Detecção de Intrusões em Redes de Computadores com Base nos Algoritmos KNN, K-Means++ e J48

Mauricio Mendes Faria

Dezembro/ 2016

Dissertação de Mestrado em Ciência da Computação

Detecção de Intrusões em Redes de Computadores com Base nos Algoritmos KNN, K-Means++ e J48

Esse documento corresponde à dissertação apresentada à Banca Examinadora no curso de Mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista.

Campo Limpo Paulista, 16 de dezembro de 2016.

Mauricio Mendes Faria

Profa. Dra. Ana Maria Monteiro (Orientadora)

Faculdade Campo Limpo Paulista

Programa de Mestrado em Ciência da Computação

"Detecção de Intrusões em Redes de Computadores com Base nos Algoritmos KNN, K-Means++ e J48"

Maurício Mendes Faria

Dissertação de Mestrado apresentado ao Programa de Mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Membros da Banca:

Profa. Dra. Ana Maria Monteiro
(Orientadora - FACCAMP)

Profa. Dra. Maria do Carmo Nicoletti
(FACCAMP)

Prof. Dr. Norton Trevisan Roman
(USP)

Campo Limpo Paulista, 16 de dezembro de 2016.

FICHA CATALOGRÁFICA

Dados Internacionais de Catalogação na Publicação (CIP)

Câmara Brasileira do Livro, São Paulo, Brasil.

Faria, Mauricio Mendes

Detecção de intrusões em redes de computadores com base nos algoritmos KNN, K-Means++ e J48 / Mauricio Mendes Faria. Campo Limpo Paulista, SP: FACCAMP, 2016.

Orientadora: Profª. Drª. Ana Maria Monteiro

Dissertação (Programa de Mestrado em Ciência da Computação) – Faculdade Campo Limpo Paulista – FACCAMP.

1. Detecção. 2. Intrusão. 3. KNN. 4. K-Means++. 5. J48. I. Monteiro, Ana Maria. II. Campo Limpo Paulista. III. Título.

CDD-005.1

Resumo. *Hoje em dia muitas aplicações de grandes organizações geram peta bytes de dados que são armazenados na intenção de serem processados no futuro para produzirem informações uteis para as tomadas de decisões. Dados provenientes de logs de aplicações também são gerados em ordem exponencial e dessa massa de dados é possível extrair conhecimento para detectar se as aplicações estão passando por alguma instabilidade por conta de usuários mal-intencionados. Para detectar acessos mal-intencionados em um sistema, usa-se uma ferramenta chamada de IDS (Sistema de Detecção de Intrusão) que pode utilizar diferentes técnicas para classificar uma conexão de rede como intrusão ou normal. Nesse trabalho, são analisados algoritmos de mineração de dados que possam ser integrados em um IDS para detectar instruções. Experimentos foram realizados utilizando o ambiente WEKA, o conjunto de dados NSL-KDD e os algoritmos supervisionado KNN (K Nearest Neighbor) e J48, e o algoritmo não supervisionado, o K-Means++ com o intuito de avaliar as possibilidades.*

Abstract: *Nowadays many applications in large organizations generate petabytes of data that are stored with the intention of be processed in the future to produce useful information for decision making. Data from applications logs are also generated in exponential order and from this mass of data it is possible to extract knowledge to detect if the applications are experiencing some instability due to malicious users. Tools called IDS (Intrusion Detection System) are used to detect a malicious access. An IDS can make use of different techniques to classify a network connection as intrusion or normal. In this work, data mining algorithms that can be integrated into an IDS to detect intrusions are analysed. An experiment was conducted using the WEKA environment, the NSL-KDD dataset and the supervised algorithms KNN (K Nearest Neighbor) and J48, and the unsupervised algorithm K-means ++ in order to assess them.*

Agradecimentos

À Deus por permitir que essa caminhada seja findada com sucesso.

À minha orientadora professora Dra. Ana Maria Monteiro que foi amiga, irmã, professora e motivadora de todo o processo de desenvolvimento desse trabalho.

À Dra. Maria do Carmo Nicoletti pela motivação e aconselhamento sobre a importância e seriedade do conhecimento e ao Dr. Osvaldo Luiz de Oliveira por permitir e motivar a pesquisa desse trabalho, além do apoio acadêmico.

À minha mãe Ana Mendes Faria (*in memoriam*), que me apoiou no início dessa caminhada, e meu pai Geraldo Nogueira Faria (*in memoriam*), que foi exemplo de espírito investigativo, questionando e buscando respostas.

Aos meus sobrinhos Edson Faria Lobo e Luiz Henrique Faria Lobo e a minha irmã Maria de Lourdes Mendes Faria por terem paciência diante de minha ausência no período de pesquisa.

Aos amigos pelo auxílio nos estudos, conversas, companheirismo, convivência e amizade durante todo o período em que estive dedicado à realização deste trabalho. Aos professores e funcionários do programa de mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista.

Sumário

Glossário

ACM	*Association for Computing Machinery*
ARFF	*Attribute Relation File Format*
DARPA	*Defense Advanced Research Projects Agency*
DA	Detecção Anomalia
DoS	*Denial of Service*
DTA	Detecção de Tipo de Anomalia
FN	Falso Negativo
FP	Falso Positivo
GNU	*General Public License*
HIDS	*Host Based Intrusion Detection System*
HTTP	*Hipertext Transfer Protocol*
IDS	*Intrusion Detection System*
IP	*Internet Protocol*
IPS	*Intrusion Prevention System*
K-Means	K-Médias
KNN	*K - Nearest Neighbor*
MIT	*Massachusets Institute of Technology*
NIDS	*Network Based Intrusion Detection System*
NN	*Nearest Neighbor*
OLAP	*Online Analitcs Process*
R2L	*Remote to Local*
SOM	*Self Organization Feature Map*

SSD	*Solid State Drive*
SVM	*Support Vector Machine*
$T_{ACURÁCIA1}$	Taxa de Acurácia para o experimento DA
$T_{ACURÁCIA2}$	Taxa de Acurácia para o experimento DTA
TCP	*Transfer Control Protocol*
TE	Taxa de Erro
$TE_{ANOMALIA}$	Taxa de Erro Anomalia
TE_{DOS}	Taxa de Erro por falso positivo para a classe "dos"
TE_{NORMAL}	Taxa de Erro Normal
$TE_{PROBING}$	Taxa de Erro por falso positivo para a classe “probing"
TE_{R2L}	Taxa de Erro por falso positivo para a classe "r2l"
TE_{TOTAL}	Taxa Total de Erro
TE_{U2R}	Taxa de Erro por falso positivo para a classe "u2r"
U2R	*User to Root*
UDP	*User Datagram Protocol*
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Atualmente enormes massas de dados são geradas através de diversificados tipos de aplicações, tais como as bancárias, médicas, tecnológicas, ambientais e o comércio eletrônico, disponíveis em diversos tipos de arquiteturas como: web, mobile ou *desktops*. Isso tudo não seria um problema se não tivéssemos as aplicações acessadas de formas diversas, por diferentes usuários em qualquer parte do mundo, através da Internet.

Ao longo dos anos as tecnologias de redes tornaram-se muito difundidas nas organizações, ambientes domésticos, shoppings, aeroportos e restaurantes, por serem de grande utilidade no que tange ao rompimento de barreiras, à diminuição das distâncias, além da ampliação do conhecimento.

A popularização dos acessos compartilhados à Internet contribuiu, cada vez mais, para o aumento exponencial dos dados armazenados pelas organizações. Um simples acesso a um *site* de Internet, a uma rede social ou então um clique em uma fotografia de um produto, pode gerar vários registros em um banco de dados. Contudo essa geração de dados acarreta às organizações, que hospedam as aplicações, muitas preocupações no âmbito da segurança dessa informação. De maneira geral as redes de computadores e os sistemas informatizados tornaram-se ferramentas indispensáveis no uso dos sistemas de informação e grande parte das informações gerenciadas por eles, em algum grau, são confidenciais e sua proteção é necessária.

A palavra intrusão tornou-se popular devido ao seu uso para evidenciar a fragilidade da proteção dos dados privados dos usuários que fazem uso de sistemas informatizados. O roubo de informações passou a ser o assunto em voga nas mídias e no âmbito científico. A intrusão nos termos da tecnologia da informação acontece quando, por exemplo, um usuário tenta acessar informações que ele não está autorizado a acessar, a ele atribui-se o nome de intruso e o processo é chamado de intrusão. Uma intrusão pode acontecer por um usuário externo ou interno. Um ataque por usuário externo, acontece quando esse não tem acesso ao sistema, nem mesmo uma conta de usuário básica e utiliza

técnicas para acessar o ambiente e criar o usuário, já um ataque de um usuário interno acontece quando esse já possui uma conta com permissões básicas e se aproveita das fragilidades do ambiente para atribuir permissões que o façam ter um perfil de um usuário verdadeiro, com acesso a outras áreas do ambiente de rede.

Diante deste cenário questionamentos são feitos, de forma recorrente, sobre o ambiente de armazenamento dos dados, ou como são estabelecidas as estratégias de armazenamento e acesso a esses dados:

Como manter os dados seguros?

Quem está acessando os dados?

Que usuários deveriam acessar determinados dados?

Existe uma estratégia alternativa para a prevenção de uma intrusão ao sistema?

E o questionamento principal: Qual a melhor forma de detectar um intruso no ambiente computacional de uma organização?

Para responder a estas perguntas um grande esforço está sendo feito para o desenvolvimento de sistemas de detecção de intrusão (*Intrusion Detection System*-IDS), capazes de ajudar na detecção das tentativas de acesso a sistemas por pessoas não autorizadas ou, até mesmo, por dispositivos automáticos, para obter acesso às redes de computadores e às informações neles armazenadas.

Por isso, é preciso entender que um IDS é um conjunto de softwares e dispositivos de hardware que monitoram uma rede ou as atividades de um sistema, para detectar atividades maliciosas ou violações de políticas de segurança, produzindo relatórios para uma estação de gerenciamento (Singh & Bansal, 2013).

Alguns sistemas podem tratar de impedir tentativas de intrusões, mas isso não é esperado de um sistema de monitoramento (IDS). Já os IPS (*Intrusion Prevention System*) são direcionados para as ações preventivas com base nos dados da intrusão detectada pelo IDS. O objetivo de um IPS é impedir que a tentativa de intrusão detectada seja consolidada e, nesse aspecto, um IPS tem total dependência dos alertas gerados pelo IDS (Singh & Bansal, 2013).

O desenvolvimento de um IDS é motivado pelas condições relativamente precárias com relação à segurança lógica dos sistemas informatizados, existentes hoje que são suscetíveis a invasões. É quase impossível ter um sistema totalmente seguro, mas mesmo os sistemas mais seguros são vulneráveis a ataques internos. Novos tipos de invasões surgem continuamente e são necessárias novas técnicas ou adaptações das já existentes para impedir a progressão desses ataques (Singh & Bansal, 2013).

Os IDS são introduzidos para detectar possíveis violações das políticas de segurança das atividades de um sistema. Também são chamados de segunda linha de defesa, uma vez que entram em cena depois da ocorrência de uma intrusão ou de uma tentativa de intrusão. Se for detectada uma tentativa de intrusão, uma ação pode ser iniciada para evitar ou minimizar os danos no sistema. As informações provenientes dos IDS podem ajudar a melhorar as técnicas de prevenção, fornecendo informações detalhadas sobre as intrusões.

1.1. Objetivos e Métodos

O objetivo principal deste trabalho foi investigar a viabilidade da aplicação de técnicas de mineração de dados para detecção de intrusões em redes de computadores. Para abordar o assunto de forma prática e experimental foi elaborada uma série de experimentos utilizando os algoritmos KNN, J48 e K-Means++, para desempenhar o papel de sensores em um sistema de detecção de intrusão.

Para conduzir essa investigação foram realizados dois experimentos. Um deles teve por objetivo classificar instâncias de acessos de redes de computadores como “normal” ou “anomalia”, registrando qual algoritmo conseguiu o melhor desempenho, no que tange a qualidade de classificação, e o outro experimento a classificação por categorias de ataques (Normal, DoS, R2L, U2R e *Probing*), descritos na Seção 4.1.

A abrangência deste trabalho contempla desde o estudo da área segurança de redes, a obtenção de dados reais e a preparação destes dados até a escolha e configuração dos algoritmos disponíveis no ambiente WEKA, para extração de conhecimento relevante para a tomada de decisão do auditor encarregado da segurança da rede. O conhecimento obtido também poderia vir a alimentar outros sistemas automatizados responsáveis pela prevenção da intrusão.

Não foi escopo deste trabalho criar, produzir ou alterar qualquer ferramenta comercial de detecção de intrusão. O trabalho focalizou, essencialmente, os testes e experimentações em uma hipotética etapa de sensoriamento de um sistema de detecção de intrusão experimental.

1.1.1. Objetivos Específicos

Para alcançar o objetivo geral, surgiram os seguintes desafios:

- Conhecer a área de segurança de rede e os tipos de intrusões que podem ocorrer em uma conexão de rede;
- Definir um estudo de caso baseado em dados reais sobre intrusões de redes de computadores;
- Descobrir conhecimento relevante para os usuários de sistemas de auditoria de segurança de rede;
- Gerar novos conjuntos de dados, separados por categorias de ataques, derivados de um dos conjuntos de dados utilizados pela comunidade de mineração de dados aplicada na detecção de intrusões, o NSL-KDD (Tavallaee *et al.*, 2009), para melhor elaboração de experimentos;
- Registrar um caso de uso utilizando o ambiente WEKA para descoberta de conhecimento sobre detecção de intrusão e disponibilizá-lo para a comunidade de segurança de rede.

Portanto, este estudo pretende através de um conjunto de experimentos, fornecer informações compreensíveis e relevantes à tomada de decisão para um auditor de segurança de rede ou para um sistema de prevenção de intrusão.

1.2. Trabalhos relacionados

Foram estudados trabalhos relacionados com detecção de intrusões e com o uso de mineração de dados para detectar essas intrusões. Alguns artigos destacaram-se pela aderência ao assunto e, a seguir, é apresentado um breve relato desses artigos:

O trabalho de Vaccaro & Liepins (1989) mostra o inicio dos estudos sobre detecção de intrusão em redes de computadores em um período em que não existia a Internet como

se conhece hoje. O artigo merece destaque porque consta como primeiro relato científico sobre detecção de anomalia em redes de computadores. Os autores discutem brevemente o *Winsdom and Sense* (W & S), um sistema de detecção de anomalias atrelado à segurança de computadores, desenvolvido no Laboratório Nacional de Los Alamos (LANL)[1]. O W & S é baseado em estatística.

Como pontos positivos é importante salientar que, pela época da publicação do artigo, a criação de um sistema de detecção de intrusão baseado em regras e árvores de decisão era considerada uma inovação. Levando em conta a pouca evolução dos sistemas operacionais que não permitiam capturar com facilidade dados de auditoria, o trabalho foi um dos mais importantes e promissores da época.

A grande contribuição do artigo diz respeito aos critérios de *design* de um IDS, que são:

- Reduzir os dados brutos de auditoria para formas mais adequadas para a detecção de intrusões;
- Construir, de forma automática, uma base de regras sem interferência humana (auto aprendizado de regras);
- Tolerância a conflitos de regras;
- Lidar com o conhecimento incerto e errôneo;
- Continuar o aprendizado de experiências, e se adaptar às condições transientes;
- Aceitar modificações humanas para a base de regras, mas não ser excessivamente dependente da *expertise* humana;
- Tomar decisões em tempo real em função de comportamentos classificados como anômalos;
- Prover uma interface amigável para exibir as anomalias e funcionalidades para realizar ações corretivas e preventivas de forma interativa;
- Criar uma mínima interferência com o funcionamento real do sistema;
- Ser portável para diferentes aplicações, sistemas operacionais e hardwares.

Como pontos negativos da pesquisa vale destacar que nos critérios de *design* foram estabelecidas algumas metas que deveriam ser atingidas, mas que não foram totalmente

[1] http://www.lanl.gov/

implementadas, por exemplo, a capacidade de determinar atividade anômala dos computadores e determinar se as anomalias são significativas. Segundo os autores, na época era preciso mais experiência em ambientes operacionais e com invasões simuladas antes de projetar ferramentas de análise adicionais para essa finalidade e ajustar corretamente essa detecção. Um problema difícil de ser tratado era que os sistemas operacionais da época não conseguiam capturar dados gerais corretos para análise.

Por ser um artigo antigo, o detalhamento do desenvolvimento dos recursos não está em um padrão que possa ser aproveitado nos tempos atuais, porém muitos ensinamentos sobre os critérios de *design* continuam vigentes hoje em dia.

A pesquisa de Lee & Stolfo (1998) discute métodos gerais de detecção de intrusão em redes de computadores, principalmente os baseados em mineração de dados para classificação de intrusões. Mais especificamente, o artigo trata de um estudo comparativo de dois métodos de detecção de intrusão: algoritmo de regras de associação e algoritmo de episódios frequentes, ambos supervisionados. Porém, o destaque do artigo é o desenvolvimento e teste dos métodos de detecção de intrusão, através da implementação dos algoritmos.

Os autores também propõem uma arquitetura de agentes para os sistemas de detecção de intrusão, onde mecanismos de aprendizagem contínua calculam e fornecem para os agentes novos modelos de detecção atualizados.

Como ponto positivo destaca-se a produção de um quadro sistêmico sobre técnicas de mineração de dados para detecção de intrusão. Este quadro é composto por classificadores, regras de associação e programas de episódios frequentes que podem ser usados para construir (automaticamente) modelos de detecção. Porém, não é feito qualquer detalhamento da ferramenta utilizada para apoio à aplicação dos algoritmos.

Por sua vez, Jones & Sielken (2000) apresentam um estudo comparativo entre algoritmos que utilizam abordagens baseadas em desvios e anomalias, porém, o âmbito do estudo está mais relacionado com as abordagens do que com os algoritmos propriamente ditos. Na detecção por anomalia é destacada a função do NIDS *(Network Intruson Detection System)*, que faz o sensoriamento de pacotes e gera uma resposta imediata à análise desses pacotes. Nesse trabalho não é feito qualquer apontamento sobre ferramentas de análise de dados, porém tem um detalhamento profundo do funcionamento

do NIDS, bem como das abordagens relatadas. Como pontos positivos pode-se incluir os apontamentos históricos sobre outras pesquisas voltadas à detecção de intrusão estática e dinâmica, seus funcionamentos e fragilidades.

O trabalho de Tavallaee *et al*. (2009) propõe um novo conjunto de dados para treinamento e teste baseados no conhecido conjunto KDDCup99. Os autores analisaram o conjunto KDDCup99 e acharam uma série de inconsistências e redundâncias nos registros desse conjunto, que ocasionava a falsa impressão de boa acurácia nos algoritmos de mineração que utilizavam esses dados. Em função da análise realizada, os autores fizeram a proposta de um outro conjunto de dados, o NSL-KDD, mais preciso e adequado. Esse conjunto de dados foi o escolhido para os experimentos realizados nesta dissertação, visto que na literatura não foram achados outros trabalhos que abordassem tão amplamente o problema do conjunto de dados KDDCup99. Apesar de todo o trabalho dos autores, realizado com o apoio da ferramenta WEKA, não foi encontrado o detalhamento dos experimentos por eles realizados para reprodução ou estudo.

Singh & Bansal (2013) apresentam em seu artigo técnicas utilizadas em IDS que fazem uso de aprendizado supervisionado e não supervisionado. As técnicas são categorizadas com base em diferentes abordagens como a estatística, a mineração de dados, as redes neurais (RN) e os mapas auto organizáveis. Os algoritmos abordados no estudo são RN, SVM *(Support Vector Machines)*, uma variação do algoritmo K-Means e o SOM (Mapas Auto Organizáveis). A análise desse artigo viabilizou o nosso estudo comparativo de algoritmos, utilizando técnicas de aprendizado supervisionado e não supervisionado, bem como o detalhamento de um IDS. Porém não foi encontrado um detalhamento apropriado dos experimentos que pudesse evidenciar o passo a passo da aplicação dos conceitos.

1.3. Organização e Estrutura do Trabalho

A organização deste trabalho está estruturada da seguinte forma:

- O Capítulo 1 apresenta o cenário atual das aplicações que são usadas por organizações que geram enormes massas de dados, e o conhecimento que pode estar implícito nesses dados e as implicações relacionadas à segurança da informação. É apresentado também o objetivo dos IDS e a sua relação com

detecção de intrusões. Os objetivos gerais e específicos são discutidos nesse capítulo.

- O Capítulo 2 apresenta referencial teórico sobre o processo de KDD (*Knowledge Discovery in Databases*) e suas etapas, até chegar ao conhecimento extraído dos dados, bem como salienta a importância da mineração de dados e das tarefas de classificação e agrupamento, relacionados à detecção de intrusão.
- O Capítulo 3 apresenta referencial teórico sobre os algoritmos KNN, K-Means++ e J48, que neste trabalho foram utilizados na indução de modelos para a detecção de intrusões. Os detalhamentos dos algoritmos são apresentados por meio de pseudocódigos.
- O Capitulo 4 apresenta uma análise para justificar a escolha do conjunto de dados NSL-KDD, utilizado nos experimentos, e discute a forma de uso no ambiente WEKA, utilizada para a configuração e aplicação dos algoritmos.
- O Capítulo 5 apresenta dois tipos de experimentos e as configurações utilizadas para a realização dos experimentos de detecção de intrusão DA (Detecção de Anomalia) e DTA (Detecção de Tipo de Anomalia) para os algoritmos KNN, K-Means++ e J48. Também são apresentados os esquemas de experimentação adotados, atributos dos conjuntos de dados utilizados nos experimentos, as matrizes de confusão e sua interpretação e os resultados dos experimentos DA e DTA, seguido de uma breve discussão comparativa.
- O Capítulo 6 apresenta a análise dos resultados dos experimentos DA e DTA, conclusões e trabalhos futuros.

Capítulo 2

Descoberta de Conhecimento e Detecção de Anomalias

O termo “Descoberta de Conhecimento em Banco de Dados” (KDD – *Knowledge Discovery in Databases*) surgiu no primeiro *workshop* de KDD em 1989. Segundo Fayyad *et al.* (1996), KDD é o processo, não trivial, de extração de informações implícitas, previamente desconhecidas e potencialmente úteis, a partir de grandes volumes de dados armazenados, geralmente, em um banco de dados.

Nessa definição, um conjunto de dados representa fatos de um determinado domínio de conhecimento e, a partir desses dados, padrões ou modelos são descobertos. Os padrões descobertos devem ser novos, compreensíveis e úteis, ou seja, deverão ser relevantes para o domínio de dados.

Para descobrir conhecimento que seja relevante é importante estabelecer metas bem definidas. Segundo Fayyad *et al.* (1996), no processo de descoberta de conhecimento, as metas são definidas em função dos objetivos do processo, podendo ser de dois tipos básicos: a previsão ou a descrição. Em termos gerais, na previsão, o sistema irá encontrar padrões com o propósito de estimar o comportamento futuro de algumas entidades, enquanto que, na descrição, o sistema deverá encontrar padrões com o propósito de apresentá-los em uma forma compreensível para seu uso futuro. As fronteiras entre previsão e descrição, entretanto, não são bem definidas (Fayyad *et al.*, 1996).

A descoberta de conhecimento em uma base de dados pode geralmente ser abordada como um processo com cinco etapas: seleção de dados, pré-processamento ou limpeza dos dados, transformação, mineração de dados e interpretação dos resultados, conforme apresentado na Figura 2.1.

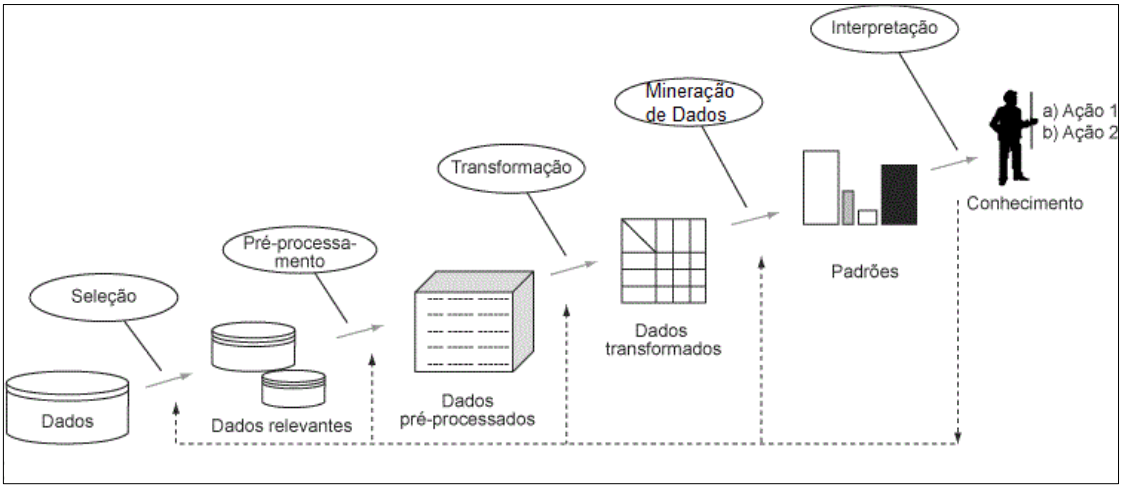


Figura 2.1 - Etapas que compõem o processo da descoberta de conhecimento, apresentadas em (Fayyad *et al.*, 1996)

Na seleção são recuperados os dados relevantes para o processo de descoberta de conhecimento. Para isso é necessário um profundo conhecimento do domínio do problema uma vez que é necessário saber o que se deseja procurar.

Já no pré-processamento, os dados selecionados são submetidos a um processo de "limpeza" relacionada com a detecção, remoção ou tratamento de dados inválidos, eliminação de dados inconsistentes ou duplicados que não serão utilizados no processo.

O processo de transformação é feito para reduzir o volume dos dados e adaptá-los para os algoritmos usados na etapa de mineração. Assim por exemplo, categorias de produtos tais como áudio, vídeo, suprimentos, aparelhos eletrônicos, câmera, e acessórios podem ser transformados em códigos 0,1,2,3,4 e 5.

Na etapa de mineração de dados são utilizados algoritmos para extrair conhecimento a partir dos dados. Segundo Fayyad *et al*. (1996), às vezes o termo mineração de dados é usado como sinônimo de todo o processo de descoberta de conhecimento. A seguir são apresentados alguns exemplos que ilustram o tipo de conhecimento que pode ser extraído dos dados:

- Sempre que um cliente compra um equipamento de vídeo também compra outro aparelho eletrônico. Isso é um exemplo de regra de associação que pode ser descoberta.
- Suponha que um cliente compre uma câmera e dentro de três meses compre suprimentos fotográficos e, depois de seis meses, provavelmente comprará

um item de acessório. Isso define um padrão de transações conhecido como padrão sequencial.

- Os clientes de um estabelecimento podem ser classificados por frequências de visitas, tipos de financiamento utilizado, valor da compra ou afinidade por determinados produtos. Isto pode ser feito com um mecanismo de classificação como, por exemplo, uma árvore de decisão.

Já na etapa final, os resultados obtidos na mineração são interpretados e analisados. Os resultados da mineração podem ser informados através de diversos formatos, como listagens, saídas gráficas, tabelas de resumo ou visualizações. A seguir, algumas etapas do processo de descoberta de conhecimento são apresentadas com mais detalhes.

2.1.1. Pré-Processamento e Transformação de Dados

A descoberta de conhecimento enfrenta muitos desafios e um dos principais é não dispor de dados de boa qualidade, já que a descoberta de conhecimento é feita a partir dos dados. Todo o processo de mineração é comprometido quando os dados têm erros, gerando resultados inconclusivos e que podem levar ao erro de quem os interpreta. Faz-se necessário então um pré-processamento, que pode ser realizado manualmente ou de forma automatizada (Camilo & da Silva, 2010).

Os dados utilizados no processo de KDD se originam em diferentes fontes e têm estruturas diversas: registros, grafos e redes, sequências ordenadas e dados multimídia. A estrutura de dados mais utilizada é o registro composto por um conjunto de campos que representam as diferentes características ou atributos do objeto de dados associado com o registro.

Em geral dados estão sujeitos a erros. Basicamente existem dois tipos de erros: os sistemáticos e os não sistemáticos. Os erros sistemáticos podem acontecer, por exemplo, com a falha na calibração de um equipamento, e são introduzidos de forma previsível e são potencialmente detectáveis e corrigíveis. Os erros não sistemáticos (ruídos) são introduzidos de forma imprevisíveis e são difíceis de detectar e corrigir. Para o problema da detecção de intrusão, o ruído nos dados pode ser a informação que leva à detecção de intrusão.

Um outro problema que os dados podem apresentar é a falta de alguns valores nos atributos. Neste caso, a causa pode estar na coleta dos dados, remoção de dados, inconsistências ou pelo próprio significado do atributo ser incompreensível. Segundo Camilo & da Silva (2010), no caso de ausência de valores de atributos, existem as seguintes alternativas para contornar o problema:

- Ignorar os registros com valores ausentes;
- Predizer o valor faltante com base nos valores de outros atributos;
- Assumir os valores faltantes e controlar essa situação dentro dos algoritmos que utilizam esses dados;
- Substituir os valores ausentes para um atributo pela moda[2] ou pela média[3] ou a mediana[4] (a média e a mediana se aplicam só a valores contínuos).

Outro problema recorrente é a redundância de dados. Atributos diferentes de um mesmo dado podem conter praticamente a mesma informação. Por exemplo, data de nascimento e idade. Uma das fontes de redundância pode ser a integração de dados de fontes diferentes.

Em alguns domínios de aplicação existe o problema de dados dependentes do tempo, como por exemplo dados provenientes do mercado acionário, que mudam dinamicamente ou, então, dados de pacientes com doenças crônicas.

Contudo, Fayyad *et al*. (1996) afirmam que para a eficiente incorporação das técnicas de mineração de dados, antes de tudo, é necessário um processo de preparação dos dados que contempla as seguintes etapas:

1) Integração dos dados: remover inconsistências nos nomes ou em valores de atributos de diferentes origens;
2) Limpeza dos dados: detectar e corrigir erros nos dados, substituir valores perdidos, etc.;
3) Conversão de dados nominais, ou em forma de códigos, para números inteiros;

[2] A moda é o valor que detém o maior número de observações, ou seja, o valor que ocorre com maior frequência num conjunto de dados, isto é, o valor mais comum.

[3] A média é a soma dos valores divididos pela quantidade de valores somados.

[4] A mediana é o valor numérico que separa a metade superior da metade inferior de uma amostra de dados ordenados de forma crescente ou decrescente.

4) Redução do domínio (valores possíveis) para reduzir a distribuição dos valores no espaço de valores originalmente possíveis;
5) Construir ou derivar novos atributos;
6) Discretização: transformar atributos contínuos em categóricos, quando os algoritmos utilizados não trabalham com atributos contínuos ou, então, para melhorar a compreensão do conhecimento descoberto;
7) Seleção de atributos: escolher atributos relevantes para a tarefa em questão. Por exemplo, o atributo "sobrenome do cliente", em geral, não é relevante para muitos problemas.

Dentre as etapas mencionadas acima, um dos maiores desafios é a seleção de atributos relevantes para o problema, uma vez que esse fator pode influenciar o resultado final, tanto na precisão dos resultados, quanto no desempenho dos algoritmos.

Feita a limpeza dos dados, estes passam pela etapa de transformação. Nesta etapa os dados passam para os formatos requeridos pelos algoritmos utilizados na etapa de mineração.

2.1.2. Mineração de Dados

Como já foi mencionado, dentro do processo de KDD temos a etapa de mineração de dados que é de grande relevância no processo de descoberta do conhecimento, a ponto de, muitas vezes, ser usado mineração de dados como sinônimo do processo de KDD.

A mineração de dados é definida como o processo de descoberta de padrões nos dados. O processo pode ser automático ou semiautomático. Normalmente os dados a serem analisados são encontrados em grandes volumes; o desafio da mineração de dados encontra-se em processar grandes quantidades de dados e, para isso, se faz necessário o uso de algoritmos apropriados (Witten *et al.*, 2011).

No assunto mineração de dados, Witten *et al.* (2011) colocam em evidência que os dados são armazenados eletronicamente e o processo de busca é automatizado ou, pelo menos, apoiado computacionalmente. Mesmo isso não é particularmente novo, pois economistas, estatísticos, analistas e engenheiros de comunicação há muito tempo trabalham com a ideia de que padrões nos dados podem ser procurados automaticamente,

identificados, validados e utilizados para a previsão. O que é novo é o aumento vertiginoso das oportunidades para encontrar padrões em dados.

O crescimento desenfreado das bases de dados nos últimos anos traz a mineração de dados para a vanguarda das novas tecnologias. Estima-se que a quantidade de dados armazenados em bancos de dados do mundo dobra a cada 20 meses. Neste cenário a mineração de dados torna-se uma importante opção para a identificação de padrões escondidos. Dados analisados de forma inteligente são recursos valiosos e podem levar a novas perspectivas e, em ambientes comerciais, a vantagens competitivas (Witten *et al.*, 2011).

2.1.3. Pós-Processamento

A etapa de pós-processamento pode ser definida pelos processos de filtragem, estruturação e classificação dos resultados obtidos na mineração. Somente após esta fase, o conhecimento descoberto é apresentado ao usuário. O conhecimento descoberto pode ser filtrado por alguma medida estatística, por exemplo, suporte, confiança ou outro critério definido pelo usuário. Estruturação significa que o conhecimento pode ser organizado de forma hierárquica (Camilo & da Silva, 2010).

O conhecimento gerado pelo processo de KDD é, via de regra, utilizado para dar suporte à tomada de decisões humana na resolução de problema em domínios específicos.

2.2. Mineração de Dados e as Tarefas de Classificação e Agrupamento

Os bancos de dados são ricos em informações ocultas que podem ser utilizadas para as tomadas de decisões. Por exemplo, a partir dos dados disponíveis em uma organização pode ser construído um modelo de classificação para categorizar as aplicações financeiras ou empréstimos bancários como seguros ou de risco, ou pode ser feito um modelo de previsão para os gastos em valores monetários de clientes potenciais com equipamentos de informática, conforme a sua renda e ocupação. Essas formas de análise dos dados são denominadas de tarefas e podem ser classificadas como preditivas ou descritivas.

As tarefas preditivas geram modelos que caracterizam propriedades de um conjunto de dados. O modelo permitirá mapear dados em saídas discretas que representam rótulos de classes através das chamadas tarefas de classificação, ou o modelo estabelece relações entre variáveis independentes (chamadas de preditoras) e um variável dependente (chamada de resposta) através das chamadas tarefas de regressão. Muitos métodos de predição foram propostos por pesquisadores nos campos de aprendizado de máquina, reconhecimento de padrões e estatística (Webb & Copsey, 2011).

Já as tarefas descritivas são úteis para descobrir relacionamentos ocultos em grandes conjuntos de dados. Entre os métodos de descrição podem ser mencionados os de agrupamento, os que descobrem regras que descrevem associações, os de detecção de anomalias e os de detecção de padrões sequenciais (Theodoridis & Koutroumbas, 2009).

A seguir são apresentados alguns conceitos associados com as tarefas de classificação e agrupamento, que são utilizadas na dissertação.

2.2.1. A Tarefa de Classificação

A tarefa de classificação consiste em construir um modelo que possa ser aplicado a dados não classificados visando categorizá-los em classes. Um dado é examinado e classificado de acordo com um conjunto de classes predefinidas (Han & Kamber, 2006).

As tarefas de classificação podem ser utilizadas nos seguintes contextos:

- Um funcionário de um banco, responsável pelas análises das propostas de empréstimos para clientes, precisa fazer a análise dos dados a fim de saber se os empréstimos para esses clientes são seguros ou de risco para o banco;
- Um gerente de Marketing de uma empresa que produz aparelhos eletrônicos, precisa da análise dos dados para ajudar a prever se um cliente com um determinado perfil vai comprar um novo computador;
- Um pesquisador médico quer analisar dados de câncer de mama a fim de prever qual tratamento deve receber um paciente.

Nesses exemplos um modelo ou classificador é construído para prever rótulos tais como *"seguro" ou "de risco",* para os dados de um pedido de empréstimo, *"Sim"* ou

"Não", para o exemplo da compra de um novo computador ou *"tratamento A", "tratamento B" ou "tratamento C"* para os dados médicos.

Han & Kamber (2006), relatam que estas categorias podem ser representadas por valores discretos, em que a ordenação entre os valores não tem qualquer significado. Por exemplo, os valores 1, 2, e 3, podem ser usados para representar os tratamentos A, B, e C, onde não há nenhuma ordenação implícita entre as classes de tratamento.

Na tarefa de classificação são utilizadas técnicas ou algoritmos de aprendizado supervisionado. No aprendizado supervisionado, o objetivo é induzir conceitos a partir de exemplos que estão pré-classificados, ou seja, exemplos que estão rotulados com uma classe conhecida (Han & Kamber, 2006).

As técnicas utilizadas em classificação constroem um modelo; esse modelo é criado com base na análise de um conjunto de dados, chamado conjunto de treinamento, cuja classificação é conhecida. O modelo obtido permite predizer a classificação de futuros dados como, por exemplo, classificar um cliente que pede um empréstimo como "seguro" ou de "risco".

O modelo obtido pelos algoritmos de classificação pode ser representado como uma árvore de decisão (algoritmos ID3, C4.5, J48, etc.), um conjunto de regras de classificação ou uma rede Bayesiana dentre vários outros.

2.2.2. A Tarefa de Agrupamento *(Clustering)*

A tarefa de agrupamento consiste em, dado um conjunto de dados, evidenciar um conjunto de agrupamentos, com base na similaridade entre esses dados. No agrupamento são utilizadas técnicas ou algoritmos de aprendizado não supervisionado. Neste tipo de aprendizado, existe incerteza sobre a saída esperada. Os algoritmos de agrupamento ou *clustering* geralmente dividem os dados em grupos ou *clusters*, nos quais a distância dos dados dentro do mesmo *cluster* é mínima e a distância entre dados de diferentes *clusters* é máxima. Encontrar a solução ótima para o problema ainda é uma questão em aberto.

A entrada para um algoritmo de agrupamento, normalmente, é um vetor de atributos ou tupla. Os tipos de dados desses atributos podem ser numéricos, categóricos ou booleanos.

Para dados numéricos, a forma mais natural de definir similaridade é através do cálculo de distâncias. Tradicionalmente usam-se valores de similaridade entre 0 e 1, sendo 0 para dados sem similaridade e 1 para dados idênticos.

Para o cálculo de distância, os dados inicialmente devem ser normalizados para que os valores das distâncias fiquem no intervalo de 0 a 1. Uma vez obtida a distância, o grau de similaridade é calculado como $1 - d$, onde d é a distância entre os dados.

Como neste estudo abordam-se algoritmos que usam as medidas de similaridade com técnicas baseadas em distância faz-se necessário uma discussão sobre tipos de medidas utilizadas para calcular a distância entre dois dados.

Por exemplo, para encontrar a distância entre dois dados, representados por pontos em um plano, basta encontrar o comprimento do segmento da reta que os une. Já no caso de encontrar a distância entre duas cidades, isso não seria adequado, uma vez que não necessariamente tem como se chegar de uma cidade a outra em linha reta. Nesse caso, uma medida de distância mais adequada seria encontrar o comprimento do segmento da curva que liga as duas cidades através de estradas. A Figura 2.2 apresenta alguns exemplos dos principais tipos de distância entre dois pontos do plano.

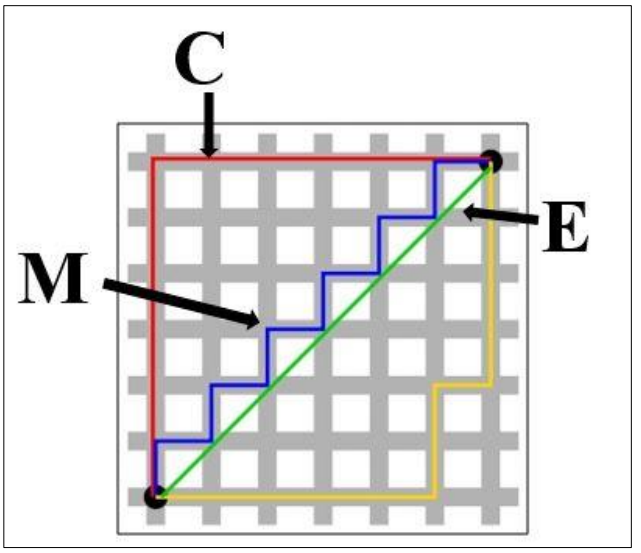


Figura 2.2 - Exemplo de distâncias Euclidiana (E), Manhattan (M), Chebyshev (C), em um plano bidimensional adaptado de (Wikipédia, a enciclopédia livre, s.d.).

Na Figura 2.2 a distância Euclidiana é representada pela linha identificada pela letra “E” ligando os dois pontos, e é considerada a menor distância entre eles. A distância Manhattan é representada pela linha identificada pela letra “M”, e pode ser considerada

como a menor distância que contorna os obstáculos entre os dois pontos. Já a distância Chebyshev é identificada pela letra "C".

Dados dois dados associados com os pontos E_i e E_j pertencentes a um espaço M-dimensional, notados respectivamente por $E_i = (x_{i1}, x_{i2}, ..., x_{iM})$ e $E_j = (x_{j1}, x_{j2}, ..., x_{jM})$, a distância Euclidiana *(dist)* entre esses dois pontos E_i e E_j (ou, alternativamente, entre E_j e E_i) é dada pela Equação 2.1. A distância Euclidiana entre os pontos E_i e E_j representa o comprimento do segmento de reta que os conecta.

$$dist(E_i, E_j) = \sqrt{\sum_{l=1}^{M} (x_{il} - x_{jl})^2} \qquad (2.1)$$

A distância Euclidiana é utilizada por alguns algoritmos como, por exemplo, o KNN apresentado no Capítulo 3. Uma alternativa à medida de distância da Equação 2.1 é a distância Euclidiana ao quadrado, definida pela Equação 2.2. A grande vantagem dessa medida é a diminuição do tempo computacional para efetuar o cálculo (Han & Kamber, 2006). A distância Euclidiana ao quadrado não é uma métrica, uma vez que não satisfaz à desigualdade triangular, mas, contudo, é frequentemente usada em problemas de otimização em que distâncias apenas têm que ser comparadas (Wikipédia, a enciclopédia livre, s.d.).

$$dist(E_i, E_j) = \sum_{l=1}^{M} (x_{il} - x_{jl})^2 \qquad (2.2)$$

Uma outra distância usada é a distância Manhattan definida pela equação 2.3 (Han & Kamber, 2006).

$$dist(E_i, E_j) = \sum_{l=1}^{M} |x_{il} - x_{jl}| \qquad (2.3)$$

Esta distância leva esse nome pois era o cálculo utilizado pelos taxistas da ilha de Manhathan para ir de um lugar a outro, já que nas cidades é praticamente impossível

estabelecer uma rota entre dois pontos através de uma reta devido ao fato das cidades serem frequentemente subdivididas em quadras.

Por último, a distância Chebyschev que é definida pela Equação 2.4 e representa a máxima diferença absoluta entre os valores de cada componente (Han & Kamber, 2006).

$$dist(E_i, E_j) = max_{l=1}^{M} |x_{il} - x_{jl}| \qquad (2.4)$$

Existem outras formas de calcular distância ver, por exemplo (Minkowski, Mahalanobis, etc) relatadas por Linden (2009), mas as mencionadas anteriormente são as mais utilizadas pelos algoritmos de agrupamento.

Capítulo 3

Algoritmos de Detecção de Intrusão

Neste capítulo são apresentadas as descrições dos algoritmos utilizados na detecção de intrusão assim como alguns conceitos relacionados.

A Seção 3.1 descreve o algoritmo KNN pertencente à família de algoritmos baseados em instâncias. A Seção 3.2 descreve o algoritmo K-Means, um dos algoritmos de agrupamento mais utilizados e que é a base do algoritmo K-Means++, disponível no ambiente WEKA (versão 3.6.12). Por último, a Seção 3.3 apresenta o conceito de árvore de decisão e o algoritmo ID3, descritas por Quinlan (1993), que deu origem ao algoritmo C4.5 cuja implementação em Java está disponível no ambiente WEKA com o nome de J48.

3.1. Algoritmo KNN (*K-Nearest Neighbor*)

O algoritmo KNN (*K-Nearest Neighbor*) pertence à família de algoritmos IBL *(Instance-based Learning)* (Cover & Hart, 1967). Os algoritmos desta família armazenam todas as instâncias de treinamento e, quando uma nova instância é apresentada ao algoritmo para ser classificada, um conjunto de instâncias similares (próximas) à nova instância é recuperada do conjunto de treinamento e utilizada para classificar a nova instância.

No caso do algoritmo KNN, para classificar uma nova instância são recuperados os *k* vizinhos mais próximos e é atribuída, a nova instância, a classe mais frequente entre esses *k* vizinhos.

A Figura 3.1 apresenta um exemplo para ilustrar o funcionamento do algoritmo KNN. Na figura existe uma instância a ser classificada representada pela interrogação, e instâncias de treinamento já classificadas associadas à classe triângulo e à classe quadrado.

Para o valor de k = 1, pelo funcionamento do algoritmo KNN, a nova instância será classificada como pertencente à classe quadrado, uma vez que a classe do vizinho mais próximo é a classe quadrado. Caso o valor de k = 3, a classe da nova instância será triângulo, dado que duas instâncias dos três vizinhos mais próximos têm classe triângulo e uma tem classe quadrado. Já no caso de k = 7, a classe da nova instância será a classe quadrado. Neste algoritmo, o que determina a classificação é a maior frequência das classes dos k vizinhos mais próximos da instância a ser classificada.

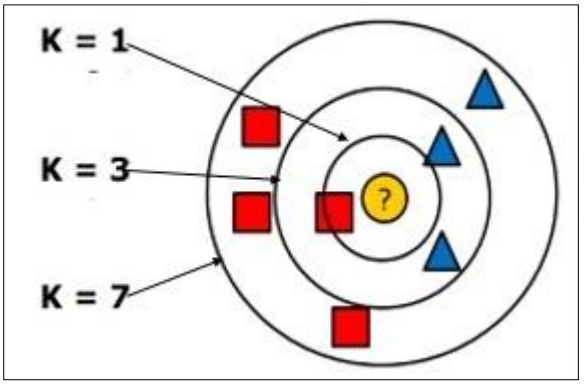


Figura 3.1–Exemplo de funcionamento do KNN, mostrando como o valor de *k*, influencia a classificação de uma nova instância, adaptado de (Wikipédia, a enciclopédia livre, s.d).

Para determinar os vizinhos mais próximos ou similares é utilizado o conceito de distância entre a instância a ser classificada e as instâncias do conjunto de treinamento mais próximas a ela. As medidas de distâncias mais utilizadas são a Euclidiana, a Manhattan e a Chebyschev, apresentadas na Seção 2.2.2. No algoritmo KNN, a medida mais usada para determinar similaridade é a distância Euclidiana. Na Tabela 3.1 é apresentado o pseudocódigo do algoritmo KNN.

O classificador KNN pertence a um grupo de algoritmos de apredizado preguiçoso, já que não constroi um modelo explícito, a partir das instâncias de treinamento. Cada vez que uma nova instância é apresentada ao algoritmo devem ser calculados os k vizinhos mais próximos da instância a ser classificada. Isto, em geral, requer cálculos da distância que podem ser onerosos para um conjunto de treinamento grande. Uma das principais limitações do uso deste algoritmo é que a busca pelos k vizinhos mais próximos necessita realizar uma passagem completa por todas as instâncias de treinamento; esta limitação torna-se um problema quando se trata de grandes volumes de dados.

Tabela 3.1–Pseudocódigo do algoritmo classificador KNN (K-*Nearest Neighbor*), adaptado de (Wu, *et al.*, 2007).

Algoritmo KNN (T, k, C, z)

Entrada: *T* é o conjunto de instâncias de treinamento, *k* o número de vizinhos mais próximos, *C* o conjuntos de valores possíveis para a classe e *z* a instância a ser classificada.

Saida: a classe de *z* (c')

Inicio

Calcular $d(x, z)$, a distância entre *z* e cada instância $x \in T$.

Selecionar $T_z = \{t_1, t_2, \dots, t_k\}$, o conjunto de *k* instâncias de treinamento mais próximas a *z*.

Calcular $c' = \underset{c \in C}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(c, Cla(t_i))$, onde $\delta(x,y) = 1$ se $x = y$ e

$\delta(x,y) = 0$ se $x \neq y$ e

Cla(t) retorna o valor da classe da instância *t*.

retornar c'

Fim

3.2. Algoritmo K-Means++ (K-Médias++)

Os algoritmos de agrupamento são algoritmos não supervisionados e lidam com instâncias não classificadas que particionam em grupos (*clusters*) baseados na similaridade entre as instâncias.

O algoritmo K-Means é um dos algoritmos de agrupamento mais utilizados. O K-Means é um algoritmo de agrupamento particional que busca encontrar a melhor partição das instâncias de entrada em *k* grupos $\{G_1, G_2, ..., G_k\}$, onde cada G_i está associado a um centroide C_i. A melhor partição seria aquela que deixa instâncias semelhantes no mesmo grupo e instâncias não semelhantes em grupos diferentes. O valor de *k* é um parâmetro do algoritmo e esse valor determina o número de grupos (*clusters*) em que são divididas as instâncias (Araar & Haddad, 2015).

O algoritmo depende do parâmetro *k* (número de grupos) definido de forma *ad hoc* pelo usuário. Isto costuma ser um problema, tendo em vista que a priori, normalmente,

não se sabe quantos *clusters* existem. A escolha de *k* pode ser feita utilizando conhecimento próprio do problema que está sendo resolvido.

Tabela 3.2 - Descrição em detalhes do K-Means, adaptado de (Real, 2014).

Algoritmo K-Means(I, k)

Entrada: o conjunto de instâncias a serem agrupadas, $I = \{I_1, I_2, ..., I_N\}$ e o número de grupos k

Saída: a partição de I em k grupos $\{G_1, G_2, ..., G_k\}$

Inicio

$C \leftarrow$*calcular_centroides*(I) { $C=\{C_1, ..., C_k\}$, centroides escolhidos aleatoriamente em I}

centroide_mudou ← **verdadeiro**

Enquanto centroide_mudou **fazer**

Inicio

Para i = 1 **até** k **fazer**

$G_i \leftarrow \varnothing$ {inicializa os grupos para a nova iteração}

Para i = 1 **até** N **fazer**

Inicio

$mp \leftarrow$ encontrar_indice_centroide_mais_perto (I_i, C)

$G_{mp} \leftarrow G_{mp} \cup \{I_i\}$

Fim

recalcular_centroides (C , $\{G_1,G_2,...G_k\}$, $\{NC_1,...,NC_k\}$) { $NC_1,...,NC_k$ *são os novos centroides calculados}*

centroide_mudou←*checar_mudanca_centroide*(C, $\{NC_1,...,NC_k\}$)

$C \leftarrow \{NC_1,...,NC_K\}$

Fim

retornar $\{G_1, G_2, ..., G_k\}$

Fim

O pseudocódigo do K-Means é descrito na Tabela 3.2. Primeiramente, o valor do parâmetro *k* é fornecido pelo usuário e *k* instâncias são escolhidas aleatoriamente (representando os centroides iniciais dos *k* grupos) do conjunto de instâncias a serem agrupadas. Toda instância do conjunto de entrada para o algoritmo é associada àquele centroide que lhe seja mais próximo; o conjunto de instâncias associadas a um centroide constitui um grupo. O centroide de cada grupo é então atualizado de maneira a refletir a

média das instâncias que pertencem ao grupo. O processo se repete até que nenhuma instância mude de grupo.

O K-Means é um algoritmo simples e eficiente que tem sido adaptado para ser usado em muitos domínios de problemas. Uma das características que o torna viável é a velocidade, geralmente convergindo em poucas iterações para uma configuração estável, na qual nenhum elemento está designado para um *cluster* cujo centro não seja o mais próximo (Linden, 2009).

Embora possa ser provado que o algoritmo K-Means sempre termina, ele não necessariamente encontra a configuração ótima de grupos e, também, é bastante sensível ao conjunto de centroides inicialmente escolhidos (Bottou & Bengio, 1995).

Para evitar um possível efeito negativo devido à escolha aleatória dos centroides iniciais foi proposta uma alteração do algoritmo K-Means, denominada K-Means++ (Arthur & Vassilvitskii, 2007), onde os centroides iniciais não são escolhidos aleatoriamente. Segundo Arthur & Vassilvitskii (2007), a ideia da modificação do algoritmo K-Means++ é selecionar um bom conjunto de centroides iniciais. O algoritmo K-Means++ só difere do algoritmo K-Means na escolha inicial dos centroides que é feita usando uma distribuição de probabilidades ponderada na qual uma instância *x* é escolhida com probabilidade proporcional ao quadrado de sua distância ao centroide mais próximo.

As vantagens encontradas com as modificações atribuídas ao K-Means++ em relação ao K-Means são:

- Melhoria no tempo de execução;
- Melhoria na qualidade do resultado;
- Melhoria nos resultados à medida que o número de *clusters* aumenta.

3.3. O algoritmo ID3

O algoritmo ID3, desenvolvido por Quinlan (1993), e os algoritmos dele derivados pertencem à família de algoritmos de aprendizado supervisionado que constroem uma árvore de decisão (modelo) a partir de um conjunto de instâncias de treinamento e essa árvore é utilizada para a classificação de novas instâncias. Estes algoritmos utilizam a estratégia de dividir para conquistar: um problema complexo é decomposto em subproblemas mais simples e recursivamente esta técnica é aplicada a cada subproblema.

Algoritmos que induzem árvores de decisão estão entre os algoritmos de classificação mais populares e são aplicados em várias áreas como, por exemplo, o diagnóstico médico e a análise de riscos de crédito. A partir das árvores de decisão é que se extraem regras do tipo *"IF-THEN"* que são facilmente interpretadas (Quinlan, 1993).

A Figura 3.2 representa uma árvore de decisão onde cada nó de decisão contém um teste para algum atributo, cada ramo descendente corresponde a um possível valor desse atributo, o conjunto de ramos são distintos, cada folha está associada a uma classe e, cada percurso da árvore, da raiz à folha corresponde uma regra de classificação. No espaço definido pelos atributos, cada folha corresponde a um hiper-retângulo (Figura 3.3) onde a intersecção destes é vazia e a união é todo o espaço (Silva, 2005).

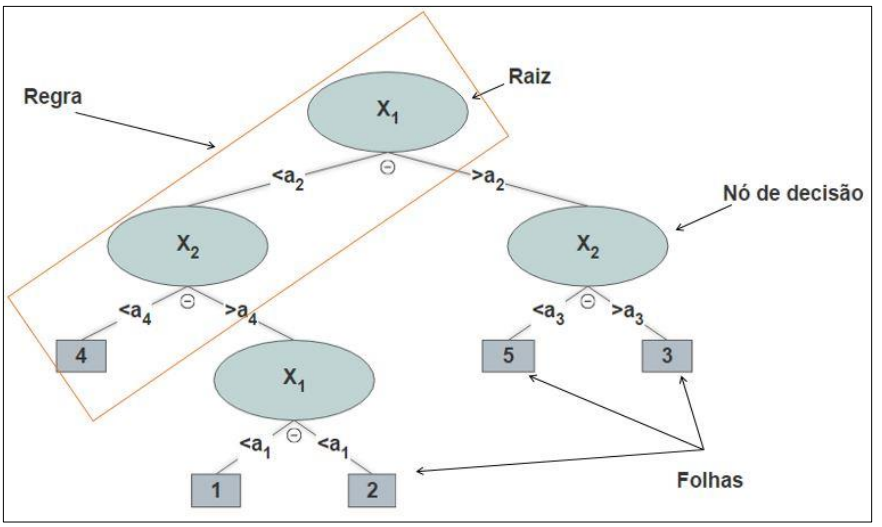


Figura 3.2 - Representação de uma Árvore de Decisão.

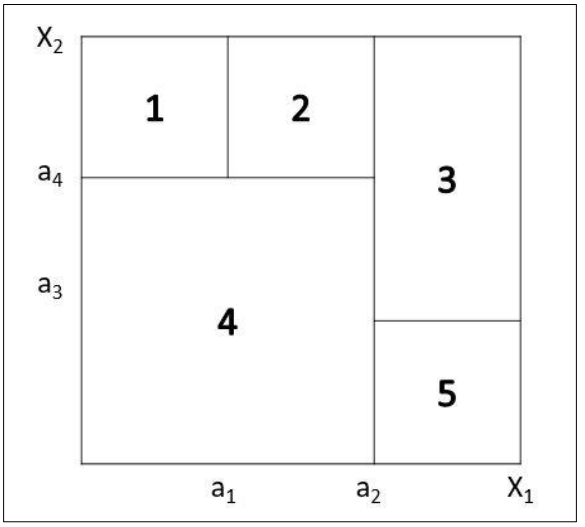


Figura 3.3 - Interpretação da Árvore de Decisão no espaço.

No algoritmo ID3 o critério para a escolha do atributo que melhor separa um conjunto de instâncias baseia-se no cálculo de entropia e ganho de informação (Witten *et al.*, 2011). Em um conjunto de dados a entropia caracteriza a impureza dos dados; ela é uma medida da falta de homogeneidade das instâncias de entrada em relação à sua classificação. Por exemplo, a entropia é máxima (igual a 1) quando o conjunto de instâncias é completamente heterogêneo.

Dado um conjunto *I* de instâncias de entrada que podem pertencer a *c* classes diferentes, a entropia de *I* é calculada pela Equação 3.4.

$$Entropia\ (I) = -\sum_{j=1}^{c} p(I,j) \times log_2\, p(I,j) \qquad (3.4)$$

Onde, *p(I,j)* é a proporção de instâncias de *I* que pertencem à classe *j*.

O ganho de informação que resulta ao particionar um conjunto de instâncias *I* segundo um atributo *A* cujos possíveis valores são $\{a_1, ..., a_v\}$ é definido pela Equação 3.5.

$$Ganho\ (I,A) = Entropia(I) - \sum_{v \in \{a_1,\ldots,a_v\}} \frac{|I_v|}{|I|} Entropia\ (I_v) \qquad (3.5)$$

Onde, I_v é o subconjunto de *I* no qual o atributo $A = v$, e $|I|$ e $|I_v|$ representam a quantidade de elementos do conjunto *I e* I_v, respectivamente.

Dado um conjunto de instâncias *I* e uma lista de atributos *LA*, o algoritmo ID3 escolhe aquele que melhor particiona *I*, ou seja, aquele com maior ganho de informação.

O pseudocódigo do algoritmo ID3 é descrito na Tabela 3.4.

O algoritmo C4.5 foi proposto por Quinlan (1993), como uma melhora do algoritmo ID3. O algoritmo C4.5, que permite trabalhar com valores discretos e contínuos e, também, com valores de atributos faltantes na etapa de treinamento. Este algoritmo permite realizar a poda daqueles ramos da árvore que não forem significativos para o processo de decisão.

Tabela 3.3 - Descrição em detalhes do ID3, adaptado de (Dai & Ji, 2014).

Algoritmo gera_arvore(I, LA)

Entrada: I conjunto de instâncias de treinamento já associados a suas respectivas classes e LA a lista de atributos que descrevem os dados de I

Saida: Árvore de Decisão

Inicio

criar um nó N

Se mesma_classe(I) = verdadeiro **então** { todas as instâncias tem a mesma classe}

retornar N como uma folha com o rótulo da classe dos elementos de *I*

Se *LA* =∅ **então**

retornar *N* como uma folha com o rótulo da classe mais frequente em *I*

A ← melhor_atributo(I, LA) {devolve o atributo A que melhor particiona *I* usando o ganho de informação}

V← valores_do_atributo(A) { devolve o conjunto de valores possíveis para *A*}

LA←*LA* – *{A}*

Para cada $v \in V$ **fazer**

Inicio

I_v← subconjunto(I,v) { devolve o subconjunto de *I* em que A = *v*}

Se I_v=∅ **então** associar o nó N com o rótulo da classe majoritária de I

senão associar o nó N com o nó que devolve gera_arvore(I_v, LA)

Fim

retornar N

Fim

O algoritmo C4.5 foi implementado em C. Já o algoritmo J48 é uma implementação em Java do algoritmo C4.5 e está disponível no ambiente WEKA.

Capítulo 4

Conjuntos de Dados e Configurações

Neste capítulo são apresentados os conjuntos de dados KDDCup99 e NSL-KDD, a análise dos conjuntos de dados, bem como uma introdução ao ambiente WEKA.

4.1. Conjunto de Dados KDDCup99 e NSL-KDD

O conjunto de dados escolhido para os experimentos de detecção de intrusão é tão importante quanto a escolha dos algoritmos para detecção de intrusão. A precisão da detecção de intrusão e o tempo de processamento são influenciados de maneira substancial quando o conjunto de dados não está adequadamente ajustado para o domínio do problema.

Esta seção descreve os conjuntos de dados KDDCup99 e NSL-KDD que são utilizados pela comunidade de pesquisa de detecção de intrusão para a elaboração de testes consistentes e comparáveis. Também são discutidos assuntos relacionados à evolução histórica, configurações dos subconjuntos de dados, arquivos e formatos.

Para discutir o conjunto de dados KDDCup99 é importante apresentar o seu histórico. Em 1998 a DARPA (*Defense Advanced Research Projects Agency*)[5] em conjunto com o laboratório Lincoln do MIT (*Massachusetts Institute of Technology*)[6], fizeram a proposta do conjunto de dados DARPA 1998 para ser utilizado na avaliação de sistemas de detecção de intrusão.

O DARPA 1998 contém a captura do tráfego de rede em um período de sete semanas para dados de treinamento e duas semanas de dados de teste, isso contabiliza cerca de 1.419.663 conexões de redes. No total existem 38 tipos de ataques nos dados de

[5] http://www.darpa.mil/
[6] http://web.mit.edu/

treinamento bem como nos dados de teste (*Lincoln Laboratory Massachusetts Institute of Technology*, 2015).

O conjunto de dados KDDCup99 foi extraído do DARPA 1998 e foi utilizado na 3ª Competição Internacional de Descoberta do Conhecimento e Mineração de Dados. O objetivo dessa competição foi o de construir um modelo capaz de detectar conexões normais ou anómalas em uma rede (ACM - *Association for Computing Machinery*, 2015).

O conjunto de dados KDDCup99 contabiliza cerca de 4.900.000 instâncias de conexões individuais, onde cada uma das instâncias de conexão é composta por 41 atributos, e um deles representa a classe associada, cujos possíveis valores são “Normal” ou “Anomalia”. Esses atributos estão divididos em três categorias:

- A categoria A contém os atributos intrínsecos de cada conexão TCP. Estes atributos incluem, entre outros, a duração da conexão, o tipo de protocolo (TCP, UDP, etc.) e serviços de rede (HTTP, TELNET, ETC) e estão listados na Tabela 4.1;
- A categoria B tem atributos, chamados de conteúdo, que sugerem um comportamento suspeito. São atributos tais como o número de tentativas de *login*, número de acessos *root*, etc. Os atributos pertencentes a essa categoria estão listados na Tabela 4.2;
- A categoria C tem atributos com características obtidas através de uma janela de tempo de dois segundos. Informações importantes referentes a certos ataques somente podem ser obtidas levando em consideração o tempo. Estes atributos estão listados na Tabela 4.3.

Na Tabela 4.1 são apresentados atributos que contém as características básicas das conexões TCP individuais. A descrição desses atributos na detecção de intrusão é importante para a classificação de anomalias que podem fazer usos específicos, por exemplo, dos tipos de serviços, tipos de protocolos, etc. No serviço de sensoriamento de um sistema de detecção de intrusão, esses atributos constituem o cabeçalho básico do que está trafegando na rede.

Tabela 4.1 – Atributos que contém as características básicas das conexões TCP individuais, apresentado em (https://kdd.ics.uci.edu/databases/kddcup99/task.html, s.d.).

Nome	Descrição	Tipo
duration	Tempo em segundos de conexão	Contínuo
protocol_type	Tipo de conexão (TCP, UDP)	Discreto
service	Tipo de serviço no destino (HTTP, Telnet)	Discreto
src_byte	Número de bytes da origem ao destino	Contínuo
dst_byte	Número de bytes do destino à origem	Contínuo
flag	Estado da conexão (normal ou erro)	Discreto
land	1 se o host e a porta da origem e destino são os mesmos, 0 caso contrário	Discreto
wrong_fragment	Número de fragmentos “errados”	Contínuo
urgent	Número de pacotes urgentes	Contínuo

Na Tabela 4.2 são apresentados os atributos que contém informações extraídas com o auxílio de conhecimento do domínio para chegar a conclusões de quais padrões associados às conexões podem representar um determinado tipo de ataque. Um exemplo seria o registro de tentativas de *login* como usuário privilegiado em determinada conexão ou o número de acessos a arquivos.

Tabela 4.2- Atributos que contém informações sugeridas pelo conhecimento de uma conexão TCP, apresentado em (https://kdd.ics.uci.edu/databases/kddcup99/task.html,sd).

Nome	Descrição	Tipo
hot	Número de indicadores “importantes”	Contínuo
num_failed_logins	Número de tentativa de login com falha	Contínuo
logged_in	1 se o login obteve sucesso, e 0 caso contrário	Discreto
num_comprissed	Número de condições comprometedoras	Contínuo
root_shell	1 se o Shell root é obtido, 0 caso contrário	Discreto
su_attempted	1 se houver tentativa e acesso “su root”, 0 caso contrário	Discreto
num_root	Número de acessos como root	Contínuo

num_file_creations	Número de operações de criação de arquivos	Contínuo
num_shells	Números de *shell prompts* abertos	Contínuo
num_access_files	Número de operações a arquivos de controle de acesso	Contínuo
num_outbund_cmds	Número de comandos externos (sessão FTP)	Contínuo
is_hot_login	1 se o *login* pertence à lista “hot”, 0 caso contrário	Discreto
is_guest_login	1 se *login* é do tipo “guest”, 0 caso contrário	Discreto

Na Tabela 4.3 são apresentados os atributos que contém informações do mesmo *host*, ou seja, que examinam apenas as conexões nos últimos dois segundos que tenham tido o mesmo destino da conexão atual. Também são apresentados os atributos que contém informação do mesmo serviço, que examinam apenas as conexões nos últimos dois segundos que tenham o mesmo serviço que a conexão atual.

Os atributos relacionados ao mesmo *host* e ao mesmo serviço são utilizados em conjunto para determinar as características de tráfego de rede com base no tempo dos registros de conexão.

Tabela 4.3 - Atributos que contém informações sobre o tráfego de rede registrados usando uma janela de tempo de 2 segundos, apresentado em (https://kdd.ics.uci.edu/databases/kddcup99/task.html,sd).

Nome	**Descrição**	**Tipo**
count	Número de conexões para o mesmo host da conexão atual nos últimos 2 segundos	Contínuo
Mesmo Host		
serror_rate	% de conexões que tiveram erros do tipo “SYN”	Contínuo
rerror_rate	% de conexões que tiveram erros do tipo “REJ”	Contínuo
same_srv_rate	% de conexões ao mesmo serviço	Contínuo
diff_srv_rate	% de conexões a diferentes serviços	Contínuo

srv_count	Número de conexões ao mesmo serviço como conexão atual nos últimos 2 segundos	Contínuo
Mesmo serviço		
srv_serror_rate	% de conexões que tiveram erros “SYN”	Contínuo
srv_rerror_rate	% de conexões que tiveram erros “REJ”	Contínuo
srv_diffe_host_rate	% de conexões a diferentes hosts	Contínuo

Os subconjuntos de dados que compõem o KDDCup99 estão descritos na Tabela 4.4

Tabela 4.4 - Arquivos do conjunto de dados KDDCup99.

Nome dos arquivos	Descrição
kddcup.names	Lista de atributos
kddcup.data.gz	Conjunto de dados completo
kddcup.data_10_percent.gz	Subconjunto do Kddcup.data.gz com 10% dos dados.
kddcup.newtestdata_10_percent_unlabeled.gz	Subconjunto do Kddcup.data.gz com 10% dos dados sem descrição da classe do tipo de invasão.
kddcup.testdata.unlabeled.gz	Conjunto de dados de teste sem a descrição da classe do tipo de invasão
kddcup.testdata.unlabeled_10_percent.gz	Subconjunto de dados de teste sem descrição da classe do tipo de invasão,
corrected.gz	Conjunto de dados de teste com descrição da classe do tipo de invasão corretos.
training_attack_types	Lista de tipos de ataques
typo-correction.txt	Nota de correção dos dados e arquivos elaborados em 26/06/2007

É destacado em ACM - Association for Computing Machiner (2015) os tipos de ataques presentes no conjunto de dados KDDCup99, que foram agrupados em quatro grandes categorias:

- Ataques de negação de serviço - DoS (*Denial of Service Attacks*): Um ataque de negação de serviço acontece quando um invasor solicita recursos computacionais, tais como memória, processador, etc., ocupando-os totalmente e deixando o sistema indisponível para gerenciar os pedidos de recursos legítimos, ou então causando a rejeição de usuários legítimos que tem o direito de usar a máquina.
- Ataques de usuários *Root* – U2R (*User Root Attacks*): Um ataque de usuários *root* acontece quando um usuário normal do sistema, que não tem direitos de *root*, tira proveito de algumas fragilidades para alcançar o acesso *root* ao sistema.
- Ataques de usuários remotos – R2L (*Remote to User Attacks*): Um ataque de usuário remoto ocorre quando um invasor, que tem a capacidade de enviar pacotes para uma máquina através de uma rede, mas não tem uma conta na máquina, faz uso de alguma vulnerabilidade para conseguir acesso local como um usuário dessa máquina.
- Ataques por sondagem - *Probing* (*Probing Attacks*)*:* Sondagem é uma categoria de ataque por meio do qual um intruso examina uma rede para coletar informações ou, então, descobrir vulnerabilidades. Estas coletas de informação da rede são razoavelmente valiosas para um *hacker* que está ensaiando um ataque futuro. Um invasor que tem um registro de máquinas e serviços que estão acessíveis em uma determinada rede pode fazer uso desta informação para procurar pontos frágeis.

A Tabela 4.5 descreve a classe de ataques que são encontrados dentro do conjunto de dados KDDCup99. Eles estão organizados em quatro categorias principais (DoS, U2R, R2L e *Probing*), que podem ser observados na Tabela 4.11.

Alguns ataques por sondagem (*Probing*) verificam os *hosts* (ou portas), utilizando um intervalo de tempo muito maior do que dois segundos, por exemplo, um cada minuto. Portanto, uma janela de 2 segundos não permite detectar padrões deste tipo de ataque, por

isso, neste trabalho foram recalculados os valores utilizando uma janela de conexões de 100 conexões em vez de uma janela de tempo.

Tabela 4.5 - Categoria de ataques e as classes de ataques existentes no conjunto de dados KDDCup99, apresentado em (Lincoln Laboratory Massachusets Institute of Technology, s.d.).

Categorias de Ataques	Classes de Ataques das Conexões
Denial of Service Attacks (DoS): Ataques de negação de serviço.	back land neptune pod smurf teardrop
User Root Attacks (U2R): Ataques através de usuários Root.	buffer_overflow loadmodule perl, rootkit
Remote to Local Atacks (R2L): Ataques de usuários remotos que não tem contas.	ftp_write guess_passwd imap, multihop phf sp warezclient warezmaster
Probing attack: Ataques por sondagens	satan ipsweep nma portsweep

Ao contrário da maioria dos ataques DoS e por sondagem, não parecem existir padrões sequenciais frequentes nos registros de R2L e ataques U2R. Isto acontece porque os ataques DoS e sondagem se relacionam com a aceitação do pedido de conexão num período muito curto de tempo, mas os ataques R2L e U2R estão presentes em partes dos pacotes de dados e, normalmente, envolvem apenas uma única conexão.

A Tabela 4.6 informa os atributos que não estão documentados no material oficial do MIT.

Tabela 4.6 - Atributos não documentados.

Nome	Tipo
dst_host_count	Contínuo
dst_host_srv_count	Contínuo
dst_host_same_srv_rate	Contínuo
dst_host_diff_srv_rate	Contínuo
dst_host_same_src_port_rate	Contínuo
dst_host_srv_diff_host_rate	Contínuo
dst_host_serror_rate	Contínuo
dst_host_srv_serror_rate	Contínuo
dst_host_rerror_rate	Contínuo
dst_host_srv_rerror_rate	Contínuo

4.2. Conjunto de Dados NSL-KDD

Algumas pesquisas sobre detecção de anomalias relatam o uso de métodos de aprendizado de máquina que tiveram uma taxa elevada de detecção (98%), mantendo uma taxa pequena (1%) de falsos positivos. Por outro lado, considerando os IDS e ferramentas comerciais existentes, poucos produtos usam abordagens de detecção de anomalias, sendo que as pessoas que trabalham com segurança de redes consideram que essa tecnologia de detecção de intrusão ainda não está suficientemente consolidada (Tavallaee *et al.*, 2009).

Para encontrar a razão deste contraste, os autores Tavallaee *et al.* (2009), estudaram os detalhes de pesquisas feitas sobre detecção de anomalias e consideraram diversos aspectos, tais como a abordagem de aprendizado e de detecção utilizadas, os conjuntos de dados de treinamento e de teste e os métodos de avaliação. Nesse estudo foram encontrados alguns problemas no conjunto de dados KDDCup99 disponível para avaliar sistemas de detecção de intrusões baseados em rede.

Um dos principais problemas no conjunto de dados KDDCup99 é o grande número de instâncias redundantes. Foi verificado por Tavallaee *et al.* (2009), que aproximadamente 78% e 75% das instâncias estão duplicadas no conjunto de treinamento e de teste, respectivamente. Esta grande quantidade de instâncias redundantes no conjunto de treinamento faz com que os algoritmos direcionem o seu resultado para as instâncias

mais frequentes. Já no conjunto de teste, essa redundância faz com que os resultados da avaliação sejam tendenciosos, já que os métodos têm melhores taxas de detecção nas instâncias frequentes.

A seguir, as Tabela 4.7 e 4.8 mostram dados estatísticos relacionados à redundância de instâncias no conjunto de treinamento e testes.

Tabela 4.7 - Estatísticas de redundância de instâncias no conjunto KDDCup99 de Treinamento, apresentada em (Tavallaee *et al.*, 2009).

	Instâncias originais	**Instâncias distintas**	**Taxa de redução**
Ataques	3.925.650	262.178	93,32%
Normais	972.781	812.814	16,44%
Total	4.898.431	1.074.992	78,05%

Tabela 4.8 - Estatísticas de redundância de registros no conjunto KDDCup99 de Teste, apresentada em (Tavallaee *et al.*, 2009).

	Instâncias originais	**Instâncias distintas**	**Taxa de redução**
Ataques	250.436	29.378	88,26%
Normais	60.591	47.911	20.92%
Total	311.027	77.289	75.15%

Analisando os dados da Tabela 4.7 é constatada uma taxa de redução de cerca de 93,32% na quantidade de instâncias identificadas como ataques, 16,44% na quantidade de instâncias identificadas como “normal”, contabilizando uma redução total de 78,05%. Já para o conjunto de teste é verificada uma redução de 88,26% das instâncias de ataques, 20,92% de instâncias normais, totalizando uma redução de 75,15%.

Para ter um melhor entendimento do comportamento do conjunto de dados KDDCup99, Tavallaee *et al.* (2009) escolheram sete técnicas de aprendizado de máquina e, cada uma destas técnicas, foi executada três vezes com diferentes conjuntos de treinamento, com o intuito de rotular as instâncias do conjunto de dados do KDDCup99.

Depois de analisar os resultados obtidos com esses experimentos, Tavallaee *et al.* (2009) fizeram a proposta de um novo conjunto de dados que, embora seja composto por instâncias do conjunto original, não sofre dos problemas mencionados anteriormente. Segundo Tavallaee *et al.* (2009) a quantidade de instâncias nos conjuntos de treinamento e teste é satisfatória para a realização de experimentos com algoritmos de naturezas diversas. Esta vantagem torna-o acessível para executar os experimentos sobre o conjunto completo, sem a necessidade de selecionar aleatoriamente uma pequena porção.

O conjunto de dados proposto pelos autores Tavallaee *et al.* (2009) foi denominado NSL-KDD e pelas qualidades descritas nesta seção foi utilizado para realizar os experimentos apresentados nesta na dissertação.

4.2.1. Análise dos dados do conjunto NSL-KDD

Os arquivos que compõem os subconjuntos de dados NSL-KDD utilizados neste trabalho estão descritos na Tabela 4.9 e foram obtidos pelos autores Tavallaee *et al.* (2009) através de experimentos a partir do conjunto de dados KDDCup99, descrito na Seção 4.2. Os arquivos em formato TXT foram criados para serem manipulados por ferramentas como o Microsoft Excel ou a ferramenta R e os arquivos em formato ARFF (*Attribute-Relation File Format*)[7] para serem manipulados no ambiente WEKA.

Tabela 4.9 - Arquivos do conjunto de dados NSL-KDD.

Nome dos arquivos	Descrição
KDDTrain+.ARFF	Arquivo NSL-KDD de treinamento completo com atributos binários em formato ARFF.
KDDTrain+.TXT	Arquivo de treinamento NSL-KDD completo que inclui classe do tipo de ataque e nível de dificuldade [8] em formato CSV (*Comma Separated Values*).
KDDTrain+_20Percent.ARFF	Arquivo contendo 20% do arquivo KDDTrain+.arff.
KDDTrain+_20Percent.TXT	Arquivo com 20% do arquivo KDDTrain+.txt.

[7] Um arquivo ARFF (Atributo-Relação File Format) é um arquivo de texto ASCII que descreve uma lista de instâncias que compartilham um conjunto de atributos. Os arquivos ARFF foram desenvolvidos pelo Projeto de Aprendizado de Máquina do Departamento de Ciência da Computação da Universidade Waikato para uso com o software de aprendizado de máquina Weka. Site do Projeto: http://www.cs.waikato.ac.nz/ml/weka/ .

[8] O nível de dificuldade foi calculado por Tavallaee *et al.*, (2009 em função de quantos dos 21 classificadores aplicados por eles sobre os dados do conjunto KDDCup 99 classificaram corretamente cada instância.

KDDTest+.ARFF	Arquivo completo NSL-KDD de teste com atributos binários em formato ARFF.
KDDTest+.TXT	Arquivo completo NSL-KDD que inclui a classe do tipo de ataque e nível de dificuldade, em formato CSV.
KDDTest-21.ARFF	Subconjunto do arquivo KDDTest+.ARFF que não inclui registros com nível de dificuldade 21.
KDDTest-21.TXT	Subconjunto do arquivo KDDTest+.TXT que não inclui registros com nível de dificuldade 21.

A Tabela 4.10 descreve alguns dados quantitativos sobre as categorias e tipos de ataque dos subconjuntos de dados presentes nos arquivos em formato ARFF, que foram extraídos com a ajuda da ferramenta de manipulação do ambiente WEKA.

Tabela 4.10 - Detalhamento das quantidades de instâncias por tipos de ataques dos subconjuntos de dados que compõem o NSL-KDD.

Nome Arquivo do subconjunto de dados	**Quantidades de instâncias ɤ Tipos de ataques**					
	Total de Instâncias (%)	**Classe Normal (%)**	**Classe DoS (%)**	**Classe *Probing* (%)**	**Classe U2R (%)**	**Classe R2L (%)**
KDDTrain+.ARFF	125.973 (100%)	67.343 (53,46)	45.927 (36,46)	11.656 (9,25)	52 (0,04)	995 (0,79)
KDDTrain+_20Percent.ARFF	25.192 (100%)	13.449 (53,39)	9.234 (36,65)	2.289 (9,09)	11 (0,04)	209 (0,83)
KDDTest+.ARFF	22.544 (100%)	9.711 (43,08)	7.458 (33,08)	2.421 (10,74)	200 (0,89)	2.754 (12,22)
KDDTest-21.ARFF	11.850 (100%)	2.152 (18,16)	4.342 (36,64)	2.402 (20,27)	200 (1,69)	2.754 (23,24)

Na Figura 4.1 podemos observar o gráfico com a distribuição da quantidade de instâncias por tipo de ataque.

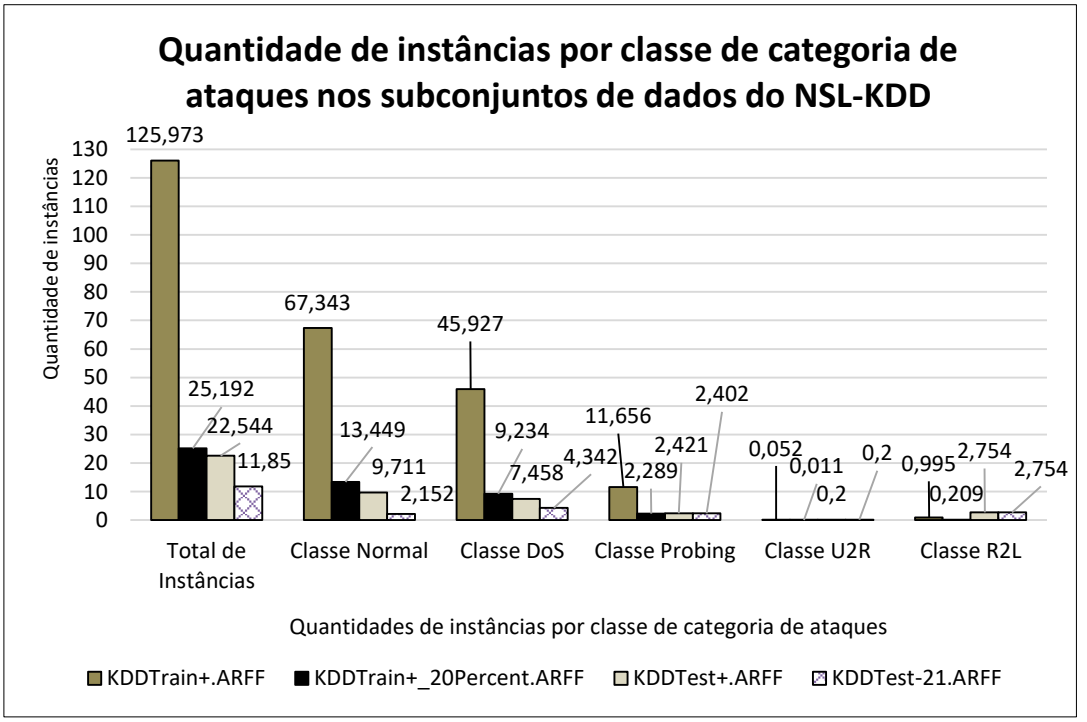


Figura 4.1 - Distribuição das instâncias por categorias de ataques nos subconjuntos de dados presentes no NSL-KDD.

Também é possível observar a equalização das quantidades de dados nos diversos arquivos que compõem o NSL-KDD. Essa equalização dos dados é um dos motivos pelo qual o conjunto de dados NSL-KDD foi escolhido para este estudo, objetivando a realização dos experimentos. A diminuição dos registros não afetou a seletividade das categorias dos ataques, proporcionando bons resultados nos experimentos, sem a influência da grande redundância e inconsistência apresentada no conjunto de dados KDDCup99. Notou-se que essa característica também é fundamental para as comparações dos resultados com outros trabalhos de pesquisa. Na Tabela 4.11 é possível observar os tipos de ataques distribuídos no conjunto de dados NSL-KDD.

Tabela 4.11 – Ataques por Categorias de ataques presentes nos subconjuntos de dados do NSL-KDD.

Ataques	Categoria de ataques
apache2	DoS
back	
land	
mailbomb	
neptune	
pod	

processtable	
smurf	
teardrop	
udpstorm	
buffer_overflow	U2R
httptunnel	
loadmodule	
perl	
ps	
rootkit	
sqlattack	
xterm	
ftp_write	R2L
guess_passwd	
imap	
multihop	
named	
phf	
sendmail	
snmpgetattack	
snmpguess	
spy	
warezclient	
warezmaster	
worm	
xlock	
xsnoop	
ipsweep	Probing
mscan	
nmap	
portsweep	
saint	
satan	

Na Tabela 4.12 é apresentada a distribuição quantitativa dos ataques por subconjunto de dados do NSL-KDD. Nota-se que em relação ao KDDCup99 a diminuição significativa dos registros é compensada pela distribuição da diversidade de ataques descritos na Tabela 4.11.

Tabela 4.12 – Quantidades de instâncias por ataque, em cada subconjunto de dados do NSL-KDD.

Ataques	KDDTRAIN+.ARFF	KDDTest+.ARFF	KDDTest-21.ARFF	KDDTrain+_20Percent.ARFF
apache2	0	737	737	0
back	956	359	359	196
buffer_overflow	30	20	20	6
ftp_write	8	3	3	1
guess_passwd	53	1.231	1.231	10
httptunnel	0	133	133	0
imap	11	1	1	5
ipsweep	3.599	141	141	710
land	18	7	7	1
loadmodule	9	2	2	1
mailbomb	0	293	293	0
mscan	0	996	996	0
multihop	7	18	18	2
named	0	17	17	0
neptune	41.214	4.657	1.579	8.282
nmap	1.493	73	73	301
normal	**67.343**	**9.711**	**2.152**	**13.449**
perl	3	2	2	0
phf	4	2	2	2
pod	201	41	41	38
portsweep	2.931	157	156	587
processtable	0	685	685	0
os	0	15	15	0
rootkit	10	13	13	4
saint	0	319	309	0
satan	3.633	735	727	691
sendmail	0	14	14	0
smurf	2.646	665	627	529
snmpgetattack	0	178	178	0
snmpguess	0	331	331	0
spy	2	0	0	1
sqlattack	0	2	2	0
teardrop	892	12	12	188
udpstorm	0	2	2	0
warezclient	890	0	0	181
warezmaster	20	944	944	7
worm	0	2	2	0

xlock	0	9	9	0
xsnoop	0	4	4	0
xterm	0	13	13	0
Total de registros	125.973	22.544	11.850	25.192

4.3. WEKA (*Waikato Environment for Knowledge Analysis*)

O ambiente WEKA (*Waikato Environment for Knowledge Analysis*) começou a ser escrito em 1993, usando Java, na Universidade de *Waikato* da Nova Zelândia sendo adquirido posteriormente por uma empresa no final de 2006. O WEKA encontra-se licenciado ao abrigo da GNU *(General Public License)* sendo, portanto, público o acesso ao código fonte.

Este ambiente tem como objetivo agregar algoritmos provenientes de diferentes abordagens da área da inteligência artificial dedicada ao estudo de aprendizado de máquina.

Uma das grandes vantagens desse ambiente está relacionada à interação através de telas no padrão GUI (*Graphical Universal Interface*), construídas na linguagem JAVA, que permitem a parametrização dos algoritmos através de listas de opções, botões e opções de checagem. Isso faz com que o tempo de trabalho seja direcionado para a preparação dos dados e a parametrização dos algoritmos a serem testados, dispensando grandes esforços na construção de ferramentas de visualização e parametrização.

Uma desvantagem dessa facilidade está no grande consumo de memória que provoca uma limitação no tamanho do conjunto de dados que será processado pelos algoritmos. Essa situação é evidenciada quando se tem algoritmos supervisionados que, no processamento, necessitam de uma etapa de treinamento e outra de teste. Esse problema é minimizado com o uso da interface através de *prompt,* que se resume apenas ao acionamento das funções com parametrizações para o processamento do conjunto de dados utilizado pelos algoritmos, sem o ambiente gráfico. Esse modo simplificado de execução permite que mais memória seja direcionada para o processamento dos conjuntos de dados.

Para evidenciar a versatilidade do ambiente, esta seção descreve um exemplo composto por um dos testes apresentados nesse trabalho. O teste utilizado determina se

uma instância é um acesso “normal” ou uma “anomalia” utilizando o algoritmo KNN. A apresentação desse algoritmo na exemplificação da ferramenta está relacionada com o fato do algoritmo ser da família do aprendizado supervisionado, necessitando de etapas de treinamento e teste, isso fará com que a explicação do uso da ferramenta seja mais completa.

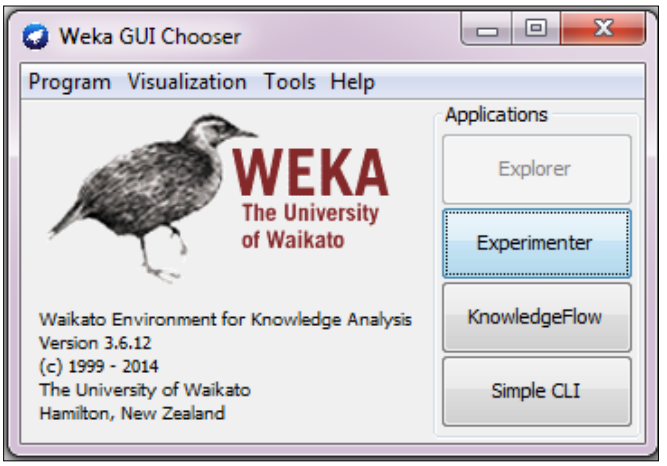


Figura 4.2 - Tela inicial do WEKA e opções iniciais para as atividades de mineração.

Na Figura 4.2 é apresentada a tela inicial da ferramenta, e para o teste em questão foi acionada a opção *Explorer* que direciona para a tela apresentada na Figura 4.3, que permite a seleção do conjunto de dados de treinamento ou teste e algumas configurações do conjunto de dados. No caso, o arquivo selecionado é o de treinamento KDDTrain+.arff, do conjunto de dados NSL-KDD, descrito na Tabela 4.9.

Também é importante salientar que nessa tela existe a opção de selecionar os tipos de filtros que podem ser usados no conjunto de dados. Alguns dos principais filtros de processamento são normalização por atributo e por instância, discretização, entre outros.

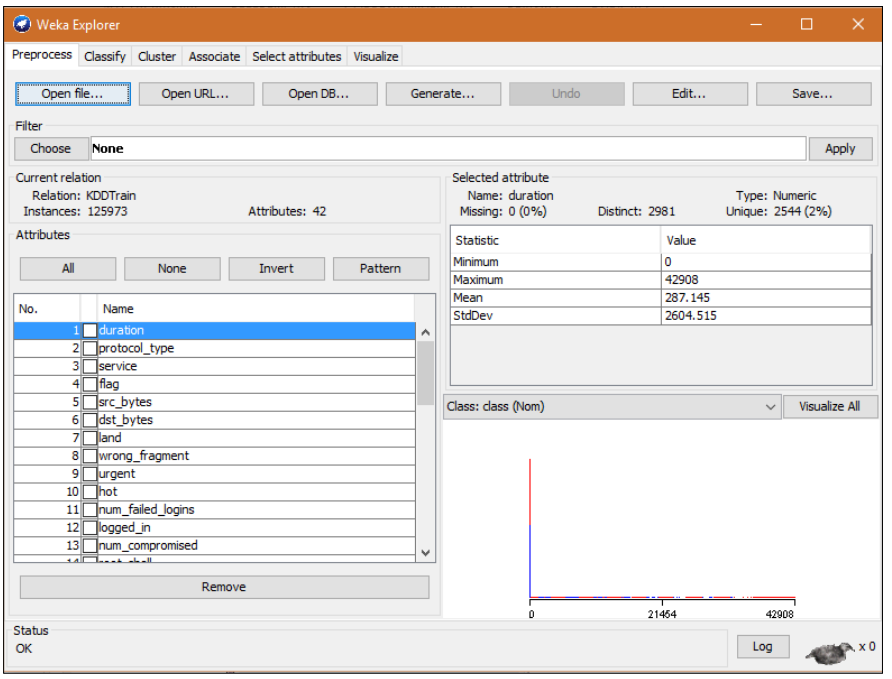


Figura 4.3 - Tela do explorador do WEKA que permitem a aplicação dos filtros e algoritmos no processamento.

Após a seleção e configuração do conjunto de dados é feita a escolha do tipo de algoritmo que será utilizado para processar o conjunto de treinamento. No caso não se optou por aplicar filtros.

Os algoritmos estão distribuídos em abas conforme o tipo de processamento que realizam; no caso do algoritmo KNN, escolhido para este exemplo, ele é listado na aba *Classify*, indicando que é um algoritmo classificador. O detalhamento da tela *Classify* está apresentado na Figura 4.4. O algoritmo KNN está dentro de um grupo denominado *Lazy* (aprendizado preguiçoso), com a denominação de IBK. Nessa tela também é feita a escolha do tipo de processamento que será empregado no algoritmo, como treinamento, teste, *Cross-Validation* e *Percentage-Split*; este último permite o particionamento do mesmo arquivo em treinamento e teste de forma cruzada. No caso do teste aqui apresentado a opção foi *usetraining set*, que permite o carregamento do conjunto de treinamento.

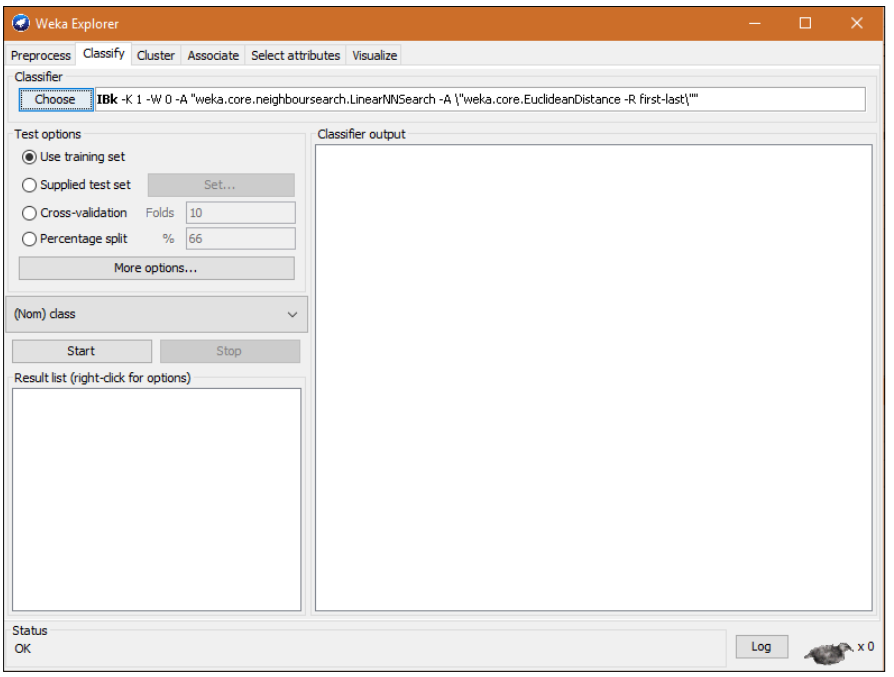


Figura 4.4 - Tela de seleção e configuração dos algoritmos de classificação.

Ao selecionar o algoritmo é possível configurá-lo para execução sobre o conjunto de treinamento. Ao clicar sobre o algoritmo é possível modificar a configuração padrão, tais como número de *k* (vizinhos mais próximos) e a medida de distância dentro do algoritmo de busca. Esses detalhes são apresentados na Figura 4.5. Para o exemplo em questão foi selecionado apenas o número de *k* vizinhos mais próximos com o valor de 1 e o *nearestNeighbourSearchAlgorithm* que assume o valor *CoverTree* com a distância Euclidiana. Os parâmetros *debug*, *distanceWeighting* e *meanSquared* permaneceram com os valores padrões do ambiente WEKA.

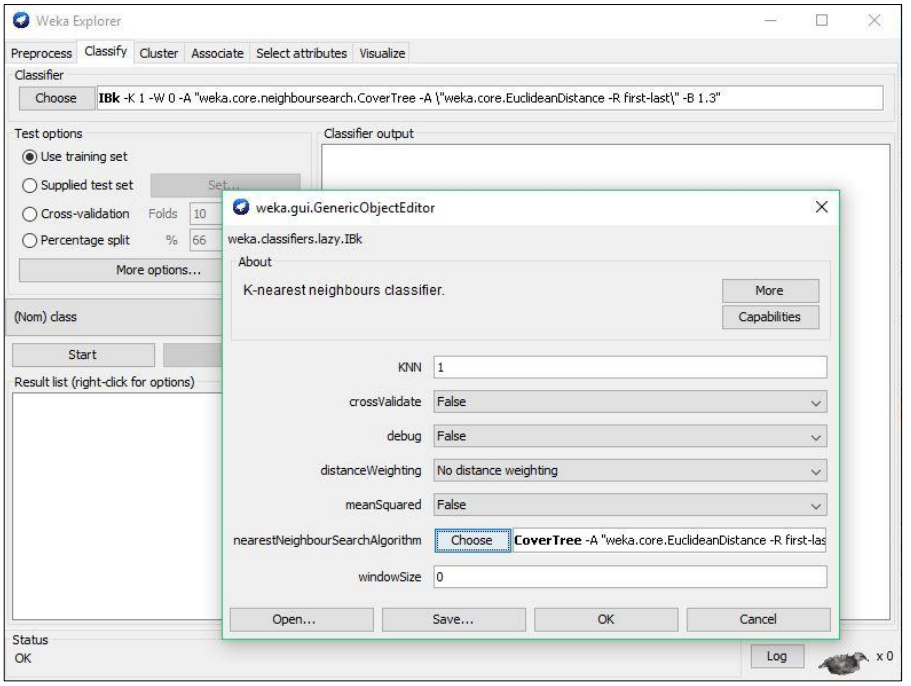


Figura 4.5 - Opções de configuração do algoritmo KNN.

O resultado do processamento do algoritmo KNN sobre o conjunto de dados de treinamento pode ser observado na Figura 4.6. Após a execução do algoritmo no conjunto de treinamento foi feito o processamento do conjunto de teste, e pode ser observado na Figura 4.7. Nessa figura também podem ser observadas algumas das informações fornecidas pelo ambiente como a quantidade de instâncias classificadas de forma correta e incorretas, alguns indicadores como o erro médio absoluto, o coeficiente Kappa, bem como o detalhamento da acurácia dos algoritmos através dos indicadores de precisão, falsos positivos e negativos para as classes em análise.

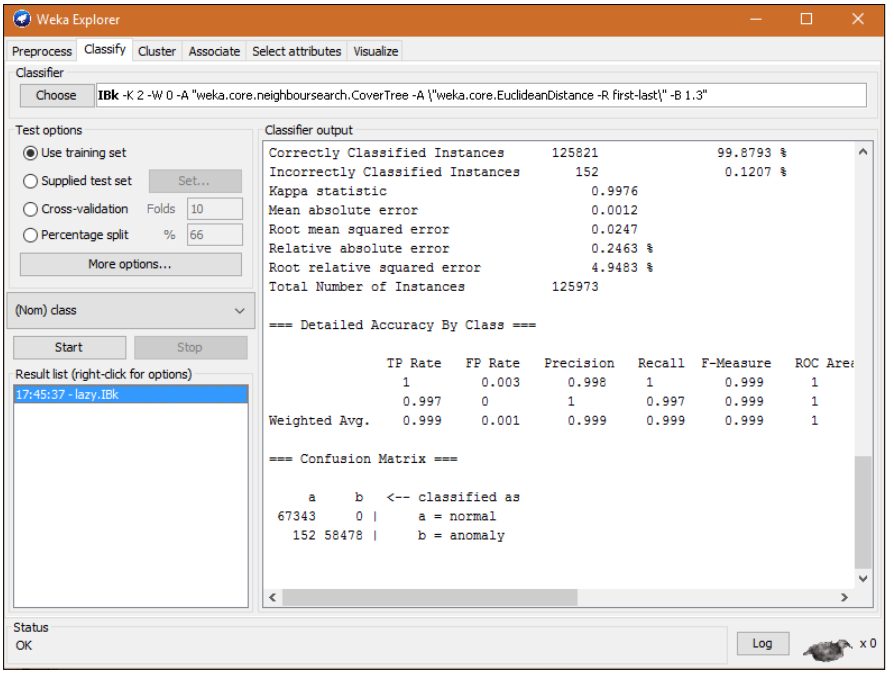


Figura 4.6 - Tela com o resultado do processamento do algoritmo KNN sobre o conjunto de treinamento.

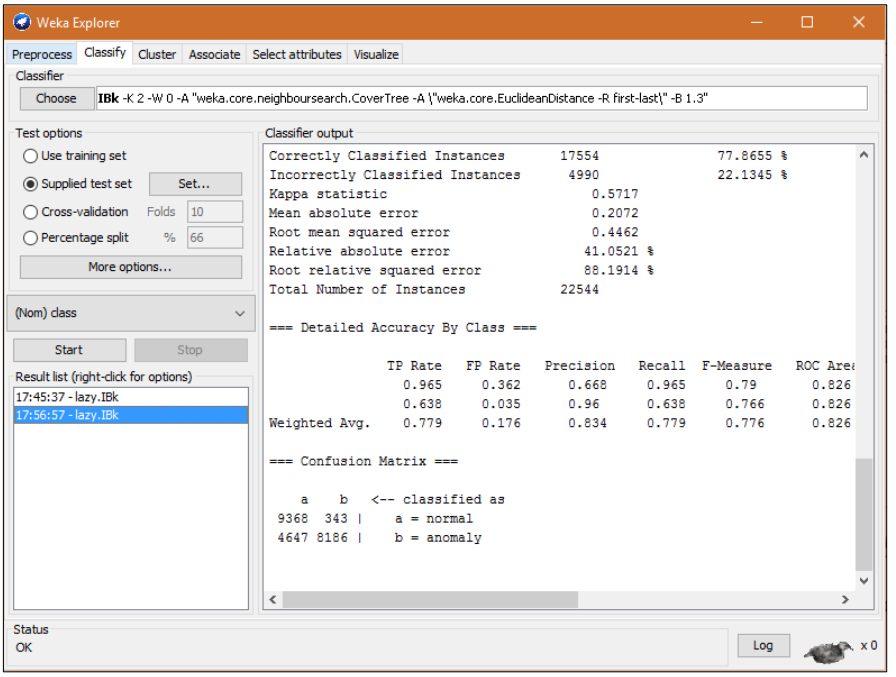


Figura 4.7 - Tela com o resultado do processamento do algoritmo KNN sobre o conjunto de teste.

Esse passo a passo foi elaborado para mostrar, de maneira simplificada, as funcionalidades do ambiente WEKA no auxílio da parametrização dos algoritmos e pré-processamento dos conjuntos de dados.

Na versão 3.6.12 do ambiente WEKA utilizada neste trabalho é possível realizar escolhas referentes aos:

- **Métodos de classificação:** Árvore de decisão induzida, Regras de aprendizagem, *Naive Bayes*, Tabelas de decisão, Regressão local de pesos, Aprendizado baseado em instância, Regressão lógistica e SVM;
- **Métodos para predição numérica:** Regressão linear, Geradores de árvores modelo, Regressão local de pesos, Aprendizado baseado em instância, Tabela de decisão e *Perceptron* multicamadas;
- **Métodos de Agrupamento:** EM, *Cobweb*, *SimpleKMeans*, *DBScan* e CLOPE;
- **Métodos de Associação:** *Apriori, FPGrowth, PredictiveApriori* e *Tertius.*

Capítulo 5

Experimentos e Resultados

Neste capítulo é apresentada uma descrição geral das configurações dos experimentos que foram realizados assim como as configurações utilizadas para a realização dos experimentos de detecção de intrusão DA (Detecção de Anomalia) e DTA (Detecção de Tipo de Anomalia) para os algoritmos KNN, K-Means++ e J48. Também são apresentados os esquemas de experimentação, atributos dos conjuntos de dados utilizados nos experimentos, as matrizes de confusão e sua interpretação e os resultados dos experimentos DA e DTA, seguidos de uma breve discussão comparativa dos resultados.

5.1. Descrição Geral do Experimento

Essa seção tem como objetivo apresentar a descrição geral das configurações dos experimentos que foram realizados. Para melhor equalização dos resultados, todos os experimentos foram realizados utilizando um computador Ultrabook Samsung i5 1,8 Giga-hertz, 8 Gigabytes de memória, HD de 200 Gigabytes SSD.

Os experimentos foram divididos em duas categorias, a primeira chamada de “Detecção de Anomalia” (DA) e a segunda “Detecção de Tipo de Anomalia” (DTA). Ambos experimentos utilizaram os algoritmos KNN, K-Means++ e J48, disponíveis no ambiente WEKA.

O objetivo do experimento DA é determinar qual dos algoritmos utilizados consegue classificar melhor as instâncias do conjunto de dados como um acesso normal ou uma anomalia. Já o experimento DTA tem como objetivo determinar qual algoritmo consegue classificar melhor as instâncias do conjunto de dados como um acesso normal ou como um acesso pertencente a uma das quatro categorias de anomalias: DoS, R2L, U2R e Probing.

Para a realização do experimento DA com os algoritmos KNN e J48, que têm natureza supervisionada, foi escolhido do conjunto de dados NSL-KDD, o subconjunto de

treinamento compatível com o ambiente WEKA denominado “KDDTrain+.arff” e para teste o subconjunto “KDDTest+.arff”. Algumas amostras desses subconjuntos podem ser observadas nas Tabelas 5.1 e 5.2.

Para o experimento com o algoritmo K-Means++, de natureza não supervisionada, foi escolhido apenas o subconjunto de dados “KDDTest+.arff”.

Tabela 5.1 - Amostras do subconjunto de treinamento KDDTrain+.arff do NSL-KDD.

0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,150,25,0.17,0.03,0.17,0.00,0.00,0.00,0.05,0.00,**normal**
0,udp,other,SF,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,1,0.00,0.00,0.00,0.00,0.08,0.15,0.00,255,1,0.00,0.60,0.88,0.00,0.00,0.00,0.00,0.00,**normal**
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,123,6,1.00,1.00,0.00,0.00,0.05,0.07,0.00,255,26,0.10,0.05,0.00,0.00,1.00,1.00,0.00,0.00,**anomaly**
0,tcp,http,SF,232,8153,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,5,0.20,0.20,0.00,0.00,1.00,0.00,0.00,30,255,1.00,0.00,0.03,0.04,0.03,0.01,0.00,0.01,**normal**

Tabela 5.2 - Amostras do subconjunto de teste KDDTest+.arff do NSL-KDD.

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,229,10,0.00,0.00,1.00,1.00,0.04,0.06,0.00,255,10,0.04,0.06,0.00,0.00,0.00,0.00,1.00,1.00,**anomaly**
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,136,1,0.00,0.00,1.00,1.00,0.01,0.06,0.00,255,1,0.00,0.06,0.00,0.00,0.00,0.00,1.00,1.00,**anomaly**
2,tcp,ftp_data,SF,12983,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,134,86,0.61,0.04,0.61,0.02,0.00,0.00,0.00,0.00,**normal**
0,icmp,eco_i,SF,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,65,0.00,0.00,0.00,0.00,1.00,0.00,1.00,3,57,1.00,0.00,1.00,0.28,0.00,0.00,0.00,0.00,**anomaly**

Para o experimento DTA, o conjunto de dados NSL-KDD foi modificado para atender a indicação do tipo de anomalia. As modificações consistiram em alterar os arquivos “KDDTrain+.txt” e “KDDTest+.txt” (algumas amostras desses subconjuntos podem ser observadas nas Tabelas 5.3 e 5.4), acrescentando o atributo *ClassCategory* que indica o tipo de anomalia (“dos”, “r2l”, “u2r” e “probing”). A outra modificação foi a

aplicação do cabeçalho específico para o formato ARFF, para atender ao requisito do ambiente WEKA. Os arquivos que foram gerados com as modificações citadas foram nomeados como "por_tipo_de_ataque-KDDTrain+.arff" e "por_tipo_de_ataque-KDDTest+.arff".

O processo de inserção dos valores no atributo *ClassCategory* foi elaborado com a aplicação de regras "*IF*", *"THEN"* e *"ELSE"* através da ferramenta Excel da Microsoft, com base no atributo *classAtack.* A relação entre os valores do atributo *classAtack* e *classCategory* está disponível na Tabela 4.11.

Algumas amostras dos conjuntos de dados modificados podem ser observadas nas Tabelas 5.5 e 5.6 e amostras do novo cabeçalho podem ser observadas nas Tabelas A.1 e A.2 do apêndice A.

Tabela 5.3 – Amostras do conjunto de dados de treinamento KDDTrain+.txt antes da transformação.

0,tcp,http,SF,287,2251,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0.00,0.00,0.00,0.00,1.00,0.00,0.43,8,219,1.00,0.00,0.12,0.03,0.00,0.00,0.00,0.00,**normal**,21
0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,2,20,1.00,0.00,1.00,0.20,0.00,0.00,0.00,0.00,**warezclient**,15
0,tcp,name,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,233,1,1.00,1.00,0.00,0.00,0.00,0.06,0.00,255,1,0.00,0.07,0.00,0.00,1.00,1.00,0.00,0.00,**neptune**,19
0,tcp,netbios_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,96,16,1.00,1.00,0.00,0.00,0.17,0.05,0.00,255,2,0.01,0.06,0.00,0.00,1.00,1.00,0.00,0.00,**neptune**,18

Tabela 5.4 – Amostras do conjunto de dados de testes KDDTest+.txt antes da transformação.

1,tcp,telnet,RSTO,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,8,0.00,0.12,1.00,0.50,1.00,0.00,0.75,29,86,0.31,0.17,0.03,0.02,0.00,0.00,0.83,0.71,**mscan**,11
0,tcp,smtp,SF,1022,387,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,3,0.00,0.00,0.00,0.00,1.00,0.00,1.00,255,28,0.11,0.72,0.00,0.00,0.00,0.00,0.72,0.04,**normal**,21
0,tcp,telnet,SF,129,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.01,0.01,0.02,0.02,**guess_passwd**,15

0,tcp,http,SF,327,467,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,33,47,0.00,0.00,0.00,0.00,1.00,0.00,0.04,151,255,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,**normal**,21

Tabela 5.5 – Amostras do conjunto de dados de treinamento modificado, denominado por_tipo_de_ataque-KDDTrain+.arff, com o atributo que especifica a categoria de ataque.

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,199,3,1.00,1.00,0.00,0.00,0.02,0.06,0.00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,**dos**,21

0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,2,20,1.00,0.00,1.00,0.20,0.00,0.00,0.00,0.00,warezclient,**r2l**,15

0,tcp,http,SF,300,13788,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,9,0.00,0.11,0.00,0.00,1.00,0.00,0.22,91,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,**normal**,21

0,icmp,eco_i,SF,18,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,1,16,1.00,0.00,1.00,1.00,0.00,0.00,0.00,0.00,ipsweep,**probing**,18

Tabela 5.6 – Amostras do novo conjunto de dados de teste modificado, denominado por_tipo_de_ataque-KDDTest+.arff, com o atributo que especifica categoria de ataque.

1,tcp,telnet,RSTO,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,8,0.00,0.12,1.00,0.50,1.00,0.00,0.75,29,86,0.31,0.17,0.03,0.02,0.00,0.00,0.83,0.71,mscan,**probing**,11

0,tcp,http,SF,267,14515,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,155,255,1.00,0.00,0.01,0.03,0.01,0.00,0.00,0.00,normal,**normal**,21

0,tcp,telnet,SF,129,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.01,0.01,0.02,0.02,guess_passwd,**r2l**,15

0,tcp,http,SF,327,467,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,33,47,0.00,0.00,0.00,0.00,1.00,0.00,0.04,151,255,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,normal,**normal**,21

5.2. Configurações dos Algoritmos no ambiente WEKA

No Capítulo 3 foram apresentados os parâmetros básicos para o funcionamento dos algoritmos KNN, K-Means++ e J48, e nesta seção serão apresentadas as combinações dos parâmetros e detalhes de pré-processamento utilizados nos experimentos DA e DTA.

Outras configurações que não são mencionadas a seguir não tiveram seus valores alterados, assumindo os valores iniciais (padrão) do ambiente WEKA.

5.2.1. KNN

O algoritmo KNN tem como parâmetros o valor de *k*, que corresponde ao número de vizinhos mais próximos, a medida de distância utilizada e a normalização como condição de pré-processamento. As combinações entre esses parâmetros, nos experimentos DA e DTA, estão descritas na Tabela 5.7.

Tabela 5.7 - Combinações de parâmetros do algoritmo KNN para os experimentos DA e DTA.

k	**NORMALIZAÇÃO**	**TIPO DE NORMALIZAÇÃO**	**NNSEARCH**	**DISTÂNCIA**
1	NÃO	-	COVERTREE	EUCLIDIANA
	SIM	ATRIBUTO		
	SIM	INSTÂNCIA		
3	NÃO	-		
	SIM	ATRIBUTO		
	SIM	INSTÂNCIA		
5	NÃO	-		
	SIM	ATRIBUTO		
	SIM	INSTÂNCIA		
7	NÃO	-		
	SIM	ATRIBUTO		
	SIM	INSTÂNCIA		

Para a opção de *NNSearch = covertree* é permitido apenas a configuração por distância Euclidiana.

5.2.2. K-Means++

O algoritmo K-Means++ tem como parâmetros o valor de *k*, que corresponde ao número de centroides, a medida de distância utilizada e, como condição de pré-processamento, a normalização. As combinações entre esses parâmetros, nos experimentos DA e DTA, estão descritas na Tabela 5.8.

Tabela 5.8 - Combinações de parâmetros do algoritmo K-Means++ para os experimentos DA e DTA.

k		**NORMALIZAÇÃO**	**TIPO DE NORMALIZAÇÃO**	**DISTÂNCIA**
DA	***DTA***			
2	5	SIM	INSTÂNCIA	EUCLIDIANA
		NÃO	-	
		SIM	INSTÂNCIA	MANHATTAN
		NÃO	-	
		SIM	ATRIBUTO	EUCLIDIANA
		SIM	ATRIBUTO	MANHATTAN

O valor de *k* para o experimento DA foi fixado em 2, pois o intuito é classificar cada instância do conjunto de dados de teste como “normal” ou “anomalia”. Já para o experimento DTA, a configuração do valor de *k* é 5, pois o intuito é classificar cada instância do conjunto de dados de teste como “normal”, “dos”, “r2l”, “u2r” ou “probing”.

5.2.3. J48

O algoritmo J48 tem como parâmetros a opção de poda ou não a árvore induzida e a normalização. As combinações entre esses parâmetros estão descritas na Tabela 5.9.

Tabela 5.9- Combinações de parâmetros do algoritmo J48 para os experimentos DA e DTA.

PODA	NORMALIZAÇÃO	TIPO DE NORMALIZAÇÃO
SIM	SIM	INSTÂNCIA
SIM	SIM	ATRIBUTO
NÃO	SIM	INSTÂNCIA
NÃO	SIM	ATRIBUTO
SIM	NÃO	-
NÃO	NÃO	-

5.3. Esquemas de experimentação

Após a criação dos subconjuntos de treinamento e teste para os experimentos DA e DTA, e o estabelecimento da configuração geral para os algoritmos KNN, K-Means++ e J48, estabeleceram-se os esquemas de experimentação para os algoritmos supervisionados e não supervisionados. Estes esquemas são apresentados na Figura 5.1 e Figura 5.2 para os experimentos com algoritmos supervisionados e não supervisionados, respectivamente.

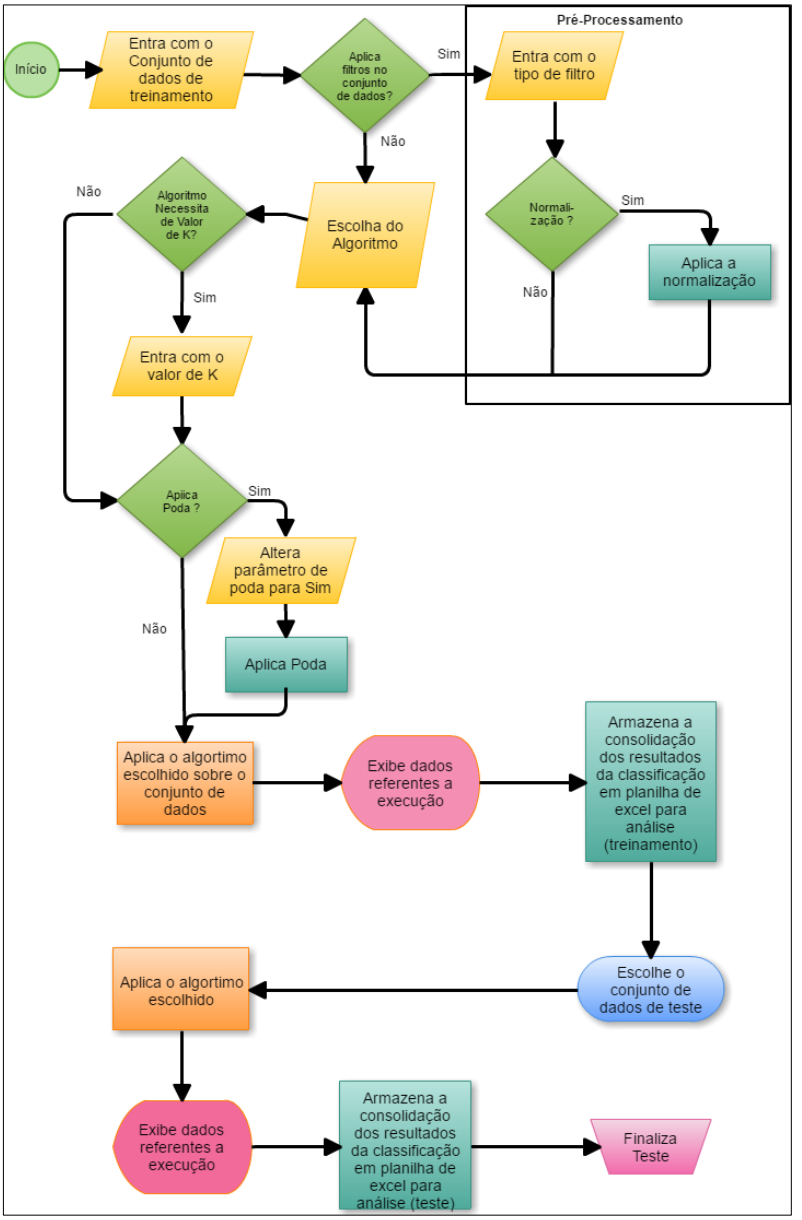


Figura 5.1- Fluxo do teste para algoritmos supervisionados, contemplando etapas de treinamento e teste.

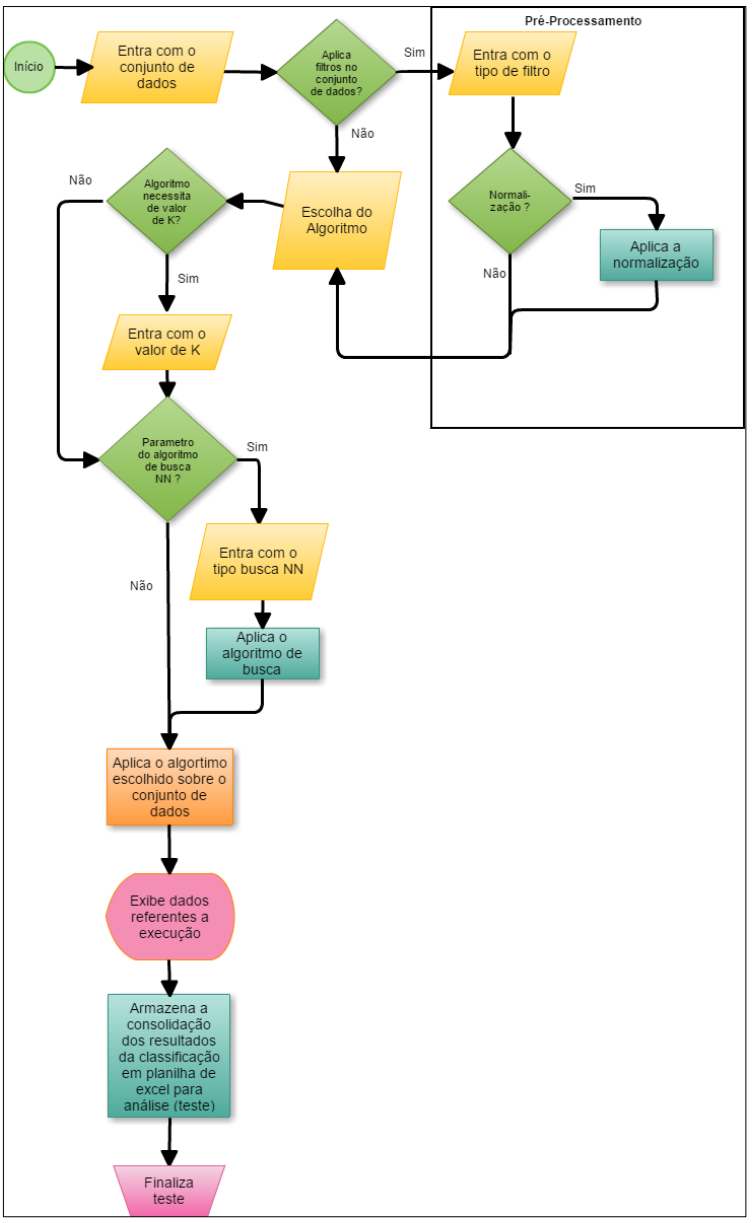


Figura 5.2 - Fluxo do teste para os algoritmos não supervisionados.

5.4. Atributos utilizados nos experimentos

Estabelecida a escolha do conjunto de dados na Seção 4.2 e os esquemas de experimentações na Seção 5.2, o passo seguinte foi a escolha dos atributos dos conjuntos de dados a serem usados nos experimentos DA e DTA.

A escolha dos atributos levou em consideração que, caso o experimento fosse aplicado em um subconjunto de dados, com a intenção de extrair as categorias do tipo Normal e Dos ou Normal e *Probing*, alguns atributos necessários para as outras categorias deveriam ser suprimidos, por não serem relevantes nesse processamento. Como neste

trabalho são consideradas simultaneamente todas as anomalias descritas na Tabela 4.11, todos os atributos foram considerados.

Para os algoritmos KNN e J48, de natureza supervisionada, no experimento DA foram utilizados todos os atributos do subconjunto de dados, descritos na Tabela 5.10, para o algoritmo K-Means++ de natureza não supervisionada, todos os atributos foram selecionados, com exceção do atributo *class*, que ficou em modo ignorado para o agrupamento; esse atributo, porém, permitiu através do ambiente WEKA, indicar a quantidade de instâncias corretamente agrupadas. Lembrando que o atributo *class* é o que índica se a instância é “normal” ou uma “anomalia”.

Para os experimentos com o algoritmo K-Means++ utilizou-se o *layout* do subconjunto de teste.

Tabela 5.10 – Atributos para experimentos com algoritmos supervisionados e não supervisionados para dos testes DA.

Nome do atributo	Tipo de dados	Valores aceitáveis
Duration	Real	Valores >= 0 (mili segundos)
protocol_type	String	{'tcp','udp', 'icmp'}
Service	String	{'aol', 'auth', 'bgp', 'courier','csnet_ns', 'ctf','daytime', 'discard','domain', 'domain_u','echo', 'eco_i','ecr_i','efs', 'exec','finger','ftp', 'ftp_data','gopher', 'harvest', 'hostnames', 'http','http_2784', 'http_443', 'http_8001','imap4', 'IRC','iso_tsap', 'klogin','kshell', 'ldap', 'link', 'login', 'mtp','name', 'netbios_dgm', 'netbios_ns', 'netbios_ssn', 'netstat', 'nnsp', 'nntp', 'ntp_u','other', 'pm_dump','pop_2', 'pop_3','printer', 'private','red_i', 'remote_job','rje', 'shell','smtp', 'sql_net','ssh', 'sunrpc','supdup', 'systat','telnet', 'tftp_u', 'tim_i', 'time', 'urh_i', 'urp_i', 'uucp', 'uucp_path','vmnet', 'whois','X11', 'Z39_50'}

Flag	String	{'OTH', 'REJ', 'RSTO', 'RSTOS0', 'RSTR', 'S0', 'S1', 'S2', 'S3', 'SF', 'SH'}
src_bytes	Inteiro	Valores >= 0
dst_bytes	Inteiro	Valores >= 0
Land	Binário	{0,1}
wrong_fragment	Inteiro	Valores >= 0
Urgente	Inteiro	Valores >= 0
Hot	Inteiro	Valores >= 0
num_failed_logins	Inteiro	Valores >= 0
logged_in	Binário	{0,1}
num_compromised	Inteiro	Valores >= 0
root_shell	Binário	{0,1}
su_attempted	Binário	{0,1}
num_root	Inteiro	Valores >= 0
num_file_creations	Inteiro	Valores >= 0
num_shells	Inteiro	Valores >= 0
num_access_files	Inteiro	Valores >= 0
num_outbound_cmds	Inteiro	Valores >= 0
is_host_login	Binário	{0,1}
is_guest_login	Binário	{0,1}
Count	Inteiro	Valores >= 0
srv_count	Real	Valores >= 0
serror_rate	Real	Valores >= 0
srv_serror_rate	Real	Valores >= 0
rerror_rate	Real	Valores >= 0
srv_rerror_rate	Real	Valores >= 0

same_srv_rate	Real	Valores >= 0
diff_srv_rate	Real	Valores >= 0
srv_diff_host_rate	Real	Valores >= 0
dst_host_count	Inteiro	Valores >= 0
dst_host_srv_count	Inteiro	Valores >= 0
dst_host_same_srv_rate	Real	Valores >= 0
dst_host_diff_srv_rate	Real	Valores >= 0
dst_host_same_src_port_rate	Real	Valores >= 0
dst_host_srv_diff_host_rate	Real	Valores >= 0
dst_host_serror_rate	Real	Valores >= 0
dst_host_srv_serror_rate	Real	Valores >= 0
dst_host_rerror_rate	Real	Valores >= 0
dst_host_srv_rerror_rate	Real	Valores >= 0
Class	**String**	**{'normal', 'anomalia'}**

Para o experimento DTA com os algoritmos KNN e J48 foram utilizados todos os atributos apresentados na Tabela 5.11, e para o algoritmo K-Means++ de natureza não supervisionada, todos os atributos dessa tabela foram selecionados, com exceção dos atributos, *classAtack, classCategory e dificult*, que ficaram em modo ignorado para classificação. Neste caso também, o atributo *classCategory* permitiu determinar a quantidade de instâncias corretamente classificadas nas classes “normal”, “dos”, “r2l”, “u2r” ou “probing”.

O *layout* do conjunto de dados que foi utilizado para o experimento DTA faz uso dos novos subconjuntos de dados que foram criados para este estudo. Os arquivos são: “por_tipo_de_ataque-KDDTrain+.arff” e “por_tipo_de_ataque-KDDTest+.arff”, descritos nas amostras das Tabelas 5.5 e 5.6 e que têm seu detalhamento de tipo de dados e valores especificados na Tabela 5.11.

Tabela 5.11 - Atributos para experimentos com algoritmos supervisionados e não supervisionados para os testes DTA.

Nome do atributo	Tipo de dados	Valores aceitáveis
Duration	Real	Valores >= 0
protocol_type	String	('tcp','udp', 'icmp')
Service	String	{'aol', 'auth', 'bgp', 'courier','csnet_ns', 'ctf','daytime', 'discard','domain', 'domain_u','echo', 'eco_i','ecr_i','efs', 'exec','finger','ftp', 'ftp_data','gopher', 'harvest', 'hostnames', 'http','http_2784', 'http_443', 'http_8001','imap4', 'IRC','iso_tsap', 'klogin','kshell', 'ldap', 'link', 'login', 'mtp','name', 'netbios_dgm', 'netbios_ns', 'netbios_ssn', 'netstat', 'nnsp', 'nntp', 'ntp_u','other', 'pm_dump','pop_2', 'pop_3','printer', 'private','red_i', 'remote_job','rje', 'shell','smtp', 'sql_net','ssh', 'sunrpc','supdup', 'systat','telnet', 'tftp_u', 'tim_i', 'time', 'urh_i', 'urp_i', 'uucp', 'uucp_path','vmnet', 'whois','X11', 'Z39_50'}
Flag	String	{'OTH', 'REJ', 'RSTO', 'RSTOS0', 'RSTR', 'S0', 'S1', 'S2', 'S3', 'SF', 'SH'}
src_bytes	Inteiro	Valores >= 0
dst_bytes	Inteiro	Valores >= 0
Land	Binário	{0,1}
wrong_fragment	Inteiro	Valores >= 0
Urgente	Inteiro	Valores >= 0
Hot	Inteiro	Valores >= 0
num_failed_logins	Inteiro	Valores >= 0
logged_in	Binário	{0,1}
num_compromised	Inteiro	Valores >= 0

root_shell	Inteiro	Valores >= 0
su_attempted	Inteiro	Valores >= 0
num_root	Inteiro	Valores >= 0
num_file_creations	Inteiro	Valores >= 0
num_shells	Inteiro	Valores >= 0
num_access_files	Inteiro	Valores >= 0
num_outbound_cmds	Inteiro	Valores >= 0
is_host_login	Binário	{0,1}
is_guest_login	Binário	{0,1}
Count	Inteiro	Valores >= 0
srv_count	Inteiro	Valores >= 0
serror_rate	Real	Valores >= 0
srv_serror_rate	Real	Valores >= 0
rerror_rate	Real	Valores >= 0
srv_rerror_rate	Real	Valores >= 0
same_srv_rate	Real	Valores >= 0
diff_srv_rate	Real	Valores >= 0
srv_diff_host_rate	Real	Valores >= 0
dst_host_count	Inteiro	Valores >= 0
dst_host_srv_count	Inteiro	Valores >= 0
dst_host_same_srv_rate	Real	Valores >= 0
dst_host_diff_srv_rate	Real	Valores >= 0
dst_host_same_src_port_rate	R**eal**	Valores >= 0
dst_host_srv_diff_host_rate	Real	Valores >= 0
dst_host_serror_rate	Real	Valores >= 0

dst_host_srv_serror_rate	Real	Valores >= 0
dst_host_rerror_rate	Real	Valores >= 0
dst_host_srv_rerror_rate	Real	Valores >= 0
classAtack	**String**	**({'normal','apache2','back','land','mailbomb','neptune','pod','processtable','smurf','teardrop','udpstorm','buffer_overflow','httptunnel','loadmodule','perl','ps','rootkit','sqlattack','xterm','ftp_write','guess_passwd','imap','multihop','named','phf','sendmail','snmpgetattack','snmpguess','spy','warezclient','warezmaster','worm','xlock','xsnoop','ipsweep','mscan','nmap','portsweep','saint','satan'}**
classCategory	**String**	**{‘normal’, ‘dos’, ‘r2l’, ‘u2r’, ‘probing’}**
Dificult	**Inteiro**	**Valores >= 0**

5.5. Avaliação de Desempenho

Uma vez obtidos os classificadores e os agrupamentos correspondentes à execução dos algoritmos, o desempenho de cada um deles deve ser avaliado. Existem diferentes métricas para avaliar os resultados obtidos. Entre essa métricas, as usadas com maior frequência são a acurácia e a taxa de erro.

Quatro conceitos são necessários para calcular essas métricas: verdadeiros positivos (VP), verdadeiros negativos (VN), falsos positivos (FP) e falsos negativos (FN). Esses conceitos serão definidos em função das instâncias positivas, que são as instâncias da principal classe de interesse e as instâncias negativas, que são todas as instâncias restantes.

- Verdadeiros positivos: instâncias positivas classificadas corretamente como positivas.
- Verdadeiros negativos: instâncias negativas classificadas corretamente como negativas.
- Falsos positivos: instâncias negativas classificadas erroneamente como positivas.
- Falsos negativos: instâncias positivas classificadas erroneamente como negativas.

Uma forma usual de apresentar os conceitos anteriores é por meio de uma tabulação cruzada entre a classe predita pelo modelo e a classe verdadeira (real) das instâncias. Essa tabulação é chamada de matriz de confusão.

A seguir será apresentada a interpretação da matriz de confusão para os experimentos DA e DTA. Logo após serão apresentadas as equações que definem as métricas utilizadas para avaliar os resultados obtidos nos experimentos.

5.5.1. Matrizes de confusão para os experimentos DA e DTA

No ambiente WEKA os resultados dos experimentos DA e DTA são disponibilizados através de matrizes de confusão com *layouts* específicos. Na Tabela 5.12 é disponibilizado o *layout* da matriz de confusão para o experimento DA e na Tabela 5.13, a matriz de confusão para o experimento DTA.

Tabela 5.12 – ***Layout*** **da matriz de confusão disponibilizada no ambiente WEKA para os experimentos DA.**

		PREDITO	
		NORMAL	**ANOMALIA**
VERDADEIRO	**NORMAL**	VP	FN
	ANOMALIA	FP	VN

Na Tabela 5.12 VP (Verdadeiros Positivos) representa a quantidade de instâncias classificadas corretamente como "normal". FN (Falsos Negativos) representa a quantidade instâncias pertencentes à classe "normal" que foram classificadas incorretamente como "anomalia". FP (Falsos Positivos) representa a quantidade de instâncias que são classificadas incorretamente como "normal" e que pertencem à classe "anomalia". VN (Verdadeiros Negativos) representam a quantidade de instâncias classificadas corretamente como "anomalia".

Tabela 5.13 - ***Layout*** **da matriz de confusão disponibilizada no ambiente WEKA, para os experimentos DTA.**

		PREDITO				
		NORMAL	**DOS**	**R2L**	**U2R**	**PROBING**
VERDADEIRO	**NORMAL**	**VP (NO|NO)**	FP (DO| NO)	FP (R2|NO)	FP (U2|NO)	FP (PR|NO)
	DOS	FP (NO|DO)	**VP (DO|DO)**	FP (R2|DO)	FP (U2|DO)	FP (PR|DO)
	R2L	FP (NO|R2)	FP (DO|R2)	**VP (R2|R2)**	FP (U2|R2)	FP (PR|R2)
	U2R	FP (NO|U2)	FP (DO|U2)	FP (R2|U2)	**VP (U2|U2)**	FP (PR|U2)
	PROBING	FP (NO| PR)	FP (DO|PR)	FP (R2|PR)	FP (U2|PR)	**VP (PR| PR)**

Na Tabela 5.13 VP (NO|NO) representa a quantidade de instâncias classificadas corretamente como "normal". VP (DO|DO) representa a quantidade de instâncias classificadas corretamente como "dos". VP (R2|R2) representa a quantidade de instâncias classificadas corretamente como "r2l". VP (U2|U2) representa a quantidade de instâncias

classificadas corretamente como “u2r”. VP (PR/PR) representa a quantidade de instâncias classificadas corretamente como “probing’.

O FP (NO|DO) representa a quantidade de instâncias classificadas incorretamente como “normal” e que pertencem à classe “dos”. FP (NO|R2) representa a quantidade de instâncias classificadas como “normal” e que pertencem a classe “r2l”. FP (NO|U2) representa a quantidade de instâncias classificadas como “normal” e que pertencem à classe “u2r”. FP (NO|PR) representa à quantidade de instâncias classificadas como “normal” e que pertencem a classe “probing’.

Por sua vez, FP (DO|NO) representa a quantidade de instâncias classificadas incorretamente como “dos” e que pertencem à classe “normal”. FP (DO|R2) representa a quantidade de instâncias classificadas como “dos” e que pertencem à classe “r2l”. FP (DO|U2) representa a quantidade de instâncias classificadas como “dos” e que pertencem à classe “u2r”. FP (DO|PR) representa a quantidade de instâncias classificadas como “dos” e que pertencem à classe “probing’.

O FP (R2|NO) representa a quantidade de instâncias classificadas incorretamente como “r2l” e que pertencem à classe “normal”. FP (R2|DO) representa a quantidade de instâncias classificadas como “r2l” e que pertencem à classe “dos”. FP (R2|U2) representa a quantidade de instâncias classificadas como “r2l” e que pertencem à classe “u2r”. FP (R2|PR) representa a quantidade de instâncias classificadas como “r2l” e que pertencem à classe “probing’.

O FP (U2|NO) representa a quantidade de instâncias classificadas incorretamente como “u2r” e que pertencem à classe “normal”. FP (U2|DO) representa a quantidade de instâncias classificadas como “u2r” e que pertencem à classe “dos”. FP (U2|R2) representa a quantidade de instâncias classificadas como “u2r” e que pertencem à classe “r2l”. FP (U2|PR) representa a quantidade de instâncias classificadas como “u2r” e que pertencem à classe “probing’.

Finalmente, o FP (PR|DO) representa a quantidade de instâncias classificadas como “probing” e que pertencem à classe “dos”. FP (PR|R2) representa a quantidade de instâncias classificadas como “probing” e que pertencem à classe “r2l”. FP (PR|U2) representa a quantidade de instâncias classificadas como “probing” e que pertencem à classe “u2r”.

Na interpretação dessa matriz de confusão, no momento que é feita a análise de um dos quadrantes, por exemplo o de VP (NO|NO), os outros quadrantes VP (DO|DO), VP (R2|R2), VP (U2|U2) E (PR|PR) tornam-se VN.

5.5.2. Equações usadas nos experimentos DA

Para avaliar os resultados obtidos nos experimentos DA foram utilizadas as seguintes métricas:

- Taxa de acurácia ($T_{ACURÁCIA1}$), que indica a proporção de acertos ocorridos na classificação
- Taxa de erro por falsos positivos para a classe "normal" (TE_{NORMAL}), que é a proporção de erros ocorridos na classificação da classe normal
- Taxa de erro por falsos positivos para a classe "anomalia" ($TE_{ANOMALIA}$), que é a proporção de erros ocorridos na classificação da classe anomalia
- Taxa de Erro Total (TE_{TOTAL}), que indica a proporção total de erros ocorridos na classificação.

No experimento DA para a $T_{ACURÁCIA1}$ utilizou-se a Equação 5.1, para TE_{NORMAL} a Equação 5.2, para a $TE_{ANOMALIA}$ a Equação 5.3 e para TE_{TOTAL} utilizou-se a Equação 5.4.

A $T_{ACURÁCIA1}$ é calculada através da soma dos valores de VP e VN dividido pela soma dos valores de VP, VN, FP e FN (total de instâncias).

$$T_{ACURÁCIA1} = \frac{VP + VN}{VP + VN + FP + FN} \tag{5.1}$$

A taxa de erro por FP para a classe "normal" (TE_{NORMAL}) é calculada pelo valor FP dividido pela soma dos valores de FP e VN.

$$TE_{NORMAL} = \frac{FP}{FP + VN} \tag{5.2}$$

A taxa de erro por FP para a classe "anomalia" ($TE_{ANOMALIA}$) é calculada pelo valor dos FN divido pela soma dos valores de VP e FN.

$$TE_{ANOMALIA} = \frac{FN}{VP + FN} \quad (5.3)$$

A taxa de erro total (TE_{TOTAL}) é calculada pelo valor da soma dos FP e FN pela soma dos VP, VN, FP e FN.

$$TE_{TOTAL} = \frac{FP + FN}{VP + VN + FP + FN} \quad (5.4)$$

5.6. Equações usadas nos experimentos DTA

Para avaliar os resultados obtidos nos experimentos DTA foram utilizadas as seguintes métricas:

- Taxa de acurácia ($T_{ACURÁCIA2}$), que indica a proporção geral de acertos ocorridos na classificação.
- Taxa de erro total (TE_{TOTAL2}), que indica a proporção total de erros ocorridos na classificação.
- Taxa de erro por falsos positivos para a classe "dos" (TE_{DOS}), que é a proporção de erros ocorridos na classificação da classe "dos".
- Taxa de erro por falsos positivos para a classe "r2l" (TE_{R2L}), que é a proporção de erros ocorridos na classificação da classe "r2l".
- Taxa de erro por falsos positivos para a classe "u2r" (TE_{U2R}), que é a proporção de erros ocorridos na classificação da a classe "u2r"
- Taxa de erro por falsos positivos para a classe "probing" ($TE_{PROBING}$), que é a proporção de erros ocorridos na classificação da classe "probing'.

No experimento DTA para a $T_{ACURÁCIA2}$ utilizou-se a Equação 5.5, para TE_{TOTAL2} a Equação 5.6, TE_{NORMAL} a Equação 5.7, TE_{DOS} a Equação 5.8, TE_{R2L} a Equação 5.9, TE_{U2R} a Equação 5.10 e para $TE_{PROBING}$ utilizou-se a Equação 5.11.

A taxa de acurácia é calculada através da soma dos valores de VP das classes "normal", "dos", "r2l", "u2r" e "probing" dividido pelo total de instâncias.

$$T_{ACURÁCIA2} = \frac{VP_{NO|NO} + VP_{DO|DO} + VP_{R2|R2} + VP_{U2|U2} + VP_{PR|PR}}{Total\ de\ Instâncias} \quad (5.5)$$

A taxa de erro total (TE_{TOTAL2}) é calculada subtraindo de 1 a $T_{ACURÁCIA2}$.

$$TE_{TOTAL2} = 1 - T_{Acurácia2} \quad (5.6)$$

A taxa de erro por falsos positivos para a classe "normal" (TE_{NORMAL}) é calculada dividindo a soma dos valores FP da classe "normal" pela soma dos valores de FP da classe "normal" e $VN_{DO,R2,\ U2\ e\ PR}$.

$$TE_{NORMAL} = \frac{Total\ de\ FP\ Normal}{Total\ de\ FP\ Normal + Total\ de\ VN_{DO,R2,U2\ e\ PR}} \quad (5.7)$$

Onde:

$$Total\ de\ FP\ Normal = (FP_{NO|DO} + FP_{NO|R2} + FP_{NO|U2} + FP_{NO|PR})$$

$$Total\ de\ VN_{DO,R2,U2\ e\ PR} = (VN_{DO|DO} + VN_{R2|R2} + VN_{U2|U2} + VN_{PR|PR})$$

A taxa de erro por falsos positivos para a classe "dos" (TE_{DOS}) é calculada dividindo a soma dos valores FP da classe "dos" pela soma dos valores FP da classe "dos" e $VN_{NO,\ R2,\ U2\ e\ PR}$.

$$TE_{Dos} = \frac{Total\ de\ FP\ Dos}{Total\ de\ FP\ Dos + Total\ de\ VN_{NO,R2,U2\ e\ PR}} \quad (5.8)$$

Onde:

$$Total\ de\ FP\ Dos = FP_{DO|NO} + FP_{DO|R2} + FP_{DO|U2} + FP_{DO|PR}$$

$$Total\ de\ VN_{NO,R2,U2\ e\ PR} = VN_{NO|NO} + VN_{R2|R2} + VN_{U2|U2} + VN_{PR|PR}$$

A taxa de erro por falsos positivos para a classe "r2l" (TE_{R2L}) é calculada dividindo a soma os valores FP da classe "r2l" divida pela soma dos valores de FP da classe "r2l" e $VN_{NO,\ DO,\ U2\ e\ PR}$.

$$TE_{R2l} = \frac{Total\ de\ FP\ R2l}{Total\ de\ FP\ R2l + Total\ de\ VN_{NO,DO,U2\ e\ PR}} \quad (5.9)$$

Onde:

$$Total\ de\ FP\ R2l = FP_{R2|NO} + FP_{R2|DO} + FP_{R2|U2} + FP_{R2|PR}$$

$$Total\ de\ VN_{NO,DO,U2\ e\ PR} = VN_{NO|NO} + VN_{DO|DO} + VN_{U2|U2} + VN_{PR|PR}$$

A taxa de erro por falsos positivos para a classe "u2r" (TE_{U2R}) é calculada dividindo a soma dos valores FP da classe "u2r" pela soma dos valores de FP da classe "u2r" e $VN_{NO, DO, R2 e PR}$.

$$TE_{U2r} = \frac{Total\ de\ FP\ U2r}{Total\ de\ FP\ U2r + Total\ de\ VN_{NO,DO,R2\ e\ PR}} \quad (5.10)$$

Onde:

$$Total\ de\ FP\ U2r = FP_{U2|NO} + FP_{U2|DO} + FP_{U2|R2} + FP_{U2|PR}$$

$$Total\ de\ VN_{NO,DO,R2\ e\ PR} = VN_{NO|NO} + VN_{DO|DO} + VN_{R2|R2} + VN_{PR|PR}$$

A taxa de erro por falsos positivos para a classe "probing" ($TE_{PROBING}$) é calculada dividindo a soma dos valores FP da classe "probing" pela soma dos valores de FP da classe "probing" e $VN_{NO, DO, R2 e U2}$.

$$TE_{Probing} = \frac{Total\ de\ FP\ Probing}{Total\ de\ FP\ Probing + Total\ de\ VN_{NO,DO,R2\ e\ U2}} \quad (5.11)$$

Onde:

$$Total\ de\ FP\ Probing = FP_{PR|NO} + FP_{PR|DO} + FP_{PR|R2} + FP_{PR|U2}$$

$$Total\ de\ VN_{NO,DO,R2\ e\ U2} = VN_{NO|NO} + VN_{DO|DO} + VN_{R2|R2} + VN_{U2|U2}$$

5.7. Experimento Utilizando o Algoritmo KNN

Essa seção trata da execução dos experimentos DA e DTA para o algoritmo KNN no ambiente WEKA, bem como as configurações e escolha da melhor configuração para comparação com os resultados obtidos com os algoritmos K-Means++ e J48.

No ambiente WEKA a configuração do algoritmo foi feita pelos parâmetros *Cross Validate: false, Debug: false, Distance Weighting: No distance Weighting, Mean Square: false, Nearest Neighbour Search Algorithm: CoverTree (Euclidian Distance).* Essas configurações são válidas tanto para etapa de treinamento quanto para a etapa de teste dos experimentos DA e DTA para o algoritmo KNN. O valor de *k* é o único valor de parâmetro variável dessa configuração, que é escolhido para cada experimento do KNN. A

configuração pode ser observada na Figura 5.3, que representa a tela de configuração do algoritmo KNN no ambiente WEKA.

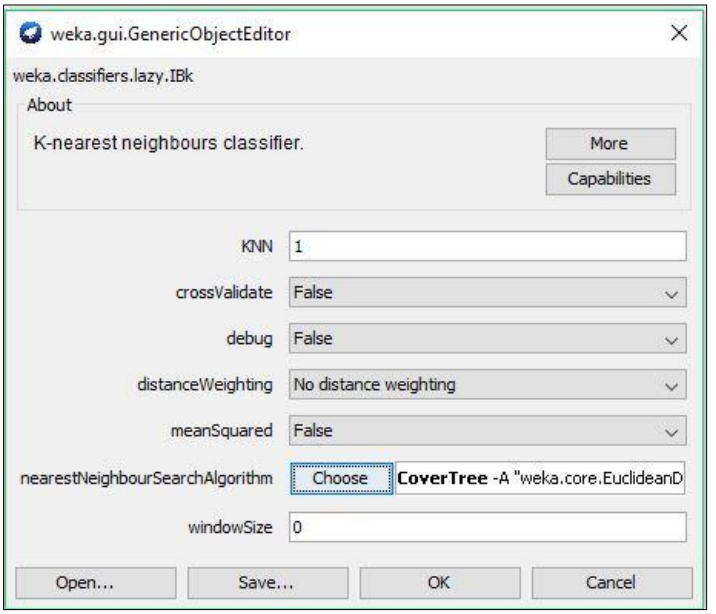


Figura 5.3 – Tela de configurações do WEKA para o algoritmo KNN

No experimento DA e DTA, para o algoritmo KNN, foram utilizados os parâmetros de pré-processamento descritos Tabela 5.7, através do ambiente WEKA. Os filtros aplicados no subconjunto de dados antes do processamento têm a função de padronizar os dados. A Tabela 5.14 exibe duas amostras, uma sem a normalização e a outra com a normalização.

Tabela 5.14 - Exemplos de instâncias sem e com o filtro de normalização.

INSTÂNCIA SEM A APLICAÇÃO DE FILTROS.
0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,150,25,0.17,0.03,0.17,0.00,0.00,0.00,0.05,0.00,normal
INSTÂNCIA COM A APLICAÇÃO DO FILTRO DE NORMALIZAÇÃO
0.0,tcp,ftp_data,SF, 0.5882352941176471, 0.0, 0,0.0,0.0,0.0,0.0,0,0.0,0.0,0.0,0.0,0.0,0.0, 0.0, 0.0, 0, 0, 0.003913894324853229, 0.003913894324853229, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.5882352941176471, 0.09803921568627451, 0.17, 0.03, 0.17, 0.0, 0.0, 0.0, 0.05, 0.0, normal

5.7.1. Algoritmo KNN no Experimento DA

O primeiro passo para o experimento DA, no ambiente WEKA, foi processar o algoritmo KNN com o subconjunto de dados "KDDTrain+.arff" como treinamento. Nessa etapa o algoritmo armazena uma base de dados de treinamento que será utilizada como modelo aplicado ao conjunto de teste. A aplicação do filtro de normalização e os valores de *k* foram estabelecidos, para cada experimento, conforme a Tabela 5.7. Após o processamento do conjunto de treinamento foi executado o algoritmo KNN para o conjunto de dados KDDTest+.arff.

Os cinco melhores resultados obtidos utilizando o algoritmo KNN no experimento DA são apresentados na Tabela 5.15, na ordem da esquerda para a direita, o código do experimento, uso ou não de normalização, instâncias corretas para as classes "normal" e "anomalia", instâncias incorretas para as classes "normal" e "anomalia", taxa de erro por falsos positivos para as classes "normal" e "anomalia", taxa de acurácia e taxa de erro total. O resultado de código 8, destacado em negrito e itálico, mostra a melhor configuração do algoritmo KNN no experimento DA no que se refere a qualidade de classificação.

Tabela 5.15 - Os cinco melhores resultados para o experimento DA para o algoritmo KNN.

KNN (LAZY IBK)										
			INSTÂNCIAS CORRETAS		INSTÂNCIAS INCORRETAS					
COD. EXP.	NORM.	*k*	CLASSE NORMAL (VP)	CLASSE ANOMALIA (VN)	CLASSE NORMAL (FN)	CLASSE ANOMALIA (FP)	TE_{NORMAL}	$TE_{ANOMALIA}$	T ACURÁCIA	TE_{TOTAL}
8	***NÃO***	***1***	***9342***	***8548***	***369***	***4285***	***0,334***	***0,038***	***0,794***	***0,206***
11	SIM	1	9096	8646	615	4187	0,326	0,063	0,787	0,213
10	NÃO	7	9346	8375	365	4458	0,347	0,038	0,786	0,214
7	NÃO	3	9345	8372	366	4461	0,348	0,038	0,786	0,214
31	SIM	5	9032	8545	679	4288	0,334	0,070	0,780	0,220

Outras informações gerais e importantes do experimento estão descritas na Tabela 5.16, onde são apresentados, na ordem da esquerda para a direta, o número do experimento, tempo de construção do modelo em segundos, total de instâncias corretas e instâncias classificadas (corretas/incorretas) de cada experimento.

Tabela 5.16 - Os cinco melhores resultados e o total de instâncias corretas e incorretas pelo tempo de construção do modelo para o algoritmo KNN.

KNN (LAZY IBK)				
COD. EXP.	**TEMPO CONSTRUÇÃO. DO MODELO EM SEGUNDOS**	**DISTÂNCIA**	**TOTAL DE INSTÂNCIAS CORRETAS**	**TOTAIS INSTÂNCIAS INCORRETAS**
8	***5,05***	***EUCLIDIANA***	***17890***	***4654***
11	5,29	EUCLIDIANA	17742	4802
10	5,31	EUCLIDIANA	17721	4823
7	8,43	EUCLIDIANA	17717	4827
31	4,84	EUCLIDIANA	17577	4967

Na Tabela B.1 do Apêndice B são apresentados todos os experimentos DA executados para o algoritmo KNN.

Na Figura 5.4 pode-se observar a taxa de erro por FP dos melhores resultados, mostrando que o experimento de código 8 tem a melhor configuração do algoritmo KNN, obtendo a menor taxa de erro sobre as demais configurações, tanto para a detecção de instâncias “normal” quanto para detecção de “anomalia”.

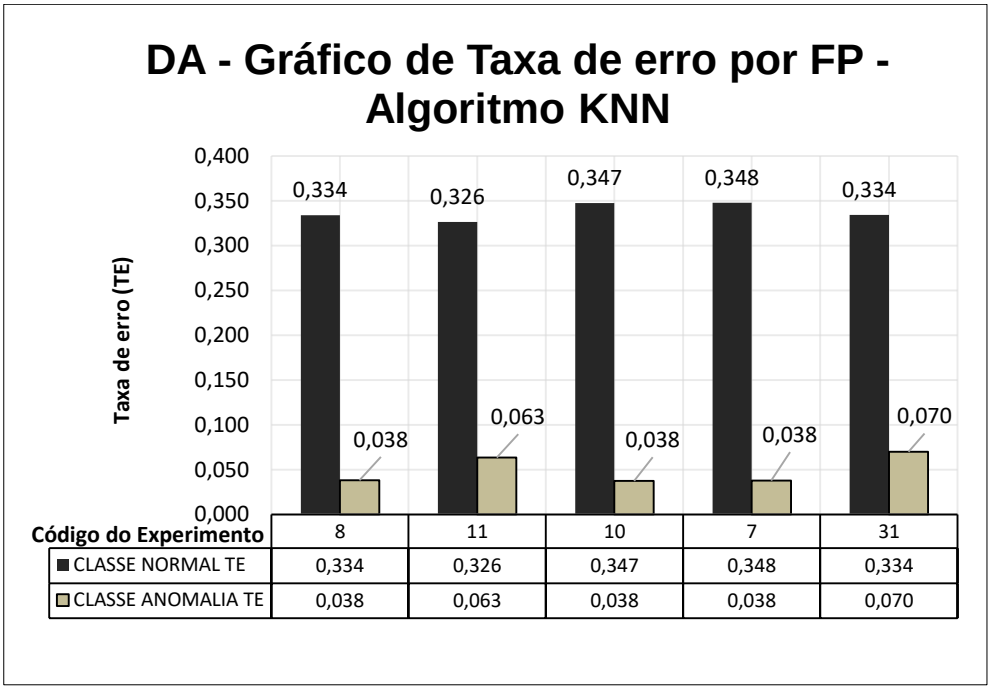


Figura 5.4 - Gráfico com a Taxa de Erro por FP (TE) do algoritmo KNN para os 5 melhores resultados.

Na Figura 5.5 pode-se observar a taxa de acurácia dos 5 melhores resultados do experimento de DA, para o algoritmo KNN. O experimento de número 8 obteve o melhor

índice de acurácia, mostrando que ele tem a melhor configuração para a detecção de anomalias, segundo os critérios utilizados.

Analisando os resultados do teste DA para o algoritmo KNN referente ao desempenho do experimento 8, chegou-se à conclusão pelo seu desempenho de maior acurácia e menor taxa de erro, que o mesmo terá os seus indicadores utilizados no comparativo com o algoritmo K-Means++ e J48, para o mesmo experimento.

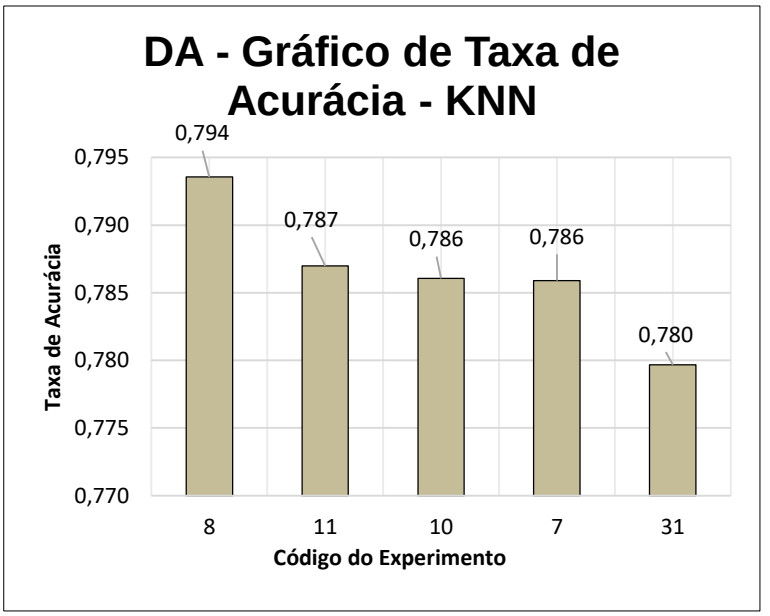


Figura 5.5 - Gráfico com a Taxa de Acurácia do algoritmo KNN.

5.7.2. Algoritmo KNN no Experimento DTA

O primeiro passo para o experimento DTA, no ambiente WEKA, foi processar o algoritmo KNN com o subconjunto de dados "por tipo de ataque - KDDTrain+.arff" como treinamento. Nessa etapa o algoritmo armazenou uma base de dados de treinamento que será utilizada como modelo, aplicado ao conjunto de teste. A aplicação do filtro de normalização e os valores de *k* foram estabelecidos para cada experimento conforme a Tabela 5.7. Após a criação do banco de treinamento foi executado o algoritmo KNN para o subconjunto de dados "por tipo de ataque - KDDTest+.arff".

Os cinco melhores resultados obtidos utilizando o algoritmo KNN no experimento DTA são apresentados na Tabela 5.17, na ordem da esquerda para a direita, o código do experimento, aplicação da normalização no subconjunto de dados, o valor de *k*, quantidade de instâncias processadas, taxa de falsos positivos para a classe "normal", taxa

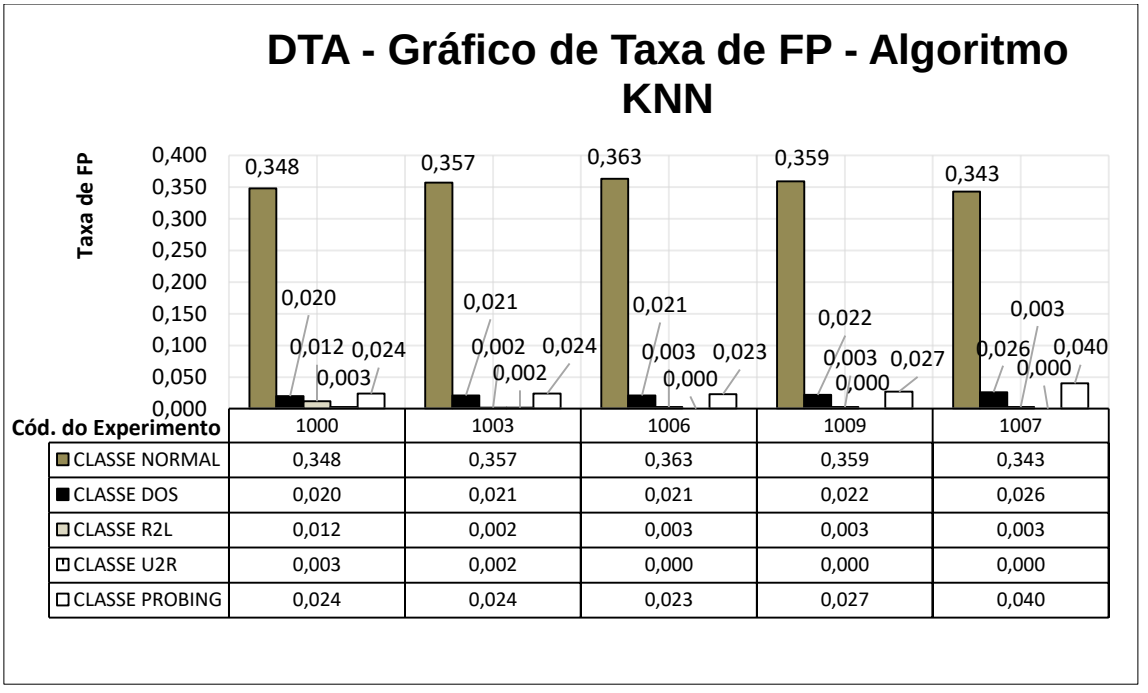
de falsos positivos para a classe “dos”, taxa de falsos positivos para a classe “r2l”, taxa de falsos positivos para a classe “u2r”, taxa de falsos positivos para a classe “probing”, taxa de acurácia, taxa de erro total e a distância utilizada em cada experimento. Na Tabela 5,17 o resultado de código 1000, destacado em negrito e itálico, mostra a melhor configuração, no que se refere a qualidade de classificação, do algoritmo KNN no experimento DTA.

Tabela 5.17 - Os 5 melhores resultados do experimento DTA para o algoritmo KNN.

KNN (LAZY IBK)											
COD. EXP.	**NORM.**	***k***	**QTD. INST.**	**TE_{NORMAL}**	**TE_{DOS}**	**TE_{R2L}**	**TE_{U2R}**	**$TE_{PROBING}$**	**$T_{ACURÁCIA2}$**	**TE_{Total2}**	**DISTÂNCIA**
1000	***NÃO***	***1***	***22544***	***0,348***	***0,020***	***0,012***	***0,003***	***0,024***	***0,771***	***0,229***	***EUCLIDIANA***
1003	NÃO	3	22544	0,357	0,021	0,002	0,002	0,024	0,771	0,229	EUCLIDIANA
1006	NÃO	5	22544	0,363	0,021	0,003	0,000	0,023	0,769	0,231	EUCLIDIANA
1009	NÃO	7	22544	0,359	0,022	0,003	0,000	0,027	0,769	0,231	EUCLIDIANA
1007	SIM	5	22544	0,343	0,026	0,003	0,000	0,040	0,765	0,235	EUCLIDIANA

Na Tabela C.1 do Apêndice C são apresentados todos os experimentos DTA executados para o algoritmo KNN.

Na Figura 5.6 pode-se observar a taxa de erro por falsos positivos nos cinco melhores resultados, mostrando que o experimento de código 1000 tem a melhor configuração do algoritmo KNN, obtendo a menor taxa de erro sobre as demais configurações, para a detecção de instâncias “normal”, “dos”, “r2l”, “u2r” e “probing”.



Cód. do Experimento	1000	1003	1006	1009	1007
CLASSE NORMAL	0,348	0,357	0,363	0,359	0,343
CLASSE DOS	0,020	0,021	0,021	0,022	0,026
CLASSE R2L	0,012	0,002	0,003	0,003	0,003
CLASSE U2R	0,003	0,002	0,000	0,000	0,000
CLASSE PROBING	0,024	0,024	0,023	0,027	0,040

Figura 5.6 - Gráfico com a Taxa de Erro por FP (TE) do algoritmo KNN para os 5 melhores resultados.

Na Figura 5.7 pode-se observar a taxa de acurácia dos cinco melhores resultados do experimento de DTA, para o algoritmo KNN. O experimento de número 1000 obteve o melhor índice de acurácia, mostrando que ele tem a melhor configuração para a detecção das instâncias “normal”, “dos”, “r2l”, “u2r” e “probing”.

Analisando os resultados do experimento DTA para o algoritmo KNN, referente ao desempenho do experimento 1000, chegou-se à conclusão, pelo seu desempenho de maior acurácia e menor taxa de erro, que o mesmo terá os seus indicadores utilizados no comparativo com os algoritmos K-Means++ e J48, para o mesmo experimento.

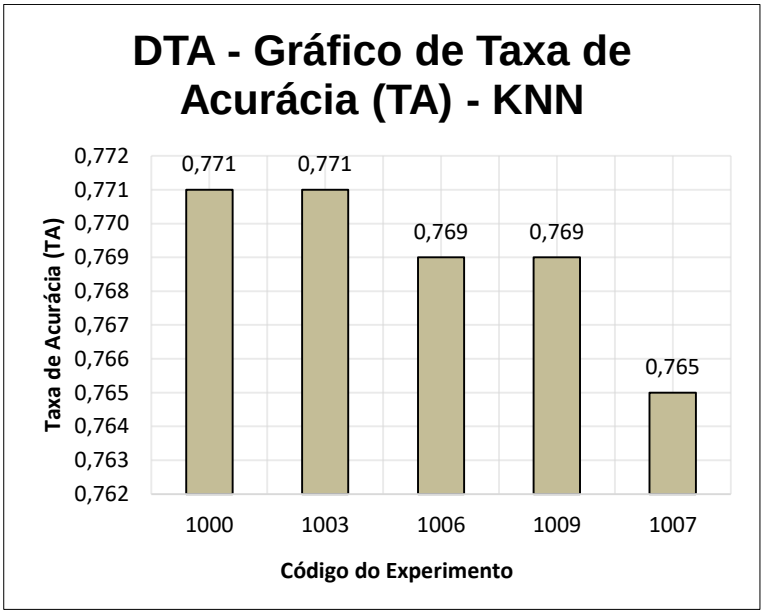


Figura 5.7 - Gráfico com a Taxa de Acurácia do algoritmo KNN.

5.8. Experimento Utilizando o Algoritmo K-Means++

Essa seção trata da execução dos testes de tipo DA e DTA usando o algoritmo K-Means++ no ambiente WEKA, bem como a escolha do melhor resultado para comparações com resultados obtidos com os algoritmos KNN e J48. Para isso serão considerados os resultados a taxa de erro por falsos positivos, a taxa de acurácia e a taxa de erro total.

Para a configuração do algoritmo K-Means++ na ferramenta WEKA foram utilizado os parâmetros *Display Std Devs: False, Dont Replace Missing Values: False,*

MaxInterations: 500, Preserve Instances Order: False, Seed: 10, para os testes DA e DTA.

Os parâmetros de configuração podem ser observados na Figura 5.8, que apresenta a tela de configuração do algoritmo K-Means++ no ambiente WEKA.

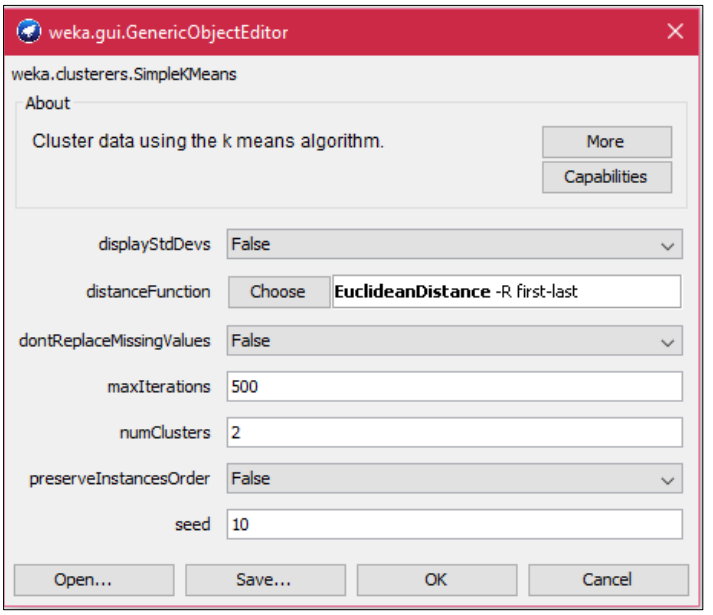


Figura 5.8 – Tela de configuração do algoritmo K-Means++ na ferramenta WEKA.

Além das configurações descritas anteriormente, a configuração do parâmetro relacionado ao número de *clusters (NumCluster)* para o experimento DA é 2, com o intuito de que os dados classificados nos dois agrupamentos reflitam as duas classes, *"*normal*"* e *"*anomalia*"*, do conjunto de dados. Para o experimento DTA, o número de *clusters (NumCluster)* é 5, para que os dados classificados nos cinco agrupamentos reflitam as cinco classes "normal", "dos", "r2l", "u2r" e "probing".

A opção *Distance Function*, que permite selecionar a distância para a medida de similaridade, foi configurada como distâncias Euclidiana ou Manhattan dependendo do experimento.

No ambiente WEKA o algoritmo K-Means++ é encontrado na categoria *Cluster* com o nome de *Simple Kmeans*.

5.8.1. Algoritmo K-Means++ no Experimento DA

O primeiro passo para o experimento DA, no ambiente WEKA, para o algoritmo K-Means++ foi a escolha do subconjunto de dados "KDDTest+.arff". Como o K-

Means++ é de natureza não supervisionada a etapa de treinamento não existe. A aplicação do filtro de normalização e os valores de *k* foram estabelecidos conforme a Tabela 5.8.

Para o experimento DA utilizando o algoritmo K-Means++, os parâmetros de pré-processamento utilizados no ambiente WEKA estão descritos na Tabela 5.14. Os filtros aplicados no subconjunto de dados antes do processamento têm a função de padronizar os dados em busca de melhorias de velocidade e qualidade do processamento, uma vez que cada instância do subconjunto de dados é composta por dados de diversos tipos.

Na Tabela 5.18 são apresentados os experimentos DA executados para o algoritmo K-Means++. São apresentados, na ordem da esquerda para a direita, o código do experimento, valor de *k*, quantidade de instâncias corretas para as classes "normal" e "anomalia", quantidade de instâncias incorretas para as classes "normal" e "anomalia", taxa de erro por falsos positivos das classes "normal" e "anomalia", taxa de acurácia e taxa de erro total. O resultado de código 38 é destacado em negrito e itálico como a melhor configuração para o algoritmo K-Means++ e o resultado geral nos aspectos de qualidade de classificação.

Tabela 5.18 - Os cinco melhores resultados para o experimento DA para o algoritmo K-Means ++.

K-Means++ (Simple K-Means)									
		INSTÂNCIAS CORRETAS		**INSTÂNCIAS INCORRETAS**					
COD. EXP.	***k***	**CLASSE NORMAL (VP)**	**CLASSE ANOMA-LIA (VN)**	**CLASSE NORMAL (FN)**	**CLASSE ANOMAL-IA (FP)**	**TE_{NORMAL}**	**$TE_{ANOMALIA}$**	**T ACURÁCIA**	**TE_{TOTAL}**
38	***2***	***7401***	***10175***	***2658***	***2310***	***0,185***	***0,264***	***0,780***	***0,220***
37	2	7375	10183	2650	2336	0,187	0,264	0,779	0,221
34	2	9477	7018	5815	7018	0,500	0,380	0,562	0,438
35	2	9477	7018	5815	7018	0,500	0,380	0,562	0,438
36	2	9616	5729	7104	5729	0,500	0,425	0,545	0,455

Outras informações do experimento aparecem na Tabela 5.19, na qual são apresentados, na ordem da esquerda para a direta, o número do experimento, tempo de construção do modelo em segundos, normalização por instância, normalização por atributo, medida de similaridade, total de instâncias corretas e instâncias incorretas de cada experimento DA para o algoritmo K-Means++.

Tabela 5.19 - Os cinco melhores resultados, total de instâncias corretas e incorretas pelo tempo de construção do modelo para o algoritmo K-Means++.

K-Means++ (Simple KMeans)						
		NORMALIZAÇÃO				
COD. EXP.	TEMPO CONST. DO MODELO EM SEG.	POR INSTÂNCIA	POR ATRIBUTO	DISTÂNCIA	TOTAL DE INSTÂNCIAS CORRETAS	TOTAIS INSTÂNCIAS INCORRETAS
38	***1,48***	***SIM***	***NÃO***	***MANHATTAN***	***17576***	***4968***
37	1,36	SIM	NÃO	EUCLIDIANA	17558	4986
34	4,02	NÃO	NÃO	EUCLIDIANA	16495	6049
35	3,64	NÃO	SIM	EUCLIDIANA	14495	6049
36	1,00	NÃO	NÃO	EUCLIDIANA	15345	7199

Na Figura 5.9 pode-se observar a taxa de erro por falsos positivos mostrando que o experimento de código 38, com o algoritmo K-Means++, obteve a menor taxa de erro com relação às demais configurações, quanto para a detecção de instância da classe "normal" como para detecção da classe "anomalia".

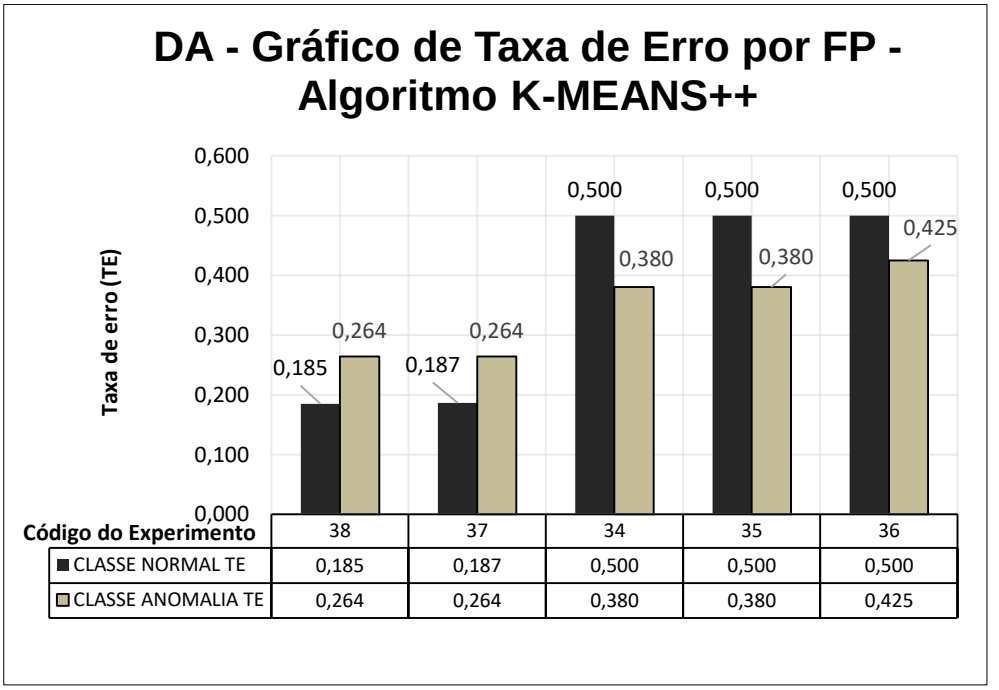


Figura 5.9 -Gráfico que exibe a taxa de erro por falsos positivos (TE) do algoritmo K-Means++ para os 5 melhores resultados.

Na Figura 5.10 pode-se observar a taxa de acurácia dos cinco melhores resultados do experimento de DA para o algoritmo K-Means++. O experimento de número 38 obteve o melhor índice de acurácia, mostrando que ele teve a melhor configuração para a detecção de anomalias.

Analisando os resultados do teste DA para o algoritmo KNN, referente ao desempenho do experimento 38, foi decidido que, devido a seu desempenho de maior acurácia e de ter tido a menor taxa de erro, esta configuração teria os seus indicadores utilizados na comparação com o algoritmo KNN e J48, para o mesmo experimento.

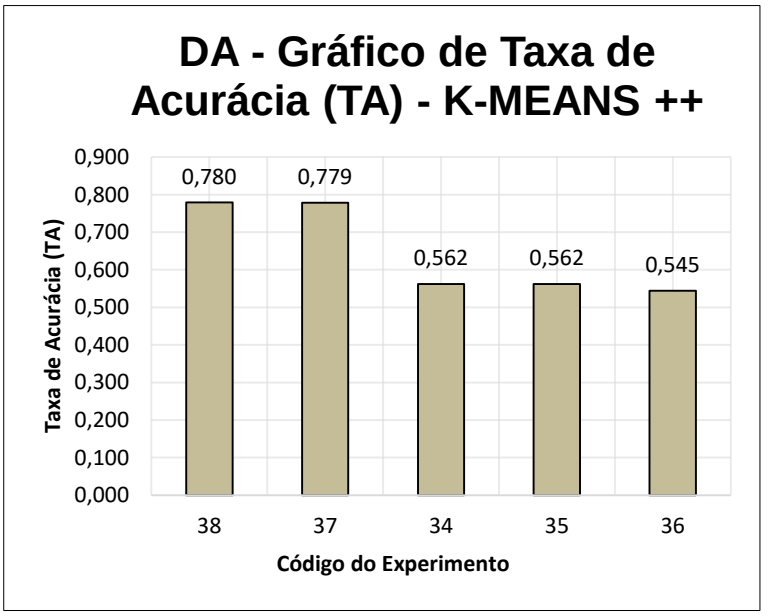


Figura 5.10 - Gráfico da taxa de acurácia do Algoritmo K-Means++.

5.8.2. Algoritmo K-Means++ no Experimento DTA

O primeiro passo para o experimento DTA no ambiente WEKA para o algoritmo K-Means++ foi a escolha do subconjunto de dados "por tipo de ataque - KDDTest+.arff". Como o K-Means++ é de natureza não supervisionada a etapa de treinamento não existe. A aplicação do filtro de normalização e os valores de *k* foram estabelecidos conforme a Tabela 5.8.

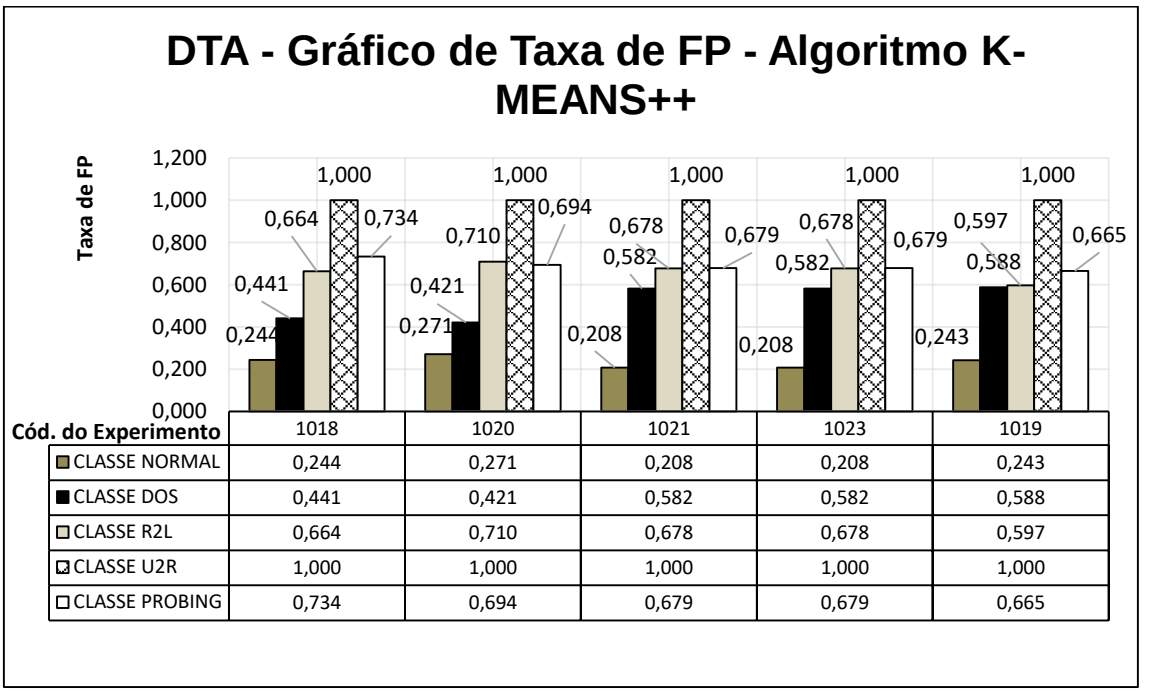
Os 5 melhores resultados de configurações obtidos estão relacionados na Tabela 5.20. Nessa tabela são apresentados, na ordem da esquerda para a direita, o código do experimento, valor de *k,* tipo de normalização, taxa de erro por falsos positivos da classe "normal", taxa de erro por falsos positivos da classe "dos", taxa de erro por falsos positivos da classe 'r2l', taxa de erro por falsos positivos da classe "u2r", taxa de erro por falsos positivos da classe "probing", taxa de acurácia e taxa de erro total. O resultado obtido com o experimento de código 1018 é destacado em negrito e itálico é a melhor configuração para o algoritmo K-Means++.

Tabela 5.20 - Os cinco melhores resultados para o experimento DTA para o algoritmo K-Means ++.

K-Means++ (Simple K-Means)											
COD. EXP.	*k*	NORMALIZAÇÃO		TE_{NORMAL}	TE_{DOS}	TE_{R2L}	TE_{U2R}	$TE_{PROBING}$	$T_{ACURÁCIA2}$	TE_{Tota2l}	DISTÂNCIA
		ATRIB.	INST.								
1018	***5***	***NÃO***	***SIM***	***0,244***	***0,441***	***0,664***	***1,000***	***0,734***	***0,580***	***0,420***	***EUCLIDIANA***
1020	5	NÃO	SIM	0,271	0,421	0,710	1,000	0,694	0,574	0,426	MANHATAN
1021	5	NÃO	NÃO	0,208	0,582	0,678	1,000	0,679	0,553	0,447	MANHATAN
1023	5	SIM	NÃO	0,208	0,582	0,678	1,000	0,679	0,553	0,447	MANHATAN
1019	5	NÃO	NÃO	0,243	0,588	0,597	1,000	0,665	0,548	0,452	EUCLIDIANA

Na Tabela C.2 do Apêndice C são apresentados os resultados de todos os experimentos DTA executados para o algoritmo K-Means++.

Na Figura 5.11 pode-se observar a taxa de erro por falsos positivos dos melhores resultados, mostrando que o experimento de código 1018 tem a melhor configuração do algoritmo KNN, obtendo a menor taxa de erro sobre as demais configurações, para a detecção de instâncias “normal”, “dos”, “r2l”, “u2r” e “probing”.



Cód. do Experimento	1018	1020	1021	1023	1019
CLASSE NORMAL	0,244	0,271	0,208	0,208	0,243
CLASSE DOS	0,441	0,421	0,582	0,582	0,588
CLASSE R2L	0,664	0,710	0,678	0,678	0,597
CLASSE U2R	1,000	1,000	1,000	1,000	1,000
CLASSE PROBING	0,734	0,694	0,679	0,679	0,665

Figura 5.11 - Gráfico que exibe a Taxa de Erro por FP (TE) do algoritmo K-Means++ para os 5 melhores resultados.

Na Figura 5.12 pode-se observar a taxa de acurácia dos cinco melhores resultados do experimento DTA para o algoritmo KNN. O experimento de número 1018 obteve o melhor índice de acurácia, mostrando que ele tem a melhor configuração para a detecção

de instâncias “normal” ou pertencentes a categorias de anomalias, tais como, “dos”, “r2l”, “u2r” e “probing”.

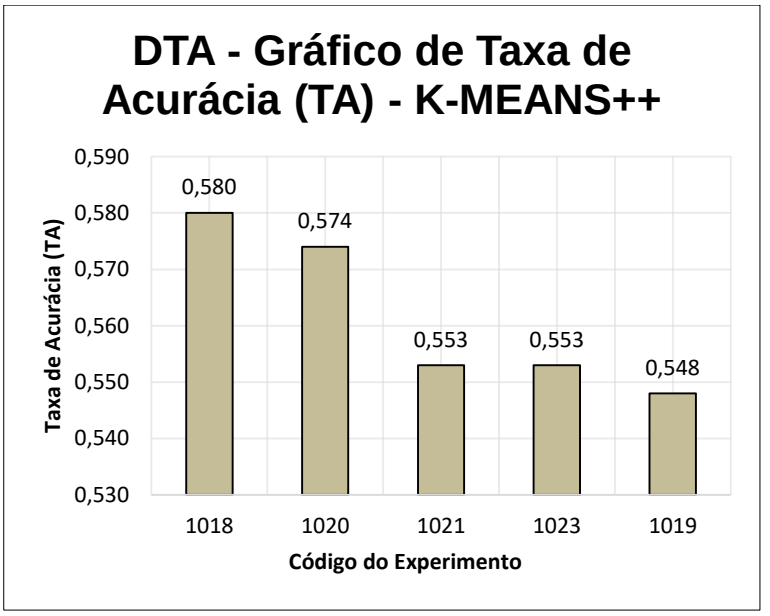


Figura 5.12 - Gráfico com a Taxa de Acurácia do Algoritmo K-Means++.

Analisando os resultados do experimento DTA para o algoritmo K-Means++ referente ao desempenho do experimento 1018, foi decidido que devido a seu desempenho de maior acurácia e menor taxa de erro, teriam seus indicadores utilizados em comparações com os resultados obtidos com os algoritmo KNN e J48 (C4.5), para o mesmo experimento.

5.9. Experimento Utilizando o Algoritmo J48

Essa seção trata da execução dos testes do tipo DA e DTA para o algoritmo J48, no ambiente WEKA, bem como as configurações e a escolha do melhor resultado para o comparativo com o algoritmo K-Means++ e o KNN. Os indicadores como a taxa de erro por falsos positivos, e a taxa de acurácia são utilizados como parâmetros para a escolha do melhor resultado de configuração do algoritmo.

Para o experimento DA e DTA, utilizando o algoritmo J48, os parâmetros de pré-processamento descritos na Tabela 5.9 foram utilizados no ambiente WEKA. Os filtros aplicados no subconjunto de dados antes do processamento têm a função de padronizar o conjunto de dados em busca de melhorias de velocidade e qualidade do processamento,

uma vez que cada instância do subconjunto de dados é composta por valores de vários tipos.

No ambiente WEKA o algoritmo J48 é encontrado na categoria de algoritmos classificadores *Tree* (árvore).

Para a configuração do algoritmo J48 na ferramenta WEKA foram utilizados os parâmetros *BynarySlits: False, ConfidenceFactor: 0.25, Debug: False, MinNumObj: 2, NumFolds: 3, ReducedErrorPruning: False, SaveInstanceData: False, Seed: 1, SubtreeRaising: True, UseLaplace: False,* para os experimentos DA e DTA.

Essas configurações são válidas tanto para a etapa de treinamento como para a etapa de teste, do experimento DA e DTA. Os parâmetros de configuração são apresentados na Figura 5.13 que representa a tela de configuração do algoritmo J48 no ambiente WEKA. O parâmetro relacionado à poda da árvore *(Unpruned)* foi alterado conforme o experimento.

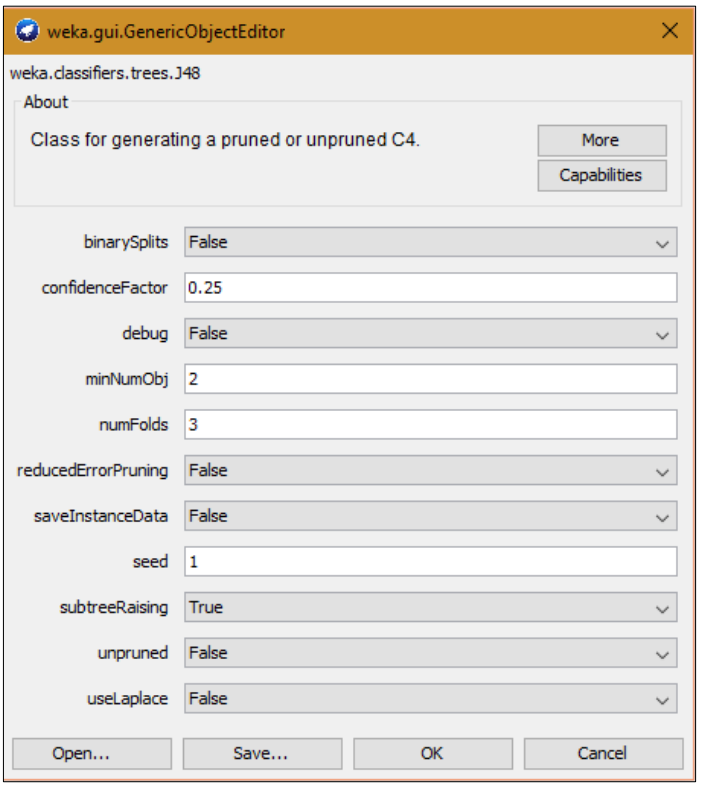


Figura 5.13 - Tela de configurações do WEKA para o algoritmo J48.

5.9.1. Algoritmo J48 no Experimento DA

O primeiro passo para o experimento DA no ambiente WEKA foi executar o J48 com o subconjunto de dados "KDDTrain+.arff" para realizar o treinamento. Nessa etapa o algoritmo se encarrega de construir a árvore que será utilizada como modelo e foi avaliada utilizando o conjunto de teste. A aplicação do filtro de normalização e o parâmetro de poda da árvore foram estabelecidos para cada experimento conforme a Tabela 5.9. Após a criação da árvore (modelo), esta foi utilizada para classificar as instâncias de teste armazenadas no arquivo "KDDTest+.arff".

Os cinco melhores resultados obtidos são apresentados na Tabela 5.21, na ordem da esquerda para a direita, o código do experimento, instâncias corretas para as classes "normal" e "anomalia", instâncias incorretas para as classes "normal" e "anomalia", taxa de erro por falsos positivos das classes "normal" e "anomalia", taxa de acurácia e taxa de erro total. Na Tabela 5.21 é destacado em itálico e negrito o resultado do experimento de código 2 como o melhor resultado de configuração do algoritmo J48, nos aspectos de qualidade de classificação, porém esse experimento não obteve o melhor tempo de processamento para construção do modelo.

Tabela 5.21 - Os cinco melhores resultados para o teste DA para o algoritmo J48.

J48								
	INSTÂNCIAS CORRETAS		INSTÂNCIAS INCORRETAS					
COD. EXP.	CLASSE NORMAL (VP)	CLASSE ANOMA-LIA (VN)	CLASSE NORMAL (FN)	CLASSE ANOMALIA (FP)	TE_{NORMAL}	$TE_{ANOMALIA}$	$T_{ACURÁCIA1}$	TE_{TOTAL}
2	***9448***	***8933***	***263***	***3900***	***0,304***	***0,027***	***0,815***	***0,185***
3	9443	8903	268	3930	0,306	0,028	0,814	0,186
6	1725	11744	7986	1089	0,085	0,822	0,597	0,403
4	722	11335	8989	1498	0,117	0,926	0,535	0,465
14	2858	5437	6853	7396	0,576	0,706	0,368	0,632

Outras informações do experimento são apresentadas na Tabela 5.22 na ordem da esquerda para a direta, o código do experimento, tempo de construção do modelo em segundos, aplicação de poda, total de instâncias corretas e instâncias incorretas de cada experimento.

Tabela 5.22 - Os 5 melhores resultados total de instâncias corretas e incorretas pelo tempo de construção do modelo para o algoritmo J48.

J48						
			NORMALIZAÇÃO			
COD. EXP.	TEMPO CONST. DO MODELO EM SEG.	SEM PODA (UNPRUNED)	POR INSTÂNCIA	POR ATRIBUTO	TOTAL DE INSTÂNCIAS CORRETAS	TOTAIS INSTÂNCIAS INCORRETAS
2	***31,1***	***NÃO***	***NÃO***	***NÃO***	***18381***	***4163***
3	29,42	SIM	NÃO	NÃO	18346	4198
6	31,13	SIM	NÃO	SIM	13469	9075
4	32,83	NÃO	NÃO	SIM	12057	10487
14	32,28	SIM	SIM	NÃO	8295	14249

Na Tabela B.3 do Apêndice B são apresentados todos os experimentos DA executados para o algoritmo J48, e de forma específica,

Na Figura 5.14 pode-se observar a da taxa de erro por falsos positivos, mostrando que o experimento de código 2, com o algoritmo J48, obteve a menor taxa de erro sobre as demais configurações, tanto para a detecção de instância "normal" quanto para detecção de "anomalia".

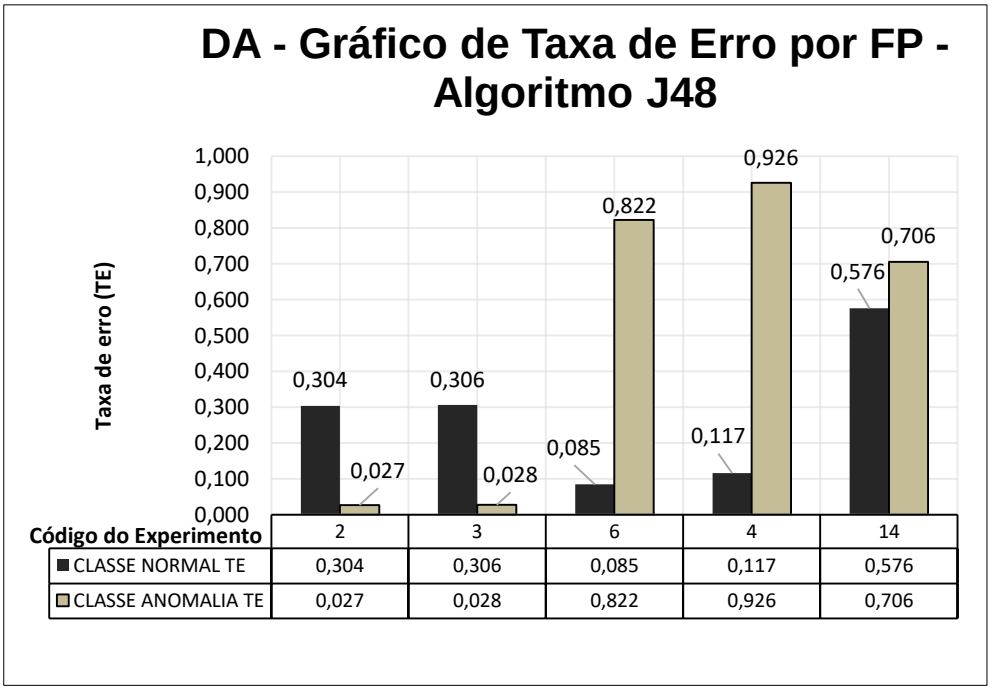


Figura 5.14 - Gráfico que exibe a Taxa de Erro por FP (TE) do algoritmo J48 para os seis melhores resultados.

Na Figura 5.15 pode-se observar a taxa de acurácia dos cinco melhores resultados do experimento de DA para o algoritmo J48. O experimento de número 2 obteve o melhor

índice de acurácia, evidenciando que ele tem a melhor configuração para a detecção de anomalias. Por isso, chegou-se à conclusão que, devido a seu desempenho de maior acurácia e sua menor taxa de erro, esta configuração teria os seus indicadores utilizados nas comparações com os resultados obtidos com o KNN e K-Means++, para o mesmo experimento.

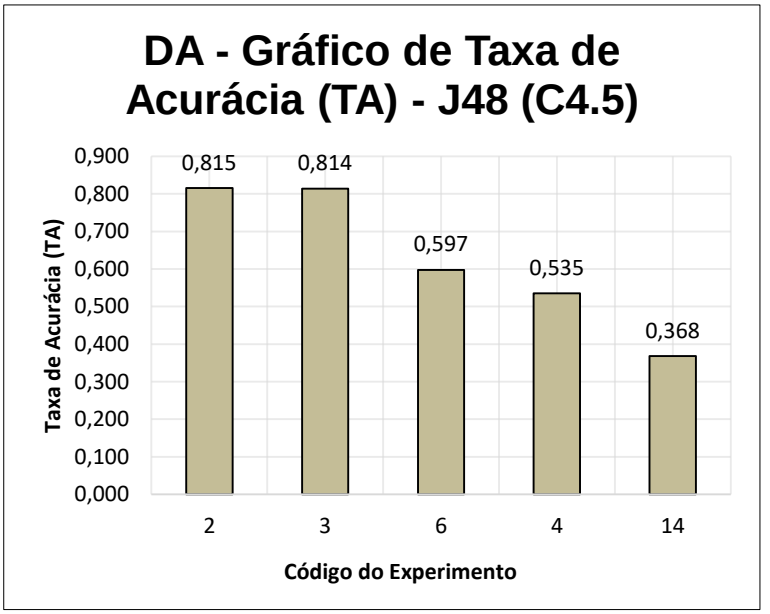


Figura 5.15 - Gráfico com a Taxa de Acurácia do algoritmo J48.

5.9.2. Algoritmo J48 no Experimento DTA

O primeiro passo para o experimento DTA, no ambiente WEKA, foi executar o J48 com o conjunto de treinamento "por tipo de ataque - KDDTrain+.arff". Nessa etapa o algoritmo constrói o modelo que será utilizado para classificar as instâncias de teste. A aplicação do filtro de normalização e os valores de *k* foram estabelecidos conforme a Tabela 5.9, para cada experimento. Após a criação do modelo (árvore), este foi utilizado com as instâncias do conjunto de dados "por tipo de ataque - KDDTest+.arff".

Os resultados do experimento DTA são apresentados na Tabela 5.23, na ordem da esquerda para a direita, o código do experimento, aplicação da normalização no subconjunto de dados, aplicação de poda, taxa de falsos positivos para a classe "normal", taxa de falsos positivos para a classe "dos", taxa de falsos positivos para a classe "r2l", taxa de falsos positivos para a classe "u2r", taxa de falsos positivos para a classe "probing", taxa de acurácia e a taxa de erro total. Nessa tabela, o resultado de código 1015, destacado em negrito e itálico, mostra a melhor configuração, no que se refere a qualidade

de classificação, do algoritmo J48 no experimento DTA. Os cinco melhores resultados obtidos estão relacionados nas Tabela 5.23.

Tabela 5.23 - Os cinco melhores resultados para o teste DTA para o algoritmo J48.

J48										
COD. EXP.	NORMALIZAÇÃO		(SEM PRODA) UNPRUNED	TE_{NORMAL}	TE_{DOS}	TE_{R2L}	TE_{U2R}	$TE_{PROBING}$	$T_{ACURÁCIA2}$	TE_{Total2}
	ATRIB.	INST.								
1015	***SIM***	***NÃO***	***NÃO***	***0,270***	***0,115***	***0,009***	***0,000***	***0,011***	***0,783***	***0,217***
1017	NÃO	NÃO	NÃO	0,383	0,026	0,000	0,000	0,037	0,753	0,247
1016	NÃO	SIM	SIM	0,392	0,025	0,001	0,000	0,048	0,741	0,259
1013	SIM	SIM	SIM	0,063	0,816	0,007	0,000	0,013	0,446	0,554
1012	NÃO	SIM	SIM	0,585	0,600	0,065	0,002	0,505	0,272	0,728

Na Tabela C.3 do Apêndice C são apresentados todos os experimentos DTA executados para o algoritmo J48.

Na Figura 5.16 pode-se observar a taxa de erro por falsos positivos, mostrando que o experimento de código 1015, com o algoritmo J48, obteve a menor taxa de erro sobre as demais configurações, tanto para a detecção de instância "normal" quanto para as categorias de anomalia, tais como, "dos", "r2l", "u2r" e probing.

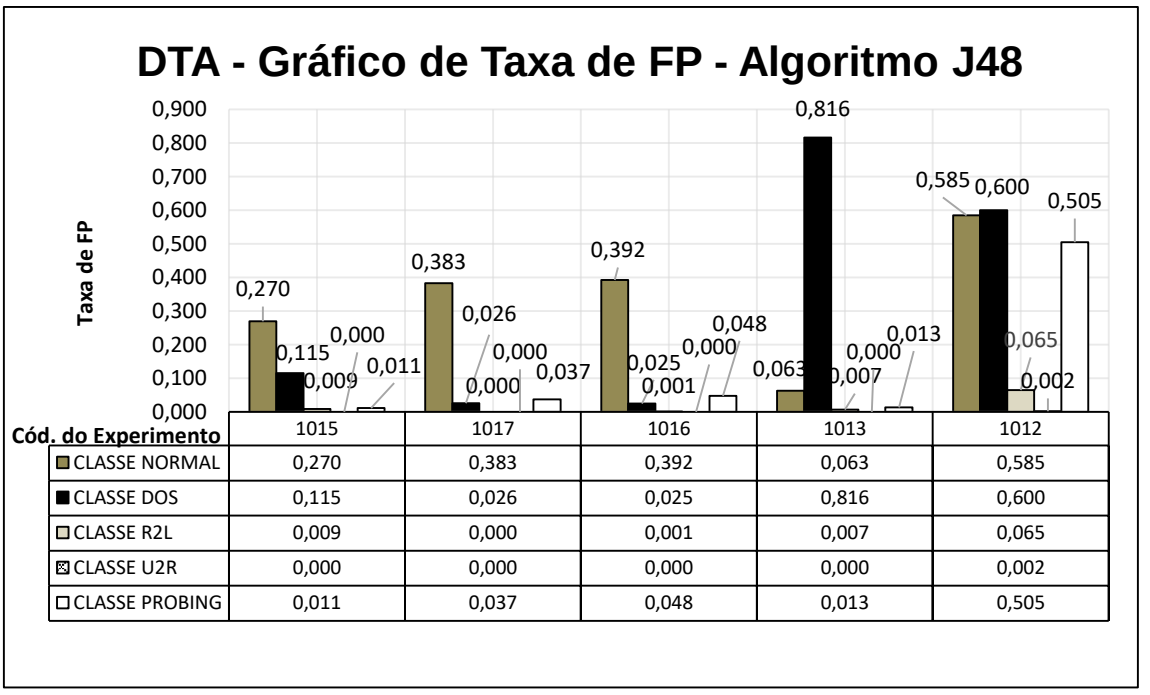


Figura 5.16 - Gráfico que exibe a Taxa de Erro por FP (TE) do algoritmo J48 para os cinco melhores resultados.

Na Figura 5.17 pode-se observar a taxa de acurácia dos cinco melhores resultados do experimento DTA para o algoritmo J48. O experimento de código 1015 obteve o

melhor índice de acurácia, evidenciando que tem a melhor configuração para a detecção de anomalias. Por isso, chegou-se à conclusão que, devido ao seu desempenho de maior acurácia e menor taxa de erro, a configuração teria os seus indicadores utilizados nas comparações com os resultados obtidos com os algoritmos KNN e K-Means++, para o mesmo experimento.

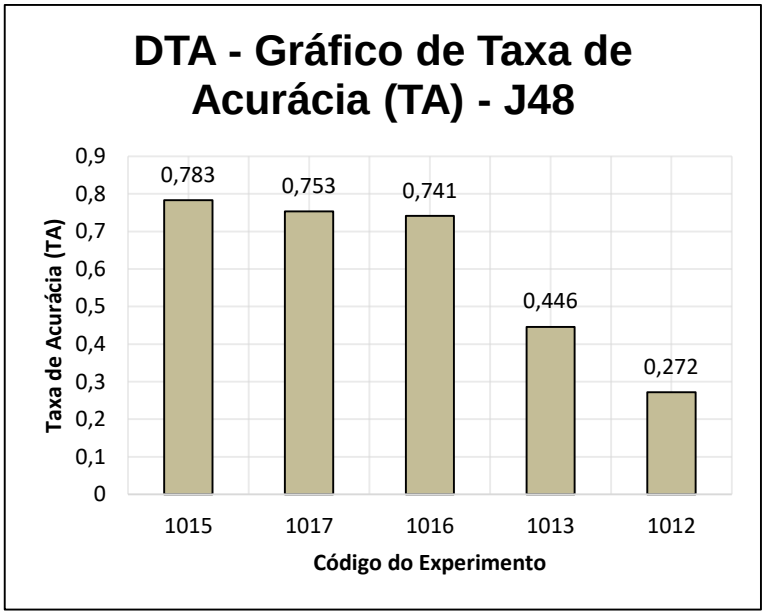


Figura 5.17 - Gráfico com a Taxa de Acurácia do algoritmo J48.

Capítulo 6
Análises, Conclusões e Trabalhos Futuros

Neste capítulo inicialmente são apresentadas algumas análises dos resultados dos experimentos DA e DTA, os objetivos e contribuições da pesquisa realizada, através do resumo dos pontos investigados. Por fim, são apresentados alguns possíveis trabalhos futuros, que podem ser iniciados em continuidade à pesquisa desenvolvida e descrita nesta dissertação.

6.1. Análise dos resultados do experimento DA

Para o experimento DA a análise dos resultados foi feita individualmente, por algoritmo, observando os aspectos dos falsos positivos. Para isso foram usadas as TE_{NORMAL} e $TE_{ANOMALIA,}$ para verificar o desempenho por classe "normal" e "anomalia". O segundo ponto da análise está relacionado ao melhor desempenho de TE_{NORMAL} e $TE_{ANOMALIA1}$ entre os algoritmos (KNN, K-Means++ e J48), para apontar os pontos fortes de cada algoritmo, em relação aos FP. Foram utilizados os resultados das $T_{ACURÁCIA}$ e TE_{TOTAL} para apontar, entre eles, o que tem melhor desempenho geral na detecção de instâncias de classe "normal" ou "anomalia".

6.1.1. Análise do algoritmo KNN

No experimento DA, de código 8 obtiveram-se valores de 0,334 para a TE_{NORMAL} e de 0,038 para a $TE_{ANOMALIA}$. Na configuração escolhida é possível constatar que o algoritmo KNN consegue classificar de maneira significativa as instâncias que pertencem a classe "anomalia". Essa evidencia é constatada pela menor taxa de erro.

6.1.2. Análise do Algoritmo K-Means++

No experimento de DA, de código 38 obteve-se valores de 0,185 para a TE_{NORMAL} e de 0,264 para a $TE_{ANOMALIA}$. Na configuração escolhida é possível constatar que o

algoritmo K-Means++ conseguiu classificar de maneira consistente as instâncias que pertencem a classe “normal”. Essa evidencia é constatada pela menor taxa de erro.

6.1.3. Análise Algoritmo J48

No experimento de DA, de código 2 obteve-se valores de 0,264 para a TE_{NORMAL} e de 0,027 para a $TE_{ANOMALIA}$. Na configuração escolhida é possível constatar que o algoritmo J48 consegue classificar de maneira consistente as instâncias que pertencem a classe de “anomalia”. Essa evidencia é constatada pela menor taxa de erro.

6.1.4. Análise Geral da TE_{NORMAL} e $TE_{ANOMALIA1}$

É possível constatar através dos indicadores de TE_{NORMAL} e $TE_{ANOMALIA}$, descritos na Tabela 6.1, que o algoritmo K-Means++ consegue classificar de maneira significativa as instâncias pertencentes a classe “normal”. Já o algoritmo J48, consegue classificar de maneira considerável as instâncias pertencentes a classe de “anomalia”. Isso é evidenciado pela menor taxa de erro.

Tabela 6.1 – Comparativo de TE_{NORMAL} e $TE_{ANOMALIA1}$ para os algoritmos KNN, K-Means++ e J48.

Algoritmo	TE_{NORMAL}	$TE_{ANOMALIA}$
KNN	0,334	0,038
K-Means++	0,185	0,264
J48	0,304	0,027

6.1.5. Análise das $T_{ACURÁCIA1}$ e TE_{TOTAL} do Experimento DA

Observa-se na Tabela 6.2 a consolidação dos indicadores com valores significativos no experimento DA, para os algoritmos KNN, K-Means++ e J48, e na Tabela 6.3 os dados relacionados aos indicadores com os melhores tempos de construção dos modelos, as medidas de distância utilizadas e totais de instâncias corretas e incorretas.

Com base nos valores dos indicadores do comparativo ($T_{ACURÁCIA1}$ e TE_{TOTAL}), o algoritmo que obteve os melhores indicadores foi o J48, o segundo melhor resultado foi obtido através do algoritmo KNN e o terceiro melhor indicador foi o algoritmo K-Means++, de natureza não supervisionada, que não ficou distante dos dois primeiros

algoritmos, conseguindo discriminar adequadamente as duas classes “normal” e “anomalia”.

Nota-se que na classificação das instâncias da classe “anomalia”, para ser detectada, são necessários todos os atributos do conjunto de dados, tendo em vista que, dentro da classe de “anomalias”, existem diferentes categorias de ataque (“dos”, “r2l”, “u2r” e “probing”), e que torna sua detecção complexa.

Tabela 6.2 – Consolidação dos resultados do comparativo entre os algoritmos KNN, K-Means++ e J48.

COMPARATIVO ENTRE OS ALGORITMOS KNN, K-Means++ e J48					
ALG.	***k***	**TE_{NORMAL}**	**$TE_{ANOMALIA}$**	**$T_{ACURÁCIA1}$**	**TE_{TOTAL}**
J48	***-***	***0,304***	***0,027***	***0,815***	***0,185***
KNN	1	0,334	0,038	0,794	0,206
KMeans++	2	0,185	0,264	0,780	0,220

Tabela 6.3 – Complemento da consolidação dos resultados do comparativo entre os algoritmos KNN, K-Means++ e J48.

COMPARATIVO ENTRE OS ALGORITMOS KNN, K-Means++ e J48				
		NORMALIZAÇÃO		
ALG.	**TEMPO CONST. DO MODELO EM SEG.**	**POR INST.**	**POR ATRIB.**	**MEDIDA SIMILARIDADE**
J48	***31,1***	***NÃO***	***NÃO***	**-**
KNN	5,05	-	NÃO	EUCLIDIANA
KMeans++	1,48	SIM	NÃO	MANHATTAN

6.2. Análise dos resultados do experimento DTA

Para o experimento DTA a análise dos resultados foi feita individualmente por algoritmo, observando os FP, para isso foram usadas as $TE_{NORMAL,}$ $TE_{DOS,}$ $TE_{R2L,}$ TE_{U2R} e $TE_{PROBING}$ para verificar o desempenho da classificação por classes “normal”, “dos”, “r2l”, “u2r” e “probing”. O segundo ponto da análise realizada está relacionado ao melhor desempenho das $TE_{NORMAL,}$ $TE_{DOS,}$ $TE_{R2L,}$ $TE_{U2R,}$ $TE_{PROBING}$, entre os algoritmos KNN, K-Means++ e J48, para apontar os pontos fortes de cada algoritmo, em relação aos FP. Foram utilizados os resultados das $T_{ACURÁCIA2}$ e TE_{TOTAL2}, para apontar o algoritmo que

com melhor desempenho geral na detecção de instâncias pertencentes as classes “normal” e “dos”, “r2l”, “u2r” e “probing”.

6.2.1. Análise do algoritmo KNN

No experimento de DTA de código 1000 obtiveram-se valores de 0,348 para a TE_{NORMAL}, 0,020 para a TE_{DOS}, 0,012 para a TE_{R2L}, 0,003 para a TE_{U2R} e de 0,024 para a $TE_{PROBING}$. Na configuração escolhida é possível constatar que o algoritmo KNN consegue classificar de maneira significativa as instâncias que pertencem a classe “u2r”. Essa evidencia é constatada pela menor taxa de erro.

6.2.2. Análise do Algoritmo K-Means++

No experimento de DTA de código 1018 obtiveram-se valores de 0,244 para a TE_{NORMAL}, 0,441 para TE_{DOS}, 0,664 para a TE_{R2L}, 1,00 para a TE_{U2R} e de 0,734 para a $TE_{PROBING}$. Na configuração escolhida é possível constatar que o algoritmo K-Means++ consegue classificar de maneira considerável as instâncias que pertencem a classe “normal”. Essa evidencia é constatada pela menor taxa de erro.

6.2.3. Análise Algoritmo J48

No experimento de DTA de código 1015 obtiveram-se valores de 0,270 para a TE_{NORMAL}, 0,115 para a TE_{DOS}, 0,009 para a TE_{R2L}, 0,000 para a TE_{U2R} e 0,011 para $TE_{PROBING}$. Na configuração escolhida é possível constatar que o algoritmo J48 consegue classificar de maneira considerável as instâncias que pertencem a classe “u2r”. Essa evidencia é constatada pela menor taxa de erro.

6.2.4. Análise Geral da TE_{NORMAL}, TE_{DOSL}, TE_{R2L}, TE_{U2R} e $TE_{PROBING}$

É possível constatar através dos indicadores de TE_{NORMAL}, TE_{DOS}, TE_{R2L}, TE_{U2R} e $TE_{PROBING}$, descritos na Tabela 6.4 que o algoritmo K-Means++ conseguiu classificar de maneira considerável as instâncias pertencentes a classe “normal”. O algoritmo KNN, conseguiu classificar de maneira significativa as instâncias pertencentes a classe de “dos”. Já o algoritmo J48 conseguiu classificar de maneira significativa as instâncias pertencentes a classe “r2l”, “u2r” e “probing”.

Tabela 6.4 – Comparativo de TE_{NORMAL}, TE_{DOS} TE_{R2L}, TE_{U2R} e $TE_{PROBING}$, para os algoritmos KNN, K-Means++ e J48.

Algoritmo	TE_{NORMAL}	TE_{DOS}	TE_{R2L}	TE_{U2R}	$TE_{PROBING}$
KNN	0,348	0,020	0,012	0,003	0,024
K-Means++	0,244	0,441	0,664	1,000	0,734
J48	0,270	0,115	0,009	0,000	0,011

6.2.5. Análise das $T_{ACURÁCIA2}$ e TE_{TOTAL} do Experimento DTA

Observa-se na Tabela 6.5 a consolidação dos indicadores com valores no experimento DTA, para os algoritmos KNN, K-Means++ e J48.

Com base nos valores dos indicadores que dizem a respeito dos valores de $T_{ACURÁCIA2}$ e TE_{TOTAL2}, o algoritmo que obteve os melhores indicadores foi o J48, o segundo melhor resultado foi obtido através do algoritmo KNN e o terceiro melhor resultado foi do algoritmo K-Means++. É importante salientar que a análise do desempenho, por meio dos indicadores gerais, não reflete os melhores valores dos indicadores encontrados por categorias de anomalias.

Tabela 6.5 – Indicadores das $T_{ACURÁCIA2}$ TE_{TOTAL2}, para os algoritmos KNN, K-Means++ e J48.

Algoritmo	$TE_{ACURÁCIA2}$	TE_{TOTAL2}
J48	0,783	0,217
KNN	0,771	0,229
K-Means++	0,580	0,420

6.3. Conclusões

O trabalho de pesquisa realizado teve como objetivos investigar: (1) a viabilidade de técnicas de mineração de dados aplicadas à detecção de intrusão. (2) realização de experimentos com intuito de comparar o desempenho dos algoritmos KNN, K-Means++ e J48, na tarefa de detectar intrusões, identificando instâncias de ocorrências de eventos como “normal” ou “anomalia” no experimento DA, e “normal”, “dos”, “r2l”, “u2r” e “probing”, no experimento DTA.

Os experimentos foram realizados utilizando o ambiente WEKA, apresentado na Seção 4.3, que permitiu a aplicação do pré-processamento dos conjuntos de dados, configuração e execução dos algoritmos, bem como a consolidação dos resultados em forma de indicadores, através de matrizes de confusão.

Nas seções 5.5.1, 5.6.1 e 5.7.1 foram apresentados os experimentos de DA, onde se investigou o desempenho dos algoritmos KNN, K-Means++ e J48, na identificação de instâncias pertencentes à classe “normal” ou “anomalia”. Os resultados dos experimentos DA sugerem algumas conclusões:

(1) Os algoritmos supervisionados (KNN e J48) tendem a classificar de maneira mais adequadas os dados. Porém os resultados do algoritmo K-Means++, que não é supervisionado, é promissor, tendo em vista a técnica que utiliza para realizar a classificação.

(2) O algoritmo J48 conseguiu obter o melhor desempenho no que diz respeito a acurácia dos dados e taxa de erro geral, o KNN o segundo melhor desempenho e o K-Means++ o terceiro melhor.

(3) No aspecto da $TE_{NORMAL,}$ o algoritmo K-Means++ obteve a menor taxa de erro por FP, viabilizando o seu uso, na etapa de sensoriamento de um IDS, podendo detectar de forma adequada os acessos normais em detrimento das anomalias.

(4) No aspecto da $TE_{ANOMALIA}$ o algoritmo J48 obteve a menor taxa de erro por FP, viabilizando o seu uso na etapa de sensoriamento de um IDS, podendo detectar corretamente os acessos anômalos, em detrimento aos normais.

Nas seções 5.5.2, 5.6.2 e 5.7.2 foram apresentados os experimentos de DTA, onde se investigou o desempenho dos algoritmos KNN, K-Means++ e J48, na identificação de instâncias pertencentes as classes “normal”, “dos”, “r2l”, “u2r” e “probing”. Os resultados dos experimentos DTA sugerem algumas conclusões:

(1) Os algoritmos supervisionados (KNN e J48) também conseguiram classificar de maneira mais adequadas os dados. Nesse caso o algoritmo K-Means++, teve seu desempenho aquém do experimento DA.

(2) O algoritmo J48 conseguiu obter o melhor desempenho no que diz respeito à acurácia dos dados e taxa de erro geral; o KNN apresentou o segundo melhor desempenho e o K-Means++ o terceiro. Esse último ficou distante dos dois primeiros colocados.

(3) No aspecto da $TE_{NORMAL,}$ o algoritmo K-Means++ obteve a menor taxa de erro por FP, podendo detectar, de forma adequada, os acessos normais. No aspecto da TE_{DOS} o algoritmo KNN obteve a menor taxa de erro por FP, podendo detectar de forma adequada as anomalias da categoria "dos". Nos aspectos TE_{R2L}, TE_{U2R} e $TE_{PROBING}$ o algoritmo J48 obteve a menor taxa de erro por FP, podendo detectar de forma adequada os acessos anômalos pertencentes às categorias de "r2l", "u2r" e "probing". Apesar do desempenho geral do algoritmo K-Means++ se apresentar inferior aos outros, na análise dos indicadores de FP, por categoria, ele se mostrou promissor para a classificação de instâncias como "normal".

(4) Após a execução dos experimentos de DTA, com a obtenção dos resultados resumidos acima, foi possível validar os conjuntos de dados criados (por_tipo_de_ataque-KDDTrain+.arff e por_tipo_de_ataque-KDDTest+.arff), separados por categorias de ataque, que poderá ser disponibilizado para a comunidade científica.

(5) Um ambiente de sensoriamento de um IDS, composto por algoritmos de forma hibrida pode se mostrar adequado, no que diz respeito à detecção dos vários tipos de anomalias, através da qualidade de cada algoritmo testado.

Através dos experimentos DA e DTA é possível afirmar que o uso de técnicas de mineração de dados, a partir dos algoritmos estudados, é viável na detecção de intrusões, identificando acessos normais de anômalos.

6.4. Trabalhos Futuros

Nesse estudo foram realizados diversos experimentos que puderam consolidar o uso de técnicas de mineração de dados na detecção de intrusões em rede de computadores.

Alguns possíveis trabalhos que podem ser iniciados dando continuidade a esta pesquisa são:

1) Investigar os dados gerados nos experimentos de DA e DTA, e realizar uma análise com foco na Taxa de Verdadeiros Positivos. Essa análise poderá relatar o comportamento dos algoritmos, KNN, K-Means++ e J48, na classificação de instâncias consideradas como um acesso normal.
2) Estudar outros algoritmos tais como, SVM, Redes Neurais, DBScan e Cobweb, e consolidar o conhecimento adquirido para o desenvolvimento de um IDS que possa vir a utilizar, de forma colaborativa, os algoritmos analisados.
3) Investigar técnicas de detecção de intrusões que levem em consideração as séries temporais do conjunto de dados NSL-KDD. Esse tipo de análise ajudará a obter mais precisão na detecção da categoria de intrusão “u2r”, que correspondem aos usuários com permissões básicas se passando por usuários *root*.

Referências

ACM - Association for Computing Machinery, n.d. KDD Cup 1999: Computer network intrusion detection. [Online] Available at: http://www.sigkdd.org/kdd-cup-1999-computer-network-intrusion-detection [Accessed 27 07 2015].

Araar, A. & Haddad, A., 2015. Identification of New Connections for IP Intrusion Detections using WEKA Platform and KDD Cup 99. Journal of Emerging Trends in Computing and Information Sciences, pp. 7-14.

Arthur, D. & Vassilvitskii, S., 2007. k-means++: The Advantages of Careful Seeding. SODA '07 Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms', pp. 1027-1035 .

Bottou, L. & Bengio, Y., 1995. Convergence Properties of the K-Means Algorithms. Advances in Neural Information Processing Systems, Issue 7, pp. 585-592.

Camilo, C. O. & da Silva, J. C., 2010. Uma Metodologia para Mineração de Regras de Associação Usando Ontologias para Integração de Dados Estruturados e Não-Estruturados [dissertação]. Goiania: UFG.

Dai, W. & Ji, W., 2014. A MapReduce Implementation of C4.5 Decision Tree Algorithm. International Journal of Database Theory and Application, pp. 49-60.

Fayyad, U., Haussler, D. & Stolorz, P., 1996. KDD for Science Data Analysis: Issues and Examples. KDD-96 Proceedings, pp. 51-56.

Han, L. & Kamber, M., 2006. Data Mining Concepts And Techniques. 2ª ed. São Francisco : Morgan Kaufmann & Elsevier .

Hart, P. E. & Cover, T. M., 1967. Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, pp. 21-27.

Jones, A. K. & Sielken, R. S., 2000. Computer System Intrusion Detection: A Survey [Tecnical Report], Charlottesville: s.n.

Lee, W. & Stolfo, S. J., 1998. Data Mining Approaches for Intrusion Detection [Tecnical Report]. Texas, USENIX, pp. 79-94.

Lincoln Laboratory Massachusets Institute of Technology, n.d. CYBER SYSTEMS AND TECHNOLOGY. [Online] Available at: http://www.ll.mit.edu/ideval/data/ [Accessed 27 07 2015].

Linden, R., 2009. Técnicas de Agrupamento. Revista de Sistemas de Informação da FSMA, N°. 4, pp. 18-36.

Quinlan, J. R., 1993. C4.5: Programs for Machine Learning. San Mateo: Morgan Kaufmann Publisher.

Real, E. M., 2014. Investigação de Algoritmos Sequenciais de Agrupamento com Pré-processamento de Dados em Aprendizado de Máquina [dissertação]. Campo Limpo Paulista: PMCC-FACCAMP.

Silva, L. M. O. d., 2005. Uma Aplicação de Árvores de Decisão, Redes Neurais e Knn para a Identificação de Modelos Arma Não-Sazonais e Sazonais [dissertação]. Rio de Janeiro (RJ): Pontifícia Universidade Católica do Rio de Janeiro - Puc-Rio.

Singh, S. & Bansal, M., 2013. A Survey on Intrusion Detection System in Data Mining. International Journal of Advanced Research in Computer Engineering & Technology, Junho, pp. 2190-2194.

Stolfo, S. J. *et al.*, n.d. Intrusion Detector Learning. [Online] disponível em: https://kdd.ics.uci.edu/databases/kddcup99/task.html [Accessado 28 2016 2016].

Tavallaee, M., Bagheri, E., Lu, W. & Ghorbani, A. A., 2009. A Detailed Analysis of the KDD CUP 99 Data Set. Otawa, Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009, pp. 53-58.

Theodoridis, S. & Koutroumbas, K., 2009. Pattern Recognition. 4 ed. San Diego: Elsevier.

Vaccaro, H. S. & Liepins, G., 1989. Detection of Anomalous Computer Session Activity. pp. 280-289.

Webb, A. R. & Copsey, K. D., 2011. Statistical Pattern Recognition. 3 ed. Chichester: John Wiley & Sons, Ltd.

Wikipédia, a enciclopédia livre, n.d. Geometria do táxi. [Online] Available at: http://pt.wikipedia.org/wiki/Geometria_do_táxi [Accessed 26 04 2015].

Witten, I. H., Frank, E. & Hall, M. A., 2011. Data Mining Pratical Machine Learning Tools And Techniques. 3° ed. Burlington: Morgam Kaufman & Elsevier.

Wu, X. *et al.*, 2007. Top 10 algorithms in data mining. Knowledge and Information Systems, 04 12, Volume 14, pp. 1-37.

Apêndice A

Conjunto de Dados do Experimento DTA

A Tabela A.1 tem informações a respeito da configuração do cabeçalho que foi desenvolvido para o conjunto de dados de treinamento (por_tipo_de_ataque-KDDTrain+.arff) que tem instâncias separadas por tipo de ataque.

Tabela A.1 - Cabeçalho do subconjunto de dados por_tipo_de_ataque-KDDTrain+.arff

@relation “KDDTrainCategory'
@attribute 'duration' real
@attribute 'protocol_type' {'tcp','udp', 'icmp'}
@attribute 'service' {'aol', 'auth', 'bgp', 'courier', 'csnet_ns', 'ctf', 'daytime', 'discard', 'domain', 'domain_u', 'echo', 'eco_i', 'ecr_i', 'efs', 'exec', 'finger', 'ftp', 'ftp_data', 'gopher', 'harvest', 'hostnames', 'http', 'http_2784', 'http_443', 'http_8001', 'imap4', 'IRC', 'iso_tsap', 'klogin', 'kshell', 'ldap', 'link', 'login', 'mtp', 'name', 'netbios_dgm', 'netbios_ns', 'netbios_ssn', 'netstat', 'nnsp', 'nntp', 'ntp_u', 'other', 'pm_dump', 'pop_2', 'pop_3', 'printer', 'private', 'red_i', 'remote_job', 'rje', 'shell', 'smtp', 'sql_net', 'ssh', 'sunrpc', 'supdup', 'systat', 'telnet', 'tftp_u', 'tim_i', 'time', 'urh_i', 'urp_i', 'uucp', 'uucp_path', 'vmnet', 'whois', 'X11', 'Z39_50'}
@attribute 'flag' { 'OTH', 'REJ', 'RSTO', 'RSTOS0', 'RSTR', 'S0', 'S1', 'S2', 'S3', 'SF', 'SH' }
@attribute 'src_bytes' real
@attribute 'dst_bytes' real
@attribute 'land' {'0', '1'}
@attribute 'wrong_fragment' real

@attribute 'urgent' real

@attribute 'hot' real

@attribute 'num_failed_logins' real

@attribute 'logged_in' {'0', '1'}

@attribute 'num_compromised' real

@attribute 'root_shell' real

@attribute 'su_attempted' real

@attribute 'num_root' real

@attribute 'num_file_creations' real

@attribute 'num_shells' real

@attribute 'num_access_files' real

@attribute 'num_outbound_cmds' real

@attribute 'is_host_login' {'0', '1'}

@attribute 'is_guest_login' {'0', '1'}

@attribute 'count' real

@attribute 'srv_count' real

@attribute 'serror_rate' real

@attribute 'srv_serror_rate' real

@attribute 'rerror_rate' real

@attribute 'srv_rerror_rate' real

@attribute 'same_srv_rate' real

@attribute 'diff_srv_rate' real

@attribute 'srv_diff_host_rate' real

@attribute 'dst_host_count' real

@attribute 'dst_host_srv_count' real

@attribute 'dst_host_same_srv_rate' real

@attribute 'dst_host_diff_srv_rate' real

@attribute 'dst_host_same_src_port_rate' real

@attribute 'dst_host_srv_diff_host_rate' real

@attribute 'dst_host_serror_rate' real

@attribute 'dst_host_srv_serror_rate' real

@attribute 'dst_host_rerror_rate' real

@attribute 'dst_host_srv_rerror_rate' real

@attribute 'classatack' {'normal','apache2','back','land','mailbomb','neptune','pod','processtable','smurf','teardrop','udpstorm','buffer_overflow','httptunnel','loadmodule','perl','ps','rootkit','sqlattack','xterm','ftp_write','guess_passwd','imap','ipsweep','mscan','multihop','named','phf','sendmail','snmpgetattack','snmpguess','spy','warezclient','warezmaster','worm','xlock','xsnoop','tune','scan','nmap','portsweep','saint','satan'}

@attribute 'classcategory' {'normal','dos','r2l','u2r','probing'}

@attribute 'dificult' real

@data

A Tabela A.2 tem informações a respeito da configuração do cabeçalho que foi desenvolvido para o conjunto de dados de teste (por_tipo_de_ataque-KDDTest+.arff) que tem instâncias separadas por tipo de ataque.

Tabela A.2 - Cabeçalho do subconjunto de dados por_tipo_de_ataque-KDDTest+.arff

@relation 'KDDTestCategory'

@attribute 'duration' real

@attribute 'protocol_type' {'tcp','udp', 'icmp'}

@attribute 'service' {'aol', 'auth', 'bgp', 'courier', 'csnet_ns', 'ctf', 'daytime', 'discard', 'domain', 'domain_u', 'echo', 'eco_i', 'ecr_i', 'efs', 'exec', 'finger', 'ftp', 'ftp_data', 'gopher',

'harvest', 'hostnames', 'http', 'http_2784', 'http_443', 'http_8001', 'imap4', 'IRC', 'iso_tsap', 'klogin', 'kshell', 'ldap', 'link', 'login', 'mtp', 'name', 'netbios_dgm', 'netbios_ns', 'netbios_ssn', 'netstat', 'nnsp', 'nntp', 'ntp_u', 'other', 'pm_dump', 'pop_2', 'pop_3', 'printer', 'private', 'red_i', 'remote_job', 'rje', 'shell', 'smtp', 'sql_net', 'ssh', 'sunrpc', 'supdup', 'systat', 'telnet', 'tftp_u', 'tim_i', 'time', 'urh_i', 'urp_i', 'uucp', 'uucp_path', 'vmnet', 'whois', 'X11', 'Z39_50'}

@attribute 'flag' { 'OTH', 'REJ', 'RSTO', 'RSTOS0', 'RSTR', 'S0', 'S1', 'S2', 'S3', 'SF', 'SH' }

@attribute 'src_bytes' real

@attribute 'dst_bytes' real

@attribute 'land' {'0', '1'}

@attribute 'wrong_fragment' real

@attribute 'urgent' real

@attribute 'hot' real

@attribute 'num_failed_logins' real

@attribute 'logged_in' {'0', '1'}

@attribute 'num_compromised' real

@attribute 'root_shell' real

@attribute 'su_attempted' real

@attribute 'num_root' real

@attribute 'num_file_creations' real

@attribute 'num_shells' real

@attribute 'num_access_files' real

@attribute 'num_outbound_cmds' real

@attribute 'is_host_login' {'0', '1'}

@attribute 'is_guest_login' {'0', '1'}

@attribute 'count' real

@attribute 'srv_count' real

@attribute 'serror_rate' real

@attribute 'srv_serror_rate' real

@attribute 'rerror_rate' real

@attribute 'srv_rerror_rate' real

@attribute 'same_srv_rate' real

@attribute 'diff_srv_rate' real

@attribute 'srv_diff_host_rate' real

@attribute 'dst_host_count' real

@attribute 'dst_host_srv_count' real

@attribute 'dst_host_same_srv_rate' real

@attribute 'dst_host_diff_srv_rate' real

@attribute 'dst_host_same_src_port_rate' real

@attribute 'dst_host_srv_diff_host_rate' real

@attribute 'dst_host_serror_rate' real

@attribute 'dst_host_srv_serror_rate' real

@attribute 'dst_host_rerror_rate' real

@attribute 'dst_host_srv_rerror_rate' real

@attribute 'classatack'
{'normal','apache2','back','land','mailbomb','neptune','pod','processtable','smurf','teardrop','udpstorm','buffer_overflow','httptunnel','loadmodule','perl','ps','rootkit','sqlattack','xterm','ftp_write','guess_passwd','imap','ipsweep','mscan','multihop','named','phf','sendmail','snmpgetattack','snmpguess','spy','warezclient','warezmaster','worm','xlock','xsnoop','tune','scan','nmap','portsweep','saint','satan'}

@attribute 'classcategory' {'normal','dos','r2l','u2r','probing'}

```
@attribute 'dificult' real

@data
```

Apêndice B
Resultados do Experimento de DA

Na Tabela B.1, B.2 e B.3 são apresentados todos os resultados dos experimentos DA realizados respectivamente para o algoritmo KNN, K-Means++ e J48.

Para a Tabela B.1 são apresentados da esquerda para a direita o código do experimento, o algoritmo em teste, o conjunto de dados de treinamento, a quantidade de instâncias do conjunto de treinamento, a quantidade de instâncias classificadas corretamente em treinamento, a porcentagem das instâncias classificadas corretamente em treinamento em relação ao total de instâncias do conjunto de treinamento, a quantidade de instâncias incorretamente classificadas em treinamento, a porcentagem de instâncias incorretamente classificadas em teste, o conjunto de dados de teste empregado, a quantidade de instâncias do conjunto de dados, a quantidade de instâncias corretamente classificadas para o teste, a porcentagem de instâncias corretamente classificadas em relação ao conjunto de dados de teste, a quantidade de instâncias incorretamente classificadas para o teste, a porcentagem de instâncias incorretamente classificadas para o teste, se na preparação de dados o conjunto de dados foi normalizado, o valor de *k* e um campo de observação geral para o experimento.

Para a Tabela B.2 são apresentados, da esquerda para a direita, o código do experimento, o algoritmo, conjunto de dados de teste, a quantidade de instâncias corretas, porcentagem de instâncias corretas, quantidade de instâncias incorretas, porcentagem das instâncias incorretas, filtros aplicados, valor de k e um campo de observações gerais.

Para a Tabela B.3 são apresentados, da esquerda para direta, o código do teste, o algoritmo, o conjunto de dados de treinamento, indicação de não poda da árvore, quantidade de instâncias , quantidade de instâncias corretas de treinamento, porcentagem das instâncias corretas em treinamento, conjunto de dados de teste, indicação de não poda da árvore, quantidade de instâncias classificadas corretas em teste, porcentagem de

instâncias classificadas como corretas em teste, quantidade de instâncias classificadas como incorretas em teste, porcentagem de instâncias classificadas como incorretas, filtros aplicados, e observações gerais.

Tabela B.1 - Relação dos experimentos DA executados para o algoritmo KNN.

COD. DO EXP.	ALGORITMO	CONJUNTO TREINA-MENTO	QTD. INST.	CLASSIF. INSTÂNCIA CORRETAS TREIN. (%)	CLASSIF. INSTÂNCIA INCORRETAS TREIN. (%)	CONJUNTO DE TESTE	QTD. INST.	CLASSIF. INSTÂNCIA CORRETAS TESTE (%)	CLASSIF. INSTÂNCIA INCORRETAS TESTE (%)	NORMAL.	*k*
8		***KDDTrain+.arff***		***125966 (99,994)***	***7 (0,0056)***			***17890 (79,356)***	***4654 (20,6441)***	***NÃO***	***1***
12		KDDTrain+.arff		125966 (99,994)	7 (0,0056)			17890 (79,356)	4654 (20,6441)	NÃO	1
11		KDDTrain+.arff		125966 (99,994)	7 (0,0056)			17742 (78,699)	4802 (21,3006)	SIM	1
10		KDDTrain+.arff		125595 (99,700)	378 (0,3001)			17721 (78,606)	4823 (21,3937)	NÃO	7
7		KDDTrain+.arff		125789 (99,854)	184 (0,1461)			17717 (78,589)	4827 (21,4115)	NÃO	3
9		KDDTrain+.arff		125686 (99,772)	287 (0,2278)			17650 (78,291)	4894 (21,7087)	NÃO	5
31	***KNN (LAZY IBK)***	KDDTrain+.arff	***125973***	125686 (99,772)	287 (0,2278)	***KDDTest+.arff***	***22544***	17577 (77,968)	4967 (22,0325)	SIM	5
29		KDDTrain+.arff		125789 (99,854)	184 (0,1461)			17315 (76,805)	5229 (23,1946)	SIM	3
30		KDDTrain+.arff		125562 (99,674)	411 (0,3263)			17315 (76,805)	5229 (23,1946)	SIM	3
32		KDDTrain+.arff		125789 (99,854)	184 (0,1461)			17315 (76,805)	5229 (23,1946)	SIM	3
13		KDDTrain+.arff		125715 (99,795)	258 (0,2048)			16711 (74,126)	5833 (25,8738)	SIM	5
27		KDDTrain+.arff		125966 (99,994)	7 (0,0056)			6351 (28,172)	16193 (71,8284)	SIM	1
33		KDDTrain+.arff		125595 (99,699)	378 (0,3001)			17585 (78,000)	4959 (21,9960)	SIM	7

Tabela B.2 - Relação dos experimentos DA executados para o algoritmo K-Means ++.

CÓD. DO EXP.	ALGORITMO	CONJUNTO DE DADOS DE TESTE	QTD. INST.	CLASSIF. INSTÂNCIA CORRETAS TESTE (%)	CLASSIF INSTÂNCIA INCORRETAS TESTE (%)	FILTRO		OBS.
						NORM.	*k*	
38	**SIMPLE KMEANS**	**KDDTest+.arff**	**22544**	**17576 (77,963)**	**4968 (22,0369)**	**SIM**	**2**	**NORMALIZADO POR INSTÂNCIA UTILIZANDO A DISTÂNCIA MANHATTAN**
37				17558 (77,883)	4986 (22,1167)	SIM		NORMALIZADO POR INSTÂNCIA UTILIZANDO A DISTÂNCIA EUCLIDIANA
34				16495 (73,168)	6049 (26,8320)	NÃO		UTILIZANDO A DISTÂNCIA EUCLIDIANA
35				16495 (73,168)	6049 (26,8320)	SIM		NORMALIZADO POR ATRIBUTO UTILIZANDO A DISTÂNCIA EUCLIDIANA
36				15345 (68,067)	7199 (31,9331)	NÃO		UTILIZANDO A DISTÂNCIA EUCLIDIANA

Tabela B.3 - Relação dos experimentos DA executados para o algoritmo J48.

COD. DO EXP.	ALG.	CONJUNTO DE DADOS DE TREINAMENTO	UNPRUNED	QTD. INST.	CLASSIF. INSTÂNCIA CORRETAS TREIN. (%)	CLASSIF. INSTÂNCIA INCORRETAS TREIN. (%)	CONJUNTO DE DADOS DE TESTE	QTD. INST.	CLASSIF. INSTÂNCIA CORRETAS TESTE (%)	CLASSIF INSTÂNCIA INCORRETAS TESTE (%)	FILTRO	OBS.
											NORMAL.	
2	**J48**	**KDDTrain+.arff**	**NÃO**	**125973**	**125861 (99,911)**	**112 (0,0889)**	**KDDTest+.arff**	**22544**	**18381 (81,534)**	**4163 (18,4661)**	**NÃO**	-
3			SIM		125918 (99,956)	55 (0,0437)			18346 (81,379)	4198 (18,6214)	NÃO	-
1			SIM		125918 (99,956)	55 (0,0437)			18346 (81,379)	4198 (18,6214)	NÃO	-
6			SIM		125863 (99,913)	110 (0,0873)			13469 (59,745)	9075 (40,2546)	SIM	NORMALIZADO POR ATRIBUTO - UMPRUNED = SIM
4			NÃO		125774 (99,842)	199 (0,1580)			12057 (53,482)	10487 (46,5179)	SIM	NORMALIZADO POR ATRIBUTO
14			SIM		125912 (99,952)	61 (0,0484)			8295 (36,795)	14249 (63,2053)	SIM	NORMALIZADO POR INSTÂNCIA - UNPRUNED = TRUE
5			NÃO		125815 (99,875)	158 (0,1254)			7465 (33,113)	15079 (66,8870)	SIM	NORMALIZADO POR INSTÂNCIA

Apêndice C
Resultados do Experimento de DTA

Na Tabela C.1, C.2 e C.3 são apresentados todos os resultados dos experimentos DTA realizados respectivamente para o algoritmo KNN, K-Means++ e J48.

Para a Tabela C.1 são apresentados, da esquerda para a direita, o código do experimento, o algoritmo em teste, o conjunto de dados de teste, a quantidade de instâncias do conjunto de treinamento, taxa de falsos positivos para a classe “normal”, taxas de falsos positivos para classe “dos”, taxa de falsos positivo para classe “r2l”, taxa de falsos positivos para a classe “u2r”, taxa de falsos positivos para a classe “probing”, taxa de acurácia, taxa de erro total, valor de *k*, normalização por atributo e por instância, distância e *NNSearch*.

Para a Tabela C.2 são apresentados, da esquerda para a direita, o código do experimento, o algoritmo, o conjunto de dados, quantidade de instância, a taxa de falsos positivos para a classe “normal”, a taxa de falsos positivos para a classe “dos”, a taxa de falsos positivos para a classe “r2l”, a taxa de falsos positivos para a classe “u2r”, a taxa de falsos positivos para a classe “probing”, a taxa de acurácia, a taxa de erro total , valor de *k*, normalização por atributo e por instância e a distância.

Para a Tabela C.3 são apresentados, da esquerda para a direita, o código do experimento, o algoritmo, o conjunto de dados, quantidade de instância, a taxa de falsos positivos para a classe “normal”, a taxa de falsos positivos para a classe “dos”, a taxa de falsos positivos para a classe “r2l”, a taxa de falsos positivos para a classe “u2r”, a taxa de falsos positivos para a classe “probing”, a taxa de acurácia, a taxa de erro total, normalização por atributo e por instância, a distância e a poda da árvore.

Tabela C.1 - Relação dos experimentos DTA executados para o algoritmo KNN.

COD. DO EXP.	ALGORITMO	CONJUNTO TREINAMENTO	CONJUNTO TESTE	QTD. INST.	NORMAL		DOS		R2L		U2R		PROBING		TAXA DE ACURACIA	TAXA DE ERRO TOTAL	*k*	NORMALIZAÇÃO		DISTÂNCIA
					TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP				ATRIBUTO	INSTÂNCIA	
1000	***KNN (LAZY IBK)***	***KDDTrain+.arff***	***KDDTest+.arff***	***22544***	***0,962***	***0,348***	***0,811***	***0,020***	***0,067***	***0,012***	***0,100***	***0,003***	***0,737***	***0,024***	***0,771***	***0,229***	***1***	***NÃO***	***NÃO***	***EUCLIDIANA***
1003					0,962	0,357	0,814	0,021	0,061	0,002	0,095	0,002	0,735	0,024	0,771	0,229	3	NÃO	NÃO	
1006					0,963	0,363	0,815	0,021	0,044	0,003	0,085	0,000	0,729	0,023	0,769	0,231	5	NÃO	NÃO	
1009					0,963	0,359	0,818	0,022	0,044	0,003	0,065	0,000	0,726	0,027	0,769	0,231	7	NÃO	NÃO	
1007					0,930	0,343	0,815	0,026	0,118	0,003	0,050	0,000	0,741	0,040	0,765	0,235	5	SIM	NÃO	
1010					0,930	0,343	0,816	0,027	0,119	0,003	0,030	0,000	0,737	0,041	0,765	0,235	7	SIM	NÃO	
1001					0,937	0,341	0,798	0,027	0,138	0,011	0,095	0,000	0,724	0,041	0,763	0,237	1	SIM	NÃO	
1004					0,930	0,365	0,776	0,028	0,130	0,003	0,045	0,000	0,727	0,042	0,752	0,248	3	SIM	NÃO	
1008					0,907	0,564	0,296	0,021	0,179	0,005	0,030	0,000	0,462	0,289	0,560	0,440	5	NÃO	SIM	
1011					0,908	0,572	0,312	0,022	0,102	0,005	0,000	0,000	0,474	0,291	0,557	0,443	7	NÃO	SIM	
1005					0,906	0,588	0,260	0,022	0,180	0,012	0,005	0,014	0,440	0,296	0,545	0,455	3	NÃO	SIM	
1002					0,193	0,780	0,149	0,078	0,231	0,096	0,005	0,039	0,253	0,716	0,188	0,812	1	NÃO	SIM	

Tabela C.2 - Relação dos experimentos DTA executados para o algoritmo K-Means++.

COD. DO EXP.	ALGORITMO	CONJUNTO TESTE	QTD. INST.	NORMAL		DOS		R2L		U2R		PROBING		TAXA DE ACURACIA	TAXA DE ERRO TOTAL	VALOR DE K	NORMALIZAÇÃO		DISTÂNCIA
				TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP				ATRIBUTO	INSTÂNCIA	
1018	***SIMPLE KMEANS++***	***KDDTest+.arff***	***22544***	***0,756***	***0,244***	***0,559***	***0,441***	***0,336***	***0,664***	***0,000***	***1,000***	***0,266***	***0,734***	***0,580***	***0,420***	***5***	***NÃO***	***SIM***	***EUCLIDIANA***
1020				0,729	0,271	0,579	0,421	0,290	0,710	0,000	1,000	0,306	0,694	0,574	0,426		NÃO	SIM	MANHATTAN
1021				0,792	0,208	0,418	0,582	0,322	0,678	0,000	1,000	0,321	0,679	0,553	0,447		NÃO	NÃO	MANHATTAN
1023				0,341	0,208	0,138	0,582	0,039	0,678	0,000	1,000	0,321	0,679	0,553	0,447		SIM	NÃO	MANHATTAN
1019				0,757	0,243	0,412	0,588	0,403	0,597	0,000	1,000	0,335	0,665	0,548	0,452		NÃO	NÃO	EUCLIDIANA
1022				0,727	0,243	0,412	0,588	0,403	0,597	0,000	1,000	0,335	0,065	0,548	0,452		SIM	NÃO	EUCLIDIANA

Tabela C.3 - Relação dos experimentos DTA executados para o algoritmo J48.

COD. DO EXP.	ALG.	CONJUNTO TREINAMENTO	CONJUNTO TESTE	QTD. INST.	NORMAL		DOS		R2L		U2R		PROBING		TAXA DE ACURACIA	TAXA DE ERRO TOTAL	NORMALIZAÇÃO		UNPRUNED
					TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP	TAXA VP	TAXA FP			ATRIBUTO	INSTÂNCIA	
1015	***J48***	***KDDTrain+.arff***	***KDDTest+.arff***	***22544***	***0,941***	***0,270***	***0,901***	***0,115***	***0,218***	***0,009***	***0,005***	***0,000***	***0,490***	***0,011***	***0,783***	***0,217***	***SIM***	***NÃO***	***NÃO***
1017					0,970	0,383	0,774	0,026	0,065	0,000	0,045	0,000	0,065	0,037	0,753	0,247	NÃO	NÃO	NÃO
1016					0,954	0,392	0,774	0,025	0,019	0,001	0,050	0,000	0,660	0,048	0,741	0,259	NÃO	NÃO	SIM
1013					0,174	0,063	0,996	0,816	0,000	0,007	0,000	0,000	0,390	0,013	0,446	0,554	SIM	NÃO	SIM
1012					0,304	0,585	0,207	0,600	0,005	0,065	0,000	0,002	0,672	0,505	0,272	0,728	NÃO	SIM	SIM
1014					0,304	0,596	0,207	0,600	0,004	0,060	0,000	0,007	0,672	0,493	0,272	0,728	NÃO	SIM	NÃO

Apêndice D

Matrizes de Confusão do Experimento de DTA

Na Tabela D.1 são apresentadas as matrizes de confusão geradas pelo ambiente WEKA no experimento DTA, consolidadas para este estudo.

As interpretações das matrizes de confusão, dos experimentos de DA e DTA, são apresentadas na Seção 5.3.1.

Tabela D.1 - Matrizes de confusão utilizadas no experimento DTA.

Experimento	**1000**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa VP**	**Taxa FP**	**Taxa de erro por categoria**
Normal	**9342**	57	3	2	307	9711	0,962	0,348	0,962
dos	1139	**6050**	189	10	70	7458	0,811	0,020	0,811
r2l	2523	2	**184**	40	5	2754	0,067	0,012	0,067
u2r	170	0	2	**20**	8	200	0,100	0,003	0,100
Probing	453	174	8	2	**1784**	2421	0,737	0,024	0,737
Total por categoria	13627	6283	386	74	2174	22544	**Total de instâncias**		
Precisão	0,686	0,963	0,477	0,270	0,821	17380	**Total de instâncias corretas**		
Taxa de acurácia	0,771					5164	**Total de instâncias incorretas**		
Taxa erro total	0,229								
Experimento	**1001**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9096**	81	11	1	522	9711	0,937	0,341	0,937
dos	1242	**5950**	167	0	99	7458	0,798	0,027	0,798

r2l	2337	2	**381**	5	29	2754	0,138	0,011	0,138
u2r	171	0	3	**19**	7	200	0,095	0,000	0,095
Probing	437	229	0	2	**1753**	2421	0,724	0,041	0,724
Total por categoria	13283	6262	562	27	2410	22544	**Total de instâncias**		
Precisão	0,685	0,950	0,678	0,704	0,727	17199	**Total de instâncias corretas**		
Taxa de acurácia	0,763					5345	**Total de instâncias incorretas**		
Taxa erro total	0,237								
Experimento	**1002**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**1875**	25	356	33	7422	9711	0,193	0,780	0,193
dos	4773	**1112**	0	0	1573	7458	0,149	0,078	0,149
r2l	1980	4	**637**	16	117	2754	0,231	0,096	0,231
u2r	169	0	12	**1**	18	200	0,005	0,039	0,005
Probing	1435	234	15	124	**613**	2421	0,253	0,716	0,253
Total por categoria	10232	1375	1020	174	9743	22544	**Total de instâncias**		
Precisão	0,183	0,809	0,625	0,006	0,063	4238	**Total de instâncias corretas**		

Taxa de acurácia	0,188					18306	**Total de instâncias incorretas**		
Taxa erro total	0,812								
Experimento	**1003**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9346**	56	3	5	301	9711	0,962	0,357	0,962
dos	1331	**6071**	0	0	56	7458	0,814	0,021	0,814
r2l	2554	2	**167**	27	4	2754	0,061	0,002	0,061
u2r	163	0	1	**19**	17	200	0,095	0,002	0,095
Probing	422	187	33	0	**1779**	2421	0,735	0,024	0,735
Total por categoria	13816	6316	204	51	2157	22544	**Total de instâncias**		
Precisão	0,676	0,961	0,819	0,373	0,825	17382	**Total de instâncias corretas**		
Taxa de acurácia	0,771					5162	**Total de instâncias incorretas**		
Taxa erro total	0,229								
Experimento	**1004**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**

Normal	**9029**	83	14	1	584	9711	0,930	0,365	0,930
dos	1610	**5788**	0	0	60	7458	0,776	0,028	0,776
r2l	2381	5	**357**	3	8	2754	0,130	0,003	0,130
u2r	174	0	1	**9**	16	200	0,045	0,000	0,045
Probing	388	235	39	0	**1759**	2421	0,727	0,042	0,727
Total por categoria	13582	6111	411	13	2427	22544	**Total de instâncias**		
Precisão	0,665	0,947	0,869	0,692	0,725	16942	**Total de instâncias corretas**		
Taxa de acurácia	0,752					5602	**Total de instâncias incorretas**		
Taxa erro Total	0,248								
Experimento	**1005**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**8798**	46	44	33	790	9711	0,906	0,588	0,906
dos	1885	**1936**	81	0	3556	7458	0,260	0,022	0,260
r2l	1864	4	**496**	16	374	2754	0,180	0,012	0,180
u2r	199	0	0	**1**	0	200	0,005	0,014	0,005
Probing	1037	179	16	123	**1066**	2421	0,440	0,296	0,440
Total por categoria	13783	2165	637	173	5786	22544	**Total de instâncias**		

Precisão	0,638	0,894	0,779	0,006	0,184	12297	**Total de instâncias corretas**		
Taxa de acurácia	0,545					10247	**Total de instâncias incorretas**		
Taxa erro total	0,455								
Experimento	**1006**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9352**	57	3	3	296	9711	0,963	0,363	0,963
dos	1329	**6081**	0	0	48	7458	0,815	0,021	0,815
r2l	2627	2	**121**	0	4	2754	0,044	0,003	0,044
u2r	165	0	1	**17**	17	200	0,085	0,000	0,085
Probing	431	182	43	0	**1765**	2421	0,729	0,023	0,729
Total por categoria	13904	6322	168	20	2130	22544	**Total de instâncias**		
Precisão	0,673	0,962	0,720	0,850	0,829	17336	**Total de instâncias corretas**		
Taxa de acurácia	0,769					5208	**Total de instâncias incorretas**		
Taxa erro total	0,231								
Experimento	**1007**	KNN							

	Normal	dos	r2l	u2r	Probing	Total por categoria	Taxa de VP	Taxa de FP	Taxa de erro por categoria
Normal	**9032**	85	15	1	578	9711	0,930	0,343	0,930
dos	1330	**6077**	0	0	51	7458	0,815	0,026	0,815
r2l	2412	5	**326**	3	8	2754	0,118	0,003	0,118
u2r	177	0	1	**10**	12	200	0,050	0,000	0,050
Probing	374	212	42	0	**1793**	2421	0,741	0,040	0,741
Total por categoria	13325	6379	384	14	2442	22544	**Total de instâncias**		
Precisão	0,678	0,953	0,849	0,714	0,734	17238	**Total de instâncias corretas**		
Taxa de acurácia	0,765					5306	**Total de instâncias incorretas**		
Taxa erro total	0,235								
Experimento	**1008**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**8806**	45	43	2	815	9711	0,907	0,564	0,907
dos	1764	**2205**	2	0	3487	7458	0,296	0,021	0,296
r2l	1870	4	**492**	4	384	2754	0,179	0,005	0,179
u2r	194	0	0	**6**	0	200	0,030	0,000	0,030
Probing	1107	180	15	0	**1119**	2421	0,462	0,289	0,462

Total por categoria	13741	2434	552	12	5805	22544	**Total de instâncias**		
Precisão	0,641	0,906	0,891	0,500	0,193	12628	**Total de instâncias corretas**		
Taxa de acurácia	0,560					9916	**Total de instâncias incorretas**		
Taxa erro Total	0,440								
Experimento	**1009**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9348**	59	2	4	298	9711	0,963	0,359	0,963
dos	1317	**6097**	0	0	44	7458	0,818	0,022	0,818
r2l	2624	2	**120**	1	7	2754	0,044	0,003	0,044
u2r	100	0	1	**13**	86	200	0,065	0,000	0,065
Probing	429	192	43	0	**1757**	2421	0,726	0,027	0,726
Total por categoria	13818	6350	166	18	2192	22544	**Total de instâncias**		
Precisão	0,677	0,960	0,723	0,722	0,802	17335	**Total de instâncias corretas**		
Taxa de acurácia	0,769					5209	**Total de instâncias incorretas**		
Taxa erro total	0,231								

Experimento	**1010**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9032**	85	14	1	579	9711	0,930	0,343	0,930
dos	1318	**6088**	0	0	52	7458	0,816	0,027	0,816
r2l	2413	2	**328**	1	10	2754	0,119	0,003	0,119
u2r	177	0	1	**6**	16	200	0,030	0,000	0,030
Probing	373	221	42	0	**1785**	2421	0,737	0,041	0,737
Total por categoria	13313	6396	385	8	2442	22544	**Total de instâncias**		
Precisão	0,678	0,952	0,852	0,750	0,731	17239	**Total de instâncias corretas**		
Taxa de acurácia	0,765					5305	**Total de instâncias incorretas**		
Taxa erro total	0,235								
Experimento	**1011**	KNN							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**8815**	43	39	0	814	9711	0,908	0,572	0,908
dos	1645	**2324**	2	0	3487	7458	0,312	0,022	0,312
r2l	2088	5	**280**	0	381	2754	0,102	0,005	0,102
u2r	200	0	0	**0**	0	200	0,000	0,000	0,000

Probing	1079	180	15	0	**1147**	2421	0,474	0,291	0,474
Total por categoria	13827	2552	336	0	5829	22544	**Total de instâncias**		
Precisão	0,638	0,911	0,833	0,000	0,197	12566	**Total de instâncias corretas**		
Taxa de acurácia	0,557					9978	**Total de instâncias incorretas**		
Taxa erro total	0,443								
Experimento	**1012**	J48							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**2950**	6503	51	10	197	9711	0,304	0,585	0,304
dos	1504	**1545**	344	0	4065	7458	0,207	0,600	0,207
r2l	2388	116	**13**	0	237	2754	0,005	0,065	0,005
u2r	99	0	8	**0**	93	200	0,000	0,002	0,000
Probing	495	275	24	0	**1627**	2421	0,672	0,505	0,672
Total por categoria	7436	8439	440	10	6219	22544	**Total de instâncias**		
Precisão	0,397	0,183	0,030	0,000	0,262	6135	**Total de instâncias corretas**		
Taxa de acurácia	0,272					16409	**Total de instâncias incorretas**		

Taxa erro total	0,728								
Experimento	**1013**	j48							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**1694**	7911	0	0	106	9711	0,174	0,063	0,174
dos	23	**7426**	0	0	9	7458	0,996	0,816	0,996
r2l	516	2237	**0**	0	1	2754	0,000	0,007	0,000
u2r	1	199	0	**0**	0	200	0,000	0,000	0,000
Probing	24	1380	73	0	**944**	2421	0,390	0,013	0,390
Total por categoria	2258	19153	73	0	1060	22544	**Total de instâncias**		
Precisão	0,750	0,388	0,000	0,000	0,891	10064	**Total de instâncias corretas**		
Taxa de acurácia	0,446					12480	**Total de instâncias incorretas**		
Taxa erro total	0,554								
Experimento	**1014**	j48							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**2953**	6503	19	42	194	9711	0,304	0,596	0,304
dos	1504	**1545**	344	0	4065	7458	0,207	0,600	0,207
r2l	2589	116	**12**	0	37	2754	0,004	0,060	0,004

u2r	103	0	4	**0**	93	200	0,000	0,007	0,000
Probing	495	275	24	0	**1627**	2421	0,672	0,493	0,672
Total por categoria	7644	8439	403	42	6016	22544	**Total de instâncias**		
Precisão	0,386	0,183	0,030	0,000	0,270	6137	**Total de instâncias corretas**		
Taxa de acurácia	0,272					16407	**Total de instâncias incorretas**		
Taxa erro total	0,728								
Experimento	**1015**	j48							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9140**	345	59	0	167	9711	0,941	0,270	0,941
dos	720	**6717**	3	0	18	7458	0,901	0,115	0,901
r2l	1762	390	**599**	0	3	2754	0,218	0,009	0,218
u2r	106	80	13	**1**	0	200	0,005	0,000	0,005
Probing	557	604	73	0	**1187**	2421	0,490	0,011	0,490
Total por categoria	12285	8136	747	1	1375	22544	**Total de instâncias**		
Precisão	0,744	0,826	0,802	1,000	0,863	17644	**Total de instâncias corretas**		

Taxa de acurácia	0,783					4900	**Total de instâncias incorretas**		
Taxa erro Total	0,217								
Experimento	**1016**	j48							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**
Normal	**9265**	70	3	2	371	9711	0,954	0,392	0,954
dos	1594	**5775**	0	0	89	7458	0,774	0,025	0,774
r2l	2408	1	**51**	3	291	2754	0,019	0,001	0,019
u2r	176	1	6	**10**	7	200	0,050	0,000	0,050
Probing	619	205	0	0	**1597**	2421	0,660	0,048	0,660
Total por categoria	14062	6052	60	15	2355	22544	**Total de instâncias**		
Precisão	0,659	0,954	0,850	0,667	0,678	16698	**Total de instâncias corretas**		
Taxa de acurácia	0,741					5846	**Total de instâncias incorretas**		
Taxa erro total	0,259								
Experimento	**1017**	j48							
	Normal	**dos**	**r2l**	**u2r**	**Probing**	**Total por categoria**	**Taxa de VP**	**Taxa de FP**	**Taxa de erro por categoria**

Normal	**9420**	82	3	1	205	9711	0,970	0,383	0,970
dos	1598	**5772**	3	0	85	7458	0,774	0,026	0,774
r2l	2279	0	**180**	3	292	2754	0,065	0,000	0,065
u2r	181	1	2	**9**	7	200	0,045	0,000	0,045
Probing	620	216	0	0	**1585**	2421	0,655	0,037	0,655
Total por categoria	14098	6071	188	13	2174	22544	**Total de instâncias**		
Precisão	0,668	0,951	0,957	0,692	0,729	16966	**Total de instâncias corretas**		
Taxa de acurácia	0,753					5578	**Total de instâncias incorretas**		
Taxa erro total	0,247								
Experimento	**1018**	**K-Means++**							
Cluster	**0**	**1**	**2**	**3**	**4**	**Categorias**	**Total de instâncias por categoria**	**Total de inst. Erro por categoria**	**Taxa de VP**	**Taxa de FP**
Cluster 0 <-- r2l	478	28	1815	45	**7345**	**normal**	9711	2366	0,756	0,244
Cluster 1 <-- probing	1175	957	14	**4169**	1143	**dos**	7458	3289	0,559	0,441
Cluster 2 <-- No class	**924**	27	513	3	1287	**r2l**	2754	1830	0,336	0,664
Cluster 3 <-- dos	37	**0**	0	109	54	**u2r**	200	200	0,000	1,000

Cluster 4 <-- normal	182	**644**	516	1072	7	**probing**	2421	1777	0,266	0,734
Total instâncias no cluster	2796	1656	2858	5398	9836		22544			
% distribuição cluster	12%	7%	13%	24%	44%					
	Total de instâncias				22544					
		N° instâncias								
		22.544								
Taxa de acurácia		13.082	0,580	**%total acertos = Soma do Total de acertos de cada categoria por cada cluster**						
Taxa de Erro Total		9.462	0,420	**%total erros = tota de instâncias - total de acertos**						
Experimento	**1019**	**K-Means++**								
Cluster	**0**	**1**	**2**	**3**	**4**	**Categorias**	**Total de instâncias por categoria**	**Total de inst. Erro por categoria**	**Taxa de VP**	**Taxa de FP**
Cluster 0 <-- No class	7	20	2265	63	**7356**	**normal**	9711	2355	0,757	0,243
Cluster 1 <-- probing	1957	78	829	**3071**	1523	**dos**	7458	4387	0,412	0,588
Cluster 2 <-- r2l	4	29	**1110**	2	1609	**r2l**	2754	1644	0,403	0,597

Cluster 3 <-- dos	0	4	33	109	54	**u2r**	200	200	0,000	1,000
Cluster 4 <-- normal	312	**810**	582	712	5	**probing**	2421	1611	0,335	0,665
Total instâncias no cluster	2280	941	4819	3957	10547		22544			
% distribuição cluster	10%	4%	21%	18%	47%					
			Total de instâncias		22544					
		N° instâncias								
		22.544								
				%total acertos = soma do total de acertos de cada categoria por cada cluster						
Taxa de acurácia		12.347	0,548							
Taxa de erro total		10.197	0,452	**%total erros = total de instâncias - total de acertos**						
Experimento	**1020**	**K-Means++**								
Cluster	**0**	**1**	**2**	**3**	**4**	**Categorias**	**Total de instâncias por categoria**	**Total de inst. Erro por categoria**	**Taxa de VP**	**Taxa de FP**
Cluster 0 <-- r2l	706	70	1817	42	**7076**	**normal**	9711	2635	0,729	0,271
Cluster 1 <-- probing	1468	810	14	**4316**	850	**dos**	7458	3142	0,579	0,421

Cluster 2 <-- No class	**799**	95	527	3	1330	**r2l**	2754	1955	0,290	0,710
Cluster 3 <-- dos	21	1	0	109	69	**u2r**	200	200	0,000	1,000
Cluster 4 <-- normal	95	**740**	518	1053	15	**probing**	2421	1681	0,306	0,694
Total instâncias no cluster	3089	1716	2876	5523	9340		22544			
% distribuição cluster	14%	8%	13%	24%	41%					
				Total de instâncias	22544					
		N° instâncias								
		22.544								
Taxa de acurácia		12.931	0,574	**%total acertos = Soma do Total de acertos de cada categoria por cada cluster**						
Taxa de Erro Total		9.613	0,426	**%total erros = tota de instâncias - total de acertos**						
Experimento	**1021**	**K-Means++**								
Cluster	**0**	**1**	**2**	**3**	**4**	**Categorias**	**Total de instâncias por categoria**	**Total de inst. Erro por categoria**	**Taxa de VP**	**Taxa de FP**
Cluster 0 <-- No class	6	8	1934	75	**7688**	**normal**	9711	2023	0,792	0,208

Cluster 1 <-- probing	1853	25	720	**3117**	1743	**dos**	7458	4341	0,418	0,582
Cluster 2 <-- r2l	2	20	**886**	3	1843	**r2l**	2754	1868	0,322	0,678
Cluster 3 <-- dos	0	**0**	9	112	79	**u2r**	200	200	0,000	1,000
Cluster 4 <-- normal	312	**776**	559	741	33	**probing**	2421	1645	0,321	0,679
Total instâncias no cluster	2173	829	4108	4048	11386		22544			
% distribuição cluster	10%	4%	18%	18%	51%					
	Total de instâncias				22544					
		N° instâncias								
		22.544								
Taxa de acurácia		12.467	0,553	**%total acertos = soma do total de acertos de cada categoria por cada cluster**						
Taxa de erro total		10.077	0,447	**%total erros = total de instâncias - total de acertos**						
Experimento	**1022**	**K-Means++**								
Cluster	**0**	**1**	**2**	**3**	**4**	**Categorias**	**Total de instâncias por categoria**	**Total de inst. Erro por categoria**	**Taxa de VP**	**Taxa de FP**

Cluster 0 <-- No class	7	20	2265	63	**7356**	**normal**	9711	2355	0,757	0,243
Cluster 1 <-- probing	1957	78	829	**3071**	1523	**dos**	7458	4387	0,412	0,588
Cluster 2 <-- r2l	4	29	**1110**	2	1609	**r2l**	2754	1644	0,403	0,597
Cluster 3 <-- dos	0	4	33	109	54	**u2r**	200	200	0,000	1,000
Cluster 4 <-- normal	312	**810**	582	712	5	**probing**	2421	1611	0,335	0,665
Total instancias no cluster	2280	941	4819	3957	10547		22544			
% distribuição cluster	10%	4%	21%	18%	47%					
	Total de instâncias				22544					
		N° instâncias								
		22.544								
Taxa de acurácia		12.347	0,548	**%total acertos = soma do total de acertos de cada categoria por cada cluster**						
Taxa de erro total		10.197	0,4523	**%total erros = total de instâncias - total de acertos**						
Experimento	**1023**	**K-Means++**								

Cluster	0	1	2	3	4	Categorias	Total de instâncias por categoria	Total de inst. Erro por categoria	Taxa de VP	Taxa de FP
Cluster 0 <-- No class	6	8	1934	75	**7688**	**normal**	9711	2023	0,341	0,208
Cluster 1 <-- probing	1853	25	720	**3117**	1743	**dos**	7458	4341	0,138	0,582
Cluster 2 <-- r2l	2	20	**886**	3	1843	**r2l**	2754	1868	0,039	0,678
Cluster 3 <-- dos	0	**0**	9	112	79	**u2r**	200	200	0,000	1,000
Cluster 4 <-- normal	312	**776**	559	741	33	**probing**	2421	1645	0,321	0,679
Total instancias no cluster	2173	829	4108	4048	11386		22544			
% distribuição cluster	10%	4%	18%	18%	51%					
	Total de instâncias				22544					
		N° instâncias								
		22.544								
Taxa de acurácia		12.467	0,553	**%total acertos = Soma do Total de acertos de cada categoria por cada cluster**						
Taxa de erro total		10.077	0,447	**%total erros = total de instâncias - total de acertos**						