UNIVERSIDADE FEDERAL DO PAMPA

Rafael Duarte Beltran

Detecção de Fraudes Bancárias Utilizando Métodos de Clustering

Alegrete
2019

Rafael Duarte Beltran

Detecção de Fraudes Bancárias Utilizando Métodos de Clustering

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Alessandro Bof de Oliveira

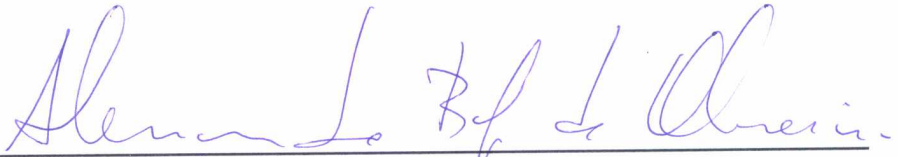
Alegrete
2019

Rafael Duarte Beltran

Detecção de Fraudes Bancárias Utilizando Métodos de Clustering

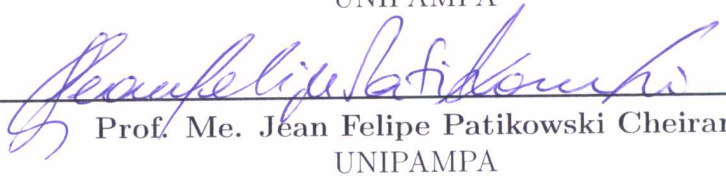
Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Trabalho de Conclusão de Curso defendido e aprovado em 28 de junho de 2019
Banca examinadora:



Prof. Dr. Alessandro Bof de Oliveira
Orientador
UNIPAMPA

Prof. Dr. Marcelo Caggiani Luizelli
UNIPAMPA



Prof. Me. Jean Felipe Patikowski Cheiran
UNIPAMPA

RESUMO

Instituições financeiras tem utilizado extensivamente sistemas informatizados para a prestação de seus serviços. Dessa forma, é crescente o número de soluções financeiras por meio digital. A conveniência da utilização dessas ferramentas traz consigo a importante questão da segurança dos dados nessas operações financeiras. Dentre essas questões de segurança, este trabalho se concentra na detecção de fraudes de cartão de crédito. Ao utilizar um meio de pagamento eletrônico, como o cartão de crédito, para efetuar compras ou contratação de serviços. No momento que o cartão de crédito é utilizado, são adicionados dados dessa compra para uma tabela de transações, alguns desses dados são os horários das compras e seus valores. Esses dados podem ser utilizados para construir um padrão de comportamento típico. Quando a segurança do sistema é comprometida como, por exemplo, a utilização do cartão de crédito por outra pessoa que não o usuário, temos a ocorrência de uma fraude. O objetivo deste projeto é detectar comportamentos não usuais, ou seja, transações atípicas em comparação às outras.
Os dados utilizados são provenientes de uma base de dados real com histórico de transações de cartões de crédito e anônimas. A base de dados é rotulada com 284.807 transações normais e 492 fraudes. Nesse trabalho, foi realizada uma análise da base de dados em questão, após essa análise os recursos desnecessários foram removidos. Tendo o conjunto de dados pronto, foi realizada uma seleção, dos métodos de *clustering* que seriam aplicados e, posteriormente testadas abordagens híbridas desses algoritmos. Como resultados, nos melhores casos, foram obtidas taxas de detecção de fraude maiores que 88%, utilizando os métodos de *clustering* DBSCAN e *K-Means*. O DBSCAN obteve melhores resultados superiores, utilizando métrica Manhattan.

Palavras-chave: Fraudes eletrônicas, agrupamento de dados, métodos híbridos de clusterização.

ABSTRACT

Financial institutions have extensively used computerized systems for the provision of their services. Therefore, the number of financial solutions through digital means is increasing. The convenience of using these tools brings with it the important issue of data security in these financial transactions. Among these security issues, this paper focuses on the detection of credit card fraud. By using an electronic payment method, such as a credit card, to make purchases or contract services. When the credit card is used, data from that purchase is added to a transaction table, some of which are the purchase moment and their values. These data can be used to construct a typical pattern of behavior. When system security is compromised, such as when the credit card is used by someone other than the user, we have a fraud. The purpose of this project is to detect unusual behaviors, that are, atypical transactions in comparison to others.
The data used comes from a real database with history of credit card transactions and anonymous. The database is labeled with 284,807 standard transactions and 492 frauds. In this study, an analysis of the database in question was performed, after this analysis the unnecessary resources were removed. Having the data set ready, a selection was made of the clustering methods that would be applied, and subsequently tested hybrid approaches of these algorithms. As a result, in the best cases, fraud detection rates greater than 88% were obtained, using the clustering methods DBSCAN and K-Means.The DBSCAN achieved better results using Manhattan metric.

Key-words: Electronic fraud, clustering of data, hybrid clustering.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CPU Central Processing Unit

DBSCAN Density-based spatial clustering of applications with noise

GB Gigabyte

IA Inteligência Artificial

KNN k-nearest neighbors

PCA Principal Component Analysis

PIN Personal Identification Number

RAM Random Access Memory

TCC Trabalho de Conclusão de Curso

SUMÁRIO

1 INTRODUÇÃO

O sistema bancário, que inclui empresas que prestam serviços financeiros para pessoas físicas e jurídicas, tem utilizado de forma crescente a automatização e informatização de suas atividades. Essa informatização traz inúmeros benefícios para o setor, como por exemplo, a redução de custos operacionais, bem como a realização de operações financeiras de forma mais ágil e simples para os usuários.

Entretanto, o aumento na informatização traz também o aumento de problemas relacionados à utilização dessas novas tecnologias como, por exemplo, o aumento de fraudes e crimes cibernéticos. Segundo relatório de fraudes globais da empresa Kroll para os anos de 2016 e 2017 (CARVALHO, 2016) no Brasil, 89% das empresas do setor financeiro sofreram algum tipo de incidente cibernético (por exemplo, fraudes, ataques a seus sistemas informatizados ou roubos de informação) e 91% dessas empresas acreditam que a exposição à fraude aumentou em 2016/2017. Esses dados evidenciam a grande incidência de eventos fraudulentos em operações financeiras informatizadas.

Com o objetivo de inibir esse número crescente no aumento das fraudes e crimes cibernéticos, podemos citar duas linhas principais de ação. A primeira delas diz respeito ao aumento da segurança nas transações bancárias e acesso a informações. Nesse item a pesquisa e utilização de métodos de criptografia, verificação de identidade do usuário (sistemas biométricos como leitor de digitais ou íris), segurança de redes e comunicação, entre outros. Outro modo de ação, é buscar operações que não são habituais para o usuário verdadeiro. Dessa forma, é possível identificar que uma operação não autorizada está em curso.

Dentro desse contexto, inúmeros trabalhos tem sido desenvolvidos aplicando métodos de inteligência artificial ou aprendizagem de máquinas para identificar padrões não usuais de comportamento em atividades financeiras (DAWEI et al., 2017), (VADOOD-PARAST; RAZAK; S., 2015), (AHMED; MAHMOOD; ISLAM, 2016) e (ABDALLAH; MAAROF; ZAINAL, 2016).

Além da utilização de redes neurais (BEKIREV et al., 2015) para a identificação de comportamentos não usuais para reconhecer fraudes, outro método que tem apresentado bom resultados é o *clustering* de dados (VAISHALI, 2014).

Nessa abordagem os dados de transações financeiras são adequadamente transportados para um espaço de feições N dimensional onde cada ponto nesse espaço representa uma transação financeira. Esses pontos podem ser agrupados em *clusters*, ou seja, em agrupamentos de pontos os quais apresentam características ou feições semelhantes. Atividades não usuais, com maior chance de representar uma atividade fraudulenta, por estarem fora do padrão do usuário original, aparecem com pontos isolados ou distantes dos *clusters* de dados.

Desse modo, podemos classificar os pontos referentes a transações financeiras em no mínimo dois grandes grupos, um referente à atividades genuínas, e outro referente

à atividades ilícitas. No presente trabalho de Trabalho de Conclusão de Curso (TCC), será explorada a utilização de diferentes métodos de *clustering*, como o *K-Means* (MACQUEEN, 1967) e o DBSCAN (MARTIN et al., 1996) para o *clustering* dos dados em agrupamentos de operação financeiras que representam fraude e não fraude (ou normais) e posteriormente vai ser utilizado um método híbrido para validar os resultados obtidos.

O objetivo principal desse TCC é identificar as transações fraudulentas, tornando-se possível distinguí-las das transações normais.

1.1 Objetivos Específicos

- Agrupar as transações fraudulentas, com o mínimo de transações normais possíveis.
- Testar mais de um método de agrupamento.
- Realizar testes diferentes e comparativos.
- Comparar com método supervisionado.
- Propor e testar abordagem híbrida.

2 TRABALHOS RELACIONADOS

Atualmente as medidas de segurança tomadas por instituições bancárias, para assegurar que seus clientes não estejam sendo alvos de nenhuma fraude são baseadas em Inteligência Artificial (IA). O uso de IA abriu as portas para o aprendizado de máquina, também conhecido como *machine learning*, esse subcampo consiste em fazer um reconhecimento de padrões (MARYALENE, 2017). Esse campo de estudo possibilitou o desenvolvimento de diversos métodos, que podem ser utilizados em muitas áreas de estudo, como financeira, processamento de imagens, medicina, entre outras. Alguns dos métodos mais conhecidos são árvores de decisão, classificação *naive bayes*, regressão logística, algoritmos de agregação, redes neurais e aprendizado profundo. A seguir estudos que aplicam algumas dessas técnicas.

ZAMINI; MONTAZER (ZAMINI; MONTAZER, 2018), propôs a utilização de duas classes de aprendizagem de máquina, a aprendizagem não-supervisionada, que abrange algoritmos que não necessitam de dados previamente rotulados e semi-supervisionada, que necessitam apenas que poucas amostras de dados estejam rotuladas. Respectivamente em relação às classes selecionadas, foram escolhidos o algoritmo *KàMeans* e *autoencoder*. *Autoencoders* são redes neurais auto-associativas, isso é, o algoritmo tenta reproduzir os neurônios de entrada na saída. O objetivo do *autoencoder* é comprimir os dados e reduzir sua dimensionalidade, assim sendo possível a extração de recursos úteis e refinando os não úteis. O algoritmo *K-Means* é utilizado para agrupar dados considerados semelhantes, através de cálculos de distância, isso permite diferenciar os grupos em questão. O algoritmo *K-Means* será explicado com mais detalhes no Capítulo 3. Após o *autoencoder* ter reduzido a dimensionalidade da base de dados, os mesmos podem ser agrupados pelo algoritmo *K-Means*. Como resultado final foram agrupadas 75% das fraudes. A base de dados *credit fraude detection*[1] utilizada pelo autor é a mesma utilizada no TCC, ao final será possível fazer uma comparação de resultados.

ASSAD (ASSAD, 2015), propôs a utilização de uma arquitetura de classificação baseada em regras de associação e modelos de pontuação. A classificação baseada em regras pertencem a classe de aprendizagem supervisionada, esse classificador é composto por diversas regras, seguindo o formato, "Se (condição) Então (ação consequente)". As regras criadas por esse método podem ter duas propriedades:

- Conjunto de Regras Completo: Essa propriedade se aplica quando um dos registros é condizente com pelo menos uma regra.

- Regras Mutuamente Excludentes: Essa propriedade se aplica quando não há nenhum registro que seja condizente com mais de uma regra.

A determinação das regras são feitas por dois métodos, são eles:

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

- Métodos Diretos: Extraem as regras direto dos dados. Algumas implementações utilizadas para esse fim são AQ, CN2, PRISM e RIPPER.
- Métodos Indiretos: Utilizam outros métodos de classificação, posteriormente os resultados da classificação são convertidos em regras. Geralmente os classificadores utilizados são árvores de decisão e algoritmos genéticos.

Modelo de pontuação é um método que gera uma pontuação para um determinado evento. Neste caso, tratando-se de transações financeiras, quanto mais alta a pontuação, maior a probabilidade da transação ser fraudulosa. Por esse meio são criadas notas de corte, agrupadas da seguinte forma:

- Transações com nota de corte menor ou igual a primeira são aprovadas imediatamente.
- Se a transação obtiver uma pontuação maior que a primeira nota de corte, porém menor ou igual a segunda nota de corte, um time de especialistas faz uma análise do caso.
- Se a transação obtiver uma pontuação maior que a segunda nota de corte, a transação é negada imediatamente.

Foi utilizado uma base de dados com transações de cartões de crédito entre julho de 2009 e janeiro de 2014, esse conjunto possui 43 variáveis (dimensões), nessa base há 7.716.091 transações, 22.615 são fraudes. Após um processo de preparação de dados foram adicionadas mais 37 variáveis, totalizando 80 na base de dados final. O passo seguinte foi realizar a mineração de regras de associação que mapeiem atributos presentes nas variáveis de transações fraudulosas. Como resultado inicial foram geradas 38.003 regras, isso tornou inviável uma análise regra à regra. Essas regras foram reduzidas e divididas em grupos e aplicadas nas transações, grupos de transações com um volume menor de fraudes, tiveram uma taxa de detecção de fraudes de 94%, grupos com volume de fraudes maior, obtiveram 77% de precisão. Com essa solução sempre podem ser inseridas regras novas e dessa forma, testar transações que são englobadas por essas regras novas.

MÓNICA; LIZBETH (MÓNICA; LIZBETH, 2010) propôs a utilização do método *K-Means* e *Fuzzy K-Means*, para agrupar o comportamento de usuários de cartão de crédito, sem conhecimento prévio dos mesmos. Com isso é possível observar comportamentos atípicos, nos grupos formados. Foi utilizado uma base de dados de duas dimensões, com 3000 amostras. Existem nessa base de dados 5 amostras que podem ser consideradas como fraudes, MÓNICA; LIZBETH realizou diversas iterações no algoritmo para encontrar os melhores centros, como resultado, foram analisados os grupos formados, após plotá-los é visualizável 3 prováveis fraudes.

KAZEMI; ZARRABI (KAZEMI; ZARRABI, 2017) propôs a utilização de *deep autoencoder* para extrair padrões das ocorrências de fraudes mais complexas, foi utilizado

o conjunto de dados *German Credit Data*[2]. Para treinar a rede foram utilizadas 900 transações e 100 amostras para os testes. O *deep autoencoder* foi escolhido para otimizar os recursos desse conjunto de dados, e classifica-los, com isso foi obtido 81,5% de precisão, para fins de comparação também foi implementado o algoritmo k-nearest neighbors (KNN), que obteve 76,1% de precisão, ou seja, o modelo baseado em *deep autoencoder* tem 5% a mais de precisão.

SHAKYA; MAKWANA (SHAKYA; MAKWANA, 2017) propôs a utilização de um método híbrido, que consiste na cadeia de utilização DBSCAN, *K-Means* e *SMO Classifier*. Esses algoritmos foram selecionados para realizar no conjunto de dados selecionados, a redução de ruído com o DBSCAN, após isso foram removidos manualmente os elementos considerados ruído da base de dados, com esses dados já filtrados é realizado o agrupamento com o *K-Means* e são gerados 5 grupos. O *SMO Classifier* substitui valores ausentes, transforma atributos nominais em binário, e normaliza os valores, esse processo foi realizado em cada grupo, a precisão média do método ficou em 96,92%. Foi utilizada a base de dados *KDD'99*[3], essa base é um conjunto de conexões ao qual há intrusões na rede.

MALINI; PUSHPA (MALINI; PUSHPA, 2017) propôs a utilização do algoritmo K-Nearest Neighbor (KNN) e detecção de *outliers* para detectar fraudes de cartão de crédito, o KNN é utilizado para classificar um dado novo, de acordo com a distância desse dado até os dados já classificados, esse algoritmo será explicado melhor na seção 3.2. A detecção de *outliers*, é utilizada para detectar comportamento incomum, elas são classificadas em dois tipos:

- Detecção de *outlier* supervisionada, onde um grupo de especialistas no domínio, modelam um sistema para aprender a classificar os *outliers* através de treinamento.
- Detecção de *outlier* não-supervisionada, onde os dados são agrupados com base nos recursos.

A utilização desas técnicas tem como objetivo minimizar a taxa de alarmes falsos, que outros métodos podem provocar. Em sua aplicação ele extraiu a direção dos dados, e utilizou o KNN para determinar a instancia destino, dessa forma utilizando menos recursos computacionais.

ADA; HARSONO; BASUKI (ADA; HARSONO; BASUKI, 2018) propôs a utilização dos algoritmos *Meng Hee Heng K-Means* e DBSCAN, para segmentar imagens meteorológicas de nuvens, com isso é possível detectar a forma das nuvens. O Meng Hee Heng K-Means separou as nuvens baseado em uma área, mesmo elas não sendo um *cluster*, e o DBSCAN separou as nuvens com base nos *clusters* que elas formam, isso permite contar quantos grupos de nuvens existem.

[2] https://www.kaggle.com/uciml/german-credit
[3] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Esses trabalhos foram selecionados em buscas que utilizaram, as *strings* de busca: *Credit Fraud*, *clustering*, *fraud detection* e *credit fraud detection*.

3 DESENVOLVIMENTO

Nesse TCC é documentada uma proposta, da aplicação de métodos de agrupamento de dados, conhecidos como *clustering*, na problemática de fraudes de cartão de crédito, esse é um problema que segundo ZAMINI; MONTAZER (ZAMINI; MONTAZER, 2018), os danos causados por fraudes em meios de pagamentos eletrônicos já atingiram US $ 7,6 bilhões em 2010, e cresceram para US $ 21,81 bilhões em 2015 e estima-se que essas fraudes chegarão a US $ 31,67 bilhões em 2020 (Valores em dólar americano). Toda transação gera um log (registro de evento relevante) na base de dados, de seu respectivo serviço bancário.

Serviços bancários utilizam diversas combinações de métodos para detectar suas fraudes, porém, a maioria de seus métodos são baseados na classe de aprendizado supervisionado, ou seja, eles sabem o que buscam, e tem modelos treinados para isso. Porém, com o surgimento de novos métodos de fraudes, o modelo que eles utilizam pode demorar meses para aprender a identificar esses novos métodos de fraude. Com a utilização de métodos não-supervisionados, qualquer comportamento fora da curva pode levantar suspeitas, ou seja, agrupando esses comportamentos é possível, que modelos para identificação de fraudes sejam treinados mais rapidamente e, também minimizar erros de identificação de fraudes e não fraudes.

3.1 Fundamentação Teórica

3.1.1 Fraudes de Cartão de Crédito

Atualmente cartões de crédito são uma das modalidades de pagamento mais utilizadas no mundo segundo MIRET; BRUNO (MIRET; BRUNO, 2015), a principal vantagem dessa modalidade de pagamento é sua segurança e o fato de que, não é mais necessário estar sempre com cédulas. O desenvolvimento tecnológico permitiu aumentar a segurança dos cartões de crédito uma vez que foi adotado o uso de chaves Personal Identification Number (PIN). O PIN é um código utilizado para confirmar algumas operações bancárias. Outra medida de segurança que foi adotada são os *chips* de segurança, esses *chips* trazem as informações do cartão codificadas e efetuam algumas análises quando são inseridos nos leitores de cartão, uma delas é a autenticação do leitor de cartão que está sendo utilizado.

Segundo MÓNICA; LIZBETH (MÓNICA; LIZBETH, 2010), para cada dólar gasto com cartão de crédito, 0,05 centavos correspondem a uma fraude. Transações fraudulosas utilizando cartões de crédito, são definidas pelo seu uso sem consentimento e ou autorização do responsável do cartão. Alguns meios utilizados pelos cibercriminosos são (ASSAD, 2015):

- Invasão de Conta: Esse é o método mais básico de fraude realizado, o cibercriminoso obtém acesso a conta da vítima, através da obtenção de dados sensíveis como número

de conta e senha. Com acesso a conta da vítima o cibercriminoso pode solicitar serviços como empréstimos e novos cartões.

- *Skimming*: O método mais conhecido para captura de dados de cartões de crédito é o *skimming* de cartão, ele necessita de um *skimmer* para ler os dados contidos na tarja magnética do cartão. O *skimming* pode ser feito através de leitores de cartões que lêem a tarja magnética dos cartões e fazem a coleta dos dados, outra forma de realizar é implantando *bugs* em caixas eletrônicos através da troca do terminal por um modelo parecido (GUO; JIN, 2010).

- Extrativo: Consiste no roubo do cartão e senha no processo de envio do cartão para o portador.

- Auto-fraude: Ocorre quando o portador do cartão age de má fé e, após realizar uma compra, contata o emissor de cartão e alega não ter feito a compra.

- Roubo de Identidade: É o roubo dados sensíveis das vítimas, seja por métodos simples ou por meio digital, através de cavalos de troia, *phishing* e engenharia social.

O número de fraudes cibernéticas tem aumentado nos últimos anos, e as perspectivas, bem como a percepção dos usuários e profissionais do sistema bancário é de que esse crescimento seja mais pronunciado futuramente, devido ao aumento da informatização e a entrada de novos mercados no sistema econômico eletrônico. Dessa forma, é de grande importância desenvolver metodologias e técnicas para impedir tais fraudes. Entre as medidas de segurança para fraudes eletrônicas, podemos dividí-las em dois grandes grupos. O primeiro, diz respeito a metodologias que visam impedir o acesso não autorizado de usuários ao sistema financeiro. O segundo, já pressupõe que o invasor possui acesso ao sistema, e procura identificar padrões atípicos de comportamento para detectar transações fraudulentas.

3.1.2 Clustering de Dados

Todos os dias, grandes quantidades de dados são analisados por corporações, pesquisas em universidades, ou por pessoas que os estudam. Dados são atributos ou características atribuídas a um determinado objeto, circunstância ou evento. Como há uma abundância de dados associados a seus próprios eventos, a sociedade atual os vê como meio para superar seus concorrentes, e assim como o fazem os resultados são positivos. Porém, dados são abstratos e podem ter uma quantidade alta de características. Para tal tarefa, são utilizadas técnicas de aprendizagem de máquina, que é um dos subcampos da ciência da computação para reconhecimento de padrões e gerar previsões.

Será abordada a técnica de aprendizagem de máquina chamada de *clustering*. O *clustering* de dados é uma técnica de aprendizado de máquina não supervisionado, isso

significa que é uma análise exploratória de dados (técnica para examinar os dados antes da aplicação de qualquer técnica estatística), onde não há conjuntos de dados de treinamento, e há extração de padrões ocultos em dados não rotulados (BINDRA; MISHRA, 2017). A técnica de *clustering* a grosso modo faz o particionamento de um conjunto de dados, de forma que esses dados são agrupados em subconjuntos, esses subconjuntos são conhecidos como *clusters* (HAN; PEI; KAMBER, 2011). O *clustering* aumenta a proximidade dos pontos no conjunto de dados que pertencem ao mesmo *cluster* e aumenta a dissimilaridade entre os outros *clusters*. Conjuntos de dados não possuem a mesma distribuição de dados, para isso existem categorias de algoritmos de *clustering*, que usam outros esquemas de agrupamento. Esses outros esquemas de agrupamento se diferem em desempenho, medidas de similaridade ou dissimilaridade, seleção ou extração de recursos, validação do *cluster* e complexidade (BINDRA; MISHRA, 2017). Os métodos de agrupamento são divididos nas seguintes categorias:

- Método baseado em Particionamento: É a categoria mais básica de agrupamento, há necessariamente dois *inputs*, o conjunto de dados, e o número de *clusters*. Primeiramente, o centro de cada *cluster* é selecionado, e pode ser chamado de centroide. Os centroides podem ser gerados de maneira randômica pelo algoritmo, ou podem ser escolhidos a partir de um ponto qualquer do conjunto de dados. A cada iteração os dados são associados aos *clusters* mais próximos, de acordo com o critério de similaridade. Após os dados estarem associados aos seus *clusters*, os centroides são atualizados (NETO, 2015).

- Método Hierárquico: Métodos hierárquicos criam uma hierarquia ou árvore de relacionamentos entre os dados do conjunto de dados, também conhecido como *dataset*, essa estrutura é conhecida como dendrograma de *cluster*. Quando essa hierarquia é vista como uma árvore, as folhas são os pontos de dados, e a raiz contém os pontos de um *cluster* (RICARDO, 2009)(BINDRA; MISHRA, 2017). Há duas versões desse algoritmo, são elas:

  - Aglomerativo: Cria conjuntos, a partir de elementos isolados. Seguindo os seguintes passos (RICARDO, 2009):
    - É criado um *cluster* para cada dado.
    - De acordo com a medida de similaridade utilizada são criados pares.
    - Esses pares são fundidos formando um *cluster* maior, e a distância deste *cluster* para os outros é recalculada.
    - Repetir passos 2 e 3 até restar apenas um *cluster*.
  - Divisivo: Quebra o conjunto de dados sucessivas vezes. Esse processo é repetido até que hajam apenas conjuntos compostos apenas por um único elemento de dados. No momento da partição de um *cluster* ele não é obrigado a ser

partido em dois, um meio de escolher o número de *clusters* a serem gerados, é utilizando a métrica de otimização de custo de cortes. A função dessa métrica é maximizar a relação entre a similaridade *intra-cluster* (homogeneidade entre os dados dentro desse *clusters*) e *extra-cluster* (Heterogeneidade entre *clusters*) (RICARDO, 2009).

- Método Baseado em Densidade: Métodos de particionamento e hierárquicos conseguem encontrar *clusters* em formato esférico, e tem dificuldade em encontrar *clusters* de forma arbitrária, por esse motivo foram desenvolvidos os métodos baseados na densidade. A ideia desse método é que um *cluster* vai crescer, desde que a densidade (pontos de dados) em sua vizinhança atinja um limite. Esse limite é o número mínimo de dados para que haja um *cluster*, para identificar os pontos presentes na vizinhança, há um raio que delimita até onde os pontos de dados são verificados. Esse método é utilizado com frequência para encontrar pontos discrepantes, também chamados de *outliers*, eles são pontos que não pertencem a nenhuma vizinhança (HAN; PEI; KAMBER, 2011).

- Métodos Baseados em *Grid*: Esse método quantiza o espaço entre os dados em um número finito de células, isso cria uma estrutura de grade. A maioria dos algoritmos de *clustering* possui um tempo proporcional ao tamanho do conjunto de dados, no caso dos métodos baseados em *grid*, dependem apenas do espaço entre os dados quantizados. O *clustering* baseado em *grid* é feito nos seguintes passos (GAN; MA; WU, 2007)(HAN; PEI; KAMBER, 2011):

  - Os dados são particionados, criando um número finito de células com os espaços dos dados.
  - Calcular a densidade de uma célula para a outra.
  - As células são separadas de acordo com suas densidades.
  - Identificar os centroide dos *clusters*.
  - As células vizinhas são visitadas, uma a uma, e são classificadas.

3.1.2.1 Requisitos para Clustering

Como não há um algoritmo que forneça solução para todos problemas de *clustering*, há algumas características que proporcionam resultados melhores, são elas (BINDRA; MISHRA, 2017):

- Escalabilidade: Algoritmos de *clustering* altamente escaláveis são necessários, pois algoritmos que agrupam apenas um conjunto de amostras de dados, podem levar a resultados insatisfatórios. Já trabalhando com várias amostras, os agrupamentos tendem a ser satisfatórios.

- Poder Lidar com Tipos de Atributos: Cada vez mais aplicações precisam fazer o *clustering* de dados complexos, como imagens, gráficos, sequências e documentos. Então os algoritmos de *clustering* precisam ser projetados para trabalhar com tipos de dados diferentes.

- Encontrar *Clusters* de Forma Arbitrária: Alguns algoritmos de *clustering* usam por padrão medidas distância Euclidiana ou distância *Manhattam*, no entanto, essas medidas tendem a encontrar *clusters* esféricos e com densidade semelhantes. Por isso, torna-se importante o desenvolvimento de algoritmos que encontrem *clusters* de forma arbitrária, assim os agrupamentos podem ter formatos mais precisos.

- Conhecimento do Domínio para Selecionar Parâmetros de Entrada: Os resultados do *clustering* são sensíveis aos parâmetros de entrada, eles são difíceis de determinar especialmente em conjuntos de dados de alta dimensionalidade, o número de *clusters*, raio e as medidas de distância são exemplos desses parâmetros de entrada.

- Lidar com *Outliers*: A maioria dos conjuntos de dados do mundo real contém *outliers*, o que não é necessariamente um problema, pois, às vezes, são exatamente esses dados *outliers* que buscamos. Mas também há casos em que esses *outliers* são provenientes de interferências captadas por sensores que coletam esses dados. E, essas leituras erradas podem diminuir a qualidade dos *clusters* durante o processo de *clustering*.

- *Clustering* Incremental e Sensibilidade: Após realizar o *clustering* de um conjunto de dados, podem chegar dados novos a qualquer momento, e alguns algoritmos de *clustering* não suportam incorporar atualizações em uma estrutura de *clustering* existente, ou seja, é necessário recalcular um *clustering* novo com os dados novos. Outro aspecto importante é a ordem do conjunto de dados, ou seja, dependendo da ordem dos dados o processo de *clustering* pode retornar diferenças drásticas.

- Capacidade de Indexar Dados de Alta Dimensionalidade: A maioria dos algoritmos de *clustering* é boa para lidar com conjuntos de dados com poucas dimensões, como duas ou três. Quando há mais dimensões os dados podem estar muito esparsos e distorcidos.

3.1.2.2 Medidas de Similaridade e Dissimilaridade

Medidas de similaridade e dissimilaridade são formas de avaliar como dados iguais ou não iguais são comparados entre si. Um exemplo para aplicar esses conceitos é se houvesse um aglomerado de pessoas, essas pessoas possuem diversas características físicas, pode-se separar essas pessoas em *clusters*, agrupando-as por gênero, familiares, tons de pele, dentre outras características. Esses grupos serão similares entre si, e dissimilares

em comparação com outros *clusters*. Uma medida de similaridade para duas amostras de dados x e y, retornará um valor. Quanto maior for esse valor, maior a similaridade entre as duas amostras de dados, normalmente se o valor for 1, significa similaridade completa, ou seja, as amostras são completamente iguais. Já a medida e dissimilaridade funciona de forma oposta, se o valor retornado pelas duas amostras for 0, significa que eles são os mesmos. Contudo quanto maior o valor retornado, maior a dissimilaridade entre as amostras (BINDRA; MISHRA, 2017). Há diversas métricas que podem ser utilizadas em algoritmos de *clustering*, neste trabalho foram selecionadas as seguintes:

- A distância Euclidiana é uma métrica que descreve a distância em linha reta entre dois pontos no espaço Euclidiano (ALI; AGHABOZORGI; TEH, 2015).

$$d_{euc} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3.1}$$

- A distância de Manhattan é uma métrica que calcula a diferença de característica retilínea, descrita pela soma do valor absoluto de todas as diferenças (ALI; AGHABOZORGI; TEH, 2015).

$$d_{man} = \sum_{i=1}^{n} (x_i - y_i) \tag{3.2}$$

- A distância de Canberra é uma métrica que calcula a soma de uma série de frações da diferença das coordenadas de pares de objetos. Possui grande sensibilidade para valores próximos a zero (ALI; AGHABOZORGI; TEH, 2015).

$$d_{canb} = \sum_{i=1}^{n} |x_i - y_i| \tag{3.3}$$

3.1.3 Método de Clustering: K-Means

O algoritmo de *clustering K-Means*, é um método de aprendizagem de máquina não-supervisionado, sua função é realizar a aproximação de dados semelhantes, dessa forma criando *clusters*, e deixando *clusters* vizinhos dissimilares. O número de grupos que serão formados é definido pelo parâmetro K. É necessário selecionar o método de inicialização do *K-Means*, as inicializações disponíveis são as seguintes (MARSLAND, 2009):

- A inicialização cria pontos de forma randômica.
- A inicialização pode ser selecionada pelo usuário.
- A inicialização pode ser o *K-Means++*, essa é uma inicialização, cujo ponto inicial é escolhido de forma randômica, e os seguintes são escolhidos com probabilidade proporcional a distância.

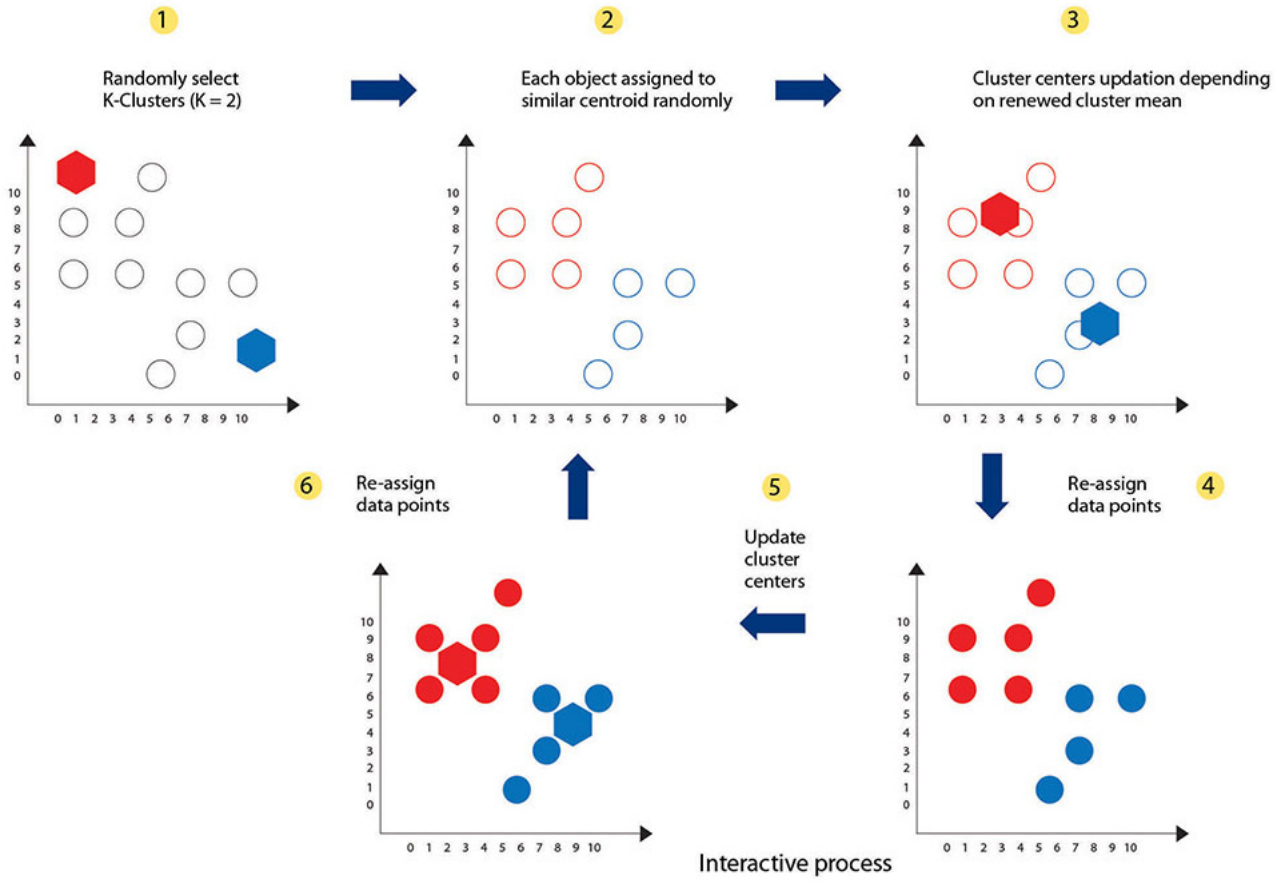
Na Figura 1 são demonstrados os passos do *K-Means* utilizando 2 *clusters*. A utilização de métodos de inicialização randômicos no *K-Means*, pode resultar em agrupamentos diferentes, em execuções diferentes. A execução do algoritmo *K-Means*, funciona em duas etapas, explicadas a seguir (MARSLAND, 2009):

- **Inicialização**
  - Escolha do K.
  - Escolha do método de inicialização.
- **Aprendizado**
  - Para cada ponto, é calculada a distância até os centro dos *clusters*.
  - É atribuído o ponto, para o *cluster* com centro mais próximo.
  - A posição do centro é movida para o ponto médio dos dados.
  - Esses três passos são repetidos, até que a convergência desses pontos estabilize.

Há mais dois parâmetros opcionais, que podem ser escolhidos na fase de inicialização do algoritmo *K-Means*, são eles:

- O número de inicializações (n_init), representa a quantidade de vezes que o algoritmo será executado, para sejam testados mais centroides, gerados pelo método de inicialização.
- A tolerância (tol), verifica se o erro (soma das distâncias dos novos centros à partir dos antigos) é maior que a tolerância, se for a etapa do aprendizado é repetida até que ela caia.

Figura 1 – Demonstração do funcionamento do *K-Means* com 2 *clusters*.



Fonte: (K-MEANS, 2017)

3.1.4 Método de Clustering: DBSCAN

DBSCAN é um método de agrupamento baseado na densidade de dados, seu objetivo é encontrar concentrações de elementos que estão espacialmente próximos. Os grupos são definidos pelo algoritmo da seguinte forma, são buscados pontos que possuam mais que um limiar de vizinhos dentro de um certo raio. Caso um elemento atenda as condições de possuir o número de elementos definido, os vizinhos deste elemento pertencerão ao mesmo *cluster*, esse processo será aplicado a todos os vizinhos. Quando os pontos de um *cluster* alcançam pontos de outro *cluster*, eles são ligados, isso acontece repetidamente enquanto houverem pontos alcançáveis, dessa forma são criadas regiões.

Enquanto o algoritmo *K-Means* agrupa todos os pontos, o algoritmo DBSCAN, pode não agrupar alguns elementos, esses elementos são conhecidos como ruídos, ou *outliers*. Esse método pode ser aplicado a qualquer base de dados contendo dados de um espaço métrico (conjunto onde as distâncias entre quaisquer elemento é definida).

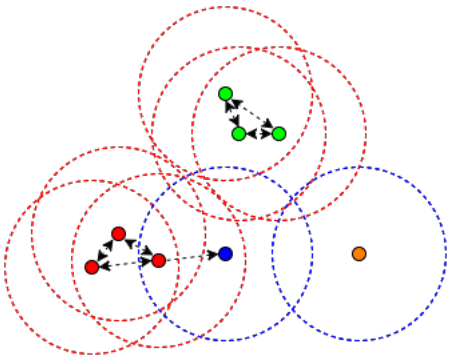
Além de encontrar *clusters* de forma arbitrária, também é capaz de lidar com

ruídos nos dados. Há dois parâmetros de entrada necessários, o *minPoints* que define a quantidade mínima de pontos no raio da vizinhança, e o *EPS* que é o raio ao qual a vizinhança será verificada. Há um parâmetro opcional, que serve para selecionar a métrica que será utilizada para calcular a distância entre os pontos.

O conceito de densidade é abstrato, quando estamos acostumados a métodos de agrupamento particionais, podemos traçar um raio em torno de cada ponto no *dataset*, e definir os pontos em três tipos: (TENDOLKAR, 201-)

- Pontos centrais: Quando há o ponto central cujo raio é traçado a sua volta, e *minPoints* pontos dentro de sua fronteira. Na Figura 2 se o *minPoints* for 3, todos os pontos com limites vermelhos, serão considerados pontos centrais. Os pontos centrais são acessíveis por todos os outros pontos centrais dentro de sua fronteira.
- Não centrais, mas alcançáveis: São pontos que tem menos *minPoints*, dentro de sua fronteira necessários para se tornarem centrais. Na Figura 2, o ponto azul é alcançável por um ponto central, mas não tem pontos suficientes dentro de sua fronteira para ser central. Ele pode ser alcançado pelo ponto vermelho dentro de sua fronteira.
- Outliers: São os pontos que não são centrais e também não são acessíveis a partir de outros pontos, no caso o ponto laranja na Figura 2.

Figura 2 – Demonstração do funcionamento do DBSCAN e como ele mapeia seus pontos.



Fonte: (TENDOLKAR, 201-)

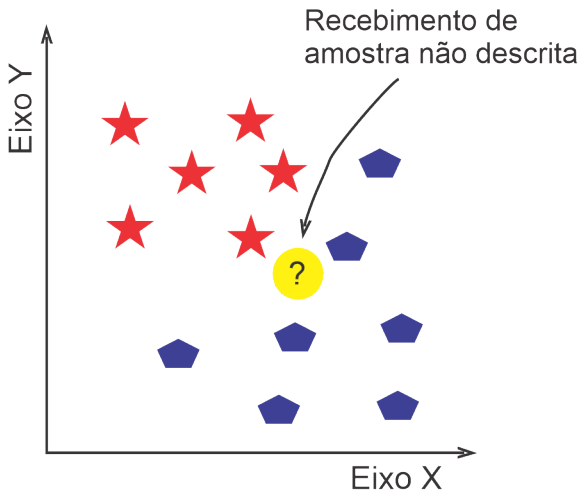
3.2 Método Supervisionado: k-nearest neighbors (KNN)

O KNN é um método de classificação supervisionado, que tem como objetivo encontrar as K amostras descritas. Cada amostra representa um espaço vetorial de k dimensões, quando uma amostra não classificada é descrita em k dimensões, o algoritmo busca a amostra mais próxima da amostra que foi descrita. A semelhança das amostras

são utilizadas como peso, a amostra que foi descrita será classificada no grupo que tiver maior peso (CARLOS, 2009) (JING; GOU; ZHU, 2013). Esse comportamento do KNN pode ser descrito nos seguintes passos:

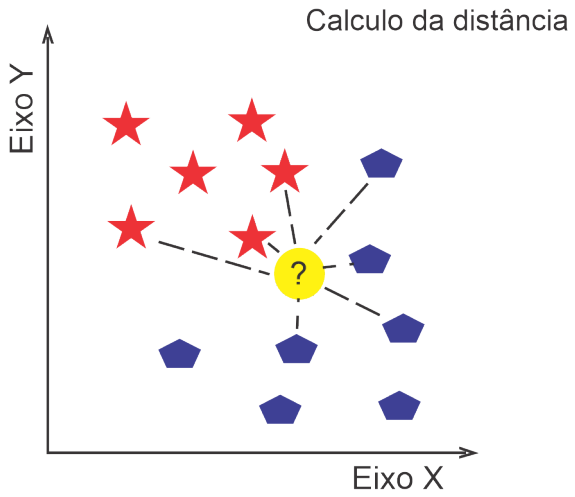
- Recebimento de amostra não descrita, representado na Figura 3.
- É medida a distância da amostra nova, para todos dados classificados. Representado na Figura 4.
- É selecionada a K menor distância. Representado na figura Figura 5.
- É feita a conta de quantas vezes a classe com distância menor aparece em relação à amostra não descrita.
- A amostra não descrita é classificada na classe que aparecer mais vezes.

Figura 3 – Amostra não descrita é inserida.



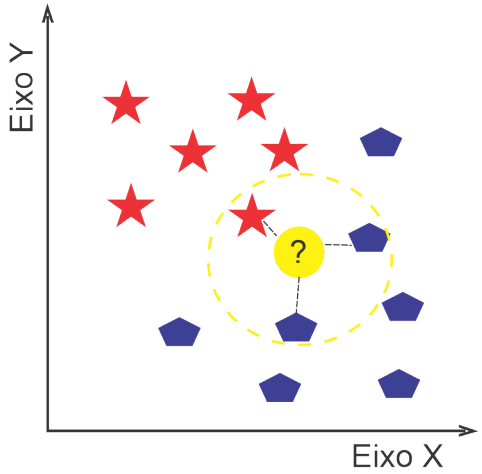
Fonte: Autor

Figura 4 – Distância da amostra nova calculada.



Fonte: Autor

Figura 5 – Amostra encontra vizinhos e calcula os pesos.



Fonte: Autor

4 METODOLOGIA

4.1 Ambiente de Desenvolvimento

O ambiente de desenvolvimento utilizado, foi o *Kernel* fornecido pela plataforma on-line *Kaggle* [1]. *Kaggle* é uma plataforma on-line de cientistas de dados, onde são disponibilizadas várias competições de *data science*, conjuntos de dados, e compartilhamento de conhecimento entre usuários. O *Kernel* on-line oferecido pela plataforma, disponibiliza dois ambientes, o *notebook* e *script*. Foi escolhido o ambiente *Kernel notebook*, para evitar as execuções desnecessárias de trechos de código e visualizar os dados, o ambiente disponibiliza as seguintes especificações técnicas:

- 9 horas de tempo de execução máximo.
- 5 Gigabyte (GB) de espaço em disco.
- 4 Central Processing Unit (CPU) *cores*.
- 17 GB de Random Access Memory (RAM).
- Linguagem de programação: Python 3.6.6 (ROSSUM; JR, 1995).[2]
- Pacotes utilizados:
  - *scikit-learn*: Pacote de métodos de aprendizagem de máquina. (PEDREGOSA et al., 2011).[3]
  - *Numpy*: Pacote para trabalhar com arranjos, listas e matrizes de forma mais eficiente (WALT; COLBERT; VAROQUAUX, 2011).[4]
  - *Pandas*: Pacote para manipulação e análise de dados (WES, 2010).[5]

4.2 Análise e Preparação da Base de Dados

O *dataset* escolhido para esta monografia, foi publicado no *Kaggle*, chama-se "Credit Card Fraud Detection"[6], essa base de dados contém 284.807 transações, 31 colunas, ao total são 8.829.017 valores no *dataset*. A primeira coluna é o *time* (tempo), iniciando em 0, corresponde ao momento em segundos que a transação ocorreu, as colunas de 2 até 29 são dados das transações, eles não são rotulados, a coluna 30 corresponde ao valor dessas transações e a coluna 31 indica se a transação correspondente dessa linha é uma fraude ou transação normal, se o valor for 1 é uma fraude e se for 0 é uma transação normal.

[1] https://www.kaggle.com/kernels
[2] https://www.python.org
[3] https://scikit-learn.org/stable/
[4] https://www.numpy.org
[5] https://pandas.pydata.org
[6] https://www.kaggle.com/mlg-ulb/creditcardfraud

São transações reais de bancos europeus, realizadas em setembro de 2013. Esse *dataset* contém dados desproporcionais, pois 99.83% dos dados são transações normais e 0.17% são fraudes.

Os dados das transações referentes as colunas 2 até 29, passaram por um processo chamado Principal Component Analysis (PCA), esse é um método para redução dimensional de bases de dados, esse processo mantém as características principais, que mais contribuem para sua variação. Os dados que menos contribuem para a variação dos componentes são projetados para o fim das dimensões (PUYATI; WALAIRACHT, 2008).

4.3 Escolha dos Métodos de Clustering

Os métodos de *clustering* foram escolhidos utilizando a Tabela 1, dois fatores de seleção foram escolhidos, são eles a escalabilidade de cada método e seus casos de uso, como estamos lidando com uma base de dados composta por várias dimensões, através do fator escalabilidade foram selecionados aqueles rotulados como "Muito Larga", em relação aos casos de uso, foi selecionado o método *K-Means* por permitir o agrupamento em uma quantidade alta de *clusters* se for necessário, o segundo método selecionado foi o DBSCAN, ele é recomendável pois não estamos trabalhando com geometria plana, e não é conhecido o tamanho dos *clusters* que serão agrupados. O método OPTICS não foi selecionado pois não há uma implementação estável.

Tabela 1 – Tabela de Metodos de Clustering.

Método Nome	**Escalabilidade**	**Caso de uso**
K-Means	Muito Larga	Geometria plana, muitos clusters
Affinity Propagation	Não escalável	Geometria não plana, muitos clusters
Mean-Shift	Não escalável	Geometria não plana, muitos clusters
Spectral Clustering	Média	Geometria não plana, poucos clusters
Ward Hierarchical Clustering	Larga	Poucos clusters
Agglomerative Clustering	Larga	Poucos clusters, restrição para distâncias Euclidianas
DBSCAN	Muito Larga	Geometria não plana, tamanhos irregulares de clusters
OPTICS	Muito Larga	Geometria não plana, tamanhos irregulares de clusters
Gaussian Mixtures	Não escalável	Geometria plana, boa para estimativa de densidade
Birch	Larga	Grandes dataset's, remoção de outliers e redução de dados

Fonte: (PEDREGOSA et al., 2011)

5 RESULTADOS

Para realizar o processo de *clustering*, foram selecionadas as colunas que passaram pelo processo de PCA, que é referente da coluna 2 até a 29, ou seja, estamos trabalhando com uma dimensionalidade igual a 28, nenhuma transação foi removida do processo, ou alterada, pois como vimos na subseção subseção 3.1.2.1, onde é explicada a sensibilidade de *clustering*, qualquer alteração na base de dados pode gerar um resultado diferente. Os algoritmos de *clustering* selecionados, utilizam métodos de particionamento diferentes, ou seja, os parâmetros para realizar o agrupamento são diferentes, foi realizada uma variação de parâmetros, os resultados dessas variações, e seus dados técnicos, estão a seguir. As porcentagens de fraude são referentes apenas ao valor total das fraudes, e a porcentagem das transações normais são referentes ao valor total de transações normais.

5.1 K-Means: Configurações e Resultados

O algoritmo *K-Means*, teve dois parâmetros variados, são eles o número de inicializações e a tolerância. O número de *k clusters* selecionado foi 2, pois o objetivo é realizar um agrupamento de fraudes e não fraudes, por padrão o algoritmo *K-Means* utiliza a métrica Euclidiana.

5.1.1 K-Means: Rotina de Testes 1

Na primeira rotina de testes do *K-Means*, o número de inicializações utilizado foi 1, foram testadas 4 tolerâncias diferentes, são elas: 1, 2, 10, 70. Esses valores foram selecionados por terem proporcionado melhores resultados durante os testes. Comparando os dados resultantes na Tabela 2 e Tabela 3, as tolerâncias que mostraram melhores resultados foram 1 e 70. Utilizando tolerância igual a 1 em seu melhor *cluster* a detecção de fraudes foi de 72,76% e 52,57% de transações normais. O segundo melhor resultado utilizando a tolerância igual a 70, teve 89,02% de fraudes detectadas, e 7,07% de transações normais. Na Figura 8 podemos ver como a detecção de fraudes decresce quando a tolerância é maior que 1, e a partir de 10 ela é crescente, quando a tolerância é maior que 2, a quantidade de transações normais decai, segundo a Figura 9. Resultados presentes também na Figura 6 e Figura 7.

Tabela 2 – *K-Means*: Resultados da primeira rotina, utilizando tolerância 1 e 2, e uma inicialização.

Número de Clusters = 2 || Número de Inicializações = 1								
Cluster nº	Tolerância = 1				Tolerância = 2			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	67	13.61	134568	47,24	348	70,73	146145	51,31
2	358	72,76	149747	52,57	144	29,26	138170	48,51

Fonte: Autor

Figura 6 – *K-Means*: Matriz de confusão, utilizando tolerância 1 e 2, e uma inicialização.

Reais		Previsão: Sim	Previsão: Não
	Sim	358	67
	Não	134568	149747

Reais		Previsão: Sim	Previsão: Não
	Sim	348	144
	Não	146145	138170

Fonte: Autor

Tabela 3 – *K-Means*: Resultados da primeira rotina, utilizando tolerância 10 e 70, e uma inicialização.

Número de Clusters = 2 || Número de Inicializações = 1								
Cluster nº	Tolerância = 10				Tolerância = 70			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	262	53,25	122423	42,98	438	89,02	20153	7,07
2	230	46,74	141892	49,82	54	10,97	264132	92,75

Fonte: Autor

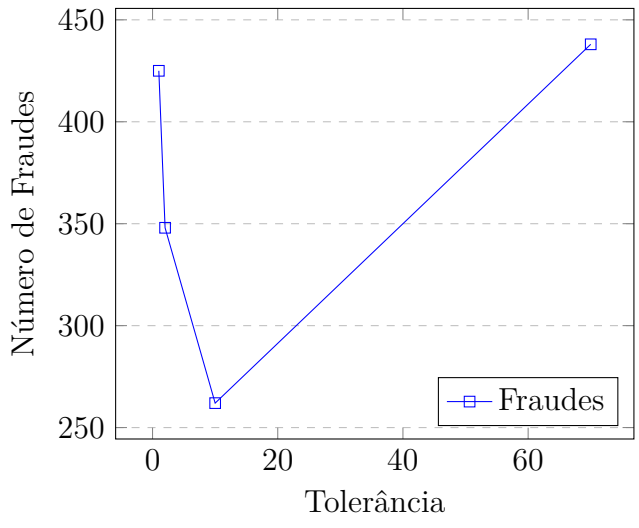
Figura 7 – *K-Means*: Matriz de confusão, utilizando tolerância 10 e 70, e uma inicialização.

Reais		Previsão: Sim	Previsão: Não
	Sim	262	230
	Não	122423	141892

Reais		Previsão: Sim	Previsão: Não
	Sim	438	54
	Não	20153	264132

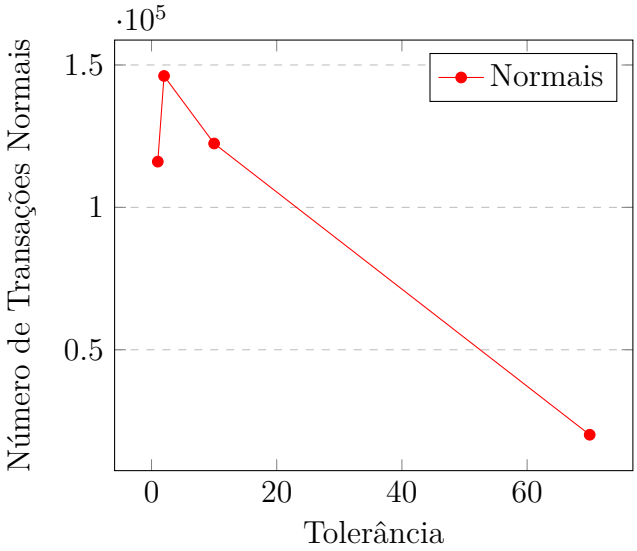
Fonte: Autor

Figura 8 – Detecção de fraudes do *K-Means*, variando a tolerância na primeira rotina



Fonte: Autor

Figura 9 – Transações normais do *K-Means*, variando a tolerância na primeira rotina



Fonte: Autor

5.1.2 K-Means: Rotina de Testes 2

A segunda rotina de testes, teve o número de inicializações definido em 2, a variação de tolerância foi igual aos testes da primeira rotina, os dois melhores resultados foram obtidos com as tolerâncias 2 e 70, utilizando a tolerância 2, foi obtido 97,96% fraudes detectadas, porém 78,32% de transações normais, com a tolerância definida em 70, a taxa de detecção foi de 90,24% e a taxa de transações normais foi de 40,65% , dados de todos os testes estão presentes na Tabela 4 e Tabela 5. Na Figura 12, o pico de fraudes detectadas foi com a tolerância definida em 2, quando é definida em 10 a taxa de detecção cai. Segundo a Figura 13, o menor número de transações normais foi detectado com a tolerância definida em 70. Resultados presentes também na Figura 10 e Figura 11.

Tabela 4 – *K-Means*: Resultados da primeira rotina, utilizando tolerância 1 e 2, e duas inicialização.

Cluster nº	Número de Clusters = 2 || Número de Inicializações = 2 Tolerância = 1				Tolerância = 2			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	357	72,56	149651	52,54	482	97,96	223073	78,32
2	135	27,43	134664	47,28	10	2,03	61242	21,50

Fonte: Autor

Figura 10 – *K-Means*: Matriz de confusão, utilizando tolerância 1 e 2, e duas inicializações.

		Previsão	
		Sim	Não
Reais	Sim	357	137
	Não	149651	134664

		Previsão	
		Sim	Não
Reais	Sim	438	54
	Não	20153	264132

Fonte: Autor

Tabela 5 – *K-Means*: Resultados da primeira rotina, utilizando tolerância 10 e 70, e duas inicialização.

Número de Clusters = 2 || Número de Inicializações = 2								
Cluster nº	Tolerância = 10				Tolerância = 70			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	111	22,56	135618	47,61	444	90,24	115786	40,65
2	381	77,43	148697	52,2	48	9,75	168529	59,17

Fonte: Autor

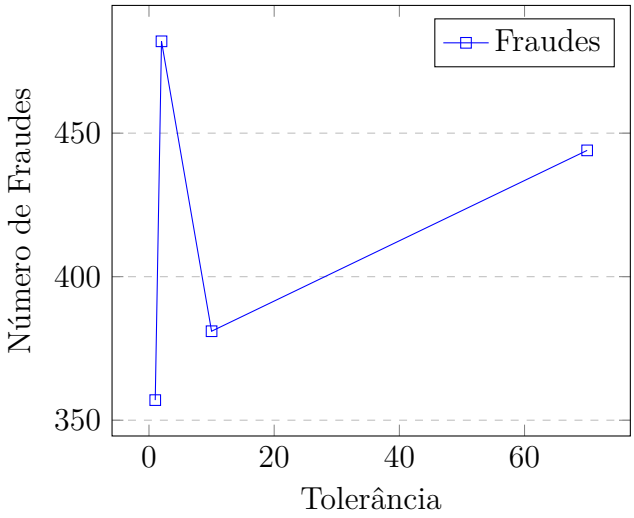
Figura 11 – *K-Means*: Matriz de confusão, utilizando tolerância 10 e 70, e duas inicializações.

		Previsão	
		Sim	Não
Reais	Sim	381	111
	Não	135618	148697

		Previsão	
		Sim	Não
Reais	Sim	444	48
	Não	115786	168529

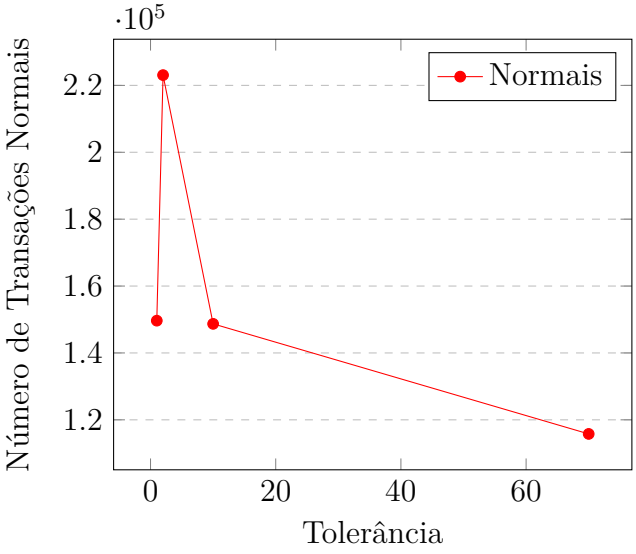
Fonte: Autor

Figura 12 – Detecção de fraudes *K-Means* na segunda rotina



Fonte: Autor

Figura 13 – Transações normais *K-Means* na segunda rotina



Fonte: Autor

5.2 DBSCAN: Configurações e Resultados

O algoritmo DBSCAN teve dois parâmetros principais variados, o EPS e a métrica, a métrica influencia diretamente o EPS, o número mínimo de pontos necessário para haver um *cluster* foi setado em 56 para todos os testes.

5.2.1 DBSCAN com Métrica Euclidiana

Utilizando a métrica Euclidiana há uma fragmentação em diversas regiões de dimensionalidade, foram selecionadas as duas melhores, apresentadas na Tabela 6, utili-

zando o EPS igual a 2, foram detectadas 94,91% das fraudes e 26% de transações normais, aumentando o EPS para 3 a taxa de detecção de fraudes diminui para 87,80% mas a taxa de transações normais também diminuiu para 7,92%, se o EPS for aumentado não ocorre a separação de regiões. Na Figura 15 e Figura 16 mostradas a detecção de fraudes e detecção de transações normais respectivamente variando em relação ao EPS. Resultados presentes também na Figura 14.

Tabela 6 – Resultados do DBSCAN utilizando distância Euclidiana com EPS 2 e 3.

Região nº	Métrica de Distância: Euclidiana EPS = 2				EPS = 3			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	467	94,91	74058	26,00	432	87,80	22560	7,92
2	11	2,23	74456	26,14	54	10,97	228764	80,32

Fonte: Autor

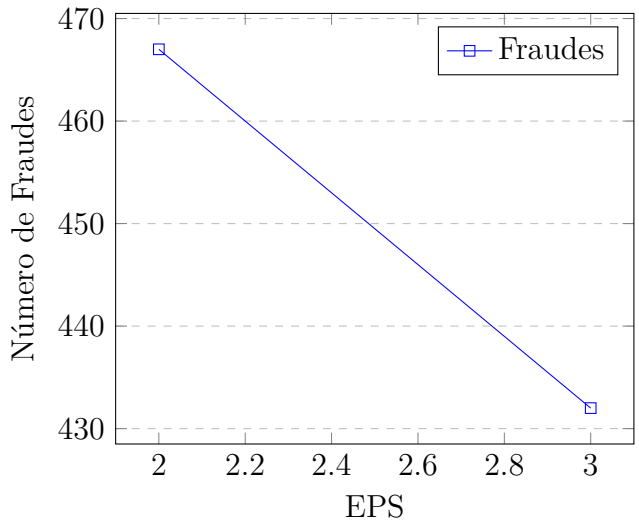
Figura 14 – DBSCAN: Matriz de confusão utilizando distância Euclidiana com EPS 2 e 3.

		Previsão	
		Sim	Não
Reais	Sim	467	11
	Não	74058	74456

		Previsão	
		Sim	Não
Reais	Sim	432	54
	Não	228764	22560

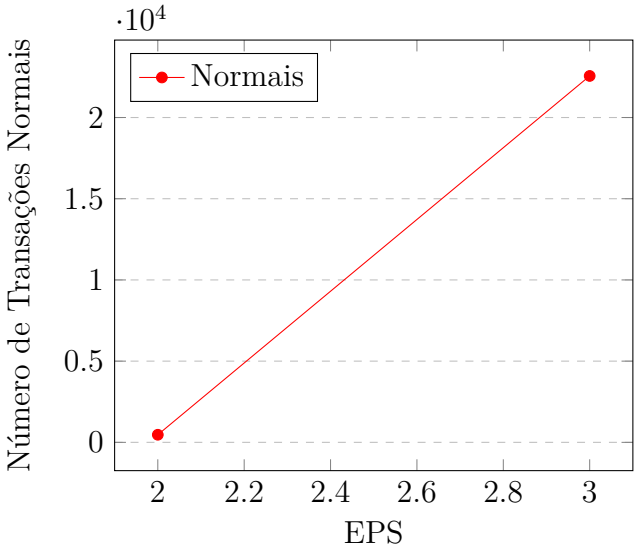
Fonte: Autor

Figura 15 – Detecção de fraudes utilizando DBSCAN e métrica Euclidiana



Fonte: Autor

Figura 16 – Transações normais utilizando DBSCAN e métrica Euclidiana



Fonte: Autor

5.2.2 DBSCAN com Métrica Canberra

Utilizando a métrica *Canberra* foram obtidas três regiões nas três rotinas de testes. O parâmetro de variação foi o EPS, foram feitos rotinas utilizando o EPS igual a 10, assim foram detectadas 50,4% de fraudes e 19,21% de transações normais, com o EPS igual a 11 houve detecção de 64,22% de fraudes e 0,02% de transações normais, a ultima rotina utilizou o EPS igual a 12, foram detectadas 69,10% de fraude e 37,17% de transações normais, os dados completos estão na Tabela 7 e Tabela 8. Na Figura 18 fica visível que o número de fraudes detectadas aumenta conforme o EPS é aumentado e na Figura 19 o número de transações normais cai muito quando o EPS vai de 10 para 11, mas de 11 para 12 a mudança é baixa. Resultados presentes também na Figura 17.

Tabela 7 – Resultados do DBSCAN utilizando distância Canberra com EPS 10 e 11.

Métrica de Distância: Canberra								
Região nº	EPS = 10				EPS = 11			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	248	50,40	54712	19,21	105	21,34	18666	6,55
2	49	9,95	228950	80,3	71	14,43	265440	93,19
3	195	39,63	45	0,01	316	64,22	61	0,02

Fonte: Autor

Tabela 8 – Resultados do DBSCAN utilizando distância Canberra com EPS 12.

Métrica de Distância: Canberra Região nº	EPS = 12			
	Fraudes	%	Normais	%
1	132	26,82	133853	46,99
2	20	4,06	44583	15,65
3	340	69,10	105879	37,17

Fonte: Autor

Figura 17 – DBSCAN: Matriz de confusão utilizando distância Canberra com EPS 10, 11 e 12.

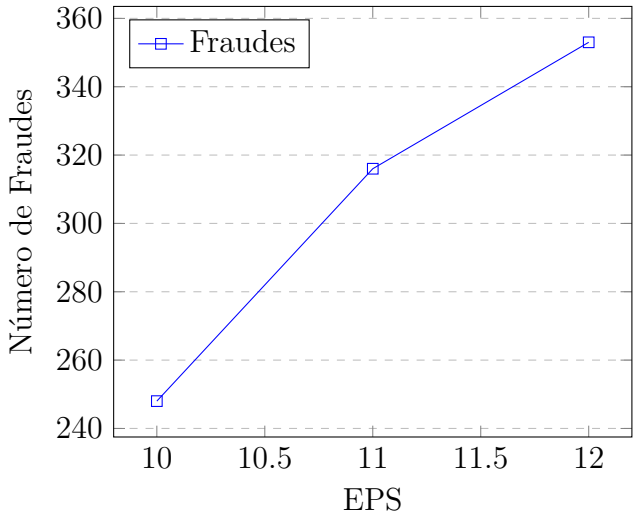
		Previsão	
		Sim	Não
Reais	Sim	248	195
	Não	45	54712

		Previsão	
		Sim	Não
Reais	Sim	316	105
	Não	18666	61

		Previsão	
		Sim	Não
Reais	Sim	340	132
	Não	133853	105879

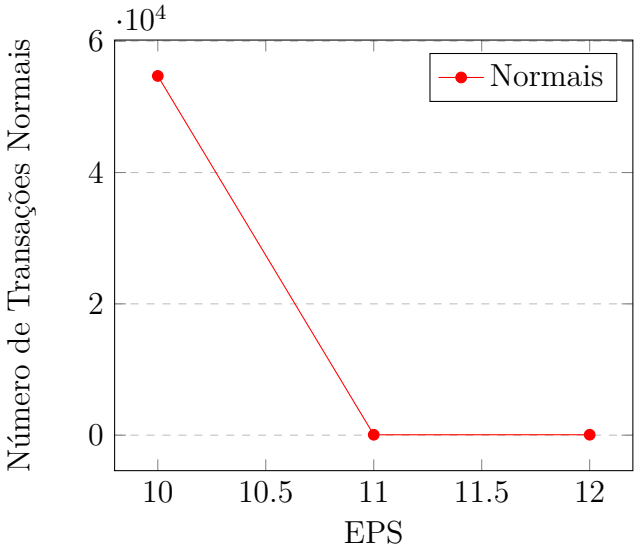
Fonte: Autor

Figura 18 – Detecção de fraudes utilizando DBSCAN e métrica Canberra



Fonte: Autor

Figura 19 – Transações normais utilizando DBSCAN e métrica Canberra



Fonte: Autor

5.2.3 DBSCAN com Métrica Manhattan

Utilizando a métrica *Manhattan*, o EPS foi variado de 9 até 11, utilizando o EPS igual a 9, foram encontradas 91,66% das transações fraudulentas e 17,35% de transações normais, com EPS igual a 10 foram detectadas 89,83% de transações fraudulentas e 12,71% de transações normais, a última rotina com EPS igual a 11, foram detectadas 88,61% das transações fraudulentas e 9,36% de transações normais, mais detalhes estão presentes na Tabela 9 e Tabela 10. Observando a Figura 21, podemos ver que conforme o EPS vai aumentando o número de fraudes é reduzido, e na Figura 22 é visível que o número de transações normais detectadas diminui conforme o EPS é aumentado. Resultados presentes também na Figura 20.

Tabela 9 – Resultados do DBSCAN utilizando distância Manhattan com EPS 9 e 10.

Região nº	Métrica de Distância: Manhattan EPS = 9				EPS = 10			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	451	91,66	49416	17,35	442	89,83	36216	12,71
2	33	6,7	160882	56,48	37	7,52	168892	59,3
3	14	2,84	19914	6,99	2	0,4	20825	7,31

Fonte: Autor

Tabela 10 – Resultados do DBSCAN utilizando distância Manhattan com EPS 11.

Métrica de Distância: Manhattan Região nº	EPS = 11			
	Fraudes	%	Normais	%
1	436	88,61	26684	9,36
2	50	10,16	241863	84,92
3	6	1,21	151	0,05

Fonte: Autor

Figura 20 – DBSCAN: Matriz de confusão utilizando distância Manhattan com EPS 9, 10 e 11.

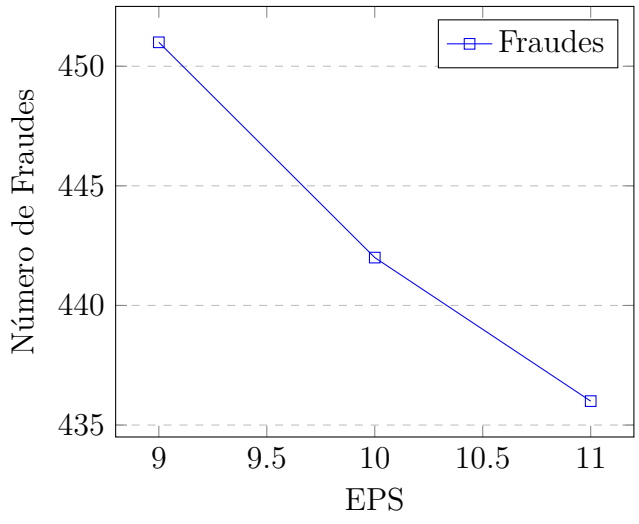
		Previsão	
		Sim	Não
Reais	Sim	451	33
	Não	160882	49416

		Previsão	
		Sim	Não
Reais	Sim	442	37
	Não	168892	36216

		Previsão	
		Sim	Não
Reais	Sim	436	50
	Não	241863	26684

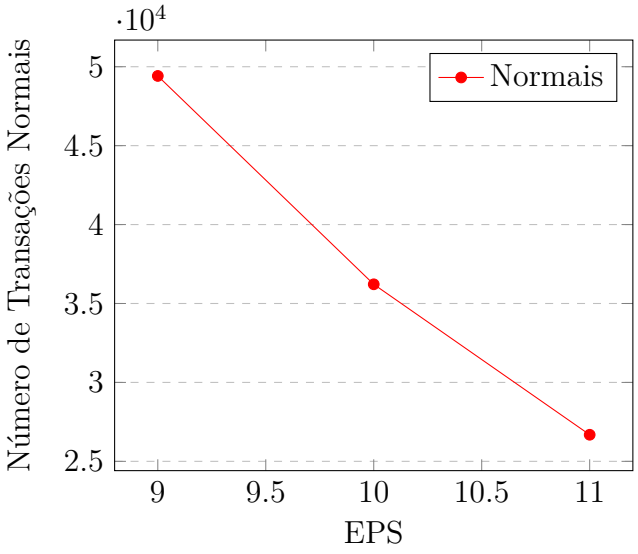
Fonte: Autor

Figura 21 – Detecção de fraudes utilizando DBSCAN e métrica Manhattan



Fonte: Autor

Figura 22 – Transações normais utilizando DBSCAN e métrica Manhattan



Fonte: Autor

5.3 Modelo Híbrido: DBSCAN + K-Means

Esse modelo consiste na utilização dos métodos DBSCAN e *K-Means* em conjunto, as seguintes etapas são seguidas:

- O algoritmo DBSCAN é aplicado na base de dados.
- Os centros detectados pelo DBSCAN são copiados para uma lista.
- Ao invés de utilizar método de inicialização "*K-means++*" no *K-means*, a lista de centros do DBSCAN é utilizada no lugar.

Como há vários centros contidos nas regiões de dimensionalidade do DBSCAN, a lista de centros faria o *K-Means* fragmentar em vários *clusters*. Para contornar isso foram selecionados os primeiros 50 centros para o teste, mas são apresentados apenas os com maior relevância. Para esses testes, os parâmetros do DBSCAN foram baseados nos testes utilizando a métrica *Manhattan*, o algoritmo *K-Means* recebeu 50 *clusters* como parâmetro, e tolerância igual a 50. Segundo as Tabela 11 e Tabela 12, a utilização dessa técnica não se mostrou satisfatória, utilizando o EPS igual a 9, a detecção de fraude foi 39,43% e 28,03% de transações normais, com a utilização do EPS igual a 10 e 11, os resultados foram iguais, a taxa de fraudes detectadas foi de 66,43% e 29,1% de transações normais. Observando as Figura 24 e Figura 25, é notável que a taxa de fraudes cresce junto com a taxa de transações normais, e estabilizam quando o EPS é maior ou igual a 10. Resultados presentes também na Figura 23.

Tabela 11 – Resultados utilizando DBSCAN para gerar centros no *K-Means*.

Modelo Híbrido: DBSCAN + *K-Means*								
cluster nº	EPS = 9				EPS = 10			
	Fraudes	%	Normais	%	Fraudes	%	Normais	%
1	109	22,15	56987	20	327	66,46	82901	29,10
2	189	38,41	147490	51,78	156	31,7	144716	50,81
3	194	39,43	79838	28,03	9	1,82	56698	19,9

Fonte: Autor

Tabela 12 – Resultados utilizando DBSCAN para gerar centros no *K-Means*.

Modelo Híbrido: DBSCAN + *K-Means*				
cluster nº	EPS = 11			
	Fraudes	%	Normais	%
1	327	66,43	82901	29,1
2	156	31,7	144716	50,81
3	9	1,82	56698	19,90

Fonte: Autor

Figura 23 – Resultados utilizando matriz de confusão no DBSCAN para gerar centros no *K-Means*.

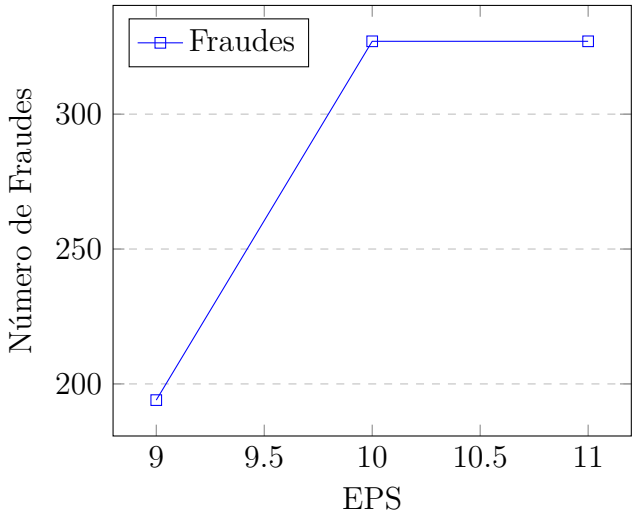
		Previsão	
		Sim	Não
Reais	Sim	194	189
	Não	147490	79838

		Previsão	
		Sim	Não
Reais	Sim	327	156
	Não	144716	82901

		Previsão	
		Sim	Não
Reais	Sim	327	156
	Não	144716	82901

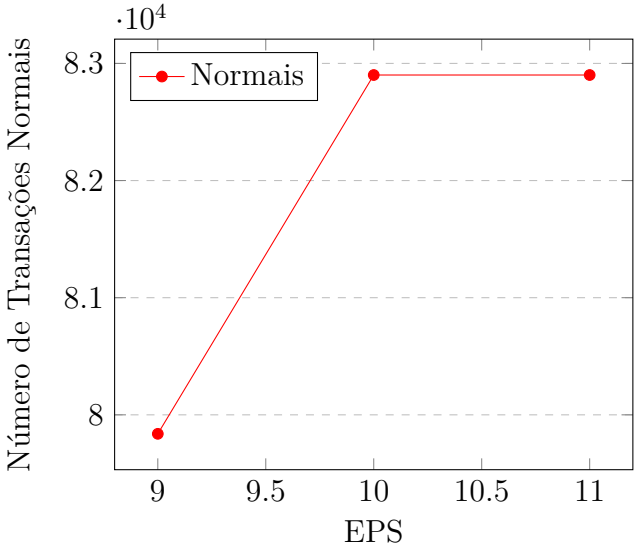
Fonte: Autor

Figura 24 – Detecção de fraudes utilizando modelo híbrido: DBSCAN + *K-Means*



Fonte: Autor

Figura 25 – Transações normais utilizando modelo híbrido: DBSCAN + *K-Means*



Fonte: Autor

5.4 Resultado do KNN

Para fins de comparação com um método supervisionado, foi testado o algoritmo KNN, como feito nos métodos de *clustering*, não houve nenhuma alteração ou remoção no conjunto de dados. Foram utilizados 213.605 valores para o treinamento do algoritmo, os testes foram realizados utilizando os 71.202 valores restantes, 120 desses valores correspondem a fraudes. O método classificou corretamente 78,33% das fraudes, os outros 21,66% foram classificados como falsos positivos, ou seja, foram classificados como tran-

sações normais.Houve um consumo de memória RAM equivalente a 721 MB e tempo de execução igual a 187 segundos. Resultados presentes na Figura 26.

Figura 26 – Resultados utilizando matriz de confusão no KNN.

		Previsão	
		Sim	Não
Reais	Sim	71079	3
	Não	26	94

Fonte: Autor

5.5 Clustering Híbrido

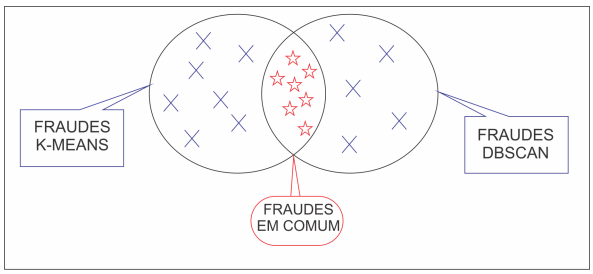
Com objetivo de realizar mais um caso de teste, foi selecionado a melhor configuração de resultados do algoritmo *K-Means* e DBSCAN. Com esses dois casos de teste foi feita uma interseção entre os conjuntos de fraudes, para que dessa forma haja uma validação se ambos algoritmos agruparam as mesmas transações fraudulentas. Na Figura 27 é representada a ideia da interseção dos conjuntos de fraudes. Para o algoritmo *K-Means* foi utilizada a seguinte configuração:

- Número de inicializações: 1.
- Tolerância: 70.

As configurações utilizadas no DBSCAN foram:

- Métrica utilizada: *Manhattan*.
- EPS: 11.

Figura 27 – Representação da união entre o conjunto *K-Means* e DBSCAN.



Fonte: Autor

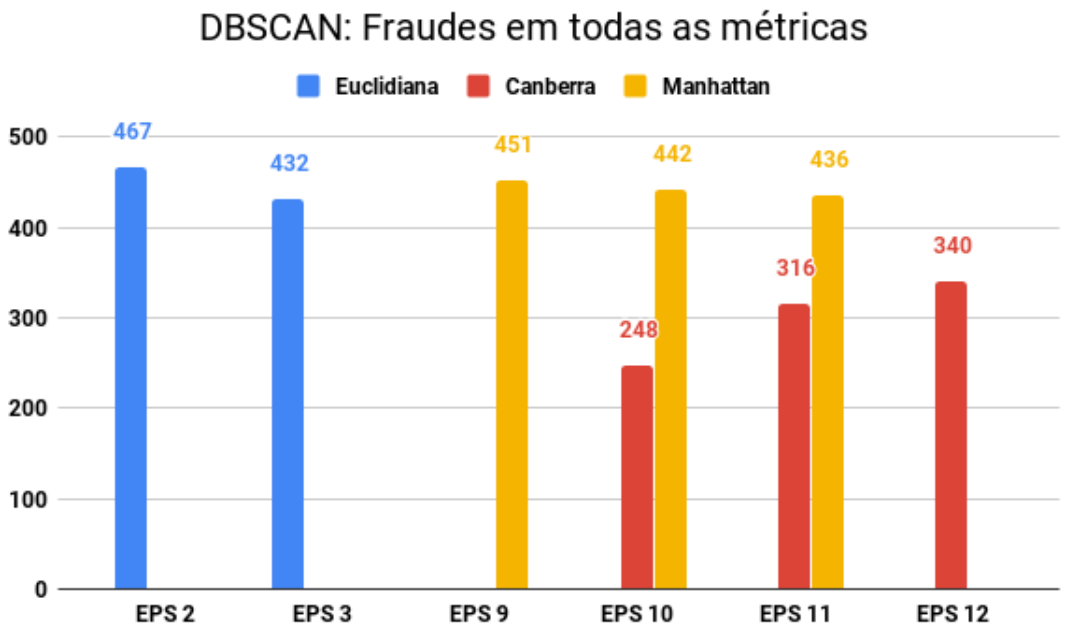
Nessa execução nos melhores *clusters* o *K-Means* agrupou 68,9% das fraudes e o DBSCAN agrupou 88,61% das fraudes. Aplicando a interseção nesses métodos foram

encontrados 307 transações fraudulentas em comum, isso é, 62,39% de transações fraudulentas. Isso não quer dizer que as outras transações fraudulentas presentes nesses *clusters* são falsas, mas sim que elas podem estar agrupadas em outros *clusters*.

5.6 Comparação de Resultados

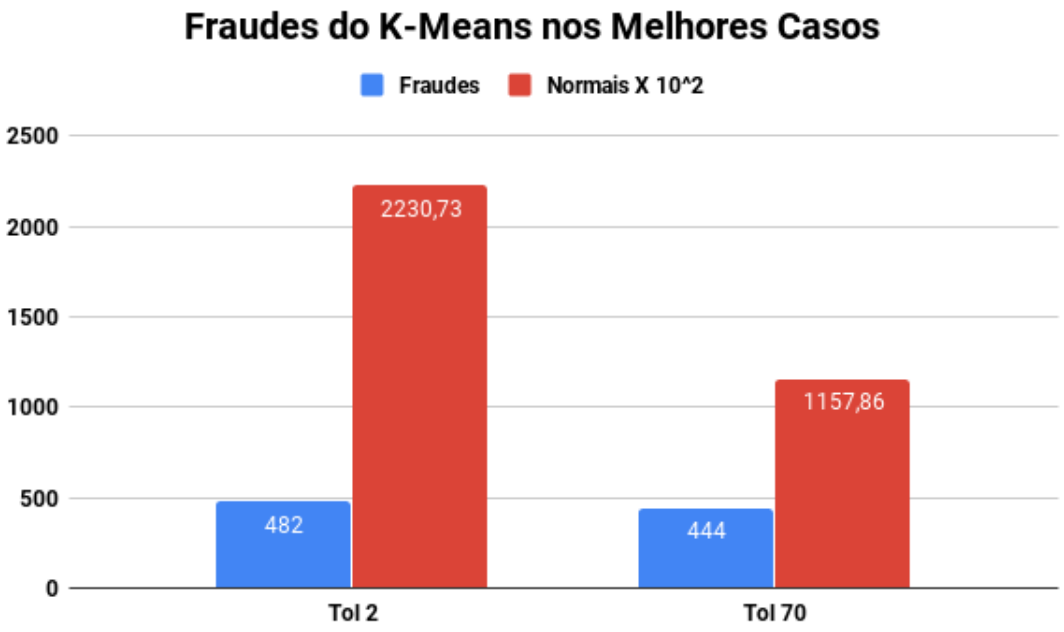
O algoritmo DBSCAN teve melhores resultados, utilizando as métricas Euclidiana e *Manhattan*, utilizando elas as melhores regiões de *clusters* tiveram a taxa de detecção de fraudes superior a 85% e a taxa de transações normais foi de no máximo 26%, não foi possível elevar o valor do EPS para obter melhores resultados, pois o consumo de memória RAM excedia o limite disponível. Segue na Tabela 13 dados do consumo de memória e tempo das execuções. Na Figura 28 é apresentado uma comparação da detecção entre todas as métricas e na Figura 29, é apresentada uma comparação dos melhores casos do *K-Means*.

Figura 28 – Resultados gerais da detecção de fraudes utilizando DBSCAN.



Fonte: Autor

Figura 29 – Resultados gerais da detecção de fraudes utilizando *K-Means*.



Fonte: Autor

Tabela 13 – DBSCAN: Consumo de memória e tempo.

DBSCAN: Recursos Consumidos			
Métrica	EPS	Memória RAM	Tempo
Euclidiana	2	2.5 GB	8 horas
	3	8.8 GB	2.36 horas
Canberra	10	1.2 GB	5,84 horas
	11	1.6 GB	5,89 horas
	12	2.4 GB	6,14 horas
Manhattan	9	3.1 GB	3,29 horas
	10	5.6 GB	3,35 horas
	11	7.1 GB	3,41 horas

Fonte: Autor

O algoritmo *K-Means* teve maior taxa de detecção de fraudes, nos melhores *clusters* de mais de 72% e com as tolerâncias 2 e 70 a taxa de fraudes detectadas foi mais de 90%, infelizmente com a tolerância 2, foram agrupadas 78% de transações normais também. Na Tabela 14, são apresentados dados técnicos dos testes, como o tempo de execução do algoritmo e a memória RAM utilizada.

Tabela 14 – K-Means: Consumo de memória e tempo.

K-Means: Recursos Consumidos				
Rotina	N_init	Tolerância	Memória	Tempo
1ª	1	1	825 Mb	0,8 segundos
	1	2	843 Mb	0,6 segundos
	1	10	840 Mb	1 segundo
	1	70	853 Mb	0,49 segundos
2ª	2	1	846 Mb	0,83 segundos
	2	2	860 Mb	0,82 segundos
	2	10	827 Mb	078 segundos
	2	70	900 Mb	0,69 segundos

Fonte: Autor

A técnica de utilizar os centros dos *clusters* do DBSCAN no *K-Means* teve resultado inferior a utilização dos algoritmos individualmente, foram agrupadas apenas 66% de transações fraudulentas, em contraponto, a taxa de transações normais foi de 29%.

A utilização do *clustering* híbrido, é uma forma de realizar uma validação mútua, infelizmente como o *K-Means* agrupou menos fraudes nessa execução, não foi possível comparar com todas as fraudes do DBSCAN. Em relação a todos os testes realizados, levando em conta a taxa de fraudes detectadas, e o menor número de transações normais agrupadas, o algoritmo DBSCAN em conjunto com as métricas Euclidiana e *Manhattan*, teve melhores resultados, porém, deixa a desejar, em tempo de execução e consumo de memória, isso limitou testes com parâmetros diferentes.

6 CONIDERAÇÕES FINAIS

6.1 Conclusão

O desenvolvimento no presente estudo, teve como objetivo, a utilização de métodos de *clustering*, para detectar comportamentos atípicos, em uma base de dados, composta por transações de cartões de crédito. Ao perceber que a quantidade de fraudes é muito desproporcional em relação as transações normais, percebeu-se que seria difícil realizar agrupamentos eliminando completamente as transações normais, mas foi possível minimizar esse problema até o limite que os recursos computacionais permitiram. Ambos métodos de *clustering* utilizados se mostraram eficientes, mas ambos com seus problemas, o *K-Means* pode gerar agrupamentos diferentes em cada execução, mas recorrentemente ele proporciona o resultado esperado, o DBSCAN teve bons resultados em relação a proposta do trabalho, mas demanda de muitos recursos, o que limita os testes. Como foi mostrado no decorrer do trabalho, as medidas de segurança precisam ser aprimoradas para o que elas não foram treinadas ainda. A quantidade de informações exposta cresce cada vez mais, e isso impulsiona a quantidade de fraudes bancárias realizadas. Esse estudo mostrou que essas transações fraudulentas podem ser identificadas utilizando algoritmos de *clustering* e essa é uma área com potencial de desenvolvimento. Nos melhores casos a taxa de fraudes detectadas foi superior a 88%.

6.2 Trabalhos Futuros

Como possíveis trabalhos proponho:

- Realizar o mesmo experimento com melhores recursos, para otimização do tempo, e uma variação de parâmetros mais abrangente.
- A utilização do algoritmo OPTICS, no presente estudo.
- A utilização dos *clusters* com mais fraudes para realizar o treinamento de algoritmos supervisionados.

REFERÊNCIAS

ABDALLAH, A.; MAAROF, M. A.; ZAINAL, A. Fraud detection system: A survey. **Journal of Network and Computer Applications**, v. 2016, n. 68, p. 90 – 113, jun. 2016. ISSN 1084-8045. Citado na página 17.

ADA, N.; HARSONO, T.; BASUKI, A. Cloud satellite image segmentation using meng hee heng k-means and dbscan clustering. **International Electronics Symposium on Knowledge Creation and Intelligent Computing (IES-KCIC)**, v. 2018, n. 5, p. 367–371, Oct 2018. Citado na página 21.

AHMED, M.; MAHMOOD, A. N.; ISLAM, M. R. A survey of anomaly detection techniques in financial domain. **Future Generation Computer Systems**, v. 2016, n. 55, p. 278 – 288, fev. 2016. ISSN 0167-739X. Citado na página 17.

ALI, S. S.; AGHABOZORGI, S.; TEH, Y. W. A comparison study on similarity and dissimilarity measures in clustering continuous data. **PLOS ONE**, Public Library of Science, v. 10, n. 12, p. 1–20, 12 2015. Citado na página 28.

ASSAD, O. P. H. M. **Detecção de fraudes em cartões: um classificador baseado em regras de associação e regressão logística**. Dissertação (Pós-Graduação em Ciência da Computação) — Universidade de São Paulo, São Paulo, jan. 2015. Citado 2 vezes nas páginas 19 e 23.

BEKIREV, A. S. et al. Payment card fraud detection using neural network committee and clustering. **Optical Memory and Neural Networks**, v. 24, n. 3, p. 193–200, Jul 2015. ISSN 1934-7898. Citado na página 17.

BINDRA, K.; MISHRA, A. A detailed study of clustering algorithms. **International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)**, v. 2017, n. 6, p. 371–376, Sep. 2017. Citado 3 vezes nas páginas 25, 26 e 28.

CARLOS, F. **Algoritmo kNN para previsão de dados temporais: funções de previsão e critérios de seleção de vizinhos próximos aplicados a variáveis ambientais em limnologia**. 129 p. Dissertação (Mestrado em Ciências) — Universidade de São Paulo - USP, São Carlos, mar. 2009. Citado na página 32.

CARVALHO, J. **Índice da Kroll aponta que 89% das empresas brasileiras já sofreram fraude cibernética**. [S.l.], 2016. Disponível em: <https://ipnews.com.br/indice-da-kroll-aponta-que-89-das-empresas-brasileiras-ja-sofreram-fraude-cibernetica/>. Acesso em: 2018-20-07. Citado na página 17.

DAWEI, Z. et al. Discovering rare categories from graph streams. **Data Mining and Knowledge Discovery**, v. 31, n. 2, p. 400–423, Mar 2017. ISSN 1573-756X. Disponível em: <https://doi.org/10.1007/s10618-016-0478-6>. Citado na página 17.

GAN, G.; MA, C.; WU, J. **Data clustering: theory, algorithms, and applications**. [S.l.]: SIAM, 2007. v. 20. Citado na página 26.

GUO, H.; JIN, B. Forensic analysis of skimming devices for credit fraud detection. **IEEE International Conference on Information and Financial Engineering**, v. 2010, n. 2, p. 542–546, Sep. 2010. Citado na página 24.

HAN, J.; PEI, J.; KAMBER, M. **Data mining: concepts and techniques**. 3. ed. [S.l.]: Elsevier, 2011. Citado 2 vezes nas páginas 25 e 26.

JING, Y.; GOU, H.; ZHU, Y. An improved density-based method for reducing training data in knn. **International Conference on Computational and Information Sciences**, v. 2013, n. 4, p. 972–975, June 2013. Citado na página 32.

K-MEANS. **K-Means and X-Means Clustering**. [S.l.], 2017. Disponível em: <https://www.brandidea.com/kmeans.html>. Acesso em: 2018-15-08. Citado na página 30.

KAZEMI, Z.; ZARRABI, H. Using deep networks for fraud detection in the credit card transactions. **IEEE International Conference on Knowledge-Based Engineering and Innovation (KBEI)**, v. 2017, n. 4, p. 0630–0633, Dec 2017. Citado na página 20.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. **Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics**, University of California Press, Berkeley, Calif., v. 1967, n. 1, p. 281–297, jul. 1967. Citado na página 18.

MALINI, N.; PUSHPA, M. Analysis on credit card fraud identification techniques based on knn and outlier detection. **International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)**, v. 2017, n. 3, p. 255–258, fev. 2017. Citado na página 21.

MARSLAND, S. **Machine Learning: An Algorithmic Perspective**. 1st. ed. [S.l.]: Chapman & Hall/CRC, 2009. Citado 2 vezes nas páginas 28 e 29.

MARTIN, E. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. **KDD'96 Proceedings of the Second International Conference on Knowledge Discovery and Data Mining**, v. 1996, n. 34, p. 226–231, ago. 1996. Citado na página 18.

MARYALENE, L. **How Banks Are Working to Protect You From Fraud**. [S.l.], 2017. 1 p. Disponível em: <https://money.usnews.com/banking/articles/how-banks-are-working-to-protect-you-from-fraud>. Acesso em: 2019-20-04. Citado na página 19.

MIRET, R.; BRUNO, V. **52 milhões de brasileiros usam o cartão de crédito como forma de pagamento, diz SPC Brasil**. [S.l.], 2015. 2 p. Disponível em: <https://www.spcbrasil.org.br/pesquisas/pesquisa/936>. Acesso em: 2019-10-05. Citado na página 23.

MÓNICA, G.; LIZBETH, Q. **Detección de fraudes usando técnicas de clustering**. 77 p. Dissertação (Profissional em Engenharia de Sistemas) — UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS, Lima, fev. 2010. Citado 2 vezes nas páginas 20 e 23.

NETO, A. C. A. **G2P-DBSCAN: Estratégia de Particionamento de Dados e de Processamento Distribuído fazer DBSCAN com MapReduce**. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal do Ceará, Ceará, abr. 2015. Citado na página 25.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, n. 1, p. 2825–2830, jan. 2011. Citado 2 vezes nas páginas 35 e 36.

PUYATI, W.; WALAIRACHT, A. Efficiency improvement for unconstrained face recognition by weightening probability values of modular pca and wavelet pca. **International Conference on Advanced Communication Technology**, v. 2, n. 10, p. 1449–1453, Feb 2008. ISSN 1738-9445. Citado na página 36.

RICARDO, L. Técnicas de Agrupamento. **Revista de Sistemas de Informação da Faculdade Salesiana Maria Auxiliadora**, v. 2009, n. 4, p. 18–36, dez. 2009. Citado 2 vezes nas páginas 25 e 26.

ROSSUM, G. V.; JR, F. L. D. **Python tutorial**. [S.l.]: Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995. Citado na página 35.

SHAKYA, V.; MAKWANA, R. R. S. Feature selection based intrusion detection system using the combination of dbscan, k-mean++ and smo algorithms. **International Conference on Trends in Electronics and Informatics (ICEI)**, v. 2017, n. 1, p. 928–932, May 2017. Citado na página 21.

TENDOLKAR, G. **Machine Learning Notebook**. Estados Unidos: [s.n.], 201–. Disponível em: <https://sites.google.com/site/machinelearningnotebook2/clustering/dbscan>. Acesso em: 2018-20-09. Citado na página 31.

VADOODPARAST, M.; RAZAK, H. A.; S., H. M. Fraudulent electronic transaction detection using kda model. **ArXiv**, v. 13, n. 2, p. 1–10, February 2015. Citado na página 17.

VAISHALI. Fraud detection in credit card by clustering approach. **International Journal of Computer Applications**, v. 98, n. 3, p. 29–32, July 2014. Citado na página 17.

WALT, S. V. D.; COLBERT, S. C.; VAROQUAUX, G. The numpy array: A structure for efficient numerical computation. **Computing in Science Engineering**, v. 13, n. 2, p. 22–30, March 2011. ISSN 1521-9615. Citado na página 35.

WES, M. Data structures for statistical computing in python. **Proceedings of the 9th Python in Science Conference**, v. 2010, n. 9, p. 51 – 56, jul. 2010. Citado na página 35.

ZAMINI, M.; MONTAZER, G. Credit card fraud detection using autoencoder based clustering. **International Symposium on Telecommunications (IST)**, v. 2018, n. 9, p. 486–491, Dec 2018. Citado 2 vezes nas páginas 19 e 23.