



UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

Anderson Claiton Lopes

Otimização de aplicações de Processamento de Linguagem Natural para Análise de Sentimentos

São José do Rio Preto
2024

Anderson Claiton Lopes

Otimização de aplicações de Processamento de Linguagem Natural para Análise de Sentimentos

Dissertação apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista “Júlio de Mesquita Filho”, como parte dos requisitos necessários para a aprovação da disciplina Estudos Especiais.

Orientador:

Prof. Dr. Geraldo Francisco Donegá
Zafalon

**São José do Rio Preto
2024**

L864o Lopes, Anderson Claiton
Otimização de aplicações de Processamento de Linguagem Natural
para Análise de Sentimentos / Anderson Claiton Lopes. -- São José do
Rio Preto, 2024
72 p.

Dissertação (mestrado) - Universidade Estadual Paulista (UNESP),
Instituto de Biociências Letras e Ciências Exatas, São José do Rio
Preto
Orientador: Prof. Dr. Geraldo Francisco Donegá Zafalon

1. Processamento de linguagem natural. 2. Análise de Sentimentos.
3. Aprendizado de máquina. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Universidade
Estadual Paulista (UNESP), Instituto de Biociências Letras e Ciências Exatas, São José do Rio
Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Anderson Claiton Lopes

Otimização de aplicações de Processamento de Linguagem Natural para Análise de Sentimentos

Dissertação apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista “Júlio de Mesquita Filho”, como parte dos requisitos necessários para a aprovação da disciplina Estudos Especiais.

Prof. Dr. Geraldo Francisco Donegá Zafalon Anderson Claiton Lopes

Banca Avaliadora:

Prof. Dr. Angelo Cesar Colombini

Prof. Dr. Carlos Roberto Valêncio

São José do Rio Preto
05 de Março de 2024

Força, fé e coragem.

"A sabedoria é provada justa pelas suas obras."

Jesus Cristo

Agradecimentos

Agradeço a minha esposa e minhas filhas pelo apoio e compreensão durante toda essa jornada. Sem o apoio delas, nada seria.

Agradeço também aos meu pais pela educação, amor e dedicação, proporcionando forças para lidar com todas as fases da vida.

Agradeço ao meu orientador Geraldo F. D. Zafalon pelos ensinamentos, liderança, dedicação, que me proporcionou a possibilidade de realizar um sonho.

Agradeço também de forma especial ao Prof. Dr. Angelo Cesar Colombini, ao Prof. Dr. Carlos Roberto Valêncio, a Profa. Dra. Carina Alexandra Rondini e o Prof. Dr. Jorge Rady de Almeida Junior, que participaram de todo o processo de qualificação e de defesa deste trabalho.

Um agradecimento também a todos os professores e professoras do programa, que tiveram muito profissionalismo e compromisso com os alunos.

Resumo

Os recentes avanços tecnológicos proporcionaram um aumento no volume de dados disponíveis nas redes sociais. Nesse tipo de ambiente as pessoas, na sua grande maioria, procuram desenvolver diálogos, expressar opiniões e sentimentos. Esse comportamento ganha cada vez mais a atenção de empresas e organizações, influenciando na tomada de decisões das mesmas. Cada rede social possui uma característica, o que influencia no comportamento do usuário. O Twitter, por exemplo, é uma plataforma na qual os usuários expressam suas opiniões por meio de pequenas mensagens e recebem comentários de outros usuários, que podem ser por meio de texto, áudio, imagens e vídeos. Analisar essas informações em tempo real com técnicas manuais, prova-se impraticável e, com isso, ferramentas computacionais tornaram-se relevantes nesse tipo de análise. O Processamento de Linguagem Natural (PLN) é uma área da Inteligência Artificial e da Linguística, dedicada a fazer os computadores entenderem as declarações ou palavras escritas em línguas humanas. A PLN surgiu para facilitar o trabalho do usuário e para satisfazer o desejo de se comunicar com o computador em linguagem natural. Nos últimos anos, surgiram diversos estudos com diferentes heurísticas, afim de resolver o problema. Uma destas é o uso de algoritmos combinados com o uso de aprendizado de máquina. Diversos estudos destacam a importância da combinação de algoritmos e técnicas computacionais, sempre com o objetivo de chegar aos melhores resultados. Assim, o presente trabalho teve como objetivo combinar algoritmos de processamento de linguagem natural e ferramentas computacionais, com a obtenção de melhores resultados de acurácia, precisão, revocação e F1-score, quando comparado a outras propostas, no desafio de analisar sentimentos.

Palavras-chave: Processamento de linguagem natural; Análise de Sentimentos; Aprendizado de máquina.

Abstract

Recent technological advances have led to an increase in the volume of data available on social networks. In such environments, the vast majority of people seek to develop dialogues, express opinions, and feelings. This behavior is increasingly attracting the attention of companies and organizations, influencing their decision-making processes. Each social network has its own characteristics, which influence user behavior. Twitter, for example, is a platform where users express their opinions through short messages and receive comments from other users, which can be in the form of text, audio, images, and videos. Analyzing this information in real-time with manual techniques proves impractical, and thus, computational tools have become relevant in this type of analysis. Natural Language Processing (NLP) is an area of Artificial Intelligence and Linguistics, dedicated to making computers understand statements or words written in human languages. NLP was developed to facilitate user work and to fulfill the desire to communicate with the computer in natural language. In recent years, various studies with different heuristics have emerged to solve the problem. One of these is the use of algorithms combined with machine learning. Many studies highlight the importance of combining algorithms and computational techniques, always with the goal of achieving the best results. Thus, the present work aimed to combine natural language processing algorithms and computational tools, achieving better results in accuracy, precision, recall, and F1-score, when compared to other proposals, in the challenge of analyzing sentiments.

Keywords: Natural language processing, Sentiment analysis, Machine learning.

Sumário

Sumário	x
Lista de Figuras	xii
Lista de Tabelas	xiii
1 Introdução	1
1.1 Motivação	2
1.2 Objetivo	2
1.3 Justificativa	3
2 Fundamentação Teórica	5
2.1 Processamento de Linguagem Natural (PLN)	5
2.1.1 Níveis de PLN	7
2.1.2 Geração de Linguagem Natural	11
2.1.3 Etapas da PLN	12
2.1.4 Fases de um projeto de PLN	19
2.2 Aprendizado de Máquina	22
2.2.1 Algoritmos e Métodos	24
2.3 Ensemble Learning	31
2.4 Análise de Sentimentos	32
2.4.1 Aplicações	34
2.4.2 Classificação e Extração	35
2.4.3 Classificações no nível do documento	35

2.4.4	Extração de recursos	38
2.4.5	Análise de Sentimentos Multimodal (ASM)	40
2.4.6	Sumarização	41
3	Trabalhos Relacionados	45
3.1	Análise de Sentimento utilizando PLN	45
3.2	Análise de Sentimento utilizando Aprendizado de Máquina	45
4	Desenvolvimento do Trabalho	48
4.1	Estrutura proposta	48
4.2	Carregar dados	49
4.3	Pré-processamento	50
4.4	Treinamento de modelos	51
4.5	Modelo Ensemble	54
4.6	Avaliação de Modelos	55
5	Resultados	58
5.1	Plataforma de testes	58
5.2	Testes e Resultados	59
5.2.1	Resultados dos algoritmos Regressão Logística, SVM e Naive Bayes	59
5.2.2	Resultados do Modelo Ensemble	60
5.2.3	Análise do Custo Computacional	63
6	Conclusão	65
6.1	Contribuições Obtidas	65
6.2	Publicação	65
6.3	Trabalhos Futuros	66
	REFERÊNCIAS	67

Lista de Figuras

2.1	Classificação ampla da PLN.	6
2.2	Fases da arquitetura de PLN.	8
2.3	Fases de um projeto de PLN.	19
2.4	Matriz BoW.	21
2.5	Ilustração passo a passo do algoritmo K-Means.	25
2.6	Dendrograma para um agrupamento hierárquico aglomerativo.	27
2.7	Usando SVM em uma classificação problema.	30
2.8	Abordagens de classificação.	37
2.9	A arquitetura básica da estrutura de sumarização.	42
4.1	Fluxo de trabalho do projeto	49
4.2	Comandos em Python para carregar dados.	50
4.3	Comandos em Python para pré-processamento dos dados.	51
4.4	Comandos em Python para treinamento dos modelos.	54
4.5	Comandos em Python para avaliação dos modelos.	57
5.1	Comparação da Acurácia Média dos Modelos.	61

Lista de Tabelas

4.1	Tabela de Polaridade	50
5.1	Média das rodadas do algoritmo Regressão Logística	59
5.2	Média das rodadas do algoritmo SVM	59
5.3	Média das rodadas do algoritmo Naive Bayes	60
5.4	Média das rodadas do Modelo Ensemble	61
5.5	Mediana das rodadas do Modelo Ensemble	62
5.6	Desvio Padrão das rodadas do Modelo Ensemble	62
5.7	Tabela comparativa ExtraTree vs Ensemble (BoW + TF-IDF)	63
5.8	Tabela comparativa Modelo Ensemble vs Ensemble (BoW + TF-IDF)	63
5.9	Tempo médio de execução em segundos	64
5.10	Especificações do Sistema	64

Capítulo 1

Introdução

O Processamento de Linguagem Natural (PLN) é uma área da Inteligência Artificial e da Linguística, dedicada a fazer os computadores entenderem as declarações ou palavras escritas em línguas humanas. A PLN surgiu para facilitar o trabalho do usuário e para satisfazer o desejo de se comunicar com o computador em linguagem natural (KHURANA et al., 2022).

Para realizar PLN de maneira consistente é preciso estabelecer previamente critérios a serem seguidos, o que levou ao uso de algoritmos e ferramentas para esse problema. O primeiro algoritmo que teve destaque na história foi o Georgetown–IBM experiment, em 1954 (HUTCHINS, 2004). Entre 1964 e 1966, Weizenbaum (1966) desenvolveram o programa ELIZA, tido como o primeiro *chatbot*. O grande avanço em PLN veio na década de 80, graças a avanços tanto em hardware e software com o uso de *Machine Learning* (BONACCORSO, 2017).

Análise de Sentimentos, foco deste trabalho, é uma área da PLN que envolve ensinar os computadores a identificar o sentimento por trás do conteúdo escrito ou do áudio analisado. Adicionar essa capacidade de detectar automaticamente o sentimento em grandes volumes de texto e fala abre novas possibilidades para desenvolvimento de algoritmos (GHOSH; GUNNING, 2019).

Com a explosão das plataformas Web 2.0 como blogs, fóruns de discussão, redes sociais e vários outros tipos de mídia, os consumidores tiveram ao seu alcance o poder para compartilhar suas experiências e opiniões, positivo ou negativo, em relação a qualquer produto ou serviço (PANG; LEE et al., 2008).

Na literatura atual é possível encontrar uma diversidade de técnicas utilizadas para Análise de Sentimentos, tais como Aprendizado de Máquina e Abordagem baseada em Léxico (MEDHAT;

HASSAN; KORASHY, 2014).

Apesar dessas técnicas serem amplamente utilizadas, como Aprendizagem Supervisionada e Não Supervisionada, que utilizam algoritmos como Naive Bayes (RISH et al., 2001), Rede Bayesiana (KITSON et al., 2021), Entropia Máxima (WU, 2012) e classificadores lineares com uso de Redes Neurais (ABDI; VALENTIN; EDELMAN, 1999) e Máquina de Vetor de Suporte (MAMMONE; TURCHI; CRISTIANINI, 2009), essas abordagens sofrem com o problema da correta classificação de polaridade. Isso significa que erros cometidos nas etapas iniciais ao longo da execução, influenciam a qualidade final do resultado.

1.1 Motivação

A obtenção de melhores resultados na classificação de polaridade é de suma importância, considerando a gama de atividades que se servem dos resultados da Análise de Sentimentos. Polaridade diz respeito a classificação de um texto se é uma citação positiva, negativa ou neutra. Essas atividades realizadas por empresas e organizações trazem importantes retorno para a sociedade, seja em forma de monitoramento de reputação das empresas, qualidade de produtos e serviços, monitoramento de segurança e fraudes (PANG; LEE et al., 2008).

1.2 Objetivo

Assim, o objeto do presente trabalho é obter melhores resultados no Processamento de Linguagem Natural com Análise de Sentimentos em redes sociais, através da combinação de diferentes técnicas, realizada nos seguintes passos:

1. Aplicação de pré-processamento linguístico para redução de ruídos, impulsionando a ação do processamento de PLN afim de facilitar o processo de classificação.
2. Utilização de técnicas de *Machine Learning* (Aprendizado de Máquina), incluindo os algoritmos Regressão Logística, SVM e Naive Bayes, para classificação dos dados e análise de sentimento.

3. Implementação do Ensemble, por meio das Ensemble (BoW + TF-IDF), que se mostrou fundamental para alcançar os melhores resultados na análise de sentimento.

1.3 Justificativa

É possível encontrar na literatura diversos trabalhos bem-sucedidos que justificam a hipótese do presente trabalho.

Dentro do escopo do Processamento de Linguagem Natural com Análise de Sentimentos utilizando Aprendizado de Máquina, trabalhos como o de Chong, Selvaretnam e Soon (2014), Reddy et al. (2020) e Rahman, AlOtaibi e AlShehri (2019) aplicam essas técnicas para resolver os desafios de classificação de subjetividade, associação semântica e de polaridade. Apesar dos bons resultados obtidos, os autores dos trabalhos observam uma deteriorização dos resultados e/ou o baixa qualidade na precisão da análise de sentimento causada pela baixa qualidade dos dados, como por exemplo o uso de gírias, textos não estruturados, emogis, entre outros. Os autores sugerem que isso pode ser melhorado com técnicas no pré-processamento dos dados combinado com uso de diferentes algoritmos.

O Aprendizado de Máquina, também conhecido como *Machine Learning* é observada em outros campos do Processamento de Linguagem Natural, como Glaz et al. (2021); Rainey, Brown e Kirk (2017); Ziems e Wu (2021), reforçando a eficácia da mesma, apesar dos desafios encontrados, como falta de comparações quantitativas por conta de estudos heterogêneos, diferenças entre culturas e idiomas. O comportamento dos usuários e a evolução tecnológica é uma etapa de vital importância tanto para Análise de Sentimentos quanto para outros métodos do Processamento de Linguagem Natural, pois é a partir desse conjunto de respostas que o algoritmo trabalha e busca resultados. A ferramenta *pSenti*, por exemplo, é uma ferramenta de Análise de Sentimentos em nível de conceito que combina léxico e métodos de classificação de sentimentos. Em comparação com métodos baseados em léxico, a ferramenta alcançou maior precisão em detecção de força de sentimento e classificação de polaridade, porém quando a ferramenta foi comparada com a máquina pura, que são métodos baseados em aprendizagem, rendeu uma precisão menor. É observada melhoria na solução obtida quando métodos são combinados como baseado em léxico e baseado em aprendizagem. Assim, pode-se afirmar que um pré-processamento dos dados de boa qualidade pode alavancar os

resultados dos algoritmos (AHMAD et al., 2017a).

A influência dos emogis é discutida por Mohta et al. (2020) e Novak et al. (2015), que demonstram o impacto desse tipo de comunicação presente. Um pré-processamento nessa fase, propicia melhores resultados e um aproveitamento dos dados analisados. Desse modo, a utilização de técnicas de aprendizado de máquina combinado com técnicas de processamento de linguagem natural auxilia na decisão sobre a polaridade de frases, textos, áudios sob a perspectiva de diversos contextos, o que resulta em melhor aproveitamento dos dados analisados.

Por fim, trabalhos como o de Samal, Behera e Panda (2017) e Shi e Li (2011), demonstram e justificam a hipótese de que a Aprendizagem de Máquina através do Aprendizado Supervisionado para Análise de Sentimentos pode solucionar desafios como extração de polaridade.

Capítulo 2

Fundamentação Teórica

Neste capítulo serão apresentados os conceitos fundamentais para o desenvolvimento deste trabalho, que são o Processamento de Linguagem Natural (PLN), Análise de Sentimentos, Aprendizado Supervisionado, Algoritmos e ferramentas.

2.1 Processamento de Linguagem Natural (PLN)

O Processamento de Linguagem Natural (PLN) é um ramo da Inteligência Artificial e da Linguística, dedicado a fazer com que os computadores entendam as declarações ou palavras escritas em línguas humanas. O Processamento de Linguagem Natural surgiu para facilitar o trabalho do usuário e satisfazer o desejo de se comunicar com o computador em linguagem natural (KHURANA et al., 2022).

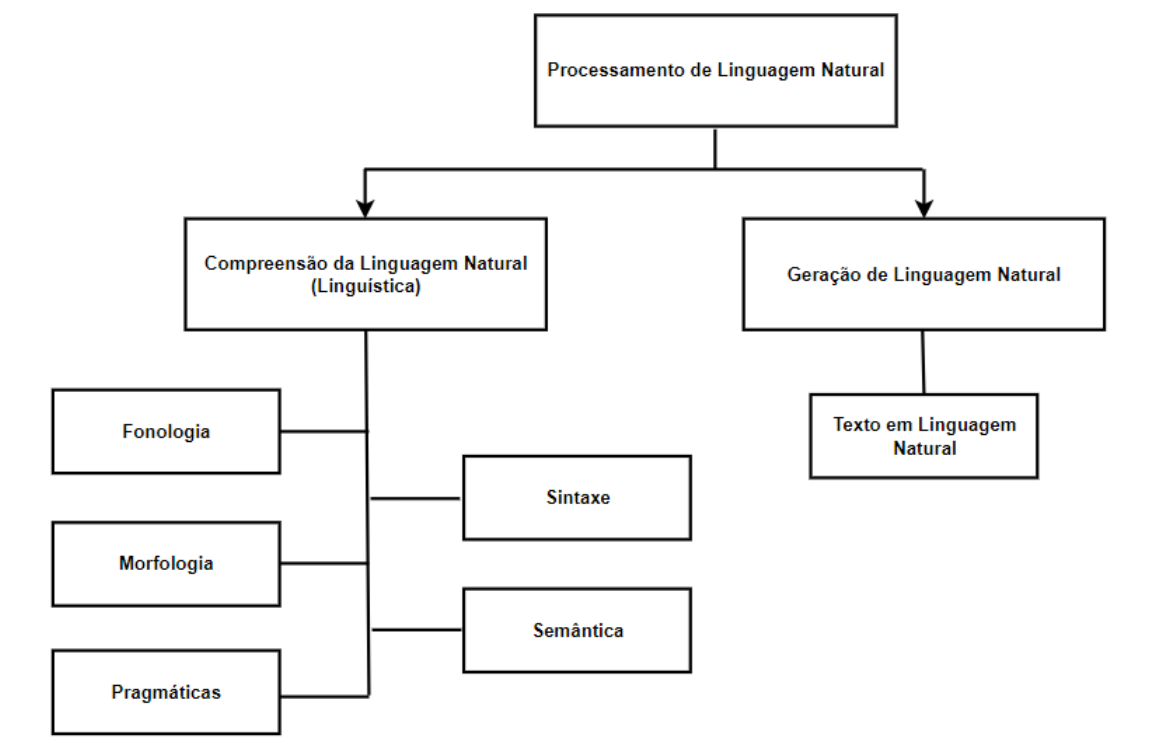
Segundo Ghosh e Gunning (2019), a PLN emergiu de campos como Inteligência Artificial, Linguística, Linguagem formal e Compiladores. Hoje, os cálculos em linguagem natural estão sendo feitos usando técnicas de Aprendizado de Máquina (Machine Learning) e Aprendizado Profundo (Deep Learning).

Uma linguagem pode ser definida como um conjunto de regras ou um conjunto de símbolos. Os símbolos são combinados e usados para transmitir informações. Os símbolos são tiranizados pelas regras (KHURANA et al., 2022).

O Processamento de Linguagem Natural basicamente pode ser classificado em duas partes, ou seja, Compreensão da Linguagem Natural e Geração de Linguagem Natural, que envolve a tarefa de

entender e gerar o texto, conforme demonstrado na Figura 2.1.

Figura 2.1: Classificação ampla da PLN.



Fonte: Elaborado pelo autor.

Ainda segundo Khurana et al. (2022), a Linguística é a ciência da linguagem que inclui Fonologia que se refere ao som, morfologia, formação de palavras, sintaxe, estrutura da frase, semântica, sintaxe e pragmática que se refere ao entendimento.

Noah Chomsky, um dos primeiros linguistas do século XII que iniciou as teorias sintáticas, marcou uma posição única no campo da lingüística teórica porque revolucionou a área da sintaxe (CHOMSKY, 1965). Através de seus estudos a PLN pode ser amplamente categorizada em dois níveis, sendo o nível superior, que inclui o reconhecimento de fala e nível inferior, que corresponde à linguagem natural.

Algumas das tarefas de PLN têm aplicações diretas no mundo real, como tradução automática, reconhecimento de entidade nomeada, reconhecimento óptico de caracteres, entre outras. A sumarização automática por exemplo, produz um resumo compreensível de um conjunto de texto e fornece resumos ou informações detalhadas de texto de um tipo conhecido. A resolução de co-referência refere-se a uma frase ou conjunto maior de texto que determina qual palavra se refere ao mesmo objeto. A análise do discurso refere-se à tarefa de identificar a estrutura do discurso do

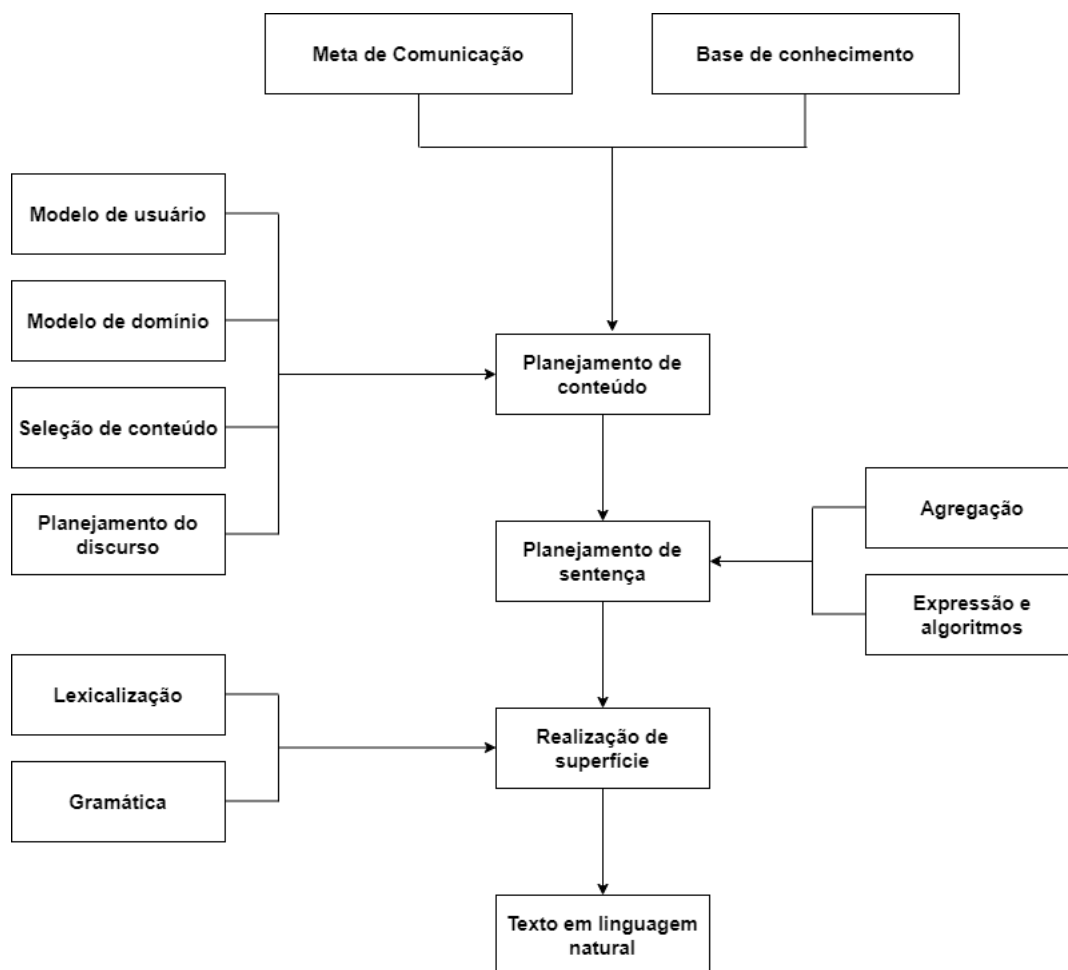
texto conectado. A tradução automática, refere-se à tradução automática de texto de um idioma humano para outro. Segmentação morfológica refere-se a separar palavras em morfemas individuais e identificar a classe dos morfemas. Reconhecimento de entidade nomeada (REN) descreve um fluxo de texto, determina quais itens no texto estão relacionados a nomes próprios. Reconhecimento óptico de caracteres (OCR) fornece uma imagem que representa o texto impresso, que ajuda a determinar o texto correspondente ou relacionado. Marcação de parte do discurso descreve uma frase, determina a parte do discurso para cada palavra. Embora as tarefas da PLN estejam obviamente muito entrelaçadas, elas são usadas com frequência, por conveniência. Algumas das tarefas, como resumo automático, análise de co-referência, atuam como subtarefas usadas na resolução de tarefas maiores (KHURANA et al., 2022).

A maior parte do trabalho em Processamento de Linguagem Natural é conduzida por cientistas da computação, enquanto vários outros profissionais também demonstraram interesse, como linguistas, psicólogos, filósofos etc. Um dos aspectos mais interessantes da PLN é que ela se soma ao conhecimento da linguagem humana. A área de Processamento de Linguagem Natural está relacionada com diferentes teorias e técnicas que tratam do problema da linguagem natural de comunicação com os computadores. A ambigüidade é um dos maiores problemas da linguagem natural que geralmente é enfrentado no nível sintático, que tem como subtarefas o léxico e a morfologia, que se preocupam com o estudo das palavras e a formação de palavras (KHURANA et al., 2022).

2.1.1 Níveis de PLN

Os níveis de linguagem são um dos métodos mais explicativos para representar o Processamento de Linguagem Natural, o que ajuda a gerar o texto PLN realizando as fases de planejamento de conteúdo, planejamento de frases e realização de superfície, demonstrado na Figura 2.2.

Figura 2.2: Fases da arquitetura de PLN.



Fonte: Elaborado pelo autor.

Linguística é a ciência que envolve o significado da linguagem, o contexto da linguagem e as várias formas da linguagem. As várias terminologias importantes do Processamento de Linguagem Natural são:

Fonologia

Fonologia é a parte da Linguística que se refere ao arranjo sistemático dos sons. O termo fonologia vem do grego antigo, o termo "Fono" significa voz ou som, e o sufixo "logia" refere-se a palavra ou fala (KHURANA et al., 2022).

A Fonologia é o estudo do som pertencente ao sistema da linguagem. Ela se refere amplamente aos sons da linguagem, preocupada com a subdisciplina da lingüística, e pode ser explicada como, a área que se preocupa com a função, comportamento e organização dos sons. A fonologia inclui o uso semântico do som para codificar o significado de qualquer linguagem humana (NATION;

SNOWLING; CLARKE, 2007).

Morfologia

As diferentes partes da palavra representam as menores unidades de significado conhecidas como morfemas. A morfologia que compreende a natureza das palavras, é iniciada pelos morfemas. Um exemplo de morfema poderia ser a palavra pré-cancelamento, que pode ser morfologicamente examinada em três morfemas separados: o prefixo pré, a raiz cancela e o sufixo -mento. A interpretação do morfema permanece a mesma em todas as palavras, apenas para entender o significado, os humanos podem quebrar qualquer palavra desconhecida em morfemas. Por exemplo, adicionar o sufixo -ed a um verbo no idioma inglês, indica que a ação do verbo ocorreu no passado. As palavras que não podem ser divididas e têm significado por si mesmas são chamadas de morfema lexical, exemplo mesa ou cadeira. As palavras terminadas em -ed, -ing, -est, -ly, -ful no idioma inglês, combinadas com o morfema lexical, são conhecidas como morfemas gramaticais, por exemplo, worked, consulting, smallest, likely e use. Esses morfemas gramaticais que ocorrem em combinação são chamados de morfemas vinculados. Os morfemas gramaticais podem ser divididos em morfemas vinculados e morfemas derivacionais (KHURANA et al., 2022).

Léxico

No léxico, os humanos, assim como os sistemas de PLN, interpretam o significado de palavras individuais. Diversos tipos de processamento conferem compreensão em nível de palavra. O primeiro deles sendo uma marcação de parte da fala para cada palavra. Nesse processamento, as palavras que podem atuar como mais de uma parte do discurso são atribuídas à etiqueta de parte do discurso mais provável com base no contexto em que ocorrem. No nível léxico, as representações semânticas podem ser substituídas pelas palavras que possuem um significado. No sistema PLN, a natureza da representação varia de acordo com a teoria semântica empregada (KHURANA et al., 2022).

Sintático

Este nível enfatiza o exame minucioso das palavras em uma frase, a fim de descobrir a estrutura gramatical da frase. Tanto a gramática quanto o analisador são necessários neste nível. A saída

deste nível de processamento é a representação da sentença que divulga as relações de dependência estrutural entre as palavras. Existem várias gramáticas que podem ser impedidas e que, por sua vez, atrapalham a opção de um analisador sintático. Nem todos os aplicativos de PLN exigem uma análise completa de sentenças, portanto, os desafios de permanência na análise de anexação de frase preposicional e auditoria de conjunção não impedem mais aquele fundamento para o qual dependências frasais e clausais são adequadas (LIDDY, 2001).

A sintaxe transmite significado na maioria dos idiomas porque a ordem e a dependência contribuem para a conotação. Por exemplo, as duas sentenças: 'O gato perseguiu o rato' e 'O rato perseguiu o gato' diferem apenas em termos de sintaxe, mas transmitem significados bastante diferentes (KHURANA et al., 2022).

Semântica

Na semântica, a maioria das pessoas pensa que o significado é determinado, no entanto, não é isso, são todos os níveis que conferem ao significado. O processamento semântico determina os possíveis significados de uma frase, girando em torno das interações entre os significados em nível de palavra na frase. Esse nível de processamento pode incorporar a desambiguação semântica de palavras com múltiplos sentidos; de uma maneira cognata de como a desambiguação sintática de palavras que podem ser classificadas como múltiplas partes do discurso é hábil no nível sintático. Por exemplo, entre outros significados, 'arquivo' como substantivo pode significar uma pasta para reunir papéis, ou uma ferramenta para formar as unhas, ou uma fila de indivíduos em uma fila (LIDDY, 2001).

O nível semântico perscruta as palavras para sua elucidação dicionarizada, mas também para a elucidação que derivam do meio da frase. Meio semântico que a maioria das palavras tem mais de uma elucidação, mas que podemos identificar a apropriada olhando para o restante da frase (FELDMAN, 1999).

Discurso

Enquanto a sintaxe e a semântica trabalham com unidades de comprimento de sentença, o nível de discurso da PLN, trabalha com unidades de texto maiores que uma sentença, ou seja, não

interpreta textos com várias sentenças como apenas sentenças de sequência, cada uma das quais pode ser elucidada individualmente. Em vez disso, o discurso se concentra nas propriedades do texto como um todo que transmitem significado fazendo conexões entre as frases componentes (LIDDY, 2001).

Os dois níveis mais comuns são resolução anáfora, que é a substituição de palavras como pronomes, que são semanticamente encaixados, pela entidade pertinente à qual se referem. Reconhecimento da estrutura do discurso/texto influencia as funções das frases no texto, o que, por sua vez, contribui para a representação significativa do texto (KHURANA et al., 2022).

Pragmática

A pragmática preocupa-se com o uso firme da linguagem em situações e utiliza o núcleo acima do núcleo do texto para entender o objetivo e explicar como o significado extra é lido em textos sem ser literalmente codificado neles. Isso requer muito conhecimento do mundo, incluindo a compreensão de intenções, planos e metas. Por exemplo, as duas frases a seguir precisam de aspiração do termo anafórico 'eles', mas essa aspiração requer conhecimento pragmático ou de mundo (LIDDY, 2001).

2.1.2 Geração de Linguagem Natural

Geração de Linguagem Natural (GLN) é o processo de produção de frases, sentenças e parágrafos que são significativos a partir de uma representação interna. Faz parte do Processamento de Linguagem Natural e ocorre em quatro fases: identificar os objetivos, planejar como os objetivos podem ser alcançados, avaliando a situação e as fontes comunicativas disponíveis e realizando os planos como um texto (KHURANA et al., 2022).

Locutor e Gerador

Para gerar um texto precisamos ter um locutor ou aplicativo e um gerador ou programa que traduza as intenções do aplicativo em frases fluentes relevantes para a situação (KHURANA et al., 2022).

Componentes e Níveis de Representação

O processo de geração de linguagem envolve as seguintes tarefas entrelaçadas. Seleção de conteúdo: as informações devem ser selecionadas e incluídas no conjunto. Dependendo de como essas informações são analisadas em unidades representacionais, partes das unidades podem ter que ser removidas enquanto outras podem ser adicionadas por padrão. Organização Textual: A informação deve ser organizada textualmente de acordo com a gramática, deve ser ordenada tanto sequencialmente quanto em termos de relações linguísticas como modificações.

- Recursos linguísticos: Para apoiar a realização da informação, os recursos linguísticos devem ser escolhido. No final, esses recursos serão reduzidos a escolhas de palavras específicas, expressões idiomáticas, construções sintáticas, etc.
- Realização: Os recursos selecionados e organizados devem ser realizados como um texto real ou saída de voz (KHURANA et al., 2022).

Aplicativo ou Palestrante

Isso é apenas para manter o modelo da situação. Aqui o palestrante apenas inicia o processo, não participa da geração da linguagem. Armazena a história, estrutura o conteúdo potencialmente relevante e desdobra uma representação do que realmente conhece. Tudo isso forma a situação, enquanto seleciona o subconjunto de proposições que o palestrante possui. O único requisito é que o orador tenha que entender a situação (KHURANA et al., 2022).

2.1.3 Etapas da PLN

O principal trabalho em Processamento de Linguagem Natural, baseado em Aprendizado de Máquina começou durante a década de 1980. Foi nessa época que houve o desenvolvimentos em várias disciplinas, levando ao surgimento de um projeto interdisciplinar chamado PLN.

De acordo com Joshi (2019), existem várias tarefas padrão de PLN. Além dessas tarefas, podemos projetar nossas próprias tarefas de acordo com nossas necessidades. Essas tarefas são conhecidas como tarefas de pré-processamento.

Tokenização

A tokenização é o procedimento de dividir uma frase em suas palavras constituintes. Por exemplo, considere esta frase: "Estou lendo um livro". Após passar esta frase para um programa de tokenização, as palavras/tokens extraídas seriam "Eu", "estou", "lendo", "um", "livro", e ".". Este exemplo extrai um token por vez e esses tokens são chamados de unigramas (GHOSH; GUNNING, 2019).

No entanto, também podemos extrair dois ou três tokens por vez. Extrair tokens é importante porque, por conveniência, tendemos a analisar a linguagem natural palavra por palavra. Se extrairmos dois tokens de cada vez, isso é chamado de bigramas. Com base nos requisitos, n-gramas podem ser extraídos (onde "n", é um número natural) (GHOSH; GUNNING, 2019).

PoS Tagging

PoS tagging significa "part-of-speech tagging", que em português pode ser traduzido como "marcação de partes do discurso". É uma técnica usada em Processamento de Linguagem Natural (PLN) para identificar e marcar a classe gramatical (parte do discurso) de cada palavra em um determinado texto, e, finalmente, rotulá-las. Parte do discurso de tokens que constituem uma frase são extraídos, para que possa filtrar os PoS que interessam e analisá-los (GHOSH; GUNNING, 2019).

Por exemplo, se olharmos para a frase "O céu é azul", obteremos quatro tokens - "O", "céu", "é", "azul". Agora, usando o PoS Tagging, marcamos partes do discurso para cada palavra/token. Isso terá a seguinte aparência:

[("O", "DT"), ("céu", "NN"), ("é", "VBZ"), ("azul", "JJ)],

onde:

DT = determinante;

NN = substantivo;

VBZ = verbo;

JJ = Adjetivo.

Palavras de paradas

Palavras de paradas é um processo de pré-processamento de texto na área de Processamento de Linguagem Natural (PLN), que consiste em remover palavras consideradas irrelevantes para a análise do texto. Essas palavras são conhecidas como "stop words"(palavras de parada, em português), que são palavras muito frequentes em um idioma, mas que não carregam um significado importante para a análise do texto, como preposições, artigos, conjunções e pronomes (GHOSH; GUNNING, 2019).

Segundo Daniel, H et al. (2007), a remoção de stop words é uma técnica comum na PLN para reduzir a dimensionalidade dos dados, melhorar a eficiência do processamento e, muitas vezes, melhorar a qualidade da análise. Isso porque, ao remover as stop words, a quantidade de palavras do texto é reduzida, facilitando a identificação de padrões relevantes e reduzindo a interferência de palavras irrelevantes nos resultados.

As stop words mais comuns incluem "o", "a", "e", "de", "para", entre outras, dependendo do idioma em questão.

Normalização de texto

Normalização de texto em Processamento de Linguagem Natural (PLN) refere-se ao processo de transformar o texto de entrada em um formato padrão ou uniforme, de modo a eliminar inconsistências e variações que possam afetar negativamente a qualidade do processamento e análise do texto (SCHÜTZE; MANNING; RAGHAVAN, 2008).

A normalização pode incluir várias etapas, tais como:

- Conversão para minúsculas ou maiúsculas;
- Remoção de pontuação;
- Remoção de caracteres especiais;
- Correção ortográfica;
- Redução de palavras a suas raízes (stemming).

Nessa linha, a normalização de texto é um processo em que diferentes variações de texto são convertidas em um formato padrão (GHOSH; GUNNING, 2019).

Correção ortográfica

A correção ortográfica em Processamento de Linguagem Natural (PLN) refere-se ao processo de identificar e corrigir erros ortográficos em um texto. Isso pode incluir a correção de erros de digitação, erros de ortografia comuns e erros de pontuação (KUKICH, 1992),

Segundo Ghosh e Gunning (2019), a correção ortográfica em Processamento de Linguagem Natural (PLN) refere-se ao processo automatizado de identificar e corrigir erros ortográficos em textos naturais escritos em um determinado idioma.

A correção ortográfica é importante para melhorar a qualidade e a precisão da análise de texto em PLN. Ela ajuda a garantir que as palavras sejam reconhecidas corretamente e, portanto, que a análise do texto seja mais precisa e confiável.

As técnicas de correção ortográfica em PLN geralmente envolvem o uso de dicionários, onde as palavras são verificadas para determinar se elas estão escritas corretamente ou não. Algumas abordagens mais avançadas também utilizam técnicas de aprendizado de máquina, como modelos de linguagem, para prever a palavra correta com base no contexto em que ela aparece no texto.

A correção ortográfica é uma das várias etapas de pré-processamento do texto em PLN, e geralmente é realizada em conjunto com outras técnicas, como tokenização, normalização e remoção de stop words, para preparar o texto para análises mais avançadas, como análise de sentimento e classificação de texto.

Stemming

Stemming é uma técnica de Processamento de Linguagem Natural (PLN) que envolve a redução de palavras flexionadas para sua forma base ou raiz, conhecida como stem. Por exemplo, o stem da palavra "corrida" é "corr", o stem da palavra "correria" é "corr" e o stem da palavra "correr" é "corr" (GHOSH; GUNNING, 2019).

O stemming é usado em PLN para reduzir a dimensionalidade dos dados, permitindo que algoritmos de PLN lidem com palavras semelhantes, mas com variações em sua forma, como tempo verbal, gênero, número, entre outros. Isso é útil para tarefas como classificação de texto, recuperação de informação e Análise de Sentimentos, onde a variação de formas de palavras pode dificultar a identificação de padrões e relações.

Existem vários algoritmos de stemming disponíveis em PLN, incluindo o Porter Stemmer, Snowball Stemmer e Lancaster Stemmer Bounabi, Moutaouakil e Satori (2019). Cada algoritmo tem suas próprias regras e heurísticas para reduzir palavras a seus stems, e a escolha de qual algoritmo usar depende do domínio do problema e do conjunto de dados em questão.

Embora o stemming possa melhorar a eficácia dos algoritmos de PLN em alguns casos, ele também pode levar a resultados imprecisos em outros. Isso ocorre porque o stemming não leva em consideração o contexto das palavras e pode reduzir palavras distintas a um mesmo stem. Por exemplo, a palavra "amigo" e "amiga" terão o mesmo stem "amig", o que pode não ser desejável em algumas situações de análise de texto (GHOSH; GUNNING, 2019).

Lematização

A lematização em Processamento de Linguagem Natural (PLN) é o processo de reduzir as palavras em sua forma base ou raiz, conhecida como "lema". O objetivo da lematização é simplificar o texto para facilitar a análise, reduzindo a inflexão das palavras para sua forma mais básica. Isso é útil porque muitas palavras na língua portuguesa têm várias formas, como variações de tempo verbal ou flexão de gênero e número, mas essas variações não são relevantes em muitas tarefas de análise de texto (ALAM et al., 2021).

Segundo Ghosh e Gunning (2019), a lematização é um processo em que as palavras são reduzidas às suas formas básicas, chamadas de "lemas". Essa técnica é usada em processamento de linguagem natural para transformar diferentes formas de uma mesma palavra em uma única forma canônica, o que permite uma análise mais precisa e eficiente.

Por exemplo, as palavras "corri", "correndo" e "correria" podem ser lematizadas para o lema "correr". Isso permite que o texto seja analisado com mais eficiência, eliminando a necessidade de tratar cada variação como uma palavra separada.

A lematização é frequentemente utilizada em tarefas como a análise de sentimento, classificação de texto e extração de informações em que as palavras são importantes, mas a flexão não é relevante. A lematização também é útil para reduzir a dimensionalidade dos dados em modelos de linguagem, reduzindo o número de recursos a serem considerados.

Reconhecimento de Entidades Nomeadas (REN)

Reconhecimento de Entidades Nomeadas (REN) em Processamento de Linguagem Natural (PLN) é uma tarefa que envolve a identificação e extração de entidades nomeadas em um texto, como nomes de pessoas, organizações, locais, datas, valores monetários, entre outros (NADEAU; SEKINE, 2007).

A principal finalidade do REN é a extração de informações relevantes de textos em larga escala, possibilitando a construção de bases de dados mais completas e facilitando o acesso a informações específicas. O reconhecimento de entidades nomeadas também é utilizado em sistemas de perguntas e respostas, sistemas de tradução automática e sistemas de resumo automático de texto.

Segundo (GHOSH; GUNNING, 2019), entidades nomeadas geralmente não estão presentes em dicionários. Então, é preciso tratá-los separadamente. O principal objetivo desse processo é identificar as entidades nomeadas (como nomes próprios) e mapeá-las para as categorias já definidas.

O processo de reconhecimento de entidades nomeadas envolve a segmentação do texto em sentenças e palavras, seguida da identificação das entidades nomeadas presentes no texto. Para isso, são utilizadas técnicas de aprendizado de máquina e análise estatística, combinadas com dicionários de entidades nomeadas e regras gramaticais. O reconhecimento de entidades nomeadas é uma tarefa complexa em PLN, pois as entidades nomeadas podem apresentar variações em sua escrita, gênero, número, grafia e ordem de palavras, entre outras características. No entanto, as técnicas de reconhecimento de entidades nomeadas têm se aprimorado com o avanço das tecnologias de PLN, possibilitando a identificação de entidades com maior precisão e eficiência.

Desambiguação do sentido da palavra

A desambiguação do sentido da palavra em Processamento de Linguagem Natural (PLN) é uma tarefa que consiste em identificar o significado correto de uma palavra que pode ter múltiplos sentidos em um determinado contexto. Esta tarefa é importante em PLN, pois a ambiguidade das palavras pode levar a erros de interpretação e compreensão de textos (NAVIGLI, 2009).

A desambiguação do sentido da palavra é uma tarefa complexa, uma vez que o significado de uma palavra pode depender do contexto em que é utilizado. Por exemplo, a palavra "banco" pode ter diferentes sentidos em frases como "vou ao banco" (instituição financeira) e "me sento no

banco"(assento).

Segundo (PAL; SAHA, 2015), para realizar a desambiguação do sentido da palavra, são utilizadas técnicas de aprendizado de máquina e análise estatística, combinadas com informações lexicais e sintáticas. Um dos métodos mais comuns é o uso de algoritmos de classificação supervisionada, que treinam modelos a partir de um conjunto de dados rotulados para identificar o sentido correto de uma palavra em um determinado contexto.

A desambiguação do sentido da palavra é uma tarefa importante em várias aplicações de PLN, como sistemas de tradução automática, sistemas de resumo automático de texto e sistemas de perguntas e respostas. Além disso, a desambiguação do sentido da palavra também pode ser utilizada em aplicações mais específicas, como Análise de Sentimentos em redes sociais e reconhecimento de voz.

Deteccção de limite de sentença

A detecção de limite de sentença em Processamento de Linguagem Natural (PLN) é uma tarefa que consiste em identificar os limites de uma sentença em um texto. Isso é importante porque a maioria das tarefas de PLN requer o processamento de frases individuais, e a detecção de limites de sentença é a primeira etapa para dividir um texto em frases separadas (KISS; STRUNK, 2006).

A detecção de limite de sentença é uma tarefa desafiadora em PLN, pois as sentenças podem ser estruturadas de maneiras complexas e podem incluir abreviações, siglas, números, nomes próprios e outros elementos que podem confundir os algoritmos de detecção de limite de sentença. Por exemplo, uma sentença pode terminar em uma abreviação, como "Dr." ou "Sr.", ou pode incluir uma lista de itens numerados, onde cada item termina com um número seguido de um ponto.

Segundo (GHOSH; GUNNING, 2019), detecção de limite de sentença é o método de detectar onde uma sentença termina e onde outra sentença começa. Apesar de um ponto final (.) indicar o final de qualquer frase e o início de outra frase, pode haver casos em que as abreviações são separadas por pontos finais. Várias análises precisam ser realizadas em um nível de sentença, portanto, detectar os limites das sentenças é essencial.

Existem várias abordagens para a detecção de limite de sentença em PLN. Uma abordagem comum é usar modelos estatísticos de aprendizado de máquina para identificar os padrões de

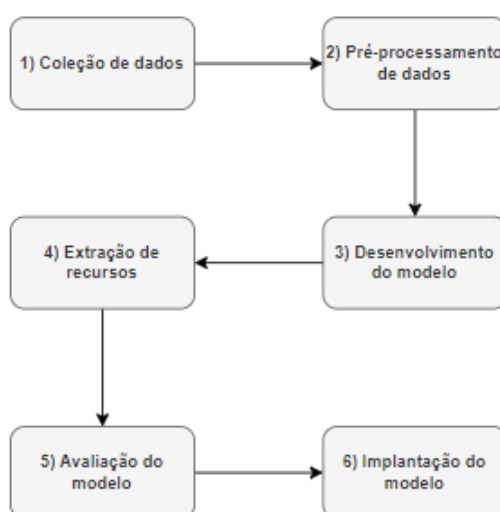
pontuação que geralmente indicam o fim de uma sentença, como pontos, pontos de exclamação e pontos de interrogação. Outras abordagens incluem o uso de regras gramaticais para identificar os limites de sentença com base na estrutura sintática das frases.

A detecção de limite de sentença é uma tarefa fundamental em muitas aplicações de PLN, como Análise de Sentimentos, análise de tópicos, resumo automático de texto, tradução automática, entre outras. A detecção precisa de limites de sentença pode melhorar significativamente o desempenho dessas aplicações, garantindo que cada sentença seja processada corretamente e isoladamente.

2.1.4 Fases de um projeto de PLN

Normalmente um projeto de PLN é dividido em vários sub-projetos ou fases. Isso aumenta a eficiência geral do processo, pois cada fase geralmente é realizada por recursos especializados. Um projeto de PLN deve passar por seis fases principais, que estão descritas na Figura 2.3.

Figura 2.3: Fases de um projeto de PLN.



Fonte: Elaborado pelo autor.

Coleção de dados

Coleção de dados em Processamento de Linguagem Natural (PLN) refere-se a um conjunto de amostras de linguagem natural que foram coletadas e organizadas para serem usadas como entrada

para modelos de PLN. Essas amostras podem incluir textos escritos, transcrições de fala ou qualquer outra forma de dados que contenha linguagem natural (GHOSH; GUNNING, 2019).

A coleção de dados é um componente fundamental do processo de treinamento de modelos de PLN. Os modelos de PLN são alimentados com uma grande quantidade de dados para aprender padrões e regras da linguagem natural. Quanto maior e mais diversificada for a coleção de dados, mais preciso e eficaz será o modelo.

Além disso, a coleção de dados pode ser usada para outras tarefas, como avaliação de modelos de PLN e criação de conjuntos de teste e validação. Ao usar uma coleção de dados padronizada para avaliação, é possível comparar diferentes modelos e algoritmos de PLN de forma justa e objetiva.

Pré-processamento de dados

O pré-processamento de dados é uma etapa fundamental do processamento de linguagem natural (PLN) que envolve a limpeza e transformação dos dados brutos em um formato adequado para análise por algoritmos de PLN. O pré-processamento é importante porque a linguagem natural é complexa e variada, o que significa que os dados brutos podem conter erros, redundâncias, informações irrelevantes e outras irregularidades que podem afetar negativamente o desempenho dos algoritmos de PLN (GHOSH; GUNNING, 2019).

Algumas das técnicas de pré-processamento de dados mais comuns em PLN incluem: limpeza de texto, tokenização, remoção de palavras de paradas, normalização de texto e part-of-speech tagging.

Desenvolvimento do modelo

Assim que o conjunto de recursos estiver pronto, é preciso desenvolver um modelo adequado que possa ser treinado para obter conhecimento dos dados. Esses modelos são treinados usando os dados coletados e pré-processados durante as etapas iniciais do projeto e são geralmente estatísticos, baseados em aprendizado de máquina (GHOSH; GUNNING, 2019).

Neste trabalho, iremos utilizar como modelo os algoritmos Regressão Logística, SVM, Naive Bayes e Método Ensemble, dado as características e complexidade da base de dados escolhida.

Extração de recursos

Os computadores entendem apenas dígitos binários 0 e 1. Assim, cada instrução que é alimentada em um computador é transformada em dígitos binários. Da mesma forma, os modelos de aprendizado de máquina tendem a entender apenas dados numéricos. Como tal, torna-se necessário converter os dados de texto em sua forma numérica equivalente. Para representar as conversas limpas, podemos usar a matriz BoW Juluru et al. (2021). Essa é a forma mais simples de representação de texto em números. Assim como o próprio termo, podemos representar uma frase como um vetor de saco de palavras (uma sequência de números). Como exemplo, vamos usar três tipos de resenhas de filmes.

Comentário 1: Este filme é muito apavorante e longo.

Comentário 2: Este filme não é assustador e é lento.

Comentário 3: Este filme é assustador e bom.

Primeiro é necessário construir um vocabulário de todas as palavras únicas nas três revisões acima. O vocabulário consiste em 11 palavras: 'Este', 'filme', 'é', 'muito', 'apavorante', 'e', 'longo', 'não', 'lento', 'assustador', 'bom'.

Representado na matriz BoW, seria:

Figura 2.4: Matriz BoW.

	1 Este	2 filme	3 é	4 muito	5 apavorante	6 e	7 longo	8 não	9 lento	10 assustador	11 bom	Duração da revisão
Comentário 1	1	1	1	1	1	1	1	0	0	0	0	7
Comentário 2	1	1	2	0	0	1	0	1	1	1	0	8
Comentário 3	1	1	1	0	0	1	0	0	0	1	1	6

Fonte: Elaborado pelo autor.

Vetor de Revisão 1: [1 1 1 1 1 1 0 0 0 0]

Vetor de Revisão 2: [1 1 2 0 0 1 0 1 1 1 0]

Vetor de Revisão 3: [1 1 1 0 0 1 0 0 0 1 1]

O número 1 (Um) representa a presença da palavra na frase e o número 0 (Zero) a não presença, respectivamente. Esse tipo de notação é conhecida como código binário (HEATH, 1972).

Avaliação do modelo

Segundo Ghosh e Gunning (2019), para um modelo é essencial fazer um benchmarking, sendo esse processo conhecido como avaliação de modelo. Nesta etapa, é avaliado o desempenho do modelo comparando-o com outros. Isso pode ser feito usando diferentes indicadores de desempenho, como acurácia, precisão, revocação e F1-Score.

Com base nos resultados da validação, é possível ajustar os hiperparâmetros do modelo, como taxa de aprendizado, número de camadas, tamanho do vocabulário, entre outros. Esse processo de ajuste é iterativo e busca melhorar o desempenho do modelo.

Após o ajuste, finalmente o modelo é avaliado usando o conjunto de teste, que contém dados não vistos anteriormente. Essa avaliação final permite obter uma medida objetiva do desempenho do modelo em um ambiente mais próximo do mundo real. Essa etapa é crucial para determinar se o modelo é suficientemente preciso e confiável para a aplicação pretendida.

Implantação do modelo

Uma vez que o modelo tenha sido treinado e avaliado, ele pode ser implantado em um ambiente de produção para realizar a tarefa de PLN desejada. Isso pode envolver a integração do modelo em um sistema existente, o desenvolvimento de uma interface de usuário ou a criação de uma API para acesso ao modelo (GHOSH; GUNNING, 2019).

2.2 Aprendizado de Máquina

Algoritmos de aprendizado de máquina são técnicas que buscam aprender padrões de comportamento ou características de um objeto, de modo a tomarem decisões sobre o mesmo sem terem sido explicitamente programadas para tal (ZHANG, 2020). Esse aprendizado pode ser classificado em três tipos: Supervisionado, não supervisionado e por reforço.

Aprendizado supervisionado

Esse tipo de aprendizado pode ser comparado à atuação de um professor ou mentor, uma vez que o algoritmo recebe dados já rotulados de entrada para que possa identificar a regra ou padrão

que define seu rótulo (ZHANG, 2020), assim sendo capaz de rotular um novo dado. Essa etapa é chamada de treino, e após ela, é realizada uma etapa de teste para verificar o desempenho do método. Tendo uma taxa de acertos dentro do esperado, ele passa a lidar somente com dados não rotulados, tomando as decisões necessárias por si mesmo (BONACCORSO, 2017).

Aprendizado não supervisionado

Neste caso, não há um guia inicial para o funcionamento do algoritmo, de forma que temos somente a etapa de teste. O método irá rotular os dados de entrada e ao final receberá sua taxa de acerto e de erro, ajustando seus parâmetros e testando novamente, de modo a atingir uma taxa de acerto suficiente para considerar o aprendizado um sucesso. Essa abordagem é útil quando não se sabe quais aspectos devem ser observados nos dados para classificá-los, de modo que o algoritmo encontrará a melhor forma (BONACCORSO, 2017; ZHANG, 2020).

O aprendizado também pode ser realizado utilizando um conjunto de dados onde alguns possuem rótulos e os demais não. Essa abordagem recebe o nome de **aprendizado semi supervisionado** (BONACCORSO, 2017).

Aprendizado Profundo (Deep Learning)

Aprendizado Profundo é uma subárea de Aprendizado de Máquina que se baseia em redes neurais artificiais compostas por múltiplas camadas, permitindo a aprendizagem de representações de dados de forma hierárquica (GOODFELLOW; BENGIO; COURVILLE, 2016). Uma das principais vantagens do Aprendizado Profundo é sua capacidade de aprender automaticamente características relevantes dos dados, eliminando a necessidade de engenharia manual de atributos (LECUN; BENGIO; HINTON, 2015). Isso o torna especialmente eficaz em tarefas complexas de processamento de dados, como reconhecimento de imagens, processamento de linguagem natural e análise de séries temporais.

No entanto, o Aprendizado Profundo também enfrenta alguns desafios significativos, como a necessidade de grandes volumes de dados para treinamento e o alto custo computacional associado ao treinamento de modelos complexos (LECUN; BENGIO; HINTON, 2015). Além disso, a interpretação dos modelos de Aprendizado Profundo pode ser difícil devido à sua complexidade e falta de

transparência.

Autores relevantes na área de Aprendizado Profundo incluem Geoffrey Hinton, Yann LeCun e Yoshua Bengio, que foram pioneiros no desenvolvimento de arquiteturas de redes neurais profundas e fizeram contribuições significativas para o avanço do campo (GOODFELLOW; BENGIO; COURVILLE, 2016).

Além disso, uma ferramenta amplamente utilizada no desenvolvimento e implementação de modelos de Aprendizado Profundo é o TensorFlow, um framework de código aberto desenvolvido pelo Google (ABADI, 2016). O TensorFlow oferece suporte para diversas tarefas de Aprendizado de Máquina, incluindo classificação, regressão, clustering e, principalmente, treinamento de redes neurais profundas. Sua flexibilidade e eficiência têm sido fundamentais para impulsionar os avanços em áreas como reconhecimento de padrões em imagens, processamento de linguagem natural e até mesmo em projetos de robótica e veículos autônomos.

Aprendizado por reforço

O aprendizado por reforço não conta com um guia, como no supervisionado, porém possui um sistema de *feedback* qualitativo. Isso significa que o algoritmo não recebe um valor numérico como retorno, mas sim uma espécie de premiação ou de penalidade. O algoritmo, então, não busca melhorar quantitativamente sua taxa de acerto, mas sim obter o maior número possível de premiações consecutivas (BONACCORSO, 2017).

Essa abordagem é muito utilizada em ambientes não-determinísticos, ambientes cujas informações mudam a todo instante ou onde não é possível aferir uma taxa de erro (BONACCORSO, 2017).

2.2.1 Algoritmos e Métodos

Os algoritmos de aprendizado de máquina são divididos em algoritmos de aprendizado não supervisionado e algoritmos de aprendizado supervisionado.

No campo de aprendizado de máquina não supervisionado, existem vários algoritmos que são amplamente utilizados em diferentes problemas e domínios. Alguns dos algoritmos mais populares são: K-Means, PCA (Principal Component Analysis), Apriori, Autoencoders, t-SNE (t-Distributed Stochastic Neighbor Embedding) e Agrupamento Hierárquico (AHMAD et al., 2017b).

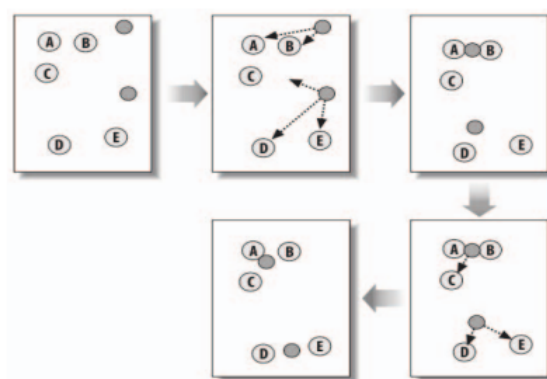
Neste trabalho iremos abordar o Algoritmo K-means e Agrupamento Hierárquico.

Algoritmo K-Means

O algoritmo K-Means é um dos métodos de particionamento mais conhecidos que são usados em pesquisas científicas e aplicações industriais. O algoritmo K-Means descobre k clusters, encontrando k centróides (pontos centrais), também chamados representante do cluster no conjunto de dados. Em seguida, atribui cada dado apontando para o cluster com o centróide mais próximo; cada ponto de dados pode ser atribuído a apenas um dos k clusters (um centróide de cluster é a média ou mediana dos pontos em seu cluster e proximidade é definido por uma função de distância ou similaridade). K-médias algoritmo é aplicável apenas quando o número de clusters ou meio é definido. Isso significa que requer o valor de k em antecedência, o que é muito difícil (AHMAD et al., 2017b).

Na Figura 2.5, apresentamos uma explicação visual do passo a passo do algoritmo K-Means.

Figura 2.5: Ilustração passo a passo do algoritmo K-Means.



Fonte: Elaborado pelo autor.

Para um dado k, o algoritmo K-Means consiste em quatro etapas:

- 1) Selecione k pontos arbitrários como os centróides iniciais;
- 2) Atribua todos os pontos ao centróide mais próximo;
- 3) Recalcule o centroide de cada cluster;
- 4) Repita os passos 2 e 3 até que os centróides não mudem.

O algoritmo K-Means é eficiente computacionalmente e é amplamente utilizado em diferentes aplicações, como segmentação de clientes, agrupamento de documentos, análise de imagens, entre outros. No entanto, é importante considerar algumas limitações do algoritmo como a necessidade

de definir o número de clusters K a priori e a sensibilidade a pontos de dados atípicos (outliers) (BONACCORSO, 2017).

Segundo Nazari et al. (2015), o algoritmo K-Means é um procedimento de busca local e sofre de uma séria desvantagem no seu desempenho e ele depende muito da partida inicial.

Agrupamento Hierárquico

O agrupamento hierárquico é um método de agrupamento bem conhecido que pode ser pensado como um conjunto de métodos de agrupamento, organizados em uma estrutura de árvore. Esses métodos constroem os clusters, particionando recursivamente os dados de forma *top-down* ou *bottom-up*, aplicável a diferentes domínios e regiões. Segundo Murtagh e Contreras (2012), algoritmos com abordagens de baixo para cima tendem a ser mais precisos, mas têm um custo computacional maior do que as abordagens *top-down*. No entanto, essa maior complexidade computacional não coincide com o aumento da complexidade conceitual ou algorítmica.

Já um agrupamento hierárquico divisivo, tem uma visão de cima para baixo, abordagem de que todos os objetos pertencem inicialmente a uma única raiz cluster e particiona iterativamente os clusters existentes em subclusters (NAZARI et al., 2015).

Algoritmos hierárquicos são de grande interesse para uma série de domínios de aplicação e são pensados para produzir melhor qualidade aglomerados. Flexibilidade incorporada no que diz respeito ao nível de granularidade, facilidade de manuseio de quaisquer formas de similaridade e distância e aplicabilidade a qualquer tipo de atributo são alguns dos vantagens dos métodos de agrupamento hierárquico (KUMAR; SINGH, 2019).

O agrupamento hierárquico em PLN pode ser usado para várias tarefas, como agrupamento de documentos com base em seu conteúdo, segmentação de clientes com base em seus padrões de comportamento, organização de tópicos em coleções de texto e assim por diante. Ele permite a identificação de estruturas e relações complexas entre os dados de texto, fornecendo uma visão hierárquica dos agrupamentos.

Uma vantagem do agrupamento hierárquico é que não é necessário especificar o número de clusters antecipadamente, pois a estrutura hierárquica permite explorar diferentes níveis de granularidade. No entanto, o agrupamento hierárquico pode ser computacionalmente mais exigente em comparação

com outros métodos de agrupamento e a interpretação da estrutura hierárquica resultante pode ser desafiadora (KUMAR; SINGH, 2019).

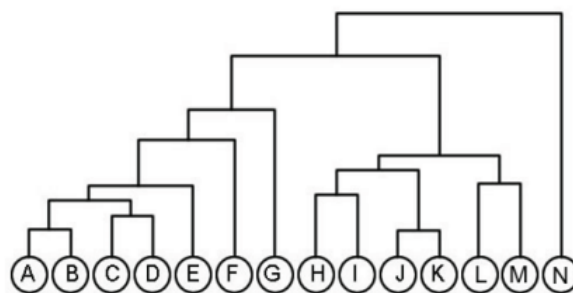
Enquanto isso existem duas desvantagens para esses métodos:

- A maioria dos algoritmos hierárquicos não revisitam clusters uma vez construídos com o objetivo de melhoria;
- Imprecisão da rescisão critério.

Enquanto a maioria dos algoritmos de agrupamento hierárquico envolvem juntar dois subclusters ou dividir um cluster em dois subclusters em uma única etapa, alguns algoritmos hierárquicos unem mais de dois subclusters ou divide um cluster em mais de dois subgrupos. No agrupamento hierárquico, pode-se representar o agrupamento de objetos como uma árvore chamada dendrograma, na qual o nós representam subconjuntos do conjunto de dados de entrada. Os nós folha correspondem aos elementos individuais do conjunto de dados, e o *root* corresponde a todo o conjunto de dados (KUMAR; SINGH, 2019).

Cada borda no dendrograma representa uma relação de inclusão, conforme Figura 2.6.

Figura 2.6: Dendrograma para um agrupamento hierárquico aglomerativo.



Fonte: Elaborado pelo autor.

No campo de aprendizado de máquina supervisionado, vários algoritmos podem ser utilizados. Neste trabalho iremos abordar o Algoritmo Naive Bayes e SVM.

Classificador Naive Bayes

O Naive Bayes é um classificador probabilístico simples e eficiente baseado no Teorema de Bayes, que foi proposto por Thomas Bayes. Ele é frequentemente usado para problemas de classificação em

aprendizado de máquina e mineração de dados.

O Naive Bayes é baseado na suposição de independência condicional entre as características dadas a classe. Isso significa que ele considera que as características são independentes entre si, dada a classe a que pertencem (daí o termo "naive" ou "ingênuo").

O algoritmo usa o Teorema de Bayes para calcular a probabilidade de uma amostra pertencer a uma classe específica. O Teorema de Bayes é dado por:

A probabilidade posterior da classe dado um conjunto de características é dada por:

$$P(\text{classe} \mid \text{características}) = \frac{P(\text{características} \mid \text{classe}) \cdot P(\text{classe})}{P(\text{características})}$$

Onde:

- $P(\text{classe} \mid \text{características})$ é a probabilidade posterior da classe dado um conjunto de características;
- $P(\text{características} \mid \text{classe})$ é a probabilidade de observar as características dado que a amostra pertence a uma classe específica;
- $P(\text{classe})$ é a probabilidade prior da classe, ou seja, a probabilidade da classe ocorrer antes de observarmos qualquer característica;
- $P(\text{características})$ é a probabilidade das características observadas.

A suposição de independência condicional simplifica o cálculo, pois permite que as probabilidades condicionais sejam estimadas independentemente para cada característica (BONACCORSO, 2017).

Um dos pontos fortes do Naive Bayes é sua simplicidade, sendo considerado um algoritmo simples e fácil de implementar, o que torna sua implementação eficiente e rápida. Outro ponto forte é seu baixo requisito de dados de treinamento, mesmo com um pequeno conjunto de dados de treinamento, o Naive Bayes pode produzir boas estimativas. O Naive Bayes em geral, é computacionalmente eficiente, tornando-o uma escolha popular para problemas de classificação em grande escala e possui

bom desempenho em certos cenários, embora suposição de independência condicional nem sempre seja verdadeira na prática (WICKRAMASINGHE; KALUTARAGE, 2021).

Um dos pontos fracos do Naive Bayes é a suposição de independência condicional, que nesse caso pode ser muito simplista para certos tipos de dados, o que pode levar a uma precisão inferior em alguns casos. Outro pontos que o Naive Bayes não lida bem, são as características que não estavam presentes no conjunto de treinamento, pois não poderia calcular as probabilidades corretamente e estimativas de probabilidade zero, que em alguns casos, pode haver combinações de características que nunca foram vistas no conjunto de treinamento, o que pode levar a uma probabilidade zero e resultados não confiáveis (WICKRAMASINGHE; KALUTARAGE, 2021).

O Naive Bayes é amplamente utilizado em diversas aplicações, incluindo:

- Classificação de texto: É frequentemente usado para tarefas de classificação de texto, como categorização de documentos, análise de sentimentos e filtragem de spam;
- Mineração de dados: Pode ser usado para classificar itens em sistemas de recomendação;
- Diagnóstico médico: É aplicado em problemas de diagnóstico médico para prever doenças com base em sintomas observados;
- Detecção de fraudes: Pode ser usado para identificar transações fraudulentas em sistemas de detecção de fraude.

Em resumo, o Naive Bayes é um classificador simples e eficiente, especialmente útil em situações com um conjunto de dados relativamente pequeno e quando a suposição de independência entre as características é razoável. No entanto, é importante ter em mente suas limitações, especialmente em cenários onde as características são altamente dependentes entre si.

Support Vector Machine (SVM)

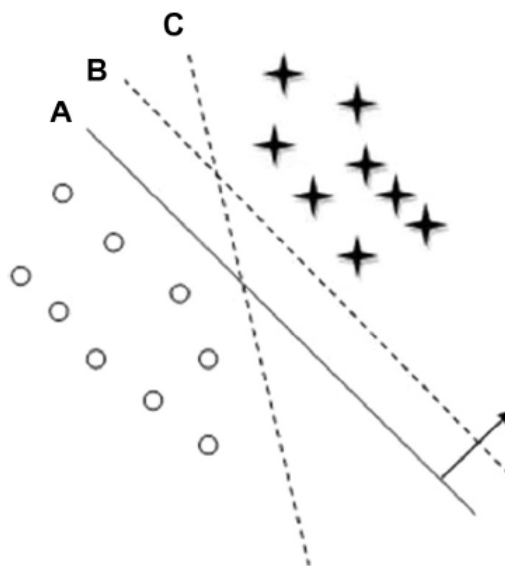
O algoritmo Support Vector Machine (SVM) é uma técnica de aprendizado de máquina supervisionado amplamente utilizada para tarefas de classificação e regressão. O SVM é especialmente eficaz em problemas de classificação binária, onde o objetivo é separar dados em duas classes distintas (VISHWANATHAN; MURTY, 2002).

O SVM encontra um hiperplano (um plano em um espaço de dimensão superior) que melhor separa os dados em classes distintas. O hiperplano é escolhido de forma que a margem (a distância entre o hiperplano e os pontos de dados mais próximos de cada classe, conhecidos como vetores de suporte) seja maximizada (VISHWANATHAN; MURTY, 2002).

O algoritmo busca encontrar o hiperplano ótimo através da minimização de uma função de custo que considera a margem e a classificação correta dos dados de treinamento. Quando os dados não são linearmente separáveis, o SVM pode usar truques de kernel para mapear os dados para um espaço de maior dimensão, onde se torna possível encontrar um hiperplano separador.

Na Figura 2.7, existem 2 classes e 3 hiperplanos A, B e C. O hiperplano "A" fornece a melhor separação entre as classes, pois a distância normal de qualquer um dos pontos de dados é a maior, portanto representa a margem máxima de separação. Dados de texto são ideais para classificação SVM devido à natureza esparsa do texto, em que poucos recursos são irrelevantes, mas tendem a ser correlacionados entre si e geralmente organizados em categorias linearmente separáveis (VISHWANATHAN; MURTY, 2002).

Figura 2.7: Usando SVM em uma classificação problema.



Fonte: Elaborado pelo autor.

O SVM pode construir uma superfície de decisão não linear no espaço de características original mapeando as instâncias de dados de forma não linear para um espaço de produto interno onde as classes podem ser separadas linearmente com um hiperplano (VISHWANATHAN; MURTY, 2002).

Os SVMs são utilizados em muitas aplicações, dentre essas aplicações estão a classificação de revisões de acordo com sua qualidade.

Chen e Tseng (2011) usaram duas abordagens multiclasse baseadas em SVM: One-versus-All SVM e Single-Machine Multiclass SVM para categorizar revisões. Eles propuseram um método para avaliar a qualidade da informação em revisões de produtos considerando-a como um problema de classificação. Eles também adotaram uma estrutura de qualidade da informação para encontrar um conjunto de recursos orientados à informação. Eles trabalharam em câmeras digitais e revisões de MP3. Seus resultados mostraram que seu método pode classificar com precisão as revisões em termos de sua qualidade superando significativamente os métodos de última geração.

2.3 Ensemble Learning

O Ensemble Learning, uma técnica amplamente reconhecida em Inteligência Artificial e Aprendizado de Máquina, se destaca por sua capacidade de combinar múltiplos modelos de aprendizado para melhorar o desempenho preditivo e a generalização. Este método, fundamentado na premissa de que a combinação de modelos individuais pode compensar as fraquezas de cada um, tem sido objeto de extensa pesquisa e aplicação em diversas áreas da computação (HASTIE et al., 2009).

Segundo (DIETTERICH, 2000), o Ensemble Learning busca construir um modelo preditivo robusto através da combinação ponderada de várias hipóteses individuais $h_i(x)$, onde x representa os dados de entrada. Formalmente, dado um conjunto de modelos $H = \{h_1, h_2, \dots, h_N\}$, o modelo ensemble $H_{ens}(x)$ é definido como:

$$H_{ens}(x) = \sum_{i=1}^N \alpha_i h_i(x)$$

onde α_i representa os pesos atribuídos a cada modelo, geralmente determinados durante o processo de treinamento.

As principais estratégias de Ensemble Learning são:

1. Bagging (Bootstrap Aggregating):

- Bagging (Bootstrap Aggregating): O Bagging consiste em treinar vários modelos independentes usando subconjuntos aleatórios dos dados de treinamento.

- Esses modelos são então combinados por votação (para problemas de classificação) ou média (para problemas de regressão) para fazer previsões mais robustas.

2. **Boosting:**

- O Boosting é uma abordagem iterativa que foca em construir uma sequência de modelos fracos, onde cada modelo subsequente é treinado para corrigir as deficiências dos modelos anteriores.
- Algoritmos populares de boosting incluem AdaBoost, Gradient Boosting Machines (GBM) e XGBoost.

3. **Random Forest:**

- Uma extensão do Bagging, o Random Forest cria uma coleção de árvores de decisão durante o treinamento.
- Cada árvore é construída usando uma amostra aleatória dos dados e, durante a previsão, as previsões de todas as árvores são combinadas para determinar a saída final.

O Ensemble Learning é amplamente considerado uma das abordagens mais eficazes para melhorar o desempenho preditivo e a robustez dos modelos de aprendizado de máquina. Sua aplicação é de grande relevância na área computacional e de inteligência artificial por conta da melhoria do desempenho, redução do overfitting, flexibilidade, adaptabilidade, robustez e tolerância a falhas.

Segundo (ZHOU, 2012), o Ensemble Learning pode utilizar uma variedade de algoritmos como base para seus modelos individuais, incluindo Regressão Logística, Naive Bayes e Máquinas de Vetores de Suporte (SVM). Cada modelo contribui com suas próprias perspectivas e abordagens para a tarefa de aprendizado, e a combinação desses modelos pode levar a um desempenho geral melhorado e a uma maior capacidade de generalização.

2.4 **Análise de Sentimentos**

Com a crescente popularidade da análise e da ciência de dados, os métodos da inteligência computacional estão provando serem importantes ferramentas competitivas em muitas indústrias. Na

análise de negócios, os dados são extraídos para padrões que ajudariam entender melhor os clientes, as vendas e o marketing. Métodos de inteligência computacional permitem que métodos probabilísticos sejam usados para encontrar padrões em dados. Esses métodos normalmente funcionam em dados de baixo nível e não são guiados pelo conhecimento absoluto como é o caso dos métodos gerais de IA. Além disso, uma grande quantidade de dados que justifica a análise é agora gerado em forma escrita. Por exemplo, os usuários podem deixar comentários escritos sobre um produto ou serviço em sites ou redes sociais (KAUTISH, 2017).

O texto escrito está sujeito a interpretação e representa os dados em uma sintaxe absoluta (como um sistema binário). No entanto, os métodos de inteligência computacional permite tal imprecisão, e pode ser o mais apropriado métodos para encontrar padrões em tais dados (KAUTISH, 2017).

Embora a linguística e o Processamento de Linguagem Natural (PLN) tenham uma longa história, poucas pesquisas foram feitas sobre as opiniões e sentimentos das pessoas antes do ano 2000. Desde então, o campo tornou-se uma área de pesquisa muito ativa (KAUTISH, 2017).

Segundo Chong, Selvaretnam e Soon (2014), a Análise de Sentimentos é a área da PLN que envolve ensinar os computadores a identificar os sentimentos por trás do conteúdo escrito ou áudio analisado.

Os principais desafios e problemas na Análise de Sentimentos tem sido investigado em três níveis:

- **Nível de documento:** A tarefa neste nível é classificar se um documento de opinião inteiro expressa um sentimento positivo ou negativo (PANG; LEE; VAITHYANATHAN, 2002). Por exemplo, dada uma avaliação do produto, o sistema determina se a avaliação expressa uma opinião geral positiva ou negativa sobre o produto;
- **Nível de sentença:** A tarefa neste nível vai para as frases e determina se cada frase expressou uma opinião positiva, negativa ou neutra. Esse nível de análise está intimamente relacionado à classificação de subjetividade, que distingue sentenças (chamadas sentenças objetivas) que expressam informações factuais de sentenças (chamadas sentenças subjetivas) que expressam visões e opiniões subjetivas (WIEBE; BRUCE; O'HARA, 1999);
- **Nível de entidade e aspecto:** Tanto o nível do documento quanto a análise do nível da frase

não descobrem exatamente o que as pessoas gostaram e não gostaram. O nível de aspecto executa uma análise mais refinada. O nível de aspecto foi anteriormente chamado de nível de recurso (mineração e sumarização de opinião baseada em recursos). Em vez de olhar para construções de linguagem (documentos, parágrafos, sentenças, orações ou frases), o nível de aspecto olha diretamente para a opinião em si (HU; LIU, 2004).

A questão essencial na Análise de Sentimentos é identificar como os sentimentos são expressos e se as expressões indicam opiniões positivas (favorável) ou negativas (desfavoráveis) em relação ao assunto (NASUKAWA; YI, 2003).

2.4.1 Aplicações

A Análise de Sentimentos nos negócios pode ser um grande avanço para a melhoria completa. A chave é administrar um negócio produtivo com a informação emocional, que é o poder de usar a informação não estruturada para percepções injustas. Os modelos de aprendizado de máquina dependem de opções criadas manualmente, antes da classificação. No entanto, a aprendizagem profunda também pode ser uma possibilidade robusta (MISHNE; GLANCE et al., 2006).

Aplicações em domínios completamente diferentes

Técnicas de classificação de sentimentos aplicadas em informações totalmente diferentes, como por exemplo: resenhas, discursos na internet, artigos de notícias, análise e revisão de produtos e análises de músicas. Análise de sentimentos do discurso inclui dispositivo de computação social, grupos de notícias e comentários (Facebook, Twitter, WhatsApp). Este domínio geralmente extrai sentimentos de questões específicas como aquecimento global, política, entre outros.

Aplicações para sites relacionados a avaliações

A aplicação mais comum da Análise de Sentimentos é dentro da área de revisões de mercadorias e serviços de consumo. Existem muitos sites que oferecem resumos automáticos de revisões sobre mercadorias e seus aspectos específicos. Um exemplo notável disso é o "Google" (KAUTISH, 2017).

Twitter e Facebook ambos são um destaque das várias análises de sentimento. A análise de sentimento pode oferecer valor substancial aos candidatos que concorrem a cargos variados. Isto

permite que os gerentes de campanha rastreiem, os assuntos que preocupam os eleitores e também os meios com que se relacionam com os discursos e ações do candidatos.

Aplicações em finanças

Outro domínio muito importante para a Análise de Sentimentos é o mercado financeiro (à vista). São variadas notícias, artigos, blogs e tweets sobre as companhias. Um sistema de análise de grau de sentimentos, pode usar essas variadas fontes para buscar artigos que dialogam com as empresas e combinam sentimentos em relação a eles coletivamente (KAUTISH, 2017).

2.4.2 Classificação e Extração

A classificação pode ser realizada de acordo com várias dimensões ("eixos") ortogonais entre si (KAUTISH, 2017).

- Por tópico: de longe o caso mais frequente, suas aplicações são onipresentes;
- Por sentimento: útil em pesquisa de mercado, gerenciamento de reputação online, ciência e ciencia politica;
- Por idioma: útil, no processamento de consultas nos motores de busca;
- Por gênero: por exemplo, Automotivo Noticias vs Automotivo Blogs, útil no site classificação e outros;
- Por autor: por idioma nativo ("identificação do idioma nativo");
- Por gênero: útil em aplicações forense e cibersegurança. Exemplo, produto de avaliações.

2.4.3 Classificações no nível do documento

A Análise de Sentimentos no nível do documento, visa analisar todo o documento e atribuir um sentimento geral para isso, assumindo que o documento expressa um sentimento. Essa suposição é criticada como irracional como resultado de um texto que pode ter uma opinião e, portanto, a análise deve visar níveis mais sutis (KAUTISH, 2017).

No entanto, essa suposição vale para alguns ramos, como revisões, sempre que é necessária uma declaração final sobre a mercadoria que pode ser uma ponderação conclusão decorrente de aspectos completamente diferentes, embora a revisão carregue opiniões diferentes.

Classificações no nível da frase

A classificação no nível de um frase pode ser um nível mais refinado de pesquisa que inspeciona sentenças que especificam uma opinião e sua orientação. Segundo Kautish (2017), é vital diferenciar sentenças subjetivas de não subjetivas. As declarações subjetivas indicam uma opinião específica de uma pessoa. A detecção de subjetividade e o ato de separar frases que contém opiniões de pessoas e contém fatos.

Classificação no nível do recurso

Classificação no nível do recurso ocorre quando duas tarefas importantes de mineração são executadas. A primeira, determina todas as características expressas de um objeto e a segunda, identifica as opiniões expressas sobre as características de um objeto, que pode ser uma observação positiva, negativa ou neutra (KAUTISH, 2017).

Estratégias de classificação de sentimentos

Existem 3 estratégias completamente diferentes na Análise de Sentimentos: Estratégias baseadas em aprendizado de máquina; Estratégias baseada em léxico e Análise linguística (THELWALL; BUCKLEY; PALTOGLOU, 2011).

Estratégias baseadas em aprendizado de máquina são modelos suportados, que são marcados, para conhecimento categorizado. Este conhecimento é chamado de dados de treinamento. O modelo de marca ou máquina será então usado para categorizar novos conhecimentos, assim como uma equação parametrizada pode ser usada para prever o valor da variável resposta na análise multivariada. O treinamento se baseia em opções que influenciam a polaridade da informação, e que eles são escolhidos usando estratégias de seleção de recursos.

A abordagem baseada em léxico depende da construção de um léxico, que pode ser uma "estrutura que mantém o registro de palavras e presumivelmente informações sobre elas" onde quer que as

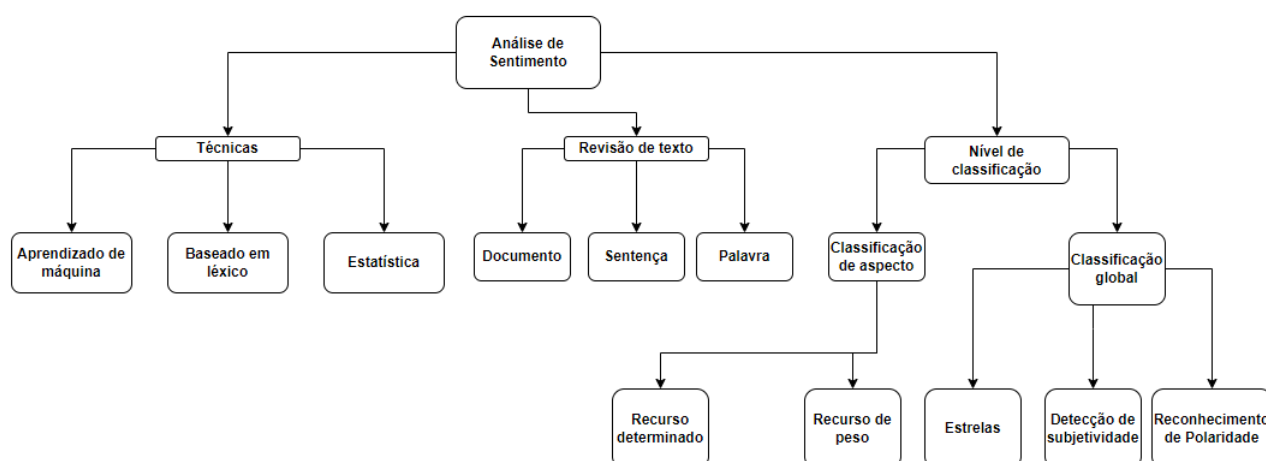
palavras sejam citadas como "ítems lexicais"(CHARNIAK, 1996). Uma vez que o léxico é construído, a polaridade geral do texto é então encontrada por uma contagem presumivelmente ponderada desses coisas lexicais.

Por fim, a abordagem linguística usa as características sintáticas das palavras, frases, negações, e também a estrutura do texto para estimar a orientação do texto (DING; LIU; YU, 2008).

Todas as estratégias utilizadas na abordagem linguística se baseia em Partes-da-Fala (POS). POS define os padrões sintáticos ou classes das palavras. Padrões de POS podem ser usados como frases a serem escolhidas a dedo no texto. Esses padrões indicam um sentimento específico ou um tópico específico. Isto conteria adjetivos ou advérbios ou uma outra parte do discurso.

Segundo Kautish (2017), para delinear os padrões, n-grams são usados. Um n-gram pode ser uma sequência de n palavras de uma determinada sequência de fala. Pode-se usar unigramas, bigramas, trigramas e n-gram para três palavras. Suponha que n-gramas sejam usados para procurar frases casuais. As três diferentes abordagens podem ser usadas individualmente ou combinadas, como apresentado na Figura 2.8.

Figura 2.8: Abordagens de classificação.



Fonte: Elaborado pelo autor.

2.4.4 Extração de recursos

A extração de recursos identifica os aspectos do produto que estão sendo comentados por clientes. A previsão de sentimento identifica o texto que contém sentimento ou opinião decidindo a polaridade do sentimento como positiva, negativa ou neutra. O último módulo é a sumarização, que agrega os resultados obtidos. O método de extração de recursos recebe o texto como entrada e gera opções em qualquer uma das formas como léxico-sintática ou retórica, sintática e principalmente discurso baseado (DAS; BANDYOPADHYAY, 2010); (ABBASI; CHEN; SALEM, 2008).

Abordagens baseadas principalmente em texto

Bag-of-Words (BoW): Pode ser um recurso padrão, com um valor pré-definido vocabulário, bem como um grupo de termos (palavras individuais ou n-gramas de palavras), um documento é representado como um vetor de recursos, onde cada componente pode ser um valor binário indicando a aparência do termo correspondente ou um valor de enumeração que indica a frequência do termo (TF). O vocabulário é às vezes criado pela escolha dos primeiros termos que apareceram dentro do corpus. Para Análise de Sentimentos, termos relacionados à emoção podem ser adicionalmente anexo. Além disso, considerando que os termos poderiam ter importância diferente, cada termo pode ser atribuído com um peso como o inverso frequência de documentos (IDF) (KOBAYASHI; INUI; MATSUMOTO, 2007).

Léxico: Consiste em palavras emocionais que são atribuídas manualmente com sentimento/-pontuações. Além das abordagens de aprendizagem da matemática aplicada com a matéria, os léxicos podem ser usados diretamente para decidir o sentimento do tweet. Normalmente, um tweet é positivo se a pontuação geral for positiva, caso contrário o tweet é negativo (KOBAYASHI; INUI; MATSUMOTO, 2007).

Dependência Sintática: É definida como dependência de palavras que é baseada em recursos gerados a partir de árvores de dependência ou análise. Vários pesquisadores podem usá-lo em suas pesquisas no campo da Análise de Sentimento (KOBAYASHI; INUI; MATSUMOTO, 2007).

Marcação de parte de fala (POS): Componentes de fala ou marcação de POS podem ser uma técnica linguística usada desde 1960 e recentemente tem recebido atenção explícita de pesquisadores de tecnologia de linguagem humana Jeong, Shin e Choi (2011), para extração de recursos do produto

como aspectos do produto, que são geralmente substantivos ou frases nominais. Marcação POS Aggarwal e Zhai (2012) atribui uma tag para cada palavra durante um texto e classifica uma palavra para uma determinada classe morfológica como norma, verbo, adjetivo, etc.

Parar a remoção de palavras: Parada de palavras são palavras comuns e de alta frequência como "uma", "a", "de", "e", "um". Estratégias completamente diferentes no mercado para eliminação de palavras de parada; em última análise, melhorar o desempenho da extração de recursos algoritmos (AGGARWAL; ZHAI, 2012).

Limpeza de HTML: As páginas da Web contêm, além dos textos principais, anúncios e outras informações irrelevantes, como tags HTML (por exemplo, <p>,
). Estes são organizados em diferentes elementos de objeto, ou seja, as chamadas tags <div>. Evitar problemas de eficiência que surgem dessa informação irrelevante, o texto deve ser limpo para reter apenas as informações de interesse. Existem muitas maneiras de extrair os textos de notícias relevantes do código-fonte HTML. Por exemplo, pode-se usar "HTML Cleanup" que foi usado em Das (2011), ou o modelo de objeto de documento (DOM), ou a unidade de HTML da biblioteca Apache que pode analisar as características específicas de HTML de textos, e organizá-los em um objeto-estrutura de árvore baseada para ser distinguida e separada umas das outras. Essa estrutura permite extrair o texto principal de interesse.

Abreviação expandida: As abreviações podem criar ruído durante a análise e este problema é resolvido por abreviação expandida. Por exemplo, "they're" é substituídos por "they are", "hasn't" é substituído por "has not". Isso ajuda, por um lado, a obter a frequência correta das palavras e, em seguida, a dimensão correta do texto. Por outro lado, expandir a parte de negação como em "don't like" para "do not like" ajuda a detectar e marcar a negação nas frases e facilitar sua detecção (DAS, 2011).

Derivação: Stemming é o processo de deletar o sufixo da palavra e colocá-lo em sua base. Se um texto contém as palavras admirar e admirando, eles não devem ser tratados como palavras diferentes, especialmente em um sentimento classificador onde eles têm o mesmo significado e a mesma polaridade. Desdobrá-los fará com que eles voltem a "admirar" e então a frequência da palavra será 3, ou seja, em vez de três palavras com frequência 1. A importância deste passo está em diminuir a dimensionalidade do texto (FEINERER; HORNIK; MEYER, 2008).

2.4.5 Análise de Sentimentos Multimodal (ASM)

Devido ao desenvolvimento adequado de ferramentas, a Análise de Sentimentos se limitou principalmente a textos, com exceção de alguns conteúdos visuais. A pesquisa mostra que, a expressão visual das emoções contém mais informação do que o texto. No entanto, no caso do ASM, os dados de cada modalidade deve ser processada separadamente para obter sentimentos. Em seguida, os resultados de cada modalidade devem ser integrados para obter polaridade final ou estados afetivos. Além disso, estudos mostram que, a pesquisa sobre cada modalidade apresenta diferentes dificuldades apenas ou como um todo.

Os pesquisadores (VOHRA; TERAIIYA, 2013) fazem um levantamento sobre ideias de Análise de Sentimentos, sua aplicação e dificuldades potenciais pela acentuação apenas na AS textual, ignorando os problemas da ASM.

Bhat, Deshpande e Sugandhi (2014) realizaram uma pesquisa no conjunto de dados Multimodal AS, utilizando estratégias para organizar o conjunto de informações, técnicas de AS e dificuldades futuras e oportunidades neste campo até certo ponto. Outro trabalho foi a análise sobre dados e modelos de sentimentos multimodais sendo revisados por Fulse, Sugandhi e Mahajan (2014), onde, mencionaram as dificuldades e oportunidades de integração do resultados de vários modos em um sistema de reorganização emocional.

Ravishankara, Dhanush e Srajan (2020) pesquisou as abordagens de tarefas e aplicações de AS e mencionou em detalhes quanto às oportunidades e dificuldades da análise do AS, por mais despercebidas análise ASM.

Medhat, Hassan e Korashy (2014) realizaram uma pesquisa minuciosa apenas em algoritmos AS, não levando em consideração o ASM. Eles categorizaram os algoritmos na ideia de seu uso dentro do conjunto de dados e dentro do domínio.

Sikandar (2014) conduziu a revisão, onde mencionou técnicas de fusão completamente diferentes e seu resultado do sistema de reconhecimento de emoções. Além disso, ela mencionou os problemas de desempenho e robustez com a ASM em detalhes, juntamente com futuros desafios e oportunidades.

Osimo e Mureddu (2012) se concentrou em encontrar a lacuna de análise ao lado dos desafios de análise em AS para futuros pesquisadores neste campo. Eles também resumem a análise atual, que está sendo conduzida durante este campo, juntamente com problemas de análise de curto prazo e

futuros, ignorando o ASM. A maioria dos pesquisadores no topo da discussão não notou problemas de ASM. Embora alguns deles tenham notado os problemas, no entanto, eles eram de natureza restrita.

2.4.6 Sumarização

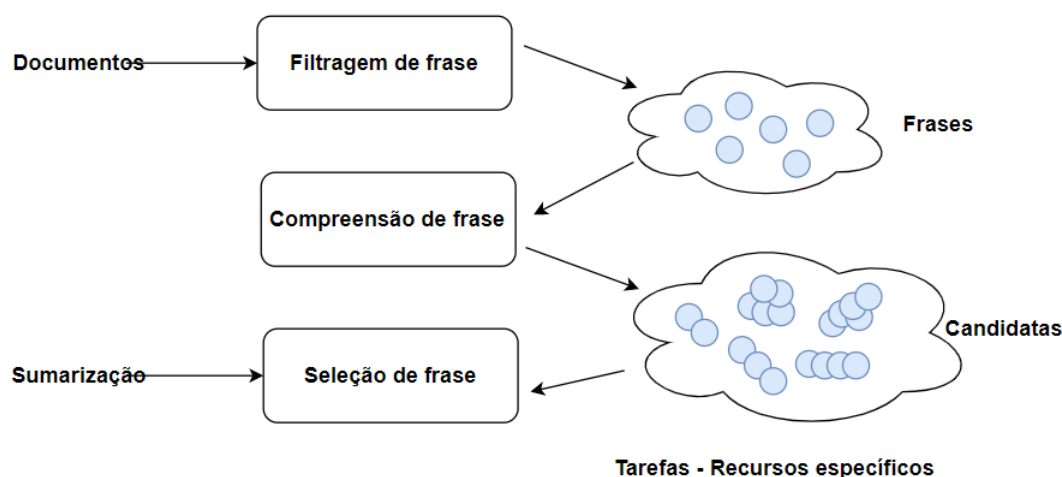
A sumarização produz uma versão de metade do texto que contém informações vitais ou relevantes para o usuário. Ela descreve ou abstrai o documento e simplifica o texto comprimindo frases. Duas coisas são feitas resumindo um único documento para fornecer resumo e resumindo vários documentos para fornecer séries de histórias ou histórias no mesmo evento ou conjunto de conteúdo da web sobre algum tema (BAHRAINIAN; DENGEL, 2013).

Existem basicamente dois tipos de sumarização, primeiro é a sumarização genérica, onde tendemos a resumir o conteúdo do documento e a segunda focada na qual tendemos a resumir em relação ao desejo expresso na pergunta do usuário. Existem duas maneiras de sumarizações, a primeira é a sumarização extrativa que resultam em esboços de frases ou sentenças no documento de origem e, em segundo lugar, resumo abstrato sempre que esboçam idéias específicas no documento de origem usando palavras completamente diferentes. A fórmula de sumarização da linha de base requer a sentença inicial porque representa melhor o documento. Em suma, há principalmente três estágios (RADEV et al., 2004); (GANESAN; ZHAI; HAN, 2010).

- Seleção de conteúdo - extraíndo frases que queremos do documento;
- Ordenação da Informação - opte pela ordem para colocar as frases em esboço;
- Realização de Sentença - simplificação da frase no esboço.

Para previsão, queremos investigar a enorme quantidade de informações e resumir para exigir a chamada. Primeiro, tendemos a estabelecer as categorias de sentimento e resumir para pesquisar domínios completamente diferentes e criar análises suportadas por chamadas. A Figura 2.9 apresenta um arquitetura básica da estrutura de sumarização.

Figura 2.9: A arquitetura básica da estrutura de sumarização.



Fonte: Elaborado pelo autor.

Sumarização de documento único

É o processo de extração de dados de um único texto. Geralmente gera pequenos resumos (RADEV; BLAIR-GOLDENSOHN; ZHANG, 2001).

A maioria das abordagens para sumarização de documento único envolve a extração de frases do documento e a maioria dos sistemas de sumarização de documento único, até certo ponto, faz uso da estrutura monolítica do documento.

Por exemplo, uma maneira simples, mas bastante eficaz, de escrever um resumo para um único documento é pegar a primeira frase de cada parágrafo e juntá-las em sua ordem original.

Sumarização de vários documentos

É um procedimento automático voltado para a extração de dados de vários textos escritos sobre temas idênticos. A sumarização de esboços subsequente permite que usuários individuais, como compradores de informações experientes, se familiarizem rapidamente com as informações contidas em um grande conjunto de documentos. Dessa forma, os sistemas de sumarização de vários documentos estão complementando as artes cênicas do agregador de notícias na etapa subsequente de lidar com a sobrecarga de informações. A sumarização de vários documentos também pode ser desgastada resposta a um assunto (BOSSARD; RODRIGUES, 2011).

A sumarização de vários documentos cria sumarizações de informações compactas e abrangentes. Com opiniões completamente diferentes sendo colocadas e tornadas públicas, cada tópico é representado a partir de várias visualizações dentro de um documento. Considerando que o objetivo de um esboço rápido é mudar a busca de informações e reduzir o tempo apontando para a oferta mais relevante documentos, o esboço abrangente de vários documentos deve conter as informações desejadas, limitando, portanto, o requisito de acesso aos arquivos originais aos casos, uma vez que o refinamento é necessário.

A sumarização de documento único é muito relevante para simplificar as informações. Um sistema de sumarização multidocumento é perfeito não só encurta os textos, mas também apresenta informações organizadas em torno dos aspectos-chave para representar inúmeros Visualizações. O sucesso produz um resumo de um determinado tópico. Tais compilações de texto também devem atender às necessidades básicas de um texto de resumo compilado por uma pessoa. Os critérios de qualidade do esboço de vários documentos são:

- Estrutura clara, bem como uma visão de nível superior da maior parte do conteúdo;
- O texto dentro das seções é dividido em parágrafos significativos;
- Transição gradual de muitos aspectos temáticos gerais para muitos aspectos específicos;
- Boa legibilidade.

Sumarização automática

A sumarização automática de texto é o método de fazer mecanicamente uma versão compactada de um determinado texto que contém informações úteis para o usuário. Além disso, o próprio fato de as informações continuarem aumentando e as pessoas acabarem com a antiga metodologia (baseada em papel) de armazenamento de informações, a sumarização automática é uma das maneiras de recuperar informações do banco de dados, principalmente da web. Segundo Kautish (2017), a sumarização automática é o ato de recuperar informações convincentes do(s) documento(s) e apresentá-las como esboço.

Com a computação em nuvem, data warehousing e big data onde bancos de dados e repositórios totalmente diferentes podem ser acessados, obter informações corretas tem sido difícil. Todas as

informações relevantes e discutíveis são abertas aos usuários, no entanto, para induzir as informações precisas necessárias, normalmente leva mais tempo e, além disso, causa sobrecarga de informações. Uma quantidade excessiva de informações é obtida, mas obter o texto de conexão desses documentos eletrônicos, apesar de milhões de documentos conectados, torna a sumarização automática uma tarefa a ser explorada. A sumarização automática ajuda a economizar tempo de processo de conhecimento e, além disso, ajuda a criar, por remoção em informações, uma quantidade menor que se espera usar.

Ainda, segundo Kautish (2017), a ambiguidade das palavras pode ser um grande problema que ainda precisa ser resolvido na AS (resumo automático). Palavras com significados totalmente diferentes (como banco ao lado de um riacho ou onde quer que o dinheiro seja guardado) podem aparecer no mesmo texto, mas isso pode ser mal interpretado por significados diferentes. Também anáforas como a palavra de função (por exemplo, ele) que se refere a uma entidade na(s) sentença(s) anterior(es) podem ser uma boa preocupação, particularmente quando a sentença é estratificada acima das sentenças de referência.

Soriyan e Omodunbi (2014) usou a distinção de texto para se livrar de frases duplicadas, definindo o limite da frase para zero. No entanto, isso não removeu totalmente a redundância como resultado de sua falta de treinamento em linguística. Depois de desenvolver resumos automáticos para imitar resumos humanos, uma enorme lacuna ainda deve ser preenchida.

Capítulo 3

Trabalhos Relacionados

Nas Subseções 3.1 e 3.2 são descritos trabalhos relacionados com o presente trabalho, tanto no âmbito de análise de sentimento utilizando processamento de linguagem natural, quanto na análise de sentimento utilizando aprendizado de máquina e aprendizado profundo.

3.1 Análise de Sentimento utilizando PLN

Em Qiu et al. (2010), é usada uma abordagem baseada em dicionário para identificar o sentimento de sentenças no contexto de publicidade. No trabalho, foi proposta uma estratégia de publicidade para melhorar a relevância dos anúncios e a experiência do usuário. Os autores trabalharam com dados oriundos de fóruns da web. Os resultados demonstraram que o modelo proposto teve bom desempenho na extração de palavras-chave de publicidade e seleção de anúncios. Já em (HATZIVASSILOGLOU; MCKEOWN, 1997), foi usado uma lista inicial de adjetivos de opinião juntamente com um conjunto de restrições linguísticas para identificar mais palavras de opinião e suas orientações.

3.2 Análise de Sentimento utilizando Aprendizado de Máquina

Em Kang, Yoo e Han (2012), é usado um classificador Naive Bayes otimizado para resolver o problema de tendência de maior acurácia na classificação de amostras positivas (cerca de 10% maior que em amostras negativas). Isto cria o problema de diminuição da precisão média quando a acurácia para as duas classes é contabilizada de maneira conjunta. O trabalho mostrou que usando

este algoritmo juntamente com um banco de dados de análises de restaurantes foi possível diminuir a diferença de precisão entre as classes quando comparado ao Naive Bayes tradicional e ao Support Vector Machine (SVM). As medidas de recall e precision também melhoraram.

Em Tiwari e Singh (2019), os autores propõe o Modelo Ensemble, que consiste na combinação de técnicas de pré-processamento e algoritmos sobre uma base de dados. Esse dataset é composto de comentários de passageiros das companhias aéreas americanas, disponível no site Kaggle.com. O Ensemble é conhecido por ser uma técnica que visa melhorar a acurácia na identificação das polaridades, que nesse caso, poderia ser "negativo", "neutro" ou "positivo". Os algoritmos utilizados foram: Decision tree, KNeighbor, Naive Bayes, AdaBoost e ExtraTree. Além dos algoritmos, o pré-processamento também foi utilizado, afim de melhorar a qualidade dos dados para obtenção de melhores resultados. Apesar que o método ter obtido uma boa classificação das polaridades, sendo o melhor resultado obtido através do algoritmo ExtraTree, com uma acurácia de 76%, o presente apresenta resultados de precisão, revocação e F1-Score, colocando em dúvida a estabilidade dos resultados obtidos.

A abordagem não supervisionada foi usada também em Xianghua et al. (2013) para detectar automaticamente os aspectos discutidos em redes sociais chinesas e também os sentimentos expressos em diferentes aspectos. Foi usado um modelo LDA (Latent Dirichlet Allocation) para descobrir temas globais multi aspecto de comentários sociais, onde os pesquisadores extraíram o tema local e sentimento associado com base em um contexto de janela deslizante sobre o texto. Eles trabalharam em comentários sociais que foram extraídos de um conjunto de dados do blog (2000-SINA) e um léxico (300-SINA HowNet). Mostraram ainda que a sua abordagem obteve bons resultados na separação dos temas e melhorou a precisão da análise de sentimento. O modelo também ajudou a descobrir diversos aspectos nos tópicos e sentimento associado.

Em Phan et al. (2020), propôs um método para melhorar o desempenho da análise de sentimentos em tweets contendo sentimentos difusos. O modelo de conjunto de recursos foi criado pela concatenação de informações de cinco vetores de recursos extraídos do léxico, do tipo de palavra, da semântica, da polaridade do sentimento e da posição das palavras em tweets contendo frases de sentimento difuso. O resultado obtido com esse modelo é o tweet embeddings, que foi usado como vetores de recursos na camada de entrada do modelo CNN. A análise experimental revelou que o

método proposto melhorou significativamente o desempenho na análise de sentimentos de tweets contendo sentimentos difusos. Há algumas possíveis limitações da abordagem proposta: o método considerou apenas os tweets que continham sentimento difuso sem considerar a influência de outros elementos neles, como gírias e sarcasmo. No futuro, planejamos analisar o sentimento dos tweets considerando outras informações usando o modelo BERT para tweets.

Capítulo 4

Desenvolvimento do Trabalho

O presente trabalho foi dividido em cinco etapas, cumprindo com seu objetivo que era primeiramente carregar dados da rede social Twitter, pré-processar os dados, utilizando técnicas de PLN, treinamento dos modelos, em seguida aplicando o Modelo Ensemble e avaliando o resultado final.

4.1 Estrutura proposta

Este trabalho adota um framework para análise de sentimentos em tweets das principais companhias aéreas dos EUA. Um framework, no contexto deste estudo, pode ser entendido como uma estrutura conceitual que fornece suporte e orientação para a construção e o desenvolvimento de sistemas de análise de sentimentos. Ele define uma arquitetura básica que orienta o processo desde o carregamento dos dados até a avaliação do desempenho dos modelos de aprendizado de máquina.

O framework proposto inicia carregando os dados brutos do arquivo CSV "Tweets.csv", que contém avaliações reais de clientes sobre as companhias aéreas em tweets. Este arquivo é localizado no diretório "/kaggle/input/twitter-airline-sentiment/". Uma vez carregados, os dados passam por um processo de pré-processamento, que inclui a remoção de stopwords e o stemming. Em seguida, são treinados diversos modelos de aprendizado de máquina, como Naive Bayes, Regressão Logística e SVM, utilizando uma abordagem de validação cruzada.

Após o treinamento dos modelos individuais, é utilizado um Modelo Ensemble para combinar as previsões desses modelos e produzir uma previsão final mais robusta. Por fim, os modelos são avaliados em termos de métricas de desempenho, como acurácia, precisão, revocação e F1-Score,

concluindo o fluxo de trabalho.

A Figura 4.1 ilustra o fluxo de trabalho do framework proposto, destacando as principais etapas do processo de análise de sentimentos em tweets.

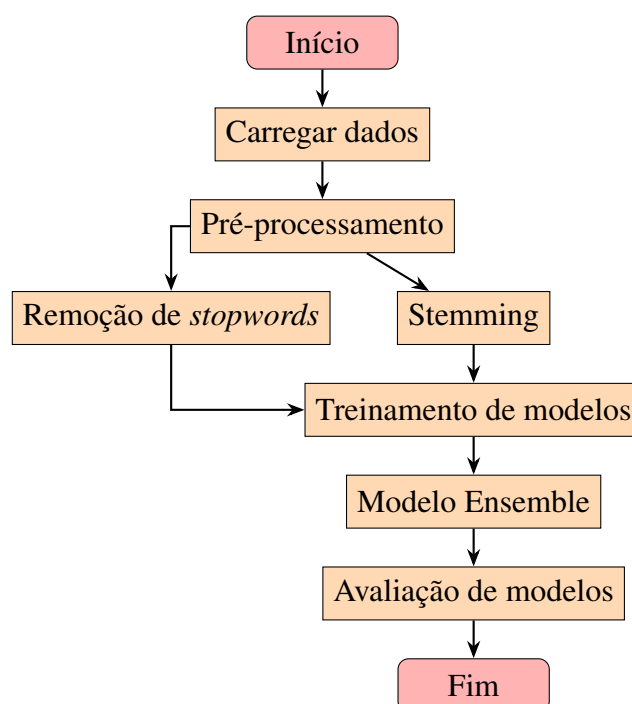


Figura 4.1: Fluxo de trabalho do projeto

4.2 Carregar dados

O processo de coleta de dados neste caso envolve a leitura de um arquivo CSV denominado "Tweets.csv" localizado no diretório "/kaggle/input/twitter-airline-sentiment/". Este arquivo contém dados reais de avaliações de clientes das principais companhias aéreas dos EUA, relacionados aos sentimentos expressos sobre as companhias aéreas em tweets. Inclui informações como o texto do tweet, a companhia aérea mencionada e o sentimento associado. É importante destacar que esses dados reais aumentam o desafio da análise, pois não se trata de dados coletados através de um questionário com respostas controladas, o que pode introduzir maior variabilidade e complexidade na interpretação dos resultados. Os dados são carregados em um DataFrame do pandas, facilitando a manipulação e análise posterior dos mesmos.

A figura 4.2 apresenta os comandos utilizados em Python para carregar os dados.

O conjunto de dados está dividido da seguinte forma:

Figura 4.2: Comandos em Python para carregar dados.

```

import numpy as np
import pandas as pd
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.tokenize import TweetTokenizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import VotingClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

# Carregando o conjunto de dados
df = pd.read_csv('/kaggle/input/twitter-airline-sentiment/Tweets.csv')

```

Fonte: Elaborado pelo autor.

Tabela 4.1: Tabela de Polaridade

Polaridade	Quantidade
Negativo	9178
Positivo	2363
Neutro	3099

4.3 Pré-processamento

O pré-processamento dos dados é uma etapa essencial na análise de texto, especialmente ao lidar com dados não estruturados, como os provenientes de tweets. Nesse contexto, o objetivo primordial é purificar e estruturar os dados brutos para análises subsequentes, eliminando elementos irrelevantes e normalizando o texto.

Inicialmente, o processo de pré-processamento envolve a identificação e remoção de stopwords, que são palavras comuns em um idioma que não contribuem significativamente para o contexto da frase. Por exemplo, palavras como "the", "and" e "is" são frequentemente identificadas como stopwords. Após a aplicação da remoção de stopwords, observou-se a eliminação de palavras como "and", "the" e "is", que são comuns e não agregam significado ao conteúdo dos tweets.

Em seguida, é aplicada a técnica de stemming, que visa reduzir as palavras à sua forma base ou raiz, removendo sufixos e prefixos. Por exemplo, após a aplicação do stemming, as palavras "running", "ran" e "runs" seriam reduzidas ao radical "run". Isso simplifica o vocabulário e melhora a eficácia da análise, permitindo que palavras com significados semelhantes sejam agrupadas.

Além disso, a tokenização é empregada para dividir o texto em unidades menores chamadas

tokens. No contexto dos tweets, a tokenização é especialmente importante devido à presença de elementos específicos, como hashtags, menções de usuários e emojis. Por exemplo, o texto "I love machine learning" seria dividido em tokens individuais, incluindo "I", "love" e "machine learning", permitindo uma análise mais granular do conteúdo dos tweets.

A tokenização é o processo de dividir o texto em unidades menores chamadas tokens. Essas unidades podem ser palavras individuais, partes de palavras (como sufixos ou prefixos), caracteres ou até mesmo símbolos especiais, dependendo das regras definidas durante a tokenização.

A figura 4.3 apresenta os comandos utilizados em Python para realizar o pré-processamento dos dados.

Figura 4.3: Comandos em Python para pré-processamento dos dados.

```
# Pré-processamento e vetorização de texto
stop_words = set(stopwords.words('english'))
ps = PorterStemmer()
tokenizer = TweetTokenizer()

def preprocess_text(text):
    tokens = tokenizer.tokenize(text)
    tokens = [ps.stem(token) for token in tokens if token not in stop_words]
    return ' '.join(tokens)

df['text'] = df['text'].apply(preprocess_text)
```

Fonte: Elaborado pelo autor.

A função `preprocess_text`, apresentada na Figura 4.3, implementa essas etapas de pré-processamento para cada tweet no conjunto de dados. Após o pré-processamento, os tweets resultantes estão prontos para serem vetorizados e utilizados em modelos de aprendizado de máquina para análise de sentimentos ou outras tarefas de processamento de linguagem natural.

Esses exemplos ilustram como as técnicas de pré-processamento, como remoção de stopwords, stemming e tokenização, podem ser aplicadas para preparar dados de texto para análises mais eficazes e significativas.

4.4 Treinamento de modelos

A fase de treinamento do modelo constitui uma etapa crucial na análise de sentimentos em dados de mídia social, como tweets. Neste estágio, os dados coletados anteriormente passam por um processo de preparação e são utilizados para treinar diferentes algoritmos de aprendizado de

máquina. Esses algoritmos são então capazes de aprender padrões nos dados e fazer previsões sobre o sentimento associado a novos tweets.

A definição de um algoritmo e modelo é um passo fundamental para obter resultados precisos e confiáveis. No presente trabalho, escolhemos o algoritmo Regressão Logística, SVM e Naive Bayes para treinar e avaliar os resultados.

A escolha desses algoritmos foi por considerar ser algoritmos de fácil interpretabilidade, sendo especialmente útil para entender a importância de cada variável, além de sua simplicidade, o que a torna-os rápidos de treinar e entender.

A definição do modelo é representada por:

$$h(z) = \frac{1}{1 + e^{-z}}$$

$$z = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_N x_N$$

Dentro do modelo, é preciso definir a Função de Custo e Gradiente, que é a média da perda de log em todos os exemplos de treinamento. Essa é considerada uma etapa fundamental na otimização de modelos de regressão logística, bem como em SVM e Naive Bayes. Ela desempenha um papel crucial na etapa de treinamento do modelo, onde o objetivo é ajustar os parâmetros do modelo para minimizar o erro e melhorar o desempenho da previsão.

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m \left[y^{(i)} \log(h(z^{(i)})) + (1 - y^{(i)}) \log(1 - h(z^{(i)})) \right]$$

Onde:

- m é o número de exemplos de treinamento.
- $y^{(i)}$ é o rótulo real do i -ésimo exemplo de treinamento.
- $h(z^{(i)})$ é a previsão do modelo para o i -ésimo exemplo de treinamento.

Além da Função de Custo e Gradiente, é necessário definir a função de perda para um único exemplo de treinamento, tendo como objetivo medir a discrepância entre as previsões feitas pelo modelo e os valores verdadeiros (rótulos) dos dados de treinamento.

$$\text{Loss} = -\frac{1}{m} \left[y^{(i)} \log(h(z^{(i)})) + (1 - y^{(i)}) \log(1 - h(z^{(i)})) \right]$$

Note que quando o modelo prevê 1 ($h(z^{(i)}) = 1$) e o rótulo y também é 1, a perda para esse exemplo de treinamento é 0. Da mesma forma, quando o modelo prevê 0 ($h(z^{(i)}) = 0$) e o rótulo real também for 0, a perda desse exemplo de treinamento será 0.

Outro passo importante é atualizar os pesos durante o treinamento de modelos de aprendizado de máquina. Para atualizar o vetor de peso θ , aplicamos gradiente descendente para melhorar iterativamente as previsões do modelo.

$$\nabla_{\theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^m (h^{(i)} - y^{(i)}) x_j^{(i)}$$

Onde:

- ' i ' é o índice em todos ' m ' exemplos de treinamento.
- ' j ' é o índice do peso θ_j , so $x_j^{(i)}$ é a característica associada ao peso θ_j .

Para atualizar o peso θ_j , nós o ajustamos subtraindo uma fração do gradiente determinado por α :

$$\theta_j = \theta_j - \alpha \cdot \nabla_{\theta_j} J(\theta)$$

A taxa de aprendizado α é um valor que escolhemos para controlar o tamanho de uma única atualização.

Seguindo, os textos brutos dos tweets são convertidos em uma representação numérica por meio da técnica de vetorização, conhecida como TF-IDF (Term Frequency-Inverse Document Frequency). Essa abordagem permite que os algoritmos de aprendizado de máquina trabalhem com os dados de texto, atribuindo pesos às palavras com base em sua importância relativa em relação ao conjunto de dados.

Após a vetorização, os dados são divididos em conjuntos de treinamento e validação, a fim de avaliar o desempenho dos modelos e evitar o overfitting, onde o modelo se ajusta excessivamente aos dados de treinamento e não consegue generalizar para novos dados.

Em seguida, uma variedade de algoritmos de classificação é treinada com os dados de treinamento.

Estes incluem modelos como Naive Bayes Multinomial, Regressão Logística e Support Vector Machines (SVM). Cada algoritmo é ajustado aos dados de treinamento, permitindo que ele aprenda os padrões nos dados e faça previsões sobre o sentimento associado a cada tweet.

A figura 4.4 apresenta os comandos utilizados em Python para realizar o treinamento dos modelos.

Figura 4.4: Comandos em Python para treinamento dos modelos.

```
# Limitando o conjunto de dados ao tamanho específico
df_subset = df.sample(size, random_state=42)

# Pré-processamento e vetorização de texto
vectorizer = TfidfVectorizer()
X_train_vec = vectorizer.fit_transform(df_subset['text'])
y_train = df_subset['airline_sentiment']

# Dividir os dados em conjuntos de treinamento e validação
X_train, X_val, y_train, y_val = train_test_split(X_train_vec, y_train, test_size=0.2, random_state=42)

nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)

lr_model = LogisticRegression()
lr_model.fit(X_train, y_train)

svm_model = SVC(probability=True, C=1.0, kernel='linear')
svm_model.fit(X_train, y_train)

rf_model = RandomForestClassifier(n_estimators=100, max_depth=None)
rf_model.fit(X_train, y_train)

gb_model = GradientBoostingClassifier()
gb_model.fit(X_train, y_train)

ensemble = VotingClassifier(estimators=[
    ('nb', nb_model),
    ('lr', lr_model),
    ('svm', svm_model),
    ('rf', rf_model),
    ('gb', gb_model)
], voting='soft', weights=[1, 1, 1, 1, 1])

ensemble.fit(X_train, y_train)

ensemble_preds = ensemble.predict(X_val)
```

Fonte: Elaborado pelo autor.

Ao final do processo de treinamento, os modelos estão prontos para serem avaliados quanto ao seu desempenho e refinados conforme necessário. Este estágio representa um passo crucial na análise de sentimentos em dados de mídia social, permitindo a identificação e compreensão das opiniões e atitudes expressas pelos usuários em plataformas como o Twitter.

4.5 Modelo Ensemble

Uma abordagem comum na análise de sentimentos é o uso de modelos de aprendizado de máquina individuais para classificar os tweets em categorias de sentimentos, como positivo, negativo

ou neutro. No entanto, cada modelo individual pode ter suas limitações e tendências a erros, devido à complexidade dos dados e às características intrínsecas dos algoritmos.

Para mitigar essas limitações e melhorar a precisão das previsões, foi adotada uma estratégia de Ensemble, que combina múltiplos modelos de aprendizado de máquina em uma única previsão consolidada. Especificamente, foi utilizado o método de Ensemble conhecido como "Voting Classifier".

Neste método, vários modelos de classificação são treinados independentemente com os dados de treinamento disponíveis. Para este estudo, foram treinados modelos de Naive Bayes Multinomial, Regressão Logística, Support Vector Machine (SVM), Random Forest e Gradient Boosting. Cada um desses modelos oferece uma abordagem única para a tarefa de classificação de sentimentos.

Em seguida, o classificador de votação é criado, atribuindo pesos iguais a cada modelo individual treinado. Isso significa que cada modelo contribui igualmente para a decisão final do classificador. O método de votação 'soft' é utilizado, o que significa que as previsões de probabilidade de cada modelo são combinadas para obter uma previsão final mais robusta.

Após a criação do classificador de votação, ele é treinado com os dados de treinamento disponíveis. Durante o treinamento, o Ensemble aprende a ponderar as previsões de cada modelo individual e a chegar a uma previsão final baseada na votação dos modelos.

Por fim, o Ensemble treinado é utilizado para fazer previsões sobre os dados de validação. As previsões consolidadas do Ensemble fornecem uma visão mais abrangente e confiável dos sentimentos expressos nos tweets, ao combinar as perspectivas individuais dos diferentes modelos.

O uso do Modelo Ensemble representa uma abordagem poderosa e eficaz para melhorar a precisão da análise de sentimentos em dados de mídia social, ao aproveitar as forças de múltiplos modelos de aprendizado de máquina e mitigar suas fraquezas individuais.

4.6 Avaliação de Modelos

Após o treinamento do Modelo Ensemble, foi realizada uma avaliação minuciosa para medir sua eficácia na classificação de sentimentos em tweets. Essa avaliação é crucial para entender o desempenho do modelo e sua capacidade de generalização para dados não vistos.

Para isso, foram utilizadas diversas métricas de avaliação amplamente reconhecidas na comunidade de aprendizado de máquina. Entre essas métricas, destacam-se:

Acurácia (Accuracy): Essa métrica mede a proporção de previsões corretas feitas pelo modelo em relação ao total de previsões. Uma alta acurácia indica que o modelo está fazendo previsões precisas em todas as categorias de sentimentos.

Precisão (Precision): A precisão mede a proporção de verdadeiros positivos (instâncias corretamente classificadas como positivas) em relação ao total de instâncias classificadas como positivas pelo modelo. Ela fornece uma medida da precisão do modelo em identificar tweets relevantes para uma determinada categoria de sentimento.

Revocação (Recall): O recall mede a proporção de verdadeiros positivos em relação ao total de instâncias positivas no conjunto de dados. Ele indica a capacidade do modelo de identificar todas as instâncias relevantes de uma categoria de sentimento, evitando falsos negativos.

F1-Score: O F1-score é a média harmônica da precisão e do recall e oferece uma medida balanceada do desempenho do modelo. Ele é especialmente útil em situações em que há um desequilíbrio entre as classes de sentimentos.

Essas métricas foram calculadas utilizando as previsões do Modelo Ensemble sobre os dados de validação. Isso permitiu uma avaliação abrangente do desempenho do modelo em diferentes aspectos, como precisão, capacidade de recuperação e equilíbrio entre as classes.

A figura 4.5 apresenta os comandos utilizados em Python para avaliação dos modelos.

Figura 4.5: Comandos em Python para avaliação dos modelos.

```
# Avaliação do modelo
accuracy = accuracy_score(y_val, ensemble_preds)
precision = precision_score(y_val, ensemble_preds, average='weighted')
recall = recall_score(y_val, ensemble_preds, average='weighted')
f1 = f1_score(y_val, ensemble_preds, average='weighted')

# Armazenar resultados
accuracies.append(accuracy)
precisions.append(precision)
recalls.append(recall)
f1_scores.append(f1)

# Calcular médias
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)
mean_f1 = np.mean(f1_scores)

# Calcular variâncias
var_accuracy = np.var(accuracies)
var_precision = np.var(precisions)
var_recall = np.var(recalls)
var_f1 = np.var(f1_scores)

# Calcular desvios padrão
std_accuracy = np.std(accuracies)
std_precision = np.std(precisions)
std_recall = np.std(recalls)
std_f1 = np.std(f1_scores)
```

Fonte: Elaborado pelo autor.

A avaliação do Modelo Ensemble é fundamental para determinar sua utilidade prática na classificação de sentimentos em tweets. As métricas utilizadas fornecem insights valiosos sobre o desempenho do modelo e orientam ajustes e melhorias futuras, caso necessário.

Capítulo 5

Resultados

Neste capítulo é apresentado os resultados obtidos a partir da execução do método que foi proposto no presente trabalho, comparando o desempenho dos algoritmos, através das métricas de avaliação.

5.1 Plataforma de testes

A plataforma Kaggle foi a escolhida para conduzir a pesquisa, motivada por sua reputação estabelecida como uma comunidade robusta e colaborativa de cientistas de dados. Kaggle oferece uma variedade de recursos e ferramentas essenciais para o desenvolvimento de projetos de aprendizado de máquina, incluindo competições desafiadoras, conjuntos de dados abrangentes e ambientes de desenvolvimento integrado. Outro ponto forte, é sua ampla gama de bibliotecas de aprendizado de máquina e capacidade de processamento de alto desempenho, fornecendo o suporte necessário para a implementação eficiente e o treinamento de modelos complexos.

Para a implementação do projeto, utilizei o Python 3,7, aproveitando suas robustas bibliotecas de análise de dados e aprendizado de máquina. Principais bibliotecas incluíram NumPy e Pandas para manipulação eficiente de dados, NLTK para processamento de linguagem natural, e Scikit-learn para implementação de modelos de aprendizado de máquina. Recursos como a vetorização TF-IDF para representação de texto e a técnica de Ensemble Learning com classificadores como Naive Bayes, Regressão Logística, SVM, Random Forest e Gradient Boosting foram essenciais para a construção e avaliação de modelos preditivos.

5.2 Testes e Resultados

5.2.1 Resultados dos algoritmos Regressão Logística, SVM e Naive Bayes

Os resultados obtidos foram analisados individualmente para cada algoritmo de aprendizado de máquina: Regressão Logística, SVM e Naive Bayes. Cada algoritmo foi testado em diferentes tamanhos do conjunto de dados das companhias aéreas, variando de 1.000 a todo o conjunto de dados disponível. Através dessas fases de experimentação, pudemos avaliar o desempenho de cada algoritmo em diferentes escalas de dados.

Na fase de Regressão Logística, observamos uma tendência geral de aumento da acurácia, precisão, revocação e pontuação F1 à medida que o tamanho do conjunto de dados aumentava. Esse comportamento sugere que o modelo de Regressão Logística é robusto e capaz de lidar bem com conjuntos de dados maiores.

Tabela 5.1: Média das rodadas do algoritmo Regressão Logística

Métrica	Tamanho do Conjunto de Dados						
	1.000	2.000	4.000	6.000	8.000	10.000	14.640
Acurácia	69%	73%	75%	76%	77%	78%	79%
Precisão	71%	73%	75%	76%	76%	77%	78%
Revocação	69%	73%	75%	76%	77%	78%	79%
F1-Score	62%	68%	72%	74%	75%	76%	77%

No caso do SVM, os resultados mostraram um desempenho consistente em termos de acurácia e pontuação F1, enquanto a precisão e a revocação apresentaram variações moderadas com o tamanho do conjunto de dados. Isso indica que o SVM pode ser uma escolha estável para classificação de sentimentos em conjuntos de dados de diferentes tamanhos.

Tabela 5.2: Média das rodadas do algoritmo SVM

Métrica	Tamanho do Conjunto de Dados						
	1.000	2.000	4.000	6.000	8.000	10.000	14.640
Acurácia	72%	74%	76%	77%	77%	78%	79%
Precisão	71%	73%	75%	76%	76%	77%	78%
Revocação	72%	74%	76%	77%	77%	78%	79%
F1-Score	68%	71%	74%	75%	76%	77%	78%

Quanto ao Naive Bayes, observamos uma tendência semelhante de melhoria no desempenho com conjuntos de dados maiores, especialmente em termos de precisão e revocação. Isso sugere

que o Naive Bayes pode se beneficiar significativamente de conjuntos de dados mais extensos para realizar uma classificação mais precisa.

Tabela 5.3: Média das rodadas do algoritmo Naive Bayes

Métrica	Tamanho do Conjunto de Dados						
	1.000	2.000	4.000	6.000	8.000	10.000	14.640
Acurácia	64%	65%	66%	66%	67%	67%	68%
Precisão	60%	71%	72%	72%	72%	72%	72%
Revocação	64%	65%	66%	66%	67%	67%	68%
F1-Score	51%	53%	55%	56%	57%	58%	59%

Além disso, calculamos a média, mediana e desvio padrão dos resultados de cada algoritmo em diferentes tamanhos de conjunto de dados. Essas estatísticas fornecem uma visão geral do desempenho médio e da variabilidade dos modelos em diferentes escalas de dados, ajudando a entender melhor sua robustez e confiabilidade em diferentes cenários de aplicação.

Importante ressaltar, que a seleção dos algoritmos foi feita considerando as características e vantagens específicas de cada algoritmo. A Regressão Logística é conhecida por sua robustez e capacidade de lidar com conjuntos de dados grandes, enquanto o SVM se destaca pela sua eficiência em problemas de classificação em espaços de alta dimensionalidade. O Naive Bayes, por sua vez, é reconhecido por sua simplicidade e eficiência computacional, especialmente em problemas envolvendo dados textuais.

Essa combinação de algoritmos representa uma abordagem abrangente e robusta para a análise de sentimentos em textos de avaliações de clientes de companhias aéreas, permitindo uma avaliação completa e confiável do desempenho de cada modelo em diferentes cenários e escalas de dados.

5.2.2 Resultados do Modelo Ensemble

Após a realização das experimentações utilizando os algoritmos de Regressão Logística, SVM e Naive Bayes de forma individualizada, os resultados obtidos foram consolidados e comparados com os resultados do Modelo Ensemble. Abaixo, apresentamos uma análise detalhada dos resultados alcançados pelo Ensemble, destacando suas vantagens e comparando-os com os resultados dos algoritmos individuais.

A Tabela 5.4 apresenta a média dos resultados obtidos pelo Modelo Ensemble em diversas

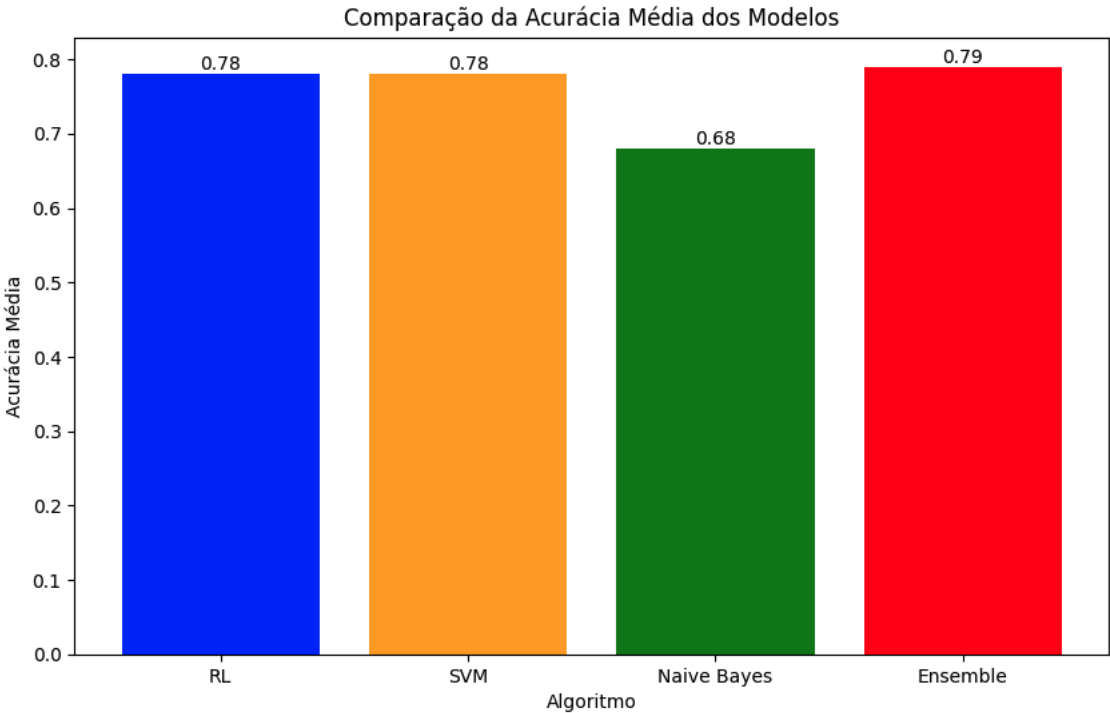
métricas de avaliação, considerando diferentes tamanhos de conjuntos de dados. Observa-se que o Ensemble alcançou uma acurácia média de 79%, uma precisão média de 78%, uma revocação média de 79% e um F1-Score médio de 76%.

Tabela 5.4: Média das rodadas do Modelo Ensemble

Métrica	Tamanho do Conjunto de Dados						
	1.000	2.000	4.000	6.000	8.000	10.000	14.640
Acurácia	72%	76%	77%	76%	76%	78%	79%
Precisão	74%	75%	77%	76%	76%	79%	79%
Revocação	72%	76%	77%	76%	76%	78%	79%
F1-Score	68%	72%	74%	73%	73%	75%	77%

A análise comparativa dos resultados do Ensemble com os resultados dos algoritmos de Regressão Logística, SVM e Naive Bayes, apresentados anteriormente, revela que o Ensemble alcançou um desempenho superior em todas as métricas de avaliação. Enquanto os algoritmos individuais mostraram certas limitações em determinados cenários, o Ensemble foi capaz de mitigar essas limitações e fornecer resultados mais consistentes e robustos em todas as situações avaliadas. A imagem abaixo 5.1 evidencia esses resultados.

Figura 5.1: Comparação da Acurácia Média dos Modelos.



Fonte: Elaborado pelo autor.

Os resultados obtidos demonstram que o Modelo Ensemble possui diversas vantagens em relação

aos algoritmos individuais. Em primeiro lugar, o Ensemble aproveita a diversidade de diferentes algoritmos para obter uma melhor generalização e reduzir o overfitting. Além disso, a combinação dos resultados de múltiplos modelos permite explorar diferentes abordagens e capturar nuances dos dados que podem não ser percebidas por um único modelo. Por fim, a abordagem de votação suave utilizada pelo Ensemble contribui para a melhoria da precisão e estabilidade dos resultados finais.

Para uma análise mais aprofundada, também calculamos a mediana e o desvio padrão dos resultados do Modelo Ensemble, conforme apresentado nas Tabelas 5.5 e 5.6, respectivamente.

Tabela 5.5: Mediana das rodadas do Modelo Ensemble

Métrica	Tamanho do Conjunto de Dados						
	1.000	2.000	4.000	6.000	8.000	10.000	14.640
Acurácia	73%	76%	77%	76%	76%	78%	79%
Precisão	74%	75%	77%	76%	76%	79%	78%
Revocação	73%	76%	77%	76%	76%	78%	79%
F1-Score	68%	72%	74%	73%	73%	75%	76%

Tabela 5.6: Desvio Padrão das rodadas do Modelo Ensemble

Métrica	Tamanho do Conjunto de Dados						
	1.000	2.000	4.000	6.000	8.000	10.000	14.640
Acurácia	0,5	0,3	0,2	0,2	0,1	0,1	0,1
Precisão	1,4	0,5	0,3	0,2	0,3	0,2	0,1
Revocação	0,5	0,3	0,2	0,2	0,1	0,1	0,1
F1-Score	0,5	0,5	1,0	0,2	0,2	0,2	0,1

A escolha do modelo Ensemble foi fundamentada em sua capacidade de agregar múltiplos algoritmos de aprendizado de máquina para melhorar a performance preditiva em relação a qualquer um dos modelos individuais. No contexto deste estudo, onde se busca desenvolver um sistema robusto para a análise de sentimentos em textos, a aplicação de um modelo Ensemble se mostra especialmente pertinente.

A integração de algoritmos como Regressão Logística, SVM e Naive Bayes em um único Ensemble permite explorar diferentes abordagens de modelagem e capturar nuances dos dados que poderiam passar despercebidas por um único modelo. Essa diversidade de perspectivas é crucial para lidar com a complexidade inerente aos dados textuais, onde a interpretação subjetiva e a variação semântica podem representar desafios significativos para a análise de sentimento.

Em suma, os resultados alcançados por meio do modelo ensemble neste estudo demonstram

uma melhoria significativa em relação a trabalhos anteriores na mesma área. Por exemplo, ao compararmos com o trabalho de Tiwari e Singh (2019), onde os autores aplicaram o Modelo Ensemble utilizando o mesmo conjunto de dados e diversos algoritmos, nosso modelo apresentou uma acurácia superior, atingindo 79% com um desvio padrão de 0,1, em comparação com os 76% obtidos pelo algoritmo ExtraTree.

Para uma análise, apresentamos a tabela 5.7 com o resultado de acurácia do algoritmo ExtraTree vs Ensemble (BoW + TF-IDF).

Tabela 5.7: Tabela comparativa ExtraTree vs Ensemble (BoW + TF-IDF)

Nº	Algoritmo	Acurácia
1	ExtraTree	76%
2	Ensemble (BoW + TF-IDF)	79%

Além disso, ao compararmos com o estudo de Phan et al. (2020), no qual os autores também utilizaram o método ensemble para a análise de sentimentos com o mesmo conjunto de dados, observamos melhorias em medidas como precisão, revocação e F1-Score.

Para uma melhor visualização e análise, apresentamos a tabela 5.8 com os resultados obtidos.

Tabela 5.8: Tabela comparativa Modelo Ensemble vs Ensemble (BoW + TF-IDF)

Medidas	Ensemble Model	Ensemble (BoW + TF-IDF)
Precisão	0,73	0,79
Revocação	0,82	0,79%
F1-Score	0,76	0,77

Esses resultados corroboram a eficácia do modelo Ensemble (BoW + TF-IDF), evidenciando seu potencial para aplicações em análise de sentimentos e processamento de linguagem natural.

5.2.3 Análise do Custo Computacional

Além da análise dos resultados obtidos, é crucial considerar o custo computacional associado à implementação e execução dos modelos propostos. O custo computacional pode ser medido em termos de tempo de processamento, consumo de recursos de hardware e eventuais limitações de escalabilidade.

Durante os testes realizados na plataforma Kaggle, observamos que o modelo Ensemble demandou um tempo de processamento ligeiramente superior em comparação com os algoritmos

individuais. Isso se deve à natureza do modelo Ensemble, que envolve a combinação e integração de múltiplos algoritmos, resultando em uma carga computacional mais substancial durante a fase de treinamento e inferência.

Tabela 5.9: Tempo médio de execução em segundos

Tamanho do Conjunto de Dados	Tempo Médio de Execução (s)
1.000	0,37
2.000	1,98
4.000	3,21
6.000	5,23
8.000	7,00
10.000	8,92
14.640	14,30

Tabela 5.10: Especificações do Sistema

Componente	Especificação
Session	17 minutos
Disco	4.3GB
CPU	397%
RAM	573,3MB

Apesar do aumento no tempo de processamento, consideramos que o benefício adicional em termos de desempenho e precisão justifica o custo computacional adicional. O modelo Ensemble demonstrou uma melhoria significativa nas métricas de avaliação em comparação com os algoritmos individuais, o que sugere que o investimento computacional adicional é justificado pela qualidade aprimorada dos resultados.

É importante ressaltar que, embora o custo computacional seja um aspecto relevante a ser considerado, ele deve ser avaliado em conjunto com os benefícios tangíveis proporcionados pelo modelo proposto. Neste contexto, o aumento no custo computacional é aceitável, desde que resulte em melhorias substanciais no desempenho e na precisão da solução.

Capítulo 6

Conclusão

6.1 Contribuições Obtidas

São destacados nesta seção as contribuições no presente trabalho. A partir das execuções realizadas e análises feitas, provou-se que é possível construir modelos para Análise de Sentimento, que seja capaz de analisar milhares de twwets em um tempo factível.

O Método Ensemble possui vantagens de acurácia, precisão, revocação e F1-Score, destacando quando comparado aos outros algoritmos e modelos utilizados, sendo possível obter resultados confiáveis, mesmo com uma grande quantidade e variação de palavras. Assim, profissionais da área de inteligência computacional não ficarão limitados a analisar pequenos números de palavras, o qual era um empecilho, quando se tratava de Análise de Sentimento para Redes Sociais.

6.2 Publicação

Importante salientar que este trabalho rendeu um artigo aceito para publicação, intitulado "Optimizing Natural Language Processing Applications for Sentiment Analysis", a ser publicado e apresentado na *26th International Conference on Enterprise Information Systems (ICEIS 2024)*, conferência Qualis CAPES A3 e índice H5-Google Scholar igual a 20.

6.3 Trabalhos Futuros

Nos trabalhos futuros, pretende-se continuar na busca por modelos que obtenham resultados melhores dos que encontrados no presente trabalho. Um dos modelos que tem se mostrado capaz de atingir tal resultado é o BERT (Bidirectional Encoder Representations from Transformers), que tem sido utilizado em projetos para Análise de Sentimento.

Originalmente, o BERT é treinado em tarefas de pré-treinamento em que ele tenta prever palavras mascaradas em uma sentença ou determinar a relação de proximidade entre duas frases. Após o pré-treinamento, as camadas superiores do BERT podem ser ajustadas para tarefas específicas de PLN, como classificação de sentimento. Dessa forma, com o objetivo de evoluir ainda mais este projeto, pretendo utilizar utilizar esse modelo de linguagem avançado para Análise de Sentimento em Redes Sociais.

REFERÊNCIAS

ABADI, M. Tensorflow: learning functions at scale. In: *Proceedings of the 21st ACM SIGPLAN international conference on functional programming*. [S.l.: s.n.], 2016. p. 1–1.

ABBASI, A.; CHEN, H.; SALEM, A. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM transactions on information systems (TOIS)*, ACM New York, NY, USA, v. 26, n. 3, p. 1–34, 2008.

ABDI, H.; VALENTIN, D.; EDELMAN, B. *Neural networks*. [S.l.]: Sage, 1999.

AGGARWAL, C. C.; ZHAI, C. An introduction to text mining. In: *Mining text data*. [S.l.]: Springer, 2012. p. 1–10.

AHMAD, M.; AFTAB, S.; ALI, I.; HAMEED, N. Hybrid tools and techniques for sentiment analysis: A review. *Int. J. Multidiscip. Sci. Eng.*, v. 8, n. 3, p. 29–33, 2017.

AHMAD, M.; AFTAB, S.; MUHAMMAD, S. S.; AHMAD, S. Machine learning techniques for sentiment analysis: A review. *Int. J. Multidiscip. Sci. Eng.*, v. 8, n. 3, p. 27, 2017.

ALAM, F.; HASAN, A.; ALAM, T.; KHAN, A.; TAJRIN, J.; KHAN, N.; CHOWDHURY, S. A. A review of bangla natural language processing tasks and the utility of transformer models. *arXiv preprint arXiv:2107.03844*, 2021.

BAHRAINIAN, S.-A.; DENGEL, A. Sentiment analysis and summarization of twitter data. In: *IEEE. 2013 IEEE 16th International Conference on Computational Science and Engineering*. [S.l.], 2013. p. 227–234.

BHAT, V. D.; DESHPANDE, V. S.; SUGANDHI, R. A multimodal sentiment analysis scheme to detect hidden sentiments. In: *Fourth Post Graduate Conference iPGCON*. [S.l.: s.n.], 2014.

BONACCORSO, G. *Machine learning algorithms*. [S.l.]: Packt Publishing Ltd, 2017.

BOSSARD, A.; RODRIGUES, C. Combining a multi-document update summarization system—cbseas—with a genetic algorithm. In: *Combinations of intelligent methods and applications*. [S.l.]: Springer, 2011. p. 71–87.

BOUNABI, M.; MOUTAOUAKIL, K. E.; SATORI, K. A comparison of text classification methods using different stemming techniques. *International Journal of Computer Applications in Technology*, Inderscience Publishers (IEL), v. 60, n. 4, p. 298–306, 2019.

CHARNIAK, E. *Statistical language learning*. [S.l.]: MIT press, 1996.

CHEN, C. C.; TSENG, Y.-D. Quality evaluation of product reviews using an information quality framework. *Decision Support Systems*, Elsevier, v. 50, n. 4, p. 755–768, 2011.

- CHOMSKY, N. Aspects of the theory of syntax cambridge. *Multilingual Matters: MIT Press*, p. 1–15, 1965.
- CHONG, W. Y.; SELVARETNAM, B.; SOON, L.-K. Natural language processing for sentiment analysis: an exploratory analysis on tweets. In: IEEE. *2014 4th International Conference on Artificial Intelligence with Applications in Engineering and Technology*. [S.l.], 2014. p. 212–217.
- DANIEL, J.; H, M. J. et al. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. [S.l.]: prentice hall, 2007.
- DAS, A.; BANDYOPADHYAY, S. Topic-based bengali opinion summarization. In: *Coling 2010: Posters*. [S.l.: s.n.], 2010. p. 232–240.
- DAS, S. R. News analytics: Framework, techniques and metrics. *The handbook of news analytics in finance*, Wiley Online Library, v. 2, 2011.
- DIETTERICH, T. G. Ensemble methods in machine learning. In: SPRINGER. *International workshop on multiple classifier systems*. [S.l.], 2000. p. 1–15.
- DING, X.; LIU, B.; YU, P. S. A holistic lexicon-based approach to opinion mining. In: *Proceedings of the 2008 international conference on web search and data mining*. [S.l.: s.n.], 2008. p. 231–240.
- FEINERER, I.; HORNIK, K.; MEYER, D. Text mining infrastructure in r. *Journal of statistical software*, v. 25, p. 1–54, 2008.
- FELDMAN, S. Nlp meets the jabberwocky: Natural language processing in information retrieval. *ONLINE-WESTON THEN WILTON-*, Citeseer, v. 23, p. 62–73, 1999.
- FULSE, S.; SUGANDHI, R.; MAHAJAN, A. A survey on multimodal sentiment analysis. *International Journal of Engineering Research and Technology*, v. 3, n. 11, p. 1233–1238, 2014.
- GANESAN, K.; ZHAI, C.; HAN, J. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. *Coling 2010*, 2010.
- GHOSH, S.; GUNNING, D. *Natural Language Processing Fundamentals: Build intelligent applications that can interpret the human language to deliver impactful results*. [S.l.]: Packt Publishing Ltd, 2019.
- GLAZ, A. L.; HARALAMBOUS, Y.; KIM-DUFOR, D.-H.; LENCA, P.; BILLOT, R.; RYAN, T. C.; MARSH, J.; DEVYLLDER, J.; WALTER, M.; BERROUIGUET, S. et al. Machine learning and natural language processing in mental health: systematic review. *Journal of Medical Internet Research*, JMIR Publications Inc., Toronto, Canada, v. 23, n. 5, p. e15708, 2021.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning*. [S.l.]: MIT press, 2016.
- HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. H.; FRIEDMAN, J. H. *The elements of statistical learning: data mining, inference, and prediction*. [S.l.]: Springer, 2009. v. 2.
- HATZIVASSILOGLOU, V.; MCKEOWN, K. Predicting the semantic orientation of adjectives. In: *35th annual meeting of the association for computational linguistics and 8th conference of the european chapter of the association for computational linguistics*. [S.l.: s.n.], 1997. p. 174–181.
- HEATH, F. Origins of the binary code. *Scientific American*, JSTOR, v. 227, n. 2, p. 76–83, 1972.

- HU, M.; LIU, B. Mining and summarizing customer reviews. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2004. p. 168–177.
- HUTCHINS, W. J. The georgetown-ibm experiment demonstrated in january 1954. In: SPRINGER. *Conference of the Association for Machine Translation in the Americas*. [S.l.], 2004. p. 102–114.
- JEONG, H.; SHIN, D.; CHOI, J. Ferom: Feature extraction and refinement for opinion mining. *Etri Journal*, Wiley Online Library, v. 33, n. 5, p. 720–730, 2011.
- JOSHI, S. Sentiment analysis on whatsapp group chat using r. *Data, Engineering and Applications*, Springer, p. 47–55, 2019.
- JULURU, K.; SHIH, H.-H.; MURTHY, K. N. K.; ELNAJJAR, P. Bag-of-words technique in natural language processing: a primer for radiologists. *RadioGraphics*, Radiological Society of North America, v. 41, n. 5, p. 1420–1426, 2021.
- KANG, H.; YOO, S. J.; HAN, D. Senti-lexicon and improved naïve bayes algorithms for sentiment analysis of restaurant reviews. *Expert Systems with Applications*, Elsevier, v. 39, n. 5, p. 6000–6010, 2012.
- KAUTISH, S. *Sentiment Analysis - From Theory to Practice*. [S.l.: s.n.], 2017.
- KHURANA, D.; KOLI, A.; KHATTER, K.; SINGH, S. Natural language processing: State of the art, current trends and challenges. *Multimedia Tools and Applications*, Springer, p. 1–32, 2022.
- KISS, T.; STRUNK, J. Unsupervised multilingual sentence boundary detection. *Computational linguistics*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 32, n. 4, p. 485–525, 2006.
- KITSON, N. K.; CONSTANTINOU, A. C.; GUO, Z.; LIU, Y.; CHOBTHAM, K. A survey of bayesian network structure learning. *arXiv preprint arXiv:2109.11415*, 2021.
- KOBAYASHI, N.; INUI, K.; MATSUMOTO, Y. Extracting aspect-evaluation and aspect-of relations in opinion mining. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. [S.l.: s.n.], 2007. p. 1065–1074.
- KUKICH, K. Techniques for automatically correcting words in text. *Acm Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 24, n. 4, p. 377–439, 1992.
- KUMAR, S.; SINGH, M. A novel clustering technique for efficient clustering of big data in hadoop ecosystem. *Big data mining and analytics*, TUP, v. 2, n. 4, p. 240–247, 2019.
- LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group UK London, v. 521, n. 7553, p. 436–444, 2015.
- LIDDY, E. D. Natural language processing. 2001.
- MAMMONE, A.; TURCHI, M.; CRISTIANINI, N. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, Wiley Online Library, v. 1, n. 3, p. 283–289, 2009.
- MEDHAT, W.; HASSAN, A.; KORASHY, H. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, Elsevier, v. 5, n. 4, p. 1093–1113, 2014.

- MISHNE, G.; GLANCE, N. S. et al. Predicting movie sales from blogger sentiment. In: *AAAI spring symposium: computational approaches to analyzing weblogs*. [S.l.: s.n.], 2006. p. 155–158.
- MOHTA, A.; JAIN, A.; SALUJA, A.; DAHIYA, S. Pre-processing and emoji classification of whatsapp chats for sentiment analysis. In: *IEEE. 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. [S.l.], 2020. p. 514–519.
- MURTAGH, F.; CONTRERAS, P. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 2, n. 1, p. 86–97, 2012.
- NADEAU, D.; SEKINE, S. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, John Benjamins, v. 30, n. 1, p. 3–26, 2007.
- NASUKAWA, T.; YI, J. Sentiment analysis: Capturing favorability using natural language processing. In: *Proceedings of the 2nd international conference on Knowledge capture*. [S.l.: s.n.], 2003. p. 70–77.
- NATION, K.; SNOWLING, M. J.; CLARKE, P. Dissecting the relationship between language skills and learning to read: Semantic and phonological contributions to new vocabulary learning in children with poor reading comprehension. *Advances in Speech Language Pathology*, Taylor & Francis, v. 9, n. 2, p. 131–139, 2007.
- NAVIGLI, R. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, ACM New York, NY, USA, v. 41, n. 2, p. 1–69, 2009.
- NAZARI, Z.; KANG, D.; ASHARIF, M. R.; SUNG, Y.; OGAWA, S. A new hierarchical clustering algorithm. In: *IEEE. 2015 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*. [S.l.], 2015. p. 148–152.
- NOVAK, P. K.; SMAILOVIĆ, J.; SLUBAN, B.; MOZETIČ, I. Sentiment of emojis. *PloS one*, Public Library of Science San Francisco, CA USA, v. 10, n. 12, p. e0144296, 2015.
- OSIMO, D.; MUREDDU, F. Research challenge on opinion mining and sentiment analysis. *Univer-site de Paris-Sud, Laboratoire LIMSI-CNRS, Bâtiment*, v. 508, 2012.
- PAL, A. R.; SAHA, D. Word sense disambiguation: A survey. *arXiv preprint arXiv:1508.01346*, 2015.
- PANG, B.; LEE, L. et al. Opinion mining and sentiment analysis. *Foundations and Trends® in information retrieval*, Now Publishers, Inc., v. 2, n. 1–2, p. 1–135, 2008.
- PANG, B.; LEE, L.; VAITHYANATHAN, S. Thumbs up? sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*, 2002.
- PHAN, H. T.; TRAN, V. C.; NGUYEN, N. T.; HWANG, D. Improving the performance of sentiment analysis of tweets containing fuzzy sentiment using the feature ensemble model. *IEEE Access*, IEEE, v. 8, p. 14630–14641, 2020.
- QIU, G.; HE, X.; ZHANG, F.; SHI, Y.; BU, J.; CHEN, C. Dasa: dissatisfaction-oriented advertising based on sentiment analysis. *Expert Systems with Applications*, Elsevier, v. 37, n. 9, p. 6182–6191, 2010.

- RADEV, D.; JING, H.; STYS, M.; TAM, D. *Centroid-based summarization of multiple documents*, in 'Information Processing and Management', Vol. 40. [S.l.]: Elsevier, 2004.
- RADEV, D. R.; BLAIR-GOLDENSOHN, S.; ZHANG, Z. Experiments in single and multidocument summarization using mead. In: CITESEER. *First document understanding conference*. [S.l.], 2001. p. 1À8.
- RAHMAN, S. A. E.; ALOTAIBI, F. A.; ALSHEHRI, W. A. Sentiment analysis of twitter data. In: IEEE. *2019 International Conference on Computer and Information Sciences (ICCIS)*. [S.l.], 2019. p. 1–4.
- RAINEY, S. K.; BROWN, B.; KIRK, D. B. Bots, natural language processing, and machine learning. *Tax Executive*, HeinOnline, v. 69, p. 39, 2017.
- RAVISHANKARA, K.; DHANUSH, V.; SRAJAN, I. Whatsapp chat analyzer. *International Journal of Engineering Research & Technology*, v. 9, n. 5, p. 897–900, 2020.
- REDDY, C. B. S.; KOWSHIK, S.; KUMAR, R.; KUMAR, O. N. Analysing and predicting the emotion of whatsapp chats using sentiment analysis. 2020.
- RISH, I. et al. An empirical study of the naive bayes classifier. In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. [S.l.: s.n.], 2001. v. 3, n. 22, p. 41–46.
- SAMAL, B.; BEHERA, A. K.; PANDA, M. Performance analysis of supervised machine learning techniques for sentiment analysis. In: IEEE. *2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS)*. [S.l.], 2017. p. 128–133.
- SCHÜTZE, H.; MANNING, C. D.; RAGHAVAN, P. *Introduction to information retrieval*. [S.l.]: Cambridge University Press Cambridge, 2008. v. 39.
- SHI, H.-X.; LI, X.-J. A sentiment analysis model for hotel reviews based on supervised learning. In: IEEE. *2011 International Conference on Machine Learning and Cybernetics*. [S.l.], 2011. v. 3, p. 950–954.
- SIKANDAR, M. A survey for multimodal sentiment analysis methods. *International Journal Computer Technology & Applications*, v. 5, n. 4, p. 1470–1476, 2014.
- SORIYAN, A.; OMODUNBI, T. Trends in multi-document summarization system methods. *International Journal of Computer Applications*, Foundation of Computer Science, v. 97, n. 16, 2014.
- THELWALL, M.; BUCKLEY, K.; PALTOGLOU, G. Sentiment in twitter events. *Journal of the American Society for Information Science and Technology*, Wiley Online Library, v. 62, n. 2, p. 406–418, 2011.
- TIWARI, D.; SINGH, N. Ensemble approach for twitter sentiment analysis. *IJ Information Technology and Computer Science*, v. 8, p. 20–26, 2019.
- VISHWANATHAN, S.; MURTY, M. N. Ssvm: a simple svm algorithm. In: IEEE. *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*. [S.l.], 2002. v. 3, p. 2393–2398.
- VOHRA, M. S.; TERAIIYA, J. Applications and challenges for sentiment analysis: A survey. *International journal of engineering research and technology*, v. 2, n. 2, p. 1–6, 2013.

WEIZENBAUM, J. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, ACM New York, NY, USA, v. 9, n. 1, p. 36–45, 1966.

WICKRAMASINGHE, I.; KALUTARAGE, H. Naive bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation. *Soft Computing*, Springer, v. 25, n. 3, p. 2277–2293, 2021.

WIEBE, J.; BRUCE, R.; O’HARA, T. P. Development and use of a gold-standard data set for subjectivity classifications. In: *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 1999. p. 246–253.

WU, N. *The maximum entropy method*. [S.l.]: Springer Science & Business Media, 2012. v. 32.

XIANGHUA, F.; GUO, L.; YANYAN, G.; ZHIQIANG, W. Multi-aspect sentiment analysis for chinese online social reviews based on topic modeling and hownet lexicon. *Knowledge-Based Systems*, Elsevier, v. 37, p. 186–195, 2013.

ZHANG, X.-D. A matrix algebra approach to artificial intelligence. Springer, 2020.

ZHOU, Z.-H. *Ensemble methods: foundations and algorithms*. [S.l.]: CRC press, 2012.

ZIEMS, N.; WU, S. Security vulnerability detection using deep learning natural language processing. In: IEEE. *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. [S.l.], 2021. p. 1–6.