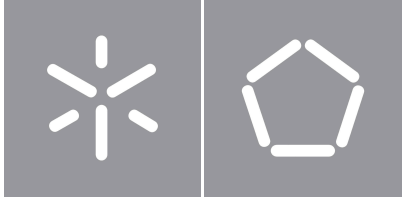
Universidade do Minho
Escola de Engenharia

Afonso Xavier Cardoso Marques

Utilização de Bases de Dados Vetoriais em Processos de Similaridade de Preferências de Clientes

novembro 2025



Universidade do Minho
Escola de Engenharia

Afonso Xavier Cardoso Marques

Utilização de Bases de Dados Vetoriais em Processos de Similaridade de Preferências de Clientes

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho efetuado sob a orientação de
Professor Doutor Orlando Manuel Oliveira Belo

novembro 2025

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.

Licença concedida aos utilizadores deste trabalho:

CC BY

https://creativecommons.org/licenses/by/4.0/

Agradecimentos

Em primeiro lugar, expresso o meu devido agradecimento ao Professor Doutor Orlando Manuel Oliveira Belo, o meu orientador, pelo seu auxílio e paciência durante a elaboração deste trabalho.

Agradeço aos meus pais, irmão e restante família, pela paciência, apoio e incentivo constantes a seguir a área de estudo que eu mais gostava, confiando sempre nas minhas decisões, e por terem sempre acreditado no meu sucesso profissional.

Declaração de Integridade

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Universidade do Minho, Braga, novembro 2025

Afonso Xavier Cardoso Marques

Resumo

Um sistema de similaridade realiza um processo que identifica e recupera dados ou objetos que são semelhantes a uma determinada consulta (query). Em vez de procurar correspondências exatas (como um motor de busca tradicional), este tipo de sistemas avalia a semelhança de dois itens com base em características e padrões. A utilização de processos de análise de similaridade de preferências de clientes permite agrupar clientes em grupos com interesses comuns. Esta abordagem possibilita a criação de recomendações personalizadas de produtos e serviços oferecidos, promovendo a satisfação e fidelização do cliente. Este trabalho de dissertação abordou o estudo e a aplicação da análise de similaridade para sistemas de preferências de clientes com recurso a bases de dados vetoriais, através de algoritmos e estruturas de dados especificamente projetados para medir a "distância"entre dois elementos de dados representados como vetores. Utilizando técnicas como a similaridade do cosseno ou a distância euclidiana, foi possível avaliar e classificar as similaridades entre vários objetos, quer sejam do tipo texto, imagem ou qualquer outro tipo de dados. Para construir este sistema de análise de similaridade, fez-se primeiro um estudo comparativo entre os vários tipos de medidas de similaridade de vetores e sistemas que as usam seguido de outra análise comparativa para diferentes implementações de sistemas de bases de dados vetoriais. Após esta preparação inicial, procedeu-se à devida implementação do sistema, usando um conjunto de cenários de análise para proceder à avaliação do comportamento do sistema construído. Por fim, com base nos resultados obtidos, são apresentadas algumas propostas orientadas para a melhoria da qualidade do sistema.

Palavras-chave Sistemas de Bases de Dados Vetoriais, Análise de Similaridade, Preferências de Clientes, *Profiling* de Clientes

Abstract

A similarity system performs a process that identifies and retrieves data or objects that are similar to a given **query**. Instead of looking for exact matches (like a traditional search engine), these types of systems evaluate the similarity of two items based on characteristics and patterns. The use of customer preference similarity analysis allows grouping customers into clusters with common interests. This approach enables the creation of personalized recommendations for products and services offered, promoting customer satisfaction and loyalty. This dissertation addresses the study and application of similarity analysis for customer preference systems using vector databases, through algorithms and data structures specifically designed to measure the "distance"between two data elements represented as vectors. Using techniques such as cosine similarity or Euclidean distance, it is possible to evaluate and classify similarities between different objects, whether they are text, images or any other type of data. To build this similarity analysis system, a comparative study was first carried out between the various types of vector similarity measures and systems that use them, followed by another comparative analysis for different implementations of vector database systems. After this initial preparation, the system was properly implemented, using a set of analysis scenarios to evaluate the behavior of the constructed system. Finally, based on the results obtained, some proposals aimed at improving the quality of the system are presented.

Keywords Vector Database Systems, Similarity Analysis, Customer Preferences, Custom-er Profiling

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

AI - Artificial Intelligence.

API - Application Programming Interface.

BoW - Bag of Words.

CRM - Customer Relationship Management.

CSG - Continuous Skip-Gram.

CTO - Chief Technology Officer.

ERP - Enterprise Resource Planning.

HNSW - Hierarchical Navigable Small Worlds.

IA - Inteligência Artificial.

LLM - Large Language Models.

LSH - Locality Sensitive Hashing.

NNH - Nearest Neighbor Search.

PLN - Processamento de Linguagem Natural.

RAG - Retrieval Augmented Generation.

SSIM - Structural Similarity Index Measure.

SVM - Support Vector Machine.

TF-IDF - Term Frequency-Inverse Document Frequency.

Glossário

back-end - camada do sistema que gere a lógica, processamento, armazenamento de dados e outras operações que não sejam visíveis ao utilizador.

big data - enorme quantidade de informação estruturada e não estruturada que os humanos e as máquinas geram.

cluster - coleção de itens com características semelhantes.

código aberto - código fonte de um produto de *software* que é disponibilizado gratuitamente para consulta, modificação e redistribuição.

corpus - conjunto de documentos que servem de base para a descrição ou o estudo de um fenómeno.

data science - área de estudo que combina estatística, programação e análise para extrair informações relevantes de um conjunto de dados.

dataset - conjunto estruturado de dados, geralmente organizado em tabelas, usado para análises, treino de modelos ou extração de informações.

deep learning - subárea de **machine learning** que utiliza redes neuronais profundas para modelar e aprender representações complexas de dados.

embedding - representação vetorial densa de dados, como palavras ou imagens, projetada para capturar características semânticas ou estruturais.

espectrograma - gráfico que analisa dinamicamente a densidade espectral de energia.

feature - caraterística relevante de um objeto computacional.

feedback - resposta ou reação dada a uma ação, desempenho ou informação, usada para avaliar, corrigir ou melhorar comportamentos ou resultados.

focus group - realização de uma discussão em grupo, moderada por um facilitador, com o objetivo de obter perceções, opiniões e atitudes dos participantes sobre um determinado tema, produto ou serviço.

hardcoded - valores ou comportamentos definidos diretamente no código-fonte de um programa.

hashing - processo de conversão de dados numa sequência de letras e números de comprimento fixo, denominado valor hash.

insight - compreensão profunda ou descoberta significativa obtida a partir da análise de dados, observações ou experiências.

k-means - algoritmo não supervisionado de aprendizagem automática que agrupa conjuntos de dados não rotulados em diferentes *clusters*.

lematização - processo para identificar o lema de uma palavra flexionada ou para agrupar palavras pelo lema.

machine learning - ramo da inteligência artificial que permite que sistemas aprendam padrões e façam previsões a partir de dados, sem serem explicitamente programados para isso.

meta dados - dados que descrevem outros dados, como origem, formato, autor, data de criação ou outros atributos adicionais relevantes.

payload - carga útil de um dado computacional.

pipeline - conjunto de etapas ou processos sequenciais que transformam dados ou realizam tarefas específicas.

query - pedido para aceder a dados de uma base de dados para serem manipulados ou recuperados.

sharding - técnica de partição de dados que divide uma base de dados em subconjuntos menores, chamados *shards*, que são distribuídos por múltiplos servidores.

stemming - processo do **PLN** que reduz palavras ao seu radical (stem), eliminando sufixos e prefixos. O objetivo é tratar palavras com a mesma raiz como equivalentes, melhorando análises de texto e modelagem de linguagem.

stopwords - palavras comuns num idioma que geralmente são removidas durante o **PLN** porque não carregam muito significado semântico em análises baseadas em texto.

streaming - transmissão contínua de dados, como áudio ou vídeo, pela Internet, permitindo o consumo em tempo real sem a necessidade de um download completo.

survey - coleção de informações sobre um tópico específico.

tokenização - processo de segmentar frases em componentes menores, conhecidos como *tokens*. Estes *tokens* facilitam a compreensão e manipulação de texto por parte dos computadores.

web scraping - técnica usada para extrair dados da Internet, através da leitura e recolha de informações estruturadas de páginas web.

Capítulo 1

Introdução

1.1 Enquadramento

Um sistema de similaridade (Shimomura et al., 2021) realiza um processo que identifica e recupera dados ou objetos que são semelhantes a uma determinada consulta (**query**). Em vez de procurar correspondências exatas (como um motor de busca tradicional), este tipo de sistemas avalia a semelhança de dois itens com base em características e padrões. A pesquisa por similaridade é amplamente utilizada em muitos setores de atividade, ajudando a realizar uma grande diversidade de tarefas, como ajudar os sites de comércio eletrónico a sugerir produtos de que se pode gostar ou permitir que sistemas médicos identifiquem casos semelhantes em processos de diagnóstico.

A tecnologia associada a um sistema de similaridade é sofisticada. Usualmente, envolve a utilização de algoritmos e estruturas de dados, especificamente projetados para medir a "distância"entre dois elementos de dados. Baseando-se em representações vetoriais dos dados, nas quais cada objeto é transformado num vetor multidimensional, a similaridade entre vetores é medida utilizando-se técnicas como a similaridade do cosseno, a distância euclidiana ou a distância de Minkowski (Al-Shamri, 2022). Na prática, todas estas técnicas permitem avaliar e classificar as similaridades entre vários objetos, quer sejam do tipo texto, imagem ou qualquer outro tipo de dados, assumindo que previamente foram incorporados em vetores.

Na década de 1970 surgiu um modelo que utilizava um espaço vetorial para a recuperação de informação, o que permitia representar documentos como vetores num espaço de várias dimensões. Esse modelo foi formalizado pela primeira vez por Gerard Salton e os seus colegas, A. Wong e C. S. Yang, num trabalho que publicaram em 1975 intitulado "A Vector Space Model for Automatic Indexing"(Salton et al., 1975). Foi a partir desse modelo que surgiram as bases de dados vetoriais, estruturas que permitem armazenar

dados como vetores de alta dimensão. Nestas bases de dados, um vetor é uma representação matemática que codifica múltiplas características ou atributos de um objeto. Cada vetor tem um conjunto de dimensões, que pode variar de dezenas a milhares de dimensões, dependendo da complexidade dos dados. Os vetores são gerados aplicando algum tipo de transformação ou função de **embedding** sobre os dados brutos, como texto, imagens, áudio, vídeo, entre outros. A função de **embedding** dos dados pode ser baseada em modelos de **machine learning**, **embeddings** de palavras e algoritmos de extração de características (Han et al., 2023).

As bases de dados vetoriais destacam-se das bases de dados ditas tradicionais pelas suas características específicas, tais como o suporte para dados complexos e não estruturados, alta escalabilidade e desempenho e a possibilidade de realizar pesquisas rápidas e precisas com base na similaridade de vetores (Taipalus, 2024).

1.2 Motivação

A motivação para conceber e implementar sistemas de análise de similaridade surgiu da necessidade que as empresas têm em servir melhor os seus clientes e de se manterem ativas num mercado em constante evolução tecnológica. Os processos de análise de similaridade, que podem ser aplicados em diferentes propósitos, são essenciais para categorizar melhor os clientes do setor empresarial, permitindo assim que este atenda às necessidades específicas de diferentes segmentos de consumidores. Por meio da análise de similaridade, as empresas conseguem identificar relações ocultas nos dados e otimizar as suas estratégias de marketing, vendas e retenção de clientes.

De forma a traçar o perfil de um cliente, as empresas necessitam de armazenar dados sobre ele. Usualmente, esses dados provêm de várias fontes, como o histórico de compras, interações em plataformas digitais e o comportamento de navegação (Brown, 2024). Esses dados são então transformados em vetores para permitir comparações com base em métricas de similaridade que melhor se adaptem ao serviço pretendido. A identificação de padrões úteis requer uma infraestrutura robusta e algoritmos sofisticados para lidar com a alta dimensão dos dados e as complexidades dos relacionamentos entre variáveis.

As bases de dados vetoriais tornam-se particularmente úteis neste contexto, pois possibilitam a execução eficiente de pesquisas por similaridade, tornando-se cada vez mais comuns em sistemas de recomenda-

ções empregues atualmente por empresas como Google, Spotify e Amazon (Bussler, 2023). Para além destas, outras empresas utilizam também análise de similaridade e bases de dados vetoriais para otimizar as suas operações. De referir:

1. **Amazon.** A Amazon começou recentemente a utilizar bases de dados vetoriais e análise de similaridade no serviço MemoryDB. A empresa afirma que o seu modelo de pesquisa vetorial simplifica a arquitetura das aplicações que o utilizam, enquanto lhes oferece pesquisa vetorial de alta velocidade, geração de recuperação aumentada, deteção de anomalias, recuperação de documentos e recomendações em tempo real (Yun, 2024).

2. **Spotify.** O Spotify introduziu em 2023 uma biblioteca de **código aberto** chamada Voyager, uma evolução do antigo Annoy, baseado na biblioteca hnswlib. O Voyager é capaz de realizar pesquisas de vizinho mais próximo de forma eficaz e foi disponibilizado pelo Spotify para utilização em produção (Sobot, 2023).

3. **Google.** O Google implementou recentemente bases de dados vetoriais e pesquisa vetorial no serviço Memorystore. Esta atualização permite aos programadores realizar pesquisas vetoriais em latências ultra baixas em milhares de milhões de vetores. Esta melhoria é particularmente benéfica para aplicações que dependem de Inteligência Artificial (**IA**) generativa, como a geração aumentada de recuperação (**RAG**), sistemas de recomendação e pesquisa semântica (Palriwal, 2024).

4. **Chroma DB.** O Chroma DB é uma base de dados vetorial de código aberto concebida para o armazenamento e recuperação eficientes de incorporações vetoriais. A abordagem inovadora do ChromaDB para consultar bases de dados define um novo padrão de precisão e relevância, transformando a forma como a informação é acedida, tornando-a mais intuitiva e eficiente (Nidhiworah, 2024).

Estes exemplos ilustram como as empresas estão a impulsionar o uso de análise de similaridade e bases de dados vetoriais para obter **insights** mais profundos e fornecer experiências personalizadas aos seus clientes. Estas ferramentas continuam a evoluir, permitindo uma compreensão mais detalhada dos padrões de consumo e contribuindo para o sucesso das organizações num mercado cada vez mais competitivo.

1.3 Objetivos

A utilização de processos de análise de similaridade de preferências de clientes permite agrupar clientes em grupos com interesses comuns. Esta abordagem possibilita a criação de recomendações personalizadas de produtos e serviços oferecidos, promovendo a satisfação e fidelização do cliente.

Com base neste contexto, este trabalho de dissertação procurou atingir os seguintes objetivos:

1. Conhecer as bases de dados vetoriais, compreender como são implementadas e como se integram em aplicações."
2. Estudar as diferentes abordagens de desenvolvimento e implementação de sistemas de análise de similaridade.
3. Estudar, identificar e analisar os processos, técnicas e modelos de análise de similaridade utilizados recentemente, com especial foco na sua aplicação em sistemas de *customer profiling*.
4. Estabelecer um modelo computacional para análise de similaridade de preferências de clientes utilizando um sistema de bases de dados vetoriais.

1.4 Trabalho Realizado

O trabalho realizado nesta dissertação iniciou-se com a compreensão das diferentes abordagens existentes para realizar análise de similaridade e os conceitos de base de dados vetorial e *customer profiling*. Visto que estas eram as bases fundamentais para o desenvolvimento desta dissertação, foram realiza-das três análises detalhadas destas áreas, explorando como as suas técnicas podem ser aplicadas e quais ferramentas estão disponíveis para tal.

De forma a explorar estes conceitos, foram realizadas três **surveys**. A primeira aborda tópicos tais como a noção de análise de similaridade nas suas várias formas, desde o contexto matemático, textual e visual, e as diferentes aplicações práticas que existem de análise de similaridade. Analisa-se também os diferentes processos de pré-processamento de dados bem como as técnicas de **embedding** mais conhecidas para

gerar vetores.

A segunda **survey** explora as bases de dados vetoriais, resumindo a sua essência e a sua utilidade prática, dando exemplos reais de aplicações que as utilizam. Aborda-se a temática da indexação de vetores e da pesquisa nessas bases de dados, fazendo também uma descrição mais detalhada de duas medidas utilizadas para calcular a similaridade vetorial.

A terceira (e última) aborda o conceito de *customer profiling*, explicando como este constitui um processo essencial em contextos de marketing, apresentando modelos e técnicas, tanto manuais como automatizadas, que possibilitam a criação de perfis distintos de clientes. Adicionalmente, são referidas algumas aplicações práticas, com o intuito de evidenciar a relevância e a utilidade deste processo em cenários reais.

Concluída a exposição dos conceitos teóricos, avançou-se para a fase de implementação do sistema. O primeiro passo consistiu na definição da área de aplicação, uma vez que o *customer profiling* pode ser utilizado em múltiplos contextos. A escolha da área foi devidamente justificada com base na adequação ao objetivo do projeto. Posteriormente, foi concebido um caso de estudo alinhado com essa área, definindo-se os requisitos e funcionalidades do sistema a desenvolver.

Seguiu-se a apresentação da organização e funcionamento do sistema, com destaque para aspetos fundamentais como a arquitetura adotada e os serviços disponibilizados. Após esta descrição, realizou-se uma análise crítica do sistema com o intuito de verificar o cumprimento dos objetivos estabelecidos.

1.5 Estrutura do Documento

Para além do presente capítulo, esta dissertação está estruturada em mais cinco capítulos. De referir:

- Capítulo 2. Análise de Similaridade - Neste capítulo é feita uma contextualização do conceito de análise de similaridade, bem como os diferentes contextos onde esta pode ser aplicada. É apresentada uma visão geral sobre o estado da arte, as suas principais aplicações e as ferramentas que permitem a sua aplicação.
- Capítulo 3. Bases de Dados Vetoriais – Nesta parte da dissertação explora-se o conceito de bases de dados vetoriais. Inclui uma noção básica deste tipo de base de dados, alguns exemplos reais,

a enumeração de alguns métodos de indexação de vetores e ainda uma pequena introdução aos métodos de pesquisa nestas bases de dados.

- Capítulo 4. *Customer Profiling* – Aqui aborda-se o conceito de *profiling* de clientes, dando especial relevo aos processos que existem e em que contextos reais são aplicados.

- Capítulo 5. Um Sistema de *Profiling* - Neste capítulo é exposto o sistema que foi desenvolvido, no qual são devidamente justificadas as decisões que foram tomadas durante o seu desenvolvimento. É também defini-do o caso de estudo que contextualiza a área de aplicação do sistema e são indicadas as principais ferramentas utilizadas para a implementação do sistema proposto, seguido da devida justificação para a sua escolha.

- Capítulo 6. Conclusões e Trabalho Futuro - neste último capítulo apresenta-se as conclusões do trabalho a um nível global, indicando possíveis melhorias para o futuro caso se justifique.

Capítulo 2

Análise de Similaridade

2.1 Noção e Aplicações

A análise de similaridade tem como objetivo identificar e quantificar o grau de semelhança entre diferentes objetos, sendo uma área essencial em sistemas de recomendação, recuperação de informação, visão computacional, entre outros. Para se falar de análise de similaridade, primeiro é necessário definir esse conceito. O termo "similaridade" é utilizado desde a antiguidade, tendo evoluído ao longo dos anos devido aos avanços nas diversas áreas científicas e filosóficas (Grigoriadou et al., 2021). Isto faz com que a sua definição fique sujeita a alguma subjetividade. Por exemplo, numa perspetiva filosófica, a similaridade é a existência de algo comum, como um atributo ou propriedade, que esteja presente num conjunto de objetos (Grigoriadou et al., 2021), sendo um exemplo disto um sistema de filtragem numa loja online, como os encontrados na Amazon e na Temu, por exemplo, nos quais se definem regras, inseridas manualmente, para comparar os produtos e determinar se estes são semelhantes ou não. Por outro lado, em domínios científicos, como a Geometria ou a Física, a análise de similaridade consiste na comparação de dimensões e relações físicas (Grigoriadou et al., 2021), sendo necessário frequentemente o uso de métodos matemáticos para a medir. De seguida, abordar-se-ão as diferentes abordagens, metodologias, processos e técnicas utilizadas para medir similaridade, bem como as suas aplicações mais generalizadas.

2.2 Análise de similaridade em diferentes contextos

Na análise de similaridade, é necessário medir a proximidade entre dois objetos, em diferentes tipos de contexto. Por exemplo, se o contexto for puramente matemático e se estiverem a avaliar apenas objetos desse ramo, isto é, objetos formalmente definidos e com a qual se podem fazer deduções e provas matemáticas, tais como vetores, formas geométricas, conjuntos e funções, a similaridade pode ser medida através de fórmulas bem conhecidas que podem ser encontradas na Teoria dos Conjuntos e na Álgebra

Linear.

No entanto, se o contexto for semântico, visual ou auditivo, nos quais a precisão é menor dando espaço à subjetividade, surge a necessidade de introduzir métodos que permitam desconstruir ou converter os objetos de forma a conseguir determinar a sua aproximação. Por exemplo, existem várias razões para se classificar duas canções como semelhantes: terem sido escritas pelo mesmo compositor, serem tocadas pela mesma banda ou serem do mesmo género. A semelhança entre elas pode ser avaliada com base num dos muitos componentes ou num subconjunto arbitrário de componentes, ignorando o resto (Changpinyo et al., 2013). Assim, surgiu a necessidade de combinar e aplicar diferentes medidas de similaridade de forma que estas se adequem ao contexto estudado.

Similaridade na Matemática

Nos contextos puramente matemáticos, existem métricas que permitem determinar a proximidade entre dois objetos. Estas métricas regem-se pela precisão numérica. As mais básicas são as distâncias matemáticas, mais precisamente a distância euclidiana, usada para medir similaridade em espaços métricos (e.g., atributos contínuos), a distância de Manhattan para cenários em que caminhos retos importam e a distância de Minkowski, que surge como uma combinação generalizada das distâncias Euclidiana e Manhattan. Quanto mais curta for a distância, mais similares são os dois pontos no espaço vetorial. Um exemplo prático surge nos algoritmos aplicados em **machine learning**, onde a distância de Minkowski é fundamental para algoritmos que dependem da medição da semelhança ou diferença entre pontos de dados. Um exemplo proeminente é o algoritmo k-Nearest Neighbors (k-NN), que classifica os pontos de dados com base nas categorias dos seus vizinhos mais próximos (Chugani, 2024).

Existem coeficientes de similaridade, tais como a similaridade de cossenos e o coeficiente de Jaccard. A similaridade de cossenos (COS) mede a aproximação entre dois vetores num espaço vetorial, avaliando o valor do cosseno do ângulo compreendido entre eles conjuntos (Taipalus, 2024). Esta medida é usada em várias áreas nas quais o comprimento dos vetores não é tão importante quanto a sua direção, o que poderá resultar em valores menos fiáveis (Al-Shamri, 2022). Um exemplo prático inclui a sua ampla utilização como medida de avaliação da similaridade de textos, onde cada valor do vetor indica um peso para uma palavra ou conceito usado

no texto, criando assim um espaço vetorial.

Fora do domínio dos vetores, o coeficiente de Jaccard surge como uma forma de medir a similaridade em conjuntos binários ou discretos. É amplamente aplicado em diversas áreas, como Biologia, Ecologia, Recuperação de Informação ou Ciência de Dados (Learn Statistics Easily, 2024). Este coeficiente é calculado como a razão entre o número de elementos comuns aos dois conjuntos e o número total de elementos que pertencem a pelo menos um dos conjuntos (Taipalus, 2024). Todavia, possui algumas limitações como, por exemplo, não leva em consideração a frequência dos elementos dentro dos conjuntos, o que pode ser uma desvantagem em situações onde a quantidade de elementos é relevante. Além disso, pode ser sensível a conjuntos pequenos, nos quais a variação nos elementos pode resultar em mudanças significativas no valor do índice (Learn Statistics Easily, 2024). Para formas geométricas, a similaridade pode ser medida recorrendo à comparação entre área e volume ou então através de um conjunto de critérios pré-definidos, como é o caso dos triângulos, onde se podem aplicar os critérios AA (Ângulo-Ângulo), LAL (Lado-Ângulo-Lado) e LLL (Lado-Lado-Lado) (Homework Helpr, n.d.).

Similaridade Semântica

A comparação entre dois textos, quando realizada sem o auxílio de tecnologia, pode introduzir muita subjetividade, pois a interpretação vai variar de pessoa para pessoa. Assim, de forma a realizar comparações entre segmentos de texto, é necessário recorrer a métodos computacionais que desconstroem o texto de forma que este possa ser representado numa vertente mais matemática. Aqui a única abordagem possível é a conversão em vetor, onde diferentes componentes do texto são distribuídas pelos coeficientes de um vetor. Um dos exemplos mais simples desta abordagem é o modelo Bag of Words (**BoW**), que transforma uma sequência de texto num vetor numérico baseado na frequência das palavras (Gurung, 2021). Por exemplo, se considerarmos as frases:

1. "Eu gosto de matemática"
2. "Matemática é interessante"
3. "Eu gosto de ciência"

e criarmos um vocabulário com todas as palavras únicas presentes nas frases, obtemos o seguinte:

Vocabulário = [eu, gosto, de, matemática, é, interessante, ciência]

Cada frase pode ser representada como um vetor binário, no qual é indicado a presença das palavras do vocabulário (Tabela 1).

FRASE	eu	gosto	de	matemática	é	interessante	ciência
"Eu gosto de matemática"	1	1	1	1	0	0	0
"Matemática é interessante"	0	0	0	1	1	1	0
"Eu gosto de ciência"	1	1	1	0	0	0	1

Tabela 1: Modelo BoW

Cada linha da tabela representa um vetor numérico correspondente à frase. No entanto este modelo apresenta baixa complexidade, não sendo ideal para lidar com documentos maiores.

Também existem outros modelos de espaço vetorial mais complexos que são empregues por aplicações que lidam com documentos de texto. Um grande exemplo é o Term Frequency-Inverse Document Frequency ou **TF-IDF** que é uma métrica estatística utilizada no ramo do processamento de linguagem natural (**PLN**) e recuperação de informações. O objetivo é de avaliar a importância relativa de uma palavra ou termo num documento dentro de um corpus, que é um conjunto de documentos (Karabiber, n.d.). Esta métrica é amplamente utilizada em várias aplicações, como classificação de documentos, recuperação de informação, mecanismos de pesquisa, análise de sentimentos e recomendação de conteúdo (Karabiber, n.d.), ajudando a identificar e destacar os termos mais relevantes num conjunto de documentos. O **TF-IDF** não exige pré-processamento dos dados, no entanto este pode ser aplicado, usando técnicas conhecidas como a **tokenização** (divisão em palavras ou termos), a remoção de **stopwords** (palavras muito comuns que não contribuem muito para o significado semântico) e, opcionalmente, a aplicação de técnicas de normalização, como a redução de palavras ao seu radical, eliminando sufixos e prefixos através do **stemming** (Karabiber, n.d.). O **TF-IDF** por si só não

pode ser considerado uma métrica de similaridade, mas constitui um passo importante para realizar análise de similaridade num contexto semântico. É amplamente utilizado em aplicações de motores de busca e de extração de palavras-chave (Gurung, 2021).

O **BoW** e o **TF-IDF** possuem algumas limitações. A primeira é que os vetores obtidos têm o mesmo tamanho do vocabulário, que é o conjunto de todas as palavras que ocorrem no texto, o que se torna um problema quando se tem vocabulários muito grandes, o que é bastante comum no dia a dia, em que os documentos podem possuir vocabulários com muitos milhares de palavras. A segunda é que não conseguem captar significado entre palavras, isto é, extraem pouca informação semântica e sintática dos textos (Mustapic, 2024). Como alternativa ao **BoW** e **TF-IDF**, existem os modelos de Word Embedding. Esta abordagem tem a mesma finalidade, isto é, transformar segmentos de texto em vetores matemáticos para serem analisados por um computador. No entanto a complexidade do resultado obtido é diferente. Modelos como Word2vec, GloVe e FastText são alguns dos mais conhecidos.

Desenvolvido em 2013 por Tomáš Mikolov e os seus colegas na Google, o Word2Vec é um conjunto de abordagens de **machine learning** que é capaz de encontrar palavras com contextos semelhantes e agrupá-las. Neste método, considera-se como "contexto" o grupo de palavras que envolve uma determinada palavra numa frase ou num documento. Este agrupamento resulta num conjunto de vetores com significados semelhantes. Os vetores que representam palavras com significados semelhantes são posicionados próximos uns dos outros num espaço de alta dimensão (Swimm Team, n.d.). Esta aproximação no espaço vetorial facilita o processo de comparação computacional.

Similaridade Visual e Auditiva

A exploração da análise de similaridade em contextos matemáticos e semânticos abre a porta a outras áreas nas quais a similaridade pode ser estudada, tais como a análise de imagem e de áudio. Faz parte da natureza humana analisar tudo à nossa volta, comparando e detetando padrões em tudo o que nos rodeia. Os nossos estímulos visuais e auditivos permitem que sejamos capazes de detetar, por exemplo, se dois quadros são parecidos ou reconhecer um som que ouvimos e identificar o que o produziu. Traduzir estas capacidades para um computador

exige que primeiro seja realizado o **embedding** dos dados e só depois se faça a comparação.

Atualmente, existem várias aplicações que estendem este conceito para que praticamente todo o tipo de dados possa ser convertido num vetor para ser comparado computacionalmente. No que diz respeito aos dados visuais, estes podem ser comparados de forma direta, usando métodos como o índice de similaridade estrutural (**SSIM**), que mede diretamente a semelhança estrutural entre imagens focando-se na luminância, no contraste e na estrutura, detetando assim as diferenças mais percetíveis (Bakurov et al., 2022). Não usa qualquer tipo de **embedding** para obter um resultado, o que pode gerar algumas limitações. Se se pretender converter uma imagem num vetor, será necessário usar técnicas baseadas em **deep learning**, tais como redes neuronais convolucionais, como o modelo VGG16, que extrai características hierárquicas de imagens que são robustas a variações na aparência do objeto e ao ruído de fundo ou modelos pré-treinados, como ResNet (rede neural residual), que realiza a extração de características para criar incorporações (Daniella, 2024) que podem ser comparadas utilizando métricas como a similaridade do cosseno, que foi mencionada anteriormente. No áudio, as opções disponíveis incluem ferramentas como o CLAP e o OpenL3, sendo a primeira a mais conhecida e amplamente usada.

2.3 Processos e técnicas

Pré-Processamento de Dados

Os sistemas que lidam com muitos dados necessitam de ser capazes de fazer o seu devido pré-processamento. Relembrando o capítulo introdutório deste documento, o sistema proposto nesta dissertação envolve o uso de bases de dados vetoriais e **embedding** de dados, pelo que é essencial que o processamento seja devidamente feito, evitando o máximo possível de perda de informação. Existem várias abordagens para pré-processamento de dados, que variam com o tipo de dado que se está a abordar. No espectro dos dados em formato textual, a pré-preparação inclui processos como a remoção de **stopwords** e a **tokenização**, onde se eliminam palavras muito frequentes (ex.: "o", "e", "de") que não agregam significado e se divide o texto em palavras ou caracteres. Adicionalmente pode-se usar o **stemming** para reduzir

palavras à sua raiz (ex.: "correndo"→ "correr") mas sem considerar a gramática o que pode gerar palavras inexistentes, ou a **lematização** que faz essencialmente o mesmo, mas levando em conta o significado e a gramática, gerando sempre palavras válidas no idioma. Ambas as for-mas atuam como uma forma de normalização de dados textuais (Figura 1).

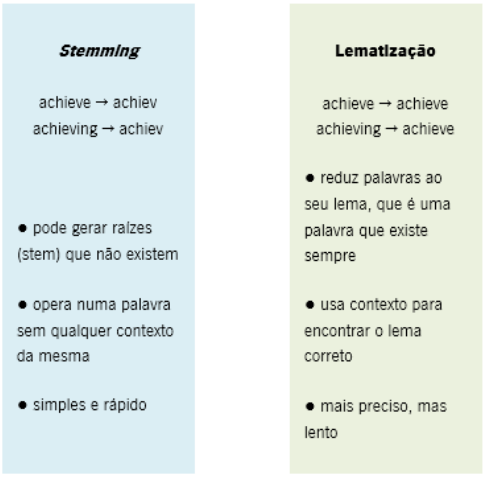


Figura 1: Diferenças entre *stemming* e lematização

Por outro lado, quando se trata de dados visuais, como imagens, é essencial garantir que estas estão num formato compatível com o modelo computacional usado para a extração de **features**. Assim, alguns dos passos que podem ser tomados incluem redimensionar as imagens de forma a corresponder ao tamanho esperado pelo modelo, garantindo assim um tamanho de entrada consistente, sendo especialmente útil em modelos que foram treinados em imagens de tamanho fixo, a normalização dos valores dos pixeis, para que fiquem dentro de um intervalo específico, normalmente entre 0 e 1 ou -1 e 1, ajudando a estabilizar o treino, tornando os da-dos mais comparáveis, e a conversão da imagem para a escala de cinzentos ou outros formatos para simplificar a informação e reduzir a complexidade dos dados de entrada (Daniella, 2024).

No áudio, tal como as imagens, será necessário garantir que os dados estão no formato espe-rado pelo modelo que os vai avaliar, permitindo assim que todos os dados sejam avaliados nas mesmas condições. Neste contexto, o pré-processamento passa pela remoção de ruído, tais como barulho de fundo ou interferências, e longos silêncios, eliminando assim **embeddings** desnecessários. Também é comum proceder à normalização dos dados, através do dimensio-namento da amplitude dos sinais de áudio, o que garante que o modelo não é tendencioso para

sinais com níveis de energia mais altos ou mais baixos (GeeksforGeeks, 2023).

Técnicas de Embedding

Previamente foram mencionadas técnicas que permitem transformar dados de diversos formatos (texto, imagem, áudio), que já foram submetidos a pré-processamento, em vetores que podem ser comparados por métodos matemáticos. De seguida, apresentaremos com mais detalhe algumas das ferramentas que geraram maior interesse no âmbito deste trabalho.

- Word2Vec

O Word2Vec é uma das ferramentas mais comuns para *word embedding* atualmente em uso. Falando em termos técnicos, o Word2Vec é uma rede neural de duas camadas que processa texto proveniente de lotes de dados textuais, processando-os e devolvendo um espaço vetorial multidimensional. Cada palavra única nos dados recebe um vetor correspondente no espaço, onde o seu posicionamento é determinado pelos significados semânticos das palavras e pela proximidade com outras palavras (Swimm Team, n.d.). A nível de arquitetura, este modelo pode ser implementado utilizando ou o modelo Continuous Bag of Words (C**BoW**) ou o modelo Continuous Skip-Gram (**CSG**), onde o C**BoW** usa o contexto, isto é, as palavras vizinhas, para prever a palavra central, sendo mais rápido e eficaz para conjuntos de dados menores; e o **CSG** usa a palavra central para prever o contexto, funcionando melhor com dados grandes sendo capaz de aprender melhor as relações entre palavras raras. Assumindo a escolha da arquitetura em **CSG**, o próximo passo consiste em inicializar os vetores para treinar o modelo.

Inicialmente, as palavras de um **corpus** são convertidas por *one-hot enconding* em vetores binários, semelhante ao exposto anteriormente na Tabela 1. Assumindo que temos um **corpus** composto pelas seguintes frases em baixo, o tamanho do vocabulário será igual a dez, gerando os seguintes vetores de entrada:

1. “O cão viu um gato”
2. “O cão perseguiu o gato”

3. “O gato subiu a uma árvore”

”a”	[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
”árvore”	[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
”cão”	[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
”gato”	[0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
”o”	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
”perseguiu”	[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
”subiu”	[0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
”um”	[0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
”uma”	[0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
”viu”	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1]

Neste formato, os vetores têm pouca utilidade prática, servindo apenas como valores de entrada para a primeira camada do modelo. Na segunda camada, ou camada escondida, o modelo cria uma matriz de pesos que permite medir as características de cada palavra (Ali, 2021). Antes de se começar a treinar o modelo, esta matriz é gerada com valores pequenos e aleatórios, sendo progressivamente atualizada pela própria rede neuronal durante o treino. Uma possível matriz de pesos para os vetores poderia ser:

[(0.2 0.4 0.6)
(0.1 0.5 0.7)
(0.3 0.8 0.2)
(0.5 0.9 0.1)
(0.7 0.3 0.5)
(0.6 0.2 0.8)
(0.4 0.7 0.3)
(0.9 0.1 0.4)
(0.8 0.5 0.2)
(0.3 0.6 0.9)]

Portanto, se quisermos um **embedding** para a palavra gato, o seu vetor [0, 0, 0, 1, 0, 0, 0, 0, 0, 0] é multiplicado pela matriz de pesos, retornando, por enquanto, a linha (0.5 0.9

0.1) da matriz. Após subsequentes iterações de treino usando o modelo **CSG**, a matriz de pesos na camada escondida será atualizada de forma a representar a aproximação sintática e semântica das palavras. Os **embeddings** resultantes serão atualizados na camada de saída, e serão usados para medir a similaridade entre as palavras.

- VGG16

Concebida e implementada por Karen Simonyan e Andrew Zisserman, a tecnologia VGG16 consiste numa rede neuronal convolucional, mais precisamente treze camadas convolucionais, onde cada uma aplica filtros com um pequeno campo recetivo com o tamanho mais pequeno possível para captar a noção de esquerda ou direita, cima ou baixo, e centro. Estas treze camadas são seguidas por três camadas totalmente conectadas (*Fully-Connected* ou FC), onde as duas primeiras têm 4096 canais cada, sendo que a terceira tem 1000. A camada final é a camada soft-max (Simonyan et al., 2015).

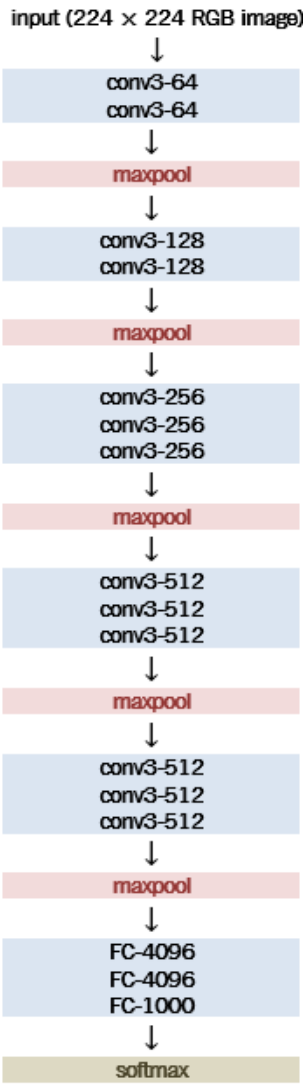


Figura 2: Arquitetura VGG16

De acordo com a Figura 2, uma imagem que é submetida a este método de **embedding**, primeiramente é redimensionada uniformemente em todas as direções até atingir um tamanho pré-definido. A seguir, a imagem é submetida a cada uma das camadas da rede, onde serão retiradas as várias características (**features**) que a compõem. Depois de atravessar várias camadas convolucionais, a imagem atinge as três últimas camadas, as FC. Antes do soft-max, na saída da última camada FC, é devolvido um vetor que pode ser usado para análise de similaridade.

- CLAP

O Contrastive Language-Audio Pretraining (CLAP) é um modelo que permite realizar a extração de **features** tanto de texto como de áudio, devolvendo vetores como resultado (Co-

chard, 2024). O CLAP acolhe um modelo de **deep learning** que, durante o treino, aprende a mapear simultaneamente textos e áudios para um espaço vetorial comum, onde vetores correspondentes a pares texto-áudio semelhantes ficam próximos entre si (Cochard, 2024). Esta abordagem facilita tarefas como recuperação de áudio baseada em texto e classificação de sons. O **embedding** dos excertos de áudio é realizado com o auxílio da ferramenta SwinTransformer, que utiliza **espectrogramas** para representar a energia do sinal de áudio em diferentes frequências ao longo do tempo (Unreal Speech, 2024).

Desde a sua introdução, o CLAP é aclamado como um dos mais importantes modelos de **deep learning** em vários domínios da Inteligência Artificial. Isto deve-se ao seu desempenho excecional quando lida com recuperação e classificação de áudio (Unreal Speech, 2024). No entanto, possui algumas limitações, tais como: variações de desempenho em diferentes conjuntos de dados, em particular em conjuntos que possuem desvios significativos dos dados de treino, dificuldades de adaptação, para determinados contextos aplicacionais, e elevados custos, computacionais (Unreal Speech, 2024). Após a geração dos vetores, estes são indexados num espaço vetorial seguindo determinados critérios, que serão abordados posteriormente nesta dissertação.

2.4 Sistemas e Aplicações

As técnicas previamente abordadas encontram aplicação em diversos contextos práticos. O Word2Vec, por exemplo, tem sido utilizado na construção de sistemas de recomendação para plataformas de comércio eletrónico. Neste cenário, os **embeddings** gerados pelo Word2Vec permitem representar produtos em formato vetorial, possibilitando a medição da similaridade entre eles e, consequentemente, a sugestão de produtos relevantes para os utilizadores (Joshi, 2024). Já o VGG16 tem sido explorado em áreas como a Medicina, mais particularmente nos diagnósticos. Um exemplo concreto é a análise de imagens de raio-X em casos de pneumonia, na qual o modelo é utilizado para extrair vetores representativos das imagens, permitindo identificar padrões de forma mais rápida e eficaz do que a observação humana direta (Mustafa, 2023). No que diz respeito ao CLAP, este já foi integrado em sistemas de síntese de voz com o objetivo de melhorar a prosódia, isto é, os padrões de entoação e ritmo, na fala gerada. Através da sua capacidade de associar variações prosódicas a diferentes contextos textuais, o modelo contribui para uma produção vocal mais natural e expressiva (Ye et al., 2023).

Capítulo 3

Bases de Dados Vetoriais

3.1 Noção e Utilidade

Nesta secção serão explorados os principais conceitos, técnicas e abordagens relacionadas ao desenvolvimento e utilização de bases de dados vetoriais, que têm emergido como uma componente fundamental em aplicações que envolvem dados não estruturados, como texto, imagens, áudio e vídeo. A sua principal característica é a capacidade de armazenar e consultar dados representados como vetores de alta dimensão, otimizando operações como a pesquisa por similaridade. Atualmente existem mais de vinte sistemas de gestão de bases de dados vetoriais, que foram impulsionados pelo surgimento de novas aplicações que lidam intensivamente com dados em grandes proporções, usufruindo da capacidade de realizar pesquisas de forma rápida, escalável e fiável (Pan et al., 2023). Alguns desses sistemas incluem o LanceDB, ChromaDB e o Oracle **AI** Vector Search, que serão explorados posteriormente.

Apesar das várias vantagens, tais como a escalabilidade e rapidez, os sistemas deste tipo e com esta dimensão apresentam alguns desafios na sua implementação e gestão, dentre os quais estão a ambiguidade na análise da similaridade semântica, o grande tamanho que os vetores podem atingir, o elevado custo de comparação de similaridade e a falta de propriedades estruturais que poderiam ser utilizadas para indexação (Pan et al., 2023).

As bases de dados vetoriais tendencialmente atingem elevadas dimensões, sendo por isso necessário garantir que o seu armazenamento e distribuição sejam feitos de forma a não perturbar o bom desempenho. Assim, após o **embedding** dos dados, os vetores gerados são armazenados em memória persistente e são utilizadas técnicas de indexação em espaço vetorial, com recurso a uma estrutura computacional, como uma árvore ou um grafo (Pan et al., 2023). É esta estrutura que vai contribuir para melhorar o desempenho das pesquisas que serão efetuadas.

3.2 Estrutura e Serviços

No que respeita à estrutura ou arquitetura, as bases de dados vetoriais tendem, de forma geral, a seguir um modelo genérico baseado na segmentação funcional em diferentes camadas, em que cada uma desempenha um papel específico no processamento e gestão dos dados (WEKA, 2025). Abaixo apresentam-se algumas das camadas mais comuns neste tipo de sistemas. De referir:

- **Inserção.** A camada responsável pelos processos de inserção de dados em diversos formatos e pela respetiva conversão para representações vetoriais. Envolve, por exemplo, a aplicação de modelos de **embedding** para transformar dados brutos em vetores numéricos (WEKA, 2025).
- **Armazenamento.** Nesta camada ocorre o armazenamento dos vetores em memória persistente. Além dos vetores propriamente ditos, são também aqui armazenados meta dados associados, como identificadores únicos, atributos estruturados do dado original (Taipalus, 2024) e outras informações contextuais relevantes.
- **Indexação.** Esta camada encarrega-se da construção e manutenção das estruturas de indexação vetorial, bem como da implementação dos algoritmos de pesquisa associados. Estas estruturas permitem acelerar a pesquisa por similaridade e são exploradas com maior detalhe na secção seguinte.
- **Execução de Queries.** A camada que lida com o processamento das **queries** submetidas à base de dados, desde a construção do vetor de consulta até à devolução dos resultados baseados na métrica de similaridade definida (WEKA, 2025).

Adicionalmente, podem ser consideradas outras camadas complementares que, embora nem sempre explicitamente documentadas, são frequentemente implementadas em sistemas de bases de dados vetoriais de grande escala. Entre estas, destaca-se uma camada de distribuição e resiliência, responsável pela gestão da distribuição dos dados através de técnicas como **sharding** e replicação, com o objetivo de garantir escalabilidade horizontal e tolerância a falhas. Existe também, por norma, uma camada de monitorização e análise, que visa recolher e analisar métricas de desempenho e estatísticas do sistema, contribuindo para a sua manutenção e otimização operacional (WEKA, 2025). Por fim, muitas implementações destas bases de dados incluem ainda uma camada de integração, que expõe uma interface de

programação de aplicações, ou **API**, permitindo a comunicação entre o motor da base de dados e os sistemas ou aplicações externas.

Vale a pena salientar que o modelo estrutural apresentado resulta da consolidação de padrões arquitetónicos recorrentes observados em várias bases de dados vetoriais contemporâneas. Trata-se, portanto, de uma representação genérica do seu funcionamento típico e dos serviços que prestam, o que não impede, contudo, a existência de motores que adotem abordagens diferentes ou que não implementem integralmente esta estrutura.

3.3 Indexação e Pesquisa de Dados

A indexação e pesquisa de dados em bases de dados vetoriais seguem uma abordagem distinta das bases de dados tradicionais, que operam sobre dados estruturados e recorrem a chaves ou filtros exatos para as consultas de informação, enquanto as bases de dados vetoriais lidam com representações numéricas de alta dimensão. Esta diferença fundamental implica que sejam adotadas técnicas próprias de indexação, que visam acelerar as pesquisas por similaridade e assegurar um desempenho eficiente em ambientes com grandes volumes de dados.

3.3.1 Métodos de Indexação

Quando se fala de indexação de dados numa base de dados tradicional, geralmente a primeira abordagem consiste em aplicar uma estrutura tabelar para fazer essa indexação, onde cada dado tem um identificador único seguido de vários atributos. Esta abordagem pode também ser aplicada às bases de dados vetoriais, semelhante ao exemplo da Oracle **AI** Vector Search. No entanto, tendo em conta que uma das principais metas das bases de dados vetoriais é otimizar a pesquisa e análise de similaridade dos dados, será mais adequado aplicar outras estruturas para indexar os dados vetoriais. Aqui, o objetivo consiste em criar uma estrutura que representa um espaço vetorial onde os vetores mais similares estão mais próximos uns dos outros, permitindo assim acelerar o processo de pesquisa pelo vizinho mais próximo, pois em vez de se iterar sobre todos os vetores, apenas se procura por aqueles que são mais similares, filtrando logo uma parte do conjunto de dados total.

Algumas das técnicas mais comuns de indexação vão ser exploradas em baixo, cada uma implementando uma estrutura computacional diferente. Vejamos cada uma delas.

KD-trees

O uso de árvores como estrutura para guardar dados não é nada de novo. São em muitos casos a opção mais preferível, pois permitem organizar os dados de forma eficiente enquanto simultaneamente aceleram o processo de pesquisa. No contexto dos dados vetoriais, existem muitas opções para indexar este tipo de informação, desde as KD-*trees*, *ball-trees*, R-*trees* e M-*trees* (Han et al., 2023). Para efeitos de simplificação, e por serem as mais conhecidas, irão ser exploradas as KD-*trees*.

Uma árvore KD é uma estrutura de dados, sob a forma de uma árvore binária, que armazena um conjunto finito de pontos de um espaço dimensional de tamanho K, em que K é normalmente um número muito grande (Moore, 1991). Cada nodo desta árvore é um ponto de dimensão K e cada nodo que não seja folha atua como um plano de separação que divide o espaço em duas partes, sendo perpendicular ao eixo escolhido de uma das K dimensões (Han et al., 2023).

De acordo com Moore, um nodo de uma árvore KD pode ser especificado de acordo com a Tabela 2.

Campo	**Tipo de dado**	**Descrição**
dom-elt	*array*	um ponto do espaço vetorial
range-elt	*array*	um ponto do espaço vetorial
split	inteiro	a dimensão onde ocorre a divisão
left	kd-*tree*	árvore com pontos à esquerda do plano de divisão
right	kd-*tree*	árvore com pontos à direita do plano de divisão

Tabela 2: Nodo da árvore KD

Nesta especificação, o campo *dom-elt* representa o vetor de domínio e atua principalmente como o índice (ID) de um nodo. O *range-elt* é o vetor de raio desse nodo. O nodo pode ter até duas ramificações, que resultam da introdução de um plano de divisão. Este plano pode ser aplicado em todas as K dimensões, pelo que o valor *split* deve especificar em qual delas a divisão vai ocorrer. O resultado da divisão são duas novas árvores KD, *left* e *right*, uma com os

pontos do sub-espaço à esquerda do plano e outra com os pontos do sub-espaço à direita do plano (Moore, 1991).

Durante a pesquisa pelo vizinho mais próximo, é utilizada uma estrutura de *queue*, que essencialmente consiste numa fila, neste caso uma fila com os nós prioritários a visitar de acordo com a sua distância ao ponto da **query**. Durante a sua execução, é retirado o ponto com menor valor de distância da fila, verificando se se trata de uma folha ou não. Caso seja folha, trata de comparar a proximidade entre o ponto da **query** e o vetor de dados no nodo, atualizando, se se justificar, o valor atual da melhor distância e o nodo que lhe corresponde. Caso não seja folha, apenas empurra para a fila as árvores da esquerda e direita (Han et al., 2023).

O facto de ser conceptualmente mais simples e mais fácil de implementar quando comparada com outras estruturas de árvores, faz com que estas sejam as principais vantagens de usar árvores KD. O desempenho do algoritmo de pesquisa neste tipo de árvores varia muito com a dimensão K do espaço, o número de pontos e a distribuição dos mesmos (Han et al., 2023).

Locality-Sensitive Hashing

O *Locality-Sensitive Hashing* (**LSH**) consiste numa técnica de indexação introduzida em 1998 que, na sua essência, serve para mapear itens semelhantes para o mesmo código *hash* com maior probabilidade do que os itens diferentes (Wang et al., 2014). Este algoritmo de indexação faz uso de uma família de funções de **hashing**, nas quais são empregues projeções aleatórias que sejam localmente sensíveis, permitindo assim que o mapeamento dos dados vetoriais seja realizado como foi descrito no parágrafo anterior. O mapeamento pretende satisfazer a seguinte expressão:

$$P(h(p) = h(q)) = f(d(p, q)) \tag{3.1}$$

na qual **h** é uma função de **hashing**, **p** e **q** são pontos, **f** é uma função que mede a similaridade (por exemplo a similaridade do cosseno), e **d** a distância entre os pontos. No fundo, a propriedade expressa que a probabilidade de h(p) ser igual a h(q) é igual à similaridade entre

eles, ou seja, quanto mais similares forem dois vetores de dados, maior é a probabilidade de colisão entre eles ou de serem o mesmo dado (Han et al., 2023).

Hierarchical Navigable Small World

O Hierarchical Navigable Small World (**HNSW**) consiste na utilização de grafos para a indexação dos vetores. Nesta abordagem, o espaço vetorial é representado como um grafo, que por sua vez possui várias camadas. A camada mais baixa contem todos os vetores do espaço e as camadas sobrejacentes agrupam os vetores pela proximidade entre eles. Isto significa que, conforme se vai subindo pelas camadas do grafo, o número de vetores é menor e a similaridade entre eles é maior (Cantarero, 2023). A camada de topo é o local onde se encontra o ponto de entrada, que é o ponto onde a pesquisa é iniciada quando é realizada uma **query**. Este ponto é escolhido pelo algoritmo com base na sua proximidade ao ponto de **query**. Conforme a pesquisa avança para camadas inferiores, o algoritmo trata de recolher um conjunto de vetores candidatos que estão mais próximos do vetor **query** (Tai, 2024). Em cada camada, são avaliadas as distâncias entre o vetor **query** e os vetores candidatos, o que poderá substituir alguns dos vetores candidatos caso sejam encontrados vetores que sejam mais semelhantes. A pesquisa termina quando uma exploração mais aprofundada não consegue encontrar vetores mais próximos (Tai, 2024).

3.3.2 Pesquisa e Cálculo de Similaridade

Quando se usa uma base de dados relacional para executar uma **query**, devem ser concretamente especificadas as tabelas, os identificadores e as relações entre os dados, de forma a obter um resultado. No entanto, quando se usa uma base de dados vetorial, a execução é mais simples, sendo apenas necessário usar um ou mais vetores de **query** (Taipalus, 2024). Assumindo que se usou um dos métodos de indexação listados na secção anterior, a pesquisa em bases de dados vetoriais irá consistir na procura pelo vizinho mais próximo - Nearest Neighbor Search (**NNH**) - na estrutura de indexação escolhida. No exemplo dado por Taipalus, assumindo que se tem uma base de dados vetorial onde os dados são peças de teatro gregas, se a procura for por peças que tenham elevado grau de tragédia, a pesquisa por **NNH** devolveria peças gregas que sejam similares a uma determinada peça em termos de tragédia, comédia ou drama. Quando aplicado às diferentes abordagens de indexação, o **NNH** irá ter valores de complexidade diferentes, pois cada abordagem usa estruturas diferentes para o armazenamento dos vetores. Por exemplo, o **NNH**, quando aplicado a uma estrutura de árvore KD, normalmente resulta num tempo de

execução de O(log N) (Moore, 1991). Na abordagem por **LSH**, que utiliza tabelas de **hashing**, a pesquisa pelo vizinho mais próximo possui uma complexidade temporal que varia com o número de vetores de dados, N, a dimensão desses vetores, D, e o número de tabelas usadas, K, resultando numa complexidade temporal de O(N*D+K) (Otten, 2023). Finalmente, para o método de indexação **HNSW**, a complexidade temporal, ao percorrer a estrutura durante uma **query**, é de O(log N) (Elliott, 2024).

Caso não se tenha usado um método de indexação, isto é, os vetores estão guardados numa estrutura não ordenada, é possível utilizar um algoritmo de "força-bruta" para a pesquisa (Cantarero, 2023). Esta técnica descarta qualquer tipo de estratégia de filtragem inicial, essencialmente percorrendo todos os vetores guardados na base de dados, analisando cada um, de forma a obter precisão total no resultado obtido. Como é de prever, este algoritmo tem um alto custo computacional, que é proporcional ao tamanho do conjunto de dados explorado, atingindo uma complexidade O(N) (Han et al., 2023). É apenas ideal para cenários de teste e exploração, onde a precisão é a prioridade e a velocidade é pouco relevante.

A Tabela 3 apresenta todos os valores de execução, em notação *Big-O*, para a pesquisa por similaridade em cada uma das estruturas de indexação abordados neste documento.

Algoritmo Indexação	**Complexidade-Tempo na Pesquisa**
KD-*tree*	O(log N)
LSH	O(N*D+K)
HNSW	O(log N)
"Força Bruta"	O(N)

Tabela 3: Valores de Complexidade NNS

Com base na Tabela 3, é seguro afirmar que, entre todos os algoritmos analisados até ao momento, os algoritmos KD-*tree* e **HNSW** são os mais eficientes quando a escalabilidade do sistema é a principal prioridade. Isto deve-se ao facto de possuírem complexidade logarítmica ao invés de linear (Olawanle, 2022).

De forma a determinar quais dos vetores indexados é próximo à **query**, o algoritmo **NNH** precisa de saber calcular a proximidade entre vetores. Para isso pode fazer uso de diversas fórmulas matemáticas que servem precisamente para medir a distância ou similaridade entre dois pontos de dados (Elastic Plat-

form Team, 2024). A escolha da fórmula fica a cargo de quem implementa o algoritmo. Existem várias abordagens, mas irão ser exploradas duas das mais utilizadas: a similaridade do cosseno e a distância de Minkowski.

A similaridade por cosseno consiste em avaliar o ângulo formado pelos dois vetores, segundo a seguinte expressão:

$$CosineSim(\vec{A}, \vec{B}) = cos(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \times \|\vec{B}\|} \tag{3.2}$$

Tomando como exemplo os seguintes vetores, que foram gerados aleatoriamente, para representar produtos e uma **query**, pode-se calcular quais dos produtos são mais similares à **query**.

Produto 1: [0.8, 0.4, 0.7, 0.9]

Produto 2: [0.6, 0.3, 0.2, 0.4]

Produto 3: [0.9, 0.5, 0.8, 0.6]

Produto 4: [0.2, 0.1, 0.3, 0.2]

Vetor query: [0.85, 0.45, 0.75, 0.85]

Aplicando a fórmula a cada produto, onde $\vec{A}$ é o vetor **query** e $\vec{B}$ o produto a ser comparado, obtiveram-se os seguintes valores:

$CosineSim_{\text{Produto 1}}$ = 0.9980

$CosineSim_{\text{Produto 2}}$ = 0.9470

$CosineSim_{\text{Produto 3}}$ = 0.9842

$CosineSim_{\text{Produto 4}}$ = 0.9672

Neste exemplo, o produto 1 seria o mais semelhante à pesquisa realizada, enquanto o produto 2 seria o mais distante. Se estes vetores estivessem indexados em **HNSW** numa base de dados vetorial, todos os vetores de produto poderiam ser considerados como bons candidatos para a pesquisa (Tai, 2024).

Por outro lado, usando a distância de Minkowski, é possível ajustar os parâmetros da fórmula de forma a mudar o tipo de distância (Euclidiana ou Manhattan) que se vai usar para calcular o valor de proximidade entre os pontos (Chugani, 2024). Nesta medida, ao contrário da anterior, quanto menor for o resultado, maior é a similaridade. A distância de Minkowski é calculada pela seguinte expressão:

$$Minkowski(\vec{A}, \vec{B}) = \left(\sum_{i=1}^{n} \left| \vec{A}_i - \vec{B}_i \right|^p \right)^{\frac{1}{p}} \tag{3.3}$$

na qual $\vec{A}$ e $\vec{B}$ são vetores, **i** é a dimensão e **p** determina qual das distâncias vai ser usada. Se **p** for 1, será usada a Manhattan, se for 2 será usada a euclidiana. Além disso, também é realizado um somatório das diferenças dos valores de cada dimensão entre os dois vetores.

Usando o mesmo exemplo dos quatro vetores de produtos e o mesmo vetor de **query**, a distância de Minkowski com **p** igual a 2 para o produto 1 é a seguinte:

$$\begin{aligned} & DistMink_{\text{Produto 1}} \\ &= \sqrt{(0.85 - 0.80)^2 + (0.45 - 0.40)^2 + (0.75 - 0.70)^2 + (0.85 - 0.90)^2} \\ &= 0.1 \end{aligned}$$

Para os restantes produtos os valores obtidos foram:

$DistMink_{\text{Produto 2}}$ = 0.7680

$DistMink_{\text{Produto 3}}$ = 0.2640

$DistMink_{\text{Produto 4}}$ = 1.0821

As conclusões são ligeiramente parecidas. O produto 1 continua a ser o mais similar, mas o 4 passou a ser o mais distante do vetor **query**, ou seja, o menos semelhante. O produto 1 é o mais semelhante em ambos os casos porque os seus valores são próximos ao vetor **query** tanto em direção quanto em magnitude. O produto 4 é mais distante na distância de Minkowski porque os seus valores absolutos diferem mais do vetor **query**, mas é razoavelmente semelhante na similaridade do cosseno porque a sua orientação ainda é próxima.

Esta diferença de resultados entre as duas abordagens deve-se principalmente a dois fatores: a similaridade do cosseno foca-se mais no ângulo entre os dois vetores, sendo independente da magnitude deles (Taipalus, 2024), isto é, não importa se os vetores são grandes ou pequenos, o foco está na orientação relativa; a distância de Minkowski mede a diferença absoluta entre os valores das dimensões de dois vetores tendo em consideração a magnitude dos vetores (Taipalus, 2024), sendo por isso sensível a diferenças em escalas. Determinar qual delas é mais precisa vai acabar por depender do contexto do problema e da decisão do programador, mas vale a pena mencionar que foi revelado em alguns estudos que a similaridade do cosseno é mais propicia a dar falsas indicações de similaridade quando comparado com a distância de Minkowski (Al-Shamri, 2022).

3.4 Sistemas e Aplicações

A vasta panóplia de exemplos reais de sistemas de sistemas de gestão de bases de dados vetoriais atualmente a serem introduzidos no mercado com sucesso constitui uma das principais motivações para a realização deste trabalho de dissertação. Nesta secção vão-se explorar alguns desses sistemas e os seus casos de aplicação mais conhecidos. Vejamos alguns deles.

LanceDB. O LanceDB é um sistema de gestão de bases de dados de código aberto que foi projetado para aplicações que trabalham com **IA** num contexto multimodal e de larga escala, permitindo o armazenamento, gestão e consulta de **embeddings** e dados multimodais (LanceDB, n.d.). Este sistema de gestão de bases de dados tem como principal vantagem a capacidade de poder ser instanciado de diversas formas, o que facilita a sua integração em aplicações já existentes. Por exemplo, a instanciação pode ser feita diretamente no **back-end** de uma aplicação em Flask ou Node.js, permitindo assim que o acesso aos dados seja direto, ou então pode ser instanciado numa aplicação de cliente, como o Jupyter Notebook, tornando-o ideal para projetos de **data science** e

machine learning (Restack, 2025). A eficácia e utilidade deste sistema de gestão de bases de dados é elogiada por várias personalidades do mundo empresarial, sendo uma delas o diretor de tecnologia (**CTO**) da TubiTV, Marios Assiotis, que o elogiou pela sua adaptabilidade às tecnologias que já estavam a ser empregues pela Tubi, referindo também a simplicidade e facilidade na gestão dos **embeddings** e vetores dos dados (LanceDB, n.d.).

ChromaDB. Uma das opções mais populares no domínio das bases de dados vetoriais é o ChromaDB. Este sistema de gestão de bases de dados contrasta com o LanceDB em alguns pontos cruciais, sendo a principal diferença a questão do formato de armazenamento utilizado, onde o LanceDB utiliza o formato colunar Lance, enquanto o ChromaDB usa um formato de armazenamento baseado em ficheiros simples, priorizando assim a velocidade de acesso em tempo real, limitando a sua escalabilidade (Restack, 2025). Isto significa que o ChromaDB é ideal para cenários onde a baixa latência é fundamental. No entanto, à medida que o conjunto de dados cresce, o desempenho pode diminuir devido a restrições de memória (Restack, 2025). O ChromaDB também abstrai todo o processo de **embedding**, indexação de vetores e cálculo de similaridade, o que permite aos programadores focarem-se apenas nos dados e nas **queries** (Suard, 2024).

Oracle AI Vector Search. A Oracle é uma gigante da tecnologia, fundada em 1977, que oferece soluções nas áreas da computação em nuvem, aplicações empresariais como **ERP** e **CRM** e ferramentas de inteligência artificial. O seu principal foco é a gestão de dados, fornecendo soluções para **big data** e análise de dados (Hayes et al., 2024). Em 2024, a Oracle introduziu o sistema **AI** Vector Search. Este sistema surge com o objetivo de fornecer uma panóplia de ferramentas que, quando combinadas, permitem interagir com grandes modelos de linguagem (**LLM**) e implementar **pipelines** completos para processos de **RAG** (Chiappara, 2024). A característica mais forte deste sistema é a capacidade de conseguir executar tarefas como a geração dos **embeddings**, indexação ou divisão de dados internamente, isto é, dentro da própria base de dados, através de processos próprios. No entanto, caso o programador deseje, existe a possibilidade destes processos serem feitos externamente, tendo por isso maior flexibilidade que outras bases de dados vetoriais (Chiappara, 2024). A maior novidade introduzida pela Oracle neste sistema foi o tipo de dados denominado VECTOR, que foi concebido especificamente para armazenar representações vetoriais de dados de forma eficiente (Chiappara, 2024). O VECTOR é inserido numa tabela, com uma coluna dedicada. Um exemplo de como se pode criar uma tabela em Oracle com dados vetoriais pode ser analisado na Figura 3.

```
CREATE TABLE vetores (
  id NUMBER,
  embedding VECTOR(768, INT8)
);
```

vetores	
id	embedding
1	(0.345; 0,564; 0,73)
...	...

Figura 3: Geração de tabela com vetores em Oracle

As bases de dados vetoriais analisadas oferecem abordagens distintas consoante os requisitos da aplicação, variando em termos de escalabilidade, desempenho e integração. A escolha da solução mais adequada dependerá, portanto, do contexto de utilização, das exigências em tempo real, da complexidade dos dados manipulados e da preferência do utilizador.

Capítulo 4

Customer Profiling

4.1 Noção e Utilidade

Customer profiling é um processo concebido para criar representações detalhadas de um conjunto de clientes com base em características comportamentais, demográficas e transacionais, agrupando-os em grupos por grau de semelhança, o que permite compreender melhor quem são os clientes de uma determinada empresa, o que os motiva e como interagem com um determinado produto ou serviço (Mahalakshmi et al., 2020). Além disso, também possibilita às empresas segmentar, personalizar e otimizar as suas estratégias de marketing e vendas, tornando-se também mais amigáveis ao consumidor (Whittle et al., 1989).

As técnicas de *profiling* de clientes variam em termos de complexidade e esforço aplicado. Por exemplo, antes da introdução dos modelos de aprendizagem automática, as abordagens utilizadas exigiam mais esforço, pois consistiam em processos manuais e personalizados, nos quais os profissionais analisavam diretamente o comportamento e os dados dos clientes para construir perfis detalhados (Softweb Solutions - An Avnet Company, 2025), podendo demorar entre semanas a meses a concluir o estudo. Por outro lado, as técnicas automatizadas, baseadas em **machine learning**, permitem processar grandes volumes de dados e criar perfis de clientes de forma eficiente e escalável. Aqui, usam-se frequentemente algoritmos supervisionados e não supervisiona-dos, em que através da deteção de padrões e tendências, anomalias e correlações, se consegue agrupar clientes com comportamentos semelhantes, tornando possível prever as suas preferências atuais e futuras (Softweb Solutions - An Avnet Company, 2025).

Os dados dos clientes podem ser bastante diversos. A sua escolha e utilização dependem do tipo de análise de perfil que se pretende fazer. Os dados podem ser agrupados nas seguintes categorias:

- Dados demográficos, que englobam informação relativa ao género, idade, estatuto civil, etnicidade, rendimentos, emprego e educação (Galic, 2024).

- Dados psicográficos, que são semelhantes aos anteriores, mas a um nível psicológico. Permitem perceber as razões pelas quais os clientes escolhem interagir com um serviço ou comprar um determinado produto (Galic, 2024). Aqui são principalmente agrupados dados sobre o estilo de vida dos clientes, as suas opiniões, valores, crenças e a sua afiliação política. Este tipo de dados é mais subjetivo, sendo por isso mais difícil de identificar (Galic, 2024), mesmo usando técnicas de **machine learning**.

- Dados de comportamento, que englobam as interações dos clientes com um determinado serviço, permitindo agrupar clientes por tendências de comportamento, senda elas os padrões de compra, hábitos de consumo, interações com determinadas marcas, o tipo de uso dado ao produto adquirido e o **feedback** deixado. Este tipo de dados é particularmente útil para o desenvolvimento de sistemas de recomendações personalizadas (Galic, 2024) com base nas preferências dos clientes.

- Dados geográficos, que permitem caraterizar um cliente pela sua posição geográfica, local de trabalho, língua ou cultura.

4.2 Modelos e Técnicas para Profiling

A realização de *profiling de clientes*, idealmente, começa com a recolha de dados (Lee, 2025). Numa ação de *profiling* podemos utilizar vários métodos, tais como: a concretização de inquéritos num ambiente de diálogo e interação, sob a forma de questionários online, entrevistas e discussões em **focus groups**; a observação direta do histórico e do comportamento dos clientes (Traqueia et al., 2021) num determinado serviço ou plataforma; ou a extração ou compra de dados a entidades terceiras (Lee, 2025), sendo a extração realizada através de **web scraping**, que consiste na mineração automática de dados de páginas na Internet e de outras bases de dados comuns (Lotfi et al., 2021).

Após a recolha dos dados, o próximo passo num processo de *profiling* consiste na aplicação de técnicas e modelos que permitam agrupar os clientes em grupos distintos (Lee, 2025). Para tal, utilizam-se técnicas que empregam, na sua grande maioria, algoritmos de **machine learning**, que permitem acelerar a obtenção de resultados. Dos vários algoritmos, destaca-se a segmentação utilizando **k-means**, um método não supervisionado, que permite segmentar dados em grupos distintos, denominados **clusters**, com

base nas semelhanças. Quando aplicado a processos de *profiling*, o algoritmo de **k-means** apresenta como principais vantagens a sua capacidade de agrupar objetos de elevada dimensão e de acelerar significativamente o processo de agrupamento (Al Ayubi et al., 2024). No entanto, a escolha inadequada do número de **clusters** pode comprometer a precisão dos resultados obtidos (Al Ayubi et al., 2024). Através da análise das características de cada **cluster** resultante da aplicação do **k-means**, torna-se possível identificar diferentes padrões de comportamento e preferências entre os vários segmentos de clientes, permitindo traçar o perfil representativo de cada **cluster** e destacar as principais características que definem os clientes que o compõem (Al Ayubi et al., 2024).

Outro modelo de **machine learning** usado em *profiling* é o Support Vector Machine (**SVM**). Este modelo é suportado por um algoritmo supervisionado utilizado para processos de classificação e regressão, que permite fazer o reconhecimento de padrões e o **PLN** (Tibrewal, 2023). Quando aplicado a um cenário de segmentação de clientes, o **SVM** permite obter um híper-plano que separa os clientes em grupos distintos, maximizando a margem entre eles (Venkatesh et al., 2023), o que permite identificar perfis claros e distintos mesmo em conjuntos de dados de alta dimensão. Comparativamente ao algoritmo **k-means**, o **SVM** apresenta um desempenho superior em cenários nos quais existem **clusters** com diferenças significativas de tamanho, revelando-se também mais robusto à presença de ruído e discrepâncias nos dados (Venkatesh et al., 2023). Esta abordagem mostra-se eficaz em contextos de *customer profiling*, particularmente quando a separação entre classes ou perfis de clientes é bem definida. Isso permite uma segmentação dos dados mais precisa e confiável (Venkatesh et al., 2023).

4.3 Sistemas e Aplicações

Os sistemas de *profiling* de clientes encontram-se atualmente integrados em diversas plataformas e organizações, assumindo um papel crucial na personalização de serviços e na otimização da experiência do utilizador. Um exemplo ilustrativo são as plataformas de **streaming**, como a Netflix, que utilizam perfis de clientes para recomendar conteúdos com base no histórico e no tempo de visualização dos utilizadores (Tudor, 2021). Por outro lado, no domínio do retalho online, empresas como a Amazon recorrem à análise de padrões de compra para antecipar necessidades futuras e personalizar promoções dirigidas aos seus clientes (Tudor, 2021). Já no setor das telecomunicações, as operadoras estudam os perfis de consumo dos seus clientes, considerando métricas como o volume de dados e o número de chamadas realizadas, com o objetivo de oferecer planos tarifários ajustados ao comportamento individual (Dwivedi, 2023).

Existem também diversas ferramentas de análise de mercado, tais como o Google Analytics, o Salesforce ou o Tableau (Lee, 2025), que não apenas possibilitam a realização de *customer profiling*, como também disponibilizam um vasto conjunto de estatísticas e métricas analíticas que se revelam fundamentais para a definição de estratégias empresariais mais eficazes e para o aumento da competitividade no mercado.

Capítulo 5

Um Sistema de Profiling

5.1 Área de Aplicação e Características

O *profiling* de clientes pode ser aplicado praticamente em qualquer organização na qual existam interações entre clientes, utilizadores, ou prestadores de serviços, entre outros. A seleção da área de aplicação do sistema proposto baseou-se na avaliação do seu grau de complexidade, no nível de familiaridade já adquirido com os seus conceitos e na relevância de temas como as preferências dos clientes e a recomendação de conteúdo. Assim, no âmbito deste trabalho, escolhemos o setor do **streaming** como a área de aplicação do sistema que pretendíamos desenvolver. Podemos justificar esta opção de diversas maneiras. Em primeiro lugar, trata-se de um setor com elevada intensidade de interação entre utilizadores e serviço (Brown, 2024), o que gera uma grande quantidade de dados comportamentais que, tal como referido anteriormente, são ideais para sistemas de recomendações personalizadas. Em segundo lugar, trata-se de uma área em que a personalização da experiência do utilizador é um elemento central da estratégia de fidelização e competitividade (Brown, 2024), o que torna o *profiling* uma ferramenta particularmente pertinente. Por último, a escolha foi também influenciada pela perceção de que, entre as várias alternativas possíveis, se trata de um domínio com menor complexidade relativa no que respeita à estrutura de dados e aos fluxos de interação entre utilizadores e serviço. Para além disso, a decisão foi igualmente influenciada pelo facto de se já possuir algum conhecimento sobre o funcionamento deste tipo de plataformas, o que facilita a definição de requisitos, a modelação do sistema e a interpretação dos resultados obtidos.

Atendendo à área de aplicação escolhida, o caso de estudo desenvolvido foi construído com base em amostras da Sakila, que se trata de uma base de dados relacional amplamente utilizada para fins didáticos. Dessa base de dados selecionámos especificamente as tabelas “film”, “customer” e “category”. Este cenário, de natureza fictícia, tem como principal objetivo a exploração dos conceitos de *customer*

profiling e de bases de dados vetoriais. Para efeitos de simplificação, foi concebido um cenário com poucas entidades, envolvendo apenas uma empresa de **streaming** de filmes e os seus clientes.

A narrativa considera uma plataforma de **streaming** fictícia, designada "Uncle Broke Movie Catalog", caracterizada por ser de baixo orçamento e de recente criação, com apenas alguns meses de atividade. Dada a sua curta existência no mercado, a empresa dispõe de uma base de clientes reduzida e de um catálogo limitado, composto por um número restrito de filmes, maioritariamente títulos populares de Hollywood. A plataforma dirige-se a um público-alvo com rendimentos mais baixos, procurando oferecer uma solução acessível de entretenimento digital. Atualmente, a infraestrutura tecnológica da empresa encontra-se ainda numa fase inicial, com funcionalidades bastante básicas. Neste contexto, pretende-se desenvolver e integrar um sistema capaz de recomendar filmes aos clientes com base no seu histórico de visualizações. Este sistema deverá não só armazenar toda a informação relativa a clientes e filmes disponíveis na plataforma, como também identificar e associar a cada cliente as categorias de filmes que visualiza com maior frequência.

Recorrendo a uma base de dados vetorial, o sistema proposto tem como objetivo agrupar os clientes em perfis predefinidos, num total de cinco categorias distintas. A cada um destes perfis será recomendada uma seleção de filmes, adequada ao contexto e preferências típicas dos clientes enquadrados nesse perfil. O funcionamento do sistema baseia-se na associação, para cada cliente, dos filmes visualizados e respetivas categorias, sendo esta informação convertida em representações vetoriais. Através de mecanismos de similaridade vetorial, o vetor do cliente é comparado com os vetores representativos de cada perfil, permitindo identificar o grupo ao qual o cliente mais provavelmente pertence. O sistema também terá de realizar recomendações de filmes que sejam adequadas aos diversos perfis.

O sistema de *profiling* foi inicialmente desenvolvido sob a forma de um protótipo, designado MyMovieProfiler (Figura 4). Este protótipo funciona de forma independente da plataforma de **streaming**, conferindo-lhe a flexibilidade necessária para ser testado e validado de forma mais eficiente e controlada. Os aspetos técnicos e funcionais deste protótipo serão explorados em maior detalhe ao longo do presente capítulo.

Figura 4: Logótipo do protótipo

Para o desenvolvimento do protótipo, foram atribuídos dois conjuntos de dados (**datasets**): um contendo a informação relativa a cem clientes, e outro com sessenta e quatro filmes, incluindo as suas respetivas características. O conteúdo armazenado na base de dados vetorial do protótipo deverá corresponder idealmente à combinação destes dois conjuntos, associando os clientes aos filmes e às categorias correspondentes, de modo a construir perfis que integrem dados de natureza demográfica e comportamental.

Relativamente ao **dataset** "clientes" (Figura 5), as informações consideradas mais relevantes incluem o género, a idade e a lista de filmes assistidos. Cada cliente está identificado por um identificador único, o ID, que tem como única função garantir a unicidade dos registos, particularmente nos casos em que dois clientes apresentem exatamente os mesmos dados. Este identificador, porém, não possui qualquer influência no processo de *profiling*.

ID	Gender	Age	Movies Watched
...	---	---	---
9	Male	24	Forrest Gump, La La Land, The Green Mile
10	Female	30	Jurassic Park, The Exorcist, Coco, The Revenant
11	Male	74	Beauty and the Beast, The Greatest Showman
...	---	---	---

Figura 5: Um extrato do *dataset* "clientes"

O **dataset** relativo aos filmes (Figura 6) inclui quatro atributos principais: o nome do filme, que funciona também como identificador único; a categoria, que pode assumir um ou mais valores, correspondentes aos diferentes géneros; o ano de lançamento; e a duração, expressa em minutos.

Name	Category	Year of Launch	Duration (min)
...	---	---	---
The Godfather	drama, action	1972	175
Titanic	romantic, drama	1997	195
Toy Story	animation, children's	1995	81
...	---	---	---

Figura 6: Um extrato do *dataset* "filmes"

A base de dados vetorial que suporta este sistema irá conter duas coleções distintas: uma com os filmes do sistema e outra com os clientes, sendo que os clientes serão acompanhados de informação adicional relativa aos filmes que viram.

O protótipo teve de satisfazer um conjunto de requisitos funcionais fundamentais, para que fosse assegurada a sua utilidade e aplicação prática. Em primeiro lugar, o sistema tem de ser capaz de armazenar na base de dados, em formato vetorial, informação relativa tanto aos clientes como aos filmes. Deve igualmente possuir a capacidade de processar uma coleção de clientes e, a partir dessa análise, construir perfis distintos que sirvam de base ao processo de *customer profiling*.

Adicionalmente, o sistema deve permitir a realização de *profiling* para novos clientes, atribuindo-os ao perfil mais adequado com base nas suas características e preferências. Após esta atribuição, o sistema tem ainda de ser capaz de recomendar um conjunto de filmes que se enquadrem no perfil correspondente.

No que respeita à interação com o utilizador, este deverá ter a possibilidade de selecionar qual a coleção de clientes a ser utilizada para efeitos de *profiling*, bem como criar coleções ou eliminar aquelas que já não sejam relevantes. O sistema deve igualmente permitir a consulta dos elementos que integram as coleções de clientes e de filmes, assim como a visualização das coleções existentes na base de dados.

5.2 O Sistema Desenvolvido

O protótipo desenvolvido assenta numa base de dados vetorial, contendo algumas funcionalidades adicionais que permitem fazer a segmentação e classificação de clientes.

5.2.1 Organização e Arquitetura

A arquitetura do sistema assenta numa estrutura modular, promovendo a separação de responsabilidades entre os seus componentes. Contudo, atendendo à natureza simplificada do protótipo, não foram adotadas convenções formais de arquitetura de sistemas. A divisão dos módulos é a seguinte:

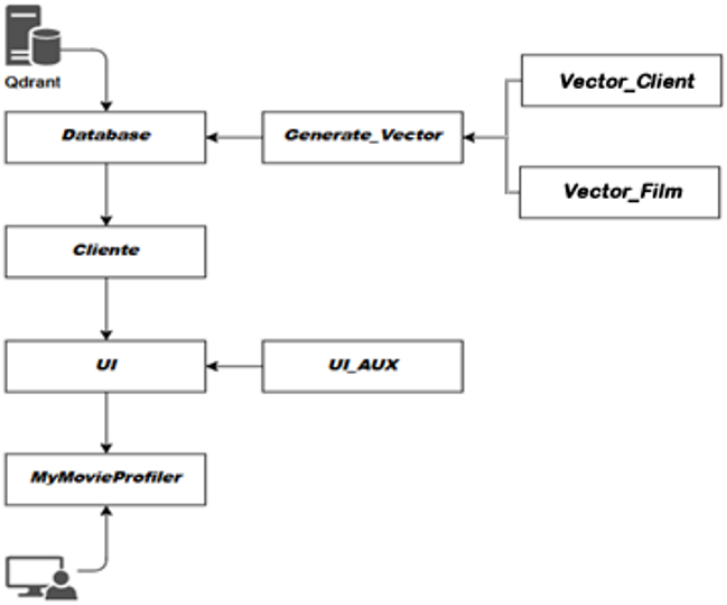


Figura 7: Arquitetura do protótipo desenvolvido

O módulo "MyMovieProfiler" tem como principal função o arranque do sistema, sendo, por conseguinte, de estrutura simples e dimensão reduzida no que respeita ao seu conteúdo funcional, contendo apenas uma função que arranca a interface e a base de dados. A interação com o protótipo é realizada através de uma interface gráfica elementar, desenvolvida no módulo de interface do utilizador, recorrendo-se à biblioteca Tkinter. Este componente é responsável pela geração das diversas janelas da aplicação e pela gestão de todos os inputs do utilizador, encaminhando as respetivas solicitações para o componente imediatamente superior na hierarquia. A UI recorre ainda a um módulo auxiliar designado UI-AUX, o qual agrega funções de uso recorrente no contexto da interface, promovendo assim a reutilização de código e a redução da redundância.

O módulo "Cliente" atua como um intermediário, pois a sua lógica concentra-se no encaminhamento das requisições provenientes da UI para o módulo responsável pela gestão da base de dados e pela lógica associada à pesquisa vetorial, designado por "Database". Adicionalmente, dispõe da capacidade de ve-

rificar a disponibilidade do servidor que aloja a instância da base de dados, sendo que, neste caso, o servidor é local.

O módulo de maior relevo é o "Database", visto que engloba toda a lógica associada à gestão dos dados, à pesquisa vetorial e ao *profiling* dos clientes. Durante a sua inicialização, este componente cria uma instância de cliente do Qdrant, o que lhe permite comunicar diretamente com o servidor local da base de dados, podendo assim executar todas as operações relacionadas com a inserção, remoção e pesquisa de dados, através de uma **API** dedicada. Por intermédio dessa **API**, é possível proceder à criação das coleções vetoriais que irão compor a base de dados. Para cada coleção, é necessário definir previamente a dimensão dos vetores e a métrica de similaridade a utilizar nas operações de pesquisa, sendo que, no contexto deste sistema, foi adotada a métrica de similaridade do cosseno. A cada coleção devem ser posteriormente atribuídos os vetores correspondentes, cuja geração é realizada num módulo separado. É também neste módulo que se encontram implementados os métodos para *profiling* e segmentação dos clientes. Considerando que estes métodos recorrem à utilização do algoritmo **k-means**, o referido módulo inclui também um método específico para a análise dos **clusters** gerados, com o intuito de extrair estatísticas relevantes sobre os perfis de clientes presentes em cada grupo. A secção dedicada aos serviços apresenta, de forma mais detalhada, a descrição completa do processo de *profiling*, bem como dos algoritmos aplicados.

Para a geração dos vetores, o módulo "Database" recorre ao módulo "Generate Vector", o qual utiliza as classes "Vector Client" e "Vector Film". Em "Generate Vector" executa-se toda a lógica associada à geração de **embeddings** para os clientes e filmes, sendo que no caso dos clientes, o vetor obtido contém dezassete campos, sendo eles: o género, com valor binário, sendo 1 para masculino e 2 para feminino; a idade, cujo valor no índice é exatamente igual; a média dos anos de lançamento dos filmes que viu; e a média das durações. Os treze campos restantes do vetor correspondem ao número de ocorrências das diferentes categorias de filmes. Na Figura 8 podemos ver os campos do vetor, que correspondem à classe "Vector Client".

Figura 8: Estrutura de *Vector Client*

Neste contexto, o vetor (1, 32.0, 2, 2, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2012.0, 102.33) representa um cliente do sexo masculino, com trinta e dois anos de idade, que viu dois filmes de terror, dois de suspense (*thriller*), um de drama e um de romance. Esse cliente demonstra preferência por filmes lançados por volta do ano de 2012, com duração média de 102 minutos, ou seja, aproximadamente uma hora e quarenta e dois minutos.

O preenchimento de cada um dos campos, incluindo o cálculo das médias referentes aos dois últimos índices, é realizado através de métodos próprios da classe "Vector Client". Aí, estabelece-se que os primeiros quinze campos devem conter exclusivamente valores inteiros, enquanto os dois últimos são representados por valores de ponto flutuante, devidamente arredondados a duas casas decimais.

No que respeita à classe "Vector Film", responsável pela representação vetorial dos filmes armazenados na base de dados, a sua estrutura é conceptualmente semelhante à da classe "Vector Client", com a distinção de que os campos relativos ao género e à idade são omitidos. Como consequência, a dimensão dos vetores é reduzida de dezassete para quinze. Os primeiros treze campos do vetor correspondem à presença, ou ausência, das diferentes categorias atribuídas ao filme, enquanto os dois campos finais dizem respeito ao ano de lançamento e à duração do filme em minutos. Estes vetores são fundamentalmente utilizados no processo de recomendação, posterior ao *profiling* do cliente, permitindo estabelecer a correspondência entre os interesses do utilizador e os conteúdos disponíveis. A estrutura definida na classe "Vector Film" está apresentada na Figura 9.

Figura 9: Estrutura de *Vector Film*

De acordo com a estrutura definida, o vetor (0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1997, 195) representa o filme "Titanic", lançado em 1997, com 195 minutos de duração, inserido nas categorias de romance e drama.

5.2.2 Serviços Implementados

O protótipo desenvolvido disponibiliza uma seleção de serviços que, no contexto do problema em estudo, cumprem os requisitos definidos. De seguida, descrevem-se, de forma mais detalhada, as funcionalidades de cada um desses serviços:

Profiling da coleção de clientes. Este é o serviço de maior relevância. Quando existe uma coleção ativa de clientes na base de dados, este serviço permite aplicar um algoritmo de segmentação que distribui os clientes entre cinco perfis distintos. Foram testadas três abordagens para a realização dessa segmentação. A primeira, denominada "Profiling HNSW", utiliza o algoritmo **HNSW** para fazer a indexação dos vetores, em conjunto com a **API** do Qdrant, com o objetivo de realizar pesquisas na base de dados e retornar, para um vetor de consulta, o conjunto de clientes mais semelhantes. Neste caso, o vetor de consulta corresponde à representação de um dos cinco perfis previamente definidos. Esses perfis foram declarados diretamente no código-fonte, ou **hard-coded**, resultando em cinco vetores de consulta estáticos. Devido a limitações da **API** do Qdrant, esta abordagem requereu um esforço adicional de codificação para as-segurar que um cliente não fosse atribuído a mais de um perfil. A segunda abordagem, denominada "Profiling Brute Force", consiste em recolher todos os clientes da base de dados em formato vetorial e aplicar uma estratégia de força bruta, visando uma maior precisão na segmentação. Esta abordagem percorre todos os vetores de clientes e calcula a similaridade com cada um dos perfis, associando cada cliente ao perfil com o qual possui maior grau de similaridade. Assim como na abordagem baseada em **HNSW**, os perfis utilizados são definidos diretamente no código. Comparativamente à abordagem anterior, o custo computacional é maior, com um resultado pouco satisfatório. Por fim, a terceira abordagem, denominada "Profiling K-Means", baseia-se na aplicação do algoritmo de **k-means** para agrupar os clientes em cinco **clusters** distintos. Diferentemente das abordagens anteriores, esta não utiliza perfis pré-definidos codificados manualmente. Em vez disso, os perfis são gerados automaticamente a partir dos **clusters** formados pelo algoritmo. A partir desses **clusters**, é possível extrair informações relevantes que permitem caracterizar o tipo de cliente que compõe cada **cluster**, bem como retirar o vetor centroide, que pode ser utilizado para representar o perfil em formato vetorial. Das três abordagens mencionadas, a estratégia baseada no algoritmo **k-means** foi a selecionada para realizar o *profiling* da coleção de clientes. A escolha fundamenta-se, principalmente, no facto do algoritmo operar diretamente sobre vetores para realizar a segmentação, além de não requerer a implementação de passos adicionais para evitar a atribuição repetida de

um mesmo cliente a múltiplos perfis. O funcionamento geral deste serviço, com a abordagem por **k-means**, segue o esquema apresentado na Figura 10.

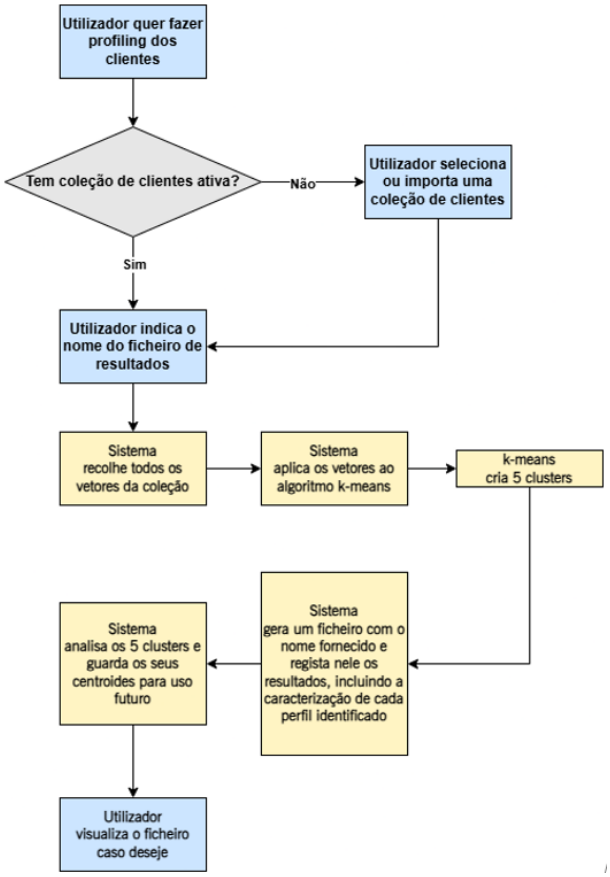


Figura 10: *Profiling* da Coleção de Clientes

Os resultados do *profling* são comunicados diretamente ao utilizador, por meio da geração automática de um ficheiro de texto contendo o resumo da distribuição obtida.

Profiling de apenas um cliente. Um dos requisitos do sistema consiste na capacidade de, dado um novo cliente, atribuí-lo a um perfil existente, de modo que este possa receber recomendações de filmes alinhadas com as suas preferências. Para tal, o utilizador é solicitado a preencher um breve formulário contendo as informações mais relevantes sobre o cliente, nomeadamente o género, a idade e a lista de filmes previamente vistos. A seguir, o cliente é convertido para o formato "Vector Client", conforme descrito na secção anterior. Atribui-se então o cliente ao perfil mais adequado com base no cálculo da similaridade do cosseno entre o vetor do cliente e os vetores

centroides de cada perfil. O resultado desta comparação é apresentado na interface do utilizador, na qual é possível ver o valor de similaridade para cada perfil bem como a caraterização de cada um. Caso ainda não existam vetores centroides disponíveis, o sistema procede previamente à segmentação da coleção ativa de clientes. Juntamente com os resultados, é também apresentada uma lista de filmes considerados adequados ao perfil com o qual o cliente apresenta maior proximidade. Esta lista é obtida por um método dedicado, que utiliza a biblioteca de funções do Qdrant. O processo consiste em extrair o vetor centroide correspondente ao perfil atribuído ao cliente, proceder à remoção dos campos relativos ao género e à idade, reduzindo a dimensão do vetor de dezassete para quinze campos, e, com base nesse vetor ajustado, realizar uma pesquisa na coleção de filmes da base de dados, identificando os cinco filmes mais próximos com base na similaridade do cosseno. Na Figura 11 podemos ver um esquema da forma como o serviço é realizado.

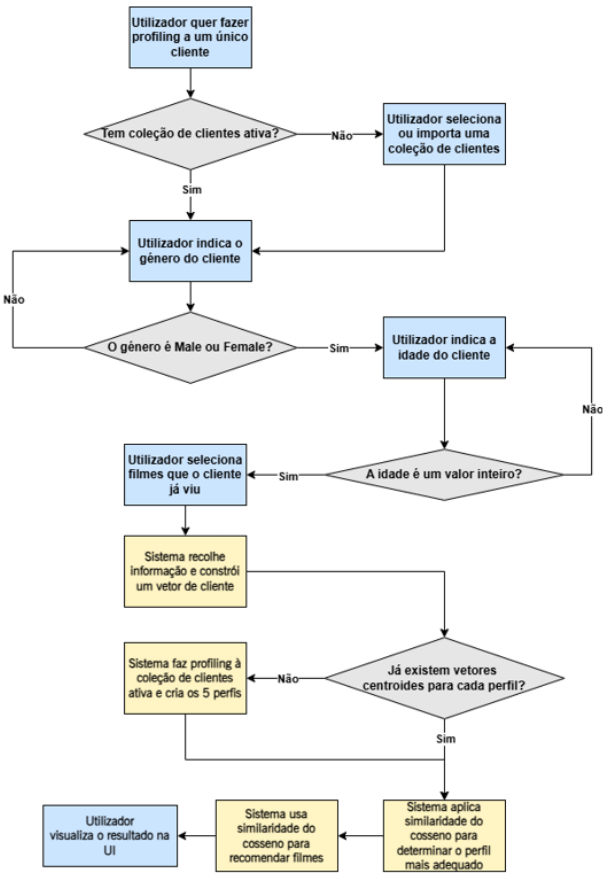


Figura 11: *Profiling* de um Único Cliente

Consultar todos os clientes. Este serviço permite a consulta integral do conteúdo de uma determinada coleção de clientes presente na base de dados. O seu funcionamento está condicio-

nado apenas à existência de uma coleção de clientes previamente selecionada ou importada para o sistema. Em termos funcionais, consiste numa simples chamada ao motor de base de dados, no qual se recupera todos os pontos com vetor e **payload** incluídos.

Consultar todos os filmes. Semelhante ao serviço anterior, este permite a consulta do conteúdo da coleção de filmes presente no sistema, sendo a única condição prévia que a referida coleção se encontre ativa e não tenha sido previamente eliminada.

Alterar a coleção dos clientes. Este serviço permite ao utilizador alterar a coleção de clientes atualmente em uso pelo sistema. Essa alteração pode ser realizada de duas formas: selecionando uma das coleções já existentes na base de dados ou importando um ficheiro no formato JSON para criar uma coleção. Caso a base de dados se encontre vazia, será necessário que o utilizador crie uma coleção para que o sistema possa realizar qualquer tipo de *profiling*. A formatação do ficheiro JSON deve ser rigorosamente respeitada para garantir a correta importação dos dados. De seguida, apresenta-se um exemplo válido de um ficheiro JSON de clientes compatível com o sistema:

```
[ {
    "ID": 1,
    "Gender": "Male",
    "Age": 73,
    "Movies Watched": "The Grand Budapest Hotel"
},
{
    "ID": 2,
    "Gender": "Male",
    "Age": 77,
    "Movies Watched": "Deadpool, Moana, A Star is Born, It"
} ]
```

Em termos funcionais, o processo de importação consiste na leitura do ficheiro em formato JSON e na sua conversão para um dicionário em Python, que se trata de uma estrutura de dados que permite associar uma chave única a um determinado valor. Este dicionário é posteriormente submetido ao módulo “Generate Vector”, que, em conjunto com um dicionário adicional contendo a informação relativa aos filmes, procede à criação dos vetores no formato ”Vector Client”. Os vetores gerados são então devolvidos ao módulo “Database” e inseridos na coleção respetiva na base de dados vetorial.

Mostrar todas as coleções de clientes na base de dados. Trata-se de um serviço simples que permite ao utilizador, por meio de uma função disponibilizada pela **API** do Qdrant, consultar todas as coleções existentes na base de dados, sejam elas de clientes ou de filmes. Não são mostrados os itens das coleções, apenas os seus nomes.

Eliminar uma coleção. O último serviço do sistema consiste na eliminação completa de uma coleção de clientes da base de dados. O sistema verifica previamente se a coleção a ser eliminada existe de facto e, em caso afirmativo, procede à remoção total de todos os seus registos, eliminando qualquer vestígio da sua existência. Também é permitido eliminar a coleção de filmes, no entanto, por ser integral para o funcionamento do sistema, é pedida uma palavra-chave.

5.2.3 Ferramentas de Desenvolvimento

O sistema aqui exposto utiliza uma base de dados vetorial e é capaz de realizar processos de *profiling* de clientes. Para atingir esse fim, procedeu-se à escolha de utensílios adequados à sua implementação. Procurou-se assim utilizar ferramentas bem conhecidas, bem documentadas e acima de tudo estáveis.

A base de dados constitui o pilar do sistema que foi desenvolvido, pelo que a escolha do motor que gere os dados é de extrema importância. Existem vários motores de base de dados que suportam dados vetoriais de alta dimensão, pelo que a escolha foi baseada na facilidade de compreensão e utilização do motor, a facilidade da sua instalação, o preço da utilização do serviço e a compatibilidade com o sistema operativo Windows. Também se procurou dar preferência aos motores que utilizem exclusivamente o modelo vetorial ao invés de modelos híbridos.

Consultando um *ranking* global de bases de dados vetoriais disponível no sítio DB-Engines, foi possível perceber quais são as mais populares, bem como aquelas que estão a cair em desuso. Nas primeiras posições, três motores foram destacados, sendo que alguns já foram mencionados neste documento, sendo eles o ChromaDB, o Milvus e o Qdrant (DB-Engines Ranking, n.d.). Optou-se por escolher este último. Estando bem colocado na lista referida, e cada vez mais popular (DB-Engines Ranking, n.d.), o Qdrant possui também um serviço gratuito e está bem documentado, com uma rede de suporte bastante boa. A sua fácil integração em diversas linguagens de programação, tais como Python, Java e Go foi também um fator decisivo.

O Qdrant como ferramenta possui caraterísticas que facilitam a implementação do sistema proposto, desde a capacidade de suportar vetores de altas dimensões, o facto de já trazer incorporado na sua **API** métodos para calcular a distância e similaridade entre os pontos e para realizar a pesquisa na estrutura de indexação. Outra vantagem do Qdrant é que utiliza como forma de indexação um grafo multi camada, mais precisamente o **HNSW** (Qdrant, 2001), já referido neste documento como sendo uma das melhores estruturas de indexação para pesquisa vetorial. Relativamente à arquitetura do Qdrant, esta pode ser interpretada como uma concretização de um modelo arquitetónico genérico comum às bases de dados vetoriais modernas, semelhante ao que foi descrito anteriormente, no terceiro capítulo desta dissertação. A estrutura inclui componentes fundamentais como as coleções de vetores, os mecanismos de indexação aproximada em **HNSW**, o suporte a **meta dados** estruturados, sobre a forma de **payloads**, e uma camada de **API** para integração com aplicações externas.

Quanto à escolha da linguagem de programação, utilizou-se os seguintes critérios: a quantidade de documentação prontamente disponível, bibliotecas existentes, a facilidade de uso e de leitura, e a adequação ao motor de base de dados escolhido. Para isso, foram consideradas linguagens que priorizam a orientação a objetos e a modulação, tais como o C#, o Java e o Python, sendo que o último acabou por ser o escolhido, devido ao facto de ter uma sintaxe mais simples e limpa, e ser pouco verboso. Para a sua escolha também contribuiu a facilidade de compilação e execução do código e a já mencionada facilidade de integração da **API** do Qdrant nesta linguagem.

Relativamente às bibliotecas mais usadas, para além da **API** do Qdrant, utilizou-se o Tkinter para a construção da interface gráfica, uma biblioteca de Regex para comparação de expressões regulares, quando necessário, e modelos de Sentence Transformer para criar **embeddings**, isto é representações vetoriais, a partir de dados textuais.

5.3 Avaliação do Sistema

A avaliação da qualidade e do desempenho do sistema desenvolvido face aos objetivos definidos foi realizada através da realização de testes específicos sobre os seus métodos de segmentação e *profiling*, utilizando para tal diversos conjuntos de dados. Para além dos **datasets** originais descritos no caso de estudo apresentado no início deste capítulo, foram também criados alguns conjuntos adicionais com o propósito de reforçar a validade dos testes realizados.

Teste de Profiling de Coleções. O primeiro teste centrou-se na capacidade do protótipo em realizar o processo de *profiling* sobre diferentes coleções de clientes armazenadas na base de dados. Importa referir que a coleção de filmes utilizada neste processo se manteve constante, contendo um total de sessenta e quatro filmes com características distintas. Numa primeira experiência, foi utilizada uma coleção composta por cem clientes, com o objetivo de os segmentar em cinco perfis distintos, numerados de 0 a 4. O sistema executou a segmentação conforme esperado, gerando um ficheiro de texto com os resultados obtidos. Neste ficheiro é possível consultar as principais características associadas a cada perfil identificado, permitindo uma análise clara das tendências comportamentais e demográficas dos clientes em cada perfil.

```
These five profiles were obtained with the use of k-means:


--- Basic information about Profile 4 ---
Nr. Clients: 10
Average Age: 58.4
Gender Proportion: {'Male': 4, 'Female': 6}
Main Movie Categories: ['animation', 'thriller', 'action', 'comedy', 'romantic']
Average Year of Launch: 2007.96
Average Movie Duration: 106.67 min

--- Basic information about Profile 1 ---
Nr. Clients: 19
Average Age: 69.95
Gender Proportion: {'Male': 11, 'Female': 8}
Main Movie Categories: ['drama', 'action', 'thriller', 'musical', 'animation']
Average Year of Launch: 2006.26
Average Movie Duration: 133.0 min

--- Basic information about Profile 0 ---
Nr. Clients: 29
Average Age: 41.9
Gender Proportion: {'Female': 13, 'Male': 16}
Main Movie Categories: ['thriller', 'action', 'horror', 'drama', 'animation']
Average Year of Launch: 2008.35
Average Movie Duration: 126.04 min

--- Basic information about Profile 3 ---
Nr. Clients: 30
Average Age: 17.53
Gender Proportion: {'Female': 10, 'Male': 20}
Main Movie Categories: ['thriller', 'action', 'horror', 'animation', 'drama']
Average Year of Launch: 2006.45
Average Movie Duration: 125.63 min

--- Basic information about Profile 2 ---
Nr. Clients: 12
Average Age: 34.0
Gender Proportion: {'Male': 9, 'Female': 3}
Main Movie Categories: ['drama', 'action', 'fantasy', 'musical', 'biopic']
Average Year of Launch: 2004.03
Average Movie Duration: 154.64 min
```

Figura 12: Extrato do *Profiling* de 100 Clientes

A Figura 12 apresenta um extrato do ficheiro de resultados criado pelo sistema. Nessa figura é possível observar que a segmentação por cinco perfis não foi uniforme, sendo o Perfil 3 o mais numeroso e também o mais jovem da amostra, com uma média de idade a rondar apenas os dezassete anos. A maioria dos seus membros são do sexo masculino, cujas preferências centram-se

em géneros como *thriller*, ação, terror, animação e drama. A duração média dos filmes ronda as duas horas, o que, juntamente com o ano médio de lançamento, 2006, indica um gosto por conteúdos de ação rápida e relativamente modernos. Com um total de vinte e nove clientes, o Perfil 0 apresenta características largamente semelhantes às do Perfil 3, distinguindo-se deste, sobretudo, pela diferença na idade média dos seus clientes, pelo ano médio de lançamento dos filmes que estes viram e pela distribuição mais igualitária entre os dois géneros. Os Perfis 1 e 4 revelam-se igualmente bastante semelhantes, apresentando uma proximidade considerável no que respeita à média de idades dos clientes, aos géneros mais apreciados e ao ano médio de lançamento dos filmes visualizados. As diferenças mais marcantes entre ambos residem na preferência por filmes de menor duração no Perfil 4, com uma média próxima de uma hora e quarenta minutos, bem como na valorização de géneros como a comédia e o romance. Importa ainda referir que o Perfil 4 se distingue por ser o único com predominância do género feminino. Por fim, o Perfil 2, composto por doze clientes, destaca-se por apresentar a maior média de duração dos filmes visualizados, atingindo aproximadamente duas horas e meia. Com uma idade média de trinta e quatro anos e uma predominância do sexo masculino, este perfil evidencia uma distribuição de preferências particularmente diversificada, com especial apreciação por géneros como a fantasia, a biografia e o musical. Importa ainda salientar que este grupo apresenta a média de ano de lançamento mais baixa entre todos os perfis analisados. Globalmente, é possível concluir que os clientes desta coleção revelam uma preferência consistente por filmes dos géneros ação e drama, com uma tendência orientada para filmes lançados no início dos anos 2000, o que reforça a importância destes géneros e deste período temporal no comportamento dos utilizadores analisados.

De seguida, quisemos analisar uma nova coleção composta por duzentos clientes, ou seja, o dobro da quantidade presente na coleção anterior. Os clientes desta nova amostra são distintos dos utilizados na análise anterior, o que permitiu avaliar a robustez do sistema de *profiling* face a diferentes populações e aferir a consistência dos padrões identificados. Mais uma vez, pretendeu-se, criar cinco perfis distintos, a partir da nova coleção de clientes. A segmentação foi realizada com base nas mesmas premissas anteriormente definidas, recorrendo ao algoritmo e métricas estabelecidas para o *profiling*, mencionados anteriormente na secção sobre os serviços implementados. O resultado da segmentação encontra-se apresentado na Figura 13, na qual se pode ver as principais características de cada perfil obtido.

```
These five profiles were obtained with the use of k-means:


--- Basic information about Profile 4 ---
Nr. Clients: 24
Average Age: 35.58
Gender Proportion: {'Male': 19, 'Female': 5}
Main Movie Categories: ['drama', 'action', 'thriller', 'fantasy', 'romantic']
Average Year of Launch: 1995.9
Average Movie Duration: 164.59 min

--- Basic information about Profile 3 ---
Nr. Clients: 40
Average Age: 30.32
Gender Proportion: {'Male': 17, 'Female': 23}
Main Movie Categories: ['animation', "children's", 'horror', 'drama', 'thriller']
Average Year of Launch: 2008.3
Average Movie Duration: 100.64 min

--- Basic information about Profile 1 ---
Nr. Clients: 40
Average Age: 28.12
Gender Proportion: {'Male': 30, 'Female': 10}
Main Movie Categories: ['action', 'drama', 'thriller', 'animation', "children's"]
Average Year of Launch: 2004.12
Average Movie Duration: 131.22 min

--- Basic information about Profile 0 ---
Nr. Clients: 45
Average Age: 66.13
Gender Proportion: {'Male': 26, 'Female': 19}
Main Movie Categories: ['drama', 'action', 'thriller', 'animation', 'musical']
Average Year of Launch: 2004.47
Average Movie Duration: 138.64 min

--- Basic information about Profile 2 ---
Nr. Clients: 51
Average Age: 63.67
Gender Proportion: {'Male': 28, 'Female': 23}
Main Movie Categories: ['animation', "children's", 'drama', 'horror', 'thriller']
Average Year of Launch: 2010.05
Average Movie Duration: 110.01 min
```

Figura 13: Extrato do *Profling* de 200 Clientes

Nesta nova amostra, o Perfil 0 e o Perfil 2 são compostos maioritariamente por clientes mais velhos, com idades médias superiores a sessenta anos. O Perfil 0 distingue-se por uma duração média de filmes mais longa, com cerca de 139 minutos, e por uma ligeira predominância masculina. Já o Perfil 2, embora semelhante em termos etários, aprecia filmes mais curtos e mais recentes. Os dois perfis têm preferências parecidas. No entanto existe uma maior preferência por filmes de ação e musical no Perfil 0 e filmes infantis e de terror no Perfil 2. O Perfil 1 e o Perfil 3 representam faixas etárias mais jovens, com idades médias na casa dos vinte a trinta anos. O Perfil 1 é maioritariamente masculino e associa-se a filmes mais longos, com preferência por ação, drama e *thrillers*. Por outro lado, o Perfil 3 é o mais equilibrado em termos de género, com ligeira predominância feminina, e apresenta gostos semelhantes ao Perfil 2. Por fim, o Perfil 4 destaca-se pela duração média mais elevada dos filmes, quase 165 minutos, pela preferência masculina acentuada e pelo ano médio de lançamento mais antigo, 1995. Apresenta preferência por filmes de fantasia, romance, *thriller* e ação.

Analisando comparativamente os resultados obtidos a partir das duas amostras, é possível identi-

ficar padrões e comportamentos recorrentes entre os perfis gerados. Observa-se, por exemplo, a presença constante de determinadas categorias cinematográficas, nomeadamente drama, ação, e *thriller*, que surgem em praticamente todos os perfis de ambas as amostras. Verifica-se também uma consistência nas preferências em termos temporais, com a maioria dos filmes preferidos situando-se cronologicamente no intervalo dos anos 2000 a 2010. Além disso, em ambos emerge um perfil com características que podem ser consideradas como de nicho: na primeira amostra, o Perfil 2 distingue-se pela preferência por filmes de fantasia, biográficos e musicais de longa duração; na segunda amostra, é o Perfil 4 que se destaca pela valorização de filmes longos e de data de lançamento mais antiga.

Teste de Profiling de Cliente Único. Pretende-se agora proceder à validação da funcionalidade responsável por alocar novos clientes a perfis previamente definidos, com base no cálculo da similaridade do cosseno. Para tal, serão criados três clientes distintos, com características representativas, e será atribuído o perfil mais adequado a cada um, de acordo com os vetores centroide obtidos na segmentação anterior. Esta segmentação corresponde aos perfis derivados da amostra de cem clientes, apresentados na Figura 12. Paralelamente, serão também identificados os filmes recomendados a cada cliente, tendo por base a proximidade vetorial entre o centroide do perfil atribuído e os vetores dos filmes disponíveis na base de dados. Esta abordagem visa aferir a eficácia do mecanismo de atribuição e a coerência das recomendações geradas.

O primeiro cliente, C1, do sexo masculino e com 75 anos de idade, visualizou os filmes "The Exorcist" de 1973 com 122 minutos e "Titanic" de 1997 com 195 minutos. O segundo cliente, C2, do sexo feminino e com 22 anos, assistiu aos filmes "Frozen" e "The Conjuring", ambos lançados em 2013 e com 102 e 112 minutos de duração respetivamente. Por fim, o terceiro cliente, C3, também do sexo masculino, com 38 anos, visualizou os filmes "The Matrix" de 1999 com 136 minutos, "The Lord of the Rings: The Fellowship of the Ring" de 2001 com 178 minutos e "Mad Max: Fury Road" de 2015 com 120 minutos. Para cada um destes clientes é criado um vetor que o representa, seguindo o formato "Vector Client":

C1 – (1, 75.0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1985.0, 158.5)
C2 – (2, 22.0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 2013.0, 107.0)
C3 – (1, 38.0, 0, 1, 0, 0, 0, 0, 0, 0, 3, 0, 1, 1, 0, 2005.0, 144.67)

Conforme referido na secção dedicada aos serviços implementados, procede-se ao preenchimento de um formulário no qual são inseridas as informações relativas ao género, idade e filmes visualizados por cada um dos clientes.

Relativamente ao cliente C1, os resultados obtidos indicam uma elevada semelhança com todos os perfis existentes; contudo, o perfil com maior grau de correspondência é o Perfil 1, apresentando um valor de similaridade do cosseno de 0.999912. A ordem decrescente de similaridade entre os perfis é a seguinte: 1, 2, 0, 4, 3. Os valores detalhados de similaridade correspondentes aos restantes perfis serão apresentados posteriormente na Tabela 4. O Perfil 1 caracteriza-se por uma média de idades de aproximadamente 69 anos e uma preferência por filmes dos géneros drama, ação, *thriller*, musical e animação, com uma duração média de aproximadamente 133 minutos. Considerando a idade do cliente C1, bem como as durações dos filmes por ele visualizados, este perfil revela-se adequado às suas características e preferências. Importa referir, no entanto, que o Perfil 2 também apresenta um grau de compatibilidade relevante, sendo igualmente apropriado para este cliente. Foram recomendados cinco filmes, os quais se encontram apresentados na Tabela 5 e na Figura 14.

```
The client with info:
- Gender: Male
- Age: 75
- Movies: "Titanic", "The Exorcist"

... fits better in profile 1 with a scoring of 0.999912

... and the order of similarity to profiles is:
 - 1: 0.999912
 - 2: 0.999784
 - 0: 0.999706
 - 4: 0.999585
 - 3: 0.999431


Recommended movies:

-> {'Title': 'Les Misérables', 'Category': 'musical, drama', 'Year of Launch': 2012, 'Duration (min)': 158}
-> {'Title': 'Django Unchained', 'Category': 'action, drama', 'Year of Launch': 2012, 'Duration (min)': 165}
-> {'Title': 'Gladiator', 'Category': 'action, drama', 'Year of Launch': 2000, 'Duration (min)': 155}
-> {'Title': 'The Revenant', 'Category': 'drama, biopic', 'Year of Launch': 2015, 'Duration (min)': 156}
-> {'Title': 'Pulp Fiction', 'Category': 'drama, thriller', 'Year of Launch': 1994, 'Duration (min)': 154}
```

Figura 14: *Profiling* do C1

No caso do C2, verifica-se uma maior correspondência com o Perfil 3, no qual se regista um valor de similaridade do cosseno de 0.999955. Este perfil apresenta uma média de idades próxima dos 17 anos, constituindo-se como o grupo etário mais jovem entre os cinco perfis identificados. Os clientes associados ao Perfil 3 demonstram preferência por filmes de ação, *thriller*, terror, animação e

drama, com uma duração média próxima das duas horas, mais precisamente 125 minutos. Tendo em conta as características de C2, conclui-se que o Perfil 3 é o mais adequado à sua realidade. A ordem decrescente de similaridade com os perfis é a seguinte: 3, 0, 4, 2 e 1. Por fim, o C3 apresenta maior similaridade com o Perfil 2, registando um valor de similaridade do cosseno de aproximadamente 0.999978. Este perfil é caracterizado por uma média de idades de exatamente 34 anos, refletindo um grupo etário semelhante ao do cliente em questão. Em termos de preferências, o Perfil 2 demonstra afinidade por filmes dos géneros drama, ação, fantasia, musical e biográfico. Tal como anteriormente mencionado, o Perfil 2 pode ser interpretado como um perfil de nicho, dado que se destaca pelos gostos mais distintos e pelas maiores durações médias dos filmes. Tendo em consideração os títulos previamente visualizados por C3, esta associação revela-se particularmente apropriada. A ordem decrescente de similaridade relativamente aos perfis é: 2, 0, 3, 1 e 4. As Tabela 4 e 5 resumem os resultados obtidos para os três clientes.

	Ordem de similaridade do cosseno por perfil				
C1	**1:** 0.999912	**2:** 0.999784	**0:** 0.999706	**4:** 0.999585	**3:** 0.999431
C2	**3:** 0.999955	**0:** 0.999909	**4:** 0.999835	**2:** 0.999665	**1:** 0.999621
C3	**2:** 0.999978	**0:** 0.999950	**3:** 0.999900	**1:** 0.999859	**4:** 0.999751

Tabela 4: Ordem de similaridade dos perfis

	Filmes recomendados
C1	Les Misérables, Django Unchained, Gladiator, The Revenant, Pulp Fiction
C2	Zootopia, Moana, Black Swan, Whiplash, Deadpool
C3	Spider-Man: No Way Home, Inception, The Shawshank Redemption, Forrest Gump, Wonder Woman

Tabela 5: Filmes recomendados por cliente

Com base nos resultados obtidos, constata-se que a diferença nos valores de similaridade entre os vários perfis é extremamente reduzida para cada cliente. Tal facto sugere que os clientes presentes na base de dados apresentam características relativamente moderadas, não se verificando casos evidentes de perfis com preferências atípicas ou extremas como, por exemplo, a visualização exclusiva de filmes de terror da década de 1970 com durações superiores a três horas. Ainda assim, os dados analisados permitem concluir que o sistema demonstra a capacidade para gerar perfis suficientemente distintos entre si, possibilitando a atribuição de um perfil a cada cliente com base nas suas preferências e características fundamentais, bem como a geração de recomendações de filmes relevantes e adequados.

Os resultados obtidos nos dois testes realizados revelam-se, em termos globais, coerentes com os objetivos previamente estabelecidos para o sistema. Contudo, após uma análise crítica mais aprofundada, constata-se que o processo de *profiling* desenvolvido, embora seja capaz de gerar perfis distintos, apresenta sobreposição entre alguns destes perfis, nomeadamente ao nível da proximidade dos valores médios de idade e das categorias cinematográficas preferidas. Esta sobreposição sugere que, apesar de os vetores centroides associados a cada perfil apresentarem alguma distinção, a divergência entre eles não é suficientemente significativa para que se possa afirmar que são ortogonais, ou mesmo aproximadamente ortogonais, isto é, que possuem uma similaridade do cosseno nula ou próxima de zero, característica essencial para garantir a completa separação entre perfis.

No que respeita à aplicação da métrica de similaridade do cosseno, verificou-se que, apesar da sua eficácia na identificação de correspondências entre vetores, esta apresenta alguma sensibilidade a pequenas variações nos dados de entrada, o que resulta frequentemente em valores de similaridade muito próximos entre si. Importa salientar que a eficácia do processo de *profiling* depende de forma significativa da qualidade, variedade e representatividade dos dados utilizados, nomeadamente no que diz respeito à seleção dos filmes e à distribuição demográfica das amostras consideradas. Face ao exposto, considera-se relevante, para desenvolvimentos futuros, a incorporação de variáveis contextuais adicionais, tais como a frequência de visualização, os períodos do dia mais propensos ao consumo de conteúdos ou indicadores de satisfação, com o intuito de enriquecer os vetores de representação dos clientes e, assim, aumentar a granularidade dos perfis gerados.

Capítulo 6

Conclusões e Trabalho Futuro

6.1 Conclusões

Nesta dissertação foram explorados conceitos e técnicas fundamentais relacionados com a análise de similaridade, estudando-se diferentes abordagens, desde métricas tradicionais, como a distância de Minkowski e a similaridade do cosseno, contextos de aplicação e métodos avançados baseados em **embeddings** vetoriais e redes neuronais. Foi também analisado o impacto de diferentes métodos de pré-processamento para garantir representações otimizadas dos dados, seguido de uma exploração de aplicações reais de analise de similaridade e de serviços que a utilizam. Além disso, procedeu-se igualmente à análise da noção e da aplicação prática de bases de dados vetoriais em contextos reais, com especial foco em soluções como ChromaDB, LanceDB e Qdrant. Foram exploradas diversas técnicas de indexação de vetores, nomeadamente **HNSW** (Hierarchical Navigable Small World) e **LSH** (Locality-Sensitive Hashing), seguidas de uma avaliação comparativa do desempenho dos algoritmos de pesquisa associados a cada uma dessas abordagens. Abordou-se ainda a estrutura organizacional típica adotada por este tipo de bases de dados, destacando os princípios que orientam o seu funcionamento e armazenamento da informação vetorial. De seguida, fez-se uma introdução mais aprofundada sobre *customer profiling*, destacando a sua relevância prática em contextos comerciais e de marketing. Foram exploradas as diferentes abordagens que possibilitam a segmentação e a categorização de grandes volumes de clientes com base em características e comportamentos comuns, incluindo técnicas tradicionais, como métodos automatizados suportados por algoritmos de **machine learning**, e fornecendo-se alguns exemplos reais de aplicação, com o objetivo de ilustrar a utilidade e a eficácia do *customer profiling* na personalização de serviços e na tomada de decisões estratégicas.

Também se desenvolveu um protótipo computacional destinado à realização de *customer profiling* com base nas preferências cinematográficas dos clientes, permitindo a análise e segmentação de diferentes

coleções de utilizadores. Ao longo do trabalho, foram apresentadas e justificadas as decisões técnicas e metodológicas adotadas, com destaque para as tecnologias utilizadas, a arquitetura do sistema e o conjunto de funcionalidades implementadas. Seguidamente, foram conduzidos testes ao sistema, os quais permitiram aferir que, de forma geral, este cumpre adequadamente os objetivos a que se propõe, ainda que tenham sido identificadas oportunidades de melhoria, as quais serão discutidas na secção seguinte.

Por fim, deve-se referir que, este trabalho de dissertação permitiu consolidar um conjunto alargado de conhecimento, tanto em vertente teórica como prática, abrindo caminho para futuras investigações e aplicações no domínio do *customer profiling* e dos sistemas de recomendação baseados em dados vetoriais.

6.2 Trabalho Futuro

Relativamente a possíveis evoluções deste trabalho, uma das direções mais promissoras consiste na integração completa do sistema numa plataforma de **streaming** real. Tal integração permitiria aceder a dados mais ricos e complexos, refletindo o comportamento autêntico dos utilizadores, nomeadamente a frequência de visualização de conteúdos, avaliações atribuídas aos filmes e o tempo efetivo de visualização. Este tipo de dados comportamentais enriqueceria consideravelmente o processo de *profiling*, permitindo construir perfis mais precisos e personalizados.

No que respeita às representações vetoriais utilizadas para modelar os clientes e os filmes, a introdução de técnicas de **deep learning** surge como uma via particularmente relevante. Neste contexto, destaca-se a possibilidade de utilizar *autoencoders*, que são redes neuronais artificiais treinadas para aprender representações comprimidas dos dados de entrada, preservando as suas características mais significativas (Bergmann & Stryker, 2025), o que permitiria gerar vetores mais informativos e adequados para a segmentação dos clientes e a recomendação de filmes. Outra abordagem igualmente promissora consiste na utilização de *transformers*, que se trata de uma arquitetura de redes neuronais inicialmente desenvolvida para **PLN**, mas que tem demonstrado elevada eficácia em várias outras áreas. Os *transformers* são capazes de captar relações contextuais complexas entre elementos dos dados de entrada, possibilitando a geração de representações vetoriais altamente sofisticadas e contextualmente ricas (Stryker & Bergmann, 2025), que podem contribuir de forma decisiva para uma segmentação e recomendação mais eficaz. Adicionalmente, embora neste trabalho tenha sido utilizada com sucesso a métrica de similaridade do cosseno para medir a proximidade entre vetores, seria pertinente investigar o impacto de outras métricas,

como a distância de Manhattan, euclidiana e Minkowski. Cada uma destas métricas introduz diferentes sensibilidades ao cálculo da proximidade, o que pode influenciar a qualidade dos agrupamentos e, consequentemente, a pertinência das recomendações fornecidas pelo sistema.

Por fim, seria também pertinente realizar o estudo da escalabilidade do sistema em cenários com volumes significativamente superiores de dados, como coleções com milhares ou mesmo milhões de clientes e filmes. Tal análise permitiria validar a robustez e a eficiência do sistema em contextos mais exigentes, aproximando-o de uma eventual aplicação em ambientes produtivos de larga escala.

Bibliografia

Al Ayubi, A., & Achmadi, H. (2024). Customer profiling with k-means clustering and product recommendation with market basket analysis for strategy marketing MSMEs. Enrichment : Journal of Management, 14(2), 298-309. https://doi.org/10.35335/enrichment.v14i2.1914

Al-Shamri, M. (2022). Similarity Measures for Recommender Systems: Drawbacks and Neighbors Formation.

Ali, Z. (2021, December 7). A simple Word2vec tutorial. https://medium.com/@zafaralibagh6/a-simple-word2vec-tutorial-61e64e38a6a1

Ana Traqueia, Emilce Pacheco, & Eugénia Taveira. (2021). REFLEXÃO CRÍTICA SOBRE MÉTODOS E TÉCNICAS DE RECOLHA DE DADOS: INVESTIGAÇÃO-AÇÃO. UA Editora Universidade de Aveiro, (Vol. 1).

Bakurov, I., Buzzelli, M., Schettini, R., Castelli, M., & Vanneschi, L. (2022). Structural similarity index (SSIM) revisited: A data-driven approach. Expert Systems with Applications, 189. https://doi.org/10.1016/j.eswa.2021.116087

Bergmann, D., & Stryker, C. (2025, April 16). What is an autoencoder?. IBM. https://www.ibm.com/think/topics/autoencoder

Brown, J. (2024, December 30). Netflix Case Study: Unveiling the Data-Driven Strategies Behind the Streaming Giant. https://www.33rdsquare.com/netflix-case-study-eda-unveiling-data-driven-strategies-for-streaming/

Bussler, C. (2023, August 28). Vector Databases (are All The Rage). https://medium.com/google-cloud/vector-databases-are-all-the-rage-872c888fa348

Cantarero, A. (2023, August 1). Vector Indexing Explained: Everything You Need to Know. https://www.datastax.com/guides/what-is-a-vector-index

Changpinyo, S., Liu, K., & Sha, F. (2013). Similarity component analysis. Advances in Neural Information Processing Systems.

Chiappara, G. (2024, August 7). Exploring Oracle AI Vector Search: Beyond Vector Databases. https://blog.marvik.ai/2024/08/07/oracle-ai-vector-search/

Chugani, V. (2024, October 9). Minkowski Distance: A Comprehensive Guide https://www.datacamp.com/tutorial/minkowski-distance

Cochard, D. (2024, March 6). CLAP: Feature Extraction Model for Searching Audio From Text. https://medium.com/axinc-ai/clap-feature-extraction-model-for-searching-audio-from-text-dcfd4c93756e

Daniella. (2024, June 25). Image Embedding: the future of visual artificial intelligence? https://en.innovatiana.com/post/image-embedding

DB-Engines ranking. (n.d.). DB-Engines. https://db-engines.com/en/ranking/vector+dbms

Dwivedi, D. (2023, September 22). Cracking the Code: Segmentation and Personalization Strategies in Telecom. https://webengage.com/blog/personalization-strategies-in-telecom/

Elliott, O. (2024, June 4). Understanding Recall in HNSW Search. https://www.marqo.ai/blog/understanding-recall-in-hnsw-search

Elastic Platform Team. (2024, April 17). Understanding the approximate nearest neighbor (ANN) algorithm. https://www.elastic.co/blog/understanding-ann

Galic, D. (2024, November 21). What are customer profiles? A complete guide, examples, and free templates. https://www.zendesk.com/blog/create-data-rich-customer-profile/

GeeksforGeeks. (2023, December 9). Preprocessing the Audio Dataset https://www.geeksforgeeks.org/preprocessing-the-audio-dataset/

Grigoriadou, V., Frank, C., & Kostas, T. (2021). History of the concept of similarity in natural sciences. Conatus - Journal of Philosophy, 6(1). https://doi.org/10.12681/cjp.22955

Gurung, P. (2021, December 14). Magic of TF-IDF - Analytics Vidhya - Medium. https://medium.com/analytics-vidhya/magic-of-tf-idf-202649d39c2f

Han, Y., Liu, C., & Wang, P. (2023). A comprehensive survey on vector database: Storage and retrieval technique, challenge. arXiv preprint arXiv:2310.11703.

Hayes, M., & Downie, A. (2024, June 5). What is Oracle? https://www.ibm.com/think/topics/oracle

Homework Helpr. (n.d.). Similarity of triangles. https://www.homeworkhelpr.com/study-guides/maths/triangles/similarity-of-triangles/

Joshi, P. (2024, October 21). Building a Recommendation System using Word2vec: A Unique Tutorial with Case Study in Python. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/07/how-to-build-recommendation-system-word2vec-python/

Karabiber, F. (n.d.). TF-IDF — Term Frequency-Inverse Document Frequency – LearnDataSCI. https://www.learndatasci.com/glossary/tf-idf-term-frequency-inverse-document-frequency/

LanceDB. (n.d.). LanceDB. https://lancedb.com/

Learn Statistics Easily. (2024, October 14). What is: Jaccard Similarity Explained in Detail. LEARN STATISTICS EASILY. https://statisticseasily.com/glossario/what-is-jaccard-similarity-explained-in-detail/

Lee, S. (2025, April 12). Easy ways to map and analyze customer profiles. https://www.numberanalytics.com/blog/easy-ways-map-analyze-customer-profiles

Lotfi, C., Srinivasan, S., Ertz, M., & Latrous, I. (2021). Web Scraping Techniques and Applications: A Literature Review. In SCRS CONFERENCE PROCEEDINGS ON INTELLIGENT SYSTEMS. https://doi.org/10.52458/978-93-91842-08-6-38

Mahalakshmi, V., Jhoncy, A., & Geetha, A. (2020). A study on customer profiling. Malaya Journal of Matematik, S (2).

Moore, A. W. (1991). An introductory tutorial on kd-trees. Efficient Memory-Based Learning for Robot Control, 209, 1–20.

Mustafa, H. (2023, September 2). VGG-16 Model Applications 2021 - CodeX - Medium. Medium. https://medium.com/codex/vgg-16-model-applications-2021-e57f17157147

Mustapic, B. (2024, February 28). An introduction to TF-IDF: what it is & how to use it. https://www.semrush.com/blog/tf-idf/

Nidhiworah. (2024, June 3). Chroma DB - Introduction. https://medium.com/@nidhiworah02/chroma-db-introduction-25718915bae6

Olawanle, J. (2022, October 5). Big O Cheat Sheet – Time Complexity Chart. https://www.freecodecamp.org/news/big-o-cheat-sheet-time-complexity-chart/

Otten, N V. (2023, January 16). Local Sensitive Hashing (LSH)— Complete Guide & How To Get Started In Python. https://spotintelligence.com/2023/01/16/local-sensitive-hashing-lsh/

Palriwal, M. (2024, October 19). Google Cloud Adds Scalable Vector Search to Memorystore for Valkey & Redis Cluster. https://www.infoq.com/news/2024/10/vector-search-memorystore/

Pan, J.J., Wang, J., & Li, G. (2023). Survey of Vector Database Management Systems. VLDB J., 33, 1591-1615.

Qdrant. (2001, January 1). Indexing. Qdrant. https://qdrant.tech/documentation/concepts/indexing

Restack. (2025, January 25). Chroma Vs Lancedb: Comparison. https://d2wozrt205r2fu.cloudfront.net/p/lancedb-answer-vs-chroma-cat-ai

Restack. (2025, January 26). Lancedb Review: Performance Insights. https://d2wozrt205r2fu.cloudfront.net/p/lancedb-answer-review-performance-cat-ai

Salton, G., Wong, A., & Yang, C. S. (1975). A Vector Space Model for Automatic Indexing. Communications of the ACM, 18(11). https://doi.org/10.1145/361219.361220

Shimomura, L. C., Oyamada, R. S., Vieira, M. R., & Kaster, D. S. (2021). A survey on graphbased methods for similarity searches in metric spaces. In Information Systems (Vol. 95). https://doi.org/10.1016/j.is.2020.101507

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations, ICLR 2015.

Sobot, P. (2023, October 25). Introducing Voyager: Spotify's New Nearest-Neighbor Search Library. https://engineering.atspotify.com/2023/10/introducing-voyager-spotifys-new-nearest-neighbor-search-library/

Softweb Solutions - An Avnet Company. (2025, April 29). What is Customer Profiling? Smarter Way to Implement in 2025. https://www.softwebsolutions.com/resources/what-is-customer-profiling.html

Stryker, C., & Bergmann, D. (2025, April 16). What is a transformer model?. IBM. https://www.ibm.com/think/topics/transformer-model

Suard, T. (2024, July 20). Why I Love ChromaDB: The Elegance of Simplicity. https://medium.com/@ceo_44783/why-i-love-chromadb-the-elegance-of-simplicity-b5e67e8c524a

Swimm Team. (n.d.). What Is Word2Vec and How Does It Work? https://swimm.io/learn/large-language-models/what-is-word2vec-and-how-does-it-work

Taipalus, T. (2024). Vector database management systems: Fundamental concepts, usecases, and current challenges. Cognitive Systems Research, 85. https://doi.org/10.1016/j.cogsys.2024.101216

Tai, W. (2024, March 2). HNSW indexing in Vector Databases: Simple explanation and code. https://medium.com/@wtaisen/hnsw-indexing-in-vector-databases-simple-explanation-and-code-3ef59d9c1920

Tibrewal, T. P. (2023, July 1). Support Vector Machines (SVM): an intuitive explanation. Medium. https://medium.com/low-code-for-advanced-data-science/support-vector-machines-svm-an-intuitive-explanation-b084d6238106

Tudor, N. (2021, February 10). 7 Real-World Examples of How Brands are Using Big Data Analytics. https://www.bornfight.com/blog/7-real-world-examples-of-how-brands-are-using-big-data-analytics/

Unreal Speech. (2024, February 23). Introduction to CLAP: Unveiling the Symphony of Language and Sound. https://blog.unrealspeech.com/introduction-to-clap-unveiling-the-symphony-of-language-and-sound/

Venkatesh, R., Keerthi, C. V. N., Harshitha, A., Deepika, K. S., & Rajyalakshmi, E. (2023). Customer segmentation using support vector machine. AIP Conference Proceedings, 2724. https://doi.org/10.1063/5.0130188

Wang, J., Shen, H.T., Song, J., & Ji, J. (2014). Hashing for Similarity Search: A Survey. ArXiv, abs/1408.2927.

WEKA. (2025, February 26). Vector Database: Everything you need to know - WEKA. https://www.weka.io/learn/guide/ai-ml/vector-dabase/

Whittle, S., & Foster, M. (1989). Customer Profiling: Getting into your Customer's Shoes. Management Decision, 27 (6). https://doi.org/10.1108/00251748910132575

Ye, Z., Huang, R., Ren, Y., Jiang, Z., Liu, J., He, J., Yin, X., & Zhao, Z. (2023). CLAPSpeech: Learning Prosody from Text Context with Contrastive Language-Audio Pre-training. Proceedings of the Annual Meeting of the Association for Computational Linguistics, 1. https://doi.org/10.18653/v1/2023.acl-long.518

Yun, C. (2024, July 10). Vector search for Amazon MemoryDB is now generally available. https://aws.amazon.com/pt/blogs/aws/vector-search-for-amazon-memorydb-is-now-generally-available/