Instituto de MATEMÁTICA E ESTATÍSTICA UFRGS

UFRGS UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Trabalho de Conclusão de Curso

Vetorização de Textos com *Bag-of-Words* e Aprendizado de Máquina para Cadastro de NCMs - Nomenclatura Comum do MERCOSUL

Francisco Barbosa Pires

15 de fevereiro de 2024

Francisco Barbosa Pires

Vetorização de Textos com *Bag-of-Words* e Aprendizado de Máquina para Cadastro de NCMs - Nomenclatura Comum do MERCOSUL

Trabalho de Conclusão apresentado à comissão de Graduação do Departamento de Estatística da Universidade Federal do Rio Grande do Sul, como parte dos requisitos para obtenção do título de Bacharel em Estatística.

Orientador: Prof. Dr. Hugo Henrique Kegler dos Santos

Porto Alegre
15 de fevereiro de 2024

Francisco Barbosa Pires

Vetorização de Textos com *Bag-of-Words* e Aprendizado de Máquina para Cadastro de NCMs - Nomenclatura Comum do MERCOSUL

Este Trabalho foi julgado adequado para obtenção dos créditos da disciplina Trabalho de Conclusão de Curso em Estatística e aprovado em sua forma final pela Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Hugo Henrique Kegler dos Santos
Universidade Federal do Rio Grande do Sul, Porto Alegre, RS

Banca Examinadora:

Prof. Dr. João Henrique Ferreira Flores
Universidade Federal do Rio Grande do Sul, Porto Alegre, RS

Profa. Dra. Márcia Helena Barbian
Universidade Federal do Rio Grande do Sul, Porto Alegre, RS

Porto Alegre
15 de fevereiro de 2024

"Eu disse que acreditassem! Eu pedi que acreditassem! Eu nunca deixei de acreditar..."

Armindo Antônio Ranzolin (1983)

Agradecimentos

Agradeço à Universidade Federal do Rio Grande do Sul e ao Instituto de Matemática e Estatística pela jornada acadêmica. Minha profunda gratidão pela excelência do ensino e pelas portas que me abriram.

Agradeço aos meus pais, Sérgio e Nair, pelo amor que me deram e pelo apoio incondicional as escolhas que fiz na minha vida, mesmo muitas vezes sem entender ao certo meus pensamentos e anseios.

Agradeço aos meus irmãos de sangue: Charline, Cibele e Gabriel por serem os melhores irmãos que eu poderia querer e sempre estarem comigo para me aconselhar nesses primeiros anos da vida adulta.

Agradeço aos meus irmãos de coração: Fernando Dadalt e Gabriel "Debem", e aqueles com os quais a UFRGS me presenteou: Guilherme Elias Doering e Lorenzo Copetti. Sem vocês os dias que passei em Porto Alegre não teriam sido tão bons quanto foram.

Agradeço aos amigos Vinícius Zaltron e Rafael Chaves pelo convívio e o companheirismo em sala de aula que culminou em duas grandes amizades além dos portões da universidade.

Agradeço ao professor Hugo pela orientação e amizade ao longo da graduação, principalmente durante a produção deste trabalho.

Agradeço ao *Ragnarök Online*, a Rádio Eldorado FM e a Rádio da Universidade por terem fornecido a trilha sonora que embalou as madrugadas em que escrevi este trabalho.

Agradeço a todos os parentes, amigos, colegas e professores que contribuíram de alguma forma para que eu chegasse até aqui. Estejam próximos, distantes ou ausentes, guardarei sempre boas recordações de vocês.

Resumo

O cadastro da Nomenclatura Comum do MERCOSUL (NCM) de uma mercadoria é uma atividade executada por analistas em empresas com o objetivo de atribuir a classificação mais adequada a um material produzido. Cadastros feitos de forma errada podem resultar em problemas para as empresas tais como multas e recolhimento de tributos erroneamente. Este trabalho tem como objetivo estudar o método de vetorização *Bag-of-Words* e construir um algoritmo de aprendizado de máquina para auxiliar no cadastro de NCMs. A base de dados utilizada nesta pesquisa é oriunda de uma metalúrgica situada em Porto Alegre – RS, e foi inicialmente filtrada e processada pelo algoritmo de *stemming* RSLP. Em seguida, foram testadas e avaliadas as métricas do *Bag-of-Words* (*Boolean*, *Frequency* e *tfidf*) por visualização gráfica usando o método de visualização de dados em alta dimensão t-SNE. A métrica escolhida para a construção do algoritmo de aprendizado de máquina foi a *tfidf*. A base de dados vetorizada foi submetida ao método de Mapa de Difusão para reduzir a dimensionalidade dos dados em cinquenta componentes. Os dados redimensionados foram usados para construir o modelo de aprendizado de máquina, Máquina de Vetores de Suporte (SVM). A escolha dos melhores parâmetros do modelo SVM foi feita através do método de *Grid-Search*. O modelo final apresentou uma acurácia de 88,06%. Os resultados indicaram a adequação da metodologia utilizada para a construção desta pesquisa, entretanto, sugere-se alterações no método de redução de dimensão dos dados vetorizados e testagem de outras técnicas de modelagem para a construção do modelo de classificação final. Esta pesquisa foi construída na linguagem de programação Python e seu código pode ser acessado através da plataforma GitHub.

Palavras-Chave: Nomenclatura Comum do MERCOSUL, NCM, RSLP, Bag-of-Words, t-SNE, Mapa de Difusão, aprendizado de máquina, Máquina de Vetores de Suporte, Grid-Search.

Abstract

The Mercosur Common Nomenclature (NCM) register of a product is a process developed by analysts in companies, aiming to attribute the most appropriate classification to a produced material. Incorrect registration may result in problems for companies such as fines and incorrect tax payments. This work aims to study the Bag-of-Words vectorization method and build a machine learning algorithm to assist in registering NCMs. The database used in this research, which was initially filtered and processed by the RSLP stemming algorithm, comes from a metallurgy in Porto Alegre – RS. After that, the Bag-of-Words metrics (Boolean, Frequency, and tfidf) were tested and evaluated, using the t-SNE graphical visualization for high-dimensional data. The metric chosen to build the machine learning algorithm was tfidf. The vectorized database was subjected to the Difusion Map method to reduce the dimensionality of the data into fifty components. The resized data were used to build the machine learning model, Support Vector Machine (SVM). The best parameters of the SVM model were chosen through the Grid-Search method. The final model showed an accuracy of 88,06%. The results indicated the suitability of the used methodology in order to carry out this research, however, it is advisable some changes to the dimension reduction method for vectorized data and testing other modeling techniques for building the final classification model. This research was done using the Python language and its code can be accessed through the GitHub platform.

Keywords: Mercosur Common Nomenclature, NCM, RSLP, Bag-of-Words, t-SNE, Diffusion Map, machine learning, Support Vector Machine, Grid-Search.

Sumário

1 Introdução **11**
1.1 Contexto e Problematização . . . 11
1.2 Trabalhos Relacionados . . . 13
1.3 Questões de Pesquisa . . . 13
1.4 Objetivo Principal . . . 13
1.5 Objetivos Secundários . . . 13
1.6 Metodologia . . . 14

2 Referencial Teórico **17**
2.1 *Stemmers* . . . 17
2.2 *Bag-of-Words* . . . 18
2.3 t-SNE . . . 20
2.4 Diffusion Map . . . 22
2.5 Aprendizado de Máquina . . . 23
2.5.1 Support Vector Machine . . . 24
2.5.2 *Grid-Search* . . . 29

3 Resultados **30**
3.1 Fonte de dados . . . 30
3.2 Pré-Processamento dos Dados . . . 31
3.2.1 Filtragem . . . 31
3.2.2 *Stemming* dos Dados . . . 32
3.2.3 Divisão dos Dados . . . 33
3.3 Vetorização via *Bag-of-Words* . . . 33
3.4 Aprendizado de Máquina na Base Vetorizada . . . 35
3.4.1 Redução de Dimensionalidade . . . 35
3.4.2 Treinamento dos Modelos . . . 36
3.4.3 Testagem dos Modelos . . . 36

4 Conclusão **38**

Referências Bibliográficas **38**

Lista de Figuras

Figura 2.1: Fluxograma do Algoritmo RSLP 18
Figura 2.2: Resultado do *Diffusion Map* em Dados com Estrutura de "Rocambole" 22
Figura 2.3: Representação da Estrutura do Algoritmo *Support Vector Machine* 25
Figura 2.4: Gráfico de Dispersão de Dados Não Linearmente Separáveis . . . 27
Figura 2.5: Representação do Funcionamento do Algoritmo *Grid-Search* . . . 29

Figura 3.1: Aplicação do Algoritmo `stem.rslp` 32
Figura 3.2: Visualização Gráfica do *Bag-of-Words* - Função *Boolean* 34
Figura 3.3: Visualização Gráfica do *Bag-of-Words* - Função *tf* 34
Figura 3.4: Visualização Gráfica do *Bag-of-Words* - Função *tfidf* e Complexidade 50 34
Figura 3.5: Visualização Gráfica do *Bag-of-Words* - Função *tfidf* e Complexidade 70 35
Figura 3.6: Visualização Gráfica do *Bag-of-Words* na Base Teste - Função *tfidf* 36
Figura 3.7: Matriz de Confusão do Modelo SVM Construído 37

Lista de Tabelas

Tabela 1.1: Quadro de Descrição do Código NCM 8409.91.12 11

Tabela 2.1: Representação de Documentos via BoW 18
Tabela 2.2: Representação de Documentos via BoW - Função: *Boolean* 20
Tabela 2.3: Representação de Documentos via BoW - Função: *tf* 20
Tabela 2.4: Representação de Documentos via BoW - Função: *tfidf* 20
Tabela 2.5: Matriz de Confusão para Modelos de Classificação 23
Tabela 2.6: Valores do *Grid-Search* 29

Tabela 3.1: Modificação dos Nomes das Variáveis 30
Tabela 3.2: Tabela de Frequências da Distribuição de NCMs pelos Dados . . . 31
Tabela 3.3: Total de Observações de NCMs com 500 ou mais Casos 32
Tabela 3.4: Termos mais Frequentes em cada NCM 33
Tabela 3.5: Resultados do Método *Grid-Search* 36
Tabela 3.6: Predições do Modelo SVM Construído 37

1 Introdução

1.1 Contexto e Problematização

Em 1989, o Brasil adotou Sistema Harmonizado de Designação e de Codificação de Mercadorias (SH) desenvolvido pela Organização Mundial das Alfândegas (OMA) (Dias, 2008) para classificar as mercadorias que circulavam na economia do país. O SH tem como objetivo ser um sistema único de uso global para designação e decodificação de mercadorias, que facilita a elaboração de políticas tributárias, direitos aduaneiros e de frete entre os países aderentes (Siscomex, 2022).

A codificação do sistema SH usa de um código numérico de seis dígitos, sendo cada um desses responsável por um atributo da mercadoria baseado na tabela de referência do SH (Ministério da Fazenda, 2019): os dois primeiros dígitos do código indicam o capítulo em que a mercadoria se encontra na tabela, o terceiro e o quarto dígito mostram a posição da mercadoria dentro do capítulo de referência; e os últimos dois algarismos indicam a subposição da mercadoria a fim de melhor categorizá-la (Ministério do Desenvolvimento, Indústria, Comércio e Serviços, 2023).

Com a formação do MERCOSUL em 1995, o bloco econômico desenvolveu seu próprio sistema de classificação de mercadorias, a Nomenclatura Comum do MERCOSUL (NCM), com os mesmos objetivos do SH, mas com o intuito de facilitar os acordos comerciais entre os países do bloco (Dias, 2008). A NCM é baseada no sistema SH e utiliza de sua estrutura de classificação através de um código numérico, porém, foram adicionados dois dígitos a mais ao fim da codificação para ampliar a distinção das mercadorias, chamados de "item"(7 º dígito) e "subitem"(8 º dígito) (Dias, 2008; Ministério da Fazenda, 2019). Na Tabela 1.1, tem-se como exemplo a classificação de um cabeçote de motor de carro.

Tabela 1.1: Quadro de Descrição do Código NCM 8409.91.12

Dígitos	**Descrição**
84	Reatores nucleares, caldeiras, máquinas, aparelhos e instrumentos mecânicos, e suas partes.
8409	Partes reconhecíveis como exclusiva ou principalmente destinadas aos motores das posições 84.07 ou 84.08.
8409.91	Outras - reconhecíveis como exclusiva ou principalmente destinadas aos motores de pistão, de ignição por centelha (faísca).
8409.91.1	Bielas, blocos de cilindros, cabeçotes, cárteres, carburadores, válvulas de admissão ou de escape, coletores de admissão ou de escape, anéis de segmento e guias de válvulas.
8409.91.12	Blocos de cilindros, cabeçotes e cárteres.

Fonte: MERCOSUL (2023).

Seguindo as normas vigentes no país, é necessário que as empresas residentes no Brasil realizem o cadastro da NCM em suas mercadorias. Essa atividade é um processo administrativo que vincula o código NCM mais apropriado a uma mercadoria dadas as suas características e funções. A definição do código é fundamental para determinar os tributos envolvidos nas operações de comércio exterior e de saída de produtos industrializados (Ministério da Fazenda, 2019). Dessa forma, quando a NCM é cadastrada erroneamente, pode haver problemas para a empresa, como multas ou recolhimento de tributos de forma incorreta (Brasil, 2001), criando situações em que há diferenças entre a adequada definição e a aplicação normativa consequente (Goulart, 2021). Dentro do processo de cadastro, o analista responsável pode encontrar dificuldades e cometer erros ao definir a NCM mais adequada a uma mercadoria, já que é necessário que as informações disponíveis sobre o produto como características intrínsecas e complexidade de sua constituição sejam convertidas, baseando-se em sua própria interpretação, para uma classificação jurídica objetiva (Araujo, 2013).

Os desafios envolvidos nesse processo revelam a necessidade de criação de uma ferramenta para auxiliar o cadastro de NCM, a fim de evitar erros e apoiar o analista na tomada de decisão quanto ao código mais adequado. Esta ferramenta deve ser capaz de analisar os atributos de uma mercadoria, estejam eles salvos em variáveis qualitativas, quantitativas ou em campos texto no sistema da empresa, e sugerir um código NCM para classifica-la.

Nesse contexto, uma solução é aplicação dos conceitos de *data-driven*, que propõem a utilização de dados para a criação de instrumentos para gestão empresarial e melhoria de processo (Anderson, 2015), junto às ferramentas relacionadas ao Processamento de Linguagem Natural (NLP), presentes em redes sociais e sites de pesquisa para análise de texto (Tableau, 2019). Entretanto, deve-se ter ciência de que ao analisar variáveis texto produzidas dentro de uma empresa, pode-se encontrar obstáculos, entre eles o nível de qualidade dos dados, visto que campos de descrição de materiais podem apresentar textos sem normas de padronização pré-estabelecidas; e o baixo poder computacional que impede a aplicação de análises mais sofisticadas nos dados disponíveis. Logo, faz-se necessária uma técnica para converter esse campo para a linguagem computacional, permitindo seu processamento com o mínimo de perda de informação e sem comprometer a capacidade de processamento.

Diante destes obstáculos, o emprego do método de vetorização *Bag-of-Words* (BoW) pode ser a solução para esse problema, abrindo a possibilidade de se analisar computacionalmente dados do tipo texto de empresas em geral. Como esta técnica não leva em consideração o ordenamento das palavras no texto e nem suas propriedades gramaticais (McTear et al., 2016), são contornados os problemas que podem surgir ao analisar dados de campos texto que carecem de metologias de preenchimento e normas gramaticais.

A contribuição teórica deste estudo consiste em propor a aplicação do método de vetorização BoW em contextos diferentes daqueles já utilizados (Cortes, 2019; Altenbernd, 2021). A contribuição prática, por sua vez, está relacionada à melhoria do processo de cadastro de NCM, tornando-o mais robusto contra falhas, a fim de evitar as consequências de uma empresa possuir uma mercadoria classificada erroneamente, além de favorecer outros processos que precisem de informações vindas de campos texto.

1.2 Trabalhos Relacionados

Outras pesquisas também se dedicaram à construção de algoritmos de aprendizado de máquina para auxiliar no processo de Cadastro de NCMs ou de outros códigos de classificação fiscal. Cada trabalho procurou tratar o problema com diferentes abordagens de modelagem.

O trabalho de Batista (2017) construiu um algoritmo de classificação de mercadorias em capítulos específicos da tabela NCM através do algoritmo de *Naïve Bayes*. Os conjuntos de dados utilizados por Batista (2017) foram classificados em fácil, médio e difícil e atingiram a acurácia de 98%, 90% e 83%, respectivamente.

Pinheiro e Amaris (2022) utilizaram a NCM e sua hierarquia para realizar a classificação de produtos presentes em banco de dados de Notas Ficais eletrônicas (NF-e) através de uma metodologia de cascata. Os modelos testados em sua pesquisa foram os algoritmos de classificação *Support Vector Machine* (SVM) e *Naïve Bayes*. O SVM apresentou uma acurácia superior ao *Naïve Bayes* em seus resultados.

Em Luppes (2019) utilizou-se Redes Neurais Convolucionais para realizar a classificação de descrições curtas no sistema SH. O modelo proposto foi testado em dois conjuntos de dados que obtiveram os resultados de *F1-Score* de 92% e 90%, respectivamente, para a classificação de mercadorias nos dois primeiros dígitos do código SH.

Todos os trabalhos utilizaram bases de dados diferentes, entretanto, em todas as pesquisas as descrições foram vetorizadas utilizando a métrica *tfidf* para a representação matricial dos dados. Os trabalhos de Pinheiro e Amaris (2022) e Luppes (2019) utilizam metodologia de cascata em seus algoritmos para a classificação de mercadorias, enquanto Batista (2017), assim como este trabalho, não fez uso dela. Devido aos contextos das pesquisas, em nenhum dos trabalhos citados foi utilizado um método de redução de dimensionalidade dos dados além de algoritmos de *stemming* e remoção de *stop words*.

1.3 Questões de Pesquisa

Considerando o contexto apresentado, levanta-se a seguinte questão de pesquisa: De que forma o *Bag-of-Words* pode ser utilizado para vetorização de variáveis texto empregadas no cadastro semiautomatizado de NCM?

1.4 Objetivo Principal

Assim apresentado o cenário atual e a referente questão de pesquisa, este trabalho tem como objetivo principal verificar a aplicação do método de vetorização *Bag-of-Words* para variáveis do tipo texto por meio do uso de seus dados vetorizados em um algoritmo de aprendizado de máquina que auxilie no cadastro de NCM.

1.5 Objetivos Secundários

A fim de atingir o objetivo principal deste estudo, traçam-se os seguintes objetivos específicos:

I. Analisar e limpar os dados para definir as observações que serão utilizadas;

II. Explorar as formas de aplicação do BoW;

III. Reduzir a dimensão dos dados vetorizados com o método *Diffusion Map*;

IV. Aplicar o algoritmo de aprendizado de máquina *Support Vector Machine* (SVM) nos dados gerados na etapa anterior, e

V. Avaliar a eficiência do algoritmo final.

1.6 Metodologia

Usando os critérios estabelecidos por Gerhardt e Silveira (2009) para a caracterização da pesquisa, esse estudo faz abordagem de forma quantitativa, pois a análise das variáveis envolvidas e seus resultados podem ser apresentados e mensurados pela linguagem matemática. A natureza desta pesquisa é aplicada, já que se busca gerar conhecimento para a aplicação prática da vetorização através do *Bag-of-Words* por meio do auxílio no cadastro de NCM. O objetivo da pesquisa é descritivo, visto que envolve descrever como o BoW se comporta na vetorização de campos de descrição de material. Quanto aos procedimentos, a pesquisa é uma pesquisa documental, pois analisa-se dados secundários coletados por uma metalúrgica sem qualquer tratamento analítico anterior. E, por fim, categoriza-se este trabalho como transversal, uma vez que não ocorre um acompanhamento das observações estudadas (Luna Fº, 1998).

A base de dados utilizada nesta pesquisa foi fornecida por uma empresa metalúrgica sediada na cidade de Porto Alegre (RS). As observações são constituídas pelos materiais cadastrados pela própria empresa e possuem como campos a descrição do material em forma de texto e outras características de cada item como peso, área, tipo de depósito para armazenagem e o seu respectivo código NCM.

Quanto às técnicas usadas na pesquisa, para pré-processar as variáveis texto foi aplicado um algoritmo *stemmer* e o método de vetorização *Bag-of-Words*. O *stemmer* resume palavras relacionadas a uma forma mínima comum e, em seguida, agrupa-as em uma única representação, o *stem* (Coelho, 2007), enquanto a vetorização via BoW, segundo McTear et al. (2016), tem como ideia verificar a frequência com que uma palavra aparece em um texto, desconsiderando a ordem em que ela aparece, representando-a através de um valor numérico (Matsubara et al., 2003).

Como critério para a escolha da melhor configuração de BoW empregou-se o modelo *t-Distributed Stochastic Neighbor Embedding*, t-SNE (Van der Maaten e Hinton, 2008), uma técnica de redução de dimensionalidade apropriada para visualização de dados com 4 ou mais dimensões, que reduz a dimensão de dados de $\mathbb{R}^n$ para $\mathbb{R}^s$, sendo $n > s$, através da minimização das diferenças entre as distribuições que medem a similaridade dos dados em $\mathbb{R}^n$ e $\mathbb{R}^s$.

Para reduzir a dimensão da matriz BoW, foi adotado o algoritmo *Diffusion Map*, um método de redução de dimensionalidade não linear e de baixo custo computacional (Porte et al., 2008). Uma vez reduzido o número de componentes do modelo BoW, as etapas seguintes requiriram menos capacidade computacional de processamento em comparação a vetorização BoW sem dimensionalidade reduzida.

Na etapa de elaboração do algoritmo de aprendizado de máquina para cadastro de NCM, foi empregado o *Support Vector Machine* (SVM), uma técnica de aprendizado supervisionado de reconhecimento de padrões, utilizado para classificação e análise de regressão. O SVM cria, através de conjunto de dados treino, um hiperplano que forma a margem entre os grupos de interesse, neste caso códigos de NCM diferentes, e usa deste hiperplano para classificar observações novas (Denig, 2015).

Esta pesquisa foi desenvolvida a partir da linguagem de programação Python (Van Rossum e Drake, 2009), usando os pacotes *pandas* (McKinney, 2010), *numpy* (Harris et al., 2020), *random* (Van Rossum, 2020), *NLTK* (Bird et al., 2009), *scikit-learn* (Pedregosa et al., 2011), *seaborn* (Waskom, 2021), *pyDiffMap* (Banisch et al., 2020), e *matplotlib* (Hunter, 2007).

O método de pesquisa seguiu as seguintes etapas:

1. Pré-Processamento dos Dados:
    i. Importação dos dados: importação dos registros armazenados em um arquivo *.xlsx* para um *DataFrame* na linguagem Python;
    ii. Análise descritiva: identificação de informações sobre os dados presentes como dados faltantes e composição, como a distribuição de observações pelo código de NCM;
    iii. Definição de amostras: seleção de observações que serão usadas nas próximas etapas;
    iv. *Stemming*: resumo de palavras relacionadas a um radical usando um algoritmo *stemmer*;
    v. Divisão dos dados: separação das observações em conjunto a ser utilizado em treino e outro em teste.
2. Vetorização:
    i. Teste de parâmetros: construção de diferentes matrizes de vetorização BoW com os dados selecionados, alterando parâmetros de definição de termo e valor númerico atribuído ao mesmo;
    ii. Escolha de vetorização: emprego do resultado gráfico do modelo t-SNE para definição da matriz BoW mais adequada aos dados.
3. Aprendizado de Máquina:
    i. Redução de dimensão: uso do método de redução de dimensionalidade *Diffusion Map* para obtenção de um conjunto de dados com menor tamanho de dimensão;
    ii. Ajuste de algoritmo: seleção dos melhores parâmetros para o algoritmo SVM, dado o conjunto de dados treino;
    iii. Treino do algoritmo: treino do algoritmo SVM usando os argumentos apontados no subetapa anterior e a base de treino.
4. Obtenção de resultados:
    i. Testagem: aplicação do algoritmo treinado no conjunto de dados teste;

ii. Análise de resultados: verificação dos valores do algoritmo treinado;

iii. Apresentação dos resultados: conclusão sobre a qualidade do algoritmo com base nos valores apresentados.

2 Referencial Teórico

O conjunto de técnicas empregadas nesta pesquisa é apresentado nos tópicos abaixo. Cada seção busca abordar a teoria e características de um tópico de forma que esclareça a motivação de seu uso. Inicialmente, são apresentadas as técnicas utilizadas de pré-processamento e visualização gráfica: *Stemmers*, *Bag-of-Words* e t-SNE. Em seguida, explica-se os métodos e conceitos voltados para a construção do modelo de aprendizado de máquina final: redução de dimensionalidade (*Diffusion Map*), modelagem dos dados (*Support Vector Machine*) e escolha de hiperparâmetros (*Grid-Search*).

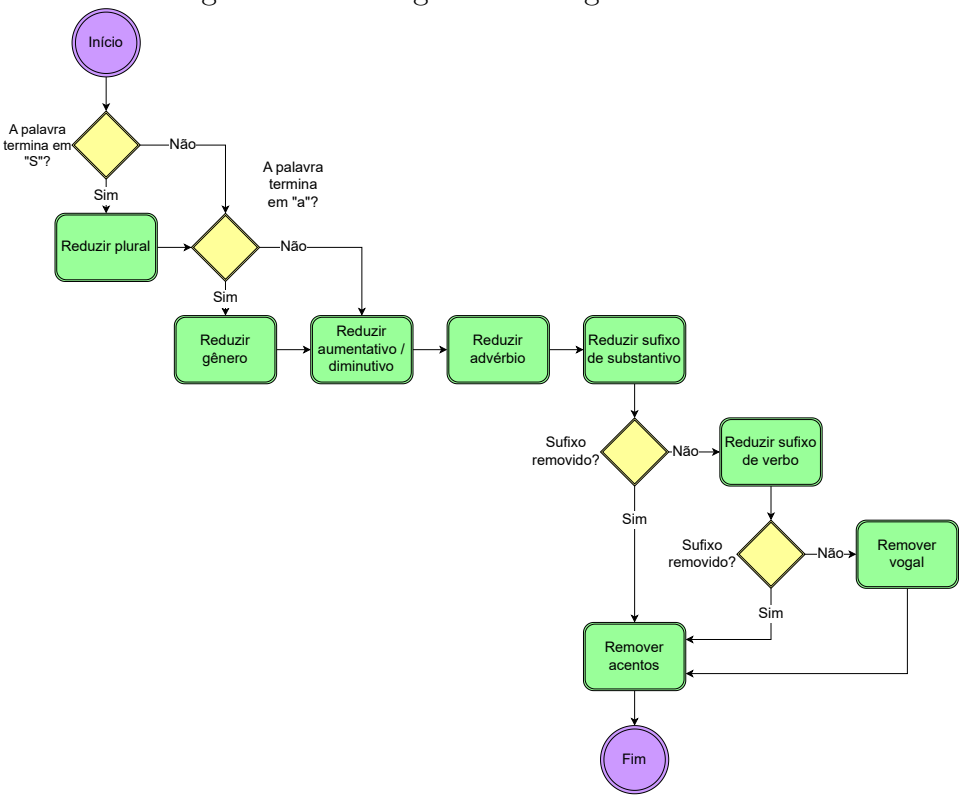
2.1 *Stemmers*

Com o objetivo de reduzir o custo computacional de processamento dos dados, é interessante aplicar métodos no pré-processamento que resumam os dados e eliminem redundâncias ao mesmo tempo que minimizem a perda de informação dos dados. O banco de dados aqui utilizado pode comportar até mil caracteres na variável descrição e seu preenchimento do campo pode apresentar um termo com diferentes desinências, sufixos que indicam a flexão das palavras em: número, gênero, pessoa, tempo, entre outros (Houaiss e Villar, 2001). Desta forma, o emprego de um algoritmo *Stemmer* consegue reduzir o número de diferentes termos similares na descrição dos materiais resumindo-os a um único termo, o que evita redundâncias e reduz o tamanho dos dados.

O termo *Stemmers* refere-se ao conjunto de algoritmos que reduzem variantes de uma mesma palavra a uma forma comum de representação (Orengo e Huyck, 2001), chamada *stem* ou 'radical' (Coelho, 2007). O *stemming* de palavras é feito com o uso do método de conflação, que é a fusão ou combinação das formas morfologicamente variantes de um termo (Frakes e Baeza-Yates, 1992). Por exemplo, as palavras "protetora", "protetor", "protetores" e "protetivo", presentes na base de dados, representam em termos gerais o significado do verbo "proteger" (Alvares et al., 2005), e podem ser reduzidas ao *stem*: *prote*.

Na linguagem de programação Python, a biblioteca NLTK (Bird et al., 2009) fornece funções para o uso de *Stemmers* em textos de língua portuguesa e inglesa. O método utilizado para reduzir termos em português usado na biblioteca encontra-se no comando `stem.rslp` e é baseado no método apresentado por Orengo e Huyck (2001), chamado Removedor de Sufixos da Língua Portuguesa - RSLP. A sequência de passos do algoritmo RSLP pode ser vista na Figura 2.1.

Figura 2.1: Fluxograma do Algoritmo RSLP



Fonte: Orengo e Huyck (2001).

2.2 *Bag-of-Words*

A fim de realizar o processamento dos dados texto presentes no banco de dados deste trabalho, se faz necessária a aplicação de uma técnica que represente suas palavras em uma forma capaz de ser compreendida pelo algoritmo de aprendizado de máquina. O método *Bag-of-Words* (BoW) é uma forma de representar um documento (texto) por meio de um vetor numérico onde cada entrada é uma função dos termos (palavras, frases ou sentenças) que o compõem, desconsiderando a ordem em que estes aparecem ao longo do texto e outras regras gramaticais (McTear et al., 2016).

Considere a lista $D = [d_1, d_2, \ldots, d_N]$, sendo a lista de todos os N documentos que compõem o conjunto de dados e $T = [t_1, t_2, \ldots, t_M]$, a lista de todos os M termos que compõem todos os documentos do conjunto. Assim, pode-se representar um conjunto de textos vetorizados via BoW na Tabela 2.1 (Martins, 2003):

Tabela 2.1: Representação de Documentos via BoW

	t_1	t_2	$\ldots$	t_M
d_1	a_{11}	a_{12}	$\ldots$	a_{1M}
d_2	a_{21}	a_{22}	$\ldots$	a_{2M}
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
d_N	a_{N1}	a_{N2}	$\ldots$	a_{NM}

Desta forma, cada documento d_i é representado pelo vetor $v_i = [a_{i1}, a_{i2}, \ldots, a_{iM}]$, sendo a_{ij} o valor numérico associado ao j-ésimo termo no documento i. O valor atribuído a cada componente a_{ij} pode ser definido por diferentes funções (Matsubara et al., 2003), dentre elas, destacam-se as seguintes:

- ***Boolean ou Binária***: as componentes a_{ij} do vetor v_i recebem seus valores conforme a função indicadora do j-ésimo termo. Ou seja a_{ij} é igual a 1 se o j-ésimo termo estiver presente no documento d_i, e 0 caso contrário (Rossi, 2011);

- ***Term of Frequency (tf)***: a_{ij} é calculada como sendo a frequência que o j-ésimo termo aparece no documento d_i, define-se esta função por:

$$tf(t_j, d_i) = freq(t_j, d_i); \tag{2.1}$$

- ***Term Frequency–Inverse Document Frequency (tfidf)***: pondera o valor de a_{ij} dado pela função *tf* considerando a frequência do j-ésimo termo em todos os documentos do conjunto, dando maior importância para termos menos frequentes no conjunto de dados. Assim, os termos que aparecem na maioria dos documentos tem uma influência menor na representatividade de um documento, enquanto termos menos frequentes tem uma maior importância (Matsubara et al., 2003). O cálculo da função *tfidf* é indicado por:

$$tfidf(t_j, d_i) = freq(t_j, d_i) \log\left(\frac{N}{a(t_j)}\right), \tag{2.2}$$

  sendo $freq(t_j, d_i)$ a frequência do j-ésimo termo no documento d_i, N o total de documentos do conjunto D e $a(t_j)$ o número de documentos onde o termo j aparece. O fator de ponderação $\log\left(\frac{N}{a(t_j)}\right)$ da função *tf* está contido no intervalo $[0, \log(N)]$ (Matsubara et al., 2003).

Em exemplo, seja $D = [d_1, d_2, d_3, d_4]$ um conjunto de 4 documentos composto por:

d_1 = "Ele gosta de pizza de brócolis e ela gosta de pizza de milho.",

d_2 = "Pizza, brócolis e milho são comidas",

d_3 = "Ele fez salada de brócolis e salada de milho, ela fez salada de palmito com brócolis." e

d_4 = “Ela gosta de brócolis no vapor e brócolis com milho.”.

Considerando como termo: palavras com 3 ou mais caracteres. Então a vetorizaçãode D usando o método BoW e as funções *Boolean*, *tf* e *tfidf* é dado pelas Tabelas 2.2, 2.3 e 2.4, respectivamente:

Tabela 2.2: Representação de Documentos via BoW - Função: *Boolean*

	brócolis	**com**	**comidas**	**ela**	**ele**	**fez**	**gosta**	**milho**	**palmito**	**pizza**	**salada**	**são**	**vapor**
d_1	1	0	0	1	1	0	1	1	0	1	0	0	0
d_2	1	0	1	0	0	0	0	1	0	1	0	1	0
d_3	1	1	0	0	1	1	0	1	1	0	1	0	0
d_4	1	1	0	1	0	0	1	1	0	0	0	0	1

Tabela 2.3: Representação de Documentos via BoW - Função: *tf*

	brócolis	**com**	**comidas**	**ela**	**ele**	**fez**	**gosta**	**milho**	**palmito**	**pizza**	**salada**	**são**	**vapor**
d_1	1	0	0	1	1	0	2	1	0	2	0	0	0
d_2	1	0	1	0	0	0	0	1	0	1	0	1	0
d_3	2	1	0	0	1	2	0	1	1	0	3	0	0
d_4	2	1	0	1	0	0	1	1	0	0	0	0	1

Tabela 2.4: Representação de Documentos via BoW - Função: *tfidf*

	brócolis	**com**	**comidas**	**ela**	**ele**	**fez**	**gosta**	**milho**	**palmito**	**pizza**	**salada**	**são**	**vapor**
d_1	0,000	0,000	0,000	0,301	0,301	0,000	0,602	0,000	0,000	0,602	0,000	0,000	0,000
d_2	0,000	0,000	0,602	0,000	0,000	0,000	0,000	0,000	0,000	0,301	0,000	0,602	0,000
d_3	0,000	0,301	0,000	0,000	0,301	1,204	0,000	0,000	0,602	0,000	1,806	0,000	0,000
d_4	0,000	0,301	0,000	0,301	0,000	0,000	0,301	0,000	0,000	0,000	0,000	0,000	0,602

Na linguagem de programação Python, o método BoW pode ser encontrado na biblioteca scikit-learn (Pedregosa et al., 2011), mediante as funções `CountVectorizer` e `TfidfVectorizer`, com a possibilidade de escolher qual das métricas deseja-se aplicar, tratar os textos conforme questões de acentuação, tamanho das palavras, diferenciação ou não de letras maiúsculas e minúsculas, assim como definir qual é o conjunto de termos e como devem ser analisados. Neste trabalho, testou-se as métricas *Boolean*, *tf* e *tfidf* junto com duas definições de termos: palavra por palavra e palavra por palavra junto com palavras agrupadas dois a dois, totalizando 6 cenários diferentes.

2.3 t-SNE

O *t-Distributed stochastic neighbor embedding*, t-SNE, é uma técnica de redução de dimensionalidade não linear utilizado para visualização gráfica (Van der Maaten e Hinton, 2008) oriundo do algoritmo *Stochastic Neighbor Embedding*, SNE (Hinton e Roweis, 2002). Segundo Pio e Sodré (2019), o t-SNE baseia-se no posicionamento de coordenadas que minimizam a diferença entre duas distribuições: a que mede a semelhança entre os pares de objetos de entrada na dimensão $\mathbb{R}^n$, denotada por P, e a que mede a similaridade dos pares dos mesmos objetos projetados em um espaço de menor dimensionalidade $\mathbb{R}^s$, representada por Q, sendo $s < n$.

O t-SNE usa das distâncias euclidianas entre os dados em $\mathbb{R}^n$ para construir probabilidades condicionais entre as observações (Lopes, 2020). Desta forma, o modelo define a similaridade entre duas observações através de sua probabilidade conjunta, calculada pela simetrização das probabilidades condicionais entre dois pontos (Pio e Sodré, 2019). Portanto, sejam x_i e x_j observações no espaço de dimensão $\mathbb{R}^n$, a probabilidade condicional das observações $p_{j|i}$ é definida por:

$$p_{j|i} = \frac{\exp\left(\frac{-\|x_i, x_j\|^2}{2\sigma_i^2}\right)}{\sum\limits_{k \neq i} \exp\left(\frac{-\|x_i, x_k\|^2}{2\sigma_i^2}\right)}, \tag{2.3}$$

exceto no caso especial em que $i = j$, como o t-SNE busca definir a similaridade entre pares, define-se $p_{i|i} = 0$ (Van der Maaten e Hinton, 2008). Além disso, σ_i^2 é a variância da distribuição Normal que está centrada em x_i (Prado, 2020). Cabe destacar que não existe um valor que se adeque para σ_i em todo o mapa de dados, uma vez que em regiões com maior densidade o valor de σ_i é geralmente mais baixo, enquanto em regiões mais esparsas o valor de σ_i é mais alto (Lopes, 2020).

Isto posto, define-se a medida de similaridade entre os pontos como a soma das probabilidades condicionais, dividida pelo dobro de observações do conjunto de dados, conforme mostra a equação:

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2n}. \tag{2.4}$$

Já no espaço com s dimensões, os pontos x_i e x_j são representados por y_i e y_j, e sua similaridade é mensurada de modo análogo ao espaço dimensional $\mathbb{R}^n$. Entretanto, a probabilidade condicional entre os pontos y_i e y_j é calculada utilizando a distribuição *t-student* com um grau de liberdade (Lopes, 2020), desta forma a similaridade é definida como:

$$q_{ji} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_{ik} - y_{jl}\|)^{-1}}, \tag{2.5}$$

e assim como no espaço $\mathbb{R}^n$, tem-se por definição: $q_{i|i} = 0$ (Van der Maaten e Hinton, 2008), e consequentemente $q_{ii} = 0$.

Diante disso, a fim de definir o posicionamento das observações x_i e x_j no espaço $\mathbb{R}^s$, o t-SNE busca a minimização da divergência de Kullback-Leibler entre as distribuições P e Q, representada por (Van der Maaten e Hinton, 2008):

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}, \tag{2.6}$$

a minimização da divergência 2.6 é calculada com base em seu gradiente em relação a y_i - que por sua vez teve sua probabilidade calculada com auxílio da distribuição *t-student*. A minimização é computada por:

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}. \tag{2.7}$$

Van der Maaten e Hinton (2008) apontam que o valor de complexidade dos dados, o número de vizinhos próximos que cada observação possui, usado no algoritmo deve estar entre 5 e 50, entretanto, Wattenberg et al. (2016) sugerem a utilização de valores fora deste intervalo a fim de melhor entender o comportamento do conjunto de dados. Neste trabalho, aplica-se os valores 50 e 70 no algoritmo t-SNE, afim de comparar a vizinhança das observações em diferentes modelos de *Bag-of-Words*.

O hiperparâmetro taxa de aprendizado, *learning rate*, aplicado no algoritmo t-SNE ao longo deste trabalho é o sugerido por Belkina et al. (2019), definido por:

$$\mathrm{lr}(N, \kappa) = \max(\frac{N/\kappa}{4}, 50), \tag{2.8}$$

sendo N o número de observações e κ o hiperparâmetro *early exaggeration* que controla a compacidade entre os *clusters* naturais do espaço original $\mathbb{R}^n$ na dimensão $\mathbb{R}^s$ e quanto de espaço haverá entre eles (Van der Maaten e Hinton, 2008).

Em Python, o t-SNE pode ser encontrado na biblioteca *scikit-learn* (Pedregosa et al., 2011) no comando `TSNE`, sendo possível ajustar parâmetros de complexidade e taxa de aprendizado.

Neste trabalho o t-SNE é empregado para a visualização gráfica dos resultados dos modelos *Bag-of-Words* com o intuito de se observar o comportamento de cada modelo em relação à dispersão das diferentes NCMs presentes. Os gráficos oriundos do t-SNE serão levados em consideração para a escolha do modelo BoW mais adequado para as etapas seguintes da pesquisa, busca-se encontrar o modelo que melhor distingue o tipo de NCM graficamente.

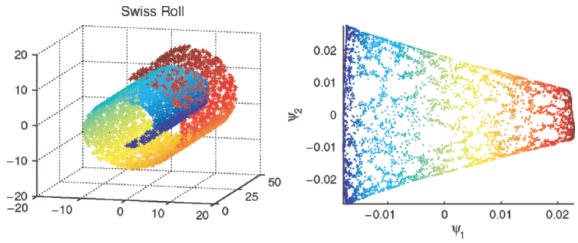
2.4 Diffusion Map

O *Diffusion Map* é um modelo desenvolvido por Coifman e Lafon (2006) para redução de dimensão de dados. Sua aplicação torna mais agradável a visualização de dados com alta dimensionalidade (Fonseca, 2019) e reduz o custo computacional de processamento dos mesmos (Porte et al., 2008). O modelo se faz vantajoso para resumir as informações dos dados processados pelo algoritmo *Bag-of-Words* a fim de obter um conjunto de dados de menor dimensão para ser aplicado ao algoritmo de aprendizado de máquina.

O algoritmo reduz a dimensão dos dados usando o cálculo do *kernel* gaussiano da matriz de dados como uma forma de mensurar a similaridade dos dados em sua dimensão inicial (Fonseca, 2019; Lopes, 2020). Com isso é possível empregar os autovalores e autovetores da matriz do *kernel* a fim de calcular as coordenadas de difusão dos dados em um espaço dimensional menor (Moreira, 2010).

Como forma de exemplificar a capacidade do *Diffusion Map* em reduzir a dimensionalidade dos dados de forma adequada, a Figura 2.2 apresenta o resultado da aplicação do modelo num conjunto de dados em forma de *swiss roll* ou "rocambole". A estrutura dos dados em "rocambole" é uma forma bidimensional em aspiral inserida em $\mathbb{R}^3$, e o uso do *Diffusion Map* neste conjunto de dados proporciona a representação de suas observações em $\mathbb{R}^2$ na forma de um quadrilátero (Nadler et al., 2008).

Figura 2.2: Resultado do *Diffusion Map* em Dados com Estrutura de "Rocambole"



Fonte: Nadler et al. (2008).

A redução de dimensionalidade fazendo o uso do *Diffusion Map* pode ser aplicada em conjuntos de dados por meio da biblioteca *DiffMap* do Python, desenvolvida por Banisch et al. (2020). Cabe destacar que até o fechamento deste estudo, a biblioteca encontrava-se ainda na versão beta 0.2.0.1. Mais detalhes sobre o *Diffusion Map* podem ser encontrados em Coifman e Lafon (2006) e Porte et al. (2008).

2.5 Aprendizado de Máquina

O Aprendizado de Máquina, ou *Machine Learning* em inglês, é uma área dentro do campo da Inteligência Artificial voltada para o estudo de algoritmos e modelos que ensinam computadores a realizar tarefas sem necessariamente programa-los (Géron, 2019; Batta, 2020). Os modelos de *Machine Learning* usam de conjuntos de dados fornecidos pelo seu desenvolvedor como fonte de informação para realizar suas funções.

Os modelos de *Machine Learning* podem ser divididos em duas categorias: modelos de aprendizado supervisionados e não supervisionados. O primeiro grupo é composto de algoritmos que assimilam atributos dos dados com demais informações previamente fornecidas pelo usuário, como por exemplo um modelo de classificação que usa um conjunto de dados com seus respectivos atributos e classes, e por meio disso tenta definir a categoria (variável resposta) de novas observações (Géron, 2019). Já os algoritmos de aprendizado não supervisionado não possuem uma resposta pré-determinada como os modelos supervisionados, estas técnicas visam a exploração dos dados e o estabelecimento de novas relações entre eles (Batta, 2020), uma das aplicações dos modelos não supervisionados é a resolução de problemas de clusterização onde busca-se agrupar dados em categorias não pré-estabelecidas usando como base sua similaridade (Géron, 2019).

Neste trabalho, utiliza-se as técnicas de aprendizado de máquina: *Bag-of-Words*, t-SNE e *Diffusion Map*, para realizar as etapas de pré-processamento e redução de dimensionalidade dos dados. Por outro lado, a construção do algoritmo final de classificação de NCMs por meio dos valores das variáveis de cada mercadoria no banco de dados treino pré-processado emprega o modelo supervisionado *Support Vector Machine* (SVM). Esta escolha deve-se ao fato de que o SVM é capaz de operar com grandes espaços de atributos de uma observação e tem eficiência na identificação de padrões em conjuntos de dados (dos Santos, 2015; Amasifuen, 2021).

Em modelos supervisionados de classificação, como o SVM, faz-se uso de indicadores para a avaliação da qualidade do algoritmo, essas métricas são calculadas com base nas classificações feitas na etapa de teste do algoritmo. Considere a matriz de confusão 2.5, em que pode-se separar as classificações feitas por um modelo em quatro grupos: verdadeiro-positivo (VP), verdadeiro-negativo (VN), falso-positivo (FP) e falso-negativo (FN). Essas atribuições definem se os dados foram classificados de forma correta ou não, baseando-se nas adequadas classificações deles (Géron, 2019).

Tabela 2.5: Matriz de Confusão para Modelos de Classificação

	Positivo	**Negativo**
Verdadeiro	VP	VN
Falso	FP	FN

Dentre as métricas mais comumente usadas para avaliar um modelo de classificação destacam-se:

- ***Acurácia ou Accuracy***: proporção de itens classificados corretamente pelo total de classificações feitas:

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

- ***Precisão***: proporção de classificações corretas positivas em relação ao total de atribuições positivas:

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

- ***Revocação ou Recall***: razão entre verdadeiro-positivos e sua soma com as atribuições falso-negativas:

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

- ***F1-Score***: relação entre os resultados de precisão e *recall* de um modelo, ou seja, relaciona todos os valores classificados positivos com todos os resultados que deveriam ser classificados como positivos:

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

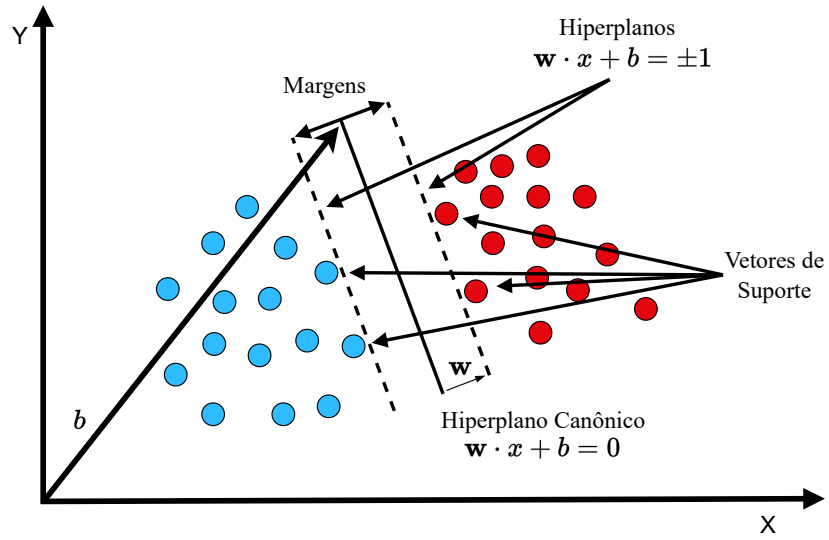
É importante destacar que a relevância de cada métrica depende do modelo computado, já que o comportamento de cada indicador pode variar de acordo com o balanceamento dos dados. Conjuntos de dados desbalanceados, isto é, quando a quantidade de observações de uma categoria supera as demais de forma significativa, requerem maior atenção em métricas diferentes das utilizadas em dados balanceados (Saito e Rehmsmeier, 2015).

2.5.1 Support Vector Machine

O *Support Vector Machine* (SVM), ou Máquina de Vetores Suporte, é um algoritmo de aprendizado de máquina supervisionado (Denig, 2015), proposto por Boser et al. (1992), que possui a versatilidade de realizar classificações lineares, não lineares, regressões e detecções de *outliers* (Géron, 2019). Em suma, o SVM constrói um hiperplano de dimensão $\mathbb{R}^{n-1}$ em espaço $\mathbb{R}^n$-dimensional e o utiliza junto das observações mais próximas desta estrutura para construir uma superfície de decisão para a classificação dos dados (Denig, 2015).

A Figura 2.3 mostra a classificação de duas categorias como dois atributos e as componentes de um SVM. Note que, neste caso em que a dimensão dos dados é igual a dois, o hiperplano trata-se de uma reta.

Figura 2.3: Representação da Estrutura do Algoritmo *Support Vector Machine*



Fonte: Kavzoglu e Colkesen (2009).

Nesta seção, apresenta-se o modelo SVM para dados linearmente separáveis, em seguida métodos usados para a aplicação do SVM em dados não lineares e por último técnicas empregadas para a classificação de dados com mais de duas categorias.

SVM Linear

Para a construção do modelo de classificação SVM para dados linearmente separáveis, considere que: sejam $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$ um conjunto T de N dados de treinamento, sendo $\mathbf{x}_i$ o vetor de características da i-ésima observação pertencente ao conjunto $\mathbf{X}$ de observações independentes e d_i a sua respectiva classificação atribuída, em que d_i tem o valor pertencente ao conjunto $\mathbf{D} = \{-1, +1\}$ (dos Santos, 2015; Amasifuen, 2021). Assumindo que as classificações do subconjunto d_i sejam linearmente separáveis, define-se um hiperplano que realiza a divisão das categorias por:

$$\mathbf{w^T x}_i + b = 0, \tag{2.13}$$

sendo $\mathbf{w^T}$ o vetor transposto de pesos ajustáveis e b o viés associado que move o hiperplano em relação à origem (dos Santos, 2015). Como isso, a divisão das categorias ocorre pela função:

$$\text{Classificação}(\mathbf{x}_i) = \begin{cases} d_i = +1, & \text{se} \quad \mathbf{w^T x}_i + b \geq 0 \\ d_i = -1, & \text{se} \quad \mathbf{w^T x}_i + b < 0. \end{cases} \tag{2.14}$$

Entretanto, a equação 2.13 implica na existência de infinitos hiperplanos equivalentes oriundos da multiplicação de $\mathbf{w}$ e b por uma mesma constante (Altenbernd, 2021). Então, a fim de usar o SVM, escolhe-se o hiperplano canônico, ou ótimo, este, por sua vez, tem a máxima distância possível com a observação mais próxima. Esta distância também é chamada de margem de separação (Amasifuen, 2021). Isto posto, define-se o hiperplano canônico por:

$$\mathbf{w}_0^{\mathbf{T}} \mathbf{x} + b_0 = 0, \tag{2.15}$$

sendo $\mathbf{w}_0$ e b_0 os valores que maximizam a margem de separação (Denig, 2015). Assim, a distância de $\mathbf{x}$ até o hiperplano canônico é fornecida pela função:

$$g(\mathbf{x}) = \mathbf{w}_0^{\mathbf{T}}\mathbf{x} + b_0. \tag{2.16}$$

Ao se expressar $\mathbf{x}$ em termos de sua projeção ortogonal ao hiperplano canônico ($\mathbf{x}_p$) e a distância algébrica desejada (r), fica mais evidente esta relação:

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}_0}{\|\mathbf{w}_0\|}; \tag{2.17}$$

como por definição $g(\mathbf{x}_p) = 0$, tem-se que:

$$g(\mathbf{x}) = \mathbf{w}_0^{\mathbf{T}}\mathbf{x} + b_0 = r\|\mathbf{w}_0\|, \tag{2.18}$$

ou de forma correspondente (Amasifuen, 2021):

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}_0\|}. \tag{2.19}$$

Com isso, a fim de encontrar os parâmetros do hiperplano canônico, define-se as restrições:

$$\mathbf{w}_0^{\mathbf{T}}\mathbf{x}_i + b \geq 0 \quad \text{se} \quad d_i = +1 \tag{2.20}$$

e

$$\mathbf{w}_0^{\mathbf{T}}\mathbf{x}_i + b < 0 \quad \text{se} \quad d_i = -1. \tag{2.21}$$

As condições de 2.20 e 2.21 serão verdadeiras sempre que as classes dos dados forem linearmente separáveis, isto é, se é válida a equação 2.15. Portanto, define-se os vetores de suporte do SVM com os pontos $(\mathbf{x}_i, d_i)$ que satisfazem qualquer uma das restrições 2.20 e 2.21 (Denig, 2015). Os vetores de suporte são os pontos de uma base de dados teste que ficam mais próximos do hiperplano canônico, por consequência os mais difíceis de classificação, e também são a base para a construção da superfície de decisão do algoritmo (Amasifuen, 2021; Denig, 2015). Com isso, considere um vetor de suporte $\mathbf{x}_s$, por definição tem-se que:

$$g(\mathbf{x}_s) = \begin{cases} +1, & \text{se} \quad d_i = +1 \\ -1, & \text{se} \quad d_i = -1, \end{cases} \tag{2.22}$$

implicando, junto ao resultado 2.19 aplicado ao vetor de suporte, $\mathbf{x}_s$, em:

$$\frac{1}{\|\mathbf{w}_0\|} \quad \text{se} \quad d_s = +1 \tag{2.23}$$

e

$$-\frac{1}{\|\mathbf{w}_0\|} \quad \text{se} \quad d_s = -1. \tag{2.24}$$

Finalmente, considerando ρ o valor que otimiza a margem de separação em dois padrões da base de treino, tem-se:

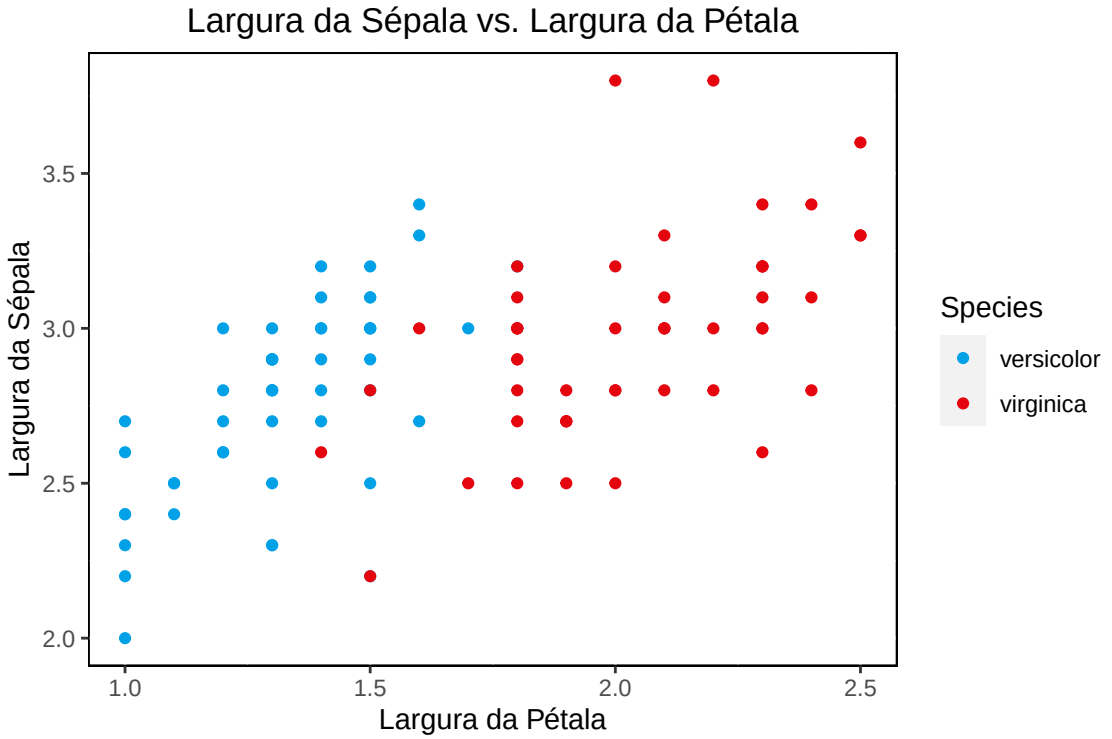
$$\rho = 2r = \frac{2}{\|\mathbf{w}_0\|}, \tag{2.25}$$

evidenciando que maximizar a margem de separação entre padrões é o mesmo que minimizar a norma euclidiana do vetor de pesos $\mathbf{w}$. Com isso, conclui-se que o vetor $\mathbf{w}_0$ do hiperplano canônico tem a máxima separação possível entre as categorias (Denig, 2015).

SVM Não Linear

Não é incomum que dados não sejam linearmente separáveis, isto é, existem pontos dentro da região de separação independentemente do lado do hiperplano, afetando a precisão do modelo SVM (Denig, 2015). Entretanto, existem métodos que permitem a adequação dos dados de forma satisfatória, podendo ser aplicáveis ao próprio algoritmo ou anteriormente como pré-processamento (Amasifuen, 2021; Géron, 2019). A Figura 2.4, exemplifica um caso bidimensional de dados não linearmente separáveis, usando o banco de dados *Iris* presente na linguagem R (R Core Team, 2022).

Figura 2.4: Gráfico de Dispersão de Dados Não Linearmente Separáveis



Uma das técnicas possíveis de aplicação é o uso do modelo SVM com margens de separação suaves. Para isso, considere novamente a amostra de treino T, $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$, e o conjunto de variáveis não negativas $\{\xi_i\}_{i=1}^{N}$, chamado de variáveis soltas. Desta forma, define-se a superfície de decisão do caso não linear por:

$$d_i \times (\mathbf{w}^{\mathbf{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, N. \tag{2.26}$$

O conjunto de variáveis soltas pode ser interpretado como o desvio de uma observação em relação à separação maximizada dos dados (Denig, 2015), os membros deste conjunto podem assumir valores entre o intervalo $(0 : 1]$ caso estejam do lado correto do hiperplano e superiores a 1 se estiverem do lado incorreto. Com base na

interpretação apresentada para as observações fora da separação, deseja-se construir um novo hiperplano que busca minimizar a média de erros de classificação calculada usando a amostra de treinamento (Amasifuen, 2021). A função,

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w}^{\mathbf{T}}\mathbf{w} + C\sum_{i=1}^{N}\xi_i, \tag{2.27}$$

ao ser minimizada, retorna o hiperplano que satisfaz a minimização da média de erros (Haykin, 2001). A função 2.27 divide-se em duas partes: a primeira parte está atribuída a maximização das distâncias entre as classificações do SVM; e a segunda está voltada à quantidade de erros de classificação (Amasifuen, 2021), sendo C um hiperparâmetro atribuído pelo usuário que define a complexidade do SVM e a quantidade de variáveis soltas. Um valor alto de C viola menos a margem de decisão, mas reduz o tamanho dela, enquanto um valor de C menor acaba por resultar no inverso: uma maior invasão da margem e um aumento no seu tamanho (Géron, 2019).

Além do método de margens suaves, é possível aplicar transformações ao conjunto de dados relacionado a sua estrutura, anteriormente à execução do algoritmo SVM a fim de criar um conjunto linearmente separável (Géron, 2019). Vale destacar que a implementação de uma transformação não inviabiliza a aplicação de método de margens suaves.

Neste trabalho, utilizam-se técnicas de pré-processamento junto ao SVM de margens suaves. A escolha da transformação aplicada aos dados é feita utilizando o método *Grid-Search*, a lista de métodos avaliadas são os *kernels* disponibilizados pela biblioteca *scikit-learn* (Pedregosa et al., 2011) da linguagem de programação Python: *linear*, *polynomial*, *gaussian* (RBF) e *sigmoid*.

SVM Multiclasse

Até agora, os conceitos aplicados basearam-se em classificar observações de forma binária. Entretanto, é comum encontrar problemas de classificação que envolvem mais de duas categorias, como no caso deste trabalho, em que deseja-se definir a NCM de uma mercadoria. Estas circunstâncias implicam na utilização de técnicas para a decisão de classificação de observações com várias categorias. Existem métodos para trabalhar com este tipo de situação usando o modelo SVM, sendo os mais comuns *one-versus-one* (ovo) e o *one-versus-rest* (ovr) (Altenbernd, 2021).

O método *one-versus-one* constrói $\binom{C}{2}$ algoritmos SVMs, sendo C o total de categorias presentes no conjunto de dados. Cada um destes modelos compara uma categoria com outra, fazendo com que todas as categorias sejam comparadas com todas as outras um a um. Ao fim, atribui-se a categoria a uma nova observação de acordo com seu resultado em cada um dos modelos (Altenbernd, 2021).

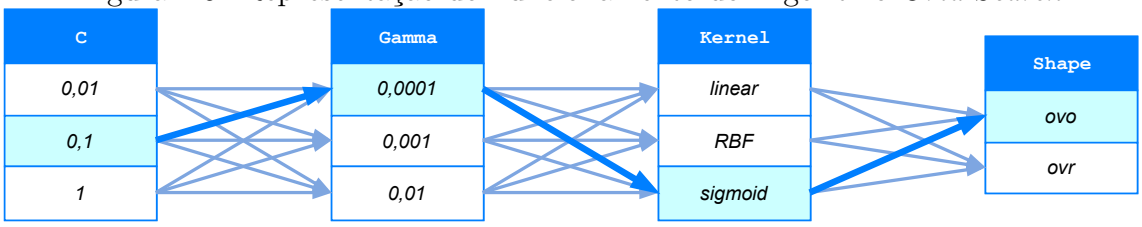
Já o método *one-versus-rest* constrói C algoritmos SVM e realiza a comparação de uma categoria com relação as demais categorias, conforme explicita seu nome. Assim, ao fim de sua execução, uma nova observação será categorizada conforme os resultados que apresentar nas C classificações que receber (Géron, 2019).

Em Python na biblioteca *scikit-learn* (Pedregosa et al., 2011), os métodos de classificação *one-versus-one* e *one-versus-rest* estão disponíveis no modelo SVM através da escolha do parâmetro `decision_function_shape`.

2.5.2 *Grid-Search*

O método de *Grid-Search* é uma técnica aplicada para se encontrar os melhores hiperparâmetros de um modelo. Em suma, o algoritmo usa uma lista de possíveis valores para os hiperparâmetros atribuída pelo usuário e realiza a comparação de todas as combinações possíveis a fim de escolher o conjunto mais adequado (Géron, 2019), conforme ilustra a Figura 2.5.

Figura 2.5: Representação do Funcionamento do Algoritmo *Grid-Search*



A biblioteca *scikit-learn* (Pedregosa et al., 2011) do Python apresenta uma modalidade de método *Grid-Search* chamada *GridSearchCV*, em que o modelo faz a escolha dos hiperparâmetros mais adequados levando em consideração os resultados da validação cruzada dos dados (Alhakeem et al., 2022). Desta forma, ao mesmo tempo que se encontra o melhor conjunto de valores é estabelecida uma proteção aos dados contra o *overfitting* dos mesmos.

Géron (2019) sugere que quando não se possui conhecimento prévio quanto à estrutura dos dados, utilize-se de valores para os hiperparâmetros iguais a 10^n, sendo $n \in \mathbb{Z}$. Entretanto, alerta-se que este método pode agravar o tempo de processamento dos dados dependendo da quantidade de parâmetros sugeridos. No contexto deste trabalho, constrói-se um modelo *Grid-Search* para a escolha dos melhores valores para o algoritmo SVM, conforme os campos da Tabela 2.6.

Tabela 2.6: Valores do *Grid-Search*

Hiperparâmetro	**Valores**
`C`	`0.01, 0.1, 1, 10, 100, 1000`
`gamma`	`0.0001, 0.001, 0.01, 0.1, 1, 10`
`kernel`	`linear, poly, rbf, sigmoid`
`decision_function_shape`	`ovo, ovr`

Os hiperparâmetros avaliados definem a complexidade do SVM e a quantidade de variáveis soltas (`C`), a transformação feita nos dados (`gamma` e `kernel`) e o modo de atuar em casos multiclasse (`decision_function_shape`).

3 Resultados

Este capítulo tem por objetivo apresentar a base de dados utilizada e os resultados deste trabalho obtidos seguindo a metodologia definida anteriormente. Com o objetivo de melhor organizar esta etapa, o conteúdo está dividido em quatro seções: apresentação da fonte de dados; pré-processamento dos dados, onde apresenta-se os resultados da filtragem, stemização e divisão dos dados; vetorização, em que são vistas as saídas da aplicação do algoritmo *Bag-of-Words* e suas representações gráficas com o t-SNE; e, aprendizado de máquina, onde se exibe os resultados da redução de dimensionalidade via *Diffusion Map* e do algoritmo SVM escolhido pelo método *Grid-Search*. O código dos resultados desta pesquisa e as especificações do computador utilizado podem ser acessados no repositório GitHub.

3.1 Fonte de dados

A base de dados utilizada nesta pesquisa foi fornecida por uma empresa metalúrgica localizada na cidade de Porto Alegre – RS. As observações são materiais cadastrados pela própria empresa no período de janeiro de 2015 até setembro de 2022. Ao todo, a base possui um total de 267.600 e 26 variáveis.

Neste trabalho, leva-se em consideração os campos *Número do item*, *Descrição* e *Código de classificação fiscal*, que contém o código ID de cada produto no sistema da empresa, a sua descrição em texto livre com até 1000 caracteres e o código NCM atribuído, respectivamente. Além destas variáveis, o banco de dados também possui informações quantitativas sobre cada material, como dimensões e peso, e qualitativas, como unidade de compra e estoque, tipo de embalagem e grupo de compradores pertencente.

Para fins de simplificação, principalmente em etapas que envolvem programação, as variáveis utilizadas da fonte de dados foram renomeadas conforme a Tabela 3.1.

Tabela 3.1: Modificação dos Nomes das Variáveis

Nome Anterior	**Nome Atual**
Número do item	ID
Descrição	DESC
Código de classificação fiscal	NCM

3.2 Pré-Processamento dos Dados

Antes de qualquer tratativa a base de dados apresentava o total de 267.600 observações e 26 variáveis. Como este estudo está delimitado a apenas três atributos de cada mercadoria da base: o código ID, a descrição em texto e sua classificação NCM, as demais variáveis foram removidas do banco de dados.

3.2.1 Filtragem

A fim de evitar repetições nos dados e variáveis faltantes que possam atrapalhar o processamento das observações, foram aplicados os seguintes filtros:

i. Remover IDs duplicados, mantendo a primeira ocorrência (112.141 observações removidas);

ii. Remover observações com qualquer dado faltante (23 observações removidas);

iii. Remover observações com código NCM e Descrição repetidas (58.156 observações removidas);

iv. Remover NCMs com valores 00000000 e 99999999 e com número de caracteres diferentes de oito (13 observações removidas).

Além dos filtros utilizados para filtrar casos, a variável Descrição também foi filtrada para a remoção de *stop words* presentes em seus textos.

Com isso, a base filtrada apresentou um total de 97.267 observações. A distribuição da classificação NCM dos dados filtrados contemplou um total de 695 classificações com suas frequências entre 1 e 54.939 mercadorias. Entretanto, 75% das classes possuíam no máximo 17 observações, conforme mostra a Tabela 3.2 de medidas. Estes resultados revelam a diversidade de mercadorias produzidas pela empresa, ao mesmo tempo que explicitam que há poucas NCMs com um número de observações grande o suficiente para se usar nas próximas etapas do trabalho.

Tabela 3.2: Tabela de Frequências da Distribuição de NCMs pelos Dados

Medidas	**Resultados**
Total	695
Média	139,95
Moda	1
1° Quartil	1
Mediana	4
3° Quartil	17
Mínimo	1
Máximo	54.939

Logo, a fim de que a base de dados tenha equilíbrio entre a frequência das classes e observações suficientes para um melhor uso dos dados nas próximas etapas, estabeleceu-se reduzir a base para estudo a amostras aleatórias sem reposição de tamanho 500 para todas as categorias com 500 ou mais observações. A Tabela 3.3

exibe os códigos NCM que atingiram o critério de possuir 500 observações ou mais. Com isso a base ficou reduzida a um total de 5.500 observações e 11 códigos de classificação NCM.

Tabela 3.3: Total de Observações de NCMs com 500 ou mais Casos

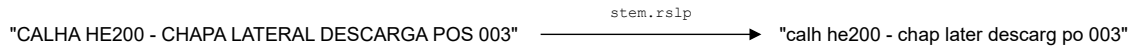
NCM	**Observações**
4008.21.00	1.941
6909.12.90	718
7208.51.00	1.083
7308.90.10	614
7308.90.90	4.438
7318.15.00	698
7325.99.10	54.939
8431.39.00	7.362
8437.90.00	7.073
8483.30.90	1.082
8537.10.90	659

Destaca-se que, conforme a Tabela 3.3, nos dados restantes existem classificações NCMs que compartilham os mesmos membros hierárquicos. O caso mais extremo ocorre entre as classes 7308.90.10 e 7308.90.90, que se diferem apenas a partir do sétimo dígito de classificação.

3.2.2 *Stemming* dos Dados

O uso do algoritmo `stem.rslp` (Orengo e Huyck, 2001; Bird et al., 2009) na base de dados filtrada diminuiu o número total de termos no texto de descrição de mercadorias de 7.035 para 6.730, reduzindo termos morfologicamente similares ao mesmo *stem*. Em exemplo, a Figura 3.1 mostra o efeito da stemização em uma observação.

Figura 3.1: Aplicação do Algoritmo `stem.rslp`



Embora esta técnica tenha contribuído em reduzir o custo computacional, o resultado do *stemming* também evidenciou uma frequência de termos acima de 50% em sete NCMs das onze filtradas, conforme mostra a Tabela 3.4.

Tabela 3.4: Termos mais Frequentes em cada NCM

NCM	Termo	%
4008.21.00	"borrach"	99.8%
6909.12.90	"ceram"	88.4%
7208.51.00	"chap"	97.8%
7308.90.10	"de"	38.8%
7308.90.90	"grad"	79.8%
7318.15.00	"paraf"	81.2%
7325.99.10	"chap"	6.8%
8431.39.00	"de"	76.4%
8437.90.00	"chap"	19.8%
8483.30.90	"rol"	98.8%
8537.10.90	"painel"	43.4%

3.2.3 Divisão dos Dados

A fim de prosseguir aos próximos passos, a base foi dividida em 70% e 30% para a construção da base de treinamento e teste respectivamente. Com isso, a base treino criada contém ao todo 5.350 palavras em sua composição, enquanto a base teste manteve um total de 3.074 termos.

3.3 Vetorização via *Bag-of-Words*

A base de dados treino foi submetida a seis modelos de vetorização *Bag-of-Words* que se diferem quanto a escolha de agrupamento de palavras que compõem a descrição dos dados e as métricas aplicadas: *Boolean*, *tf* e *tfidf*. Ao se aplicar os modelos que usam o hiperparâmetro `ngram_range=(1,2)`, isto é, o uso do conjunto de todas as palavras que compõem a descrição mais suas combinações dois a dois, o resultado foi um *dataset* com 17271 variáveis.

O produto de cada BoW é visto graficamente nas Figuras 3.2, 3.3 e 3.4 utilizando o método t-SNE com complexidade igual a 50 para visualização de dados com alta dimensionalidade. Percebe-se que o modelo t-SNE não está bem ajustado para as vetorizações que usaram as métricas *Boolean* e *tf*, visto que um número alto de observações concentra-se na região central do gráfico. Em contrapartida, os modelos BoW que usaram a métrica *tfidf* agruparam as observações por categoria de melhor forma, mantendo mercadorias com a mesma NCM mais próximas entre si ao mesmo tempo que se distanciaram das demais categorias.

Figura 3.2: Visualização Gráfica do *Bag-of-Words* - Função *Boolean*

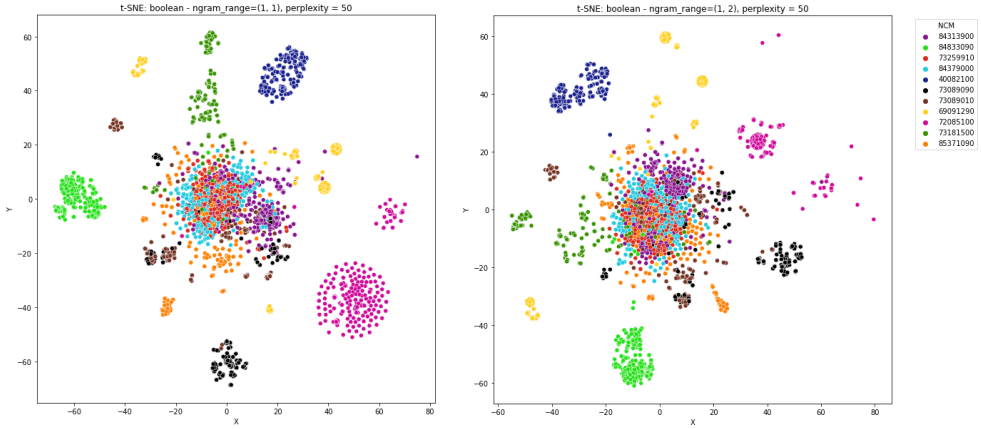


Figura 3.3: Visualização Gráfica do *Bag-of-Words* - Função *tf*

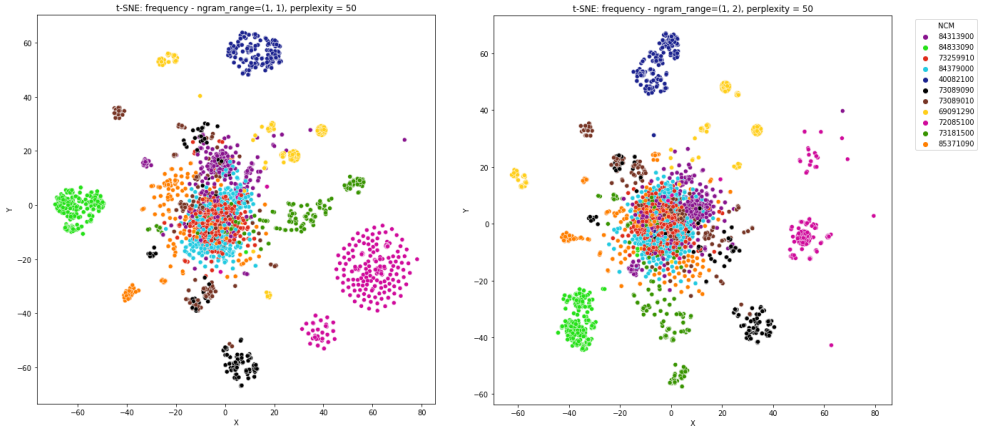
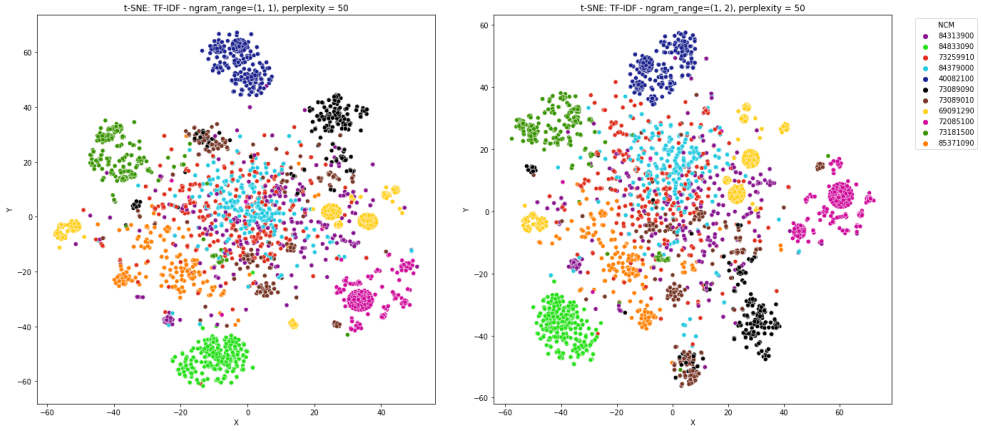
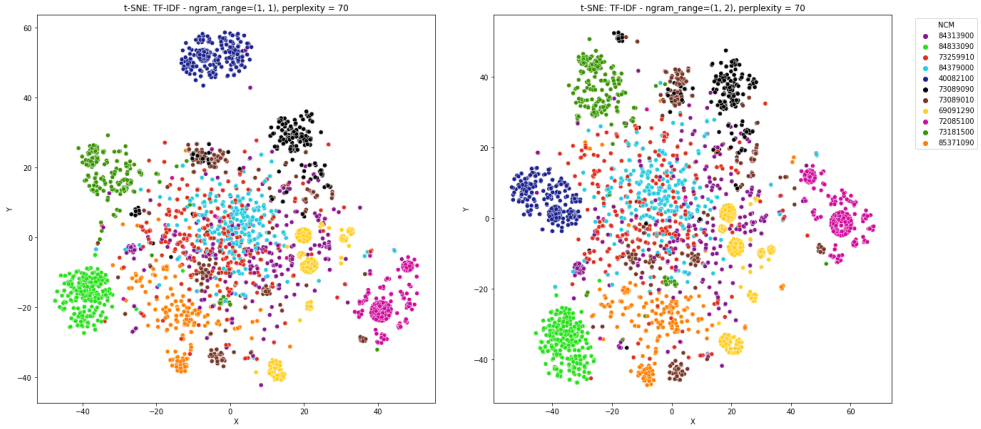


Figura 3.4: Visualização Gráfica do *Bag-of-Words* - Função *tfidf* e Complexidade 50



Com isso, optou-se por prosseguir com um modelo BoW que utilize a métrica *tfidf*, entretanto ainda não é possível selecionar o tipo ideal de agrupamento de palavras que deve-se empregar: palavra por palavra, `ngram_range=(1,1)`, ou palavra por palavra mais palavras agrupadas dois a dois `ngram_range=(1,2)`. A fim de concluir sobre o agrupamento de palavras ideal, utilizou-se do método t-SNE com o parâmetro de complexidade igual a 70, conforme mostra a Figura 3.5.

Figura 3.5: Visualização Gráfica do *Bag-of-Words* - Função *tfidf* e Complexidade 70



Percebeu-se que no modelo t-SNE com complexidade 70 as observações com NCMs 7325.99.10, 8431.39.00 e 8437.90.00 seguiram próximas, o que pode ser explicado pela despadronização no método de cadastramento de mercadorias, visto que estes códigos são os mais frequentes no banco de dados e poderiam ter recebido uma vasta variedade de textos sem padrões. Ainda assim, o modelo BoW com `ngram_range=(1,2)` conseguiu agrupar observações que compartilhavam a mesma NCM de uma melhor forma, condensando grupos de dados que já eram visíveis na Figura 3.4.

Diante do exposto, escolheu-se o modelo *Bag-of-Words* com a métrica *tfidf* e o agrupamento palavra por palavra junto com palavras agrupadas dois a dois para prosseguir para as etapas posteriores de treinamento do modelo SVM. O modelo selecionado possui uma qualidade relativa aos demais modelos em separar os dados em grupos que compartilhavam a mesma NCM.

3.4 Aprendizado de Máquina na Base Vetorizada

Para aplicar o modelo SVM na base de dados transformada escolhida anteriormente, primeiramente foi necessário aplicar a técnica *Diffusion Map* para redução de dimensionalidade. Com isso, tornou-se possível realizar o treinamento do algoritmo de aprendizado de máquina com o uso do método *Grid-Search* e por último a testagem do modelo utilizando a base de dados teste pré-processada pelas mesmas técnicas aplicadas com antecedência na base treino.

3.4.1 Redução de Dimensionalidade

Utilizando a base de treino e o método de vetorização *Bag-of-Words* escolhidos nas etapas anteriores, empregou-se o algoritmo *Diffusion Map* para redução de dimensionalidade. Com isso os dados que eram representados por 17.271 variáveis foram apresentados por 50 componentes. Desta forma, o custo computacional do processamento dos dados reduziu e tornou-se mais ágil a execução do algoritmo por computadores.

3.4.2 Treinamento dos Modelos

Fazendo uso do método *Grid-Search* para a escolha de hiperparâmetros do modelo SVM, chegou-se à combinação de valores apresentada na Tabela 3.5. O tempo de execução do algoritmo nesta etapa foi de 10 minutos.

Tabela 3.5: Resultados do Método *Grid-Search*

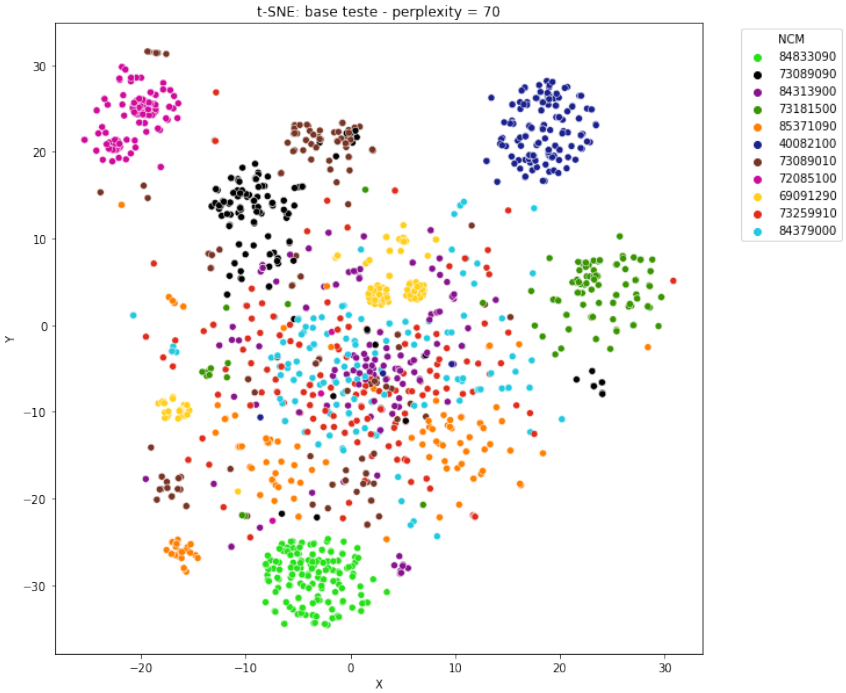
Hiperparâmetros	***Grid-Search***
`C`	`100`
`gamma`	`10`
`kernel`	`rbf`
`decision_function_shape`	`ovo`

Desta forma, prosseguiu-se para a etapa final de construção do modelo de aprendizado de máquina utilizando este conjunto.

3.4.3 Testagem dos Modelos

Antes de realizar a testagem do modelos, aplicou-se na base de dados teste os algoritmos *Bag-of-Words* e *Diffusion Map* ajustados pela base de treino para que os dados estivessem apropriados para seu processamento. A Figura 3.6 exibe a visualização da vetorização via BoW na base teste através do t-SNE com complexidade igual a 70. Destaca-se que assim como na base treino as observações das categorias estão próximas e também ocorre uma aproximação entre as observações das categorias 7325.99.10, 8431.39.00 e 8437.90.00.

Figura 3.6: Visualização Gráfica do *Bag-of-Words* na Base Teste - Função *tfidf*



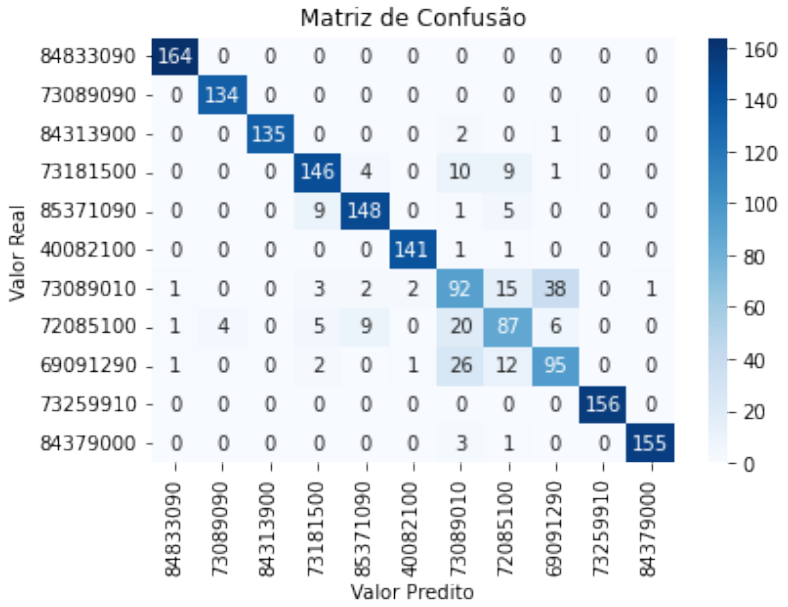
Com a realização das transformações foi possível testar o modelo SVM resultante da base de dados treino junto dos resultados do método *Grid-Search* da Tabela 3.5. A Tabela 3.6 apresenta os resultados obtidos pelo modelo final, onde atingiu-se 88.06% de acurácia.

Tabela 3.6: Predições do Modelo SVM Construído

NCM	Precision	Recall	F1-Score	Support
4008.21.00	0.9820	1.0000	0.9909	164
6909.12.90	0.9710	1.0000	0.9853	134
7208.51.00	1.0000	0.9783	0.9890	138
7308.90.10	0.8848	0.8588	0.8716	170
7308.90.90	0.9080	0.9080	0.9080	163
7318.15.00	0.9792	0.9860	0.9826	143
7325.99.10	0.5935	0.5974	0.5955	154
8431.39.00	0.6692	0.6591	0.6641	132
8437.90.00	0.6738	0.6934	0.6835	137
8483.30.90	1.0000	1.0000	1.0000	156
8537.10.90	0.9936	0.9748	0.9841	159
Accuracy			0.8806	1650
Macro Avg	0.8777	0.8778	0.8777	1650
Weighted Avg	0.8810	0.8806	0.8807	1650

A Figura 3.7 traz a matriz de confusão do modelo e indica que a maioria dos erros de classificação ocorreram com as NCMs 6909.12.90, 7208.51.00, e 7308.90.10.

Figura 3.7: Matriz de Confusão do Modelo SVM Construído



As classificações erradas de novas observações explicam-se pela composição do conjunto de dados. Termos frequentes em uma única NCM conforme mostra a Tabela 3.4 induzem ao erro de classificação de observações da mesma categoria onde esses termos se ausentam.

4 Conclusão

O modelo de aprendizado de máquina, *Support Vector Machine*, resultante da escolha de hiperpametros através do método de *Grid-Search* apresentou uma acurácia de $88,06\%$, tornando-o uma ferramenta de auxílio apropriada para o analista no cadastro de NCM de uma mercadoria. Usando o modelo construído é possível obter sugestões de NCMs para novos registros com base em classificações feitas anteriormente na empresa. Uma interface gráfica para o emprego do algoritmo pode ser criada com a biblioteca *Tkinter* (Lundh, 1999) do Python.

A base de dados aplicada a este estudo mostrou atributos em sua composição que influenciaram nas etapas da pesquisa. Visto que o conjunto possuía um total de 695 códigos NCM, mas no máximo 17 observações em 75% dos casos. Adicionalmente, os 11 códigos NCM utilizados na pesquisa exibiram características em seus itens como a aparição recorrente de um termo na maioria das vezes.

O método de stemização RSLP (Orengo e Huyck, 2001) se mostrou adequado para o cenário deste estudo em virtude de ter conseguido reduzir o número de termos do conjunto de dados de 7035 para 6730, diminuindo o tempo de processamento, evitando a existência de características escritas no campo texto armazenadas em dois ou mais termos.

Os resultados obtidos pela vetorização via *Bag-of-Words* mostraram-se influenciados pelas características do banco de dados. A frequência recorrente de termos iguais em observações que compartilham a mesma NCM resultou no método de visualização t-SNE ter indicado a métrica *tfidf* como a mais adequada para este conjunto de observações, já que o objetivo da métrica é destacar a frequência de termos menos observados em relação ao todo do conjunto.

O algoritmo de redução de dimensionalidade *Diffusion Map* (Coifman e Lafon, 2006) conseguiu atingir seu objetivo e tornou a base de dados mais leve e adequada para as etapas seguintes de processamento. O total de 17.271 dimensões do conjunto de dados foi reduzido para 50. Diminuindo o custo de processamento computacional.

Desta forma, conclui-se que a metodologia aplicada a esta pesquisa é adequada e atingiu seus objetivos propostos. Entretanto, é conveniente realizar ajustes para estudos futuros: utilizar de outras bases de dados com mais diversidade de cadastros; aplicar métodos de otimização ao modelo *Diffusion Map* para redução de dimensionalidade a fim de perder-se o mínimo de informação; usar de técnicas com menor custo computacional e mais eficazes que o *Grid-Search* para escolha de hiperparâmetros do modelo SVM; ou aplicar diferentes modelos de aprendizado de máquina similares ao SVM como *Random Machines* (Ara et al., 2021).

Referências Bibliográficas

Alhakeem, Z. M., Jebur, Y. M., Henedy, S. N., Imran, H., Bernardo, L. F., e Hussein, H. M. (2022). Prediction of ecofriendly concrete compressive strength using gradient boosting regression tree combined with gridsearchcv hyperparameter-optimization techniques. *Materials*, 15(21):7432.

Altenbernd, B. (2021). Classificação de texto de reclamações de empreendimentos imobiliários utilizando machine learning. [trabalho de conclusão de curso, graduação em estatística]. Instituto de Matemática e Estatística, Universidade Federal do Rio Grande do Sul.

Alvares, R. V., Garcia, A. C., e Ferraz, I. (2005). Stembr: A stemming algorithm for the brazilian portuguese language. *Progress in Artificial Intelligence*, 3808:693–701.

Amasifuen, F. S. T. (2021). Uso de aprendizado de máquinas para reconhecimento de padrões. [trabalho de conclusão de curso, graduação em estatística]. Departamento de Estatística, Universidade Federal Fluminense.

Anderson, C. (2015). *Creating a data-driven organization: Practical advice from the Trenches*. O'Reilly.

Ara, A., Maia, M., Louzada, F., e Macêdo, S. (2021). Random machines: A bagged-weighted support vector model with free kernel choice. *Journal of Data Science*, 19(3):409–428.

Araujo, A. C. M. d. S. (2013). *Tributação aduaneira: à luz da jurisprudência do CARF*. MP Editora.

Banisch, R., Trstanova, Z., Bittracher, A., Klus, S., e Koltai, P. (2020). Diffusion maps tailored to arbitrary non-degenerate ito processes. *Applied and Computational Harmonic Analysis*, 48:242–265.

Batista, R. d. A. (2017). Classificação automática de códigos ncm utilizando o algoritmo naïve bayes. [trabalho de conclusão de curso, pós-graduação em engenharia de software, ênfase em soluções de governo]. Universidade de Santa Cruz do Sul.

Batta, M. (2020). Machine learning algorithms - a review. *International Journal of Science and Research (IJSR)*, 9(1):381–386.

Belkina, A. C., Ciccolella, C. O., Anno, R., Halpert, R., Spidlen, J., e Snyder-Cappione, J. E. (2019). Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and allow analysis of large datasets. *bioRxiv*.

Bird, S., Loper, E., e Klein, E. (2009). *Natural language processing with python*. O'Reilly.

Boser, B. E., Guyon, I. M., e Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. Pittsburgh, Pennsylvania, USA.

Brasil (2001). Medida provisória nº 2158-35, de 24 de agosto de 2001. *Diário Oficial da República Federativa do Brasil*. Seção 1, p. 26.

Coelho, A. R. (2007). Stemming para a língua portuguesa: estudo, análise e melhoria do algoritmo rslp. [trabalho de conclusão de curso, graduação em ciência da computação]. Instituto de Informática, Universidade Federal do Rio Grande do Sul.

Coifman, R. R. e Lafon, S. (2006). Diffusion maps. *Applied and Computational Harmonic Analysis*, 21:5–30.

Cortes, E. G. (2019). Quando, onde, quem, o que ou por que? um modelo híbrido de classificação de perguntas para sistemas de question answering. [dissertação de mestrado em ciência da computação]. Instituto de Informática, Universidade Federal do Rio Grande do Sul.

Denig, M. R. d. S. (2015). Uso de técnicas de estatística e redes neurais para previsão de incêndio em Áreas florestais. [trabalho de conclusão de curso, graduação em estatística]. Instituto de Matemática e Estatística, Universidade Federal do Rio Grande do Sul.

Dias, A. C. (2008). O sistema harmonizado pode ser utilizado como barreira técnica? : análise dos casos da cachaça, da sandália de dedo, de borracha, e dos cortes de frango, salgados e congelados, no período de 2002 a 2007. [dissertação de mestrado em economia em comércio exterior e relações internacionais]. Centro de Ciências Socias Aplicadas, Universidade Federal de Pernambuco.

dos Santos, L. S. F. C. (2015). Estudo online da dinâmica espaço-temporal de crimes através de dados da rede social twitter. [dissertação de mestrado em estatística]. Instituto de Ciências Exatas, Universidade Federal de Minas Gerais.

Fonseca, A. H. d. O. (2019). Detection and classification of ultrasonic vocalizations from neonatal mice using machine learning. [dissertação de mestrado em microeletrônica]. Instituto de Informática, Universidade Federal do Rio Grande do Sul.

Frakes, W. B. e Baeza-Yates, R. (1992). *Information retrieval: Data Structures & Algorithms*. Prentice Hall.

Gerhardt, T. E. e Silveira, D. T. (2009). *Métodos de pesquisa*. Editora da UFRGS.

Goulart, P. J. (2021). Classificação de mercadorias: O uso das definições das agências reguladoras sob o enfoque do resp nº 1.555.004/sc. *Revista Tributária e de Finanças Públicas: RTrib*, (147):247–264.

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., e Oliphant, T. E. (2020). Array programming with numpy. *Nature*, pages 357–362.

Haykin, S. (2001). *Redes neurais: Principios e Pratica*. Bookman.

Hinton, G. E. e Roweis, S. (2002). Stochastic neighbor embedding. *Advances in Neural Information Processing Systems*, 15.

Houaiss, A. e Villar, M. d. S. (2001). *Minidicionário Houaiss da Língua Portuguesa*. Objetiva.

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9:90–95.

Kavzoglu, T. e Colkesen, I. (2009). A kernel functions analysis for support vector machines for land cover classification. *International Journal of Applied Earth Observation and Geoinformation*, 11(5):352–359.

Lopes, M. A. d. S. (2020). t-sne paralelo: Uma técnica paralela para redução de dimensionalidade de dados aplicada em cidades inteligentes. [tese de doutorado em ciências]. Centro de Tecnologia, Universidade Federal do Rio Grande do Norte.

Luna Fº, B. (1998). Seqüência básica na elaboração de protocolos de pesquisa. *Arquivos Brasileiros de Cardiologia*, 71(6):735–740.

Lundh, F. (1999). An introduction to tkinter. https://ftp.math.utah.edu/u/ma/hohn/linux/tcl/an-introduction-to-tkinter.pdf. Acessado em: 2023-11-27.

Luppes, J. (2019). Classifying short text for the harmonized system with convolutional neural networkss. [tese de mestrado em ciência da computação]. Radboud University.

Martins, C. A. (2003). Uma abordagem para pré-processamento de dados textuais em algoritmos de aprendizado. [tese de doutorado em ciências de computação e matemática computacional]. Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.

Matsubara, E. T., Martins, C. A., e Monard, M. C. (2003). Pretext: uma ferramenta para pré-processamento de textos utilizando a abordagem bag-of-words. Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.

McKinney, W. (2010). Data structures for statistical computing in python. *Proceedings of the 9th Python in Science Conference*, pages 56–61.

McTear, M., Callejas, Z., e Griol, D. (2016). *Conversational interface: Talking to smart devices*. Springer International Publishing.

MERCOSUL (2023). Ncm em vigor. https://www.mercosur.int/politica-comercial/ncm/. Acessado em: 2023-11-27.

Ministério da Fazenda (2019). Ncm. https://www.gov.br/receitafederal/pt-br/assuntos/aduana-e-comercio-exterior/classificacao-fiscal-de-mercadorias/ncm. Acessado em: 2023-11-27.

Ministério do Desenvolvimento, Indústria, Comércio e Serviços (2023). Nomenclatura comum do mercosul - ncm. http://mdic.gov.br/index.php/comercio-exterior/contatos/9-assuntos/categ-comercio-exterior/343-certificado-form-13. Acessado em: 2023-11-27.

Moreira, L. A. S. (2010). Funções de difusão e regiões de zero de campos de vetores planares. [dissertação de mestrado em matemática]. Departamento de Matemática, Pontifícia Universidade Católica do Rio de Janeiro.

Nadler, B., Lafon, S., Coifman, R., e Kevrekidis, I. G. (2008). Diffusion maps - a probabilistic interpretation for spectral embedding and clustering algorithms. *Principal Manifolds for Data Visualization and Dimension Reduction*, 58:238–260.

Orengo, V. M. e Huyck, C. (2001). A stemming algorithmm for the portuguese language. *Proceedings Eighth Symposium on String Processing and Information Retrieval*. Laguna de San Rafael, Chile.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., e Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pinheiro, P. e Amaris, M. (2022). ma metodologia em cascata de quatro etapas para classificar códigos ncm usando técnicas de pln. Faculdade de Engenharia de Computação, Universidade Federal do Pará.

Pio, P. B. e Sodré, I. C. (2019). Visualização exploratória e predição de desempenho de dados educacionais da universidade de brasília. [trabalho de conclusão de curso, graduação em ciência da computação]. Departamento de Ciência da Computação, Universidade de Brasília.

Porte, J., Herbst, B., Hereman, W., e van der Walt, S. (2008). An introduction to diffusion maps. *Proceedings of the 19th Symposium of the Pattern Recognition Association of South Africa (PRASA 2008)*. Cidade do Cabo, África do Sul.

Prado, P. d. O. (2020). Aplicação de técnicas multivariadas para visualização de dígitos manuscritos. [trabalho de conclusão de curso, graduação em estatística]. Instituto de Matemática e Estatística, Universidade Federal Fluminense.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing.

Rossi, R. G. (2011). Representação de coleções de documentos textuais por meio de regras de associação. [dissertação de mestrado em ciências de computação e matemática computacional]. Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.

Saito, T. e Rehmsmeier, M. (2015). The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3).

Siscomex (2022). Sistema harmonizado. https://www.gov.br/siscomex/pt-br/servicos/aprendendo-a-exportarr/planejando-a-exportacao-1/sistema-harmonizado. Acessado em: 2023-11-27.

Tableau (2019). 8 natural language processing (nlp) examples. https://www.tableau.com/learn/articles/natural-language-processing-examples. Acessado em: 2023-11-27.

Van der Maaten, L. e Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Van Rossum, G. (2020). *The Python Library Reference, release 3.8.2*. Python Software Foundation.

Van Rossum, G. e Drake, F. L. (2009). *Python 3 Reference Manual*. CreateSpace.

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021.

Wattenberg, M., Viégas, F., e Johnson, I. (2016). How to use t-sne effectively. *Distill*.