Centro Universitário de Brasília – UniCEUB
Faculdade de Tecnologia e Ciências Sociais Aplicadas (FATECS)

AUGUSTO OLIVEIRA SILVA

ENGENHARIA REVERSA DE MALWARE USANDO MACHINE LEARNING: IMPACTOS E APLICABILIDADE NA ÁREA DA CIBERSEGURANÇA

Brasília
2025

AUGUSTO OLIVEIRA SILVA

ENGENHARIA REVERSA DE MALWARE USANDO MACHINE LEARNING: IMPACTOS E APLICABILIDADE NA ÁREA DA CIBERSEGURANÇA

Projeto de Pesquisa apresentado como requisito parcial para a obtenção do título de bacharel em Ciência da Computação pela Faculdade de Tecnologia e Ciências Sociais Aplicadas – FATECS do Centro Universitário de Brasília (CEUB). Orientador: Professor Valdemir dos Santos Silva.

BRASÍLIA

2025

AUGUSTO OLIVEIRA SILVA

ENGENHARIA REVERSA DE MALWARE USANDO MACHINE LEARNING: IMPACTOS E APLICABILIDADE NA ÁREA DA CIBERSEGURANÇA

Projeto de Pesquisa apresentado como requisito parcial para a obtenção do título de bacharel em Ciência da Computação pela Faculdade de Tecnologia e Ciências Sociais Aplicadas – FATECS do Centro Universitário de Brasília (CEUB). Orientador: Professor Valdemir dos Santos Silva.

Brasília, ____ de _____ 2025.

BANCA EXAMINADORA

_____

Professor(a): Valdemir dos Santos Silva

Orientador(a)

DEDICATÓRIA

AGRADECIMENTOS

RESUMO

O presente trabalho tem como objetivo a aplicação de modelos de Machine Learning juntamente com a Engenharia Reversa para a identificação de padrões de tráfego de redes, buscando detectar malwares e ataques cibernéticos. Para isso, foram utilizados os conjuntos de dados CIC-IDS2017 e CSE-CIC-IDS2018. Conjuntos que são amplamente utilizados e reconhecidos como ótimas fontes para estudos na área da cibersegurança. O pré-processamento incluiu o tratamento de valores nulos, colunas, duplicados, tratamento de colunas e normalização de variáveis. Para a modelagem preditiva, foram testados diferentes algoritmos, com ênfase em modelos interpretáveis como Random Forest e XGBoost. A abordagem proposta pode ser aplicada em sistemas de defesa cibernética para aprimorar a segurança de redes contra ameaças emergentes.

Palavras-chaves: Engenharia Reversa. Malware. Machine Learning. Dados. Cibersegurança. Sistemas. Softwares.

LISTA DE ILUSTRAÇÕES

QUADROS

FIGURAS

GRÁFICOS

SUMÁRIO

1. INTRODUÇÃO

A crescente sofisticação das ameaças cibernéticas tem alavancado a necessidade de técnicas avançadas para a detecção de ataques em redes de comunicação. Com o crescimento exponencial do volume de dados e maior complexidade dos padrões de tráfego, os sistemas tradicionais de segurança têm enfrentado desafios nunca vistos. Estes avanços tecnológicos ampliam significativamente a superfície de ataque, tornando redes e dispositivos IoT mais suscetíveis a sofrerem com atividades maliciosas, como ataques distribuídos de negação de serviço (DDoS), injeções de código malicioso e exploração de vulnerabilidades.

Neste caso, o uso da Inteligência Artificial (IA) e Machine Learning (ML) pode ser uma boa solução para possíveis riscos e ameaças virtuais. Os modelos de aprendizagem maquina têm a habilidade de analisar grandes quantidades de tráfego na rede, achar padrões e estruturas que podem indicar ataques. Mas, devido à complexidade dos modelos de ML, um dos grandes problemas é a interpretabilidade, ou seja, a análise e o entendimento das decisões tomadas pelo próprio algoritmo. Isso se deve ao fato de que não é fácil entender como e porque esses algoritmos tomam suas decisões. Essa falta de clareza pode trazer efeitos negativos no processo de implantação dessas técnicas, principalmente em locais corporativos, onde compreender o que levou o modelo a tomar uma decisão é a chave confiar ou não nos sistemas.

À vista disso, o presente projeto visa o desenvolvimento de modelos de aprendizado de máquina de fácil entendimento para análise de tráfego de rede e identificação de malwares. Para que isso fosse realizado, foram utilizados os conjuntos de dados CIC-IDS2017 e CSE-CID-IDS2018, na qual ambos contêm registros de tráfego de rede. O principal objetivo é aplicar técnicas de ML para desenvolver um modelo capaz não somente de detectar ataques, mas também de apresentar alta performance em contextos reais de tráfego de rede.

Ao final desta pesquisa, espera-se que os resultados obtidos colaborem para o avanço de soluções na área da cibersegurança, permitindo entender, de forma mais clara, qual o impacto causado por um arquivo malicioso. Por fim, esta pesquisa possibilitará o desenvolvimento de soluções mais confiáveis e seguras, promovendo o aumento da utilização da inteligência artificial na detecção de malwares e melhorando a proteção e segurança contra ataques em redes modernas.

1.1. Tema

Engenharia Reversa de Malware Usando Machine Learning: Impactos e Aplicabilidade na Área da Cibersegurança

1.2. Pergunta de Pesquisa

Como detectar e se prevenir de malwares e qual o seu impacto em sistemas de cibersegurança?

1.3. Justificativa

Esta pesquisa tem o objetivo de cumprir os requisitos necessários para a conclusão do curso de Ciência da Computação pela Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS do Centro Universitário de Brasília (CEUB).

1.4. Objetivos

1.4.1. Objetivo Geral

Desenvolver um sistema de machine learning que, através da engenharia reversa, seja capaz de analisar, identificar e prever se um arquivo é ou não malicioso.

1.4.2. Objetivo Específicos

- Coletar e analisar características de malwares;
- Desenvolver um algoritmo de machine learning;
- Identificar e prever arquivos maliciosos;

1.5. Metodologia

Nesta seção, foi descrito o processo adotado para desenvolver e avaliar modelos de aprendizado de máquina voltados para a detecção de ataques cibernéticos. O fluxo metodológico seguiu as etapas de coleta e preparação dos dados, análise exploratória, engenharia de atributos, treinamento dos modelos, teste e avaliação de desempenho.

1.5.1. Coleta e Seleção dos Dados

Para o desenvolvimento do projeto, foram utilizados os conjuntos de dados CIC-IDS2017 e CSE-CIC-IDS2018, dois conjuntos de dados amplamente utilizados para estudos de segurança cibernética e desenvolvimento de modelos preditivos. Esses conjuntos de dados contém características gerais sobre o sobre tráfego de rede sendo, em sua maioria, classificados como tráfego benigno.

Os dados foram adquiridos no formato CSV e passaram por uma análise exploratória com o intuito de compreender a estrutura, distribuição das classes e identificar possíveis problemas como valores ausentes e redundâncias.

1.5.2. Pré-processamento dos Dados

Após coletar os dados, foi feito o pré-processamento com o intuito de garantir a qualidade e estrutura dos dados de modo a facilitar a análise, posteriormente, a modelagem. Para isso, as seguintes etapas foram aplicadas:

1. Unificação dos dados: Tanto os dados do ano de 2017, quanto de 2018, possuíam, cada um, oito e nove arquivos no formato CSV, por isso, esses arquivos foram unificados formando um arquivo csv para o ano de 2017 e três arquivos para o ano de 2018;
2. Tratamento de colunas: Etapa responsável por organizar os nomes das colunas e, quando necessário, tratar o tipo de dado que cada coluna tem. No dados referentes a 2018, esses tratamento envolveu duas fases, uma para converter as colunas em numéricas e outra para converter as colunas do tipo float64 para int64;
3. Tratamento de valores duplicados: Foram identificados e removidos todos os valores duplicados para que estes não prejudicasse ou, futuramente, enviasse os modelos de ML;
4. Tratamento de valores ausentes e infinitos: Foram identificados e substituídos pela mediana da colunas os valores faltantes ou infinitos;
5. Tratamento de colunas categóricas: Etapa onde foram feitas renomeações dos rótulos presentes na coluna ‘Label’ de ambos os conjuntos de dados, visando facilitar a leitura e compreensão dos dados;

6. Análise inicial: Foi realizada uma análise inicial da proporção de tráfegos benignos e maliciosos.

1.5.3. Análise Exploratória dos Dados

Nesta etapa foi feita toda a análise de caráter exploratória e descritiva sobre os dados em questão. Para isso foram desenvolvidos gráficos, como gráficos de barra de ataques mais frequentes, gráficos de calor que mostram a correlação entre as variáveis numéricas dos conjuntos de dados, gráficos que mostram as portas de destino mais utilizadas e gráficos do tipo violino que mostram a duração de fluxo de acordo com o tipo de tráfego, etc.

1.5.4. Feature Engineering

Para melhorar o desempenho dos modelos, foram aplicadas técnicas de seletesteção e identificação de atributos relevantes ou não para a modelagem do ML, isso incluiu a remoção de colunas irrelevantes, onde foram identificadas e removidas as colunas que possuíam valores únicos em todos os campos;

1.5.5. Modelagem e Treinamento

Etapa onde foram desenvolvidos, treinados e avaliados diversos modelos de ML. Esse processo, após a separação do conjunto de dados em subconjuntos de treinamento e teste que torna possível garantir que os modelos sejam avaliados em dados que não foram vistos durante o treinamento, simulando assim, um cenário real de generalização, seguiu para a etapa de balanceamento dos dados, tendo em vista que, originalmente, a relação entre tráfego benigno e malicioso era muito desproporcional. Passados estes processos, a modelagem seguiu duas abordagens principais, são elas:

- Classificação Binária: Criação de modelos cujo resultado é uma saída booleana.
    - Regressão Logística: Algoritmo supervisionado utilizado para classificação e construído a partir da transformação de uma função sigmóide sobre a regressão linear;
    - SVM (Support Vector Machine): Algoritmos supervisionados que classificam os dados através de uma linha ou hiperplano de N-dimensões.

- Classificação Multiclasse: Criação de modelos capazes de categorizar dados em duas classes ou mais.
    - Random Forest: Algoritmo composto por diversas árvores de decisão, formando assim uma floresta cujo aprendizado ocorre pelo método de conjunto;
    - K-Nearest Neighbors (KNN): Utiliza a proximidade para realizar previsões sobre um conjunto de pontos de dados individuais;
    - XGBoost: Algoritmo de boosting que utiliza árvores de decisão com esforço gradativo para aprendizado supervisionado, principalmente em problemas com classes desbalanceadas.

1.5.6. Testes e Avaliação dos Modelos

Para medir o desempenho dos modelos criados, foram utilizadas diferentes métricas de avaliação, como acurácia, precisão, recall (sensibilidade), F1-Score, ROC-AUC, Matriz de confusão, etc. Além das métricas, foi realizada uma comparação entre os modelos para entender qual deles apresentou melhor desempenho em cada abordagem.

1.5.7. Implementação e Considerações Finais

Os modelos treinados foram salvos para implementações de interface para interação de usuários e testes adicionais. Além disso, os resultados obtidos foram analisados para identificar:

- O melhor modelo para cada abordagem;
- Possíveis melhorias, incluindo ajuste de hiperparâmetros e experimentação com redes neurais.

2. ENGENHARIA REVERSA

Com o passar do tempo, o processo de criação de um sistema que, através de instruções e algoritmos, fosse capaz de realizar uma tarefa específica em um computador, se tornou cada vez mais sofisticado. Cada processo de criação de um software, por mais complexo que seja, envolve diversas etapas que vão desde a idealização até a entrega do produto final.

Portanto, é comum que hoje existam softwares muito antigos, comumente chamados pelos profissionais da área de tecnologia da informação de “sistemas legados”, ou seja,

sistemas que estão em processo de obsolescência mas que, por muitos anos, estão em uso dentro de uma empresa. Com isso, é de se esperar que, quanto mais antigo for um software, maior é a necessidade e, também, a dificuldade de realizar manutenções e atualizações no mesmo.

A engenharia reversa dentro da área da tecnologia vem como um conjunto de atividades que permitem, a partir de um produto já existente, entender todos os conceitos e ideias que ali foram utilizadas e que busca entender todas as etapas de construção, a partir do mais baixo nível, até chegar ao produto final que está em uso dentro de uma empresa .

Visando alcançar objetivos com a utilização da engenharia reversa, como otimização do desempenho, melhorias na segurança e personalização de um produto, é necessário realizar um processo de desconstrução, ou seja, revisar todas as etapas realizadas a fim de se obter um sistema que entregue o mesmo produto final, porém com mais segurança, confiabilidade e eficiência

2.1. Funcionamento

Conforme dito anteriormente, a engenharia reversa realiza o processo de desconstrução. Em objetos físicos, isso significa desmontar um equipamento, dispositivo eletrônico ou uma ferramenta, para analisar e compreender exatamente o funcionamento de cada peça do objeto e, a partir do entendimento de cada componente, construí-lo novamente com melhorias aplicadas.

Por exemplo, em uma empresa automobilística, foi detectado um problema crônico em um modelo de carro e é necessária a reformulação do processo de fabricação. O carro foi desenvolvido há 15 anos e os trabalhadores que o criaram não trabalham mais na empresa. A empresa tem criar um novo modelo com as mesmas características, ou seja, deve ter as mesmas dimensões, carroceria e desempenho. Para que isso seja feito, é necessário pegar o atual modelo com o problema crônico e recriá-lo do zero. Dessa forma, a empresa consegue detectar a fonte e o motivo do problema para desenvolver o novo carro.

Já na área da tecnologia, quando se usa o processo de desconstrução de um software, se faz uma análise e buscas por documentações, relatórios e códigos que foram feitos durante o desenvolvimento primário de uma determinada aplicação.

Por exemplo, um sistema legado de um banco apresenta lentidão e queda no desempenho durante seu funcionamento. A aplicação foi desenvolvida a 20 anos, logo, a linguagem utilizada para o desenvolvimento foram linguagens de programação que já estão

em desuso e a qual existem poucas ferramentas para a manutenção e atualização do mesmo. Para não perder os clientes, o banco deve atualizar o software e melhorar suas funcionalidades Para isso pode ser necessária a recriação do mesmo, utilizando os mesmos princípios de 20 anos atrás. Ademais, devido a idade da aplicação, ela não funciona em computadores atuais e não atende às regras de negócios atuais do banco. Fazendo todo esse processo, o banco consegue melhorar a aplicação e manter os clientes que a utilizam.

2.2. Surgimento e Aplicações

Embora não se tenha uma data exata que marque o surgimento da engenharia reversa, seu uso pôde ser notado durante a Segunda Guerra Mundial (1939-1945), onde algumas potências da época, como Alemanha, Estados Unidos e Japão, passaram a estudar as tecnologias empregadas em armamentos e equipamentos militares de seus inimigos. Um exemplo a ser citado é o caso da máquina Enigma, um dispositivo alemão de comunicação criptografada. Visando entender como os alemães se comunicavam sem serem interceptados, os britânicos, liderados por Alan Turing, utilizaram técnicas de engenharia reversa para entender o funcionamento da máquina Enigma e, a partir desse estudo, foi desenvolvida a máquina Bombe, que realizava a decodificação das mensagens enviadas de forma automática. Esse avanço tecnológico possibilitou o acesso à informações estratégicas e, consequentemente, a antecipação de ataques e operações alemãs.

Graças aos avanços tecnológicos, hoje a engenharia reversa pode ser aplicada em diversas áreas e suas aplicações variam de acordo com a necessidade de cada área. Algumas aplicações estão no controle de qualidade, possibilitando entender e identificar erros e falhas da produção, nos softwares, avaliando as funções e interface aplicadas para entender quais caminhos e decisões foram tomadas até chegar ao produto que está em produção. Além disso, ela é amplamente aplicada à área da segurança da informação na análise malware, coletando informações sobre malwares e desenvolvendo sistemas cada vez mais seguros e, também, nos testes de vulnerabilidades, facilitando as empresas a encontrarem falhas em sistemas e, dessa forma, protegê-los contra os ataques hackers.

3. MALWARE

3.1. Surgimento

O termo malware é utilizado para descrever qualquer software malicioso, código ou programa de computador desenvolvido intencionalmente para prejudicar ou interromper o funcionamento normal de um dispositivo. Esses programas maliciosos podem variar desde ataques altamente prejudiciais e caros para a vítima até um simples arquivo irritante que não danifica o dispositivo mas sim, dificulta o usuário de acessar pastas ou arquivos desejados.

Apesar de não se ter uma data exata de surgimento do primeiro malware criado, no ano de 1966, foi desenvolvido por John von Neumann, o conceito de um programa que poderia se reproduzir e se espalhar dentro de um sistema. Isso fez com que, apenas 5 anos após a publicação do seu trabalho, “A teoria dos autômatos auto reprodutíveis”, em 1971, o programador Bob Thomas criasse um programa experimental, conhecido como Creeper, que posteriormente seria modificado por seu colega, Ray Tomlinson, para que, além de se mover entre os computadores, o programa também se copiasse de uma máquina para a outra.

Outro marco importante na história dos malwares, foi no ano de 1988, quando Robert Morris, estudante do MIT, sem a intenção maliciosa, criou um software que, futuramente, seria considerado um dos precursores de malware. O Worm Morris, assim chamado, foi um software que não apenas se copiava de um computador para o outro mas, em máquinas já infectadas, ele se auto copiava repetidamente, sem um limite de parada.

Portanto, devido a sua eficiência, além da esperada, o software infectou e levou à falha aproximadamente 10% dos 60.000 computadores que possuíam o acesso à internet, fazendo com que esses computadores parassem completamente. Em razão disso, seu criador, Robert Morris, foi considerado o primeiro cibercriminoso e foi condenado por fraude cibernética nos Estados Unidos.

Mais para frente, em 2000, surgiram diversos tipos de malwares porém, um destaque foi o ransomware chamado de CryptoLocker. Um software com alta capacidade de disseminação e recursos de criptografia que era capaz de atacar recursos compartilhados em uma rede local, como escritórios ou bibliotecas. Seu principal objetivo era a mineração de bitcoins e as vítimas, para recuperarem o acesso aos recursos criptografados por ele, deveriam pagar o valor de dois bitcoins, avaliados na época em, aproximadamente, 715 USD.

3.2. Tipos de Malware

- Vírus: Trecho de código utilizado para o “sequestro” de um software, o que significa que os vírus não podem agir sozinhos. Para isso é necessário um programa executável e seu objetivo pode ir desde a interrupção de operações até a exclusão de arquivos importantes da máquina na qual o vírus se instala;
- Botnets: Rede de dispositivos infectados e que estão conectados à internet e são controlados por um hacker. Normalmente, as vítimas não percebem que fazem parte de um botnet e seu objetivo costuma ser o lançamento de ataques DDoS (Distributed Denial of Service — Ataque de Negação de Serviço Distribuído) e o sobrecarregamento sistemas;
- Spyware: Softwares que se escondem dentro de uma máquina com o objetivo de monitorar as atividades do usuário e coletar informações confidenciais e transmiti-las de volta ao invasor. Um tipo comum de spyware, é o Keylogger, que grava todas as teclas digitadas pela vítima, coletando senhas, nomes, contas bancárias e etc;
- Ransomware: Um tipo crítico de malware que pode destruir ou bloquear o acessos a informações sensíveis do usuário exigindo um pagamento pelo resgate dessas informações. Em um ataque chamado de extorsão dupla, os cibercriminosos roubam os dados e ameaçam o vazamento caso o resgate não seja pago, já no ataques de extorsão tripla, os cibercriminosos criptografam, roubam e ameaçam desconectar a vítima dos sistemas por meio de ataques DDoS;
- Phishing: Normalmente feito através de emails, sites e mensagens de texto, o ataques de phishing rouba as informações confidenciais da vítima, como senhas, cartões bancários e nomes de usuários. Por exemplo, o cibercriminoso pode se passar por um banco alertando sobre uma compra indevida ou sobre o congelamento de sua conta bancária onde, para resolver o problema, é necessário abrir um link enviado pelo email.

Além desses, existem outros tipos de malwares, por exemplo, Adwares, Rootkits, Trojans, Worms, Malwares sem arquivo, Mineradores de criptomoeda, Cavalos de Tróia, etc.

3.3. Detecção e Prevenção

Em sua maioria, os malwares buscam ficar escondidos das vistas do respectivo usuário de uma máquina. Mas, ainda assim, existem algumas formas de detecção, são elas:

- Queda no desempenho: Quando um malware é instalado na máquina, os recursos computacionais são utilizados para executar e atingir o objetivo do software malicioso, muitas das vezes ocupando espaço de armazenamento, consumindo memória RAM e interrompendo processos legítimos. Com isso, a equipe de TI pode notar uma lentidão do fluxo de usuários, travamentos das máquinas ou, até mesmo, o excesso de pop-ups nas telas;
- Alteração das configurações: Alguns malwares podem alterar as configurações estabelecidas pelas pela organização ou pelo próprio usuário, como regras de firewall ou aumento do acesso do usuário a algumas funcionalidades da máquina;
- Alerta de eventos de segurança: Com alguns malwares, este pode ser o primeiro alerta para organizações que possuem sistemas de detecção como IDS ou SIEM. Com esses sistemas, os profissionais da área de segurança podem gerenciar e ter um panorama mais abrangente de todas as máquinas e domínios da rede.

Atualmente, existem diversas formas para se proteger de um malware. Quando se trata de uma máquina pessoal, algumas formas de prevenção incluem a atualização do software, utilização de um programa antivírus e backups offline dos arquivos da máquina. Quando a prevenção está relacionada a empresas, as mesmas podem realizar treinamentos constantes sobre cibersegurança, estabelecer políticas de seguranças, como autenticação multifatorial, uso de VPNs, backups locais e na nuvem e criação de uma plano de resposta a incidentes. Todas essas formas facilitam e ajudam o usuário final a se manter protegido e longe de ameaças à suas máquinas e sistemas.

4. COLETA DE DADOS

A obtenção de dados é um passo fundamental para uma modelagem assertiva e, para isso, foram utilizados dois conjuntos de dados principais, referentes ao ano de 2017 e 2018. Ambos os conjuntos possuem registros de tráfegos de redes classificados tanto como benignos quanto maliciosos.

4.1. Origem dos dados

Os conjuntos de dados utilizados para a presente pesquisa foram coletados a partir dos conjuntos CIC-IDS2017 e CSE-CIC-IDS2018. Sendo desenvolvidos em um ambiente seguro

controlado pelo CIC — Canadian Institute for Cybersecurity — que buscam gerar dados que estejam próximos dos dados reais.

4.1.1. CIC-IDS2017

Este conjunto de dados contém os ataques mais recorrentes do ano de 2017 e que se assemelham aos dados do mundo real. Com isso, o mesmo inclui análises de tráfego de rede que registram endereços e portas de origem e destino. Para realizar a coleta desses dados foi simulado o comportamento abstrato de 25 usuários e reservado um período de, aproximadamente, cinco dias. E ainda, foram levados em consideração alguns critérios, como a configuração de rede completa, diversidade de ataques e interação completa.

4.1.2. CSE-CIC-IDS2018

Os dados existentes neste conjunto foram gerados a partir de dois cenários principais. O primeiro cenário, chamado de “B-Profile”, foi utilizado para a produção e coleta de tráfegos benignos. Tendo sido projetado para extrair o comportamento de usuários, o sistema tenta encapsular eventos de rede, como número de pacotes enviados e tamanho de pacotes por fluxo através de técnicas de aprendizado de máquina. Logo, tendo em vista que os “B-Profiles” são derivados de usuários, eles podem ser utilizados para gerar eventos benignos na rede.

O último cenário, nomeado como “M-Profile”, foi utilizado para gerar sete tipos de ataques diferentes, onde para cada ataque foi criado um ambiente específico e ambos foram executados em uma ou mais máquinas. E para simular um ambiente corporativo real, foram implementadas cinco sub-redes, cada uma caracterizando um setor em específico.

Sendo assim, foi criado este conjunto de dados de forma ordenada e controlada, abarcando os mais diversos cenários de ataques possíveis em três momentos de coleta diferentes, cada um com três dias de duração, aproximadamente.

4.2. Processamento inicial

Ambos os conjuntos de dados, 2017 e 2018, incluíam vários arquivos no formato “.csv”, por isso, foi feita a unificação desses arquivos gerando um número menor de arquivos a serem analisados, ou seja, concatenar todos os arquivos de coletas de tráfego em um só arquivo e, consequentemente, melhorar a eficácia da análise exploratória dos dados.

Para diminuir o tempo de processamento dos dados, e também, tendo em vista o tamanho dos arquivos ".csv", o conjunto de dados do ano de 2018 foi subdividido em 3 partes menores onde cada uma possui, aproximadamente, um gigabyte. A tabela a seguir demonstra o resultado dos arquivos unificados de cada um dos conjuntos de dados:

Quadro 1 - Unificando Datasets

	Nº de Arquivos	**Arquivos Gerados**	**Tamanho**
CIC-IDS2017	8	1	943 Mb
CSE-CIC-IDS2018	9	Parte 1	1,1 Gb
		Parte 2	1,15 Gb
		Parte 3	708 Mb

4.3. Pré-processamento dos dados

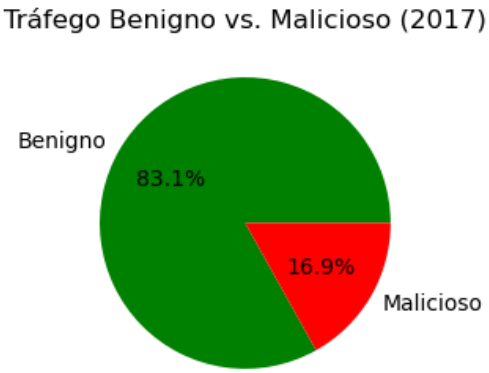
Esta etapa busca realizar a limpeza de todo o conjunto de dados para que, no momento de analisar os dados, um erro não passe sem ser notado, caso contrário, o mesmo pode vir a causar erros significativos tanto na análise quanto no desenvolvimento de um modelo preditivo. Os tratamentos realizados para ambos os conjuntos de dados incluem:

- Tratamento de valores nulos e ausentes;
- Tratamento de valores duplicados;
- Tratamento de colunas categóricas;
- Renomeação de colunas;
- Inspeção do Data Frame gerado.

4.4. Análise inicial

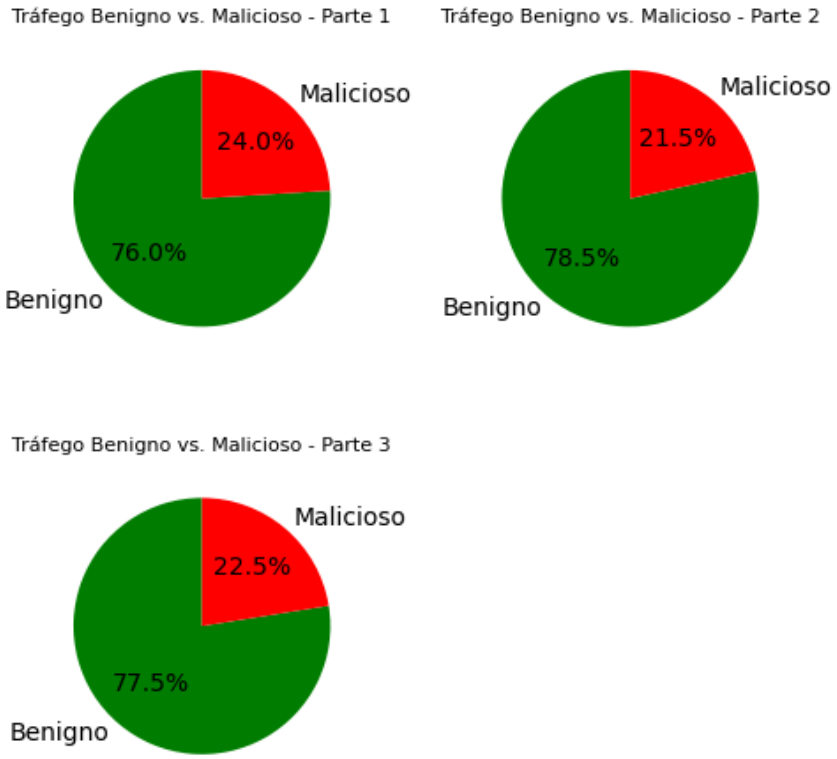
Por fim, tendo realizado todo o processamento e tratamento primário dos dados, foi realizada a análise inicial da proporcionalidade com relação aos tráfegos benignos e maliciosos presentes em cada um dos datasets conforme visto nos gráficos a seguir:

Gráfico 1 - Proporção Tráfego Benigno vs Malicioso 2017



Fonte: Elaborado pelo autor

Gráfico 2 - Proporção Tráfego Benigno vs Malicioso 2018



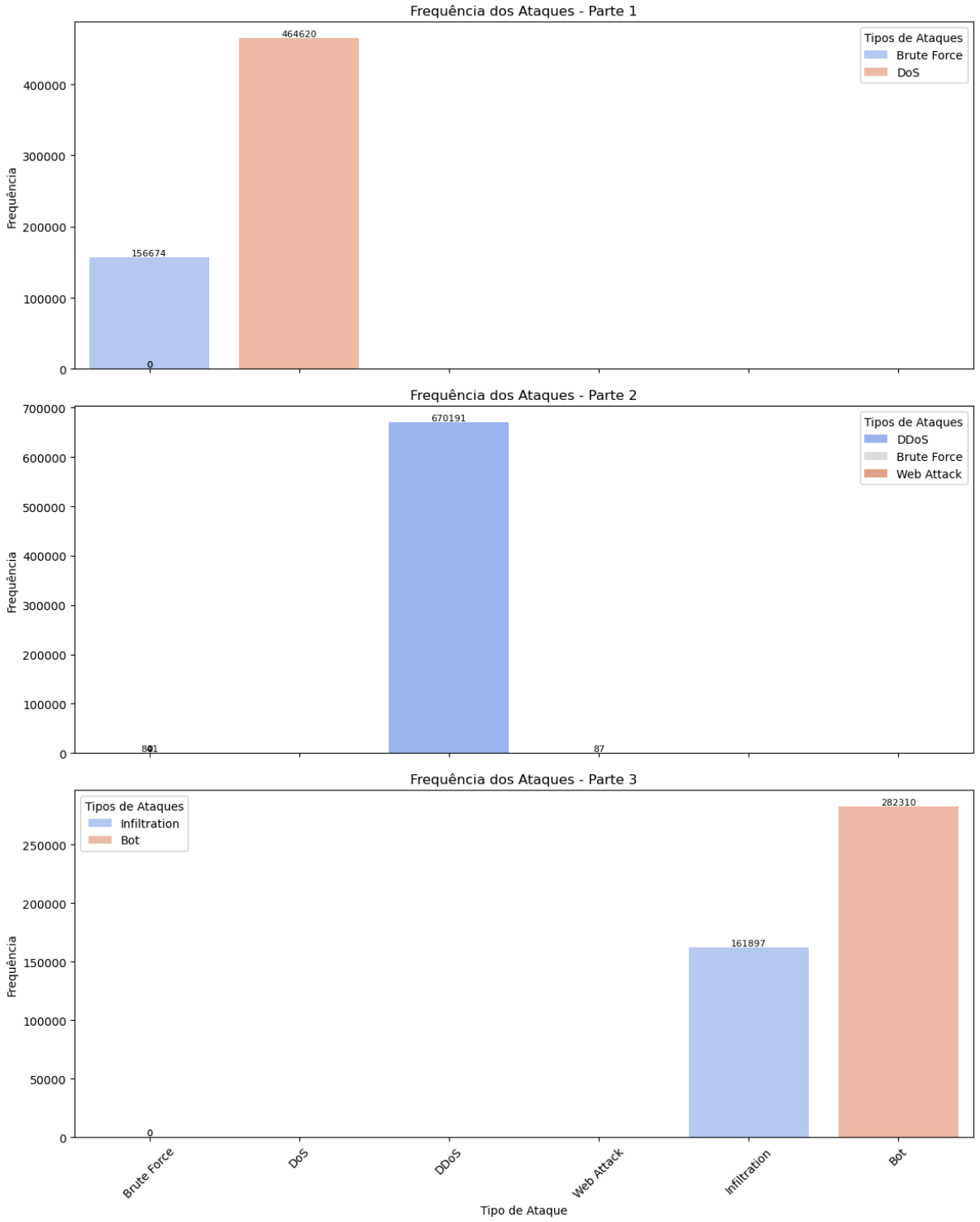
Fonte: Elaborado pelo autor

5. ANÁLISE EXPLORATÓRIA DOS DADOS

Após a etapa de coleta e tratamento inicial dos dados, outra etapa fundamental é a Análise Exploratória dos Dados. Ela consiste no processo de extração e obtenção de insights e informações valiosas sobre o conjunto de dados no qual o cientista ou analista de dados está trabalhando. Ela é responsável por dar, ao desenvolvedor, uma visão panorâmica e, dessa forma, entender padrões, estruturas, distribuição de variáveis, valores incomuns, etc. Seu principal foco é garantir que os dados estejam prontos para serem usados de maneira eficaz em uma abordagem ou desenvolvimento de um modelo preditivo. Por conseguinte, para esta pesquisa, o processo de análise dos dados foi dividido em duas partes principais, uma para o conjunto de dados CIC-IDS2017 e outra para o conjunto de dados CSE-CIC-IDS2018.

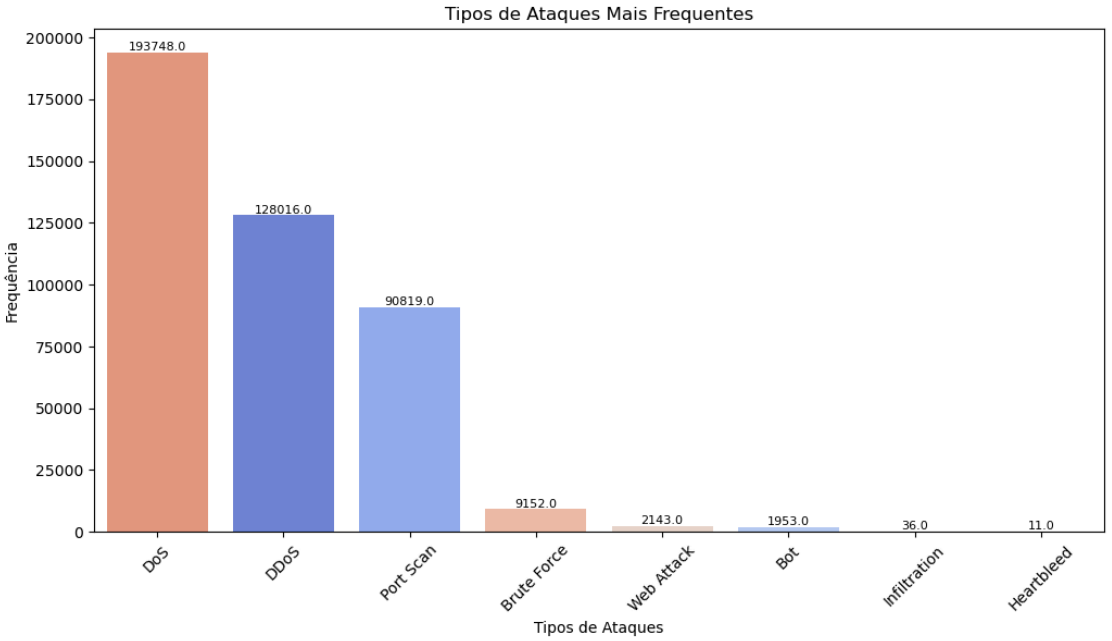
Ao analisar os dados, é perceptível um tráfego de rede complexo e volumoso e, além disso, nota-se também, que existe um desequilíbrio significativo da proporção entre tráfego benigno e malicioso, conforme visto nos gráficos presentes na etapa de coleta de dados. Portanto, buscando analisar e filtrar os tipos de ataques mais frequentes, observa-se que nos dados de 2017, ataques como DoS, DDoS e PortScan foram os mais frequentes durante todo o período de coleta. Já nos dados de 2018, os ataques de maior frequência foram DDoS, Bot e ataques DoS. Isso sugere pontos específicos que foram explorados por invasores. Podemos ver essa relação nos gráficos abaixo:

Gráfico 3 - Frequência de Ataques 2018



Fonte: Elaborado pelo autor

Gráfico 4 - Frequência de Ataques 2017

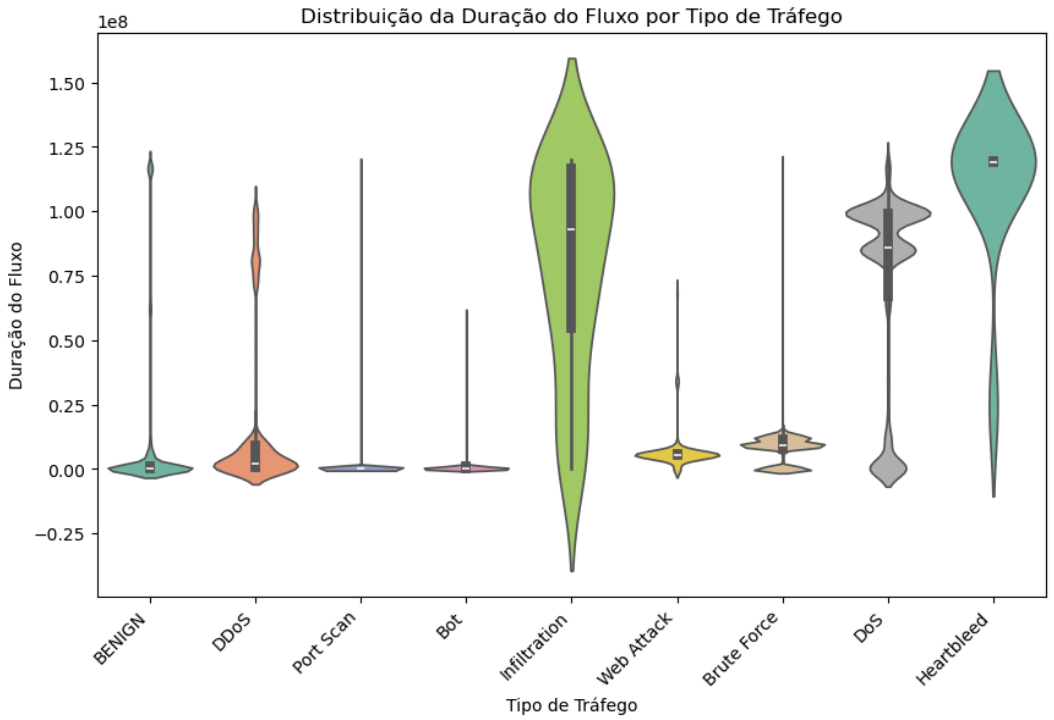


Fonte: Elaborado pelo autor

Observando os tipos de ataques existentes, foi feita uma análise que objetivava entender a relação entre a duração do fluxo de tráfego e os tipos de tráfego. Por isso, devido a quantidade e dispersão dos dados do ano de 2018, foi utilizada a escala logarítmica[1], uma vez que ela consegue manter a precisão dos dados. Através dos gráficos abaixo, é possível perceber que, os ataques com a menor duração indicam tentativas rápidas de invasão, já os ataques com maior tempo de duração, indicam tentativas prolongadas e um maior tempo de exploração de alguma vulnerabilidade.

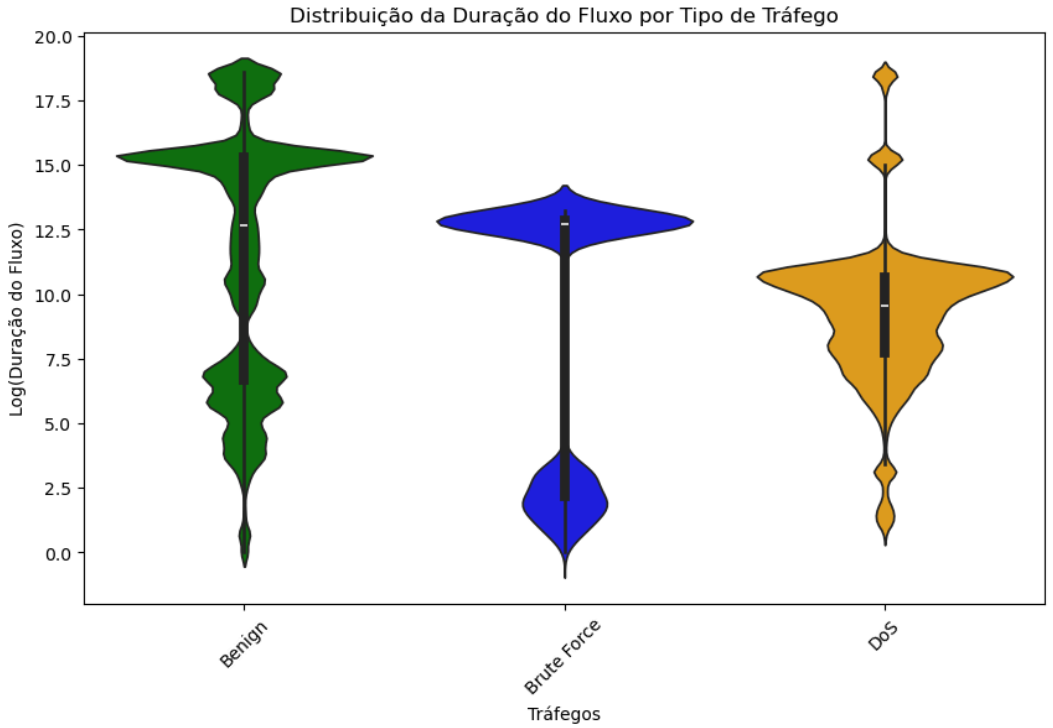
[1] Escala Logarítmica: Forma de representação dos dados cuja ordem de grandeza são muito diferentes. Ela permite que os dados com uma ampla amplitude sejam visualizados de forma mais clara.

Gráfico 5 - Duração do Fluxo por Tipo de Tráfego 2017



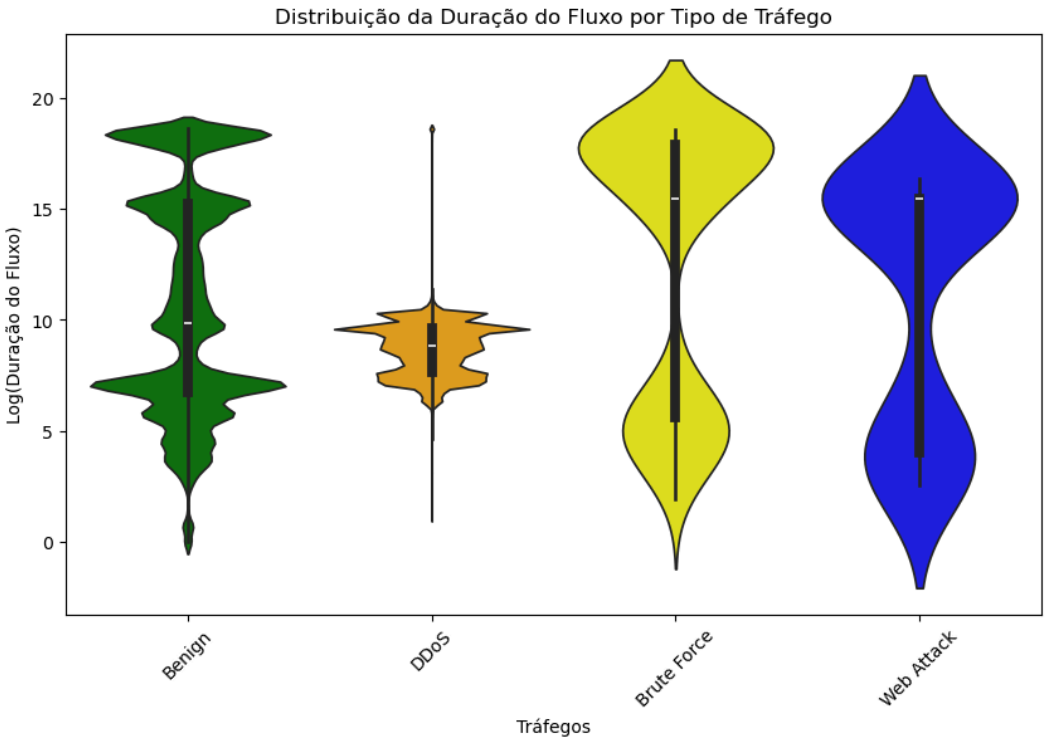
Fonte: Elaborado pelo autor

Gráfico 6 - Duração do Fluxo por Tipo de Tráfego 2018 - Parte 1



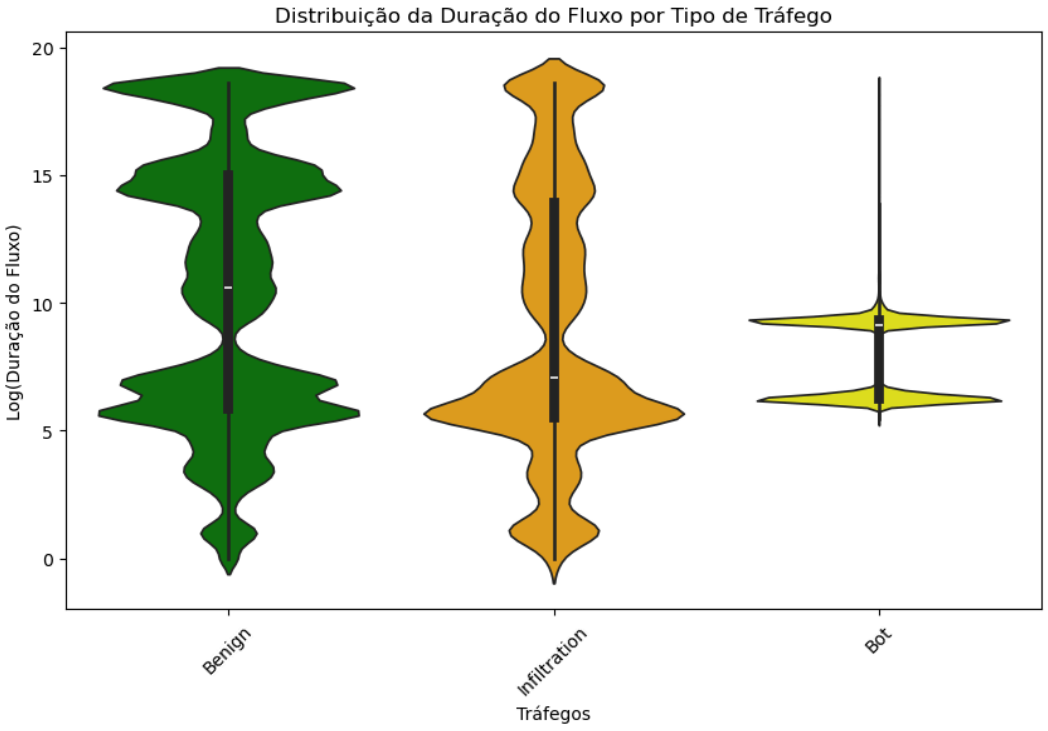
Fonte: Elaborado pelo autor

Gráfico 7 - Duração do Fluxo por Tipo de Tráfego 2018 - Parte 2



Fonte: Elaborado pelo autor

Gráfico 8 - Duração do Fluxo por Tipo de Tráfego 2018 - Parte 3



Fonte: Elaborado pelo autor

Uma vez observada a relação entre frequência de ataques e duração do fluxo, é necessário verificar se há algum ponto de maior fragilidade. Para isso, foram identificadas as portas de destino que mais sofreram ataques. Portas de destino indicam serviços e protocolos, ou seja, cada número de uma porta de destino está associado a um serviço. Na tabela a seguir é possível o serviço que as portas principais indicam:

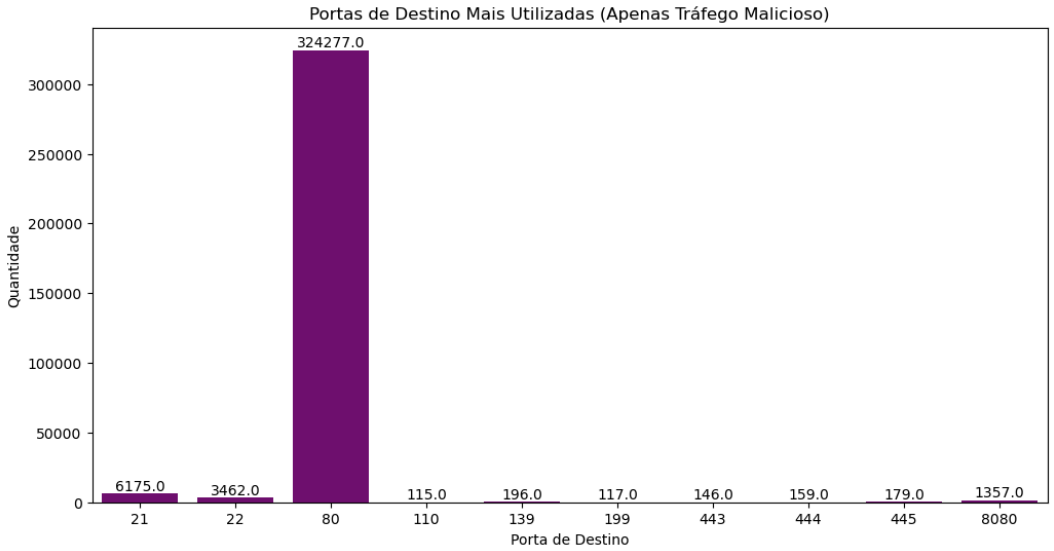
Quadro 2 - Portas de Destino

Portas de Destino	Serviço	Ataques mais Comuns
21	FTP (File Transfer Protocol): Utilizado para transferência de arquivos.	Força bruta e exploração de falhas de segurança
22	SSH (Secure Shell): Utilizado para comunicação em servidores	Força bruta e exploração de vulnerabilidades
53	DNS (Domain Name System): Nomes de domínio em endereços IP	DDoS e DNS Amplification
80	HTTP (Hypertext Transfer Protocol): Comunicação cliente-servidor	SQL Injection e DoS (Denial of Service)
443	HTTPS (Hypertext Transfer Protocol Secure): Comunicação cliente-servidor	MitM (Man-in-the-Middle), SSL Stripping e exploração de falhas
445	Microsoft-DS: Utilizada para o compartilhamento de arquivos no Windows	Exploração do SMB, por exemplo, o ataque EternalBlue
3389	RDP (Remote Desktop Protocol): Utilizado para acesso remoto à máquinas	Força bruta para adivinhação de senhas
8080	Servidores Web	SQL Injection, Cross-Site Scripting

Ao explorar as portas de destino mais utilizadas e conforme mostram os gráficos abaixo, é notório que, em ambos os anos, houveram pontos de vulnerabilidade mais comuns que os demais, nesse caso, as portas 21, 22 e 80. Já no ano de 2018, houve um aumento na

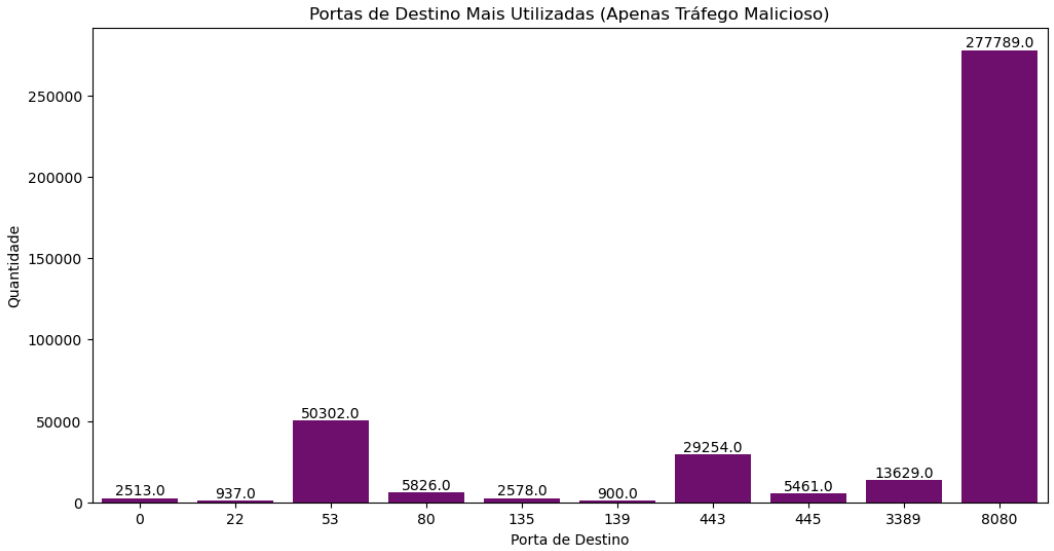
frequência de ataques às portas de número 53, 443 e 8080. Isso reforça a necessidade de fortalecer as estratégias de segurança no que diz respeito ao tráfego na rede.

Gráfico 9 - Portas de Destino mais Utilizadas 2017



Fonte: Elaborado pelo autor

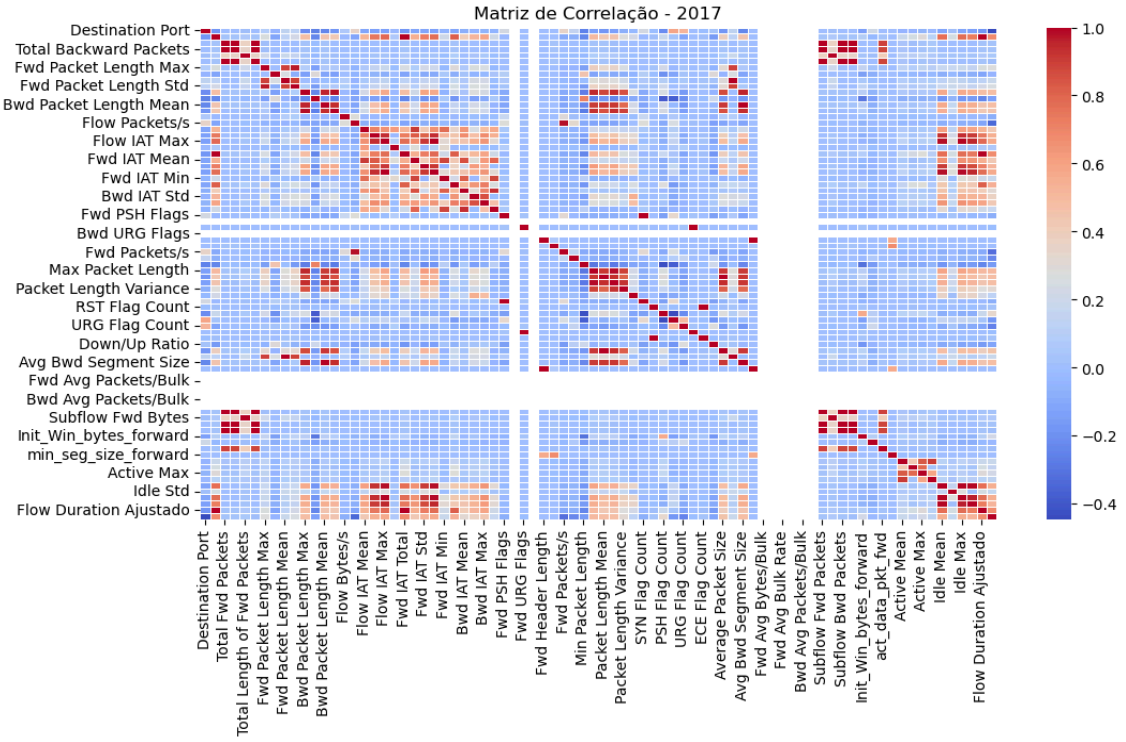
Gráfico 10 - Portas de Destino mais Utilizadas 2018



Fonte: Elaborado pelo autor

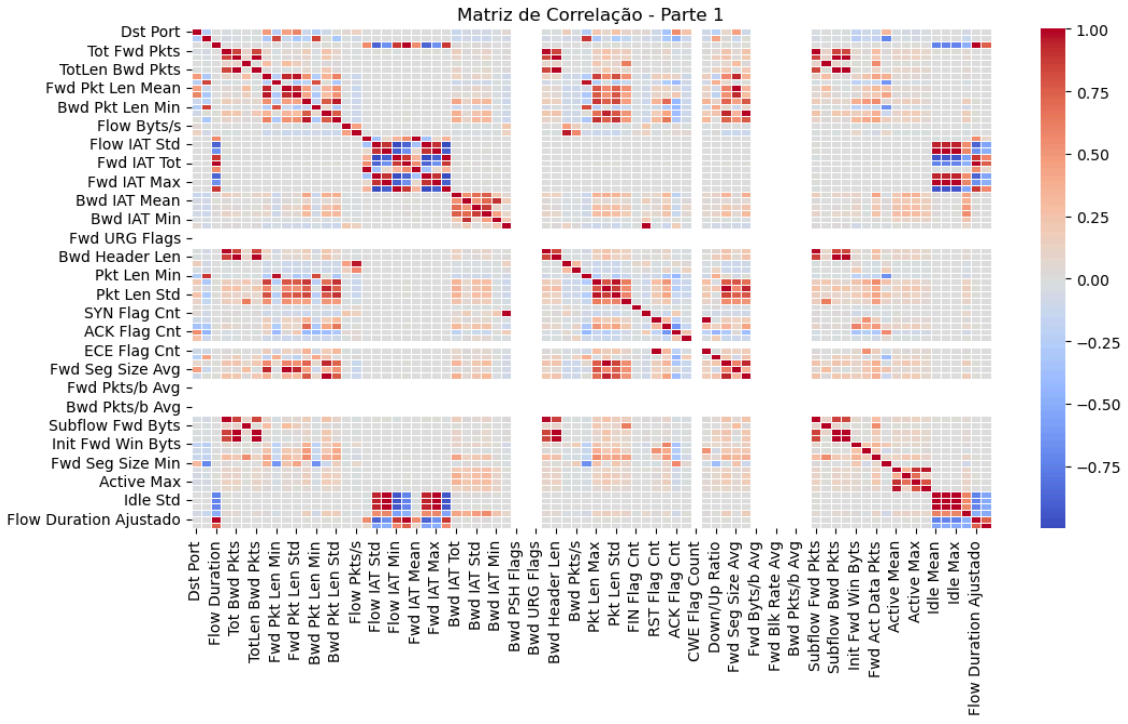
Por fim, outro ponto importante a ser destacado é a correlação entre as colunas numéricas presentes nos conjuntos de dados, tanto para 2017, quanto para 2018. Como os conjuntos dados possuem milhares de linhas e informações a serem extraídas, através da matriz de correlação é possível identificar padrões e entender como as diferentes características e comportamentos do tráfego se relacionam. Conforme mostram os gráficos.

Gráfico 11 - Matriz de Correlação 2017



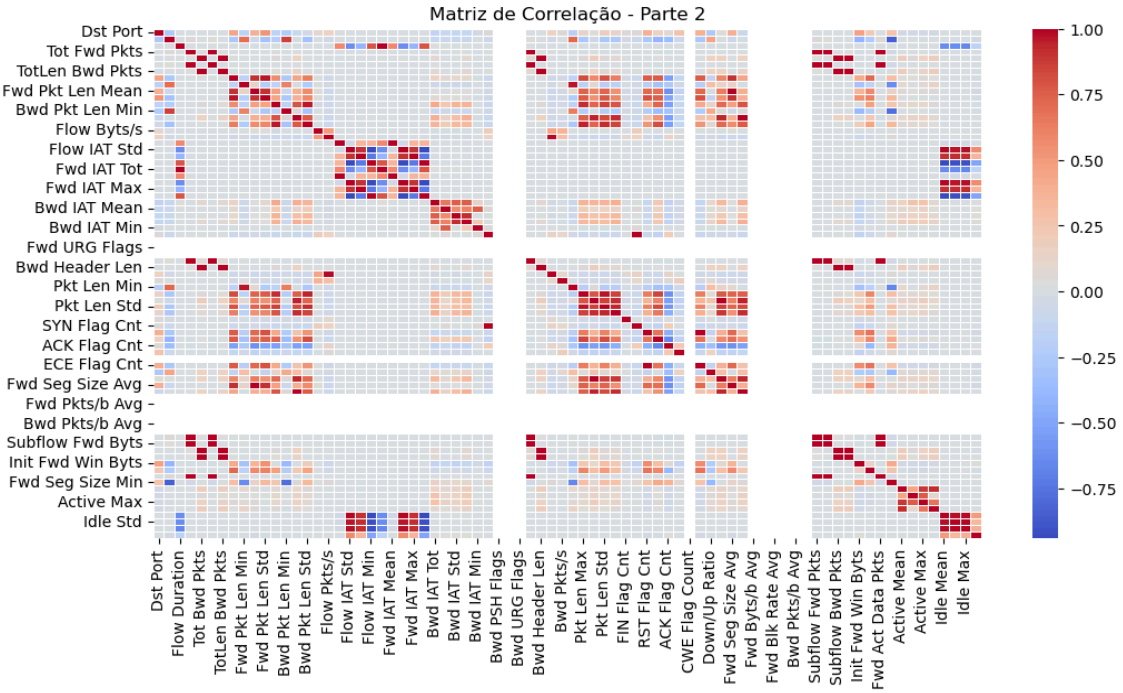
Fonte: Elaborado pelo autor

Gráfico 12 - Matriz de Correlação 2018 - Parte 1



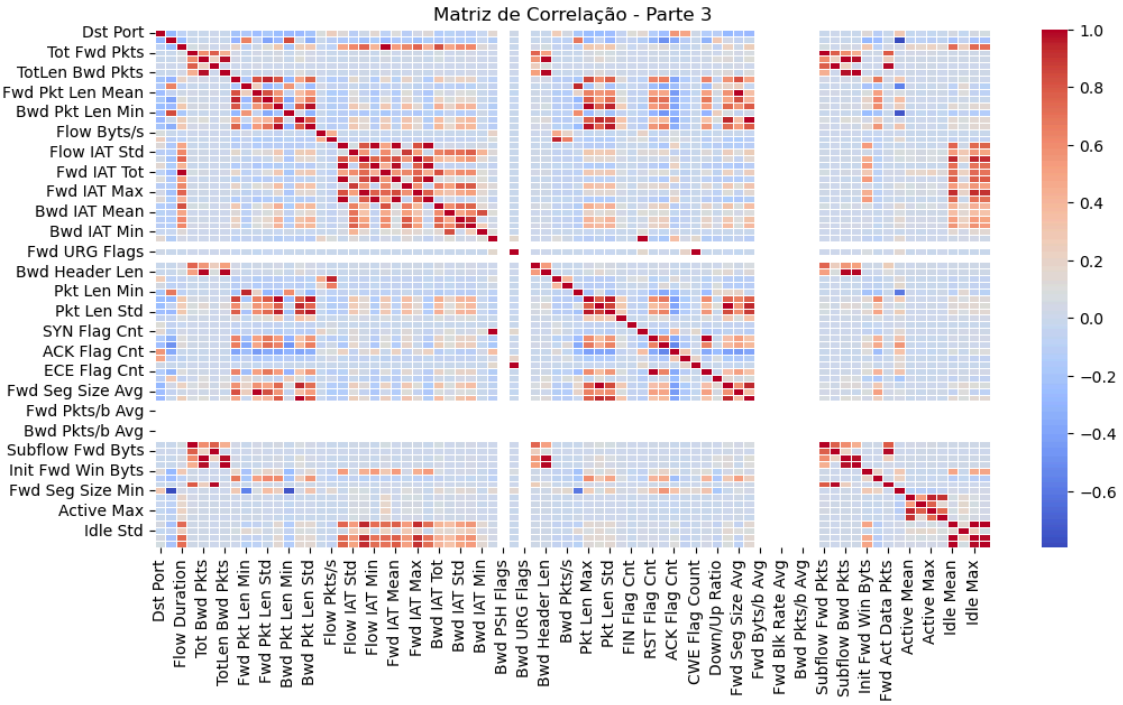
Fonte: Elaborado pelo autor

Gráfico 13 - Matriz de Correlação 2018 - Parte 2



Fonte: Elaborado pelo autor

Gráfico 14 - Matriz de Correlação 2018 - Parte 3



Fonte: Elaborado pelo autor

6. MODELO DE MACHINE LEARNING

Com base nas informações obtidas através da análise exploratória dos dados referentes aos anos de 2017 e 2018, foi possível definir o processo de construção e modelagem de um algoritmo de machine learning, que tem o seu foco na classificação binária e na classificação multiclasse.

O processo de construção, preparação e modelagem consiste, primeiramente, na criação de um dataset balanceado, ou seja, criação de um conjunto de dados com as mesmas quantidades de registros em cada categoria, por exemplo, na classificação binária foi criado um conjunto que possui 7 mil categorias benignas e 7 mil categorias maliciosas.

Após essa etapa é feita a separação do respectivo conjunto de dados em dados de treino e dados de teste. A mesma consistiu em, utilizando os dados já balanceados, separar 75% dos mesmos para o treinamento dos modelos e os outros 25% restantes para os testes. Os dados de treino têm sua utilização destinada, exclusivamente, para a criação dos modelos, já os dados de treino são utilizados durante as avaliações de desempenho.

Após a preparação os dados, o principal objetivo da modelagem deve ser a criação e avaliação de modelos de aprendizado de máquina que são capazes de detectar um possível ataque cibernético a partir de padrões que o mesmo identificou nos dados (CIC-IDS2017 e CSE-CIC-IDS2018). Com isso, ao final da modelagem, foram respondidas as seguintes perguntas:

- Qual modelo apresenta o melhor desempenho na identificação de ataques?
- Qual o melhor algoritmo para cada tipo de classificação?

6.1. Modelagem e Treinamento

Como citado, a modelagem foi feita em duas abordagens diferentes, classificação binária e classificação multiclasse.

A classificação binária trata de modelos que classificam o tráfego na rede como benigno (0) ou malicioso (1). Essa abordagem é útil quando se trata de sistemas de segurança que precisam agir de forma rápida para bloquear ameaças em tempo real. Por isso, os modelos binários, ao utilizar menos poder computacional, simplificam a tomada de decisão e facilitam a implementação em sistemas já existentes. Porém, os mesmos contam com a limitação de

não fornecer informações específicas sobre o tipo de ataque que foi detectado, o que, consequentemente, pode acarretar uma resposta imprecisa por parte da equipe de segurança.

Já na classificação multiclasse, o objetivo é identificar exatamente qual a categoria exata do tráfego em questão, isso significa que, ao invés do modelo classificar como benigno (0) ou malicioso (1), o modelos classifica os tráfegos, ou ataques, em diferentes categorias, sendo elas, por exemplo: DDoS, força bruta, SQL Injection e entre outras.

Alguns dos desafios que podem surgir com essa abordagem devido a sua complexidade é a exigência de um maior poder computacional e a possibilidade de aumento da taxa de erro nas previsão de um arquivo por conta das diversas possibilidades existentes de classificação. Porém, para equipes de segurança, onde a atuação e prevenção contra ataques não é ato primário, a classificação multiclasse é extremamente útil, pois categorizando os ataques é possível desenvolver estratégias de melhor eficácia que garantam a segurança dos usuários e dos clientes existentes dentro do sistema.

A seguir, foram explanados os modelos desenvolvidos ao longo da pesquisa mas, é importante frisar que o desenvolvimento dos modelos seguiu uma mesma lógica, tendo em vista que, na linguagem de programação em que foram feitos, linguagem python, a estrutura do código aplicada pôde ser repetida para ambos.

6.2. Classificação Binária

Para a classificação binária foram utilizados dois modelos, a Regressão Logística e o Support Vector Machine (SVM).

6.2.1. Regressão Logística

É um algoritmo supervisionado utilizado para classificação que foi construído a partir da transformação de uma função, chamada de sigmóide sobre a regressão linear. Quando se trata de um problema de regressão, a variável alvo é um número real, o que significa que, o valor que a mesma variável pode assumir está dentro do intervalo de número reais.

Por exemplo, considerando que tem-se a variável alvo — target — “Pagar” , que pode assumir o valor de 1 ou 0, sendo 1 para valor pago e 0 para valor não pago, Além disso, temos também o atributo — feature — “Salário”, que indica que, quanto maior o salário da pessoa, maiores as chances do cliente realizar o pagamento. Nesse caso, o resultado que o algoritmo irá imprimir estará dentro do intervalo [0,1]. Porém, para facilitar a interpretação dos

resultados, é utilizada a função sigmóide, ou seja, essa função retornará um único resultado cujo valor representa a probabilidade do objeto que está sendo analisado pertencer ao intervalo mencionado. Função sigmóide:

$$p = \frac{1}{1+e^{-y}}$$

p: Probabilidade de uma instância pertencer a classe que está sendo analisada
y: Número real dado pela combinação linear dos atributos utilizados na predição

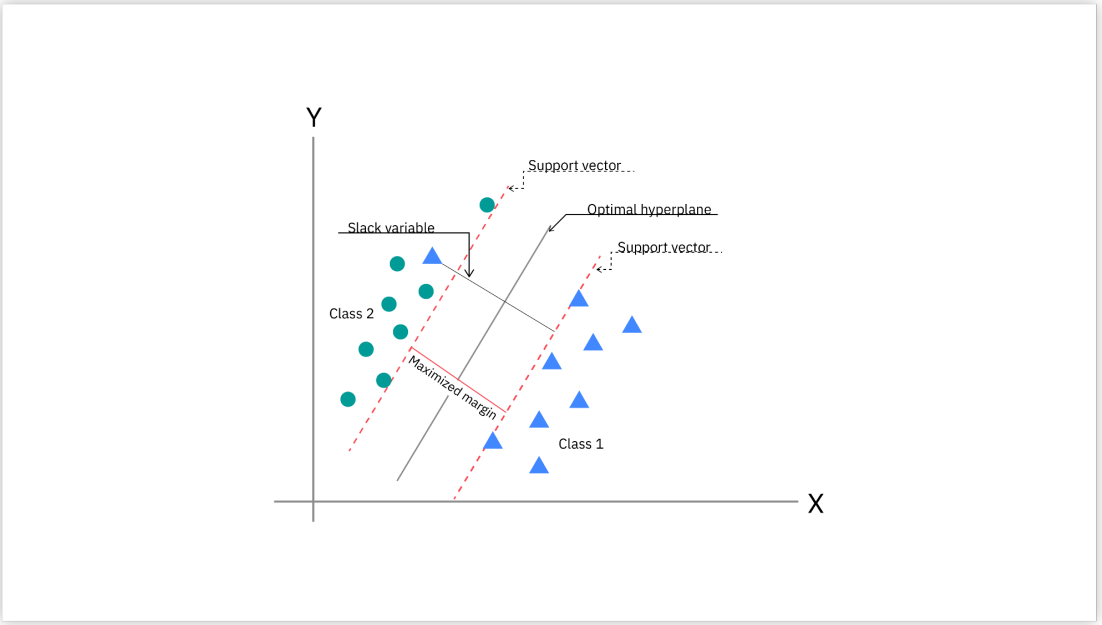
Quando criado e aplicado aos dados, o algoritmo de regressão logística, na etapa de treinamento obteve, através da validação cruzada (cross-validation[2]), uma média de resultado, entre 0 e 1, de, aproximadamente, 0.9330, o que indica resultados excelentes para o algoritmo em questão.

6.2.2. Support Vector Machine (SVM)

Assim como a regressão logística, os algoritmos de SVMs são algoritmos supervisionados que classificam os dados através de uma linha ou hiperplano que maximiza a distância entre cada classe em um espaço N-dimensional. Isso significa que, quando se tem duas classes, benigna (0) e maliciosa (1), esse algoritmo é capaz de encontrar o hiperplano ideal onde a margem entre os pontos de dados mais próximos estão maximizadas, ou seja, através dessa maximização, é possível realizar classificações precisas. Pode-se ver um através da Figura 01 abaixo:

[2] Cross-validation: Técnica de avaliação de modelos de machine learning por meio de treinamento de vários modelos em subconjuntos de dados de entrada disponíveis e avaliação deles no subconjunto complementar dos dados.

Figura 01 - Algoritmo de Classificação - SVM



Fonte: IBM - O que são máquinas de vetores de suporte (SVMs)

6.3. Classificação Multiclasse

Para a classificação multiclasse foram utilizados quatro modelos, o Random Forest, K-Nearest Neighbours (KNN) e XGBoost.

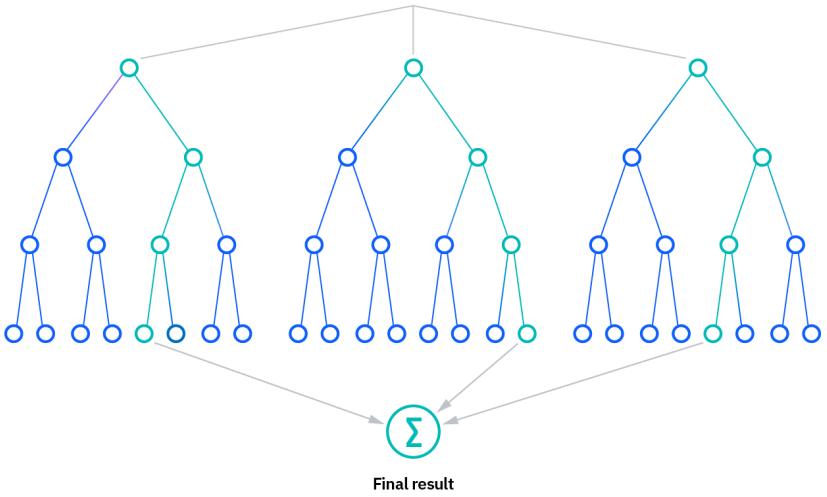
6.3.1. Random Forest

O algoritmo random forest pode ser denominado também, como “método de conjunto”, ou seja, métodos de aprendizado que são compostos por diversos classificadores, que no caso do random forest, é composto por diversas árvores de decisão, que visam alcançar um único resultado. Apesar do modelo random forest ser composto por diversas árvores de decisão, ambos tem suas diferenças que implicam diretamente no resultado final.

Atualmente, um dos métodos de conjuntos mais conhecido, é o método bagging, Nesse método, é selecionada, com substituição, uma amostra aleatória dos dados pertencentes ao conjunto de treinamento, o que significa que os pontos de dados podem ser selecionados mais de uma vez. Após a geração de diversas amostras de dados, os modelos são treinados de forma independente.

À vista disso, o algoritmo random forest é uma extensão do método bagging, descrito acima. Ao utilizar o bagging e a aleatoriedade de características, também chamada de feature bagging, o algoritmo cria uma floresta de árvores de decisão, onde essas árvores não se correlacionam. Após a criação de uma floresta de árvores, ambas são treinadas de forma independente gerando previsões que, em sua maioria, fornecem uma estimativa mais precisa. A figura a seguir ilustra o funcionamento e a estrutura de um algoritmo random forest.

Figura 02 - Random Forest



Fonte: IBM - O que é random forest?

6.3.2. K-Nearest Neighbours

Os modelos de algoritmo k vizinhos mais próximos, é um classificador que utiliza a proximidade para realizar previsões sobre um conjunto de pontos de dados individuais. Para que isso seja possível, o algoritmo atribui um rótulo de classe com base em uma votação por pluralidade, ou seja, o rótulo mais frequente existente em um certo ponto de dados é utilizado. A figura abaixo demonstra o funcionamento do algoritmo KNN.

Figura 03 - Diagrama KNN

Fonte: IBM - O que é o algoritmo k-nearest neighbours (KNN)?

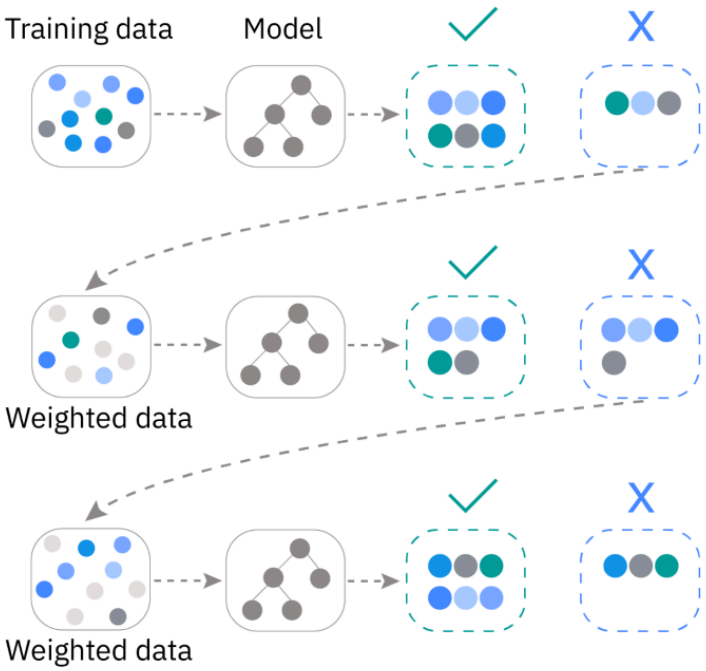
6.3.3. XGBoost

Segundo a IBM (2025), "o XGBoost é uma biblioteca otimizada de aprendizado de máquina que usa árvores de decisão com esforço gradativo" (IBM, 2025). Esse modelo é utilizado como uma alternativa às árvores de decisão, tendo em vista que as árvores são propensas a overfitting[3], o método de Boosting, também conhecido como método de aprendizagem em conjunto, é um método que combina árvores fracas individuais (aprendiz fraco) para formar um aprendiz forte, ou seja, cada aprendiz fraco é treinado em sequência para corrigir os erros cometidos pelas árvores anteriores. Esse processo ocorre por milhares de iterações até as árvores que antes eram fracas, se tornem árvores fortes (aprendiz forte).

Os algoritmos de boosting e random forest são frequentemente empregados como técnicas de aprendizado em conjunto que utilizam árvores de decisão individuais buscando melhorar o desempenho da sua predição. Porém, enquanto modelos random forests são baseados e utilizam o método bagging, ou seja, selecionam amostras aleatórias e treinam cada uma de forma independente de cada árvore para, ao final, combinar suas decisões, os algoritmos de boosting, utilizam uma abordagem aditiva, ou seja, os aprendizes fracos são treinados de forma sequencial de modo que, o próximo a ser treinado, consiga corrigir os erros dos modelos anteriores até que se alcance o resultado esperado. A figura abaixo ilustra o processo que ocorre no XGBoost.

[3] Overfitting: quando um algoritmo se adapta excessivamente ou até mesmo de forma precisa aos dados de treinamento, levando a um modelo que não consegue fazer previsões ou conclusões precisas com outros dados que não sejam os de treinamento

Figura 04 - XGBoost



Fonte: IBM - O que é XGBoost?

6.4. Testes e Avaliação

Após o desenvolvimento e treinamento, foi utilizada a parcela separada somente para testes a fim de aferir qual o desempenho obtido dos modelos em questão. Porém, para garantir a eficácia na avaliação foram utilizadas algumas métricas, como matriz de confusão, F1-Score, Recall e Precision. Ambas são amplamente utilizadas e reconhecidas no meio da tecnologia para definir se um modelo é ótimo ou não. Na tabela a seguir, é possível ver a média de desempenho dos modelos de acordo com as métricas aplicadas.

Quadro 3 - Média de resultados dos modelos nos testes

Modelos	Precisão	Recall	F1-Score
Regressão Logística	0,935	0,935	0,935
Support Vector Machine	0,965	0.927	0.942

Random Forest	0.994	0.993	0.993
K-Nearest Neighbours	0.970	0.973	0.973
XGBoost	0.999	0.999	0.999

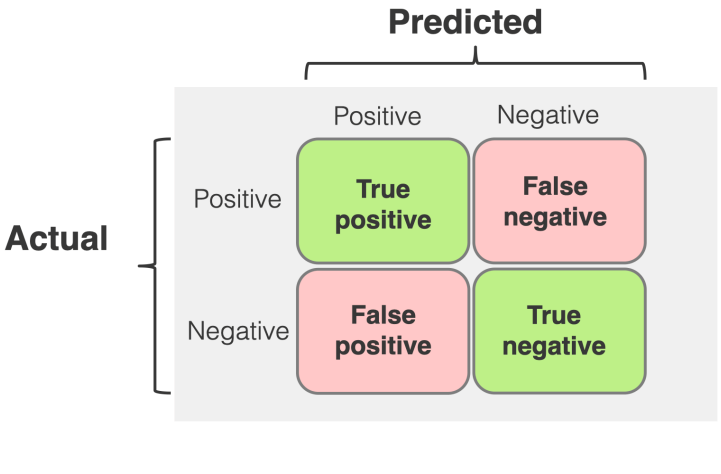
Outra métrica utilizada, como citado anteriormente, foi a matriz de confusão. Essa métrica é, também, um método de visualização de algoritmos de classificação. Funciona como uma tabela que mostra o número de instâncias verdadeiras de uma classe específica em relação ao número de instâncias previstas dessa mesma classe.

Uma matriz de confusão é uma tabela que permite visualizar o desempenho de um modelo de classificação. Ela compara os valores preditos pelo modelo com os valores reais do conjunto de dados e além disso, a matriz de confusão é composta pela seguinte estrutura, dividida em quatro quadrantes:

- **1º Quadrante:** Verdadeiro Positivo (TP), ou seja, o modelo previu "positivo" e o valor real também era "positivo";
- **2º Quadrante:** Falso Negativo (FN), ou seja, o modelo previu "negativo", mas o valor real era "positivo";
- **3º Quadrante:** Falso Positivo (FP), ou seja, modelo previu "positivo", mas o valor real era "negativo";
- **4º Quadrante:** Verdadeiro Negativo (TN), ou seja, modelo previu "negativo" e o valor real também era "negativo";

Por essa matriz, é possível avaliar e calcular algumas métricas que são de extrema importância para a modelagem de um algoritmo de ML, como a acurácia, sensibilidade e F1-Score. A figura a seguir exemplifica a estrutura de uma matriz de confusão.

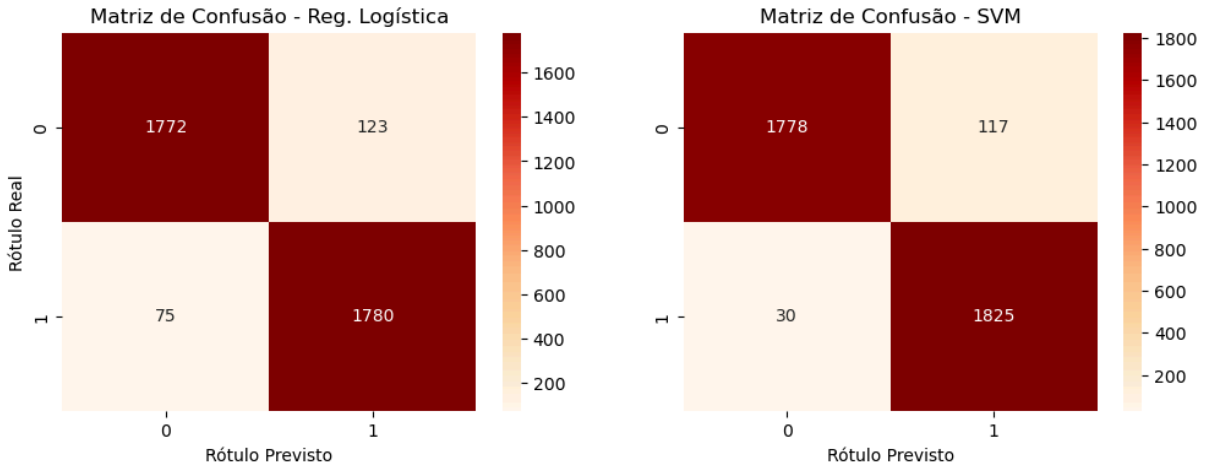
Figura 05 - Matriz de confusão



Fonte: EVIDENTLY AI - Confusion Matrix

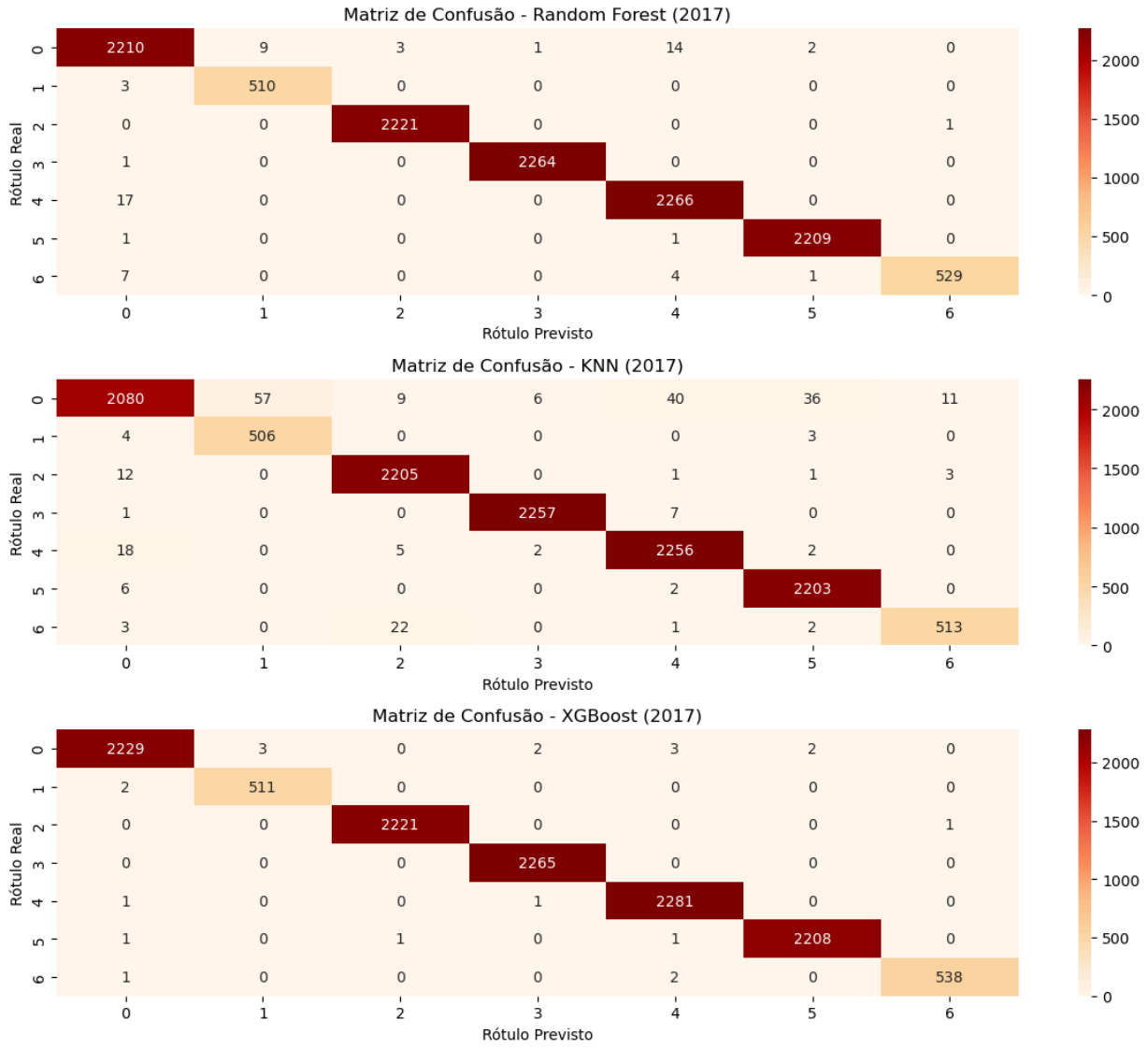
Ao aplicá-la aos modelos de classificação binária e multiclasse, pode-se chegar aos resultados mostrados nos gráficos 15 e 16, onde, de acordo com estes gráficos, é possível perceber que os modelos alcançaram resultados ótimos, uma vez que as matrizes obedecem ao Verdadeiro Positivo (TP) e ao Verdadeiro Negativo (TN).

Gráfico 15 - Matriz de confusão - Binária



Fonte: Elaborado pelo autor

Gráfico 16 - Matriz de confusão - Multiclasse



Fonte: Elaborado pelo autor

Ao analisar os resultados, foi notado que, para a classificação binária, o modelo de maior desempenho foi o Support Vector Machine — SVM. Já para a classificação multiclasse, o XGBoost alcançou o melhor resultado.

Desse modo, e buscando simular um cenário real, foi criada uma interface em Streamlit que permite ao usuário interagir e entender um pouco mais sobre como funcionam os modelos desenvolvidos.

7. INTERFACE DO USUÁRIO

A interface do sistema foi feita utilizando o Streamlit, um framework de código aberto destinado à criação de aplicações na área de ciência de dados e aprendizado de máquina. O objetivo principal foi desenvolver uma aplicação web leve e de fácil navegação e que, ao mesmo tempo, fornecesse ao usuário a interação com os modelos de machine learning, sem a necessidade de conhecimento técnico profundo sobre o assunto.

7.1. Funcionalidades

A aplicação adota um layout e paleta de cores pensada na experiência visual do usuário final. O seu layout é configurado em tela ampla e com a barra lateral inicialmente recolhida, otimizando o uso do espaço principal da aplicação. Através disso, algumas de suas principais funcionalidades são:

- Upload de arquivos CSV com dados de tráfego de rede;
- Detecção automática do tipo de classificação: Binária ou Multiclasse;
- Escolha de modelos de ML previamente treinados;
- Visualização dos resultados com gráficos;
- Barra lateral com breves informações sobre o projeto.

7.2. Tecnologias Utilizadas

Tendo em vista que todas as etapas foram feitas utilizando a linguagem de programação Python, a interface seguiu a mesma lógica, a fim de evitar possíveis problemas técnicos. Abaixo podemos ver algumas bibliotecas e recursos utilizados:

- Streamlit: Framework principal da interface;
- Python: Linguagem base de todo o sistema e, também, do projeto;
- Matplotlib e Seaborn: Bibliotecas utilizadas para gerar gráficos interativos;
- Joblib: Biblioteca utilizada para carregar os modelos pré-treinados salvos no arquivo modelos.pkl;
- Pandas: Biblioteca utilizada para o tratamento e manuseio de dados;
- VSCode: Ambiente de desenvolvimento utilizado na criação da aplicação.

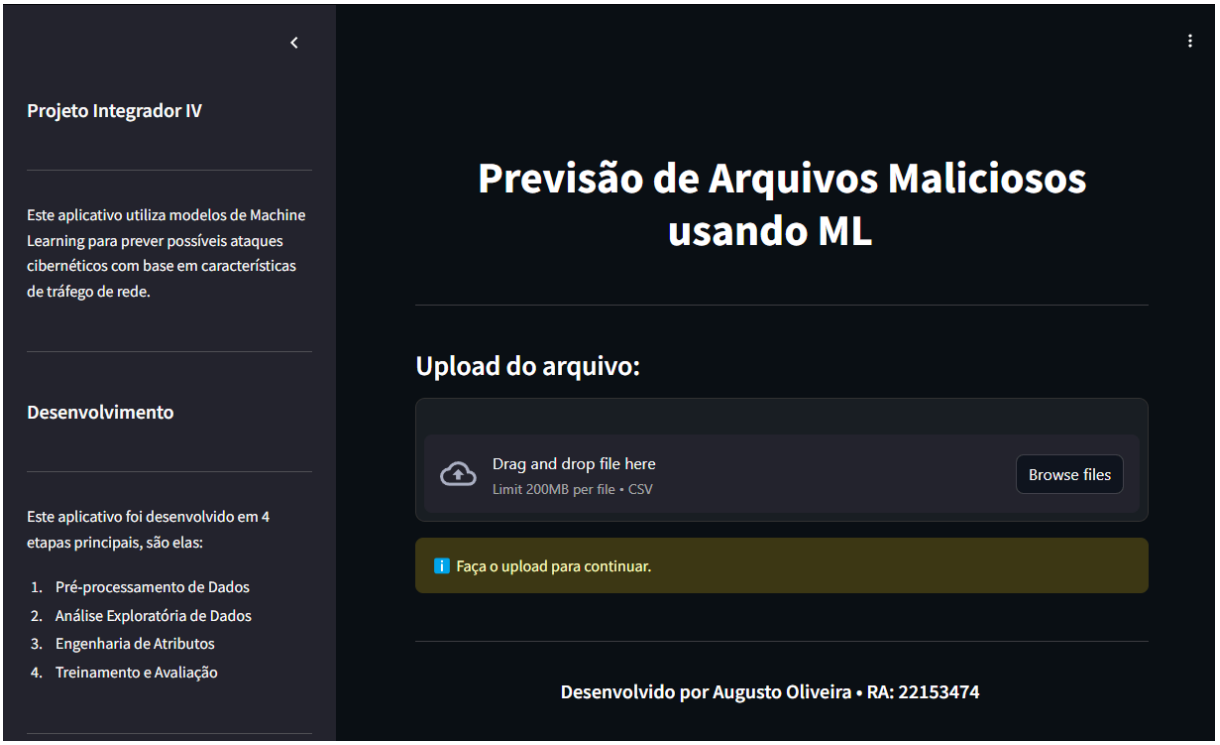
7.3. Instalação e execução

Todas as etapas necessárias de instalação e execução local da aplicação foram descritas em um repositório do GitHub. Para acessá-lo, clique no link a seguir: https://github.com/gut0oliveira/Data-Science-Capstone

7.4. Funcionamento

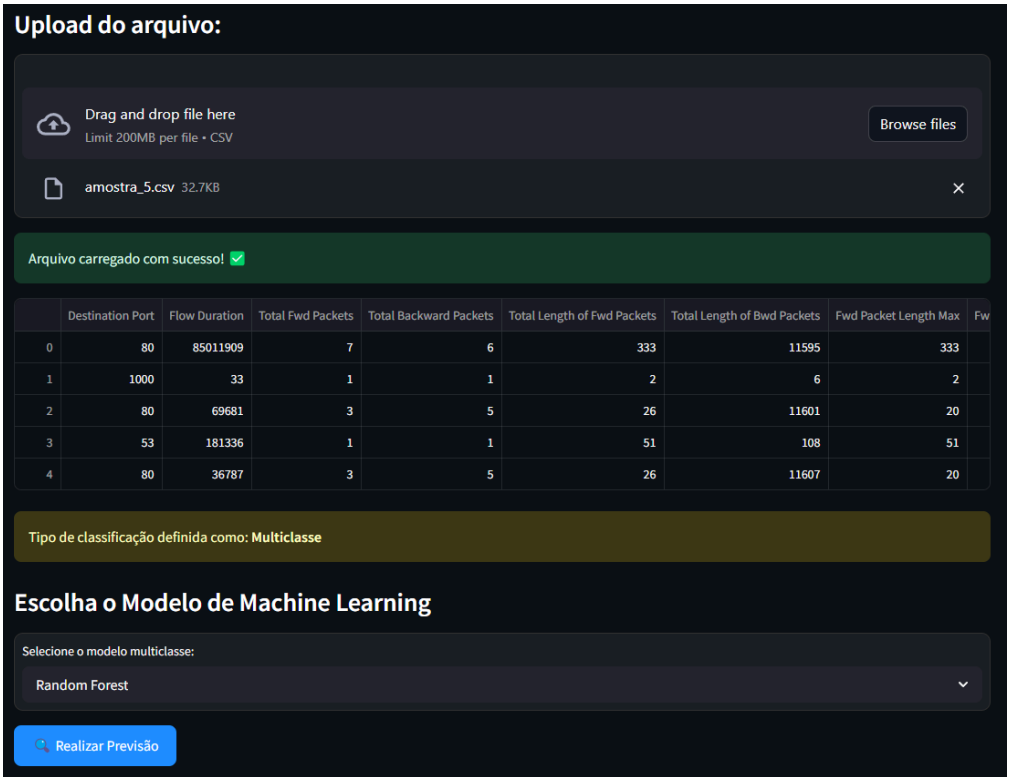
Após seguir os passos descritos no repositório do GitHub, o usuário conseguirá interagir, testar e verificar a eficácia obtida pelos modelos no que diz respeito às amostras de dados aleatórios criados e armazenados dentro do próprio projeto. Uma vez executada e funcionando, o usuário verá uma tela como as imagens abaixo:

Figura 06 - Aplicação 1



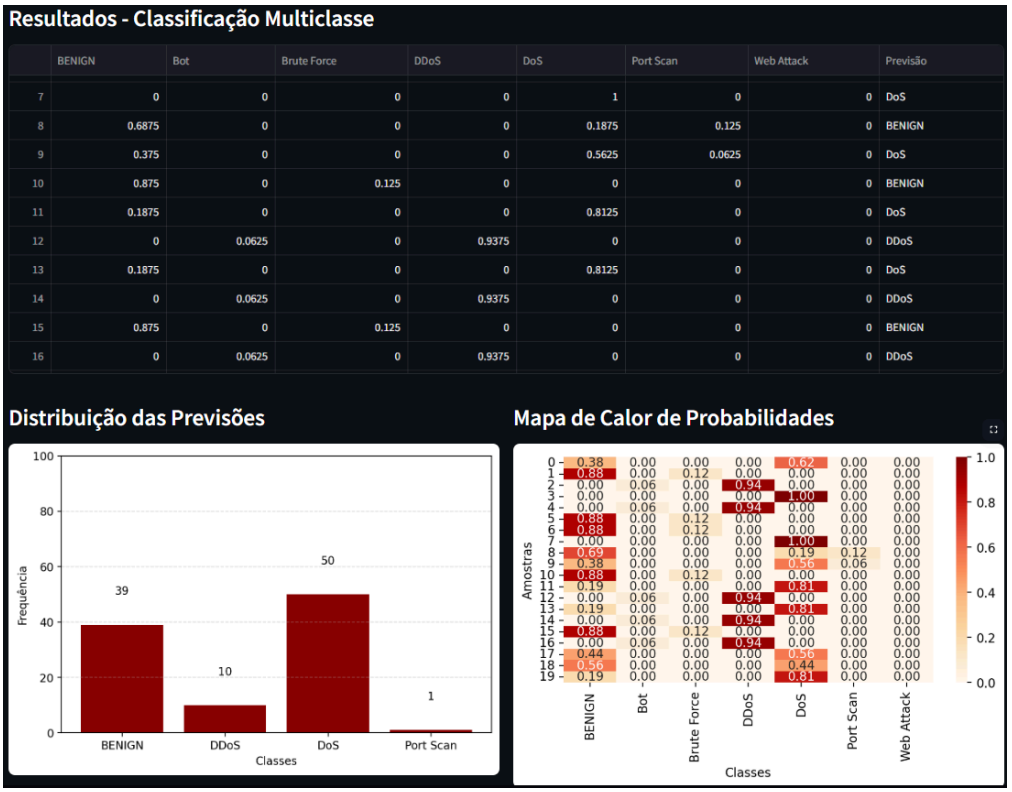
Fonte: Elaborado pelo autor

Figura 07 - Aplicação 2



Fonte: Elaborado pelo autor

Figura 08 - Aplicação 3



Fonte: Elaborado pelo autor

8. CONSIDERAÇÕES FINAIS

Este projeto de pesquisa teve como proposta investigar e colocar em prática o uso da engenharia reversa associada a modelos de machine learning para a identificação e prevenção de malwares nos tráfegos de rede, tendo como principal foco a análise do impacto e da aplicabilidade dessas tecnologias dentro da área da cibersegurança. Através da criação de algoritmos e experimentos baseados nos conjuntos de dados CIC-IDS2017 e CSE-CIC-IDS2018, conseguiu-se realizar a previsão e verificar a eficácia, tanto da classificação binária — benigno ou malicioso — quanto da classificação multiclasse — ataques específicos —.

Ao olhar para a pergunta de pesquisa — "Como detectar e se prevenir de malwares e qual o seu impacto em sistemas de cibersegurança?" — pode-se concluir que a mesma foi respondida ao longo do estudo com evidências práticas. Provou-se que, com modelos estruturados, como o Support Vector Machine (SVM) e o XGBoost, explanados nos tópicos anteriores, a identificação com elevada acurácia, é possível, desde que feita da maneira mais adequada para o contexto em que está inserida. Ademais, após a análise das portas de destino mais vulneráveis e dos tipos de ataques mais frequentes, é perceptível a importância de reforçar as ações preventivas específicas e estratégicas para a respectiva variante de ataque.

No que diz respeito aos objetivos gerais da pesquisa, os mesmos foram alcançados através da criação dos algoritmos e de uma interface interativa que permite ao usuário, de forma automatizada, ter uma previsibilidade do risco de segurança que um arquivo pode ou não oferecer. Quantos aos objetivos específicos, através da literatura, da prática e da utilização de modelos eficazes, testados e avaliados por métricas amplamente reconhecidas no mercado e na área da ciência de dados, é constatável que os respectivos objetivos atingidos com êxito.

Por fim, este projeto apresenta importantes contribuições para o cenário atual da cibersegurança no mundo. Mostrando que, o emprego de técnicas transparentes na análise de um malware, facilita e abre caminhos para futuras investigações, como o uso de Deep Learning — Aprendizado Profundo — em ambientes com tráfego de rede elevado e, também, a análise de arquivos maliciosos em tempo real.

Portanto, a junção de engenharia reversa e machine learning não é somente viável, mas também, altamente propícia como solução estratégica de defesa cibernética frente ao rápido avanço das ameaças digitais.

9. REFERÊNCIAS

AMAZON WEB SERVICES. *Validação cruzada*. Disponível em: https://docs.aws.amazon.com/pt_br/machine-learning/latest/dg/cross-validation.html. Acesso em: 13 maio 2025.

CRONAPP. *Engenharia reversa*. Disponível em: https://blog.cronapp.io/engenharia-reversa/. Acesso em: 28 fev. 2025.

CRONAPP. *Engenharia reversa de software*. Disponível em: https://blog.cronapp.io/engenharia-reversa-de-software/#Como_a_engenharia_reversa_de_software_funciona_na_pratica. Acesso em: 28 fev. 2025.

DEVMedia. *Trabalhando com engenharia reversa – Revista Engenharia de Software Magazine 59*. Disponível em: https://www.devmedia.com.br/trabalhando-com-engenharia-reversa-revista-engenharia-de-software-magazine-59/28203. Acesso em: 28 fev. 2025.

ESCOLA DE DADOS. *Leve seus projetos de dados a outro nível utilizando Streamlit*. Disponível em: https://escoladedados.org/tutoriais/leve-seus-projetos-de-dados-a-outro-nivel-utilizando-streamlit/. Acesso em: 2 mai. 2025.

EVIDENTLY AI. *Confusion Matrix*. Disponível em: https://www.evidentlyai.com/classification-metrics/confusion-matrix. Acesso em: 13 maio 2025.

IBM. *A história do malware*. Disponível em: https://www.ibm.com/br-pt/think/topics/malware-history. Acesso em: 10 mar. 2025.

IBM. *O que é malware?*. Disponível em: https://www.ibm.com/br-pt/topics/malware. Acesso em: 10 mar. 2025.

IBM. *O que são SVMs?*. Disponível em: https://www.ibm.com/br-pt/think/topics/support-vector-machine. Acesso em: 1 abr. 2025.

IBM. *Random Forest*. Disponível em: https://www.ibm.com/br-pt/think/topics/random-forest. Acesso em: 1 abr. 2025.

IBM. *Decision Trees*. Disponível em: https://www.ibm.com/br-pt/topics/decision-trees. Acesso em: 1 abr. 2025.

IBM. *K-Nearest Neighbors (KNN)*. Disponível em: https://www.ibm.com/br-pt/topics/knn. Acesso em: 1 abr. 2025.

IBM. *XGBoost*. Disponível em: https://www.ibm.com/br-pt/think/topics/xgboost. Acesso em: 1 abr. 2025.

IBM. *Overfitting*. Disponível em: https://www.ibm.com/br-pt/think/topics/overfitting. Acesso em: 13 mai. 2025.

MARKOVML. *LIME vs SHAP: uma comparação prática*. Disponível em: https://www.markovml.com/blog/lime-vs-shap. Acesso em: 15 abr. 2025.

MICROSOFT. *O que é malware?*. Disponível em: https://www.microsoft.com/pt-br/security/business/security-101/what-is-malware. Acesso em: 10 mar. 2025.

OTTONI, Otton. *Engenharia reversa*. Disponível em: http://www2.ic.uff.br/~otton/graduacao/informaticaI/apresentacoes/eng_reversa.pdf. Acesso em: 28 fev. 2025.

REMÍGIO, M. S. *Regressão logística (Logistic Regression)*. Medium, 2021. Disponível em: https://medium.com/@msremigio/regressão-logística-logistic-regression-997c6259ff9a. Acesso em: 1 abr. 2025.

SINGH, Simon. *O livro dos códigos: a história da criptografia, da antiguidade à era da informática*. Rio de Janeiro: Record, 2000.

UNIVERSITY OF NEW BRUNSWICK. *CIC IDS 2017*. Disponível em: https://www.unb.ca/cic/datasets/ids-2017.html. Acesso em: 20 mar. 2025.

UNIVERSITY OF NEW BRUNSWICK. *CSE-CIC-IDS2018*. Disponível em: https://www.unb.ca/cic/datasets/ids-2018.html. Acesso em: 20 mar. 2025.

UNIVERSITY OF NEW BRUNSWICK. *Canadian Institute for Cybersecurity – About*. Disponível em: https://www.unb.ca/cic/about/index.html. Acesso em: 15 mar. 2025.

SHARAFALDIN, Iman; LASHKARI, Arash Habibi; GHORBANI, Ali A. *Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization*. In: *4th International Conference on Information Systems Security and Privacy (ICISSP)*, Portugal, jan. 2018.

WISHBOX. *Engenharia reversa: o que é, como funciona e onde aplicar*. Disponível em: https://www.wishbox.net.br/blog/engenharia-reversa/. Acesso em: 2 mar. 2025.

10. APÊNDICE A - GLOSSÁRIO DE TERMOS TÉCNICOS

Adwares - Programas que exibem ou instalam publicidade automaticamente no computador do usuário.

Backups locais - Cópias de segurança armazenadas em dispositivos físicos locais, como HDs externos.

Backups na nuvem - Cópias de segurança armazenadas em dispositivos virtuais.

Cavalos de Tróia (Trojans) - Malware disfarçado de software legítimo que executa ações maliciosas sem o conhecimento do usuário.

Cibercriminoso - Indivíduo que realiza atividades criminosas através de redes e sistemas computacionais.

Colunas categóricas - Colunas de um dataset cujos valores representam categorias, como "tipo de ataque" ou "classe".

Data Frame - Estrutura de dados bidimensional (como uma tabela) amplamente usada em bibliotecas Python como Pandas.

Datasets - Conjuntos de dados organizados utilizados para treinamento e teste de modelos.

Feature - Atributo ou variável utilizada como entrada em um modelo de machine learning.

F1-Score - Média harmônica entre precisão e recall, usada para avaliar o desempenho de modelos.

Framework - Conjunto de bibliotecas e ferramentas que facilitam o desenvolvimento de aplicações.

Framework de código aberto - Framework cujo código-fonte está disponível publicamente para modificação e uso livre.

GitHub - Plataforma para hospedagem e controle de versão de projetos de software.

IDS - Intrusion Detection System – sistema de detecção de intrusos que monitora atividades suspeitas na rede.

IA (Inteligência Artificial) - Área da ciência da computação que busca criar sistemas capazes de simular a inteligência humana.

Malwares sem arquivo - Tipo de malware que opera diretamente na memória RAM, sem deixar rastros em disco.

Memória RAM - Memória de acesso aleatório usada para armazenar dados temporários enquanto programas estão em execução.

Mineradores de criptomoeda - Softwares maliciosos que usam o poder computacional da máquina infectada para minerar criptomoedas.

ML (Machine Learning) - Subárea da IA que utiliza algoritmos para aprender padrões a partir de dados e fazer previsões.

Precisão (Precision) - Proporção de acertos entre os positivos previstos por um modelo.

Recall (Sensibilidade) - Capacidade de um modelo identificar corretamente os positivos reais.

ROC-AUC - Área sob a curva ROC, métrica que avalia a capacidade de distinção entre classes.

Rootkits - Conjunto de ferramentas que permite acesso privilegiado e oculto a um sistema.

Saída booleana - Resultado binário de um modelo, que pode ser 0 (falso) ou 1 (verdadeiro).

Função Sigmóide - Função matemática que transforma valores reais em uma saída entre 0 e 1.

SIEM - Security Information and Event Management – sistema que coleta e analisa informações de segurança em tempo real.

Target - Variável alvo em modelos de machine learning, ou seja, o que se deseja prever.

Usuário final - Pessoa que efetivamente utiliza o sistema desenvolvido.

VPN - Virtual Private Network – tecnologia que cria uma conexão segura entre o usuário e a rede.

Worms - Tipo de malware que se propaga automaticamente entre computadores sem intervenção humana.