UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Bauru

Carlos Alexandre Carvalho Tojeiro

Identificação de *Portscan* com *Machine Learning* - Uma Análise dos Impactos de Algoritmos de *Oversampling* e *Undersampling* no Desenvolvimento de Sistemas de Detecção de Intrusão

Bauru
2024

Carlos Alexandre Carvalho Tojeiro

Identificação de *Portscan* com *Machine Learning* - Uma Análise dos Impactos de Algoritmos de *Oversampling* e *Undersampling* no Desenvolvimento de Sistemas de Detecção de Intrusão

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de Bauru.

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Coorientador: Prof. Dr. Thiago José Lucas

Bauru
2024

T646i

Tojeiro, Carlos Alexandre Carvalho

Identificação de Portscan com Machine Learning : Uma Análise dos Impactos de Algoritmos de Oversampling e Undersampling no Desenvolvimento de Sistemas de Detecção de Intrusão / Carlos Alexandre Carvalho Tojeiro. -- Bauru, 2024

100 p. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru

Orientador: Kelton Augusto Pontara da Costa

Coorientador: Thiago José Lucas

1. Aprendizado do computador. 2. Redes de computadores. 3. Sistemas de detecção de intrusão (Medidas de segurança). I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Bauru

Faculdade de Ciências

ATA DA DEFESA PÚBLICA DA DISSERTAÇÃO DE MESTRADO DE CARLOS ALEXANDRE CARVALHO TOJEIRO, DISCENTE DO PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO, DA FACULDADE DE CIÊNCIAS - CÂMPUS DE BAURU.

Aos 18 dias do mês de julho do ano de 2024, às 10:00 horas, por meio de Videoconferência, realizou-se a defesa de DISSERTAÇÃO DE MESTRADO de CARLOS ALEXANDRE CARVALHO TOJEIRO, intitulada **Identificação de Portscan com Machine Learning - Uma Análise dos Impactos de Algoritmos de Oversampling e Undersampling no Desenvolvimento de Sistemas de Detecção de Intrusão.** A Comissão Examinadora foi constituida pelos seguintes membros: Prof. Dr. KELTON AUGUSTO PONTARA DA COSTA (Orientador(a) - Participação Virtual) do(a) Departamento de Computação / UNESPCâmpus de Bauru, Prof. Dr. DOUGLAS RODRIGUES (Participação Virtual) do(a) Computação / Faculdade de Ciências - UNESP, Prof. Dr. JOÃO BAPTISTA CARDIA NETO (Participação Virtual) do(a) Faculdade de Tecnologia de Catanduva. / Centro Estadual de Educação Tecnológica Paula Souza. Após a exposição pelo mestrando e arguição pelos membros da Comissão Examinadora que participaram do ato, de forma presencial e/ou virtual, o discente recebeu o conceito final:_ **APROVADO** . Nada mais havendo, foi lavrada a presente ata, que após lida e aprovada, foi assinada pelo(a) Presidente(a) da Comissão Examinadora.

Prof. Dr. KELTON AUGUSTO PONTARA DA COSTA

gov.br Documento assinado digitalmente
KELTON AUGUSTO PONTARA DA COSTA
Data: 18/07/2024 09:16:32-0300
Verifique em https://validar.iti.gov.br

Digitalizado com CamScanner

Carlos Alexandre Carvalho Tojeiro

Identificação de *Portscan* com *Machine Learning* - Uma Análise dos Impactos de Algoritmos de *Oversampling* e *Undersampling* no Desenvolvimento de Sistemas de Detecção de Intrusão

Comissão Examinadora

Prof. Dr. Kelton Augusto Pontara da Costa
UNESP - Câmpus de Bauru
Orientador

Prof. Dr. Thiago José Lucas
FATEC - Câmpus de Ourinhos
Coorientador

Prof. Dr. Douglas Rodrigues
UNESP - Câmpus de Bauru

Documento assinado digitalmente
gov.br
DOUGLAS RODRIGUES
Data: 29/08/2024 16:08:56-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. João Baptista Cardia Neto
FATEC - Câmpus de Catanduva

Documento assinado digitalmente
gov.br
JOAO BAPTISTA CARDIA NETO
Data: 28/08/2024 11:18:59-0300
Verifique em https://validar.iti.gov.br

Bauru
18 de Julho de 2024

Resumo

A necessidade de trabalhar com as informações digitais nas empresas gerou um crescimento nos números de ataques cibernéticos em busca de ativos valiosos. Em contraproposta, pesquisadores e administradores de redes vêm buscando uma precisão maior do processo que visa descobrir portas "abertas" em dispositivos de rede, sendo que as "portas" são pontos de acesso pelos quais os dados transitam nos dispositivos. Dentro desta proposta, este trabalho apresenta um estudo da aplicação de combinações de algoritmos de Aprendizado de Máquina (AM) que possam identificar uma maior quantidade de tentativas de "escaneamento de portas", técnica usada para identificar o estado de uma porta de rede, já que os cibercrimes normalmente iniciam-se com um ataque de Portscan. O trabalho proposto compara dois métodos de seleção de características, aplicando métodos de undersampling e oversampling para o balanceamento das classes "normal" e "ataque", classificando-os com Rede Neural, Regressão Logística, Máquinas de Vetores de Suporte, Random Forest (RF), Decision Tree (DT) e k-Nearest Neighbors (kNN) no conjunto de dados CICIDS2017. O resultado adquirido com as diferentes combinações de algoritmos de AM demonstrou-se que a combinação dos classificadores RF, DT e kNN juntamente com a técnica de SMOTEENN, obtiveram desempenhos superiores as demais combinações propostas no trabalho.

Palavras-chave: *Machine Learning*; Balanceamento de Dados; Portscan; Sistemas de Detecção de Intrusão; Redes de Computadores.

Abstract

The need to work with digital information in companies has led to an increase in the number of cyber attacks in search of valuable assets. In counter-proposal researchers and network administrators have been searching for greater precision of the process of discovering "open ports" on network devices, where "ports" are access points for which information passes through the devices. Within this proposal, this work presents a study of the application of combinations of Machine Learning (ML) algorithms that can identify a greater number of "port scanning" attempts, a technique used to identify the state of a network port, since cybercrime cybercrimes usually begin with a port scanning attack. The proposed work compares two feature selection methods, applying undersampling and oversampling methods to balance the "normal" and "attack" classes, classifying them with using Neural Networks (NN), Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), Decision Tree (DT) and k-Nearest Neighbours (kNN) on the CICIDS2017 dataset. The results obtained with the different combinations of algorithms showed that the combination of the RF, DT and kNN classifiers together with the SMOTEENN technique performed better than the other combinations proposed in the work.

***Keywords**: Machine Learning; Data Balancing;Portscan; Intrusion Detection Systems; Computer Networks.*

Lista de ilustrações

Figura 1 – Metodologia do trabalho utilizada na revisão sistemática da literatura. . . . 18
Figura 2 – Distribuição dos artigos obtidos pelo ano de publicação. 19
Figura 3 – Fluxo do procedimento realizado na Revisão Sistemática da Literatura. . . 20
Figura 4 – Conjuntos de dados extraídos nos trabalhos selecionados - visualização ano a ano. 43
Figura 5 – Classificadores utilizados nos artigos correlatos. 45
Figura 6 – Técnicas de balanceamento utilizadas nos trabalhos correlatos. 47
Figura 7 – Métodos de balanceamento utilizadas nos trabalhos correlatos. 48
Figura 8 – Sistemas de Detecção de Intrusão. 59
Figura 9 – Curva da Regressão Logística Hosmer, Taber e Lemeshow (1991). 61
Figura 10 – Máquina de Vetor de Suporte 62
Figura 11 – Random Forest 64
Figura 12 – Decision Tree 65
Figura 13 – k-Nearest Neighbors 66
Figura 14 – Rede Neural Artificial 67
Figura 15 – Fluxo de trabalho do processo metodológico 72
Figura 16 – Validação Cruzada - 10-fold cross-validation 79

Lista de tabelas

Tabela 1 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2019. Fonte: Elaborado pelo autor. 23
Tabela 2 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2020. Fonte: Elaborado pelo autor. 27
Tabela 3 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2021. Fonte: Elaborado pelo autor. 33
Tabela 4 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2022. Fonte: Elaborado pelo autor. 37
Tabela 5 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2023. Fonte: Elaborado pelo autor. 42
Tabela 6 – Arquitetura de uma Matriz de Confusão. Fonte: Adaptada de Phetlasy et al. (2019) 50
Tabela 7 – Compilação dos resultados publicados para os artigos citados. Fonte: Elaborado pelo autor. 51
Tabela 8 – *Features* selecionadas por RFE e MI. Fonte: Elaborado pelo autor. 77
Tabela 9 – Combinações das Técnicas. Fonte: Elaborado pelo autor. 81
Tabela 10 – Modelos - Resultados das Combinações. Fonte: Elaborado pelo autor. . . . 84

Lista de abreviaturas e siglas

AM	- Aprendizado de Máquina
ANN	- *Artificial Neural Network*
AB	- *AdaBoost*
ACC	- *Accuracy*
A-SUWO	- *Adaptive Semi-Unsupervisioned Weighted Oversampling*
AUC	- *Area Under the ROC Curve*
BBBC	- *Big Bang-Big Crunch*
BFS	- *Best First Search*
BSMOTE	- *Borderline Synthetic Minority Oversampling TEchnique*
CC	- *ClusterCuntroids*
CERT.br	- Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil
CFS	- *Correlation Based Feature Selection*
CNN	- *Convolutional Neural Network*
CNNID	- *Convolutional Neural Networks for Intrusion Detection*
DDoS	- *Distributed Denial of Service*
DJ	- *Decision Jungle*
DT	- *Decision Tree*
E	- *Especificidade*
ENN	- *Edited Nearest Neighbor*
ERR	- Eliminação de Recurso Recursivo
ET	- *ExtraTrees*
FCBF	- *Fast Correlation Based Filter for Feature Selection*
FSA	- *Forward Selection Algorithm*

GB - *Gradient-Boosting*

GH - *Gametic Hereditica*

GI - *Gini Importance*

GNB - *Naive Bayes Gaussiano*

GRU - *Gated Recurrent Units*

IA - *Artificial Intelligence*

IBK - *Instance-based learning algorithms*

IDS - *Intrusion Detection System*

J48DT - *J48 Decision Tree*

kNN - *k Nearest Neighbor*

LBFGS - *Limited Broyden-Fletcher-Goldfarb-Shanno*

LL - *Log Loss*

LSTM - *Long Short Term Memory*

MC - *Matriz de Confusão*

MDI - *Feature Importance Measure*

MI - *Mutual Information*

ML - *Machine Learning*

MLP - *Multilayer Perceptron*

MRMR - *Maximum Relevance — Minimum Redundancy*

MTL - *Nonoidal T-Normal Basead Logic*

NB - *Naive Bayes*

NI - *Não Informado*

NN - *Neural Network*

OSS - *One-Sided Selection*

P - *Precision*

PCA - *Principal Component Analysis*

PI - *Permutation Importance*

PPV - *Positive Predictive Value*

RF - *Random Forest*

RFE - *Recursive Feature Elimination*

RN - *Neural Network*

RNN - *Recurrent Neural Networks*

ROC - *Receiver Operating Characteristic*

RUS - *Random under-sampling*

S - *Sensitivity*

SA - *Sem Aplicação*

SAE - *Sparse Auto-Encode*

SMOTE - *Synthetic Minority Over-sampling Technique*

STL - *Standard Template Library*

SVM - *Support Vector Machine*

t-SNE - *t-Distributed Stochastic Neighbor Embedding*

UCI-MLR - *UCI Machine Learning Repository*

UMAP - *Uniform Manifold Approximation and Projection*

VSTG-MTL - *Variable Selection and Task Grouping for Multi-Task Learning*

VT - *Vision Transformer*

XB - *eXtreme Gradient Boosting* XGBoost

Sumário

1 INTRODUÇÃO 14

2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 Revisão Sistemática da Literatura 17
2.1.1 Trabalhos Correlatos - ano de 2019 21
2.1.2 Trabalhos Correlatos - ano de 2020 24
2.1.3 Trabalhos Correlatos - ano de 2021 28
2.1.4 Trabalhos Correlatos - ano de 2022 34
2.1.5 Trabalhos Correlatos - ano de 2023 38
2.2 Inclinações Extraídas dos Trabalhos 43
2.2.1 Métricas de Desempenho Utilizadas nos Trabalhos Correlatos 48
2.2.2 Métricas de Desempenho Extraídas dos Trabalhos 50
2.2.3 Discussão dos Trabalhos em Aberto 56
2.3 Sistema de Detecção de Intrusão 58
2.4 Aprendizado de Máquina 59
2.5 Classificação 60
2.5.1 Regressão Logística (LR) 60
2.5.2 Máquinas de Vetores de Suporte (SVM) 62
2.5.3 Random Forest (RF) 63
2.5.4 Decision Tree (DT) 64
2.5.5 k-Nearest Neighbors (kNN) 65
2.5.6 Rede Neural Artificial (RNA) 66
2.6 Métodos de Balanceamento 67
2.6.1 SMOTE 68
2.6.2 SVMSMOTE 68
2.6.3 NearMiss 69
2.6.4 SMOTEENN 70

3 METODOLOGIA 72
3.1 Conjunto de Dados 73
3.2 Pré-Processamento 74
3.2.1 Limpeza e Preparação dos Dados 75
3.2.2 Seleção de Características 75
3.2.3 Hiperparâmetros e Validação Cruzada 78

4 RESULTADOS E DISCUSSÃO 81

5 CONSIDERAÇÕES FINAIS 92
5.1 Publicações 93

REFERÊNCIAS 94

1 Introdução

Com o crescente número de informações armazenadas em computadores, houve um expressivo aumento de ataques cibernéticos, provocando indisponibilidade de serviços de redes, e roubo de ativos, resultando em sérios prejuízos para empresas e organizações.

Para realização destes ataques, surgem as varreduras (scan), que abrange, buscas minuciosas em redes de computadores, força bruta de senhas, exploração de vulnerabilidades. De acordo com o Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br), de Janeiro a Dezembro de 2023 as notificações de ataques de Portscan somaram 75,27% dos incidentes reportados no Brasil.

De acordo com Abdi et al. (2024), estes ataques estão tornando-se cada vez mais sofisticados e difíceis de detectar. Por outro lado, para proteger as informações e manter os dispositivos de rede disponíveis aos usuários, vêm surgindo vários estudos direcionados ao processo de identificação de ataques, que utilizam conjuntos de dados pré-coletados de ferramentas de detecção de intrusão, treinados com técnicas de AM para a identificação de ações maliciosas.

Dentro deste contexto IDSs geram grande quantidade de alertas, sendo a maioria falsos positivos, que podem induzir o sistema à se tornar menos eficaz ou inútil devido a identificação de ameaças com muita frequência. Por outro lado, considera-se que a diminuição dos alertas de falsos negativos seja o fator mais importante na detecção de intrusões na rede, por serem ataques verdadeiros classificados como falsos, o que pode acarretar uma avaliação equivocada dos resultados.

Desta forma, caso o número de falsos negativos apresentar uma quantidade elevada, o acompanhamento pode ser impreciso. Logo, a Sensibilidade e *F1-Score* são métricas de avaliação essenciais para este trabalho por apresentar alto contraste entre as classes com muitos registros e outras com poucos registros (RAHMA et al., 2023).

Em solução a estes problemas são aplicadas diversas técnicas de diferentes combinações que implicam em um melhor desempenho do modelo de AM para reconhecimento de verdadeiras ameaças, que utilizam de tática para identificar e explorar vulnerabilidades na rede.

Entre essas melhorias surgem os métodos que trabalham com conjuntos de dados desbalanceados, que podem ser classificados em duas categorias principais. A primeira categoria trabalha ao nível de dados para equilibrar a quantidade das classes presentes no conjunto de dados, sendo os mais comuns são *oversampling* e *undersampling*. A segunda categoria trabalha modificando a classificação existente no algoritmo, aplicando penalidades quando a classificação tende a classe majoritária (NGUYEN; COOPER; KAMEI, 2009).

Os métodos de *oversampling* e *undersampling* são utilizados quando existe uma distorção nas distribuições das classes dos conjuntos de dados usados para treinamentos de modelos de AM, problema que denominamos de classificação desbalanceada.

Também destaca-se um terceiro método ao nível de dados onde é aplicado a combinação dos métodos de *oversampling* e *undersampling* nomeada de *resampling* (BAGUI; LI, 2021).

A classificação desbalanceada implica em resultados tendenciosos, ou seja, quando o algoritmo treinado aprende demais sobre uma classe dominante do conjunto de dados, enquanto negligencia todas as outras classes menores que contém baixas quantidade de dados, havendo um sobreajuste da classe majoritária.

A problemática elencada no presente trabalho baseou-se para identificar quais combinações de técnicas de balanceamento de dados podem ser trabalhadas buscando um melhor modelo de AM, *oversampling*, *undersampling* ou a terceira abordagem híbrida que combina ambas as abordagens anteriores nomeada *resampling*?

O Objetivo Geral da presente Dissertação foi analisar os impactos de custo computacional e desempenho nos erros de classificação pelo emprego combinado de algoritmos de balanceamento de dados e de seleção de características aplicados a Detecção de Intrusão.

Os Objetivos Específicos foram os que se seguem:

- Realização de uma Revisão Sistemática na Literatura de forma com que fosse possível compreender o estado-da-arte do tema estudado;
- De posse do estado-da-arte, a extração e análise das principais variáveis envolvidas nos trabalhos correlatos, quais sejam: classificadores, conjuntos de dados, algoritmos de balanceamento de dados e métodos de seleção de características;
- Quanto aos materiais e métodos: pré-processamento dos dados; classificação dos objetos e análise dos resultados;
- Compilação dos resultados obtidos e;
- Análise, considerações e sugestões de pesquisas futuras;

Em Fundamentação Teórica (Capítulo 2), o trabalho resumiu-se no emprego de Revisão Sistemática da Literatura (Seção 2.1) onde foram extraídos os dados relevantes dos trabalhos correlatos obtidos na pesquisa. Em seguida na (Seção 2.2) são verificadas as principais inclinações extraídas dos trabalhos. Logo após (Seção 2.3) são apresentados os conceitos relevantes para compreensão de Sistema de Detecção de Intrusão, em sequencia (Seção 2.4) apresenta-se a definição de Aprendizado de Máquina, os classificadores (Seção 2.5) utilizados nas predições dos modelos propostos deste trabalho e por fim (Seção 2.6), mostram-se os conceitos dos algoritmos de balanceamento das classes que foram aplicados no presente trabalho.

Já em Metodologia (Capítulo 3), pode-se verificar a descrição do processo de pesquisa do trabalho empregada no conjunto de dados (Seção 3.1), o pré-processamento (limpeza e preparação dos dados) e o processo utilizado para seleção das características dos dados no conjunto de dados podem ser visto na (Seção 3.2). Em Resultados e Discussão (Capítulo 4) apresentam-se as combinações dos algoritmos para gerar modelos de AM e os resultados parciais obtidos com os testes a partir do desempenho das combinações propostas. Por fim, em Considerações Finais (Capítulo 5) contém a conclusão dos resultados obtidos do desempenho do modelo proposto e as publicações durante a execução das disciplinas obrigatórias e do desenvolvimento da Dissertação.

De acordo com Souza, Dias e Santos (2019), os métodos de pesquisa utilizados, abordam os seguintes passos: Planejamento (formalização por meio de um protocolo), Condução (identificação, seleção, extração e análise dos dados) e Apresentação dos Resultados (sistematização dos dados).

2 Fundamentação Teórica

O presente capítulo tem por objetivo especificar os conceitos relevantes para compreensão de IDS e AM.

2.1 Revisão Sistemática da Literatura

A Revisão Sistemática da Literatura abrange o levantamento de trabalhos correlatos de forma que o processo possa ser replicável por pesquisadores e estudantes. É um componente de identificação, avaliação e interpretação de toda a pesquisa, ou área pesquisada. Kitchenham e Charters (2007) destacam a importância da definição dos critérios de busca para que haja uma análise ponderada em relação aos artigos que serão selecionados, também, possibilitando que pesquisadores consigam dar continuidade ao segmento estudado.

Já Martins et al. (2013), descrevem a revisão sistemática da literatura, como a parte do trabalho onde os estudos da pesquisa buscam incluir uma classificação empírica ou analítica, também os métodos (teste, investigação, revisão) e técnicas que serão aplicadas.

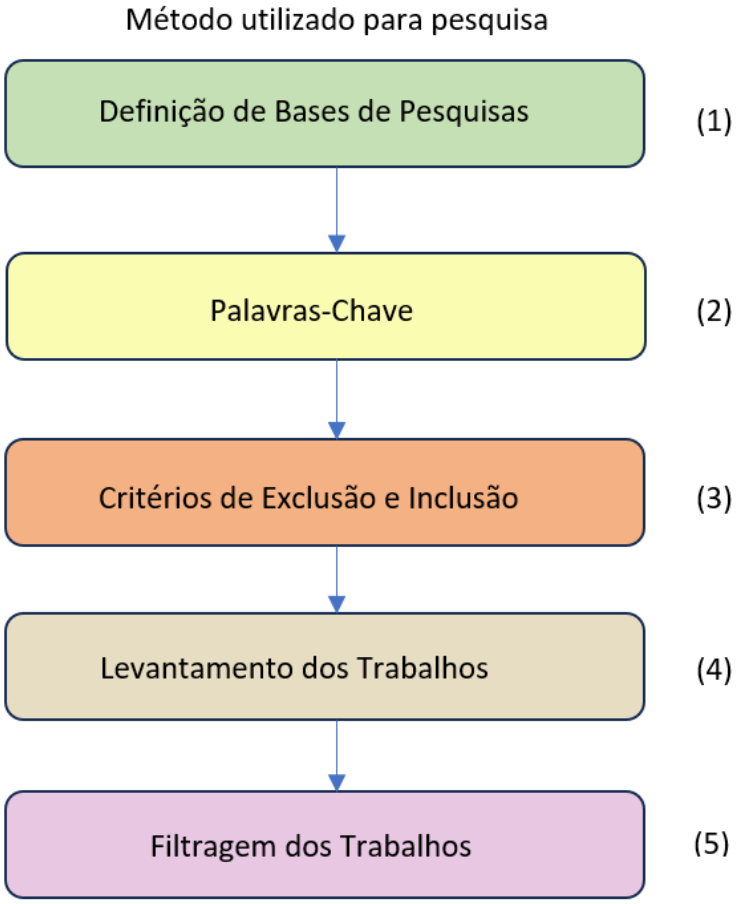
Na mesma linha de pensamento, Lopes e Fracolli (2008), diz que entre as mais importantes características da revisão sistemática estão: fontes de pesquisa abrangentes, seleção dos primeiros estudos da pesquisa sob critérios aplicados de forma homogênea e avaliação fundamentada da amostra. Também salienta, que o tratamento estatístico dado aos resultados obtidos a partir de revisões sistemáticas é visto como meta-análise e vem sendo amplamente utilizado por pesquisadores.

Biolchini et al. (2005) relata o procedimento como sendo uma forma de se obter evidência na literatura capaz de justificar a própria pesquisa, que deve ser realizada após a definição de uma formalidade de buscas e de uma metodologia geral.

Para Souza, Dias e Santos (2019), uma revisão sistemática deve considerar os seguintes passos: estabelecer a técnica a ser utilizada na pesquisa, definir as palavras-chave encontrando palavras e sinônimos que melhor possa definir o escopo do trabalho, definir a *strings* de busca onde se possa fazer concatenação com o conector *OR* ou *AND*, definir as bases de busca e fazer o refinamento da busca aplicando filtros.

A revisão sistemática obedeceu aos critérios de Souza, Dias e Santos (2019), conforme se observa nas Figuras 1 e 3 ilustram os procedimentos usados.

Figura 1 – Metodologia do trabalho utilizada na revisão sistemática da literatura.



Fonte: Elaborado pelo autor

Detalham-se as etapas ilustradas na Figura 1 conforme segue:

1. Na primeira etapa (1) definiu-se em quais bases seriam feitas as pesquisas dos artigos. As seguintes plataformas de trabalhos publicados foram utilizadas: IEEExplore[1], ACM *Digital Library*[2], SPRINGER[3] e MDPI [4], também realizou-se pesquisas nas bases Scopus [5] obedecendo os mesmos critérios, mas obteve como retorno artigos repetidos que já tinham sidos adicionados, então manteve-se apenas as quatro primeiras bases de pesquisas.

2. Logo em seguida, na etapa (2), buscou-se trabalhos com palavras contidas no título, resumo e palavras-chave como "*intrusion*" e "*imbalance*" ou "*oversampling*" ou "*under-sampling*", além de limitar o intervalo de publicação entre o ano de 2019 até 2023 para utilizar-se os trabalhos mais recentes neste estudo.

[1] https://ieeexplore.ieee.org/
[2] https://dl.acm.org/
[3] https://rd.springer.com/
[4] https://www.mdpi.com/
[5] https://www.scopus.com/

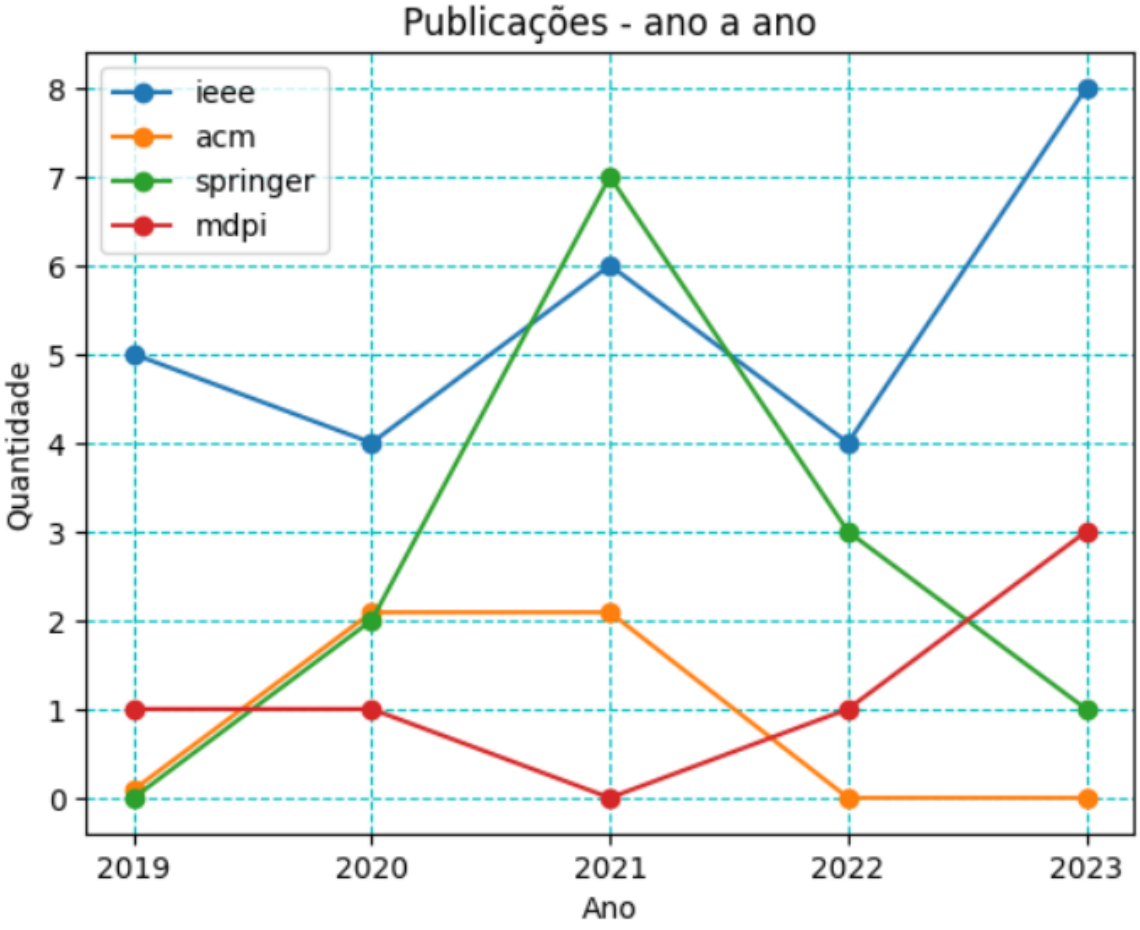
3. Na terceira etapa (3), por meio de leitura dos resumos para verificar os trabalhos mais importantes para pesquisa, definiram-se os critérios de exclusão e inclusão dos trabalhos pesquisados, separando apenas aqueles que continham combinações de classificadores e técnicas de balanceamento que melhoram os resultados após a aplicação visando a classificação de detecção de ataques de intrusão, excluindo os que trabalhavam com predição de defeitos em softwares, predições utilizando imagens.

4. Na etapa (4), contabilizou-se a contagem dos trabalhos para uma nova seleção, eliminando os trabalhos repetidos.

5. Por fim na etapa (5), realizou-se nova seleção de forma que os trabalhos menos relevantes que não faziam parte do foco da pesquisa, descartando-se após a leitura mantendo-se somente os artigos que davam ênfase para os métodos e técnicas de balanceamento para descoberta de ataques de intrusão. Manteve-se também aqueles que apresentavam uma metodologia adequada e instrumentos de pesquisas voltados para o presente trabalho.

   Na Figura 2 pode-se observar os trabalhos selecionados perante as plataformas de pesquisas:

Figura 2 – Distribuição dos artigos obtidos pelo ano de publicação.



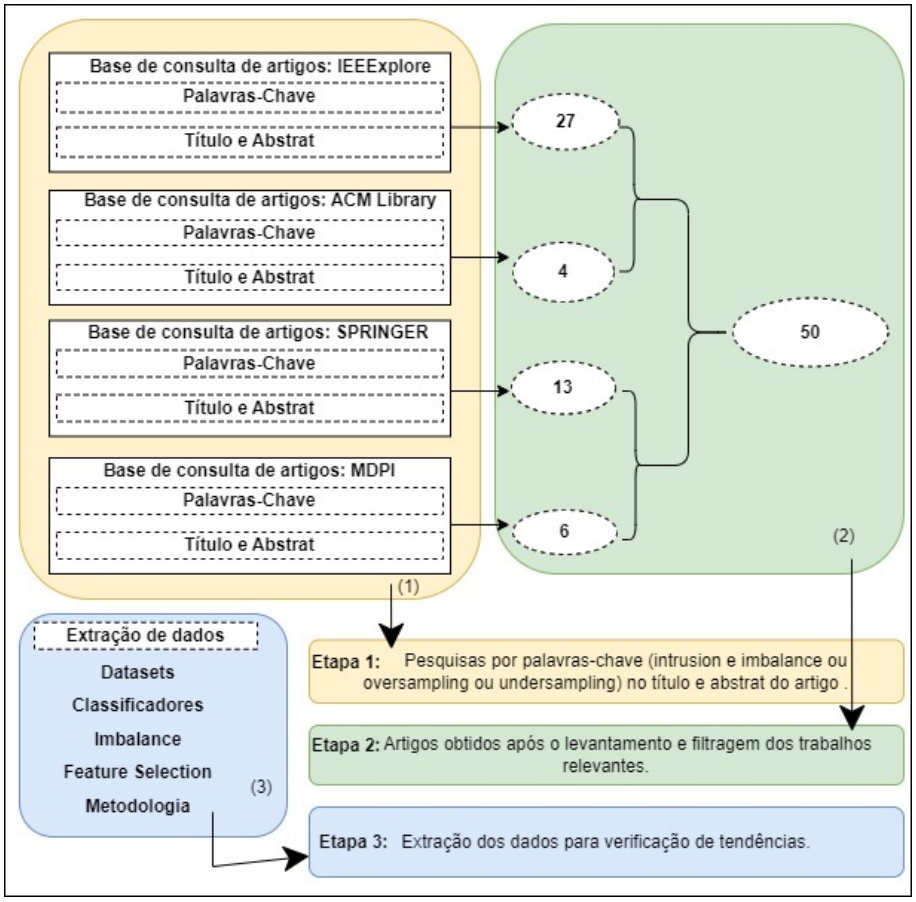
Fonte: Elaborado pelo autor

Com os trabalhos já selecionados e filtrados, procurou-se extrair os dados mais importantes observando as tendências envolvidas nos artigos como:

- Tipo de algoritmo de AM usados como classificadores;
- *Dados*: os conjuntos de dados utilizados nos trabalhos selecionados;
- As técnicas de *Feature Selection* aplicadas ou sem nenhuma aplicação (NA) de técnicas, ou não informado (NI) se foi aplicada alguma técnica para seleção de características, limpeza dos dados e redução da dimensionalidade e otimização dos dados;
- Quais técnicas/métodos foram usados para o balanceamento das classes: "*oversampling*" ou/e "*undersampling*".
- Quais métricas foram utilizadas pelos autores para avaliarem seus modelos e que resultados atingiram, como Acurácia (ACC), Precisão (P), *F1-Score*(F1); *Recall* ou Sensibilidade (S); e *Detection Rate*, por exemplo;

Para facilitar o entendimento do processo de Revisão Sistemática da Literatura, desde início até a escolha dos trabalhos mais relevantes, a Figura 3 exibe um fluxograma do processo adotado:

Figura 3 – Fluxo do procedimento realizado na Revisão Sistemática da Literatura.



Fonte: Elaborado pelo autor

Os artigos correlatos selecionados e resumidos para destacar os detalhes mais importantes na pesquisa foram organizados da seguinte maneira:

As Tabelas 1, 2, 3 e 4 apresentam os trabalhos correlatos e os dados extraídos para os anos de 2019, 2020, 2021, 2022 e 2023 podendo ser encontrados nas subseções 2.1.1, 2.1.2, 2.1.3 e 2.1.4 respectivamente.

2.1.1 Trabalhos Correlatos - ano de 2019

O artigo de Pang et al. (2019) tem como proposta um método de *oversampling* para detecção de *Malwares*[6], que agrupam ataques por características nominais, formando assinaturas utilizadas para identificar instâncias maliciosas. O problema relatado pelos autores é o desbalanceamento das classes causado pela classe normal ser dez por cento da classe maliciosa. Desta forma, como solução os autores aplicam o BorderlineSMOTE[7](BSMOTE) para equilibrar as classes e usam o *Support Vector Machine*[8](SVM) como classificador. Como resultado os autores conseguiram atingir uma taxa de 89,83% com a métrica *F1-Score*.

O trabalho de Jimoh, Ismaila e Olalere (2019), visa implementar a técnica de SMOTE[9] e utilizar AM para detecção de *Botnet*[10]. O classificador usado no trabalho foi o *J48 Decision Tree*[11] (J48DT) e o conjunto de dados CICIDS2017[12], segundo os autores, este conjunto possui um desbalanceamento das classes de 99:1. Após a aplicação do SMOTE, eles conseguem uma melhoria significativa na taxa de detecção com as métricas de avaliação de desempenho como Curva Operacional dos Receptores (ROC), Acurácia, Sensibilidade, *F-Measure*, Precisão, Taxa de Verdadeiros Positivos (TPR), Taxa de Falsos Positivos (FPR) e Coeficiente de Correlação Mathews[13] (MCC). Os autores comentam que devido a desigualdade das classes, o algoritmo de treinamento pode encontrar dificuldade em identificar os dados considerados benignos, podendo com isto aumentar a Taxa de Falsos Positivos. Outro apontamento feito pelos autores é que a MCC, como medida de precisão, é uma das melhores abordagens para os dados de desbalanceados. Os resultados mostraram-se satisfatórios com o uso do classificador J48DT com a técnica SMOTE de forma que atingiram 99,95% de Acurácia.

No trabalho de Lu et al. (2019), a proposta tem como finalidade usar uma combinação da técnica de *oversampling* SMOTE com a regra *Edited Nearest Neighbor*[14] (ENN) para

[6] Qualquer software intencionalmente feito para causar danos a um computador, servidor, cliente, ou a uma rede de computadores.
[7] Algoritmo que é uma variante do SMOTE detecta as amostras limite para gerar novas amostras sintéticas.
[8] Conjunto de métodos de aprendizagem supervisionada usados para classificação e regressão.
[9] Técnica de sobreamostragem Minoritária Sintética proposto por Chawla et al. (2002).
[10] Redes de terminais "escravos", que vão de computadores e celulares a dispositivos da Internet das Coisas, que hackers usam para cometer crimes dos mais diversos, de roubo de dados a ataques DDoS.
[11] Algoritmo de Árvores de Decisão.
[12] Conjunto de dados que compreende comportamento de tráfego de rede normal e de ataques conforme Sharafaldin, Lashkari e Ghorbani (2019).
[13] Medida de qualidade de duas classificações binárias.
[14] Técnica de *undersampling* de dados.

resolver o desbalanceamento do conjunto de dados KDDCup'99 e classificar os ataques com *Random Forest*[15] (RF). Como resultado conseguem alcançar valores maiores de Precisão, Sensibilidade e F1, atingindo 99% para as três métricas.

O método proposto por Phetlasy et al. (2019) para melhorar o reconhecimento do tráfego malicioso detectando ataques no conjunto de dados NSL-KDD é de aplicar classificadores sequenciais após o uso de SMOTE para equilíbrio das classes. Como resultado, os autores obtiveram a taxa de Acurácia de 89,95% usando J48DT + MLP[16] + SMOTE e 74,63% sem balanceamento com os mesmos classificadores. Também atingiram uma Sensibilidade 84,60%, Especificidade de 97,02% e a taxa de Falsos Negativos igual 1,97% com três classificadores aplicados sequencialmente J48DT + MLP + IBK[17] + SMOTE, contra 81,36% e 4,08% respectivamente classificados sem métodos de *oversampling*. Desta forma concluem que a combinação de três classificadores sequenciais juntamente com SMOTE podem reduzir falsos negativos e melhorar a Sensibilidade e a Acurácia.

No estudo de Pristyanto e Dahlan (2019), os autores aplicam a técnica de *resampling* utilizando a combinação dos algoritmos *One-Sided Selection*[18] (OSS) e SMOTE, em dois conjuntos de dados públicos do repositório *UCI Machine Learning Repository* para testes de performance de comparação em classificações de detecção de ataques de *phishing*[19]. Os classificadores utilizados no trabalho são *Decision Tree*[20] (DT), *k-Nearest-Neighbor*[21] (kNN), Redes Neurais Artificiais[22] (ANN) e SVM com os dados divididos na proporção de 80% para treino e 20% para testes tanto para classificação binária quanto para multiclasses. Como resultado, atingem um aumento satisfatório com as métricas de avaliação de Acurácia (96,26%) e G-Mean[23] (96,26%) usando ANN + SMOTE + OSS, com os dados originais apenas 95,91% e 95,88% respectivamente no primeiro conjunto de dados. Já no segundo conjunto a Acurácia obteve (92,87%) e G-Mean (94,64%) como o classificador DT, com os dados originais somente 88,82% e 90,35% na devida ordem.

O trabalho de Tan et al. (2019), tem como proposta usar o classificador RF com a técnica de SMOTE para identificar ataques de invasão no conjunto de dados KDDCup'99[24]. Os autores relatam que atingiram uma Acurácia de 92,39% sem o método de *oversampling*, já

[15] Algoritmo de árvore de decisão.
[16] Rede neural convolucional.
[17] Proposto por Aha, Kibler e Albert (1991) o algoritmo IBK (*Instance-based learning algorithms*) e um algoritmo do tipo *K-Nearest Neighbors* (kNN)
[18] Técnica de *undersampling* proposto por Kubat, Matwin et al. (1997).
[19] Crime de enganar as pessoas para compartilharem informações confidenciais como senhas e número de cartões de crédito.
[20] Algoritmo de aprendizagem supervisionado não paramétrico, utilizado para tarefas de classificação e regressão.
[21] Algoritmo de aprendizagem supervisionada usado no campo de *Data Mining* e *Machine Learning*.
[22] Técnicas computacionais que apresentam um modelo matemático inspirado na estrutura neural.
[23] A média de sensibilidade e especificidade que mede o desempenho geral do algoritmo de aprendizagem.
[24] Conjunto de dados usado para a Terceira Competição Internacional de Descoberta de Conhecimento e Ferramentas de Mineração de Dados.

com SMOTE atingiram 92,57%, superando até a classificação de outros classificadores como LibSVM[25], Naive Bayes[26].

Na Tabela 1 estão organizados os principais detalhes extraídos dos artigos publicados no ano de 2019.

As Tabelas 1, 2, 3, 4 e 5 estão organizadas da seguinte forma:

- Artigo - O trabalho que se refere os dados;
- Dados - Os conjuntos de dados usados no artigo;
- Classificador - Os classificadores usados no artigo;
- Técnica - A técnica de balanceamento dos dados aplicadas no trabalho;
- SC - A técnica de seleção de característica aplicadas no trabalho (NI - Não Informada, NA - Não Aplicada);
- Métricas - As métricas utilizadas para avaliação do modelo proposto no trabalho.

Tabela 1 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2019. Fonte: Elaborado pelo autor.

Artigo	***Dados***	**Classificador**	**Técnica**	**SC**	**Métricas**
Pang et al. (2019)	Próprio	SVM	BSMOTE	NI	AUC; F1; G-Mean
Jimoh, Ismaila e Olalere (2019)	CICIDS2017	J48DT	SMOTE	NI	ACC; AUC; F1; P; S;
Lu et al. (2019)	KDDCup99	RF	SMOTEENN	NI	F1; P; S
Phetlasy et al. (2019)	NSL-KDD	J48DT; MLP; IBK	SMOTE	NI	ACC; E; S
Pristyanto e Dahlan (2019)	UCI-MLR	kNN; SVM; ANN; DT	SMOTE+OSS	NA	ACC; G-Mean
Tan et al. (2019)	KDDCup99	RF	SMOTE	NA	ACC; AUC

[25] Bibliotecas de aprendizagem de máquinas de código aberto.
[26] Classificador probabilístico baseado no "Teorema de Bayes".

2.1.2 Trabalhos Correlatos - ano de 2020

A presente subseção descreve os trabalhos relacionados obtidos por meio de Pesquisa de Revisão Sistemática da Literatura para o ano de 2020.

A proposta de Prayogo e Karimah (2020) apresenta uma abordagem para classificar ataques de *phishing* utilizando a aplicação de SMOTE para resolver o desequilíbrio das classes. O conjunto de dados utilizado no estudo abrange 11.055 instâncias com 30 características disponibilizado pela UCI *Machine Learning Repository*. No trabalho, os autores informam que é feito uma seleção de características reduzindo de 30 para 12, mas não esclarecem o método utilizado para escolha. O classificador aplicado foi o kNN com k=5, e as métricas utilizadas para avaliação do modelo são a Matriz de Confusão, Acurácia, Precisão, Sensibilidade e F-Measure. Os resultados encontrados pelos autores apontam que neste caso, kNN com SMOTE consegue atingir uma Acurácia de 97,47%, Precisão 97,50%, Sensibilidade 97,50% e F1-Score de 97,50%, melhorando efetivamente a classificação em comparação com os dados originais.

Já no trabalho de Alfrhan, Alhusain e Khan (2020), fizeram uso a ferramenta *Azure Microsoft Machine Learning Studio* para aplicar os algoritmos de AM (NB, RF, kNN e SVM) e fazer a classificação e identificação de ataques no conjunto de dados CICIDS2017. Como o conjunto de dados possui um grande desbalanceamento das classes, aplicam a técnica de SMOTE, para resolver o problema e comparar os resultados. O trabalho é dividido em duas etapas: na primeira são realizados os testes com os algoritmos de AM sem o uso do SMOTE. Já na segunda etapa aplicam o SMOTE e logo após comparam os resultados obtidos com as métricas de Precisão, Sensibilidade e F1-Score e Curva Roc (ROC). Nos resultado, observam melhorias com todas as métricas de avaliação utilizadas no trabalho atingindo taxas de 100% de reconhecimento de ataques, sendo que o algoritmo SVM teve o melhor desempenho. Com os dados originais os resultados foram 98,20%, 99,10% e 100% respectivamente com as mesmas métricas. Os autores também ressaltam a afirmação que a medida F1-Score é a melhor avaliação quando se trabalha com o conjunto de dados desbalanceados.

O trabalho de Sun et al. (2020), visa como proposta de balanceamento de dados o método de *oversampling* para igualar a quantidade de dados de tráfego benigno e de ataques existentes nos conjuntos de dados UNSW-NB15[27] e CICIDS-2018[28]. Apontam que existem dois desafios-chave em detecção de intrusão no mundo real, dados desbalanceados onde a maioria dos dados é composto de tráfego normal e fluxo provindo de diferentes terminais de rede. Como solução, apresentam um método em que aplicam vários algoritmos de AM: *Support Vector Machine* (SVM), *Decision Tree*[29] (DT), *k-Nearest Neighbor* (kNN) e *AdaBoost* (AB) e três algoritmos MTL (*monoidal t-norm based logic*[30]). Logo após, aplicam o algoritmo

[27] Conjunto de dados de ataques de rede criado pelo Centro Australiano de Segurança Cibernética.
[28] Conjunto de dados de ataques de rede criado pelo Instituto Canadense de Segurança Cibernética.
[29] Algoritmo de árvore de decisão.
[30] Algoritmos que utilizam funções chamadas de t-normas para possíveis interpretações de conjunção lógica.

Standard Template Library[31](STL) para comparação. Para o desbalanceamento dos dados aplicam o SMOTE juntamente com o A-SUWO[32] e para verificação dos resultados fazem uso das métricas F1-Score, Taxa de Falsos Negativos (FNR) e Taxa de Falsos Positivos (FPR). Os resultados mostraram que o VSTG-MTL e FSP-VSTG-MTL com SMOTE têm os melhores métodos desempenho para detecção de intrusão atingindo F1-Score 77,58%; FNR 9,95%; FPR 56,52%.

No artigo de Pawlicki et al. (2020), a proposta é avaliar formas de equilibrar a quantidade de dados benignos e de ataques para utilizar classificadores de AM como Rede Neural Artificial (RNA), *Random Forest* e um classificador *Naive Bayes* em aplicações de cibersegurança. Segundo os autores, os testes realizados com os métodos de *undersampling* (NearMiss[33]; TomekLinks[34]; ClusterCentroids[35](CC) e RandomUnderSampler[36] (RUS)) juntamente com o conjunto de dados CICIDS2017, alcançaram resultados ótimos ou melhores do que os de *oversampling*, equiparando ao mesmo nível que o método de BSMOTE. A média da Acurácia para detecção de ataques com RF+RUS usando até 1.174 instâncias por classe atingiu 96,57% e com até 7.141 instâncias 98,72% respectivamente. Também afirmaram que existe uma ligação forte entre o tamanho da classe majoritária, a Precisão e o Sensibilidade apresentada pelo número de amostras falsamente classificadas como da classe minoritárias.

A proposta de Pan e Xie (2020), baseia-se em fazer uma classificação de ataques de intrusão com o algoritmo *XGBoost*[37](XGB) e compará-la com os algoritmos de *Support Vector Machine*, *Random Forest*. Para a preparação dos dados usando o conjunto de dados KDDCup'99, os autores utilizam-se PCA[38] para selecionar as características e diminuir a redundância dos dados. Para resolver o problema do balanceamento dos dados no conjunto de dados utilizam a aplicação de ADASYN[39]. Os resultados apontados pelos autores mostrou que é possível reduzir a taxa de Falsos Positivos (FPR) do modelo de detecção das amostras minoritárias e otimizar a taxa F1-Score. As taxas de FPR do modelo proposto para amostras minoritárias classificando ataques de R2L e U2R foram de 17,3% e 19,7%, e a métrica F1 alcançou 90,10% e 84,50% com o classificador XGB.

Os autores Zhang, Zhang e Li (2020), usaram em seu trabalho o algoritmo ReliefF[40] juntamente com BSMOTE, e como classificador utilizaram três tipos diferentes de algoritmos combinados, kNN, NB e C4.5[41]. O conjunto de dados selecionado para o trabalho foi o

[31] Algoritmo usado para aplicar ruído gaussiano aditivo centrado em zero.
[32] Método de *oversampling Adaptive Semi-Unsupervisioned Weighted Oversampling*.
[33] Algoritmos de *undersampling* tem como função eliminar aleatoriamente amostras da classe majoritária.
[34] Algoritmo de *undersampling* para classificação desbalanceada.
[35] Técnica de *undersampling* que geram um novo conjunto baseado em centróides por métodos de agrupamento.
[36] Algoritmo de subamostragem aleatória (RUS)
[37] Algoritmo de AM, baseado em árvore de decisão que utiliza uma estrutura de *Gradient boosting*.
[38] *Principal Component Analysis* método de seleção de características.
[39] Método de *oversampling Oversample using Adaptive Synthetic* .
[40] Algoritmo para seleção de *features*.
[41] Algoritmo utilizado para criar uma árvore de decisão.

NSL-KDD. Os resultados experimentais apresentaram um bom desempenho no conjunto e atingiram uma melhora considerável da Acurácia com a combinação de kNN com C4.5 atingindo 99,89%, 99,91%, 99,89%, 99,96% e 99,98% para as classes Normal, DoS, R2L, Probe e U2R, respectivamente. A métrica F1 usando a mesma combinação, obteve a atuação de 99,90%, 99,88%, 97,67%, 99,79% e 98,69%, também respectivamente para as mesmas classes testadas, apresentando superioridade para com as outras combinações do trabalho.

Engly, Larsen e Meng (2020), tem como proposta utilizar AM para identificar ataques no conjunto de dados NSL-KDD. Desta forma, os autores realizaram experimentos com RF, ANN e *Gradient-Boosting Machines*[42](GB). Para ajustar o desequilíbrio das classes usam o SMOTE e para reduzir as *features* utilizam-se de métodos de seleção de características como *Correlation Based Feature Selection*[43] (CFS), *Fast Correlation-based Filter*[44] (FCBF) e seleção baseada na consistência. Como resultado de seus experimentos, afirmam que o melhor desempenho alcançado foi usando *Ensemble Boosting* com duas redes neurais e GB, atingindo uma taxa de Acurácia de 81% e F1-Score de 68%, uma melhoria de 10% e 24% respectivamente, em comparação com o treinamento do modelo no conjunto de dados original.

Os autores Gonzalez-Cuautle et al. (2020), citam em seu artigo que AM tem melhorado as práticas clássicas de detecção de intrusão, mas que os dados demasiadamente desbalanceados gerados pelos Sistemas de Detecção de Intrusão (IDS) podem produzir resultados tendenciosos em relação a classe majoritária (*Overfitting*) [45]. Como solução apresentam uma proposta da aplicação da técnica de SMOTE para ajustes em dois conjuntos de dados (CIDDS-001[46] e ISCX-Bot-2014[47]), logo após fazem a divisão de (80-20) para treino e teste respectivamente. Prosseguem realizando a extração das *features* mais pontuadas com PCA e classificam os dados com *k-Nearest-Neighbor* (kNN); *Support Vector Machine*; *Logistic Regression*[48] (LR); *Decision Trees* (DT); e *Random Forest* . Nos resultados, concluem que, ao testar vários classificadores juntamente com a técnica SMOTE e a otimização de parâmetros *Limited Broyden-Fletcher-Goldfarb-Shanno*[49] (LBFGS) foram fatores essenciais para conseguirem desempenhos satisfatórios na classificação. Os autores citam que em comparação com outras abordagens que utilizaram os mesmos conjunto de algoritmos na literatura, o modelo proposto por eles atingiram melhores resultados. O conjunto de algoritmos, SMOTE + LBFGS apresenta a seguinte Acurácia com respectivos algoritmos de AM: kNN 98,72%, SVM 97,35%, LR 97,89%, DT 98,65%, e RF 98,84%. Também a Característica Operacional do Receptor (ROC) para o

[42] Algoritmo de AM, baseado em árvore de decisão
[43] Algoritmo de estratégia de busca heurística.
[44] Método de seleção de características.
[45] Termo usado em estatística para descrever quando um modelo estatístico se ajusta muito bem ao conjunto de dados
[46] Conjunto de dados de avaliação para sistemas de detecção de intrusão de rede baseados em anomalias criado pela Universidade de Coburgo na Alemanha.
[47] Conjunto de dados de ataque fornecido pelo Instituto Canadense de Segurança Cibernética com 16 tipos diferentes de botnets.
[48] Modelo estatístico usado para determinar a probabilidade de um evento acontecer.
[49] Algoritmo de otimização para obter melhores resultados com classificadores.

subconjunto balanceado CIDDS-001 obteve 99,58% com kNN e para subconjunto balanceado ISCX-Bot-2014 atingiu 96,26% com RF.

Karatas, Demir e Sahingoz (2020) realizam primeiramente a análise de seis conjuntos de dados para escolherem qual utilizar em seu trabalho, logo após ao escolherem o conjunto de dados CICIDS2018 por ser mais atualizado, aplicam a técnica SMOTE, classificando os dados com seis classificadores individualmente (GB, Análise Discriminante Linear[50] (LDA), kNN, RF, DT e AB) e comparam com os resultados aplicados no conjunto de dados original. Os resultados mostram que as taxas de detecção obtiveram um aumento substancial de 4,01% e 30,59% em comparação com os dados originais. As melhores pontuações de Acurácia foram para AB 99.99% na identificação das classes de ataques (Botnet, DoS, Brute Force) e 100% em (SQLInject); RF 99.99% nas classes de ataques (DoS, Brute Force e SQLInject); kNN 100% na identificação de (SQLInject) e GB 99.99% em (DoS) e 100% para (SQLInject).

Tabela 2 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2020. Fonte: Elaborado pelo autor.

Artigo	***Dados***	**Classificador**	**Técnica**	**SC**	**Métricas**
Prayogo e Karimah (2020)	UCI-MLR	kNN	SMOTE	NI	ACC; F1; P; S
Alfrhan, Alhussain e Khan (2020)	CICIDS2017	DF; DJ; SVM	SMOTE	NI	AUC; F1; S
Sun et al. (2020)	UNSW-NB15; CICIDS2018	SVM; DT; kNN; AB; VSTG-MTL	SMOTE; A-SUWO	NI	F1; FNR; FPR
Pawlicki et al. (2020)	CICIDS2017	RNA; RF; NB	NearMiss; RU; TomekLinks; CC; BSMOTE	NA	ACC; F1; P; S
Pan e Xie (2020)	KDDCUP99	XGB; SVM; RF	ADASYN	PCA	F1; FPR
Zhang, Zhang e Li (2020)	NSL-KDD	kNN; C4.5; NB	BSMOTE	ReliefF	F1; P; S; FPR; ACC; G-Mean
Engly, Larsen e Meng (2020)	NSL-KDD	RF; ANN; GB	SMOTE	CFS; FCBF	ACC; F1; P; S

[50] Método usado em estatística e outros campos, para encontrar uma combinação linear de recursos que caracteriza duas ou mais classes de objetos, ou eventos.

Tabela 2 – Continuação.

Gonzalez-Cuautle et al. (2020)	CIDDS-001; ISCX-Bot-2014	kNN; SVM; LR; DT; RF	SMOTE	PCA	ACC; AUC; F1; P; S
Karatas, Demir e Sahingoz (2020)	CICIDS2018	kNN; RF; GB; AB; DT; LDA;	SMOTE	NA	ACC; F1; P; S

Tabela 2 – Fim.

2.1.3 Trabalhos Correlatos - ano de 2021

Os autores Mohseni e Tanha (2021), afirmam em seu trabalho que o desbalanceamento dos dados das classes benigna e de ataques gerados pelos IDSs, dificultam o trabalho de classificação correta dos algoritmos tradicionais que tentam descobrir características na classe minoritária. Também citam, que a técnica de *undersampling* se ajusta melhor do que a de *oversampling* que leva ao sobreajustamento dos dados. Desta forma, fazem uma proposta de agrupamento (*clustering*) baseada em densidade, depois aplicam a técnica para balanceamento *RUSBoost Classifier*[51] no conjunto de dados CICIDS2017. Usando 20000 exemplos, realizam a seleção de características com PCA para reduzir de 83 para 30 *features* e aplicam o classificador RF com os dados nas proporções (1:15, 1:25, 1:50, e 1:100). Como melhor resultado, atingem 99% de *G-Mean* e 99,70% Acurácia na proporção de (1:15).

No trabalho de Anh et al. (2021), a proposta é a construção de um novo conjunto de dados de intrusão baseado no CICIDS-2017 para resolver o desbalanceamento das classes. Para isto fazem uso do simulador GNS3[52] e ferramentas de ataques, os autores produziram um tráfico de dados com as mesmas características do CICIDS-2017. Neste modelo de proposta, eles aumentam o tráfego malicioso para 40% dos dados originais e assim resolveram o problema de desbalanceamento das classes. Após a criação do conjunto de dados, os autores realizam uma limpeza dividindo os dados em treino e testes numa proporção 80% para treino e 20% para testes. Os classificadores usados são o kNN com k=5, *AdaBoost*, *Random Forest*, Redes Neurais (*Multilayer Perceptron*[53]) MLP, *convolutional neural network* (CNNID), *Long Short-Term Memory*[54](LSTM) e testam seus modelos com as métricas de Acurácia, Precisão, Sensibilidade e *F1-Score* calculados a partir da Matriz de Confusão. Os resultados conseguidos pelos autores mostram uma melhora em relação ao conjunto de dados original, atingindo 99% de Acurácia, *F1-Score* e Sensibilidade, com o classificador RF.

A proposta de Szczepanski et al. (2021), investiga o impacto dos métodos que fazem o

[51] Algoritmo para lidar com problemas de desequilíbrio de classe em dados com rótulos de classe discretos.
[52] Simulador de redes, que emula diversos equipamentos de uma rede.
[53] Rede neural com uma ou mais camadas ocultas.
[54] Rede neural recorrente capaz de aprender a dependência da ordem de aprendizagem.

balanceamento de dados equilibrando as classes no conjunto de dados. Para isto, os autores utilizam o conjunto de dados IoT-23[55] e aplicam SMOTE como método de *oversampling*, *Random Under-Sampling*[56] (RUS) para *undersampling* e SMOTEENN[57] como *resampling* e assim realizar uma comparação entre elas. Logo em seguida, realizam a classificação por meio de uma Rede Neural Artificial (ANN) de duas camadas e finalizam utilizando as métricas de Acurácia, Precisão e Sensibilidade para verificação dos resultados. Segundo os autores, embora o modelo treinado com o conjunto de dados desbalanceado tenha alcançado uma Acurácia superior, não garante que o resultado esteja corretamente classificado pelo algoritmo, isto acontece em uma situação com grandes discrepâncias entre as quantidades de amostras, caracterizando ser uma métrica muito enganosa (*Overfitting*). Observando as outras métricas, torna-se claro que praticamente classificou todas as amostras como benignas. Os resultados obtidos com a Acurácia foram 99% com os dados originais, 91% aplicando os métodos de *oversampling*, também 91% com *undersampling* e 97% com o método híbrido SMOTEENN.

O artigo de Wang e Septian (2021), tem como proposta uma combinação de algoritmos de *Deep Learning Neural Networks*[58] baseados em *Long short-term memory* (LSTM) com técnicas de SMOTE para detecção de intrusão nos conjuntos de dados CICIDS-2017 e CICIDS-2018. Após a limpeza dos dados, os autores realizam a seleção de características a partir de árvores de decisão com Oracle-explainer[59] avaliando os recursos selecionados com LIME[60] e a aplicação de SMOTE para equilibrar os dados. Em seguida, dividem os dados em 80% para treino e 20% para testes, classificando-os com Redes Neurais Recorrentes (RNNs) e LSTM em busca de melhores resultados. Segundo os autores, o método de *oversampling* melhora o desempenho da detecção de intrusão tanto para o AM como para métodos de Aprendizagem Profunda. Nos resultados, os autores obtiveram melhor desempenho com o conjunto de dados CICIDS-2017 e LSTM atingindo *F1-Score* de 98,90%, já com CICIDS-2018 usando as mesmas técnicas conseguiram uma pontuação *F1-Score* de 98,80%.

No trabalho de Alshamy et al. (2021), a proposta tem como base a aplicação dos classificadores RF, AB, LR, e SVM no conjunto de dados NSL-KDD para verificar a melhor atuação entre os algoritmos. Devido à desigualdade da quantidade de dados das classes, os autores aplicaram a técnica SMOTE. Como resultado, na classificação binária alcançaram a Acurácia de 99,89%, 99,26%, 99,13%, e 97,26% respectivamente. Na classificação multiclasses os resultados foram 99,88%, 99,82%, 87,23% e 75,86% também respectivamente com os mesmos classificadores e a mesma métrica. Os autores consideram que o modelo proposto por eles obteve excelentes resultados em comparação com outros encontrados na literatura, tanto

[55] Conjunto de dados de tráfego de rede de dispositivos da Internet das Coisas (IoT).
[56] Método de *undersampling* que escolhe aleatoriamente amostras com ou sem reposição.
[57] Combina a abordagem de *oversampling* do SMOTE com *undersampling* de Edited Nearest Neighbors.
[58] Algoritmo que utiliza técnica de inteligência artificial conhecida como Aprendizado Profundo.
[59] Informa quais dados devem ser usados para calcular a Zona de Eficiência ou módulo de Comparação de Vizinhos e fornece maneiras de melhorar ou corrigir os recursos utilizados.
[60] Efetua a interpretação dos dados na tomada de decisão dos modelos de algoritmos de AM.

para classificação binária como para multiclasses.

Já os autores Al-Imran e Ripon (2021), utilizam o conjunto de dados Kyoto Honeypot[61] com a proposta de melhorar a identificação de intrusão usando classificadores de *Deep Learning*. Fazendo uma minuciosa preparação dos dados, os pesquisadores realizam a limpeza, indexação dos dados categóricos, normalização e discretização atribuindo uma etiqueta a cada valor numérico para manter uma relação ordenada e melhorar o desempenho do modelo. Na seleção de características, fazem uso do algoritmo MRMR (*Minimum Redundancy Maximum Relevance*[62].), reduzindo de 24 para 19 características. Em seguida, equilibram as classes aplicando SMOTETomek[63] que combina a técnica de *oversampling* com a de *undersampling* e separam os dados em 70% para treino e 30% para testes. Segundo os autores, os melhores classificadores foram o algoritmo *XGBoost* que atingiu Acurácia de 99,08% e RF, que também atingiu uma Sensibilidade de 98,65%, sendo superiores aos outros classificadores, apesar dos modelos de redes neurais, LSTM e *Gated Recurrent Unit*[64](GRU) também conseguiram bons resultados.

Wang et al. (2021), realizam uma combinação de *Sparse Auto-Encoder*[65] (SAE) para extração da *features* e RF para classificação de ataques de detecção de intrusão. Para uma melhora nos resultados, aplica-se a técnica de ADASYN ajustando o desbalanceamento dos dados no conjunto de dados de referência NSL-KDD. A divisão para treino é teste adotada é de 80% e 20% respectivamente, feita após a limpeza e normalização nos dados. Para avaliar os resultados, os autores comparam com vários métodos de AM comumente utilizados na literatura como *Naive Bayesian*, *Support Vector Machine*, *Decision Tree* e *k-Nearest Neighbor*. Os resultados mostraram que o modelo proposto pelos autores usando SAE e RF atingiram 99,80%, 99,78%, 99,77%, 99,77% de Precisão, F1, Acurácia e Sensibilidade respectivamente, seguido dos classificadores kNN, DT, SVM e NB que obtiveram resultados de ao menos 2 pontos percentuais abaixo.

No artigo de Seth, Singh e Chahal (2021), a proposta é classificar ataques de intrusão selecionando as características com uma abordagem híbrida utilizando PCA e RF. O conjunto de dados usado no trabalho foi o CICIDS-2018 que possui 80 *features*, e um alto desbalanceamento das classes. Nesta proposta, os autores reduziram as *features* para 24 e solucionaram o problema de desbalanceamento com a técnica de IsolationForest[66] com método de *undersampling* a um nível aceitável. Na etapa de classificação foi utilizado cinco algoritmos de AM, porém o que apresentou melhores resultados foi o LightGBM[67] com uma taxa de Acurácia de 97%, Precisão 99% e uma taxa de Sensibilidade de 96% com uma baixa latência de previsão. O modelo proposto pelos autores aumentou 1,5% na taxa de Acurácia e 3% na Precisão em relação aos

[61] Conjunto de dados de tráfego de rede dos Honeypots da Universidade de Kyoto.
[62] Método de seleção de características Relevância máxima - Redundância mínima.
[63] Técnica que combina o SMOTE e Tomek Links.
[64] Visa resolver o problema da dissipação do gradiente comum em uma rede neural recorrente padrão.
[65] Tipo de rede profunda que pode ser usada para a redução da dimensionalidade.
[66] Algoritmo para detecção de anomalias de dados desenvolvido inicialmente por Fei Tony Liu em 2008.
[67] Uma estrutura de reforço de gradiente que utiliza algoritmos de aprendizagem baseados em árvores.

outros classificadores testados no trabalho.

Os autores Chen, Zhou e Yu (2021), relatam que os conjuntos de dados de detecção de intrusão são propensos a um baixo desempenho de classificação devido ao desbalanceamento das classes de ataques e normal. Desta forma, sugerem o uso de ADASYN e RF para classificação no conjunto de dados CICIDS2017. Os resultados apresentados nos experimentos mostraram uma melhoria, com as métricas de Precisão de 98,50%, Sensibilidade de 92,30%, *F1-Score* de 95,30 e valores de AUC de 98,78% após o uso de ADASYN, comparando com SMOTE que obteve 98,39%, 91,72%, 94,94% e 99,74% com as mesmas métricas respectivamente. A combinação de RF+ADASYN também se mostrou superior a técnica RUS e para com os dados originais.

No trabalho de Chindove e Brown (2021), os autores afirmam que devido ao tamanho e à aplicabilidade dos conjuntos de dados de detecção de intrusão, as técnicas de AM são essenciais para classificação do tráfego de redes. Diante deste fator, eles avaliam os conjuntos de dados CICIDS2017 e CICIDS2018, com as técnicas de seleção de *features Gini Importance*[68] (GI), *Permutation Importance*[69] (PI) e PCA e os classificadores MLP, SVM, KNN, DT, RF e Rede Neural Recorrente (RNN), juntamente com SMOTE. Os melhores resultados, apresentaram as taxas de *F1-Score* de 73% com a combinação RNN + PI e 87% com RF + PI, aplicado no conjunto de dados CICIDS2017. Já no CICIDS2018 as taxas com *F1-Score* foram de 73% e 72%, utilizando respectivamente as mesmas combinações.

O trabalho de Bagui e Li (2021), afirma que a utilização de dados desbalanceados para classificar ataques com Redes Neurais Artificiais é um grande desafio, devido a enorme influência da classe majoritária, especialmente em classificação de multiclasses. Os autores também relatam em seus estudos que a técnica de *oversampling* tende a aumentar o tempo de treinamento e a de *undersampling* tende a diminuir, já *resampling* quase não impacta no tempo de treinamento e também detecta mais dados da classe minoritária. O modelo proposto por eles é a aplicação de combinações das técnicas de *undersampling* e *oversampling* em seis conjuntos de dados com cinco técnicas de reamostragens diferentes e classificá-los com Rede Neural Artificial (ANN). Os autores concluíram em seus experimentos, que para os conjuntos de dados que possuem um grande desbalanceamento de classes a técnica de *resampling* é a mais adequada para ser aplicada, mas em conjuntos com um baixo desbalanceamento de classes a melhora é pouco significativa. A técnica RURO[70] foi a que teve o melhor desempenho para identificação de dados da classe minoritária, apesar da diferença para com as técnicas de RU-SMOTE[71] e RU-ADASYN[72] serem mínimas. A métrica de Sensibilidade Macro de RURO no conjunto de dados KDD'99 foi a mais alta atingindo 96%, enquanto a Sensibilidade Macro

[68] O coeficiente de Gini mede a desigualdade entre os valores de uma distribuição de frequência.
[69] Técnica que mede a contribuição de cada recurso para o desempenho estatístico de um modelo.
[70] Técnia de *resampling* que utiliza subamostragem aleatória com técnica de sobreamostragem aleatória.
[71] Técnica de *resampling* que utiliza subamostragem aleatória com técnica de sobreamostragem minoritária sintética.
[72] Técnica de *resampling* que utiliza subamostragem aleatória com método de amostragem sintética adaptativa.

de RU-SMOTE e RU-ADAYSN ficou em 95,59%, já com RU o valor foi de 90,5% mostrando-se inferior aos outros métodos de *resampling*, mas muito melhor do que com os dados originais sem aplicações de técnicas de balanceamento com 73% de Sensibilidade.

Já no trabalho de Varunram et al. (2021), a proposta considera uma classificação binária de ataques DDoS[73] no conjunto de dados CICIDS2017, com os algoritmos kNN, AB, SVM, LR, RF, RNN e NB juntamente com SMOTE. Também são aplicadas três técnicas de redução e seleção de características PCA, t-SNE[74] e UMAP[75]. Logo após, fazem uma comparação dos resultados, para descobrir qual classificador e técnica de redução de dimensionalidade combinadas apresentam melhor performance em AM. No trabalho proposto, a Precisão média dos modelos com a aplicação das técnicas de redução de dimensionalidade aumentou entre 4,01% e 30,59% respectivamente. A combinação que apresentou melhor desempenho foi kNN com Acurácia, Precisão, Sensibilidade e F1 de 99,98%, nos dados reduzidos do t-SNE. Quanto ao tempo gasto de treino, observaram que kNN e RF se destacam dos demais, com apenas 3 minutos cada.

O estudo de Sapre, Islam e Ahmadi (2021) apresenta uma proposta para classificar dois tipos de ataques de intrusão: *root-to-local*[76] (R2L) e *user-to-root*[77] (U2R) no conjunto de dados NSL-KDD. Dentro da ideia, utilizam *oversampling* com *Generative Adversarial Networks*[78] (GAN), *oversampling* com kNN e *undersampling* com *NearMiss* para balancear os dados e verificarem os resultados da classificação feita com Redes Neurais Artificiais. Os resultados mostraram que os melhores desempenhos apresentados para ataques U2R e R2L com a métrica *F1-Score* de 11% e 22%, respectivamente foram com os dados originais sem modificação. Já a classificação utilizando *oversampling* baseado em GAN conseguiu a maior taxa de Sensibilidade atingindo 94% na classificação da classe U2R. E o uso de *undersampling* com *NearMiss* obteve a melhor Sensibilidade para com a classe R2L atingindo 18%.

O artigo de Tauscher et al. (2021) propõem uma classificação binária no conjunto de dados NSL-KDD com vários classificadores (DT, RF, NB, SVM, AB, GB, MLP e Autoencoder com 3 camadas e 15 neurônios), balanceando os dados com SVM-SMOTE[79]. Os resultados mostraram que o classificador SVM atingiu a melhor Precisão com 97,56%, GB a melhor Sensibilidade com 95.13% e Autoencoder produz a melhor Acurácia e *F1-Score* com 87,52% e 88,48% respectivamente.

O trabalho de Jui et al. (2021) baseia-se na aplicação de AM utilizando combinações de técnicas de pré-processamento, seleção de características, *oversampling*, *undersampling* e

[73] Ataques de negação de serviço distribuído.
[74] Algoritmo de aprendizado de máquina usado para visualização e também para redução de dimensionalidade.
[75] Técnica de redução de dimensionalidade baseada em gráficos, introduzida em 2018 por Leland McInnes.
[76] Ataque para obter acesso não autorizado a uma máquina vítima na rede.
[77] Ataque para obter ilegalmente os privilégios de root ao acessar legalmente uma máquina local.
[78] Classe de estruturas de AM com uma estrutura proeminente para abordar a IA generativa.
[79] Variante do algoritmo SMOTE que utiliza um algoritmo SVM

de algoritmos de classificação, nos conjunto de dados MQTT-IoT-IDS-2020[80] e NSL-KDD. Primeiramente, aplicam a normalização dos dados, já em uma segunda etapa aplicam SMOTE e *undersampling*. Na terceira etapa utilizam a seleção de recursos com *Best First Search*[81](BFS) e *Genetic Search*[82]. Por fim, realizam a classificação com os algoritmos NB, *Bagging*, *AdaBoost*, J48 e RF. Nas diversas combinações aplicadas concluíram que para o conjunto de dados MQTT-IoT-IDS-2020, o algoritmo *AdaBoost* juntamente com BFS para técnica de seleção de recursos e J48 como algoritmo de classificação apresentam o melhor desempenho (99,86% de Acurácia em 2,81 segundos). Já para o conjunto de dados NSL-KDD, as combinações de *oversampling*, *Bagging* e *Genetic Search* apresentam o melhor desempenho (84,35% de Acurácia em 0,32 segundo).

Tabela 3 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2021. Fonte: Elaborado pelo autor.

Artigo	***Dados***	**Classificador**	***Técnica***	**SC**	**Métricas**
Mohseni e Tanha (2021)	CICIDS2017	RF	RUSBoost	PCA	ACC; G-Mean
Anh et al. (2021)	CICIDS2017	kNN; AB; RF; MLP; CNNID; LSTM	Própria	NI	ACC; F1; S
Szczepanski et al. (2021)	IoT-23	ANN	SMOTE; RUS; SMOTEENN	NA	ACC; P; S
Wang e Septian (2021)	CICIDS2017; CICIDS2018	RNNs; LSTM	SMOTE	Oracle-explainer	ACC; P; S; F1
Alshamy et al. (2021)	NSL-KDD	RF; SVM; LR; AB	SMOTE	NA	ACC; F1; P; S
Al-Imran e Ripon (2021)	Kyoto 2013	RF; XGB; LSTM; GRU	SMOTETomek	MRMR	ACC; F1; MCC; P; S
Wang et al. (2021)	NSL-KDD	RF	ADASYN	SAE	ACC; F1; P; S
Seth, Singh e Chahal (2021)	CICIDS2018	RF; ET; XGB; GB; KNN; LightGBM	IF	PCA; RF	ACC; P; S

[80] Conforme Hindy et al. (2020), é um conjunto de dados com base no protocolo Message Queuing Telemetry Transport (MQTT) usado para comunicação de dispositivos IoT.
[81] Método de seleção de recurso que pesquisa o espaço do subconjunto de recursos usando escalada gananciosa.
[82] Método que realiza uma pesquisa simples dos atributos usando um algoritmo genético.

Tabela 3 – Continuação.

Chen, Zhou e Yu (2021)	CICIDS2017	RF	ADASYN; SMOTE; RUS	NI	AUC; F1; P; S
Chindove e Brown (2021)	CICIDS2017; CICIDS2018	MLP; SVM; KNN; DT; RF; RNN	SMOTE	GI; PI; PCA	ACC; F1; P; S
Bagui e Li (2021)	KDD99; UNSW-NB15; UNSW-NB17; UNSW-NB18	ANN	SMOTE; RU-SMOTE; ADASYN; RU-ADAYSN; RU; RO; RURO	NA	F1; P; S
Varunram et al. (2021)	CICIDS2017	RF; kNN; AB; LR; SVM; ANN; NB	SMOTE	PCA; t-SNE; UMAP	ACC; F1; P; S
Sapre, Islam e Ahmadi (2021)	NSL-KDD	RNA	GAN; kNN (*Oversampling*); NearMiss	NA	S
Tauscher et al. (2021)	NSL-KDD	DT; RF; NB; SVM; AB; GB; MLP	SVM-SMOTE	NA	ACC; F1; S
Jui et al. (2021)	MQTT-IoT-IDS-2020; NSL-KDD	NB; AB; J48; RF	SMOTE	BFS	ACC

Tabela 3 – Fim.

2.1.4 Trabalhos Correlatos - ano de 2022

Na proposta de Nagpal, Kaushal e Sharma (2022), a identificação de ataques de detecção de intrusão baseia-se em usar o classificador SVM de modo otimizado juntamente com o *Big Bang-Big Crunch*[83] (BBBC), aplicando uma redução de *features* e fusão de características com *Information Gain*[84] e *Chisquare*[85], que mostra uma relação entre duas variáveis categóricas, mantendo apenas as características mais importantes do conjunto de dados. Por meio desta combinação de algoritmos de redução de dimensionalidade, os autores as reduzem de 41 para 27 *features* e conduzem as experiências com criação de quatro conjuntos de dados diferentes, a partir do conjunto de dados KDDCUP99. Logo após, aplicam *undersampling*, *oversampling* e *resampling*, com e sem redução de *features*, avaliando-os. Segundo os autores, os melhores

[83] Método de otimização que se baseia em uma das teorias da evolução do universo foi proposto por Erol e Eksin (2006).
[84] Técnica de seleção de características que calcula a redução da entropia.
[85] Técnicas de seleção de características.

resultados foram atingidos com SVM, BBBC e a técnica de *oversampling* com a Acurácia de 97% e uma taxa de identificação de verdeiros positivos da classe de ataques tipo Probe[86] de 99,92% e para classe Normal de 99%, porem com baixas taxas de identificação de Falsos Positivos.

No trabalho de Gu et al. (2022) é realizada uma classificação de ataques de intrusão por meio de uma Rede Neural Convolutiva (CNN) e um algoritmo chamado *CenterBoderline_SMOTE* (CB_SMOTE), para resolver o problema de limitação de ataques. O método CB_SMOTE proposto pelos autores, baseia-se na ideia do BSMOTE que os autores consideram o que melhor desempenha a função para uma boa classificação. A diferença do CB_SMOTE para o BSMOTE segue apenas na escolha dos dados da região de bordas, que são ampliados a partir do limite do conjunto de dados. Os autores nomearam o modelo proposto de Sistema de Detecção de Intrusão e Expansão (DEIDS). Os dados utilizados no estudo foram dois conjuntos de dados de controle industrial o SWaT (Sistema de Tratamento de Água Segura) e o S7, que simula a mineração e refino reais do processo industrial contendo ciberataques. Os autores também, relataram que conseguiram alcançar uma taxa média de Acurácia de 98,29% com o conjunto S7 e 98,88% com SWaT e uma taxa de detecção de 95% e 97% de acertos respectivamente.

A proposta de Rani et al. (2022) propõem como solução para o desbalanceamento das classes dos conjuntos de dados NSL-KDD e UNSW-NB15 utilizar o classificador de Rede Neural aplicando penalidades por meio de pesos quando o algoritmo se comporta de modo tendencioso à classe majoritária. Os autores comparam a performance deste modelo com outros métodos já tradicionais de *oversampling* (SMOTE) e *undersampling* (*Clustering Centroids*), encontrados na literatura. Como resultado, concluíram que a técnica SMOTE produziu melhores resultados do que o método (*Clustering Centroids*), atingindo a pontuação de 85% e 91% e a medida de AUC de 94% e 97% para NSL-KDD e UNSW-NB15, respectivamente. Os autores recomendam que para trabalhos futuros poderão ser aplicadas técnicas de redução de dimensionalidade como uma melhoria do modelo proposto por eles.

O trabalho de Lee, Kim e Choi (2022) tem como proposta testar vários métodos de *oversampling* (BSMOTE, ROS, SMOTE e ADASYN) aplicados isoladamente e também combinados com OSS, utilizando o conjunto de dados CICIDS2017. Na seleção de características os autores removem 8 recursos que contém os mesmos valores ('Bwd PSH Flags', 'Bwd URG Flags', 'Fwd Byts/b Avg', 'Fwd Pkts/b Avg', 'Fwd Blk Rate Avg', 'Bwd Byts/b Avg', ' Bwd Pkts/b Avg', 'Bwd Blk Rate Avg'). Para a classificação utilizam CNN e avaliam o desempenho, com as métricas de Acurácia, Precisão, Sensibilidade e *F1-Score* com base nos quatro indicadores da matriz de confusão (TP, FP, FN, TN). Em conclusão, observaram que apesar dos métodos de *oversampling* apresentar uma melhora de 3,98% na classificação em média, o método

[86] Tipo de ataque onde um dispositivo envia muitos pacotes de autenticação diferentes, com a intensão de sobrecarregar o ponto de acesso vítima do ataque, de modo a forçar uma reinicialização do dispositivo.

híbrido de OSS+BSMOTE atingiu o melhor desempenho com 94,58% de Acurácia, 94,58% com *F1-Score* (micro), e 91,36% com *F1-Score* (macro).

Na proposta de Abedzadeh e Jacobs (2022) os autores dividem os dados em treino e teste e logo após realizam a limpeza dos dados e uma seleção de características híbrida com *Forward Selection Algorithm*[87](FSA) e PCA diminuindo de 79 para 8 *features* o conjunto de dados CICIDS2018. Na segunda etapa aplicam os algoritmos *Markov Chain Monte Carlos* (MCMC) e GAN como técnicas de *oversampling* para analisar a atuação dos modelos. Na terceira etapa, já com os dados balanceados, fazem uma classificação binária aplicando um conjunto de algoritmos (Linear Regression, Lasso, Ridge, ElasticNet, LR, Gaussian Naïve Bayes, Nearest Centroid, NN). Os resultados mostraram que o MCMC e GAN não apresentou melhoras em comparação com a aplicação nos dados originais. Os melhores desempenhos foram obtidos com LR e NN aplicado no conjunto de dados original. Também relatam que a LR gera um custo computacional menor, atingindo a Acurácia de 88%, Precisão de 88% e Sensibilidade de 99%.

Neste artigo os autores Surya e Selvam (2022) procuram classificar ataques de intrusão no conjunto de dados IoTID20[88] utilizando AM. Para seleção de características removeram 9 colunas com zero de entradas. Neste contexto, utilizam cinco algoritmos para classificação binária (*Gaussian Naive Bayes* (GNB), LR, DT, kNN e RF) e três para uma classificação multiclasses (GNB, DT, RF), aplicando SMOTE para balanceamento dos dados. Nos resultados da classificação binária atingem a Acurácia de 92.03% tanto para DT, como para RF, Precisão de 99%, Sensibilidade de 100% com GNB e *F1-Score* de 92% utilizando DT e RF. Já na classificação multiclasses os melhores resultados foram a Acurácia de 90,25% com DT e a métrica de Precisão e Sensibilidade alcançando 100% com os três classificadores para com a classe de *Scan*[89].

Na proposta de Abdulkareem et al. (2022) os autores propõem uma classificação com Ensemble Stack usando DT, NB, LR para detecção de intrusão. Para o problema de desbalanceamento do conjunto de dados Bot-IoT[90] aplicam a técnica de SMOTE testando o desempenho de 11 categorias de ataques, divididos em 80% para treino e 20% para testes, no qual apenas 3 das 43 características do conjunto são descartadas por serem especificas do dispositivo que coletou os dados. Para avaliar o desempenho utilizam as métricas de Acurácia, Precisão, Sensibilidade, Especificidade, Pontuação F1, Taxa de Falso Positivos, Taxa de Falso Negativos, Especificidade e o tempo de treino e teste, com e sem SMOTE. Nos resultados, o SMOTE-Stack superou outros classificadores de última geração em todas as 11 categorias de ataques, porém o tempo de treino e teste aumentou devido a aplicação de *oversampling*.

[87] Tipo de regressão passo a passo que começa com um modelo vazio e adiciona variáveis uma por uma fornecendo melhoria ao modelo.
[88] Conjunto de dados desenvolvido para detecção de ameaças no ecossistema da Internet.
[89] Técnica comum que os hackers usam para descobrir portas abertas ou pontos fracos em uma rede.
[90] Conjunto de dados criado através da concepção de um ambiente de rede realista no Cyber Range Lab da UNSW Canberra.

As métricas de Acurácia, F1, Precisão e Sensibilidade atingiram 100%, 100%, 99% e 99% respectivamente.

Com o objetivo de construir um modelo de detecção de intrusão baseado no balanceamento de dados e seleção de recursos, o trabalho de Sun et al. (2022) aplica uma metodologia híbrida utilizando as técnicas RUS (*undersampling*), seguida de BSMOTE (*oversampling*) no conjunto de dados CICIDS2017. Logo em seguida, dividem os dados em treino e teste (70-30) e realizam uma seleção de características baseada em *Information Gain*, separando os resultados em 6 grupos de dados contendo as seguintes quantidades de recursos (4, 22, 36, 62, 70 e 78) a partir do mais pontuado para o menos pontuado. Na etapa seguinte classificaram os dados com os algoritmos kNN, DT, RF. A partir dos resultados concluem que ao utilizarem o grupo com 62 características atingem a melhor taxa de Sensibilidade de 96% com DT, melhorando principalmente o reconhecimento de ataques de *Web Attack Brute Force* utilizando qualquer dos três algoritmos.

Tabela 4 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2022. Fonte: Elaborado pelo autor.

Artigo	***Dados***	**Classificador**	***Técnica***	**SC**	**Métricas**
Nagpal, Kaushal e Sharma (2022)	KDDCUP99	SVM	*undersampling*; *oversampling*; *resampling*	IG; *Chis-quare*	ACC; F1; P; S;
Gu et al. (2022)	SWaT; S7	CNN	CB_SMOTE (*oversampling*)	NI;	ACC; F1; P; S; FNR; FPR
Rani et al. (2022)	NSL-KDD; UNSW-NB15	NN	SMOTE; CC	NA;	ACC; AUC; F1; P;
Lee, Kim e Choi (2022)	CICIDS2017	CNN	BS;ROS; SMOTE; ADASYN;OSS	Próprio	ACC; F1; P; S
Abedzadeh e Jacobs (2022)	CICIDS2018	Linear Regression; Lasso; Ridge; ElasticNet; LR; NB; Nearest Centroid; NN	SMOTE	FSA; PCA	P; S
Surya e Selvam (2022)	IoTID20	Ensemble Stack (DT, NB, LR)	SMOTE	Próprio	ACC; F1; P; S

Tabela 4 – Continuação.

Abdulkareem et al. (2022)	Bot-IoT	Ensemble Stack (GNB; LR; DT; kNN; RF)	SMOTE	Próprio	ACC; E; F1; P; S; FNR; FPR
Sun et al. (2022)	CICIDS2017	kNN; DT; RF	BS; RUS	*IG*	S

Tabela 4 – Fim.

2.1.5 Trabalhos Correlatos - ano de 2023

O estudo de Xiang, Xu e Tang (2023) explora os algoritmos de *oversampling Random Over Sampling*[91] (ROS) e SMOTE em várias proporções de desbalanceamento que podem impactar na classificação utilizando CNN, *Vision Transformer*[92] (VT), *XGBoost* e *CatBoost* no conjunto de dados CICIDS-2017. Os resultados apontaram que o classificador VT juntamente com SMOTE obteve a melhor performance atingindo com Sensibilidade, Precisão, F2-Score e AUC de 99,10%, 97,23%, 97,52% e 99,42% respectivamente. Concluíram também que o uso da técnica ROS, quando aplicado em uma taxa de desbalanceamento de 99:1 aumenta a AUC dos algoritmos de classificação utilizados nos experimentos. Complementam ainda que à medida que o desequilíbrio aumenta, a Acurácia dos quatro classificadores diminui, resultando em uma diminuição na métrica *F2-Score*[93] o que implica que o aumento das amostras melhora a capacidade de aprendizado do modelo.

Para os autores Verma et al. (2023) a presença de dados com classes desbalanceadas representam um grande desafio para sistemas de detecção de intrusão, dentro deste contexto, os autores propõem uma abordagem de classificação binária no conjunto de dados UNSW-NB15, selecionando as características com PCA e aplicando *oversampling* baseado na utilização da técnica *Gametic Hereditical*(GH), algoritmo inspirado em princípios da biologia genética para equilibrar classes em aplicações de segurança cibernética em sistemas IoT. Para classificação utilizam os algoritmos RF, kNN, LR, SVM, DT e validam a eficácia do modelo com as métricas de Acurácia, Sensibilidade e *F1-Score* atingindo o desempenho de 97,70%, 99,10% e 98,10% respectivamente com RF. Os autores afirmam que o modelo proposto garante a propagação de amostras sintéticas que se encontram dentro dos limites da classe minoritária, superando os métodos convencionais nos valores das métricas em 6% para Acurácia, 8% para taxa de Sensibilidade e 7% para *F1-Score*.

Os autores de Dasari et al. (2023) sugerem um modelo de Ensemble com uso de

[91] Método de sobreamostragem aleatória usado para equilibrar a distribuição de classes multiplicando aleatoriamente as amostras de rótulos de classes minoritárias.

[92] Mecanismo para analisar imagens, dividindo-as em fragmentos menores e aproveitando mecanismos de autoatenção conforme Dosovitskiy et al. (2020).

[93] Métrica utilizada para diminuir a importância da Precisão e aumentar a importância da Sensibilidade.

Machine Learning Analytics[94] utilizando RF, XGB e comparam com classificadores únicos NB, DT e kNN. Para preparação do modelo realizam uma seleção de características utilizando Relevância Mínima de Redundância Máxima (MRMR) e para o balanceamento dos dados no conjunto de dados KDDCUP99 aplicam SMOTE, após converter os dados para uma classificação binária. Para validação do modelo utilizam a métricas de Acurácia e *F1-Score* conseguindo um desempenho de 98,09% e 97,89% respectivamente. Também afirmam que o modelo proposto quando comparado com outras abordagens de Ensemble com *Extra Trees* (ET) possui maior rapidez de classificação, proporcionando um melhor desempenho.

Visando o problema de dados de intrusão desbalanceados, o trabalho de Wang et al. (2023) tem como proposta um método de *oversampling* chamado RB-SMOTE, que utiliza BSMOTE como base aplicando uma camada de refinamento por meio de Rede Neural Convolucional (CNN) de oito camadas para classificação. Os autores utilizam o conjunto de dados NSL-KDD, no qual verificam as dimensões dos dados para converte-los em imagens descobrindo semelhanças do mesmo tipo de dado e assim aplicam penalizações em regiões que podem melhorar a representação dos recursos. Comparando os resultados com outros métodos como RF + BSMOTE que atingiu 93,53% e SMOTEENN + CNN com 99,35% de Acurácia, mostraram que o método proposto por eles apresentou um bom desempenho atingindo 99,84% com mesma métrica.

O trabalho de Madwanna et al. (2023) tem como proposta uma classificação com dois modelos de *Deep Learning* para detecção de intrusão. O primeiro baseia-se na utilização de CNN juntamente com RNN aplicando *LuNet*[95] para classificar dados de tráfego de entrada extraindo dados temporais. No segundo modelo utilizam o método *Temporal Neighborhood Coding*[96] (TNC) com CNN para calcular características temporais de baixo nível e logo após, esses recursos alimentam uma RNN para obter os recursos temporais de alto nível realizando uma classificação unificada de captura das informações. Os conjuntos de dados utilizados são UNSW-NB15 e NSL-KDD aplicando SMOTE para o balancear as classes. A validação dos modelos propostos é realizada por meio de uma comparação com outros modelos selecionados na literatura. Os resultados mostraram que o primeiro modelo proposto alcançou 82,19% e 98,87% de Acurácia na classificação de ataques nos conjuntos UNSW-NB15 e NSL-KDD, respectivamente. No segundo modelo alcançou 98,87% de Acurácia usando o conjunto de dados NSL-KDD. A melhor Acurácia nos modelos utilizados para comparação foi de 82,20% com aplicação de SVM e *AdaBoost* para o balanceamento dos dados.

Com o intuito de classificar ataques de intrusão de forma binária no conjunto de dados UNSW-NB15, os autores Mouiti et al. (2023) utilizam ADASYN para lidar com o desequilíbrio das classes e aplicam os classificadores LR, RF, DT e *Multilayer Perceptron* (MLP) com e

[94] Processo analítico de avaliação de dados e descoberta de insights para tomada de decisões.
[95] Múltiplos níveis de convolução combinados e sub-redes recorrentes.
[96] Codificação de Vizinhança Temporal, aproveita a suavidade local do processo gerador de um sinal para definir vizinhanças no tempo com propriedades estacionárias.

sem ajustes de hiperparâmetros para uma comparação. Dentro deste cenário, afirmam que o não ajuste dos hirperparâmetros e a não aplicação da validação cruzada implica a um modelo impreciso com um elevado viés (*underfitting*) ou alta variância (*overfitting*) e ainda provoca um aumento no tempo de treinamento. Como resultado, verificam que o desempenho do modelo proposto melhoram a classificação, principalmente para o uso de RF, superando 98% para com as métricas de Acurácia, Precisão, Sensibilidade, F1 e AUC; Logo após, o DT, variando entre 94% para Sensibilidade e 96% para Precisão; LR, entre 86% para o Sensibilidade e 96% para o AUC; e MLP, atingindo entre 89% e 97% para as taxas de Sensibilidade e a AUC, respectivamente.

O artigo de Gou, Zhang e Zhang (2023), propõem uma classificação de ataques utilizando *Ensemble Bagging* com os algoritmos RF, ET, XGB e LightGBM nos conjuntos de dados CICIDS-2017 e Car-Hacking[97]. Devido ao desbalanceamento dos conjuntos usam as técnicas de SMOTE+ RUS e ROS+RUS para solucionar o problema de desequilíbrio das classes e também aplicam PCA reduzindo os 78 atributos do CICIDS2017 para 27. No dataset Car-Hacking evitam a seleção de recursos por possuir apenas 8 recursos. Os resultados mostraram que na classificação do conjunto de dados CICIDS2017 usando a combinação híbrida de SMOTE e RUS o modelo atinge 96,65% de F1 e no conjunto Car-Hacking 99,99% com a mesma métrica. Os autores relatam que o modelo proposto por eles pode proporcionar uma melhoria de 6,8% na pontuação F1 comparado com os dados originais. Outras métricas como Sensibilidade, Acurácia e Precisão também foram aplicadas na classificação de ambos conjuntos de dados e tendo como resultados 99,99% quando aplicado no conjunto Car-Hacking e 96,65%, 94,90% e 96,65% respectivamente no CICIDS2017.

A proposta dos autores Jin et al. (2023) visa a classificação com os algoritmos CNN e LSTM para ataques de intrusão. Para resolver o desbalanceamento do conjunto de dados utilizado no trabalho, que foi originado do tráfego da rede de um sistema de controle de armazenamento de água da província de Hebei[98], aplicam SMOTE. Na validação dos resultados os autores comparam o modelo com outros que aplicam RF, DT, SVM e LR. Com isso, percebem que após a aplicação de SMOTE a detecção de ataques (classe minoritária) obteve uma melhora significativa. Na aplicação de forma multiclasses, a Acurácia do modelo proposto atingiu 97,04%, a Precisão 97,17%, F1 97,03% e a taxa de falsos negativos foi de 2,95%. Já na classificação binária a Acurácia foi de 99,30% e a taxa de falsos negativos de 0,7%. Em comparação com os modelos propostos, DT obteve a segunda melhor Acurácia com 95,43%, F1 com 94,39% e taxa de falsos negativos de 4,57%. O classificador RF atingiu segunda melhor Precisão com 94,98%.

No trabalho de Rahma et al. (2023) os autores afirmam que o desbalanceamento das classes tem sido um problema decisivo em sistemas de detecção de intrusão por serem

[97] Conjunto de dados fornecido e coletado durante o Car Hacking: Attack & Defense Challenge em 2020.
[98] Província da República Popular da China.

tendenciosos para a classe majoritária. Desta forma, exploram em seu trabalho várias técnicas de *oversampling* e *undersampling* (*Random Over Sampling*, SMOTE, ADASYN, *Random Under Sampling*, AllKNN[99], TomekLinks, SMOTEENN e SMOTETomek) para melhorarem a detecção de ataques usando a técnica de aprendizado profundo combinando Rede Neural Convolucional e LSTM Bidirecional (BiLSTM). Os resultados apontaram que com os dados brutos a pontuação de Acurácia obteve o melhor resultado atingindo 96,50%. Mas utilizando o método *Random Over Sampling* conseguem a maior pontuação F1 com 58,95%. Em conclusão relatam que as técnicas de *undersampling* e *oversampling* podem melhorar o desempenho de IDS de forma singular, mas ainda precisam ser aprimorados. O conjunto de dados utilizado nos experimentos foi UNSW-NB15.

O estudo de Gavrylenko, Vladislav e Khatsko (2023) visa melhorar a qualidade da classificação dos dados desbalanceados no conjunto de dados UNSW-NB15. Diante de várias abordagens de balanceamento de dados como SMOTEENN, SVM-SMOTE, BSMOTE, ADASYN, SMOTE, KMeansSMOTE[100] testadas pelos autores, o SMOTEENN foi a que atingiu a melhor performance, destacando-se com uma pontuação F1 de 94% juntamente com os classificadores Gradient Boosting ou Random Forest. Segundo os autores, em comparação com a classificação aplicada nos dados originais usando a mesma métrica atingiu apenas 61%.

Os autores Bagui et al. (2023) citam em seu trabalho que apesar do aumento dos ataques nas redes de computadores crescerem frequentemente, a porcentagem desses ataques no tráfego real da rede é significativamente menor, formando conjuntos de dados altamente desbalanceados. Em solução utilizam de *undersampling* e *oversampling* nos dados e vice-versa, com as técnicas BSMOTE e SVM-SMOTE e RUS no conjunto de dados UNSW-NB15 para observarem a melhor porcentagem para ser aplicada nos dados. Devido a Acurácia ser uma métrica tendenciosa para classes com mais dados, os autores utilizam a Precisão, Sensibilidade, F1 para avaliar os modelos. Após uma seleção de características com *Information Gain*, a classificação se realiza com algoritmo kNN, testando os valores de k = 3, 5 e 10. Em conclusão notam que usando uma sobreamostragem de 10% tanto com BSMOTE ou SVM-SMOTE seguidos da subamostragem com RUS alcançam os melhores resultados e também relatam que para classes mais raras, o uso de kNN com k mais alto levou a um aumento nas porcentagens dos resultados de *resampling*. Com o método de *oversampling* SVM-SMOTE seguido do método de *undersampling* RUS e classificando kNN com k = 10 atingiram a Precisão 91,40%, Sensibilidade de 95,70%, F1 de 93,50% e Macro precisão de 95,70% e com BSMOTE seguido de RUS alcançaram 96,80%, 94,40%, 95,60%, 98,40% respectivamente, classificando ataques de Backdoors.[101].

Na proposta de Mohammadi e Babagoli (2023), o objetivo é a detecção de ataques de

[99] Técnica de undersampling derivação de Edited Nearest Neighbours proposto por Tomek (1976).
[100] Técnica de oversampling baseado em agrupamento k-means juntamente com a técnica SMOTE.
[101] Tipo de ataque onde um mecanismo de segurança do sistema é contornado de forma indetectável para acessar um computador.

Brute force, *Infiltration*, *Botnet* e *Portscan* baseado em modelagem de regressão não linear utilizando o conjunto de dados CICIDS2017. No pré-processamento, o algoritmo SMOTETomek é utilizado para o balanceamento dos dados e na seleção dos recursos os algoritmos meta-heurísticos *Gray Wolf*[102] e *Hunger Games Search* (HGS)[103], que removeu mais de 80% dos recursos do conjunto de dados classificando-os com irrelevante. Em sequência realizam uma classificação binária com Regressão Logística. Os resultados apontaram para uma melhor atuação do algoritmo híbrido HGS em comparação aos resultados nos trabalhos relacionados, obtendo uma taxa média de Acurácia de 99,17% contra a média de 94,61% dos trabalhos relacionados.

Tabela 5 – Artigos obtidos por meio da Revisão Sistemática da Literatura publicados no ano de 2023. Fonte: Elaborado pelo autor.

Artigo	***Dados***	**Classificador**	***Técnica***	**SC**	**Métricas**
Xiang, Xu e Tang (2023)	CICIDS2017	CNN; VT; XGB; *CatBoost*	ROS; SMOTE;	NA	AUC; F2; P; S
Verma et al. (2023)	UNSW-NB15	RF; kNN; LR; SVM; DT	GH	PCA	ACC; F1; S
Dasari et al. (2023)	KDD99	RF, XGB	ROS; SMOTE	MRMR	ACC; F1
Wang et al. (2023)	NSL-KDD	RF; CNN;	RB-SMOTE; BS; SMOTEENN	Correlação	ACC
Madwanna et al. (2023)	UNSW-NB15; NSL-KDD	RNN; CNN; SVM; AB	BS; RBSMOTE	NA	ACC
Mouiti et al. (2023)	UNSW-NB15	LR; RF; DT; MLP	ADASYN	NA	ACC; AUC; F1; P; S
Gou, Zhang e Zhang (2023)	CICIDS2017	RF; ET; XGB; LightGBM	SMOTE; ROS; RUS	PCA	ACC; F1; P; S
Jin et al. (2023)	Próprio (Hebei)	CNN; LSTM	SMOTE	NA	ACC; F1; FNR; P; S
Rahma et al. (2023)	UNSW-NB15	CNN; LSTM	ROS,SMOTE; ADASYN;RUS; AllKNN;Tomek; SMOTEENN; SMOTETomek	NA	ACC; F1; S

[102] Técnica meta-heurística estocástica de inteligência de enxame, desenvolvida em 2014, cuja ideia é baseada no modelo de caça em bando dos lobos cinzentos.
[103] Algoritmo projetado conforme as atividades motivadas pela fome e a escolha comportamental dos animais.

Tabela 5 – Continuação.

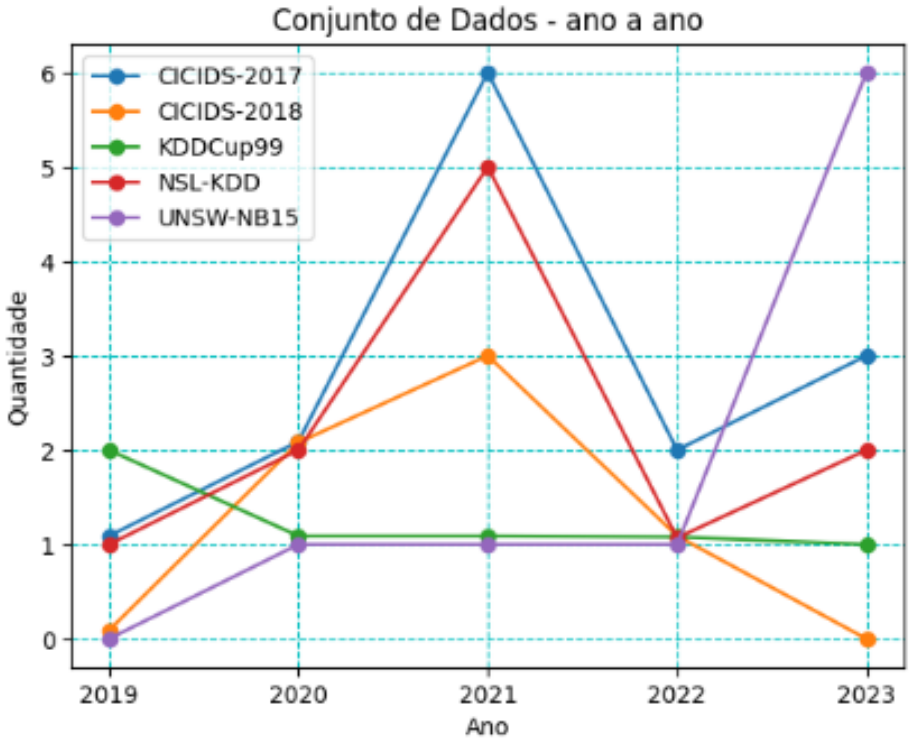
Gavrylenko, Vladislav e Khatsko (2023)	UNSW-NB15	Ensemble; GB; RF	SVMSMOTE;BS; ADASYN; SMOTE; KMeansSMOTE; SMOTEENN	NA	F1;
Bagui et al. (2023)	UNSW-NB15	kNN	SVM-SMOTE; BS; RUS	IG	F1; P; S
Mohammadi e Babagoli (2023)	CICIDS2017	LR	SMOTETomek	*Gray Wolf*; HGS	ACC

Tabela 5 – Fim.

2.2 Inclinações Extraídas dos Trabalhos

Uma análise dos conjuntos de dados, classificadores, métodos de seleção de características, técnicas de balanceamento de dados aplicadas e métricas de avaliações, permitem uma visualização das inclinações mais utilizadas empregadas na presente pesquisa. A Figura 4 apresenta os cinco conjuntos de dados mais utilizados por quantidade e ano a ano, nos trabalhos selecionados nesta pesquisa:

Figura 4 – Conjuntos de dados extraídos nos trabalhos selecionados - visualização ano a ano.



Fonte: Elaborado pelo autor.

Observa-se, que o conjunto de dados CICIDS2017 aparece presente nos trabalhos dos anos de 2019 até 2023, tendo uma expressiva representação nos artigos selecionados, somando um total de 14 repetições. Já CICIDS2018 apesar de contabilizar 6 trabalhos não aparece nos anos de 2019 e 2023.

Nota-se que o conjunto de dados UNSW-NB15, obteve um crescimento expressivo em 2023, porém em 2019 não foi encontrado em nenhum trabalho selecionado, contabilizando um total de 9 trabalhos.

Também representativos no gráfico apresentam-se o NSL-KDD com 11 aparições e o KDDCup99 com 6 ambos representados de 2019 até 2023 nos *papers* da pesquisa, mas possuem dados obsoletos se comparados aos demais, por se tratar de conjuntos de dados com mais de duas décadas, não contendo ataques mais recentes como *Ransomware*[104] e com muita publicações existentes. No entanto, cabe ressaltar que embora eles sejam antigos, são muito usados ainda, pois permitem comparar desempenhos atuais com desempenho passados.

Neste sentido, pode-se dizer que quanto mais novos os dados de ataques nos conjuntos de dados, mais viáveis para construção de um modelo de classificação eficaz, por conter tanto ataques e vulnerabilidades do cenário atual, quanto também tráfego normal de redes, compatíveis com os dispositivos atualizados.

Segundo Khraisat et al. (2019), que pesquisaram a eficiência de vários conjuntos de dados de IDS usados para as técnicas de AM, relatam que diversas abordagens foram utilizadas para detecção de ataques de dia zero, mas estas técnicas apresentam problemas para gerar e atualizar estes dados, fazendo produzir alarmes falsos ou apresentarem uma baixa Acurácia em suas classificações. Complementam ainda, que como as atividades normais estão modificando frequentemente perdem sua precisão com o tempo, o que torna necessário que AM utilize conjunto de dados novos e mais abrangentes.

É importante ressaltar que em alguns trabalhos selecionados na pesquisa, os autores utilizaram mais de um conjunto dados para seus experimentos, o que também foi contabilizado na contagem apresentada na Figura 4.

A Figura 5 ilustra os classificadores utilizados por ano extraídos dos trabalhos selecionados na pesquisa. Trata-se dos cinco classificadores com mais utilização ao longo dos anos, ficando visível um grande uso de *Random Forest* (RF), que aparece em enorme quantidade em 2021 contabilizando 11 vezes e totalizando 26 de 2019 a 2023.

Percebe-se também, uma grande utilização de Rede Neurais (NN) principalmente no ano de 2021 somando 9 trabalhos. Mas não se pode deixar de considerar o uso de Máquinas de Vetores de Suporte (SVM), embora tenha apresentado queda em 2022 que aprece em apenas 1 trabalho, porém somando 13 representações em todo o período de 2019 a 2023, igualmente

[104] Tipo de código malicioso que torna inacessíveis os dados armazenados em um equipamento, geralmente usando criptografia.

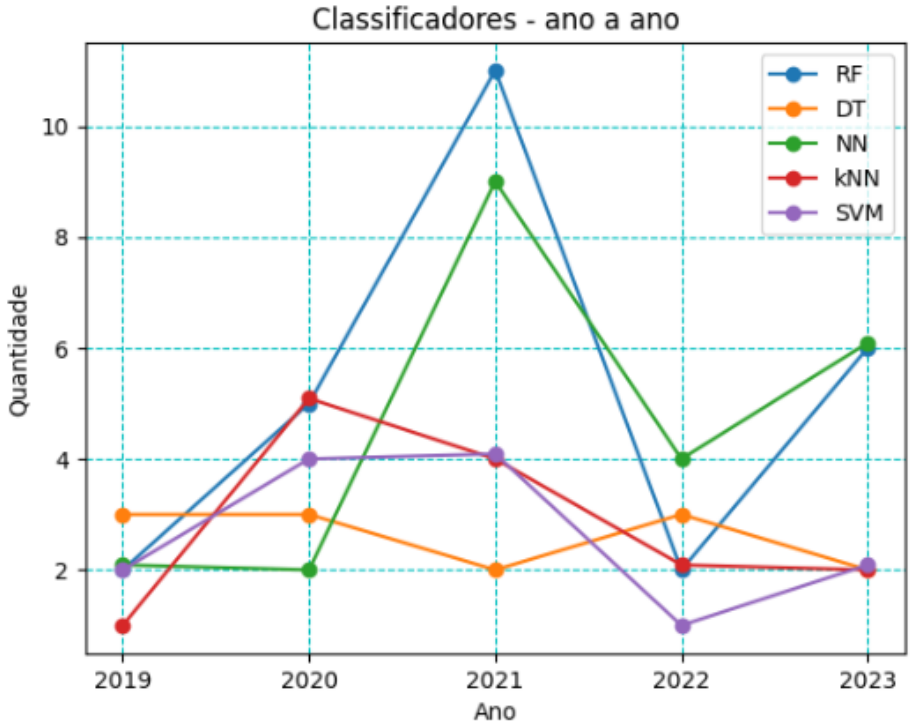
Decison Tree (DT) também com 13 representações aparecendo em todos os anos da pesquisa. E por fim, *k-Nearest Neighbor* (kNN) que aprece em 5 trabalhos no ano de 2020 e 14 no período de 2019 a 2023. Esses dados podem ajudar a entender os classificadores escolhidos nos anos de 2019 a 2023 para trabalhar juntamente com técnicas de balanceamento dos dados em conjunto de dados de intrusão.

É importante evidenciar que em alguns trabalhos selecionados na pesquisa, os autores utilizaram mais de um classificador, o que também foi contabilizado na contagem apresentada na Figura 5.

Outro fator observado nos trabalhos, é que alguns autores associaram classificadores, aplicando-os sequencialmente para conseguir atingir uma melhor performance, RF é um exemplo disso sendo citado em vários trabalhos que foram selecionados.

No artigo de Sun et al. (2020), a proposta pretende aplicar o algoritmo de classificação individualmente para cada classe detectada pelo ataque do IDS[105], tanto nos conjuntos de dados UNSW-NB15 como no CICIDS-2018 e logo após aplicam a média dos resultados de cada métrica por classe de ataque antes e depois da aplicação da técnica de SMOTE para uma comparação dos resultados.

Figura 5 – Classificadores utilizados nos artigos correlatos.



Fonte: Elaborado pelo autor.

A Figura 6 apresenta as 5 técnicas mais aplicadas nos trabalhos correlatos, pode-se observar que a técnica SMOTE é utilizada na maioria dos trabalhos correlatos para tratar o

[105] Sistema de detecção de intrusão.

desbalanceamento das classes nos conjuntos de dados, totalizando 26 trabalhos. Com isto, nota-se que mais da metade dos trabalhos aplicaram a técnica de SMOTE para equilibrar as classes ou para validar o modelo proposto, fazendo comparações.

Também, aparece a derivação do método SMOTE (BSMOTE), que faz-se notar que não foi usado apenas no ano de 2021 em trabalhos escolhidos na pesquisa, mas somam 9 trabalhos que fizeram o uso desta técnica.

A técnica SMOTE surgiu no ano de 2002 proposto por Chawla et al. (2002), sendo uma das mais conhecidas na aplicação de conjunto de dados desbalanceados para ser usado na classificação de modelos de AM. O SMOTE usa o algoritmo kNN para selecionar as k amostras da classe minoritária mais próximas de uma amostra também da classe minoritária, e amostras sintéticas são criadas e posicionadas no segmento entre elas.

Já BSMOTE, surgiu em 2005 quando Han, Wang e Mao (2005), baseando-se na técnica SMOTE apresentam dois novos métodos de *oversampling*: BSMOTE1 e BSMOTE2, que usa apenas os exemplos minoritários próximos à fronteira entre as classes para serem superamostrados.

O trabalho de Pawlicki et al. (2020), testou com vários algoritmos de balanceamento de dados aplicado no conjunto de dados CICIDS2017 e BSMOTE foi o que melhor se ajustou em comparação com outros métodos de *oversampling* do modelo proposto por eles. Também relatam em seu trabalho, que muitas abordagens derivadas de SMOTE vem sendo encontrada na literatura.

A técnica ADASYN foi aplicada em 8 trabalhos selecionados para a pesquisa, este algoritmo foi proposto por He et al. (2008) e tem como base a geração dos dados da classe minoritária de forma adaptativa conforme as distribuições das classes. Este modelo procura amostras que são difíceis de aprender, fazendo uma sobreamostragem para equilibrar as classes dos conjuntos de dados.

No trabalho de Chen, Zhou e Yu (2021), os autores aplicaram a técnica de ADASYN e classificaram o modelo proposto com *Random Forest*, combinação que comparada com outras técnicas de (*oversampling*) aplicadas no conjunto de dados CICIDS2017, foi a que atingiu melhores resultados.

A técnica RUS também contabilizou 8 trabalhos na pesquisa, ficando com a mesma quantidade ano a ano que ADASYN.

A técnica *Random Under-Sampling* (RUS) é fundamentada da seguinte forma: cada elemento da classe onde será feita a predição, possui uma chance igual de ser escolhida para a amostra. Determina-se a quantidade de dados da amostra e com um método de sorteio, a cada tipo de amostra dos dados é atribuído um número, em seguida os números são selecionados aleatoriamente. Esta abordagem de *undersampling* é comumente usada para lidar com conjuntos de dados desbalanceados em problemas de classificação em AM.

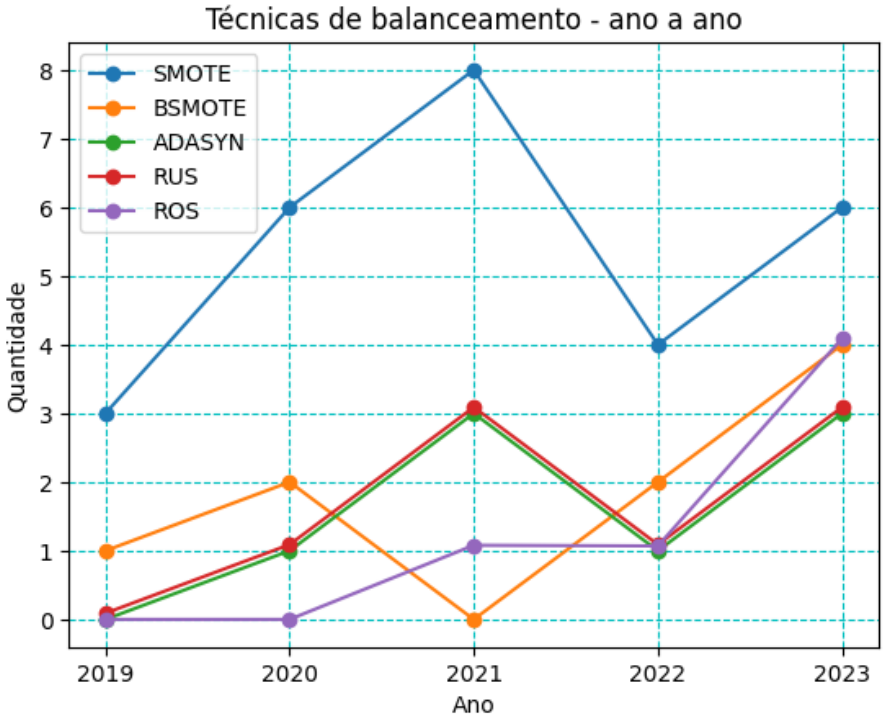
Em Bagui et al. (2023), os autores aplicaram várias técnicas de *undersampling* e *oversampling* no conjunto de dados UNSW-NB15, perceberam que os melhores resultados foram atingidos na aplicação com RUS após a aplicação de BSMOTE.

Outra técnica entre as 5 mais aplicadas nos trabalhos correlatos foi *Random Oversampling* (ROS), que utiliza a classe minoritária para balancear o conjunto de dados, duplicando aleatoriamente os dados com ou sem reposição as observações de dados da classe minoritária.

No trabalho de Xiang, Xu e Tang (2023) os autores perceberam que ao aplicar a técnica ROS no conjunto de dados CICIDS2017 em uma taxa de desbalanceamento de 99:1 aumenta a AUC dos algoritmos de classificação utilizados em seus experimentos.

Faz-se importante ressaltar que foram aplicadas pelos autores uma ou mais técnicas a cada trabalho, sendo todas contabilizadas na contagem da Figura 6.

Figura 6 – Técnicas de balanceamento utilizadas nos trabalhos correlatos.



Fonte: Elaborado pelo autor.

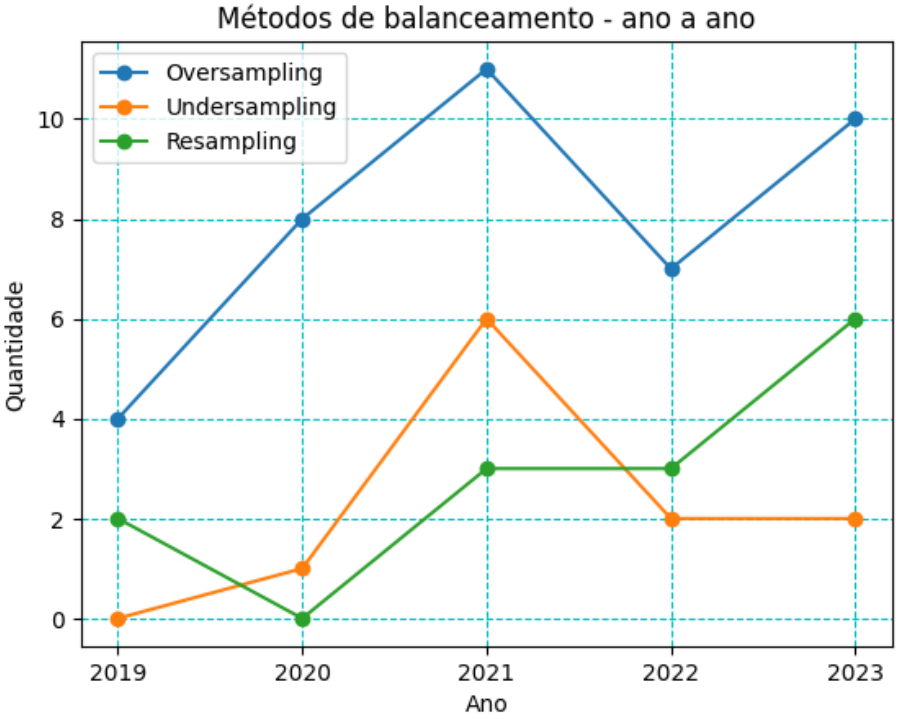
Na Figura 7, percebe-se que o método de *oversampling* aparece com mais frequência nos trabalhos correlatos para resolver o problema de desbalanceamento dos dados nos conjuntos de dados gerados por IDS. Porém, nota-se que o método de *resampling* vêm crescendo nos últimos anos, sendo que no ano de 2023 aparece em 6 trabalhos na pesquisa. Notou-se também, que a maioria dos autores dos trabalhos correlatos aplicaram mais de um método de balanceamento de classes, para fazer uma comparação dos resultados e verificar o melhor resultado por meio das métricas de avaliação.

Na contagem do gráfico da Figura 7 contabilizou-se todos os métodos descritos pelos autores dos trabalhos correlatos.

Conforme Chen, Zhou e Yu (2021), os conjuntos de dados gerados por IDS apresentam um grande e grave desequilíbrio que resultam em um baixo desempenho de classificação em comportamentos de ataque de pequeno tamanho de amostra, dificultando a detecção ataques de rede com precisão e eficiência.

Percebe-se que devido à pequena quantidade nas classes de ataques, os autores aplicam o método de *oversampling* com mais frequência, porém os métodos de *undersampling* e de *resampling* em alguns casos podem trazer bons resultados.

Figura 7 – Métodos de balanceamento utilizadas nos trabalhos correlatos.



Fonte: Elaborado pelo autor.

2.2.1 Métricas de Desempenho Utilizadas nos Trabalhos Correlatos

As diversas métricas de classificação têm como alvo comum medir a performance do modelo proposto, mas realizam a tarefa de formas diferentes. Segundo Alfrhan, Alhusain e Khan (2020), a métrica *F1-Score* é a que produz um resultado mais correto ao trabalhar com conjuntos de dados que possuem dados desbalanceados, por possuir um baixo Valor Preditivo Positivo (PPV) das classes de ataques, que contêm pequenas porcentagens de dados, apesar de ser afetado pelos valores decrescentes da Sensibilidade e Precisão. Entretanto, citam que Acurácia também pode ser conveniente para conjunto de dados com classes desbalanceadas, por conter um leve efeito sobre a classe minoritária. Pode-se verificar algumas destas métricas nas equações:

A Acurácia pode ser obtida pela soma de Verdadeiros Positivos (VP) e Verdadeiros Negativos (VN) sobre a soma de Verdadeiros Positivos (VP), Verdadeiros Negativos (VN), Falsos Positivos (FP) e Falsos Negativos (FN).

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{1}$$

A Precisão é a métrica que avalia a quantidade de Verdadeiros Positivos(VP) sobre a soma de Verdadeiros Positivos(VP) e Falsos Positivos(FP).

$$Precisão = \frac{VP}{VP + FP} \tag{2}$$

A Sensibilidade ou *Recall* é a métrica que avalia a capacidade de detectar os resultados classificados como positivos e pode ser definida como Verdadeiros Positivos(VP) sobre a soma de Verdadeiros Positivos(VP) e Falsos Negativos(FN).

$$Sensibilidade = \frac{VP}{VP + FN} \tag{3}$$

A Especificidade avalia a capacidade do método detectar resultados negativos e pode ser definida como: Verdadeiros Negativos(VN) sobre a soma de Verdadeiros Negativos(VN) com Falsos Positivos(FP).

$$Especificidade = \frac{VN}{VN + FP} \tag{4}$$

A métrica *F1-Score* ou *F–Measure* é uma medida harmônica entre a Precisão e a Sensibilidade.

$$F1 - Score = 2 * \frac{(precision * sensibilidade)}{(precision + sensibilidade)} \tag{5}$$

Outra métrica utilizada nos trabalhos correlatos para avaliação dos modelos de classificação de intrusão usadas por Rani et al. (2022) e por Alfrhan, Alhusain e Khan (2020) é a Curva ROC (*Receiver Operating Characteristic Curve*), que se resume em um gráfico que permite medir um classificador binário levando em conta os Verdadeiros Positivos (VP) e Falsos Positivos (FP), podendo ser obtida por meio do cruzamento gráfico da Precisão com o *Recall* (Sensibilidade) em diferentes limiares de decisão, estimando os valores da área sob esta curva. O que também Chawla et al. (2002), confirma que o mais adequado como medida para dados desbalanceados seria usar a Curva ROC para resumir o desempenho do classificador em modelos de AM.

No trabalho Phetlasy et al. (2019), os autores fizeram uso da classificação por Matriz de Confusão como forma de avaliar a eficácia dos classificadores utilizados. Em suas pesquisas, conseguiram aplicando classificadores de forma sequenciais, reduzir a taxa de Falsos Negativos (FN) considerando a taxa de Falsos Positivos (PF) de forma aceitável.

Sokolova, Japkowicz e Szpakowicz (2006), argumentam que para relacionar os erros e acertos do modelo de classificação e não ficar apenas com métricas comumente usadas na

literatura, como a Acurácia, Precisão e *F1-Score* deve-se usar a Matriz de Confusão. É possível observar um exemplo na Tabela 5:

Tabela 6 – Arquitetura de uma Matriz de Confusão. Fonte: Adaptada de Phetlasy et al. (2019)

Verdade / Previsto	**como Positivos**	**como Negativos**
Positivo	*Verdadeiros Positivos* (TP)	*Falsos Negativos* (FN)
Negativo	*Falsos Positivos* (FP)	*Verdadeiros Negativos* (TN)

Quando dados de intrusão são classificados, os conjuntos de dados originados do tráfego de rede podem ser ataques Verdadeiros Positivos (TP) ou um tráfego benigno (normal) Verdadeiros Negativos (TN). Desta forma, considerando que dentro deste fluxo de ataques e não ataques (normal) no processo de classificação encontraremos:

- TP: tráfego classificado como ataque e que de fato era um ataque;
- FN: tráfego classificado como benigno (normal) mas que era originalmente um ataque;
- FP: tráfego que era benigno originalmente, mas foi detectado como ataque, e
- TN: tráfego benigno (normal) que foi classificado como benigno (normal).

No trabalho de Al-Imran e Ripon (2021), os autores usaram o Coeficiente de Correlação de Matthews (MCC). Essa métrica baseia-se na correlação entre os valores verdadeiros (observados) e os previstos pelo modelo de classificação. Esse valor pode variar de -1 (predição errada) a 1 (predição perfeita) e 0 (predição aleatória média). Pode-se verificar o cálculo na equação:

$$MCC = \frac{(VP * VN) - (FP * FN)}{\sqrt{(VP + FP)(VP + FN)(VN + FP)(VN + FN)}} \tag{6}$$

2.2.2 Métricas de Desempenho Extraídas dos Trabalhos

A Tabela 7, traz o desempenho das melhores métricas apontadas pelos autores conforme os valores que constam nas publicações dos trabalhos correlatos, sendo organizadas da seguinte forma:

- Artigo - O trabalho que se refere os dados;
- Dados - O conjunto de dados usado para resultado;
- Classificador - O classificador usado resultado;
- Técnica - A técnica e método de balanceamento dos dados (*undersampling*, *oversampling* ou *resampling*), aplicada no trabalho;

- Métricas - As métricas utilizadas nos resultados dos trabalhos;
- Valores - Os valores alcançados pelas métricas em porcentagem.

Tabela 7 – Compilação dos resultados publicados para os artigos citados. Fonte: Elaborado pelo autor.

Artigo	**Dados**	**Classificador**	**Técnica**	**Métricas**	**Valores**
Pang et al. (2019)	Própria	SVM	BSMOTE (*oversampling*)	AUC	97,19
				F1	89,83
				G-Mean	95,92
Jimoh, Ismaila e Olalere (2019)	CICIDS2017	J48	SMOTE (*oversampling*)	ACC	99,95
				AUC	99,74
				F1	99,71
				P	99,75
				S	99,62
Lu et al. (2019)	KDDCup99	RF	SMOTEENN (*resampling*)	F1	99,99
				P	99,99
				S	99,99
Phetlasy et al. (2019)	NSL-KDD	J48	SMOTE (*oversampling*)	ACC	89,95
				E	97,02
				S	84,60
Pristyanto e Dahlan (2019)	UCI MLR	ANN	OSS+SMOTE (*resampling*)	ACC	96,26
				G-Mean	96,26
Tan et al. (2019)	KDDCup99	RF	SMOTE (*oversampling*)	ACC	92,57
				AUC	92,58
Prayogo e Karimah (2020)	UCI MLR	kNN	SMOTE (*oversampling*)	ACC	97,47
				F1	97,50
				P	97,50
				S	97,50
Alfrhan, Alhusain e Khan (2020)	CICIDS2017	DF	SMOTE (*oversampling*)	AUC	100
				F1	100
				S	100
Sun et al. (2020)	CICIDS2018	VSTG-MTL	SMOTE (*oversampling*)	F1	77,58
				FNR	9,95
				FPR	56,52
Pawlicki et al. (2020)	CICIDS2017	RF	RUS (*undersampling*)	ACC	98,72
				F1	74
				P	69
				S	96

Tabela 7 – Continuação.

Pan e Xie (2020)	KDDCUP99	XGB (R2L)	ADASYN (*oversampling*)	F1 FPR	90,10 17,30
Zhang, Zhang e Li (2020)	NSL-KDD	kNN+C4.5 (normal)	BSMOTE (*oversampling*)	ACC F1 P S	99,89 99,90 99,93 99,87
Engly, Larsen e Meng (2020)	NSL-KDD	NN+GB	SMOTE (*oversampling*)	ACC F1 P S	81 68 75 70
Gonzalez-Cuautle et al. (2020)	ISCX-Bot2014	kNN	SMOTE (*oversampling*)	ACC AUC F1 P S	98,72 96,76 96,76 98,58 96,77
Karatas, Demir e Sahingoz (2020)	CICIDS2018	AB	SMOTE (*oversampling*)	ACC F1 P S	99,60 99,60 99,60 99,61
Mohseni e Tanha (2021)	CICIDS2017	RF	RUSBoost (*undersampling*)	ACC G-Mean	99,70 99
Anh et al. (2021)	CICIDS2017	RF	Própria (*oversampling*)	ACC F1 S	99,96 99 99
Szczepanski et al. (2021)	IoT-23	RNN	SMOTEENN (*resampling*)	ACC P S	97 100 97
Wang e Septian (2021)	CICIDS2017	LSTM	SMOTE (*oversampling*)	ACC F1 P S	98,90 98,90 98,90 98,90
Alshamy et al. (2021)	NSL-KDD	RF	SMOTE (*oversampling*)	ACC F1 P S	99,89 99,88 99,88 99,89

Tabela 7 – Continuação.

Al-Imran e Ripon (2021)	Kyoto 2013	XGB	SMOTETomek (*resampling*)	ACC F1 MCC P S	99,08 98,79 98,17 99,83 98,65
Wang et al. (2021)	NSL-KDD	RF	ADASYN (*oversampling*)	ACC F1 P S	99,77 99,78 99,80 99,77
Seth, Singh e Chahal (2021)	CICIDS2018	LightGBM	IsolationForest (*undersampling*)	ACC P S	97,70 , 99,30 96
Chen, Zhou e Yu (2021)	CICIDS2017	RF	ADASYN (*oversampling*)	AUC F1 P S	98,78 95,30 98,50 92,30
Chindove e Brown (2021)	CICIDS2017	RNN	SMOTE (*oversampling*)	ACC F1 P S	100 73 74 82
Bagui e Li (2021)	KDDCUP99	ANN	RUS+ROS (*resampling*)	F1 P S	61,99 60,18 96,84
Varunram et al. (2021)	CICIDS2017	kNN	SMOTE (*oversampling*)	ACC F1 S P	99,98 99,98 99,98 99,98
Sapre, Islam e Ahmadi (2021)	NSL-KDD	RNN (U2R)	GAN (*oversampling*)	S	94
Tauscher et al. (2021)	NSL-KDD	NN	SVM-SMOTE (*oversampling*)	ACC F1 P S	87,52 88,48 93,20 84,22
Jui et al. (2021)	MQTT-IoT	J48	SMOTE (*oversampling*)	ACC	99,86

Tabela 7 – Continuação.

Nagpal, Kaushal e Sharma (2022)	KDDCUP99	SVM (normal)	NI (*oversampling*)	ACC F1 P S	99 79 85 74
Gu et al. (2022)	SWaT	CNN	CB_SMOTE (*oversampling*)	ACC FNR FPR	98,88 1,37 1
Rani et al. (2022)	UNSW-NB15	NN	SMOTE (*oversampling*)	ACC AUC F1 P S	90,76 97,46 91,85 89,32 94,52
Lee, Kim e Choi (2022)	CICIDS-2018	CNN	OSS+BSMOTE (*resampling*)	ACC F1	94,58 91,36
Abedzadeh e Jacobs (2022)	CICIDS-2018	Regressão Linear	Dados Originais	ACC P S	88 88 99
Surya e Selvam (2022)	IoTID20	RF	SMOTE (*oversampling*)	ACC F1 P S	92,03 92 89 95
Abdulkareem et al. (2022)	Bot-IoT	DT+NB+LR	SMOTE (*oversampling*)	ACC F1 P S	100 100 99,96 99,91
Sun et al. (2022)	CICIDS-2017	DT	BSMOTE (*oversampling*)	ACC S	98,08 96,19
Xiang, Xu e Tang (2023)	CICIDS-2017	VT	SMOTE (*oversampling*)	AUC F2 P S	99,42 97,52 97,23 99,10
Verma et al. (2023)	UNSW-NB15	RF	GH (*oversampling*)	ACC S F1	97,70 99,10 98,10
Dasari et al. (2023)	KDDCUP99	RF+XGB	SMOTE (*oversampling*)	ACC F1	98,09 97,89

Tabela 7 – Continuação.

Wang et al. (2023)	NSL-KDD	CNN	RB-SMOTE (*oversampling*)	ACC	99,84
Madwanna et al. (2023)	NSL-KDD	RNN+CNN	SMOTE (*oversampling*)	ACC	98,87
Mouiti et al. (2023)	UNSW-NB15	RF	ADASYN (*oversampling*)	ACC AUC F1 P S	98 98 98 98 98
Gou, Zhang e Zhang (2023)	CICIDS-2017	RF+ET+XGB +LightGBM	SMOTE+RUS (*resampling*)	ACC F1 P S	96,65 96,65 94,90 94,40
Jin et al. (2023)	Própria	CNN	SMOTE (*oversampling*)	ACC F1 FNR P S	93,30 93,30 0,7 93,30 93,30
Rahma et al. (2023)	UNSW-NB15	RNN+LSTM	ROS (*oversampling*)	ACC F1 P S	95,30 58,95 57,57 58,95
Gavrylenko, Vladislav e Khatsko (2023)	UNSW-NB15	GB	SMOTEENN (*resampling*)	ACC F1 P S	94 92 92 84
Bagui et al. (2023)	UNSW-NB15	kNN (k=10) (backdoors)	BSMOTE+RUS (*resampling*)	F1 P S	95,60 96,80 94,40
Mohammadi e Babagoli (2023)	CICIDS-2017	HGS	SMOTETomek (*resampling*)	ACC	99,17

Tabela 7 – Fim.

Na Tabela 7 pode-se observar, uma grande predominância do método de *oversampling* com a técnica de SMOTE que produziram os melhores resultados nas métricas dos trabalhos selecionados. Nota-se, presentes o conjunto de dados CICIDS2017 e o classificador *Random Forest* aparecendo repetidas vezes nos melhores cenários dos trabalhos envolvidos na revisão bibliográfica.

Outro fator que se percebe no trabalho de Alfrhan, Alhusain e Khan (2020), os autores conseguiram resultados com 100% de acertos, apresentando um ótimo ajuste do modelo, podendo ter sofrido um sobreajuste (*Overfitting*). Com isto, o trabalho deve ser investigado mais a fundo, para verificar se a quantidade de dados foram suficientes para treino e o pré-processamento adequado para o processo de classificação.

Porém faz-se necessário ressaltar também, que apesar da Tabela 7 mostrar apenas um método e uma técnica de balanceamento de cada trabalho selecionado, os autores utilizaram outras também para compararem e avaliarem seus resultados.

2.2.3 Discussão dos Trabalhos em Aberto

Nesta seção, são apresentados e discutidos os resultados obtidos na Revisão Sistemática da Literatura por meio de levantamento de trabalhos correlatos. Também informações dos conjuntos de dados, modelos de AM propostos, métodos de balanceamento de dados e técnicas de seleção de características utilizados pelos autores.

Observou-se que *oversampling* é o método mais utilizado pelos autores nos trabalhos selecionados, talvez isto se deva ao fato do grande desbalanceamento das classes nos conjuntos de dados de intrusão, onde a classe minoritária não contém dados suficientes para que o classificador seja eficaz nos resultados, o que torna inviável a aplicação da *undersampling* em alguns conjuntos de dados selecionados para classificação de ataques.

Percebe-se isto no trabalho de Jimoh, Ismaila e Olalere (2019), que utiliza o conjunto de dados CICIDS2017 e cita o grande desbalanceamento das classes, podendo atingir 99:1. Também no trabalho de Karatas, Demir e Sahingoz (2020), os autores utilizam o conjunto de dados CICIDS2018, afirmando que a grande diferença entre as classes interferem na eficiência dos modelos de classificação. Ainda relatam que a relação de desigualdade das classes pode ser medida pela fração entre o número de instâncias da classe (C_i) majoritária (max_i) e da classe minoritária (min_i), onde C_i é o tamanho dados na classe i.

$$Imbalance\ Ratio = \rho = \frac{max_i\{C_i\}}{min_i\{C_i\}} \tag{7}$$

Verificou-se que o SMOTE foi utilizado na maioria dos trabalhos selecionados. Esta técnica tem como função gerar novas amostras sintéticas para classe minoritária utilizando o algoritmo *k-Nearest Neighbors* equilibrando a distribuição das classes de tráfego normal e de ataques.

Também pode-se observar que as derivações de SMOTE como BSMOTE, SMOTEENN e SMOTETomek e SVM-SMOTE aparecem nos trabalhos selecionados proporcionando melhores resultados, do que a classificação com os dados originais, sem a aplicação das técnicas de *oversampling*, *undersampling* e *resampling*.

Conforme o artigo de Chawla et al. (2002), o objetivo da técnica de SMOTE é selecionar exemplos da classe que possua características mais próximas e assim criar novos exemplos sintéticos de modo aleatório no espaço. Um ponto negativo desta abordagem é que os exemplos sintéticos são criados sem considerar a classe majoritária, teoricamente resultando em dados imprecisos se houver uma forte sobreposição para as classes utilizando o BSMOTE, uma vez que este visa selecionar instâncias da classe minoritária que estão mal classificadas, exemplos que são provavelmente ambíguos e em uma região da fronteira onde as classes estão sobrepostas.

Porém, alguns autores sugerem aplicar o SMOTE após uma subamostragem aleatória da classe majoritária, reduzindo um pouco os dados para trabalhar com uma nova amostra na classificação (método de *resampling*). O SMOTE permite especificar as proporções desejadas como argumentos para as classes.

Observou-se que a maioria dos trabalhos selecionados que aplicaram a técnica de *resampling* usaram a técnica de SMOTE ou alguma de suas derivações juntamente com métodos de *undersampling*, como (SMOTE + ENN), (OSS + BSMOTE), (OSS + SMOTE) e (RUS+SMOTE) para alcançarem melhores performance do modelo.

Em afirmação da melhora da atuação com o método de criação de reamostras, observou-se que no trabalho de Bagui e Li (2021), os autores conseguiram ótimos resultados aplicando (RUS + ROS), combinações das técnicas de *undersampling* e *oversampling* em vários conjuntos de dados para produzir reamostragens dos dados e classificá-los com Rede Neural Artificial (ANN). Também no trabalho de Al-Imran e Ripon (2021), os autores conseguiram bons resultados com o método de *resampling* utilizando SMOTETomek.

Deve-se também levar em conta o tipo da classificação a ser feita, por exemplo, a classificação será apenas para uma única classe de ataque ou para várias classes de ataques. Caso seja uma classificação multiclasses, esta deverá ser balanceada de forma individual e somente após aplicar o classificador.

O tempo que leva para fazer a classificação do modelo é outro fator que pode implicar no bom desempenho do modelo desta forma considerando que o conjunto possua poucos dados e o método de *undersampling* pode comprometer a classificação e também que o método de *oversampling* aplicado sozinha não considera a classe majoritária ao criar os dados sintéticos, então podemos usar o método de *resampling* como solução. No trabalho de Bagui e Li (2021), os autores afirmam que *resampling* quase não impacta no tempo de treinamento e também detecta mais dados da classe minoritária, o que torna o método de *resampling* interessante para conjuntos de dados pequenos.

O classificador também possui sua importância no modelo, sua escolha deve ser feita com critérios, levando em conta o tipo de classificação binária ou multiclasses. Percebe-se que o uso de classificadores de árvore de decisão possui uma preferência em cenários de vários trabalhos selecionados. O classificador *Random Forest* foi um dos mais utilizados no trabalho

de Mohseni e Tanha (2021), os autores concluíram que o uso de RF como classificador de base juntamente com o RUSBoost, método de *undersampling* e PCA para redução de *features* foi a combinação perfeita para os resultados atingidos, chegando a Acurácia de 99,70% e G-Mean de 99%.

No caso de uma classificação binária, pode-se explorar mais afundo o classificador de Regressão Logística, recomendado nestas situações e utilizado por Abdulkareem et al. (2022) juntamente com DT e NB, que obtiveram bons resultados com Acurácia e F1 de 100%, Precisão de 99,96% e Sensibilidade de 99,91%. Este algoritmo possui facilidade para lidar com variáveis independentes categóricas, fornecendo resultados em termos de probabilidade com grau de confiabilidade.

A fase da preparação dos dados, em que são aplicadas técnicas de indexação, normalização, redução de dimensionalidade e escolha das melhores *features* é igualmente relevante para construção de um bom modelo. Nos trabalhos selecionados na literatura perceberam-se várias técnicas de redução de características utilizadas, onde alguns autores mencionam que esta é uma das principais etapas para que os dados possam estar consistentes para uma boa classificação. Apesar de muitos trabalhos não informarem a técnica aplicada, percebeu-se que alguns autores utilizaram mais de uma técnica de seleção das características e a mais aplicada nos trabalhos foi a *Principal Component Analysis* (PCA).

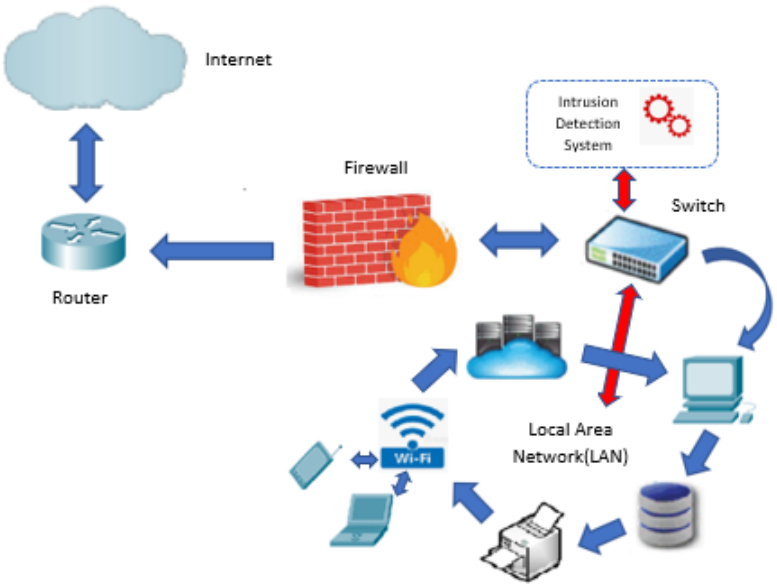
No trabalho de Gonzalez-Cuautle et al. (2020), os autores fazem uso de Análise de Componentes (PCA) para uma redução de dimensionalidade e extração de recursos afirmando que a seleção das principais características descritas nos conjuntos de dados tem alto impacto nas rotinas de treinamento do modelo por ter capacidade de capturar uma máxima variabilidade das entradas.

Desta forma, entende-se que para obter uma classificação com resultados relevantes é necessário testar combinações de diferentes técnicas de diversas formas e conhecer profundamente os dados que serão trabalhados. Modelos de AM devem ser testados com diferentes proporções de dados e classificadores, utilizando métricas que possam proporcionar melhor avaliação dos resultados, garantindo o desempenho do modelo.

2.3 Sistema de Detecção de Intrusão

Os administradores de redes normalmente utilizam vários mecanismos de segurança para proteger suas redes como controle de acessos, criptografia, senhas e *firewalls*. Apesar disso, alguns administradores optam por proteger ainda mais suas redes, usando Sistemas de Detecção de Intrusão (IDSs) para detectar ataques e monitoração do tráfego dos dados pela rede Karatas, Demir e Sahingoz (2020).

Figura 8 – Sistemas de Detecção de Intrusão.



Fonte: Adaptado de Karatas, Demir e Sahingoz (2020) .

Para identificar possíveis ataques, os IDSs baseiam-se em assinaturas ou anomalias de pacotes que entram ou saem de uma rede. Os que utilizam de anomalias para identificar ataques tentam criar um padrão de comportamento de usuários em relação a programas, arquivos e dispositivos para fazerem descobertas de utilizações não autorizadas de dispositivos de rede ou de computadores. Já os por assinaturas comparam os pacotes da rede com assinaturas pré estabelecidas de ataques já identificados Engly, Larsen e Meng (2020).

Conforme Gonzalez-Cuautle et al. (2020), os IDSs levam a altas proporções de Falsos Positivos, o que dificulta e atrasa a identificação de ataques verdadeiros, também podem detectar comportamentos maliciosos muitas vezes equivocados considerando-os como tráfego benigno.

Dentro deste contexto, Anh et al. (2021) citam que os modelos baseados no AM, oferecem uma perspectiva mais precisa e abrangente na identificação de ataques nas redes de computadores. A abordagem normalmente utilizada em AM é treinar os modelos em conjuntos de dados rotulados para fazerem predições das classes pré classificadas como benignas ou maliciosas.

2.4 Aprendizado de Máquina

Aprendizagem de Máquina ou *Machine Learning* é um ramo da Inteligência Artificial que pertence a Ciência da Computação no qual os algoritmos computacionais são programados para imitar a inteligência humana, aprendendo com ambiente no qual está envolvido. Suas técnicas estão sendo empregadas em diversas áreas e obtendo ótimos resultados, como reconhecimento

de padrões, visão computacional, engenharia, entretenimento, biologia e até medicina Naqa e Murphy (2015).

Conforme Janiesch, Zschech e Heinrich (2021) o grande desafio de AM baseia-se em desenvolver modelos capazes de aprender por si, por meio de experiências e padrões de comportamentos passados (aprendizagem não supervisionada), também por meio de entrada de mapas de dados (aprendizagem supervisionada) ou interagindo com o ambiente (aprendizagem por reforço). A entrada da modelagem de AM deve ser utilizada por um algoritmo para identificação padrões e relações importantes. Para esta tarefa AM requer características bem definidas, podendo utilizar diferentes mecanismos nos algoritmos para construção de modelos analíticos.

No trabalho de Rashid, Siddique e Ahmed (2020) é relatado que o Aprendizado de Máquina desempenha uma função importante para a análise detalhada da detecção de intrusão utilizando SVM, kNN e NN para verificar indicadores de desempenho de modelos como a Acurácia, Precisão e taxas de Falsos Positivos (FPR) e Falsos Negativos (FNR) aplicando técnicas em conjunto de dados disponibilizados de forma pública.

2.5 Classificação

Nesta sessão, apresentam-se os algoritmos utilizados para classificação dos conjuntos de dados treinados com as combinações dos processos de seleção de características e técnicas de balanceamento de dados.

2.5.1 Regressão Logística (LR)

A Regressão Logística é uma técnica que permite estimar a probabilidade associada a ocorrência de um evento em vista de um conjunto de variáveis explanatórias. Sua característica principal é estimar a probabilidade da variável dependente assumindo um determinado valor, em função dos dados conhecidos provindos de outras variáveis. Os resultados da análise ficam dentro do intervalo de zero a um. Sua maior utilização é para problemas de previsão de riscos e classificação. Possui facilidade para lidar com variáveis independentes categóricas, fornecendo resultados de probabilidade, requerendo pequeno numero de suposições e com um alto grau de confiabilidade.

Na Regressão Logística, a probabilidade de ocorrência de um evento pode ser estimada diretamente. Exemplo, no caso da variável dependente Y assumir apenas dois possíveis estados (1 ou 0) e haver um conjunto de p variáveis independentes $X1, X2, X3, ..., Xp$, o modelo de Regressão Logística pode ser escrito da seguinte forma:

$$P(Y = 1) = \frac{1}{1 + e^{-g(x)}}$$

onde,

$$g(x) = B_0 + B_1X_1 + ... + B_pX_p$$

Os coeficientes da Regressão Logística

$$B_0, B_1, ..., B_p$$

são estimados a partir do conjunto dados, pelo método da máxima verossimilhança, em que encontra uma combinação de coeficientes que maximiza a probabilidade da amostra ter sido observada.

Nota-se que a curva logística tem um comportamento probabilístico no formato da letra S, o que é uma característica da regressão logística.

a) Quando

$$g(x) \to +\infty$$

então,

$$P(Y = 1) \to 1$$

b) Quando

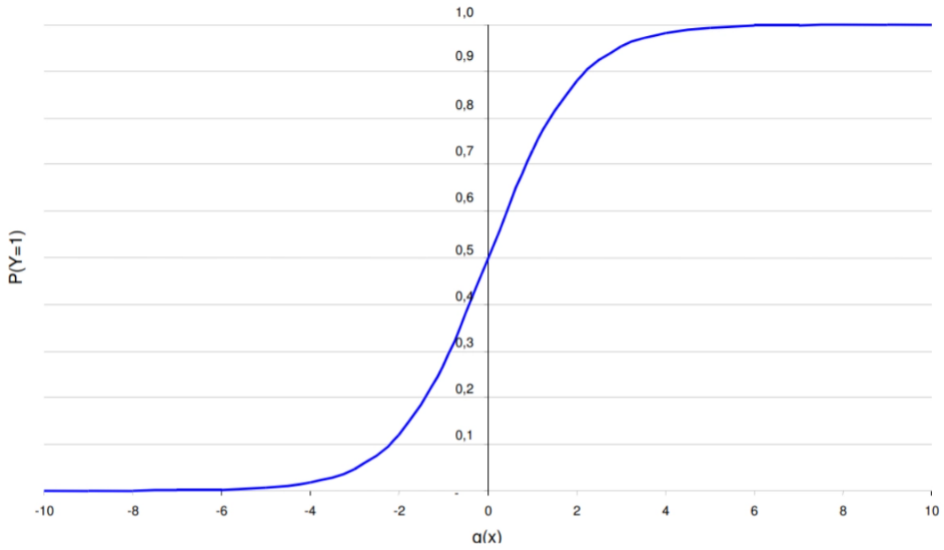
$$g(x) \to -\infty$$

então,

$$P(Y = 1) \to 0$$

Pode-se observar um exemplo de curva logística na Figura 9:

Figura 9 – Curva da Regressão Logística Hosmer, Taber e Lemeshow (1991).



Fonte: Adaptado de Hosmer, Taber e Lemeshow (1991).

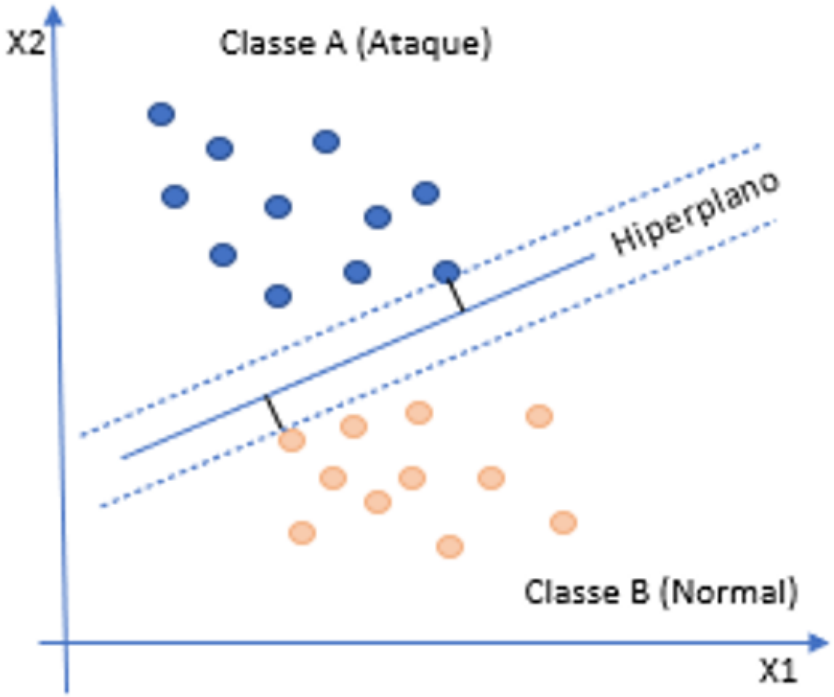
O modelo de Regressão Logística está com seu uso em ascensão, porém a falha em tratar o ajuste do modelo pode conduzir a inferências errôneas e falsas interpretações. Para que isso não aconteça, pode-se obter um modelo matemático, o qual possa interpretar a relação entre observações do resultado ou variável dependente e um conjunto de características independentes, sem referência ao número e escala de medição Hosmer, Taber e Lemeshow (1991).

O classificador LR, foi um dos escolhidos para o modelo do presente trabalho, por se tratar de uma técnica recomendada para uso em classificações binárias com variáveis categórica ou numéricas, e de alta confiabilidade.

2.5.2 Máquinas de Vetores de Suporte (SVM)

O classificador SVM para classes binárias é um algoritmo que procura uma linha divisória entre duas classes distintas, traçando um hiperplano de forma que os exemplos de cada classe fiquem separados. Após, descoberta deste hiperplano, o algoritmo conseguirá predizer a qual classe pertence um novo dado ao checar de qual lado da reta ele pertence. Pode-se observar um exemplo na Figura 10.

Figura 10 – Máquina de Vetor de Suporte



Fonte: Elaborado pelo autor.

Nagpal, Kaushal e Sharma (2022) comenta que a função do SVM é encontrar vetores de suporte para separar a classe negativa e positiva com uma reta que maximiza a margem entre as classes, selecionando pontos de dados próximos ou distantes conforme o limite de

decisão. Completam ainda que em seu trabalho SVM melhorou notavelmente os resultados da classificação.

As vantagens das máquinas de suporte é sua eficácia quando os dados possuem espaços dimensionais elevados, também quando o número de dimensões é grande podendo convergir para um ótimo hiperplano. A desvantagem é que, conforme o tamanho do conjunto de dados vai crescendo, o tempo necessário para fazer os cálculos aumenta rapidamente.

No trabalho de Alfrhan, Alhusain e Khan (2020), os autores comentam que o classificador SVM é indicado para ser utilizado em dados supervisionados e adequado para prever dois resultados. Completam ainda que, dos três classificadores usados por eles, o SVM foi o classificador que conseguiu atingir a melhor performance no conjunto de dados CICIDS2017 juntamente com a técnica de SMOTE.

O SVM foi escolhido para integrar as combinações dos experimentos no presente trabalho por tratar-se de um método eficaz ao lidar com a classificação binária, as diversas dimensões, e pelo baixo custo computacional à medida que mais dimensões e *features* são analisadas.

2.5.3 Random Forest (RF)

O algoritmo de AM *Random Forest* fundado por Breiman (2017) é usado tanto para classificação como também para regressão. Este classificador baseia-se na ideia de combinações de várias árvores de decisão em um único modelo preditivo.

Em RF cada árvore na floresta é treinada em uma amostra aleatória dos dados de treino, considerando cada "nó" e em cada divisão da árvore, onde apenas um subconjunto aleatório das características (variáveis) é considerado. Pode-se observar um exemplo de RF na Figura 11.

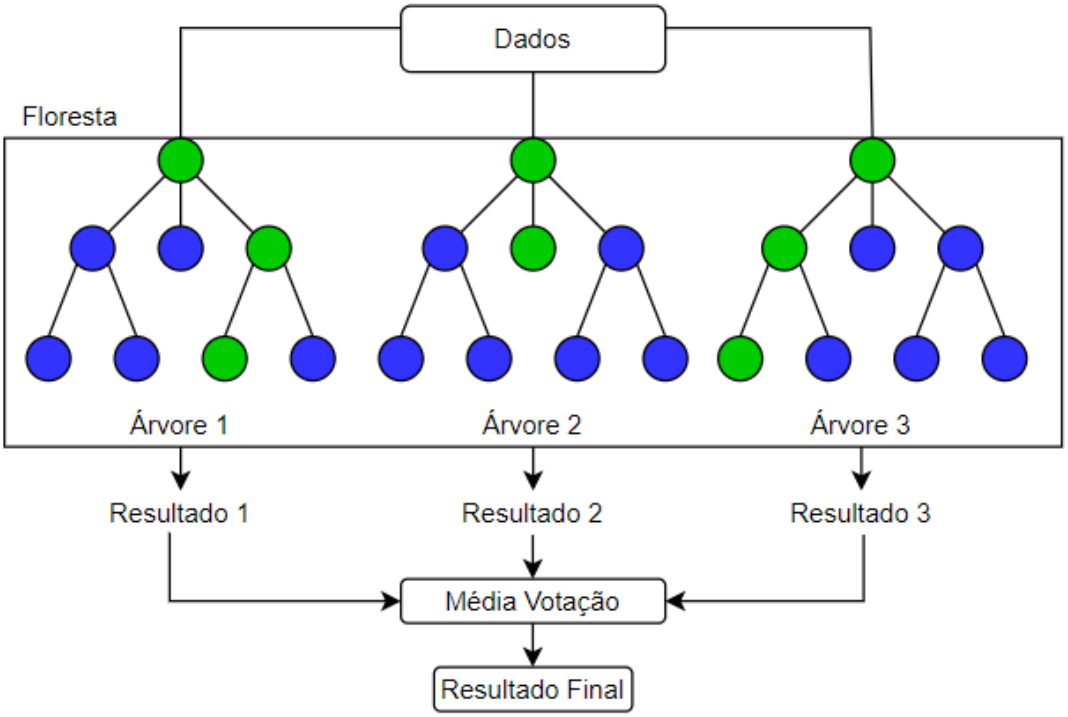
Surya e Selvam (2022) afirmam em seu trabalho que a aplicação de *Random Forest* e o balanceamento os dados com SMOTE pode-se obter bons resultados de Acurácia nos modelos classificados de forma multiclasses ou binária.

As vantagens do *Random Forest* é que favorece a correlação entre as árvores individuais do modelo de dados, tornando um modelo robusto que ajuda a reduzir o risco de *overfitting*.

Random Forest é um tipo de algoritmo de ensemble[106] que procura melhorar o desempenho preditivo em relação ao que poderia ser alcançado pelos modelos individuais.

[106] Técnica que combina as previsões de vários modelos de aprendizado de máquina para produzir uma previsão final.

Figura 11 – Random Forest



Fonte: Elaborado pelo autor.

O classificador RF é um metaestimador segundo Alshamy et al. (2021) que ajusta vários classificadores de árvore de decisão em diferentes subamostras do conjunto de dados e usa uma média para melhorar a precisão preditiva e controlar o sobreajuste.

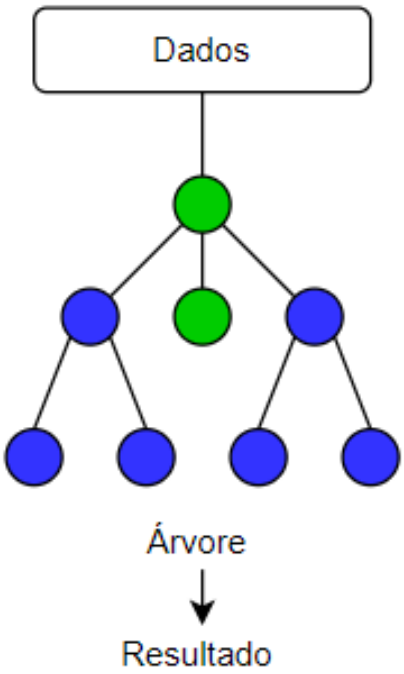
Random Forest foi selecionado para o presente trabalho por ser robusto e ajustar o *overfitting*.

2.5.4 Decision Tree (DT)

O algoritmo de AM *Decision Tree* baseia-se em uma árvore de decisão, ou seja uma representação visual dos processos de tomada de decisão onde um fluxograma com "nós" e ramificações, onde cada "nó" representa um evento e cada ramificação representa um resultado ou um caminho possível. A Figura 12 apresenta uma representação gráfica do exemplo de *Decision Tree*.

Sun et al. (2022) comentam em seu trabalho que o processo de tomada de decisão de *Decision Tree* se inicia no "nó" raiz, logo em seguida, afere os atributos característicos correspondentes do item a ser classificado e seleciona os ramos de saída de conforme os seus valores até atingir o "nó" folha. Por fim, armazena as categorias no "nó" folha como resultado da decisão.

Figura 12 – Decision Tree



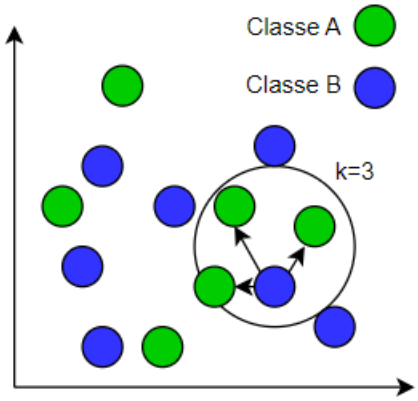
Fonte: Elaborado pelo autor.

No trabalho de Jimoh, Ismaila e Olalere (2019) é implementado o algoritmo de AM *Decision Tree* após os dados balanceados com SMOTE. Os autores comentam que esta combinação pode resolver o problema de desequilíbrio de conjuntos de dados e apresenta-se mais eficaz na detecção de ataques em uma determinada rede. O classificador *Decision Tree* foi escolhido para integrar as combinações do presente trabalho por ser um dos cinco mais utilizados na pesquisa do trabalho.

2.5.5 k-Nearest Neighbors (kNN)

O kNN é um tipo de algoritmo de aprendizado supervisionado usado tanto para regressão quanto para classificação. O kNN tenta prever a classe correta para os dados de teste por meio da distância entre os dados de teste e todos os pontos de treinamento. Em seguida, ele seleciona o número "k" de pontos que estão mais próximo dos dados de teste. O classificador kNN calcula a probabilidade dos dados de teste pertencerem às classes de dados de treinamento "k" e a classe que detém a maior probabilidade será selecionada. A Figura 13 mostra um exemplo da representação gráfica para kNN.

Figura 13 – k-Nearest Neighbors



Fonte: Elaborado pelo autor.

No trabalho de Varunram et al. (2021) a combinação que apresentou melhor desempenho foi kNN com as métricas de Acurácia, Precisão, Sensibilidade e F1. Os autores citam que kNN é o mais simples entre todos os algoritmos de AM supervisionado, funcionando da seguinte forma: Ele procura k vizinhos ou pontos de dados mais próximos e atribui a classe majoritária ao ponto de dados recém-adicionado, calculando as distâncias por meio das medidas Manhattan, Euclidiana, Minkowski ou Hamming. Complementam ainda que quanto maior o valor de k, mais tempo será gasto para a classificar os dados.

O classificador kNN foi selecionado para o trabalho por ser um dos cinco mais utilizados pelos autores nos trabalhos pesquisados, também por sua simplicidade e por ser extremamente fácil de implementar.

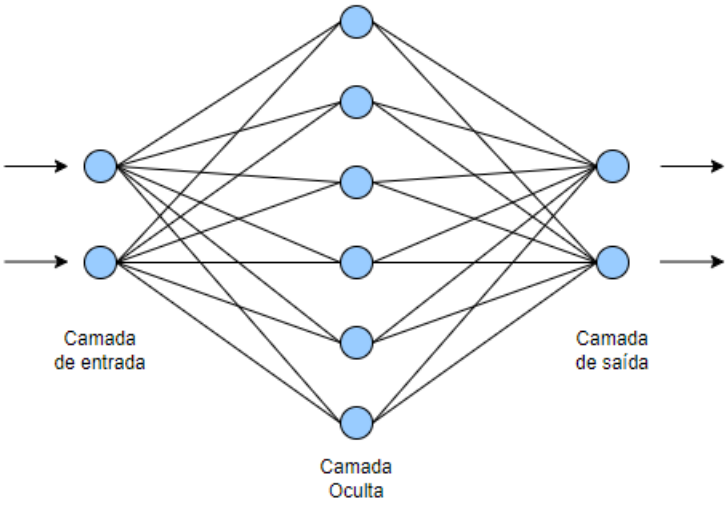
2.5.6 Rede Neural Artificial (RNA)

O classificador RNA selecionado para as classificações no presente trabalho foi do tipo Perceptron Multicamadas (MLP). Este tipo de RNA é composta de camadas de neurônios com camada de entrada e de saída. Os neurônios de uma camada conecta-se a todos os neurônios da camada seguinte, formando uma rede altamente conectada.

Os autores Mouiti et al. (2023) aplicaram MLP e obtiveram resultados satisfatórios de desempenho do modelo sem o uso de ajustes de hiperparâmetros atingindo com a métrica de AUC de 99,90% e com uso de ADASYN 99,6% respectivamente apesar do resultado geral apontar RF conseguir melhores resultados.

Na Figura 14 pode-se observar uma representação de MLP com uma camada de entrada, uma camada interna e uma camada de saída.

Figura 14 – Rede Neural Artificial



Fonte: Elaborado pelo autor.

A vantagem deste classificador em relação aos outros algoritmos de classificação, é a capacidade de aprender funções não lineares complexas, devido à presença de camadas ocultas. Rahma et al. (2023) aplicaram NN para classificar ataques aplicando métodos de *oversampling*, *undersampling* e *resampling* porém relatam que os melhores resultados foram obtidos a partir da classificação com os dados brutos.

Os autores Bagui e Li (2021), citam que em dados desbalanceados a influência do modelo RNA reconhece bem os dados majoritários, mas terá um desempenho inferior no reconhecimento de dados minoritários.

O classificador de Rede Neural foi selecionado para integrar as combinações do presente trabalho por ser um dos cinco mais utilizados na pesquisa do trabalho e apresentar resultados relevantes.

2.6 Métodos de Balanceamento

Nesta seção serão apresentados os algoritmos de balanceamento das classes utilizados no presente trabalho antes do treinamento dos modelos com os classificadores.

Segundo Engly, Larsen e Meng (2020) o desbalanceamento das classes nos conjuntos de dados são utilizados anteriormente dos modelos de treinamento para corrigir desigualdades na quantidade de exemplos evitando perdas de desempenho, o que pode implicar em resultados de baixa precisão de detecção.

Szczepanski et al. (2021), relatam que o modelo treinado com um conjunto de dados que contenham classes de tamanho muito diferentes pode atingir uma maior taxa de Acurácia,

porém pode ser um valor enganoso. Complementa ainda que ao observar outras métricas pode-se perceber que todas as amostras foram classificadas de modo tendencioso para a classe majoritária (*Overfitting*).

2.6.1 SMOTE

O SMOTE (Técnica de sobreamostragem de Minoritários Sintéticos) proposto por Chawla et al. (2002) baseia-se em um método de sobreamostragem usado para tratar o desbalanceamento de classes em conjuntos de dados. Esta técnica visa equilibrar a distribuição da classe minoritária para com a majoritária por meio do aumento aleatório dos registros de treinamento utilizando interpolação linear.

Esses registros sintéticos, segundo Tarawneh et al. (2020), são gerados aleatoriamente usando o método vizinho mais próximo (kNN) para cada exemplo de desequilíbrio da classe; logo após a aplicação desta técnica, a classificação de modelos de aprendizagem podem ser processados.

O Algoritmo SMOTE funciona da seguinte forma:

Etapa 1: Definição do conjunto de classes minoritárias A, para cada $x \in A$, os vizinhos k mais próximos de x são obtidos através do cálculo da distância euclidiana entre x e cada outra amostra em conjunto A.

Etapa 2: A taxa de amostragem n é definida de acordo com o desbalanceamento da proporção das classes. Para cada $x \in A$, n exemplos $(i.e. x1, x2, ...xn)$ são selecionados aleatoriamente a partir de seus vizinhos mais próximos, construindo o conjunto $A1$.

Passo 3: Para cada exemplo $xk \in A1(k = 1, 2, 3, ...n)$, a seguinte fórmula é utilizada para gerar um novo exemplo: $x = x + rand(0, 1)|x - xk|$ no qual $rand(0, 1)$ representa o número aleatório entre 0 e 1.

Haixiang et al. (2017), comentam que o SMOTE pode ser usado tanto em cenários com uma grande quantidade de dados, como para outros com apenas algumas dezenas de amostras da classe minoritária ou juntamente com alguma técnica de subamostragem.

O SMOTE foi uma das técnicas de sobreamostragem escolhidas para implementação no presente trabalho por ser uma das mais aplicadas pelos autores nos trabalhos selecionados.

2.6.2 SVMSMOTE

O método de *oversampling* utilizando SVMSMOTE proposta por Nguyen, Cooper e Kamei (2009), é uma derivação do algoritmo BSMOTE que diferentemente do algoritmo kNN, utiliza-se do *Support Vector Machines* para detectar amostras a serem usadas para gerar novas amostras sintéticas.

Nesta abordagem, o algoritmo SVM localiza os exemplos no limite de decisão entre as classes do conjunto de dados para criar novos exemplos sintéticos na classe minoritária.

Também o SVMSMOTE utiliza-se da área limítrofe na qual as instâncias são aproximadas pelos vetores de suporte. Desta forma, SVMSMOTE trabalha criando novos exemplos com o foco apenas nos exemplos de classes minoritárias situadas ao redor da fronteira usando interpolação.

Tauscher et al. (2021) realizam testes para avaliar a técnica SVMSMOTE e concluem claramente que as novas amostras sintéticas geradas pelo SVMSMOTE ajudam o modelo a aprender padrões e melhoram significativamente a pontuação F1 e diminuírem o impacto na métrica de Acurácia.

O SVMSMOTE foi selecionado para implementação dos testes aplicado neste trabalho por se tratar de um método de *oversampling* que conforme Bagui et al. (2023) difere do BSMOTE porque gera de novos dados sintéticos na linha criada pela união do vetor de suporte da classe minoritária com seus vizinhos mais próximos, não utilizando valores kNN para gerar o limite de decisão, eles são usados apenas para selecionar o k-ésimo vizinho mais próximo na geração de novas instâncias.

2.6.3 NearMiss

Uma dos métodos para equilibrar uma distribuição de classes em um conjunto de dados com um arranjo distorcido é *undersampling*. Este método visa a remoção aleatória de exemplos da classe majoritária no conjunto de dados de treinamento para uma melhor distribuição.

NearMiss foi proposto por Jianping e Inderjeet (2003), baseia-se no método *undersampling* selecionando exemplos com base na distância de exemplos de classe majoritária para exemplos de classe minoritária.

No trabalho de Mqadi, Naicker e Adeliyi (2021), os autores citam que existem várias técnicas que utilizam *undersampling* e uma delas é o algoritmo NearMiss, que introduz algumas heurísticas no processo de seleção de amostras para a análise dos vizinhos mais próximos, com *k-Nearest Neighbour* (kNN) a fim de selecionar as instâncias do conjunto de dados que será subamostrado.

Quando os registros de duas classes diferentes estão muito próximos um do outro, a função do algoritmo NearMiss é igualar a distribuição das classes, equilibrando as quantidades de amostra entre elas. O algoritmo, resolve os problemas de subamostragem, melhorando o processo de classificação por meio de algoritmos de Aprendizagem de Máquina Tarawneh et al. (2020).

A técnica NearMiss possui três versões: na versão 1, encontra os dados do teste com a menor distância média para as amostras negativas mais próximas da classe. Na versão 2,

escolhe as amostras positivas com a distância média mais curta para as amostras mais distantes da classe negativa. Na versão 3, o procedimento passa por duas etapas: na primeira etapa, os vizinhos mais próximos de cada amostra negativa devem ser conservados. As amostras positivas são então escolhidas com base na distância média entre elas e seus vizinhos mais próximos.

Em suma, o algoritmo NearMiss funciona da seguinte forma: o algoritmo primeiro calcula a distância entre todos os pontos da classe majoritária com os pontos da classe minoritária. Logo após, seleciona instâncias da classe majoritária que possuam a menor distância com a classe minoritária para eliminação. Então, se tiver m exemplos da classe minoritária, o algoritmo retornará $m * n$ exemplos da classe majoritária.

De acordo com Fernández et al. (2018), a limitação do método *undersampling* é a eliminação de dados sem se atentar da importância que possuem para a classificação. Também ressaltam que as técnicas que utilizam-se de métodos de *undersampling* fazem uso de heurísticas para superar as limitações com abordagens criteriosas são mais vantajosas.

O NearMiss foi escolhido como técnica que usa *undersampling* para ser aplicada no presente trabalho por utilizar de heurísticas no processo de remoção de instâncias que menos impactam no processo de uniformizar as classes do conjunto de dados, também por ser uma técnica pouco testada nos trabalhos selecionados da presente literatura.

2.6.4 SMOTEENN

A técnica SMOTEENN combina dois métodos, *oversampling* realizada por SMOTE com a regra de *undersampling* aplicada com *Edited Nearest Neighbor* (ENN).

A regra editada do vizinho mais próximo (ENN), foi proposta por Wilson (1972), é uma técnica que se utiliza do método de *undersampling* procurando ruído nos dados, calculando kNN para cada amostra de classe minoritária, e decidindo entre os registros se existe um impostor, se sim, essa amostra é excluída.

A combinação dos dois métodos combinados foi testada por Lu et al. (2019), que comentam que apesar do método de *oversampling* equilibrar as classes do conjunto de dados, podem também sobrepor registros da classe minoritária, o que pode causar um desempenho ruim do classificador. Como solução sugerem combinar com a regra editada do vizinho mais próximo (ENN).

Complementam também que, com SMOTE+ENN, os dados treinados são sobreamostrados por SMOTE, em seguida ENN, procura em cada amostra 'três mais próximos vizinhos', removendo os registros classificados erroneamente, produzindo dados mais limpos. Assim, pode-se não somente ajustar a distribuição de dados, mas também os limites entre as classes ficam mais distintos.

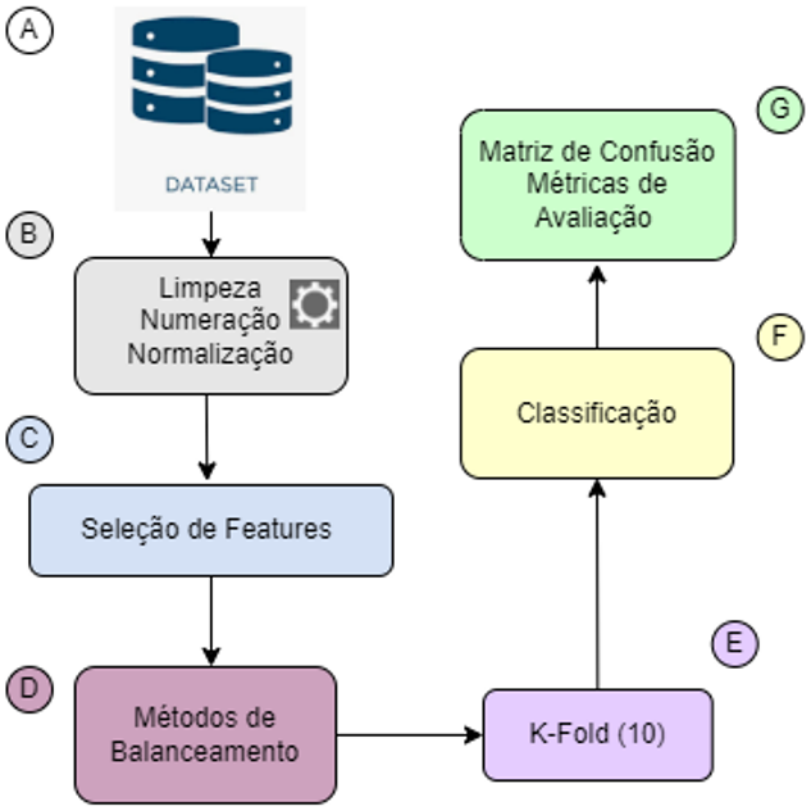
A técnica SMOTEENN apesar de não ser uma das técnicas mais utilizadas nos trabalhos

selecionados na Revisão Sistemática da Literatura foi escolhida para ser aplicada no presente trabalho por mostrar-se eficaz obtendo atuações ótimas nos trabalho de Lu et al. (2019) que classificaram os dados com RF e também no trabalho de Szczepanski et al. (2021) que executaram com Redes Neurais.

3 Metodologia

Este capítulo apresenta as informações das Etapas do fluxo de trabalho, a descrição do conjunto de dados utilizado, os materiais e o pré-processamento (Limpeza, Preparação dos Dados e Seleção de Características), os hiperparâmetros e validação cruzada utilizados nos experimentos.

Figura 15 – Fluxo de trabalho do processo metodológico



Fonte: Elaborado pelo autor.

Na Figura 15 nota-se o processo metodológico aplicado no Trabalho. Cada etapa do pipeline está devidamente rotulada e o fluxo de trabalho é explicado na sequência.

Etapa A: Aquisição dos dados originais do *dataset* CICIDS2017 podem ser observados de forma detalhada na seção 3.1. Nesta etapa, os dados ainda não estão discretizados ou normalizados, também não estão balanceados.

Etapa B: Compreende-se o pré-processamento de dados onde se aplica a Limpeza, Numeração e Normalização, deixando-os preparados para a divisão (separação dos dados de treino e teste). Pode-se verificar esta etapa de forma detalhada na subseção 3.2.1.

Etapa C: São realizadas as seleções de características aplicando duas técnicas *Recursive*

Feature Elimination e *Mutual Information* detalhados na subseção 3.2.2. Estes métodos de seleção de características serão combinados com os classificadores escolhidos e com os métodos de balanceamento para predições dos *Portscan*.

Etapa D: Representa a aplicação ds métodos de *undersampling*, *oversampling* e *resampling*, aplicados por meio das técnicas de balanceamento selecionadas para o trabalho, podendo serem observadas na seção 2.6.

Etapa E: A aplicação da validação cruzada de 10 vezes que será utilizada para todas as classificações. Esta etapa pode ser verificada de maneira especificada na subseção 3.2.3.

Etapa F: Realiza o ajuste adequado dos classificadores selecionados ajustando os hiperparâmetros. Os ajustes também podem ser observados detalhadamente na subseção 3.2.3.

Etapa G: Representa a geração da matriz de confusão para verificar a quantidade de Falsos Positivos e Falsos Negativos e avaliar a capacidade de identificação de *Portscan* em cada modelo. Também aplicam-se as métricas de ACC, Precisão, Sensibilidade, Pontuação F1, MCC, Curva Roc. Também foram aplicadas as métricas Raiz Do Erro Quadrático Médio (RMSE)[107], a Função de Perda, ou Entropia Cruzada (*Log Loss*)[108] e o também o cálculo do tempo gasto de treinamento para uma avaliação mais completa dos experimentos.

3.1 Conjunto de Dados

O CICIDS2017 foi desenvolvido pelo *Canadian Institute for Cybersecurity* (CIC), com o objetivo de avaliar desempenho para detecção de anomalias, após testes realizados utilizando conjuntos de dados disponibilizados publicamente na Internet. Destes, a maioria continha registros desatualizados e não confiáveis, e outros possuíam falta de diversidade devido a grande variedade de ataques conhecidos.

O CICIDS2017 foi inicialmente extraído de um total de 84 características do tráfego de rede a partir de um arquivo pcap, através do extrator de características baseado em fluxos *CICFlowMeter*, incluindo IP de origem, porta de origem, IP de destino, porta de destino e protocolo. Todas as características foram definidas e explicadas no site da ferramenta *CICFlowMeter* conforme Sharafaldin, Lashkari e Ghorbani (2019).

Neste trabalho, foi selecionado o CICIDS2017 para os experimentos por ser um dos mais utilizados nos trabalhos pesquisados na revisão sistemática da literatura e também por ser um conjunto de dados ainda recente, possuindo dados de diversos ataques.

No trabalho de Alfrhan, Alhusain e Khan (2020), os autores relatam que o conjunto de dados desenvolvido pelo Instituto Canadense de Segurança Cibernética com a intenção de uso

[107] Métrica que calcula a raiz quadrada da média dos quadrados dos erros, onde o erro bruto é a diferença entre o valor previsto pelo modelo e o valor real.
[108] Métrica com objetivo medir a performance do classificador e encontrar o menor valor de perda possível.

para detecção de intrusões, possui algumas deficiências como registros redundantes e ausência de etiqueta para identificação da classe, também contém um alto desbalanceamento de classes que pode conduzir a erros de classificação tendenciosa para classe majoritária.

Porém, de acordo com Chen, Zhou e Yu (2021), o conjunto de dados CICIDS2017 possui tráfego realista de redes, com fluxos de dados que contém os últimos ataques comuns do mundo real. O conjunto de dados é dividido em oito arquivos:

1- Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv;

2- Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv;

3- Friday-WorkingHours-Morning.pcap_ISCX.csv;

4- Monday-WorkingHours.pcap_ISCX.csv;

5- Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv;

6- Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv;

7- Tuesday-WorkingHours.pcap_ISCX.csv;

8- Wednesday-workingHours.pcap_ISCX.csv.

Os arquivos possuem dados de IPs de origem e destino, portas de origem e destino, protocolos, *timestamp* de diversos ataques em arquivos com extensão .csv Chindove e Brown (2021).

No trabalho de Pawlicki et al. (2020) os autores citam que o CICIDS2017 é um conjunto de dados, desenvolvido com uma interpolação do comportamento de 25 usuários utilizando múltiplos protocolos, sendo uma coleta rotulada de 5 dias, em que 4 dias a estrutura foi disposta sob o cerco de muitos ataques.

Para a implementação da proposta foi utilizado o arquivo Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv, com 79 colunas e 286.462 linhas sendo 127.522 de tráfego normal e 158.930 de ataques de *Portscan* e 1 linha contendo os rótulos sendo a coluna Label o *Target* (coluna com as etiquetas) para classificação.

Percebe-se, que o desbalanceamento das classes no arquivo (Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv), não possui uma acentuada diferença, o que não inviabiliza a aplicação do método de *undersampling*, de onde são retirados exemplos da classe majoritária para igualar a quantidade de registros.

3.2 Pré-Processamento

Na opinião de Nagpal, Kaushal e Sharma (2022) a etapa do pré-processamento é principal fase para atingir bons resultados em modelos de AM, pois neste momento aplica-se a remoção de dados redundantes, irregularidades e ruídos.

Já Al-Imran e Ripon (2021) comentam que a realização de uma preparação detalhada dos dados, como a limpeza, indexação dos dados categóricos, normalização e discretização atribuindo uma etiqueta para cada valor numérico mantendo uma relação ordenada, aumenta o desempenho do modelo. Complementa ainda que a utilização de um conjunto de dados contendo ruído, valores nulos e forma irregular de dados, pode caminhar ao desempenho desastroso do modelo.

Nesta seção, será apresentado a execução do pré-processamento da proposta aplicada neste trabalho como limpeza dos dados, métodos de seleção de características e métodos de balanceamento de classes realizados. A técnica de pré-processamento e classificação dos dados fora realizado utilizando-se a linguagem Python.

3.2.1 Limpeza e Preparação dos Dados

Em uma primeira etapa na preparação dos dados substituiu-se na coluna "Label" nosso *target*, as etiquetas "BENIGN" (Normal) e "PortScan" (Ataque) por valores numéricos de "0" e "1" respectivamente, pertencentes ao arquivo Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv, com 79 colunas e 286467 linhas. Em seguida, foram removidos os valores *missing* (dados faltantes) e NaN (*Not a Number*), que são valores geralmente retornado como resultado de uma operação com operandos de entrada inválidos, especialmente em cálculos de ponto flutuante. Exemplo, quando uma operação tenta dividir zero por zero, isso retorna um resultado NaN. Por fim, transformou-se colunas que possuíam valores categóricos em numéricos e os tipos de dados de algumas características que não estavam apropriados, foram alterados de (*int64*) para o tipo de dados (*float64*) e logo após indexados.

Não remover esses valores citados no parágrafo acima, pode implicar em ruídos na classificação. Wang et al. (2021), comentam também que dados ilegíveis podem incluir algum ruído na classe que pertencem.

Na segunda etapa, separou-se as *features* na variável "X" e inserimos na variável "y" somente o *target*, ou seja a coluna "Label" (o que pretende-se prever), passando assim para etapa de seleção de características.

3.2.2 Seleção de Características

Os conjuntos de dados utilizados para classificações de ataques de intrusão abrangem várias características coletadas pelos softwares IDSs. Segundo, Rashid, Siddique e Ahmed (2020) apesar dos conjuntos de dados de IDS conter muitos dados importantes, algumas características possuem dados que são pouco relevantes para os modelos de Aprendizado de Máquina.

No artigo de Nagpal, Kaushal e Sharma (2022), afirmam que a seleção de características é um passo importante na construção da classificação dos modelos de AM, otimizando o processo,

removendo as redundâncias e atribuindo pontuações aos recursos de entrada, apontando o valor relativo de cada recurso em busca de uma melhor predição.

No modelo proposto neste trabalho, testou-se dois processos de seleção de características para serem selecionados 20 *features* de 78 no pré-processamento do conjunto de dados CICIDS2017:

O *Recursive Feature Elimination* (RFE), proposto por Guyon et al. (2002) baseia-se em uma forma simples de seleção de características para construção de modelos de predição, no qual um estimador externo insere pesos às características (coeficientes de um modelo linear). Sua função é de selecionar as características calculando a pontuação do preditor de cada recurso no conjunto de dados, os recursos que obtiverem uma pontuação baixa do classificador são removidos até que as melhores "n" características para o conjunto de dados sejam selecionadas.

O outro método escolhido para os testes de seleção de características deste trabalho foi Informação Mútua, do inglês *Mutual Information* (MI), em que um valor não negativo mede a dependência entre as variáveis. De acordo com Kraskov, Stögbauer e Grassberger (2004), MI faz uma estimação das informações mútuas para uma variável de destino discreta (variáveis numéricas), classificando com "0" quando são independentes, e com valores crescentes conforme uma maior correlação entre as variáveis.

O RFE é classificado como um método de seleção de características *Wrapper*, sua validação é medida pela performance do modelo no conjunto de dados, fornecendo um rank de pontuação a partir da quantidade de *features* desejada para o modelo. Já o MI é considerado um método *Filter*, que utiliza-se de técnicas estatísticas com base no relacionamento dos recursos (correlação), pontuando-as de forma que as com melhores pontuações são as que possuem uma maior ligação.

A escolha dos métodos de seleção de características partiu de uma pesquisa das técnicas de seleção do tipo *Wrapper* e *Filter* que poderiam ser mais relevantes para o modelo:

- Matriz de Correlação método que usa os coeficientes de correlação entre as variáveis apresentando um relacionamento linear entre as variáveis e quanto maiores a correlação melhor para o modelo.
- *Feature Importance* método *Wrapper* que retorna um array onde cada elemento dele é uma *feature* do seu modelo e quanto maior o valor, mais importante a *feature*.

Chen e Jeong (2007) relatam que o RFE busca melhorar a performance da generalização removendo os recursos menos importantes cuja exclusão terá menor efeito nos erros de treinamento. também remove recursos redundantes e fracos e mantém recursos independentes que melhoram a separação das classes.

Após a pesquisa, optou-se por RFE e MI. Já a quantidade de *features* que seriam

selecionadas foi com base nas mais pontuadas pelos métodos de seleção, que a partir da vigésima o resultado da pontuação diferenciava-se enormemente, assim manteve-se as 20 características mais pontuadas com MI e RFE.

Huang, Cai e Xu (2006) comentam em seu trabalho que as regras para seleção de recursos desempenham um papel essencial nas abordagens de filtros. Eles são usados para classificar os recursos e medir a qualidade de um recurso ou subconjunto de recursos, as medidas são facilmente afetados pelo ruído ou dados discrepantes nas *features*. Já as medidas de informação, como entropia ou informação mútua, analisam a quantidade de informação (incerteza) de uma característica ou subconjunto de características e têm sido amplamente usufruídas para seleção de recursos.

Os autores Al-Imran e Ripon (2021) citam que a seleção de recursos precisam de percepções relevantes, que podem ser identificados com o coeficiente de relação que fornece o grau de correlação linear entre duas características.

Na Tabela 8 observa-se as 20 características (*features*) mais bem avaliadas por RFE e MI no arquivo escolhido do conjunto de dados CICIDS2017 e também seus índices. As *features* destacadas em negrito foram as escolhidas por ambos métodos.

Tabela 8 – *Features* selecionadas por RFE e MI. Fonte: Elaborado pelo autor.

RFE	**MI**
0 - Destination Port	**40 - Packet Length Mean**
1 - Flow Duration	**41 - Packet Length Std**
3 - Total Backward Packets	42 - Packet Length Variance
5 - Total Length of Bwd Packets	**0 - Destination Port**
6 - Fwd Packet Length Max	52 - Average Packet Size
13 - Bwd Packet Length Std	69 - min_seg_size_forward
14 - Flow Bytes/s	67 - Init_Win_bytes_backward
21 - Fwd IAT Mean	**14 - Flow Bytes/s**
23 - Fwd IAT Max	**66 - Init_Win_bytes_forward**
24 - Fwd IAT Min	62 - Subflow Fwd Bytes
28 - Bwd IAT Max	4 - Total Length of Fwd Packets
29 - Bwd IAT Min	8 - Fwd Packet Length Mean
34 - Fwd Header Length	53 - Avg Fwd Segment Size
36 - Fwd Packets/s	**6 - Fwd Packet Length Max**
37 - Bwd Packets/s	**5 - Total Length of Bwd Packets**
40 - Packet Length Mean	65 - Subflow Bwd Bytes
41 - Packet Length Std	54 - Avg Bwd Segment Size
47 - ACK Flag Count	12 - Bwd Packet Length Mean
66 - Init_Win_bytes_forward	10 - Bwd Packet Length Max

77 - Idle Min	11 - Bwd Packet Length Min

De acordo com Kuhn, Johnson et al. (2013) as pontuações são úteis e podem ser usadas em situações em problemas de modelagem preditiva, para uma melhor compreensão dos dados, melhor entendimento do modelo e redução do número de recursos de entrada.

Faz-se ressaltar que o método de redução de dimensionalidade Análise de Componentes Principais (PCA) encontrado em vários trabalhos selecionados, também foi testado para seleção de características, porém obteve-se resultados inferiores do que RFE e MI, por perda de Informação ou pela assimetria dos dados utilizados. Assim foram eliminados para as combinações do trabalho.

Logo após a seleção das características prossegue com a Validação Cruzada e o ajuste dos valores adicionados nos classificadores.

3.2.3 Hiperparâmetros e Validação Cruzada

Os classificadores necessitam de alguns ajustes para serem aplicados e o modelo possa ser reproduzido, garantindo obter a mesma divisão sempre que o roteiro for executado, desta forma o parâmetro "random_state" foi definido como 42 na validação cruzada e nos classificadores.

Com o classificador de LR também aplicou-se o número máximo de iterações necessárias de 200 para a convergência dos solucionadores e otimização na classificação, os demais parâmetros ficaram como padrão.

Para otimizar a classificação com SVM aplicaram-se também os parâmetros "kernel" tipo linear[109], útil para combinação linear das características originais, "shrinking" como True para ativar o uso da heurística de encolhimento acelerando o treinamento, "max_inter" foi configurado como 50000 para definir o limite de iterações e otimizar a classificação e "chache_size" como 1000 megabytes usado para armazenar parte da matriz de kernel e para acelerar o processo de treinamento, reduzindo o número de acessos à memória principal.

Em kNN adicionou o valor de "k" definindo como 3 no parâmetro "n_neighbors" para usar na classificação os 3 vizinhos mais próximos.

Para o uso de NN o parâmetro que define a estrutura da rede neural "hidden_layer_sizes" foi definido com 100 neurônios na camada interna, "activation" como relu *Rectified Linear Activation Function* função que retorna "0" para entradas negativas e a própria entrada para entradas positivas, "solver" como adam otimizador usado para ajustar os pesos da rede durante o treinamento, "alpha" como 0.0001 parâmetro de regularização que controla a força da regularização L2 (penalidades) aplicada aos pesos da rede ajuda a evitar o *overfitting*,

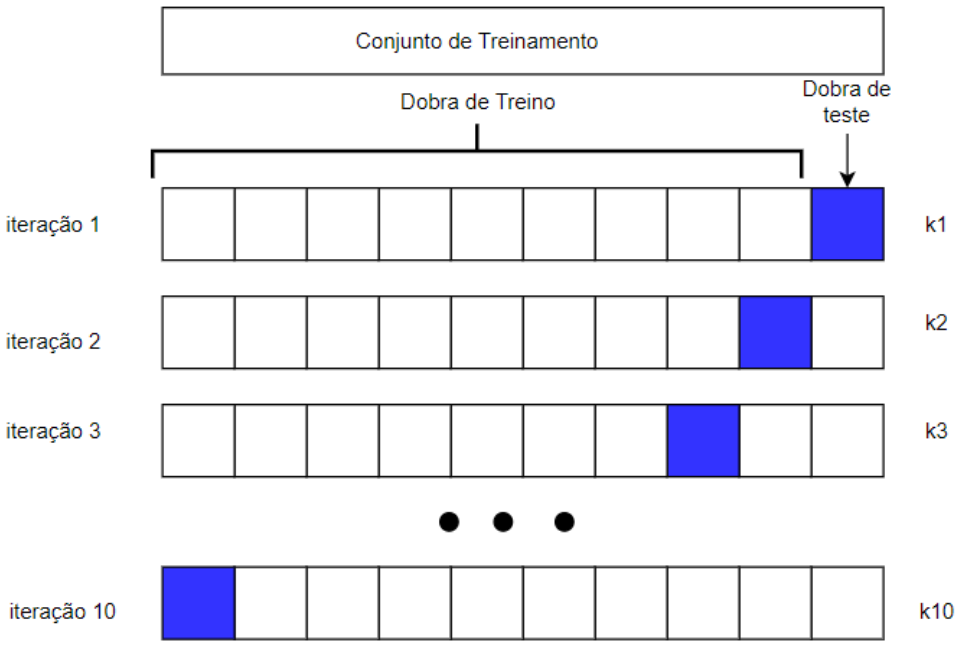
[109] Kernel usado para encontrar o hiperplano que melhor separa as classes no espaço de características transformado.

"batch_size" como auto, ajuste do mini-lote de forma automática durante o treinamento, "learning_rate" ajustado como constante, determina a taxa de aprendizado usada pelo otimizador durante o treinamento, "max_iter" como 200, número de iterações e "shuffle" como True propriedade que indica se os dados de treinamento devem ser embaralhados em cada iteração.

Estes parâmetros foram configurados com base na biblioteca scikit-learn de Pedregosa et al. (2011), biblioteca da linguagem Python desenvolvida especificamente para aplicação prática de AM e utilizado no presente trabalho.

Na Figura 16 pode-se observar um exemplo de validação cruzada de 10 dobras, modelo aplicado no trabalho para calcular a média das métricas.

Figura 16 – Validação Cruzada - 10-fold cross-validation



Fonte: Elaborado pelo autor.

A validação do modelo de AM é uma etapa de extrema importância, afirmam Jimoh, Ismaila e Olalere (2019) que utilizaram a validação cruzada de 10 vezes para validar o modelo de DT com SMOTE e obtiveram a Acurácia de 99.95%.

A validação cruzada funciona da seguinte forma: o conjunto de dados é dividido aleatoriamente em "k" grupos. Logo após definir-se o número de "k", utiliza-se o valor no lugar de "k" fazendo referência ao teste exemplo *10-fold cross-validation*. Em seguida, um grupo (dobra) é separado para teste, enquanto os demais são utilizados para treinamento. Assim treina-se o modelo com os dados de treinamento, e realiza-se o teste com os dados de teste, salvando o valor da métrica e descartando o modelo, em sequência passa-se a selecionar o próximo grupo. Por fim analisa-se os resultados com as métricas em cada uma das iterações.

Os autores Rahma et al. (2023), afirmam em seu trabalho que a validação cruzada segmentada *k-fold* é um método importante para estimar a atuação do modelo, reduzindo o risco de sobreajuste.

4 Resultados e Discussão

Nos testes realizados para este trabalho, utilizaram-se combinações dos algoritmos para gerar modelos de AM para uma comparação dos resultados com o uso de RFE e MI, aplicando NearMiss, SMOTE, SMOTEENN, SVMSMOTE no conjunto de dados CICIDS2017 e classificando-os com os algoritmos LR, SVM, RF, DT, kNN e também com Rede Neural. Para a validação dos resultados nos modelos fez-se uso das métricas ACC, Sensibilidade, Precisão, F1-Score, RMSE, MCC, curva ROC e a função de custo de Entropia Cruzada ou *Log Loss* .

Também utilizou-se da Matriz de Confusão para verificação dos totais de VP, FN, FP, VN e por fim foram calculados o tempo de treinamento que cada classificador gastou para predizer os registros de cada classe do conjunto de dados e assim ser possível analisar a viabilidade do custo computacional dos modelos.

Desta forma, foram construídos 90 modelos no total aplicando combinações de todas as técnicas selecionadas para o trabalho.Também foram realizadas as classificações com os dados originais sem técnicas de seleção de características e de balanceamento, para comparações dos resultados. Os modelos com as combinações podem ser observados na Tabela 9 composta com seguintes itens:

1. Nº - Número de identificação de cada modelo em ordem crescente.
2. Class - Classificadores utilizados para cada modelo nas combinações.
3. Dados - Formato dos dados do *dataset*: Original - sem aplicação de seleção de características; RFE - com seleção de características utilizando *Recursive Feature Elimination*; MI - com seleção de características usando *Mutual Information*.
4. Técnica - Algoritmo de balanceamento aplicado, "Nenhuma" indica que os dados não receberam técnicas de balanceamento.

Tabela 9 – Combinações das Técnicas. Fonte: Elaborado pelo autor.

Nº	***Class***	**Dados**	**Técnica**	**Nº**	***Class***	**Dados**	**Técnica**
#1	NN	Original	Nenhuma	#46	RF	RFE	SMOTE
#2	LR	Original	Nenhuma	#47	DT	RFE	SMOTE
#3	SVM	Original	Nenhuma	#48	kNN	RFE	SMOTE
#4	RF	Original	Nenhuma	#49	NN	RFE	SMOTEENN
#5	DT	Original	Nenhuma	#50	LR	RFE	SMOTEENN
#6	kNN	Original	Nenhuma	#51	SVM	RFE	SMOTEENN

Tabela 9 – Continuação.

#7	NN	Original	NearMiss	#52	RF	RFE	SMOTEENN
#8	LR	Original	NearMiss	#53	DT	RFE	SMOTEENN
#9	SVM	Original	NearMiss	#54	kNN	RFE	SMOTEENN
#10	RF	Original	NearMiss	#55	NN	RFE	SVMSMOTE
#11	DT	Original	NearMiss	#56	LR	RFE	SVMSMOTE
#12	kNN	Original	NearMiss	#57	SVM	RFE	SVMSMOTE
#13	NN	Original	SMOTE	#58	RF	RFE	SVMSMOTE
#14	LR	Original	SMOTE	#59	DT	RFE	SVMSMOTE
#15	SVM	Original	SMOTE	#60	kNN	RFE	SVMSMOTE
#16	RF	Original	SMOTE	#61	NN	MI	Nenhuma
#17	DT	Original	SMOTE	#62	LR	MI	Nenhuma
#18	kNN	Original	SMOTE	#63	SVM	MI	Nenhuma
#19	NN	Original	SMOTEENN	#64	RF	MI	Nenhuma
#20	LR	Original	SMOTEENN	#65	DT	MI	Nenhuma
#21	SVM	Original	SMOTEENN	#66	kNN	MI	Nenhuma
#22	RF	Original	SMOTEENN	#67	NN	MI	NearMiss
#23	DT	Original	SMOTEENN	#68	LR	MI	NearMiss
#24	kNN	Original	SMOTEENN	#69	SVM	MI	NearMiss
#25	NN	Original	SVMSMOTE	#70	RF	MI	NearMiss
#26	LR	Original	SVMSMOTE	#71	DT	MI	NearMiss
#27	SVM	Original	SVMSMOTE	#72	kNN	MI	NearMiss
#28	RF	Original	SVMSMOTE	#73	NN	MI	SMOTE
#29	DT	Original	SVMSMOTE	#74	LR	MI	SMOTE
#30	kNN	Original	SVMSMOTE	#75	SVM	MI	SMOTE
#31	NN	RFE	Nenhuma	#76	RF	MI	SMOTE
#32	LR	RFE	Nenhuma	#77	DT	MI	SMOTE
#33	SVM	RFE	Nenhuma	#78	kNN	MI	SMOTE
#34	RF	RFE	Nenhuma	#79	NN	MI	SMOTEENN
#35	DT	RFE	Nenhuma	#80	LR	MI	SMOTEENN
#36	kNN	RFE	Nenhuma	#81	SVM	MI	SMOTEENN
#37	NN	RFE	NearMiss	#82	RF	MI	SMOTEENN
#38	LR	RFE	NearMiss	#83	DT	MI	SMOTEENN
#39	SVM	RFE	NearMiss	#84	kNN	MI	SMOTEENN
#40	RF	RFE	NearMiss	#85	NN	MI	SVMSMOTE
#41	DT	RFE	NearMiss	#86	LR	MI	SVMSMOTE
#42	kNN	RFE	NearMiss	#87	SVM	MI	SVMSMOTE
#43	NN	RFE	SMOTE	#88	RF	MI	SVMSMOTE

Tabela 9 – Continuação.

#44	LR	RFE	SMOTE	#89	DT	MI	SVMSMOTE
#45	SVM	RFE	SMOTE	#90	kNN	MI	SVMSMOTE

Tabela 9 – Fim.

Observa-se que o resultado das combinações da Tabela 9 produziram diferentes modelos para ser possível avaliar diferentes formas de treinamento para predizer *Portscan* no conjunto de dados CICIDS2017.

No trabalho de Pawlicki et al. (2020), os autores destacam que os métodos de balanceamentos de conjuntos de dados têm um forte impacto sobre o comportamento dos classificadores de AM, e as diversas abordagens de balanceamento devem ser medidas cuidadosamente no contexto de detecção de invasões de rede.

A proposta de Nagpal, Kaushal e Sharma (2022) utiliza o classificador SVM de modo otimizado para os dados de treinamento rotulados (aprendizado supervisionado), juntamente com uma redução de *features*, que remove dados ruidosos, combinado-os para conseguir bons resultados.

A Matriz de Confusão foi utilizada por Alfrhan, Alhusain e Khan (2020), para avaliar diferentes modelos de AM com as métricas de Acurácia, Precisão, Sensibilidade e *F1-Score*. Também citam que a MC, além das 4 métricas básicas ainda fornecem o total de Verdadeiros Positivos que é o número de exemplos de ataques classificados corretamente; Falsos Positivos, número de tráfego normal classificado erroneamente; Verdadeiros Negativos, que são o número de exemplos da classe normal classificado de forma correta pelo algoritmo; e os Falsos Negativos quantidade de amostras de ataques preditas de forma incorreta pelo modelo.

A Curva ROC (*Receiver Operating Characteristic*) ou Curva Característica de Operação do Receptor, utilizada na análise de desempenho de classificadores binários, representada pela relação entre a taxa de Verdadeiros Positivos (Sensibilidade) e a taxa de Falsos Positivos (1 - Especificidade), para diferentes limiares de classificação. A área sob a curva ROC (AUC) é uma medida resumida da performance geral do modelo apontando a capacidade de distinguir as classes. Chawla et al. (2002), cita como a métrica mais adequada como medida para dados desbalanceados.

No presente trabalho optou-se também em aplicar a função de custo Entropia Cruzada ou Função de Perda (*Log Loss*), que avalia se as probabilidades previstas pelo modelo são boas ou ruins. Essa medida é importante, pois penaliza fortemente previsões incorretas e aquelas com maior incerteza. A Função de Perda tem como principal objetivo medir a performance do classificador e encontrar o menor valor de perda possível. O cálculo da função de custo associada à Entropia Cruzada é:

$$L = -y * log(p) - (1 - y) * log(1 - p) \tag{8}$$

Onde "y" é o valor real da instância e "p" a probabilidade da instância pertencer a classe positiva. Como "p" é uma probabilidade, resultará em um valor entre "0" até infinito. A Função de Perda, ou Entropia Cruzada, mede o desempenho de um modelo de classificação cuja saída ficar mais próximo de "0", melhor será a performance do classificador. Segundo Gao et al. (2021) *Log Loss* cresce conforme a probabilidade prevista diverge da etiqueta verdadeira.

Também aplicou-se a Raiz Do Erro Quadrático Médio (RMSE), calculada a partir da raiz quadrada da média dos quadrados dos erros, onde o erro bruto é a diferença entre o valor previsto pelo modelo e o valor real.

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{N}} \tag{9}$$

Onde "N" é o número de amostras, "yi" é o valor observado para a amostra "i" e "ŷi" é o valor previsto pelo modelo para a amostra "i".

Uma das métricas de avaliação no trabalho de Imran et al. (2022) classificando os dados a partir de NN é a métrica RMSE que produz valores positivos, onde "0" denota um ajuste perfeito aos dados. Neste pensamento, os autores relatam que um valor pequeno de RMSE é melhor do que um valor grande.

Em sequência, pode-se verificar os resultados das combinações dos modelos propostos na Tabela 9. Dentre as métricas de validação encontram-se: Acurácia (ACC), Precisão (P), Sensibilidade (S) *F1-Score* (F1), Curva ROC (ROC), as taxas de Verdadeiros Negativos (VN), Falsos Positivos (FP), Falsos Negativos (FN) e Verdadeiros Positivos (VP), Coeficiente de Correlação Mathews (MCC), Raiz Do Erro Quadrático Médio (RMSE), a Função de Perda *Log Loss* (LL) e o tempo gasto para o treinamento dos modelos em minutos. Os resultados obtidos pelos modelos podem ser observados na Tabela 10 que fazem referência as combinações da Tabela 9.

Tabela 10 – Modelos - Resultados das Combinações. Fonte: Elaborado pelo autor.

ACC	**P**	**S**	**F1**	**ROC**	**VN**	**FP**	**FN**	**VP**	**MCC**	**RMSE**	**LL**
Modelo #1 Tempo: 32,6											
98,99	99,36	98,80	99,08	99,01	126294	998	1890	156914	97,95	0.10	0.36
Modelo #2 Tempo: 5,9											
95,51	94,36	97,75	96,02	95,23	118024	9268	3571	155233	90,94	0.21	1.61
Modelo #3 Tempo: 19,7											
50,42	57,69	40,04	47,27	51,70	80650	46642	95203	63601	3,48	49.57	17.87

Tabela 10 – Continuação.

Modelo #4 Tempo: 4,6											
99,99	99,99	99,98	99,99	99,99	127291	1	21	158783	99,98	0.00	0.00
Modelo #5 Tempo: 0,6											
99,98	99,98	99,98	99,98	99,98	127274	18	28	158776	99,98	0.01	0.00
Modelo #6 Tempo: 14,8											
99,81	99,74	99,92	99,83	99,79	126879	413	127	158677	99,61	0.04	0.06
Modelo #7 Tempo: 52,5											
99,63	99,76	99,50	99,63	99,63	126996	296	627	126665	99,27	0.06	0.13
Modelo #8 Tempo: 9,7											
96,42	94,46	98,63	96,50	96,42	119932	6360	1734	125558	92,94	0.18	1.28
Modelo #9 Tempo: 22,9											
65,25	60,10	90,73	72,31	65,25	50630	76662	11790	115502	35,46	58.94	12.52
Modelo #10 Tempo: 8,1											
99,78	99,99	99,57	99,78	99,78	127291	1	539	126753	99,57	0.04	0.07
Modelo #11 Tempo: 5,6											
99,78	99,99	99,57	99,78	99,78	127286	6	541	126751	99,57	0.04	0.07
Modelo #12 Tempo: 17											
97,62	99,82	95,42	97,57	97,62	127083	209	5829	121463	95,34	0.15	0.85
Modelo #13 Tempo: 52,5											
97,31	95,30	99,52	97,37	97,31	151019	7785	752	158052	94,71	0.16	0.96
Modelo #14 Tempo: 11,2											
94,68	93,80	95,69	94,74	94,68	148766	10038	6830	151974	89,39	0.23	1.91
Modelo #15 Tempo: 24											
52,23	52,33	50,08	51,18	52,23	86363	72441	79271	79533	4,47	0.69	17.21
Modelo #16 Tempo: 10,7											
99,99	99,99	99,98	99,99	99,99	158802	2	23	158781	99,98	0.00	0.00
Modelo #17 Tempo: 6,1											
99,98	99,99	99,98	99,98	99,98	158793	11	28	158776	99,97	0.01	0.00
Modelo #18 Tempo: 24,6											
99,88	99,86	99,91	99,88	99,88	158588	216	135	158669	99,77	0.03	0.03
Modelo #19 Tempo: 69,1											
98,97	98,00	99,98	98,98	98,97	155277	3230	21	158571	97,96	0.10	0.36
Modelo #20 Tempo: 29,9											
95,73	94,36	97,27	95,79	95,73	149299	9208	4325	154267	91,50	0.02	1.53
Modelo #21 Tempo: 48											
62,81	57,40	99,40	72,77	62,81	41519	116992	937	157656	37,59	0.60	13.40

Tabela 10 – Continuação.

Modelo #22 Tempo: 31,7											
99,99	99,99	99,99	99,99	99,99	158506	1	1	158591	99,99	0.00	0.00
Modelo #23 Tempo: 27,1											
99,99	99,99	99,99	99,99	99,99	158493	14	12	158580	99,98	0.00	0.00
Modelo #24 Tempo: 42,5											
99,98	99,98	99,99	99,98	99,98	158480	27	14	158578	99,97	0.01	0.00
Modelo #25 Tempo: 320,1											
95,05	93,58	96,74	95,14	95,05	148275	10529	5166	153638	90,16	0.22	1.78
Modelo #26 Tempo: 310,1											
85,81	89,38	81,28	85,14	85,81	143470	15334	29722	129082	71,92	0.37	5.11
Modelo #27 Tempo: 579,7											
86,12	79,53	97,27	87,51	86,12	119063	39741	4326	15478	74,11	0.37	5.00
Modelo #28 Tempo: 311,8											
99,99	99,99	99,98	99,99	99,99	158803	1	23	148781	99,98	0.00	0.00
Modelo #29 Tempo: 308,2											
99,98	99,98	99,98	99,98	99,98	158783	21	22	158782	99,97	0.01	0.00
Modelo #30 Tempo: 318,5											
99,86	99,85	99,88	99,86	99,86	158566	238	182	158622	99,73	0.03	0.04
Modelo #31 Tempo: 22											
99,91	99,93	99,91	99,92	99,91	127191	101	134	158670	99,83	0.02	0.02
Modelo #32 Tempo: 4,7											
99,48	99,23	99,83	99,92	99,53	126063	1229	258	158546	98,94	0.07	0.18
Modelo #33 Tempo: 249,9											
99,11	99,14	99,25	99,20	99,09	125933	1359	1182	157622	98,20	0.09	0.32
Modelo #34 Tempo: 9,5											
99,99	99,99	99,98	99,99	99,99	127291	1	20	158784	99,98	0.08	0.02
Modelo #35 Tempo: 4											
99,98	99,98	99,98	99,98	99,98	127275	17	22	158782	99,98	0.04	0.01
Modelo #36 Tempo: 17,4											
99,98	99,97	99,98	99,98	99,97	127252	40	23	158781	99,95	0.01	0.00
Modelo #37 Tempo: 32,7											
99,51	99,11	99,10	99,50	99,51	127180	112	1134	126158	99,02	0.06	0.17
Modelo #38 Tempo: 9,6											
99,42	99,03	99,81	99,42	99,42	126060	1232	233	127059	98,85	0.07	0.20
Modelo #39 Tempo: 16,7											
99,42	99,03	99,81	99,42	99,42	126060	1232	234	127058	98,85	0.07	0.20

Tabela 10 – Continuação.

Modelo #40 Tempo: 13,4											
99,55	99,99	99,10	99,55	99,55	127291	1	1135	127157	99,11	0.06	0.16
Modelo #41 Tempo: 8,6											
99,57	99,99	99,15	99,57	99,57	127282	10	1074	127218	99,15	0.06	0.15
Modelo #42 Tempo: 20,9											
99,30	99,97	98,62	99,29	99,30	127261	31	1749	125543	98,61	0.08	0.25
Modelo #43 Tempo: 34,6											
99,93	99,93	99,92	99,93	99,93	158713	91	127	158677	99,86	0.26	0.02
Modelo #44 Tempo: 10											
99,45	99,07	99,83	99,94	99,45	127323	1481	260	158544	98,90	0.07	0.19
Modelo #45 Tempo: 120											
99,16	98,47	99,87	99,16	99,16	156350	2454	204	158600	98,33	0.09	0.30
Modelo #46 Tempo: 16											
99,99	99,99	99,98	99,99	99,99	158802	2	19	158785	99,98	0.00	0.00
Modelo #47 Tempo: 8,9											
99,98	99,98	99,98	99,98	99,98	158787	17	19	158785	99,97	0.01	0.00
Modelo #48 Tempo: 25,8											
99,98	99,99	99,98	99,98	99,98	158795	9	23	158781	99,97	0.01	0.00
Modelo #49 Tempo: 46,6											
99,94	99,94	99,95	99,94	99,94	158700	89	76	158703	99,89	0.02	0.01
Modelo #50 Tempo: 25											
97,61	95,89	99,48	97,65	97,61	152049	6755	820	157984	95,29	0.15	0.85
Modelo #51 Tempo: 143,9											
99,10	99,10	99,00	99,10	99,21	157199	1590	1249	157530	98,21	0.09	0.32
Modelo #52 Tempo: 29,8											
99,99	99,99	99,99	99,99	99,99	158788	1	4	158775	99,99	0.00	0.00
Modelo #53 Tempo: 23,7											
99,99	99,99	99,99	99,99	99,99	158779	10	8	158771	99,98	0.08	0.00
Modelo #54 Tempo: 36,4											
100	100	100	100	100	158789	0	0	158779	100	0.00	0.00
Modelo #55 Tempo: 68,7											
99,79	99,68	99,89	99,79	99,79	158308	496	165	158639	99.58	0.04	0.07
Modelo #56 Tempo: 40,7											
97,61	95,89	99,48	97,65	97,61	152049	6755	820	157984	95,29	0.15	0.85
Modelo #57 Tempo: 272											
69,22	72,14	62,61	67,04	69,22	120414	38390	5963	99441	38,77	0.55	11.09

Tabela 10 – Continuação.

Modelo #58 Tempo: 36,4											
99,99	99,99	99,98	99,99	99,99	158803	1	27	158777	99,98	0.00	0.00
Modelo #59 Tempo: 38,6											
99,97	99,95	99,98	99,97	99,97	158737	67	180	1587786	99,94	0.01	0.00
Modelo #60 Tempo: 50,2											
99,99	99,99	99,98	99,99	99,99	158797	7	24	158780	99,98	0.00	0.00
Modelo #61 Tempo: 53,9											
99,79	99,78	99,83	99,81	99,78	126951	341	256	158548	99,57	0.04	0.07
Modelo #62 Tempo: 5,7											
90,03	87,18	96,19	91,46	89,27	104842	22450	6046	152758	80,12	0.31	3.59
Modelo #63 Tempo: 153,2											
90,83	88,30	96,23	92,09	90,10	107047	20245	5977	152827	81,66	0.30	3.30
Modelo #64 Tempo: 9,4											
99,95	99,99	99,91	99,95	99,95	127288	4	130	158674	99,90	0.02	0.01
Modelo #65 Tempo: 5,5											
99,94	99,95	99,91	99,95	99,95	127275	17	129	158675	99,89	0.02	0.01
Modelo #66 Tempo: 14,3											
99,94	99,97	99,91	99,94	99,94	127252	40	130	158674	99,80	0.02	0.02
Modelo #67 Tempo: 38,2											
99,62	99,83	99,40	99,62	99,62	127088	204	757	126535	99,24	0.06	0.13
Modelo #68 Tempo: 13,5											
95,88	94,56	97,36	95,94	95,99	120169	7123	3357	123935	91,80	0.20	1.482
Modelo #69 Tempo: 217,5											
50	100	0	0	50	127292	0	127275	17	0	0.70	18.01
Modelo #70 Tempo: 15,5											
99,78	99,99	99,57	99,78	99,78	127291	1	539	126753	99,57	0.04	0.07
Modelo #71 Tempo: 11,5											
99,78	99,99	99,57	99,78	99,78	127286	6	541	126751	99,57	0.04	0.07
Modelo #72 Tempo: 25,4											
97,62	99,82	95,42	97,57	97,62	127083	209	5829	121463	95,34	0.15	0.85
Modelo #73 Tempo: 33,3											
99,76	99,83	99,68	99,76	99,76	158544	260	496	158308	99,52	0.00	0.08
Modelo #74 Tempo: 8,3											
99,75	99,82	99,68	99,75	99,75	158519	285	501	158303	99,50	0.04	0.08
Modelo #75 Tempo: 117,7											
90,31	86,13	96,10	90,84	90,31	134236	24568	6183	152621	81,18	0.31	3.48

Tabela 10 – Continuação.

Modelo #76 Tempo: 17,1											
99,95	99,99	99,81	99,95	99,95	158801	3	130	158674	99,91	0.20	0.01
Modelo #77 Tempo: 8											
99,95	99,98	99,91	99,95	99,95	158793	11	130	158674	99,91	0.02	0.01
Modelo #78 Tempo: 21,3											
99,95	99,99	99,91	99,95	99,95	158793	11	130	158674	99,91	0.02	0.01
Modelo #79 Tempo: 39,9											
99,80	99,80	99,80	99,80	99,80	158470	306	308	158366	99,61	0.04	0.06
Modelo #80 Tempo: 16,1											
89,50	8494	96,02	90,14	89,51	131781	26995	6310	152364	79,69	0.32	3.78
Modelo #81 Tempo: 221,5											
90,41	86,18	96,25	90,94	90,41	134296	24487	5935	152739	81,39	0.30	3.45
Modelo #82 Tempo: 23,9											
99,99	99,99	99,99	99,99	99,99	158781	1	3	158776	99,99	0.0	0.0
Modelo #83 Tempo: 15,2											
99,99	99,99	99,99	99,99	99,99	158773	9	2	158672	99,99	0.0	0.0
Modelo #84 Tempo: 22,3											
99,99	99,99	100	99,99	99,99	158775	1	0	158674	99,99	0.0	0.0
Modelo #85 Tempo: 89,4											
99,72	99,55	99,88	99,72	99,72	158099	705	180	158624	99,44	0.05	0.10
Modelo #86 Tempo: 112,7											
86,70	80,92	96,04	87,83	86,70	122859	35945	6286	152518	74,72	0.36	4.79
Modelo #87 Tempo: 190,8											
90,24	86,88	94,80	90,67	90,24	136086	22718	8250	150554	80,83	0.31	3.51
Modelo #88 Tempo: 62,2											
99,95	99,99	99,91	99,95	99,95	158798	6	130	158674	99,91	0.02	0.01
Modelo #89 Tempo: 54,6											
99,95	99,98	99,91	99,95	99,95	158779	25	132	158672	99,90	0.02	0.01
Modelo #90 Tempo: 64,8											
99,75	99,59	99,91	99,59	99,75	158155	649	130	158674	99,50	0.01	0.04

Tabela 10 – Fim.

Percebe-se que vários modelos atingiram resultados satisfatórios, porém nota-se que os classificadores de árvores de decisão RF e DT tiveram uma performance superior na maioria das combinações das técnicas de seleção de *features* e métodos de *undersampling*, *oversampling* e *resampling* e também na aplicação com os dados originais.

A combinação do modelo #54 kNN+RFE+SMOTEENN atingiu o melhor resultado levando em conta o conjunto de todas as métricas e as taxas de FP e FN com 100% de acertos nas predições de *Portscan*. Mas o tempo de treinamento ficou quase 7 minutos acima de RF e 12 minutos a mais que DT combinados respectivamente com as mesmas técnicas.

As melhores taxas de Acurácia, Precisão, Sensibilidade, *F1-Score* foram obtidas com o classificador kNN também no modelo #54 com de 100%, mas vários modelos ficaram com uma diferença mínima atingindo 99.99% nas respectivas métricas, porém kNN atingiu valores de VN, FP, FN e VP melhores.

No trabalho de Alfrhan, Alhusain e Khan (2020), relatam que avaliar um modelo que possua classes desbalanceadas com base apenas na métrica de Acurácia pode ser desastroso, assim deve-se levar em conta outras medidas em conjunto para uma boa validação. Complementam ainda que a MC produz além das 4 métricas básicas o total de Verdadeiros Positivos, que é o número de exemplos de ataques classificados corretamente, o que ajuda a verificação se modelo obteve um número assertivo de classificações.

Levando em conta os resultados com todos os classificadores, percebe-se que RFE como técnica de seleção de características atingiu resultados superiores dos que utilizaram MI, contudo observa-se que os modelos #82, #83 e #84 que foram aplicados MI com SMOTEENN apresentaram atuações ótimas, conseguindo taxas de FN e FP muito baixas com os classificadores RF, DT e kNN, indicando que acertaram a maioria das predições.

Constatou-se ainda, que as classificações com os dados originais conseguiram bons resultados, diferenciando pouco dos modelos que tiveram combinações com seleção de características e técnicas de balanceamento. Pode-se notar isto no modelo #4 classificado com RF e que teve um tempo de treinamento de 4,6 minutos e no modelo #5 com DT que apesar de aumentar os FP e FN obteve um tempo de 0,6 minutos.

Também faz-se necessário destacar o modelo #22 RF+SMOTEENN apresentando resultados excelentes com apenas 1 Falso Positivo e 1 Falso Negativo, Curva ROC de 99,99%, RMSE 0.00, LL 0.00 e tempo de treinamento de 31,7 minutos.

Outro modelo que atingiu bons resultados foi o de referência #52 que combina RF com RFE e SMOTEENN. Este modelo apresentou apenas 1 Falso Positivo e 4 Falsos Negativos e atingiu uma Função de Perda (*Log Loss*) e RMSE de 0.00 indicando que o modelo não foi superajustado (*Overfitting*).

Percebeu-se, que o modelo #69 que utiliza o classificador SVM com MI e NearMiss classificou quase todos os dados que eram verdadeiros positivos como FN, dados de ataques como não ataques apontado uma classificação tendenciosa para a classe majoritária. Percebe-se que este modelo resultou em métrica de LL de 18.01 considerada extremamente alta e indicando um problema com o modelo. Possivelmente está falhando em identificar corretamente todas as instâncias positivas no conjunto de dados.

Nota-se, que o tempo de treinamento sem a aplicação de técnicas de seleção de *features* e técnicas de balanceamento foi menor, porém é importante ressaltar que o tempo gasto para seleção de *features* juntamente com a técnica de balanceamento foram calculadas e adicionadas no tempo total de treino, podendo assim analisar a viabilidade do impacto do custo computacional do tempo total gasto no processo.

Percebe-se que os modelos classificados com SVM resulta em tempo de treinamento elevado comparado com os outros classificadores. Também o tempo de todos os classificadores quando combinado com a técnica SVMSMOTE também tem seu tempo consideravelmente aumentado.

Com o uso de NN nota-se que apesar dos resultados com as métricas atingirem bons resultados, os números de FN e FP ficaram maiores indicando uma menor identificação de *Portscan*.

No trabalho de Lu et al. (2019), os resultados com SMOTEENN verificadas com as métricas de F1, P e S foram de 99,99% aplicadas no conjunto de dados KDDCup99, e comentam em seu trabalho que SMOTEENN produz dados mais limpos, removendo amostras que foram classificadas incorretamente, fazendo uma melhor distribuição entre os registros das classes deixando os limites mais claros.

5 Considerações Finais

Diferentes combinações de algoritmos de AM vêm sendo estudadas por pesquisadores em busca de uma melhor detecção de intrusão nas redes de computadores.

Neste estudo, após investigar alguns algoritmos de balanceamento dos dados e extrair tendências e combinações de classificadores, conjunto de dados, técnicas de seleção de características e métricas relevantes utilizadas nos trabalhos investigados, optou-se em testar duas técnicas de seleção de características (RFE e MI), combinadas com quatro técnicas de balanceamento de dados (NearMiss, SMOTE, SVMSMOTE e SMOTEENN) e com os classificadores (NN, LR, SVM,RF, DT e kNN), aplicando-as no conjunto de dados CICIDS2017 para uma comparação dos resultados.

A partir da análise de desempenho dos modelos reconhecidos com as métricas de validação e os dados fornecidos pela Matriz de Confusão, concluiu-se que a técnica de seleção de características do tipo *Wrapper* RFE, como a de correlação MI juntamente com a técnica de *Resampling* (SMOTEENN) utilizando os classificadores kNN, RF e DT produziram ótimos resultados nos testes de identificação de *Portscan*.

Percebeu-se também que a técnica de SMOTEENN com RF (modelo #22) e sem seleção de caraterísticas apresentou resultados ótimos e com um bom custo computacional (tempo de treinamento) se comparados com outros modelos aplicados no trabalho.

Estas combinações superaram as demais, comprovando a importância do método *Resampling* usando a técnica de SMOTEENN que aponta que a subamostragem da classe majoritária combinada com a sobreamostragem da classe minoritária aplicada com SMOTEENN, já testada por Lu et al. (2019) ajustou-se bem a distribuição dos dados no conjunto de dados CICIDS2017.

Na etapa da classificação, em uma análise geral, a predição com os algoritmos de árvore de decisão (RF e DT) superaram os demais testados no trabalho, diante do uso das mesmas combinações. Exceto a combinação (kNN+RFE+SMOTEENN) do modelo #54 que resultou uma performance de 100% a partir das métricas aplicadas, porém com um tempo de treinamento de aproximadamente 7 minutos superior a RF e 12 minutos para com DT.

Mohseni e Tanha (2021), afirmam que RF se trata de um classificador que tende a comportar-se como um método de aprendizado conjunto que constrói múltiplas árvores de decisão (floresta) em dados amostrados do conjunto de dados original e logo em seguida, coleta a previsão de cada árvore para cada instância de dados escolhendo a previsão mais votada como resultado.

Os resultados satisfatórios apresentados com os classificadores kNN, RF e DT e a

técnica SMOTEENN mostram a eficiência que estas combinações exercem sobre o conjunto de dados desbalanceado selecionado para o trabalho, superando as outras combinações deste experimento.

Contudo, verificou-se que para construir modelos de AM para detecção de intrusão deve-se fazer experimentos com diversas combinações de técnicas e observar as que melhores podem atender ao cenário proposto, desde a escolha do conjunto de dados até as métricas corretas para melhor avaliação do modelo.

Para trabalhos futuros estas combinações devem ser testadas utilizando outros conjuntos de dados que possuam um desbalanceamento mais expressivo, também testar novas combinações que não foram aplicadas no presente trabalho.

5.1 Publicações

Faz-se importante ressaltar que durante a execução das disciplinas obrigatórias, além da disciplina de Estudos Especiais e do desenvolvimento da Dissertação para a qualificação, o aluno autor deste documento também conseguiu algumas publicações listadas a seguir:

1. Lucas, T. J., Tojeiro, C. A. C., Pires, R. G., Da Costa, K. A. P., & Papa, J. P. (2020). Machine Learning for Web Intrusion Detection: A Comparative Analysis of Feature Selection Methods mRMR and PFI. In Artificial Intelligence and Soft Computing: 19th International Conference, ICAISC 2020, Zakopane, Poland, October 12-14, 2020, Proceedings, Part I 19 (pp. 535-546). Springer International Publishing.

2. Moreira, A. A., Tojeiro, C. A., Reis, C. J., & Massaro, G. H. , Brito, I. A., Da Costa, K. A. P., (2022). Técnicas De Ensemble Learning Para Sistema De Detecção De Intrusão No Contexto Da Cibersegurança, ISBN 978-989-8704-45-0. IADIS.

3. Lucas, T. J., De Figueiredo, I. S., Tojeiro, C. A. C., De Almeida, A. M. G., Scherer, R., Brega, J. R. F., ... & Da Costa, K. A. P. (2023). A Comprehensive Survey on Ensemble Learning-Based Intrusion Detection Approaches in Computer Networks. IEEE Access.

4. Tojeiro, C. A., Lucas, T. J., Passos, L. A., Rodrigues, D., Prado, S. G., Papa, J. P., & Da Costa, K. A. (2024, July). Enhancing Cyberattack Detection in IoT Environments Through Advanced Resampling Techniques. In 2024 31st International Conference on Systems, Signals and Image Processing (IWSSIP) (pp. 1-6). IEEE.

Referências

ABDI, A. H.; AUDAH, L.; SALH, A.; ALHARTOMI, M. A.; RASHEED, H.; AHMED, S.; TAHIR, A. Security control and data planes of sdn: A comprehensive review of traditional, ai and mtd approaches to security solutions. *IEEE Access*, IEEE, 2024.

ABDULKAREEM, S. A.; FOH, C. H.; CARREZ, F.; MOESSNER, K. Smote-stack for network intrusion detection in an iot environment. In: *2022 IEEE Symposium on Computers and Communications (ISCC)*. [S.l.: s.n.], 2022. p. 1–6.

ABEDZADEH, N.; JACOBS, M. Using markov chain monte carlo algorithm for sampling imbalance binary ids datasets. In: IEEE. *2022 International Conference on Computer Communications and Networks (ICCCN)*. [S.l.], 2022. p. 1–7.

AHA, D. W.; KIBLER, D.; ALBERT, M. K. Instance-based learning algorithms. *Machine learning*, Springer, v. 6, p. 37–66, 1991.

AL-IMRAN, M.; RIPON, S. H. Network intrusion detection: An analytical assessment using deep learning and state-of-the-art machine learning models. *International Journal of Computational Intelligence Systems*, Springer, v. 14, n. 1, p. 1–20, 2021.

ALFRHAN, A. A.; ALHUSAIN, R. H.; KHAN, R. U. Smote: Class imbalance problem in intrusion detection system. In: IEEE. *2020 International Conference on Computing and Information Technology (ICCIT-1441)*. [S.l.], 2020. p. 1–5.

ALSHAMY, R.; GHURAB, M.; OTHMAN, S.; ALSHAMI, F. Intrusion detection model for imbalanced dataset using smote and random forest algorithm. In: SPRINGER. *International Conference on Advances in Cyber Security*. [S.l.], 2021. p. 361–378.

ANH, N. T.; MINH, V. D.; HAI, T. H. et al. anh2021bkidset. In: IEEE. *2021 15th International Conference on Advanced Computing and Applications (ACOMP)*. [S.l.], 2021. p. 106–111.

BAGUI, S.; LI, K. Resampling imbalanced data for network intrusion detection datasets. *Journal of Big Data*, SpringerOpen, v. 8, n. 1, p. 1–41, 2021.

BAGUI, S. S.; MINK, D.; BAGUI, S. C.; SUBRAMANIAM, S. Determining resampling ratios using bsmote and svm-smote for identifying rare attacks in imbalanced cybersecurity data. *Computers*, MDPI, v. 12, n. 10, p. 204, 2023.

BIOLCHINI, J.; MIAN, P. G.; NATALI, A. C. C.; TRAVASSOS, G. H. Systematic review in software engineering. *System Engineering and Computer Science Department COPPE/UFRJ, Technical Report ES*, v. 679, n. 05, p. 45, 2005.

BREIMAN, L. *Classification and regression trees*. [S.l.]: Routledge, 2017.

CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, v. 16, p. 321–357, 2002.

CHEN, X.-w.; JEONG, J. C. Enhanced recursive feature elimination. In: IEEE. *Sixth international conference on machine learning and applications (ICMLA 2007)*. [S.l.], 2007. p. 429–435.

CHEN, Z.; ZHOU, L.; YU, W. Adasyn- random forest based intrusion detection model. In: *2021 4th International Conference on Signal Processing and Machine Learning*. [S.l.: s.n.], 2021. p. 152–159.

CHINDOVE, H.; BROWN, D. Adaptive machine learning based network intrusion detection. In: *Proceedings of the International Conference on Artificial Intelligence and its Applications*. [S.l.: s.n.], 2021. p. 1–6.

DASARI, A. K.; BISWAS, S. K.; SANYAL, S.; PURKAYASTHA, B. Intrusion detection system using ensemble learning analytics. In: IEEE. *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*. [S.l.], 2023. p. 1–5.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S. et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

ENGLY, A. H.; LARSEN, A. R.; MENG, W. Evaluation of anomaly-based intrusion detection with combined imbalance correction and feature selection. In: SPRINGER. *International Conference on Network and System Security*. [S.l.], 2020. p. 277–291.

EROL, O. K.; EKSIN, I. A new optimization method: big bang–big crunch. *Advances in Engineering Software*, Elsevier, v. 37, n. 2, p. 106–111, 2006.

FERNÁNDEZ, A.; GARCÍA, S.; GALAR, M.; PRATI, R. C.; KRAWCZYK, B.; HERRERA, F. *Learning from imbalanced data sets*. [S.l.]: Springer, 2018. v. 10.

GAO, Y.; HASEGAWA, H.; YAMAGUCHI, Y.; SHIMADA, H. Malware detection using gradient boosting decision trees with customized log loss function. In: IEEE. *2021 International Conference on Information Networking (ICOIN)*. [S.l.], 2021. p. 273–278.

GAVRYLENKO, S.; VLADISLAV, Z.; KHATSKO, N. Methods for improving the quality of classification on imbalanced data. In: IEEE. *2023 IEEE 4th KhPI Week on Advanced Technology (KhPIWeek)*. [S.l.], 2023. p. 1–5.

GONZALEZ-CUAUTLE, D.; HERNANDEZ-SUAREZ, A.; SANCHEZ-PEREZ, G.; TOSCANO-MEDINA, L. K.; PORTILLO-PORTILLO, J.; OLIVARES-MERCADO, J.; PEREZ-MEANA, H. M.; SANDOVAL-OROZCO, A. L. Synthetic minority oversampling technique for optimizing classification tasks in botnet and intrusion-detection-system datasets. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 10, n. 3, p. 794, 2020.

GOU, W.; ZHANG, H.; ZHANG, R. Multi-classification and tree-based ensemble network for the intrusion detection system in the internet of vehicles. *Sensors*, MDPI, v. 23, n. 21, p. 8788, 2023.

GU, H.; LAI, Y.; WANG, Y.; LIU, J.; SUN, M.; MAO, B. Deids: a novel intrusion detection system for industrial control systems. *Neural Computing and Applications*, Springer, p. 1–19, 2022.

GUYON, I.; WESTON, J.; BARNHILL, S.; VAPNIK, V. Gene selection for cancer classification using support vector machines. *Machine learning*, Springer, v. 46, n. 1, p. 389–422, 2002.

HAIXIANG, G.; YIJING, L.; SHANG, J.; MINGYUN, G.; YUANYUE, H.; BING, G. Learning from class-imbalanced data: Review of methods and applications. *Expert systems with applications*, Elsevier, v. 73, p. 220–239, 2017.

HAN, H.; WANG, W.-Y.; MAO, B.-H. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: SPRINGER. *International conference on intelligent computing*. [S.l.], 2005. p. 878–887.

HE, H.; BAI, Y.; GARCIA, E. A.; LI, S. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In: IEEE. *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. [S.l.], 2008. p. 1322–1328.

HINDY, H.; BAYNE, E.; BURES, M.; ATKINSON, R.; TACHTATZIS, C.; BELLEKENS, X. Machine learning based iot intrusion detection system: An mqtt case study (mqtt-iot-ids2020 dataset). In: SPRINGER. *International networking conference*. [S.l.], 2020. p. 73–84.

HOSMER, D. W.; TABER, S.; LEMESHOW, S. The importance of assessing the fit of logistic regression models: a case study. *American journal of public health*, American Public Health Association, v. 81, n. 12, p. 1630–1635, 1991.

HUANG, J.; CAI, Y.; XU, X. A filter approach to feature selection based on mutual information. In: IEEE. *2006 5th IEEE International Conference on Cognitive Informatics*. [S.l.], 2006. v. 1, p. 84–89.

IMRAN, M.; KHAN, S.; HLAVACS, H.; KHAN, F. A.; ANWAR, S. Intrusion detection in networks using cuckoo search optimization. *Soft Computing*, Springer, v. 26, n. 20, p. 10651–10663, 2022.

JANIESCH, C.; ZSCHECH, P.; HEINRICH, K. Machine learning and deep learning. *Electronic Markets*, Springer, v. 31, n. 3, p. 685–695, 2021.

JIANPING, Z.; INDERJEET, M. Knn approach to unbalanced data distributions: A case study involving information extraction. In: *Proceeding of International Conference on Machine Learning (ICML 2003)*. [S.l.: s.n.], 2003. v. 8.

JIMOH, I. A.; ISMAILA, I.; OLALERE, M. Enhanced decision tree-j48 with smote machine learning algorithm for effective botnet detection in imbalance dataset. In: IEEE. *2019 15th International Conference on Electronics, Computer and Computation (ICECCO)*. [S.l.], 2019. p. 1–8.

JIN, K.; ZHANG, L.; ZHANG, Y.; SUN, D.; ZHENG, X. A network traffic intrusion detection method for industrial control systems based on deep learning. *Electronics*, MDPI, v. 12, n. 20, p. 4329, 2023.

JUI, T. T.; HOQ, M. N.; MAJUMDAR, S.; HOSSAIN, M. S. Feature reduction through data preprocessing for intrusion detection in iot networks. In: *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. [S.l.: s.n.], 2021. p. 41–50.

KARATAS, G.; DEMIR, O.; SAHINGOZ, O. K. Increasing the performance of machine learning-based idss on an imbalanced and up-to-date dataset. *IEEE Access*, IEEE, v. 8, p. 32150–32162, 2020.

KHRAISAT, A.; GONDAL, I.; VAMPLEW, P.; KAMRUZZAMAN, J. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, Springer, v. 2, n. 1, p. 1–22, 2019.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. Citeseer, 2007.

KRASKOV, A.; STÖGBAUER, H.; GRASSBERGER, P. Estimating mutual information. *Physical review E*, APS, v. 69, n. 6, p. 066138, 2004.

KUBAT, M.; MATWIN, S. et al. Addressing the curse of imbalanced training sets: one-sided selection. In: CITESEER. *Icml*. [S.l.], 1997. v. 97, n. 1, p. 179.

KUHN, M.; JOHNSON, K. et al. *Applied predictive modeling*. [S.l.]: Springer, 2013. v. 26.

LEE, B.-S.; KIM, J.-W.; CHOI, M.-J. Experimental comparison of hybrid sampling methods for an efficient nids. In: IEEE. *2022 23rd Asia-Pacific Network Operations and Management Symposium (APNOMS)*. [S.l.], 2022. p. 1–4.

LOPES, A. L. M.; FRACOLLI, L. A. Revisão sistemática de literatura e metassíntese qualitativa: considerações sobre sua aplicação na pesquisa em enfermagem. *Texto & Contexto-Enfermagem*, SciELO Brasil, v. 17, n. 4, p. 771–778, 2008.

LU, T.; HUANG, Y.; ZHAO, W.; ZHANG, J. The metering automation system based intrusion detection using random forest classifier with smote+ enn. In: IEEE. *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*. [S.l.], 2019. p. 370–374.

MADWANNA, Y.; B., A.; R., R. A.; R., S. H. Yars-ids: A novel ids for multi-class classification. In: *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*. [S.l.: s.n.], 2023. p. 1–6.

MARTINS, A. I.; QUEIRÓS, A.; ROCHA, N. P.; SANTOS, B. S. Avaliação de usabilidade: uma revisão sistemática da literatura. *Revista Ibérica de Sistemas e Tecnologias de Informação*, Associação Ibérica de Sistemas e Tecnologias de Informacao, n. 11, p. 31, 2013.

MOHAMMADI, S.; BABAGOLI, M. A novel hybrid hunger games algorithm for intrusion detection systems based on nonlinear regression modeling. *International Journal of Information Security*, Springer, p. 1–19, 2023.

MOHSENI, M.; TANHA, J. A density-based undersampling approach to intrusion detection. In: IEEE. *2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA)*. [S.l.], 2021. p. 1–7.

MOUITI, M.; ELHARIRI, A.; HABIBI, O.; LAZAAR, M. Toward improving internet of things (iot) networks security using machine learning based intrusion detection system. In: *2023 International Conference on Digital Age Technological Advances for Sustainable Development (ICDATA)*. [S.l.: s.n.], 2023. p. 46–51.

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

NAGPAL, M.; KAUSHAL, M.; SHARMA, A. A feature reduced intrusion detection system with optimized svm using big bang big crunch optimization. *Wireless Personal Communications*, Springer, v. 122, n. 2, p. 1939–1965, 2022.

NAQA, I. E.; MURPHY, M. J. What is machine learning? In: _____. *Machine Learning in Radiation Oncology: Theory and Applications*. Cham: Springer International Publishing, 2015. p. 3–11. ISBN 978-3-319-18305-3. Disponível em: <https://doi.org/10.1007/978-3-319-18305-3_1>.

NGUYEN, H. M.; COOPER, E. W.; KAMEI, K. Borderline over-sampling for imbalanced data classification. In: IEEE SMC HIROSHIMA CHAPTER. *Proceedings: Fifth International Workshop on Computational Intelligence & Applicantions*. [S.l.], 2009. v. 2009, n. 1, p. 24–29.

PAN, L.; XIE, X. Network intrusion detection model based on pca+ adasyn and xgboost. In: *Proceedings of the 2020 3rd International Conference on E-Business, Information Management and Computer Science*. [S.l.: s.n.], 2020. p. 44–48.

PANG, Y.; CHEN, Z.; PENG, L.; MA, K.; ZHAO, C.; JI, K. A signature-based assistant random oversampling method for malware detection. In: IEEE. *2019 18th IEEE International conference on trust, security and privacy in computing and communications/13th IEEE international conference on big data science and engineering (TrustCom/BigDataSE)*. [S.l.], 2019. p. 256–263.

PAWLICKI, M.; CHORAŚ, M.; KOZIK, R.; HOŁUBOWICZ, W. On the impact of network data balancing in cybersecurity applications. In: SPRINGER. *International Conference on Computational Science*. [S.l.], 2020. p. 196–210.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PHETLASY, S.; OHZAHATA, S.; WU, C.; KATO, T. Applying smote for a sequential classifiers combination method to improve the performance of intrusion detection system. In: IEEE. *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*. [S.l.], 2019. p. 255–258.

PRAYOGO, R. D.; KARIMAH, S. A. Optimization of phishing website classification based on synthetic minority oversampling technique and feature selection. In: IEEE. *2020 International Workshop on Big Data and Information Security (IWBIS)*. [S.l.], 2020. p. 121–126.

PRISTYANTO, Y.; DAHLAN, A. Hybrid resampling for imbalanced class handling on web phishing classification dataset. In: IEEE. *2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*. [S.l.], 2019. p. 401–406.

RAHMA, F.; RACHMADI, R. F.; PRATOMO, B. A.; PURNOMO, M. H. Assessing the effectiveness of oversampling and undersampling techniques for intrusion detection on an imbalanced dataset. In: IEEE. *2023 IEEE Industrial Electronics and Applications Conference (IEACon)*. [S.l.], 2023. p. 92–97.

RANI, M. et al. Effective network intrusion detection by addressing class imbalance with deep neural networks multimedia tools and applications. *Multimedia Tools and Applications*, Springer, v. 81, n. 6, p. 8499–8518, 2022.

RASHID, A.; SIDDIQUE, M. J.; AHMED, S. M. Machine and deep learning based comparative analysis using hybrid approaches for intrusion detection system. In: IEEE. *2020 3rd International Conference on Advancements in Computational Sciences (ICACS)*. [S.l.], 2020. p. 1–9.

SAPRE, S.; ISLAM, K.; AHMADI, P. A comprehensive data sampling analysis applied to the classification of rare iot network intrusion types. In: IEEE. *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. [S.l.], 2021. p. 1–2.

SETH, S.; SINGH, G.; CHAHAL, K. K. A novel time efficient learning-based approach for smart intrusion detection system. *Journal of Big Data*, SpringerOpen, v. 8, n. 1, p. 1–28, 2021.

SHARAFALDIN, I.; LASHKARI, A. H.; GHORBANI, A. A. A detailed analysis of the cicids2017 data set. In: SPRINGER. *Information Systems Security and Privacy: 4th International Conference, ICISSP 2018, Funchal-Madeira, Portugal, January 22-24, 2018, Revised Selected Papers 4*. [S.l.], 2019. p. 172–188.

SOKOLOVA, M.; JAPKOWICZ, N.; SZPAKOWICZ, S. Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation. In: SPRINGER. *Australasian joint conference on artificial intelligence*. [S.l.], 2006. p. 1015–1021.

SOUZA, D. da S.; DIAS, J.; SANTOS, K. S. "não ligue o computador": a computação desplugada como estratégia metodológica para o desenvolvimento do pensamento computacional na educação básica–uma revisão sistemática da literatura. *RENOTE*, v. 17, n. 3, p. 427–436, 2019.

SUN, L.; ZHOU, Y.; WANG, Y.; ZHU, C.; ZHANG, W. The effective methods for intrusion detection with limited network attack data: multi-task learning and oversampling. *IEEE Access*, IEEE, v. 8, p. 185384–185398, 2020.

SUN, Y.; QUE, H.; CAI, Q.; ZHAO, J.; LI, J.; KONG, Z.; WANG, S. Borderline smote algorithm and feature selection-based network anomalies detection strategy. *Energies*, MDPI, v. 15, n. 13, p. 4751, 2022.

SURYA, V.; SELVAM, M. M. An effective machine learning approach for lot intrusion detection system based on smote. In: IEEE. *2022 6th International Conference on Electronics, Communication and Aerospace Technology*. [S.l.], 2022. p. 905–911.

SZCZEPANSKI, M.; KOMISAREK, M.; PAWLICKI, M.; KOZIK, R.; CHORAŚ, M. The proposition of balanced and explainable surrogate method for network intrusion detection in streamed real difficult data. In: SPRINGER. *International Conference on Computational Collective Intelligence*. [S.l.], 2021. p. 241–252.

TAN, X.; SU, S.; HUANG, Z.; GUO, X.; ZUO, Z.; SUN, X.; LI, L. Wireless sensor networks intrusion detection based on smote and the random forest algorithm. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 19, n. 1, p. 203, 2019.

TARAWNEH, A. S.; HASSANAT, A. B.; ALMOHAMMADI, K.; CHETVERIKOV, D.; BELLINGER, C. Smotefuna: Synthetic minority over-sampling technique based on furthest neighbour algorithm. *IEEE Access*, IEEE, v. 8, p. 59069–59082, 2020.

TAUSCHER, Z.; JIANG, Y.; ZHANG, K.; WANG, J.; SONG, H. Learning to detect: A data-driven approach for network intrusion detection. In: *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. [S.l.: s.n.], 2021. p. 1–6.

TOMEK, I. Two modifications of cnn. 1976.

VARUNRAM, T.; SHIVAPRASAD, M.; AISHWARYA, K.; BALRAJ, A.; SAVISH, S.; ULLAS, S. Analysis of different dimensionality reduction techniques and machine learning algorithms for an intrusion detection system. In: IEEE. *2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA)*. [S.l.], 2021. p. 237–242.

VERMA, P.; BRESLIN, J. G.; O'SHEA, D.; MEHTA, N.; BHAROT, N.; VIDYARTHI, A. Leveraging gametic heredity in oversampling techniques to handle class imbalance for efficient cyberthreat detection in iiot. *IEEE Transactions on Consumer Electronics*, IEEE, 2023.

WANG, J.-H.; SEPTIAN, T. W. Combining oversampling with recurrent neural networks for intrusion detection. In: SPRINGER. *International Conference on Database Systems for Advanced Applications*. [S.l.], 2021. p. 305–320.

WANG, K.; JI, S.; DENG, L.; LIU, Y. Intrusion detection method based on rb-smote and double attention. In: IEEE. *2023 4th International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*. [S.l.], 2023. p. 367–373.

WANG, Z.; JIANG, D.; HUO, L.; YANG, W. An efficient network intrusion detection approach based on deep learning. *Wireless Networks*, Springer, p. 1–14, 2021.

WILSON, D. L. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE, n. 3, p. 408–421, 1972.

XIANG, Z.; XU, Y.; TANG, Z. How does oversampling affect the performance of classification algorithms? In: IEEE. *2023 IEEE Symposium on Computers and Communications (ISCC)*. [S.l.], 2023. p. 1001–1006.

ZHANG, J.; ZHANG, Y.; LI, K. A network intrusion detection model based on the combination of relieff and borderline-smote. In: *Proceedings of the 2020 4th High Performance Computing and Cluster Technologies Conference & 2020 3rd International Conference on Big Data and Artificial Intelligence*. [S.l.: s.n.], 2020. p. 199–203.