



**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
IFCE CAMPUS ARACATI
COORDENADORIA DE CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

VINICIUS NUNES BARBOSA

SIRENE: Solução Inteligente para Detecção de Notícias Falsas

**ARACATI-CE
2020**

VINICIUS NUNES BARBOSA

SIRENE: SOLUÇÃO INTELIGENTE PARA DETECÇÃO DE NOTÍCIAS FALSAS

Trabalho de Conclusão de Curso (TCC) apresentado ao curso de Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Aracati, como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Reinaldo Bezerra Braga

Aracati-CE
2020

Dados Internacionais de Catalogação na Publicação

Instituto Federal do Ceará - IFCE

Sistema de Bibliotecas - SIBI

Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

N972s Nunes Barbosa, Vinícius.

SIRENE: Solução Inteligente para Detecção de Notícias Falsas / Vinícius Nunes Barbosa.
- 2020.

60 f. : il. color.

Trabalho de Conclusão de Curso (graduação) - Instituto Federal do Ceará, Bacharelado
em Ciência da Computação, Campus Aracati, 2020.

Orientação: Prof. Dr. Reinaldo Bezerra Braga .

1. Fake News. 2. Aprendizado de Máquina. 3. Sistema Web. I. Título.

VINICIUS NUNES BARBOSA

SIRENE: SOLUÇÃO INTELIGENTE PARA DETECÇÃO DE NOTÍCIAS FALSAS

Trabalho de Conclusão de Curso (TCC) apresentado ao curso de Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Aracati, como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Aprovada em 17/02/2020

BANCA EXAMINADORA

Prof. Dr. Reinaldo Bezerra Braga (Orientador)
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE

Prof. Dr. Carina Teixeira de Oliveira
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE

Prof. Me. Silas Santiago Lopes Pereira
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE

DEDICATÓRIA

A todos os meus professores.

A todos os alunos e pesquisadores que possam usufruir do meu trabalho.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, por abençoar meu caminho, nos momentos mais difíceis é a ele a quem recorro.

Agradeço à minha família, que para mim, foi a base para que eu pudesse chegar aonde estou hoje. Pois mesmo com problemas e dificuldades eles sempre me ajudaram e incentivaram.

Ao professor Mauro Oliveira, que me proporcionou uma bolsa de pesquisa e extensão desde o início do curso, o que me motivou até o fim deste.

Ao meu professor e orientador Reinaldo, por todo o auxílio, paciência e orientação ao longo de todo o trabalho. Além de ter sido fundamental em minha formação.

A todos do LAR (Laboratório de Redes de Computadores e Sistemas), que auxiliaram em minha formação, pois é um ambiente de muita inspiração.

Aos meus amigos, que me acompanharam no decorrer de todo o curso e que foram uma grande fonte de aprendizado ao longo dos anos e que sempre me ajudaram tanto no meu desenvolvimento acadêmico quanto como pessoa e que levarei para a vida. Em especial Ruan Gondim, Renato Alves, Leonardo Freitas, Igor Galdino, Rômulo Henrique, Roberta Alencar, Edgar dos Santos e Guilherme Oliveira.

Aos meus professores, que além de me ensinarem, foram muito compreensivos e pacientes.

RESUMO

Com o avanço das tecnologias e a popularização das redes sociais, o volume de dados na web têm aumentado. Assim, dada a abrangência da Internet, o acesso e compartilhamento de informações têm ficado cada vez mais fácil e rápido, principalmente com a ampla utilização dos dispositivos móveis. Nesse contexto, o aumento da proliferação de notícias falsas na Internet tem impactado significativamente na qualidade e veracidade das informações recebidas pela sociedade. O uso mal-intencionado de informações pode comprometer a democracia, manipulando a opinião das pessoas expostas a esse tipo de notícia. Além disso, existem poucos mecanismos facilitadores que classifiquem e ajudem o cidadão a saber se determinada notícia propagada é verídica ou não. Essa problemática tem impulsionado novas direções de pesquisa na tentativa de classificar e identificar essas notícias. Neste contexto, este trabalho apresenta o SIRENE, uma solução inteligente para detecção de notícias falsas em português. Foram extraídas 4.742 notícias dos portais G1 e Sensacionalista. Essas notícias, foram unidas com o conjunto de dados do *Fake.br* que possui 7.200 notícias tanto verdadeiras quanto falsas. Foram utilizadas técnicas de pré-processamento no conjunto de dados para a análise dos padrões dessas notícias, bem como para reduzir os ruídos e eliminar informações nulas. Os algoritmos utilizados e comparados para a realização dessa proposta foram o *Logistic Regression* (LR), *Stochastic Gradient Descent* (SGD), *Support Vector Machine* (SVM) e o *Multilayer Perceptron* (MLP). Os resultados obtidos mostram que os modelos gerados pelos quatro algoritmos obtiveram uma acurácia superior a 90%. Assim, para a escolha do melhor algoritmo foram utilizadas outras métricas como a precisão, *recall* e a medida-f para cada um dos modelos. O algoritmo SVM obteve o melhor desempenho, com 96.39% de acurácia. Além dos resultados analíticos apresentados, este trabalho traz como contribuições a disponibilização de uma base de dados contendo notícias em português (*fake news* e verdadeiras) e uma aplicação *web* do SIRENE para apresentar ao usuário a probabilidade de uma notícia informada ser verdadeira ou falsa, a partir do texto da notícia ou através de um link.

Palavras-chaves: *Fake News*. Aprendizado de Máquina. Sistema *Web*.

ABSTRACT

With the advancement of technologies and the popularization of social networks, the volume of data on the web has increased. Thus, given the scope of the Internet, accessing and sharing information has become increasingly easier and faster, especially with the widespread use of mobile devices. In this context, the increase in the proliferation of fake news on the Internet has significantly affected the quality and veracity of the information received by society. Malicious use of information can compromise democracy by manipulating the opinions of people exposed to this type of news. In addition, there are few facilitating mechanisms that classifies and helps the citizen to know if a certain news spread is true or not. This problem has driven new research directions in an attempt to classify and identify this news. In this context, this work presents SIRENE, an intelligent solution for detecting fake news in Portuguese. Was extracted 4.742 news from the portals G1 and Sensacionalista, and this news were combined with the Fake.br dataset, which has 7.200 news both true and false. Pre-processing techniques are used in the dataset to analyze the patterns of this news, as well as to reduce noise and eliminate null information. The algorithms used and compared for the realization of this proposal were Logistic Regression (LR), Stochastic Gradient Descent (SGD), Support Vector Machine (SVM) and Multilayer Perceptron (MLP). The results obtained show that the models generated by the four algorithms obtained an accuracy greater than 90%. Thus, to choose the best algorithm, other metrics were used, such as precision, recall and the f-measure for each model. The SVM algorithm achieved the best performance, with 96.39% accuracy. In addition to the analytical results presented, this work brings as contributions the availability of a database containing news in Portuguese (real and fake news) and a web application (SIRENE), to present the user with the probability of an informed news being true or false, from the news text or through the link.

Keywords: Fake News. Machine Learning. Web System.

LISTA DE ILUSTRAÇÕES

Figura 1 – Diagrama do processo de mineração de texto.	20
Figura 2 – Curva da regressão logística.	24
Figura 3 – Representação do Gradiente Estocástico.	25
Figura 4 – Representação de uma Rede Neural Artificial.	26
Figura 5 – Exemplo de hiperplano do SVM.	28
Figura 6 – Interface Web da Aplicação Desenvolvida	31
Figura 7 – Interface Web da Aplicação <i>FakeCheck</i>	33
Figura 8 – Visão conceitual da solução SIRENE.	35
Figura 9 – Técnica de <i>holdout</i> para comparação dos algoritmos.	41
Figura 10 –Validação cruzada com 5 iterações.	43
Figura 11 –Interface do SIRENE (<i>Web</i>).	44
Figura 12 –Módulo da Aplicação <i>Web</i> SIRENE.	45
Figura 13 –Nuvem de palavras.	48
Figura 14 –Gráfico comparativo dos algoritmos.	52
Figura 15 –Classificação pela Notícia.	53

LISTA DE TABELAS

Tabela 1 – Comparação dos Trabalhos Relacionados	34
Tabela 2 – Divisão das amostras.	37
Tabela 3 – Matriz de Confusão	42
Tabela 4 – Informações dos dados Sirene-News.	47
Tabela 5 – Informações dos dados <i>Fake.br</i>	47
Tabela 6 – Exemplo de uma notícia Original.	48
Tabela 7 – Exemplo de uma notícia com o pré-processamento.	49
Tabela 8 – Resultado dos Algoritmos.	49
Tabela 9 – Matriz de Confusão LR.	50
Tabela 10 –Matriz de Confusão SGD.	50
Tabela 11 –Matriz de Confusão MLP.	50
Tabela 12 –Matriz de Confusão SVM.	50
Tabela 13 –Tempo de execução no processo de treinamento.	50
Tabela 14 –Resultados da Validação Cruzada.	51

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizado de Máquina
CSV	<i>Comma Separated Values</i>
DARPA	<i>Defense Advanced Research Projects Agency</i>
EUA	Estados Unidos da América
IA	Inteligência Artificial
IBGE	Instituto Brasileiro de Geografia e Estatística
IFCE	Instituto Federal de Educação, Ciência e Tecnologia do Ceará
LR	<i>Logistic Regression</i>
MLP	<i>Multilayer Perceptron</i>
NLTK	<i>Natural Language Toolkit</i>
PLN	Processamento de Linguagem Natural
RNA	Rede Neural Artificial
SGD	<i>Stochastic Gradient Descent</i>
SVM	<i>Support Vector Machine</i>
TF-IDF	<i>Term Frequency–Inverse Document Frequency</i>

SUMÁRIO

1	INTRODUÇÃO	13
2	FUNDAMENTAÇÃO TEÓRICA	17
2.1	<i>Fake News</i>	17
2.2	Inteligência Artificial	18
2.3	Mineração de Texto	19
2.4	Aprendizado de Máquina	21
2.4.1	Regressão Logística	23
2.4.2	Gradiente Descendente Estocástico	24
2.4.3	Redes Neurais Artificiais	26
2.4.4	<i>Support Vector Machine</i>	28
2.5	Processamento de Linguagem Natural	29
3	TRABALHOS RELACIONADOS	31
4	PROPOSTA	35
4.1	Extração dos Dados	36
4.2	Normalização dos Dados	37
4.2.1	Pré-processamento	37
4.2.2	Transformação dos Dados	38
4.3	Treinamento e Avaliação	38
4.3.1	Treinamento	38
4.3.2	Comparação dos Modelos	41
4.4	SIRENE - Aplicação <i>Web</i>	44
4.4.1	Probabilidade do SVM	46
5	RESULTADOS	47
5.1	Análise da Base de Dados	47
5.2	Comparação dos Algoritmos	48
5.3	Aplicação <i>Web</i>	52
6	CONCLUSÕES	54
	REFERÊNCIAS	55

1 INTRODUÇÃO

A sociedade tem buscado maneiras práticas e rápidas de se comunicar e realizar suas tarefas cotidianas. Em paralelo, é evidente o crescimento do número de internautas trocando informações em todo o mundo. De acordo com a matéria “*We Are Social*” (KEMP, 2018) existem, atualmente, 4 bilhões de pessoas conectadas à Internet. Destaca-se que a população mundial atual gira em torno de 7.6 bilhões de pessoas (CIRIACO, 2018). Logo, pode-se concluir que mais da metade da população mundial está conectada à rede.

No cenário brasileiro, um levantamento realizado em (STATISTA, 2017) mostra que em 2016 cerca de 81.4 milhões de pessoas acessavam a Internet através de dispositivos móveis (*smartphones, tablets* e afins), ou seja, quase 40% da população brasileira. Em 2017, segundo o IBGE (SILVEIRA, 2018), o país tinha aproximadamente 126.4 milhões de usuários da Internet, representando 69,8% da população acima de 9 anos.

Assim, dada a abrangência da Internet, o acesso e compartilhamento da informação têm ficado cada vez mais fácil e rápido, principalmente com a ampla utilização dos dispositivos móveis. O fato de ela poder ser acessada a partir de dispositivos móveis, seja em sites ou redes sociais, torna o processo de compartilhamento mais rápido. Em particular, uma pesquisa realizada em 2018 pelo *Reuters Institute*¹ concluiu que 66% dos entrevistados no Brasil usavam as mídias sociais como fonte de notícias. Falando em números específicos, no ano de 2018 haviam 4.021 bilhões de pessoas online nas redes sociais, o que representava 53% de todas as pessoas do planeta, um aumento de 7% em relação ao ano anterior (CIRIACO, 2018). Embora as redes sociais tenham aumentado a facilidade de disseminar a informação, sua popularidade potencializou o problema das notícias falsas, acelerando a velocidade e o escopo no qual essas notícias são disseminadas. Essa rápida divulgação de notícias falsas já é colocada como um dos problemas do século 21 e é comumente chamada de *fake news* (DELMAZO; VALENTE, 2018).

As *fake news* ganharam grande destaque na imprensa internacional no ano de 2016, durante a eleição presidencial dos EUA (Estados Unidos da América). Empresas de tecnologia fizeram uma análise e descobriram vários conteúdos falsos na Internet, sendo a maioria para denegrir a imagem de um dos candidatos à presidência daquele ano (BATISTA, 2018). Ainda nos EUA, notícias falsas alegando que Barack Obama havia sido ferido em uma explosão consumiram US\$ 130 bilhões em ações,

¹ <http://www.digitalnewsreport.org>

causando um impacto na economia do país (RAPOZA, 2017), o índice *Dow Jones* (principal índice acionário das bolsas de valores dos Estados Unidos da América) chegou a cair mais de 100 pontos, poucos minutos depois da publicação dessa notícia falsa. Outro exemplo ocorreu em 2014, quando uma notícia falsa resultou no assassinato de uma mulher no Brasil após circularem boatos na Internet acusando-a de sequestrar crianças para rituais de magia negra (D'AGOSTINO, 2017). Segundo o *Google Trends* (ferramenta do *Google* que expõe os temas de maior interesse dos usuários), o termo *fake news* foi um dos mais pesquisados pelos brasileiros no ano de 2018 e continuou em crescimento no ano de 2019. Exemplos como estes ratificam a relevância de reduzir a disseminação de notícias falsas na Internet (SHU et al., 2019).

Segundo Marchi (MARCHI, 2012), os jovens tendem a consumir menos os meios de comunicação mais antigos, como jornais, rádio e TV, pois acreditam que, além de serem desinteressantes, são repetitivos. Assim, eles acabam utilizando as redes sociais como principal meio de informação, resultando em jovens mais direcionados a pensar e acreditar em assuntos provenientes de informações manipuladas. As notícias publicadas em redes sociais e aplicativos de troca de mensagens possuem um baixo nível de confiabilidade (NEWMAN et al., 2019), causado principalmente pela falta de um corpo editorial e de uma filtragem de *Fake News* ou a inexistência dela. Ainda assim, as *fake news* conseguem atingir um público que "...quer acreditar nelas, as consome mesmo que suspeite delas, pois quer ver sua ideologia e seus preconceitos confirmados..." (HERMIDA, 2018).

O *Facebook* e *Google* vêm buscando sanar a problemática das notícias falsas em suas plataformas, pois recebem constantemente fortes críticas, tanto por parte da população quanto da mídia, por ainda não implementarem ferramentas eficientes capazes de detectar *fake news* e barrar sua disseminação (WINGFIELD; ISAAC; BANNER, 2016). Recentemente, a Comissão Europeia afirmou que *Google*, *Facebook* e *Twitter* devem se empenhar mais para combater as notícias falsas em suas redes, estando sujeitas a ações de autoridades regulatórias (SZAFRAN, 2019). O alerta do órgão executivo da União Europeia foi emitido um ano após as gigantes da tecnologia, juntamente com *Microsoft*, *Mozilla* e sete outros órgãos comerciais europeus, assinarem um código de conduta voluntário para combater as *fake news*.

O tema notícias falsas não só vem recebendo grande atenção do público em geral, mas também vem atraindo uma crescente atenção da comunidade acadêmica. Um dos principais objetivos dos pesquisadores é detectar esse tipo de notícia na tentativa de mitigar seus impactos negativos, o que não é uma tarefa trivial. O fato de as *fake news* serem intencionalmente escritas para enganar os leitores torna o processo de detecção complexo, sendo necessárias soluções computacionais sofisticadas para detectá-las. Além disso, as notícias são disseminadas em larga escala, abrangendo

diversos temas e, muitas vezes, são escritas por usuários anônimos. Assim, alguns trabalhos estão propondo mecanismos capazes de reconhecer as *fake news* entre as notícias disseminadas na Internet. Dentre as soluções computacionais utilizadas, uma das mais difundidas é a Aprendizagem de Máquina (AM) (MONARD; BARANAUSKAS, 2003), que permite aos computadores desenvolver o reconhecimento de padrões e a capacidade de aprender continuamente com os dados, fazendo previsões neles baseados e, então, ajustando-se sem serem especificamente programados para isso.

Neste contexto, este trabalho apresenta a SIRENE, uma solução inteligente para detecção de notícias falsas em português. O objetivo principal do SIRENE é reconhecer padrões existentes em notícias na Internet com informações reais e falsas. Para tanto, foram utilizados diversos algoritmos de AM supervisionado. Neste trabalho, foram extraídas 4.742 notícias dos portais G1 e Sensacionalista. Essa base foi intitulada *Sirene-news*². Além disso, foram utilizadas 7.200 notícias da base de dados do *Fake.br* (MONTEIRO et al., 2018), aumentando o corpus de dados. As bases passaram por uma fase de pré-processamento antes do treinamento e teste. Após o processo de classificação, os resultados foram analisados, sendo selecionado o algoritmo que melhor se adequou aos dados. Por fim, com o intuito de disponibilizar para o público geral os resultados da proposta, é disponibilizado um serviço web do SIRENE³ para apresentar ao usuário a probabilidade de uma notícia informada ser verdadeira ou falsa.

Portanto, o SIRENE tem como principal contribuição disponibilizar uma solução que visa reduzir o impacto das notícias falsas e servir como um modelo para outras propostas que intentam resolver a mesma problemática. Os resultados podem auxiliar tanto no desenvolvimento de ferramentas que ajudem no combate de notícias falsas quanto em outras áreas que envolvam aprendizagem de máquina. É importante destacar que a proposta não trata direcionamento ideológico das notícias. Ou seja, o trabalho foca apenas na detecção de notícias com uma maior ou menor probabilidade de trazerem conteúdos falsos.

Durante os processos de pesquisa e desenvolvimento deste Trabalho de Conclusão de Curso (TCC), obteve-se o seguinte artigo aceito e publicado em uma conferência internacional: **AuFa - Automatic Detection and Classification of Fake News Using Neural networks** no *8th International Workshop on ADVANCEs in ITC Infrastructures and Services* (BARBOSA; OLIVEIRA; BRAGA, 2020).

O trabalho está dividido em capítulos para melhor entendimento. O Capítulo 2 apresenta a Fundamentação Teórica, que introduz todo o embasamento teórico necessário para o desenvolvimento desse trabalho. Os Trabalhos Relacionados, no

² Base disponível em <https://github.com/ViniciusNunes0/SIRENE-news>

³ <http://lar.ifce.edu.br:5000>

Capítulo 3, apresentam os trabalhos e pesquisas realizadas por outros pesquisadores que possuem relação com esse trabalho. O Capítulo 4, denominado Proposta, contém as etapas realizadas para alcançar os objetivos traçados. O Capítulo 5 de Resultados traz os resultados obtidos com o algoritmo implementado para a classificação. Por fim, o Capítulo 6 de Conclusão, que apresenta as considerações finais do trabalho e os trabalhos que poderão ser realizados no futuro.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a fundamentação teórica necessária para uma melhor compreensão da proposta do trabalho. Os principais conceitos são: *Fake News*, Inteligência Artificial, Mineração de Texto, Aprendizado de Máquina e os respectivos algoritmos utilizados nessa solução, e por fim, o processamento de linguagem natural.

2.1 *Fake News*

O termo *fake news* (notícias falsas), de acordo com (JR; LIM; LING, 2018), é todo conteúdo falso e sem embasamento, com o intuito de enganar e manipular o leitor. Esse termo tem sido muito mencionado atualmente, inclusive, a imprensa tem feito várias matérias para relatar sobre o tema. Este tópico voltou a ser bastante utilizado após as eleições presidenciais de 2016 nos Estados Unidos da América, bem como em diversos outros acontecimentos. A criação de *fake news* tem também a finalidade de denegrir uma pessoa ou entidade. Além disso, elas também visam atrair a atenção das pessoas para aumentar o acesso aos sites.

Os padrões existentes entre as notícias verdadeiras e falsas têm sido frequentemente estudados para identificar as características que podem ajudar a diferenciá-las. Os trabalhos de detecção de notícias falsas dependem essencialmente da exploração destas diferenças para classificar as informações. De acordo com (SHARMA et al., 2019), a maioria dos trabalhos existentes visa principalmente a classificação de forma binária (falso/verdadeiro, boato/não-boato, confiável/não-confiável). Esse tipo de abordagem está sendo utilizada nesse trabalho.

Seguindo o estudo do (SHARMA et al., 2019), existem três tipos de abordagens para o combate as *fake news*. O primeiro tipo é a identificação de notícias falsas usando métodos baseados em conteúdo, que classificam as notícias com base no conteúdo das informações a serem verificadas. O segundo tipo é a identificação usando métodos baseados em *feedback*, que classificam as notícias com base nas respostas dos usuários que recebem nas mídias sociais. Por fim, o terceiro tipo são soluções baseadas em intervenção, que fornecem soluções computacionais para identificar e conter a disseminação dessas informações e, assim, aplicar métodos para mitigar o impacto das informações falsas na sociedade. Nesse trabalho está sendo utilizado o método baseado em conteúdo, por ser uma abordagem simples e eficaz, pois utiliza um conjunto de dados (*dataset*) para aprender os padrões e características que diferenciam ambas as classes (Verdadeiro/Falso).

Segundo um estudo realizado por (DIZIKES, 2018), notícias falsas se espalham de forma muito mais rápida no Twitter, sendo 70% mais propensas a serem compartilhadas do que as notícias verdadeiras. A propagação de *fake news* se torna cada vez mais fácil devido aos mecanismos que compartilham essas notícias e as disseminam na rede de forma automática. Uma delas é chamada de *bot* ou robô, (SHAO et al., 2017) que são algoritmos usados para automatizar uma atividade. Estes robôs podem ser usados de forma positiva, auxiliando atendimentos virtuais, bem como de forma maliciosa, compartilhando automaticamente conteúdos selecionados, como propagandas políticas. Tais robôs são treinados por meio de aprendizado de máquina, se disfarçando de usuários e propagando as notícias falsas. Uma das plataformas que mais propicia essas atividades é o Twitter, por não proibir a criação de contas falsas ou automatizadas.

Além de *bots*, a propagação de *fake news* também é feita por usuários que compartilham as informações sem analisar determinada notícia. Além disso, jornalistas acabam não verificando a veracidade das notícias e divulgam com o objetivo de colocar a notícia em primeira mão (BAKIR; MCSTAY, 2018).

2.2 Inteligência Artificial

A Inteligência Artificial (IA) de acordo com (NING; YAN, 2010), é uma área da ciência da computação que pesquisa e desenvolve métodos, técnicas e aplicações para simular, estender e expandir a teoria da inteligência humana. A inteligência artificial possibilita que máquinas aprendam com experiências, se ajustem a novas entradas de dados e realizem tarefas de forma similar aos seres humanos. Existem várias pesquisas na área de IA voltadas para o reconhecimento de fala, reconhecimento de imagem, processamento de linguagem natural, sistemas especialistas e robótica.

As primeiras pesquisas de IA nos anos 50 exploraram temas como a resolução de problemas e métodos simbólicos. Na década de 60, o Departamento de Defesa dos EUA se interessou por este tipo de tecnologia e começou a treinar computadores para imitar o raciocínio humano básico. Por exemplo, a *Defense Advanced Research Projects Agency* (DARPA) completou um projeto de mapeamento de ruas nos anos 70. A DARPA criou assistentes pessoais inteligentes em 2003, muito tempo antes de Siri, Alexa ou Cortana serem desenvolvidos (SAS, 2019).

Com base no trabalho dos autores (HOSEA S.; HARIKRISHMAN, 2011), a Inteligência Artificial tem atuado principalmente nas seguintes áreas:

- Jogos Eletrônicos: Que possibilitam programar computadores para jogos de xadrez e damas, os quais demandam algoritmos baseados em raciocínio lógico.

Esses jogos são desenvolvidos com o intuito de fazer o computador vencer os seres humanos;

- **Sistemas Especialistas:** São sistemas desenvolvidos para os computadores, que são capazes de tomar decisões em situações da vida real. Por exemplo, sistemas especialistas que ajudam médicos a diagnosticar doenças a partir dos sintomas;
- **Linguagem Natural:** Sistemas capazes de compreender a linguagem humana, muito utilizado em tradutores e sistemas que possuem uma interação direta com os seres humanos, como os *chatbots*;
- **Redes Neurais:** Sistemas que simulam a neurônios na tentativa de reproduzir os tipos de conexões físicas que ocorrem no cérebro humano;
- **Robótica:** Criação de máquinas capazes de reconhecer, perceber e reagir a outros estímulos sensoriais dentro de um ambiente.

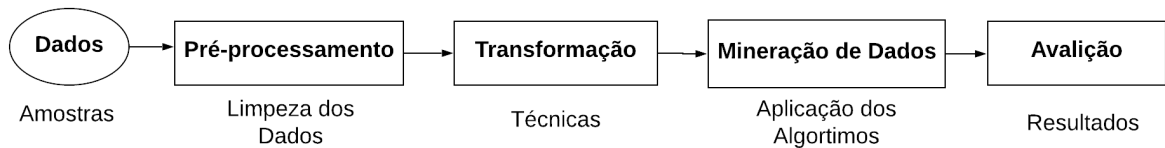
2.3 Mineração de Texto

A mineração de texto pode ser considerada uma busca de informações relevantes, capazes de descobrir informações estruturadas, através do uso de bancos de dados e de procedimentos estatísticos, utilizando um grande volume de texto escrito em linguagem natural. A Mineração de texto é um conjunto de métodos usados para navegar, organizar, achar e descobrir informação em bases textuais. Pode ser vista como uma extensão da área de mineração de dados (ARANHA; PASSOS, 2006).

Segundo (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996), o processo de extrair conhecimento de dados e utilizá-lo como estratégias para tomada de decisões é chamado Descoberta de Conhecimento em Bancos de Dados (*Knowledge Discovery in Databases* - KDD). Ou seja, o KDD pode ser dividido nas seguintes etapas: aquisição de dados, pré-processamento de dados, transformação, mineração de dados e avaliação. Esses passos devem ser feitos em sequência, pois cada etapa depende do resultado da etapa anterior.

A Figura 1 apresenta as etapas da mineração de textos utilizada nesse trabalho. A seguir é descrito cada uma dessas etapas:

- **Coleta de Dados:** É realizada a extração de textos a ser trabalhada, os textos podem ser extraídos de redes sociais, sites e diversos outros meios, a coleta de amostras/dados dependendo do objetivo a ser alcançado pode ser uma etapa trabalhosa e custosa;

Figura 1 – Diagrama do processo de mineração de texto.

Fonte: Baseado no autor (MARUMO, 2018)

- **Pré-processamento:** Tem como objetivo transformar os dados não estruturados para representação atributo-valor, melhorando a qualidade dos dados e organizando os mesmos (LEAL, 2018). Esse processo é necessário porque, na maioria dos casos, os dados originais podem ser inadequados para o processo de treinamento dos algoritmos. Por isso é necessário realizar o pré-processamento, que consistem em realizar uma limpeza no texto para garantir a qualidade dos dados;
- **Transformação:** Para que os dados possam ser processados pelos algoritmos de Aprendizado de Máquina, elas devem ser convertidas para um formato apropriado. Para isso, existem várias técnicas que realizam essa conversão, nesse trabalho foi utilizada a técnica de representação de palavras TF-IDF, disponível na implementação da classe *TfidfVectorizer* da biblioteca *scikit-learn*¹, utilizada para determinar o peso de uma palavra de um texto em uma coleção de textos. Ele converte essa coleção de textos brutos em uma matriz de recursos TF-IDF. A lógica utilizada pelo TF (2.1) e IDF (2.2) está representada a seguir:

$$TF(t) = \frac{\text{n}^{\circ} \text{ de vezes que } t \text{ aparece no texto}}{\text{total de termos no texto}} \quad (2.1)$$

$$IDF(t) = \log_e\left(\frac{\text{quantidade total de textos}}{\text{numero de textos em que } t \text{ aparece}}\right) \quad (2.2)$$

O t representa o termo que está sendo calculado. O *TF-IDF* é o produto entre as duas equações;

- **Mineração de dados:** No processo de Mineração de Dados, é escolhido o algoritmo que realiza a extração do conhecimento dos dados, levando em consideração o problema que deve ser resolvido. Esse processo pode ser repetido até que sejam alcançados resultados adequados (MORAIS; AMBRÓSIO, 2007). De acordo com (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996), existem algumas tarefas nas quais as técnicas de Mineração de Dados podem ser aplicadas: Classificação, Regressão e Agrupamento. Nesse trabalho, é adotada a classificação, que consiste na criação de um modelo capaz de classificar dados que não estão classificados;

¹ https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

- **Avaliação:** Após o fim do processo de mineração de dados é realizada a análise dos resultados alcançados pelo algoritmo, no fim do processo de treinamento. Existem várias métricas para analisar o modelo, como: acurácia, precisão, Revocação(Recall), entre outros. Caso as avaliações não sejam satisfatórias, é possível repetir o processo todo até obter um resultado adequado.

2.4 Aprendizado de Máquina

De acordo com (GOLDSCHMIDT, 2010), o Aprendizado de Máquina (AM) é uma área da inteligência artificial voltada para o desenvolvimento de algoritmos e técnicas computacionais, que possibilitam que os computadores sejam capazes de aprender. Esse processo de aprendizagem permite que os computadores desenvolvam o reconhecimento de padrões e a capacidade de aprender continuamente com os dados, fazendo previsões neles baseadas e, então, ajustando-se sem serem especificamente programados para isso. Tanto a qualidade quanto a quantidade das informações dos dados utilizados para o AM são relevantes para uma melhor performance e acerto de previsão (MOHRI; ROSTAMIZADEH; TALWALKAR, 2018).

Segundo (PEREIRA, 2013), padrões são conjuntos de características que determinam um objeto, sendo que essas características são determinadas buscando a separação das classes a fim de simplificar as tarefas de um classificador. Existem quatro categorias comuns para o aprendizado de máquina, sendo elas: supervisionada, não supervisionada, semi-supervisionado e aprendizado por reforço.

Aprendizado supervisionado

Nos modelos de aprendizado de máquina supervisionada o ser humano controla a entrada e saída e é capaz de dar pesos ou calibrar o nível de assertividade e de precisão de um modelo, por meio da rotulação dos dados. O objetivo é encontrar os parâmetros ótimos que ajustem um modelo para prever rótulos desconhecidos em outros objetos (KOTSIANTIS; ZAHARAKIS; PINTELAS, 2007). Eles são divididos em regressão e classificação. A regressão busca encontrar como uma variável evolui em relação a outras. Estão entre os métodos mais comuns nas aulas de estatística nas universidades. Já a classificação, busca explicar uma variável categórica, com duas categorias (variável binária) ou mais.

Dentre as técnicas mais conhecidas para resolver problemas de aprendizado supervisionado estão regressão linear, regressão logística, redes neurais artificiais, máquinas de suporte vetorial, árvores de decisão e k-vizinhos mais próximos.

Aprendizado não supervisionado

Os algoritmos não supervisionados, são treinados com dados não rotulados, e utilizam mecanismos de agrupamento, como *clusterização*, para agrupar esses dados de acordo com seus atributos. O objetivo, é observar algumas similaridades entre os dados e incluí-los em cada grupo apropriado.

O aprendizado não supervisionado, podem ser aplicados em sistemas de recomendação de filmes ou músicas, detecção de anomalias e visualização de dados. Dentre as técnicas mais conhecidas para resolver problemas de aprendizado não supervisionado estão redes neurais artificiais, clusterização k-médias, *word2vec*, entre outros (KOTSIANTIS; ZAHARAKIS; PINTELAS, 2007). Os problemas de aprendizado não supervisionado são consideravelmente mais complicados do que problemas de aprendizado supervisionado, principalmente porque não tem a resposta anotada nos dados. Como consequência, é complicado avaliar um modelo de aprendizado não supervisionado e esse tipo de modelo está na fronteira do conhecimento em aprendizado de máquina.

Aprendizado semi-supervisionado

O aprendizado semi-supervisionado utiliza os exemplos rotulados para se obter informações sobre o problema e utilizá-las para guiar o processo de aprendizado a partir dos dados não rotulados (BRUCE, 2001). Utilizado geralmente quando uma pequena parcela dos dados é rotulada e a outra parte não é. O aprendizado semi-supervisionado pode ser utilizado tanto em tarefas de classificação quanto em tarefas de *clustering* (BASU; BANERJEE; MOONEY, 2002).

Em uma classificação semi-supervisionada, a ideia é rotular, com uma certa margem de segurança, alguns dos exemplos no conjunto de exemplos não rotulados, os quais são posteriormente utilizados durante a fase de treinamento do classificador, frequentemente resultando em uma classificação mais precisa. Já em *clustering* semi-supervisionado, os exemplos rotulados são utilizados no processo de formação dos *clusters*, servindo geralmente como um conhecimento de fundo, expresso, por exemplo, na forma de restrições, e resultando em melhores *clusters*.

Aprendizado por reforço

O aprendizado por reforço tem como meta reforçar ou recompensar uma ação considerada positiva e punir uma ação considerada negativa. Um exemplo de tarefa de reforço é a de ensinar um robô a encontrar a melhor trajetória entre dois pontos.

Algoritmos de aprendizado utilizados nessa tarefa, em geral, punem a passagem por trechos pouco promissores e recompensam a passagem por trechos promissores (VINICIUS, 2017). Basicamente, o aprendizado por reforço é quando a máquina tenta aprender qual é a melhor ação a ser tomada, dependendo das circunstâncias na qual essa ação será executada. Ela é caracterizada pela tentativa e erro, onde o algoritmo é treinado e realiza esse ciclo, até obter o melhor resultado (KAELBLING; LITTMAN; MOORE, 1996).

Algumas das aplicações de AM por reforço são construir oponentes em videogames, movimentação de robôs, simulações de ambientes complexos e aprender estratégias de troca no mercado financeiro.

2.4.1 Regressão Logística

A regressão logística é uma técnica recomendada para situações em que a variável dependente é de natureza binária, ou seja, os dados a serem classificados são 0 (Verdadeiro) ou 1 (Falso). A regressão logística é um recurso que permite estimar a probabilidade associada à ocorrência de um determinado evento com base em um conjunto de variáveis explanatórias.

De acordo com (HOSMER; JOVANOVIĆ; LEMESHOW, 1989), na regressão logística, a probabilidade de ocorrência de um evento pode ser estimada diretamente. Conforme apresentado na Figura 2, no caso da variável Y assumir apenas dois possíveis estados (1 ou 0) e haver um conjunto de p variáveis independentes X_1, X_2, \dots, X_p , o modelo de regressão logística pode ser escrito da seguinte forma:

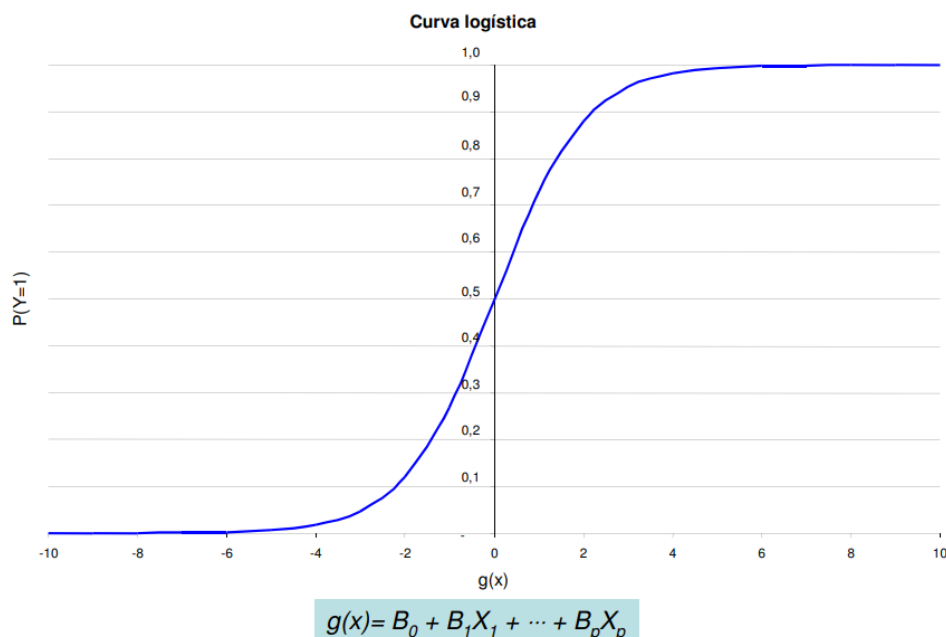
$$P(y = 1) = \frac{1}{1 + e^{-g(x)}} \quad (2.3)$$

onde, $g(x) = B_0 + B_1X_1 + \dots + B_pX_p$.

Os coeficientes B_0, B_1, \dots, B_p são estimados a partir do conjunto dados, pelo método da máxima verossimilhança, em que encontra uma combinação de coeficientes, que maximiza a probabilidade da amostra ter sido observada. Considerando uma certa combinação de coeficientes B_0, B_1, \dots, B_p e variando os valores de X , observou-se que a curva logística tem um comportamento probabilístico no formato da letra S (Figura 2), o que é uma característica da regressão logística (HOSMER; JOVANOVIĆ; LEMESHOW, 1989).

- Quando $g(x) \rightarrow -\infty, P(Y = 1) \rightarrow 0$
- Quando $g(x) \rightarrow +\infty, P(Y = 1) \rightarrow 1$

Figura 2 – Curva da regressão logística.



Fonte: (CLAÚDIO; CAMPI, 2018)

O modelo de regressão logística, de acordo com o *PortalAction*² é semelhante ao modelo de regressão linear. No entanto, no modelo logístico, a variável resposta Y_i é binária. Uma variável binária assume dois valores, como por exemplo, $Y_i = 0$ e $Y_i = 1$, denominados “fracasso” e “sucesso”, respectivamente. Além disso, a regressão linear utiliza o método dos mínimos quadrados, já a regressão logística utiliza o método da máxima verossimilhança.

Neste trabalho, o modelo estatístico tem como objetivo produzir uma saída binária, a partir de um conjunto de textos de acordo com cada notícia.

2.4.2 Gradiente Descendente Estocástico

Gradiente descendente, de acordo com os autores de (GEOINFORMAÇÃO, 2019), é um método que consiste em encontrar, de forma iterativa, os valores dos parâmetros que minimizam determinada função de interesse. Para entender esse método, inicialmente, é necessário lembrar como é a obtenção dos parâmetros de forma analítica. Considere um problema de regressão linear em que $X \in M_{n \times p}(R); n > p; (X) = p; Y_i, i = 1, \dots$ dado por:

$$Y_i = \beta^t x_i + \varepsilon_i; \varepsilon \sim N(0, \sigma^2). \quad (2.4)$$

² <http://www.portalaction.com.br/analise-de-regressao/regressao-logistica>

O método do Gradiente Descendente utiliza – em cada iteração – toda a amostra de treino. Trata-se de um algoritmo útil para matrizes bem condicionadas e problemas fortemente convexos. Entretanto, frequentemente traz problemas, pois problemas interessantes não são fortemente convexos ou bem condicionados. Além disso, tal abordagem pode ser muito custosa computacionalmente, pois para cada atualização do conjunto de treinamento, precisa-se ensinar o algoritmo desde o início. Isso inviabiliza qualquer método online (expediente em que os exemplos chegam em um fluxo contínuo) (GEOINFORMAÇÃO, 2019).

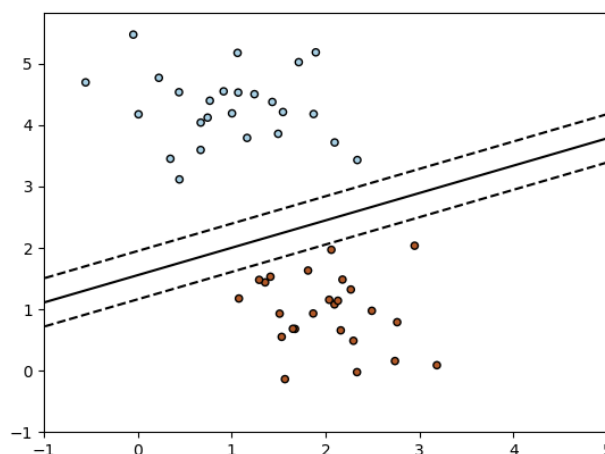
O Gradiente Descendente Estocástico (*Stochastic Gradient Descent* - SGD) minimiza esses problemas, pois utiliza, em cada iteração, apenas uma observação. Considere o par (x_i, y_i) amostrado do treinamento, a atualização dos parâmetros é dado como segue (GEOINFORMAÇÃO, 2019).

Algoritmo: Escolha um valor aleatório inicial, $\beta^0 \in R^p$ e repita até atingir convergência:

$$\beta^{(k+1)} = \beta^{(k)} - \alpha_k J(\beta^{(k)}; x_i, y_i), k = 0, 1, \dots \quad (2.5)$$

A aprendizagem pode ser muito mais rápida com o gradiente descendente estocástico para conjuntos de dados de treinamento muito grandes. Em muitos casos, é necessário apenas um pequeno número de passagens através do conjunto de dados para alcançar um conjunto de coeficientes suficientemente bom (DEEPLARNING-BOOK, 2019).

Figura 3 – Representação do Gradiente Estocástico.



Fonte: (SCIKITLEARN, 2019b).

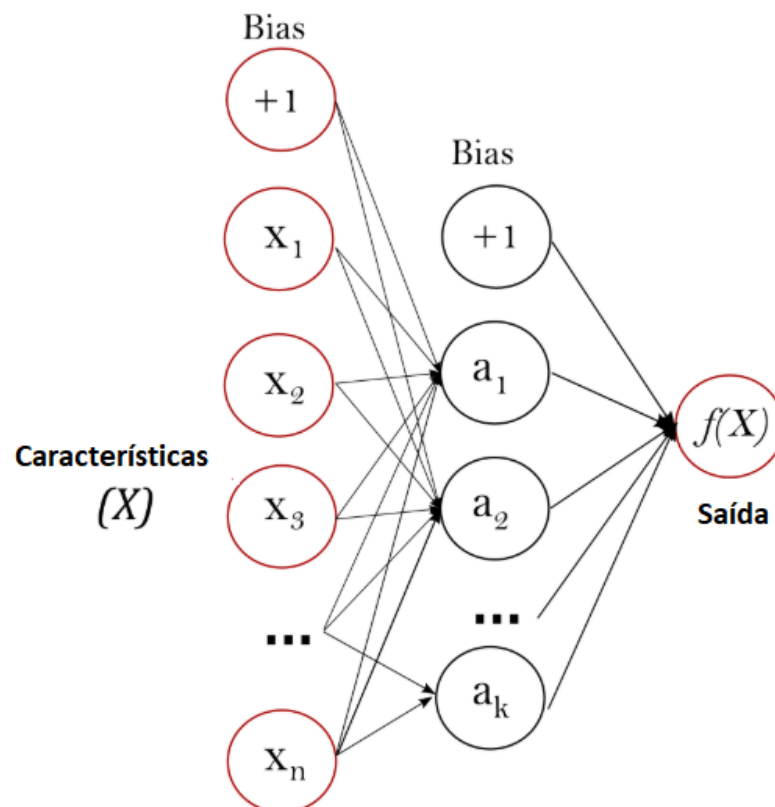
O algoritmo *SGDClassifier* implementa uma rotina simples de aprendizado de descida de gradiente estocástico, que suporta diferentes funções de perda e penalidades para classificação, como exemplificado na Figura 3.

Um conjunto de dados iniciais são submetidos ao algoritmo *SGDClassifier*, que aprende com esses dados e a partir daí, executa a função da descida de gradiente estocástico, explicado anteriormente, e separa os conjunto traçando uma linha para delimitar as classes, como exemplificado na Figura 3. Por exemplo, as bolinhas verdes podem ser vistas como o conjunto de notícias verdadeiras e as bolinhas vermelhas o conjunto de notícias falsas.

2.4.3 Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) usam algoritmos computacionais baseados na estrutura neural do cérebro. O primeiro modelo proposto foi chamado de *perceptron*, que consiste em um único neurônio na camada de saída, a Figura 4 é um exemplo de um *perceptron*, com p neurônios na camada de entrada e um único neurônio na camada de saída. Esse modelo é capaz de resolver problemas linearmente separáveis de classificação binária (HAYKIN et al., 2009).

Figura 4 – Representação de uma Rede Neural Artificial.



Fonte: (SCIKITLEARN, 2019a).

Uma RNA possui várias camadas para o processamento da informação. Cada camada possui vários nós, sendo que cada nó simula um "neurônio". Todas as camadas são interligadas, e cada "neurônio" possui um peso de acordo com a informa-

ção armazenada (CABRAL et al., 2007). Por fim, o peso de um neurônio é recalculado a cada interação do algoritmo de aprendizagem (*backpropagation*) para correção dos pesos, o funcionamento mais detalhado desse algoritmo encontra-se no artigo (HECHT-NIELSEN, 1992).

O *Multilayer Perceptron* (MLP), é uma variação da *perceptron* que possui pelo menos uma camada intermediária, conhecida como camada de neurônios oculta. Para que a MLP não seja reduzida a uma *perceptron*, há a necessidade de se usar uma função que seja diferenciável e com fator de não-linearidade suave. Com isso, utiliza-se, como funções de ativação, a função logística e tangente hiperbólica (HAYKIN, 2007). Por sua flexibilidade de funções, a MLP pode ser encontrada em diversos sistemas, como de processamento de sinais, reconhecimento de padrões e previsão de séries temporais (GOMIDE, 2012).

A informação nessa topologia de rede da Figura 4, é passada em um único sentido, chamada de arquitetura *feedforward*. O seu treinamento é feito por meio do algoritmo de *backpropagation*, de forma supervisionada, em que deseja-se encontrar o mínimo local ou global da função do erro entre entrada e a saída (JR, 1985).

A etapa *forward* da MLP pode ser avaliado matematicamente pela equação:

$$Y = g_2(g_1(X_x W^t)_x W'^t) \quad (2.6)$$

Onde:

Y representa o vetor de saída da rede neural;

g_1 e g_2 são as funções de ativação tangente hiperbólica e logística, respectivamente.

x são os vetores dos dados de entrada;

W^t e W'^t representam as matrizes de pesos sinápticos transpostos.

Uma rede neural aprende através de um processo iterativo de ajuste de seus pesos sinápticos. O processo de aprendizagem segue a seguinte sequência:

- a rede neural é estimulada pelo ambiente de informação;
- a estrutura interna da rede é alterada como resultado do estímulo;
- devido às alterações que ocorrem em sua estrutura interna, a rede tem modificada sua resposta aos estímulos do ambiente.

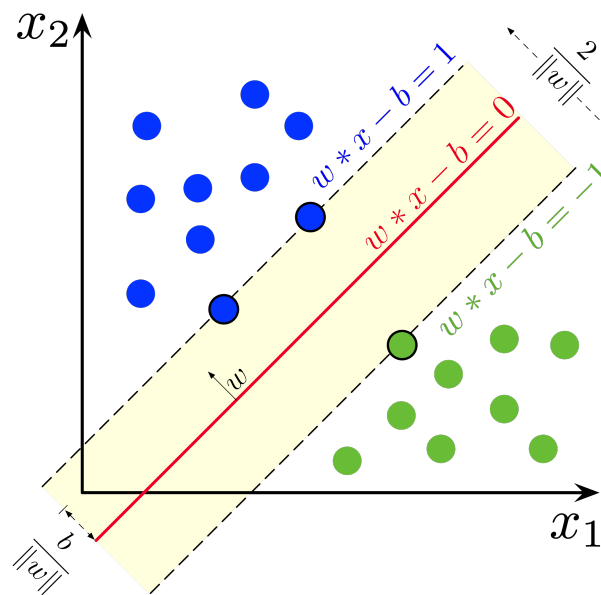
Utilizando vários neurônios em múltiplas camadas é possível resolver problemas lineares e não lineares. Essa estrutura é composta por três camadas interconectadas: camada de entrada, camada de saída e camadas ocultas (que ficam entre as

camadas de entrada e saída). Essas camadas são representadas pela Figura 4, onde cada camada é composta por um ou mais neurônios, representados pelos círculos, e cada neurônio é totalmente conectada aos neurônios da próxima camada de acordo com o (HAYKIN et al., 2009).

2.4.4 Support Vector Machine

O *Support Vector Machines* (SVM), é um método de aprendizado supervisionado usado para classificação e regressão. Esse algoritmo é uma abordagem relativamente nova e tem apresentado bom desempenho nos últimos anos (KUDO; MATSUMOTO, 2001). O SVM é um classificador muito utilizado por ser efetivo em classificação de texto. O mesmo é baseado na teoria de aprendizagem estatística, desenvolvida por (DRUCKER et al., 1997).

Figura 5 – Exemplo de hiperplano do SVM.



Fonte: (WIKIWAND, 2020)

O classificador SVM é baseado em classificadores lineares, ou seja, os dados são separados por uma linha (Figura 5) e suas classes são de ordem binária (0 ou 1). Um conjunto de treinamento, segundo os autores (HAYKIN, 2007; SEMOLINI et al., 2002), é dito linearmente separável se for possível separar os padrões de classes diferentes contidos no mesmo por pelo menos um hiperplano (linha). Para alcançar essa “linha ótima”, é necessário resolver a equação 2.7:

$$W^t x + b = 0, \quad (2.7)$$

onde $W^t x$ é o produto escalar entre os vetores w e x . O vetor x representa os padrões de entrada do conjunto de treinamento, w é o vetor de pesos ajustáveis e b é um limiar

de separação.

Como exemplificado na Figura 5, o melhor hiperplano separa o conjunto de treinamento em duas partes. A primeira (bolinhas azuis) é o conjunto classificado como positivo e a segunda (bolinhas verdes) é o conjunto classificado como negativo. Sendo assim, quando um novo conjunto (teste) é submetido ao SVM, a sua respectiva posição é calculada e então é analisado o quão próxima ou distante esse novo ponto está do hiperplano (definido no processo de treinamento), por fim, o algoritmo indica a que classe essa nova entrada pertence. Dessa forma o trabalho do classificador consiste em determinar de que lado do hiperplano de \vec{w} a entrada de teste se encontra.

De forma geral, a estratégia utilizada pela SVM (BOSER; GUYON; VAPNIK, 1992) (CORTES; VAPNIK, 1995) consiste em determinar o hiperplano ótimo, o qual representa a superfície de separação com a margem maximizada. Para isso, é utilizado o princípio indutivo do método de minimização estrutural de risco. A implementação deste método de aprendizagem para problemas de classificação permite que a máquina atinja um bom desempenho de generalização ainda que o domínio do problema seja desconhecido. Os vetores de suporte, sobre os quais foi concebida esta categoria de máquina, são padrões extraídos dos dados de entrada que se encontram mais próximos da superfície de decisão. Isso torna a classificação mais complexa e faz com que os dados sejam mais significativos para a determinação do hiperplano ótimo de separação (HAYKIN, 2007).

2.5 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN), de acordo com (SANTOS, 2001), é uma área da Inteligência Artificial que foca no processamento e entendimento de textos. Suas primeiras aplicações surgiram com o que era chamado de Máquinas de Tradução (do inglês, *Machine Translation*), que trabalhavam, como o nome sugere, com a tradução automática de textos entre linguagens diferentes. Com o avanço da tecnologia, novas aplicações foram dadas ao PLN. Na década de 1950, a capacidade de processar, entender e gerar textos foi considerada como uma das qualidades necessárias de uma Inteligência Artificial, as quais foram citadas por Allan Turing em seu Teste de Turing (ZILIO, 2009).

De acordo com os autores (OLIVEIRA; NAVAU, 2004), PLN é uma área de pesquisa que objetiva tratar a informação/dado de forma mais natural possível, por meio da linguagem que os seres humanos estão acostumados. Assim, elimina-se a necessidade de adaptação a formas e variações de interação.

Segundo o autor (MARTINS K KATAOKA, 2010), o entendimento da lin-

guagem natural toma como base as seguintes classificações:

- Fonético: Relacionamento das palavras com os sons tanto para a fala quanto para a escuta;
- Morfológico: Construção das palavras a partir de unidades de significado primitivas e de como classifica-las em categorias morfológicas;
- Sintático: Relacionamento das palavras entre si. Cada palavra tem seu papel estrutural nas sentenças;
- Semântico: Relacionamento das palavras com seus significados e de como essas palavras são compostos para que a sentença se torne compreensível;
- Pragmático: Uso de sentenças em diferentes contextos, afetando o significado;

Em PLN, existe o processo de *tokenização*, que divide um texto em vários *tokens*, seja a *tokenização* de sentenças, dividindo o texto em várias sentenças diferentes, ou de palavras, dividindo cada sentença em várias palavras. Seguindo o estudo de (HABASH; RAMBOW; ROTH, 2009), a *tokenização* é utilizada como uma etapa de pré-processamento do texto, antes da execução de outros procedimentos, como a etiquetagem.

Além da *tokenização*, existem outros processos como o *stemming*, que é o processo de reduzir uma palavra ao seu radical, como por exemplo: “meninas” seria reduzido a “menin”. Este processo é útil para a redução de vocabulário e abstração de significado.

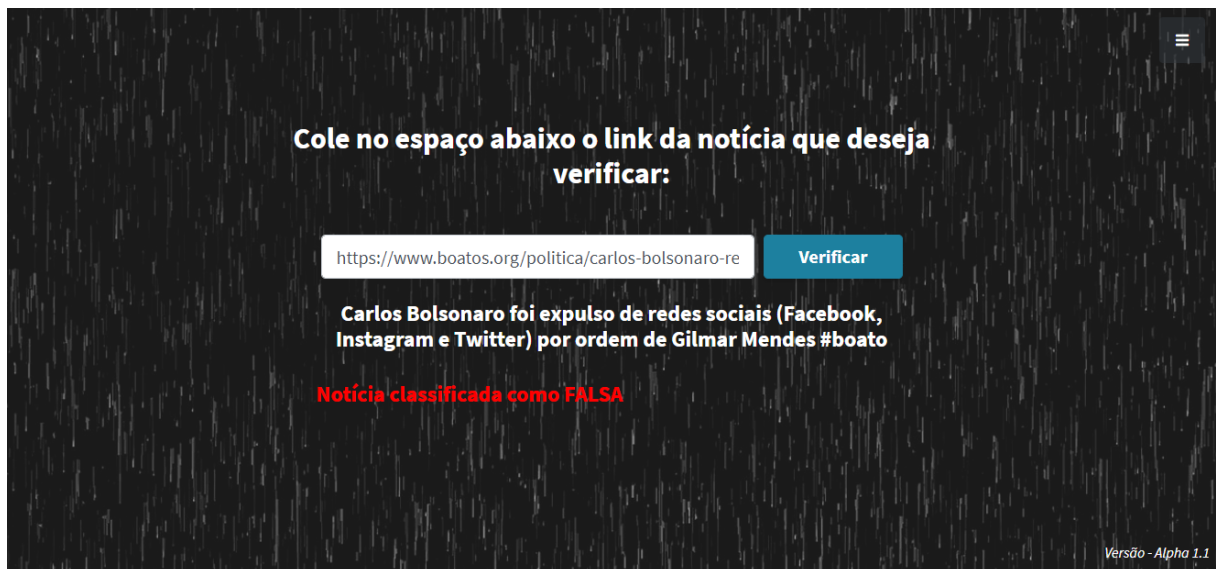
Um detalhe muito importante no processamento de linguagens naturais é a identificação das chamadas *stopwords* do idioma. *Stopword* nada mais é que uma palavra que possui apenas significado sintático dentro da sentença, porém não traz informações relevantes sobre o seu sentido. As *stopwords* possuem uma frequência muito grande em todos os idiomas e, por esse motivo, é preciso eliminá-las das palavras extraídas do texto. Caso contrário, o algoritmo pode dar importância para palavras como: “e”, “ou”, “para”, entre outras. O que, certamente, atrapalharia a análise.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta os principais trabalhos relacionados à pesquisa em classificação de notícias falsas. Os trabalhos estão listados em ordem cronológica.

No que se refere às *fake news* e a aplicação de *Machine Learning*, os autores de (MONTEIRO; NOGUEIRA; MOSER, 2019) propõem o desenvolvimento de um sistema para a classificação de notícias falsas em português, como demonstrado na Figura 6. Este trabalho utilizou métodos de aprendizado de máquina para descobrir, classificar e armazenar *fake news* (a base de dados construída não está disponibilizada). Posteriormente, elas foram submetidas à etapa de *Extract Transform Load* (ETL), Extrair Transformar Carregar, cuja função é a extração de dados de diversos sistemas e a aplicação de um *Data Warehouse*. Para isso foi criado um *dataset* e foram avaliados os métodos de Regressão Logística, *Gradient Boosting*, *Naive Bayes* e SVM. Os experimentos realizados, segundo o autor, utilizaram apenas os títulos das notícias. Dessa forma, o algoritmo que obteve o melhor desempenho e está sendo utilizado nessa aplicação é o SVM. Após alguns testes, percebeu-se que a aplicação não funciona bem com os links das notícias do Sensacionalista (site de notícias falsas).

Figura 6 – Interface Web da Aplicação Desenvolvida



Fonte: (MONTEIRO; NOGUEIRA; MOSER, 2019).

No artigo denominado *dEFEND: Explainable Fake News Detection* (SHU et al., 2019), foi realizado um estudo da detecção explicável de notícias falsas. A proposta é utilizar um mecanismo de co-atenção para compreender e explicar uma notícia falsa. Para isso é utilizado as frases das notícias e os comentários dos usuários e assim melhorar o desempenho de detecção de notícias falsas. Para isso foi desenvolvido

uma sub-rede de co-atenção com comentários dos usuários para explorar o conteúdo das notícias e capturar conjuntamente sentenças valiosas para verificação e detecção de notícias falsas. Realizaram experiências nos conjuntos de dados disponíveis nos sites de notícias e demonstraram no seu trabalho que o método proposto não apenas superou 7 métodos avançados de detecção de notícias falsas em pelo menos 5,33% na medida-F1, mas também (simultaneamente), identificou os principais comentários dos usuários, explicando por que uma notícia é falsa.

O trabalho de (CASTELO et al., 2019) trata de um estudo para detectar notícias falsas na *web*. Os autores propõem uma nova abordagem de detecção que usa recursos independentes de tópico (*Tags*). Um banco de dados foi criado a partir da seleção de sites *Politifact*, *BuzzFeed* e *OpenSources.co*, como sites de notícias não confiáveis e 242 sites de notícias mais visitados como sites confiáveis. Os algoritmos utilizados foram: *Support Vector Machine* (SVM), *K-Nearest Neighbors* (KNN) e *Random Forest* (RF). Todos os recursos morfológicos, psicológicos e de contagem de palavras foram extraídos do banco de dados e aplicados à validação cruzada com a precisão métrica de desempenho para incrementar o algoritmo. Após os testes, os autores obtiveram uma precisão máxima de 86%, concluindo que, a partir dessa abordagem, a classificação é mais precisa com notícias de mesmo domínio.

O objetivo do trabalho do (LEAL, 2018) é o estudo e implementação de técnicas para classificação e identificação de *fake news* referentes à eleição presidencial brasileira de 2018. Na implementação, foram utilizados algoritmos de aprendizagem de máquina e mineração de dados. O *dataset* foi construído através de dados coletados do *Twitter*, com objetivo de obter mensagens e notícias referentes aos candidatos da eleição presidencial. Após o treinamento e teste dos algoritmos, os resultados obtidos atingiram uma acurácia de 90% de acertos.

Marumo (MARUMO, 2018) propõe a utilização de uma *Deep Learning* para classificação de *fake news* por sumarização de texto. Em particular, é utilizada uma base de dados com notícias verdadeiras e falsas, sendo analisados os padrões dessas notícias, adotando as técnicas de pré-processamento: Sumarização e *Word2Vec*. O objetivo desse trabalho é apresentar um modelo capaz de classificar textos entre *fake news* e notícias reais, usando Aprendizagem Profunda e Aprendizagem de Máquina Tradicional. Os autores utilizam os algoritmos *Long Short-Term Memory*, *Random Forest* e *Support Vector Machine*. Dentre os resultados obtidos, os experimentos com *Long Short-Term Memory* obtiveram a maior acurácia, com 79.3% de acertos. De acordo com os autores, o LSTM pela sua característica de possuir controle de longa memória, mostrou lidar bem quando o texto possui uma maior quantidade de dados e estão pré-processados, obtendo um valor equilibrado na predição das notícias. Ao contrário das redes neurais de avanço padrão, o LSTM possui conexões de *feedback*,

permitindo que o algoritmo mantenham a memória das entradas antigas e dos problemas do modelo no momento.

O trabalho (MONTEIRO RONEY L. DE SALES, 2018) apresenta uma aplicação *web* chamada *FakeCheck*, como demonstrado na Figura 7. Foi desenvolvido para mostrar os resultados obtidos pelo projeto Detecção Automática de Notícias Falsas para o Português (CNPq/CAPEs). O projeto visa estudar métodos para a detecção automática de notícias falsas utilizando Processamento de Linguagem Natural (PLN) e AM. Ao receber um texto, o sistema aplica métodos para extrair atributos linguísticos desse texto e os utiliza em um modelo de aprendizado de máquina, que classifica a notícia como verdadeira ou falsa. Os atributos extraídos do texto são aplicados em um classificador SVM, que infere, automaticamente, a classe da notícia (verdadeira ou falsa). Nos testes realizados em um ambiente controlado, o sistema obteve cerca de 89% de acerto (acurácia geral).

Figura 7 – Interface Web da Aplicação *FakeCheck*.

Detector de Fake News

Como funciona?

Copie o texto de uma notícia, cole na caixa abaixo e clique em "Enviar". O sistema irá processar o texto para identificar características de escrita, como palavras usadas ou classes gramaticais mais frequentes, e utilizar essas características em um modelo de aprendizado de máquina que classificará a notícia em verdadeira ou falsa. Para mais informações sobre como o sistema funciona e sua taxa de acerto, clique [aqui](#). Você também pode utilizar o nosso [bot do WhatsApp](#).

ATENÇÃO: Utilize o texto completo da notícia! O texto deve ter pelo menos 100 palavras. O sistema pode não funcionar corretamente com apenas partes de notícias.

Insira o texto da sua notícia aqui.

Fonte: <https://nilc-fakenews.herokuapp.com>

Os autores (PÉREZ-ROSAS et al., 2017) propõem um mecanismo de identificação automática de conteúdo falso online. São apresentados dois conjuntos de

dados para a tarefa de detecção de notícias falsas em inglês, cobrindo sete domínios de notícias diferentes. Além disso, os autores descrevem todo o processo de coleta, anotação e validação em detalhes e apresentam análises exploratórias sobre a identificação de diferenças linguísticas no conteúdo de notícias falsas e verdadeiras. São realizados experimentos de aprendizagem para criar detectores de *fake news* precisos, obtendo precisões de até 76%. Além disso, é realizada uma análise comparativa entre a identificação automática e manual de notícias falsas.

A Tabela 1 apresenta um comparativo dos trabalhos relacionados, dando destaque para o SIRENE. Como será apresentado na próxima seção, um dos grandes diferenciais do SIRENE é a disponibilização de uma aplicação final capaz de reconhecer as notícias falsas em português, tanto pelo conteúdo da notícia quanto por um *link*, retornando a respectiva probabilidade da classificação.

Tabela 1 – Comparação dos Trabalhos Relacionados

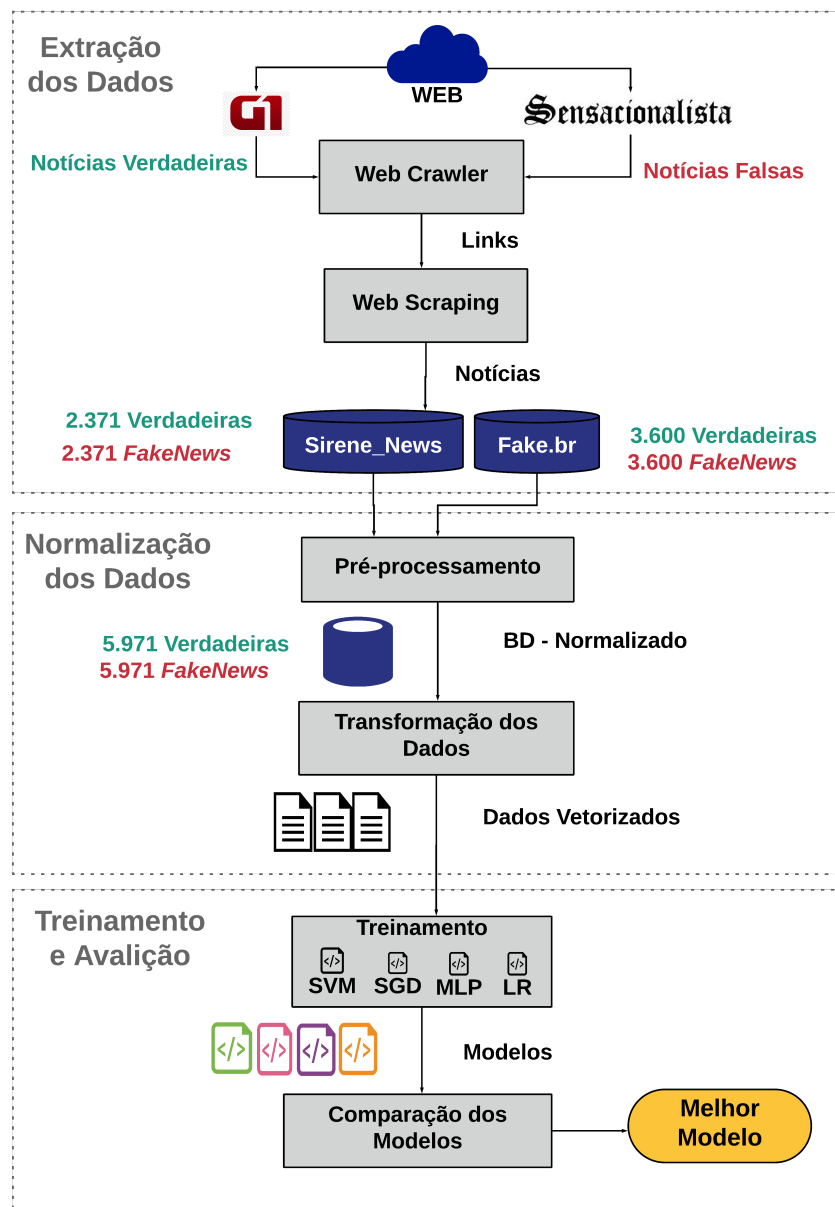
Trabalhos	Notícias em Português	Sistema Web	Classificação pelo Texto da Notícia	Classificação pelo Link da Notícia	Retornar Probabilidade
Monteiro, 2019	✓	✓		✓	
SHU, 2019		✓	✓		
Castelo, 2019			✓		
Leal, 2018	✓		✓		
Marumo, 2018	✓		✓		
FakeCheck, 2018	✓	✓	✓		
Pérez, 2017			✓		
SIRENE	✓	✓	✓	✓	✓

Fonte: Elaborado pelo autor

4 PROPOSTA

Esta seção detalha o SIRENE, uma solução inteligente capaz de classificar notícias de forma probabilística utilizando Aprendizagem de Máquina. Além disso, o trabalho disponibiliza, ao usuário final, um serviço *web* capaz de verificar a probabilidade de uma notícia ser verdadeira ou falsa. A Figura 8 detalha o fluxo utilizado neste trabalho, que foi dividido em módulos: Extração dos dados, Normalização dos dados e Treinamento e Avaliação. A partir do melhor modelo, é desenvolvido então a aplicação *web* do SIRENE. Na sequência, cada módulo será apresentado de forma mais detalhada.

Figura 8 – Visão conceitual da solução SIRENE.



Fonte: Elaborado pelo autor.

4.1 Extração dos Dados

Para formar a base de dados deste trabalho foi utilizado o *framework Scrapy* versão 1.8.0, da linguagem de programação *Python*. Com ele, é possível coletar dados de sites, redes sociais, fóruns e diversos outros canais utilizando uma linguagem simples e objetiva, sendo extremamente útil para a geração de bases de dados. Além do *Scrapy*, existem outros *frameworks* e bibliotecas capazes de fazer essas extrações, como o *beautifulsoup*¹ e o *urllib3*². Porém, estes, foram comparados com o *Scrapy* e executaram de maneira mais lenta, o processo de extração.

Os sites escolhidos para extração das notícias foram o G1³ e o Sensacionalista⁴. Estes foram escolhidos por serem grandes sites de propagação de notícias verdadeiras e falsas, respectivamente, bem como têm sido utilizados por outros trabalhos científicos na área (SANTOS et al., 2019) e (GUIMARÃES et al., 2019).

Para realizar a extração das notícias, foi criado um algoritmo em linguagem de programação *Python 3* com o *framework Scrapy*, seguindo a sequência da Figura 8. O *Web Crawling* (Rastreador da *Web*) é um *bot* que ajuda na indexação das páginas *Web*. Eles rastreiam uma página de cada vez, através de um site, até que todas as páginas tenham sido indexadas. O funcionamento de um *crawler* inicia-se com a inserção de *URLs* iniciais, chamadas *seeds* (geralmente definidas de forma manual). Ao início da sua execução, o *crawler* irá identificar e extrair todas as *URLs* existentes em *hyperlinks* nessas páginas e inseri-las em uma lista.

Com os links já extraídos, é possível então aplicar o *Web Scraping* realizando a raspagem dos dados/notícias e armazenando na base de dados. Do G1, foram extraídas 2.371 notícias (classificadas como verdadeiras) e 2.371 notícias do Sensacionalista (classificadas como *fake news*), totalizando uma base de dados com 4.742 notícias em português. Foram coletadas notícias de diversos assuntos, como política, saúde, entretenimento, esporte e assuntos gerais, de ambos os sites. Os dados extraídos foram referentes ao mês de dezembro de 2019 e registrados em um arquivo *Comma Separated Values* (CSV), que é um formato de arquivo de texto onde cada informação representa uma linha de uma planilha e cada célula é separada por “;”. Essa base de dados foi denominada *Sirene-News* e está disponível em um repositório do Github⁵.

Além das 4.742 notícias do *Sirene-News*, foi utilizado o *dataset Fake.br* (MONTEIRO et al., 2018), até então o único corpus disponível semelhante para esse idioma

¹ <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

² <https://urllib3.readthedocs.io/en/latest/>

³ <https://g1.globo.com>

⁴ <https://www.sensacionalista.com.br>

⁵ <https://github.com/ViniciusNunes0/SIRENE-news>

e que está dividido em diversos assuntos, como demonstrado na Tabela 2. Esse *dataset* possui 7.200 notícias, com exatamente 3.600 verdadeiras e 3.600 falsas. Todas as notícias estão dispostas em arquivos separados em formato de texto simples (.txt). Os dados foram extraídos em um intervalo de 2 anos, de janeiro de 2016 a janeiro de 2018, o que contribui para que a base de dados tenha notícias genéricas de diferentes períodos de tempo. Além disso, o *dataset* possui todos os links e outras informações de metadados (como autor, título, data de publicação, entre outros) disponíveis no repositório⁶. Essa quantidade de dados contribui muito para o processo de avaliação de um algoritmo, pois o fato de as notícias serem balanceadas ajuda a evitar problemas no processo de treinamento dos algoritmos.

Tabela 2 – Divisão das amostras.

	Número de amostras
Política	4.180
TV e celebridades,	1.544
Sociedade e Notícias Diárias	1.276
Ciência e Tecnologia	112
Economia	44
Religião	44

Fonte: Tabela baseado no autor ([MONTEIRO et al., 2018](#)).

Por fim, os conjunto de dados *Sirene-News* e *Fake.br* foram unidas em uma única base, totalizando 11.942 notícias. Essa base é utilizada para a experimentação dos algoritmos e na aplicação *web* do SIRENE.

4.2 Normalização dos Dados

4.2.1 Pré-processamento

Como explicado na Seção 4.1, as bases de dados do *Sirene-News* e *Fake.br* foram aglutinadas em um única base, totalizando 11.942 notícias. Após a coleta dos dados, é iniciado o pré-processamento, utilizando expressões regulares e as bibliotecas *Pandas*⁷ e *Natural Language Toolkit*⁸ (NLTK) versão 3.4.5, da linguagem de programação *Python*, a fim de eliminar campos nulos e quebras de linha, limpar os dados (retirando caracteres especiais que não contribuem com o aprendizado do algoritmo), remover algumas *stopwords* e padronizar todo o corpus do texto em minúsculo.

Além do *stopwords*, existem outras técnicas como o *stemming* e a lematização. De acordo com ([JABEEN, 2018](#)), o *stemming* é o processo de conversão de uma

⁶ <https://github.com/roneysco/Fake.br-Corpus>

⁷ <https://pandas.pydata.org/docs/index.html>

⁸ <https://www.nltk.org>

palavra para o seu radical, como por exemplo: "recuperação" e "recuperado", são reduzidos para "recupera". A lematização, diferentemente do *stemming*, reduz as palavras flexionadas para a sua raiz do idioma. Na lematização, a raiz da palavra é chamada de lema/lemas. Por exemplo: "executando", "executamos", "executam", são todas as formas da palavra "executar". Porém, ambas as técnicas causam uma perda semântica no texto. Dessa forma, essas duas técnicas não são utilizadas neste trabalho, haja vista que a semântica da notícia é fundamental.

Essa fase de pré-processamento é indispensável, pois caracteres especiais como “!?,@;#.%\$&”, influenciam no processo de análise dos textos. Por exemplo, palavras como: “importante!”, “importante,”, “importante.”, “Importante”. São vistas como palavras distintas pelo algoritmo, mesmo tendo o mesmo valor semântico, influenciando negativamente no processo de aprendizado.

4.2.2 Transformação dos Dados

Com a base de dados limpa e pré-processada, é iniciada a etapa de transformação dos dados, fase necessária para converter os dados brutos (que estão em formato de texto) para um formato numérico (interpretável pelo classificador). Após essa etapa, é aplicada a técnica chamada de *Frequência do Termo–Inverso da Frequência nos Documentos* (TF-IDF). Essa técnica é utilizada pelo fato de que algumas palavras (como artigos e preposições) são muito frequentes. Por outro lado, não costumam possuir carga semântica significativa para o processo de classificação. Então, utiliza-se o TF-IDF para limitar a importância destas palavras, de maneira que elas não causem mais influência do que o necessário.

Além do *TfidfVectorizer*, existe o *CountVectorizer*, que apenas conta as frequências da palavra em um conjunto de texto, basicamente é uma implementação do *bag of words*. Já o *TfidfVectorizer*, o valor aumenta proporcionalmente com a contagem, mas é compensado pela frequência da palavra no corpus. Isso ajuda a ajustar o fato de que algumas palavras aparecem com mais frequência. O *TfidfVectorizer* e o *CountVectorizer*, são classes do *scikit-learn* (BUITINCK et al., 2013) que servem para representação de palavras.

4.3 Treinamento e Avaliação

4.3.1 Treinamento

Após a normalização e transformação dos dados, é iniciada a fase de treinamento dos algoritmos. São utilizadas algumas implementações disponíveis no pa-

cote *Scikit-Learn* (BUTINCK et al., 2013) versão 0.20.2: o *SGDClassifier* (SGD) para o Gradiente Descendente Estocástico, a *LogisticRegression* (LR) para a Regressão Logística, o *MLPClassifier* (MLP) para o Perceptron Multicamadas e o *SVC* (*kernel* linear) para a Máquina de Vetores de Suporte.

O *LogisticRegression* do *sklearn.linear_model* é utilizado da seguinte maneira (Algoritmo 1). O *random_state* é a semente do gerador de números pseudo-aleatórios a ser usado ao embaralhar os dados. Utilizado quando o *solver* (solucionador) é igual a “saga” ou “liblinear”. O *solver* é o atributo para otimização. Além do “saga” existem o “newton-cg”, “lbfgs”, “liblinear”, o padrão desse atributo é o “lbfgs” (utilizado em problemas multiclasse). Para conjuntos de dados pequenos, o “liblinear” é melhor ao se ajustar com os dados, enquanto “saga” é mais rápido para os conjuntos de dados grandes (BUTINCK et al., 2013), que é o caso desta solução. Para problemas multiclasse, apenas “newton-cg”, “saga” e “lbfgs” lidam com perdas multinomial (com três categorias ou mais). O *multi_class*, quando a opção escolhida é “ovr”, o algoritmo resolve problemas de ordem binário (como é o caso desta solução). Se setado como “multinomial”, o algoritmo irá resolver problemas de classificação em casos que possui três ou mais classes. Os outros parâmetros ficam no modo padrão da documentação.

Algoritmo 1: Implementação do LR.

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(random_state=0, solver='saga, multi_class = 'ovr')
clf.fit(treino_vetor, classificacao_treino)
```

O *clf.fit* é a função para iniciar o treinamento do algoritmo, que recebe como parâmetros o *treino_vetor* que são as notícias vetorizadas e a *classificacao_treino* que é a lista com os rótulos 0 (verdade) ou 1 (*fakenews*).

O *SGDClassifier* do *sklearn.linear_model*, está implementada como no Algoritmo 2. O atributo *loss* mantido com o padrão “hinge”, é a função de perda a ser usada. O *penalty* é o termo de regularização. Por padrão, é “l2” por ser um regularizador muito utilizado em diversos modelos, como por exemplo, modelos baseados em SVM linear. O *alpha* é a constante que multiplica o termo de regularização (O padrão é 0,0001 == 1e-3). O *random_state* é o gerador de números aleatórios, este é inicializado da mesma forma dos outros algoritmos (ou seja, com o valor 0), para que se mantivesse um padrão. O *max_iter*, é o número máximo de passes nos dados de treinamento (também conhecidos como épocas), esse atributo afeta o comportamento no método de ajuste. Foi escolhido o valor 5, por ter retornado um resultado satisfatório no processo de teste. Vale ressaltar que quanto maior o *max_iter*, mais tempo o algoritmo levará para treinar. O *tol* é o critério de parada, ou seja, é a tolerância para os critérios de parada. Isso diz ao *Scikit-Learn* para parar de procurar um mínimo (ou máximo) quando alguma tolerância for atingida. O *tol* está configurado com

None (Nenhum), dessa forma as iterações são interrompidas somente quando *loss* (perda) $>$ *best_loss* (melhor perda) $-$ *tol* (0,0001), isso ocorre pelo número de épocas consecutivas.

Algoritmo 2: Implementação do SGD.

```
from sklearn.linear_model import SGDClassifier
clf = SGDClassifier(loss='hinge', penalty='l2', alpha=1e-3, random_state=0,
max_iter=5, tol=None)
clf.fit(treino_vetor, classificacao_treino)
```

O *MLPClassifier* do *sklearn.neural_network*, como exemplificado no Algoritmo 3, tem como primeiro atributo o *solver* “adam”, esse solucionador funciona muito bem em conjuntos de dados relativamente grandes (com milhares de amostras de treinamento ou mais) em termos de tempo de treinamento e pontuação de validação. Para conjuntos de dados pequenos, o “lbfgs” pode convergir mais rapidamente e ter um desempenho melhor. O *alpha* como “1e-3” (padrão 0.0001) com a penalidade L2, seguindo o mesmo padrão do algoritmo anterior. O *hidden_layer_sizes*, representa o número de neurônios na *i*-ésima camada oculta. Tendo em vista que quanto maior o *hidden_layer_sizes*, mais demorado é o processo de treinamento. Dessa forma, é mantido o valor como 100, evitando que o processo demore mais que o necessário. Por fim, o *random_state* iniciado com 0.

Algoritmo 3: Implementação do MLP.

```
from sklearn.neural_network import MLPClassifier
clf = MLPClassifier(solver='adam', alpha=1e-3, hidden_layer_sizes=(100, ),
random_state=0)
clf.fit(treino_vetor, classificacao_treino)
```

O *Scikit-Learn* contém a biblioteca SVM (*Support Vector Machine*), que possui classes internas para diferentes implementações. Neste trabalho é utilizado o *SVC* que é a implementação do SVM com *kernel* linear (Algoritmo 4). Dessa forma, pode-se classificar todos os dados separáveis linearmente. Além do linear, existem outras implementações como: polinomial, gaussiano e sigmóide. Todos estes foram testados e o que melhor aprendeu com os dados foi o de *kernel* linear. O exemplo que serviu como base para realizar os testes com o SVM, está disponível no tutorial realizado por (MALIK, 2018), que faz um comparativo entre todos os *kernel* disponíveis. O *probability* como *true*, é o atributo responsável por ativar a probabilidade do SVM (função essencial desta solução), o que causa uma redução da velocidade no processo de treinamento, como explicado na secção 4.4.1. O SVM não implementa diretamente a probabilidade, portanto se o *probability* estiver como *false*, ele irá classificar cada

notícia, sem calcular a probabilidade de ambas as classes.

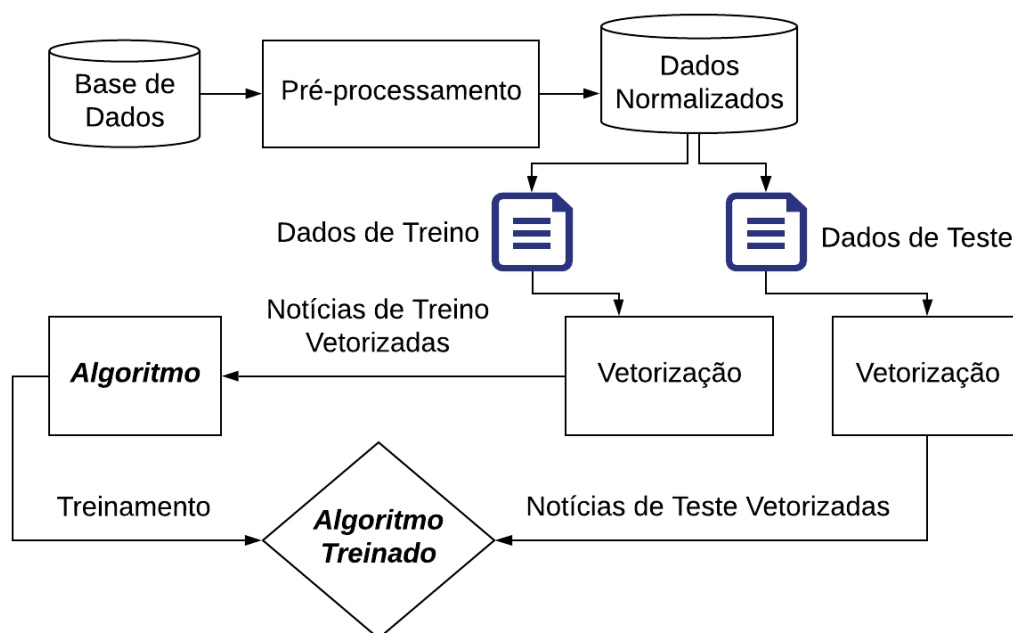
Algoritmo 4: Implementação do SVM.

```
from sklearn.svm import SVC
clf = SVC(kernel='linear', probability=True)
clf.fit(treino_vetor, classificacao_treino)
```

4.3.2 Comparação dos Modelos

O *holdout* (Figura 9) e a validação cruzada são as duas técnicas utilizadas nesse trabalho para avaliar os algoritmos LR, SGD, MLP e SVM, junto com a matriz de confusão, objetivando analisar e escolher o algoritmo que melhor se adaptou aos dados.

Figura 9 – Técnica de *holdout* para comparação dos algoritmos.



Fonte: Elaborado pelo autor.

O *holdout* é a técnica de divisão do conjunto de dados em treino e teste. Além disso, é uma técnica rápida para testar os algoritmos. Uma divisão comum ao usar o método de *holdout* é dividir os dados em 70% para treinamento e 30% restantes para teste, por ser um padrão utilizado na literatura (RASCHKA, 2018). Porém, essa divisão depende da quantidade de dados no *dataset* e da maneira de avaliação. O conjunto de treinamento é utilizado para gerar o modelo (algoritmo treinado e capaz de classificar) e o conjunto de testes é usado para ver o desempenho do modelo com os dados não vistos (dados novos). Os dados então pré-processados e transformados,

são submetidos aos algoritmos. Após isso, é possível analisar os resultados de cada um dos modelos gerados, através dos rótulos, informando a classificação que cada notícia se encaixa.

Os dados obtidos estão organizados em uma matriz de confusão/contingências (Tabela 3), que é uma tabela que mostra as frequências de classificação para cada classe do modelo, o que ajuda no processo de avaliação, pois com ela, é possível pegar os dados retornados dos algoritmos e assim analisar o rendimento de cada um dos modelos gerados. Essa matriz é uma forma de organizar os resultados dos testes positivos e negativos em positivos (TP e FP) e negativos (TN e FN).

Tabela 3 – Matriz de Confusão

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

Fonte: Elaborado pelo autor.

- Verdadeiro positivo (VP): todos os dados da classe positivo que foram classificados corretamente.
- Verdadeiro negativo (VN): todos os dados da classe negativo que foram classificados corretamente.
- Falso positivo (FP): todos os dados que eram da classe negativo e que foram classificados como positivo.
- Falso negativo (FN): todos os dados que eram da classe positivo e que foram classificados como negativo.

Para avaliar os resultados obtidos, são utilizadas as métricas de acurácia, precisão, *recall* e medida-f (F1). Esses valores são apresentados em detalhes no Capítulo 5. Suas fórmulas se encontram nas seguintes equações:

$$Acurácia = (VP + VN)/Total \quad (4.1)$$

$$Precisão = VP/(VP + FP) \quad (4.2)$$

$$Recall = VP/(VP + FN) \quad (4.3)$$

$$F1 = 2 * ((Precisão * Recall)/(Precisão + Recall)) \quad (4.4)$$

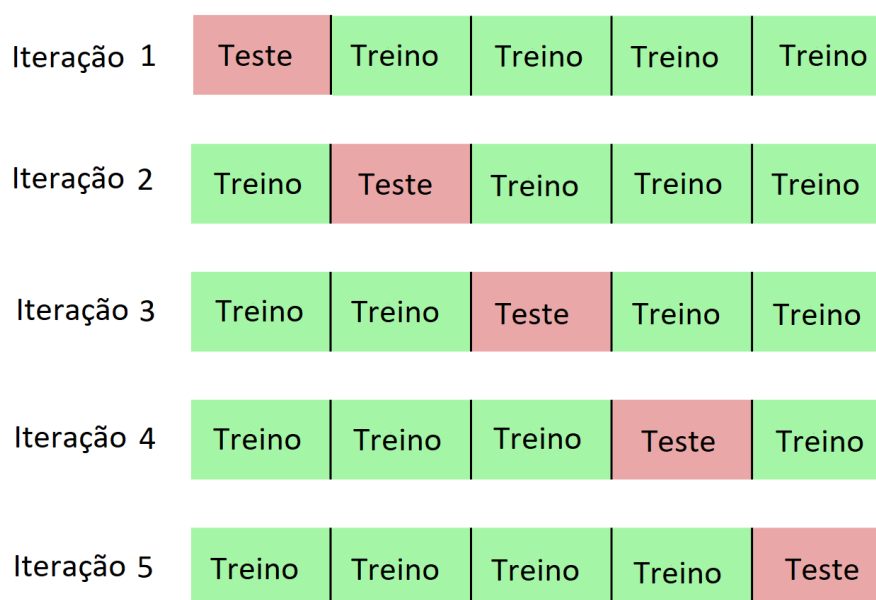
A acurácia (4.1) indica uma performance geral do modelo, onde dentre todas as classificações, quantas o modelo classificou corretamente. A precisão (4.2) dentre todas

as classificações da classe Positivo que o modelo fez, quantas estão corretas. O *recall* (4.3), dentre todas as situações de classe Positivo como valor esperado, quantas estão corretas. A medida-f (4.4) é a média harmônica entre precisão e *recall*.

Além do *holdout*, é utilizada a validação cruzada (Figura 10), essa técnica avalia a capacidade de generalização de um modelo, a partir de um conjunto de dados. Ela é amplamente empregada em problemas onde o objetivo da modelagem é a predição. A validação cruzada, ou *k-fold cross-validation*, ocorre quando o conjunto de dados é dividido aleatoriamente em K partições. Um dos grupos é usado como o conjunto de testes e o restante é usado como o conjunto de treinamento. O modelo é treinado no conjunto de treinamento e pontuado no conjunto de teste. Em seguida, o processo é repetido até que cada grupo exclusivo seja usado como o conjunto de testes.

Por exemplo, para uma validação cruzada com 5 iterações, o conjunto de dados é dividido em 5 grupos e o modelo é treinado e testado 5 vezes em separado, para que cada grupo tenha a chance de ser o conjunto de testes, como demonstrado na Figura 10.

Figura 10 – Validação cruzada com 5 iterações.



Fonte: Elaborado pelo autor.

Como a validação cruzada usa várias divisões de treino e teste, é preciso mais poder computacional e tempo para executar do que usar o método de *holdout*. Os resultados detalhados estão no Capítulo 5.

4.4 SIRENE - Aplicação Web

Após o processo de comparação dos resultados, foi selecionado o melhor modelo. Após isso, a biblioteca do *Python Pickle*⁹, é utilizada para serializar o modelo em memória para um arquivo e com isso, armazenar no sistema de arquivos para que possa ser utilizado no módulo *web* do SIRENE. Esse modelo gerado é o algoritmo já treinado capaz de classificar. Em seguida, foi iniciado o desenvolvimento da aplicação *web* do SIRENE utilizando o *Flask*¹⁰ (*microframework web* escrito em *Python* para desenvolvimento *web*). O *Flask* foi projetado para facilitar e acelerar o desenvolvimento de aplicações e possui a capacidade de expandir para aplicações mais complexas. Além disso, o *Flask* oferece sugestões, mas não impõe nenhuma dependência ou layout do projeto. Cabe ao desenvolvedor escolher as ferramentas e bibliotecas que deseja usar.

Figura 11 – Interface do SIRENE (Web).



The screenshot displays the SIRENE web interface. At the top center is the SIRENE logo, which consists of the word "SIRENE" in a bold, black, sans-serif font, followed by a red graphic element resembling a stylized 'S' or a wave. Below the logo is an observation: "OBSERVAÇÃO: Utilize o texto completo da notícia! pode não funcionar corretamente com apenas partes de notícias." Underneath this is a form with two input fields. The first field is a text input with the placeholder text "Informe o Link da Notícia...". The second field is a larger text area with the placeholder text "ou informe a notícia...". At the bottom center of the form is a blue button with the white text "Enviar".

Fonte: Elaborado pelo autor.

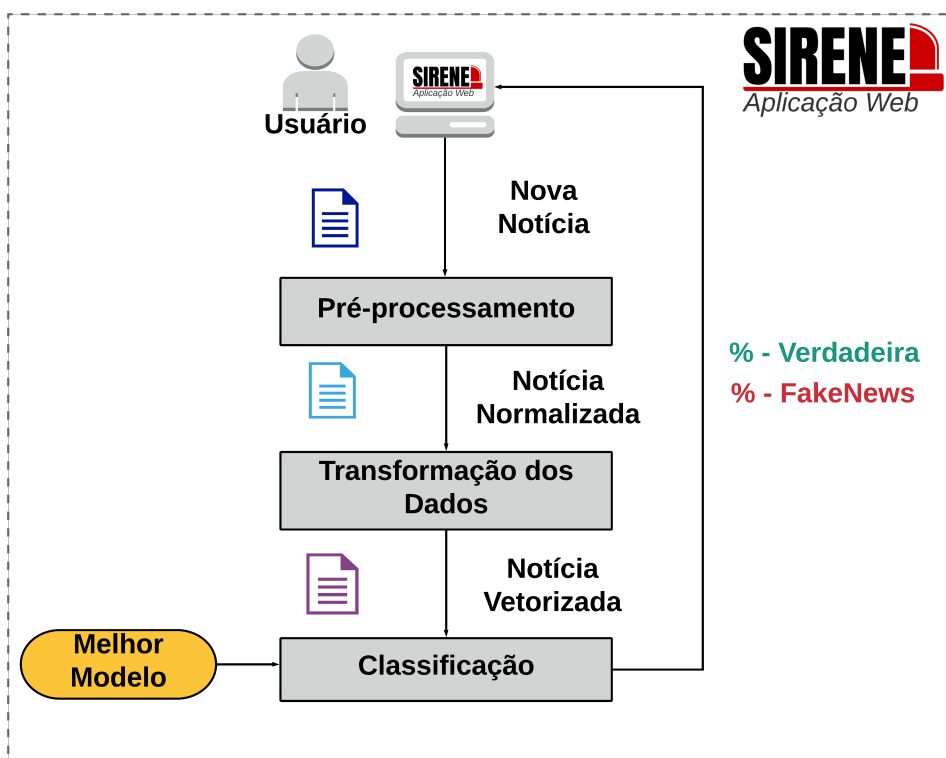
⁹ <https://docs.python.org/3/library/pickle.html>

¹⁰ <https://flask.palletsprojects.com/en/1.1.x/>

A aplicação *web* do SIRENE é composta por dois campos de *input*, o primeiro recebe o *link* da notícia, que passa pelo processo de raspagem (extraíndo a notícia) para ser classificada. O segundo recebe a notícia em si, que corresponde ao texto integral. Como demonstrado na Figura 11, esses dois campos permitem que o usuário possa verificar a notícia através do *link* ou do próprio texto na aplicação *web* do SIRENE.

Após o usuário submeter a notícia no sistema, é então iniciada a fase do pré-processamento da aplicação *web* SIRENE (Figura 12). Para que a notícia seja classificada com êxito, é necessário realizar os mesmos passos de pré-processamento e transformação, antes feitos no processo de treinamento, para que o algoritmo siga um padrão e possa então compreender a nova notícia e por fim, classificá-la. Nessa primeira etapa, a notícia é limpa a fim de eliminar quebras de linha, caracteres especiais (que não contribuem com a classificação), *stopwords* e padronizar o texto em minúsculo. Em seguida, a notícia é transformada em números com o TF-IDF e então submetida ao modelo que retorna a classificação “VERDADEIRO” ou “FAKENEWS” junto com a probabilidade para o usuário.

Figura 12 – Módulo da Aplicação Web SIRENE.



Fonte: Elaborado pelo autor.

4.4.1 Probabilidade do SVM

Apesar do SVM possuir inúmeras vantagens, tais como eficácia em espaços multidimensionais e versatilidade na escolha do *kernel*, ele não provê diretamente estimativas de probabilidade. Logo, para obter os valores probabilísticos em problemas binários, recorre-se a um modelo de regressão para problemas binários chamado escala de Platt (PLATT, 1999).

A Escala Platt é uma maneira de transformar as saídas de um modelo de classificação em uma distribuição de probabilidade sobre as classes. Este método, treina os parâmetros de uma função sigmóide/logística para mapear uma resposta da SVM usando probabilidade, inalterando seu funcionamento. Trata-se de um modelo paramétrico para ajustar a probabilidade condicional.

Considere o problema de classificação binária: para a entrada x_i , pretende-se determinar se ele pertence a uma das duas classes, arbitrariamente rotulados $+1$ e -1 . Para obter a probabilidade é utilizada $P(y = +1|f(x_i))$, ou seja, uma classificação que não só dá uma resposta, mas também um grau de certeza sobre a resposta. Por meio da Regra de Bayes para obtenção da probabilidade a posteriori e considerando que as condicionais são funções exponenciais, Platt define a seguinte relação:

$$p(x_i) = P(y = 1|f(x_i)) = \frac{1}{1 + \exp(Af(x_i) + B)}. \quad (4.5)$$

Os parâmetros A e B são dois parâmetros escalares que o algoritmo aprendeu. Nota-se que as previsões podem agora ser feitos de acordo com $y = 1$ (classe positiva) ou $y = -1$ (classe negativa), onde y representa o label do SVM.

Esses parâmetros são ajustados utilizando os dados de treinamento $(f(x_i), t_i)$, onde $t_i = \frac{(y_i+1)}{2}$, sendo $y_i = +1$, então $t_i = 0$ ou $y_i = -1$ e $t_i = 1$, ou seja, a intenção dessa fórmula é converter a escala de -1 a 1 (SVM) para escala de 0 a 1 (probabilidade). Obtêm-se estes parâmetros pela minimização do logaritmo negativo da função de máxima verossimilhança, detalhada no artigo (PLATT, 1999).

Para evitar o *overfitting* (o sobreajuste dos dados), é utilizado a validação cruzada de 5 vezes (*5-fold cross validation*), que basicamente faz uma regressão para saber qual seria a probabilidade originária do padrão classificado. Dessa forma, a probabilidade retornada é a melhor possível, evitando que o algoritmo retorne constantemente 100% de certeza sobre uma classe.

5 RESULTADOS

Este capítulo apresenta os resultados dos experimentos realizados no capítulo 4. Inicialmente, são realizadas análises na base de dados, para entender os padrões existentes entre as classes verdadeiro e *FakeNews*. Depois dessa análise da base, é realizado o comparativo entre os algoritmos, utilizando as métricas de acurácia, precisão, *recall* e medida-f. Por fim, é demonstrada a aplicação final do SIRENE, comprovando a eficiência da proposta.

5.1 Análise da Base de Dados

Após alguns testes, foi possível extrair estatísticas das amostras contidas no conjunto de dados *Sirene-News* e *Fake.br*, conforme mostrado nas Tabelas 4 e 5. Os dados mostram que as notícias verdadeiras apresentam um número maior de palavras e sentenças em comparação às notícias falsas. Isso devido ao fato da notícia verdadeira, basear-se em informações reais contendo mais fundamento do que as notícias falsas que buscam influenciar ou ridicularizar a imagem de alguém.

Tabela 4 – Informações dos dados Sirene-News.

	Falsa	Verdadeira
Média de palavras por notícia	142.66	230.73
Média de sentenças por notícia	8.63	13.53

Fonte: Elaborado pelo autor.

Tabela 5 – Informações dos dados Fake.br.

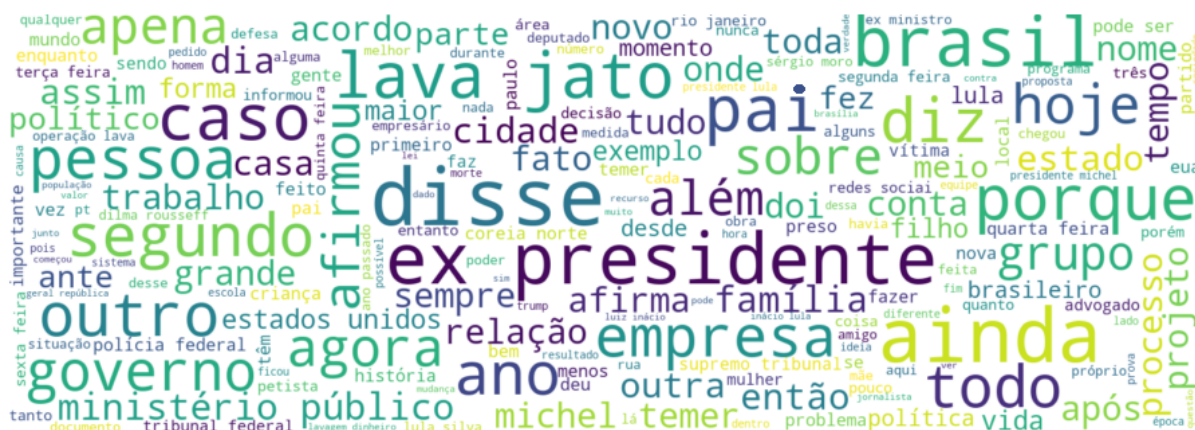
	Falsa	Verdadeira
Média de palavras por notícia	206.40	427.95
Média de sentenças por notícia	10.68	21.20

Fonte: Elaborado pelo autor.

Para ter uma noção das palavras que mais se repetem nas bases de dados, utilizou-se a biblioteca *WordCloud*¹ (que gera uma nuvem de palavras). Na Figura 13 é possível perceber que as palavras “ex-presidente”, “lava jato”, “governo” e “Brasil” são as que mais se repetem e estão em destaque, caracterizando que grande parte das notícias, tanto verdadeiras quanto falsas, foca no assunto “Política”.

¹ https://amueller.github.io/word_cloud/

Figura 13 – Nuvem de palavras.



Fonte: Elaborado pelo autor.

5.2 Comparação dos Algoritmos

Inicialmente, foi realizado um comparativo entre os algoritmos *Stochastic Gradient Descent* (SGD), *Logistic Regression* (LR), *Multilayer Perceptron* (MLP) e *Support Vector Machine* (SVM) para que fosse selecionado o algoritmo que mais se adequasse com os dados. Para a aplicação dos testes, as bases de dados *Sirene-News* e *Fake.br*, foram divididas em 70% para treinamento e 30% para teste, o que corresponde respectivamente a 8.360 notícias de treinamento e 3.582 notícias de teste. Com a base de dados dividida, pré-processada e transformada, as notícias de treinamento foram aplicadas para cada algoritmo. Um exemplo desse pré-processamento está descrito nas Tabelas 6 e 7.

Tabela 6 – Exemplo de uma notícia Original.

Notícia
Filipe Toledo sofreu uma derrota dura em Pipeline. Com a queda no round 3 da última etapa da temporada, nesta terça-feira, no Havaí, o paulista deu adeus às chances de conquistar o seu primeiro título mundial e de levar uma das duas vagas do Brasil na Olimpíada de Tóquio 2020. Ítalo Ferreira e Gabriel Medina garantiram a classificação para o Japão e também para as oitavas de final do Pipe Masters. O brasileiros venceram seus confrontos e seguem na briga pelo título mundial com o americano Kolohe Andino. Terceiro lugar do ranking, o sul-africano Jordy Smith também foi eliminado na terceira fase pelo paulista Jessé Mendes e saiu da disputa. Na primeira bateria do dia, Ítalo derrotou o amigo e conterrâneo Jadson André em um duelo acirrado.

Fonte: Elaborado pelo autor.

Após o treinamento, cada algoritmo gerou seu modelo, que foi submetido aos dados de teste para, então, ser avaliado. Após isso, foi utilizada a matriz de confusão,

Tabela 7 – Exemplo de uma notícia com o pré-processamento.

Notícia
<p>filipe toledo sofreu derrota dura pipeline queda round 3 última etapa temporada nesta terça feira havaí paulista deu adeus chances conquistar primeiro título mundial levar duas vagas brasil olimpíada tóquio 2020 ítalo ferreira gabriel medina garantiram classificação japão oitavas final pipe masters brasileiros venceram confrontos seguem briga título mundial americano kolohe andino terceiro lugar ranking sul africano jordy smith eliminado terceira fase paulista jessé mendes saiu disputa na primeira bateria dia ítalo derrotou amigo conterrâneo jadson andré duelo acirrado</p>

Fonte: Elaborado pelo autor.

pois com ela é possível pegar os valores retornados pelos algoritmos e calcular as suas respectivas: acurácia, precisão, *recall* e medida-f. Os resultados obtidos de cada algoritmo encontram-se na Tabela 8.

Tabela 8 – Resultado dos Algoritmos.

Algoritmos	Acurácia	Precisão		<i>Recall</i>		Medida-f	
		Verdadeiro	FakeNews	Verdadeiro	FakeNews	Verdadeiro	FakeNews
SGD	92,90%	92,51%	93,29%	93,24%	92,57%	92,88%	92,93%
LR	94,30%	94,71%	93,91%	93,75%	94,84%	94,23%	94,37%
MLP	95,14%	94,60%	95,68%	95,66%	94,62%	95,13%	95,15%
SVM	96,39%	95,38%	96,40%	95,72%	95,45%	96,35%	96,43%

Os valores da Tabela 8 mostram que os algoritmos obtiveram dados relativamente próximos entre si, visto que o SGD, LR e SVM são algoritmos lineares de classificação. Ambos traçam uma linha separando os dados de classe binária, como é o caso do SIRENE. Já o *MLPClassifier*, que segue um modelo de aproximação por função não linear, obteve valores próximos do SVM. Diferindo-se da regressão logística, o MLP pode obter uma ou mais camadas não lineares, chamadas de camadas ocultas, entre a camada de entrada e a de saída. As redes neurais têm sido aplicadas com sucesso em diversos problemas.

Os algoritmos alcançaram uma precisão acima dos 90% no processo de classificação, tanto das notícias reais quanto falsas, o que é um ponto positivo na implementação desta solução. A medida-f, que é a média harmônica entre precisão e *recall*, atingiu valores altos, o que significa que a acurácia obtida é relevante. Ou seja, os valores de Verdadeiro Positivo, Verdadeiro Negativo, Falso Positivo e Falso Negativo aferidos não apresentam grandes distorções. Essa medida também pode ser entendida como uma medida de confiabilidade da acurácia, mostrando que os algoritmos atingiram valores satisfatórios.

Os valores de cada matriz de confusão estão demonstradas nas tabelas a se-

guir:

Tabela 9 – Matriz de Confusão LR.

LR		
Classe Real	Classe Predita	
	Verdadeiro	<i>FakeNews</i>
Verdadeiro	1666	111
<i>FakeNews</i>	93	1712

Tabela 10 – Matriz de Confusão SGD.

SGD		
Classe Real	Classe Predita	
	Verdadeiro	<i>FakeNews</i>
Verdadeiro	1657	120
<i>FakeNews</i>	134	1671

Tabela 11 – Matriz de Confusão MLP.

MLP		
Classe Real	Classe Predita	
	Verdadeiro	<i>FakeNews</i>
Verdadeiro	1700	77
<i>FakeNews</i>	97	1708

Tabela 12 – Matriz de Confusão SVM.

SVM		
Classe Real	Classe Predita	
	Verdadeiro	<i>FakeNews</i>
Verdadeiro	1714	63
<i>FakeNews</i>	62	1743

Dentre os algoritmos testados (Tabela 10), o SGD teve a menor taxa de acertos, obtendo somente 3.328 notícias classificadas corretamente (classe Verdadeiro e *FakeNews*). O SVM (Tabela 12), é o algoritmo que classificou mais notícias de forma correta, totalizando assim, 3.457 notícias de um total de 3.582 submetidas ao algoritmo.

Tabela 13 – Tempo de execução no processo de treinamento.

	Tempo
LR	46s
SGD	53s
SVM	75s
MLP	87s

Fonte: Elaborado pelo autor.

O tempo para execução do treinamento (Tabela 13) depende do processamento da máquina que está sendo executada e do volume de dados que está sendo submetido ao algoritmo. Como dito anteriormente, foram submetidas 8.360 notícias para o treinamento. Os algoritmos executaram em um tempo excelente, levando em conta o volume de dados.

Além da técnica de *holdout*, os quatro algoritmos foram submetidos a técnica de *cross validation* (do português, validação cruzada), cujo valor de K (número de iterações) é igual a 10. Ou seja, os dados foram divididos em 10 grupos dentro da base de dados. Inicialmente, o valor de k era 5, porém, o valor de acurácia de cada algoritmo estava muito próximo o que dificultava uma melhor visualização e comparação entre os algoritmos. Portanto, o valor de k foi aumentado até chegar em 10. Dessa forma, a cada iteração o algoritmo recebe 10.747 notícias para treino e 1.195 para

teste. É importante que a escolha do k permita que o tamanho de cada partição de validação seja grande o suficiente para fornecer uma estimativa justa do desempenho do modelo. Ao mesmo tempo, o k não deve ser muito pequeno, de forma que haja modelos treinados suficientes para avaliar.

É utilizada a validação cruzada para obter um panorama da acurácia de cada um dos algoritmos, dividindo a base em partes de treinamento e teste. Dessa forma, cada grupo tem a chance de ser o conjunto de teste. Os resultados obtidos estão na Tabela 14.

Tabela 14 – Resultados da Validação Cruzada.

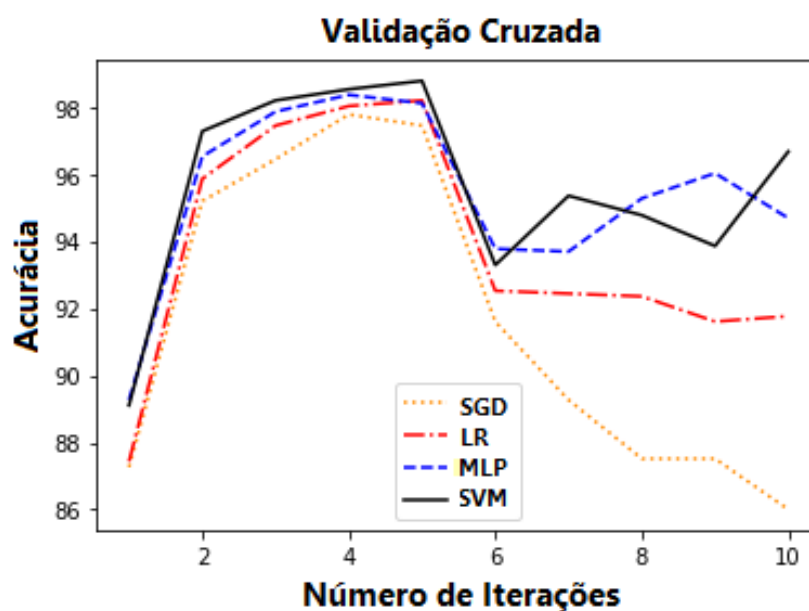
	SGD	LR	MLP	SVM
Iteração 1	87,26%	87,43%	89,27%	89,11%
Iteração 2	95,22%	95,89%	96,56%	97,31%
Iteração 3	96,48%	97,48%	97,90%	98,24%
Iteração 4	97,82%	98,07%	98,40%	98,57%
Iteração 5	97,48%	98,24%	98,15%	98,82%
Iteração 6	91,62%	92,54%	93,80%	93,31%
Iteração 7	89,27%	92,46%	93,71%	95,38%
Iteração 8	87,52%	92,37%	95,30%	94,80%
Iteração 9	87,52%	91,62%	96,06%	93,88%
Iteração 10	86%	91,78%	94,71%	96,71%
Média	91,62%	93,79%	95,39%	95,61%

Fonte: Elaborado pelo autor.

Conforme pode ser visto na Tabela 14, os algoritmos obtiveram uma média de acurácia superior a 90% o que é considerado um excelente resultado para problemas de classificação, levando em conta o quanto a língua portuguesa é complexa. O algoritmo SVM teve o melhor resultado (95,61%), comparando-se aos demais algoritmos.

Além da Tabela 14, para facilitar a visualização dos algoritmos ao decorrer de cada iteração, seus respectivos valores foram plotados no gráfico, demonstrado na Figura 14. Percebe-se que todos os valores, até a iteração 6, mantiveram-se próximos entre si, e a partir dela os valores de cada um dos modelos começaram a se distanciar. Isso ocorre devido à mudança dos dados entre as bases de dados, pois nas primeiras iterações estava utilizando a base do *Sirene-news* e a partir da 6^a começa a mudança para a base de dados do *Fake.br* (MONTEIRO et al., 2018), cujos dados foram extraídos em períodos diferentes, o que causa essa diferença entre cada modelo. O SGD teve uma redução drástica, comparada aos outros chegando a uma taxa de acertos inferior a 90%. O SVM e o MLP tiveram valores bem próximos a cada iteração, porém, levando em conta a média entre eles, o SVM foi melhor.

Figura 14 – Gráfico comparativo dos algoritmos.



Fonte: Elaborado pelo autor.

5.3 Aplicação Web

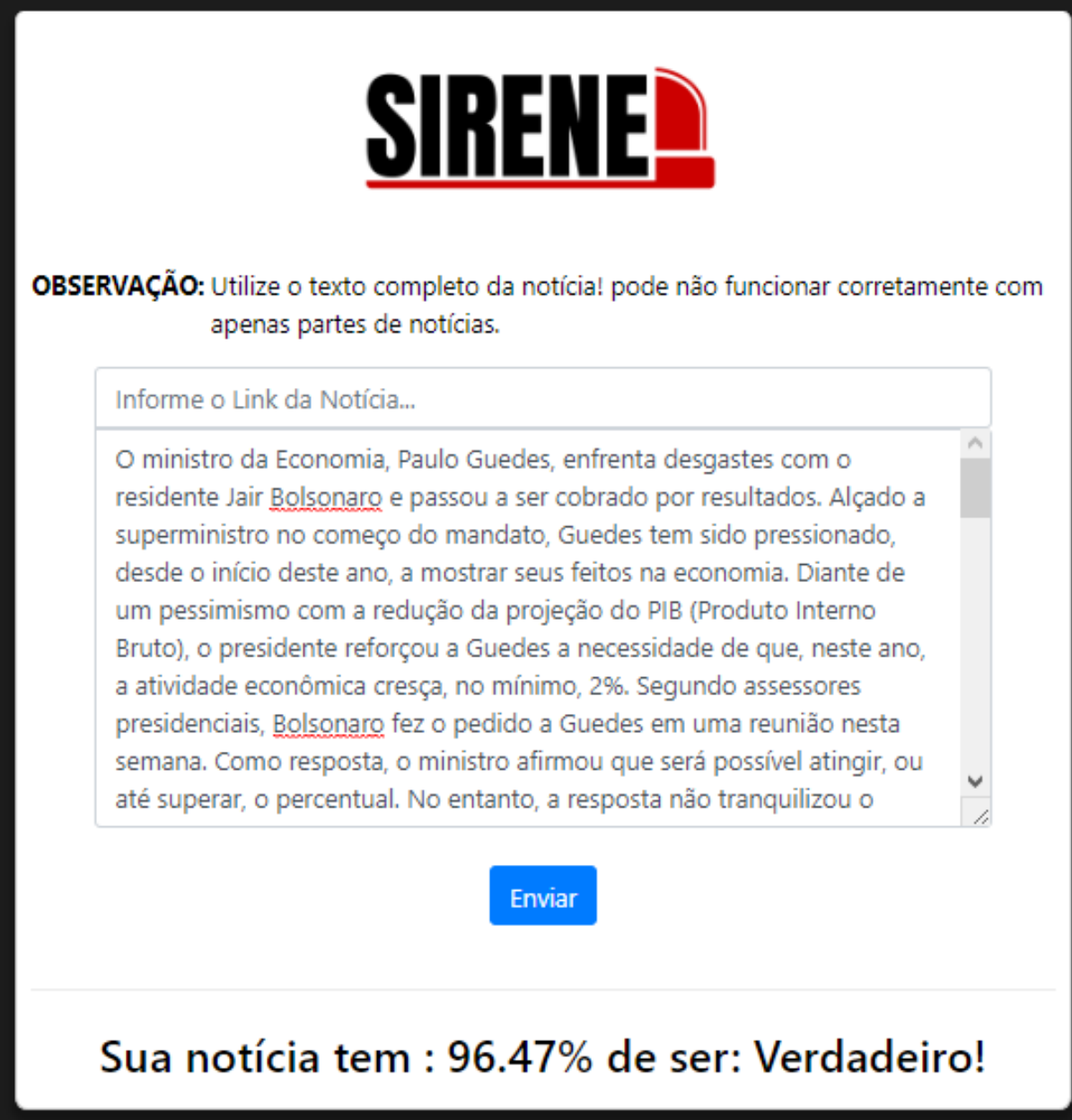
Dos resultados obtidos tanto o MLP quanto SVM obtiveram valores altos e próximos entre si. Porém, o modelo que obtém o melhor resultado, sendo então escolhido para a implementação da proposta, é o SVM. Além disso, o SVM mostrou-se bastante recorrente nos trabalhos relacionados (Capítulo 3). Portanto, após o processo de avaliação dos algoritmos, toda a base de dados foi submetida ao algoritmo SVM, que gerou um modelo e este foi aplicado ao sistema *web* do SIRENE². A classificação por notícia está representada na Figura 15, essa notícia (verdadeira) foi extraída do UOL³ para demonstrar como é o retorno da classificação. Caso a notícia possua uma quantidade inferior a 100 palavras, o modelo tende a classificar como *fake news*.

Por fim, o SIRENE tem seu grande diferencial por ser uma aplicação *web* capaz de classificar as notícias em verdadeira ou *fakenews*, utilizando tanto o conteúdo da notícia quanto o seu respectivo *link*, retornando a probabilidade da classificação. Função esta, que existe somente na aplicação *web* do SIRENE se comparado com as demais aplicações existem para esse fim.

² <http://lar.ifce.edu.br:5000>

³ <https://www1.folha.uol.com.br/mercado/2020/02/bolsonaro-cobra-guedes-a-entregar-crescimento-minimo-de-2-neste-ano.shtml>

Figura 15 – Classificação pela Notícia.



SIRENE

OBSERVAÇÃO: Utilize o texto completo da notícia! pode não funcionar corretamente com apenas partes de notícias.

Informe o Link da Notícia...

O ministro da Economia, Paulo Guedes, enfrenta desgastes com o residente Jair Bolsonaro e passou a ser cobrado por resultados. Alçado a superministro no começo do mandato, Guedes tem sido pressionado, desde o início deste ano, a mostrar seus feitos na economia. Diante de um pessimismo com a redução da projeção do PIB (Produto Interno Bruto), o presidente reforçou a Guedes a necessidade de que, neste ano, a atividade econômica cresça, no mínimo, 2%. Segundo assessores presidenciais, Bolsonaro fez o pedido a Guedes em uma reunião nesta semana. Como resposta, o ministro afirmou que será possível atingir, ou até superar, o percentual. No entanto, a resposta não tranquilizou o

Enviar

Sua notícia tem : 96.47% de ser: Verdadeiro!

Fonte: Elaborado pelo autor.

6 CONCLUSÕES

A necessidade do combate às *fake news* e suas consequências na sociedade, que influenciam negativamente o pensamento e a tomada de decisão das pessoas, foram a motivação inicial para o desenvolvimento desse trabalho. Ou seja, a ideia principal é tornar possível a detecção desse tipo de notícia. A solução proposta utiliza o algoritmo SVM, que obteve resultados satisfatórios, para criar um modelo capaz de analisar os padrões dos textos pré-processados e detectar as notícias falsas.

Através do desenvolvimento desse trabalho, foi possível validar a utilização de aprendizagem de máquina para a identificação e classificação das *fake news*. O aprendizado de máquina tem sido utilizado na resolução de diversos problemas da sociedade. Durante o desenvolvimento do trabalho. Percebeu-se que, no Brasil, essa técnica ainda é pouco explorada em comparação a outros países como os Estados Unidos.

Analisando os algoritmos em sua totalidade, obteve-se uma acurácia acima dos 90% nos quatro algoritmos testados. Algumas dificuldades foram encontradas durante o desenvolvimento, como a coleta dos dados. Parte desse processo foi realizado de forma manual, sendo necessário analisar os sites cujas notícias seriam extraídas. Além disso, na língua portuguesa, existem poucos sites que divulgam abertamente as *fake news*, dificultando o processo da construção da base de dados. Foi fundamental para os resultados obtidos nesse trabalho a utilização do corpus disponibilizado por (MONTEIRO et al., 2018), pois possuem um bom volume de dados no *dataset*. Adicionalmente, esse *dataset* já encontrava-se balanceado, evitando, assim, mais trabalho no processo de treinamento dos algoritmos. Por fim, a base de dados construída na solução do SIRENE está disponível para futuros trabalhos no GitHub¹.

Como trabalhos futuros, espera-se aumentar a base de dados (com mais notícias verdadeiras e falsas), automatizar o processo de coleta e monitoramento dos sites de notícias, melhorar o modelo gerado pelo algoritmo SVM e aprimorar a aplicação *web* do SIRENE. Além disso, aprimorar os algoritmos para possibilitar a detecção de *fake news* com pouco conteúdo textual, que tem sido a principal estratégia usada pelos usuários que publicam esse tipo de notícia, na tentativa de burlar os mecanismos de detecção automática.

¹ <https://github.com/ViniciusNunes0/SIRENE-news>

REFERÊNCIAS

ARANHA, C.; PASSOS, E. A tecnologia de mineração de textos. *Revista Eletrônica de Sistemas de Informação*, v. 5, n. 2, 2006. Citado na página 19.

BAKIR, V.; MCSTAY, A. Fake news and the economy of emotions: Problems, causes, solutions. *Digital journalism*, Taylor & Francis, v. 6, n. 2, p. 154–175, 2018. Citado na página 18.

BARBOSA, V. N.; OLIVEIRA, C. T. D.; BRAGA, R. B. AuFa -Automatic Detection and Classification of Fake News Using Neural networks. In: MENA, F. M. et al. (Ed.). *8th International Workshop on ADVANCEs in ICT Infrastructures and Services (ADVANCE 2020)*. Cancún, Mexico, 2020. (Proc. of the 8th International Workshop on ADVANCEs in ICT Infrastructures and Services (ADVANCE 2020)), p. 1–8. Disponível em: <<https://hal.archives-ouvertes.fr/hal-02495259>>. Citado na página 15.

BASU, S.; BANERJEE, A.; MOONEY, R. Semi-supervised clustering by seeding. In: CITESEER. *In Proceedings of 19th International Conference on Machine Learning (ICML-2002)*. [S.l.], 2002. Citado na página 22.

BATISTA, R. *A divulgação de notícias falsas, conhecidas como fake news, pode interferir negativamente em vários setores da sociedade, como política, saúde e segurança*. 2018. Acessado em 20/12/2019. Disponível em: <<https://mundoeducacao.bol.uol.com.br/curiosidades/fake-news.htm>>. Citado na página 13.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: ACM. *Proceedings of the fifth annual workshop on Computational learning theory*. [S.l.], 1992. p. 144–152. Citado na página 29.

BRUCE, R. F. A bayesian approach to semi-supervised learning. In: *NLPRS*. [S.l.: s.n.], 2001. p. 57–64. Citado na página 22.

BUITINCK, L. et al. API design for machine learning software: experiences from the scikit-learn project. In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. [S.l.: s.n.], 2013. p. 108–122. Citado 2 vezes nas páginas 38 e 39.

CABRAL, B. C. P. et al. Construção de exercício para verificação do aprendizado utilizando redes neurais artificiais-fase ii medula espinhal. In: SPRINGER. *IV Latin American Congress on Biomedical Engineering 2007, Bioengineering Solutions for Latin America Health*. [S.l.], 2007. p. 1221–1224. Citado na página 27.

CASTELO, S. et al. A topic-agnostic approach for identifying fake news pages. In: ACM. *Companion Proceedings of The 2019 World Wide Web Conference*. [S.l.], 2019. p. 975–980. Citado na página 32.

CIRIACO, D. *Mais de 4 bilhões de pessoas usam a internet ao redor do mundo*. 2018. Acessado em 20/12/2019. Disponível em: <<https://www.tecmundo.com.br/internet/126654-4-bilhoes-pessoas-usam-internet-no-mundo.htm>>. Citado na página 13.

CLAÚDIO, E.; CAMPI, M. Uso da inteligência artificial para identificação de fraudes financeiras: uma proposição de metodologias e ferramentas para detecção. 2018. Pós-graduação (Transformação Digital e Gestão de TI), Faculdade IETEC. Disponível em: <<https://www.ietec.com.br/clipping/2018/08-agosto/Usoda-inteligencia-artificial-para-identificacao-de-fraudes-financeiras.pdf>>. Citado na página 24.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine learning*, Springer, v. 20, n. 3, p. 273–297, 1995. Citado na página 29.

D'AGOSTINO, R. *Três anos depois, linchamento de Fabiane após boato na web pode ajudar a endurecer lei*. 2017. Acessado em 20/12/2019. Disponível em: <<https://g1.globo.com/e-ou-nao-e/noticia/tres-anos-depois-linchamento-de-fabiane-apos-boato-na-web-pode-ajudar-a-endurecer-lei.ghtml>>. Citado na página 14.

DEEPLARNINGBOOK. *Aprendizado Com a Descida do Gradiente*. 2019. Acessado em 20/12/2019. Disponível em: <<http://deeplearningbook.com.br/aprendizado-com-a-descida-do-gradiente/>>. Citado na página 25.

DELMAZO, C.; VALENTE, J. C. Fake news nas redes sociais online: propagação e reatificação desinformação em busca de cliques. *Media Jornalismo*, scielopt, v. 18, p. 155 – 169, 04 2018. ISSN 2183-5462. Disponível em: <http://www.scielo.mec.pt/scielo.php?script=sci_arttextpid=S2183-54622018000100012nrm=iso>. Citado na página 13.

DIZIKES, P. *Study: On Twitter, false news travels faster than true stories: Research project finds humans, not bots, are primarily responsible for spread of misleading information*. 2018. Acessado em 20/12/2019. Disponível em: <<http://news.mit.edu/2018/study-twitter-false-news-travels-faster-true-stories-0308>>. Citado na página 18.

DRUCKER, H. et al. Support vector regression machines. In: *Advances in neural information processing systems*. [S.l.: s.n.], 1997. p. 155–161. Citado na página 28.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37–37, 1996. Citado 2 vezes nas páginas 19 e 20.

GEOINFORMAÇÃO. *Gradiente descendente*. 2019. Acessado em 20/12/2019. Disponível em: <http://cursos.leg.ufpr.br/ML4all/apoio/Gradiente.html_gradiente_descendente_>. Citado 2 vezes nas páginas 24 e 25.

GOLDSCHMIDT, R. R. *Uma Introdução à Inteligência Computacional: fundamentos, ferramentas e aplicações*. Rio de Janeiro: Nome da editora, 2010. 143 p. ISBN 978-85-98931-08-1. Citado na página 21.

GOMIDE, F. A. Redes neurais artificiais para engenharia e ciências aplicadas: curso prático. In: . [S.l.]: SciELO Brasil, 2012. Citado na página 27.

GUIMARÃES, D. P. et al. Análise de sites disseminadores de fake news. In: *Anais Estendidos do XV Simpósio Brasileiro de Sistemas de Informação*. Porto Alegre,

RS, Brasil: SBC, 2019. p. 17–20. ISSN 2177-9384. Disponível em: <https://sol.sbc.org.br/index.php/sbsi_estendido/article/view/7431>. Citado na página 36.

HABASH, N.; RAMBOW, O.; ROTH, R. Mada+ token: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In: *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt*. [S.l.: s.n.], 2009. v. 41, p. 62. Citado na página 30.

HAYKIN, S. *Redes neurais: princípios e prática*. [S.l.]: Bookman Editora, 2007. Citado 3 vezes nas páginas 27, 28 e 29.

HAYKIN, S. et al. *Neural networks and learning machines*. vol. 3 pearson. *Upper Saddle River, NJ, USA*, 2009. Citado 2 vezes nas páginas 26 e 28.

HECHT-NIELSEN, R. Theory of the backpropagation neural network. In: *Neural networks for perception*. [S.l.]: Elsevier, 1992. p. 65–93. Citado na página 27.

HERMIDA, X. “Fake news tornam o jornalismo de qualidade mais necessário do que nunca”, diz diretor do EL PAÍS. 2018. Acessado em 20/12/2019. Disponível em: <https://brasil.elpais.com/brasil/2018/02/20/actualidad/1519128638_053924.html>. Citado na página 14.

HOSEA S.; HARIKRISHMAN, V. R. K. Artificial intelligence. In: . [S.l.]: Electronics Computer Technology (ICECT), 2011. p. 124–129. Citado na página 18.

HOSMER, D. W.; JOVANOVIC, B.; LEMESHOW, S. Best subsets logistic regression. *Biometrics*, JSTOR, p. 1265–1270, 1989. Citado na página 23.

JABEEN, H. *Stemming and Lemmatization in Python*. 2018. Acessado em 20/12/2019. Disponível em: <<https://www.datacamp.com/community/tutorials/stemming-lemmatization-python>>. Citado na página 37.

JR, E. C. T.; LIM, Z. W.; LING, R. Defining “fake news” a typology of scholarly definitions. *Digital journalism*, Taylor & Francis, v. 6, n. 2, p. 137–153, 2018. Citado na página 17.

JR, E. S. G. Exponential smoothing: The state of the art. *Journal of forecasting*, Wiley Online Library, v. 4, n. 1, p. 1–28, 1985. Citado na página 27.

KAELBLING, L. P.; LITTMAN, M. L.; MOORE, A. W. Reinforcement learning: A survey. *Journal of artificial intelligence research*, v. 4, p. 237–285, 1996. Citado na página 23.

KEMP, S. *Digital in 2018: World's Internet users pass the 4 billion mark*. 2018. Acessado em 20/12/2019. Disponível em: <<https://wearesocial.com/blog/2018/01/global-digital-report-2018>>. Citado na página 13.

KOTSIANTIS, S. B.; ZAHARAKIS, I.; PINTELAS, P. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, Amsterdam, v. 160, p. 3–24, 2007. Citado 2 vezes nas páginas 21 e 22.

- KUDO, T.; MATSUMOTO, Y. Chunking with support vector machines. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*. [S.l.], 2001. p. 1–8. Citado na página 28.
- LEAL, I. H. d. S. *O uso de aprendizagem de máquina para identificação e classificação de fake news no twitter referentes a eleição presidencial de 2018*. 2018. Monografia (Bacharelado em Ciência da Computação), Faculdade Doctum de Caratinga. Citado 2 vezes nas páginas 20 e 32.
- MALIK, U. *Implementing SVM and Kernel SVM with Python's Scikit-Learn*. 2018. Acessado em 20/10/2019. Disponível em: <<https://stackabuse.com/implementing-svm-and-kernel-svm-with-pythons-scikit-learn/>>. Citado na página 40.
- MARCHI, R. With facebook, blogs, and fake news, teens reject journalistic “objectivity”. *Journal of Communication Inquiry*, SAGE Publications Sage CA: Los Angeles, CA, v. 36, n. 3, p. 246–262, 2012. Citado na página 14.
- MARTINS K KATAOKA, L. T. *Processamento de linguagem natural*. 2010. Citado na página 29.
- MARUMO, F. S. *Deep Learning para Classificação de fake news por sumarização de texto*. 2018. Monografia (Bacharelado em Ciência da Computação), Universidade Estadual de Londrina. Citado 2 vezes nas páginas 20 e 32.
- MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of machine learning*. [S.l.]: MIT press, 2018. Citado na página 21.
- MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. In: *Sistemas Inteligentes Fundamentos e Aplicações*. 1. ed. Barueri-SP: Manole Ltda, 2003. p. 89–114. ISBN 85-204-168. Citado na página 15.
- MONTEIRO, R.; NOGUEIRA, R.; MOSER, G. Desenvolvimento de um sistema para a classificação de fakenews acoplado à etapa de etl de um data warehouse de textos de notícias em língua portuguesa. In: *Anais da XV Escola Regional de Banco de Dados*. Porto Alegre, RS, Brasil: SBC, 2019. p. 131–140. ISSN 2595-413X. Disponível em: <<https://sol.sbc.org.br/index.php/erbd/article/view/8486>>. Citado na página 31.
- MONTEIRO, R. A. et al. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In: *Computational Processing of the Portuguese Language*. [S.l.]: Springer International Publishing, 2018. p. 324–334. ISBN 978-3-319-99722-3. Citado 5 vezes nas páginas 15, 36, 37, 51 e 54.
- MONTEIRO RONEY L. DE SALES, T. A. S. P. R. A. Detecção automática de notícias falsas para o português. 2018. Acessado em 11/12/2019. Disponível em: <<https://nilc-fakenews.herokuapp.com/about>>. Citado na página 33.
- MORAIS, E. A. M.; AMBRÓSIO, A. P. L. Mineração de textos. *Relatório Técnico–Instituto de Informática (UFG)*, 2007. Citado na página 20.
- NEWMAN, N. et al. *Reuters institute digital news report 2019*. [S.l.]: Reuters Institute for the Study of Journalism, 2019. Citado na página 14.

- NING, S.; YAN, M. Discussion on research and development of artificial intelligence. In: IEEE. *2010 IEEE International Conference on Advanced Management Science (ICAMS 2010)*. [S.l.], 2010. v. 1, p. 110–112. Citado na página 18.
- OLIVEIRA, F. A.; NAVAU, P. O. A. Processamento de linguagem natural: princípios básicos e a implementação de um analisador sintático de sentenças da língua portuguesa. *Rio Grande do Sul*, 2004. Citado na página 29.
- PEREIRA, L. d. S. *Metodologia para avaliar técnicas de redução de protótipos: protótipos gerados versus protótipos selecionados*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2013. Citado na página 21.
- PÉREZ-ROSAS, V. et al. Automatic detection of fake news. *arXiv preprint arXiv:1708.07104*, 2017. Citado na página 33.
- PLATT, J. C. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: *ADVANCES IN LARGE MARGIN CLASSIFIERS*. [S.l.]: MIT Press, 1999. p. 61–74. Citado na página 46.
- RAPOZA, K. *Can 'Fake News' Impact The Stock Market?* 2017. Acessado em 20/12/2019. Disponível em: <<https://www.forbes.com/sites/kenrapoza/2017/02-26/can-fake-news-impact-the-stock-market/6f93aa252fac>>. Citado na página 14.
- RASCHKA, S. *Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning*. 2018. Citado na página 41.
- SANTOS, D. Processamento de linguagem natural através das aplicações. *quot; In Elisabete Ranchhod (ed) Tratamento das Línguas por Computador Uma introdução à linguística computacional e suas aplicações Lisboa: Caminho 2001, Caminho, 2001. Citado na página 29.*
- SANTOS, W. et al. Trendsbot: Verificando a veracidade das mensagens do telegram utilizando data stream. In: *Anais Estendidos do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. Porto Alegre, RS, Brasil: SBC, 2019. p. 65–72. ISSN 2177-9384. Disponível em: <https://sol.sbc.org.br/index.php/sbrc_estendido/article/view/7771>. Citado na página 36.
- SAS. *Inteligência Artificial*. 2019. Acessado em 20/12/2019. Disponível em: <https://www.sas.com/pt_br/insights/analytics/inteligencia-artificial.html>. Citado na página 18.
- SCIKITLEARN. *Neural Network Models*. 2019. Acessado em 20/12/2019. Disponível em: <https://scikit-learn.org/stable/modules/neural_networks_supervised.html>. Citado na página 26.
- SCIKITLEARN. *SGD: Maximum margin separating hyperplane*. 2019. Disponível em: <https://scikit-learn.org/stable/auto_examples/linear_model/plot_sgd_separating_hyperplane.html>. Citado na página 25.
- SEMOLINI, R. et al. Support vector machines, inferência transdutiva e o problema de classificação. [sn], 2002. Citado na página 28.

SHAO, C. et al. The spread of fake news by social bots. *arXiv preprint arXiv:1707.07592*, ArXiv e-prints, p. 96–104, 2017. Citado na página 18.

SHARMA, K. et al. Combating fake news: A survey on identification and mitigation techniques. *ACM Transactions on Intelligent Systems and Technology (TIST)*, ACM, v. 10, n. 3, p. 21, 2019. Citado na página 17.

SHU, K. et al. defend: Explainable fake news detection. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. New York, NY, USA: ACM, 2019. (KDD '19), p. 395–405. ISBN 978-1-4503-6201-6. Citado 2 vezes nas páginas 14 e 31.

SILVEIRA, D. *Brasil ganha 10 milhões de internautas em 1 ano, aponta IBGE*. 2018. Acessado em 20/12/2019. Disponível em: <<https://g1.globo.com/economia-/tecnologia/noticia/2018/12/20/numero-de-internautas-cresce-em-cerca-de-10-milhoes-em-um-ano-no-brasil-aponta-ibge.ghtml>>. Citado na página 13.

STATISTA. *Internet usage in Brazil - Statistics Facts*. 2017. Acessado em 20/12/2019. Disponível em: <<http://www.digitalnewsreport.org/survey/2018/brazil-2018/>>. Citado na página 13.

SZAFRAN, V. *Google, Twitter e Facebook são pressionados a agir contra as fake news*. 2019. Acessado em 20/12/2019. Disponível em: <<https://olhardigital.com.br/noticia-/google-twitter-e-facebook-sao-pressionados-a-agir-contra-as-fake-news/92222>>. Citado na página 14.

VINICIUS, A. *Introdução ao Aprendizado de Máquina*. 2017. Acessado em 21/12/2019. Disponível em: <<https://medium.com/@avinicius.adorno/introdução-a-aprendizado-de-máquina-e39ec5ef459b>>. Citado na página 23.

WIKIWAND. *Support-vector machine*. 2020. Acessado em 20/12/2019. Disponível em: <https://www.wikiwand.com/en/Support-vector_machine>. Citado na página 28.

WINGFIELD, N.; ISAAC, M.; BENNER, K. Google and facebook take aim at fake news sites. *The New York Times*, v. 11, p. 12, 2016. Citado na página 14.

ZILIO, D. Inteligência artificial e pensamento: redefinindo os parâmetros da questão primordial de turing. *Ciências & Cognição*, v. 14, n. 1, p. pp–208, 2009. Citado na página 29.