Uma Abordagem Interpretável para a Detecção de Ransomware Baseada no Consenso de SHAP

Waldemar Patrique Flores Silva

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

Monte Carmelo - MG
2026

Waldemar Patrique Flores Silva

Uma Abordagem Interpretável para a Detecção de Ransomware Baseada no Consenso de SHAP

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: Diego Nunes Molinos

Monte Carmelo - MG
2026

Aos meus familiares, pelo apoio incondicional ao longo desta jornada, e aos meus pais (in memoriam), Estefania Maria Flores da Cunha e Leonardo Resende da Silva, cuja presença permanece viva em tudo que sou e conquisto.

Agradecimentos

Agradeço ao Professor Dr. Diego Nunes Molinos, pela orientação, disponibilidade e contribuições essenciais ao desenvolvimento deste trabalho. Agradeço também a todos os colegas e amigos que, direta ou indiretamente, contribuíram para a realização desta pesquisa.

"Existem dois tipos de empresas: as que já foram hackeadas e as que ainda não sabem que foram."

John Chambers (ex-CEO da Cisco)

Resumo

Este trabalho investiga a detecção de malware do tipo ransomware a partir de artefatos extraídos da memória utilizando técnicas de machine learning. Foi utilizado o dataset *Canadian Institute for Cybersecurity Malware Memory Dataset 2022 (CIC-MalMem-2022)*, que contém informações comportamentais de processos benignos e de variantes de ransomware. Inicialmente, realizou-se um pré-processamento dos dados, incluindo o balanceamento das classes, a remoção de atributos irrelevantes e a eliminação de redundâncias entre características altamente correlacionadas. Em seguida, quatro algoritmos supervisionados: *Decision Tree, Random Forest, K-Nearest Neighbors (KNN) e AdaBoost*, foram avaliados por meio de validação cruzada estratificada de 10 partições, utilizando métricas como acurácia, precisão, recall, F1-score e matrizes de confusão. Posteriormente, aplicou-se a técnica de explicabilidade *SHapley Additive exPlanations (SHAP)* para identificar a importância das características utilizadas pelos modelos. Com base nessa análise, foi gerado um conjunto reduzido de atributos por meio de consenso entre os modelos, permitindo o retreinamento e a comparação de desempenho. Os resultados demonstraram que o conjunto reduzido manteve desempenho equivalente ao conjunto completo, alcançando acurácia superior a 99,9% em todos os modelos avaliados. Conclui-se que a seleção de atributos baseada na explicabilidade permite reduzir a dimensionalidade dos dados sem comprometer a capacidade de detecção, favorecendo soluções mais leves e interpretáveis para a detecção de ransomware.

Palavras-chave: Ransomware, Machine Learning, Detecção, Memória, Análise.

Lista de ilustrações

Lista de tabelas

Lista de siglas

API Application Programming Interface

AUC Area Under the Curve

CIC-MalMem-2022 Canadian Institute for Cybersecurity Malware Memory Dataset 2022

CVSS Common Vulnerability Scoring System

DT Decision Tree

FN Falsos Negativos

FP Falsos Positivos

FPR False Positive Rate

KNN K-Nearest Neighbors

MBR Master Boot Record

ML Machine Learning

OOB Out-Of-Bag Error

RAM Random Access Memory

ROC Receiver Operating Characteristic

RF Random Forest

RQ Research Question

SHAP SHapley Additive exPlanations

VN Verdadeiros Negativos

VP Verdadeiros Positivos

XAI Explainable Artificial Intelligence

XGBoost Extreme Gradient Boosting

Sumário

Capítulo 1

Introdução

Com o crescimento do uso de tecnologias digitais e da interconexão entre sistemas computacionais, as ameaças cibernéticas têm se tornado cada vez mais sofisticadas e impactantes. Entre elas, destaca-se o *ransomware*, um tipo de *malware* que criptografa os dados das vítimas, exigindo pagamento para sua liberação e, além disso, alguns *ransomwares* mais avançados vão além e exfiltram os dados da vítima, utilizando-os para chantagem sob a ameaça de divulgação na *deepweb*, o que acaba ferindo a confiabilidade e a credibilidade da empresa atacada.

Em 2024, os ataques de *ransomware* alcançaram um recorde de 5.414 incidentes globais, um aumento de 11% em relação a 2023, (poderTech, 2024). Além disso, houve um crescimento de 90% no número de vítimas extorquidas publicamente globalmente por esses ataques (Check Point Software Technologies Ltd., 2024). A gravidade do problema é intensificada pela evolução das técnicas empregadas pelos cibercriminosos.

Variantes como o *LockBit 3.0* utilizam estratégias avançadas de ofuscação, falhas *zero-day*, modularização, o que dificulta a detecção por métodos tradicionais (CISA, 2023). Nesse contexto, a análise de binários de memória surge como uma abordagem promissora, pois permite identificar comportamentos dinâmicos do *ransomware*, como injeções de código e chamadas de *Application Programming Interface (API)* suspeitas, que frequentemente escapam às técnicas de detecção estática

Nos últimos anos, técnicas de aprendizado de máquina, *Machine Learning (ML)*, têm sido amplamente empregadas na detecção de *malware*, pois apresentam maior capacidade de generalização quando comparadas a métodos baseados em assinaturas. Enquanto soluções tradicionais dependem da identificação de padrões previamente conhecidos, modelos de aprendizado de máquina conseguem identificar comportamentos maliciosos mesmo em variantes ainda não catalogadas (ALRAIZZA; ALGARNI, 2023; ALZAHRANI et al., 2025). No entanto, esses métodos frequentemente utilizam um grande número de características extraídas dos artefatos analisados, o que pode aumentar significativamente o custo computacional do sistema. Embora isso nem sempre afete diretamente métricas de desempenho como acurácia ou precisão, o elevado volume de atributos pode impactar

a eficiência do processamento, dificultando a aplicação dessas técnicas em cenários que exigem detecção rápida ou em tempo real (LEONEL; MOLINOS; MIANI, 2024).

A proposta deste estudo consiste em avaliar o desempenho de diferentes algoritmos de aprendizado de máquina aplicados à classificação de processos benignos e maliciosos a partir de características extraídas da memória. Para isso, são analisadas métricas de desempenho que permitem verificar a eficácia dos modelos na identificação desse tipo de ameaça. Além disso, o trabalho investiga a redução do conjunto de atributos por meio de métodos estatísticos e técnicas de explicabilidade baseadas em SHAP, com o objetivo de identificar as características mais relevantes para a classificação e otimizar o desempenho computacional dos modelos.

1.1 Motivação

A detecção de *ransomware* permanece um dos maiores desafios da segurança da informação. Técnicas avançadas de ofuscação, modularização e camuflagem permitem que variantes modernas operem de forma silenciosa, dificultando sua identificação por ferramentas tradicionais, como os antivírus baseados em assinaturas (CISA, 2023).

A maioria das soluções atuais foca em características estáticas, como assinaturas binárias ou extensões de arquivo, que podem ser facilmente burladas por pequenas modificações no código do *ransomware*. Métodos dinâmicos, que monitoram o comportamento em tempo de execução, ainda são pouco utilizados, especialmente no contexto de *ransomwares*, devido à velocidade de atuação do *malware*. O aprendizado de máquina é promissor nesse cenário, pois pode identificar padrões complexos em dados dinâmicos.

Entretanto, a aplicação de técnicas de aprendizado de máquina nesse contexto também apresenta desafios relevantes. *Datasets* utilizados para detecção de *malware*, como *dataset CIC-Malmem-2022* (CARRIER et al., 2022) , frequentemente contêm um grande número de atributos extraídos de artefatos de memória, incluindo métricas de processos, chamadas de API e informações de sistema. Embora esse elevado volume de características possa enriquecer a representação dos dados e favorecer a capacidade de aprendizado dos modelos, ele também pode aumentar o custo computacional do treinamento e da inferência, além de introduzir redundâncias e atributos pouco relevantes para a classificação.

Em cenários que exigem respostas rápidas, essa complexidade pode comprometer a eficiência da solução. Nesse sentido, técnicas de seleção de atributos tornam-se importantes para reduzir a dimensionalidade dos dados sem comprometer o desempenho dos modelos. Entre essas abordagens, métodos baseados em explicabilidade, como o SHAP, têm se destacado por permitir a identificação das características mais relevantes para a tomada de decisão dos modelos, contribuindo tanto para a otimização computacional quanto para uma melhor interpretação dos resultados.

1.2 Objetivos

O objetivo geral deste trabalho é desenvolver e avaliar modelos de detecção de *ransomware* baseados em aprendizado de máquina a partir da análise de artefatos extraídos da memória, investigando estratégias de seleção e redução de atributos por meio de métodos estatísticos e técnicas de explicabilidade baseadas em SHAP, visando identificar as características mais relevantes para a classificação e melhorar a eficiência dos modelos.

1.2.1 Objetivos Específicos

- ❏ Realizar o pré-processamento dos dados do *dataset* CIC-Malmem-2022, incluindo a filtragem de classes, o balanceamento e a preparação das características utilizadas pelos modelos.
- ❏ Treinar e avaliar diferentes algoritmos de aprendizado de máquina para a classificação de processos benignos e maliciosos.
- ❏ Analisar o desempenho dos modelos por meio de métricas de avaliação, como acurácia, precisão, *recall*, *F1-score* e matrizes de confusão.
- ❏ Investigar técnicas de seleção e redução de atributos baseadas em métodos estatísticos e na análise de importância de características por meio do SHAP).
- ❏ Avaliar o impacto da redução do conjunto de atributos no desempenho e na eficiência computacional dos modelos de classificação.

1.3 Questões de Pesquisa

Este trabalho é guiado por duas questões de pesquisa (*Research Question (RQ)*):

- ❏ RQ1: A utilização de técnicas de explicabilidade baseadas em SHAP pode auxiliar na identificação das características mais relevantes para a detecção de *ransomware*, contribuindo para modelos mais eficientes e interpretáveis?;
- ❏ RQ2: A redução do conjunto de atributos, realizada por meio de métodos estatísticos e análise de importância baseada em SHAP, é capaz de manter o desempenho dos modelos de aprendizado de máquina na detecção de *ransomware*?

1.4 Contribuições

Este trabalho apresenta as seguintes contribuições principais:

- Avaliação comparativa do desempenho de quatro algoritmos de aprendizado de máquina (*Decision Tree*, *Random Forest*, *KNN* e *AdaBoost*) para a classificação de processos benignos e maliciosos (*ransomware*);
- Investigação da importância das características utilizadas pelos modelos por meio de técnicas de explicabilidade baseadas em SHAP, permitindo identificar os atributos mais relevantes para a detecção de *ransomware*.
- Construção e avaliação de um conjunto reduzido de atributos, com base na análise de importância e no consenso entre modelos, para otimizar a eficiência computacional dos classificadores, sem comprometer o desempenho de detecção.

1.5 Organização da Monografia

Esta monografia está estruturada da seguinte forma:

- **Capítulo 1 – Introdução**: Apresenta o contexto do problema, motivação, objetivos e a proposta geral do trabalho.
- **Capítulo 2 – Fundamentação Teórica**: Aborda os conceitos fundamentais de *ransomware*, análise de memória e aprendizado de máquina, além de revisões de trabalhos relacionados.
- **Capítulo 3 – Metodologia**: Descreve o processo de extração de características, os algoritmos utilizados, o *dataset CIC-MalMem-2022* e o procedimento de treinamento e validação dos modelos.
- **Capítulo 4 – Resultados e Discussões**: Apresenta os resultados obtidos, análises das métricas de desempenho e discussões sobre os achados.
- **Capítulo 5 – Conclusão e Trabalhos Futuros**: Resume as conclusões obtidas, limitações do estudo e sugestões para pesquisas futuras.

Capítulo 2

Fundamentação Teórica

A fundamentação teórica deste trabalho reúne os principais fundamentos teóricos necessários à compreensão da detecção de *ransomware* por meio da análise de memória, com o apoio de técnicas de aprendizado de máquina, do inglês, *machine learning*. Inicialmente, aborda-se o conceito de *ransomware*, suas principais características e formas de atuação em sistemas computacionais. Em seguida, discute-se a análise de memória como estratégia *forense* voltada à identificação de evidências voláteis e de comportamentos maliciosos em tempo de execução. Na sequência, apresentam-se as características dinâmicas e os atributos extraídos da memória, que servem de base para a modelagem computacional do problema. Por fim, são introduzidos os princípios do aprendizado de máquina supervisionado, com destaque para os algoritmos empregados neste estudo, que permitem a classificação de comportamentos benignos e maliciosos a partir dos dados coletados.

2.0.1 *Ransomware*

A palavra *ransomware* é a junção de dois termos em inglês: *ransom*, que significa `resgate`, e *malware*, que se refere a um tipo de `software malicioso`, projetado para causar danos ou obter acesso não autorizado a sistemas computacionais. O *ransomware* é um tipo de ameaça cibernética que restringe o acesso aos arquivos ou até mesmo ao sistema operacional como um todo, exigindo o pagamento de um resgate para restaurar esse acesso (SILVA, 2021).

De acordo com Anghel e Racautanu (2019), a Tabela 1 apresenta os principais tipos de *ransomware*. Cada categoria utiliza diferentes estratégias para extorquir a vítima, que variam desde a criptografia de arquivos até o bloqueio completo do sistema ou a ameaça de vazamento de dados sensíveis.

Tabela 1 – Principais tipos de ransomware

Tipo	Descrição	Exemplos
Crypto Ransomware	Criptografa arquivos da vítima utilizando algoritmos criptográficos. Em variantes modernas, pode também realizar exfiltração de dados e aplicar esquemas de dupla extorsão, ameaçando a divulgação das informações.	CryptoLocker, WannaCry, Locky, Maze, REvil
Leakware (Doxware)	Não bloqueia diretamente o sistema ou os arquivos, mas coleta informações sensíveis da vítima e ameaça divulgá-las caso o resgate não seja pago.	Maze, REvil
Mobile Ransomware	Variante direcionada a dispositivos móveis, como smartphones e tablets, geralmente bloqueando a interface do dispositivo ou exibindo mensagens falsas de autoridades.	Simplocker, LockerPin
MBR Ransomware	Ataca o *Master Boot Record (MBR)*, impedindo o sistema de inicializar corretamente até que o resgate seja pago.	Petya

Fonte: Elaborado pelo autor (2026).

As infecções por *ransomware* frequentemente ocorrem por meio de campanhas de *phishing*, nas quais as vítimas são levadas a executar arquivos maliciosos por meio de *engenharia social* (ARORA, 2024). No entanto, nem sempre há necessidade de interação humana: atacantes também exploram vulnerabilidades do tipo *zero-day* para infiltrar-se em sistemas. Um exemplo notório é a vulnerabilidade `CVE-2023-4966`, conhecida como *Citrix Bleed*, de severidade crítica (pontuação Common Vulnerability Scoring System (CVSS) de 9,4). Essa falha afeta versões específicas dos produtos NetScaler ADC e NetScaler Gateway, frequentemente utilizados para otimizar o desempenho e a segurança em ambientes corporativos. Um caso emblemático de sua exploração ocorreu no ataque à empresa Boeing, no qual o grupo utilizou a vulnerabilidade como vetor de acesso inicial, resultando no roubo e na criptografia de dados sensíveis da organização (YUCEEL, 2023).

2.0.2 Análise de Binários de Memória

A análise forense de memória, ou *memory forensics*, refere-se ao processo de investigação de dados voláteis armazenados na memória RAM[1] de um sistema computacional.

[1] RAM (*Random Access Memory*) é uma memória volátil utilizada para armazenar temporariamente dados e instruções que estão sendo processados pelo sistema.

Essa técnica permite identificar atividades maliciosas que podem não deixar vestígios permanentes no disco rígido (LIGH et al., 2014b).

2.0.2.1 Dados Voláteis

Dados voláteis correspondem a informações armazenadas temporariamente na memória principal de um sistema computacional. Esses dados permanecem disponíveis enquanto o dispositivo está em funcionamento, mas são perdidos quando o sistema é desligado ou reiniciado e quando os processos ou *threads* são finalizados. Devido a essa característica, a memória volátil contém registros transitórios de atividades do sistema, como processos em execução, conexões de rede e dados manipulados pelas aplicações (Digital Guardian, 2017).

2.0.2.2 Dados Voláteis em Segurança

A manipulação e a análise de dados voláteis em segurança da informação são realizadas principalmente por meio do despejo de memória, também conhecido como *core dump* ou *memory dump*. Esse procedimento consiste na captura do conteúdo armazenado na memória Random Access Memory (RAM) de um sistema em determinado instante (ZHOU et al., 2015).

A imagem de memória, chamada de *Footprint*, permite que analistas examinem detalhadamente o estado do sistema no momento da coleta, o que possibilita a identificação de códigos maliciosos ou de processos suspeitos. Ao preservar essas informações antes que sejam perdidas, por exemplo, em decorrência do desligamento ou da reinicialização do sistema, o despejo de memória fornece uma base confiável para a condução de investigações posteriores (Digital Guardian, 2017; ZHOU et al., 2015).

2.0.3 Características Dinâmicas e Extração de Atributos

A detecção de atividades maliciosas diretamente na memória está fortemente associada à análise de atributos observados durante a execução de um programa. Entre os principais indicadores utilizados nesse processo estão chamadas suspeitas de API do sistema, padrões incomuns de alocação ou manipulação de memória, presença de seções executáveis injetadas, o que pode indicar técnicas de injeção de código e níveis elevados de entropia, frequentemente associados a conteúdos criptografados ou mecanismos de ofuscação (OR-MEIR et al., 2019).

A extração dessas características normalmente é realizada por meio de ferramentas especializadas de análise de memória. Soluções amplamente utilizadas incluem *frameworks* como *Volatility* (Volatility Foundation, 2018) e *Rekall* (Rekall Forensics, 2014), que permitem analisar artefatos na memória e identificar comportamentos suspeitos. Além dessas ferramentas, podem ser empregadas abordagens personalizadas, como a implementação

de *hooks* em API do sistema operacional, o que possibilita o monitoramento detalhado das interações realizadas por processos em execução.

Em cenários que exigem detecção em tempo real, são utilizados sistemas de monitoramento contínuo capazes de interceptar eventos potencialmente maliciosos e de aplicar mecanismos automatizados de análise comportamental (LAN; YUAN, 2019). A obtenção eficiente desses dados desempenha um papel fundamental no treinamento de modelos de aprendizado de máquina, permitindo a construção de sistemas de detecção mais robustos, capazes de identificar ameaças de forma proativa e com maior precisão.

2.0.4 Aprendizado de Máquina Supervisionado

O aprendizado de máquina supervisionado é uma abordagem baseada em dados rotulados, em que o modelo é treinado para aprender padrões a partir de exemplos previamente rotulados. Entre os principais paradigmas estão a classificação, que associa entradas a categorias discretas, e a regressão, que estima valores contínuos, sendo o enfoque na classificação, dado o objetivo de distinguir entre comportamentos benignos e maliciosos (SHALEV-SHWARTZ; BEN-DAVID, 2014).

O processo de aprendizado supervisionado envolve múltiplas etapas:

- **Pré-processamento:** normalização de atributos, tratamento de valores ausentes e codificação de categorias;
- **Seleção de atributos:** identificação das variáveis mais relevantes, reduzindo a dimensionalidade e o ruído (LIU; MOTODA, 1998);
- **Treinamento:** ajuste dos parâmetros do modelo com base nos dados de entrada e seus respectivos rótulos;
- **Validação e teste:** avaliação da generalização do modelo, geralmente por meio de validação cruzada e de um conjunto de testes separado (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

2.0.4.1 Árvores de Decisão - *Decision Tree*

As árvores de decisão são algoritmos que particionam recursivamente o espaço de atributos com base em critérios de pureza, como o índice de Gini ou a entropia de Shannon. Cada nó interno representa uma decisão baseada em um atributo, enquanto os nós folha indicam a classe atribuída. A interpretabilidade e simplicidade de implementação tornam essa técnica uma escolha frequente em problemas de detecção de anomalias (QUINLAN, 1986).

2.0.4.2 *Random Forest*

O algoritmo *Random Forest* consiste em um conjunto (*ensemble*) de árvores de decisão treinadas sobre subconjuntos aleatórios dos dados e dos atributos, uma técnica conhecida como *bagging* (bootstrap aggregating). Ao combinar os resultados de cada árvore, essa abordagem reduz a variância e melhora a robustez, mitigando o sobreajuste comum em modelos baseados em uma única árvore (BREIMAN, 2001).

2.0.4.3 *K-Nearest Neighbors (KNN)*

O algoritmo KNN baseia-se na proximidade entre instâncias, considerando que amostras próximas no espaço de atributos pertencem à mesma classe. A escolha do parâmetro K influencia diretamente a sensibilidade do modelo: valores baixos aumentam o risco de ruído, enquanto valores altos podem suavizar excessivamente as decisões. As medidas de distância mais utilizadas incluem a euclidiana, a Manhattan e a Minkowski (COVER; HART, 1967).

2.0.4.4 *AdaBoost*

O *Adaptive Boosting (AdaBoost)*, proposto por Freund e Schapire (1997), é um algoritmo de *ensemble* baseado em *boosting*, no qual múltiplos classificadores fracos são combinados sequencialmente. A cada iteração, o modelo é treinado para corrigir os erros cometidos pelos classificadores anteriores, aumentando o peso das amostras classificadas incorretamente. Dessa forma, o algoritmo direciona progressivamente sua atenção para instâncias mais difíceis de classificar.

2.0.5 Explicabilidade de Modelos de Aprendizado de Máquina

Com o avanço do uso de modelos de aprendizado de máquina em aplicações críticas, a capacidade de compreender e interpretar as decisões desses modelos tornou-se um aspecto fundamental. Muitos algoritmos modernos, especialmente modelos de *ensemble* e redes neurais profundas, apresentam alto desempenho preditivo, porém operam frequentemente como caixas-pretas (*black-box*), dificultando a compreensão dos fatores que influenciam suas decisões (NEIVA, 2023).

Nesse contexto, surge o campo da explicabilidade em aprendizado de máquina (*Explainable Artificial Intelligence – XAI)*, cujo objetivo é tornar os modelos mais transparentes e interpretáveis para humanos. Métodos de explicabilidade permitem identificar quais atributos exercem maior influência nas predições do modelo, além de possibilitar a análise de como diferentes características contribuem para a decisão final (ARRIETA et al., 2020). Isso é particularmente relevante em domínios sensíveis, como segurança da informação, saúde e sistemas financeiros, onde a justificativa das decisões automatizadas é essencial para confiança e validação dos resultados.

No contexto deste trabalho, técnicas de explicabilidade são empregadas para analisar quais atributos do conjunto de dados contribuem de forma mais significativa para a identificação de comportamentos maliciosos, permitindo não apenas avaliar o desempenho do modelo, mas também compreender os fatores que influenciam suas decisões.

2.0.5.1 *SHapley Additive exPlanations - SHAP*

O SHAP é um método de explicabilidade baseado na teoria dos jogos cooperativos, especificamente nos valores de Shapley, proposto por Lundberg e Lee (2017). O método atribui a cada característica (atributo) do modelo um valor de importância que quantifica sua contribuição marginal para a predição final, levando em consideração todas as possíveis combinações de atributos.

Os valores SHAP possuem propriedades matemáticas desejáveis que os diferenciam de outros métodos de explicabilidade. São **aditivos**: a soma dos valores SHAP de todos os atributos para uma predição específica é igual à diferença entre a predição do modelo e a predição base (valor esperado). São **consistentes**: se um modelo é alterado de modo que a contribuição marginal de um atributo aumenta ou permanece a mesma para qualquer subconjunto de atributos, o valor SHAP desse atributo não diminui. E são **localmente fiéis**: explicam com precisão a predição individual de cada amostra, capturando interações complexas entre atributos (LUNDBERG; LEE, 2017).

Justificativa para o uso do SHAP neste trabalho: A escolha do SHAP como método de explicabilidade se fundamenta em três aspectos principais. Primeiro, a **fundamentação teórica sólida** baseada em valores de Shapley garante propriedades matemáticas rigorosas de consistência, aditividade e fidelidade local, diferenciando-o de métodos heurísticos. Segundo, a **aplicabilidade agnóstica ao modelo** permite analisar de forma uniforme e comparável os quatro algoritmos utilizados neste trabalho (*Decision Tree*, *Random Forest*, *KNN* e *AdaBoost*), mesmo que seus paradigmas de aprendizado sejam fundamentalmente distintos. Terceiro, a **capacidade de capturar interações não-lineares** entre atributos é essencial para identificar padrões comportamentais complexos presentes em processos de *ransomware*, que frequentemente envolvem combinações específicas de características de memória operando em conjunto.

No contexto de detecção de *ransomware*, o SHAP permite identificar quais artefatos comportamentais extraídos da memória, como o número de serviços registrados, a quantidade de DLLs carregadas ou *handles* de sincronização , que exercem maior influência nas decisões do modelo. Essa interpretabilidade é fundamental tanto para validar o comportamento do classificador quanto para orientar melhorias futuras na engenharia de atributos e na seleção de características relevantes para detecção em tempo real.

2.1 Trabalhos Correlatos

Com o avanço das ameaças cibernéticas, especialmente do *ransomware*, surgiram diversas propostas de detecção baseadas em análise de memória e em aprendizado de máquina. A seguir, apresentam-se os principais trabalhos correlatos, com ênfase nas técnicas, nos resultados e nas contribuições relevantes para a área.

Comprehensive Ransomware Detection: Optimization of Feature Selection through Machine Learning Algorithms and Explainable AI on Memory Analysis

Este trabalho apresenta um método de detecção de *ransomware* baseado na análise de memória, aliado a algoritmos de *Machine Learning* e técnicas de *Inteligência Artificial Explicável (XAI)*. Utilizando o *dataset CIC-Malmem-2022*, foram aplicadas técnicas de seleção de características, reduzindo o número de atributos de 55 para apenas 5, o que diminui significativamente o ruído e mantém alta acurácia. Os algoritmos empregados foram *Decision Tree, SVM, Naive Bayes* e redes neurais artificiais, combinados com o método SHAP para tornar as decisões mais interpretáveis. Os resultados demonstraram que a otimização de atributos melhora a eficiência, a precisão e a transparência dos sistemas de detecção de ransomware, reforçando a importância da análise de memória para a segurança cibernética moderna.(LEONEL; MOLINOS; MIANI, 2024)

Ransomware Detection using Process Memory

Neste estudo, os autores propuseram uma abordagem de detecção baseada na análise dinâmica da memória de processos em execução. Para isso, foi utilizado o ambiente virtual Cuckoo Sandbox para coletar informações comportamentais de diferentes regiões da memória, como áreas de código, heap[2] e DLLs[3]. A partir dos dados extraídos, características de baixo nível foram utilizadas no treinamento de diversos algoritmos de aprendizado de máquina. Os modelos alcançaram acurácia entre 81,38% e 96,26% na distinção entre processos benignos e maliciosos, evidenciando que a análise comportamental da memória, independentemente de APIs de alto nível, constitui uma estratégia viável e eficaz para a detecção precoce de *ransomware*.(SINGH; IKUESAN; VENTER, 2022)

Ransomware Detection Based on Machine Learning Using Memory Features

Um sistema de aprendizado de máquina foi desenvolvido para detectar *ransomware* a partir de *dumps* de memória. Para isso, foi criado um novo conjunto de dados contendo amostras dinâmicas de famílias recentes de *ransomware* (como *Revil, LockBit e BlackCat*) e de aplicativos benignos, obtidas via Cuckoo Sandbox. Após a coleta, diversos classificadores foram treinados, destacando-se o *Extreme Gradient Boosting (XGBoost)*,

[2] Região da memória utilizada para alocação dinâmica durante a execução do programa, permitindo que estruturas de dados sejam criadas e destruídas conforme necessário.

[3] *Dynamic Link Libraries*: bibliotecas de código reutilizável carregadas dinamicamente pelos programas em tempo de execução, contendo funções e recursos compartilhados.

que apresentou o melhor desempenho utilizando apenas 47 das 58 características extraídas. O modelo atingiu 97,85% de precisão e uma taxa de falsos positivos de apenas 2%, mostrando que as características obtidas dos registros de memória permitem detectar variantes desconhecidas de *ransomware* com alta eficácia.(ALJABRI et al., 2024)

Enhanced Detection of Obfuscated Malware in Memory Dumps: A Machine Learning Approach for Advanced Cybersecurity

Outra proposta apresenta uma solução robusta e diversificada para a identificação e análise de *malwares* ofuscados em capturas de memória. O estudo empregou um conjunto de dados com *dumps*[4] de memória benignos e maliciosos, incluindo amostras de *ransomware*. Foram aplicadas técnicas de pré-processamento, balanceamento de classes com o algoritmo *SMOTE*, e métodos de seleção de atributos baseados em testes qui-quadrado e análise de correlação para isolar as características mais relevantes. Em seguida, um classificador do tipo *ensemble* foi utilizado para distinguir processos maliciosos e benignos. O modelo alcançou uma acurácia superior a 99% na detecção de *ransomware* e outros *malwares* ofuscados, com destaque para o uso de técnicas de Explainable Artificial Intelligence (XAI) que aumentam a transparência das decisões.(HOSSAIN; ISLAM, 2024)

Enhancing ransomware defense: deep learning-based detection and family-wise classification of evolving threats

Apostando em técnicas de *Deep Learning*, foi desenvolvido o modelo GN-BiLSTM (*Group Normalized Bidirectional Long Short-Term Memory*) para a detecção e classificação de variantes de *ransomware* a partir de *dumps* de memória. A validação foi realizada utilizando o dataset *CIC-MalMem-2022*, amplamente citado na literatura. O GN-BiLSTM alcançou aproximadamente 99,99% de precisão na detecção, além de elevados índices de acurácia na classificação por categoria e família do *malware*. O estudo evidencia o potencial das redes neurais profundas na análise de memória volátil, superando abordagens tradicionais e demonstrando eficácia na detecção em tempo real de novas variantes de *ransomware* (HUSSAIN et al., 2024).

[4] Snapshots do estado da memória de um processo ou sistema em um determinado instante, contendo dados brutos que podem ser analisados para identificar comportamentos, estruturas e artefatos de execução.

Tabela 2 – Comparação entre os trabalhos correlatos e a proposta deste estudo

Trabalho	**Base de Dados**	**Técnicas Utilizadas**	**Diferencial**
Comprehensive Ransomware Detection (SBSEG)	CIC-MalMem-2022	ML + Seleção de Features + SHAP	Redução significativa de atributos mantendo desempenho
Ransomware Detection using Process Memory	Dados via Cuckoo Sandbox	Análise de memória + ML tradicional	Análise comportamental de baixo nível independente de APIs
Ransomware Detection Based on ML Using Memory Features	Dataset próprio (Cuckoo)	ML (XGBoost) + Seleção de Features	Baixa taxa de falsos positivos e generalização para variantes
Enhanced Detection of Obfuscated Malware	Dumps de memória diversos	ML + SMOTE + Seleção de atributos + XAI	Detecção robusta de malware ofuscado
Enhancing Ransomware Defense (GN-BiLSTM)	CIC-MalMem-2022	Deep Learning (BiLSTM)	Classificação por família com redes neurais profundas
Proposta deste trabalho	CIC-MalMem-2022 (filtrado)	ML + SHAP + Consenso de Features	**Extração de um consenso global de atributos relevantes via SHAP**

Fonte: Elaborado pelo autor (2026).

Capítulo 3

Método

Este capítulo descreve o método adotado para avaliar os modelos de aprendizado de máquina aplicados à detecção de *ransomware* a partir de artefatos extraídos da memória. São apresentados o *dataset* utilizado, as etapas de pré-processamento, a engenharia de atributos baseada em análise SHAP, os algoritmos empregados com suas respectivas configurações, e as métricas utilizadas na avaliação experimental.

3.1 Dataset

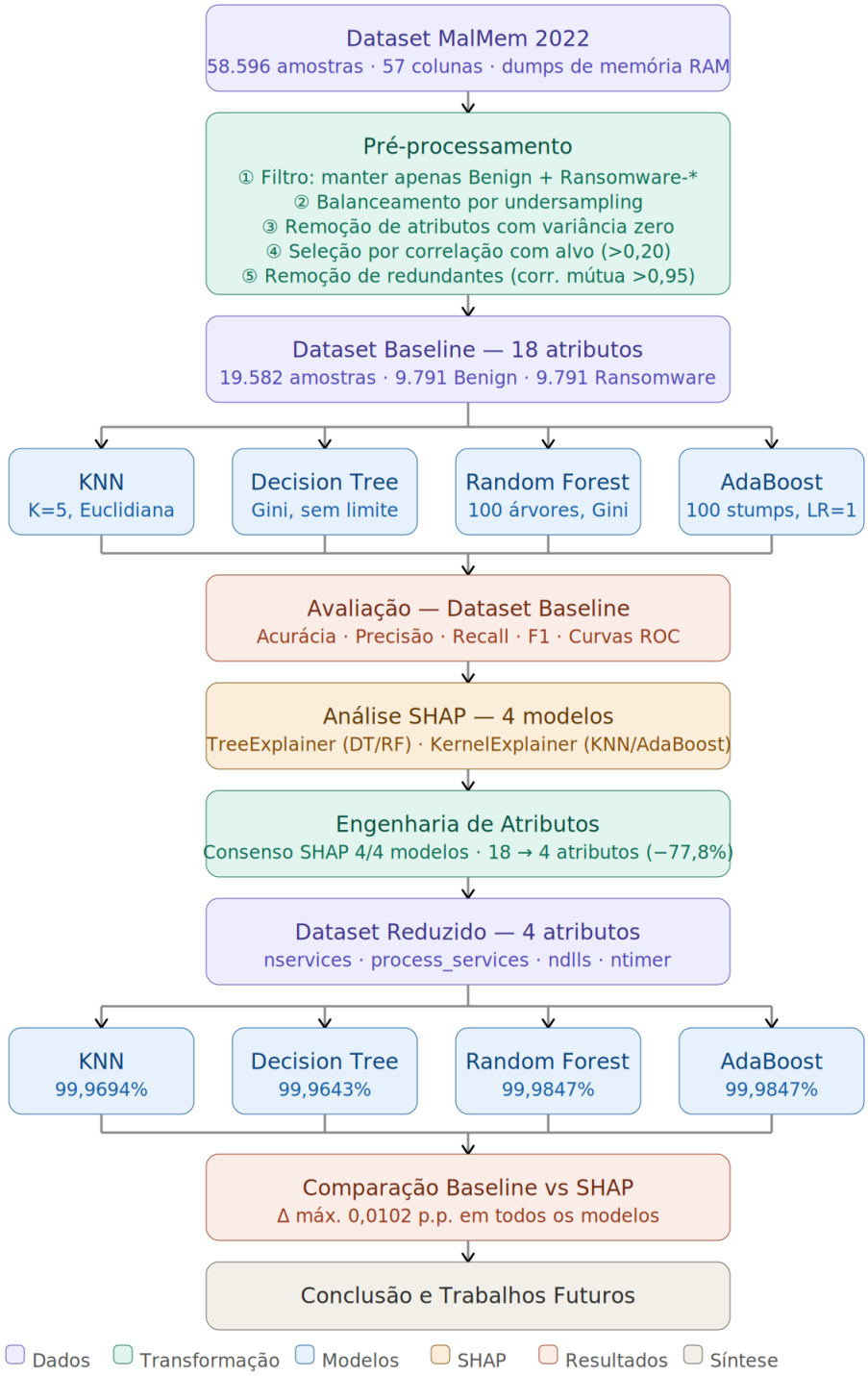
Os experimentos foram conduzidos sobre o conjunto de dados *CIC-Malmem-2022* (CARRIER et al., 2022), disponibilizado pelo *Canadian Institute for Cybersecurity*. O dataset contém amostras extraídas de *dumps* de memória RAM por meio da ferramenta *Volatility* (Volatility Foundation, 2018), produzindo 57 atributos comportamentais por processo analisado. Esses atributos descrevem características de execução observáveis em memória, como contagem de processos, *handles*, serviços, módulos carregados e regiões de memória suspeitas, sem qualquer dependência de assinaturas estáticas ou nomes de arquivos.

O conjunto original contém amostras de diversas categorias de *malware*, incluindo *spyware*, *trojans*, *adware* e *ransomware*, além de processos benignos. Para este trabalho, foram consideradas exclusivamente as amostras das categorias *Benign* e *Ransomware-**, com foco no problema de detecção binária de *ransomware*. Todos os subtipos de *ransomware* presentes no *dataset* foram unificados sob o rótulo *Malware*, preservando a distinção em relação aos processos benignos.

3.2 Pré-processamento

O pré-processamento foi realizado em cinco etapas sequenciais, aplicadas sobre o conjunto filtrado de amostras benignas e de *ransomware*. A Figura 1 ilustra o fluxo completo do pipeline.

Figura 1 – Fluxo do pipeline de pré-processamento aplicado ao dataset *CIC-MalMem-2022*.



Fonte: Elaborado pelo autor (2026).

3.2.1 Remoção de Colunas de Metadados

Quatro colunas foram removidas por não representarem atributos comportamentais extraídos da memória: `Category` (rótulo categórico original com o tipo de malware), `pslist.nprocs64bit` (contagem de processos de 64 bits, altamente correlacionada com outras contagens de processo), `handles.nport` (informação de rede, fora do escopo comportamental) e `svcscan.interactive_process_services` (atributo com variância praticamente nula na partição analisada). Essa etapa é importante para evitar que variáveis de identificação ou de baixa informatividade influenciem indevidamente o aprendizado dos modelos.

3.2.2 Balanceamento de Classes

Após a filtragem, o *dataset* apresentava desbalanceamento entre as classes. Para garantir que os modelos não desenvolvessem viés em favor da classe majoritária, foi aplicada a técnica de *undersampling* , a classe majoritária foi amostrada aleatoriamente até igualar o tamanho da classe minoritária, com semente fixa (`random_state = 42`) para reprodutibilidade. O conjunto balanceado resultante contém **19.582 amostras**: 9.791 benignas e 9.791 de *ransomware*.

3.2.3 Remoção de Atributos com Variância Zero

Atributos cujo desvio padrão é igual a zero em todo o conjunto de dados não carregam qualquer informação discriminativa, apresentam o mesmo valor para todas as amostras, independentemente da classe. Esses atributos foram identificados e removidos automaticamente. Na partição resultante do filtro anterior, nenhum atributo com variância zero foi encontrado, de modo que esta etapa não eliminou colunas adicionais, mas foi mantida no pipeline para garantir robustez em execuções futuras com *datasets* distintos.

3.2.4 Seleção por Correlação com o Alvo

Foram retidos apenas os atributos cuja correlação de Pearson absoluta com a variável alvo, codificada numericamente como 0 para *Benign* e 1 para *Malware*, fosse superior ao limiar $\tau_{alvo} = 0{,}20$.

Justificativa do limiar de correlação 0,20: O limiar $\tau_{alvo} = 0{,}20$ foi estabelecido com base em critérios estatísticos e práticos consolidados na literatura sobre seleção de atributos. Do ponto de vista estatístico, correlações absolutas inferiores a 0,20 são geralmente classificadas como "muito fracas" ou "negligenciáveis" segundo as escalas de interpretação de Cohen (1988) e de Evans (1996), indicando que a relação linear entre o atributo e a variável alvo é insuficiente para contribuir de forma significativa à capacidade discriminativa do modelo. Do ponto de vista prático, a inclusão de atributos com

correlação muito baixa tende a introduzir ruído nos dados sem agregar poder preditivo, aumentando desnecessariamente a dimensionalidade do espaço de entrada e, consequentemente, o custo computacional do treinamento e da inferência (LIU; MOTODA, 1998). Em *datasets* balanceados como o utilizado neste trabalho, um limiar de 0,20 representa um ponto de equilíbrio entre a retenção de informação potencialmente relevante e a filtragem de variáveis espúrias ou redundantes.

3.2.5 Remoção de Atributos Redundantes

Entre os atributos selecionados na etapa anterior, foram removidos aqueles com correlação mútua de Pearson superior a $\tau_{red} = 0{,}95$. Atributos altamente correlacionados entre si carregam informação essencialmente idêntica, aumentando a dimensionalidade do espaço sem agregar poder discriminativo (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). A remoção foi realizada de forma sistemática: para cada par de atributos cuja correlação excedia o limiar, um dos membros do par era descartado.

3.2.6 Dataset Baseline Resultante

Após as cinco etapas de pré-processamento, o conjunto de dados utilizado nos experimentos, denominado *dataset baseline*, contém **19.582 amostras** e **18 atributos**. Os atributos foram derivados a partir de artefatos de memória extraídos por meio do Volatility Framework e seus respectivos plugins, amplamente utilizados em análise forense de memória e detecção de malware (FOUNDATION, 2023b; FOUNDATION, 2023a; LIGH et al., 2014a; SCAIFE et al., 2016).

A Tabela 3 apresenta os 18 atributos retidos com uma breve descrição semântica de cada um.

3.3 Engenharia de Atributos por Consenso SHAP

A engenharia de atributos foi realizada com o objetivo de identificar um subconjunto mínimo e robusto de características capaz de preservar o desempenho de classificação, com menor custo computacional, requisito fundamental para aplicações de detecção em tempo real.

O método adotado utilizou a análise SHAP (LUNDBERG; LEE, 2017), que atribui a cada atributo um valor de importância baseado na teoria dos jogos cooperativos de Shapley. Diferentemente de medidas de importância tradicionais, os valores SHAP são aditivos, consistentes e localmente fiéis à predição de cada amostra individualmente (LUNDBERG; LEE, 2017).

Para cada um dos quatro modelos treinados com o *dataset baseline*, foram calculados os valores SHAP de todas as amostras. A partir desses valores, os atributos foram ranque-

Tabela 3 – Atributos do dataset baseline após pré-processamento

Atributo	**Descrição**
`svcscan.nservices`	Número total de serviços registrados no sistema
`svcscan.process_services`	Serviços ativos vinculados a processos em execução
`dlllist.avg_dlls_per_proc`	Média de DLLs carregadas por processo
`dlllist.ndlls`	Contagem total de DLLs carregadas
`handles.ntimer`	Número de *handles* de *timer* abertos
`svcscan.kernel_drivers`	Número de drivers de *kernel* registrados
`handles.nkey`	Número de *handles* de chave de registro
`pslist.avg_handlers`	Média de *handles* por processo
`pslist.avg_threads`	Média de *threads* por processo
`handles.nmutant`	Número de *handles* de mutex
`ldrmodules.not_in_load`	Módulos ausentes na lista de carregamento
`ldrmodules.not_in_load_avg`	Média de módulos ausentes por processo
`handles.nsemaphore`	Número de *handles* de semáforo
`malfind.commitCharge`	Carga de *commit* em regiões suspeitas
`callbacks.ncallbacks`	Número de *callbacks* de *kernel* registrados
`pslist.nppid`	Número de processos com PID pai inválido
`svcscan.nactive`	Número de serviços ativos no momento da captura
`handles.ndesktop`	Número de *handles* de área de trabalho

Fonte: Elaborado pelo autor (2026).

ados por importância média absoluta (*mean* |SHAP|) em cada modelo. Em seguida, foi aplicado o critério de **consenso por unanimidade**: um atributo foi incluído no conjunto reduzido somente se figurou simultaneamente no *top-10* de importância SHAP de todos os quatro modelos. Esse critério garante que os atributos selecionados sejam relevantes independentemente do paradigma de aprendizado, seja por limiares de divisão (*Decision Tree* e *Random Forest*), por distância geométrica KNN ou por ajuste iterativo adaptativo (*AdaBoost*).

Para *Decision Tree* e *Random Forest*, os valores SHAP foram calculados com o *TreeExplainer*, que acessa diretamente a estrutura interna das árvores e produz valores exatos em tempo computacional eficiente. Para KNN e *AdaBoost*, modelos sem estrutura de árvore acessíveis, foi utilizado o `KernelExplainer`, com 50 centroides como conjunto de referência (*background*) e 150 amostras explicadas por execução. O processo resultou na seleção de **4 atributos** por unanimidade, descritos na Seção 4.2.1.1 do capítulo de Experimentos.

3.4 Configuração dos Modelos

Foram implementados quatro algoritmos de aprendizado de máquina amplamente utilizados em tarefas de classificação e com características complementares entre si. A Tabela 4 apresenta os hiperparâmetros adotados em todos os experimentos.

O KNN foi o único modelo que exigiu normalização prévia dos atributos via *Stan-*

Tabela 4 – Hiperparâmetros dos modelos utilizados nos experimentos

Modelo	**Parâmetro**	**Valor**
KNN	n_neighbors	5
	metric	Euclidiana
	weights	uniform
	normalização	*StandardScaler*
Decision Tree	criterion	gini
	max_depth	sem limite
	min_samples_split	2
	min_samples_leaf	1
	random_state	42
Random Forest	n_estimators	100
	criterion	gini
	max_depth	sem limite
	min_samples_split	2
	random_state	42
AdaBoost	estimador base	*DecisionTree*(*max_depth*=1)
	n_estimators	100
	learning_rate	1,0
	random_state	42

Fonte: Elaborado pelo autor (2026).

dardScaler, por ser sensível à escala das variáveis: a distância euclidiana entre amostras é distorcida quando os atributos possuem magnitudes muito diferentes. Os demais modelos, baseados em estruturas de árvore, tomam decisões por limiares em cada atributo individualmente e não requerem normalização prévia (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

O *AdaBoost* foi configurado com *decision stumps*[1] como estimadores base. Essa escolha é teoricamente justificada: estimadores de baixa complexidade evitam sobreajuste progressivo ao longo das iterações de *boosting* e convergem de forma mais estável (SCHAPIRE; FREUND, 2012).

3.5 Métricas de Avaliação

O desempenho dos classificadores foi avaliado por meio de métricas derivadas da matriz de confusão, que representa a relação entre as classes reais e as preditas pelo modelo. Em um problema de classificação binária, a matriz é composta por quatro elementos: verdadeiros positivos (VP), verdadeiros negativos (VN), falsos positivos (FP) e falsos negativos (FN). Neste trabalho, a classe *Malware* é adotada como positiva e a classe *Benign* como negativa. Os falsos negativos, amostras de *ransomware* classificadas como benignas,

[1] Decision stumps são árvores de decisão com profundidade igual a 1, realizando apenas uma única divisão no espaço de atributos, sendo amplamente utilizados como *weak learners* em algoritmos de *boosting*.

representam o erro de maior criticidade em segurança da informação, pois correspondem a ameaças que passam despercebidas pelo sistema de detecção.

3.5.1 Acurácia

A acurácia mede a proporção total de classificações corretas em relação ao número total de amostras avaliadas:

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

Essa métrica fornece uma visão geral do desempenho, sendo confiável quando as classes estão balanceadas, condição garantida pelo pré-processamento descrito na Seção 3.2.

3.5.2 Precisão

A precisão mede a proporção de predições positivas que estão corretas, indicando quantas das amostras classificadas como *malware* realmente pertencem à classe *malware*:

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

Alta precisão indica baixa ocorrência de falsos positivos, ou seja, poucos alarmes desnecessários para processos benignos.

3.5.3 Revocação (*Recall*)

A revocação, também denominada sensibilidade ou *True Positive Rate*, mede a proporção de amostras de *ransomware* detectadas corretamente:

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

No contexto de segurança da informação, o *recall* é a métrica prioritária: falsos negativos representam infecções de *ransomware* não detectadas, com potencial de causar danos irreversíveis ao sistema.

3.5.4 F1-Score

O *F1-score* corresponde à média harmônica entre precisão e revocação, oferecendo uma medida balanceada quando se deseja considerar simultaneamente os dois tipos de erro:

$$F1 = 2 \cdot \frac{Precisão \cdot Revocação}{Precisão + Revocação} \tag{4}$$

3.5.5 Curva ROC e AUC

A curva ROC representa graficamente a relação entre a taxa de verdadeiros positivos (*recall*) e a False Positive Rate (FPR) para diferentes limiares de decisão:

$$FPR = \frac{FP}{FP + VN} \tag{5}$$

A área sob a curva ROC fornece uma medida agregada da capacidade discriminativa do modelo para qualquer limiar de decisão. Valores próximos de 1,0 indicam separação perfeita entre as classes; valores próximos de 0,5 indicam desempenho equivalente ao acaso (FAWCETT, 2006).

3.6 Protocolo Experimental

O *dataset baseline* utilizado nos experimentos é resultado de um pipeline de pré-processamento em cinco etapas aplicado ao conjunto filtrado do *CIC-MalMem-2022*, que parte originalmente de **57 atributos** por amostra. Na primeira etapa, quatro colunas foram descartadas por não representarem atributos comportamentais extraídos da memória, a saber, `Category` (rótulo categórico), `pslist.nprocs64bit` (altamente colinear com outras contagens de processo), `handles.nport` (informação de rede fora do escopo comportamental) e `svcscan.interactive_process_services` (variância praticamente nula), reduzindo o espaço para **53 atributos**. A etapa de verificação de variância zero não eliminou atributos adicionais no conjunto balanceado resultante. Na etapa seguinte, foi aplicado um filtro de correlação de Pearson com a variável alvo: apenas os atributos com $|\rho| > 0{,}20$ foram retidos, eliminando as variáveis com relação fraca à distinção entre as classes. Por fim, a remoção de redundâncias, pares de atributos com correlação mútua superior a $\tau_{red} = 0{,}95$, descartou as colunas cujas informações eram essencialmente duplicadas. O conjunto resultante dessas cinco etapas sequenciais é o *dataset baseline*, composto por **18 atributos**, descritos na Tabela 3. O número 18 não é, portanto, uma escolha arbitrária, mas o produto determinístico e reprodutível das decisões de filtragem aplicadas ao espaço original de 57 dimensões.

Todos os modelos foram avaliados por meio de validação cruzada estratificada *Stratified K-Fold Cross-Validation* (K-Fold), com $k = 10$, garantindo a preservação da proporção entre as classes em cada partição. As métricas foram calculadas sobre o conjunto de predições acumuladas ao longo dos 10 *folds*, equivalente a uma avaliação sobre o conjunto completo de amostras sem contaminação entre treino e teste. O mesmo protocolo foi aplicado nas duas etapas experimentais, treinamento com o *dataset baseline* 1 (18 atributos) e retreinamento com o *dataset* reduzido (4 atributos), garantindo que as comparações de desempenho entre as duas configurações sejam controladas e livres de viés de avali-

ação. A semente aleatória foi fixada em `random_state = 42` em todos os procedimentos estocásticos para assegurar reprodutibilidade integral dos resultados.

Capítulo 4

Experimentos e Análise dos Resultados

Este capítulo apresenta os procedimentos experimentais adotados, as métricas utilizadas para avaliar o desempenho dos modelos e a análise dos resultados obtidos. O objetivo é demonstrar, com base em evidências empíricas, se a hipótese proposta neste trabalho é válida: a de que um subconjunto mínimo de atributos comportamentais extraídos da memória é suficiente para detectar *ransomware* com desempenho equivalente ao de um conjunto completo de atributos. Para isso, são descritos o método de avaliação, as configurações dos experimentos, os conjuntos de dados utilizados e a discussão dos resultados produzidos pelos modelos de detecção.

4.1 Método para a Avaliação

Esta etapa foi realizada por meio da construção e comparação de quatro modelos de aprendizado de máquina KNN, *Decision Tree, Random Forest e AdaBoost*, aplicados ao problema de detecção de *ransomware* a partir de artefatos extraídos da memória. Foram utilizadas métricas amplamente aceitas na literatura, incluindo acurácia, precisão, *recall* e *F1-score*, além de matrizes de confusão, curvas ROC e curvas de aprendizagem, que permitem medir de forma equilibrada o desempenho dos classificadores e identificar padrões de erro relevantes do ponto de vista de segurança.

Dois conjuntos de dados foram empregados ao longo dos experimentos: uma versão completa, denominada *dataset baseline*, contendo os 18 atributos resultantes do pré-processamento do conjunto *CIC-Malmem-2022*; e uma versão reduzida, contendo apenas os 4 atributos selecionados por consenso entre os quatro algoritmos, utilizando o método SHAP (LUNDBERG; LEE, 2017). A seleção foi baseada na unanimidade de votos: um atributo só foi incluído no conjunto reduzido se figurou simultaneamente no *top-10* de importância SHAP de todos os quatro modelos, garantindo relevância agnóstica ao paradigma de aprendizado.

O protocolo de avaliação foi idêntico nas duas etapas: validação cruzada estratificada com 10 *K-folds*, mantendo a proporção entre as classes em cada partição. As métricas

foram calculadas sobre o conjunto de predições acumuladas ao longo dos 10 *folds*, equivalente a uma avaliação sobre o conjunto completo de amostras sem contaminação entre treino e teste. A classe positiva adotada é *Malware*, de modo que o *recall* corresponde à sensibilidade do modelo à detecção de *ransomware* , a métrica de maior criticidade em segurança da informação, pois falsos negativos representam ameaças que passam despercebidas pelo sistema de detecção.

4.2 Experimentos

Os experimentos foram organizados em duas etapas sequenciais e complementares. A **Etapa 1** estabelece o desempenho de referência dos quatro classificadores com o conjunto completo de 18 atributos e produz as análises SHAP que fundamentam a seleção por consenso. A **Etapa 2** retreina os mesmos modelos com o subconjunto de 4 atributos consensuais, repetindo integralmente os procedimentos de avaliação e análise da etapa anterior para permitir comparação direta e controlada. Ao final, os resultados das duas etapas são consolidados em tabelas de variação de desempenho e gráficos comparativos.

4.2.1 Etapa 1 - Experimentos com o Dataset Baseline

A primeira etapa teve por objetivo estabelecer o patamar de referência de desempenho dos quatro classificadores no conjunto completo de 18 atributos resultante do pré-processamento. Além das métricas quantitativas, foram gerados: matrizes de confusão, curvas ROC, curvas de aprendizagem, curvas de convergência (*AdaBoost staged score* e erro Out-Of-Bag Error (OOB) do *Random Forest*) e análises SHAP individuais por modelo. Ao término desta etapa, as importâncias SHAP foram utilizadas para identificar os atributos mais relevantes e fundamentar a seleção por consenso descrita na Seção 4.2.1.1.

4.2.1.1 Atributos do Dataset Baseline e Seleção por Consenso SHAP

A Tabela 5 apresenta os 18 atributos do *dataset baseline* ordenados por importância SHAP média entre os quatro modelos, com indicação dos atributos incluídos no critério cumulativo de 75% e dos selecionados por unanimidade (4/4 votos), que formam o *dataset* reduzido.

Os quatro atributos selecionados por unanimidade são: `svcscan.nservices` (número total de serviços registrados, maior importância SHAP, responsável por 28,2% da importância cumulativa), `svcscan.process_services` (serviços ativos vinculados a processos em execução), `dlllist.ndlls` (contagem total de DLLs carregadas) e `handles.ntimer` (*handles* de *timer* abertos). Cada um desses atributos captura um aspecto comportamental distinto do *ransomware* em memória: persistência via registro de serviços, execução

Tabela 5 – Atributos do dataset baseline ordenados por importância SHAP média

Atributo	SHAP Médio	Cumul.	Votos	75%	Consenso
`svcscan.nservices`	0,1255	0,282	4	Sim	**Sim**
`svcscan.process_services`	0,0589	0,415	4	Sim	**Sim**
`dlllist.avg_dlls_per_proc`	0,0404	0,506	3	Sim	Não
`dlllist.ndlls`	0,0312	0,576	4	Sim	**Sim**
`handles.ntimer`	0,0251	0,632	4	Sim	**Sim**
`svcscan.kernel_drivers`	0,0215	0,681	1	Sim	Não
`handles.nkey`	0,0210	0,728	2	Sim	Não
`pslist.avg_handlers`	0,0209	0,775	3	Sim	Não
`pslist.avg_threads`	0,0176	0,815	3	Não	Não
`handles.nmutant`	0,0172	0,853	2	Não	Não
`ldrmodules.not_in_load`	0,0155	0,888	3	Não	Não
`ldrmodules.not_in_load_avg`	0,0103	0,911	1	Não	Não
`handles.nsemaphore`	0,0087	0,931	2	Não	Não
`malfind.commitCharge`	0,0086	0,950	1	Não	Não
`callbacks.ncallbacks`	0,0084	0,969	1	Não	Não
`pslist.nppid`	0,0065	0,983	1	Não	Não
`svcscan.nactive`	0,0053	0,995	0	Não	Não
`handles.ndesktop`	0,0020	1,000	1	Não	Não

Votos = nº de modelos que incluíram o atributo no top-10 SHAP;
Cumul. = importância cumulativa normalizada.

Fonte: Elaborado pelo autor (2026).

ativa de serviços de processo, carga de bibliotecas criptográficas e sincronização temporal de operações maliciosas.

4.2.1.2 Métricas - Dataset Baseline

A Tabela 6 apresenta as métricas completas dos quatro modelos avaliados com o *dataset baseline*.

Tabela 6 – Métricas dos modelos com o dataset baseline - 18 atributos (validação cruzada 10-fold)

Modelo	Acurácia (%)	Precisão (%)	Recall (%)	F1-Score (%)
AdaBoost	**99,9898**	**99,9796**	**100,0000**	**99,9898**
Random Forest	99,9796	99,9592	100,0000	99,9796
KNN	99,9745	99,9490	100,0000	99,9745
Decision Tree	99,9643	99,9694	99,9591	99,9643

Classe positiva = Malware. Recall = sensibilidade à classe Malware.

Fonte: Elaborado pelo autor (2026).

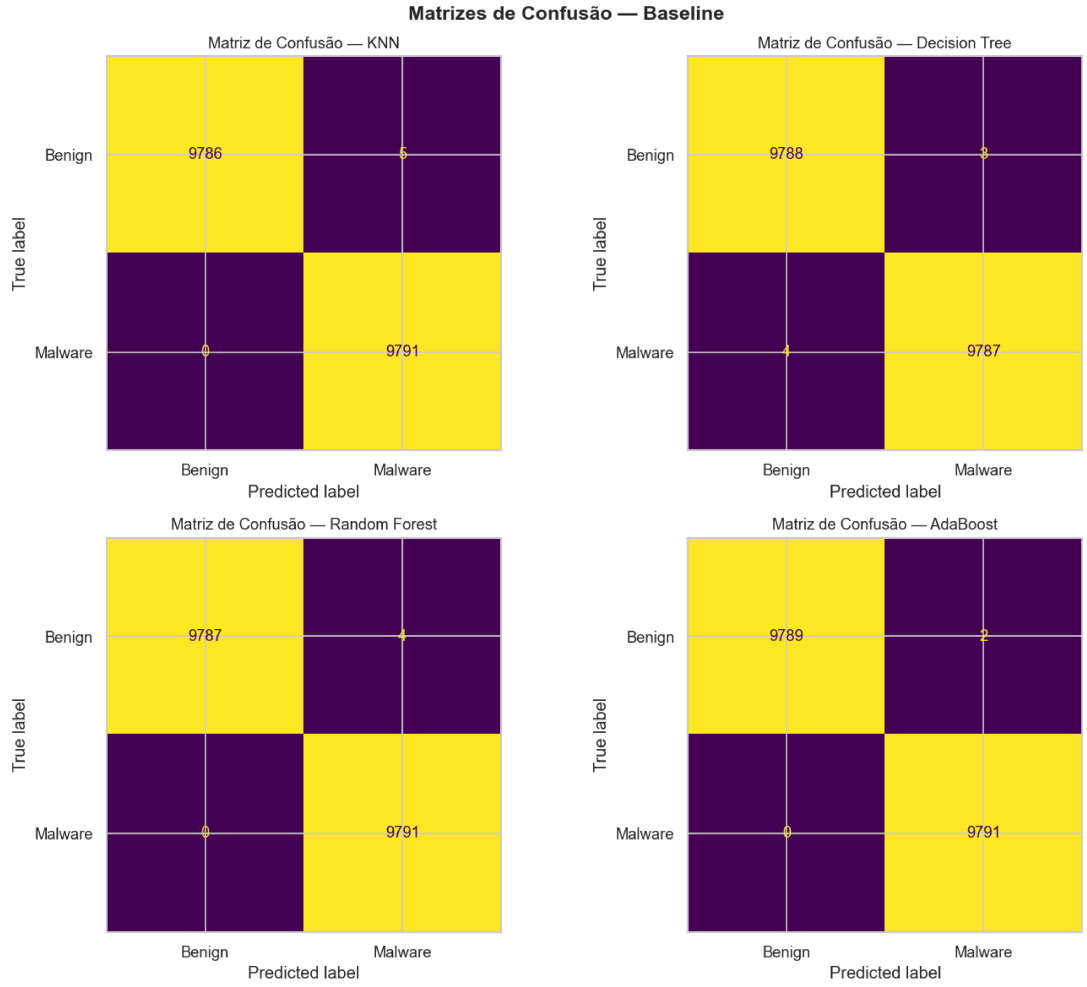
Todos os quatro classificadores atingiram acurácia superior a 99,96%, com *AdaBoost* registrando o melhor resultado isolado: *F1-score* de 99,9898%, zero falsos negativos e apenas 2 falsos positivos em 19.582 amostras avaliadas. *Random Forest* e KNN também alcançaram *recall* perfeito (100%), não deixando nenhuma amostra de *ransomware* passar sem detecção. A *Decision Tree* foi o único modelo a registrar falsos negativos (4 amostras),

o que, embora numericamente desprezível, representa a classe de erro de maior risco em aplicações de segurança da informação.

4.2.1.3 Matrizes de Confusão - Baseline

A Figura 2 apresenta as matrizes de confusão dos quatro modelos no *dataset baseline*. A análise dos valores absolutos revela diferenças importantes no comportamento de cada classificador, especialmente nos Falsos Negativos (FN) *ransomware* classificado como benigno que representam o erro de maior criticidade em segurança da informação.

Figura 2 – Matrizes de confusão dos quatro modelos no dataset baseline(validação cruzada 10-fold).



Fonte: Elaborado pelo autor (2026).

O **AdaBoost** registrou **zero falsos negativos** (9.791/9.791 amostras de *ransomware* detectadas corretamente) e apenas 2 falsos positivos, o melhor resultado absoluto entre todos os modelos. O **Random Forest** e o **KNN** também atingiram zero falsos negativos, com 4 e 5 falsos positivos, respectivamente. A **Decision Tree** foi o único modelo a registrar falsos negativos (4 amostras não detectadas) e 3 falsos positivos. Em todos os casos, os erros totais são desprezíveis em relação ao volume de 19.582 amostras, mas a

distinção entre FN = 0 e FN = 4 é relevante do ponto de vista de segurança. A Tabela 7 sistematiza os valores absolutos.

Tabela 7 – Resumo dos erros absolutos nas matrizes de confusão - dataset baseline

Modelo	**VP**	**VN**	**FP**	**FN**
AdaBoost	9.791	9.789	2	0
Random Forest	9.791	9.787	4	0
KNN	9.791	9.786	5	0
Decision Tree	9.787	9.788	3	4

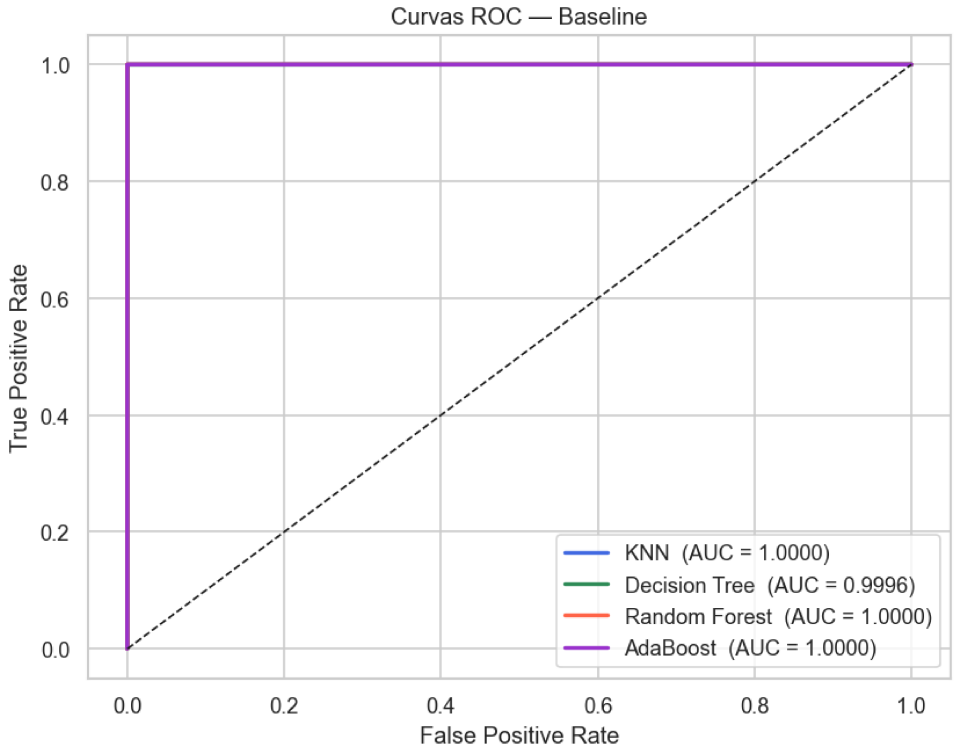
Verdadeiros Positivos (VP) = Malware classificado como malware;
Verdadeiros Negativos (VN) = Benigno classificado como benigno;
Falsos Positivos (FP) = Benigno classificado como malware;
FN = Malware classificado como benigno.

Fonte: Elaborado pelo autor (2026).

4.2.1.4 Curvas ROC - Baseline

A Figura 3 apresenta as curvas ROC dos quatro modelos no *dataset baseline*.

Figura 3 – Curvas ROC dos quatro modelos no dataset baseline. KNN, Random Forest e AdaBoost atingem Area Under the Curve (AUC)=1,0000, enquanto a Decision Tree obtém AUC=0,9996.



Fonte: Elaborado pelo autor (2026).

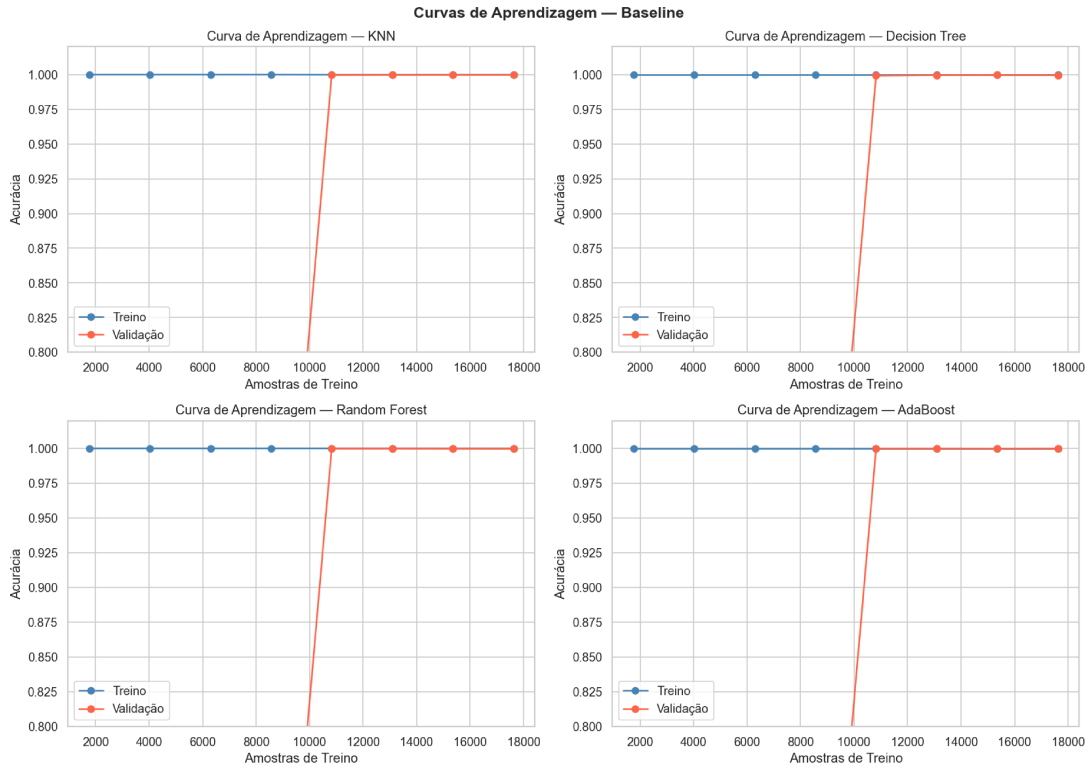
KNN, *Random Forest* e *AdaBoost* atingiram AUC = 1,0000 - curvas que alcançam o canto superior esquerdo do gráfico praticamente de imediato, indicando discriminação perfeita entre as classes para qualquer limiar de decisão. A *Decision Tree* obteve

AUC = 0,9996, ligeiramente inferior, o que reflete sua natureza determinística: por ser um modelo de limiar fixo (sem saída probabilística contínua), sua curva ROC apresenta menor granularidade nos pontos intermediários. As quatro curvas sobrepõem-se visivelmente no gráfico, confirmando a equivalência prática de todos os modelos neste critério.

4.2.1.5 Curvas de Aprendizagem - Baseline

A Figura 4 apresenta as curvas de aprendizagem dos quatro modelos com o *dataset baseline*.

Figura 4 – Curvas de aprendizagem dos quatro modelos no *dataset baseline*. As curvas de treino estabilizam em 1,0 desde o início, enquanto as curvas de validação convergem para o mesmo nível a partir de aproximadamente 11.000 amostras.



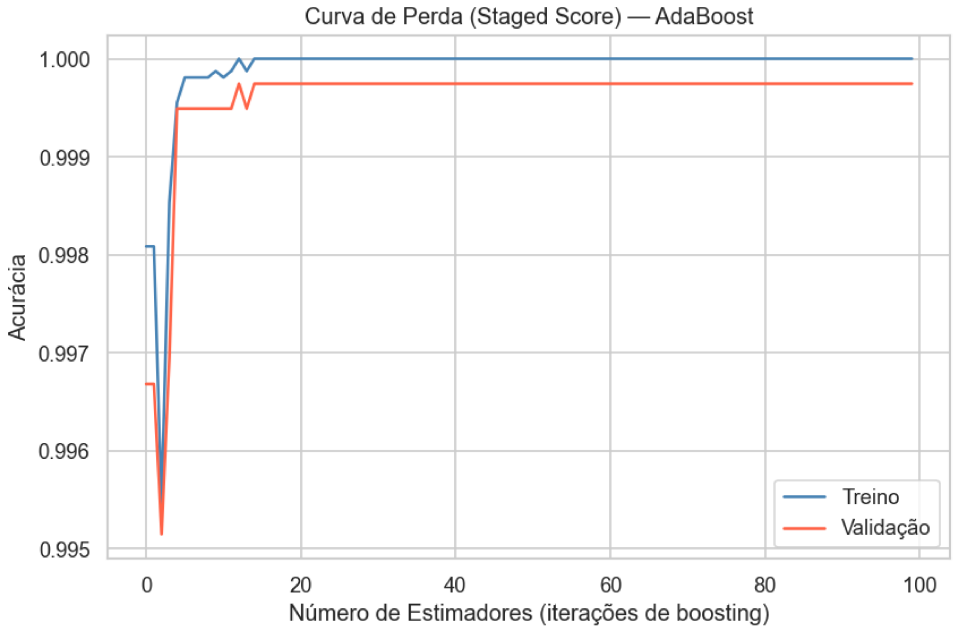
Fonte: Elaborado pelo autor (2026).

Todos os quatro modelos exibem o mesmo padrão: a curva de treino permanece em acurácia 1,0 (ou muito próxima) desde os primeiros pontos avaliados, enquanto a curva de validação parte de valores entre 0,80 e 0,82 com cerca de 2.000 amostras de treino e converge abruptamente para valores próximos a 1,0 a partir de aproximadamente 11.000 amostras. Esse comportamento indica que os padrões discriminativos entre ransomware e processos benignos no *CIC-Malmem-2022* são altamente concentrados: existe um limiar crítico de volume abaixo do qual a cobertura das variações comportamentais é insuficiente. Acima desse limiar, todos os classificadores generalizam de forma consistente, sem evidências de sobreajuste as curvas de treino e validação convergem e se estabilizam.

4.2.1.6 Curva de Convergência do AdaBoost - Staged Score

A Figura 5 apresenta a curva de convergência do *AdaBoost* ao longo das 100 iterações de *boosting*.

Figura 5 – Curva de convergência do *AdaBoost* (*staged score*). O modelo converge rapidamente nas primeiras 15 iterações e estabiliza a acurácia de validação próxima a 0,9990 ao longo das 100 rodadas.



Fonte: Elaborado pelo autor (2026).

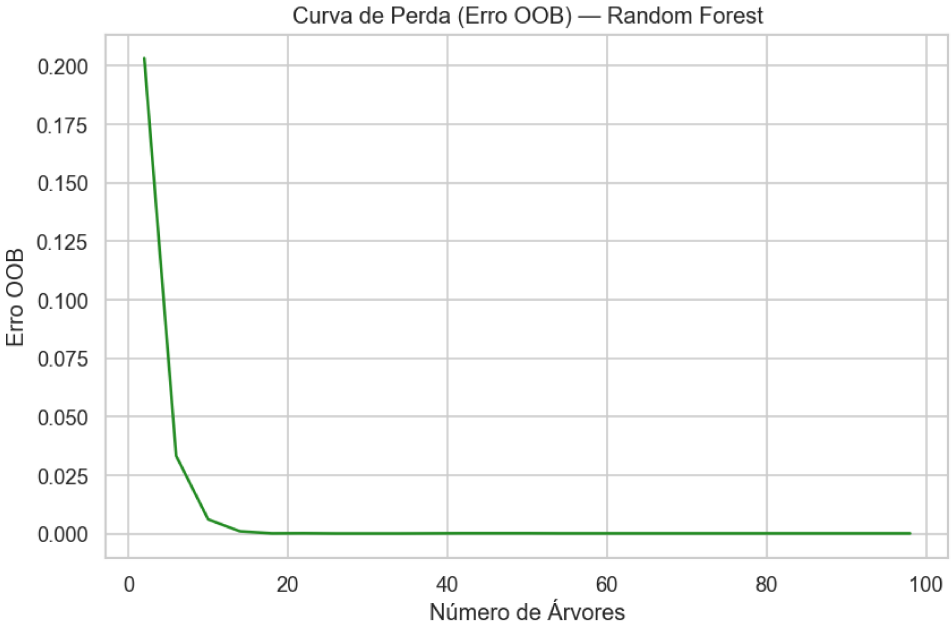
A curva revela três fases distintas. Na fase inicial (iterações 1 a 5), ambas as curvas exibem oscilação enquanto os primeiros estimadores ajustam seus pesos. Na fase de convergência rápida (iterações 5 a 15), ambas sobem de forma acentuada, com a curva de treino atingindo valores próximos a 1,0000 e a de validação estabilizando em torno de 0,9987. Na fase estável (iterações 15 a 100), ambas permanecem praticamente constantes, sem qualquer tendência de queda. A ausência de degradação na curva de validação ao longo de 100 iterações confirma que o modelo não incorre em sobreajuste progressivo, comportamento teoricamente esperado quando os classificadores base são *stumps* de baixa complexidade (SCHAPIRE; FREUND, 2012). A diferença residual entre as curvas é inferior a 0,001 ponto percentual, indicando excelente capacidade de generalização. Praticamente todo o potencial discriminativo do *AdaBoost* é alcançado nas primeiras 15 a 20 iterações; as 80 iterações restantes estabilizam o modelo sem introduzir instabilidade.

4.2.1.7 Curva de Erro OOB - Random Forest

A Figura 6 apresenta o erro *OOB* do *Random Forest* em função do número de árvores.

Com apenas 2 árvores, o erro OOB parte de 0,205. A queda é extremamente abrupta: com 10 árvores o erro já é inferior a 0,025, e com 20 árvores está essencialmente em zero,

Figura 6 – Erro OOB do *Random Forest* em função do número de árvores. O erro reduz-se rapidamente nas primeiras árvores e estabiliza-se próximo de zero ao longo do restante da curva.



Fonte: Elaborado pelo autor (2026).

onde permanece estável até as 100 árvores configuradas. Essa convergência ultrarrápida confirma que os padrões discriminativos do *dataset* são suficientemente regulares para que poucas árvores independentes já capturem a fronteira de decisão essencial. A configuração de 100 árvores é amplamente conservadora para este problema, 25 a 30 árvores seriam suficientes para atingir o mesmo desempenho, o que sugere que em cenários de produção com restrições computacionais seria possível reduzir drasticamente o número de estimadores sem perda perceptível de desempenho.

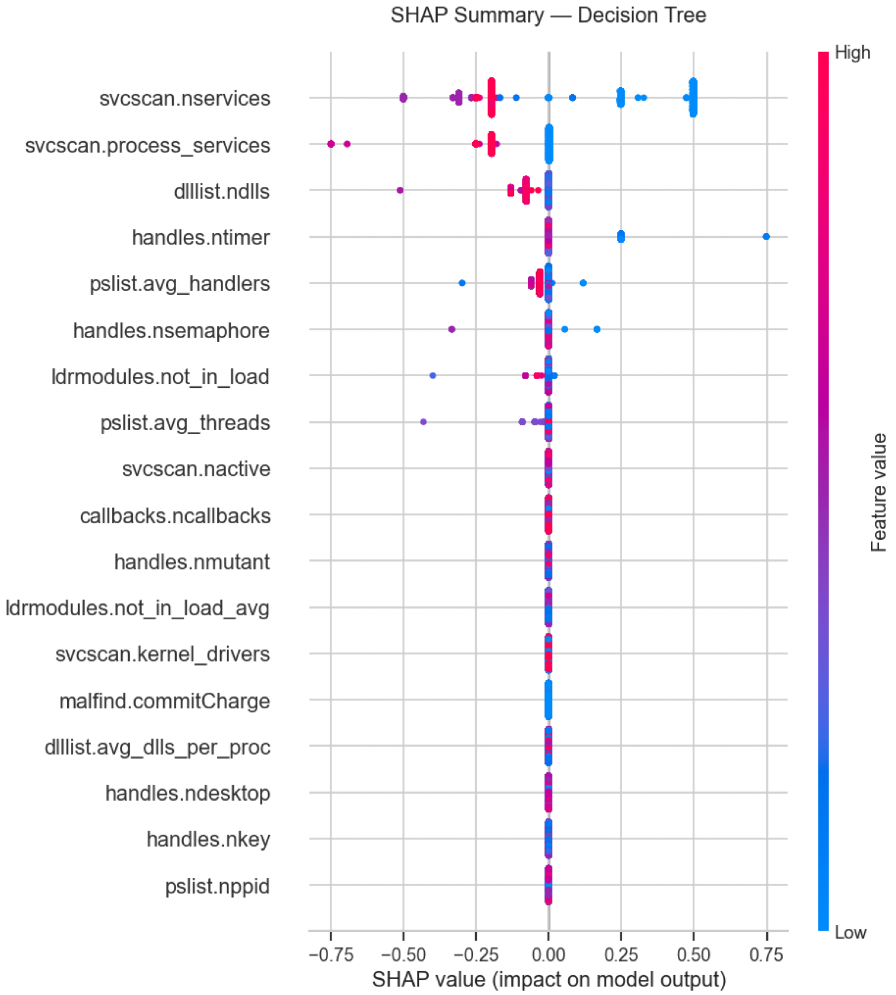
4.2.1.8 Análise SHAP Individual por Modelo - Baseline

As figuras a seguir apresentam os gráficos SHAP *beeswarm* e *bar plot* para cada um dos quatro modelos treinados com o *dataset baseline*. No *beeswarm*, cada ponto representa uma amostra: o eixo horizontal indica o valor SHAP (positivo empurra a predição para *Malware*; negativo para *Benign*), e a cor representa o valor do atributo naquela amostra (vermelho = alto; azul = baixo).

Decision Tree

A *Decision Tree* apresenta o padrão mais concentrado de todos os modelos: `svcscan.n services` domina de forma quase exclusiva, com SHAP médio absoluto de aproximadamente 0,325, valor 4,3 vezes superior ao segundo atributo mais importante. Isso indica que a árvore constrói sua decisão primária em um único nó de divisão baseado no nú-

Figura 7 – SHAP beeswarm - Decision Tree: `svcscan.nservices` domina de forma extrema, com valores SHAP superiores a ±0,5. Os quatro atributos consensuais ocupam as quatro primeiras posições.



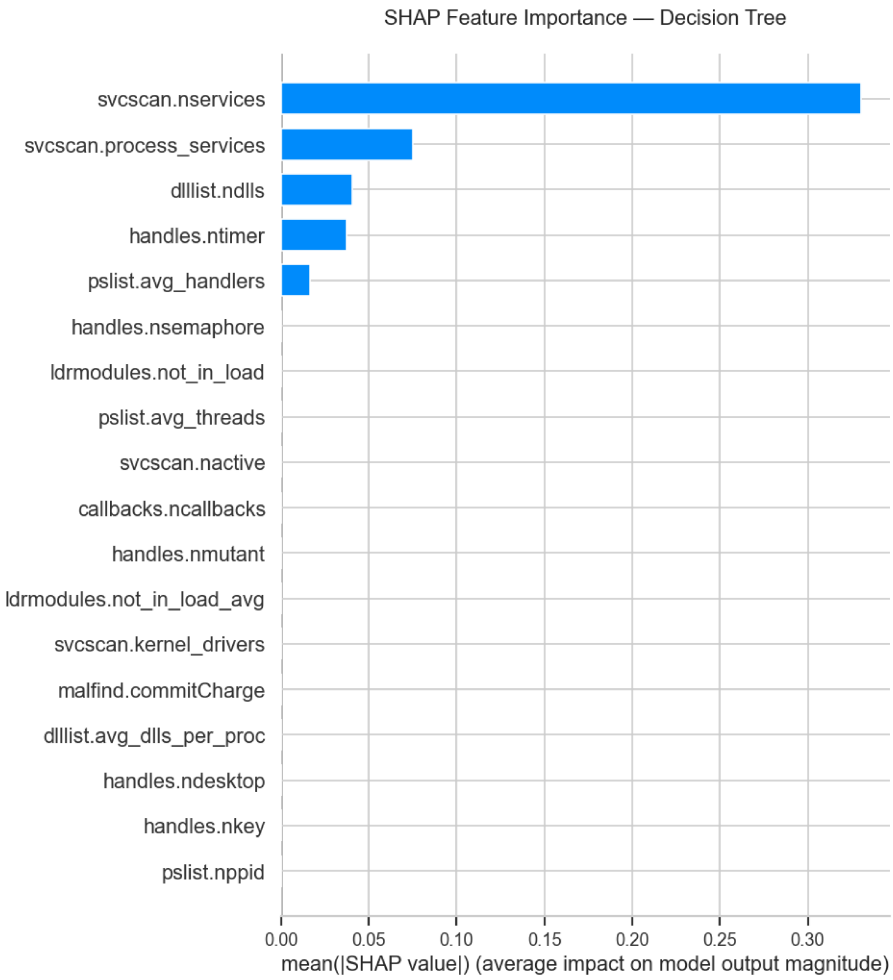
Fonte: Elaborado pelo autor (2026).

mero de serviços, relegando os demais atributos a nós secundários de refinamento. Os 14 atributos restantes apresentam importância próxima de zero no gráfico de barras, o que explica a estabilidade perfeita desse classificador após a redução dimensional descrita na Etapa 2.

Random Forest

O *Random Forest* apresenta um perfil de importância mais distribuído, reflexo da diversidade entre as 100 árvores que compõem o *ensemble*. `svcscan.nservices` permanece como atributo mais importante (SHAP $\approx 0{,}107$), mas a diferença para os seguintes é menor do que na Decision Tree: `dlllist.avg_dlls_per_proc` (0,082) e `svcscan.kernel_drivers` (0,079) também apresentam contribuições relevantes. Notavelmente, `svcscan.kernel_drivers` aparece em segundo ou terceiro lugar no *Random Forest*, mas recebeu apenas 1 voto no

Figura 8 – SHAP bar plot - Decision Tree: `svcscan.nservices` concentra SHAP médio de 0,325, enquanto o segundo atributo (`svcscan.process_services`) apresenta valor de 0,075, correspondendo a uma diferença de 4,3 vezes.



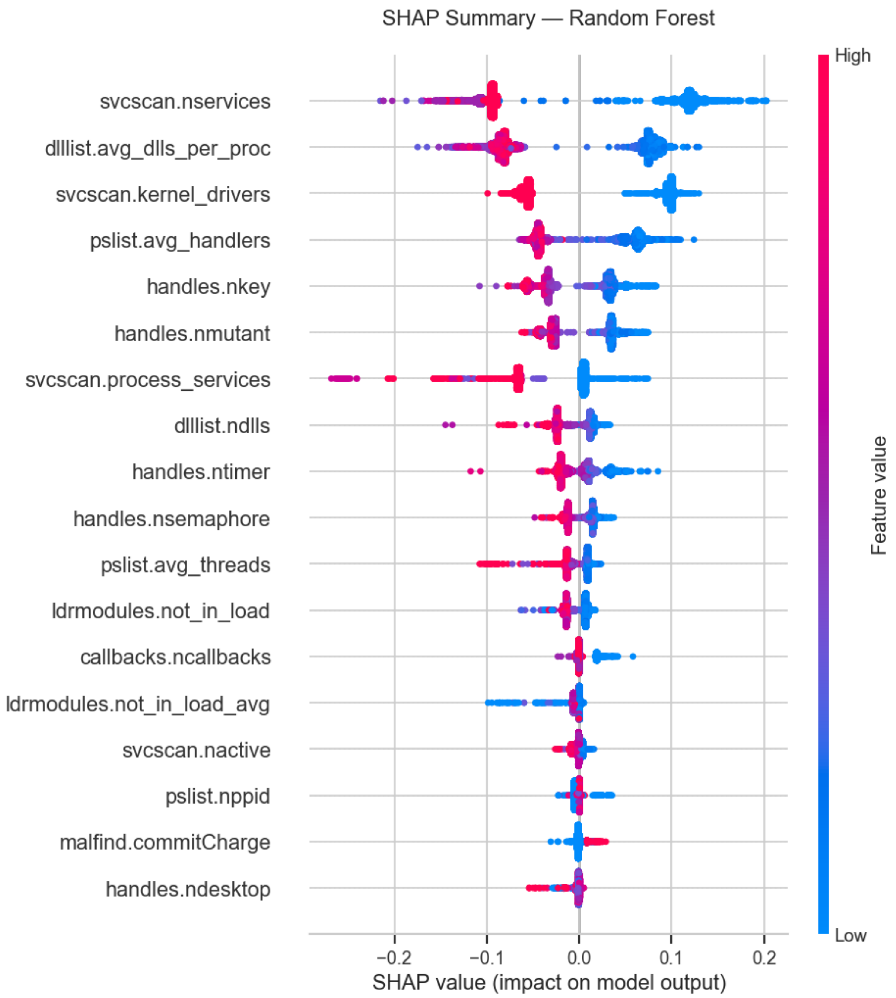
Fonte: Elaborado pelo autor (2026).

consenso - evidência de que esse atributo é importante especificamente para o mecanismo de *bagging*, mas não para os demais paradigmas de aprendizado.

KNN

O KNN apresenta o perfil de importância mais heterogêneo entre os quatro modelos, o que é esperado dado que seu mecanismo de classificação , baseado em distância geométrica no espaço normalizado dos atributos , difere fundamentalmente dos algoritmos baseados em árvore. `dlllist.avg_dlls_per_proc` lidera com SHAP médio de 0,058, seguido de `svcscan.process_services` (0,054) e `pslist.avg_threads` (0,044). Atributos como `ldrmodules.not_in_load`, `handles.nkey` e `handles.ntimer` aparecem em posições de destaque no KNN mas são pouco relevantes para *Decision Tree* e *Random Forest*, revelando que o KNN captura padrões de similaridade local em um espaço de alta dimen-

Figura 9 – SHAP beeswarm - Random Forest: padrão mais distribuído que a Decision Tree, com `svcscan.nservices` ainda dominante, mas com participação relevante de `dlllist.avg_dlls_per_proc` e `svcscan.kernel_drivers`.



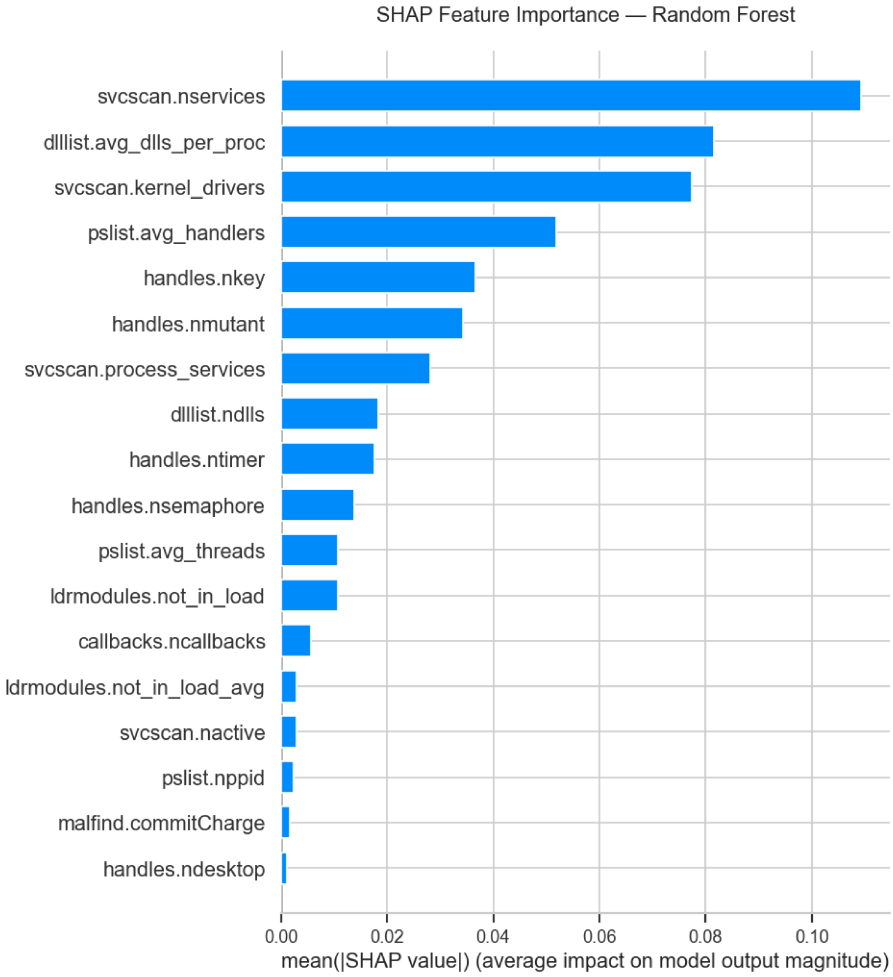
Fonte: Elaborado pelo autor (2026).

são, sensível a atributos que definem geometria de vizinhança mas não necessariamente limites de decisão globais.

AdaBoost

O *AdaBoost* apresenta um perfil de importância com características próprias do mecanismo de *boosting*: `svcscan.process_services` lidera com SHAP médio de 0,077. Essa inversão em relação aos demais modelos é explicável pelo mecanismo iterativo: nos primeiros estimadores, o modelo aprende a separação mais óbvia baseada em `svcscan.nservices`; nas iterações subsequentes, o algoritmo foca nas amostras de maior dificuldade, mais bem discriminadas por `svcscan.process_services` e atributos complementares. No *beeswarm*, valores altos (vermelhos) de `svcscan.process_services` apresentam SHAP negativo , processos com muitos serviços ativos são empurrados para *Benign* pelo AdaBoost, enquanto o padrão oposto é associado a *Malware*. Essa inversão reflete o fato de

Figura 10 – SHAP bar plot - Random Forest: `svcscan.nservices` lidera (0,107), seguido por `dlllist.avg_dlls_per_proc` (0,082) e `svcscan.kernel_drivers` (0,079).



Fonte: Elaborado pelo autor (2026).

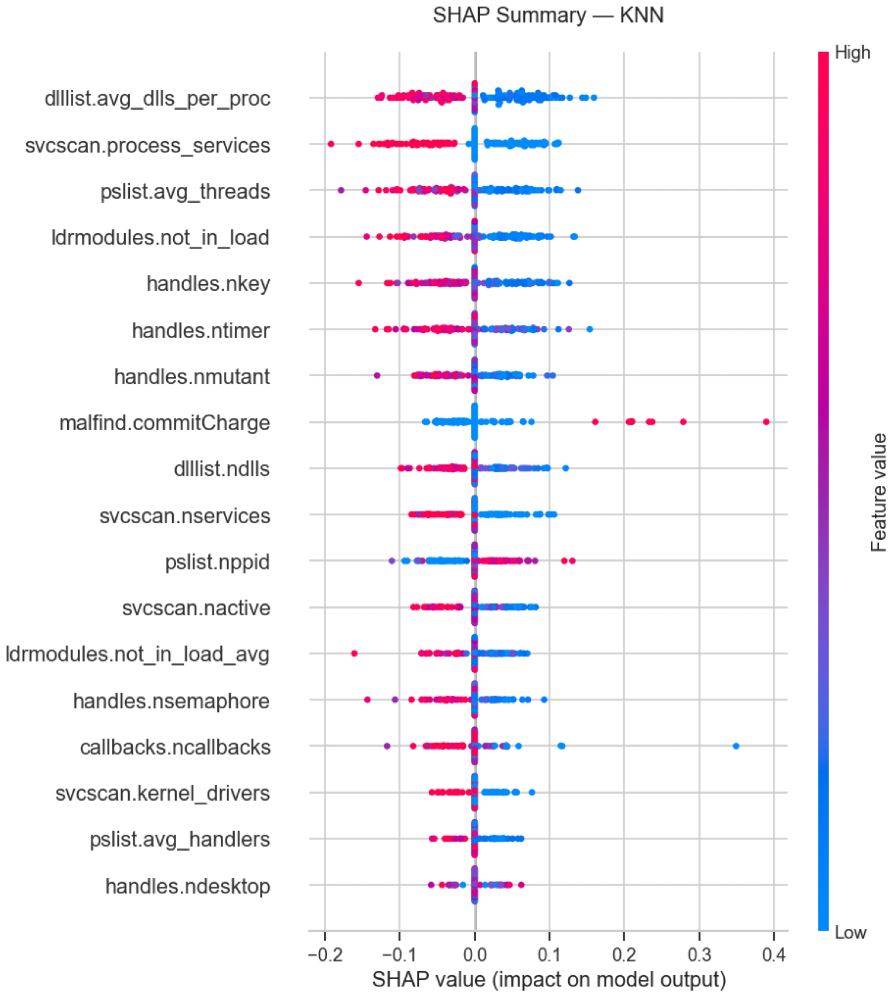
que o *AdaBoost* calibra seus pesos nas amostras mais ambíguas, capturando nuances que os demais modelos tratam de forma mais direta.

4.2.1.9 Correlação das Importâncias SHAP entre Modelos - Baseline

A Figura 15 apresenta a correlação entre os vetores de importância SHAP dos quatro modelos no *dataset baseline*.

A maior correlação observada é entre *Decision Tree* e *Random Forest* (0,619), esperada dado que ambos compartilham o mecanismo de divisão por limiares em atributos individuais. O *AdaBoost* apresenta correlação moderada com a *Decision Tree* (0,493) e com o *KNN* (0,455), funcionando como ponte entre os dois paradigmas. A correlação mais baixa é entre *KNN* e *Decision Tree* (0,041) , valor praticamente nulo, confirmando que esses dois algoritmos exploram aspectos completamente distintos dos dados. Essa heterogeneidade de perspectivas é o que confere robustez ao método de seleção por consenso: um atributo

Figura 11 – SHAP beeswarm - KNN: distribuição mais uniforme entre os 18 atributos, com `dlllist.avg_dlls_per_proc` no topo e importância expressiva para atributos de baixa relevância nos demais modelos.



Fonte: Elaborado pelo autor (2026).

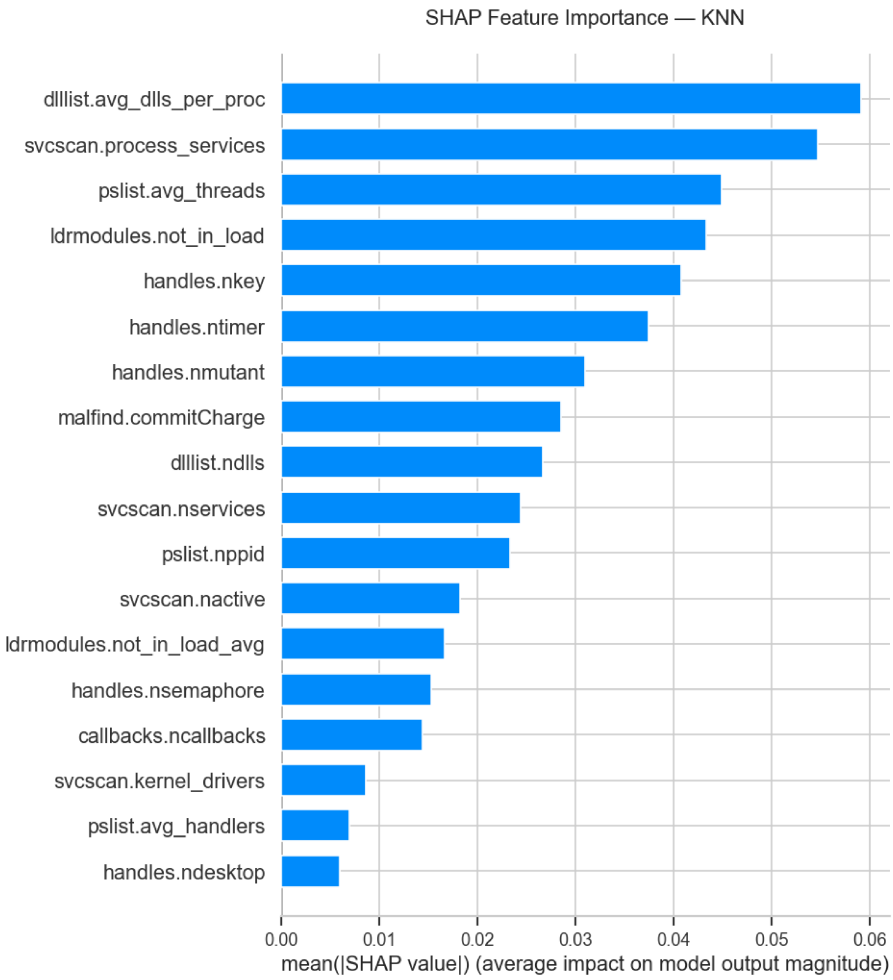
selecionado unanimemente por quatro modelos com correlações de importância tão baixas quanto 0,041 entre si é, por definição, relevante para os mais diferentes mecanismos de aprendizado.

4.2.1.10 Feature Engineering - Consenso SHAP

As Figuras 16 e 17 apresentam, respectivamente, os gráficos de seleção por consenso SHAP e o heatmap de importância normalizada por modelo.

O *heatmap* confirma visualmente a heterogeneidade das perspectivas de cada modelo. Decision Tree e Random Forest atribuem importância máxima normalizada (1,00) a `svcscan.nservices`. O *KNN* elege `dlllist.avg_dlls_per_proc` como atributo mais importante (1,00) e atribui a `svcscan.nservices` apenas 0,24 , contraste de 4,2 vezes em relação aos modelos de árvore. O *AdaBoost* atribui importância máxima (1,00) a `svcscan.process_services` e 0,49 a `svcscan.nservices`. Apesar dessas diferenças in-

Figura 12 – SHAP bar plot - KNN: ranking distinto dos modelos baseados em árvore. `dlllist.avg_dlls_per_proc` lidera (0,058), seguido por `svcscan.process_services` (0,054) e `pslist.avg_threads` (0,044).



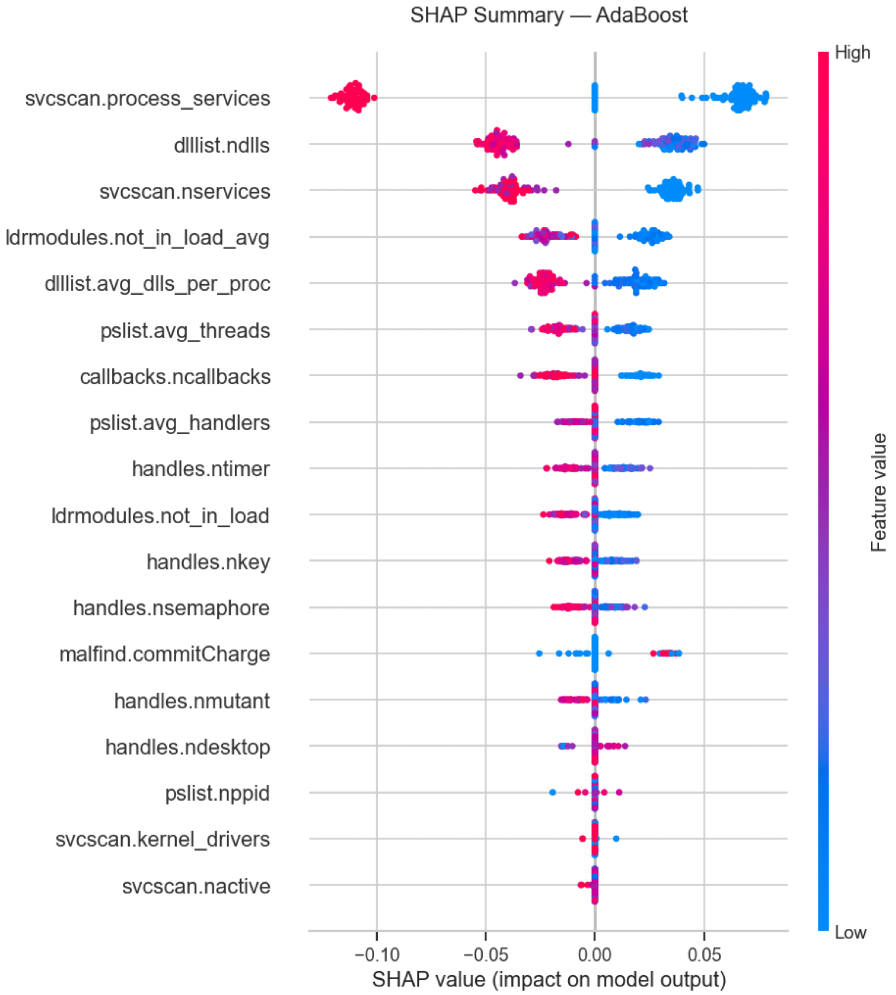
Fonte: Elaborado pelo autor (2026).

ternas, os quatro atributos consensuais aparecem com importância não nula em todos os quadrantes do *heatmap* , condição necessária e suficiente para que o critério de unanimidade os selecione.

4.2.2 Etapa 2 - Experimentos com o Dataset Reduzido por Consenso SHAP

A segunda etapa consistiu no retreinamento dos quatro modelos utilizando exclusivamente os 4 atributos selecionados por unanimidade na Etapa 1: `svcscan.nservices`, `svcscan.process_services`, `dlllist.ndlls` e `handles.ntimer`. Os mesmos protocolos de avaliação e análises SHAP foram integralmente repetidos, permitindo comparação direta e controlada com o *baseline*. O objetivo foi verificar empiricamente se a eliminação de 78% dos atributos , de 18 para 4 , implica degradação mensurável no desempenho dos classificadores.

Figura 13 – SHAP beeswarm - AdaBoost: `svcscan.process_services` lidera, com pontos vermelhos (valores altos) associados a SHAP negativo - padrão invertido em relação à Decision Tree, reflexo da orientação do *boosting* adaptativo.



Fonte: Elaborado pelo autor (2026).

4.2.2.1 Métricas - Dataset Reduzido

A Tabela 8 apresenta as métricas obtidas com o *dataset* reduzido.

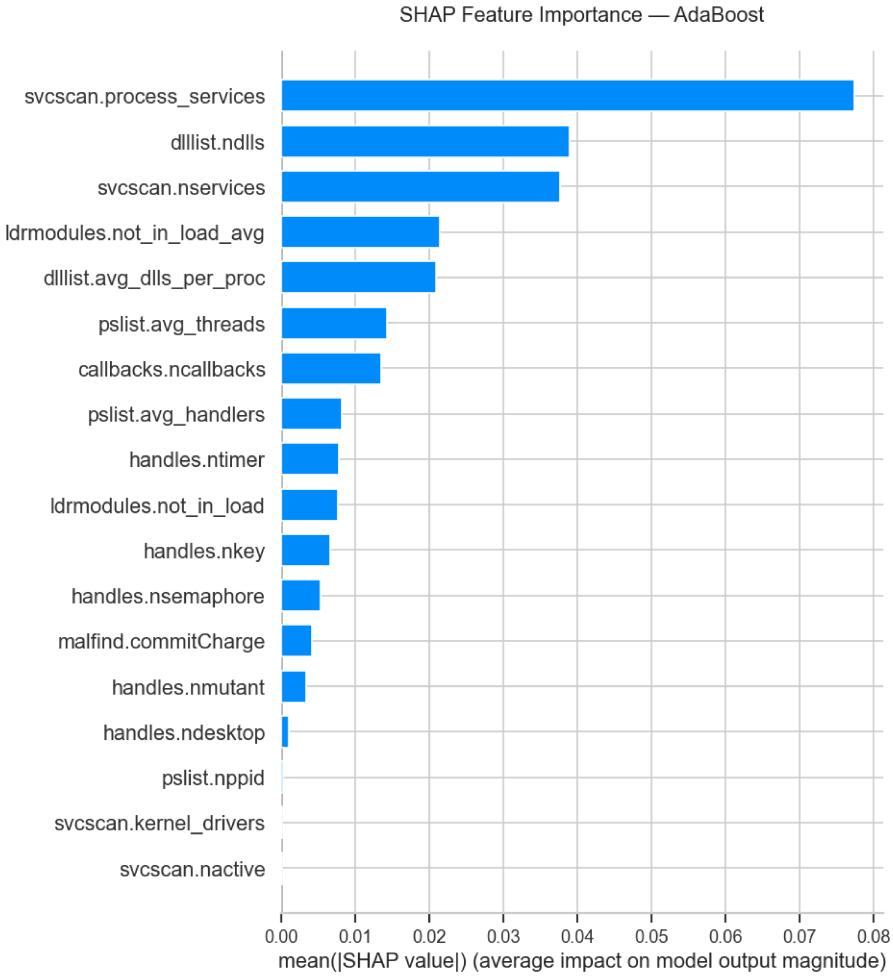
Tabela 8 – Métricas dos modelos com o dataset reduzido - 4 atributos (validação cruzada 10-fold)

Modelo	Acurácia (%)	Precisão (%)	Recall (%)	F1-Score (%)
Random Forest	**99,9847**	99,9694	**100,0000**	**99,9847**
AdaBoost	**99,9847**	**99,9796**	99,9898	**99,9847**
KNN	99,9694	99,9388	100,0000	99,9694
Decision Tree	99,9643	99,9694	99,9591	99,9643

Classe positiva = Malware. Recall = sensibilidade à classe Malware.

Fonte: Elaborado pelo autor (2026).

Figura 14 – SHAP bar plot - AdaBoost: `svcscan.process_services` é o atributo de maior impacto (0,077), diferentemente dos demais modelos onde `svcscan.nservices` lidera.



Fonte: Elaborado pelo autor (2026).

4.2.2.2 Matrizes de Confusão - Dataset Reduzido

A Figura 18 apresenta as matrizes de confusão dos quatro modelos retreinados. A Tabela 9 sistematiza os erros absolutos e permite a comparação direta com o *baseline* (Tabela 7).

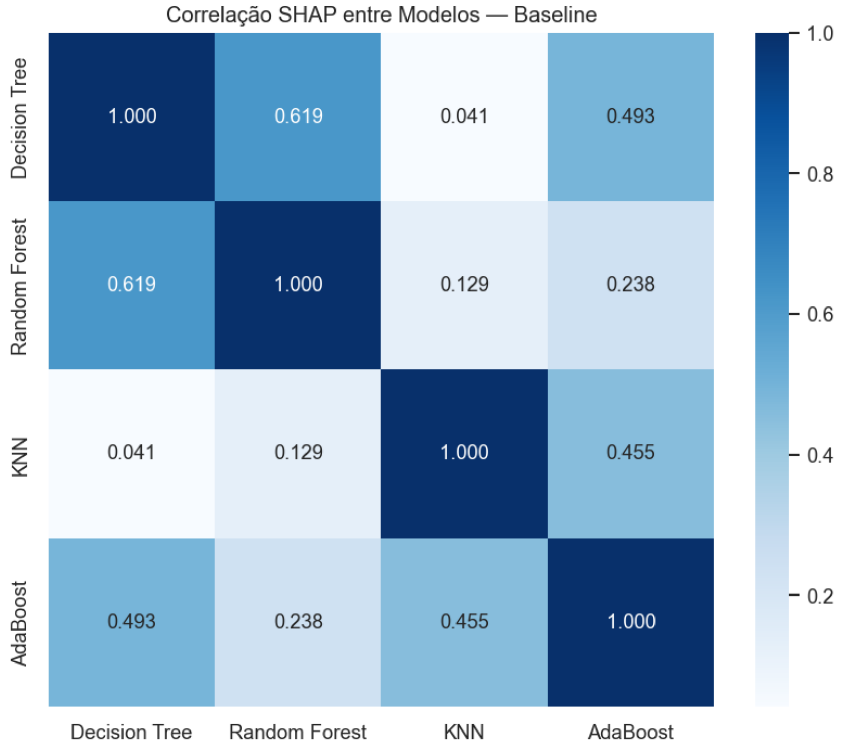
Tabela 9 – Erros absolutos nas matrizes de confusão - dataset reduzido (4 atributos SHAP)

Modelo	VP	VN	FP	FN	Erros totais
Random Forest	9.791	9.788	3	0	3
AdaBoost	9.790	9.789	2	1	3
KNN	9.791	9.785	6	0	6
Decision Tree	9.787	9.788	3	4	7

VP = verdadeiros positivos; VN = verdadeiros negativos; FP = falsos positivos; FN = falsos negativos.

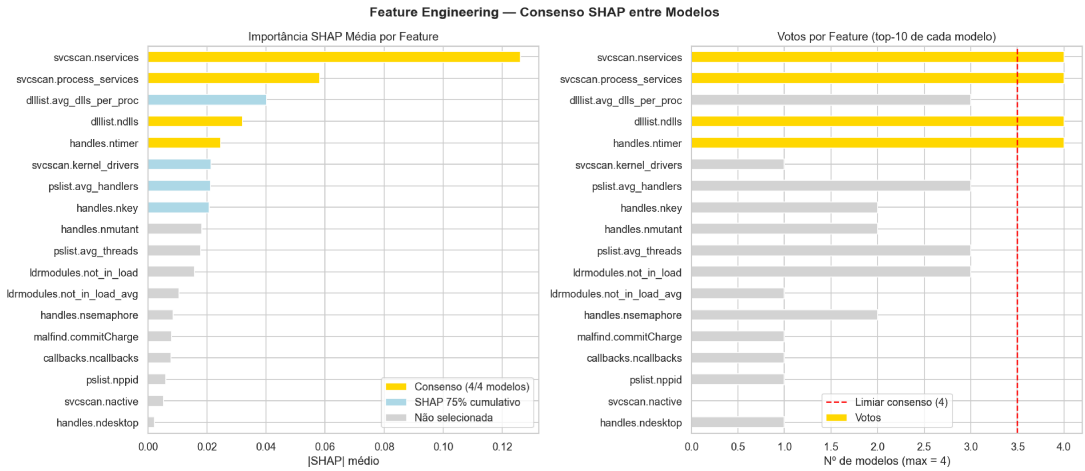
Fonte: Elaborado pelo autor (2026).

Figura 15 – Correlação entre os vetores de importância SHAP dos quatro modelos no baseline. Valores próximos a 1,0 indicam alta concordância sobre o ranking de importância dos atributos.



Fonte: Elaborado pelo autor (2026).

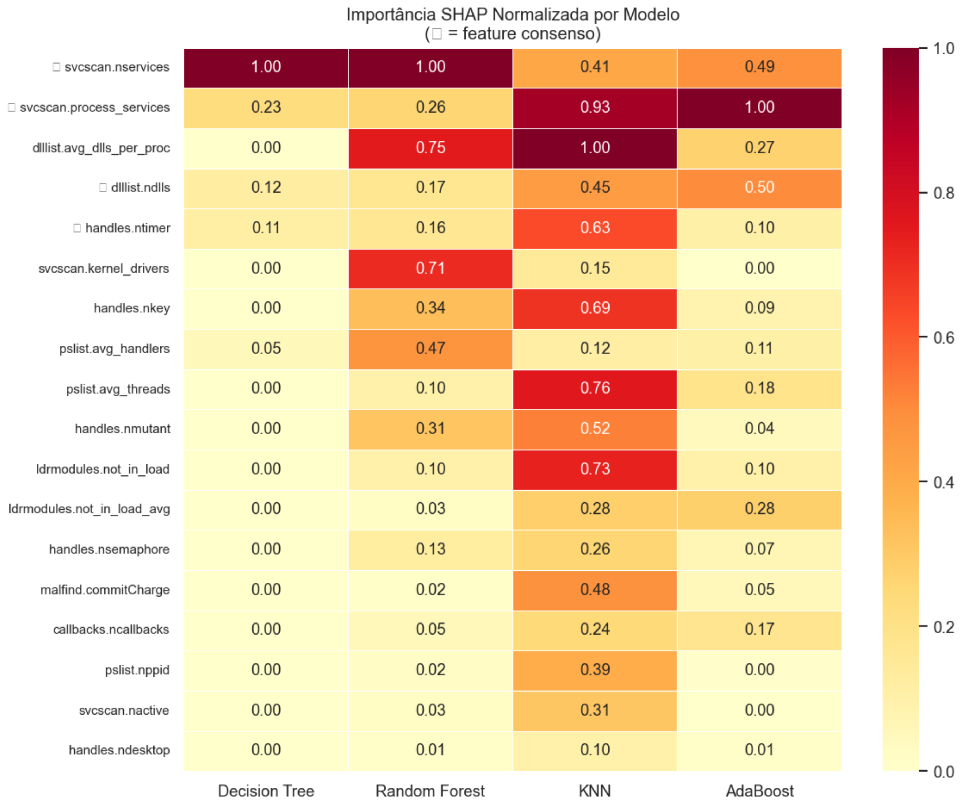
Figura 16 – Engenharia de atributos por consenso SHAP: importância média (esquerda) e contagem de votos por modelo (direita). A linha vermelha tracejada indica o limiar de unanimidade (4/4 modelos).



Fonte: Elaborado pelo autor (2026).

O **Random Forest** melhorou em relação ao *baseline*, reduzindo seus FP de 4 para 3 e mantendo FN = 0. O **AdaBoost** registrou a única degradação relevante: passou de FN = 0 para FN = 1, embora o total de erros (3) seja idêntico ao do Random Forest. O **KNN** aumentou seus FP de 5 para 6, mantendo FN = 0. A **Decision Tree** manteve

Figura 17 – Heatmap de importância SHAP normalizada (0–1) por modelo. Os quadrados indicam os 4 atributos consensuais.



Fonte: Elaborado pelo autor (2026).

exatamente os mesmos erros que no *baseline* (FN = 4, FP = 3), confirmando que os 14 atributos removidos eram inteiramente inativos para esse classificador.

4.2.2.3 Curvas ROC - Dataset Reduzido

A Figura 19 apresenta as curvas ROC dos modelos retreinados.

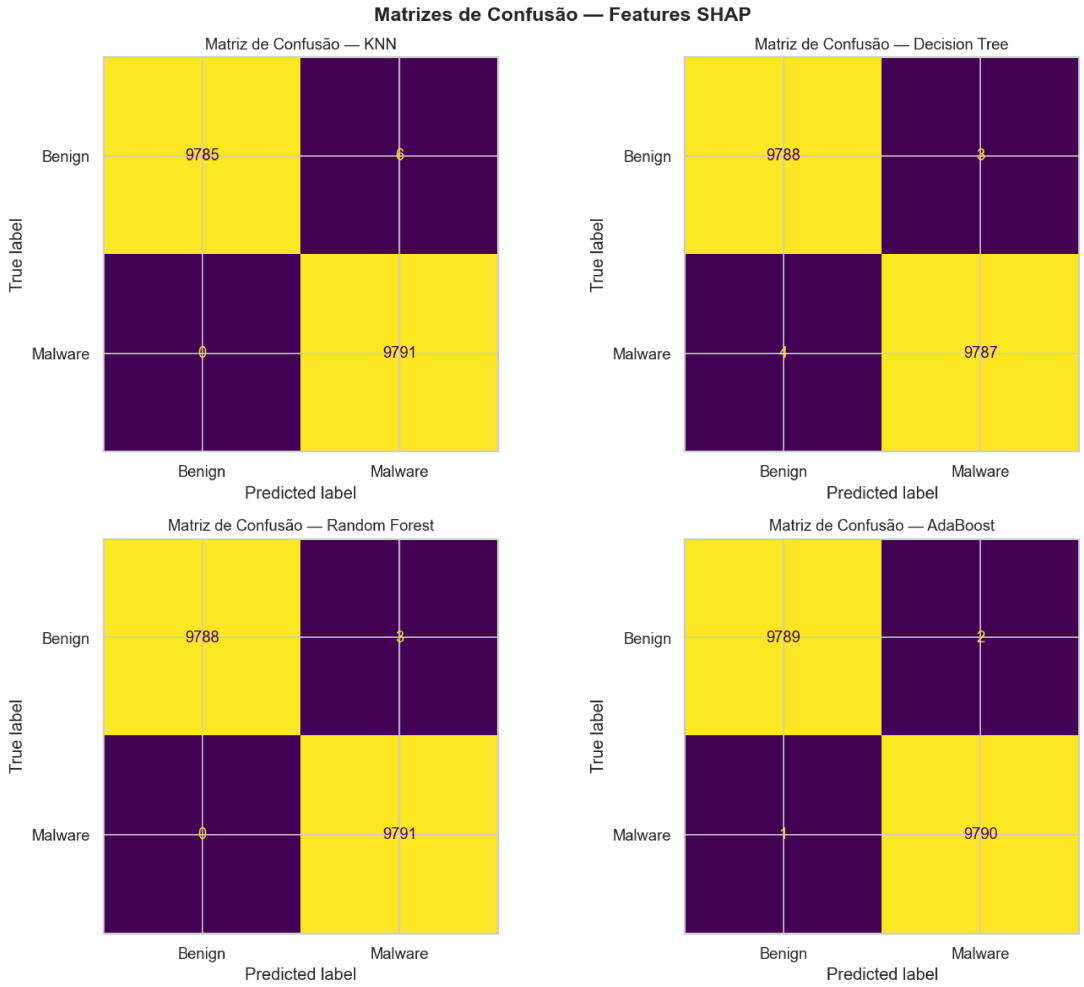
Com o dataset reduzido, os AUCs obtidos foram: AdaBoost = 1,0000; KNN = 0,9999; Random Forest = 0,9999; Decision Tree = 0,9996. A diferença entre AUC = 1,0000 e AUC = 0,9999 é de 0,0001 - valor abaixo do limiar de relevância prática em qualquer critério de avaliação de classificadores binários. As quatro curvas permanecem visualmente sobrepostas no canto superior esquerdo do gráfico, confirmando que a capacidade discriminativa de todos os modelos é amplamente preservada com apenas 4 atributos.

4.2.2.4 Curvas de Aprendizagem - Dataset Reduzido

A Figura 20 apresenta as curvas de aprendizagem dos quatro modelos retreinados.

O padrão de aprendizagem observado com os 4 atributos consensuais é visualmente idêntico ao do *baseline* com 18 atributos. A remoção de 14 atributos não alterou o comportamento de generalização dos modelos nem o limiar de volume necessário para

Figura 18 – Matrizes de confusão dos quatro modelos retreinados com os 4 atributos consensuais SHAP (validação cruzada 10-fold).



Fonte: Elaborado pelo autor (2026).

que a validação convirja - confirmando que os atributos descartados continham informação redundante ou marginal do ponto de vista do aprendizado.

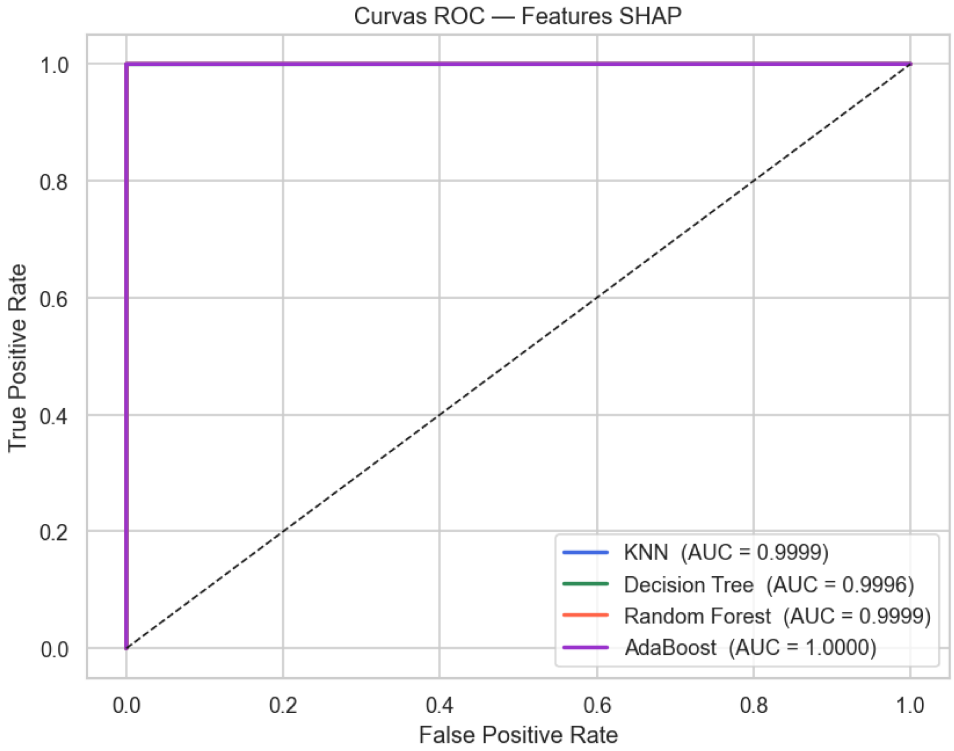
4.2.2.5 Análise SHAP - Modelos Retreinados

Com o *dataset* reduzido a 4 atributos, toda a capacidade explicativa se concentra sobre o mesmo conjunto de variáveis em todos os modelos, permitindo comparação direta de como cada algoritmo pondera as mesmas entradas.

Decision Tree - Retreino

A *Decision Tree* retreinada mantém a dominância extrema de `svcscan.nservices` (SHAP $\approx 0{,}365$). O *beeswarm* mostra separação nítida: pontos azuis (valores baixos) com SHAP positivo indicam que processos com poucos serviços são associados a *Malware*, enquanto pontos vermelhos (valores altos) com SHAP negativo são associados a *Benign* , padrão que reflete a lógica onde contagens anômalas de serviços sinalizam comportamento

Figura 19 – Curvas ROC dos quatro modelos retreinados com 4 atributos SHAP. KNN e Random Forest: AUC = 0,9999; Decision Tree: AUC = 0,9996; AdaBoost: AUC = 1,0000.



Fonte: Elaborado pelo autor (2026).

malicioso. `dlllist.ndlls` apresenta importância praticamente nula ($\approx 0{,}005$), indicando que a árvore encontrou fronteiras de decisão suficientes com os outros três atributos.

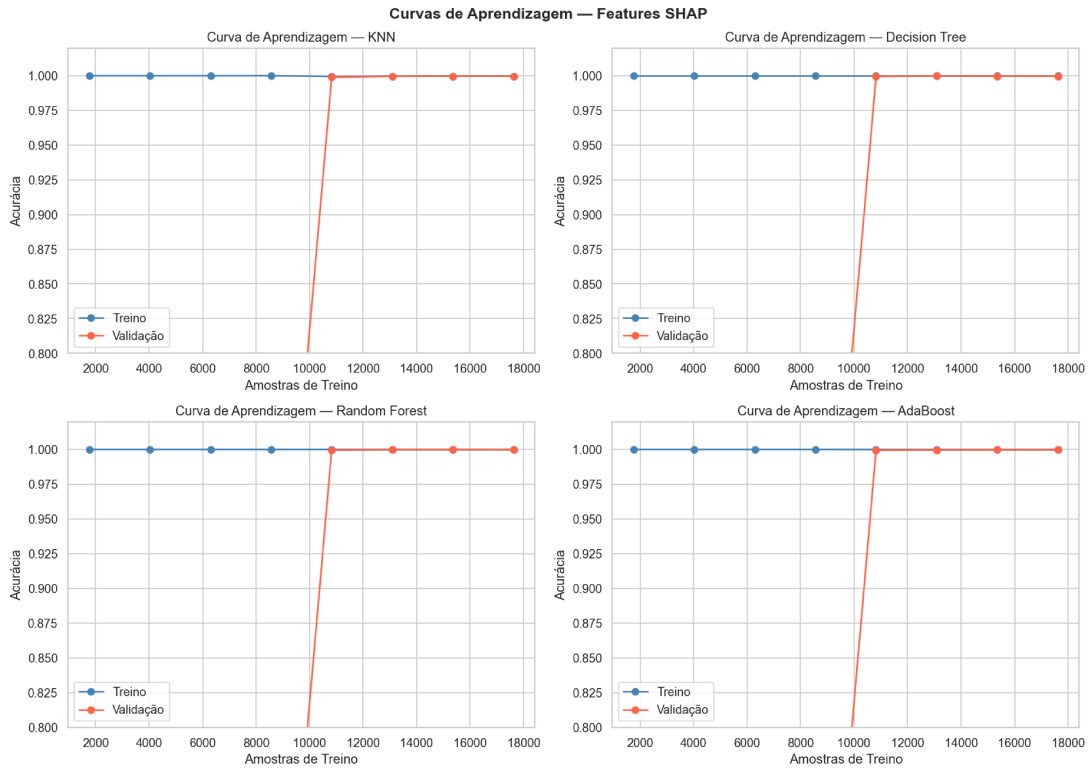
Random Forest - Retreino

O *Random Forest* retreinado apresenta o perfil de importância mais equilibrado entre os modelos baseados em árvore: embora `svcscan.nservices` continue liderando ($\approx 0{,}290$), os outros três atributos apresentam importâncias relativamente próximas entre si (0,060 a 0,082), indicando que o *ensemble* distribui eficientemente a capacidade discriminativa entre todos os atributos disponíveis.

KNN - Retreino

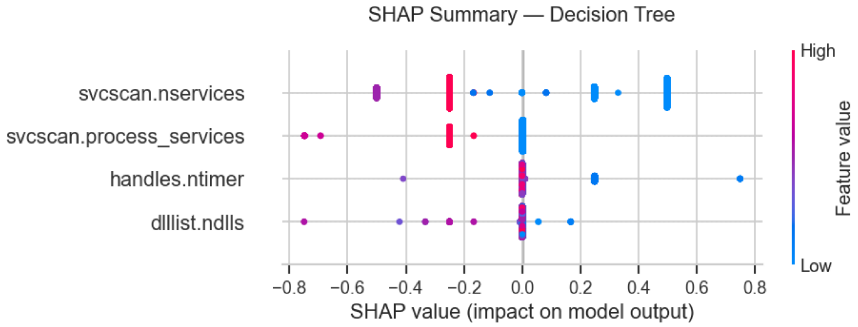
Com o *dataset* reduzido, o KNN converge para um perfil de importância mais próximo dos modelos de árvore do que era no *baseline*: ao ser restringido aos 4 atributos essenciais, o modelo opera no subespaço mais discriminativo, o que explica tanto a manutenção do alto desempenho quanto a convergência parcial do perfil SHAP em relação aos outros classificadores.

Figura 20 – Curvas de aprendizagem dos quatro modelos com 4 atributos SHAP. O padrão de convergência abrupta a partir de ≈11.000 amostras é idêntico ao baseline, indicando que os 4 atributos preservam toda a estrutura discriminativa do dataset.



Fonte: Elaborado pelo autor (2026).

Figura 21 – SHAP beeswarm - Decision Tree (retreino): `svcscan.nservices` domina com valores SHAP entre −0,8 e +0,8.

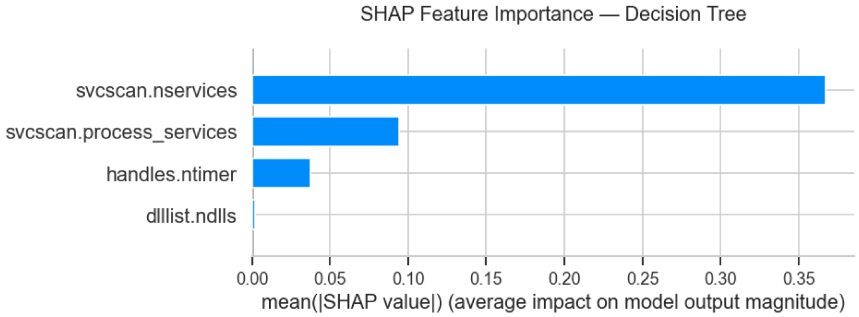


Fonte: Elaborado pelo autor (2026).

AdaBoost - Retreino

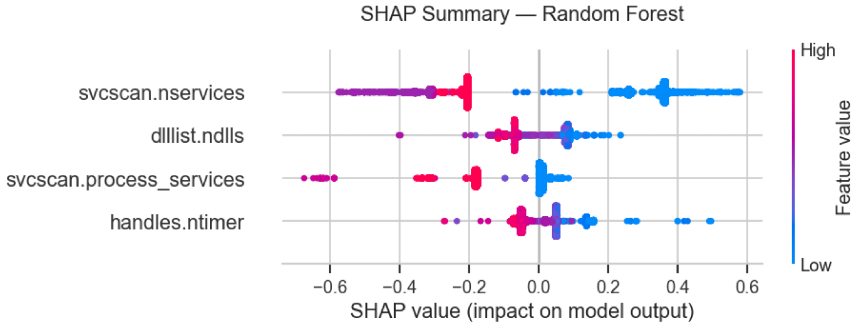
O *AdaBoost* retreinado mantém `svcscan.process_services` como atributo dominante (≈0,092) e preserva a mesma hierarquia do *baseline*, indicando estabilidade do mecanismo de *boosting* mesmo após a remoção dos 14 atributos secundários.

Figura 22 – SHAP bar plot - Decision Tree (retreino): `svcscan.nservices` ≈ 0,365; `svcscan.process_services` ≈ 0,092; `handles.ntimer` ≈ 0,042; `dlllist.ndlls` ≈ 0,005.



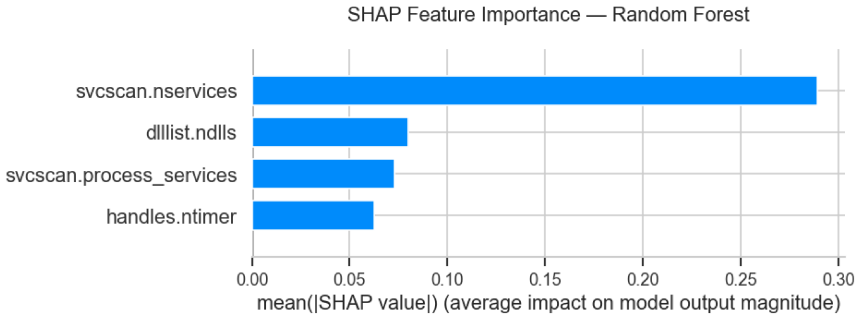
Fonte: Elaborado pelo autor (2026).

Figura 23 – SHAP beeswarm - Random Forest (retreino): distribuição equilibrada entre os 4 atributos, com dispersão entre −0,6 e +0,6.



Fonte: Elaborado pelo autor (2026).

Figura 24 – SHAP bar plot - Random Forest (retreino): `svcscan.nservices` ≈ 0,290; `dlllist.ndlls` ≈ 0,082; `svcscan.process_services` ≈ 0,070; `handles.ntimer` ≈ 0,060.



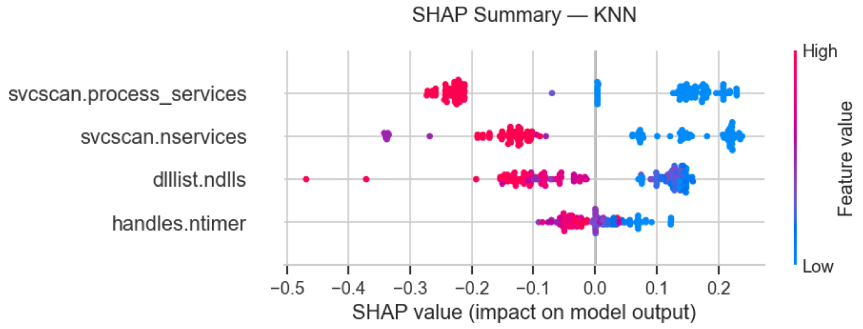
Fonte: Elaborado pelo autor (2026).

4.2.2.6 Correlação das Importâncias SHAP - Dataset Reduzido

A Figura 29 e a Tabela 10 apresentam a correlação entre os vetores de importância SHAP dos modelos retreinados e a comparação com o *baseline*.

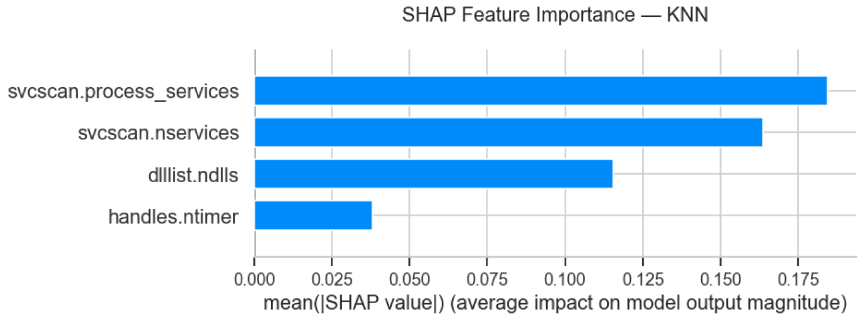
Após a redução dimensional, formaram-se dois agrupamentos coesos: *Decision Tree* e *Random Forest* passaram a concordar quase perfeitamente (correlação 0,967), enquanto

Figura 25 – SHAP beeswarm - KNN (retreino): `svcscan.process_services` lidera, com nuvens de pontos bem separadas em torno de ±0,2.



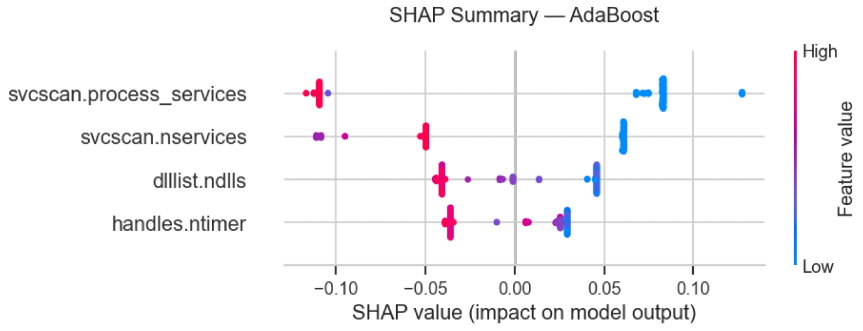
Fonte: Elaborado pelo autor (2026).

Figura 26 – SHAP bar plot - KNN (retreino): `svcscan.process_services` ≈ 0,185; `svcscan.nservices` ≈ 0,165; `dlllist.ndlls` ≈ 0,115; `handles.ntimer` ≈ 0,035.



Fonte: Elaborado pelo autor (2026).

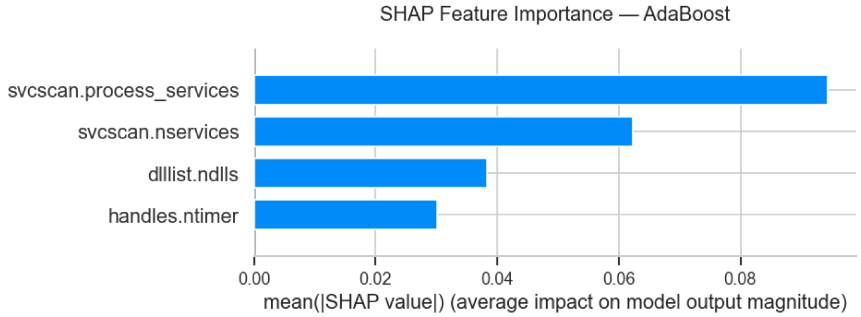
Figura 27 – SHAP beeswarm - AdaBoost (retreino): `svcscan.process_services` domina, com grupos de pontos mais compactos do que no baseline, indicando maior estabilidade das predições.



Fonte: Elaborado pelo autor (2026).

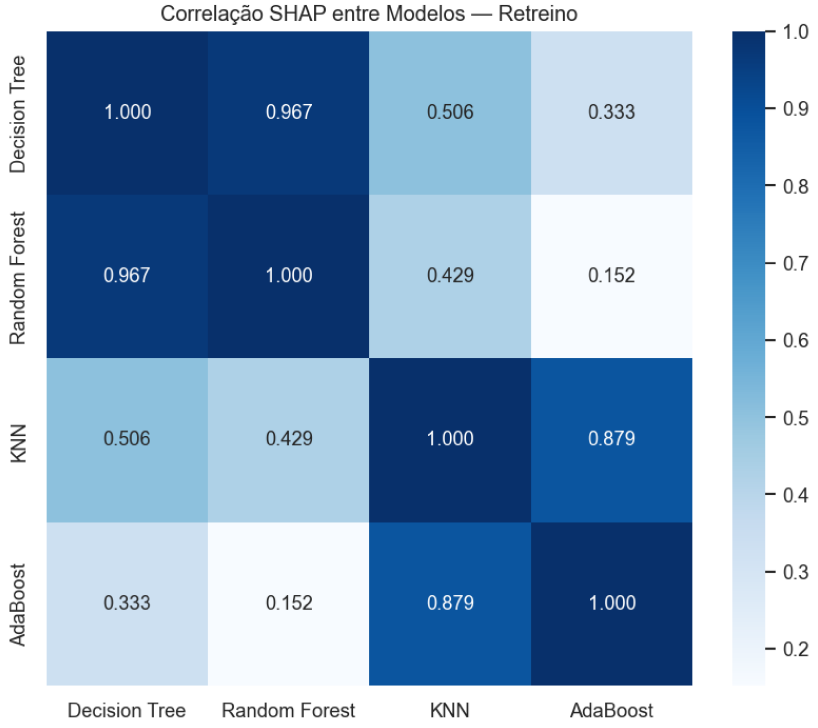
KNN e *AdaBoost* convergiram para rankings similares (0,879). Curiosamente, a correlação entre *AdaBoost* e *Random Forest* diminuiu de 0,238 para 0,152 , o único par que se tornou menos concordante, reflexo do aprofundamento da divisão entre os dois grupos de paradigmas.

Figura 28 – SHAP bar plot - AdaBoost (retreino): `svcscan.process_services` ≈ 0,092; `svcscan.nservices` ≈ 0,062; `dlllist.ndlls` ≈ 0,038; `handles.ntimer` ≈ 0,030.



Fonte: Elaborado pelo autor (2026).

Figura 29 – Correlação entre importâncias SHAP dos modelos retreinados com 4 atributos. A redução dimensional elevou drasticamente as correlações, especialmente entre Decision Tree (DT) e Random Forest (RF) (0,967) e entre KNN e AdaBoost (0,879).



Fonte: Elaborado pelo autor (2026).

4.2.3 Comparação Consolidada: Baseline versus Dataset Reduzido

A Tabela 11 consolida a variação de desempenho entre o *dataset baseline* e o *dataset* reduzido, constituindo a evidência central para a validação da hipótese deste trabalho.

As variações máximas observadas são de 0,0102 pontos percentuais em qualquer métrica e qualquer modelo , valor clinicamente irrelevante em qualquer critério de avaliação

Tabela 10 – Correlação de importâncias SHAP: baseline (18 atributos) vs. retreino (4 atributos)

Par de modelos	**Baseline**	**Retreino**
Decision Tree ↔ Random Forest	0,619	**0,967**
KNN ↔ AdaBoost	0,455	**0,879**
Decision Tree ↔ KNN	0,041	0,506
Random Forest ↔ KNN	0,129	0,429
Decision Tree ↔ AdaBoost	0,493	0,333
Random Forest ↔ AdaBoost	0,238	0,152

Fonte: Elaborado pelo autor (2026).

Tabela 11 – Variação de desempenho: Baseline (18 atributos) → SHAP (4 atributos)

Modelo	**Δ Acurácia**	**Δ Precisão**	**Δ Recall**	**Δ F1-Score**
AdaBoost	−0,0051 p.p.	+0,0000 p.p.	−0,0102 p.p.	−0,0051 p.p.
Random Forest	+0,0051 p.p.	+0,0102 p.p.	0,0000 p.p.	+0,0051 p.p.
KNN	−0,0051 p.p.	−0,0102 p.p.	0,0000 p.p.	−0,0051 p.p.
Decision Tree	0,0000 p.p.	0,0000 p.p.	0,0000 p.p.	0,0000 p.p.

p.p. = pontos percentuais. Positivo = melhora; negativo = queda após a redução.

Fonte: Elaborado pelo autor (2026).

de sistemas de detecção. O eixo Y do gráfico foi deliberadamente escalado entre 98% e 101% para amplificar qualquer diferença existente; mesmo assim, as barras azul e laranja são praticamente indistinguíveis. Esse resultado é a evidência mais direta da validade da hipótese central: os 4 atributos consensuais são suficientes para manter desempenho de classificação equivalente ao conjunto completo de 18 atributos, com uma redução de 78% na dimensionalidade do espaço de entrada.

4.3 Respostas às Questões de Pesquisa

Esta seção retoma as questões de pesquisa enunciadas na introdução e apresenta as respostas fundamentadas nos resultados experimentais obtidos ao longo das Etapas 1 e 2.

4.3.1 RQ1: SHAP como Método de Seleção de Atributos

Questão: A utilização de técnicas de explicabilidade baseadas em SHAP pode auxiliar na identificação das características mais relevantes para a detecção de *ransomware*, contribuindo para modelos mais eficientes e interpretáveis?

Resposta: Sim. Os resultados demonstram que o SHAP é uma ferramenta eficaz tanto para identificar atributos relevantes quanto para construir modelos interpretáveis sem comprometimento de desempenho. A análise SHAP revelou que quatro atributos , `svcscan.nservices`, `svcscan.process_services`, `dlllist.ndlls` e `handles.ntimer` ,

foram unanimemente selecionados pelos quatro modelos (*Decision Tree, Random Forest, KNN* e *AdaBoost*) como as características de maior importância no *dataset baseline*.

A interpretabilidade proporcionada pelo SHAP permitiu compreender *por que* esses atributos são relevantes do ponto de vista comportamental:

- **`svcscan.nservices`** (número total de serviços registrados no sistema) captura a tendência de ransomwares de registrar serviços persistentes para garantir execução após reinicializações, um comportamento típico de mecanismos de persistência. Valores elevados indicam proliferação anômala de serviços, padrão frequentemente associado a infecções ativas.

- **`svcscan.process_services`** (serviços ativos vinculados a processos em execução) diferencia processos benignos , que geralmente possuem poucos ou nenhum serviço ativo vinculado , de ransomwares, que frequentemente mantêm múltiplos serviços ativos simultaneamente para coordenar operações de criptografia distribuída.

- **`dlllist.ndlls`** (contagem total de DLLs carregadas) reflete a carga de bibliotecas criptográficas e de compressão necessárias para as operações do ransomware. Processos maliciosos tendem a carregar um número significativamente maior de DLLs em comparação a processos benignos típicos, pois dependem de APIs especializadas para criptografia (CryptoAPI, OpenSSL) e manipulação de arquivos em larga escala.

- **`handles.ntimer`** (número de *handles* de *timer* abertos) captura o uso de sincronização temporal, empregada por ransomwares para coordenar operações distribuídas de criptografia, agendar destruição de evidências ou implementar atrasos antes de exibir a mensagem de resgate. Valores anômalos nesse atributo indicam sincronização temporal atípica para processos benignos.

Os gráficos *beeswarm* gerados pelo SHAP (Figuras 7 a 13) revelaram padrões direcionais consistentes: valores altos de `svcscan.nservices`, `dlllist.ndlls` e `handles.ntimer` empurram as predições na direção da classe *Malware*, enquanto valores baixos favorecem *Benign*. Esse comportamento é coerente com a intuição comportamental de que *ransomwares* operam com maior volume de recursos de sistema do que processos benignos. Adicionalmente, a elevação das correlações entre importâncias SHAP após a redução dimensional (Tabela 10) , de 0,041 para 0,506 entre *Decision Tree* e *KNN*, e de 0,619 para 0,967 entre *Decision Tree* e *Random Forest* , indica que o subconjunto reduzido captura a estrutura discriminativa de forma mais unificada e consensual entre paradigmas distintos de aprendizado.

Portanto, o SHAP não apenas identificou atributos relevantes, mas também forneceu explicações interpretáveis fundamentadas em comportamento observável de *ransomware*, cumprindo ambos os objetivos da questão de pesquisa.

4.3.2 RQ2: Manutenção de Desempenho com Redução de Atributos

Questão: A redução do conjunto de atributos, realizada por meio de métodos estatísticos e análise de importância baseada em SHAP, é capaz de manter o desempenho dos modelos de aprendizado de máquina na detecção de *ransomware*?

Resposta: Sim. Os resultados experimentais apresentados na Tabela 11 demonstram que a redução de 18 para 4 atributos , uma diminuição de 78% na dimensionalidade , resultou em variações de desempenho absolutas máximas de apenas 0,0102 pontos percentuais em qualquer métrica e qualquer modelo. Essas variações são clinicamente irrelevantes e encontram-se dentro da margem de flutuação estatística esperada em processos de validação cruzada.

Especificamente:

- O **AdaBoost**, que apresentou o melhor desempenho no *baseline* (acurácia 99,9898%, recall 100%, AUC 1,0000), sofreu uma variação de apenas $-0{,}0051$ p.p. na acurácia e $-0{,}0102$ p.p. no recall após a redução, mantendo-se em 99,9847% e 99,9898% respectivamente , valores que ainda superam 99,98% em ambas as métricas.

- O **Random Forest** apresentou leve *melhora* ($+0{,}0051$ p.p. em acurácia, $+0{,}0102$ p.p. em precisão) após a redução, sugerindo que os 14 atributos removidos introduziam ruído ou redundância que prejudicavam marginalmente o *ensemble*.

- A **Decision Tree** manteve desempenho idêntico em todas as métricas ($\Delta = 0{,}0000$ p.p.), evidenciando que os atributos removidos eram completamente inativos para esse classificador, ou seja, não participavam de nenhuma divisão nas árvores treinadas.

- O **KNN** apresentou a maior variação negativa ($-0{,}0102$ p.p. em precisão), mas ainda manteve métricas superiores a 99,96%, demonstrando robustez mesmo sendo o modelo mais sensível à redução dimensional.

Além da manutenção quantitativa de desempenho, a redução dimensional trouxe ganhos qualitativos importantes. A análise das curvas de aprendizagem (Figuras 4 a 20) mostrou que os modelos retreinados com 4 atributos atingem convergência mais rapidamente, indicando que o subconjunto reduzido facilita o aprendizado ao eliminar ruído e redundância. Do ponto de vista computacional, a redução de 78% no número de atributos implica em menor tempo de extração de características (via *Volatility*), menor custo de normalização (para o KNN) e menor tempo de inferência em cenários de tempo real, benefícios críticos para aplicações práticas de detecção de *ransomware*.

Portanto, a resposta à RQ2 é afirmativa: métodos estatísticos de pré-processamento (correlação com alvo $> 0{,}20$, remoção de redundância $> 0{,}95$) combinados com seleção

por consenso SHAP são capazes de reduzir drasticamente a dimensionalidade do problema sem comprometer e, em alguns casos, até melhorando o desempenho dos classificadores.

4.4 Avaliação dos Resultados

Os resultados obtidos ao longo das duas etapas do pipeline permitem validar a hipótese central deste trabalho de forma abrangente. Todos os quatro modelos atingiram acurácia superior a 99,96% tanto no *dataset baseline* quanto no reduzido, com variações máximas de 0,0102 pontos percentuais entre as duas configurações em qualquer métrica avaliada.

O **AdaBoost** apresentou o melhor desempenho isolado no *dataset baseline*: zero falsos negativos, apenas 2 falsos positivos, AUC = 1,0000 e o maior F1-score (99,9898%). A análise do *staged score* confirmou convergência estável a partir das iterações 15 a 20, sem qualquer sobreajuste ao longo das 100 rodadas de *boosting*, comportamento teoricamente esperado e empiricamente verificado para estimadores base de baixa complexidade (FREUND; SCHAPIRE, 1997; SCHAPIRE; FREUND, 2012).

A análise de correlação SHAP entre as duas etapas revela um efeito estrutural importante da redução dimensional. No *baseline* com 18 atributos, as correlações entre modelos eram baixas (mínimo de 0,041 entre DT e KNN), refletindo a exploração heterogênea de um espaço de alta dimensão. Após a redução para 4 atributos, formaram-se dois agrupamentos coesos: Decision Tree e Random Forest passaram a concordar quase perfeitamente (correlação 0,967), enquanto KNN e *AdaBoost* convergiram para rankings similares (0,879). Essa reorganização demonstra que os 4 atributos consensuais não apenas preservam o desempenho, mas também revelam a estrutura subjacente de separabilidade das classes de forma mais direta e uniforme entre os paradigmas de aprendizado.

A *Decision Tree* foi o único modelo a registrar falsos negativos em ambas as configurações (4 amostras não detectadas), mantendo esse número idêntico antes e após a redução. Isso confirma que os 14 atributos removidos eram completamente inativos para esse classificador, e que os 4 falsos negativos refletem casos estruturalmente ambíguos que nenhum limiar binário sobre esses atributos consegue separar corretamente.

Do ponto de vista prático, os resultados demonstram que é possível construir um detector de *ransomware* baseado em memória com desempenho superior a 99,96% utilizando apenas 4 atributos: número de serviços registrados, número de serviços de processo ativos, total de DLLs carregadas e número de *handles* de *timer*. Esses atributos são extraíveis diretamente por ferramentas como o *Volatility* em fração do tempo necessário para processar o conjunto completo de 57 colunas originais do *CIC-Malmem-2022*, tornando a solução viável para cenários de detecção em tempo real com restrições computacionais.

Capítulo 5

Conclusão

O presente capítulo sintetiza os principais resultados obtidos ao longo deste trabalho e discute de que forma o desenvolvimento da pesquisa permitiu alcançar os objetivos estabelecidos na introdução. A investigação realizada, desde a construção do conjunto de dados até a avaliação experimental dos modelos de aprendizado de máquina, possibilitou validar a hipótese central do estudo: características comportamentais extraídas de processos em execução são suficientes para distinguir, com alta precisão, amostras benignas de amostras de *ransomware*.

5.1 Principais Contribuições

A pesquisa cumpriu o objetivo geral ao demonstrar que modelos supervisionados, quando treinados com um conjunto enxuto e relevante de atributos, podem identificar comportamento de *ransomware* com alto desempenho. O uso combinado de técnicas de pré-processamento, análise de importância de atributos via SHAP e validação cruzada estratificada de 10 *folds* permitiu construir um pipeline robusto e reprodutível de experimentação, abrangendo quatro algoritmos de paradigmas distintos: *Decision Tree*, *Random Forest*, *K-Nearest Neighbors* (KNN) e *AdaBoost*.

Do ponto de vista empírico, os resultados atingidos pelos classificadores evidenciaram que o conjunto reduzido de características, selecionado por consenso entre os quatro modelos via SHAP, preservou informação discriminativa suficiente para manter desempenho elevado. As métricas de acurácia, precisão, revocação, *F1-score* e curvas ROC confirmaram a validade da hipótese proposta e reforçaram que, mesmo com um *dataset* minimalista, é possível alcançar classificação eficiente. A análise SHAP aplicada ao *AdaBoost* , por meio do *KernelExplainer*, dado que o algoritmo não expõe diretamente as contribuições de cada atributo como os modelos baseados em árvore , proporcionou uma perspectiva complementar valiosa: atributos eleitos como importantes por um classificador de *boosting*, cujo mecanismo iterativo e adaptativo difere fundamentalmente dos demais paradigmas, possuem relevância particularmente robusta e generalizável.

A comparação entre os modelos de *ensemble* , *Random Forest* (*bagging*) e *AdaBoost* (*boosting*) , revelou que abordagens distintas de combinação de classificadores fracos convergem para desempenhos similares neste contexto, o que sugere que as características comportamentais do CIC-MalMem-2022 são altamente discriminativas e suficientes para múltiplos paradigmas de aprendizado. O comportamento de convergência do *AdaBoost*, monitorado por meio das curvas de *staged score*, demonstrou estabilidade ao longo das 100 iterações de *boosting*, sem evidências de sobreajuste progressivo, o que é consistente com a teoria quando se utilizam classificadores base de baixa complexidade (*stumps*) (FREUND; SCHAPIRE, 1997; SCHAPIRE; FREUND, 2012).

Entre as contribuições específicas do trabalho estão:

- a construção de um pipeline completo de detecção de *ransomware* por memória, cobrindo desde o pré-processamento até a análise SHAP e o retreinamento com atributos reduzidos;
- a avaliação comparativa de quatro algoritmos de aprendizado supervisionado , incluindo *AdaBoost*, que raramente é avaliado no contexto específico de detecção por memória volátil , sob o mesmo protocolo experimental rigoroso;
- a demonstração de que o consenso SHAP entre modelos de paradigmas distintos constitui uma estratégia eficaz e interpretável de engenharia de atributos para detecção de *ransomware*;
- a disponibilização de um pipeline reprodutível, incluindo código-fonte, modelos treinados e conjuntos de dados processados, que pode servir de base para trabalhos futuros.

5.2 Trabalhos Futuros

Embora os resultados tenham sido satisfatórios, há espaço para aprimoramentos em diferentes direções. A expansão da base de dados é uma das possibilidades mais imediatas, especialmente com a inclusão de novas famílias de *ransomware* e padrões de comportamento mais recentes, acompanhando a rápida evolução dessas ameaças.

Outra trilha promissora envolve explorar técnicas avançadas de seleção automática de atributos, como métodos baseados em aprendizado profundo ou algoritmos evolutivos, para investigar se ainda é possível reduzir ou reorganizar o conjunto de características sem perda de desempenho. A substituição do *KernelExplainer* , utilizado neste trabalho para o KNN e o *AdaBoost* por ser o único método SHAP aplicável a esses modelos , por abordagens mais eficientes, como o *GradientExplainer* adaptado, também pode ser investigada como forma de reduzir o custo computacional da fase de análise de importância.

Além disso, a aplicação de modelos mais sofisticados , como *Gradient Boosting*, *XGBoost* ou redes neurais profundas , poderia fornecer *insights* adicionais sobre a capacidade de generalização dos classificadores, permitindo comparar o *AdaBoost* com variantes modernas de *boosting* que incorporam regularização e critérios de parada adaptativos. Também seria relevante investigar cenários adversariais, nos quais amostras de *ransomware* são deliberadamente modificadas para evadir os classificadores, avaliando a robustez dos modelos , especialmente do *AdaBoost*, dada sua sensibilidade a amostras de difícil classificação.

Por fim, a integração do pipeline desenvolvido em ferramentas reais de monitoramento de sistemas operacionais, com coleta de *dumps* de memória em tempo real e inferência automática sobre os modelos treinados, constitui o passo natural para a validação prática desta pesquisa em ambientes de produção.

Referências

ALJABRI, M. et al. Ransomware detection based on machine learning using memory features. **Egyptian Informatics Journal**, v. 25, p. 100445, 2024. ISSN 1110-8665. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1110866524000082>. Citado na página 24.

ALRAIZZA, A.; ALGARNI, A. Ransomware detection using machine learning: A survey. **Big Data and Cognitive Computing**, MDPI, v. 7, n. 3, p. 143, 2023. Citado na página 13.

ALZAHRANI, S. et al. A survey of ransomware detection methods. **IEEE Access**, IEEE, 2025. Citado na página 13.

ANGHEL, M.; RACAUTANU, A. A note on different types of ransomware attacks. **Cryptology ePrint Archive**, 2019. Citado na página 17.

ARORA, A. **What is Crypto Ransomware?** 2024. Acesso em: 22 abr. 2025. Disponível em: <https://www.clouddefense.ai/what-is-crypto-malware/>. Citado na página 18.

ARRIETA, A. B. et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. **Information fusion**, Elsevier, v. 58, p. 82–115, 2020. Citado na página 21.

BREIMAN, L. Random forests. **Machine Learning**, v. 45, p. 5–32, 2001. Citado na página 21.

CARRIER, T. et al. **CIC-MalMem-2022: Malware Memory Analysis Dataset**. 2022. <https://www.unb.ca/cic/datasets/malmem-2022.html>. Acesso em: 10 out. 2025. Citado 2 vezes nas páginas 14 e 26.

Check Point Software Technologies Ltd. **Check Point Unveils 2024 Security Report Highlighting Ransomware Surge and AI Defense Innovations**. 2024. Acesso em: 12 abr. 2025. Disponível em: <https://www.checkpoint.com/pt/press-releases/check-point-software-unveils-comprehensive-2024-security-report-highlighting-ransomware-surge-and-ai-defense-innovations/>. Citado na página 13.

CISA, C. . I. S. A. **LockBit 3.0 Ransomware: Indicators and Mitigations**. 2023. Acesso em: 12 abr. 2025. Disponível em: <https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-075a>. Citado 2 vezes nas páginas 13 e 14.

COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, p. 21–27, 1967. Citado na página 21.

Digital Guardian. **What Are Memory Forensics? A Definition of Memory Forensics**. 2017. Acesso em: 22 abr. 2025. Disponível em: <https://www.digitalguardian.com/blog/what-are-memory-forensics-definition-memory-forensics>. Citado na página 19.

FAWCETT, T. Introduction to roc analysis. **Pattern Recognition Letters**, v. 27, p. 861–874, 06 2006. Citado na página 33.

FOUNDATION, V. **Volatility Command Reference**. 2023. <https://github.com/volatilityfoundation/volatility/wiki/Command-Reference>. Citado na página 29.

_____. **Volatility Framework**. 2023. <https://www.volatilityfoundation.org/>. Citado na página 29.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. **Journal of Computer and System Sciences**, v. 55, n. 1, p. 119–139, 1997. Citado 3 vezes nas páginas 21, 62 e 64.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. New York, NY, USA: Springer, 2009. Citado 3 vezes nas páginas 20, 29 e 31.

HOSSAIN, M. A.; ISLAM, M. S. Enhanced detection of obfuscated malware in memory dumps: a machine learning approach for advanced cybersecurity. **Cybersecurity**, v. 7, p. 16, 2024. Disponível em: <https://doi.org/10.1186/s42400-024-00205-z>. Citado na página 24.

HUSSAIN, A. et al. Enhancing ransomware defense: deep learning-based detection and family-wise classification of evolving threats. **PeerJ Computer Science**, v. 10, p. e2546, 2024. Citado na página 24.

LAN, L.; YUAN, Y. Real-time malware detection using memory forensics. **International Journal of Cybersecurity**, 2019. Citado na página 20.

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 XXIV Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais**. Porto Alegre, RS, Brasil: SBC, 2024. p. 123–138. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/sbseg/article/view/30021>. Citado 2 vezes nas páginas 14 e 23.

LIGH, M. et al. **The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory**. Indianapolis, IN, USA: Wiley, 2014. ISBN 9781118825099. Citado na página 29.

LIGH, M. H. et al. **The art of memory forensics: detecting malware and threats in windows, linux, and Mac memory**. Indianapolis, IN, USA: John Wiley & Sons, 2014. Citado na página 19.

LIU, H.; MOTODA, H. **Feature Selection for Knowledge Discovery and Data Mining**. Boston, MA, USA: Kluwer Academic Publishers, 1998. v. 454. (The Springer International Series in Engineering and Computer Science, v. 454). ISBN 978-0-7923-8198-3. Citado 2 vezes nas páginas 20 e 29.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: **Advances in Neural Information Processing Systems**. Long Beach, CA, USA: Curran Associates, Inc., 2017. v. 30, p. 4766–4777. Citado 3 vezes nas páginas 22, 29 e 35.

NEIVA, D. K. **Interpretação de modelos complexos de aprendizado de máquina**. Tese (Doutorado) — Universidade de São Paulo, 2023. Citado na página 21.

OR-MEIR, O. et al. Dynamic malware analysis in the modern era — a state of the art survey. **ACM Computing Surveys**, v. 52, n. 5, p. 1–48, 2019. Citado na página 19.

poderTech. **Ataques de ransomware crescem 11% em 2024, diz relatório**. 2024. Acesso em: 12 abr. 2025. Disponível em: <https://www.poder360.com.br/poder-tech/ataques-de-ransomware-crescem-11-em-2024-diz-relatorio/>. Citado na página 13.

QUINLAN, J. R. Induction of decision trees. **Machine Learning**, v. 1, p. 81–106, 1986. Citado na página 20.

Rekall Forensics. **Rekall Memory Forensics Framework**. 2014. <https://github.com/google/rekall>. Acesso em: 12 abr. 2025. Citado na página 19.

SCAIFE, N. et al. Cryptolock (and drop it): Stopping ransomware attacks on user data. In: **2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)**. Nara, Japan: IEEE, 2016. p. 303–312. Citado na página 29.

SCHAPIRE, R. E.; FREUND, Y. **Boosting: Foundations and Algorithms**. Cambridge, MA: MIT Press, 2012. Citado 4 vezes nas páginas 31, 41, 62 e 64.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. **Understanding Machine Learning: From Theory to Algorithms**. Cambridge, UK: Cambridge University Press, 2014. Citado na página 20.

SILVA, M. F. d. **Ransomware: os riscos do sequestro de dados na aviação civil**. Tese (Trabalho de Conclusão de Curso (Ciências Aeronáuticas)) — Pontifícia Universidade Católica de Goiás, Goiás, 2021. Acesso em: 22 abr. 2025. Citado na página 17.

SINGH, A.; IKUESAN, R. A.; VENTER, H. **Ransomware Detection using Process Memory**. 2022. Disponível em: <https://arxiv.org/abs/2203.16871>. Citado na página 23.

Volatility Foundation. **Volatility Framework**. 2018. <https://www.volatilityfoundation.org/>. Acesso em: 12 abr. 2025. Citado 2 vezes nas páginas 19 e 26.

YUCEEL, H. P. **CVE-2023-4966: LockBit Exploits Citrix Bleed in Ransomware Attacks**. 2023. Acesso em: 22 abr. 2025. Disponível em: <https://www.picussecurity.com/resource/blog/cve-2023-4966-lockbit-exploits-citrix-bleed-in-ransomware-attacks#references>. Citado na página 18.

ZHOU, F. et al. Dump and analysis of android volatile memory on wechat. In: IEEE. **2015 IEEE International Conference on Communications (ICC)**. London, UK, 2015. p. 7151–7156. Citado na página 19.

APÊNDICE **A**

Código-Fonte dos Experimentos

O código-fonte completo desenvolvido neste trabalho encontra-se disponível em repositório público, podendo ser acessado por meio do seguinte endereço eletrônico:

<https://github.com/wpfsilvaa/Ransomware-Detection-through-Memory-Forensics-Analysis>

O repositório contém todos os artefatos necessários para a reprodução dos experimentos, incluindo scripts, dependências e instruções de execução.