UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO

Victor Matos Watanabe

Análise de Detecção de Intrusões Usando Clusterização

Bauru

2019

Victor Matos Watanabe

Análise de Detecção de Intrusões Usando Clusterização

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru. Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Bauru
2019

Victor Matos Watanabe
Análise de Detecção de Intrusões Usando Clusterização/ Victor Matos Watanabe. – Bauru, 2019-
46 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Monografia (Graduação) – Universidade Estadual Paulista "Júlio de Mesquita Filho". Faculdade de Ciências, Bauru, 2019
1. Sistemas de Detecção de Intrusão 2. Clusterização 3. KDD CUP 99 4. Seleção de atributos 5. Detecção de anomalias

VICTOR MATOS WATANABE

ANÁLISE DE DETECÇÃO DE INTRUSÕES USANDO CLUSTERIZAÇÃO

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru. Orientador: Prof. Dr. Kelton Augusto Pontara da Costa, pela seguinte banca examinadora:

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa
Departamento de Computação
Faculdade de Ciências
UNESP - BAURU

Profa. Dra. Simone das Graças Domingues Prado
Departamento de Computação
Faculdade de Ciências
UNESP - BAURU

Profa. Assoc. Roberta Spolon
Departamento de Computação
Faculdade de Ciências
UNESP - BAURU

Bauru, doze de junho de 2019

Para todos os que acreditam em um mundo onde todos possam acreditar.

Agradecimentos

Agradeço veementemente a todos os meus familiáres, em especial a Natalia, minha incrível companheira de todas as horas, aos meus pais, por seu apoio intenso a essa jornada incrível que estou a caminhar, aos meus companheiros de universidade, e aos meus professores, que me instruíram e me instigaram a aprender sobre o que eu mais gosto: computação.

"Wisdom's a gift, but you'd trade it for youth"
- Erza Koenig

Resumo

Com o constante crescimento do uso de ferramentas computacionais e de sistemas ligados a redes, o crescimento de ataques e invasões se torna pertinente.

Os Sistemas de Detecção de Intrusão (ou SDI) surgiram como uma camada de proteção contra ataques, analisando conjuntos de dados de fontes confiáveis (como o próprio sistema observado) a fim de detectar as intrusões geralmente por meio das características dos ataques.

Detecção de anomalias é um dos conceitos que os SDIs podem aplicar. Esta técnica aplicada a SDIs visa a detecção de ataques por meio da detecção de atividades que fogem de padrões de atividades normais ao sistema.

A clusterização é uma das formas utilizadas para que os Sistemas de Detecção de Intrusão consigam detectar padrões de atividades normais (e anormais). Este método utiliza dos conceitos de agrupamento de dados.

Neste trabalho, são discutidas e analisadas técnicas para aprimoramento da clusterização no meio de SDIs, utilizando técnicas como a seleção de atributos mais relevantes deste conjunto de dados, bem como técnicas de detecção de *clusters* normais ou anômalos. Também são vistos técnicas e métodos para uma melhor utilização do conjunto de dados *KDD CUP 99* para o uso em clusterização.

As técnicas de seleção de atributos geram resultados próximos ao original tanto na taxa de detecção quanto na taxa de falsos-positivos, enquanto diminuem o tempo de treinamento consideravelmente. As técnicas aplicadas de rotulamento de *clusters* atingem resultados que melhoram a precisão geral, reduzindo o número de falsos-positivos e mantendo a taxa de detecção estável.

Palavras-chaves: Sistemas de Detecção de Intrusão, clusterização, KDD CUP 99, seleção de atributos, detecção de anomalias

ABSTRACT

As the use of computational tools and systems networks are in constant growth, the growth of attacks and intrusions becomes relevant.

The Intrusion Detection Systems (or IDS) emerged as a layer of protection against attacks, analyzing trustworthy datasets to detect those intrusions by its features and attributes.

Anomaly detection is one of the concepts that the IDSs can apply to. This technique applied on IDSs aim to detect attacks by detecting activities that doesn't fit the normal activities patterns.

Clustering is one of the techniques that are used so that Intrusion Detection Systems can detect patterns for those normal activities (and for the abnormal ones too). This method uses the concept of grouping data.

In this undergraduate thesis, techniques to enhance the clustering itself on the IDS scope are discussed and analyzed, using techniques such as feature selection of the most relevant attributes of the dataset, and normal and abnormal detection methods. Also, methods for a better use of the KDD CUP 99 dataset are discussed.

The feature selection techniques generate results that are close to the original ones in both the detection rate as the false-positive rate, while decreasing the training time. The cluster labeling techniques applied generate results that increase the general precision, reducing the false-positive rate and maintaining the detection rate stable.

Key-words: Intrusion Detection Systems, clustering, KDD CUP 99, feature selection, anomaly detection

Lista de ilustrações

Lista de tabelas

Sumário

1 INTRODUÇÃO

Com a crescente difusão das tecnologias e redes computacionais, cresce também a preocupação com a segurança das informações armazenadas nos computadores conectados a elas (CANSIAN, 1997). Com este crescimento, a confiabilidade de serviços, junto com a sua qualidade de disponibilização, são cada vez mais imprescindíveis na atual conjuntura (SOUSA, 2008).

Segundo Debar (2000, tradução nossa), “Sistemas de deteção de intrusões emergiram na área da segurança de computadores por causa da dificuldade de garantir que um sistema será livre de falhas.”

Estes sistemas, os Sistemas de Detecção de Intrusão (SDI) surgiram para suprir uma necessidade de sistemas modernos: combater intrusões. A detecção de intrusões funciona como uma camada extra de defesa de sistemas (CHEBROLU; ABRAHAM; THOMAS, 2005). Segundo Perlin, Nunes e Kozakevicius (2011), “Após a detecção de um ataque, um SDI deve gerar uma resposta, que pode ser uma intervenção automatizada no sistema ou um alerta para intervenção humana.”

Uma intrusão bem sucedida pode causar diversos problemas aos sistemas atingidos, bem como sistemas adjacentes, como uso de recursos não autorizados, negação de serviços, falta de confiabilidade e de integridade de dados (CHEBROLU; ABRAHAM; THOMAS, 2005).

Clusterização, segundo Jain, Murty e Flynn (1999, tradução nossa), "é a organização de uma coleção de padrões (geralmente representados como um vetor de medidas, ou um ponto em um espaço multidimensional) em *clusters* baseados em similaridade".

Neste trabalho, é utilizado o *dataset KDD CUP 99* para o treinamento não supervisionado e testes do modelo, através do algoritmo de clusterização *k-means*.

Este trabalho é motivado pela importância da detecção de intrusões atualmente, bem como levantar considerações práticas quanto ao desempenho e relevância das técnicas de pré-processamento para os SDIs baseados em anomalia. A clusterização e, em específico, o k-means, são muito usados em estudos sobre o tema, e as informações geradas por este trabalho são de alta relevância para o âmbito acadêmico.

1.1 TAXONOMIA

Para um maior entendimento dos termos utilizados neste trabalho, encontram-se as seguintes definições.

- Definição 1: um ataque, na visão de um sistema, é uma atividade que pode causar consequências ao sistema, enquanto na visão de um invasor é o meio de se conseguir um objetivo no sistema (ALLEN et al., 2000).
- Definição 2: uma invasão é um ataque bem-sucedido (ALLEN et al., 2000).
- Definição 3: um alerta *true-positive* é quando um ataque é detectado (BOLZONI, 2009).

- Definição 4: um alerta *false-positive* é um alerta para um comportamento normal (BOLZONI, 2009).
- Definição 5: um alerta *true-negative* é a ausência de alerta para um comportamento normal (BOLZONI, 2009).
- Definição 6: um alerta *false-negative* é a ausência de um alerta para um ataque (BOLZONI, 2009).

1.2 OBJETIVOS

Os objetivos deste trabalho estão descritos a seguir.

1.2.1 Objetivo geral

O objetivo geral deste trabalho foi analisar e gerar resultados de experimentos de intrusões em rede por meio de detecção de anomalias utilizando o *dataset KDD CUP 99* e o método *k-means* de clusterização, comparando resultados com técnicas de pré-processamento como filtragem de dados, seleção de atributos, normalização de dados e rotulamento de *clusters*.

1.2.2 Objetivos específicos

Os objetivos específicos deste trabalho foram:

- Gerar um subconjunto do *dataset KDD CUP 99* que será apropriado para trabalhos relacionados à clusterização e detecção de anomalias.
- Gerar perspectivas e noções relacionadas a seleção de parâmetros e métodos de pré-processamento e de detecção de anomalias para futuras pesquisas.

Para cumprir estes objetivos, os passos realizados são descritos nos próximos capítulos, com uma intrudução a sistemas de detecção de intrusão e clusterização, bem como a metodologia e os resultados do trabalho.

2 SISTEMAS DE DETECÇÃO DE INTRUSÃO

Sistemas de Detecção de Intrusão, ou SDI, são sistemas que monitoram e analisam a atividade de um computador ou rede, selecionando atividades suspeitas (BACE; MELL, 2001). Um SDI deve monitorar um sistema e interpretar se uma atividade é legítima ou se é uma atividade potencialmente maliciosa (DEBAR; DACIER; WESPI, 1999).

Detecção de intrusões, por sua vez, é o conceito de identificar intrusos a um sistema (usuários sem permissão de utilização) ou usuários com permissões abusando de recursos ou informações de um sistema (MUKHERJEE; HEBERLEIN; LEVITT, 1994).

> "Detecção de intrusão é o processo de monitoração de eventos que ocorrem em um sistema computacional ou rede os analisa os sinais de intrusão, que são as tentativas de comprometer a confidencialidade, integridade, disponibilidade, ou para burlar os mecanismos de segurança de um computador ou rede." (BACE; MELL, 2001, p. 5, tradução nossa)

Segundo Bass (2000), os SDI tem como objetivo a proteção de dados contra ataques de negação de serviço, acesso não desejável de informações e a modificação ou deleção de dados. É considerado um ataque uma atividade que viole os seguintes princípios de segurança: disponibilidade, confidencialidade, integridade e controle (BACE; MELL, 2001).

A detecção de intrusões assume que o usuário e as aplicações de um sistema são observáveis, e atividades normais tem comportamento diferente de atividades anormais (ZHANG; LEE, 2000).

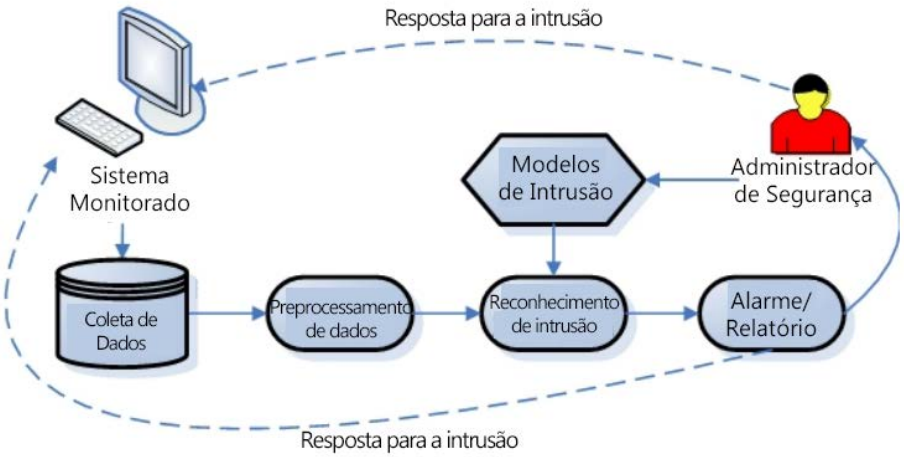
Bace e Mell (2001) cita os seguintes itens como motivos para se utilizar SDIs em sistemas:

- prevenir problemas aumentando a percepção do sistema quanto a intrusões;
- detectar problemas que não são prevenidos por outras medidas de segurança;
- detectar vulnerabilidades a ataques;
- documentar ameaças existentes;
- controle de qualidade de segurança e administração;
- prover informações úteis sobre intrusões.

Para que um SDI seja preciso nas detecções, a fonte dos dados em que este deve se basear para tomar decisões deve ser confiável e vasta (KEMMERER; VIGNA, 2002). Uma atividade também precisa ter informações precisas e confiáveis para que o sistema possa analisar e classificar esta atividade (KEMMERER; VIGNA, 2002).

A Figura 1 mostra como um SDI genérico se comporta durante suas fases de execução.

Figura 1: Organização de um SDI genérico



Fonte: Wu e Banzhaf (2010, tradução nossa)

2.1 ATAQUES

Para definir um ataque, deve-se considerar os seguintes pontos de vista (MCHUGH; CHRISTIE; ALLEN, 2000):

- no ponto de vista de um intruso, um ataque é caracterizado pela intenção e pelo risco de exposição;
- no ponto de vista de uma vítima, um ataque é caracterizado pela manifestações que a intrusão pode causar, podendo ser percebida ou não.

Kendall (1999) cita alguns exemplos de como um intruso pode comprometer um sistema:

- Engenharia social: um intruso ganha acesso por conseguir informação privilegiada para conseguir acessar o sistema;
- Implementação de *bug*: o intruso explora uma vulnerabilidade de um sistema ou programa para conseguir se aproveitar de brechas de segurança através de erros;
- Abuso de recurso: um intruso ganha acesso ao sistema por abusar de recursos do sistema, causando falhas e erros;
- Má-configuração do sistema: um intruso ganha acesso ao sistema por falhas de configuração, como usuários sem senha;
- Mascarando informações: um intruso pode ganhar acesso ao sistema mascarando informações, como a alteração de informações em protocolos de rede.

> "Um ataque é malsucedido, do ponto de vista do intrusor, se nenhum dos seus objetivos são satisfeitos; enquanto isso, a vítima vê um ataque como malsucedido caso não hajam consequências que resultam do ataque." (ALLEN et al., 2000, p. 5, tradução nossa)

2.2 SISTEMAS *HOST-BASED* X SISTEMAS *NETWORK-BASED*

Uma forma comum de se classificar Sistemas de Detecção de Intrusões é entre *host-based* (ou baseado no hospedeiro) e *network-based* (ou baseado na rede). Sistemas de detecção de intrusão *host-based*, como o nome sugere, monitoram o sistema hospedeiro utilizando logs de sistema e analisando recursos (BOLZONI, 2009). Dentre as suas vantagens, nota-se a funcionalidade independente da rede e a monitoração de recursos que os SDIs *network-based* não tem acesso; porém, como desvantagem, nota-se a baixa escalabilidade, especificações de cada host, vulnerabilidade a ataques específicos como os do tipo *Denial of Service*, imensa quantidade de recursos a serem analisados e a perda de performance do hospedeiro (BACE; MELL, 2001).

Os SDIs *network-based*, por sua vez, monitoram dados oriundos da rede. Entre as suas vantagens, nota-se a alta escalabilidade (uma vez que um SDI pode verificar a rede de diversos sistemas), a preservação de performance em hosts. baixo impacto na rede e a baixa percepção por parte dos ataques; entretanto, podem sofrer com picos de tráfego na rede, necessidade de compatibilidade com a rede e os equipamentos envolvidos, a ineficiência com dados criptografados, a incerteza quanto ao sucesso de um ataque (BACE; MELL, 2001).

2.3 TÉCNICAS DE DETECÇÃO

A detecção de intrusões são predominantemente divididas entre duas categorias: detecção por anomalia (ou *anomaly detection*) e detecção por abuso (ou *misuse detection*) (KUMAR, 1995).

A detecção por abuso é caracterizada pela detecção de ataques pela sua assinatura, ou seja, suas principais características previamente conhecidas que se diferem de outros ataques (AXELSSON, 2000). Este método utiliza algoritmos codificados específicos criados por humanos para a detecção (PORTNOY, 2000).

A detecção por anomalia detecta a qualquer atividade que não seja considerada normal (AXELSSON, 2000). Este método cria modelos de atividades consideradas normais e detecta anomalias quando um dado não se enquadra neste modelo (WU; BANZHAF, 2010), e assume que intrusões são altamente relacionadas a comportamento anormal tanto por parte do usuário quanto pela aplicação ou sistema (GHOSH; WANKEN; CHARRON, 1998).

A detecção por abuso tem como principal característica a produção de poucos falso-positivos (ataques classificados como atividades normais ou atividades normais classificadas como ataques), por ter um análise focada em ataques pré-estabelecidos, porém, fica apenas restrito a eles (KEMMERER; VIGNA, 2002). A detecção por anomalia, por sua vez, pode detectar ataques desconhecidos até então, porém, produz uma quantidade maior de *false-positives* (BOLZONI, 2009).

Como a detecção por abuso consegue detectar intrusões com uma taxa baixa de *false-positives* (atividades normais classificadas como ataques), esta técnica é muito utilizada em sistemas comerciais, mas deixa os sistemas mais vulneráveis a mudanças de comportamento por parte dos ataques (WU; BANZHAF, 2010).

2.4 REAÇÕES DOS SISTEMAS DE DETECÇÃO DE INTRUSÃO

Uma vez detectada a intrusão, um SDI pode reagir conforme a sua configuração. Segundo Bace e Mell (2001), esta reação pode ser classificada como:

- ativa: o SDI coleta dados adicionais do ataque, podendo ainda modificar o ambiente para que o ataque não avance para outras camadas do sistema e até tentar atacar o intruso de volta, a fim de conseguir mais informações sobre ele;
- passiva: o SDI emite alarmes ou notificações para os administradores do sistema.

3 CLUSTERIZAÇÃO

Clusterização é um conceito de agrupar ou encontrar grupos em uma dada massa de dados, a fim de que cada item dessa base se agrupe com outros itens que se assemelham ou se aproximam (KAUFMAN; ROUSSEEUW, 2009). Segundo Vesanto e Alhoniemi (2000, p. 586, tradução nossa), "na clusterização, em sua raiz, cada dado pertence a um *cluster*, exatamente".

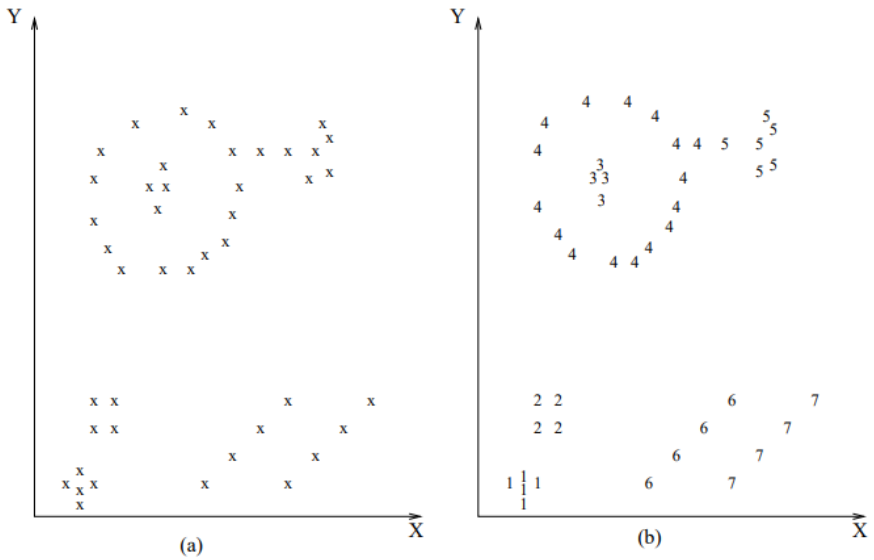
Segundo (JAIN; MURTY; FLYNN, 1999, p. 2, tradução nossa), "clusterização é útil em vários casos de análises exploratórias de padrões, agrupamento, tomada de decisão, e aprendizado de máquina, incluindo mineração de dados, recuperação de documentos, segmentação de imagens, e reconhecimento de padrões". Ainda, métodos de clusterização estão sendo cada vez mais utilizados em outras áreas de estudos, como biologia, psicologia, arqueologia, geologia e *marketing* (JAIN; DUBES, 1988).

Na área da ciência da computação, os métodos de clusterização são muito úteis para resolver problemas como a segmentação de imagem, reconhecimento de pessoas e objetos, recuperação de documentos e *data mining*, por exemplo (JAIN; MURTY; FLYNN, 1999).

A clusterização é uma técnica de aprendizado muito utilizada para casos onde o aprendizado será não-supervisionado, pois os algoritmos não precisam saber rótulos de dados (WAGSTAFF et al., 2001). A clusterização vai separar grupos, que posteriormente poderão ser rotulados de acordo com o método necessário.

A Figura 2 mostra um exemplo de clusterização, encontrando 7 *clusters* em um plano de coordenadas.

Figura 2: Clusterização de dados



Fonte: Jain, Murty e Flynn (1999)

Os processos gerais de uma clusterização são descritos por Jain, Murty e Flynn (1999), onde "padrão" é um dado que contém um vetor de atributos:

- representação de padrões (os dados, em si, e sua extração ou obtenção);
- definição de medidas de aproximação de padrões apropriados para os dados;
- agrupamento (ou clusterização);
- abstração de dados, caso necessário;
- análise de saída, caso necessário.

A figura 3 mostra um fluxo de clusterização genérico.

Figura 3: Fluxo de clusterização

Fonte: Jain, Murty e Flynn (1999, tradução nossa)

3.1 ALGORITMO *K-MEANS*

O método de clusterização *k-means* é bastante comum em estudos que envolvem algoritmos de clusterização, e isso se deve ao fato de este método ser facilmente programado e por ser econômico computacionalmente (MACQUEEN, 1967).

Jain, Murty e Flynn (1999) descrevem o algoritmo *k-means* da seguinte forma:

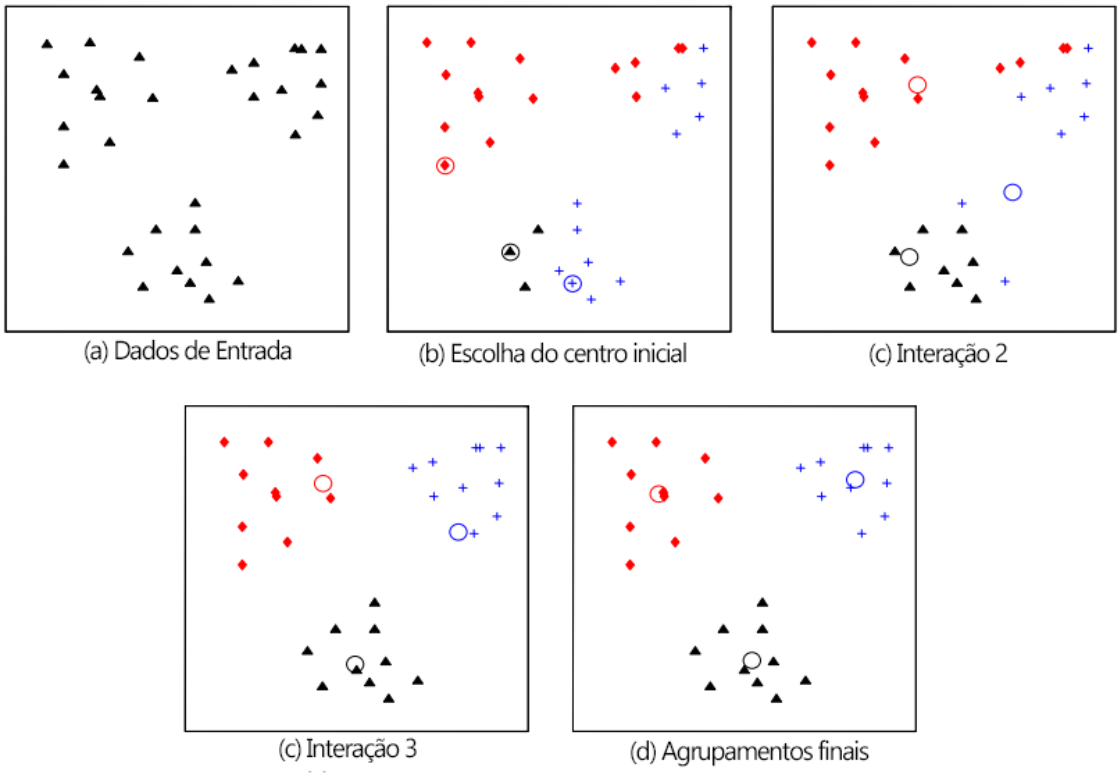
- escolher k centros de *clusters* para coincidir com k itens aleatórios ou k pontos aleatórios dentro do conjunto de dados;
- atribuir cada item ao *cluster* cujo centro esteja mais próximo deste item;
- recalcular os centros usando os membros atuais dos *clusters*;
- caso um critério de convergência não foi alcançado, voltar ao segundo passo. Senão, finalizar o processo.

O algoritmo *k-means* tem como objetivo encontrar agrupamentos em que a soma dos erros quadráticos de cada *cluster* é o mínimo (JAIN, 2010). Este cálculo é dado por:

$$J(C_k) = \sum_{k=1}^{K} \sum_{x_i \epsilon C_k} ||x_i - \mu_k||^2 \tag{3.1}$$

A Figura 4 mostra um exemplo de clusterização por *k-means* em um sistema com duas dimensões utilizando 3 centros. Neste exemplo é possível visualizar como o processo de seleção de novos centros e atribuição de entradas ao *cluster* de cada respectivo centro. Neste exemplo, também, nota-se que os pontos iniciais foram entradas aleatórias no sistema.

Figura 4: Clusterização por *K-means* em um sistema bidimensional com 3 centros



Fonte: Jain (2010, tradução nossa)

4 METODOLOGIA

Neste trabalho foi utilizado o dataset *KDD CUP 99* como base de dados para aprendizado (arquivo de 10% do *dataset* original) e teste (arquivo de teste) (KDD..., 1999). O algoritmo de clusterização para formação de *clusters* utilizado é o *k-means*, este sendo o método escolhido por ser um dos métodos de clusterização mais populares (JAIN, 2010).

4.1 *KDD CUP 99*

O *dataset KDD CUP 99* contém 41 atributos, além da classificação dos status da conexão (normal ou ataque). Esses atributos são mistos entre qualitativos e quantitativos.

Os tipos de ataque abordados neste *dataset* podem ser classificados como: *Denial of Service Attack (DoS)*: ataques relacionados à negação de serviços; *Remote-to-local (R2L)*: ataques relacionados a acessos remotos não autorizados; *User-to-root (U2R)*: ataques relacionados a fornecer privilégios a usuários não autorizados; *Probing*: ataques relacionados a análises de arquitetura de serviços e rede.

O *dataset* contém o rótulo dos dados, ou seja, informa se uma entrada é considerada normal ou considerada um ataque, dando, inclusive, a especificação do ataque. Este dado não será levado em conta durante os processos de treino e teste, sendo somente utilizada após os dois processos para ser possível avaliar a taxa de detecção, bem como a taxa de *false-positives*.

Tabela 1: Classificação dos ataques encontrados no *dataset KDD CUP 99*

Classificação	**Ataques encontrados no *dataset KDD CUP 99***
Denial of Service - DOS	*neptune*, *back*, *smurf*, *pod*, *land* e *teardrop*
User to Root - U2R	*buffer_overflow*, *loadmodule*, *rootkit* e *perl*
Remote to Local - R2L	*warezclient*, *multihop*, *ftp_write*, *imap*, *guess_passwd*, *warezmaster*, *spy* e *phf*
Probbing - PROBE	*portsweep*, *satan*, *nmap* e *ipsweep*

Fonte: elaborada pelo autor.

4.2 REDUÇÃO DE DADOS

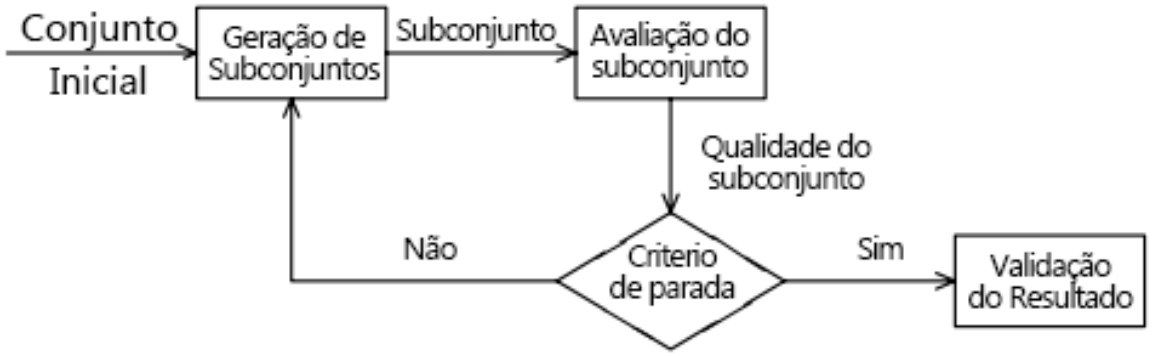
Para que a detecção por anomalia seja feita com sucesso, é necessário que haja um número muito maior de atividades classificadas como normais do que atividades classificadas como ataque. Será usada a mesma métrica usada em Portnoy (2000), em que as atividades consideradas "normais" devem compor cerca de 99% do *dataset*.

Apesar de muito utilizado, o *KDD CUP 99* apresenta muitos dados duplicados, o que pode causar problemas na formação de *clusters*. Removendo os itens duplicados, além de evitar problemas de formação de *clusters* viciados, o número de processos classificados como normais será muito mais relevante (TAVALLAEE et al., 2009).

4.3 SELEÇÃO DE ATRIBUTOS

Seleção de atributos, ou *feature selection*, se refere ao processo de selecionar um subconjunto de atributos de um determinado conjunto inicial através de algumas técnicas de avaliação (LIU; YU, 2005). A Figura 5 mostra os passos gerais para a realização do processo de seleção de atributos.

Figura 5: Fluxo geral para seleção de atributos



Fonte: Liu e Yu (2005, p. 3, tradução nossa)

O processo de selecionar os atributos desejados é importante em casos em que conjuntos de dados extensos e com muitos atributos a serem processados pois estes podem acabar causando relações falsas entre dados, o que pode ocasionar má formação de *clusters* (CHEBROLU; ABRAHAM; THOMAS, 2005).

Ainda, se existe conhecimento sobre os atributos e domínio sobre o problema a ser estudado, e a relevância que cada atributo tem quanto ao domínio do problema, é possível fazer o processo de seleção de atributos utilizando deste conhecimento (GUYON; ELISSEEFF, 2003).

O método utilizado neste trabalho para a seleção de atributos é o método *Variance Threshold*, que tem como objetivo a remoção de todos os atributos com baixa variância. A equação de variância usada para definir qual é a variância mínima para a seleção do atributo, onde p é o valor da probabilidade de sucesso, é dada por:

$$Var(X) = p(1-p) \tag{4.1}$$

A utilização deste método neste trabalho se deve ao fato de ser uma implementação simples de seleção de atributos, dando prioridade a demonstrar a relevância quanto ao tempo de execução dos algoritmos de clusterização antes e após o processo de seleção de atributos, e por não necessitar de domínio prévio sobre os atributos do *dataset*. Em outras palavras, a relação entre número de atributos e o tempo de clusterização. O valor de p será alterado durante os experimentos deste trabalho para ser possível visualizar esta diferença.

4.4 NORMALIZAÇÃO

Segundo Wang et al. (2009, p. 2, tradução nossa), "normalização de dados serve para nivelar os valores de cada atributo contínuo em um limite bem-proporcionado para que o efeito de um atributo não domine sobre os outros".

Caso um atributo tenha um valor máximo muito maior do que o valor máximo de outro atributo, o resultado do aprendizado ficará "viciado" no valor mais alto, fazendo com que os outros atributos tenham sua importância reduzida (WANG et al., 2009).

Para que a normalização aconteça, é necessário que os dados sejam todos quantitativos, ou seja, todos numéricos. Então, os valores qualitativos são transformados em números, e depois normalizados normalmente. Por exemplo, no caso do atributo "protocol_type", será atribuido o valor "1" para "TCP‘", "2" para "UDP" e "3" para "ICMP".

O método de normalização utilizado neste trabalho é o *mean range*, que se dá pela fórmula 4.2, onde v_i é o valor atual, $min(v_i)$ é o valor mínimo do atributo a ser normalizado e $max(v_i)$ é o valor máximo do atributo a ser normalizado. Esta função retornará sempre valores entre 0 e 1. A equação 4.2 representa este método, que foi escolhido para o uso neste trabalho por conta da fácil implementação (WANG et al., 2009):

$$x_i = \frac{v_i - min(v_i)}{max(v_i) - min(v_i)} \tag{4.2}$$

4.5 ROTULAMENTO DE *CLUSTERS*

Neste trabalho, o número de *clusters* a serem gerados pelo método de clusterização será dado por um valor k, que irá variar durante os experimentos.

Sabendo que k é o número de *clusters*, nos experimentos relacionados a rotulamento de *clusters*, será definido um outro valor t, que é definido como o grau de tolerância de rotulamento e pode variar entre 0 (todos os *clusters* são rotulados como normais) e 1 (somente o maior *cluster* será rotulado como normal). Este valor t será usado da seguinte forma: seja n a quantidade de dados do maior *cluster* encontrado, o valor $m = n * t$ será o limite inferior para que outros *clusters* possam ser rotulados como normais. Por exemplo: se o maior *cluster* tiver 1000 entradas e t for definido como 0.8, todos os *clusters* que tiverem pelo menos 800 entradas são rotulados como normais.

Esta ideia é derivada do método usado por Jain, Murty e Flynn (1999). Porém, o parâmetro usado pelo autor era usado como uma porcentagem N dos maiores *clusters* que seriam considerados como normais.

A ideia dos dois métodos é similar - escolher n dos maiores *clusters* como normais - porém, após testes preliminares, o método proposto coube melhor nas expectativas do autor e nas representações que foram pensadas para efeitos de comparação.

4.6 TECNOLOGIAS

Todos os algoritmos usados neste trabalho foram escritos na linguagem de programação *Python 3*, para que fosse possível utilizar as bibliotecas *Pandas*, para utilização

facilitada de *dataframes* (ou matrizes extensas), e *scikit-learn*, para utilização de métodos de seleção de atributos e clusterização - em especial, o método *k-means*.

O computador utilizado nos experimentos possui as especificações mostradas na Tabela 2.

Tabela 2: Especificações do computador usado para realização dos experimentos

Nome do Componente	**Especificação**
Processador	Intel Core i3-7100 @3.90GHz
Memória RAM	8,00 GB
Sistema Operacional	*Windows 10 Pro, 64 bit* - Compilação 17134
Placa de vídeo	NVIDIA GeForce GTX 1050 Ti - 4,00 GB

Fonte: elaborada pelo autor.

5 RESULTADOS

Os experimentos realizados neste trabalho utilizam os métodos descritos no tópico anterior. As equações 5.1, 5.2 e 5.3, que representam a taxa de detecção, a taxa de *false-positives* e a precisão dos resultados, respectivamente, são utilizadas neste capítulo.

$$detecção = \frac{true\text{-}positives}{true\text{-}positives + false\text{-}positives + true\text{-}negatives + false\text{-}negatives} \tag{5.1}$$

$$fp = \frac{false\text{-}positives}{true\text{-}positives + false\text{-}positives + true\text{-}negatives + false\text{-}negatives} \tag{5.2}$$

$$precisão = \frac{true\text{-}positives + true\text{-}negatives}{true\text{-}positives + false\text{-}positives + true\text{-}negatives + false\text{-}negatives} \tag{5.3}$$

5.1 REDUÇÃO DE DADOS

Após a fase de redução de dados, o *dataset* de aprendizado é composto por 145585 entradas, onde a quantidade de ataques e atividades normais é exibida na Tabela 3, juntamente das quantidades de ataques após as etapas de remoção de duplicatas e redução de dados.

O *dataset* de treino, ao passar pelas etapas de remoção de duplicatas e redução de dados, é detalhado pela Tabela 4.

Através destes resultados, pode-se notar a importância destes métodos para a utilização do *dataset KDD CUP 99* como métodos de pré-processamento. O *dataset* possui diversas entradas duplicadas, cujos malefícios foram descritos no capítulo anterior.

5.2 SELEÇÃO DE ATRIBUTOS

A Tabela 5 exibe os atributos selecionados para cada valor de p, onde p é o valor usado para calcular a variância mínima desejada. Quanto menor o valor de p, menos atributos são selecionados para o treinamento e teste.

Os resultados comparativos por grau de seleção de atributos podem ser verificados na Tabela 6, que exibe a taxa de detecção por valor de p variando entre 0,98, 0,96 e 0,94, e na Tabela 7, que exibe os valores de *false-positives* pelos mesmos valores de p.

Na Figura 6 é possível visualizar a diferença no tempo de treinamento por grau de seleção de atributos, variando o valor de p.

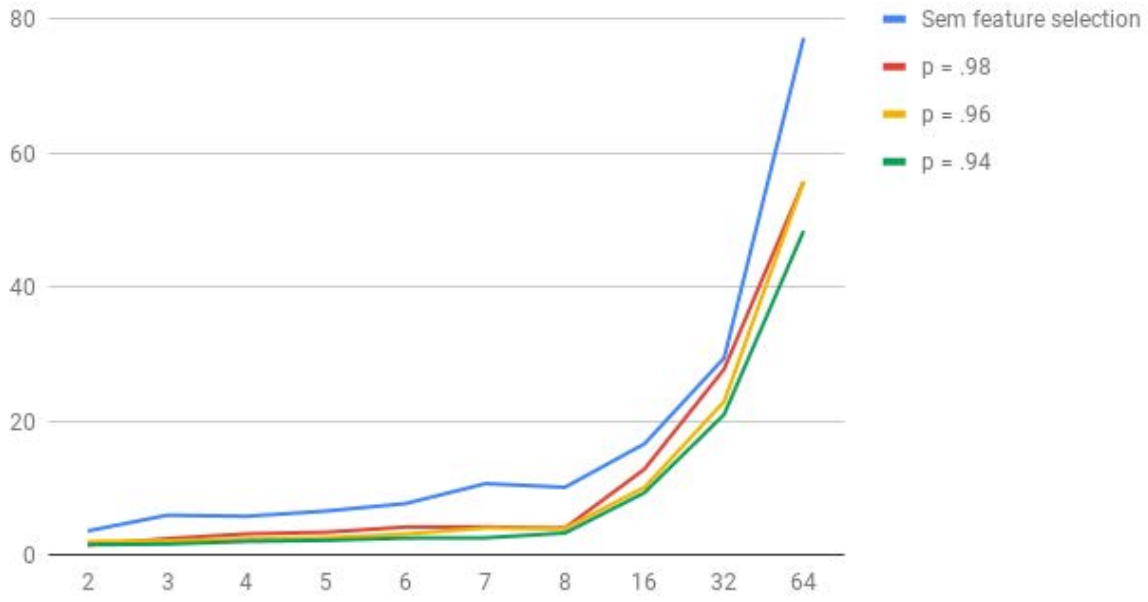
A partir destes dados, é possível verificar que a aplicação de seleção de atributos não teve grande impacto tanto na taxa de detecção quanto na taxa de *false-positives*, mantendo os valores geralmente próximos, variando até 10%, exceto um caso onde a variação foi quase de 40%.

Tabela 3: Quantidade de ataques no *dataset* de treino antes e após redução de dados

Ataque	**10% Original**	**Após remoção de duplicatas**	**Após redução**
normal	97277	87831	87831
neptune	107201	51820	789
back	2203	968	15
teardrop	979	918	14
satan	1589	906	14
warezclient	1020	893	14
smurf	280790	641	10
ipsweep	1247	651	10
portsweep	1040	416	7
pod	264	206	4
nmap	231	158	3
spy	2	2	1
perl	3	3	1
warezmaster	20	20	1
phf	4	4	1
buffer_overflow	30	30	1
loadmodule	9	9	1
guess_passwd	53	53	1
imap	12	12	1
land	21	19	1
ftp_write	8	8	1
rootkit	10	10	1
multihop	7	7	1

Fonte: elaborada pelo autor.

Figura 6: Tempo de treinamento por grau de seleção de atributos por valor de p



Fonte: elaborada pelo autor.

Tabela 4: Quantidade de ataques no *dataset* de teste antes e após redução de dados

Ataque	**10% Original**	**Após remoção de duplicatas**	**Após redução**
normal	60592	47913	47913
smurf	164091	936	16
neptune	58001	20332	332
snmpgetattack	7741	179	3
mailbomb	5000	308	6
guess_passwd	4367	1302	22
snmpguess	2406	359	6
satan	1633	860	15
warezmaster	1602	1002	17
back	1098	386	7
mscan	1053	1049	18
apache2	794	794	13
processtable	759	744	13
saint	736	364	6
portsweep	354	174	3
ipsweep	306	155	3
httptunnel	158	145	3
pod	87	45	1
nmap	84	80	2
buffer_overflow	22	22	1
multihop	18	18	1
sendmail	17	15	1
named	17	17	1
ps	16	16	1
rootkit	13	13	1
xterm	13	13	1
teardrop	12	12	1
xlock	9	9	1
land	9	9	1
xsnoop	4	4	1
ftp_write	3	3	1
phf	2	2	1
loadmodule	2	2	1
perl	2	2	1
sqlattack	2	2	1
worm	2	2	1
udpstorm	2	2	1
imap	1	1	1

Fonte: elaborada pelo autor.

Na Figura 7, é possível avaliar a precisão dos resultados obtidos em cada experimento, por valor de p.

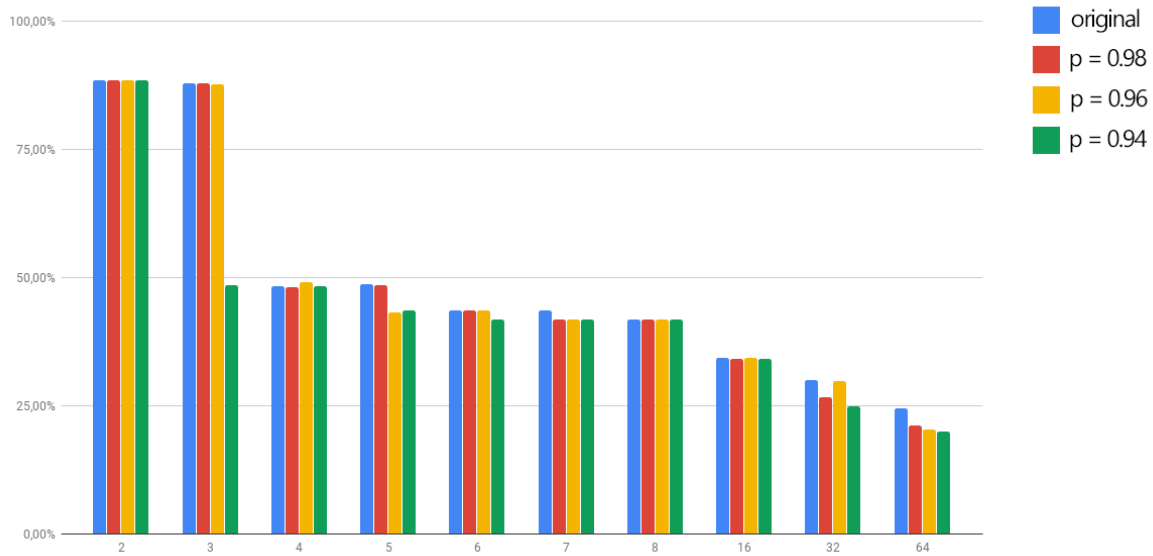
Pode-se notar que as taxas de detecção encontadas com $p = 0.98$ são iguais ou melhores que as taxas de detecção originais, enquanto em $p = 0.96$ obteve resultados

Tabela 5: Atributos selecionados por valor de p

p	**Atributos selecionados**
0.98	*protocol_type*, *logged_in*, *rerror_rate*, *srv_rerror_rate*, *srv_diff_host_rate*, *dst_host_count*, *dst_host_srv_count*, *dst_host_same_srv_rate*, *dst_host_diff_srv_rate*, *dst_host_same_src_port_rate*, *dst_host_rerror_rate* e *dst_host_srv_rerror_rate*
0.96	*logged_in*, *rerror_rate*, *srv_rerror_rate*, *srv_diff_host_rate*, *dst_host_count*, *dst_host_srv_count*, *dst_host_same_srv_rate*, *dst_host_same_src_port_rate*, *dst_host_rerror_rate* e *dst_host_srv_rerror_rate*
0.94	*logged_in*, *srv_diff_host_rate*, *dst_host_count*, *dst_host_srv_count*, *dst_host_same_srv_rate* e *dst_host_same_src_port_rate*

Fonte: elaborada pelo autor.

Figura 7: Precisão dos resultados dos experimentos de seleção de atributos por valor de p



Fonte: elaborada pelo autor.

Tabela 6: Taxa de detecção por grau de seleção de atributos por valor de p

Número de *clusters*	$p = 1$ **(original)**	$p = 0.98$	$p = 0.96$	$p = 0.94$
2	89,11%	89,11%	88,12%	86,93%
3	91,49%	91,49%	92,28%	99,01%
4	98,61%	98,61%	98,81%	98,81%
5	98,61%	98,61%	90,30%	100%
6	100%	100%	100%	100%
7	100%	100%	100%	100%
8	100%	100%	100%	100%
16	94,46%	94,26%	94,65%	94,46%
32	95,45%	95,45%	96,83%	95,05%
64	97,43%	97,03%	97,03%	97, 03%

Fonte: elaborada pelo autor.

Tabela 7: Taxa de *false-positives* por grau de seleção de atributos por valor de p

Número de *clusters*	$p = 1$ **(original)**	$p = 0.98$	$p = 0.96$	$p = 0.94$
2	11,58%	11,58%	11,47%	11,43%
3	12,08%	12,05%	12,33%	52,03%
4	52,30%	52,44%	51,37%	52,15%
5	51,93%	51,98%	57,23%	57,01%
6	56,98%	57,01%	57,02%	57,01%
7	56,99%	58,81%	58,80%	58,80%
8	58,79%	58,79%	58,82%	58,81%
16	66,24%	66,45%	66,38%	66,61%
32	70,69%	66,45%	70,83%	75,80%
64	76,33%	79,61%	80,49%	80,49%

Fonte: elaborada pelo autor.

melhores do que 0.96, com exceção dos resultados para 2 e 64 *clusters*, e $p = 0.94$ obteve resultados piores para números elevados de *clusters*, porém aumentou a taxa de detecção em relação aos outros valores (com exceção dos resultados para 2 *clusters*.

As taxas de *false-positives*, por sua vez, apresentaram uma piora (aumento) geral dos valores. Nota-se que onde $p = 0.94$, os valores para 3 *clusters* sofreu a maior piora registrada.

A aplicação de seleção de atributos ainda diminui consideravelmente o tempo necessário para aprendizado, conforme mostrado na Figura 6. Estes dados mostram que quanto maior o número de atributos, maior o tempo para a formação dos *clusters*. Nota-se, também, a grande quantidade de *false-positives* conforme o número de *clusters* aumenta. Isto é causado pela falta de rotulamento de *clusters* considerados "normais", o que será discutido na próxima seção.

5.3 ROTULAMENTO DE *CLUSTERS*

Na Tabela 8 e na Tabela 9 são exibidos os resultados obtidos pela variação de *clusters* considerados “normais” pela aplicação. Os resultados para 2 e 3 *clusters* foram ocultados porque não houve alteração na quantidade de *clusters* considerados “normais".

Tabela 8: Taxa de detecção por grau de tolerância de rotulamento de *clusters* por valor de t

Número de clusters	$t = 1$ **(original)**	$t = 0.1$	$t = 0.2$	$t = 0.4$
4	98,61%	87,33%	87,33%	87,33%
5	98,61%	58,81%	58,81%	90,50%
6	100,00%	59,01%	59,01%	87,52%
7	100,00%	73,66%	73,66%	87,52%
8	100,00%	73,66%	73,66%	94,46%
16	94,46%	37,43%	90,10%	94,46%
32	95,45%	86,93%	95,45%	95,45%
64	97,43%	95,25%	97,03%	97,43%

Fonte: elaborada pelo autor.

A partir dos dados da Tabela 8, é possível notar que os valores onde $t = 0.4$ possuem a melhor taxa de detecção de anomalias comparado com os outros valores, com exceção dos valores obtidos sem os métodos de rotulamento de *clusters*.Os resultados obtidos por $t = 0.2$ e $t = 0.1$ possuem desempenho consideravelmente pior comparados aos outros valores.

Os resultados de taxa de detecção obtidos por $t = 0.4$, inclusive, mantém o mesmo desempenho original obtido sem o rotulamento de *clusters* adicionais (onde apenas o maior *cluster* é considerado o *cluster* "normal") a partir de 16 *clusters*.

Nota-se também, que quanto maior o número de *clusters*, melhor é o desempenho comparado aos resultados de taxas de detecção com o mesmo grau de tolerância t com número menor de *clusters*.

Tabela 9: Taxa de *false-positives* por grau de tolerância de rotulamento de *clusters* por valor de t

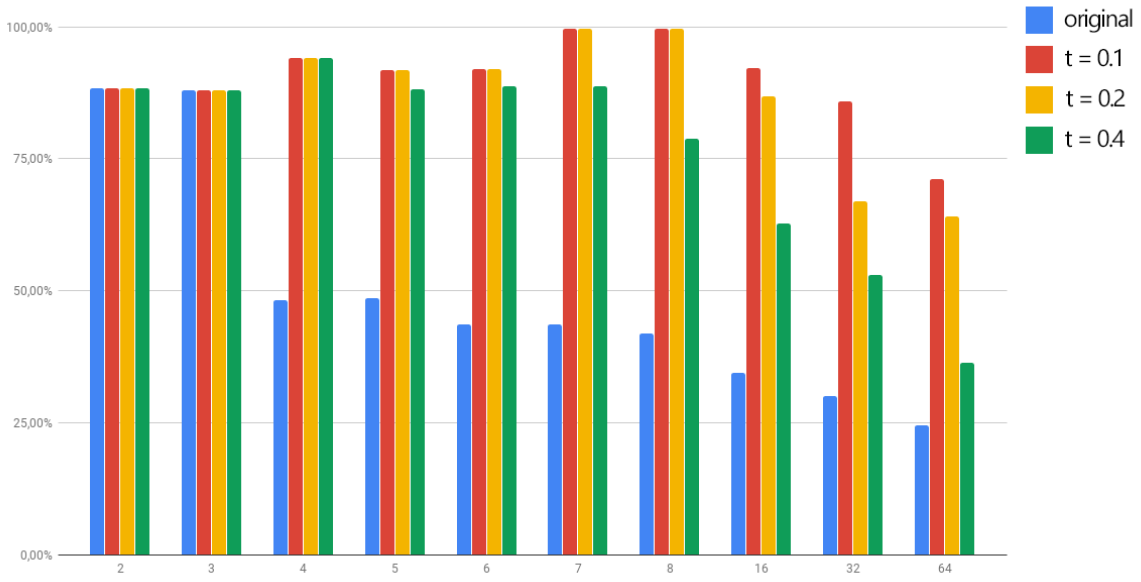
Número de *clusters*	$t = 1$ **(original)**	$t = 0.1$	$t = 0.2$	$t = 0.4$
4	52,30%	5,82%	5,82%	5,82%
5	51,93%	7,80%	7,80%	11,95%
6	56,98%	7,69%	7,69%	11,30%
7	56,99%	0,28%	0,18%	11,28%
8	58,79%	0,28%	0,28%	21,31%
16	66,24%	7,26%	13,17%	37,63%
32	70,69%	14,13%	33,30%	47,45%
64	76,33%	29,17%	36,26%	64,30%

Fonte: elaborada pelo autor.

A partir dos resultados obtidos pela Tabela 9, nota-se que a taxa de *false-positives* diminui consideravelmente em todas as situações, principalmente em números menores de *clusters* analisados. Em especial, nota-se os resultados onde $t = 0.1$ e $t = 0.2$ com 7 e 8 *clusters*, onde a taxa de *false-positives* caiu para menos de 1%.

A Figura 8 mostra a precisão dos experimentos de rotulamento de *clusters* por valor de t.

Figura 8: Precisão dos resultados dos experimentos de rotulamento de *clusters* por valor de t



Fonte: elaborada pelo autor.

A partir dos resultados desta seção, é possível notar a relação de que quanto maior o número de *clusters*, maior é a precisão com as técnicas de rotulamento, em especial com valores menores para t.

6 CONCLUSÃO

Este trabalho teve como objetivo realizar experimentos de detecção de anomalias usando o como base o *dataset KDD CUP 99* e mostrar resultados empíricos quanto a técnicas de *feature selection* e *detecção de anomalias* utilizando a linguagem *Python*.

Através dos resultados obtidos, é possível concluir que, embora não hajam resultados ótimos ou uma fórmula para se calcular os parâmetros utilizados, a manipulação do *dataset* com as técnicas apresentadas pode gerar resultados melhores ou similares com menor tempo de execução.

Os resultados gerados e suas análises satisfazem os objetivos iniciais, uma vez que estes resultados e análises contribuem para o meio científico no estudo de clusterização aplicada a detecção de anomalias em rede, uma vez que as técnicas aplicadas demonstram a importância da aplicação de tais técnicas. Ainda, abre-se caminho para futuros estudos quanto ao rotulamento de *clusters*.

Referências

ALLEN, J.; CHRISTIE, A.; FITHEN, W.; MCHUGH, J.; PICKEL, J. *State of the practice of intrusion detection technologies*. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2000.

AXELSSON, S. *Intrusion detection systems: A survey and taxonomy*. Technical report, 2000.

BACE, R.; MELL, P. *NIST special publication on intrusion detection systems*. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA, 2001.

BASS, T. Intrusion detection systems and multisensor data fusion: Creating cyberspace situational awareness. *Communications of the ACM*, April, v. 43, n. 4, p. 99–105, 2000.

BOLZONI, D. *Revisiting Anomaly-based Network Intrusion Detection Systems*. University of Twente, Enschede, Netherlands, 2009.

CANSIAN, A. M. *Desenvolvimento de um sistema adaptativo de detecção de intrusos em redes de computadores*. Tese (Doutorado) — Universidade de São Paulo, 1997.

CHEBROLU, S.; ABRAHAM, A.; THOMAS, J. P. Feature deduction and ensemble design of intrusion detection systems. *Computers & security*, Elsevier, v. 24, n. 4, p. 295–307, 2005.

DEBAR, H. An introduction to intrusion-detection systems. *Proceedings of Connect*, v. 2000, 2000.

DEBAR, H.; DACIER, M.; WESPI, A. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, Elsevier, v. 31, n. 8, p. 805–822, 1999.

GHOSH, A. K.; WANKEN, J.; CHARRON, F. Detecting anomalous and unknown intrusions against programs. In: IEEE. *Proceedings 14th Annual Computer Security Applications Conference (Cat. No. 98EX217)*. Sterling, VA, USA, p. 259–267, 1998.

GUYON, I.; ELISSEEFF, A. An introduction to variable and feature selection. *Journal of machine learning research*, v. 3, n. Mar, p. 1157–1182, 2003.

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, Elsevier, v. 31, n. 8, p. 651–666, 2010.

JAIN, A. K.; DUBES, R. C. *Algorithms for clustering data*. Englewood Cliffs: Prentice hall, v. 6, 1988.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: a review. *ACM computing surveys (CSUR)*, Acm, v. 31, n. 3, p. 264–323, 1999.

KAUFMAN, L.; ROUSSEEUW, P. J. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, v. 344, 2009.

KDD CUP 99 Intrusion Detection - 10_percent_data set. 1999. Disponível em: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>. Acesso em: 31 de maio de 2019.

KEMMERER, R. A.; VIGNA, G. Intrusion detection: a brief history and overview. *Computer*, IEEE, v. 35, n. 4, p. supl27–supl30, 2002.

KENDALL, K. K. R. *A database of computer attacks for the evaluation of intrusion detection systems*. Dissertação (Mestrado) — Massachusetts Institute of Technology, 1999.

KUMAR, S. *Classification and detection of computer intrusions*. Tese (PhD) — Purdue University, 1995.

LIU, H.; YU, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge & Data Engineering*, IEEE, n. 4, p. 491–502, 2005.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: UNIVERSITY OF CALIFORNIA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Oakland, CA, USA, v. 1, n. 14, p. 281–297, 1967.

MCHUGH, J.; CHRISTIE, A.; ALLEN, J. Defending yourself: The role of intrusion detection systems. *IEEE software*, IEEE, v. 17, n. 5, p. 42–51, 2000.

MUKHERJEE, B.; HEBERLEIN, L. T.; LEVITT, K. N. Network intrusion detection. *IEEE network*, IEEE, v. 8, n. 3, p. 26–41, 1994.

PERLIN, T.; NUNES, R. C.; KOZAKEVICIUS, A. de J. Detecção de anomalias em redes de computadores e o uso de wavelets. *Revista Brasileira de Computação Aplicada*, v. 3, n. 1, p. 2–15, 2011.

PORTNOY, L. *Intrusion detection with unlabeled data using clustering*. Tese (Doutorado) — Columbia University, 2000.

SOUSA, E. P. d. *Estudo sobre sistema de detecção de intrusão por anomalias: uma abordagem utilizando redes neurais*. Dissertação (Mestrado) — Universidade Salvador, 2008.

TAVALLAEE, M.; BAGHERI, E.; LU, W.; GHORBANI, A. A. *A detailed analysis of the KDD CUP 99 data set*. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, p. 1–6, 2009.

VESANTO, J.; ALHONIEMI, E. Clustering of the self-organizing map. *IEEE Transactions on neural networks*, Citeseer, v. 11, n. 3, p. 586–600, 2000.

WAGSTAFF, K.; CARDIE, C.; ROGERS, S.; SCHRÖDL, S. Constrained k-means clustering with background knowledge. In: ICML. *Proceedings of the Eighteenth International Conference on Machine*. Williamstown, MA, USA, p. 577-584, 2001.

WANG, W.; ZHANG, X.; GOMBAULT, S.; KNAPSKOG, S. J. *Attribute normalization in network intrusion detection*. 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks, IEEE, p. 448-453, 2009.

WU, S. X.; BANZHAF, W. The use of computational intelligence in intrusion detection systems: A review. *Applied soft computing*, Elsevier, v. 10, n. 1, p. 1–35, 2010.

ZHANG, Y.; LEE, W. Intrusion detection in wireless ad-hoc networks. In: ACM. *Proceedings of the 6th annual international conference on Mobile computing and networking*. Malibu, CA, USA, p. 275–283, 2000.