UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LETÍCIA TAVARES DA SILVA

BAMPORT: CONSTRUÇÃO DE UMA BASE DE DADOS MULTIMODAL PARA ANÁLISE DE MÚSICAS EM PORTUGUÊS

RIO DE JANEIRO
2024

LETÍCIA TAVARES DA SILVA

BAMPORT: CONSTRUÇÃO DE UMA BASE DE DADOS MULTIMODAL PARA ANÁLISE DE MÚSICAS EM PORTUGUÊS

Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. João Carlos Pereira da Silva

RIO DE JANEIRO
2024

CIP - Catalogação na Publicação

S586b

Silva, Letícia Tavares da
BAMPORT: construção de uma base de dados multimodal para análise de músicas em português / Letícia Tavares da Silva. -- Rio de Janeiro, 2024.
108 f.

Orientador: João Carlos Pereira da Silva.
Trabalho de conclusão de curso (graduação) - Universidade Federal do Rio de Janeiro, Instituto de Computação, Bacharel em Ciência da Computação, 2024.

1. Classificação automática de gêneros musicais. 2. Banco de dados de músicas. 3. Aprendizado de máquina. 4. Processamento de linguagem natural. I. Silva, João Carlos Pereira da, orient. II. Título.

Elaborado pelo Sistema de Geração Automática da UFRJ com os dados fornecidos pelo(a) autor(a), sob a responsabilidade de Miguel Romeu Amorim Neto - CRB-7/6283.

LETÍCIA TAVARES DA SILVA

BAMPORT: CONSTRUÇÃO DE UMA BASE DE DADOS MULTIMODAL PARA ANÁLISE DE MÚSICAS EM PORTUGUÊS

Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 03 de Setembro de 2024

BANCA EXAMINADORA:

gov.br Documento assinado digitalmente
JOAO CARLOS PEREIRA DA SILVA
Data: 09/09/2024 15:57:43-0300
Verifique em https://validar.iti.gov.br

João Carlos Pereira da Silva
D.Sc. (Instituto de Computação - UFRJ)

gov.br Documento assinado digitalmente
GISELI RABELLO LOPES
Data: 09/09/2024 18:19:48-0300
Verifique em https://validar.iti.gov.br

Giseli Rabello Lopes
D.Sc. (Instituto de Computação - UFRJ)

gov.br Documento assinado digitalmente
JOAO ANTONIO RECIO DA PAIXAO
Data: 09/09/2024 19:16:25-0300
Verifique em https://validar.iti.gov.br

João Antonio Recio Paixão
D.Sc. (Instituto de Computação - UFRJ)

Dedico esta pesquisa a todos que me apoiaram e inspiraram ao longo deste trabalho e de toda a minha trajetória. Em especial, agradeço a meus pais, Danielle Pinheiro e Lenilson Machado, pelo amor e encorajamento incondicional. Sou profundamente grata à minha amiga Aline Rezende, cujo incentivo e apoio foram essenciais, e ao meu amigo Gilberto Lopes, que esteve ao meu lado durante grande parte desta jornada. Dedico também a todos os meus amigos que torceram por mim, com um carinho especial para Larah Tavares, Marina Japiassú, Nina Sofia Salomon, Wellington Cadinelli e aos meus amigos da "Patotinha", que me ofereceram palavras de ânimo e apoio. Por fim, dedico um carinho especial aos meus amados gatinhos Murilo, Hannah e meu eterno Chinguinho, que foram uma fonte constante de conforto e alegria nos momentos mais difíceis. Esta pesquisa é para todos que, assim como eu, encontram na música a sua mais sincera forma de viver e sentir.

AGRADECIMENTOS

Agradeço ao meu orientador, professor João Carlos Pereira da Silva, do Instituto de Computação da UFRJ, por sua paciência, dedicação e contribuições ao desenvolvimento deste trabalho de conclusão de curso. Sua crença em mim e no projeto foi fundamental. Agradeço também ao aluno Gilberto Lopes pelo apoio na elaboração e avaliação de códigos e textos ao longo de todo o processo. Sou grata a todo o corpo docente do Instituto de Computação da UFRJ pela formação recebida, especialmente ao professor João Antonio Paixão, que despertou meu interesse por dados e, como membro da banca examinadora, contribuiu com valiosos comentários que melhoraram a qualidade final deste trabalho, assim como a professora Giseli Rabello, cujas observações também foram importantes para o aprimoramento dessa monografia. Agradeço à minha família pelo constante apoio e paciência durante minha trajetória, com um agradecimento especial aos meus pais, Danielle Pinheiro e Lenilson Machado. Um enorme agradecimento aos amigos e amigas que acompanharam minha jornada. Em especial, agradeço à Aline Rezende por ser um grande porto seguro ao longo desse trabalho.

"After silence, that which comes nearest to expressing the inexpressible is music."

Aldous Huxley

RESUMO

A música, muito presente no cotidiano das pessoas, tornou-se ainda mais acessível com a popularização das plataformas digitais, o que gerou uma crescente demanda por métodos eficazes de classificação automática de gêneros musicais. Este trabalho apresenta a base de dados BAMPORT, uma base robusta projetada para integrar características textuais e acústicas de músicas em português, com o objetivo de aprimorar a precisão na categorização musical. A BAMPORT resultou na criação de um conjunto de dados abrangente, que inclui 198 atributos e 27.777 instâncias. Esses atributos são compostos por 13 atributos de metadados, 14 métricas de áudio, 25 rótulos de gênero e 146 variáveis de Processamento de Linguagem Natural extraídas das letras das músicas. A análise inicial demonstrou que a combinação dos atributos textuais mais relevantes com as métricas de áudio resultou em um aumento significativo na pontuação F1-Macro dos modelos de classificação, em comparação com a utilização exclusiva das métricas de áudio, com a rede neural se destacando como o modelo de melhor desempenho. No entanto, esse aumento foi observado apenas quando a base foi filtrada para incluir os cinco gêneros nacionais mais populares. Esse resultado pode ser atribuído ao desbalanceamento entre gêneros e à presença reduzida de alguns gêneros na base de dados, o que pode ter limitado a capacidade dos modelos de aprender características distintas de gêneros menos representados.

Palavras-chave: base de dados; classificação automática de gêneros musicais; processamento de linguagem natural; métricas acústicas; letras de músicas; modelos de aprendizado de máquina; análise de sentimentos; python.

ABSTRACT

Music, a prominent part of everyday life, has become even more accessible with the rise of digital platforms, leading to an increased demand for effective methods of automatic genre classification. This work presents the BAMPORT dataset, a robust database designed to integrate textual and acoustic features of Portuguese-language music with the goal of improving accuracy in musical categorization. BAMPORT resulted in the creation of a comprehensive dataset, which includes 198 attributes and 27,777 instances. These attributes consist of 13 metadata attributes, 14 audio metrics, 25 genre labels, and 146 Natural Language Processing (NLP) variables extracted from song lyrics. Initial analysis demonstrated that combining the most relevant textual attributes with audio metrics resulted in a significant increase in the F1-Macro score of classification models, compared to using audio metrics alone, with neural networks standing out as the best-performing model. However, this improvement was observed only when the dataset was filtered to include the five most popular national genres. This result may be attributed to the imbalance between genres and the reduced presence of some genres in the dataset, which might have limited the models' ability to learn distinctive features of less represented genres.

Keywords: database; automatic genre classification; natural language processing; acoustic metrics; song lyrics; machine learning models; sentiment analysis; python.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CSV	Comma-separated Values
ER	Entidade-Relacionamento
ET	Extra Trees (Árvores Extramamente Aleatórias)
GPU	Graphics processing unit
kNN	K-Nearest Neighbors (k-Vizinhos Mais Próximos)
LDA	Latent Dirichlet Allocation
OVO	One-vs-One
PLN	Processamento de Linguagem Natural
RBF	Radial Basis Function
RF	Random Forest (Floresta Aleatória)
RNA	Rede Neural Artificial
SELU	Scaled Exponential Linear Unit
SQL	Structured Query Language
SVM	Support Vector Machine (Máquina de Vetores de Suporte)
TF-IDF	Term Frequency-Inverse Document Frequency

SUMÁRIO

1 INTRODUÇÃO

A música desempenha um papel significativo na vida cotidiana, acompanhando as pessoas em diversos contextos, desde ambientes domésticos e de trabalho até momentos de lazer e atividades físicas. Com o surgimento de novas tecnologias e plataformas digitais, o acesso à música tornou-se mais fácil e imediato, permitindo aos indivíduos desfrutar de uma vasta gama de conteúdos musicais a qualquer momento e em qualquer lugar.

Nesse cenário, a classificação automática de músicas surge como uma ferramenta importante, pois ela não apenas possibilita recomendações personalizadas, ajustadas aos gostos individuais, como também oferece recursos para uma melhor organização das bibliotecas e *playlists* ao possibilitar o agrupamento de músicas por atributos como gênero e humor.

Classificar gêneros musicais é um desafio complexo, amplificado pelo crescimento do conteúdo musical disponível na *internet*. A urgência por métodos eficazes para organizar e gerenciar essas coleções é cada vez maior. Técnicas de análise e classificação são fundamentais para categorizar músicas em gêneros distintos, como pop, rock ou samba. Contudo, essa tarefa é dificultada pela sobreposição de características entre gêneros, como ritmo e instrumentos, o que pode gerar variações nas classificações feitas por especialistas (CHIEN; LEE; YANG, 2018).

A classificação automática de gêneros musicais é uma área que começou a ganhar destaque a partir de trabalhos como o de Wold et al. (1996), que explorou métodos de classificação e recuperação de áudio com base no conteúdo. Enquanto os humanos utilizam audição e processos cognitivos, a classificação automática recorre a algoritmos para realizar essa tarefa.

Com o avanço das técnicas de aprendizado de máquina e a crescente demanda por maior precisão na categorização musical, modelos aprimorados têm sido desenvolvidos para enfrentar esses desafios. Um exemplo é o estudo de Guo, Gu e Liu (2017) que propôs a utilização de técnicas de aprendizado de máquina para a classificação de gêneros musicais utilizando atributos de áudio. O trabalho demonstrou uma eficácia em classificar 16 gêneros, destacando a necessidade de mais dados para melhorar ainda mais a precisão.

Além das métricas acústicas, as letras das músicas, que são amplamente acessíveis *online* e comuns na maioria das músicas populares, oferecem uma fonte de informação adicional. A análise das letras, utilizando ferramentas de Processamento de Linguagem Natural, permite a extração de aspectos textuais que podem complementar a compreensão das características musicais.

Um exemplo de como as letras podem ser utilizadas é o trabalho de Fell e Sporleder (2014), que introduziu um modelo baseado em n-gramas para a classificação de músicas. Este modelo analisa a frequência de diferentes frases de comprimento n e é complemen-

tado por uma abordagem multifacetada que avalia cinco aspectos distintos das letras: vocabulário, estilo, semântica, orientação e estrutura musical. A eficácia desses modelos demonstrou variações dependendo do gênero das letras analisadas, evidenciando a complexidade e a evolução contínua na área de classificação automática de gêneros musicais.

Outro estudo relevante, Mayerl et al. (2020) comparou diversas abordagens de extração de características textuais de músicas em inglês, incluindo métricas semânticas e estilísticas. Os resultados indicaram que características semânticas foram particularmente eficazes para identificar padrões de gêneros, com a combinação de múltiplas características melhorando a precisão da classificação.

No que diz respeito às letras em português, Oliveira e Filho (2023) investigou a atribuição de rótulos a gêneros musicais, destacando os desafios da subjetividade e diversidade das obras. Este trabalho propôs um sistema de classificação de letras de músicas em português, explorando tanto modelos avançados, como Redes Neurais, quanto técnicas mais simples, como a Regressão Logística. Os resultados mostraram que modelos mais avançados obtiveram melhor desempenho, alcançando uma acurácia de 61.6% na classificação de dez gêneros musicais.

Além da investigação da eficácia de modelos, no contexto da classificação de gêneros musicais, a construção de uma base de dados abrangente é essencial para capturar as variações e sutilezas entre diferentes gêneros, o que melhora a capacidade dos modelos de reconhecer padrões e fazer previsões precisas. Um exemplo dessa abordagem é o trabalho de Zangerle et al. (2018), que se concentrou na criação de uma base de dados para a classificação automática de músicas em inglês. Eles propuseram um método de construção de variáveis que combina características textuais e acústicas, utilizando métricas de áudio obtidas através da API do Spotify[1], letras de músicas obtidas do Genius[2] e tags para rótulos de gêneros obtidas do Last.fm[3]. Os resultados demonstraram que a combinação de múltiplas fontes de dados, aliada ao uso de técnicas avançadas de processamento de linguagem natural e análise de áudio, elevou significativamente a precisão da classificação. Isso ressalta a importância de uma base de dados bem estruturada para alcançar resultados eficientes em tarefas de classificação musical.

Inspirado por pesquisas em classificação automática de gêneros musicais, este trabalho foca no desenvolvimento da base de dados BAMPORT (Base de Atributos de Músicas em PORTuguês). A BAMPORT oferece uma abordagem integrada ao combinar métricas de áudio com letras de músicas em português. Além disso, o estudo constrói um conjunto de dados a partir da base de dados que não apenas reúne métricas acústicas, mas também inclui variáveis construídas a partir da análise das letras das músicas, como a frequência de termos, proporção de palavras ofensivas e uso de gírias. Essa integração entre os aspectos

[1] Spotify disponível em: https://open.spotify.com/.
[2] Genius disponível em: https://genius.com/.
[3] Last.fm disponível em: https://www.Last.fm/.

linguísticos e acústicos das canções amplia as possibilidades de análise e classificação. A contribuição deste trabalho se estrutura em três partes principais:

1. **Criação da Base de Dados BAMPORT**: O primeira contribuição é uma base de dados robusta que integra tanto métricas acústicas quanto textuais das músicas. A BAMPORT inclui um conjunto abrangente de 27.777 instâncias, combinando métricas de áudio com letras e tags de músicas. Esta base de dados serve como um recurso para a análise dos gêneros musicais em português.

2. **Desenvolvimento de um Conjunto de Dados Final**: A segunda contribuição foi a criação de um conjunto de dados final com 27.777 instâncias e 198 atributos que une os metadados das músicas, as métricas de áudio e 146 atributos construídos a partir das letras que capturam aspectos léxicos, linguísticos, semânticos e sintáticos das letras das músicas. Esses atributos foram desenhadas para complementar as métricas acústicas e fornecer uma visão detalhada dos textos musicais. O intuito foi explorar as nuances linguísticas presentes nas letras e como elas podem influenciar a categorização dos gêneros musicais.

3. **Avaliação do Impacto dos Atributos Textuais na Classificação Automática**: A terceira parte da contribuição envolve a investigação do impacto dos atributos textuais na precisão da classificação automática. Nosso estudo avalia se a integração das variáveis textuais com as métricas de áudio pode melhorar o desempenho dos modelos de classificação.

Nossa base de dados, disponibilizada para futuras pesquisas no repositório do Github[4], permite a replicação de experimentos e a validação dos resultados, promovendo a transparência e a colaboração na comunidade científica. Para garantir a consistência dos dados, empregamos técnicas de Emparelhamento de Dados, seguindo os métodos descritos por Bilenko (2003) e Mastub e Lacerda (2021).

Apesar dos progressos na área, muitos modelos existentes focam predominantemente em músicas em inglês. A criação de uma base de dados específica para músicas em português permite uma análise mais detalhada das características linguísticas e acústicas únicas desse idioma. Nos baseamos nos estudos de Mayerl et al. (2020), que utilizam variáveis derivadas do banco de dados ALF200k[5], e nas metodologias descritas por Zangerle et al. (2018), que demonstram a eficácia da integração de características textuais e acústicas na melhoria da precisão da classificação musical. Aplicar essas metodologias a um corpus de músicas em português oferece novas oportunidades para explorar nuances linguísticas e culturais específicas do idioma.

[4] Repositório do Trabalho disponível em: https://github.com/leticiatavaresds/BAMPORT.
[5] ALF200K disponível em: https://github.com/dbis-uibk/ALF200k/.

Para avaliar a combinação dos atributos textuais criados com as métricas de áudio, adotamos uma abordagem comparativa utilizando cinco modelos distintos: Floresta Aleatória (RF), Árvores Extremamente Aleatórias (ET), k-Vizinhos Mais Próximos (kNN), Rede Neural Artificial (RNA) e Máquina de Vetores de Suporte (SVM). Essa estratégia visa proporcionar uma visão abrangente do desempenho dos diferentes algoritmos na nossa base de dados, permitindo uma análise mais completa e alinhada com as metodologias dos estudos de referência.

O restante do trabalho está estruturado da seguinte maneira. Inicialmente, o Capítulo 2 apresenta conceitos básicos sobre aprendizado de máquina, com uma ênfase nas abordagens de aprendizado supervisionado, fundamental para o desenvolvimento deste projeto. Em seguida, o Capítulo 3 explora as ferramentas de Processamento de Linguagem Natural utilizadas na análise musical, detalhando como essas ferramentas foram selecionadas e aplicadas para a construção das variáveis analíticas.

No Capítulo 4, discutimos o projeto e a composição da base de dados, cobrindo todo o processo desde a coleta de dados em várias fontes até a identificação de registros que correspondem às mesmas músicas provenientes de diferentes fontes e a elaboração e integração desses dados no banco de dados

No Capítulo 5, analisamos a construção das variáveis de Processamento de Linguagem Natural, destacando como elas são usadas para caracterizar e analisar as letras das músicas. O Capítulo 6 apresenta os resultados obtidos dos experimentos, discutindo em detalhe a configuração dos modelos e os resultados encontrados, com um foco particular na configuração que alcançou o melhor desempenho.

Finalmente, o Capítulo 7 conclui o projeto com uma reflexão sobre as dificuldades enfrentadas, as principais descobertas e sugestões para futuras pesquisas, consolidando as principais lições aprendidas e propondo direções para trabalhos futuros.

2 APRENDIZADO DE MÁQUINA

O aprendizado de máquina é um campo da inteligência artificial que se concentra no desenvolvimento de modelos capazes de aprender e se adaptar automaticamente a partir da detecção automatizada de padrões em dados, sem serem explicitamente programados (DOMINGOS, 2012).

Existem diversas abordagens no aprendizado de máquina, cada uma com suas características e aplicações específicas. No entanto, para manter o foco e a clareza deste trabalho, daremos ênfase ao aprendizado supervisionado e aprendizado por comitês, que serão detalhados nas seções seguintes.

2.1 APRENDIZADO SUPERVISIONADO

O aprendizado supervisionado é um método onde o algoritmo é treinado usando um conjunto de dados rotulados, ou seja, dados de entrada associados às saídas desejadas (MITCHELL, 1997). Este tipo de técnica é amplamente utilizado para problemas de classificação e regressão, onde o objetivo é prever a saída correta para novas entradas com base nos exemplos fornecidos durante o treinamento. O aprendizado supervisionado já se mostrou uma solução viável para problemas em diversas áreas, incluindo reconhecimento de imagem, processamento de fala e áudio, processamento de linguagem natural, diagnóstico médico, previsão financeira e sistemas de recomendação (DOMINGOS, 2012).

Apesar de sua utilidade e clareza, o aprendizado supervisionado enfrenta alguns desafios. A principal dificuldade é a necessidade de conjuntos de dados rotulados de alta qualidade. A eficácia dos modelos depende da disponibilidade de dados representativos e bem rotulados. Dados insuficientes ou não representativos podem levar a um problema conhecido como *underfitting*, no qual o modelo falha em capturar e generalizar adequadamente os padrões presentes nos dados. Isso ocorre porque o modelo não tem informações suficientes ou diversificadas para aprender padrões significativos. Em contraste, um modelo pode também sofrer de *overfitting*, onde se ajusta excessivamente aos dados de treinamento, comprometendo sua capacidade de generalizar para novos dados (DOMINGOS, 2012).

2.2 APRENDIZADO BASEADO EM COMITÊS

O aprendizado baseado em comitês é um paradigma de aprendizado de máquina onde múltiplos aprendizes são treinados para resolver o mesmo problema (DIETTERICH, 2000). Estes múltiplos aprendizes são chamados de aprendizes base e podem ser homogêneos, usando o mesmo algoritmo, ou heterogêneos, empregando algoritmos diferentes.

Nessa abordagem, um conjunto de hipóteses é construído e combinado para atingir um objetivo comum.

Geralmente, um comitê começa a ser construído quando os aprendizes base são produzidos. Esta produção pode acontecer de forma sequencial ou paralela. Na forma sequencial, cada aprendiz gerado tem influência na geração de aprendiz seguinte, enquanto na forma paralela todos são gerados "ao mesmo tempo", sem interconexão entre aprendizes. Em seguida, os aprendizes gerados são combinados. Para gerar essa combinação, as técnicas mais populares são votação, para classificação, e média ponderada, para regressão (CARUANA; NICULESCU-MIZIL, 2006).

Para conseguir um bom comitê, analogamente à vida real, os aprendizes base devem ter a maior acurácia e diversidade possível. Existem algumas técnicas para estimar a acurácia dos aprendizes, como a validação cruzada. No entanto, para diversidade, não é possível ainda definir o que é um conjunto diverso. Tentativas de medir a diversidade não têm comprovação definitiva, mas ela pode ser promovida através da seleção de dados, controle de atributos, introdução de aleatoriedade ou combinações dessas abordagens (KUNCHEVA, 2002).

2.2.1 Florestas Aleatórias

As florestas aleatórias são uma técnica de aprendizado de máquina que tem ganhado destaque significativo devido à sua eficácia em lidar com problemas de classificação e regressão. Esta abordagem é uma extensão das árvores de decisão, projetada para mitigar o *overfitting* e melhorar a precisão das previsões através da combinação de múltiplas árvores de decisão (BREIMAN, 2001).

As árvores de decisão são algoritmos de aprendizado supervisionado não paramétricos que, a partir do nó raiz, criam ramos para nós internos que realizam testes em atributos específicos dos dados. Esses testes dividem os dados em conjuntos mais homogêneos até que os nós folha sejam alcançados, representando as classes de saída ou valores previstos para os dados de entrada (QUINLAN, 1986).

O processo de construção de uma árvore de decisão envolve uma busca gulosa para identificar os pontos de divisão ótimos em cada nó. Esse processo é repetido de forma recursiva e descendente até que todos os dados sejam classificados ou a maioria dos registros sejam atribuídos a rótulos de classes específicos (BREIMAN et al., 1984). O ponto de divisão é determinado com base em um critério de pureza que avalia a qualidade da divisão. Para problemas de classificação, a impureza de Gini é um critério comum, calculado como:

$$Gini(D) = 1 - \sum_{i=1}^{C} p_i^2 \tag{2.1}$$

onde p_i é a proporção de amostras pertencentes à classe i no conjunto de dados D. O atributo e o ponto de divisão que resultam na maior redução da impureza são escolhidos, e o conjunto de dados é dividido em dois subconjuntos. Esse processo é repetido recursivamente para cada novo nó até que uma condição de parada seja atingida.

Nas florestas aleatórias, a seleção de amostras para cada árvore é realizada por meio do método de *bootstrapping*, onde amostras são retiradas com reposição do conjunto de dados original. Além disso, em cada nó, um subconjunto aleatório de atributos é escolhido, em vez de considerar todos os atributos disponíveis. Essa seleção aleatória promove diversidade entre as árvores, tornando-as mais independentes umas das outras, pois cada árvore analisa diferentes variáveis para tomar suas decisões. Isso aumenta a robustez do modelo, reduz o *overfitting* e melhora a precisão das previsões.

2.2.2 Árvores Extremamente Aleatórias

Introduzidas por Geurts, Ernst e Wehenkel (2006), as Árvores Extremamente Aleatórias são uma extensão das Florestas Aleatórias com maior ênfase na aleatoriedade, resultando em possíveis melhorias de desempenho e eficiência em certos cenários. Assim como as Florestas Aleatórias, as Árvores Extremamente Aleatórias constroem múltiplas árvores de decisão de forma independente, e cada árvore contribui para a previsão final do modelo (BREIMAN, 2001). No entanto, as Árvores Extremamente Aleatórias adaptam as árvores de decisão para selecionarem um ponto de divisão de maneira completamente aleatória dentro do intervalo dos valores possíveis para cada atributo, acelerando o processo de construção e aumentando a diversidade entre as árvores. Isso pode ajudar a capturar melhor as características dos dados, especialmente em problemas com relações complexas e não lineares (GEURTS; ERNST; WEHENKEL, 2006).

Outra distinção é que as Árvores Extremamente Aleatórias utilizam o conjunto de treinamento completo para construir cada árvore, modelando mais eficientemente a diversidade presente nos dados e reduzindo o viés. No entanto, isso também pode aumentar a variância, o que precisa ser gerido durante o treinamento e a validação do modelo (GEURTS; ERNST; WEHENKEL, 2006).

2.3 MÁQUINAS DE VETORES DE SUPORTE

As Máquinas de Vetores de Suporte (SVM) são uma técnica em que o princípio central é encontrar um hiperplano ótimo que separa as classes de forma linear em um conjunto de dados, maximizando a margem entre elas (CORTES; VAPNIK, 1995). A margem é a distância entre o hiperplano e os exemplos mais próximos de cada classe, conhecidos como vetores de suporte. Uma maior margem tende a melhorar a capacidade de generalização do modelo para novos dados (CORTES; VAPNIK, 1995).

Os vetores de suporte são os pontos de dados mais críticos, pois estão mais próximos do hiperplano e têm um impacto direto na sua posição. A solução ideal do hiperplano é obtida minimizando a capacidade do modelo, ou seja, o número de parâmetros independentes que podem ser ajustados (BURGES, 1998).

Em um espaço bidimensional, a separação é feita por uma linha, enquanto em dimensões superiores, a separação é realizada por hiperplanos. A forma geral do hiperplano separador é dada por

$$w^T x + b = 0 \tag{2.2}$$

onde w é o vetor de pesos, x é o vetor de entrada e b é o viés. A distância d entre um ponto x e o hiperplano é calculada pela fórmula

$$d = \frac{|w^T x + b|}{\|w\|} \tag{2.3}$$

onde $\|w\|$ é a norma euclidiana do vetor de pesos w. O objetivo é maximizar a margem de separação, que corresponde à distância d entre o hiperplano e os pontos mais próximos. Para isso, é necessário minimizar a norma do vetor de pesos $\|w\|$, uma vez que a margem é inversamente proporcional a essa norma. Esse problema é formulado como um problema de programação quadrática, onde a função objetivo a ser minimizada é

$$\frac{1}{2}\|w\|^2 \tag{2.4}$$

sujeita a restrições que garantem a correta classificação de todos os pontos de dados (CORTES; VAPNIK, 1995).

Quando os dados não são linearmente separáveis no espaço original, o SVM utiliza funções de *kernel* para mapear os dados em um espaço dimensional superior onde a separação linear se torna possível. Um dos mais utilizados para isso é o *kernel* RBF, que lida com relações não lineares através de transformações baseadas em funções gaussianas (SCHöLKOPF; SMOLA, 2002a).

O parâmetro de regularização C do SVM, controla o equilíbrio entre a maximização da margem e a minimização do erro de classificação. Um valor alto de C resulta em um ajuste mais rigoroso aos dados de treinamento, podendo levar ao *overfitting*, enquanto um valor baixo favorece uma margem maior, podendo resultar em mais erros de classificação. Outro parâmetro importante é o parâmetro de ajuste do *kernel* γ, utilizado com *kernels* não lineares. O γ define a influência de um único ponto de dados no modelo; valores altos podem causar *overfitting*, enquanto valores baixos promovem uma influência mais global (SCHöLKOPF; SMOLA, 2002b).

Durante o treinamento, a função de perda utilizada é relevante. A *hinge loss* penaliza erros lineares e é comum em SVMs lineares para manter uma margem ampla, enquanto

a *squared hinge loss* penaliza erros quadraticamente, ajudando a refinar as decisões do modelo ao ser mais severa com grandes desvios (CORTES; VAPNIK, 1995).

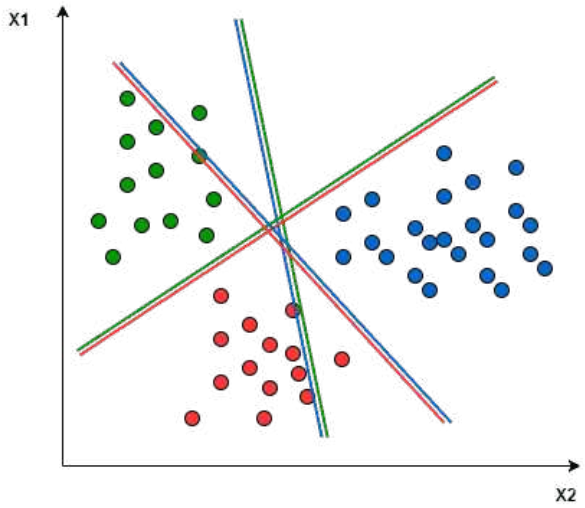

Figura 1 – Exemplo de classificação multiclasse usando o SVM com abordagem One-vs-One (OvO). Os pares de linhas coloridas representam os hiperplanos de decisão entre pares de classes de mesmas cores das linhas. Fonte: https://www.baeldung.com/cs/svm-multiclass-classification. Acesso em 02/08/2024.

Embora o SVM seja originalmente um algoritmo de classificação binária, ele pode ser adaptado para problemas de classificação multiclasse. Uma das abordagens principais é o "*one-vs-one*", onde o SVM é treinado para distinguir entre cada par de classes. Para um problema com K classes, são treinados $\frac{K\times(K-1)}{2}$ classificadores binários. Cada classificador decide entre duas classes específicas, e a classe que obtém o maior número de vitórias entre todos os classificadores binários é escolhida como a previsão final (RIPLEY, 1996).

A Figura 1 mostra um exemplo de classificação multiclasse usando SVM com a abordagem OvO. Temos três classes representadas por pontos vermelhos, verdes e azuis. Os pares de linhas coloridas representam os hiperplanos de decisão entre pares de classes, mostrando a fronteira de decisão onde um classificador binário decide entre duas classes específicas, formando uma rede de decisões que permite classificar os pontos no espaço.

2.4 K-VIZINHOS MAIS PRÓXIMOS

Baseado na premissa de que dados semelhantes tendem a estar próximos no espaço de características, o modelo k-Vizinhos Mais Próximos classifica novos dados com base na proximidade às amostras de treinamento (ALTMAN, 1992). Este método funciona de maneira intuitiva: durante o treinamento, o algoritmo armazena todas as amostras e, ao classificar uma nova amostra, calcula a distância entre ela e todas as amostras de treinamento usando uma métrica de distância específica (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

A quantidade de vizinhos considerados ao fazer uma previsão é um dos com mais impacto do kNN. Este parâmetro define quantos dos vizinhos mais próximos influenciam a decisão sobre a classe ou o valor da nova amostra. Em geral, um número menor de vizinhos pode fazer o modelo ser mais sensível a ruídos e *outliers*, enquanto um número maior tende a suavizar a decisão, promovendo uma maior generalização (SIDDIQI; SIDDIQI, 2005).

No processo de previsão, os k vizinhos mais próximos são selecionados com base na menor distância, e a classe ou valor predito é determinado pela maioria dos rótulos ou valores desses vizinhos (SIDDIQI; SIDDIQI, 2005). Em problemas de classificação, a classe mais frequente entre os vizinhos é atribuída à nova amostra, enquanto, na regressão, a previsão é feita com base na média ou mediana dos valores dos vizinhos (FRIEDMAN; HASTIE; TIBSHIRANI, 2001).

Uma técnica adotada pelo modelo é a ponderação das contribuições dos vizinhos. Em alguns casos, cada vizinho tem um impacto igual na decisão final. No entanto, pode ser mais eficaz ajustar a influência dos vizinhos com base na proximidade, dando maior peso aos vizinhos mais próximos, o que pode melhorar a precisão do modelo (SIDDIQI; SIDDIQI, 2005), para isso é realizada a ponderação das contribuições dos vizinhos.

Além disso, a escolha da métrica de distância utilizada no kNN pode afetar significativamente o desempenho do modelo. Métricas comuns incluem a distância Euclidiana, que calcula a raiz quadrada da soma das diferenças ao quadrado entre as coordenadas dos pontos, e a distância de Manhattan, que soma as diferenças absolutas entre as coordenadas (MANNING; RAGHAVAN; SCHüTZE, 2008). A distância de Minkowski, uma generalização dessas métricas, é parametrizada por um valor p, permitindo ajustar a sensibilidade do algoritmo (BISHOP, 2006).

O kNN oferece vantagens como simplicidade e flexibilidade, pois não faz suposições sobre a distribuição dos dados e é aplicável a uma ampla gama de problemas (ALTMAN, 1992). No entanto, o método apresenta desvantagens, como o elevado custo computacional em grandes conjuntos de dados, uma vez que cada nova amostra precisa ser comparada com todas as amostras de treinamento.

2.5 REDE NEURAL ARTIFICIAL

As redes neurais artificiais (RNAs) são algoritmos de aprendizado de máquina inspirados no funcionamento do cérebro humano, compostos por unidades chamadas neurônios artificiais, organizadas em camadas (LECUN; BENGIO; HINTON, 2015). A estrutura básica de uma rede neural inclui uma camada de entrada para receber os dados, uma ou mais camadas ocultas para processamento e uma camada de saída para geração das previsões finais. Quando uma rede neural possui muitas camadas ocultas, ela é denominada rede neural profunda.

O treinamento de redes neurais é estruturado em épocas, onde cada época representa

uma iteração completa sobre o conjunto de dados. Durante esse processo, os dados de entrada passam pela rede, e cada neurônio aplica uma função de ativação para gerar suas saídas. Em seguida, é calculado o valor da função de perda, que mede o erro entre as previsões da rede e os valores reais, fornecendo uma métrica quantitativa do desempenho do modelo. O objetivo é minimizar essa perda ajustando os pesos da rede para melhorar a precisão das previsões (LECUN; BENGIO; HINTON, 2015). Os gradientes da perda em relação aos pesos são então calculados, e os pesos são ajustados usando algoritmos de otimização, como o Gradiente Descendente. Esse ciclo é repetido por várias épocas até que a rede atinja um desempenho satisfatório.

Durante o treinamento de uma rede neural, tanto o número de épocas quanto o tamanho dos lotes de dados influenciam a eficiência e eficácia do processo. Cada época consiste em uma iteração completa através do conjunto de dados, e o número de épocas afeta diretamente o tempo de treinamento e o uso de recursos computacionais. Se o número de épocas for muito baixo, o modelo pode não ter tempo suficiente para aprender padrões significativos, resultando em *underfitting*. Em contraste, um número muito alto de épocas pode levar a *overfitting*.

Além disso, o tamanho dos lotes de dados impacta a convergência do modelo. Lotes menores permitem atualizações mais frequentes dos pesos, o que pode acelerar a convergência e capturar padrões sutis, mas também pode aumentar a variabilidade e o ruído. Por outro lado, lotes maiores proporcionam atualizações mais estáveis e consistentes, porém exigem mais memória e processamento, e podem ser mais suscetíveis ao *overfitting* (RUDER, 2016).

Os gradientes são usados para ajustar os pesos das conexões entre os neurônios para minimizar o erro nas previsões. No entanto, pode ocorrer o problema do desaparecimento ou explosão dos gradientes. O desaparecimento dos gradientes ocorre quando eles se tornam muito pequenos, impedindo a rede de aprender, enquanto a explosão dos gradientes acontece quando se tornam muito grandes, causando atualizações instáveis dos pesos e prejudicando o treinamento.

A capacidade de aprender representações não-lineares é um dos principais fatores que contribuem para a eficácia das redes neurais em aprender e generalizar padrões complexos (LECUN; BENGIO; HINTON, 2015). São as funções de ativação que adicionam não-linearidade às redes neurais, permitindo que elas modelem relações complexas entre variáveis, superando as limitações das relações lineares. Cada função de ativação possui características específicas que a tornam adequada para diferentes tipos de tarefas e estruturas de rede. A seguir, são descritas algumas das funções de ativação mais comuns, incluindo suas fórmulas e características:

- **Sigmoid**: Esta função mapeia qualquer valor real para um intervalo entre 0 e 1. A fórmula da função sigmoid é

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.5}$$

No entanto, pode sofrer com o problema do desvanecimento do gradiente quando os valores são muito grandes ou muito pequenos.

- **Tanh (Tangente Hiperbólica)**: Similar à função sigmoid, a função tangente hiperbólica mapeia valores para um intervalo entre -1 e 1. A fórmula é

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.6}$$

Esse intervalo centrado em torno de zero ajuda a reduzir o desvanecimento do gradiente, melhorando o treinamento das redes.

- **ReLU (Rectified Linear Unit)**: A função ReLU é conhecida por sua simplicidade e eficiência computacional. Ela transforma valores negativos em zero e mantém valores positivos inalterados. A fórmula é

$$\text{ReLU}(x) = \max(0, x) \tag{2.7}$$

No entanto, pode sofrer com o problema dos "neurônios mortos", onde alguns neurônios podem ficar inativos durante o treinamento.

- **SELU (Scaled Exponential Linear Unit)**: A função SELU é projetada para manter a média e a variância das ativações em uma rede neural, o que ajuda a estabilizar o treinamento. A fórmula é

$$\text{SELU}(x) = \lambda \begin{cases} x & \text{se } x > 0 \\ \alpha(e^{x} - 1) & \text{se } x \leq 0 \end{cases}, \tag{2.8}$$

onde λ e α são parâmetros que ajustam a escala e a forma da função.

Além de escolher a função de ativação adequada, técnicas adicionais podem ser aplicadas para melhorar o desempenho do treinamento das redes neurais. Uma dessas técnicas é o *AlphaDropout*, técnica que desativa aleatoriamente unidades durante o treinamento e ajusta as ativações para manter a distribuição estatística estável. Esta técnica é especialmente eficaz em redes que utilizam a função de ativação SELU, pois ajuda a garantir uma normalização e regularização eficazes. Ao manter a média e a variância das ativações constantes, o *AlphaDropout* contribui para uma melhor capacidade de generalização do modelo (SRIVASTAVA et al., 2014).

2.6 VALIDAÇÃO DOS MODELOS

No aprendizado de máquina, a validação dos modelos é aplicada visando garantir que eles generalizem bem para novos dados não vistos durante o treinamento. Entre os métodos mais comuns, a validação cruzada se destaca como uma técnica eficaz para estimar a performance do modelo (HASTIE; TIBSHIRANI; FRIEDMAN, 2009; KUHN; JOHNSON, 2013).

A validação cruzada divide aleatoriamente o conjunto de dados em k-partes chamados de dobras, onde uma dobra é utilizada para validação e as $k - 1$ dobras restantes são usadas para treinamento (KOHAVI, 1995).

A principal vantagem do procedimento é a redução do viés na estimativa da performance do modelo, uma vez que todos os dados disponíveis são utilizados tanto para treinamento quanto para validação (VARMA; SIMON, 2006). Isso proporciona uma avaliação mais robusta da capacidade de generalização do modelo, ajudando a identificar problemas de *overfitting* ou *underfitting*.

2.6.1 Métricas de Validação dos Modelos

Na avaliação de modelos de aprendizado de máquina, utilizam-se métricas que quantificam o desempenho do modelo em relação ao problema específico. Antes de apresentar as métricas, é importante entender as seguintes variáveis:

- VP (Verdadeiros Positivos): São as instâncias que o modelo previu corretamente como pertencentes à classe positiva. Por exemplo, se o modelo está classificando músicas como "rock", um verdadeiro positivo seria uma música que realmente é rock e foi corretamente identificada como rock pelo modelo.

- VN (Verdadeiros Negativos): São as instâncias que o modelo previu corretamente como pertencentes à classe negativa. No mesmo exemplo, um verdadeiro negativo seria uma música que não é rock e foi corretamente identificada como não sendo rock.

- FP (Falsos Positivos): São as instâncias que o modelo previu incorretamente como pertencentes à classe positiva. Neste caso, um falso positivo seria uma música que não é rock, mas foi erroneamente classificada como rock pelo modelo.

- FN (Falsos Negativos): São as instâncias que o modelo previu incorretamente como pertencentes à classe negativa. Um falso negativo seria uma música que é rock, mas foi erroneamente classificada como não sendo rock.

Abaixo, são descritas algumas das métricas mais comuns, juntamente com suas respectivas fórmulas:

- **Acurácia**: proporção de previsões corretas em relação ao total de previsões realizadas. Ela é calculada pela fórmula:

$$\text{Acurácia} = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.9}$$

- **Precisão**: proporção de previsões positivas corretas entre todas as previsões positivas realizadas pelo modelo. Essa métrica é importante em casos onde o custo de um falso positivo é alto, pois reflete a exatidão do modelo em identificar corretamente as instâncias positivas.

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{2.10}$$

- **Recall**: O *recall* é a proporção de previsões positivas corretas entre todas as instâncias que realmente são positivas.

$$\text{Recall} = \frac{VP}{VP + FN} \tag{2.11}$$

- **F1-Score**: é uma métrica que combina precisão e recall em uma única medida, usando a média harmônica entre as duas. Isso significa que o F1-*Score* oferece uma visão mais equilibrada do desempenho do modelo, especialmente em situações onde as classes estão desbalanceadas.

$$\text{F1-}Score = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}} \tag{2.12}$$

- **F1-Macro**: é a média do F1-*Score* calculado individualmente para cada classe, sem levar em consideração o tamanho das classes. Essa métrica é relevante quando se deseja avaliar o desempenho do modelo de forma equitativa em todas as classes.

$$X = \{\text{“de”, “e ”, “ l”, “la”, “ad”, “di”, “in”}\} \tag{2.13}$$

$$\text{F1-Macro} = \frac{1}{C} \sum_{i=1}^{C} F1_i \tag{2.14}$$

Onde C é o número total de classes e $F1_i$ é o F1-Score de cada classe i.

A escolha das métricas para avaliar nossos modelos, com destaque para o F1-Macro, foi motivada pelo desbalanceamento presente em nossa base de dados. Embora a acurácia seja uma métrica comum, ela tende a favorecer as classes majoritárias, o que seria inadequado para nosso conjunto de dados, onde algumas classes são muito mais representadas que outras. O F1-Score, que equilibra precisão e *recall*, é relevante quando há a necessidade

de considerar ambos os aspectos ao mesmo tempo. No entanto, o F1-Macro, que calcula o F1-Score para cada classe de forma independente do seu tamanho, é especialmente importante para nosso cenário, garantindo uma avaliação justa do desempenho do modelo em todas as classes, incluindo aquelas com menor representatividade.

2.7 INTEGRAÇÃO E CORRESPONDÊNCIA DE DADOS: O PAPEL DO PAREAMENTO DE DADOS

A qualidade dos dados é fundamental para o desempenho dos modelos de aprendizado de máquina, e o pareamento de dados, que une registros de diferentes conjuntos, desempenha um papel importante na construção de bases integradas e consistentes. Técnicas como algoritmos baseados em distância e métodos probabilísticos são usadas para resolver problemas de dados dispersos e incompletos, garantindo que a qualidade das entradas melhore a eficácia das previsões e classificações dos modelos (FELLEGI; SUNTER, 1969; HERZOG; SCHEUREN; BROWN, 2007). A abordagem adotada para o pareamento de dados pode ter um impacto significativo na qualidade das entradas para os modelos de aprendizado, influenciando diretamente a eficácia das previsões e classificações realizadas.

Abordagens determinísticas para definir se dois registros se referem à mesma música podem apresentar limitações, especialmente quando exigem correspondências exatas. Devido a variações na escrita e possíveis erros de digitação, essas abordagens podem não ser viáveis. Em contraste, métodos probabilísticos oferecem uma alternativa mais eficaz, calculando a probabilidade de correspondência entre registros e adaptando-se melhor às imperfeições nos dados.

A metodologia probabilística de pareamento de registros, inspirada nos trabalhos de Winkler (1999), utiliza as probabilidades m e u. A probabilidade m representa a chance de que um determinado par de registros seja uma correspondência verdadeira, enquanto u é a probabilidade de que o par não seja uma correspondência. Essas probabilidades são ajustadas com base na presença de erros e na incompletude dos dados.

2.7.1 Coeficiente de Sørensen-Dice

O coeficiente de Sørensen-Dice, desenvolvido a partir dos trabalhos de Sørensen (1948) e Dice Dice (1945), é uma medida estatística usada para determinar a similaridade entre dois conjuntos de dados. Este coeficiente é especialmente útil na comparação de strings através da análise de bigramas. A fórmula para calcular o coeficiente é:

$$DSC\,(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{2.15}$$

onde X e Y são conjuntos de bigramas, que são pares de caracteres adjacentes dentro das strings, extraídos a serem comparadas. Para ilustrar a comparação de cadeias de caracteres no contexto musical, podemos considerar as strings “de ladin” e “de ladinho”,

para aplicar a fórmula do coeficiente de Sørensen-Dice aos bigramas, identificamos os bigramas únicos em cada *string*. Vamos definir X como o conjunto de bigramas da *string* "de ladin" e Y como o conjunto de bigramas da *string* "de ladinho". Os conjuntos serão:

$$X = \{\text{“de”, “e ”, “ l”, “la”, “ad”, “di”, “in”}\} \tag{2.16}$$

$$Y = \{\text{“de”, “e ”, “ l”, “la”, “ad”, “di”, “in”, “nh”, “ho”}\} \tag{2.17}$$

$$X \cap Y = \{\text{“de”, “e ”, “ l”, “la”, “ad”, “di”, “in”}\} \tag{2.18}$$

Aqui, $|X|$ é o número de bigramas únicos em "de ladin", $|Y|$ é o número de bigramas únicos em "de ladinho", e $|X \cap Y|$ é o número de bigramas compartilhados entre as duas strings.

Aplicando a fórmula do coeficiente de Sørensen-Dice:

$$DSC(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{2.19}$$

Substituindo os valores:

$$|X| = 7 \tag{2.20}$$

$$|Y| = 9 \tag{2.21}$$

$$|X \cap Y| = 7 \tag{2.22}$$

Portanto:

$$DSC(X, Y) = \frac{2 \times 7}{7 + 9} = \frac{14}{16} = 0.875 \tag{2.23}$$

Assim, o coeficiente de Sørensen-Dice entre as strings "de ladin" e "de ladinho" é aproximadamente 0.875, indicando uma alta similaridade entre as duas strings.

2.7.2 Algoritmo de Damerau-Levenshtein

O algoritmo de Damerau-Levenshtein é uma extensão do tradicional algoritmo de Levenshtein que calcula a distância de edição entre duas strings. Esta distância é definida como o número mínimo de operações necessárias para transformar uma *string* em outra, onde as operações permitidas são inserção, exclusão, substituição de um único caractere e, adicionalmente, a transposição de dois caracteres adjacentes. A inclusão da operação de transposição torna o algoritmo de Damerau-Levenshtein particularmente eficaz na correção de erros comuns de digitação, tornando-o uma excelente escolha para cálculos

de similaridade entre strings (LOWRANCE; WAGNER, 1975; BRILL; MOORE, 2000; NAVARRO, 2001).

Vamos novamente utilizar as strings “de ladin” e “de ladinho” para ilustrar o cálculo da distância de Damerau-Levenshtein.

1. **Inicialização**: Criamos uma matriz onde o número de linhas é igual ao comprimento da primeira *string* mais um, e o número de colunas é igual ao comprimento da segunda *string* mais um. Inicializamos a primeira linha e a primeira coluna com índices crescentes.

2. **Preenchimento**: Preenchemos a matriz considerando os custos de inserção, exclusão, substituição e transposição. Cada célula (i,j) na matriz contém o custo mínimo para transformar o prefixo da primeira *string* (até o i-ésimo caractere) no prefixo da segunda *string* (até o j-ésimo caractere), sendo que custo mínimo é 0 caso os caracteres nas posições i e j sejam iguais.

Após preenchimento, a matriz de cálculo da distância de Damerau-Levenshtein para as strings “de ladin” e “de ladinho” é apresentada abaixo:

		d	*e*		*l*	*a*	*d*	*i*	*n*	*h*	*o*
	0	1	2	3	4	5	6	7	8	9	10
d	1	0	1	2	3	4	5	6	7	8	9
e	2	1	0	1	2	3	4	5	6	7	8
	3	2	1	0	1	2	3	4	5	6	7
l	4	3	2	1	0	1	2	3	4	5	6
a	5	4	3	2	1	0	1	2	3	4	5
d	6	5	4	3	2	1	0	1	2	3	4
i	7	6	5	4	3	2	1	0	1	2	3
n	8	7	6	5	4	3	2	1	0	1	2

O valor na célula inferior direita da matriz é 2, o que significa que a distância de edição entre “de ladin” e “de ladinho” é 2. Isso indica que são necessárias 2 operações, a inserção de “h” e “o” no final da *string*, para transformar “de ladin” em “de ladinho”.

2.7.3 Similaridade Normalizada de Damerau-Levenshtein

Além de calcular a distância de edição, o algoritmo de Damerau-Levenshtein também pode determinar a similaridade normalizada entre duas strings. Essa métrica fornece uma medida padronizada da semelhança entre as strings, levando em conta não apenas a distância de edição, mas também o comprimento máximo entre elas.

A similaridade normalizada é calculada pela fórmula:

$$\text{Similaridade} = 1 - \frac{\text{Distância de Damerau-Levenshtein}}{\text{Comprimento máximo entre as strings}} \tag{2.24}$$

Nesta fórmula:

- A distância de Damerau-Levenshtein é o número mínimo de operações necessárias para transformar uma *string* na outra.
- O comprimento máximo entre as strings é o comprimento da *string* mais longa entre as duas.

A similaridade normalizada é então calculada subtraindo a proporção da distância de Damerau-Levenshtein pelo comprimento máximo das strings de 1. Isso resulta em um valor que varia de 0 a 1, onde 1 indica uma correspondência perfeita entre as strings e 0 indica nenhuma semelhança.

Por exemplo, ao calcular a similaridade normalizada entre "de ladin" e "de ladinho", onde a distância de Damerau-Levenshtein é 2 e o comprimento máximo entre as strings é 9, a similaridade normalizada é de aproximadamente 0.78. Como essa métrica fornece uma medida mais precisa da semelhança entre as strings, levando em conta o comprimento máximo das strings comparadas à distância de Damerau-Levenshtein, a utilizamos ao invés da distância[1].

[1] Cálculo realizado através do método `normalized_similarity` da biblioteca `textdistance` disponível em: https://pypi.org/project/textdistance/.

3 FERRAMENTAS DE PROCESSAMENTO DE LINGUAGEM NATURAL

Neste capítulo, exploraremos algumas ferramentas de Processamento de Linguagem Natural utilizadas para a análise de sentimentos em textos, as quais foram aplicadas neste projeto. Cada uma dessas ferramentas possui características e metodologias distintas, adaptando-se a diferentes necessidades e contextos de análise.

Na Seção 3.1, apresentamos o AFINN, uma ferramenta conhecida por seu dicionário de palavras e suas respectivas pontuações sentimentais. Em seguida, na Seção 3.2, o VADER (*Valence Aware Dictionary and sEntiment Reasoner*) será também analisado. Por fim, na Seção 3.3 discutiremos o RID(*Regressive Imagery Dictionary*), que categoriza termos para identificar tipos de pensamentos e emoções.

Além das ferramentas mencionadas para análise de sentimento, também abordaremos na Seção 3.4 a marcação de partes do discurso, utilizada para identificar e classificar palavras de acordo com suas categorias gramaticais, utilizando o MAC-MORPHO e o modelo BILL tagger.

Para complementar nossa análise na Seção 3.5, exploraremos técnicas de modelagem de tópicos e recuperação de informação. Especificamente, discutiremos a técnica TF-IDF, utilizada para a transformação e análise dos dados textuais, e a técnica LDA, que contribui significativamente para a identificação de tópicos e padrões nos textos.

3.1 DICIONÁRIO POLARIZADO AFINN

O AFINN (NIELSEN, 2011) é um léxico que apresenta um conjunto de palavras previamente classificadas com valores numéricos que indicam sua polaridade emocional. Desenvolvido originalmente em inglês, o AFINN atribui valores de sentimento às palavras com base em seu uso comum e contexto na língua. A versão mais recente do léxico contém 3.385 palavras ou expressões, cada uma com um peso que varia de -5 (extremamente negativo) a +5 (extremamente positivo), representando a intensidade do sentimento associado à palavra.

3.2 VALENCE AWARE DICTIONARY AND SENTIMENT REASONER

O *Valence Aware Dictionary and sEntiment Reasoner* (VADER) (HUTTO; GILBERT, 2014) é um modelo projetado para avaliar o tom emocional de textos que utiliza não apenas um léxico de valências de palavras, mas também três outros léxicos: o de intensificadores, o de *emoticons* e o de negadores. Esses léxicos adicionais permitem ao modelo capturar melhor as nuances do sentimento, considerando a intensidade e modu-

lação das emoções expressas em um texto. Em português, o léxico foi adaptado pela biblioteca LeIA[1] (ALMEIDA, 2018).

O léxico de sentimento do VADER classifica termos com base em polaridade e intensidade, em uma escala de "-4: Extremamente Negativo" a "+4: Extremamente Positivo". Essas avaliações são médias de dez avaliações independentes, abrangendo palavras individuais e, expressões como "*lavagem cerebral*". Ao contrário de abordagens mais simplistas que atribuem valores inteiros às palavras, o VADER utiliza pontuações de sentimento em precisão de ponto flutuante. Por exemplo, "okay" recebe uma avaliação positiva de 0.9, "bom" 1.9, e "ótimo" alcança 3.1. Em contrapartida, "horrível" recebe -2.5. Palavras não listadas recebem uma pontuação neutra de "0".

O léxico de intensificadores inclui palavras que aumentam ou diminuem a intensidade do sentimento dos termos que acompanham como "muito" ou "pouco". O léxico de *emoticons* ajuda a interpretar símbolos visuais que expressam emoções, atribuindo-lhes valências emocionais apropriadas. Por exemplo, o *emoticon* ":)" é interpretado como um indicador de felicidade. Já o léxico de negadores inclui palavras como "não", "nunca" e "nenhum", que podem inverter ou modificar a polaridade emocional de palavras subsequentes.

O VADER utiliza todos esses léxicos para aplicar diversas regras heurísticas na análise de sentimentos em textos. O modelo avalia se uma palavra funciona como intensificadora, ajustando a emoção associada às palavras adjacentes, ou se é uma palavra de negação, como "não", que pode alterar a polaridade do texto. Esse ajuste é realizado mesmo quando a palavra de negação não está imediatamente antes de um termo sentimental, influenciando negativamente a pontuação.

Além disso, o VADER adota regras adicionais para aprimorar a análise. Uma delas examina a presença de conjunções adversativas, como "mas", "entretanto", "todavia" e "porém". Ao detectar uma dessas conjunções, o modelo divide a frase em duas partes, reduzindo pela metade a valência das palavras antes da conjunção e aumentando em 50% a valência das palavras após ela, refletindo a mudança no significado semântico do texto.

O modelo também considera a ênfase fornecida por palavras em maiúsculas, interpretando como um aumento na intensidade emocional. Por exemplo, em "A música aqui é BOA.", o uso de maiúsculas serve para destacar a intensidade da expressão, elevando a valência emocional associada ao termo.

Por fim, sinais de pontuação, como pontos de exclamação e de interrogação, são analisados para ajustar a intensidade emocional do texto. Pontos de exclamação, por exemplo, são interpretados como indicadores de emoções mais intensas, como evidenciado em "A música aqui é boa!", onde a exclamação adiciona entusiasmo e ênfase ao sentimento expresso.

[1] Biblioteca LeIA disponível em: https://github.com/rafjaa/LeIA.

O cálculo das pontuações de sentimento positivo, negativo e neutro em um texto é realizado através de um processo que envolve a soma das valências de todas as palavras presentes. Inicialmente, a cada palavra no texto é atribuída uma valência ajustada de acordo com o conjunto de regras citadas anteriormente. Essas valências classificam as palavras como positivas, negativas ou neutras.

Para calcular o sentimento positivo, as valências positivas das palavras são somadas, com ajustes leves para compensar as palavras neutras que não têm impacto significativo no cálculo do sentimento positivo. Da mesma forma, para o sentimento negativo, as valências negativas das palavras são somadas, também com ajustes para neutralizar as palavras neutras. As palavras neutras são simplesmente contadas, sem contribuir diretamente para o cálculo das pontuações de sentimento positivo ou negativo, mas ajudam a contextualizar o texto e a ajustar os pesos das valências positivas e negativas.

Depois disso, o modelo considera a ênfase dada pela pontuação no texto, ajustando a soma total das valências de acordo com a intensidade indicada pelos sinais de pontuação. Com essas somas ajustadas, as pontuações de sentimentos positivos, negativos e neutros são normalizadas. Isso é feito dividindo cada soma ajustada pelo total combinado das somas positivas, negativas e neutras, resultando em valores proporcionais que representam a distribuição dos sentimentos no texto.

O modelo também calcula uma "pontuação composta", que é uma normalização da soma total das valências, resultando em um valor entre -1 e 1. Essa pontuação fornece uma visão geral do sentimento do texto, indicando se o sentimento geral é mais positivo, negativo ou neutro, além de prevenir distorções em textos extensos, sendo calculada pela fórmula:

$$\textit{pontuação composta} = \frac{\sum \text{valências}}{\sqrt{\left(\sum \text{valências}^2\right) + \alpha}} \tag{3.1}$$

Onde α é ajustado para 15 no VADER, utilizado para normalizar a *pontuação composta*. Essa escolha foi determinada empiricamente durante o desenvolvimento do VADER para ajudar a evitar distorções significativas na pontuação, especialmente em textos extensos ou com muitas palavras neutras, otimizando assim a precisão e a eficácia do modelo, garantindo que o resultado seja sensível às variações de sentimento ao longo do texto.

A interpretação do *pontuação composta* varia de acordo com seu valor:

- **Sentimento Positivo**: *pontuação composta* entre 0.5 e 1.
- **Sentimento Neutro**: *pontuação composta* entre -0.5 e 0.5.
- **Sentimento Negativo**: *pontuação composta* entre -1 e -0.5.

3.3 REGRESSIVE IMAGERY DICTIONARY

O *Regressive Imagery Dictionary* (RID), desenvolvido por Martindale (1975), Martindale (1990), é um sistema de codificação de análise de conteúdo projetado para medir o pensamento primordial (intuitivo, emocional) versus o pensamento conceitual (abstrato, racional) em textos, baseado na teoria de que a linguagem pode refletir processos mentais inconscientes e primordiais.

No contexto do RID, o pensamento primordial refere-se a ideias, imagens ou conceitos intuitivos e emocionais, frequentemente ligados a experiências sensoriais diretas e ao inconsciente. Esses pensamentos são menos elaborados cognitivamente e conectados às emoções básicas. Em contraste, o pensamento conceitual envolve ideias que são mentalmente elaboradas e menos associadas a experiências sensoriais imediatas, sendo influenciadas por processos conscientes de racionalização.

O RID funciona categorizando termos em diferentes tipos de pensamentos e emoções, de acordo com um dicionário predefinido composto por aproximadamente 3200 palavras e radicais, distribuídos em 3 categorias e 43 subcategorias predefinidas. Além de palavras, o léxico do RID também apresenta radicais seguidos por "*" para capturar qualquer palavra que se encaixe no padrão especificado. Por exemplo, o radical "abus" captura palavras como "abuso", "abusada" e "abusador", abrangendo diversas variações.

A classificação dos termos é realizada com base na frequência e na intensidade dos padrões de linguagem associados aos processos mentais inconscientes e primordiais. Exemplos de termos das subcategorias podem ser observados na Tabela 1. As categorias e subcategorias são as seguintes:

- **Cognição de Processo Primário** - Reflete processos mentais mais arcaicos e simbólicos, com 29 subcategorias agrupadas em 5 grupos:
  - **Impulso** - Oral; Anal; Sexo;
  - **Sensação** - Sensação Geral; Toque; Paladar; Olfato; Audição; Visão; Frio; Dureza; Maciez;
  - **Simbolização Defensiva** - Passividade; Viagem; Movimento Aleatório; Difusão; Caos;
  - **Cognição Regressiva** - Desconhecimento; Eternidade; Alteração de Consciência; Passagem Limítrofe; Narcisismo; Concretude;
  - **Imagens Icarianas** - Ascender; Altura; Descer; Profundidade; Fogo; Água;
- **Cognição de Processo Secundário** - Caracteriza a racionalidade e a lógica no texto, com 7 subcategorias: Abstração; Comportamento Social; Comportamento Instrumental; Restrição; Ordem; Referências Temporais; Imperativo Moral;

- **Emoções** - Identifica a presença de expressões emocionais, com 7 subcategorias: Afeto Positivo; Ansiedade; Tristeza; Afeto; Agressão; Comportamento Expressivo; Glória;

Tabela 1 – Categorias do Método RID e termos de exemplo.

Categoria	**Exemplo de Termos Classificados**
PROCESSO PRIMÁRIO	
Impulso	
Oral	arroto, mamar, apetite
Anal	diarreia, estomacal, imundo
Sexo	amante, beijo, prazer
Sensação	
Sensação Geral	encantado, charme, luxo
Toque	espesso, grosso, acariciar
Paladar	doce, sabor, amargo
Olfato	narina, perfume, aroma
Audição	ouvir, voz, música
Visão	ver, luz, pintar
Frio	frio, inverno, neve
Dureza	rocha, pedra, dureza
Maciez	moleza, sedosa, mole
Simbolização Defensiva	
Passividade	morta, fica, repouso
Viagem	caravana, percurso, embarcar
Movimento Aleatório	onda, rolar, espalhar
Difusão	sombra, neblina, escuro
Caos	selvagem, multidão, ruína
Cognição Regressiva	
Desconhecimento	segredo, estranho, desconhecido
Eternidade	eterno, incessante, imortal
Alteração de Consciência	sonhar, psicose, despertar
Passagem Limítrofe	estrada, parede, borda
Narcisismo	espelho, olho, corpo
Concretude	aqui, fora, perto
Imagens Icarianas	
Subida	salto, levantar, lançar
Altura	ave, céu, alto
Descida	cair, diminuir, afundar
Profundidade	abismo, fundo, cavidade
Fogo	sol, fogo, calor
Água	mar, água, molha
PROCESSO SECUNDÁRIO	
Abstração	algo, pensa, signo
Comportamento Social	dizer, consolar, propor
Comportamento Instrumental	fazer, encontrar, trabalhar
Restrição	autoridade, detida, tabu
Ordem	simples, medir, arranjo
Referências Temporais	quando, agora, era
Imperativo Moral	dever, moral, virtude
EMOÇÕES	
Imperativo Moral	dever, moral, virtude
Afeto Positivo	alegre, prazer, divertido
Ansiedade	medo, crise, trauma
Tristeza	depressão, choro, infelicidade
Afeto	amiga, casamento, querido
Agressão	mau, matar, furiosa
Comportamento Expressivo	arte, poeta, cantar
Glória	admirável, herói, trono

Fonte: Elaborado pela autora com base no corpus RID.

O dicionário fornece uma medida de conteúdo primordial, calculada somando a porcentagem de palavras em um texto que pertencem a qualquer uma das categorias de conteúdo primordial. Esta medida é comparada à porcentagem de palavras em um texto que pertencem às categorias de conteúdo conceitual, permitindo uma análise relativa do pensamento expresso em um texto específico (MARTINDALE, 1990).

As análises fatoriais das categorias do RID têm consistentemente confirmado a validade do dicionário, mostrando que as categorias primordiais carregam alto fator negativo na direção das categorias conceituais, conforme teorizado por Martindale (1990) sobre a evolução literária.

3.4 ETIQUETAS DE PARTE DO DISCURSO

O Mac-Morpho[2] é um corpus de textos em português do Brasil com anotação morfossintática dos *tokens* que conta com 53.374 sentenças, 1.221.465 *tokens* e 22 classes gramaticais diferentes. As etiquetas de parte do discurso (POS tags) incluem substantivos, verbos, adjetivos, advérbios, entre outras, cada uma com exemplos específicos de *tokens* anotados. Na Tabela 2 temos a descrição das etiquetas presentes no corpus e alguns exemplos de *tokens* presentes.

Tabela 2 – Descrição do corpus Mac-Morpho versão 10

Etiqueta	**Significado**	**Exemplos**
|EST	Estrangeirismo	show, mix, mouse, free
ADJ	Adjetivo	especial, nacional, segunda, social, anterior, maiores
ADV	Advérbio	arbitrariamente, consideravelmente, predominantemente
ADV-KS	Advérbio Conectivo Subordinativo	como, onde, quando, por, que
ADV-KS-REL	Advérbio Relativo Subordinativo	onde, quando, que, como, aonde
ART	Artigo (definido ou indefinido)	o, a, os, um, uma, as, uns, umas
CUR	Símbolo de moeda	us, r, cr, rs, u$, R$
IN	Interjeição	ah, ora, não, né, ô
KC	Conjunção Coordenativa	e, mas, ou, pois, porém, nem
KS	Conjunções Subordinativas	que, se, quando, como, porque
N	Substantivo	governo, milhões, presidente, país, mercado
NPROP	Substantivo Próprio	silva, santos, fhc, tavares
NUM	Numeral	cinco, 20, 30, uma, 15
PCP	Particípio	sido, feito, feita, passada, feitas
PDEN	Palavra Denotativa	também, só, apenas, mesmo, até
PREP	Preposição	de, em, para, com, a
PROADJ	Pronome Adjetivo	sua, seu, seus, mais, suas
PRO-KS	Pronome Conectivo Subordinativo	que, o, quem, qual, quais
PRO-KS-REL	Pronome Relativo Conectivo Subordinativo	que, qual, quais, quem, cujo
PROPESS	Pronome Pessoal	me, ele, eu, ela, eles
PROSUB	Pronome Substantivo	isso, um, o, uma, tudo
V	Verbo	estão, ter, afirmou, faz, será
VAUX	Verbo Auxiliar	foi, ser, pode, vai, é

Fonte: Elaborado pela autora com base no corpus Mac-Morpho versão 10.

O modelo *Brill Tagger* é um modelo de etiquetagem de partes do discurso (POS *Tagging*) que se distingue dos modelos estatísticos tradicionais ao empregar uma abordagem

[2] Corpus Mac-Morpho disponível em: http://nilc.icmc.usp.br/macmorpho/.

baseada em regras (BRILL, 1992). Este modelo combina um etiquetador inicial simples com um processo iterativo de refinamento, onde regras de transformação são aprendidas e aplicadas para corrigir erros de etiquetagem (BRILL, 1995).

Inicialmente, um etiquetador simples é utilizado para atribuir etiquetas de partes do discurso às palavras em um corpus de texto. Esse etiquetador pode ser tão básico quanto atribuir a etiqueta mais frequente a cada palavra, uma abordagem conhecida como *baseline tagger*. Por exemplo, se a palavra "casa" é mais frequentemente etiquetada como substantivo ($NOUN$) no corpus de treinamento, o etiquetador inicial atribuirá "$NOUN$" a todas as ocorrências de "casa" (BRILL, 1992).

O passo seguinte envolve o aprendizado e a aplicação de regras de transformação. As regras de transformação no *Brill Tagger* são definidas na forma "se condição, então substitua a etiqueta X pela etiqueta Y". Por exemplo, uma regra pode ser "Se a palavra anterior é um artigo, então substitua a etiqueta "NOUN" por "ADJ". Essas regras são geradas a partir do corpus de desenvolvimento, observando padrões de erros de etiquetagem. Se, por exemplo, "casa" for frequentemente etiquetada como "NOUN" quando deveria ser "ADJ" após um artigo, essa condição será considerada para a criação de uma regra (BRILL, 1995).

Cada regra candidata é então avaliada quanto à sua capacidade de corrigir erros no corpus de desenvolvimento. A regra que corrige o maior número de erros líquidos (correções menos introduções de novos erros) é selecionada. A eficiência de uma regra pode ser expressa matematicamente pela fórmula:

$$\text{Eficiência}(R) = \frac{\text{Erros Corrigidos} - \text{Erros Introduzidos}}{\text{Total de Ocorrências da Regra}} \tag{3.2}$$

As regras selecionadas são aplicadas iterativamente ao corpus etiquetado, refinando progressivamente a etiquetagem. Formalmente, considere um corpus C e uma sequência de palavras $W = \{w_1, w_2, \ldots, w_n\}$ com etiquetas iniciais $T_0 = \{t_{0,1}, t_{0,2}, \ldots, t_{0,n}\}$. As regras de transformação $R = \{r_1, r_2, \ldots, r_k\}$ são aplicadas para gerar etiquetas refinadas T_i (BRILL, 1995).

Cada regra r_j é uma função que mapeia etiquetas T_{i-1} para T_i, onde uma regra específica pode alterar a etiqueta de uma palavra w_k baseada na etiqueta da palavra anterior w_{k-1}:

$$r_j(t_{i-1,k}) = \begin{cases} t'_{i,k} & \text{se } t_{i-1,k} = t_{i,k} \text{ e } t_{i-1,k-1} = t_{prev} \\ t_{i-1,k} & \text{caso contrário} \end{cases} \tag{3.3}$$

onde $t'_{i,k}$ é a nova etiqueta e t_{prev} é a condição de etiqueta da palavra anterior (BRILL, 1992).

3.5 FERRAMENTAS DE MODELAGEM DE TÓPICOS E RECUPERAÇÃO DE INFORMAÇÃO

A modelagem de tópicos e a recuperação de informação são técnicas essenciais para a análise e interpretação de grandes volumes de dados textuais. Essas metodologias permitem a extração de informações significativas e a identificação de padrões nos textos. A seguir, exploramos duas abordagens amplamente utilizadas nessa área: TF-IDF e LDA.

3.5.1 TF-IDF

O TF-IDF é uma técnica utilizada em recuperação de informação e mineração de texto para avaliar a importância de uma palavra em um documento dentro de um corpus, considerando tanto a frequência no documento quanto a raridade no conjunto de documentos (SALTON; BUCKLEY, 1988; RAMOS, 2003). Essa abordagem combina duas medidas principais: a frequência do termo (TF) e a frequência inversa de documentos (IDF).

A **frequência do termo (TF)** calcula quantas vezes uma palavra aparece em um documento específico, refletindo sua relevância dentro desse documento, e sua fórmula é dada por:

$$\text{TF}(t, d) = \frac{\text{Número de vezes que o termo } t \text{ aparece no documento } d}{\text{Número total de termos no documento } d} \tag{3.4}$$

A **frequência inversa de documentos (IDF)**, por outro lado, mede a importância do termo em todo o corpus, penalizando termos que aparecem em muitos documentos. Ela é calculada como:

$$\text{IDF}(t) = \log\left(\frac{\text{Número total de documentos}}{\text{Número de documentos que contêm o termo } t}\right) \tag{3.5}$$

Dessa forma, o IDF ajusta a importância atribuída a termos muito frequentes, como preposições e artigos, que podem ser exageradamente valorizados pelo TF, apesar de não serem significativos. A combinação TF-IDF corrige essa tendência, proporcionando uma medida mais precisa da relevância do termo. O valor TF-IDF de um termo é obtido multiplicando o TF pelo IDF:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t) \tag{3.6}$$

3.5.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) é um modelo de geração probabilística utilizado para a descoberta de tópicos em grandes coleções de documentos de texto. Desenvolvido por Blei, Ng e Jordan (2003), o LDA assume que os documentos são compostos por uma mistura de diferentes tópicos, onde cada tópico é representado por uma distribuição de

palavras. Essa abordagem permite que o modelo identifique padrões de palavras que tendem a aparecer juntas, revelando assim os tópicos latentes nos textos.

O LDA considera que cada documento d é gerado por um processo estatístico envolvendo os seguintes passos:

1. Para cada tópico k em um conjunto de K tópicos, uma distribuição de palavras β_k é extraída de uma distribuição de Dirichlet de parâmetro η, onde $\beta_k \sim Dir(\eta)$.

2. Para cada documento d:

   a) Uma distribuição de tópicos θ_d é extraída de uma distribuição de Dirichlet de parâmetro α, onde $\theta_d \sim Dir(\alpha)$.

   b) Para cada palavra w_{dn} no documento d:

      i. Um tópico z_{dn} é escolhido a partir da distribuição de tópicos θ_d.

      ii. A palavra w_{dn} é gerada a partir da distribuição de palavras $\beta_{z_{dn}}$ associada ao tópico z_{dn}.

O processo acima pode ser descrito matematicamente pela seguinte função de probabilidade conjunta:

$$P(\theta, z, w \mid \alpha, \eta) = \prod_{d=1}^{D} P(\theta_d \mid \alpha) \prod_{n=1}^{N_d} P(z_{dn} \mid \theta_d) P(w_{dn} \mid \beta_{z_{dn}}) \tag{3.7}$$

onde:

- θ_d é a distribuição de tópicos para o documento d,
- z_{dn} é o tópico atribuído à n-ésima palavra do documento d,
- w_{dn} é a palavra observada, e
- α e η são hiperparâmetros que controlam a distribuição de Dirichlet para θ e β, respectivamente.

O objetivo do LDA é inferir as distribuições de tópicos θ_d para cada documento e as distribuições de palavras β_k para cada tópico, dado um conjunto de documentos observados. Isso é geralmente realizado através de métodos de inferência aproximada, como a amostragem de Gibbs (GRIFFITHS; STEYVERS, 2004) ou a variação de Expectativa-Maximização (EM) (BLEI; NG; JORDAN, 2003).

Em termos práticos, o LDA é uma ferramenta poderosa para a análise exploratória de grandes volumes de texto, permitindo a organização e a identificação de temas recorrentes. Contudo, sua aplicação requer a escolha apropriada do número de tópicos K, o que pode influenciar significativamente a qualidade dos tópicos extraídos (STEVENS et al., 2012).

3.6 INTEGRAÇÃO DAS FERRAMENTAS DE PROCESSAMENTO DE LINGUAGEM NATURAL

A escolha das abordagens de Processamento de Linguagem Natural empregadas neste trabalho foi estrategicamente orientada para explorar e extrair diversas dimensões dos textos analisados, visando otimizar a construção dos modelos de aprendizado de máquina. Cada técnica selecionada oferece uma perspectiva: a análise de sentimento captura nuances emocionais e polaridades que podem influenciar a categorização em contextos sensíveis ao sentimento; o Pos-Tagging fornece uma visão gramatical que revela padrões linguísticos e facilita a extração de características mais robustas; enquanto as técnicas de recuperação de documentos, como TF-IDF e LDA, são essenciais para reduzir a dimensionalidade, identificar padrões subjacentes e transformar dados textuais em representações numéricas processáveis. A combinação dessas metodologias não só amplia a cobertura das dimensões linguísticas relevantes, mas também melhora a capacidade dos algoritmos de aprendizado de máquina em aprender e generalizar a partir dos dados textuais.

4 BASE DE DADOS

Considerando nossa intenção de construir uma base de dados própria para armazenar informações estruturadas sobre músicas em português, com foco em dados relacionados a áudio e letras, buscamos desenvolver uma base que ofereça variáveis semelhantes às presentes na base ALF200k[1] (ZANGERLE et al., 2018). Esta base foi projetada para incluir exclusivamente músicas com letras em português, coletadas de diversas fontes disponíveis na web. Além da coleta de informações, nossa base foi concebida para integrar os resultados da correspondência entre os registros obtidos.

Destaca-se que essa estrutura que desenvolvemos serve como uma modelagem intermediária, utilizada para a construção do conjunto de dados final. Ao longo deste processo, organizamos os dados em um formato estruturado que possibilitasse a posterior geração de um conjunto de dados definitivo e pronto para análise, contendo variáveis específicas tanto de áudio quanto de letras. As etapas de concepção e desenvolvimento desta base de dados serão detalhadas nas seções seguintes, conforme descrito a seguir:

1. **Levantamento de Fonte de Dados:** Em um esforço para reunir uma coleção abrangente de músicas, exploramos uma variedade de fontes *online*. Durante esse processo, priorizamos a seleção de músicas com letras em português para garantir a relevância para nossa análise.

2. **Pareamento de Registros:** No pareamento de registros, desenvolvemos e aplicamos métodos para identificar e unir registros de músicas de diferentes fontes, compensando a ausência de um identificador universal. Utilizamos técnicas baseadas em distância e métodos probabilísticos para lidar com variações e inconsistências nos dados, garantindo a precisão na correspondência entre as músicas.

3. **Obtenção dos Dados:** Nesta etapa, realizamos a coleta e filtragem dos dados de músicas utilizando APIs das plataformas Spotify e Last.fm, além de consultar a base de dados de Lima. Processos de busca e verificação foram implementados para garantir a integridade e a relevância dos dados obtidos, considerando tanto informações textuais quanto atributos de áudio.

4. **Seleção:** Com base na análise prévia, selecionamos os atributos mais pertinentes para comparação e armazenamento no banco de dados. Esses atributos incluem informações sobre o áudio, como acústica e dançabilidade, bem como dados textuais, como letras das músicas.

[1] Base de Dados ALF200k disponível em: https://github.com/dbis-uibk/ALF200k

5. **Modelagem Conceitual:** Desenvolvemos um modelo conceitual, identificando as principais entidades envolvidas no contexto de músicas. Estabelecemos também os relacionamentos entre essas entidades.

6. **Modelagem Lógica:** Com base no modelo conceitual, elaboramos um modelo lógico do banco de dados. Nessa etapa, traduzimos as entidades e relacionamentos em estruturas de tabelas, definindo as chaves primárias e estrangeiras necessárias para garantir a integridade dos dados.

7. **Modelagem Física:** Por fim, implementamos o modelo físico do banco de dados, detalhando os aspectos técnicos de sua implementação. Isso inclui a definição de tipos de dados específicos, índices e otimizações de desempenho para garantir uma operação eficiente da base de dados.

4.1 LEVANTAMENTO DE FONTE DE DADOS

A construção de um conjunto de dados adequado é fundamental para o sucesso de qualquer pesquisa ou experimento. Nesta Seção, descreveremos o processo as fontes de dados que exploramos para construir o conjunto de dados que utilizamos para a realização dos experimentos propostos neste trabalho.

4.1.1 Conjunto de Dados Inicial

Para conduzir as análises e investigações propostas por esse trabalho, foi necessário garantir a disponibilidade de uma coleção abrangente de letras de músicas. Inicialmente, buscamos fontes confiáveis e acessíveis para obter um conjunto de dados representativo. Optamos, então, por explorar um conjunto de dados[2] disponibilizado gratuitamente na plataforma Kaggle. Esse conjunto, datado de 2022, foi aprimorado a partir de dados obtidos pela API do Genius[3]. Na versão aprimorada disponibilizada, foram adicionadas informações sobre o idioma das músicas, obtidas por meio de modelos de PLN capazes de identificar o idioma nativo de cada entrada. Os atributos foram utilizados para garantir a inclusão apenas de músicas com letras em português em nossa análise, alinhando-se aos objetivos específicos da pesquisa.

No entanto, a manipulação de conjuntos de dados volumosos pode ser desafiadora. A considerável magnitude do conjunto de dados, com cerca de 8.86 GB de informações e mais de um bilhão de palavras contidas no formato CSV, foi um dos primeiros desafios enfrentados.

Para contornar esse obstáculo, desenvolvemos um código na linguagem Python que realiza a leitura do arquivo CSV apenas uma vez e, em seguida, importa os dados para

[2] Kaggle - Genius Song Lyrics: https://www.kaggle.com/datasets/carlosgdcj/genius-song-lyrics-with-language-information/data

[3] Genius API: https://docs.genius.com/

Tabela 3 – Metadados do conjunto de dados "Genius Song Lyrics"

INFORMAÇÃO	METADADO
Número de músicas	5.063.837
Número de atributos	11
Tamanho	8,857 GB
Tipo do arquivo	.csv
Total de Palavras do atributo "Lyrics" (atributo que apresenta a letra completa da música)	1.347.684.676

Fonte: Autora.

uma tabela em um banco de dados SQL utilizando o SQLite[4], biblioteca escolhida devido à sua leveza, eficiência e capacidade de armazenar todo o banco de dados em um único arquivo no disco, eliminando a necessidade de um servidor dedicado e simplificando o processo de gerenciamento e análise de dados.

4.1.2 Fonte de Dados Complementares

Tendo em vista o objetivo principal do trabalho de analisar exclusivamente músicas com letras em português, realizamos inicialmente uma etapa de filtragem para reter somente as músicas classificadas na linguagem "pt" dentro do conjunto de dados. Isso resultou em um conjunto de 167.947 músicas.

Para enriquecer nossa base de dados com atributos de áudio, como acústica e dançabilidade, utilizamos a API do Spotify[5]. Essas métricas são calculadas automaticamente pelo Spotify por meio de algoritmos proprietários, cujas fórmulas exatas não são divulgadas na documentação oficial, o que limita nossa compreensão detalhada sobre a geração dessas variáveis. Utilizamos a API para acessar essas informações e, em seguida, exploramos as músicas em nosso conjunto de dados com base nos nomes dos artistas e das faixas, integrando os dados adicionais para enriquecer a análise.

Para a obtenção dos rótulos de gênero musical, utilizamos a API do Last.fm[6], uma plataforma de recomendação musical que possibilitou a busca por músicas com base nos nomes dos artistas e das faixas. Recuperamos os rótulos atribuídos pelo Last.fm e os agrupamos em gêneros principais com base em uma correspondência previamente definida[7]. Nesse processo, selecionamos e categorizamos certos rótulos, agrupando subgêneros sob categorias mais amplas. Por exemplo, "funk carioca" e "funk brasil" foram agrupados no

[4] SQLite: https://sqlite.org/

[5] Spotify API: https://developer.spotify.com/documentation/web-api

[6] Last.fm API: https://www.Last.fm/api

[7] O arquivo de depara utilizado para mapear e organizar os rótulos de gênero atribuídos pelo Last.fm pode ser acessado em: hhttps://github.com/leticiatavaresds/BAMPORT/blob/main/00_Data_Base/Data_Input/genres_tags.json

gênero “funk”, criando assim uma base de dados onde as músicas estão classificadas em um ou mais gêneros principais.

Como complemento, e para lidar com a limitação de músicas não encontradas no Last.fm, empregamos também um conjunto de dados fornecido pelo estudo (LIMA et al., 2020), contendo 138.368 letras de músicas brasileiras distribuídas em 14 gêneros.

4.2 PAREAMENTO DE DADOS

Um desafio significativo encontrado foi a ausência de um identificador comum entre as diferentes fontes de dados, o que dificultou a garantia de que os registros se referissem à mesma música. Embora o uso de um identificador universal, como o ISRC (*International Standard Recording Code*), fosse uma solução ideal, nossa base inicial do Genius não possuía esse atributo.

Para contornar essa limitação e assegurar a correspondência entre registros de diferentes fontes, desenvolvemos e implementamos estratégias de pareamento de registros. Essas estratégias foram projetadas para reconhecer não apenas os metadados das músicas, mas também para lidar com possíveis variações e inconsistências nos dados. Integradas à fase de extração dos dados, essas estratégias permitiram realizar novas buscas e ajustes quando não havia correspondência direta entre os registros, aumentando a precisão na identificação de correspondências e melhorando a integridade da base de dados final.

A solução implementada envolveu a aplicação de técnicas de emparelhamento probabilístico de registros, que foram integradas diretamente na fase de extração dos dados. Isso permitiu que, ao não encontrar uma correspondência entre registros ou encontrar uma baixa, novas buscas pudessem ser realizadas para alcançar ou refinar essa correspondência.

4.2.1 Identificação dos atributos chaves

A primeira etapa do processo de emparelhamento de registros foi a identificação dos atributos que poderiam ser utilizados para realizar a correspondência entre as músicas. Após análise dos atributos fornecidos pela base de dados inicial e aqueles aceitos como parâmetro de busca pelas APIs das fontes externas, constatamos que os únicos campos disponíveis para esse fim eram o título da música e o nome do artista.

Além disso, como a base inicial também inclui um atributo que lista os artistas convidados, esse atributo foi considerado como uma segunda opção para busca. Cantores convidados são artistas que contribuem com vocais ou outros elementos musicais em uma faixa, além do artista principal da música. Dessa forma, caso a busca utilizando o título e o nome do artista principal não retornasse um registro correspondente, novas buscas adicionais foram realizadas considerando o título da música e o nome do cantor convidado como parâmetros de busca para cada cantor convidado. Essa abordagem foi adotada

para aumentar as chances de encontrar uma correspondência entre as músicas, levando em conta diferentes possibilidades de formatação dos dados entre as fontes.

Um exemplo que ilustra a relevância dessas buscas adicionais ocorreu com a música "Tudo por Nós", interpretada pelo cantor "Chris MC" em colaboração com o cantor "Knust". Ao tentarmos localizar essa música na base do Spotify utilizando apenas o título e o nome do cantor principal, não foi encontrado nenhum registro correspondente. No entanto, ao realizar a busca utilizando o título da música e o nome do cantor convidado, um registro correspondente foi encontrado com sucesso. Isso demonstra como a consideração dos cantores convidados como uma segunda opção de busca foi importante para aprimorar a obtenção de registros correspondentes entre as fontes de dados.

4.2.2 Pré-Processamento

O objetivo do pareamento de registros proposto é possibilitar a identificação de pares de registros que representem a mesma música, sendo um registro proveniente de uma fonte externa, como o Spotify, e o outro do Genius. No entanto, cada plataforma possui sua própria estrutura de dados e formato de escrita, o que torna desafiador encontrar correspondências entre eles. Esta seção busca ilustrar como foi implementada a etapa de pré-processamento, fundamental para preparar os dados para as etapas seguintes.

Devido à falta de padronização na escrita dos campos entre as diferentes fontes de dados, não era possível realizar uma comparação direta entre as strings. Essa inconsistência decorria de diversos fatores, como a presença de múltiplos espaços em branco consecutivos, a utilização de acentos, caracteres especiais como pontuações, e a variação entre letras maiúsculas e minúsculas. Para tornar essas comparações mais precisas, foi essencial aplicar uma padronização nesse sentido.

Buscando contornar esse obstáculo, desenvolvemos uma função que realiza esse tratamento para qualquer texto que fosse ser comparado, realizando as seguintes ações:

- Remoção de acentuação.
- Substituição da *substring* " & " pela *substring* " e ".
- Remoção de caractere especial.
- Remoção de espaço no início e no fim do texto.
- Remoção de espaços múltiplos.
- Conversão de todos os caracteres para letras minúsculas.

Optamos por manter os números durante o processo de tratamento, pois sua presença pode distinguir músicas com títulos semelhantes. Por exemplo, as músicas "Automaticamente" e "Automaticamente 2", ambas interpretadas pelo artista "Mc Léleco" e presentes na base de dados, diferem apenas pelo número 2 no título, mas são músicas distintas.

4.2.3 Cálculo da Similaridade entre Strings

Para calcular as probabilidades de correspondência entre identificadores musicais, utilizamos dois algoritmos que medem a similaridade entre strings: o coeficiente de Sørensen-Dice e a similaridade normalizada de Damerau-Levenshtein, cada um com suas vantagens específicas na comparação de cadeias de caracteres.

O coeficiente de Sørensen-Dice é eficaz para medir a similaridade entre conjuntos de bigramas formados a partir das strings, enquanto o algoritmo de Damerau-Levenshtein calcula a distância mínima de edição. Cada métrica tem suas forças: o coeficiente de Sørensen-Dice é sensível à presença de caracteres comuns, e o Damerau-Levenshtein lida bem com erros comuns de digitação. Combinando as duas, obtemos uma análise mais equilibrada e precisa. Esta combinação permite obter uma probabilidade mais assertiva de que dois registros representem a mesma música, ao considerar diferentes aspectos da similaridade textual.

Para calcular a probabilidade final de correspondência entre dois registros, definimos pesos para cada métrica. Baseado no estudo (MASTUB; LACERDA, 2021), atribuímos um peso de 80% ao coeficiente de Sørensen-Dice e 20% à saída do algoritmo de Damerau-Levenshtein por ter alcançado o resultado mais satisfatório. Esta ponderação reflete a importância relativa de cada métrica na determinação da similaridade global.

Vamos ilustrar o cálculo completo utilizando novamente as strings “de ladin” e “de ladinho“:

1. **Coeficiente de Sørensen-Dice:** Como visto na Seção 2.7.1, o coeficiente de Sørensen-Dice entre as duas strings é dado por:

$$DSC = \frac{2 \cdot 7}{7 + 9} \approx 0.875 \tag{4.1}$$

2. **Distância de Damerau-Levenshtein:** A distância entre as strings é 2 (calculada em 2.7.2) e a similaridade normalizada é dada por:

$$\text{Similaridade} = 1 - \frac{2}{9} \approx 0.78 \tag{4.2}$$

3. **Combinação das Métricas:** A similaridade final é então calculada ponderando as duas métricas:

$$\text{Similaridade Final} = 0.8 \cdot 0.875 + 0.2 \cdot 0.78 \approx 0.856 \tag{4.3}$$

Portanto, ao combinar o coeficiente de Sørensen-Dice e a similaridade normalizada de Damerau-Levenshtein, obtemos uma probabilidade final de aproximadamente 0.8556 de que os registros “de ladin” e “de ladinho” se refiram à mesma música.

4.2.4 Processo de Classificação

Inicialmente, poderíamos ter optado por obter todos os dados das bases externas antes de calcular a similaridade entre os registros e, posteriormente, eliminar as músicas com baixa similaridade. No entanto, através de testes manuais utilizando as APIs das fontes externas, constatamos que uma música pode não ser encontrada ao buscarmos pelo seu artista principal, mas pode ser localizada ao buscar por um artista convidado. Para lidar com essa situação, desenvolvemos um processo adaptativo e iterativo.

Para cada música na base inicial do Genius[8], seguimos o seguinte procedimento:

1. Realizamos uma busca pela música na base externa utilizando como parâmetros de pesquisa o nome da música e o nome do artista principal, capturando o primeiro resultado retornado.

2. Tratamos os campos "Nome da Música" e "Nome do Artista" retornados nos metadados encontrados.

3. Calculamos a similaridade entre os campos nome da música e o nome do artista nas duas bases utilizando a combinação do coeficiente de Sørensen-Dice e a distância de Damerau-Levenshtein, conforme descrito anteriormente.

4. Caso a similaridade obtida na busca inicial não seja 1, realizamos uma nova busca para cada artista convidado listado. Esta abordagem aumenta a probabilidade de encontrar uma correspondência correta, considerando que algumas músicas podem ser indexadas de diferentes maneiras nas bases externas.

5. Por fim, se múltiplas amostras forem retornadas para a mesma música, selecionamos a que possui o maior score de similaridade. Esta etapa assegura que a correspondência mais precisa é mantida, melhorando a qualidade dos dados interligados.

Apenas realizar esse procedimento não se mostrou suficiente, então uma abordagem adicional foi o cálculo da similaridade para algumas variações do nome do artista em cada busca, visando obter uma correspondência maior, se preciso. O processo funciona da seguinte maneira: primeiro, calculamos a similaridade entre o nome do artista principal retornado na busca e o presente na base. Se a similaridade for menor que 1 e o nome do artista da base inicial contiver múltiplos artistas separados por " & ", separamos esses nomes, ordenamos alfabeticamente e unimos novamente para realizar um novo cálculo de similaridade, agora comparando com a junção de todos os cantores retornados pela fonte externa como participantes da música.

Por exemplo, consideremos a música "Lei do Retorno Ao Vivo" no conjunto de dados inicial, onde o campo artista principal é dado por "MC Hariel & MC Don Juan". Ao

[8] Kaggle - Genius Song Lyrics: https://www.kaggle.com/datasets/carlosgdcj/genius-song-lyrics-with-language-information/data

procurarmos por esse registro no Spotify, devido à diferença na estrutura de dados, obtemos apenas "MC Hariel" como o artista principal. Isso resulta em uma correspondência baixa ao compararmos as strings. No entanto, podemos melhorar essa correspondência ao ajustar os nomes dos artistas.

Primeiro, além do artista principal, pegamos os nomes de todos os artistas participantes retornados pelo Spotify (que também apresenta os artistas convidados), ordenamos alfabeticamente e unimos usando vírgulas como conectores, obtendo "MC Hariel, MC Don Juan". Em seguida, repetimos o processo para a *string* da base inicial: separamos os nomes dos cantores utilizando como separadores "," e "&", ordenamos alfabeticamente e unimos novamente com vírgulas, resultando também em "MC Hariel, MC Don Juan". Ao calcular novamente a similaridade entre essas strings ajustadas, o score será igual a 1, indicando uma correspondência perfeita.

Contudo, ainda observamos casos em que ambas as similaridades resultaram em valores baixos e um novo cálculo poderia ser feito. Assim, caso nenhuma das similaridades seja igual a 1, realizamos uma última comparação usando apenas o primeiro artista da *string* de artista principal da base inicial e o principal artista retornado pela fonte externa.

Um exemplo da necessidade dessa aplicação ocorreu com a música "Malokera", onde o campo artista é dado por "MC Lan, Skrillex & Troyboi" no conjunto de dados inicial. Ao procurarmos por esse registro no Spotify, obtemos apenas "MC Lan" como o artista principal, resultando em um score de similaridade de apenas 0.288. Ao realizar a transformação para o cálculo da segunda opção de similaridade, obtemos "MC Lan, Skrillex, Troyboi" da base inicial. Contudo, ao considerar todos os artistas retornados pelo Spotify, temos "Ludmilla, MC Lan, Skrillex, Troyboi, Ty Dolla $ign", resultando num score de 0.643.

Por fim, ao fazer essa última tentativa de similaridade, ficamos com "MC Lan" da base inicial e "MC Lan" da fonte externa, chegando finalmente a um score igual à 1.

4.2.5 Dificuldades

Durante o processo de pareamento de dados, uma das maiores dificuldades enfrentadas foi identificar o momento ideal para calcular a similaridade entre os registros. Inicialmente, tentamos realizar a extração dos dados em várias etapas para, somente após essa fase, calcular a similaridade. No entanto, após algumas tentativas frustradas, verificamos que seria mais eficiente calcular a similaridade ainda no processo de extração. Dessa forma, conseguimos realizar novas tentativas de extração imediatamente quando a similaridade observada era baixa, otimizando o processo.

Além disso, outra questão crítica foi o estabelecimento do método de cálculo da similaridade. Realizamos alguns experimentos para testar e avaliar qual combinação entre o coeficiente de Sørensen-Dice e a distância de Damerau-Levenshtein apresentava o melhor desempenho. Também foi necessário analisar pares com similaridades não perfeitas para

identificar os tratamentos necessários nas strings para assegurar que os dados estivessem adequadamente preparados e compatíveis para o cálculo.

4.3 OBTENÇÃO DOS DADOS DAS FONTES EXTERNAS

Nesta seção, descrevemos o processo de obtenção dos dados das fontes externas utilizadas em nossa pesquisa. Inicialmente, realizamos a busca e filtragem das músicas na base de dados do Spotify e do Last.fm, abordando as particularidades e desafios encontrados durante a coleta. Adicionalmente, exploramos a base de dados de Lima para complementar nosso conjunto final com rótulos de gênero adicionais. Cada subseção apresenta os métodos e resultados específicos para cada fonte, destacando a abordagem adotada e os impactos na base de dados final utilizada para a análise.

4.3.1 Análise e Filtragem de Músicas na Base do Spotify

Durante a busca de correspondências na base de dados do Spotify, foram realizadas chamadas para a API para todas as 167.947 músicas em português presentes na base inicial. Destas, 106.895 músicas retornaram algum dado, enquanto 61.039 não foram encontradas na base do Spotify e 13 músicas não retornaram resultados devido a erros na chamada da API.

Tabela 4 – Resultados das Buscas Iniciais na Base do Spotify

Total de Músicas Buscadas	167.947
Músicas com Dados Retornados	106.895
Músicas Não Encontradas	61.039
Músicas com Erros na API	13
Tempo Total de Execução	2 dias, 5:43:04

Das 106.895 músicas que retornaram dados, 88.307 músicas apresentaram scores de similaridade perfeitos (1, 1) tanto para o título quanto para o nome do artista, esses pares puderam ser considerados correspondências exatas, indicando que os dados se tratavam da mesma música em ambas as bases de dados. As 18.588 músicas restantes não tiveram scores perfeitos, necessitando de uma análise adicional para determinar uma linha de corte que permitisse a identificação das correspondências mais prováveis.

Tabela 5 – Distribuição de Scores de Similaridade

Total de Músicas com Scores Perfeitos (1, 1)	88.307
Total de Músicas sem Scores Perfeitos	18.588

Ao analisar as músicas sem scores perfeitos, notamos que muitas delas tinham um score de similaridade do nome do artista igual a 1, mas o score do título era ligeiramente menor devido à presença de termos adicionais como “acústico” ou “remasterizado”. Para resolver

esse problema, realizamos um novo tratamento nos títulos das músicas (tanto da base do Genius quanto do Spotify), removendo *substrings* específicas que poderiam causar essas discrepâncias. Os termos removidos dos títulos foram: “ao vivo”, “live”, “feat”, “acústico”, “playback”, “bônus”, “bônus track”, “remasterizado”, “versão brasileira”, “original album”, “album”, “remastered”, “radio edit” e “remaster”.

Após esse refinamento, recalculamos os scores de similaridade. Como resultado, conseguimos alcançar scores de (1, 1) para mais 7.348 músicas.

Tabela 6 – Resultados Após Refinamento dos Títulos

Músicas com Dados Retornados	106.895
Total de Músicas com Scores Perfeitos (1, 1)	88.307
Músicas com Scores Aperfeiçoados (1, 1)	7.378
Músicas Restantes com Scores Diferentes de (1, 1)	11.210

4.3.2 Análise e Filtragem de Músicas na Base do Last.fm

Primeiro, realizamos um filtro inicial para tentar capturar os dados apenas das músicas encontradas na base do Spotify, pois precisávamos ter todos os dados de uma música antes de considerar buscar informações na base do Last.fm. Dessa forma, ignoramos as 61.052 músicas não encontradas ou que apresentaram erro. Das 106.895 músicas procuradas, 75.452 retornaram dados e 117 músicas não retornaram resultados devido a erros na chamada da API. O processo completo de busca e armazenamento levou 23 horas e 13 minutos para ser executado.

Tabela 7 – Resultados das Buscas Iniciais na Base do Last.fm

Total de Músicas Buscadas	106.895
Músicas com Dados Retornados	75.452
Músicas Não Encontradas	31.426
Músicas com Erros na API	17
Tempo Total de Execução	23:13:06

De um total de 75.452 músicas encontradas na base de dados, 17.419 continham um ou mais rótulos. As demais 58.033 músicas foram descartadas, pois o foco do nosso estudo são as músicas com *tags* para a criação de modelos supervisionados. Dentre as 17.419 músicas restantes, 17.326 apresentaram scores de similaridade perfeitos (1, 1) tanto para o título quanto para o nome do artista. Esses pares foram considerados correspondências exatas, indicando que os dados se referem à mesma música em ambas as bases de dados. Isso demonstra que a grande maioria das músicas com *tags* foram identificadas como correspondências exatas.

Tabela 8 – Distribuição de Scores de Similaridade Last.fm

Músicas com uma ou mais Tags	17.419
Músicas com Scores Perfeitos (1, 1)	17.326

4.3.3 Determinação do Limiar de Similaridade

Restaram apenas 93 músicas com todos os dados desejados, incluindo os atributos de áudio obtidas do Spotify e as *tags* obtidas do Last.fm, que não foram consideradas correspondências exatas. Decidimos não estabelecer um limiar de similaridade para a inclusão de novas músicas, pois isso exigiria um estudo adicional, que poderia ser um esforço desnecessário. Em vez disso, acrescentamos uma fase em que um arquivo é gerado com essas músicas para análise manual.

Adicionamos a variável *manual_match* tanto para a relação Genius-Spotify quanto para a relação Genius-Last.fm, indicando que, apesar dos scores de similaridade não serem 1, a música foi considerada manualmente como uma correspondência exata. Das 93 músicas, 58 foram confirmadas manualmente como correspondências exatas com base na verificação dos nomes das músicas e dos artistas em cada base. Dessa forma, o número total de músicas consideradas correspondências exatas (1, 1) entre Genius e Spotify e (1, 1) entre Genius e Last.fm, seja pelo cálculo de similaridade ou manualmente, foi de 17.384.

Em nossos códigos, essa fase manual é opcional; caso reproduzam o código sem realizar essa etapa, não haverá erro, ocorrerá apenas o descarte das músicas que não foram automaticamente classificadas como correspondências exatas. Isso garante flexibilidade na aplicação do código, permitindo que os usuários escolham entre uma análise totalmente automatizada ou uma abordagem que inclui validação manual para maior precisão.

Tabela 9 – Tamanho da Base de Dados

Tamanho da Base com correspondências automáticas)	17.328
Correspondências Manuais	58
Estatísticas da Base após cruzamento com a base do Last.fm	17.384

4.3.4 Filtragem de Músicas na Base Lima

Como nossa base final ficou relativamente pequena e desbalanceada, decidimos buscar uma nova fonte de dados que pudesse fornecer rótulos de gêneros para as músicas que não foram encontradas na base do Last.fm. Encontramos uma base de dados construída no estudo de Lima et al. (2020), que contém músicas obtidas da plataforma Vagalume[9], rotuladas com gêneros musicais.

[9] Plataforma Vagalume disponível em: https://www.vagalume.com.br/

Procuramos, então, na conjunto de dados Lima por músicas que também estavam presentes na base do Spotify, com o objetivo de obter mais rótulos de gênero. Sem estabelecer um limiar de similaridade, selecionamos as músicas que apresentaram scores de similaridade perfeitos (1,1) tanto para o título quanto para o nome do artista, totalizando 25.274 músicas.

Após essa etapa, removemos as músicas que já possuíam rótulos de gênero obtidos através da base do Last.fm, resultando em uma base final de 27.777 músicas rotuladas. Dessas, 17.384 músicas foram rotuladas com as *tags* do Last.fm, enquanto as 10.393 restantes receberam rótulos de gênero oriundos da base de dados do estudo de Lima et al. (2020).

4.4 SELEÇÃO DE DADOS

Após uma análise das informações disponíveis, precisamos identificar quais dados são imprescindíveis para integrar ao nosso banco de dados. Como lidamos com uma variedade de fontes, essa etapa se torna ainda mais relevante. Após avaliar os resultados de cada fonte e nosso processo de obtenção dos dados, optamos por selecionar as informações mais relevantes para essa integração: o título da música e o artista.

Esses dados são essenciais para compor cada entrada em nosso banco de dados, proporcionando uma base sólida para nossas análises. Além disso, é importante incluir detalhes sobre a origem dos registros e informações necessárias para conectar os dados.

Paralelamente, realizamos uma análise das variáveis disponíveis, com o objetivo de identificar aquelas que são descartáveis para o escopo do projeto. Durante esse processo, observamos que a variável de linguagem não contribui de forma significativa, visto que, após a filtragem inicial, todas as músicas presentes na base de dados foram classificadas com o idioma português. Dessa forma, decidimos remover essa variável da análise, mantendo apenas as informações consideradas relevantes para nossos objetivos, como o nome da música, artistas, ano de lançamento, código ISRC, propriedades de áudio, rótulos e letras. Essa abordagem visa garantir que nossa base de dados seja composta apenas por informações pertinentes e úteis para as análises e investigações propostas neste trabalho.

4.5 CONSTRUÇÃO DA BASE DE DADOS

A construção de uma base de dados envolve a criação de uma estrutura de banco de dados que não só armazena dados de maneira organizada, mas também facilita o pareamento e a análise dos registros. Para alcançar esses objetivos, foi preciso desenvolver um modelo de dados que fosse capaz de integrar informações de diferentes plataformas de maneira coerente e sem redundâncias.

O processo de população da base de dados, ilustrado na Figura 2, envolveu a integração de informações provenientes de diversas fontes, como Genius, Spotify, Last.fm e o conjunto

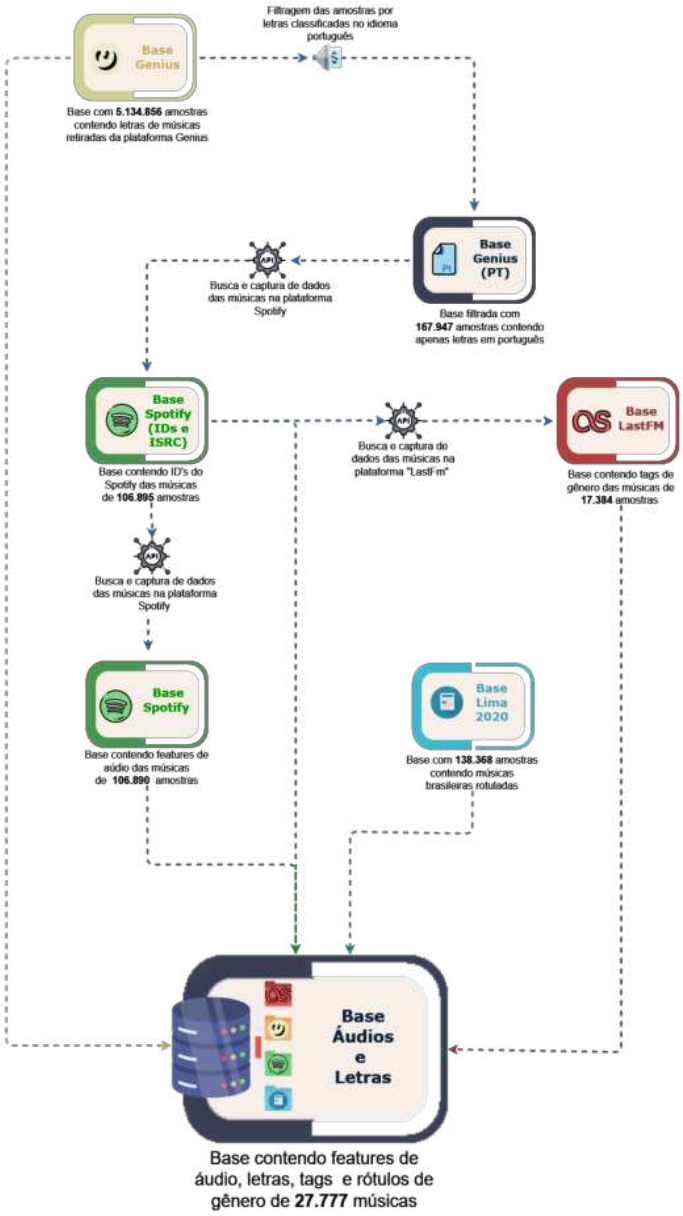

Figura 2 – Ilustração do Processo de População da Base de Dados. Disponível em: https://github.com/leticiatavaresds/BAMPORT/blob/main/Data%20Base%20Info/db_population.png.

de dados Lima. Esse processo incluiu múltiplas etapas de filtragem, visando obter uma base de dados completa e abrangente para nossas análises.

4.5.1 Modelo Conceitual

O modelo conceitual de banco de dados é uma representação abstrata que descreve a estrutura lógica dos dados sem considerar como esses dados serão fisicamente armazenados. Ele serve como uma ponte entre os requisitos do negócio e o modelo físico de dados, garantindo que todas as necessidades de informação da organização sejam atendidas (ELMASRI; NAVATHE, 2015).

Para garantir uma estrutura de banco de dados coesa e evitar redundâncias de informações, criamos entidades específicas para cada uma das fontes de dados. Além disso, definimos entidades para os gêneros musicais associados a cada rótulo e para as características do áudio. O diagrama ER (Entidade-Relacionamento) apresentado na Figura 3 ilustra o modelo conceitual proposto.

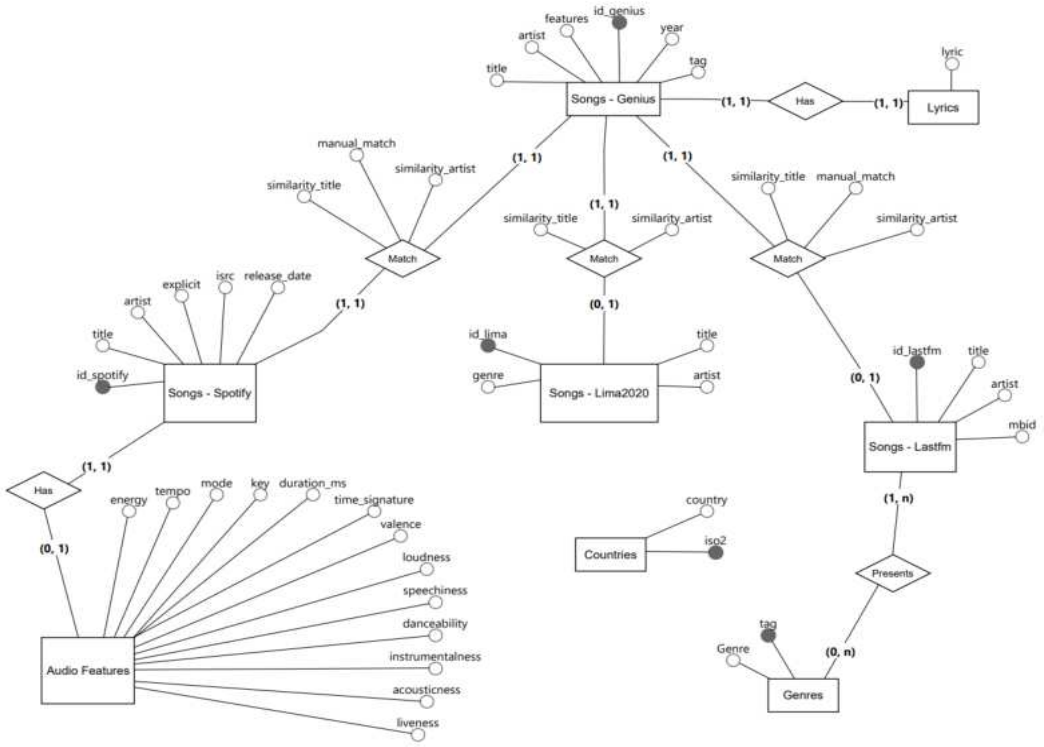

Figura 3 – Modelo Conceitual do Banco de Dados. Disponível em: https://github.com/leticiatavaresds/BAMPORT/blob/main/Data%20Base%20Info/db_conceptual_model.png.

A principal focalização está na apresentação dos metadados das músicas do conjunto de dados inicial, utilizando o ID do Genius como chave. Embora o atributo "*lyric*" pudesse ser diretamente adicionado aqui, optamos por criar uma entidade separada para manter a organização e clareza do banco de dados, facilitando a manutenção e manipulação dos dados. Essa abordagem permite uma separação clara entre as informações da música e suas letras. Além disso, foram desenvolvidas estruturas específicas para cada fonte externa, que incluem informações adicionais como propriedades de áudio, código ISRC, *tags* e códigos dos países.

Para as músicas disponíveis na plataforma Spotify, utilizamos a "*Songs - Spotify*", que também mantém um relacionamento "*has*" com a "*Áudio Features*", contendo propriedades do áudio. Como nem todas as músicas no Spotify têm esses dados, a cardinalidade da relação é de 1 para 0.

"Last.fm Songs" armazena informações sobre músicas disponíveis na plataforma Last.fm. Cada música pode estar associada a uma ou mais *tags*, então criamos uma entidade sepa-

rada para classificar essas *tags* de acordo com seus gêneros.

Quanto à entidade "Countries", embora não esteja diretamente relacionada a outras partes do modelo fornecido, ela pode ser utilizada para atribuir nacionalidades aos códigos ISRC das músicas. Isso se deve ao fato de que todo código ISRC começa com o código ISO 3166-1 alpha-2 associado ao país de registro da música, sendo este código o atributo chave da estrutura.

4.5.2 Modelo Lógico e Modelo Físico

O modelo lógico de banco de dados é uma representação que descreve a estrutura do banco de dados em termos de tabelas, colunas e relacionamentos, refletindo a organização dos dados de maneira que possa ser implementada fisicamente em um sistema de gerenciamento de banco de dados. Ele traduz o modelo conceitual em um formato que define claramente como os dados serão armazenados e manipulados, assegurando que os requisitos de informação sejam atendidos de forma eficiente (ELMASRI; NAVATHE, 2015).

Com base no modelo conceitual, desenvolvemos o modelo lógico do banco de dados, que detalha a estrutura e as relações entre as diferentes entidades de maneira mais técnica. Este modelo lógico, apresentado na Figura 4, foi projetado para garantir consistência e eficiência na manipulação dos dados. A seguir, descrevemos as tabelas presentes na modelagem lógica:

- **Tbl_Songs_Genius:** Armazena informações detalhadas sobre as músicas provenientes da base inicial, utilizando o ID do Genius como chave primária para garantir a identificação exclusiva de cada música. Além disso, inclui chaves estrangeiras que fazem referência às tabelas das fontes externas para integrar e relacionar dados adicionais associados a cada música.

- **Tbl_Songs_Spotify:** Contém propriedades específicas do áudio das músicas extraídas do Spotify.

- **Tbl_Songs_Lima:** Apresenta rótulos de gêneros obtidas no conjunto de dados Lima.

- **Tbl_Songs_Lastfm:** Armazena informações sobre as músicas disponíveis na plataforma Last.fm, incluindo dados como o título da música e MBID.

- **Tbl_Songs_Tags:** Classifica as *tags* associadas às músicas no Last.fm.

- **Tbl_Tags_Genres:** Apresenta a relação entre as *tags* e os gêneros musicais, permitindo a classificação das músicas por gêneros baseados nas *tags* associadas a elas.

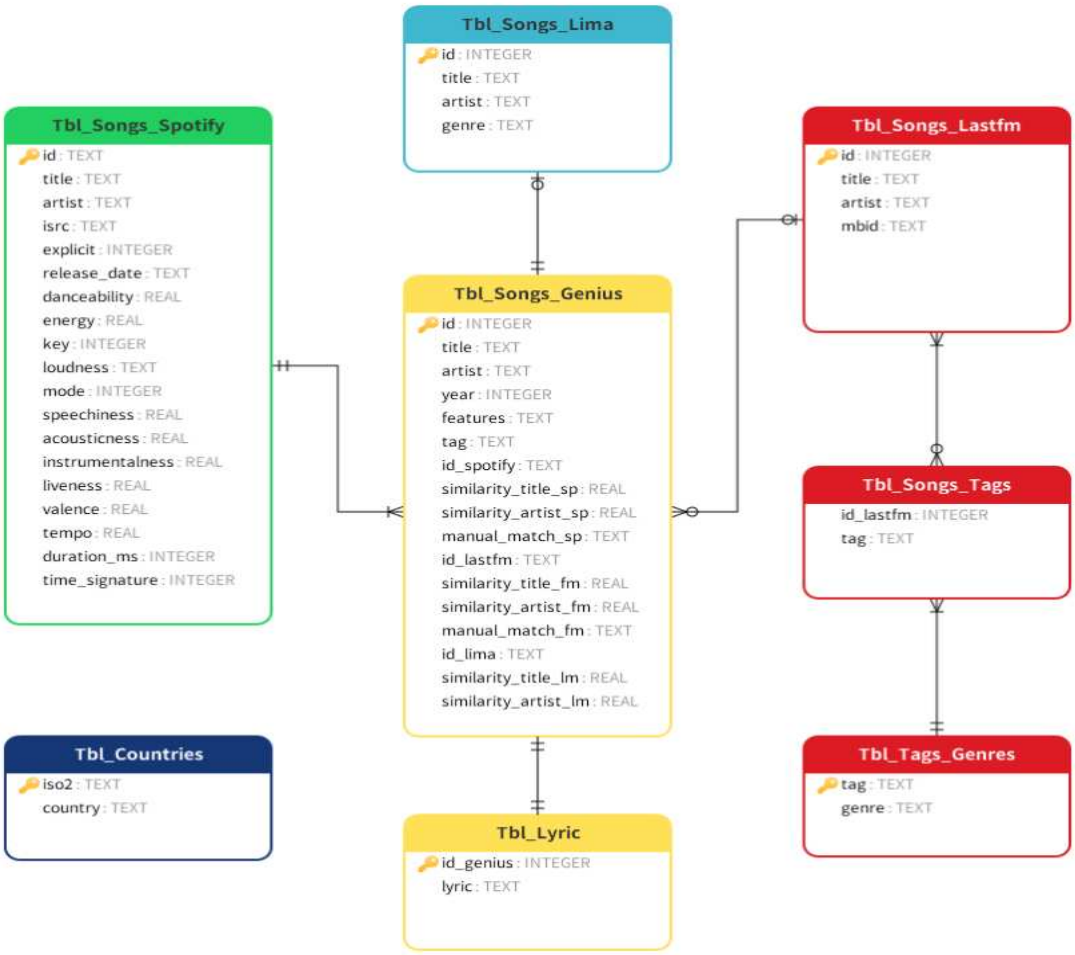

Figura 4 – Modelo Lógico do Banco de Dados. Disponível em: https://github.com/leticiatavaresds/BAMPORT/blob/main/Data%20Base%20Info/db_logic_model.png.

- **Tbl_Lyric:** Armazena as letras das músicas.
- **Tbl_Countries:** Armazena informações dos códigos ISO 3166-1 alpha-2 dos países.

Além de armazenar as informações básicas sobre as músicas, estabelecemos uma relação entre cada música e suas fontes externas, permitindo o registro de como os dados foram associados entre diferentes origens. Esta relação inclui atributos para medir a similaridade entre os títulos das músicas em cada base de dados e entre os nomes dos artistas. Também há um atributo que indica se os registros foram manualmente associados como referentes à mesma música.

É importante observar que, ao contrário das outras fontes, o Genius não foi categorizado como uma fonte externa, uma vez que seus dados já estavam incluídos no conjunto de dados inicial. Para mais detalhes sobre as variáveis e tabelas presentes em nossa base de dados, o dicionário de dados pode ser encontrado no Apêndice A.

4.6 CONJUNTO DE DADOS

A partir da base de dados construída, geramos um conjunto de dados para a análise de letras de músicas que inclui um total de 27.777 instâncias, representando uma amostra abrangente do corpus musical analisado, com 27.113 músicas únicas. A diferença entre o número total de instâncias e músicas únicas se deve a diferentes versões ou registros de uma mesma faixa no Spotify e no Genius.

Este conjunto inclui 14 atributos relacionados às métricas acústicas, obtidas através da API do Spotify, e um atributo textual com a letra de cada música. Adicionalmente, criamos 25 atributos *booleanos* que indicam a categorização das músicas em diferentes gêneros, com base nos rótulos presentes na base de dados obtidos das fontes Last.fm e Lima. A distribuição de músicas por gênero no conjunto de dados é apresentada na Tabela 10.

Tabela 10 – Número de músicas por gênero em nosso conjunto de dados.

GÊNERO	**Nº DE MÚSICAS**
mpb	6.667
rock	4.975
sertanejo	3.438
gospel	2.467
pop	2.222
samba	2.205
bossanova	1.598
pagode	1.367
forro	1.112
funk	741
axe	702
alternativo	631
jazz	630
rap	627
infantil	539
fado	501
hiphop	483
indie	462
jovem-guarda	387
punk	238
soul	210
eletronica	173
metal	163
velha-guarda	107
rnb	93

5 CONSTRUÇÃO DAS VARIÁVEIS DE PROCESSAMENTO DE LINGUAGEM NATURAL

Nossa base de dados apresentava uma limitação significativa: pouca informação detalhada sobre aspectos específicos de Processamento de Linguagem Natural sobre as letras de músicas. Para superar essa limitação e enriquecer a base com atributos relevantes de PLN, utilizamos o trabalho de Mayerl et al. (2020) como referência. Recriamos e adaptamos as variáveis descritas no estudo para o ambiente Python[1], versão 3.9, que foi nossa escolha preferencial para este projeto.

As variáveis de PLN usadas no estudo de Mayerl foram originalmente desenvolvidas com base no trabalho de Zangerle et al. (2018) e estavam implementadas em Java[2], com especificações disponíveis em um repositório no GitHub[3]. Portanto, todas as implementações descritas neste capítulo refletem a adaptação e tradução desses atributos de PLN para a linguagem Python, ajustando-os às necessidades da nossa análise.

As variáveis foram construídas em quatro classes principais, capturando diferentes aspectos do texto e permitindo uma análise abrangente e multifacetada:

- **Variáveis Lexicais**: Métricas que quantificam características relacionadas ao uso e à estrutura das palavras em textos.
- **Variáveis Linguísticas**: Focadas em aspectos linguísticos como o vocabulário e a estrutura da linguagem utilizada.
- **Variáveis Semânticas**: Relacionadas ao significado das palavras e suas interpretações.
- **Variáveis Sintáticas**: Métricas que descrevem a estrutura e organização das frases e sentenças.

Nas próximas seções, descreveremos minuciosamente as variáveis utilizadas nas análises, explicando sua implementação e relevância na análise textual. Focamos apenas nas variáveis mais importantes para a compreensão dos resultados e das conclusões do estudo. Para uma visão abrangente de todas as variáveis, os dicionários nos Apêndices B, C, D e E fornecem uma lista completa, enquanto o repositório do trabalho[4] apresenta uma descrição detalhada de todas as variáveis construídas.

[1] Linguagem Python disponível em: https://www.python.org/
[2] Linguagem Java disponível em: https://www.java.com/pt-BR/
[3] Repositório ALF200k disponível em: https://github.com/dbis-uibk/ALF200k
[4] Repositório do Trabalho disponível em: https://github.com/leticiatavaresds/BAMPORT

5.1 LIMPEZA DE DADOS

A criação das variáveis de PLN foi baseada nos textos das letras de músicas acessíveis pelo atributo “lyric” da base de dados. No entanto, esses textos frequentemente contêm caracteres especiais como pontuações e quebras de linha, o que pode prejudicar algumas análises. Para garantir a consistência e a qualidade dos dados, especialmente na criação de variáveis sensíveis à estrutura textual uniforme, desenvolvemos dois novos atributos que refinam esses textos. O primeiro atributo é uma versão semi-tratada da letra, que passou pelos seguintes processos:

- Eliminação de espaços extras no início e no fim do texto.
- Remoção de múltiplos espaços.
- Remoção da primeira linha caso contenha o termo “Letra:”.
- Exclusão de strings entre colchetes.

Esse atributo foi concebido para criação de variáveis que podem se beneficiar da presença de quebras de linha, caracteres especiais e letras maiúsculas. Os dois últimos passos do tratamento aplicado foram projetados para lidar com características específicas encontradas nos textos das letras. Em muitos casos, a primeira linha do texto começava com a palavra “Letra:” seguida pelo nome da música. Esta linha introdutória não faz parte da letra da música propriamente dita e pode interferir na análise se não for removida. Portanto, optamos por excluir essa linha inicial nas letras em que aparece.

Além disso, algumas letras de música continham marcações estruturais entre colchetes para indicar partes específicas, como o refrão. Se a maioria das letras tivesse essas marcações, poderíamos usá-las para criar variáveis, como a contagem de refrões. No entanto, como apenas uma pequena porcentagem delas apresentava essas marcações, optamos por remover qualquer texto contido entre colchetes. Assim, garantimos que nossa análise se concentre exclusivamente na letra da música, sem incluir elementos estruturais que não são considerados na análise de conteúdo textual.

Pensando nas variáveis que exigem uma estrutura textual uniforme para resultados mais precisos, criamos um segundo atributo com a letra completamente tratada, onde além dos tratamentos acima, também é feito:

- Remoção de caracteres especiais.
- Eliminação do caractere de nova linha.
- Conversão de todos os caracteres para minúsculas.

Esses processos visam assegurar que os dados estejam em um formato ideal para análise, minimizando potenciais distorções e otimizando a eficácia dos modelos aplicados posteriormente.

5.2 VARIÁVEIS LEXICAIS

No contexto de PLN, as variáveis lexicais são métricas que quantificam características relacionadas ao uso e à estrutura das palavras em textos focando no conjunto de palavras e expressões em um idioma ou corpus específico.

A relevância das variáveis lexicais reside na capacidade de revelar aspectos subjacentes dos textos que podem não ser imediatamente evidentes. Por exemplo, a razão de palavras únicas em um texto pode indicar a variedade vocabular de um autor, enquanto a contagem de linhas em branco pode oferecer uma melhor compreensão sobre a estrutura e o estilo de escrita (JURAFSKY; MARTIN, 2019). Além disso, métricas como o comprimento médio das palavras e a frequência de bigramas ou trigramas únicos podem refletir a complexidade linguística e a coesão textual (MANNING; SCHüTZE, 1999).

As variáveis que criamos se dividem em diferentes grupos, cada um capturando aspectos específicos da estrutura e do uso das palavras nas letras de músicas:

- Contagem de Linhas: Essas variáveis examinam a extensão e estrutura dos textos analisados.
- Razões de Pontuação e Dígitos: Este grupo de variáveis avalia a presença e a frequência de caracteres de pontuação e dígitos nos textos.
- Frequências por Minuto: As variáveis neste grupo calculam a frequência de palavras, linhas e caracteres por minuto de música.
- Tokenização e Variáveis de *Tokens*: Este conjunto de variáveis envolve a segmentação do texto em *tokens* e a análise dessas unidades.
- *Stopwords*: As variáveis relacionadas às *stopwords* quantificam a presença de palavras comuns e frequentemente não informativas, como artigos e preposições.

5.3 VARIÁVEIS SEMÂNTICAS

A semântica explora a influência de variáveis como contexto e uso sobre o significado das palavras. Por exemplo, “banco” pode referir-se a uma instituição financeira ou a um assento, dependendo do contexto. Além do significado literal, conotação e denotação também são importantes: enquanto a denotação é literal, a conotação inclui significados emocionais e culturais. A palavra “lar” por exemplo, denota um lugar onde se vive, mas conota sentimentos de conforto e segurança.

Além do significado literal e conotativo, a semântica envolve a interpretação de implicaturas, onde o significado é derivado de forma indireta ou implícita, e inferências. Palavras de negação, por exemplo, podem inverter o sentido de uma palavra, e intensificadores como "muito" aumentam a força do significado.

A análise de sentimentos baseada em léxicos desempenha um papel de destaque no estudo e na construção de variáveis semânticas, oferecendo uma estrutura sistemática para compreender as emoções e atitudes expressas em textos. Segundo Liu (2012), léxicos especializados contêm palavras associadas a valores de polaridade, classificando-as como positivas, negativas ou neutras. Esses léxicos permitem que modelos de análise de sentimento atribuam uma pontuação quantitativa aos textos, refletindo a intensidade e a direção emocional expressa em cada documento analisado.

Ao examinar a polaridade das palavras em um texto, podemos inferir o sentimento geral associado a ele. Ao utilizar léxicos como base, os modelos identificam e ponderam a contribuição de palavras específicas para o sentimento global do texto. Por exemplo, palavras como "amor" são tipicamente associadas a valores positivos, enquanto termos como "tristeza" têm uma conotação negativa. A aplicação desses léxicos envolve o cálculo de uma pontuação de sentimento ao somar e ponderar os valores de polaridade das palavras encontradas no texto, oferecendo uma métrica objetiva do sentimento predominante.

Em nossa pesquisa, replicamos todas as pontuações de sentimentos descritas no artigo (ZANGERLE et al., 2018), garantindo uma reprodução precisa do processo. Desenvolvemos duas pontuações de sentimento usando léxicos específicos: OpLexicon (SOUZA et al., 2012) e AFINN (NIELSEN, 2011). Além disso, aplicamos modelos baseados em léxico, como SentiStrength (THELWALL et al., 2010) e VADER (HUTTO; GILBERT, 2014), que utilizam algoritmos para atribuir pontuações de sentimento aos textos. Também criamos variáveis de frequência de temas utilizando o dicionário RID (*Regressive Imagery Dictionary*) (MARTINDALE, 1975).

5.3.1 Pontuação AFINN

Para aplicar esse modelo em textos em português, foi preciso encontrar uma versão traduzida do léxico. Felizmente, uma versão adaptada para o idioma português foi disponibilizada no GitHub[5] contendo 2.922 palavras e expressões traduzidas.

O processo de calcular a pontuação de sentimento para uma letra de música utilizando o léxico AFINN funciona da seguinte maneira: cada palavra ou expressão no texto é analisada para determinar seu impacto emocional. Ao somar todos os scores individuais, obtemos um total que reflete a intensidade emocional geral. Esse total é então dividido pelo número de palavras que tiveram uma pontuação atribuída, resultando em uma média ponderada que captura os sentimentos expressos ao longo da letra.

[5] Dicionário AFINN em português disponível em: https://github.com/dkocich/afinn-165-multilingual.

Para lidar com palavras compostas e expressões que podem estar presentes no léxico, adotamos uma abordagem que visa garantir precisão na avaliação. Implementamos a geração de n-gramas começando pelos quadrigramas e indo até os unigramas, pois as expressões compostas no léxico podem ter até quatro palavras. Essa estratégia permite identificar tanto palavras individuais quanto expressões compostas por até 4 palavras, como "muito bom" ou "de tirar o fôlego".

Começamos a análise desses n-gramas em ordem decrescente de n, o que significa que primeiro examinamos os quadrigramas, seguidos pelos trigramas, bigramas e unigramas. Para cada n-grama encontrado, consultamos o léxico AFINN para verificar se a expressão ou palavra está presente.

Para garantir que não contemos repetidamente n-gramas que compartilhem palavras já pontuadas em outros n-gramas, utilizamos um mecanismo de verificação de índices, mantendo um registro dos índices já processados. Assim que um n-grama é localizado no dicionário, marcamos imediatamente os índices de suas palavras como contabilizado, o que impede a contagem duplicada dessas palavras na pontuação final de sentimento. Isso assegura que cada palavra seja considerada apenas uma vez no cálculo final da pontuação de sentimento, seja individualmente ou dentro de uma expressão idiomática.

Por exemplo, ao analisar o texto "Sonhar é muito bom." nosso método tokeniza o texto em n-gramas, avaliando a expressão "muito bom" como uma unidade coesa de significado emocional. Dessa forma, apenas "muito bom" é contabilizado, e a palavra "bom" não é considerada separadamente.

Para exemplificar o cálculo do score de sentimento utilizando o modelo AFINN em uma parte da letra da música "Mania de Você" de Rita Lee, consideremos o trecho: "Você me dá água na boca, vestindo fantasias, tirando a roupa". Após tokenizar o trecho em n-gramas e aplicar o léxico AFINN, suponhamos que tenhamos as seguintes pontuações: "água na boca" (score +1), "fantasias" (score +2), "tirando a roupa" (score +3). Somando esses scores, teríamos um total de +6. Dividindo pelo número de palavras ou expressões pontuadas (três, incluindo as expressões compostas), obteríamos uma média ponderada de +2, que reflete a intensidade emocional geral desse trecho da música.

5.3.2 Pontuações VADER

Diferente do léxico AFINN, que se limita a uma lista de palavras e suas pontuações de sentimento, o VADER combina um léxico com um modelo de regras para análise de sentimento, como descrito na Seção 3.2. Utilizamos a biblioteca LeIA (ALMEIDA, 2018), uma adaptação do modelo VADER com léxicos em português, para calcular as pontuações positiva, neutra, negativa e o *compound score* fornecidas pelo modelo para cada letra de música.

Como o VADER leva em consideração a presença de pontuações e a repetição de letras para influenciar a pontuação de um texto, utilizamos um texto semi-limpo para análise.

Esse texto semi-limpo tem apenas a remoção de linhas vazias e de linhas não pertencentes à letra, como indicadores de verso, para garantir que a análise considere esses aspectos relevantes na atribuição de sentimento.

5.3.3 Regressive Imagery Dictionary

Para aplicar o RID ao nosso estudo em português, utilizamos dicionários[6] específicos traduzidos oficialmente. Esses dicionários contêm traduções e adaptações das palavras e expressões categorizadas no RID original, garantindo a precisão da análise em nosso idioma.

Utilizamos expressões regulares (regex) para realizar a correspondência de radicais, o que nos permite identificar termos que compartilham um radical comum, independentemente de suas variações morfológicas ou sintáticas. Para calcular a pontuação de cada subcategoria, começamos classificando todas as palavras relevantes do texto conforme as subcategorias definidas. Em seguida, somamos o número de termos pertencentes a cada subcategoria e dividimos pelo total de termos classificados, obtendo assim a pontuação correspondente a cada subcategoria.

Por exemplo, ao analisar um trecho da música "Trem das Cores" de Caetano Veloso, como "Eu vou pro mar / Pra ver você / Pra ver se o mar ainda tá lindo" o RID categorizaria palavras como "mar", "ver" e "lindo" em subcategorias relacionadas à natureza, percepções visuais e sentimentos gerais, respectivamente. Com um total de 5 palavras categorizadas, as pontuações de percepções visuais e natureza seriam 2/5 (considerando que "mar" e "ver" aparecem duas vezes cada), enquanto a de sentimentos gerais seria 1/5 (considerando "lindo" que aparece uma vez).

5.4 VARIÁVEIS SINTÁTICAS

As variáveis sintáticas são métricas que analisam a estrutura gramatical de um texto, utilizadas para entender a organização e a complexidade da escrita. Elas revelam padrões de uso de palavras e estruturas gramaticais, identificando o uso de diferentes categorias gramaticais como pronomes, verbos, substantivos, adjetivos e advérbios.

A análise da frequência de pronomes, por exemplo, pode mostrar como o autor se refere a personagens e objetos no texto. Assim, construímos variáveis para calcular a frequência de pronomes nas seis categorias principais: primeira pessoa do singular, primeira pessoa do plural, segunda pessoa do singular, segunda pessoa do plural, terceira pessoa do singular e terceira pessoa do plural.

Além de identificar e contar a frequência de cada pronome, calculamos também a proporção de pronomes autorreferenciados (primeira pessoa do singular/plural) em relação

[6] Dicionários RID disponíveis em: https://provalisresearch.com/products/content-analysis-software/wordstat-dictionary/regressive-imagery-dictionary/.

aos pronomes não autorreferenciados. Adicionalmente, determinamos a proporção de pronomes da primeira pessoa do singular comparados aos da segunda pessoa. Essas métricas oferecem uma visão mais aprofundada sobre como os autores se posicionam e se referem aos outros em seus textos.

Já para analisar as diferentes classes gramaticais, como verbos, substantivos, adjetivos e advérbios, utilizamos o modelo BrillTagger, baseado em regras e pré-treinado com o corpus *Mac-Morpho*. Este modelo nos permite etiquetar partes do discurso e calcular a frequência dessas classes em relação ao total de *tokens* da música.

Além disso, para codificar a dimensão temporal das músicas, implementamos uma variável que mede a fração de verbos no pretérito em relação ao total de verbos. Utilizamos um modelo de POS *tagging* para identificar os verbos e verificamos as terminações das palavras para confirmar a conjugação no tempo passado, baseando-nos em um dicionário específico para diferentes pessoas gramaticais. Esse processo inclui contar o número total de verbos, verificar quais estão no tempo passado e calcular a proporção correspondente, oferecendo uma visão mais clara das características sintáticas dos textos analisados.

5.5 VARIÁVEIS LINGUÍSTICAS

A linguística é o estudo científico da linguagem, abrangendo áreas como fonética, fonologia, sintaxe e pragmática, com o objetivo de entender a estrutura e o uso da linguagem. No contexto do português, a análise linguística utiliza variáveis quantitativas que capturam aspectos específicos da linguagem em um texto, como a frequência de palavras e características complexas como o uso de gírias.

Para analisar a diversidade e o uso de vocabulário agressivo em um texto, criamos uma variável que mede a proporção de palavras consideradas insultos ou xingamentos em relação ao número total de palavras, utilizando uma lista de palavras insultuosas encontrada em um repositório.

Além disso, utilizamos a técnica de lematização para facilitar a análise linguística e a compreensão semântica dos textos. A lematização reduz uma palavra à sua forma básica ou canônica. Por exemplo, no português, “correr”, “correu”, “correndo” e “correria” são lematizados como “correr”. Criamos então uma variável para indicar a proporção de lemas em relação ao número total de palavras. Para realizar a lematização das palavras e calcular o número de lemas em um texto, utilizamos a biblioteca spaCy e o modelo pré-treinado “*pt_core_news_sm*”.

5.6 CONJUNTO DE DADOS FINAL

Neste capítulo, foram criadas 146 variáveis de Processamento de Linguagem Natural distribuídas por tipo da seguinte forma:

- Decimal: 124 colunas, compreendendo variáveis numéricas contínuas, como proporções, frequências relativas e pontuações derivados de modelos de PLN.
- Inteiro: 19 colunas, consistindo em contagens que foram gerados a partir do texto das letras.
- *String*: 3 colunas, que armazenam informações relacionadas às análises de PLN, especificamente no que diz respeito à estrutura gramatical e sintática das letras das músicas, como rótulos de *Part-of-Speech* (POS).

Para concluir a preparação do corpus para análise, integrarmos as variáveis desenvolvidas neste capítulo ao conjunto de dados descrito na Seção 4.6, que já continha métricas acústicas e informações de gênero. O resultado é um conjunto de dados final com 27.777 instâncias e 198 atributos, conforme detalhado na Tabela 11. Este conjunto de dados final, salvo em um arquivo CSV[7] de aproximadamente 400 megabytes, permite uma análise abrangente das músicas e suas respectivas classificações de gênero que exploraremos essa integração e suas implicações no capítulo seguinte.

Tabela 11 – Sumário do Conjunto de Dados Final destinado à análise.

DESCRIÇÃO	**QUANTIDADE**
Total de Linhas	27.777
Músicas Únicas	27.113
Total de Atributos	198
Atributos de Metadados	13
Atributos de PLN	146
Atributos de Áudio	14
Atributos de Rótulos de Gênero	25

As atributos de Processamento de Linguagem Natural e de áudio foram organizadas em grupos temáticos com base em suas características específicas. A seguir, apresentamos esses grupos:

- Áudio: É composto por características de áudio como acústica, dançabilidade e valência, obtidas através da API do Spotify.
- Classes Gramaticais: Contém atributos que apresentam a frequência de algumas classes gramaticais e relacionadas à proporção de lemas em relação ao número total de *tokens*.

[7] Conjunto de Dados Final disponível em: https://github.com/leticiatavaresds/BAMPORT/blob/main/02_Analysis/final_database.zip

- Estatístico: Inclui características calculadas a partir do texto completo das letras, como contagem de *tokens*, contagem de linhas, proporção de *stopwords*, proporção de palavras novas e proporção de linhas repetidas.
- Explicitude: Apresenta atributos que analisam a presença de conteúdo explícito nas letras, como linguagem ofensiva ou referências a temas violentos, sexuais, ou de teor impróprio.
- Pronomes: Apresenta atributos que medem a frequência e proporção de pronomes em relação ao total de *tokens*.
- Sentimento: Inclui atributos derivados dos modelos de Processamento de Linguagem Natural que analisam o sentimento das letras.
- Tempo Estatístico: Contém atributos de frequência por minuto que fornecem uma visão dinâmica da densidade e ritmo das letras de música ao longo do tempo.

6 RESULTADOS EXPERIMENTAIS

Para determinar a importância dos diferentes tipos de recursos textuais que construímos para a tarefa de reconhecimento de gênero, e a extensão em que esses recursos se complementam, realizamos uma série de experimentos usando diferentes tipos de recursos e modelos de aprendizado de máquina. Neste capítulo, forneceremos os resultados dos nossos experimentos, bem como detalhes sobre os recursos e modelos de aprendizado de máquina usados.

Em cada experimento, selecionamos grupos específicos de variáveis e treinamos os modelos utilizando exclusivamente os atributos de cada grupo, a fim de analisar o impacto isolado de cada conjunto de características nas classificações. Além disso, realizamos testes utilizando todos os atributos dos grupos relacionados ao Processamento de Linguagem Natural. Por fim, consideramos os atributos construídos a partir do Processamento de Linguagem Natural com as características de áudio para avaliar o desempenho dos modelos quando todas as informações disponíveis foram utilizadas em conjunto.

6.1 REPRODUTIVIDADE

Visando facilitar a reprodução dos experimentos realizados neste estudo, procuramos ser o mais explicativos possível nas seções práticas. Todo o código-fonte está disponível no repositório do estudo[1]. A maioria dos arquivos está no formato .py, com alguns disponíveis como notebooks Python (.ipynb). O repositório está organizado da seguinte forma:

- Pasta "00_Data_Base": Contém os *scripts* para a criação da base de dados e base de dados criada compactada em um arquio .zip.
- Pasta "01_NLP": Inclui os códigos para a criação das variáveis de Processamento de Linguagem Natural (PLN) e os resultados das implementações compactados em arquivos .zip.
- Pasta "02_Analysis": Abrange os códigos dos experimentos, desde a implementação dos modelos até a geração de gráficos e matrizes para análise.

Este projeto foi desenvolvido integralmente em um único computador, utilizando a versão 3.9.19 do Python e a versão 2.91.1 do Visual Studio Code. As bibliotecas empregadas nos experimentos incluíram a Scikit-learn, na versão 1.5.1, e o TensorFlow, na versão 2.12.0. O experimento foi realizado em um ambiente controlado, com as seguintes especificações do computador:

[1] Repositório do Estudo disponível em: https://github.com/leticiatavaresds/BAMPORT.

- Sistema: Dell G15 5520
- Processador: Intel(R) Core(TM) i5-12500H 12th Gen @ 2.50GHz
- Memória RAM: 24GB (1x Crucial 16GB DDR4 4800MHz CL40 SODIMM e 1x Hynix 8GB DDR4 4800MHz CL40 SODIMM)
- Armazenamento: 1TB (CT1000P5PSSD8 NVMe Crucial 1024GB)
- Sistema Operacional: Windows 11 Home Single Language

A execução dos modelos no primeiro experimento demorou mais

6.2 REPLICAÇÃO DA METODOLOGIA DE CATEGORIZAÇÃO E AVALIAÇÃO DE MODELOS COM DADOS EM PORTUGUÊS

Iniciamos nossa investigação seguindo a metodologia do estudo de referência (MAYERL et al., 2020), mantendo a categorização dos atributos e os mesmos modelos de aprendizado de máquina com mesmos parâmetros. Nosso objetivo foi comparar o desempenho dos modelos em nossa base de dados em português e avaliar a consistência e eficácia dos métodos em um novo contexto linguístico.

6.2.1 Base de Dados

Optamos por não adotar a abordagem de classificação múltipla para este estudo. A classificação múltipla, na qual uma música pode ser atribuída a vários gêneros simultaneamente, pode complicar a interpretação dos resultados e a avaliação da eficácia dos modelos de aprendizado de máquina. Em vez disso, decidimos focar em músicas que pertencem a um único gênero, a fim de assegurar que o modelo se concentre na atribuição de um gênero específico.

Tabela 12 – Contagem de linhas por gênero musical na base de dados filtrada.

Gênero	Contagem
mpb	5000
rock	3827
sertanejo	3388
gospel	2467
samba	1570
pop	1330
pagode	1215
bossanova	914
alternativo	114
indie	100

Como resultado dessa decisão, o conjunto de dados final foi refinado para incluir apenas músicas com somente um gênero atribuído. Esse filtro reduziu o número de instâncias de 27.777 para 19.925. A Tabela 12 apresenta a distribuição final dos gêneros musicais após a aplicação desse filtro.

6.2.2 Atributos

Para categorizar os atributos, filtramos nosso conjunto de dados final para empregarmos as mesmas variáveis utilizadas pelo estudo (MAYERL et al., 2020). No entanto, não conseguimos replicar a construção das variáveis do grupo de "rima" para a base em português. Assim, dos cinco grupos de atributos usados no estudo referência, utilizamos os seguintes (a lista detalhada de atributos para cada categoria está disponível no Apêndice F):

- Estatístico: totalizando 31 características derivadas das letras.
- Tempo Estatístico: composto por 3 atributos que medem a frequência por minuto.
- Explicitude: contém um único atributo, um rótulo binário fornecido pela API do Spotify, que indica se a música possui letras explícitas.
- Áudio: engloba 10 características de áudio de alto nível obt

Além dos grupos mencionados, também calculamos dois tipos adicionais de características diretamente a partir dos textos brutos das letras das músicas, sem realizar qualquer pré-processamento além da remoção de caracteres especiais e *stopwords*:

- TF-IDF: Calculamos vetores TF-IDF utilizando n-gramas (sequências de palavras de um a três termos). Para limitar a complexidade e garantir relevância, restringimos o vetor resultante aos 2.000 n-gramas mais frequentes, que são aqueles com maior impacto e relevância no conjunto de dados analisado.
- LDA: Utilizamos a técnica de LDA para gerar vetores de características que capturam os tópicos abordados nas letras das músicas. Para nossos experimentos, configuramos a LDA para extrair 25 tópicos distintos, criando assim 25 atributos que representam esses temas principais. Cada um desses atributos é do tipo *float*, pois o LDA não retorna uma classificação binária para a presença ou ausência de um tópico, mas sim uma probabilidade ou proporção que indica o quanto cada tópico está presente em uma determinada letra.

6.2.3 Modelo de Linha de Base

No desenvolvimento deste trabalho, implementamos inicialmente um modelo de linha de base, que serviu como referência para avaliar se os outros modelos classificadores desenvolvidos posteriormente são capazes de fornecer melhores resultados do que apenas uma previsão aleatória.

Para esse fim, utilizamos o modelo DummyClassifier da biblioteca scikit-learn[2], que simula um classificador básico. Esse modelo faz previsões seguindo estratégias simples, como sempre prever a classe mais comum (*most_frequent*), realizar previsões aleatórias que respeitam a proporção das classes (*stratified*), ou atribuir probabilidades iguais a todas as classes (*uniform*). Testamos essas estratégias e a abordagem *stratified* demonstrou o melhor desempenho e foi, portanto, escolhida como nossa linha de base. Os resultados detalhados de cada estratégia estão disponíveis nas Tabelas do Apêndice H.

6.2.4 Modelos de Aprendizado Supervisionado

Em nossos experimnetos, testamos cinco algoritmos distintos: *K-nearest neighbors* (kNN), florestas aleatórias (RF), árvores extremamente aleatórias (ET), máquinas de vetores de suporte (SVM) e uma rede neural artificial (RNA).

Visando encontrar de maneira empírica, porém automatizada, os parâmetros mais adequados a cada algoritmo, realizamos uma busca em grade com validação cruzada quíntupla (com k = 5), utilizando a biblioteca *scikit-learn* para a implementação dos modelos kNN, RF, ET e SVM. Já o modelo de rede neural foi implementado com o uso do *TensorFlow*[3]. Para avaliar o desempenho dos modelos, utilizamos a métrica F1-Macro, indicada para lidar com bases de dados desbalanceadas, conforme destacado por (BAEZA-YATES; RIBEIRO-NETO, 1999).

Para garantir que os resultados fossem comparáveis e que o processo de treinamento e validação fosse conduzido de maneira uniforme, estabelecemos três parâmetros fixos para todos os modelos:

- random_state = 42: Valor que serve como semente para geração dos números aleatórios utilizados no algoritmo. Fixamos o valor para garantir a reprodutibilidade dos resultados, permitindo que as mesmas divisões de dados sejam utilizadas em cada execução.

- cv = 5: O parâmetro informa a quantidade de vezes em que será realizada a validação cruzada.

- n_jobs = -1: O parâmetro estabelece quantos processadores são disponibilizados para realizar o processamento paralelo de tarefas durante o treinamento dos mode-

[2] Biblioteca Scikit-learn disponível em: https://scikit-learn.org/.
[3] Biblioteca TensorFlow disponível em: https://www.tensorflow.org/.

los. A configuração utilizada indica que podem ser utilizados tantos processadores quanto estiverem disponíveis.

Com esses parâmetros definidos, conduzimos experimentos para cada técnica de aprendizado de máquina de forma individual. Inicialmente, exploramos todas as combinações de parâmetros para cada conjunto de atributos, conforme apresentado na Tabela 13.

Tabela 13 – Parâmetros testados para cada modelo a fim de encontrar a configuração com maior pontuação F1-Macro.

	Parâmetros	**Nº Combinações**
Árvores Extras	`n_estimators` = [10, 100, 300]	3
Floresta Aleatória	`n_estimators` = [10, 100, 300]	3
Kneighbors	`n_neighbors`: [3, 4, 5, 10], `weights`: [''`distance`''], `p`: [1, 2]	8
SVM	`estimator_C`: [0.1, 0.5, 1.0, 2.0, 5.0], `estimator_loss`: [''`squared_hinge`'']	5
Rede Neural	`dense_sizes_list` = [(32, 32), (64, 64)] `dropout_rates` = [0.1] `epochs` = 50 `batch_size` = 32	2

Para os modelos de Árvores Extremamente Aleatórias e Floresta Aleatória, avaliamos diferentes quantidades de estimadores (*n_estimators*). No modelo K-Nearest Neighbors, exploramos combinações dos seguintes parâmetros: número de vizinhos (*n_neighbors*), tipo de ponderação (*weights*) e a potência do parâmetro de distância (p). Para o modelo SVM, variamos o parâmetro C (*estimator_C*), que controla a penalidade por erros de classificação, e o *estimator_loss*, que especifica a função de perda a ser utilizada. Neste caso, utilizamos a função de perda "squared hinge", que penaliza erros de classificação de forma quadrática, ou seja, penaliza mais severamente erros maiores, incentivando o modelo a distinguir melhor as classes, mesmo em situações onde as margens de decisão são estreitas. No modelo de Rede Neural, testamos diferentes combinações do número de unidades nas camadas densas (*dense_sizes_list*), a taxa de *dropout* (*dropout_rates*), o número de épocas (*epochs*) e o tamanho do *batch* (*batch_size*). Esses parâmetros foram ajustados para identificar a configuração que alcançasse a maior pontuação F1-Macro.

6.2.5 Resultados

Ao iniciar a seção de resultados, destacamos que, no modelo de linha de base, a melhor pontuação F1-Macro foi de 0.095, utilizando a técnica estratificada. Esse valor nos serviu como ponto de referência para avaliar o desempenho dos outros modelos desenvolvidos.

Para avaliar o impacto das variáveis de PLN construídas a partir das letras de músicas, foi adotada a seguinte estratégia: primeiro, treinamos os modelos apenas com os atributos

Tabela 14 – Pontuação F1-Macro dos testes realizados em cada grupo de atributos nos experimentos.

Grupo de Atributos	ET	RNA	RF	SVM	kNN	Melhor
linha de base	-	-	-	-	-	**0.095**
áudio	0.357	0.342	**0.369**	0.217	0.349	**0.369**
estatístico	**0.225**	0.207	0.217	0.157	0.215	**0.225**
tempo estatístico	**0.132**	0.130	0.128	0.084	0.128	**0.132**
explicitude	**0.045**	**0.045**	**0.045**	**0.045**	0.016	**0.045**
tf-idf	0.097	0.100	0.093	0.107	**0.111**	**0.111**
lda	0.096	0.040	**0.104**	0.045	0.100	**0.104**
PLN	0.164	0.138	**0.170**	0.159	0.111	**0.170**
PLN +áudio	0.256	0.229	**0.274**	0.242	0.114	**0.274**
PLN(sem tf-idf e lda)	**0.241**	0.222	0.233	0.166	0.225	**0.241**
PLN +áudio (sem tf-idf e lda)	0.342	**0.385**	0.354	0.292	0.325	**0.385**

de áudio (grupo áudio). Em seguida, realizamos o treinamento considerando todos os atributos, ou seja, utilizando todo o conjunto de dados contendo as métricas de áudio com as variáveis de PLN (grupo PLN + áudio). Para investigar o efeito isolado de cada grupo de variáveis de PLN, treinamos também os modelos separadamente para cada um desses grupos de atributos de PLN, de modo a avaliar sua contribuição individual.

Nos primeiros experimentos, o grupo de atributos de áudio analisado pelo modelo de Floresta Aleatória alcançou a maior pontuação F1-Macro, com um valor de 0.369, o que destaca a maior eficácia dessas características em comparação às textuais. Dentre as características textuais, o grupo "estatístico" obteve a maior pontuação, 0.225, utilizando o modelo de Árvores Extremamente Aleatórias. Em contrapartida, o grupo "explícito", composto por um único atributo booleano, registrou a menor pontuação, de 0.045. Esse resultado é condizente com a natureza do atributo, que, por ser booleano, apresenta pouca variabilidade, limitando a capacidade preditiva do modelo. Dessa forma, é razoável que um conjunto de atributos tão simples não consiga superar o desempenho da linha de base, que já reflete uma distribuição equilibrada das classes.

Outros grupos textuais, como TF-IDF e LDA, apesar de superarem a linha de base, apresentaram pontuações relativamente baixas, em torno de 0.100. Além disso, a utilização de características textuais com as de áudio (PLN + áudio) não resultou em uma melhoria, e, na verdade, piorou a pontuação.

Devido aos baixos resultados obtidos com TF-IDF e LDA, que também aumentaram substancialmente o número de atributos (mais de 2000), decidimos testar os modelos nos grupos "PLN" e "PLN + áudio" sem incluir esses grupos de atributos. Essa abordagem tinha o objetivo de verificar se a remoção de TF-IDF e LDA poderia melhorar os resultados globais. De fato, a exclusão desses atributos levou a uma melhora: a pontuação F1-Macro

no grupo "PLN" aumentou para 0.241, e no grupo "PLN + áudio" para 0.385, demonstrando um desempenho levemente superior ao resultado obtido com as características de áudio isoladas.

6.3 AVALIAÇÃO DO DESEMPENHO DOS MODELOS COM NOVOS GRUPOS DE ATRIBUTOS

Para aprimorar o desempenho dos nossos algoritmos, além de remover os grupos TF-IDF e LDA, conduzimos novos experimentos mantendo os mesmos modelos e parâmetros, mas testando novos grupos de atributos.

Primeiro, adicionamos duas variáveis ao grupo "explecitude" visando melhorar o baixo desempenho do grupo. Essas variáveis, desenvolvidas durante o processo de construção das variáveis, analisam o linguajar utilizado nas letras. A primeira variável representa a proporção de palavras ofensivas na letra, enquanto a segunda quantifica a proporção de gírias utilizadas.

Além da adição dessas variáveis, realizamos um processo de seleção de atributos, analisando todas as outras características textuais apresentadas no Capítulo 5. Eliminamos os atributos altamente correlacionados e criamos mais três grupos textuais (a lista completa de atributos para cada grupo utilizados nos experimentos dessa seção pode ser encontrada no Apêndice G):

- Classes Gramaticais: contém no total 13 atributos.
- Pronomes: inclui um total de 7 atributos.
- Sentimento: inclui 17 atributos derivados de modelos de PLN.

Tabela 15 – Pontuação F1-Macro dos Experimentos para cada um dos novos atributos textuais.

Grupo de Atributos	**ET**	**RNA**	**RF**	**SVM**	**kNN**	**Melhor**
linha de base	-	-	-	-	-	**0.095**
áudio	0.357	0.345	**0.369**	0.217	0.349	**0.369**
estatístico	**0.225**	0.207	0.217	0.157	0.215	**0.225**
tempo estatístico	**0.132**	0.130	0.128	0.084	0.128	**0.132**
classes gramaticais	**0.195**	0.154	0.193	0.118	0.189	**0.195**
explicitude	0.089	0.082	**0.095**	0.063	0.081	**0.095**
pronomes	0.163	0.118	**0.166**	0.080	0.164	**0.166**
sentimento	**0.243**	0.176	**0.243**	0.160	0.236	**0.243**
PLN	0.281	**0.295**	0.281	0.226	0.271	**0.295**
PLN + áudio	0.363	**0.415**	0.354	0.335	0.340	**0.415**

Os novos experimentos mostraram mudanças significativas no desempenho dos modelos. O grupo explicitude, agora composto por três variáveis, ultrapassou a linha de base, mostrando ser melhor de uma classificação aleatória. E dessa vez, o grupo de atributos "audio" com os atributos textuais (grupo "PLN + audio"), resultou nos melhores desempenhos. O modelo de Rede Neural destacou-se, alcançando uma pontuação de 0.415. Árvores Extremamente Aleatórias e Floresta Aleatória mantiveram uma performance próxima para "PLN + audio". Por outro lado, SVM continuou a apresentar desempenhos inferior em comparação com os outros modelos.

6.4 APRIMORAMENTO FINAL: SELEÇÃO DE GÊNEROS NACIONAIS

Pensando em uma última melhoria, executamos os modelos com os mesmos parâmetros e atributos da segunda etapa de aprimoramento. Contudo, selecionamos apenas os cinco gêneros nacionais mais representativos na base de dados, excluindo gêneros universais como pop e rock. Essa abordagem resultou em um conjunto de dados menor, com um total de 14.968 linhas, porém mais balanceado e voltado para gêneros culturais nacionais, o que contribuiu para uma melhor representatividade dos estilos musicais locais. A distribuição final dos gêneros após essa filtragem está apresentada na Tabela 16.

Tabela 16 – Distribuição dos gêneros musicais na conjunto de dados após filtragem pelos 5 gêneros brasileiros mais populados.

Gênero	Contagem
mpb	6215
sertanejo	3407
gospel	2467
samba	1660
pagode	1219

Com a mudança na base de dados, precisamos recalcular a linha de base. Dessa vez, a técnica com o melhor desempenho foi a uniforme, apresentando uma pontuação F1-Macro de 0.198. Esse aumento em relação ao valor anterior de 0.095 sugere que a redução do número de classes ajudou a equilibrar melhor a distribuição entre elas, permitindo que a linha de base capturasse com mais precisão as nuances da classificação, mesmo em um cenário simplificado.

Analisando os resultados, observamos que o desempenho dos modelos melhorou consideravelmente. O modelo de Rede Neural destacou-se, alcançando a maior precisão de 0.599 ao utilizar o conjunto de atributos "PLN + áudio". Em comparação, o melhor desempenho isolado ao utilizar apenas os atributos de áudio foi de 0.555, evidenciando a contribuição positiva da integração dos atributos textuais com os atributos de áudio para a melhora na performance do modelo.

Tabela 17 – Pontuação F1-Macro dos Experimentos para cada um dos Atributos Textuais para a base filtrada pelos 5 gêneros nacionais mais populados.

Grupo de Atributos	ET	RNA	RF	SVM	kNN	Melhor
linha de base	-	-	-	-	-	**0.198**
áudio	0.530	0.526	**0.555**	0.376	0.510	**0.555**
estatístico	0.341	**0.345**	**0.345**	0.262	0.335	**0.345**
tempo estatístico	**0.265**	0.208	0.244	0.167	0.260	**0.265**
classes gramaticais	0.269	0.240	0.272	0.182	**0.292**	**0.292**
explicitude	0.201	0.172	**0.201**	0.144	0.196	**0.201**
sentimento	**0.360**	0.315	0.352	0.272	0.336	**0.360**
pronomes	0.272	0.211	**0.276**	0.174	**0.277**	**0.277**
PLN	0.398	**0.478**	0.410	0.385	0.435	**0.478**
PLN + áudio	0.491	**0.599**	0.535	0.517	0.526	**0.599**

6.5 ANÁLISE DO MELHOR MODELO

A Rede Neural demonstrou o melhor desempenho entre todos os métodos avaliados, alcançando uma pontuação de 59.9% em F1-Macro com a base filtrada por gêneros brasileiros, conforme detalhado na seção anterior. Esse desempenho superior foi obtido tanto com os grupos de atributos definidos pelo estudo de referência quanto com aqueles selecionados posteriormente. Dada a sua performance destacada, decidimos utilizar a Rede Neural para uma análise mais aprofundada.

Nos experimentos detalhados a seguir, o conjunto de dados foi dividido em duas partes: 80% para treinamento e 20% para teste. Tanto os dados de treinamento quanto os de teste foram normalizados utilizando a função *StandardScaler* da biblioteca *Scikit-Learn*.

6.5.1 Experimento I

Primeiro, analisamos o desempenho da Rede Neural com a configuração que proporcionou o melhor resultado. Os parâmetros utilizados foram: tamanho do lote (`batch_size`) de 32, número de épocas (`epochs`) de 50, tamanhos das camadas densas (`dense_sizes`) de 32 e 32 unidades, e taxa de *dropout* (`dropout_rate`) de 0.1. Para otimizar o desempenho da Rede Neural, utilizamos a função de ativação `SELU` nas camadas densas ocultas e a função `softmax` na camada de saída (Seção 2.5).

Para avaliar o modelo, geramos as curvas de aprendizado e de perda durante as fases de treinamento e validação. A Figura 5 ilustra essas curvas. No gráfico à esquerda, que exibe a acurácia ao longo das épocas, percebe-se que a acurácia no treinamento continua aumentando de forma constante, alcançando valores próximos a 0.740. No entanto, a acurácia de validação se estabiliza em torno de 0.685 após as primeiras épocas e varia, sem apresentar crescimento significativo. Essa diferença entre as curvas de treinamento e

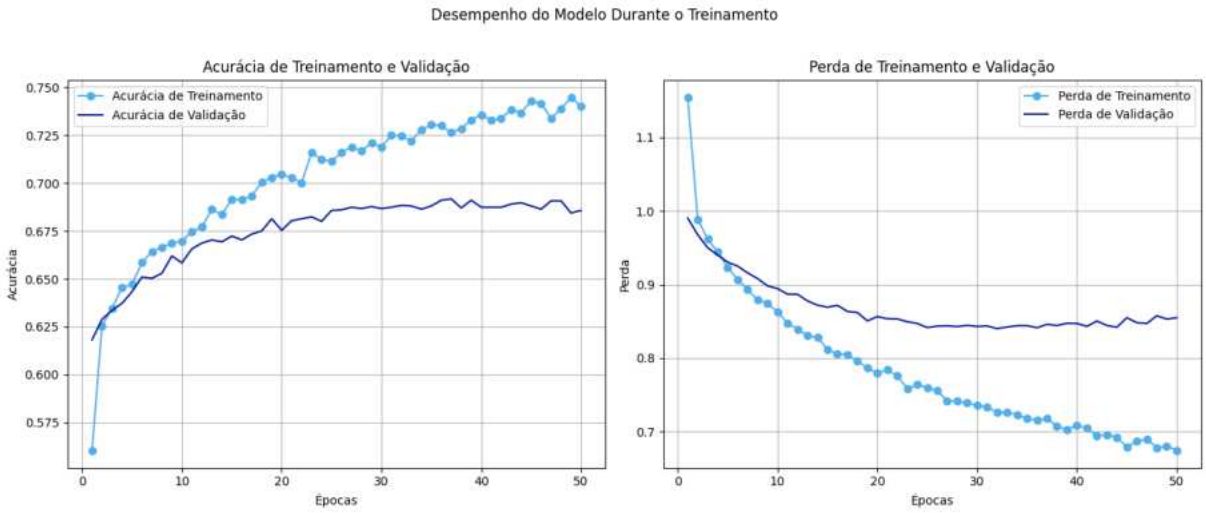

Figura 5 – Evolução da acurácia (esquerda) e perda (direita) do modelo ao longo de 50 épocas de treinamento. As curvas demonstram a progressão do modelo em termos de acurácia e perda tanto para os dados de treinamento quanto de validação.

validação é um sinal de *overfitting*.

O gráfico à direita, que mostra a perda ao longo do treinamento e da validação, confirma essa observação. A perda no treinamento diminui consistentemente, sugerindo que o modelo está se ajustando aos dados de treinamento. No entanto, a perda na validação, embora diminua inicialmente, logo se estabiliza em um valor maior, com oscilações que indicam que o modelo está começando a memorizar os padrões específicos do conjunto de treinamento, ao invés de aprender características gerais que possam ser aplicadas a novos dados.

Essa situação de *overfitting* sugere que o modelo pode estar excessivamente complexo ou que as técnicas de regularização, como o *dropout*, não estão sendo aplicadas de forma suficiente para evitar esse ajuste excessivo (GOODFELLOW; BENGIO; COURVILLE, 2016).

6.5.2 Experimento II

Para mitigar o *overfitting* observado na implementação anterior, realizamos ajustes nos hiperparâmetros do modelo, aumentando o valor de *dropout* em 50%, de 0.1 para 0.15, e adicionando uma regularização L2 com valor de $1e-3$ (0.001), conforme recomendado por (CHOLLET, 2017). Essas modificações visaram reduzir a complexidade do modelo e, consequentemente, melhorar sua capacidade de generalização. Os resultados obtidos após a aplicação desses novos parâmetros são ilustrados na Figura 6, onde, embora tenha ocorrido uma piora nos valores de acurácia e perda em comparação à implementação anterior, observamos uma redução do *overfitting*. Essa melhoria foi particularmente evidente na convergência dos valores de perda entre o treino e a validação, indicando uma

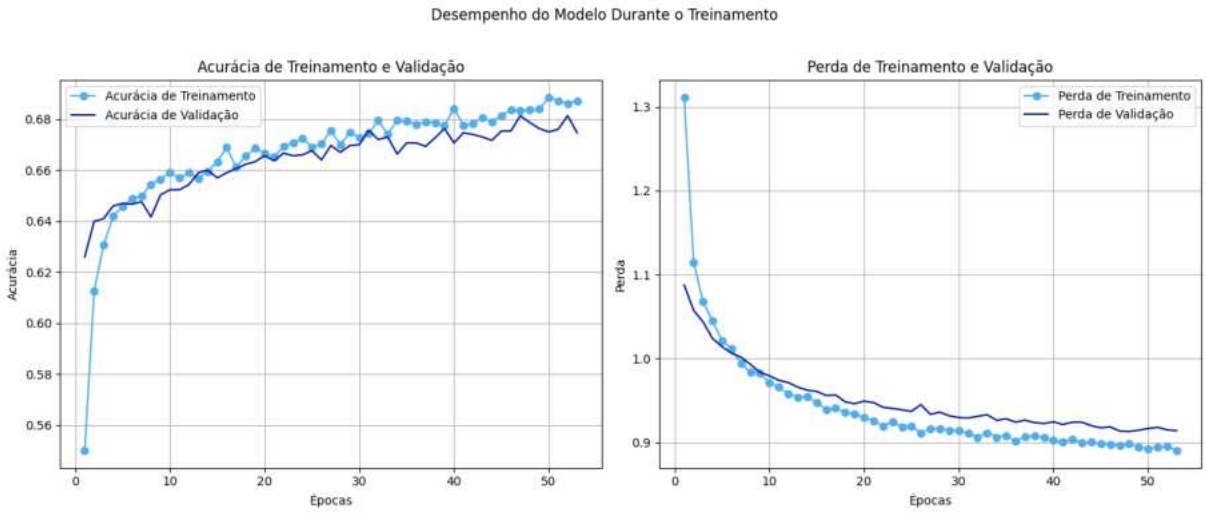

Figura 6 – Evolução da acurácia (esquerda) e perda (direita) do modelo com o valor de *dropout* igual 0.15 e regularização L2 com valor de $1e-3$ (0.001) ao longo de 53 épocas. As curvas demonstram a progressão do modelo em termos de acurácia e perda tanto para os dados de treinamento quanto de validação.

aproximação maior e, portanto, uma melhor generalização do modelo.

Além de avaliar o *overfitting*, também geramos a matriz de confusão, apresentada na Figura 7, onde o eixo horizontal representa as classes (gêneros) previstas pelo modelo, e o eixo vertical corresponde às classes reais das músicas. Essa matriz permite uma avaliação do desempenho do modelo em cada categoria, destacando as áreas de maior confusão entre diferentes classes.

Complementando essa análise, geramos um relatório de performance contendo as métricas de *recall*, precisão e F1-Macro, que fornecem uma visão quantitativa do desempenho do modelo em termos de equilíbrio entre os diferentes aspectos da classificação. Além disso, para facilitar a interpretação desses resultados, criamos uma matriz visual, apresentada na Figura 8.

Analisando os resultados obtidos, podemos destacar alguns pontos importantes. Em relação à precisão, a classe gospel se sobressai ao apresentar uma precisão significativamente maior (74%) em comparação com as outras classes, que variam de 49% à 63%. Isso indica que o modelo é mais confiável ao identificar corretamente instâncias de gospel, enquanto as outras classes apresentam mais falsas previsões. Já o *recall* apresenta uma grande variância entre as classes, oscilando de 83% (MPB) a 27% (samba). Esse dado sugere que o modelo tem maiores dificuldades em identificar corretamente todas as instâncias de algumas classes, especialmente no caso do samba, onde a maioria das instâncias reais não é reconhecida corretamente.

No que diz respeito ao F1-Macro, as classes MPB, sertanejo e gospel apresentam valores relativamente próximos (0.75, 0.69 e 0.71, respectivamente), o que indica um bom equilíbrio entre precisão e *recall* para essas classes. Entretanto, as classes pagode e samba

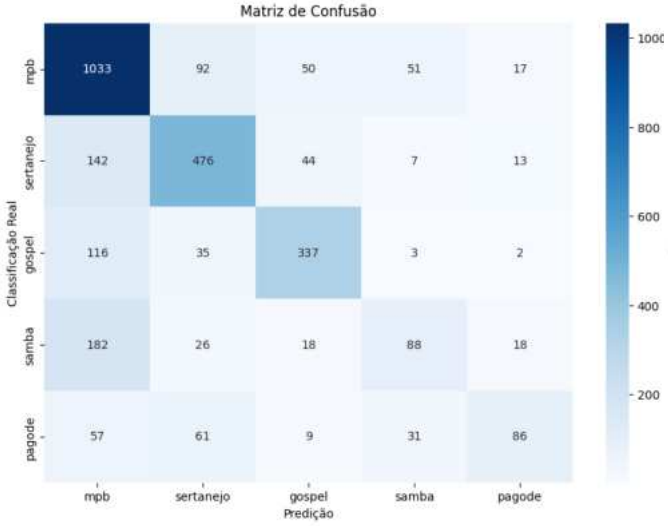

Figura 7 – Matriz de Confusão para as classes

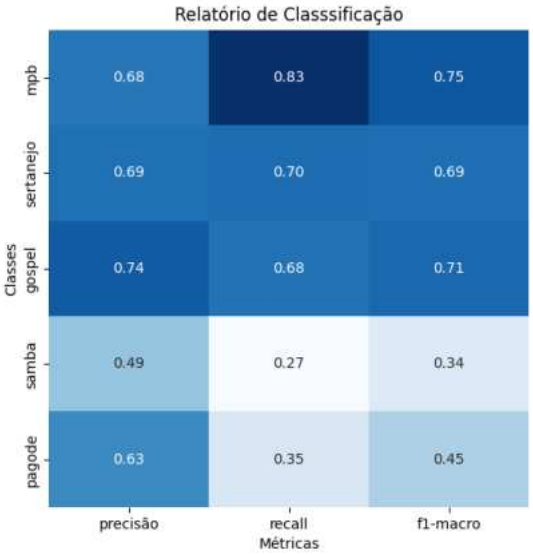

Figura 8 – Relatório com as métricas de precisão, *recall* e F1-Macro para cada classe

mostram valores de F1-Macro mais baixos (0.45 e 0.34, respectivamente), o que sugere a necessidade de melhorias para que o modelo possa lidar melhor com essas categorias.

Além disso, a matriz de confusão revela tendências específicas de confusão entre algumas classes. Por exemplo, a classe samba foi erroneamente classificada 51 vezes como Pagode, enquanto MPB apresentou erros de classificação frequentes com sertanejo (142 vezes), samba (182) e gospel (116 vezes). Essas sobreposições indicam que as características das músicas nesses gêneros podem ter similaridades que confundem o modelo, especialmente entre MPB, sertanejo e samba.

Para aprofundar nossa compreensão dos fatores que influenciam o desempenho do modelo, além das análises anteriores de métricas e matrizes de confusão, avaliamos por último a importância dos atributos utilizando a técnica de permutação. Essa técnica mede a relevância de cada atributo ao observar o impacto na performance do modelo quando os valores de um atributo específico são embaralhados aleatoriamente. Se a permutação de um atributo resulta em uma diminuição significativa na precisão do modelo, esse atributo é considerado importante para a previsão (BREIMAN, 2001). Utilizamos a biblioteca *Scikit-learn* para aplicar a técnica e empregamos a métrica F1-Macro para avaliar a precisão em cada teste.

Conforme a Figura 9, as características de áudio, como "*Loudness*" e "*Valence*", são as mais influentes na classificação de músicas, destacando-se em relação às demais. Isso sugere que as propriedades sonoras desempenham um papel predominante no modelo de classificação. Outras características de áudio, como "*Energy*" e "*Speechiness*", também mostram importância relevante, embora em menor grau, reforçando a ideia de que aspectos específicos do áudio possuem grande influência para a categorização.

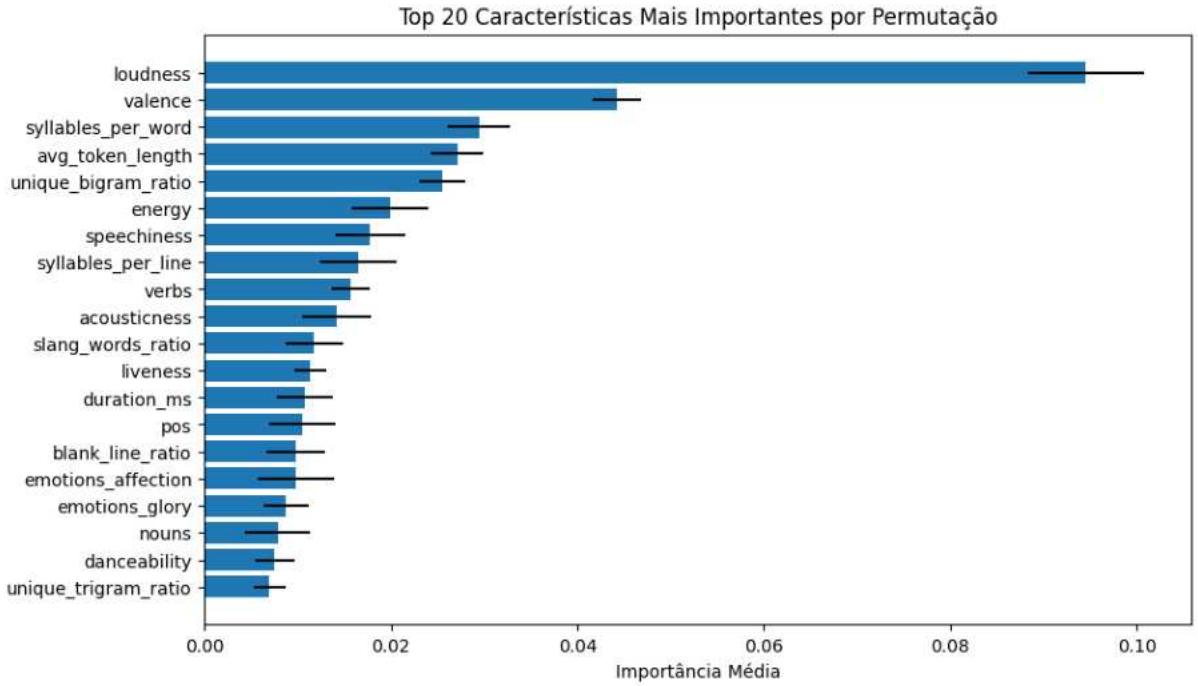

Figura 9 – Importância dos atributos para o modelo de rede neural avaliada por permutação, destacando os 20 atributos com maior relevância. As barras horizontais representam a importância média de cada característica para a previsão, enquanto os traços pretos indicam a variação da importância estimada.

Por outro lado, características textuais, especialmente aquelas relacionadas à estrutura das letras, como "*syllables_per_word*", "*avg_token_length*" e "*unique_bigram_ratio*", também apresentam uma importância considerável. Isso indica que a complexidade e a variedade nas letras têm um impacto notável quando combinadas com as características de áudio. Além disso, a presença de "*slang_words_ratio*" como uma característica relevante sugere que o uso de gírias ou palavras explícitas nas letras pode influenciar a classificação das músicas.

As características associadas a sentimentos e emoções, como "*emotions_glory*" e "*emotions_affection*", apresentam uma importância menor em comparação com as características de áudio e as textuais mais estruturais. Isso pode indicar que, embora as emoções expressas nas letras tenham algum impacto, elas são menos determinantes do que as características de áudio ou aspectos mais técnicos das letras.

6.6 TEMPO DE EXECUÇÃO DOS EXPERIMENTOS

Em termos de execução, o primeiro experimento apresentou maior tempo de processamento devido ao elevado número de atributos considerados na construção dos modelos, principalmente devido à inclusão de técnicas como TF-IDF e LDA. Entre os modelos, o K-Nearest Neighbors (KNN) e a Rede Neural foram os mais demorados, levando cerca de 5 horas para serem executados, considerando o tempo total da rodagem dos modelos para

cada grupo de variáveis individualmente, além da integração dos grupos de atributos de PLN e áudio.

No segundo experimento, que envolveu uma redução significativa no número de atributos, e no terceiro, em que além dessa redução, o conjunto de dados estava mais filtrado, o tempo de execução foi substancialmente mais rápido. Nessas condições, mesmo os modelos mais lentos completaram a execução em cerca de 2 hora, o que tornou o processo muito mais eficiente em comparação ao experimento inicial.

6.7 COMPARAÇÃO DOS RESULTADOS

Uma limitação enfrentada em nossa análise de comparação foi a indisponibilidade integral da base de dados utilizada por Mayerl et al. (2020), bem como a impossibilidade de reproduzir os códigos fornecidos. Essas restrições dificultaram uma investigação mais aprofundada das diferenças entre os estudos, especialmente no que tange ao desempenho de certos grupos de atributos.

Primeiramente, é importante destacar as diferenças entre as bases de dados utilizadas. A nossa base de dados inicial, que compreendeu 19.925 músicas distribuídas em 10 classes, adotou uma classificação única para cada música. Em contraste, a base de dados de Mayerl et al. (2020) continha 35.045 músicas e 16 classes, permitindo a classificação *multi-label*. Essas escolhas metodológicas foram deliberadas em nosso estudo para simplificar o modelo e focar na precisão da classificação.

No que diz respeito à categorização dos atributos, utilizamos variáveis semelhantes às empregadas no estudo de referência. No entanto, não foi possível replicar integralmente a construção das variáveis relacionadas ao grupo de rima devido às particularidades da língua portuguesa.

A comparação dos resultados entre os nossos modelos e o estudo de Mayerl apresentou algumas diferenças. O desempenho do modelo de linha de base em nosso estudo, com uma pontuação F1-Macro de 0.095, foi inferior ao resultado obtido no estudo Mayerl et al. (2020), que alcançou 0.156. Essa diferença sugere uma maior complexidade ou variabilidade em nossa base de dados.

Notamos também uma discrepância no desempenho do grupo de explicitude. Enquanto no estudo de Mayerl et al. (2020) esse grupo demonstrou um desempenho superior ao da linha de base, indicando que a explicitude contribuiu positivamente para a classificação, em nosso estudo, o grupo não conseguiu superar o desempenho da linha de base. Embora essa discrepância possa estar relacionada às características específicas do nosso conjunto de dados, não foi possível investigar detalhadamente as razões para essa diferença devido à indisponibilidade da base de dados do outro estudo.

O grupo de atributos de áudio, quando analisado pelo modelo de Floresta Aleatória, obteve a maior pontuação F1-Macro de 0.356. Este resultado corrobora as descobertas de

Mayerl et al. (2020), que também atribuíram uma forte contribuição das características de áudio para o desempenho dos modelos. No entanto, ao juntar os atributos de áudio com os textuais, o grupo "PLN", que considera todos os grupos compostos por variáveis construídas a partir das letras de músicas, não apresentou uma melhoria considerável em relação ao grupo de áudio, com a pontuação F1-Macro permanecendo próxima à observada por Mayerl et al. (2020). A principal diferença foi a pontuação consideravelmente mais alta obtida pelo nosso grupo de áudio, onde a contribuição desses atributos foi notavelmente robusta em nossa análise.

A inclusão das técnicas TF-IDF e LDA prejudicou o desempenho dos modelos, em contraste com o estudo de Mayerl et al. (2020), onde essas técnicas foram benéficas. Excluindo TF-IDF e LDA, e testando a inclusão de outros grupos de atributos, observamos uma melhora notável, com a pontuação F1-Macro aumentando de 0.170 para 0.241 no grupo "PLN" e de 0.274 para 0.385 no grupo "PLN + áudio". Esse resultado destaca a importância de selecionar atributos com cuidado para evitar a complexidade excessiva e o *overfitting*.

Nos experimentos subsequentes, incorporamos variáveis ao grupo de explicitude e criamos três novos grupos textuais: pronomes, classes gramaticais e sentimento. Essas adições resultaram em uma melhora significativa, com o resultado da integração de atributos sendo 0.409, maior que o grupo de áudio sozinho. Esse avanço reflete a eficácia de utilizar um maior número de características textuais, algo que não foi amplamente explorado no estudo de referência, o qual utilizou apenas algumas das variáveis textuais disponíveis.

Finalmente, ao filtrar a base de dados para focar nos gêneros nacionais mais representativos, observamos uma melhoria substancial no desempenho dos modelos, indicando que a abordagem adaptada ao contexto cultural brasileiro proporcionou uma representação mais precisa dos estilos musicais locais, em contraste com a abordagem mais generalista de Mayerl et al. (2020).

Em síntese, embora o estudo de referência tenha obtido bons resultados com a integração de atributos de áudio e características textuais, nosso estudo evidenciou a importância de ajustar modelos e atributos ao contexto linguístico e cultural específico da base de dados. A inclusão de novos atributos textuais e a escolha de gêneros musicais permitiram alcançar um desempenho superior, especialmente com o uso de modelos de Rede Neural, refletindo uma adaptação mais eficaz às particularidades do nosso conjunto de dados.

7 CONCLUSÃO E TRABALHOS FUTUROS

A principal contribuição deste trabalho foi a criação da BAMPORT, uma base de dados multimodal robusta composta por músicas em português, que integra letras das músicas, métricas de áudio e outros conjuntos de dados relevantes, juntoa acriação de conjunto de dados construído a partir da base com 27.777 músicas e 198 atributos, sendo 13 atributos de metadados, 14 métricas de áudio, 25 rótulos de gênero e 146 variáveis de Processamento de Linguagem Natural extraídas das letras.

O estudo abordou todas as etapas da construção dessa base, desde a coleta e integração dos dados até a sua estruturação e organização. Embora a análise da eficácia dos recursos textuais na classificação de gêneros musicais tenha sido um aspecto adicional, o foco principal residiu na construção da base de dados e na geração de um conjunto abrangente de atributos. Este conjunto inclui atributos de Processamento de Linguagem Natural, métricas acústicas e rótulos, permitindo uma avaliação da interação entre os atributos textuais e acústicos.

Apesar dos avanços, a base de dados construída ainda é relativamente pequena e desbalanceada. Expandir a coleção de músicas e incluir uma gama mais ampla de estilos e características pode aprimorar a identificação de características distintas e melhorar a capacidade de generalização dos modelos. Para isso, os *scripts* desenvolvidos para a construção da base de dados podem ser adaptados e reutilizados para facilitar essa expansão. Futuras pesquisas devem focar na coleta de mais dados, considerando tanto a variedade de estilos quanto a especialização em estilos específicos, aproveitando a flexibilidade e escalabilidade dos *scripts* existentes.

A combinação de recursos textuais com atributos de áudio levou a um aumento na pontuação F1-Macro, evidenciando a importância desses atributos. No entanto, esse resultado foi alcançado apenas após filtrar o conjunto de dados para desconsiderar músicas de gêneros internacionais, como pop e rock, bem como gêneros nacionais com poucas instâncias. Esses ajustes ajudaram a tornar a base de dados mais equilibrada e representativa da diversidade musical brasileira, proporcionando uma base mais sólida para futuras análises e desenvolvimentos.

Durante o estudo, construímos 146 variáveis a partir das letras das músicas, mas 72 delas foram descartadas na seleção de atributos. Muitas dessas variáveis descartadas apresentavam uma alta proporção de zeros, como a maioria das variáveis criadas pelo modelo RID. A análise revelou que algumas características das selecionadas impactam mais o desempenho do modelo do que outras. Focar na criação e melhoria de características importantes, além de uma seleção mais rigorosa, pode levar a melhores resultados e reduzir o impacto de atributos menos relevantes, minimizando o *overfitting*.

Além disso, é importante destacar que não foi realizado um processo rigoroso de ve-

rificação das *tags* de gênero obtidas por meio da plataforma Last.fm, atribuídas pelos usuários. Isso pode comprometer a qualidade e a precisão dos rótulos na base de dados, impactando a confiabilidade das análises e a eficácia dos modelos. A validação e o refinamento das tags de gênero representam áreas críticas a serem exploradas em pesquisas futuras, a fim de garantir a integridade e a precisão dos dados.

A implementação de atributos líricos relacionados à rima foi limitada pela falta de recursos e ferramentas adequadas para a análise de fonemas em português. Além disso, modelos de Processamento de Linguagem Natural, como o *Vader*, enfrentaram dificuldades devido a dicionários desatualizados para o português, comprometendo a análise precisa de sentimentos e nuances nas letras. Investir no desenvolvimento de ferramentas de PLN mais adaptadas ao português, com dicionários mais abrangentes e métodos avançados de análise fonética, seria altamente benéfico.

Outro desafio identificado foi a diferenciação entre classes semelhantes, como MPB e sertanejo, que apresentaram uma significativa sobreposição de características. Técnicas mais refinadas são necessárias para distinguir entre essas classes, e métodos como aprendizado por transferência e técnicas avançadas de pré-processamento podem ajudar a melhorar a discriminação entre categorias.

Finalmente, a exploração de modelos híbridos, que combinem redes neurais com métodos baseados em árvores de decisão ou técnicas de aprendizado por amostras, pode unir os pontos fortes de cada abordagem e melhorar a acurácia geral dos modelos.

REFERÊNCIAS

ALMEIDA, R. J. A. **LeIA - Léxico para Inferência Adaptada**. [S.l.]: GitHub, 2018. https://github.com/rafjaa/LeIA.

ALTMAN, N. S. **An Introduction to Statistical Learning with Applications**. [S.l.]: Springer, 1992.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval: The Concepts and Technology Behind Search**. Boston, MA: Addison-Wesley, 1999.

BILENKO, M. **Learnable Similarity Functions and Their Application to Record Linkage and Clustering**. Monografia (Doctoral Dissertation Proposal) — University of Texas at Austin, 2003. Disponível em: http://hdl.handle.net/2152/2681.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. [S.l.]: Springer, 2006.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. **Journal of Machine Learning Research**, MIT Press, v. 3, p. 993–1022, 2003.

BREIMAN, L. Random forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 2001.

BREIMAN, L. et al. Classification and regression trees. **Wadsworth international group**, v. 37, 1984.

BRILL, E. A simple rule-based part of speech tagger. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the workshop on Speech and Natural Language**. [S.l.], 1992. p. 112–116.

BRILL, E. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. **Computational linguistics**, MIT Press, v. 21, n. 4, p. 543–565, 1995.

BRILL, E.; MOORE, R. C. An improved error model for noisy channel spelling correction. In: **Proceedings of the 38th Annual Meeting on Association for Computational Linguistics**. [S.l.: s.n.], 2000. p. 286–293.

BURGES, C. J. A tutorial on support vector machines for pattern recognition. **Data Mining and Knowledge Discovery**, Springer, v. 2, n. 2, p. 121–167, 1998.

CARUANA, R.; NICULESCU-MIZIL, A. An empirical comparison of supervised learning algorithms. In: ACM. **Proceedings of the 23rd International Conference on Machine Learning**. [S.l.], 2006. p. 161–168.

CHIEN, D. H. J. S.; LEE, G. M. H.; YANG, H. S. S. Music genre classification: A review. **IEEE Transactions on Multimedia**, v. 20, n. 8, p. 2041–2055, 2018.

CHOLLET, F. **Deep Learning with Python**. Shelter Island, NY: Manning Publications, 2017.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine Learning**, Springer, v. 20, n. 3, p. 273–297, 1995.

DICE, L. R. Measures of the amount of ecologic association between species. **Ecology**, v. 26, n. 3, p. 297–302, 1945.

DIETTERICH, T. G. Ensemble methods in machine learning. **Multiple Classifier Systems**, Springer, p. 1–15, 2000.

DOMINGOS, P. A few useful things to know about machine learning. **Communications of the ACM**, ACM, v. 55, n. 10, p. 78–87, 2012.

ELMASRI, R.; NAVATHE, S. B. **Fundamentals of Database Systems**. [S.l.]: Pearson, 2015.

FELL, M.; SPORLEDER, C. Lyrics-based analysis and classification of music. In: **Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers**. [S.l.: s.n.], 2014. p. 620–631.

FELLEGI, I. P.; SUNTER, D. A. A theory for record linkage. **Journal of the American Statistical Association**, v. 64, n. 328, p. 1183–1210, 1969. Disponível em: https://www.jstor.org/stable/2283318.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. **The Elements of Statistical Learning**. [S.l.]: Springer, 2001.

GEURTS, P.; ERNST, D.; WEHENKEL, L. Extremely randomized trees. **Machine learning**, Springer, v. 63, n. 1, p. 3–42, 2006.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. [S.l.]: MIT press, 2016.

GRIFFITHS, T. L.; STEYVERS, M. Finding scientific topics. **Proceedings of the National Academy of Sciences**, National Academy of Sciences, v. 101, n. suppl 1, p. 5228–5235, 2004.

GUO, L.; GU, Z.; LIU, T. **Music Genre Classification via Machine Learning**. 2017.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2nd. ed. [S.l.]: Springer, 2009.

HERZOG, T. N.; SCHEUREN, F. J.; BROWN, G. **Data Quality and Record Linkage Techniques**. Springer, 2007. Disponível em: https://www.springer.com/gp/book/9780387747894.

HUTTO, C.; GILBERT, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: **Proceedings of the Eighth International Conference on Weblogs and Social Media (ICWSM-14)**. [S.l.: s.n.], 2014. p. 216–225.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing**. [S.l.]: Pearson, 2019.

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. **International Joint Conference on Artificial Intelligence**, 1995.

KUHN, M.; JOHNSON, K. **Applied Predictive Modeling**. [S.l.]: Springer, 2013.

KUNCHEVA, L. I. A theoretical study on six classifier fusion strategies. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 24, n. 2, p. 281–286, 2002.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LIMA, R. de A. et al. Brazilian lyrics-based music genre classification using a blstm network. In: RUTKOWSKI, L. et al. (Ed.). **Artificial Intelligence and Soft Computing**. Cham: Springer International Publishing, 2020. p. 525–534. ISBN 978-3-030-61401-0.

LIU, B. Sentiment analysis and opinion mining. **Synthesis Lectures on Human Language Technologies**, v. 5, n. 1, p. 1–167, 2012.

LOWRANCE, R.; WAGNER, R. A. An extension of the string-to-string correction problem. **Journal of the ACM**, v. 22, n. 2, p. 177–183, 1975.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. **Introduction to Information Retrieval**. Cambridge: Cambridge University Press, 2008.

MANNING, C. D.; SCHüTZE, H. **Foundations of Statistical Natural Language Processing**. [S.l.]: MIT Press, 1999.

MARTINDALE, C. **Regressive imagery dictionary [Computer software]**. Wordstat. Provalis Research, Canada, 1975. Disponível em: https://provalisresearch.com/products/content-analysissoftware/wordstat-dictionary/regressive-imagery-dictionary/.

MARTINDALE, C. Regressive imagery dictionary: Theory and application. **Psychological Bulletin**, v. 108, n. 3, p. 441–461, 1990.

MASTUB, D. A.; LACERDA, W. S. **Sistema de Integração de Dados: Um Estudo de Caso sobre Vinhos**. Páginas p. Monografia (Trabalho de conclusão de graduação) — Universidade Federal do Rio de Janeiro, 2021. Disponível em: http://hdl.handle.net/11422/14644.

MAYERL, M. et al. Comparing lyrics features for genre recognition. 2020. Disponível em: https://aclanthology.org/2020.nlp4musa-1.15.pdf.

MITCHELL, T. M. **Machine Learning**. [S.l.]: McGraw Hill, 1997.

NAVARRO, G. A guided tour to approximate string matching. **ACM computing surveys (CSUR)**, ACM New York, NY, USA, v. 33, n. 1, p. 31–88, 2001.

NIELSEN, F. Årup. **AFINN: A new word list for sentiment analysis**. 2011. https://github.com/fnielsen/afinn. Accessed: 2024-07-03.

OLIVEIRA, M. de; FILHO, J. B. e S. Classificação de gêneros a partir de letras de músicas em português. In: **Anais do XIV Simpósio Brasileiro de Tecnologia da Informação e da Linguagem Humana**. Porto Alegre, RS, Brasil: SBC, 2023. p. 43–52. ISSN 0000-0000. Disponível em: https://sol.sbc.org.br/index.php/stil/article/view/25436.

QUINLAN, J. R. Induction of decision trees. **Machine learning**, Springer, v. 1, n. 1, p. 81–106, 1986.

RAMOS, J. Using tf-idf to determine word relevance in document queries. In: **Proceedings of the First Instructional Conference on Machine Learning**. [S.l.: s.n.], 2003. v. 242, n. 1, p. 133–142.

RIPLEY, B. D. **Pattern Recognition and Neural Networks**. [S.l.]: Cambridge University Press, 1996.

RUDER, S. An overview of gradient descent optimization algorithms. **arXiv preprint arXiv:1609.04747**, 2016.

SALTON, G.; BUCKLEY, C. Term-weighting approaches in automatic text retrieval. **Information Processing & Management**, Elsevier, v. 24, n. 5, p. 513–523, 1988.

SCHöLKOPF, B.; SMOLA, A. J. **Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond**. [S.l.]: MIT Press, 2002.

SCHöLKOPF, B.; SMOLA, A. J. **Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond**. [S.l.]: MIT Press, 2002.

SIDDIQI, M. H.; SIDDIQI, M. A. K-nearest neighbor classification algorithm. **Journal of Computer Science**, v. 1, n. 2, p. 16–20, 2005.

SOUZA, J. G. M. et al. A study on the combination of machine learning and lexicon based approaches for sentiment analysis in portuguese. In: **Proceedings of the 24th International Conference on Computational Linguistics (COLING)**. Mumbai, India: [s.n.], 2012.

SRIVASTAVA, N. et al. Dropout: A simple way to prevent neural networks from overfitting. **Journal of Machine Learning Research**, Microtome Publishing, v. 15, n. 1, p. 1929–1958, 2014.

STEVENS, K. et al. Exploring topic coherence over many models and many topics. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning**. [S.l.], 2012. p. 952–961.

SøRENSEN, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. **Biologiske Skrifter / Kongelige Danske Videnskabernes Selskab**, 1948.

THELWALL, M. et al. Sentiment strength detection in short informal text. **Journal of the American Society for Information Science and Technology**, v. 61, n. 12, p. 2544–2558, 2010.

VARMA, S.; SIMON, R. Bias in error estimation when using cross-validation for model selection. **BMC bioinformatics**, BioMed Central, v. 7, n. 1, p. 91, 2006.

WINKLER, W. E. The state of record linkage and current research problems. **Statistics of Income Division, Internal Revenue Service Publication R99/04**, 1999.

WOLD, E. et al. Content-based classification, search and retrieval of audio. **IEEE Multimedia**, IEEE, v. 3, n. 3, p. 27–36, 1996.

ZANGERLE, E. et al. Alf-200k: Towards extensive multimodal analyses of music tracks and playlists. In: **Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2018**. Cham: Springer, 2018. p. 584–590. ISBN 978-3-319-76941-7. Disponível em: https://dbis-informatik.uibk.ac.at/sites/default/files/2018-04/ecir-2018-alf.pdf.

APÊNDICE A – DICIONÁRIO DO BANCO DE DADOS

Tabela 18 – Dicionário da tabela Tbl_Song_Genius

Tbl_Song_Genius	
Coluna	**Descrição**
id	Identificador da música no Genius
title	Nome da música, como na base do Genius
artist	Nome do artista principal, como na base do Genius
year	Ano de lançamento da música, como na base do Genius
features	Nome dos artistas participantes, como na base do Genius
tag	Gênero da música de acordo com a classificação do Genius
id_spotify	Foreign Key para a tabela Tbl_Song_Spotify
similarity_title_sp	Score de similaridade do título da música retornado pela API do Spotify com o presente na base Genius, retornado pelo algoritmo de comparação
similarity_artist_sp	Score de similaridade do artista da música retornado pela API do Spotify com o presente na base Genius, retornado pelo algoritmo de comparação
manual_match_sp	boolean
id_lastfm	Foreign Key para a tabela Tbl_Song_Lastfm
similarity_artist_fm	Score de similaridade do título da música retornado pela API do LastFm com o presente na base Genius, retornado pelo algoritmo de comparação
similarity_title_fm	Score de similaridade do título da música retornado pela API do LastFm com o presente na base Genius, retornado pelo algoritmo de comparação
manual_match_fm	boolean

Tabela 19 – Dicionário da tabela Tbl_Lyric

Tbl_Lyric	
Coluna	**Descrição**
id_genius	Identificador da música no Genius
lyric	Letra completa da música

Tabela 20 – Dicionário da tabela Tbl_Songs_Lastfm

Tbl_Song_Lastfm	
Coluna	**Descrição**
id	Identificador artificial para música do LastFM
title	Nome da música, como retornado pela API do LastFm
artist	Nome do artista, como retornado pela API do LastFm
mbid	Identificador da faixa exclusivo utilizado no banco de dados MusicBrainz, como retornado pela API do LastFm

Tabela 21 – Dicionário da tabela Tbl_Songs_Tags

Tbl_Songs_Tags	
Coluna	**Descrição**
id_lastfm	Identificador artificial para música do LastFM
tag	Tag atribuída por usuários à música no LastFm

Tabela 22 – Dicionário da tabela Tbl_Tags_Genres

Tbl_Tags_Genres	
Coluna	**Descrição**
tag	Tag atribuída por usuários no LastFm
genre	Nome do gênero musical

Tabela 23 – Dicionário da tabela Tbl_Songs_Lima

Tbl_Song_Lima	
Coluna	**Descrição**
id	Identificador artificial para música do conjunto de dados Lima
title	Nome da música, como encontrado no conjunto de dados Lima
artist	Nome do artista, como encontrado no conjunto de dados Lima
genre	Nome do gênero musical, como encontrado no conjunto de dados Lima

Tabela 24 – Dicionário da tabela Tbl_Countries

Tbl_Countries	
iso2	Código ISO 3166-1 Alpha-2
country	Nome do País

Tabela 25 – Dicionário da tabela Tbl_Songs_Spotify

Tbl_Songs_Spotify	
Coluna	**Descrição**
id	Identificador da música no Spotify
title	Nome da música, como retornado pela API do Spotify
artist	Nome do artista, como retornado pela API do Spotify
isrc	Código de registro ISRC da música, como retornado pela API do Spotify
explicit	Indicador se a faixa tem ou não letras explícitas
release_date	Data de lançamento da música, como retornado pela API do Spotify
danceability	Descreve o quão dançante uma faixa é, com base em uma combinação de elementos musicais. Um valor de 0.0 indica baixa propensão para dança, enquanto 1.0 representa alta propensão para dança.
energy	Medida que varia de 0.0 a 1.0, representando a intensidade e atividade perceptível de uma faixa.
key	A tonalidade em que a faixa está. Se nenhuma chave for detectada, o valor será -1.
loudness	O volume geral de uma faixa em decibéis (dB). Os valores normalmente variam entre -60 e 0 dB.
mode	Indica a modalidade (maior ou menor) de uma faixa. Maior é representado por 1 e menor é 0.
speechiness	Probabilidade de presença de palavras faladas em uma faixa.
acousticness	Uma medida de confiança variando de 0.0 a 1.0, indicando a probabilidade de a faixa ser acústica.
instrumentalness	Probabilidade de a faixa não conter vocais.
liveness	Probabilidade de ter a presença de público na gravação.
valence	Uma medida de 0.0 a 1.0 que descreve a positividade musical transmitida por uma faixa.
tempo	O andamento geral estimado de uma faixa em batidas por minuto (BPM).
duration_ms	Duração da faixa em milissegundos.
time_signature	Uma fórmula de compasso estimada, variando de 3 a 7 indicando fórmulas de compasso de "3/4"a "7/4".

APÊNDICE B – TABELA DE VARIÁVEIS LEXICAIS.

Abaixo encontram-se todas as variáveis lexicais construídas.

Tabela 26 – Descrição das variáveis lexicais construídas para análise textual.

Recurso	**Descrição**
token_count	Total de tokens
unique_token_ratio	Nº de tokens únicos / Nº de tokens
unique_bigram_ratio	Nº de bigramas únicos / Nº de bigramas
unique_trigram_ratio	Nº de trigramas únicos / Nº de trigramas
repeat_word_ratio	(Nº de tokens - Nº de tokens únicos) / Nº de tokens
avg_token_length	Nº de caracteres/ Nº de tokens
hapax_legomenon_ratio	Nº de tokens que aparecem apenas uma vez / Nº de tokens
dis_legomenon_ratio	Nº de tokens que aparecem apenas duas vezes / Nº de tokens
tris_legomenon_ratio	Nº de tokens que aparecem apenas três vezes / Nº de tokens
unique_tokens_per_line	Nº de tokens únicos / Nº de linhas
average_tokens_per_line	Nº de tokens / Nº de linhas
line_count	Total de linhas
unique_line_count	Nº de linhas únicas
blank_line_count	Nº de linhas em branco
blank_line_ratio	Nº de linhas em branco / Nº de linhas
repeat_line_ratio	(Nº de linhas - Nº de linhas únicas) / Nº de linhas
digits	Nº de dígitos
exclamation_marks	Nº de pontos de exclamação
question_marks	Nº de pontos de interrogação
colons	Nº de dois-pontos
semicolons	Nº de pontos e vírgulas
quotes	Nº de aspas
commas	Nº de vírgulas
dots	Nº de pontos finais
hyphens	Nº de hífens
stopwords_ratio	Nº de stop words / Nº de tokens
stopwords_per_line	Nº de stop words / Nº de linhas

APÊNDICE C – TABELA DE VARIÁVEIS LINGUÍSTICAS.

Abaixo encontram-se todas as variáveis linguísticas construídas.

Tabela 27 – Descrição das variáveis linguísticas construídas para análise textual.

Variável	**Descrição**
uncommon_words_ratio	Nº de palavras agressivas / Nº de tokens
slang_words_ratio	Nº de gírias / Nº de tokens
unique_uncommon_words_ratio	Nº de palavras agressivas únicas / Nº de tokens
lemmas_ratio	Nº de lemas / Nº de tokens
syllables_per_line	Nº de sílabas / Nº de linhas
syllables_per_word	Nº de sílabas / Nº de palavras
syllable_variation	Desvio padrão do Nº de sílabas por palavra / Nº de linhas
block_count	Nº total de blocos
average_block_size	Nº total de linhas em blocos / block_count
blocks_per_line	block_count / Nº total de linhas
repetitivity	Nº total de linhas em blocos / Nº total de linhas
block_reduplication	Nº de blocos únicos / block_count

APÊNDICE D – TABELA DE VARIÁVEIS SEMÂNTICAS.

Abaixo encontram-se todas as variáveis semânticas construídas.

Tabela 28 – Descrição das variáveis semânticas construídas para análise textual.

Variável	**Descrição**
positive_mood	(Soma da Pontuação de sentimento postivo retornado pelo modelo SentiStrength para cada linha)/Nº de linhas
negative_mood	(Soma da Pontuação de sentimento negativo retornado pelo modelo SentiStrength para cada linha)/Nº de linhas
neutral_mood	(Soma da Pontuação de sentimento neutro retornado pelo modelo SentiStrength para cada linha)/Nº de linhas
opinion_score	(Soma das pontuações das palavras encontradas no léxico OpLexicon) / Nº de palavras encontradas no léxico OpLexicon
afinn_score	(Soma das pontuações das palavras encontradas no léxico AFINN) / Nº de palavras encontradas no léxico AFINN
neg	Pontuação de sentimento postivo retornado pelo modelo VADER
neu	Pontuação de sentimento negativo retornado pelo modelo VADER
pos	Pontuação de sentimento neutro retornado pelo modelo VADER
compound	Nº de tokens que aparecem apenas três vezes / Nº de tokens
word_count	Nº de tokens classificados em qualquer subcategoria do RID
primary_need_orality	Nº de tokens classificados na subcategoria do RID “Oral” / word_count
primary_need_anality	Nº de tokens classificados na subcategoria do RID “Anal” / word_count
primary_need_sex	Nº de tokens classificados na subcategoria do RID “Sexo” / word_count
primary_sensation_touch	Nº de tokens classificados na subcategoria do RID “Toque” / word_count

Variável	Descrição
primary_sensation_taste	Nº de tokens classificados na subcategoria do RID "Paladar" / word_count
primary_sensation_odor	Nº de tokens classificados na subcategoria do RID "Olfato" / word_count
primary_sensation_general	Nº de tokens classificados na subcategoria do RID "Sensação Geral" / word_count
primary_sensation_sound	Nº de tokens classificados na subcategoria do RID "Audição" / word_count
primary_sensation_vision	Nº de tokens classificados na subcategoria do RID "Visão" / word_count
primary_sensation_cold	Nº de tokens classificados na subcategoria do RID "Frio" / word_count
primary_sensation_hard	Nº de tokens classificados na subcategoria do RID "Dureza" / word_count
primary_sensation_soft	Nº de tokens classificados na subcategoria do RID "Maciez" / word_count
primary_defensive_passivity	Nº de tokens classificados na subcategoria do RID "Passividade" / word_count
primary_defensive_voyage	Nº de tokens classificados na subcategoria do RID "Viagem" / word_count
primary_defensive_symbol_move	Nº de tokens classificados na subcategoria do RID "Movimento Aleatório" / word_count
primary_defensive_symbol_diffus	Nº de tokens classificados na subcategoria do RID "Difusão" / word_count
primary_defensive_symbol_chaos	Nº de tokens classificados na subcategoria do RID "Caos" / word_count
primary_rg_unknown	Nº de tokens classificados na subcategoria do RID "Desconhecimento" / word_count
primary_rg_timelessness	Nº de tokens classificados na subcategoria do RID "Eternidade" / word_count
primary_rg_consciousness_alt	Nº de tokens classificados na subcategoria do RID "Alteração de Consciência" / word_count
primary_rg_brink_passage	Nº de tokens classificados na subcategoria do RID "Passagem Limítrofe" / word_count
primary_rg_narcissism	Nº de tokens classificados na subcategoria do RID "Narcisismo" / word_count
primary_rg_concreteness	Nº de tokens classificados na subcategoria do RID "Concretude" / word_count
primary_icarian_imagery_ascend	Nº de tokens classificados na subcategoria do RID "Subida" / word_count

Variável	Descrição
primary_icarian_imagery_height	Nº de tokens classificados na subcategoria do RID "Altura" / word_count
primary_icarian_imagery_descent	Nº de tokens classificados na subcategoria do RID "Descida" / word_count
primary_icarian_imagery_depth	Nº de tokens classificados na subcategoria do RID "Profundidade" / word_count
primary_icarian_imagery_fire	Nº de tokens classificados na subcategoria do RID "Fogo" / word_count
primary_icarian_imagery_water	Nº de tokens classificados na subcategoria do RID "Água" / word_count
secondary_abstraction	Nº de tokens classificados na subcategoria do RID "Abstração" / word_count
secondary_social_behavior	Nº de tokens classificados na subcategoria do RID "Comportamento Social" / word_count
secondary_instrumental_behavior	Nº de tokens classificados na subcategoria do RID "Comportamento Instrumental" / word_count
secondary_restraint	Nº de tokens classificados na subcategoria do RID "Restrição" / word_count
secondary_order	Nº de tokens classificados na subcategoria do RID "Ordem" / word_count
secondary_temporal_references	Nº de tokens classificados na subcategoria do RID "Referências Temporais" / word_count
secondary_moral_imperative	Nº de tokens classificados na subcategoria do RID "Imperativo Moral" / word_count
emotions_positive_affect	Nº de tokens classificados na subcategoria do RID "Afeto Positivo" / word_count
emotions_anxiety	Nº de tokens classificados na subcategoria do RID "Ansiedade" / word_count
emotions_sadness	Nº de tokens classificados na subcategoria do RID "Tristeza" / word_count
emotions_affection	Nº de tokens classificados na subcategoria do RID "Afeto" / word_count
emotions_aggression	Nº de tokens classificados na subcategoria do RID "Agressão" / word_count
emotions_expressive_behavior	Nº de tokens classificados na subcategoria do RID "Comportamento Expressivo" / word_count
emotions_glory	Nº de tokens classificados na subcategoria do RID "Glória" / word_count

Variável	Descrição
primary_need	Nº de tokens classificados em qualquer subcategoria do grupo "Impulso" / word_count
primary_sensation	Nº de tokens classificados em qualquer subcategoria do grupo "Sensação" / word_count
primary_defensive_symbol	Nº de tokens classificados em qualquer subcategoria do grupo "Simbolização Defensiva" / word_count
primary_rg	Nº de tokens classificados em qualquer subcategoria do grupo "Cognição Regressiva" / word_count
primary_icarian_imagery	Nº de tokens classificados em qualquer subcategoria do grupo "Imagens Icarianas" / word_count
primary	Nº de tokens classificados em qualquer subcategoria do grupo "Primário" / word_count
secondary	Nº de tokens classificados em qualquer subcategoria do grupo "Processo Secundário" / word_count
emotions	Nº de tokens classificados em qualquer subcategoria do grupo "Emoções" / word_count

APÊNDICE E – TABELA DE VARIÁVEIS SINTÁTICAS.

Abaixo encontram-se todas as variáveis sintáticas construídas.

Tabela 29 – Descrição das variáveis sintáticas construídas para análise textual.

Variável	**Descrição**
verbs	Nº de tokens classificados como verbo/ Nº total de tokens
participles	Nº de tokens classificados como particípio/ Nº total de tokens
nouns	Nº de tokens classificados como substantivo/ Nº total de tokens
adjectives	Nº de tokens classificados como adjetivo/ Nº total de tokens
adverbs	Nº de tokens classificados como advérbio/ Nº total de tokens
denotatives_particle	Nº de tokens classificados como particípio denotativo/ Nº total de tokens
pronouns	Nº de tokens classificados como pronome/ Nº total de tokens
conjunctions	Nº de tokens classificados como conjunção/ Nº total de tokens
interjectios	Nº de tokens classificados como interjeição/ Nº total de tokens
prepositions	Nº de tokens classificados como preposição/ Nº total de tokens
foreignisms	Nº de tokens classificados como estranjeirismo/ Nº total de tokens
wh_questions	Nº de tokens classificados como uma palavra questionadora/ Nº total de tokens
special_characters	Nº de tokens classificados como caractere especial/ Nº total de tokens
past_tense_ratio	Nº de tokens classificados como verbo conjugado no pretérito especial/ Nº total de tokens
i	Nº de tokens classificados como pronome referente a primeira pessoa do singular/ Nº total de tokens
you	Nº de tokens classificados como pronome referente a segunda pessoa do singular ou plural

Variável	Descrição
we	Nº de tokens classificados como pronome referente a primeira pessoa do plural
they	Nº de tokens classificados como pronome referente a terceira pessoa do singular ou plural
i_vs_you	Nº de tokens classificados como pronome referente a primeira pessoa do singular
excentricity	Nº de tokens classificados como pronome referente a primeira pessoa do singular
noun_phrases_ratio	Nº de frases nominais/ Nº de frases
adj_and_adv_phrases_ratio	Nº de frases adverbiais ou adjetivas/ Nº de frases
prepositional_phrases_ratio	Nº de frases preposicionais/ Nº de frases
verb_phrases_ratio	Nº de frases verbais/ Nº de frases

APÊNDICE F – TABELA DE GRUPOS E VARIÁVEIS UTILIZADAS NAS SEÇÃO 6.2

Abaixo encontram-se as variáveis presentes em cada grupo utilizado pelos modelos de aprendizado de máquina na Seção 6.2.

Tabela 30 – Variáveis presentes em cada grupo utilizado pelos modelos de aprendizado de máquina na Seção 6.2.

Grupo	**Atributos**
estatístico	token_count, unique_token_ratio, unique_bigram_ratio, unique_trigram_ratio, avg_token_length, unique_tokens_per_line, average_tokens_per_line, repeat_word_ratio, line_count, unique_line_count, blank_line_count, blank_line_ratio, repeat_line_ratio, digits, exclamation_marks, question_marks, colons, semicolons, quotes, commas, dots, hyphens, stopwords_ratio, stopwords_per_line, hapax_legomenon_ratio, dis_legomenon_ratio, tris_legomenon_ratio, syllables_per_line, syllables_per_word, syllable_variation
tempo estatístico	words_per_minute, chars_per_minute, lines_per_minute
explicitude	explicit
audio	tempo, energy, liveness, speechiness, acousticness, danceability, loudness, valence, instrumentalness, duration_ms

APÊNDICE G – TABELA DE GRUPOS E VARIÁVEIS UTILIZADAS NAS SEÇÕES 6.3 E 6.4

Abaixo encontram-se as variáveis presentes em cada grupo utilizado pelos modelos de aprendizado de máquina nas Seções 6.3 e 6.4.

Tabela 31 – Variáveis presentes em cada grupo utilizado pelos modelos de aprendizado de máquina na Seção 6.3.

Grupo	**Atributos**
estatístico	token_count, unique_token_ratio, unique_bigram_ratio, unique_trigram_ratio, avg_token_length, unique_tokens_per_line, average_tokens_per_line, repeat_word_ratio, line_count, unique_line_count, blank_line_count, blank_line_ratio, repeat_line_ratio, digits, exclamation_marks, question_marks, colons, semicolons, quotes, commas, dots, hyphens, stopwords_ratio, stopwords_per_line, hapax_legomenon_ratio, dis_legomenon_ratio, tris_legomenon_ratio, syllables_per_line, syllables_per_word, syllable_variation
tempo estatístico	words_per_minute, chars_per_minute, lines_per_minute
explicitude	uncommon_words_ratio, slang_words_ratio
pronomes	i, you, it, we, they, i_vs_you, excentricity
classes gramaticais	verbs, participles, nouns, adjectives, adverbs, denotatives_particle, pronouns, conjunctions, interjections, prepositions, foreignisms, wh_questions, special_characters, lemmas_ratio
sentimento	afinn_score, compound, primary_sensation, primary_defensive_voyage, primary_defensive_symbol_random_movement, primary_regressive_cognition, primary_icarian_imagery, secondary_instrumental_behavior, secondary_restraint, secondary_temporal_references, emotions_positive_affect, emotions_anxiety, emotions_sadness, emotions_affection, emotions_aggression, emotions_expressive_behavior, emotions_glory
audio	tempo, energy, liveness, speechiness, acousticness, danceability, loudness, valence, instrumentalness, duration_ms

APÊNDICE H – MÉTRICAS DE DESEMPENHO DOS MODELOS DE LINHA DE BASE

As tabelas apresentadas nesse apêndice as métricas de desempenho dos modelos de linha de base para cada abordagem testada com a base completa.

Tabela 32 – Métricas de desempenho por gênero utilizando a abordagem "Most frequent"para a base de dados com 10 gêneros.

	Precision	**Recall**	**F1-Score**	**Support**
alternativo	0.000000	0.000000	0.000000	23
bossanova	0.000000	0.000000	0.000000	183
gospel	0.000000	0.000000	0.000000	493
indie	0.000000	0.000000	0.000000	20
mpb	0.250941	1.000000	0.401204	1000
pagode	0.000000	0.000000	0.000000	243
pop	0.000000	0.000000	0.000000	266
rock	0.000000	0.000000	0.000000	765
samba	0.000000	0.000000	0.000000	314
sertanejo	0.000000	0.000000	0.000000	678
accuracy	0.250941	0.250941	0.250941	3985
macro avg	0.025094	0.100000	**0.040120**	3985
weighted avg	0.062971	0.250941	0.100678	3985

Tabela 33 – Métricas de desempenho por gênero utilizando a abordagem "Stratified"para a base de dados com 10 gêneros.

	Precision	**Recall**	**F1-Score**	**Support**
alternativo	0.000000	0.000000	0.000000	23
bossanova	0.039409	0.043716	0.041451	183
gospel	0.113360	0.113590	0.113475	493
indie	0.000000	0.000000	0.000000	20
mpb	0.243902	0.250000	0.246914	1000
pagode	0.050209	0.049383	0.049793	243
pop	0.078571	0.082707	0.080586	266
rock	0.146438	0.145098	0.145765	765
samba	0.110345	0.101911	0.105960	314
sertanejo	0.172360	0.163717	0.167927	678
accuracy	0.151066	0.151066	0.151066	3985
macro avg	0.095460	0.095012	**0.095187**	3985
weighted avg	0.151477	0.151066	0.151221	3985

Tabela 34 – Métricas de desempenho por gênero utilizando a abordagem "Uniform"para a base de dados com 10 gêneros.

	Precision	**Recall**	**F1-Score**	**Support**
alternativo	0.009368	0.173913	0.017778	23
bossanova	0.046875	0.098361	0.063492	183
gospel	0.113801	0.095335	0.103753	493
indie	0.007614	0.150000	0.014493	20
mpb	0.263923	0.109000	0.154282	1000
pagode	0.065491	0.106996	0.081250	243
pop	0.072386	0.101504	0.084507	266
rock	0.216418	0.113725	0.149100	765
samba	0.070175	0.089172	0.078541	314
sertanejo	0.164491	0.092920	0.118756	678
accuracy	0.103388	0.103388	0.103388	3985
macro avg	0.103054	0.113093	**0.086595**	3985
weighted avg	0.166439	0.103388	0.120254	3985

Tabela 35 – Métricas de desempenho por gênero utilizando a abordagem "Most frequent"para a base de dados filtrada com 5 gêneros.

	precision	**recall**	**f1-score**	**support**
gospel	0.000000	0.000000	0.000000	493
mpb	0.415164	1.000000	0.586736	1243
pagode	0.000000	0.000000	0.000000	244
samba	0.000000	0.000000	0.000000	332
sertanejo	0.000000	0.000000	0.000000	682
accuracy	0.415164	0.415164	0.415164	0.415164
macro avg	0.083033	0.200000	**0.117347**	2994
weighted avg	0.172361	0.415164	0.243591	2994

Tabela 36 – Métricas de desempenho por gênero utilizando a abordagem "Stratified"para a base de dados filtrada com 5 gêneros.

	precision	**recall**	**f1-score**	**support**
gospel	0.161417	0.166329	0.163836	493
mpb	0.398074	0.399035	0.398554	1243
pagode	0.068493	0.061475	0.064795	244
samba	0.084592	0.084337	0.084465	332
sertanejo	0.215942	0.218475	0.217201	682
accuracy	0.257181	0.257181	0.257181	0.257181
macro avg	0.185704	0.185930	**0.185770**	2994
weighted avg	0.255997	0.257181	0.256565	2994

Tabela 37 – Métricas de desempenho por gênero utilizando a abordagem "Uniform"para a base de dados filtrada com 5 gêneros.

	precision	**recall**	**f1-score**	**support**
gospel	0.200972	0.251521	0.223423	493
mpb	0.416810	0.195495	0.266156	1243
pagode	0.079787	0.184426	0.111386	244
samba	0.116505	0.216867	0.151579	332
sertanejo	0.250000	0.224340	0.236476	682
accuracy	0.212759	0.212759	0.212759	0.212759
macro avg	0.212815	0.214530	**0.197804**	2994
weighted avg	0.282505	0.212759	0.227040	2994

APÊNDICE I – PARÂMETROS DOS MELHORES RESULTADOS APRESENTADOS PELOS MODELOS NOS EXPERIMENTOS

As tabelas apresentadas nesse apêndice apresentam os parâmetros utilizados para alcançar o melhor desempenho dos modelos em cada grupo de atributos analisado para os experimentos realizados. Cada coluna representa um tipo de modelo, e cada linha corresponde a um grupo específico de atributos.

- **n** refere-se ao número de estimadores (ou árvores) para os modelos Árvores Extremamente Aleatórias e Floresta Aleatória.
- **d** indica o tamanho das camadas densas para as Redes Neurais Artificiais.
- **C** é o parâmetro de regularização para a Máquina de Vetores de Suporte (SVM).
- **nn** representa o número de vizinhos para o algoritmo k-Vizinhos Mais Próximos (kNN).
- **p** é o parâmetro de potência para o k-Vizinhos Mais Próximos (kNN).

Tabela 38 – Parâmetros utilizados no melhor resultado de cada modelo para cada grupo de atributo nos experimentos com a base de dados com 10 gêneros.

Grupo de Atributos	ET	RNA	RF	SVM	kNN
áudio	n=300	d=(64, 64)	n=300	C=0.5	nn=10; p=1
estatístico	n=300	d=(64, 64)	n=300	C=1.0	nn=10; p=1
tempo estatístico	n=100	d=(64, 64)	n=300	C=1.0	nn=10; p=1
explicitude	n=10	d=(32, 32)	n=10	C=0.1	nn=10; p=1
tf-idf	n=10	d=(32, 32)	n=10	C=5.0	nn=3; p=2
lda	n=10	d=(32, 32)	n=300	C=0.1	nn=3; p=2
combinado	n=100	d=(64, 64)	n=300	C=0.1	nn=3; p=2
combinado + áudio	n=100	d=(64, 64)	n=300	C=0.1	nn=3; p=2
combinado (sem tf-idf+lda)	n=300	d=(64, 64)	n=300	C=5.0	nn=10; p=1
combinado (sem tf-idf+lda) + áudio	n=100	d=(64, 64)	n=300	C=2.0	nn=10; p=1

Tabela 39 – Parâmetros utilizados no melhor resultado de cada modelo para cada grupo de atributo nos experimentos com a base de dados com 10 gêneros com adição de novos grupos textuais.

Grupo de Atributos	ET	RNA	RF	SVM	kNN
áudio	n=300	d=(64, 64)	n=300	C=0.5	nn=10; p=1
estatístico	n=300	d=(64, 64)	n=300	C=1.0	nn=10; p=1
tempo estatístico	n=100	d=(64, 64)	n=300	C=1.0	nn=10; p=1
classes gramaticais	n=300	d=(64,64)	n=300	C=0.1	nn=5; p=1
explicitude	n=300	d=(64,64)	n=100	C=0.1	nn=10; p=1
pronomes	n=300	d=(64,64)	n=300	C=0.1	nn=10; p=2
sentimento	n=300	d=(64,64)	n=300	C=2.0	nn=10; p=1
combinado	n=300	d=(64,64)	n=300	C=0.5	nn=10; p=1
combinado + áudio	n=100	d=(64,64)	n=100	C=1.0	nn=10; p=1

Tabela 40 – Parâmetros utilizados no melhor resultado de cada modelo para cada grupo de atributo nos experimentos com a base de dados com os 5 gêneros brasileiros mais populados com adição de novos grupos textuais.

Grupo de Atributos	ET	RNA	RF	SVM	kNN
áudio	n=300	d=(64, 64)	n=300	C=5.0	nn=10; p=2
estatístico	n=300	d=(64, 64)	n=100	C=5.0	nn=10; p=1
tempo estatístico	n=300	d=(64, 64)	n=10	C=0.5	nn=10; p=2
classes gramaticais	n=10	d=(64, 64)	n=10	C=0.1	nn=5; p=1
explicitude	n=100	d=(64, 64)	n=100	C=0.1	nn=10; p=1
sentimento	n=300	d=(64, 64)	n=300	C=0.5	nn=4; p=1
pronomes	n=100	d=(64, 64)	n=300	C=0.5	nn=10; p=1
combinado	n=100	d=(64, 64)	n=300	C=1.0	nn=10; p=1
combinado + áudio	n=100	d=(64, 64)	n=300	C=0.5	nn=10; p=1