UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

Marcos Rogério Silveira

Detecção de domínios maliciosos por meio de DNS passivo utilizando XGBoost

São José do Rio Preto
2021

Marcos Rogério Silveira

Detecção de domínios maliciosos por meio de DNS passivo utilizando XGBoost

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de São José do Rio Preto.

Financiador: NIC.br - Proc. 2764/2018

Orientador:

Prof. Dr. Adriano Mauro Cansian

São José do Rio Preto
2021

S587d

Silveira, Marcos Rogério

Detecção de domínios maliciosos por meio de DNS passivo utilizando XGBoost / Marcos Rogério Silveira. -- São José do Rio Preto, 2021

67 f. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto

Orientador: Adriano Mauro Cansian

1. Ciência da computação. 2. Computadores Medidas de segurança. 3. Crime por computador. 4. Domínios maliciosos. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Marcos Rogério Silveira

Detecção de domínios maliciosos por meio de DNS passivo utilizando XGBoost

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de São José do Rio Preto.

Financiador: NIC.br - Proc. 2764/2018

Comissão Examinadora

Prof. Dr. Adriano Mauro Cansian
Unesp - Câmpus São José do Rio Preto
Orientador

Prof. Dr. Leandro Alves Neves
Unesp - Câmpus São José do Rio Preto

Prof. Dr. Robson de Oliveira Albuquerque
UnB - Universidade de Brasília

São José do Rio Preto
19 de Janeiro de 2021

Agradecimentos

Aos meus pais, Ana Cristina Donda Silveira e Ademir Silveira, por acreditarem nesse sonho comigo e não medirem esforços para estarem ao meu lado e me apoiando incondicionalmente.

À minha namorada, Alexa Chaves, por acreditar em mim, me ajudar em todos os momentos.

Ao meu orientador, Prof. Dr. Adriano Mauro Cansian, que acreditou em mim e me deu a oportunidade de poder realizar o estudo descrito neste trabalho e participar do Laboratório ACME!.

Ao Laboratório ACME! e a todos os seus membros, que durante estes anos foi possível criar laços de amizade, e colaborar um para com os outros. Gostaria de agradecer em especial ao "Team DNS"e ao amigo Leandro Bertini Lara Gonçalves.

Agradeço, também, à Universidade Estadual Paulista "Júlio de Mesquita Filho", campus de São José do Rio Preto, pelo curso de pós-graduação ali ministrado por professores de excelência.

À Mnemonic por ter cedido gentilmente a base de dados de DNS Passivo utilizada neste trabalho.

O presente trabalho foi realizado com o apoio do Núcleo de Informação e Coordenação do Ponto BR (NIC.br) - Processo nº 2764/2018 - CCP, o qual agradeço.

"Restlessness is discontent and discontent is the first necessity of progress."

-Edison, Thomas A.

Resumo

Este trabalho apresenta um método para detecção de domínios maliciosos por meio do tráfego de DNS passivo. Para tanto, a abordagem utilizada é um *dataset* de DNS passivo como fonte de dados para a tarefa de classificação dos domínios entre maliciosos e legítimos. A partir deste *dataset*, são extraídas doze *features* exclusivas do tráfego DNS. Os registros presentes no *dataset* DNS passivo são rotulados utilizando *allowlists* e *blocklists* de nomes de domínios e IPs. Para balanceamento das classes, foi utilizado a técnica de Random Undersampling. Na etapa de treinamento, foram utilizados e comparados o desempenho dos três algoritmos de aprendizado de máquina supervisionado baseados em árvores de decisão. Os modelos foram testados considerando suas capacidades de identificar domínios maliciosos, o modelo com melhor desempenho foi o que utilizou o algoritmo XGBoost, com uma AUC média de 0,9776 e sem indicativos de *overfitting* presente.

Palavras-chave: Domain Name System, Domínios Maliciosos, DNS Passivo, Machine Learning.

Abstract

This paper presents a method for detecting malicious domains through passive DNS traffic. For this, the approach used is a passive DNS dataset as a data source for the task of classifying the domains between malicious and legitimate. From this dataset, twelve exclusive features of DNS traffic are extracted. The records present in the passive DNS dataset are labeled using allowlists and blocklists of domain names and IPs. To balance the classes, the Random Undersampling technique was used. In the training stage, the performance of the three supervised machine learning algorithms based on decision trees was used and compared. The models were tested considering their ability to identify malicious domains, the model with the best performance was the one that used the XGBoost algorithm, with an average AUC of 0.9776 and with no indications of overfitting present.

Keywords: *Domain Name System, Malicious Domain, Passive DNS, Machine Learning.*

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

AS *Sistema Autônomo*

AUC *Area under the ROC Curve*

C&C *Command and Control*

ccTLD *Country Code top-level domain*

CNAME *Canonical Name*

DDoS *Distributed Denial of Service*

DNS *Domain Name System*

DT *Decision Tree*

E Especificidade

FN Falso Negativo

FP Falso Positivo

FQDN *Fully Qualified Domain*

gTLD *Generic Top Level Domain*

IoT *Internet of Things*

IP *Internet Protocol*

ISP *Internet Service Provider*

NS *Name Server*

pDNS *Passive DNS*

PTR *Pointer*

RF *Random Forest*

ROC *Receiver Operating Characteristics*

RRs *Resource Records*

RUS *Random Undersampling*

SOA *Start of Authority*

TFN Taxa de Falso Negativo

TFP Taxa de Falso Positivo

TLD *Top Level Domain*

TTL *Time to Live*

TVN Taxa de Verdadeiro Negativo

TVP Taxa de Verdadeiro Positivo

VN Verdadeiro Negativo

VP Verdadeiro Positivo

XGBoost *Extreme Gradient Boosting*

Sumário

Capítulo 1

Introdução

Antes da internet ser amplamente difundida, a resolução dos nomes de domínios era feita por meio de um arquivo em texto plano armazenado localmente no computador como descrito na RFC 952 (HARRENSTIEN; STAHL; FEINLER, 1985). Este arquivo armazenado era denominado de *hosts.txt*, onde possuía nomes de domínios e seus respectivos IPs.

Com a expansão exponencial da internet, a solução até então utilizada passou a se tornar inviável e o surgimento de serviço distribuído para a resolução dos nomes de domínio foi criado, o DNS *(Domain Name System)* (MOCKAPETRIS, 1987a) (MOCKAPETRIS, 1987b). O DNS se tornou, desse modo um protocolo essencial na internet, tendo como sua principal função traduzir nomes de domínio em IPs (Internet Protocol), e a resolução reversa, na qual ele traduz IP em nome de domínios.

Devido a importância que o DNS possui na internet, atacantes o utilizam para realização de atividades maliciosas como disseminação de malware, *botnets*, *fast-flux domains* e *phishings*. Uma forma de realizar a detecção destes domínios maliciosos, é através da utilização de *blocklists*, entretanto, para que um domínio esteja presente em uma *blocklist*, é necessário que seja reportado por um usuário, e que seja analisado por um responsável por manter a *blocklist* em questão, demorando assim alguns dias até que essa intervenção humana seja concluída, e o domínio inserido na lista. Portanto, há uma necessidade de manter a internet um ambiente seguro, realizando assim a detecção de forma automática destes domínios maliciosos.

1.1 Motivação e Justificativas

O Registro.br é o *registry* (autoridade de registro) responsável pelo `.br`, sendo ele o *Country Code Top-Level Domain* (ccTLD) do Brasil. Atualmente, há registrados

4.039.913 domínios `.br` com uma média de registro de 4 mil domínios por dia (REGISTRO.BR, 2019). No mundo há um total de 525.677.429 domínios registrados, divididos sobre os 1.581 TLDs existentes (STAT, 2020), e portanto, há necessidade de se garantir que todos esses domínios sejam legítimos, ou seja, não serão usados para fins maliciosos como a disseminação de *malwares*, a facilitação de *command and control* (C&C), o envio de mensagens de spam, páginas de *phishing*, ataques de negação de serviço (DDoS) e *botnets*.

Conforme o Relatório de Ameaças à Internet 2019, divulgado pela Symantec, ataques web aumentaram cerca de 56% em 2018. Assim, 1 em cada 10 URLs analisadas pela empresa, foi identificada como maliciosa (SYMANTEC, 2019) e acredita-se que entre 16 e 25% dos computadores conectados à internet são membros de botnets (STURGEON, 2007) (ASSADHAN et al., 2009). O relatório da *Kaspersky* referente ao primeiro trimestre de 2019 relata que o Brasil é o país com a maior participação de usuários atacados por *phishers*, com 21,66%, com uma alta de 1,53 pontos percentuais em relação ao trimestre anterior, em que o país também liderou o ranking (VERGELIS; SHCHERBAKOVA; SIDORINA, 2019). Cerca de 12,9% dos usuários de internet no Brasil acessaram pelo menos um site malicioso entre abril e junho de 2020. A média global para este mesmo período foi de 8,26% (KULIKOVA; SIDORINA; SHCHERBAKOVA, 2020).

Um cenário típico de ataques são campanhas de *phishing*, em que os atacantes criam sites muito semelhantes aos originais, com URL semelhante ao site original, e usuários desavisados acessam estas páginas da web e inserem seus dados confidenciais, como número e senha do cartão de crédito, conta bancária e senha para acesso (Ex: *internet banking*). Há outros cenários de ataques, como o de *botnets*, por exemplo, em que o invasor infecta o computador de um usuário final transformando este host em um bot, no qual responde a comandos de um *bot master*, e normalmente são usados para ataques DDoS, roubo de informações confidenciais destes usuários, além do envio em larga escala de mensagens de spam a fim de gerar lucro financeiro (BILGE et al., 2011).

O problema abordado por este trabalho afeta desde usuários comuns até empresas, causando os mais diversos danos e perdas.

Há soluções propostas para fases de pré-registro, e pós-registro do domínio. O pré-registo é compreendido entre a compra do domínio e a primeira atualização da zona. O pré-registro é a fase em que o registrante informa seus dados pessoais ou empresariais para a aquisição do domínio, forma de pagamento, endereço (KIDMOSE et al., 2018). O pós-registro pode ser definido como o tempo após a primeira atualização

de zona do DNS (KIDMOSE et al., 2018). No pós-registro, sistemas fazem o uso de coleta ativa e/ou passiva de DNS para geração de uma pontuação para a classificação do domínio.

1.2 Objetivos

O presente trabalho possui como objetivo a detecção de domínios maliciosos por meio do tráfego passivo de DNS utilizando o algoritmo de aprendizagem de máquina supervisionado XGBoost. O modelo classificador deverá obter uma AUC elevada, indicando, assim uma alta probabilidade do modelo estar certo em suas predições.

Os objetivos específicos desenvolvidos durante a execução deste trabalho, são: a) Detecção de domínios maliciosos por meio de classificação binária; b) Utilização de *features* extraídas exclusivamente do tráfego DNS; c) Otimização dos parâmetros do algoritmo XGBoost por meio de otimização Bayesiana.

1.3 Contribuições Obtidas

Com o desenvolvimento do estudo de detecção de domínios maliciosos por meio do tráfego passivo de DNS com XGBoost, obteve-se as seguintes contribuições:

- Desenvolvimento de um método voltado exclusivamente para o tráfego DNS, uma vez que o modelo é capaz de realizar detecção de domínios maliciosos utilizando apenas o tráfego DNS como fonte de dados;
- Modelo proposto utilizando o algoritmo XGBoost na detecção de domínios maliciosos;
- Utilização de técnica de *Undersampling* para balanceamento dos dados presentes no *dataset*;
- Um artigo publicado na *"IEEE International Conference on Intelligence and Security Informatics (ISI)"* intitulado *"Detection of Malicious Domains Using Passive DNS with XGBoost"*;
- Um artigo de título *"XGBoost Applied to Identify MaliciousDomains Using Passive DNS"* e publicado na *"IEEE International Symposium onNetwork Computing and Applications (NCA 2020)"*.

1.4 Organização do Trabalho

O presente trabalho está dividido em cinco capítulos. Neste primeiro capítulo é apresentado a introdução sobre o trabalho e o que este abordará; o Capítulo 2 expõe a fundamentação teórica e o levantamento bibliográfico; no Capítulo 3 é apresentada a metodologia aplicada ao trabalho; no Capítulo 4 são apresentados resultados obtidos e, por fim, no Capítulo 5 são apresentadas as conclusões obtidas a partir deste trabalho.

Capítulo 2

Fundamentação Teórica

Este capítulo tem como finalidade expor os conceitos utilizados neste trabalho, apresentando suas características e sua relevância neste estudo. Na primeira seção 2.1 é apresentado o protocolo DNS, seu funcionamento, abusos e uso indevido podem ocorrer no DNS; na seção 2.2 é apresentado Aprendizado de Máquina, juntamente com os principais algoritmos apresentados em trabalhos correlatos; na seção 2.3 são exibidas as métricas de avaliação que serão utilizadas nesse trabalho; a seção 2.4 apresenta formas de validação do modelo classificador; na seção 2.5 apresenta conceitos sobre balanceamento de classes bem como métodos de balanceamento; na seção 2.6 é apresentado o Intervalo de Confiança e sua importância; a seção 2.7 apresenta o levantamento bibliográfico presente e os trabalhos correlatos, por fim na 2.8 são apresentadas as conclusões parciais deste capítulo.

2.1 Sistema de Nomes de Domínio

O *Domain Name System* (DNS) foi originalmente implementado em meados da década de 1980, como uma forma aprimorada para mapeamento de nome de domínios para endereços IP. O DNS é um banco de dados distribuído que além do mapeamento nome de domínio-IP (e vice-versa), possui outras informações técnicas em seus Resource Records (LYNN, 2001). Essas informações são enviadas como respostas para as consultas realizadas por resolvers. Resolvers são aplicações que possuem acesso ao DNS e realizam estas consultas.

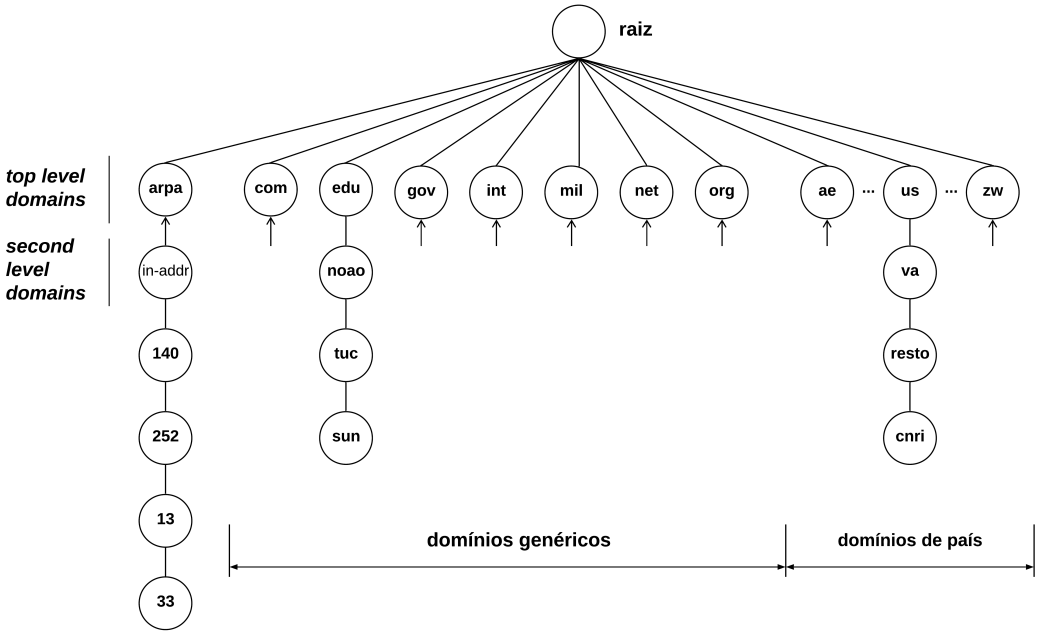
O DNS funciona como uma aplicação na internet fazendo o uso de IPv4 ou IPv6 (ou ambos). Por escalabilidade, o DNS possui uma estrutura hierárquica assim como os servidores que podem executar a resolução dos nomes de domínios. A consulta DNS de forma simplificada funciona da seguinte forma: um cliente executando uma

aplicação DNS envia uma consulta DNS para algum servidor DNS com o nome a ser resolvido. Após a realização da consulta o cliente recebe como resposta o endereço IP associado aquele nome de domínio.

2.1.1 Hierarquia

Há uma hierarquia no espaço DNS, que é insensível a maiúsculas e minúsculas, apresentando, desse modo semelhança com os diretórios e arquivos do sistema de arquivos do computador (FALL; STEVENS, 2011). A Hierarquia do espaço DNS é ilustrada na Figura 2.1

Figura 2.1: Estrutura hierárquica do DNS.



Fonte: Adaptado de (STEVENS, 1994).

No topo da hierarquia encontram-se os *Root Servers*, nos quais existem 13 *Root Servers*, e sendo por eles que a consulta DNS se inicia. No segundo nível da árvore da estrutura hierárquica temos os *Top Level Domains* que são divididos em dois tipos: *Country Code Top Level Domain* (ccTLD), que são designados para países em específico, como `.br` para o Brasil. E os *Generic Top Level Domain* (gTLD), que são como o próprio nome diz, são TLDs genéricos, ou seja, não dependem de posição geográfica.

Após os TLDs estão os *Second Level Domains*, e então os *Third Level Domains* e, assim, sucessivamente, até que forme o nome completo do domínio, intitulado de *Fully Qualified Domain* (FQDN).

Analisando a Figura 2.1, é possível concluir que a formação de FQDN é simples, em que basta apenas percorrer os nós da folha até a raiz, juntando os rótulos e separando-os com um ponto (*Root Servers* não possuem rótulo). Nota-se que na imagem formam-se três FQDNs, sendo ocnri.reston.va.us, sum.tuc.noao.edu e 33.13.253.140.in-addr.arpa, considerando que este último é especial, pois é utilizado para uma consulta reversa, na qual é consultado um número de IP para obter o nome de domínio. Portanto, a consulta apresentada é referente ao endereço 140.252.13.33.

Todos os nós da hierarquia devem, obrigatoriamente, conter ponteiros para os nós abaixo, por exemplo, um *Root Server* deve conter os endereços IP de todos os TLDs, enquanto que o TLD ".com"contém os endereços IP de todos os *Second Level Domains* e assim por diante.

2.1.2 Consultas DNS

Conforme apresentado na seção 2.1.1 há uma estrutura hierárquica para a resolução da consulta DNS. Essa consulta pode ser exemplificada na Figura 2.2.

1. O cliente envia uma mensagem de consulta DNS ao seu servidor recursivo com o nome do domínio a ser traduzido, como por exemplo, o domínio exemplo.acmesecurity.org;

2. O Servidor recursivo transmite esta mensagem de consulta a um dos *Root Servers*;

3. Este identifica o sulfixo .org, e retorna ao servidor recursivo uma lista de endereços IP contendo servidores responsáveis pelo TLD ".org";

4. O recursivo retransmite a mensagem de consulta a um desses servidores TLD;

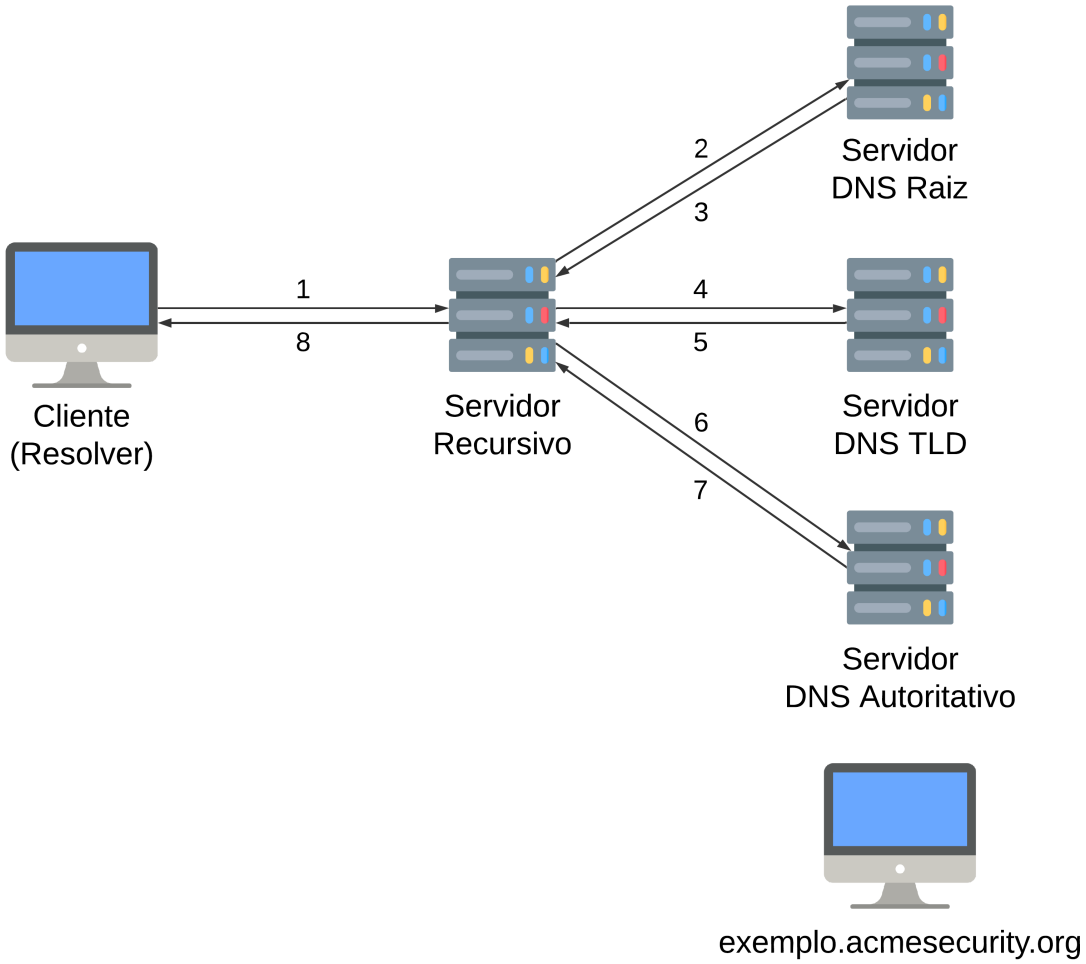
5. O servidor TLD percebe o sufixo "acmesecurity.org"e retorna ao servidor recursivo o endereço IP do servidor DNS autoritativo do laboratório ACME!;

6. Por fim, o servidor DNS recursivo reenvia a mensagem de consulta diretamente ao servidor DNS Autoritativo;

7. O servidor autoritativo responde com o endereço IP de exemplo.acmesecurity.org;

8. O cliente então recebe como resposta do servidor DNS recursivo o IP.

Figura 2.2: Consultas ao sistema de nomes de domínio.



Fonte: Adaptado de (KUROSE; ROSS, 2013)

2.1.3 *Resource Records*

Resource Records (RRs) são registros de recursos que fornecem mapeamentos de nomes de *hosts* para endereços IP e vice-versa. A cada resposta que o servidor DNS atende, na verdade, ele envia um ou mais RRs como resposta. É importante ressaltar que é possível realizar uma consulta DNS solicitando RRs específicos. Kurose e Ross (2013) definem os RRs como uma tupla composta pelos quatro campos seguintes: nome, valor, tipo e tempo de vida (TTL - *time to live*). Os campos nome e valor dependem do tipo do RR, enquanto que o TTL determina quanto tempo os RRs devem ser mantidos em cache por servidores DNS. A seguir, serão apresentados os tipos de RRs necessários para compreensão deste projeto.

***Address* (A) ou (AAAA):** Define o endereço IPv4 ou IPv6 respectivamente de um FQDN, sendo que o campo nome corresponde ao nome do hospedeiro e o campo valor

a seu respectivo endereço IP.

***Canonical Name* (CNAME):** Define um apelido para um FQDN. O campo valor corresponde ao nome canônico do hospedeiro e o campo nome ao apelido desejado.

***Pointer* (PTR):** Este tipo é utilizado para consultas reversas. É semelhante ao A, entretanto define o FQDN relacionado a um endereço IP. O campo nome corresponde ao endereço IP, e o valor ao FQDN.

***Name Server* (NS):** Define um servidor DNS autoritativo para um domínio, ou seja, um servidor responsável para responder por este domínio. Os domínios devem possuir pelo menos um NS. O campo nome corresponde ao domínio e campo valor ao nome do servidor DNS.

***Start of Authority* (SOA):** É um RR especial encontrado no início do arquivo de configuração dos servidores DNS. O campo nome corresponde ao domínio, enquanto que o campo valor possui diversas informações relativas as suas configurações.

2.1.4 Abusos e uso indevido de DNS

Nomes de domínios podem sofrer abusos ao serem registrados e a estrutura do DNS pode ser utilizada de forma mal intencionada de diversas formas. Estes abusos podem implementar, aprimorar, dar suporte e/ou ocultar diversas atividades maliciosas. A seguir, serão apresentados, abusos que ocorrem no DNS e seu uso indevido (TORABI et al.,).

Abuso no Registro de Nome de Domínio: Registrantes vendem a todo momento nomes de domínios e, nesta etapa de aquisição no pré-registro, pode ocorrer o *Cybersquatting* ou *domain squatting*. Pessoas mal intencionadas registram um nome de domínio que representa uma pessoa conhecida ou entidades comerciais, tendo como objetivo principal a geração de lucros com a venda deste domínio para empresa legítima. (Ex: googlechevron.com, googlecoors.com, googledonaldtrump.com e google-gaycruises.com, sites registrados por Chris Gillespie, no período de 29 de fevereiro a 10 de março de 2010 (MCGEE,)). Ainda nesta etapa de pré-registro há o *Typosquartting*, que consiste em registrar nomes de domínio muito semelhantes a domínios bem conhecidos a fim de gerar confusão nas pessoas (Ex: Nome de domínio legítimo americanas.com.br, domínio malicioso americana.com.br). Estes domínios nem sempre são maliciosos, entretanto (ALRWAIS et al., 2014) apresenta que grande parte destes nomes de domínios estavam de fato envolvidos em diferentes tipos de atividades maliciosas e ataques de *phishing*.

Domínios/endereços IP maliciosos ou comprometidos: Atacantes fazem o uso da estrutura aberta do DNS para operar e manter domínios/hosts maliciosos na internet,

a fim de ocultar atividades maliciosas e dificultar sua detecção. Atacantes podem fazer o uso de serviços legítimos para a implementação de atividades maliciosas. O Mirai botnet realizou ataques DDoS por meio de mais de 50.000 IPs pertencentes a dispositivos IoT comprometidos. A seguir, serão apresentados três formas de abuso de DNS por meio de nomes de domínios e endereços IPs maliciosos/comprometidos.

- **Botnets:** Atacantes realizam a instalação de softwares maliciosos (bots) em diversas máquinas. Estes bots normalmente são programados para possuírem capacidade de interação com outros bots para a formação de uma botnet. Um botmaster controla e opera toda botnet via C&C, podendo haver uma diversidade de ataques, incluindo campanhas de spam, aquisição de informações, distribuição de *malwares*, cyber-fraudes, computação distribuída e ataques DDoS. As máquinas comprometidas pela botnet não necessariamente são de propriedade ou totalmente controladas pelo atacante e, portanto, a operação das botnets é sempre vinculada à operação das máquinas nas quais estão instaladas.

- ***Fast Flux Domais e Flux Networks:*** *Domain name flux* foi implementado com boas intenções como *load balancing* e CDN, entretanto, atacantes fazem o uso deste recurso para a ocultação de vestígios de atividades maliciosas, uma vez que o TTL destes domínios normalmente são valores extremamente baixos, resultando assim em um novo domínio para resolução IP no DNS, quase com todas solicitações de domínio. O *Fast-flux hosting* opera sobre redes distribuídas bem estabelecidas de sistemas maliciosos ou comprometidos (ex: botnets), que fornecem recursos para hospedagem de *fast flux* ou serviço de nomes. Os atacantes utilizam essas *Flux Networks* para a realização de ataques distribuídos na internet. Em casos mais sofisticados de fluxo de domínio, os invasores complementam a rede de serviços que hospeda sites com uma segunda rede de serviços que, por sua vez hospeda servidores (flux duplo). Esse tipo de hospedagem *fast-flow* oculta o local do provedor de conteúdo malicioso (ex: bot) por trás de um nível extra de resoluções de DNS que não são vistos pelos resolvedores de DNS.

- **Malicious Domain Generation Algorithms:** DGAs são utilizados para geração automática de domínios em diferentes TLDs para que possa ter uma distribuição global por meio da internet. Podendo ser utilizados por *botnets*, C&C, *fast flux domain* e DDoS. Botnets se beneficiam de DGAs, tornando sua estrutura resiliente à *blocklists* de domínios e derrubando os esforços registrando diversos domínios e alterando entre eles. O Conficker malware gerou 50.000 nomes de

domínios aleatórios que foram espalhados por 113 TLDs.

Serviços Benignos e Aplicações: O DNS possui uma estrutura aberta, com isso proporcionou que indivíduos mal intencionados fizessem o uso destes servidos para fins maliciosos, como número de *resolvers* de DNS abertos que apresentaram discrepâncias nas resoluções DNS, as quais foram implementadas maliciosamente para censurar canais de comunicação, injetar anúncios, transferir arquivos maliciosos ou até a realização de *phishing*. Atacantes podem fazer o uso de ferramentas de varreduras para obtenção de informações sobre servidores e registros DNS, identificando vulnerabilidades que poderiam ser utilizadas para realização de ataques direcionados. Os atacantes podem fazer o uso de serviços e aplicativos legítimos para ocultar seus rastros evitando, assim sua detecção.

2.1.5 DNS Passivo

O DNS Passivo foi proposto pela primeira vez por Weimer (2005), cujo objetivo era deter ataques de malwares. Além disso, fazia o uso de servidores de nomes de domínios recursivos para registrar respostas recebidas de diferentes servidores de nomes. O DNS Passivo é a coleta da comunicação entre os servidores DNS, que é realizada por meio de sensores instalados com acessos a servidores DNS para obtenção de consultas e repostas reais do DNS. O tráfego DNS contém uma quantidade significativa de recursos para identificação de nomes de domínios associados a atividades maliciosas, além de poder haver um enriquecimento destes dados com informações associadas a estes, como o número de Sistemas autônomos (AS), Whois, geolocalização, dentre outras.

Diante do exposto, há diversos locais para se realizar a coleta deste DNS passivo, que pode ser feita através de sensores instalados em *Internet Service Provider* (ISP), ou em servidores autoritativos de TLDs.

2.2 Aprendizado de Máquina

De acordo com (WEISS, 1992), cientista da computação pioneiro no assunto, definiu em 1959 o aprendizado de máquina como "campo de estudo que dá aos computadores a habilidade de aprender sem serem explicitamente programados". Outra definição presente no livro *Foundations of Machine Learning* "métodos computacionais usando a experiência para melhorar o desempenho ou para fazer previsões precisas"(M Mohri, A Rostamizadeh, 2012).

O aprendizado de máquina está presente nos mais diversos problemas e setores da atualidade, como uma forma de predição, classificação, compreensão dos sentimentos e identificação de tendências. Esta área pode ser aplicada a uma vasta gama de problemas, na busca de soluções com mais precisão a partir de conhecimentos prévios que obtemos.

Há basicamente quatro tipos de técnicas de aprendizado de máquina, sendo elas: o aprendizado de máquina supervisionado, aprendizado não supervisionado, semi-supervisionado e o aprendizado por reforço.

Algoritmos de aprendizado supervisionados fazem o uso de conjuntos de dados previamente rotulados como dados de treinamento e, a partir deste aprendizado, é capaz de realizar previsões para outros dados não conhecidos. O cenário mais comum são problemas de classificação e regressão.

Algoritmos de aprendizado não supervisionados fazem o uso de conjunto de dados não rotulados para treinamento e, a partir deste treinamento realizado, é capaz de prever para todos dados desconhecidos. Como normalmente nenhum dado de treinamento está rotulado neste cenário, pode ser difícil a avaliação quantitativa do desempenho de um algoritmo. Exemplos de uso desta forma de aplicação de aprendizado de máquina é para redução de dimensionalidade e agrupamento.

Algoritmos de aprendizado semi-supervisionados são capazes de aprender, utilizando uma pequena parte dos dados rotulados, e uma grande parte não rotulado. Normalmente algoritmos que utilizam essa técnica de aprendizado são resultados de combinações de algoritmos supervisionados e não supervisionados.

Na aprendizagem por reforço, a máquina tenta aprender qual melhor ação a ser tomada a partir de determinada situação, podendo ganhar recompensa ou punições. A cada decisão correta tomada, é gerada uma recompensa e para cada decisão errada, uma punição.

2.2.1 Aprendizado Supervisionado

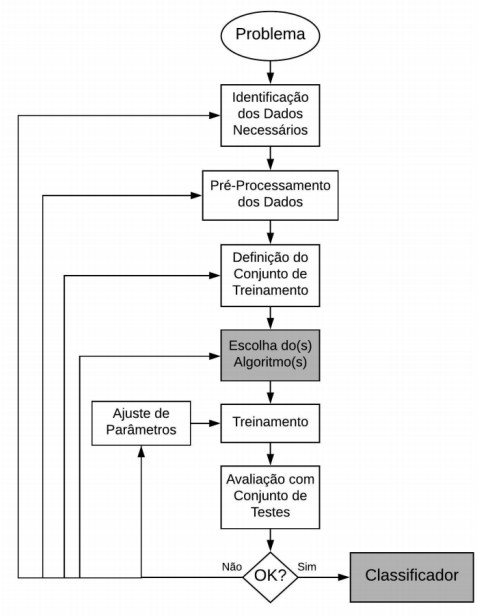
Normalmente, para classificação de nomes de domínio, faz-se o uso de algoritmos de aprendizado de máquina supervisionados, pois o objetivo é treinar o classificador com duas listas de tráfego de domínios, uma com tráfego malicioso, e outra com tráfego legítimo apenas, resultando, assim em dois possíveis grupos "legítimo"ou "malicioso".

A partir dos dados de treinamento, o classificador irá montar seu modelo do que é malicioso e do que é legítimo, sendo possível, então, a classificação do nome de domínio por seu tráfego DNS passivo. Uma observação importante em relação aos dados

de treinamento é que estes precisam ter a mesma quantidade, para que o classificador funcione de maneira imparcial.

Kotsiantis, Zaharakis e Pintelas (2007) definem o aprendizado de máquina supervisionado, como sendo o processo de aprender um conjunto de regras a partir de instância, no caso os dados de treinamento. Ou ainda de forma mais geral, o processo de criar um classificador que pode se generalizar a partir de instâncias. Os autores apresentam também um fluxograma ilustrado na Figura 2.3, no qual é descrito o processo de aplicação de aprendizado de máquina supervisionado para um problema real.

Figura 2.3: Aprendizado de Máquina Supervisionado.



Fonte: Adaptado de (KOTSIANTIS; ZAHARAKIS; PINTELAS, 2007)

Um dado relevante em relação a algoritmos de aprendizado supervisionado é a engenharia de *features*, nos quais, esta etapa consiste em definir atributos relevantes nos dados que serão analisados. É importante ressaltar que estas características retiradas dos dados de entrada, obrigatoriamente devem ser as mesmas dos dados de treinamento.

Após a etapa de definição dos atributos, é necessário a definição dos dados para o treinamento, os quais devem estar rotulados e sua representação é uma matriz, cujas linhas representam dados de entrada, as colunas representam os atributos selecionados,

e a última coluna a rotulação destes dados de entrada.

Após a análise dos dados pelo algoritmo escolhido em sua fase de treinamento, este está pronto para a classificação de dados desconhecidos.

Algoritmos diferentes fazem o uso de abordagens diferentes para aprender, portanto, os resultados obtidos na classificação sofrem interferência do algoritmo que fora utilizado. O desempenho dos algoritmos sofrem alterações com os mais variados fatores e o escopo no qual foi aplicado. Portanto, fazer a escolha do algoritmo correto para o problema que será enfrentado é de extrema importância para que se obtenha excelentes resultados.

2.2.2 Árvores de Decisão

Árvores de decisão (DT - *Decision Tree*) fazem o uso da estratégia "dividir para conquistar", considerando que este algoritmo, pega um problema complexo e o divide em problemas menores e mais simples, recursivamente até encontrar a solução para o problema. As soluções destes subproblemas resolvidos podem ser combinados, a fim de que produza a solução de um problema complexo (CARVALHO et al., 2011).

Na Figura 2.4 é apresentado um exemplo de árvore de decisão para uma empresa que realiza entrega apenas para seu estado, e que fornece frete grátis se for percorrido menos de 100 quilômetros de distância.

Figura 2.4: Exemplo Árvore de Decisão.

Fonte: Elaborado pelo autor, 2020.

Cada nó da árvore corresponde a uma característica dos dados, cada ramificação

a um valor que o nó pode assumir, e em cada folha há um rótulo cuja entrada pode ser classificada. Portanto, para que este realize a classificação, os algoritmos percorrem a árvore a partir da raiz, realizando a análise das características de uma instância qualquer até encontrar um nó folha, que corresponde ao rótulo da mesma.

Safavian e Landgrebe (1991) discutem com profundidade o funcionamento de classificador baseado neste algoritmo. Um ponto crítico relatado no estudo é a construção da árvore propriamente dita, pois o algoritmo realiza a análise das amostras de dados de treinamento para criar as relações entre as características dos dados e seus respectivos rótulos, portanto, para que se obtenha um classificador que tenha uma boa eficiência, se faz necessário um levantamento criterioso de características.

2.2.3 Floresta Aleatória

De acordo com Donges (2018), Floresta Aleatória (RF) *(Random Forest)* é um algoritmo de aprendizado de máquina flexível e fácil de usar, que, na maioria das vezes, produz excelentes resultados, mesmo sem ajuste de hiperparâmetros.

Este algoritmo de aprendizado de máquina supervisionado gera uma floresta de modo aleatório, realizando a combinação de árvores de decisão (floresta) para a obtenção de uma predição com maior acurácia e mais estável.

O algoritmo insere a aleatoriedade quando está criando as árvores. Em vez de realizar busca pela melhor característica ao fazer a partição dos nodos, ele faz a busca da melhor característica em um subconjunto aleatório das características. Este processo resulta em uma grande diversidade, o que normalmente leva a geração de modelos melhores.

Para produzir uma resposta final de classificação referente a uma amostra não rotulada, o algoritmo a fornece como entrada para todas árvores contidas na floresta e, a partir de suas respectivas saídas, calcula a média entre elas obtendo o resultado.

Por meio desta abordagem, é agregada uma diversidade mais ampla neste processo de classificação, gerando um resultado mais satisfatório tanto por parte do modelo, como por parte das predições.

2.2.4 Extreme Gradient Boosting (XGBoost)

É um algoritmo de aprendizado de máquina apresentado por Chen e Guestrin (2016). Trata-se de um sistema escalonável de aumento de árvores ponta-a-ponta chamado *Extreme Gradient Boosting* (XGBoost), obtendo bons resultados em muitos desafios de aprendizado de máquina. Entretanto, quando utilizado com dados estruturados/tabu-

lares de pequeno a médio porte, os algoritmos baseados em árvore de decisão são considerados os melhores da categoria no momento.

XGBoost implementa um processo de *Boosting* para a geração de modelos precisos. Ele é um algoritmo que impulsiona a árvore de decisão, em que impulsionar faz referência à técnica de aprendizado de conjunto muitos modelos de forma sequencial. A cada novo modelo que este algoritmo gera, tenta corrigir as deficiências do modelo anterior.

O fator de maior relevância do sucesso do XGBoost é sua escalabilidade em todos cenários, o que permite, desse modo, sua execução desde uma máquina comum até configurações distribuídas para bilhões de exemplos. A escalabilidade apresentada é possibilitada por otimizações algorítmicas e vários sistemas importantes. Inovações como um novo algoritmo de aprendizado em árvore; procedimento de esboço quantil ponderado teoricamente justificado possibilita lidar com pesos de instância no aprendizado aproximado das árvores. A computação paralela e distribuída permite o aprendizado mais rápido, possibilitando assim, uma exploração mais rápida do modelo. Outro fator explorado por este algoritmo de aprendizado de máquina é a computação fora do núcleo, o que permite o processamento de centenas de milhões de exemplos em um desktop. XGBoost possui consigo algumas técnicas para evitar *overfitting*, sendo delas:

- **Objetivo de aprendizagem regularizado:** Esta técnica ajuda na suavização dos pesos finais aprendidos, evitando assim o *overfitting*. Intuitivamente, o objetivo regularizado tenderá a selecionar um modelo que emprega funções simples e preditivas.

- ***Shrinkage:*** Este encolhimento reduz a influência de cada árvore individual e deixa espaço para futuras árvores melhorarem o modelo.

- ***Column Subsampling:*** Ao fazer o uso desta técnica, previne *overfitting*, além de um ganho na velocidade de processamento dos cálculos na versão paralelizada do algoritmo.

Para obtenção de uma melhor performance do XGBoost, é necessário o ajuste de hiperparâmetros. Dentre os hiperparâmetros presentes no algoritmo XGBoost estão:

- ***learning_rate:*** A taxa de aprendizagem definida neste hiperarâmetro define o tamanho do passo usado na atualização para evitar *overfitting*. Após cada etapa de *bosting*, obtém diretamente os pesos das novas *features*.

- ***n_estimators:*** Número de árvores com aumento de gradiente. Equivalente ao número de rodadas de reforço.
- ***max_depth:*** Profundidade máxima da árvore. Quanto maior este valor, maior complexo o modelo se torna e com maior probabilidade de *overfit*.
- ***subsample:*** Proporção de subamostra das instâncias de treinamento.
- ***gamma:*** Redução de perda mínima necessária para fazer uma partição em um nó folha da árvore. Quanto maior for o valor de gamma, mais conservador será o algoritmo.
- ***reg_lambda:*** Termo de regularização L2 sobre os pesos. Aumentar o valor deste hiperparâmetro torna o modelo mais conservador.

2.3 Métricas de Avaliação

Nesta sessão serão apresentadas as métricas utilizadas para a avaliação dos modelos classificadores.

Antes se faz necessário a definição de P e N, onde P é número de domínios maliciosos e N o número de domínios legítimos. Verdadeiros Positivos (VP) e Verdadeiro Negativo (VN), são os números de domínios maliciosos e legítimos corretamente identificados. Essas métricas são complementares aos Falsos Positivos (FP) e Falsos Negativos (FN), em que FP é o número de domínios legítimos que foram incorretamente identificados como maliciosos, e FN é número de domínios maliciosos que foram incorretamente identificados como legítimos. Ambas as métricas citadas acima, são utilizadas para métricas mais elaboradas, como taxa de verdadeiro positivo, taxa de verdadeiro negativo, taxa de falso positivo, taxa de falso negativo, precisão, F1-score e área sobre a curva ROC.

Uma das métricas mais utilizadas para avaliação de modelos classificadores binários é a Área Sob a Curva ROC (AUC) portanto, ela foi a escolhida para avaliação e comparação de performance dos modelos construídos. Abaixo será apresentada a métrica.

A AUC é obtida por meio dos gráficos da Receiver Operating Characteristics (ROC). Esta é uma técnica muito útil na visualização de desempenho dos classificadores, e tem sido cada vez mais adotada nas comunidades de de aprendizagem de máquina (FAWCETT, 2004).

A Curva ROC é obtida por meio de um gráfico contendo duas dimensões das quais no eixo X é representada a Taxa de Falsos Positivos (TFP), no eixo Y é representada a Taxa de Verdadeiros Positivos (TVN).

A Taxa de Falso Positivo (TFP) é definida pela Equação 2.1 ou por 1-TVN e representa a taxa de amostras positivas que foram erroneamente classificadas.

$$TFP = \frac{FP}{TN + FP} \tag{2.1}$$

Taxa de Falso Negativos (TFN) é definida pela equação 2.2, ou 1-TFP e representa a taxa de amostras negativas que foram erroneamente classificadas.

$$TFN = \frac{FN}{TP + FN} \tag{2.2}$$

A Taxa de Verdadeiro Negativo (TVN), é definida pela Equação 2.3, também conhecida como especificidade (E), representa as amostras negativas que foram corretamente classificadas.

$$E = \frac{TN}{TN + FP} \tag{2.3}$$

Por sua vez, a Taxa de Verdadeiro Positivo (TVP), conhecida por *recall*, é definida pela Equação 2.4 e representa a taxa de amostras positivas que foram classificadas corretamente.

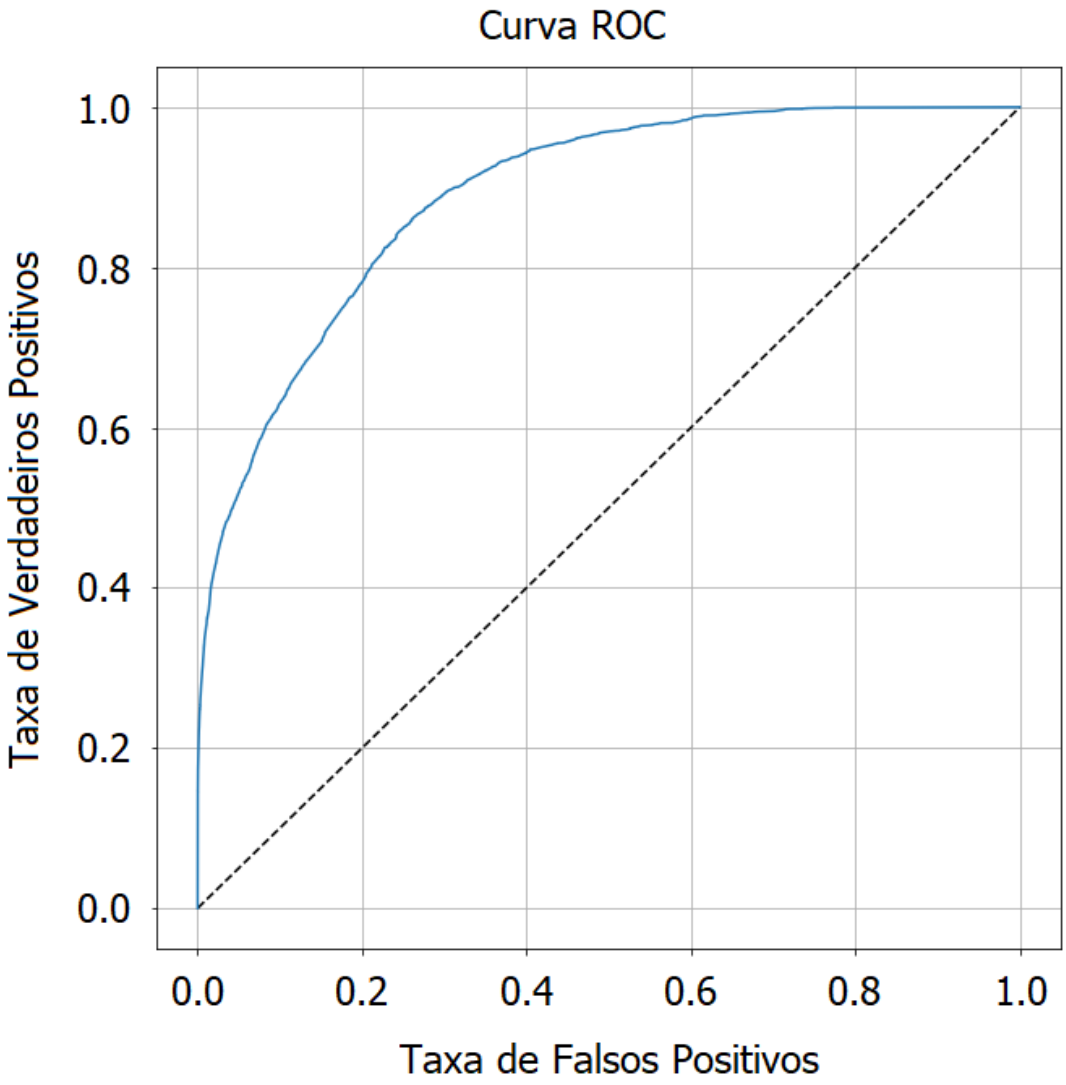
$$recall = \frac{TP}{TP + FN} \tag{2.4}$$

Na Figura 2.5 é representada uma Curva ROC qualquer, na qual é possível observar uma linha diagonal pontilhada. Esta linha representa um classificador que se acerta 50% de suas predições, sendo assim um classificador aleatório.

Uma forma de visualizar os resultados da curva ROC mais facilmente é usar a AUC, na qual se calcula o valor da área sob a curva ROC, resultando em um número, e tornando, assim, a comparação de desempenho entre modelos mais fácil. A AUC está associada às previsões do modelo. Outra interpretação possível sobre a Área da Curva ROC é probabilidade de o modelo classificar uma instância positiva selecionada acima de uma instância negativa selecionada aleatoriamente.

Apesar da AUC ser uma boa métrica para avaliação do desempenho como discutido anteriormente, não se deve utiliza-lá como métrica, caso os dados estejam muito desbalanceados. Assim, intuitivamente, a taxa de falsos positivos para conjuntos de dados altamente desequilibrados é reduzida devido a um grande número de verdadeiros negativos. Portanto, é recomendado a utilização quando se tem um conjunto de

Figura 2.5: Exemplo de Curva ROC.



Fonte: Elaborado pelo autor, 2020.

dados balanceados, e quando há preocupação com ambas as classes (positiva e negativa).

Portanto, a partir da métrica AUC, apresentada anteriormente, é possível analisar o desempenho dos modelos classificadores desenvolvidos neste trabalho, possibilitando, assim, a qualidade da abordagem apresentada.

2.4 Validação do Modelo

Majoritariamente há duas técnicas para treino e teste do modelo, sendo a primeira e mais simples a técnica Hold-out. Nesta técnica divide-se o conjunto de dados entre

treino e teste, normalmente as proporções são 80% para etapa de treinamento do modelo e 20% para testes, ou até mesmo 70% para treino e 30% para testes. O modelo é ajustado aos dados de treinamento, no qual o modelo tem acesso a sua rotulação, e a partir dele, tenta-se fazer as predições (RASCHKA, 2018).

A segunda técnica e provavelmente a mais comum utilizada para avaliação seleção de modelos é a K-fold Cross-validation (RASCHKA, 2018), em que a ideia principal por trás deste método de validação cruzada é que cada amostra do conjunto de dados tenha a oportunidade de ser testada. Portanto, K-fold Cross Validation é um caso de validação cruzada no qual é iterado sobre um conjunto de dados K vezes. A cada rodada executada, divide-se o conjunto em partes, em que uma parte é utilizada para validação do modelo e o restante (K-1) é utilizado na etapa de treinamento, assim como é ilustrado na Figura 2.6. Para o exemplo apresentado na Figura 2.6, definiu-se o valor de K sendo 5.

Uma extensão de *cross-validation* é o *stratified cross-validation* (BREIMAN JEROME FRIEDMAN, 1984). No *stratified cross-validation*, o *dataset* é repartido em n *folds*, e cada classe presente é distribuída uniformemente por esses *folds*, o que resulta em *folds* similares ao *dataset*, mantendo, assim sua proporcionalidade de classes. No *cross-validation* regular, os *folds* gerados não possuem esse "balanceamento" para manter proporcionalidade, e sua distribuição é selecionada aleatoriamente pelos dados que compõem o *dataset*, ou seja, alguns *folds* podem conter mais registros de uma determinada classe do que outros.

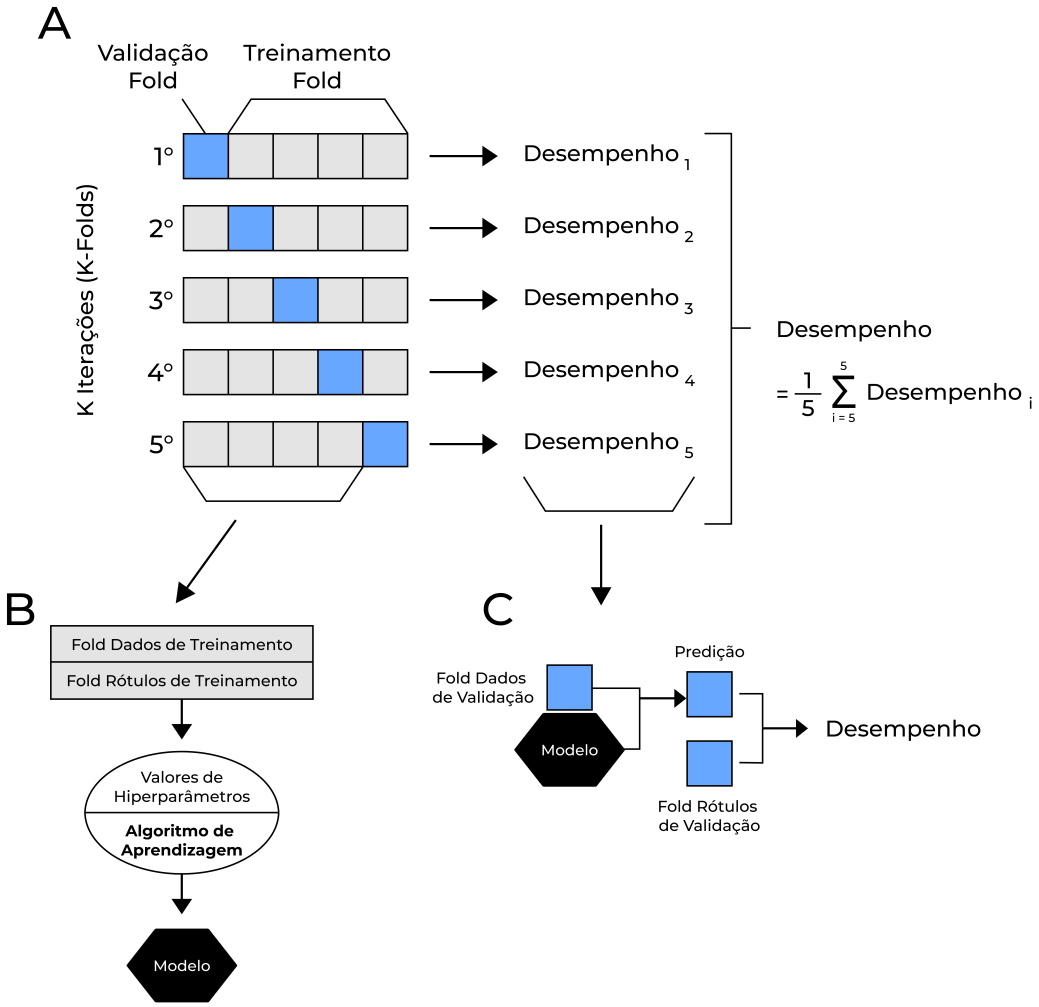
2.5 Balanceamento de Classes

Quando o conjunto de dados a ser trabalhado é extremamente desbalanceado, isto é, em que há uma grande quantidade de dados legítimos e uma pequena porção de maliciosos, por exemplo, faz-se necessário a realização de um balanceamento das classes. Há três formas de realizar este balanceamento, sendo divididos em sobreamostragem, subamostragem e métodos híbridos (FERNÁNDEZ et al., 2018).

O balanceamento de classes com métodos de sobreamostragem é realizado a partir da geração sintética de dados da classe minoritária, ou até mesmo a partir da réplica destes dados da classe minoritária, resultando, assim, em um superconjunto do conjunto de dados original.

Os métodos de balanceamento de classes por subamostragem resultam em um subconjunto do conjunto de dados original e sua atuação é na eliminação de dados da classe majoritária obtendo, assim, um balanceamento entre as classes.

Figura 2.6: K-fold Cross Validation.



Fonte: Adaptado de (RASCHKA, 2018).

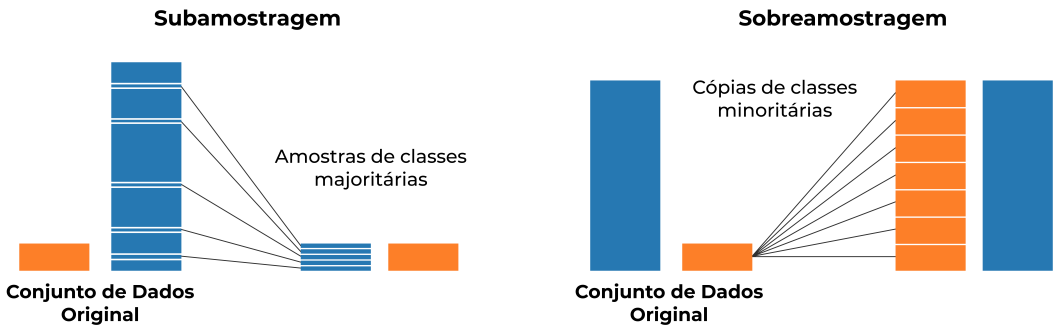
Por fim, há o balanceamento dos dados com métodos híbridos, nos quais combinam as duas abordagens anteriores apresentadas.

O método de sobreamostragem e subamostragem, estão ilustrados na Figura 2.7.

Balanceamento de classes utilizando subamostragem pode resultar em perda de informações importantes para o aprendizado do modelo classificador, uma vez que dados da classe majoritária são removidos. Entretanto, a utilização de técnicas de sobreamostragens pode facilmente resultar em *overfitting* do modelo, pois dados sintéticos são gerados a partir dos originais, ou até mesmo replicados.

Neste trabalho optou-se pela técnica de balanceamento de dados Random Undersampling (RUS), a qual faz o uso do método de Subamostragem, em que seleciona os dados da classe majoritária aleatoriamente. Como discutido anteriormente, para

Figura 2.7: Sobreamostragem e subamostragem.



Fonte: Adaptado de (VAZ, 2019).

realizar a prevenção de possíveis perda de dados, experimentos foram realizados e discutidos na seção 4.2.

2.6 Intervalo de Confiança

O intervalo de confiança permite prever o valor aproximado de um conjunto de dados amostrados. Assim, fornece um intervalo de valores que contém um parâmetro do conjunto de dados em estudo, com determinado grau de probabilidade, como por exemplo, a média dos dados. O grau de probabilidade é conhecido como grau ou nível de confiança, simbolizado por $\gamma = 1 - \alpha$, em que α é o nível de significância, isto é, probabilidade de erro. Ademais, sempre tem um nível de cobertura, expressado como uma alta porcentagem, isto é, comumente $\gamma = 95\%$ (MONTGOMERY, 2002). Quanto maior o grau de confiança, maior o intervalo. Além disso, quanto menor a amostra, maior o intervalo, isto é, maior a incerteza. Em outras palavras, quanto mais confiante pretende ser, e menos dados dispor, maior deve ser o intervalo de confiança para ser suficientemente assegurada a captura do valor real (BRUCE et al., 2020).

Na Equação 2.5 é apresentado como é realizado o cálculo do intervalo de confiança para a média, em que o grau de confiança é de 95%.

$$IC_{95} = \overline{X} \pm t_{95} \times \frac{\sigma}{\sqrt{n}} \tag{2.5}$$

Na Equação 2.5 tem-se que: X é a média dos dados, t_{95} é um valor constante calculado para um tipo específico de distribuição, que se denomina distribuição t, é o desvio padrão e n é o número de elementos amostrados. Observa-se que para um grau de confiança de 95%, ou seja, o que é o mais abordado na literatura, t é 1,96.

Sobre a distribuição *t*, verifica-se que é similar com a distribuição normal, contudo tem influência do seu formato pelo tamanho da amostra em questão. Além disso, a distribuição *t* conta como uma lista de valores pré-determinados para cálculos que a abrangem, tal como a distribuição normal (URDAN, 2010).

2.7 Levantamento Bibliográfico

Diversos estudos foram realizados sobre a classificação de domínios baseados em DNS passivo, os quais distinguem-se por diversos fatores, como o algoritmo utilizado, a base de dados DNS passivo, qual a localidade de coleta deste tráfego DNS, podendo ser dos recursivos ou autoritativos. Todas estas particulares são importantes para que se tenha uma visão dos objetivos diferentes, para que o foco do trabalho possa abranger diferentes frentes.

Os dados podem vir de fontes diversas, bem como quais dados foram utilizados para fazer um enriquecimento maior, a fim de se obter uma melhoria na predição do classificador. A seguir serão apresentados os trabalhos correlatos.

Antonakakis et al. (2010) propuseram o Notos, que foi o primeiro sistema de reputação dinâmica projeto para DNS em 2010. Ele utiliza o número de *features* para computar o valor da reputação de um nome de domínio.

Suas *features* são Network-based features, *features* baseadas em zonas e *features* baseadas em evidências, em que a primeira faz o uso de endereços de IPs historicamente associados a um domínio, diversidade da localização geográfica e o número distinto de ASs.

Em geral, o Notos calcula uma pontuação de reputação para decidir se um nome de domínios tem características semelhantes ao nome de domínios legítimos ou mal intencionados. Para gerar esta pontuação de reputação, há um conjunto de dados de treinamento de DNS passivo, utilizado em módulo offline para a criação de três módulos diferentes, sendo o primeiro módulo a criação de um perfil de rede, constituído por diversas classes de domínios benignos. O segundo são domínios agrupados conforme seus recursos compartilhados baseados em rede e em zona (IP, e TLDs associados). E o terceiro módulo é baseado em evidências, em que é criado para identificar domínios maliciosos usando amostras de malware analisadas e nomes de domínios na *blocklist*. Há o treino em que o algoritmo é executado com dados rotulados a fim de aprendizado deste e, após a fase de treino, Notos é executado online. Dado qualquer novo nome de domínio, o mecanismo de reputação calculará um vetor de recurso que será usado para alimentar a função de reputação, a fim de correlacionar o novo domínio com os domí-

nios rotulados anteriormente do conjunto de dados de treinamento. A saída será uma pontuação de reputação, que indica as semelhanças entre os recursos do novo domínio e os domínios rotulados anteriormente.

Bilge et al. (2011) propuseram o Exposure, que é um sistema projetado para detecção de nomes de domínios maliciosos fazendo o uso de DNS passivo, coletado por meio de um servidor DNS recursivo para a extração de *features* relacionados aos domínios maliciosos e benignos conhecidos. Estas *features* são usadas para realizar a etapa de treino do classificador que, por sua vez, detecta atividades maliciosas no domínio em tempo hábil. O Exposure foi projetado para realizar a detecção de domínios relacionados a spam e malware, entretanto, ele também detecta domínios maliciosos de Fast Flux e domínios relacionados a DGAs baseados em suas *features*. Neste sentido, é realizado uma anonimização dos dados e a computação das *features*.

Análise de tráfego passivo de DNS, relacionado a domínios maliciosos/legítimos bem conhecidos, o Exposure realiza a extração de *features* para a identificação de novos domínios legítimos e maliciosos. O sistema possui duas fases, sendo a primeira de treinamento e a segunda de detecção e retreinamento. A validação do Exposure ocorreu após a execução deste sistema offline por dois meses e meio. Foi implantado na rede de um ISP, em que houve o fornecimento completo de acesso ao servidor DNS por duas semanas, tendo aproximadamente 30.000 clientes. Exposure utilizou os primeiros 7 dias para treinamento inicial e um retreinamento diário.

Flux Buster é um sistema proposto por Perdisci, Corona e Giacinto (2012), projetado para detecção de redes fluxo maliciosos. O sistema faz uso de dados DNS passivo, coletado de diversos servidores DNS recursivos de todo o mundo.

As redes de fluxo fazem o uso de botnets e diversos domínios de fluxo rápido para a execução de atividades maliciosas. Um bootmaster frequentemente altera os agentes de fluxo (endereços IP resolvidos) para os quais os nomes de domínios mal intencionados resolvem, afim de ocultar seus rastreamentos e evitar a identificação pelas autoridades. Portanto, assume-se que os domínios de fluxo fazem a especificação de um TTL muito curto, além disso a frequência com que estes domínios realizam a alteração no conjunto de endereços IP resolvidos e o número total de endereços IP resolvidos ao longo do tempo se torne muito maior, se comparado a domínios de fluxo com domínios legítimos. Considerando as suposições feitas anteriormente, o FluxBuster realiza uma pré-filtragem dos dados para remoção de domínios legítimos. Em seguida, utiliza um algoritmo de clusterização para agrupar domínios que resolvem conjuntos semelhantes de endereços IP. Após isso, o sistema calcula um número de *features* estatísticas que serão usadas para classificar redes de fluxo.

FlaxBluster usa DNS passivo para analisar e detectar fluxos, o sistema foi treinado com dados de 4 meses de coleta passiva de DNS, tendo que ser rotulado manualmente os clusters de domínios por meio da análise de suas características e identificando os domínios e fluxo e não-fluxo. A rotulagem automática foi usada sempre que possível para incorporação de informações disponíveis sobre os domínios maliciosos conhecidos.

Antonakakis et al. (2011), em seu estudo, propõem o Kopis. Kopis é um sistema de detecção de nomes de domínios relacionados a malware por meio de monitoramento de DNS passivo na hierarquia superior do DNS e identificando padrões de consulta/-resposta mal-intencionados. Este sistema monitora o tráfego em TLDS e em servidores autoritativos, introduzindo, assim, novos recursos de tráfego que aproveitam a visibilidade global. A análise do fluxo de consultas e respostas DNS é realizada através de épocas, considerando que cada época significa um dia de monitoramento deste tráfego. Em cada época, o Kopis realiza diversos cálculos estatísticos para cada nome de domínio, sendo esses Requerster Diversity (RD), Requester Profile (RP), Resolved-IPs Reputation (IPR).

Kopis faz o uso de um sistema de aprendizado supervisionado a partir de um conjunto de domínios conhecidos legítimos e relacionados a malwares, como dados de treinamento. Então, é criado um módulo de classificação estatística que pode prever se um novo domínio está associado a malware com base dos padrões de resolução observados. Para a fase inicial coletou-se 8 meses de tráfego passivo DNS de dois servidores de DNS autoritativos e do TLD .ca. O módulo de aprendizagem foi treinado por meio de recursos estatísticos para cada domínio e rotulando estes com base em listas extensas de domínios previamente conhecidos relacionados a malwares e *blocklists*. Foi utilizado também uma lista de domínios legítimos identificados a partir de diferentes recursos para a rotularização de domínios benignos no módulo de aprendizagem. Para cada nome de domínio solicitado, o Kopis usa os recursos extraídos e correlaciona os resultados com a verdade do solo rotulada usando o módulo classificador Random Forest (RF). O domínio é classificado em benigno ou relacionado à malware.

Lison e Mavroeidis (2017) propõem *"Neural Reputation Models learned from Passive DNS Data"*, que apresenta o uso de abordagem neural para a identificação de domínios maliciosos e legítimos por meio do DNS passivo, fazendo, desse modo o uso de uma Rede Neural Recorrente, e de uma RNN feed forward.

Os autores utilizaram listas legítimas como Alexa, Statvoo e Cisco para a rotularização de domínios legítimos e de listas como maltrail, Mnemonic e DGArchive para

a rotulação de domínios contidos em *blocklists*. Para considerar a reputação dos IPs, os autores consideraram a sua geolocalização, por meio de tabelas que relacionam IPs a países, como a GeoName databases, IP geolactions. Além disso, foi utilizado grafos bipartidos, em que analisaram, se um nome de domínio legítimo estava relacionado a um endereço IP legítimo, e se outro nome de domínio desconhecido estava ligado a este mesmo IP, portanto, há chances deste nome de domínio desconhecido ser legítimo.

Foram usadas 3 tipos de *features* neste trabalho, as numéricas, categóricas e a sequência. Além disso, foram utilizadas uma camada de entrada, duas camadas densas, e uma camada de saída, em que a *feature* sequencial (nome do domínio) passa por uma RNN LSTM para análise da probabilidade do domínio ser gerado por um malware, e após isso este valor entra na camada densa 1. A *feature* Categórica tem um pré-processamento antes da entrada na camada densa 1. E as *features* numéricas vão direto da camada de entrada para a camada densa 1. As Camadas densas 1 e 2 são RNN feed forward.

Com o processamento nesta rede neural obtém-se a seguinte classificação, domínio legítimo, malicioso ou sinkhole.

O trabalho se mostrou eficiente com a melhor performance do modelo com três camadas ocultas, com cada uma de 1024 de tamanho, com entrada de todas *features* e dois passos de aprendizagem.

Tran et al. (2019) propõem o *"Multi-Confirmations and DNS Graph Mining for Malicious Domain Detection"*, que faz uso de uma combinação de mineração de grafos juntamente com uma abordagem estatística.

Os autores do *paper* fizeram o uso do DNS Passivo, entretanto, utilizaram apenas três campos, sendo eles o nome de domínio, o IP de resposta do host, e o endereço de IP do cliente, nos quais os domínios e os IPs de resposta foram utilizados para a mineração de grafos. Os domínios e os IPs dos clientes foram utilizados para calcular a probabilidade maliciosa do domínio pela lei do método da probabilidade total. Como *allowlist* os autores utilizaram a lista da Alexa, e como *blocklists* utilizaram Malware Domains, URL blocklist, o feed de DGA da Bambernek Consulting, alguns domínios maliciosos e geração de malware, além de alguns domínios maliciosos coletados dos autores dos trabalhos com trabalhos relacionados.

O trabalho mostrou-se uma forma aprimorada de detecção de domínios maliciosos, com melhorias na detecção, se comparado ao trabalho anteriormente feito pelos autores, com bons índices de acurácia, precisão e recall.

Bao, Wang e Lan (2019) apresentam uma abordagem para detecção de domínios maliciosos relacionados a DGAs e a detecção de domínios pornográficos com base no

vetor de palavras em combinação com o ambiente de rede chinês.

Os autores coletaram DNS Passivo por três dia e fizeram o uso da técnica de Downsampling para a redução do desbalanceamento das classes, uma vez que, a proporção inicial era 1:20, por meio da técnica reduziram para 1:5 a desproporcionalidade. Foram utilizadas *features* tanto de acesso, quanto léxicas. Como métrica de avaliação os autores utilizaram a AUC e para classificação utilizaram o algoritmo XGBoost.

Os autores construíram, então, dois classificadores, um para detecção de DGAs, no qual utilizando todas *features* obtiveram uma AUC de 0,994. Considerando o desempenho do classificador, utilizando apenas *features* de acesso, este desempenho é reduzido para uma AUC de 0.974 e o último teste realizado na detecção destes domínios relacionados a DGA utilizando apenas *features* de caracteres, o desempenho foi uma AUC de 0.966

O desempenho do modelo classificador utilizado, para detecção de domínios relacionados à pornografia nos dois testes realizados, obteve como resultado uma AUC de 0.859 e 0.749.

Wang et al. (2020) propõem um método de detecção de domínios maliciosos utilizando uma combinação de DNS passivo e ativo para realizar a detecção de domínios maliciosos.

Os autores precisaram trabalhar com classes desbalanceadas, e, portanto, propuseram como forma de tratar o desbalanceamento dos dados a utilização do algoritmo K-Means juntamente com SMOTE, que foi chamado pelos autores de KMSMOTE, resultando em um *resampling* dos dados por meio da técnica de *oversampling*. Os autores extraíram 16 *features* agrupadas por 3 categorias, sendo elas: *Features* baseadas no nome do domínio, *features* baseadas em respostas DNS e por fim, *features* contextuais.

Os autores utilizaram o algoritmo CatBoost em seu sistema de detecção chamado KSDom, e compararam sua performance com outros três algoritmos, sendo eles SVM, GDBT, XGBoost, no qual o método proposto pelos autores foi superior aos três algoritmos comparados, nas 4 métricas de desempenho escolhida por eles, sendo elas a acurácia, precisão, F1-Score, e tempo de execução.

O *dataset* utilizado pelos autores na etapa de treinamento era composto por 10.000 domínios legítimos e 1.000 maliciosos.

2.8 Considerações Parciais

No Capítulo 2 foi apresentado a fundamentação teórica que ampara este trabalho, bem como a conceituação do DNS, aprendizado de máquina, técnicas de aprendizado de máquina, métricas utilizadas para avaliação de modelos de aprendizado de máquina, formas de validação do modelo e técnicas de balanceamento de classes.

A partir do conjunto de trabalhos presentes no estado da arte, é possível notar a importância de que a detecção de domínios maliciosos possui, bem como as diversas estratégias utilizadas até o presente momento para a obtenção de uma melhor performance, além da obtenção de resultados partindo de abordagens diferentes. Observa-se também a diversidade de formas nas quais são extraídas as *features* utilizadas nos trabalhos correlatos, como também, de onde os dados utilizados foram extraídos, como DNS Passivo de servidores recursivos e/ou autoritativos, e até a combinação com dados provenientes de DNS ativo. Nota-se que a grande maioria faz o uso de *features* léxicas para detecção de domínios maliciosos. É possível observar que trabalhos como (ANTONAKAKIS et al., 2010) (BILGE et al., 2011) (ANTONAKAKIS et al., 2011) (BAO; WANG; LAN, 2019) (WANG et al., 2020) apresentam bons resultados na detecção de domínios maliciosos e fazem o uso de algoritmos baseados em árvores de decisão para tarefas de classificação, sendo eles DT, RF, XGBoost e CatBoost.

Este trabalho busca realizar a detecção de domínios maliciosos, utilizando *features* extraídas exclusivamente do DNS passivo, sem qualquer uso de *feature* léxica, utilizando o algoritmo de aprendizagem de máquina supervisionado XGBoost, que possui uma série de melhorias, dentre elas técnicas de prevenção a *overfitting*, escalabilidade, processamento paralelo e distribuído, e que já provou sua eficiência na detecção de ataques DDoS (CHEN et al., 2018).

Capítulo 3

Metodologia

Este capítulo tem como objetivo apresentar a metodologia utilizada neste trabalho. Na seção 3.1 são apresentadas as tecnologias utilizadas para a realização deste estudo; na seção 3.2 é apresentada a metodologia utilizada; na seção 3.3 são feitas explanações sobre a aquisição dos dados e o pré-processamento dos dados; na seção 3.4 é apresentado o processo de extração de *features* que são utilizadas posteriormente para o treinamento dos algoritmos; na seção 3.5 é apresentado o procedimento e as variáveis analisadas para a execução da rotulação dos dados do *dataset* de pDNS; na seção 3.6 são apresentados os modelos de aprendizado de máquina selecionados, bem como as *features* a serem utilizadas, e a descrição das *features* e dos modelos; na seção 3.7 é apresentado como o modelo classificador foi avaliado; na seção 3.8 são apresentados os dados de treinamento e a técnica utilizada para o balanceamento das classes; na seção 3.9 as considerações parciais são apresentadas.

3.1 Tecnologias Utilizadas

Para execução deste trabalho, foi utilizado um computador com processador i7-4790, com 16GB de memória RAM e uma placa de vídeo NVIDIA GeForce GTX 745.

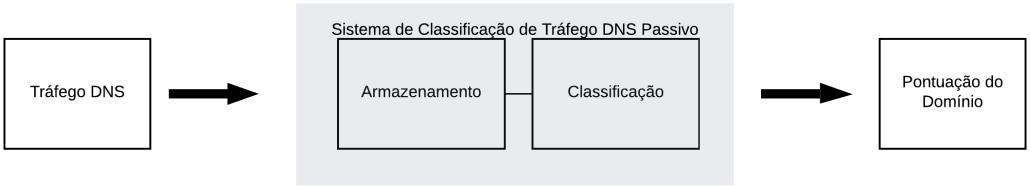
Para a execução do ambiente de desenvolvimento do projeto, foi utilizada a linguagem de programação Python na versão 3.7. A biblioteca de aprendizado de máquina utilizada foi a Scikit-Learn, que possui diversos algoritmos de aprendizado de máquina supervisionados, não-supervisionados, de regressões, pré-processamento, entre outros. Portanto, os algoritmos DT e RF foram fornecidos por essa biblioteca. O algoritmo XGBoost possui biblioteca própria. Scikit-Learn foi utilizada também para treinamento e testes dos modelos, bem como suas avaliações e extração de métricas

para mensurar o desempenho dos algoritmos. Para etapa de processamento dos dados, foi utilizada a biblioteca Pandas, na qual, os dados são carregados em *data frames*, possibilitando, assim, toda manipulação dos dados no decorrer do projeto. Para a tarefa de otimização dos hiperparâmetros do algoritmo XGBoost, foi utilizada a biblioteca Bayesian Optimization. Por fim, após o treinamento dos modelos, é necessário exportá-los em arquivos, para depois importar os modelos e fazer o uso dos mesmos. A biblioteca pickle foi utilizada para fazer tais exportações e importações dos modelos.

3.2 Visão Geral do Sistema

O sistema desenvolvido visa a detecção de domínios maliciosos com base em seu tráfego DNS como apresentado na Figura 3.1.

Figura 3.1: Diagrama geral da aplicação do sistema proposto.

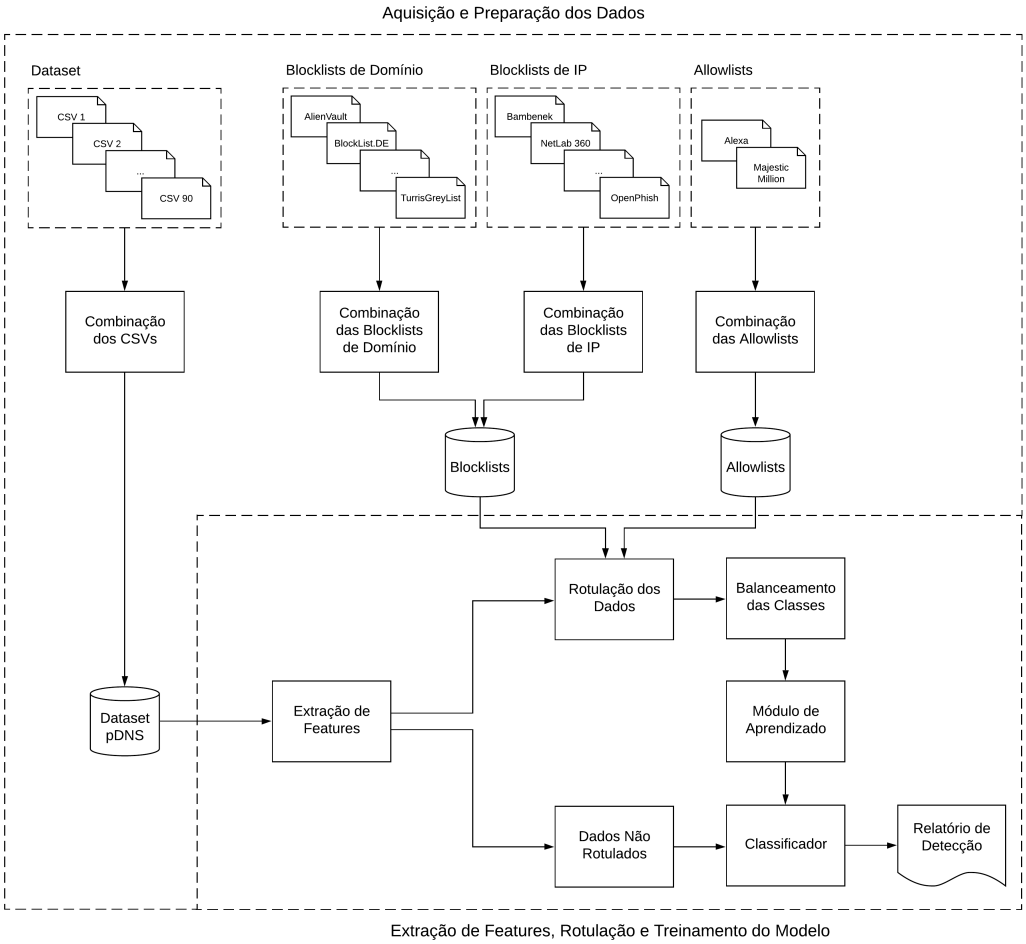


Fonte: Elaborado pelo Autor, 2020.

A base de dados, utilizada neste trabalho, foi gentilmente cedida pela Mnemonic. Este *dataset* é enviado para a etapa de extração de *features*. Nesta etapa pega-se os dados brutos do *dataset*, realiza o pré-processamento destes dados, gera-se as *features* necessárias e, por fim, descarta-se dados que não serão utilizados nas etapas futuras. Realizada esta extração de *features*, os dados são enviados para etapa de rotulação. Este passo consiste na detecção de domínios presentes em *Blocklists* e *Allowlists*, que também estão presentes no *dataset*, com a finalidade de rotular os dados. Após esta checagem, os dados rotulados como malicioso ou legítimo são enviados para o módulo de aprendizado. Neste módulo, os algoritmos são treinados e testados e são inferidas as métricas de avaliação. Modelo classificador treinado, ele está pronto para receber dados não rotulados, e detectar a partir de domínios desconhecidos se não maliciosos ou legítimos, e além gerar relatórios de detecção do tráfego DNS.

O método descrito anteriormente está exposto na Figura 3.2

Figura 3.2: Diagrama de funcionamento do sistema detalhado.



Fonte: Elaborado pelo Autor, 2020.

As subseções, a seguir, apresentam detalhes do funcionamento das etapas descritas anteriormente na Figura 3.2.

3.3 Aquisição e Preparação dos Dados

Algumas bases de dados foram necessárias para a execução deste trabalho, sendo elas, o *dataset* de DNS passivo, *allowlists* e *blocklists* para rotulação dos dados presentes no *dataset*.

Dataset DNS Passivo

A base de dados de DNS Passivo, foi gentilmente cedida pela Mnemonic e foi disponibilizada por meio de 1096 arquivos no formato ".csv". O Conjunto de dados brutos

consiste em pouco mais de 1 bilhão consultadas DNS agregadas e coletadas. Cada entrada dessas consultas possuem as seguintes variáveis:

- Uma consulta que foi realizada;
- Uma resposta para a consulta realizada;
- Uma classe do RR;
- Um tipo de registro (os registros DNS mais comuns para esta base de dados são registros A, CNAME, AAAA ou PTR;
- Um valor de TTL mínimo e máximo para o par de consulta e resposta;
- Um registro de *timestamp* para a primeira ocorrência do par;
- Um registro de *timestamp* para a última ocorrência do par;
- O número total de ocorrências do par, durante o período em que os dados foram coletados.

Majoritariamente, os registros da base utilizada são entradas do tipo A, ou seja, os registros mapeiam nomes de domínios aos seus respectivos endereços IP. Entretanto, há registros do tipo CNAME, cuja função é fornecer *aliases* entre nomes de domínios. E minoritariamente na base há registros AAAA e PTR.

A partir destes dados, algumas características foram extraídas para as combinações dos pares, como a quantidade de *queries*, o valor mínimo de TTL, a quantidade de mudanças do valor do TTL, tempo entre a primeira e última visualização do domínio, dentre outras.

Pré-processamento dos dados

Para a utilização de técnicas de aprendizado e máquina supervisionada, é necessário que haja uma rotulação dos dados para que os algoritmos aprendam por meio destes exemplos. Esta etapa de rotulação se deu por meio de *allowlists e blocklists* que serão apresentadas a seguir. Os registros foram rotulados em 3 tipos: legítimos, maliciosos e desconhecidos.

As *allowlists* utilizadas foram a Alexa e a Majestic Million. Estas listas apresentam 1 milhão de domínios mais acessados de acordo com suas métricas, entretanto, isso não garante que estes domínios estão livres de *malwares*. Porém, há uma baixa

probabilidade de domínios maliciosos compor estas listas, uma vez que estes são normalmente transitórios e não permaneçam nestas listas por um longo período de tempo (LISON; MAVROEIDIS, 2017).

Estão sendo utilizados dois tipos de *blocklists* para a rotulação dos dados. *Blocklists* de IP como a AlienVault, Blocklistde, CIArmy, DangerRulez, NoThink, TurrisGreyList, e as *Blocklists* de nomes de domínio como Bambenek e Netlab360 para DGAs, MalwareDomain para malwares, OpenPhish para phishing, e a URLVir que possui tanto IPs como nome de domínios maliciosos.

Para esta etapa de pré-processamento dos dados, é realizada uma fusão das fontes de *allowlists*, gerando uma lista com nomes de domínios únicos contendo 1.743.745 milhões de domínios, com as seguintes entradas, nome do domínio e posição em que ocupa na lista. Para as *blocklists* foram geradas duas listas separadas, uma contendo a fusão de todas fontes que possuem nomes de domínio, tendo como saída apenas uma única lista com todos nomes de domínios maliciosos, cujo seu tamanho é de 1.673.353 domínios e outra fusão para as listas que possuem IPs, resultado assim em uma lista com 154.798 IPs.

3.4 Extração de *Features*

As *features* utilizadas neste trabalho foram extraídas exclusivamente do tráfego de DNS passivo e são um *subset* de *features* apresentadas por Lison e Mavroeidis (2017), e estão apresentadas e descritas na Tabela 3.1.

Tabela 3.1: *Features* extraídas do pDNS e suas descrições.

Features	Descrição
nb_queries	Números de consultas observadas no DNS passivo para o par (domínio, IP)
min_ttl	Valor do TTL mínimo para o par
ttl_changes	Número de mudanças do valor do TTL para o par
nb_domain_queries	Número total de consultas para o domínio independente do IP resolvido
nb_ips	Número de endereços IP distintos que foram resolvidos para o domínio
nb_aliases	Número de aliases para o domínio
nb_address_queries	Número total de consultas observadas para o endereço IP, independente do domínio resolvido
nb_domains	Número de domínios resolvidos para este endereço IP
lifespan	Tempo (em segundos) entre a primeira e a última observação do par
domain_lifespan	Tempo (em segundos) entre a primeira e a última observação do domínio
doamin_inactivity	Tempo (em segundos) desde a última observação do domínio
address_lifespan	Tempo (em segundos) entre a primeira e a última observação do IP

Para extração de algumas *features* utilizadas neste trabalho, não foi necessário realizar algum tipo de modificação no campo fornecido pelo *dataset* em seu formato bruto. As *features* que foram providas por meio do *dataset* bruto são: “nb_queries”,

“min_ttl” e “lifespan”. A *feature* “nb_queries”, refere-se à quantidade de consultas observadas pelo par (domínio e IP), campo este já fornecido pelo *dataset* adquirido. O “min_ttl” é a *feature* responsável por indicar qual valor mínimo do campo TTL para o par. E, por fim, o tempo (em segundos) entre a primeira e a última observação do par representado na *feature* "lifespan".

Entretanto, para extração das demais *features* foi necessário o cruzamento dos dados presentes em cada arquivo que compunha o *dataset* para sua extração. A seguir, serão apresentados os processos necessários para extração de tais *features*.

Uma das *features* utilizadas é a “ttl_changes”, na qual ela significa a quantidade de mudanças no valor do TTL para o par (domínio e IP). Para computação de tal *feature*, com os dados brutos carregados no *data frame*, é contado a quantidade de mudanças do valor no campo TTL do DNS para o par, inserindo, assim, o valor contado no *data frame* de *features*.

A *feature* “nb_domain_queries” é a responsável por contar a quantidade de vezes que o domínio foi consultado, independente de qual IP a consulta retornou. Portanto, conta-se a quantidade de vezes que o domínio estava presente do campo de consultas realizadas nos dados brutos. Análogo a esta *feature* tem a “nb_address_queries”, entretanto para tal *feature* conta-se a quantidade de vezes em que o IP foi consultado, independente do domínio resolvido por ele.

A *feature* “nb_ips” realiza a quantidade de IPs distintos que foram resolvidos para tal domínio. Portanto, pesquisa-se no *dataset* por um domínio e conta-se a quantidade de IPs distintos que foi resolvido para tal domínio. O mesmo acontece para *feature* “nb_aliases”, contado assim a quantidade de aliases para o domínio.

A *feature* “nb_domains” é extraída realizando a contagem total do número de domínios resolvidos para tal IP.

Por fim, há *features* que são providas por meios dos campos de *timestamp* presentes do *dataset* de dados brutos. Todas as *features* extraídas apresentam o tempo em segundos. A primeira *feature* do tipo extraída é a “domain_lifespan”, que apresenta o tempo entre a primeira e última observação do domínio. Para realizar a extração de tal *feature*, verifica-se no *dataset* bruto a primeira e última vez que o domínio foi consultado e subtrai-se os valores para obtenção da *feature*. O mesmo procedimento é realizado para *feature* “address_lifespan”, entretanto, para esta *feature* dve-se levar em consideração a primeira e última observação do IP, assim, ao subtrair os valores é gerado a *feature*. É importante ressaltar de que no *dataset* cedido não há um campo específico para primeira aparição do domínio, ou IP. Para obter este valor é necessário percorrer o *dataset* em busca da primeira vez em que aquele domínio apareceu, e se há

outra aparição anterior a esta, uma vez que se busca a primeira aparição de um campo específico (domínio ou IP), e no *dataset* contém apenas a primeira e última ocorrência para o par (domínio e IP). Localizados os valores da primeira e última ocorrência, realiza-se a operação de subtração e obtém a *feature* desejada. Por fim, a última *feature* a ser apresentada seu processo de extração é a "domain_inactivity". Esta *feature* representa o tempo desde a última observação do domínio, para extraí-la é consultada a última ocorrência do domínio, e subtrai pela data atual (no formato de segundos). Como resultado, obtém-se a *feature* em questão.

Ao término da extração das *features*, apresentadas nesta seção, tem-se como resultado um *dataset* contendo o nome do domínio consultado e resposta para tal consulta (campos herdados *dataset* fornecido), acrescido das 12 *features*, resultando, assim, em um *dataset* contendo 14 campos. Os campos nomes de domínio e resposta serão descartados futuramente do processo de aprendizado do algoritmo, entretanto é necessário mantê-los no momento, pois são essenciais para o processo de rotulação dos dados.

3.5 Rotulação do Dataset de DNS Passivo

Para etapa de rotulação do DNS Passivo, executa-se um script em que se verifica se o domínio em questão está presente no compilado de *allowlists*, *blocklists*, gerados anteriormente, e se o IP de resposta também está no compilado de *blocklists* e insere no *data frame* o respectivo valor indicando sua presença em alguma destas listas.

Apesar da importância desta rotulação dos dados, estas listas estão propensas a erros e imprecisões. Portanto, para uma melhor qualidade dos dados de treinamentos, adotou-se um mecanismo de reputação semelhante ao apresentado por Lison e Mavroeidis (2017), no caso para este trabalho, há 3 possibilidades de rotulação para os registros DNS, sendo eles: malicioso, legítimo ou desconhecido.

Para que um domínio seja rotulado como malicioso, é necessário que uma das condições seja satisfeita:

- A reputação do domínio e IP forem conhecidos, e ambas foram maliciosos.
- A reputação de um dos campos (domínio ou IP) for desconhecido, e que o campo conhecido seja malicioso.

Para que um domínio seja rotulado como legítimo, é necessário que:

- A reputação apenas de um dos campos (domínio ou IP) for conhecida, e ela for legítima.

- A reputação dos campos domínio e IP forem conhecidas, e ambos campos serem legítimos.

Para que um domínio seja rotulado como desconhecido, é necessário que:

- A reputação do domínio e IP forem conhecidos e há uma divergência na reputação.
- A reputação do domínio e IP forem desconhecidos.

Portanto, um domínio será rotulado como desconhecido, apenas se os dados obtidos pelas listas forem conflituosos, ou ainda, quando não houver dados suficientes para rotulação deste domínio. Deste modo, ele será excluído da etapa de treinamento do modelo classificador.

3.6 Modelos de Machine Learning

A partir do problema abordado neste trabalho e do levantamento bibliográfico realizado, é possível observar que, em sua maioria, os trabalhos presentes no estado da arte usam algoritmos baseados em árvore de decisão, e que obtiveram ótimos resultados na classificação dos domínios utilizando os algoritmos de Árvore de Decisão (DT), Floresta Aleatória (RF) e mais recentemente o uso do Extreme Gradient Boosting (XGBoost) para detecção de domínios maliciosos relacionados a DGAs e a pornografia.

Neste trabalho, optou-se pela utilização do algoritmo XGBoost, que pode ser considerado uma evolução dos algoritmos baseados em árvore de decisão, e que o algoritmo escolhido possui uma série de melhorias, se comparado aos outros dois apresentados anteriormente.

Portanto, os algoritmos selecionados para a detecção de domínios maliciosos foram os de Árvore de Decisão (DT), Floresta Aleatória (RF) e XGBoost.

Todos algoritmos selecionados, e apresentados anteriormente, tiveram como entrada o mesmo *dataset*. As *features* utilizadas neste trabalho são um *subset* de *features* apresentadas em (LISON; MAVROEIDIS, 2017), que estão apresentadas na Tabela 3.1.

Os algoritmos de aprendizagem de máquina DT e RF foram executados em seu modo padrão, sem a alteração de quaisquer parâmetros fornecidos pela biblioteca scikit-learn. Entretanto, para que o algoritmo XGBoost obtenha uma melhor performance, é necessário a inserção dos parâmetros iniciais.

Para a seleção dos melhores valores dos parâmetros a serem inseridos no algoritmo XGBoost, foi utilizado otimização bayesiana por meio da biblioteca apresentada por Nogueira (2014). Esta biblioteca é construída com base na inferência bayesiana e no processo gaussiano, e busca encontrar o valor máximo e mínimo de uma função desconhecida no menor número possível de iterações. Portanto, para que se consiga otimizar os parâmetros em questão, criou-se uma lista com os parâmetros a serem otimizados, bem como valores máximos e mínimos para cada. Os parâmetros otimizados por meio desta otimização bayesiana são: learning_rate, n_estimators, max_depth, subsample, gamma, reg_lambda.

Para esta otimização utilizou-se de 21 pontos iniciais e 300 iterações, no qual obteve como resposta os parâmetros apresentados no algoritmo que geraram os resultados apresentados no Capítulo 4. Os valores dos parâmetros "pontos inciais"e "iterações"foram definidos empiricamente pelo autor.

3.7 Avaliação do Modelo Classificador

A avaliação da abordagem apresentada neste trabalho foi feita com base no *dataset* de DNS Passivo. Foi utilizada a técnica de K-fold Cross-Validation para etapas de treinamento e teste do modelo, na qual foi utilizado K = 5. E a métrica utilizada para validar a performance do classificador foi a Área sobre a Curva ROC (AUC). Portanto, cada vez que se executa o treino e teste do modelo classificador é gerado uma Curva ROC, com o valor de sua área. Após as cinco curvas geradas, é gerada uma sexta curva, sendo ela a média das outras cinco curvas anteriores, e sua área é calculada.

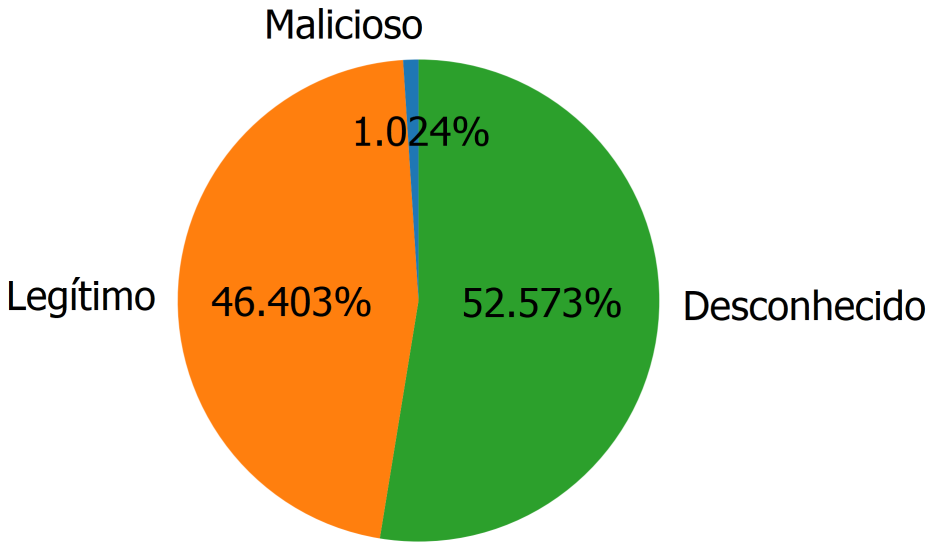
3.8 Dados de Treinamento

A base de dados utilizada neste trabalho, foi gentilmente cedida pela Mnemonic como apresentado na seção 3.3, entretanto, foi utilizada apenas uma fração deste *dataset*, o que corresponde aos noventa primeiros arquivos do formato "CSV"fornecido.

O *sub dataset* utilizado é composto por 89.997.413 milhões de registros DNS, que após executado a rotulação destes dados, há 46,40% de tráfego de DNS passivo é legítimo, 1,02% malicioso, e 52,57% desconhecido como é apresentado na Figura 3.3.

Neste trabalho, optou-se pela utilização de técnicas de aprendizado de máquina supervisionado, portanto, os dados a serem utilizados precisam ser rotulados. Removendo os dados rotulados como "desconhecidos", tem-se 2,16% dos dados rotulados como maliciosos e legítimos ocupando 97,84% do *dataset*. A análise relatada acima

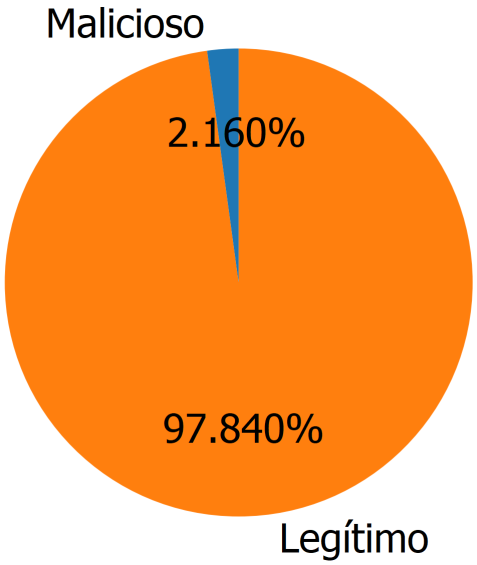
Figura 3.3: Análise do tráfego DNS



Fonte: Elaborado pelo Autor, 2020.

fica explícita na Figura 3.4.

Figura 3.4: Análise do *subset* do tráfego DNS.



Fonte: Elaborado pelo Autor, 2020.

Como ilustrado na Figura 3.4 as classes estão extremamente desbalanceadas, e isso pode gerar alguns impactos em algoritmos de aprendizado de máquina supervisionado, como por exemplo, o enviesamento dos dados. Há dois conjuntos de técnicas utilizados para balanceamento dos dados, sendo elas técnicas de *Oversampling* e *Undersampling*. Técnicas de balanceamento baseadas em *Oversampling* utilizam diferentes

técnicas para geração de dados sintéticos, a fim de obter a mesma quantidade de dados em ambas as classes. Técnicas baseadas em *Undersampling* optam por equalizar as classes reduzindo a quantidade de dados da classe majoritária, até que haja o equilíbrio entre ambas. Ambas técnicas geram consequências aos dados, conforme discutido no Capitulo 2.5. Optou-se pela técnica de Random Undersamplig (RUS)(FERNÁNDEZ et al., 2018), que escolhe aleatoriamente quais dados da classe majoritária serão selecionados para que ambas as classes presentes no *dataset* possuam a mesma quantidade de dados.

Após a aplicação da técnica de RUS para balanceamento das classes, resultou em um *subset* contendo 1.843.970 registros de DNS, estando dividido igualmente entre as classes legítimo e malicioso. A partir deste *subset* balanceado, serão realizados testes com os 3 algoritmos diferentes, para que seja possível a análise de desempenho de ambos.

Como discutido anteriormente, técnicas de *Undersampling* podem resultar em perdas significativas de dados importantes. Para realizar tal avaliação, e checar também se há indícios de *overfitting* ou *underfitting*, foi executado a técnica RUS dez vezes. O resultado, portanto, foi 10 *subsets* aleatórios de registro DNS contendo os mesmos registros de DNS malicioso. Entretanto, com uma grande variação de registros de DNS legítimo por conta da proporção apresentada no *dataset*, que foi exposto na Figura 3.4.

3.9 Considerações Parciais

A abordagem apresentada neste trabalho faz o uso de uma base de DNS passivo, que após rotulado em três classes - legítimos, maliciosos e desconhecidos - utiliza-se apenas as classes legítimos e maliciosos em busca de extração de *features* comuns das classes, e que possam diferenciá-las com base nos mais diversos usos que domínios maliciosos possam vir a ter, a fim de que, algoritmos de aprendizagem de máquina supervisionados possam aprender a partir dessas amostras, e que sejam capazes de generalizar para realizar a classificação de tráfego desconhecido.

Devido ao desbalanceamento das classes, um balanceamento se fez necessário, optando-se então, pela técnica de RUS. Para treino e teste dos modelos, será utilizado Stratified K-fold Cross-Validation, e como métrica de validação de performance, AUC.

As *features* extraídas, que serão enviadas à etapa de treinamento, são um *subset* de *features* apresentadas no estado da arte. E por meio da optimização bayesiana, deseja-se obter um bom desempenho com o modelo classificador utilizando o algoritmo XGBoost. Para este trabalho assume-se que tráfego de DNS passivo de uma

mesma classe possuam similaridades.

Capítulo 4

Resultados

Este capítulo possui o objetivo de apresentar os resultados obtidos durante a execução deste trabalho por meio dos experimentos realizados. Foram analisados três modelos de classificação diferentes, dentre eles Árvore de Decisão, Floresta Aleatória e XGBoost, e os resultados obtidos foram comparados entre si. Posteriormente realizou-se outro experimento com o melhor modelo dentre os três avaliados com objetivo de analisar se houve perda significativa de dados ou *overfitting* do modelo. Por fim, no último experimento realizado executa-se o melhor modelo sobre os dados rotulados inicialmente como "desconhecidos".

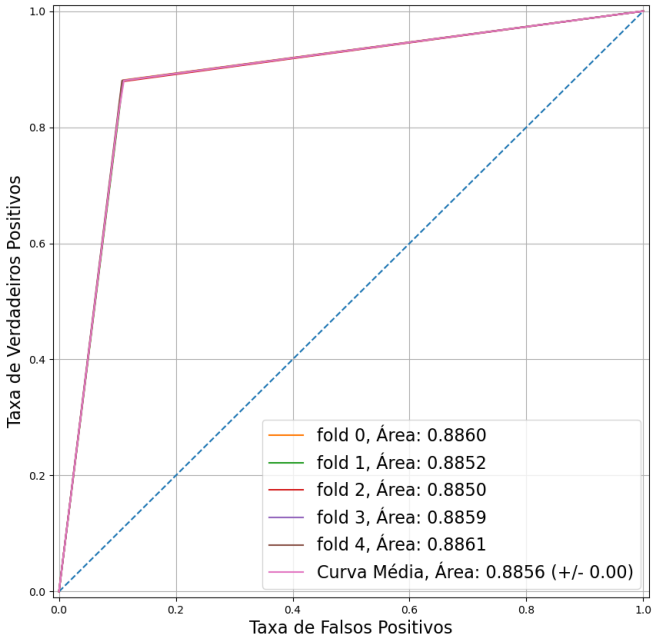
Na seção 4.1 serão apresentados os resultados obtidos dos três modelos treinados, bem como um comparativo entre eles, observando, assim, a melhor performance. A seção 4.2 apresenta uma análise do melhor modelo e um novo experimento em busca de detectar *overfitting* ou perda de dados por conta da técnica de RUS. Na seção 4.3 o melhor modelo apresentado anteriormente realiza predições sobre os dados previamente rotulados como desconhecidos. E, por fim, na seção 4.4 são apresentadas as considerações parciais deste capítulo.

4.1 Comparativo Entre os Modelos

Utilizando a técnica de Stratified K-Fold Cross-Validation, com k=5, os três algoritmos (DT, RF, XGBoost) foram treinados e testados utilizando o mesmo conjunto de dados. Após cada uma das cinco etapas de treinamento e validação, foi gerado uma AUC, na qual foram plotadas as cinco na mesma imagem. Uma sexta AUC foi plotada também, representando uma média das cinco AUCs plotadas nos gráficos.

Na Figura 4.1 estão plotadas as AUCs referentes ao algoritmo de Árvore Decisão, que obteve uma AUC média de 0.8856.

Figura 4.1: AUC DT.

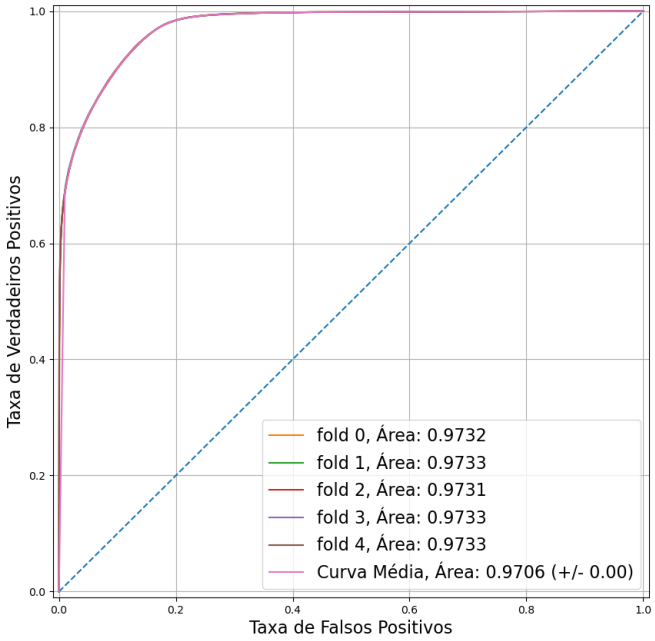


Fonte: Elaborado pelo Autor, 2020.

A Figura 4.2 representa as AUCs geradas pelo uso do algoritmo de Floresta Aleatória, tendo uma AUC média no valor de 0.9706.

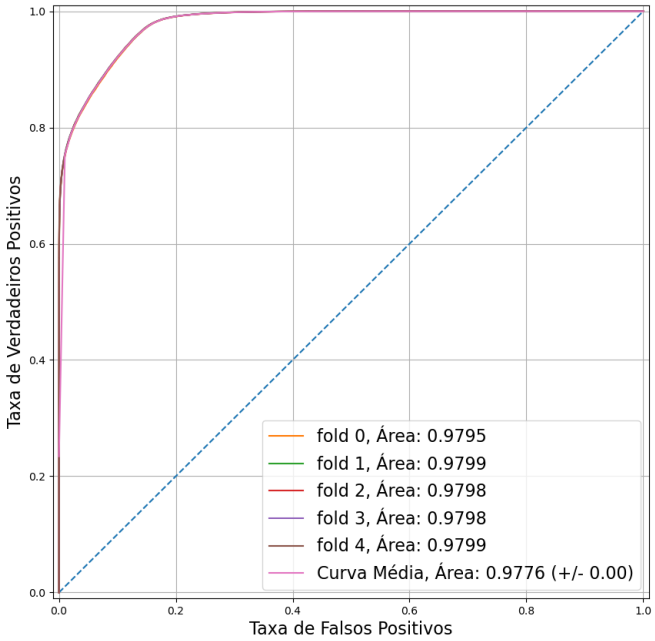
Na Figura 4.3, estão plotadas as AUCs geradas após a execução da etapa de treinamento e teste do algoritmo XGBoost, após sua otimização Bayesiana, que resultou em uma AUC média de 0.9776.

Figura 4.2: AUC RF.



Fonte: Elaborado pelo Autor, 2020.

Figura 4.3: AUC XGBoost.

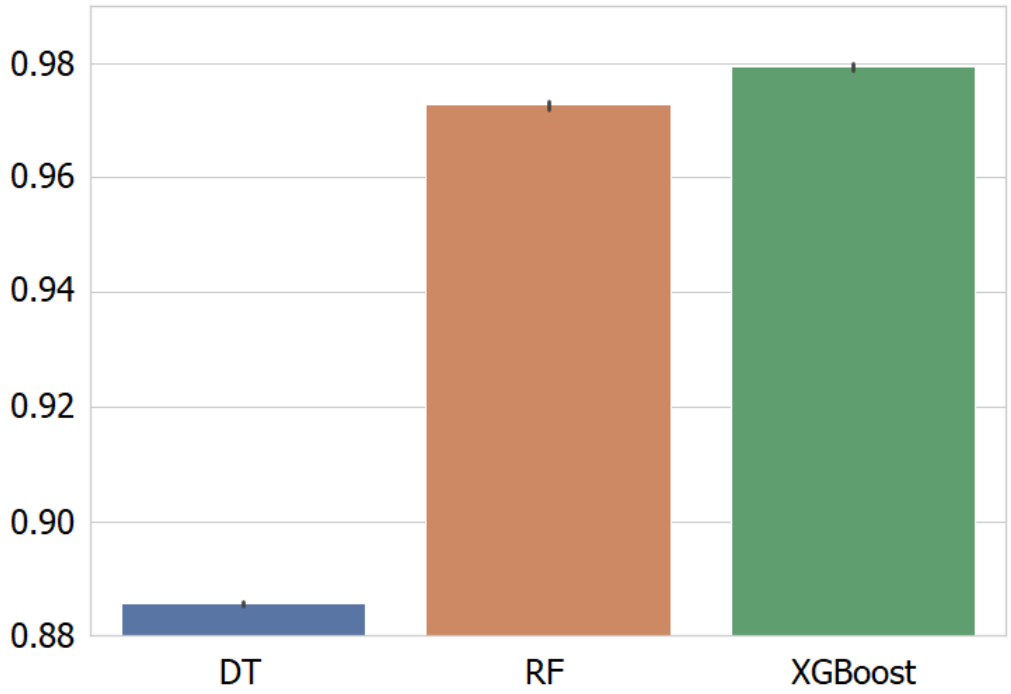


Fonte: Elaborado pelo Autor, 2020.

É possível observar que o resultado do classificador que utilizou o algoritmo XGBoost possui uma AUC maior em aproximadamente 0,007 do que o algoritmo que fez o uso do algoritmo de Floresta Aleatória, e que ambos são muito superiores ao modelo que fez o uso de Árvore de Decisão como seu algoritmo.

Na Figura 4.4 é apresentado um gráfico de intervalo de confiança, sintetizando os resultados apresentados anteriormente de cada modelo classificador.

Figura 4.4: Intervalo de Confiança.



Fonte: Elaborado pelo Autor, 2020.

Como apresentado na Figura 4.4 o modelo que fez o uso do algoritmo XGBoost, foi o que obteve o melhor resultado, conforme esperado. Por meio do resultado de suas AUCs é possível concluir que a chance do modelo acertar sua predição é de 97,76%. O gráfico do intervalo de confiança representado na Figura 4.4 apresenta uma baixa variação nas AUCs resultantes dos modelos classificadores.

4.2 Análise de *Overfitting* no Modelo

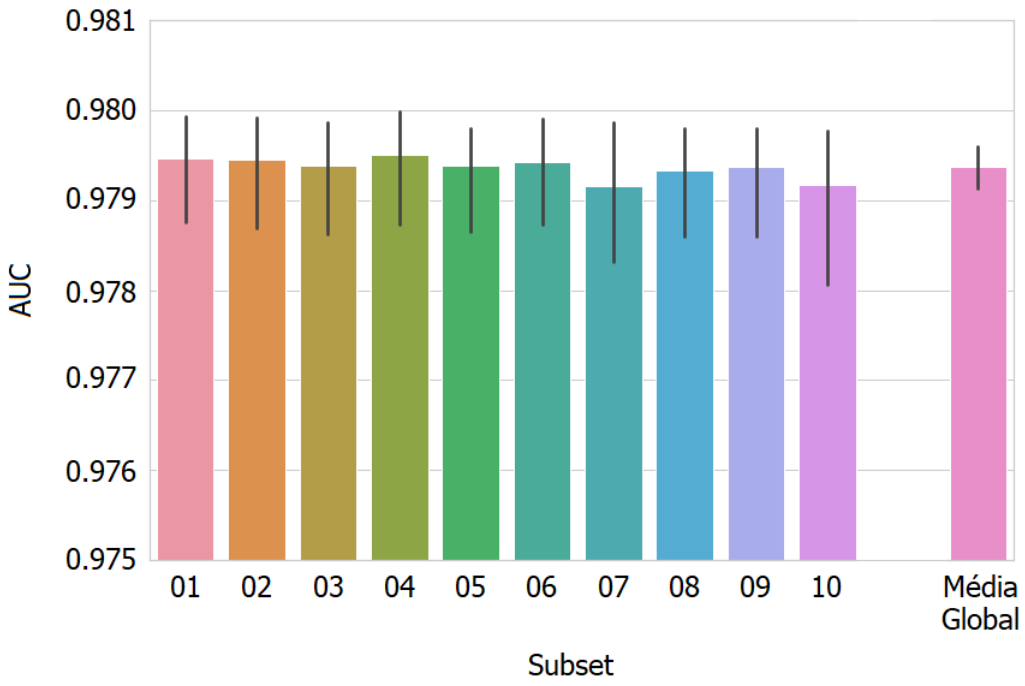
Como apresentado na seção 4.1, o modelo que apresentou o melhor resultado, foi o que teve o uso do algoritmo XGBoost. Entretanto, como foi utilizado o *subset* balanceado pela técnica de *undersampling*, há possibilidade de que haja uma perda significativa de dados, ou que ocorra *overfitting*. No primeiro caso, estaria perdendo dados

significativos para o aprendizado do modelo, o que resultaria em uma dificuldade do modelo em reconhecer outros domínios maliciosos. No caso de *overfitting*, não é gerado um aprendizado no modelo treinado, ele simplesmente decora os dados nos quais foi apresentado na etapa de treinamento, resultando, assim, em predições ruins para dados desconhecidos.

Para realizar a detecção ou não destes possíveis problemas, além da técnica de Stratified K-fold Cross-Validation utilizada na etapa de treino e teste do modelo, foram gerados dez novos *subsets* para avaliação do comportamento do modelo classificador. A partir do *dataset* original, extremamente desbalanceado como apresentado na Figura 3.3, executou-se a técnica RUS dez vezes, o que resultou em dez *subsets* aleatórios de registro DNS contendo os mesmos registros de DNS malicioso. Entretanto, há uma grande variação dos dados de DNS passivo dos domínios legítimos por conta da proporção presente no *dataset* original.

O modelo classificador, utilizando XGBoost, foi treinado por cada um destes dez *subsets*, com o objetivo de observar a variação de sua AUC para cada *dataset*. O resultado está apresentado na Figura 4.5.

Figura 4.5: Gráfico de intervalo de confiança para cada *subset* testado, e média global destes 10 *subsets* com XGBoost.



Fonte: Elaborado pelo Autor, 2020.

A baixa variância da AUC nos dez *subsets* diferentes, apresentados na Figura 4.5,

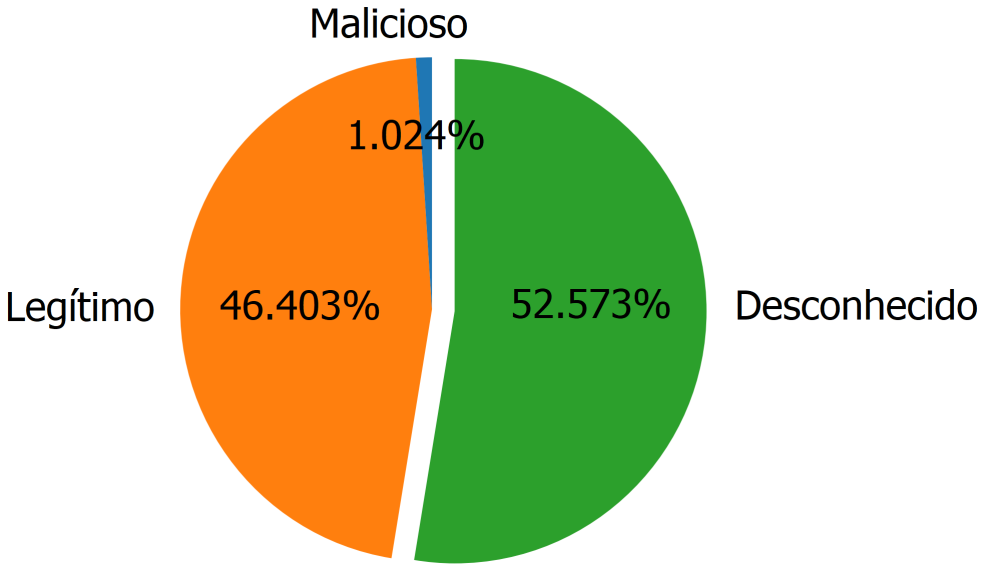
é um indicativo de que não há *overfitting* no modelo apresentado e de que a técnica aplicada para o balanceamento dos dados não resultou em uma perda significativa de dados em seu processo. A Figura 4.5 apresenta também um décimo primeiro *subset* com o nome de "média global", para a composição desta informação no gráfico, que foram utilizados os resultados dos dez *subsets* apresentados. E por meio desta média global apresentada, é possível observar que ela possui um intervalo de confiança muito baixo 0,000239, o que indica que há uma alta consistência do resultado dos experimentos realizados.

4.3 Teste do Modelo em Domínios Desconhecidos

Como apresentado na seção 4.1 o modelo com maior AUC foi o que utilizou o algoritmo de aprendizagem de máquina XGBoost. Na seção 4.2 o experimento realizado permite observar que não há indicativo de *overfitting* no modelo. Nesta seção um novo experimento realizado, utilizou-se do modelo treinado no experimento apresentado na seção 4.1 para realizar predições sobre o conjunto de dados rotulados como desconhecidos na seção 3.8.

O *dataset* inicial era composto por 1,024% domínios rotulados como maliciosos, 46,403% legítimos e 52,573% rotulados como desconhecidos como apresentado na Figura 4.6.

Figura 4.6: Análise do tráfego DNS.

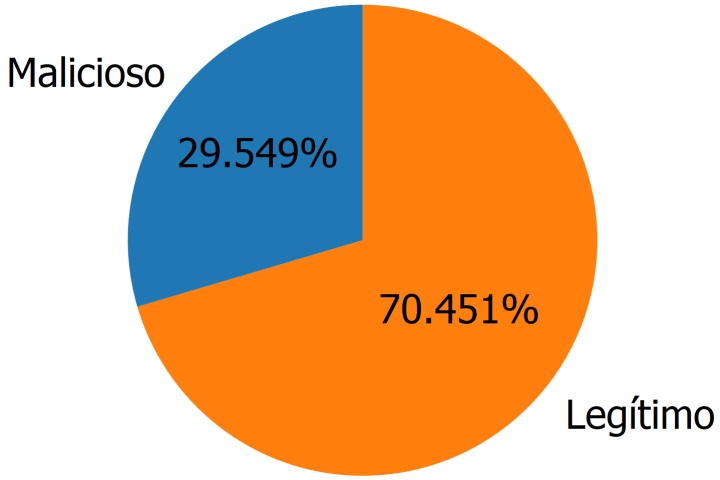


Fonte: Elaborado pelo Autor, 2020.

Ao todo, domínios rotulados como desconhecidos correspondem a 47.314.343 de

registros DNS. Estes registros foram dados como entrada para que o modelo realizasse a predição de suas classes e a probabilidade deste registro DNS estar associado à classe de domínios malicioso e legítimos. Considera-se que o domínio pertence a tal classe se o valor de predição da probabilidade for superior a 0,5. O resultado deste experimento está apresentado na Figura 4.7, no qual 29,549% dos domínios foram detectados como maliciosos e 70,451% como legítimos.

Figura 4.7: Detecção de domínios maliciosos por meio do modelo classificador proposto.



Fonte: Elaborado pelo Autor, 2020.

O resultado do experimento apresentado na Figura 4.7 expressa a importância de um modelo de detecção de domínios maliciosos. Os domínios que até o momento eram rotulados como desconhecidos, sejam por falta de informações, ou por conflito entre os dados fornecidos pelas listas, foram detectados por meio do modelo classificador.

4.4 Considerações Parciais

A Abordagem utilizada neste trabalho possui como fonte dados o tráfego DNS passivo, que gerou um *subset* balanceado e igualmente dividido entre domínios maliciosos e legítimos, por meio da técnica RUS. Este *subset* foi fornecido como entrada para três algoritmos baseados em árvore de decisão e conhecidos na literatura.

É possível observar que somente *features* extraídas por meio do DNS passivo consegue realizar a detecção de domínios maliciosos e com uma boa performance de algoritmos como XGBoost e RF. XGBoost obteve uma AUC melhor diante dos outros dois algoritmos e possui alguns diferenciais, dentre eles, a capacidade ter processamento paralelo e distribuído.

XGBoost é um algoritmo que possui diversos hiperârametros e é necessário que se faça o ajuste de tais para que o algoritmo tenha uma boa performance, e na abordagem utilizada, optou-se pela otimização bayesiana.

O modelo desenvolvido foi capaz de identificar domínios maliciosos que até o momento não estavam inseridos em *blocklists*, realizando a detecção destes de forma automática.

Comparando o resultado obtido pelo modelo classificador proposto, utilizando o algoritmo XGBoost com otimização bayesiana com o trabalho de Lison e Mavroeidis (2017), os autores não relatam a utilização de qualquer abordagem para o balanceamento das classes, considerando que eles possuíam três classes para classificação do DNS passivo, sendo elas malicioso, legítimo e sinkhole. Os autores também fizeram o uso de uma abordagem neural para resolução de tal problema.

No trabalho aqui apresentado, optou-se por uma classificação binária de maliciosos e legítimos e no algoritmo XGBoost. Comparando as AUC dos classificadores, a abordagem apresentada obteve uma AUC média de 0.9776, ligeiramente melhor que o pior resultado obtido por um dos doze modelos utilizando redes neurais artificiais apresentado em (LISON; MAVROEIDIS, 2017). Entretanto, os autores utilizaram uma gama maior de *features* incluindo as léxicas, e no trabalho apresentado utilizou um *subset* das *features* utilizadas, e exclusivamente extraídas do tráfego DNS, ignorando assim *features* léxicas e proveniente de outras fontes de dados que não sejam o próprio trafego DNS passivo.

A AUC pode não ser uma boa métrica quando há um *dataset* desbalanceado (SAITO; REHMSMEIER, 2015) como o apresentado pelos autores em (BAO; WANG; LAN, 2019). Ainda que comparados os autores que desenvolveram um modelo classificador utilizando XGBoost e uma técnica de *undersampling* para redução do desbalanceamento das classes no *dataset* utilizado. No trabalho aqui apresentado, foi utilizado o mesmo algoritmo e uma técnica específica de *undersampling* a RUS. Entretanto, os autores desenvolveram um trabalho com o objetivo de identificar domínios relacionados a DGAs, enquanto que o estudo em questão possui o foco de identificar uma diversidade de domínios maliciosos, mesmo que este não especifique qual tipo de maliciosidade tal domínio está relacionado. Sobre os resultados, Bao, Wang e Lan (2019) atingiram uma AUC de 0.974 utilizando *features* de acesso, sendo assim um resultado abaixo do que este trabalho. A melhor AUC obtida pelos autores foi de 0.994 utilizando todas *features*, inclusive as léxicas.

Capítulo 5

Conclusões

Este capítulo possui o objetivo de apresentar as conclusões obtidas por meio deste trabalho, e também apresentar as recomendações para trabalhos futuros e as produções geradas.

Na seção 5.1 serão apresentadas as conclusões gerais do trabalho, na seção 5.2 as recomendações para trabalhos futuros, e, por fim, na seção 5.3 as produções geradas a partir do estudo aqui apresentado.

5.1 Conclusões

O DNS é um componente essencial para o funcionamento da Internet, tendo como sua principal função a tradução de nomes de domínio em IPs e vice-versa. Considerando sua importância, pessoas mal intencionadas podem fazer o uso de tais estruturas para prática de atos maliciosos. A partir deste processo de resolução do nome de domínio, é possível realizar a coleta passiva destas consultas e respostas, bem como realizar a extração de *features* deste tráfego de DNS passivo, com o objetivo de identificar domínios maliciosos.

Neste trabalho foi desenvolvido um método para detecção automática destes domínios maliciosos por meio do tráfego DNS passivo juntamente com o algoritmo de aprendizado de máquina supervisionado XGBoost, o qual obteve uma AUC média de 0.9776 utilizando *features* extraídas exclusivamente do DNS passivo.

O bom desempenho obtido pelo modelo classificador é expressa por sua AUC que se aproxima de resultados de trabalhos correlatos, nos quais utilizaram diferentes abordagens. Na abordagem apresentada, neste trabalho, utilizou-se apenas um *subset* de *features*, que foram utilizadas em (LISON; MAVROEIDIS, 2017).

O método apresentado faz o uso de um compilado de *blocklists* das quais contém

domínios relacionados a DGAs, Phishing, Malwares dentre outros tipos de maliciosidades. Portanto, o classificador detecta apenas se o domínio em questão é maliciosos ou legítimo, não distinguindo, assim, que tipo de maliciosidade o domínio pertence.

A otimização dos hiperparâmetros do algoritmo XGBoost foi realizado por otimização Bayesiana, na qual se busca encontrar o valor máximo ou mínimo de função desconhecida no menor número possível de iterações. Desse modo, para esta abordagem foi utilizado vinte e um pontos iniciais e trezentas iterações, definidos de forma empírica. Com isso, aumentando a quantidade de iterações, aumenta-se também o tempo no qual a otimização será executada, logo, pode-se obter resultados melhores.

A baixa variância da AUC expressa no segundo experimento permite concluir que a abordagem apresentada utilizando RUS e XGBoost não resultou na perca significativa de dados, e também é um indicativo de que o modelo não apresenta indícios de *overfit*.

O resultado do terceiro experimento realizado, possibilita afirmar de que o método apresentado neste trabalho é capaz de realizar a detecção automática de domínios maliciosos de forma precoce, ou seja, antes que estes domínios possam estar presentes em *blocklists*.

Portanto, pode-se concluir que este trabalho atingiu seus objetivos, sendo capaz de realizar a detecção de domínios maliciosos de forma automática utilizando o algoritmo XGBoost, com uma elevada probabilidade de acerto da predição realizada, utilizando *features* extraídas exclusivamente do tráfego DNS passivo, tendo o modelo classificador feito o uso de otimização Bayesiana para obtenção de um bom desempenho. Este trabalho possibilita também desenvolvimentos futuros nos quais estão apresentados na seção 5.2.

5.2 Trabalhos futuros

O trabalho realizado, utiliza um *subset* de *features* apresentadas em (LISON; MAVROEIDIS, 2017) sendo extraídas exclusivamente do tráfego DNS. Um dos trabalhos futuros que pode ser realizado, é a expansão de tais *features* utilizadas, podendo incluir *features* como geolocalização do IP resolvido por aquele domínio, ou até mesmo a extração de novas *features* voltadas ao tráfego DNS.

O método para detecção de domínios maliciosos aqui apresentado utiliza compilados de *blocklists* de diferentes tipos de maliciosidade, entretanto, o modelo faz o uso de uma classificação binária o que possibilita saber apenas se o domínio checado é malicioso ou não. Uma melhoria que poder ser realizada neste quesito é o desenvolvimento de uma abordagem nas quais aplique técnicas de classificação multiclasse,

possibilitando assim saber a qual tipo de maliciosidade o domínio pertence.

Ao longo do tempo, domínios maliciosos podem vir a mudar o seu comportamento. Caso isso ocorra, o desempenho deste modelo classificador utilizado para detecção de domínios maliciosos cairia. Uma vez que, o escopo definido para este projeto não engloba a adaptabilidade do modelo. Portanto, outra sugestão de trabalho futuro é a aplicação de *Concept Drift* para o modelo continue ajustado na detecção de domínios maliciosos, mesmo que estes possam vir a mudar seu comportamento com o objetivo de se esquivarem da detecção.

5.3 Produções

Sobre as produções obtidas neste trabalho, foram publicados dois artigos. O primeiro, intitulado *"Detection of Malicious Domains Using Passive DNS with XGBoost"* foi publicado na *"IEEE International Conference on Intelligence and Security Informatics (ISI)"*. O segundo, com o título: *"XGBoost Applied to Identify Malicious Domains Using Passive DNS"* foi publicado na *"IEEE International Symposium on Network Computing and Applications (NCA 2020)"*.

Referências

ALRWAIS, S.; YUAN, K.; ALOWAISHEQ, E.; LI, Z.; WANG, X. Understanding the dark side of domain parking. In: *23rd {USENIX} Security Symposium ({USENIX} Security 14)*. [S.l.: s.n.], 2014. p. 207–222.

ANTONAKAKIS, M.; PERDISCI, R.; DAGON, D.; LEE, W.; FEAMSTER, N. Building a dynamic reputation system for DNS. *Proceedings of the 19th USENIX Security Symposium*, p. 273–289, 2010.

ANTONAKAKIS, M.; PERDISCI, R.; LEE, W.; VASILOGLOU, N.; DAGON, D. Detecting malware domains at the upper DNS hierarchy. *Proceedings of the 20th USENIX Security Symposium*, p. 411–426, 2011.

ASSADHAN, B.; MOURA, J. M.; LAPSLEY, D.; JONES, C.; STRAYER, W. T. Detecting botnets using command and control traffic. In: IEEE. *2009 Eighth IEEE International Symposium on Network Computing and Applications*. [S.l.], 2009. p. 156–162.

BAO, Z.; WANG, W.; LAN, Y. Using Passive DNS to Detect Malicious Domain Name. *ACM International Conference Proceeding Series*, 2019. Disponível em: <https://dl.acm.org/doi/abs/10.1145/3387168.3387236>.

BILGE, L.; KIRDA, E.; KRUEGEL, C.; BALDUZZI, M.; ANTIPOLIS, S. EXPOSURE : Finding Malicious Domains Using Passive DNS Analysis. *Ndss*, p. 1–17, 2011. ISSN 10949224. Disponível em: <http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:EXPOSURE+:+Finding+Malicious+Domains+Using+Passive+DNS+Anal>.

BREIMAN JEROME FRIEDMAN, R. A. O. C. J. S. L. *Classification and regression trees*. [S.l.]: CRC, 1984. (The Wadsworth statistics / probability series). ISBN 0-412-04841-8.

BRUCE, P.; BRUCE, A.; GEDECK, P.; SAFARI, a. O. M. C. *Practical Statistics for Data Scientists, 2nd Edition*. [S.l.: s.n.], 2020. OCLC: 1138943866. ISBN 978-1492072942.

CARVALHO, A.; FACELI, K.; LORENA, A.; GAMA, J. Inteligência artificial: uma abordagem de aprendizado de máquina. *Rio de Janeiro: LTC*, 2011.

CHEN, T.; GUESTRIN, C. XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, v. 13-17-Augu, p. 785–794, 2016.

CHEN, Z.; JIANG, F.; CHENG, Y.; GU, X.; LIU, W.; PENG, J. XGBoost Classifier for DDoS Attack Detection and Analysis in SDN-Based Cloud. *Proceedings - 2018 IEEE International Conference on Big Data and Smart Computing, BigComp 2018*, p. 251–256, 2018.

DONGES, N. *The Random Forest Algorithm*. Towards Data Science, 2018. Disponível em: <https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd>.

FALL, K. R.; STEVENS, W. R. *TCP/IP illustrated, volume 1: The protocols*. [S.l.]: addison-Wesley, 2011.

FAWCETT, T. ROC Graphs: Notes and Practical Considerations for Researchers BT - Tech Report HPL-2003-4, HP Laboratories. p. 1–38, 2004. Disponível em: <http://binf.gmu.edu/mmasso/ROC101.pdf>.

FERNÁNDEZ, A.; GARCÍA, S.; GALAR, M.; PRATI, R. C.; KRAWCZYK, B.; HERRERA, F. *Learning from Imbalanced Data Sets*. 1. ed. Cham: Springer International Publishing, 2018. XVIII, 377 p. ISSN 10414347. ISBN 978-3-319-98073-7. Disponível em: <https://www.springer.com/gp/book/9783319980737http://link.springer.com/10.1007/978-3-319-98074-4>.

HARRENSTIEN, K.; STAHL, M.; FEINLER, E. Rfc 952: Dod internet host table specification. *DDN Protocol Handbook, vol. Three*, p. 249–254, 1985.

KIDMOSE, E.; LANSING, E.; BRANDBYGE, S.; PEDERSEN, J. M. Detection of malicious and abusive domain names. *Proceedings - 2018 1st International Conference on Data Intelligence and Security, ICDIS 2018*, p. 49–56, 2018.

KOTSIANTIS, S. B.; ZAHARAKIS, I.; PINTELAS, P. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, v. 160, p. 3–24, 2007.

KULIKOVA, T.; SIDORINA, T.; SHCHERBAKOVA, T. *Spam and phishing in Q2 2020*. 2020. Disponível em: <https://securelist.com/spam-and-phishing-in-q2-2020/97987/>.

KUROSE, J. F.; ROSS, K. W. *Redes de Computadores e a Internet*. [S.l.: s.n.], 2013. 1–658 p. ISBN 9788543014432.

LISON, P.; MAVROEIDIS, V. Neural reputation models learned from passive DNS data. *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, v. 2018-Janua, n. June, p. 3662–3671, 2017.

LYNN, M. S. A Unique, Authoritative Root for the DNS. *The Internet Protocol Journal*, v. 4, n. 3, 2001. Disponível em: <https://www.icann.org/resources/pages/unique-authoritative-root-2012-02-25-en>.

M Mohri, A Rostamizadeh, A. T. *Foundations of Machine Learning*. London, England: The MIT Press, 2012. ISSN 21915776. ISBN 9780262018258.

MCGEE, M. *Google Wins 750+ Domains From Cybersquatter Who Wants 'Google' Trademark Canceled*.

MOCKAPETRIS, P. *Domain names - concepts and facilities*. [S.l.]: RFC Editor, 1987. RFC 1034. (Request for Comments, 1034).

MOCKAPETRIS, P. *Domain names - implementation and specification*. [S.l.]: RFC Editor, 1987. RFC 1035. (Request for Comments, 1035).

MONTGOMERY, D. C. *Introduction to Statistical Quality Control, Student Resource Manual - 4th Edition*. 4. ed. [S.l.: s.n.], 2002. ISBN 9780471318286,0471318280.

NOGUEIRA, F. *Bayesian Optimization: Open source constrained global optimization tool for Python*. 2014. Disponível em: <https://github.com/fmfn/BayesianOptimization>.

PERDISCI, R.; CORONA, I.; GIACINTO, G. Early detection of malicious flux networks via large-scale passive dns traffic analysis. *IEEE Transactions on Dependable and Secure Computing*, IEEE, v. 9, n. 5, p. 714–726, 2012.

RASCHKA, S. Model evaluation, model selection, and algorithm selection in machine learning. *CoRR*, abs/1811.12808, 2018. Disponível em: <http://arxiv.org/abs/1811.12808>.

REGISTRO.BR. *Domínios .br registrados até o momento*. 2019. <https://registro.br/estatisticas.html>. Accessed on April 01, 2019.

SAFAVIAN, S. R.; LANDGREBE, D. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, IEEE, v. 21, n. 3, p. 660–674, 1991.

SAITO, T.; REHMSMEIER, M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLoS ONE*, v. 10, n. 3, p. 1–21, 2015. ISSN 19326203. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4349800/>.

STAT, D. N. *The overall number of domains registered, by TLD type*. 2020. <https://domainnamestat.com/statistics/overview>. Accessed on January 15, 2019.

STEVENS, W. R. *TCP/IP illustrated vol. I: the protocols*. [S.l.]: Pearson Education India, 1994.

STURGEON, W. Net pioneer predicts overwhelming botnet surge. *ZDNet News, January*, v. 29, 2007.

SYMANTEC. Internet Security Threat Report VOLUME 21, February 2019. *Network Security*, v. 21, n. February, p. 61, 2019. ISSN 13534858. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S1353485805001947>.

TORABI, S.; BOUKHTOUTA, A.; ASSI, C.; DEBBABI, M. Detecting internet abuse by analyzing passive DNS traffic: A survey of implemented systems. *IEEE Communications Surveys and Tutorials*, IEEE, n. 4, p. 3389–3415. ISSN 1553877X.

TRAN, H.; DANG, C.; NGUYEN, H.; VO, P.; VU, T. Multi-confirmations and dns graph mining for malicious domain detection. In: SPRINGER. *Intelligent Computing-Proceedings of the Computing Conference*. 2019. v. 998, p. 639–653. ISBN 978-3-030-22867-5. Disponível em: <http://link.springer.com/10.1007/978-3-030-22868-2>.

URDAN, T. C. *Statistics in plain English, 3rd Edition*. [S.l.]: Routledge, 2010. 211 p. ISBN 9780415872911.

VAZ, A. L. *Como lidar com dados desbalanceados em problemas de classificação*. Medium, 2019. Disponível em: <https://medium.com/data-hackers/como-lidar-com-dados-desbalanceados-em-problemas-de-classifica\%C3\%A7\%C3\%A3o-17c4d4357ef9>.

VERGELIS, M.; SHCHERBAKOVA, T.; SIDORINA, T. *Spam and phishing in Q1 2019*. 2019. <https://securelist.com/spam-and-phishing-in-q1-2019/90795/>. Accessed on May 15, 2019.

WANG, Q.; LI, L.; JIANG, B.; LU, Z.; LIU, J.; JIAN, S. Malicious Domain Detection Based on K-means and SMOTE. In: . Springer, Cham, 2020. p. 468–481. ISBN 978-3-030-50416-8. Disponível em: <http://link.springer.com/10.1007/978-3-030-50417-5{_}.>

WEIMER, F. *Passive DNS Replication*. 2005.

WEISS, E. A. Biographies: Eloge: Arthur lee samuel (1901-90). *IEEE Annals of the History of Computing*, IEEE, v. 14, n. 3, p. 55–69, 1992.

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

TERMO DE REPRODUÇÃO XEROGRÁFICA

Autorizo a reprodução xerográfica do presente Trabalho de Conclusão, na íntegra ou em partes, para fins de pesquisa.

São José do Rio Preto, 19 / 02 / 2021

_____

Assinatura do autor