A detecção de implicaturas conversacionais da ironia em textos de redes sociais através do Aprendizado de Máquina para português

Rayssa Küllian Martins

Dissertação apresentada
ao
Instituto de Matemática e Estatástica
da
Universidade de São Paulo
para
Obtenção do Título
de
Mestre em Ciências

Programa: Ciência da Computação
Área de Concentração: Inteligência Artificial
Orientador: Prof. Dr. Marcelo Finger

São Paulo, abril de 2018

A detecção de implicaturas conversacionais da ironia em textos de redes sociais através do Aprendizado de Máquina para português

Esta é a versão original da dissertação elaborada pela candidata Rayssa Küllian Martins, tal como submetida à Comissão Julgadora.

Agradecimentos

Quero agradecer à Deus por ter me sustentado e permitido chegar até aqui. Também agradeço ao meu esposo Gabriel, minha família e amigos, foram quatro anos de muitas lutas e sacrifícios em meio à muitas pausas inesperadas.

Faltam palavras para agradecer também ao Prof. Dr. Marcelo Finger por todo o conhecimento transferido e principalmente pela compreensão que teve durante esta jornada, é um privilégio tê-lo como orientador.

Deixo registrada minha eterna gratidão, não teria conseguido sem o apoio de todos vocês.

Resumo

KÜLLIAN, R. **A detecção de implicaturas conversacionais da ironia em textos de redes sociais através do Aprendizado de Máquina para português.** 2018. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2018.

A ironia é uma expressão de linguagem comumente utilizada e interpretada por seres humanos. Porém, esta simplicidade não é perceptível quando se trata de um diálogo textual, tornando complexa sua definição formal e consequente detecção. Ao treinar um modelo de classificação preditiva para realizar Análise de Sentimento em conversas em redes sociais ou avaliações de produtos em sites de comércio eletrônico, *e.g.,* exemplos rotulados binariamente não são suficientes e tem seu desempenho acentuadamente reduzido quando um usuário é irônico. Por este motivo, a ironia textual se torna um ruído ou um inversor de polaridade no classificador e o problema é agravado quando se trata do idioma Português, onde não existem córpus públicos anotados para estas ocorrências.

Este trabalho apresenta um estudo relacionado à ironia do ponto de vista da Linguística Computacional, abordando desde a discussão acerca de sua definição até nuances implícitas do texto e sugestões de como processá-las. A complexidade deste tema é abordada no decorrer do texto e seus desafios peculiares são evidenciados através de exemplos, apontando, inclusive, possíveis lacunas de pesquisa.

A proposta desta pesquisa é apresentar um conjunto de técnicas de Aprendizado de Máquina e Processamento Natural de Linguagem para realizar a detecção automática de ironias textuais, tendo como principal aplicação a detecção aplicada às opiniões postadas publicamente no Twitter utilizando a *hashtag #metrosp* no contexto de metrô e trens da CPTM na cidade de São Paulo, Brasil.

Duas abordagens são comparadas ao longo de 51 experimentos e 900 testes: a classificação de componentes linguísticos com Processamento de Língua Natural através do texto dos *tweets*, e a classificação de novos atributos que representam este texto a partir de principais características identificadas na análise exploratória. O melhor desempenho foi encontrado com a utilização do algoritmo de Bayes com 96% de f1 na classificação de atributos em uma base balanceada de 538 *tweets*, cujo desempenho também foi o mais estável com uma média de 0.8014 em todos os experimentos realizados.

Palavras-chave: Computação Linguística, Processamento Natural de Linguagem, Mineração de Textos, Análise de Sentimento, Análise Preditiva, Aprendizado de Máquina, Sistemas Colaborativos, ironia textual, redes sociais.

Abstract

KÜLLIAN, R. **The detection of irony's conversational implicatures in social media through Machine Learning applied to Portuguese language**. 2018. Dissertation (MSc) - Institute of Mathematics ans Statistics, University of Sao Paulo, Sao Paulo, 2018.

Irony is an expression language commonly used and interpreted by human beings. However, this simplicity is not quite distinguishable when talking about a textual dialogue, making it puzzling to formally define and detect. When training a predictive classification model to provide Sentiment Analysis in social media chats or product reviews on any retail website, *e.g.*, binary labeled records are not enough and has its performance sharply reduced when some customer is ironic. Hence, textual irony becomes a noise or a polarity inverter within the classifier and the issue gets worse when the language is Portuguese, where there is no public labeled corpus for these scenario.

This project presents a study of works related to irony in the Computational Linguistics point of view, approaching its philosophical concept, textual implied nuances and proposals for how to process it. The complexity of this subject is presented in the course of the text and its peculiar challenges are pointed out through examples, also indicating possible research gaps.

The proposal of this research is to offer a set of Machine Learning and Natural Language Processing techniques to accomplish the automatic detection of textual irony, whereas Twitter's reviews will be its main application and the main goal is to collect all tweets with the hashtag *#metrosp* for the context of subways in the city of Sao Paulo, Brazil.

Two approaches are compared across 41 experiments and 900 tests: the classification of linguistic components with Natural Language Processing using only the text of the tweets, and the classification of new attributes built to represent this text with its main identified characteristics during the exploratory analysis. The best performance was found using the Bayes algorithm with 96% of f1 when classifying the attributes on a balanced training set of 538 tweets, which performance was also the most stable one with a average of 0.8014 in all the performed experiments.

Palavras-chave: Computational Linguistics, Natural Language Processing, Text Mining, Sentiment Analysis, Predictive Analysis, Machine Learning, Groupware, textual irony, social media.

Sumário

Lista de Abreviaturas

ABSA	Aspect-based Sentiment Analysis (*Análise de Sentimento Baseada em Aspectos*)
API	Application Programming Interface (*Interface de Programação de Aplicação*)
AWS	Amazon Web Services
NER	Named-entity Recognition (*Reconhecimento de Entidades Nomeadas*)
NN	Neural Networks (*Redes Neurais*)
NLP	Natural Language Processing (*Processamento de Linguagem Natural*)
ML	Machine Learning (*Aprendizado de Máquina*)
SGD	Stochastic Gradient Descent (*Gradiente Descendente Estocástico*)
SVM	Support Vector Machines (*Máquinas de Vetores de Suporte*)
TF-IDF	Term Frequency–Inverse Document Frequency (*Termo–Inverso da Frequência nos Documentos*)
TM	Text Mining (*Mineração de Texto*)

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Atualmente temos uma grande quantidade de dados em crescimento exponencial proveniente de fontes como arquivos, bancos de dados, sensores, mídias sociais, Internet das Coisas, entre outros, sejam estruturados, semi-estruturados ou não-estruturados. A previsão é de que a quantidade de dados dobrará a cada dois anos até 2020, quando teremos 44 zettabytes ou 44 trilhões de gigabytes (Villars *et al.*, 2011). De nada adianta acumularmos dados se não extraírmos nenhuma informação e nenhum valor for agregado. Estima-se que 23% dos dados (643 exabytes) existentes no universo digital em 2012 seriam úteis se estivessem rotulados (Gantz e Reinsel, 2012).

Grande parte destes dados, especialmente em redes sociais, são textos e requerem o processamento de uma língua natural, uma atividade complexa já que cada língua possui sua própria gramática e características culturais específicas. A ideia de possibilitar o processamento de língua natural através de computadores é tão antiga quanto os próprios computadores (Jurafsky e James, 2000). Uma das premissas para a sua realização é a identificação da estrutura de uma sentença, um processo chamado de análise sintática ou parseamento. Em seguida, se faz necessário a representação do conhecimento adquirido e, outra etapa também conhecida como *inferência*, responsável pela interpretação do significado representado.

A maioria das atividades que envolvem conhecimento linguístico com este tipo de processamento pode ser vista como uma tentativa de resolver inúmeros problemas de ambiguidade (Jurafsky e James , 2000), seja *léxica*, *sintática*, *semântica* ou de qualquer outra natureza. Um dos tipos de ambiguidade mais complexos e, até por este motivo foi muito pouco pesquisado no âmbito computacional, é a ironia. Diferente de ambiguidades comuns, onde uma palavra pode ser tanto um substantivo como um verbo e, assim, alterar o significado semântico da sentença como um todo, a ambiguidade da ironia se dá pelo fato de que a mesma frase pode significar o que realmente representa ou justamente a sua negação, além de que outros fatores textuais e contextuais podem colaborar para a sua identificação.

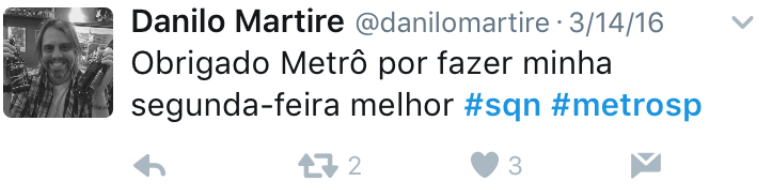

Figura 1.1: *Tweet irônico de um usuário do Metrô de São Paulo.*

Este cenário tem se consolidado no universo digital através da interação de pessoas nas redes sociais. Neste projeto iremos focar na detecção de implicaturas conversacionais da ironia em Português do Brasil, onde temos uma expressão característica: *"Só que não"*. Antes utilizada como um marcador de refutação, *e.g.,* "Levei advertência, *só que não* quero mostrar", agora tem sido utilizada no final das sentenças, causando um efeito de releitura e reinterpretação da sentença que a precede. Uma *implicatura conversacional* nada mais é do que um acessório linguístico, onde as

pessoas utilizam elementos linguísticos com valores que não fazem parte dos seus sentidos literais (Martelotta, 2014). Na Figura 1.1, por exemplo, podemos observar um usuário relatando sua opinião sobre o funcionamento do Metrô de São Paulo em uma segunda-feira. Apesar de agradecer ao Metrô por *"fazer sua segunda-feira melhor"*, é utilizada uma *hashtag* *#sqn* no final do tweet, um indício de uma expressão de ironia e um comportamento definido por Kreuz (1996) como o *princípio da inferabilidade*, onde afirma que as pessoas utilizam este tipo de conotação apenas em situações onde tenham uma certeza significativa de que serão compreendidas corretamente e o uso da *hashtag* deste exemplo enfatiza esse comportamento. Na Figura 1.2, podemos perceber a ironia através da imagem postada pela usuária Ana Paula, além da *hashtag* #sqn.

Figura 1.2: *Tweet reportando problemas na Linha Verde do Metrô através da ironia.*

Com o objetivo de manter o foco na mineração de textos e não em processar imagens, outros componentes figurativos também foram identificados: formas diminutivas, pontuações, aspas duplas, expressões de risos e emojis (Carvalho *et al.*, 2009), como vemos na irônica forma de relatar o erro em um anúncio do Oscar 2017 através da *hashtag* e também emojis na Figura 1.3. Vale ressaltar que a ênfase é dada à ironia *verbal* e não *situacional*. Da mesma forma, abordamos a ironia partindo do princípio da negação *contraditória* e não *contrária*. As definições e distinções destes termos também serão mencionadas no decorrer do texto.

Este trabalho se propõe a detectar os diferentes componentes da ironia textual em *tweets* através de técnicas de Aprendizado de Máquina e Processamento de Língua Natural. Serão realizados comparativos de desempenho entre diferentes algoritmos, comparando suas métricas para encontrar aquele que melhor se adapta a este cenário específico.

1.1 Motivação

Capturar a estrutura de uma sentença e atribuir um significado a ela pode parecer uma atividade trivial quando mentalmente realizada. Porém, estabelecer regras sintáticas, semânticas e, também neste caso, pragmáticas, que traduzam seu processo cognitivo e tornem possível a realização através de uma máquina faz com que toda a simplicidade desapareça. É fácil identificar uma ironia durante uma conversa, mas o que necessariamente caracteriza uma ironia? Quais aspectos gramaticais e contextuais possibilitam a identificação de um trecho textual irônico?

Estima-se que entre textos classificados como *positivos* em experimentos de Análise de Sentimento, cerca de 35% das instâncias classificadas erroneamente foram causadas por ironia de acordo com experimento realizado por Carvalho *et al.* (2009). Um classificador treinado para identificar

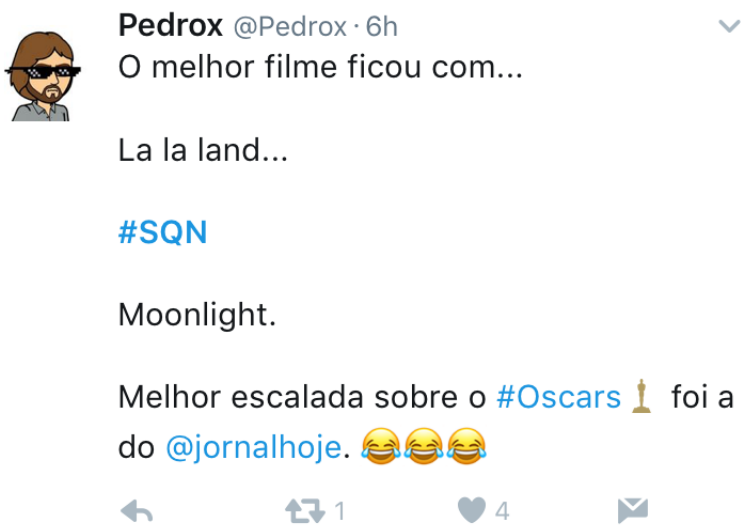

Figura 1.3: *Tweet relatando erro no anúncio de vencedor do Oscar 2017.*

classes binárias (e.g., *positivo* ou *negativo*), não consegue lidar com textos irônicos, uma vez que irá replicar o trabalho para o qual foi treinado e classificar apenas o valor explícito da sentença, e não a sua contradição. Pode ser que consiga classificar uma instância com o valor oposto à sua classificação treinada, porém, será por um erro na classificação e não por ter conseguido capturar as nuances implícitas da sentença.

Através deste trabalho, pretendemos reduzir estes ruídos de classificação e obter melhores resultados de Análise de Sentimento.

1.2 Objetivos

Este projeto de pesquisa possui os seguintes objetivos:

- Identificar características sintáticas, semânticas e pragmáticas de implicaturas conversacionais da ironia.
- Construir uma base de ironias textuais em Português do Brasil.
- Desenvolver um classificador de ironias do Twitter através das *hashtags* descritas neste trabalho.
- Avaliar o desempenho dos algoritmos de Aprendizado de Máquina utilizando as métricas propostas neste trabalho.

1.3 Contribuições Pretendidas

Além de validar a hipótese de detectar automaticamente implicaturas conversacionais da ironia, este trabalho entregará as seguintes contribuições:

- Base pública de *tweets* do Metrô e CPTM em São Paulo em Portugês do Brasil.
- Base pública rotulada de ironias no âmbito do Metrô e CPTM em São Paulo em Portugês do Brasil.

1.4 Organização do Trabalho

No Capítulo 2, apresentamos breves definições dos conceitos utilizados e necessarários para o desenvolvimento e entendimento deste trabalho, além das tecnologias envolvidas. No Capítulo 3 são levantadas as características, estratégias, abordagens e discussões de trabalhos de pesquisa similares desenvolvidos em outros idiomas. No Capítulo 4, apresentamos os detalhes da metodologia utilizada para a execução deste projeto. Por fim, os resultados são apresentados no Capítulo 5 e discutidos no Capítulo 6, onde também são mencionados os próximos passos de pesquisa sugeridos.

Capítulo 2

Conceitos

Este capítulo tem por objetivo definir os conceitos essenciais para o entendimento deste trabalho, bem como apresentar as áreas de pesquisa relevantes para sua execução.

2.1 Ironia

De acordo com o dicionário, a ironia *"é uma expressão que dá a entender, em determinado contexto, o contrário ou algo diferente do que significa"*. É quando alguém diz *"claro que vou!"* quando na verdade está explicitando e enfatizando sua falta de vontade de ir, por exemplo. Compreender este tipo de expressão é uma tarefa relativamente simples, mas até mesmo os seres humanos não conseguem capturá-la em determinados momentos. Na retórica clássica, a ironia é definida como um *tropo*, isto é, o emprego de uma palavra em sentido figurado. Em uma metáfora, o sentido figurado é uma analogia ao signifiado real, já na ironia é o oposto do sentido literal (Wilson, 2006). Também podemos dividir a ironia em dois termos distintos: a ironia *situacional*, também conhecida por alguns como *ironia do destino*, e a ironia *verbal*.

Uma situação pode ser irônica quando as circunstâncias a levam a um desfecho inesperado e a ironia pode ser encontrada mesmo sem comunicação verbal. Geralmente estes casos são caracterizados pelo uso de expressões como *"é irônico que..."*, porém as expressões em si não são irônicas, apenas a situação descrita (Utsumi, 1996), ao mesmo tempo que a sentença realmente representa o significado literal expressado.

Já a ironia conversacional, a mais usual, é classicamente definida como *"o processo retórico de intencionalmente utilizar palavras para expressar um significado diferente ao que teria se utilizado literalmente"* (Carvalho *et al.*, 2009), geralmente o significado *oposto*. Porém, é importante ressaltar que, apesar desta definição genérica, a ironia pode ser caracterizada por diversos *"acessórios figurativos"*, como sarcasmo, perguntas retóricas, *emojis*, pontuações, interjeições, entre outros, e não puramente pela identificação do significado oposto de uma sentença.

2.2 Negação

A negação é definida como o ato de negar, ou seja, afirmar que algo não é verdadeiro. Na lógica, existem diversas formas de negações. O *quadrado lógico* (ou *quadrado das oposições*) da lógica aristotélica apresenta diferentes tipos de oposições e é necessário diferenciar dois casos específicos para situar a abordagem deste trabalho: a negação *contrária* e a negação *contraditória*.

A negação contraditória é a relação entre duas proposições (*e.g.*, p e $\neg p$) de forma que p exclue $\neg p$. De acordo com Johnson (1987), *"contradições têm valores exatamente opostos"*, ou seja, se A é verdadeiro então B é falso, e vice-versa. Esta definição também pode ser encontrada no trabalho de Gensler (2002) onde afirma-se que as sentenças *"nenhum A é B"* e *"algum A é B"* são contraditórias, pois ao dizer que uma é falsa assume-se que a outra é verdadeira.

Já a lógica da contrariedade define que duas sentenças são contrárias caso não possam ser ambas verdadeiras, mas possam ser ambas falsas. Ou seja, se A e B não podem ser verdadeiras, deduz-

se que caso uma seja verdadeira a outra deve ser falsa. Porém, dada uma sentença falsa não há como inferir nada sobre o valor da outra (Johnson, 1987). Diferente do que ocorre na contradição, onde podemos descobrir o valor de uma variável a partir de outra variável em qualquer tipo de circunstância, pois sempre será o valor oposto, como no exemplo mencionado anteriormente *(p e* $\neg p$*)*.

A ironia é um caso linguístico de negação que deve ser tratado de acordo com a lógica contraditória e, caso uma sentença seja irônica, devemos assumir o seu valor irônico como verdadeiro e a sua contradição é automaticamente considerada falsa. Em outras palavras, ou uma sentença é irônica ou não é irônica, os dois casos não podem ser verdadeiros ao mesmo tempo. Já o escopo da negação de uma sentença pode ser delimitado de diversas formas, conforme será apresentado na Subseção 3.2.

Considerar apenas aspectos linguísticos da ironia, como a negação, pode não ser suficiente para detectá-la em textos de redes sociais. Diversos outros componentes podem aparecer, como exclamações, *emojis*, aspas, entre outros. Diferentes abordagens serão levantadas na Seção 3.3.

2.3 Sarcasmo

O dicionário Aurélio define sarcasmo como *"uma ironia amarga e insultuosa"*. De acordo com Joshi *et al.* (2016), é uma forma de ironia que *"tende a ridicularizar"*. Muitos consideram que a ironia é uma das principais características do sarcasmo, porém o sarcasmo é mais *forte* ou *intenso*. Camp (2012) introduz quatro tipos:

- **Proposicional**: a implicatura é exatamente o contrário daquilo que o enunciado deveria expressar, *e.g.,* "*Adoro a CPTM e seus trens congelantes.*"
- **Léxico**: valor composto invertido para uma única expressão, *e.g.,* "*Se o metrô continuar funcionando bem dessa forma, o Alckimin com certeza vai se eleger para presidente.*"
- **"Como se"pré-fixado**: enfática negação epistêmica de um enunciado declarativo, *e.g.,* "*Como se eu andasse de metrô toda semana.*"
- **Ilocutório**: expressa uma atitude oposta àquilo que o enunciado deveira expresssar, *e.g.,* "*Obrigada aos demais passageiros por segurarem a porta.*"

Todas estas variações *invertem* de alguma forma aquilo que o interloculator pretende dizer na realidade, mas de maneira muito sutil, o que faz com que sua detecção seja tão complexa quanto à da própria ironia. Apresentaremos as abordagens levantadas para lidar com esta situação na Seção 3.4.

2.4 Língua Natural

O propósito de uma linguagem é possibilitar a comunicação e provém do intelecto natural do ser humano (Chomsky, 1975). Searle (1969) diferencia dois aspectos: a *filosofia linguística*, que é a tentativa de resolver problemas filosóficos pelo uso de palavras e outros elementos específicos de uma linguagem, basicamente o nome de um método, e a *filosofia da linguagem*, que aborda a tentativa de prover significados filosóficos de certas características da linguagem como referências, significados, necessidades, entre outros, basicamente o nome do assunto tratado.

Independente da abordagem, ambos assumem a utilização de elementos de linguagens e suas relações, mesmo com a vasta varidade de línguas naturais. O foco deste trabalho é a língua natural do Português do Brasil, apesar das complexidades e limitações que apresentaremos na Subseção 2.4.2.

2.4.1 Processamento de Linguagem Natural

O Processamento de Linguagem Natural é o estudo de modelos matemáticos e computacionais a partir de vários aspectos de linguagem (Joshi, 1991) e este tipo de processamento pode requerer

capacidades simbólicas como a criação e propagação de ligações, manipulação de estruturas recursivas, aquisição de conteúdos léxicos e semânticos, representação de aspectos abstratos (Dyer, 1995). Desde suas primeiras pesquisas em meados de 1950, tem tido seu foco de pesquisa em atividades como tradução, recuperação e extração de informação, sumarização, mineração de opiniões, agentes conversacionais, entre outros.

2.4.2 Cenário da Língua Portuguesa do Brasil

Apesar de ser uma área de pesquisa com muitas aplicações e evoluções, é importante ressaltar a falta de recursos disponíveis na comunidade quando se trata do Português. Não existem córpus de ironias ou dicionários de *part-of-speech* completos deste idioma que não exijam uma customização nas ferramentas e bibliotecas utilizadas, diferentemente do que encontramos em idiomas como o Inglês utilizando *OpenNLP*[1] e *WordNet*[2], por exemplo.

Mesmo em termos de pesquisa, apesar de mencionarmos um trabalho desevolvido em Portugal no Capítulo 3, não existe nenhum outro específico para o Português do Brasil até o momento desta pesquisa. Também não existem trabalhos suficientes, em geral, para debatermos se a ironia diverge em diferentes idiomas.

2.5 Aprendizado de Máquina

Para que um mecanismo computacional seja capaz de classificar informações, por mais estruturadas que estejam, é necessário alimentá-lo com parâmetros de comparação para que seja possível realizar inferências próximas às desejadas. Porém, a identificação de padrões em textos escritos em língua natural, através de métodos comparativos pré-determinados, não são suficientes para prever a infinita gama de formas de expressão possíveis. Para tanto, os classificadores de texto têm por princípio funcional a implementação de algoritmos de Aprendizado de Máquina, responsáveis pelo comportamento lógico dedutivo de aplicações.

O Aprendizado de Máquina é um campo de Inteligência Artificial e, segundo Corrêa *et al.* (2003), utiliza conceitos fundamentais de estatística e heurística avançada para tomar decisões diferentes baseadas nas características dos dados estudados. Nilsson (1965) define uma máquina de aprendizado como *"qualquer dispositivo cujas ações são influenciadas por experiências anteriores"*. De maneira formal, também temos a definição de aprendizado de máquina por Mitchell (1997): *"é dito a um programa de computador para aprender a partir de uma experiência E com respeito a alguma classe de tarefas T e medida de desempenho P, se seu desempenho em tarefas de T, medido por P, melhora com a experiência E"*. Esta subárea da Inteligência Artificial é composta por diferentes algoritmos de classificação, supervisionada ou não-supervisionada, para possibilitar este aprendizado.

Teremos como base as abordagens levantadas no Capítulo 3 com os trabalhos relacionados. A intenção é identificar quais mais se adequam ao cenário de detecção e classificação de ironias e quais delas poderemos adaptar ao Português do Brasil, assim como já fizemos com a *hashtag #sqn* adaptada da *hashtag #not* do Inglês. Se necessário, combinaremos abordagens diferentes ou iremos propor novas, desde que se atinja o objetivo proposto inicialmente.

2.5.1 Tipos de Aprendizado e Algoritmos de Classificação

De acordo com Russell *et al.* (2003), o campo de Aprendizagem de Máquina distingue-se em três casos: aprendizagem supervisionada, não-supervisionada e por reforço. Neste trabalho utilizaremos as duas primeiras abordagens.

O paradigma *supervisionado* é composto pelo aprendizado de padrões e classificações baseados em rótulos fornecidos juntamente com os valores de entrada da base de treinamento. Ou seja, se

[1] https://opennlp.apache.org. Acessado em 07/04/2017.
[2] https://wordnet.princeton.edu. Acessado em 07/04/2017.

temos dois rótulos A e B, a base de treinamento possuirá exemplos de registros rotulados como A e outros exemplos rotulados como B. Desta forma, o algoritmo reproduzirá a classificação especificada em novos registros não rotulados.

No escopo deste paradigma, a aprendizagem estatística descreve que *"os agentes podem manipular a incerteza usando os métodos de probabilidade e teoria da decisão após aprenderem suas teorias probabilísticas do mundo através de suas experiências"* (Russell *et al.*, 2003). A aprendizagem através do teorema de Bayes, base do algoritmo NaiveBayes utilizado neste projeto, calcula a probabilidade de cada hipótese, considerando os dados, e faz previsões de acordo com esta probabilidade. Ou seja, são realizadas previsões com o uso de todas as hipóteses, ponderadas por suas probabilidades, assumindo independência entre elas, ao invés de usar apenas a melhor hipótese. Assim, a aprendizagem é realizada apenas através da inferência probabilística.

$$P(h_i|\mathbf{d}) = \frac{P(\mathbf{d}|h_i)P(h_i)}{P(\mathbf{d})} \tag{2.1}$$

Supondo $\mathbf{D}$ como representação de todos os dados, com valor observado $\mathbf{d}$, a probabilidade de cada hipótese é obtida pela regra observada na Fórmula 2.1.

Outro algoritmo de SVM, ou *Máquinas de Vetores de Suporte*, também fazem parte da aprendizagem estatística. Segundo Smola *et al.* (2000), suas vantagens são uma boa capacidade de generalização, robustez em grandes dimensoões, convexidade da função objetivo uma teoria bem definida.

Este modelo constrói um *hiperplano* para distribuir os dados de forma a otimizar a divisão de classificação, atividade chamada de *kernelling*. Os *vetores de suporte* são os pontos de dados utilizados como base para a classificação.

Outra abordagem é através de um tratamento similar ao do racíocino humano com *Redes Neurais*. Um neurônio é uma célula no cérebro cuja função é coletar, processar e disseminar sinais elétricos. Segundo Russell *et al.* (2003), acredita-se que *"a capacidade de processamento de informações do cérebro emerge principalmente de redes de tais neurônios"* e, por isso, um dos principais trabalhos iniciais da Inteligência Artificial foi criar redes neurais artificiais. Uma rede que possui todas as suas entradas conectadas diretamente as suas saídas é chamada de *rede neural de uma única camada* ou rede *perceptron*. Neste trabalho, foi utilizado o algoritmo *MultiLayerPerceptron*, onde são adicionadas outras camadas e cuja vantagem é aumentar o espaço de hipóteses que a rede pode representar. A Figura 2.1 exemplifica este caso.

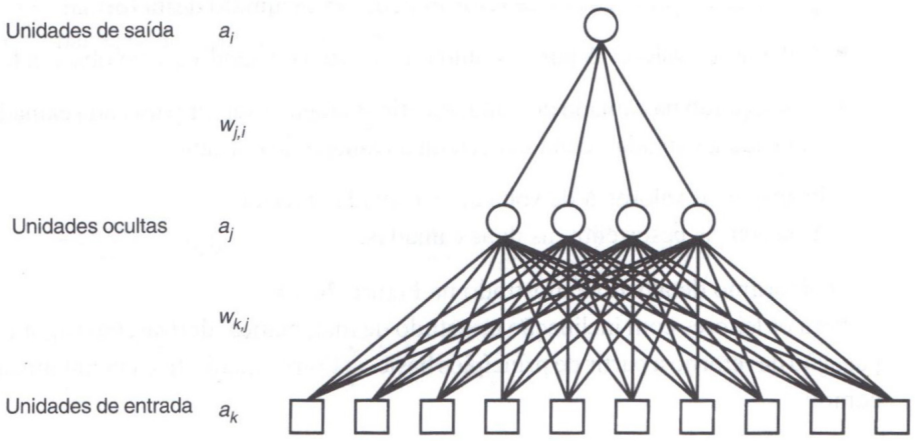

Figura 2.1: *Exemplo de rede neural com várias camadas.*[3]

Já no paradigma *não-supervisiondo* não temos exemplos rotulados, temos apenas registros e a atividade de classificação se baseia em métricas de distância ou similaridade, e o próprio algoritmo é

[3]Fonte: (Russell *et al.*, 2003)

responsável por encontrar a melhor forma de dividir os dados. É importante mencionar que, apesar de gerar uma saída com os dados classificados através de alguma medida, o algoritmo não é capaz de *interpretar* estes resultados. Ou seja, não serão gerados *rótulos* para os dados e a interpretação pode ser necessária.

Nesta categoria, existem utilizados para agrupar os dados de acordo com alguma métrica específica de similaridade, seja através de alguma medida de distância, correlação ou associação. O tipo de algoritmo utilizado neste trabalho para agrupar os dados foi o *k-means*, um algoritmo que utiliza um modelo de otimização onde o usuário especifica um conjunto **k** de partições, chamados *centróides*, e o algoritmo busca exaustivamente pela combinação ótima de k-partições, partindo de um conjunto de centróides iniciais aleatórios. A distância utilizada por este algoritmo é a *euclidiana*.

2.6 Técnicas de Manipulação de Dados

As subseções seguintes descrevem diferentes abordagens para preparar os dados em formatos específicos para algoritmos de Aprendizado de Máquina, conforme descrito na Seção 2.5 anterior.

2.6.1 Pré-Processamento de Textos

Os algoritmos de classificação são capazes de interpretar apenas valores numéricos, ou seja, quaisquer outros valores precisam ser transformados para valores numéricos, como é o caso da língua natural. Dentro deste contexto, algumas atividades pré-definidas são necessárias e são apresentadas nas próximas seções.

A Mineração de Textos é composta por técnicas de Processamento de Linguagem Natural e Aprendizado de Máquina, e atividades de pré-processamento, categorização, extração, armazenamento e visualização de resultados (Feldman e Sanger, 2007). Pode ser definida como a busca por extrair informações úteis de fontes textuais estruturadas, semi-estruturadas ou não-estruturadas.

Uma das atividades de Mineração de Texto é a Análise de Sentimento, onde tenta se extrair sentimentos expressados em um texto, podendo ser classificada em polaridade, emoções, opiniões, atitudes, entre outros. Com o crescimento da utilização de redes sociais e Sistemas Colaborativos, sua aplicação tem se popularizado na academia e indústria por possibilitar a automatização da interpretação humana.

Independente da abordagem utilizada, algumas etapas de pré-processamento de texto são comuns (Jurafsky e James, 2000) para lidar com a língua natural, conforme revisado na Seção 2.4.1 e com as complexidades da Língua Portuguesa mencionadas na Seção 2.4.2. A primeira delas é a remoção de *stop words*, que são palavras que não acrescentam significado semântico às frases, como por exemplo as palavras *"que", "de", "para"*, entre outras.

Em seguida, temos a atividade de *tokenização*, onde cada palavra de uma frase é dividida e representada por *tokens*. A frase *Este metrô está horrível hoje!* seria representada por um *vetor de palavras*: *"esta, cidade, está, horrível, hoje"* onde as letras também seriam convertidas para minúsculas e as pontuações removidas.

Diversas palavras possuem derivações gramaticais. O verbo *organizar*, por exemplo, também pode ser variar como *"organizando", "organizado", "organizei", "organizaram"*, etc, além de também possuir uma representação como o substantivo *"organização"*. Porém, todas estas diferentes palavras derivam de uma mesma palavra com o mesmo significado semântico. Por este motivo, uma das principais atividades executadas no tratamento de textos é *reduzir* as palavras para as suas *raízes gramaticais*.

Ainda por este motivo, existe outra abordagem conhecida como etiquetagem de palavras (*part of speech*) para possibilitar a diferenciação de *categorias sintáticas*, onde cada palavra recebe uma *etiqueta* com a representação de sua classe gramatical. A combinação das palavras e suas respectivas etiquetas também foi experimentada neste trabalho para comparar diferentes abordagens.

2.6.2 TF-IDF

O conceito de Termo–Inverso da Frequência nos Documentos, conhecido por *TF-IDF*, se trata de uma abordagem estatística para descrever a relevância de um termo em um conjunto de documentos.

$$TF(t) = \frac{quantidade\ de\ \mathbf{t}\ no\ documento}{total\ de\ termos\ no\ documento} \tag{2.2}$$

A frequência de um termo **t** é calculada dividindo-se sua frequência pelo número total de termos em um conjunto de documentos, conforme mostra a Fórmula 2.2 (Salton e Buckley, 1988). Porém, os termos mais frequentes não são necessariamente os mais *relevantes*. Considere a palavra *"que"* da língua Portuguesa. É uma das palavras mais frequentes no vocabulário, porém, não tem relevância semântica na maioria dos casos. Por este motivo, apenas a frequência de um termo não é suficiente para calcular sua relevância e justamente os termos *menos* utilizados tem uma maior probabilidade de terem um significado semântico dentro de um contexto de vários documentos.

$$IDF(t) = log_e(\frac{quantidade\ de\ documentos}{total\ de\ documentos\ com\ \mathbf{t}}) \tag{2.3}$$

A Fórmula 2.3 é responsável por esta estratégia que tem sido amplamente adotada em cenários de mineração de textos.

2.6.3 Análise de Componentes Principais

Não apenas a quantidade de dados gerados e explorados tem crescido consideravelmente, mas também a quantidade de atributos de complexas bases de dados multi-dimensionais, e diversas estratégias tem sido pesquisadas para suportar sua análise através da visualização de dados.

A redução de dimensões é uma das principais técnicas para possibilitar o suporte com a visualização, onde usamos uma transformação linear para projetar dados multi-dimensionais em um espaço de baixa dimensão (Liu *et al.*, 2015). Um dos métodos mais clássicos é o de Análise de Componentes Principais, conhecido por *PCA*, onde é utilizada uma matriz de covariância dos dados para encontrar a transformação linear ortogonal que maximize sua variância (Jolliffe, 1986).

2.6.4 Validação Cruzada

Quando se estuda profundamente um cenário específico de modelagem para produzir um classificador, pode ocorrer um problema conhecido desde 1930 (Larson, 1931) como *overfitting*, onde a modelagem foi tão adaptada às características dos dados em questão que o modelo se ajusta rigorosamente à uma função de aprendizado e, caso novos dados sejam apresentados, eles não farão parte desta função e o classificador será incapaz de prever o rótulo correto. Ou seja, o algoritmo pode possuir bons resultados de treinamento e ainda assim possuir um desempenho ruim nos testes com novos registros.

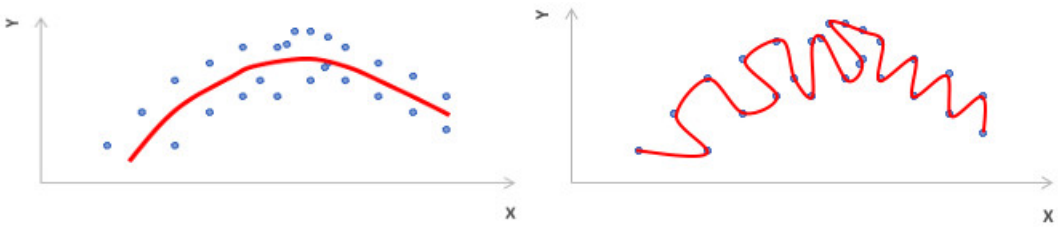

Figura 2.2: *Cenário com overfitting.*

Na maioria das vezes existe apenas uma quantidade limitada de dados rotulados para utilização, o que faz com que este problema seja ainda mais comum. Mesmo que se divida bases diferentes

de treino e teste com o rótulo ocultado, pode ser que mesmo as atividades de pré-processamento possam enviesar os dados e facilitar este cenário. A Figura 2.2 mostra um comparativo entre um cenário de aprendizado e o mesmo cenário com o problema de *overfitting*.

A técnica conhecida por Validação Cruzada foi criada para lidar com este tipo de problema e consiste na divisão da base de dados dispoível em mais conjuntos de testes e validações, ao invés de apendas dois para treino e testes (Arlot *et al.*, 2010). Isto faz com que mais testes sejam realizados com diferentes porções dos dados, minimizando a possibilidade de *overfitting*.

2.6.5 Métricas de Avaliação

Existem diversas métricas de avaliação dos algoritmos de Inteligência Artificial descritos na Subseção 2.5.1, mas todas se baseiam na matriz de confusão, uma matrix que combina o número de classificações corretas e incorretas para cada classe predita, conforme o exemplo da Figura 2.3.

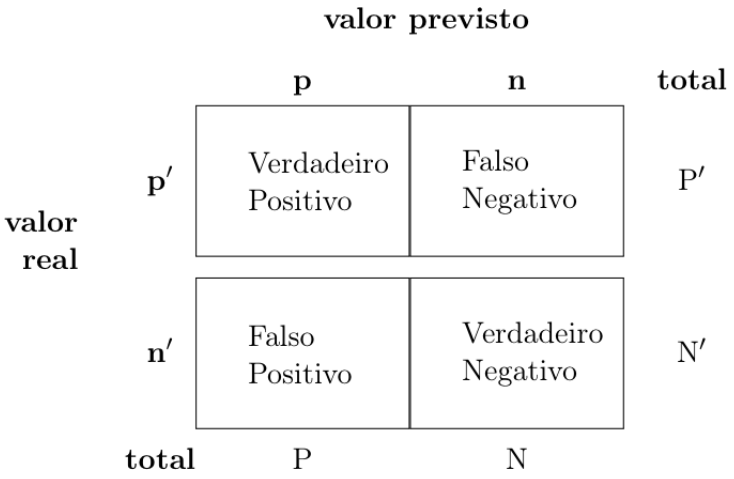

Figura 2.3: *Exemplo de uma matrix de confusão.*

Existem duas métricas principais para análise a partir desta matrix: a *cobertura* demonstrada na Fórmula 2.4 que considera a proporção de exemplos positivos classificados corretamente entre tudo o que foi classificado como positivo, incluindo os erros de falso-positivos, e a *precisão* demonstrada na Fórmula 2.5 que considera a proporção de exemplos positivos classificados corretamente entre o que realmente era positivo e também aquilo que deveria ser positivo, ou seja, os falso-negativos.

$$cobertura = \frac{VP}{VP + FN} \tag{2.4}$$

$$\text{precisão} = \frac{VP}{VP + FP} \tag{2.5}$$

Em outras palavras, a cobertura é a proporção de elementos relevantes recuperados entre o número total de elementos recuperados, enquanto a precisão é a proporção de elementos relevantes recuperados entre o total de elementos relevantes. Uma como uma medidade de *especificidade* e outra como uma medida de *sensibilidade*.

É necessário também considerar uma terceira métrica que representa a média harmônica entre precisão e cobertura, chamada de *F1* e representada na Fórmula 2.6:

$$F1 = 2 * \frac{\text{precisão} * cobertura}{\text{precisão} + cobertura} \tag{2.6}$$

2.7 Tecnologias

Para a implementação deste projeto de acordo com a metodologia apresentada no Capítulo 4, será necessário utilizar ferramentas, bibliotecas e serviços. Nesta seção iremos apresentar cada uma destas tecnologias e explicar brevemente sua responsabilidade.

2.7.1 Serviços

2.10.1.1 S3

O *Simple Storage Service*, conhecido como S3[4], é um serviço de armazenamento em nuvem da *Amazon Web Services* (AWS) que possui um alto nível de disponibilidade, acessível através de uma simples interface demonstrada na Figura 2.5.

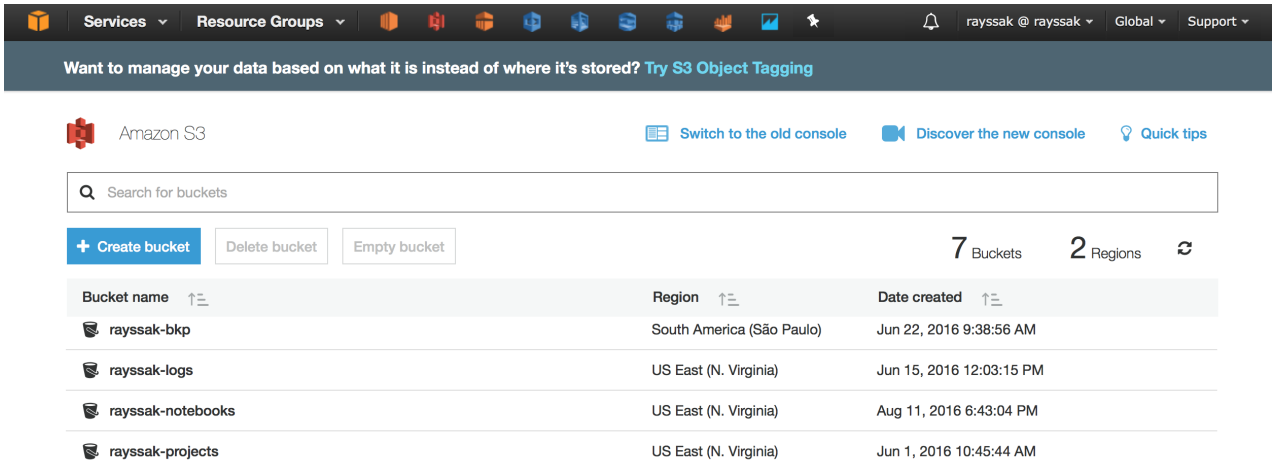

Figura 2.4: *AWS S3.*

2.10.1.2 Glacier

Apesar de também ser um serviço de armazenamento em nuvem da AWS, o *Glacier*[5] se diferencia por ter um custo mais acessível já que seu objetivo é armazenar *backup* de dados. Neste caso, ele é utilizado para armazenar arquivos não utilizados frequentemente, como dados históricos, por exemplo.

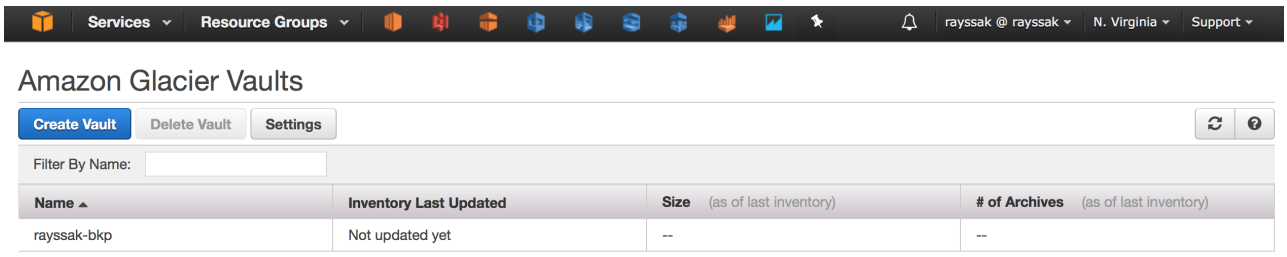

Figura 2.5: *AWS Glacier.*

2.10.1.3 Amazon SageMaker

[4] https://aws.amazon.com/s3/. Acessado em 07/04/2017.
[5] https://aws.amazon.com/glacier/. Acessado em 07/04/2017.

O *Amazon SageMaker*[6], é um serviço que disponibiliza um ambiente pré-configurado de desenvolvimento em nuvem através do *Jupyter*, apresentado na próxima Seção 2.7.2, onde são disponibilizadas diversas bibliotecas de inteligência artificial comuns na comunidade. A Figura 2.6 mostra a interface deste serviço.

Nele é possível configurar uma máquina, chamada de *instância*, de acordo com os recursos computacionais necessários, e executá-la remotamente na nuvem ao invés de utilizar os recursos de sua máquina local.

Figura 2.6: *Jupyter notebook no Amazon SageMaker.*

2.10.1.4 Lambda

O *Lambda*[7] é um serviço para execução de código sem o provisionamento de máquina e qualquer outro tipo de configuração, você simplesmente submete um arquivo com seu código de programação para ser executado. Atualmente o serviço suporta as linguagens Python, Java e Node.js e possui uma limitação de tempo de execução de código de cinco minutos. A Figura 2.7 mostra um exemplo de uma função Lambda em Node.js.

2.7.2 Ferramentas e Bibliotecas

2.10.2.1 Jupyter

O *Jupyter*[8] é uma aplicação web de código aberto que possibilita a criação de documentos, chamados de *notebooks*, com código *"vivo"*, equações, gráficos, textos descritivos, entre outros, conforme ilustrado na Figura 2.8.

É a principal interface de desenvolvimento utilizada pela comunidade de ciência de dados e apoiada por diversos empresas como Google, Microsoft, IBM, O'Reilly, NASA e diversas universidades como Berkeley, Northwestern, NYU.

2.10.2.2 Spark

[6] https://aws.amazon.com/sagemaker/. Acessado em 06/03/2018.
[7] https://aws.amazon.com/lambda/. Acessado em 07/04/2017.
[8] http://jupyter.org. Acessado em 07/04/2017.

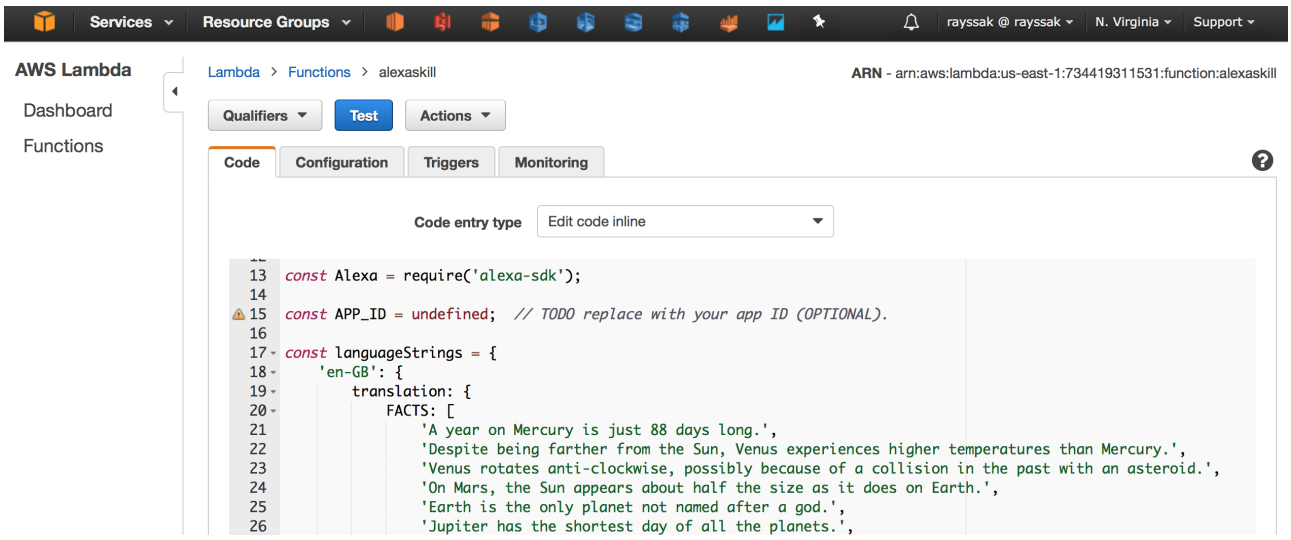

Figura 2.7: *AWS Lambda.*

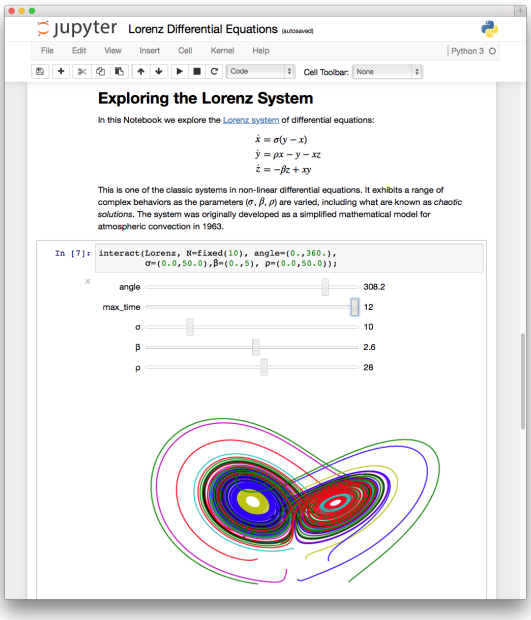

Figura 2.8: *Exemplo de um notebook no Jupyter.*

O *Spark*[9] é uma ferramenta para processamento distribuído desenvolvido inicialmente na Universidade da Califórnia, *Berkeley AMPLab*, em 2009, e atualmente mantido pela Apache Software Foundation.

Ele é composto por cinco módulos: *Spark SQL* para o processamento de SQL, *Spark.ML* para o aprendizado de máquina, *GraphX* para o processamento de grafos e *Spark Streaming*. Suporta as linguagens de programação Java, Scala, Python e R.

2.10.2.3 Pandas

O *Pandas*[10] é uma biblioteca para manipulação, processamento e visualização de dados em Python muito utilizada na comunidade e possui as seguintes principais características:

[9] http://spark.apache.org. Acessado em 07/04/2017.
[10] https://pandas.pydata.org. Acessado em 07/04/2017.

- Objetos chamados de *DataFrame* para manipulação de dados com índice integrado;
- Ferramentas para a leitura e escrita de dados estruturados em memória e diferentes formatos de arquivos;
- Alinhamento de dados e tratamento integrado de dados inválidos;
- Remodelamento de bases de dados;
- Funcionalidade para realizar funções de *join* em bases de dados;
- Manipulação de séries temporais.

2.10.2.4 Weka

O *Weka* é uma biblioteca de código aberto sob a licença *GNU General Public License* com algoritmos de Aprendizado de Máquina, desenvolvida na Nova Zelândia, que contém ferramentas específicas para as seguintes atividades (Witten *et al.*, 2016) :

- pré-processamento;
- classificação;
- regressão;
- clusterização;
- visualização.

2.10.2.5 OpenNLP

A *OpenNLP*[11] é uma biblioteca de código aberto de Aprendizado de Máquina com enfoque no Processamento de Língua Natural mantida pela *Apache Software Foundation*. Suporta apenas as linguagens de programação Java e C++, mas também possui uma interface de fácil utilização. Possui suporte às atividades mais comuns de NLP, como:

- tokenização;
- segmentação de sentenças,
- rotulamento de *part-of-speech*;
- extração de entidades (NER);
- *chunking*;
- parseamento.

2.10.2.6 NLTK

A NLTK[12] é uma biblioteca de código aberto para o Processamento de Língua Natural amplamente utilizada pela comunidade. Também possui suporte às mesmas atividades executadas pela biblioteca anterior, *OpenNLP*, porém suporta apenas uma linguagem de programação: Python.

2.10.2.7 Scikit-learn

[11] https://opennlp.apache.org. Acessado em 07/04/2017.
[12] http://www.nltk.org. Acessado em 07/04/2017.

O scikit-learn[13] é uma biblioteca de algoritmos de Aprendizado de Máquina em Python, construído com base em outras bibliotecas populares na comunidade, como *Numpy*[14], *SciPy*[15], *matplotlib*[16], entre outros, sob a licença *Berkeley Software Distribution*. Suporta as seguintes atividades e algoritmos:

- **Classificação**: SVM, Random Forest, Bayes, SGD;
- **Regressão**: SVR, Ridge Regression, Lasso;
- **Clusterização**: k-means, mean-shift, spectral;
- **Redução de Dimensões**: PCA, matrix de fatorização;
- **Seleção de Modelo**: grid search, validação-cruzada;
- **Redução de Dimensões**: PCA, matrix de fatorização;
- **Pré-processamento**: tokenização, *stemming*, extração de *features*.

[13] http://scikit-learn.org/stable/. Acessado em 07/04/2017.
[14] http://www.numpy.org. Acessado em 07/04/2017.
[15] https://www.scipy.org. Acessado em 07/04/2017.
[16] https://matplotlib.org. Acessado em 07/04/2017.

Capítulo 3

Trabalhos Relacionados

O objetivo deste capítulo é apresentar as abordagens levantadas através dos trabalhos relacionados pesquisados.

3.1 Histórico

Por conter diversos aspectos implícitos em sua caracterização, a ironia por si só representa um grande desafio para a Linguística Computacional. Conseguir capturar nuances contrárias ao que um texto diz, enquanto se processa o signficado literal deste mesmo texto, é uma tarefa complexa em qualquer idioma.

A falta de referências para esta pesquisa demonstra a magnitude deste desafio. Uma curiosidade a respeito dos trabalhos levantados é que resumem-se a idealização de modelos para a detecção de ironia através de sua caracterização, porém muitos deles nem sequer foram aplicados computacionalmente, apenas propostos.

Mesmo que se consiga mapear e formalizar a lógica de um texto irônico, com suas características linguísticas e componentes figurativos, ainda existirão exceções, situações não mapeadas ou até mesmo novas formas de expressão. Da mesma forma como existem casos em que nós, seres-humanos, não captamos a sutileza da ironia, é preciso ter consciência de que também teremos limitações daquilo que se pode concretizar computacionalmente. Uma pessoa pode escrever um trecho irônico sem utilizar nenhuma propriedade textual descrita neste trabalho.

A premissa da ironia parte do princípio da negação, abordada em detalhes na Seção 3.2, foi utilizada como a única propriedade para a sua detecção durante algum tempo, mas concluiu-se que suas características vão muito além deste simples caso de negação e envolvem outras propriedades textuais que detalharemos na Seção 3.3 deste capítulo. Até mesmo sua definição filosófica é abordada de diferentes maneiras e ainda gera certa confusão. Podem também existir casos onde a delicada e sútil distinção entre *ironia* e *sarcasmo* seja importante. Também traremos detalhes de como abordar este cenário na Seção 3.4 a seguir.

Considerando um trecho irônico, todas as abordagens levantadas requerem técnicas mais refinadas e dependentes da interpretação dos demais elementos linguísticos relacionados, de forma que uma análise a *nível de documento* não é apropriada para este tipo de situação, mas ao mesmo tempo uma análise a *nível de sentença* talvez também não seja suficiente. A melhor estratégia pode então requerer a análise do trecho textual como um todo para captar nuances específicas no contexto. A maior parte dos trabalhos relacionados analisa textos curtos, como é caracterizado o conteúdo de um *tweet*.

Além destas complexidades, foram realizadas poucas pesquisas neste assunto. Ainda mais preocupante é o fato de que não existe nenhum trabalho para o Português brasileiro (pt-BR) até o momento desta pesquisa, o que também pode ser uma grande oportunidade de estudo. É difícil acreditar que um trabalho iniciado em 1991 por Littman e Mey ainda não foi pesquisado no Brasil ou não foi divulgado devidamente.

O primeiro trabalho nesta linha de pesquisa, porém focado apenas na negação propriamente

dita, foi realizado na Universidade Federal de Minas Gerais (UFMG) em 1999 por Vitral que, apesar de concluir que as etapas de mudança linguística previstas pela negação são visíveis para o sistema computacional, foi pesquisado com aplicação no Inglês. Existe apenas um trabalho de ironia propriamente dita para o Português (Carvalho *et al.*, 2009), concentrado nas características de Portugal (pt-PT), cujos resultados foram apresentados em um artigo em Inglês sem mencionar detalhes profundos do modelo de classificação utilizado.

As próximas seções apresentam os trabalhos mencionados, apesar de terem sido aplicados ao Inglês em sua maioria.

3.2 Ironia Por Negação

Os experimentos realizados por Wiegand *et al.* (2010) apresentam uma maneira de identificar a ironia através apenas da negação, apesar de representá-la de diferentes maneiras para determinar qual parte da expressão é modificada pela presença da negação (*e.g.*, a sentença inteira ou as duas palavras seguintes a negação). Um dos experimentos identifica apenas a polaridade do verbo para determinar o número de argumentos com que ele pode ser combinado, como no Exemplo 1 demonstrado abaixo. Por polaridade defino o ato de representar uma palavra por seu valor semântico *positivo* ou *negativo*, de acordo com o contexto no qual foi aplicada. O verbo *amar* possui valor positivo e o *odiar* um valor negativo, por exemplo.

1. Eu **gosto**$^{+}$ *de sorvete*
2. $[$*esperança*$^{+}$ *perdida*$^{-}$$]^{-}$

Em um outro modelo mais elaborado proposto, três características são combinadas: *negação*, *shifters* (mais fracos do que uma negação comum, mas ainda importantes em termos de polaridade, *e.g.*, *"pouco"*) e *modificações de polaridade* (descrevem expressões que modificam ou são modificadas por expressões de polaridade, como demonstrado acima no Exemplo 2).

Em outro momento, são combinados também propriedades como *intensificadores* e *diminuidores* para determinar um *score* de intensidade para a negação. Esta abordagem pode ser útil para distinguir casos como *"não tão ruim"* e *"não tão bom"* que, apesar de terem a mesma polaridade, diferem na intensidade (*"não tão ruim"* é menos positivo do que *"não tão bom"*). Porém, em muitos momentos isto não é suficiente para identificar a polaridade de uma sentença. Foi então apresentada uma abordagem para a *resolução de conflitos de polaridade* que nada mais é do que uma semântica composicional, agrupando as seguintes características:

- Representação sintática das sentenças;
- Cálculo de palavras de negação (contador);
- Expressões de polaridade (subdividido em palavras cuja intensidade de negação é suficiente para alterar ou a polaridade da frase ou o constituinte).

Ainda no trabalho mencionado, são sugeridas técnicas de Aprendizado de Máquina para a detecção da negação. O autor não entra em muitos detalhes, mas destaca que uma abordagem de classificação utilizando *bag of words* pode não ter um desempenho aceitável por conter palavras "soltas" (uma unidade individual de representação de conhecimento, sem conexões) e a classificação de polaridade ser realizada nestas palavras soltas, o que faz com que se perca a ligação entre a negação e os constituintes, impossibilitando a identificação do alcance da negação na sentença. Uma abordagem que adicione um rótulo negativo como prefixo das palavras $NÃO\_x$, como no Exemplo 3, também não é capaz de modelar o escopo da negação:

3. *"Eu não* $NÃO\_$*gosto* $NÃO\_$*do* $NÃO\_$*novo* $NÃO\_$*celular* $NÃO\_$*da* $NÃO\_$*Nokia."*

Outro problema desta abordagem é no caso de caso um documento grande com cerca de vinte expressões de polaridade, mas apenas uma ou outra negação. Não faria diferença em termos de classificação a *nível de documentos*. Abordagens mais refinadas, como em (Wilson *et al.*, 2005), sugerem classificações a *nível de expressões* para evitar este tipo de problema.

3.3 Ironia Além da Negação

Um simples exemplo de como palavras compostas que contenham polarização, *e.g.*, *"não-aplicável"*, consegue demonstrar como a detecção de ironia através apenas da negação começa a se tornar uma tarefa um tanto quanto delicada. Diferente de uma expressão de polaridade, onde temos diferentes palavras com valores polares diferentes que se anulam e resultam em outro resultado, palavras compostas formam uma única palavra que deve conter um único valor polar. Desta forma, depender unicamente da negação torna-se inviável e os pouquíssimos trabalhos desenvolvidos na linha desta pesquisa se voltaram a interpretação de outros elementos textuais que complementam esta característica básica da ironia.

No trabalho desenvolvido na Universidade de Lisboa (Carvalho *et al.*, 2009), algumas *dicas* foram enumeradas para complementação, definidas como na Tabela 3.1:

Padrão	Descrição
P_{dim}	formas diminutivas (*e.g., "futebolzinho"*)
P_{dem}	pronomes demonstrativos (*e.g., "**esse** Lula"*)
P_{itj}	interjeições (*e.g., "muito obrigado, hein?"*)
P_{cross}	adjetivos modificam um substantivo através da preposição *"de"* (*e.g., "o comunista **do** ministro"*)
P_{punct}	pontuações (*e.g.*, sequencia de exclamações ou interrogações)
P_{quote}	aspas duplas, *especialmente* se o conteúdo tiver polarização positiva
P_{laugh}	expressões de risos e *emoticons* (*e.g.*, LOL, LMAO, " :) ", " ;-) ")

Tabela 3.1: *Componentes textuais da ironia.*

Dados estes padrões, um experimento foi realizado utilizando como base Conteúdo Gerado Pelo Usuário (*User-Generated Content*) de um jornal popular em Portugal, contendo 8.211 notícias, cerca de 250.000 comentários de usuários que totaliza cerca de um milhão de sentenças. Este conteúdo foi coletado durante um período de cinco meses e possui enfoque em política. A Tabela 3.2 contabiliza os padrões utilizados e demonstra o desempenho de cada abordagem. Cenários sem nenhuma contagem como o P_{dim} não são abordagens propícias para testes, enquanto P_{quote} e P_{laugh} devem ser mais estudados.

Outro trabalho mais recente de 2014 também propõe a detecção de uma ironia além da negação (Reyes e Rosso, 2013), porém as propriedades textuais levantadas pouco diferem das já apresentadas, apesar de serem mais detalhadas através de *camadas* distintas de identificação de padrões. As únicas propriedades descritas e não apresentadas anteriormente são chamadas de *compressão temporal* e *suavidade*. A *compressão temporal* tem por objetivo identificar elementos relacionados à oposição no tempo, termos que sugerem uma mudança brusca na sequência narrativa (*e.g.*, *"inesperadamente" ou "já"*). Já a *suavidade* mede o grau de agrado sugerido por uma palavra (*e.g.*, *amor* é mais agradável do que *dinheiro*).

Padrão	Quantidade
P_{dim}	0
P_{dem}	42
P_{itj}	127
P_{verb}	22
P_{cross}	11
P_{punct}	385
P_{quote}	697
P_{laugh}	548

Tabela 3.2: *Contagem de padrões encontrados.*

3.4 Ironia vs. Sarcasmo

Conforme definido anteriormente na Seção 2.3, o conceito do sarcasmo muito se aproxima ao da própria ironia. Por este motivo, também foram levantados trabalhos relacionados com este enfoque de pesquisa. O levantamento realizado por Joshi *et al.* (2016) traz um comparativo de diversas abordagens utilizadas para lidar este cenário.

Dois tipos de conteúdos foram encontrados: textos *curtos* e *longos*. Dado que este trabalho será realizado no Twitter e existe a delimitação de 140 caracteres, vamos analisar apenas os trabalhos que utilizam textos curtos. Os comparativos então, abordam desde a estratégia de rotulagem, manual ou *automatizada* através de *hashtags*, até o tipo de aprendizado utilizado pelos algoritmos de classificação, *supervisionado* ou *semi-supervisionado*. A Tabela 3.3 compara as abordagens:

referência	aprendizado **supervisionado**	aprendizado **semi-supervisionado**	rotulagem **manual**	rotulagem **automatizada**
González-Ibánez *et al.* (2011)	x			x
Reyes *et al.* (2012)	x			x
Riloff *et al.* (2013)		x	x	
Liebrecht *et al.* (2013)	x			x
Reyes *et al.* (2013)	x			x
Reyes e Rosso (2013)	x			x
Barbieri *et al.* (2014)	x			x
Joshi *et al.* (2015)	x		x	x
Rajadesingan *et al.* (2015)	x			x
Bamman e Smith (2015)	x			x
Ghosh *et al.* (2015)	x	x		x
Bharti *et al.* (2015)		x		x
Bouazizi e Ohtsuki (2015)	x		x	

Tabela 3.3: *Abordagens para o tratamento de sarcasmo.*

Pode se observar que a maioria dos trabalhos faz uso do aprendizado supervisionado e utiliza *hashtags* para rotular exemplos de maneira automatizada. As duas estratégias serão comparadas em detalhes na Seção 4.3.

Alguns trabalhos específicos fogem à regra e, ao invés de utilizar estratégias de Aprendizado de Máquina, se baseiam em um conjunto de regras específicas e características do sarcasmo. Veale e Hao (2010) analisam se um símile é ou não irônico através do resultados de buscas no Google. Já Maynard e Greenwood (2014) possui regras específicas para o tratamento de *hashtags*.

Capítulo 4

Metodologia de Desenvolvimento

Esta seção apresenta a metodologia utilizada neste projeto, descreve detalhadamente a arquitetura construída, qual estratégia foi utilizada em cada uma das atividades, desde a coleta e transformação de dados, a análise exploratória, a engenharia de atributos, a rotulagem de exemplos para o aprendizado supervisionado e a classificação de ironias. Ao longo deste capítulo também são descritas as dificuldades encontradas e as abordagens utilizadas para superá-las.

4.1 Arquitetura

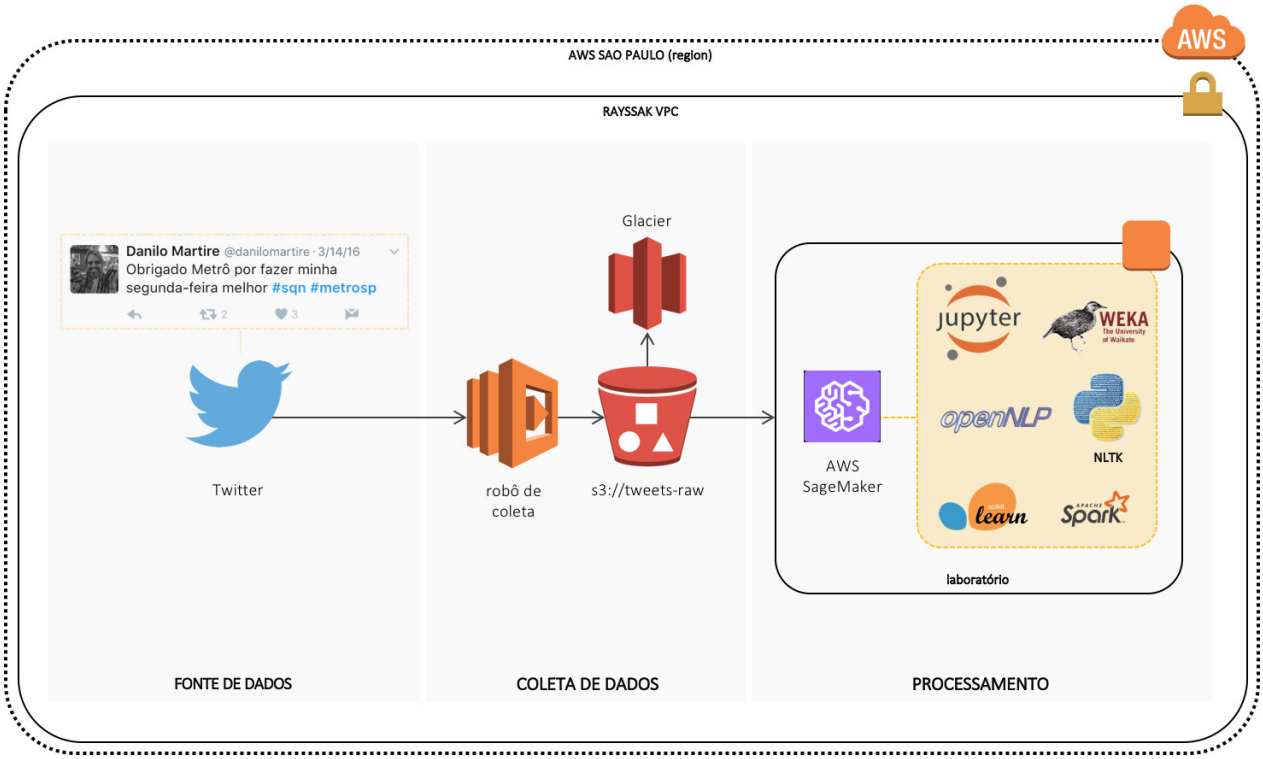

Figura 4.1: *Arquitetura proposta.*

Para atingir o objetivo deste projeto foi necessário escolher algumas tecnologias e ferramentas específicas. A Figura 4.1 retrata a arquitetura construída e o fluxo estabelecido. Todo o ambiente está em uma conta pessoal na nuvem da AWS com a utilização dos serviços dentro dos limites disponibilizados gratuitamente, chamado de *free tier*[1].

Assim que um usuário postar qualquer *tweet* com o conteúdo monitorado, o *tweet* será coletado imediatamente, armazenado no repositório S3, definido na Sub-seção 2.7.1, e enviado ao motor de

[1] https://aws.amazon.com/pt/free/. Acessado em 07/04/2017.

classificação da etapa de processamento. Em seguida, o *tweet* é rotulado binariamente em *irônico* ou *não irônico*, seu rótulo é incrementado ao seu conteúdo e essa saída é disponibilizada para consulta através de uma API.

Temos dois cenários distintos em nosso ambiente de desenvolvimento: um para a execução do robô de coleta de *tweets*, demonstrado nos quadrantes "fonte de dados"e "coleta de dados"da arquitetura, e outro para a execução do mecanismo de classificação propriamente dito, demonstrado no quadrante "processamento". Não há mais necessidade de se executar o robô para a construção de bases de dados para o treinamento dos algoritmos, porém, este fluxo é repetido e contínuo também para o mecanismo de classificação uma vez que continuamos coletando *tweets* para refinar a base de treinamento e também para manter uma base histórica.

Temos um algoritmo em Python descrito na Seção 4.2.1 sendo executado no serviço de execução de código *Lambda*, conforme definido na Sub-seção 2.7.1, e armazenando os *tweets* com a *hashtag* *#metrosp* no repositório S3. Caso os dados não sejam utilizados em tempo real, serão movidos para o Glacier após um período de um mês.

Para o processamento, temos o ambiente de desenvolvimento necessário para este projeto através do serviço *AWS SageMaker*, descrito na Sub-seção 2.7.1, que disponibiliza um *notebook Jupyter* com as bibliotecas necessárias pré-configuradas e apresentadas na Seção 2.7.2.

4.2 Obtenção de dados

4.2.1 Tweets

A primeira atividade para a classificação através do Aprendizado de Máquina é a construção de bases de dados para servirem de entrada do algoritmo, tanto para a etapa de treinamento quanto de testes. Entretanto, não existia nenhuma base pública disponível para esta mineração. Por este motivo, foi construído um robô para coletar *tweets* através da API de *streaming*[2] do Twitter. Não existe nenhuma restrição para extrair as informações necessárias, uma vez que os usuários da rede social consentiram com suas postagens públicas, sem méritos éticos ou jurídicos (Giles *et al.*, 2010).

Após estudo inicial e justificativa apresentada no Capítulo 1, decidiu-se pela utilização da *hashtag* *#metrosp* na coleta de postagens em Português delimitadas no contexto do Metrô e trens em São Paulo. O robô foi construído utilizando a linguagem de programação Python e possui a estrutura demonstrada no Apêndice A.

Foram coletados **329.473** *tweets* durante seis meses: de agosto de 2015 à janeiro de 2016. Cada *tweet* coletado contém informações do usuário, texto e informações da postagem na estrutura JSON do Apêndice B e exemplificada na Figura 4.2:

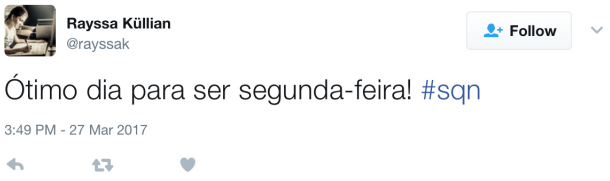

Figura 4.2: *Tweet irônico da estrutura JSON exemplificada.*

4.2.2 Emojis

Após os primeiros testes de classificação descritos na Seção 4.4, identificou-se a possibilidade de criar novos atributos para representar os *emojis* contidos em cada *tweet*. Porém, cada *emoji*, em si, não é interpretado como uma figura, mas como uma sequência de caracteres *unicode*. Utilizamos expressões regulares para capturar estas sequências iniciadas por "\U".

[2] https://developer.twitter.com/en/docs/tutorials/consuming-streaming-data. Acessado em 07/04/2017.

Dada a dificuldade de se mapear todos os *emojis* existentes e seus respectivos sentimentos, foi utilizada uma base de dados[3] contruída por Novak *et al.* . Esta base também foi coletada no *Twitter* e possui **969** *emojis* balanceados igualmente em três classes: *positivo*, *negativo* e *neutro*, bem como os respectivos números de ocorrências de cada *emoji* e os percentuais de probabilidade para cada classe de sentimento.

Utilizamos um total de 200 emojis contendo todos aqueles que possuem pelo menos 100 ocorrências, ou seja, apenas os mais utilizados, com o objetivo de otimizar a performance de nosso código. A lista pode ser consultada no Apêndice C e está exemplificada na Tabela 4.1:

emoji	**unicode**	**ocorrências**	**posição**	**negativo**	**neutro**	**positivo**
😂	0x1f602	14.622	0.805101	3.614	4.163	6.845
categoria	**descrição**					
Emoticons	FACE WITH TEARS OF JOY					

Tabela 4.1: *Base de dados de emojis e respectivos sentimentos.*

4.3 Rotulagem de Exemplos

O primeiro pré-requisito para se treinar um algoritmo de classificação supervisionado é a construção de bases de treinamento e teste, onde o treinamento nada mais é do que um conjunto de exemplos rotulados de acordo com a classificação que se espera do algoritmo. A quantidade total de *tweets* coletados é de **329.473**, conforme mencionado na Seção 4.2.1, e rotular manualmente esta quantidade de exemplos em *tweets irônico* ou *não-irônico* para construir uma base de treinamento adequada levaria muito tempo e, portanto, não foi considerada como uma alternativa viável.

Além disto, ainda contaríamos com o risco de não se conseguir homogenizar as definições de ironia entre diversos colaboradores para minimizar a discrepância pelo juízo pessoal de cada um. É importante também mencionar a possibilidade de inserirmos ironias mesmo em *tweets* aleatórios, bem como *tweets* sem contexto, componentes figurativos e qualquer outro tipo de característica que o algoritmo possa se basear para distinguir entre um *tweet* irônico e um *tweet* não-irônico. Assim, os *tweets* deixariam de ser bons exemplos e se tornaram ruídos na base de treinamento.

Uma abordagem alternativa foi experimentada onde a *hashtag* *#sqn* serviu como o rótulo da classe representando a *ironia* e foi removida para os testes seguintes ao treino servindo como validação das classificações do algoritmo. Um total de **269** *tweets* com a *hashtag* *#sqn* foi automaticamente rotulado como exemplos de *"ironias"* e outros *tweets* aleatórios serviram como exemplos de *"não-ironias"* para o treinamento do classificador. Esta quantidade de 269 exemplos demonstrou não ser representativa o suficiente em uma população total de mais de 329 mil *tweets* durante os experimentos e, por isso, uma nova abordagem foi utilizada para automaticamente rotular mais exemplos.

A nova abordagem consiste na utilização de um classificador não-supervisionado com um número de **n** agrupamentos, na tentativa de observar se o algoritmo é capaz de distinguir *tweets* irônicos e não-irônicos. Esta técnica onde a visualização de dados serve como apoio para uma classificação possui vasta aplicação, como nos trabalhos de Rauber *et al.* (2016) e Rauber *et al.* (2017). Porém, nossa base possui um total de 13 atributos e isto dificulta a tentativa de visualizar e interpretar os agrupamentos encontrados, então transformamos nossos dados através de PCA para reduzir suas dimensões, conforme demonstra a Figura 4.3. Sem isso, precisaríamos tentar interpretar gráficos com uma quantidade de eixos, ou dimensões, igual ao número de atributos. Outro motivador é que esta atividade também gera ganhos significativos no tempo de execução se realizada antes de atividades de agrupamento (Rauber *et al.*, 2017).

[3] https://www.clarin.si/repository/xmlui/handle/11356/1048. Acessado em 22/02/2018.

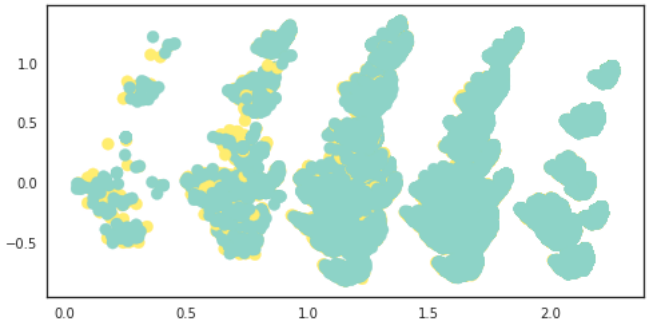

Figura 4.3: *Tweets com dimensionalidade reduzida.*

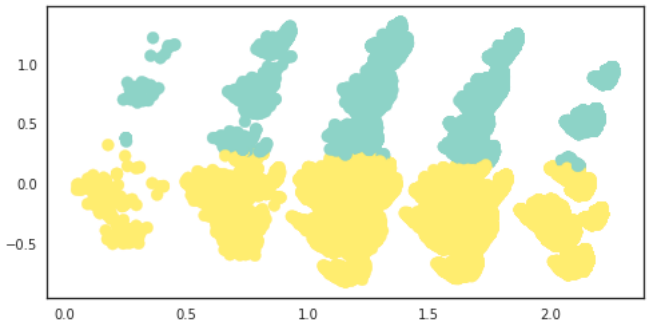

Figura 4.4: *Tweets divididos em dois agrupamentos.*

Foram realizados testes com um número **n** de agrupamentos igual a **2**, **3** e **5**. Porém, as divisões observadas nos *tweets* quando **n>2** não parecem convergir para os grupos que podem visualmente ser observados e, assim, **n=2** parece ser mais apropriado para o contexto deste trabalho, como vemos nas Figuras 4.4, 4.5 e 4.6. Os agrupamentos são constituídos por 104.075 registros com o rótulo 0 e 190.757 registros com o rótulo 1. O algoritmo utilizado da biblioteca scikit-learn, descrita na Sub-seção 2.7.2, foi o *"TruncatedSVD()"* para a transformação de PCA e o *k-means* para classificar os agrupamentos.

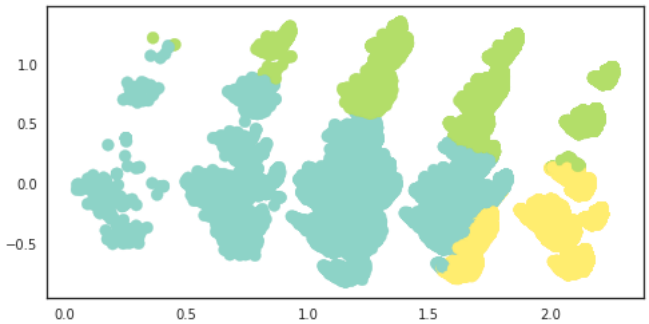

Figura 4.5: *Tweets divididos em três agrupamentos.*

Como a divisão dos agrupamentos foi realizada na *horizontal* e o trecho de dados ressaltado na Figura 4.7 não foi classificado com uma divisão tão clara, também foram realizados testes com um algoritmo de agrupamentos hierárquicos para testar outras medidas de distância (*manhattan* e *coseno*). A classificação também apresentou a mesma dificuldade e a divisão dos agrupamentos continuou na horizontal, conforme a Figura 4.8. Além disto, este algoritmo possui restrições de memória e não foi possível executá-lo em toda a base deste trabalho. Sendo assim, continuaremos com o algoritmo *k-means* e a medida *euclidiana*.

Após observar os registros classificados para entender se os *tweets* foram divididos corretamente dentro deste contexto e o rótulo no qual cada registro pertence, observou-se que, em geral, o rótulo

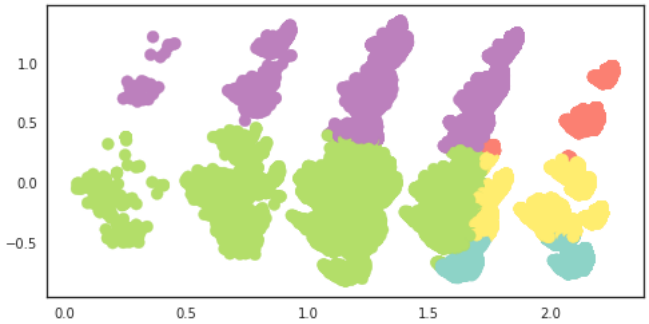

Figura 4.6: *Tweets divididos em cinco agrupamentos.*

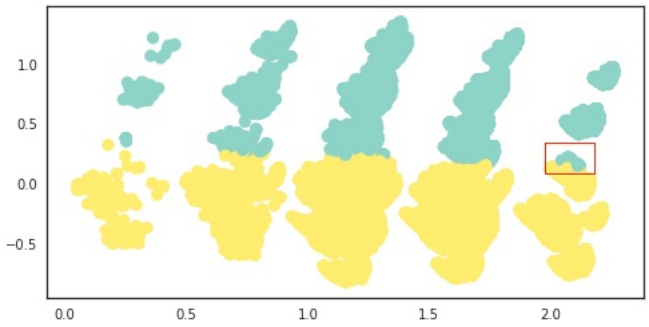

Figura 4.7: *Ressalva nos agrupamentos com k-means.*

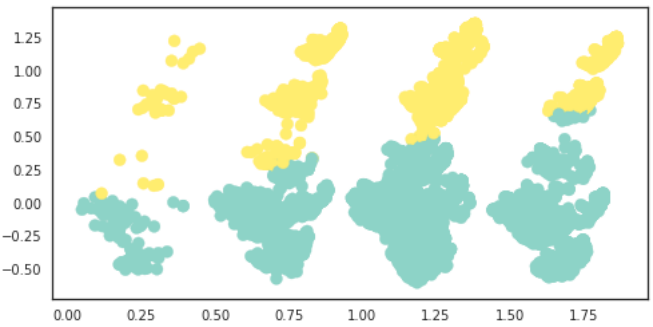

Figura 4.8: *Agrupamento hierárquico.*

0 é equivalente aos *tweets não-irônicos* e o 1 aos *tweets irônicos*. Grande parte dos *falsos positivos* encontrados se tratavam de *tweets* jornalísticos, ou seja, publicações formais de veículos de comunicação. Por este motivo, a base teve 4.004 registros dos usuários da Tabela 4.2 excluídos e o algoritmo foi executado novamente, mas é importante ressaltar que ainda assim é possível que outros usuários tenham *replicado* estes *tweets* e, portanto, ainda restem ruídos deste tipo.

A estratégia de construir uma base de treinamento a partir dos resultados de agrupamentos não faz com que se tenha uma base rotulada corretamente em sua plenitude, porém, também não seria possível garantir que uma base rotulada manualmente por seres humanos teria apenas exemplos corretos de classificação. Portanto, a hipótese desta pesquisa é que este conjunto seja suficientemente representativo para identificar as características propostas inicialmente.

Neste trabalho, a saída deste método de agrupamento serviu de base para acelerar a construção de uma base de treinamento rotulada manualmente. As bases utilizadas no experimento foram com *t, 2t e 4t tweets*, dobrando a quantidade de *tweets* rotulados manualmente até que o comportamento dos algoritmos de classificação convergisse.

usuário	descrição	tweets removidos
@Estadao	versão on-line do jornal O Estado de S.Paulo.	139
@metrosp_oficial	Twitter oficial do Metrô de São Paulo.	543
@folha	Perfil oficial do jornal Folha de S.Paulo.	172
@g1	O portal de notícias da Globo.	557
@DiariodaCPTM	Twitter do Diário da CPTM.	714
@Direto_da_CPTM	Perfil com informações do metrô.	648
@DiretodaCPTM_	Perfil com informações do metrô.	82
@UsuarioCPTM	Perfil com informações do metrô.	358
@Direto_do_Metro	Perfil com informações do metrô.	791

Tabela 4.2: *Lista de veículos de comunicação desconsiderados.*

4.4 Classificação de Ironias

4.4.1 Pré-Processamento

Após a coleta dos dados necessários para este estudo, não é possível iniciar a classificação de ironias sem estudar quais características podem revelar este comportamento dos usuários. Na maioria das vezes, somos capazes de identificar ironias em um diálogo presencial, porém a dificuldade descrita no Capítulo 1 que serviu como motivação deste projeto é justamente entender como este comportamento é refletido nas maneiras de uma pessoa expressar ironias em textos livres e curtos em uma rede social. Além disto, também é nesta etapa de Análise Exploratória de Dados que identificamos transformações necessárias nos dados brutos para utilização em algoritmos de classificação.

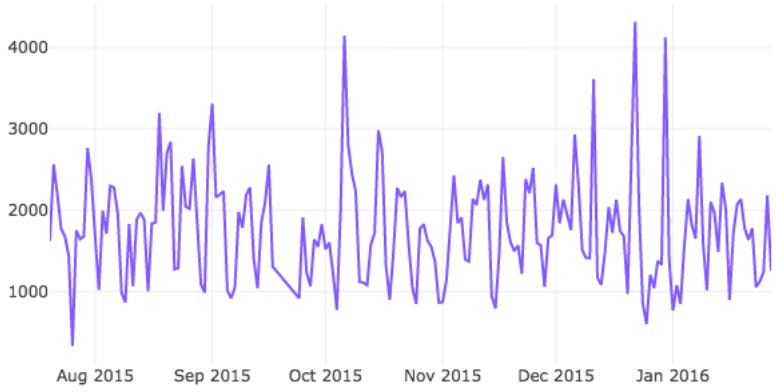

Figura 4.9: *Quantidade de tweets coletados ao longo do tempo.*

A manipulação e processamento da base coletada foi feita com as ferramentas descritas na Seção 2.7.2: Jupyter e Pandas. Após a leitura inicial dos dados, a primeira necessidade de tratamento identificada foi no atributo correspondente a **data** do *tweet*. Ao invés de um texto livre, a data foi convertida para um formato apropriado como demonstrado a seguir:

exemplo	Fri Aug 07 00:01:10 BRT 2015
formato	dia da semana, dia do mês, dia, horário, fuso horário, ano

A base tem como início o dia **20 de julho de 2015**, termina em **27 de janeiro de 2016** e

foi reagrupada por **datas** ao invés de segundos como estava originalmente, conforme observado na Figura 4.9.

Existem algumas atividades necessárias antes de treinar um classificador para transformar os dados para que os algoritmos possam interpretá-los adequadamente, como detalhamos na literatura da Seção 2.6.1. Estes foram os tratamentos realizados:

- **Textos:** O tratamento escolhido para o texto livre dos *tweets* foi a conversão para TF-IDF após remoção de *stopwords* e demais atividades de pré-processamento, conforme descrito na Sub-seção 2.6.1, *e.g.:*

tweet	"CPTM rassando hj SQN estou mais de 15 min aguardando a lata velha do trem.. CCO funcione por gentileza!
TF-IDF vetor de valores	0.0495560809912 (cptm), 0.130636057214 (rassando), 0.0820038766113 (hj), 0.0507560454781 (sqn), 0.119287722071 (15), 0.0987772526929 (min), 0.115634374158 (aguardando), 0.110125587836 (lata), 0.12399770671 (velha), 0.0683126677201 (trem), 0.130636057214 (cco), 0.130636057214 (funcione), 0.130636057214 (gentileza).

- **Valores categóricos:** Os atributos com este tipo de dado foram mapeados para valores numéricos:
  - *"negativo"*: **-1**
  - *"neutro"*: **0**
  - *"positivo"*: **1**

Uma outra atividade também foi experimentada para tratar valores numéricos com amplitudes diferentes de valores. Porém, observou-se que, na verdade, o classificador estava sendo enviesado e causava *overfitting*, uma vez que a normalização era aplicada à todos os valores e o viés era identificado nas bases de treino e testes após sua divisão.

4.4.2 Abordagem 1

O primeiro experimento de classificação de ironias foi realizado apenas com o conteúdo normalizado da língua natural dos *tweets* com base nestas transformações. Foi utilizada uma base rotulada automaticamente através de *tweets* com a *hashtag "sqn"*, onde os algoritmos tiveram como entrada uma base de treinamento balanceada com 269 exemplos de *tweets* irônicos e a mesma quantidade de registros como exemplos não-irônicos do restante da base total, mas que atendam os seguintes requisitos na tentativa de não capturar *tweets* irônicos inesperados: não possuir a *hashtag sqn*, não possuir sequencias de caracteres de exclamação e interrogação, e possuir alguma palavra.

Os métodos padrões para pré-processamento de texto fornecidos pelas bibliotecas utilizadas não tiveram o desempenho esperado e também foi necessário executar experimentos com pré-processamento customizado para o Português e etiquetas de suas classes gramaticais, conforme descrito na Seção 2.6.1.

É importante ressaltar que as bibliotecas não fornecem etiquetas para todas as palavras existentes no vocabulário de um idioma, portanto, algumas palavras não são etiquetadas. Seria necessário customizar o córpus, porém, isto não foi realizado por não fazer parte do objetivo inicial proposto. Abaixo temos um exemplo de um *tweet* etiquetado, onde o rótulo *"None"* representa uma palavra que não faz parte do córpus disponibilizado pela biblioteca e foi adicionado à palavras características do contexto deste trabalho (*e.g,* "CPTM"e "L7"):

tweet	"CPTM cuidando da minha saúde! Anda a pé na via era tudo que eu precisava pra hoje! No sol ainda pegando um bronze. Obrigada CPTM L7."
tweet etiqutado	/NPROP; CPTM/None; cuidando/V; da/NPROP; minha/PROADJ; saude!/None; Anda/V; a/ART; pe/None; na/NPROP; via/PREP; era/V; tudo/PROSUB; que/PRO-KS-REL; eu/PROPESS; precisava/V; pra/PREP; hoje!/None; No/KC; sol/N; ainda/ADV; pegando/V; um/ART; bronze/N; /NPROP; obrigada/PCP; CPTM/None; L7/None

Ao longo da exploração também foram identificadas algumas características importantes dos *tweets* que talvez indiquem um possível comportamento irônico e que, por este motivo, foram levantadas como hipóteses de estudo:

- **Contagem de caracteres de exclamação, interrogação, outros caracteres especiais e *emojis*:** quantidades podem representar uma tentativa de intensificar o sentimento expressado?
- **Contagem de letras minúsculas e maiúsculas:** se o *tweet* possui mais letras maiúsculas do que minúsculas pode indicar alguma tentativa de expressar um sentimento?

created_at	username	tweet
2016-01-13	_evelinsa	*o metrô é público , o meu corpo não .." LACROU , CONCORDO PLENAMENTE !!* 👏

- **Possui alguma palavra de fato?**: alguns tweets possuem apenas expressões de risos ou *links* para *websites*, sem nenhum texto em língua natural para ser analisado no escopo deste projeto.

created_at	username	tweet
2015-09-17	_cabeIlo97s	*!!! https://t.co/nRQvxcoki4*

- **Os caracteres estão todos em letras maiúsculas?**: também pode inidicar uma tentativa de intensificar sentimentos.

created_at	username	tweet
2015-07-31	_Emithcrf	*AMEM, METRÔ VAZIO* 😉

- **Contagem de erros gramaticais:** algumas pessoas escrevem propositalmente um *tweet* contendo erros gramaticais para expressar humor ou ironia?

created_at	username	tweet
2015-11-23	srta_marquesa	*Abriu um Ragazzo no metrô Alvim. Xonei* 😍

- **Possui alguma sequência de caracteres que expressem risos?**: sequências como *"rsrs"*, *"hahaha"* ou *"kkkkk"*, por exemplo.

created_at	username	tweet
2015-09-15	MaysaGOliveira	*CPTM mt obg, ótimo serviço* 👌 *hahahah :)*

- **Possui a sequência de caracteres "sqn"?**: uma hashtag que pode indicar ironias.

created_at	username	tweet
2015-09-16	lasleandro	*Como o metrô é aconchegante essa hora, sqn*

- **Possui alguma sequência de exclamações ou interrogações?**: estes caracteres são utilizados mais de uma vez para intensificar o sentimento.

created_at	username	tweet
2015-11-24	_pequenarafa	*Esse ar condicionado do metrô que não resolve nada!!!!!!!!!!*

tweet	*"#tbt de hj vai para o metrô de Nova Iorque... Igualzinho ao do Rio #sqn 😜 #viagemfittrips #newyork..."*	
atributos	**count_minusculas**	69
	conta_maiusculas	4
	conta_emojis	2
	conta_exclamacoes	0
	conta_interrogacoes	0
	conta_chars_especiais	7
	tem_palavra_real	1
	apenas_maiuscula	0
	tem_risada	0
	conta_erros_gramaticais	5
	tem_sqn	1
	tem_sequencia_excl_interrog	0

Tabela 4.3: *Exemplo de tweet e sua representação com atributos a serem estudados.*

4.4.3 Abordagem 2

Com base nestas hipóteses, construímos novos atributos capazes de representar os textos dos *tweets* e os utilizamos como base de nosso segundo experimento. Assim, temos a oportunidade de comparar diferentes abordagens: a classificação da língua natural e a classificação dos atributos representativos. A Tabela 4.3 mostra um *tweet* de exemplo e os respectivos atributos criados, sejam como contadores ou atributos binários. Nenhum peso foi ponderado já que são apenas hipóteses que desejamos observar para, se necessário, entendermos o peso que cada um deve ter posteriormente.

Foi necessário fazer um ajuste específico para o atributo *"tem_sqn"*. Muitos *tweets* possuem endereços eletrônicos que incluem a sequencia de caracteres *sqn*, por exemplo: *"Suspeita de bomba esvazia estação Guaianases da linha 11 da CPTM https://t.co/**sQn**E4rZNnw"*. Por este motivo, os endereços eletrônicos foram removidos do texto do *tweet*.

Também analisamos os relacionamentos entre os atributos criados para evitar duplicidades e ruídos na base utilizada. Como pode ser observado na Figura 4.10, não há nenhuma correlação significativa a ponto de ser necessário eliminar algum deles.

Após processar toda a base para extrair os novos atributos, observou-se uma grande quantidade de *emojis* através do novo atributo *"conta_emojis"*. Do total de 329.473 *tweets* da base coletada, 33.019 possuem *emojis*. A maioria dos usuários tem utilizado este componente para expressar sentimentos, seja para reforçar o que foi expresso no texto ou até mesmo para contradizê-lo, o que vai de encontro ao objetivo deste trabalho com ironias. É possível que consigamos extrair outras informações importantes a partir deste componente e, portanto, coletamos uma nova base específica de *emojis*, conforme descrito na Seção 4.2.2. Porém, nem todos os atributos desta base são necessários em nossas análise (*e.g., "posição"*) e precisamos apenas de atributos que representem o sentimento de cada *emoji*, seu respectivo percentual de probabilidade e a sequência de caracteres em *unicode* correspondente ao *emoji*.

Transformamos os contadores de ocorrências das classes de sentimento (*"negativo"*, *"neutro"* e *"positivo"*) de cada *emoji* da Tabela 4.1 em atributos que representem o sentimento e sua respectiva probabilidade para cada *emoji*. Desta forma, assumimos que a classe do sentimento de um *emoji* é aquela que possui um número de ocorrências de sentimento maior do que 40% da ocorrência total. O resultado desta transformação pode ser observado na Tabela 4.4.

Como cada tweet também pode possuir mais de um *emoji*, dois novos atributos foram criados da seguinte maneira: *"emojis"* como um vetor dos *emojis* contidos no *tweet*, e *"emojis_classe"* com a classe de sentimento dos *emojis*.

A Tabela 4.5 mostra o exemplo de um registro da base de dados com os novos atributos. Esta

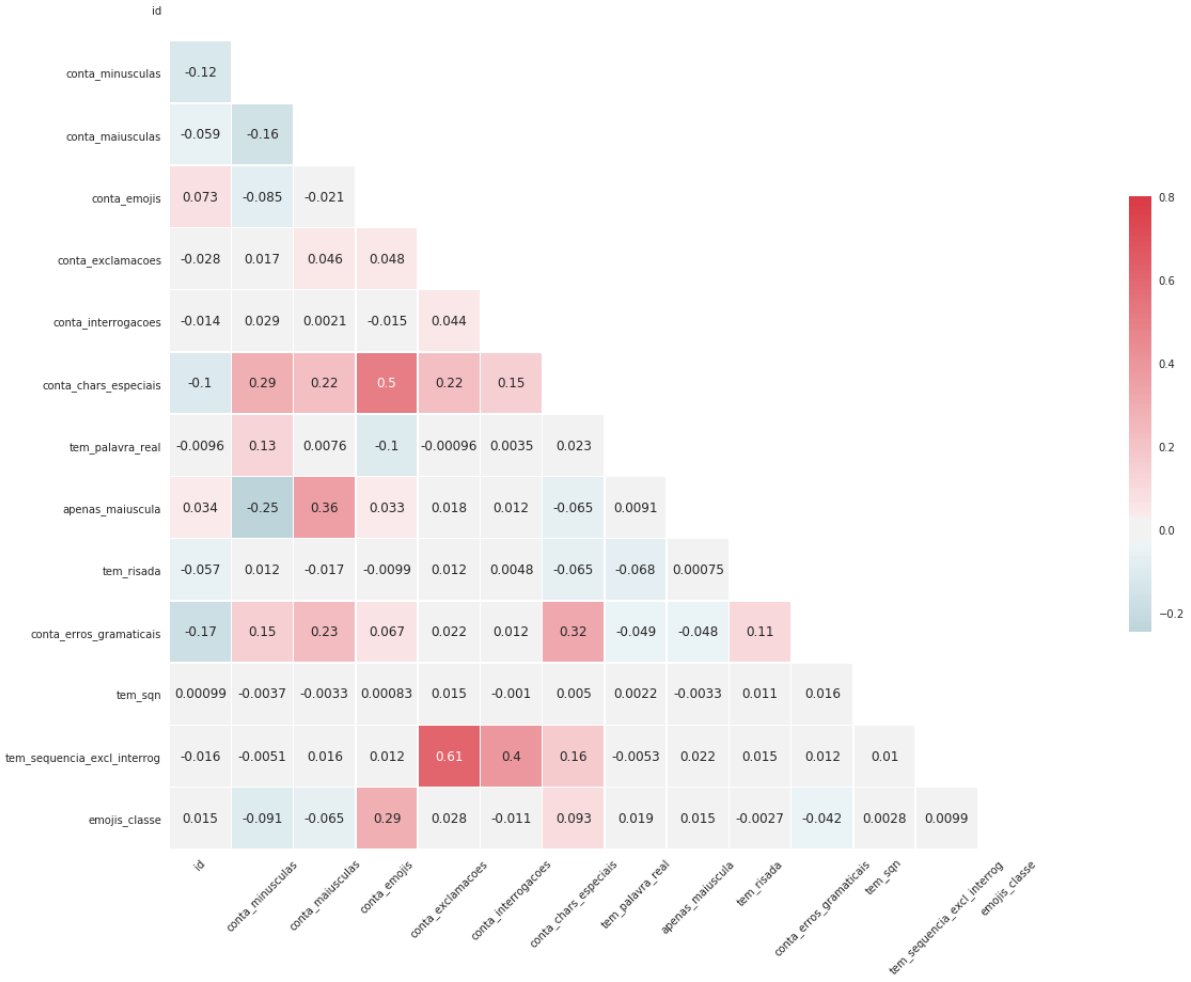

Figura 4.10: *Correlação entre os atributos criados para cada tweet.*

emoji	unicode	percentual	classse
😂	0x1f602	46.813021	positivo

Tabela 4.4: *Exemplo de registro da base de emojis.*

base serviu como entrada para os novos experimentos de classificação por atributos ao invés de língua natural, ainda que também tenham sido combinados para efeito de comparação.

4.5 Experimentos

As técnicas e abordagens utilizadas e descritas na seção anterior foram combinadas de diferentes formas e, por isso, foram realizados diversos experimentos a fim de compará-los. Os detalhes de cada um dos experimentos está no Apêndice D, mas eles englobam os seguintes cenários:

- **Classificação do texto do tweet:** texto normalizado com TF-IDF ou com as atividades de pré-processamento ou com etiquetas linguísticas.
- **Classificação dos atributos:** novos atributos criados a partir das características identificadas na Seção 4.4, sem e com os atributos específicos para o tratamento de *emojis*.
- **Bases de treinamento:** desbalanceada com todos os *tweets* coletados ou balanceada em **t,**

tweet	*"#tbt de hj vai para o metrô de Nova Iorque... Igualzinho ao do Rio #sqn 😝 #viagemfittrips #newyork..."*	
atributos	**conta_minusculas**	69
	conta_maiusculas	4
	conta_emojis	2
	conta_exclamacoes	0
	conta_interrogacoes	0
	conta_chars_especiais	7
	tem_palavra_real	1
	apenas_maiusculas	0
	tem_risada	0
	conta_erros_gramaticais	5
	tem_sqn	1
	tem_sequencia_excl_interrog	0
	emojis	[😝]
	emojis_classe	[positivo]

Tabela 4.5: *Exemplo de tweet e sua representação com atributos a serem estudados.*

2t e 4t *tweets* irônicos e não-irônicos, tendo **t** iniciado em **269** *tweets* irônicos identificados com a *hashtag #sqn* e a mesma quantidade sem a *hashtag* para *tweets* não-irônicos.

- **Teste de validação:** sem ou com validação-cruzada.

Capítulo 5

Resultados

Os resultados de cada um dos objetivos propostos por este trabalho é descrito nas próximas seções.

5.1 Características linguísticas

Na Seção 4.4 foram levantadas algumas hipóteses de estudo para identificar as características sintáticas, semânticas e pragmáticas de implicaturas conversacionais da ironia mencionadas. Dentre estas hipóteses, estas foram as características observadas como mais relevantes, baseado em suas frequências na base de ironias:

- **Possui ou não alguma palavra de fato:** estes casos foram excluídos previamente no tratamento de notícias e remoção de links para *websites*.
- **Sequência de caracteres "sqn":** *hashtag #sqn* foi utilizada para expressar ironia em todos os 269 em que esteve presente.
- **Sequência de exclamações ou interrogações:** a sequência destes caracteres foi identificada em 53 *tweets*, apenas 4.94% dos casos. Também foi notado que 176 casos (16.39%) possuem o caracter de exclamação mais de duas vezes no mesmo *tweet*.
- **Caracteres em letras maiúsculas:** ao contrário do que se imaginava, apenas 4 vezes um *tweet* foi completamente escrito em letras maiúsculas e se tratava de uma expressão de sentimento de raiva ao invés de ironia.
- **Contagem de erros gramaticais:** também ao contrário do que se imaginava, as pessoas utilizaram uma linguagem o mais formal possível nos *tweets* irônicos, contendo erros gramaticais apenas para abreviar palavras e se encaixar na limitação de caracteres nas 602 vezes (56.01%) onde erros foram encontrados.
- **Sequência de caracteres que expressem risos:** presente em 42 *tweets*, 3.92% da base de ironias, reforça a hipótese de que as pessoas utilizam linguagem mais formal para expressar ironias.
- **Contagem e sentimento de emojis:** 901 *tweets* de um total de 1074 *tweets* irônicos possuem *emojis*, ou seja, esta característica é relevante para 83.89% da base de ironias rotulada e vai de acordo com a hipótese levantada. Além disto, 851 *tweets* possuem *emojis* com sentimento negativo, ou seja, 81.36% das pessoas associaram a ironia a um sentimento negativo.

5.2 Classificação de ironias

Apesar de termos abordagens tão distintas, algumas relacionadas à língua natural expressada pelos usuários da rede social e outras com atributos que representam suas principais características, os resultados são muito próximos. Os resultados detalhados por métricas estão no Apêndice E.

Os primeiros experimentos foram realizados com uma base desbalanceada com todos os *tweets* coletados e 269 exemplos de *tweets* irônicos considerando a *hashtag #sqn*. Independente de utilizar o texto do *tweet* ou os novos atributos criados, os experimentos com esta abordagem foram invalidados, uma vez que as métricas de availação atingiram 1.0 em todos os experimentos e mostram que o classificador provavelmente não encontrou as características de ironia e seguiu apenas a condição de possuir ou não a *hashtag #sqn*, ou seja, não houve aprendizado.

O primeiro experimento com os mesmos *tweets* com a *hashtag #sqn*, porém, com a mesma quantidade de *tweets* não-irônicos do restante da base é observado nas tabelas E.1, E.2, E.3, E.4 e E.5. Temos maior concentração de métricas de avaliação na faixa de valores acima de 0.8, porém em apenas 54.44% dos testes, conforme mostra a Tabela 5.1.

Métricas de Avaliação		
Acima de 0.6	9	10.00%
Acima de 0.7	15	16.67%
Acima de 0.8	49	54.44%
Acima de 0.9	20	22.22%

Tabela 5.1: *Desempenho da classificação de ironia com base balanceada.*

Utilizar a base total coletada com poucos *tweets* rotulados manualmente não demonstrou ser a melhor abordagem para nosso estudo. A próxima seção descreve a abordagem alternativa experimentada para a construção da base de ironias.

5.3 Base de ironias

Foi construída uma base de ironias textuais em Português do Brasil. Inicialmente, utilizou-se como base os *tweets* que possuem a *hashtag #sqn*. Porém, como ela está presente em apenas 269 de um total de 329.473 *tweets*, os experimentos demonstraram não ser suficiente como base de treinamento balanceada para a classificação de ironias, conforme previamente observado na seção anterior.

Portanto, foi adotada uma estratégica de duplicar a quantidade de *tweets* da base de treinamento e rotulá-los manualmente até que o desempenho dos algoritmos de classificação fossem estabilizados. As Tabelas E.6, E.7, E.8, E.9 e E.10 mostram os resultados de cada um dos algoritmos em cada um dos cenários experimentados e descritos na Seção 4.4, utilizando bases com:

- *2t tweets* resultando em 538 *tweets* irônicos e uma base balanceada de 1076 *tweets*.
- *2t e mais 20% de tweets* resultando em 915 *tweets* irônicos em uma base balanceada de 1830 tweets.
- *4t tweets* resultando em 1074 *tweets* irônicos em uma base balanceada de 2148 *tweets*.

Analisando as métricas de *precisão, cobertura e f1* definidas na Seção 2.6.5 e seguindo esta estratégia na construção da base de treinamento, temos o desempenho na classificação de ironia exibido na Tabela 5.2. Observa-se que o desempenho se manteve distribuído acima de 0.6 nas métricas nas três abordagens com uma variação máxima de apenas oito testes. Portanto, a motivação para buscar estabilizar os resultados não foi relacionado apenas ao desempenho das métricas de avaliação, mas também aos casos de resultados inválidos.

Base balanceada com *2t* tweets		
Acima de 0.6	19	21.11%
Acima de 0.7	74	82.22%
Acima de 0.8	31	34.44%
Acima de 0.9	2	2.22%
Base balanceada com *2t* + 20% de tweets		
Acima de 0.6	24	26.67%
Acima de 0.7	75	83.33%
Acima de 0.8	14	15.56%
Acima de 0.9	12	13.33%
Base balanceada com *4t* tweets		
Acima de 0.6	22	24.44%
Acima de 0.7	62	68.89%
Acima de 0.8	27	30.00%
Acima de 0.9	6	6.67%

Tabela 5.2: *Desempenho da classificação de ironia.*

Obseva-se também que os melhores desempenhos estão sempre concentrados no intervalo de 0.7 a 0.8 e, aparentemente, o melhor desempenho em geral está na *base balanceada com 2t + 20% de tweets*. Talvez na *base balanceada com 4t tweets* foram adicionados exemplos com características mais divergentes e, desta forma, foi adicionada mais complexidade ao invés de reafirmar as características dos exemplos fornecidos previamente.

Bases balanceadas *2t* e *2t + 20% de tweets*					
>0.05	55	>0.04	67	>0.03	79
<=0.05	125	<=0.04	113	<=0.03	101
%	30.56	%	37.22	%	43.89
Bases balanceadas *2t + 20% de tweets* e *4t*					
>0.05	28	>0.04	35	>0.03	53
<=0.05	152	<=0.04	145	<=0.03	127
%	15.56	%	19.44	%	29.44

Tabela 5.3: *Diferenças nas métricas de avaliação entre as bases.*

O melhor desempenho válido foi do algoritmo de Bayes com 0.9624 na métrica de f1 quando utilizada a base balanceada com t tweets, com validação-cruzada e utilizando os novos atributos criados. O algoritmo mais estável ao longo de todos os experimentos também foi o algoritmo de Bayes com uma média de 0.8014 nas métricas de todos os testes executados, conforme mostra a Figura 5.1. É importante ressaltar que os algoritmos que mais apresentaram resultados inválidos foram AdaBoost e Árvore de Decisão em 50% dos experimentos e, portanto, foram desconsiderados na análise de estabilidade de desempenho.

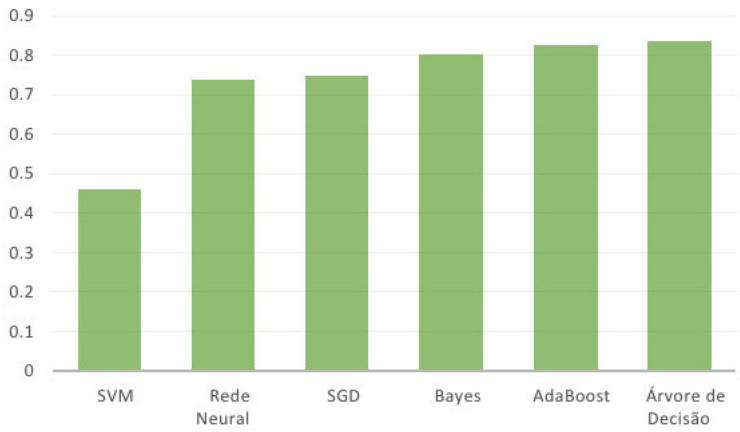

Figura 5.1: *Desempenho dos algoritmos.*

Em termos de abordagens, os cenários que utilizaram os novos atributos criados ao invés do texto do *tweet* tiveram melhores resultados, conforme demonstra a Figura 5.2, apesar de mais casos inválidos onde não houve aprendizado e as métricas foram de 1.0.

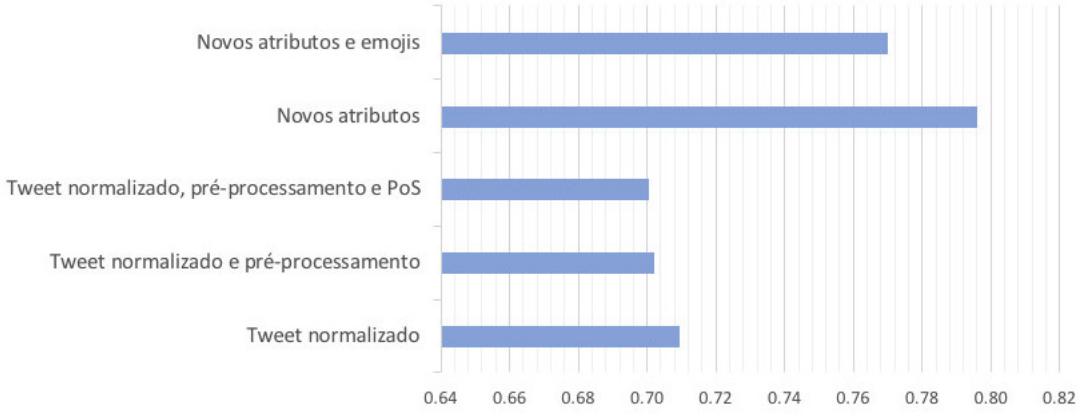

Figura 5.2: *Desempenho das abordagens.*

Além disto, temos cerca de 30.56% das métricas com uma diferença acima de 0.05 quando comparadas as bases *2t tweets* e *2t + 20% de tweets*. Comparando a última com a base *4t tweets*, temos apenas 15.56% com uma diferença maior do que 0.05. Os detalhes são demonstrados na Tabela 5.3. Assim, consideramos que os resultados dos experimentos foram estabilizados e não foram realizados novos testes.

Capítulo 6

Conclusões

Este trabalho propôs a identificação de características sintáticas, semânticas e pragmáticas de implicaturas conversacionais da ironia para a implementação de um classificador de ironias com o objetivo de construir uma base em Português do Brasil, através de abordagens com Aprendizado de Máquina.

Os trabalhos relacionados não disponibilizam bases para serem reaproveitadas e também não foram feitos para o idioma em questão, então se fez necessário coletar *tweets* no contexto proposto durante meses para construir uma base de estudos para, no final deste trabalho, disponibilizá-la em um repositório público.

Em seguida, as bases de treino e teste que são pré-requisito para o classificador foram construídas. Diferentes abordagens foram experimentadas tanto para construir as bases automaticamente quanto para acelerar a rotulagem manual.

A principal dificuldade no comportamento do classificador foi em lidar com *overfitting*, exigindo a utilização da técnica de validação-cruzada descrita na Seção 2.6.4.

Uma vez que as bases foram preparadas, um total de seis algoritmos de classificação foram experimentados das mais diferentes formas, abrangendo cenários onde utilizou-se apenas o texto do *tweet*, a fim de estudar seus componentes linguísticos, ou também cenários contendo apenas os novos atributos criados a partir deste mesmo texto.

Algumas hipóteses levantadas de início se confirmaram, no caso da *hashtag* *#sqn* e *emojis* principalmente, mas a maioria não apresentou a relevância antes imaginada. O caso de erros gramaticais e expressões de risos, *e.g.,* foram contraditas pelo linguajar mais formal utilizado quando uma pessoa deseja ser irônica. Um comportamento exatamente oposto ao que se espera de um humor da ironia. Inclusive, o comportamento observado da ironia neste contexto se aproxima mais ao sarcasmo do que ao humor.

Os resultados das métricas de avaliação estabilizaram em uma média de 122 testes acima de 0.6 com maior concentração no intervalo de 0.7 a 0.8, onde temos uma média de 70.33% de desempenho.

A continuidade desta pesquisa deve abordar a melhoria da métrica de precisão na classificação automática de ironias. Assim, os seguintes pontos devem ser observados:

- Coleta de novos *tweets*.
- Refinamento de métodos de agrupamento para classificação não-supervisionada.
- Rotulagem manual colaborativa.
- Refinamento na classificação de *emojis*, dada a relevância identificada.

Além disto, pode-se disponibilizar o classificador de ironias construído como uma API para utilização pública na comunidade através de ferramentas como o Github.

Com relação às características da ironia, em si, duas novas hipóteses também podem ser levantadas como trabalho futuro:

- As características identificadas da ironia em Português do Brasil podem ser aplicadas em outros idiomas?
- O classificador de ironias construído no contexto do transporte público de São Paulo conseguem ter desempenho semelhante em outros contextos? Toda ironia tem as mesmas características linguísticas?

E o próximo passo será relacionado à segunda hipótese: a coleta de novos *tweets* com a *hashtag* *#sqn*, sem especificar o contexto de metrôs de São Paulo, para verificar o comportamento do classificador e possivelmente construir uma base de ironias com muito mais exemplos.

Apêndice A

Apêndice Robô de Coleta

```
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream

access_token = "33389696-
    v7MmlYmt1TpQRms8WCnDY6OtIKizViB6v8i01Tur2"
access_token_secret = "
    apXaMbJHHtPK54fBfJCwwOnwFkjU3KEYECLdT69hjhWL0"
consumer_key = "CtDYfwGEQ02hZwlqfSFRSyuPT"
consumer_secret = "
    l2WE6VeZ0LBHQ0jFZpt9z8SxUkALs1I4ZUONxoRzwZ9LayCs9D"


class StdOutListener(StreamListener):

    def on_data(self, data):
        print(data)
        return True

    def on_error(self, status):
        print(status)


if __name__ == '__main__':

    l = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    stream = Stream(auth, l)

    stream.filter(track=['sqn', '#sqn'])
```

Apêndice B

Apêndice Tweet JSON Exemplo


```
{
  "created_at": "Mon Mar 27 22:49:31 +0000 2017",
  "id": 846494420363067393,
  "id_str": "846494420363067393",
  "text": "otimo dia para ser segunda-feira! #sqn",
  "source": "href=http://twitter.com/download/iphone" rel="Twitter
     for iPhone",
  "truncated": false,
  "in_reply_to_status_id": null,
  "in_reply_to_status_id_str": null,
  "in_reply_to_user_id": null,
  "in_reply_to_user_id_str": null,
  "in_reply_to_screen_name": null,
  "user": {
    "id": 33389696,
    "id_str": "33389696",
    "name": "Rayssa Kullian",
    "screen_name": "rayssak",
    "location": "Sao Paulo, Brazil",
    "url": "http://www.rayssak.com.br",
    "description": "Christian, geek, musician, Data Scientist at
       @awscloud. Big Data, AI, Machine Learning, Natural Language
       Processing, Text Mining, Groupware.",
    "protected": false,
    "verified": false,
    "followers_count": 590,
    "friends_count": 443,
    "listed_count": 39,
    "favourites_count": 202,
    "statuses_count": 14546,
    "created_at": "Mon Apr 20 02:34:41 +0000 2009",
    "utc_offset": -10800,
    "time_zone": "Brasilia",
    "geo_enabled": true,
    "lang": "en",
    "contributors_enabled": false,
    "is_translator": false,
    "profile_background_color": "4A4664",
```

```
    "profile_background_image_url": "http://pbs.twimg.com/
       profile_background_images/160658924/twitter4.jpg",
    "profile_background_image_url_https": "https://pbs.twimg.com//
       profile_background_images/160658924/twitter4.jpg",
    "profile_background_tile": true,
    "profile_link_color": "3456A1",
    "profile_sidebar_border_color": "FFFFFF",
    "profile_sidebar_fill_color": "DCD0E8",
    "profile_text_color": "474646",
    "profile_use_background_image": false,
    "profile_image_url": "http://pbs.twimg.com/profile_images/656665
       008298971136/BepCSURf_normal.jpg",
    "profile_image_url_https": "https://pbs.twimg.com/profile_images
       /656665008298971136/BepCSURf_normal.jpg",
    "profile_banner_url": "https://pbs.twimg.com/profile_banners/333
       89696/1423187143",
    "default_profile": false,
    "default_profile_image": false,
    "following": null,
    "follow_request_sent": null,
    "notifications": null
  },
  "geo": null,
  "coordinates": null,
  "place": null,
  "contributors": null,
  "is_quote_status": false,
  "retweet_count": 0,
  "favorite_count": 0,
  "entities": {
    "hashtags": [
      {
        "text": "sqn",
        "indices": [
          34,
          38
        ]
      }
    ],
    "urls": [

    ],
    "user_mentions": [

    ],
    "symbols": [

    ]
  },
  "favorited": false,
  "retweeted": false,
  "filter_level": "low",
```

```
83    "lang": "pt",
84    "timestamp_ms": "1490654971577"
85 }
```

Apêndice C

Apêndice *Emojis* Utilizados

	emoji	unicode	ocorrências	negativo	neutro	positivo
0	😂	0x1f602	14622	3614	4163	6845
1	❤️	0x2764	8050	355	1334	6361
2	♥	0x2665	7144	252	1942	4950
3	😍	0x1f60d	6359	329	1390	4640
4	😭	0x1f62d	5526	2412	1218	1896
5	😘	0x1f618	3648	193	702	2753
6	😊	0x1f60a	3186	189	754	2243
7	👌	0x1f44c	2925	274	728	1923
8	💕	0x1f495	2400	99	683	1618
9	👏	0x1f44f	2336	243	634	1459
10	😁	0x1f601	2189	278	648	1263
11	☺	0x263a	2062	128	449	1485
12	♡	0x2661	1975	102	448	1425
13	👍	0x1f44d	1854	213	460	1181
14	😩	0x1f629	1808	1069	336	403
15	🙏	0x1f64f	1539	124	648	767
16	✌	0x270c	1534	173	476	885
17	😏	0x1f60f	1522	170	676	676
18	😉	0x1f609	1521	151	513	857
19	🙌	0x1f64c	1506	152	358	996
20	🙈	0x1f648	1456	238	350	868
21	💪	0x1f4aa	1409	101	424	884
22	😄	0x1f604	1398	191	426	781
23	😒	0x1f612	1385	819	266	300
24	💁	0x1f483	1344	59	237	1048
25	💖	0x1f496	1263	54	254	955
26	😃	0x1f603	1206	86	361	759

	emoji	unicode	ocorrências	negativo	neutro	positivo
27	😔	0x1f614	1205	559	263	383
28	😱	0x1f631	1130	298	319	513
29	🎉	0x1f389	1125	43	207	875
30	😜	0x1f61c	1035	115	333	587
31	☯️	0x262f	992	5	981	6
32	🌸	0x1f338	946	37	255	654
33	💜	0x1f49c	939	41	241	657
34	💙	0x1f499	912	29	186	697
35	✨	0x2728	848	43	463	342
36	😳	0x1f633	846	277	277	292
37	💗	0x1f497	836	42	201	593
38	★	0x2605	828	25	543	260
39	▉	0x2588	798	71	682	45
40	☀️	0x2600	786	21	377	388
41	😡	0x1f621	756	403	81	272
42	😎	0x1f60e	754	79	224	451
43	😢	0x1f622	749	288	168	293
44	💋	0x1f48b	734	28	169	537
45	😊	0x1f60b	734	33	203	498
46	🐵	0x1f64a	725	97	197	431
47	😴	0x1f634	718	303	170	245
48	🎶	0x1f3b6	701	67	189	445
49	💞	0x1f49e	687	27	123	537
50	😌	0x1f60c	665	93	157	415
51	🔥	0x1f525	651	80	400	171
52	💯	0x1f4af	637	179	202	256
53	🔫	0x1f52b	604	298	126	180
54	💛	0x1f49b	602	25	123	454
55	💁	0x1f481	549	105	159	285
56	💚	0x1f49a	537	41	101	395
57	♫	0x266b	533	33	313	187
58	😞	0x1f61e	532	255	85	192
59	😆	0x1f606	527	81	148	298
60	😝	0x1f61d	496	65	155	276
61	😪	0x1f62a	482	208	105	169
62	□	0xfffd	472	78	275	119
63	😫	0x1f62b	467	227	81	159
64	😅	0x1f605	462	135	109	218

	emoji	unicode	ocorrências	negativo	neutro	positivo
65	👊	0x1f44a	458	121	111	226
66	💀	0x1f480	456	214	123	119
67	😀	0x1f600	439	37	114	288
68	😚	0x1f61a	424	20	81	323
69	😻	0x1f63b	417	28	101	288
70	©	0xa9	416	54	259	103
71	👀	0x1f440	410	93	198	119
72	💘	0x1f498	395	18	87	290
73	🐓	0x1f413	384	41	291	52
74	☕	0x2615	382	53	182	147
75	👋	0x1f44b	382	60	103	219
76	✋	0x270b	378	124	82	172
77	🎊	0x1f38a	363	20	59	284
78	🍕	0x1f355	352	19	166	167
79	❄️	0x2744	349	34	103	212
80	😢	0x1f625	341	108	83	150
81	😕	0x1f615	340	205	66	69
82	💥	0x1f4a5	329	23	234	72
83	💔	0x1f494	328	136	96	96
84	😤	0x1f624	327	157	82	88
85	😈	0x1f608	325	66	106	153
86	▶	0x25ba	325	40	192	93
87	✈️	0x2708	322	13	161	148
88	🔝	0x1f51d	303	26	106	171
89	😰	0x1f630	302	132	44	126
90	⚽	0x26bd	299	15	83	201
91	😑	0x1f611	299	154	85	60
92	👑	0x1f451	298	12	65	221
93	😹	0x1f639	295	88	77	130
94	👉	0x1f449	292	20	137	135
95	🌿	0x1f343	291	29	122	140
96	🎁	0x1f381	288	11	45	232
97	😠	0x1f620	288	163	49	76
98	🐧	0x1f427	284	11	131	142
99	☆	0x2606	282	8	144	130
100	🍀	0x1f340	278	6	186	86
101	🎈	0x1f388	277	4	68	205
102	👼	0x1f385	274	7	172	95

	emoji	unicode	ocorrências	negativo	neutro	positivo
103	😓	0x1f613	273	118	59	96
104	😣	0x1f623	271	147	35	89
105	😐	0x1f610	270	150	76	44
106	✊	0x270a	270	37	79	154
107	😨	0x1f628	269	117	73	79
108	😖	0x1f616	268	130	50	88
109	💤	0x1f4a4	267	38	91	138
110	💗	0x1f493	259	15	55	189
111	👎	0x1f44e	258	128	51	79
112	💦	0x1f4a6	252	35	62	155
113	✔	0x2714	249	31	119	99
114	😁	0x1f637	246	117	54	75
115	⚡	0x26a1	246	10	182	54
116	🙋	0x1f64b	236	30	60	146
117	🎄	0x1f384	236	15	79	142
118	💩	0x1f4a9	229	94	68	67
119	🎵	0x1f3b5	223	23	64	136
120	➡	0x27a1	222	2	185	35
121	😛	0x1f61b	220	18	50	152
122	😬	0x1f62c	214	57	58	99
123	👯	0x1f46f	211	24	69	118
124	💎	0x1f48e	209	11	68	130
125	🌿	0x1f33f	208	1	125	82
126	🎂	0x1f382	201	16	44	141
127	🌟	0x1f31f	199	11	111	77
128	🔮	0x1f52e	199	6	133	60
129	❗	0x2757	198	31	116	51
130	👫	0x1f46b	197	43	60	94
131	🏆	0x1f3c6	194	6	39	149
132	✖	0x2716	193	8	116	69
133	☝	0x261d	191	27	77	87
134	😙	0x1f619	191	3	34	154
135	☃	0x26c4	191	14	62	115
136	👿	0x1f445	190	23	55	112
137	♪	0x266a	190	15	57	118
138	🍂	0x1f342	189	6	72	111
139	💏	0x1f48f	185	33	46	106
140	🔪	0x1f52a	183	38	94	51

	emoji	unicode	ocorrências	negativo	neutro	positivo
141	🌴	0x1f334	178	13	57	108
142	👈	0x1f448	174	14	71	89
143	🌹	0x1f339	172	3	61	108
144	🙆	0x1f646	171	15	54	102
145	➜	0x279c	170	8	126	36
146	👻	0x1f47b	168	28	73	67
147	💰	0x1f4b0	168	26	73	69
148	🙈	0x1f37b	165	17	45	103
149	🙅	0x1f645	165	76	47	42
150	🌞	0x1f31e	162	3	64	95
151	🍁	0x1f341	161	5	72	84
152	⭐	0x2b50	159	5	55	99
153	▪	0x25aa	159	15	97	47
154	🎀	0x1f380	156	6	44	106
155	—	0x2501	156	14	100	42
156	☷	0x2637	154	2	140	12
157	🐷	0x1f437	152	11	73	68
158	🙉	0x1f649	150	26	47	77
159	🌺	0x1f33a	150	2	62	86
160	💅	0x1f485	149	27	36	86
161	🐶	0x1f436	148	12	37	99
162	🌚	0x1f31a	148	23	32	93
163	👽	0x1f47d	146	13	73	60
164	🎤	0x1f3a4	144	17	40	87
165	👭	0x1f46d	144	20	36	88
166	🎧	0x1f3a7	142	18	46	78
167	👆	0x1f446	138	16	60	62
168	🍸	0x1f378	138	12	38	88
169	🍷	0x1f377	137	7	68	62
170	®	0xae	137	9	80	48
171	🍉	0x1f349	136	10	33	93
172	😇	0x1f607	135	9	36	90
173	☑	0x2611	135	2	117	16
174	🏃	0x1f3c3	135	12	55	68
175	😿	0x1f63f	134	78	29	27
176	|	0x2502	134	0	87	47
177	💣	0x1f4a3	131	40	50	41
178	🍺	0x1f37a	131	8	49	74

	emoji	unicode	ocorrências	negativo	neutro	positivo
179	▶️	0x25b6	131	7	89	35
180	😗	0x1f632	129	50	38	41
181	🎸	0x1f3b8	125	1	57	67
182	🍹	0x1f379	123	5	30	88
183	💫	0x1f4ab	121	4	51	66
184	📚	0x1f4da	119	22	34	63
185	😶	0x1f636	117	50	34	33
186	🌷	0x1f337	116	0	52	64
187	💝	0x1f49d	115	8	23	84
188	💨	0x1f4a8	115	12	46	57
189	🏈	0x1f3c8	114	7	38	69
190	💍	0x1f48d	112	16	25	71
191	☔	0x2614	111	21	36	54
192	👶	0x1f478	111	6	30	75
193	🇪	0x1f1ea	109	2	36	71
194	▒	0x2591	108	25	63	20
195	🍩	0x1f369	107	15	35	57
196	👾	0x1f47e	105	1	64	40
197	☁️	0x2601	104	14	43	47
198	🌻	0x1f33b	104	1	41	62
199	😵	0x1f635	103	33	28	42

Apêndice D

Apêndice Experimentos Realizados

Esta é a descrição detalhada de cada um dos experimentos realizados neste trabalho através das técnicas e abordagens descritas no Capítulo 4, base dos resultados do Capítulo 5:

- **Experimento 1**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com a base total de todos os tweets coletados, sem validação-cruzada.
- **Experimento 2**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com a base total de todos os tweets coletados, com validação-cruzada.
- **Experimento 3**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos, sem validação-cruzada.
- **Experimento 4**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos, com validação-cruzada.
- **Experimento 5**: classificação com os novos atributos criados, com a base total de todos os tweets coletados, sem validação-cruzada.
- **Experimento 6**: classificação com os novos atributos criados, com a base total de todos os tweets coletados, com validação-cruzada.
- **Experimento 7**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos, sem validação-cruzada.
- **Experimento 8**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos, com validação-cruzada.
- **Experimento 9**: classificação com os novos atributos criados e atributos de *emojis*, com a base total de todos os tweets coletados, sem validação-cruzada.
- **Experimento 10**: classificação com os novos atributos criados e atributos de *emojis*, com a base total de todos os tweets coletados, com validação-cruzada.
- **Experimento 11**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos, sem validação-cruzada.
- **Experimento 12**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos, com validação-cruzada.
- **Experimento 13**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com a base total de todos os tweets coletados, sem validação-cruzada.

- **Experimento 14**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com a base total de todos os tweets coletados, com validação-cruzada.
- **Experimento 15**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos, sem validação-cruzada.
- **Experimento 16**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos, com validação-cruzada.
- **Experimento 17**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com a base total de todos os tweets coletados, sem validação-cruzada.
- **Experimento 18**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com a base total de todos os tweets coletados, com validação-cruzada.
- **Experimento 19**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos, sem validação-cruzada.
- **Experimento 20**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos, com validação-cruzada.
- **Experimento 21**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos dobrada, sem validação-cruzada.
- **Experimento 22**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos dobrada, com validação-cruzada.
- **Experimento 23**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos dobrada, sem validação-cruzada.
- **Experimento 24**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos dobrada, com validação-cruzada.
- **Experimento 25**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos dobrada, sem validação-cruzada.
- **Experimento 26**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos dobrada, com validação-cruzada.
- **Experimento 27**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos dobrada, sem validação-cruzada.
- **Experimento 28**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos dobrada, com validação-cruzada.
- **Experimento 29**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos dobrada, sem validação-cruzada.

- **Experimento 30**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos dobrada, com validação-cruzada.
- **Experimento 31**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, sem validação-cruzada.
- **Experimento 32**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, com validação-cruzada.
- **Experimento 33**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, sem validação-cruzada.
- **Experimento 34**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, com validação-cruzada.
- **Experimento 35**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, sem validação-cruzada.
- **Experimento 36**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, com validação-cruzada.
- **Experimento 37**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, sem validação-cruzada.
- **Experimento 38**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, com validação-cruzada.
- **Experimento 39**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, sem validação-cruzada.
- **Experimento 40**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos dobrada com mais 20% de exemplos rotulados de *tweets*, com validação-cruzada.
- **Experimento 41**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, sem validação-cruzada.
- **Experimento 42**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, com validação-cruzada.
- **Experimento 43**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, sem validação-cruzada.
- **Experimento 44**: classificação apenas com o texto do *tweet* normalizado com TF-IDF e etapas de pré-processamento, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, com validação-cruzada.

- **Experimento 45**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, sem validação-cruzada.
- **Experimento 46**: classificação apenas com o texto do *tweet* normalizado com TF-IDF, etapas de pré-processamento e etiquetas linguísticas; com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, com validação-cruzada.
- **Experimento 47**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, sem validação-cruzada.
- **Experimento 48**: classificação com os novos atributos criados, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, com validação-cruzada.
- **Experimento 49**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, sem validação-cruzada.
- **Experimento 50**: classificação com os novos atributos criados e atributos de *emojis*, com uma base balanceada de tweets irônicos e não-irônicos dobrada duas vezes, com validação-cruzada.

Apêndice E

Apêndice Resultados Detalhados por Métricas

Estes são os resultados detalhados por métricas dos experimentos de classificação de ironias.

Base desbalanceada						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
SVM	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
Árvore de Decisão	1.0000	1.0000	1.0000	0.9750	0.9818	0.9780
Rede Neural	1.0000	1.0000	1.0000	0.8000	0.5554	0.6547
SGD	1.0000	1.0000	1.0000	1.0000	0.5229	0.6854
AdaBoost	1.0000	1.0000	1.0000	0.9750	0.9818	0.9780
Base balanceada com t tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.8400	0.8300	0.8300	0.8190	0.9464	0.8780
SVM	0.2300	0.4800	0.3100	0.0000	0.0000	0.0000
Árvore de Decisão	0.9900	0.9900	0.9900	1.0000	0.9875	0.9875
Rede Neural	0.7700	0.7700	0.7700	0.7316	0.8250	0.7963
SGD	0.8200	0.8200	0.8200	0.9265	0.9124	0.9120
AdaBoost	0.9800	0.9800	0.9800	1.0000	0.9844	0.9935

Tabela E.1: *Experimento apenas com tweet normalizado.*

Base desbalanceada						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
SVM	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
Árvore de Decisão	1.0000	1.0000	1.0000	0.9900	0.8619	0.9196
Rede Neural	1.0000	1.0000	1.0000	0.8000	0.3818	0.5149
SGD	1.0000	1.0000	1.0000	0.4000	0.0277	0.0515
AdaBoost	1.0000	1.0000	1.0000	0.9809	0.8710	0.9200
Base balanceada com t tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9000	0.9000	0.8900	0.7194	0.9466	0.8171
SVM	0.2500	0.5000	0.3300	0.5181	1.0000	0.6831
Árvore de Decisão	0.9000	0.9000	0.9000	0.8728	0.8565	0.8600
Rede Neural	0.8500	0.8400	0.8400	0.6786	0.7751	0.7199
SGD	0.9100	0.9000	0.9000	0.8279	0.8648	0.8486
AdaBoost	0.8300	0.8100	0.8100	0.8759	0.8747	0.8739

Tabela E.2: *Experimento com tweet normalizado e pré-processamento.*

Base desbalanceada						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
SVM	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
Árvore de Decisão	1.0000	1.0000	1.0000	0.9805	0.9595	0.9692
Rede Neural	1.0000	1.0000	1.0000	0.6000	0.3681	0.4505
SGD	1.0000	1.0000	1.0000	0.9714	0.2516	0.3938
AdaBoost	1.0000	1.0000	1.0000	0.9810	0.9795	0.9814
Base balanceada com t tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.8800	0.8800	0.8800	0.9611	0.6526	0.7710
SVM	0.2000	0.4500	0.2800	0.0000	0.0000	0.0000
Árvore de Decisão	0.9900	0.9900	0.9900	1.0000	0.9789	0.9889
Rede Neural	0.8200	0.8200	0.8200	0.8069	0.7789	0.7841
SGD	0.9800	0.9800	0.9800	0.9533	0.8842	0.9106
AdaBoost	0.9800	0.9800	0.9800	1.0000	0.9789	0.9889

Tabela E.3: *Experimento com tweet normalizado, pré-processamento e etiquetas linguísticas.*

Base desbalanceada						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	1.0000	1.0000	1.0000	0.6087	1.0000	0.7356
SVM	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
Árvore de Decisão	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rede Neural	1.0000	1.0000	1.0000	1.0000	0.9909	0.9953
SGD	1.0000	1.0000	1.0000	0.1235	0.0364	0.0513
AdaBoost	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Base balanceada com t tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9800	0.9800	0.9800	0.9578	0.9678	0.9624
SVM	0.8300	0.8300	0.8300	0.9085	0.6993	0.7799
Árvore de Decisão	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rede Neural	0.6700	0.5200	0.4000	0.4038	0.8304	0.5432
SGD	0.8900	0.8900	0.8900	0.9251	0.6503	0.7279
AdaBoost	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Tabela E.4: *Experimento com novos atributos criados.*

Base desbalanceada						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	1.0000	1.0000	1.0000	0.5320	0.9909	0.7204
SVM	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
Árvore de Decisão	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rede Neural	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
SGD	1.0000	1.0000	1.0000	0.2000	0.0091	0.0174
AdaBoost	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Base balanceada com t tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9300	0.9200	0.9200	0.9394	0.9723	0.9553
SVM	0.8900	0.8900	0.8900	0.7521	0.8117	0.7775
Árvore de Decisão	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rede Neural	0.2000	0.4500	0.2800	0.6000	0.2294	0.1181
SGD	0.8600	0.8200	0.8200	0.6564	0.8580	0.6905
AdaBoost	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Tabela E.5: *Experimento com novos atributos criados e atributos de emojis.*

Base balanceada com *2t* tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.7800	0.7500	0.7500	0.7511	0.8093	0.7750
SVM	0.1900	0.4400	0.2700	0.9044	0.5130	0.6746
Árvore de Decisão	0.7300	0.7300	0.7300	0.7460	0.7378	0.7419
Rede Neural	0.7600	0.7600	0.7500	0.7382	0.7395	0.7366
SGD	0.8100	0.8100	0.8100	0.8021	0.7209	0.7559
AdaBoost	0.8000	0.8000	0.7900	0.7404	0.6837	0.7078
Base balanceada com *2t* + 20% de tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.8100	0.8000	0.8000	0.7447	0.8225	0.7814
SVM	0.2400	0.4900	0.3200	0.0000	0.0000	0.0000
Árvore de Decisão	0.7800	0.7800	0.7800	0.7416	0.7521	0.7458
Rede Neural	0.7800	0.7800	0.7800	0.7051	0.7521	0.7272
SGD	0.8100	0.8100	0.8100	0.7866	0.7944	0.7897
AdaBoost	0.7800	0.7800	0.7800	0.7615	0.7014	0.7300
Base balanceada com *4t* tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.8000	0.8000	0.8000	0.7242	0.8265	0.7704
SVM	0.2400	0.4900	0.3200	0.0000	0.0000	0.0000
Árvore de Decisão	0.8200	0.8200	0.8200	0.7231	0.7441	0.7331
Rede Neural	0.7800	0.7800	0.7800	0.7267	0.7326	0.7271
SGD	0.8100	0.8100	0.8100	0.7753	0.7492	0.7592
AdaBoost	0.8000	0.8000	0.8000	0.7497	0.6855	0.7150

Tabela E.6: *Novo experimento com tweet normalizado.*

Base balanceada com *2t* tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.8000	0.8000	0.7900	0.6733	0.8355	0.7440
SVM	0.2200	0.4700	0.3000	0.5093	0.8333	0.6749
Árvore de Decisão	0.7600	0.7600	0.7600	0.7437	0.7436	0.7415
Rede Neural	0.7500	0.7400	0.7400	0.6781	0.7307	0.7013
SGD	0.7700	0.7700	0.7700	0.7249	0.7261	0.7173
AdaBoost	0.7400	0.7400	0.7400	0.7269	0.7442	0.7346
Base balanceada com *2t* + 20% de tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.7900	0.7900	0.7900	0.7369	0.7355	0.7349
SVM	0.2400	0.4900	0.3200	0.0000	0.0000	0.0000
Árvore de Decisão	0.7700	0.7700	0.7700	0.7191	0.7045	0.7098
Rede Neural	0.7800	0.7800	0.7800	0.7038	0.7128	0.7067
SGD	0.7800	0.7700	0.7700	0.7216	0.7213	0.7204
AdaBoost	0.7400	0.7400	0.7400	0.7218	0.6736	0.6948
Base balanceada com *4t* tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.7900	0.7800	0.7800	0.7409	0.7942	0.7658
SVM	0.2100	0.4600	0.2900	0.0000	0.0000	0.0000
Árvore de Decisão	0.8100	0.8100	0.8100	0.7049	0.7295	0.7164
Rede Neural	0.7900	0.7900	0.7900	0.7094	0.7368	0.7218
SGD	0.7800	0.7700	0.7700	0.7362	0.7679	0.7507
AdaBoost	0.7400	0.7400	0.7400	0.6821	0.6555	0.6668

Tabela E.7: *Novo experimento com tweet normalizado e pré-processamento.*

Base balanceada com *2t* tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.7600	0.7200	0.7100	0.8019	0.8310	0.8147
SVM	0.2400	0.4900	0.3200	0.0000	0.0000	0.0000
Árvore de Decisão	0.7900	0.7900	0.7900	0.7650	0.7922	0.7756
Rede Neural	0.7400	0.7400	0.7400	0.7419	0.7829	0.7605
SGD	0.7800	0.7800	0.7800	0.8010	0.8411	0.8184
AdaBoost	0.8500	0.8500	0.8400	0.8488	0.7341	0.7837
Base balanceada com *2t* + 20% de tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.7900	0.7800	0.7800	0.7313	0.7103	0.7201
SVM	0.2300	0.4800	0.3100	0.0000	0.0000	0.0000
Árvore de Decisão	0.7900	0.7900	0.7800	0.6788	0.7045	0.6913
Rede Neural	0.8200	0.8100	0.8100	0.7029	0.7331	0.7176
SGD	0.8200	0.8200	0.8200	0.7332	0.7757	0.7531
AdaBoost	0.7700	0.7700	0.7700	0.6940	0.6705	0.6815
Base balanceada com *4t* tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.8100	0.8000	0.8000	0.7438	0.7622	0.7515
SVM	0.2400	0.4900	0.3300	0.0000	0.0000	0.0000
Árvore de Decisão	0.8100	0.8100	0.8100	0.7374	0.7306	0.7339
Rede Neural	0.8000	0.7900	0.7900	0.7096	0.7573	0.7320
SGD	0.8300	0.8300	0.8300	0.7536	0.7838	0.7677
AdaBoost	0.7500	0.7500	0.7500	0.7738	0.7283	0.7490

Tabela E.8: *Novo experimento com tweet normalizado, pré-processamento e etiquetas linguísticas.*

Base balanceada com *2t* tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9100	0.8900	0.8900	0.7697	0.5777	0.5820
SVM	0.8800	0.8800	0.8800	0.5777	0.5876	0.5820
Árvore de Decisão	1.0000	1.0000	1.0000	0.6818	0.6857	0.6834
Rede Neural	1.0000	1.0000	1.0000	0.5059	1.0000	0.6719
SGD	0.8100	0.7300	0.7200	0.5375	0.5488	0.3695
AdaBoost	1.0000	1.0000	1.0000	0.7829	0.6334	0.6978
Base balanceada com *2t* + 20% de tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9200	0.9100	0.9100	0.6784	0.4915	0.5681
SVM	0.9400	0.9400	0.9400	0.5760	0.5391	0.5559
Árvore de Decisão	1.0000	1.0000	1.0000	0.6848	0.6900	0.6861
Rede Neural	1.0000	1.0000	1.0000	0.4884	1.0000	0.6563
SGD	0.7900	0.6400	0.5900	0.4126	0.2500	0.2049
AdaBoost	1.0000	1.0000	1.0000	0.7097	0.5700	0.6309
Base balanceada com *4t* tweets						
	sem validação cruzada			com validação cruzada		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9600	0.9500	0.9500	0.6213	0.4456	0.5184
SVM	0.9400	0.9400	0.9400	0.6068	0.5592	0.5813
Árvore de Decisão	1.0000	1.0000	1.0000	0.6619	0.6588	0.6600
Rede Neural	1.0000	1.0000	1.0000	0.6053	0.5364	0.5539
SGD	0.9800	0.9800	0.9800	0.5309	0.8698	0.6340
AdaBoost	1.0000	1.0000	1.0000	0.6824	0.5338	0.5958

Tabela E.9: *Novo experimento com novos atributos criados.*

Base balanceada com *2t* tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9500	0.9500	0.9500	0.7533	0.5401	0.6267
SVM	0.8400	0.8400	0.8400	0.5445	0.5358	0.5368
Árvore de Decisão	1.0000	1.0000	1.0000	0.6813	0.6633	0.6707
Rede Neural	1.0000	1.0000	1.0000	0.8067	0.5308	0.6361
SGD	0.7500	0.5200	0.3700	0.4287	0.6667	0.4847
AdaBoost	1.0000	1.0000	1.0000	0.7817	0.6112	0.6838
Base balanceada com *2t* + 20% de tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9300	0.9200	0.9200	0.7123	0.5472	0.6170
SVM	0.9100	0.9100	0.9100	0.6175	0.6368	0.6262
Árvore de Decisão	1.0000	1.0000	1.0000	0.6616	0.6532	0.6560
Rede Neural	1.0000	1.0000	1.0000	0.8000	0.0244	0.0472
SGD	0.7500	0.4900	0.3300	0.7080	0.3084	0.3080
AdaBoost	1.0000	1.0000	1.0000	0.6800	0.6314	0.6543
Base balanceada com *4t* tweets						
	sem validação cruzada			**com validação cruzada**		
	precisão	**cobertura**	**f1**	**precisão**	**cobertura**	**f1**
Bayes	0.9800	0.9800	0.9800	0.6615	0.4683	0.5476
SVM	0.9400	0.9300	0.9300	0.6339	0.7243	0.6749
Árvore de Decisão	1.0000	1.0000	1.0000	0.6540	0.6419	0.6451
Rede Neural	1.0000	1.0000	1.0000	0.6646	0.5897	0.5996
SGD	0.9900	0.9900	0.9900	0.7126	0.2967	0.2887
AdaBoost	1.0000	1.0000	1.0000	0.6578	0.6439	0.6492

Tabela E.10: *Novo experimento com novos atributos criados e atributos de emojis.*

Referências Bibliográficas

Arlot *et al.* **(2010)** Sylvain Arlot, Alain Celisse *et al.* A survey of cross-validation procedures for model selection. *Statistics surveys*, 4:40–79. Citado na pág. 11

Bamman e Smith (2015) David Bamman e Noah A Smith. Contextualized sarcasm detection on twitter. Em *ICWSM*, páginas 574–577. Citeseer. Citado na pág. 20

Barbieri *et al.* **(2014)** Francesco Barbieri, Horacio Saggion e Francesco Ronzano. Modelling sarcasm in twitter, a novel approach. Em *Association for Computational Linguistics*, páginas 50–58. Citado na pág. 20

Bharti *et al.* **(2015)** Santosh Kumar Bharti, Korra Sathya Babu e Sanjay Kumar Jena. Parsing-based sarcasm sentiment recognition in twitter data. Em *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*, páginas 1373–1380. IEEE. Citado na pág. 20

Bouazizi e Ohtsuki (2015) Mondher Bouazizi e Tomoaki Ohtsuki. Opinion mining in twitter: How to make use of sarcasm to enhance sentiment analysis. Em *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*, páginas 1594–1597. IEEE. Citado na pág. 20

Camp (2012) Elisabeth Camp. Sarcasm, pretense, and the semantics/pragmatics distinction. *Noûs*, 46(4):587–634. Citado na pág. 6

Carvalho *et al.* **(2009)** Paula Carvalho, Luís Sarmento, Mário J Silva e Eugénio De Oliveira. Clues for detecting irony in user-generated contents: oh...!! it's so easy;-. Em *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, páginas 53–56. ACM. Citado na pág. 2, 5, 18, 19

Chomsky (1975) Noam Chomsky. Reflections on language. *New York*, 3. Citado na pág. 6

Corrêa *et al.* **(2003)** Adriana Cristina Giusti Corrêa *et al.* Recuperação de documentos baseados em informação semântica no ambiente ammo. Citado na pág. 7

Dyer (1995) Michael G Dyer. Connectionist natural language processing: a status report. Em *Computational architectures integrating neural and symbolic processes*, páginas 389–429. Springer. Citado na pág. 7

Feldman e Sanger (2007) Ronen Feldman e James Sanger. *The text mining handbook: advanced approaches in analyzing unstructured data*. Cambridge university press. Citado na pág. 9

Gantz e Reinsel (2012) John Gantz e David Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the future*, 2007(2012):1–16. Citado na pág. 1

Gensler (2002) Harry J Gensler. *Introduction to logic*. Psychology Press. Citado na pág. 5

Ghosh *et al.* **(2015)** Aniruddha Ghosh, Guofu Li, Tony Veale, Paolo Rosso, Ekaterina Shutova, John Barnden e Antonio Reyes. Semeval-2015 task 11: Sentiment analysis of figurative language in twitter. Em *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, páginas 470–478. Citado na pág. 20

Giles *et al.* **(2010)** C Lee Giles, Yang Sun e Isaac G Councill. Measuring the web crawler ethics. Em *Proceedings of the 19th international conference on World wide web*, páginas 1101–1102. ACM. Citado na pág. 22

González-Ibánez *et al.* **(2011)** Roberto González-Ibánez, Smaranda Muresan e Nina Wacholder. Identifying sarcasm in twitter: a closer look. Em *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*, páginas 581–586. Association for Computational Linguistics. Citado na pág. 20

Johnson (1987) Robert M Johnson. *A Logic Book*. Wadsworth Publishing Company Belmont, California. Citado na pág. 5, 6

Jolliffe (1986) Ian T Jolliffe. Principal component analysis and factor analysis. Em *Principal component analysis*, páginas 115–128. Springer. Citado na pág. 10

Joshi *et al.* **(2015)** Aditya Joshi, Vinita Sharma e Pushpak Bhattacharyya. Harnessing context incongruity for sarcasm detection. Em *ACL (2)*, páginas 757–762. Citado na pág. 20

Joshi *et al.* **(2016)** Aditya Joshi, Pushpak Bhattacharyya e Mark James Carman. Automatic sarcasm detection: A survey. *arXiv preprint arXiv:1602.03426*. Citado na pág. 6, 20

Joshi (1991) Aravind K Joshi. Natural language processing. *Science*, 253(5025):1242. Citado na pág. 6

Jurafsky e James (2000) Daniel Jurafsky e H James. Speech and language processing an introduction to natural language processing, computational linguistics, and speech. Citado na pág. 1, 9

Kreuz (1996) Roger J Kreuz. The use of verbal irony: Cues and constraints. *Metaphor: Implications and applications*, páginas 23–38. Citado na pág. 2

Larson (1931) Selmer C Larson. The shrinkage of the coefficient of multiple correlation. *Journal of Educational Psychology*, 22(1):45. Citado na pág. 10

Liebrecht *et al.* **(2013)** CC Liebrecht, FA Kunneman e APJ van den Bosch. The perfect solution for detecting sarcasm in tweets# not. Citado na pág. 20

Littman e Mey (1991) David C Littman e Jacob L Mey. The nature of irony: Toward a computational model of irony. *Journal of Pragmatics*, 15(2):131–151. Citado na pág. 17

Liu *et al.* **(2015)** Shusen Liu, Dan Maljovec, Bei Wang, Peer-Timo Bremer e Valerio Pascucci. Visualizing high-dimensional data: Advances in the past decade. Em *Proc. Eurographics Conf. Visualization*, páginas 20151115–127. Citado na pág. 10

Martelotta (2014) Mário Eduardo Martelotta. *Mudança linguística: uma abordagem baseada no uso*. Cortez Editora. Citado na pág. 2

Maynard e Greenwood (2014) Diana Maynard e Mark A Greenwood. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. Em *LREC*, páginas 4238–4243. Citado na pág. 20

Mitchell (1997) Tom M Mitchell. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45(37): 870–877. Citado na pág. 7

Nilsson (1965) Nils J Nilsson. *Learning machines: foundations of trainable pattern-classifying systems*. McGraw-Hill. Citado na pág. 7

Novak *et al.* **(2015)** Petra Kralj Novak, Jasmina Smailović, Borut Sluban e Igor Mozetič. Sentiment of emojis. *PloS one*, 10(12):e0144296. Citado na pág. 23

Rajadesingan *et al.* **(2015)** Ashwin Rajadesingan, Reza Zafarani e Huan Liu. Sarcasm detection on twitter: A behavioral modeling approach. Em *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, páginas 97–106. ACM. Citado na pág. 20

Rauber *et al.* **(2016)** Paulo E Rauber, Alexandre X Falcão e Alexandru C Telea. Visualizing time-dependent data using dynamic t-sne. *Proc. EuroVis Short Papers*, 2(5). Citado na pág. 23

Rauber *et al.* **(2017)** Paulo E Rauber, Samuel G Fadel, Alexandre X Falcao e Alexandru C Telea. Visualizing the hidden activity of artificial neural networks. *IEEE transactions on visualization and computer graphics*, 23(1):101–110. Citado na pág. 23

Reyes e Rosso (2013) Antonio Reyes e Paolo Rosso. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems*, páginas 1–20. Citado na pág. 19, 20

Reyes *et al.* **(2012)** Antonio Reyes, Paolo Rosso e Davide Buscaldi. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering*, 74: 1–12. Citado na pág. 20

Reyes *et al.* **(2013)** Antonio Reyes, Paolo Rosso e Tony Veale. A multidimensional approach for detecting irony in twitter. *Language resources and evaluation*, 47(1):239–268. Citado na pág. 20

Riloff *et al.* **(2013)** Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert e Ruihong Huang. Sarcasm as contrast between a positive sentiment and negative situation. Em *EMNLP*, volume 13, páginas 704–714. Citado na pág. 20

Russell *et al.* **(2003)** Stuart Jonathan Russell, Peter Norvig, John F Canny, Jitendra M Malik e Douglas D Edwards. *Artificial intelligence: a modern approach*, volume 2. Prentice hall Upper Saddle River. Citado na pág. 7, 8

Salton e Buckley (1988) Gerard Salton e Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523. Citado na pág. 10

Searle (1969) John R Searle. *Speech acts: An essay in the philosophy of language*, volume 626. Cambridge university press. Citado na pág. 6

Smola *et al.* **(2000)** Alexander Smola *et al.* Introduction to large margin classifiers. Em *Advances in large margin classifiers*. MIT Press. Citado na pág. 8

Utsumi (1996) Akira Utsumi. Implicit display theory of verbal irony: Towards a computational model of irony. *Hulstijn and Nijholt*, páginas 29–38. Citado na pág. 5

Veale e Hao (2010) Tony Veale e Yanfen Hao. Detecting ironic intent in creative comparisons. Em *ECAI*, volume 215, páginas 765–770. Citado na pág. 20

Villars *et al.* **(2011)** Richard L Villars, Carl W Olofson e Matthew Eastwood. Big data: What it is and why you should care. *White Paper, IDC*, página 14. Citado na pág. 1

Vitral (1999) Lorenzo Vitral. A negação: teoria da checagem e mudança lingüística. *DELTA: Documentação de Estudos em Lingüística Teórica e Aplicada*, 15(1). Citado na pág. 18

Wiegand *et al.* **(2010)** Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow e Andrés Montoyo. A survey on the role of negation in sentiment analysis. Em *Proceedings of the workshop on negation and speculation in natural language processing*, páginas 60–68. Association for Computational Linguistics. Citado na pág. 18

Wilson (2006) Deirdre Wilson. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116 (10):1722–1743. Citado na pág. 5

Wilson *et al.* **(2005)** Theresa Wilson, Janyce Wiebe e Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. Em *Proceedings of the conference on human language technology and empirical methods in natural language processing*, páginas 347–354. Association for Computational Linguistics. Citado na pág. 19

Witten *et al.* **(2016)** Ian H Witten, Eibe Frank, Mark A Hall e Christopher J Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann. Citado na pág. 15