UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES
CAMPUS PATOS DE MINAS

JOÃO GUILHERME DO NASCIMENTO TELES

DETECÇÃO DE AMEAÇAS DDOS COM APRENDIZAGEM DE MÁQUINA

Patos de Minas - MG

2022

JOÃO GUILHERME DO NASCIMENTO TELES

DETECÇÃO DE AMEAÇAS DDOS COM APRENDIZAGEM DE MÁQUINA

Projeto de pesquisa apresentado à banca examinadora como requisito parcial de avaliação da disciplina de TCC2 da graduação em Engenharia Eletrônica e de Telecomunicações, da Faculdade de Engenharia Elétrica, da Universidade Federal de Uberlândia, Campus Patos de Minas.

Orientador: Prof. Eliana Pantaleão

Co-Orientador: Pedro Luiz Lima Bertarini

Patos de Minas - MG

2022

JOÃO GUILHERME DO NASCIMENTO TELES

DETECÇÃO DE AMEAÇAS DDOS COM APRENDIZAGEM DE MÁQUINA

Projeto de pesquisa apresentado à banca examinadora como requisito parcial de avaliação da disciplina de TCC2 da graduação em Engenharia Eletrônica e de Telecomunicações, da Faculdade de Engenharia Elétrica, da Universidade Federal de Uberlândia, Campus Patos de Minas.

Patos de Minas, 31 de Março de 2022

COMISSÃO EXAMINADORA

Prof.ª Dr.ª Eliana Pantaleão

(Orientadora)

Prof. Dr. Pedro Luiz Lima Bertarini

(Co-Orientador)

Prof. Dr. Laurence Rodrigues do Amaral

(Examinador)

Prof.ª Dr.ª Karine Barbosa Carbonaro

(Examinadora)

RESUMO

O desenvolvimento exponencial das telecomunicações promove o surgimento de diversas aplicações que são beneficiadas pelas altas taxas de transmissão. Serviços de *streaming* e empresas que utilizam transações monetárias digitais são alguns exemplos dentre a infinidade de aplicações possíveis a partir desses avanços tecnológicos. Contudo, esse crescente desenvolvimento das telecomunicações trouxe consigo um impacto negativo: criminosos que utilizam das altas taxas de transmissão para a realização de ataques de força bruta. Entre esses ataques, uma classe que tem se popularizado é a de ataques DDoS, onde os criminosos enviam múltiplas requisições a vítima com o objetivo de sobrecarregar seus servidores, impactando diretamente nos serviços prestados por ela. Esse trabalho busca propor duas abordagens diferentes para a realização da tarefa de detecção de ataques dessa natureza: uma metodologia baseada na análise de séries temporais utilizando a entropia de certos atributos de fluxo de rede e outra abordagem utilizando a análise de componentes principais em conjunto com redes neurais artificiais e máquina de vetores de suporte para a realização da classificação entre situações normais e situações de ataque. Por fim será realizada uma comparação entre as duas abordagens propostas utilizando as métricas adequadas.

Palavras-Chave: DDoS, Análise de Componentes Principais, Aprendizagem de Máquina, Redes Neurais Artificiais, Entropia, Máquina de Vetores de Suporte, ARIMA, Análise de Séries Temporais

ABSTRACT

The exponential development of telecommunications promotes the appearance of several applications that benefit from the high transmission rates. Streaming services and companies that use digital monetary transactions are some examples among the infinity of possible applications based on these technological advances. However, this growing development in telecommunications has had a negative impact: criminals who use high transmission rates to carry out brute force attacks. Among these attacks, a class that has become popular is DDoS attacks, where criminals send multiple requests to the victim to overload their servers, directly impacting the services provided by them. This work proposes two different approaches to perform the task of detecting attacks of this nature: a methodology based on the analysis of time series in the entropy of certain network flow attributes and another approach using principal component analysis in conjunction with artificial neural networks and a support vector machine to perform the classification between normal situations and attack situations. Finally, a comparison will be made between the two proposed approaches using the appropriate metrics.

Keywords: DDoS, Principal Component Analysis, Machine Learning, Artificial Neural Networks, Entropy, Support Vector Machine, ARIMA, Time Series Analysis

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AR	*Autoregressive*	Autorregressivo
ARIMA	*Autoregressive Integrated Moving Average*	Média Móvel Integrada Autorregressiva
ARMA	*Autoregressive Moving Average*	Média Móvel Autorregressiva
ANN	*Artificial Neural Network*	Rede Neural Artificial
CSV	*Comma-Separated-Values*	Valores Separados por Vírgulas
DDoS	*Distributed Denial of Service*	Negação de Serviço Distribuído
EDoS	*Economic Denial of Service*	Negação de Serviço Econômico
FTP	*File Transfer Protocol*	Protocolo de Transferência de Arquivo
HTTP	*Hyper Text Transfer Protocol*	Protocolo de Transferência por Hipertexto
HTTPS	*Hyper Text Transfer Protocol Secure*	Protocolo de Transferência por Hipertexto Seguro
IDS	*Intrusion Detection System*	Sistema de Detecção de Intrusão
IP	*Internet Protocol*	Protocolo de Internet
KL	*Karhunen-Loève*	-
MA	*Moving Average*	Média Móvel
ML	*Machine Learning*	Aprendizado de Máquina
PCA	*Principal Component Analysis*	Análise de Componentes Principais
PCAP	*Packet Capture Data Format*	Formato de dados de captura de pacotes
ROC	*Receiver Operating Characteristic*	Características Operacionais do Receptor
SSH	*Secure Shell*	Capsula Segura
SVM	*Suport Vector Machine*	Máquina de Vetores de Suporte

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

Os avanços tecnológicos das últimas décadas trouxeram inúmeros benefícios para a sociedade. No que diz respeito às telecomunicações, as altas taxas de transmissão de dados aliadas à possibilidade de ter diversos dispositivos conectados, proporcionaram um grande progresso na área da tecnologia. No entanto, esses avanços trouxeram, também, novos desafios no que se refere à segurança de dados.

Entre os novos desafios, um que se destaca entre os demais é o DDoS (*Distributed Denial of Service*), um tipo de ataque em que os criminosos utilizam um grande volume de dados para causar danos às suas vítimas [1]. Grandes empresas, como: PayPal, Netflix, Spotify e Twitter já sofreram com ataques dessa natureza, e a crescente modernização dos ataques torna mais difícil a detecção dos mesmos [2] [3].

Os ataques DDoS têm como principal objetivo inviabilizar o funcionamento do recurso Web da vítima, gerando o que é chamado de "negação de serviço" (em inglês, *Denial of Sevice*), que deu origem ao nome do tipo de ataque. Esse objetivo é alcançado por meio de múltiplas solicitações para recursos Web enviadas pelo criminoso, que sobrecarregam os servidores da vítima. Por fim, essa sobrecarga pode gerar o mau funcionamento, ou até mesmo a indisponibilidade, do serviço oferecido pela vítima [4].

Esses ataques parecem inofensivos, visto que muitas das vezes não há roubo de informação por parte dos criminosos. Contudo, foi relatado pela empresa Karpesky Lab, que um ataque DDoS resulta, em média, num prejuízo de 444.000,00USD por ano em perdas de negócios e gastos de reparo [5]. Com isso, diversas formas de solução foram propostas, podendo ser divididas em três classes distintas: prevenção, detecção e mitigação [6].

Uma vez que a solução preventiva não consegue impedir o ataque, é necessário utilizar recursos para a mitigação do problema. Contudo, antes de implementar um plano de ação para interromper o ataque, é necessário que ele seja detectado. Para isso, diversas abordagens são exploradas, desde sistemas baseados na entropia do fluxo de dados [7] [8] até sistemas que utilizam modelos estatísticos de teste de hipótese [9].

Outra importante abordagem de que se tem conhecimento é a utilização de sistemas inteligentes cujo objetivo é aprender, por meio de métodos de classificação, utilizando os dados

de fluxo de rede. Essa abordagem, conhecida como aprendizagem de máquina (ML do inglês *machine learning*), tem como característica principal a sua adaptabilidade a problemas de diversas naturezas que possuam dados para o "treinamento" do sistema.

No estudo de sistemas de ML a aprendizagem pode se dar, no geral, de duas formas: utilizando dados não rotulados para treinar o sistema (aprendizagem não supervisionada) ou utilizando dados rotulados para o treinamento (aprendizagem supervisionada). No entanto, há ainda, sistemas que utilizam a abordagem semi-supervisionada, em que o sistema utiliza as duas abordagens anteriores de forma conjunta [1].

De modo geral, o estudo das características de ataques cibernéticos é de extrema importância para o planejamento de sistemas que possam oferecer segurança contra ameaças desse tipo.

1.1. Tema do Projeto

Realização de um estudo comparativo entre um sistema de ML que utilize uma abordagem supervisionada para a detecção de ameaças DDoS e outro que utilize um modelo baseado na previsão de entropia por meio do método ARIMA [10] [11]. A abordagem supervisionada terá como descritor os dados de tráfego em uma determinada janela de tempo. E a abordagem baseada na entropia dos dados terá como base, apenas, quatro atributos (pacotes e bytes enviados e recebidos).

1.2. Problematização

A modernização dos ataques DDoS oferece inúmeros impactos paras as grandes empresas. O estudo de técnicas de detecção para essas ameaças, visando a melhor forma de analisar os dados obtidos ao longo do ataque, possui grande relevância no que diz respeito a segurança da informação.

O grande volume de dados e de características a serem analisados estão entre os principais problemas no desenvolvimento de sistemas de detecção de ameaças. Com isso, métricas como tempo de processamento, acurácia e precisão serão os principais indicadores a serem comparados entre os sistemas a serem estudados.

1.3. Hipóteses

Hipótese 1: A implementação da análise de principais componentes na tarefa de detecção de ataques DDoS proporciona melhores resultados em relação a sistemas sem essa análise.

Hipótese 2: A análise de entropia fornece informação suficiente para a detecção de ataques cuja natureza se baseia no disparo consecutivo de requisições.

1.4. Objetivos

1.5.1. Objetivos Gerais

Esse trabalho tem como objetivo realizar um estudo comparativo entre dois algoritmos para detecção de ameaças DDoS baseados em diferentes abordagens. Além disso, será analisado, também, o impacto da técnica PCA na implementação de algoritmos de aprendizado supervisionado.

1.5.2. Objetivos Específicos

- Implementação de um sistema para o tratamento dos dados de rede no formato CSV;
- Desenvolvimento dos algoritmos para a detecção de ameaças DDoS;
- Computar as principais métricas de avaliação de cada algoritmo;
- Comparar as métricas para diferentes cenários.

1.5. Justificativas

A idealização de uma sociedade mais conectada por meio da tecnologia trouxe consigo diversos problemas envolvendo segurança da informação. Não só os cidadãos comuns, mas também as grandes empresas sofrem com as diversas possibilidades de ataques que podem resultar no roubo de informações sigilosas ou na fragilização de serviços prestados, no caso das empresas.

Contudo, o estudo de algoritmos de reconhecimento de padrões através de ML tem se mostrado muito promissor em dados de diversas naturezas. Assim, a avaliação da eficiência desses métodos na área de detecção de ameaças cibernéticas é de fundamental importância.

1.6. Considerações Finais

O objetivo desse trabalho é expor de forma clara e concisa todas as etapas de um algoritmo capaz de detectar ameaças DDoS a partir de dados de fluxo rede no formato CSV. O trabalho tem como finalidade realizar uma comparação entre duas abordagens diferentes para a detecção de ameaças DDoS utilizando as principais métricas de avaliação. Além disso, será avaliado, também, o impacto de técnicas de redução de dimensionalidade nos algoritmos de aprendizado supervisionado.

Esse trabalho está dividido da seguinte forma: o Capítulo 2 contém o referencial teórico onde serão abordados os conceitos de modelagem de bases de dados que simulem situações reais, utilização da entropia de um conjunto de dados para a análise de fluxos de rede, aplicação de processos estocásticos para a previsão de séries temporais e utilização de algoritmos preditivos supervisionados para o treinamento de algoritmos. O Capítulo 3 tem como objetivo explanar a metodologia proposta no que diz respeito a arquitetura de código e de processos que, aliada aos conceitos do Capítulo 2, dará sentido ao trabalho. O Capítulo 4 se refere aos recursos utilizados para a execução de todo o projeto, desde *hardware* até os *softwares* que serão utilizados para o desenvolvimento. O capítulo 5 apresenta os resultados e dos testes realizados e, por fim o Capítulo 6 conclui o trabalho e mostra sugestões para pesquisas futuras.

CAPÍTULO 2

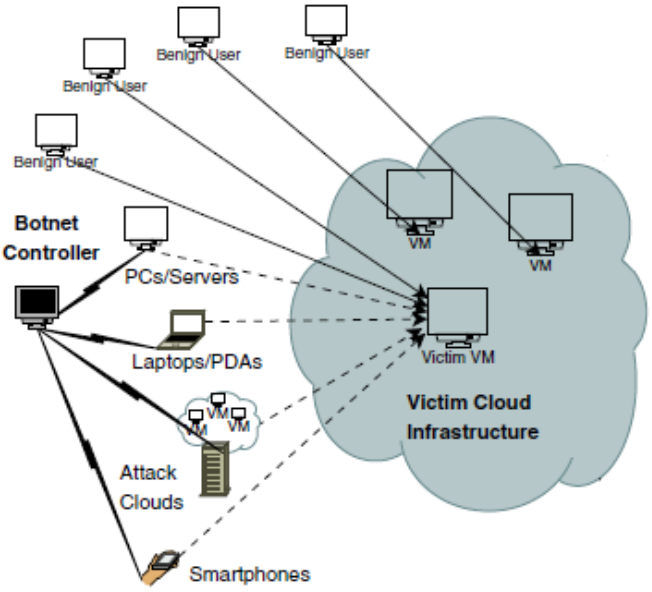
REFERENCIAL TEÓRICO

Nesse capítulo são apresentados os principais conceitos teóricos para o desenvolvimento dos dois algoritmos propostos, bem como os conceitos necessários para uma comparação justa entre ambos. Primeiramente é importante entender a estrutura de um ataque DDoS e estudar a estrutura das bases de dados que serão utilizadas para testar os algoritmos. Em seguida, alguns tópicos importantes para o desenvolvimento do algoritmo serão apresentados: entropia em um conjunto de dados, utilização do modelo ARIMA para processos de limiarização, técnicas de redução de dimensionalidade e, por fim, classificadores para aprendizagem supervisionada.

2.1. Estrutura Básica de um Ataque DDoS

Por mais que esse não seja o escopo desse trabalho, é importante ter uma noção básica da taxonomia de um ataque DDoS. Uma infraestrutura típica para o lançamento de ataques DDoS é o da Figura 1 onde o criminoso utiliza máquinas comprometidas como espelho para o ataque, aumentando a eficiência do ataque por meio da repetição de solicitações à vítima do ataque [12].

Figura 1 – Estrutura de um ataque DDoS a uma infraestrutura em nuvem



Fonte: Adaptado de [12]

A Figura 1 mostra a topologia básica de um ataque DDoS onde o ataque é distribuído para dispositivos eletrônicos diversos (*laptops*, *desktops*, *smartphones*, *servidores*) cujo objetivo é prejudicar os serviços prestados pelo servidor. Esses tipos de ataque possuem diversas motivações: competição entre empresas, política, extorsões e até guerras cibernéticas.

2.2. Bases de Dados

O principal objeto de estudo dos algoritmos de aprendizagem são os dados. De nada adianta desenvolver o algoritmo perfeito para o reconhecimento de padrões se a base de dados fornecida para o seu funcionamento não fornece quantidade de informação suficiente para a aplicação.

Em virtude disso, uma das principais etapas para o desenvolvimento de qualquer algoritmo de aprendizagem é o estudo dos dados que serão utilizados para o treinamento dos modelos de aprendizagem. Dessa forma, duas abordagens são possíveis para a escolha desse recurso tão importante, sendo elas: a geração de dados ser, também, escopo do desenvolvimento do algoritmo ou trabalhar com bases de dados já existentes, focando o desenvolvimento nas etapas posteriores a captação de informações. Como este trabalho tem como objetivo o desenvolvimento dos modelos de predição e não a captação dos dados de rede que são necessários para o desenvolvimento do modelo, optou-se por utilizar as bases de dados, já utilizadas no meio acadêmico, UNB CIC-IDS 2017 [13] e UNB CIC-DDoS 2019 [14].

É importante frisar que a coleta de dados para análises e desenvolvimento de sistemas de detecção de intrusão (IDS do inglês *Intrusion Detection System*) precisa acontecer de forma constante. Essa necessidade é fruto da alteração de dados de tráfego ao longo do tempo e dos novos tipos de ataques que são desenvolvidos. Além disso, dados antigos, além de não traduzirem a realidade atual, podem sofrer com a falta de atributos importantes.

2.2.1. UNB CIC-IDS 2017

Essa base de dados contém dados benignos, de tráfego comum, e dados dos mais diferentes tipos de ataques, reunidos em arquivos no formato PCAP. Além disso, essa base de dados inclui um arquivo de análise no formato CSV, com fluxos de dados classificados em benigno ou maligno, IP de envio e destino e até horário do fluxo, obtido a partir do software *CICFlowMeter* [15] [16].

Essa base de dados foi gerada a partir do sistema *B-profile* [17] proposto por Sharalfadin *et al.* com objetivo de caracterizar o comportamento humano gerando dados de tráfego benignos. A base de dados é composta pelo comportamento de 25 usuários comuns baseados nos protocolos HTTP, HTTPS, FTP, SSH e protocolos de e-mail.

O sistema foi implementado do dia 3 de julho de 2017 até o dia 7 de julho de 2017. Nessa faixa de tempo diversos ataques foram implementados: *Brute Force FTP*, *Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS*. Contudo, como este trabalho busca detectar apenas ataques DDoS, apenas o arquivo de fluxo de rede desse tipo de ataque foi utilizado. O Quadro 1 mostra as caraterísticas do arquivo utilizado na implementação do algoritmo.

Quadro 1 – Características dos ataques realizados no dia 7 de julho de 2017 para a construção da base de dados

Tipo de Ataque	**Período do Ataque**
Botnet ARES	10:02 - 11:02
Port Scan	-
DDoS LOIT	15:56 - 16:16

Fonte: [13]

Os fluxos de dados obtidos por meio do sistema de coleta fornecem 78 atributos para cada linha de dados.

2.2.2. UNB CIC-DDoS 2019

Essa base de dados é uma base exclusiva de ataques DDoS e que, inclusive, explora os mais diferentes tipos de abordagens dentro dessa classe de ataques. A base utilizada nos estudos de Sharalfadin et al. [14] foi montada em conjunto com a proposta de uma nova visão para a taxonomia dos ataques DDoS, onde os ataques são divididos em duas classes: DDoS baseados em exploração e DDoS baseados em reflexão, onde ambos exploram um terceiro componente com objetivo de utilizá-lo como "refletor" para o lançamento do ataque.

Essa base de dados foi gerada, também, a partir do sistema *B-profile* [17] para a simulação do comportamento humano, obtendo assim, dados de tráfego naturais benignos. A base contém o comportamento abstrato de 25 usuários baseado nos protocolos HTTP, HTTPS, FTP, SSH e protocolos de e-mail. Foram simulados os fluxos de dados por dois dias e captados na forma PCAP e as informações foram extraídas pela plataforma *CICFlowMeter-V3* [15] [16] e exportadas para o formato CSV. Os ataques realizados constam no Quadro 2.

Quadro 2 – Características dos ataques realizados para a construção da base de dados

Dia	Tipo de Ataque DDoS	Horário dos ataques
1	PortMap NetBIOS LDAP MSSQL UDP UDP-Lag SYN	9:43 - 9:51 10:00 - 10:09 10:21 - 10:30 10:33 - 10:42 10:53 - 11:03 11:14 - 11:24 11:28 - 17:35
2	NTP DNS LDAP MSSQL NetBIOS SNMP SSDP UDP UDP-Lag WebDDoS SYN TFTP	10:35 - 10:45 10:52 - 11:05 11:22 - 11:32 11:36 - 11:45 11:50 - 12:00 12:12 - 12:23 12:27 - 12:37 12:45 - 13:09 13:11 - 13:15 13:18 - 13:29 13:29 - 13:34 13:35 - 17:15

Fonte: [14]

Os fluxos de dados obtidos através do sistema de coleta fornecem 87 atributos para cada linha de dados.

É importante destacar que os tipos de ataques são discriminados de acordo com o protocolo ou o serviço explorado para o ataque. Por exemplo, o ataque do tipo SNMP tem como base o protocolo da camada de aplicação SNMP (do inglês *Simple Network Management Protocol*), utilizado para o gerenciamento de redes. Outro exemplo é o ataque DNS (do inglês *Domain Name System*) que busca explorar o serviço de DNS de servidores para o lançamento do ataque.

2.3. Entropia de um conjunto de dados

Uma medida amplamente utilizada na detecção e análise de fluxos de rede é a medida de entropia. Monge *et al.* [8] utilizaram o conceito de entropia para criar um algoritmo capaz de detectar ataques EDoS. Seguindo o mesmo conceito, Idhammad *et al.* [1] aplicaram o conceito junto a técnicas de aprendizagem de máquina para o aumento da performance de detecção de ataques cibernéticos.

A entropia é definida, para uma fonte de informação contínua e sem memória, como uma medida do conteúdo médio de informação por símbolo-fonte [18] (1). Neste trabalho, essa medida terá como objetivo principal indicar a quantidade de informação contida em um evento a partir de um subconjunto dos atributos dos dados de rede. O evento em questão se trata de uma janela temporal *w* dentro do conjunto de dados de treinamento sob a qual o algoritmo para cálculo de entropia será aplicado.

$$H(X) = -\sum_{i=1}^{n} p(x_i) \cdot \log\big(p(x_i)\big) \quad (1)$$

Essa etapa, por ter como base a quantidade de informação contida dentro de um conjunto de dados, é aplicada apenas aos atributos mais relevantes para o problema e que, de fato, representam as informações mais sensíveis dentro de um ataque DDoS.

Os atributos utilizados foram as quantidades de bytes e pacotes, enviados e recebidos, por fluxo de dados, totalizando quatro atributos para a análise de entropia. É importante ressaltar que a motivação por trás da escolha desses atributos se deve às características intrínsecas de um ataque DDoS que tem como objetivo sobrecarregar a vítima com solicitações de serviço. Por esse motivo, os atributos escolhidos para a análise precisam ser os que mais sofrem alterações no momento de um ataque, impactando diretamente a entropia do conjunto de dados.

A implementação da análise de entropia, em conjunto com um algoritmo de seleção de dados por limiar inferior e superior (T_d e T_u), tem como objetivo detectar as situações de ataque.

2.3.1. Algoritmo para estabelecimento de limiares

Algoritmos preditivos baseados em séries temporais são amplamente utilizados para diversas aplicações [19] [20] [21] [22]. Sua utilização é possível tanto para a previsão de variáveis futuras ligadas a séries temporais como também para o processo de estabelecimento de limiares máximos e mínimos para essas variáveis futuras.

O estabelecimento de limiares, por meio da análise de séries temporais, é comumente implementado a partir de previsões anteriores aos dados que serão submetidos ao processo de seleção por limiar conforme as equações (2) e (3).

$$T_u = \hat{y} + k \times \sqrt[2]{var(y - \hat{y})} \quad (2)$$

$$T_d = \hat{y} - k \times \sqrt[2]{var(y - \hat{y})} \quad (3)$$

Os dois limiares (T_u e T_d), estabelecidos pelas equações (2) e (3), se referem aos limiares de seleção superior e inferior, respectivamente, para a seleção dos dados para as etapas posteriores do problema. A constante *k*, dependente do tipo de distribuição de probabilidade utilizado para o cálculo do erro de previsão, representa o índice de confiança da previsão.

A variável $\hat{y}$ representa a previsão de um valor futuro dessa série temporal e y refere-se ao valor real dessa mesma previsão. O próximo tópico desse trabalho será destinado para o desenvolvimento da base necessária para o entendimento do algoritmo empregado para prever os dados futuros da série temporal utilizada neste trabalho.

2.3.2. Previsão de séries temporais a partir do algoritmo ARIMA

Uma solução possível para a previsão de variáveis futuras dentro de uma série temporal é utilizando o modelo autorregressivo integrado de média móvel (ARIMA do inglês *autoregressive integrated moving average*) proposto por Box e Jenkins [10]. Esse modelo de solução para a previsão de dados futuros foi proposto para dados cuja natureza apresenta características não-estacionárias, não podendo ser expressos por meio dos processos AR (do

inglês *autoregressive*), MA (do inglês *moving average*) e ARMA (do inglês *autoregressive moving average*).

Para uma boa compreensão do processo ARIMA, é necessário entender os outros dois processos que quando integrados, dão forma ao mesmo. Seguindo a ordem proposta pela sigla do processo completo, o primeiro a ser explicado será o processo AR, comumente aplicado a séries temporais com característica estacionária, ou seja, uma série que apresenta equilíbrio estatístico podendo ser descrita a partir de sua média, variância e função de densidade espectral.

A princípio, é necessário definir o operador *backshift* (*B*) como o operador cuja aplicação em uma variável z_t, pertencente a uma série temporal, resulta em:

$$Bz_t = z_{t-1} \qquad (4)$$

Se aplicado novamente:

$$B(Bz_t) = B^2 z_t \quad (5)$$

$$B^2 z_t = z_{t-2} \qquad (6)$$

Dessa forma:

$$B^k z_t = z_{t-k} \qquad (7)$$

A partir da equação (8) e representando uma série temporal a partir de uma função de p valores passados, z_{t-1}, z_{t-2}, z_{t-3}, ... , z_{t-p}, onde p determina quantos passos foram dados para realizar a previsão do valor atual x_t, um processo de AR de ordem p, denotado $AR(p)$ poderá ser definido como [11]:

$$z_t = \Phi_1 z_{t-1} + \Phi_2 z_{t-2} + \cdots + \Phi_p z_{t-p} + a_t \quad (8)$$

Sendo a_t modelado, neste caso, como um ruído do tipo AWGN com média zero e variância σ^2. Além disso, Φ_p é dado como uma constante diferente de zero.

Dessa forma, utilizando as propriedades do operador *backshift* é possível reescrever a equação 8 da seguinte forma:

$$\left(1 - \Phi_1 B - \Phi_2 B^2 - \cdots - \Phi_p B^p\right) z_t = a_t \quad (9)$$

Tomando como situação basal o método Autorregressivo de primeira ordem, denotado por $AR(1)$, pode-se dizer que:

$$(1 - \Phi_1 B) z_t = a_t \ (10)$$

Reescrevendo a equação (10)

$$z_t = (1 - \Phi_1 B)^{-1} a_t = \sum_{j=0}^{\infty} \Phi_1^j a_{t-j} \quad (11)$$

Sendo a equação (11) convergente, é possível denotar as seguintes equações

$$\Psi(B) = (1 - \Phi_1 B)^{-1} = \sum_{j=0}^{\infty} \Phi_1^j B^j \quad (12)$$

$$z_t = \Phi^{-1}(B) a_t \equiv \Psi(B) a_t = \sum_{j=0}^{\infty} \Psi_j a_{t-j} \quad (13)$$

Sendo a equação (13) o processo geral do processo $AR(p)$. Supondo a convergência do somatório em (13), pode-se afirmar que

$$\Phi(B) = \left(1 - G_1 B)(1 - G_2 B) \dots (1 - G_p B\right) \quad (14)$$

Sendo $G_1, G_2, \dots, G_p$ as raízes de $\Phi(B) = 0$.

Logo, o processo autorregressivo pode ser interpretado como uma saída z_t de um filtro linear com função de transferência $\Phi^{-1}(B) = \Psi(B)$ sendo sua entrada o ruído branco a_t.

Seguindo a ordem proposta pela sigla, o segundo processo necessário para o entendimento do algoritmo ARIMA é o processo de média móvel (MA do inglês *moving average*). Supondo, como anteriormente, uma série temporal representada a partir de uma função de *p* valores passados, z_{t-1}, z_{t-2}, z_{t-3}, ... , z_{t-q}, onde q determina quantos passos foram dados para realizar a previsão do valor atual z_t, um processo de MA de ordem q, denotado $MA(q)$ poderá ser definido como [11]:

$$z_t = a_t \ - \ \theta_1 a_{t-1} - \theta_2 a_{t-2} - \dots - \theta_q a_{t-q} \quad (15)$$

Utilizando o operador *backshift* pode-se dizer que

$$z_t = (1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q) a_t \ (16)$$

Ou ainda

$$z_t = \theta(B) a_t \ (17)$$

Unindo ambas as abordagens, é possível atingir o processo autorregressivo de média móvel (ARMA do inglês *autoregressive-moving average*), tendo como modelo resultante a equação (18)

$$z_t = \Phi_1 z_{t-1} + \Phi_2 z_{t-2} + \cdots + \Phi_p z_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \cdots - \theta_q a_{t-q} \quad (18)$$

Dessa forma, é denominado modelo ARMA de ordem *p*, *q* o modelo resumido pela seguinte equação (19)

$$\Phi(B) z_t = \theta(B) a_t \quad (19)$$

Tomando $\nabla = 1 - B$, o método ARIMA pode ser definido a partir do operador autorregressivo não-estacionário $\varphi(B)$, de forma que

$$\varphi(B) z_t = \theta(B) a_t \quad (20)$$

Sendo

$$\varphi(B) = \Phi(B)(1 - B)^d z_t \quad (21)$$

Tomando $\nabla^d = (1 - B)^d$ e $c_t = \nabla^d z_t$ é possível representar a equação (21) de forma que

$$\Phi(B) \nabla^d z_t = \theta(B) a_t \quad (22)$$

$$\mathrm{z}_t = S^d c_t \quad (23)$$

Sendo S^d o operador, inverso a ∇^d, de somatório infinito.

Dessa forma, z_t pode ser representado como uma soma do processo estacionário por *d* vezes, dando origem ao nome *integrated*, dando sentido à integração.

A partir desse processo, é possível realizar previsões futuras a partir das variáveis *p, q* e *d*.

2.4. Análise de Componentes Principais

Um tópico de extrema relevância dentro do reconhecimento de padrões refere-se à análise de atributos dos dados. A partir desse estudo é possível tomar algumas situações como ideais para a correta análise de um padrão. Entre essas situações, uma que se destaca é a utilização de dados que não possuam informações redundantes entre si. Ou seja, tomando *x* como os dados de entrada de um sistema de transformação de atributos, tendo *y* como a saída desse mesmo sistema, pode-se dizer que dados otimamente não relacionados são dados que respeitam a equação (24).

$$E[y(i)y(j)] = 0, i \neq j \quad (24)$$

Sendo *E* o operador de covariância aplicado aos dados.

Dessa forma, sendo A uma matriz ortogonal, o processo de transformação pode ser dado como

$$y = A^T x \quad (25)$$

Ou seja, os dados de entrada *x* passam por uma transformação tal que a correlação entre os valores de *y* seja mínima.

A partir da definição da matriz de correlação (R_x), pode-se dizer que:

$$R_y \equiv E[yy^T] = E[A^T x x^T A] = A^T R_x A \quad (26)$$

A simetria de R_x garante a ortonormalidade mútua entre os autovetores associados a mesma, sendo os autovetores as colunas da matriz *A*.

Dessa forma, a relação estabelecida pela equação (26) garante que R_y seja uma matriz diagonal sendo os elementos diagonais os autovalores associados. Assumindo, ainda, que R_x é uma matriz definitivamente positiva, ou seja, que a igualdade final da equação (26) seja diferente de uma matriz nula, têm-se a definição básica da transformada de Karhunen-Loève (KL), gerando atributos mutuamente não relacionados.

Supondo um espaço de atributos de tamanho *N*, cuja transformada KL o levará para o tamanho *n* temos que:

$$x = \sum_{i=0}^{N-1} y(i)a(i) \quad (27)$$

$$y(i) = a_i^T x \quad (28)$$

Definindo o novo subespaço como:

$$\hat{x} = \sum_{i=0}^{m-1} y(i)a(i) \quad (29)$$

Dessa forma, aplicando a aproximação de mínimos erros quadráticos, temos que:

$$Erro[\|x - \hat{x}\|^2] = \sum_{i=m}^{N-1} a_i^T \lambda_i a_i = \sum_{i=m}^{N-1} \lambda_i \quad (30)$$

Sendo λ_i os autovalores associados aos autovetores.

Dessa forma, a partir da escolha dos autovetores associados aos maiores autovalores, a equação (30) é minimizada sendo representada pela soma dos menores autovalores. Por isso a transformada KL é conhecida também como Análise de Componentes Principais (do inglês *Principal Componente Analysis* ou PCA).

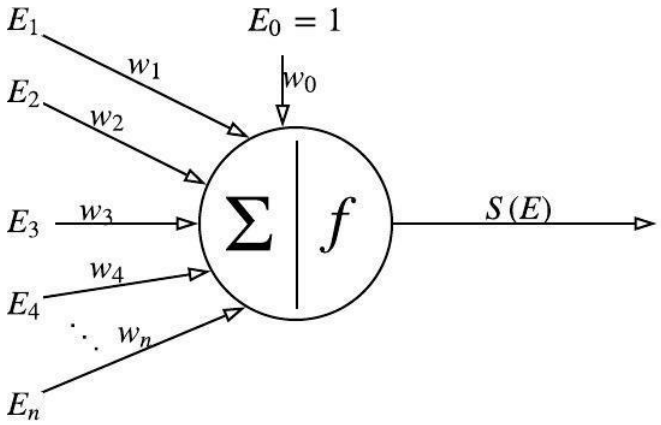
De modo geral, o método PCA utiliza os princípios de álgebra linear com o objetivo de remover redundâncias presentes em conjuntos de dados, de forma a melhorar a performance de determinadas tarefas de aprendizado de máquina.

2.5. Redes Neurais Artificiais

As redes neurais artificiais (ANN do inglês *Artificial Neural Network*) tem o modelo de grafo como característica principal. Nesse modelo os nós são os neurônios artificiais e as arestas são as conexões entre entrada e saída dos neurônios [23].

A Figura 2 representa o modelo do neurônio proposto por esse classificador. Pode-se dizer que $E = [E1, E2, E3, \ldots, En]$ é o vetor que representa o vetor de características, *E1* é a entrada padrão do sistema, $w = [w0, w1, w2, w3, \ldots, wn]$ são os pesos atribuídos a cada entrada, *f* é a função de ativação e, por fim, $S[E]$ é a saída do sistema [24].

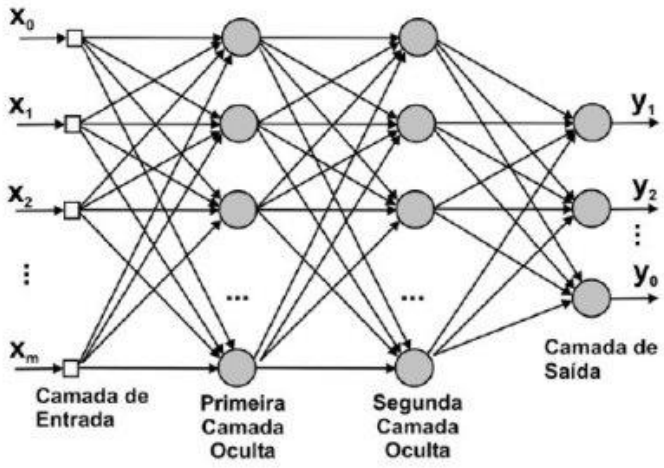
Figura 2 – Representação gráfica de um neurônio artificial



Fonte: O autor

A Figura 2 representa apenas um neurônio, com função de ativação *f*. Contudo, uma ANN pode possuir esquemas complexos com vários neurônios e várias camadas como mostra a Figura 3.

Figura 3 – Representação gráfica de uma ANN



Fonte: [23]

2.5.1. Parametrização de uma Rede Neural Artificial

Diversos parâmetros podem ser modelados para a avaliação de algoritmos de aprendizagem de máquina. Cada tipo de algoritmo possui uma classe de parâmetros que podem ser variados para o aperfeiçoamento do sistema.

Para o desenvolvimento do algoritmo de ANN foram variados quatro parâmetros diferentes, sendo eles:

- Arquitetura da Rede

  Diversas arquiteturas podem ser propostas com diversos neurônios e diversas camadas, tornando as possiblidades infinitas para esse parâmetro. Contudo, nesse trabalho, avaliaremos 3 arquiteturas diferentes, escolhidas arbitrariamente, para a camada intermediária de neurônios, sendo elas: (100, 100, 10), (100,50,10), (50,10). Esses valores se referem às camadas ocultas da rede neural artificial, ou seja, as camadas entre a camada de entrada e a camada de saída. Cada grupo de

valores refere-se a uma arquitetura diferente e cada valor dentro do grupo de valores refere-se ao número de neurônios da camada.

- *Solver*

  Esse parâmetro é responsável pela escolha do algoritmo para optimização de pesos de cada entrada. Para esse trabalho foi considerado apenas um algoritmo para esse propósito: o otimizador baseado em gradiente estocástico, também conhecido como adam.

- Função de Ativação

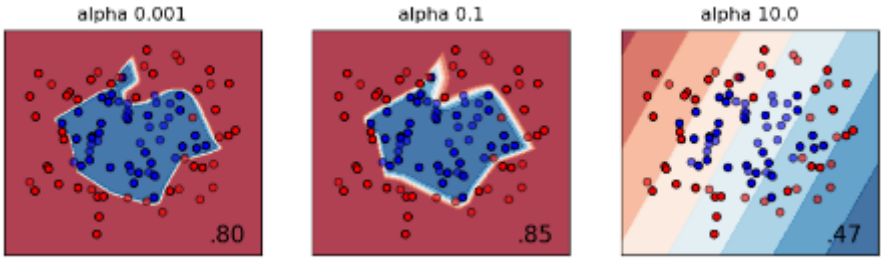
  Esse parâmetro se refere à função utilizada nos neurônios das camadas ocultas. Essa função é responsável por definir o valor de saída do neurônio. A função aplicada no desenvolvimento do trabalho foi a *rectfied linear unit function*.

- Alfa

  Esse parâmetro cumpre um papel de regulação da fronteira de decisão estabelecida. Valores altos para esse parâmetro resultam em uma fronteira de decisão menos flexível combatendo um alto valor de variância. Em contrapartida, valores baixos para o parâmetro proporcionam uma fronteira de decisão mais flexível e mais sensível à presença de dados. A Figura 4 mostra os resultados da variação deste parâmetro.

É importante ratificar a função da Figura 4, visto que algumas versões dela se repetiram ao longo do capítulo. Essa figura mostra a tarefa de classificação aplicada a elementos de duas classes diferentes (azul e vermelho). A área destacada representa a qual classe determinado elemento foi classificado. Ou seja, se algum elemento da classe azul estiver numa área demarcada como vermelho, esse elemento foi erroneamente classificado. A diferença dos tons, mais destacada no exemplo alpha 10.0, indica a probabilidade do elemento ser de determinada classe. Ou seja, quanto mais destacada é a cor, maior a possiblidade de um elemento naquela área ser da classe representada pela cor.

Figura 4 – Conjunto de dados classificado a partir de diferentes valores de alfa

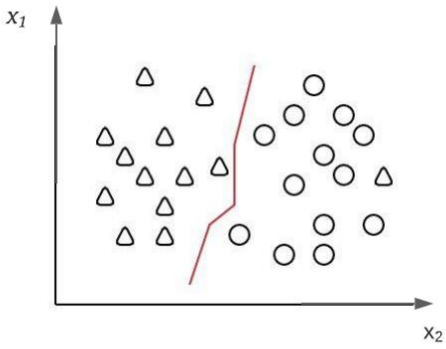


Fonte: [25]

2.6. Máquina de Vetores de Suporte

A máquina de vetores de suporte (SVM do inglês *Support Vector Machine*) é uma técnica de aprendizagem de máquina baseada na representação geométrica dos atributos. Essa abordagem tem como objetivo estabelecer uma função, baseada nas distâncias entre os pontos dessa representação geométrica, capaz de discriminar classes diferentes.

Figura 5 – Representação gráfica de uma fronteira estabelecida pelos vetores de suporte



Fonte: O Autor

A Figura 5 mostra um exemplo simples de classificação de formas geométricas em duas classes distintas. Nesse exemplo o objeto de estudo possui dois atributos, representados pelos eixos horizontal e vertical ($x_1 e\ x_2$). A partir da disposição das formas geométricas, a curva vermelha é definida, sendo ela uma fronteira de decisão para o sistema definir a qual classe o objeto pertence. Essa curva é definida baseada na distância euclidiana entre as formas na representação gráfica.

2.5.1. Parametrização de uma Máquina de Vetores de Suporte

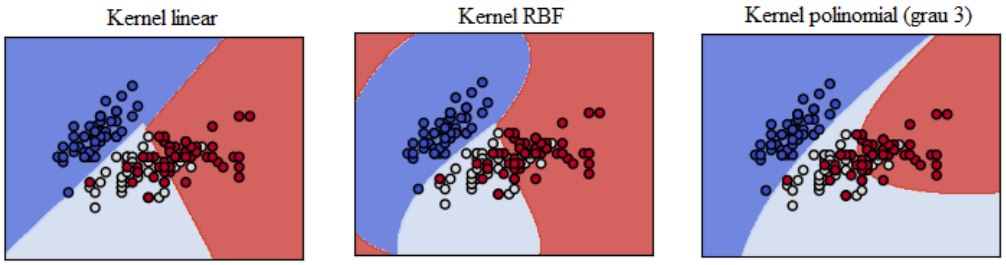
Analogamente a ANN, à máquina de vetores de suporte também possui parâmetros importantes que definem seu funcionamento. Cada combinação de parâmetros gera um resultado exclusivo, tornando a possibilidade de combinações infinita.

Para o desenvolvimento desse trabalho, os parâmetros considerados para a modelagem foram:

- Kernel

  Esse parâmetro se refere ao tipo de função que será utilizada na formatação da fronteira de decisão, em vermelho na Figura 5. A Figura 6 ilustra três exemplos com diferentes funções kernel para a resolução de um problema com três classes.

Figura 6 – Diferentes tipos de Kernel aplicados à tarefa de classificação

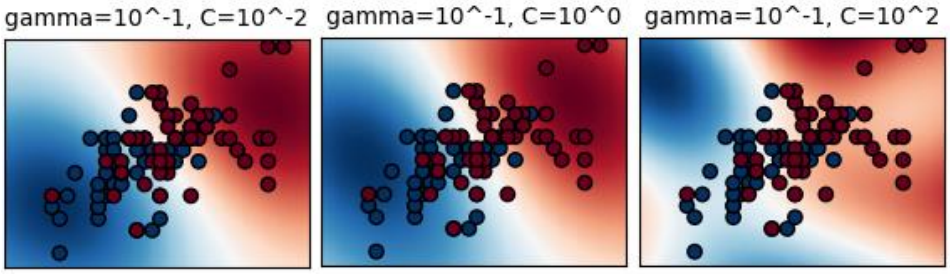


Fonte: [26]

- Parâmetro C

  O parâmetro C determina a sensibilidade do sistema a variações. Ou seja, quanto maior o valor desse parâmetro, maior é a sensibilidade da fronteira de decisão. A vantagem de se ter um sistema menos sensível e mais propenso a erros é a de evitar um possível condicionamento do mesmo a um modelo fixo de dados. A Figura 7 mostra um exemplo de classificação com diferentes valores para esse parâmetro.

Figura 7 – Tarefa de classificação para diferentes valores do parâmetro C

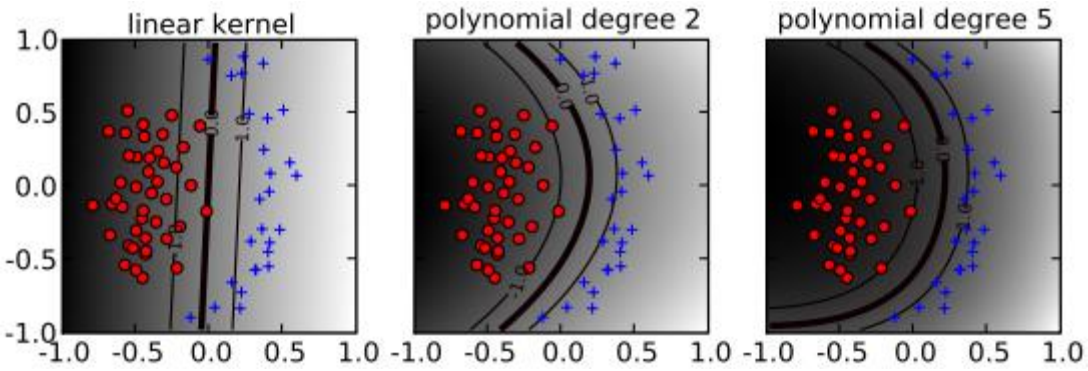


Fonte: [27]

- Grau

  Esse parâmetro promove mudanças apenas no kernel polinomial, não alterando o resultado dos demais. Essa constante se refere ao grau do polinômio utilizado como fronteira de decisão. A Figura 8 mostra a diferença entre diferentes graus para o kernel polinomial.

Figura 8 – Kernel com diferentes graus

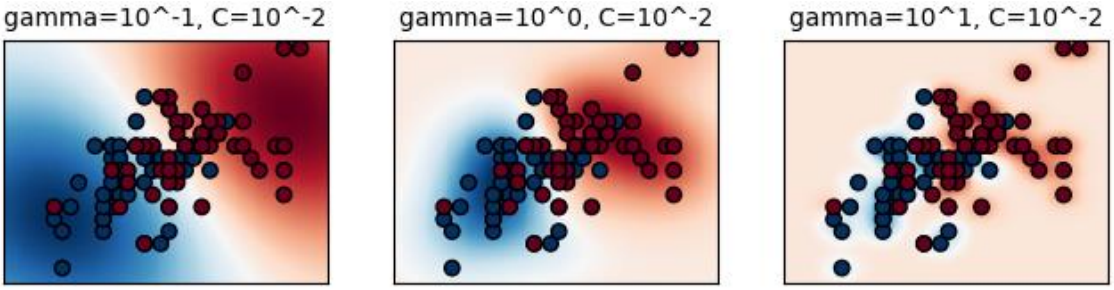


Fonte: [28]

- Gamma

  Entre os tipos de kernel utilizados nesse trabalho, apenas o RBF sofre alterações com a alteração do parâmetro gamma. Esse parâmetro influencia o quanto uma única amostra de treinamento afeta o modelo. O parâmetro gamma pode ser visto como o inverso do raio de influência de uma única amostra. Ou seja, quanto maior essa variável, menor é o raio de influência de uma amostra e quanto menor, maior é a influência dessa amostra na construção do modelo. A Figura 9 mostra como esse parâmetro influencia o modelo de treinamento.

Figura 9 – Kernel RBF com diferentes valores de gamma

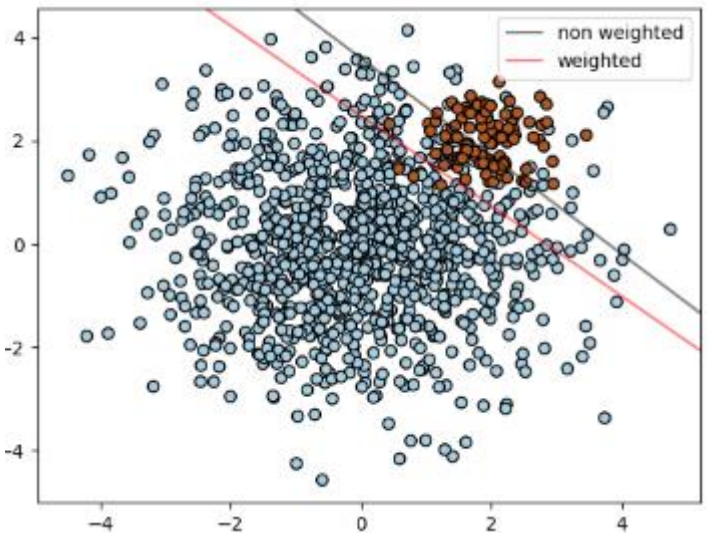


Fonte: [27]

- Peso das classes

  Em problemas cujos dados são desbalanceados, ou seja, há mais amostras de uma classe em relação a outra, é comum que esse parâmetro seja utilizado para atribuir um peso a uma determinada classe. Esse método tem como objetivo corrigir a falta de equilíbrio entre as classes. Esse parâmetro age diretamente no parâmetro C a partir de um produto entre ambos. A Figura 10 mostra a influência desse parâmetro na modelagem do problema.

Figura 10 – Comparação entre uma modelagem com pesos (curva em vermelho) e uma sem pesos (curva em preto)



Fonte: [26]

CAPÍTULO 3

METODOLOGIA

A partir dos conceitos discutidos no capítulo anterior, é necessário que haja uma metodologia por trás da execução do estudo comparativo. Essa etapa do trabalho determina uma arquitetura de processos capaz de orientar o desenvolvimento das duas abordagens que compõe o projeto. Além disso, esse capítulo busca orientar a melhor metodologia para se efetuar o estudo comparativo entre as abordagens, bem como a escolha das melhores métricas de comparação.

3.1 Tratamento dos Dados

A primeira etapa da metodologia se refere à escolha das bases de dados que serão utilizadas na implementação do estudo. O segundo tópico do Capítulo 2 ratifica a importância de se utilizar bases de dados que sejam condizentes com a realidade do fenômeno que está sendo estudado. Contudo, a escolha das fontes de dados não é baseada apenas nisso, é importante que um estudo seja realizado no formato e na disposição dos dados dentro da base.

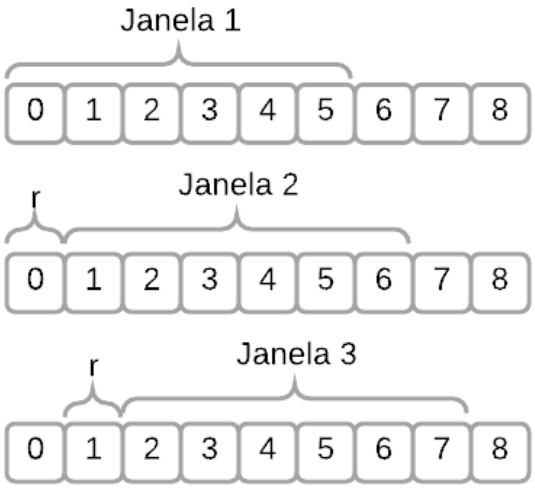
Para este trabalho, foi estabelecido como padrão trabalhar com arquivos no formato *CSV* pela agilidade da análise por meio de ferramentas de controle como o editor de consultas do Excel e, também, pela facilidade de manipulação através da linguagem de programação Python, utilizada ao longo de todo o projeto.

O primeiro passo efetuado no que diz respeito a arquitetura de processos e, que é comum as duas abordagens desenvolvidas nesse trabalho, é a leitura dos arquivos *CSV* e seleção das colunas (atributos) que precisarão ser excluídos da análise, ou transformados de alguma forma.

3.2 Abordagem baseada no método ARIMA

O primeiro passo realizado nessa abordagem diz respeito à implementação de um algoritmo para o janelamento das linhas de dados de acordo com o tempo conforme a Figura 11. Estabeleceu-se como padrão utilizar uma janela de cinco minutos para a realização das análises. Além disso, o processo de janelamento implementado não é estático, ou seja, ele possui um fator de deslizamento de um minuto responsável por uniformizar os dados para análise de forma a detectar a ameaça o quanto antes.

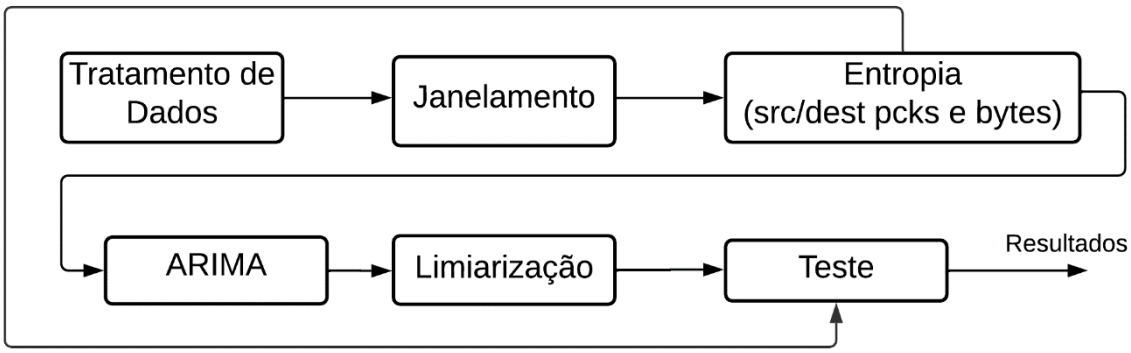
Figura 11 – Demonstração de janelamento a cada 5 minutos com fator de deslizamento r igual a 1 minuto



Fonte: O autor.

As etapas posteriores da abordagem baseada na metodologia ARIMA terão como foco o cálculo da entropia de cada janela de dados e, posteriormente, a aplicação do método para realizar a previsão de entropias de janelas futuras. Em seguida, um processo de limiarização para definição de situações de ataque é efetuado e aplicado nos dados de teste. A Figura 12 mostra o fluxograma do algoritmo proposto.

Figura 12 – Fluxograma de processos da abordagem baseada no método ARIMA



Fonte: O autor

Dessa forma, as situações de ataque serão detectadas a partir de casos que têm valores de entropia que destoam totalmente das situações normais.

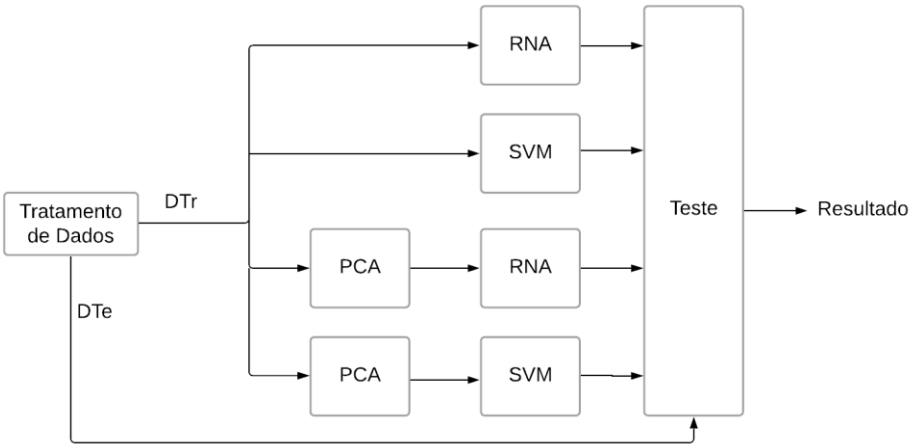
3.3 Abordagem Supervisionada

Na abordagem supervisionada serão implementados os algoritmos de aprendizagem de máquina para o treinamento de um modelo capaz de realizar predições sobre valores futuros. Nessa abordagem a avaliação é feita linha a linha, ou seja, sem a aplicação do janelamento citada na seção anterior. Os algoritmos testados, modelados a partir das informações contidas nas janelas de dados, serão o ANN e o SVM.

Além disso, um outro algoritmo será desenvolvido com o acréscimo de uma etapa anterior ao treinamento do modelo, a etapa PCA, cujo objetivo já descrito anteriormente é reduzir a dimensão dos dados de forma a atribuir peso maior a atributos mais redundantes para a classificação de um determinado dado.

Dessa forma, a Figura 13 mostra o fluxograma de dados capaz de resumir os processos implementados nessa abordagem.

Figura 13 – Fluxograma de processos de abordagem supervisionada



Fonte: O autor

Os dados, após o tratamento necessário, são divididos em duas partições, de forma que uma delas seja utilizada na análise PCA e no treinamento dos algoritmos de aprendizado de máquina, representada na Figura 14 por *DTr*, enquanto a outra é utilizada para a avaliação dos modelos de treinamento, representada por *DTe*.

3.4 Método de Avaliação

A métrica base utilizada para a definir um método ou conjunto de parâmetros como melhor que o outro será o *f-score*, medida obtida através da média geométrica entre a precisão e a sensitividade do modelo como mostra a equação (31).

$$f - score\ (\%) = \frac{2}{sensitividade^{-1} + precisão^{-1}} \times 100 \quad (31)$$

Por sua vez sensitividade e precisão podem ser definidas respectivamente como a eficiência da detecção para identificação de casos positivos e; dos casos classificados como positivos, quantos efetivamente eram positivos. As equações (32) e (33) mostram a forma de calcular ambas as medidas [29].

$$Sensitividade\ (\%) = \frac{Verdadeiros\ Positivos}{Verdadeiros\ Positivos + Falsos\ Positivos} \times 100 \quad (32)$$

$$Precisão\ (\%) = \frac{Verdadeiros\ Positivos}{Verdadeiros\ Positivos + Falsos\ Negativos} \times 100 \quad (33)$$

O primeiro critério de desempate, após a análise da *f-score*, será feito utilizando a acurácia geral, ou seja, a taxa de acertos de cada método. Essa métrica pode ser resumida pela equação (34).

$$Acurácia\ (\%) = \frac{Quantidade\ de\ acertos}{Quantidade\ de\ testes} \times 100 \quad (34)$$

Outra métrica importante para o estudo, que é fator determinante para o desenvolvimento de sistemas de detecção de ameaças dessa natureza, é o tempo de execução dos processos. Essa métrica será utilizada como terceiro critério de desempate.

CAPÍTULO 4

RECURSOS

Os recursos que serão utilizados para o desenvolvimento do algoritmo proposto possuem grande importância. Para a análise visual dos dados e auxílio no desenvolvimento do trabalho, o editor de consultas do Excel foi utilizado. Como linguagem de programação para o desenvolvimento optou-se pela linguagem Python [30], dentro do ambiente de desenvolvimento PyCharm, principalmente pela sintaxe de fácil compreensão e pelo seu uso comum na execução de tarefas voltadas para a aprendizagem de máquina.

Junto com o Python, algumas bibliotecas serão fundamentais para a execução do trabalho de forma mais simples, sendo elas:

- csv: biblioteca responsável pela leitura e operação de arquivos csv;
- ipaddress: biblioteca para conversão de endereço ip para um número inteiro
- Datetime: biblioteca responsável pela leitura e operação de dados no formato de data
- Scipy e Math: para operações matemáticas
- Pandas: para a manipulação de dados
- Numpy: para a operação com vetores e listas
- Matplotlib: para a criação de imagens e gráficos
- StatsModels: para implementação do modelo ARIMA
- Scikit Learning: para a implementação de algoritmos de aprendizagem de máquina
- Excel: utilização do editor de consultas para análise inicial

A partir desses recursos de software, será possível realizar a implementação do estudo proposto por esse trabalho.

CAPÍTULO 5

RESULTADOS E DISCUSSÕES

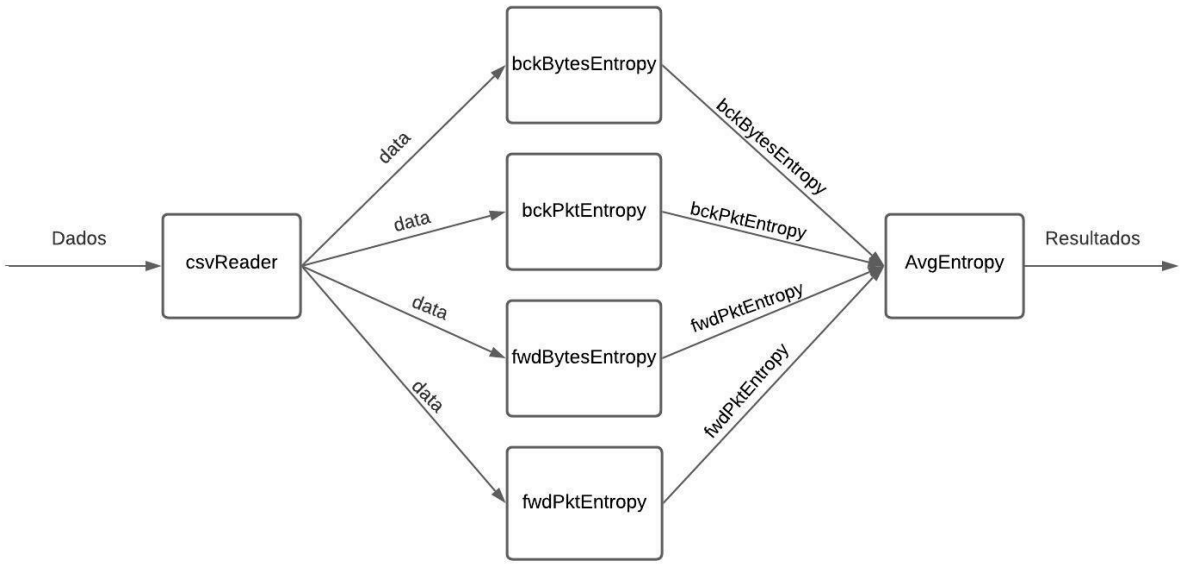
Esse capítulo elucidará a estrutura do código desenvolvido para cada um dos métodos propostos. Além disso, o capítulo trará os resultados para cada um dos métodos de acordo com suas variáveis particulares, bem como comparará os resultados para cada metodologia.

Além disso, nem todos os resultados dos testes foram expostos nesse trabalho devido à grande quantidade de testes realizados. Com isso, foi disponibilizado um diretório [31] dentro da plataforma GitHub para o armazenamento dos scripts utilizados e dos resultados do trabalho.

5.1 Método ARIMA

O método ARIMA foi implementado a partir de seis *scripts* diferentes com o objetivo de melhorar a organização e a documentação do código. A Figura 15 mostra um diagrama que representa a arquitetura do código e o fluxo de dados durante sua execução.

Figura 14 – Fluxograma de scripts para metodologia ARIMA



Fonte: O autor

Durante a implementação do código ficou claro, a partir dos gráficos de entropia ao longo do tempo, que apenas a utilização do limiar inferior bastava para a classificação de um ataque.

A implementação do método, que possui como atributos *p*, *q* e *d* previamente apresentados no Capítulo 3, possibilitou a criação da Tabela 1.

Tabela 1 – Resultados em relação aos atributos do modelo

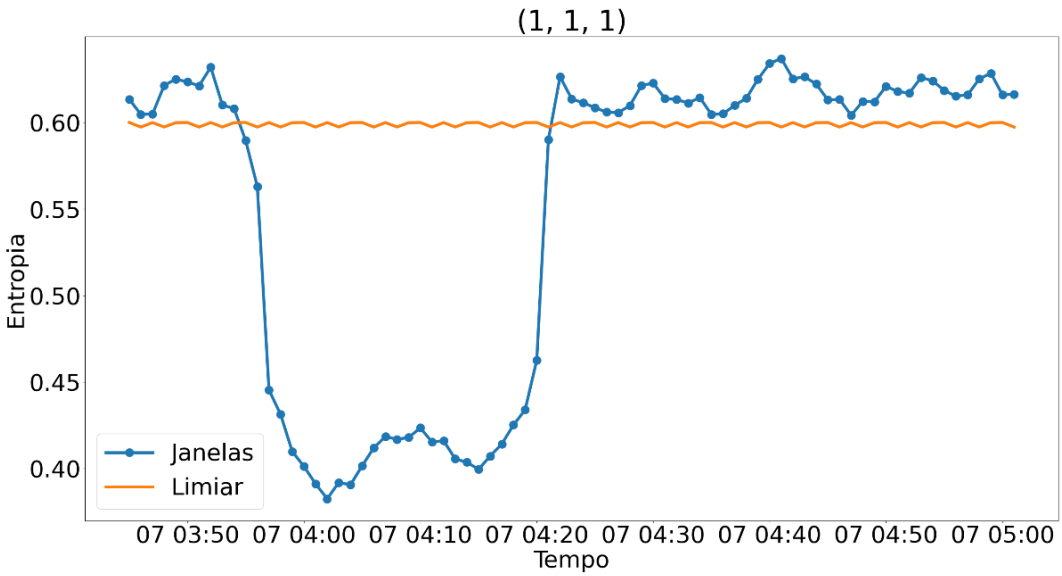
Modelo	*Atributos (p,q,d)*	*f-score (%)*	*Sensitividade (%)*	*Acurácia (%)*	*Precisão (%)*
A	(1, 1, 1) (2, 1, 1) (3, 1, 1) (1, 1, 2) (2, 1, 2) (3, 1, 2) (1, 1, 3) (2, 1, 3) (3, 1, 3) (1, 2, 1) (2, 2, 1) (3, 2, 1) (1, 2, 2) (2, 2, 2) (3, 2, 2) (1, 2, 3) (2, 2, 3) (3, 2, 3)	67,04	69,34	62,02	64,88
B	(1, 3, 1) (2, 3, 1) (3, 3, 1) (1, 3, 2) (2, 3, 2) (3, 3, 2) (1, 3, 3) (3, 3, 3)	59,78	69,96	60,81	59,78
C	(2, 3, 3)	57,66	69,75	58,93	69,75

Fonte: O autor

A partir disso foi possível montar os gráficos das Figuras 15, 16 e 17 para representação da tarefa de seleção através do limiar obtida pelo método ARIMA. É importante frisar que, devido à semelhança entre a maioria dos gráficos e a quantidade de gráficos obtidos, apenas os três casos que possuem o comportamento mais diferente entre si foram escolhidos para compor o trabalho. Além disso, o tempo para construção de cada modelo foi omitido do trabalho devido

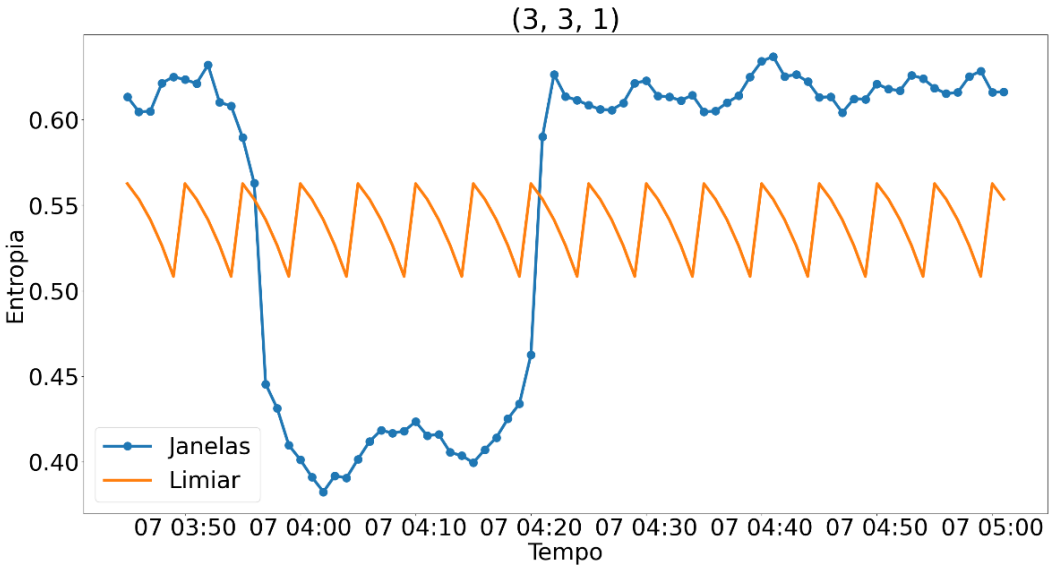
ao baixo tempo de processamento, todos os modelos levaram menos de 1 segundo para serem processados. Além do aspecto visual, os resultados também foram base para a escolha dos gráficos utilizados no trabalho, sendo escolhidos um de cada modelo proposto pela Tabela 1.

Figura 15 – Gráfico de Entropia por Janelas e limiar (1,1,1) em função do tempo



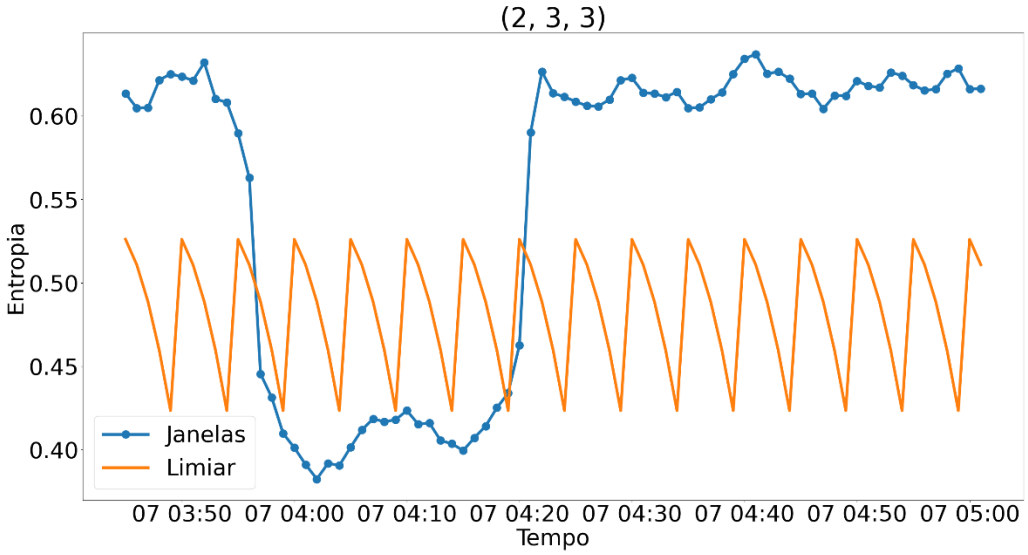
Fonte: O autor

Figura 16 – Gráfico de Entropia por Janelas e limiar (3,3,1) em função do tempo



Fonte: O autor

Figura 17 – Gráfico de Entropia por Janelas e limiar (2,3,3) em função do tempo

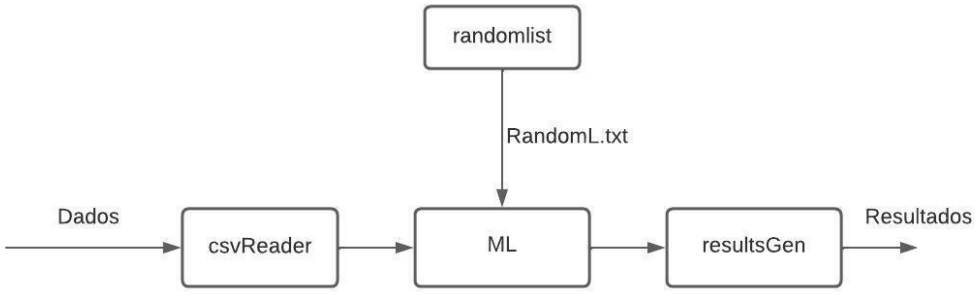


Fonte: O autor

5.2 Método de Aprendizagem Supervisionada

De forma análoga ao método ARIMA, o método de aprendizagem supervisionada foi implementado a partir de módulos para melhorar a organização do código. De modo geral, os dois modelos de classificador foram aplicados a um subconjunto de 10000 dados retirado do conjunto de dados principal e os dados restantes foram utilizados para os testes. Foram utilizados 5 vetores de mapeamento, definidos aleatoriamente, para realização de testes utilizando diferentes conjuntos de treinamento.

Figura 18 – Fluxograma do código



Fonte: O autor

O fluxograma da Figura 18 mostra a execução dos *scripts*. O bloco ML se refere aos diferentes métodos de aprendizagem aplicados.

Devido ao grande número de testes realizados, os modelos foram divididos nas seguintes categorias: SVM, ANN, SVM+PCA e ANN+PCA. A partir dessa divisão e dos critérios de seleção destrinchados no item 3.4 deste trabalho, os seguintes modelos foram selecionados:

Tabela 2 – Métricas em relação ao método utilizado

Método	*Atributos*	*f-score (%)*	*Sensitividade (%)*	*Acurácia (%)*	*Precisão (%)*	*Tempo (s)*
ANN	Arquitetura: 50,10 Alfa: 0,001	99,77	99,60	99,69	99,94	83,0
ANN+PCA	Arquitetura: 100,50,10 Alfa: 0,001 Componentes: 49	99,99	99,99	99,99	99,99	108,0
SVM	Tipo: Polinomial C: 0,001 Grau: 4	77,29	100	66,37	63,14	114,0
SVM+PCA	Tipo: RBF C: 1000 Gamma: 0,1 Componentes: 35	99,99	99,98	99,98	99,99	44,0

Fonte: O autor

Com o objetivo de entender o impacto da aplicação da PCA, a Tabela 3 mostra as métricas dos dois modelos utilizados na Tabela 2 em ANN+PCA e SVM+PCA sem a aplicação da PCA.

Tabela 3 – Métricas dos modelos da Tabela 2 sem a aplicação da PCA

Método	*Atributos*	*f-score (%)*	*Sensitividade (%)*	*Acurácia (%)*	*Precisão (%)*	*Tempo (s)*
ANN	Arquitetura: 100,50,10 Alfa: 0,001	99,73	99,5	99,63	99,97	83,0
SVM	Tipo: RBF C: 1000 Gamma: 0,1	0	0	43,27	0	83,0

Fonte: O autor

Dessa forma, comparando os resultados das Tabelas 1 e 2 o método ANN+PCA com redução de dimensionalidade para 49 componentes, arquitetura (100,50,10) e alfa igual a 0,001 e foi o que demonstrou maior desempenho na tarefa proposta, utilizando os critérios estipulados no item 3.4. Além disso, é possível perceber a efetividade da aplicação do método PCA para esse tipo de aplicação. Ao comparar as Tabelas 2 e 3 é notável a melhora de todas as métricas.

Se comparado ao método SVM+PCA, as métricas se aproximam consideravelmente, sendo superadas pelo método ANN+PCA por apenas um centésimo no critério acurácia. Contudo, o tempo de processamento do método SVM+PCA é cerca de 40% menor se comparado ao ANN+PCA.

CAPÍTULO 6

CONSIDERAÇÕES FINAIS

Este trabalho apresentou diferentes metodologias de reconhecimento de padrões aplicadas a tarefa de detecção de ataques DDoS. Inicialmente foi realizada uma análise exploratória dos dados preliminar com o objetivo de escolher a base de dados a ser utilizada no trabalho, sendo a selecionada a UNB CIC-IDS 2017, destrinchada no item 2.2.1 deste trabalho. Após a análise exploratória, realizada com o editor de consultas do Excel, a base de dados passou por um processo de seleção e normalização com o objetivo de transformá-la em uma base capaz de ser utilizada nos processos seguintes.

A metodologia ARIMA, utilizando a entropia em janelas de tempo como base para o estabelecimento de limiares para casos normais e anormais demonstrou um f-score máximo de 67,04% com acurácia geral de 62,02% e tempo médio de processamento menor que 1 segundo.

Por outro lado, os métodos clássicos de *machine learning* se mostraram extremamente eficientes quando aliados a uma boa escolha de parâmetros. Dentre as centenas de testes realizados combinando os parâmetros pré-estabelecidos a combinação que mais obteve sucesso foi a ANN+PCA, obtendo f-score de 99,99% e acurácia geral de 99,99%. Contudo é importante frisar que o modelo SVM+PCA obteve métricas significativamente importantes, pois igualou a maioria das métricas, perdendo apenas por um centésimo na medida de acurácia geral do modelo ANN+PCA, mas obteve um tempo de treinamento menor que o modelo.

Outra consideração importante a ser feita é o impacto positivo que a análise de principais componentes proporcionou, sendo ela imprescindível para a melhora das métricas da maioria dos modelos testados.

Para trabalhos futuros, sugere-se a aplicação das técnicas analisadas em novas bases de dados. Outra possibilidade é utilizar seletores de dados para melhorar a eficiência do treinamento utilizando menos amostras. A realização de testes em tempo real permitiria uma melhor avaliação do desempenho dos métodos. Outra sugestão é a aplicação de outros métodos de classificação. O uso de árvores de decisão, por exemplo, poderia facilitar a interpretação do processo de decisão, permitindo a identificação dos atributos mais importantes nesse processo. Também poderiam ser usadas técnicas mais avançadas, como redes neurais convolucionais. Em

qualquer caso, sugere-se o uso de uma base de dados disponibilizada remotamente para maior facilidade de interação a partir de outros computadores.

REFERÊNCIAS

[1] M. Idhammad, K. Afdel e M. Belouch, “Semi-supervised machine learning approach for DDoS detection,” *Applied Intelligence,* pp. 3193-3208, 2018.

[2] J. Snow, “Os ciberataques mais famosos dos últimos tempos,” Kaspersky Lab, 7 Novembro 2018. [Online]. Available: https://www.kaspersky.com.br/blog/five-most-notorious-cyberattacks/11042/. [Acesso em 27 Julho 2020].

[3] Kaspersky, “Cibercriminosos estão recorrendo às técnicas mais sofisticadas de ataque DDoS,” Kaspersky Lab, 12 Fevereiro 2019. [Online]. Available: https://www.kaspersky.com.br/about/press-releases/2019_kaspersky-lab-ddos-intelligence-report-para-traducao-br. [Acesso em 27 Julho 2020].

[4] Kaspersky, “O que são ataques de DDoS?,” Kaspersky Lab, 25 Abril 2013. [Online]. Available: https://www.kaspersky.com.br/resource-center/threats/ddos-attacks. [Acesso em 27 Julho 2020].

[5] Kaspersky, “Global Security Risks Survey 2014 - Distributed Denial of Service (DDoS) attacks,” Kaspersky Lab, Moscou, 2014.

[6] G. Somani, M. S. Gaur, D. Sanghi, M. Conti e R. Buyya, “DDoS attacks in cloud computing: Issues, taxonomy, and future directions,” *Computer Communications,* vol. 107, pp. 30-48, 2017.

[7] N. Jeyanthi, N. C. S. Iyengar, P. M. Kumar e A. Kannammal, “An Enhanced Entropy Approach to Detect and,” *International Journal of Communication Networks and Information Security,* vol. 5, nº 2, p. 110, 2013.

[8] M. A. S. Monge, J. M. Vidal e L. J. G. Villalba, “Entropy-Based Economic Denial of Sustainability Detection,” *Entropy,* vol. 19, p. 649, 2017.

[9] K. Bhushan e B. B. Gupta, “Hypothesis test for low-rate DDoS attack detection in cloud computing environment,” *Procedia computer science,* vol. 132, pp. 947-955, 2018.

[10] G. E. P. Box, G. M. Jenkins, G. C. Reinsel e G. M. Ljung, Time series analysis, forecasting, and control, Hoboken: Wiley, 2016.

[11] R. H. Shumway e D. S. Stoffer, Time Series Analysis and Its Applications: With R Examples, Gewerbestrasse: Springer, 2016.

[12] G. a. S. G. Somani, M. S. Gaur, D. Sanghi, M. Conti e R. Buyya, "DDoS attacks in cloud computing: Issues, taxonomy, and future directions," *Computer Communications,* vol. 107, pp. 30-48, 2017.

[13] I. Sharafaldin, A. H. Lashkari e A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," em *4th International Conference on Information Systems Security and Privacy (ICISSP),* Madeira, 2018.

[14] I. Sharafaldin, A. H. Lashkari, S. Hakak e A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," em *IEEE 53rd International Carnahan Conference on Security Technology,* Chennai, 2019.

[15] G. D. Gil, A. H. Lashkari, M. Mamun e A. A. Ghorbani, "Characterization of Encrypted and VPN Traffic Using Time-Related Features," em *2nd International Conference on Information Systems Security and Privacy (ICISSP 2016),* Roma, 2016.

[16] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun e A. A. Ghorbani, "Characterization of Tor Traffic Using Time Based Features," em *3rd International Conference on Information System Security and Privacy,* Porto, 2017.

[17] A. Gharib, I. Sharafaldin, A. H. Lashkari e A. A. Ghorbani, "An Evaluation Framework for Intrusion Detection Dataset," em *2016 International Conference on Information Science and Security (ICISS),* Pattaya, 2016.

[18] S. Haykin, Sistemas de Comunicação analógicos e digitais, São Paulo: Bookman, 2007.

[19] A. L. S. Maia, F. d. A. T. de Carvalho e T. B. Ludermir, "Forecasting models for interval-valued time series," *Neurocomputing,* vol. 71, pp. 3344-3352, 2008.

[20] A. B. Koehler, R. D. Snyder e J. K. Ord, "Forecasting models and prediction intervals for the multiplicative," *International Journal of Forecasting,* vol. 17, p. 269–286, 2001.

[21] M. H. Khan, N. S. Muhammad e A. El-Shafie, “Wavelet based hybrid ANN-ARIMA models for meteorological drought,” *Journal of Hydrology*, vol. 590, 2020.

[22] J. J. Selvaraj, V. Arunachalam, K. V. Coronado-Franco, L. V. Romero-Orjuela e Y. N. Ramírez-Yara, “Time-series modeling of fishery landings in the Colombian Pacific Ocean using an ARIMA model,” *Regional Studies in Marine Science*, vol. 39, 2020.

[23] H. Pedrini e W. R. Schartz, Análise de Imagens Digitais: princípios, algoritmos e aplicações, São Paulo: Thomson Learning, 2008.

[24] A. O. Artero, Inteligência Artificial: Teoria e Prática, São Paulo: Editora Livraria da Física, 2009.

[25] Scikit-learn, “Scikit-learn: Machine Learning in Python,” Scikit-learn, 28 Setembro 2020. [Online]. Available: https://scikit-learn.org/stable/auto_examples/neural_networks/plot_mlp_alpha.html. [Acesso em 28 Setembro 2020].

[26] Scikit-learn, “Scikit-learn: Machine Learning in Python,” Scikit-learn, 3 Agosto 2020. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html. [Acesso em 28 Setembro 2020].

[27] Scikit-learn, “Scikit-learn: Machine Learning in Python,” Scikit-learn, 3 Agosto 2020. [Online]. Available: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html. [Acesso em 18 Setembro 2020].

[28] A. Ben-hur e J. Weston, “A user’s guide to support vector machines,” *Citeseer*, 2007.

[29] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 59, no. 4, pp. 427-437, 2009.

[30] G. Van Rossum e F. L. Drake Jr, Python 3 Reference Manual, Scotts Valley, CA: CreateSpace, 2009.

[31] J. G. Teles, “GitHub,” 15 Fevereiro 2021. [Online]. Available: https://github.com/JoaoGNT/DDoS_Detection. [Acesso em 16 Março 2022].