grafos não é possível. Para contornar esse cenário, uma discretização geográfica foi utilizada, onde coordenadas geográficas foram transformadas em *geohashes* [20]. *Geohash* é um método popularmente utilizado para discretização de coordenadas geográficas, seguindo uma determinada precisão. Dessa maneira, coordenadas geográficas próximas umas das outras são mapeadas para o mesmo valor de *geohash*.

No total, quatro grafos serão utilizados no extrator de características desse experimento, e seus atributos estão descritos na Tabela 8. Sendo assim, um vetor de característica gerado pelo extrator de características tem um total de 8 campos, como mostra a Tabela 9.

Tabela 8: Atributos utilizados em cada grafo do segundo experimento.

Grafo	**Atributos**
1	Conta, dispositivo
2	Conta, dispositivo, geohash de precisão 6
3	Conta, dispositivo, geohash de precisão 7
4	Conta, dispositivo, geohash de precisão 8

Tabela 9: Campos do vetor de características retornado pelo extrator de características do segundo experimento.

Feature	**Significado**
PR_grafo_1_conta	PageRank do vértice da conta no grafo 1
PR_grafo_1_dispositivo	PageRank do vértice do dispositivo no grafo 1
PR_grafo_2_conta	PageRank do vértice da conta no grafo 2
PR_grafo_2_dispositivo	PageRank do vértice do dispositivo no grafo 2
PR_grafo_2_geo_6	PageRank do vértice da geohash de precisão 6 no grafo 2
PR_grafo_3_conta	PageRank do vértice da conta no grafo 3
PR_grafo_3_dispositivo	PageRank do vértice do dispositivo no grafo 3
PR_grafo_3_geo_7	PageRank do vértice da geohash de precisão 7 no grafo 3
PR_grafo_4_conta	PageRank do vértice da conta no grafo 4
PR_grafo_4_dispositivo	PageRank do vértice do dispositivo no grafo 4
PR_grafo_4_geo_8	PageRank do vértice da geohash de precisão 8 no grafo 4

4.6 RESULTADOS

Como mostra a Figura 4, as transações dos últimos 7 dias foram utilizadas para o teste do modelo proposto, onde 0.17% das 1,547,491 transações são rotuladas como fraudulentas, e o restante como confiáveis.

As taxas de falsos positivos, verdadeiros positivos e precisão da abordagem *baseline* (Seção 4.3) estão na Tabela 10. Devido ao fato dessa abordagem consistir de uma regra binária para rejeição de transações, sem cálculo de valores de *scores*, não existe a possibilidade do uso de limiares, resultando em apenas um ponto de operação. Como pode ser visto, a taxa de falsos positivos é bastante pequena, 0.16%, como desejado em sua aplicação prática. Por outro lado, a taxa de verdadeiros positivos foi de apenas 7.62%, o que mostra que usar apenas a informação de dispositivo não é suficiente para bloquear uma grande quantidade de transações fraudulentas. A precisão alcançada foi de 7.53%, mostrando que mesmo com uma taxa de falsos positivos bastante baixa e muito menor que a taxa de verdadeiros positivos, a maioria das transações preditas como fraude pertencem, na verdade, à classe de transações confiáveis, ilustrando assim um dos desafios de trabalhar com um conjunto de dados com classes desbalanceadas.

Tabela 10: Taxa de falsos positivos, verdadeiros positivos e precisão, da abordagem *baseline*.

Taxa de falsos positivos (%)	**Taxa de verdadeiros positivos (%)**	**Precisão (%)**
0.16	7.62	7.53

Na Tabela 11 são apresentadas as taxas de verdadeiros positivos e precisão de ambos os modelos propostos para diferentes limites de TFP. Isto é, cada linha representa o último ponto de operação com um TFP menor que o limite determinado.

No ponto de operação da abordagem *baseline* (Seção 4.3), o modelo inicial alcança uma taxa de verdadeiros positivos similar mas ligeiramente menor do que o alcançado pelo

Tabela 11: Taxas de verdadeiros positivos e precisão para diferentes limites de TFP dos dois modelos propostos. Cada linha representa o último ponto de operação com um TFP menor que o limite determinado.

	Modelo Inicial		Modelo com endereço de entrega	
Limite de TFP (%)	**TVP (%)**	**Precisão (%)**	**TVP (%)**	**Precisão (%)**
0.01	0.7	18.7	5.2	46.8
0.05	4.8	14	8	21.3
0.1	6.8	10.3	10.7	15.2
0.16	7.5	9.1	12.1	11.4
0.25	9.1	7	13.7	8.4
0.5	9.1	7	15.9	5
1	9.1	7	19.3	3.1
5	9.1	7	31.4	1
10	9.1	7	41.1	0.7

baseline, mas, em compensação, apresentou uma precisão maior nesse mesmo ponto de operação. Indicando assim que o modelo inicial classifica erroneamente menos transações para conseguir detectar corretamente uma quantidade similar de fraudes em comparação com as detectadas pela abordagem *baseline*, apesar dessa quantidade de fraudes detectadas (TVP) ser ligeiramente menor. O modelo proposto que utiliza o endereço de entrega mostrou-se superior à abordagem *baseline* e ao modelo inicial em todos os pontos de operação apresentados, conseguindo alcançar melhores valores de TVP e precisão.

Como indicado pela Tabela 11, o TVP do modelo inicial não se altera a partir de certo ponto. Após uma investigação, foi possível notar que isso pode acontecer por conta da quantidade baixa de caminhos no grafo inicial. Se não existe um caminho de um vértice v até um vértice fraudulento, é impossível que o PageRank de v seja diferente de zero nesse modelo proposto, independente da quantidade de iterações usadas na execução do PageRank. Na Tabela 12 podemos observar, para cada *feature*, a proporção de transações com PageRank não nulo. Nota-se que esse valor é baixo antes da introdução do endereço de entrega no grafo, o que explicaria um baixo TVP do modelo inicial já que muitos vértices tem o PageRank zerado.

A curva PRC do modelo de ambos os experimentos está na Figura 6. Nela, pode-se observar que o modelo inicial apresenta poucos pontos de operação e, no geral, associados a taxas de verdadeiros positivos baixas. A adição do endereço de entrega nos grafos, e consequentemente, no vetor de características resultante, mostrou-se ser bastante benéfica ao modelo proposto, já que o modelo que utiliza o endereço de entrega obteve uma precisão melhor para cada taxa de verdadeiros positivos, em pontos de operação existente a ambos os modelos.

Tabela 12: Proporção de transações com PageRank não nulo para cada *feature*

Feature	**Proporção de transações com PageRank não nulo (%)**
grafo_1_conta_pagerank	0.21
grafo_1_dispositivo_pagerank	0.2
grafo_2_conta_pagerank	84.1
grafo_2_dispositivo_pagerank	81.6
grafo_2_geohash_6_pagerank	99.9
grafo_3_conta_pagerank	40.5
grafo_3_dispositivo_pagerank	39.2
grafo_3_geohash_7_pagerank	91.5
grafo_4_conta_pagerank	20.4
grafo_4_dispositivo_pagerank	19.7
grafo_4_geohash_8_pagerank	56.1

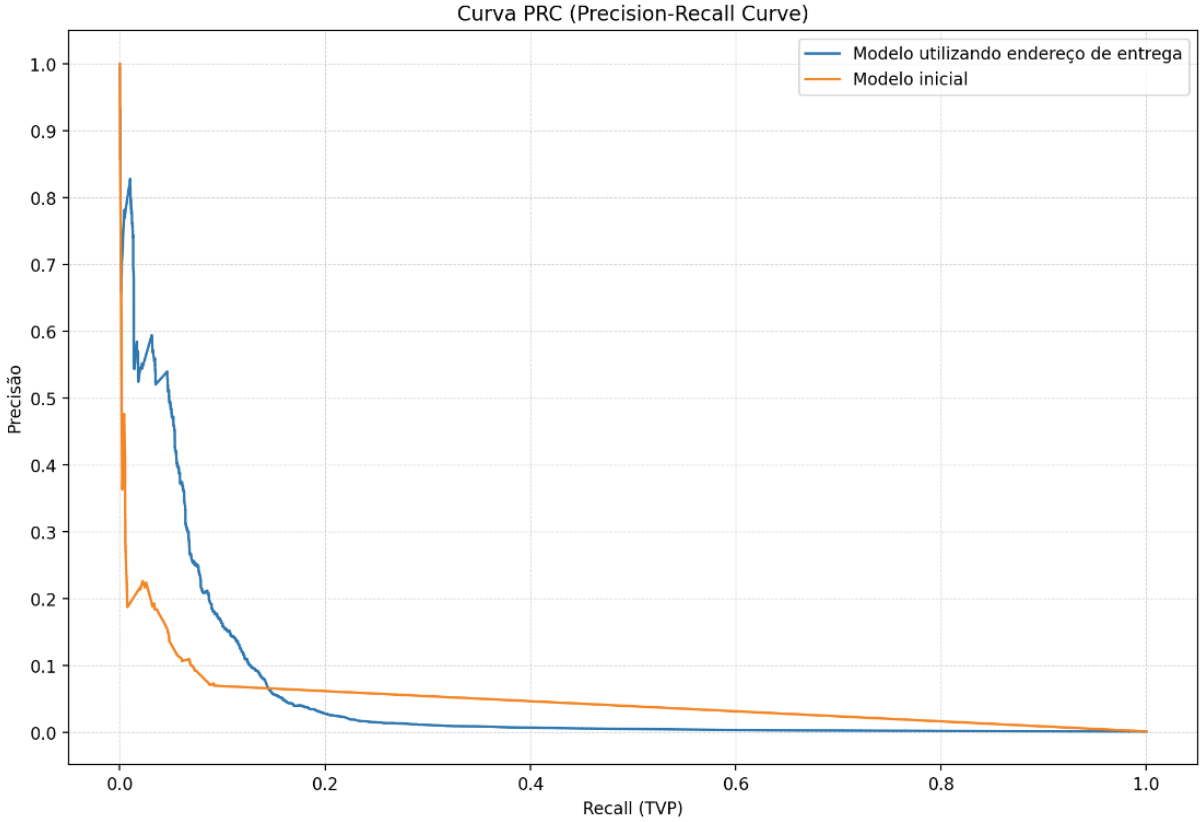

Figura 6: Curvas PRC para os dois modelos.

Na Figura 7 pode ser observado a quantidade de *splits* feitos nas árvores que compõem o classificador do segundo experimento para cada *feature*. Similarmente, a Figura 8 mostra o ganho total de informação dos *splits* de cada *feature*.

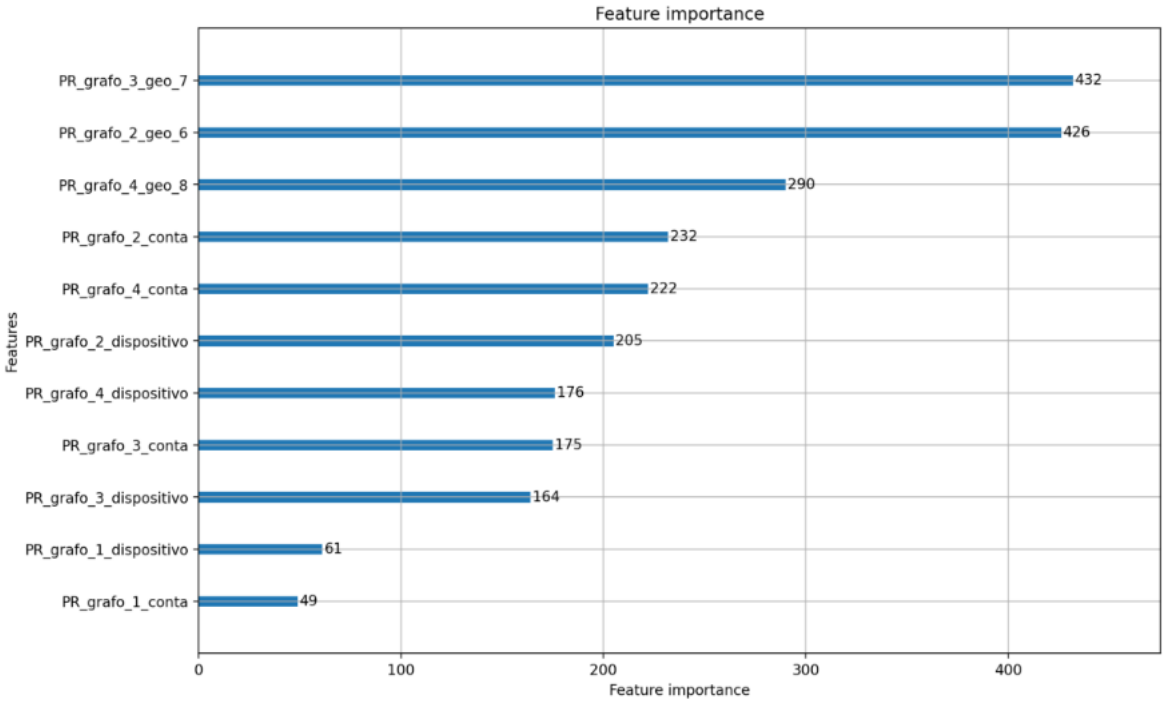

Figura 7: Quantidade total de *splits* para cada *feature*.

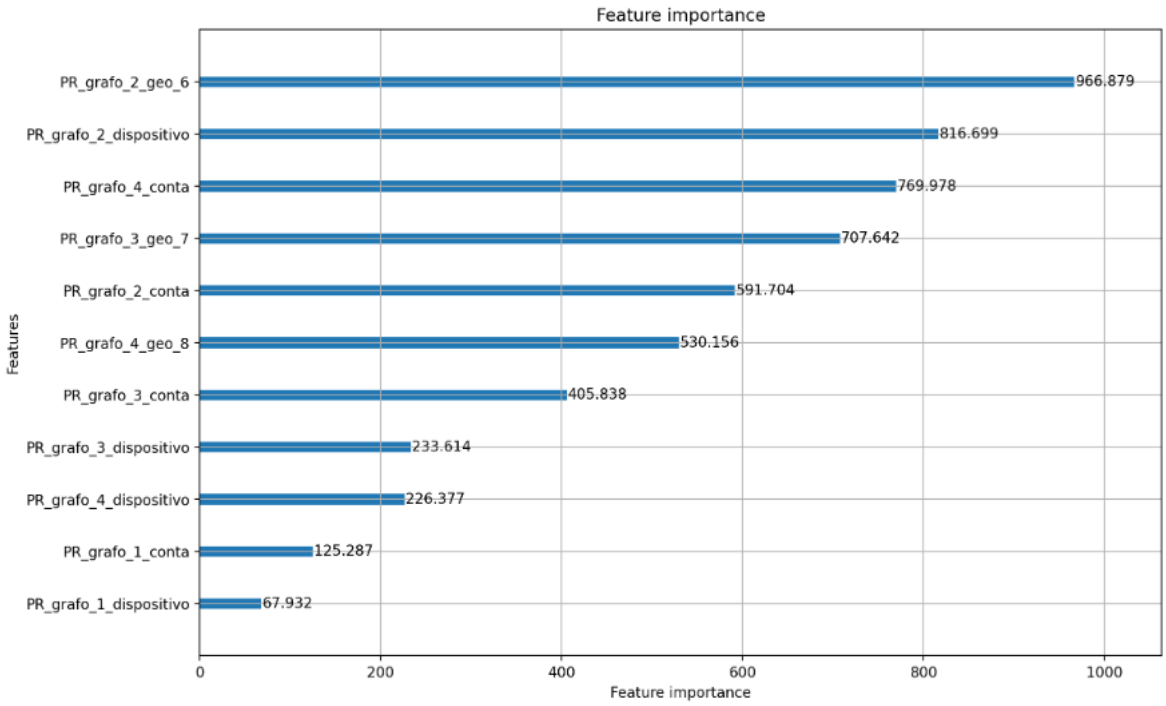

Figura 8: Ganho total de informação dos *splits* de cada *feature*.

Como apresentado pela Tabela 12, a adição do endereço de entrega foi benéfica ao extrator de características, pois criou caminhos novos de vértices fraudulentos ao resto do grafo. Isso mostra que a limitação do primeiro experimento, citada anteriormente na Seção 4.5, foi atenuada pelo endereço de entrega, e isso pode acontecer pelo fato do endereço de entrega ser mais difícil de ser modificado a cada transação em comparação com a conta e dispositivo. Juntamente com

o fato do modelo construído no segundo experimento ter apresentado um resultado melhor, e das *features* geradas pelos grafos que utilizam *geohashes* terem uma importância alta, pode-se observar a relevância das informações de geolocalização para a detecção de transações fraudulentas.

5

CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi apresentado um método de detecção de fraudes em transações com cartão de crédito. Na abordagem proposta o PageRank personalizado foi utilizado como extrator de características com base em grafos criados a partir dos atributos das transações, e um classificador responsável por classificar a transação como fraude ou confiável.

O método proposto foi desenvolvido utilizando uma base de transações de um aplicativo de entrega de alimentos com 138,001,049 amostras. Dois modelos foram propostos considerando diferentes atributos da transação: o primeiro utiliza conta e dispositivo, enquanto que o segundo adiciona a informação geográfica do endereço de entrega.

O melhor modelo conseguiu alcançar um TVP de 19.3% tendo como base um ponto de operação com TFP de 1%, demonstrando assim que é possível utilizar esses grafos construídos a partir dos atributos das transações para detectar fraudes. Também pôde-se observar que a adição do endereço de entrega foi muito benéfica ao modelo, mostrando assim a força que esse sinal de geolocalização pode ter.

5.1 TRABALHOS FUTUROS

Existem muitos pontos de melhoria para este modelo, bem como a exploração de algumas outras abordagens que podem ser interessantes. Seguem algumas delas:

- Modificações no PageRank podem ser utilizadas, como alguns tratamentos aplicados em [12], que propõe melhorias ao APATE. Além disso, o PageRank pode ser calculado utilizando mais iterações e o fator de amortecimento é um hiperparâmetro importante que pode ser otimizado.
- Além das *features* derivadas do grafo e PageRank, *features* intrínsecas podem ser adicionadas diretamente no vetor de características, similarmente ao que é feito no APATE [3].
- A exploração de redes neurais de grafos [24] para esse problema seria interessante, dado que esses são modelos poderosos em que conseguiríamos tomar proveito da estrutura do grafo utilizando também *features* intrísecas por vértice ou por aresta.

REFERÊNCIAS

[1] Y. Yazici, "Approaches to fraud detection on credit card transactions using artificial intelligence methods," in *Computer Science & Information Technology*, AIRCC Publishing Corporation, jul 2020.

[2] R. Bin Sulaiman, V. Schetinin, and P. Sant, "Review of machine learning approach on credit card fraud detection," *Human-Centric Intelligent Systems*, vol. 2, 05 2022.

[3] V. Van Vlasselaer, C. Bravo, O. Caelen, T. Eliassi-Rad, L. Akoglu, M. Snoeck, and B. Baesens, "Apate: A novel approach for automated credit card transaction fraud detection using network-based extensions," *Decision Support Systems*, vol. 75, pp. 38–48, 2015.

[4] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks*, vol. 30, pp. 107–117, 1998.

[5] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 3149–3157, Curran Associates Inc., 2017.

[6] H. Kaur, H. S. Pannu, and A. K. Malhi, "A systematic review on imbalanced data challenges in machine learning: Applications and solutions," *ACM Comput. Surv.*, vol. 52, aug 2019.

[7] S. Tyagi and S. Mittal, "Sampling approaches for imbalanced data classification problem in machine learning," in *Proceedings of ICRIC 2019* (P. K. Singh, A. K. Kar, Y. Singh, M. H. Kolekar, and S. Tanwar, eds.), (Cham), pp. 209–221, Springer International Publishing, 2020.

[8] S. Khatri, A. Arora, and A. P. Agrawal, "Supervised machine learning algorithms for credit card fraud detection: A comparison," in *2020 10th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pp. 680–683, 2020.

[9] A. M. Aburbeian and H. I. Ashqar, "Credit card fraud detection using enhanced random forest classifier for imbalanced data," *ArXiv*, vol. abs/2303.06514, 2023.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[11] O. Vynokurova, D. Peleshko, O. Bondarenko, V. Ilyasov, V. Serzhantov, and M. Peleshko, "Hybrid machine learning system for solving fraud detection tasks," in *2020 IEEE Third International Conference on Data Stream Mining Processing (DSMP)*, pp. 1–5, 2020.

[12] B. Lebichot, F. Braun, O. Caelen, and M. Saerens, "A graph-based, semi-supervised, credit card fraud detection system," in *Complex Networks & Their Applications V: Proceedings of the 5th International Workshop on Complex Networks and their Applications (COMPLEX NETWORKS 2016)*, pp. 721–733, Springer, 2017.

[13] "Pagerank," 2004. Disponível em: https://en.wikipedia.org/wiki/PageRank. Acesso em: 18 de mar. de 2023.

[14] S. Park, W. Lee, B. Choe, and S.-g. Lee, "A survey on personalized pagerank computation algorithms," *IEEE Access*, vol. PP, pp. 1–1, 11 2019.

[15] A. Moreau, "How to perform fraud detection with personalized page rank," 2019. Disponível em: https://en.wikipedia.org/wiki/Geohash. Acesso em: 21 de mar. de 2023.

[16] "Pagerank," 2007. Disponível em: https://en.wikipedia.org/wiki/Statistical_classification. Acesso em: 18 de mar. de 2023.

[17] V. Kulkarni and P. Sinha, "Random forest classifiers: A survey and future research directions," *International Journal of Advanced Computing*, vol. 36, pp. 1144–1153, 01 2013.

[18] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[19] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, and A. Anderla, "Credit card fraud detection-machine learning methods," in *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, pp. 1–5, IEEE, 2019.

[20] "Geohash," 2009. Disponível em: https://en.wikipedia.org/wiki/Geohash. Acesso em: 21 de mar. de 2023.

[21] Disponível em: https://www.incognia.com/pt/. Acesso em: 24 de abril de 2023.

[22] Disponível em: https://en.wikipedia.org/wiki/Geohash. Acesso em: 24 de abril de 2023.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[24] Y. Liu, X. Ao, Z. Qin, J. Chi, J. Feng, H. Yang, and Q. He, "Pick and choose: a gnn-based imbalanced learning approach for fraud detection," in *Proceedings of the Web Conference 2021*, pp. 3168–3177, 2021.