UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS DOIS VIZINHOS
CURSO DE ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS

NELSON JACOB DRESSLER

CLASSIFICAÇÃO DE TEXTOS ESCOLARES COM APRENDIZADO DE MÁQUINA E LÓGICA FUZZY

TRABALHO DE CONCLUSÃO DE CURSO

DOIS VIZINHOS
2022

NELSON JACOB DRESSLER

CLASSIFICAÇÃO DE TEXTOS ESCOLARES COM APRENDIZADO DE MÁQUINA E LÓGICA FUZZY

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Orientador: Prof. Dr. Rafael Gomes Mantovani
Coorientador: Prof. Dr. Francisco Carlos Monteiro

DOIS VIZINHOS
2022

NELSON JACOB DRESSLER

CLASSIFICAÇÃO DE TEXTOS ESCOLARES COM APRENDIZADO DE MÁQUINA E LÓGICA FUZZY

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Data de aprovação: 26/maio/2022

Rafael Gomes Mantovani
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Apucarana

Anderson Chaves Carniel
Doutorado
Universidade Federal de São Carlos

Luiz Fernando Carvalho
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Apucarana

DOIS VIZINHOS
2022

Dedico este trabalho à minha querida e amada esposa Ana Gabriela e aos meus pais José Manoel e Selma por terem me apoiado e me ajudado sempre e terem acreditado em mim em mais esse momento importante da vida.

AGRADECIMENTOS

Em primeiro lugar, agradeço ao Todo Poderoso, Criador do Universo, pela saúde, sabedoria e sucesso que tem me proporcionado até hoje.

Em segundo lugar, gostaria de agradecer à minha esposa e companheira para a vida toda, Ana Gabriela Ungierowicz Dressler, por me acompanhar nos últimos momentos e mais difíceis do curso, me auxiliando de diversas maneiras, principalmente me encorajando e motivando para desenvolver este trabalho de conclusão.

Em terceiro, gostaria de agradecer aos meus pais, José Manoel Dressler e Selma Kalik Dressler, que, desde sempre, me acompanham nos bons e turbulentos momentos da vida e me apoiam em todas as minhas decisões, além da excelente educação que me foi dada desde pequeno.

À Universidade Tecnológica Federal do Paraná, por ter me dado a oportunidade de entrar no curso de Especialização em Ciência de Dados e por ter oferecido sempre uma ótima qualidade de ensino, com professores renomados e uma plataforma de ensino-aprendizagem bem estruturada para a educação a distância.

Ao meu orientador, Professor e Doutor Rafael Gomes Mantovani, por ter me apoiado durante toda esta jornada do curso, principalmente na realização deste trabalho de conclusão, me mostrando sempre a intuição científica e os conceitos-chave necessários para a implementação do meu estudo.

Por fim, a todos os meus professores e colegas que fizeram parte do período em que estive estudando na instituição, compreendido de Setembro de 2020 à Maio de 2022.

Jamais considere seus estudos como uma obrigação, mas como uma oportunidade invejável para aprender a conhecer a beleza libertadora do intelecto para seu próprio prazer pessoal e para proveito da comunidade à qual seu futuro trabalho pertencer. (Albert Einstein)

RESUMO

Nos dias atuais, em meio ao processo de ensino-aprendizagem, é muito comum alunos buscarem informações acerca das disciplinas com propósitos de complementação de estudos ou para a execução de trabalhos escolares. Para isso, é possível utilizar motores de busca na *Internet*, porém nem sempre os textos possuem a complexidade compatível com o grau de escolaridade do aluno. Uma solução para tal propósito é a filtragem automática de textos, onde são criados modelos computacionais para classificação textual de acordo com seus contextos. Muitos estudos tem apresentado soluções baseadas em técnicas de Aprendizado de Máquina ou Lógica *Fuzzy*, porém não existe um comparativo, nem um estudo analisando a possibilidade da combinação de ambos os métodos. Este trabalho explora este cenário, implementando duas soluções: uma baseada apenas em algoritmos tradicionais de aprendizado de máquina, e outra combinando a execução de uma Árvore de Decisão com a automatização de um sistema de Lógica *Fuzzy*. Em ambas soluções são aplicadas técnicas de Processamento de Linguagem Natural, a fim de selecionar características dos conteúdos textuais e introduzi-los no *pipeline* de classificação. Experimentos foram conduzidos e os resultados obtidos sugerem que, para o cenário explorado, os algoritmos de Aprendizado de Máquina possuem resultado superior, com acurácia balanceada por classes em torno de 0.89 nos dados de teste. Além disso, o uso de modelos de Árvore de Decisão para a criação das regras de um Sistema *Fuzzy* pôde adicionar mais interpretabilidade e compreensão à classificação de textos escolares, porém com uma acurácia balanceada de aproximadamente 0.5, ainda superior a todos os *baselines* escolhidos para o experimento deste trabalho.

Palavras-chave: Classificação de Textos Escolares. Complexidade Textual. Processamento de Linguagem Natural. Aprendizado de Máquina. Lógica *Fuzzy*.

ABSTRACT

Nowadays, during the teaching-learning process, it is prevalent for students to seek information about the subjects to complement studies or execute school work. For this, it is possible to use search engines on the Internet, but the texts do not always have the complexity compatible with the student's level of education. A solution for this purpose is the automatic filtering of texts, where computational models are created for textual classification according to their contexts. Many studies have presented solutions based on Machine Learning or Fuzzy Logic techniques, but there is no comparison nor an analysis considering the possibility of combining both methods. This work explores this scenario, implementing two solutions: one based only on traditional machine learning algorithms and another combining the execution of a Decision Tree with the automation of a Fuzzy Logic system. In both solutions, Natural Language Processing techniques are applied in order to select characteristics of the textual contents and introduce them into the classification pipeline. Experiments were conducted and the results obtained suggest that, for the scenario explored, the Machine Learning algorithms have a superior result, with class-balanced accuracy of around 0.89 in the test data. Furthermore, the use of Decision Tree models to create the rules of a Fuzzy System could add more interpretability and understanding to the classification of school texts, but with a balanced accuracy of approximately 0.5, still superior to all the baselines chosen for the experiment of this work.

Keywords: educational texts classification. text complexity. natural language processing. machine learning. fuzzy logic.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AG	Algoritmos Genéticos
AM	Aprendizado de Máquina
ANFIS	Adaptive Neuro Fuzzy Inference System ou Sistema de Inferência Neuro-Difuso
BAC	*Balanced Accuracy by Classes* ou Acurácia Balanceada por Classes
CNN	*Convolutional Neural Networks* ou Redes Neurais Convolucionais
CV	*Cross Validation* ou Validação Cruzada
DL	*Deep Learning* ou Aprendizado Profundo
DNN	*Deep Neural Networks*
DT	*Decision Tree* ou Árvore de Decisão
EDA	*Exploratory Data Analysis* ou Análise Exploratória de Dados
FN	Falsos Positivos
FP	Falsos Negativos
GS	*Grid Search* ou Busca em Grade
HNET	*Highway Network*
HP	Hiperparâmetro
IA	Inteligência Artificial
IDF	Frequência Inversa de Documentos
K-NN	*K Nearest Neighbors* ou K - Vizinhos mais Próximos
LDB	Lei de Diretrizes e Bases da Educação Nacional
LSTM	*Long Short Term Memory*
ML	*Machine Learning*
MLP	*Multilayer Perceptron* ou Redes Neurais Multicamadas
NB	Naive Bayes

NLP	*Natural Language Processing* ou Processamento de Linguagem Natural
PLN	Processamento de Linguagem Natural
PSO	Otimização de Enxame de Partículas
ReLU	*Rectified Linear Unit* ou Unidade Linear Retificado
RF	*Random Forest* ou Floresta Aleatória
RL	Regressão Logística
RS	*Random Search* ou Busca Aleatória
SEB	Sistema Educacional Brasileiro
SGD	Gradiente Descendente Estocástico
SMBO	Sequential Model-Based Optimization ou Otimização Bayesiana
SVM	Máquinas de Vetores de Suporte
TF	Frequência de Termos
TF-IDF	Frequência de Termos versus Frequência Inversa de Documentos
TM	*Text Mining* ou Mineração de Textos
UTFPR	Universidade Tecnológica Federal do Paraná
VN	Verdadeiros Negativos
VP	Verdadeiros Positivos

SUMÁRIO

1 INTRODUÇÃO

Durante o processo de ensino-aprendizagem, muitos alunos costumam complementar seus estudos acerca das disciplinas ministradas, bem como necessitam acesso a materiais específicos para a realização de trabalhos escolares. Para tal, é possível alcançar a informação desejada por meio de diversos meios de comunicação, tais como: livros, jornais, revistas e, principalmente, a *Internet*, a partir de seus motores de busca.

Porém, os materiais encontrados estão sujeitos a não refletir com exatidão o tema e/ou o grau de escolaridade do aluno, podendo não estar relacionado ou ainda estar acima ou abaixo de suas capacidades, mesmo compatível com o tema em questão. Consequentemente, o estudante acaba tendo a necessidade de filtrar os dados textuais recuperados manualmente, a fim de garantir que os mesmos referenciam de fato o tema de estudo, assim como o nível de complexidade esperado. Isto torna a tarefa de busca de conteúdo árdua e impede que o prazo de conclusão seja estimado com antecedência.

Uma possível solução de suporte à filtragem correta de conteúdos textuais encontrados na *Internet* é por meio de processamento computacional e análise descritiva dos dados, de modo que seja possível classificar ou rotular automaticamente a complexidade textual para os diversos estágios escolares. Neste sentido, visando o processamento computacional sobre dados textuais, é possível utilizar técnicas e conceitos de **Processamento de Linguagem Natural (PLN)**, uma área interdisciplinar situada na Computação, e que faz o uso tanto de técnicas de **Inteligência Artificial (IA)** quanto conceitos de **Linguística** (FINATTO; LOPES; CIULLA, 2015). O objetivo é desenvolver ferramentas que permitam manipular, descrever e até mesmo classificar conteúdos textuais, de modo que a máquina possa compreender a linguagem humana.

1.1 Problema de Pesquisa

Neste trabalho, o principal problema investigado é a automatização do processo de filtragem de dados textuais no contexto de materiais didáticos, auxiliando estudantes na busca de conteúdo complementar na *Internet*, de maneira que este conteúdo também seja compatível com o seu grau de escolaridade.

As soluções existentes na literatura abordam o problema como uma instância de classificação de textos. Para tal, é gerada uma sequência completa de etapas bem específicas (*pipeline*), compreendendo todas as tarefas necessárias para realizar tal classificação. As etapas, por sua vez, compreendem desde: a coleta dos dados; a extração de características; a seleção de modelos preditivos; e concluindo com a avaliação do desempenho destes modelos induzidos.

Alguns trabalhos se baseiam em técnicas de Aprendizado de Máquina (AM) (ZANCHINI, 2019; SILVA, 2019; MCCALLUM; NIGAM, 2001), enquanto outros procuram desenvolver

soluções baseadas em Lógica *Fuzzy* (WU et al., 2017; SUN et al., 2002). Porém, não existem trabalhos que exploram alternativas à essas duas abordagens, ou que as combinam, ou até mesmo comparam seus desempenhos em um mesmo domínio.

Considerando o desafio ilustrado e o cenário atual da literatura específica, este trabalho implementa uma solução que compara e combina algoritmos tradicionais de Aprendizado de Máquina e Lógica *Fuzzy*. Estes algoritmos associam textos didáticos com o seu respectivo grau de escolaridade e, desta forma, é possível construir uma solução automática para esse processamento.

1.2 Justificativa e Contribuições

Considerando a necessidade dos alunos buscarem conteúdos didáticos e materiais de forma complementar aos seus estudos, existe uma escassez de soluções automatizadas. Do ponto de vista prático, isto se acentua, pois, neste contexto, a tarefa de busca se torna algo repetitivo e desmotivador aos estudantes. Tal cenário evidencia e favorece a construção de aplicações computacionais automatizadas, e que contribuam tanto à pesquisa quanto ao aprofundamento dos estudos dos alunos, possibilitando encontrar alternativas de materiais na *Internet* com maior qualidade, atendendo tanto as suas expectativas quanto ao seu nível de escolaridade.

Visando a resolução do problema supramencionado, foi proposta a criação de modelos computacionais combinando algoritmos tradicionais de Aprendizado de Máquina e Lógica *Fuzzy*. Neste caso, a combinação de ambas estratégias seria interessante pelo fato dos textos serem considerados dados não estruturados, logo poderiam não ser claramente rotulados conforme uma classe específica (tópico, assunto, matéria, área, nível). Essa "incerteza" favoreceria a aplicação da Lógica *Fuzzy*, pois conceitos nebulosos e confusos são representados por graus de pertinência, especificando o "quanto" um texto pode pertencer a uma ou mais classes, permitindo uma maior compreensão dos dados analisados.

Além disso, o uso de algoritmos tradicionais de Aprendizado de Máquina, possibilita automatizar a tarefa de classificação dos dados textuais.

Por fim, a literatura carece de soluções híbridas, que combinem ambas as estratégias. A grande maioria dos trabalhos publicados apresenta o desenvolvimento de apenas uma delas. Entretanto, o uso de ambas abordagens em conjunto poderia oferecer maior robustez e interpretabilidade aos resultados obtidos.

1.3 Objetivos

O objetivo principal deste trabalho é desenvolver uma solução computacional automatizada, que auxilie e apoie os alunos na descrição e filtragem de textos didáticos, classificando os dados e associando a complexidade e estrutura textual com os diversos estágios escolares,

conforme rege o Sistema Educacional Brasileiro (SEB), pautado pela Lei de Diretrizes e Bases da Educação Nacional (LDB) (BRASIL, 1996).

Especificamente, objetiva-se:

- Classificar dados textuais com base na indução de modelos preditivos, utilizando algoritmos de AM;
- Aplicar técnicas de PLN para realizar a extração e seleção de características dos dados textuais, de modo que reflita diretamente e melhor represente cada texto;
- Combinar algoritmos de AM e Lógica *Fuzzy* para a obtenção de uma solução mais robusta de classificação de textos, com maior grau de assertividade e, principalmente, proporcionando interpretabilidade no processo de decisão; e
- Contribuir com a área de Ciência de Dados e aplicações em análise preditiva e PLN, incentivando pesquisas científicas em áreas afins.

1.4 Organização do Trabalho

O restante do trabalho está dividido da seguinte forma: o Capítulo 2 apresenta fundamentos teóricos necessários para o desenvolvimento da solução proposta neste trabalho; o Capítulo 3 lista os trabalhos relacionados da literatura que propuseram soluções similares para o mesmo problema estudado; o Capítulo 4 descreve detalhadamente a metodologia experimental empregada para a condução dos experimentos; o Capítulo 5 apresenta e discute os principais resultados obtidos após a realização do(s) experimento(s); e, por fim, as considerações finais, limitações e propostas de trabalhos futuros são apresentadas no Capítulo 6.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos teóricos necessários para o desenvolvimento da solução proposta neste trabalho. Serão apresentados conceitos sobre as duas grandes áreas do trabalho: Processamento de Linguagem Natural e Aprendizado de Máquina. Além disso, também serão abordados conceitos necessários sobre o processo de validação e avaliação de experimentos automatizados.

2.1 Processamento de Linguagem Natural

Processamento de Linguagem Natural (PLN) (BARBOSA et al., 2017), do inglês *Natural Language Processing* (NLP), é uma área de pesquisa que explora técnicas de representação e manipulação de textos, com o propósito de criar modelos computacionais capazes de simular a linguagem humana, abstraindo tarefas como leitura e escrita. Dentre estas tarefas mais comuns realizadas nessa área, é possível citar: classificação de textos, sumarização, análise de sentimentos, tradução automática, entre outras.

Durante o desenvolvimento de uma solução baseada no processamento de textos, existem diferentes técnicas que podem ser exploradas para a extração de características ou remoção de informação que não seja útil ao domínio. Esse processo é comumente referido como pré-processamento textual. Na prática, existem diversos métodos de pré-processamento de textos (ZANCHINI, 2019), os mais comuns e que valem ser destacados são:

- **Uniformização dos textos:** consiste na transformação do texto para apenas um único padrão de escrita, convertendo os dados textuais em letras minúsculas. Nessa etapa, caso necessário, também são removidos os acentos das letras;
- **Remoção de ruídos:** consiste na remoção de dados textuais que não possuem relevância informativa, como espaços adicionais, pontuação ou demais caracteres especiais irrelevantes à análise textual;
- **Remoção de palavras vazias (stopwords):** consiste na remoção de termos sem representatividade semântica no texto, como artigos e preposições;
- **Stemização (stemming):** consiste no processo de redução das palavras ao seu radical, desconsiderando sufixos;
- **Vetorização ou Indexação:** consiste em organizar todos os termos presentes no *corpus* (conjunto de documentos) em um vetor e atribuir valores numéricos a cada posição. Estes valores representam as frequências absolutas de cada termo no documento, tratando as palavras presentes em um texto como um saco de palavras, do original em inglês *bag-of-words* (MORAIS, 2007);

- **TF-IDF:** de forma análoga à vetorização, esta técnica consiste em atribuir pesos conforme a frequência de um termo num documento específico, do inglês *Term Frequency* (TF), e ao mesmo tempo, ponderar negativamente com a frequência alta do mesmo termo em diversos documentos, do inglês *Inverse Term Frequency* (IDF), atribuindo um peso menor no documento corrente, caso este seja frequente em diversos documentos;
- **Seleção de Características:** consiste na redução de dimensionalidade do espaço de atributos de entrada, a partir da seleção das características que melhor representem os dados na sua totalidade. Neste contexto, os textos (entrada) são associados às classes (saída) e, com base em testes estatísticos univariados, são atribuídas pontuações para cada termo, representando uma característica específica. Como consequência, estes termos são filtrados e ranqueados os K atributos mais relevantes no *dataset* completo (PINTAS; FERNANDES; GARCIA, 2021); e
- **Discretização ou Binarização:** consiste em converter valores contínuos em inteiros que representem estratos de dados. Cada nova categoria possui um significado ou valor, como, por exemplo: alto, médio e baixo; superior, inferior; etc.

Segundo Morais (2007), PLN é considerado apenas uma parte específica da linguística, cujas técnicas compõem também a análise semântica, uma abordagem pertencente à Mineração de Textos, do inglês *Text Mining* (TM). Esta análise ocorre especificamente no entendimento e exploração da organização das palavras e sua função em um texto, relevantes para o contexto e compreensão do problema. Já a análise estatística, outra abordagem da TM, mensura a relevância dos termos nos textos a partir de sua frequência em cada documento, ignorando sua função na estrutura textual. Algumas das técnicas desta última abordagem foram listadas anteriormente como etapas de pré-processamento textual (vetorização, TF-IDF, seleção de características e discretização ou binarização).

2.2 Aprendizado de Máquina

O Aprendizado de Máquina (AM) (MITCHELL, 1997), do inglês *Machine Learning* (ML), é uma subárea da IA, cuja característica intrínseca é a construção de modelos matemáticos que visam mapear e identificar padrões nos dados. Isso é feito pela indução de modelos, que podem ser supervisionados ou não-supervisionados, para resolver problemas práticos e extrair conhecimento útil do domínio de estudo. Modelos supervisionados são gerados por algoritmos que trabalham com dados rotulados, gerando tarefas de classificação ou regressão. Por outro lado, modelos não-supervisionados são gerados por meio de algoritmos de agrupamento de dados (*clustering*), que não possuem essa informação de rótulo dos dados, e tentam organizar/categorizar as informações apenas com base em suas características. Independente do tipo de tarefa, é possível automatizar o processo de indução de modelos, viabilizando a inferência e a conclusão sobre dados desconhecidos ou futuros.

2.2.1 Paradigmas de Aprendizado

Considerando a manipulação de dados rotulados, a indução dos modelos preditivos pode ocorrer através de diversos algoritmos (FACELI et al., 2011), seguindo diferentes vieses de aprendizado, e organizados em diferentes paradigmas. Alguns deles são:

- **Algoritmos baseados em distâncias:** realizam predições com base na noção de "proximidade" dos dados. Novos exemplos são classificados de acordo com sua vizinhança no espaço de entrada. Neste paradigma, é possível citar o algoritmo dos K vizinhos mais próximos (K-NN), do inglês *k-Nearest Neighbors*;
- **Algoritmos probabilísticos:** realizam predições com base no Teorema de Bayes, onde é avaliada a probabilidade de ocorrência de um evento (classe), dado um conjunto de valores do conjunto de entrada (atributos). Neste paradigma, é possível citar os algoritmos de Bernoulli e multinomial;
- **Algoritmos baseados em procura:** realizam predições com base na exploração de um espaço de possíveis soluções. Neste paradigma, é possível citar as Árvores de Decisão, que, a cada iteração, realiza cortes ortogonais no espaço do domínio, sempre selecionando o melhor atributo que reduz ao máximo a incerteza na identificação dos padrões (BREIMAN et al., 1984);
- **Algoritmos baseados em otimização:** realizam predições com base na otimização de alguma função custo, seja o objetivo minimizá-la ou maximizá-la. Neste paradigma, é possível citar os algoritmos de Gradiente Descendente Estocástico (SGD), Regressão Logística (RL), Máquinas de Vetores de Suporte (SVM) e Redes Neurais Multicamadas (MLP); e
- **Algoritmos baseados em comitês (ensembles):** realizam predições com base na combinação e diversificação de uma série de classificadores fracos e na diversificação da amostragens dos dados. Esta última pode ocorrer tanto nas instâncias como nos atributos, de modo que, ao final, haja uma decisão consensual entre os classificadores (comitê). Neste paradigma, é possível citar os algoritmos de *Bagging* (BREIMAN, 1996), *Boosting* e Floresta Aleatória, do inglês *Random Forest* (RF) (BREIMAN, 2001).

2.2.2 Classificação de Dados

Entre as tarefas de aprendizado supervisionado, a tarefa de classificação é a mais comum. Esta ocorre quando existe um conjunto de dados composto por: características descritivas, que descrevem exemplos e instâncias de um problema; e um atributo preditivo, também chamado rótulo ou classe. A classe especifica qual é a categoria de cada um dos exemplos pertencentes ao problema analisado. Assim, o objetivo é criar um modelo de aprendizado que associe as entradas da tabela (características) aos seus rótulos, e seja capaz de generalizar (predizer a classe) para exemplos desconhecidos.

As tarefas de classificação podem ser subdivididas em: classificação binária, quando apenas duas classes distintas separam os dados; ou multi-classe, quando existem mais de duas categorias. Na classificação binária, geralmente as classes possuem comportamento complementar, descrevendo a ausência/presença de uma determinada característica. Já no caso de uma tarefa multi-classe, uma classe não é necessariamente o complemento da outra. Um exemplo do primeiro caso é o filtro de e-mails, podendo rotulá-los como 'spam' ou 'não-spam'. Já no segundo caso, podemos citar a identificação de números escritos à mão, onde existem valores entre 0 e 9, totalizando dez classes possíveis.

De modo geral, a resolução de problemas de classificação passa por duas grandes etapas: a primeira consistindo na **indução** do modelo sobre os dados, a fim de gerar o aprendizado; e a segunda consistindo na **predição** e avaliação do modelo ajustado sobre dados ainda não vistos previamente.

Os modelos de classificação também possuem dois tipos de atributos para aprendizado que mapeiam sua entrada e saída. São eles:

- **Atributos de Entrada ou Hiperparâmetros (HPs):** são as variáveis pertencentes ao próprio algoritmo de aprendizado, cujos valores representam uma configuração específica que o algoritmo deve seguir para a indução do modelo. Estes atributos são opções de entrada, juntamente do conjunto de treinamento, que representam escolhas livres para funcionamento do algoritmo. Bibliotecas e pacotes que implementam algoritmos de AM possuem valores padrão (*default*) como sugestões iniciais para uso. Entretanto, é possível atribuir manualmente valores diferentes para avaliar uma possível melhora de desempenho de um dado modelo. Um exemplo é a quantidade de vizinhos (k) considerada quando realiza-se uma predição com K-NN. Este valor pode ser ajustado e avaliado empiricamente para maximizar o desempenho do algoritmo; e
- **Atributos de Saída ou Parâmetros**: são variáveis que o modelo deve ajustar durante o aprendizado sobre os dados analisados. Servem como saída do processo de indução, e alimentam o processo de predição para novas instâncias não conhecidas. Estas variáveis dependem diretamente da natureza do algoritmo que está sendo utilizado para o processo de aprendizado. Um exemplo é o conjunto de regras de decisão, obtidas pela indução de uma Árvore de Decisão, e utilizada para predição de novos exemplos.

2.2.3 Problemas em Tarefas de Classificação

Durante a etapa de ajuste de modelos podem ocorrer alguns problemas de aprendizado:

- **Desbalanceamento de Classes:** consiste na existência de mais exemplos pertencentes a uma determinada classe, denominada majoritária e, em contrapartida, menos exemplos da(s) outra(s), denominada minoritária(s). Neste cenário, é possível que o modelo tenha uma tendência maior a aprender e rotular um exemplo conforme a classe majoritária, ignorando as demais classes existentes. Ainda assim, o desempenho seria alto, porém não

condizente com a realidade, já que os exemplos das outras classes não seriam corretamente preditos. Como soluções para este problema, é possível citar: a reamostragem dos dados; o uso de métricas balanceadas entre classes; e técnicas que realizam o "balanceamento" das classes artificialmente;

- **Subajuste**, do inglês *Underfitting*: consiste num cenário onde o modelo é muito simples para o aprendizado e, consequentemente, não é capaz de aprender as características mais específicas e estruturais dos dados de treinamento. Como solução para este problema, é possível citar o refinamento da indução do modelo, testando diferentes algoritmos e técnicas no pipeline; e incorporar mais dados e características variadas que possam melhorar a etapa de aprendizado;
- **Sobreajuste**, do inglês *Overfitting*: consiste no fenômeno oposto ao subajuste, em que o modelo é especializado demais e ajustado de maneira exagerada aos dados de treinamento, de modo que não consiga generalizar para dados nunca vistos anteriormente. Como solução para este problema, é possível citar: a simplificação do modelo com uma quantidade menor de atributos analisados; a remoção de possíveis ruídos nos dados aprendidos, como erros ou pontos extremos; ou até mesmo a inclusão de parâmetros de regularização para a indução do modelo.

2.3 Sistemas Fuzzy

Os Sistemas *Fuzzy*, utilizam técnicas e conceitos da Lógica *Fuzzy* (ZADEH, 1965; COPPIN; VALÉRIO, 2015), também conhecida como Lógica Nebulosa ou Difusa. A Lógica *Fuzzy* contrasta com a Lógica Tradicional ou bivalente, a qual permite apenas dois valores lógicos possíveis: verdadeiro ou falso. Nesse sentido, a Lógica *Fuzzy* é uma lógica polivalente, pois a veracidade das proposições é determinada por valores reais no intervalo $[0,1]$, representando a pertinência/pertencimento de um conceito sobre um determinado conjunto. Além disso, existem alguns conceitos importantes a serem apresentados:

- **Variáveis e Valores Linguísticos:** os conceitos do mundo real são representados por estruturas próximas à linguagem humana (substantivos) e possíveis valores que podem ser assumidos, indicando qualidade e estado (adjetivos). Por exemplo, o 'preço' (variável) de um prato num restaurante pode ser 'caro' ou 'barato' (valores);
- **Conjuntos Fuzzy e Funções de Pertinência:** são conjuntos e funções que definem o grau de verdade/pertencimento de um conceito a um conjunto. Por exemplo, existem o conjunto dos preços 'baratos' e o dos 'caros' e uma função que define o "quão" barato e caro é um prato;
- **Operações de Conjuntos Fuzzy:** são operadores que possuem uma correspondência com os utilizados na Teoria Clássica de Conjuntos, porém seu funcionamento é ligeiramente diferente, devido aos graus de pertinência em cada conjunto do universo de possibilidades;

- **Regras Fuzzy:** são regras obtidas a partir de um especialista do domínio do problema, compreendendo uma ou mais proposições condicionais não qualificadas, cujos antecedentes são componentes condicionais, e consequentes, componentes conclusivos. São estruturadas no formato SE-ENTÃO, compostas por um ou mais antecedentes ligados com operadores lógicos '*AND*' ou '*OR*'; um consequente, correspondente a classe de pertencimento; e um operador de implicação; e
- **Inferência Fuzzy:** consiste nas principais etapas do Sistema *Fuzzy*, compostas pelas seguintes: entrada de dados, fuzzyficação (conversão de valores contínuos de intensidade e precisos para graus de pertinência a conjuntos nebulosos); Inferência *Fuzzy* propriamente dita, com base nas Regras *Fuzzy* ou bases de conhecimento; defuzzyficação (processo oposto à fuzzyficação), baseando-se numa determinada medida agregada, gerando um valor preciso; e, finalmente, na saída e interpretação dos dados. É importante observar que o método de inferência mais comum é o de Mamdani (MAMDANI; ASSILIAN, 1975), onde o operador de mínimo representa '*AND*' e implicação, enquanto o operador de máximo representa '*OR*' e composição. A Figura 1 ilustra este processo.

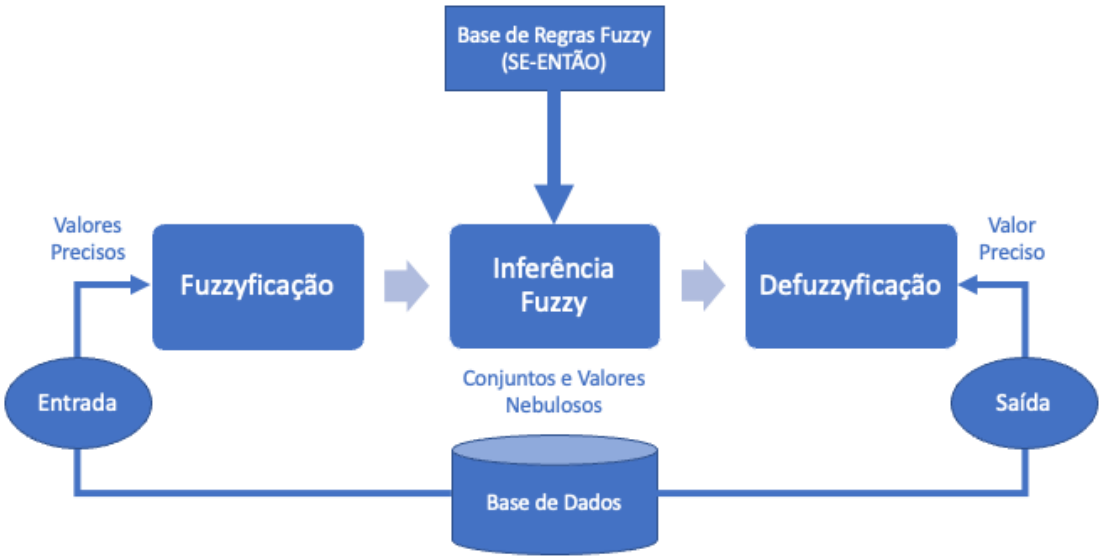


Figura 1 – Funcionamento do processo de Inferência *Fuzzy*, com suas principais etapas e componentes. Fonte: adaptado de Zimmerman (1996).

2.4 Seleção e Avaliação de Modelos Preditivos

O processo de indução e seleção de modelos computacionais para tarefas de classificação exige uma série de pressupostos e conceitos que servem como base para selecionar e avaliar um subconjunto de modelos que fazem parte de um universo de possíveis soluções.

Desta forma, metodicamente, é possível explorar processos e ferramentas estatísticas

para estimar o desempenho de um modelo, tanto durante o processo de ajuste, quanto nas predições (GéRON, 2019), e conduzir uma escolha adequada de modelo preciso.

As próximas subseções descrevem alguns dos principais conceitos envolvidos neste processo.

2.4.1 Técnicas de Amostragem

Durante o processo de análise dos dados para a criação de um modelo preditivo, existem algumas técnicas de amostragem de dados que viabilizam a generalização dos modelos e a possibilidade de medir assertivamente qual é o desempenho efetivo destes modelos no mundo "real". Essas técnicas geralmente são herdadas da Estatística e estão presentes desde a parte de preparação dos dados até a avaliação dos modelos induzidos, de fato. Entre elas:

- **Holdout:** é a técnica mais simples de divisão de dados. Consiste em dividir o conjunto de dados em dois subconjuntos: um conjunto para treinamento e outro para teste. O conjunto de treinamento é utilizado então para induzir o modelo preditivo, fornecendo os rótulos como guia durante o processo de aprendizado supervisionado. Já o conjunto de teste é usado para realizar predições, simulando exemplos do mundo real que seriam desconhecidos ao modelo gerado. Aqui, o modelo usa os rótulos dos exemplos do conjunto de teste como um "gabarito" para verificar e estimar o grau de assertividade do modelo. Ou seja, mensura-se a capacidade de generalização do modelo sobre dados desconhecidos. Em geral, esta divisão é feita empiricamente, de modo que os subconjuntos possuam tamanhos distintos. É intuitivo que existam mais dados para o treinamento do que para o teste, pois disponibilizar mais dados para a indução de um modelo permite a criação de modelos mais precisos. Por isso, frequentemente, a separação ocorre entre 80-20% e 70-30%, ou, até mesmo, 60-40%, representando, respectivamente, os conjuntos de treinamento e teste. A primeira alternativa de divisão é apresentada na Figura 2;

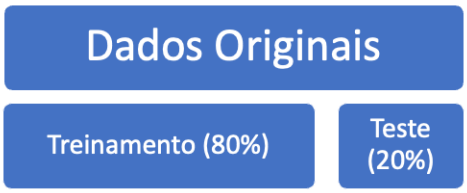


Figura 2 – Exemplo de divisão de dados usando *Holdout*: 80% dos dados originais são alocados no conjunto de treinamento e os 20% restantes no conjunto para testes. Fonte: Autoria própria.

- **Validação Cruzada**, do inglês *Cross Validation* (CV): é uma generalização do processo de *holdout*, dividindo o conjunto de dados em K grupos de tamanho igual. Na CV, são realizadas K iterações, onde $K-1$ grupos são utilizados para treino e ajuste do modelo e o grupo remanescente é utilizado para teste ou validação. A cada iteração, um grupo diferente é utilizado para validação e o restante para treinamento. Ao final, obtém-se K

modelos diferentes sobre as diferentes iterações realizadas e o desempenho do algoritmo é a média das predições realizadas em cada iteração. A Figura 3 ilustra este processo de divisão dos dados. Esta estratégia é uma das mais usadas na literatura para indução e validação de modelos preditivos (MORAIS, 2007; RASCHKA; MIRJALILI, 2017); e

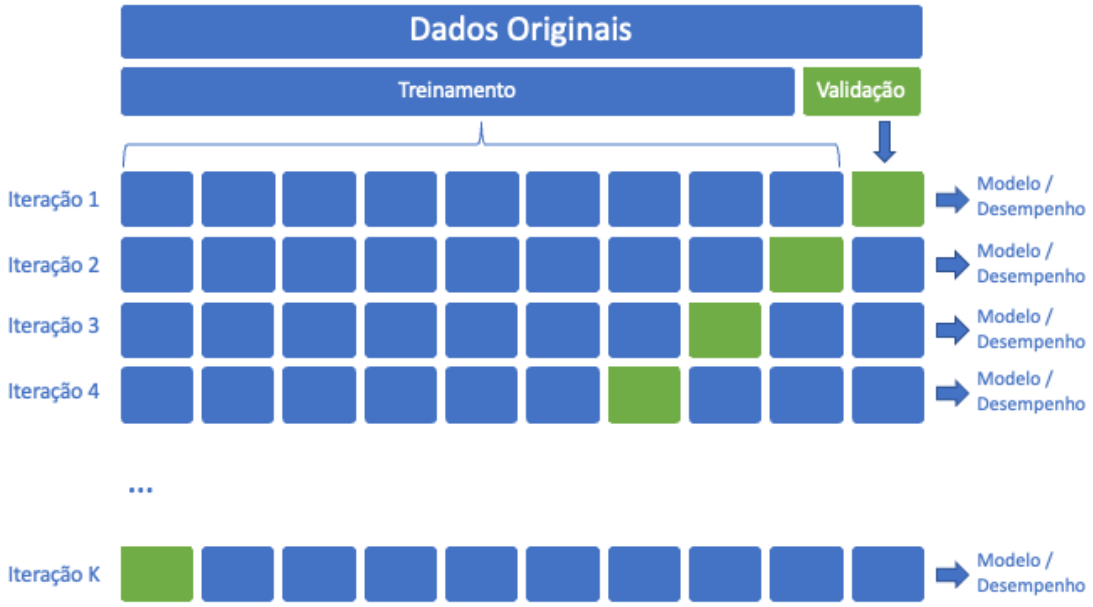


Figura 3 – Amostragem por Validação Cruzada (CV). Fonte: Autoria própria.

- **Validação Cruzada Aninhada**, do inglês *Nested Cross Validation* (Nested-CV): esta é uma estratégia comumente utilizada quando se deseja realizar o ajuste de hiperparâmetros de algoritmos de classificação, onde os dados são divididos em dois níveis diferentes, pois um processo de otimização será conduzido. Logo, existe uma combinação das técnicas de amostragem, de modo que seja gerada uma solução mais robusta e estatisticamente mais confiável. Os dois níveis de amostragens são geralmente descritos como: laço interno e laço externo. O laço interno valida os diferentes hiperparâmetros do algoritmo de classificação, enquanto o laço externo é usado para estimar o desempenho do melhor conjunto de hiperparâmetros encontrados no conjunto de teste. Qualquer uma das abordagens acima citadas pode ser usada tanto no laço interno como externo. Ao utilizar-se a CV em um dos laços, é necessário definir o número de partições. Quanto mais partições, mais preciso é o processo, e quanto menos partições, menos custoso é o processo como um todo. Um exemplo de funcionamento é ilustrado na Figura 4. O laço externo é realizado com *holdout*, enquanto o laço interno realiza CV.

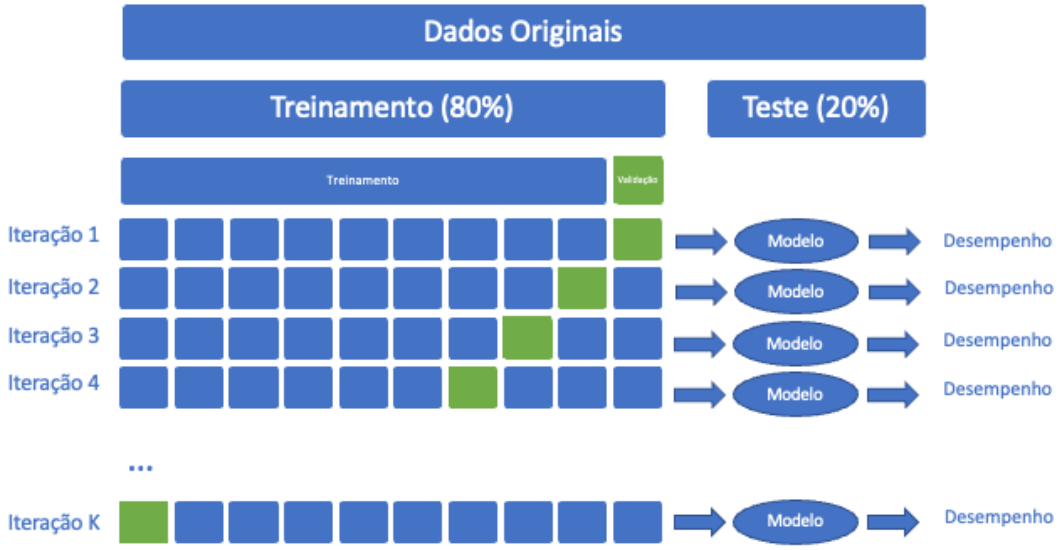


Figura 4 – Validação Cruzada Aninhada combinando *Holdout* (laço externo) com Validação Cruzada (CV) (laço interno). Fonte: Autoria própria.

Além disso, uma alternativa válida e comumente adotada quando o problema de classificação é desbalanceado é a amostragem estratificada por classes. Nesta técnica, os exemplos/instâncias do conjunto de dados são amostrados respeitando a frequência relativa de exemplos por classe, garantindo assim que a mesma proporção das classes ocorra sobre os dados amostrados, e que os resultados sejam condizentes com o conjunto original, mesmo após o processo de amostragem.

2.4.2 Ajuste de Hiperparâmetros

Dentre as etapas existentes em um *pipeline* de solução de classificação de dados, uma que permite uma maior liberdade e potencial de melhoria nos modelos induzidos é o ajuste de hiperparâmetros. Este é um processo automatizado que visa comparar e avaliar diversas combinações de hiperparâmetros dos modelos, de forma automática, com o objetivo de obter um modelo melhor ajustado e com o melhor desempenho possível.

Em geral, este processo é feito em conjunto com a validação cruzada, a fim de avaliar o desempenho do modelo em diversas amostras dos dados, com uma combinação específica de hiperparâmetros a cada avaliação. Como o processo de ajuste de hiperparâmetros é uma tarefa de otimização, em teoria, qualquer técnica de otimização pode ser empregada para sua realização. Na literatura específica (GéRON, 2019; RASCHKA; MIRJALILI, 2017), existem algumas técnicas que são mais frequentemente empregadas:

- **Busca em Grade**, do inglês *Grid Search* (GS): consiste em discretizar os valores de cada hiperparâmetro em conjunto de valores, igualmente espaçados dentro de um intervalo,

e avaliar todas as possíveis combinações desses valores. Dependendo da quantidade de hiperparâmetros do algoritmo, o custo computacional para realizar esse processo cresce exponencialmente em complexidade, trazendo um gargalo para o processamento e comprometendo a eficiência na resposta do modelo melhor ajustado; e

- **Busca Aleatória**, do inglês *Random Search* (RS): consiste na avaliação de um número específico de combinações aleatórias de hiperparâmetros, respeitando o espaço de valores introduzidos pelo usuário. Apesar desta técnica não avaliar todas as combinações possíveis, esta acelera o processo de obtenção do modelo melhor ajustado, permitindo que seja passado um espaço maior de valores pelo usuário e, assim, evitando que o modelo retorne uma combinação melhor local e não global.

A Figura 5 apresenta um comparativo visual destas duas técnicas. A GS (do lado esquerdo) discretiza os possíveis valores dos hiperparâmetros em intervalos igualmente espaçados, o que limita a região de busca aonde a técnica irá atuar. Como uma "grade" será gerada, o método fica suscetível a ficar preso em mínimos/máximos locais. Por outro lado, a RS (do lado direito) aproveita do comportamento aleatório para vasculhar regiões distintas do espaço de hiperparâmetros.

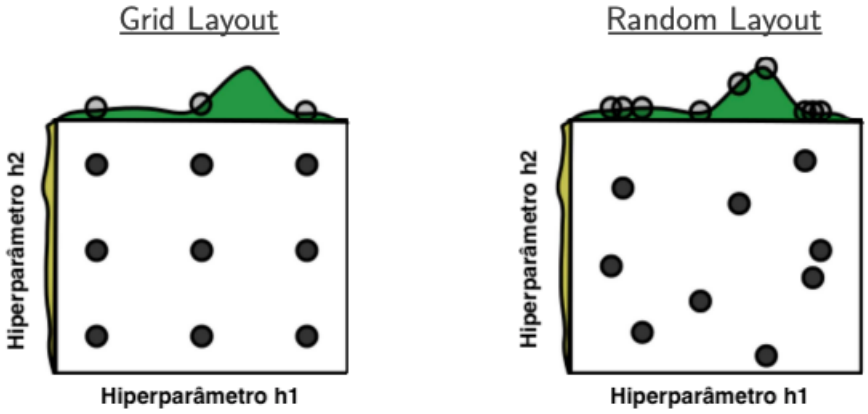


Figura 5 – Comparativo entre GS e RS, mostrando como ambas as técnicas fazem a varredura no espaço de busca. Fonte: Adaptado de Bergstra e Bengio (2012).

2.4.3 Métricas de Classificação

Uma vez induzido o modelo, este precisa ser avaliado para mensurar o quão "bom" pode ser. A literatura específica descreve diversas métricas empregadas na avaliação de modelos de classificação (GéRON, 2019).

Inicialmente, as predições (palpites) dos modelos são obtidas durante o processo de teste dos modelos. Esses palpites são, então, organizados em uma estrutura chamada Matriz de Confusão. Esta matriz compara os valores preditos com os valores reais dos rótulos dos exemplos presentes no conjunto de teste. Além disso, esta apresenta os quantitativos de

exemplos rotulados por classe, onde as linhas representam as classes verdadeiras e as colunas as predições, permitindo que seja avaliado o real versus o predito.

Em um problema binário, especificamente, a matriz de confusão é composta por duas linhas e duas colunas, onde os quantitativos da primeira classe são considerados positivos e os da subsequente negativos, além das predições corretas serem caracterizadas como verdadeiras e as incorretas como falsas. Desta forma, as células da matriz, respectivamente, da esquerda para a direita e de cima pra baixo, correspondem a: verdadeiros positivos (VP), falsos negativos (FN), falsos positivos (FP) e verdadeiros negativos (VN). A Figura 6 apresenta a estrutura desta matriz.

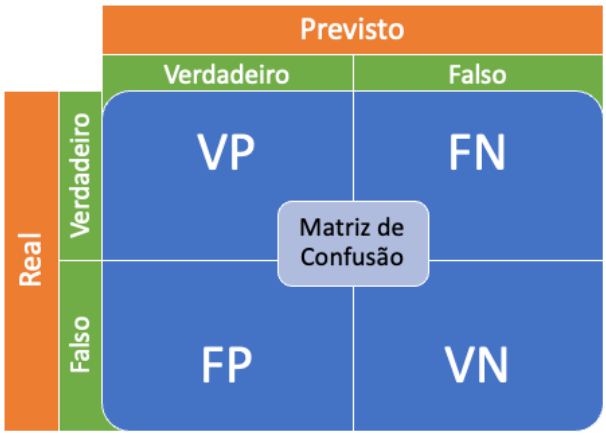


Figura 6 – Estrutura de uma Matriz de Confusão. Fonte: Autoria própria.

A partir da obtenção da matriz de confusão, são derivadas diversas métricas. Segue a descrição das principais:

- **Acurácia:** consiste na porcentagem de acertos totais. É calculada pela quantidade total de acertos (VP + VN) sobre a quantidade total de exemplos existentes. A Equação 1 apresenta a fórmula desta métrica para um problema binário;

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{1}$$

- **Acurácia Balanceada por Classe**, do inglês *Balanced Accuracy per Class* (BAC): consiste na média das porcentagens de acertos totais de cada classe individualmente avaliada. É calculada pelo somatório da quantidade total de acertos em determinada classe sobre todos os exemplos pertencentes àquela mesma classe, para cada classe possível, sobre o número total de classes existentes. É uma medida mais adequada que a Acurácia simples para problemas desbalanceados. A Equação 2 apresenta a fórmula desta métrica para um problema binário;

$$Acurácia\ Balanceada = \frac{1}{2}\left(\frac{VP}{VP+FN} + \frac{VN}{VN+FP}\right) \quad (2)$$

- **Precisão**, do inglês *Precision*: consiste na porcentagem de acertos para uma classe específica, avaliando o quanto o modelo é assertivo. É calculada pela quantidade de predições corretas de uma determinada classe sobre a quantidade de todas as predições realizadas para a mesma classe. Desta forma, é obtido um valor distinto para cada classe do problema. A Equação 3 apresenta a fórmula desta métrica para um problema binário;

$$Precisão = \frac{VP}{VP+FP} \quad (3)$$

- **Revocação ou Sensibilidade**, do inglês *Recall*: consiste na porcentagem de abrangência das identificações corretas de uma determinada classe, avaliando o quanto o modelo capturou de exemplos reais de uma classe específica. É calculada pela quantidade de predições corretas de uma determinada classe sobre a quantidade de todos os exemplos reais da mesma classe. Desta forma, também é obtido um valor distinto para cada classe do problema. A Equação 4 apresenta a fórmula desta métrica para um problema binário;

$$Revocação = \frac{VP}{VP+FN} \quad (4)$$

- **Medida F1**, do inglês *F1 Score*: consiste na média harmônica entre a precisão e revocação de uma classe específica, representando uma medida que é influenciada e depende de ambas as métricas para a composição do seu valor, sendo calculada por duas vezes a precisão vezes a revocação, sobre a soma da precisão e revocação, todas da mesma classe. De forma similar às métricas anteriores, é obtido um valor distinto para cada classe do problema. A Equação 5 apresenta a fórmula desta métrica para um problema binário;

$$F1 = \frac{2}{Prec^{-1} + Rec^{-1}} = 2 \times \frac{Prec \times Rec}{Prec + Rec} \quad (5)$$

2.4.4 *Baselines*

Após induzir um modelo, é interessante verificar se o mesmo aprendeu algo relevante dos dados. Para isso, usamos os chamados *baselines*, modelos simples e ingênuos que servem como comparativo dos modelos induzidos. Alguns exemplos de *baselines* comuns na literatura (MITCHELL, 1997) são:

- **Aleatório**: modelo que realiza predições aleatórias para cada exemplo do conjunto de teste;
- **Majoritário**: modelo que sempre prediz o valor da classe majoritária para cada exemplo; e
- **Minoritário**: modelo que sempre prediz o valor da classe minoritária para cada exemplo.

Um modelo de AM que tenha aprendido com os dados de entrada irá apresentar desempenho superior a todos estes *baselines*. Por outro lado, modelos que não conseguem superá-los são ditos ruins e, consequentemente, descartados, reiniciando o processo de elaboração de uma solução.

2.4.5 Testes Estatísticos

Quando muitos modelos são gerados, é necessário estabelecer algum critério para poder escolher qual é o melhor entre eles. Uma forma sistemática e segura de realizar esse processo é através da avaliação estatística por meio de testes de hipótese. Estes testes avaliam a significância dos resultados obtidos por cada modelo, com o objetivo de verificar se há uma diferença estatística significativa entre diferentes modelos, ou se os desempenhos são similares.

Existem testes paramétricos e não paramétricos. Os primeiros supõem que a distribuição dos resultados obtidos obedecem a uma distribuição normal. Por outro lado, os não paramétricos não assumem essa premissa.

Considerando que é muito difícil ter certeza de como serão os valores de desempenho de um algoritmo de AM quando executado várias vezes, logo, neste caso, é mais comum usar testes estatísticos não paramétricos para realizar a comparação de desempenho de diferentes algoritmos (SANTAFé; INZA; LOZANO, 2015), como, por exemplo, os testes de Wilcoxon e Friedman-Nemenyi.

3 TRABALHOS RELACIONADOS

Este capítulo estabelece uma síntese dos trabalhos que motivaram e apresentaram soluções compatíveis com o problema levantado neste estudo. Como trabalhos afins, foram relacionados dois grupos de soluções possíveis: os primeiros utilizando técnicas convencionais de AM e os demais utilizando Lógica *Fuzzy*.

A Tabela 1 sintetiza todos estes trabalhos encontrados. Para cada trabalho, é apresentado em qual paradigma se encontra (AM ou Fuzzy), quais tarefas foram realizadas, os algoritmos, métricas, conjuntos de dados e melhor desempenho observado nos modelos induzidos. Os detalhes sobre cada trabalho são apresentados a seguir.

3.1 Soluções com Aprendizado de Máquina

Em Mccallum e Nigam (2001), os autores propõem duas técnicas de classificação utilizando modelos probabilísticos bayesianos, estes compostos por soluções envolvendo Naïve Bayes de Bernoulli e Multinomiais. O primeiro modelo verifica a existência de termos em dados textuais, enquanto o segundo calcula a frequência dos termos em cada texto. Ambos os algoritmos foram aplicados em cinco *corpus* distintos para tarefas de classificação de textos. Como resultados, foi identificado que o algoritmo de Bernoulli possui desempenho superior quando os dados possuem um vocabulário reduzido. Já o algoritmo multinomial possui melhor desempenho com dados textuais que apresentam grande vocabulário, minimizando o erro em 27% sobre o obtido a partir de Bernoulli.

Já em Silva (2019), é proposto um classificador de textos referentes a crimes contra a honra, conforme a configuração do tipo de crime. A solução proposta usa Aprendizado Profundo (*Deep Learning* - DL) e PLN. Para tal, os textos do conjunto de dados foram rotulados nas seguintes categorias textuais: ameaça, calúnia, difamação, injúria e injúria racial. Os modelos foram treinados, induzidos e avaliados explorando diferentes algoritmos de DL: Redes Neurais Profundas (*Deep Neural Networks* - DNN); MLPs; Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNN's); e Redes Neurais Recorrentes *Long Short Term Memory* (LSTM) e *Highway Network* (HNET). Os resultados obtidos mostraram melhores desempenhos com DNNs, além de valores superiores das principais métricas de classificação (acurácia, precisão, revocação e F1) na solução final ao adotar-se etapas preliminares de pré-processamento, incluindo vetorização, remoção de palavras duplicadas e stemização.

Em Zanchini (2019), o autor criou um classificador de textos extraídos do Twitter com o intuito de identificar textos de natureza depressiva, configurando um problema binário, representando as classes depressivos e não-depressivos. Para tal, foi utilizado o algoritmo SVM juntamente com uma etapa anterior de pré-processamento dos *tweets*, compreendida pelas

técnicas de uniformização, remoção de ruídos e palavras vazias, lematização[1] e vetorização, todos esses com o propósito de eliminar informações irrelevantes e selecionar as melhores características do conjunto de dados. Como resultados obtidos, houve uma acurácia de 99,7% sobre os dados de treinamento, incluindo a realização de ajuste de hiperparâmetros por GS, além de concluir-se que a inexistência de algumas palavras-chave nos dados de treinamento, como "morte", comprometeram a detecção correta de certos *tweets* depressivos nos dados de teste.

3.2 Soluções com Lógica Fuzzy

Sun et al. (2002) criaram um sistema que utiliza Lógica *Fuzzy* para relacionar palavras de um *corpus* linguístico com textos obtidos a partir de reconhecimento de voz no domínio de viagens aéreas. A Lógica *Fuzzy* é usada para viabilizar a predição de palavras pronunciadas, cujas pronúncias não puderam ser identificadas pelo sistema de reconhecimento de voz padrão e, dessa forma, melhorar o desempenho deste sistema. Para tal, foram utilizadas como bases de dados o *WorldNet* em conjunto com o *corpus* ATIS. Enquanto o primeiro é composto por um conjunto de palavras organizadas em estrutura hierárquica, conforme um grupo semântico, o segundo consiste em gravações de áudio transcritas sobre perguntas feita por pessoas acerca de informações de voos. A obtenção das regras do Sistema *Fuzzy* foi feita por meio de um sistema de aprendizado não supervisionado, agrupando as palavras relevantes ao domínio de estudo conforme um grupo semântico determinado. Os resultados obtidos mostraram que o uso da Lógica *Fuzzy* aplicada ao sistema de reconhecimento de voz realiza uma predição correta de 96% das palavras pronunciadas, representando uma redução de erros de 1/3 do modelo original.

Em Wu et al. (2017), os autores realizaram um estudo que classificou textos sobre o terremoto *Sandy*, ocorrido em 2012, a partir de textos do Twitter. A solução proposta utilizou o conceito de Lógica *Fuzzy*, a fim de definir qual era o grau de relevância dos textos com o evento ocorrido. Os possíveis valores de relevância foram: pouca, moderada, alta ou sem relevância. Para compor as variáveis linguísticas e servirem de base para a construção das Regras *Fuzzy*, foram extraídas sete características sobre cada texto. As características foram as seguintes: a maior pontuação obtida dos termos; o somatório de todas as pontuações dos termos; a quantidade total de termos; o número de termos frequentemente usados; a pontuação média; o peso médio de termos frequentes; e o número total de padrões encontrados. Estas variáveis foram baseadas, principalmente, nas definições empíricas de pontuações de cada texto e nos padrões identificados. Dessa forma, a precisão obtida pela Lógica *Fuzzy* foi comparada com a obtida pelo método tradicional de busca por palavras-chave. Os resultados obtidos mostraram que o método proposto possibilitou a extração de 40% de precisão acima do método de busca

[1] Método no qual ocorre a redução das palavras aos seus radicais, de modo que ainda possa ser preservado o formato de uma palavra válida no idioma de origem.

por palavras-chave, viabilizando este tipo de classificação textual para tratar problemas de classificação com a presença de polarização ou classes binárias.

3.3 Considerações Finais

É importante salientar que os trabalhos relacionados abordam tarefas de classificação de textos de modo geral, não se atendo apenas a textos didáticos, como explora este trabalho. Entretanto, os experimentos realizados poderiam também ser aplicados na resolução deste estudo e, por isso, foram citados como parte dos trabalhos relacionados.

Outro ponto que vale ser observado também é que, apesar de existirem diversos outros trabalhos e estudos que exploram soluções baseadas em ambos paradigmas, foram citadas aqui apenas algumas delas, visto que estes foram utilizados pelo autor, de fato, como motivação para a criação dos experimentos deste trabalho.

Por fim, foi possível perceber que em nenhum dos trabalhos relacionados utilizou-se a acurácia balanceada como métrica de avaliação. Porém, no experimento deste trabalho, foi utilizada esta métrica, devido ao desbalanceamento de classes identificado na etapa de análise exploratória dos dados. Logo, foi julgado a necessidade desta medida como base para a comparação de desempenho dos modelos induzidos.

Tabela 1 – Comparativo de trabalhos relacionados.

Trabalho	Tipo	Tarefa	Algoritmos	Métricas	Datasets	Número de exemplos	Conclusões
Mccallum e Nigam (2001)	AM	Comparativo de funcionamento entre algoritmos baseados em NBs	NB Multinomial NB Bernoulli	AUC Precision, Recall	Yahoo Sceince Industry Sector 71 Newsgroup WebKB4 Reuters	40 6440 20000 4199 12902	Acurácia conforme variação de tamanho de vocabulário
Sun et al. (2002)	Fuzzy	Reconhecimento de voz	Lógica Fuzzy	Acurácia Simples	WorldNet ATIS	1967	96%
Wu et al. (2017)	Fuzzy	Identificação de tweets do terremoto Sandy	Lógica Fuzzy	Acurácia	Twitter	9000000	97%
Silva (2019)	AM	Classificação de crimes contra a honra	DNN MLP LSTM HNET	Acurácia Precision Recall F1	Sentenças de Crimes contra a Honra	328	Melhor modelo (DNN) com 97,24%
Zanchini (2019)	AM	Classificação de Tweets Depressivos	SVM	Acurácia	Twitter	3574	99,7%

4 METODOLOGIA EXPERIMENTAL

Neste capítulo são descritos o contexto e os métodos de coleta de dados, bem como, todas as etapas da elaboração da solução do problema de pesquisa proposto.

4.1 Conjunto de Dados

Os dados utilizados neste trabalho são provenientes de um conjunto de dados (*dataset)* denominado "Córpus de Complexidade Textual para Estágios Escolares do Sistema Educacional Brasileiro" (GAZZOLA; LEAL; ALUISIO, 2019). O conjunto é composto por textos didáticos rotulados de acordo com a complexidade textual, correspondendo a estágios escolares, conforme rege o Sistema Educacional Brasileiro (LDB) (BRASIL, 1996). Este *dataset* é disponibilizado publicamente e pode ser acessado no repositório do Prof. Murilo Gazzola no *GitHub*[1].

Este *dataset* possui quatro classes, que categorizam o tipo de conteúdo do texto de acordo com diferentes níveis de escolaridade: Ensino Fundamental 1, Ensino Fundamental 2, Ensino Médio e Ensino Superior. Efetivamente, todas as classes foram selecionadas para os experimentos, pois elas dispõem de termos em comum entre elas, justificando o uso de Lógica *Fuzzy* para o tratamento das incertezas. Este problema é também um problema desbalanceado, como ilustrado na Figura 7: existem 826 exemplos de textos de Ensino Superior (39,79%), 628 exemplos de textos de Ensino Médio (30,25%), 325 exemplos de textos de Ensino Fundamental 2 (15,66%) e 297 exemplos de textos de Ensino Fundamental 1 (14,31%). Portanto, existe também uma dificuldade inerente ao problema pela quantidade de textos de cada tipo de escolaridade.

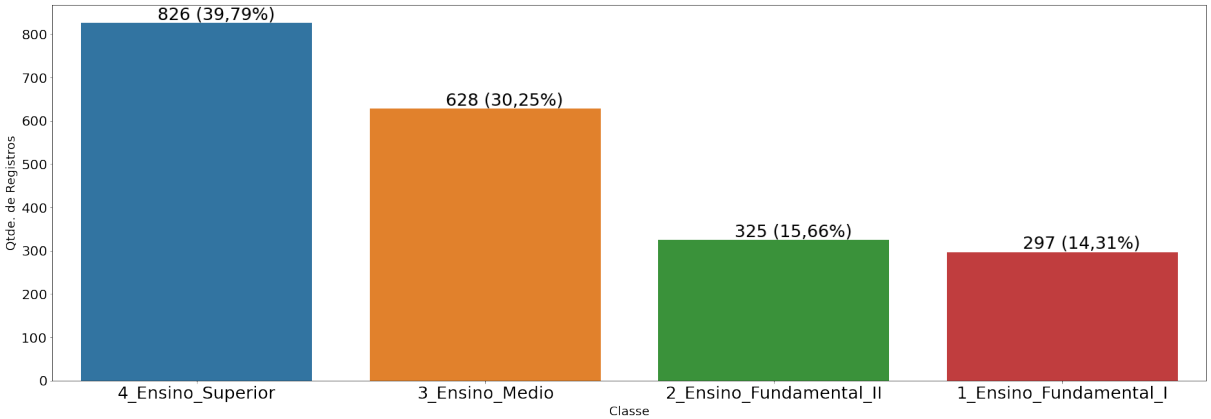


Figura 7 – Distribuição de classes do *dataset* estudado.

[1] Disponibilizado em: <https://github.com/gazzola/corpus_readability_nlp_portuguese>

4.2 Análise Exploratória dos Dados

A primeira etapa realizada foi a coleta dos dados, consistindo na obtenção, abertura, filtragem e persistência dos dados textuais e suas respectivas classes de complexidade textual em estrutura tabular. Em seguida, foi realizada uma breve Análise Exploratória de Dados, do inglês *Exploratory Data Analysis* (EDA), com o objetivo de apresentar estatísticas relevantes para o problema e obtenção de nuvens de palavras (*wordclouds*) por classe, identificando assim as palavras mais frequentes em cada estágio escolar. Esta última identificação foi importante para a criação de um dicionário de palavras adicionais, as quais também foram caracterizadas como palavras vazias, visto o contexto do problema tratado neste trabalho.

4.3 Algoritmos de AM

Nos experimentos de classificação textual foram testados diferentes algoritmos de AM: k-Vizinhos Mais Próximos (K-NN), Regressão Logística (RL), Gradiente Descendente Estocástico (SGD), Naïve Bayes (NB) Bernoulli, Naïve Bayes (NB) Multinomial, Máquinas de Vetores Suporte (SVM), Árvore de Decisão (DT), Floresta Aleatória (RF) e Perceptron Multicamadas (MLP). A escolha destes algoritmos deu-se por serem os algoritmos explorados pelos trabalhos relacionados (Capítulo 3), e por terem diferentes viéses indutivos, ou seja, possuem diferentes formas de aprender com os dados (MITCHELL, 1997).

O algoritmo de Árvore de Decisão, especificamente, além de ter sido explorado como uma solução tradicional de AM, também foi usado para extrair as Regras *Fuzzy* da segunda solução, a partir das regras induzidas pelo algoritmo em forma de árvore. A escolha pelo uso de DT justifica-se pelo fato deste algoritmo ter uma abordagem mais simples e direta para a obtenção de regras lógicas, embora nos trabalhos mencionados anteriormente no Capítulo 3 tenham sido exploradas outras técnicas de obtenção destas regras, seja de forma automatizada ou manual.

4.4 Soluções Implementadas

Ao longo dos experimentos foram desenvolvidas duas soluções para o problema de classificação de textos escolares: i) uma baseada apenas em algoritmos tradicionais de AM; e ii) uma abordagem híbrida explorando algoritmos de AM e Lógica *Fuzzy*. Ambas as soluções compartilham a mesma sequência de etapas (*pipeline*), diferindo apenas em algumas particularidades. O *pipeline* adotado contém as seguintes etapas: análise exploratória de dados (EDA); pré-processamento; seleção e indução de modelos, incluindo o ajuste de hiperparâmetros; validação dos modelos induzidos e avaliação dos modelos. Na solução híbrida há também uma etapa para geração dos conjuntos e Regras *Fuzzy*, que fica imediatamente antes da etapa de indução dos modelos.

Uma vez executado todo o *pipeline*, os resultados foram então avaliados e mensurados

em termos da Acurácia Balanceada entre Classes (BAC), além de um relatório completo mostrando outras métricas de classificação, obtidas por meio da matriz de confusão dos modelos gerados: acurácia simples, precisão, revocação e medida F1. Ademais, outra forma de mensuração adotada neste trabalho foi a comparação dos resultados dos modelos obtidos com três *baselines*: o primeiro composto por predições aleatórias simples; o segundo pela predição da classe majoritária; e o terceiro pela predição da classe minoritária. O diagrama com o universo de soluções avaliadas pode ser visualizado na Figura 8. As próximas subseções descrevem estas soluções com mais detalhes.

Figura 8 – Universo de soluções avaliadas neste trabalho. Fonte: Autoria própria.

4.4.1 Solução baseada em Aprendizado de Máquina

A Figura 9 apresenta um resumo do processo utilizado para o desenvolvimento da solução baseada em AM. Para esta solução, foi desenvolvido um *pipeline* convencional à tarefas de classificação (RASCHKA; MIRJALILI, 2017), compreendendo as seguintes etapas:

1. **Pré-Processamento:** consistiu nas etapas de uniformização textual; descarte de ruídos e palavras vazias; stemização; vetorização; obtenção de pesos baseados em TF-IDF; e conversão em matrizes numéricas densas[2];
2. **Seleção de Modelos:** os modelos foram induzidos e avaliados por meio de uma estratégia de validação cruzada aninhada, combinando amostragem por *holdout* com CV. O laço externo divide os dados usando *holdout*, com 80% dos dados selecionados para treinamento dos modelos e 20% para teste. A CV é usada em um laço interno, para treinamento e ajuste de hiperparâmetros dos algoritmos de classificação. Além disso, a CV é executada com 10 partições, as quais foram usadas para indução dos modelos pelos algoritmos descritos na seção anterior; e
3. **Ajuste de Hiperparâmetros** (HPs): os hiperparâmetros dos principais algoritmos de classificação e das funções de pré-processamento também foram ajustados internamente,

[2] Por padrão, a biblioteca *Numpy* do *Python* condensa matrizes com grandes quantidades de zeros para economizar espaço de memória. Esta conversão visa expandir a matriz para viabilizar as operações posteriores.

incluindo a seleção das melhores características com base em testes estatísticos. O processo foi realizado por meio da RS com um *budget* de 100 avaliações por partição da validação cruzada interna. É importante observar que foi utilizada a BAC para avaliação dos melhores classificadores e hiperparâmetros, devido ao desbalanceamento das classes.

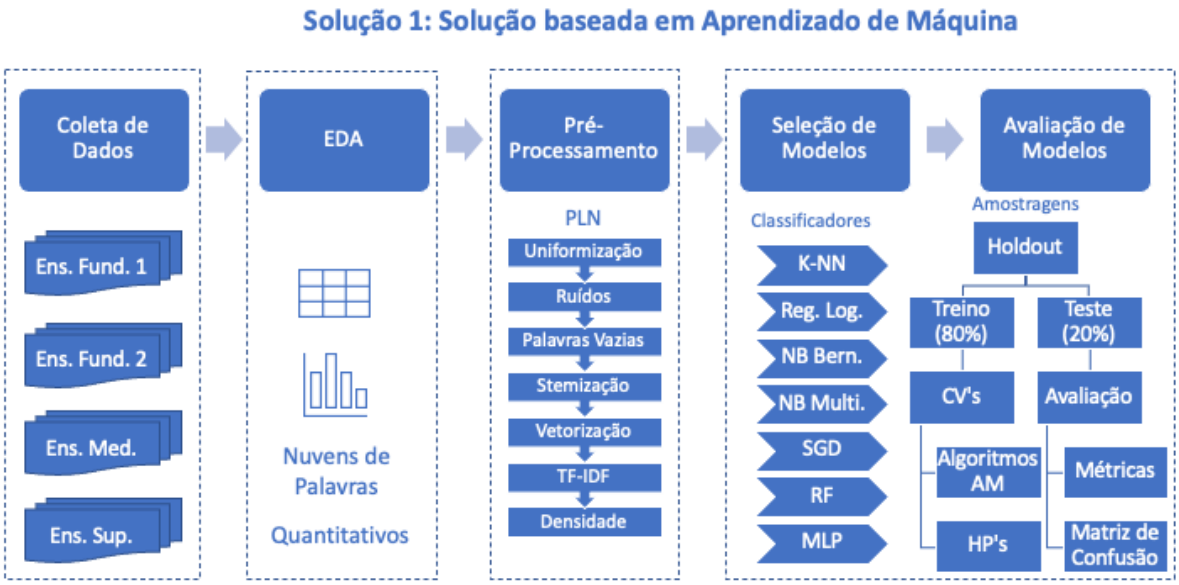


Figura 9 – Diagrama da solução baseada em AM. Fonte: Autoria própria.

4.4.2 Solução baseada em Lógica *Fuzzy*

A Figura 10 apresenta um resumo do processo utilizado para o desenvolvimento da solução baseada em Lógica *Fuzzy*. Para esta solução, o *pipeline* foi adaptado para suportar o processamento dos conceitos *Fuzzy*, compreendendo as seguintes etapas:

1. **Pré-Processamento:** consistiu nas etapas de uniformização textual; descarte de ruídos e palavras vazias; stemização; vetorização; seleção das 15 melhores características (termos), com base no teste estatístico do qui-quadrado; obtenção de pesos baseados em TF-IDF; e conversão em matrizes numéricas densas;
2. **Obtenção das Regras Fuzzy:** esta foi uma etapa de simulação do levantamento das regras a partir do contato com um especialista no assunto. O processo como um todo envolveu a aplicação de funções de transformação sobre os dados textuais, incluindo a discretização dos valores contínuos no intervalo entre [0,1] em dois grupos linguísticos distintos e equivalentes em termos de tamanho, identificados como: 'baixo' (0) e 'alto' (1). Posteriormente, induziu-se um modelo de DT, limitada a 6 níveis de profundidade máxima; e deste modelo (árvore), foram geradas regras de forma automática. Vale ressaltar que tanto a execução da etapa de pré-processamento quanto indução da DT ocorreram sobre o conjunto de dados de treinamento; e

3. **Processamento Fuzzy:** consistiu no processo de Inferência *Fuzzy* completo, com base nas Regras *Fuzzy* mapeadas na fase anterior, bem como nas variáveis linguísticas e suas funções de pertinência e operadores utilizados. Para a criação de variáveis linguísticas dos antecedentes foram utilizados os termos (características) selecionados dos textos. Para cada termo, seus valores foram discretizados em valores linguísticos 'baixo' e 'alto'. Já na criação da variável linguística do consequente, foi utilizada a variável da classe, com os estágios escolares existentes. Para as funções de pertinência, foram experimentadas empiricamente o tipo Gaussiana para os antecedentes e triangular para o consequente, muito comumente utilizadas na literatura (ZIMMERMAN, 1996). Como definição de operador, foi utilizado o mesmo proposto por *Mamdani* (padrão da biblioteca *Scikit-fuzzy*), também de forma empírica. E, finalmente, como parte integrante do processo de *Inferência Fuzzy*, destacam-se as mesmas etapas citadas na fundamentação teórica do Capítulo 2.

É importante observar que, como parte da automatização da geração das Regras *Fuzzy*, foram criadas duas funções customizadas: a primeira com o objetivo de apresentar as regras obtidas a partir do ajuste da árvore de decisão no formato de *if-else*, com sintaxe similar à de uma linguagem de programação; e a segunda a fim de extrair todos os caminhos de decisão possíveis em forma de lista, introduzindo dinamicamente cada elemento no código de criação das Regras *Fuzzy*. Estas customizações foram necessárias, visto que não foi possível encontrar soluções similares na literatura que propusessem qualquer tipo de conversão automática da biblioteca de AM (*Scikit-learn*) para a de Lógica *Fuzzy* (*Scikit-fuzzy*).

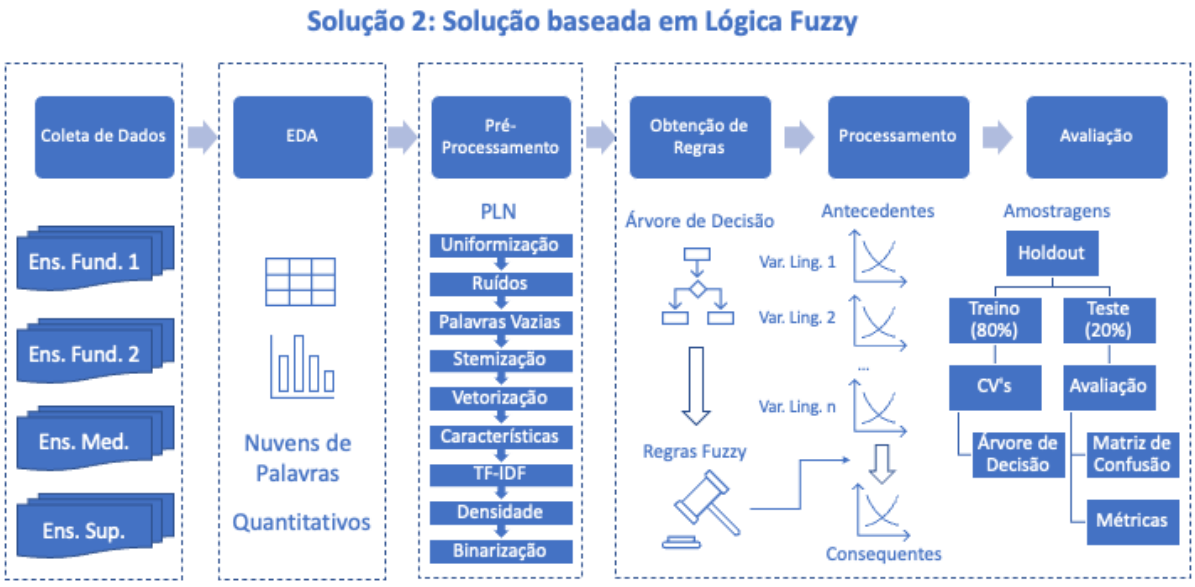


Figura 10 – Diagrama da solução baseada em *Fuzzy*. Fonte: Autoria própria.

4.4.3 Validação Estatística

Para consolidação dos resultados obtidos em ambas as soluções, foi realizada uma etapa de validação estatística e comparação dos modelos induzidos. Todos os modelos tiveram suas acurácias parciais comparadas, dois a dois, por meio do teste não paramétrico de Wilcoxon, com $\alpha = 0.05$. Este teste permite avaliar uma possível equivalência de distribuições de performances e, assim indicar diferenças estatísticas no desempenho de cada par de classificadores.

4.4.4 Reprodutibilidade dos Experimentos

Para a codificação dos experimentos, foi utilizada a linguagem *Python*, na versão 3.7.1. Foi usado também o pacote de bibliotecas do *Anaconda*, versão 4.10.1, compreendendo bibliotecas de Ciência de Dados e Lógica *Fuzzy*, destacando-se as principais extensões: *Numpy* (HARRIS et al., 2020), *Pandas* (MCKINNEY, 2010), *Matplotlib* (HUNTER, 2007), *Seaborn* (WASKOM, 2021), *Graphviz* (ELLSON et al., 2001), *Scikit-learn* (BUITINCK et al., 2013), *NLTK* (LOPER; BIRD, 2002) e *Scikit-fuzzy* (WARNER, 2012). Para reprodutibilidade dos experimentos, foi adotado um valor de semente aleatória (seed) igual a 42. O código completo desenvolvido pode ser acessado no link: <https://github.com/nelsondressler/ecd_utfpr_dv_tcc/blob/main/TCC_Text_Classification_hcv.ipynb>.

5 RESULTADOS

Este capítulo apresenta e discute os resultados experimentais do trabalho, destacando os pontos fortes e fracos das soluções utilizadas, visando a classificação de textos escolares, conforme a sua complexidade.

5.1 Pré-Processamento Textual

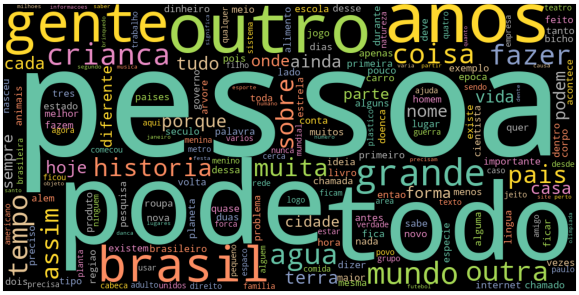
Durante as etapas de EDA e pré-processamento, mais especificamente após a limpeza inicial dos dados (uniformização, descarte de ruídos e de palavras vazias padrão[1]), foram geradas inicialmente quatro nuvens de palavras, uma para cada classe, que corresponde a um grau de escolaridade. A Figura 11 mostra as visualizações geradas.

Porém, a partir da análise das figuras geradas, foram identificados termos adicionais além das *stopwords* que também possuem uma frequência maior nos textos, visto o uso correto da língua, porém, claramente, não sinalizam informações relevantes ao domínio do problema, de modo que possam ser usados como parte integrante do modelo de classificação para serem generalizados sobre dados desconhecidos ou futuros. Por exemplo, os termos 'pode', 'voce' ou 'sim'. Assim, foi criada uma segunda etapa de limpeza, removendo também estes "termos vazios". A listagem completa dos termos removidos nesta etapa é descrita no Apêndice B. Após a remoção das palavras vazias adicionais, foram geradas novas nuvens de palavras, ilustradas na Figura 12.

Comparando as Figuras 11 e 12, é possível notar o efeito da remoção das palavras vazias filtradas. Outra característica interessante é que alguns termos possuem maior frequência em uma classe textual específica. Por exemplo: as palavras 'brasil', 'agua', 'crianca' e 'historia' aparecem com maior destaque em textos de conteúdo do Ensino Fundamental 1. Já nos textos do Ensino Fundamental 2, as palavras que se destacam são 'problema', 'vida', 'trabalho' e 'social'. Textos do Ensino Médio já ressaltam termos como 'celula', 'estado' e 'cidade'; enquanto conteúdos do Ensino Superior destacam palavras como 'sistema', 'conjunto', 'dados' e 'funcao'. Além disso, alguns termos apresentaram relevância similar em mais de uma classe: a palavra 'anos' aparece igualmente nas classes Ensino Fundamental 1 e 2, enquanto as palavras 'formula' e 'numero' se repetem nas classes Ensino Médio e Superior. Esta equivalência identificada, inclusive, favorece o uso da Lógica *Fuzzy*, representando graus de incerteza no pertencimento de certos termos em cada classe. Outro ponto que é importante salientar também é que os termos apresentados em ambos os grupos de nuvens de palavras aparecem em letras minúsculas e sem acentos, graças ao processo anterior de uniformização dos mesmos.

Por fim, as Figuras 13 e 14 apresentam histogramas que representam a distribuição da

[1] Relacionadas pela biblioteca de PLN do *Python* NLTK



(a) Ensino Fundamental 1 (b) Ensino Fundamental 2

(c) Ensino Médio (d) Ensino Superior

Figura 11 – Nuvens de palavra das classes do *dataset* sem palavras vazias padrão. Fonte: Autoria própria.

(a) Ensino Fundamental 1 (b) Ensino Fundamental 2

(c) Ensino Médio (d) Ensino Superior

Figura 12 – Nuvens de palavra das classes do *dataset* sem palavras vazias padrão e demais palavras irrelevantes. Fonte: Autoria própria.

quantidade de palavras por documento para cada classe do problema, antes e após o descarte de palavras vazias, inclusive as adicionais.

A partir das figuras é possível afirmar que a remoção de todas as palavras vazias mencionadas no Apêndice B, além de ter reduzido a quantidade de palavras por documento,

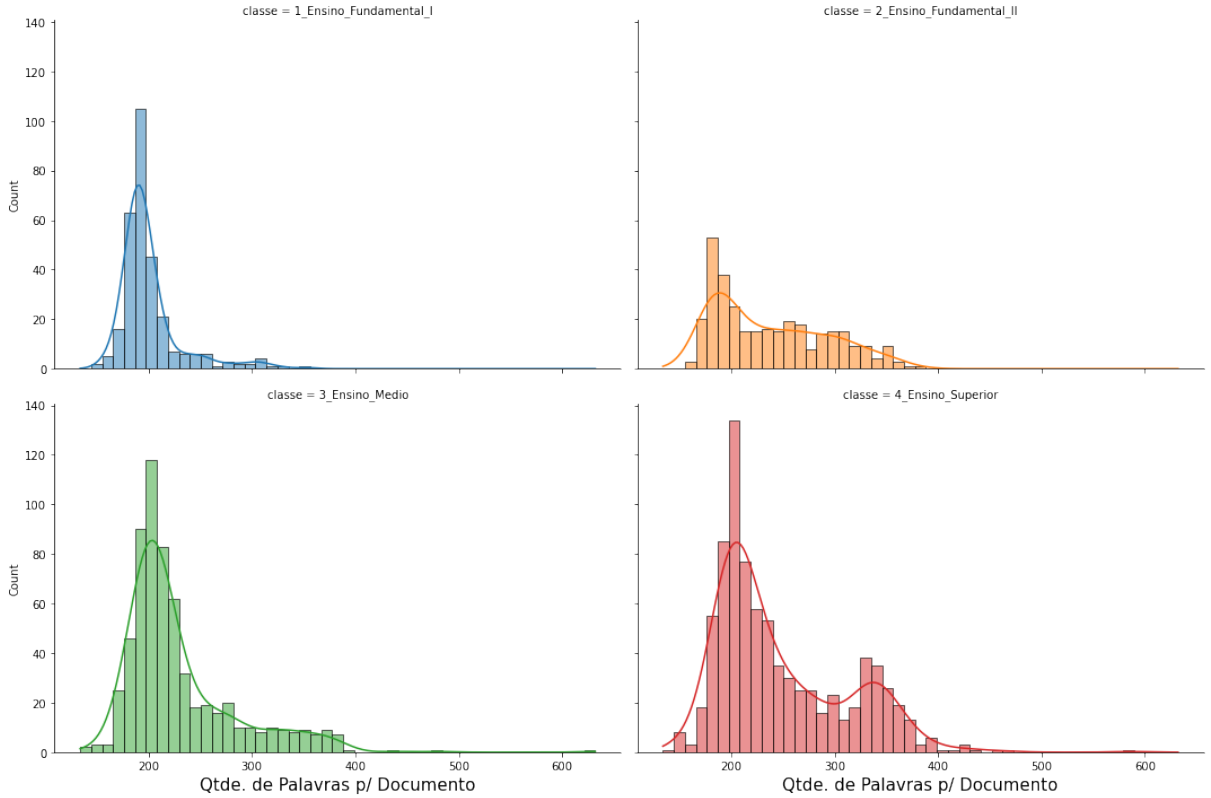


Figura 13 – Distribuição de palavras por documento em cada classe original. Fonte: Autoria própria.

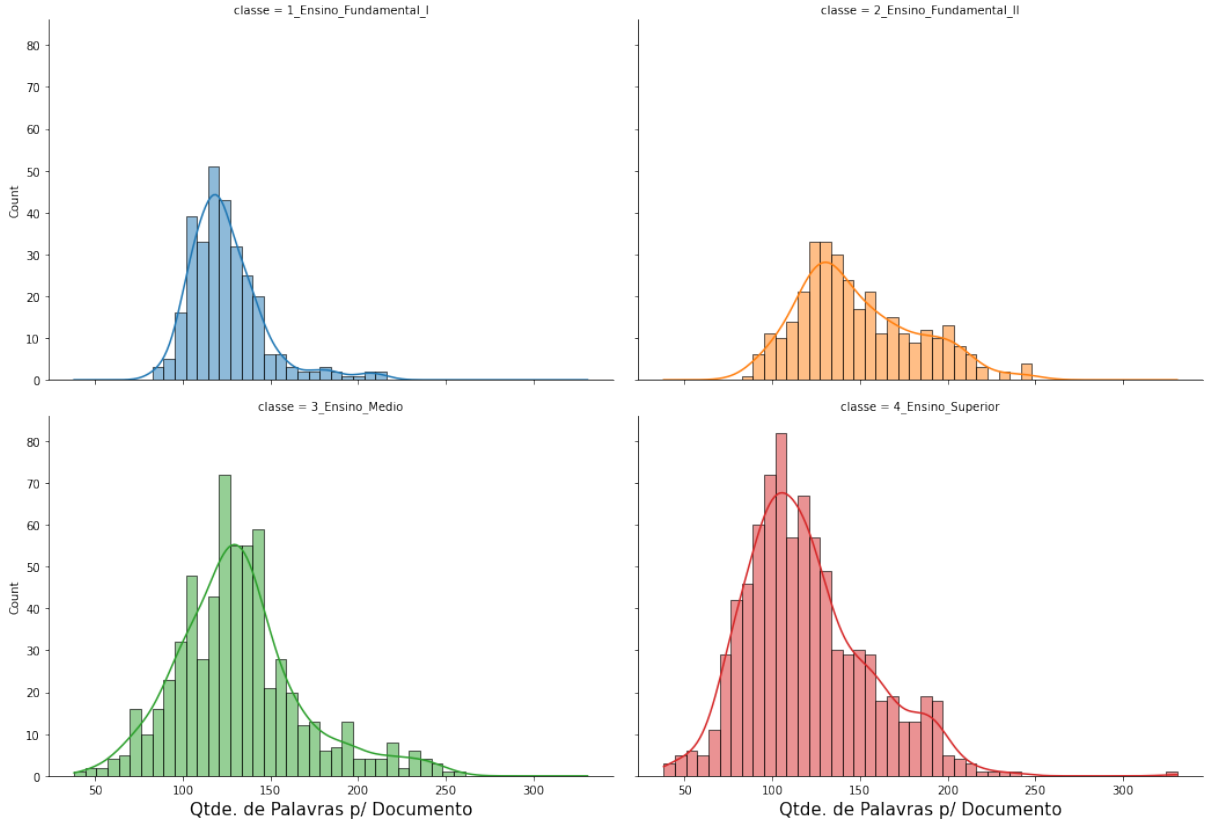


Figura 14 – Distribuição de palavras por documento em cada classe processado. Fonte: Autoria própria.

também deixou a distribuição mais simétrica para cada classe, auxiliando no processo de generalização dos modelos e, ao mesmo tempo, tornando os dados menos sensíveis à presença de palavras irrelevantes.

5.2 Desempenho Geral dos Classificadores

Após a avaliação dos algoritmos de classificação selecionados para este experimento, compostos pelos algoritmos de AM (solução 1), Sistema *Fuzzy* (solução 2) e *baselines*, foi gerado um gráfico de violino com o desempenho geral, representando a distribuição do processo de classificação textual aplicado sobre cada partição dos conjuntos decorrentes da CV sobre os dados de treinamento. O mesmo é exibido na Figura 15.

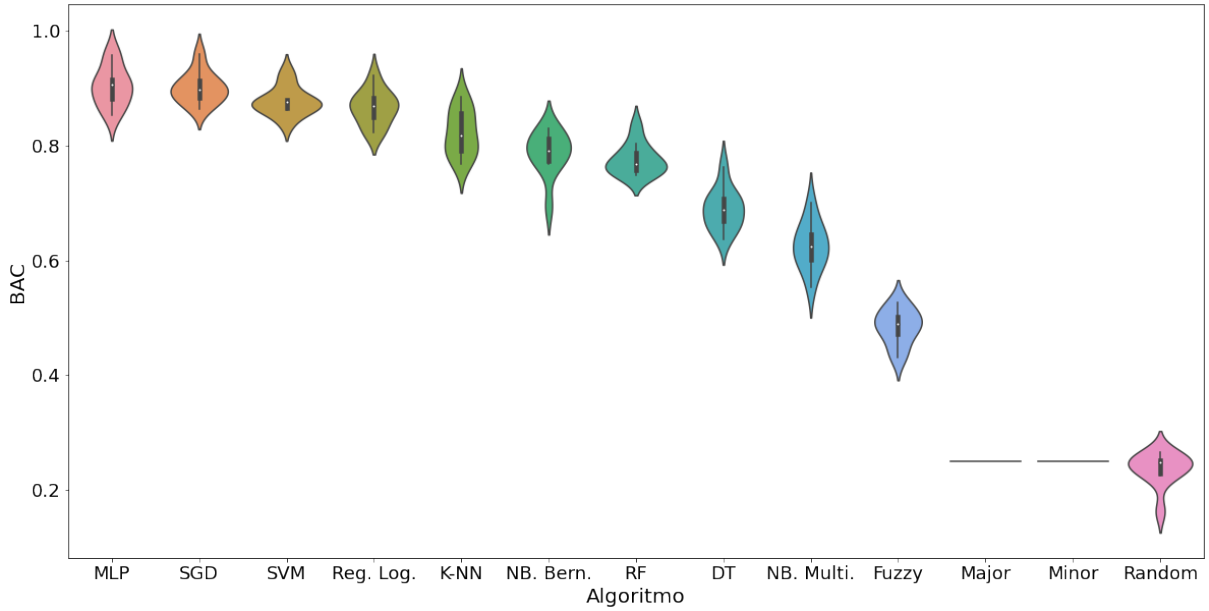


Figura 15 – Desempenho dos classificadores em termos da BAC no conjunto de treinamento via CV. Fonte: Autoria própria.

Com base na figura, é possível perceber o comportamento de uma distribuição mais homogênea em todos os classificadores. Além disso, é possível perceber que os algoritmos MLP e SGD, especificamente, obtiveram um desempenho melhor do que os demais.

Além do gráfico de distribuição por violino, os resultados médios de BAC são reportados na Tabela 2. Para cada algoritmo é apresentada a BAC média obtida no conjunto de treinamento e no conjunto de teste. Além disso, os algoritmos estão ordenados do melhor para o pior de acordo com o valor de BAC obtido no conjunto de treinamento, seguindo a mesma ordem dos classificadores no gráfico de violino.

Tanto a figura quanto a tabela acima permitem evidenciar algumas informações adicionais sobre os classificadores obtidos. Em primeiro lugar, há alguns modelos que obtiveram uma variância maior e, ao mesmo tempo, uma concentração menor dos resultados em torno de sua média, demonstrando que dependendo dos dados amostrados, estes classificadores podem

Tabela 2 – Desempenho dos classificadores em termos da BAC nos conjuntos de treino e teste. Os melhores resultados obtidos em cada partição estão destacados em negrito.

		BAC	
Grupo	**Algoritmo**	**Treinamento**	**Teste**
A	MLP	**0.9034**	0.8804
	SGD	0.9006	0.**8876**
B	SVM	0.8775	0.8623
	RL	0.8661	0.8381
	K-NN	0.8221	0.8039
C	NB-Bern.	0.7868	0.7565
	RF	0.7754	0.7865
	DT	0.6891	0.6503
	NB-Multi.	0.6252	0.6321
	Fuzzy	0.4844	0.4642
D	Major.	0.2500	0.2500
	Minor.	0.2500	0.2500
	Random	0.2367	0.2636

obter desempenhos melhores ou, pelo menos, mais próximos dos que obtiveram uma BAC média maior. Isto é possível de visualizar claramente no caso do K-NN, que obteve cerca de 88,4% em uma das partições do conjunto de treinamento, mantendo-se acima da BAC média da RL, com apenas, aproximadamente, 86,6%. Ademais, outros classificadores obtiveram uma variância menor e, ao mesmo tempo, mantendo uma concentração maior dos desempenhos, demonstrando que, mesmo nas partições do conjunto de treinamento com os piores resultados, não decaíram tanto da sua BAC média. Este é o caso do SVM e RF.

Analisando as médias dos resultados obtidos em cada partição dos classificadores e mensurados pela BAC, pode-se dividir os classificadores em quatro grupos distintos:

- **Grupo A:** composto pelos algoritmos que obtiveram uma média de BAC acima de 90% no conjunto de treinamento, sendo eles: MLP (0.9034) e SGD (0.9006);
- **Grupo B:** composto pelos algoritmos que obtiveram uma média de BAC no conjunto de treinamento entre 80 e 90%, sendo eles: SVM (0.8775), RL (0.8661) e K-NN (0.8221);
- **Grupo C:** composto pelos algoritmos que obtiveram uma média de BAC no conjunto de treinamento entre 40 e 80%, sendo eles: NB Bernoulli (0.7868), RF (0.7755), DT (0.6892), NB multinomial (0.6252) e *Fuzzy* (0.4844). Apesar deste grupo compreender um intervalo maior se comparado aos primeiros dois, é possível afirmar que, ao menos, houve algum aprendizado a partir dos dados; e
- **Grupo D:** composto pelos algoritmos que obtiveram uma média de BAC abaixo de 40%, sendo os *baselines* de classes Majoritária e Minoritária (0.25), e aleatório simples (0.2367). Logicamente, a proposta destes *baselines* é não adquirir qualquer aprendizado e, por isso, ficam nesta faixa.

Ainda sobre as BACs médias, ao analisar o ranqueamento dos classificadores, é possível inferir um relacionamento da estrutura dos dados e da natureza do problema de classificação com os viéses indutivos dos modelos que obtiveram desempenhos bons ou ruins. Seguem as considerações de conjunto de classificadores:

- Primeiramente, é possível perceber um paralelo entre ambos os modelos MLP e SGD, os dois melhores modelos em desempenho, pois possuem um viés de otimização que utiliza a função de custo de gradiente descendente estocástico, mesmo no caso do MLP, com o objetivo de realizar a retro-propagação dos sinais e gerar o aprendizado no modelo;
- Além disso, os quatro melhores modelos em desempenho, MLP, SGD, SVM e RL são algoritmos que manipulam apenas informações numéricas, o que é justificável, graças a etapa anterior de pré-processamento, onde são obtidos valores numéricos de pesos para cada termo nos documentos do *dataset*;
- Ainda analisando o desempenho da RL, é possível sugerir e propor que os dados são linearmente separáveis;
- Em contrapartida, observando os resultados obtidos pela DT e RF, é possível perceber que os dados não se comportam bem em algoritmos que realizam cortes ortogonais nos dados, mesmo num cenário onde evitam o sobreajuste aos dados com a utilização de comitês; e
- Por fim, considerando o desempenho obtido com a solução *Fuzzy*, os resultados, apesar de ficarem abaixo em comparação com os baseados em AM, ainda se mantiveram acima dos *baselines* selecionados para este experimento. Ou seja, apesar de baixo, o modelo *Fuzzy* está aprendendo com os dados, bem como apresenta um componente de interpretabilidade, algo que modelos induzidos por AM nem sempre disponibilizam. Nesse sentido, a solução *Fuzzy* pode nos fornecer regras para entender como o processo de classificação é realizado.

Quanto a Tabela 2, especificamente, é possível observar que os dados também foram agrupados conforme os grupos elencados anteriormente. Adicionalmente, identificamos que os valores de desempenho dos classificadores no conjunto de treinamento e teste, de modo geral, ficaram próximos. Dessa forma, é possível afirmar que os algoritmos foram capazes de aprender com a etapa inicial de ajuste de modelos e extrair as principais características dos dados de cada classe. Isto ocorreu também graças ao pré-processamento aplicado anteriormente, com a remoção de palavras vazias, inclusive as adicionais irrelevantes, que tornaram os dados textuais mais homogêneos e com menos ruídos.
Os resultados completos para todas as partições de dados de treinamento podem ser encontrados no Apêndice C.

Dados esses desempenhos, os modelos induzidos também foram comparados estatisticamente para verificar a significância dos resultados obtidos. Optou-se pelo teste estatístico não paramétrico de Wilcoxon, comparando as medidas de BAC obtidas no treinamento dos modelos, dois a dois. O teste foi conduzido com um nível de significância de 95%, com $\alpha = 0.05$. A

Figura 16 apresenta estes resultados, destacando os pares sem diferenças significativas, com P-Valores maiores que 0.05, na cor clara, e os demais na cor escura.

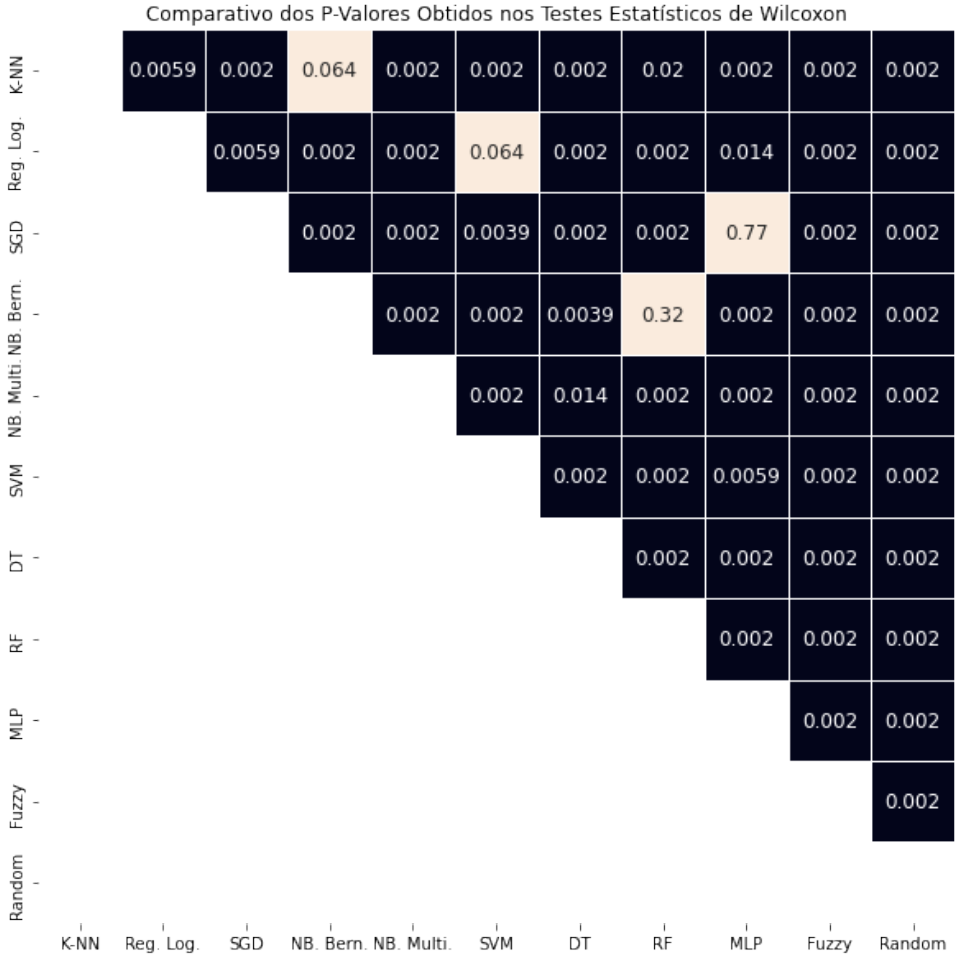


Figura 16 – Comparativo do teste de Wilcoxon realizado sobre os classificadores. Fonte: Autoria própria.

Os resultados dos testes estatísticos mostram que apenas alguns pares de modelos não apresentam diferenças significativas (células amarelas). São exemplos de modelos "equivalentes": MLP e SGD, SVM e RL, K-NN e NB Bernoulli; e NB Bernoulli e RF. Com base nos modelos equivalentes identificados, é possível observar que estes pares de modelos correspondem às proximidades diretas nas colocações de BAC média, apresentadas na Figura 15 e Tabela 2.

Analisando apenas os *baselines*, é possível observar que as classes majoritária e minoritária não foram relacionadas no gráfico, pelo fato de suas variâncias e desvios padrão serem iguais a zero, logo foram ignorados e suprimidos do gráfico. Já o *baseline* aleatório apresentou diferenças significativas com todos os demais modelos. Isto demonstra que, mesmo que a solução *Fuzzy* tenha obtido um resultado abaixo dos classificadores de AM, ainda se manteve acima e afastada estatisticamente das predições aleatórias.

5.3 Ajuste de Hiperparâmetros

Após a obtenção dos desempenhos gerais na etapa anterior, foram selecionados empiricamente os principais algoritmos de AM para compor a etapa de ajuste de hiperparâmetros, bem como as etapas posteriores de avaliação dos modelos finais ajustados.

Nesta escolha, além de ser levado em conta os melhores desempenhos nos conjuntos de treino e teste, como é o caso dos algoritmos de MLP e SGD, ainda foram considerados algoritmos que oferecem uma complexidade menor e funcionamento simples, ainda mantendo bons desempenhos gerais, como, por exemplo, o K-NN. Este último, inclusive, é considerado o mais simples e menos complexo em relação aos demais algoritmos identificados anteriormente no Grupo B. Logo, estes três algoritmos foram utilizados para as etapas posteriores deste experimento.

O ajuste de HPs, especificamente, foi aplicado com o intuito de explorar e avaliar combinações distintas de HPs, obtendo assim modelos mais refinados e com resultados de BAC ainda melhores. Dessa forma, foram selecionados empiricamente os HPs mais comuns na literatura (GéRON, 2019; BUITINCK et al., 2013) e comumente utilizados pelos algoritmos mencionados.

Além disso, esta etapa de ajuste também foi aplicada sobre a etapa anterior de pré-processamento para determinadas técnicas utilizadas, a fim de refina-las e aperfeiçoa-las ainda mais. Para tal, foram ajustadas conjuntamente as seguintes etapas: vetorização, TF-IDF e seleção de características. De modo geral, este refinamento se deu tanto na definição da quantidade e forma de avaliar as melhores características do *dataset* para o processo de criação dos vetores a serem utilizados, quanto nos cálculos das pontuações e pesos sobre os termos em cada documento.

Outro ponto a se observar é que a solução com Lógica *Fuzzy* não foi submetida a esta etapa de ajuste, devido a este tipo de prática ser mais comum em cenários onde são utilizados algoritmos de AM para classificação.

Segue a relação dos HPs utilizados juntamente com suas faixas de valor exploradas, para cada algoritmo, inclusive para as técnicas de PLN avaliadas nesta etapa:

- PLN:
  - Vetorização: quantidade máxima e mínima de n-gramas entre (1,1) e (1,2); frequência mínima de palavras entre 0 e 9; e quantidade de maiores frequências entre 0 e 99;
  - TF-IDF: uso de frequência inversa ou não;
  - Seleção de Características: quantidade dos melhores termos entre 0 e 49; tipos de teste estatístico de qui-quadrado, ANOVA ou mútua informação;
- MLP: neurônio por camada oculta entre 0 e 94, com intervalos de 5; função de ativação de identidade, logarítmica, tangente hiperbólica ou ReLU; algoritmo para otimização dos pesos de L-BFGS, SGD ou Adam; taxa de aprendizado entre 0.0001 e 0.5, com intervalos

de 0.0001; número máximo de épocas entre 0 e 99; tolerância da função de custo entre 0.0001 e 0.1, com intervalos de 0.001; e tamanho de batch entre 0 e 100;

- SGD: função de perda *hinge*, logarítmica, Huber modificado, *hinge* quadrado ou *perceptron*; tipo de regularização de L1, L2 ou *Elastic Net*; cálculo do intercepto ou não; embaralhamento dos dados ou não; e tolerância da função de custo entre 0.0001 e 0.1, com intervalos de 0.001;
- K-NN: quantidade de vizinhos entre 1 e 98, com intervalos de 2.

Após a execução das 100 avaliações pela RS, os classificadores obtiveram o melhor estimador para a sequência da modelagem sobre os dados de treinamento e predição sobre os dados de teste.

A Tabela 3 relaciona os resultados obtidos sobre os dados de treino e teste antes do ajuste de HPs com os obtidos após este processamento.

Tabela 3 – Desempenho dos melhores algoritmos em termos de BAC antes e após etapa de ajuste de hiperparâmetros.

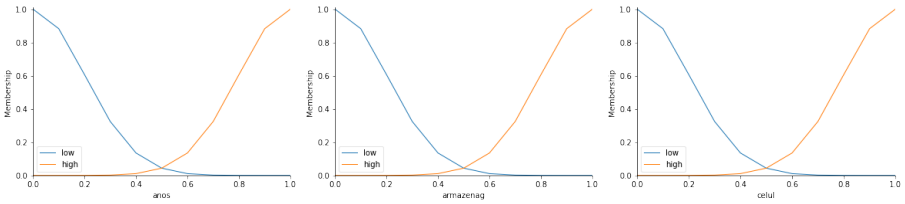
	Default		**com Tuning**	
Algoritmo	**Treino**	**Teste**	**Treino**	**Teste**
MLP	0.9034	0.8804	0.7115	0.7285
SGD	0.9006	0.8876	0.6971	0.6617
K-NN	0.8221	0.8039	0.6578	0.6219

Os melhores valores de HPs encontrados para cada algoritmos são reportados nas Tabelas 4, 5 e 6. Com base nos resultados obtidos, é possível observar que o ajuste de HPs não foi capaz de melhorar os resultados, seja pela quantidade de avaliações ou pelo espaço dos HPs avaliados. Ainda assim, mantiveram o ranqueamento de desempenhos que apresentaram previamente, demonstrando a relação dos dados com os viéses indutivos dos algoritmos, mencionada na seção anterior.

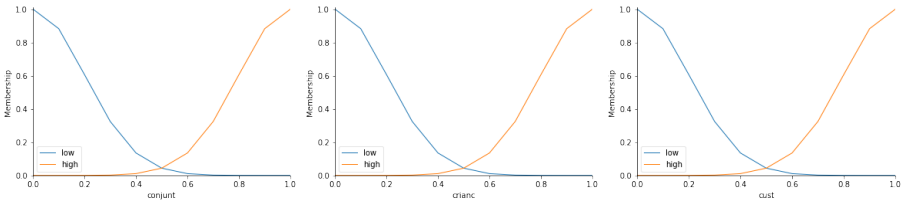
Outro ponto que vale ressaltar é na decisão da incluir o pré-processamento na realização dos ajustes de HPs em conjunto com os algoritmos de AM selecionados. Esta escolha pode ter influenciado negativamente para o decaimento do desempenho de todos os algoritmos. Uma possível comprovação deste fato é no caso do K-NN que teve apenas um HP avaliado e, mesmo assim, não conseguiu atingir o resultado inicial, antes mesmo da aplicação desta última etapa.

5.4 Parâmetros Fuzzy

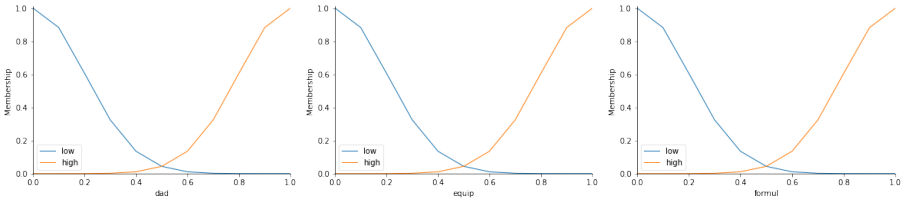
Durante a execução das etapas de pré-processamento da solução com Lógica *Fuzzy*, foi aplicada também a seleção de características, obtendo os melhores termos que melhor representaram os documentos e suas respectivas classes presentes no *dataset*, com base no teste estatístico do qui-quadrado. Dados esses termos, os mesmos foram utilizados como variáveis linguísticas do Sistema *Fuzzy*. Desta forma, as características/termos selecionados foram:



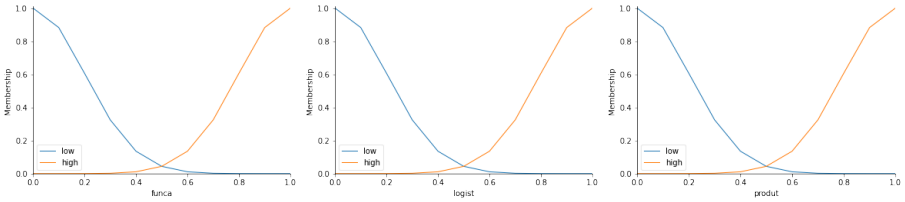
(a) Antecedente 'anos' (b) Antecedente 'armazenag' (c) Antecedente 'celul'



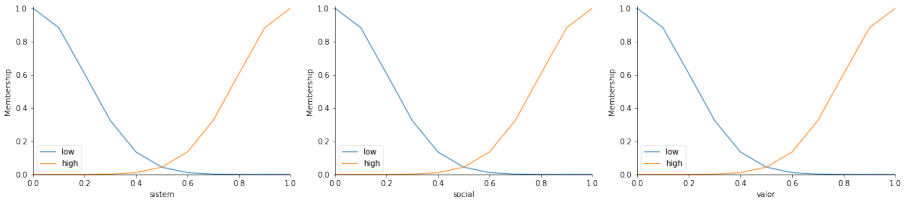
(d) Antecedente 'conjunt' (e) Antecedente 'crianc' (f) Antecedente 'cust'



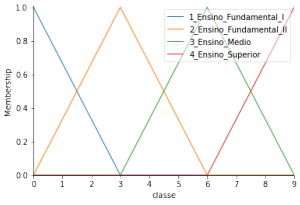
(g) Antecedente 'dad' (h) Antecedente 'equip' (i) Antecedente 'formul'



(j) Antecedente 'funca' (k) Antecedente 'logist' (l) Antecedente 'produt'



(m) Antecedente 'sistem' (n) Antecedente 'social' (o) Antecedente 'valor'



(p) Consequente Classes

Figura 17 – Funções de pertinência do sistema *fuzzy*. Fonte: Autoria própria.

Tabela 4 – Melhores valores de HP encontrados no ajuste da MLP.

Elemento	**Hiperparâmetro**	**Valor**
vetorização	quantidade máx e min de n-gramas	(1,1)
vetorização	frequência min de palavras	7
vetorização	quantidade de maiores frequências	88
TF-IDF	uso de frequência inversa	não
seleção de carac.	quantidade dos melhores termos do dataset	42
seleção de carac.	tipo de teste estatístico	ANOVA
MLP	neurônios por camada oculta	85
MLP	função de ativação utilizada	ReLU
MLP	algoritmo para otimização dos pesos	SGD
MLP	taxa de aprendizado	0.083
MLP	número máximo de épocas	68
MLP	tolerância da função de custo	0.0751
MLP	tamanho de batch	59
Desempenho obtido em BAC		**0.7285**

Tabela 5 – Melhores valores de HP encontrados no ajuste do SGD.

Elemento	**Hiperparâmetro**	**Valor**
vetorização	quantidade máx e min de n-gramas	(1,1)
vetorização	frequência min de palavras	3
vetorização	quantidade de maiores frequências	67
TF-IDF	uso de frequência inversa	sim
seleção de carac.	quantidade dos melhores termos do dataset	67
seleção de carac.	tipo de teste estatístico	qui-quadrado
SGD	função de perda	Huber modificado
SGD	tipo de regularização	L1
SGD	cálculo do intercepto	não
SGD	emabaralhamento dos dados	sim
SGD	tolerância da função de custo	0.0701
Desempenho obtido em BAC		**0.6617**

Tabela 6 – Melhores valores de HP encontrados no ajuste do K-NN.

Elemento	**Hiperparâmetro**	**Valor**
vetorização	quantidade máx e min de n-gramas	(1,1)
vetorização	frequência min de palavras	5
vetorização	quantidade de maiores frequências	59
TF-IDF	uso de frequência inversa	não
seleção de carac.	quantidade dos melhores termos do dataset	39
seleção de carac.	tipo de teste estatístico	mútua informação
K-NN	quantidade de vizinhos	9
Desempenho obtido em BAC		**0.6219**

‘anos’, ‘armazenag’, ‘celul’, ‘conjunt’, ‘crianc’, ‘cust’, ‘dad’, ‘equip’, ‘formul’, ‘funca’, ‘logist’, ‘produt’, ‘sistem’, ‘social’ e ‘valor’.

Conforme comentado anteriormente, estes termos já haviam passado pelo processo de stemização e, por isso, aparecem como radicais da língua portuguesa. Posteriormente, estes foram introduzidos na criação das funções de pertinência, juntamente com as classes utilizadas para este problema. A Figura 17 apresenta as funções de pertinência geradas para cada um dos termos, representando os antecedentes do sistema, e para as classes, representando o consequente.

Com base nas funções definidas para esta solução, é possível perceber que as variáveis linguísticas do antecedente possuem valores de incerteza mais suavizados, dado o comportamento da função Gaussiana. Já para o consequente, a sobreposição entre os valores de cada classe é maior, oferecendo valores mais altos de incerteza por região, graças a configuração e parâmetros das funções triangulares utilizados no Sistema *Fuzzy*.

Quanto ao processo de obtenção das regras *Fuzzy*, foi utilizada uma DT para definir as regras de decisão, simulando o papel do especialista. A árvore completa é apresentada no Apêndice A. Visando uma visualização melhor da estrutura da árvore, foi gerada uma versão reduzida com foco maior nos níveis iniciais, omitindo as folhas. Esta árvore pode ser visualizada na Figura 18.

Considerando a árvore utilizada neste experimento, é possível observar que foi definida a profundidade máxima de 6 níveis, com o objetivo de obter-se menos características, bem como uma quantidade reduzida de regras de decisão para a solução baseada em *Fuzzy*, evitando o sobreajuste aos dados de treinamento e, consequentemente, a dificuldade de generalização sobre os dados de teste ou desconhecidos.

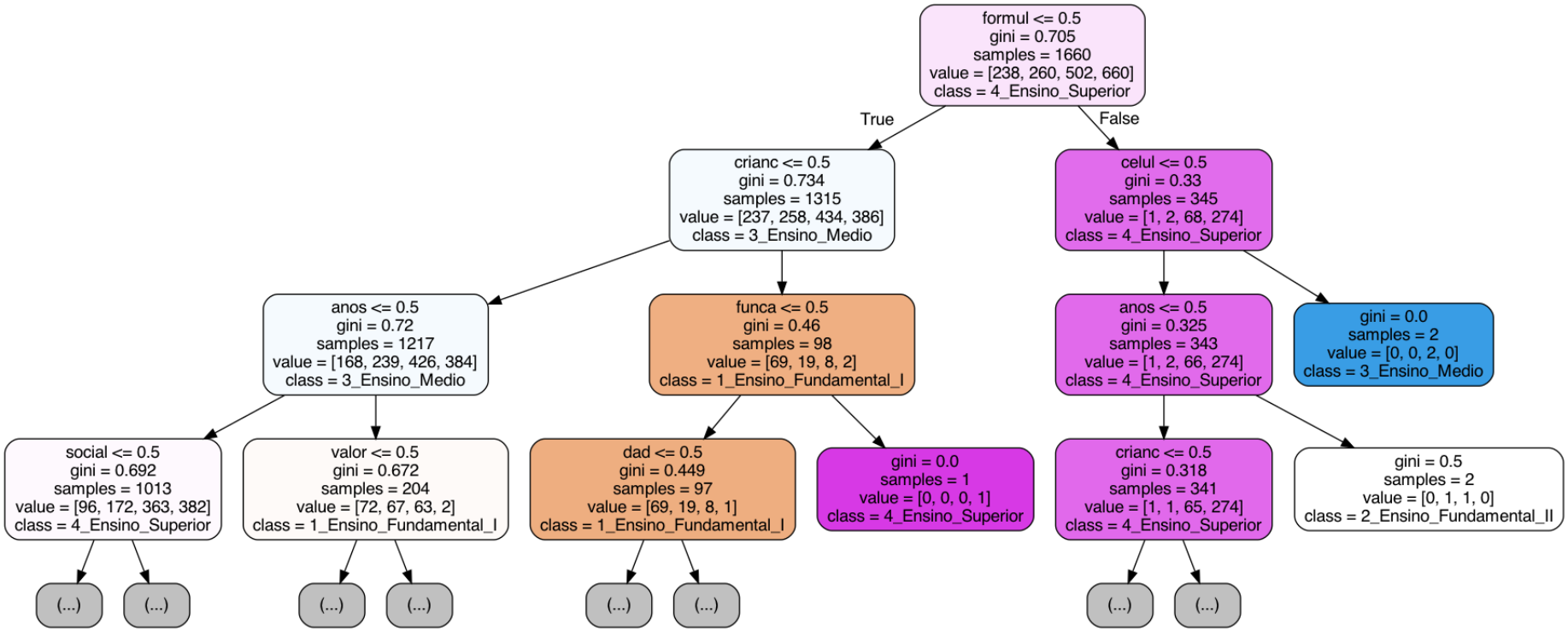


Figura 18 – Árvore de decisão resumida com as regras *fuzzy*. Fonte: Autoria própria.

Na representação gráfica apresentada, é possível perceber que, partindo do nó raiz, conforme a frequência dos termos presentes nos textos avaliados e em cada nó aumenta, a decisão é conduzida para um caminho válido. Cada caminho, por sua vez, passa pelos níveis da árvore, até atingir um nó folha específico. Assim, ao atingir um determinado nó folha, é possível classificar o texto de acordo com um grau de escolaridade específico. Estes caminhos identificados formam as regras do Sistema *Fuzzy*. Desta forma, foi obtido um total de 22 regras, descritas abaixo:

- **Regra 1:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'baixo' E celul = 'baixo' **E** equip = 'baixo') **ENTÃO** classe = 'Ensino Superior';
- **Regra 2:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'baixo' **E** celul = 'baixo' **E** equip = 'alto') **ENTÃO** classe = 'Ensino Superior';
- **Regra 3:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'baixo' **E** celul = 'alto' **E** produt = 'baixo') **ENTÃO** classe = 'Ensino Médio';
- **Regra 4:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'baixo' **E** celul = 'alto' **E** produt = 'alto') **ENTÃO** classe = 'Ensino Fundamental 1';

- **Regra 5:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'alto' **E** dad = 'baixo' **E** logist = 'baixo') **ENTÃO** classe = 'Ensino Fundamental 2';
- **Regra 6:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'alto' **E** dad = 'baixo' **E** logist = 'alto') **ENTÃO** classe = 'Ensino Superior';
- **Regra 7:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'baixo' **E** social = 'alto' **E** dad = 'alto') **ENTÃO** classe = 'Ensino Superior';
- **Regra 8:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'alto' **E** valor = 'baixo' **E** funca = 'baixo' **E** produt = 'baixo') **ENTÃO** classe = 'Ensino Fundamental 1';
- **Regra 9:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'alto' **E** valor = 'baixo' **E** funca = 'baixo' **E** produt = 'alto') **ENTÃO** classe = 'Ensino Fundamental 1';
- **Regra 10:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'alto' **E** valor = 'baixo' **E** funca = 'alto') **ENTÃO** classe = 'Ensino Fundamental 2';
- **Regra 11:** **SE** (formul = 'baixo' **E** crianc = 'baixo' **E** anos = 'alto' **E** valor = 'alto') **ENTÃO** classe = 'Ensino Médio';
- **Regra 12:** **SE** (formul = 'baixo' **E** crianc = 'alto' **E** funca = 'baixo' **E** dad = 'baixo' **E** anos = 'baixo' **E** produt = 'baixo') **ENTÃO** classe = 'Ensino Fundamental 1';
- **Regra 13:** **SE** (formul = 'baixo' **E** crianc = 'alto' **E** funca = 'baixo' **E** dad = 'baixo' **E** anos = 'baixo' **E** produt = 'alto') **ENTÃO** classe = 'Ensino Fundamental 1';
- **Regra 14:** **SE** (formul = 'baixo' **E** crianc = 'alto' **E** funca = 'baixo' **E** dad = 'baixo' **E** anos = 'alto') **ENTÃO** classe = 'Ensino Fundamental 1';
- **Regra 15:** **SE** (formul = 'baixo' **E** crianc = 'alto' **E** funca = 'baixo' **E** dad = 'alto') **ENTÃO** classe = 'Ensino Fundamental 2';
- **Regra 16:** **SE** (formul = 'baixo' **E** crianc = 'alto' **E** funca = 'alto') **ENTÃO** classe = 'Ensino Superior';
- **Regra 17:** **SE** (formul = 'alto' **E** celul = 'baixo' **E** anos = 'baixo' **E** crianc = 'baixo' **E** valor = 'baixo' **E** cust = 'baixo') **ENTÃO** classe = 'Ensino Superior';
- **Regra 18:** **SE** (formul = 'alto' **E** celul = 'baixo' **E** anos = 'baixo' **E** crianc = 'baixo' **E** valor = 'baixo' **E** cust = 'alto') **ENTÃO** classe = 'Ensino Superior';
- **Regra 19:** **SE** (formul = 'alto' **E** celul = 'baixo' **E** anos = 'baixo' **E** crianc = 'baixo' **E** valor = 'alto') **ENTÃO** classe = 'Ensino Superior';
- **Regra 20:** **SE** (formul = 'alto' **E** celul = 'baixo' **E** anos = 'baixo' **E** crianc = 'alto') **ENTÃO** classe = 'Ensino Médio';
- **Regra 21:** **SE** (formul = 'alto' **E** celul = 'baixo' **E** anos = 'alto') **ENTÃO** classe = 'Ensino Fundamental 2'; e
- **Regra 22:** **SE** (formul = 'alto' **E** celul = 'alto') **ENTÃO** classe = 'Ensino Médio'.

Com base nas regras induzidas, é possível observar que, dos 15 termos selecionados na etapa de pré-processamento, apenas 12 foram utilizados na composição das regras, visto que a árvore geradora das regras foi limitada a 6 níveis de profundidade.

Como quantidade de regras e termos por classe, existem, ao todo, 6 regras (4, 8, 9, 12, 13 e 14) e 9 termos distintos de Ensino Fundamental 1, 4 regras (5, 10, 15 e 21) e 9 termos distintos de Ensino Fundamental 2, 4 regras (3, 11, 20 e 22) e 7 termos distintos de Ensino Médio e 8 regras (1, 2, 6, 7, 16, 17, 18 e 19) e 11 termos distintos de Ensino Superior.

Além disso, é possível perceber que a DT foi capaz de aprender alguns padrões nas regras por classe, a partir do processo de ajuste sobre os dados. Estes padrões estão relacionados a pesos maiores em certos termos, o que representa foco maior na frequência de palavras relacionadas a estes radicais, e menores em outros termos, representando uma relevância menor ou inexistente nestes outros. São eles:

- Na classe de Ensino Fundamental 1, são procurados textos com pesos maiores do termo ‘celul’. Ao mesmo tempo, são buscados pesos menores dos termos ‘formul’, ‘valor’, ‘dad’, ‘funca’ e ‘social’;
- Na classe de Ensino Fundamental 2, são procurados textos com pesos maiores do termo ‘social’. Ao mesmo tempo, são buscados pesos menores dos termos ‘celul’, ‘logist’ e ‘valor’;
- Na classe de Ensino Médio, são procurados textos com pesos maiores do termo ‘valor’. Ao mesmo tempo, são buscados pesos menores dos termos ‘social’ e ‘produt’; e
- Na classe de Ensino Superior, são procurados textos com pesos maiores do termo ‘logist’ e ‘funca’. Ao mesmo tempo, são buscados pesos menores dos termos ‘anos’ e ‘celul’.

Seguindo a análise das regras, é possível visualizar algumas simplificações, de modo que alguns termos relacionados para certa classe deixem de ser relevantes, justamente por não haver diferença entre buscar pesos baixos ou altos para palavras com o mesmo radical nos textos. Por exemplo, no caso da classe de Ensino Superior, as regras 1 e 2 poderiam ser simplificadas, retirando a verificação do termo ‘equip’; e, ainda na mesma classe, as regras 17, 18 e 19 poderiam também ser reduzidas, retirando ‘valor’ e ‘cust’. Desta forma, esses termos se tornam indiferentes para a respectiva classe. Entretanto, apesar da possibilidade de simplificação, esta não ocorreu de fato, devido ao processo automatizado criado que visou apenas introduzir todas as regras geradas pela DT no Sistema *Fuzzy*, independente de qualquer redução ou simplificação possível.

Ademais, é possível estabelecer um paralelo entre os termos procurados nos textos pelas regras geradas, seja com menor ou maior peso, com as nuvens de palavras geradas para cada classe, após a etapa de pré-processamento, apresentadas na Figura 12. Seguem as considerações:

- Na classe de Ensino Fundamental 1, ficam aparentes termos relacionados aos radicais

‘crianc’, ‘anos’, e ‘produt’;

- Na classe de Ensino Fundamental 2, ficam aparentes termos relacionados aos radicais ‘social’, ‘celul’, ‘crianc’, ‘anos’ e ‘funca’;
- Na classe de Ensino Médio, ficam aparentes termos relacionados aos radicais ‘celul’, ‘valor’ e ‘formul’; e
- Na classe de Ensino Superior, ficam aparentes termos relacionados aos radicais ‘logist’, ‘funca’, ‘valor’, ‘dad’, ‘formul’, ‘cust’ e ‘equip’.

Desta forma, é justificável que estes radicais relacionados tenham um destaque maior nas regras, de modo que sejam buscados com peso maior, visando classificar os textos conforme uma classe específica. Por exemplo, na classe de Ensino Fundamental 2, o termo ‘social’ é buscado com maior relevância; já na classe de Ensino Médio, o termo ‘valor’ está em maior destaque; e, finalmente, na classe de Ensino Superior, os termos ‘logist’ e ‘funca’ possuem foco maior.

Portanto, com base nas correspondências citadas, é possível concluir que a Lógica *Fuzzy* teve um foco maior no aprendizado da classe de Ensino Superior, com uma quantidade maior de regras induzidas (8 de 22) e de radicais considerados nestas (11 de 12), além de termos com peso maior e com destaque nas nuvens de palavras (2). Logo em seguida, a classe de Ensino Fundamental 1, além de obter uma quantidade de regras maior do que as restantes (6 de 22), buscou um peso menor em mais termos (5), representando ainda uma especialização nestas características, especificamente. Já as demais classes, houve poucas diferenças, apenas na quantidade de termos das regras (9 de 12) e nos termos que aparecem nas nuvens de palavras que também foram relacionados nas regras da classe de Ensino Fundamental 2 (5 de 12), dando a esta uma relevância maior em comparação a classe de Ensino Médio.

Com isso, é possível afirmar que o desbalanceamento de classes favoreceu o aprendizado da classe majoritária, Ensino Superior. Em contrapartida, não colocou em destaque a segunda classe em quantidade de textos, Ensino Médio, pois esta influenciou menos no aprendizado do que as demais classes. Uma possível justificativa para este fenômeno é a existência de características em comum entre as 2 primeiras classes em quantidade de documentos que foram induzidas equivocadamente pela DT como pertencendo a primeira classe e, por consequência, não valorizando o aprendizado da segunda.

5.5 Analisando as Predições dos Modelos

Uma vez obtidos os três modelos de AM principais, com os melhores HPs, e o modelo de Lógica *Fuzzy*, com seus parâmetros selecionados, é interessante olhar as predições individuais dos mesmos, a fim de verificar quais classes conseguem ser melhor preditas pelos modelos. Esta análise das predições também permite identificar limitações dos modelos e conduzir um melhor processo de refinamento para novos e melhores modelos. Para este fim, as Figuras 19 e 20 apresentam um comparativo das matrizes de confusão de cada modelo, respectivamente: MLP,

SGD, K-NN, *Fuzzy* e *baselines*.

Vale lembrar que os *baselines* também foram usados nesta etapa para quantificar a melhora que os modelos de AM e *Fuzzy* apresentaram com seus processamentos intermediários em comparação com estes que não foram submetidos a qualquer processamento, apenas uma predição aleatória qualquer ou, simplesmente, fixa numa determinada classe. Além disso, foi possível focar nas predições específicas em cada classe e contrastar suas predições ingênuas com as dos demais modelos.

As matrizes de confusão apresentam um comparativo entre as predições e os dados reais, onde cada célula apresenta a quantidade de exemplos preditos pra classe em destaque nas colunas, enquanto que, na realidade, os dados pertenciam a classe destacada nas linhas. Assim, as células localizadas na diagonal principal representam as predições corretas de cada classe, enquanto as demais representam as incorretas. Outro dado adicional presente nas células foi a porcentagem do quantitativo em relação a todos os exemplos existentes no *dataset*.

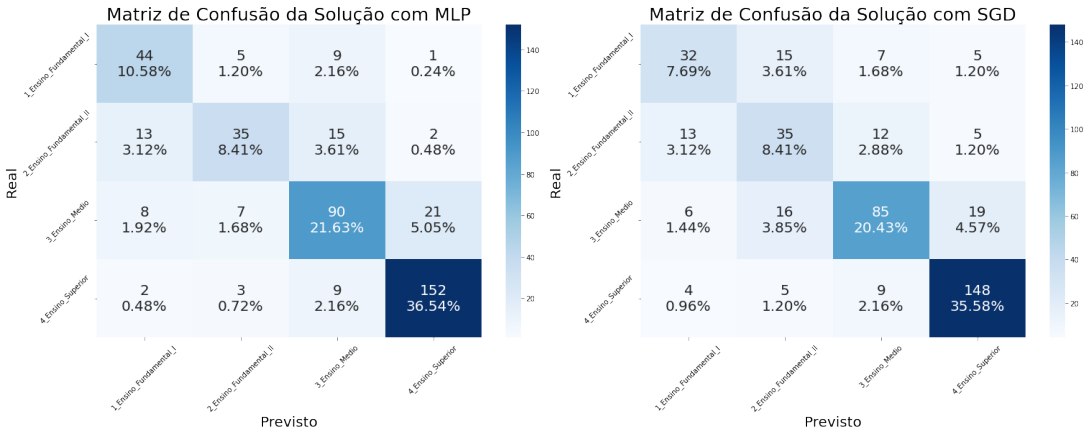
Além disso, é importante ressaltar que, além das porcentagens citadas nas matrizes de confusão, existem também porcentagens que representam o quantitativo de predições corretas em cada classe. As mesmas são comentadas na sequência. Logo, cada célula pode ser vista em relação aos quantitativos de cada classe no conjunto de teste, sendo eles: Ensino Fundamental 1 com 59 exemplos; Ensino Fundamental 2 com 63 exemplos; Ensino Médio com 126 exemplos; e Ensino Superior com 166 exemplos.

Analisando as matrizes de confusão, é possível perceber que as soluções com ML e *Fuzzy* tiveram desempenho similar para identificar a classe majoritária, que representam os textos com conteúdo do Ensino Superior. Ambas abordagens apresentaram praticamente uma mesma quantidade de predições corretas, entre 87% e 99% de acertos, com uma pequena vantagem do *Fuzzy*.

A diferença de desempenho ocorre na predição das demais classes: conteúdos textuais de Ensino Médio e Fundamental 1 e 2. Nos exemplos de textos com conteúdo de Ensino Médio, as soluções com ML atingem entre 65 e 75% de acertos. Já a solução com Lógica *Fuzzy* atinge apenas, aproximadamente, 22% de acertos. A maioria das predições incorretas ocorrem em grande parte pela atribuição errônea da classe Ensino Superior, representando, aproximadamente, 68% de predições equivocadas para esta classe. Isto se dá graças as características em comum de ambas as classes, citadas na seção de parâmetros *Fuzzy*.

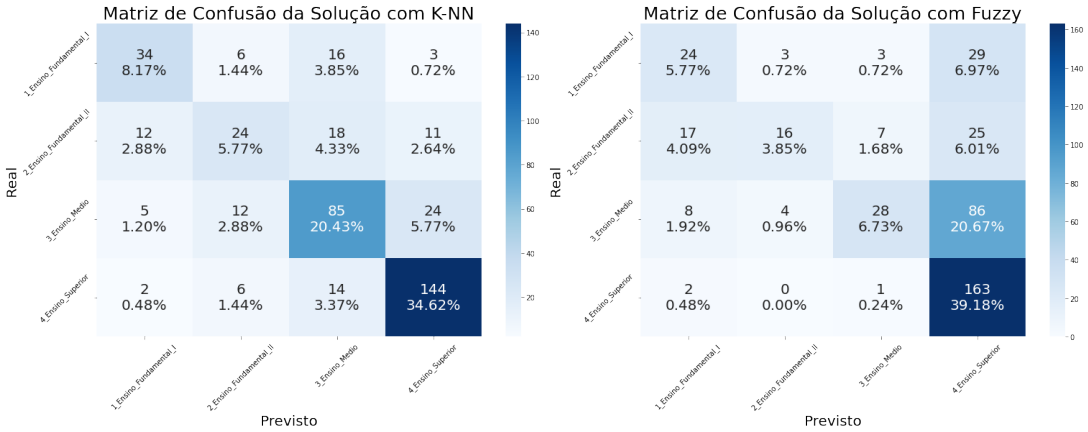
No caso das predições de Ensino Fundamental 2, enquanto as soluções com ML obtiveram um total de acertos entre 35 e 55%, a com Lógica *Fuzzy* obteve apenas, aproximadamente, 25%. Assim como na classe de Ensino Médio, este desempenho ruim do *Fuzzy* também ocorre devido ao aprendizado das Regras *Fuzzy* não ter assimilado tantas características dessa classe, também mencionado na seção de parâmetros *Fuzzy*.

Seguindo para a classe de Ensino Fundamental 1, as soluções com ML obtiveram um total de acertos entre 54 e 75%, enquanto o Sistema *Fuzzy* obteve apenas, aproximadamente, 41% de predições corretas para esta classe. Neste caso, a solução baseada em *Fuzzy* conseguiu



(a) MLP

(b) SGD



(c) K-NN

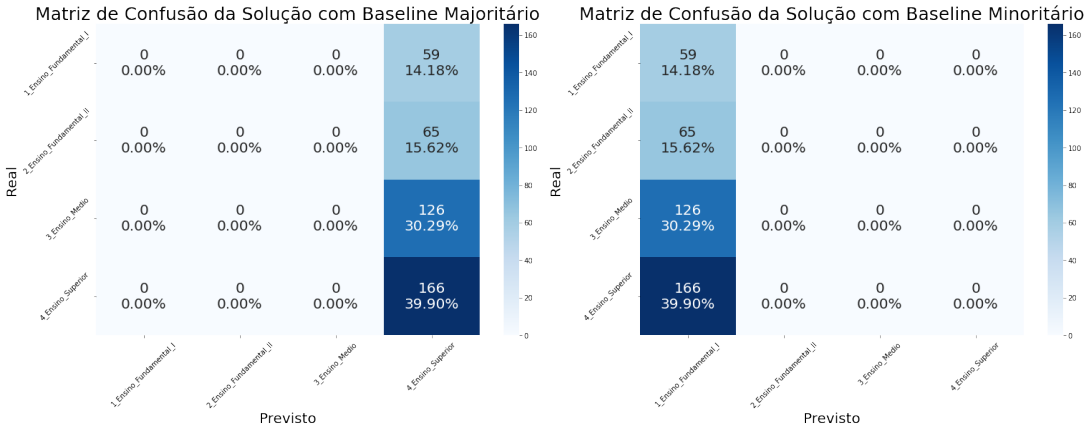
(d) *Fuzzy*

Figura 19 – Matrizes de Confusão do Universo de Soluções - parte 1 - Classificadores baseados em AM e *Fuzzy*. Fonte: Autoria própria.

obter um resultado melhor do que na Fundamental 2, justamente por ter mais características assimiladas nas regras, conforme mencionado anteriormente. Porém, ainda perde das soluções de AM em decorrência do foco de suas regras ser maior na classe majoritária.

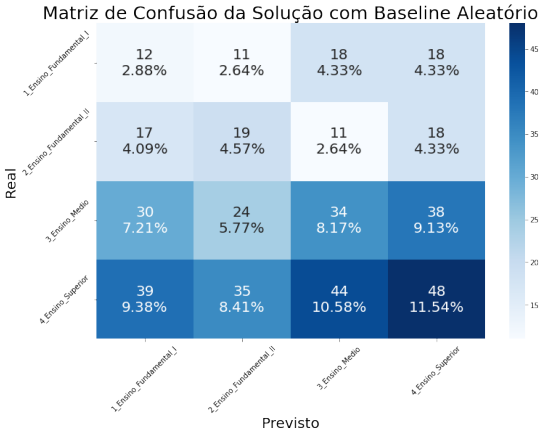
No caso dos *baselines* utilizados para este experimento, podemos perceber que os *baselines* Majoritário e Minoritário, pelo fato de realizarem suas predições em uma só classe, acabam acertando completamente suas respectivas classes (100%) e errando completamente as demais (0%). Já no caso do *baseline* aleatório simples (*Random*), é possível perceber uma vantagem sobre a solução com *Fuzzy* nas classes Ensino Médio e Fundamental 2, onde obteve-se um total de, aproximadamente, 27 e 29% de acertos, respectivamente, enquanto que nas classes Ensino Superior e Fundamental 1, os acertos chegaram apenas em, aproximadamente, 29 e 20%, respectivamente. Como mencionado nas demais classes, esta vantagem do *baseline* aleatório sobre o *Fuzzy* se dá devido ao segundo dispor de mais regras e aprendizado na classe majoritária, tendo muitas predições equivocadas nesta classe e, por consequência, não rotulando corretamente as demais.

Complementando os resultados das predições, também foram computadas outras três



(a) Classe Majoritária

(b) Classe Minoritária



(c) Aleatório

Figura 20 – Matrizes de Confusão do Universo de Soluções - parte 2 - *Baselines*. Fonte: Autoria própria.

métricas de desempenho dos classificadores obtidas sobre o conjunto de testes. A Tabela 7 apresenta um comparativo entre as métricas de precisão (*precision)*, revocação (*recall)* e F1, respectivamente.

Assim, é possível perceber que a solução baseada em Lógica *Fuzzy* acerta mais predições que os métodos de AM na revocação da classe de Ensino Superior e, praticamente, na precisão da classe Ensino Fundamental 2, perdendo apenas para a MLP. Isto mostra que o aprendizado das Regras *Fuzzy* conseguiu atingir mais exemplos corretos de Ensino Superior (revocação) do que dos modelos de AM, graças ao seu foco maior no aprendizado desta classe. Ademais, o modelo *Fuzzy* conseguiu responder corretamente mais a classe de Ensino Fundamental 2 (precisão) do que os modelos SGD e K-NN, devido ao número de predições nesta classe ter sido também menor do que os modelos de AM, já que o aprendizado assimilado nesta classe foi reduzido e, por conta disso, acabou acertando mais.

No restante das métricas e classes, as soluções com AM superam o Sistema *Fuzzy*, principalmente, nas medidas F1. Isso é natural, visto que o F1 é uma medida que pondera tanto a precisão como a revocação, e o Sistema *Fuzzy* não conseguiu superar as demais soluções em

Tabela 7 – Comparativo da precisão (*precision)*, revocação (*recall*) e medida F1 dos melhores classificadores e *baselines* em cada classe do problema (autoria própria).

Precision							
Classe	**MLP**	**SGD**	**K-NN**	**Fuzzy**	**Random**	**Major.**	**Min.**
1_Ensino_Fundamental_I	0.6657	0.5818	0.6415	0.4706	0.1224	0.0000	0.1418
2_Ensino_Fundamental_II	0.7000	0.4930	0.5000	0.6957	0.2135	0.0000	0.0000
3_Ensino_Medio	0.7317	0.7522	0.6391	0.7179	0.3178	0.0000	0.0000
4_Ensino_Superior	0.8336	0.8362	0.7912	0.5380	0.3934	0.3990	0.0000
Média	0.7328	0.6658	0.6430	0.6056	0.2618	0.0998	0.0355
Recall							
Classe	**MLP**	**SGD**	**K-NN**	**Fuzzy**	**Random**	**Major.**	**Min.**
1_Ensino_Fundamental_I	0.7458	0.5424	0.5763	0.4068	0.2034	0.0000	1.0000
2_Ensino_Fundamental_II	0.5385	0.5385	0.3692	0.2462	0.2923	0.0000	0.0000
3_Ensino_Medio	0.7143	0.6746	0.6746	0.2222	0.2698	0.0000	0.0000
4_Ensino_Superior	0.9157	0.8916	0.8675	0.9819	0.2892	1.0000	0.0000
Média	0.7286	0.6618	0.6219	0.4643	0.2637	0.2500	0.2500
Medida F1							
Classe	**MLP**	**SGD**	**K-NN**	**Fuzzy**	**Random**	**Major.**	**Min.**
1_Ensino_Fundamental_I	0.6984	0.5614	0.6071	0.4364	0.1529	0.0000	0.2484
2_Ensino_Fundamental_II	0.6087	0.5147	0.4248	0.3636	0.2468	0.0000	0.0000
3_Ensino_Medio	0.7229	0.7113	0.6564	0.3394	0.2918	0.0000	0.0000
4_Ensino_Superior	0.8889	0.8630	0.8276	0.6951	0.3333	0.5704	0.0000
Média	0.7297	0.6626	0.6290	0.4586	0.2562	0.1426	0.0621

ambas medidas. Já os *baselines* foram superados nas medidas de precisão e, consequentemente, na medida F1, mantendo seu desempenho mencionado anteriormente apenas na revocação.

5.6 Considerações Finais

Os resultados descritos nesse capítulo mostram que a aplicação de técnicas de pré-processamento para descarte de palavras vazias, incluindo as irrelevantes identificadas nas etapas iniciais dos experimentos, e o processo de stemização refinaram e aperfeiçoaram a generalização dos modelos indutivos. Isto se deu devido à homogeneização da distribuição dos termos nos textos em cada classe e a eliminação de ruídos nos dados, os quais poderiam apresentar falsas impressões sobre os dados ou até destacar certas características que, de fato, não refletem a realidade do contexto do problema analisado.

Além disso, ao comparar e selecionar os principais modelos de AM, pode-se perceber que, mesmo após o ajuste de HPs das etapas de pré-processamento e classificação, os modelos não foram capazes de melhorar os resultados anteriormente obtidos no conjunto de treinamento, muito por conta da quantidade de avaliações realizadas ou da definição do espaço avaliado em

cada HP. Entretanto, mesmo perdendo desempenho, os modelos de AM ainda continuaram mais altos do que os de *Fuzzy*, inclusive na predição dos dados de teste.

Quanto aos parâmetros *Fuzzy*, é possível perceber que estão diretamente relacionados com o desempenho do Sistema *Fuzzy*. Do ponto de vista da profundida da DT escolhida, o intuito era evitar o sobreajuste das predições no conjunto de treinamento. Já na quantidade de características selecionadas, foi possível refletir a composição das palavras com maiores frequências, identificadas inclusive pelas nuvens de palavras. Porém, na indução das Regras *Fuzzy*, pode-se perceber um aprendizado e foco maior na classe majoritária (Ensino Superior), o que, inclusive, favoreceu a maior proporção absoluta de acertos nesta classe em comparação as demais.

Por fim, com base nas predições no conjunto de teste, é possível afirmar que, em geral, as técnicas baseadas em AM conseguem ter um desempenho superior, e reduzem a quantidade de predições incorretas nas classes com menos exemplos (Ensino Fundamental 1 e 2 e Ensino Médio). Nesse sentido, o Sistema *Fuzzy* não consegue discernir todos os exemplos que podem ser dúbios, carecendo de um maior enriquecimento na geração das suas regras. Essa melhoria pode ser realizada por meio de ajustes finos em suas funções de pertinência, na geração automática de regras, ou, até mesmo, uma possível solução utilizando a criação de ensembles para o ajuste das Regras *Fuzzy*, alternativa frequentemente utilizada na literatura para soluções baseadas em AM (MARSLAND, 2009; RASCHKA; MIRJALILI, 2017). Apesar do desempenho ínfimo apresentado, a Lógica *Fuzzy* ainda possui uma vantagem em relação aos *baselines* escolhidos, além de fornecer a interpretação dos resultados de forma mais direta do que os algoritmos de AM analisados, graças as regras geradas. Este poder de interpretação também pode ser estendido em casos onde há incertezas, como no contexto do problema analisado neste trabalho, em que uma palavra pode ter uma frequência ou peso similar em textos de classes diferentes, favorecendo também a utilização da Lógica *Fuzzy* nesse sentido.

6 CONCLUSÃO

Este trabalho apresentou a implementação de duas soluções para a tarefa de classificação de textos escolares: a primeira baseada em Aprendizado de Máquina, e a segunda em Lógica *Fuzzy*. Esta última foi integrada com a execução prévia de uma Árvore de Decisão, a fim de criar as Regras *Fuzzy* que compuseram a base de conhecimento do Sistema *Fuzzy*. A partir dos resultados experimentais, pode-se observar que as técnicas de AM puderam oferecer bons resultados gerais. Já o Sistema *Fuzzy* criado é também capaz de classificar corretamente os dados, tendo desempenho superior aos *baselines* escolhidos. O desempenho do Sistema *Fuzzy* é inferior aos algoritmos de AM, mas ainda assim é capaz de aprender com os dados, principalmente devido às etapas anteriores de pré-processamento e criação automática das Regras *Fuzzy*. Entretanto, os resultados sugerem que a segunda solução ainda carece de uma análise mais detalhada: testando diferentes técnicas de seleção de modelos e extração de características; e seleção de outras funções de pertinência para as variáveis linguísticas dos antecedentes e consequente, principalmente nas classes do *dataset* com menor quantidade de exemplos.

6.1 LIMITAÇÕES E TRABALHOS FUTUROS

Apesar dos resultados positivos, o trabalho tem algumas limitações que refletem às escolhas feitas para a condução dos experimentos. Como principais limitações e soluções propostas para possíveis melhorias do trabalho, é possível pontuar os seguintes itens:

- A seleção de modelo utilizou apenas a configuração de validação cruzada aninhada com um *holdout* externo e um CV interno. Como melhoria, poderia ser avaliada também a utilização da validação cruzada aninhada com um CV externo e um *holdout* interno;
- A classificação de textos foi realizada apenas comparando-se as classes um-contra-um e não um-contra-o-resto, método no qual poderia oferecer um desempenho maior. Como melhoria, poderia ser aplicada a comparação com este método também, visando o aumento de desempenho do processo de classificação;
- Existem diversas combinações de hiperparâmetros possíveis de serem aplicadas e, por conta disso, seria preciso utilizar técnicas de ajuste de hiperparâmetros mais robustas ou ainda focar apenas nos hiperparâmetros dos algoritmos de AM e não nos das técnicas de pré-processamento. Como melhoria, poderia haver um refinamento dos métodos de seleção de modelos, utilizando ajuste de hiperparâmetros a partir de Otimização de Enxame de Partículas (PSO), Algoritmos Genéticos (AG) ou Otimização Bayesiana (SMBO), além de focar apenas nos hiperparâmetros dos modelos de AM;
- Os parâmetros *Fuzzy* não passaram por uma etapa de ajustes, como os hiperparâmetros na solução com AM. Como melhoria, poderia ser adicionada uma etapa complementar de

ajuste de parâmetros *Fuzzy*, testando diversos hiperparâmetros dos modelos de indução das Regras *Fuzzy*;

- O tamanho e método de escolha dos termos para a entrada do Sistema *Fuzzy* pode ter resultado num desempenho inferior do modelo de classificação obtido. Como melhoria, poderia haver a seleção de características adicionais dos textos para serem usados na Árvore de Decisão e compor as Regras *Fuzzy*. Além disso, seria possível comparar métodos de seleção de características adicionais, com outros testes estatísticos ou ainda diferentes perspectivas de análise de importância sobre as características dos textos;
- O método de Mandani e as funções de pertinência escolhidas podem ter gerado modelos de Inferência *Fuzzy* menos precisos, demandando testes com outras variações. Como melhoria, poderia haver testes com outras combinações de funções de pertinência para os antecedentes e consequente do Sistema *Fuzzy*, bem como a aplicação de um Sistema de Inferência Neuro-Difuso (*Fuzzy*) Adaptativo (ANFIS) (JANG, 1993); e
- A utilização de uma Árvore de Decisão para gerar as Regras *Fuzzy* pode ter levado o modelo a um sobreajuste na classe majoritária. Como melhoria, poderia haver a aplicação de algoritmos de comitês ou *ensembles* para a obtenção das Regras *Fuzzy*.

Referências

BARBOSA, J. L. N. et al. Introdução ao processamento de linguagem natural usando python. In: **Escola Regional de Informática do Piauí: Livro Anais - Artigos e Minicursos**. Teresina, PI: Escola Regional de Informática do Piauí (ERIPI), 2017. Citado na página 17.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. **J. Mach. Learn. Res.**, v. 13, p. 281–305, mar 2012. Citado 2 vezes nas páginas 8 e 26.

BRASIL. Lei nº 9.394, de 20 de dezembro de 1996. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 1996. Estabelece as diretrizes e bases da educação nacional. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/l9394.htm>. Citado 2 vezes nas páginas 16 e 34.

BREIMAN, L. Bagging predictors. **Machine Learning**, v. 24, n. 2, p. 123–140, Aug 1996. ISSN 1573-0565. Citado na página 19.

BREIMAN, L. Random forests. **Machine Learning**, v. 45, n. 1, p. 5–32, Oct 2001. ISSN 1573-0565. Citado na página 19.

BREIMAN, L. et al. **Classification and Regression Trees**. [S.l.]: Taylor & Francis, 1984. ISBN 9780412048418. Citado na página 19.

BUITINCK, L. et al. API design for machine learning software: experiences from the scikit-learn project. In: **ECML PKDD Workshop: Languages for Data Mining and Machine Learning**. [S.l.: s.n.], 2013. p. 108–122. Citado 2 vezes nas páginas 39 e 47.

COPPIN, B.; VALÉRIO, J. **Inteligência artificial**. Grupo Gen - LTC, 2015. ISBN 9788521629351. Disponível em: <https://books.google.com.br/books?id=ZEK2AQAACAAJ>. Citado na página 21.

ELLSON, J. et al. Graphviz — open source graph drawing tools. In: **Lecture Notes in Computer Science**. [S.l.]: Springer-Verlag, 2001. p. 483–484. Citado na página 39.

FACELI, K. et al. **Inteligência artificial: uma abordagem de aprendizado de máquina**. [S.l.]: LTC, 2011. Citado na página 19.

FINATTO, M. J. B.; LOPES, L.; CIULLA, A. Processamento de linguagem natural, linguística de corpus e estudos linguísticos: uma parceria bem-sucedida. **Domínios de Lingu@gem**, v. 9, n. 5, p. 41–59, ago. 2015. Disponível em: <http://www.seer.ufu.br/index.php/dominiosdelinguagem/article/view/28670>. Citado na página 14.

GAZZOLA, M.; LEAL, S. E.; ALUISIO, S. M. **Predição da Complexidade Textual de Recursos Educacionais Abertos em Português**. 2019. <https://github.com/gazzola/corpus_readability_nlp_portuguese>. Accessed: 2021-12-05. Citado na página 34.

GéRON, A. **Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & TensorFlow**. [S.l.]: Alta Books, 2019. ISBN 9788550809021. Citado 4 vezes nas páginas 23, 25, 26 e 47.

HARRIS, C. R. et al. Array programming with NumPy. **Nature**, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, sep 2020. Disponível em: <https://doi.org/10.1038/s41586-020-2649-2>. Citado na página 39.

HUNTER, J. D. Matplotlib: A 2d graphics environment. **Computing in Science & Engineering**, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007. Citado na página 39.

JANG, J. S. ANFIS: adaptive-network-based fuzzy inference system. **IEEE Trans. on Systems, Man, and Cybernetics**, v. 23, n. 3, p. 665–685, 1993. Citado na página 63.

LOPER, E.; BIRD, S. Nltk: The natural language toolkit. **CoRR**, cs.CL/0205028, 2002. Disponível em: <http://dblp.uni-trier.de/db/journals/corr/corr0205.html#cs-CL-0205028>. Citado na página 39.

MAMDANI, E.; ASSILIAN, S. An experiment in linguistic synthesis with a fuzzy logic controller. **International Journal of Man-Machine Studies**, v. 7, n. 1, p. 1–13, 1975. ISSN 0020-7373. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020737375800022>. Citado na página 22.

MARSLAND, S. **Machine Learning - An Algorithmic Perspective**. [S.l.]: CRC Press, 2009. I-XVI, 1-390 p. (Chapman and Hall / CRC machine learning and pattern recognition series). ISBN 978-1-4200-6718-7. Citado na página 61.

MCCALLUM, A.; NIGAM, K. A comparison of event models for naive bayes text classification. **Work Learn Text Categ**, v. 752, 05 2001. Citado 3 vezes nas páginas 14, 30 e 33.

MCKINNEY Wes. Data Structures for Statistical Computing in Python. In: WALT Stéfan van der; MILLMAN Jarrod (Ed.). **Proceedings of the 9th Python in Science Conference**. [S.l.: s.n.], 2010. p. 56 – 61. Citado na página 39.

MITCHELL, T. **Machine Learning**. [S.l.]: McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673. Citado 3 vezes nas páginas 18, 28 e 35.

MORAIS, E. A. M. **Mineração de Textos**. Goiás, GO, 2007. Citado 3 vezes nas páginas 17, 18 e 24.

PINTAS, J. T.; FERNANDES, L. A. F.; GARCIA, A. C. B. Feature selection methods for text classification: a systematic literature review. **Artificial Intelligence Review**, v. 54, n. 8, p. 6149–6200, Dec 2021. ISSN 1573-7462. Disponível em: <https://doi.org/10.1007/s10462-021-09970-6>. Citado na página 18.

RASCHKA, S.; MIRJALILI, V. **Python Machine Learning: Machine Learning and Deep Learning with Python, Scikit-Learn, and TensorFlow, 2nd Edition**. 2nd. ed. [S.l.]: Packt Publishing, 2017. ISBN 1787125939. Citado 4 vezes nas páginas 24, 25, 36 e 61.

SANTAFé, G.; INZA, I.; LOZANO, J. Dealing with the evaluation of supervised classification algorithms. **Artificial Intelligence Review**, v. 44, 06 2015. Citado na página 29.

SILVA, A. Classificação de texto para categorização de crimes contra a honra. In: **Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional**. Porto Alegre, RS, Brasil: SBC, 2019. p. 961–971. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/eniac/article/view/9349>. Citado 3 vezes nas páginas 14, 30 e 33.

SUN, J. et al. Fuzzy logic-based natural language processing and its application to speech recognition. 01 2002. Citado 3 vezes nas páginas 15, 31 e 33.

WARNER, A. J. Scikit-fuzzy. In: . [S.l.: s.n.], 2012. Citado na página 39.

WASKOM, M. L. seaborn: statistical data visualization. **Journal of Open Source Software**, The Open Journal, v. 6, n. 60, p. 3021, 2021. Disponível em: <https://doi.org/10.21105/joss.03021>. Citado na página 39.

WU, K. et al. A fuzzy logic-based text classification method for social media data. In: **2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)**. [S.l.: s.n.], 2017. p. 1942–1947. Citado 3 vezes nas páginas 15, 31 e 33.

ZADEH, L. Fuzzy sets. **Information and Control**, v. 8, n. 3, p. 338–353, 1965. ISSN 0019-9958. Citado na página 21.

ZANCHINI, V. A. A. **Criação de um modelo de classificação de tweets depressivos utilizando máquina de vetores de suporte**. Tese (Trabalho de Conclusão de Curso) — Universidade Federal de Uberlândia (UFU), Patos de Minas, MG, jul 2019. Citado 4 vezes nas páginas 14, 17, 30 e 33.

ZIMMERMAN, H. **Fuzzy Set Theory and Its Applications**. [S.l.]: Allied Publishers, 1996. ISBN 9788170235255. Citado 3 vezes nas páginas 8, 22 e 38.

Apêndices

APÊNDICE A – Árvore de Decisão para gerar Regras Fuzzy

Conforme comentado nos resultados, foi induzida uma Árvore de Decisão nos dados textuais pré-processados e, posteriormente, utilizada para a criação das regras do Sistema *Fuzzy*, fazendo parte da segunda solução do problema de classificação de textos escolares. A Figura 21 apresenta a árvore completa obtida.

De acordo com o que foi brevemente mencionado nos resultados, a árvore gerada representa os caminhos válidos para a decisão sobre os textos, desde a sua raiz, sendo o primeiro ponto de decisão, até suas folhas, sinalizando a decisão final, conforme uma classe específica. Além disso, conforme a imagem indica, a tomada de decisão de qualquer um dos nós à esquerda representa uma resposta verdadeira (*true*, em inglês) e, à direita, falsa (*false*, em inglês).

No entanto, os testes realizados em cada um dos nós não apresentam explicitamente os valores reais das variáveis linguísticas 'baixo' ou 'alto', apenas um valor numérico que particiona estes valores. Logo, quando o valor estiver abaixo de 0.5, a variável assume um peso baixo e, a partir de (maior ou igual a) 0.5, um peso alto, dado a aplicação da possibilidade de haver apenas dois valores linguísticos para cada variável dos antecedentes.

Finalmente, é importante observar que a árvore obtida foi configurada manualmente apenas com o hiperparâmetro de profundida máxima com 6 níveis, enquanto que os demais se mantiveram conforme o padrão usado pela biblioteca *Scikit-Learn* do *Python*.

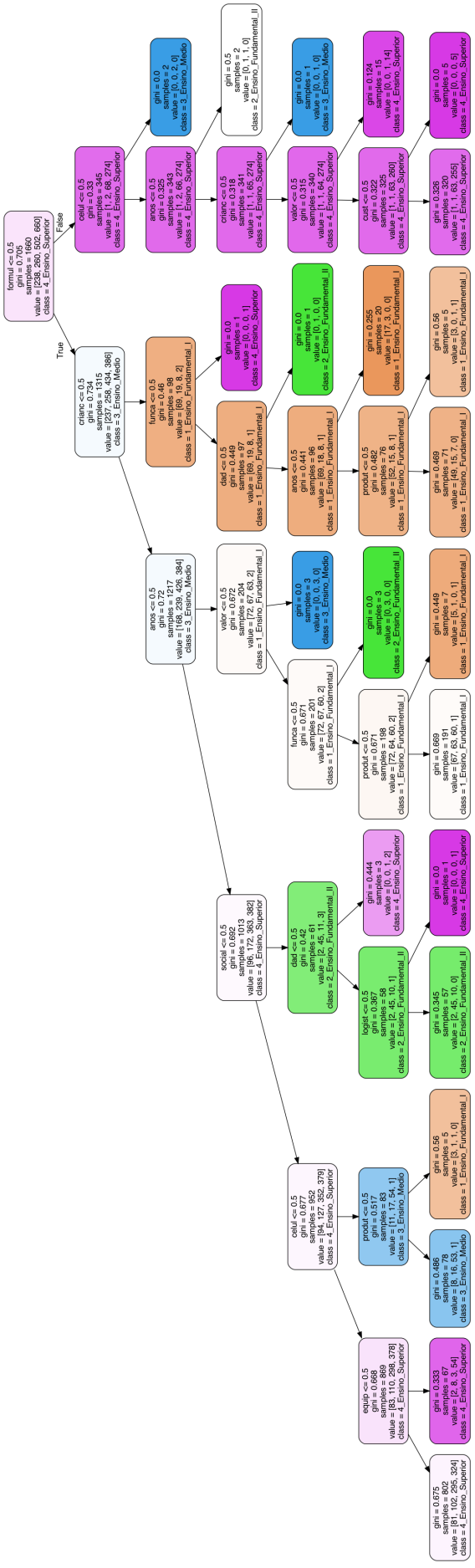


Figura 21 – Árvore de decisão completa com as regras *fuzzy*. Fonte: Autoria própria.

APÊNDICE B – Palavras Irrelevantes removidas durante o pré-processamento Textual

Segue a listagem completa das palavras removidas no segundo processo de limpeza textual:

'abaixo', 'acima', 'acordo', 'ainda', 'alem', 'alguem', 'algum', 'alguma', 'algumas', 'alguns', 'anterior', 'antes', 'apenas', 'aquela', 'aquelas', 'aquele', 'aqueles', 'aqui', 'aquilo', 'assim', 'atraves', 'atual', 'atualmente', 'baixa', 'baixo', 'bem', 'cada', 'caso', 'cima', 'cinco', 'coisa', 'coisas', 'conta', 'dado', 'define', 'definem', 'definicao', 'definida', 'definidas', 'definido', 'definidos', 'definir', 'depois', 'desde', 'deste', 'destes', 'desta', 'destas', 'desse', 'desses', 'dessa', 'dessas', 'deve', 'devem', 'dez', 'diagrama', 'disso', 'dois', 'duas', 'durante', 'ela', 'elas', 'ele', 'eles', 'enquanto', 'entao', 'entretanto', 'este', 'estes', 'esta', 'estas', 'estao', 'esse', 'esses', 'essa', 'essas', 'eu', 'esta', 'estando', 'estao', 'estar', 'estava', 'estavam', 'exemplo', 'explicacao', 'fato', 'faz', 'fazem', 'fazer', 'feito', 'fica', 'ficam', 'ficado', 'ficando', 'ficar', 'figura', 'forma', 'gente', 'geral', 'geralmente', 'grande', 'grandes', 'hoje', 'imagem', 'isto', 'isso', 'lado', 'logo', 'mal', 'meio', 'mesma', 'mesmas', 'mesmo', 'mesmos', 'muita', 'muitas', 'muito', 'muitos', 'mundo', 'nada', 'nao', 'necessaria', 'necessariamente', 'necessarias', 'necessario', 'necessarios', 'nessa', 'nessas', 'nesse', 'nesses', 'nesta', 'nestas', 'neste', 'nestes', 'nos', 'nove', 'nunca', 'oito', 'outra', 'outras', 'onde', 'outro', 'outros', 'parte', 'partir', 'pequena', 'pequenas', 'pequeno', 'pequenos', 'permite', 'permitem', 'permitindo', 'permitir', 'pessoa', 'pessoas', 'pode', 'podem', 'podemos', 'podendo', 'pois', 'por', 'porem', 'porque', 'pouca', 'poucas', 'pouco', 'poucos', 'primeira', 'primeiras', 'primeiro', 'primeiros', 'qualquer', 'quase', 'quando', 'quanto', 'quatro', 'que', 'sao', 'seguinte', 'segundo', 'seis', 'sempre', 'sendo', 'ser', 'sera', 'serao', 'serem', 'serie', 'sete', 'sido', 'sim', 'somente', 'sobre', 'tabela', 'tais', 'talvez', 'tambem', 'tanto', 'tal', 'ter', 'tem', 'tendo', 'tido', 'toda', 'todas', 'todo', 'todos', 'tudo', 'tres', 'tu', 'um', 'uma', 'umas', 'uns', 'usa', 'usam', 'usando', 'usar', 'utiliza', 'utilizam', 'utilizando', 'utilizar', 'vai', 'varias', 'varios', 'vez', 'vezes', 'voce', 'voces', 'vos', 'vossa', 'vossas' e 'vou'.

APÊNDICE C – Resultados completos da Validação Cruzada no Conjunto de Treinamento

Na Tabela 8, é possível visualizar o desempenho de cada classificador nas 10 diferentes partições dos dados de treinamento, e média das partições. É importante salientar que estes resultados foram obtidos a partir do processo de amostragem de validação cruzada (CV), este sendo executado apenas sobre os dados de treinamento, antes mesmo das fases de ajuste de hiperparâmetros, também no conjunto de treino, e avaliação de modelos, estes aplicados ao conjunto de testes.

Tabela 8 – Desempenho de cada classificador nas partições de CV em termos da BAC (autoria própria).

	Partição										
Algo.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Média**
MLP	0.8661	0.8807	0.9064	0.8522	0.9144	0.9579	0.9555	0.9133	0.9060	0.8816	0.9034
SGD	0.8788	0.8629	0.8983	0.8767	0.9180	0.9230	0.9597	0.8919	0.8971	0.8999	0.9006
SVM	0.8773	0.8643	0.8788	0.8391	0.8664	0.9082	0.9277	0.8774	0.8617	0.8746	0.8775
RL	0.8776	0.8844	0.8475	0.8244	0.856	0.8900	0.9213	0.8642	0.8223	0.8728	0.8660
K-NN	0.8360	0.8031	0.8312	0.7672	0.8632	0.8843	0.8643	0.8009	0.7843	0.7868	0.8221
NB. Bern.	0.7684	0.7844	0.6937	0.8231	0.8046	0.7957	0.8290	0.8139	0.7862	0.7691	0.7868
RF	0.7473	0.8348	0.7653	0.7509	0.7905	0.7569	0.8025	0.7767	0.7699	0.7600	0.7755
DT	0.6754	0.6698	0.7105	0.7011	0.6373	0.7041	0.7629	0.6559	0.6667	0.7082	0.6892
NB. Multi.	0.6474	0.6651	0.5933	0.6121	0.7003	0.5973	0.6094	0.6383	0.5525	0.6370	0.6253
Fuzzy	0.5264	0.4941	0.5035	0.4682	0.5200	0.4974	0.4834	0.4300	0.4433	0.4780	0.4844
Major.	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500
Minor.	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500
Random	0.1625	0.2592	0.2287	0.2316	0.2478	0.2520	0.2468	0.2242	0.2488	0.2656	0.2367