Lemonier Lima

O uso de técnicas de *Machine Learning* para melhorar a prevenção à fraude

Brasil

2022, v-0.0.1

Lemonier Lima

O uso de técnicas de *Machine Learning* para melhorar a prevenção à fraude

Projeto de Pesquisa apresentado ao Curso de Mestrado Acadêmico em Economia, Universidade de Brasília, como requisito parcial para a obtenção do título de Mestre em Economia

Universidade de Brasília - UnB

Faculdade de Administração Contabilidade e Economia - FACE

Departamento de Economia - ECO

Programa de Pós-Graduação

Orientador: Daniel O. Cajueiro

Brasil

2022, v-0.0.1

Lemonier Lima
O uso de técnicas de *Machine Learning* para melhorar a prevenção à fraude / Lemonier Lima. – Brasil, 2022, v-0.0.1-
48p. : il. (algumas color.) ; 30 cm.

Orientador: Daniel O. Cajueiro

Dissertação (Mestrado) – Universidade de Brasília - UnB
Faculdade de Administração Contabilidade e Economia - FACE
Departamento de Economia - ECO
Programa de Pós-Graduação, 2022, v-0.0.1.

1. Palavra-chave1. 2. Palavra-chave2. 3. Palavra-chave3. II. Universidade de Brasília. III. Faculdade de Administração, Contabilidade e Economia - FACE. IV. Departamento de Economia IV. O uso de técnicas de *Machine Learning* para melhorar a prevenção à fraude

Lemonier Lima

O uso de técnicas de *Machine Learning* para melhorar a prevenção à fraude

Projeto de Pesquisa apresentado ao Curso de Mestrado Acadêmico em Economia, Universidade de Brasília, como requisito parcial para a obtenção do título de Mestre em Economia

Trabalho aprovado. Brasil, 24 de novembro de 2012:

Daniel O. Cajueiro
Orientador

Professor
Convidado 1

Professor
Convidado 2

Brasil
2022, v-0.0.1

Agradecimentos

Este trabalho é dedicado ao meu Pai, Mãe, Esposa e Filho.

Agradeço aos meus pais Antônio Pereira de Lima e Maria de Lourdes Barbosa de Lima por me ensinarem que o caminho para a mudança se inicia com a educação. Durante toda a minha vida seu carinho, atenção e sua crença em mim, me deram forças para acreditar.

À minha querida esposa Rozana de Morais Pinho, que acima de tudo é uma grande amiga, pelo seu amor incondicional e por sempre estar ao meu lado nos momentos difíceis com uma palavra de incentivo.

As minhas irmãs Tatyene Barbosa de Lima São Bernardo e Tuane Barbosa de Lima pela amizade e atenção dedicadas sempre que precisei.

Resumo

Nesse estudo, com a utilização de técnicas de machine learning, procuramos construir um modelo de detecção de fraudes em cartões de crédito, que seja útil para detectar antecipadamente atos ilícitos e contribuir para reduzir prejuízos para as corporações. Utilizamos uma base de dados com transações de cartões de crédito, disponibilizada no site Kaggle e aplicamos técnicas de classificação, em especial os modelos *Random Forest*, Regressão Logistica e Redes Neurais. A expansão da tecnologia moderna e da comunicação global tem propiciado uma maior ocorrência de fraudes, que resulta em perdas substanciais para os negócios. Detectar fraudes e principalmente prever sua ocorrência, portanto, é uma questão de sobrevivência para as corporações. Modelos cada vez mais precisos podem ser elaborados com o advento da inteligência artificial e do *machine learning*, bem como, como o auxílio dos *big datas* e *advanced analytics*. Embora seja impossível impedir a fraude, considerando que é uma atividade inerentemente humana, podemos usar tecnologias de detecção e prevenção, além de engenharia social, para tentar reduzir o risco e ficar um passo à frente dos fraudadores. Como resultado o leitor é capaz de compreender e aprender sobre um *pipeline* que pode auxiliar na prevenção à fraude, e observamos que é possível obter melhores resultados por meio da análise exploratória e a customização das técnicas disponíveis, sendo o modelo de *Random Forests* o que apresentou melhor performance nos testes deste estudo.

Palavras-chave: fraude, aprendizado de máquina, risco, detecção, prevenção.

Abstract

In this study, using machine learning techniques, we seek to build a model for detecting credit card fraud, which is useful for early detection of illicit acts and contributing to reduce losses for corporations. We use a database with credit card transactions, available on the Kaggle website and apply classification techniques, especially the Random Forest models, Logistic Regression and Neural Networks. The expansion of modern technology and global communication has led to a greater occurrence of fraud, which results in substantial losses for the business. Detecting fraud and, above all, predicting its occurrence, therefore, is a matter of survival for corporations. More and more accurate models can be developed with the advent of artificial intelligence and machine learning, as well as, with the help of big data and advanced analytics. While it is impossible to prevent fraud, given that it is an inherently human activity, we can use detection and prevention technologies, as well as social engineering, to try to reduce risk and stay one step ahead of fraudsters. As a result, the reader is able to understand and learn about a pipeline that can help prevent fraud, and we observed that it is possible to obtain better results through exploratory analysis and the customization of available techniques, with the Random Forests which presented the best performance in the tests of this study. **Keywords**: Fraud, Money Laundering, machine learning, risk, detecction.

Sumário

Lista de tabelas

Lista de ilustrações

1 Introdução

Segundo uma pesquisa realizada pela *Association of Certified Fraud Examiners* - ACFE (2016), estima-se que uma empresa típica, perca aproximadamente 5% de sua receita anualmente por fraude. A prevenção e a detecção de fraudes são áreas em crescimento, pois os criminosos ajustam suas táticas de ataque e tentam outros métodos sempre que novos monitoramentos são implantados, e em consequência disso, novos métodos para detecção de fraude são elaborados, e o ciclo se repete. Entender como, onde, porque e por quem a fraude é praticada faz parte do processo de uma boa Governança Corporativa e deve ser priorizado para preservar a sobrevivência da organização, independentemente de seu tamanho e faturamento.

A definição de fraude, segundo o dicionário, diz respeito à deturpação deliberada da verdade ou fatos, geralmente com o objetivo de obter lucro ilícito. Há diferentes tipos de fraudes, entre os principais estão a fraude de cartão de crédito, fraude de seguro, fraude contábil, fraude de garantia do produto, fraude em planos de saúde, fraude de identidade, entre outros. No caso de fraudes com transações de cartão de crédito, a prevenção envolve tomar medidas para evitar que a fraude ocorra antes que uma transação seja concluída.

Há décadas que estudos são realizados com a utilização de técnicas de *machine learning*[1] para detecção e previsão dos diferentes tipos de fraudes. Por exemplo, Green e Choi (1997) e Fanning e Cogger (1998), realizaram pesquisa com foco na avaliação da eficácia de diferentes algoritmos de classificação na detecção de fraudes e já introduziram diferentes variações e redes neurais artificiais (RNA)[2]. Lin, Hwang e Becker (2003) que se concentraram em testar a utilidade de vários algoritmos estatísticos e de aprendizado de máquina, como regressão logística[3] e RNA, para detectar fraudes nas demonstrações financeiras. Ainda, Kotsiantis et al. (2006) e Kirkos, Spathis e Manolopoulos (2007), pesquisadores de fraude em demonstrações financeiras que também avaliaram esses algoritmos de classificação adicionais.

Khare e Sait (2018) exploraram os modelos *decision trees*[4], *random forest*[5], *SVM*[6],

[1] Área da inteligência artificial que estuda como os sistemas podem aprender a identificar padrões e tomar decisões com base nos dados fornecidos a eles.

[2] Modelos computacionais inspirados pelo sistema nervoso central de um cérebro que são capazes de realizar o aprendizado de máquina bem como o reconhecimento de padrões.

[3] Modelo estatístico usado para determinar a probabilidade de um evento acontecer. Ele mostra a relação entre os recursos e, em seguida, calcula a probabilidade de um determinado resultado.

[4] Modelo árvore de decisão é a representação de uma tabela de decisão sob a forma de árvore.

[5] Florestas aleatórias ou florestas de decisão aleatória é um método de aprendizado conjunto para classificação, regressão e outras tarefas que opera construindo uma infinidade de árvores de decisão.

[6] Uma máquina de vetores de suporte é um conceito na ciência da computação para um conjunto de métodos de aprendizado supervisionado que analisam os dados e reconhecem padrões, usado para classificação e análise de regressão

e logistic regression para concluíram que o algoritmo *Random Forest* tem a maior precisão para detectar fraudes. Sahayasakila.V et al. (2019) abordam uma maneira para melhorar a velocidade de convergência e resolver o problema de desequilíbrio de dados. Mqadi, Naicker e Adeliyi (2021) relatou a característica comum aos conjuntos de dados de cartões de crédito, que apresentam muito menos transações fraudulentas do que transações não fraudulentas. Ao lidar com o problema de desequilíbrio do cartão de crédito, a solução ideal deve ter baixo viés e baixa variação, e estudaram o efeito do uso de uma abordagem híbrida de pontos de dados para resolver o problema de classificação incorreta de classes em conjuntos de dados de cartões de crédito desequilibrados. O artigo propôs uma abordagem híbrida de pontos de dados combinando a seleção de recursos com a técnica de subamostragem baseada em *Near Miss*[7] que é uma técnica de *undersampling*[8]. Os resultados mostraram que a abordagem proposta melhorou a precisão preditiva, e dos algoritmos testados, *Random Forest* produziu os melhores resultados. Carcillo et al. (2021) tentou combinar aprendizado não supervisionado e supervisionado na detecção de fraudes com cartão de crédito, e dada a natureza do problema de detecção de fraude, particularmente o um para muitos relação entre cartões e transações, abordada em trabalhos como o de Carcillo et al. (2021) e Carcillo et al. (2018), propuseram uma extensão do melhor dos dois mundos. Para calcular uma pontuação externa consistente para a abordagem local, eles consideraram apenas os cartões com históricos de pelo menos 10 transações no conjunto de treinamento. Esse limite foi definido para preservar a precisão estatística nessa abordagem, uma vez que o cálculo de uma pontuação externa local (mesmo a mais simples) usando menos transações seria inevitavelmente afetado por uma grande variação, o que prejudicaria a precisão geral da abordagem. A novidade da contribuição, além de suas aplicações em conjuntos de dados reais e consideráveis de transações com cartão de crédito, é a implementação e avaliação de diferentes níveis de granularidade para a definição de uma pontuação externa.

Nosso estudo busca demonstrar uma maneira para analisar dados transacionais em uma base de dados com transações de cartões de crédito, disponibilizada no site Kaggle[9], e construir uma aplicação capaz de detectar fraudes. Utilizaremos técnicas de *machine learning* para usar modelos lineares e não-lineares, e obter o melhor índice de detecção entre os modelos analisados.

O trabalho desenvolvido por Sailusha et al. (2020), aborda o tema de fraude em cartão de crédito, pois entende que este é atualmente o problema mais frequente no mundo atual. No estudo o autor foca principalmente em algoritmos de aprendizado de máquina, e utiliza modelos supervisionados como *Random Forest* e *Adaboost*. Seu conjunto de dados de fraude de cartão de crédito, foi obtido do Kaggle, e o dataset contém transações feitas

[7] Algoritmo de undersampling que consiste em reduzir de forma aleatória os exemplos da classe majoritária, porém ele seleciona os exemplos com base na distância

[8] Técnicas de balanceamento que o reduz focando na classe majoritária.

[9] https://www.kaggle.com/datasets/vardhansiramdasu/fraudulent-transactions-prediction

pelos titulares de cartão de crédito em setembro de 2013. As observações representam dois dias, em um total de 284.807 transações, nas quais 492 transações são fraudes (0,172%). As métricas de desempenho utilizadas foram: *Accuracy*[10], *Precision*[11], *Recall*[12], *e F1-score*[13], sendo que na visão do autor, *Random Forest* apresentou o melhor conjunto de resultados para as observações que representavam fraude: *Precision:* 95%, *Recall:* 77% e um *F1:* 85%.

Comparamos o desempenho dos modelos,após a utilização de técnicas de undersampling, além da aplicação da validação cruzada no processo de seleção dos hiperparâmetros.

Nosso trabalho está organizado de maneira a apresentar a revisão da literatura no capítulo 2, o conjunto de dados e a etapa de análise e exploração dos dados no capítulo 3, os modelos e as especificações utilizadas para a previsão e o procedimento adotado para escolha dos parâmetros no capítulo 4, em seguida, mostramos nossos resultados gerais e examinamos os melhores modelos no capítulo 5. Por fim, apresentamos uma breve conclusão no capítulo 6.

[10] A proximidade de um resultado com o seu valor de referência real.
[11] A precisão dos valores indicados como verdadeiros.
[12] Do total de observações a serem observadas, quantas foram identificadas.
[13] Média harmônica entre *precision* e *recall*.

2 Dados

Há uma falta de conjuntos de dados disponíveis publicamente sobre serviços financeiros reais, em parte devido à natureza intrinsecamente privada das transações financeiras. Por consequência, existe grande dificuldade no desenvolvimento de novos métodos de detecção. Os conjuntos de dados financeiros são importantes para muitos pesquisadores, porém mais valiosos ainda para os fraudadores, que podem adquirir informações necessárias para evitar detecções.

Baseamos nosso estudo em uma base de dados pública, conhecida como *"Fraudulent Transactions Prediction"*, disponível no Kaggle [1], são simulações, descrita como sendo proveniente do setor bancário. Nosso *dataset* contém registros de transações financeiras, juntamente com o número de identificação do cliente, seu saldo anterior e posterior à transação, o valor da transação, o tipo de transação, informações do destinatário dos recursos, seu saldo anterior e posterior à transação, seguidos por uma variável responsável por mapear o espaço de tempo entre as transações. Ao todo são 6.362.620 registros com 11 campos (1 variável dependente, 10 variáveis independentes). Para cada registro há um rótulo responsável por classificar a transação em fraude e não fraude, sendo os valores "1"e "0"respectivamente. A porcentagem de registros identificados por fraude é de aproximadamente 0,1291%. A variável denominada ***"step"*** demarca uma unidade de tempo, sendo o número 1 a representação de 1 hora de tempo. No total identificamos 743 valores distintos em ordem crescente. Sendo que um dia é composto por 24 horas, nossa base trata no total, um período de 30 dias. O formato original do arquivo disponibilizado é .csv. A Tabela 1 é um resumo do nome dos campos e o percentual preenchimento de cada um.

Descrição	**Campo**	**% Preenchimento**
Variável Dependente	isFraud	100
Variável Independente Categórica	step	100
	type	100
	isFlaggerFraud	100
	nameOrig	100
	nameDest	100
Variável Independente Numérica	amount	100
	oldbalanceOrg	100
	newbalanceOrig	100
	oldbalanceDest	100
	newbalanceDest	100

Tabela 1 – Resumo dos campos e percentual de preenchimento.

[1] https://www.kaggle.com/datasets/vardhansiramdasu/fraudulent-transactions-prediction

2.1 Análise Exploratória

Como resultado da análise exploratória, algumas descobertas ajudaram a orientar uma análise mais aprofundada.

O número de transações associadas a cada cliente variou entre 1 e 3 transações, sendo 1 transação o número majoritário, representando 99,8537%, contra 2 transações 0,1461% e 3 transações 0,0002%. As 6.362.620 transações, beneficiaram 2.722.362 destinatários. Esses, por sua vez, receberam entre 1 e 113 transações. Na Tabela 2 agrupamos o número de clientes e a quantidade de transações recebidas para identificar a maior e a menor concentração.

Transações Recebidas	**Destinatários**	**%**
1-10	2.592.076	95.21423
11-20	86.959	3.19425
21-30	28.395	1.04303
31-40	10.451	0.38389
41-50	3.448	0.12665
51-60	801	0.02942
61-70	160	0.00588
71-80	40	0.00147
81-90	17	0.00062
91-100	9	0.00033
101-110	5	0.00018
111-120	1	0.00004

Tabela 2 – Resumo da quantidade de transações por destinatário.

Cada destinatário tem um código único, mas eles começam com uma letra. Constatamos que 33,81% das transações foram destinadas a contrapartes iniciadas com a letra "M" (*Merchants* em inglês ou Comerciantes em português). Não há informações de saldo pré e pós-transação neste caso, e não há registros dessas transações sinalizados como fraude. Os 66,19% restantes são transações envolvendo outros clientes cujos saldos podem ser observados antes e depois de cada transação.

A partir deste ponto, podemos inferir que as transações fraudulentas partem de um modus operandi que visa lucrar ao assumir o controle das contas dos clientes e tentar esvaziar os fundos, transferindo para outra conta e posteriormente sacando. Também concluímos que o percentual da base dito anteriormente como fraude (0,1291% - 8.213 transações) poder ser diferente se contabilizado apenas sobre as transações deste tipo, ou seja, se o total de transações realizada é de 6.362.620, e 66,19% (4.211.125) dessas não envolvem comerciantes, podemos aferir um percentual de (4.211.125/ 8.213) 0,1950%. Esse valor é 51% maior do que o apurado sob o total de transações.

O registro abaixo possui o maior valor (***"amount"***) de transação no conjunto de

dados, muito superior ao ticket médio total, ticket médio das operações que não envolvem comerciantes e dos diferentes tipos de transações.

Variável	Valor
step	276
nameOrig	C1715283297
type	TRANSFER
amount	92.445.516,64
oldbalanceOrg	0
newbalanceOrig	0
nameDest	C439737079
oldbalanceDest	9.595,98
newbalanceDest	92.455.112,62
isFlaggedFraud	0
isFraud	0

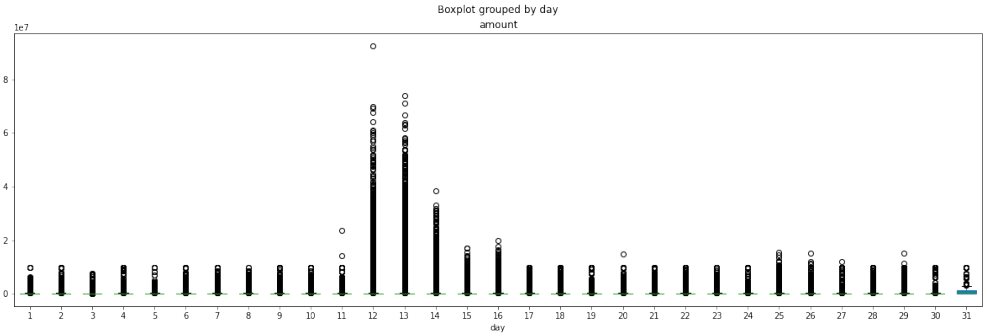
Tabela 3 – Registro com maior valor do conjunto.

Observamos um número grande de *outliers* ao tentar gerar o gráfico de boxplot da variável ***"amount"***. Dado esta condição, criamos um atributo novo, denominado ***"day"***, que foi obtido após a divisão do campo ***"step"*** por 24 (número de horas em um dia) e somamos 1, convertido posteriormente para um número inteiro. Desta maneira, todas as nossas observações passam a possuir um rótulos para representar o dia da transação em relação ao início das observações do *dataset*. Adicionalmente, incluímos um atributo, denominado ***"hour"***, para representar a hora do dia de cada observação, , um para identificar o tipo de entidade destinatária ***"typeDest"***, e um para identificar se a conta foi zerada após a transação ***"emptyAccount"***.

```
import pandas as pd
dfFraud = pd.read_csv('Fraud.csv', sep=',')
dfFraud['day'] = dfFraud['step'].apply(lambda x: int(x/24)+1)
dfFraud['hour'] =  (dfFraud['day']*24) - dfFraud['step']
dfFraud['typeDest'] = dfFraud['nameDest'].apply(lambda x: x[0:1])
dfFraud['emptyAccount'] = np.where(
    (dfFraud['oldbalanceOrg'] > 0) &
    (dfFraud['newbalanceOrig'] == 0), True, False)
```

A seguir, construímos análises das estatísticas descritivas por dia e por hora, além de uma análise do gráfico boxplot, na tentativa de exibir medidas de tendência central não-paramétrica (mediana), de dispersão (quartis), forma de distribuição ou simetria da amostra (valores pontuais mínimo e máximo), valores atípicos (outliers) e extremos. Plotamos a análise boxplot conforme Figura 1 e Figura 2, e os dados na Tabela 4 e Tabela 5.

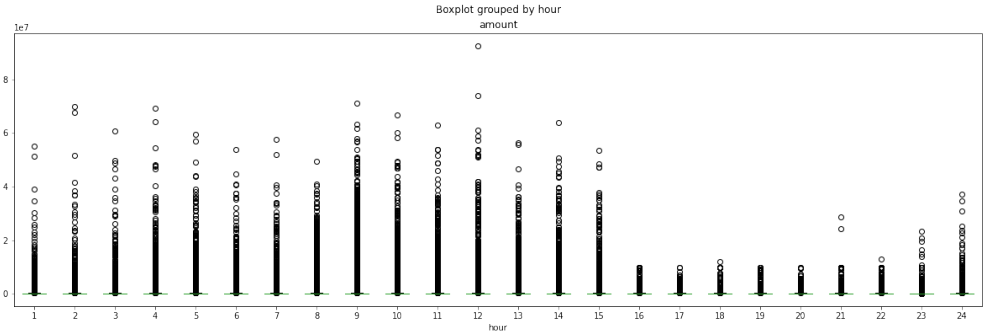
Figura 1 – ***"amount"*** boxplot dia.



Day	Count	Mean	Std	Min	25%	50%	75%	Max
1	571.039	160.654,01	266.159,66	0,10	12.422,82	75.838,36	214.965,15	10.000.000,00
2	452.761	157.014,11	249.213,80	0,14	12.472,92	80.484,06	215.181,95	10.000.000,00
3	6.749	137.527,73	464.610,61	1,59	4.512,85	11.416,16	114.274,06	7.728.992,56
4	21.904	143.893,71	430.583,39	0,24	6.695,80	23.772,37	157.711,52	10.000.000,00
5	12.995	129.752,75	385.209,66	0,68	4.312,10	12.633,57	130.923,60	10.000.000,00
6	440.626	166.604,06	269.353,24	0,21	15.851,93	84.151,62	223.527,90	10.000.000,00
7	420.282	159.430,27	263.104,48	0,06	13.680,09	79.237,75	215.606,76	10.000.000,00
8	449.147	155.992,96	267.820,80	0,02	11.818,48	76.216,89	209.482,49	10.000.000,00
9	418.103	138.632,20	237.966,11	0,00	9.998,49	67.400,22	190.314,88	10.000.000,00
10	392.886	166.065,94	297.637,71	0,02	14.332,28	76.223,64	212.801,75	10.000.000,00
11	418.006	144.542,53	248.507,08	0,00	11.516,98	70.986,60	197.486,59	23.695.249,33
12	349.800	164.142,25	1.012.523,24	0,00	10.276,30	60.617,27	174.175,51	92.445.516,64
13	429.335	323.735,48	1.563.465,55	0,02	18.578,19	83.563,02	224.873,01	73.823.490,36
14	398.210	251.601,29	917.804,06	0,01	16.985,52	78.220,67	220.350,55	38.448.653,52
15	400.706	193.326,71	475.540,47	0,04	15.315,42	75.823,80	211.977,39	17.189.559,66
16	375.514	180.877,82	465.599,59	0,09	14.142,10	74.419,14	209.193,52	19.953.778,06
17	421.098	174.957,32	359.043,80	0,00	17.138,45	78.271,88	212.028,73	10.000.000,00
18	29.251	161.594,00	478.264,77	0,00	7.815,11	45.891,68	182.619,95	10.000.000,00
19	11.286	169.153,03	548.595,85	1,76	9.680,66	51.747,59	174.655,22	10.000.000,00
20	19.739	183.716,40	469.091,55	2,15	11.162,06	68.938,02	208.303,44	14.854.169,86
21	24.589	196.523,94	506.951,26	0,40	12.122,77	71.501,75	206.623,90	10.000.000,00
22	52.510	179.775,25	457.579,43	0,77	12.976,65	66.666,86	196.963,60	10.000.000,00
23	50.432	161.221,61	347.906,44	0,43	12.113,41	69.731,90	199.947,08	10.000.000,00
24	33.349	159.252,93	327.693,09	0,00	11.157,68	72.361,73	208.425,68	10.000.000,00
25	58.712	157.098,96	407.532,82	0,00	10.253,85	68.064,38	196.495,40	15.415.938,31
26	13.893	161.031,96	525.185,38	0,00	10.029,57	59.565,63	178.972,28	15.222.880,76
27	8.574	207.140,07	656.383,47	0,00	14.410,77	85.023,03	219.279,66	11.963.358,83
28	14.522	204.740,55	652.861,37	0,00	9.796,30	70.286,29	216.490,72	10.000.000,00
29	55.037	170.726,73	465.181,53	0,36	13.255,76	73.702,90	206.127,72	15.116.218,46
30	11.283	176.962,27	561.546,19	0,00	13.955,20	76.174,12	207.955,74	10.000.000,00
31	282	1.594.693,20	2.798.297,65	0,00	115.653,73	320.304,78	1.314.169,26	10.000.000,00

Tabela 4 – Estatísticas gerais da variável ***"amount"*** por dia.

Figura 2 – ***"amount"*** boxplot hora.



Hour	Count	Mean	Std	Min	25%	50%	75%	Max
1	141.257	127.671,63	505.115,65	0,00	5.515,54	16.373,45	132.585,87	55.129.569,83
2	194.555	136.009,08	544.212,00	0,00	7.434,59	26.454,65	150.078,65	69.886.731,30
3	247.806	137.008,06	494.380,93	0,00	8.773,56	41.808,02	160.461,08	60.642.003,00
4	553.728	135.317,15	434.788,83	0,00	10.561,31	59.952,33	171.113,52	69.337.316,27
5	647.814	148.937,69	407.109,44	0,26	12.900,11	75.604,45	189.414,03	59.579.503,33
6	580.509	156.599,28	386.616,42	0,20	14.564,69	81.592,58	204.436,18	53.920.358,88
7	439.941	196.503,62	649.977,77	0,00	15.799,23	85.621,48	218.032,76	57.436.619,46
8	441.612	234.391,90	883.390,01	0,14	15.801,14	88.086,44	226.824,36	49.507.088,19
9	416.686	211.512,84	916.181,83	0,00	15.395,14	87.733,46	227.038,54	71.172.480,42
10	439.653	188.830,41	662.263,00	0,06	15.448,36	87.639,43	227.478,10	66.761.272,21
11	468.474	191.272,78	566.501,26	0,02	17.498,67	90.446,45	234.024,20	62.785.416,91
12	483.418	197.272,83	637.471,87	0,17	17.928,78	89.745,79	233.706,98	92.445.516,64
13	445.992	194.703,69	531.657,77	0,57	17.641,54	88.368,01	234.474,78	56.254.995,44
14	425.729	213.114,12	626.830,11	0,00	16.996,80	87.541,39	235.042,08	63.847.992,58
15	283.518	201.857,77	593.016,24	0,18	14.600,06	78.399,17	224.135,69	53.612.432,52
16	26.915	177.290,42	487.648,99	1,77	8.032,16	41.732,63	195.152,79	10.000.000,00
17	8.988	149.383,28	439.876,20	1,76	4.779,32	13.527,74	145.819,58	10.000.000,00
18	3.420	214.368,65	793.165,23	0,00	6.293,10	18.497,41	158.749,34	11.963.358,83
19	1.641	388.394,55	1.223.724,96	3,72	3.668,31	40.975,40	222.097,11	10.000.000,00
20	1.241	364.762,64	1.008.651,66	15,04	2.934,33	54.994,10	246.059,61	10.000.000,00
21	2.007	457.303,20	1.585.308,17	0,00	5.786,81	55.046,74	237.598,26	28.617.438,58
22	9.018	172.448,04	715.004,63	0,00	3.880,14	16.663,28	143.025,12	12.889.398,44
23	27.111	118.048,07	445.984,77	0,03	4.252,88	15.343,17	116.293,58	23.330.626,98
24	71.587	127.580,86	507.903,12	0,14	4.871,03	14.573,27	125.714,26	37.127.946,05

Tabela 5 – Estatísticas gerais da variável ***"amount"*** por hora.

Os dados demonstram que é comum ocorrerem grandes diferenças entre os valores máximos e mínimos de negociação, independente de dia e horário. Sabemos que os fraudadores procuram realizar transações que se assemelham ao comportamento padrão dos demais clientes, e buscam estar próximos aos limites estabelecidos por legislação ou impostos por regras do canal. Esse comportamento os mantém afastados dos sistemas de alertas por longos períodos. Devido a essa situação, entendemos como não necessário considerar todos os valores discrepantes como possíveis fraudes.

Na Figura 1, ao analisar o gráfico pode-se observar que alguns dias (1, 11, 12, 13, 14, 15, 16, 20, 26, 29) apresentam as maiores diferenças em relação ao intervalo interquartílico, e dentre esses, os dias 12 e 13 têm uma das maiores dispersões.

Encontramos o maior desvio padrão no 31º dia, e embora tenha apresentado o menor número de transações, possui o maior volume médio observado na Tabela 4. Ainda nesta, identificamos que as transações tendem a diminuir com o passar dos dias do período analisado, e esta queda inicia-se após o décimo sétimo dia.

Também analisamos o comportamento em relação as horas dentro do período de 1 dia (24 horas). Desta forma identificamos que o número de transações cai após 15 horas e volta a crescer as 21 horas. Na Tabela 5 notamos uma maior frequência de operações entre 4 e 6 horas, mas com um volume médio dentro dos padrões observados. Esse volume padrão é alterado entre 18 e 21 horas, momento em que o número de transações é o menor do período.

2.2 Entidades

Dividimos principalmente os dados com base em dois níveis de entidades: entidades que começam com a letra "M"e entidades que começam com a letra "C"no campo ***"nameDest"***. Observar anomalias no nível dessas duas entidades pode ajudar a explicar as diferenças entre transações de entre diferentes clientes, das transações com comerciantes. Também incluímos ***"type"*** como um nível de entidade, pois representa os tipos de transações efetuadas, analisamos com que frequência esses tipos mudam para um determinado cliente e uma determinada data, e criamos o atributo ***"emptyAccount"*** para identificar as contas que foram zeradas após a transação.

2.3 Fraudes

Nosso conjunto de dados possui um classificador de fraude, o atributo ***"isFraud"***. Ao todo, o conjunto possui 8.213 observações classificadas como fraude, sendo que a média de valores das transações fraudulentas é de $ 1.467.967,30, com um desvio padrão de $ 2.404.253,95, distribuídos conforme Figura 3.

Há uma assimetria com relação aos valores e por esse motivo a média destoa do Q1 que é $127.091,33, Q2 com $ 438.983,45 e Q3 com $ 1.517.771,48. Em média são executadas 264 fraudes por dia (***"day"***) no decorrer de um mês (Figura 5). Se observarmos a frequência por períodos de 24 horas (***"hour"***), na média, são 342 fraudes detectadas (Figura 6).

Percebemos que há um padrão na execução das fraudes, mas não descartamos a hipótese de que esse padrão possa estar relacionado com a maneira como são realizadas as

Figura 3 – Boxplot ***"amount"*** fraudes.

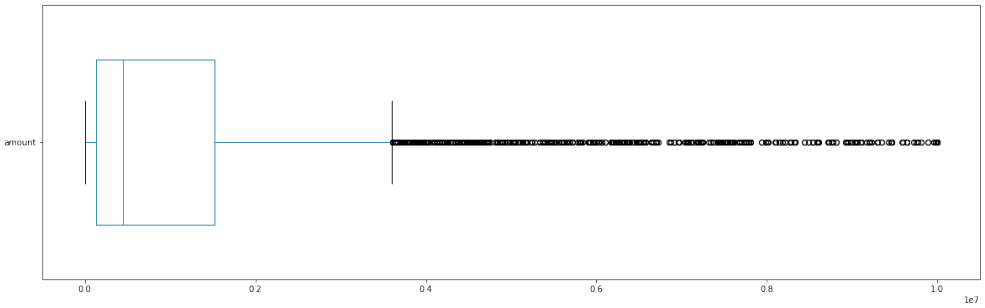


Figura 4 – Histograma ***"amount"*** fraudes.

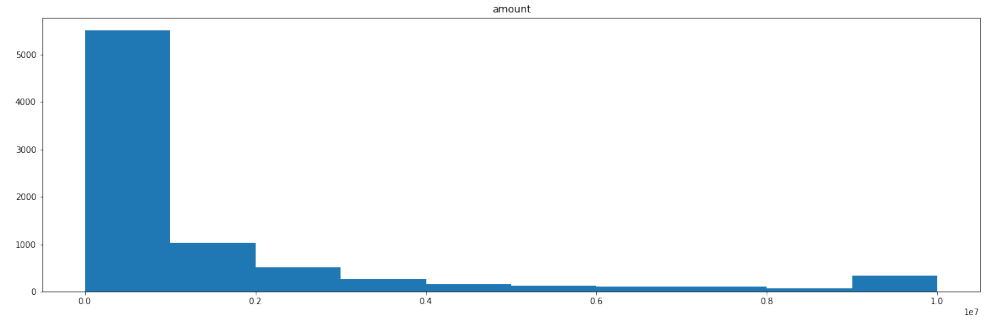
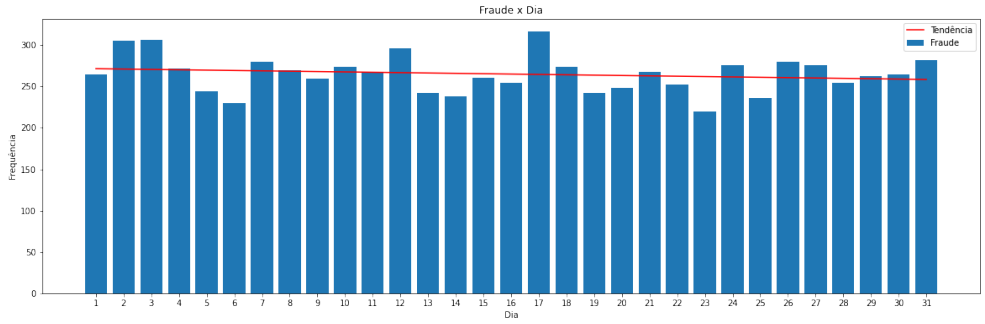


Figura 5 – Fraudes por dia.



detecções das fraudes. A proximidade do volume medido em horas, quanto em dias, pode caracterizar-se por algum método automatizado que mantém padrões para execução das fraudes, ou, o possível conflito conhecido por "Viés do Observador"[2]. A Figura 7 mostra a sazonalidade observada na série, por dia.

As fraudes pertencem a 8.169 destinatários distintos. Esses por sua vez, receberam

[2] Processo de observação e de registro de informações que inclui discrepâncias sistemáticas da verdade. O viés do observador é um tipo de viés de detecção que pode afetar a aferição de desfechos de estudos observacionais e de intervenção.

Figura 6 – Fraudes por hora.

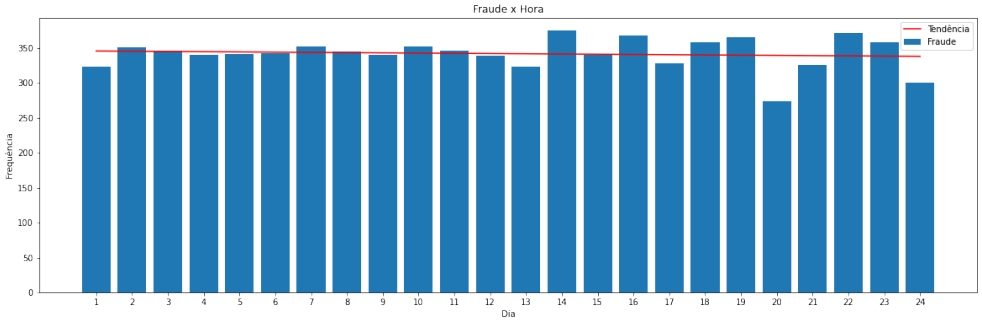
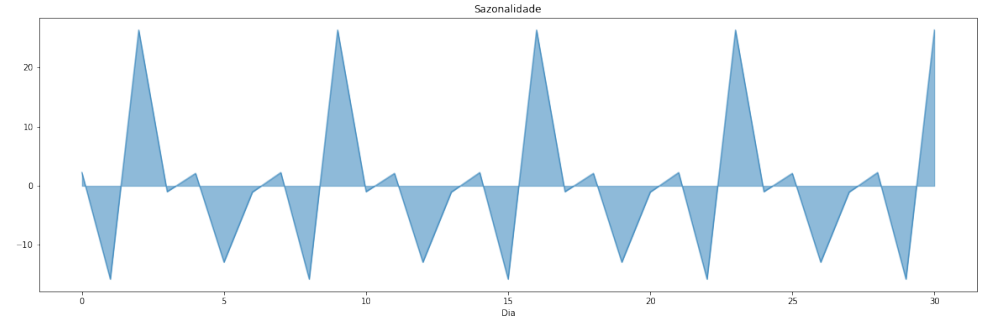


Figura 7 – Sazonalidade observada



65.648, ou seja, nem todas as transações desses destinatários foram identificadas como fraude. Das transações classificadas como fraude, 287 eram de valor $ 10.000.000. Por este motivo, analisamos o número de repetições deste valor nas transações do conjunto e, identificamos que as transações nesta quantia são as mais frequentes da série, e se repetem 3.207 vezes, para 2.124 destinatários distintos. O destinatário "C716083600"recebeu 89 transações, sendo apenas uma classificada como fraude, de valor $ 1.277.212,77. Essa transação faz parte das transações que zeram a conta do cliente remetente, e foi do tipo "CASH_OUT". Neste mesmo momento, a conta destino recebeu outras 11 transações.

2.4 Variáveis

Adicionamos um total de 75 variáveis para modelar nossos dados. Nossa intenção é encontrar anomalias com base no número de transações e os montantes da transação durante um período de tempo.

E nossas análises, identificamos que as contas alvo de fraudes recebem um alto volume de transações no início, e essa frequência diminui nas horas e dias seguintes. Calculamos o número de transações de cada cliente e destinatário, o valor dessas transações e o número de duplicatas no valor em um determinado momento em cada nível de entidade.

Devido aos padrões de fraude observados na Figura 5 e na Figura 6, selecionamos o período de tempo passado em 1, 2, 3 ou 7 dias.

Assumimos que não temos conhecimento de registros que aconteceram após cada registro existente, e utilizamos uma variável para demonstrar o volume do dia anterior. Para cada variável categóricas, criamos uma variável *dummy*.

```
1    df = pd.get_dummies(df, columns=['typeDest', 'cTypeOperation'])
```

Para cada entidade criamos variáveis de ticket médio, e para as variáveis que transmitem informações sobre volume, quantidade, definimos o número total de transações no período de tempo determinado em um determinado nível de entidade como base.

Padronizamos os valores das variáveis numéricas, por meio do método que padroniza um recurso subtraindo a média e, em seguida, dimensionando para a variação da unidade. A variação da unidade significa dividir todos os valores pelo desvio padrão. Como resultado obtemos uma distribuição com um desvio padrão igual a 1. A variância é igual a 1 também, porque variância = desvio padrão ao quadrado. E 1 ao quadrado = 1. Por fim, ele torna a média da distribuição aproximadamente 0.

Utilizamos o PCA[3], que é usado para decompor um conjunto de dados multivariado em um conjunto de componentes ortogonais sucessivos que explicam uma quantidade máxima da variância. a biblioteca usada implementa como um objeto transformador que aprende componentes em seu método de ajuste e pode ser usado em novos dados para projetá-los nesses componentes. As variáveis de tempo e valor não farão parte da redução da dimensionalidade aplicada.

[3] Redução de dimensionalidade, ou redução de dimensão, é a transformação de dados de um espaço de alta dimensão em um espaço de baixa dimensão de forma que a representação de baixa dimensão retenha algumas propriedades significativas dos dados originais, idealmente perto de sua dimensão intrínseca.

3 Métodos

Por meio do método de aprendizagem supervisionada, construímos modelos não-lineares diferentes[1] para calcular uma pontuação de fraude para cada registro e usar os rótulos de fraude para calcular nossa taxa de detecção de fraude. Um modelo linear de regressão logistica compõe o teste. O objetivo é capturar tantas transações fraudulentas quanto possível.

Iniciamos o projeto com uma análise exploratória dos dados, técnica conhecida por *(EDA - Exploratory Data Analysis)*, e abordado em alguns extudos como por exemplo Liu (2014) e Debreceny e Gray (2010), e encontramos características interessantes e inusitadas sobre os dados. Uma vez que entendemos o que os dados realmente significam, começamos a construir as variáveis e a executar o pré-processamento para construção do modelo. Como queremos estudar o comportamento das transações, nos concentramos no número de transações, quantidade de transações repetidas e tipos de transação. Seguindo essas diretrizes, construímos 75 variáveis para modelagem (Apêndice A).

O conjunto de dados mostrou-se desablanceado com relação à variável dependente, portanto, com o objetivo de obter melhores resultados de modelagem e maiores taxas de detecção de fraudes, também criamos amostras: selecionamos apenas parte do registro "não-fraude"de todos os registros fraudulentos. As proporções entre não fraude e fraude são: 1/1, 3/1, 5/1, 7/1 e 10/1.

	Não-Fraude	**Fraude**
Amostra 1/1	422	422
Amostra 3/1	1266	422
Amostra 5/1	2110	422
Amostra 7/1	2954	422
Amostra 10/1	4220	422

Tabela 6 – Amostrar aleatórias.

Dividimos o conjunto de dados em treino, validação e teste, uma técnica para avaliar o desempenho do seu modelo de aprendizado de máquina - classificação ou regressão, técnica conhecida como *(holdout[2])*. Podemos dividir os dados de duas maneiras: aleatória ou usando um componente temporal. Usar uma variável temporal é uma maneira mais

[1] Em estatística, a regressão não linear é uma forma de análise de regressão em que dados observacionais são modelados por uma função que é uma combinação não linear dos parâmetros do modelo e depende de uma ou mais variáveis independentes. Os dados são ajustados por um método de aproximações sucessivas.

[2] Neste método, o conjunto de dados é separado em dois conjuntos, conhecidos por conjunto de treinamento e conjunto de teste.

confiável de dividir conjuntos de dados sempre que o conjunto incluir uma variável deste tipo, e que a necessidade é de prever algo no futuro. Portanto, devemos usar as amostras mais recentes para criar o conjunto de dados de validação e teste. A idéia principal é sempre escolher um subconjunto de amostras que represente os dados fielmente em nosso modelo.

```
X_train, y_train, X_valid, y_valid, X_test, y_test
= train_valid_test_split(df
    , target='isFraud'
    , method='sorted', sort_by_col='step'
    , train_size=0.6, valid_size=0.2, test_size=0.2)
```

Agora possuímos um conjunto de dados de treino (60%), um conjunto de validação (20%) e um conjunto de teste (20%). O conjunto de treino será utilizado para aprendizado (pelo modelo), ou seja, para ajustar os hiperparâmetros do modelo de aprendizado de máquina ao qual submetemos os dados. O conjunto de validação auxilia na obtanção de uma avaliação imparcial do modelo treinado com o conjunto anterior enquanto pode ajustar os hiperparâmetros. Por fim, o conjunto de teste é usado para fornecer uma avaliação imparcial final do modelo já treinado.

Aplicamos validação cruzada na base de treino e avaliamos a estimativa do erro de predição associado a aplicação em novas observações, na base de teste, assim otimizamos o modelo para uma melhor configuração dos hiperparâmetros.

Os modelos como *Randon Forest, SVM, Redes Neurais, CART, Boosted Tree e KNN*, são modelos não-lineares mais sofisticados. Normalmente esses modelos apresentam um desempenho ligeiramente melhor do que a regressão logística (modelo linear). A experiência mostra que a formação de Redes Neurais e Modelos de *Random Forest* são, geralmente, mais eficientes quando as variáveis numéricas independentes são escaladas ou normalizadas, de modo que suas magnitudes são relativamente semelhantes. Por esta razão, nós dimensionamos os dados que alimentamos nos modelos e colocamos todos na mesma escala.

```
X = df.loc[:, ~df.columns.isin(['step', 'isFraud'
    , 'isFlaggedFraud', 'mAmount'])].values
preprocessing.StandardScaler().fit_transform(X)
```

Estamos trabalhando com um conjunto de dados grande, e um número de variáveis considerável e necessitamos preservar a precisão do modelo. Essa situação gera alto custo de processamento. Por esses motivos, optamos por utilizar uma técnica conhecida como redução da dimensionalidade. Quando o espaço de atributos contém somente as características mais relevantes, o classificador tende a ser mais rápido e ocupar menos

recursos computacionais. Para isso, utilizamos a função PCA disponibilizada pela *Lib SkLearn*.

```
1    pca = PCA(n_components=22)
2    pca.fit_transform(X)
```

As próximas seções apresentam as principais características de algoritmos utilizados na etapa de aprendizado de modelos preditivos ajustado para o problema de classificação do presente estudo.

3.1 *Random Forest*

É um método de aprendizado de máquina que combina os preditores de árvores de decisão para que cada árvore dependa do valor de um vetor aleatório amostrado independentemente que tenha a mesma distribuição para todas as árvores (BREIMAN, 2001). Em suma, o modelo cultiva uma floresta e permite que eles votem na categoria mais popular (se estiver em um problema de classificação) ou na média prevista (se estiver em um problema de regressão). Embora os modelos baseados em árvores de decisão tradicionais reduzam a probabilidade de *overtting* ao escolher parâmetros que controlam sua profundidade e número de folhas, este modelo também tenta reduzir o *overtting* de árvores tradicionais combinando diferentes árvores.

Quanto maior o número de árvores, mais precisos serão os resultados da previsão, pois a média de todas essas árvores de decisão é usada para melhorar a eficiência e a precisão da capacidade de previsão do algoritmo. Trata-se de explorar correlações não linearmente entre dados/características experimentais, onde o conjunto de dados permanece o mesmo, no entanto, cada vez que o modelo é treinado, um subconjunto de todo o conjunto de dados é obtido (PANDIMURUGAN et al., 2022). Além disso, o algoritmo classifica as variáveis de acordo com sua importância com base na precisão da classificação, levando em consideração as interações entre as variáveis.

3.2 *Neural Network* / Redes Neurais

Inspirados no complexo funcionamento do cérebro humano, onde centenas de bilhões de neurônios interconectados processam informações em paralelo, os pesquisadores tentaram com sucesso demonstrar um nível de inteligência no ambiente tecnológico. Os exemplos incluem tradução de idiomas e software de reconhecimento de padrões. Uma rede neural artificial (ou rede neural) consiste em uma camada de entrada de neurônios (ou nós, unidades), um ou dois (ou até três) neurônios de camada oculta e uma camada

final de neurônios de saída. Cada conexão está associada a um peso. O resultado h_i do neurônio i é obtido por uma função de ativação:

$$h_i = \sigma\Big(\sum_{j=1}^{N} V_{ij}x_j + T_i^{hid}\Big) \tag{3.1}$$

Onde σ é a função de ativação, N o número de neurônios de entrada, V_{ij} os pesos, x_j os valores de entrada recebido por cada neurônio, e T_i^{hid} os termos de resolução dos neurônios ocultos. O objetivo da função de ativação é, além de introduzir não linearidade na rede neural, limitar o valor do neurônio para que a rede neural não seja paralisada por neurônios divergentes. Um exemplo comum da função de ativação é a função sigmóide (ou logística). O conjunto de redes neurais pode melhorar significativamente a capacidade de generalização dos sistemas de aprendizagem, treinando um número finito de redes neurais e combinando seus resultados. Tem sido considerado como uma tecnologia que possui um ótimo potencial de aplicação (FANNING; COGGER; SRIVASTAVA, 1995).

3.3 *Logistic Regression* / Regressão Logística

Em um modelo de regressão linear simples ou múltipla, a variável dependente Y é uma variável aleatória de natureza contínua. No entanto, em alguns casos a variável dependente é qualitativa, representada por duas ou mais categorias, ou seja, aceita dois ou mais valores. Nesse caso, o método dos mínimos quadrados não fornece um estimador razoável. Uma boa aproximação é obtida por meio da regressão logística, permitindo o uso de modelos de regressão para calcular ou prever a probabilidade de um determinado evento.

As categorias (ou valores) assumidas pela variável dependente podem ser de natureza nominal ou ordinal. No caso da natureza ordinal, há uma ordem natural entre as categorias possíveis, então o contexto da regressão logística ordinal. Quando esta ordem não está presente nas categorias das variáveis independentes, assume-se o contexto de regressão logística nominal.

Variáveis binárias são amplamente utilizadas em estatística para modelar a probabilidade de uma determinada classe ou evento ocorrer.O modelo logístico tem sido o modelo mais comumente usado para regressão binária desde cerca de 1970 Lee (1974), e tem grande força ainda nos dias atuais, como no trabalho de Christodoulou et al. (2019), que também comparou seu resultado ao de técnicas de *machine learning*.

O modelo utiliza a seguinte função logística, responsável pela modelagem da relação entre a probabilidade de determinada resposta, $p(X) = Pr(Y = k|X = x)$, e o conjunto de preditores, X:

$$p(X) = Pr(Y = k|X = x) = \frac{exp\Big(\beta_0 + \sum_{j=1}^{p}\beta_j X_i\Big)}{1 + exp\Big(\beta_0 + \sum_{j=1}^{p}\beta_j X_i\Big)} \tag{3.2}$$

A partir da aplicação do método da máxima verossimilhança, que estima valores para o vetor de parâmetros, β, o modelo logístico pode ser ajustado. A fórmula matemática da função de verossimilhança formaliza o método, de modo que as estimativas de β serão escolhidas a fim de maximizá-la:

$$L(\beta) = \prod_{i=1}^{n}\Big(p(X)^{y_i}(1 - p(X))^{1-y_i}\Big) \tag{3.3}$$

Em problemas de classificação com respostas binárias, 0 e 1, se $p(X)$ é a probabilidade associada à presença de determinada resposta, a medida que descreve a relação entre a resposta de interesse e os preditores mensurados, $(p/1 - p)$, desse evento ocorrer será:

$$\frac{Pr(Y = 1|X = x)}{Pr(Y = 0|X = x)} = \frac{p(X)}{1 - p(X)} = exp(\beta_0 + \sum_{j=1}^{p}\beta_j X_i) \tag{3.4}$$

Quando é aplicada a transformação *logit* à essa medida, o modelo torna-se linear em X e a fronteira de decisão linear do modelo de regressão logística ficará relacionada à escolha de um ponto de corte para $p(X)$:

$$log\Big(\frac{p(X)}{1 - p(X)}\Big) = \beta_0 + \sum_{j=1}^{p}\beta_j X_i \tag{3.5}$$

3.4 *Cross-validation* / Validação Cruzada

Nosso conjunto de dados apresentar dificuldades para métodos como o *holdout* devido o desbalanceamento. Por esse motivo usamos a validação cruzada, que avalia a capacidade de generalização de um modelo por meio da divisão aleatória da base de treino em k partes de tamanhos aproximadamente iguais, em que $k - 1$ irão representar dados de treino para ajuste do modelo preditivo e a outra parte ficará reservada para a estimativa de sua performance. Esse processo se repetiu até que todas as partes tenham participado tanto do treino como da validação do modelo, resultando em k estimativas de performance que serão resumidas, geralmente, pelo cálculo da média e do erro padrão (KUHN; JOHNSON et al., 2013).

Aplicamos o método de validação cruzada chamada *leave-oneout* para estimar os melhores parâmetros de cada modelo testado, antes de avaliar o desempenho, e para isso utilizamos a ferramenta *GridSearchCV* da biblioteca *scikit-learn*.

Usamos o parâmetro $cv = 10; cv = 15; cv = 20; cv = 25; cv = 30$ e $shuffle = True$, ou seja, as informações são misturadas para eliminar qualquer influência da disposição dos dados dentro da função *StratifiedKFolds*, que em bases desbalanceadas garante que em todos os *folds* a proporção das classes sejam a mesma, auxiliando um melhor treinamento e teste em cada simulação.

A técnica de validação cruzada *"GridSearchCV"* auxilia na otimização dos hiperparâmetros, reduzindo as chances de *overfitting* e de subestimar o modelo.

3.5 Desempenho dos Modelos

Para detecção de fraude, bom desempenho significa uma alta taxa de detecção (verdadeiros positivos), ou seja, quantos casos de fraude podem ser detectados corretamente, com uma baixa taxa de falsos positivos, isto é, com que frequência um caso de não fraude é falsamente detectado como fraude.

A avaliação do desempenho dos algoritmos de *machine learning* é realizada medindo o quão bem as previsões derivadas do modelo ajustado reproduzem as observações da resposta (latência) de interesse. Portanto, quantificamos qual é o valor previsto ($\tilde{Y}_i$) da resposta observada/gravada perto de observações, Y_i (KUHN; JOHNSON et al., 2013).

Cada modelo nos gera uma matriz de confusão, indicando os erros e acertos em cada execução.As caselas da diagonal principal ($VPeVN$), Figura 8, denotam casos em que as classes são corretamente preditas, enquanto que as caselas fora da diagonal ($FPeFN$) representam os erros de classificação (KUHN; JOHNSON et al., 2013).

Figura 8 – Matriz de Confusão.

		Classe Predita	
		Positivo	Negativo
Classe Verdadeira	Positivo	Verdadeiro Positivo (VP)	Falso Negativo (FN)
	Negativo	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Com a métrica de avaliação *Precision* conseguimos identificar quantos dos casos que prevemos como positivo, realmente são. Em outras palavras, a assertividade da da previsão.

$$Precision = \frac{VP}{VP + FP} \tag{3.6}$$

Como o desafio é detectar as transações, utilizamos o *Recall* para calcular essa taxa de detecção, pois nos retorna quantos dos casos que são positivos nós acertamos.

$$Recall = \frac{VP}{VP + FN} \tag{3.7}$$

Acombinação de ambos, caso importante, pode ser dada pela métrica *F1-score*, que efetua a mérdia harmônica entre as duas metodologias. Utilizamos essa combinação para saber qual dos modelos apresenta o melhor resultado.

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3.8}$$

Para execução dos modelos e suas métricas de avaliação, desenvolvemos as funções disponíveis no Apêndice B.

4 Resultados

Nosso conjunto de dados possui um grande número de registros, e esse recurso nos apresenta um desafio de infraestrutura para a execução completa do modelo. A partir daí, os resultados obtidos são calculados com base em uma amostra válida de 5% do total. Ao final das simulações realizadas, identificamos diferentes desempenhos entre os algoritmos. Depois de termos um modelo ajustado aos nossos dados, precisamos de uma maneira de quantificar o quão bem o modelo está prevendo em dados nunca antes vistos, ou seja, precisamos de uma maneira de quantificar o quão bem ele generaliza. No método *holdout*, o conjunto de dados original é dividido em treinamento e teste, o modelo é ajustado ao conjunto de treinamento e o conjunto de teste é usado para medir o desempenho do modelo em dados nunca vistos antes. Assim, essa abordagem resulta em uma única medida de erro para um modelo treinado em um conjunto de treinamento específico e avaliado em um conjunto de validação específico.

O problema de obter diferentes medidas de desempenho em diferentes conjuntos de validação decorre do fato de não sabermos quais instâncias irão terminar em cada conjunto e qual é a representação individual de cada instância. Não sendo representativo, qualquer que seja o modelo levará a grandes erros. Por outro lado, podemos acabar com um conjunto de validação que fornece o caso "mais simples"para treinar o modelo, contendo em sua maior parte instâncias com padrões semelhantes aos que o modelo reconhece porém não necessáriamente os padrões que ocorrem no mundo real. Em outras palavras, podemos ter um conjunto de validação de alta variância, que produz uma alta variância de erros computados no conjunto. Isso é obviamente um problema, pois não temos muita garantia de que o valor do verdadeiro modelo de desempenho seja próximo ao valor calculado durante a validação.

Para enfrentar esses desafios, usamos validação cruzada *(validação cruzada - CV)*, que é uma das técnicas mais comuns para selecionar o melhor modelo de um conjunto de modelos ou "ajustar"os hiperparâmetros de um modelo selecionado. Esse método é conhecido principalmente porque resolve o problema de variância no conjunto de validação, resultando em uma avaliação mais robusta do modelo. Ao realizar a validação cruzada, é produzido um conjunto de métricas de desempenho que podem ser resumidas em uma, e esse valor final é uma métrica de desempenho mais poderosa do que simplesmente avaliar o modelo uma vez. Esse processo resulta em uma redução no viés que pode existir em um único conjunto de validação e nos aproxima da obtenção de modelos que generalizam bem para dados nunca antes vistos.

Usar esta abordagem para dividir o conjunto de treino e validação em k grupos de

tamanhos iguais (k folds) temos algo que não se consegue com a abordagem holdout, pois podemos calcular o desvio padrão do erro do modelo no conjunto de validação. Analisar a variância do erro de validação de um modelo é útil em situações em que nossa prioridade não é ter um modelo que possui altíssima acurácia em algumas instâncias específicas e que pode errar feio em outras, mas sim um modelo que possui uma acurácia razoável que se mantém na maior parte das instâncias, ou seja, menor variância.

Optamos por utilizar esse método, inicialmente, para selecionar um dentre diversos modelos. Executamos uma CV para cada modelo e comparamos seus erros médios, eventualmente escolhendo o melhor deles como modelo final.

Analisar a variância do erro de validação de um modelo pode ser útil em situações em que nossa prioridade não é ter um modelo que possui altíssima acurácia em algumas instâncias específicas e que pode errar feio em outras, mas sim um modelo que possui uma acurácia razoável que se mantém na maior parte das instâncias, ou seja, menor variância. Se você já estudou a questão especificidade x sensibilidade pode imaginar situações onde um modelo assim seria preferível.

A utilização da técnica de validação cruzada para seleção dos hiperparâmetros, também permitiu um incremento no desempenho dos modelos

Dividimos os resultados entre os modelos *Randon Forest*, Redes Neurais e Regressão Logística. Para o modelo de RNA utilizamos a composição de camadas 8-2-8.

	RF	**LR**	**RNA**
Acurácia	82.58%	91.00%	90.76%
F1	83.50%	90.45%	90.13%
Recall	88.15%	85.31%	84.36%
Precision	79.32%	96.26%	96.74%

Tabela 7 – Amostra Near Miss

	RF	**LR**	**RNA**
Acurácia	88.98%	72.99%	59.60%
F1	89.22%	72.66%	69.63%
Recall	91.23%	71.80%	92.65%
Precision	87.30%	73.54%	55.78%

Tabela 8 – Amostra 1/1

Muitas vezes precisamos otimizar os hiperparâmetros de um modelo, ou ainda comparar a performance de diferentes modelos. Nesses casos separamos uma parte dos dados para validação e não usamos para treinar os modelos, mas sim para calcular seus erros nesse conjunto e modificar seus parâmetros de forma a diminuir esses erros.

	RF	**LR**	**RNA**
Acurácia	93.48%	83.06%	61.73%
F1	86.32%	58.67%	41.91%
Recall	82.23%	48.10%	55.21%
Precision	90.84%	75.19%	33.77%

Tabela 9 – Amostra 3/1

	RF	**LR**	**RNA**
Acurácia	94.83%	83.81%	65.24%
F1	83.35%	37.31%	29.26%
Recall	77.73%	28.91%	43.13%
Precision	89.86%	52.59%	22.14%

Tabela 10 – Amostra 5/1

	RF	**LR**	**RNA**
Acurácia	95.44%	86.32%	63.95%
F1	79.84%	33.04%	26.64%
Recall	72.27%	27.01%	52.37%
Precision	89.18%	42.54%	17.87%

Tabela 11 – Amostra 7/1

	RF	**LR**	**RNA**
Acurácia	96.57%	89.83%	72.45%
F1	78.88%	29.55%	23.18%
Recall	70.38%	23.46%	45.73%
Precision	89.73%	39.92%	15.53%

Tabela 12 – Amostra 10/1

	RF	**LR**	**RNA**
Acurácia	99.89%	99.86%	99.81%
F1	39.05%	0.0%	3.18%
Recall	25.36%	0.0%	2.37%
Precision	84.92%	0.0%	4.85%

Tabela 13 – Base Geral

O resultado desmonstrado na Tabela 13, conclui que no teste real, onde o dataset é desbalanceado, o modelo *Randon Forest* provou-se o melhor. Porém o resultado obtido demonstra baixo desempenho do *F1-score*, que é altamente prejudicado pelo desbalanceamento da base. Tentamos melhorar o resultado por meio da técnica de redução da

dimensionalidade, mas os resultados não foram satisfatórios.

Após o processo de otimização dos modelos por meio de técnicas de hiperparâmetros, o modelo de *Random Forest* desempenhou melhores resultados. Desta maneira a proposta para o cenário que analisamos seria de aplicar o modelo *Random Forest* a novos conjuntos de dados e realizar as previsões de fraude, permitindo a tomada de decisões a partir do resultado gerado pelo modelo. Usando *F1-score* conseguimos obter uma alta taxa de registros com alta probabilidade de serem fraudes verdadeiras e não falsos-positivos (78,57%). Usar o resultado do modelo para suportar os analistas na decisão de classificação de transações como fraude/não-fraude pode beneficiar instituições que identifiquem-se com o cenário apresentado.

5 Conclusões

A importância de ter sistemas, controles e práticas eficazes para gerenciar o risco de fraude nos bancos requer métodos eficazes para detectar as transações com maior probabilidade de ilicitude. Como parte do controle geral, as soluções de combate à fraude podem automatizar e ajudar a reduzir a parte manual do processo de monitoramento e seleção. Formas eficazes de soluções contra a fraude são necessárias, mas extremamente desafiadoras. Em particular, o aumento nos volumes de transações do cliente e o aumento nas interações automatizadas com o cliente tornaram este ambiente mais difícil.

Este estudo fornece uma abordagem para que instituições possam mitigar o risco de operações suspeitas, e uma métrica de comparação entre os resultados dos modelos, capaz de direcionar a melhor escolha com base nos resultados mais apurados de detecção. Para atingir esses objetivos, foram demonstradas características importantes das quais a solução deve ser capaz de lidar, isso inclui uma ampla visão das operações no que diz respeito ao tempo entre as ocorrências e a relação com os envolvidos. A abordagem demonstrada não exime tantas outras possíveis, como por exemplo, usar a entidade de tempo em períodos menores do que um dia, ou diferentes tipos de modelos. No início, os dados brutos obtidos de instituições financeiras geralmente resultam em volumes extremamente grandes e os conjuntos de dados costumam ser desbalanceados. A utilização de técnicas como *feature engineering* permitem a construção variáveis explicativas de alto poder preditivo. Buscamos construir um caminho que levasse o leitor a entender as principais etapas e processos que envolvem um bom tratamento de dados, e um bom processo de monitoramento anti-fraude, que inibi possíveis não conformidades, aumentar a eficiência por meio de alertas qualificados e confiáveis. Embora a conclusão do estudo tenha se dado por completo, acreditamos que os resultados demonstrados podem ser desafiados e melhorados.

Referências

ACFE, I. F. Report to the nations on occupational fraud and abuse. 2016. Disponível em: <https://www.acfe.com/-/media/files/acfe/pdfs/2016-report-to-the-nations.ashx>. Citado na página 10.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001. Citado na página 24.

CARCILLO, F. et al. Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization. *International Journal of Data Science and Analytics*, Springer, v. 5, n. 4, p. 285–300, 2018. Citado na página 11.

CARCILLO, F. et al. Combining unsupervised and supervised learning in credit card fraud detection. *Information sciences*, Elsevier, v. 557, p. 317–331, 2021. Citado na página 11.

CHRISTODOULOU, E. et al. A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models. *Journal of clinical epidemiology*, Elsevier, v. 110, p. 12–22, 2019. Citado na página 25.

DEBRECENY, R. S.; GRAY, G. L. Data mining journal entries for fraud detection: An exploratory study. *International Journal of Accounting Information Systems*, Elsevier, v. 11, n. 3, p. 157–181, 2010. Citado na página 22.

FANNING, K.; COGGER, K. O.; SRIVASTAVA, R. Detection of management fraud: a neural network approach. *Intelligent Systems in Accounting, Finance and Management*, Wiley Online Library, v. 4, n. 2, p. 113–126, 1995. Citado na página 25.

FANNING, K. M.; COGGER, K. O. Neural network detection of management fraud using published financial data. *Intelligent Systems in Accounting, Finance & Management*, Wiley Online Library, v. 7, n. 1, p. 21–41, 1998. Citado na página 10.

GREEN, B. P.; CHOI, J. H. Assessing the risk of management fraud through neural network technology. *Auditing*, AUDITING SECTION, AMERICAN ACCOUNTING ASSOCIATION, v. 16, p. 14–28, 1997. Citado na página 10.

KHARE, N.; SAIT, S. Y. Credit card fraud detection using machine, learning models and collating machine. *International Journal of Pure and Applied Mathematics*, v. 118, n. 20, p. 825–838, 2018. ISSN 1314-3395. Disponível em: <https://acadpubl.eu/hub/2018-118-21/articles/21b/90.pdf>. Citado na página 10.

KIRKOS, E.; SPATHIS, C.; MANOLOPOULOS, Y. Data mining techniques for the detection of fraudulent financial statements. *Expert systems with applications*, Elsevier, v. 32, n. 4, p. 995–1003, 2007. Citado na página 10.

KOTSIANTIS, S. et al. Forecasting fraudulent financial statements using data mining. *International journal of computational intelligence*, v. 3, n. 2, p. 104–110, 2006. Citado na página 10.

KUHN, M.; JOHNSON, K. et al. *Applied predictive modeling*. [S.l.]: Springer, 2013. v. 26. Citado 2 vezes nas páginas 26 e 27.

LEE, E. T. A computer program for linear logistic regression analysis. *Computer programs in biomedicine*, Elsevier, v. 4, n. 2, p. 80–92, 1974. Citado na página 25.

LIN, J. W.; HWANG, M. I.; BECKER, J. D. A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal*, MCB UP Ltd, 2003. Citado na página 10.

LIU, Q. *The application of exploratory data analysis in auditing*. Tese (Doutorado) — Rutgers University-Graduate School-Newark, 2014. Citado na página 22.

MQADI, N. M.; NAICKER, N.; ADELIYI, T. Solving misclassification of the credit card imbalance problem using near miss. *Mathematical Problems in Engineering*, Hindawi, v. 2021, 2021. Citado na página 11.

PANDIMURUGAN, V. et al. Random forest tree classification algorithm for predicating loan. *Materials Today: Proceedings*, Elsevier, v. 57, p. 2216–2222, 2022. Citado na página 24.

SAHAYASAKILA.V et al. Credit card fraud detection system using smote technique and whale optimization algorithm. *International Journal of Engineering and Advanced Technology (IJEAT)*, v. 8, n. 5, 2019. ISSN 2249-8958. Disponível em: <https://www.ijeat.org/wp-content/uploads/papers/v8i5/D6468048419.pdf>. Citado na página 11.

SAILUSHA, R. et al. Credit card fraud detection using machine learning. In: IEEE. *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. [S.l.], 2020. p. 1264–1270. Citado na página 11.

Apêndices

APÊNDICE A – Variáveis

	Variável	**Descrição**
1	ID	Registro de Identificação
2	bFraud	Fraude ? (Verdadeir ou Falso)
3	typeDest	Tipo de Destino (Cliente ou Comercio)
4	cTypeOperation	Tipo de Operação (Transafer, Payment, Debit, Cash$_in, Cash_out$)
5	bEmptyAccount	A conta foi zerada após a transação? (Verdadeiro ou Falso)
6	mAmount	Volume da transação
7	Dest Trans 0	Quantidade de Transações para o destinatário no dia / Quantidade de Transações para o destinatário em 30 dias
8	Dest Trans 1	Quantidade de Transações para o destinatário no anterior / Quantidade de Transações para o destinatário em 30 dias
9	Dest Trans 2	Quantidade de Transações para o destinatário dois dias antes / Quantidade de Transações para o destinatário em 30 dias
10	Dest Trans 3	Quantidade de Transações para o destinatário três dias antes / Quantidade de Transações para o destinatário em 30 dias
11	Dest Trans 7	Quantidade de Transações para o destinatário sete dias antes / Quantidade de Transações para o destinatário em 30 dias
12	Dest Amoun Tk 0	Volume Movimentado para o destinatário no dia/ Quantidade de Transações para o destinatário em 30 dias
13	Dest Amoun Tk 1	Volume Movimentado para o destinatário no dia anterior/ Quantidade de Transações para o destinatário em 30 dias
14	Dest Amoun Tk 2	Volume Movimentado para o destinatário dois dias antes/ Quantidade de Transações para o destinatário em 30 dias
15	Dest Amoun Tk 3	Volume Movimentado para o destinatário três dias antes/ Quantidade de Transações para o destinatário em 30 dias
16	Dest Amoun Tk 7	Volume Movimentado para o destinatário sete dias antes/ Quantidade de Transações para o destinatário em 30 dias
17	Dest Amoun 0	Volume Movimentado para o destinatário no dia/ Volume Movimentado para o destinatário em 30 dias
18	Dest Amoun 1	Volume Movimentado para o destinatário no dia anterior/ Volume Movimentado para o destinatário em 30 dias

19	Dest Amoun 2	Volume Movimentado para o destinatário dois dias antes/ Volume Movimentado para o destinatário em 30 dias
20	Dest Amoun 3	Volume Movimentado para o destinatário três dias antes/ Volume Movimentado para o destinatário em 30 dias
21	Dest Amoun 7	Volume Movimentado para o destinatário sete dias antes/ Volume Movimentado para o destinatário em 30 dias
22	Orig Trans 0	Quantidade de Transações da conta origem no dia / Quantidade de Transaçõesda conta origem em 30 dias
23	Orig Trans 1	Quantidade de Transações da conta origem no anterior / Quantidade de Transaçõesda conta origem em 30 dias
24	Orig Trans 2	Quantidade de Transações da conta origem dois dias antes / Quantidade de Transaçõesda conta origem em 30 dias
25	Orig Trans 3	Quantidade de Transações da conta origem três dias antes / Quantidade de Transaçõesda conta origem em 30 dias
26	Orig Trans 7	Quantidade de Transações da conta origem sete dias antes / Quantidade de Transaçõesda conta origem em 30 dias
27	Orig Amoun Tk 0	Volume Movimentado da conta origem no dia/ Quantidade de Transaçõesda conta origem em 30 dias
28	Orig Amoun Tk 1	Volume Movimentado da conta origem no dia anterior/ Quantidade de Transaçõesda conta origem em 30 dias
29	Orig Amoun Tk 2	Volume Movimentado da conta origem dois dias antes/ Quantidade de Transaçõesda conta origem em 30 dias
30	Orig Amoun Tk 3	Volume Movimentado da conta origem três dias antes/ Quantidade de Transaçõesda conta origem em 30 dias
31	Orig Amoun Tk 4	Volume Movimentado da conta origem sete dias antes/ Quantidade de Transaçõesda conta origem em 30 dias
32	Orig Amoun 0	Volume Movimentado da conta origem no dia/ Volume Movimentado da conta origem em 30 dias
33	Orig Amoun 1	Volume Movimentado da conta origem no dia anterior/ Volume Movimentado da conta origem em 30 dias
34	Orig Amoun 2	Volume Movimentado da conta origem dois dias antes/ Volume Movimentado da conta origem em 30 dias
35	Orig Amoun 3	Volume Movimentado da conta origem três dias antes/ Volume Movimentado da conta origem em 30 dias
36	Orig Amoun 7	Volume Movimentado da conta origem sete dias antes/ Volume Movimentado da conta origem em 30 dias
37	Dest Dupli 0	Quantidade de Transações Duplicadas para o destinatário no dia / Quantidade de transações para o destinatário no dia
38	Dest Dupli 1	Quantidade de Transações Duplicadas para o destinatário no dia anterior/ Quantidade de transações para o destinatário no dia anterior

39	Dest Dupli 2	Quantidade de Transações Duplicadas para o destinatário dois dias antes/ Quantidade de transações para o destinatário dois dias antes
40	Dest Dupli 3	Quantidade de Transações Duplicadas para o destinatário três dias antes/ Quantidade de transações para o destinatário três dias antes
41	Dest Dupli 7	Quantidade de Transações Duplicadas para o destinatário sete dias antes/ Quantidade de transações para o destinatário sete dias antes
42	Orig Dupli 0	Quantidade de Transações Duplicadas da conta origem no dia / Quantidade de transações da conta origem no dia
43	Orig Dupli 1	Quantidade de Transações Duplicadas da conta origem no dia anterior/ Quantidade de transações da conta origem no dia anterior
44	Orig Dupli 2	Quantidade de Transações Duplicadas da conta origem dois dias antes/ Quantidade de transações da conta origem dois dias antes
45	Orig Dupli 3	Quantidade de Transações Duplicadas da conta origem três dias antes/ Quantidade de transações da conta origem três dias antes
46	Orig Dupli 4	Quantidade de Transações Duplicadas da conta origem sete dias antes/ Quantidade de transações da conta origem sete dias antes
47	Dest TypeOperation Trans 0	Quantidade de Transações deste tipo para o destinatário no dia / Quantidade de Transações deste tipo para o destinatário em 30 dias
48	Dest TypeOperation Trans 1	Quantidade de Transações deste tipo para o destinatário no anterior / Quantidade de Transações deste tipo para o destinatário em 30 dias
49	Dest TypeOperation Trans 2	Quantidade de Transações deste tipo para o destinatário dois dias antes / Quantidade de Transações deste tipo para o destinatário em 30 dias
50	Dest TypeOperation Trans 3	Quantidade de Transações deste tipo para o destinatário três dias antes / Quantidade de Transações deste tipo para o destinatário em 30 dias
51	Dest TypeOperation Trans 7	Quantidade de Transações deste tipo para o destinatário sete dias antes / Quantidade de Transações deste tipo para o destinatário em 30 dias
52	Dest TypeOperation Amoun Tk 0	Volume Movimentado deste tipo para o destinatário no dia/ Quantidade de Transações deste tipo para o destinatário em 30 dias

53	Dest TypeOperation Amoun Tk 1	Volume Movimentado deste tipo para o destinatário no dia anterior/ Quantidade de Transações deste tipo para o destinatário em 30 dias
54	Dest TypeOperation Amoun Tk 2	Volume Movimentado deste tipo para o destinatário dois dias antes/ Quantidade de Transações deste tipo para o destinatário em 30 dias
55	Dest TypeOperation Amoun Tk 3	Volume Movimentado deste tipo para o destinatário três dias antes/ Quantidade de Transações deste tipo para o destinatário em 30 dias
56	Dest TypeOperation Amoun Tk 7	Volume Movimentado deste tipo para o destinatário sete dias antes/ Quantidade de Transações deste tipo para o destinatário em 30 dias
57	Dest TypeOperation Amoun 0	Volume Movimentado deste tipo para o destinatário no dia/ Volume Movimentado deste tipo para o destinatário em 30 dias
58	Dest TypeOperation Amoun 1	Volume Movimentado deste tipo para o destinatário no dia anterior/ Volume Movimentado deste tipo para o destinatário em 30 dias
59	Dest TypeOperation Amoun 2	Volume Movimentado deste tipo para o destinatário dois dias antes/ Volume Movimentado deste tipo para o destinatário em 30 dias
60	Dest TypeOperation Amoun 3	Volume Movimentado deste tipo para o destinatário três dias antes/ Volume Movimentado deste tipo para o destinatário em 30 dias
61	Dest TypeOperation Amoun 7	Volume Movimentado deste tipo para o destinatário sete dias antes/ Volume Movimentado deste tipo para o destinatário em 30 dias
62	Orig TypeOperation Trans 0	Quantidade de Transações deste tipo da conta origem no dia / Quantidade de Transações deste tipo da conta origem em 30 dias
63	Orig TypeOperation Trans 1	Quantidade de Transações deste tipo da conta origem no anterior / Quantidade de Transações deste tipo da conta origem em 30 dias
64	Orig TypeOperation Trans 2	Quantidade de Transações deste tipo da conta origem dois dias antes / Quantidade de Transações deste tipo da conta origem em 30 dias
65	Orig TypeOperation Trans 3	Quantidade de Transações deste tipo da conta origem três dias antes / Quantidade de Transações deste tipo da conta origem em 30 dias
66	Orig TypeOperation Trans 7	Quantidade de Transações deste tipo da conta origem sete dias antes / Quantidade de Transações deste tipo da conta origem em 30 dias

67	Orig TypeOperation Amoun Tk 0	Volume Movimentado deste tipo da conta origem no dia/ Quantidade de Transações deste tipo da conta origem em 30 dias
68	Orig TypeOperation Amoun Tk 1	Volume Movimentado deste tipo da conta origem no dia anterior/ Quantidade de Transações deste tipo da conta origem em 30 dias
69	Orig TypeOperation Amoun Tk 2	Volume Movimentado deste tipo da conta origem dois dias antes/ Quantidade de Transações deste tipo da conta origem em 30 dias
70	Orig TypeOperation Amoun Tk 3	Volume Movimentado deste tipo da conta origem três dias antes/ Quantidade de Transações deste tipo da conta origem em 30 dias
71	Orig TypeOperation Amoun Tk 7	Volume Movimentado deste tipo da conta origem sete dias antes/ Quantidade de Transações deste tipo da conta origem em 30 dias
72	Orig TypeOperation Amoun 0	Volume Movimentado deste tipo da conta origem no dia/ Volume Movimentado deste da conta origem em 30 dias
73	Orig TypeOperation Amoun 1	Volume Movimentado deste tipo da conta origem no dia anterior/ Volume Movimentado deste da conta origem em 30 dias
74	Orig TypeOperation Amoun 2	Volume Movimentado deste tipo da conta origem dois dias antes/ Volume Movimentado deste da conta origem em 30 dias
75	Orig TypeOperation Amoun 3	Volume Movimentado deste tipo da conta origem três dias antes/ Volume Movimentado deste da conta origem em 30 dias
76	Orig TypeOperation Amoun 7	Volume Movimentado deste tipo da conta origem sete dias antes/ Volume Movimentado deste da conta origem em 30 dias
77	Dest TypeOperation Trans OneDayBefore 0	Quantidade de Transações deste tipo para o destinatário no dia anterior/ Quantidade de Transações deste tipo para o destinatário em 30 dias
78	Dest TypeOperation Trans OneDayBefore 1	Quantidade de Transações deste tipo para o destinatário no dia anterior/ Quantidade de Transações deste tipo para o destinatário em 30 dias
79	Dest TypeOperation Trans OneDayBefore 2	Quantidade de Transações deste tipo para o destinatário no dia anterior/ Quantidade de Transações deste tipo para o destinatário em 30 dias
80	Dest TypeOperation Trans OneDayBefore 3	Quantidade de Transações deste tipo para o destinatário no dia anterior/ Quantidade de Transações deste tipo para o destinatário em 30 dias

81	Dest TypeOperation Trans One-DayBefore 7	Quantidade de Transações deste tipo para o destinatário no dia anterior/ Quantidade de Transações deste tipo para o destinatário em 30 dias

Variáveis.

APÊNDICE B – Funções

```
def evaluate_model_cross(cv, X, y, tp):
  if(tp == 'RF'):
    clf = RandomForestClassifier()
  elif(tp=='LR'):
    clf = LogisticRegression()
  elif(tp=='RNA'):
    clf = MLPClassifier()
  elif(tp=='XG'):
    clf = xgboost.XGBClassifier()

  y_pred = cross_val_predict(clf, X, y, cv=cv, n_jobs=-1)
  acc = round(accuracy_score(y,y_pred) *100,2)
  auc = round(roc_auc_score(y, y_pred)*100,2)
  f1  = round(f1_score(y,y_pred)*100,2)
  rec = round(recall_score(y,y_pred)*100,2)
  pre = round(precision_score(y,y_pred)*100,2)

  df = pd.DataFrame({tp: [str(acc) + '%',
                          str(auc) + '%',
                          str(f1) + '%',
                          str(rec) + '%',
                          str(pre) +'%'
                          ]},
                    index = ['Acurácia','AUC', 'F1','Recall','Precision'])
  return df
```

APÊNDICE C – Dummy

```
df1.fillna(0,inplace=True)
df1 = pd.get_dummies(df1, columns=['typeDest', 'cTypeOperation'])
df1.drop(columns=['ID','dDateTime'], inplace=True)
X = df1.loc[:, ~df1.columns.isin(['iDay', 'iHour', 'step', 'bFraud', 'bFraudFlagged',
                                  'bEmptyAccount', 'mAmount'])]
y = df1.iloc[:,df1.columns == 'bFraud']
z = df1.loc[:, df1.columns.isin(['iDay', 'iHour', 'step', 'bFraudFlagged',
                                 'bEmptyAccount', 'mAmount'])]
X = preprocessing.StandardScaler().fit_transform(X)
X = pd.DataFrame(X, index=df1.index).join([z])
```

APÊNDICE D – Sample

```
tt_fraud = df1[df1['bFraud']==1].shape[0]
print(tt_fraud)
nrm = NearMiss()
X_nrm,y_nrm = nrm.fit_resample(X,y)

amostra0 = pd.concat([pd.DataFrame(y_nrm, columns=['bFraud']),
                      pd.DataFrame(X_nrm)], axis=1)
amostra1 = pd.concat([df1[df1['bFraud']==1].sample(tt_fraud),
                      df1[df1['bFraud']!=1].sample(tt_fraud)])
amostra3 = pd.concat([df1[df1['bFraud']==1].sample(tt_fraud),
                      df1[df1['bFraud']!=1].sample(tt_fraud*3)])
amostra5 = pd.concat([df1[df1['bFraud']==1].sample(tt_fraud),
                      df1[df1['bFraud']!=1].sample(tt_fraud*5)])
amostra7 = pd.concat([df1[df1['bFraud']==1].sample(tt_fraud),
                      df1[df1['bFraud']!=1].sample(tt_fraud*7)])
amostra10 = pd.concat([df1[df1['bFraud']==1].sample(tt_fraud),
                       df1[df1['bFraud']!=1].sample(tt_fraud*10)])
```

APÊNDICE E – Teste Amostras

```
N = 0,1,3,5,7,10
dfPredRF  = evaluate_model_cross(10, amostraN.iloc[:,
    ~amostraN.columns.isin(['bFraud'])].values,amostraN['bFraud'].values,'RF')
dfPredLR  = evaluate_model_cross(10, amostraN.iloc[:,
    ~amostraN.columns.isin(['bFraud'])].values,amostraN['bFraud'].values,'LR')
dfPredRNA = evaluate_model_cross(10, amostraN.iloc[:,
    ~amostraN.columns.isin(['bFraud'])].values,amostraN['bFraud'].values,'RNA')
dfPredRF.join([dfPredLR, dfPredRNA])

#TOTAL

dfPredRF  = evaluate_model_cross(10, X,y,'RF')
dfPredLR  = evaluate_model_cross(10, X,y,'LR')
dfPredRNA = evaluate_model_cross(10, X,y,'RNA')
dfPredRF.join([dfPredLR, dfPredRNA])
```

APÊNDICE F – Modelo Selecionado

```
X_train, y_train, X_valid, y_valid, X_test, y_test = train_valid_test_split(df1,
    target='bFraud', method='sorted', sort_by_col=['step'], train_size=0.6,
    valid_size=0.2, test_size=0.2)

clf = RandomForestClassifier()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_valid)

acc = round(accuracy_score(y_valid,y_pred) *100,2)
auc = round(roc_auc_score(y_valid, y_pred)*100,2)
f1  = round(f1_score(y_valid,y_pred)*100,2)
rec = round(recall_score(y_valid,y_pred)*100,2)
pre = round(precision_score(y_valid,y_pred)*100,2)

df = pd.DataFrame({'RF': [str(acc) + '%',str(auc)+ '%', str(f1) +'%', str(rec) +'%',
                          str(pre) +'%']},
                  index = ['Acurácia','AUC', 'F1','Recall','Precision'])
df.T

kfold = KFold(n_splits=15, shuffle=True)
kf_cv_scores = cross_val_score(clf, X_train, y_train, cv=kfold, scoring="f1")
print("K-fold CV average score: %.2f" % kf_cv_scores.mean())

y_pred = cross_val_predict(clf, X_train, y_train, cv=kfold, n_jobs=-1)

acc = round(accuracy_score(y_train,y_pred) *100,2)
auc = round(roc_auc_score(y_train, y_pred)*100,2)
f1  = round(f1_score(y_train,y_pred)*100,2)
rec = round(recall_score(y_train,y_pred)*100,2)
pre = round(precision_score(y_train,y_pred)*100,2)

df = pd.DataFrame({'RF': [str(acc) + '%',str(auc)+ '%', str(f1) +'%', str(rec) +'%',
                          str(pre) +'%']},
                  index = ['Acurácia','AUC', 'F1','Recall','Precision'])
df.T

n_estimators = [25, 50, 100, 200, 500, 900, 1100, 1500, 2500]
min_samples_split = [2, 4, 6, 10]
min_samples_leaf = [1, 2, 4, 6, 8]
max_features = ['auto', 'sqrt','log2', None]
param = {'n_estimators': n_estimators,
         'min_samples_split': min_samples_split,
         'min_samples_leaf': min_samples_leaf,
         'max_features': max_features}

clf_gv = RandomizedSearchCV(clf, param, scoring='f1', cv=kfold, refit=True, n_iter=20)
clf_gv.fit(X_test, y_test)

best_model = clf_gv.best_estimator_
yhat = best_model.predict(X_valid)
f1 = f1_score(y_valid, yhat)
```

```
52 f1
```