Detecção Inteligente de Trojans com SHAP: Comparativo de Algoritmos de Aprendizado de Máquina e Capacidade de Inferência

Dhiogo Pereira Santos

Universidade Federal de Uberlândia
Faculdade de Computação
Bacharelado em Sistemas de Informação

2025

Dhiogo Pereira Santos

Detecção Inteligente de Trojans com SHAP: Comparativo de Algoritmos de Aprendizado de Máquina e Capacidade de Inferência

Trabalho de Conclusão de Curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Área de concentração: Sistemas de Informação

Orientador: Prof. Dr. Diego Nunes Molinos

2025

Este trabalho é dedicado, principalmente, aos meus pais, Ana Maria (in memoriam) e Levindo Júnior, e à minha irmã, Vanessa Martins, que tiveram um papel vital para que eu alcançasse esta conquista.

Agradecimentos

Agradeço primeiramente a Deus, por me conceder sabedoria, saúde e perseverança ao longo desta trajetória acadêmica.

À minha mãe, Ana Maria (in memoriam) e aos meus avós (in memoriam), cujo legado de amor e força continua presente em minha vida.

Ao meu pai, Levindo Júnior, por sua dedicação incansável e exemplo de resiliência, que foram fundamentais para a conclusão desta etapa em minha vida.

À minha irmã, Vanessa Martins, pela constante presença, incentivo e apoio, especialmente nos momentos mais desafiadores.

À minha namorada, Laura Freitas, por seu companheirismo, paciência e apoio incondicional ao longo desta jornada. Sua presença foi essencial para que eu seguisse no propósito.

Agradeço também às minhas tias e a todos os membros da minha família, cujo suporte afetivo contribuiu significativamente para esta conquista.

Aos amigos que estiveram ao meu lado ao longo do curso, compartilhando experiências, desafios e aprendizados, expresso minha sincera gratidão.

Ao meu orientador Diego Nunes, agradeço pela orientação fenomenal e pelos ensinamentos durante todo o processo de confecção desta monografia.

A todos que, de alguma forma, contribuíram para esta etapa da minha vida, deixo meu mais profundo agradecimento.

"A vida se encolhe ou se expande na proporção da coragem que se tem."
(Anaïs Nin)

Resumo

Este trabalho tem o objetivo de investigar a aplicação de algoritmos de aprendizado de máquina na detecção de *malwares* do tipo *trojan*, a partir da análise de dados de memória, utilizando a metodologia de análise e treinamento de modelo descrita em (LEONEL; MOLINOS; MIANI, 2024). Utilizando o conjunto de dados CIC-MalMem-2022 *(Canadian Institute for Cybersecurity (2022))*, foram empregados quatro algoritmos supervisionados, são eles: (a) Árvore de Decisão, (b) Rede Neural Artificial, (c) *Naive Bayes* e (d) *Support Vector Machine*, com o objetivo de comparar seus desempenhos em termos de acurácia, interpretabilidade e eficiência computacional. A fim de promover maior transparência e robustez aos modelos, foi incorporada a técnica SHAP (*SHapley Additive exPlanations*), permitindo a análise da importância individual de cada atributo e a subsequente seleção do conjunto das características. Além da avaliação tradicional por métricas estatísticas como acurácia, precisão, sensibilidade e F1-score, este trabalho realiza uma análise aprofundada do tempo de inferência dos modelos em cenários de execução unitária e em blocos. Os resultados obtidos indicam que a aplicação combinada de técnicas de aprendizado de máquina e inteligência artificial explicável é eficaz não apenas na detecção de *trojans* com alta precisão, mas também na construção de classificadores mais enxutos, interpretáveis e adaptáveis a contextos com restrições de tempo e recursos.

Palavras-chave: Aprendizado de máquina, trojan, malware, detecção de ameaças, classificação supervisionada.

Lista de ilustrações

Figura 1 – Fluxograma 24
Figura 2 – Curva de Aprendizado - Rede Neural 32
Figura 3 – Matriz de Confusão - Rede Neural 32
Figura 4 – Curva de Aprendizado - Árvore de Decisão (Acurácia) 33
Figura 5 – Curva de Aprendizado - Árvore de Decisão (Loss) 34
Figura 6 – Matriz de Confusão - Árvore de Decisão 34
Figura 7 – Curva de Aprendizado - Naive Bayes (Acurácia) 35
Figura 8 – Curva de Aprendizado - Naive Bayes (Loss) 35
Figura 9 – Matriz de Confusão - Naive Bayes 36
Figura 10 – Curva de Aprendizado - SVM (Acurácia) 37
Figura 11 – Curva de Aprendizado - SVM (Loss) 37
Figura 12 – Matriz de Confusão - SVM 38
Figura 13 – Árvore de Decisão no processo de seleção das características 39
Figura 14 – Valores dos atributos usando Árvore de Decisão 41
Figura 15 – Curva de Aprendizado - Rede Neural 42
Figura 16 – Matriz de Confusão - Rede Neural 43
Figura 17 – Curva de Aprendizado - Árvore de Decisão (Acurácia) 44
Figura 18 – Curva de Aprendizado - Árvore de Decisão (Loss) 44
Figura 19 – Matriz de Confusão - Árvore de Decisão 45
Figura 20 – Curva de Aprendizado - Naive Bayes (Acurácia) 46
Figura 21 – Curva de Aprendizado - Naive Bayes (Loss) 46
Figura 22 – Matriz de Confusão - Naive Bayes 47
Figura 23 – Curva de Aprendizado - SVM (Acurácia) 48
Figura 24 – Curva de Aprendizado - SVM (Loss) 48
Figura 25 – Matriz de Confusão - SVM 49

Lista de tabelas

Tabela 1 – Comparação entre trabalhos correlatos e o presente estudo 22
Tabela 2 – Especificações da máquina utilizada nos experimentos 25
Tabela 3 – Atributos do *dataset* CIC-MalMem-2022 26
Tabela 4 – Parâmetros dos modelos de classificação utilizados 27
Tabela 5 – Fórmulas das métricas de avaliação 29
Tabela 6 – Média das acurácias dos algoritmos selecionados 38
Tabela 7 – Principais atributos segundo os valores SHAP – Árvore de Decisão . . 40
Tabela 8 – Tempo de inferência para amostras da classe *Trojan* (em segundos) . . 50
Tabela 9 – Tempo de inferência para amostras da classe *Benigno* (em segundos) . 50

Lista de siglas

CNN Rede Neural Convolucional - *Convolutional Neural Network*

DNN Rede Neural Profunda - *Deep Neural Network*

DT Árvore de Decisão - *Decision Tree*

IA Inteligência Artificial - *Artificial Intelligence*

KNN K-vizinhos mais próximos - *K-Nearest Neighbors*

ML Aprendizado de Máquina - *Machine Learning*

NB Naive Bayes

RF Random Forest

RAM Memória de Acesso Aleatório - *Random Access Memory*

SHAP Valores de Shapley Aditivos para Explicações - *SHapley Additive exPlanations*

SVM Máquina de Vetores de Suporte - *Support Vector Machine*

XAI Inteligência Artificial Explicável - *Explainable Artificial Intelligence*

Sumário

1 INTRODUÇÃO 12
1.1 Motivação 14
1.2 Hipótese 14
1.3 Objetivos 15
1.4 Divisão da Monografia 15

2 FUNDAMENTAÇÃO TEÓRICA 16
2.1 Malware 16
2.1.1 Principais categorias 16
2.1.2 Trojans 17
2.2 Inteligência Artifical na Detecção de Trojan 17
2.3 Soluções Propostas para Superar Desafios na Detecção de Trojans 18
2.3.1 Inteligência Artificial Explicável - Inteligência Artificial Explicável (XAI) 18
2.3.2 Seleção de Características 18
2.3.3 Tempo de Inferência 19

3 TRABALHOS RELACIONADOS 20
3.1 Síntese dos Trabalhos Correlatos 21
3.2 Considerações com Trabalhos Anteriores 22

4 MÉTODO 23
4.1 Percepção da Problemática 24
4.2 Materiais 24
4.3 Instrumentação 25
4.3.1 Dataset CIC-MalMem-2022 25
4.3.2 Algoritmos de Inteligência Artificial 26
4.3.3 Parâmetros dos Modelos 27

4.4 Treinamento dos Modelos . . . 28
4.5 Seleção de Características . . . 28
4.6 Métricas de Avaliação . . . 29

5 EXPERIMENTAÇÃO E SÍNTESE DE RESULTADOS . . . 31
5.1 Avaliação do Baseline . . . 31
5.1.1 Rede Neural . . . 31
5.1.2 Árvore de Decisão . . . 33
5.1.3 Naive Bayes . . . 34
5.1.4 Support Vector Machine . . . 36
5.1.5 Síntese do Baseline . . . 38
5.2 Seleção de Características . . . 39
5.2.1 Principais Características com Árvore de Decisão . . . 39
5.2.2 *SHAP - SHapley Additive exPlanations* . . . 40
5.3 (Re)Avaliação dos Modelos . . . 41
5.3.1 Rede Neural . . . 42
5.3.2 Árvore de Decisão . . . 43
5.3.3 Naive Bayes . . . 45
5.3.4 Support Vector Machine . . . 47
5.4 Avaliação do Tempo de Inferência . . . 49

6 CONCLUSÃO . . . 51
6.1 Trabalhos Futuros . . . 52
6.2 Artefatos de Software . . . 53

REFERÊNCIAS . . . 54

Capítulo 1

Introdução

Os sistemas computacionais são amplamente utilizados em diversos aspectos da vida moderna, e essa crescente disseminação os torna um campo propenso a várias formas de vulnerabilidades. Um ataque a um sistema vital pode resultar em consequências que vão desde perdas financeiras até a exposição de dados sensíveis, danos ambientais e até mesmo colocar vidas humanas em risco.

Recentemente, vários casos notórios de invasões de sistemas e roubo de informações vieram à tona. Um exemplo notável foi o ataque à Riot Games (Digital (2023)), onde o código-fonte de um de seus produtos foi comprometido, afetando o lançamento de novos conteúdos para o popular jogo *"League of Legends"*. Outro caso de ataque também ocorreu na empresa Microsoft (GLOBO, 2022), onde a marca teve uma de suas contas comprometida por *hackers*. Os criminosos tiveram acesso a parte do código fonte de produtos da empresa.

Os ataques à Riot Games e à Microsoft exemplificados refletem a crescente necessidade de estratégias robustas de segurança da informação, especialmente em um mundo onde o valor dos dados digitais e a integridade dos sistemas são de importância crítica. A violação de segurança dessas empresas não apenas expôs dados sensíveis, mas também destacou a vulnerabilidade de infraestruturas críticas a ataques sofisticados (DIGITAL, 2023; GLOBO, 2022). Tais incidentes demonstram a urgência de incorporar medidas avançadas de segurança, que podem identificar e mitigar potenciais ameaças antes que causem danos significativos. A segurança da informação não é mais uma opção, mas uma necessidade imperativa para proteger propriedades intelectuais e manter a confiança dos usuários, reforçando a segurança cibernética no desenvolvimento e manutenção de *software*.

Esses incidentes refletem uma realidade preocupante: até mesmo grandes corporações, que teoricamente contam com políticas de segurança robustas, estão suscetíveis a invasões sofisticadas. Isso reforça a necessidade de soluções mais avançadas e adaptativas no campo da segurança da informação, especialmente em um cenário onde a integridade de dados e a proteção de propriedade intelectual são cruciais. A dependência de abordagens

tradicionais, como antivírus baseados em assinaturas ou *firewalls* convencionais, mostra-se cada vez mais limitada diante da evolução constante das ameaças cibernéticas (LEONEL; MOLINOS; MIANI, 2024).

Nesse cenário, os *malwares* possuem a finalidade de comprometer a integridade, a confidencialidade ou a disponibilidade de sistemas e dados, representando uma das ameaças mais prevalentes e danosas. Entre as diversas categorias de *malwares*, destaca-se o **Cavalo de Troia**, ou simplesmente *Trojan*, devido à sua capacidade de dissimulação e à ampla gama de funcionalidades maliciosas que pode incorporar (LIAKOS et al., 2019).

Conforme definido por Mazeika et al. (2023), os *Trojans* são programas que se apresentam como legítimos e inofensivos, mas que, ao serem executados, permitem o acesso remoto ao sistema da vítima, frequentemente sem qualquer sinal visível de sua presença. Essa camuflagem facilita a infiltração em sistemas protegidos e torna sua detecção particularmente desafiadora.

No que tange à detecção e mitigação de *trojans*, diferentemente dos métodos tradicionais baseados em assinaturas ou amostras comprometidas, as abordagens de Inteligência Artificial (IA) conseguem identificar comportamentos anômalos e padrões sutis nos próprios parâmetros do modelo ou em suas respostas à entradas controladas (MAZEIKA et al., 2023).

A IA, especialmente com o uso de aprendizado de máquina, tem se mostrado promissora na superação dessas limitações. Em vez de depender de *inputs* com gatilhos conhecidos, que muitas vezes não estão disponíveis, os métodos baseados em IA analisam propriedades internas dos modelos para inferir a presença de comportamento oculto. De acordo com Mazeika et al. (2023), mesmo diante de *trojans* evasivos projetados para enganar detectores tradicionais, algoritmos mais sofisticados conseguiram manter alta taxa de detecção, alcançando acurácia próxima de 100%. Isso revela o potencial da IA não apenas para identificar ameaças ocultas, mas também para desenvolver mecanismos de defesa resilientes e generalizáveis a diferentes arquiteturas e domínios de aplicação.

Apesar dos avanços notáveis proporcionados pela aplicação de técnicas de Aprendizado de Máquina (ML) na detecção de *trojans*, é importante reconhecer que esses métodos ainda enfrentam limitações significativas, especialmente no que diz respeito à racionalização dos modelos e à seleção eficiente de atributos. Os resultados ótimos observados em Mazeika et al. (2023), muitas vezes dependem de conjuntos de dados altamente estruturados e de recursos computacionais intensivos, o que pode não refletir a complexidade dos cenários reais de implantação. Além disso, modelos sofisticados tendem a ser menos interpretáveis, o que dificulta a compreensão das decisões tomadas e a validação dos atributos mais relevantes para a classificação.

Assim, embora a IA represente uma poderosa aliada na defesa contra ameaças cibernéticas, sua eficácia está condicionada a uma engenharia cuidadosa do conjunto de atributos e à transparência dos modelos, sob risco de se tornarem caixas-pretas com decisões difíceis

de auditar ou generalizar em ambientes adversos.

Ante o exposto, este trabalho propõe o estudo e a avaliação de modelos de aprendizado de máquina aliados a técnicas de inteligência artificial explicável (XAI) com o objetivo de promover classificadores robustos, interpretáveis e eficazes na detecção de *trojans*. Além da avaliação preditiva convencional, baseada em métricas como acurácia, precisão, recall e F1-Score, este estudo fará uso de explicabilidade com o Valores de Shapley Aditivos para Explicações (SHAP) sobre o modelo de árvore de decisão para mensurar a contribuição de cada atributo na decisão dos classificadores. Essa abordagem visa garantir transparência e confiabilidade no processo de detecção, permitindo que analistas compreendam os padrões que indicam a presença de comportamento trojanizado, contribuindo para a validação do modelo em cenários reais.

1.1 Motivação

A crescente sofisticação das ameaças cibernéticas, especialmente aquelas relacionadas a *malwares* discretos como os *trojans*, evidencia um cenário no qual os mecanismos tradicionais de defesa se mostram insuficientes (MAZEIKA et al., 2023). A capacidade dos *trojans* de se disfarçarem como aplicações legítimas e operarem de maneira silenciosa compromete diretamente a segurança de sistemas computacionais e vem afetando desde usuários comuns até infraestruturas críticas de grandes corporações.

Nesse contexto, técnicas de aprendizado de máquina aplicadas à segurança cibernética têm se destacado por sua capacidade de detectar padrões comportamentais maliciosos, mesmo quando estes não apresentam características explícitas (LEONEL; MOLINOS; MIANI, 2024). No entanto, a eficácia desses modelos muitas vezes esbarra em desafios como a complexidade na interpretação dos resultados, o risco de sobreajuste e a dependência de grandes volumes de dados rotulados e estruturados.

A adoção de abordagens de XAI surge como uma alternativa promissora para mitigar tais limitações, conferindo maior transparência aos modelos preditivos. Assim, há uma motivação clara para investigar soluções que aliem alto desempenho preditivo à capacidade de interpretação, viabilizando sistemas de detecção de ameaças mais confiáveis, robustos e adequados à realidade dinâmica dos ambientes computacionais contemporâneos.

O presente trabalho é feito para explorar e comprovar essa vantagem, juntamente combinado com as técnicas explicativas.

1.2 Hipótese

A racionalização de modelos de aprendizado de máquina, por meio da seleção criteriosa de atributos relevantes, contribui para a redução do conjunto de características, o

que resulta em maior eficiência computacional e melhoria da capacidade operacional dos classificadores, sem comprometer o desempenho na detecção de ameaças como *trojans*.

1.3 Objetivos

Este trabalho tem como objetivo principal avaliar a eficácia de algoritmos de aprendizado de máquina na detecção de *trojans*, com ênfase na otimização do conjunto de atributos relevantes e na interpretabilidade dos modelos, utilizando o método SHAP como recurso explicativo.

Para atingir esse objetivo principal, são estabelecidos os seguintes objetivos secundários:

- ❏ Analisar os principais algoritmos de aprendizado de máquina utilizados para detecção de *trojans*;
- ❏ Tratar e selecionar atributos relevantes a partir de um conjunto de dados de memória CIC-MalMem-2022;
- ❏ Treinar modelos de detecção de *trojans* utilizando diferentes algoritmos de aprendizado de máquina;
- ❏ Avaliar o desempenho dos modelos quanto à acurácia, sensibilidade, especificidade e tempo de inferência;

1.4 Divisão da Monografia

Esta monografia está estruturada em seis capítulos. O **Capítulo 1** introduz o tema da pesquisa, contextualizando a problemática abordada, sua relevância científica e prática, além de apresentar os objetivos do estudo e a hipótese formulada. O **Capítulo 2** é dedicado à fundamentação teórica, reunindo os principais conceitos sobre *malwares*, com ênfase nos *trojans*, algoritmos de aprendizado de máquina e técnicas de interpretabilidade, destacando-se o método SHAP. No **Capítulo 3**, encontram-se as sínteses de alguns dos diversos trabalhos relacionados. No **Capítulo 4**, são detalhados os procedimentos metodológicos adotados, incluindo as etapas de preparação dos dados, seleção e parametrização dos algoritmos, aplicação de métodos para redução de atributos e definição das métricas de avaliação. O **Capítulo 5** apresenta os resultados obtidos nos experimentos, acompanhados de análises comparativas e interpretações críticas à luz dos objetivos propostos. Por fim, o **Capítulo 6** reúne as conclusões do trabalho, ressaltando as principais contribuições, as limitações encontradas e recomendações para investigações futuras.

Capítulo 2

Fundamentação Teórica

2.1 Malware

Com o avanço das tecnologias e a crescente dependência de sistemas digitais, o número de ameaças virtuais aumentou imensamente. Entre essas ameaças, os *malwares* se destacam por sua variedade, complexidade e impactos negativos. O termo *malware* é uma abreviação de *malicious software*. São programas desenvolvidos com o objetivo de causar danos, roubar informações, assumir o controle de sistemas ou comprometer o funcionamento de dispositivos e redes (KIM; SOLOMON, 2016). De acordo com AVTEST Institute (2022), mais de 1.3 bilhão de amostras ativas foram registradas em 2022. Esses *softwares* maliciosos podem ser separados em diversas categorias, como vírus, *worms*, *ransomwares*, *spywares* e *trojans*, cada um com características e métodos de ataque específicos (GANDOTRA; BANSAL; SOFAT, 2014).

2.1.1 Principais categorias

- **Vírus:** Dependem de um arquivo hospedeiro para se propagar e executar suas instruções maliciosas (COHEN, 1987).
- **Worms:** São programas autopropagáveis que se disseminam por redes sem a necessidade de intervenção humana (ZHANG; WANG, 2018).
- **Trojans:** Apresentam-se como *softwares* legítimos, mas executam ações maliciosas de forma dissimulada (DENER; OK; ORMAN, 2022).
- **Ransomware:** Realiza a criptografia de dados do sistema da vítima com o intuito de extorsão financeira (MARTIN; KINROSS, 2019).

2.1.2 Trojans

Trojans são uma classe de ameaças cibernéticas que se disfarçam como *softwares* legítimos ou inofensivos, mas carregam intencionalmente funções maliciosas com o objetivo de comprometer a segurança do sistema (HUANG et al., 2020). No contexto de sistemas computacionais, *trojans* podem ser inseridos em diferentes etapas do ciclo de desenvolvimento e operação, desde o *software* até os circuitos integrados (ICs), esses códigos maliciosos podem permanecer latentes por longos períodos, sendo ativados somente sob condições específicas, como determinados padrões de entrada ou estados do sistema (HUANG et al., 2020). Uma vez ativados, podem exfiltrar dados, corromper processos ou oferecer controle remoto a um agente externo, tornando sua detecção particularmente desafiadora, sobretudo quando seu comportamento mimetiza processos legítimos do sistema.

Ainda de acordo com Huang et al. (2020), a detecção eficaz de *trojans* exige abordagens que combinem análise comportamental com métodos de inteligência artificial. Neste contexto, técnicas tradicionais, como análise estática de código ou assinaturas conhecidas, muitas vezes falham ao identificar ameaças mais sofisticadas.

Nesse cenário, algoritmos de aprendizado de máquina têm se mostrado promissores, pois conseguem capturar padrões anômalos em dados brutos, como registros de memória ou traços de consumo de energia, e classificar comportamentos suspeitos mesmo sem um modelo explícito de ameaça. Além disso, o uso de técnicas explicáveis, como SHAP, permite compreender quais atributos mais influenciam na decisão do modelo, contribuindo para a criação de sistemas de detecção mais confiáveis e transparentes. A análise de inferência dos modelos, incluindo seu tempo de resposta, também é crucial em ambientes sensíveis, onde decisões em tempo real são exigidas para mitigar rapidamente possíveis compromissos à segurança (HOSSAIN; ISLAM, 2024; HUANG et al., 2020).

2.2 Inteligência Artifical na Detecção de Trojan

A detecção manual de *trojans* apresenta limitações significativas em termos de escalabilidade e eficiência, principalmente devido ao volume crescente de amostras maliciosas e às técnicas cada vez mais sofisticadas de ofuscação empregadas por agentes maliciosos. Nesse contexto, os métodos baseados em inteligência artificial emergem como solução promissora, oferecendo capacidade de análise automatizada de grandes conjuntos de dados com menor intervenção humana (UCCI; ANIELLO; BALDONI, 2019).

Os sistemas de aprendizado de máquina aplicados à detecção de *trojans* operam através da extração automática de padrões complexos a partir de conjuntos de treinamento representativos. Como demonstrado por Ahmadi et al. (2016), esses modelos são capazes de identificar relações não lineares entre características do código executável, comportamento em tempo de execução e artefatos deixados na memória do sistema. A eficácia do processo depende criticamente da seleção adequada do algoritmo, que deve considerar o

equilíbrio entre precisão e desempenho computacional (MOSLI et al., 2016), adaptando-se às particularidades do cenário de análise.

A pesquisa de Damaševičius et al. (2021) comprovou que essa técnica é particularmente adequada para a detecção de *trojans*, com taxa de falsos positivos abaixo de 0,5% em ambientes controlados, permitindo compensar as limitações individuais de cada modelo através da integração de diferentes perspectivas de análise. A combinação de classificadores especializados em diferentes aspectos do código malicioso tem demonstrado resultados significativamente melhores quando comparada ao uso de abordagens que empregam apenas uma única técnica de análise (HEMALATHA et al., 2021).

2.3 Soluções Propostas para Superar Desafios na Detecção de Trojans

Enquanto abordagens tradicionais baseadas em assinaturas demonstram boa eficácia na identificação de amostras previamente conhecidas, elas tendem a falhar diante de variantes polimórficas e metamórficas, que alteram sua estrutura para escapar da detecção (DENER; OK; ORMAN, 2022; HOSSAIN; ISLAM, 2024). Alternativamente, métodos de análise dinâmica oferecem maior robustez, mas exigem recursos computacionais elevados e apresentam maior tempo de processamento, o que limita sua aplicação em ambientes com restrições operacionais. Nesse cenário, têm ganhado destaque soluções baseadas em técnicas de aprendizado de máquina, especialmente quando aliadas à XAI, que permitem não apenas detectar padrões complexos de comportamento malicioso, mas também fornecer interpretabilidade quanto às decisões tomadas pelos modelos.

2.3.1 Inteligência Artificial Explicável - XAI

A escolha e uso de técnicas de Inteligência Artificial Explicável (XAI) tem se mostrado essencial para aumentar a transparência dos modelos de detecção baseados em aprendizado de máquina. Como os *trojans*, a cada dia mais, empregam métodos sofisticados para mascarar e dificultar seu comportamento, modelos de *deep learning* tradicionais podem atuar como "caixas pretas", dificultando a interpretação de suas decisões (RUI; GADYATSKAYA, 2024). O XAI permite identificar quais características mais contribuem para a classificação, tornando o processo mais auditável e confiável (GEARHART, 2024).

2.3.2 Seleção de Características

As alterações e escolhas das características são essenciais para melhorar a performance de modelos de detecção de *trojans* com aprendizado de máquina. Utilizar muitas carac-

terísticas pode prejudicar o desempenho do classificador, tornando as etapas de testes e treinamento lentas. Além disso, tem-se o fato de o peso que cada característica representa no modelo, ou seja, o quão importante e decisiva ela se mostra ser durante as etapas de classificação.

Neste trabalho, através da XAI, foi utilizado o SHAP para interpretar a importância de cada característica no modelo. O SHAP permite identificar quais atributos realmente contribuem para a decisão da detecção, ajudando a reduzir a dimensionalidade do conjunto de dados sem comprometer a acurácia.

Esse processo de seleção é especialmente importante no caso dos *trojans*, que costumam usar técnicas de ofuscação e metamorfismo para dificultar a análise (MAZEIKA et al., 2023). Através da utilização do SHAP, se tem uma camada adicional de transparência, permitindo entender o impacto de cada característica, inclusive em amostras adversas ou mutantes.

2.3.3 Tempo de Inferência

Em determinados ambientes, onde a capacidade de resposta em tempo hábil é essencial, modelos excessivamente complexos podem comprometer a viabilidade operacional ao demandarem tempos de execução elevados, o que pode atrasar ações críticas e resultar em consequências indesejadas. Embora métricas estáticas como acurácia, F1-score e recall sejam amplamente utilizadas para avaliar o desempenho preditivo dos modelos, elas não são, por si só, indicativas de eficiência prática. Um modelo com alta acurácia pode, por exemplo, apresentar tempos de inferência demasiadamente longos, tornando-o inadequado para aplicações em tempo real. Nesse contexto, estratégias como a redução de precisão numérica e a eliminação de neurônios redundantes em redes neurais têm sido adotadas para otimizar o equilíbrio entre desempenho e tempo de resposta (CHAUDHARY; MASOOD, 2023). Adicionalmente, a análise de memória volátil combinada com algoritmos leves e bem ajustados tem se mostrado eficaz na detecção de ameaças, alcançando resultados expressivos tanto em termos de acurácia quanto de eficiência computacional (NYHOLM et al., 2022).

Capítulo 3

Trabalhos relacionados

Em Abualhaj e Al-Khatib (2024), é apresentado um método para detecção de Cavalos de Troia (*Trojans*) por meio da análise de dados extraídos da Memória de Acesso Aleatório (RAM). O experimento tem como objetivo identificar comportamentos anormais, cujas características são de atividades maliciosas, utilizando o algoritmo de árvore de decisão como principal classificador. Os experimentos foram realizados com amostras do *dataset* Canadian Institute for Cybersecurity (2022), abrangendo várias famílias de *trojans*. Além da árvore de decisão, outros algoritmos como Naive Bayes (NB), K-vizinhos mais próximos (KNN) e Máquina de Vetores de Suporte (SVM) também foram utilizados. Os modelos foram comparados com base em métricas de desempenho como acurácia, precisão, *recall* e F1-score. Os resultados indicam que o classificador Árvore de Decisão (DT) obteve o melhor desempenho alcançando 99,96% de acurácia, enquanto o NB apresentou o menor desempenho com 98,41%. De todo modo, o estudo evidencia que a análise de memória é uma abordagem altamente eficaz para a detecção de *Trojans*, contribuindo para a melhoria dos mecanismos de defesa contra ameaças avançadas.

Wadkar, Troia e Stamp (2020) propôs uma abordagem para detectar mudanças evolutivas em famílias de *malware*, utilizando o algoritmo SVM. Os autores selecionaram 13 famílias de *malwares* e usaram 55 características extraídas desses arquivos. Durante os experimentos, os modelos SVM foram treinados para identificar diferenças entre características de famílias de *malwares* em períodos consecutivos. A análise dos pesos atribuídos pelo SVM a cada característica revelou pontos de mudança significativos no comportamento das famílias. Por exemplo, foi identificado um pico de alteração no comportamento da família *Zbot*, indicando um grande período de modificação no código. Também, algumas famílias como *DelfInject* mostraram pouca ou nenhuma alteração ao longo do tempo, enquanto outras, como a *Vobfus*, exibiram instabilidade contínua, sugerindo modificações frequentes. A partir dessa abordagem, foi possível mapear a evolução de várias famílias de *malwares*, identificando tanto picos de grandes alterações quanto períodos de estabilidade, com potencial para auxiliar na detecção de mudanças significativas no código de

malwares ao longo do tempo.

No trabalho de Carrier (2021), foi desenvolvido um modelo focado na detecção de *malwares* a partir da análise de memória, com foco em *trojan*, *spyware* e *ransomware*. O mesmo tem como objetivo apresentar um melhor desempenho no tempo de execução. Para isso, os autores utilizaram a ferramenta Volatility Volatility Foundation (2025), e o extrator VolMemLyzer Lashkari et al. (2021), que acrescentou 26 novas características voltadas especificamente à identificação de comportamentos maliciosos ocultos, como injeções de código, módulos ausentes e manipulações de processos. O modelo foi construído com uma abordagem de aprendizado de máquina em duas camadas, combinando diferentes classificadores como DT, Random Forest (RF), SVM, KNN e NB. A primeira camada processa os dados extraídos, e os resultados são refinados por um classificador final, que define se o conteúdo da memória é benigno ou maligno. Os testes demonstraram que a combinação de características específicas com o uso dos vários algoritmos resultou em uma detecção precisa e com tempo de resposta reduzido. Além disso, os autores destacam que o modelo foi implementado em Python. O estudo mostrou potencial para aprimorar métodos tradicionais de detecção, oferecendo um sistema mais leve e adaptado à detecção de ameaças ocultas em ambientes reais.

O estudo de Al-Ghanem et al. (2025) propõe um modelo de detecção de *malwares* usando uma rede neural híbrida, ou seja, a combinação entre Rede Neural Profunda (DNN) e Rede Neural Convolucional (CNN), com foco tanto em classificação binária quanto em múltiplas categorias. Os testes foram feitos com *dataset* (Canadian Institute for Cybersecurity, 2022). O processo começou com o preparo dos dados, passando pelo uso de camadas convolucionais, até chegar à classificação final. Para a parte binária, foi usada a técnica de análise de componentes principais para reduzir os dados e a função *Sigmoid* para classificar entre maligno ou benigno. Já na classificação por categorias, que são os *ransomware*, *trojan* e *spyware*, foi aplicada a técnica de superamostragem sintética para classes em menor volume para balancear os dados e a função *Softmax* para gerar as probabilidades de cada classe. O modelo atingiu 99,5% de acurácia na classificação binária e 97,9% na classificação por múltiplas classes. Além da acurácia, foram avaliadas métricas como precisão, *recall* e F1-*score*, todas com bons resultados. Além disso, a estrutura mostrou uma boa capacidade de lidar com dados desbalanceados.

3.1 Síntese dos Trabalhos Correlatos

A análise dos trabalhos relacionados foi essencial para o aprimoramento da proposta desta monografia, principalmente ao evidenciar que métricas tradicionais como acurácia, precisão e *recall* não são, por si só, suficientes para determinar a eficiência real de um

modelo em ambientes críticos. Essa constatação motivou a inclusão da análise de tempo de inferência como critério complementar na avaliação dos classificadores. Além disso, as limitações observadas na interpretabilidade dos modelos em estudos prévios reforçaram a importância da adoção de técnicas de Inteligência Artificial Explicável (XAI), como o SHAP, possibilitando não apenas um melhor entendimento das decisões do modelo, mas também a identificação e eliminação de atributos redundantes. Dessa forma, a monografia se diferencia ao propor uma abordagem que equilibra desempenho preditivo, interpretabilidade e eficiência computacional, contribuindo para soluções mais robustas e aplicáveis no contexto da detecção de *trojans* em memória.

Trabalho	**Múltiplos Modelos**	**Tempo de Inferência**	**Seleção de características**
Abualhaj et al. (2024)	×		
Wadkar, Troia e Stamp (2020)		×	
Carrier (2021)	×	×	×
Al-Ghanem et al. (2025)			
Presente trabalho	×	×	×

Tabela 1 – Comparação entre trabalhos correlatos e o presente estudo

Fonte: Autor

3.2 Considerações com Trabalhos Anteriores

Este trabalho baseou-se na metodologia de Leonel, Molinos e Miani (2024), que utilizou algoritmos de aprendizado de máquina e técnicas como SHAP para detectar *ransomwares* a partir da análise de memória do dataset CIC-MalMem-2022. A presente pesquisa adaptou essa abordagem para a detecção de *trojans*, reconhecendo que, embora ambos sejam *malwares*, suas diferenças comportamentais podem impactar o desempenho dos modelos preditivos.

Inicialmente, previa-se uma comparação direta com os resultados de (LEONEL; MOLINOS; MIANI, 2024) para investigar um possível enviesamento no *dataset*, ou seja, a tendência de favorecer certas categorias de ataque. No entanto, essa etapa não foi realizada devido a limitações de tempo e à necessidade de focar na implementação e avaliação dos modelos para *trojans*. A análise comparativa permanece como uma linha promissora para trabalhos futuros, especialmente no que diz respeito à **generalização** e à **robustez** do *dataset* frente a diferentes tipos de *malwares*.

Capítulo 4

Método

É oportuno salientar que o protocolo experimental empregado neste trabalho foi integralmente baseado na metodologia proposta por (LEONEL; MOLINOS; MIANI, 2024), a qual contempla a utilização de algoritmos de aprendizado de máquina, a replicação das configurações dos modelos e a aplicação do mesmo conjunto de dados, o *CIC-MalMem-2022*. A única distinção metodológica refere-se à classe de amostras analisadas: enquanto os autores originais concentraram-se na detecção de *ransomwares*, este estudo restringe-se à identificação de instâncias classificadas como *trojans*.

A adoção deliberada de um protocolo experimental previamente consolidado tem como finalidade assegurar a consistência metodológica e, simultaneamente, viabilizar investigações comparativas entre distintas tipologias de *malwares*. Tal estratégia favorece uma análise crítica mais ampla acerca da qualidade, da robustez e da capacidade de generalização do *dataset*, especialmente quando submetido a métodos de inteligência artificial explicável (XAI). Dessa forma, busca-se reduzir potenciais vieses decorrentes de variações no delineamento experimental, conferindo maior rigor e validade aos resultados alcançados.

Este capítulo tem como finalidade apresentar em detalhes o método de pesquisa adotado neste trabalho, destacando como principais contribuições: (i) a descrição sistemática das etapas conduzidas na construção da proposta; e (ii) a delimitação precisa do escopo do estudo, estabelecendo os limites e direcionamentos da investigação.

De acordo com a classificação proposta por Wazlawick (2009) e Marconi e Lakatos (2004), esta pesquisa pode ser caracterizada como aplicada, quantitativa e experimental, pois trata-se de uma investigação, que pode gerar conhecimento voltado à solução de problemas concretos no domínio da segurança cibernética, especialmente na detecção de *trojans* por meio de técnicas de inteligência artificial. Quanto à abordagem, trata-se de uma pesquisa quantitativa, uma vez que se fundamenta na análise estatística dos dados obtidos nos experimentos. Ainda segundo Wazlawick (2009) e Marconi e Lakatos (2004), o estudo assume natureza experimental, pois envolve a manipulação controlada de variáveis com o intuito de verificar o impacto no desempenho dos modelos. Essa tipologia

metodológica visa garantir a objetividade, a reprodutibilidade e a validade dos resultados obtidos ao longo da investigação.

O método praticado neste trabalho encontra-se dividido em cinco macros seções, são elas: (a) Percepção da Problemática, (b) Instrumentação, (c) Treinamento dos Modelos, (d) Seleção das Características e, (e) Análise de Resultados. A Figura 1 ilustra o método deste trabalho.

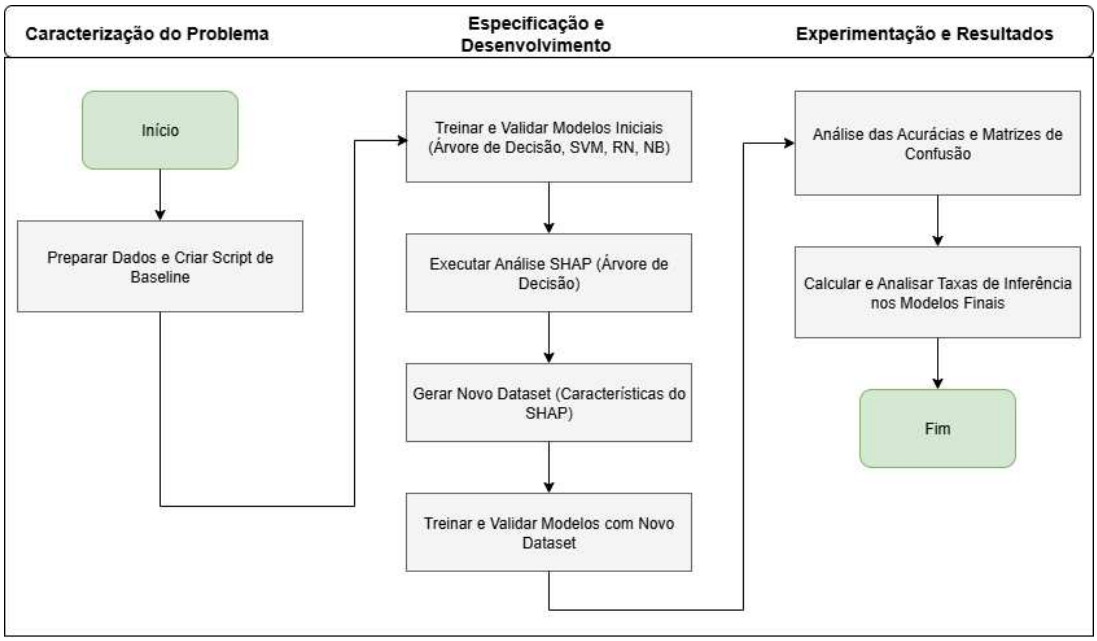

Figura 1 – Fluxograma

4.1 Percepção da Problemática

A detecção de *trojans* representa um desafio crescente na segurança digital, devido à sua natureza discreta e à capacidade de se disfarçar como *software* legítimo. Diferente de *malwares* destrutivos, os *trojans* permanecem ativos por longos períodos sem levantar suspeitas, dificultando sua identificação por métodos tradicionais baseados em assinaturas. Este trabalho propõe o uso de técnicas de aprendizado de máquina aplicadas à análise da memória, visando identificar padrões comportamentais anômalos. Trabalhar com dados da memória permite uma visão detalhada da atividade do sistema, viabilizando a classificação automática de ameaças, mesmo na ausência de sinais evidentes de ataque.

4.2 Materiais

Para a realização dos experimentos deste trabalho, foi utilizada uma única máquina com as seguintes configurações de hardware e software. A Tabela 2 resume os principais componentes e ambiente computacional empregados na execução dos algoritmos e testes:

Tabela 2 – Especificações da máquina utilizada nos experimentos

Componente	**Especificação**
Modelo do Dispositivo	Aspire A515-45 (LAPTOP-T340KQPG)
Processador (CPU)	AMD Ryzen 5 5500U with Radeon Graphics @ 2.10 GHz
Memória RAM	8 GB DDR4 (utilizável: 7,35 GB)
Sistema Operacional	Windows 11 Home Single Language 64 bits (Versão 24H2)
Armazenamento	512GB SSD
Placa de Vídeo (GPU)	Radeon Graphics integrada
Linguagem de Programação	Python 3.10
Bibliotecas Utilizadas	Scikit-learn, Pandas, NumPy, Matplotlib, SHAP
Ambiente de Execução	Jupyter Notebook

4.3 Instrumentação

Nesta seção serão detalhados o conjunto de dados, os algoritmos de inteligência artificial utilizados para os estudo juntamente com as configurações utilizadas.

4.3.1 Dataset CIC-MalMem-2022

O conjunto de dados utilizado neste trabalho é o CIC-MalMem-2022, desenvolvido pelo *Canadian Institute for Cybersecurity (CIC)*, o qual reúne amostras de memória extraídas de sistemas comprometidos por diferentes tipos de infecções maliciosas, incluindo *ransomware, trojan, spyware* e também amostras benignas (Cybersecurity Research Lab – University of New Brunswick, 2022). A escolha desse *dataset* se justifica por sua abrangência e riqueza de informações, com 58.596 registros distribuídos em 55 atributos, que englobam características relacionadas a processos em execução, módulos carregados, acessos a arquivos, conexões de rede, bibliotecas dinâmicas (DLLs) e serviços do sistema.

As amostras estão organizadas em 20 categorias distintas, cada uma representando uma família específica de *malware*, o que torna a base especialmente adequada para a análise comparativa entre comportamentos maliciosos diversos.

Tabela 3 – Atributos do *dataset* CIC-MalMem-2022

Atributos	
pslist.nproc	pslist.nppid
pslist.avg_threads	pslist.nprocs64bit
pslist.avg_handlers	dlllist.ndlls
dlllist.avg_dlls_per_proc	handles.nhandles
handles.avg_handles_per_proc	handles.nport
handles.nfile	handles.nevent
handles.ndesktop	handles.nkey
handles.nthread	handles.ndirectory
handles.nsemaphore	handles.ntimer
handles.nsection	handles.nmutant
ldrmodules.not_in_load	ldrmodules.not_in_init
ldrmodules.not_in_mem	ldrmodules.not_in_load_avg
ldrmodules.not_in_init_avg	ldrmodules.not_in_mem_avg
malfind.ninjections	malfind.commitCharge
malfind.protection	malfind.uniqueInjections
psxview.not_in_pslist	psxview.not_in_eprocess_pool
psxview.not_in_ethread_pool	psxview.not_in_pspcid_list
psxview.not_in_csrss_handles	psxview.not_in_session
psxview.not_in_deskthrd	psxview.not_in_pslist_false_avg
psxview.not_in_eprocess_pool_false_avg	psxview.not_in_ethread_pool_false_avg
psxview.not_in_pspcid_list_false_avg	psxview.not_in_csrss_handles_false_avg
psxview.not_in_session_false_avg	psxview.not_in_deskthrd_false_avg
modules.nmodules	svcscan.nservices
svcscan.kernel_drivers	svcscan.fs_drivers
svcscan.process_services	svcscan.shared_process_services
svcscan.interactive_process_services	svcscan.nactive
callbacks.ncallbacks	callbacks.nanonymous
callbacks.ngeneric	

4.3.2 Algoritmos de Inteligência Artificial

Neste estudo, foram selecionados quatro algoritmos clássicos, cada um representando uma vertente distinta do campo do aprendizado de máquina. A seleção visa garantir diversidade metodológica, possibilitando uma análise comparativa entre diferentes paradigmas de classificação aplicados à detecção de *trojans*. Os modelos escolhidos são:

- **Rede Neural Artificial**: modelo pertencente à categoria de *Aprendizado Profundo*, destaca-se pela sua capacidade de modelar relações complexas entre os atributos, sendo particularmente útil na detecção de padrões sutis e não lineares, como os presentes em ataques do tipo *trojans*. No entanto, requer alto poder computacional e apresenta maior risco de sobreajuste.

- **Naive Bayes**: classificador probabilístico baseado no Teorema de Bayes, é eficiente com dados de alta dimensionalidade e apresenta bom desempenho em problemas com atributos independentes. Sua simplicidade o torna uma boa referência para

comparação, apesar das limitações associadas à suposição de independência entre as variáveis.

- **Árvore de Decisão**: modelo fundamentado em regras, é amplamente utilizado pela sua interpretabilidade e desempenho satisfatório em bases tabulares. Na detecção de *trojans*, sua transparência é um diferencial importante, embora exija controle sobre sua complexidade para evitar sobreajuste.

- ***Support Vector Machine (SVM)***: modelo pertencente à classe dos classificadores lineares com margem máxima, o SVM oferece alta precisão, especialmente quando os dados são separáveis de forma clara. É uma técnica robusta na detecção de *trojans*, porém sensível à escolha de parâmetros e ao volume de dados, o que pode impactar seu tempo de processamento.

A diversidade de algoritmos adotada neste estudo permite uma análise comparativa eficaz da detecção de *trojans*, equilibrando modelos interpretáveis e técnicas sofisticadas, o que favorece a compreensão de suas vantagens e limitações em diferentes contextos de ameaça.

4.3.3 Parâmetros dos Modelos

Todos os modelos foram submetidos à mesma divisão de dados, com validação cruzada (k-Fold). A Tabela 4 detalha os parâmetros utilizados na confecção dos modelos.

Tabela 4 – Parâmetros dos modelos de classificação utilizados

Algoritmo	**Parâmetros Principais**
Rede Neural Artificial	Camadas: 5 (3 Dense com 15, 15 e 8 neurônios, ativação ReLU); 1 camada Dropout (taxa 0,1); 1 camada de saída com 1 neurônio (Sigmoid); 100 épocas; *batch_size* = 10; otimizador: Adam; função de perda: *binary_crossentropy*.
Árvore de Decisão	Critério: *gini*; sem profundidade máxima definida; sem restrição de número mínimo de amostras por folha (parâmetros padrão da Scikit-learn).
Naive Bayes	Classificador: GaussianNB; distribuição assumida: normal; mantido com parâmetros padrão, priorizando simplicidade e eficiência.
SVM	*Kernel*: linear; parâmetro de regularização: C = 0,1.

4.4 Treinamento dos Modelos

Os modelos de classificação foram treinados utilizando a estratégia de validação conhecida como *Holdout*, em que o conjunto de dados foi dividido em duas partes: 80% dos registros foram utilizados para o treinamento dos algoritmos e os 20% restantes reservados para o teste e avaliação final do desempenho dos modelos em dados não vistos. Essa divisão tem como objetivo simular um cenário real de generalização, avaliando a capacidade do modelo de fazer previsões sobre novas instâncias.

Complementarmente, foi aplicada a técnica de validação cruzada com $k = 5$ *(5-fold cross-validation)* durante o processo de treinamento. Essa abordagem consiste em particionar o conjunto de treinamento em cinco subconjuntos (ou *folds*) de tamanhos aproximadamente iguais, garantindo que a proporção entre as classes seja preservada em cada divisão.

Em cada iteração, quatro desses subconjuntos são utilizados para treinar o modelo, enquanto o quinto é usado para validação. O processo se repete cinco vezes, alternando o *fold* de validação, e os resultados são posteriormente agregados. Essa técnica aumenta a robustez estatística das métricas avaliadas, reduz o risco de sobreajuste e contribui para uma estimativa mais confiável do desempenho dos modelos.

Todos os modelos foram avaliados em condições experimentais padronizadas, utilizando as mesmas amostras de dados, técnicas de validação e métricas de desempenho. Essa uniformidade garante uma comparação igualitária entre os algoritmos, permitindo analisar de forma justa seu comportamento em termos de acurácia, tempo de inferência e capacidade de generalização.

Ressalta-se que o protocolo de treinamento adotado neste estudo segue os princípios descritos em (LEONEL; MOLINOS; MIANI, 2024), assegurando consistência metodológica e reprodutibilidade dos experimentos.

4.5 Seleção de Características

A seleção de características foi uma etapa fundamental para aprimorar os modelos de aprendizado de máquina aplicados à detecção de *trojans*, visando reduzir a dimensionalidade dos dados sem comprometer o desempenho preditivo. Foram utilizadas as técnicas de *Feature Importance* e SHAP aplicadas ao classificador de Árvore de Decisão, permitindo identificar os atributos mais relevantes com base na redução de impureza e na contribuição individual para as predições. Essa abordagem não apenas otimizou o tempo de inferência e mitigou o risco de sobreajuste, como também promoveu maior interpretabilidade e compreensão do comportamento dos modelos.

Após a redução do conjunto de características, todos os modelos foram (re)treinados com o novo *dataset*, produzido a partir deste processo de seleção, e a etapa de treinamento

foi realizada para observação das métricas estáticas definidas neste método.

4.6 Métricas de Avaliação

As curvas de aprendizado foram construídas com base na evolução da **acurácia** e da **função de perda** (*loss*) ao longo do processo de treinamento. Essas curvas são essenciais para identificar o comportamento dos modelos durante o ajuste aos dados, permitindo distinguir entre três cenários comuns:

Para uma avaliação quantitativa dos modelos, foram geradas as respectivas **matrizes de confusão**, compostas por quatro elementos fundamentais:

- **TP (True Positives)**: casos de *trojan* corretamente classificados como *trojan*;
- **TN (True Negatives)**: casos benignos corretamente classificados como benignos;
- **FP (False Positives)**: casos benignos incorretamente classificados como *trojan*;
- **FN (False Negatives)**: casos de *trojan* incorretamente classificados como benignos.

A partir desses valores, foram calculadas as principais métricas de avaliação, conforme apresentadas na Tabela 5:

Tabela 5 – Fórmulas das métricas de avaliação

Métrica	**Fórmula**
Acurácia (ACC)	$\text{ACC} = \frac{TP+TN}{TP+TN+FP+FN}$
Taxa de Verdadeiros Positivos (TPR)	$\text{TPR} = \frac{TP}{TP+FN}$
Taxa de Falsos Positivos (FPR)	$\text{FPR} = \frac{FP}{FP+TN}$

Essas métricas, aliadas às curvas de aprendizado, possibilitam uma avaliação detalhada do desempenho dos modelos, indicando sua capacidade de identificar corretamente ameaças (**TPR**) ao mesmo tempo em que minimizam alarmes falsos (**FPR**). Para a geração das visualizações, foi utilizada a linguagem **Python** (Rossum e Jr (2009)), uma linguagem de programação de alto nível, interpretada e multiparadigma, essencial para a manipulação de dados e execução de algoritmos. Essa linguagem foi empregada em conjunto com as bibliotecas **Matplotlib** (Hunter (2007)), que forneceu as ferramentas fundamentais para a criação de gráficos 2D estáticos e curvas de desempenho com controle preciso; **Seaborn** (Waskom (2021)), que complementou o **Matplotlib** ao oferecer uma interface de alto nível para a geração de gráficos estatísticos mais sofisticados e esteticamente atraentes, facilitando a visualização de relações complexas nos dados; e **Plotly** (Plotly Technologies Inc. (2015)), que possibilitou a criação de gráficos interativos baseados na web, adicionando uma camada de dinamicidade e exploração detalhada aos

resultados por meio de funcionalidades como zoom e *tooltips*. Juntas, essas ferramentas foram cruciais para a criação de gráficos exploratórios, curvas de desempenho e análises visuais complementares que enriqueceram a avaliação do desempenho dos modelos.

Além da avaliação estática baseada em métricas de desempenho, esta etapa do trabalho incluiu uma análise dinâmica, voltada para medir o tempo de inferência dos modelos. O objetivo foi comparar não apenas a eficácia preditiva, mas também a eficiência operacional de cada algoritmo na classificação de instâncias como benignas ou maliciosas.

A análise foi conduzida em dois cenários distintos: inferência individual e inferência em bloco. No primeiro, o tempo foi calculado com base na média de 100 classificações isoladas, enquanto no segundo, utilizou-se a função *timeit* para medir o tempo necessário para processar lotes de dados completos. Ambos os métodos foram aplicados a subconjuntos de validação com 100 instâncias, permitindo avaliar o desempenho dos modelos em diferentes condições de uso.

Além disso, foi realizado um experimento adicional para comparar o tempo de execução em blocos com diferentes tamanhos (100, 200 e 300 exemplos), a fim de investigar a escalabilidade dos modelos. Os resultados foram visualizados por meio de gráficos de colunas, construídos com auxílio das bibliotecas *Matplotlib*, *time* e *timeit*, evidenciando o impacto do volume de dados no tempo de processamento e fornecendo subsídios para avaliar a viabilidade dos modelos em aplicações práticas.

Capítulo 5

Experimentação e Síntese de Resultados

Esta seção destaca o experimento realizado neste trabalho, a saber: (a) pré-processamento do *dataset*, (b) treinamento dos modelos, que compreende o *baseline* e o (re)treinamento após o processo de seleção de características, processo de redução/seleção de características e análise de inferência dos modelos otimizados.

5.1 Avaliação do Baseline

Na ótica deste trabalho, o *baseline* refere-se à avaliação inicial de um modelo ou sistema antes de qualquer otimização ou aplicação de técnicas avançadas. Ele estabelece uma referência fundamental de desempenho, servindo como um ponto de comparação para medir a eficácia de futuras melhorias, como a seleção de características ou o ajuste de hiperparâmetros. Esse passo é crucial para entender o desempenho intrínseco do modelo e quantificar o impacto das intervenções subsequentes.

No treinamento inicial realizado no *baseline*, todos os modelos gerados obtiveram resultados muito satisfatórios, onde os modelos já demonstram um grande potencial e robustez para detectar *trojans* através da análise de memória.

5.1.1 Rede Neural

O modelo foi treinado por 100 épocas, utilizando um *batch size* de 10 amostras, com manutenção do balanceamento entre as classes benigno e *trojan* durante todo o processo de aprendizado. A Figura 2 apresenta as curvas de perda e acurácia ao longo das épocas, tanto para os conjuntos de treinamento quanto de validação. Observa-se que as curvas de perda apresentam uma queda acentuada nas primeiras iterações, seguida por uma estabilização em níveis baixos, enquanto as curvas de acurácia convergem rapidamente para valores próximos de 100%.

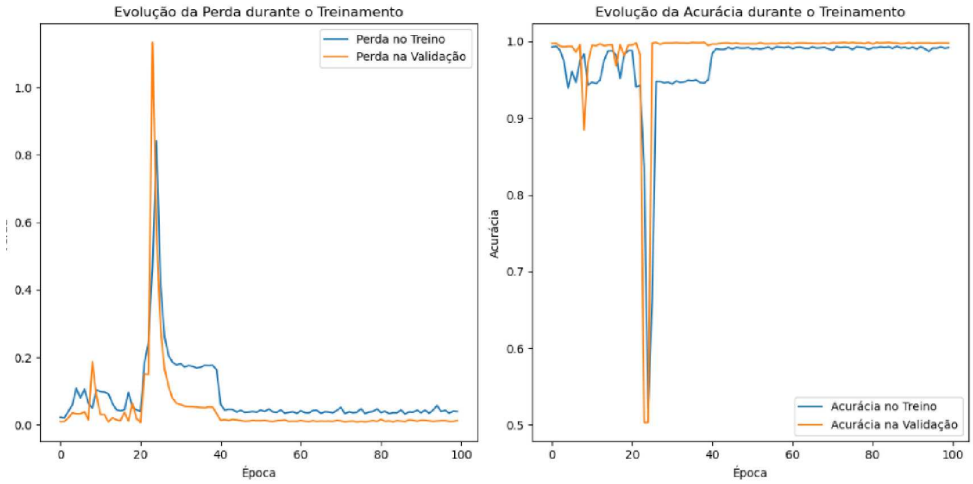

Figura 2 – Curva de Aprendizado - Rede Neural

A Figura 3 ilustra a matriz de confusão resultante da avaliação do modelo. Foram identificados **1.940 verdadeiros negativos**, correspondentes às amostras benignas corretamente classificadas, e **1.838 verdadeiros positivos**, referentes às amostras de *trojans* corretamente detectadas. Os erros de classificação foram mínimos, com apenas **15 falsos positivos** (amostras benignas classificadas erroneamente como *trojan*) e **2 falsos negativos** (casos de *trojan* não identificados). Esses resultados refletem um alto grau de precisão e sensibilidade, reforçando a efetividade do modelo.

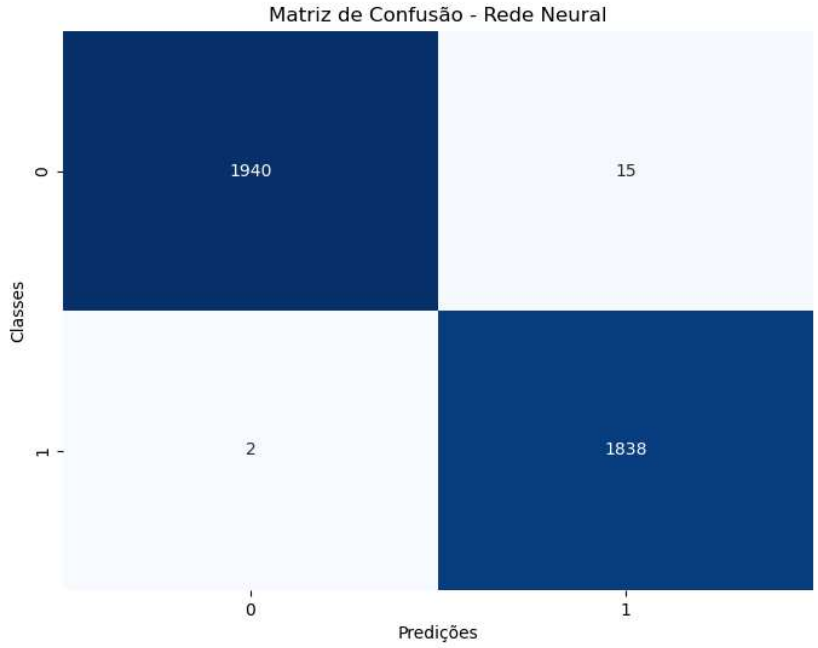

Figura 3 – Matriz de Confusão - Rede Neural

Considerando os indicadores apresentados, é possível afirmar que a rede neural alcançou um desempenho robusto, com elevada acurácia e baixa taxa de erro. A combinação

das curvas de aprendizado estáveis com a matriz de confusão altamente favorável comprova a eficiência do processo de treinamento e a adequação da arquitetura escolhida para a tarefa de detecção de ameaças do tipo *trojan*.

5.1.2 Árvore de Decisão

O modelo de Árvore de Decisão foi avaliado com base em uma estratégia de validação progressiva, variando o tamanho do conjunto de treinamento e mantendo o balanceamento entre as classes benigno e *trojan*. As Figuras 4 e 5 apresentam, respectivamente, a curva de aprendizado do modelo em termos de acurácia e perdas. Observa-se que o desempenho no conjunto de treinamento permaneceu consistentemente elevado, com acurácia próxima de 100% para todos os tamanhos de amostra. A acurácia no conjunto de validação, embora ligeiramente inferior, também apresentou valores elevados, com crescimento progressivo e estabilidade nas iterações finais.

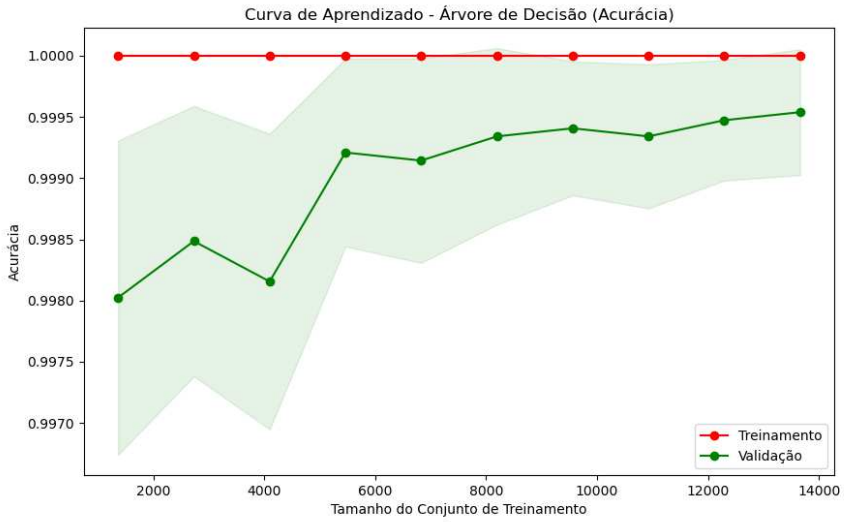

Figura 4 – Curva de Aprendizado - Árvore de Decisão (Acurácia)

A Figura 6 apresenta a matriz de confusão obtida na etapa de teste do modelo. Foram registrados **1.955 verdadeiros negativos**, representando amostras benignas corretamente classificadas, e **1.838 verdadeiros positivos**, correspondentes a amostras de *trojan* corretamente identificadas. O modelo cometeu apenas **2 falsos negativos**, ou seja, deixou de identificar dois casos de *trojan*, e não apresentou qualquer falso positivo, o que indica uma ótima classificação das amostras benignas.

Diante deste contexto, é observado que a Árvore de Decisão apresentou um desempenho satisfatório, com alta acurácia, precisão e sensibilidade. A ausência de falsos positivos é especialmente relevante para aplicações em segurança, onde a classificação incorreta de comportamentos legítimos como maliciosos pode resultar em interrupções indevidas.

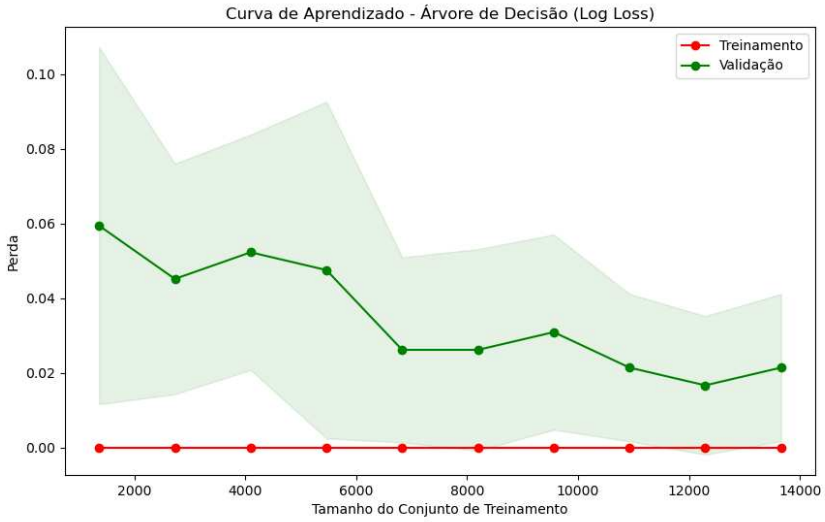

Figura 5 – Curva de Aprendizado - Árvore de Decisão (Loss)

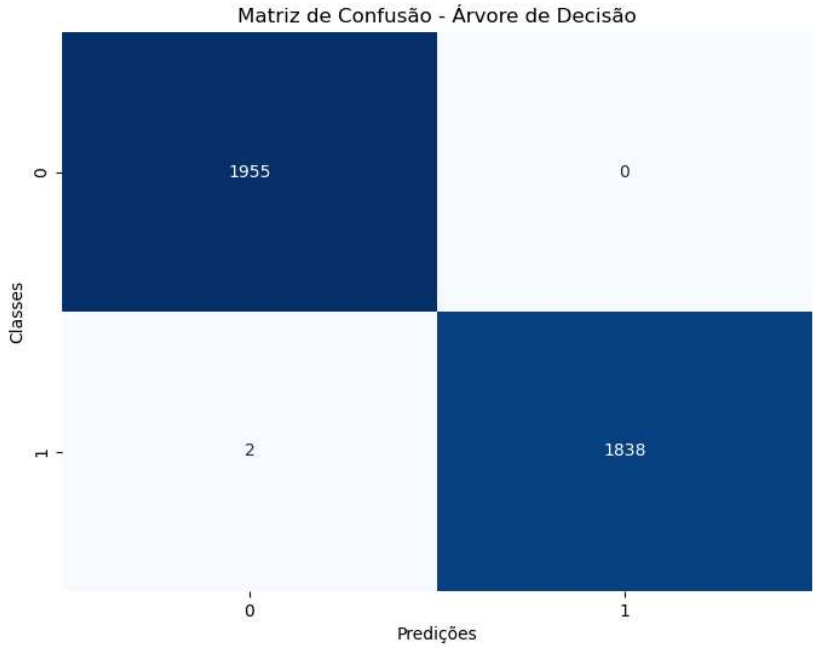

Figura 6 – Matriz de Confusão - Árvore de Decisão

5.1.3 Naive Bayes

O modelo baseado em Naive Bayes foi avaliado por meio de validação progressiva com diferentes tamanhos do conjunto de treinamento, mantendo o balanceamento entre as classes benigno e *trojan* durante todo o processo. A Figura 7 apresenta a curva de aprendizado em termos de acurácia. Observa-se que os valores de acurácia se mantêm relativamente altos, variando entre 99,1% e 99,5%, tanto para os dados de treino quanto

para os dados de validação. Apesar da oscilação inicial da curva de validação, o modelo demonstrou tendência à estabilização nas iterações com maior volume de dados, sugerindo uma capacidade de generalização aceitável.

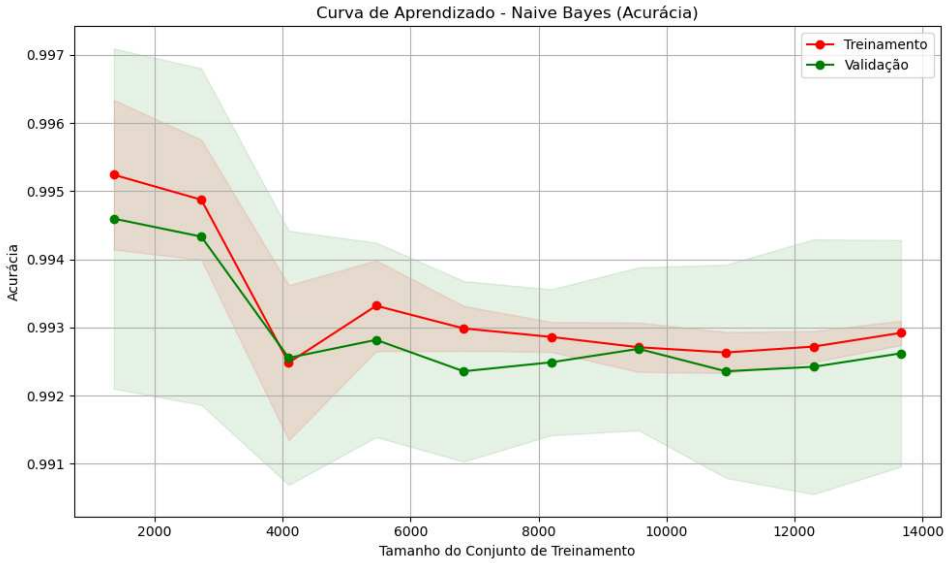

Figura 7 – Curva de Aprendizado - Naive Bayes (Acurácia)

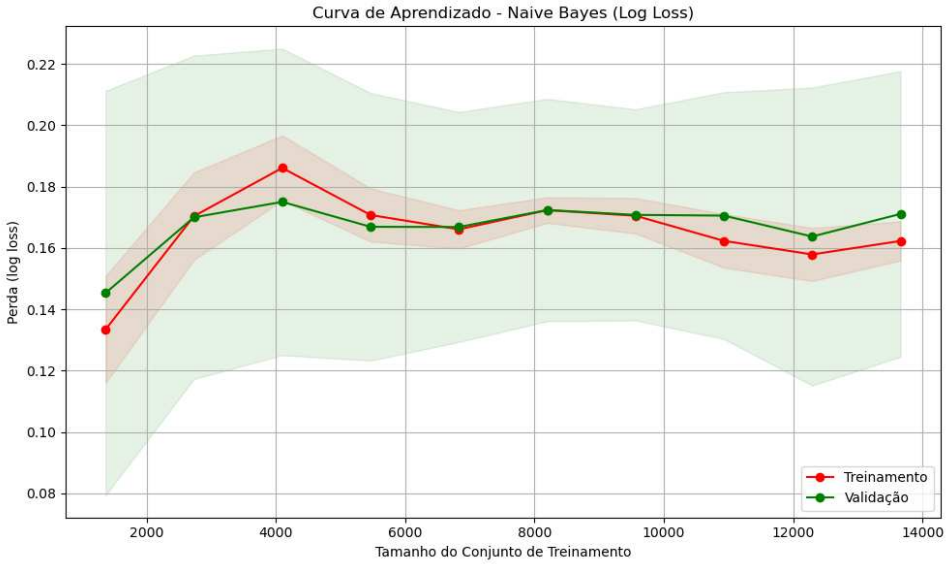

Figura 8 – Curva de Aprendizado - Naive Bayes (Loss)

A Figura 8 exibe o comportamento da função de perda logarítmica para os mesmos experimentos. Os valores de perda oscilaram levemente ao longo dos diferentes tamanhos de conjunto de treinamento, com uma leve tendência de redução à medida que o volume de dados aumentava. A proximidade entre as curvas de treino e validação nessa métrica reforça a ideia de que, embora o modelo tenda a sobreajustar com conjuntos pequenos,

ele se estabiliza com a introdução de mais dados, evitando degradação significativa no desempenho.

A Figura 9 apresenta a matriz de confusão gerada a partir da avaliação final do modelo. Foram registrados **1.936 verdadeiros negativos** e **1.835 verdadeiros positivos**, com **19 falsos positivos** (amostras benignas incorretamente classificadas como *trojan*) e **5 falsos negativos** (amostras de *trojan* não identificadas). Esses números indicam um bom desempenho geral do classificador, ainda que com erros de classificação ligeiramente superiores aos observados em modelos como Rede Neural e Árvore de Decisão.

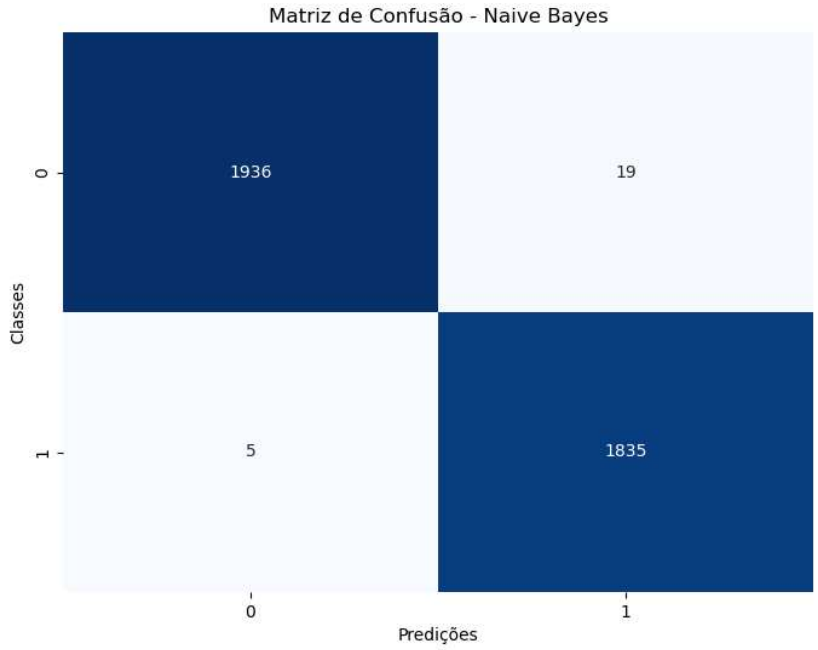

Figura 9 – Matriz de Confusão - Naive Bayes

Considerando o conjunto dos resultados apresentados, o modelo Naive Bayes demonstrou alta capacidade de classificação, apresentando acurácia elevada, perdas moderadas e desempenho consistente em diferentes volumes de dados. Embora tenha apresentado maior propensão ao erro em comparação com os modelos mais complexos, trata-se de uma alternativa viável e eficiente, especialmente em cenários com restrições de tempo ou recursos computacionais.

5.1.4 Support Vector Machine

A Figura 10 exibe a curva de aprendizado do modelo *Support Vector Machine* (SVM), considerando a acurácia em diferentes tamanhos do conjunto de treinamento. Observa-se que os valores de acurácia permanecem elevados ao longo de todas as etapas, embora sem prejuízo significativo à generalização. A maior proximidade entre as curvas ocorre por

volta de 6.000 amostras, indicando que, nesse ponto, o modelo atinge um equilíbrio mais favorável entre aprendizado e capacidade de predição.

A Figura 11 complementa a análise ao apresentar a curva de perda, uma métrica que penaliza predições incorretas com elevada confiança. Nota-se uma variação relativamente baixa ao longo dos diferentes volumes de dados, com estabilidade nas curvas de treino e validação, o que reforça a robustez do modelo, mesmo frente a pequenas flutuações nos dados.

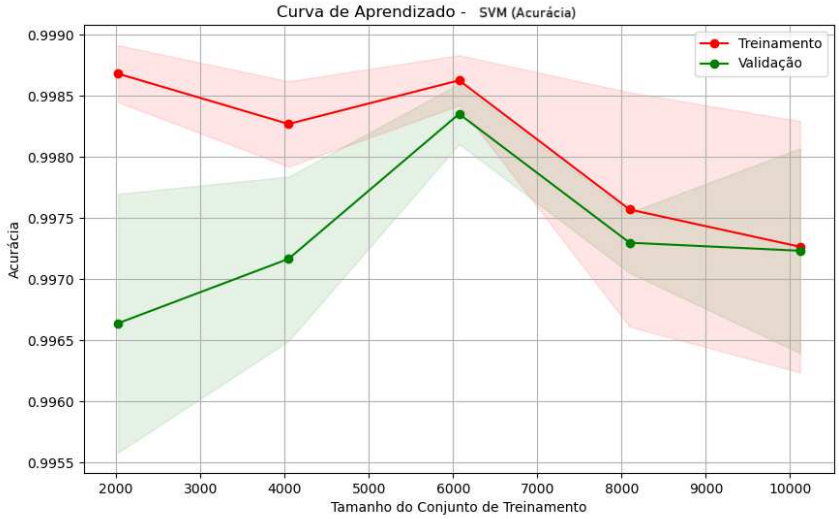

Figura 10 – Curva de Aprendizado - SVM (Acurácia)

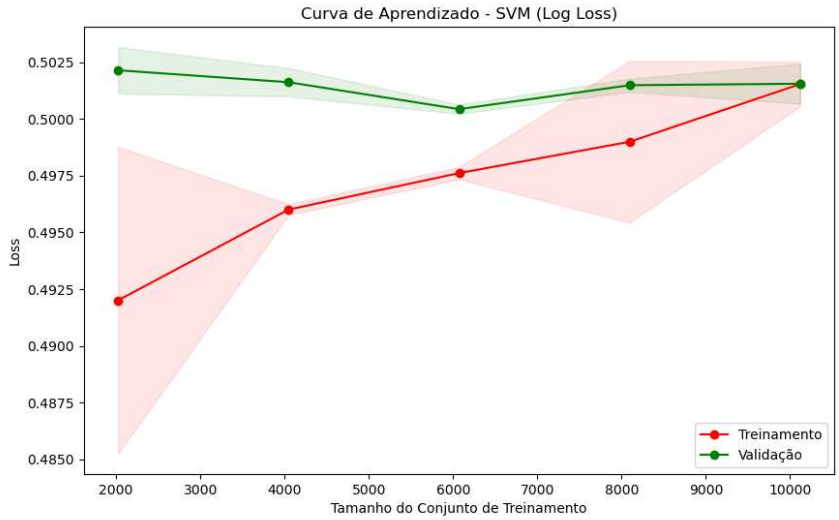

Figura 11 – Curva de Aprendizado - SVM (Loss)

Já a Figura 12 ilustra a matriz de confusão resultante da avaliação final do modelo. Foram registrados **1.954 verdadeiros negativos** e **1.839 verdadeiros positivos**, com

apenas **1 falso positivo** e **1 falso negativo**, totalizando um desempenho praticamente perfeito.

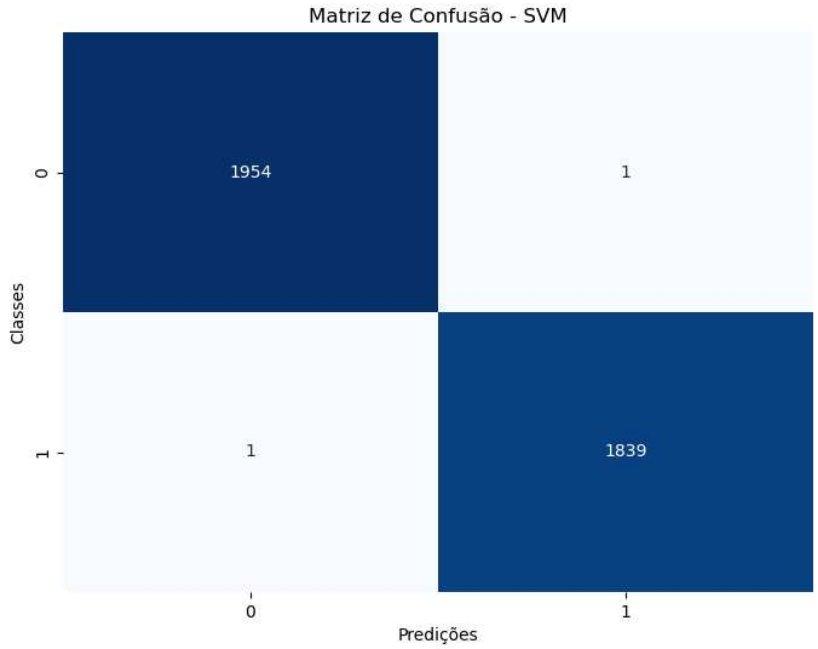

Figura 12 – Matriz de Confusão - SVM

Esses resultados evidenciam a eficácia do SVM na tarefa de classificação, com boa capacidade de discriminar entre amostras benignas e maliciosas, ao mesmo tempo em que mantém uma taxa de erro baixa.

5.1.5 Síntese do Baseline

A Tabela 6 ilustra os resultados das métricas estáticas analisadas dos algorimos durante o processo de avaliação do *baseline*. Ressalta-se que os dados ilustrados na Tabela 6 foram extraídos da média da validação cruzada (*5 folds*).

Algoritmo	**Acurácia (média)**
Redes Neurais (ANN)	0.9981
Árvore de Decisão (DT)	0.9994
Naive Bayes (NB)	0.9934
Support Vector Machine (SVM)	0.9972

Tabela 6 – Média das acurácias dos algoritmos selecionados

5.2 Seleção de Características

Visando à redução da dimensionalidade do conjunto de dados e à melhoria do desempenho dos modelos de aprendizado de máquina, foi realizada uma etapa de análise e seleção de caracaterísticas. Para isso, aplicaram-se duas abordagens complementares: a técnica de *Feature Importance*[1], associada a algoritmos baseados em árvores de decisão, e a análise *SHAP*. Essa etapa foi essencial para identificar atributos redundantes ou pouco informativos.

5.2.1 Principais Características com Árvore de Decisão

A Figura 13 ilustra a estrutura da árvore de decisão gerada pelo modelo, evidenciando os principais atributos utilizados no processo de classificação das amostras entre as classes benigno e *trojan*. Observa-se que a variável *svcscan.nservices* ocupa a raiz da árvore, sendo a mais relevante para a separação inicial dos dados. Em seguida, atributos como *svcscan.process_services*, *dlllist.ndlls*, *handles.nthread* e *callbacks.ncallbacks* são recorrentes nos níveis superiores da árvore, indicando sua forte influência na tomada de decisão.

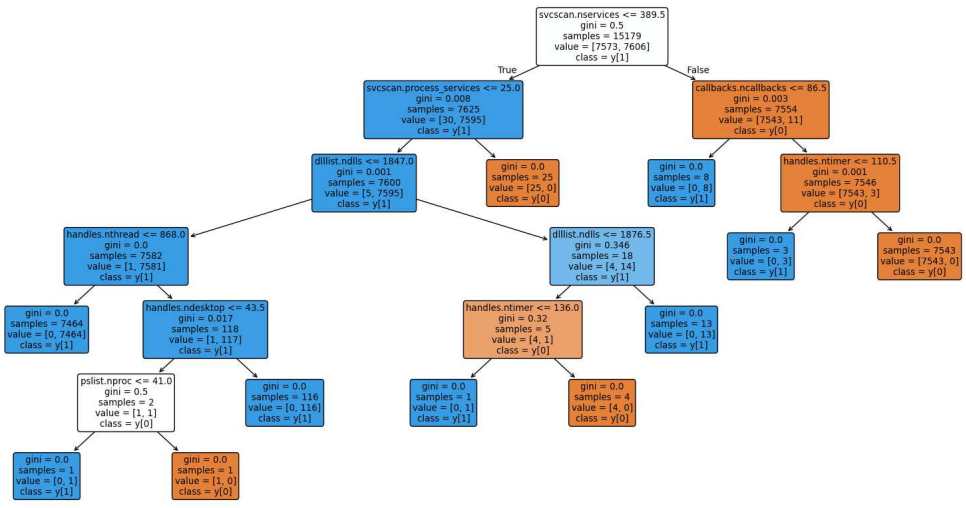

Figura 13 – Árvore de Decisão no processo de seleção das características

Ainda de acordo com a Figura 13, a métrica Gini, exibida em cada nó, demonstra que a maioria das subdivisões resulta em grupos altamente puros, com predominância

[1] A técnica de *Feature Importance* estima a relevância de cada atributo para o modelo com base na sua contribuição na redução da impureza (como Gini ou entropia) ao longo das divisões nas árvores de decisão. Atributos que mais reduzem a impureza tendem a receber maiores importâncias.

clara de uma das classes. Essa estrutura revela não apenas a capacidade explicativa do modelo, mas também destaca quais variáveis possuem maior poder discriminativo, sendo fundamentais para o entendimento do comportamento das amostras e para a interpretação transparente das decisões do modelo.

5.2.2 *SHAP - SHapley Additive exPlanations*

Baseado na análise da importância dos atributos na Árvore de Decisão, pode-se observar a predominânica de 6 atributos para a decisão do modelo. Após esta seleção inicial de atributos, a técnica SHAP foi aplicada para compreender com maior profundidade a influência de cada variável nas decisões do modelo, promovendo maior transparência e reforçando a confiança na capacidade do classificador em diferenciar comportamentos benignos e maliciosos.

A Figura 14 apresenta a distribuição dos valores SHAP para os atributos utilizados no modelo de Árvore de Decisão. Cada ponto no gráfico representa uma instância do conjunto de dados, sendo sua posição no eixo horizontal o impacto da respectiva característica na saída do modelo, enquanto as cores indicam a magnitude do valor da característica (de baixo a alto, em gradiente de azul a rosa). Observa-se que as variáveis *svcscan.nservices, dlllist.ndlls* e *svcscan.process_services* são as que exercem maior influência nas predições do modelo, evidenciando um impacto significativo tanto positivo quanto negativo, dependendo do valor da variável em cada instância. Além disso, atributos como *pslist.nproc, handles.nthread e handles.ndesktop* também contribuem de forma relevante, embora em menor escala. Este gráfico reforça a importância dessas variáveis para a distinção entre comportamentos benignos e maliciosos, fornecendo uma explicação interpretável sobre como o modelo toma suas decisões.

Ante o exposto, com base na seleção de características realizada, a Tabela 7 ilustra os principais atributos.

Tabela 7 – Principais atributos segundo os valores SHAP – Árvore de Decisão

Ordem	**Feature**	**Descrição estimada**
1	`svcscan.nservices`	Número de serviços identificados pelo scanner de serviços.
2	`dlllist.ndlls`	Quantidade de DLLs carregadas.
3	`svcscan.process_services`	Número de serviços vinculados a processos.
4	`pslist.nproc`	Número de processos ativos.
5	`handles.nthread`	Quantidade de *threads* manipuladas.
6	`handles.ndesktop`	Número de objetos do tipo *desktop*.
7	`svcscan.kernel_drivers`	Quantidade de *kernel drivers* identificados.
8	`handles.ntimer`	Número de *timers* manipulados.
9	`callbacks.ncallbacks`	Número de *callbacks* registrados.
10	`handles.nsection`	Quantidade de *sections* manipuladas.

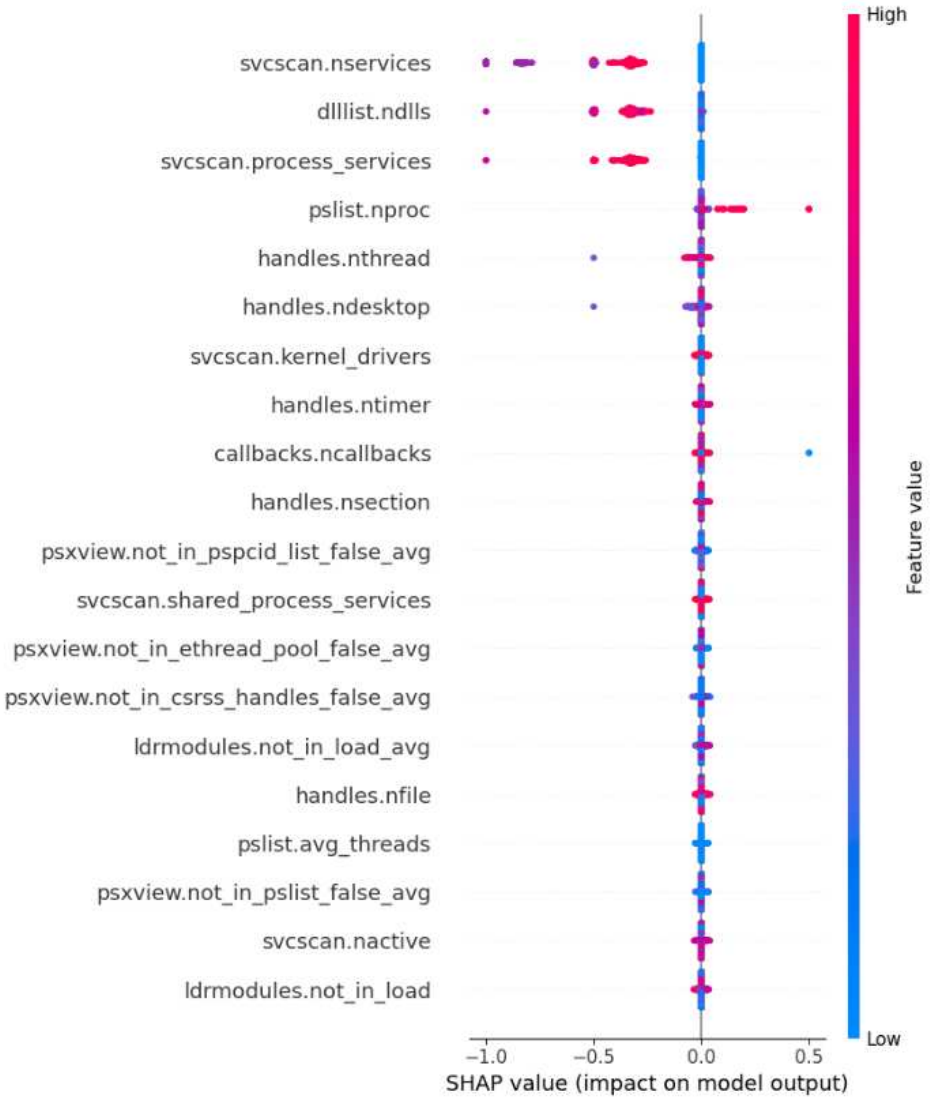

Figura 14 – Valores dos atributos usando Árvore de Decisão

5.3 (Re)Avaliação dos Modelos

Com base na análise da Árvore de Decisão e dos valores SHAP apresentados, foi possível identificar as características mais relevantes para o processo de classificação. A partir dessa seleção refinada de atributos, todos os modelos foram reavaliados com o objetivo de mensurar o impacto da redução de dimensionalidade sobre o desempenho.

Nesta nova etapa, os modelos foram novamente treinados e validados, sendo capturado novamente as métricas de **acurácia**, **precisão**, **recall** e **F1-score**, sendo ainda adicionado a análise de desempenho preditivo envolvendo o **tempo de inferência**, que reflete a eficiência computacional dos modelos durante a fase de predição.

5.3.1 Rede Neural

Após o processo de otimização por seleção de características, observa-se que o modelo de Rede Neural manteve desempenho elevado, conforme ilustrado na Figura 15. É possível observar uma diferença acentuada entre os conjuntos de treino e validação, especialmente nas épocas finais, possivelmente causada por variações nos dados ou pela menor quantidade de atributos disponíveis. No entanto, a acurácia na validação permaneceu consistentemente superior à do treinamento ao longo de grande parte do processo.

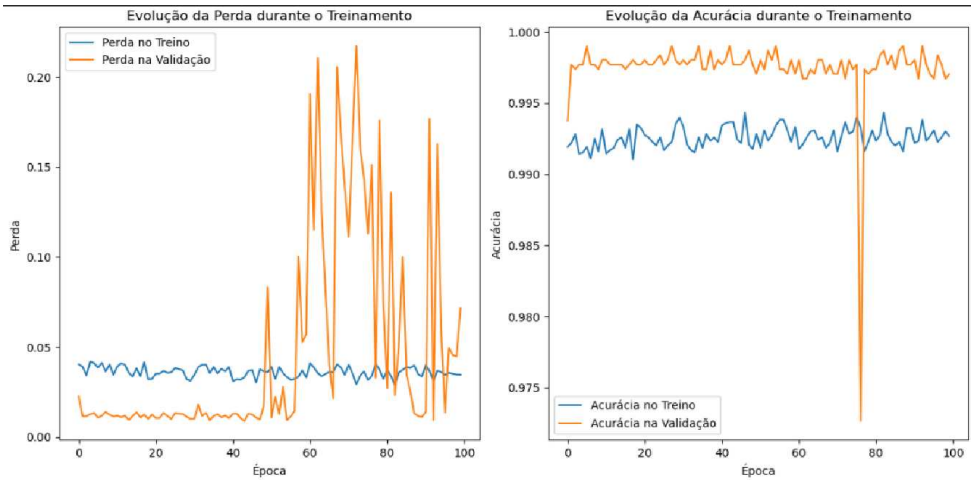

Figura 15 – Curva de Aprendizado - Rede Neural

A matriz de confusão apresentada na Figura 16 reforça a robustez do modelo após a seleção dos atributos. Foram obtidos **1.908 verdadeiros negativos** e **1.880 verdadeiros positivos**, com apenas **6 falsos positivos** e **1 falso negativo**, o que representa uma taxa de erro extremamente baixa.

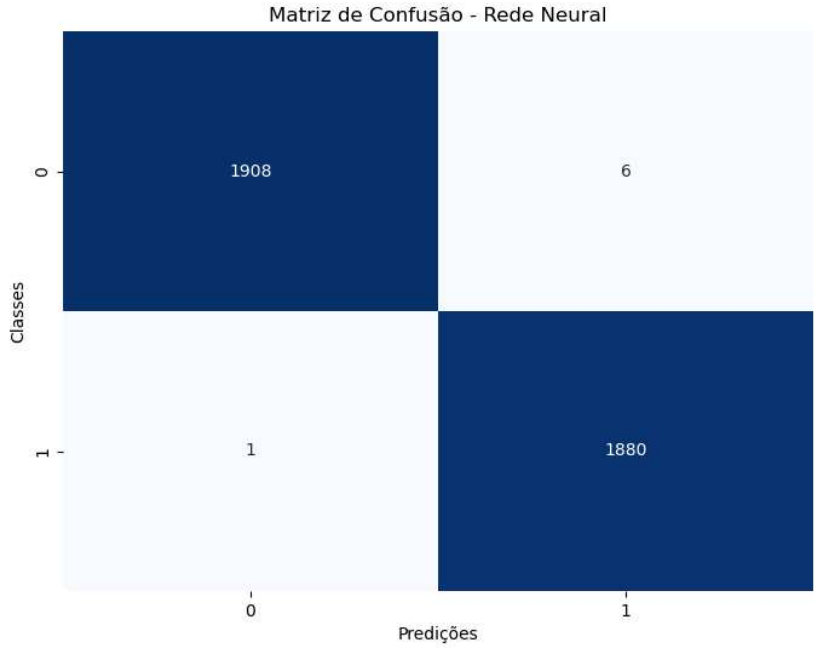

Figura 16 – Matriz de Confusão - Rede Neural

Esses resultados demonstram que a redução da dimensionalidade não comprometeu a capacidade da Rede Neural de discriminar eficientemente entre amostras benignas e *trojans*. Pelo contrário, ao reduzir a complexidade do modelo, foi possível manter a acurácia elevada e potencialmente otimizar o tempo de inferência, tornando a solução ainda mais viável para cenários reais de detecção de ameaças.

5.3.2 Árvore de Decisão

As Figuras 17 e 18, respectivamente, apresentam a curva de aprendizado e de perda do modelo de Árvore de Decisão em termos de acurácia após a seleção dos atributos. Nota-se que o desempenho no conjunto de treinamento se mantém praticamente constante, próximo de 100% em todos os tamanhos de amostras, enquanto a acurácia no conjunto de validação também se mantém elevada, com pequenas variações e uma tendência à estabilidade a partir de 6.000 amostras. Essa consistência sugere que o modelo, mesmo após a redução de atributos, preservou boa capacidade de generalização, sem sofrer com subajuste ou sobreajuste significativo.

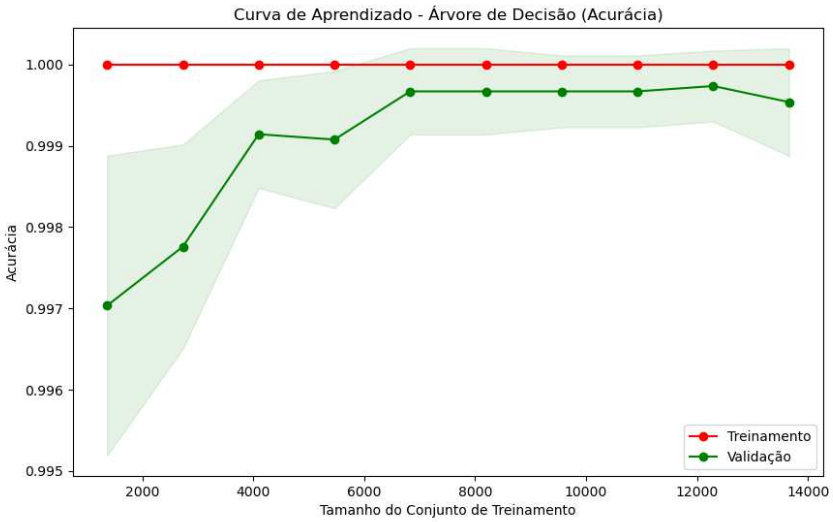

Figura 17 – Curva de Aprendizado - Árvore de Decisão (Acurácia)

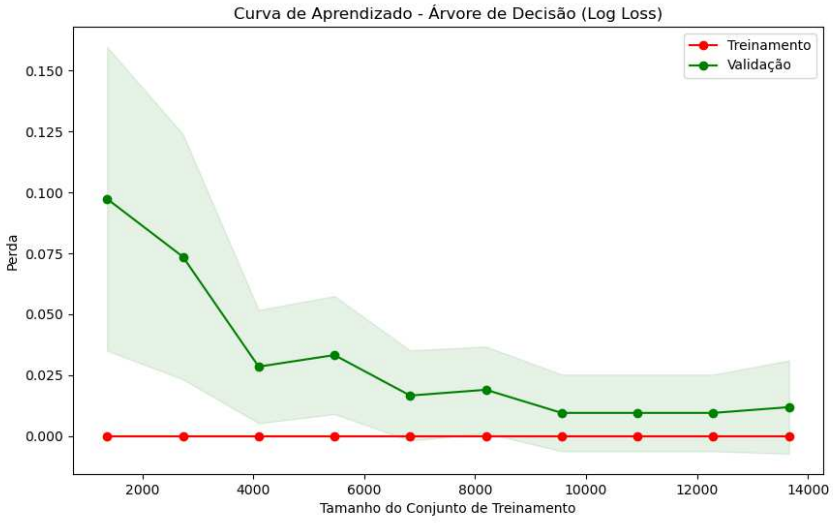

Figura 18 – Curva de Aprendizado - Árvore de Decisão (Loss)

Complementando a análise, a Figura 19 exibe a matriz de confusão obtida na avaliação final. O modelo classificou corretamente todas as amostras, totalizando **1.914 verdadeiros negativos** e **1.881 verdadeiros positivos**, sem ocorrência de falsos positivos ou falsos negativos.

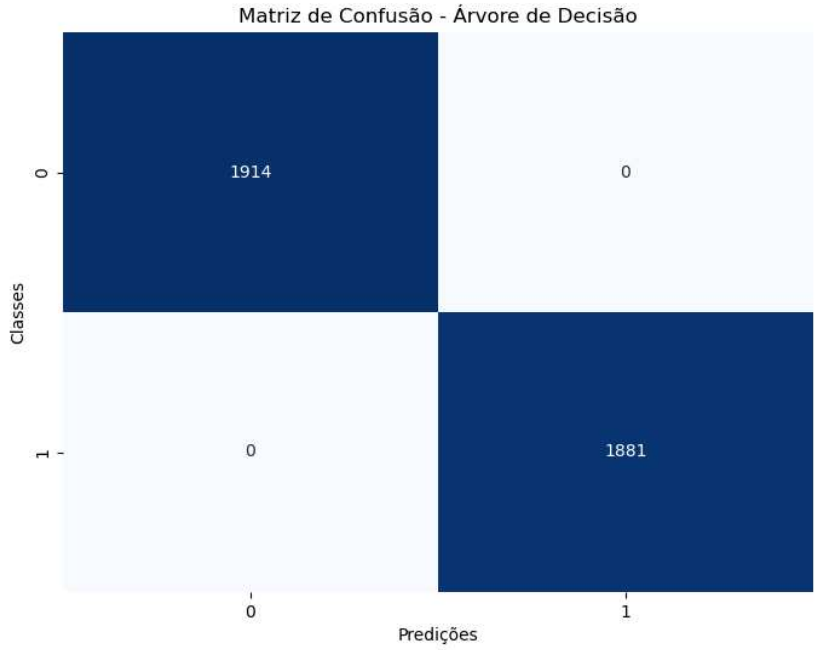

Figura 19 – Matriz de Confusão - Árvore de Decisão

Esse resultado demonstra um bom desempenho do modelo de Árvore de Decisão, confirmando sua eficiência tanto em termos preditivos quanto em simplicidade computacional.

5.3.3 Naive Bayes

A Figura 20 apresenta a curva de aprendizado do modelo Naive Bayes em relação à acurácia após a seleção das características. Observa-se que o desempenho do modelo é estável e elevado para diferentes tamanhos de conjuntos de treinamento, com acurácia acima de 99,4% tanto para os dados de treinamento quanto de validação. A proximidade entre as curvas ao longo da maior parte dos testes sugere uma boa capacidade de generalização e baixo risco de sobreajuste, mesmo após a redução dos atributos.

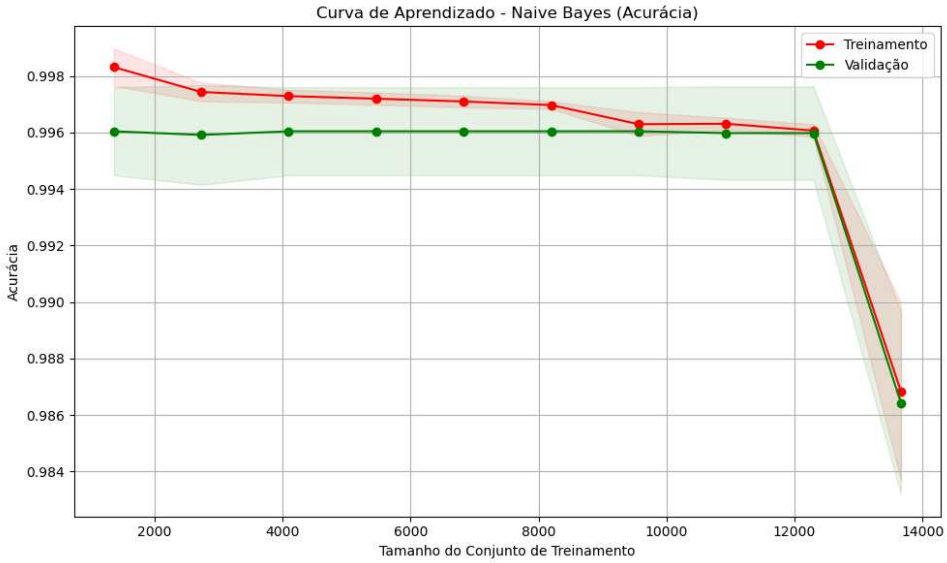

Figura 20 – Curva de Aprendizado - Naive Bayes (Acurácia)

Na Figura 21, é ilustrada a curva de perda a qual indica uma leve elevação da perda no conjunto de validação à medida que o tamanho do conjunto de dados aumenta. Ainda assim, o desempenho geral permanece sólido e compatível com os padrões esperados desse classificador.

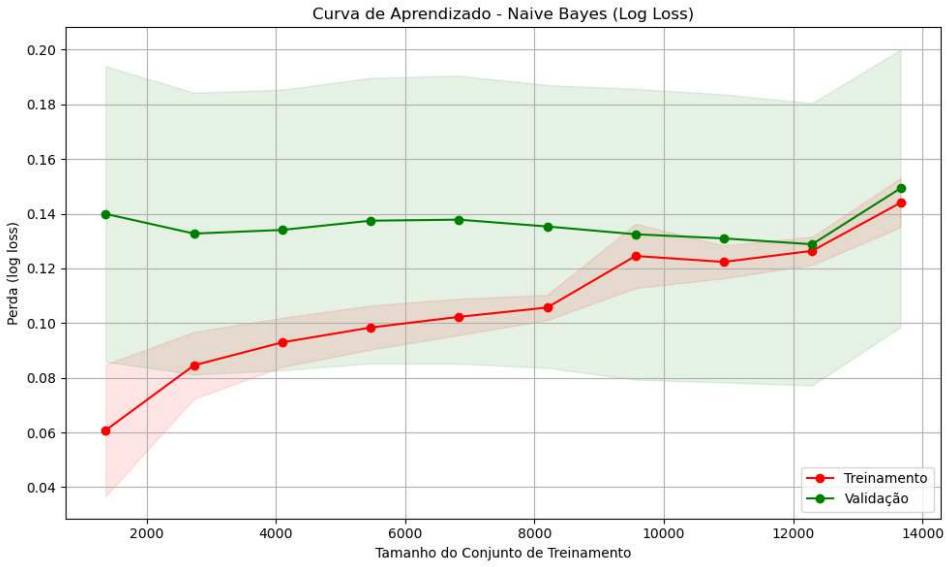

Figura 21 – Curva de Aprendizado - Naive Bayes (Loss)

A matriz de confusão exibida na Figura 22 revela um bom desempenho do modelo na classificação final. Foram registrados **1.868 verdadeiros negativos** e **1.872 verdadeiros positivos**, com **46 falsos positivos** e **9 falsos negativos**. Esses valores indicam uma leve tendência ao aumento de falsos positivos em comparação aos modelos mais ro-

bustos, como a Rede Neural ou a Árvore de Decisão, mas ainda dentro de um intervalo tolerável para aplicações práticas.

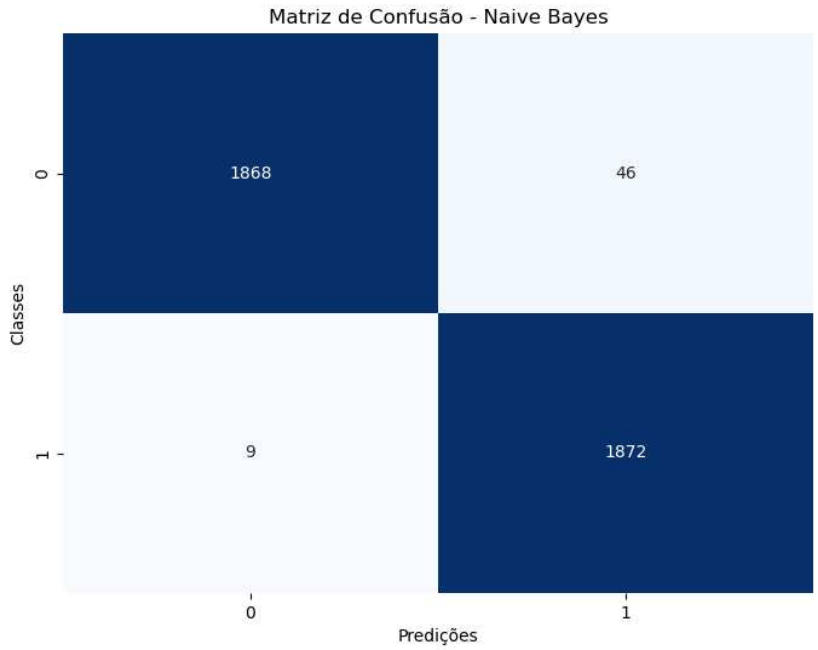

Figura 22 – Matriz de Confusão - Naive Bayes

De forma geral, o Naive Bayes demonstrou ser eficiente, simples e capaz de produzir resultados satisfatórios mesmo após a seleção de atributos.

5.3.4 Support Vector Machine

As Figuras 23 e 24 apresentam, respectivamente, as curvas de acurácia e perda do modelo SVM após a seleção de atributos. Observa-se que a acurácia no conjunto de treinamento se manteve elevada, próxima de 100% para todos os tamanhos de dados, enquanto a acurácia de validação apresentou crescimento gradual até cerca de 10.000 amostras, indicando que o modelo foi capaz de generalizar bem conforme exposto a maior volume de dados. A diferença entre as curvas é moderada e esperada para modelos de margem máxima. Apesar da oscilação pontual em alguns tamanhos de dados, o modelo demonstrou consistência e boa calibragem probabilística, reforçando sua confiabilidade mesmo com menor número de atributos.

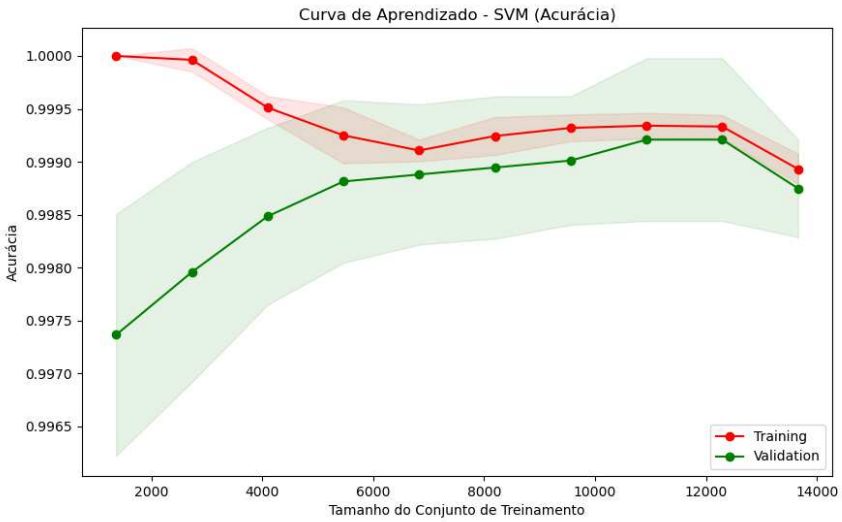

Figura 23 – Curva de Aprendizado - SVM (Acurácia)

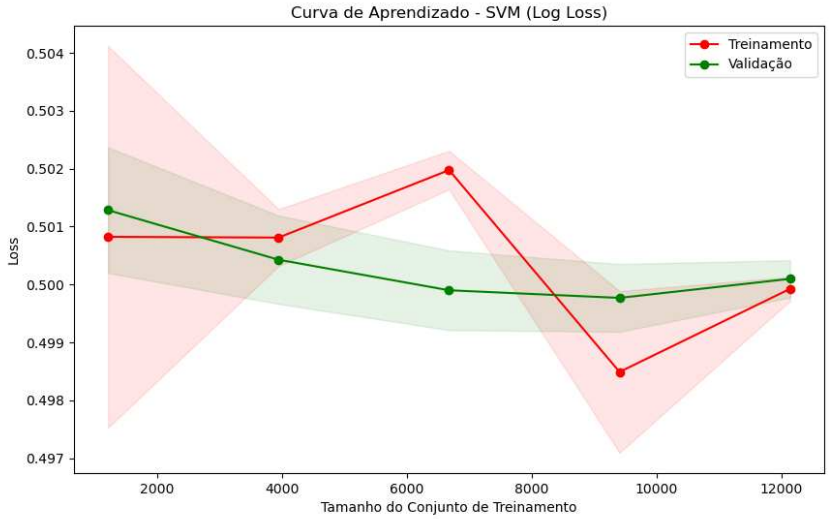

Figura 24 – Curva de Aprendizado - SVM (Loss)

A Figura 25 exibe a matriz de confusão, evidenciando um desempenho bastante sólido do classificador. Foram registrados **1.914 verdadeiros negativos** e **1.880 verdadeiros positivos**, com apenas **1 falso negativo** e nenhum falso positivo.

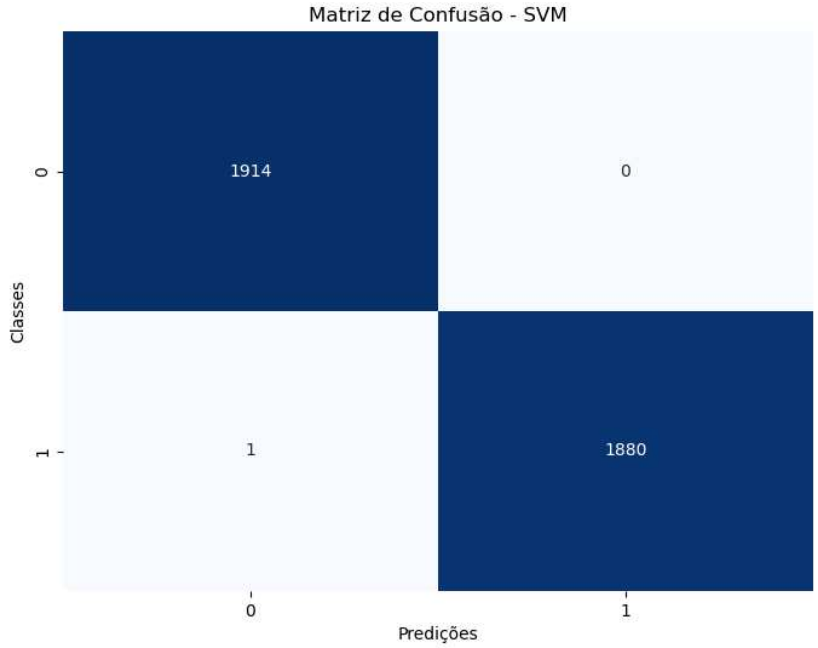

Figura 25 – Matriz de Confusão - SVM

Esses resultados destacam a precisão do modelo SVM, especialmente na identificação de instâncias benignas, além de reforçar seu baixo índice de erro e aplicabilidade prática em contextos sensíveis à segurança computacional.

5.4 Avaliação do Tempo de Inferência

Todos os modelos foram submetidos a uma análise de tempo de inferência, com o objetivo de aprofundar a avaliação quanto à sua eficiência operacional e identificar quais algoritmos apresentam melhor desempenho em termos de tempo de predição.

Para esta avaliação, foram realizados experimentos considerando dois cenários distintos: (a) a inferência individual, em que cada amostra é processada isoladamente, e (b) a inferência em blocos, com grupos de 100, 200 e 300 instâncias sendo avaliadas simultaneamente.

Essa abordagem visa não apenas medir o tempo de resposta de cada modelo em aplicações em tempo real, mas também observar o esforço computacional exigido em situações de processamento em lote. Os resultados comparativos apresentados nesta seção fornecem uma visão mais ampla sobre a viabilidade prática de cada modelo, considerando diferentes demandas operacionais e contextos de uso.

Tabela 8 – Tempo de inferência para amostras da classe *Trojan* (em segundos)

Modelo	**Individual**	**Bloco de 100**	**Bloco de 200**	**Bloco de 300**
Rede Neural (RN)	0.088250	0.103185	0.106031	0.104525
Árvore de Decisão (DT)	0.001277	0.000996	0.001016	0.000818
Naive Bayes (NB)	0.000940	0.000770	0.000706	0.000801
Support Vector Machine (SVM)	0.000983	0.001001	0.001078	0.001345

Tabela 9 – Tempo de inferência para amostras da classe *Benigno* (em segundos)

Modelo	**Individual**	**Bloco de 100**	**Bloco de 200**	**Bloco de 300**
Rede Neural (RN)	0.088646	0.126598	0.120724	0.133008
Árvore de Decisão (DT)	0.001362	0.001213	0.000910	0.001024
Naive Bayes (NB)	0.000979	0.000817	0.000749	0.000764
Support Vector Machine (SVM)	0.000955	0.000937	0.001056	0.001288

A análise dos tempos de inferência evidencia diferenças significativas no desempenho computacional entre os modelos avaliados. A Rede Neural, embora apresente elevada acurácia, demonstrou ser a menos eficiente em termos de tempo, tanto na inferência individual quanto em bloco, o que pode comprometer sua aplicação em cenários que exigem respostas rápidas.

Os modelos baseados em Árvore de Decisão, Naive Bayes e SVM apresentaram tempos de inferência muito inferiores, o que significa maior desempenho e menor esforço computacional, destacando-se como alternativas mais viáveis para ambientes com restrições de desempenho.

Entre eles, a Árvore de Decisão demonstrou excelente equilíbrio entre rapidez e precisão, especialmente no processamento em blocos. Esses resultados reforçam a importância de considerar, além da acurácia, a eficiência computacional ao selecionar modelos para aplicações práticas.

Capítulo 6

Conclusão

O presente trabalho alcançou êxito em seu objetivo de investigar, de forma exploratória e comparativa, a aplicabilidade de modelos de aprendizado de máquina na detecção de ameaças do tipo *trojan* por meio da análise de atributos extraídos da memória volátil do sistema. A metodologia empregada contemplou uma etapa inicial de pré-processamento e compreensão do conjunto de dados CIC-MalMem-2022, seguida pela construção e avaliação de quatro modelos de classificação supervisionada: Rede Neural, Árvore de Decisão, Naive Bayes e Máquina de Vetores de Suporte (SVM). Cada um desses algoritmos foi treinado, testado e validado utilizando técnicas de validação cruzada e curvas de aprendizado, possibilitando uma análise comparativa robusta em termos de acurácia, perda, capacidade de generalização e desempenho computacional.

Através da incorporação da técnica de interpretabilidade SHAP, permitiu não apenas interpretar os resultados dos modelos de forma mais transparente, mas também identificar os atributos mais relevantes para a predição, promovendo uma etapa de seleção e redução da dimensionalidade. Tal refinamento teve como objetivo eliminar atributos redundantes ou pouco informativos, preservando a performance dos modelos e, em alguns casos, promovendo melhorias nos indicadores de desempenho, especialmente na taxa de falsos positivos e falsos negativos. A técnica SHAP demonstrou ser eficaz ao evidenciar como cada atributo influencia individualmente na saída do modelo, agregando valor analítico tanto na interpretação quanto na validação dos resultados.

A avaliação dos modelos antes e depois da seleção de atributos revelou que a redução da dimensionalidade não comprometeu significativamente os resultados. Pelo contrário, modelos como Árvore de Decisão e SVM mantiveram acurácia elevada mesmo com menos atributos, evidenciando que a remoção criteriosa de variáveis pode contribuir para a construção de classificadores mais enxutos, interpretáveis e eficazes. Esse aspecto é crucial em cenários de segurança da informação, nos quais a transparência das decisões e a eficiência operacional são fundamentais.

Adicionalmente, foi realizada uma análise criteriosa do tempo de inferência dos modelos, tanto em execuções unitárias quanto em blocos de instâncias, permitindo uma

simulação mais próxima de ambientes de produção. Os testes demonstraram que algoritmos como Naive Bayes e Árvore de Decisão apresentam latência reduzida e baixo consumo computacional, sendo especialmente indicados para aplicações em tempo real, como sistemas de monitoramento contínuo, resposta automática a incidentes e detecção embarcada em dispositivos com restrição de hardware. A Rede Neural e o modelo SVM, embora mais exigentes em termos computacionais, mostraram-se competitivos e apresentaram desempenho consistente, o que reforça seu uso em contextos onde a acurácia é o fator decisivo.

Em síntese, esta pesquisa valida a viabilidade de se construir soluções inteligentes para detecção de ameaças que aliam precisão, interpretabilidade e eficiência. A combinação entre algoritmos de aprendizado de máquina, técnicas de redução de dimensionalidade e métodos de inteligência artificial explicável revelou-se promissora, oferecendo uma base sólida para aplicações práticas em segurança cibernética. Dessa forma, este trabalho representa relevância na busca por mecanismos automatizados, confiáveis e transparentes na defesa contra ameaças digitais.

6.1 Trabalhos Futuros

Como trabalhos futuros, destaca-se a possibilidade de explorar modelos mais avançados de aprendizado profundo, como redes neurais convolucionais (CNNs) aplicadas à análise de memória, com o objetivo de capturar padrões mais complexos e temporais entre os dados. Também se sugere validar os modelos desenvolvidos em ambientes dinâmicos e de coleta em tempo real, como *honeypots* ou monitores ativos de memória, o que demandaria adaptações quanto à latência, uso de recursos computacionais e robustez frente a ruídos. A implementação em dispositivos com restrições de *hardware*, como sensores de borda ou sistemas embarcados, constitui outra vertente promissora, exigindo o uso de algoritmos mais leves e eficientes. Além disso, a ampliação da base de dados para contemplar outras famílias e variantes de *malware*, como *ransomware*, *worms* e *rootkits*, permitiria avaliar a generalização do modelo frente a diferentes estratégias maliciosas. A integração dos classificadores com sistemas de resposta automática também é uma alternativa relevante, viabilizando reações imediatas como isolamento de processos ou alertas, aproximando a proposta de um sistema operável de detecção e resposta a incidentes. Também, a investigação de outras técnicas de interpretabilidade baseadas em XAI, como *LIME*, *Permutation Importance* ou *DeepLIFT*, a fim de comparar diferentes abordagens de explicação, fortalecer a confiabilidade das predições e adaptar a explicabilidade ao contexto específico de segurança da informação.

6.2 Artefatos de Software

Como parte do compromisso com a transparência e reprodutibilidade científica, este trabalho disponibiliza um artefato digital contendo os códigos, experimentos e recursos utilizados durante a pesquisa.

O repositório pode ser acessado publicamente por meio do seguinte link:
<https://github.com/dhiogosantos/notebook-tcc-2>.

Nele, encontram-se os *scripts* de execução, *dataset*, os gráficos gerados e instruções para reprodução dos resultados, permitindo que outros pesquisadores ou profissionais da área possam consultar, validar ou estender as abordagens aqui desenvolvidas.

Referências

ABUALHAJ, M. M.; AL-KHATIB, S. N. Using decision tree classifier to detect trojan horse based on memory data. **TELKOMNIKA (Telecommunication Computing Electronics and Control)**, v. 22, n. 2, p. 393–400, 2024. Citado na página 20.

ABUALHAJ, M. M. et al. Enhancing spyware detection by utilizing decision trees with hyperparameter optimization. **Bulletin of Electrical Engineering and Informatics**, v. 13, n. 5, p. 3653–3662, 2024. Citado na página 22.

AHMADI, M. et al. Novel feature extraction, selection and fusion for effective malware family classification. In: **Proceedings of the 6th ACM Conference on Data and Application Security and Privacy**. New York: ACM, 2016. p. 183–194. Citado na página 17.

AL-GHANEM, W. K. et al. Mad-anet: Malware detection using attention-based deep neural networks. **CMES - Computer Modeling in Engineering and Sciences**, v. 143, n. 1, p. 1009–1027, 2025. ISSN 1526-1492. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1526149225000712>. Citado 2 vezes nas páginas 21 e 22.

AVTEST Institute. **AV-Test malware statistics**. 2022. Disponível em: <https://www.av-test.org/en/statistics/malware/>. Citado na página 16.

Canadian Institute for Cybersecurity. **MalMem 2022 - Malware Memory Forensics Dataset**. 2022. <https://www.unb.ca/cic/datasets/malmem-2022.html>. Acesso em: 30 abr. 2025. Citado 3 vezes nas páginas 5, 20 e 21.

CARRIER, T. Detecting obfuscated malware using memory feature engineering. University of New Brunswick, 2021. Citado 2 vezes nas páginas 21 e 22.

CHAUDHARY, M.; MASOOD, A. Realmalsol: real-time optimized model for android malware detection using efficient neural networks and model quantization. **Neural Computing and Applications**, Springer, v. 35, n. 15, p. 11373–11388, 2023. Citado na página 19.

COHEN, F. **Computer Viruses: Theory and Experiments**. Tese (Doutorado) — University of Southern California, 1987. Tese de doutorado que definiu formalmente vírus de computador. Citado na página 16.

Cybersecurity Research Lab – University of New Brunswick. **CIC-MalMem-2022 Dataset**. 2022. <https://www.unb.ca/cic/datasets/malmem-2022.html>. Acesso em: 6 maio 2025. Citado na página 25.

DAMAŠEVIČIUS, R. et al. Ensemble-based classification using neural networks and machine learning models for windows pe malware detection. **Electronics**, MDPI, v. 10, n. 4, p. 485, 2021. Citado na página 18.

DENER, M.; OK, G.; ORMAN, A. Malware detection using memory analysis data in big data environment. **Applied Sciences**, v. 12, n. 17, 2022. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/12/17/8604>. Citado 2 vezes nas páginas 16 e 18.

DIGITAL, O. **Riot sofre ataque hacker: Código-fonte de League of Legends é roubado**. 2023. <https://olhardigital.com.br/2023/01/26/seguranca/riot-sofre-ataque-hacker-codigo-fonte-de-league-of-legends-e-roubado/>. [Online; accessed 20-Fevereiro-2024]. Citado na página 12.

GANDOTRA, E.; BANSAL, D.; SOFAT, S. Malware analysis and classification: A survey. **Journal of Information Security**, Scientific Research Publishing, v. 2014, 2014. Citado na página 16.

GEARHART, F. X. **A Study on the Effectiveness of Interpretable Machine Learning Explanations in Cybersecurity**. Tese (Tese (Doutorado em Tecnologia da Informação)) — Northcentral University, San Diego, CA, 2024. Citado na página 18.

GLOBO. **Hackers invadem Microsoft e roubam parte do código-fonte de produtos da empresa**. 2022. <https://g1.globo.com/tecnologia/noticia/2022/03/23/hackers-invadem-microsoft-e-roubam-parte-do-codigo-fonte-de-produtos-da-empresa.ghtml>. [Online; accessed 20-Fevereiro-2024]. Citado na página 12.

HEMALATHA, J. et al. An efficient densenet-based deep learning model for malware detection. **Entropy**, MDPI, v. 23, n. 3, p. 344, 2021. Citado na página 18.

HOSSAIN, M. A.; ISLAM, M. S. Enhanced detection of obfuscated malware in memory dumps: a machine learning approach for advanced cybersecurity. **Cybersecurity**, Springer, v. 7, n. 1, p. 16, 2024. Citado 2 vezes nas páginas 17 e 18.

HUANG, Z. et al. A survey on machine learning against hardware trojan attacks: Recent advances and challenges. **IEEE Access**, IEEE, v. 8, p. 10796–10826, 2020. Citado na página 17.

HUNTER, J. D. Matplotlib: A 2d graphics environment. **Computing in Science & Engineering**, IEEE Computer Society, v. 9, n. 3, p. 90–95, 2007. Citado na página 29.

KIM, D.; SOLOMON, M. G. **Fundamentals of Information Systems Security**. Burlington, MA: Jones & Bartlett Learning, 2016. Citado na página 16.

LASHKARI, A. H. et al. Volmemlyzer: Volatile memory analyzer for malware classification using feature engineering. In: **2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)**. Piscataway, NJ: IEEE, 2021. p. 1–8. Citado na página 21.

LEONEL, L.; MOLINOS, D.; MIANI, R. Comprehensive ransomware detection: Optimization of feature selection through machine learning algorithms and explainable ai on memory analysis. In: **Anais do Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)**. São Paulo: Sociedade Brasileira de Computação, 2024. Citado 6 vezes nas páginas 5, 13, 14, 22, 23 e 28.

LIAKOS, K. G. et al. Machine learning for hardware trojan detection: A review. In: **2019 Panhellenic Conference on Electronics & Telecommunications (PACET)**. Piscataway, NJ: IEEE, 2019. p. 1–6. Citado na página 13.

MARCONI, M. d. A.; LAKATOS, E. M. **Metodologia científica**. São Paulo: Atlas, 2004. v. 4. Citado na página 23.

MARTIN, G.; KINROSS, J. **WannaCry Global Impact Assessment**. 2019. Relatório técnico sobre o ataque do WannaCry. Citado na página 16.

MAZEIKA, M. et al. The trojan detection challenge. In: **NeurIPS 2022 Competition Track**. New Orleans, LA: Proceedings of Machine Learning Research (PMLR), 2023. p. 279–291. Citado 3 vezes nas páginas 13, 14 e 19.

MOSLI, R. et al. Automated malware detection using artifacts in forensic memory images. In: **Proceedings of the 2016 IEEE Symposium on Technologies for Homeland Security (HST)**. Waltham, MA: IEEE, 2016. p. 1–6. Citado na página 18.

NYHOLM, H. et al. The evolution of volatile memory forensics. **Journal of Cybersecurity and Privacy**, MDPI, v. 2, n. 3, p. 556–572, 2022. Citado na página 19.

Plotly Technologies Inc. **Plotly Open Source Graphing Library**. 2015. Montréal, QC. Available at: <https://plotly.com/>. Citado na página 29.

ROSSUM, G. V.; JR, F. L. D. **The Python Language Reference Manual**. Wilmington, DE, USA: Python Software Foundation, 2009. Citado na página 29.

RUI, L.; GADYATSKAYA, O. Position: The explainability paradox-challenges for xai in malware detection and analysis. In: **Proceedings of the 2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)**. Viena, Áustria: IEEE, 2024. p. 554–561. Citado na página 18.

UCCI, D.; ANIELLO, L.; BALDONI, R. Survey of machine learning techniques for malware analysis. **Computers Security**, v. 81, p. 123–147, 2019. Citado na página 17.

Volatility Foundation. **Volatility Framework**. 2025. Acessado em: 7 mai. 2025. Disponível em: <https://volatilityfoundation.org/>. Citado na página 21.

WADKAR, M.; TROIA, F. D.; STAMP, M. Detecting malware evolution using support vector machines. **Expert Systems with Applications**, Elsevier, v. 143, p. 113022, 2020. Citado 2 vezes nas páginas 20 e 22.

WASKOM, M. L. Seaborn: statistical data visualization. **Journal of Open Source Software**, v. 6, n. 60, p. 3021, 2021. Citado na página 29.

WAZLAWICK, R. S. **Metodologia de pesquisa para ciência da computação**. São Paulo: Elsevier, 2009. v. 2. Citado na página 23.

ZHANG, L.; WANG, H. Analysis of the iloveyou worm impact. In: **Proceedings of the IEEE Security Symposium**. Washington, DC: IEEE, 2018. p. 45–60. Estimativa de danos: US$15 bilhões. Citado na página 16.