INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Previsão automática de fraude em transações financeiras

Appio Indiano do Brazil Americano Neto

Mestrado em Gestão de Sistemas de Informação

Orientador:
Doutor Fernando Manuel Marques Batista, Professor Associado,
Iscte - Instituto Universitário de Lisboa

Co-orientador:
Doutor Sérgio Miguel Carneiro Moro, Professor Associado (com Agregação),
Iscte - Instituto Universitário de Lisboa

Novembro, 2021

TECNOLOGIAS
E ARQUITETURA

Departamento de Ciências e Tecnologias da Informação

Previsão automática de fraude em transações financeiras

Appio Indiano do Brazil Americano Neto

Mestrado em Gestão de Sistemas de Informação

Orientador:
Doutor Fernando Manuel Marques Batista, Professor Associado,
Iscte - Instituto Universitário de Lisboa

Co-orientador:
Doutor Sérgio Miguel Carneiro Moro, Professor Associado (com Agregação),
Iscte - Instituto Universitário de Lisboa

Novembro, 2021

Agradecimentos

A Deus pela dádiva da vida.

Agradeço à família que é um dos motivos pelos quais ingressei em mais uma etapa da jornada acadêmica, e que dá um colorido especial à minha vida.

Aos meus orientadores, Professor Fernando Batista e Professor Sérgio Moro, por toda a direção e conhecimentos transmitidos. Obrigado por terem me mostrado novas possibilidades e visões diferentes que ampliaram os horizontes. Extendo a gratidão a todos os demais professores que partilharam de seu tempo e conhecimento durante este mestrado.

Agradeço a todos os colegas e amigos de curso, que se tornaram companheiros durante a caminhada, pois compartilhamos dúvidas e ansiedades, e juntos colecionamos conhecimentos e muitas alegrias.

Deixo aqui uma homenagem ao grande Mestre Paulo Freire, patrono da educação brasileira, que me provocou e expandiu a visão através de sua obra, tão necessária nesses tempos duros em que vivem os educadores e educandos da minha querida pátria.

Resumo

A deteção de fraude em pagamentos de transações *online* é um desafio cada vez maior, principalmente com o aumento observado nos anos recentes para o consumo de produtos e serviços em *e-commerce*. Esta dissertação descreve o processo de modelação com técnicas de *Machine Learning* aplicadas a um problema de deteção de fraude, tendo como referência o desempenho das equipas participantes de uma competição promovida pela plataforma Kaggle. A atenção dirigiu-se mais especificamente às técnicas de *sampling* de dados para tratar o problema do desbalanceamento de classes, às técnicas de preparação dos dados para deteção de anomalias e mineração de conhecimento, e por fim, aos métodos de Ensemble Learning. A principal contribuição deste trabalho, face aos outros trabalhos que utilizaram o mesmo conjunto de dados, é demonstrar a importância do processo de criação em massa de *features* informativas para o desempenho do modelo. Sendo a principal técnica do processo a criação de forma iterativa de novas *features* através da comparação de um conjunto de variáveis de cada transação com diversas medidas estatísticas do grupo à qual cada transação pertence.

Palavras-Chave:

Desbalanceamento de Classes; Deteção de Anomalias; Deteção de Fraude; Machine Learning

Abstract

The detection of fraud in online transaction payments is an increasing challenge, especially with the increase observed in recent years for the consumption of products and services in e-commerce. This dissertation describes the modeling process with Machine Learning techniques applied to a fraud detection problem, having as reference the performance of teams participating in a competition promoted by the Kaggle platform. More specifically, attention was directed to data sampling techniques to deal with the problem of class Imbalance, to data preparation techniques to detect anomalies and knowledge mining, and finally, the Ensemble Learning methods. The main contribution of this work, compared to other works that used the same dataset, is to demonstrate the importance of the mass creation process of informative features for the model's performance. The main technique of the process is the iterative creation of new features through the comparison of a set of variables of each transaction with several statistical measures of the group to which each transaction belongs.

Keywords:

Anomaly Detection; Class Imbalance; Fraud Detection; Machine Learning

Índice Geral

Capítulo 1 - Introdução **1**
1.1 Enquadramento do tema 1
1.2 Motivação e relevância do tema 1
1.3 Questões e objetivos de investigação 2
1.4 Abordagem metodológica 3
1.5 Estrutura e organização da dissertação 3
Capítulo 2 - Fundamentos e Revisão da Literatura **5**
2.1 Conceitos fundamentais sobre fraude 5
2.1.1 Definição do conceito de fraude e seus impactos 5
2.1.2 A fraude de cartão de crédito 7
2.1.3 Prevenção e deteção de Fraude 9
2.2 Machine Learning aplicado para deteção de fraudes 10
2.2.1 O problema do desbalanceamento de classes 10
2.2.2 Deteção de anomalias 11
2.3 Técnicas de pré-processamento para deteção de fraude 13
2.4 Abordagens já aplicadas ao dataset IEEE CIS Fraud Detection 15
2.5 Ferramentas de análise de dados 17
2.6 Descrição dos algoritmos utilizados 18
Capítulo 3 - Dados e Metodologia **20**
3.1 Metodologia CRISP-DM 20
3.2 Compreensão do negócio 21
3.2.1 Descrição da competição 21
3.2.2 Descrição dos datasets 21
3.2.3 Métrica de avaliação de desempenho 22
3.3 Compreensão dos dados 23
3.3.1 Coleta e descrição dos dados 23
3.3.2 Exploração dos dados 25
3.3.2.1 Sumário estatístico 25
3.3.2.2 Análise visual 28
3.3.2.3 Análise das variáveis Card1 e IsFraud 32
3.3.2.4 Análise de correlação 32
3.4 Preparação dos dados 33
3.4.1 Seleção e limpeza dos dados 34

3.4.2 Formatação dos dados 36
3.4.3 Criação de novas features 36
3.4.4 Seleção de features 44
3.4.4.1 Verificação de features com NZV (Near Zero Variance) 44
3.4.4.2 Features com alta correlação 44
3.4.4.3 Avaliação individual das novas features 44
3.4.4.4 Adversarial validation 45
Capítulo 4 - Processo de Modelação 46
4.1 Escolha dos algoritmos 46
4.2 Separação dos dados para treino e validação 47
4.3 Construção dos modelos 47
4.3.1 Algoritmo Xgboost 48
4.3.2 Algoritmo Random Forest 48
4.3.3 Algoritmo GBM 49
4.3.4 Algoritmo SVM 49
4.3.5 Algoritmo MLP 50
4.4 Combinação dos modelos 50
4.4.1 Combinação pela média das previsões 51
4.4.2 Combinação através de um meta modelo 51
4.4.3 Combinação por voto maioritário das previsões 51
Capítulo 5 - Resultados e Discussão 52
5.1 Contribuição das técnicas de *sampling* 52
5.2 Contribuição das técnicas de deteção de anomalias 53
5.3 Desempenho individual dos modelos 55
5.4 Desempenho dos modelos combinados (Ensemble learning) 56
5.5 Resultado das submissões ao Kaggle 56
Capítulo 6 - Conclusões e Recomendações 59
Referências Bibliográficas 61
Anexos e Apêndices 64
Apêndice A 64
Apêndice B 66
Apêndice C 69
Apêndice D 71
Apêndice E 78

Apêndice F..... 81

Índice de Tabelas

Tabela 1 - Tipos de fraude de cartão de crédito 8
Tabela 2 - Recursos utilizados em R 17
Tabela 3 - Descrição dos datasets 22
Tabela 4 - Descrição das variáveis dos datasets transact 24
Tabela 5 - Descrição das variáveis dos datasets identity 25
Tabela 6 - Sumário estatístico para as variáveis do tipo booleano 26
Tabela 7 - Sumário estatístico para as variáveis com dados categóricos 26
Tabela 8 - Sumário estatístico para as variáveis numéricas (exceto as colunas V) 27
Tabela 9 - Amostra de dados do card1 de número 15775 38
Tabela 10 - PCA com 16 componentes 43
Tabela 11 - Parâmetros do modelo Xgboost 48
Tabela 12 - Parâmetros do modelo Random Forest 49
Tabela 13 - Parâmetros do modelo GBM 49
Tabela 14 - Parâmetros do modelo SVM 50
Tabela 15 - Parâmetros do modelo MLP 50
Tabela 16 - Resultado das técnicas de sampling 53
Tabela 17 - Distribuição das features de acordo com a característica 54
Tabela 18 - AUC dos modelos sobre os dados de validação 55
Tabela 19 - AUC dos modelos combinados 56
Tabela 20 - AUC das previsões submetidas ao Kaggle 57
Tabela 21 - Valores omissos da variável V1 até V172 do dataset de transações 78
Tabela 22 - Valores omissos da variável V173 até V339 do dataset de transações 79
Tabela 23 - Valores omissos do identity dataset 80

Índice de Figuras

Figura 1 - Triângulo da Fraude 6
Figura 2 - Técnicas de balanceamento de classes 14
Figura 3 - Exemplo de uma MLP 19
Figura 4 - Fluxo do CRISP-DM Fonte: Chapman et al (2000) 20
Figura 5 - Gráfico com curva ROC 23
Figura 6 - *Boxplot* das variáveis *card2, card3 e card5* 28
Figura 7 - Histogramas dos dados de treino e teste das variáveis categóricas 29
Figura 8 - Distribuição da fraude nas variáveis categóricas 31
Figura 9 - Matriz de correlação das 55 primeiras variáveis dos dados transacionais 33
Figura 10 – Distribuição da fraude nos *clusters* 43
Figura 11 - Gráfico da matriz de importância (*Top 21 features*) 54
Figura 12 - Resultados no site do Kaggle 57
Figura 13 - Resultados dos 10 primeiros colocados na competição do Kaggle 58
Figura 14 - Taxonomia das áreas de Fraude mais comuns, Abdallah (2016) 65
Figura 15 - Tipos de aprendizagem 67
Figura 16 - Etapas da modelação de um problema de classificação 68
Figura 17 - Gráfico de correlação das colunas V1 até V51 71
Figura 18 - Gráfico de correlação das colunas V52 até V102 72
Figura 19 - Gráfico de correlação das colunas V103 até V153 73
Figura 20 - Gráfico de correlação das colunas V154 até V204 74
Figura 21 - Gráfico de correlação das colunas V205 até V255 75
Figura 22 - Gráfico de correlação das colunas V256 até V306 76
Figura 23 - Gráfico de correlação das colunas V307 até V339 77

Glossário de Abreviaturas e Siglas

ACFE - Association of Certified Fraud Examiners

ATM - Automated Teller Machine

AUC - Area Under the ROC Curve

CEO - Chief Executive Officer

CNP - Card not present

CRISP-DM - Cross-Industry Standard Process – Data Mining

DM - Data Mining

FN - False Negative

FP - False Positive

GA - Genetic Algorithm

GBM - Gradient Boosting Model

IEEE - Institute of Electrical and Electronics Engineers

KMO - Kaiser-Meyer-Olkin factor adequacy

KNN - K Nearest Neighboors

ML - Machine Learning

MLP - Multi-Layer Perceptron

PCA - Principal Component Analysis

RNA – Rede Neuronal Artifical

ROC - Receiver Operating Characteristic

SLA - Service Level Agreement

SMOTE - Synthetic Minority Oversampling Technique

SVM - Support Vector Machine

TP - True Positive

TN - True Negative

Xgboost - Extreme Gradient Boosting

Capítulo 1 - Introdução

Neste capítulo é feita a introdução da dissertação, em que se apresenta o enquadramento do tema, relacionando o problema da utilização de cartões fraudados para o pagamento de transações *on line*, com a possível solução para deteção da fraude através do recurso a Machine Learning (ML). A secção 1.1 faz o enquadramento do tema. A secção 1.2 trata da motivação e relevância do tema. A secção 1.3 apresenta as questões e objetivos de investigação. A secção 1.4 descreve a abordagem metodológica e a secção 1.5 apresenta a estrutura e organização da dissertação.

1.1 Enquadramento do tema

O combate à fraude é um trabalho que explora de forma crescente o fenômeno da digitalização do mercado e o aumento da quantidade de dados produzidos diariamente. Os fraudadores atacam as vulnerabilidades existentes em diversas tecnologias e operam de forma a tentar não deixar visíveis os rastros de suas ações. Por sua vez, os responsáveis pela deteção das fraudes são levados a utilizar os recursos tecnológicos mais modernos, como por exemplo as imensas quantidades e variedades de dados geridos por sistemas de *Big Data*, como também as capacidades das ferramentas e técnicas de *Data Mining* (DM), para detetar as fraudes de forma precisa e veloz.

Uma área que cresce anualmente é o comércio *on line*, que tem nos cartões de crédito e débito os principais meios utilizados para a realização de pagamentos. A fraude envolvendo o uso de cartões para pagamento de transações financeiras *on line* cresce junto com esse mercado. Para os fraudadores, essa é uma modalidade de fraude com risco reduzido, pois não requer a exposição presencial em uma loja física e possibilita a atuação de forma anônima. Este trabalho será focado em investigar o estado da arte de técnicas de ML para a deteção de fraudes que utilizam cartão para pagamentos de transações *on line*.

1.2 Motivação e relevância do tema

O ML é uma área de conhecimento efervescente, com muita investigação em curso no meio acadêmico e também é alvo de grandes investimentos por parte das empresas, pois

tem imenso potencial para resolver uma ampla gama de problemas, tirando proveito dos dados produzidos e armazenados pelas empresas e outras organizações.

Embora as empresas tenham desenvolvido muita capacidade em produzir e armazenar dados, ainda existe uma lacuna na habilidade de explorar os dados de forma a gerar conhecimento que seja aplicável em benefício das organizações e da sociedade de forma mais ampla, e muitas empresas subutilizam os dados que produzem.

O tema é muito relevante, pois o combate à fraude é uma demanda em que as empresas e organizações devem se empenhar não apenas por causa das imensas perdas financeiras envolvidas, mas também pela fundamental preocupação em cultivar uma sociedade alicerçada em sólidos valores éticos e no bem-estar comum. Não é raro que a prática de um determinado tipo de fraude esteja associada a outros tipos de crime, como por exemplo, o uso de cartões em fraudes no pagamento de compras está frequentemente associado a crimes de furto ou roubo, violação e danos ao patrimônio de empresas, falsificação de *sites* e *phishing*.

Além das perdas financeiras para as empresas, a fraude também causa danos à imagem perante os clientes e perda de confiança por parte do mercado, pois empresas vítimas de fraudes expressivas transmitem uma imagem de que não estão a gerir bem os negócios e nem as informações sigilosas de seus clientes e parceiros comerciais. Do ponto de vista do custo operacional para as empresas, a fraude encarece a prestação de serviços e a comercialização de bens, pois é um custo adicional que está embutido no preço de produtos e serviços e onera os custos para o consumidor final.

1.3 Questões e objetivos de investigação

A previsão de fraude através de técnicas de ML apresenta características e desafios de modelação muito específicos. Espera-se que este trabalho contribua para avançar o conhecimento sobre as técnicas mais eficazes para pré-processamento dos dados para o problema de classificação de fraude. Outra contribuição esperada é testar e descrever diferentes técnicas de *Ensemble learning* aplicadas em deteção de fraude. Assim sendo, apresentamos a seguinte questão de investigação:

- Quais estratégias podem ser usadas para a produção de features informativas com capacidade para enriquecer um modelo de ML para previsão de fraude, a partir do conjunto de dados escolhido para o estudo?

Os principais objetivos são:

- Investigar se a utilização de técnicas de pré-processamento para promover o balanceamento de classes contribuem para o aumento do desempenho do modelo.
- Verificar se técnicas de deteção de anomalias, tais como *Benford Law, Time Window e Z-Score* contribuem para aumentar a eficácia do modelo.
- Analisar diferentes métodos de combinação de modelos *(Ensemble Learning)* para identificar qual método mais otimiza a capacidade preditiva dos modelos.

1.4 Abordagem metodológica

Inicialmente será feito um levantamento do estado da arte e revisão de literatura para identificar os processos aplicados com maior excelência no processo de deteção de fraude, através do ML e quais os trabalhos publicados mais relevantes que utilizaram o mesmo *dataset* desta dissertação.

A metodologia definida para o desenvolvimento da modelação é o CRISP-DM *(Cross-Industry Standard Process – Data Mining),* que é uma *framework* amplamente utilizada para processos de DM e ML (Chapman et al., 2000).

1.5 Estrutura e organização da dissertação

Este capítulo resume o problema da fraude e descreve como a evolução tecnológica propicia oportunidades para os fraudadores e ao mesmo tempo também provê recursos para o combate dessas práticas ilegais. A questão de investigação, os objetivos secundários e a metodologia são apresentados na sequência.

O restante deste documento está organizado da seguinte forma. O capítulo 2 foca na revisão de literatura, aprofunda a discussão sobre o pagamento com cartão fraudado, apresenta o problema do desbalanceamento de classes e deteção de anomalias, resume as

abordagens aplicadas em outros trabalhos feitos com este mesmo *dataset*, apresenta as ferramentas utilizadas para análise e processamento dos dados e conclui com um breve resumo sobre os algoritmos utilizados neste trabalho.

O capítulo 3 é focado na descrição do *dataset* e nas características dos dados, e tem como ponto de partida a compreensão do problema de negócio, passa pela descrição da metodologia utilizada para a compreensão dos dados e detalha as etapas executadas na preparação dos dados.

O capítulo 4 descreve o processo de modelação desde a escolha dos algoritmos, separação dos dados para treino e validação, detalha os hiperperâmetros utilizados para a afinação dos modelos e descreve as técnicas de Ensemble Learning aplicadas.

O capítulo 5 apresenta os resultados alcançados e discute questões importantes da fase de pré-processamento e modelação dos dados. O capítulo 6 reúne as conclusões do estudo e recomendações finais.

Capítulo 2 - Fundamentos e Revisão da Literatura

Este capítulo desenvolve a conceituação do que é o fenômeno da fraude e faz a revisão da literatura contendo as abordagens já aplicadas em outros trabalhos que utilizaram o mesmo conjunto de dados. A secção 2.1 apresenta os impactos econômicos derivados da fraude, quais os tipos mais comuns de fraude e como são feitos os processos de prevenção e deteção de fraude. A secção 2.2 descreve os problemas de desbalanceamento de classes e deteção de anomalias. A secção 2.3 aborda técnicas de pré-processamento para deteção de fraude. A secção 2.4 descreve as abordagens aplicadas ao mesmo conjunto de dados em outros estudos. A secção 2.5 apresenta as ferramentas utilizadas para a análise de dados e a secção 2.6 descreve os algoritmos utilizados.

2.1 Conceitos fundamentais sobre fraude

Esta secção apresenta a definição sobre a natureza da fraude, o que motiva a sua realização, quais são os impactos que causa, quais os tipos mais frequentes e como é feita a prevenção e deteção.

2.1.1 Definição do conceito de fraude e seus impactos

A fraude pode ser definida de forma breve, como um ato de engano ou desonestidade intencional perpetrado por um ou mais indivíduos, geralmente para ganho financeiro. Com o objetivo de obter ganho desonesto, fraudadores estudam vulnerabilidades que podem ser exploradas em sistemas, processos ou comportamentos das vítimas, que são pessoas ou empresas. Um erro ou falha não configura uma fraude se não houver a intenção de prejudicar a vítima. No caso específico das empresas, o fraudador pode ser um agente externo ou um funcionário que comete a chamada fraude ocupacional. No caso de fraude contra o empregador, o empregado normalmente abusa da confiança em si depositada para a execução do trabalho, e ao agir deliberadamente para causar dano ao empregador, comete fraude (Gee, 2015).

Um importante fator para entender a fraude é compreender as motivações do fraudador, sendo que essa compreensão contribui para o processo de prevenção da fraude.

O Dr. Donald Cressey, professor e pesquisador do fenômeno da fraude, desenvolveu a Teoria do Triângulo da Fraude, que é ilustrada na figura 1.

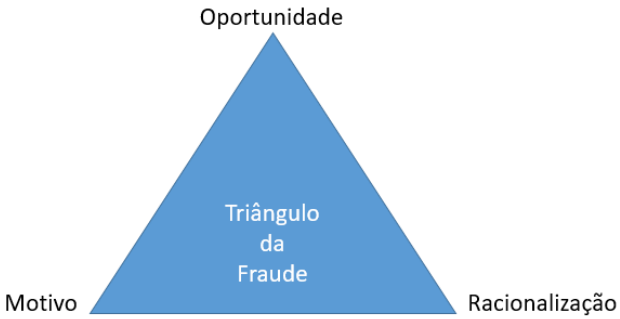


Figura 1 - Triângulo da Fraude

O Triângulo da Fraude é a síntese da conclusão proposta por Cressey para explicar por que uma pessoa comete fraude. O triângulo é formado por três elementos críticos que são Motivo, Oportunidade e Racionalização. A ausência de um desses elementos poderia evitar a realização da fraude. O Motivo é explicado como sendo o desejo ou emoção por trás da fraude e alguns exemplos são: necessidades financeiras, dívidas, vício em drogas ou jogos dentre outros. Em segundo lugar, a Oportunidade pode ser compreendida como a circunstância que possibilita a efetivação da fraude e alguns exemplos são: posição ou função de autoridade na empresa, acesso a bens e documentos e controles internos fracos. Por fim, a Racionalização é a forma como o fraudador justifica seus atos indevidos para si próprio ou para outrem e concretiza-se por meio de afirmações tais como: "Não é muito dinheiro, a companhia é rica e não sentirá falta", "Todos fazem isso", "É para um bom propósito" ou "A empresa não me remunera adequadamente" conforme Biegelman, M., Bartow, J. (2006).

De acordo com o relatório *Report to the nations* (ACFE, 2020), um estudo realizado pela *Association of Certified Fraud Examiners* no ano de 2020, com informações de 125 países, 2.504 casos de fraude ocupacional analisados entre janeiro de 2018 e setembro de 2019, em 23 categorias de indústrias, aponta perdas anuais de 3.6 bilhões de dólares, sendo que 85% dos fraudadores apresentaram ao menos um comportamento altamente suspeito. Além disso, a monitorização por meio de dados foi responsável pela redução de 33% nas perdas menores e na duração das fraudes. A ACFE estima que as organizações perdem cerca de 5% de suas receitas a cada ano devido a fraudes ocupacionais.

Uma publicação que cobre notícias e análises da indústria global de pagamentos chamada *Nilson Report*, relatou que as perdas globais por fraude de cartão de crédito totalizaram 22,8 bilhões de dólares em 2016, o que é um aumento de 4,4% em relação a 2015. Segundo o Comitê Europeu de Seguros, a fraude é responsável por perdas que vão de 5 a 10% dos valores de sinistros pagos para seguro não vida, e o FBI calcula que o custo total da fraude em seguro não saúde nos EUA é de mais de 40 bilhões de dólares por ano (Baesens et al., 2020).

2.1.2 A fraude de cartão de crédito

Abdallah (2016) refere que o cartão de crédito é um pequeno cartão de plástico que substitui o uso do dinheiro em papel nas transações de compra e venda de bens e serviços. A utilização do cartão de crédito em transações de comércio eletrônico cresce a cada ano, e com isso a fraude de cartão de crédito tem se tornado mais comum e é classificada primariamente em duas categorias que são:

1. Fraude de cartão de crédito offline: caracterizada pelo uso de cartão de crédito perdido ou roubado para a realização de pagamentos de forma presencial. Nessa situação o fraudador precisa agir rápido antes que o operador do cartão faça o bloqueio por perda ou roubo.
2. Fraude de cartão de crédito online: os dados dos cartões são roubados por fraudadores e são utilizados posteriormente em transações financeiras por telefone, internet ou fax.

A fraude de cartão de crédito online também é chamada de CNP (C*ard Not Present*) pois nem o cartão e nem o dono do cartão estão presentes fisicamente no momento da transação. Dessa forma o vendedor não consegue validar fisicamente o cartão e nem a identidade do portador do cartão (Van Vlasselaer et al., 2015)

Para a realização da fraude CNP, o fraudador precisa obter os dados de cartões de crédito válidos e ativos. A obtenção dos dados dos cartões pode ser feita através de várias técnicas, das quais destacam-se:

1. *Skimming*: roubo dos dados dos cartões por meio de um hardware instalado em equipamentos que fazem leitura de cartões como ATM e máquinas de venda. Os dados dos cartões de crédito são copiados para um dispositivo físico e posteriormente utilizados para falsificação de cartão ou fraude CNP.

2. Clonagem de site: roubo dos dados do cartão quando uma vítima acede a um site clonado, acreditando estar no site verdadeiro e insere os dados do seu cartão para realizar transações financeiras.
3. Geradores de cartões de crédito: os geradores de cartão de crédito são softwares que geram combinações de numeração e data de expiração válidas de cartões de crédito, que são utilizados frequentemente no processo de teste de serviços de pagamentos online, mas podem ser usadas de forma ilegal para fraudes.
4. *Phishing*: essa técnica é utilizada para roubar os dados de identidade das vítimas através de *Spam* por *e-mail* ou *pop-up windows* que simulam um pedido de informações feito por uma entidade que a vítima conhece e confia, como o seu banco ou um site de compras. Pede-se então que a vítima preencha algum tipo de formulário com seus dados pessoais e do cartão e essas informações são armazenados e usadas por criminosos para a prática de fraudes ou outros crimes (Patidar & Sharma, 2011).

Van Vlasselaer (2015) apresenta uma lista com as categorias de fraude de cartão de crédito mais comuns que pode ser verificada na tabela 1.

Tabela 1 - Tipos de fraude de cartão de crédito

Tipo de Fraude	**Descrição**
Application Fraud	Aquisição de cartão de crédito com a utilização de informações falsas
Cartão roubado ou perdido	Utilização de cartão que foi perdido ou roubado
Cartão falsificado	Utilização da cópia de um cartão verdadeiro
CNP Fraud (Card Not Present)	Utilização sem consentimento das informações do cartão de outra pessoa para compras não presenciais

Esta dissertação tem por objetivo trabalhar com um conjunto de dados que reúne transações de compras em comércio eletrônico, e parte das transações foram classificadas como fraude devido ao uso de cartão fraudado para a realização do pagamento. Portanto, esta categoria de fraude será o foco desta tese. O Apêndice A apresenta os tipos e categorias mais comuns de fraude.

2.1.3 Prevenção e deteção de Fraude

Todo tipo de negócio opera sob uma margem de risco que pode envolver perda financeira, perda de reputação e aumento de custos dentre outros. É uma boa prática realizar uma análise dos riscos do negócio para possibilitar a tomada de decisão sobre o que é mais barato, as perdas provenientes dos riscos, ou o custo das medidas necessárias para a mitigação destes riscos. Um operador de seguros pode por exemplo, avaliar se o investimento em um sofisticado sistema de prevenção de fraudes é inferior às perdas financeiras por fraude. Caso as perdas por fraude sejam menores que o investimento no sistema, ou a implementação do sistema também implique em um processo que cause atrito no relacionamento com os clientes, a empresa pode implementar controlos para monitorizar transações suspeitas e tratar a fraude de forma reativa. Ou seja, realizar um processo de deteção após a fraude ter sido consumada, de forma que seja detetado o máximo possível das fraudes realizadas e no mais curto espaço de tempo, para reduzir as perdas (Gee, 2015).

Os processos de prevenção e deteção de fraudes são complexos e exigem cuidados por parte das empresas, pois a demora ou a não identificação das fraudes aumenta as perdas. Um processo de prevenção que apresenta muitos falsos positivos impacta em possível perda de receita, como por exemplo quando concessões de crédito para clientes legítimos são negadas por suspeita de fraude. Os processos de prevenção e deteção de fraudes devem funcionar de forma complementar, de modo que a fraude que escapar ao processo de prevenção seja capturada pela fase de deteção, visto que os fraudadores são especialistas em contornar as medidas antifraude implementadas pelas organizações. A prevenção à fraude de cartão de crédito tem sido intensificada fisicamente pelo uso de números de identificação, chip, senha e autenticação de dois fatores para a realização de transações. Por outro lado, a realização de análise estatística e modelos de ML são utilizados cada vez mais (Bolton & Hand, 2002).

As técnicas de DM, *data analysis* e ML tornam-se fundamentais no combate à fraude à medida que aumenta anualmente o volume de transações digitais armazenadas em sistemas de informação. Essas técnicas, apoiadas no uso de estatística possibilitam automatizar o processo de deteção de fraude, pois seria impossível analisar manualmente todas as transações realizadas, que em determinados cenários precisam que a deteção da fraude ocorra em tempo real, ou seja, antes que a transação fraudulenta seja concluída.

As empresas recorrem a essas técnicas, aliadas a sofisticados sistemas que permitem configurar regras que são aplicadas durante o processamento dos dados, e gerar alertas que desencadeiam um processo de análise manual, feito por equipas especializadas na identificação de indícios de fraude por meio dos dados (Zhang et al., 2019).

2.2 Machine Learning aplicado para deteção de fraudes

Esta secção aborda dois problemas típicos na deteção de fraude por meio de *Machine Learning (ML)*. O primeiro problema é o desafio de fazer a classificação de dados com classes altamente desbalanceadas e o segundo problema é o processo de deteção de anomalias. O Apêndice B apresenta uma introdução ao processo de ML.

2.2.1 O problema do desbalanceamento de classes

O desbalanceamento de classes é o fenômeno em que uma variável apresenta uma distribuição desequilibrada nas suas instâncias, de modo que uma classe minoritária ocorre numa frequência muito inferior em relação às outras classes. Realizar o treinamento de um modelo de classificação sobre dados com classes desbalanceadas é um desafio, e *datasets* desbalanceados são comuns em diferentes segmentos como diagnósticos médicos, deteção de fraudes e análises de crédito (Kabra et al., 2020).

No contexto da fraude, a classe minoritária é a classe mais importante e o principal objetivo é classificá-la corretamente (Awad & Khanna, 2015). Em muitos *datasets* a fraude está presente em menos de 0,5% das observações. Esse tipo de desbalanceamento elevado de classes leva a previsões que favorecem a classe maioritária e o resultado é que a fraude não é detetada (Baesens et al., 2020).

Bolton & Hand (2002) refere que o desbalanceamento na ordem de cem para um é frequente em deteção de fraude, e em outras áreas essa grandeza pode atingir a proporção de 100.000 para 1. Métodos de aprendizagem supervisionada que utilizam amostras classificadas com as classes *Fraude* e *Não Fraude*, apresentam dificuldades para classificar os casos de *Fraude*, e isso é devido ao desbalanceamento das classes, pois a classe *Não Fraude* ocorre com uma frequência muito superior.

De acordo com He & Garcia (2009), a maioria dos algoritmos de aprendizagem apresenta uma queda de performance diante de dados desbalanceados e esse fenômeno é conhecido como *Imbalanced Learning Problem*. Isso acontece porque a maioria dos algoritmos assume que os dados apresentam a distribuição de classes de forma balanceada, portanto, também assume que o custo de classificações incorretas é igual para todas as classes.

O desbalanceamento de classes também apresenta desafios para a avaliação da acurácia de modelos preditivos. O problema principal com o erro em classificar a classe *Fraude* como sendo *Não Fraude* (falso negativo) é que o custo desse tipo de erro é muito maior do que o erro reverso (falso positivo). A acurácia é uma métrica comumente utilizada para avaliar o desempenho de algoritmos de ML, porém, quando há desbalanceamento de classes e os custos dos diferentes erros é muito significativo, a técnica ROC (*Receiver Operating Characteristic*) tem se mostrado uma melhor alternativa para avaliar a performance de classificação sobre uma variada gama de taxas de erro entre falsos positivos e verdadeiros positivos (Chawla et al., 2002). A secção 3.2.3 descreve com mais detalhes a curva ROC.

2.2.2 Deteção de anomalias

Uma anomalia, também chamada de *outlier* ou desvio, é um padrão de comportamento nos dados que não está em conformidade com o comportamento identificado na maior parte do *dataset*. A deteção de anomalias é o processo de identificar anomalias nos dados. A deteção de fraude, que é um subdomínio da deteção de anomalias, é uma área de pesquisa onde o uso de ML pode ter um impacto significativo para empresas que sofrem com fraudes e muitas pesquisas têm sido realizadas nesse campo (Ayman et al., 2020).

A fraude normalmente é um evento raro, um desvio do comportamento padrão. Como referido por Dang (2017), o propósito da deteção de *outlier* é identificar os itens que não estão em conformidade com outros itens em um conjunto de dados. Em algumas áreas de conhecimento, tais como diagnóstico de doenças e transações financeiras com cartão de crédito, os *outliers* são o grupo que se deseja analisar e identificar, pois o habitual em um conjunto de dados de diagnóstico de doenças é que a maioria das observações seja de indivíduos saudáveis, e em transações com cartão de crédito, que a maior parte tenha sido

efetuada de forma legítima. Nesses exemplos, embora a fraude no cartão de crédito e o diagnóstico de uma doença sejam eventos raros, são eventos importantíssimos, portanto, são os motivadores da análise dos dados.

No processo de análise de dados, uma anomalia não é necessariamente uma fraude. Há anomalias que ocorrem nos dados de forma periódica e são fruto de falhas em sistemas, entradas de dados sem processo de validação ou com validação ineficiente. Do ponto de vista analítico, a anomalia é alguma coisa que foge à norma, coisas que se comportam fora do padrão normal ou apresentam características não usuais. Exemplos de anomalias são: *outliers*, *p*resença de *inliers* onde não são esperados, excesso ou escassez de transações, itens inexplicáveis, relacionamento incomum entre elementos, tempos inesperados de transações ou eventos, inconsistências e a ausência ou duplicidade de números de itens.

As anomalias podem ocorrer quando dados de sistemas ou módulos diferentes são integrados sem que esse processo tenha sido projetado eficazmente, ou que o processo seja feito de forma manual. O conhecimento dos sistemas e do negócio permite separar quais são as anomalias esperadas ou previstas, das anomalias que caracterizam potencialmente a existência de fraude (Gee, 2015).

Os *outliers* podem ser divididos em duas categorias, que são: observações válidas e observações inválidas. Para as observações válidas, pode-se utilizar como exemplo o salário de um CEO de uma empresa no valor de 500 mil euros, que comparado aos salários pagos aos outros funcionários é um *outlier*. Um exemplo para observações inválidas seria uma instância com a idade do indivíduo igual a 200 anos. Esses exemplos são *outliers* univariados, pois são valores extremos em relação a apenas uma dimensão dos dados. Os *outliers* podem também ser classificados como observações inválidas de forma multivariada, portanto, em duas ou mais dimensões dos dados. Neste caso, toma-se o exemplo em que para uma observação distinta dos dados, a variável gênero seja igual a "Masculino" e a variável gravidez seja igual "Sim", configurando, portanto, uma observação inválida. O tratamento dos *outliers* exige um grande cuidado para não causar viés no modelo. As observações inválidas, em um exemplo hipotético, podem ser tratadas da mesma maneira que valores omissos, ou seja, procede-se à substituição dos valores inválidos pela média, mediana ou moda da variável, ou se não houver perda de

informação relevante poderá ser feita a remoção das observações que contêm os valores inválidos (Baesens et al., 2015).

No âmbito dos métodos de processamento dos dados para identificar padrões de comportamento, estes métodos fornecem conhecimentos ao modelo para fazer a separação entre as transações feitas por clientes legítimos e fraudadores. É possível, dentro de um conjunto de transações com o pagamento feito pelo mesmo cartão, separar subgrupos de transações que formam diferentes padrões comportamentais ao longo do tempo. Portanto, há formas de identificar subgrupos diferentes de transações por meio da média de gasto por transação, aumento na frequência das transações ao longo do dia ou semana, mudança no local de origem da transação, alteração nos dados cadastrais como *e-mail* ou endereço. A mudança abrupta de comportamento nas transações pagas por um mesmo cartão é uma suspeita de fraude. Devido às mudanças do perfil do próprio cliente, o agrupamento dos dados para a análise de comportamento deve ser feita dentro de um período não muito longo de tempo, que pode variar de 24, 60 ou até 168 horas (Correa Bahnsen et al., 2016).

Mudanças de comportamento do cliente e do fraudador são provocadas por variados motivos. O cliente muda o padrão de gastos por aumento de salário, desemprego, casamento, nascimento de filhos e período do ano dentre outros. O fraudador precisa mudar o *modus operandi* para escapar das técnicas e processos de deteção de fraude. Com isso, conforme abordado por Fawcett (1997), o processo de deteção de fraude deve ser de natureza adaptativa para acompanhar as mudanças de comportamento.

2.3 Técnicas de pré-processamento para deteção de fraude

O pré-processamento é a etapa à qual os problemas identificados nos dados são tratados com o objetivo de maximizar a *performance* da aprendizagem do modelo. Como já referido, o desbalanceamento de classes é um desafio na área da deteção de fraude e pode ser tratado ao nível dos dados, através de técnicas de pré-processamento ou ao nível dos algoritmos. A figura 2 apresenta opções de técnicas para endereçar o problema do desbalanceamento de classes (A. Abdallah, M. Maarof, 2016).

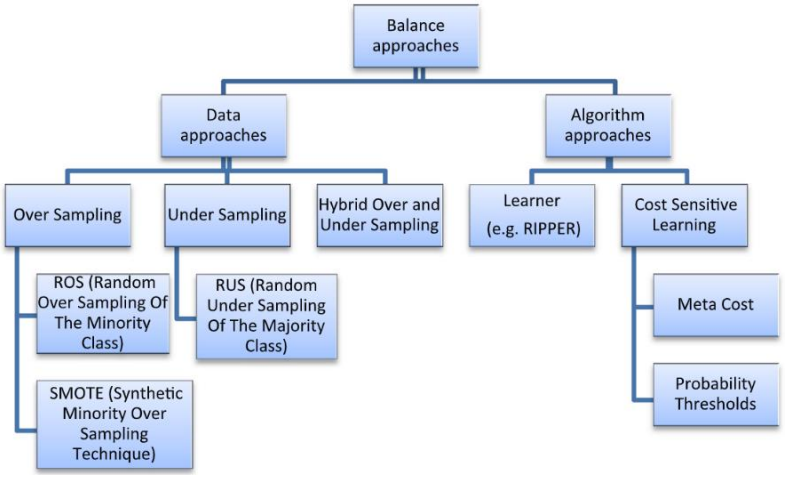


Figura 2 - Técnicas de balanceamento de classes

Especificamente ao nível dos dados, as principais técnicas utilizadas são *Over Sampling*, *Under Sampling e Hybrid Over and Under Sampling*. Os métodos *Under-Sampling* eliminam aleatoriamente instâncias da classe maioritária para balancear a distribuição de classes. Porém, a remoção dessas instâncias da classe maioritária leva à perda de informação e posteriormente degradam a performance de classificação. Os métodos *Over Sampling*, ao contrário, duplicam as instâncias da classe minoritária aleatoriamente. Há métodos mais sofisticados para o aumento de dados, como o *Synthetic Minority Oversampling Technique (SMOTE)*, que adiciona instâncias minoritárias sintéticas aos dados de treino. O *SMOTE*, porém, não utiliza os dados da classe maioritária. Há variações do *SMOTE*, tais como o *Borderline SMOTE*, *Adaptive Synthetic Sampling Technique* (*ADASYN*) e *SMOTEBoost* que geram dados sintéticos apenas da classe minoritária. Abordagens que focam apenas na classe minoritária podem degradar a performance de classificação ao sobrepor instâncias. Já os métodos focados apenas em reduzir a instância majoritária apresentam tendência a causar *overfit*, que é quando o algoritmo aprende tantos detalhes dos dados de treino que não é capaz de generalizar eficientemente sobre novos dados. O Apêndice C aprofunda a discussão sobre as técnicas de *sampling* para o pré-processamento dos dados.

2.4 Abordagens já aplicadas ao dataset IEEE CIS Fraud Detection

O *dataset IEEE-CIS fraud Detection* já foi alvo de outros estudos e a proposta desta secção é apresentar uma revisão das abordagens feitas por nove autores em termos de pré-processamento de dados, *Feature Engineering* e algoritmos utilizados.

A maior parte dos autores cita brevemente os processos utilizados para tratamento de valores omissos, *outliers* e eliminação de *features* altamente correlacionadas. Não se fará um destaque para essas etapas do processo, visto que não foram descritas com muita riqueza de detalhes. É importante ressaltar também que nos trabalhos analisados, em sua maioria, fizeram comparação de performance entre algoritmos clássicos em problemas de classificação binária, dentre os quais destacam-se: *Decision Tree, Naïve Bayes, SVM (Support Vector Machine), Logistic Regression e Random Forest*, (Huang, 2020), (Song, 2020), (Najadat et al., 2020), (Zhang et al., 2020), (Ge et al., 2020) e (Lei et al., 2020).

Na fase de pré-processamento, para efetuar o balanceamento de classes, a técnica *SMOTE* foi utilizada por Zhang et al., (2020) e Jing et al., (2019). A técnica *Bootstrapping* para tratamento do desbalanceamento de classes foi empregada por Jing et al., (2019).

Os autores Ge et al., (2020) e Zhang et al., (2020) aplicaram o processo de *Feature Selection* de forma automatizada com a técnica f*ive fold RFECV (recursive feature elimination with cross validation)* suportada por um modelo Lightgbm, enquanto Li et al., (2020) utilizou a técnica *five fold RFECV* sobre o modelo Catboost**.**

Relativamente aos algoritmos implementados nos trabalhos, em quatro estudos, o Xgboost obteve melhor resultado quando comparado a outros algoritmos, dentro de um total de nove trabalhos analisados. O Xgboost teve seu desempenho avaliado como ótimo nos estudos de Lei et al., (2020), Ge et al., (2020), Zhang et al., (2020), Jing et al., (2019) e Priscilla & Prabha, (2020).

O estudo de Priscilla & Prabha, (2020) apresenta alguns experimentos de combinação do Xgboost com as seguintes técnicas de *data sampling*: *Smote, Adasyn, Tomeklink, Nearmiss, Smotenn e SmoteTomek*. A *performance* combinada do Xgboost com cada uma dessas técnicas foi comparada com a *performance* individual do algoritmo, e com o algoritmo sob uma otimização baseada em *RandomizedSearchCV*. A melhor *performance* foi obtida com a última configuração mencionada, isto é, com o algoritmo sob otimização, que alcançou um AUC de 0,966.

O algoritmo Lightgbm utiliza uma estrutura de árvore de decisão de forma distribuída. Comparado ao Xgboost, é 6 vezes mais rápido para o treinamento e altamente eficiente, pois apresenta excelentes resultados em termos de acurácia e lida melhor com grandes volumes de dados. Este classificador obteve 0,955 na métrica AUC nos estudos de Ge et al., (2020).

Huang, (2020) utiliza em seu estudo o algoritmo Lightgbm, porém com uma otimização Bayesiana que alcançou um AUC de 0,973, enquanto o teste do Lightgbm sem a otimização teve um AUC de 0,961. Li et al., (2020) também aplicou a otimização Bayesiana, porém o algoritmo escolhido foi o Catboost, que alcançou um AUC de 0,973.

Uma proposta de utilização mista de algoritmos pode ser encontrada em Song, (2020), que faz uma experiência chamada de *Hybrid Algorithm***,** em que combina Xgboost e Lightgbm, esse conjunto apresentou um AUC de 0,951, maior valor do que a performance dos dois algoritmos quando empregados isoladamente.

Houve a utilização de *Deep Learning* com a aplicação de uma interpolação entre os algoritmos BiLSTM e BiGRU em associação com uma técnica de *Random Over Sampling* no pré-processamento dos dados. Esta abordagem realizada por Najadat et al., (2020) e obteve um AUC de 0,913.

Por fim, apresenta-se uma síntese da modelação feita pelo time vencedor da competição do Kaggle através dos códigos partilhados no site[1], e que obteve um AUC de 0,945. O processo de *Feature Engineering* teve uma importância especial para o resultado obtido. Destaca-se na abordagem os seguintes processos:

Criação de variáveis com a função de ID das transações realizadas por cada cartão para tentar separar a fraude das transações legítimas.

Criação de *features* por agregação para cálculos de métricas como média, mediana e desvio padrão sobre diversas variáveis agrupadas pelos Ids criados e pelo cartão usado na transação. Essa técnica extrai uma sumarização estatística para as transações agrupadas de cada cartão.

Seleção das novas *features* pelo método *one feature model.* As novas *features* foram adicionadas uma a uma ao modelo, e então a avaliação foi feita pela métrica AUC. Se a

[1] https://www.kaggle.com/cdeotte/xgb-fraud-with-magic-0-9600

inclusão da nova *feature* proporcionasse aumento do AUC, a *feature* era mantida no modelo, caso contrário era descartada. Neste processo 28 novas *features* foram adicionadas.

Seleção de *features* por teste de "*Time Consistency*". Este processo de validação promoveu a criação de 242 modelos, e cada modelo contendo uma *feature* foi treinado com o primeiro mês dos dados de treino, e a seguir foi feita uma previsão sobre o último mês dos dados de treino. As *features* que apresentaram AUC acima de 0,5 em treino e validação foram mantidas, e essa fase de *Feature Engineering* eliminou 19 features. O algoritmo Xgboost foi utilizado e a validação foi feita pela técnica *GroupKFold* com 6 *splits*. A versão final do modelo produziu um modelo híbrido pela combinação dos algoritmos Xgboost, Catboost e Lgbm.

2.5 Ferramentas de análise de dados

Todo o processo de análise e preparação dos dados foi desenvolvido em linguagem R versão 4.1.0 com a *interface* RStudio versão 1.4.1717, a partir de uma máquina com sistema operacional Windows 10 Enterprise versão 20H2 de 64 bits, processador intel i7 1.80GHz e 16 GB de memória RAM. Para além do uso das diversas funções que compõem nativamente a base R para análises estatísticas e manipulação dos dados, muitas bibliotecas foram instaladas para possibilitar a realização de diferentes tarefas tais como consta na tabela 2.

Tabela 2 - Recursos utilizados em R

Recurso em R	**Finalidade**
matrix, data.table, purrr, tidyr	Armazenamento e leitura otimizada dos dados, manipulação de dataframes, matrizes, listas e vetores
dplyr, stringr, bit64, varhandle, ROSE	Manipulação e conversão de dados, data sampling
ggplot2	Visualização de dados
caret, corrplot, corrr	Criação de matriz de correlação, análise de variância, seleção de features de cordo com o grau de correlação
Psych	Análises de PCA e Cluster
mltools, rminer, mlbench, randomForest, xgboost, e1071, AUC	Separação dos dados em datasets para treino e validação, processo de modelação e cálculo da métrica AUC
Plataforma H2o	A plataforma H2o é implementada de forma distribuída e in-memory. Fornece um conjunto de algoritmos para aprendizagem supervisionada e não-supervisionada

2.6 Descrição dos algoritmos utilizados

Um algoritmo de classificação aplica uma função em um conjunto de dados de treino para gerar um estimador, quando a variável resposta é formada por um conjunto de valores nominais. O classificador é capaz de atribuir uma das possíveis classes a um exemplo que não possui o rótulo da classe (Gama et al., 2017). Neste trabalho foram utilizados cinco algoritmos, sendo três da família das árvores de decisão, nomeadamente Xgboost, Random Forest e GBM *(Gradient Boost Model*), e dois baseados em otimização, que são SVM *(Support Vector Machine)* e MLP *(Multi-Layer Perceptron)*.

Xgboost. O algoritmo Xgboost implementa um modelo conjunto de árvores de decisão de classificação e regressão (CART). Diferentemente das árvores de decisão, o CART atribui um score a cada uma das folhas e depois soma a predição de múltiplas folhas em conjunto (T. Chen & Guestrin, 2016).

Random Forest. O Random Forest é um algoritmo da família das árvores de decisão que usa classificações de múltiplas árvores aleatórias para produzir uma classificação agregada através da escolha da classe que obteve mais votos de cada árvore (Livingston, 2005).

GBM. O GBM atua como um conjunto de árvores de decisão treinadas sequencialmente sobre todas as *features* do *dataset* e que a cada iteração aprende com os erros residuais (Ke et al., 2017).

SVM. Em tarefas de classificação, o algoritmo SVM procura fronteiras que separem os dados de forma linear, e define um hiperplano capaz de separar as classes de forma que ocorra a maior distância do hiperplano para os dados de exemplo das diferentes classes que estejam mais próximos (R. C. Chen et al., 2004).

MLP. O Multi-Layer Perceptron utilizado neste trabalho é um tipo de Rede Neuronal Artificial (RNA) composta por unidades de processamento, chamadas neurónios artificiais. Esses neurónios são organizados em uma ou mais camadas intermediárias e uma camada de saída que são conectadas entre si por múltiplas ligações, de forma que o fluxo de informação comece em uma camada de entrada e termine na camada de saída. As conexões entre os neurónios aplicam pesos que ponderam a entrada recebida por cada neurónio e durante o processo de aprendizagem os pesos vão sendo ajustados para reduzir

os erros cometidos e aperfeiçoar o conhecimento da rede. O neurónio implementa uma função que é a combinação das funções implementadas pelos neurónios das camadas anteriores que lhe estão conectadas, desta forma, cada camada torna o processamento mais complexo (Gama et al., 2017). A figura 3 mostra de forma simplificada uma RNA de três camadas.

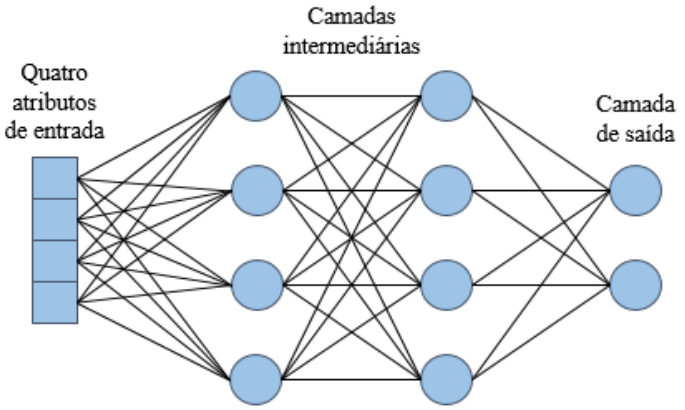


Figura 3 - Exemplo de uma MLP

Capítulo 3 - Dados e Metodologia

Este capítulo descreve as etapas de compreensão do negócio, compreensão e preparação dos dados. O capítulo está estruturado a partir das etapas da metodologia CRISP-DM e tem início com a secção 3.1 que apresenta a metodologia CRISP-DM. A secção 3.2 trata da compreensão do problema a ser resolvido. A secção 3.3 faz a descrição, exploração e análise da qualidade dos dados. A secção 3.4 descreve o processo de preparação dos dados, criação e seleção de *features*.

3.1 Metodologia CRISP-DM

A modelação foi desenvolvida segundo o método CRISP-DM que propõe um fluxo de trabalho com seis fases para conduzir o processo de *Data Mining (DM)*. A subdivisão em fases contribui para o planeamento e organização de projetos de *Data Science* (Chapman et al., 2000). Para a realização deste trabalho, as seguintes etapas foram executadas: Compreensão do negócio, compreensão dos dados, preparação dos dados, modelação e avaliação do desempenho. A fase de implementação prevista na metodologia não foi executada pois está fora do âmbito desta investigação.

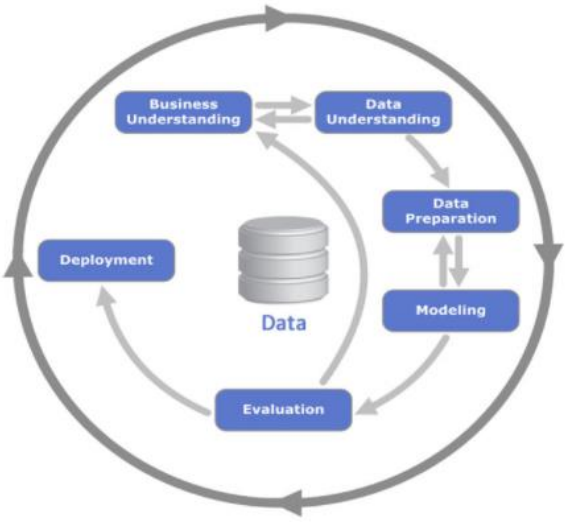


Figura 4 - Fluxo do CRISP-DM Fonte: Chapman et al (2000)

3.2 Compreensão do negócio

Esta é a etapa para o entendimento do problema a ser resolvido. A secção 3.2.1 faz a descrição dos requisitos da competição no *site* Kaggle. A secção 3.2.2 descreve as dimensões dos *datasets* de treino e teste. A secção 3.2.3 detalha as características da métrica utilizada para a avaliação dos resultados da modelação.

3.2.1 Descrição da competição

Nesta fase teve lugar a compreensão das características e regras da competição do Kaggle, e a identificação das características das transações de *e-commerce* contidas nos *datasets*. O objetivo de negócio relacionado a essa competição é melhorar a acurácia da deteção da fraude em vendas *on line*.

O cerne da competição é resolver um problema de classificação binária das transações como *Fraude* ou *Não Fraude*. A competição teve início em 15 de julho de 2019 e foi concluída em 4 de outubro de 2019 com a participação de 6.351 equipas. Houve premiação de 20 mil dólares distribuídos ente os 3 primeiros colocados. Os dados disponibilizados são dados reais de transações feitas em comércio eletrônico monitorizadas pela Vesta Corporation, que é uma das empresas líderes mundialmente em serviços de proteção de pagamento em compras *on line*. As previsões deveriam ser submetidas ao Kaggle por meio de um ficheiro no formato .csv com 506.691 transações e composto pelas colunas *TransactionID* e *isFraud*. A coluna *isFraud* contém o *score* com a probabilidade de fraude para cada transação. Após a submissão do ficheiro no *site* da competição, o cálculo do AUC é feito imediatamente pelo site e o resultado alcançado fica associado ao ficheiro submetido.

3.2.2 Descrição dos datasets

Os dados estão divididos em 4 *datasets*, sendo 2 *datasets* para treino e outros dois para teste. Os *datasets* estão listados na tabela 3.

Tabela 3 - Descrição dos datasets

	Dados de treino		Dados de teste	
Nome do *dataset*	**Nº de observações**	**Nº de variáveis**	**Nº de observações**	**Nº de variáveis**
transaction	590.540	394	506.691	393
identity	144.233	41	141.907	41

Os ficheiros *transaction* e *identity* podem ser unidos através da coluna *transactionID* que tem a função de chave primária. Esta variável contém valores numéricos únicos que identificam cada transação contida nos *datasets*. A junção do ficheiro *identity* com *transaction* apresenta o desafio de tratar valores omissos, pois, por exemplo, o ficheiro *train_transaction* apresenta 590.540 observações e o ficheiro *train_identity* apenas 144.234 observações, o que representa 76% de dados de identidade em falta quando feita a associação do *dataset train_identity* com *train_transact* através do campo *transactionID*. Para além disso, o ficheiro *train_identity* contém variáveis com até 96% de valores omissos.

3.2.3 Métrica de avaliação de desempenho

Para esta competição no Kaggle, a métrica escolhida para a medição do desempenho foi o *AUC (Area Under the ROC Curve)*. Para problemas de classificação, a métrica Acurácia é comumente utilizada, e é calculada pela fórmula apresentada abaixo.

$$Acurácia = \frac{número\ de\ verdadeiros\ positivos}{total\ de\ classificações} \quad (1)$$

Porém, a Acurácia não é a medida mais adequada quando há desbalanceamento de classes, pois a Acurácia é altamente influenciada pela distribuição das classes positiva e negativa. O AUC é preferido para medir o desempenho de classificadores com classes desbalanceadas, pois compara a performance do algoritmo por toda a gama de probabilidades da previsão das classes, indicando a capacidade do modelo em separar as classes (Ling et al., 2020). O gráfico ROC, representado na figura 5, tem duas dimensões, sendo o eixo X a taxa de FP (*False Positives*) ou Especificidade, e o eixo Y a taxa de TP (*True Positives*) ou Sensibilidade, que representam as compensações relativas entre os

benefícios das probabilidades dos TP e os custos dos FP (Fawcett, 2006). Quanto mais a curva estiver próxima do canto superior esquerdo do gráfico, melhor é o desempenho.

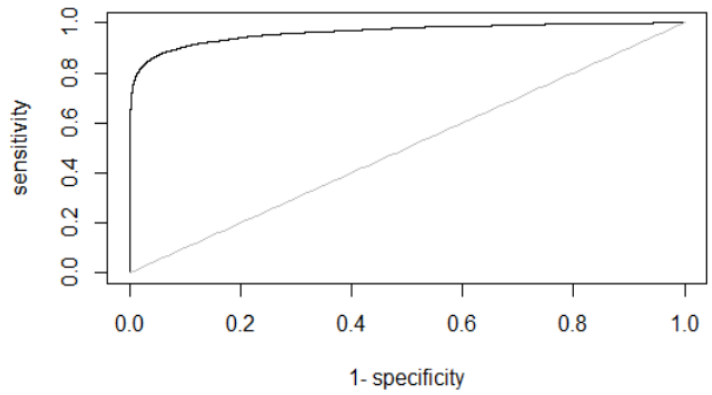


Figura 5 - Gráfico com curva ROC

3.3 Compreensão dos dados

Esta fase foi desenvolvida com a utilização de estatística descritiva e análise visual através de gráficos. O objetivo desta etapa é compreender as características de cada variável do *dataset*. A secção 3.3.1 trata da coleta e descrição dos tipos de dados e o que estes representam. A secção 3.3.2 descreve o processo de exploração dos dados através de sumarização estatística e gráficos.

3.3.1 Coleta e descrição dos dados

Os quatro *datasets* foram baixados do site Kaggle[2] e carregados em *RStudio*. A descrição dos tipos de dados e que tipo de informação cada variável representa será feita por meio de tabelas que dividirão as variáveis em grupos, pois há grupos de colunas que contêm dados com características semelhantes. As variáveis dos *datasets* *train_transaction* e *test_transaction* têm suas características descritas na tabela 4.

[2] Site com os dados da competição: https://www.kaggle.com/c/ieee-fraud-detection/data

Tabela 4 - Descrição das variáveis dos datasets transact

Variável	**Tipo**	**Descrição**
TransactionID	Integer	ID de números únicos e sequenciais para cada observação / transação;
isFraud	Boolean	1 - *Fraud*e, 0 - *Não Fraude*;
TransactionDT	Integer	Número inteiro sequencial que informa a data/hora de cada transação. A primeira transação apresenta o número 86.400 que é a quantidade total de segundos em um período de 24 horas;
TransactionAmt	Double	Regista o valor em dólar de cada transação;
ProductCD	Character	Informa o produto comprado na transação. São 5 produtos diferentes e o produto poderá ser um bem ou prestação de serviço adquiridos pelo cliente;
card1, card2, card3 e card5	Double	Apresentam informações sobre o cartão utilizado na transação tais como o tipo do cartão, categoria, banco emissor e país de origem;
card4 e card6	Character	Apresentam informações sobre o cartão utilizado na transação tais como o tipo do cartão, categoria, banco emissor e país de origem;
addr1 e addr2	Double	Fazem a indicação do país e região do endereço do cliente, porém de forma anónima. Os clientes podem ter mais de um endereço;
dist1 e dist2	Double	Marca distâncias entre endereços de faturamento, endereços de e-mail, código postal, endereços IP, área de telefone dentre outros;
P_emaildomain, R_emaildomain	Character	Indica o domínio dos e-mails do comprador e do recebedor da compra;
C1 até C14	Double	São indicações da quantidade de informações de identificação diferentes relacionadas ao cartão utilizado na transação. Alguns exemplos dessas informações são quantos endereços, números de telefone ou endereços de e-mail diferentes estão associados ao cartão. Essas informações são anônimas, portanto, as colunas indicam quantidades, mas não é possível saber a quantidade de qual informação o número se refere;
D1 até D15	Double	Representam marcações de tempo. Essas informações também são anônimas e apenas sabe-se que os números indicam, por exemplo: qual a diferença de tempo entre a transação atual e a última transação realizada com um mesmo cartão, há quantos dias foi realizada a primeira transação com o cartão e outras informações dessa natureza;
M1 até M3, M5 até M9	Boolean	Indica que há informações que coincidem entre transações feitas com um mesmo cartão quando se compara endereços, e-mails e outras informações de identificação;
M4	Character	Indica que há informações que coincidem entre transações feitas com um mesmo cartão quando se compara endereços, e-mails e outras informações de identificação;
V1 até V339	Integer	São *features* geradas a partir do enriquecimento de dados feito pela Vesta Corporation. Não é explicado o significado da informação que cada coluna contém;

Os datasets *train_identity* e *test_identity* têm as suas variáveis descritas na tabela 5.

Tabela 5 - Descrição das variáveis dos datasets identity

Variável	Tipo	Descrição
TransactionID	Integer	ID de números únicos e sequenciais para cada observação / transação;
Id_01 até id_34	Double & Character	Apresentam dados que identificam as transações de acordo com informações de conexão de rede tais como IP, ISP e Proxy. Também apresentam informações do dispositivo utilizado na transação, tais como sistema operacional, versão do OS, browser, além de registar comportamentos como o tempo que o utilizador permaneceu na página, quantas vezes ocorreu falha no login e outros dados desta natureza;
Id_35 até id_38	Boolean	Apresentam dados que identificam as transações de acordo com informações de conexão de rede tais como IP, ISP e Proxy. Também apresentam informações do dispositivo utilizado na transação, tais como sistema operacional, versão do OS, browser, além de registar comportamentos como o tempo que o utilizador permaneceu na página, quantas vezes ocorreu falha no login e outros dados desta natureza;
DeviceType	Character	Identifica se o dispositivo utilizado na transação é mobile ou desktop;
DeviceInfo	Character	Identifica marca e modelo do dispositivo utilizado na transação;

A coluna *isFraud* é a variável resposta nos dados de treino para ensinar o modelo a classificar as transações como *Fraude* ou *Não Fraude*, e esta coluna é do tipo *boolean* com o número um a identificar os registos do tipo *Fraude* e o número zero para *Não Fraude*. O *dataset* test_*transaction*.csv não apresenta a coluna *isFraud*, pois é o conjunto de dados de teste para submeter as previsões ao Kaggle.

3.3.2 Exploração dos dados

A secção 3.3.2.1 analisa os sumários estatísticos feitos para os grupos de variáveis de acordo com o tipo dos dados. A secção 3.3.2.2 apresenta gráficos com a análise visual da distribuição dos dados. A secção 3.3.2.3 aborda exclusivamente as características da variável *card1* (variável que identifica o cartão usado em cada transação), que representa os cartões utilizados nas transações. A secção 3.3.2.4 faz a análise de correlação dos *datasets*.

3.3.2.1 Sumário estatístico

A sumarização estatística foi feita através de três tabelas, sendo a tabela 6 dedicada variáveis do tipo booleano, a tabela 7 para dados categóricos e a tabela 8 para dados

numéricos. Os *datasets* de treino e teste foram unidos para a execução do sumário estatístico.

Através da análise do sumário estatístico para as variáveis do tipo booleano na tabela 6, destaca-se o alto percentual de valores omissos. Com exceção à variável M6 em que há 30% de omissos, as restantes variáveis apresentam mais de 40% de omissos.

Tabela 6 - Sumário estatístico para as variáveis do tipo booleano

	Frequência			%		
Variável	**Falso**	**Verdadeiro**	**NA's**	**Falso**	**Verdadeiro**	**NA's**
M1	56	649.436	447.739	0,0%	59,2%	40,8%
M2	61.169	588.323	447.739	5,6%	53,6%	40,8%
M3	131.248	518.244	447.739	12,0%	47,2%	40,8%
M5	240.155	196.962	660.114	21,9%	18,0%	60,2%
M6	419.433	349.499	328.299	38,2%	31,9%	29,9%
M7	444.604	71.344	581.283	40,5%	6,5%	53,0%
M8	323.650	192.325	581.256	29,5%	17,5%	53,0%
M9	74.040	441.935	581.256	6,7%	40,3%	53,0%

Através da tabela 7 é possível verificar a distribuição de cada classe nas variáveis categóricas.

Tabela 7 - Sumário estatístico para as variáveis com dados categóricos

ProductCD		
Categoria	**Freq.**	**%**
C	137.785	12,6%
H	62.397	5,7%
R	73.346	6,7%
S	23.046	2,1%
W	800.657	73,0%

card4		
Categoria	**Freq.**	**%**
NA's	4.663	0,4%
american express	16.009	1,5%
discover	9.524	0,9%
mastercard	347.386	31,7%
visa	719.649	65,6%

card6		
Categoria	**Freq.**	**%**
NA's	4.578	0,4%
charge card	16	0,0%
credit	267.648	24,4%
debit	824.959	75,2%
debit or credit	30	0,0%

P_emaildomain		
Categoria	**Freq.**	**%**
gmail.com	435.803	39,7%
yahoo.com	182.784	16,7%
NA's	163.648	14,9%
hotmail.com	85.649	7,8%
Anonymous .com	71.062	6,5%
aol.com	52.337	4,8%
(Other)	105.948	9,7%

R_emaildomain		
Categoria	**Freq.**	**%**
NA's	824.070	75,1%
gmail.com	118.885	10,8%
hotmail .com	53.166	4,8%
anonymous .com	39.644	3,6%
yahoo.com	21.405	2,0%
aol.com	7.239	0,7%
(Other)	32.822	3,0%

M4		
Categoria	**Freq.**	**%**
NA's	519.189	47,3%
M0	357.789	32,6%
M1	97.306	8,9%
M2	122.947	11,2%

Tabela 8 - Sumário estatístico para as variáveis numéricas (exceto as colunas V)

Variável	Não omissos	Média	Desvio Padrão	Min.	Máx.	Range	IQR	NA	% de Outliers
TransactionAmt	1.097.231	135	242	0	1.937	1.937	83	0%	11%
card1	1.097.231	9.926	4.894	1.000	.397	7.397	8.229	0%	0%
card2	1.079.644	363	158	100	600	500	297	2%	0%
card3	1.092.664	153	12	100	232	132	0	0%	0%
card5	1.088.425	200	41	100	237	137	60	1%	0%
addr1	965.916	291	102	100	540	440	122	12%	0%
addr2	965.916	87	3	0	102	92	0	12%	0%
dist1	453.743	104	346	0	0.286	10.286	0	59%	11%
dist2	74.063	234	543	0	11.623	1.623	0	93%	3%
C1	1.097.228	12	112	0	4.685	4.685	2	0%	14%
C2	1.097.228	13	129	0	5.691	5.691	2	0%	15%
C3	1.097.228	0	0	0	31	31	0	0%	1%
C4	1.097.228	3	58	0	2.253	2.253	0	0%	25%
C5	1.097.228	5	26	0	376	376	1	0%	16%
C6	1.097.228	8	61	0	2.253	2.253	1	0%	13%
C7	1.097.228	2	52	0	2.255	2.255	0	0%	12%
C8	1.097.228	4	72	0	3.331	3.331	1	0%	7%
C9	1.097.228	5	19	0	572	572	2	0%	11%
C10	1.097.228	4	72	0	3.257	3.257	0	0%	25%
C11	1.097.228	9	81	0	3.188	3.188	1	0%	12%
C12	1.097.228	3	73	0	3.188	3.188	0	0%	22%
C13	1.092.483	30	117	0	2.918	2.918	11	0%	20%
C14	1.097.228	7	41	0	1.429	1.429	1	0%	15%
D1	1.089.931	101	167	0	641	641	131	1%	24%
D2	581.665	178	187	0	641	641	28	47%	14%
D3	631.211	31	73	0	1.076	1.076	5	42%	16%
D4	851.458	158	224	(122)	1.091	1.213	168	22%	23%
D5	563.015	47	104	0	1.088	1.088	2	49%	17%
D6	197.970	78	176	(83)	1.091	1.174	0	82%	6%
D7	99.050	54	133	0	.088	1.088	0	91%	4%
D8	149.264	153	245	0	2.030	2.030	0	86%	5%
D9	149.264	1	0	0	1	1	0	86%	0%
D10	1.008.664	142	214	0	1.091	1.091	193	8%	24%
D11	641.426	84	226	(53)	883	936	97	42%	19%
D12	133.971	66	154	(83)	879	962	0	88%	5%
D13	185.336	18	75	0	1.066	1.066	0	83%	3%
D14	177.381	58	155	193)	1.085	1.278	0	84%	4%
D15	996.049	185	239	(83)	1.091	1.174	302	9%	20%

A tabela 8 mostra o sumário estatístico das variáveis do tipo numérico e foi realizado de forma conjunta para os dados de treino e teste. Os *datasets* apresentam muitas colunas com valores omissos e a tabela 8 apresenta o percentual de valores omissos para as 55

primeiras colunas dos *datasets* de treino e teste. As tabelas com a representatividade dos dados omissos das colunas restantes do *dataset transact* e do *dataset identity* estão no Apêndice E. A tabela 8 também mostra a análise de *outliers e o*s critérios para classificar os valores como *outliers* estão descritos nos tópicos um e dois, sendo N o valor de um ponto de dado, Q1 é o primeiro quartil, Q3 é o terceiro quartil e IQR é o *Inter Quartile Range* (Chandola et al., 2009):

1. Valor de N < Q1 - 1.5 * IQR
2. Valor de N > Q3 + 1.5 * IQR

3.3.2.2 Análise visual

A análise visual possibilita a comparação dos dados de forma rápida e intuitiva. As análises apresentadas nesta secção têm por objetivo principal comparar os *datasets* de treino e teste para verificar se há diferença na distribuição dos dados.

As variáveis *card2, card3 e card5* são numéricas e a distribuição comparativa dos dados de treino e teste é exibida na figura 6 através de gráficos do tipo *boxplot*. As 3 variáveis apresentam distribuições semelhantes nos *datasets* de treino e teste.

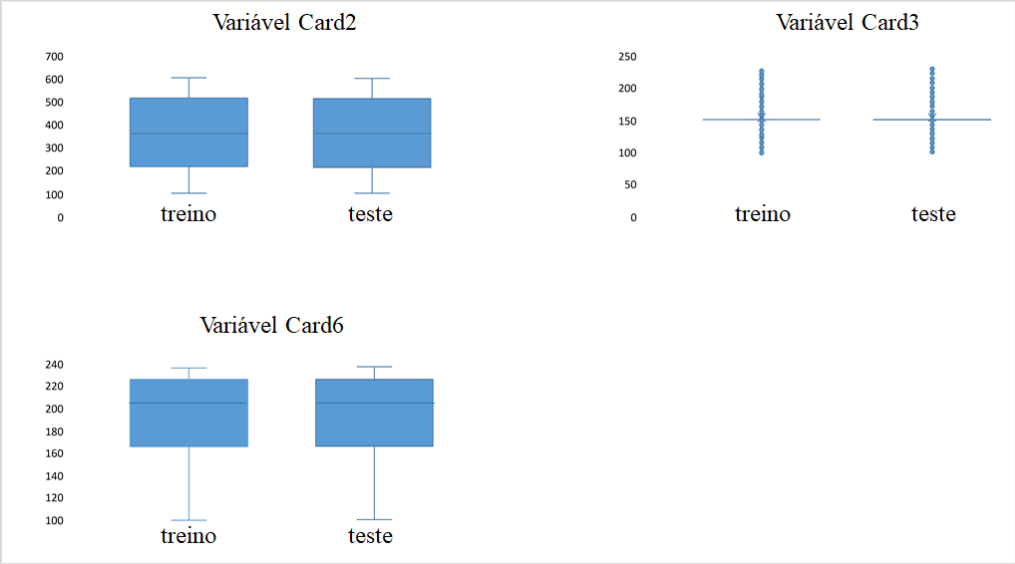


Figura 6 - *Boxplot* das variáveis *card2, card3 e card5*

As variáveis *card4 e card6*, que são categóricas, serão analisadas adiante através dos histogramas da figura 7, que demonstram que as variáveis categóricas *ProductCD, card, card6, P_emaildomain e R_emaildomain* apresentam distribuição muito semelhante nos dados de treino e teste.

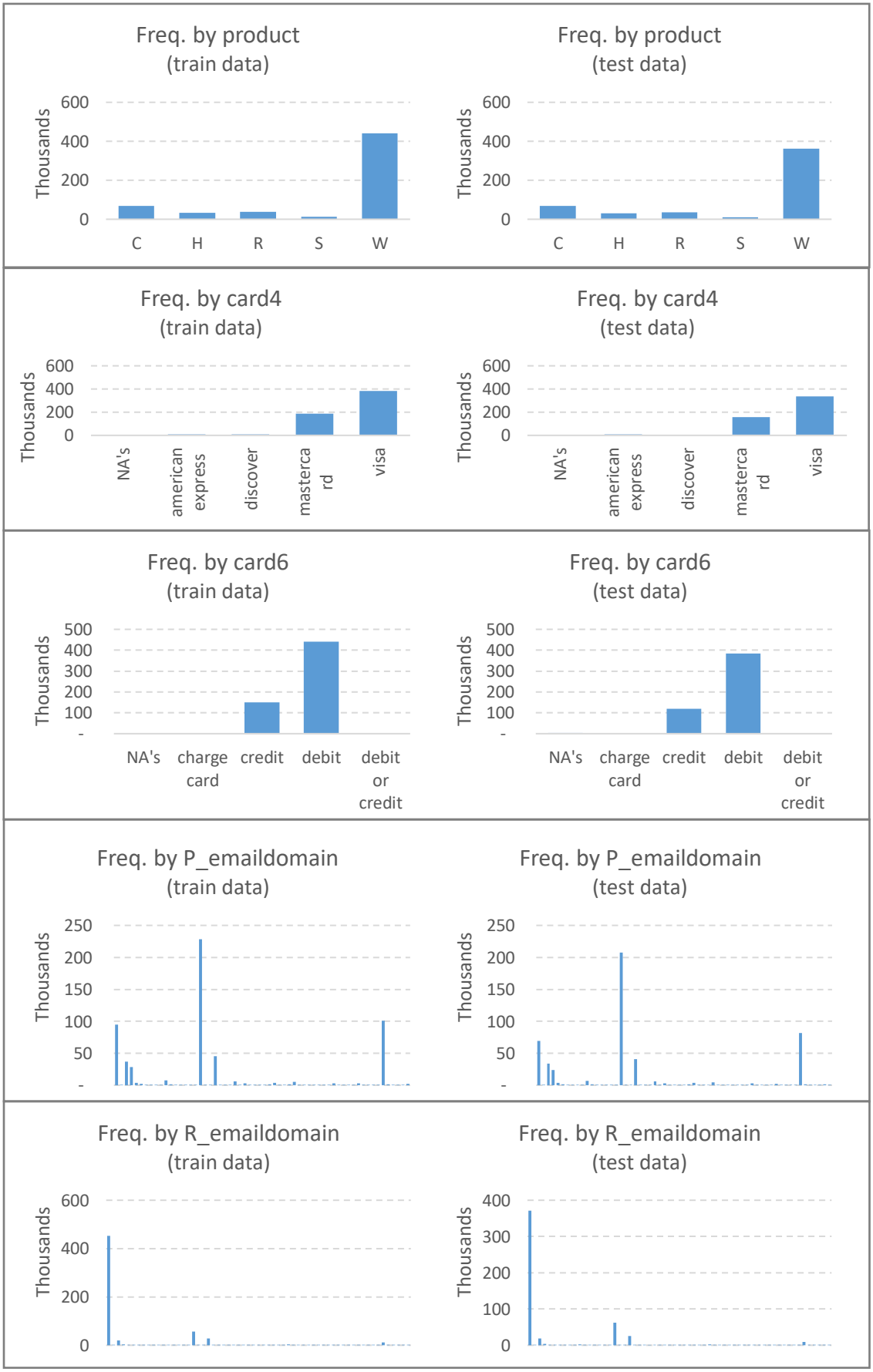


Figura 7 - Histogramas dos dados de treino e teste das variáveis categóricas

Relativamente à distribuição da fraude nas variáveis categóricas, foi feita a análise para identificar o percentual de fraude acumulado por cada categoria, sendo que esta visão é focada apenas sobre os dados de treino que possuem a classificação das transações como fraude ou não fraude.

Dentro da variável *ProductCD*, o produto "W" concentra mais de 70% das transações e apenas 43% das fraudes. O produto "C", porém, responde por 39% das fraudes.

O gráfico relativo à variável *card4* mostra que a marca Visa concentra a maior quantidade de transações, e a fraude apresenta uma distribuição alinhada com a distribuição das categorias.

No gráfico da variável *card6* verifica-se que o cartão do tipo *credit*, embora acumule apenas 25% das transações, apresenta índice de fraude de 48%, representando, portanto, um risco maior as transações feitas com cartão de crédito em relação às transações com cartão de débito.

A categoria gmail.com é a mais representativa e responde por 39% das transações e 48% das fraudes da variável *P_emaildomain*. O gráfico da variável *R_emaildomain* mostra que a categoria gmail concentra 10% das transações e 33% das fraudes e os e-mails omissos acumulam cerca de 46% do total das fraudes. Os gráficos com a distribuição da fraude são mostrados na figura 8.

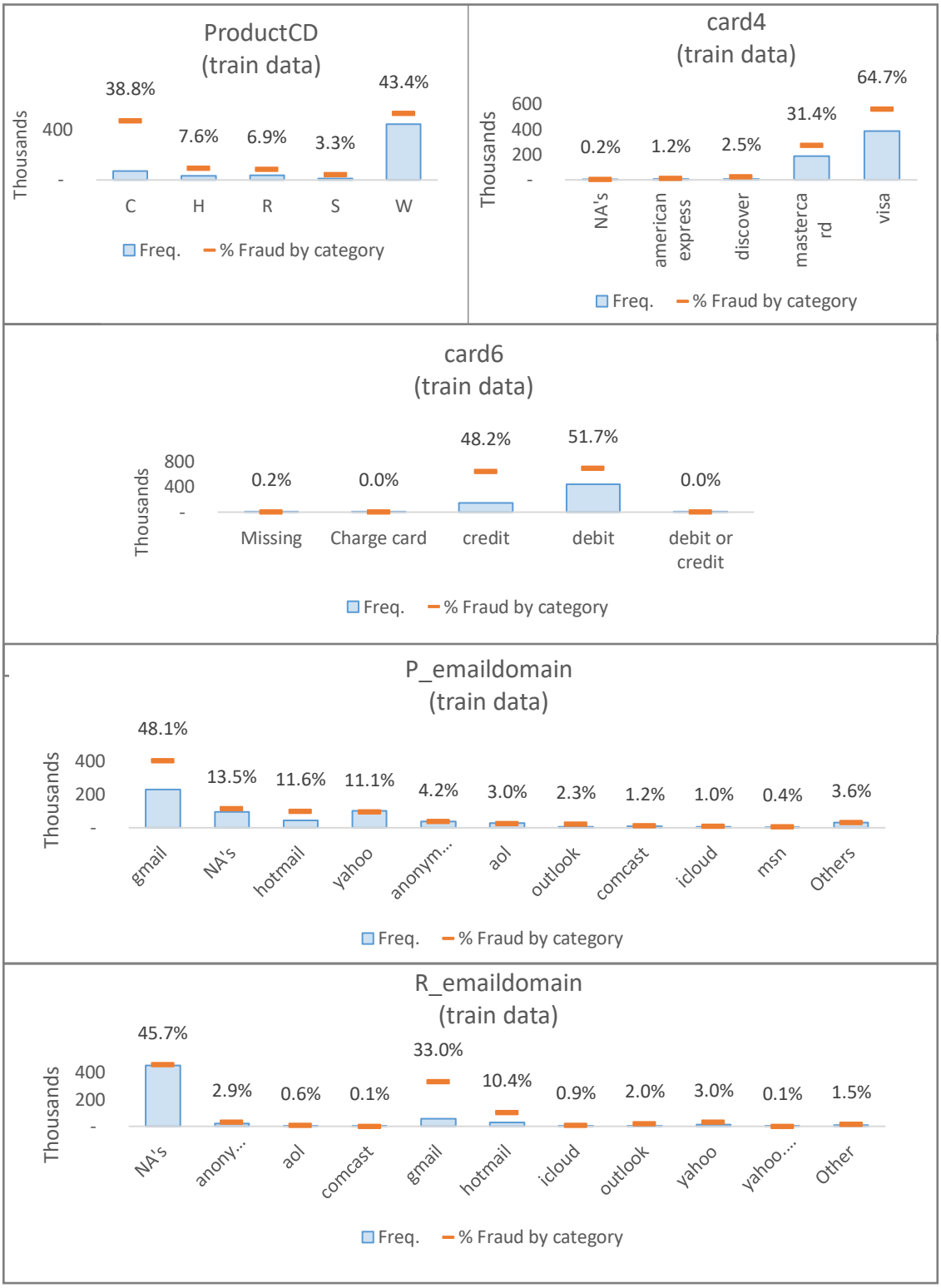


Figura 8 - Distribuição da fraude nas variáveis categóricas

3.3.2.3 Análise das variáveis Card1 e IsFraud

A variável *card1* é a variável utilizada para identificar o cartão que realizou a transação. Um único número de *card1* pode apresentar tanto transações legítimas quanto fraudes.

No *dataset* de teste há 13.244 números distintos para *card1*, e 27% destes cartões não estão no *dataset* de treino. Essa característica reforça a importância de preparar adequadamente os dados, com o foco em identificar o comportamento do fraudador de forma eficiente, e generalizar as características da fraude para um conjunto desconhecido de cartões. Uma possível solução para evitar que a variável *card1* cause viés no modelo devido às diferenças nos dados de treino e teste, é substituí-la por uma *feature* com o *frequence encoding* desta variável.

O *dataset* de treino é composto por 590.540 observações, sendo que destas, 20.663 são fraudes classificadas através da variável *isFraud*, portanto, regista um índice de 3.49% de fraude. A secção 2.3 abordou possíveis métodos para endereçar o problema do desbalanceamento de classes que vai do pré-processamento dos dados até a afinação de hiperparâmetros do modelo para que o algoritmo possa ter um desempenho ajustado ao desbalanceamento das classes.

3.3.2.4 Análise de correlação

A análise de correlação mostra a presença de muitas variáveis com forte correlação entre si. Devido à grande quantidade de colunas dos *datasets*, os gráficos de correlação de todas as variáveis serão mostrados no Apêndice D. A figura 9 mostra a correlação entre as primeiras 55 colunas e destaca-se o forte relacionamento interno entre as colunas tipo C (variáveis que indicam por exemplo, a quantidade endereços ou telefones associados a um cartão), e com menos intensidade o relacionamento entre as colunas tipo M (variáveis que indicam a existência dados que coincidem entre transações feitas por um mesmo cartão).

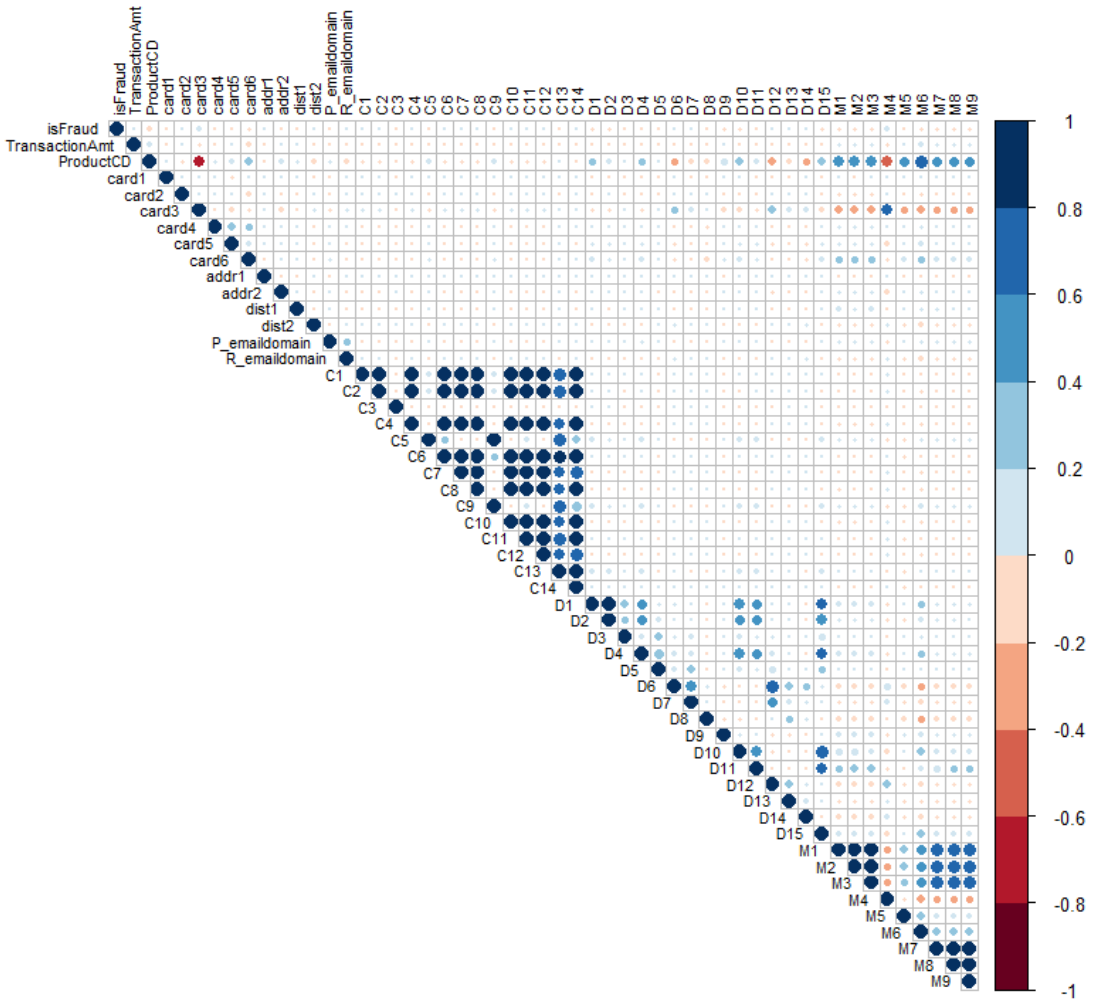


Figura 9 - Matriz de correlação das 55 primeiras variáveis dos dados transacionais

3.4 Preparação dos dados

Esta secção descreve o processo de preparação dos dados para a modelação, que é a etapa em que se concentra grande parte do esforço dedicado em resolver um problema através de *Machine Learning (ML)*. A secção 3.4.1 detalha a seleção e limpeza dos dados. A secção 3.4.2 apresenta o processo de formatação dos dados. A secção 3.4.3 descreve o processo de criação de novas *features* e a secção 3.4.4 trata dos métodos utilizados para fazer a seleção das *features*.

3.4.1 Seleção e limpeza dos dados

Relativamente ao processo de seleção dos dados, cabe apenas indicar que os *datasets* foram disponibilizados integralmente pelo patrocinador da competição no Kaggle, e não houve, portanto, uma fase de levantamento das fontes de dados para o desenvolvimento do trabalho.

A limpeza de dados tem por objetivo preparar o *dataset* de forma a corrigir erros e problemas nos dados, e também remover dados com características que resultem em diminuição do desempenho do modelo. As etapas realizadas para a limpeza dos dados são descritas a seguir.

Teste de retirada de variáveis com alta taxa de valores omissos: foi realizado um teste comparativo entre o desempenho de dois modelos de Xgboost com diferentes métodos de tratamento de valores omissos. No primeiro modelo as variáveis com mais de 60% de valores omissos foram eliminadas e os omissos restantes foram substituídos por um valor padrão de -999, sendo que, com a utilização deste critério foram removidas 179 de um total de 435 variáveis. Outro modelo foi desenvolvido e todos os valores omissos foram substituídos pelo padrão -999. A performance dos dois modelos foi comparada e o modelo que manteve todas as variáveis fazendo a imputação dos omissos pelo valor padrão -999 obteve um AUC superior. A partir deste teste foi tomada a decisão tratar os valores omissos por imputação. A análise de Schafer & Graham, 2002 demonstra que a retirada dos omissos é uma forma pobre de endereçar o problema e que sua principal virtude é a simplicidade.

Processo de imputação de omissos: este processo foi dividido de acordo com o tipo do dado a ser tratado, nomeadamente um processamento específico para dados booleanos, outro processamento para dados numéricos e por fim os dados categóricos. Foram feitas algumas tentativas de imputação de omissos através das bibliotecas *MICE*, *Amelia e Hmisc*, porém não foi possível utilizar as soluções prontas das bibliotecas do R devido ao elevado consumo de recursos computacionais, memória insuficiente do computador ou demasiado tempo de execução. A solução para a imputação dos valores omissos foi implementada de acordo com os tipos de dados do *dataset* e é descrita nos três parágrafos seguintes.

Dados booleanos: foram testadas duas abordagens para o tratamento dos omissos presentes nas variáveis booleanas. A primeira técnica foi a execução do *One Hot Encoding*, que desdobrou variáveis originais em novas variáveis representando apenas os valores zero e um de cada coluna e excluindo os omissos. A segunda abordagem foi imputar o valor -1 no lugar dos omissos. Foram desenvolvidos dois modelos Xgboost com as duas abordagens e a abordagem com a imputação do valor -1 para os omissos apresentou AUC superior, sendo mantido este método para a modelação final.

Dados numéricos: a substituição dos omissos em variáveis numéricas foi feita pelo valor da mediana da variável com agrupamento pela variável *card1* (variável que identifica o cartão usado em cada transação). Ou seja, a coluna *card1* foi utilizada como o ID do cartão usado em um conjunto de transações, e a mediana das transações para cada *card1* foi inserida no lugar dos valores omissos. Para os casos em que o cartão só tem uma ou duas transações, e não é possível calcular a mediana, foi feita a imputação dos omissos pela mediana da variável sem o agrupamento por *card1*.

Dados do tipo texto: a imputação dos omissos para as variáveis do tipo texto foi feita de forma semelhante às variáveis numéricas, agrupando os dados por *card1*, porém ao invés da mediana, utilizou-se a moda. Para os casos em que o cartão só tem uma ou duas transações, e não é possível definir a moda, foi feita a imputação dos omissos pela moda da variável sem o agrupamento por *card1*.

Tratamento de *outliers*: conforme mostrado na secção 3.3.2.1, o *dataset* apresenta muitas variáveis com diferentes taxas de *outliers*. Devido à fraude em transações com pagamento por cartão ser um problema em que, o fraudador realiza transações fora do perfil de utilização do cliente genuíno, optou-se em não aplicar tratamento aos *outliers*, pelo fato de que esses dados podem representar a ocorrência de fraudes e, portanto, o tratamento dos *outliers* causaria impacto negativo na deteção da fraude (Abbott, 2014).

Análise de variáveis com NZV *(Near Zero Variance and Zero Variance)*: A análise NZV identifica as variáveis que apresentam um único valor em todas as observações, o que configura variância igual a zero, e também verifica as variáveis com poucos valores únicos em relação ao número total de observações, e por fim as variáveis em que há um grande rácio da frequência do valor mais comum sobre a frequência do segundo valor mais comum. A depender do algoritmo utilizado na modelação, variáveis com estas

características podem atrapalhar a capacidade preditiva do modelo (Kuhn, 2008). A partir do resultado da análise NZV, 238 variáveis foram retiradas.

Análise de variáveis com alto índice de correlação: para a realização desta análise, uma matriz de correlação foi gerada. Através da função *findCorrelation* da biblioteca *Caret* em R, foi aplicado um limite de correlação maior ou igual a 0.9. A saída da função foi a recomendação da retirada de 50 variáveis altamente correlacionadas. A função *findCorrelation* inicialmente identifica pares de variáveis com a maior correlação absoluta, para ambas variáveis é calculada a correlação média entre cada variável e todas as demais variáveis, então a variável com a maior correlação média é retirada da matriz de correlação, e isto é feito em *loop* até a remoção completa de todas as variáveis acima do limite de correlação estabelecido, que neste caso é 0.9 de correlação (Kuhn, 2008).

3.4.2 Formatação dos dados

Os dados do tipo texto foram convertidos para o tipo numérico pela técnica de *label encoding*, que substituiu cada categoria por um número único. O *label encoding* foi executado sobre as variáveis: *ProductCD*, *Card4*, *Card6*, *M4*, *P_emaidomain e R_emaildomain*.

Há tipos de algoritmos, como por exemplo SVM e MLP, que atingem melhor desempenho quando os dados estão normalizados e apresentam média igual a zero e desvio padrão igual a um. Para aplicar a normalização aos dados utilizou-se a função *Scale* (Gama et al., 2017).

3.4.3 Criação de novas features

Esta secção descreve os critérios e a metodologia aplicada para a criação de novas *features*. A criação de novas *features* tem por objetivo enriquecer o *dataset* com informações que transmitem ao modelo características dos dados que não são tão evidentes nas variáveis originais, e tem por finalidade aumentar a capacidade de aprendizagem do modelo.

Foram criadas *features* para ampliar as possibilidades de análise e pré-processamento dos dados utilizando-se marcações de tempo. O *dataset* das transações está ordenado de forma crescente pela variável *TransactionDT*, e a primeira transação tem o valor 86.400, que é o total de segundos dentro de um período de 24 horas. As novas *features* temporais foram derivadas de cálculos sobre a variável *TransactionDT* e os cálculos para a criação das *features* temporais estão descritos abaixo:

1. **transaction_day**: TransactionDT / 86400
2. **transaction_week**: (transaction_day / 7) + 0.4
3. **transaction_month**: (transaction_week / 4) + 0.4
4. **transaction_hour**: (TransactionDT - (86400 * transaction_day)) / 3600
5. **transaction_minute**: ((3600 * transaction_hour) - (3600 * trunc(transaction_hour))) / 60
6. **transaction_second**: (TransactionDT - (86400 * trunc((TransactionDT / 86400)))) - ((3600 * trunc(transaction_hour)) + (60 * trunc(transaction_minute)))
7. **transaction_period**: essa *feature* marca períodos no dia em ciclos de três em três horas que têm início à hora zero. Os períodos foram definidos de acordo com a seguinte lógica:

   IF transaction_hour < 21

   THEN (transaction_period = transaction_hour / 3)

   ELSE transaction_period = 8

Através da análise de amostras de fraudes em transações realizadas por um único cartão é possível identificar variáveis que formam grupos com dados semelhantes entre si, o que possibilita separar de forma razoável a fraude das transações genuínas. Desta forma, ao concatenar essas variáveis, foram criadas novas opções de IDs para as transações feitas por um mesmo cartão.

A tabela 9 mostra a análise feita para o cartão de número 15.775 e pode-se identificar com mais clareza os grupos de variáveis com dados semelhantes entre si que separam a fraude do que não é fraude. As fraudes foram destacadas a cinzento e pode-se observar que as variáveis *addr1, P_emaildomain e dist1* formam grupos com dados semelhantes nas fraudes e diferentes das transações legítimas. Observa-se também nesta amostra, que o produto comprado pelo fraudador é mais frequentemente o de número cinco, enquanto

o cliente compra o produto quatro. E finalmente, as fraudes têm um valor médio maior para a variável *TransactionDT* em comparação com as transações legítimas.

Tabela 9 - Amostra de dados do card1 de número 15775

card1	TransactionAmt	TransactionDT	isFraud	ProductCD	addr1	P_emaildomain	dist1	D1
15775	240	137754	1	5	251	55	8	82
15775	40	165188	0	4	330	17	8	75
15775	108.95	260743	0	5	325	17	5	0
15775	120	308729	0	4	330	17	8	77
15775	260	314039	1	5	251	55	8	84
15775	70	395467	0	4	330	17	8	78
15775	35	495054	0	4	330	17	8	79
15775	120	743522	0	4	330	17	8	82
15775	120	743529	0	4	330	17	8	82
15775	250	845982	1	5	251	55	8	90
15775	115	861286	0	4	330	17	8	83
15775	115	862082	0	4	330	17	8	83
15775	115	920105	0	4	330	17	8	84
15775	315	920826	1	5	251	55	8	91
15775	115	921934	0	4	330	17	8	84
15775	115	924413	0	4	330	17	8	84
15775	115	943416	0	4	330	17	8	84
15775	390	1003956	1	5	251	55	8	92
15775	100	1043676	0	2	315	17	8	0
15775	475	1046150	1	5	251	55	8	93
15775	445	1046475	1	5	251	55	8	93
15775	445	1085966	1	5	251	55	8	93
15775	445	1086250	1	5	251	17	12	93
15775	77	1348721	0	5	184	10	3	0
15775	59	1349605	0	5	184	10	8	0

Portanto, a partir da análise com o objetivo de identificar as variáveis que formam perfis diferentes entre as fraudes e as transações legítimas, foram criadas três novas *features* pela concatenação de variáveis com essas características, e as três novas *features* são:

1. UID: concatenação das variáveis *card1, addr1 e P_emaildomain*;
2. UID1: concatenação das variáveis *card1, addr1, P_emaildomain, D1 e dist1*;
3. UID2: concatenação das variáveis *card1, addr1 e D1*;

Posteriormente, na modelação final, para evitar o fenômeno *Population Drift*, devido a presença de UIDs nos dados de treino que não estão presentes nos dados de teste, essas *features* foram substituídas por seus respetivos *frequency encoders*. Esses três novos Ids foram úteis para criar agregações de dados com a extração da média, mediana e desvio padrão para as variáveis numéricas.

Criação de *features* com o valor da média, mediana e desvio padrão para as variáveis: *TransactionAmt, addr1, addr2, dist1, dist2, P_emaildomain, R_emaildomain,* C1 a C14 (variáveis que indicam por exemplo, a quantidade endereços ou telefones associados a um cartão), M1 a M9 (variáveis que indicam a existência dados que coincidem entre transações feitas por um mesmo cartão). Foi feito o agrupamento dos dados de cada variável supramencionada pelos IDs card1 (variável que identifica o cartão usado em cada transação), UID, UID1 e UID2.

Criação de *features* com o valor da transação sobre os valores da média, mediana e desvio padrão para as variáveis *TransactionAmt, addr1, addr2, dist1, dist2, P_emaildomain, R_emaildomain,* C1 a C14, M1 a M9. Foi feito o agrupamento dos dados de cada variável supramencionada pelos IDs card1, UID, UID1 e UID2.

Criação de *features* com o *frequency encoding* das variáveis card1, UID, UID1 e UID2. A realização do *frequency encoding* destas variáveis tem a função de substituir na modelação a variável original pela frequência de seus valores, de forma que o modelo possa generalizar melhor. Com essa técnica o modelo aprende as frequências de cada categoria que compõe a variável, e não as categorias em si. O *frequency encoding* também foi feito para as variáveis categóricas *P_emaildomain, ProductCD e addr1* com o objetivo de transmitir ao modelo a frequência das categorias dessas variáveis de forma a tentar capturar melhor o padrão da fraude, pois há categorias nestas variáveis com concentração elevada de fraude.

Criação de *feature* com o valor da transação sobre o menor valor de transação do produto consumido. A lógica dessa *feature* é atribuir ao valor mínimo do produto o status de valor unitário. Ao calcular o valor da transação sobre o valor mínimo é possível identificar algo como a quantidade de produtos consumidos na transação. O objetivo é destacar através desta *feature* as transações com valores que se desviam muito do grupo das transações feitas por um cartão.

Criação de *feature* para marcar transações feitas em endereços diferentes num intervalo de tempo muito curto em relação à transação anterior. A deteção de fraude em transações de venda *on line* utiliza um método chamado Velocidade. A Velocidade ocorre quando duas transações são realizadas em sequência em um curto espaço de tempo, e em localizações distantes entre si, de forma que é impossível que as transações tenham sido realizadas pela mesma pessoa. Por exemplo, um mesmo cartão faz uma compra *on line* com um IP da Alemanha e 2 minutos depois o mesmo cartão faz uma compra com um IP de Portugal. É impossível que a mesma pessoa tenha feito as duas compras em sítios tão distantes num curto espaço de tempo, e a probabilidade de uma das transações ser uma fraude é muito alta (Fawcett & Provost, 1997).

Criação de *feature* para identificar transações feitas simultaneamente em lugares diferentes. O conceito de Colisão é parecido com o conceito de Velocidade, tendo como diferença o fato de que as duas transações são realizadas simultaneamente em sítios diferentes (Fawcett & Provost, 1997).

Criação de *feature* que marca sequências de transações com o mesmo valor de *TransactionAmt* (variável com o valor da transação em dólares) feitas em um curto espaço de tempo. Um comportamento comum do fraudador é fazer várias compras do mesmo produto e com o mesmo valor, após a primeira tentativa de fraude ser bem-sucedida. O fraudador tenta manter um padrão de compra semelhante ao da primeira transação, já que não houve o bloqueio da compra por parte da operadora do cartão que está a ser utilizado na fraude.

Inclusão de *feature* com a indicação da transação em que o *addr1* é diferente do *addr2* para um mesmo cartão. O *addr1* é o endereço do cadastro do cliente e o *addr2* o endereço de entrega da compra. Há muitos casos de fraude em que o fraudador utiliza o endereço de cadastro do cliente para *addr1*, porém insere um endereço de entrega (*addr2*) diferente. Com essa *feature* é possível passar ao modelo esse padrão de comportamento do fraudador.

A variável *TransactionAmt* (variável com o valor da transação em dólares) apresenta valores sem casas decimais e com casas decimais. Nos fóruns de discussão[3] da competição no Kaggle há análises que apontam que as transações com casas decimais

[3] https://www.kaggle.com/c/ieee-fraud-detection/discussion/101203

estão associadas a endereços e e-mails que seriam de países diferentes do país onde a venda *on line* é feita. E as casas decimais no valor da transação refletem a operação de conversão de câmbio da moeda. A variável *TransactionAmt* foi dividida em duas novas *features* com o número à esquerda do separador decimal e a outra *feature* com o número à direita do separador decimal.

Foram criadas 5 *features* baseadas na lei de Benford (*Benford Law*). A lei de Benford é uma técnica de análise para sequências numéricas que verifica desvios no padrão dos dígitos a partir da frequência do primeiro ou segundo dígito dos números (Bento et al., 2005). Por exemplo, supondo que para a variável *TransactionAmt*, o primeiro dígito tem o número dois como sendo o mais frequente, para as transações em que o primeiro dígito de *TransactionAmt* não for o número dois, essas transações serão consideradas desvios do padrão, podendo ser suspeitas de fraude. As *features* criadas a partir da análise com o critério da lei de Benford foram:

1. Desvio em relação ao primeiro dígito do valor de *TransactionAmt* à esquerda do separador decimal;
2. Desvio em relação ao segundo dígito do valor de *TransactionAmt* à esquerda do separador decimal;
3. Desvio em relação ao primeiro dígito do valor de *TransactionAmt* à direita do separador decimal;
4. Desvio em relação ao segundo dígito do valor de *TransactionAmt* à direita do separador decimal;
5. Desvio em relação aos dois primeiros dígitos do valor de *TransactionAmt* à esquerda do separador decimal;

Com o objetivo de capturar os padrões de comportamento para cada cartão, ao longo de períodos de três em três horas ao longo de um dia, a *feature* freq_transaction_period foi criada com o valor da quantidade de transações por períodos de três horas em três horas para cada cartão. Com a mesma técnica, porém, para um período de 24 horas completas, a *feature* freq_transaction_day foi criada.

Criação de *feature* que marca uma transação com valor diferente da média das transações feitas por um mesmo cartão nas 24 horas anteriores. Essa análise foi feita sobre a variável *ProductCD* para informar ao modelo as transações com perfil de compra de produtos diferente do perfil das últimas 24 horas que antecederam a transação atual.

Inclusão de *features* com o cálculo do desvio em relação ao Z Score. O Z Score é uma medida estatística calculada conforme mostrado na equação 2:

$$Z\ Score = \frac{[\text{valor da transação}]\ -\ [\text{média da váriável}]}{[\text{desvio padrão da váriável}]} \qquad (2)$$

Esta métrica pode ser utilizada para a identificação de *outliers* e foi aplicada às variáveis do tipo numérico com valores contínuos com o objetivo de identificar as transações fora do perfil do cliente legítimo (Chandola et al., 2009).

O processo de criação de *features* produziu um total de 673 novas *features* que foram posteriormente avaliadas na fase de seleção descrita na secção 3.4.4. O processo de seleção de *features* definiu total de 216 *predictors* para compor o conjunto de dados base do processo de modelação.

Antes da modelação ter início, porém, uma análise de *clusters* foi realizada e resultou na inclusão de uma *feature* com um *cluster* de sete grupos. A escolha das *features* utilizadas na análise de *clusters* foi feita através da matriz de importância com as 50 variáveis mais importantes de um modelo Xgboost, desenvolvido com as 216 *features* resultantes do processo de seleção descrito na secção 3.4.4. Sobre o conjunto das 50 *features* mais importantes foi gerada uma matriz de correlação que foi submetida ao critério do teste *Kaiser-Meyer-Olkin* (KMO) que valida se uma amostra de dados é adequada para a realização de uma análise PCA. O resultado do teste KMO foi 0.8 e este valor indica que os dados são adequados para a realização da PCA (Dziuban & Shirkey, 1974). Os dados foram normalizados com a função *Scale* e a PCA foi gerada pela função *principal* da biblioteca *Psych.* A partir dos *Eigenvalues* calculados pela função *principal* foi verificado que a decomposição dos dados em dezesseis componentes explica 74% da variância dos dados, como pode ser observado na tabela 10.

Tabela 10 - PCA com 16 componentes

	SS loadings	Proportion Var	Cumulative Var
PC1	9.310	0.186	0.186
PC2	5.133	0.103	0.289
PC3	3.305	0.066	0.355
PC4	2.964	0.059	0.414
PC5	2.583	0.052	0.466
PC6	1.843	0.037	0.503
PC7	1.651	0.033	0.536
PC8	1.489	0.030	0.566
PC9	1.242	0.025	0.590
PC10	1.177	0.024	0.614
PC11	1.112	0.022	0.636
PC12	1.048	0.021	0.657
PC13	1.022	0.020	0.678
PC14	1.005	0.020	0.698
PC15	0.982	0.020	0.717
PC16	0.948	0.019	0.736

Os *scores* das dezesseis componentes foram calculados e a análise de *cluster* foi baseada na distância Euclidiana. A figura 10 mostra como a fraude ficou distribuída dentro dos sete clusters representados no eixo X, sendo que os clusters dois e quatro apresentam maior concentração de fraude. Os *clusters* foram criados por meio da função *Kmeans* a partir dos *scores* das componentes gerados na análise PCA.

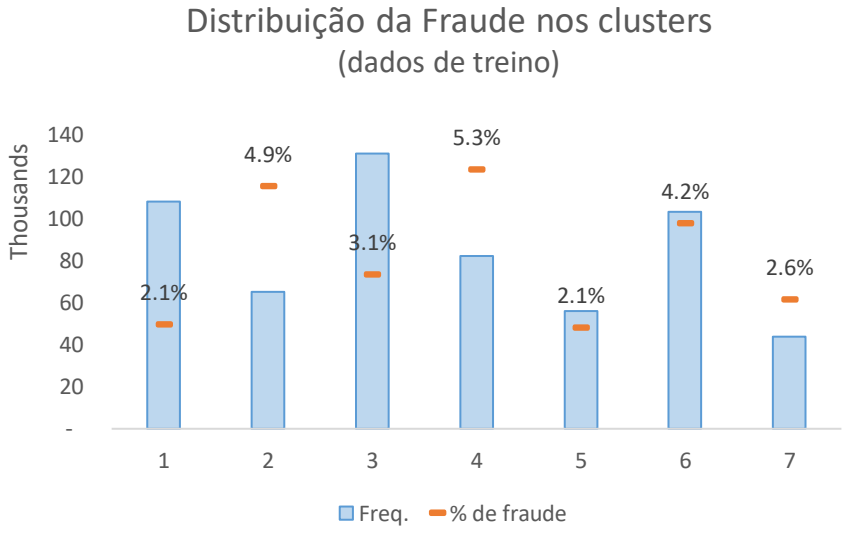


Figura 10 – Distribuição da fraude nos *clusters*

3.4.4 Seleção de features

Esta secção faz o detalhamento dos métodos utilizados para a seleção das *features* que foram utilizadas na modelação. A secção 3.4.4.1 apresenta a análise das *features* com zero variância ou variância próximo a zero. A secção 3.4.4.2 descreve o processo de eliminação de *features* com alta correlação entre si. A secção 3.4.4.3 detalha a seleção de *features* que, ao serem incluídas individualmente em um modelo desenvolvido com o algoritmo Xgboost, foram capazes de aumentar o AUC. A secção 3.4.4.4 descreve o processo de eliminação de *features* que, por terem mudança nas características entre os dados de treino e teste, causam viés na aprendizagem e por consequência o *overfit*;

3.4.4.1 Verificação de features com NZV (Near Zero Variance)

As *features* construídas na fase de *feature engineering* foram submetidas à uma avaliação NZV com o objetivo de retirar aquelas que apresentam variância igual a zero ou próximo a zero. Para a realização desta análise os dados foram normalizados, e depois foi feita a matriz de correlação. Finalmente, a matriz de correlação foi submetida à função *nearZeroVar* da biblioteca *Caret* que cria um *dataframe* com os resultados da análise (Kuhn, 2008). Através deste método foram retiradas 83 *features*.

3.4.4.2 Features com alta correlação

As *features* com alta correlação entre si foram retiradas por meio da utilização da função *findCorrelation* da biblioteca *Caret* em R, com o *cutoff* parametrizado a 0.9. A função *findCorrelation* verifica os valores absolutos das correlações em pares de variáveis. Quando duas variáveis estão correlacionadas acima do *cutoff*, a função indica a retirada da variável com a maior correlação absoluta média (Kuhn, 2008). Através desta técnica foram retiradas 278 *features*.

3.4.4.3 Avaliação individual das novas features

Este método de seleção de *features* foi aplicado por meio da criação de um modelo base com as primeiras 55 variáveis do transaction *dataset*, mais a inclusão de uma nova *feature* criada na fase de *feature engineering*. Inicialmente, um modelo em Xgboost

contendo apenas as primeiras 55 variáveis originais teve o AUC medido com dados de validação. Em seguida, de forma recursiva, cada nova *feature* foi adicionada individualmente ao modelo base, e as *features* que resultaram em um aumento do AUC foram mantidas para a fase seguinte de seleção. Aquelas *features* que não geraram aumento do AUC foram descartadas. Através deste método foram retiradas 265 *features*.

3.4.4.4 Adversarial validation

Os *datasets* apresentam uma característica chamada *Population Drift*, que é a mudança na distribuição das variáveis ao longo do tempo. O impacto negativo desse tipo de variável é que o modelo pode apresentar *overfit*, pois aprende características nos dados de treino que não estão presentes nos dados de teste resultando em queda na performance. Para endereçar este problema, a técnica *Adversarial Validation* foi aplicada aos dados.

A técnica *Adversarial Validation* é um método utilizado para identificar as variáveis que apresentam mudanças nas características entre os dados de treino e teste. A técnica foi realizada adicionando-se uma variável binária ao *dataset*, com a marcação do valor zero para os dados de treino e do valor um para os dados de teste. Então, um modelo com o algoritmo Xgboost foi treinado para prever as observações pertencentes aos *datasets* de treino e teste. Com este método foi possível verificar através da matriz de importância das variáveis do modelo, que a coluna *TransactionDT* foi a mais importante para o modelo classificar os dados de treino e teste. A variável *TransactionDT* indica a data da transação, e é uma variável numérica contínua e com valores únicos. O modelo apresentou um AUC de 0.8712 e após a retirada da variável *TransactionDT* o AUC baixou para 0.6846. Variáveis categóricas que apresentem diferenças grandes na distribuição dos dados de treino e teste podem ser substituídas por suas *frequency encoders* para reduzir o efeito *population drift*. Com esta análise, a variável *TransactionDT* foi excluída do modelo (Awad & Khanna, 2015). Adicionalmente, a análise da matriz de importância mostrou outras *features* com certo peso para o modelo classificar os dados de treino e teste, e foram retiradas as *fetaures*: *freq_full_card2, freq_full_card3, freq_full_card4, freq_full_card5, freq_full_card6, V60, D4, D15 e freq_transaction_day*.

Capítulo 4 - Processo de Modelação

Este capítulo descreve o processo de tomada de decisão para a escolha dos algoritmos, a separação entre dados para treino e validação dos modelos, e os hiperparâmetros utilizados na construção dos modelos. A secção 4.1 apresenta o processo de escolha dos algoritmos. A secção 4.2 descreve o processo de separação dos dados para treino e validação. A secção 4.3 detalha a construção dos modelos e a secção 4.4 explica os métodos de *Ensemble Learning*.

4.1 Escolha dos algoritmos

Para a definição do algoritmo a ser utilizado na fase de modelação (Chapman et al., 2000), foram feitas algumas perguntas que apoiam a tomada de decisão: Qual é o tipo de problema a ser resolvido? É aprendizagem supervisionada, ou não? Quais os tipos de dados disponíveis? Quais as características dos dados?

A partir das perguntas chegou-se à seguinte delimitação: O problema é de classificação binária com dados que possibilitam o processo de aprendizagem supervisionada. O *dataset* apresenta dados categóricos, numéricos, variáveis binárias e variáveis de marcação temporal. Além disso, na fase de preparação dos dados optou-se em manter os *outliers* sem nenhum tratamento para não descaracterizar a fraude.

Posto isto, deve-se destacar que algoritmos de regressão são sensíveis a *outliers* e podem perder desempenho com a presença destes. Algoritmos SVM e redes neuronais precisam que os dados estejam na mesma escala e, portanto, o *dataset* necessita de mais uma etapa de pré-processamento para estar adequado para a modelação. Algoritmos da família das árvores de decisão são menos sensíveis a *outliers* e até mesmo valores omissos. Dentro do ramo das árvores de decisão, o Random Forest destaca-se pelo bom desempenho, e nos últimos anos o Xgboost e LightGbm têm sido empregados com eficiência em muitas competições do Kaggle (Priscilla & Prabha, 2020). Diante do exposto, decidiu-se desenvolver as modelações com os algoritmos: Xgboost, Random Forest, SVM, GBM e MLP.

O modelo com Xgboost foi construído a partir da função xgboost da bliblioteca xgboost versão 1.4.1.1. A biblioteca fornece uma implementação de algoritmos de ML

do tipo *gradient boosting* desenvolvidos para serem rápidos e eficientes[4]. O Random Forest foi construído a partir da função H2o.randomForest da plataforma H2o e realiza a classificação a partir de florestas de árvores de decisão com *inputs* aleatórios[5]. O SVM foi construído a partir da função svm da bliblioteca e1071 versão 1.7-7[6]. O modelo GBM foi construído a partir da função h2o.gbm da plataforma H2o[7]. Finalmente, o MLP foi construído a partir da função h2o.deeplearning da plataforma H2o[8].

4.2 Separação dos dados para treino e validação

A validação do desempenho do modelo é um processo crucial para o aperfeiçoamento do trabalho. Para validar o desempenho, o *dataset* de treino foi dividido em duas partes de forma aleatória através da função *Holdout* da biblioteca *Rminer*, sendo que dois terços dos dados foram utilizados para treinamento e um terço para validação.

O processo de treinamento e validação de todos os modelos foi feito com os mesmos *datasets*, para desta forma para garantir a igualdade de condições no processo de previsão e validação, além de possibilitar uma comparação de *performance* consistente entre os algoritmos.

4.3 Construção dos modelos

Esta secção apresenta os detalhes e hiperparâmetros utilizados para a construção dos modelos utilizados neste trabalho. A secção 4.3.1 descreve o processo de modelação com o algoritmo Xgboost. A secção 4.3.2 descreve o processo de modelação com o algoritmo Random Forest. A secção 4.3.3 descreve o processo de modelação com o algoritmo GBM. A secção 4.3.4 descreve o processo de modelação com o algoritmo SVM. A secção 4.3.5 descreve o processo de modelação com MLP.

---

[4] https://xgboost.readthedocs.io/en/latest/

[5] https://cran.r-project.org/web/packages/randomForest/randomForest.pdf

[6] https://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf

[7] https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/gbm.html

[8] https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/deep-learning.html

4.3.1 Algoritmo Xgboost

A tabela 11 apresenta os parâmetros utilizados para a criação do modelo. Os parâmetros foram afinados ao longo de uma sequência de testes com 27 ciclos de treino e validação.

Tabela 11 - Parâmetros do modelo Xgboost

Parâmetros	**Valor**
max.depth	10
eta	0.2
nthread	7
nrounds	500
objective	"binary:logistic"
verbose	1
early_stopping_rounds	10
eval_metric	"auc"
tree_method	"hist"
scale_pos_weight	0.035
colsample_bytree	0.6
min_child_weight	1
subsample	1

4.3.2 Algoritmo Random Forest

Os parâmetros foram otimizados com o suporte do algoritmo GA (*Genetic Algorithm*) com uma amostra de 5.000 observações do *train_dataset*, e realizou a validação com uma outra amostra de 5.000 observações do *train_dataset*. O algoritmo genético trabalha combinando os parâmetros que recebe como entrada, e ao longo das iterações aprende quais são os valores que, combinados resultam no melhor desempenho. A métrica AUC foi utilizada para avaliação de cada iteração, sendo que, ao fim, foi impresso um relatório com os parâmetros que proporcionaram o melhor desempenho. Foi criada uma função para a otimização do Random Forest com os parâmetros do algoritmo GA, que recebeu a entrada do seguinte conjunto de parâmetros:

1. Parâmetro **ntrees** com valores no intervalo entre 100 e 1.000.
2. Parâmetro **max_depth** com valores no intervalo entre 1 e 30.
3. Parâmetro **mtries** com valores no intervalo entre 1 e 30.

A tabela 12 apresenta os parâmetros utilizados para a criação do modelo.

Tabela 12 - Parâmetros do modelo Random Forest

Parâmetros	Valor
ntrees	700
max_depth	20
mtries	-1
stopping_metric	AUC
min_rows	5

4.3.3 Algoritmo GBM

A afinação dos parâmetros foi feita com a utilização da técnica *Grid Search*, que testou um conjunto de parâmetros pré-definidos durante um ciclo de iterações. Ao fim do processo de *Grid Search* foi gerada uma tabela com os parâmetros dos modelos com o melhor desempenho. A tabela 13 apresenta os parâmetros utilizados para a criação do modelo.

Tabela 13 - Parâmetros do modelo GBM

Parâmetros	Valor
nfolds	5
ntrees	1000
max_depth	16
learn_rate	0.1
stopping_rounds	10
col_sample_rate	0.4
col_sample_rate_change_per_level	0.99
col_sample_rate_per_tree	0.26
sample_rate_per_class	(1, 0.5)
min_split_improvement	0.000001
stopping_metric	"AUC"

4.3.4 Algoritmo SVM

A afinação dos parâmetros foi feita por meio da função *tune* da biblioteca *e1071* e o modelo foi treinado pela técnica 5-*fold cross validation*. Após a afinação dos parâmetros com a função *tune*, os parâmetros que proporcionaram o melhor desempenho foram aplicados ao modelo final. A tabela 14 apresenta os parâmetros utilizados para a criação do modelo.

Tabela 14 - Parâmetros do modelo SVM

Parâmetros	Valor
type	'C-classification'
kernel	"linear"
cost	1.7
scale	TRUE
class.weights	"inverse"
cross	5
probability	TRUE

4.3.5 Algoritmo MLP

A tabela 15 apresenta os parâmetros utilizados para a criação do modelo.

Tabela 15 - Parâmetros do modelo MLP

Parâmetros	Valor
distribution	"bernoulli"
hidden	c(3, 5, 5)
epochs	10000
nfolds	5
train_samples_per_iteration	-1
reproducible	TRUE
activation	"MaxoutWithDropout"
single_node_mode	FALSE
balance_classes	FALSE
force_load_balance	FALSE
seed	23123
score_training_samples	0
score_validation_samples	0
standardize	TRUE
input_dropout_ratio	0.2
hidden_dropout_ratios	C(0.3, 0.3, 0.3)
l1	0
l2	0
stopping_metric	'AUC'
stopping_rounds	10

4.4 Combinação dos modelos

Esta secção apresenta o processo feito para a combinação das previsões (*Ensemble Models*) dos modelos apresentados na secção 4.3. A preparação inicial para os testes com

as diferentes técnicas de combinação de modelos foi criar um *dataframe* com seis variáveis, sendo uma a variável resposta e as outras cinco variáveis são os *scores* das previsões feitas pelos cinco modelos. A secção 4.4.1 descreve o processo de combinação de modelos pela média das previsões. A secção 4.4.2 descreve o processo de combinação através de um meta modelo. A secção 4.4.3 descreve o processo de combinação de modelos por voto maioritário.

4.4.1 Combinação pela média das previsões

A partir do *dataset* composto pelas previsões dos cinco modelos, foi construída uma nova variável com a média das previsões de cada observação do *dataset* com as previsões apenas dos modelos em Xgboost, Random Forest e GBM. Os modelos em SVM e MLP não foram utilizados no cálculo da média por que ambos tiveram AUC com cinco pontos percentuais abaixo dos três modelos com melhor desempenho, e quando foram introduzidos no cálculo da média das previsões causaram a redução do AUC.

4.4.2 Combinação através de um meta modelo

Este tipo de *Ensemble Learning* é feito aplicando-se uma modelação sobre as previsões realizadas por diferentes modelos. Essa abordagem permite que um algoritmo seja treinado com a variável resposta juntamente com os scores gerados por outros algoritmos, e assim o meta modelo aprende a combinar as diferentes previsões de forma otimizada, para tentar obter um resultado melhor que cada modelo individualmente. Este método foi realizado através de uma modelação em Xgboost e outra com Regressão linear sobre o *dataset* com as previsões dos três melhores modelos, nomeadamente Xgboost, Random Forest e GBM.

4.4.3 Combinação por voto maioritário das previsões

A técnica de combinar os modelos por voto maioritário consiste em fazer a classificação das observações de acordo com as previsões feitas pela maioria dos modelos. Esta técnica foi implementada de acordo com a seguinte regra: se o *score* dos três melhores modelos for superior a 0.5, então a observação recebe o maior *score* dentre os três modelos, caso contrário a observação é classificada com o menor *score*.

Capítulo 5 - Resultados e Discussão

Este capítulo apresenta os resultados alcançados na modelação e discute as experiências que mais se destacaram durante os ciclos de pré-processamento dos dados e modelação. Este capítulo corresponde à fase de avaliação prevista no CRISP-DM. A secção 5.1 aborda a contribuição das técnicas de sampling para o desempenho dos modelos. A secção 5.2 discute a contribuição das técnicas de deteção de anomalia e dos métodos de identificação de padrões de comportamento para a descoberta do perfil da fraude. A secção 5.3 avalia o desempenho dos modelos individualmente. A secção 5.4 apresenta o desempenho dos modelos de forma conjunta. A secção 5.5 descreve o processo de submissão ao Kaggle da previsão com os dados de teste e o resultado final alcançado.

5.1 Contribuição das técnicas de *sampling*

Um dos objetivos da dissertação é investigar se a utilização de técnicas de pré-processamento dos dados para promover o balanceamento das classes, contribui para o aumento do desempenho dos diferentes algoritmos utilizados na modelação. A secção 2.2.1 introduziu os fundamentos do problema do desbalanceamento de classes. A secção 2.3 em conjunto com o Apêndice C detalharam as principais técnicas utilizadas para promover o balanceamento das classes, mais especificamente no âmbito da deteção de fraude.

Foram realizados testes de pré-processamento dos dados com as técnicas de *sampling: SMOTE, ROSE, Oversampling* e *Undersampling*. Estes testes foram feitos com uma amostra de 5.000 observações dos dados de treino para a aprendizagem dos modelos e 5.000 observações dos dados de treino para a validação. A partir de cada técnica de *sampling* foi gerado um novo *dataset* para treinamento dos modelos, portanto, foram criados quatro novos *datasets* para treino.

Foram feitas modelações com os cinco algoritmos a partir dos dados de treino sem a aplicação das técnicas de *sampling*. Esta modelação foi denominada *baseline* e serviu para comparar o desempenho dos modelos entre si e também o desempenho de cada modelo com os quatro novos *datasets* de treino preparados com as técnicas de *sampling*.

A validação de todas as modelações foi realizada com o mesmo *dataset*, que não recebeu qualquer processamento para promover o balanceamento das classes. A tabela 16 mostra o AUC obtido por cada modelo treinado com os *datasets* processados com as técnicas de *sampling* em comparação ao modelo *baseline*.

Tabela 16 - Resultado das técnicas de sampling

Modelo	**Baseline**	**SMOTE**	**ROSE**	**Over sampling**	**Under sampling**
Xgboost	**0.8262**	0.7097	0.5430	0.7873	0.7133
Random Forest	**0.8302**	0.8250	0.7272	0.8240	0.8027
GBM	**0.8273**	0.8111	0.7176	0.7839	0.7838
SVM	**0.7882**	0.2126	0.7737	0.7791	0.6269
MLP	0.7463	**0.7881**	0.7839	0.7839	0.7839

Os resultados mostraram que os algoritmos Xgboost, Random Forest, GBM e SVM foram mais robustos ao lidar com o forte desbalanceamento dos dados, pois as modelações feitas com o *dataset* sem *sampling* obtiveram melhores resultados. O algoritmo *MLP* se beneficiou do balanceamento das classes e obteve melhor desempenho através da técnica *SMOTE*.

5.2 Contribuição das técnicas de deteção de anomalias

Outro objetivo desta investigação é analisar como as técnicas de processamento dos dados para deteção de anomalias contribuem para ampliar a capacidade de aprendizagem dos algoritmos sobre os padrões da fraude.

Na fase preparação dos dados foram criadas novas *features* a partir do emprego de técnicas de deteção de anomalias, tais como: *Benford Law, Time Window e Z-Score*. Com o processo de seleção de *features* foram escolhidas 216 *features* para o treinamento dos modelos. A tabela 17 mostra uma segmentação das *features* de acordo com suas características, sendo a categoria chamada **Original** composta pelas *features* originais do *dataset*, a categoria **Medida Estatística** são *features* contruídas a partir de medidas como média, mediana e desvio padrão tiradas de transações agrupadas pelo *card1* (variável que identifica o cartão usado em cada transação) ou pelos outros IDs. Por fim, a categoria

Deteção de Anomalia é o conjunto de *features* criadas a partir das técnicas de deteção de anomalia.

Tabela 17 - Distribuição das features de acordo com a característica

Característica da *feature*	**Frequência**	**Representatividade**
Original	97	45%
Medida Estatística	93	43%
Deteção de Anomalia	26	12%
Total	**216**	**100%**

A biblioteca *xgboost* em R provê a função *xgb.importance* para extrair uma medida da importância de cada *feature* para o modelo. A função retorna, dentre outras métricas, o valor do g*ain*, que é o quanto a acurácia aumentou quando um ramo da árvore foi dividido pela *feature* (Lundberg et al., 2018). A figura 11 apresenta as 21 *features* com maior *gain* para o modelo, e apenas uma *feature* derivada das técnicas de deteção de anomalia surge neste *ranking*, nomeadamente a *feature* Zscore_C11 na 17ª posição.

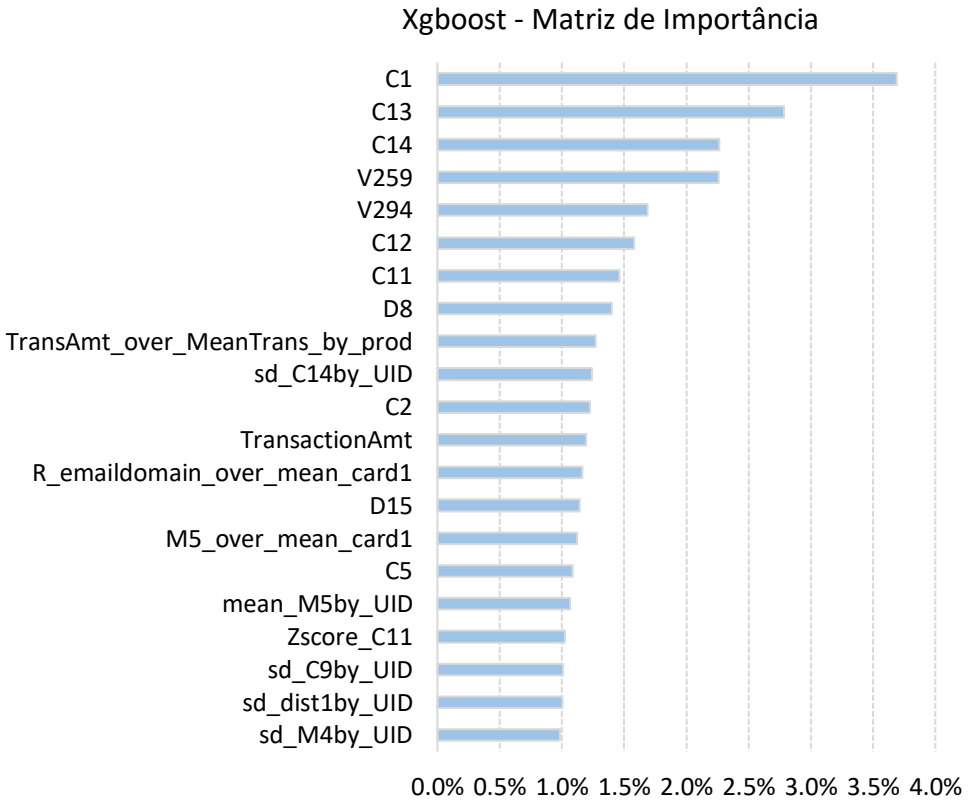


Figura 11 - Gráfico da matriz de importância (*Top 21 features*)

Uma avaliação complementar das técnicas de deteção de anomalia foi executada através da comparação de duas modelações com Xgboost. Um modelo foi construído com todas as 216 features, e outro modelo foi construído sem as features de deteção de anomalia. O modelo sem as técnicas de deteção de anomalias obteve AUC de 0.963196 e o modelo completo obteve AUC de 0.963815 contra os dados de validação, uma contribuição no valor 0.000619 pode não ser tão representativa numa utilização comercial, mas no âmbito de uma competição de *Machine Learning* esta margem de ganho possibilita a subida de muitas posições no *ranking*.

5.3 Desempenho individual dos modelos

A tabela 18 mostra em ordem decrescente o AUC obtido por cada modelo sobre os dados de validação.

Tabela 18 - AUC dos modelos sobre os dados de validação

Algoritmo	AUC Validação	AUC Kaggle
Xgboost	**0.963815**	**0.928315**
Random Forest	0.929429	0.917093
GBM	0.958435	0.917024
SVM	0.824726	0.860338
MLP	0.784159	0.816464
MLP + SMOTE	0.806435	0.847363

Os modelos Xgboost, Random Forest e GBM apresentaram desempenho bem superior na fase de validação, e com os dados de teste em relação ao SVM e MLP. O modelo em Xgboost superou os restantes, atingindo o AUC de 0.963815 no processo de validação e AUC de 0.928315 no Kaggle.

Cabe ressaltar que os modelos SVM e MLP, embora tenham conseguido AUC inferior, ao contrário dos demais modelos, o AUC obtido nos dados de teste foi superior ao AUC do processo de validação, o que aponta para uma maior capacidade de generalização e mais robustez ao lidar com o *dataset* de teste que apresenta uma distribuição diferente, principalmente nos dados omissos.

5.4 Desempenho dos modelos combinados (Ensemble learning)

A tabela 19 apresenta em ordem decrescente o AUC do processo de validação para as técnicas de combinação de modelos. Para o processo de combinação dos modelos foram utilizados apenas os modelos Xgboost, Random Forest e GBM que obtiveram AUC individualmente acima de 0.91.

Tabela 19 - AUC dos modelos combinados

Técnicas de combinação	AUC Validação	AUC Kaggle
Meta modelo (Xgboost)	**0.966400**	**0.931488**
Voto maioritário	0.964147	0.930824
Média	0.949108	0.926193

A técnica que utiliza um meta modelo para fazer a combinação das previsões obteve o AUC mais elevado. Foram feitos dois meta modelos, o primeiro feito com Regressão Linear obteve AUC de 0.9529 e o segundo com Xgboost atingiu AUC de 0.9664, sendo, portanto, ligeiramente superior ao melhor modelo individual que alcançou 0.963815. A submissão ao Kaggle com as previsões deste meta modelo obteve o melhor resultado da investigação com um **AUC de 0.931488**.

5.5 Resultado das submissões ao Kaggle

A avaliação final das equipas concorrentes na competição foi feita com o AUC calculado sobre 80% dos dados de teste, conjunto de dados chamado de *Private Leaderboard* [9], sendo que estes dados apresentam diferenças na distribuição dos valores omissos em relação aos dados de treino. Há um segundo conjunto dentro dos dados de teste que representa 20% do *dataset*, denominado *Public Leaderboard*, e estes dados apresentam distribuição mais próxima dos dados de treino. Esta divisão interna nos dados de teste foi provavelmente feita para validar a capacidade dos modelos em generalizar sobre dados que sofrem alterações ao longo do tempo.

[9] https://www.kaggle.com/c/ieee-fraud-detection/leaderboard

Após a avaliação dos modelos contra os dados de validação, a classificação foi feita sobre o *dataset* de teste e o resultado foi submetido no site da competição para o cálculo do AUC. Foram feitos diversos ciclos de envio das previsões sobre os dados de teste ao Kaggle, e os resultados alcançados serviram como uma bússola para nortear a afinação dos modelos até a obtenção do **AUC 0.909401** no *Private Leaderboard*. A tabela 20 mostra os cinco melhores resultados obtidos e a figura 12 apresenta um recorte da página do Kaggle com os resultados.

Tabela 20 - AUC das previsões submetidas ao Kaggle

Top 5 resultados	AUC Private Score
1	**0.909401**
2	0.908775
3	0.906355
4	0.906204
5	0.905235

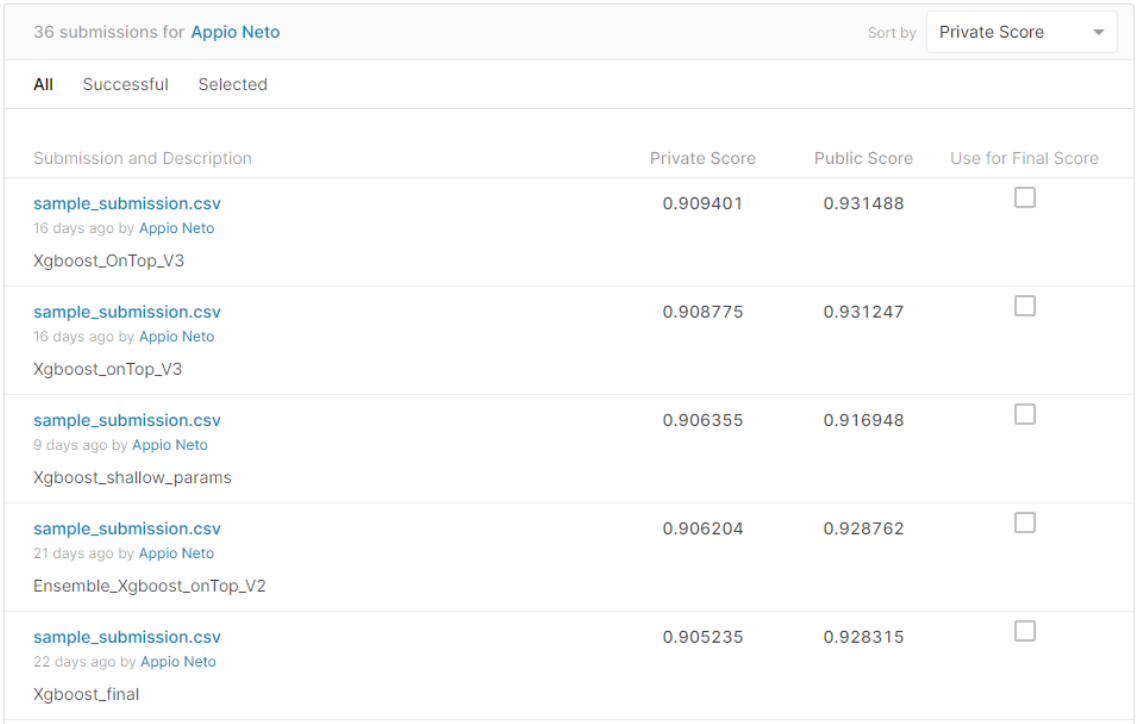
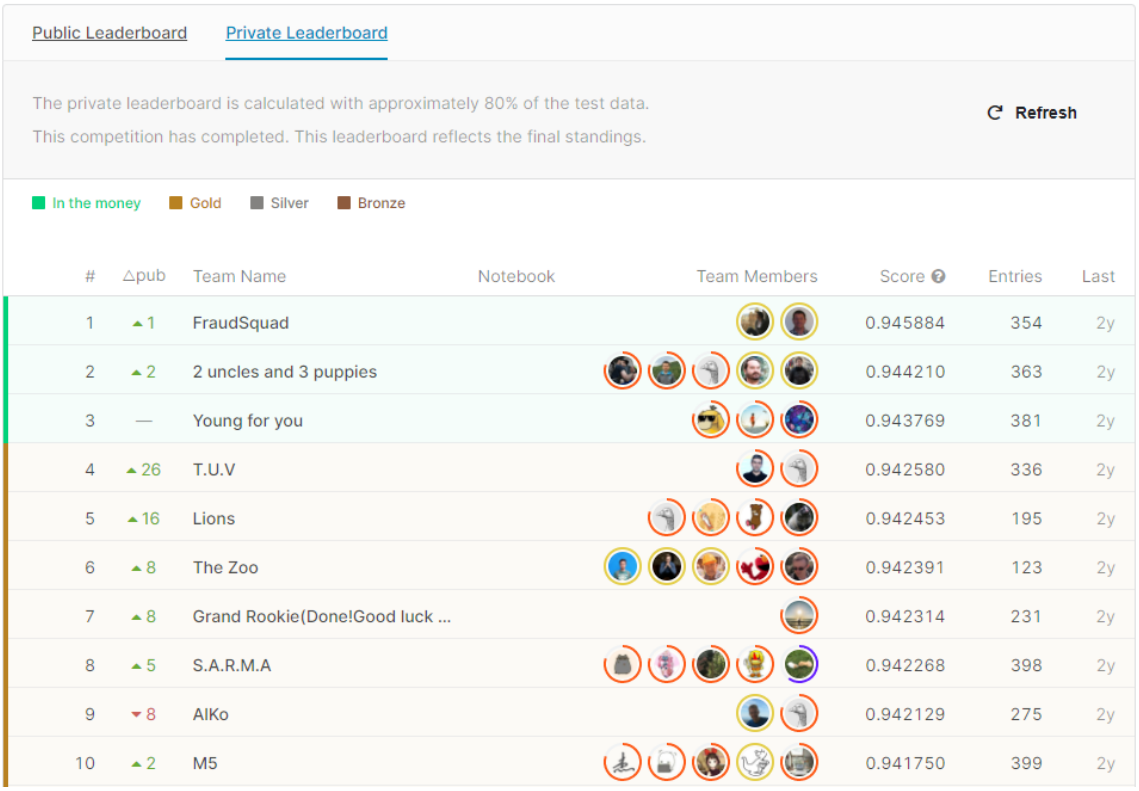


Figura 12 - Resultados no site do Kaggle [10]

[10] Imagem extraída do site: https://www.kaggle.com/c/ieee-fraud-detection/submissions

O AUC obtido pela equipa vencedora da competição com os dados do *Public Leaderboard* (20% dos dados de teste) foi 0.967722 e com os dados do *Private Leaderboard* (80% dos dados de teste) foi 0.945884. A figura 13 mostra os resultados das equipas nas dez primeiras posições do *ranking Private Leaderboard* da competição.



#	△pub	Team Name	Notebook	Team Members	Score	Entries	Last
1	▲ 1	FraudSquad			0.945884	354	2y
2	▲ 2	2 uncles and 3 puppies			0.944210	363	2y
3	—	Young for you			0.943769	381	2y
4	▲ 26	T.U.V			0.942580	336	2y
5	▲ 16	Lions			0.942453	195	2y
6	▲ 8	The Zoo			0.942391	123	2y
7	▲ 8	Grand Rookie(Done!Good luck ...			0.942314	231	2y
8	▲ 5	S.A.R.M.A			0.942268	398	2y
9	▼ 8	AIKo			0.942129	275	2y
10	▲ 2	M5			0.941750	399	2y

Figura 13 - Resultados dos 10 primeiros colocados na competição do Kaggle [11]

Um aspeto muito importante no âmbito de projetos de DM e ML é o processo de implementação da solução em produção. A tomada de decisão para implantar um modelo de deteção de fraude em tempo real, passa por avaliar a potencial perda financeira ao bloquear transações do tipo Falso Positivo versus o custo causado pelas fraudes não detetadas (Falso Negativo) pelo modelo. O Apêndice F discute o desempenho do modelo sob uma ótica de negócio, guiada tanto pelo impacto financeiro como também por alguma eventual fricção no relacionamento com o bom cliente, que poderá ter sua compra negada pelo processo de deteção de fraude.

[11] Imagem extraída do site: https://www.kaggle.com/c/ieee-fraud-detection/leaderboard

Capítulo 6 - Conclusões e Recomendações

A investigação teve como ponto de partida oferecer resposta a questão: quais estratégias podem ser usadas para a produção de features informativas com capacidade para enriquecer um modelo de *Machine Learning (ML)* para previsão de fraude, a partir do conjunto de dados escolhido para o estudo? A pergunta pode ser respondida sob algumas óticas diferentes que se fundamentam nos três objetivos da dissertação.

Sob o ponto de vista da utilização de técnicas de *data sampling* para promover o balanceamento das classes na fase de preparação dos dados, para os modelos baseados em estruturas de árvores de decisão, que foram aqueles que obtiveram os melhores resultados, conclui-se que as técnicas de *data sampling* não contribuíram para aumento do desempenho. O único algoritmo beneficiado com *data sampling* foi a Rede Neuronal Artificial, porém este modelo obteve o pior desempenho.

Sob a ótica do emprego de técnicas de deteção de anomalias e identificação de padrões de comportamento, a conclusão é que a criação de *features* construídas a partir de tais técnicas é fundamental para que o modelo aprenda os padrões da fraude mais eficientemente, pois estas features foram representam 43% das 21 features mais importantes para o modelo com melhor desempenho. A técnica de criação de *features* por medidas estatísticas de cada transação comparadas às medidas do agrupamento à qual a transação pertence, produziu imensas *features*, e o processo de *feature selection* teve grande importância para reduzir a dimensionalidade dos dados de forma a não comprometer a *performance* dos modelos.

O terceiro objetivo de investigação é perceber se técnicas de e*nsemble learning* poderiam superar os modelos individualmente. A técnica de *ensemble learning* desenvolvida a partir do aprendizado de um algoritmo sobre os scores dos três modelos com resultados mais competitivos, possibilitou um aumento considerável no AUC. A experiência de incluir os dos dois modelos com menor desempenho no processo de E*nsemble Learning*, introduziu ruído no processo de treinamento do meta modelo e por consequência, queda no valor do AUC.

Em relação à escolha do algoritmo, o Xgboost se provou extremamente eficiente ao superar os outros algoritmos testados, além de ter sido a melhor opção para a função de

meta modelo no processo de *Ensemble Learning*. Sublinha-se que os algoritmos da família das árvores de decisão alcançaram os melhores resultados neste estudo.

É importante destacar a importância da técnica *Adversarial Validation* utilizada para identificar as variáveis que apresentam mudanças na distribuição entre os dados de treino e teste, e por consequência diminuem a capacidade de generalização do modelo.

Quanto às limitações do estudo, a primeira limitação se refere à questão de que não é possível saber com exatidão o significado de muitas variáveis dos *datasets*. O significado das variáveis foi ocultado de forma proposital na competição para não comprometer a anonimidade da identidade dos clientes e empresas representados nos dados. O fato de não perceber com clareza o significado das variáveis limitou as análises e a aplicação de técnicas de *Data Mining (DM)*. Outra limitação importante se refere aos recursos computacionais disponíveis para o processamento dos dados e modelação. Algumas possibilidades de pré-processamento de dados e testes com mais algoritmos foram deixadas de lado devido ao alto custo computacional e longos tempos de execução. Muitas análises foram feitas sobre amostras de dados menores para otimizar o tempo de processamento, e essa redução no volume de dados pode causar ruído em determinados tipos de análises.

Como recomendação para futuros trabalhos em modelação para deteção de fraude de cartão de crédito, fica a indicação do aprofundamento na modelação com os algoritmos SVM e MLP + SMOTE, pois os resultados das previsões com os dados de teste indicaram robustez às mudanças na distribuição dos dados, sendo a mudança nas características da variável resposta ou nas variáveis independentes um desafio comum em *Machine Learning (ML)* dada a natureza dinâmica dos eventos registados pelos dados.

Referências Bibliográficas

Abdallah, A., Maarof, M., Zainal, A. (2016). Fraud Detection System: A Survey. *Journal of Network and Computer Applications*, 68. http://library1.nida.ac.th/termpaper6/sd/2554/19755.pdf

Abbott, D. (2014). *Applied predictive analytics: Principles and techniques for the professional data analyst*. John Wiley & Sons.

ACFE. (2018). *Global Study on Occupational Fraud and Abuse*. Association of Certified Fraud Examiners, 10, 80

Association of Certified Fraud Examiners (ACFE). (2020). *Report to the nations on occupational fraud and abuse: 2020 global fraud study*. Acfe, 88.

Ayman, A., Amaury, H., François, J., Liyun, H., Frédéric, O. (2020). Dual Sequential Variational Autoencoders for Fraud Detection. *In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 12080 LNCS*. https://doi.org/10.1007/978-3-030-44584-3_17

Awad, M., & Khanna, R. (2015). Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers. *Springer*, *4*(1), 64–75.

Baesens, B., Höppner, S., Ortner, I., & Verdonck, T. (2020). robrose: A robust approach for dealing with imbalanced data in fraud detection. ArXiv, 1–21.

Baesens, B., Vlasselaer, V., Verbeke, W. (2015). F*raud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection*. New Jersey: Wiley.

Bento, C., Cardoso, A., & Dias, G. (2005). Lecture Notes in Artificial Intelligence: Introduction. *In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 3808 LNCS*. https://doi.org/10.1007/11595014_1

Biegelman, M., Bartow, J. (2006). *Executive Roadmap to Fraud Prevention and Internal Control: Creating a Culture of Compliance*. Danvers: Wiley

Bolton, J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science*, *17*(3), 235–249.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Comput. Surv. 41, 3 Article 15, 14(1)*, 1–58. https://doi.org/10.1145/1541880.1541882

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C. & Wirth, R. (2000). *CRISP-DM 1.0 ± Step-by-step data mining guide, CRISP-DM Consortium*

Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-Sampling Technique. J*ournal of Artificial Intelligence Research* 16: 321–357.

Chen, R. C., Chiu, M. L., Huang, Y. L., & Chen, L. T. (2004). Detecting credit card fraud by using questionnaire-responded transaction model based on support vector machines. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3177, 800–806. https://doi.org/10.1007/978-3-540-28651-6_119

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/https://doi.org/10.1145/2939672.2939785

Cheng, D., Xiang, S., Shang, C., Zhang, Y., Yang, F., & Zhang, L. (2020). Spatio-Temporal Attention-Based Neural Network for Credit Card Fraud Detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(01), 362–369. https://doi.org/10.1609/aaai.v34i01.5371

Correa Bahnsen, A., Aouada, D., Stojanovic, A., & Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, 51, 134–142. https://doi.org/10.1016/j.eswa.2015.12.030

Dang, Y. (2017). *A Comparative Study of Bagging and Boosting of Supervised and Unsupervised Classifiers For Outliers Detection* (Doctoral dissertation, Wright State University).

Dey, A. (2016). Machine Learning Algorithms: A Review. *International Journal of Computer Science and Information Technologies*, *7*(3), 1174–1179. www.ijcsit.com

Dziuban, C. D., & Shirkey, E. C. (1974). When is a correlation matrix appropriate for factor analysis? Some decision rules. *Psychological Bulletin*, 81(6), 358–361. https://doi.org/10.1037/h0036316

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010

Fawcett, T., & Provost, F. (1997). *Adaptive fraud detection. Data Mining and Knowledge Discovery*, 1(3), 291–316. https://doi.org/10.1023/A:1009700419189

Gama, J., Carvalho, A. P., Lorena, A. C., Oliveira, M. (2017). *Extração de Conhecimento de Dados: Data Mining*. Lisboa: Edições Sílabo.

Ge, D., Gu, J., Chang, S., & Cai, J. H. (2020). Credit card fraud detection using lightgbm model. *Proceedings - 2020 International Conference on E-Commerce and Internet Technology*, ECIT 2020, 232–236. https://doi.org/10.1109/ECIT50008.2020.00060

Gee, S. (2015). *Fraud and Fraud Detection: A Data Analytics Approach*. Danvers: Wiley.

Grus, J. (2016). *Data Science do Zero: Primeiras Regras com Python*. Rio de Janeiro: Alta Books.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), 1263–1284. https://doi.org/10.1007/978-3-030-04663-7_4

Huang, K. (2020). An Optimized LightGBM Model for Fraud Detection. *Journal of Physics: Conference Series*, *1651*, 012111. https://doi.org/10.1088/1742-6596/1651/1/012111

Jing, R., Tian, H., Li, Y., Zhang, X., Zheng, X., Zhang, Z., Zeng, D., & Intelligence, S. A. (2019). *Improving the Data Quality for Credit Card Fraud Detection*.

Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P. E., He-Guelton, L., & Caelen, O. (2018). Sequence classification for credit-card fraud detection. *Expert Systems with Applications*, 100, 234–245. https://doi.org/10.1016/j.eswa.2018.01.037

Kabra, A., Chopra, A., Puri, N., Badjatiya, P., Verma, S., Gupta, P., & K, B. (2020). MixBoost: *Synthetic Oversampling with Boosted Mixup for Handling Extreme Imbalance. Section II*. http://arxiv.org/abs/2009.01571

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 2017-December(Nips), 3147–3155.

Kuhn, M. (2008). Building predictive models in R using the caret package. *Journal of Statistical Software*, 28(5), 1–26. https://doi.org/10.18637/jss.v028.i05

Lei, S., Xu, K., Huang, Y., & Sha, X. (2020). *An Xgboost based system for financial fraud detection*. *02042*, 1–4.

Ling, C., Huang, J., Zang, H. (2020). AUC: A Better Measure than Accuracy in Comparing Learning Algorithms. *Proceedings - 16th Conference of the Canadian Society for Computacional Studies of Intelligence*, AI 2003, 329–341.

Livingston, F. (2005). Implementation of Breiman's Random Forest Machine Learning Algorithm. *Machine Learning Journal Paper*, 1–13.

Lucas, Y., Portier, P. E., Laporte, L., He-Guelton, L., Caelen, O., Granitzer, M., & Calabretto, S. (2020). Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs. *Future Generation Computer Systems*, *102*, 393–402. https://doi.org/10.1016/j.future.2019.08.029

Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2018). *Consistent Individualized Feature Attribution for Tree Ensembles*. 2. http://arxiv.org/abs/1802.03888

Patidar, R., & Sharma, L. (2011). Credit Card Fraud Detection Using Neural Network. *International Journal of Soft Computing and Engineering*, 1, 32–38.

Priscilla, C. V., & Prabha, D. P. (2020). Influence of optimizing xgboost to handle class imbalance in credit card fraud detection. *Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology, ICSSIT 2020, Icssit*, 1309–1315. https://doi.org/10.1109/ICSSIT48917.2020.9214206

Sailusha, R., Gnaneswar, V., Ramesh, R., & Ramakoteswara Rao, G. (2020). Credit Card Fraud Detection Using Machine Learning. *Proceedings of the International Conference on Intelligent Computing and Control Systems, ICICCS 2020, Iciccs*, 1264–1270. https://doi.org/10.1109/ICICCS48265.2020.9121114

Schafer, J. L., & Graham, J. W. (2002). Missing data: Our view of the state of the art. *Psychological Methods*, 7(2), 147–177. https://doi.org/10.1037/1082-989X.7.2.147

Singh, A., Thakur, N., & Sharma, A. (2016). A review of supervised machine learning algorithms. *Proceedings of the 10th INDIACom; 2016 3rd International Conference on Computing for Sustainable Global Development, INDIACom 2016*, 1310–1315.

Song, Z. (2020). A Data Mining Based Fraud Detection Hybrid Algorithm in E-bank. *Proceedings - 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering, ICBAIE 2020*, 44–47. https://doi.org/10.1109/ICBAIE49996.2020.00016

Van Vlasselaer, V., Bravo, C., Caelen, O., Eliassi-Rad, T., Akoglu, L., Snoeck, M., & Baesens, B. (2015). APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions. *Decision Support Systems*, 75, 38–48. https://doi.org/10.1016/j.dss.2015.04.013

Vluymans, S. (2009). Learning from imbalanced data. *Studies in Computational Intelligence*, *807*(9), 81–110. https://doi.org/10.1007/978-3-030-04663-7_4

Wells, J. T. (2007). Corporate Fraud Handbook: Prevention and Detection (2ª ed.). Danvers: Wiley

Zhang, X., Han, Y., Xu, W., & Wang, Q. (2019). HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture. *Information Sciences*, *xxxx*. https://doi.org/10.1016/j.ins.2019.05.023

Zhang, Y., Tong, J., Wang, Z., & Gao, F. (2020). Customer Transaction Fraud Detection Using Xgboost Model. *Proceedings - 2020 International Conference on Computer Engineering and Application, ICCEA 2020*, 554–558. https://doi.org/10.1109/ICCEA50009.2020.00122

Anexos e Apêndices

Apêndice A

Este apêndice faz um breve resumo sobre os tipos e modalidades de fraude. As práticas de fraude podem ser segmentadas em categorias, e Baesens (2015) cita uma lista não exaustiva que é composta por:

- Fraude de cartão de crédito
- Fraude de seguros
- Corrupção
- Falsificação
- Fraude em garantia de produtos
- Fraude de seguro de saúde
- Fraude de telecomunicações
- Lavagem de dinheiro
- *Click fraud*
- Roubo de identidade
- Evasão fiscal
- Plágio

Há uma variedade mais ampla de tipos de fraude, que acompanha o desenvolvimento tecnológico das empresas e organizações, de modo que, a cada atualização de sistema, criação de novos produtos e serviços, e surgimento de inovações tecnológicas, os fraudadores estudam as vulnerabilidades dos processos e das tecnologias para tirar proveito. As fraudes, portanto, tornam-se mais sofisticadas e difíceis de se detetar conforme os fraudadores se especializam nos mercados que são os alvos de seus ataques.

A figura 14 mostra uma outra lista, também não exaustiva, com as áreas de fraude e suas subdivisões apresentada por Abdallah (2016).

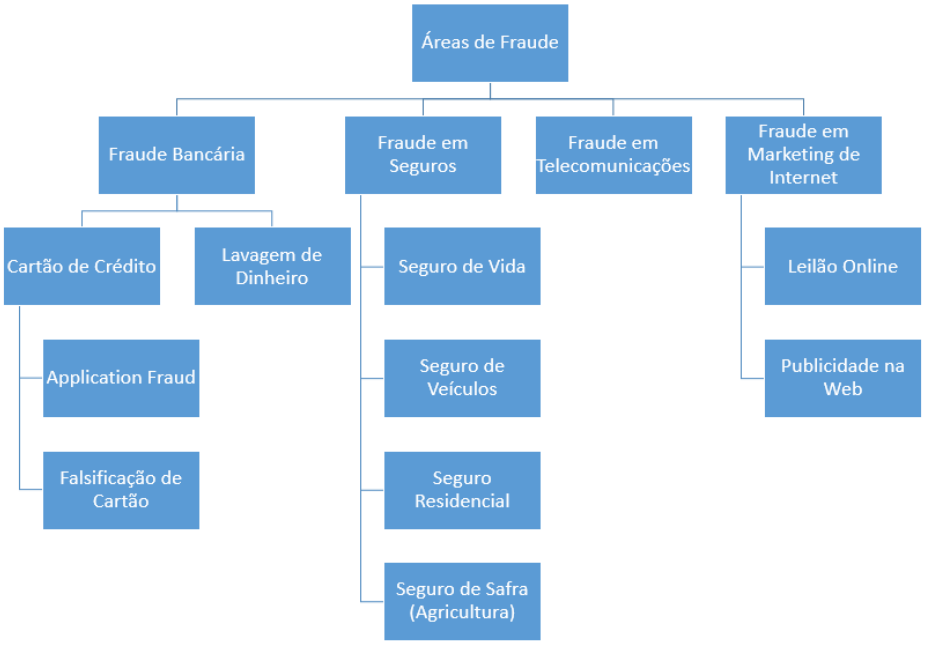


Figura 14 - Taxonomia das áreas de Fraude mais comuns, Abdallah (2016)

Apêndice B

Este apêndice tem por finalidade apresentar os fundamentos do *Machine Learning* (*ML)*. O ML é um ramo da Inteligência Artificial que utiliza algoritmos para identificar camadas de relacionamentos entre os dados. A característica computacional do ML é generalizar a experiência adquirida no treinamento com dados de exemplo, gerando uma saída de dados que estima uma função alvo. A capacidade de generalização é o que permite ao sistema fazer previsões de forma acurada quando exposto a dados ainda não vistos antes. O potencial do ML tem sido utilizado em diversos segmentos tais como: reconhecimento de voz, visão computacional, pesquisa na *web*, previsão do tempo, deteção de fraudes, análise de comportamento, sequenciamento genético e muitas outras aplicações (Awad & Khanna, 2015).

Grus (2016) afirma que ML é a criação e o uso de modelos que são aprendidos a partir dos dados, o que também é chamado de modelo preditivo ou mineração de dados.

Há três divisões mais conhecidas do ML que são: aprendizagem supervisionada, aprendizagem não-supervisionada e aprendizagem semi-supervisionada.

A aprendizagem supervisionada compreende um conjunto de técnicas que requerem rótulos nos dados para o treinamento do algoritmo classificador. O algoritmo identifica as associações entre as variáveis independentes e uma variável designada como dependente (o rótulo). O método supervisionado tem uma vantagem grande em que a saída de dados com a classificação feita pelo algoritmo é compreensível para o ser humano. Há, porém, limitações como: a dificuldade de aplicar rótulos em grandes volumes de dados, e também, em alguns casos é muito difícil superar certas ambiguidades e incertezas para rotular corretamente os dados de treino.

Na aprendizagem não-supervisionada, não há rótulos nos dados para a construção do modelo. O aprendizado é feito pela identificação de padrões estruturados nos dados e rejeição de ruído não estruturado. A principal vantagem dessa abordagem é que não depende da identificação de rótulos.

A aprendizagem semi-supervisionada é o meio termo entre aprendizagem supervisionada e não supervisionada. Neste método um pequeno número de amostras é rotulado e um grande número permanece não rotulado. O algoritmo classificador é treinado com os dados rotulados e não rotulados e a saída de dados apresenta a vantagem de ser interpretável pelo ser humano (Abdallah et al., 2016).

Para além dos métodos mencionados acima, existe uma variedade ainda maior de métodos de aprendizagem e algoritmos que podem ser aplicados dentro de cada método. Dey (2016) apresenta uma lista mais ampla, porém não completa, com as abordagens de ML que pode ser vista na figura 15.

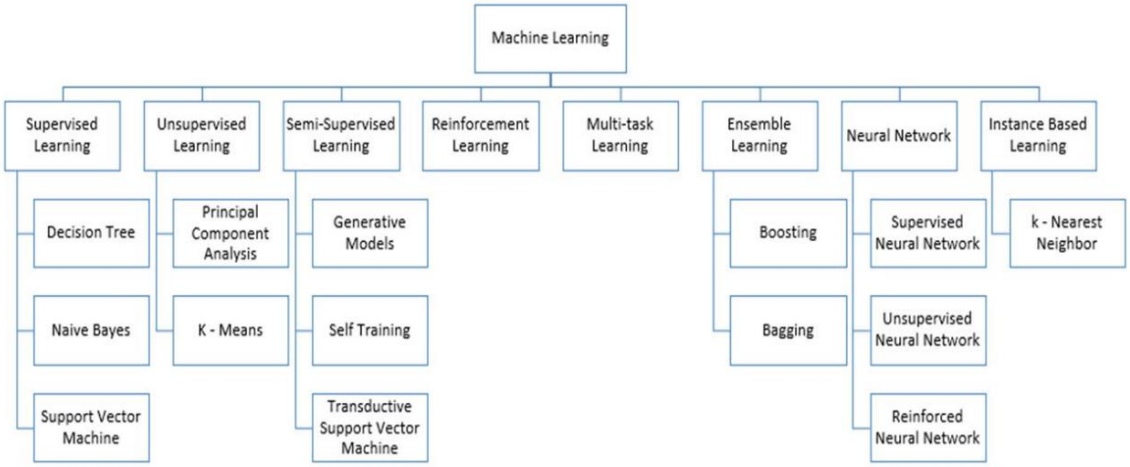


Figura 15 - Tipos de aprendizagem

O desenvolvimento de modelos de ML passa pela identificação de quais métricas são fundamentais para tomar a decisão sobre como aplicar o resultado alcançado pelo modelo. De acordo com cada tipo de problema deve-se avaliar qual modelo de ML é mais adequado, pois as técnicas de ML não produzem uma acurácia perfeita, e caso o problema exija que a acurácia seja perfeita, ML não é a solução mais indicada. A utilização de ML é em geral favorecida pelas seguintes condições: não é esperado que o modelo atinja uma acurácia altíssima, que haja imensos volumes de dados com padrões desconhecidos, que o problema não seja muito bem compreendido e exista lacunas de conhecimento, e a solução precisa se adaptar a mudanças nas características do problema (Awad & Khanna, 2015).

O processo de desenvolvimento de ML é composto por várias etapas que podem variar de acordo com a natureza do problema a ser modelado, a qualidade dos dados, os tipos e formatos dos dados e as características específicas do algoritmo utilizado. A levar em conta as especificidades da modelação de cada problema, a lista apresentada por Singh et al., (2016) descreve um exemplo com as etapas do desenvolvimento de ML para um problema de classificação. As etapas são:

- **coleta de dados:** realiza-se nesta fase a definição de quais dados farão parte da modelação, define-se a origem e faz-se a extração dos dados.

- **pré-processamento dos dados:** identificação de dados omissos e da presença de *outliers*, com a realização do devido tratamento para corrigir esses cenários, se desejado.
- ***Feature Engineering*:** esse processo pode ser subdividido em duas partes. A primeira parte é *Feature Extraction* ou *Feature Transformation* que tem por objetivo construir novas *features* a partir dos dados originais. A segunda parte é a *Feature Selection*, utilizada para identificar e remover o máximo de variáveis redundantes ou irrelevantes.
- **modelação:** criação de modelos com diferentes algoritmos para comparação de resultados.
- **validação dos algoritmos:** a acurácia dos modelos é verificada com conjuntos de dados ainda não vistos pelos modelos. Uma técnica amplamente utilizada para a validação do modelo é o método *K-fold cross validation*, que divide o *dataset* de treino em K subconjuntos de tamanho aproximadamente igual. Por exemplo, se for definido a divisão dos dados em dez partições, nove partições são utilizadas para treino e uma é utilizada para validação, e o processo é feito de forma iterativa em um ciclo de dez repetições alternando-se a partição de validação e o modelo é finalmente validado pela média das dez validações.

O processo descrito por Singh et al., (2016) é representado visualmente pela figura 16.

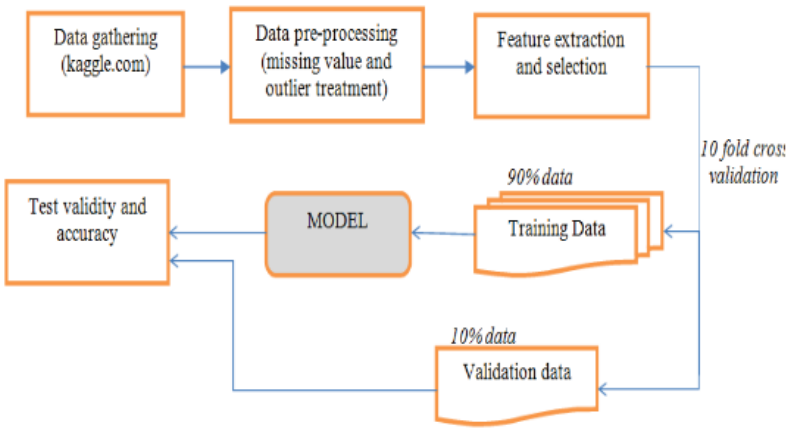


Figura 16 - Etapas da modelação de um problema de classificação

Apêndice C

Este Apêndice tem por finalidade ser um complemento à secção 2.3 ao apresentar técnicas de *sampling* de dados mais focadas no problema da fraude de cartão de crédito. A técnica chamada MixBoost desenvolvida por Kabra et al., (2020), combina instâncias da classe majoritária com a classe minoritária para criar instâncias sintéticas híbridas. Cada instância sintética híbrida contém elementos das duas classes (Kabra et al., 2020).

Para os casos em que os dados apresentam transações realizadas em sequência e elevado desbalanceamento, Ayman et al., (2020) propõe a técnica denominada DuSVAE - *Dual Sequential Variational Autoencoders*, que consiste na combinação de dois *autoencoders variacionais*. O primeiro é treinado a partir de sequências fraudulentas de transações para atribuir uma pontuação de fraude a cada sequência, o segundo VAE é treinado com uma abordagem de aprendizagem negativa, com o objetivo de maximizar o erro de reconstrução das sequências fraudulentas e minimizar o erro de reconstrução das genuínas (Ayman et al., 2020).

O artigo de Baesens et al., (2020) apresenta uma versão da técnica ROSE chamada RobROSE, que lida simultaneamente com dados desbalanceados e *outliers*, com o objetivo de identificar a fraude e ignorar as anomalias nos dados. O trabalho destaca a diferença entre as técnicas de *data sampling SMOTE, Borderline-SMOTE, ADASYN, MWMOTE e ROSE*. Sendo que a diferença entre elas é a forma pelo qual realizam o *oversampling*, que pode ser resumida em uma das três alternativas abaixo:

1. Quais as amostras minoritárias sofrerão *oversampling*? Todas as minoritárias ou há anomalias dentro das minoritárias que serão excluídas do processo?
2. Quanto cada classe minoritária será ampliada? Há classes minoritárias que devem ser aumentadas mais do que outras?
3. Como serão criados os casos minoritários sintéticos?

O artigo introduz a técnica designada por *robROSE* que utiliza casos minoritários que não são *outliers* para fazer o *oversampling* (Baesens et al., 2020).

O trabalho de Van Vlasselaer et al., (2015) apresenta a técnica designada *APATE* (*Anomaly Prevension using Advanced Transaction Exploration*), que aplica *Feature Extraction* para combinar atributos do *dataset* com *network-based attributes* para enriquecer os modelos de deteção com variáveis de *network*.

Outra abordagem dentro do campo do tratamento dos dados e geração de *features* encontra-se no artigo de Lucas et al., (2020) que apresenta um estudo sobre automação de *feature engineering* para deteção de fraude de cartão de crédito pela aplicação de *Multiperspective HMMs*. O trabalho conclui que com a utilização da técnica de *Feature Engineering* proposta, o desempenho para deteção de fraude de cartão de crédito aumentou em 18,1% na métrica AUC para transações presenciais e 9,3% para fraude do tipo CNP. Essa técnica foi considerada relevante para o desempenho de algoritmos como Random Forest, Logistic Regression e Adaboost.

Apêndice D

O Apêndice D apresenta as figuras com a matriz de correlação das variáveis dos *datasets* com os dados transacionais e de identidade.

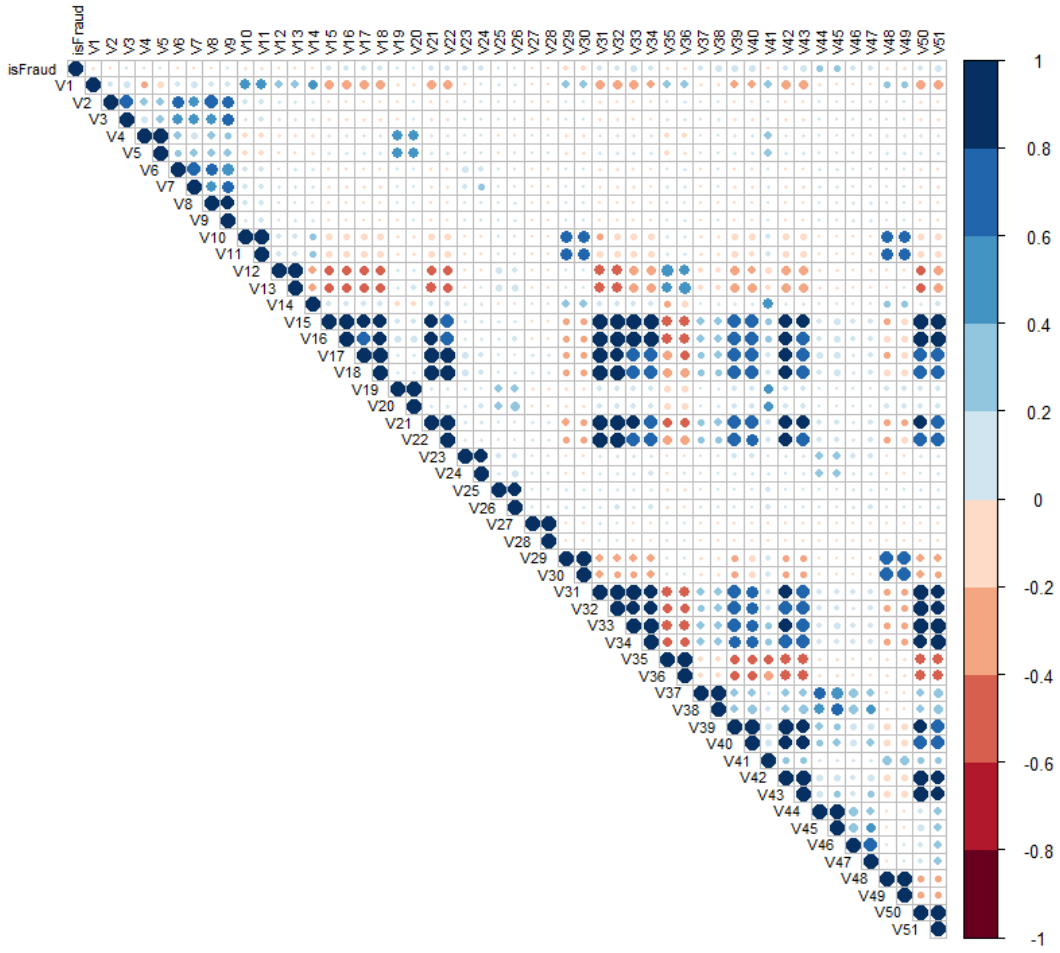


Figura 17 - Gráfico de correlação das colunas V1 até V51

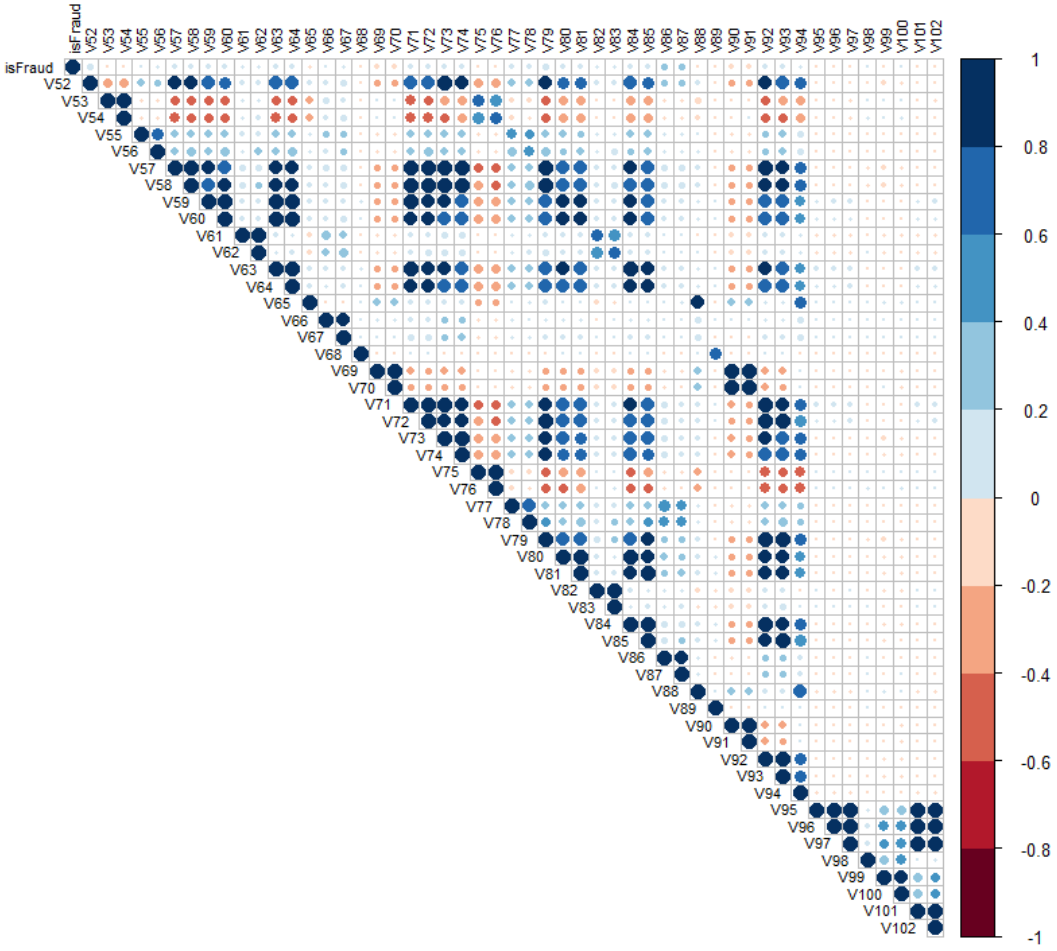


Figura 18 - Gráfico de correlação das colunas V52 até V102

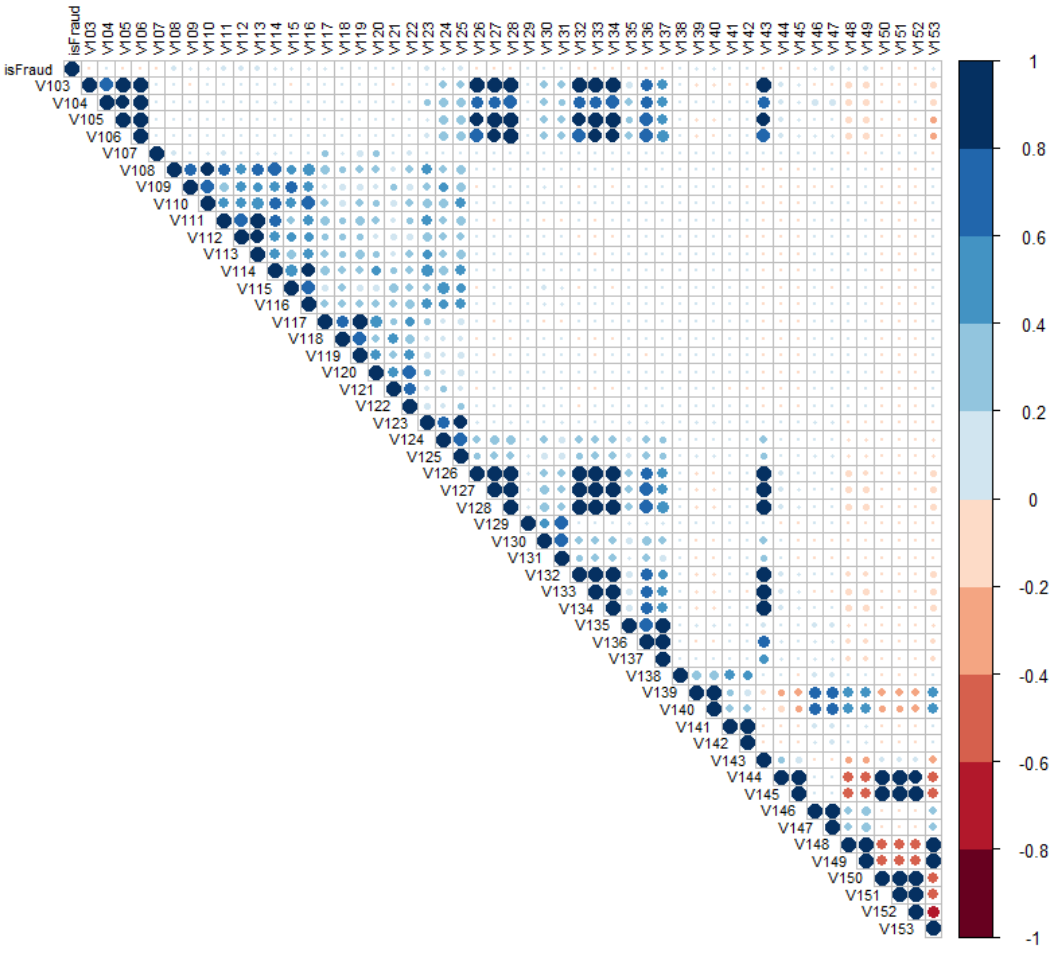


Figura 19 - Gráfico de correlação das colunas V103 até V153

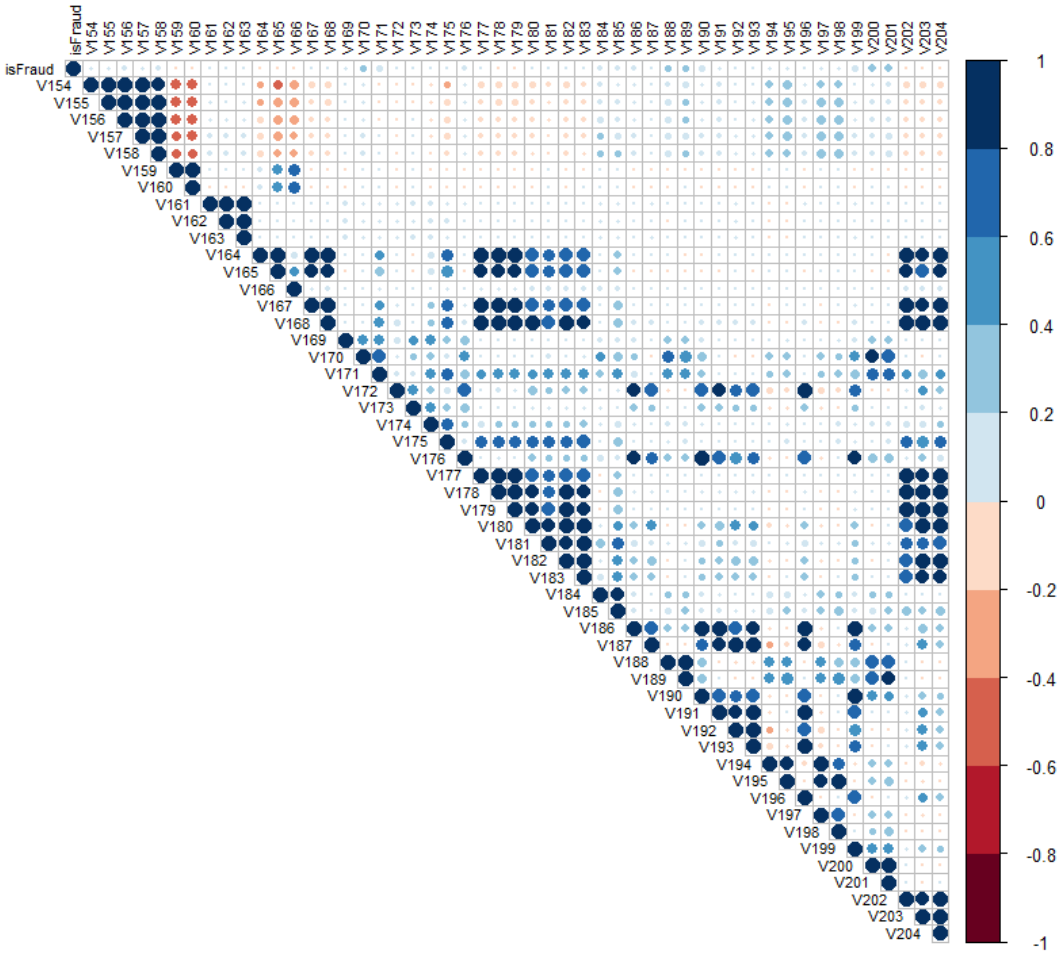


Figura 20 - Gráfico de correlação das colunas V154 até V204

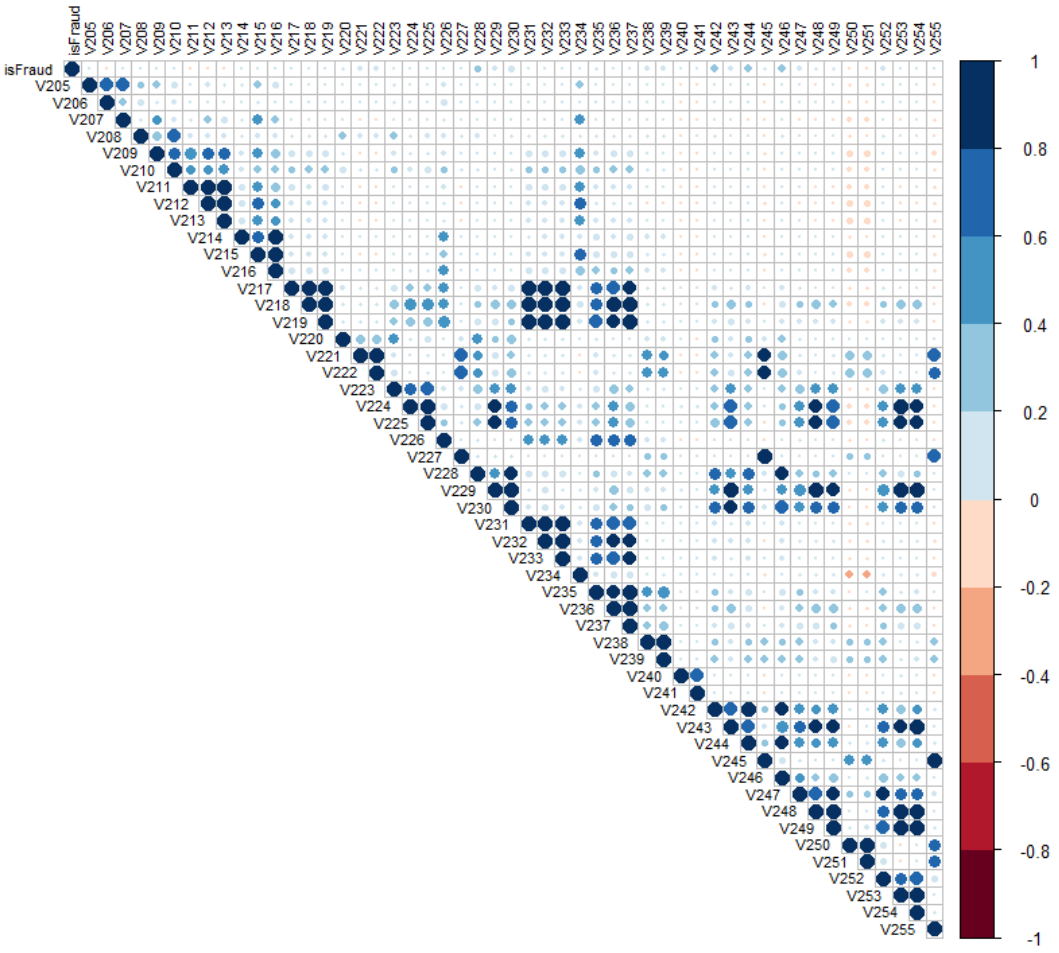


Figura 21 - Gráfico de correlação das colunas V205 até V255

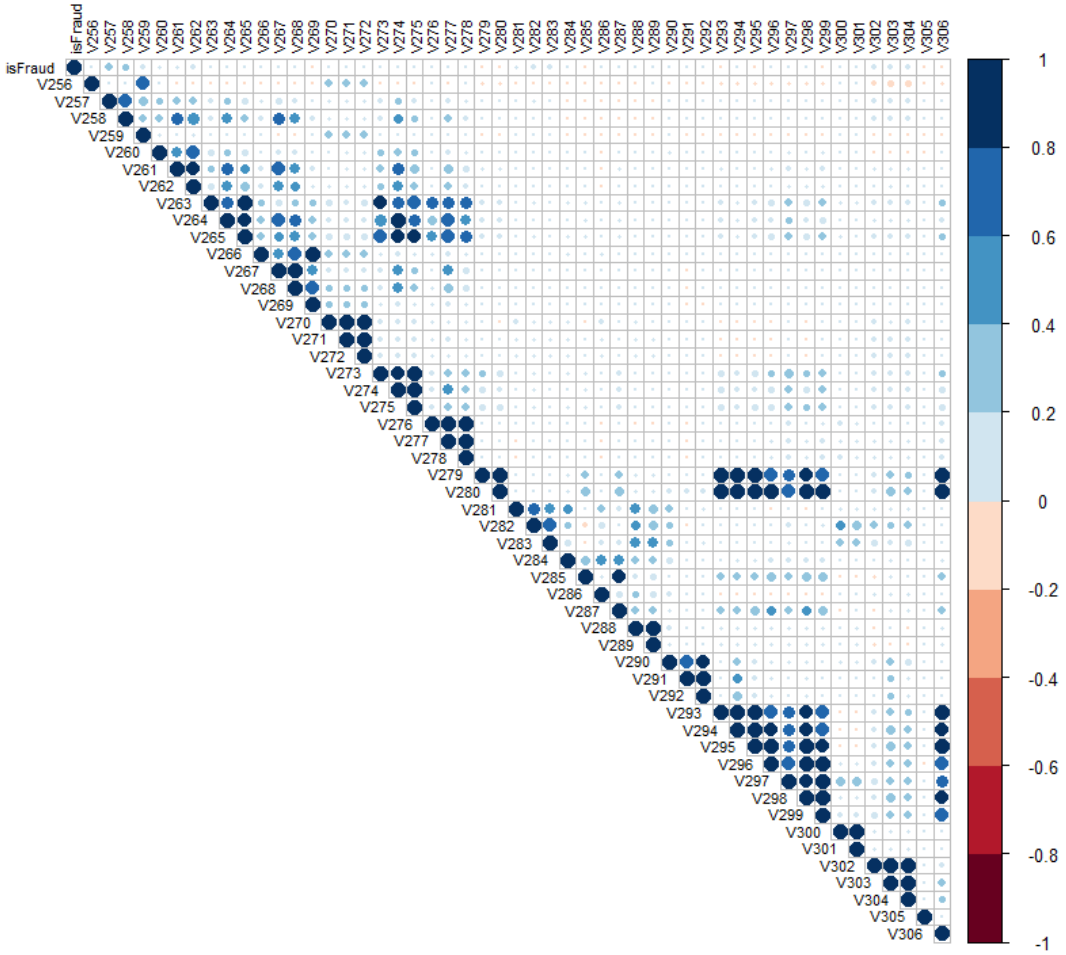


Figura 22 - Gráfico de correlação das colunas V256 até V306

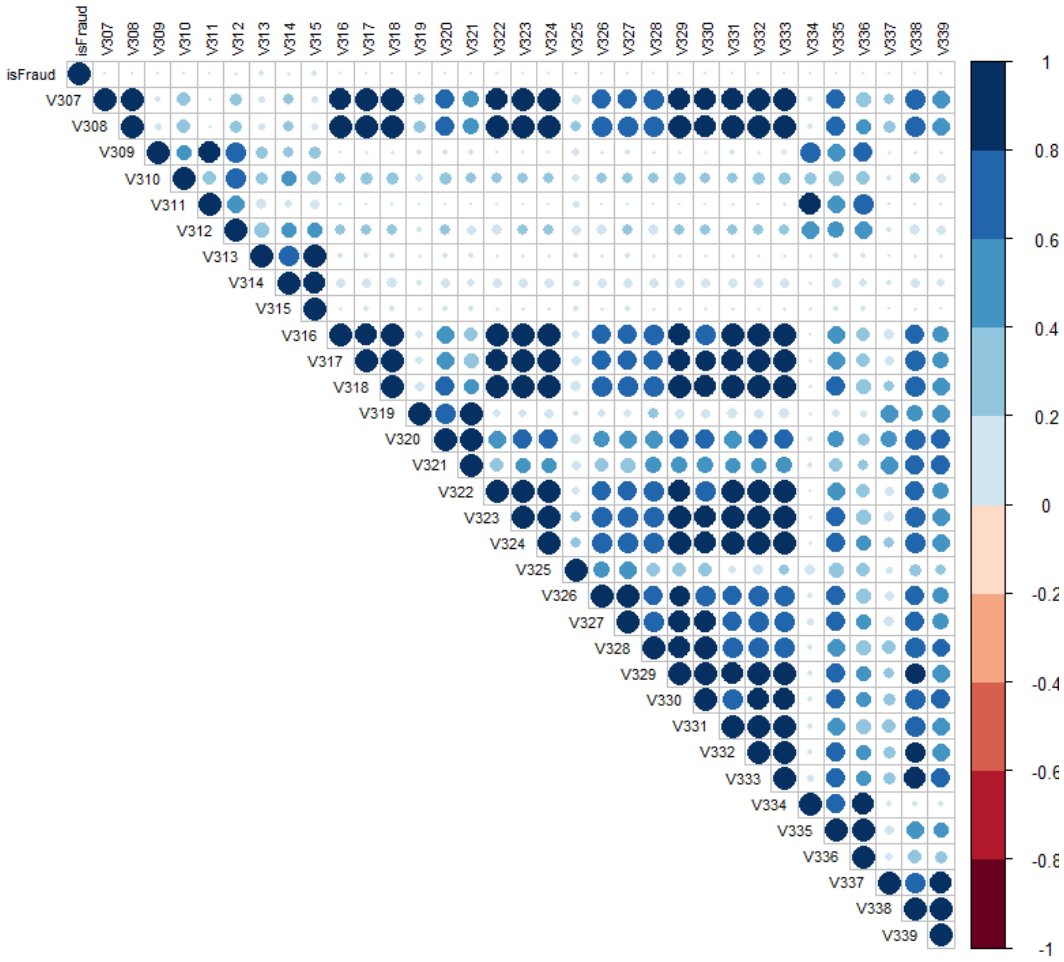


Figura 23 - Gráfico de correlação das colunas V307 até V339

Apêndice E

Este apêndice apresenta as tabelas que mostram o percentual de valores omissos para as variáveis V do *dataset* de transações e do *dataset identity*.

Tabela 21 - Valores omissos da variável V1 até V172 do dataset de transações

Variável	NA	Variável	NA	Variável	NA	Variável	NA
V1	42%	V44	22%	V87	9%	V130	0%
V2	42%	V45	22%	V88	9%	V131	0%
V3	42%	V46	22%	V89	9%	V132	0%
V4	42%	V47	22%	V90	9%	V133	0%
V5	42%	V48	22%	V91	9%	V134	0%
V6	42%	V49	22%	V92	9%	V135	0%
V7	42%	V50	22%	V93	9%	V136	0%
V8	42%	V51	22%	V94	9%	V137	0%
V9	42%	V52	22%	V95	0%	V138	86%
V10	42%	V53	8%	V96	0%	V139	86%
V11	42%	V54	8%	V97	0%	V140	86%
V12	8%	V55	8%	V98	0%	V141	86%
V13	8%	V56	8%	V99	0%	V142	86%
V14	8%	V57	8%	V100	0%	V143	86%
V15	8%	V58	8%	V101	0%	V144	86%
V16	8%	V59	8%	V102	0%	V145	86%
V17	8%	V60	8%	V103	0%	V146	86%
V18	8%	V61	8%	V104	0%	V147	86%
V19	8%	V62	8%	V105	0%	V148	86%
V20	8%	V63	8%	V106	0%	V149	86%
V21	8%	V64	8%	V107	0%	V150	86%
V22	8%	V65	8%	V108	0%	V151	86%
V23	8%	V66	8%	V109	0%	V152	86%
V24	8%	V67	8%	V110	0%	V153	86%
V25	8%	V68	8%	V111	0%	V154	86%
V26	8%	V69	8%	V112	0%	V155	86%
V27	8%	V70	8%	V113	0%	V156	86%
V28	8%	V71	8%	V114	0%	V157	86%
V29	8%	V72	8%	V115	0%	V158	86%
V30	8%	V73	8%	V116	0%	V159	86%
V31	8%	V74	8%	V117	0%	V160	86%
V32	8%	V75	9%	V118	0%	V161	86%
V33	8%	V76	9%	V119	0%	V162	86%
V34	8%	V77	9%	V120	0%	V163	86%
V35	22%	V78	9%	V121	0%	V164	86%
V36	22%	V79	9%	V122	0%	V165	86%
V37	22%	V80	9%	V123	0%	V166	86%
V38	22%	V81	9%	V124	0%	V167	75%
V39	22%	V82	9%	V125	0%	V168	75%
V40	22%	V83	9%	V126	0%	V169	75%
V41	22%	V84	9%	V127	0%	V170	75%
V42	22%	V85	9%	V128	0%	V171	75%
V43	22%	V86	9%	V129	0%	V172	75%

Tabela 22 - Valores omissos da variável V173 até V339 do dataset de transações

Variável	NA	Variável	NA	Variável	NA	Variável	NA
V173	75%	V216	75%	V259	75%	V302	0%
V174	75%	V217	77%	V260	77%	V303	0%
V175	75%	V218	77%	V261	77%	V304	0%
V176	75%	V219	77%	V262	77%	V305	0%
V177	75%	V220	75%	V263	77%	V306	0%
V178	75%	V221	75%	V264	77%	V307	0%
V179	75%	V222	75%	V265	77%	V308	0%
V180	75%	V223	77%	V266	77%	V309	0%
V181	75%	V224	77%	V267	77%	V310	0%
V182	75%	V225	77%	V268	77%	V311	0%
V183	75%	V226	77%	V269	77%	V312	0%
V184	75%	V227	75%	V270	75%	V313	1%
V185	75%	V228	77%	V271	75%	V314	1%
V186	75%	V229	77%	V272	75%	V315	1%
V187	75%	V230	77%	V273	77%	V316	0%
V188	75%	V231	77%	V274	77%	V317	0%
V189	75%	V232	77%	V275	77%	V318	0%
V190	75%	V233	77%	V276	77%	V319	0%
V191	75%	V234	75%	V277	77%	V320	0%
V192	75%	V235	77%	V278	77%	V321	0%
V193	75%	V236	77%	V279	0%	V322	86%
V194	75%	V237	77%	V280	0%	V323	86%
V195	75%	V238	75%	V281	1%	V324	86%
V196	75%	V239	75%	V282	1%	V325	86%
V197	75%	V240	77%	V283	1%	V326	86%
V198	75%	V241	77%	V284	0%	V327	86%
V199	75%	V242	77%	V285	0%	V328	86%
V200	75%	V243	77%	V286	0%	V329	86%
V201	75%	V244	77%	V287	0%	V330	86%
V202	75%	V245	75%	V288	1%	V331	86%
V203	75%	V246	77%	V289	1%	V332	86%
V204	75%	V247	77%	V290	0%	V333	86%
V205	75%	V248	77%	V291	0%	V334	86%
V206	75%	V249	77%	V292	0%	V335	86%
V207	75%	V250	75%	V293	0%	V336	86%
V208	75%	V251	75%	V294	0%	V337	86%
V209	75%	V252	77%	V295	0%	V338	86%
V210	75%	V253	77%	V296	1%	V339	86%
V211	75%	V254	77%	V297	0%		
V212	75%	V255	75%	V298	0%		
V213	75%	V256	75%	V299	0%		
V214	75%	V257	77%	V300	1%		
V215	75%	V258	77%	V301	1%		

Tabela 23 - Valores omissos do identity dataset

Variável	NA	Variável	NA
TransactionID	0%	id_21	96%
id_01	0%	id_22	96%
id_02	3%	id_23	96%
id_03	54%	id_24	97%
id_04	54%	id_25	96%
id_05	5%	id_26	96%
id_06	5%	id_27	96%
id_07	96%	id_28	3%
id_08	96%	id_29	3%
id_09	48%	id_30	48%
id_10	48%	id_31	3%
id_11	3%	id_32	48%
id_12	0%	id_33	50%
id_13	10%	id_34	48%
id_14	47%	id_35	3%
id_15	3%	id_36	3%
id_16	11%	id_37	3%
id_17	4%	id_38	3%
id_18	66%	DeviceType	3%
id_19	4%	DeviceInfo	18%
id_20	4%		

Apêndice F

A implementação de um modelo de deteção de fraude em ambiente produtivo, depende do resultado da avaliação da acurácia em prever as fraudes, e dos impactos que pode causar ao negócio, do ponto de vista financeiro e do relacionamento com os clientes.

Em um produto de *Machine Learning (ML)*, não se pode esperar uma acurácia de 100%, pois este tipo de solução sempre apresenta margem de erro. Como demonstrado na secção 5.3, o melhor modelo alcançou um AUC de 0.928315 quando submetido ao Kaggle, sendo um excelente desempenho ao utilizar como *benchmark* os resultados da própria competição no Kaggle.

Uma questão importante na avaliação de uma possível utilização do modelo em produção e em tempo real é que, deve-se analisar o *score* produzido pelo modelo versus a quantidade de bons clientes e fraudadores acumulados por faixas de *score*, para definir um ponto ótimo de corte (*threshold*), a fim de maximizar o ganho ao bloquear mais fraudes do que bloquear bons clientes.

Portanto, no momento em que uma transação de compra *online* está na fase de submeter os dados do cartão para realizar o pagamento, os dados da transação serão processados pelo modelo para realizar-se o cálculo da probabilidade de fraude, e uma decisão deverá ser tomada automaticamente para permitir a conclusão da compra, caso o *score* esteja abaixo do *threshold*, ou a compra será bloqueada se o *score* estiver acima do *threshold*.

Uma proposta de utilização do modelo desenvolvido nesta dissertação seria definir, por exemplo, três faixas de risco de acordo com o *score*. Assim, transações classificadas na faixa de alto risco seriam bloqueadas automaticamente em tempo real. As transações com *score* na faixa intermediária de risco passam por uma análise humana, o que pressupõe o emprego de uma equipa treinada nas técnicas e processos de análise de fraude, para decidir pelo bloqueio ou não da transação e cancelar a faturação da compra para o cliente. E por fim, a faixa de baixo risco permite que a transação seja concluída sem qualquer intervenção. Uma solução como esta deve ter o compromisso de obedecer *SLA's (Service Level Agreement)* que não causem o aumento no tempo de resposta da transação, principalmente as transações que, por serem de médio risco serão analisadas manualmente.

Um ponto de atenção importante é que o modelo deve periodicamente ser reavaliado para que não perca aderência às características dos dados, que mudam dinamicamente ao longo do tempo. A monitorização constante do volume de transações acumulado pelas faixas de score é fundamental, pois se numa determinada hora ocorre um crescimento ou queda brusca no volume de transações de alto risco, deve-se analisar as causas pois pode estar a acontecer um ataque de fraudadores, alguma mudança no ambiente de produção ou nos dados pode ter afetado o desempenho do modelo, ou devido a eventos comerciais, como descontos nos preços, poderá ocorrer o aumento do fluxo de vendas e por consequência alteram-se os volumes de dados processados pelo modelo.