Lorena Santos Pereira

Processo de Mineração Textual para Reconhecimento de Mensagens com Tendência a Discurso de Ódio no *Twitter*

Salvador

Junho de 2017

Lorena Santos Pereira

Processo de Mineração Textual para Reconhecimento de Mensagens com Tendência a Discurso de Ódio no *Twitter*

Monografia apresentada ao curso de Sistemas de Informação da Universidade do Estado da Bahia - UNEB, como requisito para a obtenção do grau de Bacharel, sob a orientação do Professor Doutor Eduardo Manuel de Freitas Jorge

Universidade do Estado da Bahia

Departamento de Ciências Exatas e da Terra I

Colegiado de Sistemas de Informação

Orientador: Prof. Dr. Eduardo Manuel de Freitas Jorge

Salvador

Junho de 2017

Lorena Santos Pereira

Processo de Mineração Textual para Reconhecimento de Mensagens com Tendência a Discurso de Ódio no *Twitter*

Monografia apresentada ao curso de Sistemas de Informação da Universidade do Estado da Bahia - UNEB, como requisito para a obtenção do grau de Bacharel, sob a orientação do Professor Doutor Eduardo Manuel de Freitas Jorge

Trabalho aprovado. Salvador, 20 junho de 2017:

Prof. Dr. Eduardo Manuel de Freitas Jorge
Orientador

Profa. MSc. Trícia Souto
Membro da Banca

Prof. Ph.D Diego Gervasio Frías Suárez
Membro da Banca

Salvador
Junho de 2017

Esse trabalho é dedicado ao meu pai, ao meu irmão e as mulheres que me deram a vida e a essência: Mãe, Madrinha e Avó.

Agradecimentos

Agradeço a Deus por sempre ter me sustentado.

À minha família pela formação, apoio, amor e exemplo de superação, tendo em vista de onde viemos e tudo que passamos.

Ao meu companheiro e namorado Mário Bortoli por todo carinho, cuidado e atenção, sempre me motivando a seguir em frente.

A todos os meus amigos por estarem ao meu lado em algum momento da minha vida, pela ternura e pela experiência que foi ter por perto cada um deles. Em especial aos amigos oriundos do grupo de pesquisa ACSO, que foram essenciais nessa jornada. Em especial também aos amigos Rogério do Carmo, Thereza Barros, Luiz Caribé, Débora Oliveira, Matheus Moreira, Thiago Alvoravel, Calison Ribeiro, Rafael Guimarães e Íris Ribeiro que passaram comigo os momentos finais e mais intensos do curso e que me proporcionaram um aprendizado que levarei comigo para sempre.

Ao meu orientador Eduardo Jorge por todo incentivo, ensinamentos e paciência.

A todos os educadores que tive ao longo da vida por contribuir imensamente para a minha formação acadêmica, profissional e pessoal.

À todas as pessoas que corroboraram de alguma forma para que eu chegasse até aqui e pudesse concluir essa etapa.

"Vou aprender a ler pra ensinar meus camaradas"
(Yá Yá Massemba canção de Roberto Mendes interpretada por Maria Bethânia)

Resumo

As Redes Sociais Online têm um papel importante no cenário atual da Internet, onde existe uma grande base de dados de forma descentralizada e desestruturada. Redes desse tipo proporcionam uma integração virtual entre os indivíduos que as utilizam, principalmente através do compartilhamento de conteúdo diverso. A partir dessa facilidade de disseminação de conteúdo, positivo ou negativo, é visto frequentemente a intolerância e propagação de discurso de ódio nas redes, prática nociva para a sociedade como um todo por dificultar as relações interpessoais e ferir a dignidade da pessoa humana. A rede social Twitter apresenta uma característica que a destaca dentre as demais, que é possibilidade de indicar quais os assuntos mais populares em determinada região quase em tempo real a partir das mensagens publicadas por seus usuários. Partindo desse cenário o objetivo do trabalho foi propor, implementar e avaliar um processo capaz de extrair dados do *Twitter* e identificar as mensagens que tendem a discurso de ódio utilizando as etapas da Mineração de Texto. Foram analisados dois tipos de discurso de ódio: sobre gênero e sobre etnia. Após coleta e pré-processamento, as mensagens foram submetidas a tarefa de classificação utilizando o algoritmo de Gaussian Naive Bayes da biblioteca de aprendizagem de máquina *Sklearn*. O classificador teve uma acurácia de 95% para discurso de ódio sobre etnia e 76% para discurso de ódio sobre gênero.

Palavras-chave: Mineração de Texto. Discurso de ódio. Classificação.

Abstract

Online Social Networks play an important role in the current Internet scenario, where there is a large decentralized and unstructured data base. Networks of this kind provide virtual integration between the individuals who use them, mainly through the sharing of diverse content. From this ease of positive or negative content dissemination, is viewed frequently the intolerance and propagation of hate speech in networks, a practice harmful to society as a whole by hindering interpersonal relations and hurting the dignity of the human person. The social network Twitter presents a feature that is a customer of highlights like others, which is the possibility to indicate which are the most popular subjects in a region almost in real time from the messages published by its users. From the scenario described, the objective of this work was to propose, implement and evaluate a process capable of extracting data from Twitter and identifying messages with trend to hate speech using the steps of Text Mining. Two kind of hate speech were analyzed: about gender and about ethnicity. After collecting and preprocessing, the messages were submitted to the classification task using the Gaussian Naive Bayes algorithm of the machine learning library Sklearn. The classifier had an accuracy of 95% for hate speech about ethnicity and 76% for hate speech about gender.

Keywords: Text Mining. Hate speech. Classification.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

RSO	Rede Social Online
API	Application Programming Interface
URL	Uniform Resource Locator
ONG	Organização Não Governamental
DO	Discurso de Ódio

Sumário

1 Introdução

A *Web* Social (ou participativa) foi uma evolução da Internet marcada pela disseminação dos blogs e mídias sociais, onde o usuário deixou de ser apenas consumidor e passou a ser também produtor de conteúdo (VICENTIM, 2013).

Toda essa base de informações disponível na Web encontra-se de forma descentralizada e desestruturada, dificultando a extração de conhecimento sobre a mesma. As Redes Sociais Online (RSOs) são algumas das principais responsáveis pela facilidade de compartilhamento de conteúdo. Redes como o *Facebook*, *Twitter* e *LinkedIn* enfatizam a integração virtual entre os indivíduos que as utilizam, possibilitando a cada um deles a criação de um perfil que os represente, com fotos, descrições e competências, conexões com pessoas de todo o mundo e compartilhamento de dados multimídia com essas pessoas. O conteúdo compartilhado vai desde uma simples mensagem de 140 caracteres até o mais novo vídeo de um artista no *Youtube*.

O *Twitter* possui uma característica que se destaca entre as outras redes, seus *Tweets* (mensagens enviadas pelos usuários com limite máximo de 140 caracteres) funcionam como um insumo para indicar os assuntos mais populares, em determinada região, quase que em tempo real. Um exemplo disso é a forma como um acontecimento pode intensificar as trocas de mensagens sobre um determinado assunto na rede, conforme o caso dos protestos políticos em março de 2015 (ARAGÃO, 2015).

Extrair conhecimento das RSOs não é uma tarefa trivial, envolve uma série de estudos, dentre eles: o método mais apropriado, as limitações dos serviços oferecidos pela rede escolhida, as formas de armazenamento e processamento dos dados coletados, a identificação dos padrões desejados e como representar o conhecimento adquirido de forma compreensível e capaz de apoiar uma tomada de decisão ou desempenhar apenas função informativa.

Algumas áreas de estudo são essenciais para a realização dessa tarefa, como: Mineração de Texto, que através de etapas definidas possibilita a extração de conhecimento de base de dados textuais; e Processamento de Linguagem Natural, que apresenta técnicas para auxiliar no processamento de informações em linguagem humana, visando a compreensão automática das mesmas.

Uma outra característica das RSOs é a capacidade de disseminação de ideias, positivas e/ou negativas, em larga escala. Atualmente a intolerância nas redes sociais vem sendo objeto de estudo de alguns trabalhos como NOVA/SB (2016b) e Silva et al. (2011), e também razão da criação de grupos que lutam contra isso, como o Movimento Contra o

Discurso de Ódio[1], uma campanha do Setor de Juventude do Conselho da Europa, que alcança jovens de várias partes do mundo que tem como principal objetivo combater esse tipo de discurso e sua expressão online.

> "Na busca de um conceito operacional para o discurso do ódio (hate speech), observa-se que tal discurso apresenta como elemento central a expressão do pensamento que desqualifica, humilha e inferioriza indivíduos e grupos sociais. Esse discurso tem por objetivo propagar a discriminação desrespeitosa para com todo aquele que possa ser considerado "diferente", quer em razão de sua etnia, sua opção sexual, sua condição econômica ou seu gênero, para promover a sua exclusão social." (FREITAS; CASTRO, 2013)

A disseminação desse tipo de discurso é nociva para a sociedade como um todo, pois perde-se a possibilidade de discutir de forma crítica determinados assuntos, cria-se uma tensão nas relações entre pessoas de opiniões diferentes, e muitas vezes o direito à liberdade de expressão é confundido com a externalização de opiniões que visam apenas degradar uma pessoa ou grupo de pessoas que compartilhem uma característica.

Partindo da grande quantidade de dados dispostos na WEB, a exemplo das opiniões e sentimentos expressos em redes sociais, e tendo em vista a existência e o caráter prejudicial do discurso de ódio disseminado nessas redes. Surge a seguinte questão de pesquisa: Que processo torna possível a extração de dados do *Twitter* para reconhecimento de discurso de ódio?

Sendo assim, o objetivo da pesquisa foi propor, implementar e avaliar um processo capaz de extrair dados do *Twitter* e identificar as mensagens que tendem a discurso de ódio utilizando as etapas da Mineração de Texto.

Como estratégia para a confecção do processo, alguns objetivos específicos foram estabelecidos: (i) Descrever um processo para identificação de termos que denotem tendência a discurso de ódio; (ii) Desenvolvimento de um extrator e pré-processador para os dados do *Twitter*; e (iii) Desenvolvimento de um módulo para classificação de *tweets*.

A classificação de *tweets* não é uma tarefa producente para seres humanos visto a grande quantidade de mensagens e sua variabilidade, assim, uma solução computacional que faça a extração e mineração dessas informações, classifique as mensagens de forma a facilitar a aquisição de conhecimento sobre esse domínio e o combate a esse tipo de discurso será a contribuição desse trabalho.

Ao realizar a identificação de mensagens que apresentem tendências ao discurso de ódio, esse trabalho adquire uma relevância para Organizações Não Governamentais ONGs, Governos e outras organizações interessadas nesse assunto, que poderiam se beneficiar desse processo para examinar as opiniões dispostas a fim de estabelecer programas de

[1] http://www.odionao.com.pt/

conscientização e campanhas sociais como uma forma para combater as discriminações que podem ser reflexos desses discursos.

Essa monografia está organizada em 7 capítulos. O capítulo 1 realiza uma introdução sobre o trabalho. Os capítulos 2 e 3 apresentam o referencial teórico sobre as redes sociais online e o processo de extração de dados, assim como os principais conceitos sobre mineração de dados. Os trabalhos correlatos estão dispostos no capítulo 4. O capítulo 5 descreve a solução implementado. O capítulo 6 apresenta os resultados obtidos. E por fim, o capítulo 7 expõe as conclusões do trabalho.

2 Redes Sociais Online

Esse capítulo apresenta definições e características das RSOs, técnicas para extração de dados dessa fonte e características do discurso de ódio disseminado nessas redes.

2.1 Definição e Características

As RSOs são redes que agrupam pessoas e seus relacionamentos, de forma que elas possam interagir a partir de qualquer tipo de mídia: link, vídeo, imagem, texto ou som (BENEVENUTO; ALMEIDA; SILVA, 2011).

Ellison et al. (2007) descreve as RSOs como um serviço baseado na Web que permite ao usuário:

- A criação de perfis públicos ou semipúblicos;
- A articulação de uma lista de outros usuários com os quais ele deseja se conectar;
- A visualização das suas conexões e das conexões dos outros usuários.

Além disso, essas redes têm alguns conceitos em comum:

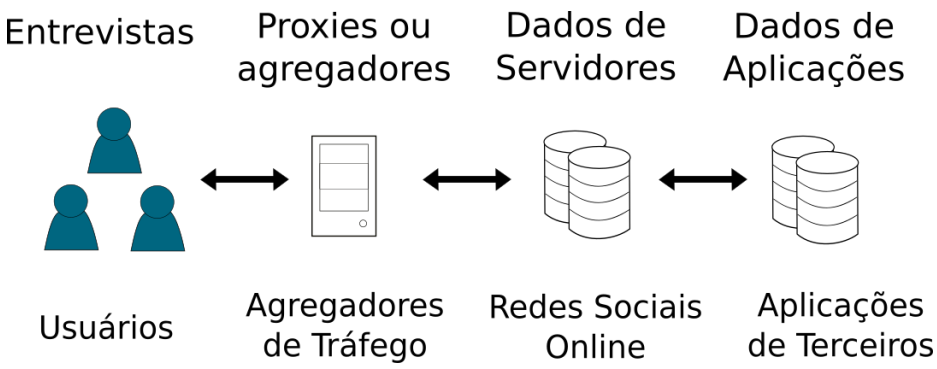
- Usuários: Pessoas que possuem um perfil na rede, podendo esse perfil representar apenas uma pessoa ou uma organização;
- Perfil: Página que representa o usuário na rede e apresenta as informações cadastradas pelo mesmo, como nome, descrição, e-mail, foto, profissão, interesses, localização, entre outras;
- Postagem: Conteúdo que o usuário publica na rede. As postagens podem receber comentários de usuários diversos, de acordo com a privacidade configurada pelo autor da postagem, ou algum tipo de avaliação, como o botão "curtir" do *Facebook*. As redes *Twitter* e *LinkedIn* possuem um mecanismo parecido. Também é possível que os usuários compartilhem as postagens de outros usuários, de forma que apareça a mensagem original e o seu autor, além de algum comentário feito pela pessoa que compartilhou a postagem;
- Linha do tempo ou *Timeline*: Página que contém o registro das atividades e/ou postagens do usuário ao longo do tempo;
- *Feed* de Notícias ou *Newsfeed*: Parte da página inicial do usuário que fica em constante atualização com as postagens dos outros usuário vinculados a ele na rede;

O *Twitter*, rede escolhida para o desenvolvimento desse trabalho, apresenta algumas peculiaridades em relação às outras redes, como os conceitos de: *Tweet* é o nome dado a postagem dessa rede. Uma mensagem de no máximo 140 caracteres; *Hashtag* é uma forma de indexar palavras chaves, onde essas palavras devem ser precedidas pelo símbolo # também chamado de cerquilha. O objetivo do *Twitter* é permitir que as pessoas sigam facilmente tópicos do seu interesse, pois ao clicar na *Hashtag* abre-se uma página listando as principais postagens que utilizaram a mesma (TWITTER, 2017a); *Retweet* é o compartilhamento de um *Tweet* (TWITTER, 2017b) e funciona como um indicador de popularidade para o usuário autor do *Tweet*.

2.2 Extração de dados em Redes Sociais Online

A extração dos dados disponíves nas RSOs dados pode ser feita de várias formas. A Figura 1 apresenta como se dividem as técnicas que podem ser utilizadas.

Figura 1: Formas possíveis de coleta de dados



Fonte: Adaptado de Benevenuto, Almeida e Silva (2011)

Benevenuto, Almeida e Silva (2011) apresentam que as formas de coleta de dados podem ser agrupadas de acordo com o recurso utilizado para a mesma:

1. **Entrevistas:** O método feito a partir dos próprios usuários da RSO consiste em entrevistas estruturadas feitas por pesquisadores no intuito de coletar dados do usuário que não estão claramente ditos na rede social como: motivação de uso e diferenças etnográficas;

2. **Pontos intermediários (*Proxies* ou Agregadores):** A coleta feita sobre pontos intermediários consiste na coleta sobre os Provedores de Serviço a Internet ISPs filtrando os pacotes referentes a uma determinada RSO, ou, sobre agregadores de redes sociais: aplicações que permitem ao usuário efetuar login em várias redes sociais

e utilizá-las a partir de uma única interface, sem a necessidade de acessar as redes separadamente;

3. **Dados de Aplicações:** Com a possibilidade de aplicação de terceiros rodar sobre a plataforma de uma rede social, a exemplo do *Facebook* onde ao utilizar uma dessas aplicações, o cliente faz uma requisição ao servidor da rede social, que repassa a requisição ao servidor de aplicação, o servidor de aplicação processa a requisição e responde de volta para o servidor da rede social, que repassa para o cliente (NAZIR et al., 2009 apud BENEVENUTO; ALMEIDA; SILVA, 2011). A coleta de dados através dessas aplicações pode trazer informações acerca da interatividade de um determinado usuário, assim como sua lista de amigos e outras informações referentes as permissões dadas no momento em que o usuário vincula sua conta à aplicação;

4. **Dados de Servidores da RSO:** Os servidores das RSO's são responsáveis por dispor todo o ambiente necessário para a utilização da mesma. Assim, poder coletar os dados que são armazenados nos servidores seria a forma mais efetiva de coleta de dados, uma vez que eles armazenam as informações de todos os usuários daquela rede. A dificuldade em acessar diretamente os servidores, por uma questão de segurança e disponibilidade do serviço, originou a utilização dos *crawlers* ou robôs para extração de dados públicos sobre as páginas das redes.

   4.1. **Coleta por amostragem:** Visto a dificuldade de coletar dados sobre todos os usuários da rede, uma alternativa é a coleta parcial, sobre uma amostragem da rede. Segundo Benevenuto, Almeida e Silva (2011) uma estratégia a ser utilizada é a *snowball* (bola de neve) que "consiste em coletar o grafo de uma rede social seguindo uma busca em largura". A busca começa a partir de um nó inicial de nível 0, expande-se os vizinhos daquele nó, muda-se o nó atual para o primeiro vizinho expandido de nível 1, e partir de cada vizinho do nível 1, expande-se os vizinhos respectivos de cada nó. Benevenuto, Almeida e Silva (2011) ressaltam que uma coleta por amostragem pode gerar resultados tendenciosos, assim os dados coletados precisam ser tradados e analisados cuidadosamente;

   4.2. **Coleta em larga escala:** Já a coleta em larga escala utiliza coletores distribuídos em máquinas distintas, devido ao processamento necessário e para evitar que os servidores identifiquem a extração de dados públicos como um ataque a rede;

   4.3. **Coleta por inspeção de identificadores:** São as coletas em sistemas que atribuem um identificador único, numérico e sequencial a cada usuário, o que possibilita apenas percorrer a sequencia de identificadores para coletar os dados, sem a necessidade de explorar a lista de amigos de cada usuário;

4.4. **Coleta em tempo real:** A coleta em tempo real busca analisar como os usuários interagem e o comportamento do fluxo de informações, a partir de ferramentas disponíveis, como bibliotecas e APIs por exemplo;

4.5. **Ferramentas e Bibliotecas:** São coletas baseadas na utilização de bibliotecas específicas que auxiliam no desenvolvimento de um coletor próprio, um programa capaz de realizar requisições ao sistema que gerenciam os dados dos usuários. Ou, na utilização de ferramentas que simplificam esse acesso como a API Console[1] do *Twitter*, que possibilita a qualquer um de seus usuários fazer consultas sobre o conteúdo da rede a partir de uma página na Internet;

4.6. **Utilizando API's:** Uma API de redes sociais "é um conjunto de tipos de requisições HTTP juntamente com suas respectivas respostas" (BENEVENUTO; ALMEIDA; SILVA, 2011). Na maioria dos casos a própria empresa que gerencia a rede desenvolve a API e a torna disponível para os usuários que desejam coletar dados ou desenvolver aplicações para a mesma. Assim, a API oferece um conjunto de rotinas prontas para serem utilizadas na interação com a rede e na pesquisa sobre os principais elementos da rede, como: informações do usuário, lista de amigos, mensagens recentes, entre outros. Uma outra característica é que as requisições retornam os resultados de forma estruturada, em formatos como: JavaScript Object Notation JSON[2] ou eXtensible Markup Language XML.

2.3 Discurso de Ódio nas Redes Sociais Online

A comunicação é parte inerente da vida do ser humano, principalmente da vida em sociedade, visto a necessidade de colaboração, resolução de conflitos, disseminação de informações e estabelecimento das relações interpessoais. Sendo assim, a liberdade de poder se expressar acerca de qualquer assunto, sem aplicação de censura, se torna um direito fundamental na vida dos indivíduos e de vasta utilização nas redes sociais. Porém, esse direito pode se contrapor a outro tão importante quanto: A dignidade da pessoa humana, que é um dos fundamentos da Constituição brasileira. Segundo (THEOPHILO, 2015) a dignidade da pessoa humana é um conceito que aborda, para todo e qualquer ser humano, a garantia da satisfação das necessidades essenciais a sua subsistência enquanto pessoa, tanto no aspecto físico quanto moral.

Nesse cenário, observa-se uma problemática entre os limites da liberdade de expressão e a disseminação do discurso de ódio, uma vez que o mesmo infringe diretamente o princípio da dignidade da pessoa humana, uma vez que esse tipo de discurso "carrega em seu conteúdo aspectos negativos que tem como principal objetivo trazer contra aquela

[1] https://dev.twitter.com/rest/tools/console
[2] Linguagem derivada do JavaScript que possui uma formatação mais leve para troca de dados.

ou aquilo sobre se (sic) discursa hostilização e degrado"(THEOPHILO, 2015). De acordo com Freitas e Castro (2013) e Theophilo (2015) o DO tem como objetivo maior encorajar a humilhação, a discriminação ou até mesmo a violência contra uma pessoa ou grupo de pessoas que compartilhem determinada característica ou crença, visando sua exclusão social.

Ainda com relação a descrição do DO Silva et al. (2011) afirma que o mesmo é constituído basicamente por 2 elementos: Discriminação e Externalização. Discriminação por manifestar um discurso que visa a degradação de uma pessoa ou grupo de pessoa com base em uma característica comum; E externalização visto que a existência do discurso de ódio exige a transferência do que antes era apenas uma ideia ou sentimento (do plano abstrato) para o plano físico, através da materialização e publicação dessa ideia (via escrita ou fala).

A pertinência da análise do DO nas redes sociais é devido ao fato da comunicação pela Internet, para alguns, ainda ser considerado um sinônimo para a anonimidade e sendo assim muitos fazem afirmações na rede sem pensar nas consequências e no alcance que elas podem ter, um exemplo é o caso da jovem que foi condenada a prestação de serviços sociais e pagamento de multa por ter postado no *Twitter* a seguinte frase "Nordestisto (sic) não é gente. Faça um favor a SP: mate um nordestino afogado!""(UOL, 2012).

O DO pode ser voltado a diferentes características como: etnia, gênero, opção sexual, religião ou nacionalidade entre outros. A partir das definições aqui apresentadas o presente trabalho buscou estabelecer um processo capaz de reconhecer o DO referente a etnia, direcionado a negros, e referente à questão de gênero, direcionado às mulheres.

Para montar a base necessária, esse trabalho coletou os dados através de bibliotecas e da API do *Twitter*, visto que essas técnicas apresentam um conjunto de rotinas passíveis de serem utilizadas para pesquisar sobre os principais elementos da rede em questão, como: lista de amigos, seguidores, mensagens recentes entre outros, o que facilitou a construção do extrator de dados necessário para o trabalho.

3 Mineração de Dados

Nesse capítulo serão apresentados os principais conceitos sobre mineração de dados, mineração de texto e mais especificamente sobre a tarefa de classificação e sua avaliação, dentro do contexto de mineração de dados.

Tan, Steinbach e Kumar (2009) apresentam a Mineração de Dados ou *Data Mining* como "Uma tecnologia que combina métodos tradicionais de análise de dados com algoritmos sofisticados para processar grandes volumes de dados.". Essa área possibilita para os pesquisadores a exploração do potencial de informações antes rotuladas como obsoletas ou com um papel secundário, em seu respectivo domínio.

Segundo Liu (2007) "Mineração de dados é comumente definida como o processo de descobrimento de padrões e conhecimentos úteis, a partir de fontes de dados como: banco de dados(tradicionais), textos, imagens e a WEB."(tradução nossa). Uma das principais características dessa área é sua multidisciplinariedade, pois para alcançar seus objetivos de encontrar padrões e conhecimentos aplicáveis é preciso usar técnicas de aprendizagem de máquina, recuperação e visualização de informações, estatística, inteligência artificial entre outras áreas.

A mineração de Dados possui uma série de tarefas e cada uma delas se aplica a um determinado contexto. Larose (2005) descreve essas tarefas da seguinte forma:

- **Descrição:** Tem por objetivo descrever padrões e tendências.
- **Classificação:** Analisa os dados para através das características semelhantes organizá-los em categorias pré-definidas. Por exemplo: avaliar se uma transação com o cartão de crédito é fraudulenta ou não.
- **Estimativa:** Consiste em estimar um valor baseado na análise de dados anteriores ou índices semelhantes. Por exemplo: estimar a média das notas de um estudante de pós graduação com base na suas notas de graduação.
- **Predição:** A partir da análise dos dados históricos de um contexto essa tarefa realiza uma estimativa futura, uma previsão. Como o aumento ou queda da bolsa de valores.
- **Agrupamento ou *Clustering*:** Realiza o agrupamento de dados de acordo com suas características semelhantes. Difere da classificação pois não possui categorias pré-definidas. É responsabilidade do algoritmo procurar sobre o conjunto de dados e organizá-los de acordo com nível de semelhança.

- **Associação:** A associação trabalha sobre a correlação das variáveis. Também é conhecida como análise de afinidades ou análise de cesta de mercado. Um exemplo é tentar relacionar quais são os produtos que estando próximos um do outro no mercado, tem mais chances de serem levados pelo consumidor

Segundo Liu (2007) uma vez obtida a base de dados o processo de mineração pode ser realizado em 3 etapas principais:

- **Pré processamento:** Essa é a etapa responsável por retirar ruídos ou dados não relevantes do conjunto original, a exemplo de itens repetidos, itens com erros de ortografia, ou base de dados com informações pessoais do usuário que não são relevantes para a correlação que se deseja descobrir. É necessário normalizar os dados de acordo com o contexto em que o trabalho está inserido, para que nenhuma das possíveis inconsistências interfira no processamento do algoritmo de mineração, o que pode gerar um resultado tendencioso ou incorreto;
- **Mineração de Dados:** Processamento dos dados através do algoritmo da respectiva tarefa de mineração que se deseja realizar;
- **Pós processamento:** Etapa de avaliação do resultado do processamento, visando identificar a relevância e a aplicação do padrão ou conhecimento encontrado. Nessa fase as técnicas de visualização de dados podem ser essenciais para facilitar o entendimento e a tomada de decisão.

3.1 Mineração de Texto

Devido a relevância e aplicabilidade da Mineração Textual, assim como Mineração na Web (que é a mineração voltada para dados dispostos na WEB, considerando sua estrutura), essas áreas derivadas da Mineração de Dados ganharam seu espaço dedicado de estudo, com desafios e técnicas característicos.

A Mineração de Texto (MT) é o processo de descoberta de dados em textos não estruturados concebidos na linguagem natural (TAN et al., 1999). Tan et al. (1999) também afirma que por texto ser a forma mais habitual de armazenamento de informação a mineração de texto tem um potencial comercial maior do que a mineração de dados em banco de dados tradicionais. A MT busca resolver as mesmas tarefas que a MD, apenas sua base que se torna diferente. Assim como, possui a mesma característica de multidisciplinar, mas nesse caso, acrescenta-se o Processamento de Linguagem Natural PLN.

PLN é a área que abrange uma série de técnicas a fim de analisar dados em linguagem natural, e dentro da mineração de dados apresenta forte impacto na etapa de pré-processamento (uma das etapas da MT, que será apresentada a seguir) e contribui

diretamente na preparação do texto para que o mesmo possa obter maior aproveitamento nas fases subsequentes (ARANHA; VELLASCO, 2007).

Semelhante à MD, Aranha e Vellasco (2007) descreve 5 etapas do processo de MT que serão apresentadas a seguir.

- **Coleta:** Essa etapa tem o objetivo formar a base textual do trabalho, pode ser estática ou dinâmica dependendo do contexto. Ao ser dinâmica pode-se utilizar de robôs automáticos que fazer a pesquisa, seleção e armazenamento dos dados;

- **Pré-processamento:** Uma vez obtido os dados é preciso realizar um processo de limpeza e normalização de acordo com o objetivo do trabalho a ser realizado. Consiste em uma série de transformações na base até se obter uma estrutura capaz de ser minerada. Algumas das técnicas são: Identificação de palavras válidas; Retirada de palavras comuns ou *Stopwords* (elementos do texto que não possui relevância semântica como preposições e artigos); Obtenção dos radicais das palavras, processo também chamado de *Stemming*; Separação do texto em palavras (ou *tokens*) processo também chamado de *tokenização* entre outros; Tan, Steinbach e Kumar (2009) essa etapa pode ser a mais trabalhosa e demorada devido a quantidade de formas que os dados podem ser coletados e armazenados.

  Depois da limpeza é preciso representar os dados para a mineração, em geral um texto pode ser representado como bag of words, nesse caso o texto é representado através de das palavras contidas nele e da frequência de cada palavra, o que desconsidera a sequencia na qual as palavras estão. Outra forma é representar o texto atraves de strings, onde cada documento é uma sequencia de palavras. Aggarwal e Zhai (2012) afirma que a maioria dos trabalhos de classificação de texto utilizam a bag of words devido a sua simplicidade para a tarefa.

- **Indexação:** Utilização de técnicas que buscam analisar e organizar os elementos do documento de forma que seu acesso seja mais eficiente;

- **Mineração:** Etapa responsável por escolher e aplicar os algoritmos para identificação dos padrões e extração do conhecimento;

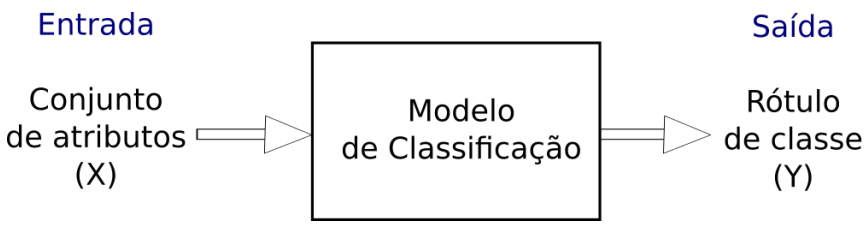
- **Análise:** Avaliação e interpretação dos resultados.

O processo de mineração de dados é quase sempre iterativo a fim de encontrar resultados satisfatórios ou explorar possibilidades percebidas apenas durante a execução das etapas. (LIU, 2007)

3.2 Tarefa de Classificação

A classificação consiste na tarefa de aprender a que classe um registro pertence baseado nas suas características, ou atributos. É uma tarefa do tipo supervisiona, ou seja, requer a inspeção humana em seu treinamento, e isso acontece através da submissão de dados com os respectivos rótulos corretos, como será explicado mais a seguir. A Figura 2 introduz o funcionamento dessa tarefa, onde um conjunto de atributos são mapeados em um rótulo que também pode ser chamado de classe ou categoria, pré-definidas.

Figura 2: Tarefa de classificação



Fonte: Adaptado de Tan, Steinbach e Kumar (2009)

Segundo Tan, Steinbach e Kumar (2009) um modelo de classificação recebe um conjunto de dados organizados em duplas - x e y, onde x é o conjunto de atributos necessários para aquela classificação, e y é o rótulo atribuído a ela. A Tabela 1 apresenta um exemplo de conjunto de dados, o conjunto em questão trata dos atributos para classificar vertebrados nas categorias: mamíferos, aves, peixes, répteis e anfíbios. O conjunto é compostos dos seguintes atributos: Temperatura corporal, cobertura da pele, dá cria, habilidade de viver na água, habilidade de voar, presença de pernas e habilidade de hibernar. O rótulo, em uma tarefa de classificação, é uma variável que só pode ser do tipo discreta, pois uma vez que seja contínua caracteriza outra tarefa chamada regressão, que é uma classificação voltada para variáveis contínuas (TAN; STEINBACH; KUMAR, 2009). Variáveis discretas são aquelas que possuem apenas um conjunto conhecido e enumerável de valores, os quais pode assumir. Já as variáveis contínuas podem assumir um número infinito de valores dentro de um intervalo, o intervalo entre 2 números racionais, por exemplo.

Tabela 1: Conjunto de dados para classificação de vertebrados

Nome	**Temperatura Corporal**	**Cobertura da Pele**	**Dá Cria**	**Ser Aquático**	**Ser Aéreo**	**Possui Pernas**	**Rótulo da classe**
Humano	Sangue quente	Cabelo	Sim	Não	Não	Sim	**Mamífero**
Píton	Sangue frio	Escamas	Não	Não	Não	Não	**Réptil**
Salmão	Sangue frio	Escamas	Não	Sim	Não	Não	**Peixe**
Baleia	Sangue quente	Cabelo	Sim	Sim	Não	Não	**Mamífero**
Sapo	Sangue frio	Nenhuma	Não	Sim	Não	Sim	**Anfíbio**
Morcego	Sangue quente	Cabelo	Sim	Não	Sim	Sim	**Mamífero**
Pomba	Sangue quente	Penas	Não	Não	Sim	Sim	**Ave**
Gato	Sangue quente	Pêlo	Sim	Não	Não	Sim	**Mamífero**
Leopardo	Sangue frio	Pêlo	Sim	Sim	Não	Sim	**Mamífero**

Fonte: Adaptado de Tan, Steinbach e Kumar (2009)

O funcionamento da classificação é descrito por Aggarwal e Zhai (2012) da seguinte forma: A partir de um conjunto de dados de treinamento, cada registro é rotulado de acordo a classe que pertence, com valores discretos, esse conjunto é utilizado para construir o modelo de classificação, que relaciona os atributos de cada registro com o seu rótulo. Uma vez construído o modelo realiza-se o teste, que consiste na submissão de um novo conjunto de dados onde os rótulos dos registros são desconhecidos, a exemplo da Tabela 2.

Tabela 2: Instância não conhecida pelo classificador

Nome	**Temperatura Corporal**	**Cobertura da Pele**	**Dá Cria**	**Ser Aquático**	**Ser Aéreo**	**Possui Pernas**	**Rótulo da classe**
Monstro de Gila	Sangue frio	Escamas	Não	Não	Não	Sim	**?**

Fonte: Adaptado de Tan, Steinbach e Kumar (2009)

Aggarwal e Zhai (2012) e Tan, Steinbach e Kumar (2009) descrevem algumas técnicas de clasificação como:

- Árvores de Decisão Essa técnica é projetada para utilizar uma divisão hierárquica do espaço de dados a partir de diferentes características do texto, dividindo-o em partições.
- Classificadores Baseados em Regras A partir de regras que buscam identificar um padrão de palavras (baseado na presença ou ausência de palavras), verificar em qual regra cada registro se encaixa e assim rotulá-lo com a respectiva classe;
- Suporte Vector Machine - SVM Esse método tenta particionar os dados utilizando delineamentos lineares entre as classes a fim de achar a fronteira ótima entre elas;

- Classificadores de Redes Neurais Através de um modelo análogo a estrutura do cérebro humano esse método aprende uma função capaz de classificar as entradas;
- Classificadores Bayesianos Generativos Consiste em tentar construir um modelo preditivo usando probabilidade das palavras subjacentes em diferentes classes, também é chamado de classificador generativo.

O *Naive Bayes* é uma classificador probabilístico baseado no teorema de *Bayes*, ele também é chamado de classificador de *Bayes* ingênuo, pois modela a distribuição de cada registro do conjunto de entrada em cada classe usando um modelo probabilístico que assume a independência entre os diferentes termos (AGGARWAL; ZHAI, 2012).

O teorema de *Bayes* aborda a probabilidade de um evento acontecer levando em conta informações *a priori* para identificar a probabilidade *a posteriori*. As informações *a priori* são obtidas através do conjunto de teste. França e Oliveira (2014) apresentam o cálculo da probabilidade como pode ser visto na Equação 3.1, onde H representa o evento a ser observado, E é a evidência, P[H] probabilidade *a priori*(antes da evidência ser vista), P(H|E) é a probabilidade *a posteriori*(depois da evidência ser vista) de H dado E e P(E) representa o somatório da probabilidade prévia do evento E.

$$P(H|E) = \frac{P(E|H) * P(H)}{P(E)} \tag{3.1}$$

No caso da classificações de texto ele calcula a probabilidade de cada atributo de um registro (palavras do texto) para determinar a classe a qual ele pertence, durante o treinamento. Uma vez treinado, o modelo classifica novos registros avaliando a probabilidade atribuída anteriormente a cada atributo contido nele.

Uma vez explicado o funcionamento da classificação é importante também entender as estratégias disponíveis para sua execução e avaliação, essas informações serão explicadas na seção a seguir.

3.3 Execução e Avaliação de Um Classificador

Segundo (TAN; STEINBACH; KUMAR, 2009) algumas estratégias podem ser usadas para a execução de um classificador, como: (i) Método *Holdout*: nesse método o conjunto de dados é dividido em conjunto de treinamento e conjunto de teste. O conjunto de treinamento é utilizado para fazer o classificador induzir o modelo e depois o modelo é avaliado a partir do seu desempenho com o conjunto de teste. A proporção entre os dados de treinamento e teste pode ser, por exemplo, 50-50 ou 2/3 para treinamento e 1/3 para o teste; (ii) Sub-Amostragem Aleatória, esse método repete o *Holdout* diversas vezes para melhorar a avaliação do desempenho, porém não tem controle sobre o número de

vezes que cada registro é usado para teste e treinamento; (iii) Validação Cruzada, uma alternativa a sub amostragem aleatória onde cada registro é usado o mesmo número de vezes no treinamento e um vez apenas para o teste.

Já a avaliação de um modelo de classificação é baseada na contagem de registros dentro do conjuntos de testes que foram rotulados de forma correta ou não (TAN; STEINBACH; KUMAR, 2009). Uma possível representação para essa análise é a Matriz de Confusão, ela mostra a relação entre o números de registros da classe A previstos corretamente através da função F para a classe A e previstos incorretamente para a classe B. E os registros da classe B previstos corretamente através da função F para a classe B e incorretamente para a classe A (TAN; STEINBACH; KUMAR, 2009). Segue um exemplo da matriz de confusão na Tabela 3.

Tabela 3: Matriz de Confusão para problemas de classificação

		Classe Prevista	
		Classe = A	**Classe = B**
Classe Real	**Classe = A**	F(AA)	F(AB)
	Classe = B	F(BA)	F(BB)

Fonte: Adaptado de Tan, Steinbach e Kumar (2009)

A fim de resumir as informações disponíveis na matriz pode-se condensá-las em 3 índices: Acurácia, Precisão e *Recall*. A acurácia é dada pelo número de previsões corretas sobre o total de previsões. A precisão é o número de previsões corretas, para uma determinada classe, dividido pelo número total de previsões para aquela classe. *Recall* é o número de previsões corretas, para uma determinada classe, dividido pelo número de exemplos daquela classe (LIU, 2007)

Segundo Liu (2007) nos casos onde a base de dados do classificador não possui classes balanceadas a acurácia não é suficiente para fazer a avaliação, nesse caso recomenda-se a utilização de outras métricas como a precisão e o *recall*. As métricas são representadas em Equação 3.2, Equação 3.3 e Equação 3.4, e tem como base a matriz de confusão Tabela 3.

$$Acurácia = \frac{F(AA) + F(BB)}{F(AA) + F(AB) + F(BB) + F(BA)} \tag{3.2}$$

$$Precisão = \frac{F(AA)}{F(AA) + F(BA)} \tag{3.3}$$

$$Recall = \frac{F(AA)}{F(AA) + F(AB)} \tag{3.4}$$

Liu (2007) afirma que embora o *recall* e a precisão não estejam relacionadas na teoria, na práticas as métricas são inversamente proporcionais. A partir disso analisamos uma outra medida que é o F-score, uma média harmônica entre a precisão e o *recall*, o F-score pode ser representado pela Equação 3.5.

$$F - score = \frac{2}{\frac{1}{Precisão} + \frac{1}{Recall}} \tag{3.5}$$

A partir dos conceitos apresentados, esse trabalho utilizou-se das etapas da mineração de texto: coleta, pré-processamento, mineração e análise, não aplicando a etapa de indexação visto que não haveria a necessidade de busca dos itens da base de dados. Utilizou-se também do algoritmo de classificação *Naive Bayes* visto o seu fácil entendimento e aplicação, além de também ter sido utilizado em trabalhos correlatos como França e Oliveira (2014), Go, Bhayani e Huang (2009) e Gomes, Neto e Henriques (2013). A avaliação do classificador foi feita a partir das métricas de acurácia, precisão, recall e F-score e a estratégia de execução utilizada foi a *Holdout*.

4 Trabalhos Correlatos

Para abordar uma solução para o problema, foi realizada uma pesquisa inicial procurando por trabalhos capazes de esclarecer os seguintes tópicos:

1. Extração e processamento de dados do *Twitter*;
2. Discurso de ódio e intolerância em redes socias;

4.1 Extração e processamento de dados do *Twitter*

A Mineração de Opiniões ou Análise de Sentimento é uma área recente que "congrega pesquisas de mineração de dados, linguística computacional, recuperação de informações, inteligência artificial, entre outras"(BECKER; TUMITAN, 2013). Segundo Becker e Tumitan (2013) os principais problemas dessa nova área giram em torno de: (i) identificação da opinião expressa sobre um conjunto de documento, (ii) identificação da polaridade da opinião expressa (positiva, negativa ou neutra) e (iii) apresentação de seus resultados de forma resumida.

Detecção Semi-Supervisionada de Posicionamento em Tweets Baseada em Regras de Sentimento desenvolvido por Dias e Becker (2016) se baseia na premissa que o problema de detecção de posicionamento possui forte componente de detecção de polaridade. O trabalho descreve uma abordagem semi-supervisionada voltada a detecção de posicionamento em *Tweets* aplicada a diferentes domínios (política, feminismo, legalização do aborto, ateísmo e mudança do clima). Apenas o texto da mensagem enviada é utilizado para análise, desconsiderando assim as menções, links e outros elementos estruturados de um *Tweet*.

A abordagem de Dias e Becker (2016) ocorre da seguinte forma:

- Inicia com a identificação do n-gramas representativos sobre o domínio que se deseja avaliar, esses n-gramas são divididos em alvo e chave, alvos são aqueles que representam o sujeito ou figura principal, a favor ou contra o assunto de interess. Já os n-gramas chave são *Hashtags* ou expressões que uma vez encontradas apresentam forte tendência sobre o posicionamento do *Tweet*;
- Depois os n-gramas são separados em favoráveis e contrários;
- Em seguida a partir das regras definidas pelos autores ocorre a rotulação automática que utiliza como critério a presença ou ausências dos n-gramas identificados anteriormente;

- A partir do conjunto rotulado cria-se um modelo preditivo utilizando o método de aprendizagem supervisionada, que é utilizado para rotular aqueles *Tweets* que não foram rotulados pelas regras.

Detecção Semi-Supervisionada de Posicionamento em Tweets Baseada em Regras de Sentimento se assemelha do presente trabalho pois também utilizou dados oriundos do *Twitter*, a partir de sua leitura foi possível identificar a estratégia de identificar termos que caracterizassem, positivamente ou negativamente, o assunto alvo da mineração.

Já o trabalho de Go, Bhayani e Huang (2009) propõe um método de extrair sentimento (positivo ou negativo) automaticamente do *Twitter*[1]. No desenvolvimento Go, Bhayani e Huang (2009) definem sentimento como positivo ou negativo, desconsiderando o neutro, e usam diferentes classificadores como: Naive Bayes, Maximum Entropy (MaxEnt), e Support Vector Machines (SVM); Assim como diferentes extratores de recursos: Unigramas, Bigramas, Unigramas e Bigramas e Unigramas com partes da fala (*tags part-of-speech*). Os autores compararam os resultados para cada combinação de classificadores e extratores de recursos considerando a acurácia de cada algoritmo de classificação, como pode ser observado na Tabela 4.

Tabela 4: Resumo dos resultados obtidos considerando a acurácia dos respectivos classificadores e extratores de features

Features	**Keyword**	**Naive Bayes**	**MaxEnt**	**SVM**
Unigrama	65.5	81.3	80.5	82.2
Bigrama	N/A	81.6	79.1	78.8
Unigrama + Bigrama	N/A	82.7	83.0	81.6
Unigrama + POS	N/A	79.9	79.9	81.9

Fonte: Adaptado de Go, Bhayani e Huang (2009)

O trabalho de Go, Bhayani e Huang (2009) foi possível observar o comportamento de vários classificadores com extratores de recursos diferentes.

Benevenuto, Almeida e Silva (2011) apresentam os principais conceitos sobre redes sociais, e discutem sobre as métricas utilizadas na análise de redes. Nesse trabalho buscou-se observar as principais formas de coleta e tratamento de dados das redes sociais. Sendo que a forma que aparentou ser mais adequada para esse trabalho foi a coleta com utilização de APIs e Bibliotecas.

No trabalho de França e Oliveira (2014), que buscava a análise de sentimentos sobre os protestos ocorridos no Brasil em 2013, foi possível observar os recursos utilizados na fase de preprocessamento do texto, anterior a tarefa de mineração. Os autores submeteram os dados coletados a 5 filtros feitos com expressões regulares que visaram retirar da base: (i)

[1] http://www.sentiment140.com/

Informações irrelevantes retornadas pela API do *Twitter* na busca executada; (ii) Menções a outros usuários da rede; (iii) *Retweets*; (iv) Caracteres de pontuação; (v) URLs. Por fim, os dados passaram pelos processos de retirada das *stopwords* e *stemming*. A partir do preprocessamento, os dados foram rotulados por 3 humanos que entraram em um consenso sobre o sentimento expresso em cada mensagem: positivo (apoio aos protestos) ou negativo (repúdio aos protestos) (FRANÇA; OLIVEIRA, 2014). Foram geradas duas bases com 100 *tweets* cada, uma base possuía apenas mensagens positivas e a outra apenas mensagens negativas. Os autores alcançaram resultados satisfatórios e analisaram a eficiência do classificador Naive Bayes a partir de medidas como: média de acerto, variância, precisão entre outras.

O trabalho de França e Oliveira (2014) foi o que mais se aproximou do trabalho desenvolvido nessa monografia, nele foi possível observar a utilização do algoritmo *Naive Bayes*, da classificação humana e manual dos *Tweets* e da retirada de elementos não significativos para a análise. Porém, os trabalhos diferem em alguns elementos como o assunto alvo da mineração e a tarefa auxiliar de identificação de termos para discurso de ódio feita no presente trabalho.

4.2 Discurso de Ódio e Intolerância em Redes Sociais Online

O dossiê Intolerâncias: Visíveis e Invisíveis no mundo digital NOVA/SB (2016b) feito pelo Comunica Que Muda, um blog da agência NOVA/SB, tinha o objetivo de monitorar como ocorriam dez tipos de intolerâncias nas redes sociais, os dados foram coletados durante 3 meses, de abril a junho de 2016, a pesquisa teve foco na intolerâncias relacionadas com: Racismo, política, classe social, aparência, homofobia, deficiência, idade/geração, religião, misoginia e xenofobia. Foram analisados 542.781 menções com a ajuda do software de monitoramento na WEB, *Torabit*, um resumo dos resultados pode ser vistos nas Tabela 5 e Tabela 6

Tabela 5: Quantidade de Menções

Assunto	**Total de menções em 3 meses**
Política	273.752
Misoginia	79.484
Homofobia	53.126
Deficiência	40.801
Racismo	32.376
Aparência	27.989
Idade/Geração	14.502
Classe Social	11.256
Religiosa	7.361
Xenofobia	2.134

Fonte: Adaptado de NOVA/SB (2016a)

Tabela 6: Percentual de menções negativas

Assunto	**Percentual de menções negativas**
Racismo	97,60%
Política	97,40%
Classe Social	94,80%
Aparência	94,20%
Homofobia	93,90%
Deficiência	93,40%
Idade/Geração	92,30%
Religiosa	89%
Misoginia	88%
Xenofobia	84,80%

Fonte: Adaptado de NOVA/SB (2016a)

A pesquisa apresentou seus resultados com base em algumas definições, como: Intolerância real (Aquela que é direta a um caso concreto ou a uma pessoa física); Intolerância abstrata (Aquela que é indireta, não atingindo uma pessoa específica mas a um grupo de pessoas que compartilham certa característica); Intolerância visível (Aquela que descrimina de forma direta e explícita uma pessoa ou grupo); Intolerância invisível (Que aparece de forma velada através de um comentário ou comportamento)(NOVA/SB, 2016b).

Para cada tema o dossiê apresenta gráficos representando características das mensagens, grafos representando as conexões, nuvem de palavras e distribuição das mensagens no território nacional. A seguir, na Figura 4.2, uma nuvem de palavras que representam as palavras mais frequentes nas menções sobre a intolerância relacionada com misoginia.

Figura 3: Nuvem de palavras

Essa nuvem de palavras representam as palavras mais frequentes nas menções sobre a intolerância relacionada com misoginia. Fonte: (NOVA/SB, 2016b)

No dossiê Intolerâncias: Visíveis e Invisíveis no mundo digital NOVA/SB (2016b) foi possível ver exemplos da disseminação da intolerância entre os usuários das redes sociais, bem como ver a pertinência da pesquisa sobre o tema, inclusive a foi utilizada no processo de extração de termos que denotassem discurso de ódio.

Já o trabalho de Silva et al. (2011) esclarece os principais pontos sobre discurso de ódio e apresenta uma pesquisa feita sobre a jurisprudência brasileira com relação a esse tipo de discurso disseminado em redes sociais online. A pesquisa foi feita em 2011, considerando as redes *Facebook* e *Orkut*, famosas nesse período(DâMASO, 2015). Através de abordagens qualitativa e quantitativa, as autoras analisaram a existência e a intensidade numérica de decisões judicias sobre discurso de ódio presentes em RSOs e o tratamento dado a esses casos, respectivamente. Chegaram a conclusão que existem casos sobre o assunto porém é um número pequeno, comparado com outras demandas de casos judiciais, e que mesmo com os poucos casos o tratamento dado pela justiça brasileira mostra o caráter não absoluto da liberdade de expressão nas redes, e o enquadramento das violações na lei Nº 7.716/89, lei que define os crimes relacionados ao preconceito de raça ou de cor. Por fim as autoras afirmam que a partir dos resultados não evidenciam respostas inquestionáveis mas apresentam outras perguntas a serem exploradas, como: "Qual é o procedimento a ser adotado para situações em que se consumar discriminação por outro critério? Como se trata o discurso odiento de gênero? E a idosos? E aquele proferido contra

homossexuais?" (SILVA et al., 2011).

A partir da leitura do trabalho Silva et al. (2011) foi possível entender a complexidade do tema discurso de ódio, já pesquisado em 2011, bem como a sua jurisprudência na justiça brasileira.

A partir da análise dos trabalhos mencionados foi possível o melhor entendimento acerca dos temas e assim maior segurança na aplicação dos conceitos na construção e execução do processo proposto. Através das pesquisas no item seção 4.1 verifica-se que o *Twitter* é de fato uma fonte rica para a mineração textual e análise de sentimentos, bem como o funcionamento dos processos e técnicas utilizadas para a extração de informações dessa rede. Já no item seção 4.2 pode-se observar a relevância do domínio de discurso de ódio, bem como as características desse domínio no cenário brasileiro.

5 Processo de Mineração Textual para Reconhecimento de Mensagens com Tendência a Discurso de Ódio no *Twitter*

As RSOs têm um papel importante no cenário atual da Internet, onde existe uma grande base de dados de forma descentralizada e desestruturada. Redes desse tipo proporcionam uma integração virtual entre os indivíduos que as utilizam, principalmente através do compartilhamento de conteúdo diverso. A partir dessa facilidade de disseminação de conteúdo, positivo ou negativo, nos deparamos com a frequente intolerância e propagação de discurso de ódio nas redes, prática nociva para a sociedade como um todo por dificultar as relações interpessoais e ferir a dignidade da pessoa humana. A partir desse cenário chegou-se ao seguinte problema de pesquisa: Que processo torna possível a extração de dados do *Twitter* para reconhecimento de discurso de ódio?

A fim de responder esse problema buscou-se através da hipótese de que uma vez obtendo resultados satisfatórios tendo o escopo reduzido para identificação do discurso de ódio contra negros e contra mulheres, será possível em trabalhos futuros fazer a expansão desse processo para outros tipos de discurso de ódio. Sendo assim o presente trabalho teve como objetivo propor, implementar e avaliar um processo capaz de extrair dados do *Twitter* e identificar as mensagens que tendem a discurso de ódio utilizando as etapas da Mineração de Texto.

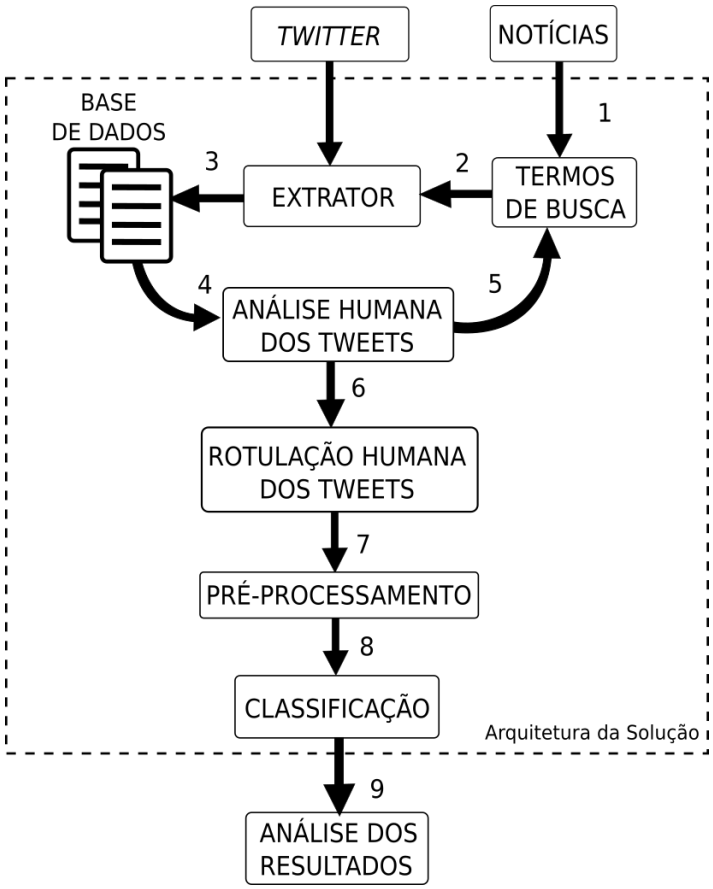
A metodologia seguida a fim de alcançar o objetivo foi:

1. Realização de testes de extração de dados com a API do *Twitter* visando o entendimento do formato dos dados retornados;
2. Revisão bibliográfica dos temas relacionados, como: Mineração de texto, discurso de ódio, análise de sentimento, extração de dados das RSOs entre outros a fim de entender os conceitos e técnicas necessárias para a concretização do trabalho;
3. Elaboração de uma arquitetura reduzida do processo e execução testes com o extrator construído a fim de entender o formato dos dados, do discurso disseminado no *Twitter* e identificar novos termos de busca (processo descrito na seção 5.1);
4. Especificação e implementação o processo final. A implementação consistiu na execução de todas as etapas da mineração de texto descritas no decorrer do capítulo;
5. Avaliação processo através da eficiência da classificação, utilizando as métricas de acurácia, precisão, *recall* e *F-score*.

Ao examinar as mensagem publicadas no *Twitter* o escopo desse trabalho tratou de observar as características e particularidades dessas mensagens, considerando apenas o idioma português, a fim de construir um modelo preditivo capaz de classificar se um *tweet* tende ou não a discurso de ódio.

Como falado anteriormente o escopo do trabalho se limitou a tentar identificar textos que tenham tendência a discurso de ódio contra negros e contra mulheres. A seguir a representação da solução trabalhada na Figura 4.

Figura 4: Arquitetura do processo proposto para alcançar o objetivo do trabalho



Fonte: A autora

Cada componente desempenha uma função dentro da solução:

1. A partir de notícias que relatavam casos de descriminação contra negros e descriminação contra mulheres nas redes sociais, foram extraídos termos para busca de *tweets*;

2. Inserção dos termos no módulo de extração;

3. Armazenamento das informações relevantes para o projeto a partir dos resultados da extração;

4. Análise dos resultados buscando entender o discurso proveniente daquelas mensagens e analisar a relevância dos *tweets* retornados a partir dos termos inseridos;

5. A partir da análise anterior foi feita uma refatoração nos termos além da identificação de novos termos para a busca;

6. Uma vez fechado o ciclo de coleta, foi necessário realizar rotulação humana dos *tweets*, a fim de submetê-los à tarefa de classificação;

7. Etapa de pré-processamento dos dados, responsável por aplicar processos como retirada de *stopwords* e *stemming*;

8. Submissão dos dados ao classificador para treinar e testar o mesmo quanto ao reconhecimento de *tweets* que tendem ou não a discurso de ódio;

9. Análise dos resultados.

A solução descrita anteriormente foi implementada utilizando a linguagem *Python* e bibliotecas de terceiros que serão listadas no decorrer da explicação do processo. Justifica-se a utilização da linguagem *Python* visto a facilidade de implementação e os vários recursos que essa linguagem possui através das bibliotecas desenvolvidas para ela, como *Twitter*, *NLTK* e *Sklearn*, além de ser fortemente recomendada pelo livro Russell (2013). A seguir será detalhado como as etapas da mineração de texto foram aplicadas nesse trabalho, assim como os passos auxiliares para as etapas.

5.1 Seleção de Termos

Com o propósito de embasar a utilização de determinadas palavras chaves e estabelecer um processo capaz de ser replicado, foram analisadas notícias veiculadas na Internet sobre casos de descriminação, misoginia, racismo e machismo.

Entre os meses de Janeiro e Fevereiro de 2017 foram realizadas pesquisa no buscador *Google*[1] utilizando *strings* de busca(palavra ou conjunto de palavras utilizado na busca) como:

“racismo” and “twitter”,
“racismo” and “twitter” and “justiça”,
“machismo” and “twitter”

[1] https://www.google.com.br/

A partir da análise das notícias retornadas, novas *strings* de busca foram se formando, muitas vezes com base em casos específicos como o da apresentadora Maria Júlia (CORREIO24H, 2016) e do ex ministro Joaquim Barbosa (MORGENSTERN, 2012).

Essa etapa de seleção de termos não faz parte das etapas da mineração de texto apresentadas na literatura, mas para esse trabalho em particular foi um passo necessário para servir de insumo para a primeira etapa da MT. O objetivo de olhar essas notícias foi entender como esse tipo de discurso era disseminado para extrair exemplos e assim fazer uma busca por *tweets* de forma embasada, ao invés de utilizar apenas o senso comum sobre palavras de cunho descriminatório.

A última versão dos termos utilizados no extrator de dados podem ser visto no código da classe Termos listada no Apêndice B.

5.2 Coleta de Dados

Para o desenvolvimento do extrator foi utilizada a biblioteca *Twitter* de versão 1.17.1[2] desenvolvida por Mike Verdone e o grupo de desenvolvedores do *Python Twitter Tools*[3]. Essa biblioteca possibilitou fácil acesso aos recursos da API do *Twitter*, pois foi criada para essa plataforma[4].

A API do *Twitter* fornece 2 mecanismos de autenticação: Autenticação *Application-Only* e Autenticação de usuário. A primeira é indicado para aplicações que planejam apenas extrair dados. Já segunda é destinada para quem deseja interagir com a rede, pois através dela ocorre um vínculo entre a aplicação desenvolvida e o respectivo usuário, e a partir desse vínculo é possível realizar ações típicas de usuário, como postar mensagens na rede(JúNIOR, 2014). Devido a natureza do trabalho o mecanismo utilizado foi o Autenticação *Application-Only*.

Para realizar a autenticação na rede foi preciso registrar uma aplicação através do gerenciador de aplicações[5] utilizando uma conta de usuário do *Twitter*. Ao criar uma aplicação foi preciso inserir algumas informações como: nome, objetivo e site da aplicação. A partir desse cadastro foram recebidas 4 informações para serem utilizadas na autenticação, foram elas:

- API Key: Chave utilizada para que a aplicação se identifique perante o Twitter.
- API Secret: Chave para a aplicação provar sua autenticidade;

[2] https://pypi.python.org/pypi/twitter
[3] Projeto no GitHub https://github.com/sixohsix/twitter/tree/master
[4] A documentação da API mostra algumas bibliotecas que possuem integração com a plataforma através do endereço https://dev.twitter.com/resources/twitter-libraries
[5] Disponível em: https://apps.twitter.com/

- Access Token: Após a identificação é preciso que a aplicação envie esse código para que o serviço possa identificar qual o nível de acesso que ela possui;
- Access Token Secret: Chave para a aplicação provar sua autenticidade com relação ao *Access Token*.

Uma vez de posse dessas chaves foi possível iniciar o desenvolvimento do módulo extrator.

Ao receber os dados da API o extrator selecionou algumas informações como: número identificador do *tweet*, data de criação, texto (mensagem publicada), e o local do usuário. A API retorna os dados no formato JSON, esse formato pode ser exemplificado na Figura 5. Porém para solução do trabalho em questão esses dados foram coletados e armazenados em um arquivo CSV Comma-Separated Values (Arquivo separado por vírgulas), para facilitar a análise manual a ser feita posteriormente.

Figura 5: Formatação do arquivo JSON resultante da consulta feita a *API Console Tool* do *Twitter*


```
{
  "created_at": "Sun Jun 05 17:08:04 +0000 2016",
  "id": 739504072953368600,
  "id_str": "739504072953368578",
  "text": "RT @oBarbudinho: Hoje vou fazer um
tutorial. Nessas olimpíadas não vamos dar sossego aos
golpistas. Vamos denunciar o golpe ao vivo em toda…",
  "truncated": false,
  "entities": {
    "hashtags": [],
    "symbols": [],
    "user_mentions": [
      {
        "screen_name": "oBarbudinho",
        "name": "O Barbudinho",
        "id": 336039456,
        "id_str": "336039456",
        "indices": [
          3,
          15
        ]
      }
    ],
    "urls": []
  },
  "metadata": {
    "iso_language_code": "pt",
    "result_type": "recent"
  },
```


API Console Tool do *Twitter* é uma ferramenta online disposta pela empresa para consulta em sua base da dados. Nesse exemplo pesquisou-se pelo termo "Olimpíadas"e solicitou-se que os resultados fossem apenas em português. Fonte: A autora

A coleta foi realizada entre 27 de fevereiro e 8 de março, visando observar o comportamento dos *tweets* durante dois eventos excepcionais: Carnaval e Dia internacional

da mulher, pois acontecimentos como esses influenciam no assunto tratado pelos usuários da rede.

O extrator foi executado de 2 em 2 horas a partir das 17:00 horas até as 7:00 horas do outro dia, visando abarcar os horários mais utilizados pelos usuários segundo PROPMARK (2016). É importante ressaltar que o método de pesquisa utilizado através da biblioteca *Twitter 1.17.1* trouxe uma amostra de *tweets* recentes e não todos os *tweets* de forma histórica, inclusive a questão histórica não é requisito para o escopo do trabalho, uma vez que estamos analisando o discurso no período atual.

Um resumo do algoritmo de extração é pode ser visto no Código 5.1.

Código 5.1: Parte do algoritmo para extração de dados

```
while rodada_atual <= rodada_total:

    with open('tweetsExtrator/' + str(datetime.now().day) + "_" +
        str(datetime.now().month) + "_" + str(datetime.now().year) +
        '_tweets_contra_negros.csv', 'a', encoding='utf-8') as f:

        for n_grama in termos.n_gramas_contra_negros:
            twitter = Extrator()
            twitter.busca(n_grama, idioma, quantidade)
            writer = csv.writer(f)
            writer.writerows(twitter.lista_tweets)

    with open('tweetsExtrator/' + str(datetime.now().day) + "_" +
        str(datetime.now().month) + "_" + str(datetime.now().year) +
        '_tweets_contra_mulheres.csv', 'a', encoding='utf-8') as g:

        for n_grama in termos.n_gramas_contra_mulheres:
            twitter = Extrator()
            twitter.busca(n_grama, idioma, quantidade)
            writer = csv.writer(g)
            writer.writerows(twitter.lista_tweets)
```

Fonte: A autora

Primeiro é criado um laço de repetição, que é uma técnica na programação quando se deseja que um bloco se instruções seja repetido enquanto determinada condição proposta for válida. Esse laço é para que o algoritmo seja executado durante N rodadas, onde o intervalo entre as rodadas é de duas horas; Dentro desse 1º laço de repetição um 2º laço é criado para percorrer todos os itens da lista de *Strings* contra negros e para cada *String* da lista chama-se a função *Busca*. Por fim os resultados da busca para aquela *String* são salvos em um arquivo CSV; Ainda dentro do 1º laço de repetição um 3º laço é criado para percorrer todos os itens da lista de *Strings* contra mulheres e para cada *String* da lista

chama-se a função *Busca*. Por fim os resultados da busca para aquela *String* são salvos em um arquivo CSV.

A função *Busca* mencionada anteriormente é ilustrada no Código 5.2.

Código 5.2: Função de Busca do extrator

```
1  def busca(self, termo, idioma, quantidade):
2      tweets_string = self.api.search.tweets(
3          q=termo, lang=idioma, count=quantidade)
4
5      for resultado in tweets_string["statuses"]:
6          retweet = re.search('RT ', resultado["text"])
7          if retweet == None:
8              self.lista_tweets.append((resultado["id_str"],
9                                        resultado["created_at"],
10                                       resultado["text"],
11                                       resultado["user"]["location"]))
```

Fonte: A autora

A busca recebe como parâmetro a expressão a ser buscada, o idioma (português) e a quantidade de *tweets* a serem retornados[6]; O retorno do método é analisado e os *tweets* que possuem a sigla "RT" no início da mensagem não são considerados, visando a não classificação de *retweets*. Justifica-se a retirada dos *retweets* para evitar o ruído que mensagens repetidas podem gerar na classificação; As informações número identificador, data de criação, texto e localização são selecionadas, colocadas em uma lista e essa lista é retornada para quem chamou a função.

O código da classe *Extrator* e todo o script de extração podem ser vistos na íntegra no Apêndice A e no Apêndice C, respectivamente. Os valores das chaves de autenticação foram retirados do código por uma questão de segurança.

A coleta foi um processo incremental, onde a cada iteração buscava-se refatorar os termos de busca para trazer resultados mais relevantes, relevantes no sentido de se aproximarem cada vez mais ou trazerem exemplos de discurso de ódio, até que fosse obtida uma base consistente com exemplos variados ou de tamanho considerável, visto a dificuldade de processar cada arquivo CSV manualmente na refatoração.

5.3 Rotulação Humana dos Tweets

A fim de preparar os dados para serem submetidos à tarefa de classificação, foi preciso rotulá-los previamente. Para isso, 500 *tweets* foram selecionados aleatoriamente

[6] O quantidade máxima de *tweets* retornados a cada termo pesquisado é 100, por questões de limites da API. Esses limites visam garantir a continuidade do serviço.

da base de dados de cada assunto e rotulados pela autora do trabalho. Esse recorte foi feito através de uma algoritmo também em *Python* e pode ser visto no Apêndice F. Como resultado foi gerado um outro arquivo CSV com 500 registros que foram rotulados da seguinte forma: acrescentou-se uma coluna na tabela indicando 0 para *tweets* que não tendiam a discurso de ódio e 1 para queles que tendiam.

O trabalho de França e Oliveira (2014) rotulou manualmente 100 mensagem para cada contexto em sua pesquisa e obtiveram resultados satisfatórios, a partir disso foi decidido utilizar um número superior ao utilizado no trabalho correlato, tendo em vista que quanto mais dados para treinamento melhor o classificador se torna. Não foi selecionado um número maior por conta da dificuldade de classificar os dados manualmente.

5.4 Pré-processamento

A Natural Language Toolkit *NTLK* (Conjunto de Ferramentas para Linguagem Natural)[7] é um biblioteca voltada para o trabalho com dados da linguagem humana, essa biblioteca fornece um conjunto de funções de processamento textual, como: *tokenização*, *stemming*, *stopwords* entre outros recursos (BIRD; KLEIN; LOPER, 2009).

Na Etapa de pré-processamento foram aplicados 4 filtros nos dados de entrada a fim de retirar itens que não são relevantes para a classificação do texto. Para isso foram utilizadas expressões regulares e funções do *Python*. Foram retirados: Menções, links, *emoticons*, e símbolos de pontuação.

A partir dessa limpeza inicial foram realizados mais dois processos: a retirada de *stopwords* e o processo de *stemming*, utilizando a biblioteca *NLTK* em sua versão 3.2.2. O código da classe construída para o pré-processamento consta no Apêndice D.

5.5 Mineração

Antes de submeter os dados ao algoritmo é preciso transformá-los em um formato adequado para o processamento, e no contexto de mineração de texto essa é uma tarefa do extrator de *features* (características).

Em geral um texto pode ter suas features representadas como *bag of words* (bolsa de palavras), nesse caso representa-se o texto através das palavras contidas nele e da frequência de cada palavra, o que desconsidera a sequência na qual as palavras estão. Outra forma é representar o texto através de *strings*, onde cada documento é uma sequencia de palavras. Aggarwal e Zhai (2012) afirma que a maioria dos trabalhos de classificação de texto utilizam a *bag of words* devido a sua simplicidade para a tarefa.

[7] http://www.nltk.org/

A extração de *features* para classificação de texto funciona da seguinte forma: primeiro ele identifica as palavras existentes no corpus de treinamento, o corpus consiste em todos os *tweets* que serão utilizados para o treinamento, essas palavras são identificadas de forma única, ou seja, ele não considera as palavras repetidas. Depois de ter essas palavras forma-se uma tabela onde as colunas representam cada palavra existente no corpus, e as linhas cada documento ou registro que está sendo analisado, nesse caso, cada *tweet*. Depois a extração de feature preenche a tabela fazendo o mapeamento indicando quais palavras então presentes em cada tweet adicionando em cada célula a frequência daquela palavra naquele *tweet*. A última coluna à direita representa a classe a qual o *tweet*, que possui aquele conjunto de palavras (*features*), pertence.

Assim, ao aplicar o algoritmo de *Naive Bayes* espera-se que ele aprenda quais as palavras que estão presentes nos casos positivos e negativos e como elas influenciam na classificação daquele *tweet*.

Segundo Lee (2000) na tarefa de classificação de texto não é incomum a representação das *features* seja de 10^4 à 10^7 embora saiba-se que apenas parte dessas informações é crucial para o algoritmo. Não faz parte do escopo desse trabalho fazer um estudo da otimização de extração das *features*, ficando assim para um trabalho futuro.

Scikit-learn ou sklearn[8] é uma biblioteca em *Python* específica para a área de aprendizagem de máquina e possui uma extensa documentação e exemplos, ela integra diversos algoritmos para essa área que é responsável por estudar algoritmos que possibilitam às máquinas aprenderem padrões de forma automática, sem uma programação específica para um padrão específico. Essa biblioteca se concentra em levar o emprego dessas técnicas para não especialistas da área através de uma linguagem de alto nível como o *Python*(PEDREGOSA et al., 2011). A versão utilizada nesse trabalho é a 0.17.1.

Para a classificação dos *tweets* utilizou-se o algoritmo *Gaussian Naive Bayes* disponível na biblioteca. Esse algoritmo é uma versão do Naive Bayes descrito na seção 3.2 porém assume a probabilidade das *features* como uma distribuição Gaussiana.

O código do classificador é resumido a seguir no Código 5.3.

Código 5.3: Resumo do classificador

```
1 p = PreProcessador()
2
3 p.normalizar_dados(arquivo_in_cn)
4 p.get_tweets_e_rotulos()
5
6 X_treino, X_teste, Y_treino, Y_teste = cross_validation.train_test_split(p.
      tweets, p.rotulos_tweets, test_size=0.3, random_state=0)
7
8 vectorizer = CountVectorizer(min_df=1, stop_words=None)
```

[8] http://scikit-learn.org/stable/

```
9
10 X_treino_vec = vectorizer.fit_transform(X_treino)
11
12 clf = GaussianNB()
13
14 clf.fit(X_treino_vec.toarray(), Y_treino)
15
16 X_teste_vec = vectorizer.transform(X_teste)
17
18 pred = clf.predict(X_teste_vec.toarray())
```

Fonte: A autora

Primeiro instancia-se o pré-processador para carregar o arquivo de entrada e fazer a limpeza nos dados. Após separar os dados de entrada entre *tweets* e rótulos, realiza-se a repartição dos 500 registros entre conjunto de treinamento e conjunto de teste, utilizando 30% para teste e 70% para treinamento, conforme a estratégia *Holdout*. Depois aplica-se a extração de *features* sobre os dados do treinamento. Em seguida o classificador Gaussian Naive Bayes é instanciado, treina-se o classificador a partir dos *tweets* e rótulos de treinamento e por fim realiza-se a predição apenas com os *tweets* de teste, sem a utilização dos rótulos.

O código do classificador encontra-se detalhado no Apêndice E.

6 Análise dos Resultados

Como foi descrito na seção 3.3 para análise dos resultados do classificador foram utilizadas as seguintes métricas: acurácia, precisão, *recall* e *F-score*.

A análise dos resultados será apresentada primeiro quanto as particularidades da base de dados coletada, a fim de introduzir a complexidade da tarefa, e depois quanto aos resultados das métricas aferidas do classificador.

6.1 Análise Sobre a Base de Dados Coletada

As informações coletadas pelo extrator (entre os dias 27 de fevereiro de 2017 e 8 de março de 2017) foram salvas em arquivos CSV dentro de uma pasta específica para cada tipo de discurso de ódio pesquisado por esse trabalho. Cada *string* de busca gerava um arquivo separado, para que assim pudesse ser feita uma análise e dos resultados que estavam sendo retornados e a refatoração das *strings*. Ao final do processo de refatoração os resultados recuperados entre a noite do dia 7 e a manhã do dia 8 de março foram integrados em um único arquivo, para cada tipo de discurso de ódio, assim foram gerados um arquivo final para DO contra mulheres e outro para DO contra negros. Foram utilizados apenas os resultados da coleta dos dias 7 e 8 visto que o extrator executado nessas datas tinha a última versão das *strings* de busca refatoradas. A partir dos arquivos com os dados integrados que foram selecionados os 500 *tweets* para classificação, 500 para cada contexto, como explicado na seção 5.3.

As Tabela 7 e Tabela 8 mostram a evolução dos termos durante a coleta no processo de refatoramento.

Essa evolução aconteceu a medida em que eram obtidos resultados mais relevantes para a análise, um exemplo disso é a necessidade da junção da palavra “preto” ou “preta” com alguma outra palavra que pudesse ser usada para denotar discurso de ódio como “imundo” ou “macaco”, pois uma vez usada apenas as palavras “preto” ou “preta” (palavras que usadas no cotidiano como sinônimo de homem negro ou mulher negra, respectivamente) a possibilidade de virem resultados relacionados a cor ou a time de futebol, por exemplo, são maiores. Ao utilizar essas palavras chaves juntas é possível um direcionamento dos resultados, porém vale ressaltar que ao analisar os resultados ainda dos testes iniciais com a extração observou-se que a busca não considerou a ordem das palavras, apenas a presença delas, e as palavras não foram buscadas exclusivamente no texto do *tweet*, a busca considerou outros campos também, como nome de usuário.

Contudo, apenas o texto do *tweet* foi usado na classificação, porém as outras

Tabela 7: *Strings* de buscas para *tweets* no início da coleta

Assunto	**Termos**
Discurso de ódio contra negros	"negro", "negros", "negra", "negras", "cotista", "cotas", "cota", "preto", "preta", "pretos", "pretas", "lugar de negro", "lugar de preto", "cor suja", "mulher negra", "mulher preta", "mulheres negras", "mulheres pretas", "macaco", "vitima", "vitimista", "mimimi"
Discurso de ódio contra mulheres	"mulher merece", "feminismo", "feminazi", "sexismo", "mulher serve", "mulher vestindo", "ela vestindo", "mulher veste", "ela veste", "estupro", "estuprada", "lugar de mulher", "lavar uns pratos", "varrer uma casa", "merece ser estuprada", "está pedindo", "ta pedindo", "assédio", "assédio carnaval", "respeitaasmina", "respeita as mina", "respeite as mina", "vagabunda", "mulher fácil", "vadia", "puta", "piranha", "mal comida", "machismo", "misógino", "estuprei", "comi", "foder", "feministas", "feminista", "aborto"

Fonte: A autora

informações foram retiradas e observadas para entender melhor o contexto da aplicação e avaliar a possibilidade de relacionar as mensagens com outras informações, como a localidade por exemplo. Contudo, foi observado que a informação da localização do usuário é um campo preenchido por ele, o que dificulta uma análise regional sobre os *tweets* visto que dessa forma essa informação se torna despadronizada. Um exemplo disso pode ser observado na Tabela 9.

Tabela 9: Exemplos da informação de localização dos usuários

Localização do Usuário
São José dos Campos, São Paulo
27
Boston
Sinop, Brasil
Sananduva - RS
Favela
Brasilia
MAO / AM
estados unidos do canada
lado escuro da cidade
Bh

Fonte: A autora

A fim de observar a diversidade presente na base de dados foram construídas 2 nuvens de palavras a partir de cada arquivo final, esses diagramas foram gerados utilizando

Tabela 8: *Strings* de buscas para *tweets* no final da coleta

Assunto	**Strings**
Discurso de ódio contra negros	"atendido por cotista", "cabelo ruim", "cotas raciais", "cotista desgraçado", "esse preto crl", "filme cotista", "gente negra", "gente preta", "lugar de negra", "lugar de negras", "lugar de negro", "lugar de negros", "lugar de preta", "lugar de pretas", "lugar de preto", "lugar de pretos", "merece mulher negra", "merece mulher preta", "não gosto de negro", "não gosto de negros", "não gosto de pretos", "não gosto de preto", "nega macaca", "nega", "nego macaco", "nego", "negra banana", "negra cotista", "negra feia", "negra imunda", "negra macaca", "negra vitima", "negra vitimismo", "negras cotista", "negras macaca", "negro banana", "negro cotista", "negro feio", "negro imundo", "negro macaco", "negro merece", "negro vitima", "negro vitimismo", "negros cotistas", "negros macacos", "odeio negro", "odeio negros", "odeio pretos", "oscar cotista", "por que negro", "por que preto", "povo negro", "povo preto", "pq preto", "pq negro", "preta banana", "preta cotista", "preta feia", "preta imunda", "preta macaca", "pretas cotistas", "pretas macacas", "preto banana", "preto cotista", "preto fedido", "preto feio", "preto imundo", "preto macaco", "preto merece", "pretos cotistas", "pretos macacos", "racismo", "racista", "racistas", "só podia ser negro", "só podia ser preto"
Discurso de ódio contra mulheres	"depois reclama assédio carnaval", "depois reclama assédio", "essa cachorra", "essa puta", "essa vadia", "essa vagabunda", "estuprada", "estuprei", "estupro", "falta de rola", "fecha as pernas", "feminazi", "feminista", "feministas", "lixo feminista", "lugar de mulher", "mal comida", "merece ser estuprada", "mulher aborto", "mulher apanha homem", "mulher cachorra", "mulher é tudo", "mulher lavar uns pratos", "mulher merece porrada", "mulher merece tiro", "mulher piranha", "mulher puta", "mulher que está pedindo", "mulher que ta pedindo", "mulher que", "mulher vadia", "mulher vagabunda", "mulher varrer uma casa", "nessa vagabunda", "respeita as mina", "respeitaasmina", "respeite as mina", "sapatão feminista"

Fonte: A autora

a ferramenta *Word Cloud Generator*[1,2] de Jason Davies[3], ao inserir o texto que deve servir de base para o diagrama a ferramenta faz o seu processamento exibindo, de acordo com a configuração, as palavras mais frenquentes. Para gerar o texto base para a nuvem de palavras foi construído um script em Python que utiliza o mesmo pré-processador usado na classificação, porém sem aplicar o processo de *steeming* para poder identificar quais eram as palavras exatas e não apenas seus radiciais, mais detalhes sobre o script pode ser visto no Apêndice G.

Depois da limpeza para geração dos diagramas o arquivo final para DO contra as mulheres tinha 1394 linhas com cerca de 14.237 palavras. Já o arquivo final de DO contra negros possuí 2388 linhas com cerca de 23.646 palavras. Os diagramas gerados foram limitados as 100 palavras mais frequentes.

A seguir a nuvem de palavras sobre DO voltado aos negros na Figura 6 e voltado às mulheres na Figura 7

Figura 6: Nuvem de palavras representativa do discurso de ódio voltado aos negros

Essa nuvem de palavras representam as palavras mais frequentes na base de dados coletada nesse contexto. Fonte: A autora

Durante a análise dos resultados outros dois pontos tiveram destaque também. O primeira foi a utilização da hashtag "#SerMulherÉ" durante o dia 8 de março. Onde pôde-se ver os principais termos que os usuários relacionaram com as mulheres. A seguir na Figura 8 a nuvem representativa do uso dessa hashtag.

Assim é possível observar os principais termos mencionados nos *tweets* coletados, bem como entender parcialmente o conteúdo do discurso disseminado. Parcialmente pois a nuvem de palavras destaca uma palavra de acordo com a sua frequência no texto, sem

[1] https://www.jasondavies.com/wordcloud/
[2] https://www.jasondavies.com/wordcloud/about/
[3] https://www.jasondavies.com/

Figura 7: Nuvem de palavras representativa do discurso de ódio voltado às mulheres

Essa nuvem de palavras representam as palavras mais frequentes na base de dados coletada nesse contexto. Fonte: A autora

Figura 8: Nuvem de palavras representativa da utilização da hashtag “#SerMulherÉ”

Fonte: A autora

levar em consideração a ordem ou semântica das mesmas, ou seja, esse diagrama considera quais as palavras mais usadas e não como são usadas.

E o segundo foi a utilização de variações do verbo "estuprar" como se fosse uma palavra trivial que não denota um ato de violência, sua utilização pode ser exemplificada na Tabela 10

Tabela 10: Exemplos da utilização de variações do verbo estuprar

Mensagens dos Tweets
HAHAHAHAHA BEIJA-FLOR SENDO ESTUPRADA NESSE QUESITO
Estuprei seus ouvidos com versos.Eles se apaixonaram quando eu ejaculei conhecimento lá no fundo. Kkk boa noite.
Acho que eu estuprei o replay do começo kkkkkkkkk (LINK)
Estuprei o replay com esse vídeo!! Mulher cadê teus remédios ? (EMOTICON) (MENÇÃO) (LINK)
(MENÇÃO) ja estuprei essa musica umas trocentas vezes ...
(MENÇÃO) (MENÇÃO) estuprei mesmo, mereceu

Exemplos de tweets onde os respectivos links, emoticons e menções foram substituídos pelas macros (LINK)(EMOTICON) e (MENÇÃO). Fonte: A autora

Essa representação não foi o objetivo fim do trabalho, e serviu apenas para uma ilustração do conteúdo da base de dados trabalhada e bem como da relevância do estudo de dados oriundos das redes sociais.

6.2 Resultados da Classificação

Para a obtenção das métricas definidas para a avaliação foram utilizadas funções disponíveis na própria biblioteca *Sklearn* dentro de um módulo chamado *Metrics* que inclui funções para métricas de desempenho. É possível ver no Código 6.1 a utilização dessas funções.

Código 6.1: Utilização das funções para aferir as métricas

```
1 print("Matriz de confusao:")
2 print(confusion_matrix(Y_teste, pred, labels=classes))
3
4 print("Acuracia:")
5 print(accuracy_score(Y_teste, pred))
6
```

```
print("Precisao:")
print(precision_score(Y_teste, pred, average=None, labels=classes))

print("Recall:")
print(recall_score(Y_teste, pred, average=None, labels=classes))

print("F-score:")
print(f1_score(Y_teste, pred, average=None, labels=classes))
```

Fonte: A autora

A seguir serão apresentadas a matriz de confusão para o DO contra negros e DO contra mulheres na Tabela 11 e na Tabela 12, respectivamente. Onde 0 indica não tendência a DO e 1 indica Tendência a DO.

Tabela 11: Matriz de Confusão para DO contra negros

		Classe Prevista	
		0	1
Classe Real	0	141	2
	1	5	2

Fonte: A autora

Tabela 12: Matriz de Confusão para DO contra mulheres

		Classe Prevista	
		0	1
Classe Real	0	87	13
	1	23	27

Fonte: A autora

A fim de melhorar o entendimento será utilizada a seguinte convenção para a análise dos resultados: os exemplos que tendem a discurso de ódio serão chamados de positivos e aqueles que não tendem serão chamados de negativos.

Dentre a base de dados utilizada, a distribuição dos exemplos ficou da seguinte forma: 21 positivos e 479 negativos para o DO contra negros, já para o DO contra mulheres foram 159 positivos e 341 negativos. A partir disso pode-se dizer que existiu claramente uma quantidade maior de exemplos negativos em detrimento dos exemplos positivos, principalmente do contexto racial.

O classificador para DO contra negros teve uma acurácia de 95% e o contra mulheres 76%, porém, conforme dito na seção 3.3 nos casos onde a base de dados do classificador não possui classes balanceadas a acurácia não é suficiente para fazer a avaliação, nesse

caso recomenda-se a utilização de outras métricas como a precisão e o *recall*(LIU, 2007). A seguir um resumos com as métricas precisão, *recall* e *F-score* para as respectivas classes, em Tabela 13 e Tabela 14.

Tabela 13: Métricas para o DO contra negros

	Precisão	**Recall**	**F-score**
Não tende a DO	0.96	0.98	0.97
Tende a DO	0.50	0.28	0.36

Fonte: A autora

Tabela 14: Métricas para o DO contra mulheres

	Precisão	**Recall**	**F-score**
Não tende a DO	0.79	0.87	0.82
Tende a DO	0.67	0.54	0.60

Fonte: A autora

Obteve-se uma acurácia alta para a classificação de DO contra negros, porém as outras métricas (precisão, *recall* e *f-score*), para os casos positivos, foram abaixo do esperado. Acredita-se que isso se deve aos poucos exemplos obtidos para treinamento e teste. Já para a classificação de DO contra as mulheres também obteve-se uma margem de acerto satisfatória, e para os casos positivos, as outras métricas (precisão, *recall* e *f-score*) foram maiores em comparação com o outro contexto.

Os resultados obtidos para os casos positivos do DO contra mulheres se aproximam dos resultados apresentados no projeto de França e Oliveira (2014), esse trabalho correlato também tem sua base de dados extraída do *Twitter* e também utiliza o algoritmo de *Naive Bayes* para a classificação. França e Oliveira (2014), dentro do seu contexto de análise de protestos políticos, obtiveram para sua classe positiva os valores de 90%, 79%, 87% e 83%, já para a classe negativa obteve 72%, 85%, 77% e 81% para as métricas de acurácia, precisão, *recall* e *f-score*, respectivamente.

Utilizou-se o termo tendência ou não tendência a discurso de ódio devido a complexidade do discurso disseminado nesse tipo de rede e ao fato do presente trabalho não avaliar as mensagens semanticamente, visto que o método utilizado não contempla esse tipo de análise, sendo assim, as mensagens rotuladas como tendem a discurso de ódio são aquelas que representam discurso de ódio em potencial.

7 Conclusão

Esse trabalho tinha como objetivo geral propor, implementar e avaliar um processo capaz de extrair dados do *Twitter* e identificar as mensagens com tendência a discurso de ódio, utilizando as etapas da Mineração de Texto. Através da conclusão dos objetivos específicos: (i) Descrever um processo para identificação de termos que denotem tendência a discurso de ódio; (ii) Desenvolvimento de um extrator e pré-processador para os dados do *Twitter*; e (iii) Desenvolvimento de um módulo para classificação de *tweets*, juntamente a execução das etapas da mineração de texto, o objetivo do trabalho foi alcançado.

Devido a classificação de *tweets* ser uma tarefa não producente para seres humanos visto a grande quantidade de mensagens e sua variabilidade, esse trabalho trouxe a contribuição de realizar um processo semi automatizado capaz de realizar essa tarefa incluindo a extração, armazenamento, pré-processamento e classificação dos dados. A solução proposta ainda precisa de alguns processos manuais, como avaliar as strings de busca usadas na coleta de dados e rotular os dados a serem utilizados no treinamento do classificador. Processos manuais também estiveram presentes no trabalho de França e Oliveira (2014) onde existiu a necessidade da rotulação manual dos exemplos a serem classificados, e no trabalho de Dias e Becker (2016) onde um inspetor manual precisa analisar os n-gramas (termos) mais relevantes para indicar o posicionamento das mensagens.

O processo proposto de extração e classificação de mensagens com tendência a discurso de ódio contra negros e contra mulheres teve como maior dificuldade a análise do discurso disseminado nas redes, pois muitas vezes palavras de cunho descriminatório e ofensivo são usadas como se fossem brincadeira porém essa utilização não deixa de causar danos ao alvo do discurso, seja um dano direto ou por meio do fortalecimento de estigmas sociais, o que mostrou a importância de campanhas que busquem conscientizar os usuários como o Movimento Contra o Discurso de Ódio[1].

Contudo o processo mostrou um potencial evolutivo e possíveis pontos de melhoria como ajustes da coleta de dados para DO contra negros, visando obter mais exemplos representativos, pois, embora a acurácia do classificador tenha sido satisfatória as outras métrica avaliadas evidenciaram a necessidade de dispor de uma base de dados com mais exemplos para um melhor treinamento e teste do classificador.

A fim de otimizar a solução foram identificadas algumas tarefas para serem abordadas em trabalhos futuros, como: extensão do processo para outros cenários de discurso de ódio, automatização do processo de seleção de strings que denotem discurso de ódio, avaliação e implantação de diferentes extratores de *features*, e avaliação e implantação de

[1] http://www.odionao.com.pt/

diferentes classificadores.

Acredita-se que um outro caminho a ser seguido para alcançar melhores resultados é a construção de um classificador com base em regras semelhante ao descrito no trabalho de Dias e Becker (2016).

Referências

AGGARWAL, C. C.; ZHAI, C. **A survey of text classification algorithms**. In: *Mining text data*. [S.l.]: Springer, 2012. p. 163–222. Citado 4 vezes nas páginas 33, 35, 36 e 52.

ARAGÃO, A. ***Protesto gera 100 mil tuítes por hora; citações negativas superam panelaço***. [s.n.], 2015. Disponível em: <//www1.folha.uol.com.br/poder/2015/03/1603246-protesto-gera-100-mil-tuites-por-hora-citacoes-negativas-superam-panelaco.shtml>. Acesso em: 08.04.2016. Citado na página 21.

ARANHA, C. N.; VELLASCO, M. **Uma abordagem de pré-processamento automático para mineração de textos em português: sob o enfoque da inteligência computacional**. 2007. Citado na página 33.

BECKER, K.; TUMITAN, D. **Introdução à Mineração de Opiniões: Conceitos, Aplicações e Desafios**. *Simpósio Brasileiro de Banco de Dados*, 2013. Disponível em: <http://www.inf.ufrgs.br/~kbecker/lib/exe/fetch.php?media=minicursosbbd_versaosubmetida.pdf>. Acesso em: 23.05.2016. Citado na página 39.

BENEVENUTO, F.; ALMEIDA, J.; SILVA, A. S. **Explorando redes sociais online: Da coleta e análise de grandes bases de dados às aplicações**. *Mini-cursos do Simpósio Brasileiro de Redes de Computadores (SBRC)*, 2011. Disponível em: <http://www.cricte2004.eletrica.ufpr.br/anais/sbrc/2011/files/mc/mc2.pdf>. Acesso em: 22.05.2016. Citado 5 vezes nas páginas 25, 26, 27, 28 e 40.

BIRD, S.; KLEIN, E.; LOPER, E. ***Natural language processing with Python: analyzing text with the natural language toolkit***. [S.l.]: "O'Reilly Media, Inc.", 2009. Citado na página 52.

CORREIO24H. **Baiano é preso por racismo contra atriz Taís Araújo e Maju em Brumado**. 2016. Disponível em: <http://www.correio24horas.com.br/detalhe/bahia/noticia/baiano-e-preso-por-racismo-contra-atriz-tais-araujo-e-maju-em-brumado/?cHash=e338b4be4050e9816634798c3e211198>. Acesso em: 20.02.2017. Citado na página 48.

DIAS, M.; BECKER, K. **Detecção semi-supervisionada de posicionamento em tweets baseada em regras de sentimento**. In: . [S.l.: s.n.], 2016. Citado 3 vezes nas páginas 39, 63 e 64.

DâMASO, L. **Linha do Tempo do Orkut: relembre vida e morte da rede social do Google**. 2015. Disponível em: <http://www.techtudo.com.br/noticias/noticia/2014/09/linha-do-tempo-do-orkut-relembre-vida-e-morte-da-rede-social-do-google.html>. Acesso em: 25.04.2017. Citado na página 43.

ELLISON, N. B. et al. **Social network sites: Definition, history, and scholarship**. *Journal of Computer-Mediated Communication*, Wiley Online Library, v. 13, n. 1, p. 210–230, 2007. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1111/j.1083-6101.2007.00393.x/epdf>. Acesso em: 22.05.2016. Citado na página 25.

FRANÇA, T. C. de; OLIVEIRA, J. **Análise de Sentimento de Tweets Relacionados aos Protestos que ocorreram no Brasil entre Junho e Agosto de 2013**. In: *Proceedings of the III Brazilian Workshop on Social Network Analysis and Mining (BRASNAN)*. [S.l.: s.n.], 2014. p. 128–139. Citado 7 vezes nas páginas 36, 38, 40, 41, 52, 62 e 63.

FREITAS, R. S. D.; CASTRO, M. F. D. **Liberdade de Expressão e Discurso do Ódio: um exame sobre as possíveis limitações à liberdade de expressão**. *Sequência (Florianópolis)*, Programa de Pós-Graduação em Direito da Universidade Federal de Santa Catarina, p. 327–355, 2013. Citado 2 vezes nas páginas 22 e 29.

GO, A.; BHAYANI, R.; HUANG, L. **Twitter sentiment classification using distant supervision**. *CS224N Project Report, Stanford*, v. 1, n. 12, 2009. Citado 2 vezes nas páginas 38 e 40.

GOMES, H.; NETO, M. de C.; HENRIQUES, R. **Text Mining: Sentiment analysis on news classification**. In: IEEE. *Information Systems and Technologies (CISTI), 2013 8th Iberian Conference on*. [S.l.], 2013. p. 1–6. Citado na página 38.

JúNIOR, V. S. **Acessando a API REST do Twitter**. 2014. Disponível em: <https://pythonhelp.wordpress.com/2014/09/07/acessando-a-api-rest-do-twitter/>. Acesso em: 20.02.2017. Citado na página 48.

LAROSE, D. T. **Discovering knowledge in data**. Wiley, 2005. Citado na página 31.

LEE, H. D. ***Seleção e construção de features relevantes para o aprendizado de máquina.*** Tese (Doutorado) — Universidade de São Paulo, 2000. Citado na página 53.

LIU, B. ***Web Data Mining: exploring hyperlinks, contents, and usage data***. [S.l.]: Springer Science & Business Media, 2007. Citado 6 vezes nas páginas 31, 32, 33, 37, 38 e 62.

MORGENSTERN, F. **O racismo contra Joaquim Barbosa no Twitter**. 2012. Disponível em: <http://www.implicante.org/blog/o-racismo-petista-contra-joaquim-barbosa-no-twitter/>. Acesso em: 20.02.2017. Citado na página 48.

NAZIR, A. et al. **Network Level Footprints of Facebook Applications**. In: *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*. New York, NY, USA: ACM, 2009. (IMC '09), p. 63–75. ISBN 978-1-60558-771-4. Disponível em: <http://doi.acm.org/10.1145/1644893.1644901>. Citado na página 27.

NOVA/SB. **Intolerâncias nas redes: Um problema crescente**. 2016. Disponível em: <http://www.comunicaquemuda.com.br/dossie/intolerancia-nas-redes/>. Acesso em: 04.03.2017. Citado na página 42.

NOVA/SB. **Intolerâncias Visíveis e Invisíveis no Mundo Digital**. 2016. Disponível em: <http://18628-presscdn-0-21.pagely.netdna-cdn.com/wp-content/themes/comunica/dist/dossie/dossie_intolerancia.pdf>. Acesso em: 04.03.2017. Citado 4 vezes nas páginas 21, 41, 42 e 43.

PEDREGOSA, F. et al. **Scikit-learn: Machine Learning in Python**. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 53.

PROPMARK. **Pesquisa mostra horários de pico no Twitter, Facebook e Instagram** . 2016. Disponível em: <http://propmark.com.br/midia/pesquisa-mostra-horarios-de-pico-no-twitter-facebook-e-instagram>. Acesso em: 23.02.2017. Citado na página 50.

RUSSELL, M. A. ***Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More***. [S.l.]: "O'Reilly Media, Inc.", 2013. Citado na página 47.

SILVA, R. L. da et al. **Discursos de ódio em redes sociais: jurisprudência brasileira**. *Revista direito GV*, SciELO Brasil, v. 7, n. 2, p. 445–467, 2011. Citado 4 vezes nas páginas 21, 29, 43 e 44.

TAN, A.-H. et al. **Text mining: The state of the art and the challenges**. In: *Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases*. [S.l.: s.n.], 1999. v. 8, p. 65–70. Citado na página 32.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. ***Introdução ao Data Mining: Mineração de Dados***. [S.l.]: "Ciência Moderna", 2009. Citado 6 vezes nas páginas 31, 33, 34, 35, 36 e 37.

THEOPHILO, M. R. B. **Liberdade de expressão e proteção dos direitos humanos na internet: reflexos do discurso de ódio nas redes sociais e a ação# Humanizaredes**. 2015. Citado 2 vezes nas páginas 28 e 29.

TWITTER. **O que são os marcadores (símbolos de "#")?** 2017. Disponível em: <https://support.twitter.com/articles/255508>. Acesso em: 07.04.2017. Citado na página 26.

TWITTER. **Perguntas frequentes sobre Retweets (RT)**. 2017. Disponível em: <https://support.twitter.com/articles/263102>. Acesso em: 07.04.2017. Citado na página 26.

UOL. **Justiça condena universitária por preconceito contra nordestinos no Twitter**. 2012. Disponível em: <https://tecnologia.uol.com.br/noticias/redacao/2012/05/16/justica-condena-universitaria-por-preconceito-contra-nordestinos-no-twitter.htm>. Acesso em: 21.05.2017. Citado na página 29.

VICENTIM, J. **Web 1.0, Web 2.0 e Web 3.0... Enfim, o que é isso?** Março 2013. Disponível em: <http://www.ex2.com.br/blog/web-1-0-web-2-0-e-web-3-0-enfim-o-que-e-isso/>. Acesso em: 12.04.2016. Citado na página 21.

Apêndices

APÊNDICE A – Classe do extrator de dados do *Twitter*

Código A.1: Classe do extrator de dados

```
1  from twitter import *
2  import re
3
4
5  class Extrator:
6
7
8      def __init__(self):
9          # Chave utilizada para que a aplicação se identifique perante o
       Twitter.
10         CONSUMER_KEY= ''
11         # Chave para a aplicação provar sua autenticidade
12         CONSUMER_SECRET = ''
13         #Após a identificação é preciso que a aplicação envie esse código
       para
14         #  que o serviço possa identificar qual o nível de acesso que ela
       possui;
15         OAUTH_TOKEN = ''
16         #Chave para a aplicação provar sua autenticidade com relação ao
       Access Token
17         OAUTH_TOKEN_SECRET = ''
18         self.api = Twitter(auth=OAuth(OAUTH_TOKEN, OAUTH_TOKEN_SECRET,
19             CONSUMER_KEY, CONSUMER_SECRET))
20         self.lista_tweets = []
21
22
23     def busca(self, termo, idioma, quantidade):
24         tweets_string = self.api.search.tweets(
25             q=termo, lang=idioma, count=quantidade)
26
27         for resultado in tweets_string["statuses"]:
28             # Expressão regular para identificar tweets marcados como
       retweets
29             retweet = re.search('RT ', resultado["text"])
30             # Se o tweete em questão não for um retweet ele é adicionado a
31             # lista de tweets a ser utilizada
32             if retweet == None:
33                 self.lista_tweets.append((resultado["id_str"],
34                                           resultado["created_at"],
```

```
35                                          resultado["text"],
36                                          resultado["user"]["location"]))
```

APÊNDICE B – Classe dos termos utilizados na busca

Código B.1: Classe dos termos utilizados na busca

```
class StringBusca:

    def __init__(self):
        self.n_gramas_contra_negros = ["atendido por cotista",
                                       "cabelo ruim",
                                       "cotas raciais",
                                       "cotista desgraçado",
                                       "esse preto crl",
                                       "filme cotista",
                                       "gente negra",
                                       "gente preta",
                                       "lugar de negra",
                                       "lugar de negras",
                                       "lugar de negro",
                                       "lugar de negros",
                                       "lugar de preta",
                                       "lugar de pretas",
                                       "lugar de preto",
                                       "lugar de pretos",
                                       "merece mulher negra",
                                       "merece mulher preta",
                                       "não gosto de negro",
                                       "não gosto de negros",
                                       "não gosto de pretos",
                                       "não gosto de preto",
                                       "nega macaca",
                                       "nega",
                                       "nego macaco",
                                       "nego",
                                       "negra banana",
                                       "negra cotista",
                                       "negra feia",
                                       "negra imunda",
                                       "negra macaca",
                                       "negra vitima",
                                       "negra vitimismo",
                                       "negras cotista",
                                       "negras macaca",
                                       "negro banana",
```

```
40                                      "negro cotista",
41                                      "negro feio",
42                                      "negro imundo",
43                                      "negro macaco",
44                                      "negro merece",
45                                      "negro vitima",
46                                      "negro vitimismo",
47                                      "negros cotistas",
48                                      "negros macacos",
49                                      "odeio negro",
50                                      "odeio negros",
51                                      "odeio pretos",
52                                      "oscar cotista",
53                                      "por que negro",
54                                      "por que preto",
55                                      "povo negro",
56                                      "povo preto",
57                                      "pq preto",
58                                      "pq negro",
59                                      "preta banana",
60                                      "preta cotista",
61                                      "preta feia",
62                                      "preta imunda",
63                                      "preta macaca",
64                                      "pretas cotistas",
65                                      "pretas macacas",
66                                      "preto banana",
67                                      "preto cotista",
68                                      "preto fedido",
69                                      "preto feio",
70                                      "preto imundo",
71                                      "preto macaco",
72                                      "preto merece",
73                                      "pretos cotistas",
74                                      "pretos macacos",
75                                      "racismo",
76                                      "racista",
77                                      "racistas",
78                                      "só podia ser negro",
79                                      "só podia ser preto"]
80
81          self.n_gramas_contra_mulheres = ["depois reclama assédio carnaval",
82                                           "depois reclama assédio",
83                                           "essa cachorra",
84                                           "essa puta",
85                                           "essa vadia",
86                                           "essa vagabunda",
```

```
87                                          "estuprada",
88                                          "estuprei",
89                                          "estupro",
90                                          "falta de rola",
91                                          "fecha as pernas",
92                                          "feminazi",
93                                          "feminista",
94                                          "feministas",
95                                          "lixo feminista",
96                                          "lugar de mulher",
97                                          "mal comida",
98                                          "merece ser estuprada",
99                                          "mulher aborto",
100                                         "mulher apanha homem",
101                                         "mulher cachorra",
102                                         "mulher é tudo",
103                                         "mulher lavar uns pratos",
104                                         "mulher merece porrada",
105                                         "mulher merece tiro",
106                                         "mulher piranha",
107                                         "mulher puta",
108                                         "mulher que está pedindo",
109                                         "mulher que ta pedindo",
110                                         "mulher que",
111                                         "mulher vadia",
112                                         "mulher vagabunda",
113                                         "mulher varrer uma casa",
114                                         "nessa vagabunda",
115                                         "respeita as mina",
116                                         "respeitaasmina",
117                                         "respeite as mina",
118                                         "sapatão feminista"]
```

APÊNDICE C – Módulo de extração de dados do *Twitter*

Código C.1: Módulo de extração de dados do *Twitter*

```
from time import sleep
from datetime import datetime
from extrator import Extrator
from stringBusca import StringBusca
import csv

rodada_atual = 0
rodada_total = 5
quantidade = 100
idioma = "pt"

strings = StringBusca()

# Laço de repetição responsável por garantir a execução das rodadas com intervalo
#  de duas horas
while rodada_atual <= rodada_total:
    print("Extração de tweets contra negros " + datetime.now().isoformat('T'))
    print("Rodada de coletas: " + str(rodada_atual) + " de " + str(rodada_total))

    # Laço de repetição para executar a busca para cada string dentro da lista
    #  de strings contra negros
    for n_grama in strings.n_gramas_contra_negros:
        # Abre arquivo CSV para salvar os resultados da busca para a string atual
        with open('./Contra_Negros/' + str(datetime.now().day) + "_" +
              str(datetime.now().month) + "_" + str(datetime.now().year) +
              "_" + n_grama + '.csv', 'a', encoding='utf-8') as f:
            # Instancia a Classe Extrator
            twitter = Extrator()
            #Chama a função de Busca
            twitter.busca(n_grama, idioma, quantidade)
            # Instancia objeto que formata os dados passados por
            # parâmetro, para dados separados por valores
            writer = csv.writer(f, dialect=csv.excel_tab)
            # Escreve no arquivo criado anteriormente
```

```
35              writer.writerows(twitter.lista_tweets)
36
37      print("Extração de tweets contra mulheres " + datetime.now().isoformat('T'))
38
39      # Laço de repetição para executar a busca para cada string dentro da lista
40      #  de strings contra mulheres
41      for n_grama in strings.n_gramas_contra_mulheres:
42          # Abre arquivo CSV para salvar os resultados da busca para a string atual
43          with open('./Contra_Mulheres/' + str(datetime.now().day) + "_" +
44                  str(datetime.now().month) + "_" + str(datetime.now().year) +
45                  "_" + n_grama + '.csv', 'a', encoding='utf-8') as g:
46              # Instancia a Classe Extrator
47              twitter = Extrator()
48              # Chama a função de Busca passando a string de busca,
49              # o idioma e a quantidade
50              twitter.busca(n_grama, idioma, quantidade)
51              # Instancia objeto que formata dados para dados separados por valores
52              writer = csv.writer(g, dialect=csv.excel_tab)
53              # Escreve no arquivo criado anteriormente
54              writer.writerows(twitter.lista_tweets)
55
56      print("Esperando próxima rodada - " + datetime.now().isoformat('T'))
57      rodada_atual = rodada_atual + 1
58      # Faz o código "dormir" por duas horas
59      sleep(7200)
```

APÊNDICE D – Pré-processador dos dados

Código D.1: Classe do pré-processador

```
import csv
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
import re


class PreProcessador:


    def __init__(self):
        # Todas as informações vindas do csv
        self.dados = []
        # Apenas os tweets pré processados
        self.tweets = []
        # Rótulos do tweet
        self.rotulos_tweets = []


    # Função para carregar os dados de um arquivo csv e fazer o
    # pré-processamento dos mesmos
    def normalizar_dados(self, arquivo):
        contador = 0
        # Abre o arquivo csv indicando o encoding como utf-8
        with open(arquivo, 'r', encoding='utf-8') as f:
            # Indica que o arquivo está separado por TAB
            reader = csv.reader(f, dialect=csv.excel_tab)
            for row in reader:
                # Aplica os filtros para retirada de menções, links,
                # emojis, pontuação, stopwords e espaços em branco
                # nas extremidades
                row[2] = self.retirar_stopwords(self.retirar_pontuacao(
                    self.retirar_emoji(self.retirar_link(self.
retirar_mention(
                    row[2])))).lower()).strip()
                # Separa a frase em tokens (palavras) indicando o idioma
                row[2] = nltk.word_tokenize(row[2], 'portuguese')
                # Realiza o processo de stemming
                row[2] = self.stemming(row[2])
                self.dados.append(row)
                contador = contador + 1
```

```
41
42          print("TOTAL TWEETS:" + str(contador))
43
44      def normalizar_dados_nuvem_palavras(self, arquivo):
45          contador = 0
46          # Abre o arquivo csv indicando o encoding como utf-8
47          with open(arquivo, 'r', encoding='utf-8') as f:
48              # Indica que o arquivo está separado por TAB
49              reader = csv.reader(f, dialect=csv.excel_tab)
50              for row in reader:
51                  # Aplica os filtros para retirada de menções, links,
52                  # emojis, pontuação, stopwords e espaços em branco
53                  # nas extremidades
54                  row[2] = self.retirar_stopwords(self.retirar_pontuacao(
55                      self.retirar_emoji(self.retirar_link(self.
    retirar_mention(
56                          row[2])))).lower()).strip()
57                  # Separa a frase em tokens (palavras) indicando o idioma
58                  row[2] = nltk.word_tokenize(row[2], 'portuguese')
59                  self.dados.append(row)
60                  contador = contador + 1
61
62          print("TOTAL TWEETS:" + str(contador))
63
64      # Organização dos dados
65      def get_tweets_e_rotulos(self):
66          self.tweets = self.get_tweets()
67          self.rotulos_tweets = self.get_rotulos()
68
69      # Separa os tweets do restante dos dados presente no arquivo e que
70      # foram carregados na função normalizar_dados. Na função
    normalizar_dados
71      # para a aplicação da função de stemming primeiro é preciso separar o
    texto
72      # do tweet em tokens(palavras), porém para a função que vai fazer a
    extração
73      # de features, para o classificador, posteriormente, é preciso que os
    tokens
74      # antes separados estejam juntos novamente, por isso essa função (
    get_tweets)
75      # foi feita dessa forma.
76      def get_tweets(self):
77          nova_lista = []
78          espaco = " "
79          t = ""
80          for palavras in self.dados:
81              nova_lista.append(t)
```

```
82              t = ""
83              for palavra in palavras[2]:
84                  t = t + palavra + espaco #concatena as palavras de cada
    tweets acrescentando um espaço
85          nova_lista.append(t)
86          nova_lista.remove('')
87
88          return nova_lista
89
90
91      # Separa os rótulos(classes) atribuído a cada tweet na classificação
92      # manual do restante dos dados presente no arquivo e que foram
    carregados
93      # na função normalizar_dados.
94      def get_rotulos(self):
95          rotulos_tweets = []
96          for row in self.dados:
97              rotulos_tweets.append(row[3])
98          return rotulos_tweets
99
100
101     # Filtro para retirar emojis
102     def retirar_emoji(self, expr):
103         regex_emoji = re.compile(u'['
104                                  u'\U0001F300-\U0001F64F'
105                                  u'\U0001F680-\U0001F6FF'
106                                  u'\u2600-\u26FF\u2700-\u27BF]+',
107                                  re.UNICODE)
108         return re.sub(regex_emoji, "", expr)
109
110     # Filtro para retirar menções
111     def retirar_mention(self, expr):
112         regex_mention = re.compile("@([\w])*")
113         return re.sub(regex_mention, "", expr)
114
115     # Filtro para retirar link
116     def retirar_link(self, expr):
117         regex_link = re.compile("http.[\W]*([\w]*.[\w]*[\W][\w]*)")
118         return re.sub(regex_link, "", expr)
119
120     # Filtro para retirar pontuação
121     def retirar_pontuacao(self, expr):
122         regex_pontuacao = re.compile("[^\w ]+")
123         return re.sub(regex_pontuacao, " ", expr)
124
125     # Filtro para retirar stopwords
126     def retirar_stopwords(self, expr):
```

```
127         stopwords = nltk.corpus.stopwords.words('portuguese')
128         for palavra in stopwords:
129             expr = re.sub("\\b(" + palavra + ")\\b", '', expr)
130         return (expr)
131 
132     # Aplica a função de stemming
133     def stemming(self, lista_palavras):
134         stemmer = nltk.stem.RSLPStemmer() #stemming para o idioma português
        do Brasil
135         palavras = [stemmer.stem(p) for p in lista_palavras]
136         return palavras
137 
138     # Imprime os itens de uma lista
139     def print_lista(self, lista):
140         for item in lista:
141             print(item)
```

APÊNDICE E – Módulo de classificação dos *tweets*

Código E.1: Módulo de classificação dos *tweets*

```
1  from sklearn.feature_extraction.text import CountVectorizer
2  from sklearn.naive_bayes import GaussianNB
3  from sklearn import cross_validation
4  from sklearn.metrics import precision_score
5  from sklearn.metrics import confusion_matrix
6  from sklearn.metrics import accuracy_score
7  from sklearn.metrics import recall_score
8  from sklearn.metrics import f1_score
9  from preprocessador import PreProcessador
10 import csv
11
12 arquivo_in_cn = "../AnaliseTweets/contraNegros/contraNegros_08_out.csv"
13 arquivo_in_cm = "../AnaliseTweets/contraMulheres/contraMulheres_08_out.csv"
14
15 arquivo_out_cn = "../AnaliseTweets/contraNegros/contraNegros_clf.csv"
16 arquivo_out_cm = "../AnaliseTweets/contraMulheres/contraMulheres_clf.csv"
17
18 # Instância do pré-processador
19 p = PreProcessador()
20
21 # Carregar os dados do arquivo e faz o pré-processamento para o arquivo
22 #  de dados referente a DO direcionado aos negros
23 p.normalizar_dados(arquivo_in_cm)
24 # Função que separa os tweets e os rótulos
25 p.get_tweets_e_rotulos()
26
27 # Carregar os dados do arquivo e faz o pré-processamento para o arquivo
28 #  de dados referente a DO direcionado às mulheres
29 #p.normalizar_dados(arquivo_in_cn)
30 #p.get_tweets_e_rotulos()
31 #p.print_lista(p.rotulos_tweets)
32
33 # Divide o conjunto de dados para treinamento e teste do classificador,
34 # de acordo com o parâmetro test_size (que nesse caso está indicando
35 # uma divisão de 30% para o conjunto de teste e 70% para o treinamento)
36 X_treino, X_teste, Y_treino, Y_teste = cross_validation.train_test_split(
37     p.tweets, p.rotulos_tweets, test_size=0.4, random_state=0)
38
39 print("Treino: ", len(X_treino), " - ", len(Y_treino))
```

```
40 print("Teste: ", len(X_teste), " - ", len(Y_teste))
41
42 # Criando o extrator de features, stop words está marcado para
43 # None porque as stopwords já foram retiradas no pré-processamento.
44
45 #Extrator de features para unigrama
46 vectorizer = CountVectorizer(min_df=1, stop_words=None)
47
48 # Treinamento do extrator de features com o conjunto de treinamento
49 X_treino_vec = vectorizer.fit_transform(X_treino)
50 print("Quantidade de features extraídas: ",len(vectorizer.get_feature_names
       ()))
51
52 # Criação do classificador
53 clf = GaussianNB()
54
55 # Treinamento do Classificador
56 clf.fit(X_treino_vec.toarray(), Y_treino)
57
58 # Extraíndo features do conjunto de teste
59 X_teste_vec = vectorizer.transform(X_teste)
60
61 # Executando o classificador com o conjunto de teste
62 pred = clf.predict(X_teste_vec.toarray())
63
64 # Rótulos usados para ordenar os resultados nas funções que retornam
65 #  os valores para avaliação do classificador
66 classes=["0", "1"]
67
68 # Avaliando a matriz de confusão
69 print("Matriz de confusão:")
70 print(confusion_matrix(Y_teste, pred, labels=classes))
71
72 # Avaliando a acurácia
73 print("Accuracy score:")
74 print(accuracy_score(Y_teste, pred))
75
76 # Avaliando a precisão
77 print("Precision score:")
78 print(precision_score(Y_teste, pred, average=None, labels=classes))
79
80 # Avaliando o recall
81 print("Recall score:")
82 print(recall_score(Y_teste, pred, average=None, labels=classes))
83
84 # Avaliando o f-score
85 print("F-score score:")
```

```
86 print(f1_score(Y_teste, pred, average=None, labels=classes))
```

APÊNDICE F – Script para selecionar de forma aleatória 500 *tweets* da base de dados

Código F.1: Script para selecionar 500 *tweets* da base de dados, de forma aleatória, e salvar em um novo arquivo para a classificação manual

```
import csv
import random

entrada = "../AnaliseTweets/nomeArquivo_in.csv"
saida = "../AnaliseTweets/nomeArquivo_out.csv"
size = 500
lista_tweets = []

# Esse trecho abre o arquivo de entrada
with open(entrada, 'r', encoding='utf-8') as fin:
    reader = csv.reader(fin, dialect=csv.excel_tab)
    for row in reader:
        # Pega as primeiras 3 colunas: número
        # identificador, data de criação e texto
        # de cada linha
        t = [row[0], row[1], row[2]]
        # Adiciona cada nova linha a lista de tweets
        lista_tweets.append(t)

# Abre arquivo de saída
with open(saida, 'a', encoding='utf-8') as fout:
    # Gera nova lista randomizada
    nova_lista = random.sample(lista_tweets, size)
    writer = csv.writer(fout, dialect=csv.excel_tab)
    # Salva no arquivo aberto anteriormente
    writer.writerows(nova_lista)
```

APÊNDICE G – Script para limpeza dos dados para a Nuvem de Palavras

Código G.1: Script para limpar o arquivo final dos dados a fim de submete-los a geração de nuvem de palavras

```
from preprocessador import PreProcessador

# Caminho e nome dos arquivos
arquivo_in_cn = "../AnaliseTweets/contraNegros/contraNegros_08_in.csv"
arquivo_in_cm = "../AnaliseTweets/contraMulheres/contraMulheres_08_in.csv"

# Instância do pré-processador
p = PreProcessador()

# Carregar os dados do arquivo e faz o pré-processamento para o arquivo
#  de dados referente a DO que se deseja analisar
p.normalizar_dados_nuvem_palavras(arquivo_in_cn)
p.get_tweets_e_rotulos()
p.print_lista(p.tweets)
```