
**MATHEUS SARAIVA ALCINO**

# MODELOS DE CLASSIFICAÇÃO EM FRAUDES FINANCEIRAS:

COMPARAÇÃO DE DESEMPENHO EM CASOS DE CRIME DE SMURFING

**LAVRAS – MG**

**2022**

**MATHEUS SARAIVA ALCINO**

# **MODELOS DE CLASSIFICAÇÃO EM FRAUDES FINANCEIRAS**: COMPARAÇÃO DE DESEMPENHO EM CASOS DE CRIME DE SMURFING

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Estatística e Experimentação Agropecuária, área de concentração em Estatística e Experimentação Agropecuária, para obtenção do título de Mestre.

Prof. Renato Ribeiro de Lima
Orientador

Prof. Erick Galani Maziero
Coorientador

**LAVRAS – MG**
**2022**

**Ficha catalográfica elaborada pelo Sistema de Geração de Ficha Catalográfica da Biblioteca Universitária da UFLA, com dados informados pelo(a) próprio(a) autor(a).**

Alcino, Matheus Saraiva
Modelos de classificação em fraudes financeiras : Comparação de desempenho em casos de crime de smurfing / Matheus Saraiva Alcino. – Lavras : UFLA, 2022.
84 p. : il.

Dissertação–Universidade Federal de Lavras, 2022.
Orientador: Prof. Renato Ribeiro de Lima.
Bibliografia.

1. Fraudes Financeiras. 2. Smurfing. 3. Machine Learning. I. Lima, Renato Ribeiro de. II. Maziero, Erick Galani. III. Título.

**MATHEUS SARAIVA ALCINO**

# **MODELOS DE CLASSIFICAÇÃO EM FRAUDES FINANCEIRAS**: COMPARAÇÃO DE DESEMPENHO EM CASOS DE CRIME DE SMURFING
# **FINANCIAL FRAUD CLASSIFICATION MODELS:** PERFORMANCE COMPARISON IN SMURFING CRIME CASES

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Estatística e Experimentação Agropecuária, área de concentração em Estatística e Experimentação Agropecuária, para obtenção do título de Mestre.

APROVADA em 03 de Fevereiro de 2022.

| | |
|---|---|
| Prof. Paulo Henrique Sales Guimarães | UFLA |
| Prof. Danilo Machado Pires | UNIFAL-MG |

Prof. Renato Ribeiro de Lima
Orientador

Prof. Erick Galani Maziero
Co-Orientador

**LAVRAS – MG**
**2022**

*À minha mãe, Renata de Oliveira Saraiva, e ao meu pai, José Éder Alcino.*

## AGRADECIMENTOS

A realização deste trabalho só foi possível porque tenho comigo pessoas que não mediram esforços para me ajudar no que fosse preciso. A todas elas eu expresso a minha profunda gratidão.

Começo por minha esposa Beatriz, que me encoraja e tenta me entender. E agradeço também aos meus irmãos Euler e Kaleb, por todo o suporte.

A universidade pública me transformou em uma pessoa melhor e ainda me deu oportunidade de conhecer pessoas e torná-las minhas amigas, as quais eu me apoio constantemente quando me encontro em desafios e dificuldades acadêmicas. Dentre todas estas amizades, é enorme a necessidade de enfatizar meus professores e amigos Lincoln Frias e Patrícia Ramos, que me fizeram acreditar em meu potencial e me apoiaram de maneira fundamental para que eu pudesse realizar minhas pesquisas de mestrado. E meus amigos de graduação e pós graduação Alice Duarte, Walef Machado e Poliana Maria Benelli, que em muitos momentos me fizeram enxergar desafios sob a luz de outras perspectivas.

Aos amigos da equipe de Ciência de Dados da Agência de Inovação Zetta, em especial Antônio Couto Jr. e meu coorientador Erick G. Maziero, tenho imensa gratidão. Ambos possuem um papel essencial no desenvolvimento desta pesquisa pois ambos me apresentaram o desafio e acreditaram que eu pudesse superá-lo.

# RESUMO

A dificuldade de identificação de fraudes financeiras possui relação direta com o avanço tecnológico, pois as novas possibilidades de formas de transações financeiras geram por sua vez novas formas de agentes fraudadores atuarem. Neste contexto, o objetivo deste estudo é explorar a teoria de seis modelos de *machine learning* (ML), além de compará-los por meio de métricas específicas de avaliação de desempenho. Ainda, este trabalho desenvolve um algoritmo de detecção de um tipo de crime financeiro conhecido como *smurfing*. Tal algoritmo não utiliza técnicas de ML, mas objetiva ranquear transações financeiras como possíveis fraudes, através de características analisadas de forma agrupada. Devido à impossibilidade de uso de dados financeiros reais, por causa de sua confidencialidade, este trabalho é desenvolvido utilizando dados simulados. Foram gerados dois diferentes cenários, ambos altamente desbalanceados, em que o comportamento das fraudes financeiras varia de acordo com parâmetros específicos. Os modelos de classificação escolhidos se referem ao modelo logístico, Sistemas Baseados em Regras Fuzzy, Redes Neurais Artificiais, *Random Forest*, *Extreme Gradient Boosting* e *Support Vector Machine*. A comparação dos modelos nos diferentes cenários foi feita através de uma combinação das métricas *Area Under de Curve*, Recall e $F_\beta$, tendo em vista o desbalanceamento dos dados. Os resultados apontaram que os modelos *Random Forest* e *Extreme Gradient Boosting* obtiveram os melhores desempenhos e, dessa forma, acredita-se que o uso de tais modelos em dados reais, ainda que com diferentes parametrizações, pode ajudar no rastreamento de operações financeiras ilegais e identificação de fraudadores.



**Palavras-chave:** Fraudes financeiras. *Machine Learning*. Smurfing.

# ABSTRACT

The difficulty in identifying financial fraud is directly related to technological advances, as the new possibilities of forms of financial transactions, in turn, generate new forms of fraudulent agents to act. In this context, the aim of this study is to explore the theoretical construction of six machine learning (ML) models, in addition to comparing them through specific performance evaluation metrics. Furthermore, this work develops an algorithm to detect a type of financial crime known as smurfing. This algorithm does not use ML techniques, but aims to classify financial transactions as possible fraud through the analysis of pooled data. Given the impossibility of using real financial data, due to its confidentiality, this work is using simulated data. Two different scenarios were generated, both highly unbalanced, in which the behavior of financial fraud varies according to specific parameters. The chosen classification models were logistic model, Fuzzy Rule Based Systems, Artificial Neural Networks, Random Forest, Extreme Gradient Reinforcement and Support Vector Machine. The comparison of the models in the different scenarios was done through a combination of the metrics *Area Under de Curve*, Recall and $F_\beta$, once data are imbalanced. The results showed that the Random Forest and Extreme Gradient Boosting models had the best performances, therefore, it is believed that the use of such models in real data, even with different parameters, can help in tracking illegal financial transactions and identifying fraudsters



**Keywords:** Financial frauds. Machine Learning. Smurfing.

## LISTA DE FIGURAS

## LISTA DE TABELAS

**SUMÁRIO**

# 1 INTRODUÇÃO

A dificuldade de um processo de investigação de crimes financeiros começa na própria definição do que são esses crimes. De acordo com Pickett e Pickett (2002) não existe uma conceituação precisa para esse tipo de crime, mas de maneira geral, crimes financeiros estão associados ao conceito de fraude, que por sua vez também não possui uma única definição neste contexto. Os crimes financeiros abordam uma vasta variedade de tipos de atividades ilegais como crimes de colarinho branco, lavagem de dinheiro, falsificação, contas bancárias de passagem, práticas de *smurfing* etc., e normalmente são crimes não violentos com fins lucrativos cometidos por meio de engano por pessoas cuja condição ocupacional é empreendedora, profissional ou semi profissional, utilizando suas habilidades ocupacionais especiais e oportunidades (GARNER et al., 2004).

Existe uma série de atividades consideradas ilegais que podem ser enquadradas como crimes financeiros. Pickett e Pickett (2002) afirmam que algumas destas atividades, que podem inclusive ocorrer tanto no setor privado quanto no público, se referem a fraude de consumidor, cartão de crédito, propinas, manipulação de licitações, faturas inflacionadas, fraudes externas, roubo de estoque, roubo de dinheiro, fraudes básicas de empresas, etc.

Dessa forma, a identificação de padrões justifica a exigência de uma definição precisa do que são crimes financeiros. Ademais, o processo de identificação em uma era de grande quantidade de dados, inviabiliza o uso de métodos tradicionais (manuais) de detecção, criando também uma demanda por métodos computacionais (WEST; BHATTACHARYA, 2016). O principal problema associado aos crimes financeiros é a natureza aparentemente insolúvel desses crimes (SADDIQ; BAKAR, 2019). A solução desses crimes passa por um processo de identificação de padrões que necessariamente demanda técnicas matemáticas e estatísticas cada vez mais robustas, dado que atualmente existem técnicas sofisticadas de ocultar crimes financeiros.

Ngai et al. (2011) fizeram uma revisão de literatura com as principais técnicas de mineração de dados voltadas para a detecção de fraudes financeiras, tais como classificação, regressão, agrupamento, previsão, detecção de outlier e visualização. De acordo com esses autores, a mineração de dados desempenha um papel importante na detecção de fraudes financeiras, já que é frequentemente aplicada para extrair e descobrir as verdades ocultas por trás de grandes quantidades de dados. Dentre as definições que os autores utilizaram para a encontrar padrões

e descobrir fraudes, os autores se limitaram às categorias de fraudes bancárias, de seguros, de títulos e commodities, e por fim, fraudes diversas que incorporam por exemplo fraudes corporativas e de marketing em massa.

Além de uma boa definição do problema a ser trado, a escolha do método também precisa se mostrar adequada. West e Bhattacharya (2016) fizeram um levantamento das principais técnicas estatísticas utilizadas na literatura no processo de identificação de fraudes financeiras. Entre os métodos utilizados para tarefas de regressão e classificação destacam-se aqueles que utilizam algoritmos de inteligência artificial como redes neurais, SVM (*support vector machine*), árvores de decisão, regressão logística, lógica fuzzy e alguns modelos híbridos.

Em um processo de modelagem estatística é importante o conhecimento das características das variáveis envolvidas, tais como a sua distribuição, independência, aleatoriedade, comportamento assintótico, etc. A modelagem estatística requer assumir algumas pressuposições sobre os dados, como por exemplo o conhecimento sobre o modelo probabilístico, a aleatoriedade dos dados, a independência. Neste sentido, a utilização de modelos de *machine learning* (ML) se torna conveniente, uma vez que eles são construídos a partir dos dados e, portanto, não dependem de uma distribuição adequada ou qualquer outra premissa para que se torne um modelo estatisticamente válido.

Neste contexto, o presente estudo teve por objetivo explorar a construção teórica de alguns modelos que compõem a classe de modelos de *machine learning*, além de compará-los através de seus desempenhos em tarefas de classificação de crimes financeiros. Para isso, existe a necessidade de definir o que este estudo considera como crimes ou fraudes financeiras. A partir da definição geral de Garner et al. (2004) são investigadas transações financeiras que configuram um tipo específico de crime financeiro, explorado na subseção 2.1.1.

Os objetivos específicos deste estudo se resumem na aplicação de modelos de ML para a classificação de crimes financeiros, visando compará-los através de métricas de desempenho analisadas conjuntamente. Além disso, este estudo ainda propõe um algoritmo inicial que auxilia a identificação de possíveis casos de um tipo específico de crime financeiro. Tal algoritmo é útil pelo fato de que em meio a uma massa densa de dados financeiros, de inúmeras variáveis, o trabalho de identificar evidências de crimes financeiros é humanamente inviável, ainda mais quando existem técnicas sofisticadas que criminosos utilizam para que suas operações fraudulentas sejam parecidas com operações regulares. Ainda, o algoritmo proposto é de viável aplicação

quando não se tem uma base de dados rotulada, o que impede que os modelos propostos sejam treinados.

A partir das revisões de literatura feitas por West e Bhattacharya (2016), Ngai et al. (2011) espera-se desempenhos semelhantes entre os modelos, uma vez que todos eles quando utilizados em problemas similares ao deste trabalho atingiram resultados relativamente expressivos. Entretanto, este estudo leva em consideração a avaliação de uma série de métricas de maneira agrupada, a partir de dados simulados e penalizando modelos que obtiveram taxas significativas de um tipo específico de erro. Dessa maneira, é possível estabelecer um ranking entre os melhores modelos, que posteriormente poderão ser utilizados em outros estudos ou até em investigações reais.

Este trabalho está estruturado através da sequência de capítulos: no Capítulo 2, estão contidas as definições de crimes financeiros abordadas neste trabalho, bem como algumas especificações do crime de *smurfing*. Ainda nesse capítulo, para todos os modelos propostos é feita uma exploração teórica de cada um, a fim de atender um dos objetivos gerais deste estudo. No Capítulo 3, é esclarecido como e com quais tipos de dados este estudo pretende alcançar os objetivos específicos. Além disso, uma análise exploratória dos dados utilizados é apresentada, bem como um algoritmo de detecção de *smurfing*. Por fim, nos Capítulos 4 e 5 são apresentados os resultados e as conclusões deste estudo, respectivamente.

# 2 REFERENCIAL TEÓRICO

## 2.1 Fraudes financeiras

O entendimento do senso comum sobre fraudes financeiras permite o pensamento de que elas são baseadas no uso da desonestidade para obter ganhos e vantagens indevidas, usufruindo frequentemente de uma posição de confiança. Entretanto, a definição empregada pela ampla literatura sociológica sobre este fenômeno dá mais ênfase à camuflagem da normalidade que torna a fraude bem sucedida (CETINA; PREDA, 2013).

É possível estabelecer um grau de associação entre fraudes bem sucedidas e o grau de confiança entre o enganador e o enganado. Isto é o que caracteriza, principalmente, golpes financeiros como fraudes de interação. Cetina e Preda (2013) justificam esta afirmação alegando que as fraudes financeiras tendem a serem mais bem sucedidas quando a vítima conhece o fraudador.

Fraudes financeiras podem ser estudadas sob a ótica de um contexto amplo e, uma vez que se deseja encontrar métodos de detecção desse tipo de irregularidade, faz-se necessário a aplicação de filtro que permita definições mais precisas. Uma forma de limitar todas as possíveis interpretações e definições do problema é a separação por categorias.

Em sua revisão de literatura, Reurink (2018) faz uma distinção conceitual entre três tipos de fraudes financeiras. O autor considera como fraudes de demonstrativo financeiro uma variedade de comportamentos em que os agentes do mercado financeiro fazem declarações falsas sobre a verdadeira natureza ou saúde financeira de um canal de investimento.

A outra categoria que Reurink (2018) apresenta diz respeito a golpes financeiros, cujo autor define como esquemas enganosos e totalmente fraudulentos. Os fraudadores, em geral, assumem identidades falsas ou demonstram ser agentes de confiança, que convencem, enganam ou induzem as pessoas a interagir voluntariamente com o fraudador. Assim, as vítimas são convencidas a entregar voluntariamente dinheiro ou informações confidenciais relacionadas às suas finanças pessoais.

Por fim, Reurink (2018) apresenta uma terceira distinção conceitual de fraudes financeiras. Esta diz respeito às negociações ou vendas enganosas, que se resumem a indução ou influência de venda, ou ainda, aconselhamento manipulador de um produto ou serviço financeiro para um

usuário final. Os fraudadores o fazem sabendo que o produto ou serviço é inadequado para algum uso final específico.

Com relação à categoria referente a golpes financeiros, fraudes em cartões de crédito e lavagem de dinheiro são exemplos de práticas irregulares em que modelos de classificação podem ser alternativas úteis para a detecção deste tipo de crime. Existem modelos matemáticos e estatísticos capazes de realizar a tarefa de reconhecer padrões a partir de atividades passadas e, a aplicação desses métodos no processo de investigação de golpes financeiros pode ajudar no rastreamento de transações irregulares e até mesmo no rastreamento de fraudadores.

A sofisticação das formas de aplicação de golpes financeiros tenta acompanhar o avanço tecnológico que ao mesmo tempo em que facilita as relações econômicas e as tornam acessíveis, também abrem mais possibilidades para fraudadores atuarem. Em seu estudo, Karpoff (2021) investigou o problema das fraudes financeiras ao longo do tempo. De acordo com o autor, existem divergências na literatura entre autores que encontraram indicadores de que este é um problema que está diminuindo ou aumentando com o passar dos anos. Para Karpoff (2021), o problema das fraudes financeiras está relacionado com o avanço tecnológico e com as mudanças do nível de riqueza das sociedades. O autor afirma que no longo prazo tais mudanças justificarão a diminuição da incidência de fraude.

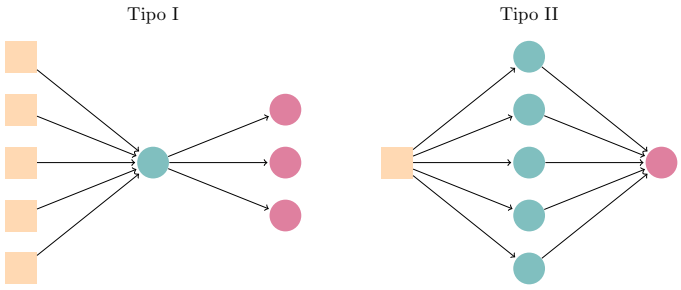
### 2.1.1 Smurfing

A diversificação das formas de lavagem de dinheiro torna a identificação desta prática um desafio para órgãos públicos responsáveis por punir aqueles que o fazem. Dentro de cada uma destas formas é possível ainda que haja inúmeras técnicas de se praticar um mesmo tipo de lavagem de dinheiro.

Starnini et al. (2021), Chadha e Kaur (2018) afirmam que existem três fases da lavagem de dinheiro: i) alocação, ii) camadas e iii) integração. Durante a fase de alocação, os ativos adquiridos ilicitamente são introduzidos no sistema financeiro legítimo, enquanto são limpos dos vestígios mais evidentes de ilegalidade. Na fase de camadas, criminosos movimentam o dinheiro por meio de uma série de transações que não têm nenhum propósito real a não ser esconder a natureza criminosa do dinheiro. Por fim, na fase de integração, esses ativos são integrados à economia legal e outros ativos podem ser utilizados regularmente (STARNINI et al., 2021).

Um dos casos que caracterizam fielmente a fase de camadas é conhecido na literatura como *smurfing* (CHADHA; KAUR, 2018; STARNINI et al., 2021). *Smurfing* é uma prática de lavagem de dinheiro em que se objetiva transacionar um montante de dinheiro considerável entre dois indivíduos através de transações repetitivas, oriundas de diferentes depósitos em dinheiro ou transferências que individualmente não representam valor relevante. O fracionamento de um valor significativo de ativos dificulta a detecção e classificação destes casos como ilegais ou não. Isto é, inclusive, uma característica da maioria das formas de lavagem de dinheiro, ou seja, o comportamento que os seus praticantes dão às movimentações financeiras é semelhante a um comportamento financeiro de uma conta regular. Starnini et al. (2021) afirmam que este fracionamento pode ser dado através de dois tipos, que são mostrados na Figura 2.1.

Figura 2.1 – Tipos de *smurfing*. Fonte dos recursos (quadrado), intermediário (círculo verde) e destino de recursos (círculo rosa).




Fonte: Starnini et al. (2021).

A própria estrutura que define cada tipo de crime de smurfing ilustra a complexidade de detecção que o problema pode ter. Inclusive, é comum o uso de ferramentas gráficas, tais como análise de grafos, na tentativa de compreender esquemas fraudulentos e identificar fraudadores.

A prática de crimes financeiros impacta diretamente a economia dos países. Didimo et al. (2011) afirmam que não há uma mensuração precisa de tal impacto, mas globalmente estes impactos podem chegar a 1 trilhão de dólares anualmente. Este é um problema que tem sido discutido na literatura e diversos autores o têm estudado e proposto soluções para sua identificação. Pseudoalgoritmos, grafos e etc., são exemplos de métodos propostos, respectivamente, por Chadha e Kaur (2018), Starnini et al. (2021) para a identificação do problema. Na seção 3.3 é proposto um algoritmo para detecção de casos de *smurfing*, mas que precisa ser confirmado através da verificação humana.

## 2.2 Modelos de classificação

### 2.2.1 Modelo logístico

Modelos Lineares Generalizados (GLM) representam uma classe extensa de modelos que buscam descrever relações lineares entres variáveis. Todos os GLM são construídos através de três pilares: uma componente aleatória, uma componente sistemática e uma função de ligação. A componente aleatória diz respeito à variável resposta, cujas premissas assumidas se referem a sua aleatoriedade e distribuição de probabilidade pertencente à família exponencial. Já a componente sistemática, diz respeito à função do preditor da variável regressora, que deve ser linear nos parâmetros. Por fim, a função de ligação é a componente responsável por conectar as demais componentes, através de uma função $g(\mu) = \boldsymbol{X}\boldsymbol{\beta}$, em que $\mu = E(Y)$ (CASELLA; BERGER, 2002).

Casella e Berger (2002) afirmam que um GLM descreve a relação entre a média de uma variável resposta $Y$ e variáveis independentes $\boldsymbol{X}$. Nem sempre a relação descrita por um GLM é tão simples quanto $E(Y) = \boldsymbol{X}\boldsymbol{\beta}$, fato que justifica toda a extensão da classe GLM. Um exemplo se refere à tentativa de estabelecer uma relação direta entre uma variável dicotômica $Y \sim Bernoulli(\pi)$ e $\boldsymbol{X}\boldsymbol{\beta}$, em que é mais apropriado o uso de uma função de ligação chamada logit ou probit.

Um desses casos, quando se utiliza o logit, é denominado de regressão logística, amplamente utilizado para tarefas de classificação binária. No caso binário, $Y_i \sim Bernoulli(\pi_i)$ e como a distribuição Bernoulli pertence à família exponencial, neste modelo assume-se que $\pi_i$ está relacionado à $X_i$ através da função

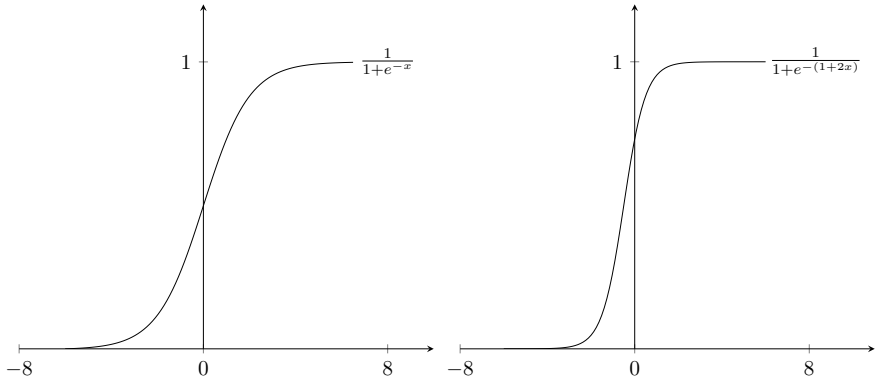
$$g(\mu) = ln\left(\frac{\pi_i}{1-\pi_i}\right) = \boldsymbol{X}\boldsymbol{\beta}, \tag{2.1}$$

em que, o termo anterior a igualdade dado em (2.1), se refere ao logaritmo da chance de sucesso $(Y = 1)$. A equação (2.1) pode ser reescrita de forma que

$$\pi(x) = \frac{e^{\boldsymbol{X}\boldsymbol{\beta}}}{1+e^{\boldsymbol{X}\boldsymbol{\beta}}} = \frac{1}{1+e^{-(\boldsymbol{X}\boldsymbol{\beta})}}. \tag{2.2}$$

O comportamento da função logística (2.2) com diferentes parametrizações é mostrado na Figura 2.2.

Figura 2.2 – Exemplos de funções sigmoidais.




Fonte: Do autor (2021).

Justamente por não haver uma conexão direta entre $Y_i$ e $\boldsymbol{X}\boldsymbol{\beta}$, Casella e Berger (2002) afirmam que o método de estimação de mínimos quadrados não é apropriado para o caso da regressão logística e, em contrapartida, pode-se utilizar a máxima verossimilhança. Dado que $Y_i \sim Bernoulli(\pi_i)$, em que a média $\pi_i$ é uma função linear $\pi(x_i)$ a função de verossimilhança é

$$L(\boldsymbol{\beta}|\boldsymbol{y}) = \prod_{i=1}^{n} \pi(x_i)^{y_i}[1-\pi(x_i)]^{1-y_i}. \tag{2.3}$$

Tomando o logaritmo natural de (2.3) e supondo, para efeitos de simplificação um caso $p$-variado, tem-se que

$$\begin{aligned}
\ell(\boldsymbol{\beta}|\boldsymbol{y}) &= \sum_{i=1}^{n} y_i ln(\pi) + (1-y_i) ln(1-\pi) \\
&= \sum_{i=1}^{n} ln(1-\pi) + \sum_{i=1}^{n} y_i ln\left(\frac{\pi}{1-\pi}\right) \\
&= \sum_{i=1}^{n} ln(1-\pi) + \sum_{i=1}^{n} y_i \eta \\
&= \sum_{i=1}^{n} -ln(1+e^{\eta}) + \sum_{i=1}^{n} y_i \eta \ , \qquad (2.4)
\end{aligned}$$

em que $\eta = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$, tal que $p > 1$.

Por fim, os estimadores não-viesados do vetor de $\beta$'s são obtidos pela derivação da equação (2.4) e, posteriormente, igualando-a a zero. Shalizi (2013) afirma que o sistema de equações gerado durante o processo de estimação de parâmetros é um sistema transcendental e não possui solução fechada, por isso, métodos numéricos são usados para alcançar soluções aproximadas. A necessidade de tais métodos é ainda maior quando $Y$ envolve mais de duas classes, o que aumenta a dificuldade de encontrar as raízes da equação. Dentre estes métodos o Newton-Raphson é um dos mais antigos e utilizados para tal tarefa (SHALIZI, 2013).

Cada coeficiente estimado representa um impacto na probabilidade de sucesso $(Y = 1)$, ou seja, o efeito de cada variável inclusa em um modelo logístico está relacionado ao aumento ou diminuição da probabilidade de sucesso $\pi(x_i)$ do evento estudado. A regressão logística é um classificador linear e para minimizar o erro de classificação assume-se que $Y_i = 1$ quando $\pi(x_i) \geq 0,5$ e $Y_i = 0$ quando $\pi(x_i) < 0,5$ (SHALIZI, 2013).

O modelo logístico tem sido utilizado com frequência na detecção de crimes financeiros, especialmente em fraudes de cartões de crédito, como nos estudos de Awoyemi, Adetunmbi e Oluwadare (2017), Bhattacharyya et al. (2011) e Campus (2018).

### 2.2.2 Redes neurais artificiais

Sendo um dos métodos de aprendizado de máquina mais utilizados em diversos campos da pesquisa, as redes neurais artificiais (RNA) são modelos matemáticos capazes de estabelecer relações entre variáveis de entrada e saída através de "neurônios" (ROSENBLATT, 1958). As estruturas de redes neurais permitem que modelos compostos por várias camadas de processamento aprendam representações de dados com vários níveis de abstração (TORRES, 2018).

Os neurônios são as menores unidades de uma RNA e são responsáveis por determinar os pesos (ou importâncias) de cada variável entrada no modelo. Os pesos são gerados através de funções denominadas funções de ativação, que por sua vez são responsáveis por dar ao modelo a capacidade de lidar com a não linearidade na modelagem (TORRES, 2018).

A construção teórica de uma rede neural tem origem naquilo que a literatura define como modelos estáticos não lineares. Nelles (2020) afirma que tais modelos descrevem o comportamento não linear da relação entre uma variável dependente $r-$dimensional $\boldsymbol{y} = [y_1, \ldots, y_r]^T$ em função de um vetor de variáveis independentes $\boldsymbol{x} = [x_1 \ldots, x_p]^T$, como é mostrado na equação (2.5).

$$\hat{y} = f(\boldsymbol{x}) \ . \tag{2.5}$$

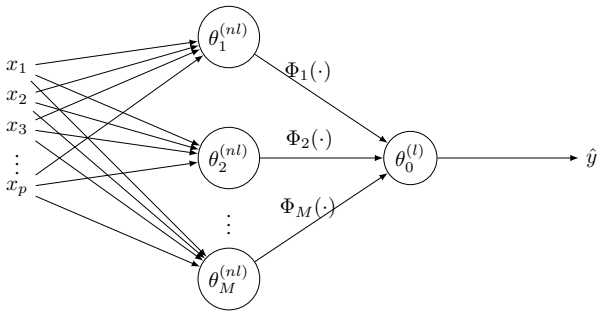
Em geral, estes modelos são subclassificados em duas categorias: MIMO – múltiplas entradas e múltiplas saídas (do inglês *multiple-input-multiple-output*; e MISO – múltiplas entradas e única saída (do inglês *multiple-input-single-output*).

A função $f$ da equação (2.5) é conhecida como função básica de um modelo estático não linear. Para qualquer possível escolha da função $f$, Nelles (2020) afirma que esta pode ser escrita através da equação (2.6) para quase todos os casos de interesse prático.

$$\hat{y} = \sum_{i=1}^{M} \theta_i^{(l)} \Phi_i(\boldsymbol{x}, \boldsymbol{\theta}_i^{(nl)}) \ , \tag{2.6}$$

em que $M$ é o total de funções base $\Phi_i(\cdot)$, que por sua vez são ponderadas por parâmetros lineares $\theta_i^{(l)}$. Além disso, cada função base depende de variáveis de entrada $\boldsymbol{x}$ e um conjunto de parâmetros não lineares reunidos em $\boldsymbol{\theta}_i^{(nl)}$. Além dos termos presentes na equação (2.6) é comum a inclusão de um termo de deslocamento, que melhora o ajuste do modelo. Na Figura 2.3 é mostrada uma representação gráfica de um modelo estático não linear evidenciando onde ocorre a atuação das funções base.

Figura 2.3 – Exemplo de sistema estático não linear.




Fonte: Nelles (2020, p. 242).

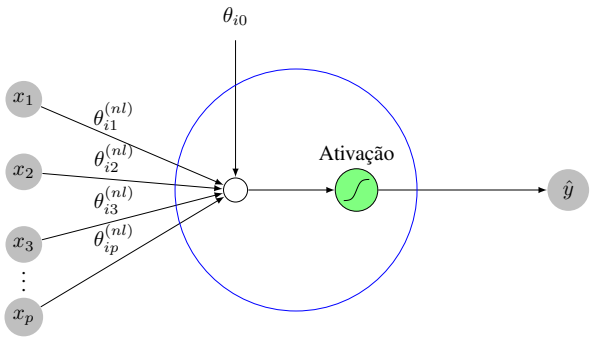
Cada nó da rede representada na Figura 2.3 pode conter função $\Phi(\cdot)$ própria e distinta das demais. Nos casos em que todas elas são do mesmo tipo em todos os nós a rede é chamada de rede neural artificial (NELLES, 2020).

Uma rede neural pode ser estruturada de diversas maneiras, entre elas, a arquitetura conhecida como *Multilayer Perceptron* (MLP) é uma das mais utilizadas (NELLES, 2020). O

termo *perceptron* se refere a um algoritmo de aprendizado supervisionado e, a arquitetura MLP é composta por diversas camadas perceptron (neurônios).

As operações matemáticas realizadas em cada neurônio de uma MLP são compostas, primeiramente, pela projeção de um vetor de entradas $\boldsymbol{x} = [x_1, \ldots, x_p]^T$ nos pesos; e depois por uma função de ativação que transforma este resultado projetado. As funções de ativação linear, sigmoidal, tangente hiperbólica, *softmax* e ReLU (unidade linear retificada) são as mais comuns (TORRES, 2018). A relação entre um modelo logístico e uma rede neural MLP pode se estreitar, a depender da escolha da função de ativação e também da quantidade de camadas utilizada. Isto porque é comum a escolha da função sigmoidal como função de ativação, conforme afirma Nelles (2020, p. 288). Tal função, que em modelo lineares generalizados é conhecida como função de ligação, é expressa pela equação (2.2). Na Figura 2.4 é mostrado um exemplo de arquitetura MLP que utiliza a função sigmoidal como função de ativação.

Figura 2.4 – *I-ésimo* neurônio de uma MLP.




Fonte: Do autor (2021).

Na Figura 2.4 tem-se uma rede neural composta por apenas uma camada escondida e que utiliza a função sigmoidal como função de ativação se resume a um modelo logístico.

Umas das propriedades importantes que a função sigmoidal carrega consigo é o fato de que sua derivada de primeira ordem é uma função que depende da própria função de entrada. Assim, se

$$\boldsymbol{\Phi}_i = \frac{1}{1+e^{-x}},$$

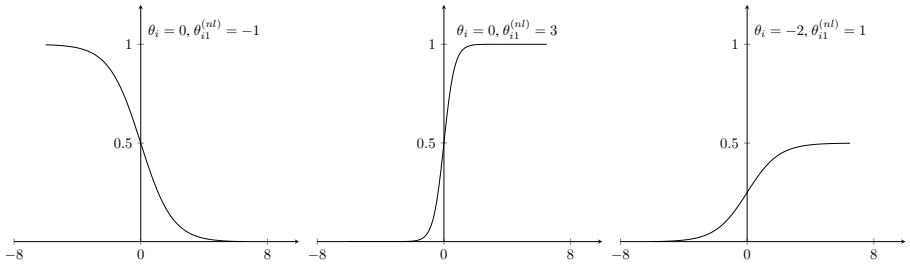
a sua derivada de primeira ordem é dada pela equação (2.7).

$$\begin{aligned}\frac{d\Phi_i}{dx} &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \frac{1+e^{-x}-1}{(1+e^{-x})^2} \\ &= \frac{1}{1+e^{-x}} - \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \Phi_i - \Phi_i^2 \\ &= \Phi_i(1-\Phi_i). \end{aligned} \tag{2.7}$$

Tanto a derivada da função sigmoidal quanto qualquer derivada de outra possível função de ativação, por exemplo a função tangente hiperbólica, é necessária em qualquer técnica de otimização baseada em gradiente aplicada para treinamento de uma rede MLP (NELLES, 2020), que é abordada na subseção 2.2.3.

A arquitetura MLP possui camadas escondidas responsáveis por lidar com a não linearidade envolvida no fenômeno que se deseja modelar. Cada peso projetado nas funções de ativação modifica suas formas, o que acarreta em ajustes distintos na medida em que a informação avança pela estrutura MLP. Na Figura 2.5 é mostrada a influência de pesos $\theta_i$ e $\theta_{ip}^{(nl)}$, escritos na equação (2.6) no formato de uma função de ativação sigmoidal.

Figura 2.5 – Influência dos pesos na função de ativação.




Fonte: Adaptado de Nelles (2020).

A quantidade de camadas escondidas e também de neurônios perceptron em cada uma delas não é necessariamente a mesma quantidade de variáveis explicativas de um determinado modelo. Desta forma, é conveniente reescrever a equação (2.6) na forma

$$\hat{y}_i = \sum_{i=0}^{M} w_i \Phi_i \left( \sum_{j=0}^{p} w_{ij} X_j \right) \quad , \tag{2.8}$$

em que $\Phi_0(\cdot) = 1$ e $X_0 = 1$, sendo $w_i$ o vetor de pesos (não linear) e $w_{ij}$ o vetor de pesos responsável pela não linearidade. Nelles (2020) afirma que o número total de parâmetros de uma rede neural MLP é $M(p+1)+M+1$.

Uma rede neural arquitetada na forma *multilayer perceptron* é considerada como um aproximador universal. Isto significa dizer que esta é uma estrutura capaz de convergir qualquer função com comportamento suave com algum grau de acurácia. Hammer e Gersmann (2003) afirmam que uma função $\mathscr{F}$ possui a capacidade de ser um aproximador universal se qualquer função contínua puder ser aproximada através de $\mathscr{F}$. Isto implica dizer que para qualquer conjunto compacto $C \subset \mathbb{R}^n$, qualquer função contínua $g : C \to \mathbb{R}$, e para qualquer $\varepsilon > 0$ existe uma função $f \in \mathscr{F}$ que pode ser encontrada tal que $|f(\vec{x}) - g(\vec{x})| \leq \varepsilon \forall \vec{x} \in C$. Redes neurais artificiais, modelos fuzzy e *suppport vectors machine* são considerados aproximadores universais (HAMMER; GERSMANN, 2003).

A precisão de tal aproximação no contexto de redes neurais artificiais está diretamente relacionada ao número de parâmetros do modelo. Entretanto, deve-se levar em consideração que à medida em que o número de parâmetros de uma rede MLP aumenta, mais complexo se torna o modelo e maior tende a ser a probabilidade de sobreajuste, do inglês *overfitting*.

### 2.2.3 A aprendizagem de uma rede neural

O processo de aprendizagem de uma rede neural está diretamente associado a otimização do vetor de pesos não lineares relacionados às camadas escondidas de uma rede, no caso MLP. Dentre as formas de aprendizagem, uma das mais utilizadas e responsáveis pela fama da modelagem via redes neurais artificiais é conhecida como *Backpropagation*.

A otimização deste vetor de pesos é dada pela derivação das funções de ativação, fato que permite Nelles (2020) considerar este processo de aprendizagem como um cálculo de vetores gradiente de uma rede neural. Portanto, o autor afirma que *backpropagation* é um processo idêntico à aplicação da regra da cadeia para o cálculo de derivadas, que posteriormente são usadas para a otimização dos vetores gradiente.

Sendo assim, esta é uma técnica iterativa que ajusta e modifica o vetor de pesos da(s) camada(s) escondida(s) até que se atinja a convergência entre os erros esperados e a saída atual (ZAJMI; AHMED; JAHARADAK, 2018).

Para o cálculo do vetor gradiente, as derivadas de $\hat{y}$ em relação ao i-ésimo peso da camada de saída são

$$\frac{\partial \hat{y}}{\partial w_i} = \Phi_i \qquad \text{para } \Phi_0 = 1 \ . \tag{2.9}$$

As derivadas da saída MLP em relação aos pesos das camadas escondidas são dados por

$$\frac{\partial \hat{y}}{\partial w_{ij}} = w_i \frac{dg(x)}{dx} X_j \qquad \text{para } X_0 = 1 \ , \tag{2.10}$$

para os pesos nas conexões entre a j-ésima entrada e o i-ésimo neurônio escondido. Na equação (2.10) a função $g(x)$ é uma função de ativação e a função sigmoidal pode ser utilizada. Desta forma, o resultado obtido na equação (2.7) transforma a equação (2.10) em

$$\frac{\partial \hat{y}}{\partial w_{ij}} = w_i(1 - \Phi_i^2) X_j \qquad \text{para } X_0 = 1 \ . \tag{2.11}$$

Uma vez que estas expressões de gradiente podem ser construídas pela propagação do erro do modelo de volta pela rede, o algoritmo é chamado de retropropagação (do inglês *backpropagation*) (NELLES, 2020). A aprendizagem via *backpropagation*, de acordo com Zajmi, Ahmed e Jaharadak (2018), é dada pelo algoritmo 1:

**Algoritmo 1** Aprendizagem de uma RNA via *backpropagation*.

1: A camada de entrada é apresentada ao vetor de entrada;
2: A camada de saída é apresentada ao conjunto de uma saída desejada;
3: Uma comparação entre os erros desejados e a saída real é feita toda vez que se atravessa a rede;
4: A comparação dos resultados determina mudanças de peso de acordo com as regras de aprendizagem.

*Backpropagation* é também um método conhecido por suas desvantagens. Entre elas, Cho, Conners e Araman (1991) afirmam que este é um método que demora para ser concluído e propõem, inclusive, um aprimoramento para tal.

Existem inúmeros estudos empregando redes neurais artificiais com o objetivo de detectar fraudes financeiras, em especial, fraudes de cartões de crédito. Dentre deles, destacam-se os trabalhos de Paasch (2008), Campus (2018), Awoyemi, Adetunmbi e Oluwadare (2017).

### 2.2.4 Modelagem fuzzy

#### 2.2.4.1 Conceitos elementares

A Teoria de Conjuntos Fuzzy foi criada inicialmente para tratar matematicamente termos linguísticos munidos de incerteza, que são naturais à linguagem humana. Portanto, termos como "aproximadamente", "em torno de", "muito", "pouco" podem ter significados diferentes para cada indivíduo e a tentativa de quantificar este tipo de incerteza foi a motivação para que Loft A. Zadeh iniciasse uma formalização matemática que pudesse modelar tal subjetividade matematicamente (BARROS; BASSANEZI; LODWICK, 2017).

O ponto de partida para que se obtenha uma formalização matemática de termos subjetivos é reconhecer que qualquer conjunto da matemática clássica pode ser caracterizado pela função característica, como mostra a Definição 1.

**Definição 1** *Seja $U$ um conjunto e $A$ um subconjunto de $U$. A função característica de $A$ é dada por*

$$\chi_A(x) = \begin{cases} 1, & se\ x \in A \\ 0, & se\ x \notin A \end{cases}. \tag{2.12}$$

Portanto, $\chi_A$ é uma função cujo domínio é $U$ e a imagem está contida no conjunto $\{0,1\}$.

**Exemplo 1** *Seja $A$ o conjunto de pessoas destras de uma determinada cidade. A função característica avaliada na amostra de moradores desta cidade é explicitada na Tabela 2.1.*

Tabela 2.1 – Exemplo função característica.

| $x$ | Qualidade | $\chi_A(x)$ |
|---|---|---|
| Morador I | Destro | 1 |
| Morador II | Canhoto | 0 |
| Morador III | Destro | 1 |

Fonte: Do autor (2021).

Avaliar a função característica em elementos que não possuem qualquer nível de incerteza, se torna uma tarefa trivial. Entretanto, não é possível aplicar a função características em termos linguísticos que possuem qualquer incerteza, antes de se definir claramente as características

de tal incerteza. Se no Exemplo 1 o conjunto $A$ fosse o conjunto das pessoas altas de uma determinada cidade, a aplicação de $\chi_A$ não seria possível, a menos que se definisse uma regra matemática para o termo "pessoas altas", dado que esta é uma expressão subjetiva.

A Teoria de Conjuntos Fuzzy, em qualquer situação (com ou sem incerteza) não trata o pertencimento de elementos a conjuntos como um resultado binário. Mas, trata do grau de pertencimento de elementos a tais conjuntos. A Definição 2 é a formalização do conceito de pertinência que foi contextualizado.

**Definição 2** (BARROS; BASSANEZI; LODWICK, 2017) *Seja U um espaço topológico. Um subconjunto fuzzy A de U é caracterizado por uma função de pertinência* $\mu_A : U \to [0,1]$, *em que* $\mu_A(x)$ *denota o grau em que o elemento x pertence ao subconjunto fuzzy A.*

Neste estudo, a notação $A(x)$ será utilizada para representar a função de pertinência, ao invés de $\mu_A(x)$. Se $A$ for um subconjunto clássico de $U$, sua função de pertinência é dada pela função característica $\chi_A(x)$. A Definição 2 permite a conclusão de que a função característica é um caso particular da matemática fuzzy, ou seja, subconjuntos clássicos, também conhecida na literatura como *crisp* são subconjuntos fuzzy em que $A(x) = 1$ (BARROS; BASSANEZI; LODWICK, 2017).

Já o grau de pertencimento, no universo fuzzy, é materializado através do conceito de $\alpha$-nível mostrado na Definição 3.

**Definição 3** *Os* $\alpha$*-níveis de um subconjunto fuzzy A são definidos pelo mapeamento do elemento conforme a pertinência*

$$[A]^{\alpha} = \begin{cases} \{x \in U; A(x) \geq \alpha\}, & 0 < \alpha \leq 1 \\ \\ \overline{\{x \in U; A(x) > 0\}}, & \alpha = 0 \end{cases}, \tag{2.13}$$

*em que* $\bar{X}$ *denota o fecho do subconjunto X de U.*

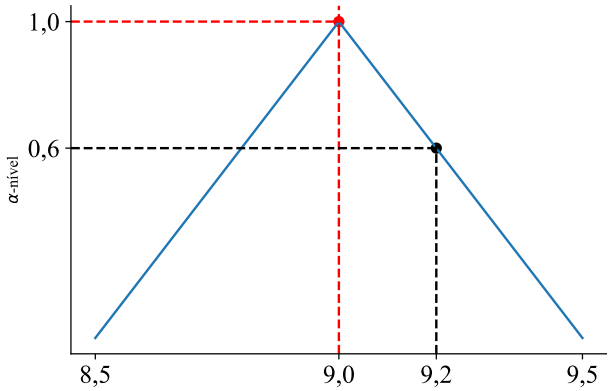
Assim, $\alpha$-níveis representam o grau com que um elemento $x$ pertence ao subconjunto $A$. Nota-se que a partir das Definições 2 e 3 o verbo "pertencer" não é mais um caso binário.

**Exemplo 2** *O número 9,2 é próximo de 9? Para responder a tal pergunta é necessário definir através de uma função de pertinência, o que significa "próximo de 9". Supondo que tal função seja dada pela equação (2.14),*

$$A(x) = \begin{cases} 0 & se\ x \leq 8,5 \\ \frac{x-8,5}{9-8,5} & se\ 8,5 < x \leq 9 \\ \frac{x-9,5}{9-9,5} & se\ 9 < x \leq 9,5 \\ 0 & se\ x \geq 9,5 \end{cases}, \tag{2.14}$$

*o problema pode ser representado graficamente através da Figura 2.6.*

Figura 2.6 – Função de pertinência triangular.




Fonte: Do autor (2021).

De acordo com a função de pertinência criada, o número real $9,2$ pertence ao conjunto dos números próximos de 9 com grau $0,6$. Além disso, os $\alpha$-níveis do subconjunto $A$, ou seja $[A]^{0,6}$ são dados por

$$[A]^{0,6} = [(9-8,5)0,6+8,5;(9-9,5)0,6+9,5] \tag{2.15}$$

$$[A]^{0,6} = [8,8;9,2]$$

A forma fechada de encontrar os $\alpha$-níveis deste exemplo dada por (2.15) é específica de um tipo de número fuzzy. Existem diversos tipos de números fuzzy, todos eles satisfazem as propriedades apresentadas na Definição 4.

**Definição 4** (BARROS; BASSANEZI; LODWICK, 2017) *Um subconjunto fuzzy A de* $\mathbb{R}$ *é um número fuzzy quando satisfaz as propriedades:*

*i) todos os* $\alpha$*-níveis de A são intervalos fechados e não vazios de* $\mathbb{R}$*.*

*ii) o conjunto* $\{x; A(x) > 0\}$ *é um conjunto limitado de* $\mathbb{R}$*.*

Pela Definição (4), os $\alpha$-níveis de um número fuzzy $A$ são representados por

$$[A]^{\alpha} = [a_{-}^{\alpha}, a_{+}^{\alpha}], \tag{2.16}$$

para todo $\alpha \in [0,1]$.

Um exemplo de número fuzzy é o número triangular fuzzy, cujo $\alpha$-níveis são dados por $[A]^{\alpha} = [(m - a_{-}^{0})\alpha + a_{-}^{0}, (m - a_{+}^{0})\alpha + a_{+}^{0}]$, para todo $\alpha \in [0,1]$, em que $[A]^{\alpha} = [a_{-}^{\alpha}, a_{+}^{\alpha}]$ e $\{m\} = [A]^{1}$ (um número real). Um número fuzzy triangular é denotado pela tripla $[a_{-}^{0}; m; a_{+}^{0}]$.

As operações aritméticas envolvendo números fuzzy estão intimamente ligadas às operações aritméticas intervalares. Considera-se por intervalo de reta um subconjunto não vazio, fechado e limitado de números reais da forma

$$[a,b] = \{x \in \mathbb{R}; a \leq x \leq b\}. \tag{2.17}$$

**Definição 5** (KLIR; YUAN, 1995) *Dados intervalos* $A = [a_1, a_2]$ *e* $B = [b_1, b_2]$ *e um número real* $\lambda$*, define-se:*

*i)* $A + B = [a_1 + b_1, a_2 + b_2]$.

*ii)* $A - B = [a_1 + b_2, a_2 + b_1]$.

*iii)* $\lambda A = [\lambda a_1, \lambda a_2]$ *se* $\lambda \geq 0$ *e* $[\lambda a_2, \lambda a_1]$ *se* $\lambda \leq 0$.

*iv)* $A \cdot B = [\min I, \max I]$ *em que* $I = \{a_1 b_1, a_1 b_2, a_2 b_1, a_2 b_2\}$.

*v)* $A/B = [a_1, a_2] \cdot \left[\frac{1}{b_2}, \frac{1}{b_1}\right]$.

**Proposição 1** (KLIR; YUAN, 1995) *Sejam os intervalos* $A = [a_1, a_2]$, $B = [b_1, b_2]$, $C = [c_1, c_2]$, $\tilde{0} = [0,0]$ *e* $\tilde{1} = [1,1]$. *Então*

*i)* $A + B = B + A$ *e* $A \cdot B = B \cdot A$.

*ii)* $(A + B) + C = A + (B + C)$ *e* $(A \cdot B) \cdot C = A \cdot (B \cdot C)$.

*iii)* $A = A + \tilde{0} = \tilde{0} + A$ *e* $A \cdot \tilde{1} = \tilde{1} \cdot A$.

*iv)* $A \cdot (B + C) \subseteq A \cdot B + A \cdot C$.

*v) Dados* $b \in B$ *e* $c \in C$ *tais que* $bc \geq 0$, *então* $A \cdot (B + C) = A \cdot B + A \cdot C$.

*vi)* $0 \in A - A$ *e* $1 \in A/A$.

As operações aritméticas para números fuzzy são definidas a partir do Princípio de Extensão para funções de duas variáveis. Por exemplo, para definir-se a soma de dois números fuzzy, estende-se a função adição de números reais, ou seja

$$\begin{aligned} + : \mathbb{R} \times \mathbb{R} &\longrightarrow \mathbb{R} \\ (x, y) &\longmapsto x + y \end{aligned} .$$

**Definição 6** (BARROS; BASSANEZI; LODWICK, 2017) *Sejam A e B números fuzzy e* $\lambda$ *um número real.*

*i) A soma dos números fuzzy A e B é o número fuzzy,* $A \oplus B$, *cuja função de pertinência é*

$$(A \oplus B)(z) = \begin{cases} \sup\limits_{\phi(z)} min\,[A(x), B(y)] & se\ \phi(z) \neq \emptyset \\ 0 & se\ \phi(z) = \emptyset \end{cases}, \tag{2.18}$$

*em que* $\phi(z) = \{(x, y); x + y = z\}$.

*ii) A multiplicação de* $\lambda$ *por A é o número fuzzy* $\lambda \odot A$, *cuja função de pertinência é*

$$(\lambda \odot A)(z) = \begin{cases} \sup\limits_{\{x; \lambda x = z\}} A(x) & se\ \lambda \neq 0 \\ \chi_{\{0\}}(z) & se\ \lambda = 0 \end{cases}, \tag{2.19}$$

*iii) A diferença dos números fuzzy A e B é o número fuzzy,* $A \ominus B$, *cuja função de pertinência é*

$$(A \ominus B)(z) = \begin{cases} \sup\limits_{\phi(z)} min\,[A(x), B(y)] & se\ \phi(z) \neq \emptyset \\ 0 & se\ \phi(z) = \emptyset \end{cases}, \tag{2.20}$$

*em que* $\phi(z) = \{(x, y); x - y = z\}$.

O Teorema 1 generaliza, através dos $\alpha$-níveis, as operações aritméticas entre números fuzzy. Além disso, ele garante que o resultado de operações aritméticas entre dois números fuzzy seja também um número fuzzy.

**Teorema 1** (BARROS; BASSANEZI; LODWICK, 2017) *Sejam $A, B \in \mathbb{R}_{\mathscr{F}}$. Os $\alpha$-níveis do conjunto fuzzy $A \otimes B$, para qualquer $\alpha \in [0,1]$, em que $\otimes$ denota qualquer operação aritmética intervalar clássica para intervalos, são dados por*

$$[A \otimes B]^{\alpha} = [A]^{\alpha} \otimes [B]^{\alpha}. \tag{2.21}$$

*Sejam A e B números fuzzy com $\alpha$-níveis dados por $[A]^{\alpha} = [a_{-}^{\alpha}, a_{+}^{\alpha}]$ e $[B]^{\alpha} = [b_{-}^{\alpha}, b_{+}^{\alpha}]$. Então, tem-se as seguintes propriedades:*

*a) A soma entre A e B é um número fuzzy $A \oplus B$ em que os $\alpha$-níveis são*

$$[A \oplus B]^{\alpha} = [A]^{\alpha} \oplus [B]^{\alpha} = [a_{-}^{\alpha} + b_{-}^{\alpha}, a_{+}^{\alpha} + b_{+}^{\alpha}]. \tag{2.22}$$

*b) A diferença entre A e B é um número fuzzy $A \ominus B$ em que os $\alpha$-níveis são*

$$[A \ominus B]^{\alpha} = [A]^{\alpha} \ominus [B]^{\alpha} = [a_{-}^{\alpha} - b_{+}^{\alpha}, a_{+}^{\alpha} - b_{-}^{\alpha}]. \tag{2.23}$$

*c) A multiplicação entre um número real k e um número fuzzy A, denotada por $k \odot A$ são os $\alpha$-níveis*

$$[k \odot A]^{\alpha} = k \odot [A]^{\alpha} = \begin{cases} [ka_{-}^{\alpha}, ka_{+}^{\alpha}], & k \geq 0 \\ [ka_{+}^{\alpha}, ka_{-}^{\alpha}], & k < 0 \end{cases}. \tag{2.24}$$

O Teorema 1, portanto, permite afirmar que realizar operações entre números fuzzy é realizar operações aritméticas intervalares com os $\alpha$-níveis de números fuzzy. A interpretação dos resultados da solução fuzzy obtida pode não ser suficiente para esgotar todas as possibilidades de aplicação ou de comparação com o caso clássico. Neste sentido, utiliza-se métodos para a remoção da incerteza atribuída ao problema via teoria de conjuntos fuzzy.

#### 2.2.4.2 Lógica fuzzy

Na teoria clássica de conjuntos o pertencimento ou o não pertencimento de um elemento $x$ a um determinado conjunto é uma questão booleana e, por isso, as tabelas verdade são suficientes para verificar a validade lógica de uma proposição composta. No caso clássico, afirmações verdadeiras possuem valor lógico 1 e 0 caso contrário. Barros, Bassanezi e Lodwick (2017) utilizam a notação $\wedge$ (mínimo) para a conjunção **e**; $\vee$ (máximo) para **ou**; $\neg$ para **negação** e $\Longrightarrow$ para **implicação**. No Exemplo 3 é mostrado como as tabelas verdade devem ser utilizadas na lógica clássica.

**Exemplo 3** *(BARROS; BASSANEZI; LODWICK, 2017) Sejam p e q duas proposições. As tabelas verdades para os conectivos são dadas pelas Tabelas 2.2, 2.3, 2.4 e 2.5.*

Tabela 2.2 – Tabela verdade de $\wedge$.

| $p$ | $q$ | $p \wedge q$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

Fonte: Barros, Bassanezi e Lodwick (2017).

Tabela 2.3 – Tabela verdade de $\vee$.

| $p$ | $q$ | $p \vee q$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

Fonte: Barros, Bassanezi e Lodwick (2017).

Tabela 2.4 – Tabela verdade de $\neg$.

| $p$ | $\neg$ p |
|---|---|
| 1 | 0 |
| 0 | 1 |

Fonte: Barros, Bassanezi e Lodwick (2017).

A lógica fuzzy estende estas afirmações para o caso não booleano e, para isso, os conectivos "e", "ou", "não" e "implicação" precisam ser ressignificados.

#### 2.2.4.3 Conectivos básicos da lógica fuzzy

A afirmação:

Tabela 2.5 – Tabela verdade de $\Longrightarrow$.

| $p$ | $q$ | $p \Longrightarrow q$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 0 | 0 | 1 |

Fonte: Barros, Bassanezi e Lodwick (2017).

"*Se o **saldo bancário** de um cliente **A** é de 9$ e o cliente é do tipo **B**, então a **capitalização do saldo** anual deste cliente é de 4%.*",

considera a inexistência de incerteza nos conjuntos **saldo bancário** e **capitalização do saldo**. Neste caso, que é determinístico, os conectivos $\wedge$ e $\Longrightarrow$ são suficientes para modelar as relações e conclusões do problema. Entretanto, estes conectivos não seriam adequados se houvesse algum tipo de incerteza nos conjuntos em questão. Por exemplo, se afirmação fosse:

"*Se o **o saldo bancário** de um cliente **A** é alto e o cliente é do tipo **B**, então a **capitalização do saldo** anual deste cliente é alta.*",

os conectivos $\wedge$ e $\Longrightarrow$ precisam ser estendidos para a lógica fuzzy para se tornarem adequados ao problema, pois neste caso, os conjuntos **saldo bancário** e **capitalização do saldo** se tornam subconjuntos fuzzy.

As extensões dos conectivos $\wedge, \vee, \neg$ e $\Longrightarrow$ são obtidas por meio de normas e conormas triangulares (BARROS; BASSANEZI; LODWICK, 2017). Nas Definições 7 e 8 são mostradas a formalização da extensão dos conectivos "e" e "ou".

**Definição 7** *(PEDRYCZ, 2021) O operador $\Delta : [0,1] \times [0,1] \to [0,1], \Delta(y,x) = x\Delta y$, é uma t-norma se satisfizer as condições:*

*i) Comutatividade: $x\Delta y = y\Delta x$*

*ii) Associatividade: $x\Delta(y\Delta w) = (x\Delta y)\Delta w = x\Delta y\Delta w$*

*iii) Monotonicidade: se $y \leq w$ então $x\Delta y \leq x\Delta w$*

*iv) Condição de fronteira: $x\Delta 1 = x\Delta 0 = 0$*

**Definição 8** *(PEDRYCZ, 2021) A t-conorma é uma operação binária $\nabla : [0,1] \times [0,1] \to [0,1], \nabla(y,x) = x\nabla y$, é uma t-norma se satisfizer as condições:*

*i) Comutatividade: $x\nabla y = y\nabla x$*

*ii) Associatividade:* $x\nabla(y\nabla w) = (x\nabla y)\nabla w = x\nabla y\nabla w$

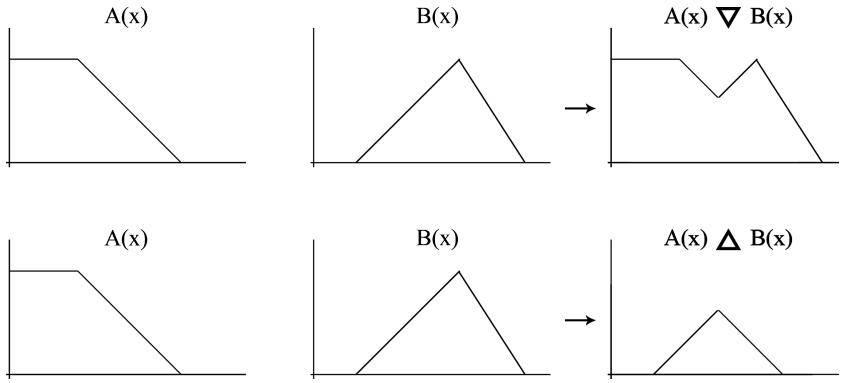
*iii) Monotonicidade: se* $y \leq w$ *então* $x\nabla y \leq x\nabla w$

*iv) Condição de fronteira:* $x\nabla 0 = x, x\nabla 1 = 1$

$x, y, w \in [0,1]$

Pedrycz (2021) afirma que os operadores $\Delta$ e $\nabla$ estendem para o caso fuzzy os operadores $\cap$ (interseção) e $\cup$ (união) de conjuntos clássicos, respectivamente. No casso fuzzy, $\Delta$ e $\nabla$ são avaliados em funções de pertinência para estabelecer suas respectivas relações. Na Figura 2.7 são mostradas graficamente tais operações no caso fuzzy.

Figura 2.7 – Representação gráfica de operações de união e interseção em funções de pertinência.




Fonte: Adaptado de Pedrycz (2021, p. 81)

Já nas Definições 9 e 10 são mostradas a formalização da extensão dos conectivos de negação e implicação.

**Definição 9** *(BARROS; BASSANEZI; LODWICK, 2017) Uma aplicação* $\eta : [0,1] \to [0,1]$ *é uma negação se satisfizer as condições:*

*i) Fronteiras:* $\eta(0) = 1$ *e* $\eta(1) = 0$*;*

*ii) Monotonicidade:* $\eta$ *é decrescente. Se* $\eta$ *for estritamente decrescente e além de i) a propriedade de involução* $\eta(\eta(x)) = x$ *também for verdade, então* $\eta$ *é chamada de negação forte.*

**Definição 10** *(BARROS; BASSANEZI; LODWICK, 2017) Um operador* $\Rightarrow: [0,1] \times [0,1] \to [0,1]$ *é uma implicação fuzzy se satisfizer as seguintes condições:*

*i) reproduz a tabela da implicação clássica;*

*ii) é decrescente na primeira variável, ou seja, para cada* $x \in [0,1]$ *tem-se* $(a \Rightarrow x) \leq (b \Rightarrow x)$ *se* $a \geq b$*;*

*iii) é crescente na segunda variável, ou seja, para cada* $x \in [0,1]$ *tem-se* $(x \Rightarrow a) \geq (x \Rightarrow b)$ *se* $a \geq b$*.*

Em modelagem estatística é elementar o conhecimento da natureza das variáveis aleatórias envolvidas, a fim de saber se elas são correlacionadas ou não, independentes ou não, aleatórias ou não, etc. Isto porque no processo de inferência envolvido em qualquer modelagem assume-se algumas premissas sobre as variáveis envolvidas no modelo e, o ferimento de alguma dessas pressuposições pode fazer com que o modelo ajustado não seja adequado. Em especial, o conceito de independência é uma premissa que garante a validade de inúmeros teoremas estatísticos utilizados no processo inferencial.

Ao tratar de números fuzzy e grau de pertinência, é natural o raciocínio de que grau de pertinência e probabilidade possuem uma interpretação aproximada. Esta é uma discussão que levou Zadeh a introduzir o conceito de não–interatividade possibilística, o qual tem estreita relação com a independência probabilística (BARROS; BASSANEZI; LODWICK, 2017). No processo de modelagem estatística se trabalha com variáveis aleatórias e, estas por sua vez, possuem distribuições de probabilidades (ou funções densidades de probabilidades). No caso fuzzy o que se tem são distribuições de possibilidades (ou de pertinências).

O que faz os termos “probabilidades” e “possibilidades” serem importantes na conceituação de independência e não-interatividade é o fato de que a construção de conceitos elementares como distribuição, condicionalidade, marginalidade, independência, etc, dependem de operações e conectivos distintos no caso clássico e fuzzy, como mostram Barros, Bassanezi e Lodwick (2017, p. 108). O conceito de probabilidade deve ser utilizado para quantificar incertezas na ocorrência de um evento estudado. Já o conceito de pertinência deve ser utilizados para mapear a incerteza na própria definição deste evento.

#### 2.2.4.4 Modelos fuzzy

Pedrycz (2021) faz a analogia de que modelos fuzzy são estruturas ou construções que utilizam subconjuntos fuzzy como blocos. Tais estruturas são, de maneira simplificada, compostas por três etapas: variáveis de entrada, fuzzificação/processamento e defuzzificação. As variáveis de entrada podem ser variáveis linguísticas munidas de incerteza, vetores construídos sob regras

da matemática clássica ou ainda os próprios subconjuntos fuzzy. Na etapa de processamento são estabelecidas as relações fuzzy com base na fuzzificação das variáveis de entrada. A forma com que estas relações são estabelecidas depende do tipo de modelagem fuzzy que está sendo feita, por exemplo, sistemas baseados em regras fuzzy e árvores de decisão fuzzy são tipos diferentes de modelos fuzzy em que as interações entre as variáveis fuzzificadas na etapa de processamento acontecem de forma distinta. Por fim, o resultado de um modelo fuzzy é um subconjunto fuzzy e, para que este resultado se torne um resultado aplicável (ou prático) a saída do modelo deve ser defuzzificada, através de alguns métodos de defuzzificação que são abordados na seção 2.2.4.7.

#### 2.2.4.5 Sistemas Baseados em Regras Fuzzy – FRBS

Sistemas Baseados em Regras Fuzzy (*Fuzzy Rules Based Systems – FRBS*) são modelos fuzzy construídos sob afirmações do tipo "se–então" que associa a conclusão à uma condição precedente. Em geral, as regras fuzzy possuem a forma:

"Se a condição é $A$ então a conclusão é $B$".

Pedrycz (2021) afirma que existem as chamadas regras funcionais, em que o objeto $B$ da sentença, ou a saída, é uma função local $f : x \in \mathbb{R}^n \to \mathbb{R}$.

A tradução de termos linguísticos munidos de incerteza é uma das tarefas que controladores humanos realizam para fazer interpretações e tomadas de decisão baseadas em regras do tipo "se–então" de forma cotidiana. A tentativa de reprodução desta estratégia é dada pelos controladores fuzzy (BARROS; BASSANEZI; LODWICK, 2017). A partir da ideia de raciocínio aproximado, os termos linguísticos devem ser traduzidos por meio de uma base de regras fuzzy, para que se obtenha relações fuzzy, que por sua vez produzirão a saída para cada entrada.

A construção de uma base de regras deve levar em conta alguns aspectos para que o modelo não se torne um modelo para casos específicos. A qualidade das regras pode ser mensurada através da análise da especificidade de cada condição e conclusão Pedrycz (2021). Regras cujas condições e conclusões são muito específicas são regras de baixa qualidade, pois sua aplicabilidade é limitada a casos particulares. Por exemplo, a regra

"Se $x$ é 5,2 então $y$ é 4,3." (R.1)

é uma regra de baixa qualidade porque tanto a condição quanto a conclusão dizem respeito a casos particulares, o que implica em uma aplicabilidade baixa. Um outro exemplo de regra com baixa qualidade seria

"Se $x$ é 5,2 então $y$ *mais ou menos puro*." (R.2)

A regra R.2 ainda continua sendo aplicável apenas para um único caso numérico de entrada e, por isso, uma regra de qualidade baixa. Já uma regra que generaliza ao máximo as condições e especifica conclusões tende a ser uma regra de boa qualidade (PEDRYCZ, 2021). A regra R.3 é um exemplo de regra com boa qualidade.

"Se $x$ é mais ou menos velho então $y$ está em torno de 4,3." (R.3)

Pedrycz (2021, p. 204) afirma que as métricas utilizadas para avaliar a qualidade das regras são os conceitos de completude e inconsistência. Além disso, estes conceitos também são necessários para quando as regras são estabelecidas por especialistas. Pode-se afirmar que uma base de regras atende ao conceito de completude se suas regras cobrem todos as situações possíveis que as entradas possam gerar. A definição de situações possíveis é análoga ao conceito de espaço amostral em teoria probabilística. Por exemplo, as regras "se $x$ é $A_i$ então $y$ é $B_i$" são incompletas se existirem entradas $x$ as quais nenhuma destas regras atingem. A inconsistência, por sua vez, é uma propriedade que caracteriza uma situação em que duas ou mais regras possuem condições próximas (ou parecidas), entretanto suas conclusões são absolutamente distintas. Por exemplo,

"Se $x$ é baixo então $y$ é alto." (R.4)

"Se $x$ é muito baixo então $y$ é muito baixo." (R.5)

são regras inconsistentes, pois o fato de envolver subconjuntos fuzzy polarizados (alto e baixo) faz com que a tomada de decisão se torne um processo confuso do ponto de vista lógico.

#### 2.2.4.6 Métodos de inferência fuzzy

Um dos componentes de um controlador fuzzy é o módulo de inferência fuzzy. Neste módulo é onde, especificamente, se traduz matematicamente os termos linguísticos munidos de incerteza através da lógica fuzzy. Dessa forma, os conectivos vistos na seção 2.2.4.3 são utilizados para estabelecer relações e construir subconjuntos fuzzy, de acordo com alguma base de regras previamente estabelecida.

O método de inferência Mamdani propõe uma relação fuzzy binária para modelar matematicamente uma base de regras e, para tanto, este método é baseado na regra de composição de inferência em que se obtém o máximo dos mínimos (max-min) através da t-norma $\wedge$ (mínimo) para modelar o conectivo lógico "e", e da t-conorma $\vee$ (máximo) para modelar o conectivo lógico "ou" (BARROS; BASSANEZI; LODWICK, 2017). A relação ou método de inferência de Mamdani é dada pela Definição 11.

**Definição 11** *(BARROS; BASSANEZI; LODWICK, 2017) A relação fuzzy $\mathscr{M}$ é o subconjunto fuzzy de $X \times U$ cuja função de pertinência é dada por*

$$\varphi_{\mathscr{M}}(x,u) = \max_{1\leq j\leq r}(\varphi_{R_j}(x,u)) = \max_{1\leq j\leq r}[\varphi_{A_j}(x) \wedge \varphi_{B_j}(u)], \tag{2.25}$$

*em que $r$ é o número de regras que compõem a base de regras e, $A_j$ e $B_j$ são os subconjuntos fuzzy da regra $j$. $\varphi_{A_j}$ e $\varphi_{B_j}$ são funções de pertinência que quando avaliadas em $x$ e $u$ devem ser interpretadas como sendo o grau com que estes elementos pertencem a seus respectivos conjuntos. Em suma, a relação $\mathscr{M}$ é a união dos produtos cartesianos fuzzy entre os antecedentes e os consequentes de cada regra.*

Além do método Mamdani, existe também o método de inferência Takagi-Sugeno-Kang (TSK), cujas principais diferenças em relação ao primeiro é a estrutura da conclusão de cada regra e no procedimento de defuzzificação para obter a saída geral do sistema. Com o método TSK a conclusão de cada regra da base é geralmente uma função avaliada nos valores de entrada. Na base base de regras:

"Se $x_1$ é $A_{11}$ e $x_2$ é $A_{12}$ ... e $x_n$ é $A_{1n}$ então $u_1 = g_1(x_1, x_2, \ldots, x_n)$"

ou

"Se $x_1$ é $A_{21}$ e $x_2$ é $A_{22}$ ... e $x_n$ é $A_{2n}$ então $u_2 = g_2(x_1, x_2, \ldots, x_n)$"

ou

$\vdots$

"Se $x_1$ é $A_{r1}$ e $x_2$ é $A_{r2}$ ... e $x_n$ é $A_{rn}$ então $u_r = g_r(x_1, x_2, \ldots, x_n)$"

existe um número $r$ de regras que é equivalente a quantidade de funções $g(\cdot)$. Barros, Bassanezi e Lodwick (2017) afirmam que a saída geral do método é dada pela equação (2.26).

$$u = \frac{\sum_{j=1}^{r} w_j \cdot g_j(x_1, x_2, \ldots, x_n)}{\sum_{j=1}^{r} w_j} = \frac{\sum_{j=1}^{r} w_j \cdot u_j}{\sum_{j=1}^{r} w_j} \tag{2.26}$$

em que os pesos $w_j$ são dados por $w_j = \varphi_{A_{j1}}(x_1) \Delta \varphi_{A_{j2}}(x_2) \ldots \Delta \varphi_{A_{jn}}(x_n)$ e $\Delta$ é uma t-norma. A interpretação para $w_j$ é a influência de cada regra $j$ para a saída geral. Além disso, é comum o uso de t-normas como sendo o produto e o mínimo. No caso do uso do operador mínimo como t-norma, os pesos $w_j$ são obtido pela equação (2.27).

$$w_i = min[\varphi_{A_{i1}}(x_1), \varphi_{A_{i2}}(x_2), \ldots, \varphi_{A_{in}}(x_n)] \tag{2.27}$$

para $i \in [1, n]$ em que $n$ é o quantidade de variáveis de entrada do sistema.

#### 2.2.4.7 Métodos de defuzzificação

A defuzzificação faz com que os resultados obtidos por meio de algum método de inferência fuzzy sejam trazidos para um valor real (*crisp*). Uma interpretação válida é encarar valores defuzzificados como medidas representativas, assim como acontece com as funções média e

variância em teoria probabilística. Existem diversos métodos de defuzzificação. Dentre todos eles, destaca-se o centróide.

**Definição 12** (BEDE; GAL, 2004) *O centróide do subconjunto fuzzy $A \in \mathscr{F}(U)$ é o número real*

$$COG(A) = \frac{\int_W xA(x)dx}{\int_W A(x)dx}, \tag{2.28}$$

*em que $W = supp(A)$.*

É possível estabelecer uma certa semelhança entre o cálculo do centróide e o cálculo da esperança matemática de uma função densidade de probabilidade. Um outro nome que é dado para este método de defuzificação é Centro de Gravidade, pois o seu resultado é uma medida central (representativa) de valores que são defuzzificados.

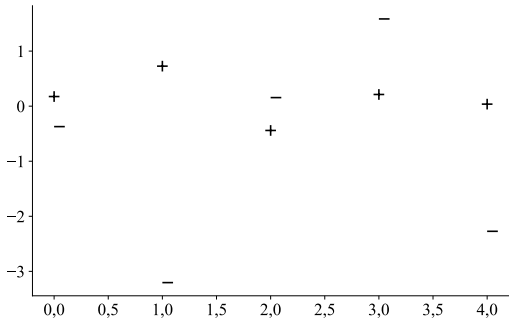
### 2.2.5 *Support Vector Machine* – SVM

*Support Vector Machine* (ou em português vetores de suporte de máquinas) corresponde a um conjunto de métodos de aprendizagem supervisionada usados para tarefas de classificação, regressão e detecção de *outliers*. Esta é um técnica computacional que a partir de dados de treinamento encontra hiperplanos que melhor segregam as classes dos dados em questão. Bishop (2006) afirma que em vetores de suporte de máquinas, o limite de decisão é escolhido para ser aquele para o qual a margem é maximizada. A solução de margem máxima pode ser motivada usando a teoria de aprendizagem computacional, também conhecida como teoria de aprendizagem estatística.

SVM são categorizados na literatura (SUYKENS; SIGNORETTO; ARGYRIOU, 2015, p. 7) como classificadores lineares que utilizam hiperplanos que particionam um espaço $r-$dimensional, de modo que cada observação $x_i$ rotulada com informação binária esteja em lados distintos do plano. O que faz os modelos SVM alcançarem resultados satisfatórios é o fato de que a escolha do hiperplano que melhor segrega as classes das observações é aquela com a melhor margem possível. Suykens, Signoretto e Argyriou (2015) definem margem como a menor distância para o hiperplano entre todas as observações.

A segregação de espaços bidimensionais ou multidimensionais para tarefas classificação através de funções lineares pode não ser possível, dependendo do comportamento dos dados disponíveis. Na Figura 2.8 é mostrado um exemplo de situação deste tipo.

Figura 2.8 – Exemplo de categorias linearmente não separáveis.




Fonte: Do autor (2021).

A questão fundamental por trás da solução de problemas exemplificados pela Figura 2.8 é a mudança de perspectiva. De fato, a separação das categorias expostas no gráfico da Figura 2.8 através de uma reta não é possível. Entretanto, a adição de uma nova dimensão faz com que exista a concepção de profundidade e, neste sentido, existe a possibilidade de que tais classes sejam separadas ainda de forma linear, não através de retas, mas de planos. Portanto, encontrar planos ou hiperplanos (no caso da necessidade de incorporação de dimensões mais altas ao problema) é a mudança de perspetiva necessária para que a segregação de classes continue sendo um problema de classificação linear.

Seja qual for a dimensão necessária para encontrar uma distância que separa as classes em questão, tal distância pode ser encontrada através de vetores suporte. Seja $\vec{u}$ um vetor cuja dimensão delimita uma fonteira entre as classes com margem previamente estabelecida e $\vec{w}$ um vetor perpendicular à mediana da margem que segrega as classes. O produto escalar entre $\vec{u}$ e $\vec{w}$ cria decisões de fronteira utilizadas para a classificação de cada observação disponível, de acordo com a equação (2.29).

$$\begin{cases} \vec{u}\cdot\vec{w}+b \geq 0, & \text{então a classe é ``+''} \\ \vec{u}\cdot\vec{w}+b < 0, & \text{então a classe é ``}-\text{''} \end{cases}, \tag{2.29}$$

em que $b$ é constante. Existem muitos vetores perpendiculares à reta (ou ao plano) que poderiam ser candidatos a $\vec{w}$. Por isso, é necessário impor as condições da equação (2.30) para a escolha deste vetor.

$$\begin{cases} \vec{w} \cdot \vec{x}_+ + b \geq 1 \\ \vec{w} \cdot \vec{x}_- + b \leq -1 \end{cases} , \tag{2.30}$$

em que $\vec{x}_+$ e $\vec{x}_-$ são vetores de observações de cada classe. Conforme Terzic et al. (2013, p. 45), é conveniente introduzir no problema (2.30) uma variável $y_i$, tal que

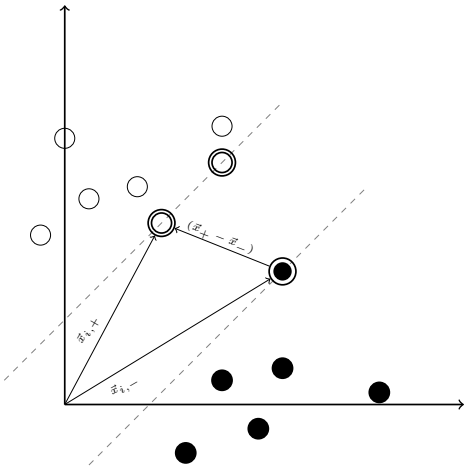
$$y_i = \begin{cases} +1, & \text{classe “+”} \\ -1, & \text{classe “−”} \end{cases} , \tag{2.31}$$

de forma que as condições da equação (2.30) se tornem apenas

$$y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 \ . \tag{2.32}$$

Como já mencionado, é possível que haja mais de uma reta, plano ou hiperplano que segrega as classes “+” e “−”. Portanto, deve ser escolhido aquele em que a margem, ou a distância criada entre cada classe é máxima. A margem é definida como a distância entre o par de observações localizados na fronteira de cada classe, conforme é mostrado na Figura 2.9.

Figura 2.9 – Visualização geométrica do cálculo da margem entre as classes.



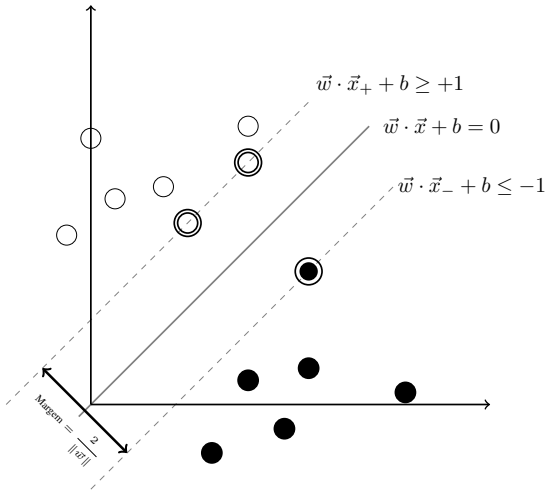
Fonte: Do autor (2021).

O cálculo da margem é dado por

$$\begin{aligned} Margem &= (\vec{x}_+ - \vec{x}_-) \cdot \frac{\vec{w}}{\|\vec{w}\|} \\ &= (1 - b - (-1 - b)) \cdot \frac{\vec{w}}{\|\vec{w}\|} \\ &= \frac{2}{\|\vec{w}\|} \qquad (2.33) \\ &= \frac{1}{2}\|\vec{w}\|^2 \ . \qquad (2.34) \end{aligned}$$

É matematicamente conveniente reescrever a equação (2.33) na forma da equação (2.34) uma vez que esta simplifica posteriormente o processo de maximização (TERZIC et al., 2013, p. 45) . Na Figura 2.10 é mostrada geometricamente a forma como os vetores $\vec{w}$ e $\vec{x}$ se relacionam para realizar a classificação desejada.

Figura 2.10 – Visualização gráfica das condições que definem $\vec{w}$.




Fonte: Do autor (2021).

Na Figura 2.10 fica evidente que o separador linear que melhor segrega as classes, ou seja, maximiza a distância entre elas, está posicionado de acordo com a mediana da distância entre tais classes.

Sendo assim, utiliza-se os multiplicadores de Lagrange para maximizar a distância dada pela equação (2.34). Ou seja,

$$L = \frac{1}{2}\|\vec{w}\|^2 - \sum_i \alpha_i \left[ y_i(\vec{w} \cdot \vec{x}_i + b) - 1 \right] \ , \tag{2.35}$$

em que $\alpha_i$ é o multiplicador de cada condição $i$. Encontrar o máximo da equação (2.35) se limita a encontrar suas derivadas parciais com relação a cada parâmetro de interesse, no caso $\vec{w}$ e $b$, e igualá-las à zero.

$$\frac{\partial L}{\partial \vec{w}} = \vec{w} - \sum_i \alpha_i y_i \vec{x}_i = 0 \implies \vec{w} = \sum_i \alpha_i y_i \vec{x}_i \ , \tag{2.36}$$

$$\frac{\partial L}{\partial b} = -\sum_i \alpha_i y_i = 0 \implies \sum_i \alpha_i y_i = 0 \ . \tag{2.37}$$

Utilizando estes resultados em (2.35), tem-se

$$L = \frac{1}{2}\left(\sum_i \alpha_i y_i \vec{x}_i\right)\left(\sum_j \alpha_j y_j \vec{x}_j\right) - \sum_i \alpha_i y_i \vec{x}_i \left(\sum_j \alpha_j y_j \vec{x}_j\right) - \sum_i \alpha_i y_i b + \sum_i \alpha_i \ . \tag{2.38}$$

Nota-se que na equação (2.38) foi introduzido o indexador $j$ que representa o vetor de observações pertencentes à classe contrária em relação a classe do vetor de observações indexado com $i$. Em outras palavras, o indexador $i$ pode se referir a i-ésima observação pertencente ao vetor $\vec{x}_+$ e, o indexador $j$ pode se referir a j-ésima observação pertencente ao vetor $\vec{x}_-$; ou vice-versa. Patle e Chouhan (2013) apresenta a equação (2.38) escrita na forma

$$L = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j \ . \tag{2.39}$$

A equação (2.39) representa a equação que maximiza a margem estabelecida entre $\vec{x}_+$ e $\vec{x}_-$ e o hiperplano, o que deve ser realizado através de métodos numéricos. Entretanto, é importante notar que tal maximização depende unicamente do produto escalar entre estes vetores. Isto implica dizer que as regras de decisão expostas na equação (2.29) podem ser reescritas na forma

$$\begin{cases} \sum_i \alpha_i y_i \vec{x}_i \cdot \vec{w} + b \geq 0, & \text{então a classe é ``+''} \\ \\ \sum_i \alpha_i y_i \vec{x}_i \cdot \vec{w} + b < 0, & \text{então a classe é ``$-$''} \end{cases} . \tag{2.40}$$

Existem funções que projetam o produto escalar $\vec{x}_i \cdot \vec{w}$ em espaços de maiores dimensões, e que por sua vez, são maximizadas a fim de alcançar a classificação desejada. Elas são chamadas de funções *kernel*, que fornecem uma solução para este problema, adicionando uma dimensão adicional aos dados (NOBLE, 2006). As funções *kernel* permitem que modelos SVM trabalhem os dados em dimensões maiores do que as que eles originalmente pertencem, tornando possível uma separação linear das classes dos dados. Toda a base teórica que permite a separação linear

de classes em altas dimensões é conhecida na literatura como substituição kernel ou truque matemático kernel (NOBLE, 2006; BISHOP, 2006; PATLE; CHOUHAN, 2013). As funções *kernel* são comumente escritas na forma

$$K(x_i, x_j) = \phi(\vec{x}_i) \cdot \phi(\vec{x}_j) \ . \tag{2.41}$$

De acordo com Patle e Chouhan (2013) algumas funções *kernel* são:

- Função *kernel* linear: $K(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j^T$;
- Função *kernel* polinomial: $K(\vec{x}_i, \vec{x}_j) = (1 + \vec{x}_i \cdot \vec{x}_j^T)^d$, em que $d$ é o grau da função *kernel*;
- Função gaussiana: $K(\vec{x}, \vec{x}_j) = e^{-\sigma \left\| \vec{x} - \vec{x}_j \right\|^2}$, para $\sigma > 0$;
- Função sigmoidal: $K(\vec{x}, \vec{x}_j) = tanh\left(\gamma \vec{x}_i^T \vec{x}_j + r\right)$, em que $\gamma$ e $r$ são parâmetros da função *kernel*.

Patle e Chouhan (2013) afirmam que a escolha da função *kernel* não é fixa e depende de cada aplicação. Bhattacharyya et al. (2011), por exemplo, utilizaram a função *kernel* gaussiana na identificação de fraudes em cartões de crédito.

### 2.2.6 *Random Forest*

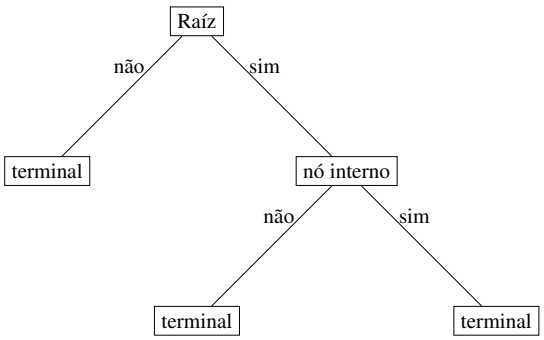
Uma árvore de decisão é a menor unidade de um modelo *Random Forest* (RF) e a sua estrutura depende da particularidade de cada aplicação. De maneira geral, Zhang e Singer (2010) afirmam que a estrutura de uma árvore de decisão é construída a partir de camadas. Na primeira camada, encontra-se a raíz da árvore, local de onde se inicia o processo de particionamento dos dados. Na raíz de uma árvore aleatória encontram-se os dados amostrais de treinamento ou mesmo o conjunto total dos dados; portanto, qualquer estrutura subsequente possui uma menor quantidade de dados (ZHANG; SINGER, 2010). Cada regra (momento de decisão ou nó) é criada de forma que, à medida que uma amostra atravessa a árvore, ela é particionada em subconjuntos até que seja finalmente classificada em um subgrupo mutuamente exclusivo (WEST; BHATTACHARYA, 2016).

A quantidade de camadas que uma árvore possui depende da quantidade de conclusões (respostas) binárias que ela realiza. Por exemplo, da raiz de uma árvore aleatória podem surgir outras duas possíveis estruturas: um nó (ou descendente) interno ou um terminal, cada uma ocupando uma posição da resposta binária atingida. Zhang e Singer (2010) afirmam que se uma

conclusão der origem à outros nós internos, o particionamento continuará ocorrendo até que o objetivo da árvore aleatória seja atingido, ou seja, classificação ou regressão e, dessa forma, outras camadas vão surgindo. Entretanto, um terminal não pode dar origem a qualquer outro elemento ou camada, pois ele representa a própria conclusão.

Na Figura 2.11 é mostrado uma estrutura de árvore de decisão com três camadas.

Figura 2.11 – Exemplo de árvore de decisão.

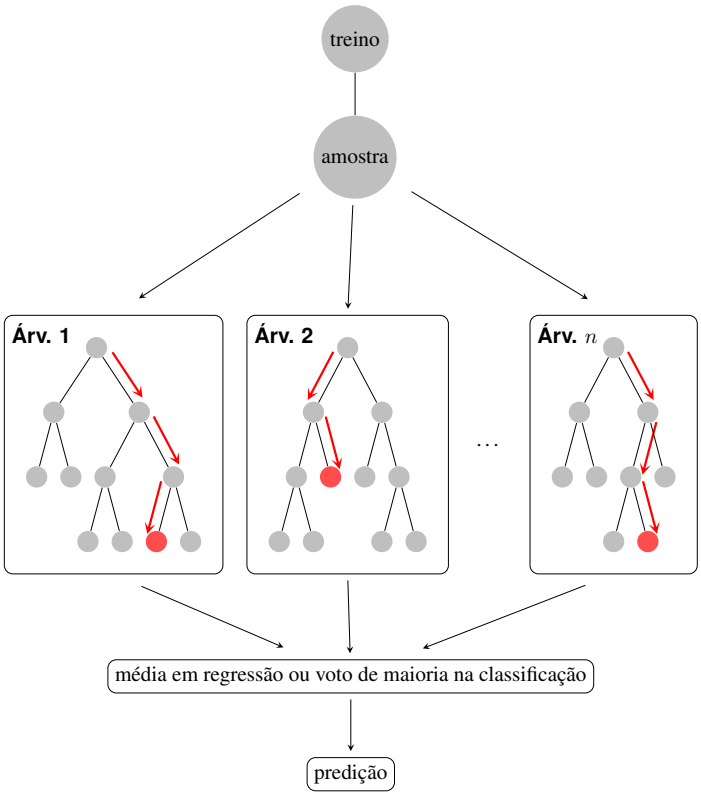



Fonte: Do autor (2021).

Com o avanço nos estudos das áreas de genética e ciência da computação, dados com altas dimensões e poucas observações se tornaram comuns, o que é conhecido como "problemas de $p$ grande e $n$ pequeno" (ZHANG; SINGER, 2010, p. 79). Dessa forma, a fim de garantir a parcimônia de um processo de modelagem e superar outras limitações oriundas da própria estrutura simples das árvores de decisão, Zhang e Singer (2010) afirmam que o método de RF surge como uma solução ideal. Os modelos RF são construídos através de centenas ou milhares de árvores de decisão. Sozinhas, cada árvore não representa um bom modelo, mas ao combiná-las tem-se um ganho de valor (ZHANG; SINGER, 2010).

As florestas aleatórias são um conjunto de árvores de decisão aleatórias que podem ser utilizadas em regressão, classificação ou mesmo em ambas as tarefas (GALL; RAZAVI; GOOL, 2012). Na Figura 2.12 é apresentado um esquema geral de funcionamento de florestas aleatórias.

Figura 2.12 – Esquema geral de estrutura de modelos *random forest*.




Fonte: Do autor (2021).

Para um vetor aleatório $p-$dimensional $X$, que representa as variáveis de entrada do modelo, e uma variável aleatória $Y$ dependente de $X$, assume-se a existência de uma distribuição conjunta $P_{XY}(x,y)$. Cutler, Cutler e Stevens (2012) afirmam que o objetivo da modelagem de uma floresta aleatória é encontrar uma função de predição $f(x)$ para $Y$, tal que esta é utilizada para determinar uma função perda $L(Y,f(x))$, em que se objetiva minimizar o valor esperado da perda, ou seja,

$$E_{XY}[L(Y,f(x))]. \tag{2.42}$$

A escolha de $L(Y,f(x))$ depende, obviamente, da natureza do problema estudado. Em tarefas de regressão, por exemplo, é comum utilizar o quadrado do erro como função perda, ou seja,

$$L(Y,f(x)) = (Y-f(x))^2 \ , \tag{2.43}$$

que mede a distância entre os valores preditos por $f(x)$ e os valores observados $Y$. Portanto, a minimização de $E_{XY}[L(Y, f(x))]$ resulta na esperança condicional $f(x) = E(Y|X = x)$.

No caso de tarefas de classificação, a função perda é dada por

$$L(Y, f(x)) = \begin{cases} 0 & \text{se } Y = f(x) \\ 1 & \text{se } Y \neq f(x) \end{cases} . \tag{2.44}$$

Neste caso, a minimização de $E_{XY}[L(Y, f(x))]$ resulta em

$$f(x) = \arg\max_{y \in Y} P(Y = y|X = x) \ , \tag{2.45}$$

também conhecida como regra de Bayes (CUTLER; CUTLER; STEVENS, 2012).

No caso de tarefas de classificação a função $f$ é construída através da combinação dos chamados *base leaners*, denotados por funções $h_1(x), h_2(x), \ldots, h_J(x)$. A combinação de cada *base learner* através da equação (2.46) faz com que o método de florestas aleatórias se torne um método *ensemble*. Estes métodos constroem vários modelos de machine learning, utilizando o resultado de cada para compor um único resultado.

$$f(x) = \arg\max_{y \in Y} \sum_{j=1}^{J} I(y = h_j(x)) \ . \tag{2.46}$$

No caso de regressão, os *base learners* são a média

$$f(x) = \frac{1}{J} \sum_{j=1}^{J} h_j(x) \ . \tag{2.47}$$

Nos estudos levantados por West e Bhattacharya (2016), Ngai et al. (2011) árvores de decisão são métodos que se mostram relevantes em pesquisas que buscam a classificação de fraudes financeiras, como em cartões de crédito.

### 2.2.7 Extreme Gradient Boosting

*Extreme Gradient Boosting* se refere a uma combinação de métodos de *machine learning* que trabalham juntos para atingir um objetivo comum (WADE, 2020). XGBoost é uma abreviação para o termo *Extreme Gradient Boosting*, porém, apenas por conveniência de notação,

a abreviação XGB é utilizada neste trabalho para se referir a este modelo. A partir da ideia de minimização de erros através de vetores gradiente, XGB é um método conveniente para tarefas de classificação e regressão quando se tem disponíveis grandes bancos de dados, uma vez que consegue atingir resultados mais robustos em termos de precisão e erro em relação a outros modelos (WADE, 2020).

Dado que o modelo XGB é uma otimização de modelos *Gradient Boosted Machine* (GBM), nesta subseção são apresentados os principais fundamentos deste.

Cada método que compõe um conjunto de *gradient boosting* é chamado de *base learners* (WADE, 2020). *Random Forest*, modelos logísticos, redes neurais, dentre outros, são exemplos de tais componentes e, o que faz o *gradient boosting* um método eficiente, é a sua forma de aprendizado com os dados. Por exemplo, ao utilizar RF para tarefa de predição ou classificação, o *gradient boosting* aprende com os erros de cada árvore individualmente, e então constrói novas árvores de decisão baseadas nos erros obtidos anteriormente (WADE, 2020). A principal diferença entre os modelos RF e GBM é que, enquanto em RF, as árvores são construídas independentemente umas das outras, o GBM adiciona uma nova árvore para complementar as já construídas (PAN, 2018).

Quando utilizados em tarefas de classificação, os modelos GBM possuem muito em comum com o modelo logístico. Isso porque durante o processo de predição, a relação entre a função sigmoidal e o logaritmo da chance de sucesso pode ser usada para minimizar os erros. Os modelos GBM normalmente utilizam a estrutura de árvores de regressão para fazer predições sobre os resíduos do modelo e, a partir de um processo iterativo, estes são minimizados.

A partir de um conjunto de dados de treino $\{(x_i, y_i)\}_{i=1}^{n}$ o processo de modelagem depende de uma constante inicial que será usada para calcular os primeiros resíduos, além de uma função perda. Esta é derivada do logaritmo da verossimilhança dos dados observados, dada uma probabilidade $\pi$, ou seja,

$$ln(y|\pi) = \sum_{i=1}^{N} y_i ln(\pi) + (1 - y_i) ln(1 - \pi) \ . \tag{2.48}$$

Quanto melhor a predição, maior é a verossimilhança $ln(y|\pi)$, o que justifica a busca da maximização da verossimilhança durante o processo de estimação de parâmetros no modelo logístico. Desta forma, para usar $ln(y|\pi)$ como a função perda, onde valores pequenos representam melho-

res ajustes de modelos, é necessária a multiplicação da verossimilhança por $-1$ (FRIEDMAN, 2002). Assim,

$$\begin{aligned}
-ln(y|\pi) &= -\sum_{i=1}^{N} y_i ln(\pi) + (1-y_i) ln(1-\pi) \\
&= -\sum_{i=1}^{N} y_i ln(\pi) + ln(1-\pi) - y_i ln(1-\pi) \\
&= -\sum_{i=1}^{N} y_i (ln(\pi) - ln(1-\pi)) + ln(1-\pi) \\
&= \sum_{i=1}^{N} -y_i \left[ ln\left(\frac{\pi}{1-\pi}\right)\right] - ln(1-\pi) \\
&= \sum_{i=1}^{N} -y_i \left[ ln\left(\frac{\pi}{1-\pi}\right)\right] + ln\left(1+exp\left\{ ln\left(\frac{\pi}{1-\pi}\right)\right\}\right) \quad . \qquad (2.49)
\end{aligned}$$

Por conveniência de notação, o quociente $\frac{\pi}{1-\pi}$, que representa a razão de chances, será substituído por $\gamma$. Portanto, a função perda é dada pela equação (2.50).

$$L(y,\gamma) = \sum_{i=1}^{n} -y_i ln(\gamma) + ln(1+e^{ln(\gamma)}) \qquad (2.50)$$

Uma vez que se tem a função perda definida, a constante inicial necessária para dar entrada no processo de predicação pode ser obtida. Tal constante é dada pela equação (2.51).

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i,\gamma) \quad . \qquad (2.51)$$

O valor de $F_0(x)$ obtido é usado como a folha inicial da árvore a ser construída. A partir dela, novas predições são realizadas e novos resíduos são computados a partir dos resíduos obtidos anteriormente. De acordo com Friedman (2002), estes são chamados de pseudo resíduos, calculados pela equação (2.52).

$$r_{i,m} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right] \quad , \qquad (2.52)$$

em que $r_{i,m}$ é o resíduo da amostra $i$ e $m$ é a árvore que está sendo construída. A cada iteração, novos resíduos são obtidos e, dessa forma, há uma atualização dos valores preditos. Em tal atualização, é incluído um termo que penaliza a chance de sucesso obtida anteriormente, chamado de taxa de aprendizado. A atualização dos valores preditos é dada pela equação (2.53) (FRIEDMAN, 2002).

$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm}) \ , \tag{2.53}$$

em que $\gamma_{jm} I(x \in R_{jm})$ se referem aos valores preditos na árvore anterior, $J_m$ se refere a possível iteração que deve ser feita quando a árvore produzida possui mais de uma folha e $\nu$ se refere à taxa de aprendizado. O parâmetro $\nu$ é diretamente relacionado com a probabilidade de *overfitting*, que está definido na subseção 2.2.8. Friedman (2002) afirma que a modelagem GBM pode ser sumarizada pelo Algoritmo 2.

**Algoritmo 2** Etapas de um processo de modelagem *Gradient Boosting Machine*.

1: Iniciar o modelo com a constante dada por $F_0(x) = \arg\min_\gamma \sum_{i=1}^{n} L(y_i, \gamma)$.
2: Para $m = 1$ até $M$:
   a) Calcular $r_{i,m} = -\left[\dfrac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$, para $i = 1, \ldots, n$
   b) Ajustar uma árvore de regressão para os resíduos $r_{i,m}$ e criar os terminais de cada árvore $R_{j,m}$, para $j = 1, \ldots, J_m$
   c) Para $j = 1, \ldots, J_m$ calcular $\gamma_{j,m} = \arg\min_\gamma \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$
   d) Atualizar o valor $F_m(x) = F_{m-1}(x) = \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$

O processo continua até que a profundidade máxima da árvore definida anteriormente seja atingida (parâmetro $M$), ou quando os erros não puderem ser mais minimizados.

A partir do processo de modelagem *Gradient Boosting Machine*, os modelos XGB surgem como uma otimização, no sentido em que as novas árvores apenas são permitidas de serem criadas quando o ganho de uma nova inclusão é superior ao ganho obtido pela árvore anterior. Portanto, uma árvore pode ser podada se o ganho de uma inclusão for inferior ao ganho da árvore que a gerou (CHEN; GUESTRIN, 2016).

XGB também são conhecidos como métodos *ensemble*, que se referem a grupos ou coleções de modelos utilizados em uma mesma estrutura para atingir um objetivo em comum. Dado um conjunto de dados $m-$dimensional $X$ e um vetor $y$, tal que $x_i \in \mathbb{R}^m, y \in \mathbb{R}$, considera-se

como $\hat{y}$ um resultado dado por um conjunto de modelos (*ensemble*) representado pelo modelo generalizado (2.54).

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i) \ , \tag{2.54}$$

em que $f_k$ é uma árvore de regressão e, $f_k(x_i)$ representa o peso da k-ésima árvore para i-ésima observação (PAN, 2018). Assim como acontece em árvores aleatórias, neste método existe uma função, aqui chamada de função objetivo (2.55), que deve ser minimizada.

$$\mathscr{L}(\phi) = \sum_i L(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \ , \tag{2.55}$$

em que $L(y_i, \hat{y}_i)$ é a função perda. Na equação (2.55), há a inclusão de um termo penalizador $\Omega$, responsável por prevenir que o modelo se torne de alta complexidade:

$$\Omega(f_k) = \alpha T + \frac{1}{2}\lambda \|w\|^2 \ , \tag{2.56}$$

em que $\alpha$ e $\lambda$ são parâmetros que controlam as penalidades para o números de folhas $T$ e seus pesos $w$ respectivamente. Além de simplificar o modelo, a função $\Omega(f_k)$ também previne o chamado *overfitting* (PAN, 2018).

### 2.2.8 O problema de *overfitting*

Modelos são simplificações da realidade e, por isso, é esperado que seus resultados contenham um certo viés ou margem de erro. Contudo, qualquer modelo matemático, estatístico ou computacional é construído com base em teoremas e pressuposições que se atendidas garantem que os resultados destes modelos se tornem úteis e confiáveis.

Modelos de regressão linear (pertencentes à classe dos modelos GLM), por exemplo, possuem uma série de pressupostos estatísticos que devem ser satisfeitos a fim de atingir a adequabilidade do ajuste e confiabilidade dos resultados. Os mais conhecidos destes pressupostos são a independência e normalidade dos resíduos com média zero e variância constante. Além destes, Filho et al. (2011) afirmam que outro pressuposto deste tipo de modelagem (especificamente utilizando mínimos quadrados ordinários como método de estimação) é que qualquer variável exógena $X$, que é relevante para explicar a variável dependente $Y$ foi inclusa no modelo e, por

outro lado, qualquer variável exógena $X$ irrelevante para explicar $Y$ foi excluída do modelo. Este último pressuposto é também conhecido como o princípio da parcimônia (HAWKINS, 2004).

O problema de sobre-ajuste, do inglês *overfitting*, surge, portanto, quando o princípio da parcimônia é violado, ou seja, quando um modelo se torna mais complexo do que precisa ser. Dessa forma, modelos de ML, sejam eles supervisionados ou não, estão expostos ao risco de *overfitting*. Hawkins (2004) afirma que existem duas formas de *overfitting* importantes de serem distinguidas: aquele que surge de um modelo mais complexo do que precisa ser e aquele que inclui componentes desnecessárias. O autor também elenca algumas razões para que este problema seja evitado. Por exemplo, adicionar ao modelo preditores que não executam nenhuma função útil implica que em previsões futuras estes devem ser medidos e podem ser substituídos por valores constantes; além disso estas adições aumentam a variabilidade do modelo.

Portanto, considera-se um modelo sobre-ajustado se este for mais complexo do que outro modelo que realiza as mesmas tarefas, porém com menor poder de previsão. Isso significa que reconhecer o ajuste excessivo envolve não apenas a comparação do modelo mais simples e do modelo mais complexo, mas também a questão de como se mede o ajuste de um modelo (HAWKINS, 2004).

# 3 METODOLOGIA

## 3.1 Dados

As pesquisas relacionadas a investigação de crimes financeiros são particularmente comprometidas devido à dificuldade do uso de dados reais para o aprimoramento de evolução de soluções propostas por pesquisadores. Os dados necessários para este fim são, em geral, de domínio privado porque contém informações sigilosas que, se expostas à público sem a devida autorização de seu proprietário violam a Lei de Proteção de Dados Pessoais (LGPD, 2018).

A partir deste cenário, Elmir (2016) propõe uma solução para o problema da disponibilidade de dados financeiros, denominada Paysim. Paysim é um simulador financeiro que gera transações financeiras com base em conjuntos de dados reais com certas parametrizações. A similaridade dos dados simulados com dados originais é medida pelo cálculo da taxa de erro (ELMIR, 2016).

Portanto, através de diferentes parametrizações foram gerados dois diferentes cenários, cada um contendo um número distinto de conjunto de dados (CD). Cada conjunto de dados de cada cenário possui uma quantidade variável de observações (amostras) que depende da própria parametrização. Na Tabela 3.1 são mostrados os parâmetros escolhidos em cada cenário.

Tabela 3.1 – Parametrização dos cenários gerados pelo Paysim.

| Parâmetro | Cenário 1 | Cenário 2 |
|---|---|---|
| Horizonte temporal (em horas) | 1.000 | 1.000.000 |
| Número de clientes | 600 | 100.000 |
| Número de fraudadores | 5 | 150 |
| Número de bancos | 4 | 50 |
| Número de transações | 10000 | 5.000.000 |
| Probabilidade de Fraude | 0,001 | 0,00001 |
| Limite de transferência | 10.000 | 10.000 |
| Número de conjuntos de dados gerados | 1000 | 100 |

Fonte: Do autor (2021).

Os conjuntos de dados pertencentes ao Cenário 2 são computacionalmente mais pesados e devido a limitações de armazenamento em nuvem e capacidade processamento de dados, este cenário possui número menor de CD.

Devido a existência da componente aleatória e ao processo de geração de dados, o número de observações dos conjuntos de dados varia. Em média, os conjuntos de dados pertencentes

ao Cenário 1 possuem 64.883 observações e, com relação ao Cenário 2, o número médio é de 272.724 observações.

Todos os conjuntos de dados são compostos pelas variáveis descritas no Quadro 3.1.

Quadro 3.1 – Descrição das variáveis que compõem os conjuntos de dados simulados.

| Variável | Descrição | Tipo |
|---|---|---|
| Passo (v1) | Mapeia uma unidade de tempo no mundo real. Neste caso, 1 passo corresponde a 1 hora de tempo. | Discreta |
| Tipo (v2) | Tipo de transação: depósito, saque, débito, pagamento e transferência | Categórica |
| Quantia (v3) | Quantia total da transação realizada. | Contínua |
| Nome Origem (v4) | Consumidor que realizou a transação. | Categórica |
| Balanço Anterior Origem (v5) | Saldo inicial antes da transação | Contínua |
| Balanço Posterior Origem (v6) | Novo saldo após a transação. | Contínua |
| Nome Destino (v7) | Consumidor beneficiário da transação. | Categórica |
| Balanço Anterior Destino (v8) | Saldo inicial do beneficiário antes da transação. | Contínua |
| Balanço Posterior Destino (v9) | Saldo posterior do beneficiário após a transação. | Contínua |
| **Fraude (v10)** | Indica a incidência de fraude. | Discreta (binária) |

Fonte: Do autor (2021).

O comportamento fraudulento dos agentes visa lucrar na ação de assumir o controle das contas dos clientes e tentar esvaziar os fundos transferindo para outra conta e retirando do sistema. Dentro desta intenção pode então haver diferentes tipos de crimes financeiros, entre eles o crime de smurfing.

A descrição das variáveis no Quadro 3.1 sugere que algumas delas são correlacionadas, tais como o saldo bancário dos agentes antes de depois de uma transação. Em um processo de modelagem cujo objetivo é inferencial, este fato poderia representar um problema, como por exemplo o ferimento de algum pressuposto estatístico, a depender do modelo escolhido. Como o objetivo deste estudo é a detecção e predição de operações fraudulentas, nenhum tratamento às variáveis correlacionadas foi realizado.

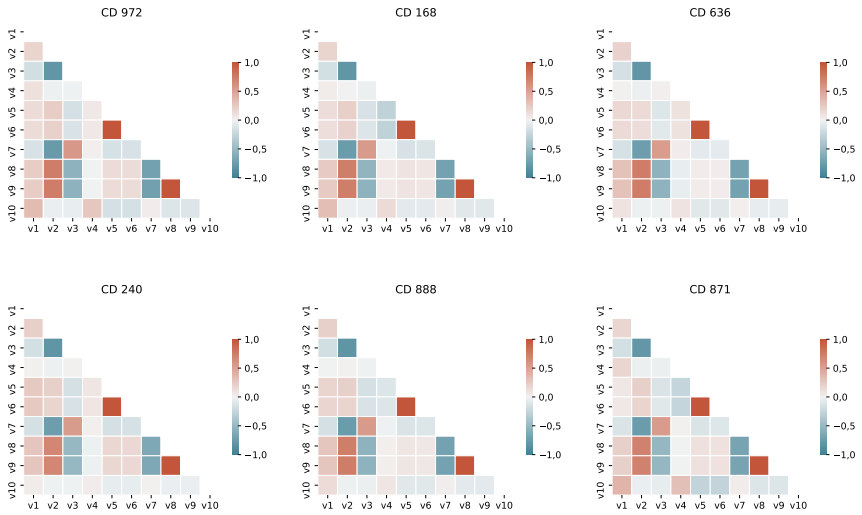
Durante a etapa de pré-processamento de dados algumas transformações foram realizadas. Primeiramente, todas as variáveis de natureza categórica foram codificadas, a fim de transformá-las em variáveis discretas. Além disso, todas as variáveis de natureza contínua passaram pelo processo de padronização, pois, o escalonamento transforma os valores das variáveis independentes de acordo com uma regra definida, de modo que elas tenham o mesmo grau de influência. Assim, o método é imune à escolha das unidades (HUANG; LI; XIE, 2015). A padronização das variáveis é dada pela equação (3.1).

$$Z = \frac{x - \mu_X}{\sigma_X} \tag{3.1}$$

em que $\mu_X$ e $\sigma_X$ são a média e o desvio padrão da variável em questão.

A relação entre cada variável dos conjuntos de dados pode ser analisada através de suas correlações. Como os conjuntos de dados de um mesmo cenário possuem comportamento semelhante foram selecionados aleatoriamente alguns deles para que fossem mostradas tais relações. Na Figura 3.1 é mostrado o grau de correlação entre as variáveis de alguns destes conjuntos.

Figura 3.1 – Correlações entre as variáveis dos conjuntos de dados pertencentes ao Cenário 1 escolhidos aleatoriamente.



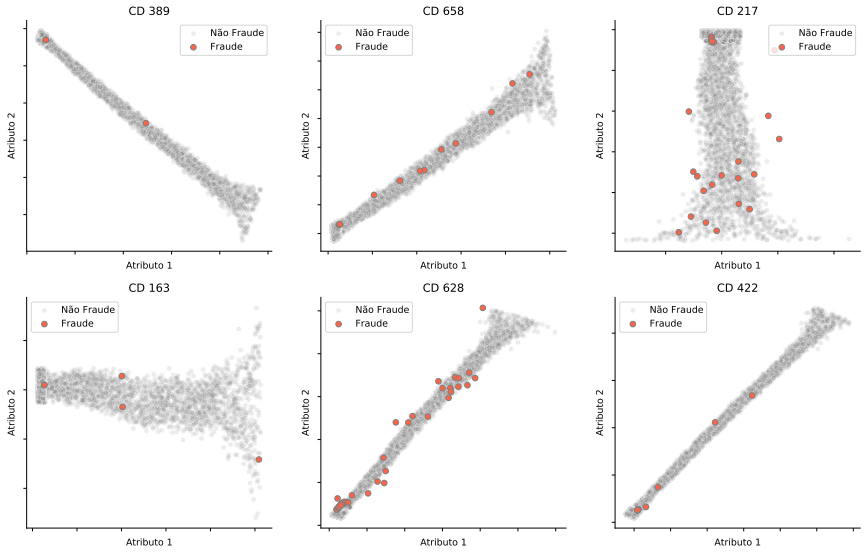

Fonte: Do autor (2021).

Justamente pelo fato dos conjuntos de dados serem gerados com as mesmas parametrizações era de se esperar que as correlações entre suas variáveis tivessem comportamento semelhante entre os conjuntos. De acordo com a Figura 3.1 muitos pares de variáveis apresentaram correlações expressivas, mas porque a própria natureza destas variáveis é de causa

e efeito, como por exemplo os pares Balanço Anterior Origem e Balanço Posterior Origem (0,99); Balanço Anterior Destino e Balanço Posterior Destino (0,99). Outros pares de variáveis apresentam fortes correlações que podem ajudar no processo investigativo, como por exemplo os pares Tipo e Quantia (-0,79); Tipo e Nome Destino (-0,74). Além disso, é possível observar na Figura 3.1 que a variável de interesse deste estudo é não correlacionada com qualquer outra.

Métodos de redução de dimensionalidade são úteis neste processo de análise exploratória de dados em problemas tratados com ML. Análise de Componente Principal (PCA) e t-SNE (*t-distributed Stochastic Neighbor Embedding*) são métodos que, além de outras funcionalidades, objetivam a visualização de dados em altas dimensões. Tais métodos buscam reduzir um conjunto de dados de alta dimensão em um conjunto de dados de 2 ou 3 dimensões, preservando o máximo possível das informações de cada dimensão (MAATEN; HINTON, 2008). Na Figura 3.2 são mostrados os mesmos conjuntos da Figura 3.1 com dimensões reduzidas a partir do método t-SNE.

Figura 3.2 – Conjuntos de dados com dimensionalidade reduzida.




Fonte: Do autor (2021).

Os novos eixos obtidos após a transformação não possuem um interpretação prática. Ainda assim, é possível observar como cada conjunto de dados é distribuído e aparentemente pode existir relação linear entre as variáveis. Em algumas projeções mostradas na Figura 3.2, as instâncias positivas não se separam de maneira evidente da massa de dados classificados como não fraude, o que pode demandar do processo de treinamento de cada modelo uma grande quantidade de dados.

Todo o processo de modelagem, análise descritiva, processamento e avaliação de modelos foram trabalhados utilizando as linguagens de programação Python (ROSSUM; JR, 1995) e R (R Core Team, 2021), através dos pacotes Scikit-Learn (PEDREGOSA et al., 2011) e FRBS (RIZA et al., 2015), respectivamente.

Os dados e códigos construídos para que este estudo pudesse ser realizado estão disponíveis no sítio eletrônico <https://github.com/osaraivamatheus/ml_models_comparison>.

## 3.2 Modelagem de dados desbalanceados

Um desafio comum que pesquisadores enfrentam ao utilizar modelos de classificação em eventos raros é o fato de se trabalhar com dados desbalanceados. A classificação de fraudes financeiras é um destes casos em que o número de eventos classificados negativamente (não fraude) é proporcionalmente superior ao número de eventos classificados positivamente (fraudes).

Em qualquer conjunto de dados reais é natural que haja um desequilíbrio entre as proporções de classes binárias, mas apenas são considerados como desbalanceados aqueles dados que apresentam desequilíbrio na ordem de 100 para 1, e no casos de fraudes, na ordem 100.000 para 1 (CHAWLA et al., 2002).

De acordo com Chawla et al. (2002) há na literatura duas principais maneiras de lidar com o problema. A primeira delas é o uso de uma técnica de sobreamostragem, que é aplicada nos dados da classe minoritária, aumentando sua proporção. A segunda delas é uma técnica de subamostragem, aplicada nos dados da classe majoritária, diminuindo sua proporção. O objetivo de ambas é equilibrar os dados para uma melhor modelagem, independente do modelo a ser utilizado. Ambas técnicas possuem algumas limitações e, neste sentido, Chawla et al. (2002) propõem uma combinação da sub-amostragem da classe majoritária com uma forma especial de sobreamostragem da classe minoritária, conhecida como SMOTE (*Synthetic Minority Over-sampling Technique*). No algoritmo proposto os autores sobre-amostram a classe minoritária criando exemplos "sintéticos" ao invés de sobre-amostragem com substituição. Ao invés da inserir perturbação nos dados amostrados, como rotação e inclinação, a criação de novos exemplos é gerada através alteração do espaço paramétrico dos dados. A classe minoritária é sobreamostrada tomando cada amostra de classe minoritária e introduzindo exemplos sintéticos ao longo dos segmentos de linha, que unem qualquer classe minoritária $k$-vizinhos mais próximos.

O algoritmo proposto pelos autores resultou em melhores modelos preditivos, considerando a curva ROC como métrica de avaliação dos modelos.

O desbalanceamento nos dados a serem modelados deve ser uma preocupação, uma vez que sua presença pode interferir diretamente na interpretação dos resultados. Por exemplo, é comum o uso da métrica acurácia para a avaliar os resultados preditivos de modelos de ML, mas o seu uso não é apropriado se os dados utilizados forem desbalanceados. Isso porque um alto percentual da acurácia na modelagem destes dados não significa necessariamente um bom modelo (CHAWLA et al., 2002).

## 3.3 Algoritmo para detecção de *smurfing*

Dada a definição de fraude financeira trazida por Reurink (2018), a qual estabelece que tal prática pode ser materializada através de golpes financeiros, este estudo pretende avaliar a eficiência de modelos estatísticos e de ML no processo de investigação de lavagem de dinheiro, especificamente a prática de *smurfing*, cuja conceituação foi explorada na subseção 2.1.1.

Apesar de possuir uma definição simples, o crime de *smurfing* possui características que tornam sua detecção um problema de alta complexidade. Isto porque as transações oriundas de práticas de *smurfing* podem ser semelhantes à transações regulares. Neste sentido, a análise exploratória dos dados disponibilizados por um Órgão Público de Fiscalização (OPF) permitiu a interpretação e a proposição de um algoritmo que combina métricas para que os casos mais prováveis de *smurfing* sejam destacados, em meio a um conjunto extenso de dados.

Todas as informações do banco de dados são organizadas em um horizonte temporal, o que torna necessário agregá-las por uma frequência de interesse (diária, semanal, mensal, etc.), indivíduo e tipo de transação (crédito ou débito). Esta operação de agregação torna possível, ainda, analisar as transações como séries temporais, o que abre possibilidade para a existência de outros tipos de crimes financeiros, como contas de passagem[1].

Em linhas gerais, a lavagem de dinheiro via práticas de *smurfing* consiste no fracionamento de um montante de ativos através de transações com valores que não são representativos, ou que chamariam a atenção das autoridades públicas responsáveis por fiscalizar este tipo de crime.

[1] Existência de padrões de saque em espécie ou transferência para outras contas em periodicidade que indica que os recursos ficam por tempo muito curto em conta, em comparação com padrões esperados em transações legítimas.

Portanto, a quantidade de transações feitas por um único indivíduo, para um ou mais destinos, é o primeiro fator considerado como importante para a detecção de *smurfing* no conjunto de dados.

Além disso, os investigadores do OPF afirmam que mesmo quando o fracionamento do valor lavado seja feito por intermediários, como é o caso do *smurfing* tipo II (mostrado na Figura 2.1), as transações tendem a ter magnitude semelhante, ou com baixa variabilidade. Isto porque existe um preço que os lavadores de dinheiro cobram para que eles utilizem suas contas como intermediárias destas operações. A medida escolhida para representar a variabilidade dos valores transacionados é o coeficiente de variação, definido na equação (3.2).

$$CV_i^{(t)} = 100 \cdot \frac{s}{\bar{x}} \tag{3.2}$$

em que $CV_i^{(t)}$ é o coeficiente de variação das transações de tipo $t$ (crédito ou débito) do indivíduo $i$; $\bar{x}$ é a média destas transações e $s$ o desvio padrão.

Há duas principais razões para que o coeficiente de variação tenha sido escolhido como medida para representar a variabilidade dos valores transacionados. Em primeiro lugar, esta é uma medida que relativiza o desvio padrão pela média, ou seja, torna possível a comparação entre indivíduos com diferentes poderes aquisitivos. Em segundo lugar, o coeficiente de variação se mostrou como um preditor linear estatisticamente válido para a quantidade de transações em um modelo de regressão, definido na equação (3.3).

$$Y_i = \beta_0 + \beta_1 CV_i + \varepsilon_i \ , \tag{3.3}$$

em que $Y_i$ é a quantidade de transações feitas por um indivíduo, $\beta_0$ é o intercepto, $\beta_1$ é o coeficiente angular da reta estimada e $\varepsilon_i \sim N(0, \sigma^2)$. Os resultados desta regressão, considerada como um estudo auxiliar e investigativo, são mostrados na Tabela 3.2.

Tabela 3.2 – Coeficiente de variação como preditor linear da quantidade de transações financeiras.

| Coef. | Est. | Erro padrão | $t$ | $P > \|t\|$ | $[0,025$ | $0,975]$ |
|---|---|---|---|---|---|---|
| Const. | -116,17 | 2,02 | -57,65 | 0,0 | -120,12 | -112,22 |
| *CV* | 94,65 | 0,83 | 114,54 | 0,0 | 93,03 | 96,27 |

Fonte: Do autor (2021).

A normalidade dos resíduos, homoscedasticidade e a autocorrelação da regressão realizada foram verificadas através dos testes Jarque-Bera (JARQUE, 2011) e Durbin-Watson (KRÄMER,

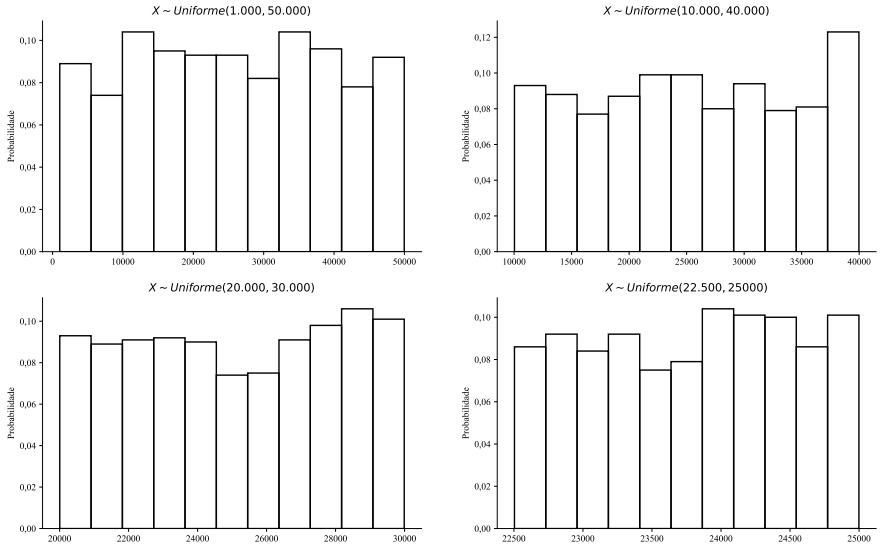
2011), respectivamente, ao nível de 5% de significância. Além disso, 54% da variabilidade total da quantidade das transações é explicada pela reta ajustada (coeficiente de determinação).

A eficiência de um processo de detecção de casos de *smurfing* pode depender de um contexto, ou de no mínimo, a observação de padrões no comportamento das transações financeiras de pessoas investigadas. Padrões como sazonalidade, quantidade de transações e repetição de valores transacionados, são fatores que podem caracterizar a prática de *smurfing*. Desta forma, estabelecer critérios como intervalo de valores a serem considerados durante o processamento do algoritmo, horizonte temporal e conjunto de indivíduos destinos é fundamental para que o algoritmo proposto aumente sua probabilidade de sucesso.

Com relação ao critério que diz respeito à definição de um intervalo de valores transacionados, é importante notar que este está diretamente relacionado com o coeficiente de variação das informações agregadas. Logo, ao se estabelecer um intervalo de valores de baixa magnitude, ou seja estreito, automaticamente o coeficiente de variação das informações agregadas é limitado e, assim, a interpretação de alta ou baixa variabilidade deve considerar tal intervalo. No Exemplo 4 é mostrada tal relação utilizando valores gerados aleatoriamente de uma distribuição Uniforme.

**Exemplo 4** *Seja um conjunto de variáveis aleatórias* $X_i \sim Uniforme(a,b)$. *Cada uma delas possui diferentes parametrizações que estreitam o intervalo fechado* $[a,b]$. *Valores de tais distribuições foram gerados aleatoriamente e seus comportamentos são mostrados na Figura 3.3.*

Figura 3.3 – Histograma de variáveis aleatórias com distribuição uniforme.




Fonte: Do autor (2021).

A relação entre o coeficiente de variação de cada vetor de valores gerados aleatoriamente com distribuição uniforme e o seu respectivo intervalo $[a,b]$ é mostrado na Tabela 3.3.

Tabela 3.3 – Relação entre o tamanho de intervalo e o coeficiente de variação de uma distribuição uniforme.

| $X(a,b)$ | $CV(X)$ |
|---|---|
| $X(1.000,50.000)$ | 0,5434 |
| $X(10.000,40.000)$ | 0,2873 |
| $X(20.000,30.000)$ | 0,1189 |
| $X(22.500,25.000)$ | 0,0301 |

Fonte: Do autor (2021).

Em dados reais é possível que os valores transacionais possam ter comportamentos de diferentes distribuições de probabilidade, mas em todos os casos, a relação explorada entre o coeficiente de variação e o intervalo de valores considerados é válida, pois ao estreitar tal intervalo a tendência é de que a variância do vetor escolhido diminua.

Portanto, a partir da quantidade de transações (débito ou crédito) feitas por um indivíduo durante um determinado período de tempo, de preferência mensal, o algoritmo deve ordenar, em ordem decrescente, tomando como referência prioritária tal quantidade e secundariamente o coeficiente de variação de tais transações. No Algoritmo 3 é mostrado o processo detecção de possíveis casos de *smurfing*.

**Algoritmo 3** Detecção de possíveis casos de *smurfing*.

1: Filtrar informações com intervalo de valores a serem considerados como importantes. Ex.: valores entre R$1.000,00 e R$5.000,00.
2: Filtrar horizonte temporal de interesse.
3: Agregar informações por indivíduo, frequência temporal e tipo de operação (débito ou crédito) utilizando o coeficiente de variação e a contagem de transações para tal agregação.
4: Organizar as informações em ordem decrescente priorizando a sequência: quantidade de transações, coeficiente de variação e valor total transacionado no período.

O resultado do Algoritmo 3 é uma tabela que ranqueia os indivíduos cujas transações realizadas têm as principais características da definição de *smurfing*. Entretanto, apenas com este resultado não é possível a identificação de que tipo de *smurfing* possa ter ocorrido em determinado conjunto de transações, fato que sugere uso de outras ferramentas, tais como a análise de grafos, como complemento investigativo (STARNINI et al., 2021).

## 3.4 Métricas de comparação de modelos

Existem diversas métricas utilizadas para comparar e avaliar o desempenho de modelos estatísticos e computacionais. A matriz de confusão é uma maneira simples de verificar a proporção de todas as possíveis classificações: falsos positivos ($FP$), falsos negativos ($FN$), verdadeiros positivos ($VP$) e verdadeiros negativos ($VN$). A definição matemática destas classificações depende de duas funções base, mostradas nas equações (3.4) e (3.5).

$$f(p|t) = \begin{cases} 1, & \text{se } p = t \\ 0, & \text{se } p \neq t \end{cases} \text{, tal que } p,t \in [0,1] \ . \tag{3.4}$$

$$g(p|t) = \begin{cases} 1, & \text{se } p \neq t \\ 0, & \text{se } p = t \end{cases} \text{, tal que } p,t \in [0,1] \ . \tag{3.5}$$

Em que para ambas funções $f$ e $g$ os parâmetros $p$ e $t$ representam o vetor de valores preditos por um modelo e o vetor de valores reais observados nos dados de teste, respectivamente. Logo, as classificações $VP, VN, FP, VN$ são dadas pelas equações

$$VP = \sum_{i=1}^{n} f(p = i|t = 1) \ , \tag{3.6}$$

$$VN = \sum_{i=1}^{n} f(p = i|t = 0) \ , \tag{3.7}$$

$$FP = \sum_{i=1}^{n} g(p = i|t = 0) \ , \tag{3.8}$$

$$FN = \sum_{i=1}^{n} g(p = i|t = 1) \ , \tag{3.9}$$

em que $n$ é o tamanho do vetor de observações separado como dados de teste. A partir destas classificações, na matriz de confusão estes valores são relativizados tornando possível a criação

de métricas como acurácia, precisão, especificidade, revocação, acurácia balanceada, entre outras. Um exemplo de matriz de confusão é mostrado no Quadro 3.2.

Quadro 3.2 – Exemplo de matriz de confusão.

| | | Real | |
|---|---|---|---|
| | | 0 | 1 |
| Predito | 0 | VN | FN |
| | 1 | FP | VP |

Fonte: Do autor (2021).

A utilidade das métricas construídas a partir da matriz de confusão pode depender do objetivo de uma modelagem, mas também da estrutura de dados. Neste estudo, por exemplo, os dados são desbalanceados e isto faz com que o uso da acurácia e da especificidade não seja adequado (CHAWLA et al., 2002). O uso da acurácia no contexto deste trabalho faria com que os modelos obtivessem um resultado satisfatório sem que, na verdade, fossem de fato bons classificadores. Isto porque a acurácia mede o percentual de classificações corretas pelo quociente $\frac{VP+VN}{VP+VN+FP+FN}$. Neste caso, um modelo que sempre prediz a classe negativa obterá um alto percentual de acurácia, sem que na verdade tenha identificado qualquer caso de fraude.

Portanto, é necessária uma métrica que penalize aqueles modelos que obtém altas taxas de falsos positivos $\left(TFP = \frac{FP}{VP+FN}\right)$ e taxas de falsos negativos $\left(TFN = \frac{FN}{VP+FN}\right)$, sem desconsiderar a sua precisão, definida na equação (3.10).

$$\text{Precisão} = \frac{VP}{VP+FP} \quad . \tag{3.10}$$

As métricas que atendem a estas demandas de avaliação de desempenho são conhecidas como *F-score*. Estas são métricas que podem ser entendidas como uma média harmônica entre a precisão e revocação (Recall), que é mostrada na equação (3.11)

$$\text{Revocação} = \frac{VP}{VP+FN} \tag{3.11}$$

A revocação é também conhecida como sensibilidade.

A métrica $F_1$, mostrada na equação (3.12), mede um equilíbrio entre a precisão e a revocação de um classificador.

$$F_1 = 2 \cdot \frac{\text{Precisão} \cdot \text{Recall}}{\text{Precisão} + \text{Recall}} \ , \tag{3.12}$$

ou seja, a métrica $F_1$ avalia o equilíbrio entre precisão e revocação sem qualquer distinção de importância entre elas. Como o que se busca é uma métrica que penalize classificadores com altas taxas de falso negativo, a métrica $F_\beta$ se mostra adequada para a proposta deste estudo, uma vez é possível ponderar o equilíbrio em questão atribuindo mais valor a revocação. O cálculo da métrica $F_\beta$ é mostrado na equação (3.13).

$$F_\beta = (1+\beta^2) \cdot \frac{\text{precisão} \cdot \text{Recall}}{(\beta^2 \cdot \text{precisão}) + \text{Recall}} \ . \tag{3.13}$$

O parâmetro $\beta$ é um número real positivo, responsável pela ponderação desejada. Quando $\beta = 1$, as métricas $F_1$ e $F_\beta$ são equivalentes. Quando $\beta > 1$ a revocação possui mais peso (ou importância) do que a precisão. Neste estudo, duas parametrizações foram utilizadas: $\beta = 2$ e $\beta = 0,5$.

Além destas métricas de avaliação, em muitos casos é importante também estudar a relação entre especificidade e a sensibilidade. Tal relação é analisada graficamente e uma das formas de utilizar tal relação como métrica de avaliação é através do cálculo da área abaixo da curva estabelecida entre a especificidade e sensibilidade. Esta, portanto, é uma métrica conhecida como Área sob a curva ROC – AUC.

## 3.5 Hiperparametrização de modelos

Em termos computacionais existem parâmetros para algoritmos que podem alcançar melhores resultados em uma modelagem, sejam eles referentes a performance de predição ou de tempo de execução. Por outro lado, há também os chamados parâmetros teóricos que cuja combinação das diferentes possibilidades capacitam os modelos para o alcance de melhores estimadores no processo inferencial.

Uma prática comum dentro do campo da modelagem computacional é a hiperparametrização de modelos. Esta prática objetiva avaliar o desempenho de um modelo com todas as possíveis combinações de parâmetros, sejam eles computacionais e/ou teóricos. Este é um processo que possui um alto custo computacional, pois as combinações de cada elemento dos conjuntos de parâmetros são testados exaustivamente. Este é um processo em que se tem um

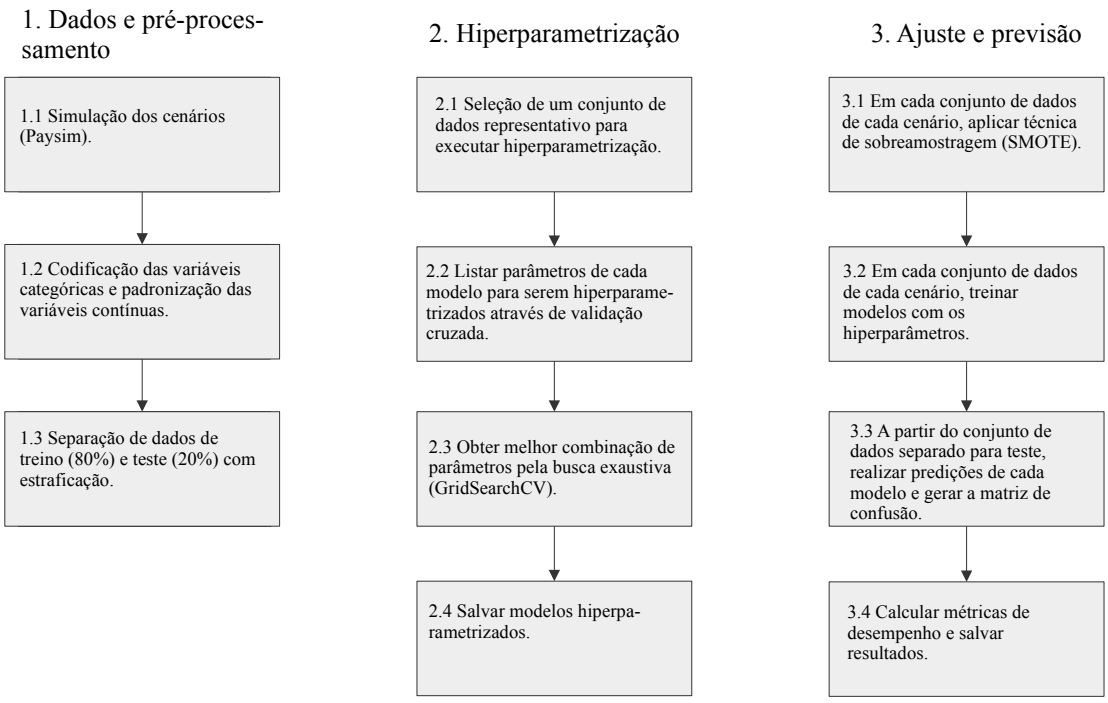
único parâmetro $k$ que se refere ao número de grupos em que uma determinada amostra de dados deve ser dividida.

Desta forma, quando o objetivo é a utilização da validação cruzada para hiperparametrizar modelos, cada subconjunto $k_i$ é utilizado como dados de teste, tornando os demais subconjuntos $k_j$, em que $j \neq i$, dados de treino. Uma grande desvantagem da validação cruzada é que o número de execuções de treinamento que devem ser realizadas é aumentado pelo parâmetro $k$, e isso pode ser problemático para modelos em que o próprio treinamento é caro computacionalmente (BISHOP, 2006, p. 33).

No processo de modelagem utilizado neste trabalho, todos os modelos estudados passaram por hiperparametrização, processada através de funcionalidades do pacote de Scikit-Learn (PEDREGOSA et al., 2011). A escolha dos elementos que compõem o conjunto de parâmetros de cada modelo a ser hiperparametrizado levou em consideração o perigo de *overfitting*.

Em resumo, todo o processo de modelagem deste trabalho é mostrado na Figura 3.4.

Figura 3.4 – Esquema do processo de modelagem utilizado.




Fonte: Do autor (2021).

Na etapa pré-processamento de dados da Figura 3.4 seria possível, ainda, a inclusão de mais um processo de subseleção de dados. Este processo se refere à aplicação do algoritmo proposto na subseção 3, que dentre as instâncias positivas, subselecionaria apenas aquelas em que se referem

à crimes de smurfing (tipo I ou II). Esta tarefa não foi possível de ser realizada, devido ao fato de que o algoritmo ainda precisa de verificação humana para checagem completa e, como este estudo trabalha com 1.100 conjunto de dados distintos, tal tarefa ainda se mostra inviável.

Já na etapa de hiperparametrização de modelos, a tarefa 2.1 utilizou um conjunto de dados gerado separadamente pelo Paysim, obedecendo as parametrizações de cada Cenário. Logo, estas amostras de cada cenário passaram pelo tratamento de dados da etapa 1 para que pudessem ser utilizadas para realizar a hiperparametrização. Com relação à tarefa 2.2, a lista de parâmetros de cada modelo e os parâmetros escolhidos são mostrados no Apêndice A. Ainda com relação a etapa de hiperparametrização, na tarefa 3 foi utilizado o método *GridSearchCV*, que faz uma busca exaustiva sobre valores de parâmetro especificados para um estimador (PEDREGOSA et al., 2011), escolhendo aqueles que obtiverem melhor resultado de acordo com uma métrica especificada. Neste caso a métrica escolhida foi $F_2$.

# 4 RESULTADOS E DISCUSSÃO

Dentro do conjunto das possíveis classificações que cada modelo pode produzir, a quantidade de falsos negativos ($FN$) é um elemento que, no contexto das fraudes financeiras, deve possuir relação direta com a qualidade das predições feitas. Uma vez que fraudes financeiras são eventos raros quando distribuídas entre transações legais, a quantidade de $FN$ que um modelo produz deve penalizá-lo pois tal fato demonstra certa fragilidade na capacidade de detecção do problema.

Desta forma, os primeiros resultados apresentados se referem justamente às classificações $FN$ produzidas por cada modelo em cada cenário, agregados de acordo com a média

$$\overline{TFN} = \frac{1}{N}\sum_{i=1}^{N} FN_i \ ,$$

em que $N$ é o número de conjuntos de dados (CD) de cada cenário e $FN_i$ é a quantidade de classificações $FN$ da matriz de confusão feita por um modelo no conjunto de dados de teste em questão. Tais resultados são mostrados na Tabela 4.1.

Tabela 4.1 – Média e desvio padrão da taxa de Falsos Negativos por modelo e cenário.

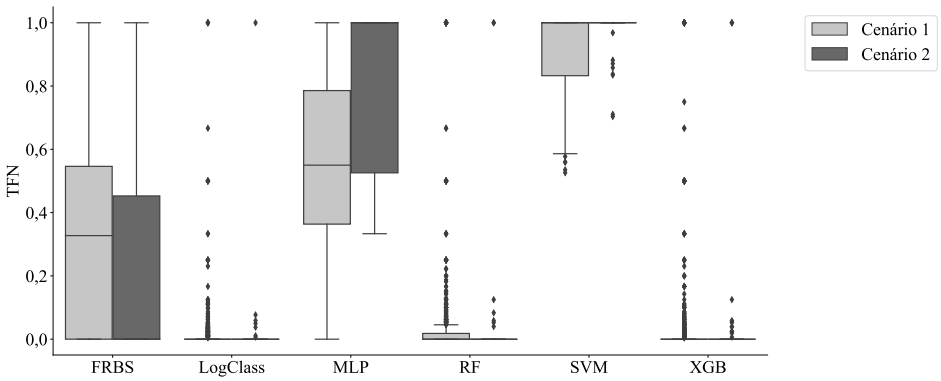
| Cenário | Modelo | $\overline{TFN}$ | Desvio padrão |
|---|---|---|---|
| Cenário 1 | LogClass | 0,0157 | 0,0836 |
| | XGB | 0,0355 | 0,1409 |
| | RF | 0,0407 | 0,1435 |
| | FRBS | 0,3335 | 0,3333 |
| | MLP | 0,5530 | 0,3068 |
| | SVM | 0,9195 | 0,1240 |
| Cenário 2 | LogClass | 0,0138 | 0,1058 |
| | RF | 0,0262 | 0,1487 |
| | XGB | 0,0385 | 0,1804 |
| | FRBS | 0,2213 | 0,2968 |
| | MLP | 0,7807 | 0,2408 |
| | SVM | 0,9852 | 0,0538 |

Fonte: Do autor (2021).

Em ambos cenários, o modelo logístico apresentou a menor $\overline{TFN}$, o que mostra que o modelo mais simples foi mais eficiente nesta avaliação. Por outro lado, o modelo SVM apresentou o pior desempenho em termos de taxa de falsos negativos. A variabilidade da $TFN$, expressa na Tabela 4.1 por meio do desvio padrão, sugere que predições feitas por cada modelo possuem qualidades relativamente próximas entre si, devido à magnitude de tal dispersão. Já na

Figura 4.1, outras estatísticas descritivas sobre a taxa de falsos negativos podem ser observadas através dos gráficos *boxplot*.

Figura 4.1 – *Boxplots* da taxa de Falsos Negativos por modelo e cenário.




Fonte: Do autor (2021).

Apesar de alguns *outliers*, o modelo logístico, RF e XGB apresentaram TFN altamente concentrada em torno de zero. Já os modelos FRBS e MLP, apresentaram as maiores variabilidades e, além disso, em termos de $\overline{TFN}$, são modelos que não atingiram resultado razoável em comparação aos modelos logístico, RF e XGB. Por fim, o modelo SVM apresentou variabilidade relativamente baixa, mas os resultados estão concentrados em torno de uma alta $\overline{TFN}$, e portanto, o pior desempenho em relação a esta métrica.

A $\overline{TFN}$ é um primeiro indício de quais modelos alcançaram os melhores desempenhos. Entretanto, não é suficiente para realizar a comparação desejada. Para este fim, a avaliação e combinação das métricas *AUC*, *RECALL*, $F_{0,5}$, $F_1$, $F_2$ é utilizada.

Para analisar a homogeneidade dos resultados de cada métrica atingida pelos modelos, na Tabela 4.2 é mostrado o coeficiente de variação de cada uma delas, em cada cenário.

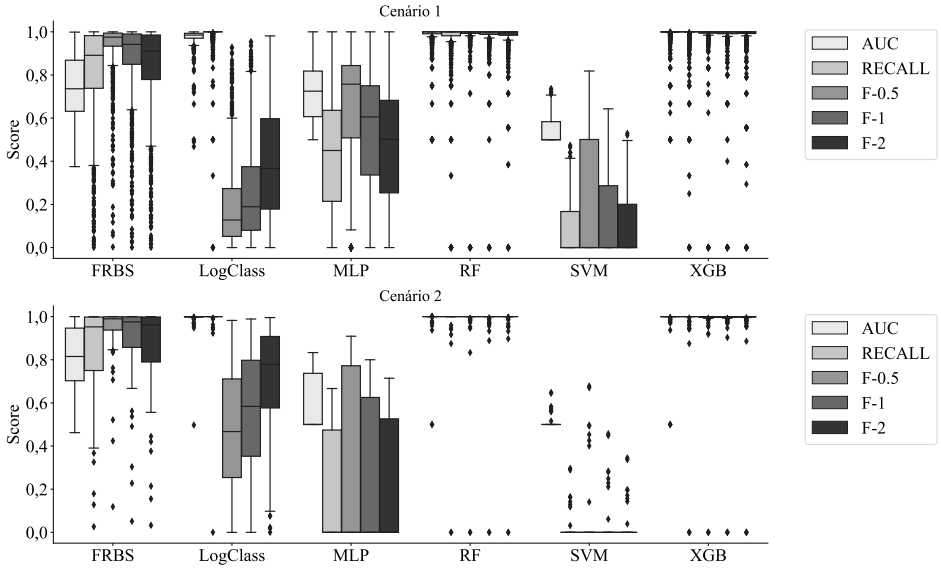
Tabela 4.2 – Coeficiente de variação (%) de cada métrica por modelo e cenário.

| Cenário | Modelo | *AUC* | *Recall* | $F_{0,5}$ | $F_1$ | $F_2$ |
|---|---|---|---|---|---|---|
| Cenário 1 | FRBS | 20,25 | 27,99 | 14,85 | **21,40** | 25,69 |
| | LogClass | 4,58 | 8,49 | 99,98 | 87,11 | 65,46 |
| | MLP | 21,17 | 68,60 | 51,44 | 57,75 | 64,11 |
| | RF | 7,32 | 14,95 | 13,19 | **13,81** | **14,50** |
| | SVM | 11,47 | 153,86 | 142,67 | 149,05 | 152,38 |
| | XGB | 7,17 | 14,60 | 12,49 | **13,29** | **14,09** |
| Cenário 2 | FRBS | 18,29 | 27,01 | 14,53 | **20,65** | **24,78** |
| | LogClass | 5,36 | 10,67 | 58,34 | 51,45 | 40,48 |
| | MLP | 19,64 | 109,19 | 105,60 | 107,14 | 108,44 |
| | RF | 7,49 | 15,19 | 15,20 | **15,16** | **15,16** |
| | SVM | 5,27 | 360,65 | 339,52 | 351,73 | 357,96 |
| | XGB | 9,15 | 18,66 | 18,62 | **18,62** | **18,64** |

Fonte: Do autor (2021).

Como as métricas $F_1$ e $F_2$ são construídas através de parâmetros que penalizam modelos com piores desempenhos no contexto deste estudo, na Tabela 4.2 estão destacados em negrito os valores destas métricas que são inferiores à 25%. Este é um limiar escolhido neste trabalho para considerar dados de resultados com maior homogeneidade. Dessa forma, os modelos FRBS, RF e XGB possuem maiores consistências destes resultados em torno de suas respectivas médias. Toda esta variabilidade nos resultados e demais estatísticas descritivas podem ser observados através dos *boxplots* apresentados na Figura 4.2.

Figura 4.2 – *Boxplots* de cada métrica por modelo e cenário.




Fonte: Do autor (2021).

A importância de se observar as variabilidades e consistências dos resultados vem do fato de que as observações que compõem a Figura 4.2 foram obtidas através do resultado de modelagens feitas em 1.100 conjuntos de dados distintos, que apesar de terem sido simulados com parametrizações semelhantes, todos eles apresentam distinções entre si. Desta forma, os modelos que atingiram resultados das métricas escolhidas que variam (pouco) em torno de 1, são modelos com maiores capacidades de detecção de fraudes, como é o caso do XGB e RF. Entretanto, ainda que estes modelos tenham apresentado as menores variabilidades em torno de 1, eles também apresentam (em todas as métricas) algumas falhas, representadas pelos seus respectivos *outliers* da Figura 4.2.

O ranqueamento final para a escolha do melhor modelo combina todas as métricas de avaliação propostas de acordo com sua relevância no contexto de identificação fraudes financeiras. Para tal classificação é necessário um vetor de pesos $\vec{W}$ que quantifica a relevância das métricas envolvidas de forma simultânea, de tal modo que os elementos de $\vec{W}$ seguem a condição

$$\sum_{i=1}^{5} w_i = 1 \ ,$$

em que $0 \leq w_i \leq 1$.

A escolha dos elementos que compõem $\vec{W}$ foi feita de tal forma que as métricas que penalizam erros tais como os Falsos Negativos, como por exemplo a métrica $F_\beta$, possuem maior relevância. Desta forma, $\vec{W}$ é dado por:

$$\vec{W} = [0.1, 0.15, 0.2, 0.25, 0.3]$$

de modo que

- $w_1$: peso referente à métrica $AUC$;
- $w_2$: peso referente à métrica $Recall$;
- $w_3$: peso referente à métrica $F_{0,5}$;
- $w_4$: peso referente à métrica $F_1$;
- $w_5$: peso referente à métrica $F_2$.

Desta forma, o ranqueamento final é dado por uma nota final auferida através do produto interno entre $\vec{W}$ e o vetor de médias das métricas obtidas por cada modelo, conforme a equação

$$N_{m,c} = \vec{W} \cdot \overline{\mu}_{m,c} \ \ ,$$

em que $\overline{\mu}_{m,c} = \dfrac{1}{N} \sum_{i=1}^{N} x_i$ é a média de uma determinada métrica de um modelo $m$ em um cenário $c$. Na Tabela 4.3 são mostrados o resultados de $\overline{\mu}_{m,c}$, além da nota final atribuída a cada modelo, já ranqueada em ordem decrescente.

Tabela 4.3 – Ranking final dos modelos. Os melhores resultados de cada métrica e cenário são mostrados em negrito.

| Cenário ($c$) | Modelo ($m$) | $\overline{\mu}_{AUC,c}$ | $\overline{\mu}_{Recall,c}$ | $\overline{\mu}_{F_{0,5},c}$ | $\overline{\mu}_{F_1,c}$ | $\overline{\mu}_{F_2,c}$ | $N_{m,c}$ |
|---|---|---|---|---|---|---|---|
| Cenário 1 | XGB | **0,9822** | 0,9645 | **0,9760** | **0,9705** | **0,9665** | **0,9707** |
| | RF | 0,9796 | 0,9593 | 0,9751 | 0,9682 | 0,9626 | 0,9677 |
| | FRBS | 0,7391 | 0,8117 | 0,9276 | 0,8723 | 0,8331 | 0,8492 |
| | MLP | 0,7233 | 0,4470 | 0,6239 | 0,5284 | 0,4725 | 0,5380 |
| | LogClass | 0,9749 | **0,9843** | 0,1973 | 0,2584 | 0,4024 | 0,4699 |
| | SVM | 0,5403 | 0,0805 | 0,1997 | 0,1274 | 0,0943 | 0,1662 |
| Cenário 2 | RF | 0,9869 | 0,9738 | **0,9735** | **0,9734** | **0,9736** | **0,9749** |
| | XGB | 0,9807 | 0,9615 | 0,9620 | 0,9618 | 0,9616 | 0,9636 |
| | FRBS | 0,8094 | 0,8402 | 0,9375 | 0,8908 | 0,8579 | 0,8745 |
| | LogClass | **0,9909** | **0,9862** | 0,4704 | 0,5508 | 0,6907 | 0,6860 |
| | MLP | 0,6096 | 0,2193 | 0,3620 | 0,2893 | 0,2425 | 0,3113 |
| | SVM | 0,5074 | 0,0148 | 0,0418 | 0,0247 | 0,0176 | 0,0728 |

Fonte: Do autor (2021).

Nota-se que no Cenário 1, o modelo com a maior nota atribuída é o modelo XGB e, no segundo cenário, o modelo RF. Para os demais modelos, em ambos cenários, a ordem de desempenho foi a mesma.

A nota final $N_{m,c}$ dos três melhores modelos, além de suas baixas taxas de Falsos Negativos provoca uma reflexão de que o uso destes modelos em contextos financeiros, em especial, na identificação de crimes de *smurfing* pode fazer com que estes sejam ferramentas úteis no rastreamento de agentes criminosos por parte de Órgãos Públicos de Fiscalização (OPF). As consequências desta utilização pode gerar velocidade neste processo, além da recuperação do montante transacionado ilegalmente.

# 5 CONCLUSÃO

O processo de investigação de crimes financeiros enfrenta uma série de desafios e, entre eles, se encontra a própria definição do que se configura especificamente tais crimes. Ainda, técnicas avançadas que agentes fraudadores utilizam para mascarar a entrada e saída de ativos no sistema financeiro e a disponibilidade de dados, são dificuldades características deste tipo de investigação por parte de Órgãos Públicos de Fiscalização (OPF).

Dentro deste contexto, o presente estudo buscou apresentar definições gerais do que são crimes financeiros, além de apresentar também um tipo especifico de crime desta natureza, conhecido como *smurfing*. Dada a definição deste tipo de crime, o algoritmo de detecção apresentado na subseção 3.3, apesar de depender da verificação humana para validação de seus resultados, pode ser utilizado para quando não se possui dados rotulados para treinamento de modelos ML supervisionados.

Ainda, com base na exploração da construção teórica dos modelos utilizados, é possível realizar adaptações na parametrização de cada modelo caso fossem utilizados em cenários reais, que por sua vez, poderiam fazer com que os desempenhos fossem distintos do que foram apresentados no capítulo 4.

O modelo Logístico, *Random Forests* (RF), *Support Vector Machine* (SVM), *Extreme Gradient Boosting* (XGB), Redes Neurais Artificiais (MLP) são modelos comumente utilizados em estudos desta natureza e, desta forma, este trabalho também apresenta uma outra alternativa, que são os Sistemas Baseados em Regras Fuzzy.

De acordo com forma de avaliação escolhida para a comparação dos modelos em tarefas de classificação de fraudes financeiras, os modelos com os melhores desempenhos foram o XGB e RF. Uma vez que a geração de dados utilizados para tal avaliação buscou ser fidedigna às características de dados desta natureza, ou seja, dados altamente desbalanceados, acredita-se que os modelos melhores ranqueados neste estudo atingiriam resultados semelhantes se aplicados em dados reais.

Sendo assim, acredita-se que esta pesquisa atingiu os seus objetivos e também confirmou suas premissas de que alguns modelos atingiriam resultados semelhantes, de tal forma que os resultados dos dois melhores modelos se diferenciam minimamente.

As limitações encontradas durante a realização desta pesquisa se referem principalmente à disponibilidade de dados reais, em especial, aqueles referentes especificamente a crimes de *smurfing*.

# REFERÊNCIAS


AWOYEMI, J. O.; ADETUNMBI, A. O.; OLUWADARE, S. A. Credit card fraud detection using machine learning techniques: A comparative analysis. In: IEEE. **2017 International Conference on Computing Networking and Informatics (ICCNI)**. [S.l.], 2017. p. 1–9. ISBN 978-1-5090-4643-0.

BARROS, L. C. d.; BASSANEZI, R. C.; LODWICK, W. A. **A first course in fuzzy logic, fuzzy dynamical systems, and biomathematics: theory and applications**. [S.l.]: Springer, 2017. ISBN 978-3-662-53324-6.

BEDE, B.; GAL, S. G. Almost periodic fuzzy-number-valued functions. **Fuzzy Sets and Systems**, v. 147, n. 3, p. 385–403, 2004. ISSN 0165-0114. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0165011403003518>.

BHATTACHARYYA, S. et al. Data mining for credit card fraud: A comparative study. **Decision Support Systems**, Elsevier, v. 50, n. 3, p. 602–613, 2011. ISSN 0167-9236. On quantitative methods for detection of financial fraud. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167923610001326>.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.]: springer, 2006. ISBN 978-0387-31073-2.

CAMPUS, K. Credit card fraud detection using machine learning models and collating machine learning models. **International Journal of Pure and Applied Mathematics**, v. 118, n. 20, p. 825–838, 2018.

CASELLA, G.; BERGER, R. L. **Statistical inference**. Pacific Grove, CA: Cengage Learning, 2002. ISBN 0-357-75312-7.

CETINA, K. K.; PREDA, A. **The Oxford Handbook of the Sociology of Finance**. 1. ed. [S.l.]: Oxford University Press, 2013. (Oxford Handbooks). ISBN 0199590168,9780199590162.

CHADHA, A.; KAUR, P. Handling smurfing through big data. In: ____. [S.l.: s.n.], 2018. p. 459–470. ISBN 978-981-10-6619-1.

CHAWLA, N. V. et al. Smote: synthetic minority over-sampling technique. **Journal of artificial intelligence research**, v. 16, p. 321–357, 2002.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: **Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining**. [S.l.: s.n.], 2016. p. 785–794.

CHO, T.-H.; CONNERS, R.; ARAMAN, P. Fast backpropagation learning using steep activation functions and automatic weight reinitialization. In: **Conference Proceedings 1991 IEEE International Conference on Systems, Man, and Cybernetics**. [S.l.: s.n.], 1991. p. 1587–1592 vol.3.

CUTLER, A.; CUTLER, D. R.; STEVENS, J. R. Random forests. Springer, p. 157–175, 2012.

DIDIMO, W. et al. An advanced network visualization system for financial crime detection. In: **2011 IEEE Pacific Visualization Symposium**. [S.l.: s.n.], 2011. p. 203–210.

ELMIR, A. **PaySim Financial Simulator: PaySim Financial Simulator**. Dissertação (Mestrado) — Blekinge Institute of Technology, 2016. Disponível em: <http://urn.kb.se/resolve?urn=urn:nbn:se:bth-14061>.

FILHO, D. F. et al. O que fazer e o que não fazer com a regressão: pressupostos e aplicações do modelo linear de mínimos quadrados ordinários (mqo). **Revista Política Hoje**, v. 20, n. 1, 2011.

FRIEDMAN, J. H. Stochastic gradient boosting. **Computational statistics & data analysis**, Elsevier, v. 38, n. 4, p. 367–378, 2002.

GALL, J.; RAZAVI, N.; GOOL, L. V. An introduction to random forests for multi-class object detection. In: **Outdoor and large-scale real-world scene analysis**. [S.l.]: Springer, 2012. p. 243–263.

GARNER, B. A. et al. Black's law dictionary. Thomson/West St. Paul, MN, 2004.

HAMMER, B.; GERSMANN, K. A note on the universal approximation capability of support vector machines. **neural processing letters**, Springer, v. 17, n. 1, p. 43–53, 2003.

HAWKINS, D. M. The problem of overfitting. **Journal of chemical information and computer sciences**, ACS Publications, v. 44, n. 1, p. 1–12, 2004.

HUANG, J.; LI, Y.-F.; XIE, M. An empirical analysis of data preprocessing for machine learning-based software cost estimation. **Information and Software Technology**, v. 67, p. 108–127, 2015. ISSN 0950-5849. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0950584915001275>.

JARQUE, C. M. Jarque-bera test. In: ____. **International Encyclopedia of Statistical Science**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 701–702. ISBN 978-3-642-04898-2. Disponível em: <https://doi.org/10.1007/978-3-642-04898-2_319>.

KARPOFF, J. M. The future of financial fraud. **Journal of Corporate Finance**, v. 66, p. 101694, 2021. ISSN 0929-1199. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0929119920301383>.

KLIR, G.; YUAN, B. **Fuzzy sets and fuzzy logic**. [S.l.]: Prentice hall New Jersey, 1995. v. 4. ISBN 0131011715.

KRÄMER, W. Durbin–watson test. In: ____. **International Encyclopedia of Statistical Science**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 408–409. ISBN 978-3-642-04898-2. Disponível em: <https://doi.org/10.1007/978-3-642-04898-2_219>.

LGPD. Lei geral de proteção de dados pessoais. **Diário Oficial da República Federativa do Brasil**, Brasília, DF, 2018. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>.

MAATEN, L. Van der; HINTON, G. Visualizing data using t-sne. **Journal of machine learning research**, v. 9, n. 11, 2008.

NELLES, O. **Nonlinear System Identification: From Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes**. Cham, Switzerland: Springer Nature, 2020. ISBN 978-3-030-47438-6.

NGAI, E. W. et al. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. **Decision support systems**, Elsevier, v. 50, n. 3, p. 559–569, 2011.

NOBLE, W. S. What is a support vector machine? **Nature biotechnology**, Nature Publishing Group, v. 24, n. 12, p. 1565–1567, 2006.

PAASCH, C. A. **Credit card fraud detection using artificial neural networks tuned by genetic algorithms**. Tese (Doutorado) — The Hong Kong University of Science and

Technology, http://hdl.handle.net/1783.1/6252, 2 2008.

PAN, B. Application of xgboost algorithm in hourly pm2. 5 concentration prediction. In: IOP PUBLISHING. **IOP conference series: earth and environmental science**. [S.l.], 2018. v. 113, n. 1, p. 012127.

PATLE, A.; CHOUHAN, D. S. Svm kernel functions for classification. In: IEEE. **2013 International Conference on Advances in Technology and Engineering (ICATE)**. [S.l.], 2013. p. 1–9.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PEDRYCZ, W. **An Introduction to Computing with Fuzzy Sets**: Analysis, design, and applications. 1. ed. Edmonton, AB, Canada: Springer, Cham, 2021. (Intelligent Systems Reference Library). ISSN 1868-4394. ISBN 978-3-030-52799-0.

PICKETT, K. S.; PICKETT, J. M. **Financial crime investigation and control**. [S.l.]: John Wiley & Sons, 2002.

R Core Team. **R: A Language and Environment for Statistical Computing**. Vienna, Austria, 2021. Disponível em: <https://www.R-project.org/>.

REURINK, A. Financial fraud: a literature review. **Journal of Economic Surveys**, Wiley Online Library, v. 32, n. 5, p. 1292–1325, 2018.

RIZA, L. S. et al. frbs: Fuzzy rule-based systems for classification and regression in R. **Journal of Statistical Software**, v. 65, n. 6, p. 1–30, 2015. Disponível em: <http://www.jstatsoft.org/v65/i06/>.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

ROSSUM, G. V.; JR, F. L. D. **Python reference manual**. [S.l.]: Centrum voor Wiskunde en Informatica Amsterdam, 1995.

SADDIQ, S. A.; BAKAR, A. S. A. Impact of economic and financial crimes on economic growth in emerging and developing countries. **Journal of Financial Crime**, Emerald Publishing Limited, 2019.

SHALIZI, C. **Advanced data analysis from an elementary point of view**. [S.l.]: Citeseer, 2013.

STARNINI, M. et al. Smurf-based anti-money laundering in time-evolving transaction networks. 2021.

SUYKENS, J. A.; SIGNORETTO, M.; ARGYRIOU, A. **Regularization, Optimization, Kernels, and Support Vector Machines**. Boca Raton, FL: Chapman & Hall/CRC, 2015. ISBN 978-1-4822-4140-2.

TERZIC, J. et al. Ultrasonic sensor based fluid level sensing using support vector machines. In: ____. **Ultrasonic Fluid Quantity Measurement in Dynamic Vehicular Applications: A Support Vector Machine Approach**. Heidelberg: Springer International Publishing, 2013. p. 37–52. ISBN 978-3-319-00633-8. Disponível em: <https://doi.org/10.1007/978-3-319-00633-8_3>.

TORRES, T. V. **First Contact with Deep Learning: Practical Introduction with Keras**.

[S.l.]: Kindle Direct Publishing, 2018. ISBN 978-1-983-21155-3.

WADE, C. **Hands-On Gradient Boosting with XGBoost and scikit-learn: Perform accessible Python machine learning and extreme gradient boosting with Python**. Birmingham: PACKT Publishing LTD, 2020. ISBN 9781839218354.

WEST, J.; BHATTACHARYA, M. Intelligent financial fraud detection: a comprehensive review. **Computers & security**, Elsevier, v. 57, p. 47–66, 2016.

ZAJMI, L.; AHMED, F. Y.; JAHARADAK, A. A. Concepts, methods, and performances of particle swarm optimization, backpropagation, and neural networks. **Applied Computational Intelligence and Soft Computing**, Hindawi, v. 2018, p. 7, 2018.

ZHANG, H.; SINGER, B. H. **Recursive partitioning and applications**. 2. ed. [S.l.]: Springer-Verlag New York, 2010. ISBN 978-1-4419-6823-4.

# APÊNDICE A – Modelos hiperparametrizados

Neste Apêndice são mostradas as parametrizações utilizadas no processo de hiperparametrização dos modelos estudados neste trabalho. Em cada Quadro apresentado neste capítulo, a coluna referente ao nome dos parâmetros de cada modelo preservou o termo utilizado pelas respectivas bibliotecas.

Em cada quadro são mostrados apenas os principais parâmetros em que este estudo se preocupou em especificar. Entretanto, muitos outros parâmetros que dizem respeito à forma como os modelos foram trabalhados computacionalmente, por exemplo, o número de núcleos dos processadores a serem utilizados para otimizar o tempo de modelagem, foram omitidos e utilizados valores padrão de cada biblioteca.

**Sistemas Baseados em Regras Fuzzy – FRBS**

Quadro .1 – Hiperparametrização do modelo fuzzy.

| Parâmetros | Descrição | Listagem | Melhor |
|---|---|---|---|
| num.labels | Número de rótulos (termos linguísticos) | 2,3,4,5,6,7,8 | 4 |
| type.mf | Função de pertinência | Triangle, trapezoid, gaussian, sigmoid, bell | sigmoid |
| type.tnorm | T-norma utilizada | Min, Hamacher, Yager, product, bounded | product |
| type.implication.func | Função de inplicação | DIENES_RESHER, LUKASIEWICZ, ZADEH, GOGUEN, GODEL, SHARP, MIZUMOTO, DUBOIS_PRADE, e MIN | ZADEH |
| type.defuz | Método de defuzificação | WAM, FIRST.MAX, LAST.MAX, MEAN.MAX, COG | COG |
| method.type | O algoritmo de aprendizagem a ser usado para criar a base de regras | FRBCS.W, FRBCS.CHI, GFS.GCCL, FH.GBML | FRBS.CHI |

Fonte: Do autor (2021).

Durante o processo de modelagem, o parâmetro mais contundente no sentido de melhoria da avaliação do modelo foi o número de termos linguísticos para modelar o termo “fraude”. Percebe-se uma relação direta entre este parâmetro e o seu desempenho. Entretanto, o melhor valor foi 4 devido à verificação de *overfitting*. Os demais parâmetros obtiveram pouca influência na variabilidade do desempenho. As demais especificações dos parâmetros omitidos são descritos no sítio eletrônico <https://www.rdocumentation.org/packages/frbs/versions/3.2-0/topics/frbs.learn>.

**Modelo Logístico**

Quadro .2 – Hiperparametrização do modelo logístico.

| Parâmetros | Descrição | Listagem | Melhor |
|---|---|---|---|
| C | Inverso da força de regularização | 0.5, 1, 3, 10 | 3 |
| class_weight | Pesos associados às classes | balanced | balanced |
| dual | Formulação dupla ou primária | False, True | False |
| fit_intercept | Especifica se uma constante (também conhecida como polarização ou interceptação) deve ser adicionada à função de decisão. | False, True | True |
| penalty | Especifica a regra da penalidade | l1, l2, elasticnet, 'none' | l2 |
| solver | Algoritmo para usar no problema de otimização do sistema de equações de derivadas parciais. | Newton-cg, lbfgs, liblinear, sag, saga. | Newton-cg |

Fonte: Do autor (2021).

Durante o processo de modelagem, os parâmetros com maior influência da variabilidade dos resultados do modelo logístico foram "solver" e "penalty".

**Redes Neurais Artificiais – MLP**

Quadro .3 – Hiperparametrização do modelo MLP.

| Parâmetros | Descrição | Listagem | Melhor |
|---|---|---|---|
| hidden_layer_sizes | O iésimo elemento representa o número de neurônios na iésima camada oculta | 10, 20, 30, 50, 100, 150, 200, 500 | 200 |
| activation | Função de ativação | identity, logistic, tanh, relu | tanh |
| solver | Método para otimização de peso | lbfgs, sgd, adam | adam |
| learning_rate | Cronograma de taxas de aprendizagem para atualizações de peso. | constant, invscaling, adaptive | constant |

Fonte: Do autor (2021).

Durante o processo de hiperparametrização deste modelo observou-se que o número de camadas escondidas do modelo e as funções de ativação são os principais parâmetros que tinham maior influência na variabilidade do desempenho. Entretanto, o risco de *overfitting* e o custo computacional possuem relação direta com o aumento do número de camadas escondidas e o uso de funções de ativação mais complexas, respectivamente. Os parâmetros omitidos no Quadro .3 são descritos no sítio eletrônico <https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html>.

**Random Forest**

Quadro .4 – Hiperparametrização do modelo Random Forest.

| Parâmetros | Descrição | Listagem | Melhor |
|---|---|---|---|
| n_estimators | Número de árvores na floresta | 10, 50, 100, 150, 200, 400 | 200 |
| criterion | A função para medir a qualidade do desdobramento | gini, entropy | gini |
| max_depth | A profundidade máxima da árvore. | 2, 3, 4, None | None |
| min_samples_split | O número mínimo de amostras necessárias para dividir um nó interno | 2, 3, 4, 5, 6 | 2 |
| min_samples_leaf | O número mínimo de amostras necessárias para estar em um nó folha. | 1, 2, 3, 4, 5, 6 | 1 |
| class_weight | Pesos associados a cada classe | balanced, None, balanced_subsample | None |

Fonte: Do autor (2021).

A quantidade de árvores escolhidas na floresta é um dos parâmetros em que se deve ter cautela durante a listagem, pois está diretamente relacionado ao aprendizado do modelo e, consequentemente, com o perigo de *overfitting*. Além disso, o parâmetro "max_depth" determina o quanto cada nó de cada árvore será expandido. Como no caso a escolha foi "None", os nós foram expandidos até que todas as folhas fossem puras ou até que todas as folhas contivessem menos do que o número mínimo de amostras necessárias para dividir um nó interno, que é o parâmetro "min_samples_leaf".

**Support Vector Machine – SVM**

Quadro .5 – Hiperparametrização do modelo SVM.

| Parâmetros | Descrição | Listagem | Melhor |
|---|---|---|---|
| C | Parâmetro de regularização | 0.5, 1, 3, 10, 15 | 3 |
| kernel | Função kernel | linear, poly, rbf, sigmoid, precomputed | rbf |
| degree | Grau da função polinomial utilizado para função kernel polinomial | 1, 2, 3, 4 | 3 |
| gamma | Coeficiente kernel para funções rbf, polinomial e sigmoidal | scale, auto | auto |
| coef | Termo independente da função kernel. Utilizado apenas na função polinomial e sigmoidal. | 0, 0.5, 1, 2 | 0 |
| class_weight | Multiplicadores do parâmetro C para cada classe. | auto | auto |

Fonte: Do autor (2021).

Assim como no estudo de Bhattacharyya et al. (2011), a função *kernel* gaussiana foi utilizada para realizar a projeção do produto escalar necessário para adicionar dimensões.

**Extreme Gradient Boosting – XGB**

Quadro .6 – Hiperparametrização do modelo XGB.

| Parâmetros | Descrição | Listagem | Melhor |
|---|---|---|---|
| n_estimators | Número de rodadas de impulsionamento | 10, 50, 100, 150, 200, 400 | 100 |
| booster | impulsionador a ser usado | gbtree, gblinear ou dart | gbtree |
| importance_type | O tipo de importância do recurso o método "feature_importances" | gain, weight, cover, total_gain, total_cover | gain |
| max_depth | Profundidade máxima da árvore para cada base learner | 2, 3, 10, 20, None | None |

Fonte: Do autor (2021).

O modelo XGB é um método *ensemble* que, neste estudo, utilizou árvores aleatórias para o processo de modelagem. Assim como aconteceu no modelo RF, a escolha do parâmetro “max_depth” foi “None”, ou seja, os nós de cada árvore foram expandidos até que todas as folhas fossem puras ou até que todas as folhas contivessem menos do que o número mínimo de amostras necessárias para dividir um nó interno.