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

Análise Preditiva Espaço-Temporal das Tendências Eleitorais Brasileiras com Base nos Dados do Twitter

Bruno Justino Garcia Praciano

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia de Computação

Orientador
Prof. Dr. -Ing João Paulo Carvalho Lustosa da Costa

Coorientador
MSc. João Paulo Abreu Maranhão

Brasília
2018

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

Análise Preditiva Espaço-Temporal das Tendências Eleitorais Brasileiras com Base nos Dados do Twitter

Bruno Justino Garcia Praciano

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia de Computação

Prof. Dr. -Ing João Paulo Carvalho Lustosa da Costa (Orientador)
ENE/UnB

Prof. Dr. Rafael Timóteo de Sousa Júnior
ENE/UnB

Dr. -Ing Ricardo Kehrlee Miranda
ENM/UnB

Prof. Dr. José Edil Guimarães de Medeiros
Coordenador do Curso de Engenharia de Computação

Brasília, 13 de novembro de 2018

Dedicatória

Dedico este trabalho a todos, que me incentivaram todos os dias e ofereceram apoio nos momentos críticos.

Agradecimentos

Agradeço primeiramente a Deus por ter me destinado pessoas incríveis para estarem ao meu lado durante esse momento árduo que é a escrita do trabalho de conclusão de curso.

Agradeço aos meus pais, Alessandra e Flávio por terem me dado o dom da vida, pelo incentivo e motivação.

Em especial à minha avó Lenita Justino, por toda formação intelectual que foi investida em mim e pela formação do meu caráter e da minha motivação que possibilitou a minha entrada na Universidade de Brasília.

Ao meu irmão, Victor Hugo, pela amizade e boa convivência.

Agradeço imensamente a minha namorada Amanda, por entender a minha ausência em diversos momentos durante a realização deste trabalho e também pelo seu incentivo, paciência, compreensão, companheirismo e carinho.

Agradeço ao meu orientador, João Paulo Lustosa, por ter me incentivado a sempre buscar a excelência e pensar fora da caixa e também pela paciência, amizade, incentivo, várias conversas produtivas que já tivemos e por todo suporte que me foi dado durante a realização deste trabalho, que sem o qual não teria sido possível a sua conclusão. E por todas as oportunidade que me deu dentro do LASP, inclusive na fundação do capítulo IEEE VTS, sendo o primeiro do Brasil.

Aos meus amigos Lucas Maciel, Yan Trindade, Cássio Lisboa, Victor Guedes, Victor Duarte e Thiago Figueiredo por todos os debates e sugestões em temas acadêmicos e temas aleatórios que me ajudaram nos momentos de distração.

Aos meus amigos e colegas de faculdade, Iure Brandão, Gabriel Pinheiro, Yuri Castro, Juliano Prettz e Bruno Cordeiro pelas inúmeras sugestões e parcerias ao longo do curso, em especial àqueles que venceram juntos comigo Organização e Arquitetura de Computadores e Software Básico.

Ao Ricardo Kehrle João Paulo Maranhão por me ajudarem imensamente nos temas acadêmicos.

Agradeço também ao Fábio Lúcio por ter me dado oportunidade de trabalhar no Laboratório de Tomada de Decisões e por ter sido um dos incentivadores a realizar a

mudança de curso.

A todos os professores que se dedicaram em minha formação, em especial aos professores Rafael Timóteo, Luisiane Santana, Felipe Lopes, Teófilo e outros.

Resumo

A análise de sentimentos é uma área de pesquisa que utiliza técnicas de mineração de textos com o objetivo de entender o comportamento das pessoas através do que foi escrito ou falado. Tais técnicas têm sido utilizadas para reconhecer padrões e extrair informações de grandes bases de dados que contenham textos. Este trabalho tem como objetivo propor um framework para a identificação de padrões em mensagens de texto no Twitter os quais, de algum modo, possam prever o resultado das eleições presidenciais brasileiras de maneira eficiente. O framework utiliza ferramentas de processamento de linguagem natural, que incluem o pré-processamento dos dados de entrada e a aplicação de algoritmos de aprendizado de máquina para classificar novos textos tendo por base textos previamente classificados. Os resultados do framework foram validados por meio da comparação com as informações disponíveis na base de dados do Tribunal Superior Eleitoral (TSE). A solução baseada no algoritmo de máquina de vetores de suporte, do inglês Support Vector Machine (SVM), apresentou o melhor desempenho.

Palavras-chave: Big Data, Aprendizado de Máquina Supervisionado, Análise de Sentimentos, Máquina de Vetor de Suporte

Abstract

The sentiment analysis is a research area which applies text mining techniques in order to understand the behavior of people. Such techniques have been used to recognize patterns and extract information from large databases. This work proposes a framework for the identification of patterns in text messages on Twitter social network, allowing us to predict the results of the Brazilian presidential elections in an efficient way. The framework uses natural language processing tools, including the input data preprocessing and the application of machine learning algorithms in order to classify new texts based on previously classified data. The results of the framework are validated through a comparison with the data available on the Superior Electoral Court (TSE) database. According to our results, the solution based on the Support Vector Machine (SVM) algorithm presents the best performance.

Keywords: Big Data, Supervised Machine Learning, Sentiment Analysis, Support Vector Machine

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AM Aprendizado de Máquina.

API Aplication Programming Interface.

IBGE Instituto Brasileiro de Geografia e Estatística.

PLN Processamento de Linguagem Natural.

SVM Support Vector Machine.

TM Text Mining.

TSE Tribunal Superior Eleitoral.

Capítulo 1

Introdução

A popularização da Internet tem revolucionado as sociedades com o passar do tempos, pois agora é possível conectar-se a diversas pessoas, ter acesso à informação de forma rápida e realizar grande parte das atividades em tempo real, e esse sucesso é devido ao seu baixo custo em relação aos veículos tradicionais de mídia [4].

A troca de dados tem acontecido maneira intensa. As redes sociais são as responsáveis por esse crescimento expressivo, pois as pessoas podem trocar ideias e opiniões acerca de determinado assunto e, com isso, facilitar o acesso de todos. As redes sociais já fazem parte da vida de várias pessoas, o que proporcionou uma modificação das relações interpessoais, gerando grande quantidade de dados de fácil e livre acesso. [5].

Por meio das redes sociais, é possível comunicar-se com pessoas de diversas nacionalidades e características. Em razão de sua grande amplitude, o volume de dados gerados é praticamente imensurável, tornando o ambiente atrativo para a aplicação de técnicas de aprendizado de máquina e outros tipos de análises.

1.1 Motivação

De acordo com o IBGE, no Brasil mais de 116 milhões de pessoas têm acesso à Internet, ou seja, grande parte da população está expressando suas ideias de forma livre nas redes sociais [6]. Como as eleições constituem-se em eventos muito importantes em qualquer democracia, a análise de sentimentos em textos provenientes de redes sociais se torna cada vez mais atrativa. O Brasil ocupa a 4ª posição no ranking mundial de países com a quantidade de pessoas com acesso à Internet [7].

Na Figura 1.1 é possível visualizar que grande parte do conteúdo disponível na Internet está em inglês. Consequentemente, existem dicionários léxicos para atividades de mineração de texto (TM), como é o caso do WordNet [8], uma das maiores bases de dados do mundo para essa atividade.

As pesquisas utilizando TM com o idioma português ainda são recentes e poucos exploradas, pois a maioria das ferramentas são desenvolvidas para atender ao idioma inglês. Entretanto, a análise de sentimentos em comentários de redes sociais utilizando o idioma português pode ser útil em diversas áreas, tais como em vendas de produtos, avaliações de qualidade de estabelecimentos comerciais e predições de eleições, sendo estas últimas o objetivo principal deste trabalho.

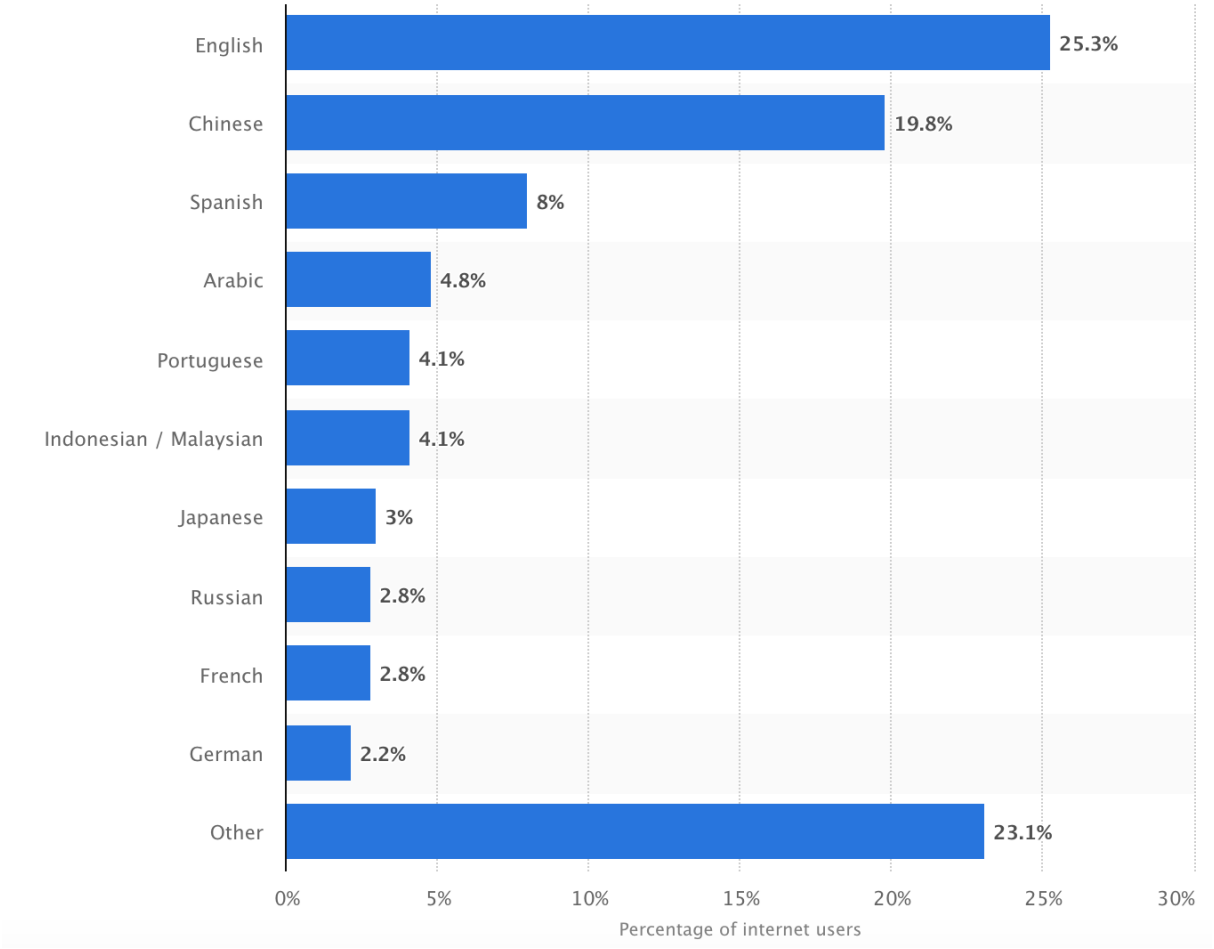


Figura 1.1: Idiomas utilizados em conteúdos disponíveis na Internet [1].

1.2 Problemas

Existem na Internet várias redes sociais, cada uma com um diferente foco. Por exemplo, o *Twitter* é uma rede social com o intuito de formação de opinião, pois grande parte dos usuários a utilizam para compartilhar texto pequenos de até 140 caracteres. Além disso, o *Twitter* apresenta uma API aberta, porém com limitações em relação ao número de requisições e ao período para realização de buscas, que atualmente é de 14 dias [9].

A rede social mais utilizada no Brasil é o *Facebook*, que tem mais de 120 milhões de usuários ativos no Brasil [10]. Entretanto, após os escândalos das eleições norte-americanas que a envolveram [11], foram adotadas inúmeras medidas de segurança para evitar a extração de dados da Atualmente, para se utilizar a API do *Facebook* é necessário ter aprovação prévia da empresa, sendo que cada token de segurança possui um número limitado de requisições permitidas.

É importante citar que os dicionários em língua portuguesa utilizados para realizar análise de sentimentos ainda são pouco desenvolvidos. Neste trabalho, com a finalidade de se obter melhores resultados, foram utilizados, em conjunto, dois dicionários abertos no idioma português: OpLexicon [12] e Sentilex [13]. Além disso, foi utilizada uma biblioteca chamada TextBlob [14]; entretanto, para se utilizar tal biblioteca, foi necessário realizar a tradução para o idioma inglês, ocasionando um viés na etapa de processamento dos dados.

1.3 Objetivos

O objetivo deste trabalho é propor um *framework* para predição de resultados de eleições brasileiras tendo por base textos extraídos da rede social Twitter que expressem a opinião de seus usuários acerca do tema. O *framework* proposto utiliza técnicas de AM em textos curtos do Twitter relacionados a políticos que estão concorrendo a cargos eletivos. Além disso, com a inserção de outros gêneros textuais na fase de treinamento do modelo de AM, é possível realizar diversas análises para distintas áreas.

1.4 Trabalho Publicado

Durante o desenvolvimento deste trabalho de graduação foi desenvolvido um artigo científico propondo um framework para análise preditiva espaço-temporal dos resultados das eleições presidenciais brasileiras tendo por base dados extraídos da rede social Twitter. O referido artigo contém os resultados iniciais deste trabalho.

B. J. G. Praciano, J. P. C. L. da Costa, J. P. A. Maranhão, J. B. Prettz, R. T. de Sousa Jr. e F. Mendonça, "Spatio-Temporal Trend Analysis of the Brazilian Elections based on Twitter Data," 2018 IEEE International Conference on Data Mining (ICDM).

1.5 Trabalhos Relacionados

Em [15] os autores definiram o conceito de análise de sentimento em vários níveis e também utilizaram algoritmos que realizam o reconhecimento de entidades,que trata

da detecção e remoção de nomes próprios com a finalidade de melhorar o desempenho dos algoritmos. O dicionário léxico utilizado para a classificação dos textos provenientes do *Twitter* foi o SentiWordNet, e as polaridades utilizadas neste trabalho foram três: Positivo, Negativo e Neutro. O trabalho abordou duas paradigmas de AM, supervisionado e não supervisionado, obtendo como melhor resultado 90

No artigo [16] foi utilizado um filtro baseado na mineração de opiniões, que é uma das áreas de análise de sentimentos. Foram utilizadas técnicas de decomposição tensorial para capturar interações intrínsecas entre dados multidimensionais. Com a aplicação de tais técnicas, houve uma grande diminuição do esforço computacional em relação à análise de sentimentos, visto que houve uma redução do tamanho do *dataset* após a decomposição tensorial.

Em [4] os autores aplicaram TM nos dados provenientes do *Twitter* que citavam as eleições presidencias de 2012 da Coréia do Sul. Foram utilizadas distintas técnicas: *topic modeling* para acompanhar as mudanças nos assuntos mais falados nas rede sociais, bem como técnicas de análise de rede para verificar quais pessoas eram citadas e por quem. Os resultados sugeriram que o *Twitter* pode ser um aliado para detectar as mudanças no contexto social enquanto são analisados o texto de quem escreveu.

Em [17] é proposto um sistema para acompanhar as eleições francesas através de tópicos escritos no *Twitter* através da análise de sentimentos. Os resultados obtidos convergiram com os resultados divulgados pelas autoridades da França e foram associados as mudanças de popularidade dos candidatos após a eleição.

Em [18], foi utilizado um *dataset* referente às eleições presidenciais colombianas de 2010. Técnicas de aprendizado supervisionado foram implementadas e também foram rotulados previamente os usuários que seriam considerados spam. Foi implementado um sistema com o objetivo de investigar o potencial que uma rede social tem de interferir em uma votação, e de acordo com os resultados obtidos, foi possível afirmar que os dados utilizados não foram consistentes.

Em [19] foi usado um dicionário léxico e apenas o algoritmo Naïve Bayes para calcular o sentimento de *tweets* que foram coletados 100 dias antes das eleições americanas de 2016. Os autores classificaram manualmente os textos extraídos do *Twitter*. Os resultados obtidos sugerem que essa rede social pode ser considerada ao realizar trabalhos com esse intuito.

1.6 Descrição dos capítulos

O presente trabalho é apresentado com a seguinte estrutura:

- Capítulo 2: Conceitos em Machine Learning e Mineração de Texto. Apresenta o conjunto de técnicas e metologias que foram necessárias para o desenvolvimento deste trabalho.
- Capítulo 3: Framework proposto. Discorre sobre a metodologia empregada no trabalho e ilustra todos os passos seguidos e necessários para entendimento do modelo.
- Capítulo 4: Conclusão. As conclusões sobre o tema são expostas.
- Capítulo 5: Trabalhos Futuros. Na última seção são discutidos quais temas que podem ser aprofundados em pesquisas futuras.

Capítulo 2

Conceitos em Aprendizado de Máquina e Mineração de Texto

O presente capítulo apresenta conceitos relacionados a Aprendizado de Máquina e Mineração de Texto, temas principais desta monografia. O capítulo está dividido da seguinte maneira. Na seção 2.1 são introduzidos os conceitos básicos para o entendimento inicial do tema. A seção 2.2 apresenta os conceitos principais acerca de mineração de texto. Por fim, a seção 2.3 ilustra os conceitos de análise de sentimentos e sua aplicação.

2.1 Aprendizado de Máquina

2.1.1 Conceitos Básicos

Com o alto volume de informações geradas no dia a dia, torna-se cada vez mais necessária a utilização de algoritmos que sejam capazes de identificar padrões, pois diversas empresas e órgãos do governo não são capazes de analisar todas as informações existentes ou entregar de maneira célere [20].

Portanto, para realizar esse tipo de atividade, é necessário entender o problema existente e ter um volume de dados razoável para realizar o treinamento do modelo. Com a técnica de AM é possível automatizar a construção de sistemas inteligentes que podem ser ajustados de acordo com a necessidade de cada tarefa [21].

Em outras palavras, AM é um conjunto de regras que possibilitam a uma máquina tomar decisões baseadas em experiências anteriores, diferentemente de um software, o qual é previamente definido para tratar cada tipo de regra. Além disso, modelos de AM podem ser aperfeiçoados e apresentar melhor desempenho quando expostos a novos dados [22].

O conceito de AM pode ser sintetizado como a capacidade de um programa de computador aprender com a experiencia (E) relacionada a alguma classe de tarefas (T), baseada em uma medida de desempenho (P). Dessa forma, o desempenho em tarefas (T), quando medido por (P), melhora com a experiencia em (E) [23].

Essas técnicas têm sido amplamente utilizadas para a resolução de diversos problemas. Atualmente, o assunto está em alta e existem diversas oportunidades para colocar em prática a matemática desenvolvida para a criação desses algoritmos. Gigantes da indústria têm investido severamente para o desenvolvimento de novas tecnologias, como os veículos autônomos, robôs advogados, veículos não tripulados e na identificação de doenças.

Existem distintos paradigmas utilizados para o ensinamento de máquinas. Entretanto, este trabalho focará apenas nos algoritmos de AM utilizados no desenvolvimento do *framework*, ou seja, não serão abordados temas como aprendizado estatístico ou redes bayesianas.

2.1.2 Paradigmas de Aprendizado de Máquina

Aprendizado Supervisionado

O aprendizado supervisionado é uma das formas de ensinar uma tarefa para a máquina, onde existe uma entrada e uma saída desejada que já foi anteriormente rotuladas. Esse processo pode ser feito de forma manual ou por meio de dicionários, quando a informação de entrada é um texto. Com os dados detalhados, a máquina é capaz de classificar novas entradas a partir de experiências antigas [23]. Na Figura 2.1 está ilustrada a etapa de treinamento do modelo. No aprendizado supervisionado, todos os dados de entrada do modelo devem ser rotuladas. No exemplo, são fornecidos três dados ao modelo, sendo dois rotulados como "cão"e um rotulado como "gato". Tendo por base as características ou *features* de cada dado de entrada, o modelo passa por uma fase de treinamento.

Na Figura 2.2 é possível visualizar o modelo preditivo em funcionamento, onde o usuário fornece uma informação de entrada e o sistema o responde com a classe que foi identificada através da entrada.

Após o treinamento descrito na Figura 2.1, o modelo é capaz de prever se um dado de entrada não rotulado pode ser classificada como "cão"ou "gato". Na Figura 2.2 é possível visualizar o modelo preditivo em funcionamento, onde o usuário fornece um dado de entrada não rotulado e o sistema responde com a classe que foi identificada.

Os algoritmos mais utilizados no aprendizado supervisionado serão detalhados a seguir.

SVM A máquina de vetor de suportes, do inglês Support Vector Machine (SVM), faz parte do aprendizado supervisionado, e permite classificar grupos de dados separando as

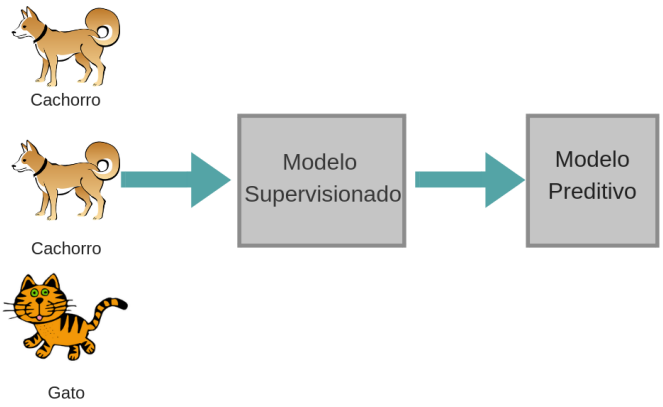


Figura 2.1: Treinamento de um modelo de aprenzidado de máquina supervisionado.

Figura 2.2: Utilização do modelo para realizar predição a partir de classes previamente treinadas.

suas margens. Essas margens são delineadas pela fração dos dados de treinamento, sendo chamadas de vetores de suporte [24].

As vantagens de utilizar SVM são [25]:

- Efetivos em espaços multidimensionais
- Eficaz em casos onde o número de dimensões é maior que o número de amostras.
- Utiliza os vetores de suporte, o que otimiza o uso de memória do computador.

- É possível utilizar diversos kernels para a função de decisão.

Como desvantagens, temos [25]:

- Se o número de entradas for muito maior que o número de amostras, existe a possibilidade de overfitting.

O SVM é construido em um hiper-plano ou em vários hiper-planos em um espaço de infinitas dimensões, que podem ser usadas para classificação ou regressão. Na Figura 2.3 é detalhado um hiper-plano que mostra uma boa separação das variáveis, possuindo a maior distância até os pontos dos dados de treinamento mais próximos de qualquer classe. No SVM, quanto maior a margem de separação, menor será o erro do classificador [26].

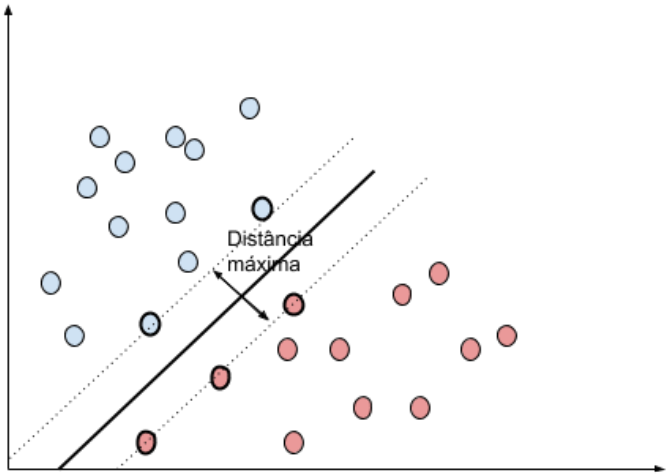


Figura 2.3: Exemplo de vetores de suporte com dimensão 2.

Neste trabalho, o SVM foi utilizado para classificar dados multi-classes, visto que foram consideradas três classes distintas durante o desenvolvimento do modelo: Positivo, Negativo e Neutro.

Tendo como os dados de treinamento $x_i \in R^p$, onde $i = 1, ..., n$. Onde $y \in 1, -1^n$, na Equação 2.1 é possível visualizar a solução matemática para esse algoritmo:

$$\begin{aligned} &\min_{\omega,\beta,\zeta} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{n}\zeta_i \\ &\text{sujeito a } y_i\left(\omega^T\phi\left(x_i\right) + b\right) \geq 1 - \zeta_i, \\ &\zeta_i \geq 0, i = 1, \cdots, n \end{aligned} \tag{2.1}$$

Considerando que o dual de um espaço de Hilbert também é um espaço de Hilbert [27], tem-se que:

$$\min_{\alpha} \frac{1}{2}\alpha^T Q\alpha - e^T\alpha$$

$$\text{sujeito a } y^T\alpha = 0 \tag{2.2}$$

$$0 \leq \alpha_i \leq C, i = 1 \cdots, n$$

onde **e** é o vetor com todos os valores, $C > 0$ é o limite superior, Q é uma matriz semi-definida positiva de tamanho n por n, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, onde $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$ é a função kernel do SVM. Os vetores de treinamento são definidos em um espaço dimensional maior pela função ϕ. Por fim na Equação 2.3, temos a função de decisão:

$$sgn(\sum_{i=1}^{n} y_i\alpha_i K(x_i, x) + \rho) \tag{2.3}$$

Onde o termo $y_i\alpha_i$ representa o coeficiente dual, $K(x_i, x)$ são os vetores de suporte e ρ é um termo independente.

Naive Bayes O Naive Bayes é outro exemplo de algoritmo de AM supervisionado. Sua formulação matemática é sustentada pelo teorema estatístico de Bayes e assume-se a independência condicional entre cada par de recursos, dado o valor da variável de classe [28]. O teorema de Bayes clássico segue a seguinte relação, onde um termo independente y e o vetor x_1 até x_n, na Equação 2.4

$$P\left(y \mid x_1, \cdots, x_n\right) = \frac{P(y)P(x_1, \cdots, x_n \mid y)}{P(x_1, \cdots, x_n)} \tag{2.4}$$

Na Equação 2.5 é desconsiderado completamente a correlação entre as variáveis,

$$P\left(x_i \mid y, x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n\right) = P(x_i|y), \tag{2.5}$$

Na Equação 2.6, para todo i, é possível simplificar através da relação:

$$P(y|x_1, \cdots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, \cdots, x_n)}, \tag{2.6}$$

Sabendo que $P(x_1, \cdots, x_n)$ é constante dada a entrada, podemos usar a seguinte regra de classificação, na Equação 2.7:

$$\widehat{y} = \arg\max_{y} P(y)\prod_{i=1}^{n} P(x_i|y), \tag{2.7}$$

O classificador Naive Bayes pode ser extremamente rápido em comparação a outros algoritmos mais sofisticados, pois existe o desacoplamento das distribuições de características condicionais de classe. Deste modo, cada uma pode ser estimada independentemente com uma distribuição unidimensional [29].

Árvores de Decisão A árvore de decisão funciona quando as classes presentes no conjunto de dados de treinamento se dividem, ou seja, esse algoritmo seleciona um problema mais complexo e divide em subproblemas mais simples e, de forma recursiva a mesma estratégia é a aplicada a cada subproblema [30].

A Figura 2.4 ilustra um exemplo de como uma árvore de decisão poderia ser utilizada para realizar análise de crédito de um cliente que deseja contrair empréstimo financeiro em um banco. O primeiro fator verificado é o salário do cliente. Caso o salário seja superior a R$ 20.000,00, o empréstimo é concedido; caso contrário, um segundo fator é analisado: grau de escolaridade. O empréstimo é concedido apenas se o cliente possui como escolaridade mínima graduação em nível universitário.

Tendo como entrada os vetores de treinamento $x_i \in R^n$, $i = 1, \cdots, n$, e o vetor com os rótulos de saída $y \in R^n$, a árvore de decisão divide recursivamente o espaço de forma que as amostras com os mesmos rótulos sejam agrupadas [31].

Considerando os dados no nó m que pode ser representado pela letra Q. Para cada candidato é dividido $\theta = (j, t_m)$ consistindo a entrada como j e o *threshold* t_m, dividindo os dados entre $Q_{left}(\theta)$ e $Q_{right}(\theta)$ e esses subconjuntos podem ser vistos na Equação 2.8

$$\begin{aligned} Q_{left}(\theta) &= (x, y)|x_j \leq t_m \\ Q_{right}(\theta) &= Q \backslash Q_{left}(\theta) \end{aligned} \tag{2.8}$$

A impureza no nó apresentado é calculada utilizando uma função de impureza denotada por $K()$, que pode ser escolhida dependendo se o problema for de classificação ou regressão. Na Equação 2.9 é calculada essa impureza:

$$G(Q, \theta) = \frac{n_{left}}{N_m} K(Q_{left}(\theta)) + \frac{n_{right}}{N_m} K(Q_{right}(\theta)) \tag{2.9}$$

Por fim, na Equação 2.10 são selecionados os parâmetros para diminuir a impureza,

$$\theta = \arg\min_{\theta} G(Q, \theta) \tag{2.10}$$

É possível gerar outros nós a partir dos subconjuntos de dados gerados para direita e para esquerda, até que a profundidade máxima permitida seja atingida, $N_m < \min_{samples}$ ou $N_m = 1$.

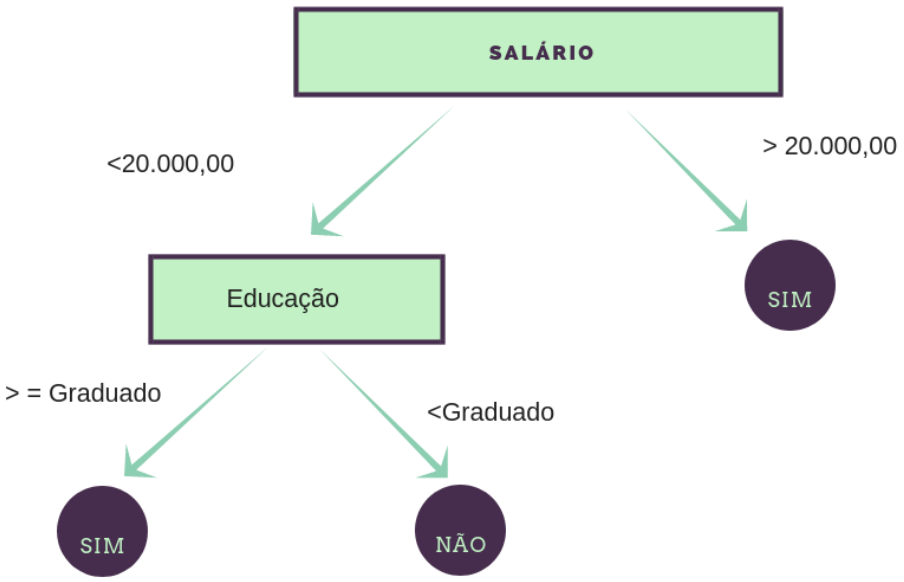


Figura 2.4: Árvore de decisão para análise de crédito de um cliente que deseja empréstimo alto no banco, levando em consideração a educação e faixa salarial.

Regressão Logística Apesar do nome, é um modelo de classificação linear e não de regressão como sugere o seu nome. Neste modelo, os dados são descritos através de um único teste e são modelados a partir de uma função logística [20].

Seja a probabilidade P(X), na Equação 2.11

$$P(x) = \frac{1}{1 + e^{-(B_0 + B_1 x)}} = \frac{e^{B_0 + B_1 x}}{1 + e^{B_0 + B_1 x}}$$

$$\frac{p(x)}{1 - p(x)} = e^{B_0 + B_1 x} \tag{2.11}$$

Onde, $_i$ são calculados através da função de máxima verossimilhança. Se $P(x) < 0.5$, a variável de resposta é classificada como 0, caso contrário, é classificada como 1.

Aprendizado não-supervisionado

Nesse tipo de aprendizado não existe a necessidade de um vetor contendo os rótulos de cada medida, ou seja, não é necessário informar à máquina o tipo de cada dado. Essa não inserção de rótulos faz com que o algoritmo aprenda de acordo com as características dos dados de entrada. Esse tipo de paradigma não tem certos benefícios que existem no aprendizado supervisionado. Para o seu funcionamento, o modelo propõe hipóteses a partir do que foi inserido na entrada [21]. Esse tipo de algoritmo é muito interessante em base de dados que ainda não foram exploradas e que necessita-se visualizar os dados agrupados.

A Figura 2.5 ilustra um exemplo do funcionamento do algoritmo não supervisionado. São fornecidos quatro dados ao modelo, sendo três com características de cães e um com características de gato. O modelo verifica os dados de entrada e forma dois clusters, sendo um referente aos cães e outro relativo ao gato. Desse modo, o modelo é capaz de agrupar em um único cluster dados de entrada que possuem características semelhantes, independentemente de rótulos.

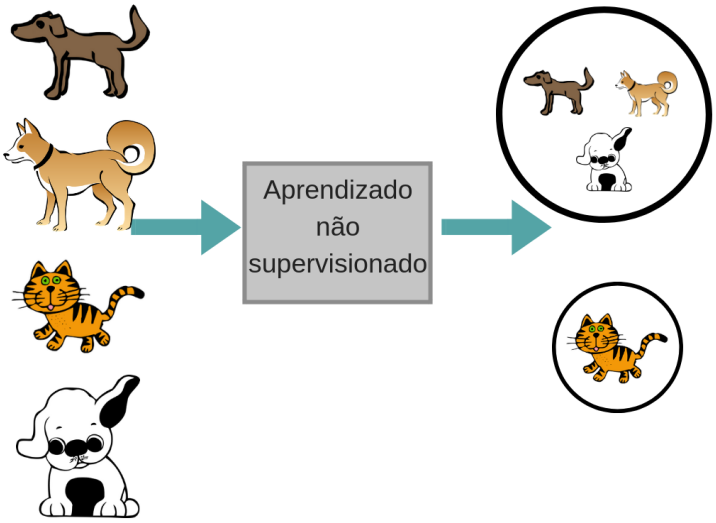


Figura 2.5: Treinamento de um modelo de aprendizado de máquina não-supervisionado.

K-means O algoritmo mais utilizado no aprendizado não-supervisionado é o *k-means*, pois a sua implementação é muito simples, pois basta comprovar o processo de minimizacão da distância quadrática total de cada ponto de um grupo, em relação ao centróide de referência [32]. Na Equação 2.12 é possível visualizar como é realizada a seleção de clusters através do número de amostras $X = x_1, x_2, ..., x_n$, descrito pela média μ_j de cada amostra nos clusters.

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2) \tag{2.12}$$

Aprendizado por reforço

O último paradigma, conhecido como aprendizado por reforço, baseia-se na forma com a qual o modelo reage de acordo com os dados passados a ele pelo ambiente, ou seja, o modelo aprende continuamente. Esse aprendizado é realizado através das ações que são executadas; caso seja a decisão correta, o modelo ganha uma recompensa, caso contrário pode ser imposta algum tipo de penalidade [33]. Nesse paradigma de aprendizado é necessário ajustar os parâmetros de forma contínua com o intuito de maximar ou minimizar um determinado índice, e sempre há um "treinador"nessa fase verificando se os estímulos proporcionados estão tendo a saída correta. Caso esses dados não estejam balanceados, eles podem alterar a performance do algoritmo [34].

Na aprendizagem por reforço, o modelo só aprende com uma série de estímulos, que podem ser recompensas ou punições no decorrer da aprendizagem. No mundo real, por exemplo, um estímulo negativo seria um cliente avaliar um motorista com nota baixa em um aplicativo de locomoção urbana, de modo que o mesmo saiba que realizou alguma conduta inadequada durante a viagem. Por outro lado, um exemplo de estímulo positivo seria um adestrador dar ao seu cão um biscoito após o mesmo realizar de forma correta algo que lhe foi solicitado.

Na Figura 2.6 é possível ver um fluxograma de treinamento de um modelo utilizando o aprendizado por reforço.

2.1.3 Medidas de avaliação

Ao utilizar um modelo AM é necessário que existam parâmetros para demonstrar se o modelo está correto ou não. Existe a possibilidade de o modelo apresentar um *overfitting*, ou seja, quando o modelo se ajusta perfeitamente ao conjunto de dados de teste, mas não é capaz de prever novos resultados. Por outro lado, o modelo apresenta um *underfitting* quando não há dados suficientes ou os dados possuem baixa qualidade, não sendo possível realizar o seu treinamento de maneira correta. [35].

Na validação cruzada, os dados são divididos de forma aleatória em n grupos de tamanho aproximadamente iguais. Este processo é realizado várias vezes, sendo que, a cada rodada do teste, considera-se um valor diferente para n. O desempenho final do algoritmo é calculado a partir da média dos desempenhos calculados em cada uma das divisões. O valor assumido por n é variável, sendo os mais comuns são 5 e 10 [36]

Figura 2.6: Fluxograma de treinamento de um modelo de aprendizado de máquina por reforço.

A matriz de confusão é outra forma de verificar a qualidade do modelo, pois ela fornece métricas importantes para avaliar o desempenho. Essa métrica é baseada em classes relacionadas à assertividade da previsão das classes. Os verdadeiros positivos (VP) e verdadeiros negativos (VN) são os valores que o sistema previu de forma correta, enquanto que os falsos negativos (FN) e falsos positivos (FP) são as entradas que foram classificadas de forma errada e tiveram suas saídas invertidas [2]. Na Figura 2.7 é possível visualizar as classes de uma matriz de confusão.

		Classe Predita	
		Positivo	Negativo
Classe Verdadeira	Positivo	**Verdadeiro** Positivo (VP)	**Falso** Negativo (FN)
	Negativo	**Falso** Positivo (FP)	Verdadeiro Negativo (VN)

Figura 2.7: Matriz de confusão [2].

Com os resultados extraídos da matriz de confusão é possível calcular outras métricas para verificar o desempenho de cada algoritmo. As Equações 2.13, 2,15, 2,15 e 2.16 apre-

sentam os cálculos das métricas Accuracy, Precision, Recall e F1-Score, respectivamente. Accuracy ilustra basicamente o percentual de acertos do modelo. A métrica Precision, por sua vez, reflete quantos dados são verdadeiramente corretos dentre todos aqueles classificados como corretos pelo modelo. O Recall, por outro lado, mostra quantos dados são verdadeiramente corretos dentre todos os dados verdadeiramente positivos. Por fim, o F1-Score combina as métricas Precision e Recall, gerando um número único que indica a qualidade geral do modelo.

$$\text{Accuracy} = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.13}$$

$$\text{Precision} = \frac{VP}{VP + FP} \tag{2.14}$$

$$\text{Recall} = \frac{VP}{VP + FN} \tag{2.15}$$

$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.16}$$

2.2 Mineração de Texto

A TM representa uma parcela da mineração de dados, também pode ser descrita como a forma de descobrir padrões em textos, que vão desde identificar palavras até mesmo sumarizar o texto ou realizar análise de sentimentos de determinada informação. A sua análise tenta recolher o maior número de informações possíveis para encontrar padrões úteis e também pode ser utilizada juntamente com algoritmos de AM. Os textos geralmente são mais difíceis de obter análises, pois em grande parte não são apresentados de forma estruturada. Portanto para que o resultado seja satisfatório é necessário uma etapa de pré-processamento dos dados, de modo que seja possível estruturar o texto.

Corpus é o conjunto de frases que foram rotulados com a sua saída que será utilizada em um modelo de AM, ou seja, ele é a base de dados que contém todos os textos disponíveis de forma rotulada.

2.2.1 Processamento de Linguagem Natural

O PLN é uma área de pesquisa que preocupa-se em explorar como um computador pode entender um texto escrito por humanos. Com esse tipo de ferramenta é possível realizar análises morfológicas, sintáticas ou léxicas no texto e também extrair informações de um determinado documento [37].

Análise de Sentimentos

A análise de sentimentos é um campo que envolve outras ciências além da computação, como a psicologia e linguística, pois é necessário a construção de dicionários para facilitar a classificação de textos. Por exemplo quando a intenção é apenas identificar a polaridade de uma palavras, ela poderá ter três saídas conhecidas: Positivo, Negativo e Neutro [17].

A análise de sentimentos no idioma português ainda é muito difícil em virtude da ausência de dicionários grandes para se realizar o tratamento de grandes volume de dados. A polaridade de cada frase pode ser verificada por meio da biblioteca [14], mas, para isso, é necessário traduzir toda a frase para o idioma inglês através da API aberta da Google, o que pode implicar um viés durante o processo de análise de sentimentos.

É possível utilizar um modelo de AM para realizar a análise de sentimentos, onde um modelo previamente treinado é aplicado para a predição da polaridade de novas frases que não existiam durante a fase de treinamento [38].

Pré-Processamento

O pré-processamento dos dados é a etapa onde é realizado o tratamento das informações, facilitando o processo de aprendizado da máquina. Nesta fase são identificados padrões de modo a facilitar a limpeza dos dados. Por exemplo, pode-se definir a utilização exclusiva de letras minúsculas, remoção de pontuação e acentos, remoção de textos contendo emotions, remoção de números, etc. Portanto, esta etapa visa filtrar o texto de forma que a máquina obtenha apenas informações que serão efetivamente utilizadas durante o treinamento. [39].

Tokenização

A tokenização acontece na etapa de pré-processamento e tem como finalidade extrair pequenas informações de um texto livre. Recebe esse nome, pois cada divisão é nomeada de *token*, o qual, na maioria das vezes, é uma palavra que está inserida no texto. Entretanto, em outros casos, podem ser apenas letras ou, ainda, duplas ou trincas de palavras [40]. Neste trabalho foi utilizado o *token* em sua forma clássica utilizando a linguagem de programação *Python* e a biblioteca *NLTK* [41].

Por exemplo, caso tenhamos a frase *"Obina é melhor do que Neymar!"*, teremos sete tokens, conforme exemplo abaixo.

[*Obina*] [é] [*melhor*] [*do*] [*que*] [*Neymar*] [!]

Stemming

A técnica de stemming tem por objetivo evitar que palavras que tenham o mesmo radical sejam classificadas de maneiras distintas. No processo de *stemming*, cada palavra é reduzida até a menor parte que mantenha o significado da palavra, ou seja, o radical. Por exemplo, palavras como caminhar, caminhada, caminharam, caminharão resultam na mesma palavra final: caminh. É necessário atentar-se na escolha do algoritmo correto para realizar o steeming, pois essa etapa está diretamente ligada ao idioma em que o texto está escrito.

Stop Words

Stop Words são os termos considerados não relevantes para a análise do texto. São palavras que, em sua maioria, são conectivos de frases ou preposições, os quais podem ser consideradas como palavras vazias. Não existe uma lista universal de *Stop Words*, mas, por padrão, é utilizada a lista presente na biblioteca NLTK [41]. Na Tabela 2.1 encontram-se algumas das *Stop Words* utilizadas neste trabalho.

de	os	tua	tem	estão	da	lhes	essas
e	é	foi	nossas	muito	o	se	tuas
tu	por	as	sua	aquele	entre	não	ele
delas	minhas	às	nos	pela	havia	me	como
ser	aqueles	nossa	vocês	eu	ter	tenho	suas
está	isso	pelos	estes	tinha	depois	foram	este
para	só	quem	deles	isto	um	eles	do
vos	mais	mesmo	num	dele	será	minha	a
no	teus	à	você	em	meus	esses	pelas
com	ao	dela	há	que	na	nosso	te
aos	dos	ou	aquela	era	uma	das	esta
teu	nem	já	até	seja	esse	mas	quando
aquelas	nossos	têm	também	seus	lhe	meu	seu
ela	elas	estas	nós	sem	essa	fosse	qual
		pelo	nas	numa	aquilo		

Tabela 2.1: *Stop Words* utilizadas neste trabalho

2.2.2 Bag of Words

Os dados utilizados neste trabalho são textos, ou seja, é um formato que não existe um padrão conhecido, o que dificulta a extração de informações. Por isso, é necessário alguma técnica que organize esse conjunto de dados e extraia informações relevantes deles [42].

Portanto, é necessário utilizar o *bag of words* para realizar a estruturação do texto que será analisado. No caso mais simples, seria uma matriz que apenas indica se determinado

termo existe ou não naquela frase, constituindo-se em uma abordagem booleana. Onde temos duas frases que serão transformadas na matriz a ser utilizada nas próximas etapas.

Por exemplo, as frases "José gosta de assistir os jogos do Flamengo"e "Maria gosta de assistir filmes"podem ser organizadas na seguinte matriz, onde o valor "1"indica que uma dada palavra aparece na frase considerada, enquanto que o valor "0"indica o caso oposto:

$$\begin{bmatrix} \text{José} & \text{gosta} & \text{assistir} & \text{jogos} & \text{Flamengo} & \text{Maria} & \text{filmes} \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

2.2.3 Term Frequency

A medida de *Term Frequency* é uma outra forma de representar os dados na matriz do modelo estudado. Ele considera a quantidade de ocorrências de um determinado termo n_t, o que difere drasticamente da abordagem bag of words [43].

Esse modelo tem uma performance superior ao método booleano apresentado na subseção anterior Sua fórmula matemática é descrita na Equação 2.17.

$$TF_t = \frac{n_t}{N}, \tag{2.17}$$

onde n_t é a frequência que o termo n apareceu dentro do *Corpus* e N é a quantidade total de termos.

2.2.4 Term Frequency - Inverse Document Frequency (TF-IDF)

A técnica *Term Frequency* considera que todas as palavras possuem a mesma importância. Uma abordagem mais eficaz é a *Term Frequency-Inverse Document Frquency* (TF-IDF), que realiza uma normalização com a finalidade de diminuir a influência de cada termo sobre o processo de análise textual. A Equação 2.18 apresenta a taxa de normalização dos valores de cada termo [44]:

$$IDF = \log \frac{N}{d}, \tag{2.18}$$

Onde N é o número total de frases e d é a quantidade de documentos nos quais o termo aparece.

A fórmula final do TF-IDF é obtida a partir do produto da entre a frequência de cada termo e a taxa de normalização dos valores, dados pelas Equações 2.17 e 2.18, respectivamente. Na Equação 2.19 é possível ver que esse tipo de normalização desconsiderá palavras que aparecem em todas as frases do *Corpus*, pois esses termos não são considerados úteis para as etapas seguintes:

$$TF - IDF = TF * IDF. \tag{2.19}$$

Capítulo 3

Framework Proposto

Neste capítulo, é detalhado o framework proposto para análise preditiva das tendências das eleições presidenciais no Brasil com base na análise de sentimentos dos dados do Twitter. Conforme mostrado na Figura 3.1, o framework é dividido em cinco blocos, que são descritos nas subseções de 3.1 a 3.5.

3.1 Extração de Dados

O primeiro bloco da Figura 3.1 corresponde ao rastreamento e extração de *tweets*. Neste bloco, os tweets são extraídos do banco de dados do *Twiter* disponível na Internet. Como a nova versão da API dessa rede social não permite extrair tweets para datas anteriores a uma semana, foi necessário desenvolver um *Web Crawler*, ou seja, um aplicativo usando a linguagem *Python* com a biblioteca *Scrapy*. As datas usadas para a extração de tweets foram uma semana antes do primeiro dia da eleição e no próprio dia da eleição. Este procedimento foi replicado para o segundo turno das eleições. Os dados utilizados neste trabalho são referentes às eleições de 2014, onde o segundo turno foi disputado entre os candidatos Aécio Neves e Dilma Rousseff.

Todos os tweets foram coletados em um intervalo pré-definido como o argumento do mecanismo de busca de tweets. Como é necessário aplicar as técnicas de Processamento de Linguagem Natural (PLN) em português, a preferência foi dada aos conteúdos escritos naquela língua. Foram utilizados mais de 100.000 tweets para desenvolver o modelo de aprendizado de máquina e quatro algoritmos de classificação foram considerados para a validação do modelo.

Além do texto do *tweet*, algumas outras informações coletadas incluem autor, data, contagem de retweets, contagem de favoritos, localização, menções, hashtags, ID de publicação e link de publicação. Essas informações são organizadas em uma estrutura de

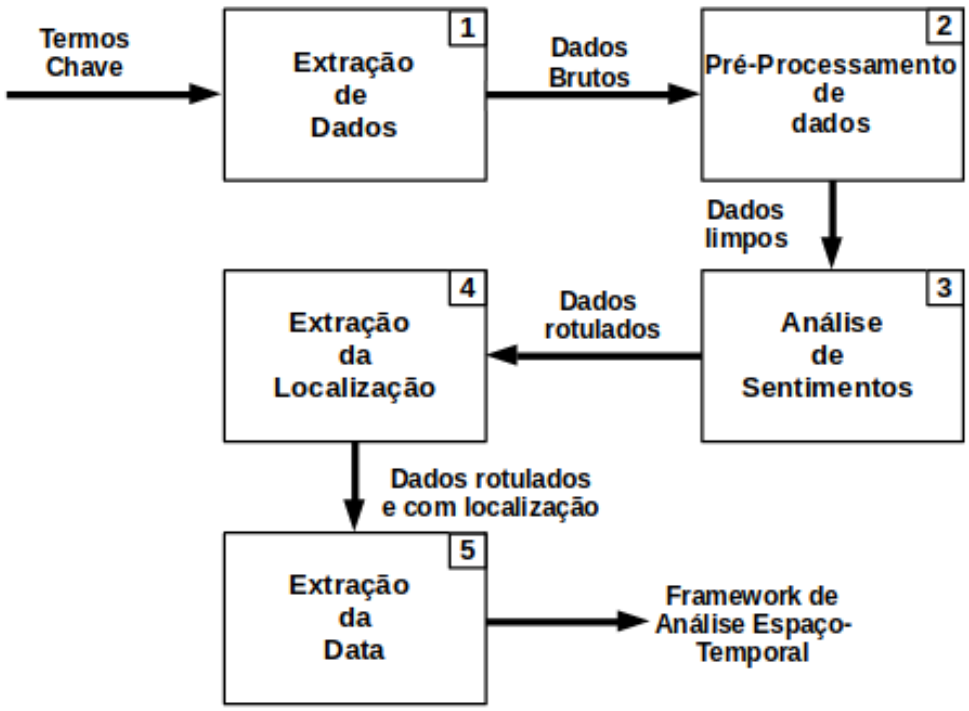


Figura 3.1: Diagrama em blocos do framework proposto para análise preditiva espaço-temporal com base nos dados do Twitter.

dados denominada *dataframe*, que é semelhante a uma matriz, mas contém o nome de cada coluna.

Para manipular os dados em formato de *dataframe*, foi utilizada a biblioteca *Pandas*, que facilita todo o processo de manipulação de dados [45]. Na Figura 3.2 apresenta um exemplo de dataframe contendo os dados utilizados para realizar a análise.

3.2 Pré-processamento de dados

O segundo bloco da Figura 3.1 corresponde ao pré-processamento de dados. As redes sociais apresentam múltiplos públicos e a intensidade da emoção é o fator que diferencia um tweet de outro, onde intensidade se refere ao grau ou quantidade de uma emoção, como positivo, negativo ou neutro. Uma estratégia usual é considerar que todas as mensagens coletadas têm a mesma importância [46]. Não há padrão de escrita definido para ser usado em redes sociais. Consequentemente, foi necessário realizar a limpeza de dados

	username	date	retweets	favorites	text	geo	mentions	hashtags	id
0	RaulDudek	05/10/2014 19:59	0	0	geraldo alckmin reeleito governador sao paulo ...	NaN	NaN	#AECIOdeVirada	5,19E+17
1	wtshatboy	05/10/2014 19:59	1	0	turno dilma aecio dois bosta	NaN	NaN	NaN	5,19E+17
2	RNemitz	05/10/2014 19:59	0	2	dilma rousseff pt aecio neves psdb vao dis...	NaN	NaN	NaN	5,19E+17
3	GeiciSantiago	05/10/2014 19:59	0	0	dilma novo nao ne	NaN	NaN	NaN	5,19E+17
4	michaelserra	05/10/2014 19:59	0	0	vai ser tao apertado votos luciana genro podem...	NaN	NaN	NaN	5,19E+17
5	potterhalder	05/10/2014 19:59	1	0	voces idolatrando aecio minimo nao	NaN	NaN	NaN	5,19E+17

Figura 3.2: Dataframe com os dados utilizados para realizar as análises.

para padronizar as sentenças. Além disso, como estamos analisando um processo sério, que são as eleições presidenciais, todos os emoticons foram desconsiderados. O fluxo de limpeza de dados pode ser visto na Figura 3.3.

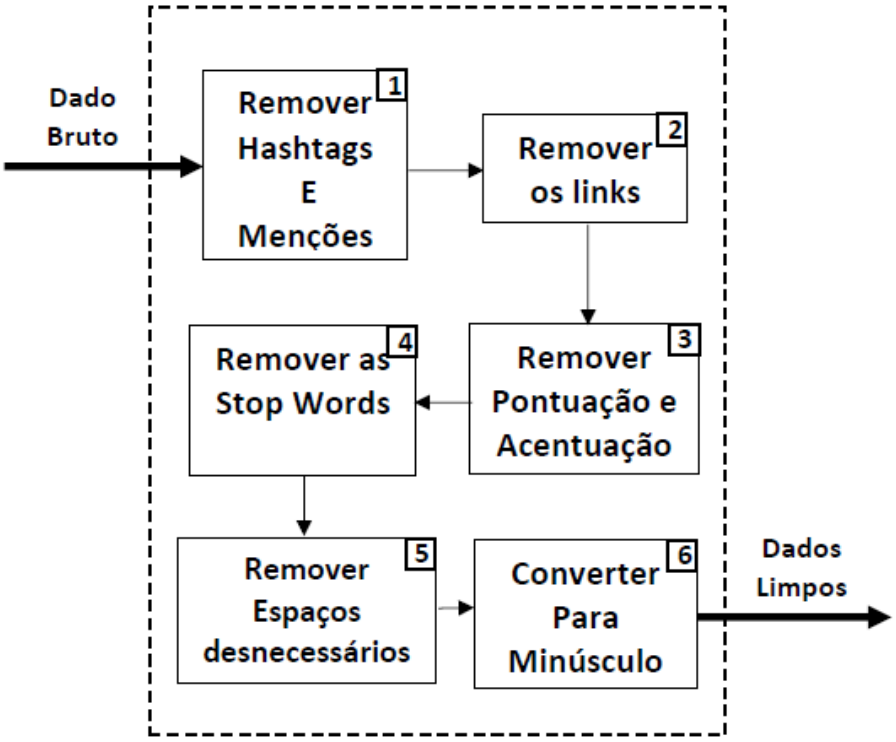


Figura 3.3: Fluxo de pré-processamento de dados.

Foram definidas seis etapas para a limpeza dos dados, conforme pode ser visualizado

Tabela 3.1: Exemplo de limpeza de textos extraídos do *Twitter*

Etapa Inicial	Mais uma vez @AecioNeves perde em Minas. Não seria a hora de ele se desculpar com o estado, ao invés de continuar. #Eleições https:/bit.ly/twitter
Remoção de Hashtags e Menções a Perfis	Mais uma vez perde em Minas. Não seria a hora de ele se desculpar com o estado, ao invés de continuar. https:/bit.ly/twitter
Remoção de Links	Mais uma vez perde em Minas. Não seria a hora de ele se desculpar com o estado, ao invés de continuar.
Remoção de Pontuação e Acentuação	Mais uma vez perde em Minas Nao seria a hora de ele se desculpar com o estado ao inves de continuar
Remoção de Stopwords	perde em Minas seria hora desculpar estado inves continuar
Remoção de Espaços Desnecessários	perde em Minas seria hora desculpar estado invés continuar
Converter para minúsculo	perde em minas seria hora desculpar estado invés continuar

na Figura 3.3. A primeira etapa é responsável por remover tags de retweet e menções de profile. A segunda e terceira etapas tratam da remoção de links de sites e de acentos e pontuações, respectivamente. Em seguida, todas as stopwords em português e os espaços desnecessários são removidos na quarta e quinta etapas, respectivamente. Por fim, a sexta etapa trata da conversão dos tweets para letras minúsculas.

As *Stop Words* são freqüentemente usadas em sentenças que desempenham um papel muito pequeno na análise de sentimento e, consequentemente, devem ser removidas [47]. Eles não contribuem para o processo de análise de sentimento e apenas retardam o processo. Além disso, os dados devem ser submetidos a um processo de stemming em que as palavras são reduzidas ao seu radical [46].

Na Tabela 3.1, mostra um exemplo de a limpeza de um tweet selecionado aleatoriamente do *dataframe* utilizado para análise.

No apêndice A.1 o código utilizado para a limpeza e estruturação dos dados é apresentado Foram utilizadas expressões regulares para substituir carácteres especiais, possibilitando a aplicação do processo de stemming numa etapa posterior.

Tabela 3.2: Classificação do tweets através da polaridade usando a biblioteca TextBlob e os dicionários OpLexicon/Sentilex.

	Positivo	Neutro	Negativo
TextBlob	42.67%	24.01%	33.27%
OpLexicon/Sentilex	25.12%	26.51%	48.35%

Tabela 3.3: Análise de sentimentos usando três tweets selecionados de forma aleatória

Tweet	TextBlob	Oplexicon/Sentilex
1	Negativo	Positivo
2	Neutro	Neutro
3	Negativo	Negativo

3.3 Análise de Sentimentos

O terceiro bloco da Figura 3.1 corresponde à análise de sentimentos. Neste trabalho foram utilizadas as bibliotecas TextBlob e OpLexicon para processamento de texto. A Tabela 3.2 mostra as porcentagens de polaridade para cada biblioteca obtida para os tweets extraídos após o pré-processamento. Após a classificação dos dados, é necessário criar dois bancos de dados distintos para que o TextBlob e o OpLexicon possam ser comparados usando modelos de aprendizado de máquina.

A Tabela 3.3 mostra um exemplo da análise de sentimento de uma amostra aleatória com três tweets para comparar os resultados obtidos com as duas bibliotecas.

Em primeiro lugar, foi utilizada a biblioteca TextBlob na linguagem Python como principal fonte de classificação. Entretanto, como tal biblioteca utiliza um dicionário léxico de palavras inglesas [8], foi necessário traduzir todos os tweets que não estavam em inglês utilizando um método presente na biblioteca para, em seguida, verificar a polaridade da frase. Nesta etapa, as palavras podem perder seu significado, visto que o processo de tradução pode apresentar um alto viés e o idioma português apresenta várias maneiras para que uma mesma ideia seja exposta.

O OpLexicon em conjunto com o Sentilex com os melhores resultados foi constituído por 30.322 palavras (23.433 adjetivos e 6.889 verbos) e foi baseado no português brasileiro. Foi classificada por sua categoria morfológica marcada com polaridades positivas, negativas e neutras [48].

A automação desse processo é necessária, pois torna-se inviável rotular mais de 100,000 entradas. O código utilizado para análise de sentimentos usando os dicionários pode ser visualizado no Apêndice A.2.

3.4 Extração da Localização e Data

O quarto bloco da Figura 3.1 corresponde à extração da localização, onde são obtidas informações sobre as coordenadas geográficas do tweet. Como o banco de dados apresenta o nome do autor da publicação, a latitude e a longitude registradas em seu perfil pessoal, quando disponíveis, podem ser obtidas. Permite determinar a localização geográfica dos usuários que apresentam boas ou más opiniões sobre um determinado candidato e, consequentemente, intensificar as ações de marketing a serem tomadas naquela região. Portanto, o objetivo do trabalho é unir a geolocalização com a análise sentimental. Foi utilizada a biblioteca denominada *tweepy* [49] e o código na seção A.5 detalha como é feita a requisição para obter os dados referentes à localização do usuário.

A Figura 3.4 ilustra um exemplo de tweet contendo a data de publicação e o respectivo conteúdo.

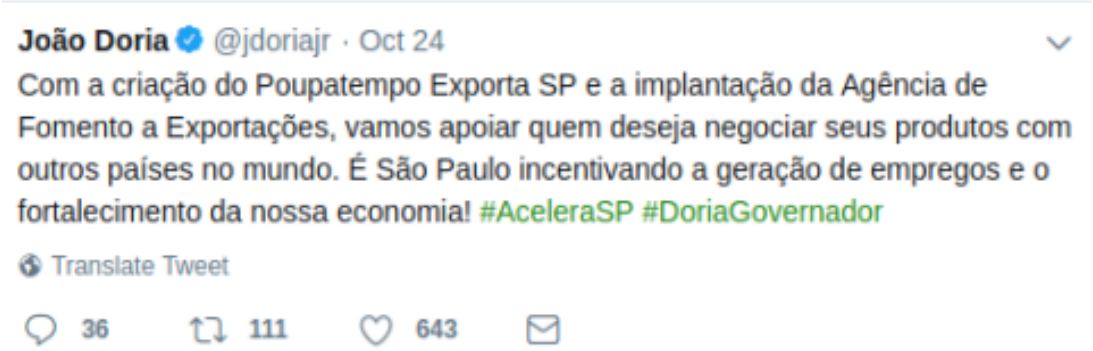


Figura 3.4: Exemplo de um tweet extraído da rede social analisada.

A Figura 3.5 apresenta as informações pessoais do perfil referente ao autor do tweet mostrado na Figura 3.4. A localização do autor pode ser facilmente extraída desse campo. A data será agrupada a cada dia, para que seja possível analisar a série temporal de forma precisa.

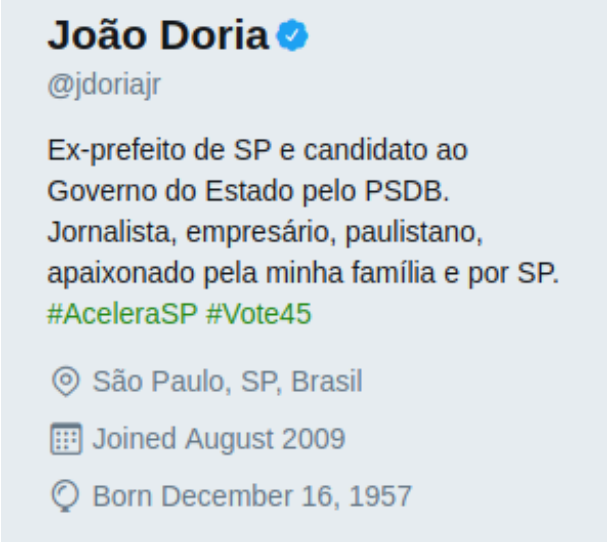


Figura 3.5: Exemplo de um perfil na rede social analisada.

3.5 Aprendizado de Máquina Supervisionado

Após realizar o tratamento dos dados, o texto está pronto para ser utilizado para treinamento. Esse *dataset* será utilizado como treinamento inicial dos classificadores que serão utilizados para analisar os textos futuros.

Os algoritmos utilizados para criar o modelo de AM supervisionado foram detalhados nas seções 2.1.2.1.1 a 2.1.2.1.4. Tais algoritmos foram utilizados para classificar os novos dados que serão extraídos.

A fase treinamento funcionou da seguinte forma. Foi utilizado o príncipio de Pareto para realização do treinamento, onde 20% dos dados foram separados para teste e validação do modelo e os outros 80% foram utilizados para a fase de treinamento [50].

Foi necessário transformar o *dataframe* obtido ao realizar a extração de dados da rede social em uma matriz utilizando a técnica *Bag-of-Words*, juntamente com a técnica que foi apresentada na seção 2.2.4, onde foi contabilizada a quantidade de informações e normalizado os valores, para que o resultado da análise de sentimentos fosse otimizado.

Após transformar todos os dados disponíveis para o formato vetorial e dividir o *dataset* entre conjunto de testes e treinamento, utilizou-se os algoritmos biblioteca *Scikit-Learn* que foram detalhados nas Seções 2.1.2 a 2.1.2 [25].

O primeiro algoritmo a ser utilizado na fase de treinamento foi o SVM, que, segundo a literatura, é o que apresenta os melhores resultados para classificação de textos. Nessa etapa foram utilizados o kernel linear e funções de *gridsearch* para encontrar o valor da constante C, que é uma variável de normalização para minimizar os erros no conjunto de treinamento e obter as melhores métricas de análise [27].

O valor encontrado foi $C = 100$ e, para otimizar a fase de treinamentos, foi utilizada uma função de paralelismo de modo que o treinamento não fosse longo, reduzindo o custo computacional.

O algoritmo de Naive Bayes (NB) foi o segundo a ser utilizado para treinamento. A implementação desse algoritmo acontece de forma simples, pois é computada apenas a probabilidade condicional para cada valor de saída a partir do texto de entrada.

As árvores de decisão foi o classificador que mais consumiu recursos e tempo na fase de treinamento, pois o dataset apresenta muitas linhas para análise e a variável de entrada pode ter até 140 caracteres. Mesmo otimizando os dados através dos processos apresentados na Seção 3.2, a árvore construída teve uma alta profundidade, tornando-se extremamente custoso ao computador que realiza o treinamento.

Por último, foi considerado o algoritmo de regressão logística, cujo tempo de treinamento é considerável em comparação à árvore de decisão. Esse classificador possui uma função implementada na biblioteca utilizada nessa etapa.

Esta seção apresenta a validação do *framework* proposto. As previsões do *framework* foram comparadas com os resultados das eleições presidenciais de 2014 extraídos do banco de dados do Tribunal Superior Eleitoral. Foram analisados apenas os resultados do segundo turno das eleições presidenciais, ou seja, somente os dados referentes aos candidatos Dilma Rousseff e Aécio Neves.

3.6 Avaliação da Performance dos Algoritmos

O desempenho dos algoritmos Naive Bayes (NB), Máquina de Vetores de Suporte (SVM), Regressão Logística (LR) e Árvores de Decisão (DT) foi avaliado através das seguintes métricas: *Accuracy*, *Precision*, *Recall* e *F1-Score*. Essas informações podem ser visualizadas na Tabela 3.4. O algoritmo de árvore de decisão apresentou os piores resultados em todas as métricas avaliadas e também apresentou maior custo computacional. Os algoritmos com os melhores resultados em termos de precisão e custo computacional foi o SVM, pois foi utilizada a função de *gridsearch* para diminuir os erros. Os dados de texto são ideais para classificação de SVM devido à natureza esparsa do texto, em que poucos recursos são irrelevantes, mas tendem a ser correlacionados entre si e geralmente organizados em categorias linearmente separáveis [51].

3.7 Validação Cruzada N-Fold

Na validação cruzada de N-fold, o conjunto de dados é particionado em N subconjuntos mutuamente exclusivos. Um subconjunto é usado como dados de validação para

Tabela 3.4: Performance dos classificadores utilizados

Métricas	Classificador	TextBlob	OpLexicon/Sentilex
Accuracy	NB	0.82	0.93
	SVM	0.94	0.98
	LR	0.70	0.65
	DT	0.64	0.85
Precision	NB	0.83	0.92
	SVM	0.94	0.98
	LR	0.79	0.83
	DT	0.69	0.89
Recall	NB	0.79	0.92
	SVM	0.93	0.97
	LR	0.60	0.58
	DT	0.56	0.81
F1-Score	NB	0.80	0.92
	SVM	0.94	0.98
	LR	0.55	0.60
	DT	0.55	0.84

testar o modelo,enquanto os outros $N - 1$ restantes restantes são usados como dados de treinamento. Como o conjunto de dados tem mais de 100.000 *tweets*, a validação cruzada de N-fold apresentaria um alto custo computacional se um grande valor de N fosse escolhido. Consequentemente, usamos $N = 5$. Os dados do teste podem ser visualizados na Tabela 3.5. Usando validação cruzada de 5-fold, obtivemos resultados semelhantes na primeira análise e o SVM permaneceu como o melhor algoritmo para mineração de texto em português.

Tabela 3.5: Validação cruzada $N - Fold$ com $N = 5$

Métrica	Algorithm	TextBlob	OpLexicon/Sentilex
Accuracy	NB	0.81	0.92
	SVM	0.93	0.98
	LR	0.86	0.66
	DT	0.64	0.84
Precision	NB	0.82	0.92
	SVM	0.92	0.98
	LR	0.78	0.83
	DT	0.70	0.88
Recall	NB	0.86	0.94
	SVM	0.93	0.98
	LR	0.90	0.80
	DT	0.54	0.81
F1-score	NB	0.79	0.91
	SVM	0.92	0.98
	LR	0.54	0.58
	DT	0.56	0.84

Tabela 3.6: Distribuição dos resultados da eleição presidencial de 2014

Dilma Rousseff	51,64%
Aécio Neves	48,36%

3.8 Avaliação do erro utilizando o modelo

Visto que o algoritmo SVM apresentou os melhores resultados dentre todos os classificadores utilizados, ele foi escolhido para se avaliar o erro da análise de sentimento. O conjunto de dados de treinamento consiste de 3000 tweets que foram divididos em três grupos de 1000 tweets cada, onde cada grupo apresenta uma determinada polaridade (positivo, neutro ou negativo). Como descrito na matriz de confusão da Figura 3.6, o *framework* classificou corretamente tweets positivos, negativos e neutros com taxas de 99,50%, 86.90 % e 70.60 %, respectivamente. Neste caso, usamos o termos-chave "Aécio"e "Dilma", que se referem respectivamente aos candidatos nomes Aécio Neves e Dilma Rousseff.

3.9 Resultado das Eleições

A presente seção apresenta a comparação entre os resultados obtidos pelo classificador e os dados reais das eleições anteriores extraídas do site do Tribunal Superior Eleitoral brasileiro [3]. Conforme observado anteriormente, este trabalho realizará a análise apenas do segundo turno das eleições presidenciais de 2014.

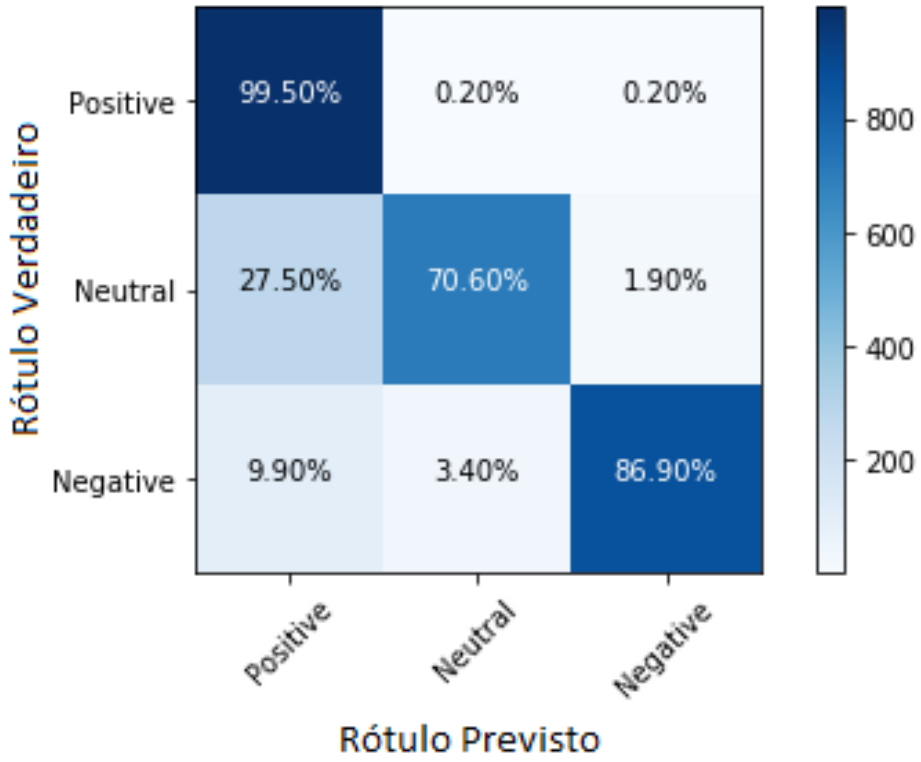


Figura 3.6: Matriz de confusão comparando a classificação manual com SVM considerando os termos chaves Dilma e Aécio.

A Tabela 3.6 mostra um exemplo da distribuição dos votos dos candidatos às eleições presidenciais brasileiras de 2014, Aécio Neves e Dilma Rousseff, obtidos do banco de dados do Tribunal Superior Eleitoral.

3.10 Análise Espaço-Temporal

Por meio da análise temporal, é possível observar a intensidade de cada candidato no dia da eleição. Tal análise destina-se a demonstrar quais as reações dos brasileiros que estavam acompanhando os resultados das eleições presidenciais em tempo real. A Figura 3.7 apresenta a quantidade de tweets classificados nas categorias positivo, negativo e neutro para os candidatos Dilma Rousseff e Aécio Neves entre os dias 12 e 28 de outubro de 2014.

De acordo com a Figura 3.7, a quantidade de informações rotuladas como positiva para a candidata Dilma Rousseff é superior com uma pequena distância em relação ao oponente Aécio Neves no dia das eleições presidenciais, ocorrido em 26 de outubro de 2014. Na figura 3.7 é possível visualizar uma queda na popularidade da candidata Dilma Rousseff.

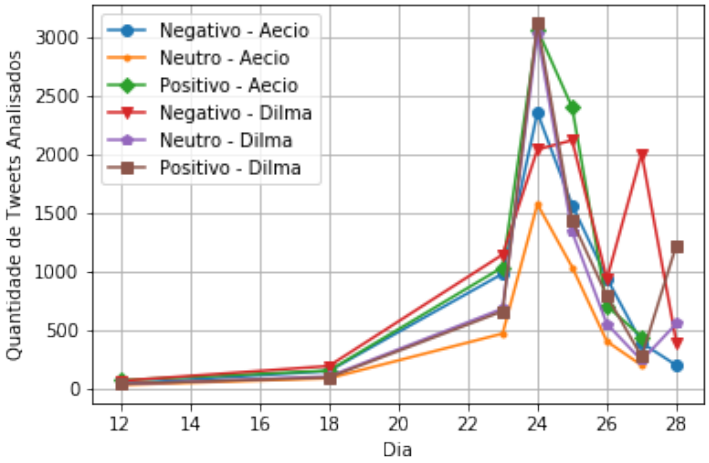


Figura 3.7: Série temporal utilizando os tweets classificados durante o segundo turno com suas polaridades.

Tal fato está diretamente ligado às atividades da operação da Polícia Federal, denominada Lava Jato, que ocasionaram a prisão de várias autoridades políticas no período.

Além disso, é possível verificar na Figura 3.7 um aumento de mensagens de teor negativo a partir do dia 24 de outubro de 2014. Tal acontecimento se deve ao fato de que a pessoa responsável pelo esquema de corrupção prestou declarações afirmando que a presidente Dilma Rousseff e o ex-presidente Luís Inácio Lula da Silva tinham conhecimento de todo o esquema de corrupção na empresa estatal Petrobras. Outro fato que colaborou para o aumento do teor negativo foi o fato de que o executivo de uma empresa envolvida em esquemas de propina ter fechado um acordo de delação premiada com os procuradores da força tarefa da operação Lava Jato [52].

A Figura 3.8 mostra os resultados das eleições presidenciais de 2014 em cada estado brasileiro. As cores vermelha e azul representam, respectivamente, as regiões onde os candidatos Dilma Rousseff e Aécio Neves tiveram maior número de votos.

A Figura 3.9 ilustra as predições dos resultados das eleições presidenciais fornecidas pelo framework conforme as polaridades dos tweets para cada candidato. As cores vermelha e azul representam respectivamente os candidatos Dilma e Aécio.

3.11 Plataforma de Análise de Sentimentos *Online*

Como resultado deste trabalho foi desenvolvida uma interface online para classificação automática de novos textos extraídos de redes sociais. Na Figura 3.10 é detalhada a tela principal que serve para que o usuário envie o arquivo extraído do *twitter* para análise. Foi utilizado o modelo criado de forma persistiva do algoritmo SVM.

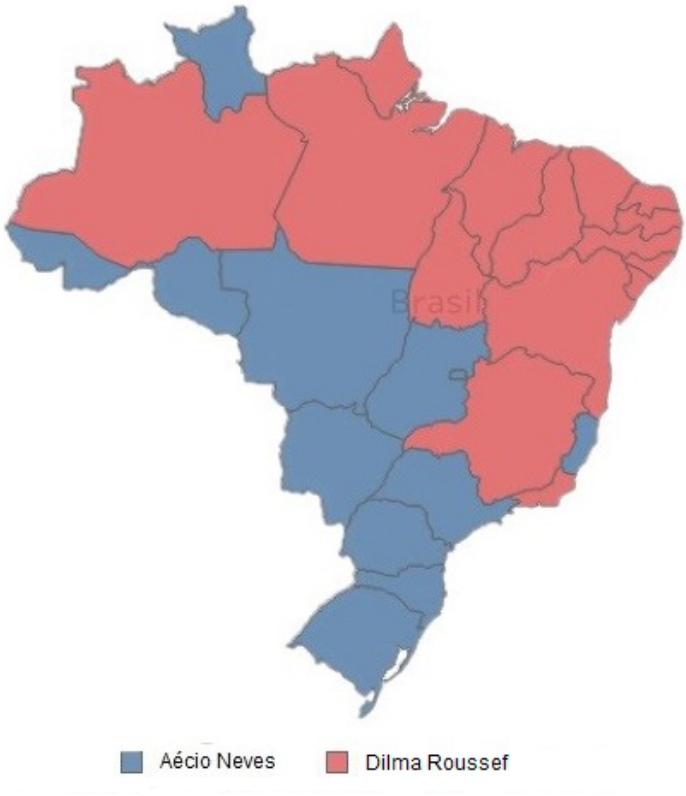


Figura 3.8: Resultado das eleições presidencias de 2014 pelo TSE [3].

Esta seção apresenta os resultados do segundo turno das eleições presidenciais de 2018 previstos pela plataforma de análise de sentimentos online. Durante o dia das eleições, em 28 de outubro de 2018, foram extraídos do site do TSE dados referentes aos candidatos Fernando Haddad e Jair Bolsonaro. Com o objetivo de agrupar um grande volume de dados para classificação, os tweets foram extraídos da base de dados do Twitter com periodicidade de uma hora ao longo do dia das eleições. Nesse processo foram extraídos mais de 200.000 tweets por candidato, totalizando mais de 400.000 linhas de informações. A Figura 3.11 apresenta uma tela contendo os detalhes do dataset utilizado para a análise de sentimentos, contendo várias informações dos tweets, tais como nome do autor, data, hora, conteúdo, geolocalização e polaridade.

A avaliação de cada sentimento no decorrer da eleição pode ser visualizada na Figura 3.12, onde é possível ver a diferença entre os dois candidatos no que diz respeito a polaridade positiva. Neste trabalho, consideramos palavras de cunho positivo como pessoas que votam no candidato Jair Bolsonaro, enquanto que as palavras negativas são consideradas como críticas de eleitores de outros candidatos.

A contabilização dessa análise pode ser vista na Tabela 3.7, onde o resultado é favorável ao candidato Jair Bolsonaro, que obteve um total de 54,47

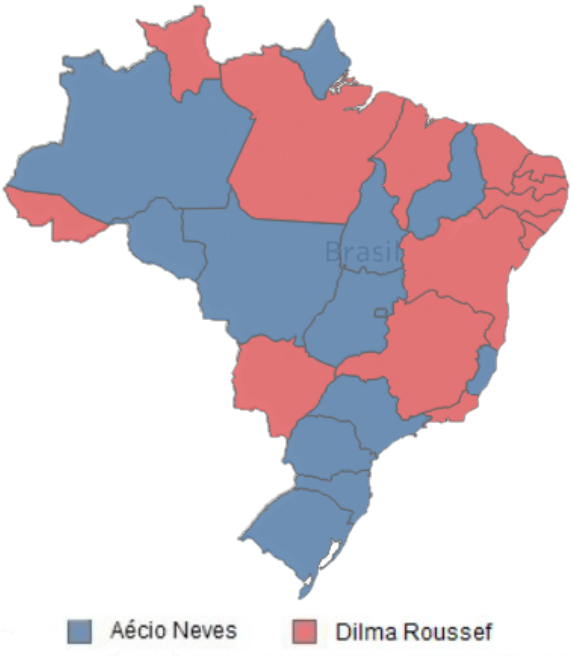


Figura 3.9: Predição da distribuição de votos utilizando o *framework* desenvolvido.

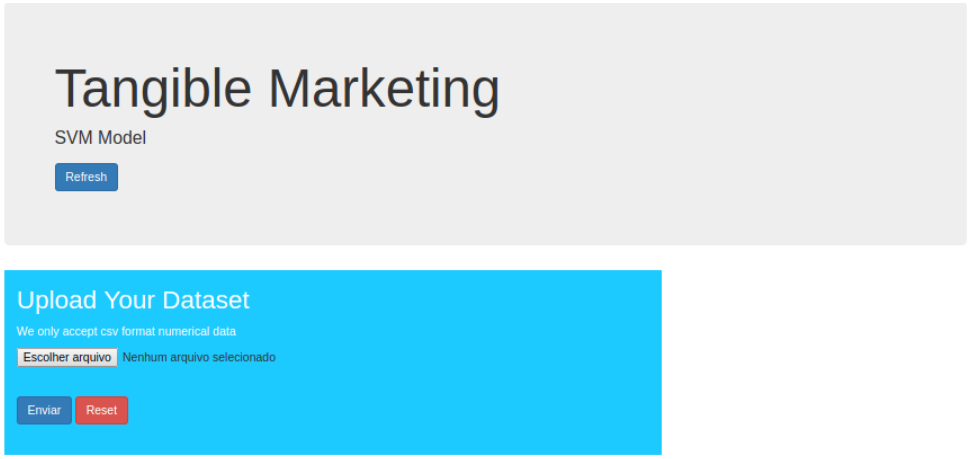
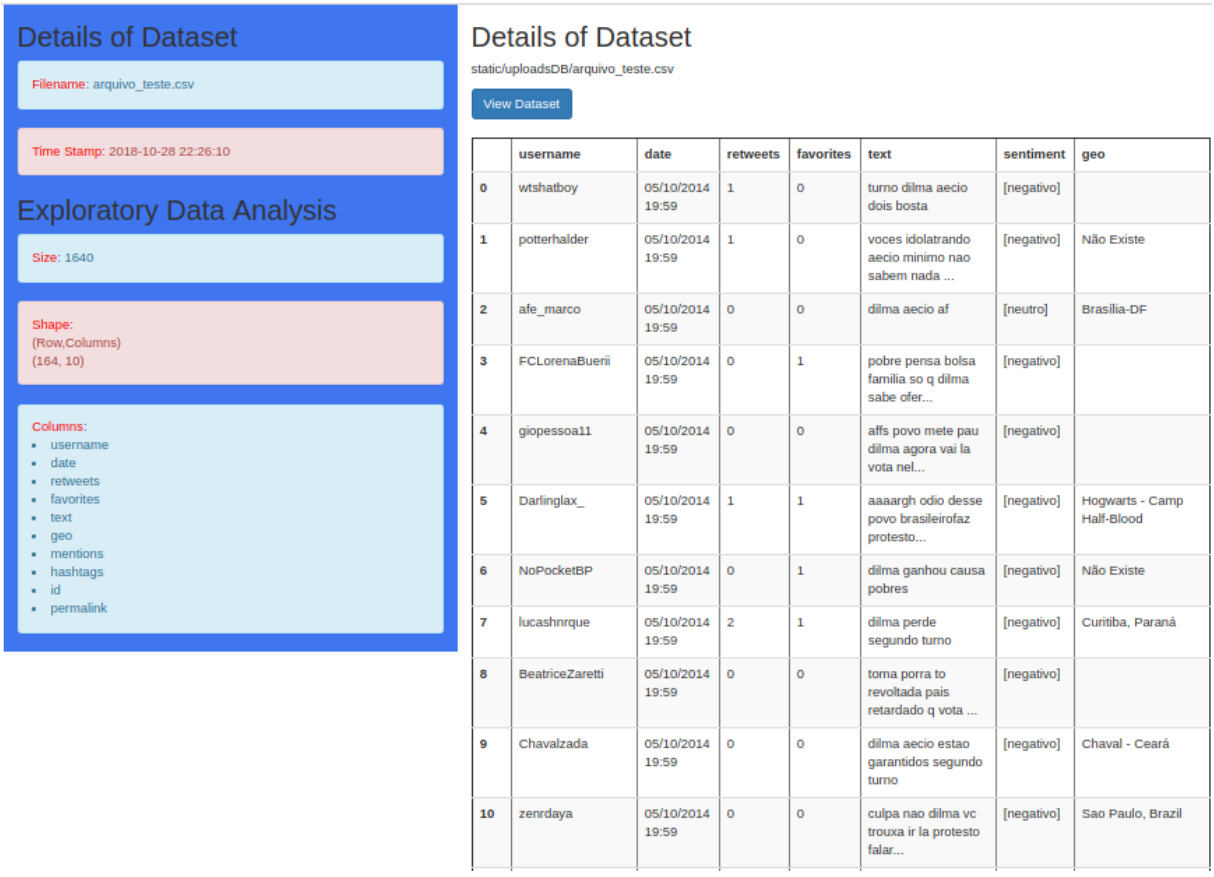


Figura 3.10: Tela principal do protótipo de classificação automática de textos provenientes de redes sociais.

Como as eleições brasileiras utilizam as urnas eletrônicas durante o processo eleitoral, é possível saber quem ganhou somente após a finalização da votação no último colégio eleitoral do Brasil, que é o estado do Acre. Na Tabela 3.8 apresenta o resultado oficial do TSE.

Ao comparar o resultado do *framework* com o resultado oficial das eleições brasileiras,



	username	date	retweets	favorites	text	sentiment	geo
0	wtshatboy	05/10/2014 19:59	1	0	turno dilma aecio dois bosta	[negativo]	
1	potterhalder	05/10/2014 19:59	1	0	voces idolatrando aecio minimo nao sabem nada ...	[negativo]	Não Existe
2	afe_marco	05/10/2014 19:59	0	0	dilma aecio af	[neutro]	Brasília-DF
3	FCLorenaBuerii	05/10/2014 19:59	0	1	pobre pensa bolsa familia so q dilma sabe ofer...	[negativo]	
4	giopessoa11	05/10/2014 19:59	0	0	affs povo mete pau dilma agora vai la vota nel...	[negativo]	
5	Darlinglax_	05/10/2014 19:59	1	1	aaaargh odio desse povo brasileirofaz protesto...	[negativo]	Hogwarts - Camp Half-Blood
6	NoPocketBP	05/10/2014 19:59	0	1	dilma ganhou causa pobres	[negativo]	Não Existe
7	lucashnrque	05/10/2014 19:59	2	1	dilma perde segundo turno	[negativo]	Curitiba, Paraná
8	BeatriceZaretti	05/10/2014 19:59	0	0	toma porra to revoltada pais retardado q vota ...	[negativo]	
9	Chavalzada	05/10/2014 19:59	0	0	dilma aecio estao garantidos segundo turno	[negativo]	Chaval - Ceará
10	zenrdaya	05/10/2014 19:59	0	0	culpa nao dilma vc trouxa ir la protesto falar...	[negativo]	Sao Paulo, Brazil

Figura 3.11: Tela com o resultado das análises utilizando o modelo desenvolvido.

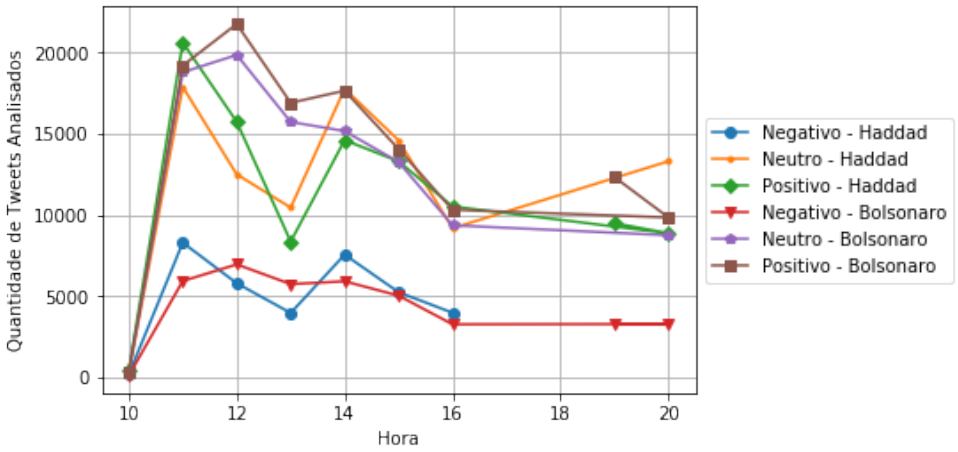


Figura 3.12: Evolução do sentimento referente a cada candidato ao longo do dia da votação.

verifica-se que as predições do modelo obtiveram resultado satisfatório. A utilização da geolocalização nessa etapa não ofereceu bons resultados, visto que a API do *Twitter* passou

Tabela 3.7: Classificação dos dados utilizando o modelo construído para o segundo turno das eleições presidenciais de 2018

	Positivo	Neutro	Negativo
Bolsonaro	54.47%	30.88%	14.63%
Haddad	41.38%	42.95%	15.66%

Tabela 3.8: Distribuição dos resultados da eleição presidencial de 2018

Bolsonaro	55,13%
Haddad	44,87%

por várias atualizações e limitou as requisições para 5,000 por dia, impossibilitando uma análise detalhada da localização de cada *tweet* postado. Outra limitação é que, por se tratar de redes sociais, as pessoas não têm uma obrigação legal de inserir a sua localização exata.

Capítulo 4

Conclusão

Neste trabalho foi desenvolvido um *framework* capaz de analisar dados extraídos de uma rede social utilizando técnicas de processamento de linguagem natural. Nosso objetivo foi explorar a análise de sentimento em textos em língua portuguesa e validar os dados, cruzando informações disponíveis pelo Tribunal Superior Eleitoral, órgão responsável pela condução das eleições no Brasil.

Verificou-se que, quanto melhor o pré-processamento dos dados, melhor será o resultado final. Durante o trabalho foram feitos vários ajustes nessa etapa para que o dado chegasse ao classificador da melhor maneira possível.

O *framework* proposto conseguiu prever, de forma eficaz, os resultados do segundo turno das eleições presidenciais de 2014. Grande parte da sociedade esteve crítica em relação à reeleição daquele governo, o que justifica o elevado número de *tweets* negativos naquele cenário. De fato, dois anos após a reeleição, a então presidente eleita, Dilma Rousseff, sofreu um processo de *impeachment*, validando todo o nosso estudo no que diz respeito à identificação de vários textos com polaridade negativa.

Ao comparar os resultados obtidos pela análise de sentimentos com os dados extraídos do Tribunal Superior Eleitoral, foi possível constatar que o *framework* apresentou bons resultados. Além disso, o trabalho pode ser aplicado em outras áreas além da política, pois tem o intuito de aquecer as pesquisas na área de análise de sentimentos no idioma português.

Após a utilização de quatro classificadores distintos para a criação de modelos de aprendizado de máquina, o algoritmo que apresentou os melhores resultados foi o SVM, com *accuracy* superior a 90validação do modelo. O algoritmo de árvore de decisões, por sua vez, apresentou o pior desempenho para a análise de sentimentos de textos provenientes de redes sociais.

De contribuições ao meio científico, entende-se que o desenvolvimento desse *framework* e do modelo treinado facilita pesquisas nessa áreas e um artigo foi publicado para que

outras pessoas possam replicar as etapas que foram aplicadas neste trabalho.

Capítulo 5

Trabalhos Futuros

Para trabalhos futuros, podemos adotar a técnica de *ensemble* com o intuito de otimizar a classificação de sentimentos de novos textos.

Otimizar a criação das matrizes utilizadas no *Bag-of-words* a fim de diminuir o tempo de treinamento e melhorar a qualidade de análise.

Pretende-se incorporar algoritmos de aprendizado de máquina não supervisionado, além da utilização de *deep learning*, afim de melhorar a classificação e entender ironias escritas no texto.

Explorar os conceitos matemáticos do algoritmos, através de técnicas tensoriais para diminuir a ordem do modelo.

Referências

[1] Speakt: *Top 10 languages used on the internet today.* https://speakt.com/top-10-languages-used-internet/, Acessado em 18.10.2018. x, 2

[2] Andreoni, Martin: *An intrusion detection and prevention architecture for software defined networking*, abril 2014. x, 15

[3] Eleitoral Tribunal Superior: *Estatísticas Eleitorais: eleições anteriores.* http://www.tse.jus.br/eleitor-e-eleicoes/estatisticas/eleicoes/eleicoes-anteriores/estatisticas-eleitorais-anos-anteriores, Acessado em 28.10.2018. x, 30, 33

[4] Song, M., M. C. Kim e Y. K. Jeong: *Analyzing the political landscape of 2012 Korean presidential election in Twitter.* IEEE Intelligent Systems, 29(2):18–26, 2014. 1, 4

[5] Araniti, G., I. Bisio e M. De Sanctis: *Towards the reliable and efficient interplanetary internet: A survey of possible advanced networking and communications solutions.* Em *2009 First International Conference on Advances in Satellite and Space Communications*, páginas 30–34, July 2009. 1

[6] IBGE: *https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/20077-nove-entre-dez-usuarios-de-internet-no-pais-utilizam-aplicativos-de-mensagens.* https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/20077-nove-entre-dez-usuarios-de-internet-no-pais-utilizam-aplicativos-de-mensage Acessado em 20.10.2018. 1

[7] Stats, Internet Live: *Elaboration of data by international telecommunication union (itu), united nations population division, internet mobile association of india (iamai), world bank.* http://www.internetlivestats.com/internet-users-by-country/. 1

[8] Miller, G. A.: *Wordnet: a lexical database for English.* Communications of the ACM, 38(11):39–41, 1995. 1, 25

[9] Twitter: *Twitter developer platform.* https://developer.twitter.com, Acessado em 20.08.2018. 2

[10] Comunicação, Empresa Brasil de: *Facebook chega a 127 milhões de usuários no brasil.* http://agenciabrasil.ebc.com.br/economia/noticia/2018-07/facebook-chega-127-milhoes-de-usuarios-no-brasil, Acessado em 20.10.2018. 3

[11] País, El: *Cambridge analytica, empresa pivô no escândalo do facebook, é fechada.* https://brasil.elpais.com/brasil/2018/05/02/internacional/1525285885_691249.html, Acessado em 20.10.2018. 3

[12] Souza, Marlo e Renata Vieira: *Sentiment analysis on twitter data for portuguese language.* Em *Proceedings of the 10th International Conference on Computational Processing of the Portuguese Language*, PROPOR'12, páginas 241–247, Berlin, Heidelberg, 2012. Springer-Verlag, ISBN 978-3-642-28884-5. http://dx.doi.org/10.1007/978-3-642-28885-2_28. 3

[13] Neuenschwander, Bruna, Adriano C.M. Pereira, Wagner Meira, Jr. e Denilson Barbosa: *Sentiment analysis for streams of web data: A case study of brazilian financial markets.* Em *Proceedings of the 20th Brazilian Symposium on Multimedia and the Web*, WebMedia '14, páginas 167–170, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-3230-9. http://doi.acm.org/10.1145/2664551.2664579. 3

[14] Loria, S.: *TextBlob: Simplified text processing.* http://textblob.readthedocs.io/en/dev/index.html. 3, 17

[15] Wagh, Rasika e Payal Punde: *Survey on sentiment analysis using twitter dataset.* Em *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, páginas 208–211. IEEE, 2018. 3

[16] Wang, Yuanhong, Yang Liu e Xiaohui Yu: *Collaborative filtering with aspect-based opinion mining: A tensor factorization approach.* Em *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, páginas 1152–1157. IEEE, 2012. 4

[17] Wegrzyn-Wolska, K. e L. Bougueroua: *Tweets mining for French presidential election.* Em *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*, páginas 138–143, Nov 2012. 4, 17

[18] Cerón-Guzmán, J. A. e E. León-Guzmán: *A sentiment analysis system of Spanish tweets and its application in Colombia 2014 presidential election.* Em *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, páginas 250–257, Oct 2016. 4

[19] Joyce, B. e J. Deng: *Sentiment analysis of tweets for the 2016 US presidential election.* Em *2017 IEEE MIT Undergraduate Research Technology Conference (URTC)*, páginas 1–4, Nov 2017. 4

[20] Nasrabadi, Nasser M: *Pattern recognition and machine learning.* Journal of electronic imaging, 16(4):049901, 2007. 6, 12

[21] Bonaccorso, Giuseppe: *Machine Learning Algorithms.* Packt Publishing Ltd, 2017. 6, 13

[22] Marks, II, Robert J., Jacek M. Zurada e Charles J. Robinson (editores): *Computational Intelligence: Imitating Life.* IEEE Press, Piscataway, NJ, USA, 1994, ISBN 0780311043. 6

[23] Mitchell, Thomas M.: *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1ª edição, 1997, ISBN 0070428077, 9780070428072. 7

[24] Chang, Chih Chung e Chih Jen Lin: *Libsvm: a library for support vector machines*. ACM transactions on intelligent systems and technology (TIST), 2(3):27, 2011. 8

[25] Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg *et al.*: *Scikit-learn: Machine learning in python*. Journal of machine learning research, 12(Oct):2825–2830, 2011. 8, 9, 27

[26] Vieira, Lucas Maciel, Clicia Grativol, Flavia Thiebaut, Thais G Carvalho, Pablo R Hardoim, Adriana Hemerly, Sergio Lifschitz, Paulo Cavalcanti Gomes Ferreira e Maria Emilia MT Walter: *Plantrna_sniffer: a svm-based workflow to predict long intergenic non-coding rnas in plants*. Non-coding RNA, 3(1):11, 2017. 9

[27] Lorena, Ana Carolina e André CPLF de Carvalho: *Uma introdução às support vector machines*. Revista de Informática Teórica e Aplicada, 14(2):43–67. 9, 27

[28] McCallum, Andrew e Kamal Nigam: *A comparison of event models for naive bayes text classification*, 1998. 10

[29] Zhang, Harry: *The optimality of naive bayes*, janeiro 2004. 11

[30] Coelho, Vinícius Coutinho Guimarães: *Análise de logs de interação em ambiente educacional corporativo via mineração de dados educacionais*. 11

[31] Breiman, Leo, J. H. Friedman, R. A. Olshen e C. J. Stone: *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984. 12

[32] MacQueen, James *et al.*: *Some methods for classification and analysis of multivariate observations*. Em *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1 de *1*, páginas 281–297. Oakland, CA, USA, 1967. 13

[33] Zurada, Jacek M.: *Introduction to Artificial Neural Systems*. PWS Publishing Co., Boston, MA, USA, 1st edição, 1999, ISBN 053495460X. 14

[34] Mazurowski, Maciej A, Piotr A Habas, Jacek M Zurada, Joseph Y Lo, Jay A Baker e Georgia D Tourassi: *Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance*. Neural networks, 21(2-3):427–436, 2008. 14

[35] Aalst, Wil MP Van der, Vladimir Rubin, HMW Verbeek, Boudewijn F van Dongen, Ekkart Kindler e Christian W Günther: *Process mining: a two-step approach to balance between underfitting and overfitting*. Software & Systems Modeling, 9(1):87, 2010. 14

[36] Kohavi, Ron *et al.*: *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Em *Ijcai*, volume 14, páginas 1137–1145. Montreal, Canada, 1995. 14

[37] Liddy, Elizabeth D: *Natural language processing*. 2001. 16

[38] Golbeck, Jennifer, Justin M Grimes e Anthony Rogers: *Twitter use by the us congress*. Journal of the American Society for Information Science and Technology, 61(8):1612–1621, 2010. 17

[39] Ferreira, Coutinho, Mariza Dosciatti e Emerson Paraíso: *Um método para o pré-processamento de textos na identificação automática de emoções para o português do brasil*, outubro 2013. 17

[40] Mcnamee, Paul e James Mayfield: *Character n-gram tokenization for european language text retrieval*. Information retrieval, 7(1-2):73–97, 2004. 17

[41] Bird, Steven e Edward Loper: *Nltk: the natural language toolkit*. Em *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, página 31. Association for Computational Linguistics, 2004. 17, 18

[42] Wallach, Hanna M: *Topic modeling: beyond bag-of-words*. Em *Proceedings of the 23rd international conference on Machine learning*, páginas 977–984. ACM, 2006. 18

[43] Salton, Gerard e Christopher Buckley: *Term-weighting approaches in automatic text retrieval*. Information processing & management, 24(5):513–523, 1988. 19

[44] Sparck Jones, Karen: *A statistical interpretation of term specificity and its application in retrieval*. Journal of documentation, 28(1):11–21, 1972. 19

[45] McKinney, Wes: *pandas: a foundational python library for data analysis and statistics*. Python for High Performance and Scientific Computing, páginas 1–9, 2011. 22

[46] Oliveira, Derick M de, Roberto CSNP Souza, Denise EF de Brito, Wagner Meira Jr, Gisele L Pappa e Belo Horizonte-MG-Brasil: *Uma estratégia não supervisionada para previsão de eventos usando redes sociais.* Em *SBBD*, páginas 137–148, 2015. 22, 24

[47] Sharma, Anuj e Shubhamoy Dey: *A comparative study of feature selection and machine learning techniques for sentiment analysis*. Em *Proceedings of the 2012 ACM Research in Applied Computation Symposium*, RACS '12, páginas 1–7, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1492-3. `http://doi.acm.org/10.1145/2401603.2401605`. 24

[48] Souza, M., R. Vieira, D. Busetti, R. Chishman e I. M. Alves: *Construction of a portuguese opinion lexicon from multiple resources*. Em *Proceedings of the 8th Brazilian Symposium in Information and Human Language Technology*, 2011. 25

[49] Roesslein, Joshua: *tweepy documentation*. Online] http://tweepy. readthedocs. io/en/v3, 5, 2009. 26

[50] Jin, Yaochu e Bernhard Sendhoff: *Pareto-based multiobjective machine learning: An overview and case studies*. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38(3):397–415, 2008. 27

[51] Medhat, Walaa, Ahmed Hassan e Hoda Korashy: *Sentiment analysis algorithms and applications: A survey*. Ain Shams Engineering Journal, 5(4):1093 – 1113, 2014, ISSN 2090-4479. http://www.sciencedirect.com/science/article/pii/S2090447914000550. 28

[52] São Paulo, Folha de: *Entenda a operação lava jato, da polícia federal*. https://www1.folha.uol.com.br/poder/2014/11/1548049-entenda-a-operacao-lava-jato-da-policia-federal.shtmls, Acessado em 20.10.2018. 32

Apêndice A

A.1 Código para limpeza dos dados

```
import pandas as pd
import numpy as np
from nltk.tokenize import WordPunctTokenizer
tok = WordPunctTokenizer()

# çãFuno utilizada para limpeza do texto utilizando ãexpresso regulares
def removestopwords(texto):
    frases = []

    for palavras in texto:

        palavras = palavras.lower()
        palavras = palavras.replace('?',' ')
        palavras = palavras.replace('#','')
        palavras = palavras.replace('!',' ')
        palavras = palavras.replace('%',' ')
        palavras = palavras.replace('. ',' ')
        palavras = palavras.replace(')',' ')
        palavras = palavras.replace('(',' ')
        palavras = palavras.replace('-',' ')
        palavras = palavras.replace(',',' ')
        palavras = palavras.replace('/ ',' ')
        palavras = palavras.replace('*',' ')
        palavras = palavras.replace('=',' ')
        palavras = palavras.replace(':','')

        palavras = palavras.replace('r\$',' ')

        palavras = palavras.replace('á','a')
        palavras = palavras.replace('à','a')
```

```
        palavras = palavras.replace('â','a')
        palavras = palavras.replace('ã','a')

        palavras = palavras.replace('é','e')
        palavras = palavras.replace('è','e')
        palavras = palavras.replace('ê','e')


        palavras = palavras.replace('í','i')
        palavras = palavras.replace('ì','i')
        palavras = palavras.replace('î','i')

        palavras = palavras.replace('ó','o')
        palavras = palavras.replace('ò','o')
        palavras = palavras.replace('ô','o')
        palavras = palavras.replace('õ','o')

        palavras = palavras.replace('ú','u')
        palavras = palavras.replace('ù','u')
        palavras = palavras.replace('û','u')

        palavras = palavras.replace('ç','c')

        frases.append(palavras)


    return frases

dataset['text'] = removestopwords(dataset['text'])
```

A.2 Código para análise de sentimentos usando dicionário

```

dicionario = open("dicionario.txt", 'r')


dic_palavra_polaridade = {}

# Neste trecho íatribumos a cada palavra uma Polaridade
for i in dicionario.readlines():
```

```
    pos_ponto = i.find('.')
    palavra = (i[:pos_ponto])
    pol_pos = i.find('POL')
    polaridade = (i[pol_pos+4:pol_pos+6]).replace(';','')
    dic_palavra_polaridade[palavra] = polaridade


# Altrando os valores pelos órtulos
def çãPontuao_sentimento(frase):
    frase = frase.lower()
    l_sentimento = []

    for p in frase.split():
        l_sentimento.append(int(dic_palavra_polaridade.get(p, 0)))

    if sum(l_sentimento) > 0:
        return 'Positivo'
    elif sum(l_sentimento) == 0:
        return 'Neutro')
    elif sum(l_sentimento) < 0:
        return 'Negativo'


# Realizando os testes de áanlise com o ádicionrio
frase = çãPontuao_sentimento('Sua ãme é feliz')
print(frase)
```

A.3 Código para extração da localização do usuário

```
    # coding: utf-8
    import pandas as pd
    import tweepy

    # Dados de acesso do Twitter
    consumer_key = ""

    consumer_secret = ""

    access_token_secret = ""

    access_token = ""

```

```
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)

    # Logando na API
    api = tweepy.API(auth)

    location = []

    # Realizando a çãextrao da çãlocalizao
    for tweet in entrada['username']:
        try:
            user = api.get_user(tweet)
            print(user.location)
            location.append(user.location)
        except tweepy.TweepError as e:
            location.append('Error')

    entrada['local'] = location
    entrada.to_csv(saida, sep=';', mode='a',index=False)
```

A.4 Código para geração da séries temporais

```
import pandas as pd      # To handle data
import numpy as np       # For number computing

# For plotting and visualization:
from IPython.display import display
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline


# Abrindo os datasets
dataset=pd.read_csv("/home/bruno/Documents/artigo/Dados Brutos/
    Dilma_final_prog1.csv",delimiter=",",encoding='latin-1').set_index('Date
    ')
dataset1=pd.read_csv("/home/bruno/Documents/artigo/Dados Brutos/
    Aecio_final_prog1.csv",delimiter=",",encoding='latin-1').set_index('Date
    ')

# Definindo os órtulos do dataset
positivo = dataset.loc[dataset['Sentiment'] == 'Positive']
```

```
negativo = dataset.loc[dataset['Sentiment'] == 'Negative']
neutro  = dataset.loc[dataset['Sentiment'] == 'Neutral']

positivo1 = dataset1.loc[dataset1['Sentiment'] == 'Positive']
negativo1 = dataset1.loc[dataset1['Sentiment'] == 'Negative']
neutro1  = dataset1.loc[dataset1['Sentiment'] == 'Neutral']

positivo.index = pd.to_datetime(positivo.index)
negativo.index = pd.to_datetime(negativo.index)
neutro.index = pd.to_datetime(neutro.index)


positivo1.index = pd.to_datetime(positivo1.index)
negativo1.index = pd.to_datetime(negativo1.index)
neutro1.index = pd.to_datetime(neutro1.index)

positivo = positivo.rename(columns={"Sentiment":"Positive - Dilma"})
positivo1 = positivo1.rename(columns={"Sentiment":"Positive - Aecio"})
positivo_plot = positivo['Positive - Dilma'].groupby( [positivo.index.day] ).count()
positivo_plot1 = positivo1['Positive - Aecio'].groupby( [positivo1.index.day] ).count()

negativo = negativo.rename(columns={"Sentiment":"Negative - Dilma"})
negativo1 = negativo1.rename(columns={"Sentiment":"Negative - Aecio"})
negative_plot = negativo['Negative - Dilma'].groupby( [negativo.index.day] ).count()
negative_plot1 = negativo1['Negative - Aecio'].groupby( [negativo1.index.day] ).count()

neutro = neutro.rename(columns={"Sentiment":"Neutral - Dilma"})
neutral_plot = neutro['Neutral - Dilma'].groupby( [neutro.index.day] ).count()
neutro1 = neutro1.rename(columns={"Sentiment":"Neutral - Aecio"})
neutral_plot1 = neutro1['Neutral - Aecio'].groupby( [neutro1.index.day] ).count()

#çãCriao do áGrfico Temporal
import matplotlib.dates as mdates
negative_plot1.plot(marker='o')
neutral_plot1.plot(marker='.')
positivo_plot1.plot(marker='D')


negative_plot.plot(marker='v')
```

```
neutral_plot.plot(marker='p')
positivo_plot.plot(marker='s')

plt.ylabel("Sentiment Count")
plt.xlabel("Day")
plt.legend()
plt.grid(True)
```

A.5 Código para utilização do modelo de aprendizado de máquina persistivo

```
import pandas as pd
import numpy as np
from sklearn.externals import joblib


# importando o modelo treinado previamente
model = joblib.load('svm.pkl')

bolsonaro = pd.read_csv('bolsonaro_use.csv',sep=';',encoding='utf-8')
haddad = pd.read_csv('haddad_use.csv',sep=';',encoding='utf-8')


bolsonaro['text'] = palavras
palavras1 = []

for index, row in bolsonaro.iterrows():
    palavras1.append(' '.join(bolsonaro['text'][index].split()))

bolsonaro['text'] = removestopwords(palavras1)
bolsonaro['text'] = bolsonaro['text'].str.strip()

sentiment = []
for tweet in bolsonaro['text']:
    sentiment.append(model.predict([tweet]))
bolsonaro['sentiment'] = sentiment



haddad['text'] =  removestopwords(palavras)
palavras1 = []
```

```
for index, row in haddad.iterrows():
    palavras1.append(' '.join(haddad['text'][index].split()))

haddad['text'] = palavras1
haddad['text'] = haddad['text'].str.strip()

sentiment = []
for tweet in haddad['text']:
    sentiment.append(model.predict([tweet]))
haddad['sentiment'] = sentiment
```