Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Aplicação de Técnicas de Mineração de Texto para Categorização de Eventos de Segurança no CTIR Gov

Albert Frederico de Menezes Il Pak

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Prof.ª Dr.ª Célia Ghedini Ralha

Brasília
2010

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

Coordenadora: Prof.ª Dr.ª Alba Cristina Magalhães de Melo

Banca examinadora composta por:

Prof.ª Dr.ª Célia Ghedini Ralha (Orientadora) — CIC/UnB
Prof. Dr. Antonio Montes Filho — CTI/MCT
Prof. Dr. Marcelo Ladeira — CIC/UnB

CIP — Catalogação Internacional na Publicação

Pak, Albert Frederico de Menezes Il.

Aplicação de Técnicas de Mineração de Texto para Categorização de Eventos de Segurança no CTIR Gov / Albert Frederico de Menezes Il Pak. Brasília : UnB, 2010.
82 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2010.

1. CTIR Gov, 2. triagem, 3. categorização, 4. mineração de textos, 5. classificação supervisionada

CDU 004.8

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

ALBERT FREDERICO DE MENEZES IL PARK

Aplicação de Técnicas de Mineração de Textos para Categorização de Eventos de Segurança no CTIR Gov

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre no Curso de Pós-graduação em Informática da Universidade de Brasília, pela Comissão formada pelos professores:

Orientadora: Profª. Drª. Célia Ghedini Ralha
(CIC/UnB)

Prof. Dr. Antonio Montes Filho
(CTI/MCT)

Prof. Dr. Marcelo Ladeira
(CIC/UnB)

Vista e permitida a impressão.
Brasília, 09 de abril de 2010.

Profª. Drª. Alba Cristina Magalhães Alves de Melo
Coordenadora de Pós-graduação
Departamento de Ciência da Computação
Universidade de Brasília

Dedicatória

Dedico essa dissertação aos amigos pela amizade, incentivo e paciência em todos momentos deste período.

Agradecimentos

Agradeço à Professora Célia por acreditar em meu potencial e por me ajudar mesmo nas horas mais difíceis. Ao Roberto Pimenta pelo grande apoio prestado e por aceitar um aprendiz em sua excelente equipe técnica. Também agradeço Beth e Rachel pelo amor com qual sempre poderei contar. À família pela convivência em todos momentos especiais. À CAPES pelo apoio financeiro. Aos amigos pela alegria de alcançar felicidade e sabedoria. A todos aqueles que acreditam em impossibilidades e que pensaram positivamente para o êxito desta dissertação.

Resumo

De acordo com a metodologia de tratamento de incidentes de segurança da Universidade de *Carnegie Mellon*, todos os eventos recebidos em um centro de tratamento de incidentes passam pela tarefa de categorização. Dependendo da importância e abrangência da atuação do grupo de resposta a incidentes, o número de eventos ou mensagens pode se tornar difícil de ser classificado manualmente ou através do uso de filtros de mensagens. Neste trabalho, foi proposta uma solução para a classificação supervisionada de eventos no âmbito do Centro de Tratamento de Incidentes de Segurança em Redes de Computadores da Administração Pública Federal (CTIR Gov). A solução adotada inclui um processo de mineração de textos com uma fase de pré-processamento e uso da ferramenta PreTexT, além da fase de classificação automática de eventos, utilizando a ferramenta de mineração Weka. Na experimentação, foram adotados três algoritmos diferentes: (i) J48, da família de árvores de decisão; (ii) *Naïve Bayes*, pela sua abordagem probabilística; e (iii) *Support Vector Machine* (SVM) com otimização *Sequential Minimal Optimization* (SMO). Para avaliação dos resultados, foram realizadas comparações entre a categorização semi-automática registrada em relatórios consolidados e os algoritmos citados. Foram obtidos resultados de classificação com índice de acerto na ordem de 73%.

Palavras-chave: CTIR Gov, triagem, categorização, mineração de textos, classificação supervisionada

Abstract

According to Carnegie Mellon University computer security incident response methodology, all events in a computer security incident response team should be categorized. Depending on the importance and scope of the incident response team, the number of events or messages can become difficult to be classified manually or only by means of message filters. This work, proposes a solution to supervised categorization of events in the National Brazilian Government Computer Security Incident Response Team (CTIR Gov). The adopted solution includes a text mining process with a preprocess stage using the PreTexT tool, and an automatic event categorization phase using the data mining tool Weka. In the experimental phase, three algorithms were adopted: (i) J48, from the decision tree family; (ii) *Naïve Bayes*, with its probabilistic approach; and (iii) Support Vector Machine (SVM) optimized by Sequential Minimal Optimization (SMO). To evaluate the results, comparisons were conducted between the semi-automatic categorization documented in reports and the cited algorithms. Classification results displayed a rate of about 73% of correctly classified instances.

Keywords: CTIR Gov, triage, categorization, text mining, supervised classification

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

De acordo com o Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br), relatos de incidentes reportados têm crescido a um ritmo preocupante. Entre 1999 e 2003, estatísticas mostram um crescimento de 1.758% no número de incidentes reportados [11]. Essa tendência não é isolada no Brasil e configura um fenômeno global, pois, conforme o relatório do *Computer Emergency Response Team Coordination Center* (CERT/CC), também entre 1999 e 2003, houve um aumento de 1.395% em registros de incidentes recebidos [12].

Diante desse cenário, nota-se uma necessidade por soluções capazes de tratar um crescente volume de comunicações, justificando o uso de formas automáticas de classificação de mensagens, diminuindo o esforço de triagem realizado pelos centros de tratamento de incidentes de segurança.

1.1 Motivação

Conforme exposto, ocorre uma necessidade por formas de tratar grandes volumes de notificações de incidentes. Para tratar esse grande volume, técnicas de categorização automática de textos vêm sendo pesquisadas com as mais diversas abordagens. Foram encontrados na literatura muitos trabalhos nesta direção, os quais objetivam a detecção de *spams* em que o problema é de classificação binária [2, 3, 16, 8, 15, 7, 6].

Alguns trabalhos apresentam soluções para problemas mais complexos, como a classificação multi-classe e o caso *multi-label*, quando pertencentes a uma ou mais classes [47]. Também podemos referenciar alguns trabalhos sobre categorização de documentos [46, 26, 39, 5, 23]. Mas infelizmente ainda existem poucos estudos sobre classificação automática de documentos na língua portuguesa [29, 30, 4, 32].

Com base nesse cenário, este trabalho aborda o problema de categorização automática de mensagens eletrônicas recebidas no Centro de Tratamento de Incidentes de Segurança em Redes de Computadores da Administração Pública Federal (CTIR Gov). O CTIR Gov possui importância de caráter nacional, sendo responsável pelo tratamento de incidentes na Administração Pública Federal (APF) e representa o Governo Brasileiro no exterior perante equipes CERT no mundo todo. A sua principal função é a coordenação de incidentes na APF [34].

Neste trabalho, a categorização automática das mensagens é realizada, utilizando-se técnicas de mineração de textos, que vão desde o pré-processamento, valendo-se da ferra-

menta PreTexT [36], até os resultados da classificação supervisionada por meio da ferramenta Weka [17]. A eficiência da classificação supervisionada será avaliada, comparando os resultados da classificação realizada atualmente pelo CTIR Gov, com os resultados dos algoritmos J48 (árvores de decisão), *Naïve Bayes* e *Sequential Minimal Optimization* (SMO) proveniente da técnica *Support Vector Machine* (SVM). Cada algoritmo possui um nível de complexidade e abordagem diferenciados conforme [46].

Atualmente, a equipe do CTIR Gov realiza a triagem de forma semi-automática, em que uma parte é tratada por filtros de mensagens eletrônicas programados e outra parte restante é tratada por um analista técnico. De acordo com estudos realizados utilizando-se no CTIR Gov modelagem de processos, verificou-se que o tempo gasto em triagem possibilita maior entendimento do fato comunicado, entretanto despende-se tempo na separação das mensagens nas classes pré-estabelecidas, além da verificação realizada nas caixas triadas automaticamente [34].

O conjunto de dados utilizado neste trabalho é definido pelas mensagens (caixas de *e-mails*) referentes ao período entre janeiro e agosto de 2009. Esse conjunto inclui mensagens triadas tanto por métodos manuais como automáticos. A cada mensagem é atribuída uma classificação adotada internamente pelo CTIR Gov. Deste modo, as mensagens já possuem suas categorizações, justificando a abordagem supervisionada desse problema. De outro lado, a ausência de rótulos configuraria um caso não-supervisionado, o que implicaria na utilização de técnicas de *clustering*, ou até abordagens mistas (semi-supervisionadas).

Entre janeiro e agosto de 2009, tramitaram no CTIR Gov cerca de 102.000 notificações. A maior parte delas, 97.000 mensagens, foram tratadas por filtros automáticos. A parte restante contabiliza aproximadamente 5.000 mensagens, as quais foram tratadas manualmente. Esse volume de notificações tende a aumentar, tendo em vista a estatística de outros centros de tratamento de incidentes de segurança.

Outro aspecto a se considerar é que os filtros utilizados pelo CTIR Gov necessitam regularmente de manutenção, isso porque a criação ou alteração de uma mensagem padrão pode interferir nas regras anteriormente programadas. Essa situação não é favorável, pois existem técnicas mais promissoras como o uso da mineração de textos, as quais podem trazer um resultado mais favorável.

Considerando o uso de técnicas de mineração de textos, neste trabalho o pré-processamento realizado foi feito pela ferramenta PreTexT, permitindo que as mensagens em língua portuguesa fossem representadas de acordo com a abordagem *bag-of-words* (BOW) mostrado na Sub-seção 2.2.2. Além disso, pode-se ajustar parâmetros que controlam problemas como o da dimensionalidade, ponderação dos atributos, normalização de valores da matriz e desbalanceamento do conjunto de dados [31].

Uma vez obtida a matriz do conjunto de dados, é possível realizar experimentos no Weka, a fim de verificar os melhores ajustes para os algoritmos mencionados. Como por exemplo, a utilização de *kernel* linear, polinomial e até outros tipos de *kernel* para o algoritmo SMO.

1.2 Apresentação do Problema

Um estudo conduzido por [34], realizado no CTIR Gov, utilizou uma abordagem de modelagem de processos para avaliar a situação corrente do centro e propor melhorias de gestão. Verificou-se que os processos adotados são semelhantes aos recomendados por [1];

mas embora parecidos, o centro possui uma equipe reduzida, em que os analistas técnicos possuem foco na resposta de incidentes, planejamento e realização de treinamentos para outras equipes de tratamento de incidentes de segurança da APF. Foi preciso adaptar esses processos para as peculiaridades e necessidades de pronta resposta exigidas pela comunidade da APF.

Durante o expediente, a equipe do CTIR Gov geralmente conta com apenas um analista responsável pela triagem e encaminhamento dos eventos ou incidentes. Os eventos chegam ao CTIR Gov por meio de *e-mails* que são categorizados por esse analista. Ele verifica mensagens novas nas caixas postais do centro, respondendo questões freqüentes ou designando-as aos integrantes do CTIR Gov. A designação de mensagens é organizada por categorias, cada qual com suas respectivas caixas de *e-mails*.

Algumas caixas são especiais, porque são categorizadas, utilizando-se filtros programados nos agentes de *e-mail*. Existem robôs e detectores que funcionam ininterruptamente na Internet, gerando *logs* que são enviados ao agente de transferência de *e-mails* do CTIR Gov. Essas mensagens são conhecidas internamente como Notificação Automática de Incidente (NAI). Cabe ao analista de triagem verificar as caixas dessas mensagens automáticas, averiguando anomalias, erros de classificação, análise de *logs*, entre outros.

A Figura 1.1 mostra uma análise de tendência nas comunicações do CTIR Gov. Note o aumento no número de notificações recebidas ao longo de 2009, entre janeiro a agosto, em que foram contabilizadas em média 280 notificações por mês totalizando, um total de 1920 *malwares* e 2250 notificações. Embora o crescimento seja pequeno, pode-se chegar a um limite de notificações inviável de ser triado por apenas uma pessoa. O gráfico supracitado não inclui meses após agosto, devido aos dados fornecidos pelo CTIR Gov.

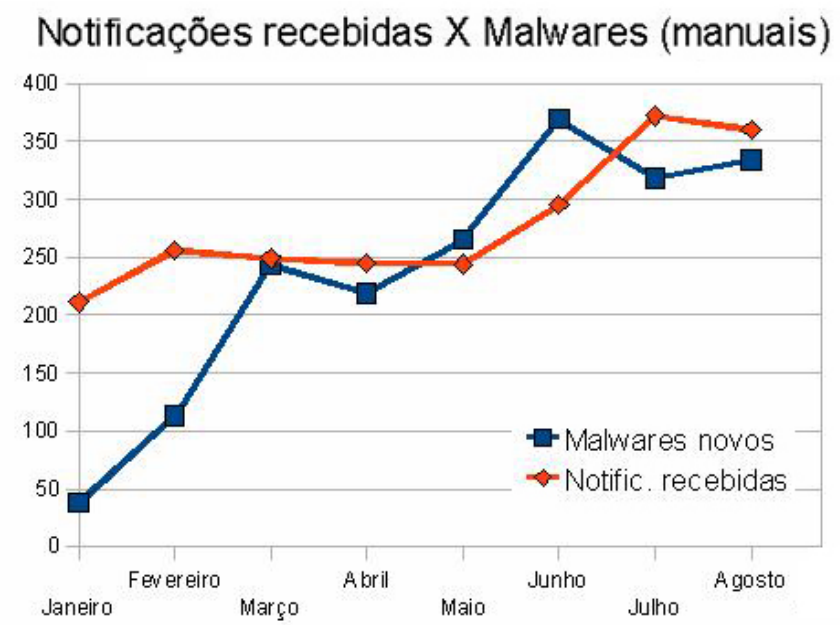

Figura 1.1: Gráfico de notificações recebidas e *malwares* no CTIR Gov.

1.3 Objetivos

O objetivo geral deste trabalho é definir e testar um processo de classificação supervisionada de mensagens de *e-mail*, na língua portuguesa, recebidas no CTIR Gov. As mensagens podem ou não estar formatadas, sendo que o melhor classificador escolhido deverá apresentar taxa de acerto semelhante à realizada pelo centro.

O objetivo secundário desta pesquisa é entender o funcionamento dos algoritmos e ferramentas utilizadas. Esse entendimento deverá ser utilizado na metodologia do trabalho e na análise dos resultados experimentais. Por fim, será adquirido um maior conhecimento sobre problemas da classificação supervisionada em textos da língua portuguesa.

1.4 Solução Adotada

Este trabalho propõe o uso de técnicas de mineração de textos para categorizar de forma automatizada mensagens de incidentes recebidas via *e-mail* pelo CTIR Gov. Como metodologia adotada, iniciamos o trabalho com o estudo dos conceitos relacionados à segurança, mais especificamente Gestão de Incidentes de Segurança e técnicas de mineração.

Por meio de várias entrevistas com integrantes do CTIR Gov, foram compreendidas as principais tarefas, prioridades e necessidades gerais da equipe de analistas técnicos. Para coleta de dados, foram feitos testes de utilização do sistema RTIR, *software open source* de rastreamento de incidentes na versão 2.4.1 [38].

O *software* foi instalado com sucesso em um servidor exclusivo, mas a organização interna da solução não permitiu uma extração simples das mensagens eletrônicas recebidas. Seria necessário averiguar todo o funcionamento do código-fonte para organizar o conjunto de mensagens eletrônicas de forma a permitir a utilização de uma ferramenta de tratamento de texto, como o PreTexT, por exemplo. O RTIR exige manutenção constante, porque à medida que os *e-mails* chegam, eles ficam acumulados nas filas de trabalho. Afim de que fosse possível utilizar o RTIR em coleta de dados para esta pesquisa, seria necessário mantê-lo funcionando adequadamente em paralelo com o modo de trabalho adotado até então no CTIR Gov.

Além desses fatores limitantes, seria necessário um esforço de integração não oportuno, tendo em vista que as mensagens eletrônicas são gravadas em um arquivo texto simples, com todas mensagens concatenadas no servidor de *e-mails*. O uso do RTIR foi abandonado e os dados foram obtidos diretamente do servidor de *e-mails* da equipe. O uso do RTIR não foi ideal para automatizar a categorização, pois seria necessário entrar em detalhes minuciosos do funcionamento da ferramenta.

O próximo passo foi escolher uma ferramenta capaz de pré-processar os textos de *e-mail*. A busca de uma solução capaz de tratar textos em português é bastante limitada. A utilização de radicalização na língua portuguesa não é difundida nas ferramentas de mineração de textos e, quando essa funcionalidade consta na solução, manipula apenas linguagens estrangeiras.

Foi realizada uma pesquisa para determinar ferramentas de processamento de textos na língua portuguesa, e o projeto PreTexT foi a solução encontrada para tratar esse problema. O PreTexT foi desenvolvido através de um projeto nacional (USP São Carlos), e, como uma solução acadêmica, tem livre utilização e publicação científica disponibilizada [31, 30], e portanto foi a ferramenta adotada neste trabalho.

Conforme o exposto, o modelo de solução proposto neste trabalho pode ser dividido de acordo com as etapas do processo de mineração, a saber:

- Coleta de Dados: Caixas postais com e-mails do CTIR Gov, no período de janeiro a agosto de 2009.
- Pré-Processamento: Ajuste dos parâmetros pela ferramenta PreTexT e obtenção da matriz BOW;
- Mineração de Textos: Aplicação dos algoritmos J48,*Naïve Bayes* e SMO, com parâmetros experimentais;
- Avaliação: Utilização de validação cruzada e *holdout*, porcentagem de instâncias corretamente classificada, *F-measure* e tempo de processamento.

1.5 Apresentação do Documento

O restante dos capítulos desta dissertação trazem o seguinte conteúdo:

- O Capítulo 2 apresentará uma visão geral sobre os temas de Gestão de Incidentes, mineração de dados e textos, algoritmos de classificação supervisionada e ferramentas de mineração;
- No Capítulo 3, será apresentado o estado da arte em classificação de textos e descrito o modelo de solução utilizado neste trabalho;
- No Capítulo 4, serão descritos os experimentos conduzidos e resultados obtidos para atingir o objetivo de classificação de mensagens eletrônicas. Serão realizados experimentos que avaliam possíveis parâmetros de mineração, com o pré-processamento dos dados utilizando os algoritmos estudados. Determinados esses parâmetros, os resultados anteriores foram utilizados em outro experimento para gerar um modelo capaz de classificar um conjunto de testes não tratados anteriormente;
- No Capítulo 5, serão apresentadas conclusões desta dissertação e propostas para trabalhos futuros, os quais puderam ser desenvolvidos por meio dos estudos realizados durante esta pesquisa.
- No Apêndice A, descreveremos os subprocessos do *framework Computer Security Incident Response Team* (CSIRT), incluindo os modelos de *workflow*.

Capítulo 2

Visão Geral das Áreas Envolvidas

Neste capítulo, serão descritos conceitos relacionados ao tratamento de incidentes de segurança e conceitos relacionados à mineração de dados e textos. Na Seção 2.1, serão abordados conceitos sobre CSIRT com a descrição geral do funcionamento de uma equipe de tratamento de incidentes. Definições básicas também serão apresentadas como a diferença entre eventos e incidentes, processos principais do *framework* CSIRT e relacionamentos entre áreas afins. Na Seção 2.2, será descrito o processo de mineração, com suas entradas, saídas e resultados, o entendimento dos dados, questões de qualidade, fase de pré-processamento de texto, redução de dimensionalidade, escolha dos atributos, desbalanceamento de dados, normalização e transformações necessárias para melhor representar os dados. Serão descritas técnicas de mineração especificamente relacionadas à classificação supervisionada, baseada em árvores de decisão, teoria de *Bayes*, SVM e SMO. Adicionalmente será apresentada uma visão geral sobre ferramentas de mineração de dados. Em seguida na Seção 2.3 uma breve apresentação de trabalhos correlatos.

2.1 Gestão de Incidentes de Segurança

O advento da Internet levou a humanidade a um fenômeno de comunidades em rede, acesso a meios de comunicação diversos, sendo um dos mais poderosos e disseminados meios de comunicação no planeta; sendo que cada vez mais a sociedade depende desse meio de comunicação para seus serviços.

O número de serviços disponíveis na Internet cresce a cada dia. Eles são conduzidos por entidades governamentais, empresas, bancos, escolas, hospitais e cidadãos em geral. Infelizmente, na maioria dos casos, esses serviços não são geridos, considerando o fator segurança da informação, e, como resultado, sistemas baseados na Internet ficam expostos a falhas de segurança comprometedoras. Essas falhas permeiam toda a arquitetura dos sistemas desde os protocolos de rede até as aplicações [44].

Problemas de segurança na Internet não são fenômenos contemporâneos, pois o primeiro grande incidente foi o *Internet Worm* ou *Morris Worm* que ocorreu em 1988. Esse incidente iniciou-se como um teste a sistemas Unix na ARPAnet. No entanto, o experimento saiu de controle e o artefato se alastrou para mais de 6 mil computadores, sendo que grande parte da rede operada pelo governo e universidades americanos ficou comprometida ou inoperante [33].

Logo após o incidente, foi conduzida uma reunião, objetivando identificar melhorias em respostas a incidentes de segurança. As recomendações resultantes incluíam a criação de um ponto de contato único para problemas de segurança na Internet. Esse ponto atuaria como um órgão confiável e centralizador de informações sobre segurança computacional na Internet. Nesta direção, surge o CERT/CC como uma das primeiras equipes para resposta a incidentes de segurança [44].

Na Sub-seção 2.1.1, será descrita a metodologia CSIRT da *Carnegie Mellon University* (CMU-CSIRT). O *framework* compreende três documentos principais: o manual descritivo sobre criação, operação e funções de CSIRTs, o manual do processo de tratamento de incidentes de segurança e os modelos organizacionais para CSIRTs [1, 44, 21].

Na literatura da área, existem outras metodologias, como descrito por [27], a qual mostra o tratamento de incidente de segurança com foco em análise forense. Autores como [22] também descrevem outros processos como descritos no Apêndice A. Os autores pesquisados apresentam muita semelhança em relação ao *framework* CSIRT, em que a maioria segue macro-processos comuns.

2.1.1 Conceitos Básicos

Um CSIRT representa a real capacidade de uma organização ou equipe de prover serviços e suporte para a organização ou outras equipes, com objetivos de prevenir, tratar e responder a incidentes computacionais de segurança. Os processos de um CSIRT envolvem atividades de detecção, triagem, análise, resposta, tratamento de vulnerabilidades, artefatos, coordenação, planejamento, treinamento, disponibilização de serviços pró-ativos e reativos, entre outros. Os conceitos apresentados nesta seção foram retirados de [1].

Para situar a conceitualização CSIRT, a Figura 2.1 mostra a relação entre resposta a incidentes, tratamento de incidentes e Gestão de Incidentes de Segurança. Cada um desses conceitos ocupam um contexto diferente, assim como uma divisão operacional, tática e estratégica. Os serviços operacionais de triagem, análise, detecção, divulgação, resposta a incidentes constituem o tratamento de incidentes, denominado também de serviço tático. A Gestão de Incidentes se insere em contexto mais amplo, como um serviço estratégico, que engloba serviços diversos como o tratamento de vulnerabilidades, artefatos, alertas, comunicações, entre outros.

A Figura 2.2 esclarece a distinção entre Gestão de Incidentes e Gestão de Segurança. Note que por motivos de simplificação a Gestão de Incidentes mencionada será sempre relativa à Gestão de Incidentes de Segurança. A Figura 2.2 mostra ainda a área de operações de Tecnologia da Informação (TI) e sua abrangência. Observe que há sobreposição de atividades entre a Gestão de Incidentes e Gestão de Segurança. Existem outras atividades que vão além do escopo definido pelas gestões de incidentes e segurança. A responsabilidade sobre essas atividades fica a cargo da área de Operações de TI.

Os setores circulares da Figura 2.2 demonstram a necessidade de coordenação e compartilhamento de informações, entre recursos organizacionais como as áreas de Assuntos Jurídicos, Relações Públicas, Recursos Humanos, Gestão de Riscos e demais. No centro da figura em formato de fechadura, os processos de Gestão de Incidentes (detectar, triar, responder, preparar, proteger) indicam a necessidade de canais de colaboração e comunicação estabelecidos com as outras áreas de interação.

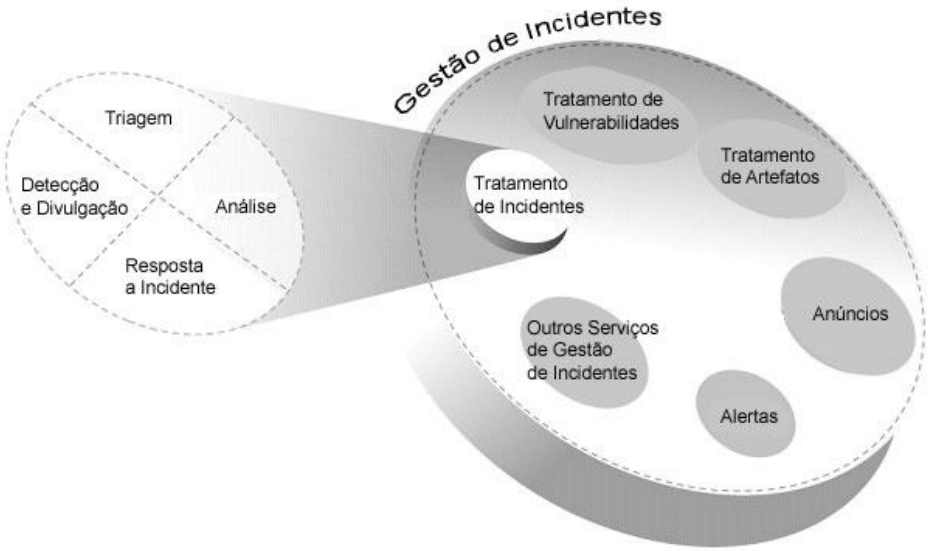

Figura 2.1: Relação entre Resposta a Incidentes, Tratamento de Incidentes e Gestão de Incidentes de Segurança. Adaptado de [1].

Outra definição importante a ser esclarecida está relacionada aos conceitos de incidente e evento. Evento é definido pelo *framework* CSIRT como uma ocorrência em um sistema que é relevante para a segurança, podendo se tratar ou não de incidente ou vulnerabilidade. Já a definição de incidente utilizada neste trabalho refere-se a qualquer evento adverso que infrinja a política de segurança adotada pela organização.

Em relação ao modelo organizacional relacionado à equipe CSIRT, há modelos organizacionais dos mais variados tipos, desde que existam formalização e definição de papéis, responsabilidades e serviços. A equipe pode ser um CSIRT interno, o qual trata incidentes de segurança, ou de coordenação, neste caso, coordena e facilita o tratamento de incidentes, vulnerabilidades e informações entre organizações internas e externas [1]. Além do modelo organizacional, as pessoas participantes de um CSIRT podem estar distribuídas ou centralizadas fisicamente [22].

A caracterização do CSIRT é fundamental para sua operacionalização, sendo que os serviços realizados por um CSIRT devem ser providos de acordo com a missão, objetivo e público alvo de uma determinada organização. Eles podem ser divididos em três categorias, que segundo descrição de [44] podem ser:

- serviços reativos - disparados por um evento ou pedido, referem-se ao recebimento de alertas e vulnerabilidades, tratamento de incidentes, tratamento de vulnerabilidades e artefatos, são componentes básicos de um CSIRT.

- serviços proativos - provê assistência e informação para auxiliar o preparo, proteção e segurança de sistemas, antecipando ataques, problemas ou eventos futuros, como exemplos podemos citar: anúncio de alertas, avaliações de segurança, configuração e manutenção de ferramentas de segurança, desenvolvimento de ferramentas de segurança, serviços de detecção de intrusão, disseminação de informações relacionadas à segurança da informação. Vale lembrar a importância no desempenho desses serviços, pois os mesmos poderão reduzir diretamente o número de futuros incidentes.

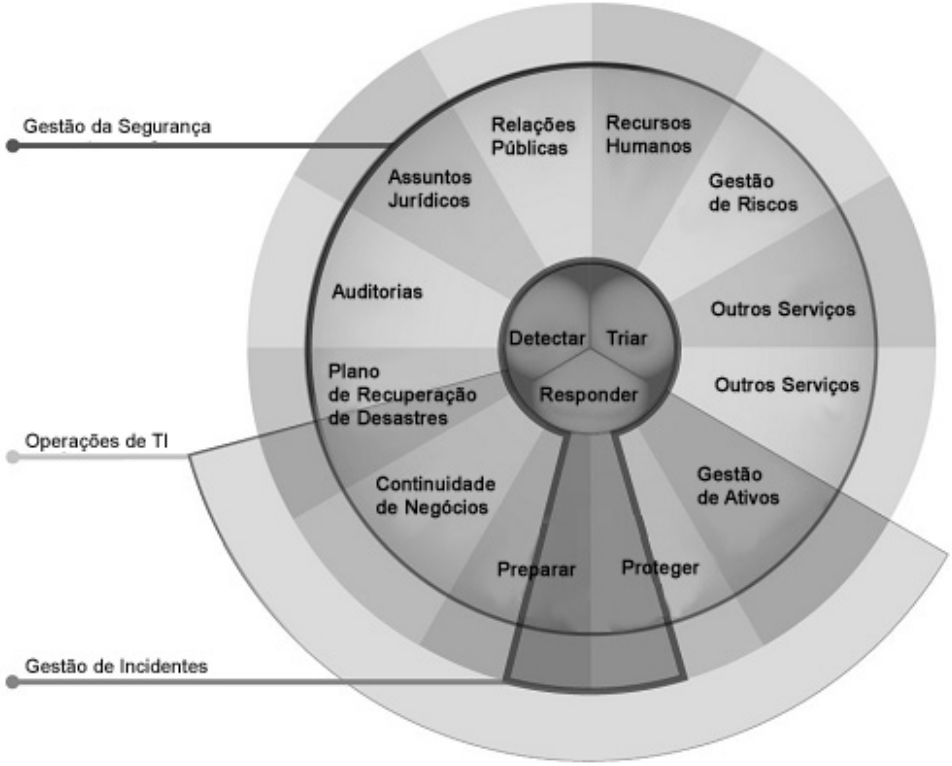

Figura 2.2: Sobreposição entre Gestão de Segurança, Gestão de Incidentes e Operações de TI (adaptado de 1).

- serviços de qualidade em gestão de segurança - atuam independentemente do tratamento de incidentes e provê uma análise própria sobre segurança da organização, identificando riscos, ameaças e vulnerabilidades em sistemas. Como exemplos podemos citar: análise de riscos, planejamento de continuidade de negócios e recuperação a desastres, consultoria em segurança, educação e treinamento, avaliação e certificação de produtos.

Segundo [1], as tarefas gerais de um centro de tratamento podem ser resumidas como:

- prevenir primeira ocorrência de incidentes e ataques, melhorando continuamente a segurança da infraestrutura computacional;
- treinar e educar a comunidade usuária acerca de questões de segurança da informação;
- monitorar e testar a infraestrutura, verificando fraquezas e vulnerabilidades;
- compartilhar informações com outras equipes quando apropriado.

Segundo [44] os processos de um CSIRT não podem ocorrer sem as funções de tratamento de incidentes, pois eles consistem em atividades que auxiliam a entrega de serviços, como a triagem, tratamento, comunicação de avisos e *feedback*, a saber:

- A função de triagem provê um ponto focal único para aceite, coleta, ordenação e transmissão das informações recebidas. A triagem pode ser considerada como um

canal em que as informações externas são passadas. Recebe entradas de canais diferentes que condizem com as necessidades da equipe de tratamento de incidentes. A qualquer evento novo devem ser dados uma prioridade e número de rastreamento. Atividades adicionais como arquivamento, tradução ou conversão de mídias podem ser realizadas para facilitar as atividades subseqüentes de tratamento de incidentes [44].

- a função de tratamento provê um suporte e consultoria para incidentes de segurança computacionais suspeitos ou confirmados. Uma revisão do laudo ou relato de incidente é realizada para determinar o que ocorreu e o tipo de atividade envolvida. Uma análise do relato pode incluir revisão das evidências ou materiais a favor do fato, com o objetivo de identificar os envolvidos ou contatos de auxílio providos. A equipe necessitará identificar respostas apropriadas e redigir a notificação para os envolvidos.

- a função de avisos gera informações sobre ataques recentes, documentos sobre proteção contra ameaças e informações sobre tendências no escopo e natureza dos ataques reportados. A aplicação dessa função pode ultrapassar o escopo de sua aplicabilidade no tratamento de incidentes, podendo ser fonte de informações para outros serviços como análise de vulnerabilidades e artefatos.

- a função de *feedback* provê suporte para o retorno de questões não especificamente relacionadas com incidentes isolados. A função pode ocorrer, utilizando pedidos explícitos, como a solicitação de um interessado por alguma questão, implícitos, como um relatório rotineiro anual ou por caso, proativamente comunicando os interessados. A função proverá um conjunto mínimo de suporte para questões freqüentes e também poderá ser visto como um canal de comunicação.

2.1.2 Processos Relacionados

Os processos básicos estão divididos em cinco subprocessos. No tratamento de incidentes, a detecção, triagem e resposta representam a parte central, restando apenas dois processos estratégicos, que atuam na melhoria contínua da qualidade do *framework*: a proteção da infraestrutura e o processo de preparar, manter e melhorar um CSIRT.

A interação entre os processos de um centro pode ser melhor entendida na Figura 2.3. Ela apresenta os processos de Gestão de Incidentes de Segurança mostrados em alto nível: preparar CSIRT refere-se ao processo preparar, manter, melhorar CSIRT; proteger ou proteger infraestrutura; detectar; triar e responder. Na Figura 2.3, as setas maiores indicam a evolução de incidentes no tempo, enquanto as setas menores indicam troca de informações entre subprocessos. Note que nem sempre todos esses processos são aplicáveis em um CSIRT, sendo apenas sugeridos a adoção dos mesmos [1].

Uma descrição detalhada dos processos e subprocessos de um CSIRT são apresentados no Apêndice A, os quais foram traduzidos de [1]. Atenção especial deverá ser dada ao processo de triagem por ser este o foco do trabalho apresentado nas Figuras A.5 e A.6.

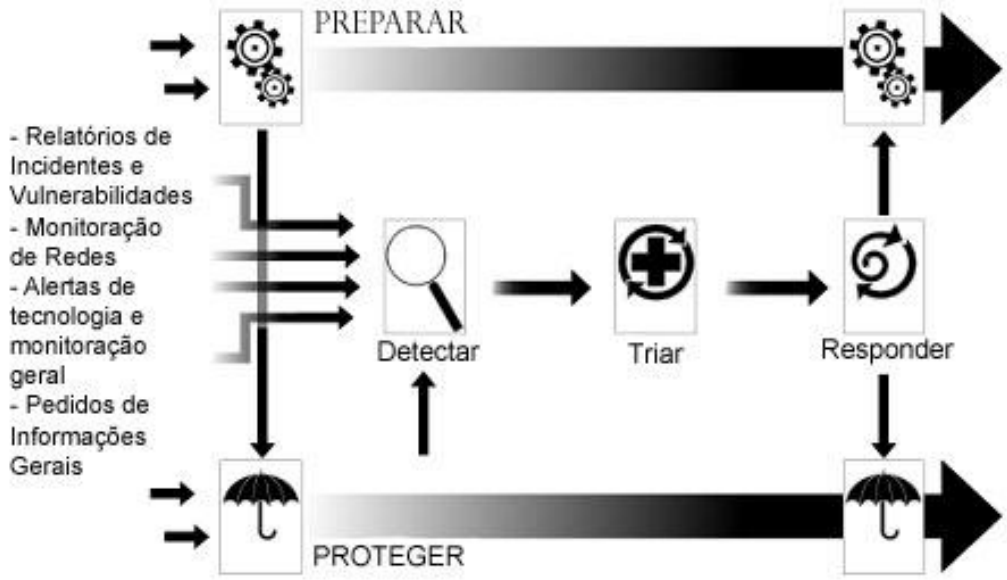

Figura 2.3: Processos de um CSIRT mostrados em alto-nível (Ilustração adaptada de 1)

2.1.3 Ferramentas de Apoio aos Processos CSIRT

Os serviços prestados por um CSIRT podem ser melhor realizados, utilizando-se um sistema capaz de gerir documentos e informações durante o ciclo de vida de um incidente. Alguns centros possuem bancos de dados, registros de trabalho sobre incidentes, formulários, documentos, repositórios, entre outros recursos para realizar as tarefas pertinentes ao CSIRT.

Para melhor administrar os processos, existem sistemas automatizados específicos para tratar a coleta de informações relacionados ao fluxo de trabalho. Sistemas de rastreamento de incidentes permitem registro, rastreamento, tratamento e divulgação de incidentes dentro de um fluxo de trabalho definido [44].

Nos sistemas de rastreamento, são registradas informações relativas aos atores envolvidos, às organizações e pessoas, aos papéis desempenhados por essas pessoas, à natureza do incidente, ao escopo do incidente, ao relato temporal, entre outros. Os sistemas permitem melhor organização das atividades e serviços realizados [10].

Segundo [10], os sistemas de rastreamento devem adotar o modelo *Incident Object Description and Exchange Format* (IODEF). Esse padrão de formatos de dados serve para comunicação de incidentes. A descrição do objeto é feita por meio de classes e subclasses e foi concebida com o propósito de definir uma formatação de dados comum e procedimentos de troca padronizados a fim de compartilhar informações necessárias para tratamento de incidentes entre diferentes CSIRT [13, 42].

Segundo [22], uma pesquisa foi realizada sobre como as informações recebidas eram reunidas em um CSIRT, sendo que 76% das equipes utilizam banco de dados para registrar dados de incidentes; 28% utilizam um banco de dados e registro em papel; 10% utilizam apenas registro em papel; 45% utilizam banco de dados adaptados e 28% utilizam uma ferramenta comercial. As ferramentas mais comuns reportadas foram:

- *Remedy Help Desk* e *Action Request System* (`http://www.bmc.com/`),

- MySQL (`http://www.mysql.com/`),
- Oracle (`http://www.oracle.com`),
- Microsoft Access (`http://office.microsoft.com/Access`),
- Lotus Notes (`http://www-01.ibm.com/software/lotus/`),
- RTIR (`http://bestpractical.com/rtir/`),
- CERIAS (`https://cirdb.cerias.purdue.edu/`).

Dependendo do tamanho do centro de tratamento de incidentes, deve ser significativo a utilização de tais sistemas, pois eles auxiliam os processos de um CSIRT para registro e documentação de eventos e incidentes. Como exemplos de soluções gratuitas podemo citar: CERIAS *Incident Response Database* - CIRDB (disponível em `https://cirdb.cerias.purdue.edu/`, *Request Tracker for Incident Response* - RTIR (disponível em `http://bestpractical.com/rt/`), *Incident Management for CERT and IT Security Teams* - SIRIOS (disponível em `http://www.sirios.org/`) que adota o padrão IODEF.

Dentre as opções de *software* livre, o sistema RTIR se mostra como opção elegível, tendo sido construído para corresponder às necessidade de um CSIRT para tratar fluxos de trabalho, registros e comunicações. O RTIR possui como fundamento básico o conceito de bilhete, em que cada evento recebe um número único e representa um pedido. O sistema é multi-usuário, permite trabalho colaborativo, registra histórico em cada bilhete, realiza notificações via *e-mail* e possui um fluxo de trabalho adaptável [43].

Resumidamente, o RTIR necessita de um servidor *Web* com Apache, diversos módulos do Perl, banco de dados MySQL, Postgre, Oracle ou SQLite e um *Mail Transfer Agent*. Após instalação de cada módulo, é necessário configurar cada elemento para que possa funcionar corretamente com o sistema RTIR. Basicamente, o RTIR foi desenvolvido como um acessório ao sistema *Request Tracker* (RT). No RT, consta todo o motor do sistema de rastreamento de bilhetes com notificações, formulários e um fluxo de trabalho adaptado ao *framework* CSIRT. O suporte para o padrão IODEF não é disponibilizado [38].

2.2 Conceitos de Mineração

Nesta seção, apresentaremos conceitos básicos de mineração e dados e mineração de textos os quais foram utilizados neste trabalho.

2.2.1 Mineração de Dados

A mineração de dados pode ser definida como a análise de um conjunto de dados observacional e volumoso para descobrir relacionamentos e sumarizar dados de forma inédita de modo que sejam inteligíveis e úteis àquele que os utiliza [19].

Segundo [18], mineração de dados é a tarefa de descoberta de padrões em grandes quantidades de dados, onde os dados podem ser armazenados em bancos de dados, *data warehouses*, ou repositórios de informações. A mineração de dados é uma área interdisciplinar, que agrupa áreas como sistemas de bancos de dados, estatística, aprendizagem por máquina, visualização de dados, busca de informações e computação de alto desempenho.

Mais ainda, de acordo com [25], a mineração de dados é uma tecnologia que mistura métodos tradicionais de análise de dados, com algoritmos sofisticados, para processar grandes volumes de dados. Essa área de conhecimento tem aberto novas possibilidades para explorar e analisar dados, para descoberta de padrões, predição e novas informações.

Relacionada aos aspectos de tratamento de informação e de conhecimento, a mineração de dados está inserida no processo de descoberta de conhecimento.

Segundo [41], os passos básicos na descoberta de conhecimento em mineração são: (i) entrada de dados; (ii) pré-processamento dos dados, incluindo seleção de características; (iii) redução da dimensionalidade; (iv) normalização e seleção do sub-conjunto de dados; (v) mineração de dados, utilização de algoritmos de mineração; (vi) pós-processamento, filtragem de padrões, visualização, interpretação de padrões; e finalmente (vii) obtenção de informações.

Os problemas na mineração de dados podem ser dos mais diversos tipos. Os diversos problemas podem ser agrupados em classes básicas no processo de descoberta de conhecimento. Essas classes de problemas em mineração são chamadas de tarefas. As tarefas são geralmente divididas em duas partes [41]:

- tarefas de predição — envolvem prever um valor para um atributo particular baseado nos valores de outros atributos. O atributo a ser descoberto é chamado de alvo ou variável dependente, enquanto atributos utilizados para fazer predição são chamados de explicativos ou variáveis independentes. A meta é a construção de um modelo para a variável alvo como uma função das variáveis explicativas. A modelagem pode ser a classificação ou regressão. A classificação é utilizada em variáveis alvo discretas. A regressão é utilizada com variáveis alvo contínuas. O objetivo dos dois tipos é aprender um modelo que minimiza o erro entre o valor real e o predito de uma variável alvo. Segundo [45] são exemplos de algoritmos classificadores: *Bayesnet*, ID3, J48, *Naïve Bayes*, SMO, entre outros.

- tarefas de descrição — estão relacionadas à derivação de padrões, correlações, tendências, agrupamentos, trajetórias e anomalias, que resume os relacionamentos nos dados. Essas tarefas são de natureza exploratória e frequentemente necessitam de técnicas de pós-processamento para validar e explicar os resultados obtidos. Segundo [45], são exemplos de algoritmos de associação: Apriori, *Predictive Apriori* e *Tertius*.

Segundo [41], os dois grupos básicos de tarefas de mineração de dados podem ser didaticamente estruturados em: classificação, agrupamento, associações e detecção de anomalias. A classificação ordena os dados em categorias pré-definidas. Já no agrupamento o objetivo é o mesmo, entretanto não existem categorias pré-definidas. A associação busca relações entre variáveis nos dados e a detecção de anomalias objetiva achar pontos que não se encaixam na modelagem funcional dos dados.

A análise de associação é utilizada para descobrir padrões que descrevem fortemente características associadas aos dados. Os padrões descobertos são tipicamente representados na forma de regras de implicação ou subconjuntos de características. Devido ao tamanho de espaço de busca exponencial, o objetivo da análise de associação é extrair padrões mais notáveis de forma eficiente, por exemplo, a descoberta de grupos de genes que possuem funcionalidades correlatas, a identificação de páginas sendo acessadas simultaneamente, entre outros [41].

A análise de agrupamento busca achar grupos de observações fortemente relacionadas. Desta forma, as observações que pertencem ao mesmo agrupamento são mais parecidas entre si, quando comparadas a outros agrupamentos. Um exemplo de agrupamento é a utilização para identificar conjuntos de clientes similares.

A detecção de anomalias é a identificação de observações que diferem do resto dos dados. Essas observações são conhecidas como anomalias ou discrepâncias. O objetivo da detecção de anomalias é descobrir anomalias reais e evitar rotular objetos normais como anômalos. Um bom detector de anomalia deve ter uma alta taxa de alarmes positivos e uma baixa taxa de alarmes falsos. São exemplos de aplicação em fraudes, redes de intrusos, padrões incomuns de doenças, entre outros.

2.2.2 Mineração de Textos

Segundo [14], a capacidade de armazenamento na era da informação tem aumentado de forma impressionante. A proliferação de documentos disponíveis na Internet e o volume de *e-mails* têm crescido de forma acentuada. Enquanto a quantidade de dados disponíveis tem aumentado, a nossa habilidade de absorver e processar essas informações têm se mantido constante. A mineração de textos tenta solucionar o problema de sobrecarga de informações, utilizando conhecimento das áreas de mineração de dados, aprendizado por máquina, processamento de linguagem natural, recuperação de informações e gestão do conhecimento.

Mineração de textos envolve o pré-processamento de coleções de documentos, com a categorização de textos, extração de palavras e informações, bem como o armazenamento de representações intermediárias, técnicas de análise, classificação, agrupamentos, análise de tendências, associações e visualização de resultados.

Um elemento chave para mineração é a coleção de documentos a qual é definida como um conjunto de documentos textuais. As coleções podem ser estáticas, quando os conjuntos permanecem inalterados, ou dinâmicas, em que os conjuntos são caracterizados por inclusão ou atualização de documentos ao longo do tempo.

O documento pode ser definido informalmente como uma unidade discreta de dados textuais. Um documento pode ser visto como um objeto estruturado. De uma perspectiva linguística, qualquer documento possui riqueza sintática e semântica em suas estruturas. Além disso, sinais de pontuação servem como uma marcação linguística, que provê dicas para identificar subcomponentes importantes no documento como parágrafos, título, autores, datas de publicação, entre outros [14].

Para realizar as tarefas propostas pelo tratamento de linguagem, os documentos necessitam ser transformados, em uma etapa crucial que é o pré-processamento. O pré-processamento envolve rotinas, processos e métodos que preparam os documentos existentes para as operações de mineração. As informações são convertidas em um formato canônico, em que as operações de mineração possam ser realizadas. O processo de mineração de textos é muito parecido com o processo usual de mineração de dados, sendo que inicialmente há uma busca de documentos. Seguida de tarefas de pré-processamento, aplicação de algoritmos de mineração, refinamento dos resultados e avaliação [14].

A qualidade dos dados é uma questão importante em aplicações de mineração, a qual precisa ser cuidadosamente manipulada para evitar problemas futuros. Técnicas de mineração têm especial atenção na detecção e correção de problemas de qualidade

nos dados, as quais são denominadas limpeza de dados. Além disso, utilizam algoritmos tolerantes à baixa qualidade de dados [41]. A qualidade pode ser verificada pelo uso de textos com vocabulário estruturado, pontuações reconhecidas e padronização. A limpeza de dados contempla casos de ortografia incorreta, uso de acrônimos, abreviações, símbolos misturados a textos, anexos, estrutura gramatical incorreta, entre outros. Esses aspectos principalmente ocorrem com textos gerados manualmente.

Os conceitos mais importantes sobre a qualidade de dados estão relacionados às medições utilizadas como fontes de dados. Geralmente os erros são inerentes, problemas de acurácia, precisão, ruídos, valores discrepantes, valores inexistentes, inconsistentes, duplicados, entre outros [41]. Quando os dados são textos, a dificuldade está na determinação semântica exata de cada palavra utilizada em um texto.

Pré-Processamento de Texto

Segundo [41], o pré-processamento consiste no uso de técnicas e estratégias para selecionar ou criar atributos. O conceito pode ser estendido não só para atributos, mas também para objetos de dados. Como exemplos de técnicas de pré-processamento podemos citar: amostragem, agregação, redução de dimensionalidade, seleção do subconjunto de atributos, criação de atributo, discretização, transformação de variáveis, entre outros.

O problema de pré-processamento compreende as etapas de preparação, tarefas de processamento de linguagem natural e tarefas relativas ao problema a ser resolvido. A preparação converte a representação original em uma estrutura própria para processamento lingüístico. As tarefas de processamento de linguagem envolvem *tokenização*, análise morfológica, análise sintática, entre outros [14].

Segundo [41], a amostragem é geralmente utilizada na seleção dos subconjuntos de objetos de dados a serem analisados. Essa abordagem é utilizada na mineração porque geralmente há um custo alto no processamento de todos os dados. O princípio básico para amostragem é que o subconjunto escolhido seja representativo. Isso significa que a amostra deve ter as mesmas propriedades do conjunto de dados original.

Dependendo da heterogeneidade dos dados, a definição de característica representativa pode variar. A escolha se torna probabilística, em que a amostra possui provável chance de ter características representativas. A estratégia de amostragem pode ser randômica, sem ou com reposição, estratificada, em que se selecionam freqüências variadas para os itens de interesse, progressiva ou adaptativa, a qual se aumenta o tamanho da amostra até que se chegue a uma condição insatisfatória como, por exemplo, aumento da taxa de erros.

A amostragem também pode ser utilizada em casos de variáveis alvo com número de instâncias desbalanceadas na classificação supervisionada. A idéia principal é modificar as distribuições originais para que classes com poucos exemplos sejam bem representadas no conjunto de dados. O caso de *undersampling* consiste na amostragem de um subconjunto de um certo rótulo. A desvantagem é a possibilidade de serem amostrados casos pouco relevantes que prejudiquem a construção do modelo classificador. No caso de *oversampling*, as instâncias de um rótulo são replicadas até que ocorra uma equalização entre as instâncias dos demais rótulos. A desvantagem no *oversampling* está na indução de um modelo classificador super-ajustado por causa da replicação de ruídos no

conjunto de dados. Métodos que combinam ambas técnicas também podem ser adotados no pré-processamento.

A redução da dimensionalidade apresenta benefícios na construção de modelos preditores. Quanto maior a dimensão, ou seja, o número de atributos, maior a dificuldade de entendimento dos dados, maior número de ruídos, maior é o custo de processamento e o tempo gasto. A redução de dimensionalidade pode ser obtida pela aplicação de técnicas que dependem do tipo de dados utilizado como fonte. Por exemplo, em dados numéricos contínuos, utilizam-se técnicas como *Principal Component Analysis* (PCA) e *Singular Value Decomposition* (SVD).

A seleção do subconjunto de atributos é uma forma de redução de dimensionalidade, sendo possível onde existem atributos redundantes ou irrelevantes e não conduz para perda de informações. Essa decisão de seleção depende fortemente da aplicação do senso comum e conhecimento do domínio, existindo três abordagens mais comuns: (i) a embutida, em que a seleção de atributos ocorre como parte do algoritmo de mineração; (ii) a abordagem de filtro, em que os atributos são escolhidos antes da execução do algoritmo de mineração; e finalmente, (iii) a abordagem de cobertura enxerga o algoritmo de mineração como uma caixa preta, buscando um subconjunto com número ótimo de atributos [41].

Segundo [14], o pré-processamento pode ser realizado de acordo com a tarefa de mineração ou o problema envolvido. No pré-processamento orientado a tarefa, cria-se uma representação estruturada de um documento envolvendo uma preparação ou problema que necessita ser resolvido, como por exemplo, a extração de palavras-chave em um texto. Outras abordagens levam em conta técnicas que derivam de métodos formais para analisar fenômenos complexos, que podem ser também aplicados a textos em linguagem natural. Essas abordagens incluem esquemas de classificação, modelos probabilísticos e sistemas baseados em regras.

O pré-processamento geralmente inicia-se com um documento parcialmente estruturado em que os atributos são refinados ou adicionados, enriquecendo a estrutura de representação do documento. Os atributos relevantes são utilizados para mineração de textos, enquanto a outra parte é descartada. Essa exclusão utiliza técnicas de redução de dimensionalidade e retirada de palavras irrelevantes ou *stop words*.

Algoritmos de classificação não podem tratar um texto diretamente. Faz-se necessária uma conversão, na qual os documentos possam ser manipulados. A representação dos documentos pode ser realizada pela estruturação de uma tabela composta por vetores de palavras, que ocorrem no documento analisado. As linhas da tabela representam os documentos e as colunas representam os atributos ou palavras, sendo a última coluna utilizada para a variável alvo, como por exemplo, o rótulo de cada documento [31].

Abordagens de Pré-Processamento

A abordagem de uso de uma tabela composta refere-se à técnica BOW. Nesta técnica, cada atributo referenciado no documento recebe um valor numérico, sendo que nem sempre todos atributos ocorrem em um texto, gerando uma matriz esparsa. Na representação mencionada, ocorre perda de informações, pois a ordem das palavras é ignorada, com prejuízo semântico. No entanto, a simplicidade de construção e a facilidade para busca de informações são pontos positivos ao se utilizar a abordagem BOW [14].

Segundo [31], a abordagem BOW representa os textos como vetores de atributos, também chamada de modelagem *vector space model*. Seja uma coleção de N documentos $D = \{d_1, d_2, ..., d_N\}$, em que d_i é um conjunto de palavras, ou seja $d_i = \{t_1, t_2, ..., t_M\}$, em que t_k é uma palavra do documento e M é o número de palavras contidas em D. Dado que exista um conjunto de N_{cl} categorias $C = \{c_1, c_2, ..., c_{N_{cl}}\}$, em que c_j é uma categoria que atribui um rótulo para um dado d_i. Vale ressaltar que $N_{cl} \leq N$ e de preferência $N_{cl} \ll N$. A representação de documentos pode ser vista por meio da Tabela 2.1.

	t_1	t_2	...	t_M	C
d_1	a_{11}	a_{12}	...	a_{1M}	c_1
d_2	a_{21}	a_{22}	...	a_{2M}	c_2
...	...	...	...	...	...
d_N	a_{N1}	a_{N2}	...	a_{NM}	$c_{N_{cl}}$

Tabela 2.1: Representação *bag-of-words* [31].

Em relação à contagem dos elementos a_{ij} que representam as palavras de cada documento, os mesmos podem ser obtidos, utilizando *term frequency* (TF), *term frequency inverse document frequency* (TFIDF), entre outros. O método TF considera a contagem do termo no documento. A medida TF é definida na Equação 2.1, em que $freq(t_j, d_i)$ é a freqüência absoluta do termo t_j no documento d_i.

$$TF(t_j, d_i) = freq(t_j, d_i) \tag{2.1}$$

Muitas vezes um termo bastante frequente não é considerado como representativo numa coleção de documentos. Neste caso o método que considera esse fator é o método TFIDF. Termos muito freqüentes na coleção de documentos recebem menor importância, daí seu nome freqüência inversa. O fator de ponderação é definido pela Equação 2.2 em que $freq(t_i, d_j)$ é a freqüência absoluta do termo t_i no documento d_j. O fator de ponderação é definido pela Equação 2.3, em que N é o número total de documentos da coleção e $d(t_j)$ é o número de documentos em que o termo t_j ocorre [31].

$$TFIDF(t_j, d_i) = freq(t_i, d_j) \cdot IDF(t_j) \tag{2.2}$$

$$IDF(t_j) = log \frac{N}{d(t_j)} \tag{2.3}$$

O método proposto por [31] propõe a utilização de uma equação IDF que não enfatize tanto termos pouco freqüentes em poucos documentos. A função é definida pelas Equações 2.4 e 2.5.

$$TFLINEAR(t_j, d_i) = freq(t_i, d_j) \cdot LINEAR(t_j) \tag{2.4}$$

$$LINEAR(t_j) = 1 - \frac{d(t_j)}{N} \tag{2.5}$$

Além dos métodos de representação, os algoritmos de radicalização, através das curvas de Zipf e cortes de Luhn (Zipf e Luhn apud [31], p. 13), se propõem a reduzir a

dimensionalidade, buscando palavras relevantes no conjunto de dados amostrado, retirando atributos não essenciais da amostragem.

Os algoritmos de radicalização auxiliam na redução de dimensionalidade, uma vez que normalizam lingüisticamente as variações de uma mesma palavra, em que são removidos prefixos e sufixos, deixando apenas a raiz do termo. Os algoritmos dependem fortemente da linguagem utilizada, como por exemplo o algoritmo de Porter para língua inglesa. Esse algoritmo também pode ser adaptado para o caso da língua portuguesa. A eficiência do algoritmo pode degradar em casos em que se busca minimizar o número de *stems* diferentes para termos com mesmo radical.

Uma forma alternativa para redução da dimensionalidade é procurar termos mais representativos nos documentos. A lei de Zipf ocorre em documentos que mostram um gráfico decrescente para um histograma de freqüência das palavras. Luhn utilizou essa hipótese para mostrar que existem pontos de corte para excluir termos desprezíveis, sendo que os termos mais representativos se situam entre os pontos de corte, especificamente na região de um pico imaginário. A Figura 2.4 mostra o histograma mencionado, e a definição dos cortes baseados na curva imaginária.

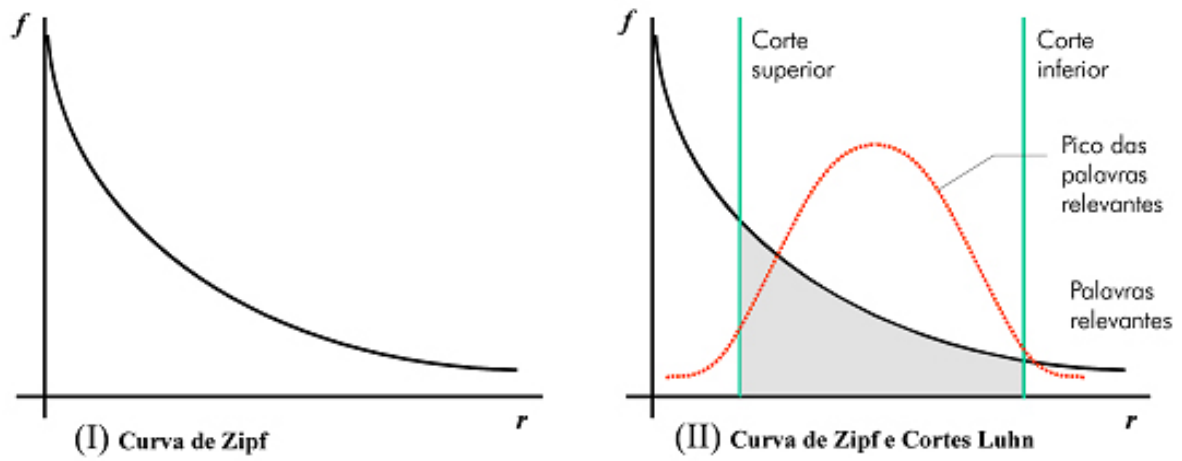

Figura 2.4: Curva de redução de dimensionalidade de termos em documentos [31].

2.2.3 Algoritmos de Classificação

Classificação pode ser definida como a tarefa de designar objetos a uma das categorias pré-definidas. No processo de classificação, faz-se necessário realizar um aprendizado do mapeamento entre um conjunto de atributos de entrada x e seu rótulo y. A entrada é uma coleção de atributos caracterizada por uma tupla (x, y), em que x é um conjunto de atributos e y é uma classe discreta de atributos chamada de rótulo, categoria ou atributo alvo [41].

No caso binário, o conjunto y possui apenas dois elementos: categoria positiva ou negativa. No caso multi-classe, pode existir n categorias diferentes $y_i \in \{1, ..., n\}$. No caso *multi-label*, cada registro pode pertencer a um ou mais rótulos, ou seja, $y_i \in 2^{\{1,...,n\}}$, ou seja, o conjunto das partes de y_i. A função alvo f, ou modelo classificador, é responsável por atribuir ao conjunto x um rótulo do conjunto y. Os casos multi-classe e *multi-label*

são na prática reduzidos a vários problemas binários, i.e. se pertence ou não a cada rótulo [32].

Segundo [41], o modelo classificador pode ser utilizado para as finalidades de modelagem descritiva e preditiva. No caso descritivo, busca-se um modelo capaz de resumir grandes quantidades de dados, além de explicar quais atributos definem cada classificação do conjunto de rótulos. No caso dos modelos preditivos, a classificação é utilizada para predizer o rótulo de registros desconhecidos.

O algoritmo de aprendizado identifica o modelo que melhor se ajusta à relação entre conjunto de atributos e rótulos. O modelo gerado deverá ser capaz de adaptar-se aos dados de entrada e também predizer a classificação correta de registros desconhecidos. O objetivo é criar modelos com boa capacidade de generalização, em que os modelos são obtidos, fornecendo conjuntos de treinamento. Durante o treinamento, um modelo classificador é induzido. Para avaliar a eficiência de classificação, um conjunto de testes em que os rótulos são desconhecidos é fornecido ao modelo.

O teste do modelo obtido pode feito com conjunto de dados criado separadamente. Caso não seja possível essa estratégia, o conjunto de treinamento pode ser utilizado. As estratégias mais comuns são o *holdout* e a validação cruzada em n dobras. O método *holdout* divide o conjunto de treinamento em duas partições proporcionais a uma dada porcentagem. Uma partição é utilizada para treinamento e a outra para testes. Na validação cruzada, o conjunto de dados é dividido em n partições (dobras) de tamanhos iguais. Em uma partição é definido o conjunto de testes. As $n-1$ partições são utilizadas para definir o conjunto de treinamento. Em seguida, uma outra partição diferente da anterior é escolhida para testes, sendo as demais $n-1$ partições para treinamento. Esse procedimento é repetido n vezes de modo que cada partição seja testada pelo menos uma vez.

Algoritmo J48

O algoritmo J48 é um algoritmo de árvores de decisão capaz de lidar com atributos numéricos, valores inexistentes e dados com ruídos. Ele particiona o problema em subproblemas menores que são iguais aos problemas originais. Os subproblemas são resolvidos recursivamente e combinados para obter uma solução do problema original. O problema é definido pela busca de árvores de decisão capazes de separar os dados nos rótulos conhecidos [45].

Segundo [41], a árvore de decisão é definida por seus nós (raiz, internos e terminais) e suas arestas. Os nós representam os atributos, e as arestas são os valores possíveis para cada atributo. O nó raiz representa um dado atributo escolhido por teste de condição, os nós folhas representam os rótulos e os nós internos são os demais atributos. Em princípio, a busca da árvore ótima é impossível visto o espaço de busca de tamanho exponencial. Para se obter uma árvore, geralmente os algoritmos empregam estratégias que particionam os dados de forma recursiva, para manter subconjunto de dados mais puros, ou seja, conjuntos de atributos de mesmo rótulo em que podem ser usados para treinamento . Seja D_t o conjunto de treinamento associado ao nó t e seja $y = \{y_1, y_2, \cdots, y_c\}$ o conjunto de rótulos:

- **Passo 1:** Se todos os registros de D_t pertencem à mesma classe y_t, então t é o nó folha rotulado de y_t.

- **Passo 2:** Se D_t contém registros que pertencem a mais de uma classe, uma condição de teste de atributo (medidas de impureza como entropia, erro e índice *Gini*) é realizada para particionar os registros em sub-conjuntos menores. Um nó filho é criado para cada resultado da condição de testes, e os registros D_t são distribuídos aos filhos baseados nos seus resultados. O algoritmo é recursivamente aplicado para cada nó filho.

Uma vez que a árvore de decisão está construída, a etapa de corte chamada de *tree-pruning* pode ser realizada para reduzir o tamanho da árvore. Geralmente árvores muito grandes estão suscetíveis a fenômenos como o super-ajustamento (*overfitting*). O método de *pruning* corta certos ramos da árvore inicial de forma que melhore a capacidade de generalização da árvore. Questões de super-ajustamento serão tratadas nas próximas seções.

Segundo [45], existem três aspectos importantes a serem considerados no algoritmo de árvores de decisão: os métodos de separação utilizados em atributos numéricos (geração das arestas), robustez com dados inexistentes, e aplicação das técnicas de *pruning* (*postpruning* e *prepruning*) [45].

A complexidade desses algoritmos pode ser estimada por $O(mn \log n) + O(n(\log n)^2)$, uma vez que os dados contêm n instâncias e m atributos. O resultado é obtido por considerar o tamanho da árvore ($O(\log n)$), o custo computacional de construção da árvore ($O(mn \log n)$), as operações de *prunning* ($O(n)$ e $O(n(\log n)^2)$) [45].

[41] apresenta algumas características que resumem os algoritmos de árvores de decisão:

- A indução de árvores de decisão ocorre sem necessidade de compreensão das distribuições probabilísticas de cada rótulo;
- Classificação rápida do conjunto de testes visto que a complexidade de pior caso é $O(w)$, em que w é a profundidade da árvore;
- Robusto diante presença de dados com ruídos, provê métodos para evitar *overfitting*;
- Dados com margem de decisão em que é necessário avaliar dois atributos ao mesmo tempo, não são favoráveis para o algoritmo de árvores de decisão. O algoritmo se especializa em particionar o espaço de atributos em regiões disjuntas até que cada região tenha apenas registros de mesmo rótulo.

Algoritmo *Naïve Bayes*

O classificador *Naïve Bayes* é um modelo classificador probabilístico que utiliza o teorema de Bayes para casos em que os atributos são condicionalmente independentes entre si. O classificador emprega todos os atributos, pois considera que os atributos são igualmente importantes e independentes entre si, dada uma classe. Trata-se de um método simples que provê resultados importantes na prática [45].

Segundo [41], o teorema de Bayes sustenta que, dadas uma hipótese Y e uma evidência X que se sustenta nessa hipótese, podemos tratar essas variáveis como aleatórias e capturar suas relações probabilísticas utilizando a Equação 2.6.

$$P(Y|\mathbf{X}) = \frac{P(\mathbf{X}|Y)P(Y)}{P(\mathbf{X})} \tag{2.6}$$

O teorema pode ser utilizado para solucionar problemas de predição da seguinte forma: seja X o conjunto de atributos e Y a variável de rótulos; se a variável alvo possui uma relação não-determinística com os atributos, então podemos tratar X e Y como variáveis randômicas, capturando a relação probabilística e utilizando $P(Y|X)$ (chamada de probabilidade posterior). Durante a fase de treinamento são aprendidas as probabilidades posteriores para cada combinação de X e Y baseadas nos dados de treinamento. Quando utilizado o cálculo dessas probabilidades, um registro de teste X' pode ser classificado, buscando o rótulo Y' que maximiza a probabilidade posterior $P(Y'|X')$.

A estimativa precisa das probabilidades posteriores é complexa, pois requer um conjunto de treinamento volumoso para que aconteçam todas ocorrências de combinações entre atributos e variável alvo. O teorema de Bayes resolve esse problema porque permite expressar a probabilidade posterior em termos da probabilidade do rótulo $P(Y)$, da evidência $P(X)$ e da probabilidade condicional do rótulo $P(X|Y)$, conforme consta da Equação 2.6. Durante a comparação das probabilidades posteriores para diferentes valores de Y, o denominador $P(X)$ pode ser ignorado e $P(Y)$ pode ser estimado do conjunto de treinamento, computando a fração de registros de treinamento que pertencem a cada rótulo.

O algoritmo *Naïve Bayes* estima a probabilidade condicional dos rótulos, assumindo que os atributos são condicionalmente independentes, dado um certo rótulo y. A independência condicional pode ser descrita mais particularmente pela Equação 2.7. Cada conjunto de atributos $\mathbf{X} = \{X_1, X_2, \cdots, X_d\}$ possui d atributos [41].

$$P(\mathbf{X}|Y = y) = \prod_{i=1}^{d} P(X_i|Y = y) \tag{2.7}$$

$$P(Y|\mathbf{X}) = \frac{P(Y)\prod_{i=1}^{d} P(X_i, Y)}{P(\mathbf{X})} \tag{2.8}$$

No cômputo das probabilidades, ao invés de calcular todas combinações possíveis de $\mathbf{X}$ calcula-se a probabilidade condicional de cada X_i dado Y. Essa abordagem é muito prática, pois não requer um conjunto de treinamento muito grande para obter uma boa estimativa da probabilidade. Para classificar um registro de teste, o classificador computa a probabilidade posterior para cada rótulo Y. A Equação 2.6 pode ser reescrita conforme a Equação 2.8. Como $P(X)$ é fixo para todo Y, é suficiente escolher a classe que maximiza o termo no numerador, $P(Y)\prod_{i=1}^{d} P(X_i|Y)$. O próximo problema é estimar as probabilidades condicionais $P(X_i, Y)$ para atributos categóricos e contínuos [41].

A estimativa da probabilidade condicional $P(X_i = x_i|Y = y)$ para atributos categóricos é contabilizada de acordo com a fração de instâncias de treinamento de um certo rótulo y que possui um certo valor de atributo x_i. Por exemplo, calcula-se a razão entre o número de ocorrências de x_i que tem rótulo y e o número de ocorrências de y no conjunto de treinamento. No caso da estimativa para atributos contínuos, há o método que discretiza cada atributo contínuo, sendo reduzido ao caso categórico. Outra forma é assumir alguma forma de distribuição probabilística para os dados contínuos, e estimar os parâmetros dessa distribuição, utilizando os dados de treinamento, e.g., distribuição Gaussiana conforme Equação 2.9. A distribuição é caracterizada pelo parâmetro média, μ, e variância, σ^2. O parâmetro μ_{ij} pode ser estimado utilizando a média da amostra de

X_i para todos conjuntos de treinamento que pertencem à classe y_j. No parâmetro σ_{ij}^2, pode ser estimado da variância da amostra de treinamento [41].

$$P(X_i = x_i|Y = y_j) = \frac{1}{\sqrt{2\pi}\sigma_{ij}} \exp - \frac{(x_i - \mu_{ij})^2}{2\sigma_{ij}^2} \tag{2.9}$$

As principais características do algoritmo *Naïve Bayes* podem ser descritas conforme [41] e [45]:

- são robustos para pontos de ruído isolados, pois as probabilidades condicionais são obtidas por uma média. Além disso, consegue trabalhar com valores inexistentes, ignorando os exemplos de treinamento;
- são robustos para atributos irrelevantes, pois não alteram a distribuição de probabilidade uniforme;
- atributos correlacionados podem degradar a eficiência do classificador, pois a hipótese de independência condicional não se aplica para os atributos.

Algoritmo SVM

Segundo [14], *Support Vector Machine* (SVM) é considerado um algoritmo muito rápido e eficiente para problemas de classificação textual. O funcionamento do algoritmo se dá geometricamente por um hiperplano no espaço, o qual separa pontos que representam instâncias positivas da categoria dos pontos que representam instâncias negativas. O hiperplano é escolhido durante a etapa de treinamento como único e com margens máximas definidas. A margem é a distância do hiperplano entre o ponto mais próximo dos conjuntos positivos e negativos. Os hiperplanos são escolhidos por meio de um sub-conjunto das instâncias de treinamento que são chamados de vetores de suporte. Apenas parte dos dados de treinamento é utilizada para escolha do hiperplano, sendo que a outra parte não influencia nessa escolha. O classificador SVM tem como ponto forte a rapidez de execução, pois não depende da dimensionalidade do espaço amostral.

Segundo [41], existem diversas classes de algoritmos SVM que atuam dependendo do tipo de problema de separação, mas os mais citados na literatura são:

- SVM linear com dados separáveis (*Hard-margin*);
- SVM linear com dados não separáveis (*Soft-margin*);
- SVM não linear.

A formulação inicial por Vapnik é conhecida como *Hard-margin* SVM. Esse algoritmo foi evoluído para o algoritmo *Soft-margin* SVM com tolerância a ruídos, erros, dados não linearmente separáveis, em que há uma escolha entre o tamanho da margem e o número de erros de treinamento na fronteira de separação.

A idéia básica por trás da técnica SVM está na escolha de um hiperplano de margem máxima possível. Conforme a Figura 2.5, a separação entre os casos apresentados pode ser realizada por duas margens B_1 e B_2, dado que as duas margens separam os casos sem que ocorram erros de classificação. Cada margem B_i está associada a um par de hiperplano b_{i1} e b_{i2}. O termo b_{i1} é obtido, movendo-se o hiperplano paralelamente da margem até

que chegue perto de um ponto (círculo ou quadrado). O separador com maior margem, no caso B_1, é escolhido como o hiperplano de margem máxima para as instâncias de treinamento.

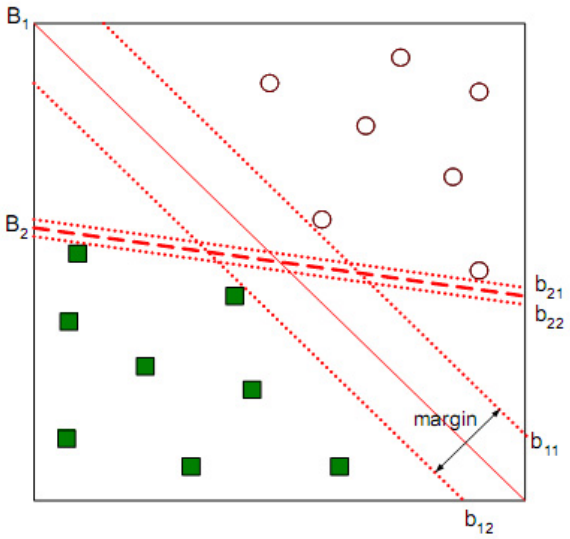

Figura 2.5: Margem de uma margem de decisão [41]

O problema pode ser definido como o problema de classificação de N exemplos de treinamento. Cada exemplo é definido por uma tupla (x_i, y_i), $(i = 1, 2, \ldots, N)$, em que $x_i = (x_{i1}, x_{i2}, \cdots, x_{id})^T$ corresponde ao conjunto de atributos para o i-ésimo exemplo. Por convenção, seja $y_i \in \{-1, 1\}$ o conjunto dos rótulos. A decisão de uma margem de um classificador linear (hiperplano separador) pode ser obtida pela Equação 2.10, em que $\mathbf{w}$ representa vetor normal perpendicular ao hiperplano e b são parâmetros do modelo [41].

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{2.10}$$

Então para o cálculo da margem de separação, definida por b_{11} e b_{12} na Figura 2.5, a Função 2.11 mostra os rótulos com valores -1 e 1 e assim eles podem ser previstos para qualquer exemplo de teste $\mathbf{z}$. Na Equação 2.12, mostra-se como pode ser obtida a margem d da margem de decisão, ou seja a distância entre a margem e as margens máximas.

$$y = \begin{cases} 1, \text{ se } \mathbf{x} \cdot \mathbf{z} + b > 0; \\ -1, \text{ se } \mathbf{x} \cdot \mathbf{z} + b < 0. \end{cases} \tag{2.11}$$

$$d = \frac{2}{\|\mathbf{w}\|} \tag{2.12}$$

O aprendizado do modelo SVM linear compreende a fase de treinamento que envolve estimar os parâmetros $\mathbf{w}$ e b da margem de decisão nos dados de treinamento. A fase de aprendizado pode ser formalizado no problema de otimização das Equações 2.13 e 2.14.

$$\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2} \tag{2.13}$$

$$\text{submetido a } y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1,\ i = 1, 2, \ldots, N. \tag{2.14}$$

No caso proposto, maximizar a margem d na Equação 2.12 é equivalente a minimizar a Equação 2.13. O problema do cálculo de 2.13 e 2.14 é conhecido como um problema de otimização convexa, pois ela descreve uma situação em que ocorre uma função objetivo do tipo quadrática e as restrições são lineares nos parâmetros $\mathbf{w}$ e b. Essa situação pode ser solucionada, utilizando o método multiplicativo de Lagrange. A função objetivo é reescrita de forma que leve em consideração as restrições impostas na solução. A nova função objetivo pode ser descrita na Equação 2.15, a qual é conhecido como o problema de otimização de Lagrange [41].

$$L_P = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \lambda_i \Bigg(y_i \left(\mathbf{w} \cdot \mathbf{x_i} + b\right) - 1 \Bigg) \tag{2.15}$$

Na Equação 2.15, no primeiro termo consta a função objetivo original, no segundo termo, consta as restrições da inequação e os multiplicadores de Lagrange são definidos pelos parâmetros λ_i. A modificação da função objetivo se deve ao caso em que ocorre o vetor nulo $\mathbf{w} = \mathbf{0}$. Esse caso viola as condições da Inequação 2.14. A reescrita da função objetivo 2.15 inclui esse caso, subtraindo o termo da função objetivo inicial [41].

Para minimizar a função objetivo, aplica-se uma derivada parcial de L_p em relação a $\mathbf{w}$ e b, igualando a zero. Assim obtêm-se as soluções apresentadas nas Equações 2.16 e 2.17. Como os multiplicadores de Lagrange são desconhecidos, não é possível resolver os parâmetros $\mathbf{w}$ e b. Se as Equações 2.13 e 2.14 fossem apresentadas com igualdades, ao invés de restrições com desigualdades, N equações de igualdade poderiam ser utilizadas com as Equações 2.16 e 2.17 para buscar soluções possíveis para $\mathbf{w}$, b e λ_i.

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x_i} \tag{2.16}$$

$$\frac{\partial L_p}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{2.17}$$

Para tratar as restrições de desigualdades, transformando-as em um conjunto de restrições de igualdades, faz-se necessário que os multiplicadores de Lagrange sejam restritos a valores não negativos. Essa transformação implica nas condições de Karush-Kuhn-Tucker (KKT). Elas estão descritas pelas Equações 2.18 e 2.19.

$$\lambda_i \geq 0, \tag{2.18}$$

$$\lambda_i[y_i(\mathbf{w} \cdot x_i + b) - 1] = 0. \tag{2.19}$$

Aplicando a Equação 2.19, a maior parte dos multiplicadores de Lagrange são zerados. Essa restrição diz que λ_i deve ser zerado a menos que a instância de treinamento x_i satisfaça a Equação $y_i(\mathbf{w} \cdot x_i + b) = 1$. Caso seja satisfeita, a instância de treinamento com $\lambda_i > 0$ estará posicionada ao longo dos hiperplanos b_{i1} ou b_{i2} sendo cunhada de vetor

de suporte. As Equações 2.16 e 2.19 sugerem que os parâmetros $\mathbf{w}$ e b definidores da margem de decisão dependerão apenas dos vetores de suporte.

A solução do problema de otimização necessita de mais simplificação, visto que existem muitos parâmetros a serem determinados ($\mathbf{w}$, b e λ_i). A simplificação é obtida, expressando a função de Lagrange 2.15 em termos dos multiplicadores, conhecido como o problema dual. As Equações 2.16 e 2.17 são substituídas na Equação 2.15, resultando na Equação 2.20.

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \mathbf{x_i} \cdot \mathbf{y_j} \tag{2.20}$$

A diferença entre as equações de Lagrange primária e dual, L_P e L_D respectivamente, está no tipo de variáveis envolvidas. No caso dual, apenas os multiplicadores e os dados de treinamento são utilizados. No caso primário, os multiplicadores e parâmetros de margem de decisão são necessários. Para questões de otimização, ambas soluções são equivalentes. A equação primária trata de um problema de minimização, enquanto a dual é um problema de maximização.

Para conjuntos de dados extensos, o problema de otimização dual pode ser resolvido, empregando técnicas numéricas como programação quadrática. Uma vez que os multiplicadores são calculados, as Equações 2.17 e 2.19 são utilizadas para soluções possíveis de $\mathbf{w}$ e b. A margem de decisão pode ser descrita pela Equação 2.17. O valor de b pode ser obtido pela resolução da Equação 2.19 dos vetores de suporte. Como os valores de λ_i possuem erros numéricos, o valor computado para b pode não ser único. Na prática, a média do valor de b é escolhida como parâmetro.

O caso do SVM até então descrito não permite a ocorrência de erros de classificação, daí seu nome *hard-margin*. No entanto, os dados reais não estão livres de ruídos. A Figura 2.6 mostra os pontos circulados para os quais não é possível obter uma margem separadora dos rótulos isenta de erros. Constatado esse problema, foi desenvolvida a abordagem *soft-margin*. Esse caso (SVM linear não separável) é definido pela situação em que é tolerável a ocorrência de erros nos dados de treinamento. As restrições do problema de otimização podem ser reescritos pelas Equações 2.21 (sujeita a minimização) e Equação 2.22 (função classificadora), introduzindo variáveis frouxas ξ com valores positivos. As equações necessitam evitar a ocorrência de alta taxa de erros de treinamento, inserindo penalidades para variáveis frouxas com valores significativos.

$$L(w) = \frac{1}{2}\|\mathbf{w}\|^2 + C(\sum_{i=1}^{N} \xi_i^k) \tag{2.21}$$

$$f(\mathbf{x_i}) = \begin{cases} 1 \text{ se } \mathbf{w} \cdot \mathbf{x_i} + b \geq 1 - \xi_i \\ -1 \text{ se } \mathbf{w} \cdot \mathbf{x_i} + b \leq -1 + \xi_i \end{cases} \tag{2.22}$$

O caso da SVM não linear trata de situações em que não existe uma separação linear nos exemplos de treinamento. A solução para esse problema é transformar os dados no espaço original em um novo espaço $\Phi(\mathbf{x})$, em que uma margem de decisão linear pode ser aplicada para separar as instâncias do espaço transformado. Na Figura 2.7, é mostrado em (a) uma margem elíptica (não linear) e sua solução linear é possível em um espaço

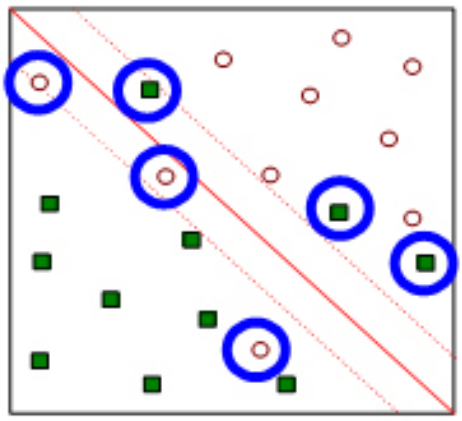

Figura 2.6: SVM linear com dados não separáveis [41].

com dimensionalidade maior em (b). A tarefa de aprendizado em uma SVM não linear pode ser formalizado nas Equações 2.23 e 2.24.

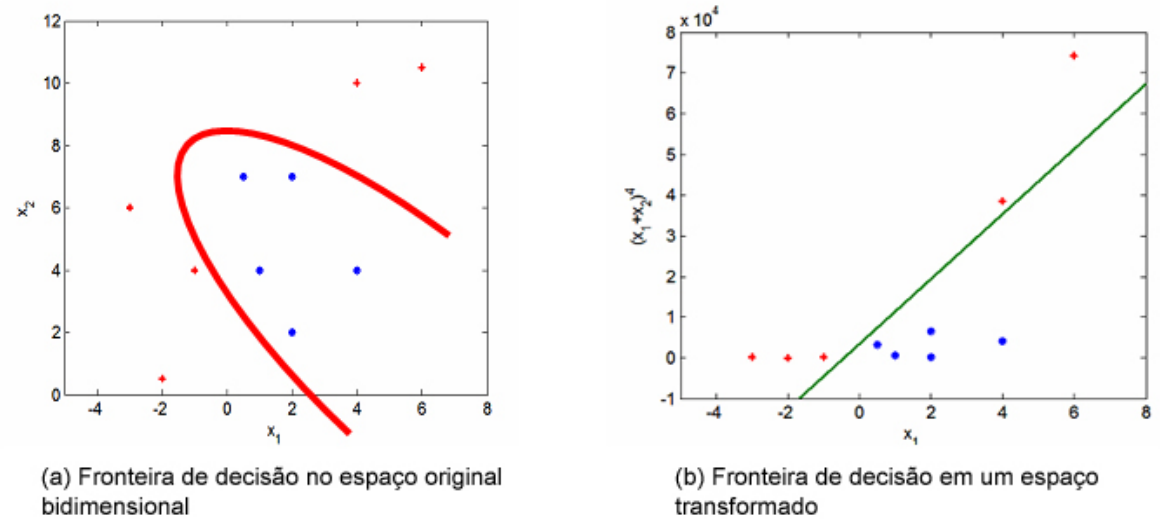

Figura 2.7: SVM não linear [41].

A resolução da função de Lagrange na Equação 2.25, há ocorrência do produto escalar $\mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j})$. O método de computar a similaridade no espaço transformado utilizando o conjunto de atributos original pra cálculo do produto escalar é denominado truque do *kernel*. A função *kernel* pode ser expressa pelos vetores de entrada $\mathbf{u}$ e $\mathbf{v}$ conforme a Equação 2.26. A utilização de funções *kernel* minimiza o custo computacional, inclusive evitando o problema de alta dimensionalidade. O *kernel Radial Basis Function* (RBF) é descrita pela Equação 2.27 [41].

$$\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2} \tag{2.23}$$

$$\text{sujeito a } y_i(\mathbf{x} \cdot \mathbf{\Phi}(\mathbf{x_i}) + b) \geq 1,\ i = 1, 2, \ldots, N. \tag{2.24}$$

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j}) \tag{2.25}$$

$$K(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^2 \tag{2.26}$$

$$K_{RBF}(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma ||\mathbf{x_i} - \mathbf{x_j}||^2), \text{para } \gamma > 0 \tag{2.27}$$

Algoritmo SMO

Segundo [35], o algoritmo *Sequential Minimal Optimization* (SMO) surgiu devido à necessidade de um algoritmo SVM com implementação rápida, simples e capaz de tratar conjuntos de dados mais extensos. Além disso, tem capacidade de tratar conjuntos de dados esparsos, que possuem um número substancial de elementos com valor zero, pois empregam maior ênfase na etapa de avaliação da função de decisão. A otimização realizada no algoritmo SMO está na programação quadrática analítica, invés da abordagem numérica utilizada tradicionalmente nas implementações SVM.

O algoritmo SMO escolhe a resolução dos problemas de otimização, optando pelas menores otimizações possíveis em cada passo. Nos problemas de programação quadrática em SVM, a menor otimização possível envolve dois multiplicadores de Lagrange, pois eles devem obedecer a restrição de igualdade linear. Em cada passo, o método SMO escolhe a otimização de dois multiplicadores, buscando valores ótimos para eles e atualizando-os para refletir os novos valores ótimos. A vantagem está em utilizar um otimizador analítico ao invés de ser chamada toda uma biblioteca de rotinas de programação quadrática. Além disso, não é necessário armazenamento de matrizes extras, o que permite manipular problemas com conjunto de treinamento volumoso.

O algoritmo SMO consiste de três componentes principais: (i) o método analítico, para solucionar os dois multiplicadores de Lagrange; (ii) uma heurística para escolher quais multiplicadores otimizados; (iii) um método para computar b, um dos parâmetros da margem de decisão.

No funcionamento do SMO, ocorre o cômputo das restrições de apenas dois multiplicadores de Lagrange escolhidos. Logo após, é solucionado o problema de maximização das restrições. Eles podem ser representados bidimensionalmente conforme apresentado na Figura 2.8, em que se buscam limites superiores e inferiores aos parâmetros dos multiplicadores. As restrições que envolvem inequação pelo método *soft-margin* cercam o valor dentro do quadrado, e a busca da minimização da função objetivo é obtida pelo posicionamento na linha diagonal. A função objetiva é calculada, tomando-se o valor mínimo na diagonal do quadrado (equação linear) [35].

As escolhas de heurísticas podem ocorrer no primeiro ou segundo multiplicador de Lagrange. Na primeira escolha heurística, o algoritmo procura casos em que ocorrem violação das condições KKT. Com isso, não é necessário percorrer completamente o conjunto de treinamento, diminuindo o tempo de processamento. O exemplo que viola as condições KKT é sujeito a uma otimização imediata. Uma vez descoberta a violação, um

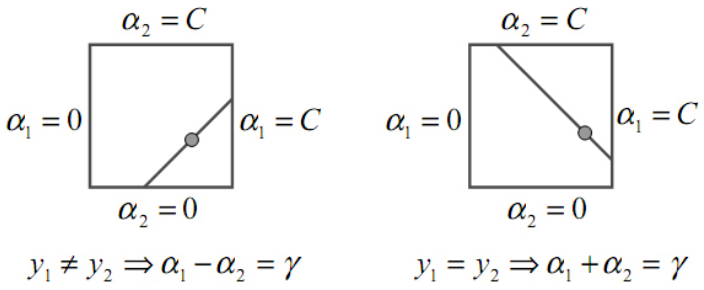

Figura 2.8: Restrições de Lagrange no algoritmo SMO [35].

segundo multiplicador é escolhido, utilizando a segunda escolha, e os dois multiplicadores são submetidos a otimização conjunta. Durante a execução, a viabilidade de programação quadrática dual é mantida. O SVM é atualizado com os valores dos novos multiplicadores, procurando por outros casos de violação para o primeiro multiplicador [35].

O algoritmo SMO permite que as condições KKT sejam cumpridas dentro de uma margem ε. Quanto maior a acurácia da saída, mais tempo levará para que o algoritmo chegue à convergência. Uma vez resolvidos os multiplicadores, é necessário determinar um valor limite para as margens b, em que as condições KKT são satisfeitas. Ele é computado, mantendo um *cache* para cada vetor de suporte e escolhe o erro para maximizar o passo.

2.2.4 Avaliação de Resultados

Segundo [41], a avaliação dos resultados do modelo de classificação é obtida por meio de contagem das classificações corretas e incorretas do conjunto de teste. As contagens são estruturadas na forma de uma tabela chamada de matriz de confusão. É possível medir a acurácia e a taxa de erros diante dos dados da matriz. Com intuito de resumir essa tabela, existem diversas métricas de avaliação, incluindo casos de classes desbalanceadas; São as métricas *precision*, *recall*, *F-measure* e gráfico de curvas *Receiver Operating Characteristic* (ROC), entre outros.

	Classe Prevista 1	Classe Prevista 0
Classe Real 1	f_{11}	f_{10}
Classe Real 0	f_{01}	f_{00}

Tabela 2.2: Matriz de confusão para o problema de classificação binária [41].

A matriz de confusão, conforme apresentada na Tabela 2.2, indica o número de instâncias classificadas corretamente pela diagonal principal. As linhas compreendem os rótulos com suas classificações reais. As colunas representam o resultado da classificação supervisionada. Uma classificação isenta de erros possui f_{10} e f_{01} igual a zero. As Equações 2.28–2.32 podem ser deduzidas da matriz de confusão. A acurácia pode ser entendida como o quão perto de uma classificação isenta de erros o aprendizado foi. A taxa de erro contabiliza os erros de classificação em todos os rótulos. *Precision* indica a razão entre os casos corretamente classificados e o número real de instâncias dado um rótulo escolhido. O *recall* pode ser entendido como a razão entre o número de casos avaliados como corretamente classificados e o número total de casos previstos dado um certo rótulo.

Por fim, a medida *F-measure* resume as métricas de precisão e *recall* por meio de uma média harmônica entre elas [41].

$$\text{Acurácia} = \frac{\text{Número de predições corretas}}{\text{Número total de predições}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}} \tag{2.28}$$

$$\text{Taxa de Erro} = \frac{\text{Número de predições incorretas}}{\text{Número total de predições}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}} \tag{2.29}$$

$$Precision = \frac{f_{11}}{f_{11} + f_{01}} \tag{2.30}$$

$$Recall = \frac{f_{11}}{f_{11} + f_{10}} \tag{2.31}$$

$$F\text{-}Measure = \frac{2 \times Recall \times Precision}{Recall + Precision} = \frac{2 \times f_{11}}{2 \times f_{11} + f_{01} + f_{10}} \tag{2.32}$$

2.2.5 Ferramentas de Mineração

Segundo [28], é alto o custo de aquisição de um *software* proprietário que faz tarefas de aprendizado por máquina, se comparado às várias soluções disponíveis comercialmente. O projeto *Waikato Environment for Knowledge Analysis* (Weka) se destaca nesse cenário como opção em *software* livre para mineração de dados [17].

De acordo com [45], o Weka é um ambiente de mineração *open source* que reúne algoritmos de aprendizado por máquina e funcionalidades de pré-processamento de dados. Ele provê suporte para experimentos em mineração, provendo esquemas de avaliação, relatório com estatísticas, avaliação e visualização dos modelos inferidos. O Weka provê uma interface única para que diferentes métodos de aprendizado possam ser comparados e auxilia identificar aqueles que são mais apropriados para solucionar o problema.

A ferramenta disponibiliza vários algoritmos de preparação de dados, mineração e validação de resultados. Escrito em código aberto Java, possui boa portabilidade entre diversos sistemas operacionais e disponibiliza interface gráfica que facilita seu uso. Os mantenedores do projeto também disponibilizam um fórum para os usuários do seu sistema como forma de manter uma rede de colaboração. Além disso, a página eletrônica do projeto disponibiliza documentação para utilização do *framework*, incluindo o código fonte implementado. Os algoritmos existentes na ferramenta fornecem relatórios dos dados estatísticos analisados no domínio minerado. Vários recursos são acessíveis diretamente via interface gráfica ao usuário, que de certa forma pode abstrair a complexidade e detalhes internos dos passos envolvidos no processo de mineração de dados como descrito por [40].

A interface gráfica é facilmente absorvida pelo usuário que inicialmente deve se ambientar na padronização das tarefas básicas. A ferramenta Weka permite o pré-processamento que compreende na escolha da base de dados, navegação nos atributos, filtragem de dados (seleção, discretização, normalização, amostragem, entre outros). Outra funcionalidade disponibilizada é a tarefa de classificação. Algoritmos de classificação diversos podem ser selecionados e ajustados. Ainda na classificação, temos a opção de teste e validação do modelo gerado. A ferramenta permite escolher o conjunto de dados de treinamento, o

conjunto de testes, o tipo de validação cruzada e a seleção do atributo alvo para classificação. É possível visualizar um resumo da tarefa efetuada com dados estatísticos do modelo, matriz de confusão, métricas de desempenho, entre outras [40].

O *framework* Weka pode ser utilizado, acessando 4 principais opções: A interface de linha de comando (CLI), o explorador, o experimentador e o fluxo de conhecimento. A linha de comando possui a vantagem de acesso a funcionalidades mais básicas ou acesso via sistema operacional. O explorador permite avaliar esquemas de mineração mais básicos por meio de uma interface gráfica. O experimentador tem o objetivo de automatizar processos de mineração, possibilitando o teste de diversas configurações de mineração, conjunto de dados, filtros, coletar estatísticas de eficiência, entre outros. O módulo de fluxo de conhecimento permite modelar por meio de grafos um fluxograma de pré-processamento de dados, mineração e avaliação dos dados [45].

Existem diversas opções de *software* aberto para mineração de dados, a saber:

- KNIME (`http://www.knime.org/`);
- Orange (`http://www.ailab.si/orange/`);
- *The R Project for Statistical Computing* (`http://www.r-project.org/`);
- R-Orange (`http://code.google.com/p/r-orange/`);
- RapidMiner (`http://rapid-i.com/`);
- Rattle (`http://rattle.togaware.com/`);
- UnBBayes (`http://unbbayes.sourceforge.net/`).

Uma alternativa de ferramenta livre de mineração de dados é o UnBBayes. A ferramenta é capaz de realizar aprendizado, utilizando redes probabilísticas como as redes de Bayes, suporta utilização de linguagens ontológicas e avaliação de rendimento dos classificadores [9]. Outra ferramenta de escolha *open source* é o *Rapid Miner* a qual provê suporte para utilização de diversos operadores de pré-processamento, algoritmos de mineração, elaboração de gráficos, linguagem de *scripts* e inclui os algoritmos utilizados no Weka.

Entre as ferramentas de processamento de texto podemos citar o PreTexT. Esse projeto aborda de maneira simples o problema supracitado, o qual em comparação a outras ferramentas é necessário buscar vários filtros que ajustam os dados e precisam estar encadeados de forma lógica e concisa. No PreText esse encadeamento é realizado utilizando um único arquivo de configurações, o que não ocorre com outras ferramentas.

De acordo com [31], a ferramenta PreTexT possibilita solucionar problemas de pré-processamento, transformando textos em representação estruturada tipo BOW a qual será utilizada como entrada para algoritmos de mineração de dados. O PreTexT é capaz de trabalhar o problema de redução de dimensionalidade se valendo de algoritmos de radicalização e normalização de atributos. Além disso, o PreTexT é capaz de trabalhar com textos em língua portuguesa, inglesa e espanhola. A ferramenta possui fácil utilização e elimina um passo difícil da mineração que é a preparação dos dados.

O funcionamento do PreTexT pode ser visto na Figura 2.9, onde são mostrados os módulos principais, as entradas e saídas. Os principais módulos são os arquivos *stem.pl* e

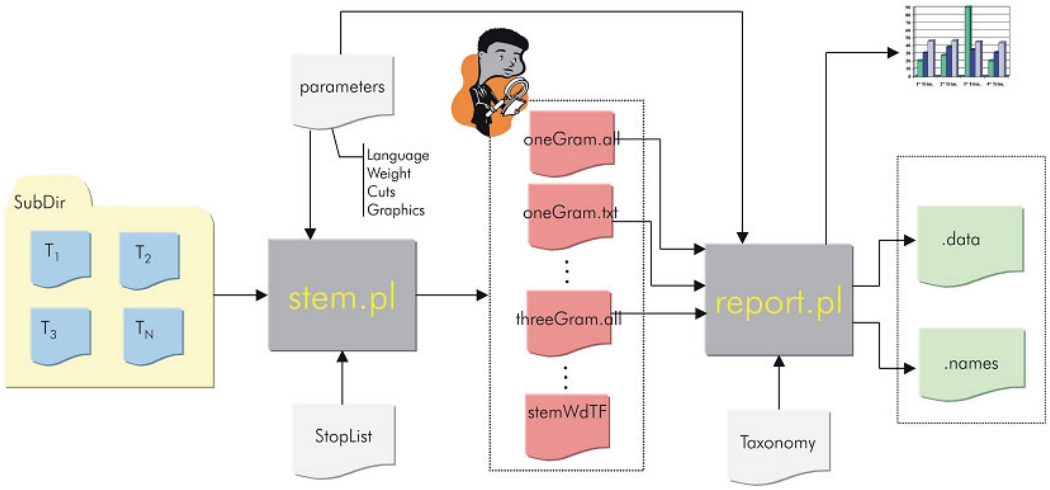

Figura 2.9: Diagrama de funcionamento da ferramenta PreTexT [31].

report.pl. O módulo *Stem* transforma termos em suas formas raízes, diminuindo a dimensionalidade dos atributos. No arquivo de parâmetros para execução do módulo de *Stem*, pode-se ajustar a linguagem do texto utilizado, o qual processa textos em português, espanhol ou inglês. Além disso, o tipo de normalização (Booleano, TF, TFIDF, TFLINEAR) pode ser configurado no arquivo *Parameters*. A remoção de palavras irrelevantes (*stopwords*) também é executada de acordo com um arquivo texto, contendo a lista de termos. Os dados de entrada são conjuntos de textos dispostos em forma de diretórios cujo nome dessa pasta é o rótulo. Cada pasta contém um conjunto de textos, em que cada texto está gravado em um arquivo.

A saída do módulo *Stem* é composta por arquivos com os radicais e suas freqüências absolutas. No caso *one-gram*, *two-gram* e *three-gram* são radicais compostos por uma, duas ou três palavras. O arquivo *stemWdTF* mostra todos os termos reunidos em um único arquivo.

O módulo *Report* é responsável por criar a tabela BOW de atributo-valor. A redução de dimensionalidade neste módulo ocorre pelos cortes de Luhn e a lei de Zipf apresentados na Seção 2.2.2. Além dessas funções, os dados passam por uma normalização de acordo com os parâmetros mostrados também na Seção 2.2.2. A redução de dimensionalidade pode ser mais explorada pela utilização de um arquivo de definições de taxonomia. Esse conceito utiliza listas de palavras radicalizadas que podem ser substituídas em um único termo radicalizado. A saída do processamento do módulo gera um arquivo *.data* contendo a matriz BOW e o arquivo *.names*, contendo os *stems* (atributos) da matriz representada.

Existem outras ferramentas para pré-processamento de textos, as quais se encontram incluídas nos *softwares* de mineração de dados. Por exemplo, o *RapidMiner* disponibiliza vários filtros de tratamento de textos. Esses filtros auxiliam nas tarefas de limpeza de dados supracitados, entretanto não tratam textos em língua portuguesa.

2.3 Trabalhos Correlatos

Serão apresentados alguns trabalhos que têm correlacionamento com o assunto pesquisado, o qual se refere à classificação automática de documentos em forma de notícias e *e-mails*.

Como trabalho tradicionalmente citado de [39], onde encontramos uma discussão das principais abordagens para o problema de classificação automática de *e-mails* e textos no âmbito dos paradigmas atuais de aprendizagem de máquina. Esse tipo de problema esconde três outros sub-problemas que se situam na representação do documento, construção do classificador e avaliação do classificador. Inicialmente é realizado um breve histórico acerca do problema.

Segundo [39], a representação do documento foi realizado pela organização BOW, sendo que outras representações mais elaboradas podem não implicar necessariamente em melhor eficácia de classificação. No entanto, a combinação entre as abordagens que consideram aspectos sintáticos e estatísticos é provavelmente a melhor escolha para solução desse problema. Quanto ao peso atribuído a cada palavra, dependerá da escolha do algoritmo classificador, como por exemplo, o peso binário ou TFIDF comumente utilizado. Existem abordagens alternativas bastante importantes como o *Darmstadt Indexing Approach* (DIA). Esse método foi criado para atender às especificações do sistema AIR/X, que classifica textos sobre literatura científica em categorias.

O método DIA considera propriedades dos termos, documentos, categorias e relacionamentos entre esses elementos. Esse método se destaca entre outros, pois leva em consideração essas propriedades adicionais. A questão da redução de dimensionalidade é destacada como um problema crucial, pois tende a reduzir o problema do super-ajustamento. Experimentos realizados mostram que o número de exemplos de treinamento deve ser proporcional ao número de termos utilizados. A redução de dimensionalidade pode atuar em cada categoria (por meio de uma lista de termos escolhidos) ou globalmente (por meio de uma listagem de termos para todas categorias). Outras técnicas também aplicadas são a seleção ou exclusão do termo.

No trabalho do autor foi realizada uma compilação de experimentos realizados por diversos autores na base de dados *Reuters*. Em seguida, o autor expõe uma análise sobre algoritmos amparados por técnicas probabilísticas, árvores de decisão, SVM, incremental *on-line*, redes neurais, entre outros. Algoritmos como o *Boosting*, SVM e outros baseados em métodos de regressão apontam como os melhores em termos de eficiência. Técnicas de redes neurais e classificadores *on-line* também situam-se nesse patamar, embora apontem eficiência levemente diferenciada. No entanto, classificadores em lote e *Naïve Bayes* indicam baixa eficiência. A técnica de árvores de decisão mostra resultados tão bons quanto à classe de algoritmos com melhor eficiência.

O autor enfatiza que os resultados obtidos poderão ser diferentes caso ocorra mudança da base de textos *Reuters* para outra qualquer. Adicionalmente, os algoritmos utilizados poderão ter suas posições de eficiência modificadas. Finalmente o autor conclui que a comunidade pesquisadora na área de pesquisa e classificação de textos deverá crescer, visto que há inúmeros domínios aplicáveis. Além disso, novas aplicações que tratam documentos numerosos deverão utilizar-se dessas técnicas, em que a classificação manual é impossível. A classificação pode não chegar a 100%, mas chegará a ser mais eficiente comparada à velocidade manual. Quanto aos classificadores, os algoritmos são propostos

com forte fundamentação matemática. Por fim, a grande disponibilidade de documentos impõe maior rigor nas técnicas que devem ser capazes de tratar casos mais realistas.

O artigo proposto por [46] mostra um estudo comparativo de significância estatística entre as técnicas SVM, *K-Nearest Neighbor* (KNN), redes neurais, *Linear Least-squares Fit Mapping* (LLSF) e *Naïve Bayes* (NB). Foi empregada a base de textos *Reuters* — 21578 com categorias desbalanceadas. Os classificadores foram avaliados por meio das métricas de precisão, *recall* e *F-measure*. Os testes de significância foram utilizados para ordenar os melhores classificadores. A seleção de atributos foi realizada na etapa de pré-processamento, valendo-se de medidas estatísticas χ^2 ou critério de ganho de informações.

Os resultados pelo método SVM mostraram melhores resultados em um *kernel* linear. Resultados mostram que os algoritmos SVM, KNN e LLSF possuem melhores resultados comparados às técnicas de redes neurais e NB quando as instâncias de treinamento são pouco numerosas. Por outro lado, quando as categorias são numerosas, todos algoritmos mostram resultados igualmente capazes.

No artigo de [24], são analisados os problemas de classificação automática de *e-mails* e filtragem de *spam*. As duas tarefas são executadas com os algoritmos *random forests* (RF), *co-training*, SVM, árvores de decisão e *Naïve Bayes*. A abordagem semi-supervisionada do *co-training* mostra ganhos marginais em relação às abordagens supervisionadas.

Os dados utilizados foram *e-mails* coletados de quatro usuários diferentes, inclusive *e-mails* referentes ao período de um ano de um dos autores. Para a filtragem de *spam*, três bases foram empregadas (LingSpam, PU1 e U5Spam). O pré-processamento dos textos é realizado por um *framework* proposto pelos autores (LINGER). A representação utilizada foi a BOW. Os termos foram extraídos de todos os campos, considerando anexos como conteúdo do corpo da mensagem. Não foram tratados *stopwords*, apenas símbolos e delimitadores especiais. Adicionalmente, não foi aplicado nenhum algoritmo de radicalização. Foram excluídos, no entanto, palavras com freqüência única e termos com mais de 20 caracteres.

Na etapa de normalização e aplicação de pesos nos atributos, foram aplicadas duas técnicas pelo *framework* LINGER. As métricas de ganho de informações e variância da freqüência do termo ou *term frequency variance* (TFV) ordenam os termos, escolhendo aqueles com maior pontuação. O método TFV é considerado como um melhoramento da contagem de freqüência dos termos. Os algoritmos classificadores foram avaliados segundo os resultados de acurácia, precisão, *recall* e *F-measure*. Os testes realizados indicaram que, durante o pré-processamento, a técnica TFV conduziu a resultados melhores. O resultado final da pesquisa atesta que o algoritmo RF superou o desempenho dentre todos os outros. Os demais algoritmos de árvores de decisão (J48), SVM (SMO) e NB — respectivamente em ordem decrescente de desempenho — ficaram em segundo plano.

Segundo o autor, o qual escolheu o algoritmo RF como uma abordagem com potencial para rotulagem automática de *e-mails*, por ser de fácil ajustamento, o qual mostra eficiência na execução de conjuntos de dados volumosos com alto número de atributos, o que justifica o uso em categorização de textos. A tarefa de encaminhar *e-mails* em pastas ou rótulos é complexa. Essa dificuldade vem do fato de que um sistema automático depende fortemente do estilo de classificação adotado pelo usuário. A tarefa de filtragem de *spam* investigada demonstrou que a abordagem *co-training* é favorável para bons resultados, além de necessitar de poucos exemplos para treinamento.

Os trabalhos correlatos que foram pesquisados mostraram semelhanças no tratamento de texto e uso de algoritmos classificadores. Foram utilizadas nas mensagens eletrônicas técnicas simples como a contagem pela freqüência dos termos, booleana ou a técnica TFIDF. Quanto ao uso de classificadores nos artigos pesquisados, foram recorrentes as abordagens de árvores de decisão, *Naïve Bayes* e SVM. A emprego dessas abordagens auxilia comparar a eficiência com outros algoritmos novos propostos.

Houve também diferenças de abordagens entre os artigos pesquisados e este trabalho. Dois dos artigos utilizaram a base de textos *Reuters*, que possui características muito diferentes se comparados aos textos de *e-mails*. No pré-processamento, foi também escolhida uma técnica que levou em consideração o valor sintático dos termos e seus relacionamentos, o qual auxilia na redução de dimensionalidade. No uso de classificadores, a utilização de técnicas de classificação *on-line* foi um diferencial importante, pois essa abordagem é capaz de tratar grandes volumes de textos e possui aprendizagem incremental.

Capítulo 3

Metodologia do Trabalho

Esta pesquisa propõe um processo para categorização de eventos e incidentes de segurança utilizando as ferramentas de *software* aberto PreTexT e Weka. A classificação é multi-classe, visto que uma mensagem pode pertencer à apenas uma das categorias estabelecidas pelo CTIR Gov. Além disso, ela é supervisionada, pois as mensagens já possuem rótulos que foram atribuídos pelos analistas técnicos. Durante realização desse trabalho não foram atribuídos pesos distintos aos cabeçalhos dos *e-mails*, para manter a simplicidade do modelo, mas como trabalhos futuros certamente seria interessante esse tratamento.

O conjunto de dados obtido deverá ser processado pelo *framework* de mineração, observando-se restrições de memória volátil, pois a classificação não será realizada de maneira incremental e *on-line*. O conjunto de documentos textuais foi obtido das mensagens eletrônicas registradas no CTIR Gov entre janeiro e agosto de 2009, totalizando cerca de 10^5 mensagens. O número de mensagens em cada rótulo configura uma situação de desbalanceamento entre classes, sendo que a coleção empregada foi estática e permaneceram inalterados conforme Seção 2.2.2.

A fase de treinamento deve ocorrer de forma rápida como o pré-processamento. Os resultados do treinamento deverão ter tempo de resposta mínimo concomitantemente com uma menor taxa de erros possível na fase de testes. Limites de memória e processamento devem ser considerados devido restrições de recursos computacionais atuais.

O pré-processamento adotará a abordagem BOW, em que podem ser escolhidos parâmetros de normalização e redução de dimensionalidade. Na fase de mineração de texto propriamente dita, os algoritmos utilizados serão o J48, *Naïve Bayes* e SVM/SMO, conforme justificativa apresentada na Seção 1.1. Cada algoritmo possui um nível de complexidade e abordagem diferente, o que permitirá a escolha do melhor algoritmo classificador, tendo em vista o espaço de combinações paramétricas possíveis, tanto no pré-processamento quanto na mineração.

Essa proposta pode auxiliar a triagem, diminuindo o tempo gasto pelo analista de triagem designado. A associação dessa proposta a sistemas de rastreamento de incidentes poderá otimizar o processo de tratamento de incidentes de segurança em qualquer centro de tratamento de incidentes, em que o analista poderá obter rapidamente uma visão geral dos incidentes e eventos recebidos. Além disso, a proposta facilita um melhor conhecimento das mensagens recebidas por qualquer CSIRT, por meio de informações gerenciais consolidadas.

Por outro lado, estudos de mineração acerca de classificação de mensagens de *e-mail* são pouco relatados na literatura. No entanto, aplicações desse problema podem facilmente adaptar-se em áreas como telecomunicações, saúde, finanças, centros de atendimento a clientes, entre outros.

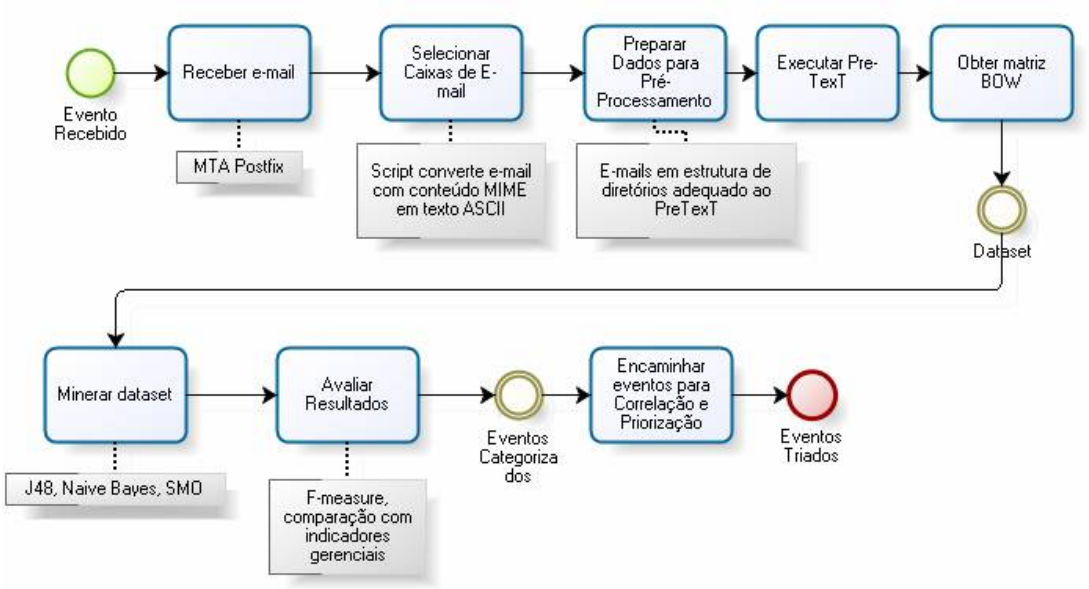

Figura 3.1: Tarefas realizadas no processo de mineração.

Na Figura 3.1, apresentamos as tarefas realizadas durante o processo de mineração. Os primeiros três elementos mostrados indicam um momento anterior à coleta de dados. A preparação dos dados para pré-processamento, com a execução do PreTexT, obtenção da matriz BOW consiste no pré-processamento de dados. A tarefa minerar *dataset* define a execução dos algoritmos citados. A eficiência do modelo classificador gerado é testada, utilizando validação cruzada em 10 dobras ou fornecendo um outro conjunto de dados. A avaliação dos resultados é uma etapa final da mineração, embora os três últimos elementos da Figura 3.1 mostrem um possível desfecho para o processo de triagem no CTIR Gov.

Com base na apresentação da problemática motivada pela Figura 1.1, pode-se perceber que a categorização automática dos eventos recebido no CTIR Gov trabalha na direção do volume crescente de comunicações recebidos pelo centro. A classificação manual torna-se praticamente irrealizável, por isso sua motivação em um caso real. Na visão da Gestão de Incidentes, uma categorização eficiente pode facilitar as tarefas posteriores de correlação e priorização.

Dessa forma, este trabalho propõe um processo para categorização automática de eventos de segurança recebidos pelo CTIR Gov. O modelo propõe um processo para classificação eficiente e ajustada que analisa escolhas técnicas possíveis para pré-processamento e algoritmos de classificação já citados. A Figura 3.2 mostra uma visão geral da arquitetura da solução.

Na Etapa 1 da Figura 3.2, o sistema de *e-mails* do CTIR Gov pode ser considerado a fonte dos dados no processo de mineração. O recebimento das mensagens é realizado por um agente de *e-mail*, *mail transfer agent* (MTA). O MTA utilizado no CTIR Gov é o PostFix que se encontra configurado em um servidor remoto (http://www.postfix.org/). O

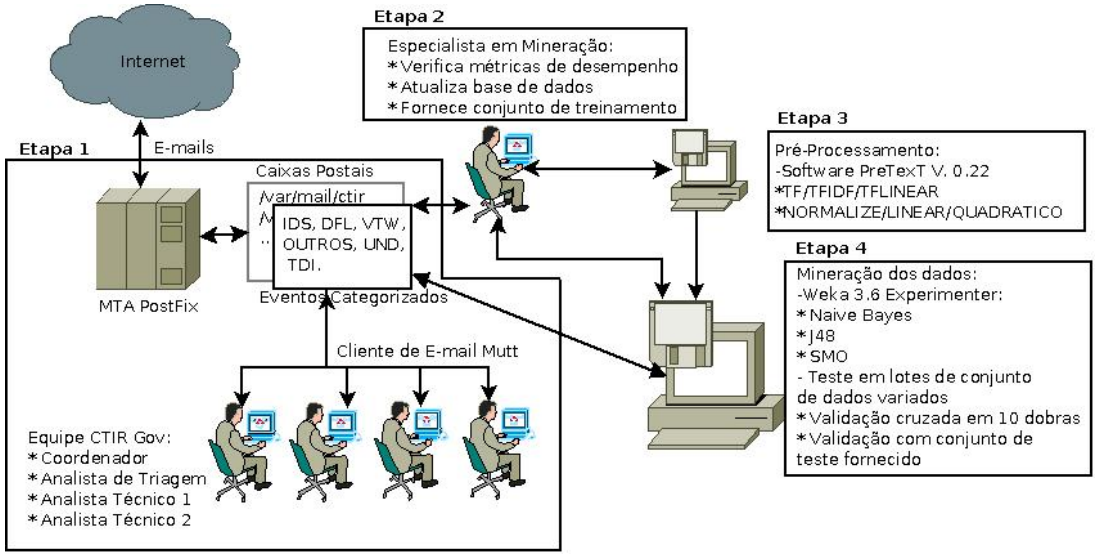

Figura 3.2: Arquitetura do modelo de solução.

servidor guarda mensagens recebidas em arquivos tipo texto, concatenando as mensagens mais recentes no final de cada arquivo (caixa postal). A equipe do centro acessa *e-mails* utilizando um *software* cliente orientado a caracteres (Mutt — `http://www.mutt.org/`). Essa arquitetura de comunicação deixa o analista técnico com total controle dos dados visualizados. O cliente Mutt não utiliza modo gráfico, reconhecimento de formatações especiais, *scripts* de execução em segundo plano, *links* ativos no corpo das mensagens, anexos associados automaticamente à execução de outros programas, entre outros.

Na Etapa 2 da Figura 3.2, ocorre a configuração dos ambientes PreTexT e Weka, a qual é realizada por um especialista responsável por conduzir o processo. A extração dos dados é iniciada pela cópia das caixas postais de *e-mails*, contendo eventos recebidos pelo CTIR Gov no período de janeiro a agosto de 2009. Elas são agrupadas de acordo com 6 rótulos adotados pelo CTIR Gov. Particularmente, o CTIR Gov utiliza classificações que são definidas por uma ou mais caixas postais. Essa categorização é motivada pelo agrupamento maior possível dos casos mais recorrentes, embora existam classificações mais completas conforme descrito por [20].

Os rótulos com suas respectivas caixas postais podem ser resumidos conforme as Tabelas 3.2 e 3.1. Os rótulos DFL, IDS e VTW são mais numerosos, pois tratam de eventos gerados por robôs sensores. O rótulo OUTROS possui definição complexa, já que engloba casos diversos como solicitações de informações, respostas de outros incidentes, eventos, informações de colaboração entre CSIRTs, entre outros.

Conforme Etapa 3 de pré-processamento apresentado na Figura 3.2, modifica-se o modo de representação dos documentos, transformando textos em valores numéricos capazes de serem tratados pelos algoritmos de classificação. A ferramenta PreTexT foi escolhida para utilização porque possibilita o tratamento de textos em língua portuguesa, além de ser uma solução de livre uso. O *software* possui fácil integração com os algoritmos mineradores, disponibiliza métodos conhecidos de redução de dimensionalidade e normalização como TF, TFIDF e outras conforme citada na Sub-seção Pré-Processamento de Texto da Seção 2.2.2.

Os arquivos copiados necessitam de uma organização prévia para uma execução com

Rótulo	Definição do Rótulo
IDS	Alertas de sistemas de detecção a intrusão.
DFL	Alerta de desfiguração em sítios da APF.
VTW	Alertas de vírus, *trojans* e *worms*.
OUTROS	Solicitações de informações em geral.
UND	Alertas de *e-mails* que não puderam ser entregues.
TDI	Tentativa de intrusão ao servidor de *e-mails*.

Tabela 3.1: Nome e definição dos rótulos.

Rótulo	Nome da Caixa Postal	Número de *E-mails*	Porcentagem
IDS	naiids_recepcao,naiids_tx	74.151	72,5%
DFL	dfl	10.681	10,4%
VTW	naivtw,nai_recepcao	9.969	9,8%
OUTROS	ctir	4.886	4,8%
UND	undeliverable	1.962	1,9%
TDI	nailogs_tx	611	0,6%
		Total: 102.260	100%

Tabela 3.2: Listagem dos rótulos, caixas postais e número de mensagens em português.

êxito no PreTexT. Cada mensagem de *e-mail* é armazenada em um único arquivo texto, em que toda formatação adicional é excluída. Os arquivos são dispostos em diretórios, em que cada diretório simboliza um rótulo possível. Previamente à execução, edita-se o arquivo de configurações do PreTexT (*parameters.cfg*). O usuário pode modificar parâmetros de normalização, métodos de valoração das palavras de cada texto, entre outros.

A utilização do PreTexT é realizada pela execução dos módulos *Stem* e *Report* (*scripts* em linguagem Perl — *stem.pl* e *report.pl*). O módulo de *Stem* transforma cada palavra na sua forma raiz, sendo computada uma freqüência de cada *stem* em relação ao documento e à coleção de documentos. Os resultados dessas estatísticas são gravados em arquivos textos no sub-diretório *stembase*. Com esses valores estipulados, a matriz BOW é construída pela execução do módulo *Report*. Essa etapa final calcula o valor de cada termo pelas técnicas *Boolean*, TF, TFIDF ou TFLINEAR.

De acordo com a Tabela 3.2, verifica-se que ocorre um desbalanceamento entre os rótulos e número de mensagens. A alternativa encontrada foi a utilização das idéias de *oversampling* e *undersampling*. Como ponto de partida foi determinado um número máximo para cada classe, utilizando-se da classe OUTROS, por ser esta a de característica mais abrangente. A categoria OUTROS possui 4.886 mensagens, sendo fixado o valor inicial de 4.500. Com relação às categorias com número de mensagens inferior ao valor estipulado, foram realizadas cópias das mensagens para equalizar o número de mensagens dos demais rótulos. Para completude dos testes, foram disponibilizados conjuntos de dados menores com decremento de 500 mensagens por rótulo em cada passo.

Para cada conjunto de dados do lote, a execução do módulo *Stem* do PreTexT, conforme apresentado na Figura 2.9, explorou as opções paramétricas disponíveis no arquivo de configurações. A Tabela 3.3 mostram todas as combinações possíveis utilizadas. As

combinações matematicamente possíveis não foram adotadas por não ter sentido escolher, por exemplo, uma medida booleana ou *term-frequency* com parâmetro *smooth* habilitado. No módulo *Report* do PreTexT, para cada conjunto de dados, foi obtida uma matriz BOW. O lote de conjunto de dados foi transformado em um lote de matrizes, em que cada matriz foi codificada segundo diversas perfis de configurações de valoração e normalização dos atributos. A matriz é descartada caso ocorram problemas de tempo de resposta no pré-processamento ou tamanho não suportado para classificação supervisionada no Weka.

Na Figura 3.2, Etapa 4, a justificativa para utilização da ferramenta Weka está na variedade dos algoritmos de mineração e facilidade de utilização. O sistema possui componentes de fácil entendimento com algoritmos e filtros comumente conhecidos. Embora haja restrições de velocidade e memória pela utilização de máquina virtual Java, outras soluções semelhantes também adotam o Java, como por exemplo o *Rapid Miner*.

Para determinação dos parâmetros possíveis, pode-se utilizar o módulo *Explorer* no Weka. O usuário pode analisar um dado algoritmo com variações em seus parâmetros para um conjunto de dados. O carregamento da matriz permite início da classificação, sendo possível visualizar os atributos por meio de uma tabela na qual consta o número da coluna e o nome do atributo.

As funcionalidades de pré-processamento no Weka também permitem relatórios estatísticos acerca de cada atributo. As medidas estatísticas de mínimo, máximo, desvio-padrão, média, inclusive um gráfico de histograma podem ser visualizados. É possível aplicar filtros diversos para normalização, transformação dos dados contidos nos atributos. O Weka aceita entrada de dados em diversos formatos, sendo utilizado o formato *comma-separated values* (CSV) gerado pelo PreTexT.

Uma vez carregada em memória a matriz de dados, é possível iniciar a etapa de configuração da classificação supervisionada. A configuração exige a escolha de um algoritmo classificador. Nessa proposta de solução, apenas três algoritmos das famílias *Trees*, *Bayes* e *Function* são utilizados. O usuário pode escolher também um classificador especial que realiza otimização de parâmetros em que é fornecido um valor inicial e final, com o número de incrementos desejável (*CVParameterSelection*).

A solução adotada com uso do J48, *Naïve Bayes* e SMO passarão por etapas de treinamento e teste. Na fase de treinamento, o modelo é induzido pelo conjunto de dados fornecido. Na fase de testes, é realizada uma validação cruzada em 10 dobras conforme descrito na Seção 2.2.3. Esse valor é empregado frequentemente nas pesquisas em classificação automática de documentos.

A configuração de parâmetros é selecionada pela análise da porcentagem de erros cometida e *F-measure*. Esse experimento inicial termina com um perfil ideal de cada algoritmo estudado. O próximo experimento é investigar o algoritmo com melhor resultado dentre aqueles já otimizados, utilizando o módulo *Experimenter* do Weka.

O módulo *Experimenter* permite que se valha de vários conjuntos de dados e algoritmos em um único experimento. A escolha do melhor método é realizada por meio de um teste exaustivo que inclui um lote de conjuntos de dados (obtidos pelos diferentes parâmetros no PreTexT), algoritmos classificadores com seus respectivos parâmetros e a estratégia de teste e treinamento adotados.

As configurações do ambiente *Experimenter* informam os testes automatizados com diversos conjuntos de testes e algoritmos. Para fins de análise, poderão ser escolhidas

repetições para acúmulo de dados estatísticos. A estratégia de treinamento e teste inclui a validação cruzada em n dobras e *holdout*.

Os resultados dos experimentos ficam registrados em um arquivo indicado pelo usuário. A ferramenta dispõe de um formulário de consulta para análise dos resultados obtidos. O relatório de resultados é mostrado de forma resumida, comparando os resultados obtidos dos algoritmos contidos na lista. O processo de classificação é encerrado pela rotulagem dos *e-mails* na caixa eventos categorizados na Figura 3.2, Etapa 1.

	Parâmetros do PreTexT								
	Measure				*Normalize*			*Smooth*	
# do Perfil	Boolean	TF	TFIDF	TFLINEAR	Desativado	Linear	Quadrático	Desativado	Ativado
1	•				•			•	
2	•					•		•	
3	•					•			•
4	•						•	•	
5	•						•		•
6		•			•			•	
7		•				•		•	
8		•				•			•
9		•					•	•	
10		•					•		•
11			•		•			•	
12			•			•		•	
13			•			•			•
14			•				•	•	
15			•				•		•
16				•	•			•	
17				•		•		•	
18				•		•			•
19				•			•	•	
20				•			•		•

Tabela 3.3: Tabela de configuração dos parâmetros PreTexT utilizados para cada conjunto de dados.

Capítulo 4

Experimentações e Resultados

Neste capítulo, serão apresentados os experimentos realizados com o corpo de mensagens recebidos pelo CTIR Gov no período de janeiro a agosto de 2009.

Descreveremos os procedimentos de forma mais detalhada, a saber:

1. Elaboração da base de textos, contendo *e-mails* e seus respectivos rótulo;

   A base de textos compreende *e-mails* em português recebidos pelo CTIR Gov entre janeiro e agosto de 2009. O número de mensagens coletadas totaliza 102.260 *e-mails*, dispostas em 8 caixas postais, conforme apresentados na Tabela 3.2. As caixas postais são internamente conhecidas por: DFL, NAIVTW, NAI_RECEPCAO, NAIIDS_RECEPCAO, NAIIDS_TX, CTIR, NAILOGS_TX, UND. As mensagens são classificadas em 6 rótulos (DFL, VTW, IDS, Outros, TDI, *Undeliverable*), de acordo com classificação seguida pelo CTIR Gov.

   A base de textos para a próxima etapa foi balanceada utilizando técnicas de *over-sampling* para caixas postais com poucos exemplos e *undersampling* para caixas postais com excesso de mensagens. O balanceamento é realizado por meio de seleção das mensagens escolhidas para evitar repetições de conteúdo e possibilitar execução das ferramentas nas etapas subseqüentes. Foram descartados anexos em forma de artefatos, sejam eles figuras, executáveis, documentos estruturados em formatações especiais, planilhas, entre outros.

2. Execução da ferramenta PreTexT para pré-processamento de texto;

   Estabelecida a coleção de documentos amostrados, a execução da ferramenta PreTexT realizará as tarefas de radicalização, remoção de *stopwords*, normalização de valores (linear ou quadrática), aplicação das técnicas Booleana, TF, TFIDF e TFLINEAR. Como resultado dessa etapa, serão obtidas as matrizes representadas pela estrutura BOW utilizando combinação de todos parâmetros do PreTexT.

3. Execução da ferramenta Weka em modo *Experimenter* e *Explorer*, utilizando os algoritmos J48, *Naïve Bayes* e SMO;

   A execução dos algoritmos escolhidos será realizada nos conjuntos de dados obtidos da etapa anterior. Alguns algoritmos poderão ter seus parâmetros padrões modificados para avaliação de resultados. A escolha do conjunto de testes e treinamento será realizada por validação cruzada em 10 dobras conforme justificado no Capítulo

3. Diante dos melhores resultados obtidos, os parâmetros de pré-processamento e classificação serão anotados. Os modelos de classificação obtidos com os parâmetros anotados deverão ser avaliados, fornecendo um conjunto de testes desbalanceado e semelhante ao tráfego de mensagens observado no CTIR Gov.

4. Avaliação dos resultados, utilizando métricas de porcentagem corretamente classificada e tempo de processamento;

   Para a avaliação dos resultados, serão utilizadas as métricas de porcentagem corretamente classificada e tempo de processamento, com objetivo de escolher o melhor algoritmo com seus parâmetros de pré-processamento e classificação.

Na Tabela 4.1 são apresentados os experimentos preliminares os quais levaram a restrições em que conjuntos de dados com mais de 3.000 mensagens (500 mensagens por rótulo) foram descartados por motivos de tempo computacional despendido, alcançando o limite de memória do computador utilizado. Verificou-se que um conjunto de dados volumoso não é possível de ser trabalhado pela abordagem adotada. Entre 9.000 e 27.000 mensagens (1.500 e 4.500 mensagens por rótulo), houve demora de 1 a 3 horas na execução do PreTexT.

Outro viés prático foi o tamanho dos dados obtidos. Na faixa de mensagens mencionada, as matrizes geradas têm tamanho entre 0,234 a 2,6 *gigabytes* (GB). Na faixa de amostragem entre 9.000 e 3.000 mensagens (1.500 a 500 mensagens por rótulo), houve problemas na limitação de memória volátil utilizada. A memória disponibilizada ao Weka em todos experimentos foi de 2,5 GB valendo-se de um computador Intel Pentium 4 com processador de 3 GHz, 3 GB de memória RAM e sistema operacional Ubuntu 8.04. As escolhas de conjuntos de dados de possível manipulação ficaram por volta de 2.400 mensagens (400 mensagens por rótulo).

Como resumo dos experimentos preliminares supracitados foram estipulados cerca de 17 conjunto de dados, de 4.500 a 100 mensagens por rótulo em decrementos de 500 mensagens por rótulo. A coluna Uso indica se o conjunto de dados foi utilizado. A coluna Número *E-mails* significa o número total de mensagens utilizadas. Na terceira coluna, o número de mensagens por rótulo. A coluna Tamanho *Dataset*, o tamanho em disco rígido de cada conjunto de dados. A coluna PreTexT, o tempo de processamento gasto. A última coluna, se foi possível utilizar os algoritmos de classificação com o Weka.

Uso	Número *E-mails*	Mensagem/Rótulo	Tamanho *Dataset*	PreTexT	Weka
Não	27.000	4.500	2,6 GB	3h	Não
Não	9.000	1.500	234 MB	1h	Não
Sim	3.000	500	$<$ 120 MB	10 min	Sim
Sim	2.400	400	$<$ 120 MB	10 min	Sim

Tabela 4.1: Limitação de mensagens obtidas pelo uso das ferramentas adotadas.

Na seção 4.1, o experimento tem como objetivo a determinação de parâmetros possíveis para cada algoritmo estudado. Esse experimento emprega um conjunto de dados reduzido para cada classificador. No classificador J48, é ajustado o parâmetro C que controla os cortes realizados nas árvores. No classificador *Naïve Bayes*, é escolhido o parâmetro *useKernelEstimator*. Essa escolha trata os atributos numéricos com um *kernel*,

sem assumi-los como uma distribuição normal. A versão *Naïve Bayes Multinomial* orientada para classificação de textos também é incluída no rol de opções. No classificador SMO, utiliza-se o parâmetro de complexidade C. Esse parâmetro controla a taxa de erros nos dados de treinamento em relação ao tamanho das margens estipuladas, conforme visto na Seção 2.2.3, *Soft-Margin SVM*. Também foram utilizados os *kernels* linear e RBF no algoritmo SMO.

Na Seção 4.2, o experimento busca avaliar uma configuração de pré-processamento possível para execução com cada algoritmo escolhido no ensaio anterior. O espaço de busca será realizado nos conjuntos de dados gerados pela combinação dos parâmetros na ferramenta PreTexT. A Tabela 3.3 mostra as combinações de valoração dos atributos e normalização utilizadas. O perfil de pré-processamento com menor custo computacional e melhor resultado de classificação será utilizado para o experimento final desta pesquisa.

Na Seção 4.3, o experimento busca a escolha do algoritmo com melhor resultados de classificação. Esse algoritmo escolhido será avaliado por meio de um conjunto de teste nunca antes aprendido pelo modelo. Os resultados obtidos serão comparados aos relatórios do mês mais recente recebido do CTIR Gov (agosto), verificando os possíveis erros percentuais entre uma classificação de aprendizado por máquina e outra gerada pelos analistas.

4.1 Experimento para Busca de Parâmetros Possíveis

O primeiro experimento tem como objetivo a determinação de parâmetros possíveis para os algoritmos estudados. Será utilizado um conjunto de dados menor, com parâmetros de pré-processamento mais simples possíveis. O objetivo é estipular para cada algoritmo estudado os parâmetros modificados que apresentam melhores resultados em relação à porcentagem de acertos.

Um conjunto de dados com 300 mensagens foi amostrado e pré-processados no PreTexT. Todos os rótulos foram utilizados e cada rótulo contabilizado com 50 mensagens. Os parâmetros empregados foram o *Term-Frequency* (TF), sem normalização.

No Weka, utilizou-se o módulo *Explorer*, escolhendo o otimizador *CVParameterSelection* para a classificação supervisionada utilizando J48, *Naïve Bayes* e SMO. O otimizador busca o melhor parâmetro de acordo com o valor inicial, final e incrementos fornecidos pelo usuário. A opção de teste selecionada foi a validação cruzada em 10 dobras. Essa estratégia foi escolhida porque é frequentemente utilizada em classificação de documentos. No campo de resultados, é disponibilizado o valor do parâmetro com melhor avaliação.

4.1.1 Algoritmo J48

O classificador J48 pode ser melhor ajustado pelo parâmetro C, que controla os cortes realizados na árvore de decisão obtida. Quanto menor a medida, maior o número de cortes na árvore. O tamanho do conjunto de dados contém 300 mensagens. Todos rótulos foram utilizados e cada rótulo com 50 mensagens. Os parâmetros do PreTexT foram ajustados com parâmetro TF habilitado, sem normalização. Valores do intervalo $C \geq 0,5$ não foram computados devido limite imposto pela ferramenta.

C=0,1	C=0,2	C=0,3	C=0,4	C=0,5
95,32%	94,98%	94,98%	94,98%	94,98

Tabela 4.2: Porcentagem de casos corretos classificados, variando valor de C para J48.

4.1.2 Algoritmo *Naïve Bayes*

O classificador *Naïve Bayes* (NB) permite que se utilize o parâmetro estimador de *kernel*, além da versão *multinomial*.

Algoritmo	Classificação
NB	**94,65%**
NB *Kernel Estimator*	91,64%
NB *Multinomial*	80,94%

Tabela 4.3: Porcentagem de casos classificados corretamente com modificação de parâmetros no algoritmo *Naïve Bayes*.

4.1.3 Algoritmo SMO

O classificador SMO pode ser otimizado por meio do parâmetro de complexidade C além das opções de *kernel* linear e RBF.

Algoritmo	C=0,01	C=0,1	C=1	C=10	C=100	C=1000
SMO Linear	87,96%	94,65%	**95,99%**	95,99%	95,99%	95,99%
SMO RBF	32,85%	74,92%	89,98%	93,99%	**95.32%**	95,32%

Tabela 4.4: Porcentagem de casos classificados corretamente com modificação do parâmetro C, *kernel* linear e RBF no algoritmo SMO.

4.1.4 Análise dos Resultados

Os testes realizados mostraram que nem sempre os valores padrões são as melhores escolhas. No classificador J48, a estimativa de C (valor de confiança) nada mais é que uma porcentagem que indica um limite superior para erros. O cálculo de C utiliza o número de instâncias de treinamento e o número de erros cometidos. À medida que a árvore de decisão é ramificada, para cada ramo é calculada uma taxa de erros. Caso esse ramo chegue ao limite permissível, as operações de corte são realizadas. Verificou-se pelos dados experimentais na Tabela 4.2 que mesmo variando os parâmetros C, a diferença nos resultados variou levemente 0,33% (95,32%–94,98%). No entanto, um valor limite muito pequeno para C pode indicar uma baixa tolerância a erros, a qual confrontada com um conjunto de testes com instâncias novas pode indicar sinais de uma árvore super-ajustada. Na Tabela 4.2 o parâmetro $C = 0,1$ mostrou o melhor resultado (95,32%).

No algoritmo *Naïve Bayes*, os resultados foram contrários ao que se esperava de acordo com os artigos pesquisados. A versão *Multinomial* é considerada como a melhor escolha

em casos de classificação de textos. O conjunto de dados utilizados possui certas peculiaridades como o alto número de mensagens únicas que podem ter influenciado esse resultado desfavorável. A utilização de um *kernel* estimador também não obteve resultado esperado, provavelmente a distribuição modelada não se encaixou à realidade dos atributos numéricos. De acordo com a Tabela 4.3, o algoritmo *Naïve Bayes* obteve melhor classificação (94,65%).

Para o algoritmo SMO, foram obtidos resultados igualmente bem classificados. O parâmetro $C = 1$ (valor padrão) no *kernel* linear foi o que obteve o melhor resultado (95,99%). O *kernel* RBF também mostrou resultados tão bons quanto o linear no valor $C = 100$. Entre eles, houve uma leve diferença de apenas $0,67\%$ (95,99%–95,32%). É interessante notar que mesmo aumentando a ordem de grandeza do parâmetro C, os resultados foram iguais aos alcançados com $C = 1$. Esses dados indicam que as margens separadoras provavelmente são distantes, por isso não influenciam seus resultados. O *kernel* RBF mostrou resultados menores que o *kernel* linear, embora tenha um custo computacional maior.

Comparando o tempo de execução dos três algoritmos, percebemos que o algoritmo J48 foi o mais rápido, seguido do *Naïve Bayes* e do SMO. O único viés no algoritmo SMO é o tempo gasto para treinamento que se mostrou mais demorado que os demais. Diante das Tabelas 4.2, 4.3 e 4.4, percebemos que os parâmetros possíveis para o experimento são: (i) J48 com parâmetro $C = 0,1$; (ii) *Naïve Bayes* e (iii) SMO com *kernel* linear e parâmetro $C = 1$.

4.2 Experimento nos Parâmetros do PreTexT

Conforme citado, o segundo experimento tem como objetivo a busca pelos melhores parâmetros de pré-processamento. Foram disponibilizados 20 conjuntos de dados conforme as combinações mostradas nos perfis da Tabela 3.3. Foi utilizado um total de 300 mensagens (50 mensagens por rótulo), contabilizando cada rótulo com 50 mensagens, para geração da matriz BOW.

Em seguida, utilizou-se o Weka *Experimenter* para avaliar: (i) as matrizes BOW cada qual configurado no PreTexT com perfil de 1 a 20 conforme a Tabela 3.3, e (ii) lista de algoritmos, J48 com parâmetro $C = 0,1$ e $C = 0,25$; *Naïve Bayes*, incluindo a versão *Multinomial* e o SMO com *kernels* linear e RBF com parâmetros $C = 1$ e $C = 100$ respectivamente. Para evitar o super-ajustamento, foi realizada a validação cruzada em 10 dobras.

A idéia principal desse experimento objetiva verificar a influência do tipo de perfil empregado na ferramenta PreTexT. Estudos das técnicas de pré-processamento indicam que a técnica TFIDF e TFLINEAR utilizadas com normalização linear ou quadrática são fatores que podem melhorar os resultados de mineração.

Seguem, na Figura 4.1, os resultados obtidos. O eixo das abscissas representa as configurações adotadas na Tabela 3.3. O eixo das ordenadas mostra a porcentagem de casos corretamente classificados de acordo com os algoritmos J48 ($C = 0,1$ e $C = 0,25$), *Naïve Bayes* (NB), *Naïve Bayes Multinomial* (NBM), SMO Linear ($C = 1$) e SMO RBF ($C = 100$).

A Figura 4.2 mostra em escala ampliada todos algoritmos exceto NB. Por fim, a Figura 4.3 mostra os tempos de treinamento gastos para cada classificador.

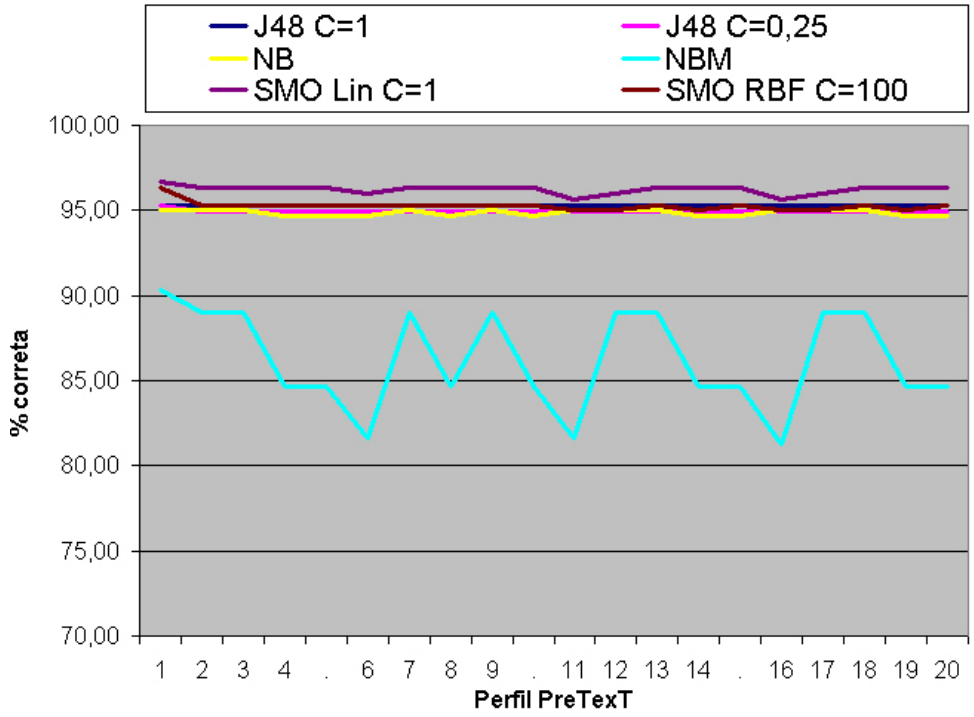

Figura 4.1: Porcentagem de casos classificados corretamente com os algoritmos J48, *Naïve Bayes*, NB *Multinomial* e SMO, utilizando conjuntos de dados com diversos perfis de configuração do PreTexT.

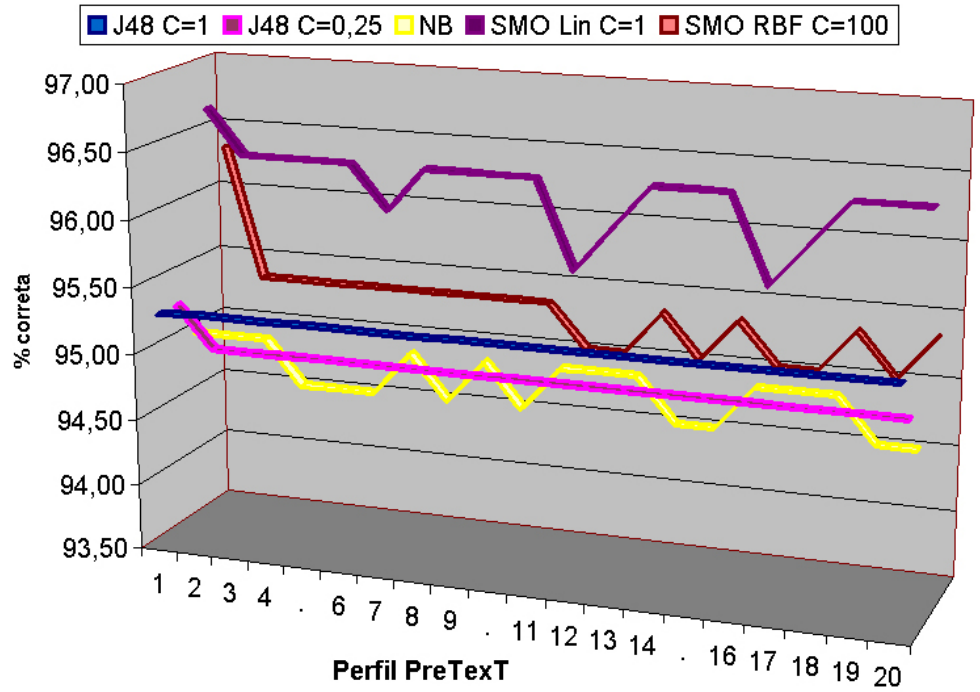

Figura 4.2: Aproximação de escala para os algoritmos utilizados exceto *Naïve Bayes Multinomial*

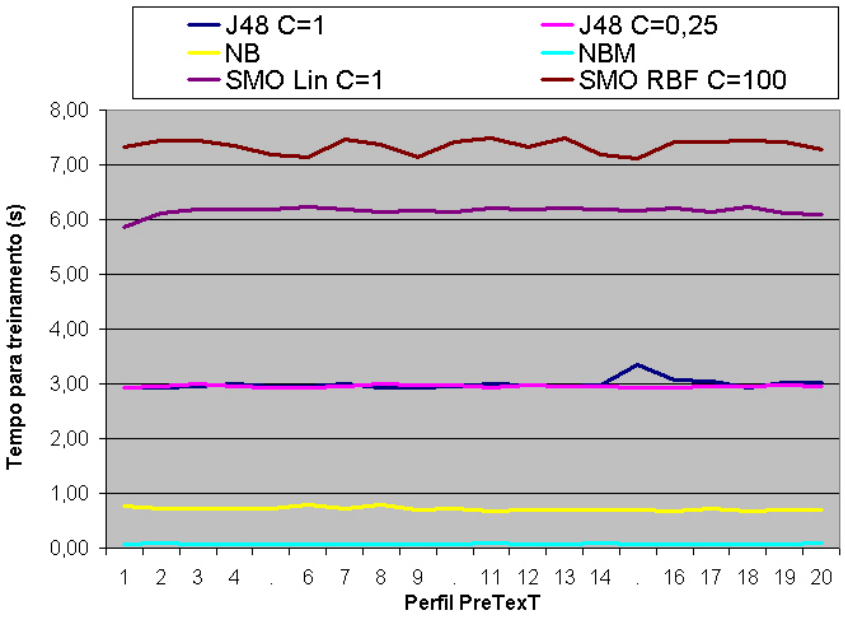

Figura 4.3: Porcentagem de casos classificados corretamente com os algoritmos J48, *Naïve Bayes* (NB) e SMO, utilizando conjuntos de dados com diversos perfis de configuração do PreTexT.

4.2.1 Análise dos Resultados

A Figura 4.1 mostra resultados interessantes para os algoritmos J48 e *Naïve Bayes*. Com relação ao algoritmo J48 com $C = 0,1$, verifica-se que o resultado independe do perfil de pré-processamento realizado. Os resultados para J48 com $C = 0,25$, valor padrão no Weka, mostram resultados piores em relação a $C = 0,1$. Portanto, o algoritmo J48 aceita qualquer tipo de configuração no pré-processamento do PreTexT. Para uso prático, poderia ser utilizado o conjunto de dados booleano, com normalização linear, que possui tempo de execução menor se comparado aos demais perfis listados.

Referentes ao *Naïve Bayes* (NB) e a versão *Multinomial* (NBM), confirma-se como no experimento anterior que a versão *Multinomial* realiza mais erros de classificação que o algoritmo padrão. Os melhores resultados obtidos (coluna **C**) mostram uma tendência de padrão ideal de pré-processamento. Os resultados com maiores valores geralmente estão associados aos métodos de valoração: booleano; TF; TFIDF; TFLINEAR. No entanto, verifica-se que a normalização quadrática diminui o percentual para $94,67$. O perfil para o *Naïve Bayes*, o qual configura o menor custo computacional com a maior taxa de acerto, é a utilização de contagem booleana com normalização linear.

Os algoritmos SMO mostram o melhor resultado para o *kernel* linear com $C = 1$. Os resultados do *kernel* RBF com $C = 100$ seguiram a mesma tendência indicada pelo experimento anterior. No *kernel* linear, nos maiores valores obtidos, pode-se perceber uma tendência à preferência por normalização linear ou quadrática. Perfis de pré-processamento sem normalização mostraram valores menores de percentuais corretamente classificados.

Os resultados de *F-measure* para esse experimento, entre 0,97 a 0,99 não representa valor significativo, tendo em vista a pequena amostra no total de 3.000 mensagens. Quanto ao tempo de processamento gasto pelos algoritmos, a Figura 4.3 apresenta os resultados,

em que o *Naïve Bayes* é o mais rápido, logo após vêm o J48, e por último o SMO. Comparando esses tempos entre J48 e SMO, existem diferenças relevantes. Em todos os conjuntos de dados, o SMO demora duas vezes mais que o J48. Uma mesma comparação entre J48 e NB mostra o algoritmo J48, sendo até duas vezes mais demorado que o NB. Nota-se que o *Naïve Bayes* é o algoritmo mais rápido dentre todos os avaliados.

4.3 Experimento de Classificação com Conjunto Teste Fornecido

Conforme citado anteriormente, o terceiro experimento tem como objetivo a avaliação dos resultados obtidos, fornecendo um conjunto de teste nunca antes aprendido pelo algoritmo J48. O conjunto de dados de treinamento totalizou 2400 mensagens, com 400 mensagens por rótulo, e o conjunto de teste totalizou 600 mensagens (100 mensagens por rótulo).

Um segundo conjunto de testes foi amostrado de forma a manter o desbalanceamento mostrado na Tabela 3.2. Foram escolhidas mensagens eletrônicas tramitadas no mês de julho de 2009. Foi necessário separar uma amostra com um total de 3.181 mensagens, as quais estavam desbalanceadas entre os rótulos. As mensagens de teste não estiveram presentes durante o treinamento, podendo ser consideradas como um caso não aprendido anteriormente. Cada rótulo totalizou um certo número de mensagens, a saber:

- VTW: 1187 mensagens;
- IDS: 677 mensagens;
- DFL: 592 mensagens;
- OUTROS: 452 mensagens;
- UND: 214 mensagens;
- TDI: 59 mensagens.

O Weka *Explorer* foi utilizado nesse experimento para obter informações mais detalhadas dos resultados do algoritmo J48. O modelo gerado minimizou os erros possíveis de classificação, utilizando validação cruzada em 10 dobras. O modelo foi novamente carregado em memória sendo reavaliado nos conjuntos de testes mencionados.

A Tabela 4.5 mostra a matriz de confusão obtida pelo primeiro conjunto de testes. Uma classificação correta deve ter valores dispostos na diagonal principal da matriz de confusão. Espera-se que ocorram erros maiores no rótulo OUTROS por causa de seu escopo de definição, com mínimos erros nas mensagens pertencentes aos outros rótulos.

As colunas R_1 a R_6 da Tabela 4.5 indicam os resultados classificados pelo algoritmo J48. As linhas indicam o número real de exemplos associados a um dado rótulo. Por exemplo, no rótulo DFL existem 100 mensagens, enquanto o modelo aprendido classificou 101 mensagens como pertencentes ao rótulo DFL.

Adicionalmente, na Tabela 4.5, na linha do rótulo IDS, por exemplo, existem 100 mensagens realmente pertencentes à classificação IDS ($= 82+18$) e o modelo gerado, conforme coluna R_2, classificou 96 mensagens como sendo IDS ($= 82 + 8 + 6$). Se analisarmos as 96 mensagens classificadas pelo modelo, veremos que 82 mensagens foram corretamente

classificadas, mas 8 deveriam pertencer ao rótulo UND e 6 mensagens deveriam pertencer ao rótulo VTW.

R_1	R_2	R_3	R_4	R_5	R_6	← Classificado como
100	0	0	0	0	0	R_1=DFL
0	82	0	18	0	0	R_2=IDS
0	0	97	0	2	1	R_3=OUTROS
0	0	0	100	0	0	R_4=TDI
1	8	9	2	75	5	R_5=UND
0	6	5	1	0	88	R_6=VTW

Tabela 4.5: Matriz de confusão obtida pelo primeiro teste no algoritmo J48.

O Weka *Experimenter* foi utilizado nessa outra prospecção para verificar questões de pré-processamento em um conjunto de dados com total de 2400 mensagens. Os conjuntos de dados foram configurados nos métodos booleano, TF e TFIDF, todos eles com normalização linear. Os algoritmos avaliados foram o J48 com $C = 0,1$; J48 com $C = 0,25$; NB; NB *Multinomial*; SMO linear com $C = 1$ e SMO RBF com $C = 100$.

Na Tabela 4.6, estão registrados os valores de porcentagem de instâncias corretamente classificadas. As colunas da Tabela 4.6 representam os algoritmos J48 $C = 0,1$; J48 $C = 0,25$; *Naïve Bayes* (NB), *Naïve Bayes Multinomial* (NBM); SMO Linear $C = 1$; SMO RBF $C = 100$. As linhas da Tabela 4.6 mostram os conjuntos de dados configurados de acordo com perfil 2 — contagem booleana com normalização linear; perfil 7 — contagem TF com normalização linear; e perfil 11 — contagem TFIDF com normalização linear. Esses perfis foram escolhidos de acordo com as tendências observadas no experimento anterior.

Nº Perfil	$J48_{C=0,1}$	$J48_{C=0,25}$	NB	NBM	$SMO_{Lin_{C=1}}$	$SMO_{RBF_{C=100}}$
2	99,25	99,25	97,71	97,00	98,92	98,04
7	99,25	99,25	97,71	97,00	98,92	98,04
11	99,25	99,25	97,75	97,00	98,92	98,04

Tabela 4.6: Porcentagem de casos corretamente classificados com o algoritmos J48, NB e SMO, utilizando configurações do PreTexT com perfis 2, 7 e 11.

Finalmente, foi utilizado o segundo conjunto de testes com classes desbalanceadas. Essa avaliação mostrou resultados de 73,05% de instâncias corretamente classificadas, *F-measure* de 0,754 e matriz de confusão conforme a Tabela 4.7

4.3.1 Análise dos Resultados

Na Tabela 4.6, o percentual de instâncias classificadas corretamente no algoritmo J48 com parâmetro $C = 0.1$ foi de 99,25% (Coluna **A**) na etapa de treinamento. O valor caiu relativamente pouco na fase de testes sendo obtido 90,33% ($= \frac{100+82+97+100+75+88}{600}$) de classicações corretas de acordo com a Tabela 4.5. A diferença no valor *F-measure* alcançado entre as fases foi de apenas 0,09. O tempo de processamento no aprendizado do modelo foi satisfatório e rapidamente convergiu para o resultado.

R_1	R_2	R_3	R_4	R_5	R_6	← Classificado como
540	0	52	0	0	0	R_1=DFL
0	644	2	31	0	0	R_2=IDS
1	0	430	4	16	1	R_3=OUTROS
0	1	0	58	0	0	R_4=TDI
0	3	2	0	208	1	R_5=UND
0	0	862	16	0	809	R_6=VTW

Tabela 4.7: Matriz de confusão obtida pelo segundo conjunto de testes no algoritmo J48.

A avaliação do classificador J48 não foi favorável no conjunto de testes desbalanceado. O percentual de instâncias corretamente classificadas totalizou 73,05%. Os resultados de *F-measure* foram 0,95; 0,97; 0,47; 0,69; 0,95; 0,648 correspondendo respectivamente aos rótulos $R_1 - - R_6$ da Tabela 4.7. A matriz de confusão apresentada na Tabela 4.7 confirmou os piores resultados para os rótulos OUTROS, TDI e VTW. A diferença de resultados entre as fases de testes e treinamento com diversos conjuntos de dados adotados leva a se pensar em super-ajustamento, o que talvez explique a baixa variação com os parâmetros e com os algoritmos.

Na prospecção realizada pelo *Experimenter*, o conjunto de dados de 2.400 mensagens mostrou resultados interessantes. Verificou-se que as métricas booleana, TF e TFIDF com normalização linear não influíram nos resultados de classificação. O método *Naïve Bayes Multinomial* conseguiu pior resultado que o *Naïve Bayes* com diferença de 0,71% $(97,71\% - 97,00\%)$ conforme valores da Tabela 4.6, coluna NB e NBM.

Inesperadamente o parâmetro $C = 0,1$ e $C = 0,25$ no algoritmo J48 não obteve diferença significativa de resultados. Nos algoritmos SMO, os resultados foram melhores para o *kernel* linear. A diferença entre os *kernels* linear e RBF foi de apenas 0,88% $(98,92\%–98,04\%)$, obtidos pelas duas últimas colunas na Tabela 4.6.

Comparando os três algoritmos, os resultados de J48 foram os melhores, seguidos do SMO e *Naïve Bayes*. O conjunto de dados utilizados nessa avaliação foi maior e notavelmente o algoritmo SMO foi o mais demorado para obter resultados. Logo após o SMO, vieram os algoritmos J48 e *Naïve Bayes*. Os três conjuntos de dados e os seis algoritmos escolhidos foram testados com validação cruzada em 10 dobras. Esse experimento maior durou um tempo aproximado de 8 horas de processamento.

Com os resultados supracitados, o algoritmo J48 mostrou-se como melhor opção, com melhores resultados de classificação e um dos menor tempo de execução.

Capítulo 5

Conclusões e Trabalhos Futuros

Este capítulo discute as conclusões relativas ao trabalho de pesquisa, mostradas na Seção 5.1 e descreve trabalhos futuros de pesquisa, os quais podem ser desenvolvidos com os resultados obtidos até o momento na Seção 5.2.

5.1 Conclusões

Neste trabalho, foram realizados estudos nas áreas de Gestão de Incidentes e mineração de textos que, apesar de serem áreas de conhecimento distintas, possuem ligações pertinentes acerca dos problemas a serem solucionados. Um estudo de processos foi realizado no CTIR Gov, verificando a conformidade com o modelo padrão adotado pelos CSIRTs. Avaliou-se também qual o processo poderia ser melhorado, utilizando técnicas automatizadas, como, por exemplo, técnicas de mineração de textos. A tendência de crescimento das mensagens recebidas pelo CTIR Gov indicou uma possível solução de pesquisa aplicada na categorização de eventos de segurança recebidos pelo centro, as quais são feitas por meio de filtros programados.

O processo de Gestão de Incidentes de Segurança da informação pode ser definido pelas atividades de detecção, triagem, resposta, planejamento, proteção, preparação, manutenção e melhoria. Eventos diversos são detectados por agentes dotados de inteligência. A inteligência vêm de pessoas, sensores programados para disparo de alarmes diante de certas condições e até mesmo de padrões derivados da coletividade de usuários na Internet.

Os processos precisam de uma organização bastante funcional de modo que o problema maior de tratamento de incidentes possa ser resolvido dentro de uma metodologia aplicável, rastreável e ágil. Dentro dessas restrições, a Internet cresce, o volume de dados e informações aumenta, sendo que o número de atividades ilícitas também seguem essa tendência. Mesmo em um ambiente adverso, uma equipe de tratamento de incidentes deve conseguir atingir seu objetivo e missão. Nesse cenário, as ferramentas passam a ser essenciais para o pleno funcionamento desse *framework*.

O caso do CTIR Gov é bastante peculiar, pois se trata de um centro de coordenação de incidentes de segurança na Administração Pública Federal. Muitos dos processos previstos em teoria acontecem distribuidamente, em diversas outras equipes de tratamento de incidente. A detecção ocorre em sua maioria em um ambiente exterior ao centro, uma vez que a triagem é o primeiro passo nas operações do CTIR Gov. A triagem é importante, pois organiza a massa de mensagens recebidas. Os passos seguintes compreendem o cor-

relacionamento e priorização. A categorização é um problema recorrente na mineração de dados. Especificamente, a categorização de textos que é uma sub-área dessa mineração.

Os objetivos gerais deste trabalho foram atingidos com sucesso. A proposta de classificação supervisionada foi definida conforme o modelo de solução utilizado. Os testes dos algoritmos classificadores obtiveram êxito conforme os experimentos realizados neste trabalho. Quanto a taxa de acerto de 73%, não foi possível obter um resultado semelhante à classificação semi-automática realizada no CTIR Gov. Verifica-se que quanto maior o conjunto de teste, mais distante ele fica do modelo de classificador gerado do conjunto de treinamento, e conseqüentemente, os erros de classificação são propagados.

Os objetivos secundários também foram alcançados com êxito. Os experimentos realizados permitiram um entendimento maior do processo de mineração com restrições e limitações práticas. O entendimento dos resultados obtidos foram possíveis mediante um conhecimento prévio dos algoritmos e ferramentas empregadas neste trabalho. A utilização de textos em língua portuguesa foi um fator limitante porque grande parte das mensagens veiculadas por *e-mails* são escritas em várias linguagens e isso dificulta a redução de dimensionalidade.

A abordagem de classificação de documentos adotada neste trabalho englobou áreas de processamento natural de linguagem e métodos de categorização supervisionada. Os métodos adotados confrontaram dois métodos simples, como o J48 e *Naïve Bayes* contra outro mais complexo, o SMO. No pré-processamento, o problema de representação dos documentos, redução de dimensionalidade e normalização puderam ser resolvidos utilizando o *software* PreTexT. O PreTexT permitiu obter matrizes representadas no formato BOW, e realizou radicalização em documentos na língua portuguesa e com normalização dos dados. O *software* se mostrou bastante útil, de fácil utilização, e capaz de tratar conjuntos de dados de tamanhos variados, mas apresentando baixo desempenho.

Na mineração de textos, a ferramenta Weka se mostrou bastante limitada para uso de matrizes esparsas com amostras mais reduzidas em número de instâncias e atributos. A vantagem de utilização desse ambiente foi verificada na possibilidade de comparação entre diversos algoritmos classificadores. O Weka provê uma interface amigável, foi capaz de tratar casos multi-classe, permitiu visualização gráfica do conjunto de dados e provê resultados estatísticos comparativos. A principal desvantagem refere-se à limitação de memória e a baixa escalabilidade, pois o conjunto de dados necessita estar presente em memória, reduzindo a possibilidade de tratar grandes volumes de dados.

Os algoritmos J48 e *Naïve Bayes* indicaram execução rápida se comparada à técnica SMO. Em certos perfis de pré-processamento PreTexT e parâmetros dos algoritmos no Weka, o classificador SMO se mostrou com resultados tão bons quanto os primeiros. Para utilização prática desse caso real, o J48 se apresentou como uma boa escolha porque possui tempo de execução menor se comparados ao tempo exigido pelo SMO. Além disso, demonstrou porcentagem de instâncias corretamente classificadas comparativamente superiores. As métricas de porcentagem de instâncias corretamente classificadas e tempo de execução se mostraram relevantes nas escolhas do processo de mineração. Infelizmente não foi possível utilizar a métrica *F-measure* para comparar resultados entre os algoritmos, pois apresentaram diferenças muito pequenas da ordem de $10^{-}1$.

Os resultados de *F-measure* se revelaram bastante semelhantes, sendo que foram mais relevantes as métricas de tempo de execução e porcentagem de instâncias corretamente classificadas.

Finalmente, o teste do modelo aprendido por meio de um conjunto de teste fornecido denotou resultados favoráveis à utilização da mineração para classificação de eventos com classificação de instâncias corretas de 73,05%. Diante do volume de *e-mails* recebidos, a porcentagem de erros possivelmente pode ser tomada como aceitável, não impactando a tendência geral de categorização a ser observada.

Este trabalho de pesquisa contribuiu para proposição de um modelo de aprendizagem por máquina que pode possivelmente vir a substituir formas comuns de classificação de *e-mails*, que se baseiam em filtros de regras programadas. Esses filtros necessitam de manutenção freqüente, com exclusão, inclusão e refinamento de regras. O novo paradigma propõe que essas regras podem ser aprendidas pelos modelos matemáticos, daí a necessidade de apenas fornecer instâncias de treinamento. De outra forma, não houve exploração de outras formas de representação e semânticas associadas às palavras utilizadas nos textos.

Outra contribuição importante deste trabalho foram os dados experimentais obtidos para escolha do melhor algoritmo classificador. Esses dados permitiram delimitar o tamanho do conjunto de dados, os parâmetros melhores para uso das ferramentas PreTexT e Weka, e estimar a eficiência dos classificadores estudados durante esta pesquisa.

A realidade do conjunto de dados trabalhados nesta pequisa se mostrou muito particular. Como se sabe, *e-mails* possuem tamanhos que diferem de artigos de jornais, relatórios e outros tipos de textos. Seria necessário uma forma mais justa de atribuir valor às palavras contidas nas mensagens tratadas. A proposição de formas de amostragens balanceadas foi um outro problema de difícil solução. Nos algoritmos classificadores, verificou-se dificuldade em rastrear as instâncias classificadas incorretamente, ruídos e mensagens erroneamente rotuladas.

Outro fator limitante foi o tamanho de memória volátil necessário para execução das ferramentas PreTexT e Weka em conjuntos de dados volumosos. Essa limitação não permitiu que fossem testados conjunto de dados de teste com número de instâncias realistas. O Weka tem baixa escalabilidade não sendo capaz de tratar conjunto de dados reais. Por essas razões, a amostragem foi um recurso muito utilizado durante o pré-processamento e classificação supervisionada. As restrições de espaço amostral possivelmente levaram a resultados com problemas de super-ajustamento.

5.2 Trabalhos Futuros

Como trabalhos futuros, podemos citar diversas aplicações e extensões de pesquisa e desenvolvimento em classificação de eventos de segurança da informação, a saber:

- Considerando a existência de uma ferramenta de mineração que viabilize testes com volume de dados mais próximos à realidade, poderíamos realizar experimentações com maior volume de amostras, como por exemplo na fase de treinamento. Considerando que a fase de treinamento é a que envolve maior gama de recursos, mas um bom treinamento reflete em um melhor resultado de classificação.
- Quanto à Gestão de Incidentes no CTIR Gov, seria uma importante melhoria a implantação de um sistema de rastreamento de incidentes *open source* como o RTIR e a integração a uma solução de categorização automática dos eventos recebidos.

- Estudo de formas alternativas de pré-processamento, representação de documentos e valoração de palavras importantes em *e-mails*, como por exemplo os cabeçalhos das mensagens.
- Utilização de técnicas alternativas como *Latent Semantic Indexing* (LSI), *Keyphrase Extraction*, indexação por meio de um dicionário controlado, com avaliação dos resultados no conjunto de dados utilizados nesta pesquisa.
- Soluções para amostragem de documentos que fundamentem o conceito principal de amostragem de textos o qual busca representatividade frente ao espaço amostral, sendo que foi utilizado neste trabalho um conjunto de documentos simplificado como espaço amostral.
- Aplicação de técnicas de classificação supervisionada *online* e incremental capaz de tratar um grande volume de textos.
- Estudo de aplicação dessas técnicas em outras áreas do conhecimento, por exemplo, na gestão de relacionamento ao cliente.
- Aplicação de Web Semântica e Sistemas Multiagentes com o objetivo de melhorar a eficiência dos elementos propostos no modelo de classificação automática de documentos.

Apêndice A

Processos CSIRT

Os processos atuantes em um CSIRT são mostrados na Figura A.8. Segue uma descrição breve sobre cada uma das caixas PC (Preparar, manter e melhorar um CSIRT), PI (Proteger Infraestrutura), D (Detectar), T (Triar) e R (Responder).

A Figura A.1 mostra o processo de preparar que é definido como todo trabalho prévio que permita resposta de ameaças, riscos ou ataques. Nesse caso, faz-se necessário ter pessoas capazes de trabalhar com incidentes, regras, políticas, procedimentos, equipamentos e infraestrutura apropriados. Também na preparação é realizada avaliação da capacidade de tratamento de incidentes, revisão *post mortem* de incidentes, vulnerabilidades e ações de resposta. Melhorias nos processos CSIRT são encaminhadas para subprocessos de planejamento e de projeto.

O processo de proteger está representado na Figura A.2, o qual está relacionado às mudanças na infraestrutura computacional para prevenir ataques ou responder à atividades maliciosas. As melhorias podem ser fundamentadas por meio de análises de artefatos, riscos conhecidos, melhores práticas recomendadas, mitigação de riscos. Nesse processo, são realizadas avaliações da infraestrutura computacional, que servirão para melhorar e diminuir os riscos de segurança dessa infraestrutura. Caso sejam encontradas vulnerabilidades, elas são passadas para processo de detecção. A resposta também interage com o processo de proteger, indicando melhorias no processo por meio de lições aprendidas.

Na detecção, apresentado na Figura A.3, o processo é disparado quando informações são recebidas acerca de uma atividade relatada como suspeita. Eventos que precisam de investigação maior são mandados para triagem, Figura A.4, que são categorizados, correlacionados, priorizados e encaminhados devidamente para uma resposta eficaz.

As Figuras A.5 e A.6 mostram como é o processo de triagem. Esse processo tem por objetivo organizar, ordenar informações de eventos e designá-la para pessoas apropriadas, dentro de prazos, documentando informações de maneira apropriada, lidando com informações dentro de um contexto de segurança [10].

Finalmente o processo de resposta a incidentes é definido pelas atividades de análise de eventos, planejamento de uma estratégia de resposta, coordenação de resposta técnica para contenção do ataque, mitigação de incidentes e ação para recuperar e reparar sistemas afetados. As saídas possíveis desse processo podem ser visualizadas na Figura A.7.

A.1 Preparar, Manter e Melhorar um CSIRT

Esse processo, conforme apresentado na Figura A.1, descreve os requisitos básicos para que a Gestão de Incidentes ocorra de forma efetiva e em tempo hábil. Basicamente são três frentes de atuação que atuam em pessoas, materiais (*software, hardware*) e procedimentos. Nesta seção utilizamos como referência básica o trabalho de [1].

O fator pessoas inclui questões como escolha de equipe técnica, treinamentos e contratações. O fator material inclui questões de ferramentas, equipamentos, infraestrutura, sistemas de comunicações e redes seguras, banco de dados de rastreamento de incidentes, formulários *on-line* de relato de incidentes e ferramentas de análise. A terceira frente é composta pelos procedimentos e normas que irão dirigir as operações de um CSIRT, suas interações com outras partes da organização ou de abrangência e acordo de níveis de serviço.

Uma parte do processo trata da construção inicial de um CSIRT, enquanto os passos envolvem coordenar o planejamento, desenvolvimento e implementação de tratamento de incidentes. Atividades como definição de requisitos e análise de necessidades são essenciais para definir os serviços que o CSIRT prestará. Os requisitos podem vir de diversas fontes sejam elas legais, de negócio, ou até mesmo pessoas envolvidas com o projeto.

A definição de requisitos é utilizada para estabelecer a missão, serviços, modelo organizacional e recursos de um CSIRT. Um processo paralelo obtém patrocínio e custeio para o projeto.

O processo de preparação trata sobre a melhoria contínua de um CSIRT. Uma vez estabelecido o funcionamento de um CSIRT, ou se já opera normalmente, ele pode ser avaliado. Se o processo de resposta necessita de uma avaliação *post mortem*, ela é realizada como um processo de preparação. Os resultados de melhorias identificados vêm dessas avaliações para que sejam feitas modificações e implementações.

A.2 Proteger Infraestrutura

O processo conforme apresentado na figura A.2 relaciona atividades de prevenção de ataques com objetivo de minimizar ou até prevenir completamente os impactos que venham a ocorrer. Nesse caso, relatamos conforme referência básica de [1].

Para que ocorra proteção é necessário que haja mudança na organização. Isso significa alterar elementos computacionais, sejam eles relacionados a aplicativos, redes, sistemas operacionais, entre outros.

Existem diversos exemplos de mudanças que necessitam ser realizadas para proteção da infraestrutura de TI. São alterações em elementos de rede como filtros, *firewalls*, *routers*, servidores de *e-mail*, atualizações em sistemas de detecção a intrusão, mudanças em configurações padrões, instalação de *patches* corretivos para vulnerabilidades, atualizações de anti-vírus para incluir assinaturas novas para novas ameaças.

A prevenção ocorre por meio de inspeções, informações vindas de outras partes do processo CSIRT e aplicação de boas práticas conhecidas. Essas atividades são fronteira entre o tratamento de incidentes e gestão de segurança da informação. É importante que essa fronteira esteja bem definida para que os processos de segurança se integrem às operações de TI.

Os subprocessos são três os quais envolvem atividades com objetivo de avaliar a infraestrutura atual, determinar as modificações que precisam ser executadas e finalmente a implementação das soluções viáveis.

A.3 Detectar Eventos

O processo de detecção geralmente é pensado como atividades restritas à utilização de sistemas de intrusão e monitoração conforme apresentado na Figura A.3. O sentido desse processo possui maior amplitude, pois qualquer observação ou informação de atividade suspeita deve ser considerada pela Gestão de Incidentes de Segurança e deve ser registrada e processada pelas etapas posteriores.

Na detecção, as informações são recolhidas de forma reativa (recebido por fontes internas ou externas por meio de notificações) ou proativa (monitoração de indicadores de possíveis incidentes, serviços de monitoração públicos, descoberta de vulnerabilidades, utilizando ferramentas de monitoração de redes ou sistemas de detecção a intrusão). Uma vez detectado o evento é passado para o subprocesso de triagem.

A.4 Triar Eventos

A triagem é considerada o ponto focal de entrada para informações de qualquer CSIRT. Nesta seção iremos utilizar como fonte básica o [10]. Conforme as figuras A.5 e A.6 esse processo é responsável por organizar os eventos recebidos e encaminhá-los às pessoas adequadas. As atividades relacionadas a esse processo ocorrem dentro de restrições de tempo, tratando informações sigilosas com seu devido cuidado, documentando informações de forma apropriada [10].

A triagem é um elemento essencial para capacidade de qualquer Gestão de Incidentes. Também é considerado como caminho crítico para entendimento daquilo que está sendo reportado pela organização. Com a triagem, é possível obter uma visão geral das atividades que estão ocorrendo na organização e correlação dos dados relatados. Na triagem, é realizada uma avaliação inicial de um relatório recebido e priorizado para tratamento posterior, provendo um local de encontro das documentações recebidas caso não tenham sido realizadas na etapa de detecção [10].

De acordo com [44], o processo de triagem envolve ordenação de eventos, categorização, correlação e priorização, designação de eventos recebidos a membros técnicos, tratamento de relatos de incidentes, confecção de relatórios de vulnerabilidade e tratamento de pedidos de informações gerais. Pode ser comparado com a triagem realizada em hospitais, em que os pacientes necessitam ser organizados por prioridades, separando aqueles que possuem urgência daqueles que podem esperar por uma consulta médica.

A triagem pode ser realizada em dois níveis diferentes: o nível tático baseado na ordenação, categorização e encaminhamento dos eventos e pedidos amparado por critérios pré-determinados; e a triagem estratégica, em que a atenção é dirigida para uma avaliação real da situação e avaliação de impacto nos serviços de negócio. Dependendo do nível utilizado, serão necessárias pessoas com um outro conjunto de conhecimentos e habilidades. No caso em que a triagem estratégica não pode ser realizada, a avaliação é passada para o processo de resposta.

A função de triagem provê um panorama instantâneo do estado atual das atividades conduzidas no CSIRT em que é possível verificar quais incidentes estão abertos ou resolvidos, quais ainda estão pendentes e quantos já foram resolvidos. Esse processo pode auxiliar na identificação de potenciais problemas de segurança e priorizar a carga de trabalho. A informação colhida durante a triagem pode ser utilizada para gerar estatísticas de tendências e vulnerabilidades para alta gestão. Na triagem dependendo da severidade do incidente, é possível que o seu tratamento seja escalado, aumentando sua prioridade, notificando os interessados e as partes envolvidas.

A categorização de eventos pode ser realizada de diversas formas, uma vez que ela pode depender do serviço envolvido, do tipo de relato ou pedido, impacto ou escopo, complexidade do incidente. Uma forma de categorizar seria a utilização de categorias que se distinguem pelo tipo de ataque utilizado. Como exemplo de categorias podemos citar: negação de serviço, vírus, *phishing*, alarme falso, requisição de informação, entre outros. Uma outra estratégia segundo [37] classifica detalhadamente um incidente, mas leva em conta a criticidade e a sensibilidade da informação. Segundo estudo de [20], são mostrados diversos sistemas de categorização com seus lados positivos e negativos. Adicionalmente, o resultado dessa pesquisa apresenta como solução um sistema de taxonomia que classifica informações em relação ao tipo de incidente, ataque utilizado e evento relatado.

A correlação consiste em relacionar os relatos recebidos para determinar o escopo e a severidade da atividade maliciosa. A determinação pode ser realizada baseando-se no número de computadores envolvidos, métodos de ataque empregados, cronologia de ataques, entre outros. Dependendo da situação, diferentes incidentes rastreados podem ser reconhecidos como parte de uma mesma atividade maliciosa.

A priorização é um processo decisório que envolve a missão do CSIRT e suas políticas de atuação. Os critérios de decisão podem ser a proteção de informação sigilosa, o prejuízo financeiro, integridade da infraestrutura, risco de vida, ameaça aos sistemas CSIRT, ameaça a infraestrutura Internet, tipo e escopo da atividade em questão [10].

Designar certo evento pode ser conduzido por critérios de categoria ou prioridade do evento, carga de trabalho atual do CSIRT, equipe técnica disponível para resposta, conhecimento existente de um CSIRT, entre outros. Como exemplos de políticas para encaminhamento de incidentes podemos citar: alternância periódica da pessoa que designa os eventos; distribuição igualitária entre equipe técnica; distribuição de acordo com perfil técnico; combinação das anteriores.

Além da distribuição de trabalho, é importante existir um agente responsável pela condução do processo de um incidente. As estratégias são essencialmente duas em que existe apenas uma pessoa para tratamento do incidente ou uma transferência de controle para um líder fixo por determinado tempo.

Para que a triagem ocorra de forma eficiente, é importante que existam atividades de mapeamento de ativos, análise de riscos e escalas de trabalho. Alguns exemplos de metodologias de análise de riscos são: OCTAVE, CRAMM, COBIT, ISO 13335, ISO 17799, ISO 21827, ISO 15408, entre outros.

Erros comuns na triagem são constituídos por três tipos: (i) pela falta de entendimento seja por envio de relato incompleto, impreciso ou não coerente; (ii) pela atribuição incorreta de categorias em que não se sabe se certos incidentes são novos ou já foram resolvidos e (iii) erro humano, pelas faltas cometidas nas informações passadas, conclusões mal interpretadas ou suposição de hipóteses incorretas [10].

As Figuras A.5 e A.6 mostram as tarefas envolvidas no subprocesso de Triagem de Eventos. Na Figura A.5, as informações sobre eventos são recebidas do processo de detecção (recebimento, monitoração, análise de indicadores), e aqueles eventos que necessitam de maiores ações são encaminhados ao processo de triagem. A triagem na tarefa T1 rotula, utilizando categorias pré-definidas, aos eventos, por exemplo, vírus, alarme falso, *phishing*, negação de serviços, entre outros, e correlaciona-os, relacionando outros eventos àqueles que estão sendo analisados [10].

Ainda na Figura A.6, os eventos que realmente interessam à equipe sejam por motivos técnicos, gerenciais, ou de missão do CSIRT são priorizados em T2. Logo em seguida, os eventos são designados, atribuindo trabalhos aos membros da equipe para resposta técnica ou gerencial. Após essa etapa, cabe aos membros da equipe o processo seguinte, que consiste na resposta de incidentes. Caso não se enquadre nesses quesitos, são arquivados ou encaminhados para fora do processo.

A.5 Responder

O processo de resposta conforme apresentado na Figura A.7 inclui etapas com objetivo de abordar, resolver ou mitigar um evento ou incidente. Existem três tipos de atividades de resposta. São as respostas técnicas, gerenciais e legais. Elas podem ocorrer ao mesmo tempo, mas, para maior efetividade, devem ocorrer de forma coordenada com membros de todas as áreas de resposta; cada membro coordenando o planejamento e execução de suas atividades de resposta. Informações entre áreas necessitam ser compartilhadas entre esses subprocessos [1].

A resposta técnica trata das ações tomadas pela equipe técnica para analisar e resolver um evento ou incidente. A equipe técnica pode incluir membros de um CSIRT, até pessoas internas ou externas à organização. Segundo [1], as ações de resposta podem incluir:

- análise de informações, dados e materiais coletados do evento ou incidente;
- pesquisa de estratégias de mitigação e opções de recuperação;
- redigir informes, alertas ou outras publicações que forneçam conselho ou orientação para resolução ou mitigação do evento ou incidente;
- realizar mudanças técnicas na infraestrutura como desconectar sistemas afetados na rede, mudar configurações de segurança, filtrar portas, serviços e endereços de internet, entre outros;
- erradicar ou apagar arquivos e processos maliciosos;
- restaurar ou recuperar sistemas afetados.

A resposta gerencial aborda atividades que requerem algum tipo de supervisão ou intervenção gerencial, comunicação, interação ou aprovação por qualquer tipo de resposta que seja tomada. Podem incluir outras áreas da organização como gestão de pessoas, relações públicas, área financeira, auditoria e aderência, entre outras.

Essa resposta gerencial pode incluir uma consulta com uma equipe jurídica sobre obrigações legais acerca de um determinado sistema de redes computacionais utilizado

para atacar uma entidade externa. Outro exemplo é a demissão de um funcionário por atividades ilícitas que contrariam a política de segurança da informação da organização.

A resposta legal se trata de ações associadas às atividades de investigação, entrada de ação na justiça, direitos, questões de *copyright* e privacidade, interpretação legal, leis e regimentos. A resposta legal somente pode ser iniciada pela resposta gerencial. Geralmente esse tipo de resposta leva mais tempo e recursos que as outras respostas, levando meses ou até anos para terem seu trânsito julgado.

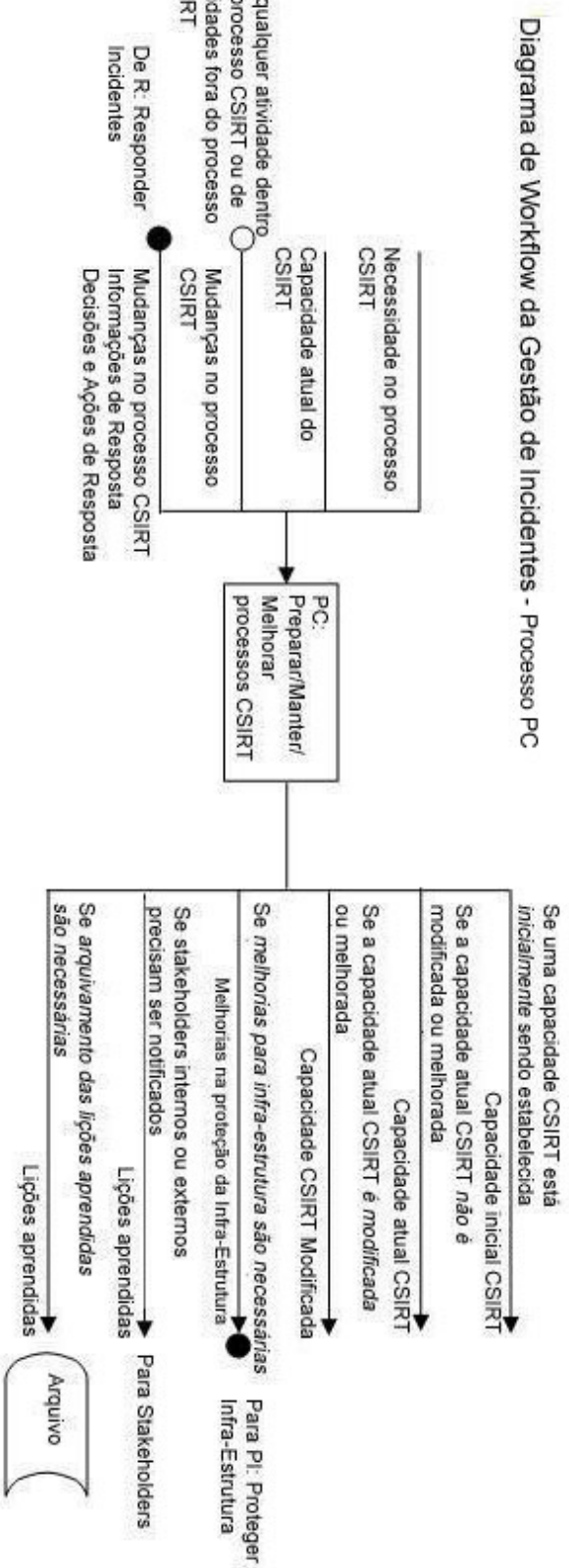

Figura A.1: *Workflow* do processo PC (Preparar, Manter e Melhorar um CSIRT) (adaptado de 1).

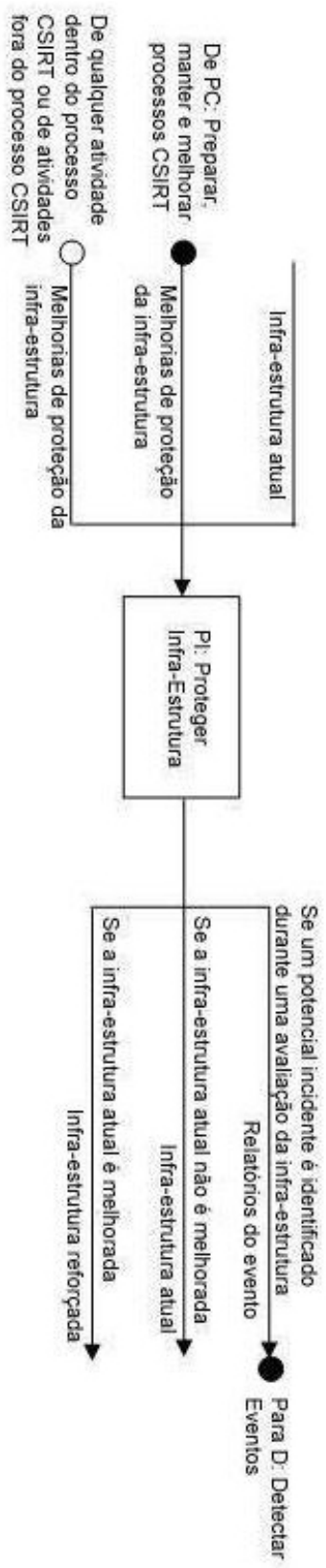

Figura A.2: *Workflow* do processo PI (Proteger infraestrutura) (adaptado de 1).

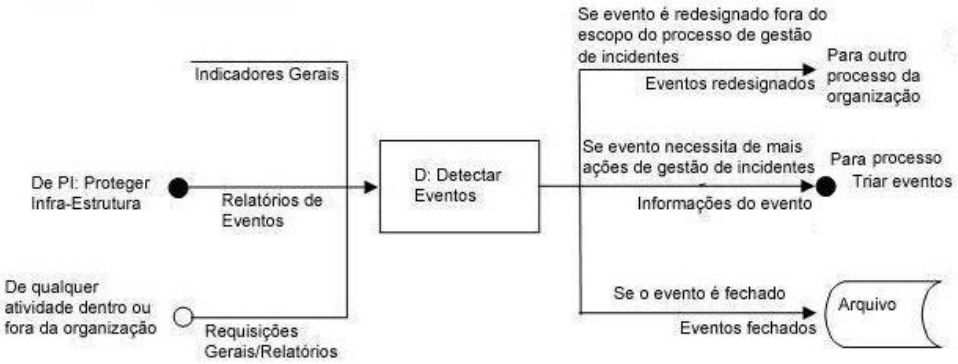

Figura A.3: *Workflow* do processo D (Detectar Eventos) (adaptado de 1).

Figura A.4: *Workflow* do processo T (Triagem) (adaptado de 1).

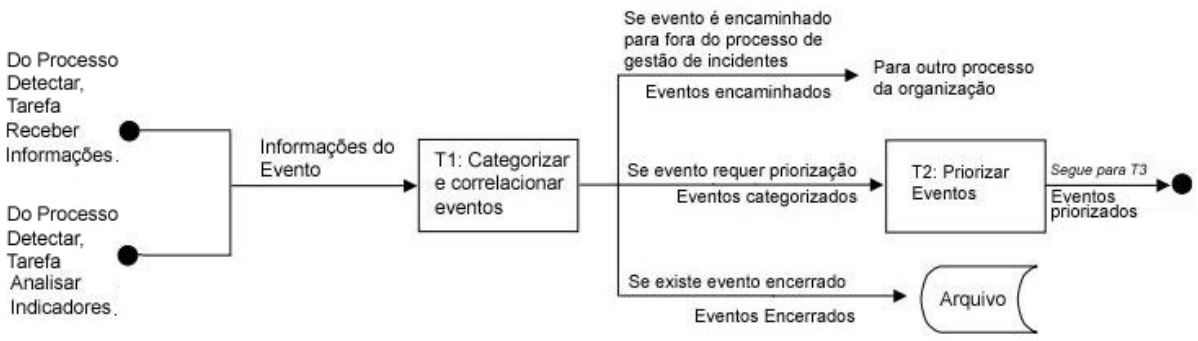

Figura A.5: Processo de triagem de eventos (adaptado de 1).

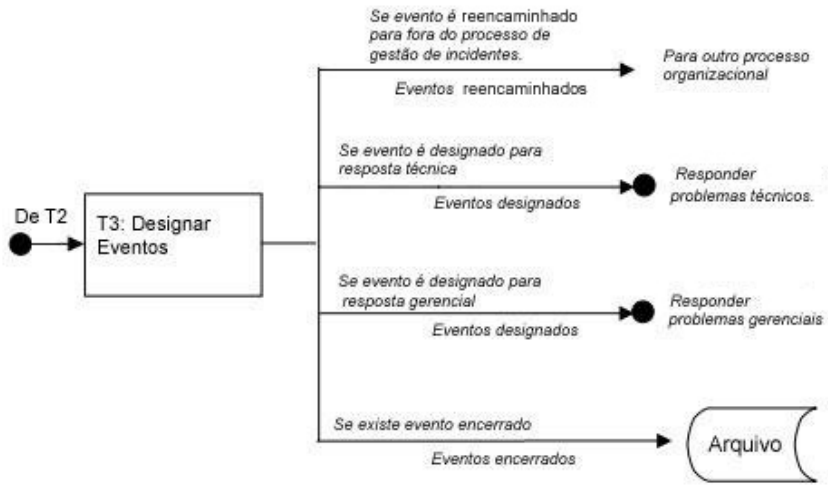

Figura A.6: Continuação do Processo de Triagem de Eventos (adaptado de 1).

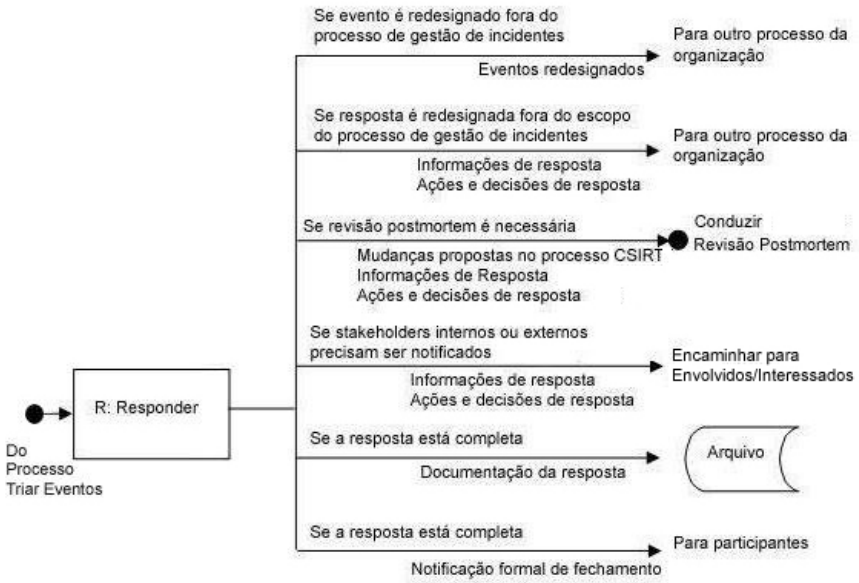

Figura A.7: *Workflow* do processo R (Responder) (adaptado de 1).

Figura A.8: Visão geral dos processos CSIRT (adaptado de 1).

Referências

[1] Chris Alberts, Audrey Dorofee, Georgia Killcrece, Robin Ruefle, and Mark Zajicek. Defining incident management processes for CSIRTs: A work in progress. Technical report, CMU/SEI-2004-TR-015. Carnegie Mellon University, 2004. Disponível em: `http://www.sei.cmu.edu/publications/documents/04.reports/04tr015.html`. Acessado em: Setembro de 2008. x, 2, 7, 8, 9, 10, 11, 57, 60, 62, 63, 64, 65, 66

[2] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras, and Constantine D. Spyropoulos. An evaluation of naive bayesian anti-spam filtering. In *Proc. of the workshop on Machine Learning in the New Information Age*, pages 9–17, 2000. 1

[3] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, and Constantine D. Spyropoulos. An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–167. ACM New York, NY, USA, 2000. 1

[4] Christian N. Aranha. *Abordagem de Pré-Processamento Automático para Mineração de Textos em Português: Sob o Enfoque da Inteligência Computacional*. PhD thesis, PUC-RJ, 2007. Disponível em: `http://www.lambda.maxwell.ele.puc-rio.br/acessoConteudo.php?nrseqoco=32304`. Acessado em Julho de 2009. 1

[5] Ron Bekkerman, Andrew McCallum, and Gary Huang. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. *Center for Intelligent Information Retrieval, Technical Report IR*, 418, 2004. 1

[6] Igor A. Braga. Uma ferramenta para filtragem adaptativa de spam, 2007. Monografia de Bacharelado em Ciência da Computação. Departamento de Ciência da Computação. Universidade de Brasília. Disponível em: `http://monografias.cic.unb.br/dspace/handle/123456789/122`. Acessado em Novembro de 2009. 1

[7] Andrej Bratko, Bogdan Filipič, Gordon V. Cormack, Thomas R. Lynam, and Blaz Zupan. Spam filtering using statistical data compression models. *The Journal of Machine Learning Research*, 7:2698, 2006. 1

[8] Xavier Carreras, Llués Márquez, and Jordi G. Salgado. Boosting trees for anti-spam email filtering. In *Proceedings of RANLP-01, 4th international conference on recent advances in natural language processing*, pages 58–64, 2001. 1

[9] Rommel Carvalho, Kathryn Laskey, Paulo Costa, Marcelo Ladeira, Laecio Santos, and Shou Matsumoto. *UnBBayes: Modeling Uncertainty for Plausible Reasoning in the Semantic Web*, volume Semantic Web, pages 1–28. INTECH, 2010. Disponível em `http://sciyo.com/books/show/title/semantic-web`. Acessado em Abril de 2010. 30

[10] CERT. Fundamentals of incident handling. Material do Curso de Treinamento pelo CERT.br, 2008. Disponível em `http://www.sei.cmu.edu/products/courses/cert/fundamentals-incident.html`. Acessado em Dezembro de 2008. 11, 56, 58, 59, 60

[11] CERT.br. Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil. Estatísticas dos incidentes reportados ao CERT.br, 2009. Disponível em: `http://www.cert.br/stats/incidentes/`. Acessado em: Março de 2009. 1

[12] CERT/CC. Carnegie Mellon University's Computer Emergency Response Team Coordination Center. CERT statistics (historical), 2009. Disponível em: `http://www.cert.org/stats/`. Acessado em: Março de 2009. 1

[13] Roman Danyliw, Jan Meijer, and Yuri Demchenko. The incident object description exchange format, 2007. Disponível em: `http://www.ietf.org/rfc/rfc5070.txt`. Acessado em: Janeiro de 2009. 11

[14] Ronen Feldman and James Sanger. *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, New York, NY, USA, 2006. 14, 15, 16, 22

[15] Giorgio Fumera, Ignazio Pillai, and Fabio Roli. Spam filtering based on the analysis of text information embedded into images. *The Journal of Machine Learning Research*, 7:2720, 2006. 1

[16] José M. Gómez Hidalgo, Manuel M. López, and Enrique P. Sanz. Combining text and heuristics for cost-sensitive spam filtering. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning*, pages 99–102, Morristown, NJ, USA, 2000. Association for Computational Linguistics. 1

[17] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009. 2, 29

[18] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques (The Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, San Francisco, CA, USA, 2006. 12

[19] David J. Hand, Padhraic Smyth, and Heikki Mannila. *Principles of data mining*. MIT Press, Cambridge, MA, USA, 2001. 12

[20] John Howard and Thomas Longstaff. A common language for computer security incidents. Technical report, SAND98-8667. Sandia National Labs, 1998. Disponível em: `http://www.cert.org/research/taxonomy_988667.pdf`. Acessado em Setembro de 2008. 37, 59

[21] Georgia Killcrece, Klaus-Peter Kossakowski, Robin Ruefle, and Mark Zajicek. Organizational models for computer security incident response teams (CSIRTs). Technical report, CMU/SEI-2003-HB-001. Carnegie Mellon University, 2003. Disponível em: `http://www.sei.cmu.edu/library/abstracts/reports/03hb001.cfm`. Acessado em: Setembro de 2008. 7

[22] Georgia Killcrece, Klaus-Peter Kossakowski, Robin Ruefle, and Mark Zajicek. State of the practice of computer security incident response teams (CSIRTs). Technical report, CMU/SEI-2003-TR-001. Carnegie Mellon University, 2003. Disponível em: `http://www.sei.cmu.edu/library/abstracts/reports/03tr001.cfm`. Acessado em: Setembro de 2008. 7, 8, 11

[23] Brian Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. *Lecture notes in computer science*, 3201:217–226, 2004. 1

[24] Irena Koprinska, Josiah Poon, James Clark, and Jason Chan. Learning to classify e-mail. *Inf. Sci.*, 177(10):2167–2187, 2007. 33

[25] Wenke Lee and Salvatore J. Stolfo. Data mining approaches for intrusion detection. In *SSYM'98: Proceedings of the 7th conference on USENIX Security Symposium*, pages 6–6, Berkeley, CA, USA, 1998. USENIX Association. 13

[26] Giuseppe Manco, Elio Masciari, and Andrea Tagarelli. A framework for adaptive mail classification. In *ICTAI '02: Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence*, page 387, Washington, DC, USA, 2002. IEEE Computer Society. 1

[27] Kevin Mandia, Matt Pepe, and Chris Prosise. *Incident Response and Computer Fosensics*. McGraw-Hill Osborne Media, 2nd edition, 2003. 7

[28] Zdravko Markov and Ingrid Russel. An introduction to the weka data mining system, 2008. Disponível em: `http://www.cs.ccsu.edu/~markov/weka-tutorial.pdf`. Acessado em: Setembro de 2008. 29

[29] Claudia A. Martins. Uma abordagem para pré-processamento de dados textuais em algoritmos de aprendizado. Master's thesis, USP, 2003. Disponível em: `http://www.teses.usp.br/teses/disponiveis/55/55134/tde-08032004-164855/`. Acessado em: Julho de 2009. 1

[30] Edson T. Matsubara. O algoritmo de aprendizado semi-supervisionado co-training e sua aplicação na rotulação de documentos. Master's thesis, Universidade de São Paulo, 2004. Disponível em: `http://www.teses.usp.br/teses/disponiveis/55/55134/tde-19082004-092311/`. Acessado em: Setembro de 2009. 1, 4

[31] Edson T. Matsubara, Claudia A. Martins, and Maria C. Monard. Pretext: Uma ferramenta para pré-processamento de textos utilizando a abordagem bag-of-words. Technical report, Universidade de São Paulo, 2003. Disponível em: http://www.icmc.usp.br/~edsontm/pretext/pretext.html. Acessado em: Junho de 2009. x, xi, 2, 4, 16, 17, 18, 30, 31

[32] Rodrigo Ormonde. Classificação automática de páginas web multi-label via MDL e support vector machines. Master's thesis, Universidade de Brasília, 2009. Disponível em: http://monografias.cic.unb.br/dspace/handle/123456789/212. Acessado em: Novembro de 2009. 1, 19

[33] PCWorld. Timeline: A 40-year history of hacking, 2008. Disponível em: http://archives.cnn.com/2001/TECH/internet/11/19/hack.history.idg/. Acessado em: Outubro de 2009. 6

[34] Roberto M. Pimenta. Um modelo de gestão de qualidade para tratamento de incidentes de redes aplicado na APF, 2008. Monografia de Conclusão do Curso de Especialização em Gestão de Segurança da Informação e Comunicações. Departamento de Ciência da Computação. Universidade de Brasília. Disponível em: http://dsic.planalto.gov.br/cegsic/83-monografias-da-1o-turma-do-cegsic. Acessado em Novembro de 2008. 1, 2

[35] John C. Platt. Fast training of support vector machines using sequential minimal optimization. pages 185–208, 1999. x, 27, 28

[36] PreTexT. Pretext: An environment for pre-processing text for text mining, 2003. Disponível em: http://www.icmc.usp.br/~edsontm/pretext/pretext.html. Acessado em: Junho de 2009. 2

[37] Gavin Reid, Dustin Schieber, and Ivo Peixinho. CSIRT case classification an example for enterprise CSIRT, 2004. Disponível em: http://www.first.org/resources/guides/csirt_case_classification.html. Acessado em Setembro de 2008. 59

[38] RTIR. Best practical's RTIR: Request tracker for incident response, 2008. Disponível em: http://bestpractical.com/rtir. Acessado em: Julho de 2008. 4, 12

[39] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002. 1, 32

[40] Marcelino Silva. Mineração de dados - conceitos, aplicações e experimentos com weka, 2004. Disponível em: https://www.sbc.org.br/bibliotecadigital/download.php?paper=35. Acessado em: Setembro de 2008. 29, 30

[41] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005. x, xi, 13, 15, 16, 18, 19, 20, 21, 22, 23, 24, 26, 28, 29

[42] Terena. Incident object description and exchange format working group, 2009. Disponível em: http://www.terena.org/activities/tf-csirt/iodef/. Acessado em: Dezembro de 2009. 11

[43] Jesse Vincent, Dave Rolsky, Darren Chamberlain, Richard Foley, and Robert Spier. *RT Essentials*. O'Reilly Media, Inc., 2005. 12

[44] Moira J. West-Brown, Don Stikvoort, Klaus-Peter Kossakowski, Georgia Killcrece, Robin Ruefle, and Mark Zajicek. Handbook for computer security incident response teams (CSIRTs). Technical report, CMU/SEI-2003-HB-002. Carnegie Mellon University, 2003. Disponível em: `http://www.sei.cmu.edu/library/abstracts/reports/03hb002.cfm`. Acessado em: Setembro de 2008. 6, 7, 8, 9, 10, 11, 58

[45] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, second edition, June 2005. 13, 19, 20, 22, 29, 30

[46] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49, New York, NY, USA, 1999. ACM. 1, 2, 33

[47] Xiaojin Zhu, Andrew Goldberg, Ronald Brachman, and Thomas Dietterich. *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers, 2009. 1