UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
ÁREA DE CIÊNCIA DA COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE ELETRICIDADE

Um Framework para Reconhecimento de Opinião Utilizando Sistema de Informação Geográfica (SIG): Um Estudo de Caso na Geração de Mapas

Gilberto Nunes Neto

São Luís
19 de Agosto de 2016

Gilberto Nunes Neto

Um Framework para Reconhecimento de Opinião Utilizando Sistema de Informação Geográfica (SIG): Um Estudo de Caso na Geração de Mapas

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Engenharia de Eletricidade, da Universidade Federal do Maranhão, como requisito para o título de Mestre em Engenharia Elétrica na área de concentração Ciência da Computação.

Orientador: Dr. Denivaldo Lopes

Co-orientador: PhD. Zair Abdelouahab

São Luís
19 de Agosto de 2016

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Núcleo Integrado de Bibliotecas/UFMA

Nunes Neto, Gilberto.
Um Framework para Reconhecimento de Opinião Utilizando Sistema de Informação Geográfica SIG : Um Estudo de Caso na Geração de Mapas / Gilberto Nunes Neto. - 2016.
91 f.

Coorientador(a): Zair Abdelouahab.
Orientador(a): Denivaldo Lopes.
Dissertação (Mestrado) - Programa de Pós-graduação em Engenharia de Eletricidade/ccet, Universidade Federal do Maranhão, São Luís, 2016.

1. Análise de Sentimento. 2. Máquina de Aprendizado. 3. Mineração de Opinião. 4. Sistema de Informação Geográfica. 5. Twitter. I. Abdelouahab, Zair. II. Lopes, Denivaldo. III. Título.

Gilberto Nunes Neto

Um Framework para Reconhecimento de Opinião Utilizando Sistema de Informação Geográfica (SIG): Um Estudo de Caso na Geração de Mapas

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por Gilberto Nunes Neto e aprovada pela comissão examinadora.

Aprovada em ____ de _____ de 2016.

BANCA EXAMINADORA

_____

Denivaldo Cicero Pavão Lopes (Orientador)
Doutor em Informática - UFMA

_____

Karla Donato Fook
Doutora em Computação Aplicada - IFMA

_____

Daniela Barreiro Claro
Doutora em Ciência da Computação - UFBA

_____

Francisco José da Silva e Silva
Doutor em Ciência da Computação - UFMA

Este trabalho é dedicado a todos que acreditam que pesquisar não é apenas alargar a fronteira do conhecimento, mas sim um instrumento de melhoria social.

‘‘Mata o tempo e matas a tua carreira.’’

Bryan Forbes

Agradecimentos

Inicio agradecendo a Deus, já que ele colocou pessoas tão importantes em minha vida, sem as quais não teria dado conta da árdua jornada do mestrado.

Aos meus pais, João Ribeiro Nunes Neto e Maria Lady de Araújo Nunes; e aos meus irmãos, Amanda Larissa de Araújo Nunes e Alécio de Araújo Nunes, por todo o carinho, atenção, amor, confiança, ensino e inspiração em toda a minha vida.

Ao meu primo, Rômulo Araújo, pela atenção e presteza, segurança e apoio durante toda a minha estadia em São Luís. Esse agradecimento estende-se aos demais primos.

Um muito obrigado cheio de amor à Eliamara Soares, uma pessoa muito especial na minha vida, por toda sua humildade, bondade e companheirismo para comigo.

Aos meus amigos, Elka Barros, José Raimundo, Otílio Paulo e Thiago Pinheiro, pela ajuda prestada na execução da minha pesquisa e rotina diárias.

Ao meu grande amigo Glaucyo Lefevre, que apesar da distância sempre se fez presente e prestativo como sempre.

Aos casais que amo, Vladimir Oliveira e Anita Myrtes, João Paulo e Maria da Cruz, por toda ajuda e incentivo, fundamentais para o meu ingresso ao Programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão. Obrigado por tudo!

Ao meu querido amigo e orientador, o professor Dr. Denivaldo Lopes, pela confiança, estima, inspiração, amizade e ajuda no momento mais difícil do mestrado. Obrigado por sempre estar disponível e disposto a ajudar-me, querendo que eu aproveitasse cada segundo de pesquisa para absorver algum tipo de conhecimento, fosse ele profissional ou pessoal.

Aos professores de graduação e pós-graduação, Anchieta Araújo, Anselmo Paiva, Atevaldo Lopes, Carlos Giovanni, Denivaldo Lopes, João Viana, Marcus Vinícius Lemos, Marcus Vinícius Ribeiro, Rônaldy Sousa, Wesley Emmanuel Lima e Zair Abdelouahab, pelos conselhos, trocas de experiências e ensinamentos adquiridos não apenas dentro de sala de aula.

Aos meus amigos que se mantiveram mais próximos durante o mestrado, Diego Carvalho, Ismael Leal, Júlio César, Júlio Portela, Otílio Paulo, Whesley Dantas, pelos esclarecimentos às minhas dúvidas e pelo apoio nos momentos de dificuldades.

Por fim, à minha equipe do Laboratório de Engenharia de Software e Rede de Computadores (LESERC), Amanda Lima, Débora Stefanello, Eder Silveira, Larissa Madeira, Osvaldo Júnior, Matheus Ribeiro, Wesley Lima, por toda ajuda em prol da realização desta pesquisa.

Aos meus amigos do Instituto Federal de Educação, Ciência e Tecnologia do Piauí (IFPI) -

Campus Picos, Francisco Diassis, Jorge Roberto, Josivaldo Barros, Juciê Xavier, Lourenilson Sousa, Messias Medeiros e Osvaldo Augusto, que se mantiveram sempre próximos, mesmo com a distância.

Resumo

Com a globalização da Internet, o número de usuários utilizando os meios de comunicação social é cada vez maior. A Rede Social Twitter é um bom exemplo disso. Frequentemente, o Twitter é utilizado para postar comentários sobre os mais variados tipos de assuntos, como: artistas, produtos, saúde pública, dentre outros. A propagação da informação nesses meios de comunicação é muito relevante, pois pode atingir pessoas de todas as classes sociais, a qualquer hora e lugar do mundo. O Twitter, além de apresentar tamanha abrangência, permite a postagem de comentários georreferenciados, ou seja, possibilita a localização de onde as postagens foram feitas. Diversos estudos propõem a utilização das postagens obtidas a partir do Twitter, para avaliar o quão esses meios de comunicação refletem o mundo real. Nesse sentido, o presente trabalho propõe um Framework genérico que, além de avaliar conceitos relacionados à mineração de opiniões, descreve a realização de estudos de caso, os quais analisam fontes de opiniões textuais e propõe minerar opiniões em nível de aspecto, utilizando como fontes de opinião comentários do Twitter. Um protótipo estende e implementa o Framework proposto para viabilizar o processo de mineração de opinião em redes sociais. Os resultados obtidos mostram a viabilidade da utilização desse Framework para suporte à tomada de decisão por parte de seus usuários.

Palavras-chave: *Análise de Sentimento, Máquina de Aprendizado, Mineração de Opinião, Sistema de Informação Geográfica, Twitter.*

Abstract

With the globalization of the Internet, the number of users using the means of social communication it is each time bigger. The social network Twitter is a good example. Twitter is often used to post comments on all kinds of subjects, such as artists, products, public health, among others. The spread of information in these media is very important because it can reach people from social class, anytime and anywhere in the world. Twitter supports geo-referenced comments. This feature allows georeferenced tweets. One can use the comments obtained from Twitter to evaluate how the reality of social network reflects the real world. In this sense, the present work proposes a generic Framework that besides evaluating concepts related to the opinion mining, describes the accomplishment of case studies, which analyze sources of textual opinions and proposes to mine opinions at the level of aspect, using as sources of opinion Twitter comments. A prototype extends and implements the proposed Framework to enable the process of opinion mining in social networks. The results show the feasibility of using this Framework to support decision making by its users.

Keywords: *Analysis Sentiment, Opinion Mining, Machine Learning, Geographic Information System, Twitter.*

Lista de Figuras

Lista de Tabelas

Sumário

Lista de Siglas

LIRAa	Levantamento Rápido de Índices para Aedes aegypti
OMS	Organização Mundial de Saúde
OPAS	Organização PanAmericana da Saúde
PNCD	Programa Nacional de Controle da Dengue
IB	Índice de Breteau
IA	Índice Amostral
IC	Índice Casual
RG	Reconhecimento Geográfico
PE	Pontos Estratégicos
KDD	Knowledge Discovery in Databases
SGBD	Sistema Gerenciador de Banco de Dados
SIG	Sistema de Informação Geográfica
API	Application Programming Interface
TF	Term Frequency
IDF	Inverse Term Document Frequency
PCA	Principle Component Analysis
SVM	Support Vector Machine
ROF	Recognising Opinion Framework
IBGE	Instituto Brasileiro de Geografia e Estatística

Capítulo 1

Introdução

Neste capítulo, são apresentados o contexto do trabalho, a problemática, a solução proposta, os objetivos da pesquisa, a metodologia empregada na pesquisa, motivação, contribuições científicas e uma visão geral dos demais capítulos.

1.1 Contexto

O crescimento do número de usuários nas redes sociais e, consequentemente, do número de informações gerenciadas por esses meios de comunicação tornou o conteúdo disponível na *Web* mais dinâmico e massivo, despertando um nicho de negócio interessante para instituições públicas e privadas, de modo geral.

Esse cenário passou a ser muito propício para essas instituições buscarem informações sobre seus produtos e serviços, ou mesmo sobre os mais diversificados assuntos, visando obter vantagens em seus negócios, seja por se tratar de uma ferramenta gratuita ou pela sua abrangência mundial. Outro fator preponderante é a capacidade de propagação da informação nesses ambientes, pois atingem todas as classes sociais, a qualquer hora e por todo o planeta.

Por causa do grande volume de conteúdo subjetivo presente em fontes textuais na *Web*, o processo de extração e de interpretação desse tipo de informação apresenta certo grau de complexidade. Assim, a *Web* veio a ser minerada por meio de mecanismos que empregam técnicas de descoberta de conhecimento em bases de dados e processamento de linguagem natural, em uma área denominada Mineração de Opinião (**Ganeshbhai e Shah (2015)**; **Liu (2012)**; **Tsytsarau e Palpanas (2012)**).

A Mineração de Opinião consiste em extrair, processar e classificar opiniões contidas em

mídias virtuais. Essa mineração possibilita classificar opiniões sobre aspectos e características de entidades através de classes, como: positivas, negativas ou neutras (**Pang e Lee (2008)**; **Lin et al. (2014)**; **Liu (2012)**). Exemplos de aspectos de uma entidade "cidade" seriam a população, informações geográficas (localização) e renda per capita por habitante. Essas informações, quando classificadas e sumarizadas são extremamente úteis e podem influenciar tomadas de decisão por parte de gestores públicos, por exemplo.

Vale ressaltar que as publicações em redes sociais são uma fonte textual de opinião bem mais complexa e difícil de minerar, pois os textos podem conter opiniões sobre várias entidades ou nenhuma opinião.

As redes sociais mais populares como *Twitter* e *Facebook* possibilitam postagens de conteúdo georreferenciadas, ou seja, o conteúdo das postagens está associado às suas respectivas informações geográficas (latitude e longitude, por exemplo).

Informações geográficas em ambiente computacional são manipuladas por Sistemas de Informação Geográfica (**SIG**). Para **Burrough (2006)**, os sistemas de informação geográfica são compreendidos como um conjunto integrado de hardware e software capaz de desempenhar funções diversas, tais como: captura, organização, manipulação, análise, modelagem e visualização de dados espacialmente referenciados. Esse tipo de sistema é importante na solução de problemas complexos de planejamento e gestão que requeiram informações geográficas, conforme aponta **Burrough (2006)**.

Segundo **Culotta (2010)**, existem diversos motivos para se aplicar a mineração de opinião em mensagens de redes sociais para prever um evento da vida real. Primeiramente, as mensagens completas fornecem informações mais descritivas do evento explorado. Além disso, os perfis dos usuários contêm informações como localização, idade e sexo, o que possibilita um estudo estatístico mais detalhado, permitindo que seja realizada uma análise demográfica, por exemplo.

De acordo com o autor supracitado, a grande maioria dos artigos que abordamo o tema utiliza o *Twitter* para coletar as mensagens publicadas. A presente pesquisa levou em consideração as definições apresentadas no trabalho de **Culotta (2010)** para sua realização.

1.2 Problemática

As opiniões utilizadas nesse trabalho foram coletas a partir de comentários do *Twitter*, o presente trabalho pode ser aplicado a opiniões oriundas de outras redes sociais, a exemplo

do *Linkedin* e *Myspace*. Os *tweets*, como são chamados os comentários do *Twitter*, podem ser polarizados em opiniões positivas, negativas ou neutras, como visto anteriomente. As opiniões polarizadas correspondem às informações e os *tweets* representam os dados a serem extraídos e transformados em informações.

Levando em consideração a complexidade do processo de extração e transformação dos dados, o trabalho proposto deve responder a seguinte pergunta:

- Como transformar dados, obtidos a partir de redes sociais, em informação útil para a detecção de opiniões que auxiliem nas tomadas de decisão com base em sistemas de informação geográfica?

1.3 Objetivos

Nesta seção, são apresentados os objetivos geral e específicos deste trabalho, que culmina com a proposta de um *Framework* genérico. Esse *Framework* é baseado em Mineração de Opinião, Redes Sociais e Sistema de Informação Geográfica para apoiar a tomada de decisão.

1.3.1 Objetivo Geral

O objetivo geral do trabalho proposto é desenvolver um *Framework* genérico para extrair e analisar de forma automatizada dados oriundos de redes sociais em específico o *Twitter*, possibilitando a geração de conhecimento para suporte à tomada de decisão.

1.3.2 Objetivos Específicos

Para a concretização do objetivo geral, objetivos específicos devem ser alcançados:

- Estudar, implementar e/ou reutilizar algoritmos voltados à mineração de opinião, sua aplicabilidade para o reconhecimento de padrões e a busca do conhecimento em conteúdos de redes sociais;
- Utilizar conceitos e ferramentas (*Cases*, por exemplo) da Engenharia de *Software* que visem aperfeiçoar o processo de extração e processamento de opinião em redes sociais;
- Representar as opiniões usando um modelo de classificação;

- Propor Estudos de Caso visando relacionar registros do *Twitter* com registros do mundo real.

1.4 Metodologia da Pesquisa

A opção metodológica da pesquisa consiste no método hipotético-dedutivo e a harmonização entre as documentações nas técnicas de pesquisa.

Tal escolha deve-se ao fato do presente estudo explorar a combinação entre a Mineração de Opinião, Redes Sociais e Sistema de Informação Geográfica para apoiar a tomada de decisão em um domínio genérico. Por sua vez, a problemática de pesquisa concentra-se na complexidade de mensurar o quanto as informações obtidas por meio das Redes Sociais refletem situações do mundo real.

A metodologia aplicada é descrita como segue:

1. **Pesquisa bibliográfica para coletar informações sobre o estado da arte dos itens abaixo**:
    - Redes Sociais: *Twitter*, no caso desta pesquisa;
    - Mineração de Opinião;
    - Sistema de Informação Geográfica.
2. **Tipo da pesquisa**: Exploratória.
3. **Universo da pesquisa**: Dados oriundos de Redes Sociais.
4. **Amostragem**: Dados extraídos do *Twitter*.
5. **Instrumentos de coleta de dados**:
    - As coletas foram realizadas por meio de um computador presente no Laboratório de Engenharia de Software e Rede de Computadores. Esse será melhor detalhado no capítulo 4, na subseção 4.3.2.
6. **Apresentação dos Resultados**:
    - **Estudo de Caso**: Propor um estudo de caso visando relacionar registros minerados junto ao *Twitter* com registros do mundo real, bem

como tabulação dos resultados para o estudo de caso proposto, mediante a plotagem de mapas.

1.5 Contribuições Científicas

Neste trabalho, podemos destacar as seguintes contribuições científicas:

- A criação de um *Framework* genérico para Mineração de Opinião em Redes Sociais, baseado em Sistemas de Informação Geográfica (**SIG**);
- Criação de uma estratégia automatizada para aquisição de informação, adaptada para redes sociais que suportem postagens georreferenciadas, utilizando **SIG**;
- Aplicação do protótipo baseado no *Framework* proposto para classificar opiniões sobre domínios distintos.

1.6 Estrutura do Trabalho

Esta dissertação está organizada em capítulos, descritos como segue:

O primeiro capítulo trata do contexto tecnológico no qual todo o trabalho de pesquisa se insere. O capítulo também expõe a problemática e a motivação da pesquisa, buscando o desenvolvimento de uma solução viável para mineração de opinião em redes sociais. Neste primeiro capítulo, são expostos o objetivo geral, os objetivos específicos e a metodologia de pesquisa.

O segundo capítulo apresenta a fundamentação teórica que norteia os assuntos básicos envolvidos nesta pesquisa. Os conceitos necessários para o entendimento da solução proposta neste trabalho são expostos, tais como: *Framework* e a sua composição; Protótipo e sua implementação; Mineração de Opinião com auxílio de Máquina de Aprendizado. Além disso, conceitos básicos sobre Sistema de Informação Geográfica são abordados a fim de permitir a compreensão de como o *Framework* proposto é aplicado na pesquisa. Finalizando o capítulo, o domínio de aplicação da solução proposta para extração e análise de dados em redes sociais é apresentado.

O terceiro capítulo faz uma explanação sobre o estado da arte da pesquisa, bem como a apresentação de trabalhos relacionados que têm como foco a mineração de opinião e/ou análise

de sentimentos, com bases de dados distintas.

O quarto capítulo aborda o *Framework* proposto para suportar a extração e análise de dados em redes sociais, assim como a metodologia adotada pelo *Framework* e o Protótipo. Também é exposto o modo como o Protótipo viabilizou o processo de mineração de opinião proposto neste trabalho.

O quinto capítulo apresenta dois estudos de caso. O primeiro utiliza o *Framework* proposto para detecção e localização de casos relacionados a dengue e chikungunya, favorecendo o suporte à tomada de decisão em saúde pública. O segundo aplica o *Framework* proposto ao processo de *impeachment* da Presidente do Brasil, visando detectar e verificar a dispersão das opiniões sobre o *impeachment*.

Capítulo 2

Fundamentação Teórica

Nesta seção, é exposto um enfoque geral da base teórica que fundamenta este trabalho, a fim de permitir a percepção de como os conceitos se articulam na construção da solução do problema proposto.

2.1 Descoberta do Conhecimento em Bases de Dados (KDD)

O presente trabalho usa métodos para mineração de dados. Segundo **Fayyad et al. (1996)**, mineração de dados é uma das etapas do processo de descoberta de conhecimento em bases de dados (*Knowledge Discovery in Databases* - **KDD**).

O conceito de **KDD** costuma gerar certa confusão com a definição de Recuperação de Informação (**RI**). Em **Abilio et al. (2015)**, os autores definem **RI** como uma consulta que atende a um anseio momentâneo do usuário, por exemplo uma consulta realizada no *Google*. Já o **KDD** trata de explicitar informação que se encontra implícita em bases de dados, por meio da extração (aquisição) e análise dos dados. O processo de **KDD** é formado por um conjunto de etapas ilustradas na Figura 2.1 e descritas na subseção 2.1.1.

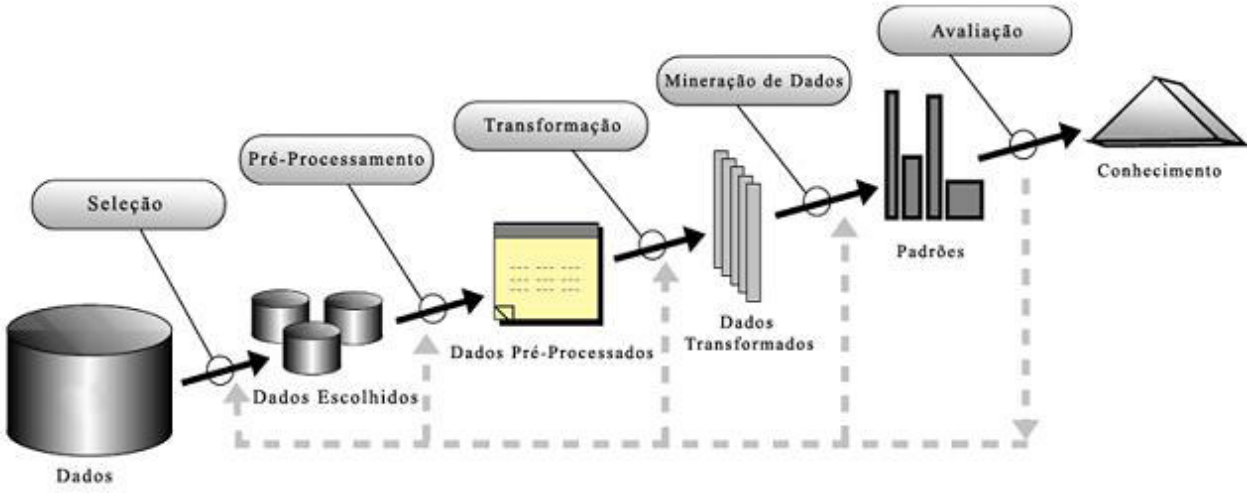

Figura 2.1: Fases do processo de KDD. Adaptado de **Fayyad et al. (1996)**.

2.1.1 Fases do KDD

A primeira fase é a de **Seleção** ou escolha da massa de dados a ser minerada. Vale lembrar que essa amostra deve conter os indivíduos com maior representatividade em relação aos demais.

A segunda fase, o **Pré-Processamento**, tem por objetivo assegurar a qualidade e a confiabilidade dos dados envolvidos no **KDD**. São realizadas operações básicas, como a remoção de ruídos, que podem ser, por exemplo, atributos nulos, conforme descrito por **Fayyad et al. (1996)**.

A terceira fase consiste na **Transformação** dos dados preparados pela fase anterior (Pré-Processamento), determinando quais serão os atributos realmente interessantes. Para tal é utilizado padrão pré-estabelecido para a aplicação de algoritmos de mineração para a fase posterior.

As etapas de Pré-Processamento e Transformação podem levar até 80% do tempo necessário para todo o processo, conforme **Fayyad et al. (1996)**.

Após a realização das fases anteriores, é dado início à quarta fase **Mineração de Dados** (*Data Mining*). Vale ressaltar que essa fase é a mais importante do **KDD**, sendo realizada através da escolha do método e do algoritmo mais compatível com o objetivo da extração, a fim de reconhecer padrões nos dados analisados que sirvam de subsídios para descobrir conhecimentos ocultos ou implícitos.

A **Avaliação** ou Interpretação é a quinta e última fase do processo de **KDD**. Nela são identificados, dentre os padrões extraídos na etapa de Mineração de Dados, os padrões inte-

ressantes que serão apresentados como resultados. Medições estatísticas ou métodos de teste de hipóteses podem ser aplicados com o objetivo de eliminar resultados não legítimos.

2.1.2 Mineração de Dados

O trabalho de **Fayyad et al. (1996)** apresenta a seguinte definição para a etapa de Mineração de Dados:

> *Processo de pesquisa em grandes volumes de dados para extração de conhecimento, utilizando técnicas de Inteligência Computacional para buscar relações de semelhança ou discordância entre dados, com o intuito de reconhecer padrões, irregularidades e regras, com o objetivo de transformar dados, aparentemente ocultos, em informações relevantes para a tomada de decisão e/ou avaliação de resultados.*

Para **Kantardzic (2002)**, a mineração de dados é subdividida em dois grupos de técnicas de acordo com os tipos de dados e objetivos, sendo elas técnicas descritivas (associação e clusterização, por exemplo) e de previsão (classificação e regressão, por exemplo), respectivamente. As descritivas são geralmente explanatórias e frequentemente necessitam de mecanismos de pós-processamento para avaliar os resultados obtidos. Já as tarefas preditivas são baseadas na construção de um modelo para determinada variável alvo como uma função das variáveis explicativas ou discriminatórias de uma entidade.

O presente trabalho faz uso de modelos de classificação para determinar se uma sentença é positiva, negativa ou neutra (caso necessário). Para isso, são utilizados os métodos de classificação descritos na subseção 2.1.2.2.

2.1.2.1 Validação Cruzada

Após a geração do modelo preditivo, o mesmo deve ser validado através de uma técnica de validação, como a Validação Cruzada (*Cross Validation*), por exemplo. A Validação Cruzada é uma técnica estatística de validação de algoritmos de classificação utilizada para avaliar os resultados de um algoritmo e como o mesmo se comporta ao receber conjuntos de dados independentes. A utilização mais comum dessa técnica, se dá quando é necessário avaliar o quanto um modelo preditivo irá funcionar na prática, como apresentado por **Refaeilzadeh et al. (2009)**.

Uma das maneiras de utilizar a Validação Cruzada é através do *k-fold Cross Validation*, onde k é o número de subconjuntos de dados e número de iterações que serão necessárias para a execução da técnica. A Figura 2.2 exemplifica a técnica de Validação Cruzada (usando 3 subconjuntos de dados, no caso, k=3).

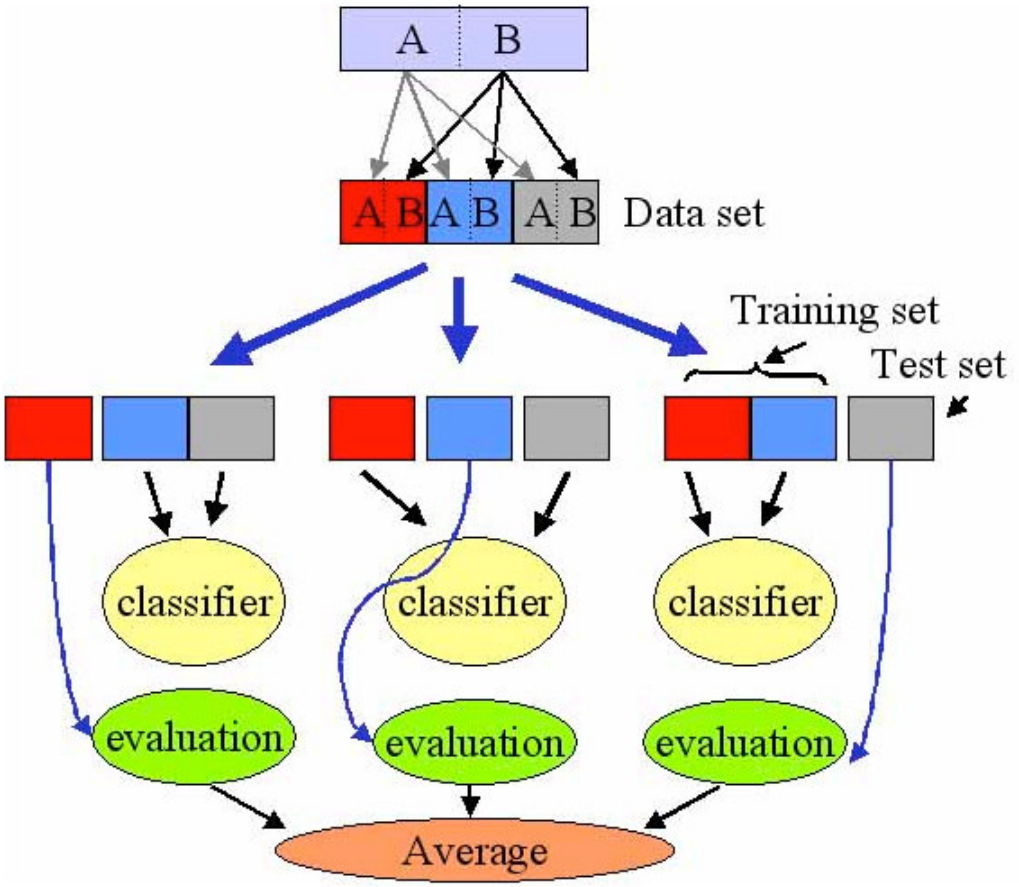

Figura 2.2: Exemplo do *k-fold Cross Validation* (para k=3). Fonte **Projects (2016)**.

A técnica consiste em dividir de forma aleatória o conjunto de dados que se deseja utilizar, para validação do algoritmo em k subconjuntos de dados igualmente distribuídos. Após a divisão, iniciam-se os testes, submetendo um dos subconjuntos ao algoritmo classificação (conjunto de teste) e os demais são utilizados como conjunto de treinamento. Repete-se o processo até que todos os subconjuntos tenham sido utilizados como conjunto de teste e como conjunto de treinamento. O resultado desse processo é uma matriz, na qual é apontada a média de acertos e erros de cada classe, conforme descrito por **Refaeilzadeh et al. (2009)**. Neste trabalho, foi utilizado o $k = 10$, ou seja, *10-fold Cross Validation*.

2.1.2.2 Métodos de classificação

A classificação é a tarefa de qualificar objetos, tais como: a identificação de e-mails spams, categorização de células em malignas ou benignas e outros, conforme apresentado por **Fu et al. (2010)**. Um método de classificação recebe como entrada um conjunto de registros (instâncias). Cada registro é um par (x, y), em que x é o conjunto de atributos explicativos e y o atributo especial, denominado rótulo da classe, atributo alvo ou de categorização. O conjunto de atributos x pode possuir valores discretos ou contínuos, enquanto que y deve ser um atributo discreto. Os modelos de classificação são usados para predizer o rótulo da classe de registros cujo atributo y não seja conhecido. O modelo de classificação pode ser entendido como um processo de atribuição automática de rótulo a uma classe, quando essa recebe o conjunto de atributos de um registro não conhecido.

Um método de classificação é um processo sistemático para a criação de modelos de classificação com base em um conjunto de dados de entrada. São exemplos de métodos de classificação: árvores de decisão — Floresta Aleatória (**Breiman (2001)**), redes neurais, classificadores — Máquina de Vetores Suporte (**Vapnik (1982)** e **Mitchell (1997)**). Apesar da existência de trabalhos cujo intuito é a Mineração de Opinião, ainda não há um consenso se a aplicação de método de classificação é realmente eficaz na detecção automática de opiniões em grandes volumes de dados. Nos trabalhos de **Silva et al. (2012)**, **Wu et al. (2014)** e **Azure (2016)**, foram avaliados diversos métodos de classificação bem conhecidos e a partir dessas pesquisas foram selecionados os algoritmos de classificação utilizados no trabalho proposto, sendo eles: Floresta Aleatória, Máquina de Vetores Suporte e Bayesiano. Esses métodos usam um algoritmo de aprendizagem para identificar o modelo mais apropriado para o relacionamento entre o conjunto de atributos e o rótulo da classe dos dados de entrada. O esquema para a construção de um modelo de classificação é apresentado na Figura 2.3.

Um conjunto de registros cujos rótulos sejam conhecidos são fornecidos como entrada. Esse conjunto também é conhecido como Conjunto de Treinamento (*Training Set*) e é utilizado para construir um modelo de classificação. O modelo criado é então aplicado em um Conjunto de Teste (*Test Set*), que possui registros com rótulos de classes desconhecidos, conforme descrito por **Tan et al. (2006)**.

A avaliação de um classificador pode ser realizada por meio de métricas de desempenho, derivadas da Matriz de Confusão, tais como: Acurácia, Precisão, Revocação e Medida-F, de acordo com o trabalho de **Chawla (2005)**.

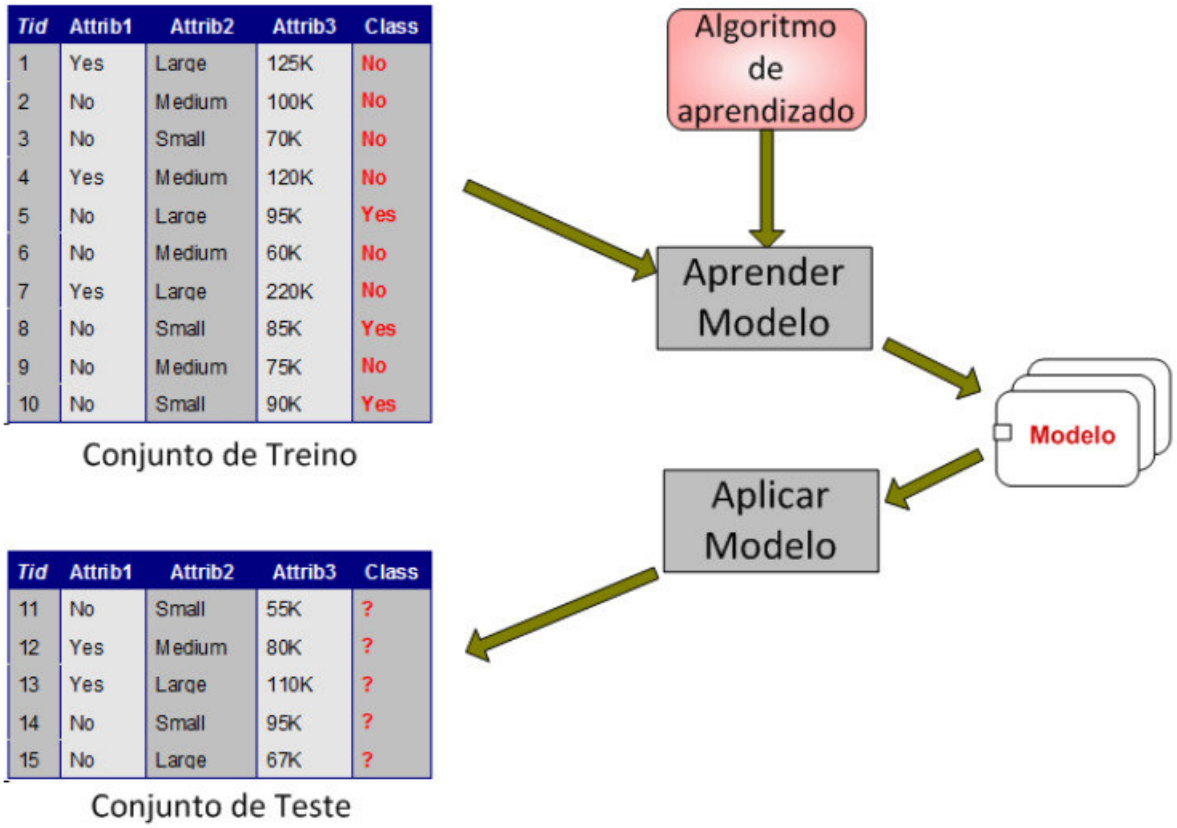

Figura 2.3: Esquema utilizado para o desenvolvimento de modelos de classificação. Adaptado de **Tan et al. (2006)**.

Essas métricas foram utilizadas no presente trabalho com o objetivo de avaliar os resultados obtidos com os algoritmos de classificação (Floresta Aleatória (**Breiman (2001)**), Máquina de Vetores Suporte (**Vapnik (1982)**) e Bayesianos (**Mitchell (1997)**)) presentes no protótipo do *Framework* proposto. Vale ressaltar que as implementações desses algoritmos foram reutilizadas por meio do módulo de análise presente no *Framework* proposto.

2.1.3 Mineração de Opinião

A expressão Mineração de Opinião foi empregada oficialmente em 2003, sendo definida como uma ferramenta que viabiliza o processo de agregação de opiniões sobre atributos de um determinado tema, questão ou algo similar a partir do processamento dos resultados de uma pesquisa sobre o mesmo, conforme descreve **Dave et al. (2003)** em seu trabalho.

A Mineração de Opinião é uma área de estudos que abrange pesquisas de mineração de dados, linguística computacional, extração de informação e inteligência artificial (**Lin et al. (2014)**). Ela pode ser definida como um estudo realizado computacionalmente para processar opiniões, sentimentos, emoções e subjetividade expressos em forma textual, sendo dividida em

três etapas (**Ganeshbhai e Shah (2015)**; **Liu (2012)**):

- **Etapa 1**: Identificar os documentos textuais com textos subjetivos sobre um assunto ou uma entidade;
- **Etapa 2**: Classificar as opiniões em diferentes classes, tais como: positivas, negativas ou neutras;
- **Etapa 3**: Sumarizar os resultados obtidos, a partir de modelos de visualização. Os resultados sumarizados podem ser a entrada para outras aplicações.

Uma opinião é definida formalmente por Liu (**Liu (2012)**) como uma quíntupla (e_i, a_{ij}, s_{ijkl}, h_k, t_l), onde:

- e_i: é o nome de uma entidade;
- a_{ij}: é o aspecto da entidade e_i. Um aspecto também é denominado tópico;
- s_{ijkl}: é a opinião sobre aspecto a_{ij} da entidade e_i;
- h_k: é a entidade que expressa a opinão, também chamado de fonte de opinião;
- t_l: é o tempo no qual a opinião foi expressa por h_k.

A opinião s_{ijkl}, explicitada sobre uma entidade ou aspecto, é medida em termos de uma polaridade, podendo ser qualificada em classes, tais como: positiva, negativa ou neutra (**Liu (2012)**).

Em fontes textuais como as postagens do *Twitter*, a identificação das entidades é realizada a partir de menções nos próprios *tweets* (nomenclatura usada para os comentários do *Twitter*). Geralmente, locuções substantivas ou substantivos são considerados aspectos ou tópicos, durante o processo de identificação dos mesmos, conforme descrito por **Liu (2012)**.

Este trabalho utiliza a representação formal da opinião para estruturá-la, além de contribuir para as três etapas da Mineração de Opinião. Levando em consideração a etapa de identificação, são utilizados métodos para identificar entidades com base em seus aspectos, conforme apresentado na subseção 2.1.3.1. Quanto à etapa de classificação, são empregados os conceitos apresentados na subseção 2.1.2.2. Por último, a etapa de sumarização, que gera os resultados que servem como entrada para outra aplicação, por meio de modelos de visualização, mais especificamente por meio de mapas — conforme será explicitado na subseção 2.2 — e gráficos.

2.1.3.1 Análise textual

A análise textual praticada pela Mineração de Opinião pode ser aplicada em três níveis distintos, sendo que a granularidade necessária à mineração a ser realizada determina a escolha do nível. Segundo **Kolkur et al. (2015)**, os níveis são:

- **Documento**: Nesse nível, a tarefa é classificar se um documento é, como um todo, positivo ou negativo. Esse tipo de análise é utiliza mediante um documento que menciona apenas uma entidade;
- **Sentença**: Esse é o nível de análise mais refinada para um documento. Determina o sentimento de uma sentença em um documento. Esse tipo de análise é útil quando um documento apresenta opiniões sobre mais de uma entidade, além de permitir identificar e distinguir sentenças objetivas (fatos) de sentenças subjetivas (opiniões);
- **Características**: Nesse nível, o foco é a opinião expressa. O objetivo é a extração de características, com a finalidade de identificar a entidade ou algum de seus aspectos no documento.

O presente trabalho adota a Mineração de Opinião em nível de características, a qual é dividida nas mesmas etapas descritas na subseção 2.1.3. A tarefa de extração de características (*features*) identifica todos os atributos de uma entidade que estão descritos em um documento ou sentença por meio de seus aspectos. Na sentença "O conforto e autonomia deste carro são excelentes, mas o sistema multimídia precisa de melhorias", são extraídos os aspectos "conforto", "autonomia" e "sistema multimídia".

Na etapa seguinte, a classificação é realizada utilizando técnicas estatísticas baseadas em aprendizagem de máquina para identificar a polarização de opinião dentro da sentença analisada, mais especificamente **TF-IDF** (abreviação do inglês *Term Frequency–Inverse Document Frequency*, que significa Frequência do Termo–Inverso da Frequência nos Documentos) (**Ramos (1999)**; **Robertson (2004)**) e **PCA** (abreviação do inglês *Principle Component Analysis*, que significa Análise dos Componentes Principais) (**Scholkopf et al. (1999)**; **Pipanmaekaporn e Li (2012)**). É importante frisar, que o módulo de análise foi o responsável por prover as implementações das técnicas estatísticas utilizadas neste trabalho.

Por último, a etapa de sumarização apresenta a opinião geral sobre uma entidade e seus aspectos, por meio de gráficos e mapas.

2.2 Sistemas de Informação Geográfica

De acordo com **Nuhcan Akçit (2014)**, **SIG** pode ser entendido como um sistema de informação computacional como qualquer outro. Todavia, uma característica relevante desse tipo de sistema é a tecnologia de banco de dados usada por ele, segundo **Nuhcan Akçit (2014)**. Em **SIG** todas as informações armazenadas em seu banco de dados devem estar ligadas a uma referência geográfica com latitude/longitude, como pode ser visto em **Lobo et al. (2015)**.

O termo Sistema de Informação Geográfica (**SIG**) é empregado para sistemas que possibilitam a manipulação de dados geográficos por meio do computador (**Casanova A. et al. (2005)**). De acordo com **Nuhcan Akçit (2014)**, **SIG** pode ser entendido como um sistema de informação computacional como qualquer outro. Esses sistemas ainda devem ser capazes de recuperar informações não apenas com base em suas características alfanuméricas, mas por meio de sua localização espacial.

Nesse tipo de aplicação, o usuário final contempla uma visão em que todas as informações disponíveis sobre o objeto de interesse estão inter-relacionadas com base na localização geográfica. Como pode ser visto em **Lobo et al. (2015)**, em **SIG** todas as informações armazenadas em seu banco de dados devem estar ligadas a uma referência geográfica com latitude/longitude.

Segundo **Carr (2003)**, a definição de **SIG** é:

> "*Um conjunto manual ou computacional de procedimentos utilizados para armazenar e manipular dados georreferenciados, possibilitando a obtenção de informação espacial*".

Um ponto bastante relevante em **SIGs** é a capacidade de armazenar a geometria dos objetos geográficos e de seus atributos, bem como suas caraterísticas (**Casanova A. et al. (2005)**). Para cada objeto geográfico, o **SIG** necessita armazenar seus atributos e as várias representações gráficas associadas. Por permitir uma boa variedade de aplicações que contemplam áreas como agricultura, floresta, cartografia, cadastro urbano e redes de concessionárias (água, energia e telefonia), têm-se no mínimo três grandes maneiras de utilização de um **SIG**, conforme descrito por **Casanova A. et al. (2005)**:

- Como ferramenta para confecção de mapas;
- Como ferramenta de suporte para análise espacial de fenômenos;
- Como um banco de dados geográficos, com funções de armazenamento e recuperação de informação espacial.

2.2.1 Estrutura Geral de um Sistema de Informação Geográfica

Segundo Huang (**Huang e Xu (2011)**), os Sistemas de Informação Geográfica devem contemplar três componentes básicos de informação:

- **Espaço**: É a componente que descreve o espaço a ser representado, normalmente referenciada a um sistema de coordenadas. Esta representação poder ser realizada por meio de três elementos básicos, sendo eles: pontos, linhas, polígonos;
- **Atributos**: São os objetos espaciais que apresentam um conjunto de propriedades associadas à natureza do espaço;
- **Metadados**: Responsável por descrever a componente espacial e não-espacial do espaço a ser representado.

Levando em consideração uma visão mais abrangente, pode-se estruturar um **SIG** com os seguintes componentes (**Casanova A. et al. (2005)**):

- Interface com usuário;
- Entrada e integração de dados;
- Funções de consulta e análise espacial;
- Visualização e plotagem;
- Armazenamento e recuperação de dados (organizados sob a forma de um banco de dados geográficos).

Os componentes estão hierarquicamente relacionados. Na primeira camada, a mais próxima ao usuário, temos a interface de interação que define como o sistema é operado e controlado. Na camada intermediária, um sistema desse tipo deve ter mecanismos de processamento de dados espaciais (entrada, edição, análise, visualização e saída). Na última camada da estrutura,

um sistema de gerência de bancos de dados geográficos oferece armazenamento e recuperação dos dados espaciais e seus atributos.

A Figura 2.4 apresenta o relacionamento hierárquico dos principais componentes de um **SIG**. Vale lembrar que cada camada funciona como um subsistema e desempenha função de seus objetivos e necessidades específicos, no entanto, primando por trabalhar de maneira integrada.

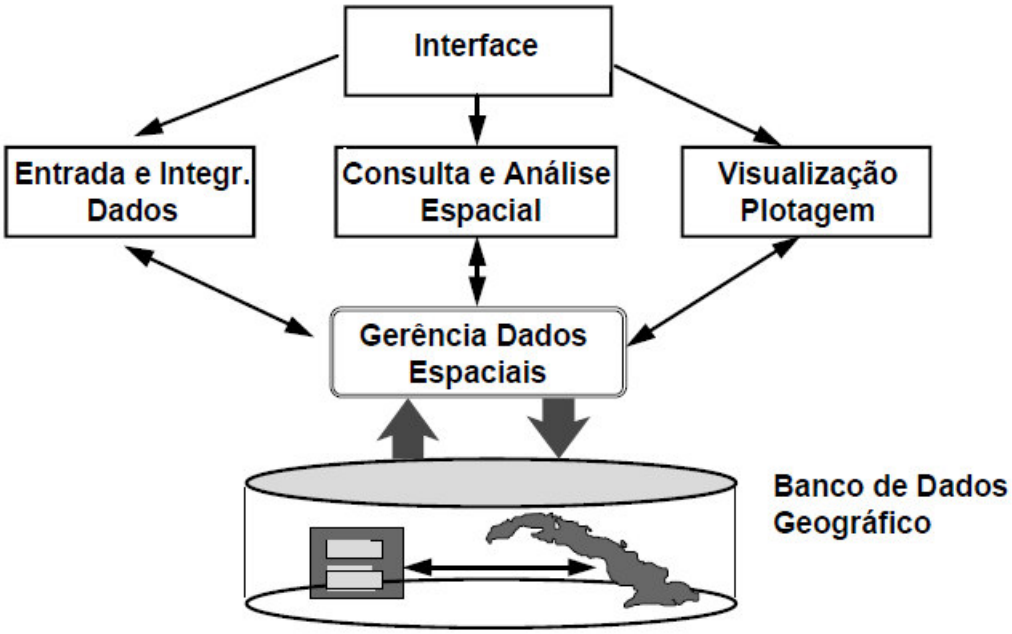

Figura 2.4: Estrutura Geral de Sistemas de Informação Geográfica. Fonte: adaptado de **Casanova A. et al. (2005)**.

2.2.2 Evolução dos Sistemas de Informação Geográfica

Atualmente, há uma variedade enorme de oferta de **SIG**, que são divididas basicamente em três gerações conforme define e ilustra **Casanova A. et al. (2005)** (vide Figura 2.5):

- **Primeira Geração (1983-1990)**: Surgiu no início da década de 80, denominada de geração de "CAD Cartográfico", e caracteriza-se por sistemas herdeiros da tradição de Cartografia, com suporte de banco de dados limitado e cujo paradigma típico de trabalho é o mapa. Tendo o *Arc/View* como exemplo de *software* dessa geração;

- **Segunda Geração (1990-1997)**: Chegou ao mercado no início da década de 90, conhecida como geração dos "Bancos de Dados Geográficos". Caracteriza-se por

ser concebida para uso em ambientes cliente-servidor, acoplado a gerenciadores de bancos de dados relacionais e com pacotes adicionais para processamento de imagens. O *AutoCAD MAP* é um exemplo de *software* pertencente a essa geração;

- **Terceira Geração (1997-Hoje)**: No final dos anos 90, surge a terceira geração, denominada "Bibliotecas Geográficas Digitais" ou "Centros de Dados Geográficos", e caracterizada pelo gerenciamento de gigantescas bases de dados geográficos, com acesso por meio da rede mundial de computadores (*Internet*), proporcionando uma interoperabilidade cada vez maior entre os sistemas existentes. Sendo o *GeoServer*[1] um *software* presente nessa geração.

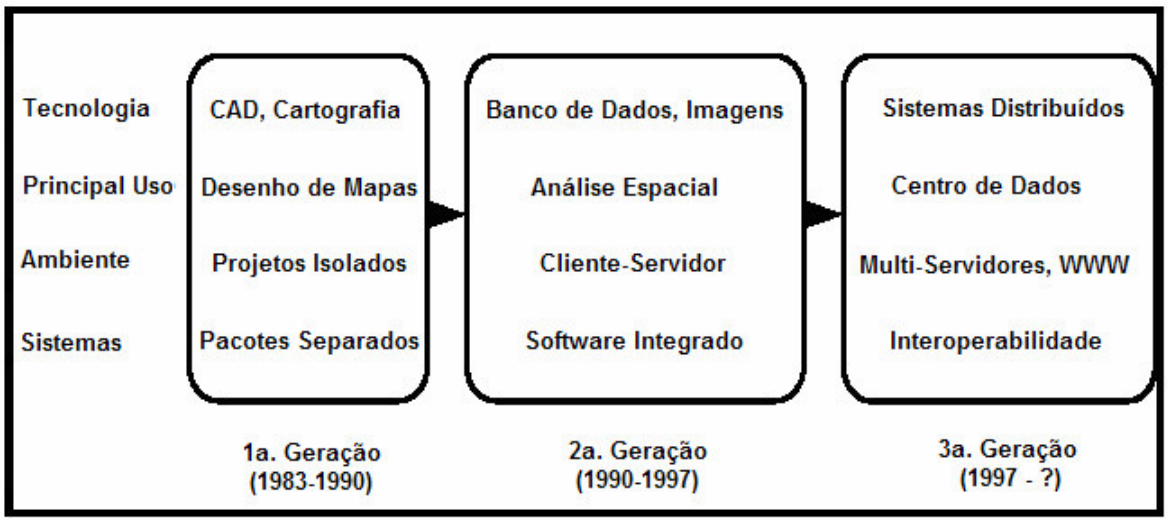

Figura 2.5: Histórico evolutivo dos Sistemas de Informação Geográfica. Fonte: adaptado de **Casanova A. et al. (2005)**.

Este trabalho adota os conceitos de Sistemas de Informação Geográfica por meio do *GeoServer*[1], o qual é responsável pela geração dos mapas baseado na mineração e o georreferenciamento das opiniões.

2.3 Síntese

Este capítulo, apresenta uma visão geral dos principais conceitos e tecnologias envolvidas no desenvolvimento deste trabalho de pesquisa. O objetivo é expor conceitos importantes para compreensão da proposta de *Framework* para suportar a Mineração de Opinião em redes sociais com base em Sistemas de Informação Geográfica.

[1]**Documentação oficial** - http://docs.geoserver.org/.

O capítulo fornece conceitos sobre Descoberta do Conhecimento em Bases de Dados, contextualizando suas aplicações, bem como a Mineração de Dados e de Opinião. Também aborda características, conceitos e tecnologias ligadas a Sistemas de Informação Geográfica, o qual torna possível a manipulação de dados georreferenciados, um dos fatores primordiais deste trabalho. Além disso, são apresentados conceitos básicos sobre métodos de classificação.

Capítulo 3

Estado da Arte

Este capítulo visa descrever as abordagens existentes na literatura para a recuperação de dados e análise de opinião dos mesmos, mediante a mineração de opinião, incluindo pesquisas e ferramentas para sua execução. A recuperação e análise de opinião desses dados constituem o foco desta pesquisa, assim, foram selecionados trabalhos que tratam dessa temática. Enumeram-se a seguir alguns desses trabalhos na linha de pesquisa, bem como uma breve explanação sobre esses trabalhos.

3.1 Trabalhos relacionados

Nesta seção, alguns trabalhos relacionados ao tema mineração de opinião e/ou análise de sentimento são apresentados. Além disso, uma avaliação sobre algumas abordagens de mineração de opinião e/ou análise de sentimentos propostas na literatura é apresentada.

3.1.1 TOM: Twitter Opinion Mining Framework Using Hybrid Classification Scheme (Khan et al. (2014))

Khan et al. (2014) propõem um *Framework* baseado no *Twitter*. Segundo os autores, a mineração de opinião é aplicada aos *tweets* com o intuito de classificá-los em positivos, negativos e neutros, conseguindo identificar atitudes e opiniões que são expressas em qualquer forma ou linguagem. Para os autores, o *Twitter* tornou-se uma das plataformas de *microblog* mais populares recentemente. Segundo eles, milhões de usuários podem compartilhar seus pensamentos e opiniões sobre diferentes aspectos e eventos na plataforma do *microblog*. Portanto, os autores consideram o *Twitter* uma fonte rica de informações para tomada de decisão e análise

de sentimento. Assim, o trabalho de **Khan et al. (2014)** busca oferecer um *Framework* para a mineração de opinião sobre o *microblog* em questão de maneira automatizada, provendo às organizações uma maneira rápida e eficaz para monitorar as opiniões do público em direção a sua marca, negócio, diretores, etc. Os autores destacam a ampla gama de recursos e métodos para mineração de opinião em conjuntos de dados oriundos do *Twitter*, que tem sido foco de várias pesquisas nos últimos anos, com os mais variados resultados. No trabalho proposto por **Khan et al. (2014)**, um dos focos da pesquisa é o problema de classificação incorreta dos *tweets* como neutros. Os pesquisadores apontam um elevado percentual de classificações errôneas. Os pesquisadores apresentam um *Framework* para mineração de opinião em *tweets* que reduz tal problema, sendo esse baseado em uma classificação híbrida, ou seja, composta por uma junção de técnicas. O *Framework* proposto inclui várias etapas de pré-processamento antes de submeter o texto ao classificador. Durante o processo de classificação, foram utilizadas técnicas de aprendizagem de máquina.

3.1.2 SMACk: An Argumentation Framework for Opinion Mining (Dragoni et al. (2016))

No trabalho de **Dragoni et al. (2016)**, os autores apresentam um *Framework* para a extração de opiniões polarizadas, onde essas são utilizadas como argumentação, por meio da teoria de argumentação apresentada e defendida pelos autores. Eles afirmam que a teoria de argumentação, bem como a mineração de opinião, estão se expandindo rapidamente devido sua utilização em várias aplicações. Para os pesquisadores, o processo de extração das opiniões relevantes e debatidas em mídias sociais *on-line* e *sites* comerciais é uma tarefa emergente no campo de pesquisa que trata das técnicas de mineração de opinião. Sua crescente relevância é devida ao impacto da exploração dessas técnicas em diferentes domínios de aplicação para analisar a publicidade pessoal, empresarial ou governamental nessas mídias, por exemplo. Neste trabalho, os autores apresentam um aplicativo para a sumarização de opiniões construído sobre o *Framework* proposto por eles e baseado na teoria da argumentação. Essa teoria é estruturada através da Inteligência Artificial. O objetivo do *Framework* é trocar, comunicar e resolver pontos de vista possivelmente conflitantes em cenários distribuídos, por meio da mineração de opinião e teoria da argumentação. Os pesquisadores mostram como o *Framework* é capaz de extrair opiniões relevantes e debatidas de um conjunto de documentos oriundos de *sites* comerciais *on-line*, sendo esses documentos gerados por usuários desses *sites*. A estrutura do

Framework pode ser aplicada a vários contextos, com diferentes níveis de complexidade. Os principais exemplos ocorrem no domínio das ciências sociais, onde uma enorme quantidade de texto precisa ser analisada para detectar o humor das pessoas com relação a diferentes tópicos debatidos, ou a análise de conteúdo *on-line* gerado por usuários de produtos ou serviços. Os autores realizaram um estudo de caso para demonstrar a aplicabilidade do *Framework*. Nesse estudo de caso, foi analisado um conjunto de revisões de produtos pertencentes a uma das categorias utilizadas no *site* da *Amazon*.

3.1.3 A Framework for Opinion Mining in Blogs for Agriculture (Valsamidis et al. (2013))

Para **Valsamidis et al. (2013)**, nos últimos anos, há muitas notícias sobre *blogs* e a maneira como eles influenciam a mídia e mudam a forma como as pessoas se comunicam e compartilham o conhecimento. Para os pesquisadores, os *blogs* também recebem uma atenção especial das empresas comerciais, motivando um grande número de pesquisas no meio acadêmico sobre eles, além de representarem uma importante fonte de conteúdo para a descoberta do conhecimento aplicada ao setor agrícola. Para os autores, tal aplicabilidade é resultado da crescente utilização de *blogs* por parte dos agricultores por razões profissionais. Nesse trabalho, os pesquisadores definem Mineração de Opinião como a tarefa de avaliar a atitude do autor com relação a um determinado assunto, atitude que pode ser uma opinião positiva ou negativa. Nessa pesquisa, os autores descrevem os desafios e oportunidades dos *blogs* para a agricultura no tocante a análise das informações contidas neles. Assim, **Valsamidis et al. (2013)** propõem um *Framework* para a Mineração de Opinião em *blogs* ligados à agricultura. Vale ressaltar que os pesquisadores também propõem uma estrutura para a concepção e implementação de um *blog* para a agricultura, que foi utilizado pelos mesmos como fonte de dados. Durante um período de seis meses os autores mantiveram o *blog* e obtiveram as opiniões dos usuários do mesmo. Em seguida, foram aplicadas técnicas de mineração de opinião para extrair opiniões sobre agricultura. Os proponentes do trabalho deixam claro que o *Framework* pode ser aplicado em *blogs* já criados.

3.1.4 Framework for Opinion Mining from Web Blogs (Bele e Kesari (2015))

Em **Bele e Kesari (2015)**, as autoras alertam para a evolução das tecnologias voltadas para *Web*, bem como para a enorme quantidade de dados presentes nela e disponíveis para

quaisquer usuários interconectados à *Internet*. As pesquisadoras alertam ainda para o fenômeno da *Web* 2.0, sendo essa a principal responsável pela grande geração de conteúdo na *web* nos últimos anos. Essa funcionando como plataforma para trocar ideias, pontos de vista, pensamentos, experiências, opiniões, compartilhar informações, entre milhões de pessoas, usando *blogs* e outros *sites* de redes sociais. As autoras chamam a atenção para os *blogs*, uma vez que suas características são diferentes de outros *sites web* normais e, portanto, exigem técnicas diferentes para o processo de extração de opinião a partir de dados contidos em *blogs*, sendo esse o escopo dessa pesquisa. Nessa pesquisa, foram realizadas descrições e discussões sobre várias técnicas para mineração de opinião, as quais foram usadas para extrair opiniões de *blogs* em geral, atráves do *Framework* proposto. Segundo as autoras, esse *Framework* serve de base para a implementação de uma aplicação que auxilia o processo automatizado de mineração de opiniões em *blogs*, como o *Twitter* (tendo sido utilizado como estudo de caso), por exemplo.

3.1.5 A Survey on Opinion Mining Framework (Selvam e Abirami (2013))

No trabalho de **Selvam e Abirami (2013)**, os autores mostram a explosão das mídias sociais e as oportunidades criadas para os cidadãos expressarem publicamente suas opiniões. Eles alertam que quando se trata de dar sentido a essas opiniões, temos então um problema sério. Para os pesquisadores, esse problema é de interesse da Mineração de Opinião. Eles definem a mesma como um tipo de processamento de linguagem natural para acompanhar as opiniões do público sobre um determinado produto, assunto, artista, dentre outros. Os autores propõem um *Framework* para a Mineração de Opinião, por meio da construção de um aplicativo para coletar e examinar opiniões. Durante a pesquisa, os autores aplicam o *Framework* em várias fontes de dados, como *microblogs* (*Twitter*) e *review sites* (*Amazon*), buscando encontrar opiniões e fornecer uma boa recomendação para uma aplicação específica. O *Framework* proposto conta com um conjunto de tarefas, sendo elas: extrair a opinião, determinar a subjetividade, polaridade e força da polaridade baseada no texto contido na fonte de dados. É importante frisar que os autores apontam esse conjunto de tarefas do *Framework* como o diferencial do seu trabalho. O *Framework* auxilia ainda os usuários a tomar decisões precisas, através das opiniões extraídas, as quais são usadas para fornecer uma boa recomendação.

3.1.6 A Lexiconizing Framework of Feature-Based Opinion Mining in Tourism Industry (Muangon et al. (2014))

Em **Muangon et al. (2014)**, é apresentado um *Framework* para mineração de opinião baseado no *site* da agência de viagens na Tailândia, Agoda (www.agoda.com). Os autores mostram o crescimento dos negócios da agência nos últimos anos com o número de agentes *on-line* oferecendo reservas de hotéis. Quando os clientes precisam tomar uma decisão, eles normalmente exploram as opiniões anexadas a cada hotel no agente *on-line*. Assim, os pesquisadores propõem um *Framework* de Mineração de Opinião baseado em recursos usando três níveis principais, sendo eles a lexiconização, caracterização e polarização das palavras. Os autores propuseram a avaliação do *Framework* para Mineração de Opinião com base no conjunto de dados coletados da Agoda. Os resultados dessa avaliação são sentenças polarizadas que indicam a real opinião dos clientes. Dessa forma, os proponentes do trabalho avaliam os resultados como aceitáveis e mostram um desempenho adequado à aplicação na vida real. Vale destacar que os pesquisadores aconselham que o *Framework* seja utilizado em outros domínios para a Mineração de Opinião baseada em lexiconização.

3.2 Análise dos trabalhos relacionados

Nesta seção, os trabalhos relacionados (**Khan et al. (2014)**, **Dragoni et al. (2016)**, **Valsamidis et al. (2013)**, **Bele e Kesari (2015)**, **Selvam e Abirami (2013)**, **Muangon et al. (2014)**) são avaliados levando em consideração parâmetros que acredita-se serem relevantes para se fazer uma comparação com o trabalho de pesquisa proposto nesta dissertação. Os parâmetros utilizados na avaliação são os seguintes:

- Fonte dos dados utilizados, ou seja, a origem dos dados;
- O *Framework* é genérico;
- Utiliza dados georreferenciados;
- Fornece informações além de opiniões polarizadas (positivas/negativas);
- Tipo de classificação utilizada.

Tabela 3.1: Análise dos Trabalhos relacionados.

Trabalho	Fonte de Dados	O *Framework* é genérico?	Utiliza dados georreferenciados?	Fornece informações além de opiniões polarizadas?	Tipo de classificação utilizada
a[1]	*Microblog.*	Não.	Não.	Não.	Supervisionado.
b[2]	*Review sites.*	Sim.	Não.	Não.	Supervisionado.
c[3]	*Blog; Microblog.*	Não.	Não.	Não.	Não-Supervisionado/ Supervisionado.
d[4]	*Blog; Microblog.*	Sim.	Não.	Não.	Supervisionado.
e[5]	*Review sites; Microblog; Dataset.*	Sim.	Não.	Não.	Não-Supervisionado/ Supervisionado.
f[6]	*Microblog.*	Sim.	Não.	Sim.	Supervisionado.

[1] *TOM: Twitter opinion mining framework using hybrid classification scheme* (**Khan et al. (2014)**);
[2] *SMACk: An Argumentation Framework for Opinion Mining* (**Dragoni et al. (2016)**);
[3] *A Framework for Opinion Mining in Blogs for Agriculture* (**Valsamidis et al. (2013)**);
[4] *Framework for Opinion Mining from Web Blogs* (**Bele e Kesari (2015)**);
[5] *A Survey on Opinion Mining Framework* (**Selvam e Abirami (2013)**);
[6] *A lexiconizing framework of feature-based opinion mining in tourism industry* (**Muangon et al. (2014)**).

3.3 Síntese

Este capítulo apresentou pesquisas acadêmicas com diversos *Frameworks* que foram propostos para tratar do problema de mineração de opinião e/ou análise de sentimento. Pode-se perceber que, ao longo dos anos, uma gama considerável de esforços vem sendo realizada pelos pesquisadores para encontrar melhores soluções que auxiliem o processo de mineração de opinião e/ou análise de sentimento de maneira satisfatória.

Seis trabalhos que tratam de *Frameworks* para mineração de opinião e/ou análise de sentimento foram descritos. Os trabalhos de **Valsamidis et al. (2013)** e **Muangon et al. (2014)** foram os únicos a serem propostos para um contexto de problema único, no caso, agricultura e turismo, respectivamente.

Para finalizar o capítulo, contemplou-se critérios de avaliação para os trabalhos sugeridos. Esses critérios foram expostos em uma tabela, na qual os mesmos encontram-se elencados. Os critérios levaram em consideração medidas que permitissem uma análise comparativa com o trabalho desenvolvido nesta pesquisa.

Capítulo 4

Arcabouço proposto

Este capítulo contém os elementos básicos para o arcabouço proposto nesta pesquisa, a metodologia utilizada pelo Framework proposto, além da implemntação de um Protótipo.

4.1 Framework

Um *Framework* pode ser visto como um conjunto de blocos de *software* pré-fabricados que pode ser usado como base de desenvolvimento para novas aplicações, conforme **Aklecha (1999)**. **Roberts e Johnson (1996)** definem um *Framework* como um projeto reusável de todo ou parte de um sistema de *software*, descrito por um conjunto de classes abstratas e pela forma como as instâncias dessas classes se relacionam entre si.

Nesta seção, é apresentado o *Recognising Opinion Framework* (**ROF**) para suportar a aquisição e análise de dados em redes sociais. O **ROF** é destinado à aquisição de informação em meios de comunicação social ou redes sociais. Ele é composto por dois módulos para realizar a tarefa de aquisição de informação: um módulo de aquisição (*Manager* e *Light Client*) e um módulo de análise (*Analysis Module*). O módulo de aquisição é usado para criar uma conexão com o servidor *web* no qual os dados serão adquiridos. Após o término do processo de aquisição dos *tweets*, eles são pré-processados e armazenados em um banco de dados (uma versão original e uma cópia), ficando à disposição do módulo de análise, o qual é responsável por classificar a opinião extratificada dos *tweets*, por meio dos algoritmos apresentados na seção 2.1.2.2. A Figura 4.1 apresenta o diagrama de bloco para o *Framework* proposto nesta pesquisa, levando em consideração o modelo *Model-View-Controller* (**MVC**)[1].

[1]**MVC** - Consiste em separar dados (Model) da interface do usuário (View) e do fluxo da aplicação (Control). Site: http://www.dsc.ufcg.edu.br/ jacques/cursos/map/html/arqu/mvc/mvc.htm. Acessado em 30/12/2015.

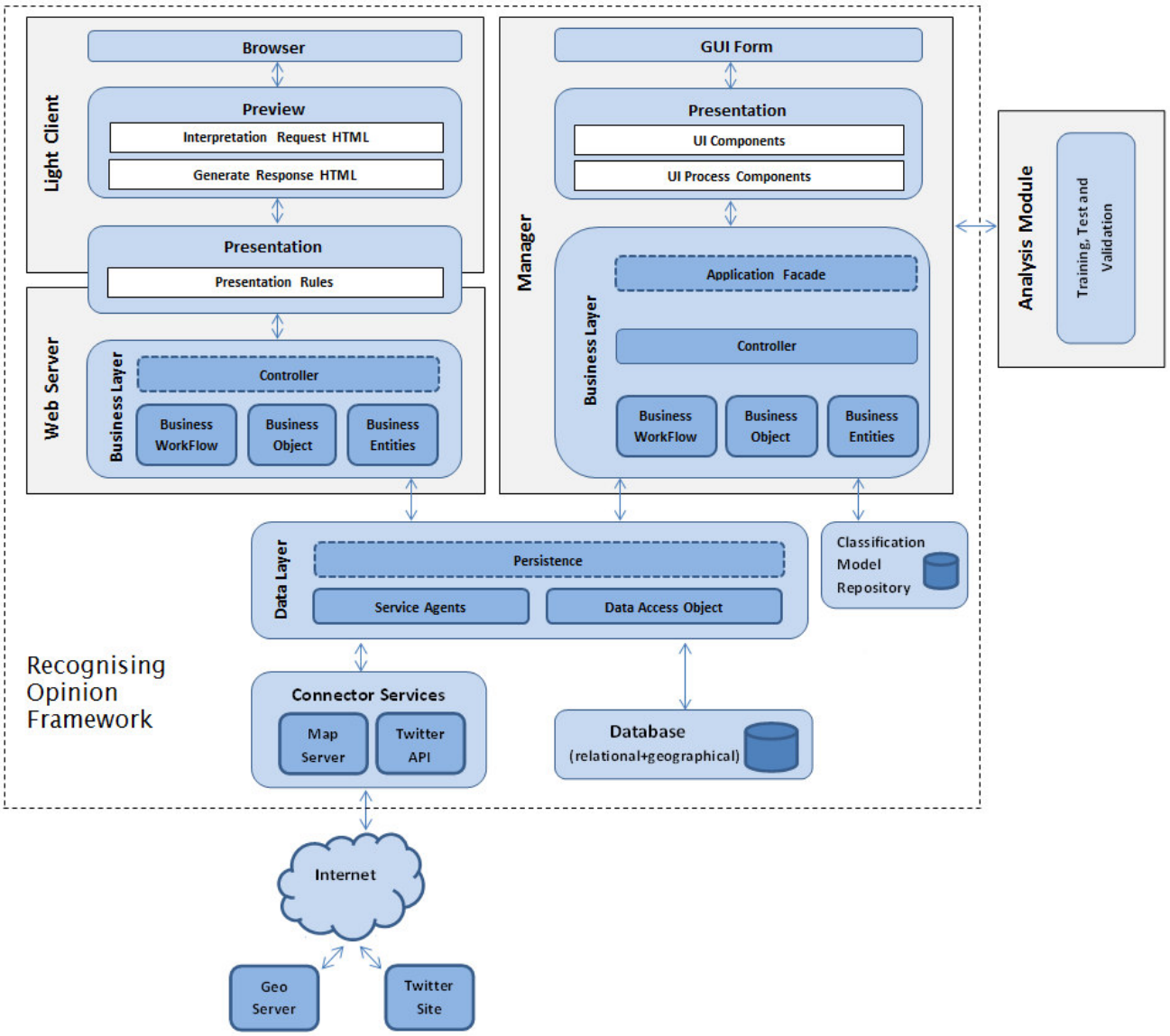

Figura 4.1: Diagrama de bloco para o *Framework* proposto.

Visualizando a Figura 4.1, pode-se identificar o módulo de aquisição à direita, formado pelo *Manager* e auxiliado pelo módulo de análise, no caso, o *Analysis Module*, esse mais à direita do *Manager*. O *Light Client* fica responsável pela apresentação. Vale ressaltar que o *Manager* e *Light Client* foram implementados usando o modelo **MVC**. Ambos contêm camadas para lógica de negócio (*Business Layer*) e acessos ao dados (*Data Layer*), contando ainda com a camada de visão (*Browser* para o *Light Client* e interface *Java Swing* para *Manager*), além da camada de fluxo da aplicação, presente nos seus respectivos *Controllers*. O *Framework* conta ainda com um repositório de modelos de classificação, no caso, o *Classification Model Repository*, o qual auxilia o trabalho do *Analysis Module* disponibilizando modelos para o processo de classificação. Ainda sobre a composição do *Framework*, observa-se o banco de dados geográfico utilizado pelo

mesmo, dando suporte ao armazenamento de dados georreferenciados. O *Connector Services* é o componente do *Framework* responsável por prover a comunicação com o *GeoServer*, utilizado para a plotagem dos mapas, além de funcionar como interface com o *site* do *Twitter*, por meio da *Search* API[2].

Por sua vez, a *Unified Modeling Language* (**UML**[3]) ou Linguagem Unificada de Modelagem, possui uma vasta gama de diagramas capazes de descrever tanto a estrutura, quanto o comportamento de um sistema de software. Dentre esses diagramas, um amplamente adotado para descrever a estrutura de sistemas é o Diagrama de Classes. Trata-se de uma representação gráfica de uma visão estática do sistema que mostra uma coleção declarativa de elementos, tais como classes (compostas por atributos e métodos), pacotes, entre outros, além de seus conteúdos e relacionamentos. A Figura 4.2 apresenta o diagrama de classe para as entidades do *Framework* proposto nesta pesquisa.

[2]**Documentação oficial** - https://dev.twitter.com/overview/documentation.

[3]**UML** - É uma linguagem padrão para modelagem orientada a objetos. Ela surgiu da fusão de três grandes métodos, do BOOCH, OMT e OOSE, de acordo com **Object Management Group (OMG) (2011)**.

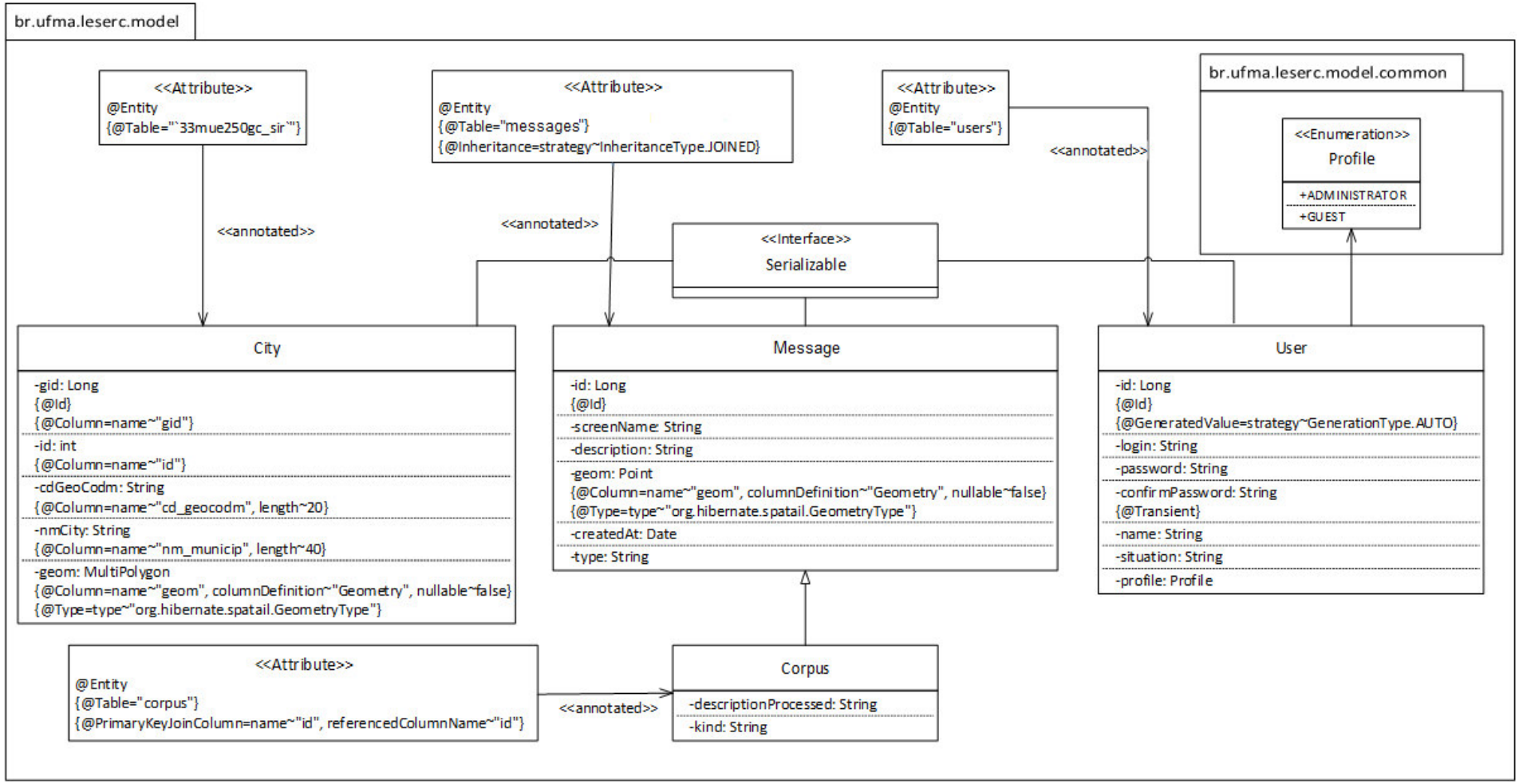

Figura 4.2: Diagrama de classe para as entidades do **ROF**.

O diagrama da Figura 4.2 apresenta o relacionamento entre as classes (*City*, *Message*, *User*, *Corpus* e *Profile* utilitárias do *Framework*), além de seus respectivos atributos e métodos.

City implementa a classe *Serializable* e conta ainda com um atributo de anotação que é responsável por informar a qual tabela do banco de dados a classe fará referência (33mue250gc_sir), bem como os atributos:

- gid (identificador geométrico, referente à coluna gid) do tipo *Long*;
- id (identificador, referente à coluna id) do tipo int, cdGeoCodm (código geométrico para o município;
- referente à coluna cd_geocodm de tamanho 20) do tipo *String*;
- nmCity (nome do município, referente à coluna nm_municip de tamanho 40) do tipo *String*;
- geom (geometria do município, referente à coluna geom) do tipo *MultiPolygon*.

Message implementa a classe *Serializable*, além de apresentar um atributo de anotação que é responsável por informar a qual tabela do banco de dados a classe fará referência (*messages*) e outro responsável por informar qual estratégia será utilizada na Herança entre *Message* e *Corpus*, bem como os atributos:

- id (identificador, referente à coluna id) do tipo *Long*;
- *screenName* (nome do usuário que realizou a postagem) do tipo *String*;
- *description* (conteúdo da postagem) do tipo *String*;
- *createdAt* (data da postagem) do tipo *Date*;
- *type* (tipo de conteúdo buscado na postagem) do tipo *String*;
- geom (geometria do ponto de postagem) do tipo *Point*.

User implementa a classe *Serializable*, conta ainda com um atributo de anotação que é responsável por informar a qual tabela do banco de dados a classe fará referência (*users*), bem como os atributos:

- id (identificador, referente à coluna id) do tipo *Long*;

- *login* (*login* utilizado para acessar a aplicação) do tipo *String*;
- *password* (senha utilizada para acessar a aplicação) do tipo *String*;
- *confirmPassword* (utilizada para confirmação de senha em caso de criação ou atualização da mesmas) do tipo *String*;
- *name* (nome do responsável pelo usuário) do tipo *String*;
- *situation* (situação do usuário, se está ativo ou inativo) do tipo *String*;
- *profile* (perfil do usuário, se é convidado ou administrador) do tipo *Profile*.

Corpus possui um atributo de anotação que é responsável por informar a qual tabela do banco de dados a classe fará referência (*corpus*), bem como os atributos:

- *descriptionProcessed* (conteúdo da postagem após a remoção de estruturas indesejadas) do tipo *String*;
- *kind* (classe atribuída após a análise do conteúdo de *descriptionProcessed*).

Por fim, a classe *Profile*, que é um *Enumeration* que compõe *User*.

4.2 Metodologia

Para atingir os objetivos deste trabalho, o *Framework* proposto conta com uma metodologia que consiste em cinco etapas. Na primeira etapa, ocorre a adquisição dos dados. Na segunda etapa, o pré-processamento é realizado, de modo a remover estruturas ruidosas. Na terceira etapa, características são extraídas dos *tweets*. Na quarta etapa, temos a classificação dos *tweets*. Na quinta e última etapa, encontram-se as opiniões polarizadas prontas para o processo de sumarização. A Figura 4.3 apresenta o conjunto de etapas da metodologia.

4.2.1 Aquisição dos dados

Os *tweets* foram coletados por meio da implementação do protótipo (vide subseção 4.3) com base no **ROF**. Para o processo de extração dos textos presentes nos *tweets*, o protótipo acessa a *Search* API[2] do *Twitter*.

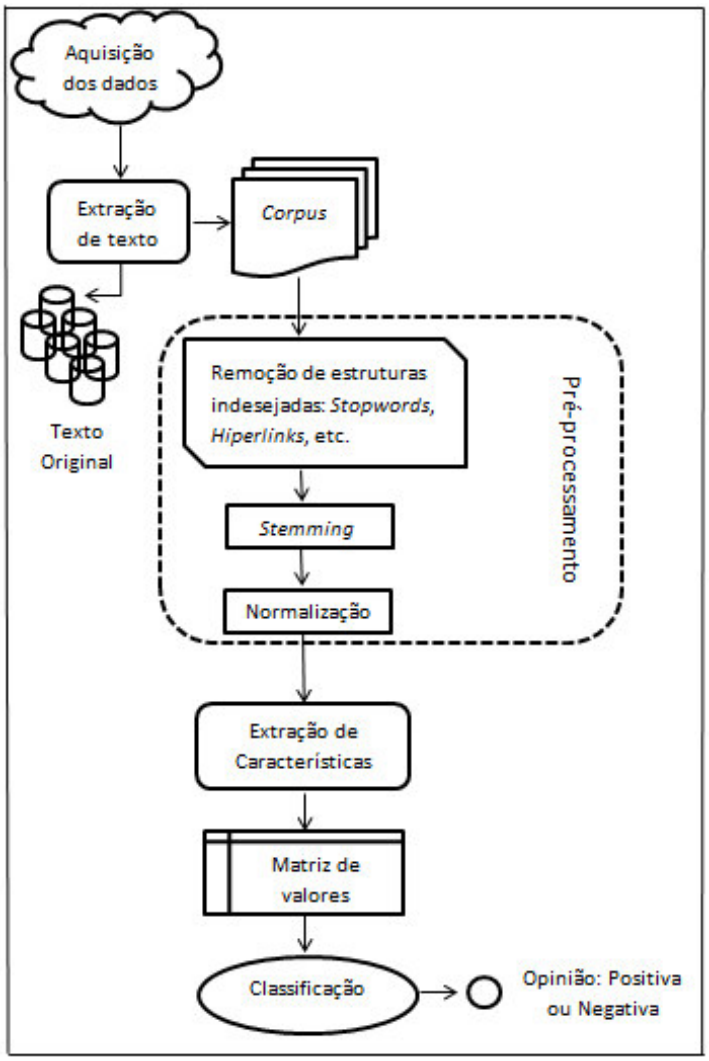

Figura 4.3: Etapas da Metodologia.

4.2.2 Pré-processamento

Antes da extração de características dos *tweets*, é importante remover as estruturas que são indesejáveis, tais como: *hiperlinks*, palavras irrelevantes, caracteres especiais, pontuação, números, dentre outros. Após as remoções, é necessário utilizar *stemming*[4] e, em seguida, aplicar uma normalização aos *tweets*. Assim, a técnica de *stemming* em conjunto com as outras técnicas apresentadas reduzem a dimensionalidade da matriz de valores gerados na próxima etapa (4.2.3), atrás da redução de termo. É importante ressaltar que o pré-processamento ocorre em cópias dos *tweets* coletados (*corpus*) na etapa anterior. Tal artifício busca manter os *tweets* originais intactos para evitar quaisquer inconsistências. Depois de passar por esses

[4]**Stemming** - O objetivo desta técnica é encontrar o radical da palavra. Esse é encontrado, na maioria das vezes, pela adequação do plural e retirada de prefixo e sufixo, resultando em um termo (**Bonzanini (2016)**). Por exemplo, as palavras computação e computador, após aplicação da técnica, são reduzidas ao mesmo radical, no caso, o termo "comput".

filtros, os *tweets* servirão como entrada para a próxima etapa, a qual será descrita na seção a seguir.

4.2.3 Extração de características

Após a etapa de pré-processamento, os *tweets* foram submetidos ao processo de extração de características, por meio dos métodos citados na subseção 2.1.3.1. Antes da extração, deve-se aplicar aos *tweets* o modelo "*bag-of-words*", que é comumente utilizado como um conjunto de recursos, levando em consideração o contexto de mineração aplicada. Nesse modelo, cada documento é representado como um vetor de termos, baseado nas técnicas de **TF-IDF** e **PCA**, onde a primeira é responsável pela representação vetorial e a última é encarregada de encontrar os termos de maior representatividade no vetor. Portanto, todos os documentos presentes no *corpus* são representados como uma matriz gigante, contendo documentos com seus termos ou palavras.

4.2.4 Classificação

Uma vez gerada a matriz de valores numéricos, esses valores são utilizados como entradas para modelos de classificação. Esses modelos determinam a polaridade do *tweet* através da identificação de entidade ou de algum aspecto.

4.3 Protótipo

Segundo **Rudd e Isensee (1994)**, protótipo é o resultado da técnica de Prototipagem, sendo essa uma técnica importante para reduzir os custos e riscos envolvidos no desenvolvimento de sistemas de *software* complexos. Para **Szekely (1994)**, o protótipo tem como principal objetivo permitir que desenvolvedores possam adquirir as informações necessárias para construir com sucesso um sistema.

A implementação do protótipo contou com algumas adequações ao modelo da *Search* API, devido a algumas restrições relacionadas à mesma. Uma dessas restrições é a busca de *tweets* por intervalo de datas. Pois, a *Search* API só consegue retroagir de 6 a 9 semanas, conforme a documentação da *Search* API. No entanto, os dados utilizados nesta pesquisa excedem esse intervalo. Visando sanar essa restrição, o *Manager* presente no módulo de aquisição do **ROF** realiza um acesso prévio ao site do *Twitter*. Nesse acesso, todos os identificadores (ids) dos

tweets de interesse são coletados, mediante uma consulta que atende a determinados critérios. A coleta dos identificadores ocorre com base no resultado da consulta HTML[5] gerada durante o acesso. Com o auxílio da ferramenta *Query*[6] (presente no *Manager*) aplicada no HTML contendo os resultados da consulta, consegue-se montar um conjunto de identificadores. Esses podem ser solicitados sem restrição de data na *Search* API e atendem aos requisitos da pesquisa, conforme a documentação *Search* API. Essa fornece uma interface REST[7]. Para obter os *tweets*, o módulo de aquisição estabelece uma conexão via HTTP[8] e, em seguida, executa solicitações do tipo GET[9].

4.3.1 Implementação do Protótipo

Os módulos de aquisição e análise presentes no **ROF** foram implementados em Java. Os componentes do módulo de aquisição, no caso, *Manager* e *Light Client*, foram implementados com base nos *Frameworks OpenSwing*[10] e *VRaptor 4*[11], respectivamente. Já o *Analysis Module* foi implementado através de uma *Interface*[12], a qual foi escrita com base na ferramenta WEKA[13]. É importante ressaltar que a ferramenta utilizada para máquina de aprendizado no presente trabalho pode ser substituída com a reescrita da interface supracitada, respeitando as definições de outras ferramentas, tais como: R[14], *RapidMiner*[15], *Scikit-learn*[16] e outras.

O conjunto de ferramentas, *softwares*, *Frameworks* e API utilizados no trabalho proposto obdecem critérios como licença de uso gratuita, portabilidade, funcionalidades disponíveis e outros, conforme apresentado no trabalho de **Neeraj Bhargava e Arya (2013)**. A Figura 4.4 apresenta o diagrama de classe para o protótipo utilizado no estudo de caso presente neste trabalho.

[5]**HTML** - Sigla para a expressão inglesa *HyperText Markup Language* (em português Linguagem de Marcação de Hipertexto) é uma linguagem de marcação utilizada na construção de páginas na *Web*, voltada para a apresentação dos dados.

[6]**Ferramenta *Query*** - É uma ferramenta de apoio a aplicação *Web*. Altamente flexível, permite muitas opções para a localização de elementos de interface no navegador e também simular comportamentos reais de um usuário, conforme descrito por **Gaur et al. (2012)**.

[7]**REST** - Sigla para a expressão inglesa *Representational State Transfer*, em português Transferência de Estado Representacional, é uma abstração da arquitetura da *Web*, focada na troca de dados.

[8]**HTTP** - Sigla para a expressão inglesa *HyperText Transfer Protocol*, em português Protocolo de Transferência de Hipertexto, é um protocolo de comunicação usado por aplicações *Web*.

[9]**GET** - Método presente no protolo HTTP responsável por realizar requisições do tipo cliente-servidor.

[10]**Documentação oficial do *OpenSwing*** - http://oswing.sourceforge.net/. Acessado em 30/12/2015.

[11]**Documentação oficial do *VRaptor 4*** - http://www.vraptor.org/pt/. Acessado em 30/12/2015.

[12]**Interface** - É uma classe contendo apenas a especificação da funcionalidade que uma classe deve prover, sem determinar como essa funcionalidade deve ser implementada, conforme descrito por **Deitel e Deitel (2004)**.

[13]**Machine Learning Group at the University of Waikato** - Documentação da Versão 3.7.12, link: http://www.cs.waikato.ac.nz/ml/weka/documentation.html.

[14]**R** - *link site*: https://www.r-project.org/.

[15]**RapidMiner** - *link site*: https://rapidminer.com/.

[16]**Scikit-learn** - *link site*: http://scikit-learn.org/.

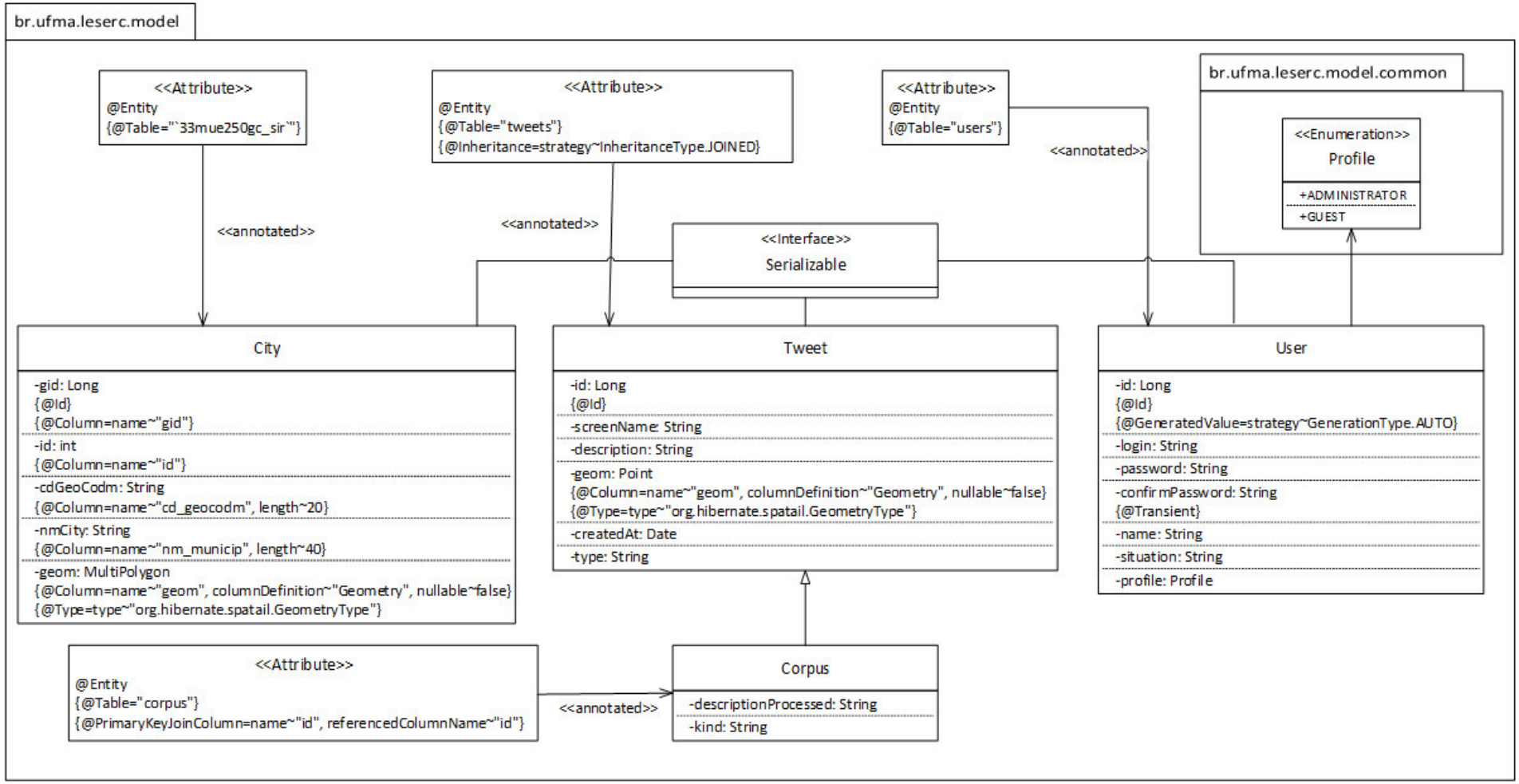

Figura 4.4: Diagrama de classe para as entidades do **ROF** especializado no **Twitter**

O diagrama da Figura 4.4 é especializado no *Twitter*, como apresentado no relacionamento de especialização entre as classes *Tweet* (superclasse) e *Corpus* (subclasse).

A Figura 4.5 apresenta um diagrama de classe que representa os diferentes componentes que formam a camada de Visão, como páginas *Web* e formulários HTML, e classes de ação (*Controllers*) do *Framework Controller*, no caso, *VRaptor 4*.

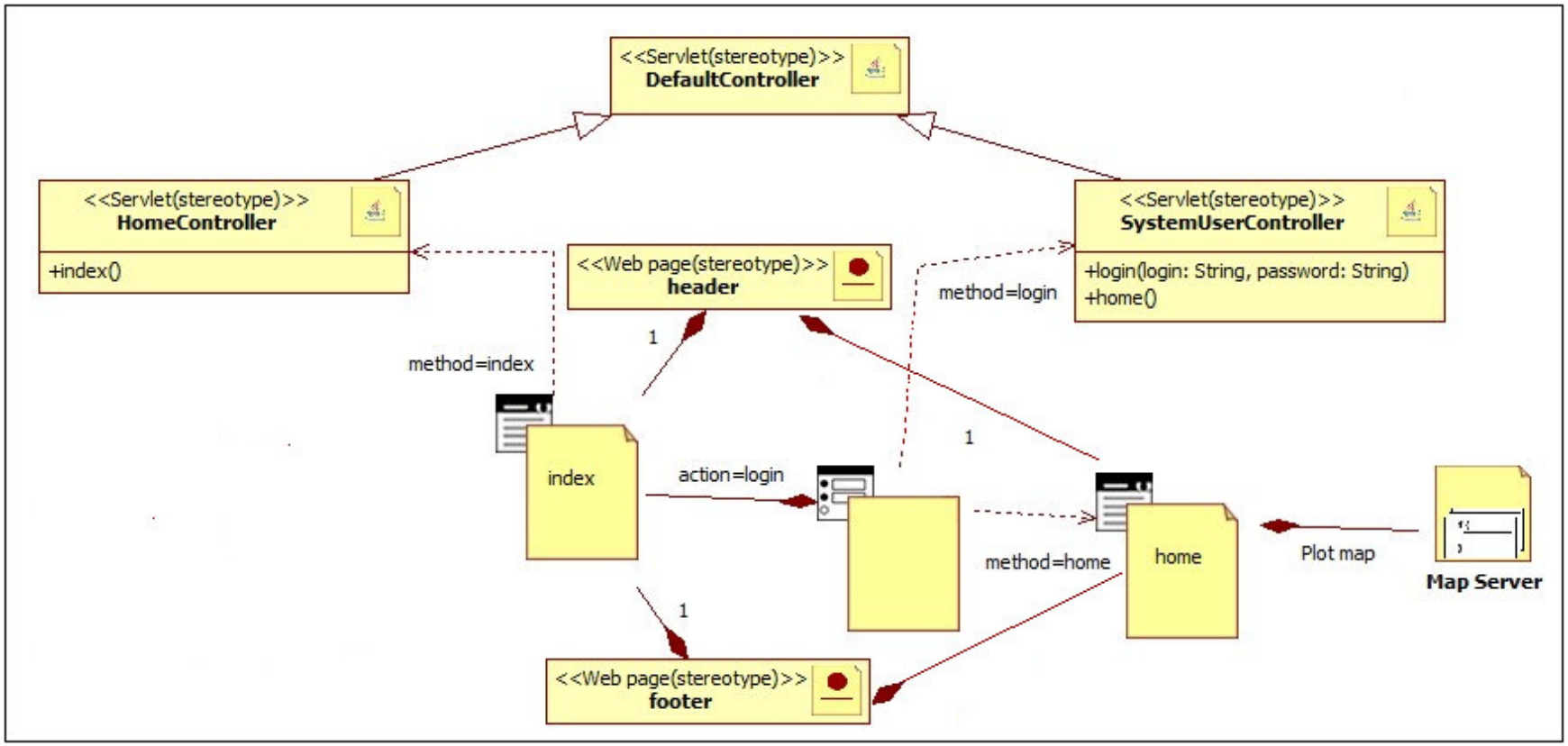

Figura 4.5: Diagrama de classe para plotagem do mapa.

A Figura 4.5 expõe o processo de plotagem do mapa, principal diferencial do *Framework* proposto, por meio do diagrama de caso de uso apresentado. O diagrama busca representar o seguinte cenário: quando o método *index* é chamado, por meio do *HomeController*, é apresentado ao usuário a página *Web index*, contendo um formulário de acesso. Quando o formulário é preenchido, o método *login* é chamado. Após a confirmação das credenciais de acesso o método *home* do *SystemUserController* redireciona o usuário para a página *home*, na qual é apresentado o mapa gerado pelo *GeoServer*, caso existam dados analisados que viabilizem a geração.

4.3.2 Ambiente de implementação

Durante o desenvolvimento do protótipo proposto, um ambiente de desenvolvimento fez-se necessário, o qual é descrito abaixo:

- ***Hardware***:
  - *UltraBook Dell System XPS L321X*;
  - Processador *Intel(R) Core(TM) i5-2467M CPU @ 1.60GHz*;
  - Memórias: *RAM 4GB e ROM SSD 128GB.*
- ***Software***:
  - *Microsoft Windows 7 Professional/64bits*;
  - *Java Development Kit (JDK) 8/64bits*;
  - *Java EE IDE for Web Developers/Mars Release (4.5.0)/64bits*, **Mars (2015)**;
  - *Apache Tomcat*:
    - 8.0.24/64*bits* (usado pelo *Light Client*);
    - 7.0.63/64*bits* (usado pelo *GeoServer*).
  - *PostgreSQL 9.3/64bits e PostGIS 2.1/64bits*;
  - *Apache Maven 3.3.3/x86_64 bits*;
  - *GeoServer 2.7.1/ x86_64 bits*;
  - *WEKA, 3.7.11/64 bits.*

Para a execução do protótipo, utilizou-se o seguinte ambiente:

- ***Hardware***:
  - Servidor Dell R/420;
  - Processador *Intel(R) Xeon(R)* CPU E5-2420 v2 @ 2x2.20GHz;
  - Memórias: RAM 32GB e ROM HD 1TB.
- ***Software***:
  - *Windows Server 2008 R2 Standard/64bits*[17];
  - *Java SE Runtime Environment 8/64bits*;
  - *Java EE IDE for Web Developers/Mars Release (4.5.0)/64bits*, **Mars (2015)**;
  - *Apache Tomcat*:
    - 8.0.24/64*bits* (usado pelo *Light Client*);
    - 7.0.63/64*bits* (usado pelo *GeoServer*).
  - *PostgreSQL 9.3/64bits e PostGIS 2.1/64bits*;
  - *GeoServer 2.7.1/ x86_64 bits*;
  - *Web application ARchive do protótipo (rof.WAR).*

4.4 Síntese

Este capítulo abordou o desenvolvimento de um *Framework* para a aquisição e análise de dados em redes sociais baseado em sistemas de informação geográfica. Para tanto, foram apresentados diagramas, protótipo, metodologia e ambientes de desenvolvimento e execução.

Assim, este capítulo apresentou uma visão geral do *Framework* e a metodologia necessária para obter e analisar dados em redes sociais. Além de apresentar os diagramas de bloco, classe e caso de uso.

[17] Os teste foram feitos na plataforma *Windows*, mas podem ser realizados em *Linux*, já que todos os *softwares* de apoio possuem versões para *Linux* ou foram feitos em Java.

Capítulo 5

Exemplo ilustrativo de aplicação do ROF

Este capítulo apresenta uma visão geral dos estudos de casos desenvolvidos, a saber, uma experiência de mineração de opiniões em nível de aspecto em dois contextos, saúde pública e política. São detalhados as bases de dados utilizadas, bem como os mapas e gráficos gerados para os distintos contextos utilizados.

5.1 Utilização do ROF

O *Framework* proposto neste trabalho foi aplicado para detecção e localização de possíveis áreas afetadas pelo mosquito *Aedes*, responsável por transmitir doenças como dengue e febre chikungunya. Doenças essas que assolam todo o Brasil, requerendo maiores cuidados por parte dos gestores em saúde pública. Outra aplicabilidade do *Framework* foi relacionada ao processo de *impeachment* da presidente Dilma Rousseff, com a intenção de determinar as opiniões acerca do *impeachment*, além de apresentar a dispersão das mesmas. Vale ressaltar que os *tweets* georreferenciados utilizados nos estudos de casos propostos restringem-se à localização da postagem, ou seja, não estão necessariamente atrelados à localização de moradia do usuário, por exemplo, uma vez que esse usuário pode realizar uma postagem a partir de seu *smartphone* ou do computador presente em seu ambiente de trabalho.

5.1.1 Contexto 1: Saúde Pública

O ROF foi utilizado no processo de detecção e localização de possíveis áreas afetadas pelo mosquito *Aedes*, para os munícipios que compõem o estado do Rio de Janeiro. Esse estado foi selecionado por apresentar dados oficiais sobre a dispersão das áreas afetadas pelo mosquito *Aedes*, conforme descrito em **LIRAa (2015)**. O processo detecção levou em consideração postagens do *Twitter* relacionadas à dengue, febre chikungunya e sintomas comuns às duas doenças (dor de cabeça e no corpo e febre), além de estarem georreferenciadas para o estado supracitado e terem sido postadas durante o mês de Maio de 2015.

5.1.1.1 Bases de Dados

Nesse contexto, a base de dados conta com 1000 amostras positivas e 1000 negativas de *tweets* para casos relacionados à dengue, febre chikungunya e sintomas comuns às duas, totalizando 2000 mil indivíduos na base de dados.

As amostras foram coletadas de maneira automática por meio da *Search* API do *Twitter*, porém o processo de rotulagem foi realizado de forma manual. Durante a rotulagem manual, primou-se pela seleção de amostras que tivessem boa representatividade para o processo de classificação, ou seja, as mais variadas possíveis. Lembrando que os *tweets* utilizados nesta subseção (Treinamento) são distintos dos usados na subseção referente ao estudo de caso deste contexto (Teste). Os *tweets* foram coletados levando em consideração postagens relacionadas as doenças de interesse e seus sintomas comuns. Esses sendo georreferenciados para os municípios que compõem o estado do Rio de Janeiro e com período de postagens distintos do mês de Maio de 2015. Assim, busca-se evitar possíveis problemas na classificação dos *tweets* como *overfitting*, por exemplo.

Abaixo é apresentada parte do arquivo montado para a base de dados, seguindo o padrão ARFF:

```
@relation 'database'

@attribute text string

@attribute class neg,pos


@data
```

```
'e eu que acordei com suspeita de dengue',pos
'sera que estou com dengue',pos
'minha vida esta mais parada do que foco de dengue ',neg
'nunca estive com dengue',neg
```

Sendo *@relation* o nome da *database*, *@attribute text* contém o texto presente nas postagens dos *tweets*, *@attribute class* é o rótulo de cada classe e *@data* contempla os indivíduos presentes na base de dados.

5.1.1.2 Modelos de Treinamento e Teste

A geração dos modelos de treinamento e teste ocorreu com o auxílio da ferramenta de máquina de aprendizado utilizada na 4.3.1. Por meio dessa, utilizou-se as implementações dos algoritmos de classificação (SVM, *Naive Bayes* e *Random Forest*), necessários para a criação dos modelos. Foram gerados cenários, respeitando os modelos de treinamento e teste, conforme:

(1) **80% das amostras para treinamento e 20% das amostras para teste;**

(2) **60% das amostras para treinamento e 40% das amostras para teste;**

(3) **40% das amostras para treinamento e 60% das amostras para teste;**

(4) **20% das amostras para treinamento e 80% das amostras para teste;**

Os modelos que apresentaram os melhores resultados irão para o repositório de modelos do **ROF**. Vale ressaltar que a seleção dos algoritmos utilizados nessa seção elegeu critérios como tempo de treinamento, número de parâmetros de configuração e número de recursos, conforme os trabalhos de **Silva et al. (2012)**, **Wu et al. (2014)** e **Azure (2016)**.

5.1.1.3 Estudo de Caso 1

Este estudo de caso busca relacionar os resultados do **ROF** com outros resultados descritos abaixo, para possibilitar uma maior contextualização do estudo por parte do leitor, bem como mostrar a viabilidade do trabalho proposto.

A Figura 5.1 apresenta na forma de mapa os resultados dos *tweets* coletados e analisados, para os municípios que compõem o estado do Rio de Janeiro. Vale frisar que cada marcador presente no mapa do **ROF** corresponde às opiniões positivas para as ocorrências de interesse.

Essas ocorrências são postagens contendo casos de suspeitas — confirmadas ou não — para as doenças, bem como denúncia de focos do mosquito *Aedes*. O intuito deste estudo de caso é mostrar possíveis tendências de dispersão em relação aos casos de doenças transmitidas pelo mosquito *Aedes* no estado do Rio de Janeiro, buscando auxiliar o combate do mesmo, sendo assim uma ferramenta auxiliar, sem todavia, substituir os métodos oficiais utilizados tradicionalmente.

A Figura 5.2 apresenta os resultados do Levantamento de Índice Rápido para *Aedes aegypti* - LIRAa (**LIRAa (2015)**) na forma de mapa, gerado com base no *shapefile*[1] do Instituto Brasileiro de Geografia e Estatística (**IBGE**), para os municípios do Rio de Janeiro. Conforme a legenda presente no mapa as regiões dos municípios são classificadas , em: risco, alerta, satisfatório e sem informação, além de estrato de risco. Essa classificação ocorre da seguinte forma: o município é dividido em grupos de 9 mil a 12 mil imóveis com características semelhantes. Em cada grupo, também chamado estrato, são pesquisados 450 imóveis. Os estratos com Índice Predial (**IP**) (**LIRAa (2015)**) de infestação inferior a 1% são considerados satisfatórios. Os índices entre 1% e 3.9% são considerados em situação de alerta. Os índices acima de 4% são considerados de risco. Todos os índices levam em consideração possíveis surtos para a dengue e febre chikungunya. Os participantes do LIRAa devem atender a pelo menos um desses critérios: apresentar mais de 100 mil habitantes, possuir um grande fluxo de turistas ou estar localizado em região de fronteira, ser capital ou município de região metropolitana.

A Figura 5.3 apresenta um Mapa da Inclusão Digital para o estado do Rio de Janeiro, com base nos microdados do Censo de 2010 do **IBGE**. O mapa faz parte de um estudo realizado pelo Centro de Políticas Sociais da Fundação Getúlio Vargas (CPS/FGV), que levou em consideração as residências que têm computador em casa com acesso à *Internet*.

Analisando as Figuras 5.2 e 5.1, pode-se perceber que em regiões onde o LIRAa classifica como sem informação, o protótipo do **ROF** consegue localizar registros positivos. É importante destacar que, apesar das divergências, o **ROF** mostrou-se capaz de localizar *tweets* positivos para todas as regiões contempladas pelo LIRAa com índices de alerta e risco.

Buscando normalizar a apresentação dos dados do mapa do **ROF** (vide Figura 5.1) em relação ao mapa do LIRAa (vide Figura 5.4), utilizou-se um gráfico, conforme a Figura 5.4. Essa normalização possibilita realizar uma comparação direta das duas abordagens com base nos dados das mesmas, por meio de valores absolutos. No caso do LIRAa, os valores são

[1] ***Shapefile download*** - ftp://geoftp.ibge.gov.br/malhas_digitais/municipio_2010/rj/rj_municipios.zip

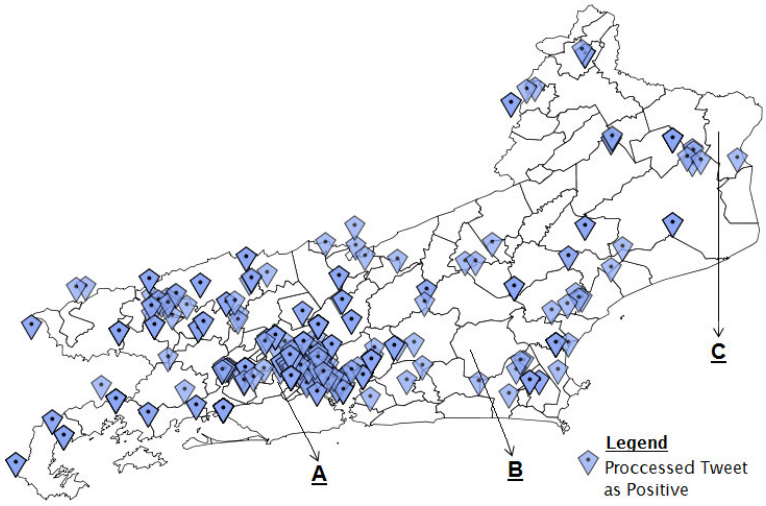

Figura 5.1: Mapa plotado pelo prótotipo do **ROF** indicando os *tweets* processados como positivos.

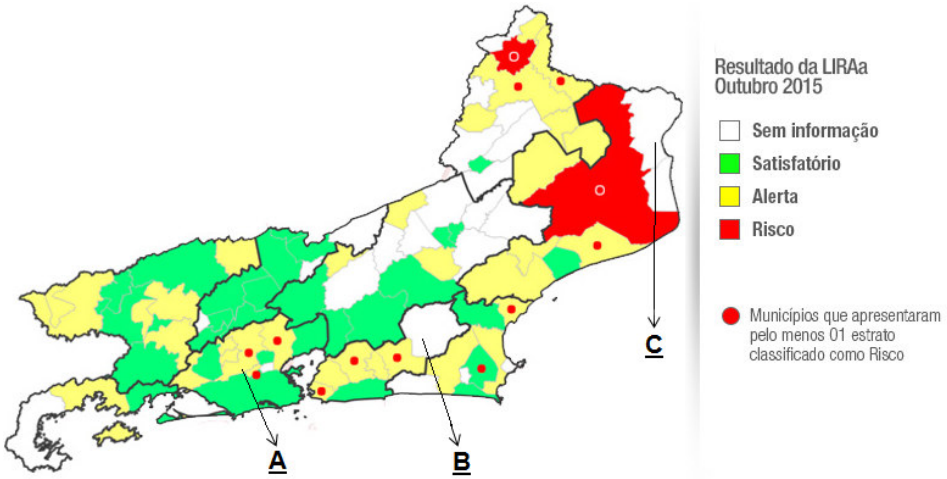

Figura 5.2: Mapa do LIRAa para o Estado do Rio de Janeiro. Imagem adaptada da fonte **LIRAa (2015)**.

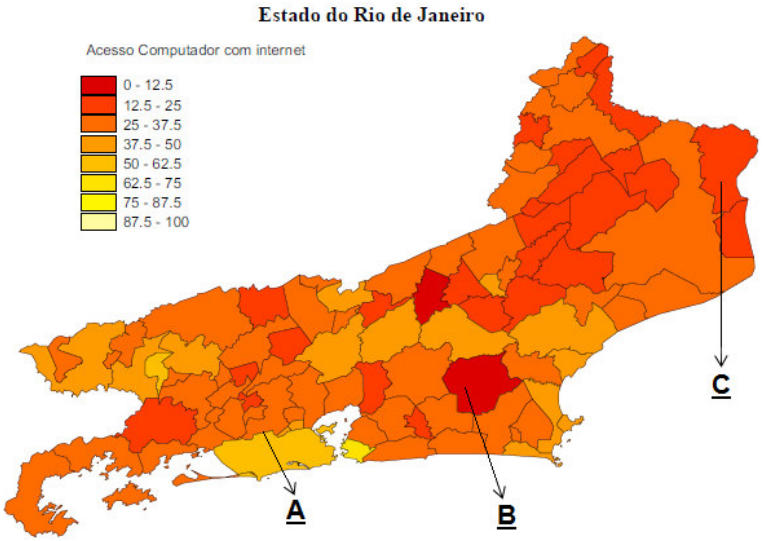

Figura 5.3: Mapa da Inclusão Digital para o estado do Rio de Janeiro. Adaptado de **CPS/FGV (2010)**.

referentes aos índices dos estratos de infestação (alerta e risco). Já para o **ROF**, são usados os valores referentes aos *tweets* coletados e classificados como positivos. A Figura 5.4 apresenta os resultados percentuais para o LIRAa e o **ROF** por região, respectivamente.

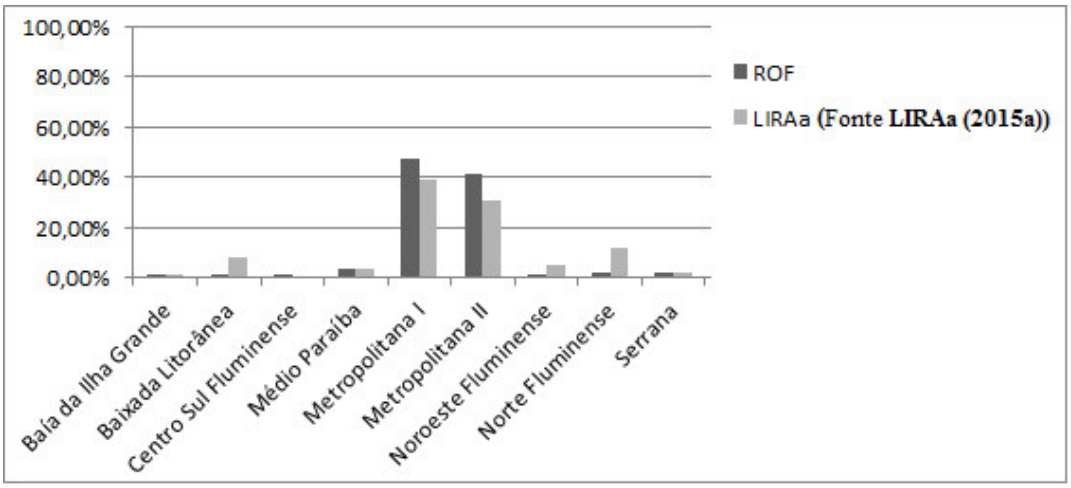

Figura 5.4: Gráfico com os valores percentuais do **ROF** e do **LIRAa**.

Com base no gráfico da Figura 5.4, pode-se verificar algumas semelhanças entre os valores percentuais obtidos pelo **ROF** e aqueles fornecidos pelo LIRAa para o estado do Rio de Janeiro. Um bom exemplo são as regiões Metropolitana I e II, que apresentam valores consideráveis para índices de alerta e risco no LIRAa. Valores esses que se mostram equivalentes aos encontrados para as mesmas regiões, na análise do **ROF**. Essa análise apresenta valores expressivos para essas regiões, considerando os *tweets* positivos para as ocorrências desejadas. Assim, em uma região do LIRAa que apresente valores consideráveis para os índices de alerta e risco, a tendência é que exista um número elevado de *tweets* classificados como positivos pelo **ROF**. No entanto, houve situações em que essa tendência não se confirmou, como é o caso da região Norte Fluminense, que apresenta valores expressivos para índices de alerta e risco no LIRAa, enquanto no **ROF** a quantidade de *tweets* positivios é bem inferior ao esperado. Além disso, na região do centro Sul Fluminense, não foi possível realizar a análise comparativa, uma vez que o LIRAa não dispõe de dados, apesar do **ROF** apresentar *tweets* positivos para essa região.

Considerando a Figura 5.3, verifica-se um dos motivos para as divergências entre os mapas do LIRAa (vide Figura 5.2) e do **ROF** (vide Figura 5.1), no caso, a falta de acesso à *Internet*. É possível perceber a predominância de áreas com baixas taxas de inclusão digital na maior parte dos municípios que compõem o estado do Rio de Janeiro, onde apenas 37.5% a 50.0% dos moradores acima dos 15 anos têm computador em casa com acesso à *Internet*, de acordo com o **CPS/FGV (2010)**.

Pode-se afirmar que o **ROF** mostra tendências para as regiões que depois podem ser comprovadas ou não com os dados do **LIRAa (2015)**. Isso ocorre devido o LIRAa utilizar dados passados, enquanto que o **ROF** pode utilizar dados passados ou não. No estudo de caso proposto para este contexto, utilizou-se dados passados, porém é possível realizar a coleta de dados em tempo real. Essa coleta diária pode verificar possíveis tendências e, com isso permitir o direcionamento de ações, que podem resultar em uma atenção prioritária para uma região detectada pelo **ROF**, a exemplo do envio de agentes de saúde para averiguar a situação, evitando assim um possível surto dessas doenças, e mesmo de outras transmitidas pelo mesmo vetor, como a zika.

Levando em consideração os mapas apresentados, pode-se realizar uma análise conjunta dos três para cada um dos municípios indicados pelas letras **A**, **B** e **C**. No município **A**, verifica-se que os dados apresentados pelos mapas do **ROF** (vide Figura 5.1), LIRAa (vide Figura 5.2) e de inclusão digital (vide Figura 5.3) encontram-se em consonância. Essa consonância é revelada mediante a alta concentração de *tweets* positivos detectados pelo **ROF**, os estratos de alerta e risco que foram apontados pelo LIRAa e uma alta disponibilidade de acesso à *Internet*, com base no mapa de inclusão digital.

Considerando o município **B**, observa-se que os dados apresentados pelos mapas do **ROF** (vide Figura 5.1), LIRAa (vide Figura 5.2) e de inclusão digital (vide Figura 5.3), mais uma vez, encontram-se em conformidade. Dessa vez, devido ao fato do **ROF** não encontrar *tweets* positivos, bem como o LIRAa (**LIRAa (2015)**) não apresentar informações em relação ao município, além de uma baixíssima disponibilidade de acesso à *Internet*, conforme o mapa de inclusão digital.

Por último, a análise do município **C**, apresenta distorções entre os mapas do **ROF** (vide Figura 5.1), LIRAa (vide Figura 5.2) e de inclusão digital (vide Figura 5.3). Essas distorções ficam evidentes pela ausência de *tweets* positivos para a análise do **ROF**, presença de estratos de alerta e risco apontados pelo LIRAa e baixa disponibilidade de acesso à *Internet*, de acordo com o mapa de inclusão digital.

O estudo de caso proposto neste contexto poderia ser realizado em relação a outro mês de ocorrência do LIRAa. Nesse novo estudo, pode-se dispensar as etapas de treinamento e teste, já que foram obtidos seus melhores modelos no estudo presente e os mesmos já residem no repositório de modelos do **ROF**.

5.1.2 Contexto 2: Política

O **ROF** foi utilizado no processo de verificação de posicionamento da população com relação ao *impeachment* da presidente do Brasil,segmentando as opiniões de acordo com as cinco regiões dos país. Esse processo levou em consideração postagens do *Twitter* relacionadas ao *impeachment* da presidente, que além de estarem georreferenciadas de acordo as regiões supracitadas, tivessem sido postadas durante o mês de Março de 2016.

5.1.2.1 Bases de Dados

A base de dados conta com 500 amostras positivas e 500 negativas de *tweets* referentes ao processo de *impeachment* da presidente Dilma, totalizando 1000 indivíduos na base de dados. As amostras foram coletadas de maneira automática por meio da *Search* API, porém o processo de rotulagem foi realizado de forma manual. Durante a rotulagem manual primou-se pela seleção de amostras que tivessem boa representatividade para o processo de classificação, ou seja, as mais variadas possíveis. Lembrando que os *tweets* utilizados nesta subseção são distintos dos usados na subseção referente ao estudo de caso. Esses sendo georreferenciados para as capitais e distrito federal que compõem o Brasil e tendo períodos de postagens distintos do mês de Março de 2016.

Abaixo é apresentado parte do arquivo montado para a base de dados, seguindo o padrão ARFF:

```
@relation 'database'

@attribute text string

@attribute class neg,pos


@data
'nao vai ter golpe vai ter impeachment',pos
'impeachment ja',pos
'impeachment e golpe',neg
'nao ao golpe do impeachment',neg
```

5.1.2.2 Modelos de Treinamento e Teste

Os modelos de treinamento e teste foram gerados de maneira similar aos gerados na subseção 5.1.1.2, respeitando as mesmas proporções.

5.1.2.3 Estudo de Caso 2

Com o intuito de ressaltar a viabilidade do trabalho e proporcionar uma melhor compreensão do presente estudo, os resultados do **ROF** foram relacionados a outros resultados, que serão descritos a seguir.

Foram analisados um total de 1.218 *tweets* georreferenciados, ressaltando que os *tweets* não-georreferenciados foram descartados. Essas postagens são referentes ao mês de Março de 2016, ligadas ao processo de *impeachment* da presidente do Brasil. Esse período foi selecionado com base em duas grandes manifestações agendadas para o referente mês. A primeira manifestação[2] favorável ao *impeachment* ocorreu no dia 13 e a segunda manifestação[3], contrária, no dia 31.

A Figura 5.5 apresenta na forma de mapa os resultados dos *tweets* coletados e analisados de acordo com as regiões do Brasil, utilizando o ROF.

[2] **Confira o local e o horário das manifestações de 13 de março | Congresso em Foco** - Site: http://congressoemfoco.uol.com.br/noticias/confira-o-horario-e-o-local-das-manifestacoes-de-13-de-marco/. Acessado em 08/03/2016.

[3] **Manifestações contra o golpe estão agendadas para esta quinta-feira (31/03)** - Site: http://www.pragmatismopolitico.com.br/2016/03/manifestacoes-contra-o-golpe-estao-agendadas-para-esta-quinta-feira-3103.html. Acessado em 08/03/2016.

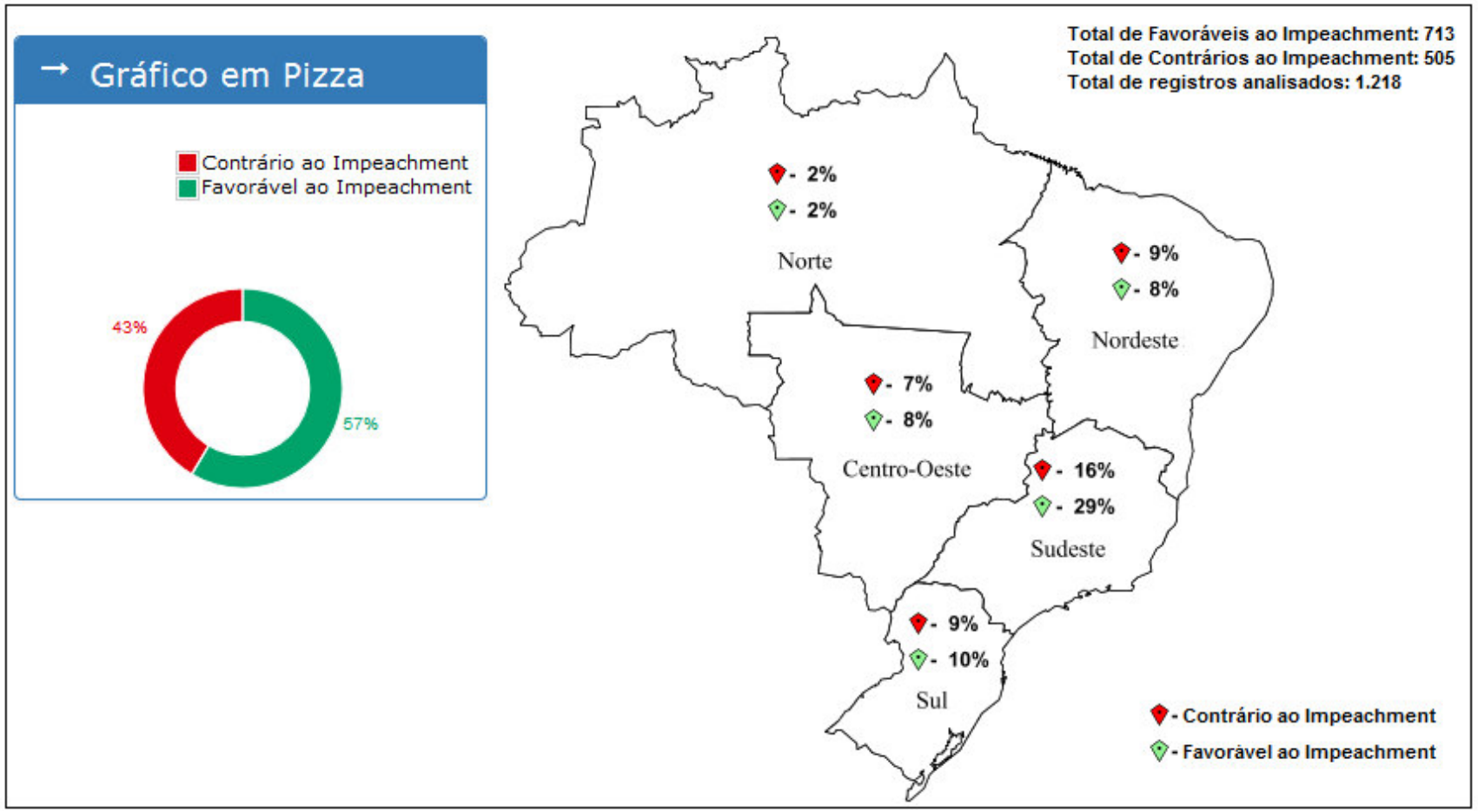

Figura 5.5: Mapa com as tendências do *impeachment* de acordo com as regiões do Brasil, com base no ROF.

Analisando Figura 5.5, pode-se perceber que nas regiões Centro-Oeste, Sudeste e Sul do país existe a maioria de registros favoráveis ao *impeachment*. Levando em consideração a região Nordeste do mapa, tem-se pequena maioria de registros contrários ao *impeachment*. A região Norte é a única entre as cinco regiões que apresenta resultados iguais para as opiniões.

É importante destacar que outras pesquisas relacionadas ao processo de *impeachment* já vêm sendo realizadas desde o ano de 2015 no Brasil, quando surgiram os primeiros indícios para o processo. Uma dessas pesquisas se assemelha bastante com a pesquisa presente neste trabalho, tendo sido apresenta na reportagem da revista Veja[4]. Nela a revista expõe resultados de uma pesquisa realizada em redes sociais pela empresa Torabit[5], na qual 49.3% das postagens em redes sociais são favoráveis ao *impeachment* e apenas 31.7% contrárias. Considerando os resultados da reportagem e os da metodologia proposta, pode-se perceber que o presente trabalho apresenta tendências válidas em relação ao processo de *impeachment*. Vale ressaltar que o trabalho proposto informa tendências por região o que não acontece com o trabalho realizado pela Torabit.

Buscando normalizar a apresentação dos dados contidos no mapa plotado pelo ROF (vide Figura 5.5), utilizou-se um gráfico (vide Figura 5.6), no qual, pode-se perceber de forma simplificada os percentuais por região para cada uma das opiniões, sejam elas favoráveis ou contrárias ao *impeachment*.

O trabalho proposto apresenta informações para as regiões que depois podem ser comprovadas por meio de pesquisas tradicionais de sondagem. Isso ocorre devido essas pesquisas utilizarem dados passados, enquanto que o trabalho proposto pode utilizar dados passados ou não. No estudo de caso proposto, utilizou-se dados atuais referentes ao mês de Março de 2016 (enquanto o artigo apresentado na seção 6.4 foi confeccionado), porém nada impede de serem coletados dados passados. Essa coleta diária permite verificar possíveis tendências e realizar direcionamento de ações, seja pelos movimentos favoráveis ao *impeachment* ou por parte dos contrários ao mesmo, ou seja, podem resultar em uma atenção prioritária para uma determinada região.

Novos estudos que tenham o intuito de verificar possíveis tendências para o processo de *impeachment* podem dispensar as etapas de treinamento e teste, já que foram obtidos os modelos no presente estudo e os mesmos poderiam ser aproveitados.

[4] **49% das menções em redes sociais são pró-impeachment, mostra estudo | Radar on-line | VEJA.com** - Site: http://veja.abril.com.br/blog/radar-on-line/sem-categoria/49-das-mencoes-em-redes-sociais-sao-pro-impeachment-mostra-estudo/. Acessado em 08/03/2016.

[5] **Home - Torabit** - Site: http://www.torabit.com.br/. Acessado em 08/03/2016.

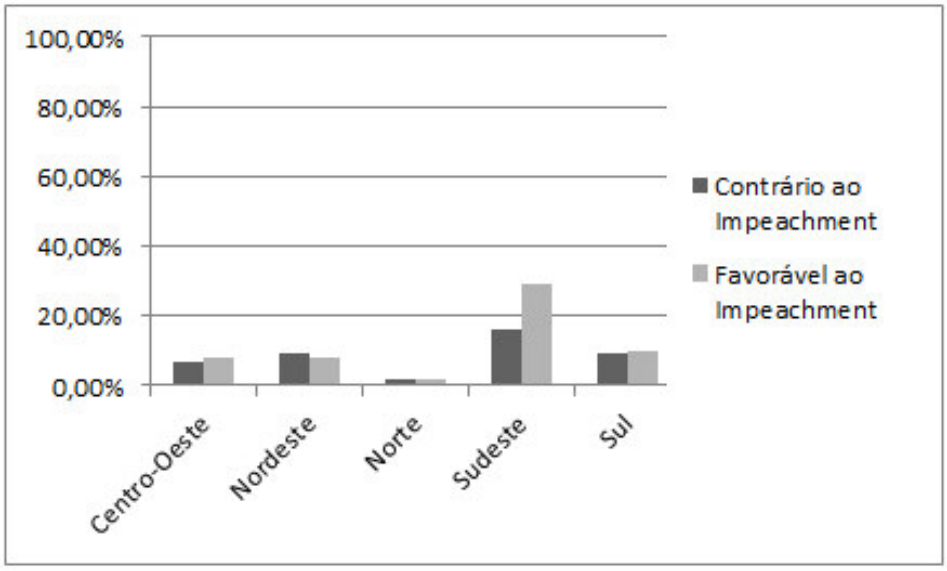

Figura 5.6: Gráfico de tendências para o *impeachment* para regiões do Brasil, com base no ROF.

5.2 Comparação do Trabalho de Pesquisa com os Trabalhos Relacionados

Nesta seção, é apresentada uma comparação entre os resultados dos trabalhos relacionados descritos no capítulo 3 e o trabalho proposto. Esse comparativo está sintetizado na Tabela 5.1, onde constam os mesmos parâmetros utilizados na avaliação dos trabalhos relacionados contemplados na subseção 3.2. Observa-se na Tabela 5.1 que o trabalho proposto apresenta valores significativos em relação aos trabalhos relacionados. Para uma comparação fiel entre esses trabalhos, faz-se necessário que as bases de dados e domínio do problema sejam os mesmos. Com base nesse preceito, utilizou-se os mesmos parâmetros contemplados na Tabela 3.1.

O diferencial desta pesquisa em comparação com os trabalhos relacionados é a utilização de dados georreferenciados, o que permite traçar um perfil mais preciso dos usuários de redes sociais e relacioná-los com um determinado tema.

Dessa forma, o trabalho proposto atingiu seu objetivo de minerar opiniões em redes sociais, baseado em sistemas de informação geográfica.

5.3 Síntese

Neste capítulo, a implementação do *Framework* para a aquisição e análise de dados em redes sociais foi apresentada. Essa implementação levou em consideração a arquitetura proposta

Tabela 5.1: Comparação do trabalho proposto com os trabalhos relacionados.

Trabalho	Fonte de Dados	O *Framework* é genérico?	Utiliza dados georreferenciados?	Fornece informações além de opiniões polarizadas?	Tipo de classificação utilizada
a[1]	*Microblog.*	Não.	Não.	Não.	Supervisionado.
b[2]	*Review sites.*	Sim.	Não.	Não.	Supervisionado.
c[3]	*Blog; Microblog.*	Não.	Não.	Não.	Não-Supervisionado/ Supervisionado.
d[4]	*Blog; Microblog.*	Sim.	Não.	Não.	Supervisionado.
e[5]	*Review sites; Microblog; Dataset.*	Sim.	Não.	Não.	Não-Supervisionado/ Supervisionado.
f[6]	*Microblog.*	Sim.	Não.	Sim (Gráficos).	Supervisionado.
g[7]	*Microblog.*	Sim.	Sim.	Sim (Gráficos e Mapas).	Supervisionado.

[1] *TOM: Twitter opinion mining framework using hybrid classification scheme* (**Khan et al. (2014)**);
[2] *SMACk: An Argumentation Framework for Opinion Mining* (**Dragoni et al. (2016)**);
[3] *A Framework for Opinion Mining in Blogs for Agriculture* (**Valsamidis et al. (2013)**);
[4] *Framework for Opinion Mining from Web Blogs* (**Bele e Kesari (2015)**);
[5] *A Survey on Opinion Mining Framework* (**Selvam e Abirami (2013)**);
[6] *A lexiconizing framework of feature-based opinion mining in tourism industry* (**Muangon et al. (2014)**).
[7] *Trabalho Proposto.*

para o *Framework*, bem como sua metodologia.

Utilizando os modelos de treinamento e teste definidos previamente e levando em consideração as bases de dados descritas neste capítulo, pôde-se avaliar os resultados gerados pelo módulo de análise do *Framework*. Os resultados do processo de avaliação também foram apresentados em forma de tabela.

Além disso, dois estudos de caso foram realizados, objetivando atestar a viabilidade do *Framework* proposto em relação ao LIRAa e outro em relação ao *impeachment* da presidente do Brasil.

Para finalizar o capítulo, critérios de avaliação dos trabalhos relacionados foram comparados com o trabalho proposto. Esta comparação foi exposta em uma tabela. Os critérios foram sugeridos levando em consideração medidas que permitissem uma análise comparativa com o trabalho proposto nesta pesquisa.

Capítulo 6

Conclusões e Trabalhos Futuros

Este capítulo expõe os objetivos alcançados com a pesquisa e suas limitações. Além disso, contribuições científicas, sociais e os trabalhos futuros são apresentados.

6.1 Conclusões do trabalho

Este trabalho apresenta a proposta de um *Framework* genérico aplicado à mineração de opinião em redes sociais para auxiliar a tomada de decisão, baseado em sistemas de informação geográfica. O *Framework* emprega uma metodologia, para aquisição e análise de dados, na busca por gerar conhecimento útil.

Um protótipo foi desenvolvido baseado no *Framework*, com a finalidade de adquirir dados do *Twitter*, processá-los e disponibilizar informação. Dois estudos de caso foram apresentados: o primeiro permite a detecção para casos de dengue, febre chikungunya e sintomas comuns às duas doenças, enquanto o segundo possibilita identificar tendências em relação ao processo de *impeachment* da presidente do Brasil.

Assim, o trabalho proposto gera uma ferramenta capaz de auxiliar a mineração de opinião em redes sociais de forma genérica, simples, com baixo custo, rápida identificação e localização das opiniões analisadas, por meio do georreferenciamento dessas opiniões.

6.2 Objetivos alcançados

O principal objetivo proposto do trabalho de pesquisa é a criação de um *framework* que se apresente como uma solução viável para auxiliar a tomada de decisão, de usuários.

Para alcançar este objetivo, alguns objetivos específicos foram propostos, como visto na

subseção 1.3.2, contida no capítulo 1. Os seguintes objetivos foram alcançados no decorrer do trabalho de pesquisa:

- Propor um *Framework* baseado em redes sociais, geoprocessamento e mineração de dados que venha a auxiliar a aquisição automática de informação (opinião) em redes sociais. Esse objetivo específico foi alcançado com a criação do **ROF**, que suporta a aquisição e análise de dados georeferenciados em redes sociais. Além disso, foi construído um protótipo para avaliação de usabilidade e viabilidade do *Framework*;

- Estudar, implementar e/ou reutilizar algoritmos voltados à mineração de dados e de opinião, sua aplicabilidade para o reconhecimento de padrões e a busca do conhecimento em conteúdos de redes sociais. O dado objetivo específico foi contemplado, uma vez que a utilização de ferramentas, como o WEKA, veio a culminar com a concepção do *Framework* e do protótipo consequentemente;

- Propor estudos de caso visando relacionar registros do *Twitter* com registros do mundo real, mais especificamente em saúde pública e política, conforme apresentado no capítulo 5. Esse objetivo específico foi alcançado mediante a utilização do **ROF** em dois estudos de caso. Um para identificar casos de dengue, febre chikungunya e sintomas comuns às duas doenças ou mesmo denúncias sobre focos do mosquito vetor, conforme apresentado na subseção 5.1.1.3. Outro para a verificação de tendências em relação ao processo de *impeachment* da presidente nas cinco regiões do Brasil, conforme apresentado na subseção 5.1.2.3.

A aplicação do protótipo do *Framework* proposto em uma situação real demonstrou uma diminuição do tempo necessário para identificar regiões que possam estar em situação de risco ou alerta para as doenças que possuam o *Aedes* como agente transmissor. No exemplo utilizado no estudo de caso aplicado à saúde pública (vide subseção 5.1.1.3), pode-se identificar que o **ROF** foi capaz de demonstrar as possíveis regiões acometidas pelas doenças supracitadas, de maneira automática, rápida e segura. Identificar essas regiões utilizando os métodos tradicionais pode ser muito demorado e muitas vezes impossível, uma vez que a simples ausência de um agente de saúde, por exemplo, pode inviabilizar todo o processo, além de requerer muito mais recursos (financeiros, humanos, tempo e outros) que o *Framework* proposto. Com o protótipo, o tempo de busca e análise foi reduzido, permitindo ainda, a redução de outros custos, principalmente

os financeiros. O mesmo acontece no estudo de caso aplicado à política (vide subseção 5.1.2.3). Pode-se determinar rapidamente tendências em relação ao *impeachment* ao longo das cinco regiões do Brasil.

A possibilidade da utilização de uma ferramenta que ofereça informações em tempo real constitui-se um avanço, uma vez que as técnicas tradicionais não remetem informações em tempo real.

6.3 Trabalhos futuros

Os trabalhos futuros e melhorias que podem vir a surgir a partir desta pesquisa são apresentados como segue:

- Aperfeiçoar o *framework* para que os módulos presentes no mesmo venham a ser implementados de forma homogênea, ou seja, utilizando um único tipo de aplicação, sendo ela totalmente *web*;
- Complementar o *framework* para que ele possa suportar outras técnicas de extração de características, como a Indexação Semântica Latente;
- Utilizar o *framework* proposto mediante o contexto de *Big Data*;
- Adaptar o *framework* para utilizar métodos de agrupamento, ao invés de apenas métodos de classificação.

6.4 Trabalhos Publicados

Como resultado desta pesquisa, foi apresentado o seguinte trabalho:

- Congresso (autor):

  Neto, Gilberto N.; Lopes, Denivaldo; Abdelouahab, Zair. “Opinion Analysis Applied to Politics: A case study based on Twitter”. 3rd Annual International Symposium on Information Management and Big Data (SIMBig 2016) on Spring, 2016.

Referências Bibliográficas

Abilio, R., Morais, F., Vale, G., Oliveira, C., Pereira, D., e Costa, H. (2015). Applying information retrieval techniques to detect duplicates and to rank references in the preliminary phases of systematic literature reviews. *CLEI Electron. J.*, 18(2).

Aklecha, V. (1999). *Object-Oriented Frameworks Using C++ and CORBA: Gold Book*. Coriolis Group Books, Scottsdale, AZ, USA.

Azure, M. (2016). How to choose algorithms for microsoft azure machine learning. `http://www.cps.fgv.br/cps/bd/mid2012/MID_sumario.pdf`. (Acessado em 09/08/2016).

Bele, N. e Kesari, B. (2015). Framework for opinion mining from web blogs.

Bonzanini, M. (2016). *Mastering Social Media Mining with Python*. PACKT PUB.

Breiman, L. (2001). Random forests. *Mach. Learn.*, 45(1):5–32.

Burrough, P. A. (2006). *Principles of geographical information systems for land resources assessment*. Monographs on soil and resources survey. Clarendon Press New York, Oxford, Oxfordshire. Reprinted with corrections: 1987, 1988, 1989 (twice), 1990, 2006.

Carr, T. R. (2003). Public information technology. chapter Geographic Information Systems in the Public Sector, páginas 252–270. IGI Global, Hershey, PA, USA.

Casanova A., C. Gilberto, D. C. J., Vinhas, L., e de Queiroz, G. R. (2005). *Representação computacional de dados geográficos*. Mundogeo, 1st edição.

Chawla, N. V. (2005). Data mining for imbalanced datasets: An overview. In Maimon, O. e Rokach, L., editors, *The Data Mining and Knowledge Discovery Handbook*, páginas 853–867. Springer.

CPS/FGV (2010). Mapa da inclusão digital. `http://www.cps.fgv.br/cps/bd/mid2012/MID_sumario.pdf`. (Acessado em 01/10/2015).

Culotta, A. (2010). Towards detecting influenza epidemics by analyzing twitter messages. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, páginas 115–122, New York, NY, USA. ACM.

Dave, K., Lawrence, S., e Pennock, D. M. (2003). Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, páginas 519–528, New York, NY, USA. ACM.

Deitel, H. M. e Deitel, P. J. (2004). *Java How to Program (6th Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Dragoni, M., da Costa Pereira, C., Tettamanzi, A. G. B., e Villata, S. (2016). Smack: An argumentation framework for opinion mining.

Fayyad, U., Piatetsky-shapiro, G., e Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54.

Fu, Z., Robles-Kelly, A., e Zhou, J. (2010). Mixing linear svms for nonlinear classification. *IEEE Transactions on Neural Networks*, 21(12):1963–1975.

Ganeshbhai, S. Y. e Shah, B. K. (2015). Feature based opinion mining: A survey. In *Advance Computing Conference (IACC), 2015 IEEE International*, páginas 919–923.

Gaur, D., Rajender, D., e Chhillar, S. (2012). www.ijcsms.com implementation of selenium with junit and test-ng. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.257.7393&rep=rep1&type=pdf`. (Acessado em 30/11/2015).

Huang, Z. e Xu, Z. (2011). A method of using geoserver to publish economy geographical information. In *Control, Automation and Systems Engineering (CASE), 2011 International Conference on*, páginas 1–4.

Kantardzic, M. (2002). *Data Mining: Concepts, Models, Methods and Algorithms*. John Wiley & Sons, Inc., New York, NY, USA.

Khan, F. H., Bashir, S., e Qamar, U. (2014). Tom: Twitter opinion mining framework using hybrid classification scheme. *Decision Support Systems*, 57:245–257.

Kolkur, S., Dantal, G., e Mahe, R. (2015). Study of different levels for sentiment analysis. *International Journal of Current Engineering and Technology*, 5(2):768–770.

Lin, L., Li, J., Zhang, R., Yu, W., e Sun, C. (2014). Opinion mining and sentiment analysis in social networks: A retweeting structure-aware approach. In *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*, páginas 890–895.

LIRAa (2015). Informe epidemiolÓgico 005/2015. `http://www.riocomsaude.rj.gov.br/Publico/MostrarArquivo.aspx?C=naDnfkrU3tw%3D`. (Acessado em 01/10/2015).

Liu, B. (2012). Sentiment analysis and opinion mining.

Lobo, M.-J., Pietriga, E., e Appert, C. (2015). An evaluation of interactive map comparison techniques. páginas 3573–3582.

Mars, E. (2015). Mars | projects.eclipse.org. `https://projects.eclipse.org/releases/mars`. (Acessado em 30/12/2015).

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edição.

Muangon, A., Thammaboosadee, S., e Haruechaiyasak, C. (2014). A lexiconizing framework of feature-based opinion mining in tourism industry. In *Digital Information and Communication Technology and it's Applications (DICTAP), 2014 Fourth International Conference on*, páginas 169–173.

Neeraj Bhargava, A. A. e Arya, R. (2013). Selection criteria for data mining software: A study. *IJCSI International Journal of Computer Science Issues*, 10(2):308 – 312.

Nuhcan Akçit, Emrah Tomur, M. O. K. (2014). Geographical information systems participating into the pervasive computing. In *GEOProcessing 2014, The Sixth International Conference on Advanced Geographic Information Systems, Applications, and Services*, páginas 129–137. ThinkMind.

Object Management Group (OMG) (2011). Uml 2.4.1 superstructure specification.

Pang, B. e Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Pipanmaekaporn, L. e Li, Y. (2012). A pattern discovery model for effective text mining. In *Proceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM'12, páginas 540–554, Berlin, Heidelberg. Springer-Verlag.

Projects, B. D. (2016). Babelomics - predictors methods - bioinformatic department projects. `http://docs.bioinfo.cipf.es/projects/1/wiki/Predictors_methods`. (Acessado em 18/10/2016).

Ramos, J. (1999). Using tf-idf to determine word relevance in document queries.

Refaeilzadeh, P., Tang, L., e Liu, H. (2009). Cross-Validation. In Liu, L. e Özsu, M. T., editors, *Encyclopedia of Database Systems*, páginas 532–538. Springer US.

Roberts, D. e Johnson, R. (1996). Evolving Frameworks: A Pattern Language for Developing Object-Oriented Frameworks. In *Proceedings of the Third Conference on Pattern Languages and Programming*, volume 3.

Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60(5):503–520.

Rudd, J. e Isensee, S. (1994). Twenty-two tips for a happier, healthier prototype. *interactions*, 1:35–40.

Scholkopf, B., Smola, A., e Müller, K.-R. (1999). Kernel principal component analysis. In *ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING*, páginas 327–352. MIT Press.

Selvam, B. e Abirami, S. (2013). A survey on opinion mining framework.

Silva, R. M., Yamakami, A., e Almeida, T. A. (2012). An analysis of machine learning methods for spam host detection. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, páginas 227–232.

Szekely, P. A. (1994). User Interface Prototyping: Tools and Techniques. In *ICSE Workshop on SE-HCI*, páginas 76–92.

Tan, P.-N., Steinbach, M., e Kumar, V. (2006). *Introduction to data mining.* Pearson Addison Wesley, Boston, San Francisco. Table des matières à l'adresse suivante http://www.loc.gov/catdir/toc/ecip0510/2005008721.html.

Tsytsarau, M. e Palpanas, T. (2012). Survey on mining subjective data on the web. *Data Min. Knowl. Discov.*, 24(3):478–514.

Valsamidis, S., Theodosiou, T., Kazanidis, I., e Nikolaidis, M. (2013). A framework for opinion mining in blogs for agriculture. *Procedia Technology*, 8:264 – 274.

Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z.-H., Steinbach, M., Hand, D. J., e Steinberg, D. (2014). Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37.

Anexos

A Manager

A.1 Implementação da rotina responsável por recuperar os ids dos *tweets* de interesse no site do *Twitter*

```
package br.ufma.leserc.util;
import static io.github.seleniumquery.SeleniumQuery.$;
import io.github.seleniumquery.SeleniumQueryObject;

import java.util.ArrayList;
import java.util.List;

import io.github.seleniumquery.browser.driver.SeleniumQueryDriver;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebElement;

/**
 *
 * @author Gilberto
 */
public class SearchTweets {

   /**
     *
     * @param Q - term of search
     * @param city - city of selected
```

```
     * @param since - date start
     * @param unitl - data end
     * @param city - state of city
     * @return the list of ids for tweets interest
     *
     */
    public static List<Long> request(String Q, String city, String since, String
        unitl, String state) {
        SeleniumQueryDriver seleniumQueryDriver = $.driver();
        seleniumQueryDriver.useFirefox();

        $.url("https://twitter.com/?lang=pt");
        $("//div[@class='username field']").children().val("*******");
        $("//div[@class='password flex-table-form']").children().val("********");
        $("//td[@class='flex-table-secondary']").children().click();
        $.url("https://twitter.com/search?q="+Q+"%20lang%3Apt%20near%3A%22"+city
        +"%22%20within%3A100km%20since%3A"+since+"%20until%3A"+unitl
        +"&src=typd&vertical=default&f=tweets");

        JavascriptExecutor javascriptExecutor = ((JavascriptExecutor)
            seleniumQueryDriver.get());
        while (!$("//span[@class='Icon Icon--large
            Icon--logo']").get(0).isDisplayed()) {
            javascriptExecutor.executeScript("window.scrollTo(0,
                document.body.scrollHeight)");
        }

        List<Long> listId = new ArrayList<Long>();
        for (WebElement webElement : $("//ol[@class='stream-items
            js-navigable-stream']").children("li").children("div")) {
            if(webElement.getText().contains(state)) {
                listId.add(Long.parseLong(webElement.getAttribute("data-item-id")));
                System.out.println(webElement.getAttribute("data-item-id"));
            }
        }
```

```
        $.quit();
        return listId;
    }
}
```

A.2 Implementação da rotina responsável por recuperar os *tweets* de interesse na API do *Twitter*

```
package br.ufma.leserc.util;

import java.text.SimpleDateFormat;

import java.util.ArrayList;
import java.util.List;

import twitter4j.RateLimitStatus;
import twitter4j.Status;
import twitter4j.TwitterException;

/**
 *
 * @author Gilberto
 */
public class SearchTweetsInApi extends Base {

    /**
     *
     * @param listId - List with ids of the tweets
     * @return the list of status by tweets
     *
     */
    public List<Status> getTweets(List<Long> listId) {
        List<Status> statusList = new ArrayList<>();
        try {
            for (Long id : listId) {
```

```
                RateLimitStatus rateLimitStatus =
                    RateLimitUtil.getRateLimit("/statuses/show/:id");
                if(rateLimitStatus.getRemaining() > 0) {
                   Status status = twitter.showStatus(id);
                     statusList.add(status);
                } else {
                   try {
                      Thread.sleep((rateLimitStatus.getSecondsUntilReset()+2)*1000);//miliseconds
                          to a second
                     } catch (InterruptedException e) {
                         // TODO Auto-generated catch block
                         e.printStackTrace();
                     }
                }
            }
        } catch (TwitterException te) {
            System.out.println("Failed to search tweets: " + te.getMessage());
            return null;
        }
        return statusList;
    }
}
```

A.3 Implementação da rotina responsável pelo pré-processamento dos *tweets*

```
package br.ufma.leserc.util;

import java.text.Normalizer;
import java.text.Normalizer.Form;

/**
 *
 * @author Gilberto
 */
```

```
public class TextUtil {

   /*Remaining omitted code*/

    /**
     *
     * @param tweet - tweet with text for proccess
     * @return the text of the tweet processed
     *
     */
    public static String removeInvalidTokens(String tweet) {
      /*remove accentuation*/
      String processedTweet = Normalizer.normalize(tweet, Form.NFKD)
        .replaceAll("\\p{InCombiningDiacriticalMarks}+", "");
      /*lower case*/
      processedTweet = processedTweet.toLowerCase();
      /*remove mentions, hashtags, urls, emails and special characters*/
      StringBuilder regexToRemove = new StringBuilder();
      String regexMentions = "^@[^ ]*|[ ]@[^ ]*";
      String regexUrls = "^http://[^ ]*|[ ]http://[^ ]*|^ftp://[^ ]*|...";
      String regexMail = "^@[A-Za-z0-9\\._-]+@ ... ";
      String regexSpecialCharacters = "[!-^@\\\'\\ ... ]";
      regexToRemove.append(regexMentions)
         .append(regexUrls)
         .append(regexMail)
         .append(regexSpecialCharacters);
      processedTweet.replaceAll(regexToRemove.toString(), "");
      processedTweet.trim();
      processedTweet.replaceAll(" +", " ");
      processedTweet.replaceAll("\\r|\\n", " ");
      return processedTweet;
    }
}
```

A.4 Implementação da rotina responsável por salvar os *tweets* e suas cópias na base de dados

```
package br.ufma.leserc.util;

/*Imports omitted*/

/**
 *
 * @author Gilberto
 */
public class TweetDetalhesController extends FormController {

   /*Remaining omitted code*/

    /**
     *
     * @param Q - term of search
     * @param city - city selected
     * @param since - date start
     * @param unitl - data end
     * @param city - state of the city
     * @param type - type of disease
     *
     *
     */
    public void save(String Q, String city, String since, String until, String type)
        {
        List<Long> listId = SearchTweets.request(Q, city, since, until);
        SearchTweetsInApi searchTweetsInApi = new SearchTweetsInApi();
        List<Status> statusList = searchTweetsInApi.getTweets(listId);
        for (Status status : statusList) {
            Corpus aux = (Corpus) dao.getSession().createCriteria(Corpus.class)
                    .setProjection(Projections.projectionList()
                    .add(Projections.property("this.id").as("id")))
                    .add(Restrictions.eq("this.id", status.getId()))
```

```
                .setResultTransformer(new
                    AliasToBeanResultTransformer(Corpus.class)).uniqueResult();
        // checks if there in the database
           if (aux == null) {
               try {
                   Long id = status.getId();processedTweet
                   String screenName = status.getUser().getScreenName();
                   String text = status.getText();
                   String processedText =
                       TextUtil.removeInvalidTokens(status.getText());
                   Point point = new
                       GeometryUtils().from(status.getGeoLocation().getLatitude(),
                       status.getGeoLocation().getLongitude()).convertTo(Point.class);
                   Date createdAt = status.getCreatedAt();
                   corpusbo.save(new Corpus(id, screenName, text, processedText,
                       point, createdAt, type));
               } catch (Exception ex) {
                   Logger.getLogger(TweetDetalhes.class.getName()).log(Level.SEVERE,
                       null, ex);
                   JOptionPane.showMessageDialog (null, ex.getMessage(), "Manager",
                       JOptionPane.ERROR_MESSAGE);
               }
           }
       }
       JOptionPane.showMessageDialog (null, "Successfully saved records",
           "Manager", JOptionPane.INFORMATION_MESSAGE);
   }
}
```

A.5 Manager dependências

As bibliotecas necessárias ao funcionamento do *Manager* estão disponíveis na Tabela 6.1.

Tabela 6.1: Tabela com as dependências necessárias ao projeto.

Biblioteca	Download link
antlr-2.7.6	http://central.maven.org/maven2/antlr/antlr/2.7.6/antlr-2.7.6.jar
BeanInfo	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/BeanInfo.jar
clientos	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/clientos.jar
commonos	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/commonos.jar
commons-lang3-3.1	http://central.maven.org/maven2/org/apache/commons/commons-lang3/3.1/commons-lang3-3.1.jar
commons-logging-1.1.3	http://central.maven.org/maven2/commons-logging/commons-logging/1.1.3/commons-logging-1.1.3.jar
cssparser-0.9.12	http://central.maven.org/maven2/net/sourceforge/cssparser/cssparser/0.9.12/cssparser-0.9.12.jar
dom4j-1.6.1	http://central.maven.org/maven2/dom4j/dom4j/1.6.1/dom4j-1.6.1.jar
gt-main-10.8	http://download.osgeo.org/webdav/geotools/org/geotools/gt-main/10.8/gt-main-10.8.jar
hessian-3.1.1	http://maven.onehippo.com/content/groups/maven/hessian/jars/hessian-3.1.1.jar
hibernate-commons-annotations-3.2.0.Final	http://central.maven.org/maven2/org/hibernate/common/hibernate-commons-annotations/3.2.0.Final/hibernate-commons-annotations-3.2.0.Final.jar
hibernate-core-4.0.0.Final	http://central.maven.org/maven2/org/hibernate/hibernate-core/4.0.0.Final/hibernate-core-4.0.0.Final.jar
hibernate-entitymanager-4.0.0.Final	http://central.maven.org/maven2/org/hibernate/hibernate-entitymanager/4.0.0.Final/hibernate-entitymanager-4.0.0.Final.jar
hibernate-jpa-2.0-api-1.0.1.Final	http://central.maven.org/maven2/org/hibernate/javax/persistence/hibernate-jpa-2.0-api/1.0.1.Final/hibernate-jpa-2.0-api-1.0.1.Final.jar
hibernate-search-4.0.0.Final	http://central.maven.org/maven2/org/hibernate/hibernate-search-orm/4.0.0.Final/hibernate-search-orm-4.0.0.Final.jar
hibernate-spatial-4.0	http://www.hibernatespatial.org/repository/org/hibernate/hibernate-spatial/4.0/hibernate-spatial-4.0.jar
itext-1.4.8	http://central.maven.org/maven2/com/lowagie/itext/1.4.8/itext-1.4.8.jar
iText-2.1.7	http://central.maven.org/maven2/com/lowagie/itext/2.1.7/itext-2.1.7.jar
iText-rtf-2.1.7	http://central.maven.org/maven2/com/lowagie/itext-rtf/2.1.7/itext-rtf-2.1.7.jar
jandex-1.0.3.Final	http://central.maven.org/maven2/org/jboss/jandex/1.0.3.Final/jandex-1.0.3.Final.jar
javassist-3.18.1-GA	http://central.maven.org/maven2/org/javassist/javassist/3.18.1-GA/javassist-3.18.1-GA.jar
jboss-logging-3.2.1.Final	http://central.maven.org/maven2/org/jboss/logging/jboss-logging/3.2.1.Final/jboss-logging-3.2.1.Final.jar
jcalendar	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/jcalendar.jar
jnlp	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/jnlp.jar
jnlp-servlet	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/jnlp-servlet.jar
jta-1.1	http://central.maven.org/maven2/javax/transaction/jta/1.1/jta-1.1.jar
jts-1.13	http://central.maven.org/maven2/com/vividsolutions/jts/1.13/jts-1.13.jar
lucene-core-4.0.0	http://central.maven.org/maven2/org/apache/lucene/lucene-core/4.0.0/lucene-core-4.0.0.jar
phantomjsdriver-1.1.0	http://central.maven.org/maven2/com/github/detro/ghostdriver/phantomjsdriver/1.1.0/phantomjsdriver-1.1.0.jar
poi-2.0-final-20040126	http://central.maven.org/maven2/poi/poi/2.0-final-20040126/poi-2.0-final-20040126.jar
pooler	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/pooler.jar
postgis-jdbc-1.5.2	http://www.hibernatespatial.org/repository/org/postgis/postgis-jdbc/1.5.2/postgis-jdbc-1.5.2.jar
postgis-stubs-1.3.3	http://central.maven.org/maven2/org/postgis/postgis-stubs/1.3.3/postgis-stubs-1.3.3.jar
postgresql-9.1-901-1.jdbc4	http://central.maven.org/maven2/postgresql/postgresql/9.1-901-1.jdbc4/postgresql-9.1-901-1.jdbc4.jar
selenium-firefox-driver-2.46.0	http://central.maven.org/maven2/org/seleniumhq/selenium/selenium-firefox-driver/2.46.0/selenium-firefox-driver-2.46.0.jar
selenium-htmlunit-driver-2.46.0	http://central.maven.org/maven2/org/seleniumhq/selenium/selenium-htmlunit-driver/2.46.0/selenium-htmlunit-driver-2.46.0.jar
selenium-java-2.46.0	http://central.maven.org/maven2/org/seleniumhq/selenium/selenium-java/2.46.0/selenium-java-2.46.0.jar
seleniumquery-0.9.0	http://central.maven.org/maven2/io/github/seleniumquery/seleniumquery/0.9.0/seleniumquery-0.9.0.jar
selenium-remote-driver-2.46.0	http://central.maven.org/maven2/org/seleniumhq/selenium/selenium-remote-driver/2.46.0/selenium-remote-driver-2.46.0.jar
selenium-server-standalone-2.46.0	http://central.maven.org/maven2/org/seleniumhq/selenium/selenium-server/2.46.0/selenium-server-2.46.0.jar
serveros	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/serveros.jar
slf4j-api-1.7.5	http://central.maven.org/maven2/org/slf4j/slf4j-api/1.7.5/slf4j-api-1.7.5.jar
slf4j-simple-1.7.5	http://central.maven.org/maven2/org/slf4j/slf4j-simple/1.7.5/slf4j-simple-1.7.5.jar
srcclientos	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/srcclientos.jar
srccommonos	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/srccommonos.jar
srcserveros	https://openswing.googlecode.com/svn/OpenSwing/2.4.2/srcserveros.jar
twitter4j-async-4.0.4	http://central.maven.org/maven2/org/twitter4j/twitter4j-async/4.0.4/twitter4j-async-4.0.4.jar
twitter4j-core-4.0.4	http://central.maven.org/maven2/org/twitter4j/twitter4j-core/4.0.4/twitter4j-core-4.0.4.jar
twitter4j-stream-4.0.4	http://central.maven.org/maven2/org/twitter4j/twitter4j-stream/4.0.4/twitter4j-stream-4.0.4.jar

B Light Client

B.1 Implementação da rotina responsável pela análise dos *tweets* processados

```
package br.ufma.leserc.weka;


import java.io.File;

import java.io.FileInputStream;

import java.io.InputStream;

import java.io.ObjectInputStream;

import java.net.URL;

import java.util.logging.Level;
```

```
import java.util.logging.Logger;
import weka.classifiers.Classifier;
import weka.core.Instances;
import weka.core.SerializationHelper;
import weka.core.tokenizers.WordTokenizer;
import weka.experiment.InstanceQuery;
import weka.filters.Filter;
import weka.filters.unsupervised.attribute.Add;
import weka.filters.unsupervised.attribute.NominalToString;
import weka.filters.unsupervised.attribute.StringToWordVector;
import weka.filters.unsupervised.attribute.Reorder;

/**
 *
 * @author Gilberto
 */
public class MyClassifier {

    //Instances instances;
    public Instances stringToWordVector(Instances instances) {
        StringToWordVector stringToWordVector = null;
        // Set the tokenizer
        WordTokenizer wordTokenizer = new WordTokenizer();
        wordTokenizer.setDelimiters("\r\n\t.,;:\"\"()?!");

        try {
            stringToWordVector = new StringToWordVector();
            stringToWordVector.setInputFormat(instances);
            stringToWordVector.setOptions(weka.core.Utils.splitOptions(" -R
                first-last -W 1000 -prune-rate -1.0 -T -I -N 0 -stemmer
                weka.core.stemmers.NullStemmer -M 1 -tokenizer
                \"weka.core.tokenizers.WordTokenizer -delimiters \\\"
                \\\\r\\\\n\\\\t.,;:\\\\\\\'\\\\\\\"()?!\\\"\""));
            instances = Filter.useFilter(instances, stringToWordVector);
        } catch (Exception ex) {
```

```
            Logger.getLogger(MyClassifier.class.getName()).log(Level.SEVERE, null,
                ex);
        }
        return instances;
    }

    public Instances reorder(Instances instances) {
        Reorder reorder = new Reorder();
        try {
            reorder.setInputFormat(instances);
            reorder.setOptions(weka.core.Utils.splitOptions(" -R last-first"));
            instances = Filter.useFilter(instances, reorder);
            instances.setRelationName("database_tweets");
        } catch (Exception ex) {
            Logger.getLogger(MyClassifier.class.getName()).log(Level.SEVERE, null,
                ex);
        }
        return instances;
    }

    public Instances loadDataBase() {
        Instances instances = null;
        InstanceQuery query;
        try {
            URL url;
            url = getClass().getResource("DatabaseUtils.props");
            query = new InstanceQuery();
            query.initialize(new File(url.getPath()));
            query.setDatabaseURL("jdbc:postgresql://localhost/rof");
            query.setUsername("********");
            query.setPassword("********");
            String sql = "SELECT c.description_processed AS text FROM corpus c ORDER
                BY c.id";
            query.setQuery(sql);
            query.close();
            instances = query.retrieveInstances();
```

```
        } catch (Exception ex) {
            Logger.getLogger(MyClassifier.class.getName()).log(Level.SEVERE, null,
                ex);
        }
        return instances;
    }

    public Instances nominalToString(Instances instances) {
        NominalToString nominalToString = new NominalToString();
        try {
            nominalToString.setInputFormat(instances);
            instances = Filter.useFilter(instances, nominalToString);
        } catch (Exception ex) {
            Logger.getLogger(MyClassifier.class.getName()).log(Level.SEVERE, null,
                ex);
        }
        return instances;
    }

    public Instances addClass(Instances instances) {
        Add add = new Add();
        try {
            add.setInputFormat(instances);
            add.setOptions(weka.core.Utils.splitOptions(" -T NOM -N class -L
                \"pos,neg\" -C last"));
            instances = Filter.useFilter(instances, add);
            instances.setClassIndex(1);
        } catch (Exception ex) {
            Logger.getLogger(MyClassifier.class.getName()).log(Level.SEVERE, null,
                ex);
        }
        return instances;
    }

    public Classifier getClassifier(String name) {
        Classifier classifier = null;
```

```
        InputStream inputStream;
        inputStream = getClass().getResourceAsStream(name+".model");
        try {
                classifier = (Classifier) SerializationHelper.read(inputStream);
        } catch (Exception ex) {
            Logger.getLogger(MyClassifier.class.getName()).log(Level.SEVERE, null,
                ex);
        }
        return classifier;
    }

    public Classifier loadModel(File path, String name) throws Exception {
        Classifier classifier;
        FileInputStream fis = new FileInputStream(path + name + ".model");
        ObjectInputStream ois = new ObjectInputStream(fis);
        classifier = (Classifier) ois.readObject();
        ois.close();
        return classifier;
    }
}
```

B.2 Light Client dependências

As bibliotecas necessárias ao funcionamento do *Light Client* estão disponíveis na Tabela 6.2.

C Links para download

Abaixo são dispostos os *links* necessários para o download de todas os softwares e frameworks utilizados no trabalho proposto.

C.1 Softwares necessários para a configuração do ambiente de desenvolvimento

Java Development Kit (JDK) 8/64bits - http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Java EE IDE for Web Developers/Mars Release (4.5.0)/64bits - https://eclipse.org/downloads/

Apache Tomcat 8.0.24/64 - https://tomcat.apache.org/download-80.cgi

Apache Tomcat 7.0.63/64 - https://tomcat.apache.org/download-70.cgi

PostgreSQL 9.3/64bits - http://www.postgresql.org/download/

PostGIS 2.1/64bits - http://postgis.net/install/

GeoServer 2.7.1/ x86_64 bits - http://geoserver.org/download/

WEKA, 3.7.11/64 bits - http://www.cs.waikato.ac.nz/ml/weka/downloading.html

Vraptor 4 - http://www.vraptor.org/pt/download/

OpenSwing - http://sourceforge.net/projects/oswing/files/oswing/

C.2 Fontes do ROF

Tabela 6.2: Tabela com as dependências necessárias ao projeto.

Biblioteca	Download link
antlr-2.7.7	http://central.maven.org/maven2/antlr/antlr/2.7.7/antlr-2.7.7.jar
cdi-api-1.1	http://central.maven.org/maven2/javax/enterprise/cdi-api/1.1/cdi-api-1.1.jar
classmate-0.5.4	http://central.maven.org/maven2/com/fasterxml/classmate/0.5.4/classmate-0.5.4.jar
commons-beanutils-1.7.0	http://central.maven.org/maven2/commons-beanutils/commons-beanutils/1.7.0/commons-beanutils-1.7.0.jar
commons-collections-3.1	http://central.maven.org/maven2/commons-collections/commons-collections/3.1/commons-collections-3.1.jar
commons-lang-2.3	http://central.maven.org/maven2/commons-lang/commons-lang/2.3/commons-lang-2.3.jar
displaytag-1.2	http://central.maven.org/maven2/displaytag/displaytag/1.2/displaytag-1.2.jar
dom4j-1.6.1	http://central.maven.org/maven2/dom4j/dom4j/1.6.1/dom4j-1.6.1.jar
gson-2.2.4	http://central.maven.org/maven2/com/google/code/gson/gson/2.2.4/gson-2.2.4.jar
guava-15.0	http://central.maven.org/maven2/com/google/guava/guava/15.0/guava-15.0.jar
hibernate-commons-annotations-4.0.1.Final	http://central.maven.org/maven2/org/hibernate/common/hibernate-commons-annotations/4.0.1.Final/hibernate-commons-annotations-4.0.1.Final.jar
hibernate-core-4.0.0.Final	http://central.maven.org/maven2/org/hibernate/hibernate-core/4.0.0.Final/hibernate-core-4.0.0.Final.jar
hibernate-entitymanager-4.0.0.Final	http://central.maven.org/maven2/org/hibernate/hibernate-entitymanager/4.0.0.Final/hibernate-entitymanager-4.0.0.Final.jar
hibernate-jpa-2.0-api-1.0.1.Final	http://central.maven.org/maven2/org/hibernate/javax/persistence/hibernate-jpa-2.0-api/1.0.1.Final/hibernate-jpa-2.0-api-1.0.1.Final.jar
hibernate-spatial-4.0	http://www.hibernatespatial.org/repository/org/hibernate/hibernate-spatial/4.0/hibernate-spatial-4.0.jar
hibernate-validator-5.1.1.Final	http://central.maven.org/maven2/org/hibernate/hibernate-validator/5.1.1.Final/hibernate-validator-5.1.1.Final.jar
hibernate-validator-cdi-5.1.1.Final	http://central.maven.org/maven2/org/hibernate/hibernate-validator-cdi/5.1.1.Final/hibernate-validator-cdi-5.1.1.Final.jar
iogi-1.0.0	http://central.maven.org/maven2/br/com/caelum/iogi/1.0.0/iogi-1.0.0.jar
itext-1.3	http://central.maven.org/maven2/com/lowagie/itext/1.3/itext-1.3.jar
jandex-1.0.3.Final	http://central.maven.org/maven2/org/jboss/jandex/1.0.3.Final/jandex-1.0.3.Final.jar
javassist-3.18.1-GA	http://central.maven.org/maven2/org/javassist/javassist/3.18.1-GA/javassist-3.18.1-GA.jar
javax.annotation-api-1.2	http://central.maven.org/maven2/javax/annotation/javax.annotation-api/1.2/javax.annotation-api-1.2.jar
javax.ejb-api-3.2	http://central.maven.org/maven2/javax/ejb/javax.ejb-api/3.2-b02/javax.ejb-api-3.2-b02.jar
javax.inject-1	http://central.maven.org/maven2/javax/inject/javax.inject/1/javax.inject-1.jar
javax.interceptor-api-1.2	http://mvnrepository.com/artifact/javax.interceptor/javax.interceptor-api/1.2
javax.transaction-api-1.2	http://central.maven.org/maven2/javax/transaction/javax.transaction-api/1.2/javax.transaction-api-1.2.jar
jboss-annotations-api_1.2_spec-1.0.0.Alpha1	http://central.maven.org/maven2/org/jboss/spec/javax/annotation/jboss-annotations-api_1.2_spec/1.0.0.Alpha1/jboss-annotations-api_1.2_spec-1.0.0.Alpha1.jar
jboss-classfilewriter-1.0.4.Fina	http://central.maven.org/maven2/org/jboss/classfilewriter/jboss-classfilewriter/1.0.4.Final/jboss-classfilewriter-1.0.4.Final.jar
jboss-el-api_3.0_spec-1.0.0.Alpha1	http://central.maven.org/maven2/org/jboss/spec/javax/el/jboss-el-api_3.0_spec/1.0.0.Alpha1/jboss-el-api_3.0_spec-1.0.0.Alpha1.jar
jboss-logging-3.1.0.CR2	http://central.maven.org/maven2/org/jboss/logging/jboss-logging/3.1.0.CR2/jboss-logging-3.1.0.CR2.jar
jboss-transaction-api_1.1_spec-1.0.0.Final	http://central.maven.org/maven2/org/jboss/spec/javax/transaction/jboss-transaction-api_1.1_spec/1.0.0.Final/jboss-transaction-api_1.1_spec-1.0.0.Final.jar
jcl104-over-slf4j-1.4.2	http://central.maven.org/maven2/org/slf4j/jcl104-over-slf4j/1.4.2/jcl104-over-slf4j-1.4.2.jar
jstl-1.2	http://central.maven.org/maven2/javax/servlet/jstl/1.2/jstl-1.2.jar
jts-1.13	http://central.maven.org/maven2/com/vividsolutions/jts/1.13/jts-1.13.jar
LibSVM-1.0.6	http://central.maven.org/maven2/nz/ac/waikato/cms/weka/LibSVM/1.0.6/LibSVM-1.0.6.jar
libsvm-3.17	http://central.maven.org/maven2/tw/edu/ntu/csie/libsvm/3.17/libsvm-3.17.jar
mirror-1.6.1	http://central.maven.org/maven2/net/vidageek/mirror/1.6.1/mirror-1.6.1.jar
paranamer-2.7	http://central.maven.org/maven2/com/thoughtworks/paranamer/paranamer/2.7/paranamer-2.7.jar
postgis-jdbc-1.5.2	http://www.hibernatespatial.org/repository/org/postgis/postgis-jdbc/1.5.2/postgis-jdbc-1.5.2.jar
postgresql-9.1-901-1.jdbc4	http://central.maven.org/maven2/postgresql/postgresql/9.1-901-1.jdbc4/postgresql-9.1-901-1.jdbc4.jar
log4j-1.2.13	http://central.maven.org/maven2/log4j/log4j/1.2.13/log4j-1.2.13.jar
javassist-3.12.1-GA	http://central.maven.org/maven2/org/javassist/javassist/3.12.1-GA/javassist-3.12.1-GA.jar
slf4j-api-1.4.2	http://central.maven.org/maven2/org/slf4j/slf4j-api/1.4.2/slf4j-api-1.4.2.jar
slf4j-log4j12-1.4.2	http://central.maven.org/maven2/org/slf4j/slf4j-log4j12/1.4.2/slf4j-log4j12-1.4.2.jar
validation-api-1.1.0.Final	http://central.maven.org/maven2/javax/validation/validation-api/1.1.0.Final/validation-api-1.1.0.Final.jar
vraptor-4.1.4	http://central.maven.org/maven2/br/com/caelum/vraptor/4.1.4/vraptor-4.1.4.jar
vraptor-jpa-4.0.3	http://central.maven.org/maven2/br/com/caelum/vraptor/vraptor-jpa/4.0.3/vraptor-jpa-4.0.3.jar
Weka-dev-3.7.11	http://central.maven.org/maven2/nz/ac/waikato/cms/weka/weka-dev/3.7.11/weka-dev-3.7.11.jar
weld-api-2.1.Final	http://central.maven.org/maven2/org/jboss/weld/weld-api/2.1.Final/weld-api-2.1.Final.jar
weld-core-2.1.2.Final	http://central.maven.org/maven2/org/jboss/weld/weld-core/2.1.2.Final/weld-core-2.1.2.Final.jar
weld-core-impl-2.1.2.Final	http://central.maven.org/maven2/org/jboss/weld/weld-core-impl/2.1.2.Final/weld-core-impl-2.1.2.Final.jar
weld-servlet-2.1.2.Final	http://central.maven.org/maven2/org/jboss/weld/servlet/weld-servlet-core/2.1.2.Final/weld-servlet-core-2.1.2.Final.jar
weld-spi-2.1.Final	http://central.maven.org/maven2/org/jboss/weld/weld-spi/2.1.Final/weld-spi-2.1.Final.jar
xml-apis-1.0.b2	http://central.maven.org/maven2/xml-apis/xml-apis/1.0.b2/xml-apis-1.0.b2.jar
xmlpull-1.1.3.1	http://central.maven.org/maven2/xmlpull/xmlpull/1.1.3.1/xmlpull-1.1.3.1.jar
xpp3_min-1.1.4c	http://central.maven.org/maven2/xpp3/xpp3_min/1.1.4c/xpp3_min-1.1.4c.jar
xstream-1.4.7	http://central.maven.org/maven2/com/thoughtworks/xstream/xstream/1.4.7/xstream-1.4.7.jar