UNIVERSIDADE FEDERAL DE GOIÁS (UFG) /
UNIVERSIDADE FEDERAL DE CATALÃO (UFCAT) em implantação
INSTITUTO DE MATEMÁTICA E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM E OTIMIZAÇÃO

Jean Tomáz da Silva

DETECÇÃO DE FAKE NEWS EM PORTUGUÊS A PARTIR DE POUCOS DADOS ROTULADOS

CATALÃO
2022

UNIVERSIDADE FEDERAL DE GOIÁS
UNIDADE ACADÊMICA ESPECIAL DE MATEMÁTICA E TECNOLOGIA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO (TECA) PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES

E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico

[X] Dissertação [ ] Tese

2. Nome completo do autor

Jean Tomáz da Silva

3. Título do trabalho

Detecção de Fake News em Português a partir de poucos dados rotulados

4. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador)

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante:
a) consulta ao(à) autor(a) e ao(à) orientador(a);
b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação.
O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Nadia Felix Felipe Da Silva**, **Professor do Magistério Superior**, em 03/02/2022, às 16:29, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **JEAN TOMÁZ DA SILVA**, **Discente**, em 03/02/2022, às 17:20, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

JEAN TOMÁZ DA SILVA

DETECÇÃO DE FAKE NEWS EM PORTUGUÊS A PARTIR DE POUCOS DADOS ROTULADOS

Dissertação apresentada ao Programa de Pós-Graduação em Modelagem e Otimização da Instituto de Matemática e Tecnologia, da Universidade Federal de Goiás (UFG)/ Universidade Federal de Catalão (UFCAT) em implantação, como requisito para a obtenção do título de Mestre em Modelagem e Otimização. Área de concentração: Modelagem e Otimização. Linha de Pesquisa: Modelagem Computacional e Otimização.

Orientador(a): Professor(a) Doutor(a) Nádia Félix Felipe da Silva

CATALÃO
2022

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFCAT.

Silva, Jean Tomáz da
Detecção de Fake News em Português a partir de poucos dados rotulados / Jean Tomáz da Silva. - 2022.
89, f.

Orientadora: Profa. Dra. Nádia Félix Felipe da Silva.
Dissertação (Mestrado) - Universidade Federal de Catalão, Instituto de Matemática e Tecnologia, Catalão, Programa de Pós-Graduação em Modelagem e Otimização, Catalão, 2022.
Bibliografia.
Inclui siglas, abreviaturas, tabelas, lista de figuras, lista de tabelas.

1. PU Learning. 2. Notícias Falsas. 3. Avaliações Falsas. 4. Sem Rótulo. 5. Dois Passos. I. Silva, Nádia Félix Felipe da, orient. II. Título.

CDU 519.6

UNIVERSIDADE FEDERAL DE GOIÁS

UNIDADE ACADÊMICA ESPECIAL DE MATEMÁTICA E TECNOLOGIA

ATA DE DEFESA DE DISSERTAÇÃO

Ata nº 12 da sessão de Defesa de Dissertação de **Jean Tomáz da Silva**, que confere o título de **Mestre(a) em Modelagem e Otimização**, na área de concentração em **Modelagem e Otimização**.

Aos três dias do mês de fevereiro de 2022, a partir das 14h00 min, na sala meet.google.com/uer-vxes-oqw, realizou-se a sessão pública de Defesa de Dissertação, intitulada *"Detecção de Fake News em Português a partir de poucos dados rotulados"* nas dependências da Universidade Federal de Catalão, onde os programas de pós-graduação *Stricto Sensu* em funcionamento encontram-se provisoriamente vinculados à Universidade Federal de Goiás, em virtude de procedimentos técnicos relacionados à CAPES, já sendo realizada a transferência da Biblioteca Digital de Dissertações e Teses (BDTD). Assim, justifica-se os nomes das instituições neste documento, uma no cabeçalho (UFG), outra no corpo do texto (UFCAT). Os trabalhos foram instalados pelo(a) **Orientador(a), Professor(a) Doutor(a) Nádia Félix Felipe da Silva (orientadora) (UFG), com a participação dos demais membros da Banca Examinadora: Professor(a) Doutor(a) Fabíola Souza Fernandes Pereira (UFU), membro titular externo; Professor(a) Doutor(a) José dos Reis Vieira de Moura Júnior (PPGMO/IMTec/UFCAT), membro titular interno.** A Banca Examinadora reuniu-se em sessão secreta a fim de concluir o julgamento da Dissertação, tendo sida o(a) candidato(a) (X) Aprovado(a) ( ) Reprovado(a) pelos seus membros. Proclamados os resultados pelo(a) Professor(a) Doutor(a) **Nádia Félix Felipe da Silva**, Presidente da Banca Examinadora, foram encerrados os trabalhos e, para constar, lavrou-se a presente ata que é assinada pelos Membros da Banca Examinadora, aos três dias do mês de fevereiro de dois mil e vinte dois.

Obs: "*Banca Examinadora de Qualificação/Defesa Pública de Dissertação/Tese realizada em conformidade com a Portaria da CAPES n. 36, de 19 de março de 2020, de acordo com seu segundo artigo:*

Art. 2o A suspensão de que trata esta Portaria não afasta a possibilidade de defesas de tese utilizando tecnologias de comunicação à distância, quando admissíveis pelo programa de pós-graduação stricto sensu, nos termos da regulamentação do Ministério da Educação."

TÍTULO SUGERIDO PELA BANCA

Documento assinado eletronicamente por **Nadia Felix Felipe Da Silva**, **Professor do Magistério Superior**, em 03/02/2022, às 16:28, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **JEAN TOMÁZ DA SILVA**, **Discente**, em 03/02/2022, às 17:25, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Jose Dos Reis Vieira De Moura Junior**, **Professor do Magistério Superior**, em 03/02/2022, às 19:49, conforme horário oficial de Brasília, com

fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fabíola Souza Fernandes Pereira**, **Usuário Externo**, em 04/02/2022, às 18:01, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2666748** e o código CRC **86C55687**.

Referência: Processo nº 23070.065907/2021-29 SEI nº 2666748

Os Programas de Pós-Graduação *stricto sensu* em funcionamento na Universidade Federal de Catalão (UFCAT), em virtude de procedimentos técnicos relacionados à CAPES, continuam provisoriamente vinculados à Universidade Federal de Goiás (UFG), por isso, todos os elementos pré-textuais do trabalho apresentado estão identificados como Universidade Federal de Goiás/Universidade Federal de Catalão em implantação, em função da migração da BDTD ter ocorrido a partir de 16 de agosto de 2021, assim como pelo fato das pesquisas e produtos serem realizados na UFCAT.

A Deus que sempre ilumina meus passos, a minha esposa Márcia, a meus pais Adelson e Ivanir, a meus irmãos Daniel, Fernando, Adelson Júnior (Tico), por sempre apoiarem meus sonhos.

Dedico (In Memorian) aos meus avós Abrão e Geralda, que não viram esse momento mas acreditaram.

Agradecimentos

Agradeço em primeiro lugar a Deus, por ter me dado força e determinação para conquistar essa vitória na minha vida.

Agradeço a minha esposa pelo apoio, serenidade e confiança, ao meu lado nesta etapa.

Agradeço a minha família por acreditar que eu encerraria mais uma jornada de sucesso.

Agradeço ao minha orientadora, Prof[a]. Dr[a]. Nádia Félix Felipe da Silva, pelas recomendações, empenho, atenção e simpatia.

Enfim, agradeço a todos os amigos e colegas que direta ou indiretamente colaboraram com este trabalho.

“O que você sabe não tem valor; o valor está no que você faz com o que sabe."

Bruce Lee

RESUMO

SILVA, J.T.. *Detecção de Fake News em Português a partir de poucos dados rotulados*. 2022. 86 f. Dissertação (Mestrado em Modelagem e Otimização) – Instituto de Matemática e Tecnologia, Universidade Federal de Goiás (UFG)/ Universidade Federal de Catalão (UFCAT)em implantação, Catalão.

Há estudos sobre detecção de *fake news* há vários anos, mas a maior parte dos estudos são baseados em notícias produzidas no idioma Inglês. Este trabalho apresenta um estudo para detecção de *fake news* em notícias produzidas no idioma Português Brasileiro. O estudo mostra uma análise experimental do comportamento de classificadores supervisionados e semi-supervisionados capazes de predizer se uma notícia é ou não categorizada como *fake news*. Considerando que um conjunto de dados possui a maior parte dos dados sem rótulo de classe e que este dados podem ser ou não *fake news*, essas definições identificam uma nova categoria de classificadores treinados a partir de exemplos positivos e exemplos sem rótulo, ou, PU (*Positive and Unlabeled*) *learning*. Os resultados do estudo mostram que o PU *learning* supera outros classificadores de aprendizado supervisionado e semi-supervisionado, pois consegue identificar com eficácia mais *fake news* em um cenário onde se tem poucos exemplos positivos.

Palavras-chaves: PU Learning, Notícias Falsas, Avaliações Falsas, Sem Rótulo, Dois Passos, Class Prior, Escore de Propensão, Frequência de Rótulo.

ABSTRACT

SILVA, J.T.. *Monography Template. Master in Modeling and Optimization.* 2022. 86 f. Master Thesis in Modelling and Optimization – Instituto de Matemática e Tecnologia, Universidade Federal de Goiás (UFG)/ Universidade Federal de Catalão (UFCAT)em implantação, Catalão.

There have been studies on detecting false news for several years, but most studies are based on news produced in the English language. This work presents a study to detect false news in news produced in Brazilian Portuguese language. The study shows an experimental analysis of the behavior of supervised and semi-supervised classifiers capable of predicting whether or not news is categorized as false news. Concepts that a dataset has most of the data without a class label and that this data may or may not be false news, these definitions identify a new category of classifiers trained from positive examples and unlabeled examples, or, PU (Positive and Unlabeled) learning. The results of the study show that PU learning outperforms other supervised and semi-supervised learning classifiers, as it can effectively identify more fake news in a scenario where there are few positive examples.

Keywords: PU Learning, Fake News, Fake Reviews, Unlabeled, Two Steps, Class Prior, Propency Score, Label Frenquency.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AdaBoost — Adaptive Boosting

AUC — Area Under The Curve

CNN — Convolutional Neural Networks

DT — Decision Tree

EM — Expectation-Maximization

I-EM — Initial EM

IA — Inteligência Artificial

IDF — Inverse Document Frequency

INDF — Inverse Negative Document Frequency

IPDF — Inverse Positive Document Frequency

KNN — k-Nearest Neighbour

LDA — Latent Dirichlet Allocation

LIWC — Linguistic Inquiry and Word Count

LPU — Learning from Positive and Unlabeled Examples

LR — Logistic Regression

MDINPUL — Mixing Population and Individual Nature PU Learning

NB — Naive Bayes

NILC — Núcleo Interinstitucional de Linguística Computacional

NLP — Natural Language Processing

PEBL — Positive Example Based Learning

PLN — Processamento da Linguagem Natural

PNLH — Positive examples and Negative examples Labeling Heuristic

POS — Parts of Speech

PSO — Particle Swarm Optmization

PU — Positive and Unlabeled

RAM — Random Access Memory

RF — Random Forest

RNC — Redes Neurais Convolucionais

Roc-SVM — Rocchio with SVM

S-EM — Spy with EM

SAR — Selected At Random

SCAR — Selected Completely At Random

SNAR — Selected Not At Random

SVM — Support Vector Machine

TF — Term Frenquency

TF-IDF — Term Frequency–Inverse Document Frequency

TFIPNDF — Term Frequency Inverse Positive-Negative Document Frequency

URL — Uniform Resource Locator

WV — Weighted Voting

SUMÁRIO

Capítulo 1

INTRODUÇÃO

O crescimento e a expansão das tecnologias de comunicação, permitiram que a informação não ficasse limitada apenas em rádios, televisão e jornais. Esse avanço proporcionou a divulgação de notícias em diversas formas no meios digitais, seja em seu formato de texto, imagem, áudio ou vídeo, por meio de páginas da Internet, sendo em grande parte de *sites* que representam agências de notícias confiáveis.

As notícias são fundamentais para levar informação íntegra para um determinado público, seja um pequeno grupo ou até mesmo uma população. Mas existem informações que são manipuladas, como as *fake news* que podem ser usadas por entidades fraudulentas para falsificar opiniões, decisões em investimentos, educação, eleições e até mesmo em campanhas publicitárias (ZHANG; GHORBANI, 2020).

Tanto as pessoas quanto as empresas fazem uso das opiniões em avaliações para tomar decisão no momento da compra de um produto ou mesmo para definir qual campanha de *marketing* é fundamental para um determinado projeto de vendas. Muitos revisores publicam avaliações (revisões) falsas (*fake reviews*) para promover a falta de confiança do consumidor em um determinado produto ou serviço, sendo que uma opinião positiva frequentemente significa lucros e fama para as empresas (LI *et al.*, 2014).

Diversas técnicas de Aprendizado de Máquina são estudadas e utilizadas para detecção desse tipo de informação enganosa, seja *fake news* ou *fake reviews*. No caso de informações enganosas em texto de notícias, faz-se uso do *Processamento da Linguagem Natural* (PLN) , traduzido do inglês *Natural Language Processing* (NLP), como mecanismo fundamental para o tratamento dos dados que servirão como entrada para algoritmos de Aprendizado de Máquina.

Os algoritmos tradicionais de aprendizado supervisionado necessitam de dados anotados (rotulados) manualmente, no entanto, a qualidade desse tipo de atividade e a quantidade de exemplos rotulados pode afetar diretamente o classificador treinado (HE *et al.*, 2020). O *Positive Unlabeled* (PU) *learning* (BEKKER; DAVIS, 2020) é capaz de superar esse

problema mesmo que o conjunto de dados para treinamento seja constituído de uma parte pequena dos dados com exemplos positivos (*fake*) que representa os dados conhecidos e de uma parte grande dos dados com exemplos sem rótulo (*unlabeled set*) que representa os dados desconhecidos.

Além disso, o PU *learning* é capaz de lidar apenas com dados que pertencem somente a uma classe (positiva), sem necessidade de ter mais de uma classe (positiva e negativa) no conjunto de dados, pois o objetivo do algoritmo é identificar a classe definida como positiva. Esse papel principal do algoritmo que define seu próprio nome *Positive and Unlabeled Learning*.

1.1 Questão, hipótese e objetivos do trabalho

O trabalho no geral, busca encontrar uma resposta para as seguintes questões:

1. É possível identificar *fake news* em notícias do idioma Português Brasileiro, levando em conta que o conjunto de dados analisado possui uma menor parte dos dados que são conhecidos (com rótulo de classe *fake news*) e uma maior parte dos dados que são desconhecidos (sem rótulo)?

2. Qual o grau da capacidade de generalização dos algoritmos supervisionados e semi-supervisionados em relação aos algoritmos de PU *learning*, aplicados em problemas de classificação de textos de notícias, onde o conjunto de dados possui poucos exemplos com rótulo *fake*?

Em relação aos questionamentos, a hipótese que se deve considerar para o desenvolvimento do trabalho leva em conta que através do uso de técnicas de Aprendizado de Máquina é possível treinar um classificador que seja capaz de categorizar um texto de uma notícia produzida no idioma Português Brasileiro, como sendo ou não *fake news*. Logo, se determina qual das técnicas mais conhecidas de Aprendizado de Máquina aplicadas para problemas de classificação de textos possui um melhor desempenho comparado com as demais.

Considerando as hipóteses apresentadas, o objetivo geral do trabalho é utilizar técnicas de Aprendizado de Máquina conhecidas como supervisionadas e semi-supervisionadas (em especial *Positive and Unlabeled Learning*) para o treinamento de um classificador a partir de um conjunto de dados que possui poucos exemplos de notícias com rótulo de classe, produzidas no idioma Português Brasileiro. Uma vez treinado o classificador, o mesmo é utilizado na etapa de validação, que consiste em avaliar o mesmo em uma parte do conjunto de dados reservado para tal.

Alinhados ao objetivo geral, os objetivos específicos seguem em destaque:

- Levantar os algoritmos de Aprendizado de Máquina mais utilizados para problemas de classificação de texto, aplicados principalmente em notícias que são *fake news*.
- Implementar os algoritmos encontrados que sejam mais relevantes para o trabalho, utilizando *frameworks* para técnicas de Aprendizado de Máquina.
- Comparar o desempenho entre os classificadores gerados pelos algoritmos, quando aplicados em um cenário onde se tem um conjunto de dados com poucos dados que representam as notícias que são *fake news* e o restante dos dados são desconhecidos.

1.2 Justificativa

A evolução das mídias sociais como *Facebook* e *Twitter* permitiu às pessoas se comunicarem com agilidade e facilidade. De acordo com Balmas (2014) essa crescente popularidade possibilita que a Internet se torne um terreno abundante e amplo para divulgação de *fake news*, tais como informações falsas, opiniões falsas, anúncios falsos, especulações, rumores, declarações políticas falsas, sátiras e assim sucessivamente. Isso faz com que as *fake news* se tornem cada vez mais populares que as demais notícias, principalmente nas redes sociais onde se localizam grande parte das pessoas que fazem uso de ferramentas digitais para comunicação.

Há basicamente dois tipos de sistemas para detecção de *fake news*, aqueles cuja abordagem é baseada em prática, que se refere à perspectiva dos usuários da Internet e abordagem baseada em pesquisas acadêmicas. A abordagem baseada em prática engloba as ferramentas *online* de checagem de fatos ou *fact-checking* disponibilizadas em plataformas de consulta que são criadas por empresas com ou sem fins lucrativos, como *FactCheck.org*[1], *PolitiFact.com*[2] e *Snopes.com*[3]. Além dessas ferramentas, outro tipo de abordagem baseada em prática é a orientação e a conscientização dos usuários que fazem uso de redes sociais, quanto aos cuidados para avaliar as informações que recebem ou encaminham. Já a abordagem baseada em pesquisas acadêmicas possui três categorias que são: baseada em componente (criador, alvo, conteúdo e contexto social), baseada em mineração de dados (modelos de Aprendizado de Máquina) e baseada na implementação (*offline* ou *realtime*)(ZHANG; GHORBANI, 2020).

A categoria baseada no componente quando se aplica ao conteúdo da informação faz uso de análises baseadas em linguística e semântica que são estudos clássicos e científicos da linguagem natural, ou seja, a linguagem puramente escrita e falada. Essas análises identificam características com base no padrão da linguagem, da estrutura e do sentido do texto.

[1] https://www.factcheck.org/
[2] https://www.politifact.com/
[3] https://www.snopes.com/

Além disso, abordagens baseadas em prática envolvem bastante trabalho manual que pode levar tempo para identificar o tipo de conteúdo de um texto (ZHANG; GHORBANI, 2020).

A classificação de texto é o processo tradicional de atribuição de rótulos de determinada categoria (classe positiva ou negativa) para novos exemplos de textos com base no conhecimento (experiência) adquirido pelo classificador durante a etapa de treinamento (LIU *et al.*, 2003). O principal problema com esse processo tradicional é que um grande número de exemplos com rótulo de cada classe são necessários para que um classificador obtenha bons resultados durante a etapa de treinamento (LIU *et al.*, 2003).

Aprender a partir de poucos dados rotulados, ou seja, poucos exemplos positivos e muitos exemplos sem rótulos, se denomina como PU *learning* que é uma variante do processo tradicional, sendo que os dados de treinamento consistem em exemplos positivos e exemplos sem rótulo. A suposição é que cada exemplo sem rótulo pode pertencer à classe positiva ou negativa (BEKKER; DAVIS, 2020).

1.3 Metodologia

Considerando as abordagens apresentadas, a pesquisa é classificada como experimental, pois faz uso da abordagem baseada em pesquisa acadêmica, empregando técnicas de Aprendizado de Máquina. Os problemas ou técnicas de Aprendizado de Máquina são divididos em supervisionado, semi-supervisionado e não supervisionado. Tais técnicas são avaliadas com métricas específicas para garantir um bom resultado na construção do classificador, de modo que o mesmo identifique textos de notícias que podem ser *fake news* em um conjunto de dados que possui poucos exemplos rotulados (FACELI *et al.*, 2011).

A pesquisa experimental implica ter uma ou mais variáveis experimentais que podem ser controladas ou observadas conforme a medição que será aplicada, pois faz uso de várias técnicas de amostragem para provocar alterações no ambiente analisado, de forma que a cada iteração possa encontrar os resultados esperados (WAZLAWICK, 2014).

Além disso, esse trabalho faz uso da pesquisa bibliográfica, pois implica o estudo de artigos, teses, livros e outras publicações disponibilizadas por editoras ou fontes indexadas (WAZLAWICK, 2014).

Em todo e qualquer trabalho científico é fundamental o uso da pesquisa bibliográfica, pois a mesma mantém o pesquisador atualizado quantos as informações que ele não tinha percepção, levando em conta que a metodologia em si não produz nenhum conhecimento novo (WAZLAWICK, 2014).

De acordo com a forma que a pesquisa é aplicada, para se atingir os objetivos, as seguintes atividades serão executadas, como:

- Levantamento bibliográfico das técnicas para identificar *fake news* em notícias produzidas no idioma Português Brasileiro.
- Pesquisa de conjunto de dados que são constituídos de textos de notícias, que foram anotados manualmente e construídos para serem utilizados para o treinamento de algoritmos de Aprendizado de Máquina, com o propósito de identificar *fake news* em textos no idioma Português Brasileiro.
- Validar e testar os algoritmos de Aprendizado de Máquina disponíveis na literatura e que atendam as necessidades do trabalho, principalmente classificação de textos.
- Comparar os resultados dos algoritmos de Aprendizado de Máquina selecionados na literatura em relação aos algoritmos baseados em PU *learning* selecionados para o experimento.
- Ao finalizar o estudo transformar os resultados em publicação para uma revista científica.

1.4 Estrutura do Trabalho

A organização do trabalho está disposta em seis capítulos. Os capítulos seguintes foram organizados como:

- Capítulo 2: Aborda brevemente os conceitos sobre *fake news*, tratando sobre as aplicações de Aprendizado de Máquina em geral, detalhando mais o aprendizado semi-supervisionado baseado em PU *learning*.
- Capítulo 3: Retrata em geral, a respeito dos trabalhos encontrados na literatura que fazem uso de Aprendizado de Máquina na identificação de textos que podem ser classificados como *fake news* ou *fake reviews*.
- Capítulo 4: Descreve o problema por completo, considerando os parâmetros e critérios adotados para a execução do experimento com algoritmos de aprendizado supervisionado e semi-supervisionado, aplicados em um conjunto de dados com poucos exemplos rotulados.
- Capítulo 5: Demonstra os experimentos aplicados, bem como os resultados alcançados e um breve comparativo entre ambos.
- Capítulo 6: Apresenta uma breve conclusão e trabalhos futuros, a fim de concluir o mestrado.

Capítulo 2

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo é apresentada a conceitualização fundamental para o entendimento desta dissertação. A Seção 2.1 apresenta uma breve definição de *fake news*. A Seção 2.2 exibe uma visão geral sobre o aprendizado supervisionado. A Seção 2.3 apresenta uma sucinta descrição de aprendizado semi-supervisionado. Na Seção 2.4 é apresentada a técnica de PU *learning*.

2.1 *Fake News*

As *fake news* são classificadas como um tipo de propaganda, pois as mesmas foram consideradas como uma categoria de manchete sensacionalista que surgiu no final do século XIX. Esta categoria foi conhecida nesse período como "jornalismo amarelo", que se constitui de conteúdos escritos com desinformações ou boatos espalhados através de mídia. Uma notícia em mídia pode surgir tanto na forma tradicionalmente impressa, no formato de jornais e revistas, quanto na sua forma digital nas mídias sociais, dando destaque aos *tweets*, as postagens em *blogs* ou em redes sociais (ZHANG; GHORBANI, 2020).

De acordo com Zhang e Ghorbani (2020), atualmente, as *fake news* se encontram em toda parte, em qualquer tipo de mídia, sendo irritantes, intrusivas e perturbadoras, pois provocam impactos profundos tanto nos indivíduos quanto na sociedade. Portanto, é importante a construção de um sistema automático de detecção eficaz para a identificação de *fake news*.

Segundo Bondielli e Marcelloni (2019), as *fake news* não são um produto da era da comunicação digital, dado que antes do advento da Internet, os jornalistas tinham a tarefa de verificação e checagem dos fatos de notícias, um processo manual de detecção. Atualmente, as *fake news* são encontradas em *websites*, criados especialmente para espalhar a desinformação e geralmente compartilhados em plataformas de redes sociais.

Existem muitos desafios em tratar *fake news*, dentre eles é possível citar:

- **Volume**: Discorre sobre a quantidade massiva de *fake news* que são distribuídas através da Internet, mesmo sem o conhecimento dos usuários, por meios de *websites* com interesse em ganhos financeiros ou políticos (ZHANG; GHORBANI, 2020).

- **Variedade**: Considera que existem várias definições aproximadas de *fake news*, como os rumores, as notícias sátiras, as críticas ou avaliações falsas (*fake reviews*), as informações incorretas, os anúncios falsos, as teorias da conspiração, as declarações falsas de políticos, etc., que afetam as pessoas em todos os aspectos (ZHANG; GHORBANI, 2020).

- **Velocidade**: Leva em consideração que os produtores de *fake news* possuem uma vida curta, pois criam um *website* falso por um determinado período de tempo e depois o encerram, tornando difícil a identificação da origem da mensagem divulgada como no caso das redes sociais, que fazem uso de perfis falsos (ZHANG; GHORBANI, 2020).

Em relação à característica "variedade" das *fake news*, segundo Zhang *et al.* (2016), as *fake reviews* são como revisões ou avaliações enganosas fornecidas com a intenção de enganar os consumidores em sua tomada de decisão de compra. Geralmente tais revisões são fornecidas por revisores com pouca ou nenhuma experiência real com os produtos ou serviços que estão sendo avaliados.

Há diversas técnicas para detecção de *fake news*, considerando que as mesmas podem ter conteúdo físico (título, corpo do texto, imagem ou vídeo) ou não físico (propósito, sentimento e novos tópicos). Uma das técnicas de detecção é a análise de conteúdo, na qual é analisado os padrões linguísticos e estilos de escrita. Os estudos baseados em análise de conteúdo podem ser categorizados em (ZHANG; GHORBANI, 2020):

- **Análise baseada em linguística e semântica**: pela extração de informações úteis pode se analisar os padrões de linguagem associados, estruturas e significado das notícias, considerando que a maioria dos produtores de *fake news* usam estratégias de escrita específica para evitar sua identificação. Na análise baseada em linguística são utilizados métodos como *bag-of-words* e *n-grams* para representar o texto de notícias em seu formato bruto, traduzido do inglês *raw format*, ao passo que na análise baseada em semântica refere-se ao processo de caracterizar as estruturas sintáticas das notícias, desde o nível da sentença até o nível semântico. Por exemplo, o título da notícia é sensacionalista para chamar atenção, e não tem relação com seu conteúdo, a partir disso é possível combinar essas informações para se aplicar as análises de semântica (ZHANG; GHORBANI, 2020).

- **Análise baseada em conhecimento**: refere-se aos mecanismos de tentar checar a veracidade dos fatos de uma determinada notícia. Há diversos *websites* com essa fina-

lidade para o idioma Inglês, como o *FactCheck.org*[1], e para o idioma Português Brasileiro tem como exemplo a Agência Lupa[2]. Esse tipo de análise é um componente fundamental da detecção de *fake news*, porém é importante enfatizar que em primeiro lugar os modelos de aprendizagem baseados em inteligência artificial são soluções viáveis, mas atualmente as tarefas de checagem dos fatos dependem muito de conhecimento humano. Em segundo lugar, a detecção de *fake news* é uma tarefa binária, que possibilita construir modelos de aprendizado supervisionado para classificar as notícias como sendo ou não *fake news* (ZHANG; GHORBANI, 2020).

- **Análise baseada em estilo**: é classificada em análise de estilo físico e análise de estilo não físico. A primeira é definida como o processo de extração de características físicas mais influentes para distinguir as *fake news* das demais notícias, que leva em consideração o estilo de redação, a sintaxe do texto e a atitude pessoal das notícias, como o número de verbos e substantivos, o número de palavras emocionais e palavras casuais. A segunda definição analisa os aspectos não físicos das notícias, como a complexidade e a legibilidade do texto (ZHANG; GHORBANI, 2020).

Segundo Zhang *et al.* (2016), a detecção automática de *fake news* ou de *fake reviews* ao usar as técnicas de Aprendizado de Máquina, pode ajudar a resolver os problemas relacionados com as limitações de quando se faz o uso da identificação manual, na qual pessoas especializadas buscam compreender e interpretar os textos a serem classificados. O processo de identificação manual é considerado uma tarefa desafiadora que produz resultados enviesados se comparado a outras formas de análise, como a comunicação face à face que possui indícios como expressão facial, gestos corporais e tom de voz. Os modelos de Aprendizado de Máquina podem ser aplicados para identificar a categoria do conteúdo como sendo *fake news*, alcançando um resultado melhor em relação ao processo manual, tanto na qualidade da classificação quanto na redução de custo operacional.

2.2 Aprendizado de Máquina

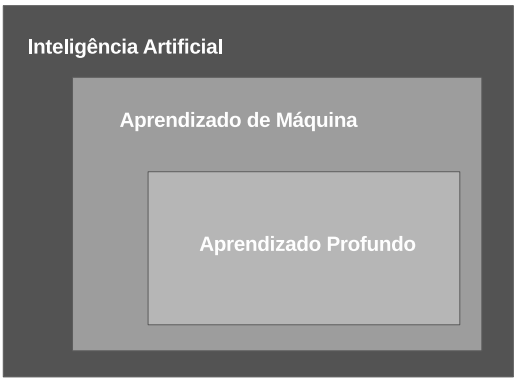
Aprendizado de Máquina é uma das subáreas da Inteligência Artificial (IA) que propõe a criação de modelos com base em métodos e técnicas de aprendizado, de forma a adquirir conhecimento e predizer sobre um grande volume de dados, permitindo tomar decisões a partir desses conhecimentos acumulados em experiências de sucesso ocorridas anteriormente (REZENDE, 2003). Há várias definições de Aprendizado de Máquina na literatura, onde na visão de Mitchell (1998) é descrita como: "A capacidade de melhorar o desempenho na realização de alguma tarefa por meio de experiência".

[1] https://www.factcheck.org/
[2] https://piaui.folha.uol.com.br/lupa/

Os computadores no Aprendizado de Máquina são preparados para aprender com experiências anteriores, usando um princípio de inferência chamada de indução, para obter conclusões genéricas a partir de um conjunto particular de exemplos (FACELI *et al.*, 2011). Esse conjunto de exemplos representam as instâncias do problema a ser resolvido, que os algoritmos de Aprendizado de Máquina usam para adquirir a capacidade de induzir uma função ou hipótese para resolver o problema (FACELI *et al.*, 2011). Tal conjunto de exemplos, é denominado conjunto de dados.

A Figura 2.1 representa de forma geral a relação entre as áreas da IA, sendo o topo a própria IA que engloba diversas teorias e tecnologias, e abaixo a subárea de Aprendizado de Máquina do qual faz parte também o aprendizado profundo.

Figura 2.1 – Visão de alto nível dos principais componentes do mundo da IA.



Fonte: Adaptado de Taulli (2019)

O aprendizado profundo traduzido do inglês *deep learning* é uma subárea de IA que utiliza redes neurais que simulam o funcionamento do cérebro humano. Durante a última década, a maioria das inovações surgiu através de pesquisas nesta subárea da IA (TAULLI, 2019).

O Aprendizado de Máquina é representado por diversos algoritmos utilizados para resolver problemas. O tipo de problema define o algoritmo adequado para ser aplicado, bem como a função a ser utilizada também (MUELLER; MASSARON, 2016).

O funcionamento geral de um algoritmo de Aprendizado de Máquina pode ser representado conforme ilustra a Figura 2.2, onde a entrada E é uma sequência de exemplos $l_1 ... l_n$ que levam o algoritmo de Aprendizado de Máquina H gerar uma saída C que é um classificador, conhecido também como hipótese ou descrição de conceito, capaz de dado um novo exemplo, predizer seu rótulo de classe. Cada exemplo é representado por uma tupla formada pelos atributos, ou vetor de características,e o rótulo da classe associada (MONARD; BARANAUSKAS, 2003).

Figura 2.2 – Aprendizado de Máquina.

Fonte: O autor.

Segundo Monard e Baranauskas (2003), um exemplo l é um par $(x_i, f(x_i))$ onde x_i é o vetor e $f(x_i)$ é a função que se tenta predizer a partir de atributos do vetor. A tarefa do algoritmo de Aprendizado de Máquina, ou indutor é, dado um conjunto de exemplos, induzir uma função h que aproxima f , normalmente desconhecida, em que h é chamada hipótese sobre a função objetivo f , ou seja, $h(x_i) \approx f(x_i)$.

O comportamento da hipótese, ou classificador, de ser capaz de predizer corretamente novos exemplos que forem utilizados como entrada no algoritmo de Aprendizado de Máquina, dá-se o nome de capacidade de generalização (FACELI *et al.*, 2011). Se a capacidade de generalização for baixa pode indicar que os dados para treinamento do algoritmo de Aprendizado de Máquina estão superajustados, comportamento conhecido como *overfitting*. Um comportamento inverso apresenta os dados de treinamento subajustados, conhecido como *underfitting*, onde a taxa de acertos é baixa (MONARD; BARANAUSKAS, 2003).

É importante considerar em relação aos algoritmos de Aprendizado de Máquina que para representar a hipótese inferida há diversas formas, por meio por exemplo de redes neurais artificiais, árvores de decisão ou conjunto de regras.

Segundo Faceli *et al.* (2011), as formas de representação de um algoritmo define seu viés (*bias*) de representação, por exemplo no caso das árvores de decisão que fazem uso de uma estrutura em árvore, onde cada nó interno representa um questão associada ao atributo (característica) e nó externo representa a classe atribuída. Sem viés um algoritmo de Aprendizado de Máquina não teria um aprendizado ou generalização para aplicar em novos dados com sucesso (MITCHELL, 1998).

De acordo com Burkov (2019), as técnicas de Aprendizado de Máquina são categorizadas em: aprendizado supervisionado, aprendizado não supervisionado, aprendizado semi-supervisionado e aprendizado por reforço.

No aprendizado supervisionado, o conjunto de dados é uma coleção de exemplos com rótulo de classe $\{(x_i, y_i)\}_{i=1}^{N}$, onde cada elemento x_i em N é um vetor de característi-

cas, traduzido do inglês *feature vector* (BURKOV, 2019). Cada vetor de características é um vetor em que cada dimensão $j = 1, ..., D$ contém um valor que descreve o exemplo, sendo esse valor chamado de *feature* e representado por $x^{(j)}$. Se o exemplo é uma notícia, a primeira *feature* $x^{(1)}$ pode ser o número de palavras, a segunda *feature* $x^{(2)}$ pode ser a classe gramatical das palavras, traduzido do inglês *Part of Speech Tag*, a terceira *feature* $x^{(3)}$ pode ser a frequência de uma palavra, e assim sucessivamente (BURKOV, 2019). O rótulo de classe y_i atribuído a um exemplo diz respeito a classe ou categoria a qual o exemplo pertence, considerando que se os exemplos são notícias e o problema é identificar se as mesmas são *fake news*, então entenda-se que se tem duas classes {*fake news*, *not fake news*} (BURKOV, 2019).

Segundo Burkov (2019), o objetivo do algoritmo de aprendizado supervisionado é produzir um modelo, também chamado de classificador, que tem como entrada um vetor de características x e pode predizer o rótulo de classe y para esse vetor de características. Como, por exemplo, o modelo gerado a partir de um conjunto de dados que possuem exemplos de notícias tem como entrada um vetor de característica que descreve a notícia e como saída a probabilidade de ser *fake news*.

No aprendizado não supervisionado não há rótulo de classe para os exemplos de entrada. O algoritmo processa os exemplos de entrada com o objetivo de obter algum agrupamento, ou, por exemplo, restruturar os dados em novas características que representam uma classe. A tarefa mais comum deste aprendizado é o agrupamento (FACELI *et al.*, 2011).

Para Burkov (2019), no aprendizado semi-supervisionado, o conjunto de dados possui exemplos com e sem rótulo de classe, sendo que a quantidade de exemplos sem rótulo sempre é maior em relação ao número de exemplos com rótulo. O objetivo do algoritmo semi-supervisionado é o mesmo do algoritmo supervisionado, com a diferença que usar muitos exemplos sem rótulo pode ajudar o algoritmo de aprendizado semi-supervisionado a encontrar um melhor modelo.

O aprendizado por reforço tem como objetivo recompensar por uma ação considerada positiva ou punir por uma ação considerada negativa, como, por exemplo, treinar um robô para encontrar o melhor trajeto entre dois pontos (FACELI *et al.*, 2011).

O escopo deste trabalho engloba o aprendizado supervisionado e semi-supervisionado. O leitor interessado em maiores detalhes sobre outras categorias mencionadas de Aprendizado de Máquina pode recorrer aos trabalhos de Boutaba *et al.* (2018), Muhammad e Yan (2015) e Agarwal e Mittal (2014).

2.3 Aprendizado Supervisionado

Uma aplicação tradicional possui funções, como, por exemplo, soma() que aceita uma entrada e produz um resultado (saída), logo é necessário para o desenvolvedor de aplicativos

conhecer e construir esta função. No Aprendizado de Máquina é de conhecimento somente as entradas e resultados, sendo necessário descobrir qual a função que deve ser aplicada para obter o resultado desejado. O processo de mapear os dados (entrada e saída) de modo que encontre uma função é conhecido como treinamento (MUELLER; MASSARON, 2016).

O treinamento basicamente visa fazer com que o algoritmo seja capaz de aprender a distinguir entre os exemplos de notícias que são *fake news* e não *fake news*, permitindo que a experiência (indução) adquirida se aplique a novos exemplos de notícias não vistos no treinamento. De modo abstrato, o aprendizado é visto como um processo de "usar experiência para se obter conhecimento". O aprendizado supervisionado descreve um cenário no qual a "experiência", ou seja, um exemplo de treinamento, contém informações significantes (de classe como *fake news* ou não *fake news*). Estas informações (rótulos de classe) são escondidas nos exemplos de testes, onde o "conhecimento" adquirido durante a etapa de treinamento será aplicado para predizer as informações que foram escondidas (SHALEV-SHWARTZ; BEN-DAVID, 2014).

De acordo com Géron (2017), no aprendizado supervisionado, os dados (exemplos ou instâncias) de treinamento que são fornecidos para o algoritmo incluem soluções desejadas chamadas de rótulos, conforme ilustra na Figura 2.3.

Figura 2.3 – Um conjunto de treinamento rotulado para aprendizado supervisionado

Fonte: Adaptado de Géron (2017)

Até o momento foi descrito a expressão "conjunto de dados", sendo que, na prática, nas tarefas de aprendizado são usados três conjuntos de dados: conjunto de dados de treinamento, conjunto de dados de validação e conjunto de dados de teste. De acordo com Burkov (2019), quando todos os exemplos de um conjunto de dados estão anotados, isto é informado seu rótulo de classe, deve-se "embaralhar" os mesmos e dividir os conjunto de dados em treinamento, validação e teste ou apenas em treinamento e teste. O conjunto de dados de treinamento é o maior deles e é usado para construir o modelo de Aprendizado de Máquina, enquanto que os outros conjuntos de dados possuem o mesmo tamanho e não são usados para construção do modelo, mas para validar o mesmo.

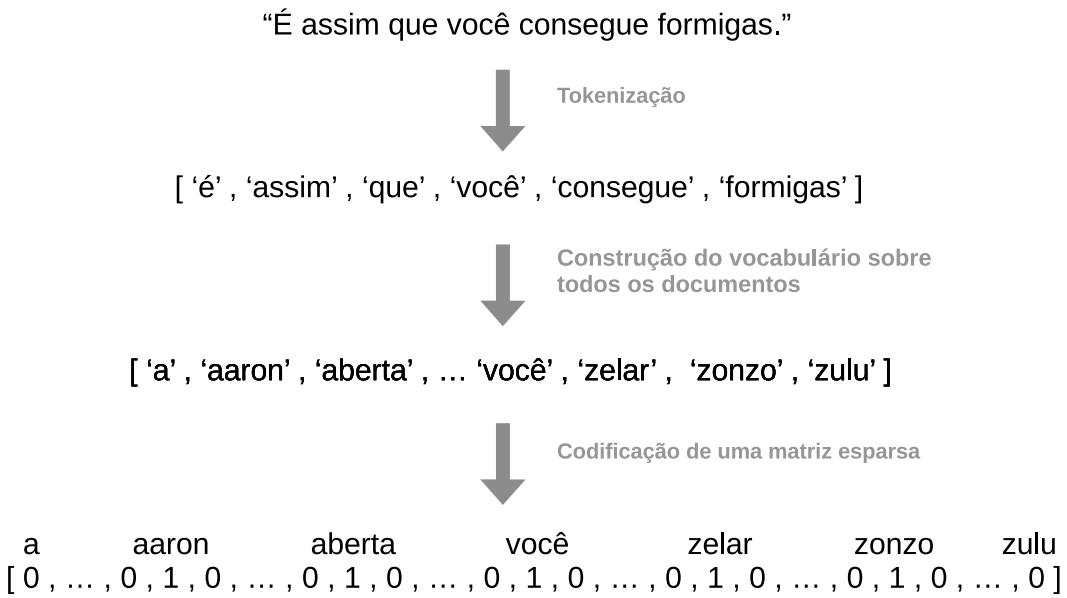
Considere que o problema a ser resolvido com Aprendizado de Máquina é a detecção

de *fake news*. A partir de então, é coletado 1000 exemplos de notícias, cada uma delas com o rótulo de classe "*fake*" ou "*not fake*". Lembrando que essa etapa envolve um processo manual de anotação desses rótulos que requer um custo operacional. Com os dados coletados há necessidade de converter cada exemplo em um vetor de caraterísticas.

Segundo Müller e Guido (2016), a forma comum de representar um texto de uma notícia para o Aprendizado de Máquina é usando *bag of words*, onde é descartado as estruturas de entrada do texto da notícia, como capítulos, parágrafos, sentenças e formatação, e somente contabilizando "com que frequência uma palavra aparece em cada texto".

A Figura 2.4 mostra como é o processo de aplicar a técnica de *bag of words* para representação do texto das notícias, seguindo três passos (MüLLER; GUIDO, 2016):

Figura 2.4 – Processamento de um texto para um vetor de características



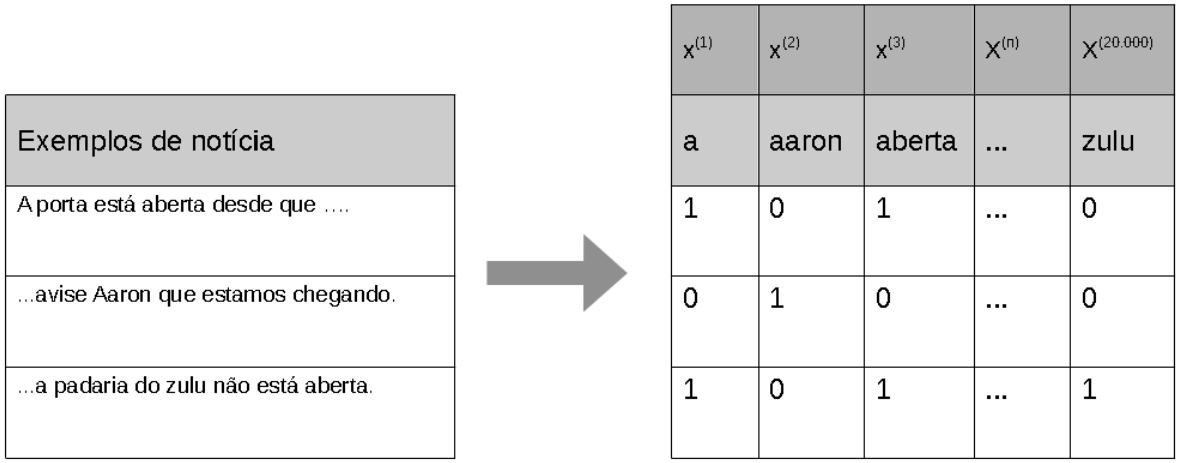
Fonte: (MüLLER; GUIDO, 2016)

- **Tokenização**: essa etapa consiste em dividir os textos das notícias em palavras que aparecem em cada um deles, como, por exemplo, separar as palavras pelo caractere de espaço ou pontuação. As fatias ou partes dessa divisão são conhecidas como *token*.
- **Construção do vocabulário**: essa etapa visa construir um vocabulário de todas as palavras que aparecem em todas as notícias que fazem parte do conjunto de dados e enumerar cada um em ordem alfabética.
- **Codificação**: essa etapa envolve para cada notícia, contabilizar com que frequência cada uma das palavras do vocabulário aparece no texto de cada notícia.

Deixando mais claro, imagine pegar 10.000 notícias de um *website* no idioma Português Brasileiro (considerando que todas as notícias contenham 20.000 palavras distintas) e representar cada exemplo de notícia em um vetor de características (BURKOV, 2019). Desse modo, conforme é ilustrado na Figura 2.5, o vetor de característica de cada exemplo de notícia será formado de $x^{(1)}$ até $x^{(20.000)}$ *features* que representam todas as palavras distintas, formando uma matriz de exemplos e *features*, também chamada de *Document-Term Matrix*, onde:

- A primeira *feature* $x^{(1)}$ receberá o valor igual a 1 se o exemplo de notícia possui a palavra "a"; caso contrário, a *feature* é igual a 0 (zero).
- A segunda *feature* $x^{(2)}$ receberá o valor igual a 1 se o exemplo de notícia possui a palavra "aaron"; caso contrário, a *feature* é igual a 0 (zero).
- A *feature* $x^{(20.000)}$ na posição 20.000 receberá o valor igual a 1 se o exemplo de notícia conter a palavra "zulu"; caso contrário, a *feature* é igual a 0 (zero).

Figura 2.5 – Exemplo de aplicação de *Bag of Wordks*



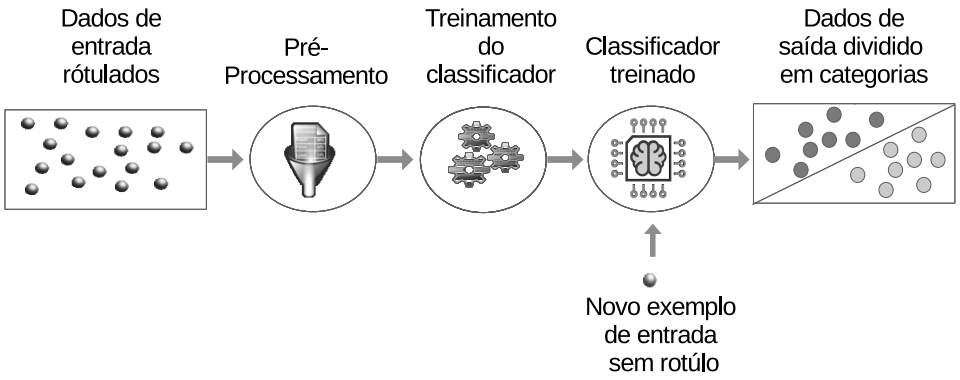
Fonte: O autor

Para cada novo exemplo de notícia será repetido o processo citado anteriormente, em que é conferido a existência de cada palavra no texto , que no final será 10.000 vetores de características, onde cada um terá um número de dimensões de 20.000 *features*, e um rótulo de classe (*fake* ou *not fake*).

De acordo com Monard e Baranauskas (2003) o objetivo do algoritmo de aprendizado supervisionado é construir um classificador que determine corretamente a classe de novos exemplos que não possuem rótulo de classe.

Na Figura 2.6 é ilustrado o processo de construção de um classificador, a partir do pré-processamento dos dados com rótulo de classe, que envolve a limpeza e preparação. Após isso, os dados são usados como entrada para um algoritmo de Aprendizado de Máquina supervisionado de modo a gerar um classificador durante o treinamento. O classificador gerado pode atribuir uma classe para os novos dados que não possuem rótulo de classe.

Figura 2.6 – Aprendizado supervisionado.



Fonte: O autor.

Na literatura são apresentados diversos métodos de aprendizado supervisionado, entre eles, os mais conhecidos são: *Naive Bayes* (NB) (RISH; others, 2001), K-vizinhos Mais Próximos, traduzido do inglês *k-Nearest Neighbour* (KNN) (CUNNINGHAM; DELANY, 2020), Árvores de Decisão, traduzido do inglês *Decision Tree* (DT) (SAFAVIAN; LANDGREBE, 1991), Máquina de Vetores de Suporte, traduzido do inglês *Support Vector Machine* (SVM) (HEARST *et al.*, 1998), assim como os modelos baseados em redes neurais (ANTHONY; BARTLETT, 2009), dentre outros.

2.3.1 *Support Vector Machine*

O SVM é baseado na teoria do aprendizado estatístico, desenvolvido principalmente para classificação binária e cuja classificação (separação) dos exemplos ocorre por meio da construção de um hiperplano, chamado de fronteira de decisão, traduzido do inglês *decision boundary* (HEARST *et al.*, 1998).

A equação 2.1 do hiperplano é representada pelo o produto escalar de $w.x$ em que o valor real do vetor w possui o mesmo número de dimensões do vetor de características x de um determinado exemplo, além disso, um número real igual a b, sendo que corresponde à distância do hiperplano em relação a origem, sendo $\frac{b}{||w||}$, formalizando:

$$\mathbf{wx} - b = 0, \tag{2.1}$$

onde a expressão wx significa $w^{(1)}x^{(1)} + w^{(2)}x^{(2)} + ... + w^{(D)}x^{(D)}$, e D é o número de dimensões do vetor de características x (BURKOV, 2019).

Para predizer um rótulo (classe) y para qualquer vetor de características x que for usado como entrada é formalizado pela equação 2.2 como:

$$y = sgn(\mathbf{wx} - b), \tag{2.2}$$

onde o sgn representa a função sinal que pode tomar qualquer valor como entrada e retornar +1 se é um número positivo e -1 se é um número negativo, sendo tal função sinal empregada na predizer os rótulos de classe (BURKOV, 2019).

Segundo Burkov (2019), o objetivo do algoritmo de Aprendizado de Máquina SVM é tomar um conjunto de dados e encontrar os valores ótimos de w^* e b^* para os parâmetros w e b. A partir disso, o modelo $f(x)$ é então definido como a equação 2.3:

$$f(x) = sgn(\mathbf{w}^*\mathbf{x} - b^*), \tag{2.3}$$

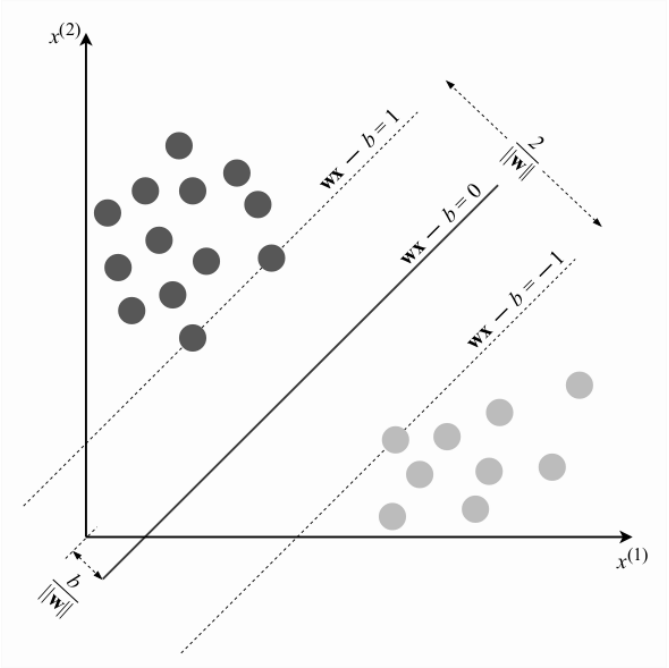
Para predizer se um exemplo de notícia é falsa ou não usando o modelo SVM, deve tomar como entrada o texto da notícia, converter para um vetor de característica, então multiplicar esse vetor por w^*, subtrair de $b*$ e pegar o sinal do resultado de $sign$. No final se tem a predição para +1 ou -1 (BURKOV, 2019).

De acordo com Burkov (2019), para resolver o problema de otimização, de forma que seja encontrado $w*$ e $b*$, as restrições aplicadas são:

- $\mathbf{wx}_i - b \geq 1$ se $y_i = +1$, e
- $\mathbf{wx}_i - b \leq -1$ se $y_i = -1$

Conforme ilustrado na Figura 2.7 , para um vetor de características de duas dimensões é possível visualizar a margem que separa os exemplos positivos(pontos escuros) dos exemplos negativos(pontos claros), sendo que a mesma contribui para se ter uma melhor generalização, fazendo com que o algoritmo(modelo) classifique bem os novos exemplos futuros. A linha expressada pela equação $\mathbf{wx} - b = 0$ é o limite(fronteira) de decisão e os hiperplanos paralelos são dados pelas equações $\mathbf{wx} - b = 1$ e $\mathbf{wx} - b = -1$. A distância entre os hiperplanos é dada por $\frac{2}{||w||}$.

Figura 2.7 – Um exemplo do algoritmo SVM para vetores de características de duas dimensões.



Fonte: Burkov (2019)

2.3.2 *Random Forest*

O paradigma de aprendizado *ensemble* (combinação) foca em treinar um grande número de algoritmos de Aprendizado de Máquina e combinar as predições encontradas pelos classificadores para obter um modelo (classificador) com alta acurácia (BURKOV, 2019).

Uma *Random Forest* (RF) é um classificador *ensemble* baseado em um conjunto de árvores de decisão, em que cada árvore depende de uma coleção de variáveis aleatórias. Cada árvore de decisão construída é usada para classificar um novo exemplo através do voto majoritário (CUTLER; CUTLER; STEVENS, 2012).

Segundo Cutler, Cutler e Stevens (2012) assume-se uma distribuição conjunta $P_{xy}(x, y)$, sendo x representado por um vetor aleatório de características, conhecido como variáveis preditoras e y uma variável aleatória que representa a saída (classe) esperada. O objetivo é encontrar uma função $f(x)$ para prever y, determinada por uma função de perda (custo) $l(y, f(x))$ que é uma medida de quão perto $f(x)$ está de y e definida de modo a minimizar o valor esperado de perda. A expectativa (valor esperado) E em relação a distribuição conjunta é representada pela equação 2.4:

$$E_{xy}(l(y, f(x))) \tag{2.4}$$

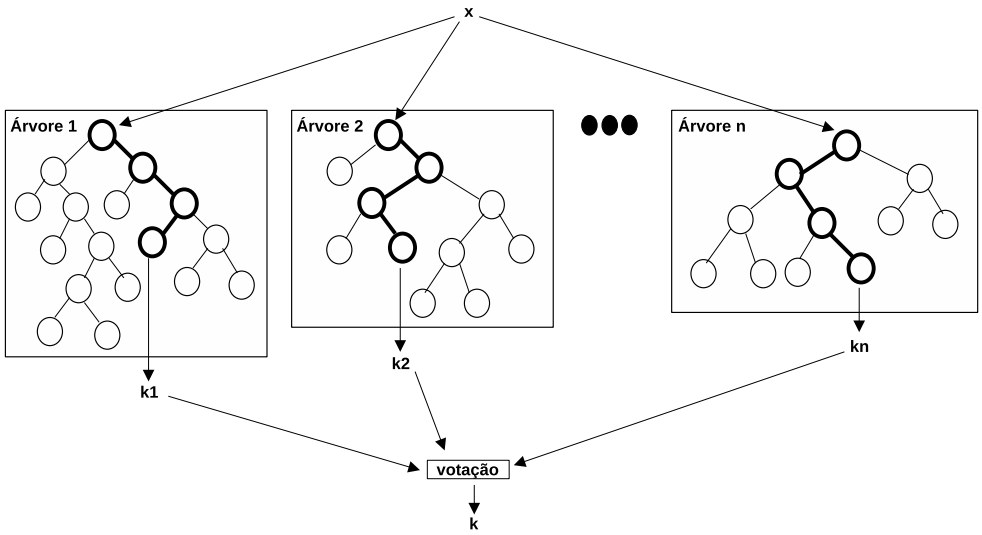
Os valores de $f(x)$ que estão muito longe de y são penalizados. Para problemas de classificação é utilizada a função de perda $0-1$, traduzido do inglês *zero-one loss function*,

descrito na equação 2.5 (CUTLER; CUTLER; STEVENS, 2012):

$$l(y, f(x)) = \begin{cases} 0 & \text{se } y = f(x) \\ 1 & \text{se } y \neq f(x) \end{cases} \tag{2.5}$$

De acordo com Cutler, Cutler e Stevens (2012) as combinações (*ensembles*) constroem f em termos de uma coleção de "algoritmos de aprendizado básico" $h_1(x), ..., hj(x)$ que são associados para gerar a "combinação predita" (*ensemble predictor*) $f(x)$. Em problemas de classificação, $f(x)$ é a classe mais predita, ou seja, a classe mais "votada" k, conforme é ilustrado na Figura 2.8.

Figura 2.8 – Arquitetura do classificador *random forest*



Fonte: Verikas *et al.* (2016)

2.3.3 *AdaBoost*

O algoritmo *Adaptive Boosting* (AdaBoost), é um algoritmo de Aprendizado de Máquina que combina vários "classificadores fracos" e integra eles em um classificador mais robusto, ou seja, um modelo com a melhor habilidade de classificação de texto (LI; XIAO; ZHANG, 2018).

Segundo Hastie *et al.* (2009), esse processo iterativo do *AdaBoost* se inicia com uma amostra de treinamento sem fornecer um valor de peso (*weight*) para construir um classificador e obter os resultados da saída (classe esperada). Se um exemplo é classificado de forma errada, o peso dos exemplos no conjunto de dados é incrementado, processo conhecido como *boosting*. O próximo classificador é construído utilizando os novos pesos, que não são iguais. O processo é repetido e uma pontuação (*score*) é atribuído para cada clas-

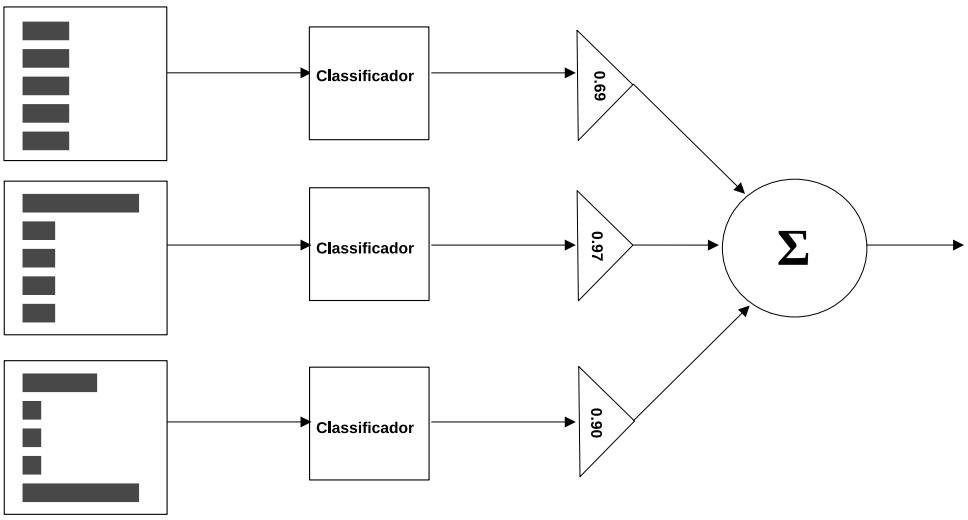
sificador, e o classificador final é definido como uma combinação linear de todos outros classificadores das etapas anteriores.

Ao final de todas as etapas $t = 1, \ldots T$, o classificador final é calculado através da votação majoritária ponderada dos classificadores fracos h_t que receberam um peso α_t, conforme é apresentado na equação 2.6 (SCHAPIRE, 2013).

$$F(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{2.6}$$

Na Figura 2.9 ilustra uma representação esquemática do *AdaBoost*, com o conjunto de dados a esquerda, as diferentes barras representando os pesos aplicados em cada etapa. As previsões dos classificadores são ponderadas, expressas pelos triângulos (valores de α). A saída de cada triângulo é somada no círculo que produz a saída final.

Figura 2.9 – Representação esquemática do *AdaBoost*



Fonte: Harrington (2012)

2.4 Aprendizado Semi-Supervisionado

Como próprio o nome revela, o aprendizado semi-supervisionado está em algum lugar entre o aprendizado supervisionado e não supervisionado e é utilizado para classificação de dados parcialmente rotulados (ZHU *et al.*, 2009). O aprendizado semi-supervisionado é aplicado em tarefas de classificação onde somente apenas uma parte dos exemplos do conjunto de dados de treinamento possui um rótulo de classe (FACELI *et al.*, 2011).

De acordo com Faceli *et al.* (2011), no aprendizado semi-supervisionado o conjunto de dados de treinamento que se constitui de uma porção de exemplos com rótulo de classe e de outra porção de exemplos sem rótulo de classe, é utilizado como entrada do algoritmo

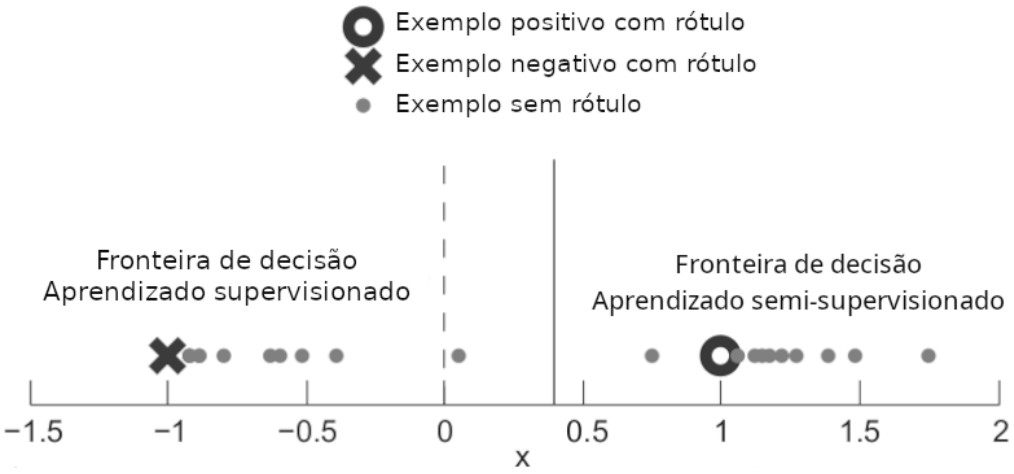
durante a etapa de treinamento. Seu objetivo é gerar um classificador que consiga atribuir um rótulo de classe para os exemplos que não possui, repetindo essa rotina até que todos os exemplos que não possuem rótulo de classe sejam rotulados. No conjunto de treinamento os exemplos sem rótulo de classe representam a maior parte do exemplos, diferente do aprendizado supervisionado que exige que todos os exemplos tenham rótulo de classe (SILVA, 2016).

Há uma escassez de exemplos rotulados em grande parte das tarefas de aprendizado, tornando a etapa manual de anotação de classes, bastante difícil pois necessitam de anotadores humanos ou dispositivos especiais, como, por exemplo, o sistema Appen Smart Labeling[3] (ZHU *et al.*, 2009).

Segundo Zhu *et al.* (2009), o aprendizado semi-supervisionado consegue um melhor desempenho comparado ao aprendizado supervisionado, quando aplicado em problemas de classificação onde o conjunto de dados possui poucos exemplos com rótulo de classe. Além disso, reduz o esforço com anotações feitas manualmente e consequentemente os custos com dispositivos especiais.

Na Figura 2.10, um simples caso de problema de classificação utilizando aprendizado supervisionado, descrito em Zhu *et al.* (2009), é ilustrado, onde cada exemplo (instância) é representado por um vetor de características $x \in \mathbb{R}$, com duas classes: positiva e negativa. No aprendizado supervisionado a fronteira (limite) de decisão (*threshold*) é $x = 0$, tendo como entrada duas instâncias de treinamento: $(x_1, y_1) = (-1, -)$ o símbolo "X", e $(x_2, y_2) = (1, +)$ o símbolo "O". As instâncias sem rótulo são representadas na forma de pontos e somente serão classificadas como positivas se $x > 0$ e negativas se $x < 0$. No caso do aprendizado semi-supervisionado, a fronteira de decisão é $x \approx 0.4$, considerando que $p(x|y)$ é um distribuição Gaussiana, de modo que as instâncias concentram em um média central.

Figura 2.10 – Um exemplo para demonstrar como aprendizado semi-supervisionado é possível.



Fonte:(ZHU *et al.*, 2009)

[3] https://appen.com/solutions/smart-labeling/

Segundo Zhu *et al.* (2009), há dois tipos de aprendizado semi-supervisionado de acordo com o objetivo:

- **Aprendizado semi-supervisionado indutivo**: busca predizer a classe de novos exemplos no conjunto de testes, semelhante ao aprendizado supervisionado. Onde dado um exemplo l (*labeled*) com rótulo $\{(x_i, y_i)\}_{i=1}^{l}$ ou um exemplo u (*unlabeled*) sem rótulo $\{x_j\}_{j=l+1}^{l+u}$ para treino, induz um modelo aprender a função $f : x \rightarrow y$ considerando que f seja um classificador treinado para predizer exemplos futuros.
- **Aprendizado semi-supervisionado transdutivo**: busca predizer as classes dos exemplos sem rótulo do conjunto de treinamento. Onde dado um exemplo l (*labeled*) com rótulo $\{(x_i, y_i)\}_{i=1}^{l}$ ou um exemplo u (*unlabeled*) sem rótulo $\{x_j\}_{j=l+1}^{l+u}$ para treino, faz com que um modelo aprenda uma função $f : X^{l+u} \rightarrow Y^{l+u}$ de modo que f seja um classificador para os exemplos sem rótulo.

Uma analogia para diferenciar o aprendizado semi-supervisionado indutivo e o aprendizado semi-supervisionado transdutivo, é que o primeiro é semelhante a uma prova sem consulta, onde o aluno deve se preparar para todas as possíveis questões, pois não tem conhecimento prévio de quais irão estar na prova, enquanto o segundo é uma prova com consulta, sem necessidade do aluno se preparar, pois, ele conhece todas as questões (ZHU *et al.*, 2009).

A família de algoritmos de aprendizado semi-supervisionado é representada por diferentes métodos na literatura de Aprendizado de Máquina, incluindo modelos generativos probabilísticos (CHAPELLE; SCHLKOPF; ZIEN, 2010), *self-training*, *co-training*, modelos baseados em grafos (*graph-based*) (ZHU *et al.*, 2009), SVM semi-supervisionado (AMINI; USUNIER, 2015), e assim por diante (ZHU *et al.*, 2009).

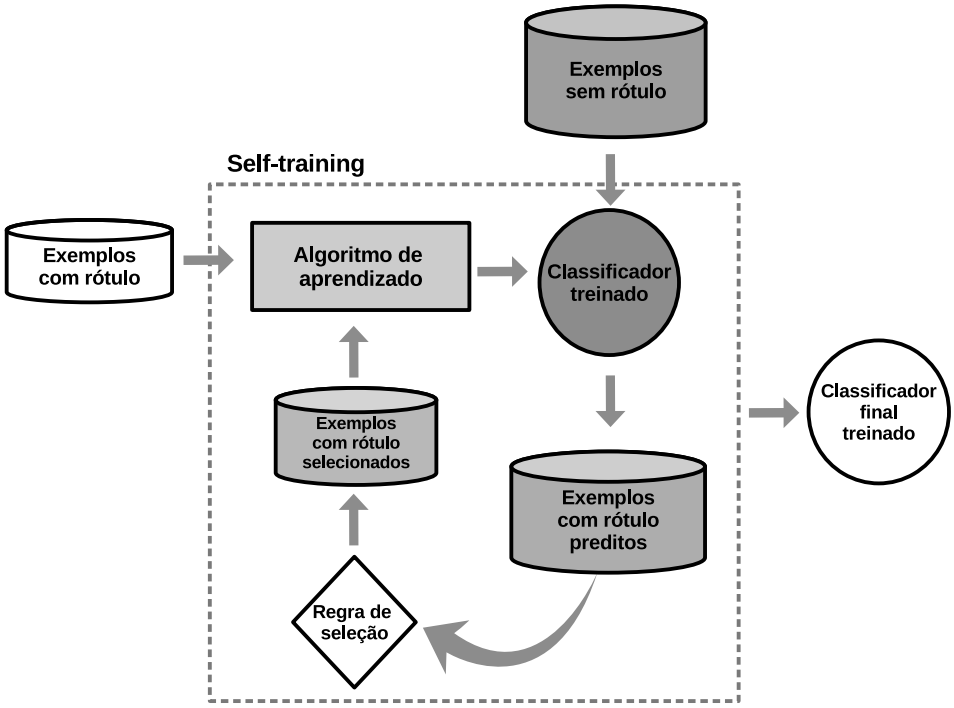
2.4.1 *Self-training*

O *self-training* tem como característica de aplicar o processo de aprendizagem usando suas próprias predições para ensinar a si mesmo. Esse algoritmo também é chamado de *self-teaching* ou *bootstrapping* (mas não é o *bootstraping aggregating*, conhecido como *bagging*) (ZHU *et al.*, 2009). O *self-training* pode ser indutivo ou transdutivo, dependendo da natureza da aplicação.

Conforme é ilustrado na Figura 2.11, o processo de aprendizagem do algoritmo de *self-training* se inicia com o treino de um classificador inicial, usando uma quantidade menor de dados para treino. Após o treinamento, o classificador é usado para predizer o rótulo de classe a partir do conjunto de dados de exemplos sem rótulo. Uma regra de seleção (função de decisão) é aplicada para definir quais exemplos sem rótulo e seus previstos rótulos de classe estão qualificados para ser usados para treinar novamente o classificador, até que

atinja um determinado critério, como, por exemplo, uma convergência do valor de uma determinada métrica (CHAPELLE; SCHLKOPF; ZIEN, 2010).

Figura 2.11 – Arquitetura do *Self-Training*



Fonte:(HASSANZADEH *et al.*, 2018)

As vantagens do *self-training* é sua simplicidade e seu processo de execução que busca "incorporar"(*wrapping*) um algoritmo de aprendizado supervisionado sem alterar seu funcionamento interno. O *self-training* pode incorporar desde um algoritmo KNN ou outro mais complexo. Essa forma de execução é importante, pois, há várias tarefas no mundo real, como, por exemplo, o sistema para detecção de objetos (ROSENBERG; HEBERT; SCHNEIDERMAN, 2005) , onde os algoritmos de aprendizado supervisionado podem ser considerados como caixas pretas complexas, que não são sujeitos a mudanças (ZHU *et al.*, 2009).

Segundo Rosenberg, Hebert e Schneiderman (2005), uma desvantagem é o problema que pode ocorrer pelo algoritmo de aprendizado supervisionado incorporado pelo *self-training*, quando se tem poucos dados com rótulo de classe para treino, fazendo que com seja gerado dados incorretamente rotulados.

Mais informações a respeito do *self-training* podem ser encontradas na literatura em trabalhos de Scudder (1965), Fralick (1967) e Agrawala (1970).

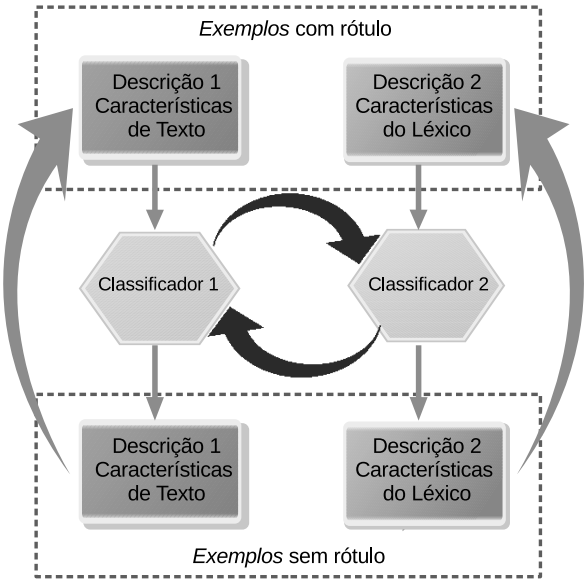
2.4.2 *Co-training*

Diferente do *self-training*, o algoritmo de aprendizado semi-supervisionado *co-training* usa dois classificadores que tem como entrada duas diferentes descrições de características

(ALBALATE; MINKER, 2011). O propósito é usar diferentes descrições (*views*) de características dos exemplos de um conjunto de dados, como, por exemplo, uma *webpage* é formada tanto por textos (*1ª descrição*) das publicações quanto por textos (*2ª descrição*) que representam *hypelinks*. A partir disso é possível treinar dois classificadores, um para cada descrição. Primeiro, uma pequena quantidade de exemplos com rótulo de classe são usados para treinar os classificadores. Após o treinamento, estes são utilizados para classificar exemplos sem rótulo de classe, de modo que as saídas dos classificadores sejam combinadas, onde os exemplos com maior grau de probabilidade ou pontuação (*score*) serão adicionados ao conjunto de dados com exemplos com rótulo (CHAPELLE; SCHLKOPF; ZIEN, 2010). O processo é repetido até se esgotar os exemplos do conjunto de dados sem rótulo de classe.

O processo de treinamento do algoritmo *co-training* utilizando um conjunto de dados cujas descrições são características textuais e características do léxico[4] de cada exemplo, é ilustrado na Figura 2.12. Os exemplos necessitam de duas descrições {Descrição 1, Descrição 2} e são entrada para o algoritmo durante a etapa de treinamento dos classificadores {Classificador 1, Classificador 2}. Os exemplos processados e com probabilidade de ter um rótulo de classe são adicionados a porção de exemplos com rótulo para que ambos classificadores sejam treinados novamente. Um exemplo com rótulo de classe que foi predito pelo Classificador 1, pode ser reclassificado pelo Classificador 2, de modo que ambos classificadores colaborem entre si.

Figura 2.12 – A relação entre os dois classificadores durante o processo de treinamento



Fonte:(SILVA; COLETTA; HRUSCHKA, 2016)

Do mesmo modo que o *self-training*, o *co-traning* é um método "*wrapping*", isto quer

[4] É o estudo das propriedades das palavras e sentenças usadas em uma linguagem e de como elas produzem o vocabulário da linguagem.

dizer, que ele não se preocupa com quais algoritmos de aprendizado supervisionado são utilizados para os dois classificadores de cada descrição, desde que o algoritmo seja capaz de fornecer um grau de probabilidade ou pontuação (*score*) para cada exemplo sem rótulo de classe que seja fornecido como entrada do algoritmo (ZHU *et al.*, 2009).

Segundo Albalate e Minker (2011), a desvantagem do *co-traning* é a necessidade de compatibilidade entre os dois classificadores, de modo que se evite que um exemplo receba rótulo de classe de um classificador e seja removido o rótulo pelo outro classificador, ou seja, ambos devem concordar com suas predições. Como vantagem é menos suscetível a erros em relação ao *self-training*, por fazer uso de dois classificadores.

2.5 *Positive and Unlabeled Learning*

É um algoritmo de aprendizado semi-supervisionado que considera que dado um conjunto de dados com exemplos com rótulo de classe de uma particular classe P (chamada de classe positiva) e um conjunto de dados de exemplos U (*unlabeled*) sem rótulo de classe, que contém ambos exemplos de classe P e não P (chamada de classe negativa), seu objetivo é gerar um classificador para predizer a classe positiva de um exemplo U.

PU *learning* é considerado como um caso especial do tradicional aprendizado semi-supervisionado, sendo que as principais diferenças são que o conjunto de dados de treinamento, no aprendizado semi-supervisionado, é composto por exemplos de todas as classes e que a abordagem vai além das tarefas de classificação binária (BEKKER; DAVIS, 2020).

De acordo com Bekker e Davis (2020), o objetivo da classificação binária é treinar um modelo que seja capaz de distinguir entre exemplos positivos e negativos. O treinamento de um modelo a partir de exemplos com rótulo de classe positiva e de exemplos sem rótulo de classe, é conhecido como PU *learning*. O PU *learning* é uma variação da classificação binária onde o conjunto de dados possui exemplos com rótulo de classe positiva e exemplos sem rótulo de classe. Os exemplos sem rótulo de classe podem ser categorizados tanto como exemplos com rótulo de classe positiva, quanto como exemplos com rótulo de classe negativa.

Um conjunto de dados de treinamento padrão para um problema de classificação binária, é composto de um subconjunto de exemplos positivos x que possui um rótulo $y = 1$ e outro subconjunto de exemplos negativos com rótulo $y = 0$. Imagine um conjunto de treinamento que possui um subconjunto com poucos exemplos positivos e um subconjunto maior de exemplos sem rótulo, sendo que o último pode conter tanto exemplos positivos quanto exemplos negativos. O desafio do PU *learning* é treinar um classificador dado um conjunto de dados de treinamento desbalanceado, onde a classe majoritária é representada pelos exemplos do menor subconjunto (de positivos) e a classe minoritária pelos exemplos sem rótulo do maior subconjunto (ELKAN; NOTO, 2008).

Para treinar um classificador binário, é necessário um conjunto de exemplos para treinamento, onde cada exemplo é dado por uma tupla (x, y), sendo o x o vetor de atributos (características) e y o valor da classe, ou seja, seu rótulo. No aprendizado supervisionado, o conjunto de dados de treinamento e o valor de y são completamente preenchidos em todos os exemplos. No caso do PU *learning* cada exemplo é representado por uma tripla (x, y, s), nesse caso o valor de s informa se a tupla foi selecionada para ter um rótulo de classe, ignorando o valor de y, ou seja, s representa uma variável "controle"para um exemplo possuir rótulo de classe. Se o exemplo é rotulado, seu valor de $s = 1$, então este pertence à classe positiva: $\Pr(y = 1|s{=}1) = 1$. Caso contrário, se $s = 0$, então o exemplo pode ser de qualquer classe (positiva ou negativa) (BEKKER; DAVIS, 2020).

Na Tabela 2.1 é ilustrado as 5 primeiras linhas do conjunto de dados de treinamento, onde cada linha representa o vetor de atributo x = [texto].

Tabela 2.1 – Exemplo de conjunto de treinamento com rótulo de classe

texto	y
Procuradoria-Geral da Fazenda indica que 2016 ...	0
Colunista do Globo diz que Cunha tem banco de ...	1
Vídeo flagra deputado do PT agredindo sargento...	0
Bolsonaro pode ser preso por suas declarações ...	1

Fonte: Adaptado de Bekker e Davis (2020)

Na Tabela 2.2 é ilustrado como é representado os mesmos exemplos da Tabela 2.1, considerando que na Tabela 2.2, o valor de s que determina se o exemplo possui ou não possui um rótulo de classe y.

Tabela 2.2 – Exemplo de conjunto de treinamento PU

texto	y	s
Procuradoria-Geral da Fazenda indica que 2016 ...	?	0
Colunista do Globo diz que Cunha tem banco de ...	1	1
Vídeo flagra deputado do PT agredindo sargento...	?	0
Bolsonaro pode ser preso por suas declarações ...	?	0

Fonte: Adaptado de Bekker e Davis (2020)

Segundo Bekker e Davis (2020), dois tipos de cenários que os exemplos de PU podem originar:

- ***Single-training-set***: exemplos com rótulo de classe positiva e sem rótulo vem do mesmo conjunto de dados. Exemplo: Anúncios personalizados onde os usuários apenas clicam em um subconjunto dos anúncios do seu interesse.
- ***Case-control***: exemplos com rótulo de classe positiva e sem rótulo pertencem a conjunto de dados distintos. Ex: Previsão da situação socioeconômica de uma pessoa conforme o histórico de saúde, onde os exemplos positivos podem ser obtidos em centros

de saúde localizados em bairros de classe alta e exemplos sem rótulos obtidos de uma seleção aleatória de centros de saúde de classe média e baixa.

O cenário *Single-training-set* atraia mais atenção na literatura, em relação aos cenários de configuração do PU *learning*. Uma vez definido o tipo de cenário de configuração a ser aplicado é fundamental entender qual o mecanismo de classificação a ser utilizado, pois é o conceito principal em PU *learning*.

2.5.1 Mecanismo de classificação (*labelling*)

Segundo Bekker, Robberechts e Davis (2019), o mecanismo de classificação para PU *learning* determina quais exemplos positivos irão receber rótulo de classe. Os exemplos positivos são selecionados de um conjunto completo (população) de exemplos positivos através de um mecanismo de classificação probabilística, onde cada exemplo x tem a probabilidade $e(x) = \Pr(s = 1|y = 1, x)$ de ser selecionado para ser rotulado, mecanismo conhecido como escore de propensão e, traduzido do inglês *propensity score*. Sendo assim, uma distribuição com rótulo de classe é uma versão enviesada da distribuição positiva[5], conforme a equação 2.7, com $f_l(x)$ representando a função de densidade de probabilidade dos exemplos com rótulo de classe e $f_+(x)$ representando a função de densidade de probabilidade dos exemplos positivos.

O escore de propensão de x é descrito como $e(x)$ que é a probabilidade de atribuir um rótulo de classe positiva para exemplos com atributos (características) x. Enquanto que a frequência de rótulo c, traduzido do inglês *label frequency*, conhecida também como *class prior* é a probabilidade de selecionar um exemplo positivo para atribuir um rótulo de classe positiva, onde $c = \Pr(s = 1|y = 1)$.

$$f_l(x) = \frac{e(x)}{c} f_+(x), \tag{2.7}$$

Para permitir o treinamento é necessário fazer suposições sobre o método de rotular ou de distribuição das classes. As duas possibilidades de explicar porque um exemplo está sem rótulo de classe é:

- Ou ele é verdadeiramente um exemplo negativo ou;
- Ele é um exemplo positivo, mas não foi selecionado pelo mecanismo de classificação para ter o rótulo de classe explícito.

As suposições a respeito de como os exemplos positivos observados são selecionados, são:

[5] É uma assimetria positiva que corresponde ao grau de afastamento de uma distribuição normal em relação ao seu eixo de simetria, que ocorre do lado esquerdo.

- ***Selected Completely At Random* (SCAR) - Selecionado completamente ao acaso**: É uma suposição onde o mecanismo de classificação não depende dos atributos do exemplo e nem da probabilidade do exemplo ser positivo, sendo que cada exemplo positivo tem a mesma probabilidade de ser classificado. Esse método foi formalizado primeiro por Elkan e Noto (2008) que notou a similaridade do cenário de PU com o problema geral de aprendizado a partir de dados ausentes (BEKKER; ROBBERECHTS; DAVIS, 2019).

- ***Selected At Random* (SAR) - Selecionado ao acaso**: É uma suposição onde o mecanismo de classificação depende dos valores dos atributos do exemplo. Sendo fornecidos os valores dos atributos, o mecanismo de classificação não depende da probabilidade do exemplo ser positivo (BEKKER; ROBBERECHTS; DAVIS, 2019).

- ***Selected Not At Random* (SNAR) - Selecionado não aleatoriamente**: É uma suposição onde o mecanismo de classificação depende dos valores dos atributos do exemplo. Sendo fornecidos os valores dos atributos, o mecanismo de classificação depende da probabilidade do exemplo ser positivo (BEKKER; ROBBERECHTS; DAVIS, 2019).

As suposições a respeito de como ocorre a distribuição dos dados, são:

- ***Negativity* (Negatividade)**: Define que todos os exemplos sem rótulo de classe pertencem à classe negativa, apesar do fato de que essa suposição obviamente não se sustenta, pois, os exemplos sem rótulo de classe podem ser tanto positivos quanto negativos, mas mesmo assim essa suposição é frequentemente usada na prática (BEKKER; DAVIS, 2020).

- ***Separability* (Separabilidade)**: Define que duas classes são naturalmente separáveis, isto é, que existe um classificador que pode distinguir perfeitamente os exemplos positivos dos negativos. Para isso, há uma função f no espaço de hipóteses que mapeia todos os exemplos positivos para um valor maior ou igual a um limite (*threshold*) τ e exemplos negativos para um valor menor para um limite τ (BEKKER; DAVIS, 2020).

- ***Smoothness* (Suavidade)**: Define que os exemplos próximos uns dos outros têm maior probabilidade de ter o mesmo rótulo de classe. Se dois exemplos x_1 e x_2 são similares, logo as probabilidades $\Pr(y = 1|x_1)$ e $\Pr(y = 1|x_2)$ serão também semelhantes (BEKKER; DAVIS, 2020).

2.5.2 Métricas para avaliação dos algoritmos de PU *learning*

Alguns conceitos são importantes para entender a métrica $F1$ *score*, como por exemplo a matriz confusão. A matriz confusão é uma tabela que sumariza o sucesso de predições do classificador, sendo o lado que representa as linhas indica o rótulo (classe) real e o lado que representa as colunas indica o rótulo previsto pelo classificador (BURKOV, 2019).

Segundo Burkov (2019), a matrix confusão representa as predições de um classificador para exemplos de notícias como *fake news* e como não *fake news* conforme a Tabela 2.3. A matrix confusão mostra que de 24 exemplos que são *fake news*, o classificador previu corretamente 23 como *fake news*, isto que dizer, que se tem 23 verdadeiros positivos ou VP = 23 e incorretamente previu um exemplo como não *fake news*, ou seja, se tem 1 falso negativo ou FN = 1. Enquanto que em 568 exemplos que não são *fake news*, foram classificados corretamente 556 (556 verdadeiro negativos ou VN = 556) e incorretamente foram 12 (12 falso positivos ou FP = 12).

Tabela 2.3 – Matriz Confusão

	fake news (previsto)	não *fake news* (previsto)
fake news (real)	23 (VP)	1 (FN)
não *fake news* (real)	12 (FP)	556 (VN)

Através da matriz confusão que se calcula duas métricas que são: precisão e *recall*. A precisão p é a taxa de positivos verdadeiros sobre o número de positivos previstos pelo classificador, conforme a equação 2.8. *Recall* r é a taxa de positivos previstos pelo classificador em relação o número total de positivos no conjunto de dados, conforme a equação 2.9 (BURKOV, 2019):

$$p = \frac{VP}{VP + FP}, \tag{2.8}$$

$$r = \frac{VP}{VP + FN}. \tag{2.9}$$

De acordo com Mueller e Massaron (2016), as métricas precisão e *recall* são muito importantes , mas olhando somente uma delas não se alcança um resultado completo. O F_1 *score* ou F-*measure* que é média harmônica entre ambas métricas pode ser definido como um resultado sumarizado delas. Além disso, o F_1 *score* é a melhor métrica em problemas de classificação binária que faz uso de conjunto de dados desbalanceados.

De acordo com Bekker e Davis (2020) a métrica mais usada para avaliar os algoritmos de PU *learning* é o F_1 *score*, a qual é definida conforme a equação 2.10 :

$$F_1(\hat{y}) = \frac{2pr}{p + r}, \tag{2.10}$$

com precisão $p = Pr(\mathbf{y} = 1|\hat{\mathbf{y}} = 1)$, que define a probabilidade condicional de um exemplo ter rótulo da classe positiva $y = 1$ dado que o rótulo estimado é da classe positiva $\hat{y} = 1$ e *recall* $r = Pr(\hat{\mathbf{y}} = 1|\mathbf{y} = 1)$, que define a probabilidade condicional de um exemplo ter rótulo estimado da classe positiva $\hat{y} = 1$, dado que o rótulo é da classe positiva $y = 1$. Em relação a suposição SCAR, o $recall$ pode ser estimado, mas a precisão não pode.

2.5.3 Métodos de classificação baseados em PU *learning*

Segundo Bekker e Davis (2020), os métodos que abordam PU *learning* podem ser divididos em três categorias: *two-steps* (dois passos), *biased learning* (aprendizado enviesado) e incorporação da *class prior*, conforme a seguir:

- **Técnica de dois passos**: o primeiro passo é identificar os exemplos negativos corretos (confiáveis) presentes no grupo de exemplos sem rótulo, pois aqueles que tem rótulo são definidos como positivos confiáveis. O segundo passo é treinar um classificador baseado nos exemplos positivos e exemplos negativos corretos. Usa a suposição de *separability* e *smoothness*, de modo que assume que todos os exemplos positivos são similares aos exemplos com rótulo e os exemplos negativos são bem diferentes deles (BEKKER; DAVIS, 2020).

- ***Biased learning***: trata os exemplos sem rótulo como exemplos negativos, cujos valores dos rótulos de classe possuem informações inconsistentes, definidos como ruídos. Esses valores inconsistentes para exemplos negativos são uma constante que representa um valor do nível de erro da classificação. Esta técnica usa a suposição SCAR e a grande parte dos métodos são baseados em SVM. O SVM enviseado é um método SVM que penaliza diferentemente os exemplos positivos e negativos classificados incorretamente (BEKKER; DAVIS, 2020).

- **Incorporação da class prior**: Usa a suposição SCAR, de modo que seja usada a *class prior*. Essa técnica usa três métodos: pré-processamento, pós-processamento e método de modificação. O primeiro método muda o conjunto de dados conforme a *class prior*, criando um novo conjunto de dados com todos os exemplos com rótulo de classe para ser usado por outros algoritmos de Aprendizado de Máquina. O segundo método treina um classificador probabilístico considerando os exemplos sem rótulo como exemplos negativos e modifica suas probabilidades. Já o terceiro método modifica os métodos para incorporar a *class prior*, visando fornecer um valor de peso (*weight*) para cada exemplo sem rótulo de classe ou ajustar os algoritmos de Aprendizado de Máquina para predizer os exemplos positivos e negativos no conjunto de dados de exemplos sem rótulo (BEKKER; DAVIS, 2020).

Para escolher um método de aprendizagem de PU *learning* deve-se verificar quais suposições são prováveis de serem válidas para a aplicação em questão. Se a suposição separabilidade é considerada, é provável o uso de técnicas de dois passos. Se a suposição SCAR é considerada, então deve-se usar *biased learning* ou métodos que incorporam a técnica de *class prior*. Caso a suposição da separabilidade e a suposição SCAR forem consideradas, vai depender o quão separadas as duas classes se encontram. No caso em que as classes são separáveis, considerando estar próximas uma das outras, separar as duas classes correta-

mente é complexo para a técnica de dois passos, consequentemente, a suposição SCAR se torna mais eficaz (BEKKER; DAVIS, 2020).

Ao considerar que o volume de dados sem rótulo de classe é extenso na realidade, nota-se a escassez de encontrar exemplos com rótulo de classe que representam o teor de uma notícia, por exemplo, se a mesma se classifica como *fake news*, se torna uma tarefa de custo alto para os seres humanos ou até mesmo contrária ao paradigma do aprendizado supervisionado, onde todos os exemplos precisam ter rótulo de classe com mesma proporção para ambas classes. Deste modo algoritmos semi-supervisionados, em destaque o PU *learning* tem um papel importante nessa necessidade de classificar exemplos sem rótulo de classe.

Capítulo 3

TRABALHOS RELACIONADOS

Neste capítulo são apresentados os trabalhos mais relevantes que fazem uso de técnicas de Aprendizado de Máquina com foco no PU *learning* aplicados para classificação de textos, detecção de *fake news* e *fake reviews*. Embora seja utilizado como referência o trabalho de Silva *et al.* (2020), o mesmo não se encontra associado aos trabalhos relacionados pois o seu estudo não fez uso de técnicas de Aprendizado de Máquina com foco no PU *learning*. A Seção 3.1 aborda os trabalhos que tratam problemas de classificação de texto. A Seção 3.2 descreve os trabalhos cujo conteúdo é categorizado como *fake news*. A Seção 3.3 denota os trabalhos associados a *fake reviews*. Para finalizar o capítulo é apresentado em tabelas, um sumário dos principais pontos relevantes dos trabalhos analisados.

3.1 Classificação de Texto

Considerando que o tipo de dado é texto, há diversos trabalhos relacionados, como, por exemplo, Liu *et al.* (2002), que definem PU *learning* como problema de classificação parcialmente supervisionada, onde somente os exemplos positivos têm rótulo de classe e os demais não tem, para desenvolver uma nova heurística que consiste em dois passos: 1) usa o algoritmo EM (*Expectation-Maximization*) em conjunto com NB para encontrar a probabilidade de um documento ser da classe positiva, chamado de I-EM (*Initial EM*) e 2) constrói um classificador usando o EM em um conjunto de dados com exemplos com rótulo de classe positiva, negativa e sem rótulo de classe, que no final é chamado de S-EM (*Spy with EM*), que atribui um rótulo de probabilidade para cada exemplo. É usado o conjunto de dados de textos em Inglês chamado *20Newsgroups* e artigos da *Usenet* (LIU *et al.*, 2002). A métrica aplicada é o F_1 *score*.

Denis *et al.* (2003) fizeram um estudo de uma adaptação de NB para PU *learning* em textos em Inglês da *Web* usando o *co-training* (BLUM; MITCHELL, 1998). Para treinar os algoritmos NB foram usados três conjuntos de dados, *The WebKB Course, Reuters-21450* e *20-newsgroups*, que possuem exemplos com rótulo de classe 0 e 1, onde 1 é a classe positiva.

Li e Liu (2003) publicaram um estudo de uso de informações de forma parcial, onde somente uma porção dos exemplos recebem rótulo de classe e a outra não, fazendo a combinação dos algoritmos *Rocchio* e SVM para construir um classificador. Este trabalho realizado por meio de dois passos: 1) extração de exemplos negativos confiáveis (maior probabilidade de serem negativos) do conjunto sem rótulo usando o método *Rocchio* e 2) aplicação do SVM para construir um classificador em um conjunto de dados de textos em Inglês do *Reuters*, avaliando com F_1 *score*.

O trabalho publicado por Liu *et al.* (2003) apresenta um estudo do problema de construir um classificador para duas classes positiva e sem rótulo, usando o método de dois passos, onde: 1) identifica os exemplos negativos confiáveis no conjunto de dados sem rótulo, com os algoritmos S-EM, PEBL (*Positive Example Based Learning*) e Roc-SVM (*Rocchio with SVM*) e 2) constrói um conjunto de classificadores iterativamente usando S-EM com NB, PEBL com SVM e Roc-SVM com SVM, em seguida avalia o F_1 *score*. O conjunto de dados de textos em inglês utilizado foi *Reuters-21578* e também artigos da *Usenet* (LIU *et al.*, 2003). A *feature* utilizada foi o TF-IDF (*Term Frequency–Inverse Document Frequency*).

Já Fung *et al.* (2005) apresentaram um estudo que usa uma heurística de rotulagem (*labeling heuristic*) chamada PNLH (*Positive examples and Negative examples Labeling Heuristic*) que aplica o treinamento nos algoritmos SVM, *Rocchio* e NB, cujo funcionamento se dá a partir de dois passos: 1) "extração"de exemplos negativos de um conjunto de dados sem rótulo e 2) "incremento"de exemplos positivos, criando um hiperplano entres as classes, sendo que os conjuntos de dados usados são somente de textos em Inglês do *Reuters-21578, Newsgroup-20* e *Web-KB*.

Já Yu e Li (2007) apresentam um classificador baseado em PU *learning* com o uso do método de dois passos, combinando método de aprendizado supervisionado baseado em grafos. Classificam a sua técnica como PE-PUC (*Positive Document Enlarging PU Classifier*), sendo: 1)usado NB para extração dos exemplos negativos confiáveis do conjunto de dados de exemplos sem rótulo e 2) criado o classificador final usando algoritmos NB e EM. O conjunto de dados usado é *20newsgroup* de textos em inglês, usando *K-Fold* com 30% de teste e 70% de treino, e a métrica utilizada é F_1 *score*.

Peng, Zuo e He (2008) apresentam um novo classificador usando PU *learning*, treinando diversos classificadores iterativamente e por fim adota-se um critério de eleição entre todos os classificadores treinados para construir a versão final, no lugar de escolher um deles, baseado em PSO (*Particle Swarm Optmization*) que se refere a combinação de pesos. No seu trabalho faz uso do termo PU *Oriented Text Classification*, que se refere a um pequeno conjunto de dados com exemplos com rótulo de classe e um grande conjunto de dados com exemplos sem rótulo de classe. As etapas utilizadas são: 1) identificar o conjunto de exemplos negativos confiáveis do conjunto de exemplos sem rótulo de classe com o algoritmo 1-DNFI (*Disjunctive Normal Forma Improved*) e 2) construir diversos classifica-

dores aplicando SVM iterativamente e, por fim, criar o classificador WV (*Weighted Voting*) e 3) construir o classificador PSO usando método de votação ponderada (*weighted voting*). O conjunto de dados usado foi *Reuters-21,578* de textos em Inglês. A métrica utilizada foi F_1 *score*.

O estudo feito por Lu e Bai (2010) propõe um classificador semi-supervisionado para PU *learning*, onde o processo envolve extrair os exemplos negativos confiáveis no conjunto de dados sem rótulo e adicionar ao conjunto de dados de exemplos positivos, apenas os exemplos extraídos do conjunto de dados sem rótulo, com maior probabilidade de serem positivos. Usa o algoritmo *Rocchio* para extrair os exemplos negativos confiáveis e depois combina os algoritmos *Rocchio* e *K-means* para extrair os exemplos positivos confiáveis do conjunto de dados sem rótulo, e após isso, é usado NB para construção do classificador. Usa o conjunto de dados *TanCorp-12* de textos em Inglês e como métrica o F_1 *score*.

O estudo de Liu e Peng (2014), apresenta um método baseado em agrupamento (*clustering*) para identificar os exemplos que são negativos verdadeiros e em seguida remover os exemplos com maior probabilidade de serem positivos do conjunto de dados sem rótulo. Usam o método de dois passos, onde: 1) aplica o método de *clustering* que remove os exemplos com maior probabilidade de serem positivos de um conjunto de dados sem rótulo e 2) Usa o TF-IDF melhorado chamado TFIPNDF (*Term Frequency Inverse Positive-Negative Document Frequency*) que calcula o IPDF (*Inverse Positive Document Frequency*) e INDF (*Inverse Negative Document Frequency*), em seguida, é criado um classificador de votação ponderada (*weighted*) aplicando iterativamente o algortimo SVM. Na etapa de treinamento são utilizados exemplos positivos, os exemplos negativos verdadeiros e os exemplos sem rótulo. As métricas avaliadas são acurácia, área sob a curva, traduzido do inglês AUC (*Area Under The Curve*) e *Inverse Negative Document Frequency*. No trabalho sãos usados os conjuntos de dados *Reuters Corpus Volume*, *Reuters-21578* e *20 Newsgroups*, onde todos são de textos em Inglês.

3.2 *Fake News*

Vários trabalhos na literatura envolvem o uso do aprendizado supervisionado ou semi-supervisionado para identificar *fake news*. No caso de PU *learning* são pouquíssimos trabalhos, como de Liu e Wu (2020) que apresentam uma rede neural profunda para detectar *fake news* na fase inicial, que extrai características de texto e de perfis de usuários, destacando a resposta de usuários conforme posição (*ranking*) e usando mecanismo de *multi-region mean-pooling* para agregar as características baseadas em vários tamanhos de janela. Os autores utilizam um *framework* de PU *learning*, de modo que para coletar as características usa-se o extrator de características *crowd response* sensível ao status, que combina o texto e o perfil do mesmo, formando um mapa de características que será a entrada do classifica-

dor baseado em RNC (Redes Neurais Convolucionais), traduzido do inglês CNN (*Convolutional Neural Networks*) que usa dois mecanimos: *position-aware attention* e *multi-region mean pooling* para definir a classe, enquanto o *framework* PU *Learning* treina diversos classificadores fracos (*weaks*), usando uma pequena parte de exemplos positivos confiáveis, de forma que fique balanceado e selecione uma subamostra de mesmo tamanho no conjunto de dados de exemplos sem rótulo de classe. Após x vezes de subamostragem é gerado o classificador final, baseado na média da saída dos classificadores fracos, para classificar o conjunto de dados sem rótulo, onde os exemplos positivos encontrados são adicionados ao conjunto de positivos confiáveis, repetindo todo processo de subamostragem, para ampliar o conjunto de dados de exemplos positivos confiáveis. Os autores utilizam um conjunto de dados público do *Twitter* de textos em Inglês e um da *Sina Weibo* (LIU; WU, 2020) de textos em Mandarim, avaliando com as métricas de acurácia, precisão, *recall* e F_1 *score*.

3.3 *Fake Reviews*

Há mais trabalhos que usam aprendizado semi-supervisionado para detectar *fake reviews* destacando o PU *learning*, se comparado a sua aplicação em *fake news*. O trabalho de Li *et al.* (2014) envolve dois estudos, sendo um baseado em um modelo de aprendizado supervisionado e outro em PU *learning* aplicado em um conjunto de dados de um restaurante Chinês, cujo objetivo é aplicar os modelos em avaliações do idioma Chinês. Os autores utilizaram o SVM para aprendizado supervisionado e o PU *learning* utilizando a técnica de dois passos por meio de um sistema disponibilizado publicamente, conhecido como LPU (*Learning from Positive and Unlabeled Examples*), que possui como opção no 1º passo os algoritmos (*Spy*, *Rocchio* e NB) e no 2º passo os algoritmos (EM e SVM), alternando as combinações. As *features* utilizadas são *unigrams*, *bigrams* e TF-IDF, bem como as métricas utilizadas são precisão, *recall* e F_1 *score* baseados somente na classe positiva (*fake*).

Fusilier *et al.* (2015) apresenta um modelo semi-supervisionado e modificado do PU *learning* baseado na técnica de dois passos, usando uma forma diferente de selecionar os exemplos negativos confiáveis em um conjunto de dados com exemplos sem rótulo de classe em relação ao PU *learning* tradicional. Os autores fazem a análise da polaridade de uma *fake news* ser positiva ou negativa, e compara o desempenho do modelo proposto com o uso de *features* como *Word Unigrams* e *Bigrams* e com outros classificadores como SVM e NB. Eles utilizam um conjunto de dados público de opiniões de 20 hotéis populares da cidade de Chicago/EUA e a métrica utilizada é o F_1 *score*.

Já Rout *et al.* (2017) apresentaram um modelo semi-supervisionado para detectar *fake reviews* em um conjunto de dados de avaliações de um hotel, para criar um classificador usando PU *learning*, onde primeiro treina-se um classificador com exemplos positivos e exemplos sem rótulo para ser usado na classificação do conjunto de dados com exemplos

sem rótulo. Repete-se o processo até que o número de exemplos sem rótulo seja menor que processo anterior, e, por fim, na última iteração é gerado o classificador final. Faz-se o uso de *features* como: *Sentiment Polarity*, POS (*Parts of Speech*) *tags*, LIWC (*Linguistic Inquiry and Word Count*) e *Bigram frequency counts*, no conjunto de dados *Amazon Mechanical Turk10* de textos em Inglês e a métrica utilizada é o F_1 *score*.

Foi proposto por Deng *et al.* (2017) um algoritmo baseado em PU *Learning* que leva em conta *features* em dois aspectos: metadados e conteúdo de *reviews*, usando o *K-means* para classificar os dados e, por fim, gerar um classificador baseado em redes de neurais. A *feature* usada para metadado é o tamanho do *review (length review)* e para o conteúdo do *review* é usado *"Text segmentation"*, *"Bag of words"* e *"Autoencoder"*. Usa-se o método de dois passos, onde: 1) usa *K-means* para dividir em grupos, e 2) calcula a porcentagem de *fake reviews* em cada grupo. O conjunto de dados usado foi construído usando a técnica *"crawling"*, extraindo dados da plataforma de *Ecommerce* JD.com (DENG *et al.*, 2017) e a métrica usada foi a acurácia.

Shinde, M. P. e Channe, H. (2018) apresentam um modelo semi-supervisionado para detectar *fake reviews* com métodos *ensemble*, usando *self-training* combinado com outros algoritmos como NB, RF e LR, *co-training*, EM e PU *Learning*. O objetivo principal é executar os algoritmos NB, RF e LR, e aplicar o método *ensemble* nesta abordagem de aprendizado semi-supervisionado, que mostra que o método Twin-SVM com NB *ensemble* supera os outros dois métodos na classificação multi-rótulo e multi-classe. As *features* utilizadas são POS *tags*, LIWC, *Sentiment polarity* e *Bigram frequency count*. Os autores utilizaram um conjunto de dados de sites de reserva de hotéis *online* do Yelp's App, e a métrica avaliada é a acurácia.

Já Shuqin e Jing (2019) propõem um modelo chamado de MDINPUL (*Mixing Population and Individual Nature PU Learning*) dividido em quatro etapas: 1) extrai-se os exemplos negativos confiáveis baseados usando *K-means*, 2) usa-se o LDA (*Latent Dirichlet Allocation*) e *k-means* para calcular várias amostras de exemplos positivos e negativos, e o *Rocchio* para exemplos positivos e negativos, para extrair os 10 mais relevantes, que em seguida são usados para calcular o rótulo de outros exemplos, 3) utiliza-se a idéia de população e indivíduos para determinar o rótulo de classe de cada amostra *"spy"*(espiã), estabelecendo um peso de probabilidade para cada amostra e 4) gera o classificador SVM melhorado. As *features* utilizadas referentes ao texto são *unigram*, POS e LDA. As características referentes ao comportamento do usuário são similaridades e tamanho do texto, número de mensagens por dia, proporção de comentários positivos e *score* de desvio padrão. As características referentes ao relacionamento entre o texto da opinião, do avaliador e o comerciante são avaliados a frequência de avaliações e a similaridade dos itens de interesse. O conjunto de dados usado é de opiniões no idioma Inglês de 99 restaurantes registrados no aplicativo Yelp. As métricas utilizadas são acurácia, precisão, *recall* e F_1 *score*.

A proposta de He *et al.* (2020) é criar um método para detectar *fake reviews* em uma

loja de aplicativos, usando um algoritmo SVM enviesado (*biased*), combinando com *user behavior density*, onde é executado em duas etapas: 1) treinar um classificador a partir dos exemplos positivos, depois treinar o classificador com os exemplos positivos e exemplos sem rótulo, 2) calcular o *behavior density* de usuários, relativo ao tempo de publicação, e o *behavior density* de aplicativos, referente ao tempo de receber opiniões. Os autores usam um conjunto de dados de textos em Mandarim, produzido manualmente com dados da *Apple Store* (HE *et al.*, 2020) e as métricas avaliadas são precisão, *recall* e F_1 *score*.

As Tabelas 3.1 e 3.2 apresentam grande parte dos trabalhos que usam conjuntos de dados com textos em Inglês ou Mandarim dos trabalhos relacionados em *fake news* e *fake reviews*.

Tabela 3.1 – Sumário dos Conjunto de Dados e *Features* Utilizados em Trabalhos Anteriores

Estudo	Ano	Tipo de *Fake*	*Features*	Conjunto de Dados	Metadados	Idioma
Li *et al.* (2014)	2014	*Reviews*	*Standard unigrams and biagrams* *Character n-grams* TF-IDF	*Restaurant reviews hosted Dianping*	Sim	Mandarim
Fusilier *et al.* (2015)	2015	*Reviews*	*Word unigrams and bigrams*	*Hotel reviews from Chiago EUA* (*Ott et al.'s corpora*)	Não	Inglês
Rout *et al.* (2017)	2017	*Reviews*	*Sentiment Polarity* *Parts of Speech* (POS) *tags* *Linguistic Inquiry and Word Count* (LIWC) *Bigram frequency counts*	*Amazon Mechanical Turk10* (AMT)	Não	Inglês
Deng *et al.* (2017)	2017	*Reviews*	*Text Segmentation* *Bag of Words* *Autoencoder*	*Product reviews from JD.com.*	Sim	Inglês
Shinde, M. P. e Channe, H. (2018)	2018	*Reviews*	*Sentiment Polarity* *Parts of Speech* (POS) *tags* *Linguistic Inquiry and Word Count* (LIWC) *Bigram frequency counts*	*Hotel reviews from Yelp App.*	Sim	Inglês
Shuqin e Jing (2019)	2019	*Reviews*	*Unigrams* *Parts of Speech* (POS) *tags* LDA (*Latent Dirichlet Allocation*)	*Restaurant reviews from Yelp App.*	Sim	Inglês
He *et al.* (2020)	2020	*Reviews*	*Stylistic analysis* (*words, characters, etc*) *Lexical analysis* (*verbs, nouns, etc*). *Psycholinguistic analysisc* *Emotional analysis*	*Apple Store*	Sim	Mandarim
Liu e Wu (2020)	2020	*News*	*Status-sensitive Crowd Response* (*Text Features + Users Features*)	*Twitter15 and Weibo16* (*Microblog*)	Sim	Inglês Mandarim

Tabela 3.2 – Sumário dos Algoritmos Aplicados em Trabalhos Anteriores

Estudo	Ano	Aprendizado	Método	Algoritmo	Métrica	Resultado
Li *et al.* (2014)	2014	Semi-supervisionado	Dois Passos	*PU Learning* (SVM)	F_1 *score*	67%
Fusilier *et al.* (2015)	2015	Semi-supervisionado	Dois Passos	*Naive Bayes* SVM PU *Learning* (*Naive Bayes*) PU *Learning* (SVM)	F_1 *score*	68% 53% 72% 64%
Rout *et al.* (2017)	2017	Semi-supervisionado	*Class Prior*	*Co-Training* *Expectation Maximization* *Label Propagation and Spreading* *PU Learning* (KNN, LR, RF e SGD)	F_1 *score*	78% 83% 83% 84%
Deng *et al.* (2017)	2017	Semi-supervisionado e Redes Neurais	Dois Passos	PU *Learning* (*K-means*)	Acurácia	89%
Shinde, M. P. e Channe, H. (2018)	2018	Semi-supervisionado	Dois Passos	*Co-Training* *Expectation Maximization* PU *Learning* *Ensemble* (NB + Twin-SVM)	Acurácia	73% 83% 81% 92%
Shuqin e Jing (2019)	2019	Semi-supervisionado	-	MDINPUL (*Mixing Population and Individual Nature PU Learning*)	Acurácia Precisão *Recall* F_1 *score*	87% 86% 92% 88%
He *et al.* (2020)	2020	Semi-supervisionado	Dois Passos	*SPy-Expectation Maximization* *NB-Expectation Maximization* PEBL Roc-SVM *Propose Method* (*Biased SVM + behavior density*)	Acurácia, *Recall*, F_1 *score*	P 70%, R 72%, F 71% P 69%, R 65%, F 67% P 74%, R 67%, F 68% P 74%, R 67%, F 70% P 78%, R 72%, F 75%
Liu e Wu (2020)	2020	Semi-supervisionado e Redes Neurais	-	FNED (CNN *and PU-Learning* (*Weak Classifier*))	Acurácia Precisão *Recall* F_1 *score*	*Twitter* 98%, *Weibo* 93% *Twitter* 97%, *Weibo* 92% *Twitter* 98%, *Weibo* 95% *Twitter* 98%, *Weibo* 94%

Capítulo 4

METODOLOGIA

Neste capítulo é apresentado o procedimento adotado para realizar os experimentos. A Seção 4.1 descreve o conjunto de dados utilizado com textos do idioma Português Brasileiro. A Seção 4.2 apresenta as técnicas de aprendizado supervisionado e semi-supervisionado empregados no experimento. A seguir, a Seção 4.3 mostra os métodos de PU *learning* selecionados para os experimentos. A Seção 4.4 exibe as *features* aplicadas no conjunto de dados. A Seção 4.5 expõe a configuração inicial dos parâmetros percentuais do experimento. A Seção 4.6 descreve o processo de otimização (ajuste) dos hiperparâmetros. Logo após, a Seção 4.7 apresenta as métricas selecionadas para avaliação e, por fim, é explicado o fluxo geral do experimento na Seção 4.8.

4.1 Conjunto de Dados Fake.Br

Nos experimentos apresentados nesta dissertação foi utilizado o conjunto de dados de exemplos de notícias com rótulo de classe *fake* e *true*, baseados no idioma Português Brasileiro, sendo o mesmo produzido pelo Núcleo Interinstitucional de Linguística Computacional[1] (NILC) e utilizado no trabalho de Silva *et al.* (2020). O conjunto de dados é chamado de Fake.Br[2] e foi construído de forma semiautomática, onde uma parte foi manual para garantir a confiabilidade e outra parte foi automática usando a técnica *crawling*. Os exemplos de notícias com rótulo de classe *fake* foram buscados na Internet e avaliados manualmente, enquanto os exemplos com rótulo de classe *true* foram buscados automaticamente.

A etapa de coleta e checagem observaram os seguintes pontos:

- O autor da notícia nunca é citado nas publicações;
- Os títulos são sensacionalistas e levam o usuário a clicar;

[1] http://www.nilc.icmc.usp.br/
[2] https://github.com/roneysco/Fake.br-Corpus

- Presença de erros de gramática e de concordância;
- Data de publicação da notícia;
- Não possui informações sobre o administrador do *website* responsável pelo conteúdo;
- O *layout* do *website* é muito poluído visualmente dando a impressão que é um grande *website* de notícias.

Os *websites* usados para extração de exemplos de notícias com rótulo de classe *fake* foram: Diário Brasil, The Folha do Brasil, The Journal Brasil e Top Five TV. Para verificar a veracidade dos exemplos de notícias foram utilizados os sites: Agência Lupa[3], Fato ou Fake[4], Aos Fatos[5] e Boatos.org[6]. Os *websites* usados para coleta dos exemplos de notícias com rótulo de classe *true* são G1[7], Folha de São Paulo[8] e Estadão[9].

O conjunto de dados Fake.Br possui 7.200 exemplos de notícias onde 3.600 possuem rótulo de classe *fake* e os outros 3.600 possuem rótulo de classe *true*. Cada exemplo de notícia com rótulo de classe *fake* tem seu respectivo exemplo de notícia com rótulo de classe *true*. Os exemplos de notícias são textos publicados no período de Janeiro de 2016 a Janeiro de 2018, onde o processo de coleta de análise levou 3 meses, de Dezembro de 2017 a Fevereiro de 2018. Os textos dos exemplos pertencem ao mesmo período, preservando a linguagem do momento, pois a mesma pode mudar ao longo do tempo.

O conjunto de dados possui duas versões, sendo um conjunto onde os exemplos são textos completos das notícias coletadas nos *websites* e o outro conjunto onde os exemplos são textos truncados das notícias, ou seja, são textos que possuem 200 *tokens*. Todas as palavras dos exemplos de textos de notícias foram convertidas para minúsculo e a tokenização foi baseada em espaço em branco e sinais de pontuação.

Os metadados extraídos foram a URL (*Uniform Resource Locator*), a identificação do autor ou autores, a data de publicação e o número de comentários e visualizações. Para este experimento não foram usados os metadados dos exemplos como *features* na construção dos classificadores.

[3] https://piaui.folha.uol.com.br/lupa/
[4] https://g1.globo.com/fato-ou-fake/
[5] https://www.aosfatos.org/
[6] https://www.boatos.org/
[7] https://g1.globo.com/
[8] https://www.folha.uol.com.br/
[9] https://www.estadao.com.br/

4.2 Algoritmos de Aprendizado Supervisionado e Semi-supervisionado

Os algoritmos de aprendizado supervisionado utilizados no experimento são os mesmos abordados no trabalho de Silva *et al.* (2020), como Regressão Logística, traduzido do inglês *Logistic Regression* (LR) (YU; HUANG; LIN, 2011), SVM, DT, NB, RF, *Bootstrap Aggregating* (Bagging) (BREIMAN, 1994) e *AdaBoost*. Em relação aos algoritmos semi-supervisionados foram aplicados *self-training* e co-training, por serem bastante utilizados na literatura.

Todos os algoritmos foram implementados utilizando a biblioteca de Aprendizado de Máquina para linguagem de programação Python chamada de *scikit-learn*[10] (GéRON, 2017), sem qualquer tipo de ajuste dos parâmetros predefinidos pela mesma.

4.3 Algoritmos de Aprendizado Semi-Supervisionado Baseados em PU *Learning*

Os algoritmos de aprendizado semi-supervisionado baseados em PU *learning* utilizados são:

- *Classic Elkanoto*: apresentado no trabalho de Elkan e Noto (2008) que pertence à categoria de métodos de PU *learning* baseados na incorporação da *class prior*.
- *Weighted Elkanoto*: apresentado no trabalho de Elkan e Noto (2008) que pertence à categoria de métodos de PU *learning* baseados na incorporação da *class prior*, especificando pesos individuais para os exemplos.
- *Bagging* baseado PU *learning*: apresentado no trabalho de Mordelet e Vert (2010) que pertence à categoria de métodos de PU *learning* baseados no *biased learning*.

Todos os algoritmos foram implementados utilizando a biblioteca chamada de *pulearn*[11]. Esses algoritmos foram escolhidos devido a construção de seu código estar bem ajustada a biblioteca *pulearn* e sua versão se manter sempre atualizada no site do projeto, cujas revisões se encontram em *Pypi.org*[12].

A técnica de dois passos não foi utilizada pois durante o levantamento do estado da arte não foi encontrada uma biblioteca de classificação com esse método e os autores não disponibilizaram o código.

[10] https://scikit-learn.org/
[11] https://pulearn.github.io/pulearn/doc/pulearn/
[12] https://pypi.org/project/pulearn/#history

Um ponto importante a destacar em relação aos algoritmos selecionados é que a finalidade do trabalho não é encontrar o melhor algoritmo de aprendizado semi-supervisionado baseado em PU *learning* para detecção de *fake news* a partir de poucos exemplos rotulados. O propósito é comparar o desempenho destes algoritmos selecionados com os algoritmos de aprendizado supervisionado descritos na Seção 4.2.

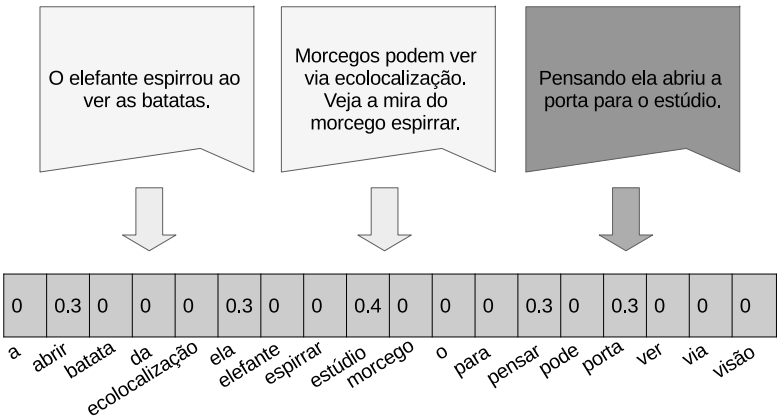
4.4 *Features*

Em relação à extração de *features*, a técnica empregada para mapear cada palavra (termo) do texto em um vetor numérico, foi o TF-IDF. O TF-IDF é uma forma de representação vetorial que tem relação com os vetores *one-hot encode* (BILBRO; OJEDA; BENGFORT, 2018).

O TF-IDF representa as palavras pelo produto da frequência do termo, traduzido inglês *term frequency* (TF) pela frequência inversa do documento (texto), traduzido do inglês *inverse document frequency* (IDF).

As palavras com muita ocorrência recebem um peso menor. Esta abordagem de codificação realça as palavras que são mais relevantes para um exemplo específico do conjunto de dados, conforme ilustra na Figura 4.1, onde a palavra (*token*) “estúdio”tem maior relevância para o texto sendo que ela aparece apenas uma vez (BILBRO; OJEDA; BENGFORT, 2018).

Figura 4.1 – Codificação com TD-IDF



Fonte:(BILBRO; OJEDA; BENGFORT, 2018)

Seja t o número de vezes que um termo aparece no documento e T o número de termos no documento, o resultado do TF é encontrado pela equação 4.1 (SRINIVASA-DESIKAN,

2018):

$$TF = \frac{t}{T}, \tag{4.1}$$

Seja N o número de documentos e n o número de documentos que um termo aparece, o resultado do IDF é encontrado pela equação 4.2 (SRINIVASA-DESIKAN, 2018):

$$IDF = \log \frac{N}{n}, \tag{4.2}$$

Segundo Srinivasa-Desikan (2018), esse dois fatores juntos armazenam mais informações em um vetor de características, no lugar de contar as palavras, igual à técnica de *bag of words*.

Um ponto importante a ressaltar é que foram extraídas apenas 3.000 características de cada exemplo, devido às limitações do *hardware* utilizado para execução deste trabalho, onde o fator limitante foi a quantidade de memória RAM (*Randon Access Memory*). Pois, quanto maior o número de características, maior o consumo de RAM (KOZHEVNIKOV; PANKRATOVA, 2020).

4.5 Métricas

Para análise dos resultados obtidos foi calculado em cada classificador (modelo) a métrica de F_1 *score* (equação 2.10).

Os valores obtidos serão comparados entre todos os classificadores gerados pelos algoritmos de Aprendizado de Máquina do experimento, de modo a avaliar o grau de capacidade de generalização de cada um, em cada execução conforme o *setup* da configuração do experimento.

4.6 Ajuste dos classificadores

O ajuste envolve encontrar os melhores valores para os hiperparâmetros, de modo elevar a taxa de acerto dos algoritmos selecionados para o experimento. Uma forma de encontrar é alterar manualmente e testar todas as combinações possíveis, mas seria um trabalho rigoroso identificar as melhores combinações.

De acordo com Géron (2017), a técnica de busca aleatória é a mais apropriada para otimização de hiperparâmetros, quando o espaço de busca dos valores é muito amplo. A biblioteca *scikit-learn* implementa esta técnica com *RandomizedSearchCV*[13].

[13] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

Com *RandomizedSearchCV*, se uma distribuição (lista) de valores é especificada, ele gera amostras aleatoriamente sem repetição dos valores daquela distribuição. Um exemplo geral seria 10 valores (pontos flutuantes) de uma distribuição uniforme de 0 até 4. Mas se a lista de distribuição é pequena, as amostras podem ter repetição quando combinadas com outros hiperparâmetros (ALBON, 2018). O número de amostras de combinações de hiperparâmetros é especificado pelo parâmetro *n_iter* no *RandomizedSearchCV*.

4.7 *Setup* da Configuração

Com o objetivo de avaliar a eficiência dos algoritmos semi-supervisionados baseados em PU *learning* na detecção de *fake news* no idioma Português Brasileiro a partir de poucos exemplos rotulados, como configuração inicial, o conjunto de dados Fake.Br foi alterado com a descrição do rótulo de classe *true* para *not_fake*.

A partir da mudança aplicada, os exemplos com rótulo de classe *fake* receberam o valor 1 para representar os exemplos positivos, e os exemplos com rótulo de classe *not_fake* receberam o valor 0 para representar os exemplos sem rótulo de classe.

O conjunto de dados foi dividido inicialmente em 80% para treino e 20% para teste. Após o treinamento e teste de todos os algoritmos com a primeira configuração do conjunto de dados, o mesmo foi dividido novamente em 60% para treino, 20% para validação (otimização dos hiperparâmetros) e 20% para teste, para aplicar o ajuste de hiperparâmetros. A porcentagem de exemplos positivos e exemplos sem rótulo de classe foram modificados durante todos os experimentos. O percentual inicial de exemplos positivos e exemplos sem rótulo de classe é balanceada (aproximadamente 50% para cada conjunto de exemplos).

Todos os experimentos foram executados 8 vezes. Para cada execução, uma porcentagem de exemplos positivos deve ser selecionada para alterar o rótulo de classe de 1 (*fake*) para 0 (*not_fake*). A porcentagem selecionada de exemplos com rótulo de classe modificada é adicionada ao conjunto de exemplos com rótulo de classe "*not_fake*"(exemplos sem rótulo).

O parâmetro de porcentagem varia entre os valores 0, 5, 10, 20, 40, 60, 80 e 90% do número de exemplos positivos selecionados em cada execução.

Para evitar problemas de baixo desempenho (como *overfitting* e *underfitting*) , ou seja, baixa generalização dos algoritmos de Aprendizado de Máquina, o procedimento adotado é:

1. Em cada execução, treinar o algoritmo por 10 vezes e extrair a média da métrica de F_1 *score* e de desvio padrão.

2. Manter o valor padrão da semente em cada execução, no qual é a hora atual do sistema.

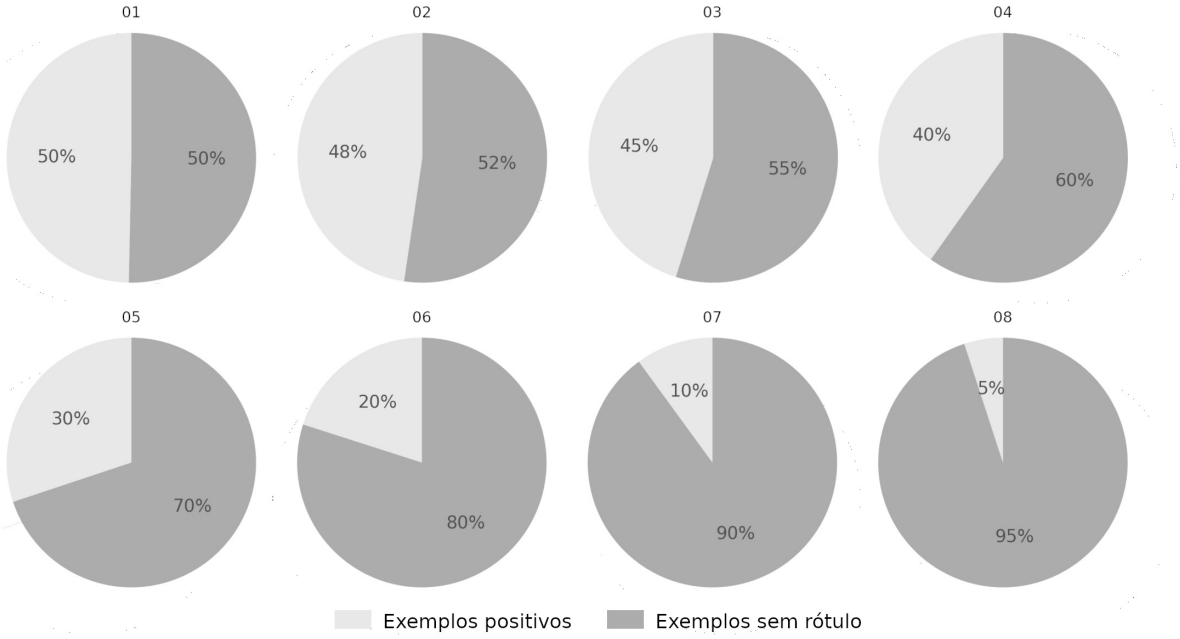
3. Em cada execução manter os mesmos exemplos positivos selecionados da execução anterior e acrescentar novos exemplos positivos para atender o valor do parâmetro de porcentagem. Exemplo: Após finalizar a segunda execução, manter os dados que foram selecionados (5% de exemplos positivos) e para a terceira execução selecionar mais 5%, de modo que sejam somados, conforme o valor do parâmetro de porcentagem que é 10%.

A Figura 4.2 ilustra a distribuição percentual de exemplos positivos (fatia na cor clara) e exemplos sem rótulo (fatia na cor escura) em cada execução, onde:

- **Execução 01**: ilustra que os exemplos positivos e exemplos sem rótulos possuem a mesma proporção.
- **Execução 02**: ilustra que 5% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.
- **Execução 03**: ilustra que 10% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.
- **Execução 04**: ilustra que 20% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.
- **Execução 05**: ilustra que 40% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.
- **Execução 06**: ilustra que 60% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.
- **Execução 07**: ilustra que 80% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.
- **Execução 08**: ilustra que 90% dos exemplos positivos são modificados para exemplos sem rótulo e adicionados a porção de exemplos sem rótulo.

Para cada execução foram treinados todos os algoritmos descritos na Seção 4.2 e Seção 4.3 e por fim, aplicado o teste e a coleta dos resultados de cada classificador gerado.

Figura 4.2 – Distribuição percentual dos exemplos para treino em cada execução



Fonte: O autor.

4.8 Fluxo Geral do Experimento

A arquitetura do experimento descrita em uma visão de alto nível é ilustrada na Figura 4.3, de modo a favorecer o entendimento da metodologia aplicada. No processo de treinamento, os exemplos positivos (*fake*) e os exemplos sem rótulo (*not_fake*) são codificados com a técnica de TF-IDF (item 1) e são submetidos como entrada para aplicação dos parâmetros percentuais em cada execução (item 2). A etapa de preparação dos dados (item 2), envolve selecionar aleatoriamente os exemplos positivos que serão modificado o rótulo de classe e adicionados ao conjunto de exemplos sem rótulo de classe, conforme os parâmetros percentuais (0, 5, 10, 20, 40, 60, 80 e 90%).

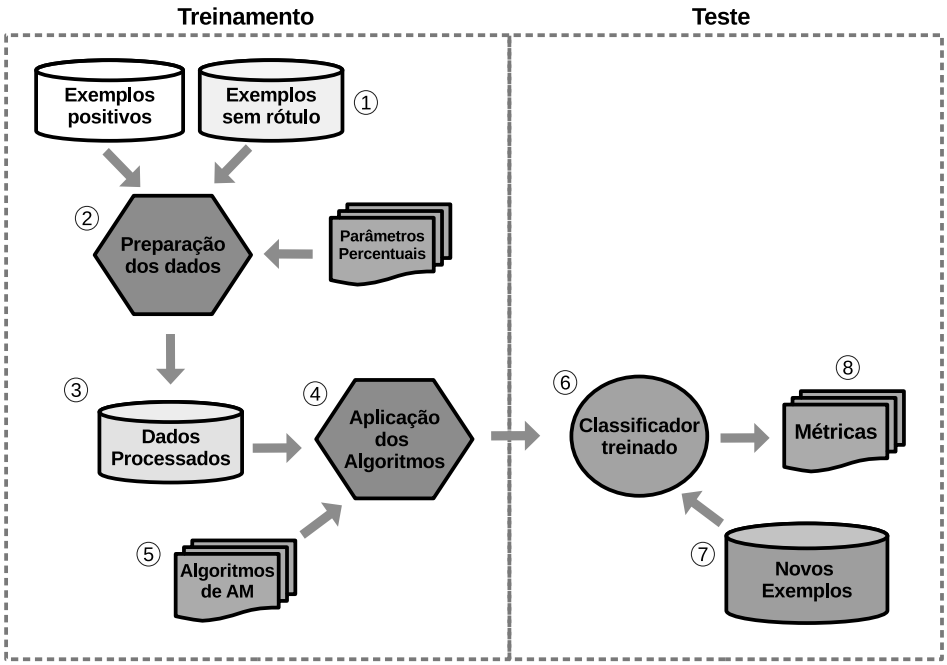
Os dados processados (item 3) conforme o parâmetro percentual aplicado é utilizado como entrada para o processo de Aprendizado de Máquina (item 4), com todos os algoritmos escolhidos para experimento (item 5). Na etapa de teste, todos os classificadores (item 6) construídos a partir do treinamento dos algoritmos de Aprendizado de Máquina recebem novos exemplos sem rótulo como entrada (item 7) para determinar qual classe pertencem.

Os novos exemplos após serem classificados, ou seja, receberem um rótulo de classe, são avaliados pelas métricas (item 8) de F_1 *score* e o desvio padrão. Finalizado a avaliação da capacidade de generalização do algoritmo, é escolhido o próximo parâmetro percentual na lista de parâmetros. O processo é repetido (treinamento e teste) até utilizar todos os parâmetros percentuais definidos e encerrar o fluxo do experimento.

Todo esse fluxo ocorre em duas fases que são:

- Quando não é aplicado os ajustes (otimização de hiperparâmetros) nos classificadores;
- E quando é aplicado os ajustes de hiperparâmetros.

Figura 4.3 – Fluxo Geral do Experimento



Fonte: O autor.

Capítulo 5

RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados dos experimentos realizados com todos os algoritmos descritos no capítulo anterior.

Os resultados foram obtidos inicialmente utilizando o *Google Colab*[1] para o treinamento dos algoritmos supervisionados mencionados na Seção 4.2. Mas devido a limitação de memória RAM da plataforma onde se é liberado de forma gratuita apenas 12 GB, ao se utilizar os algoritmos semi-supervisionados baseados em PU *learning*, foi percebido que a quantidade disponibilizada foi insuficiente, sendo necessário optar por outro meio para executar o treinamento dos algoritmos.

Assim sendo, para eliminar o problema de limitação de memória RAM, todos os algoritmos propostos para uso no experimento foram executados em um computador cuja configuração de *hardware* é:

- Processador *Intel Core i5-9300H* 9ª geração (2.4 GHz até 4.1 GHz) *cache* de 8 MB;
- Placa de vídeo dedicada *NVIDIA GeForce GTX* 1650 com 4 GB de GDDR5;
- Memória RAM de 16 GB DDR4 2666MHz.

Além da mudança de *hardware* para executar o experimento, foram extraídas somente 3.000 características dos exemplos pertencentes ao conjunto de dados Fake.Br. Cujo propósito da quantidade extraída é de não consumir toda memória RAM (KOZHEVNIKOV; PANKRATOVA, 2020). Iniciou-se usando todas as características, avaliando o consumo de RAM e reduzindo gradualmente até chegar em 3.000.

[1] https://colab.research.google.com/

5.1 Resultados da Metodologia Proposta

Para melhor visualização de cada legenda dos resultados exibidos nas tabelas a seguir, os nomes de alguns algoritmos serão sintetizados conforme as abreviaturas abaixo:

- **BG**: *Bagging*
- **AB**: *AdaBoost*
- **ST**: *Self-Training*
- **ST + AB**: *Self-Training + AB*
- **ST + SVM**: *Self-Training + SVM*
- **CT**: *Co-Training*
- **CT + AB**: *Co-Training + AB*
- **CT + SVM**: *Co-Training + SVM*
- **PUCE + RF**: PU *Learning based Classic Elkanoto* combinado com *Random Forest*
- **PUCE + SVM**: PU *Learning based Classic Elkanoto* combinado com *Support Vector Machines*
- **PUCE + AB**: PU *Learning based Classic Elkanoto* combinado com *AdaBoost*
- **PUCE + LR**: PU *Learning based Classic Elkanoto* combinado com *Logistic Regresssion*
- **PUWE + RF**: PU *Learning based Weighted Elkanoto* combinado com *Random Forest*
- **PUWE + SVM**: PU *Learning based Weighted Elkanoto* combinado com *Support Vector Machines*
- **PUWE + AB**: PU *Learning based Weighted Elkanoto* combinado com *AdaBoost*
- **PUWE + LR**: PU *Learning based Weighted Elkanoto* combinado com *Logistic Regression*
- **BPU + RF**: *Bagging based* PU *learning* combinado com *Random Forest*
- **BPU + SVM**: *Bagging based* PU *learning* combinado com *Support Vector Machines*
- **BPU + AB**: *Bagging based* PU *learning* combinado com *AdaBoost*
- **BPU + LR**: *Bagging based* PU *learning* combinado com *Logistic Regresssion*

A Tabela 5.1 mostra o *score* dos algoritmos supervisionados, abordados no trabalho de Silva *et al.* (2020), aplicados nas duas versões do conjunto de dados Fake.Br. Sendo que uma versão possui exemplos de textos completos das notícias e outra constitui-se de exemplos de textos truncados das notícias, ou seja, somente 200 *tokens* por exemplo. Os valores representam a métrica F_1 *Score* e o desvio padrão de cada algoritmo. Considerando os três melhores algoritmos treinados, ao aplicar os mesmos na versão onde os exemplos são textos completos das notícias, destaca-se o RF, o SVM e o AB que tiveram desempenho superior. Enquanto que na versão do conjunto de dados que possui exemplos de textos truncados das notícias, foram os algoritmos RF, SVM e LR que tiveram melhor resultado.

Tabela 5.1 – *Scores* obtidos com cada algoritmo no conjunto de dados com textos completos de notícias e no conjunto de dados com textos truncados de notícias

Algoritmo	**Textos Completos**	**Textos Truncados**
RF	**0.956** (± 0.002)	**0.872** (± 0.004)
NB	0.8872 (± 0.000)	0.849 (± 0.000)
SVM	**0.960** (± 0.000)	**0.914** (± 0.000)
LR	0.950 (± 0.000)	**0.907** (± 0.000)
DT	0.910 (± 0.003)	0.739 (± 0.008)
BG	0.956 (± 0.003)	0.851 (± 0.006)
AB	**0.957** (± 0.000)	0.839 (± 0.000)

Uma vez encontrados os três melhores resultados com algoritmos supervisionados em cada versão do conjunto de dados. Estes algoritmos serão combinados com os algoritmos de aprendizado semi-supervisionado baseados em PU *learning*. Como requisito eles devem possuir internamente a implementação dos métodos *fit()*[2] e *predict_proba()*[2] . O método *fit()* é utilizado para treinar o classificador e o método *predict_proba()* prediz a probabilidade de classe de cada exemplo.

A Tabela 5.2 mostra os resultados dos algoritmos combinados com o algoritmo semi-supervisionado baseado em PU *Learning Classic Elkanoto* representado pelas letras PUCE. A coluna de nome "Execução" representa a ordem das execuções para cada treinamento, de modo que, em cada execução os exemplos positivos devem ser alterados para exemplos sem rótulo de classe. Essa mudança durante a execução é conforme o parâmetro percentual descrito no capítulo anterior. A combinação de PUCE com RF permitiu detectar em média 92% das notícias falsas, quando aplicados no conjunto de dados que possui exemplos de textos completos de notícias.

O resultado da combinação de PUCE com RF mostra que o PUCE consegue identificar os exemplos positivos modificados, aplicar o rótulo de classe correto e permitir que o RF mantenha a mesma taxa de acerto quando o mesmo foi aplicado no conjunto de dados balanceado. A Tabela 5.2 mostra que o RF teve média de 92% e a Tabela 5.1 teve média de 95%. O fato do PUCE combinado com RF ter um bom resultado comparado aos demais na

[2] https://pulearn.github.io/pulearn/doc/pulearn/elkanoto.html

Tabela 5.2, mostra que ele se mantém com um bom resultado na Tabela 5.1. O resultado dos demais pode ter relação com o parâmetro *hold_out_ratio* do PUCE que reserva uma parte dos exemplos de treinamento para estimar a probabilidade de um exemplo ser positivo.

Tabela 5.2 – *Scores* obtidos combinando PUCE + RF, PUCE + SVM e PUCE + AB no conjunto de dados com exemplos de textos completos de notícias

Execução	**PUCE + RF**	**PUCE + SVM**	**PUCE + AB**
1	0.951 (± 0.002)	0.950 (± 0.003)	0.661 (± 0.001)
2	0.953 (± 0.002)	0.928 (± 0.003)	0.661 (± 0.001)
3	0.952 (± 0.002)	0.910 (± 0.002)	0.661 (± 0.001)
4	0.952 (± 0.003)	0.875 (± 0.009)	0.661 (± 0.001)
5	0.953 (± 0.003)	0.822 (± 0.005)	0.661 (± 0.002)
6	0.944 (± 0.003)	0.782 (± 0.005)	0.661 (± 0.003)
7	0.899 (± 0.005)	0.726 (± 0.011)	0.668 (± 0.005)
8	0.800 (± 0.012)	0.706 (± 0.006)	0.681 (± 0.034)
Média	**0.926** (± 0.004)	0.837 (± 0.005)	0.664 (± 0.002)

A Tabela 5.3 exibe os resultados dos algoritmos combinados com o algoritmo semi-supervisionado baseado em PU *Learning Weighted Elkanoto* representado pelas letras PUWE. O melhor resultado se conseguiu com RF, que foi capaz de detectar em média 70% das notícias falsas quanto aplicado no conjunto de dados que possui exemplos de textos completos de notícias. Somente a combinação PUWE e SVM que teve um resultado abaixo da média de 60% na 8ª execução, em que 90% dos exemplos positivos removeu-se o rótulo de classe e os acrescentou a porção de exemplos sem rótulo de classe.

Uma hipótese, quanto aos resultados baixos dos algoritmos combinados com PUWE em relação PUCE pode ser a cardinalidade atribuída ao conjunto de treinamento, que é um parâmetro do PUWE.

Tabela 5.3 – *Scores* obtidos combinando PUWE + RF, PUWE + SVM e PUWE + AB no conjunto de dados com textos completos de notícias

Execução	**PUWE + RF**	**PUWE + SVM**	**PUWE + AB**
1	0.665 (± 0.001)	0.661 (± 0.000)	0.661 (± 0.000)
2	0.666 (± 0.001)	0.661 (± 0.000)	0.661 (± 0.000)
3	0.666 (± 0.001)	0.661 (± 0.000)	0.661 (± 0.000)
4	0.667 (± 0.001)	0.661 (± 0.000)	0.661 (± 0.000)
5	0.688 (± 0.002)	0.661 (± 0.000)	0.661 (± 0.000)
6	0.701 (± 0.003)	0.661 (± 0.000)	0.661 (± 0.000)
7	0.834 (± 0.005)	0.676 (± 0.006)	0.661 (± 0.000)
8	0.685 (± 0.034)	0.426 (± 0.054)	0.661 (± 0.000)
Média	**0.697** (± 0.006)	0.634 (± 0.008)	0.661 (± 0.000)

A Tabela 5.4 apresenta os resultados dos algoritmos combinados com o algoritmo semi-supervisionado *Bagging* baseado em PU *Learning* representado pelas letras BPU. Ao utilizar o *Bagging* baseado em PU *Learning* combinado com AB (AdaBoost) no conjunto de

dados que possui exemplos de textos completos de notícias, são identificadas em média de 63% das notícias falsas. No geral todas as combinações dos algoritmos tiveram um resultado mais baixo a partir da 6ª execução. Na 6ª execução, cerca de 60% dos exemplos positivos são modificados para exemplos sem rótulo de classe e adicionados a porção de exemplos sem rótulo.

O parâmetro *oob_score* (valor padrão igual *True*) do BPU pode ser considerado como um ponto relevante perante aos resultados encontrados, pois o algoritmo *Bagging* possui o mesmo parâmetro (valor padrão igual a *False*) e teve bom resultado ilustrado na Tabela 5.1. O BPU é uma adaptação do *Bagging* para PU *learning*.

Tabela 5.4 – *Scores* obtidos combinando BPU + RF, BPU + SVM e BPU + AB no conjunto de dados com textos completos de notícias

Execução	**BPU + RF**	**BPU + SVM**	**BPU + AB**
1	0.952 (± 0.001)	0.954 (± 0.001)	0.967 (± 0.002)
2	0.953 (± 0.001)	0.944 (± 0.002)	0.967 (± 0.002)
3	0.955 (± 0.001)	0.929 (± 0.001)	0.960 (± 0.002)
4	0.957 (± 0.001)	0.902 (± 0.003)	0.938 (± 0.004)
5	0.926 (± 0.003)	0.746 (± 0.004)	0.834 (± 0.007)
6	0.254 (± 0.012)	0.286 (± 0.012)	0.390 (± 0.018)
7	0.000 (± 0.000)	0.008 (± 0.001)	0.026 (± 0.004)
8	0.000 (± 0.000)	0.000 (± 0.000)	0.015 (± 0.003)
Média	0.625 (± 0.002)	0.596 (± 0.003)	**0.637** (± 0.005)

A Tabela 5.5 destaca que a combinação entre os algoritmos PUCE e LR obteve melhor desempenho na identificação de notícias falsas. Ao serem aplicados no conjunto de dados que possui exemplos de textos truncados de notícias, atingiu em média 85% de acerto. Em todas as execuções as combinações dos algoritmos conseguiram identificar mais de 60% das notícias falsas.

Tabela 5.5 – *Scores* obtidos combinando PUCE + RF, PUCE + SVM e PUCE + LR no conjunto de dados com textos truncados de notícias

Execução	**PUCE + RF**	**PUCE + SVM**	**PUCE + LR**
1	0.813 (± 0.000)	0.888 (± 0.000)	0.885 (± 0.000)
2	0.815 (± 0.000)	0.867 (± 0.000)	0.884 (± 0.000)
3	0.816 (± 0.000)	0.849 (± 0.000)	0.886 (± 0.000)
4	0.822 (± 0.000)	0.821 (± 0.000)	0.885 (± 0.000)
5	0.819 (± 0.000)	0.786 (± 0.000)	0.878 (± 0.000)
6	0.786 (± 0.000)	0.742 (± 0.000)	0.861 (± 0.000)
7	0.743 (± 0.000)	0.701 (± 0.000)	0.817 (± 0.000)
8	0.621 (± 0.000)	0.685 (± 0.000)	0.767 (± 0.000)
Média	0.780 (± 0.000)	0.792 (± 0.000)	**0.858** (± 0.000)

A Tabela 5.6 mostra que em média 64% das notícias falsas foram detectadas através da combinação de PUWE e LR no conjunto de dados que possui exemplos de textos trunca-

dos de notícias. Na 8ª execução todas as combinações de algoritmos tiveram um resultado abaixo de 60%.

Tabela 5.6 – *Scores* obtidos combinando PUWE + RF, PUWE + SVM e PUWE + LR no conjunto de dados com textos truncados de notícias

Execução	**PUWE + RF**	**PUWE + SVM**	**PUWE + LR**
1	0.661 (± 0.000)	0.661 (± 0.000)	0.661 (± 0.000)
2	0.661 (± 0.000)	0.661 (± 0.000)	0.661 (± 0.000)
3	0.661 (± 0.000)	0.662 (± 0.000)	0.661 (± 0.000)
4	0.661 (± 0.000)	0.663 (± 0.000)	0.661 (± 0.000)
5	0.662 (± 0.000)	0.661 (± 0.000)	0.661 (± 0.000)
6	0.663 (± 0.000)	0.661 (± 0.000)	0.662 (± 0.000)
7	0.723 (± 0.000)	0.674 (± 0.000)	0.700 (± 0.000)
8	0.400 (± 0.000)	0.366 (± 0.000)	0.453 (± 0.000)
Média	0.637 (± 0.000)	0.626 (± 0.000)	**0.640** (± 0.000)

A Tabela 5.7 exibe que a média de detecção de notícias falsas entre todas as combinações dos algoritmos ficou abaixo de 60% e que todos tiveram baixo resultado a partir da 6ª execução. A combinação BPU e SVM teve média de 54% de acerto, ficando acima das demais.

Tabela 5.7 – *Scores* obtidos combinando BPU + RF, BPU + SVM e BPU + LR no conjunto de dados com textos truncados de notícias

Execução	**BPU + RF**	**BPU + SVM**	**BPU + LR**
1	0.852 (± 0.000)	0.900 (± 0.000)	0.906 (± 0.000)
2	0.861 (± 0.000)	0.891 (± 0.000)	0.905 (± 0.000)
3	0.868 (± 0.000)	0.877 (± 0.000)	0.898 (± 0.000)
4	0.872 (± 0.000)	0.844 (± 0.000)	0.857 (± 0.000)
5	0.728 (± 0.000)	0.643 (± 0.000)	0.588 (± 0.000)
6	0.077 (± 0.000)	0.208 (± 0.000)	0.109 (± 0.000)
7	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)
8	0.003 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)
Média	0.533 (± 0.000)	**0.545** (± 0.000)	0.533 (± 0.000)

A Tabela 5.8 e a Tabela 5.9 mostram a combinação dos algoritmos semi-supervisionados com os algoritmos supervisionados. Essa combinação ao ser aplicada no conjunto de dados que possui exemplos com textos completos de notícias, o ST com AB e CT com AB tiveram melhores resultados. No entanto, ao se usar as combinações dos algoritmos mencionados no conjunto de dados que possui exemplos com textos truncados de notícias, os melhores resultados são ST com SVM e CT com SVM.

Após finalizar as execuções com todas as combinações dos algoritmos e identificar qual das combinações possuem o melhor resultado, foi executado um novo treinamento. Esse treinamento foi executado com todos os algoritmos supervisionados, semi-supervisionados e semi-supervisionados baseados em PU *learning*.

Tabela 5.8 – Média dos *scores* obtidos combinando ST + RF, ST + SVM, ST + AB, CT + RF, CT + SVM e CT + AB no conjunto de dados com textos longos de notícias.

	RF	**SVM**	**AB**
ST	0.571 (± 0.003)	0.652 (± 0.000)	**0.663 (± 0.000)**
CT	0.571 (± 0.002)	0.570 (± 0.001)	**0.627 (± 0.000)**

Tabela 5.9 – Média dos *scores* obtidos combinando ST + RF, ST + SVM, ST + LR, CT + RF, CT + SVM e CT + LR no conjunto de dados com textos truncados de notícias.

	RF	**SVM**	**LR**
ST	0.461 (± 0.006)	**0.607 (± 0.000)**	0.526 (± 0.000)
CT	0.452 (± 0.004)	**0.511 (± 0.001)**	0.486 (± 0.000)

Os resultados apresentados que antecedem as Tabelas 5.10 e 5.11, são de todos os algoritmos sem qualquer tipo de ajuste de hiperparâmetro.

Ambas Tabelas 5.10 e 5.11 representam o resultado de todos os algoritmos propostos neste trabalho, sendo que apenas parte deles foi possível aplicar o ajuste de hiperparâmetros. Somente 8 algoritmos de Aprendizado de Máquina selecionados para o experimento foram aplicados otimização dos hiperparâmetros que são ST, RF, NB, LR, DT, BG e AB. Para os algoritmos PUCE, PUWE, BPU e CT não foi possível ajustar, pois eles não implementam o método *get_parameters* que retorna os parâmetros do classificador. O *RandomizedSearchCV* utiliza esse método para iniciar o processo de ajuste dos hiperparâmetros.

A Tabela 5.10 destaca que a combinação dos algoritmos PUCE com RF identificou em média 92% das notícias falsas em todas as execuções, quando treinado no conjunto de dados que possui exemplos com textos completos de notícias. A Tabela 5.11 exibe que em média 82% das notícias falsas foram detectadas usando a combinação PUCE com LR no conjunto de dados que possui exemplos com textos truncados de notícias.

As Figuras 5.1 e 5.2 exibem a mediana, as distâncias entre os quartis e a variação (valor máximo e mínimo) com níveis diferentes da porcentagem de média dos *scores* obtidos por cada algoritmo de Aprendizado de Máquina do experimento. O PUCE combinado com RF quando aplicado no conjunto de dados com textos completos de notícias não tem uma dispersão tão grande quanto os outros algoritmos ao se observar a distância entre a mediana e o terceiro quartil.

A Figura 5.3 compara os resultados da métrica de F_1 *Score* entre algoritmos baseados em PU *learning* e os algoritmos supervisionados. O eixo das abscissas (x) representa o percentual dos exemplos positivos cujo rótulo de classe é removido durante as execuções. Esses exemplos positivos pertencem ao conjunto de dados que possui exemplos de textos completos de notícias.

A Figura 5.4 se diferencia da Figura 5.3, pois compara os resultados de F_1 *Score* entre algoritmos baseados em PU *learning* e os algoritmos semi-supervisionados. A Figura 5.5 e a

Figura 5.6 em relação às Figuras 5.3 e 5.4, se distinguem, pois, apresentam resultados a partir do uso do conjunto de dados que possui exemplos com textos truncados de notícias.

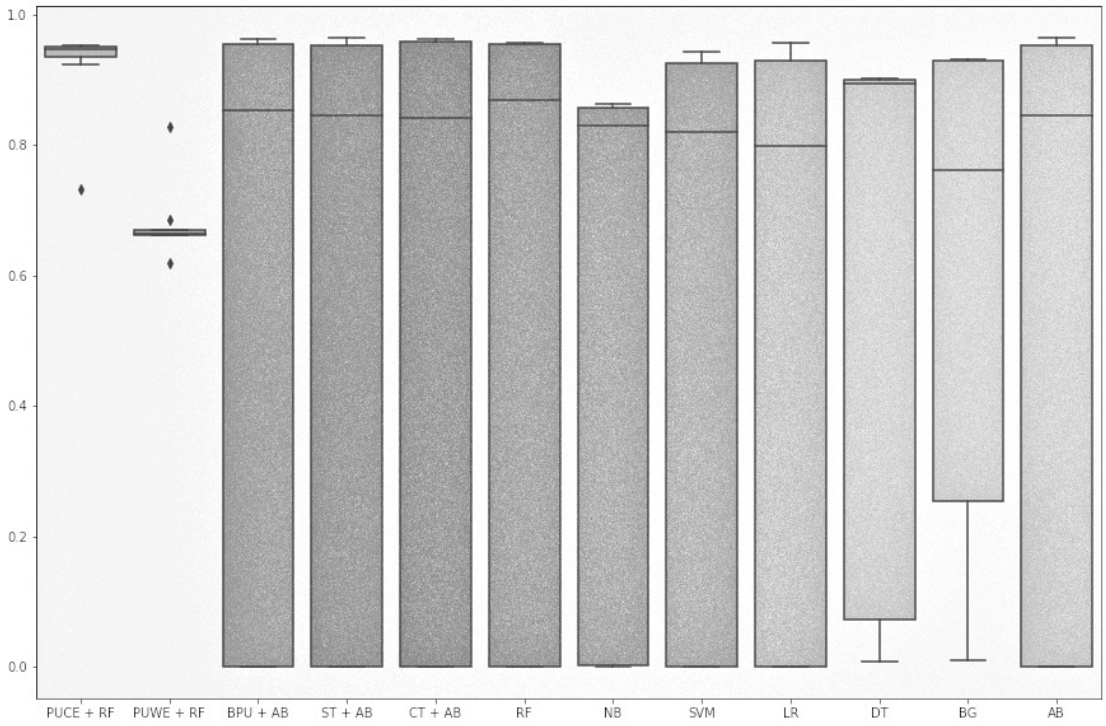
Tabela 5.10 – Sumário dos *scores* para todos os algoritmos aplicados no conjunto de dados com exemplos de textos completos de notícias

Execução	PUCE + RF	PUWE + RF	BPU + AB	ST + AB	CT + AB	RF	NB	SVM	LR	DT	BG	AB
1	0.949 (± 0.002)	0.662 (± 0.000)	0.962 (± 0.001)	0.964 (± 0.000)	0.961 (± 0.000)	0.956 (± 0.002)	0.862 (± 0.000)	0.922 (± 0.000)	0.956 (± 0.000)	0.898 (± 0.002)	0.927 (± 0.000)	0.964 (± 0.000)
2	0.945 (± 0.001)	0.661 (± 0.000)	0.956 (± 0.001)	0.956 (± 0.000)	0.961 (± 0.000)	0.953 (± 0.001)	0.861 (± 0.000)	0.930 (± 0.000)	0.927 (± 0.000)	0.899 (± 0.001)	0.930 (± 0.004)	0.956 (± 0.000)
3	0.948 (± 0.001)	0.663 (± 0.000)	0.954 (± 0.001)	0.951 (± 0.000)	0.957 (± 0.000)	0.957 (± 0.002)	0.855 (± 0.000)	0.943 (± 0.000)	0.935 (± 0.000)	0.901 (± 0.002)	0.931 (± 0.004)	0.951 (± 0.000)
4	0.952 (± 0.002)	0.665 (± 0.001)	0.936 (± 0.001)	0.935 (± 0.000)	0.931 (± 0.000)	0.948 (± 0.001)	0.852 (± 0.000)	0.917 (± 0.000)	0.903 (± 0.000)	0.890 (± 0.001)	0.899 (± 0.009)	0.935 (± 0.000)
5	0.952 (± 0.002)	0.684 (± 0.001)	0.771 (± 0.005)	0.753 (± 0.000)	0.750 (± 0.000)	0.786 (± 0.009)	0.808 (± 0.000)	0.723 (± 0.000)	0.693 (± 0.000)	0.899 (± 0.001)	0.624 (± 0.000)	0.753 (± 0.000)
6	0.923 (± 0.005)	0.661 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.002 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.090 (± 0.000)	0.319 (± 0.012)	0.000 (± 0.000)
7	0.939 (± 0.002)	0.827 (± 0.006)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.019 (± 0.007)	0.053 (± 0.014)	0.000 (± 0.000)
8	0.732 (± 0.033)	0.618 (± 0.026)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.001 (± 0.001)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.007 (± 0.001)	0.010 (± 0.007)	0.000 (± 0.000)
Média	**0.918 (± 0.006)**	0.680 (± 0.004)	0.572 (± 0.001)	0.570 (± 0.000)	0.570 (± 0.000)	0.575 (± 0.002)	0.530 (± 0.000)	0.554 (± 0.000)	0.552 (± 0.000)	0.576 (± 0.002)	0.587 (± 0.006)	0.570 (± 0.000)

Tabela 5.11 – Sumário dos *scores* para todos os algoritmos aplicados no conjunto de dados com exemplos de textos truncados de notícias

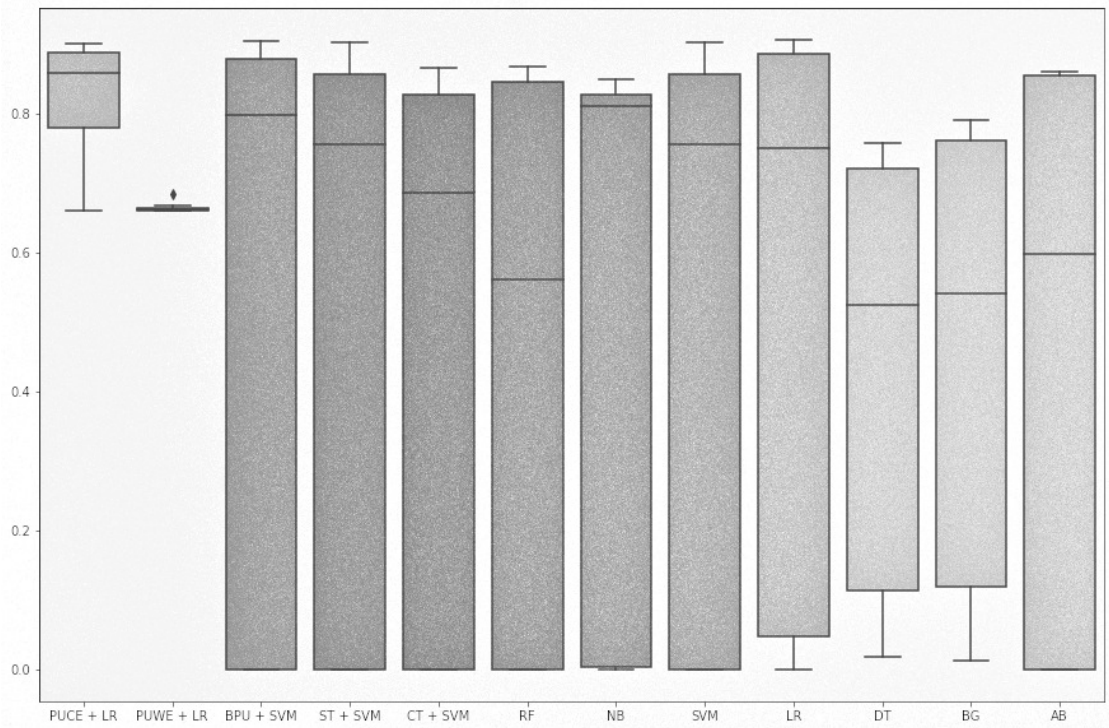
Execução	PUCE + LR	PUWE + LR	BPU + SVM	ST + SVM	CT + SVM	RF	NB	SVM	LR	DT	BG	AB
1	0.900 (± 0.004)	0.662 (± 0.000)	0.903 (± 0.000)	0.903 (± 0.000)	0.865 (± 0.005)	0.867 (± 0.005)	0.848 (± 0.000)	0.903 (± 0.000)	0.906 (± 0.000)	0.757 (± 0.003)	0.780 (± 0.000)	0.861 (± 0.000)
2	0.888 (± 0.002)	0.661 (± 0.000)	0.878 (± 0.000)	0.854 (± 0.000)	0.823 (± 0.005)	0.865 (± 0.004)	0.827 (± 0.000)	0.854 (± 0.000)	0.895 (± 0.000)	0.715 (± 0.002)	0.790 (± 0.000)	0.858 (± 0.000)
3	0.889 (± 0.003)	0.663 (± 0.000)	0.878 (± 0.000)	0.862 (± 0.000)	0.841 (± 0.004)	0.838 (± 0.005)	0.828 (± 0.000)	0.862 (± 0.000)	0.881 (± 0.000)	0.735 (± 0.001)	0.678 (± 0.000)	0.854 (± 0.000)
4	0.875 (± 0.003)	0.661 (± 0.000)	0.859 (± 0.000)	0.848 (± 0.000)	0.763 (± 0.004)	0.775 (± 0.01)	0.824 (± 0.000)	0.848 (± 0.000)	0.846 (± 0.000)	0.601 (± 0.003)	0.754 (± 0.000)	0.795 (± 0.000)
5	0.816 (± 0.007)	0.684 (± 0.001)	0.736 (± 0.000)	0.662 (± 0.000)	0.608 (± 0.011)	0.348 (± 0.017)	0.795 (± 0.000)	0.662 (± 0.000)	0.652 (± 0.000)	0.446 (± 0.000)	0.403 (± 0.000)	0.399 (± 0.000)
6	0.841 (± 0.004)	0.661 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.005 (± 0.000)	0.000 (± 0.000)	0.062 (± 0.000)	0.119 (± 0.008)	0.146 (± 0.021)	0.000 (± 0.000)
7	0.661 (± 0.000)	0.661 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.094 (± 0.004)	0.035 (± 0.009)	0.000 (± 0.000)
8	0.665 (± 0.002)	0.668 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.000 (± 0.000)	0.019 (± 0.000)	0.013 (± 0.000)	0.000 (± 0.000)
Média	**0.817 (± 0.003)**	0.665 (± 0.000)	0.532 (± 0.000)	0.516 (± 0.000)	0.487 (± 0.003)	0.461 (± 0.005)	0.516 (± 0.000)	0.516 (± 0.000)	0.530 (± 0.000)	0.436 (± 0.002)	0.450 (± 0.003)	0.471 (± 0.000)

Figura 5.1 – Distribuição dos *scores* para todos os algoritmos aplicados no conjunto de dados com exemplos de textos completos de notícias.



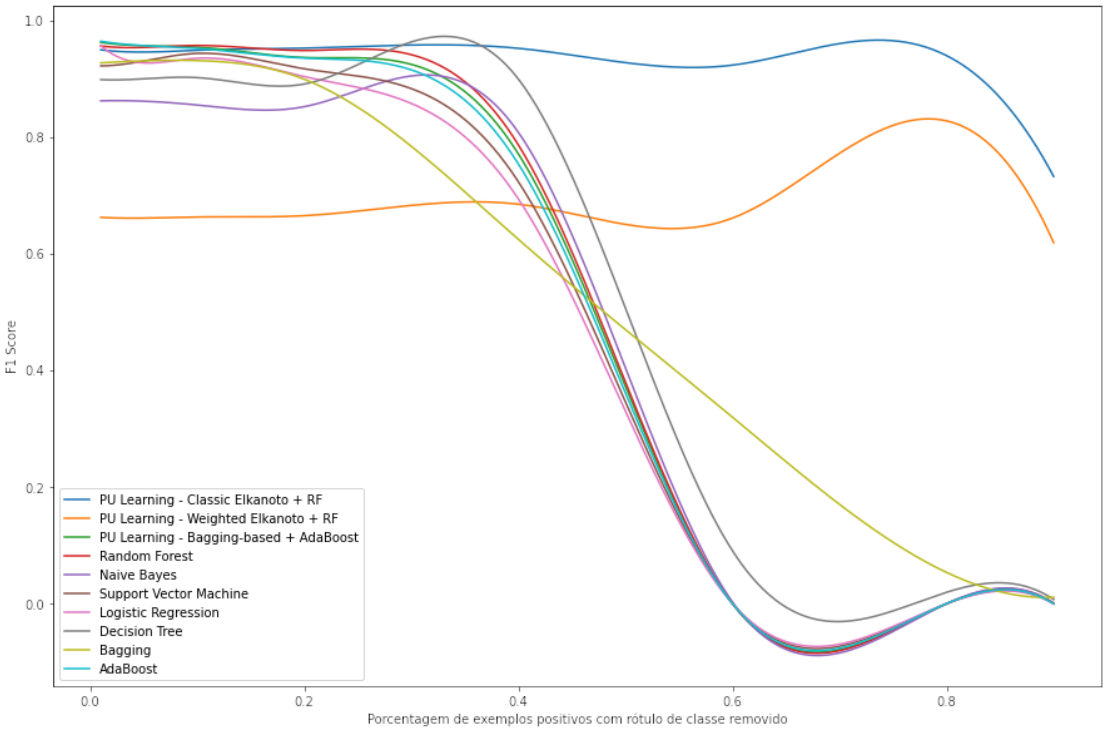
Fonte: Autor

Figura 5.2 – Distribuição dos *scores* para todos os algoritmos aplicados no conjunto de dados com exemplos de textos truncados de notícias.



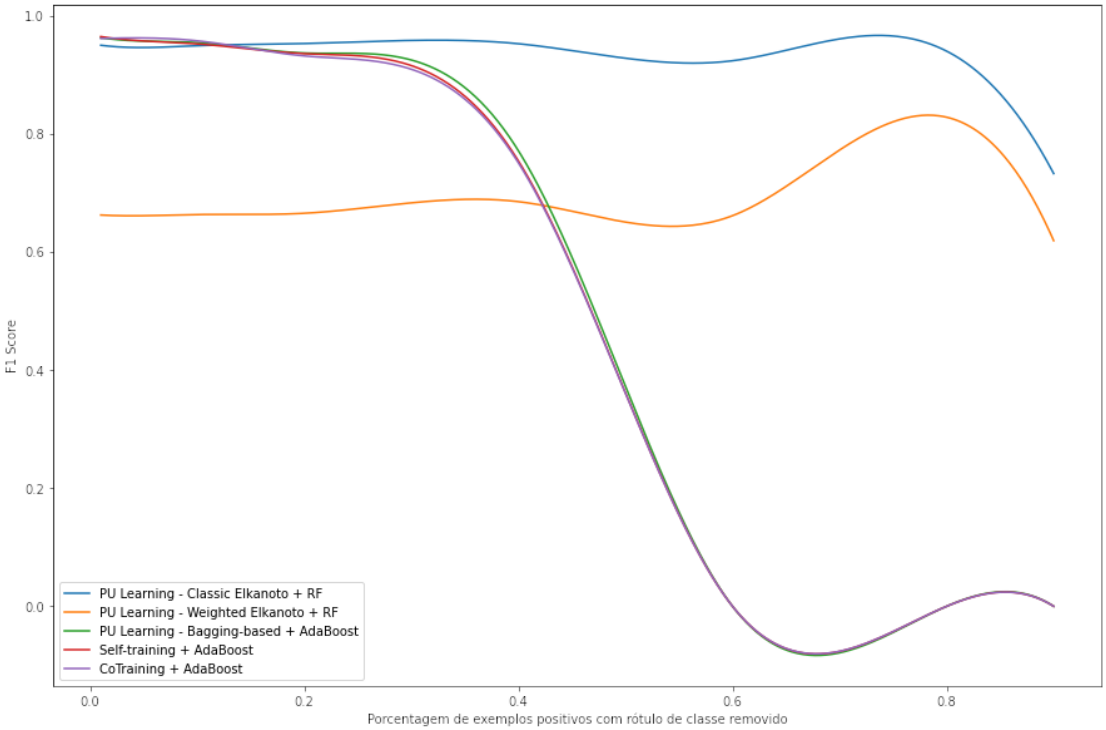
Fonte: Autor

Figura 5.3 – *Scores* dos algoritmos supervisionados e semi-supervisionados baseados em PU *learning* a medida que exemplos positivos de textos completos de notícias é removido o rótulo de classe.



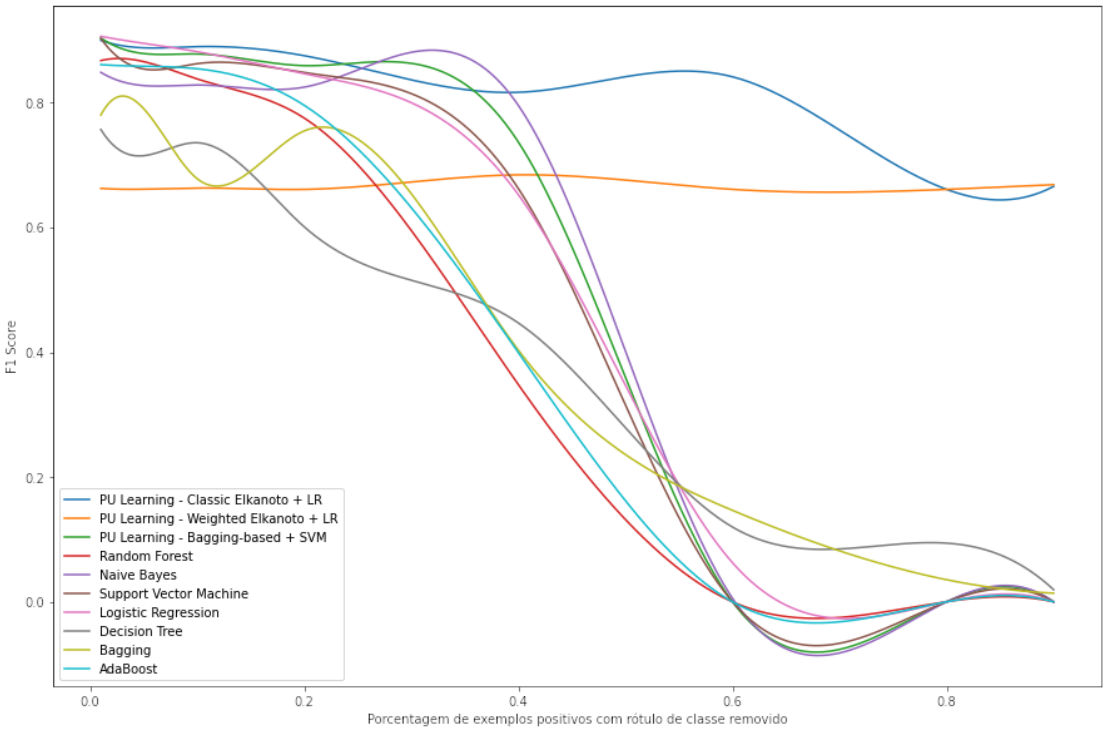
Fonte: Autor

Figura 5.4 – *Scores* dos algoritmos semi-supervisionados e semi-supervisionados baseados em PU *learning* a medida que exemplos positivos de textos completos de notícias é removido o rótulo de classe.



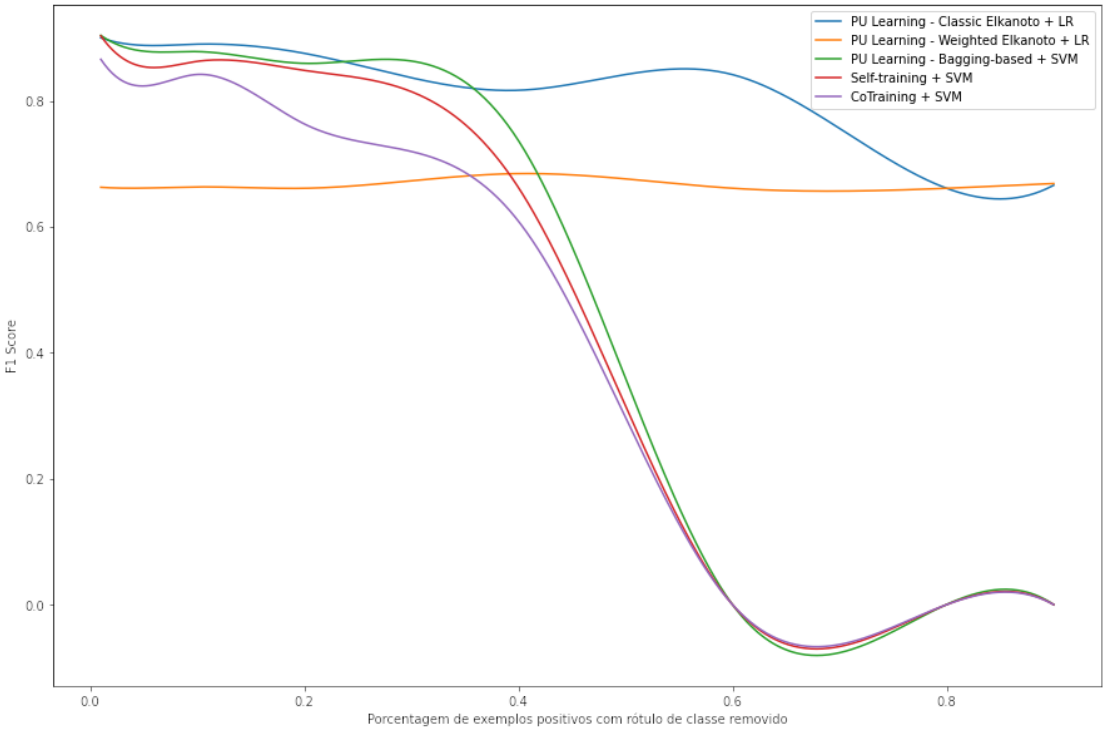
Fonte: Autor

Figura 5.5 – *Scores* dos algoritmos supervisionados e semi-supervisionados baseados em PU *learning* a medida que exemplos positivos de textos truncados de notícias é removido o rótulo de classe.



Fonte: Autor

Figura 5.6 – *Scores* dos algoritmos semi-supervisionados e semi-supervisionados baseados em PU *learning* a medida que exemplos positivos de textos truncados de notícias é removido o rótulo de classe.



Fonte: Autor

5.2 Interpretação dos Resultados

Os resultados demonstram que os métodos (algoritmos) baseados em PU *learning* podem ser aplicados em conjuntos de dados que possuem exemplos de notícias de outros idiomas como o Português Brasileiro.

Além disso, os métodos baseados em PU *learning* revelam que é possível alcançar uma capacidade de generalização eficiente em um cenário onde se há poucos dados (exemplos) rotulados.

As Tabelas de 5.2 a 5.7 demonstram que o BPU utilizando seus parâmetros com valor padrão, não apresenta bons resultados ao ser comparado com os mesmos obtidos pelo PUCE e pelo PUWE. Assim faz-se necessário ajustar tais parâmetros de modo que se encontre os valores mais adequados para que o BPU obtenha melhores resultados no experimento. O PUCE e PUWE quando combinados com SVM não tiveram bons resultados, embora na literatura seja citado o SVM como um bom algoritmo para tratar problemas de classificação de texto. De acordo com Batuwita e Palade (2013), o SVM é sensível ao desequilíbrio no conjunto de dados, ou seja, produz classificadores (modelos) abaixo do ideal quando aplicado em um conjunto de dados desbalanceado. Essa sensibilidade faz com que o hiperplano do SVM se incline mais para a classe minoritária (exemplos sem rótulo de classe).

A Tabela 5.10 mostra que o classificador gerado após treinar o algoritmo de aprendizado semi-supervisionado PU *learning Classic Elkanoto* combinado com RF (PUCE + RF) obteve um melhor desempenho em todos os cenários. À medida que um percentual de exemplos positivos é removido o rótulo de classe em cada execução no experimento, o classificador atingiu um nível de acerto em média de 92%.

O trabalho de Silva *et al.* (2020) mostrou que o classificador LR obteve melhor resultado entre os classificadores gerados a partir do treinamento dos algoritmos supervisionados e utilizando o mesmo conjunto de dados deste trabalho. Mas neste trabalho quando o cenário muda em cada execução o LR perde desempenho, obtendo em média apenas 58% de acerto. O LR possui a mesma fraqueza que o SVM quando aplicado em conjunto de dados desbalanceados, pois trata-se de um classificador linear como o SVM. O LR subestima a probabilidade de eventos raros (exemplos positivos com rótulo de classe), porque esses eventos tendem a ser enviesados por exemplos da classe menos importante (exemplos sem rótulo de classe) (MAALOUF; SIDDIQI, 2014).

A Tabela 5.11 mostra que o classificador PUCE + RF tem um resultado um pouco inferior se comparado ao resultado da Tabela 5.9. Esse resultado é questionado no trabalho de Silva *et al.* (2020) que identifica que os textos quando truncados podem perder características importantes para o treinamento de um algoritmo.

A partir dos resultados encontrados é possível responder as seguintes perguntas abaixo:

1. É possível identificar *fake news* em notícias do idioma Português Brasileiro, levando em conta que o conjunto de dados analisado possui uma menor parte dos dados que são conhecidos (com rótulo) e uma maior parte dos dados que são desconhecidos (sem rótulo)?

   Resposta: Sim, a partir do uso de algoritmos semi-supervisionados baseados em PU *learning*. Esse tipo de algoritmo mantém um nível de acerto alto a medida que se torna escasso o número de dados conhecidos (exemplos positivos).

2. Qual o grau da capacidade de generalização dos algoritmos supervisionados e semi-supervisionados em relação aos algoritmos de PU *learning*, aplicados em problemas de classificação de textos de notícias, onde o conjunto de dados possui poucos exemplos com rótulo *fake*?

   Resposta: Como esperado é abaixo em relação aos demais. Pois, à medida que o número de exemplos com rótulo *fake* é reduzido em relação ao todo conjunto de dados, o desempenho tende a diminuir dos algoritmos supervisionados e semi-supervisionados.

Capítulo 6

CONCLUSÃO E TRABALHOS FUTUROS

Durante a produção deste trabalho foi verificado que há diversas pesquisas com intuito de desenvolver técnicas para identificar *fake news* em notícias publicadas na Internet. Grande parte destas técnicas dependem de informações adicionais (metadados) para obter bons resultados. Um dos problemas é conseguir identificar *fake news* em um universo de notícias, em que sua fonte de criação busca esconder seu verdadeiro conteúdo, passando por mudanças recorrentes para se ocultar cada vez mais.

Diversas técnicas e ferramentas foram avaliadas para fazer o ajuste dos hiperparâmetros dos algoritmos em questão, entre elas se destaca-se o *irace* (LÓPEZ-IBÁÑEZ *et al.*, 2016) e a biblioteca *pycaret*[1]. No caso do *irace* sua implementação é feita na linguagem R[2], enquanto que todo trabalho foi feito utilizando a linguagem *Python*, logo seria necessário fazer um modificação no código do trabalho para ocorrer a integração de ambos. Já o *pycaret* não possui internamente implementado os algoritmos de aprendizado semi-supervisionados como o *self-training* e *co-training*. Por fim, optou-se pelo *RandomizedSearchCV* por fazer parte da biblioteca *scikit-learn*.

O Aprendizado de Máquina é uma ferramenta para o desenvolvimento de técnicas para identificar *fake news*, sendo que cada algoritmo tem suas vantagens e desvantagens quando aplicados em tarefas de classificação de texto. Há uma escassez de informações sobre os dados que serão utilizados para treinamento desse algoritmos, pois os algoritmos de aprendizado supervisionado dependem da necessidade de ter rótulo de classe. Rótulos de classe são muitas vezes caros e dependentes de conhecimento de especialistas de domínio.

Os algoritmos baseados em PU *learning* superam essa barreira de escassez de conjuntos de treinamento anotados com rótulo de classe, pois conseguem atuar em cenários

[1] https://pycaret.org/
[2] https://www.r-project.org/

onde se tem poucas informações sobre uma classe majoritária, ou seja, onde se tem poucos exemplos de notícias com rótulo de *fake news* e a presença de apenas uma classe (positiva) no conjunto de dados.

Nos experimentos, os algoritmos baseados em PU *learning* utilizados foram o *Classic Elkanoto*, o *Weighted Elkanoto* e o *Bagging* baseado em PU *learning*. Com base nos experimentos conclui-se que o *Classic Elkanoto* e o *Weighted Elkanoto* tiveram melhores resultados, ou seja, conseguiram detectar *fake news* quando se tem poucos dados conhecidos, isto é, conseguiram manter a taxa de acerto mesmo com poucos dados com rótulo de classe *fake news*. Embora o *Bagging* baseado PU *learning* esteja na mesma categoria de algoritmos baseados em PU *learning*, não teve bons resultados nos experimentos.

6.1 Trabalhos Futuros

No decorrer da produção deste trabalho, desde o levantamento dos trabalhos relacionados ao tema até a realização dos experimentos, foram encontradas diversas outras técnicas ou abordagens para detecção de *fake news* que podem ser aplicadas futuramente nesse trabalho de modo a contribuir para comunidade científica. Tais atividades podem ser pontuadas como:

1. Desenvolver um algoritmo que faz automaticamente ajustes do hiperparâmetros para os algoritmos que não foram aplicados o *RandomizedSearcCV*. Lembrando que para o algoritmo semi-supervisionado *co-training* são necessárias duas *views*, assim se faz necessário ter dois parâmetros como entrada para cada fatia do conjunto de dados,

2. Fazer uso dos metadados do conjunto de dados Fake.Br e outras ferramentas de linguística para extrair mais características que possam ser relevantes para melhorar o desempenho dos classificadores,

3. Utilizar todas as características ao aplicar a técnica de TF-IDF e não apenas 3.000 como foi feito no trabalho,

4. Identificar os hiperparâmetros que não podem ser combinados, pois alguns algoritmos implementados pela biblioteca *scikit-learn* têm isso como critério,

5. Fazer uso de outras ferramentas para otimização de parâmetros como *GridSearchCV*[3] da biblioteca *scikit-learn*, em um ambiente que não tenha problemas de complexidade computacional, ou seja, limitação de recursos de *hardware*,

6. Aplicar métodos de *deep learning* e comparar com os algoritmos baseados em PU *learning*.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

REFERÊNCIAS

AGARWAL, B.; MITTAL, N. Text classification using machine learning methods-a survey. In: *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012*. [S.l.]: Springer, 2014. p. 701–709. Citado na página 28.

AGRAWALA, A. K. Learning with a probabilistic teacher. *IEEE Trans. Inf. Theory*, v. 16, n. 4, p. 373–379, 1970. Citado na página 39.

ALBALATE, A.; MINKER, W. Semi-Supervised and Unsupervised Machine Learning: Novel Strategies. In: . [S.l.: s.n.], 2011. Citado 2 vezes nas páginas 40 e 41.

ALBON, C. *Machine learning with python cookbook: Practical solutions from preprocessing to deep learning*. [S.l.]: "O'Reilly Media, Inc.", 2018. Citado na página 61.

AMINI, M.-R.; USUNIER, N. *Learning with Partially Labeled and Interdependent Data*. [S.l.]: Springer Publishing Company, Incorporated, 2015. ISBN 3-319-15725-6. Citado na página 38.

ANTHONY, M.; BARTLETT, P. L. *Neural network learning: Theoretical foundations*. [S.l.]: cambridge university press, 2009. Citado na página 32.

BALMAS, M. When Fake News Becomes Real: Combined Exposure to Multiple News Sources and Political Attitudes of Inefficacy, Alienation, and Cynicism. *Communication Research*, v. 41, n. 3, p. 25, 2014. Citado na página 20.

BATUWITA, R.; PALADE, V. Class imbalance learning methods for support vector machines. *Imbalanced learning: Foundations, algorithms, and applications*, Wiley Online Library, p. 83–99, 2013. Citado na página 77.

BEKKER, J.; DAVIS, J. Learning from positive and unlabeled data: a survey. *Machine Learning*, v. 109, n. 4, p. 719–760, 2020. Citado 8 vezes nas páginas 18, 21, 41, 42, 44, 45, 46 e 47.

BEKKER, J.; ROBBERECHTS, P.; DAVIS, J. *Beyond the Selected Completely At Random Assumption for Learning from Positive and Unlabeled Data*. [S.l.: s.n.], 2019. _eprint: 1809.03207. Citado 2 vezes nas páginas 43 e 44.

BILBRO, R.; OJEDA, T.; BENGFORT, B. *Applied Text Analysis with Python*. [S.l.]: O'Reilly Media, Incorporated, 2018. ISBN 978-1-4919-6303-6. Citado na página 59.

BLUM, A.; MITCHELL, T. Combining Labeled and Unlabeled Data with Co-Training. In: *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*. New York, NY, USA: Association for Computing Machinery, 1998. (COLT' 98), p. 92–100. ISBN 1-58113-057-0. Event-place: Madison, Wisconsin, USA. Citado na página 48.

BONDIELLI, A.; MARCELLONI, F. A survey on fake news and rumour detection techniques. *Information Sciences*, v. 497, p. 38–55, 2019. Citado na página 23.

BOUTABA, R. *et al.* A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *Journal of Internet Services and Applications*, v. 9, n. 1, p. 1–99, 2018. Publisher: Springer. Citado na página 28.

BREIMAN, L. *Bagging Predictors*. [S.l.: s.n.], 1994. Citado na página 58.

BURKOV, A. *The Hundred-Page Machine Learning Book*. [S.l.]: Andriy Burkov, 2019. ISBN 978-1-9995795-1-7. Citado 8 vezes nas páginas 27, 28, 29, 31, 33, 34, 44 e 45.

CHAPELLE, O.; SCHLKOPF, B.; ZIEN, A. *Semi-Supervised Learning*. 1st. ed. [S.l.]: The MIT Press, 2010. ISBN 0-262-51412-5. Citado 3 vezes nas páginas 38, 39 e 40.

CUNNINGHAM, P.; DELANY, S. J. *k-Nearest Neighbour Classifiers: 2nd Edition (with Python examples)*. [S.l.: s.n.], 2020. _eprint: 2004.04523. Citado na página 32.

CUTLER, A.; CUTLER, D. R.; STEVENS, J. R. Random forests. In: *Ensemble machine learning*. [S.l.]: Springer, 2012. p. 157–175. Citado 2 vezes nas páginas 34 e 35.

DENG, H. *et al.* Semi-Supervised Learning Based Fake Review Detection. In: *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*. [S.l.: s.n.], 2017. p. 1278–1280. Citado 3 vezes nas páginas 52, 54 e 55.

DENIS, F. *et al.* Text Classification and Co-training from Positive and Unlabeled Examples. *Proceedings of the ICML 2003 Workshop: The Continuum from Labeled to Unlabeled Data*, p. 8, 2003. Citado na página 48.

ELKAN, C.; NOTO, K. Learning classifiers from only positive and unlabeled data. In: *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08*. Las Vegas, Nevada, USA: ACM Press, 2008. p. 213. ISBN 978-1-60558-193-4. Citado 3 vezes nas páginas 41, 44 e 58.

FACELI, K. *et al. Inteligência artificial: uma abordagem de aprendizado de máquina*. [S.l.]: LTC, 2011. Citado 5 vezes nas páginas 21, 26, 27, 28 e 36.

FRALICK, S. C. Learning to recognize patterns without a teacher. *IEEE Trans. Inf. Theory*, v. 13, n. 1, p. 57–64, 1967. Citado na página 39.

FUNG, G. P. C. *et al.* Text classification without negative examples revisit. *IEEE transactions on Knowledge and Data Engineering*, IEEE, v. 18, n. 1, p. 6–20, 2005. Citado na página 49.

FUSILIER, D. H. *et al.* Detecting positive and negative deceptive opinions using PU-learning. *Information Processing & Management*, v. 51, n. 4, p. 433–443, 2015. Citado 3 vezes nas páginas 51, 54 e 55.

GéRON, A. *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. Sebastopol, CA: O'Reilly Media, 2017. ISBN 978-1-4919-6229-9. Citado 3 vezes nas páginas 29, 58 e 60.

HARRINGTON, P. *Machine learning in action*. [S.l.]: Simon and Schuster, 2012. Citado na página 36.

HASSANZADEH, H. *et al. Clinical Document Classification Using Labeled and Unlabeled Data Across Hospitals*. [S.l.: s.n.], 2018. _eprint: 1812.00677. Citado na página 39.

HASTIE, T. *et al.* Multi-class adaboost. *Statistics and its Interface*, International Press of Boston, v. 2, n. 3, p. 349–360, 2009. Citado na página 35.

HE, D. *et al.* Fake Review Detection Based on PU Learning and Behavior Density. *IEEE Network*, v. 34, n. 4, p. 298–303, 2020. Citado 5 vezes nas páginas 18, 52, 53, 54 e 55.

HEARST, M. A. *et al.* Support vector machines. *IEEE Intelligent Systems and their applications*, v. 13, n. 4, p. 18–28, 1998. Publisher: IEEE. Citado na página 32.

KOZHEVNIKOV, V. A.; PANKRATOVA, E. S. Research of the Text Data Vectorization and Classification Algorithms of Machine Learning. *Theoretical & Applied Science*, n. 5, p. 574–585, 2020. Publisher: Limited Liability Partnership Theoretical and Applied Science. Citado 2 vezes nas páginas 60 e 65.

LI, H. *et al.* Spotting Fake Reviews using Positive-Unlabeled Learning. *Computación y Sistemas*, v. 18, n. 3, p. 9, 2014. Citado 4 vezes nas páginas 18, 51, 54 e 55.

LI, M.; XIAO, P.; ZHANG, J. Text classification based on ensemble extreme learning machine. *arXiv preprint arXiv:1805.06525*, 2018. Citado na página 35.

LI, X.; LIU, B. Learning to Classify Texts Using Positive and Unlabeled Data. *Morgan Kaufmann Publishers Inc.*, p. 6, 2003. Citado na página 49.

LIU, B. *et al.* Building text classifiers using positive and unlabeled examples. In: *Third IEEE International Conference on Data Mining*. Melbourne, FL, USA: IEEE Comput. Soc, 2003. p. 179–186. ISBN 978-0-7695-1978-4. Citado 2 vezes nas páginas 21 e 49.

____. Partially Supervised Classification of Text Documents. p. 8, 2002. Citado na página 48.

LIU, L.; PENG, T. Clustering-based Method for Positive and Unlabeled Text Categorization Enhanced by Improved TFIDF. *J. Inf. Sci. Eng.*, p. 1463–1481, 2014. Citado na página 50.

LIU, Y.; WU, Y.-F. B. FNED: A Deep Network for Fake News Early Detection on Social Media. *ACM Transactions on Information Systems*, v. 38, n. 3, p. 1–33, 2020. Citado 4 vezes nas páginas 50, 51, 54 e 55.

LÓPEZ-IBÁÑEZ, M. *et al.* The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, Elsevier, v. 3, p. 43–58, 2016. Citado na página 79.

LU, F.; BAI, Q. Semi-supervised text categorization with only a few positive and unlabeled documents. In: *2010 3rd International Conference on Biomedical Engineering and Informatics*. Yantai, China: IEEE, 2010. p. 3075–3079. ISBN 978-1-4244-6498-2 978-1-4244-6495-1 978-1-4244-6497-5. Citado na página 50.

MAALOUF, M.; SIDDIQI, M. Weighted logistic regression for large-scale imbalanced and rare events data. *Knowledge-Based Systems*, Elsevier, v. 59, p. 142–148, 2014. Citado na página 77.

MITCHELL, M. *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998. ISBN 0-262-63185-7. Citado 2 vezes nas páginas 25 e 27.

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 3 vezes nas páginas 26, 27 e 31.

MORDELET, F.; VERT, J.-P. *A bagging SVM to learn from positive and unlabeled examples*. [S.l.: s.n.], 2010. _eprint: 1010.0772. Citado na página 58.

MUELLER, J.; MASSARON, L. *Machine Learning For Dummies*. [S.l.]: Wiley, 2016. (For dummies). ISBN 978-1-119-24551-3. Citado 3 vezes nas páginas 26, 29 e 45.

MUHAMMAD, I.; YAN, Z. Supervised Machine Learning Approaches: A Survey. *ICTACT Journal on Soft Computing*, v. 5, n. 3, 2015. Citado na página 28.

MüLLER, A.; GUIDO, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. In: . [S.l.: s.n.], 2016. Citado na página 30.

PENG, T.; ZUO, W.; HE, F. SVM based adaptive learning method for text classification from positive and unlabeled documents. *Knowledge and Information Systems*, v. 16, n. 3, p. 281–301, 2008. Citado na página 49.

REZENDE, S. O. *Sistemas Inteligentes: Fundamentos e Aplicações*. Barueri, SP: Editora Manole Ltda, 2003. ISBN 85-204-1683-7. Citado na página 25.

RISH, I.; others. An empirical study of the naive Bayes classifier. In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. [S.l.: s.n.], 2001. v. 3, p. 41–46. Issue: 22. Citado na página 32.

ROSENBERG, C.; HEBERT, M.; SCHNEIDERMAN, H. Semi-Supervised Self-Training of Object Detection Models. In: *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*. [S.l.: s.n.], 2005. v. 1, p. 29–36. Citado na página 39.

ROUT, J. K. *et al.* Revisiting Semi-Supervised Learning for Online Deceptive Review Detection. *IEEE Access*, v. 5, p. 1319–1327, 2017. Citado 3 vezes nas páginas 51, 54 e 55.

SAFAVIAN, S. R.; LANDGREBE, D. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, v. 21, n. 3, p. 660–674, 1991. Publisher: IEEE. Citado na página 32.

SCHAPIRE, R. E. Explaining adaboost. In: *Empirical inference*. [S.l.]: Springer, 2013. p. 37–52. Citado na página 36.

SCUDDER, H. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, v. 11, n. 3, p. 363–371, 1965. Citado na página 39.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. *Understanding Machine Learning - From Theory to Algorithms*. [S.l.]: Cambridge University Press, 2014. ISBN 978-1-107-05713-5. Citado na página 29.

Shinde, M. P.; Channe, H. Semi-supervised Learning with Ensemble Method for Online Deceptive Review Detection. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, v. 3, 2018. Publisher: Technoscience Academy. Citado 3 vezes nas páginas 52, 54 e 55.

SHUQIN, Y.; JING, F. Fake Reviews Detection Based on Text Feature and Behavior Feature. In: *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. [S.l.: s.n.], 2019. p. 2007–2012. Citado 3 vezes nas páginas 52, 54 e 55.

SILVA, N. F. F. d. *Análise de sentimentos em textos curtos provenientes de redes sociais*. Tese (Doutorado em Ciências de Computação e Matemática Computacional) — Universidade de São Paulo, São Carlos, 2016. Citado na página 37.

SILVA, N. F. F. D.; COLETTA, L. F. S.; HRUSCHKA, E. R. A Survey and Comparative Study of Tweet Sentiment Analysis via Semi-Supervised Learning. *ACM Comput. Surv.*, v. 49, n. 1, 2016. Place: New York, NY, USA Publisher: Association for Computing Machinery. Citado na página 40.

SILVA, R. M. *et al.* Towards automatically filtering fake news in Portuguese. *Expert Systems with Applications*, v. 146, p. 113199, 2020. Citado 5 vezes nas páginas 48, 56, 58, 67 e 77.

SRINIVASA-DESIKAN, B. *Natural Language Processing and Computational Linguistics: A practical guide to text analysis with Python, Gensim, spaCy, and Keras*. [S.l.]: Packt Publishing, 2018. ISBN 978-1-78883-703-3. Citado na página 60.

TAULLI, T. *Artificial Intelligence Basics: A Non-Technical Introduction*. [S.l.]: Apress, 2019. ISBN 978-1-4842-5028-0. Citado na página 26.

VERIKAS, A. *et al.* Electromyographic patterns during golf swing: Activation sequence profiling and prediction of shot effectiveness. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 16, n. 4, p. 592, 2016. Citado na página 35.

WAZLAWICK, R. S. *Metodologia de Pesquisa para Ciência da Computação*. Second. [S.l.]: Elsevier, 2014. ISBN 978-85-352-7782-1. Citado na página 21.

YU, H.-F.; HUANG, F.-L.; LIN, C.-J. Dual Coordinate Descent Methods for Logistic Regression and Maximum Entropy Models. *Mach. Learn.*, v. 85, n. 1–2, p. 41–75, 2011. Place: USA Publisher: Kluwer Academic Publishers. Citado na página 58.

YU, S.; LI, C. PE-PUC: A Graph Based PU-Learning Approach for Text Classification. In: PERNER, P. (Ed.). *Machine Learning and Data Mining in Pattern Recognition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 574–584. ISBN 978-3-540-73499-4. Citado na página 49.

ZHANG, D. *et al.* What Online Reviewer Behaviors Really Matter? Effects of Verbal and Nonverbal Behaviors on Detection of Fake Online Reviews. *Journal of Management Information Systems*, v. 33, p. 456 – 481, 2016. Citado 2 vezes nas páginas 24 e 25.

ZHANG, X.; GHORBANI, A. A. An overview of online fake news: Characterization, detection, and discussion. *Information Processing & Management*, v. 57, n. 2, p. 102025, 2020. Citado 6 vezes nas páginas 18, 20, 21, 23, 24 e 25.

ZHU, X. *et al. Introduction to Semi-Supervised Learning*. [S.l.]: Morgan and Claypool Publishers, 2009. ISBN 1-59829-547-0. Citado 5 vezes nas páginas 36, 37, 38, 39 e 41.