PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS
CENTRO DE CIÊNCIAS EXATAS, AMBIENTAIS E DE TECNOLOGIA
FACULDADE DE ENGENHARIA DE TELECOMUNICAÇÕES

BRUNO MATHEUS DOS SANTOS

MODELOS DE APRENDIZADO DE MÁQUINA PARA A IDENTIFICAÇÃO DE ATAQUES DISTRIBUÍDOS DE NEGAÇÃO DE SERVIÇO (DDOS) NO THREE-WAY HANDSHAKE

CAMPINAS
2020

BRUNO MATHEUS DOS SANTOS

MODELOS DE APRENDIZADO DE MÁQUINA PARA A IDENTIFICAÇÃO DE ATAQUES DISTRIBUÍDOS DE NEGAÇÃO DE SERVIÇO (DDOS) NO THREE-WAY HANDSHAKE

Trabalho de conclusão de curso apresentada como exigência para obtenção do Título de Bacharelado em Engenharia de Telecomunicações do Centro de Ciências Exatas, Ambientais e de Tecnologia, da Pontifícia Universidade Católica de Campinas.

Orientador: Prof. Me. Ralph Robert Heinrich

PUC- CAMPINAS

2020

Pontifícia Universidade Católica de Campinas
Centro de Ciências Exatas, Ambientais e de Tecnologia Faculdade de Engenharia de Telecomunicações

MODELOS DE APRENDIZADO DE MÁQUINA PARA A IDENTIFICAÇÃO DE ATAQUES DISTRIBUÍDOS DE NEGAÇÃO DE SERVIÇO (DDOS) NO THREE-WAY HANDSHAKE

Autor: SANTOS, Bruno Matheus dos.

Título: Modelos de aprendizado de máquina para a identificação de ataques distribuídos de negação de serviço (DDoS) no three-way handshake.

Trabalho de conclusão de curso apresentada como exigência para obtenção do Título de Bacharelado em Engenharia de Telecomunicações do Centro de Ciências Exatas, Ambientais e de Tecnologia, da Pontifícia Universidade Católica de Campinas.

BANCA EXAMINADORA

Orientador: Prof. Me. Ralph Robert Heinrich

Campinas, 24 de novembro de 2020

AGRADECIMENTOS

À minha família, pela paciência que tiveram durante os últimos meses.
Ao professor orientador Ralph Robert Heinrich, por sua paciência durante todo o processo de definição do tema do trabalho e mediação durante o processo de desenvolvimento deste trabalho.
Ao amigo Felipe Gabriel Bosada, pelas conversas iniciais sobre aprendizado de máquina e pelos constantes feedbacks durante o processo.
Aos professores Daniele Cristina Uchoa Maia Rodrigues e Edmar Roberto Santana de Rezende, pelas conversas iniciais e direcionamento das ferramentas utilizadas e métodos de validação.
Aos colegas Fábio Capuano Souza e Israel Campiotti, por dedicarem um pouco de seu tempo para tirar algumas dúvidas pontuais sobre desenvolvimento de software.
E a todos outros amigos que tiveram muita paciência durante este tempo em que estive ausente.

“Ensinar não é transferir conhecimento, mas criar as possibilidades para a sua própria produção ou a sua construção.”

Paulo Freire

RESUMO

Em 2019, foi estimado que a inoperabilidade de um sistema interno causada pelo ataque de negação de serviço, pode gerar, em uma hora, prejuízos financeiros de US$ 300 mil a US$ 1 milhão para as empresas. Esta interrupção gera prejuízos financeiros, operacionais e reputacionais. O Brasil está entre os maiores emissores de ataques DDoS (*Distributed Denial of Service*, em inglês), figurando a quarta posição, atrás da China, Estados Unidos e Coreia do Sul que ocupam a primeira, segunda e terceira posição, respectivamente. O ataque de negação de serviço SYN *flood* consome os recursos dos servidores, no qual o requisitante envia diversos pacotes SYN para estabelecer uma conexão, porém ignora a confirmação do recebimento SYN-ACK para o servidor no *three-way handshake*. A baixa complexidade de execução deste ataque faz com que aumente a demanda do mercado em fornecer soluções eficazes que, mesmo não eliminando o risco, sejam capazes de reduzir os impactos. A criação de uma solução definitiva capaz de prever e detectar a finalidade dos pacotes é barrada pela constante mudança de características nos ataques. O aprendizado de máquina é uma ferramenta que pode auxiliar na predição e detecção destes ataques, porém a utilização de inteligência artificial necessita de um grande volume de dados, alimentando modelos que serão treinados. Desafios na utilização dos *datasets* incluem o número de *features* do *dataset* que descrevem o comportamento da rede na troca de pacotes, bem como o tamanho do *dataset*, que sendo consideravelmente grande, torna o tempo de treinamento do modelo muito alto, porém uma vez que o modelo está treinado, a detecção do ataque é realizada em menor tempo. Este trabalho apresenta um modelo treinado de aprendizado de máquina para a detecção de ataques distribuídos de negação de serviço, utilizando o *dataset* fornecido pela Universidade de Brunswick (Canadá). Os dados de tráfego de rede foram analisados utilizando modelos estatísticos aplicados às técnicas de aprendizado de máquina através do editor Google Colaboratory e da biblioteca para aprendizado de máquina Scikit-Learn. A seleção das *features* mais importantes, o treinamento, o teste e a validação do modelo foram feitos com o algoritmo *Random forest,* utilizando o método *StratifiedK-Fold* para separação das amostras de treino e teste, e as medidas de desempenho utilizadas foram Acurácia, Precisão, Revocação e Pontuação F1. Além da assertividade geral relatada nos resultados, este trabalho apresenta uma análise do impacto das *features* mais importantes na detecção do ataque. Para trabalhos futuros é possível alterar a saída do algoritmo para indicar, além do resultado, o tipo de ataque que foi detectado pelo modelo criado.

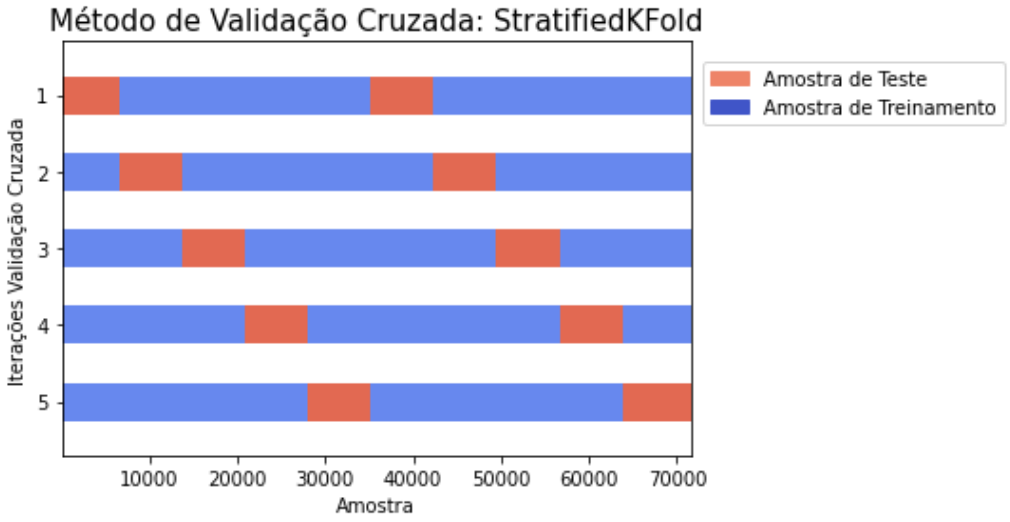
Palavras-Chave: Árvore de decisão. CICDDoS2019. DDoS. *KDD*. *Machine learning.*

ABSTRACT

In 2019, it was estimated that the inoperability of an internal system caused by the denial of service attack can generate, in an hour, financial losses of US$ 300,000 to US$ 1 million to companies. The interruption generates financial, operational and reputational losses. Brazil is among the largest emitters of DDoS (DistributedDenial of Service) attacks,with the fourthposition, behind China,a the United States and South Korea, which occupy the first, second and third positions, respectively. The SYN flood denial-of-service attack consumes the resources of the servers, in which the requester sends several SYN packets to establish a connection, but ignores the confirmation of the SYN-ACK receipt to the server on the three-way handshake. The low complexity of executing this attack increases market demand in providing effective solutions that, even if not eliminating risk, are able to reduce impacts. The creation of a definitive solution capable of predicting and detecting the purpose of packets is barred by the constant change of characteristics in attacks. Machine learning is a tool that can help in predicting and detecting these attacks, but the use of artificial intelligence requires a large volume of data, feeding models that will be trained. Challenges in using datasets include the number of dataset features that describe the behavior of the network in packet exchange, as well as the size of the dataset, which being considerably large, makes the model training time too high, but once the model is trained, attack detection is performed in a shorter time. This paper presents a trained machine learning model for detecting distributed denial of service attacks, using the dataset provided by the University of Brunswick (Canada). Network traffic data was analyzed using statistical models applied to machine learning techniques through the Google Colaboratory editor and scikit-learn machine learning library. The selection of the most important features, the training, testing and validation of the model were made with the Random forest algorithm, using the StratifiedK-Fold method for separation of training and tests amples, and the performance measurements used were Accuracy, Precision, Recall and F1 Score. In addition to the general assertiveness reported in the results, this paper presents an analysis of the impact of the most important features in the detection of the attack. For future jobs it is possible to change the output of the algorithm to indicate, in addition to the result, the type of attack that was detected by the created model.

Keywords: CICDDoS2019. DDoS. Decision tree. KDD. Machine learning.

LISTA DE FIGURAS



LISTA DE TABELAS E QUADROS

LISTA DE ABREVIATURAS E SIGLAS

Sigla	Descrição
CPU	*Central Processing Unit* (Unidade central de processamento)
DDoS	*Distributed Denial of Service* (Negação de serviço distribuído)
GPU	*Graphics processing unit* (Unidade de processamento gráfico)
IDS	*Intrusion Detection Systems* (Sistemas de detecção de intrusão)
IP	*Internet Protocol* (Protocolo Internet)
KDD	*Knowledge Discovery in Databases* (Descoberta de conhecimento em bases de dados)
OSI	*Open Systems Interconnection* (Modelo OSI)
SVM	*Support Vector Machine* (Máquina de vetor de suporte)

SUMÁRIO

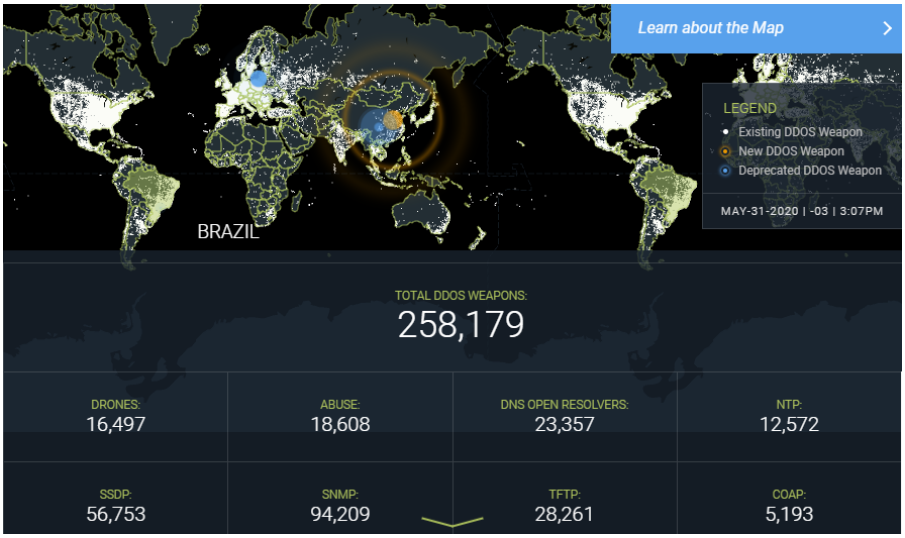
1 INTRODUÇÃO

Os mais diversos ataques ordenados às redes de computadores têm crescido globalmente de modo sofisticado com o avanço das tecnologias de comunicação de dados, com a chegada do 5G, e com a mudança no comportamento da sociedade, como a adoção do trabalho remoto, fazendo com que cresça, ao mesmo passo, a urgência para reduzir os efeitos causados por criminosos que buscam explorar as deficiências de companhias e corporações dos mais diversos portes e segmentos.

Grandes empresas e organizações como Anatel, Netflix, DYN, Marvel, BBC, DYN, Github, entre outras, já experienciaram estes ataques [1] que podem acarretar prejuízos financeiros, operacionais e reputacionais. Em 2019, foi estimado que a inoperabilidade do sistema causada pelo ataque de negação de serviço, pode gerar, em uma hora, prejuízos financeiros de US$ 300 mil a US$ 1 milhão para uma empresa [2].

Os ataques também são amplamente direcionados às empresas de telecomunicações. 85% de trezentas e vinte e cinco empresas de TI (Tecnologia da Informação) entrevistadas pelo Instituto Ponemon, nos Estados Unidos, acreditam que o ataque DDoS está crescendo e tende a continuar [3]. O mapa de ameaças DDoS mantido pela empresa A10, aponta o Brasil como um dos maiores emissores de ataques DDoS (*Distributed Denial of Service*, em inglês), figurando a quarta posição, atrás da China, Estados Unidos e Coreia do Sul que ocupam a primeira, segunda e terceira posição, respectivamente [4].

Figura 1. Mapa de inteligência de ameaças DDoS em 31 de maio de 2020 com destaque para o Brasil.

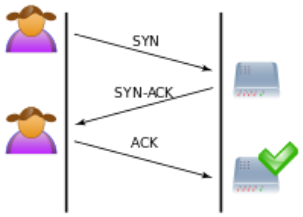


Fonte: A10 Networks

O ataque de negação de serviço SYN *flood* consome os recursos dos servidores ao enviar diversos pacotes SYN para estabelecer uma conexão, porém ignora a confirmação do recebimento SYN-ACK para o servidor no *three-way handshake*.

A Figura 2 apresenta o cenário onde o requisitante envia um pacote SYN para o servidor, a fim de estabelecer uma conexão. O servidor retorna um pacote SYN-ACK indicando que reconheceu a origem do pacote. Por fim, o cliente envia um pacote ACK para o servidor, encerrando o processo de autenticação.

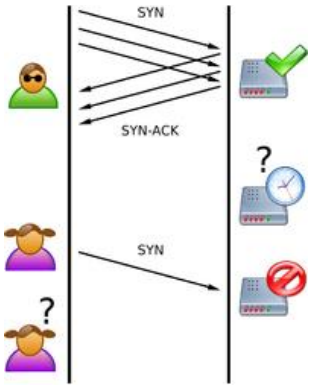
Figura 2. Conexão normal no processo de aperto de mãos em três etapas.



Fonte: Wikipédia [5]

Na Figura 3, o servidor é bombardeado com múltiplos pacotes SYN, e tenta retornar o SYN-ACK para estabelecer conexão. Porém, o responsável pelo ataque não retorna o ACK. Ao esperar pela confirmação do requisitante, enquanto continua recebendo mais solicitações SYN, o servidor é obrigado a negar o serviço aos clientes que tentariam acessá-lo legitimamente, seja por falta de banda ou por medida de segurança após o ataque ser identificado.

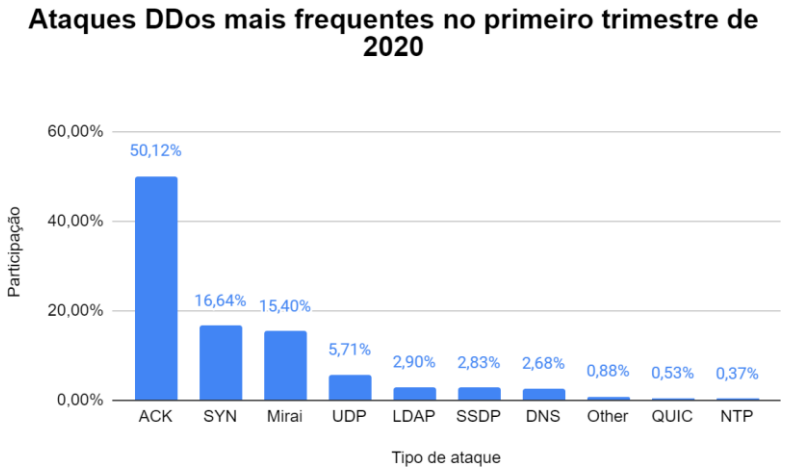
Figura 3. Conexão com SYN *flood*, onde o servidor nega o serviço ao cliente legítimo.



Fonte: Wikipédia

Até o primeiro trimestre de 2020, os ataques mais comuns de DDoS foram ACK e SYN, seguidos por Mirai, UDO e LDAP [6], como visto na Figura 4.

Figura 4. Percentual de ataques DDoS no primeiro trimestre de 2020.



Fonte: Cloudflare

A baixa complexidade de execução deste ataque faz com que aumente a demanda do mercado em fornecer soluções eficazes que, mesmo não eliminando o risco, sejam capazes de reduzir os impactos. Firewalls, aumento de banda e contratação de empresas especializadas em segurança de rede, são algumas formas de lidar com o problema. A criação de uma solução definitiva capaz de prever e detectar fraudes é barrada pela constante mudança de características nos ataques.

O aprendizado de máquina é uma ferramenta que pode auxiliar na predição e detecção destes ataques, porém a utilização de inteligência artificial necessita de um grande volume de dados, chamados *datasets,* alimentando

modelos que serão treinados. Universidades, competições e empresas de segurança disponibilizam, de tempos em tempos, estes *datasets*, contendo informações de tráfegos medidos ou simulados em ambiente controlado.

Desafios na utilização dos *datasets* incluem o número de *features* do *dataset* que descrevem o comportamento da rede na troca de pacotes, bem como o tamanho do *dataset*, que majoritariamente é grande, tornando o tempo de treinamento do modelo muito alto, embora uma vez que o modelo esteja treinado, a detecção do ataque é realizada instantaneamente.

A proposta deste trabalho é apresentar modelos treinados de aprendizado de máquina para a detecção de ataques de negação de serviço no *three-way handshake*, utilizando o *dataset* fornecido pela Universidade de Brunswick [7].

2 OBJETIVO

2.1 Objetivos gerais

Treinar um modelo de aprendizado de máquina capaz de identificar se o pacote de rede analisado é benigno ou maligno, ou seja, se é fruto de um ataque de negação de serviço distribuído (DDoS) do tipo SYN.

2.2 Objetivos específicos

- Utilizar amostra balanceada, para realização de testes do algoritmo, sendo 50% de benigno e 50% de ataque.
- Analisar as *features* que tenham pelo menos 5% de relevância na determinação dos resultados de treino do *dataset* de referência (CICDDoS2019).
- Validar o modelo treinado utilizando validação cruzada.
- Utilizar apenas *softwares* e plataformas de código livre para criação do modelo.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Modelo OSI

O Modelo OSI é um modelo de rede de computador referência da ISO dividido em camadas de funções, criado em 1971 e formalizado em 1983, com objetivo de ser um padrão, para protocolos de comunicação entre os mais diversos sistemas em uma rede local, garantindo a comunicação entre dois sistemas computacionais [8].

Quadro 1. Exemplos e protocolos por camada do modelo OSI.

Camada		**Exemplos**	***Suite* TCP/IP**
7	Camada de aplicação	HL7, Modbus	HTTP, SMTP, SNMP, FTP, NFS, NTP, BOOTP, DHCP, RMON, POP3, IMAP, TELNET, DNS, LDAP, SSL
6	Camada de apresentação	TDI, ASCII, EBCDIC, MIDI, MPEG	XDR, TLS
5	Camada de sessão	Named Pipes, NetBIOS, SIP, SAP, SDP	Estabelecimento da sessão TCP
4	Camada de transporte	NetBEUI	TCP, TFTP |UDP, RTP, SCTP, SSL, DCCP, SCTP, RSVP
3	Camada de rede	NetBEUI, Q.931	IP, ICMP, IPsec, RIP, OSPF, BGP, NAT, Roteador
2	Camada de enlace	Ethernet, Token Ring, FDDI, PPP, HDLC, Q.921, Frame Relay, Bridge, ATM, Fibre Channel	MTP-2, ARP
1	Camada física	Alicate, RS-232, V.35, V.34, Q.911, T1, E1, CDDI, 10BASE-T,100BASE-TX, ISDN, Repetidor, SONET, DSL, 802.11	N/A

Fonte: Wikipédia

A camada física é a primeira camada do modelo OSI, sendo responsável pela transmissão dos dados pelos meios físicos como cabos de rede e dispositivos como hubs. A segunda camada é a camada de enlace de dados, que garante que todos os pacotes de informações sejam transmitidos sem erros. Ela garante que o protocolo apropriado seja atribuído ao dado. As principais funções da camada de enlace de enlace são lidar com erros de transmissão, regular o fluxo de dados e fornecer uma interface bem definida para a camada de rede. Quando os dados são transferidos para a camada de rede, ela usa temporizadores e números de sequência para verificar se há erros e garantir que todos os dados sejam recebidos com sucesso. A camada de enlace possui diferentes serviços e protocolos para

completar suas tarefas. Os protocolos são as regras necessárias para passar dados com êxito para a próxima camada [9]. A terceira camada é a camada de rede, responsável pelo controle da operação das sub redes. Esta camada determina o caminho físico que o dado deve percorrer, baseado nas condições de rede, prioridade de serviço, dentre outros fatores como rota, controle de tráfego, fragmentação e remontagem de quadro e mapeamento de endereço lógico. A quarta camada é a camada de transporte, que garante que as mensagens sejam entregues sem erro, na sequência correta e sem perda ou duplicação da informação para as camadas superiores [10]. A camada de sessão é a quinta camada do modelo OSI e tem a função de estabelecer, sincronizar e encerrar conexões entre *hosts*. A camada seguinte é a de apresentação, responsável por traduzir os dados recebidos pela camada de sessão para a camada de aplicação. A camada de apresentação converte códigos para caracteres e trabalha com a criptografia dos dados, caso seja necessário. A sétima e última cada é a camada de aplicação, responsável por consumir os dados transmitidos. É nesta camada que a interação homem-máquina está presente.

3.2 Ataque distribuído de negação de serviço (DDoS) e ataque SYN *flood*

Um ataque de negação de serviço, é uma tentativa de tornar os recursos de um sistema indisponíveis para os seus utilizadores. Alvos típicos são servidores web, e o ataque procura tornar as páginas hospedadas indisponíveis na rede. Não se trata de uma invasão do sistema, mas sim da sua invalidação por sobrecarga [11]. SYN *flood* ou ataque SYN é uma forma de ataque de negação de serviço em sistemas computadorizados, na qual o atacante envia uma sequência de requisições SYN para um sistema-alvo visando uma sobrecarga direta na camada de transporte e indireta na camada de aplicação do modelo OSI [5].

3.3 *Dataset* CICDDoS2019

O *dataset* CICDDoS2019 contém registros benignos e ataques DDoS comuns e atualizados, capturados em ambiente controlado e simulando dados do mundo real (que são capturados por softwares de monitoramento de pacotes de rede). Ele também inclui os resultados da análise de tráfego de rede usando CICFlowMeter-V3 com fluxos rotulados com base no carimbo de data e hora, IPs de origem e destino, portas de origem e destino, protocolos e ataque, entre outros [7].

3.4 Machine learning

O aprendizado de máquina ou aprendizagem de máquina é uma subárea da inteligência artificial [12]. O aprendizado supervisionado é a tarefa de aprendizado de máquina de aprender uma função que mapeia uma entrada para uma saída com base em pares de entrada-saída pré-definidos, ou seja, o modelo de *machine learning* é treinado já conhecendo os resultados do *dataset* de origem, onde o resultado é apresentado na coluna chamada *Label*. O aprendizado não supervisionado é um ramo do machine learning que aprende com dados de teste que não foram rotulados, classificados ou categorizados previamente.

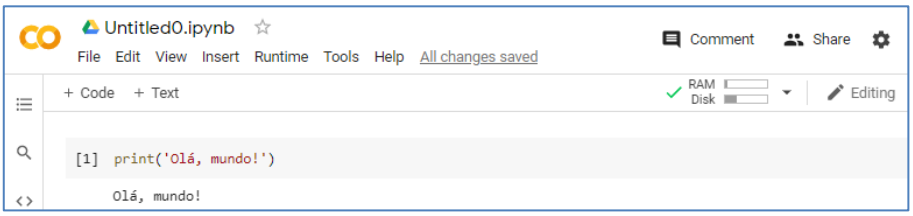
4 FERRAMENTAS COMPUTACIONAIS E MÉTODOS

4.1 Ferramentas computacionais

4.1.1 Google Colaboratory

O Google Colaboratory, ou Colab, é um editor de código online, sendo necessário apenas um navegador e conexão à internet para editar e executar códigos em Python, fornecendo CPU e GPU gratuitamente. Ele funciona como um ambiente de notebooks Jupyter disponibilizado como serviço em nuvem gratuito, hospedado pelo Google [13].

Figura 5. Interface do Google Colab.



4.1.2 Scikit-Learn

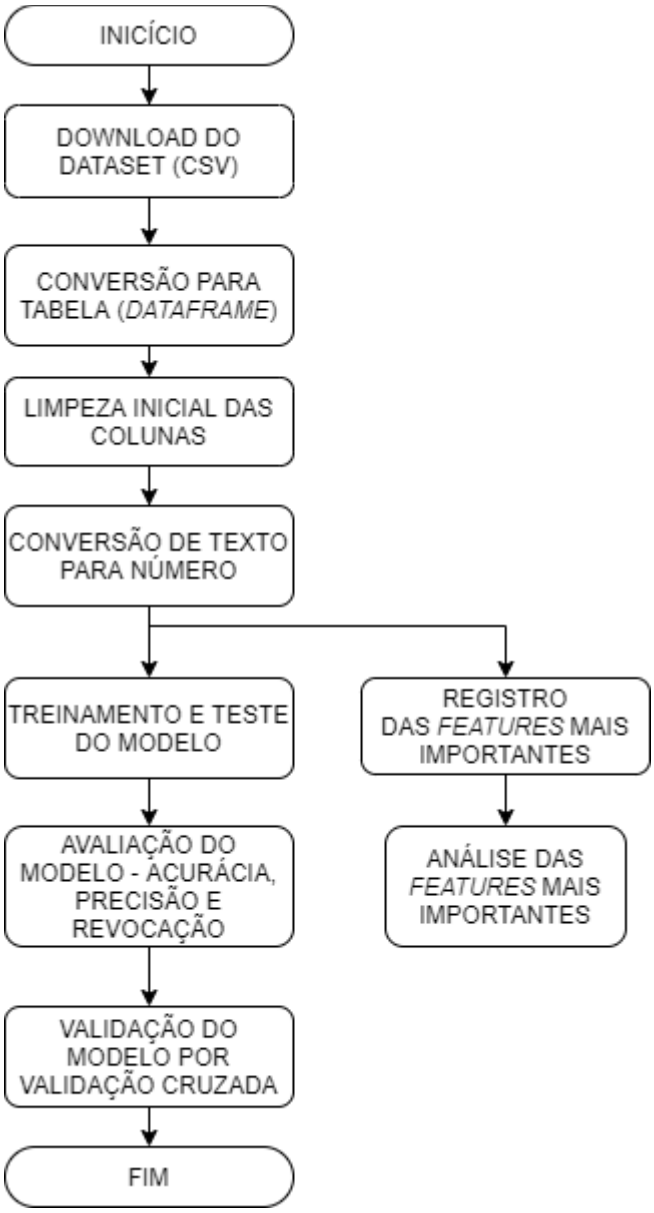
A scikit-learn é uma biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python [3]. Ela inclui diversos algoritmos de classificação, regressão e agrupamento incluindo máquinas de vetores de suporte, florestas aleatórias, *gradient boosting*, k-means e DBSCAN, e é projetada para interagir com as bibliotecas Python numéricas e científicas NumPy e SciPy, respectivamente [14].

Figura 6. Interface do website da Scikit-Learn.

4.2 Métodos

Esta seção apresenta as etapas realizadas durante a elaboração do trabalho, utilizando a técnica *KDD,* desde a coleta dos dados até os métodos utilizados para validação do modelo treinado. A Figura 7 apresenta o fluxograma desde o processo de obtenção dos dados até a validação do modelo criado.

Figura 7. Fluxograma das etapas desde a coleta dos dados até a validação do modelo criado



4.2.1 Coleta de dados

O *dataset* para a realização deste trabalho foi fornecido pela Universidade de Brunswick do Canadá, possuindo 4.320.541 (quatro milhões, trezentos e vinte mil, quinhentos e quarenta e um) registros, sendo 4.284.751 (quatro milhões, duzentos e oitenta e quatro mil, setecentos e cinquenta e um) registros de ataque SYN (99,17%) e 35.790 (trinta e cinco mil, setecentos e noventa) registros de tráfego benigno (0,83%). O *dataset* original possui 88 (oitenta e oito) colunas, sendo 87 (oitenta e sete) *features* e uma *Label*.

As *features* possuem diferentes formatos de representação, sendo número inteiros (*integer*), número reais (*float*) e componentes de texto (*string*), aleatoriamente dispersados.

Quadro 2. Estrutura resumida do *dataset* original

	Unnamed: 0	Flow ID	...	Inbound	Label
0	445444	172.16.0.5-192.168.50.4-9429-9429-6	...	1	Syn
1	113842	172.16.0.5-192.168.50.4-60224-60224-6	...	1	Syn
2	176377	172.16.0.5-192.168.50.4-33827-11746-6	...	0	Syn
3	24777	172.16.0.5-192.168.50.4-33828-1431-6	...	1	Syn
4	85100	172.16.0.5-192.168.50.4-5311-5311-6	...	1	Syn
...	...	...	...	...	...
4320536	317398	192.168.50.9-31.13.80.36-44304-443-6	...	0	BENIGN
4320537	286894	192.168.50.8-8.8.8.8-60481-53-17	...	0	BENIGN
4320538	2665	192.168.50.9-31.13.80.12-46518-443-6	...	1	BENIGN
4320539	2666	192.168.50.9-31.13.80.12-46518-443-6	...	0	BENIGN
4320540	286260	192.168.50.9-31.13.80.12-46518-443-6	...	1	BENIGN

[4320541 rows x 88 columns]

4.2.2 Preparação dos dados

Para otimizar o tratamento dos dados, foi preciso realizar uma filtragem inicial das informações relevantes. O *dataset* fornecido foi gerado através de um programa de computador chamado CICFlowMeter [15], que anexa seis colunas (*FlowID, SourceIP, DestinationIP, SourcePort, DestinationPort,* e *Protocol*) ao *dataset* original gerado pelo analisador de tráfego.

Para que o modelo fosse treinado apenas com as colunas originais, as seis colunas inseridas pelo CICFlowMeter foram eliminadas, incluindo as colunas "*Unnamed*: 0" e "*Timestamp*" que representam o identificador único da linha registrada e a marcação de data e hora em que o registro foi realizado, respectivamente.

Além do processo inicial de limpeza, a preparação dos dados envolve a seleção da amostragem que será utilizada, incluindo o número de amostras que serão testadas.

Para garantir que o modelo fosse treinado com uma amostra balanceada, foram utilizados 35.790 registros de ataque SYN (50%) e 35.790 registros de tráfego benigno (50%), totalizando 71.580 linhas (100%).

Quadro 3. Estrutura resumida da amostra utilizada no experimento

	Flow Duration	Total Fwd Packets	...	Inbound	Label
0	123	4	...	0	BENIGN
1	85281	17	...	0	BENIGN
2	2772175	3	...	0	BENIGN
3	245341	16	...	0	BENIGN
4	5506971	3	...	0	BENIGN
...	...	...	...	...	...
71575	84	2	...	1	Syn
71576	135	2	...	1	Syn
71577	54	2	...	1	Syn
71578	1	2	...	1	Syn
71579	1	2	...	1	Syn

[71580 rows x 80 columns]

Após a seleção da amostra, as *strings* de todas as *features* são convertidas em número, como visto abaixo no Quadro 4, para que o modelo possa ser treinado.

Quadro 4. Estrutura resumida da amostra codificada (Texto convertido para número)

	Flow Duration	Total Fwd Packets	...	Inbound	Label
0	3999	154	...	0	0
1	19468	70	...	0	0
2	8811	128	...	0	0
3	8006	60	...	0	0
4	14421	128	...	0	0
...	...	...	...	...	...
71575	19387	88	...	1	1
71576	4272	88	...	1	1
71577	13934	88	...	1	1
71578	1	88	...	1	1
71579	1	88	...	1	1

[71580 rows x 80 columns]

4.2.3 Escolha do modelo

O modelo escolhido foi de classificação pois a proposta deste trabalho é indicar se o tráfego foi benigno ou maligno, logo uma classificação binária. O aprendizado é considerado supervisionado, pois o modelo aprende a partir da classificação que consta na coluna “*Label*” do *dataset*.

O classificador escolhido foi o *Random forest*, também chamado de Florestas aleatórias, que é um método de aprendizado conjunto para classificação, regressão e outras tarefas que operam construindo árvores de decisão no momento

do treinamento do modelo [16]. O número de árvores utilizadas é declarado no hiper parâmetro ***n_estimators***.

O *Random forest* foi utilizado para:

1. Identificar as *features* mais importantes.
2. Treinar e testar o modelo criado.
3. Validar o modelo criado.

4.2.4 *Features* mais importantes

As *features* mais importantes foram identificadas através do algoritmo *Random forest*, onde foram consideradas importantes, as *features* que tivessem ao menos 5% de relevância para a determinação do resultado na classificação do pacote.

Foram utilizados 250 estimadores para a realização da identificação de *features* mais importantes.

A seção 5.1 apresenta os resultados das *features* mais importantes, bem como a análise de cada *feature* em relação ao *Label*, e como a detecção do ataque foi influenciada por estas *features*.

4.2.5 Treinamento e teste do modelo

O *Random forest* também foi utilizado para treinar e testar o modelo criado, utilizando 250 estimadores. A amostra foi separada em 80/20, sendo 80% da amostra inicial reservada para o treinamento, e 20% para o teste (predição).

4.2.6 Avaliação do modelo

As medidas de desempenho são visualizadas através da matriz de confusão, que permite mostrar o quanto a predição do modelo de aprendizado de máquina utilizado foi assertiva.

Os parâmetros das métricas são:

- Verdadeiro Positivo (VP), onde o modelo previu corretamente a classe positiva como sendo positiva;
- Verdadeiro Negativo (VN), onde o modelo previu corretamente a classe negativa como sendo negativa;
- Falso Positivo (FP), onde o modelo previu incorretamente a classe negativa como sendo positiva; e
- Falso negativo (FN), onde o modelo previu incorretamente a classe positiva como sendo negativa.

Acurácia

A acurácia é a assertividade geral, indicando quantos dados foram corretamente preditos. Entretanto a acurácia não indica a proporção de acerto entre os verdadeiros positivos e os verdadeiros negativos. [17]

Equação 1. Equação da métrica de avaliação: Acurácia

$$Acurácia = \frac{Número\ de\ predições\ corretas}{Número\ total\ de\ predições} = \frac{VP + VN}{VP + VN + FP + FN}$$

Precisão

A precisão indica a proporção de identificações positivas que estão corretas. Ela é a proporção de verdadeiros positivos em relação à soma de verdadeiros positivos e falsos positivos. Com a precisão é possível observar quantos valores positivos foram preditos erroneamente. Se não houver falsos positivos, o modelo terá 100% de precisão. Quanto maior for o número de falsos positivos na predição, pior será a precisão do modelo.

Equação 2. Equação da métrica de avaliação: Precisão

$$Precisão = \frac{Verdadeiro\ Positivo}{Verdadeiro\ Positivo\ +\ Falso\ Positivo}$$

Revocação

A revocação indica a proporção de positivos reais que foi identificada corretamente, sendo a proporção de verdadeiros positivos em relação à soma de verdadeiros positivos e falsos negativos. Com a revocação é possível observar o número de falsos negativos e quando é necessário reduzi-lo.

Equação 3. Equação da métrica de avaliação: Revocação

$$Revocação = \frac{Verdadeiro\ Positivo}{Verdadeiro\ Positivo\ +\ Falso\ Negativo}$$

Pontuação F1

Indica a relação entre a Precisão e a Revocação.

Equação 4. Equação da métrica de avaliação: Pontuação F1

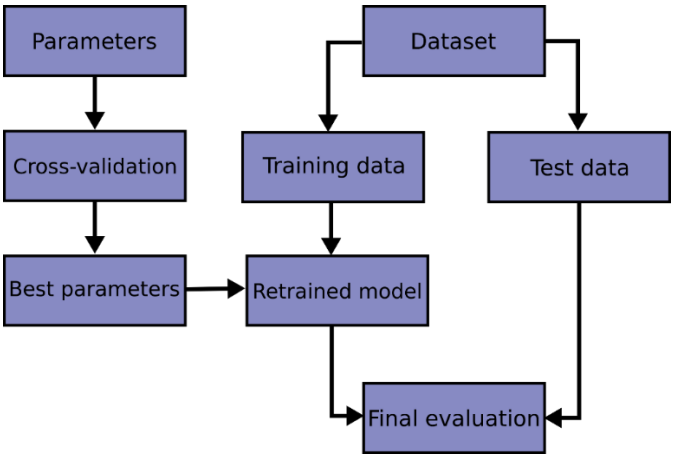
$$Pontuação\ F1 = \frac{2}{\frac{1}{Precisão} + \frac{1}{Revocação}}$$

4.2.7 Validação do modelo

O modelo criado foi validado através do algoritmo *Random forest*, após o *dataset* ser divido pelo algoritmo *K-Fold*, responsável pela divisão da amostra em teste e treino. Este método fornece índices de treinamento e teste para dividir os dados em subconjuntos de treinamento e teste. O *dataset* é dividido em k subconjuntos. Cada subconjunto é então usado uma vez como validação, enquanto os k-1 subconjuntos restantes formam o conjunto de treinamento.

Assim como o *Random forest*, utilizado para treinamento e teste do modelo, este método também fornece medidas de desempenho.

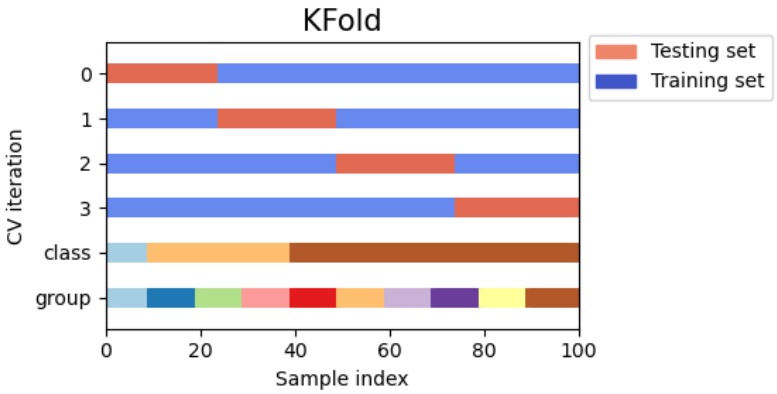
Figura 8. Diagrama de funcionamento da validação cruzada



Fonte: Scikit-Learn

Apesar do método *K-Fold* ser eficiente, as amostras de testes poderiam ter sido selecionadas em partes que contiveram apenas um único resultado (benigno ou ataque SYN).

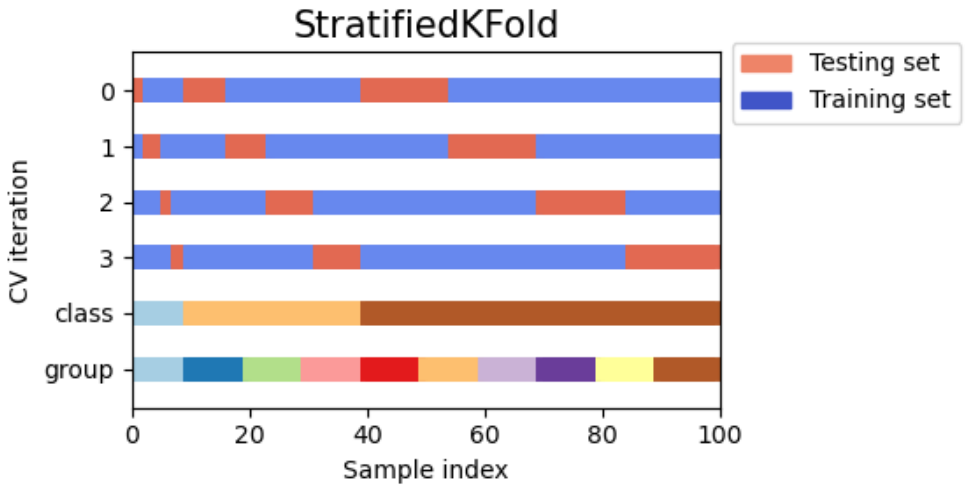
Figura 9. Diagrama de funcionamento do método K-Fold para validação cruzada



Fonte: Scikit-Learn

De modo a evitar esta ocorrência, o método de separação de amostra utilizado neste trabalho foi o *StratiefiedK-Fold*, que utiliza pequenas divisões para seleção da amostra de teste, preservando a proporção inicial do *dataset* e das classes do problema, conforme visto na Figura 10.

Figura 10. Diagrama de funcionamento do método *StratifiedK-Fold* para validação cruzada onde a amostra de teste é distribuída aleatoriamente no conjunto de dados.



Fonte: Scikit-Learn

5 RESULTADOS

5.1 *Features* mais importantes

Figura 11. *Features* com grau de importância acima de 5% para 71.580 registros e 79 *features*.

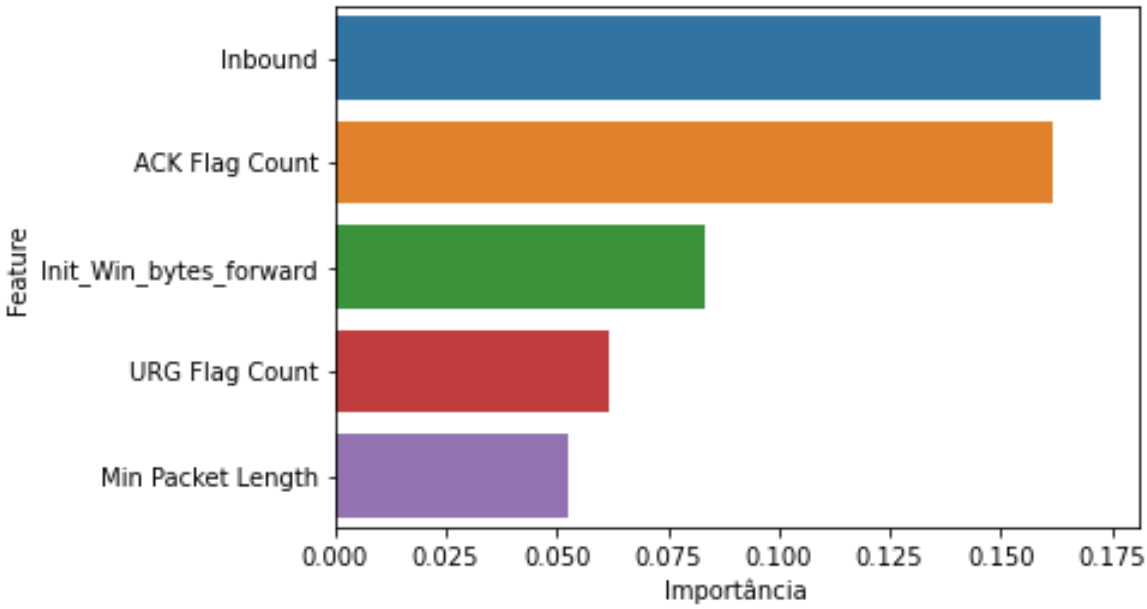
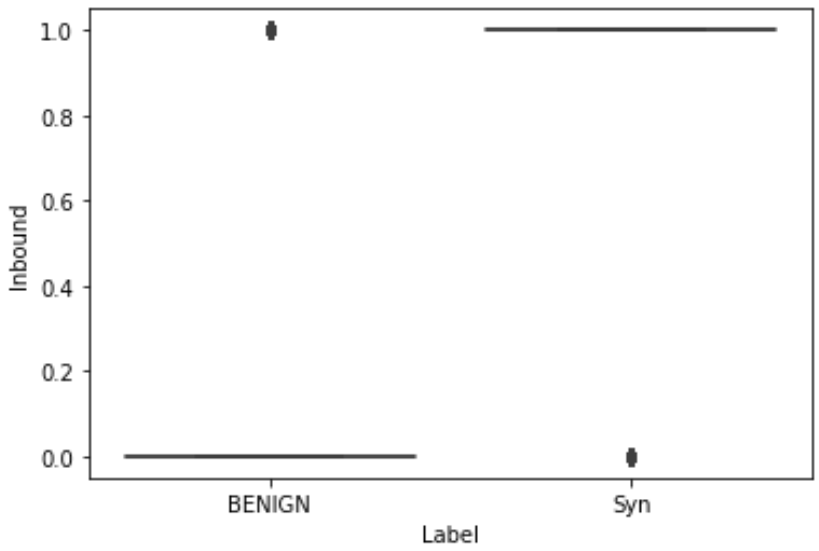


Tabela 1. Grau de relevância das *features* mais importantes na determinação do resultado

feature	importance
Inbound	0.172283
ACK Flag Count	0.161916
Init_Win_bytes_forward	0.083366
URG Flag Count	0.061627
Min Packet Length	0.052605

Figura 12. Diagrama de caixa da *feature* **Inbound** e análise dos dados

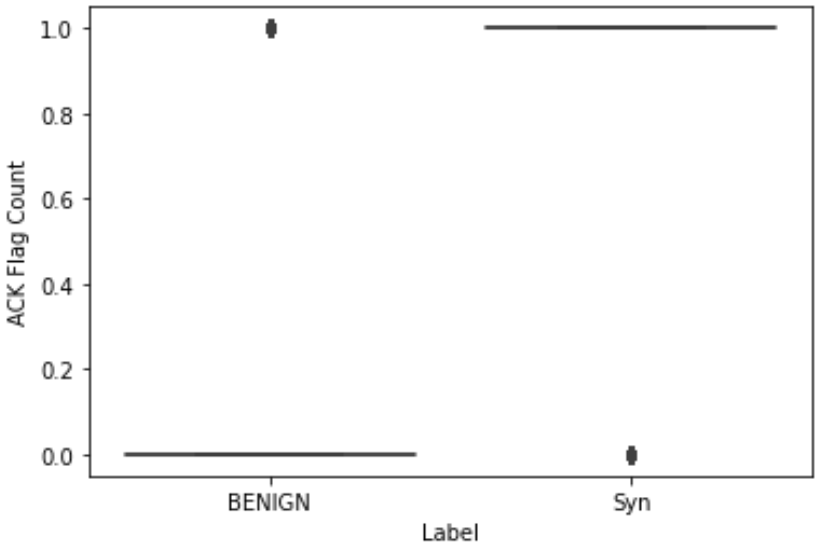



```
Feature: Inbound

 BENIGN
   Mínimo: 0.00
   Limite inferior: 0.00
   1º Quartil (25%): 0.00
   Mediana: 0.00
   3º Quartil (75%): 0.00
   Limite superior: 1.00
   Máximo: 1.00
   Média: 0.21

 Syn
   Mínimo: 0.00
   Limite inferior: 0.00
   1º Quartil (25%): 1.00
   Mediana: 1.00
   3º Quartil (75%): 1.00
   Limite superior: 1.00
   Máximo: 1.00
   Média: 1.00
```

Figura 13. Diagrama de caixa da *feature* **ACK Flag Count** e análise dos dados

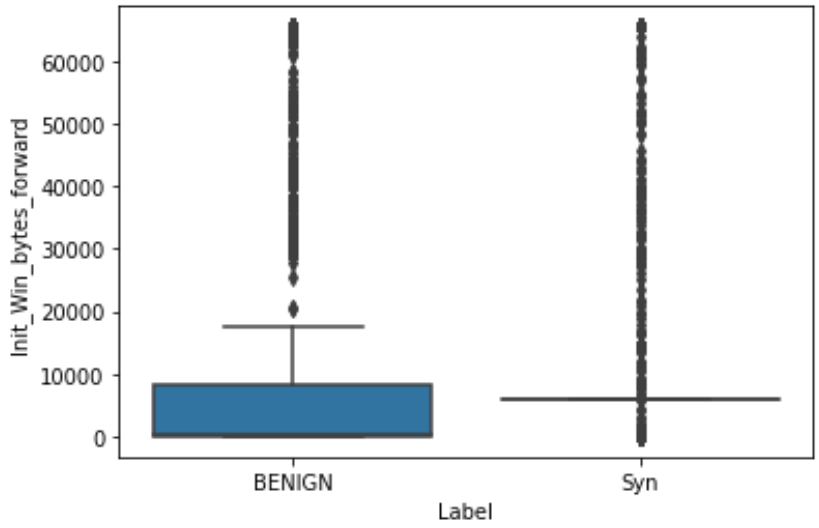



```
Feature: ACK Flag Count

 BENIGN
   Mínimo: 0.00
   Limite inferior: 0.00
   1º Quartil (25%): 0.00
   Mediana: 0.00
   3º Quartil (75%): 0.00
   Limite superior: 1.00
   Máximo: 1.00
   Média: 0.21

 Syn
   Mínimo: 0.00
   Limite inferior: 0.00
   1º Quartil (25%): 1.00
   Mediana: 1.00
   3º Quartil (75%): 1.00
   Limite superior: 1.00
   Máximo: 1.00
   Média: 1.00
```

Figura 14. Diagrama de caixa da *feature* **Init_Win_bytes_forward** e análise dos dados

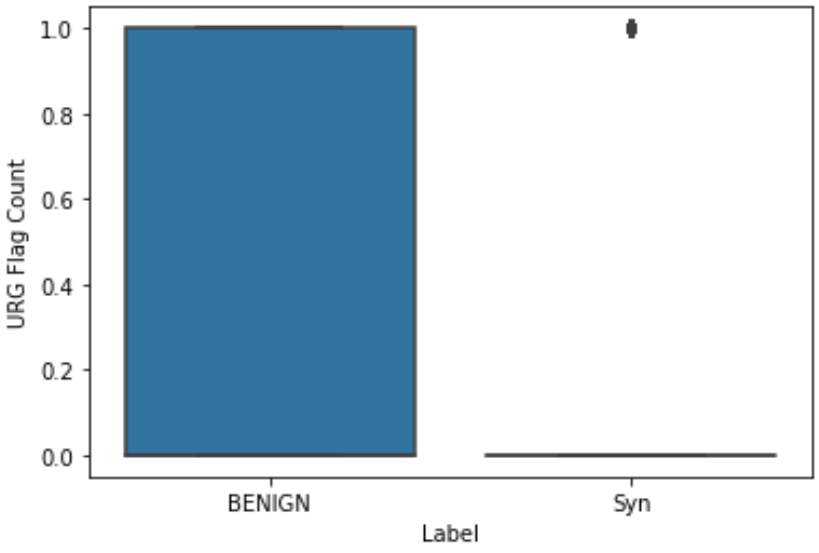



```
Feature: Init_Win_bytes_forward

BENIGN
  Mínimo: -1.00
  Limite inferior: -1.00
  1º Quartil (25%): -1.00
  Mediana: 256.00
  3º Quartil (75%): 8192.00
  Limite superior: 65535.00
  Máximo: 65535.00
  Média: 5690.44

Syn
  Mínimo: -1.00
  Limite inferior: -1.00
  1º Quartil (25%): 5840.00
  Mediana: 5840.00
  3º Quartil (75%): 5840.00
  Limite superior: 65535.00
  Máximo: 65535.00
  Média: 5834.44
```

Figura 15. Diagrama de caixa da *feature* **URG Flag Count** e análise dos dados

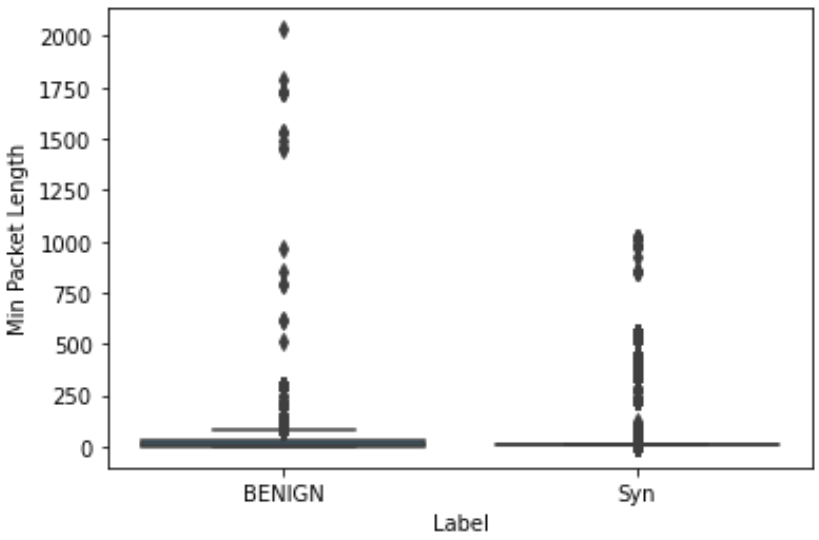



```
Feature: URG Flag Count

BENIGN
  Mínimo: 0.00
  Limite inferior: 0.00
  1º Quartil (25%): 0.00
  Mediana: 0.00
  3º Quartil (75%): 1.00
  Limite superior: 1.00
  Máximo: 1.00
  Média: 0.48

Syn
  Mínimo: 0.00
  Limite inferior: 0.00
  1º Quartil (25%): 0.00
  Mediana: 0.00
  3º Quartil (75%): 0.00
  Limite superior: 1.00
  Máximo: 1.00
  Média: 0.00
```

Figura 16. Diagrama de caixa da *feature* **Min Packet Length** e análise dos dados




```
Feature: Min Packet Length

 BENIGN
   Mínimo: 0.00
   Limite inferior: 0.00
   1º Quartil (25%): 0.00
   Mediana: 6.00
   3º Quartil (75%): 31.00
   Limite superior: 2033.00
   Máximo: 2033.00
   Média: 15.35

 Syn
   Mínimo: 0.00
   Limite inferior: 0.00
   1º Quartil (25%): 6.00
   Mediana: 6.00
   3º Quartil (75%): 6.00
   Limite superior: 1017.00
   Máximo: 1017.00
   Média: 6.56
```

5.2 Predição do modelo

Figura 17. Relação dos resultados previstos versus resultados esperados na predição do modelo

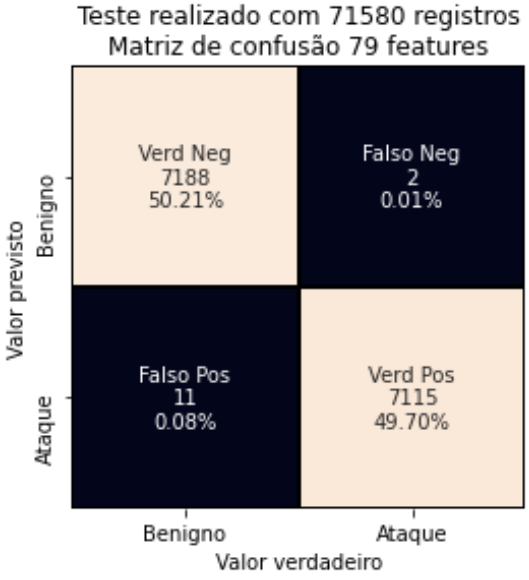


Tabela 2. Métricas de avaliação dos resultados da predição do modelo utilizando *Random forest*

	precision	recall	f1-score	support
0	0.9985	0.9997	0.9991	7190
1	0.9997	0.9985	0.9991	7126
accuracy			0.9991	14316

5.3 Validação do modelo

Figura 18. Separação das amostras utilizando o método *StratifiedK-Fold*.

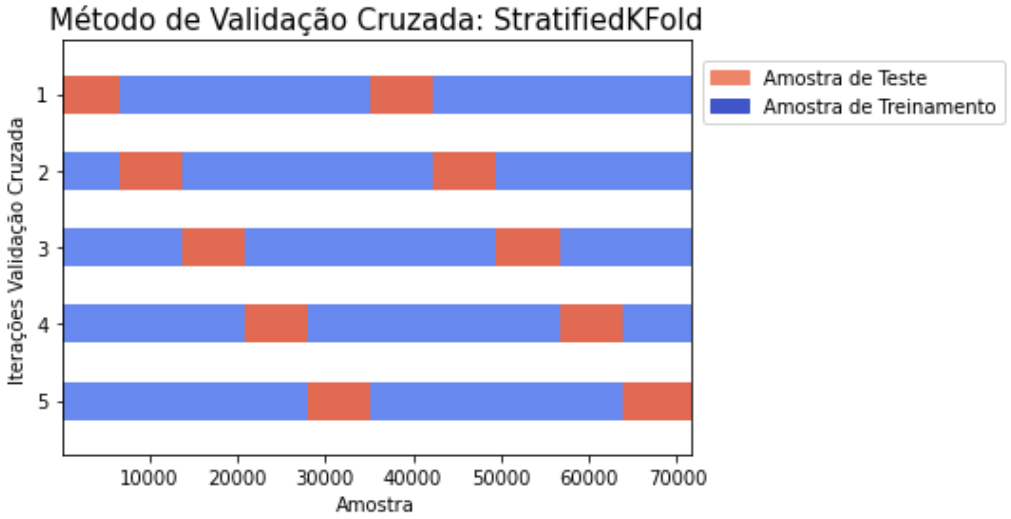


Figura 19. Matriz de confusão dos resultados da validação do modelo utilizando *Random forest* após separação do *dataset* com o método *StratifiedK-Fold*.

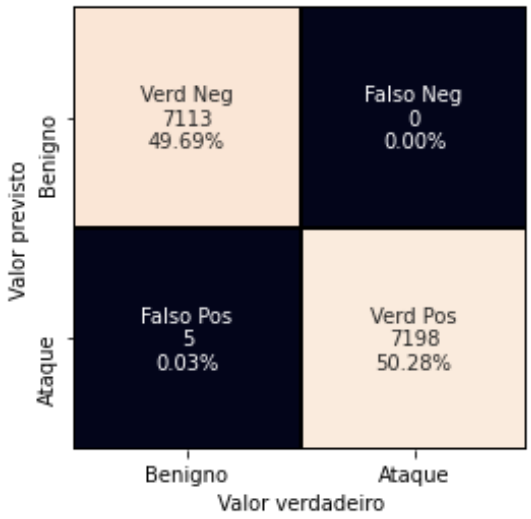


Tabela 3. Métricas de avaliação dos resultados da validação do modelo utilizando *Random forest* e *StratifiedK-Fold*

```
Acurácia: 0.99990
Precisão: 1.00000
Revocação: 0.99980
```

6 CONSIDERAÇÕES FINAIS E CONCLUSÕES

O trabalho proposto consistiu na elaboração de um modelo de aprendizado de máquina destinado a indicar se um pacote do tráfego de rede é benigno ou maligno, após ser treinado com um *dataset* fornecido previamente. As técnicas utilizadas para criação, treinamento e teste do modelo apresentaram alta taxa não apenas de acurácia, mas também de precisão e revocação, indicando que os registros dados como ataque, de fato eram ataques, e não falsos positivos.

Pelo fato do *dataset* ter passado por um processo inicial de limpeza, em que a primeira metade da tabela consistia apenas de registros benignos e a segunda metade apenas de registros malignos, a validação realizada através do algoritmo *Random forest* e do método de divisão de amostras *StratifiedK-Fold* permitiu que o teste fosse realizado em partes aleatórias do conjunto total, evitando que houvesse parcialidade do algoritmo durante o treinamento, e garantindo que a proporcionalidade original do *dataset* não interferisse na predição do modelo.

Adicionalmente, as *features* mais importantes trazem um panorama de quais características dos pacotes registrados influenciam na detecção dos ataques. Por exemplo, a *feature* ***URG Flag Count*** é a responsável por informar a estação receptora que alguns dados dentro de um determinado segmento são urgentes e devem ser priorizados. A análise dos dados de cada *feature* relevante, como mostrado da Figura 11 a Figura 16, indica que o resultado anotado no *dataset* está diretamente ligado ao valor registrado pelas *features* analisadas. Por exemplo, na Figura 12 é possível observar que o modelo treinado aprendeu com a *feature* ***Inbound***, visto que na maioria das vezes em que o valor em ***Inbound*** era 0, de fato, era um pacote benigno (associado a 0 na criação do modelo). O mesmo ocorre com a *feature* ***ACK Flag Count***. A associação acontece tanto para os pacotes maliciosos quanto para os pacotes benignos.

Por fim, os resultados obtidos apresentaram altos índices de acurácia, precisão e revocação, indicando alta chance de *overfitting*, ou seja, uma excelente performance para os dados testados, porém sem garantia da mesma performance para dados inéditos. Apesar de diversos hiper parâmetros terem sido testados durante a elaboração do trabalho, os índices continuaram altos.

6.1 Trabalhos futuros

Inicialmente, a solução proposta foi indicar o resultado de forma binária, entretanto para trabalhos futuros, e com apenas algumas modificações no código fonte, é possível alterar a saída do algoritmo para indicar, além do resultado, o tipo de ataque que foi detectado, sendo necessário apenas a inclusão de novos conjuntos de dados para treinamento do modelo criado e o uso de outros métodos de predição de dados, ou seja, trabalhando com modelos de regressão em vez de modelos de classificação.

Esta nova abordagem vai ao encontro da necessidade do mercado de identificar cada vez mais rápido os ataques que acontecem diariamente nas redes de computadores ao redor do mundo, nas mais diversas aplicações e plataformas utilizadas.

7 REFERÊNCIAS BIBLIOGRÁFICAS

[1] PRNewswire, “Os mais famosos ataques DDoS da história – e o que podem nos ensinar sobre segurança na web,” 28 outubro 2019. [Online]. Available: https://exame.com/negocios/releases/os-mais-famosos-ataques-ddos-da-historia-e-o-que-podem-nos-ensinar-sobre-seguranca-na-web.

[2] ITIC, “Hourly Downtime Costs Rise: 86% of Firms Say One Hour of Downtime Costs $300,000+; 34% of Companies Say One Hour of Downtime Tops $1Million,” 16 Maio 2019. [Online]. Available: https://itic-corp.com/blog/2019/05/hourly-downtime-costs-rise-86-of-firms-say-one-hour-of-downtime-costs-300000-34-of-companies-say-one-hour-of-downtime-tops-1million/.

[3] Ponemon Institute, “The State of DDoS Attacks against Communication Service Providers,” Abril 2019. [Online]. Available: https://www.a10networks.com/wp-content/uploads/A10-EB-14117-EN.pdf.

[4] A10 Networks, “DDoS Threat Intelligence Map,” A10 Networks, [Online]. Available: https://threats.a10networks.com/. [Acesso em Maio 2020].

[5] Wikipédia, “SYN Flood,” Wikipédia, 03 Maio 2020. [Online]. Available: https://pt.wikipedia.org/wiki/SYN_Flood.

[6] O. Yoachimik e A. Singh, “Network-Layer DDoS Attack Trends for Q1 2020,” Cloudflare, 15 Maio 2020. [Online]. Available: https://blog.cloudflare.com/network-layer-ddos-attack-trends-for-q1-2020.

[7] I. Sharafaldin, A. H. Lashkari, S. Hakak e A. A. Ghorbani, “Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy,” IEEE, Outubro 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8888419.

[8] Wikipédia, “Modelo OSI,” Wikipédia, 10 Março 2020. [Online]. Available: https://pt.wikipedia.org/wiki/Modelo_OSI.

[9] “Data Link Layer of the OSI Model: Protocol, Functions & Design,” Study.com, 2020. [Online]. Available: https://study.com/academy/lesson/data-link-layer-of-the-osi-model-protocol-functions-design.html.

[10] “The OSI Model's Seven Layers Defined and Functions Explained,” , . [Online]. Available: https://support.microsoft.com/kb/103884. [Acesso em 9 11 2020].

[11] Wikipédia, “Ataque de negação de serviço,” Wikipédia, 02 Setembro 2019. [Online]. Available: https://pt.wikipedia.org/wiki/Ataque_de_nega%C3%A7%C3%A3o_de_servi%C3%A7o.

[12] Wikipédia, “Aprendizado de máquina,” Wikipédia, 27 Março 2020. [Online]. Available: https://pt.wikipedia.org/wiki/Aprendizado_de_m%C3%A1quina.

[13] V. Lall, “Google Colab — The Beginner’s Guide,” Lean In Women in Technology, India, 01 Abril 2018. [Online]. Available: https://medium.com/machina-sapiens/google-colab-guia-do-iniciante-334d70aad531.

[14] Wikipédia, “scikit-learn,” Wikipédia, 28 Janeiro 2020. [Online]. Available: https://pt.wikipedia.org/wiki/Scikit-learn.

[15] A. H. Lashkari, “GitHub - ahlashkari/CICFlowMeter: CICFlowmeter-V4.0 (formerly known as ISCXFlowMeter),” [Online]. Available: https://github.com/ahlashkari/CICFlowMeter.

[16] Wikipédia, “Random forest,” Wikipédia, 02 Maio 2020. [Online]. Available: https://en.wikipedia.org/wiki/Random_forest.

[17] “Classification: Accuracy,” [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy.

[18] Wikipédia, “Support vector machine,” Wikipédia, 18 Maio 2020. [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine.

[19] Aliger, “Entenda o aprendizado não supervisionado no Machine Learning,” Aliger, 30 Julho 2019. [Online]. Available: https://www.aliger.com.br/blog/machine-learning-entenda-o-que-e-aprendizado-nao-supervisionado.

[20] Wikipédia, “Supervised learning,” Wikipédia, 12 Maio 2020. [Online]. Available: https://en.wikipedia.org/wiki/Supervised_learning.

[21] Wikipédia, “Reinforcement learning,” Wikipédia, 26 Maio 2020. [Online]. Available: https://en.wikipedia.org/wiki/Reinforcement_learning.

[22] Wikipédia, “Teorema de Bayes,” Wikipédia, 22 Agosto 2019. [Online]. Available: https://pt.wikipedia.org/wiki/Teorema_de_Bayes.

[23] Wikipédia, “Decision tree learning,” Wikipédia, 18 Maio 2020. [Online]. Available: https://en.wikipedia.org/wiki/Decision_tree_learning.

[24] Wikipédia, “Naive Bayes classifier,” Wikipédia, 20 Maio 2020. [Online]. Available: https://en.wikipedia.org/wiki/Naive_Bayes_classifier.

[25] Wikipédia, “Regressão logística,” Wikipédia, 19 Novembro 2019. [Online]. Available: https://pt.wikipedia.org/wiki/Regress%C3%A3o_log%C3%ADstica.

[26] scikit-learn, “1.10. Decision Trees,” scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/tree.html.