Universidade de Aveiro
2015

Departamento de Eletrónica,
Telecomunicações e Informática

RUI DANIEL
ALVES MENDES

SIMILARIDADE ENTRE PERFIS SOCIAIS

Universidade de Aveiro
2015

Departamento de Eletrónica,
Telecomunicações e Informática

RUI DANIEL ALVES MENDES

SIMILARIDADE ENTRE PERFIS SOCIAIS

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Sistemas de Informação, realizada sob a orientação científica da Doutora Ana Maria Perfeito Tomé, Professora Associada do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho às duas pessoas mais importantes da minha vida, minha esposa Andreia e, a razão da minha existência, minha filha Lara.

Friends can help each other. A true friend is someone who lets you have total freedom to be yourself - and especially to feel. Or, not feel. Whatever you happen to be feeling at the moment is fine with them. That's what real love amounts to - letting a person be what he really is.

Jim Morrison

o júri

presidente	Prof. Dra. Pétia Georgieva Georgieva Professora Auxiliar da Universidade de Aveiro
arguente	Prof. Dr. Viriato António Pereira Marinho Marques Professor Coordenador do Instituto Superior de Engenharia de Coimbra
orientadora	Prof. Dra. Ana Maria Perfeito Tomé Professora Associada da Universidade de Aveiro

palavras-chave

similaridade, *clustering*, sistemas de recomendação, perfil social, redes sociais, exploração de dados, interesses.

resumo

Na última década, a utilização da Internet tornou-se viral e de extrema importância posicionando-se, atualmente, numa parte integral das nossas vidas incluindo a parte social. As redes sociais *online* são uma das fontes mais ricas de informação sobre os perfis de utilizadores. Ao lidar com dados de redes sociais, a similaridade entre perfis representa uma área que tem tido algum destaque. Uma ferramenta capaz de identificar corretamente utilizadores semelhantes pode ser utilizada em diversas áreas e contribuir para tomadas de decisão importantes que poderão resultar em proveitos, sejam de cariz financeiro ou melhoria da qualidade de vida de pessoas (por exemplo, saúde). Nesta dissertação são realizados estudos com diferentes métricas de distância de forma a determinar a similaridade entre perfis. É possível, também, criar agrupamentos de perfis assim como correlacionar interesses. Posteriormente, é feita uma análise de *performance* entre diversos algoritmos de *clustering*, nomeadamente o K-*Means*, *Clustering* Hierárquico, DBSCAN e BIRCH. As medidas de similaridade foram também utilizadas para estimar valores associados aos interesses dos utilizadores, numa abordagem inspirada nos sistemas de recomendação.

keywords

similarity, clustering, recommendation systems, social profile, social network, data mining, interests.

abstract

Over the last decade, Internet usage has gone viral and become extremely important, positioning itself as an integral part of our lives and social interactions. Social networks are now one of the richest sources of information regarding user profiles. By dealing with social network data, an area that as recently seen growing interest, one can create methods to study profile similarity, using available user data. A possible tool to correctly identify similar users may be applied in a multitude of areas and contribute to the decision making process that may draw gains to people's lives, either in financial perspective or life quality (such as health). In this dissertation, metrics that are applied when determining profile similarity were researched and discussed, giving a complete overview on the concepts involved and difficulties experienced. Moreover, it is possible to create profile clusters as well as correlate interests. As such, a performance analysis of different clustering algorithms is done, namely K-Means, Hieratical Clustering, DBSCAN and BIRCH. Techniques used in recommendations systems are also discussed. Finally, future work is proposed where this project would serve as the basis of a recommendation and profile analysis systems.

Índice

Índice de figuras

Índice de tabelas

Índice de algoritmos

Acrónimos

	Português	Inglês
API		*Application Programming Interface*
BIRCH		*Balanced Iterative Reducing and Clustering using Hierarchies*
CF	Filtragem Colaborativa	*Collaborative Filtering*
CF-tree		*Clustering Feature Tree*
CSS		*Cascading Style Sheets*
DBSCAN		*Density-based Spatial Clustering of Applications with Noise*
FOAF		*Friend of a Friend*
HTML		*HyperText Markup Language*
IT	Tecnologias de Informação	*Information Technology*
JSON		*JavaScript Object Notation*
LSH		*Locality-Sensitive Hashing*
LSI		*Latent Semantic Indexing*
MAE	Erro Absoluto Médio	*Mean Absolute Error*
MedAE	Mediana do Erro Absoluto	*Median Absolute Error*
MSE	Erro Quadrático Médio	*Mean Squared Error*
PCA	Análise de Componentes Principais	*Principal Component Analysis*
RDF		*Resource Description Framework*
RHH		*Random Hyperplane Hashing*
SGBD	Sistema de Gestão de Base de Dados	
SVD		*Singular Value Decomposition*
SVG		*Scalable Vector Graphics*
SWE		*Software with Emotion*
UML		*Unified Modeling Language*
URI		*Uniform Resource Identifier*
XML		*EXtensible Markup Language*
YAML		*YAML Ain't Markup Language*

1. Introdução

"We are drowning in information and starving for knowledge."
Rutherford D. Rogers

Este capítulo descreve o âmbito e os objetivos do trabalho. Para além disso, descreve ainda a estrutura do documento.

1.1 Motivação

Na última década, a utilização da *Internet* tornou-se viral e de extrema importância. Atividades ou tarefas que costumavam ser manuais ou locais são, atualmente, realizadas através da *Internet*, tais como marcação de viagem para férias, adquirir produtos em lojas ou mesmo transformar negócios a um nível global.

A *Internet* está a tornar-se uma parte integral das nossas vidas estando incluída, igualmente, a componente social. As pessoas comunicam entre si através da *Internet*, falam com familiares que se encontram distantes, conhecem novas pessoas em páginas comunitárias, discutem e revêm notícias do seu interesse e podem, inclusive, encontrar o "amor da sua vida" apenas navegando na "autoestrada digital". O progresso da *Internet* e Tecnologias de Informação (TI) faz com que sejam criadas inúmeras oportunidades para a sociedade desenvolver novos produtos e serviços, comunicar e partilhar dados. Devido a estes fatores, muitas áreas de negócio e entidades científicas utilizam estes dados para extraírem informações preciosas que são utilizadas, por exemplo, para detetar rapidamente ocorrências de catástrofes naturais ou então proporcionar melhores experiências aos clientes numa compra *online*. No primeiro caso é possível, por exemplo, detetar a presença de um sismo em determinada localidade através da extração de mensagens de redes sociais em tempo real. No segundo, as lojas *online* podem utilizar técnicas de exploração de dados (*data mining*) para descobrirem relações entre caraterísticas dos clientes e o seu comportamento ao nível de compras. Uma vez identificadas as preferências do cliente, uma loja poderá melhorar o serviço prestado.

Uma das fontes de informação sobre os perfis de utilizadores são as redes sociais *online*, pelo que diversas pessoas vêm novas oportunidades na recolha e utilização de dados provenientes das mesmas. Ao lidar com dados de redes sociais, uma área que tem tido algum destaque é a similaridade entre perfis. Uma ferramenta possível de identificar corretamente utilizadores semelhantes, poderá trazer grandes vantagens para diversos contextos. Pode-se, por exemplo, utilizar essas informações para recomendações de novas ligações, novos grupos, etc..

O desafio de determinar a similaridade entre os diversos utilizadores tem recebido uma grande atenção em muitos campos das Ciências da Computação, Sistemas de Recomendação [1] ou Modelação de Utilizador [2], porém continua a ser uma área ainda com algum espaço de exploração no contexto das grandes redes sociais, como o *Facebook*.

A tarefa de identificar potenciais clientes é fundamental na área de investigação do *marketing* e no *e-commerce* [3]. De facto, as empresas podem identificar clientes com necessidades e comportamentos de consumo semelhantes e agrupá-los. A segmentação de clientes torna-se vantajoso uma vez que permite delinear estratégias comerciais de modo a que os clientes obtenham maior satisfação.

Outra das áreas onde a similaridade entre perfis é considerada bastante importante é na medicina uma vez que, encontrando pessoas com os mesmos sintomas ou doenças, pode significar ganho de informação que pode ajudar na escolha dos procedimentos de tratamento e, desta forma, obtenham melhores resultados. Encontrar "pacientes como eu" através da recolha de informações em fóruns ou redes sociais tem sido um grande desafio na área. O maior desafio é na definição de um perfil, uma vez que é uma área minuciosa onde a importância dos atributos pode ser dinâmica consoante o caso de doença alvo de estudo [4].

Como se pode concluir através dos parágrafos anteriores, a similaridade entre perfis pode ser utilizada em diversas áreas e contribuir para tomadas de decisão importantes que poderão resultar em proveitos, sejam de cariz financeiro ou melhoria da qualidade de vida de pessoas (por exemplo, saúde). A motivação para a elaboração desta dissertação deveu-se, sobretudo, aos fatores supracitados.

1.2 Objetivos

Com a realização desta dissertação prevê-se a obtenção de diversas metas. Dado um conjunto de dados, os objetivos consistem em:

- Dado um perfil, determinar os perfis mais similares.
- Determinar grupos de perfis que partilhem os mesmos atributos (interesses, por exemplo).
- Determinar grupos de interesses, ou seja, responder à questão: "Se um perfil X gosta do interesse Y, que outros interesses poderá ele estar também interessado?"

Os próximos capítulos irão descrever os principais conceitos e todas as tarefas realizadas para que fosse possível obter respostas aos três pontos supracitados.

1.3 Contribuição

O conhecimento extraído com esta dissertação irá ser vantajoso na criação de um componente que será embutido no produto *Software with Emotion*[1] (SWE), pertencente à empresa *Ubiprism*, Lda[2]., a qual gentilmente disponibilizou o conjunto de dados para que a realização deste projeto fosse possível.

O SWE consiste num produto que determina um valor de desconto personalizado tendo como base o perfil social do cliente conseguindo, desta forma, otimizar a receita para o vendedor assim como também maximizar

[1] http://www.swe.com.pt
[2] http://www.beubi.com/ e http://www.ubiprism.pt

a probabilidade de compra por parte do cliente. Utiliza técnicas de *machine learning*, pelo que um dos objetivos passa por conseguir obter o maior número de dados possível sobre o cliente utilizando, desse modo, técnicas de inferência de alguns dados a partir de perfis similares.

1.4 Organização da dissertação

A presente dissertação está dividida em seis capítulos, os quais irão ser descritos em seguida:

- **Capítulo 1: Introdução** – onde é dada uma visão geral do projeto, objetivos e uma breve descrição da estrutura do documento.
- **Capítulo 2: Enquadramento global** – este capítulo visa referenciar os principais temas que influenciaram as decisões tomadas nos seguintes assim como dar uma revisão de estudos e técnicas de outros autores relativos ao tema da dissertação.
- **Capítulo 3: Similaridade entre perfis sociais** – onde é caraterizado o conjunto de dados e descrita a estratégia abordada. São, também, definidos diversos conceitos teóricos utilizados na dissertação, tais como medidas de similaridade entre vetores e algoritmos de *clustering*.
- **Capítulo 4: *matchly* (protótipo)** – este capítulo visa fornecer uma descrição da modelação e implementação do protótipo denominado *matchly*. Requisitos, casos de utilização, diagrama de classes e exemplificação de funcionalidades são temas que constam no conteúdo do capítulo.
- **Capítulo 5: Resultados e discussão** – este capítulo tem como objetivo listar os resultados obtidos e discuti-los. É, também, descrita a estratégia de avaliar os resultados.
- **Capítulo 6: Conclusão** – Finalmente, neste capítulo são listadas as principais conclusões do projeto assim como é indicada a lista de melhorias e trabalho futuro.

As referências bibliográficas e anexos estarão no final do documento, ou seja, depois do capítulo da conclusão.

2. Enquadramento global

"*Getting information of the Internet is like taking a drink from a firehose*."

Mitchell Kapor

Este capítulo visa dar uma noção do estado de arte referente ao tópico similaridade entre perfis nas redes sociais. Temas como redes sociais, perfil social, similaridade entre perfis e sistemas de recomendação irão ser especificados nas seguintes secções.

Este capítulo é de extrema importância não só para entender o fundamento deste trabalho como também para perceber as decisões tomadas ao longo do projeto.

2.1 Redes Sociais

Grande parte das comunicações atuais entre organizações, comunidades ou mesmo entre dois indivíduos são realizadas com recursos tecnológicos através de diálogos *online* interativos. Esses recursos são conhecidos como *social media*, onde se podem incluir tecnologias baseadas na *web* e em dispositivos móveis.

Segundo [5], *social media* pode ser definido como "um grupo de aplicações baseadas na *Internet* que foi construído com bases ideológicas e tecnológicas da *Web* 2.0, e que permite a criação e partilha de conteúdo produzido pelos utilizadores". Na Figura 2.1, pode-se observar as diversas categorias e tipos de tecnologias do *social media* onde cada tecnologia suporta um tipo de comunicação diferente. Neste projeto, o foco de investigação é direcionado apenas para as redes sociais, tais como o *Facebook*, *Google+* ou *Linkedin*. Estas redes servem diferentes propósitos. O *Facebook* e *Google+* fornecem recursos que promovem a socialização entre os utilizadores enquanto que o *Linkedin* fomenta a criação de redes profissionais.

É inegável que, nos últimos anos, as redes sociais têm vindo a crescer a nível de popularidade. Oferecem modos interessantes dos utilizadores se ligarem, comunicar e partilhar informação entre outros membros que também utilizem a plataforma. A existência de um número elevado de utilizadores ativos nessas plataformas faz com que as redes sociais sejam uma extensa e valiosa fonte de informação *online* produzindo, diariamente, uma enorme quantidade de dados. Este cenário pode ser considerado como um caso típico de *Big Data*[3].

[3] *Big Data* pode ser entendido como análise de grandes quantidades de dados que, através da utilização de ferramentas específicas, é possível extrair informações em tempo útil.

Figura 2.1- Categorias de social media [4]

Uma estratégia para lidar com este volume de dados é a utilização de pesquisa por elementos similares [6]. Quando esta técnica é transposta para o contexto das redes sociais, o objetivo passa pela tentativa de otimizar os efeitos de manuseamento da informação, proporcionando uma melhor experiência aos utilizadores fornecendo-lhes, deste modo, diversas alternativas para encontrarem conteúdos que sejam relevantes segundo os seus interesses. Um exemplo prático do uso desta técnica é a recomendação de amizades tendo como base perfis similares ao utilizador.

Num sentido mais amplo, uma rede social é construída a partir de dados relacionados e pode ser definida como um conjunto de entidades sociais, tais como pessoas, grupos ou mesmo organizações, onde existe algum padrão ou interações entre elas. Estas redes são normalmente modeladas e representadas através de grafos matemáticos, onde os vértices representam as entidades sociais e as arestas dizem respeito às ligações estabelecidas entre elas [7]. Um exemplo da representação de uma rede social num grafo pode ser visto na Figura 2.2.

As redes sociais permitem, deste modo, formar comunidades que se caraterizam como sendo subgrafos em que diferentes pessoas partilham um mesmo interesse. Desta forma, os utilizadores que possuam uma ligação de amizade, geralmente denominados amigos nas redes sociais, posicionam-se como elementos chave do perfil uma vez que numa rede social os utilizadores interagem entre eles, partilham conteúdo e, frequentemente, pesquisam por outros [8].

Sendo uma das fontes de informação mais valiosas da *web*, os dados produzidos pelas redes sociais têm sido utilizados em diversas áreas. Um exemplo diz respeito às ciências sociais, onde têm sido realizados esforços em estudos de modo a perceber o comportamento da sociedade, comunidades ou estilos de vida [9]. Outro

[4] Fonte: https://www.flickr.com/photos/fredcavazza/2564571564/, acedida em Abril de 2015

exemplo é a área de *marketing*, onde o objetivo passa por extrair informações que suportem tomadas de decisão focadas em gerar mais receitas e, consequentemente, maior lucro [10]. Como exemplo real, pode ser referida a aplicação de recomendação na plataforma turística *TripAdvisor*[5].

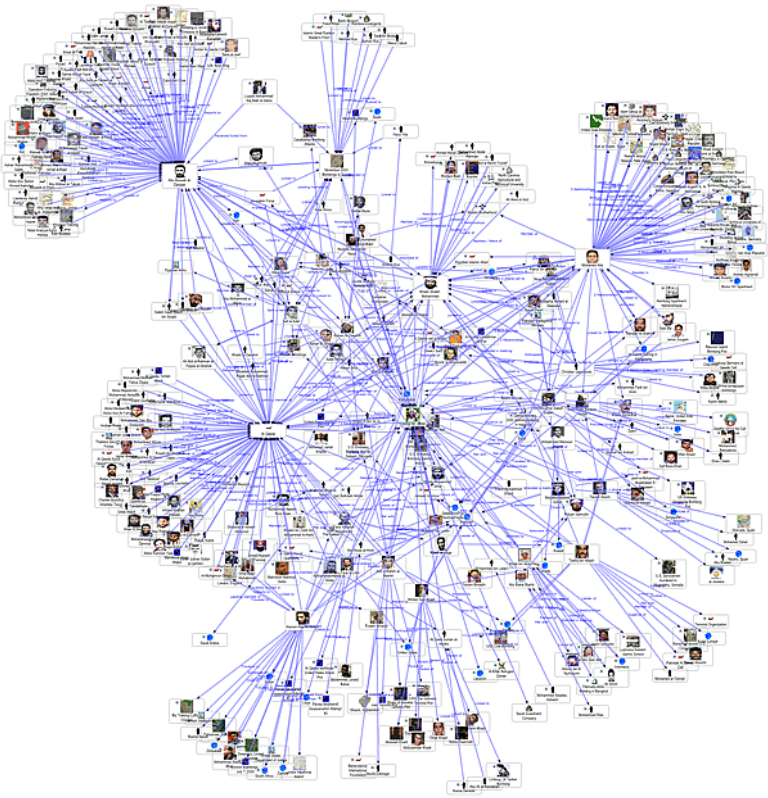


Figura 2.2 - Exemplo de um grafo de uma rede social [6]

Geralmente, as plataformas das redes sociais possuem diferentes estruturas ou esquemas de representação dos utilizadores e também é comum não permitirem a partilha de informações entre elas (por exemplo, partilha de imagens, amizades e comentários) tornando-as "ilhas isoladas de dados". Perante estes factos, a construção de um perfil social único representa um verdadeiro desafio.

2.1.1 Perfil Social

Quando um utilizador se regista numa rede social, geralmente, é-lhe pedido que preencha os dados de um formulário. Esses dados irão ser, posteriormente, utilizados para completar o perfil.

[5] http://www.tripadvisor.com/
[6] Fonte: http://www.fmsasg.com/socialnetworkanalysis/SocialNetworkAnalysis_Graph.gif ; acedida em Maio de 2015

Um perfil de utilizador típico de uma rede social é caraterizado por diversos atributos ou caraterísticas tais como localização, local de nascimento, atividades, interesses, música favorita, além de outras informações gerais. Opcionalmente, os utilizadores podem deixar alguns campos em branco, ou seja, optar por não preencher todos os atributos pedidos. Outras plataformas de redes sociais permitem ainda que os utilizadores possam criar os seus próprios campos. Adicionalmente e, caso seja pretendido, as plataformas permitem que o utilizador edite campos do perfil, como por exemplo alterar o estado civil, profissão, entre outros.

Devido à constante oferta de novas funcionalidades e recursos, por exemplo permitindo ao utilizador carregar vídeos, partilhar eventos, imagens ou outros elementos do seu interesse, o conteúdo do perfil tem vindo a ficar mais rico [8]. Deste modo, é possível extrair cada vez mais dados de forma a inferir informações que não estejam diretamente acessíveis (por exemplo, estilo de vida).

Existindo diversas redes sociais, os utilizadores podem possuir diversos perfis sociais. Caso o objetivo seja obter o máximo de informação sobre o perfil, então a integração dos diversos perfis provenientes das diferentes redes sociais torna-se indispensável para obter uma vista detalhada. Pelo que os autores em [11] levantam algumas questões relevantes:

1. Como é que se pode agregar e representar os dados do utilizador que se encontram distribuídos por diversos sistemas de redes sociais, geralmente sistemas heterogéneos, para perfilar os interesses do utilizador?
2. Existindo um grande número de fontes de dados que podem ser utilizados para perfilar o utilizador, como medir o grau de importância ou relevância de cada fonte de dados e de que maneira os dados de cada uma delas influencia na criação do perfil?
3. Como combinar dados de fontes externas (por exemplo, o conceito de *web* semântica) ao perfil de modo a enriquecer os interesses dos perfis dos utilizadores?

Antes de serem listadas diversas estratégias que visam responder às perguntas supracitadas, é importante definir neste contexto o conceito de *folksonomy*. A estrutura emergente que evolui ao longo do tempo quando utilizadores (*folks*) anotam (categorizam) recursos com etiquetas (*tags*) é chamada de *folksonomy* [12]. De uma forma resumida, *folksonomy* é simplesmente um conjunto de ligações utilizador-*tag*-recurso e um atributo de tempo que indica a data de criação da *tag*.

A criação de um perfil social "geral" foi objeto de estudo em diversos trabalhos. Por exemplo, em [13] os autores consideram duas opções distintas para definir o perfil do utilizador e apresentam as respetivas estratégias de agregação para cada um dos casos. A primeira baseia-se em formulários onde a estratégia de agregação consiste em unir todos os pares atributo-valor (ou seja, o campo do perfil e respetiva resposta) provenientes das diferentes fontes de dados. A segunda opção consiste em criar o perfil baseado em *tags*. Perfis

baseados em *tags* aparecem em sistemas sociais tais como *Flickr*[7] ou *Delicious*[8] onde é permitido que os utilizadores categorizem imagens e favoritos com *tags* escolhidas livremente. A tarefa de agregação destas caraterísticas é uma tarefa algo complicada, uma vez que as mesmas *tags* podem ter diferentes graus de relevância dependendo da rede social proveniente. Os autores consideram, então, que é necessário representar diferentes pesos consoante o contexto e, na fase de agregação, o peso final da *tag* corresponde ao somatório de todos os pesos normalizados da *tag* em cada rede social. Perfis baseados em *tags* têm sido estudados em contexto de recomendação de música [14], *bookmarking* social[9] ou em páginas de informação turística [15].

Em [16], os autores consideram o problema de integração de dados provenientes de diferentes *folksonomies*. A estratégia consiste em recolher dados das diferentes *folsksonomies* e utilizar algoritmos de *clustering* para ser possível criar grupos de *tags* que são mapeadas nos conceitos de uma ontologia. Outra abordagem, [17], consiste em recolher os dados das diferentes *folksonomies* e determinar a correlação entre eles para criar um perfil global. Em [18], os autores propõem uma abordagem onde o sistema recolhe dados sobre as atividades do utilizador em redes sociais e utilizam ontologias não só para analisar como também para modelar os próprios interesses do utilizador.

No contexto da *web* semântica[10], o termo *Linked Data* denota um conjunto de práticas para ser possível expor, partilhar e ligar dados através do uso das tecnologias URI[11] e RDF[12]. Diversos estudos [19] baseiam-se em recursos ontológicos utilizando, por exemplo, o *WordNet*[13] como ferramenta auxiliar e de validação linguística. Contudo, estas técnicas são incompatíveis no contexto das *folksonomies*, uma vez que as *tags* escolhidas pelos utilizadores são livres, os quais podem representar sinónimos ou mesmo homónimos sobre o assunto em causa.

Atualmente, é evidente que se torna mais fácil ligar perfis de utilizadores que se encontram distribuídos, muito devido ao aparecimento de tecnologias que padronizam e uniformizam entidades e/ou relações tais como o FOAF [20], SIOC [21] ou GUMO [22]. Algumas estratégias desenvolvidas no contexto de *Linked Data* lidam com problemas de recomendação de recursos a utilizadores de *sites* sociais [23]. Por exemplo, em [24] os autores descrevem um sistema de recomendação baseado em FOAF que opera no campo da música. Este sistema representa gostos musicais pelas médias de perfis FOAF e recolhe *feeds* RSS[14] para conseguir obter informações sobre concertos ou outros eventos musicais.

---

[7] https://www.flickr.com/
[8] https://delicious.com/
[9] *Bookmarks* sociais são ferramentas onde o objetivo contempla representar e organizar recursos da *web* de modo colaborativo para que facilite o acesso e partilha dos mesmos.
[10] O objetivo passa por criar uma *web* com toda a informação organizada de forma que seja entendida não apenas pelos humanos mas também pelas máquinas. Incorpora significado às informações da *web*.
[11] *Uniform Resource Identifier* – conjunto de carateres que visa identificar inequivocamente um recurso *web*.
[12] *Resource Description Framework* – modelo padrão que é utilizado para troca de dados na *Web.*
[13] https://wordnet.princeton.edu/
[14] RSS é um padrão desenvolvido em linguagem XML que permite aos responsáveis por *sites* divulgarem notícias ou novidades destes. A sigla RSS tem mais de um significado: RDF *Site Summary*, *Really Simple Syndication* ou ainda *Rich Site Summary*.

2.1.1.1. Principais adversidades

Esquema de dados não padronizado

Apesar das redes sociais, atualmente, disponibilizarem acesso a dados através de APIs[15], o esquema de informação não é uniforme, pelo que há necessidade de processar e normalizar dados provenientes das diferentes aplicações. Um exemplo pode ser representado quanto ao estado profissional onde "Eng. *Software*" e "Engenheiro de Sof." retratam o mesmo mas não se torna óbvio quando processado automaticamente.

Em [13], o autor afirma que, enriquecendo o perfil do *Twitter* com informação proveniente de outros serviços, o resultado é uma taxa de preenchimento na ordem dos 98%. Afirma, também, que o perfil pode beneficiar com a junção dos perfis de *Facebook* e *Linkedin* levando a que, em média, a taxa de preenchimento dos perfis do *Linkedin* aumente cerca de 7%. A razão deve-se ao facto que os perfis do *Facebook* fornecem atributos – interesses, educação, entre outros – que não costumam estar disponíveis noutras plataformas.

Privacidade

Um ponto importante a referir é que o utilizador pode restringir o acesso à informação que preenche no perfil, deixando de ser possível aceder aos dados livremente. Por exemplo, o utilizador pode restringir o campo da sua idade para que apenas os amigos tenham acesso.

As redes sociais também vão alterando as suas políticas de privacidade. Por exemplo, num passado recente, o *Facebook* permitia aos investigadores terem acesso a toda a informação dos utilizadores numa rede local (por exemplo, utilizadores de uma cidade, alunos de uma universidade, etc.).

Geralmente, existem diferentes níveis de privacidade que vão desde a aplicação de medidas de restrição a todos, apenas a amigos ou escolhendo utilizadores específicos. Esta restrição implica que, em alguns casos, o resultado do acesso aos perfis pode ter dados não preenchidos. Uma estratégia com o intuito de resolver este problema será visto no ponto de inferência de atributos.

Atributos com diferentes propósitos

Como referido anteriormente, existe um número elevado de redes sociais com diferentes propósitos. Por exemplo, em redes sociais como o *Facebook* os utilizadores estabelecem relações de amizade quando descobrem perfis com atributos semelhantes (por exemplo, mesma zona geográfica ou mesma universidade). No *Linkedin*, as pessoas ligam-se entre elas de modo a criarem redes profissionais e, possivelmente, encontrarem oportunidades de carreira [25].

[15] *Application Programming Interface* - ou, em português, *Interface* de Programação de Aplicativos. Esta *interface* é o conjunto de padrões de programação que permite a construção e utilização de aplicativos que, de outra forma, não seria de fácil utilização para os utilizadores.

Perante estes factos, torna-se evidente que cada rede social difere nos dados que pede ao utilizador. Os perfis públicos do *Facebook* e *Linkedin*, por exemplo, são mais específicos do que os do *Twitter*, *Flickr* ou perfis do *Google*.

Segundo [13], cerca de apenas 48,9% completam os dados que caraterizam o seu perfil na plataforma *Twitter* apesar da plataforma não obrigar o preenchimento de todos os campos. Mais especificamente, atributos como a localização do utilizador e página *web* pessoal de outra rede social raramente são preenchidos. Não obstante, os perfis de *Facebook* e *Linkedin* têm uma taxa de preenchimento de 85,4% e 82,6%, respetivamente. Estas taxas são relevantes uma vez que estas duas plataformas não apresentam um número muito grande de campos obrigatórios.

Inferência de atributos

Em [26], os autores demonstram que, quando estão a perfilar utilizadores, é possível inferir atributos que não estejam disponíveis diretamente dos dados provenientes das redes sociais através do uso das "poucas" informações que esses perfis disponibilizam. O processo de inferência consiste em utilizar as informações das ligações de amizade ou outras ligações que possuam, tais como grupos e atividades. Em sociologia, esta tendência é definida como homofilia [27].

Um estudo por Fiore e Domath [28] sobre interações entre um grande número de utilizadores num sistema de encontros *online* mostrou que é possível perfilar um utilizador utilizando estratégias de inferência dispondo apenas de 20% dos perfis que estejam de algum modo ligados ao perfil social do utilizador (por exemplo, um grupo onde esteja inserido).

Em [29], os autores consideram que inferir informações de um perfil consiste em agregar as informações dos perfis dos amigos ou então das informações dos perfis dos grupos a que ele pertence. Esta técnica foi utilizada para determinar visões políticas, orientação sexual ou mesmo caraterísticas "escondidas" do perfil. Para perfilar o utilizador, a estratégia dos autores em [30] consistiu em inferir os dados provenientes do perfil social que não estavam disponíveis através de votos, utilizando apenas informações dos amigos em comum, baseando-se também no conceito de homofilia. O processo consiste em agrupar o máximo de informações dos amigos em comum, registar o número de frequências das diversas opções possíveis e, no final, é escolhido aquele que tiver um maior número de presenças (votos). No entanto, estes dados são cada vez mais difíceis de obter devido à restrição do acesso à informação por parte das redes sociais. Deste modo, torna-se difícil ter acesso a informações de amigos do utilizador o que dificulta a inferência através desta estratégia.

2.1.2 Rede social *Facebook*

Uma vez que a rede social utilizada para a recolha de dados neste trabalho teve como base o *Facebook*, torna-se relevante especificar algumas das principais caraterísticas inerentes a esta plataforma.

O *Facebook* pode ser considerado de utilidade social uma vez que ajuda as pessoas a perceberem o mundo que os rodeia. Contém informações sobre o perfil que é a base na formação de inter-relações entre pessoas assim como disponibiliza recursos para partilha de informações entre elas.

O *Facebook* apresenta diversas funcionalidades, das quais se destacam:

- **Informação do perfil** – geralmente é preenchido na fase do registo ou então atualizado ao longo do tempo.
- **Partilha de informação privada** – o *Facebook* possui funcionalidades de partilha de informações apenas entre utilizadores que partilhem uma relação.
- O **botão** "**Like**" – começou a funcionar em 2010. A expressão "like" representa uma atitude de "aceitação" e resposta a um "post" que um utilizador do *Facebook* tenha criado e pode ser visto por outros utilizadores. Esta funcionalidade pode, também, ser utilizada na extração dos interesses do utilizador.
- **Mural de notícias** (*News feed*) – onde podem ser vistas as principais alterações dos perfis, eventos que se avizinham, aniversários, entre outras atualizações. Também podem aparecer atualizações que ocorreram em murais de amigos.
- **Grupos** – os grupos podem ser criados por utilizadores individuais. Permite que membros partilhem ligações, vídeos, *sites*, coloquem questões, eventos, documentos e comentários. Existem diversas políticas de privacidade que se podem configurar, podendo definir como sendo privado ou de acesso público, por exemplo.
- **Amigos** – um utilizador fica ligado com amigos depois de os adicionar à sua rede. O *Facebook* permite classificar os amigos em diferentes grupos (família, amigos próximos, etc.)

Um ponto especialmente importante a descrever no que diz respeito a este trabalho é o campo "Interests" do *Facebook*, onde os utilizadores podem listar as atividades das quais nutrem alguma paixão ou tópicos que estão de alguma forma interessados. Por exemplo, uma grande quantidade de utilizadores lista a música favorita como fazendo parte dos seus interesses. Uma vez que este campo pode conter múltiplas respostas, os interesses encontram-se separados por vírgulas. Como se trata de um campo de preenchimento livre poderão surgir problemas uma vez que diferentes palavras ou expressões poderão ter o mesmo significado tornando difícil classificar corretamente através de processos automáticos. Por exemplo, interesses como "jogar à bola" ou "praticar futebol" significam a mesma coisa, mas iriam ser classificados como interesses diferentes. É necessário uma análise cuidada dessas palavras e um processamento de modo a desambiguar os termos presentes neste campo.

Uma estratégia interessante, focada em [25], consiste em construir os perfis de utilizadores com valores retirados dos "Interests" provenientes do *Facebook* adicionando outros elementos, tais como a educação, profissão, género, etc. Não utilizam nenhuma fonte para validar os termos extraídos dos interesses (por exemplo, ontologias externas), mas utilizam os interesses já existentes na base de dados. Desta forma, o sistema de comparação de perfis funciona em qualquer domínio sem qualquer informação *à priori*.

Além do campo “Interests”, é possível utilizar outros dados para modelar um perfil. Por exemplo, em [31], os autores utilizam não só os “Likes” como também a informação dos “posts”, mais especificamente extraem os termos criados pelo utilizador como forma de determinar os interesses para perfilar o utilizador.

2.1.3 Similaridade entre perfis sociais

Um dos principais problemas do cenário em redes sociais é determinar se dois utilizadores podem ser considerados similares.

Nesta secção, irão ser descritas algumas estratégias que envolvem medidas de similaridade entre perfis sociais utilizadas em diversos estudos.

Existem duas grandes linhas de investigação no que diz respeito à deteção de similaridade de pares de utilizadores [32]. Uma das abordagens baseia-se em relações sociais (especialmente relações de amizade) existentes entre os utilizadores [33]. Segundo [34], a similaridade deriva de dois principais fatores: influência social, que consiste em influenciar utilizadores de modo a adotarem diversos comportamentos interagindo com eles; e homofilia, ou seja, a tendência dos indivíduos criarem relações com outros indivíduos que lhes sejam similares. Extensos estudos empíricos demonstram que existe uma grande evidência da existência da homofilia em contextos reais. Por exemplo, um estudo com 12,067 pessoas elaborado entre 1971 e 2003 indicou que uma pessoa tem muito mais probabilidade de ser obeso caso os seus amigos também façam parte do grupo dos obesos [35].

Nas redes sociais como o *Facebook*, as ligações de amizade são um indicador fiável de similaridade entre utilizadores mas poderá não ser suficiente para se determinar a similaridade entre dois utilizadores. De facto, uma vez que o número de utilizadores de uma rede social é, geralmente, grande, caso se escolhessem aleatoriamente dois utilizadores, haveria uma grande probabilidade de não se conhecerem. Logo, esses utilizadores seriam, à partida, catalogados como não similares, o que poderá estar errado uma vez que os utilizadores poderão partilhar os mesmos interesses, ideais políticos, etc. e, assim, o cálculo da similaridade estaria distorcido. Laços geográficos entre utilizadores de redes sociais *online* foi outra propriedade utilizada para compreender a homofilia entre utilizadores. O trabalho realizado em [36], onde foi dado destaque à relação geográfica entre os utilizadores e as suas amizades, concluiu que cerca de 1/3 das ligações de amizade numa rede social são independentes da região geográfica.

Outra estratégia de determinação da similaridade entre um par de utilizadores recai na ideia que, se dois utilizadores participam nas mesmas atividades, então existe entre eles uma forma de similaridade [37]. Em particular, a informação associada às atividades do utilizador contribui de forma a ser possível construir um perfil capaz de descrever as suas preferências.

Os perfis de utilizadores que resultam de dados provenientes de redes sociais *online* são geralmente esparsos[16] e pobres uma vez que, quando há campos de preenchimento livre, pode existir uma panóplia de opções possíveis. Devido a este facto, o processo de cálculo da similaridade pode não ser eficaz [38].

Os autores em [32] propõem uma solução híbrida, em que se baseiam não só no conhecimento dos laços sociais existentes entre os utilizadores como também pela análise das atividades em que estão envolvidos. A estratégia passa por três fases: definição de um conjunto de parâmetros para calcular as similaridades entre os utilizadores onde consideram diversos tipos de atividades que o utilizador pode efetuar: tornar-se amigo, juntar-se a um grupo, participar num evento, etc. Nesta fase, utilizam como medida de similaridade o coeficiente de *Jaccard*, que será definido adiante; em segundo lugar, utilizam o coeficiente de *Katz*[17] para calcular a similaridade do grafo que é construído para representar todo o mapa de ligação dos utilizadores; na última fase, juntam os valores determinados nas fases anteriores.

A estratégia em [39], consistiu em criar vetores como forma de representação das diversas caraterísticas pertencentes aos perfis. Para determinar a similaridade basta utilizarem uma medida de diferença entre vetores. Devido à existência de diferentes tipos de campos no perfil (localidade, filmes preferidos, etc.), os autores utilizam um vetor com pesos quando determinam a similaridade para, deste modo, ser possível dar maior importância a determinados campos. A estratégia divide-se em duas fases: utilizar funções de correspondência (*matching*) de *strings* para determinar a similaridade entre campos do vetor. Para esta correspondência podem ser utilizados três tipos – exata (corresponder na totalidade), parcial ou *fuzzy* (envolve lógica na comparação). O resultado desta fase é um vetor de similaridade. No segundo passo, um vetor de pesos é aplicado ao vetor de similaridade de modo a determinar a similaridade final entre um par de vetores. As razões de se utilizar um vetor de pesos deve-se ao facto de alguns campos serem únicos ou então outros poderem estar mal preenchidos. Os pesos foram escolhidos, inicialmente, por intuição e, posteriormente, através de técnicas de regressão até encontrarem o melhor resultado.

Outra alternativa estudada [25] consiste em criar árvores de decisão para determinar o grau de similaridade entre utilizadores. A estratégia divide-se nos seguintes passos: começar por definir um modelo para categorizar as palavras-chave do perfil baseando-se em relações semânticas. O modelo consiste em criar múltiplas árvores de categorização para agregar as palavras-chave semelhantes. Este modelo é definido como "Forest Model". Para validar as palavras-chave, foi utilizado um dicionário com palavras inglesas. Para conseguir agrupar as palavras, foram utilizadas técnicas de semântica utilizando a base de dados *WordNet* para construírem a estrutura das árvores; num segundo passo, é definida a noção de distância entre as palavras-chave das árvores tendo por base as distâncias das palavras. São definidas diversas funções para determinar a similaridade entre pares de utilizadores: caso estejam em níveis próximos trata-se de uma relação forte, caso contrário trata-se de uma relação fraca e o valor da similaridade é inferior. Finalmente, depois de analisarem um conjunto de dados proveniente do *Facebook* concluem que a similaridade entre dois utilizadores que estejam separados por 2

[16] Quantidade grande de zeros ou dados em falta.

[17] Segundo a definição do coeficiente *Katz*, dois utilizadores são reconhecidos como similares se existe um número grande de utilizadores que, por sua vez, são similares tanto ao primeiro utilizador como ao segundo.

níveis pouco difere da similaridade dos utilizadores que estão separados por 3, 4 ou mais níveis na rede social. Outra conclusão que conseguiram determinar foi que um aumento do número de amigos e palavras-chave para um utilizador individual diminui a similaridade média entre ele e os seus amigos.

Outra vertente é a utilização de grafos como representação das relações existentes na rede, onde cada vértice representa um utilizador e as arestas qualquer recurso que una os dois utilizadores. Os autores em [30] consideram duas medidas de similaridade para determinar semelhança entre perfis: similaridade da rede – informações do grafo dos perfis comparados, como por exemplo, o número de amigos em comum tendo em consideração o número de amigos (criando um rácio entre os dois valores, ou seja, se um utilizador tiver um número de amigos em comum em relação ao número total de amigos tem um resultado maior); e similaridade do perfil – consiste em comparar os atributos como "texto". No entanto, ressalvam que devido à falta de dados em perfis efetuam um passo que consiste em inferir alguns desses atributos antes de proceder à comparação.

Em [40], a estratégia é dividida em três fases: a primeira consiste em determinar um valor de similaridade que é calculado através do número total de atributos em comum de cada um dos perfis; no segundo passo, é utilizado um algoritmo para determinar outros valores de similaridade utilizando classificações binárias de alguns atributos (profissão, localização, etc.). Para atributos como os amigos em comum e grupos/comunidades em comum são utilizadas medidas com um maior grau de complexidade (por exemplo, medida *Jaro*, cálculo do *Cosseno* e *Edit distance*). Nesta fase também é determinado um limiar utilizando a média dos valores de similaridade que foram determinados nas fases anteriores. A fórmula para determinar o grau de similaridade final toma em consideração os pesos normalizados e o resultado é compreendido entre 0 e 1. Finalmente, utilizando o valor do limiar é definido se existe similaridade entre os perfis de utilizadores.

Medidas de distância

Determinar a similaridade ou distância entre dois pontos (ou vetores) é requisito obrigatório para diversas tarefas de *data mining* e extração de conhecimento que envolvam cálculo de distâncias. Deste modo, nesta secção serão descritas as medidas de distância utilizadas neste projeto assim como serão dadas definições matemáticas das mesmas.

No que confere aos perfis sociais há que considerar dois tipos de dados: contínuos (numéricos) e categóricos (texto).

Para dados contínuos, a distância *Minkowski* é o método mais usado para determinar a distância entre pontos multivariados. Em particular, a distância de ordem 1 (também chamada de *Manhattan*) e de ordem 2 (também conhecida como Euclidiana) são as duas mais utilizadas.

Para dados categóricos, a tarefa já não é tão simples. A caraterística chave dos dados categóricos é que esses dados podem não ter uma ordem predefinida e pode não ser possível comparar diretamente dois valores distintos. O método mais fácil para comparar dados categóricos é atribuir o valor 1 quando eles coincidem, caso contrário o valor 0. Para dados multivariados, a similaridade entre eles é diretamente proporcional ao número de atributos que coincidem. Esta medida simples é também conhecida como medida *overlap* [41]. Uma

desvantagem desta medida é que não considera a frequência dos valores, considerando apenas se coincide ou não.

Em [42], os autores identificam um conjunto de caraterísticas que consideram ser fundamentais para comparar dados categóricos:

- **Número de atributos** – geralmente as medidas normalizam os dados pelo número de atributos, mas o número de atributos pode afetar a performance dos algoritmos de deteção de *outliers*.
- **Número de valores que cada atributo pode ter** – alguns atributos podem ter poucas opções, mas outros podem ter até centenas de opções. Existem medidas que podem atribuir diferentes graus de importância aos diversos valores (por exemplo, *Eskin*).
- **Distribuição da frequência dos valores de um dado atributo** – alguns atributos podem ter uma distribuição normal no conjunto enquanto outros podem ter outro tipo de distribuição. A medida de similaridade deve atribuir mais importância a valores de atributos que raramente ocorrem, enquanto outras medidas devem dar mais importância aos que ocorrem mais vezes.

Os autores concluíram que não existe uma medida ideal para todos os casos e podem ser, inclusive, utilizadas diversas medidas para diferentes atributos no mesmo problema.

Otimização de performance no cálculo da similaridade

Determinar a similaridade entre perfis em tempo real pode obrigar a utilizar diferentes estratégias dado que, num cenário real, uma base de dados de perfis pode conter milhares ou até milhões de registos. Perante este facto, têm sido investigadas estratégias para que seja possível otimizar o processo de cálculo de similaridade entre perfis. As duas técnicas principais consistem na utilização de técnicas de *clustering* e na utilização do algoritmo *Locality-Sensitive Hashing*.

Em [43], a estratégia passa por calcular previamente *clusters* através do algoritmo *K-Means* para ser possível uma melhor performance e evitarem o cálculo em tempo real utilizando todos os utilizadores presentes no sistema. Deste modo, quando um novo perfil entra no sistema, este é inserido no *cluster* respetivo. Por fim, o sistema obtém as recomendações através dos interesses correspondentes aos utilizadores desse *cluster*. Em [44], os autores também executam testes comparativos de performance entre três algoritmos (*clustering* hierárquico, *fuzzy k-means* e *spectral clustering*) em dados sobre clientes de lojas de *eCommerce*.

Existem diversas técnicas que visam lidar com o desafio de trabalhar com um número elevado de dados, das quais se destacam pelo uso de processos de compressão da informação. Em seguida, serão definidas as técnicas *Locality-Sensitive Hashing*, *Min-Hash* e *Random Hyperplane Hashing* que se englobam neste contexto.

Locality-Sensitive Hashing (LSH) é um algoritmo que visa encontrar as assinaturas (representação do vetor de atributos) que sejam extremamente correlacionadas. O LSH foi introduzido por Indyk e Motwani [45] que propuseram utilizar uma função de *hashing*[18] das assinaturas de uma forma que a probabilidade da existência de colisões fosse diretamente proporcional às suas similaridades. Por exemplo, se duas assinaturas são bastante similares, a probabilidade de haver uma colisão entre elas é muito maior que a probabilidade de haver colisões entre assinaturas diferentes.

Min-Hash é uma técnica baseada numa função de *hashing* de modo a obter uma representação/assinatura comprimida de um conjunto de dados, por exemplo, uma coluna de 1's. A probabilidade de duas assinaturas serem iguais é diretamente proporcional à sua similaridade.

Random Hyperplane Hashing (RHH) – é uma família de funções de LSH que utiliza a similaridade a partir do cosseno do ângulo entre vetores e a distância de *Hamming* como métrica para cálculo da distância entre as *strings* binárias geradas. Quanto maior o valor do cosseno do ângulo entre um par de vetores, menor a distância de *Hamming* entre as *strings* binárias geradas. Neste contexto, essas *strings* binárias representam um perfil de utilizador cuja similaridade é passível de ser medida utilizando a distância de *Hamming*.

Em [8], a estratégia dos autores começa por representar todos o utilizadores e os seus interesses como uma matriz grande e esparsa. O passo seguinte consiste em aplicar as técnicas supracitadas (*Min-Hashing* e *Locality-Sensitive Hashing*) de forma a encontrar utilizadores que sejam potencialmente similares a um dado perfil que se pretenda comparar (denominado perfil de entrada). Estes utilizadores são apelidados de candidatos. Assim que os candidatos estejam definidos procede-se ao cálculo das distâncias de similaridade entre o perfil de entrada e os candidatos evitando, desta forma, computar o valor de similaridade para todos os utilizadores. Este processo pode ser visto na Figura 2.3.

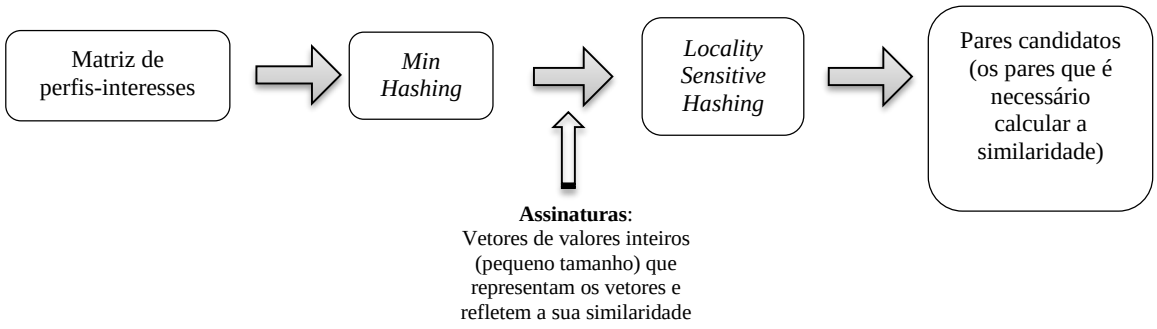


Figura 2.3 - Determinação de candidatos utilizando as técnicas Min-Hash e Locality-Sensitive-Hashing

Os autores em [6], começam por representar as caraterísticas do utilizador como um vetor. A similaridade é medida através do cálculo da distância entre vetores (euclidiana e o valor do cosseno do ângulo entre vetores). Após o processo de criação e normalização dos campos do vetor que representa o perfil, este é indexado numa base de dados utilizando a função *Random Hyperplane Hashing*. Utilizam esta medida porque não é afetada

[18] Uma função de *hashing* é um algoritmo que mapeia dados de comprimento variável para dados de comprimento fixo.

com a dimensionalidade do vetor. Com a utilização desta técnica, os autores concluíram que o custo de computação reduzia significativamente.

2.2 Sistemas de recomendação

Como referido anteriormente, uma área onde a similaridade entre perfis é utilizada diz respeito aos sistemas de recomendação.

Em [46], os autores, baseando-se em estudos de psicologia social, afirmam que recomendações de pessoas similares de alguma forma, por exemplo, familiares, amigos, utilizadores com interesses em comum, mesma comunidade, etc., têm um grande impacto na decisão dos utilizadores quando existe um cenário de decisão (por exemplo, na compra de um produto). Neste caso, o sistema de recomendação tem como foco detetar perfis semelhantes ao utilizador para visualizar opiniões ou avaliações e, deste modo, influenciar na tomada de decisão. Um facto curioso que os autores referem é que recomendam utilizar informação geográfica na geração de recomendações, visto que os utilizadores tendem a confiar em perfis da mesma localização.

Segundo [47], os sistemas de recomendação baseiam-se, essencialmente, em duas arquiteturas:

- **Sistemas baseados em conteúdo** (*Content-based*) – analisa propriedades dos *items* recomendados. Por exemplo, se um utilizador da plataforma *Netflix*[19] visualizou muitos filmes de *cowboys*, então o sistema pode recomendar um filme presente na base de dados que tenha sido previamente classificado como sendo do género *cowboys*.
- **Filtragem colaborativa** (*Collaborative Filtering* ou CF) – recomendam *items* baseando-se em medidas de similaridade entre utilizadores e/ou *items*. Os *items* recomendados ao utilizador são os preferidos pelos utilizadores similares. Este sistema de recomendação pode utilizar diversas medidas de similaridade (referidas nas secções anteriores) e em *clusterings*.

Estas duas arquiteturas irão ser abordadas com mais detalhe nas secções seguintes e, em seguida, irão ser descritos diversos conceitos importantes nos sistemas de recomendação.

Considerem-se duas entidades: **Utilizadores** e ***Items*** (podem representar produtos, categorias, filmes, interesses, entre outros). Geralmente, o resultado destes sistemas consiste em recomendar *items* aos utilizadores.

[19] https://www.netflix.com

Matriz de utilidade

Os utilizadores têm preferências por determinados *items* e essas preferências devem ser extraídas a partir dos dados. Os dados, por sua vez, são representados como uma matriz de utilidade, dando a cada par utilizador-*item* um valor que representa o grau de preferência (*rating*) do utilizador para determinado *item*.

Esta matriz, geralmente, é esparsa. O que significa que muitas entradas são de valor "desconhecido", ou seja, implica que não existe informação explícita sobre as preferências do utilizador para determinado *item*.

Na Figura 2.4, pode-se observar um exemplo de uma matriz de utilidade onde são representadas avaliações de utilizadores dadas a diversos filmes. A escala utilizada foi de 1-5, em que 5 é o valor de maior importância. Os valores em branco representam a situação em que o utilizador não avaliou o filme. As colunas (HP1, HP2, HP3, TW, SW1, SW2, SW3) representam os filmes, e os utilizadores são representados nas linhas (A, B, C e D).

	HP1	**HP2**	**HP3**	**TW**	**SW1**	**SW2**	**SW3**
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Figura 2.4 - Exemplo de matriz de utilidade que representa avaliações de filmes com escala de 1-5.

O objetivo do sistema de recomendação consiste em determinar os valores em branco da matriz.

Existem duas abordagens no que diz respeito ao preenchimento desta matriz:

1. Propôr aos utilizadores que avaliem *items*. Por exemplo, no caso dos filmes.
2. Efetuar inferências dos utilizadores a partir do seu comportamento. Por exemplo, se um cliente pesquisa por um determinado produto diversas vezes pode-se inferir que o utilizador tem interesse pelo mesmo.

Seguidamente, irão ser descritas as duas arquiteturas mais utilizadas em sistemas de recomendação.

2.2.1 Recomendações baseadas no conteúdo

Para que seja possível perceber o âmbito e funcionamento desta arquitetura é necessário introduzir alguns conceitos.

Perfis de *items* – o objetivo é representar as caraterísticas importantes do *item* num vetor. Em casos simples, o perfil consiste em diversas caraterísticas facilmente identificáveis. Por exemplo, considerando as caraterísticas de um filme que pode ser relevante para um sistema de recomendação, o perfil poderia ser o conjunto de atores, o género e o ano. Podem existir em casos que outros atributos do *item* sejam utilizados para

criar caraterísticas do mesmo. Por exemplo, a descrição do fabricante de um dado produto, tamanho do ecrã, etc.

Perfis de utilizadores – além de criar vetores que representam os *items*, também é necessário criar vetores que representem os utilizadores com os mesmos componentes que descrevem as suas preferências, dados demográficos, etc.

Como descrito anteriormente, para relacionar o conteúdo entre os *items* e os utilizadores existe a matriz de utilidade. Uma vez criados os perfis dos *items* e utilizadores, é necessário estimar o grau que um utilizador iria preferir um *item* através do cálculo da distância do cosseno do ângulo entre os vetores do *item* e do utilizador.

2.2.2 Filtragem colaborativa

Em vez de utilizar caraterísticas dos *items* para determinar a sua similaridade, esta arquitetura baseia-se na similaridade das avaliações do utilizador para dois *items*. Isto é, em vez de se considerar o vetor que representava o *item*, são utilizadas as colunas na matriz de utilidade. Adicionalmente, em vez de atribuir um vetor de perfil aos utilizadores, estes são representados pelas linhas na matriz de utilidade.

Os utilizadores são similares caso pontuem os mesmos *items* com valores semelhantes, ou seja, se os seus vetores estão próximos de acordo com alguma métrica de distância (por exemplo, *Jaccard* ou valor do cosseno do ângulo entre os vetores). O processo de recomendação para um utilizador é feito, então, através da observação dos utilizadores que lhe são mais similares, e as recomendações são os *items* preferidos destes utilizadores. O processo de identificação de utilizadores similares a um dado utilizador e recomendar o que os utilizadores similares preferem é definido como filtragem colaborativa.

Os autores em [48], destacam diversos desafios neste tipo de arquitetura:

- Matriz esparsa – a matriz que representa os utilizadores e *items* é, geralmente, bastante esparsa, pelo que determinar recomendações torna-se um desafio. Existem diversas técnicas que visam resolver este problema, tais como a redução da dimensionalidade, como por exemplo o *Singular Value Decomposition* (SVD) [49], ou então remover utilizadores ou *items* que não sejam representativos para reduzir a dimensionalidade diretamente da matriz utilizadores-*items*. Outra técnica de redução da dimensionalidade é o *Principal Component Analysis* (PCA). É importante ter presente que a redução de dimensionalidade pode levar a perda de informação e, como consequência, diminuir a eficácia das recomendações.
- Escalabilidade – outros dos desafios presentes neste cenário é que é necessário trabalhar com uma larga quantidade de dados, pelo que a capacidade de processamento em tempo real pode ser um problema. Os autores em [48] referem que técnicas de redução da dimensionalidade podem ser uma solução (SVD, por exemplo), mas mesmo assim há possibilidade do custo de processamento ser elevado.
- Sinónimos – refere-se à tendência a existirem *items* que são iguais ou semelhantes mas que possuem nomes diferentes. Este problema faz com que a eficácia das recomendações diminua. Técnicas como

SVD, particularmente *Latent Semantic Indexing* (LSI), sejam capazes de lidar com este problema. Em [50], os autores indicam o uso de técnicas de *data mining* tal como utilização de algoritmos de *clustering*, que iria permitir resolver o problema da redundância e ambiguidade das *tags* facilitando, deste modo, o sistema de recomendação.

- "Área cinzenta" – este problema aparece quando as preferências de um utilizador não se inserem em nenhum grupo, ou seja, não concordam nem discordam das opiniões dos grupos já formados, pelo que o algoritmo pode não funcionar muito bem. Uma solução possível é a inclusão de pesos aos conteúdos de forma a ser dada mais importância a determinado conteúdo em prol de outro. Mas definir os pesos torna-se ainda outro desafio.
- Ataques *Shilling* – quando não existem muitas avaliações dos produtos, as pessoas podem distorcer as avaliações dando, por exemplo, excelentes recomendações apenas aos seus produtos e avaliações negativas a produtos de concorrentes. Os autores em [48], referem um estudo onde utilizam uma compreensiva abordagem a este problema através da remoção de efeitos globais no passo de normalização dos dados do CF baseado em vizinhos, e trabalhar apenas com efeitos residuais dos efeitos globais para escolher os vizinhos. Conseguiram, com esta estratégia, melhorar a performance do algoritmo CF num conjunto de dados do *Netflix*[20].
- *Cold Start* – ocorre, geralmente, quando não existem dados suficientes para que o sistema consiga inferir *ratings* de *items* através de dados de outros utilizadores, uma vez que não é possível recolher informação suficiente.

Criação de *clusters* de utilizadores e *items*

Detetar similaridades entre os próprios *items* ou mesmo entre os próprios utilizadores não se afigura como uma tarefa trivial. A razão deve-se à escassa informação existente sobre os pares utilizador-*item* na matriz de utilidade (esparsa). Um exemplo prático: numa loja de venda de produtos, mesmo no caso em que dois *items* pertencessem à mesma categoria, era provável que poucos utilizadores comprassem ou avaliassem ambos. Da mesma forma, mesmo se dois utilizadores gostassem ambos de uma dada categoria ou categorias de produtos, podiam não ter comprado nenhum produto em comum.

Uma abordagem para lidar com este cenário é criar *clusters* de *items* e/ou utilizadores [47]. Inicialmente, deverá ser criado um número pequeno de *clusters*, devendo optar-se por uma abordagem hierárquica, onde é suficiente obter a informação dos vários *clusters* sem estarem agrupados. Tendo os *items* agrupados de certa medida, é possível alterar a matriz de utilidade para que as colunas representem *clusters* de *items*. A entrada para um dado utilizador U e o *cluster* C deverá ser a avaliação média que U deu aos membros do *cluster* C. No caso de um utilizador não ter feito nenhuma avaliação a qualquer membro do *cluster* C, então a entrada fica vazia.

Por sua vez, no passo seguinte, pode-se então rever a matriz de utilidade para criar *clusters* de utilizadores utilizando uma determinada medida de similaridade. Seguidamente, deve-se alterar a matriz de utilidade de

[20] *Netflix prize*, http://www.netflixprize.com/

forma que as linhas correspondam aos *clusters* de utilizadores, tal como as colunas representam os *clusters* de *items*. Da mesma forma que o primeiro passo, a entrada para um utilizador deverá ser calculada através da média das avaliações dos utilizadores no *cluster*. Finalmente, este processo pode ser repetido diversas vezes caso se pretenda até ser encontrado intuitivamente um número razoável de *clusters* de cada tipo.

2.2.3 Outras estratégias

Contrariamente aos sistemas de recomendação tradicionais, em [51] os autores propõem a utilização dos interesses das pessoas em vez do histórico de comportamentos numa aplicação particular. Os autores propõem, então, perfilar uma pessoa utilizando as *tags* pessoais assim como as *tags* da rede social. Em vez de um vetor de avaliações de *items*, cada utilizador é definido como um conjunto de palavras e respetivos pesos. Os pesos representam a importância da *tag* específica.

Em vez de inferir a relação entre as preferências do utilizador e o atributo do *item* a partir dos dados de avaliação como os tradicionais sistemas de recomendação, os autores propõem obter a relação através da análise desses atributos que são representados por *tags*. Desta forma, a utilização da relação baseada em *tags* abstrai a semântica entre utilizadores e *items*, evitando os problemas comuns nos sistemas de recomendação tradicionais. Para capturarem informações sobre preferências do público, ou seja, para responder a perguntas do tipo "qual a percentagem de pessoas que gostaram do tópico A que também gostaram do tópico B?", criam uma matriz *tag-tag*. Esta matriz regista a relevância entre um par de *tags* do utilizador (atributos do utilizador) e *tags* do conteúdo (atributos do conteúdo).

Finalmente, os autores em [48] concluem que é desejável que a abordagem de um sistema de recomendação seja fácil de implementar mas que consiga, ao mesmo tempo, lidar com problemas comuns enunciados anteriormente. Referem também que, na fase de testes (ou treino), devem ser utilizados conjuntos de dados reais, se possível, uma vez que os dados artificiais não costumam ser fiáveis.

2.3 Considerações finais

Através deste capítulo foi possível efetuar um enquadramento global do projeto pelo que, temas como redes sociais, perfil social, similaridade entre perfis e sistemas de recomendação foram conceitos referenciados nas diversas secções. Foi descrita a importância das redes sociais relativamente ao tema de similaridade entre perfis dando-se destaque à rede social *Facebook*. Posteriormente, foram definidas diversas métricas de distância entre vetores para ser possível determinar a similaridade entre perfis. Determinar este valor em tempo real pode obrigar a utilizar diferentes estratégias dado que, num cenário real, uma base de dados de perfis pode conter milhares ou até milhões de registos. Perante este facto, para otimizar o processo de cálculo de similaridade entre perfis alguns autores propõem a utilizam de técnicas que utilizem o algoritmo *Locality-Sensitive Hashing*. Finalmente, é definido em que consiste um sistema de recomendação e descritas as duas principais arquiteturas: sistemas baseados em conteúdo e filtragem colaborativa.

3. Similaridade entre perfis sociais

"*There are lies, damned lies, and statistics*."
Mark Twain

Este capítulo visa descrever a estratégia seguida durante a criação do protótipo onde o objetivo baseia-se em determinar a similaridade entre perfis sociais. Inicialmente, irão ser descritas as propriedades relativamente aos dados utilizados tais como a proveniência, as dificuldades na sua obtenção, processamento e tratamento necessário. Seguidamente, irão ser listadas diversas informações estatísticas sobre os atributos dos dados provenientes de uma rede social. Finalmente, nas secções seguintes irão ser descritos os passos efetuados com vista a concretizar as funcionalidades do protótipo.

3.1 Fontes de dados

Os dados recolhidos foram provenientes da rede social *Facebook*.

Para que seja possível recolher dados de utilizadores desta rede é necessário criar uma aplicação que interaja com a mesma e, além disso, os utilizadores têm ainda que aceitar as permissões sobre todos os dados que a aplicação poderá recolher.

Os dados utilizados neste projeto foram gentilmente cedidos pela empresa *Ubiprism*[21].

Nos pontos seguintes irão ser vistos os problemas detetados na obtenção dos dados, os procedimentos necessários para limpar e transformar os dados em bruto.

3.1.1 Formato dos ficheiros

Os ficheiros fornecidos pela empresa onde se encontravam os dados de perfis sociais encontravam-se em dois formatos diferentes:

- JSON (*JavaScript Object Notation*[22]) – trata-se de um formato para troca de dados entre sistemas. É de fácil interpretação e simples de utilizar uma vez que é independente de qualquer linguagem.
- YAML (YAML *Ain't Markup Language*[23]) – é um formato de serialização (codificação) de dados, legíveis por humanos, inspirado em linguagens como o XML, C, *Python* e *Perl*.

Exemplos dos dados originais (provenientes de ficheiros *json* e *yml*) podem ser visualizados na Figura 3.1.

[21] http://www.beubi.com / http://www.ubiprism.com
[22] https://tools.ietf.org/html/rfc4627, consultado em 17 de Maio de 2015
[23] http://www.yaml.org/spec/1.2/spec.html, consultado em 17 de Maio de 2015

Devido aos diferentes tipos de formatos de ficheiro como entrada dos dados, foi necessário criar dois tipos de "loaders", cada um responsável pelo formato respetivo.

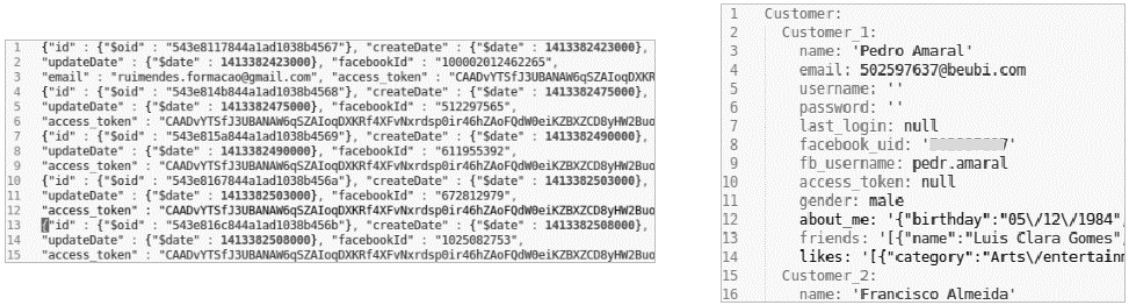


Figura 3.1 - Exemplos de dados provenientes dos ficheiros originais (json e yml)

3.1.2 Estrutura dos dados

Como supracitado, os atributos contidos nos ficheiros são correspondentes aos perfis recolhidos da rede social *Facebook*, pelo que é importante perceber a que tipo de informação correspondem. Na Tabela 3.1 estão representados os principais atributos sendo feita uma descrição de cada um. A lista completa de todos os atributos pode ser consultada em anexo.

Atributo	Descrição
facebookId	Identificador único na rede social *Facebook*.
fbName	Nome do utilizador.
gender	Género.
birthday	Data de nascimento.
fbFriends	Lista de amigos que o utilizador possui.
fbNumberOfFriends	Número de amigos que o utilizador possui.
fbGroups	Grupos a que o utilizador pertence.
fbDevices	Dispositivos móveis associados ao perfil.
fbPages	Páginas que o utilizador fez *Like*.
fbMaritalStatus	Representa o estado civil.
fbInterests	Representa os interesses do utilizador categorizados por diversos temas (filmes, música, livros, etc.).
fbEducation	Campo onde o utilizador pode colocar o histórico a nível académico, desde o secundário, a curso superior ou outros cursos que tenha realizado.
fbWork	Histórico profissional.

Tabela 3.1 - Principais atributos e descrições presentes nos ficheiros (fonte de dados)

3.1.3 Tipo de dados

Antes de descrever o processo de limpeza de dados realizados, é importante definir os diversos tipos de dados que podem existir.

O tipo define se o atributo representa quantidades, sendo então denominado quantitativo ou numérico, ou qualidades, sendo então designado de qualitativo, simbólico ou categórico pois os seus valores podem ser associados a categorias.

Exemplos de conjuntos de valores qualitativos são {pequeno, médio, grande} e {informática, matemática, português}. Com os primeiros podem estabelecer-se relações de ordem mas não podem ser realizadas operações aritméticas.

Os atributos quantitativos são numéricos, como no conjunto de valores {12, 23, 6}. Os valores de um atributo quantitativo são ordenados e podem ser utilizados em operações aritméticas. Os valores quantitativos podem ser ainda contínuos ou discretos. Os atributos contínuos podem assumir um número infinito de valores. Exemplos de atributos deste tipo são atributos que representam pesos, tamanhos ou distâncias. Os atributos discretos contêm um número finito ou infinito contável de valores. Um caso especial deste tipo de atributos são os atributos binários (ou *booleanos*), que apresentam apenas dois valores, como 0/1, sim/não, ausência/presença e verdadeiro/falso.

3.1.4 Limpeza e transformação dos dados

Apesar dos dados provenientes dos ficheiros originais apresentarem-se, desde início, com alguma estrutura predefinida, houve necessidade de se proceder a métodos de limpeza e transformação. Por exemplo, o campo "fbDevices" poderia conter o mesmo conteúdo mas com diferente ordem (*iPhone, Android* e *Android, iPhone*). Outro exemplo é o campo "fbPages", onde foi necessário extrair a categoria da página de modo a obter o interesse do utilizador. As transformações necessárias irão ser descritas pormenorizadamente nos próximos parágrafos.

Numa primeira fase, houve necessidade de organizar os diversos atributos por diversos tipos, uma vez que alguns representavam valores numéricos, outros representavam categorias ou então eram identificadores únicos, ou seja, atributos que a função é de identificar aquele utilizador (por exemplo, "facebookId").

A organização escolhida foi a seguinte:

- **Numérico**: *fbNumberOfFriends*
- **Categórico**: *fbMaritalStatus, gender, fbReligion, fbPolitical*
- **Data**: *birthday*
- **Texto**: *facebookId, email, fbName*
- **Matriz** (***array***): *fbFriends, fbGroups, fbEducation, fbFamily, fbInterests, fbPages, fbWork, fbDevices*

Para o atributo do tipo "Data" houve necessidade de converter num formato que facilitasse a leitura por parte do humano, uma vez que as datas encontravam-se no formato de *timestamp*[24].

Outra transformação necessária foi realizada nos campos que pertenciam ao tipo "Matriz" e que continham um "array" vazio, esse valor foi substituído pela expressão "NaN" (valor que representa que o campo não foi preenchido) para, desta forma, facilitar as operações matemáticas na secção 3.1.5. Ainda em relação a este tipo de dados, houve necessidade de processar a ordem das respostas provenientes dos atributos para que não surgissem dados redundantes. Por exemplo, no atributo *fbDevices* (que representa os dispositivos eletrónicos associados à conta do utilizador), respostas como "iPhone,Android" e "Android,iPhone" representam o mesmo, mas seriam consideradas respostas distintas.

Um problema relativamente aos campos de preenchimento livre, como por exemplo o *fbWork* (que representa o histórico profissional) deveu-se à possibilidade da resposta estar em diferentes idiomas (por exemplo, "internship" e "estagiário"). Foi necessário, também, um processamento de termos que têm o mesmo significado mas são escritos de forma diferente, por exemplo, "Soft. Engineer" e "Software Eng.".

Seguidamente, irão ser listadas diversas estatísticas dos dados, tais como média, taxa de preenchimento, entre outras.

3.1.5 Estatísticas

Tendo em conta a organização dos dados lista acima, foram determinadas diferentes informações, das quais se destacam:

- **Numérico**: Média, Máximo, Mínimo, Desvio Padrão, Total, Taxa de preenchimento, Nº total de valores em falta e percentis
- **Categórico**: Total, Taxa de preenchimento, Nº total de valores em falta, Lista de valores únicos, Contagem por categoria
- **Data**: Total, Taxa de preenchimento, Nº total de valores em falta, Lista de valores únicos, Contagem por data
- **Texto**: Total, Taxa de preenchimento, Nº total de valores em falta
- **Matriz** (***array***): Total, Taxa de preenchimento, Nº total de valores em falta

Nas tabelas seguintes são listadas as informações sobre alguns atributos. No entanto, a lista completa das informações estatísticas dos atributos encontra-se em anexo.

[24] *Timestamp* representa uma sequência de carateres ou informação codificada com o objetivo de identificar quando um certo evento ocorre, geralmente, sendo disponibilizando a data e altura do dia, por vezes especificando mesmo a fração de segundo [66].

gender

Este campo representa o género do utilizador. A taxa de preenchimento é bastante considerável (95,79%).

Total de dados	3702
Dados em falta	156
Taxa de preenchimento	95.79%
Moda	*male* (2018)

Tabela 3.2 - Estatísticas do atributo "gender"

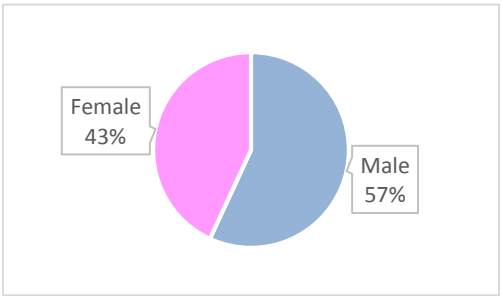


Figura 3.2 - Gráfico do atributo "gender"

fbFriends

Lista de amigos que o utilizador possui na rede social *Facebook*. Como se pode observar, a taxa de preenchimento deste campo é extremamente reduzida. A razão deve-se ao facto de não ser possível recolher os dados devido à política do *Facebook*, ou seja, não é permitido recolher os dados de amigos em 2º grau.

Total de dados	3702
Dados em falta	3686
Taxa de preenchimento	0.38%

Tabela 3.3 - Estatísticas do atributo "fbFriends"

fbPages

Páginas que o utilizador fez "Like", ou seja, são páginas das quais o utilizador demonstrou interesse. Estas páginas encontram-se associadas a uma categoria (previamente definida pela rede social *Facebook*). A sua taxa de preenchimento é elevada, como se pode observar na Tabela 3.4.

Na Figura 3.3 é possível visualizar as categorias de "Likes" que estão no topo das preferências.

Total de dados	3702
Dados em falta	542
Taxa de preenchimento	85.36%

Tabela 3.4 - Estatísticas do atributo "fbPages"

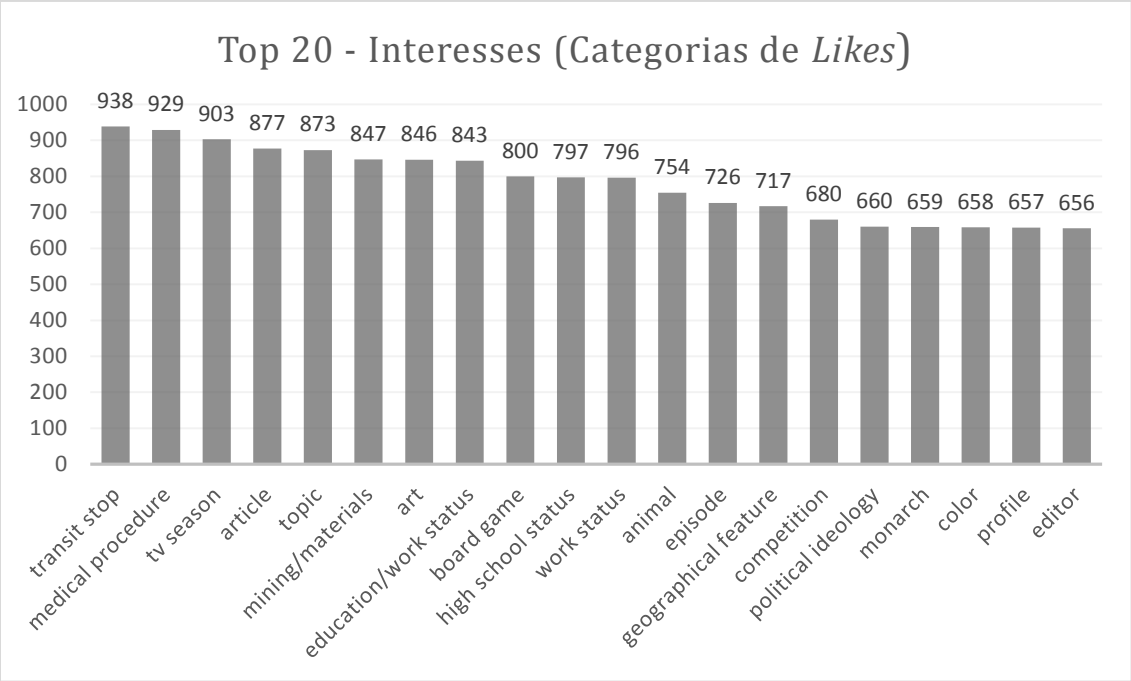


Figura 3.3 - Gráfico Top20 de Interesses (Categorias dos Likes)

fbInterests

Este campo representa os interesses do utilizador categorizados por diversos temas (filmes, música, livros, etc.).

Total de dados	3702
Dados em falta	3364
Taxa de preenchimento	9.13%

Tabela 3.5 - Estatísticas do atributo "fbInterests"

O atributo *fbInterests* representa os interesses gerais do utilizador. Pode-se observar, através da Tabela 3.5, que a taxa de preenchimento é relativamente baixa (9.13%) e, além desse facto, contém respostas de preenchimento livre tornando-se, portanto, de difícil processamento e categorização.

3.1.6 Definição da estratégia

Numa primeira fase, decidiu-se que a determinação de perfis similares a um dado perfil iria basear-se nos interesses dos utilizadores. Após uma análise dos diversos atributos dos dados tendo em consideração as informações fornecidas nas tabelas anteriores, optou-se por considerar o atributo *fbPages* para representar os

perfis dos utilizadores. Este campo representa as páginas que o utilizador demonstrou interesse, fazendo "Like" nas mesmas.

O atributo *fbPages* encontra-se, inclusive nos dados originais, associado a categorias (ex.: *camera/photo*, *computers/technology*, *arts/entertainment/nightlife*, entre outras). Estas categorias podem pertencer a diferentes níveis de hierarquia (existem dois níveis). Uma vez que nem todas as páginas contêm um segundo nível considerou-se, para este projeto, apenas o primeiro nível. Estas categorias são definidas previamente pela rede social *Facebook* e a lista completa pode ser consultada em anexo.

O processamento dos dados relativamente a este atributo consistiu em agrupar as categorias dos diversos "Likes" adicionando um contador às mesmas, ou seja, processar individualmente os "Likes" e, caso essa categoria seja repetida, basta adicionar uma unidade. O resultado final consistiu num vetor em que cada elemento corresponde a uma categoria e o seu valor corresponde ao número de "Likes" que o utilizador fez na mesma.

Perante estes factos e, como supracitado, a estratégia consistiu em utilizar o atributo *fbPages* para representar os interesses dos utilizadores. A fase seguinte consistiu construir uma matriz que representasse os perfis dos utilizadores com os interesses respetivos (denominada matriz perfis-interesses onde as linhas correspondem aos perfis e as colunas aos interesses).

3.2 Construção de matrizes

3.2.1 Matriz Perfis–Interesses

Esta matriz visa representar perfis de utilizadores através dos seus interesses.

Como referido na secção anterior, esta matriz é construída efetuando uma contagem das categorias por cada "Like" do atributo *fbPages*. Posteriormente, as linhas da matriz são normalizadas para que a soma dos valores por linha esteja compreendida entre 0 e 1. Deste modo, os pesos que representam a importância dos interesses por cada utilizador apresentam a mesma escala para comparação. No entanto, o valor zero não significa que o utilizador não tenha interesse, mas apenas que não se detém informação sobre ele, ou seja, o utilizador não fez nenhum "Like" nessa categoria.

Esta matriz irá ser utilizada para determinar os diferentes *clusters* de utilizadores. Na Figura 3.4, pode ser visto um exemplo do resultado da matriz Perfis-Interesses.

	actor/director	aerospace/defense	album	amateur sports team	animal	animal breed	app	...	video game
517	0.047619	0.0	0.25	0.000000	0	0.000000	0	...	0.00
518	0.023810	0.0	0.00	0.000000	0	0.000000	0	...	0.00
519	0.071429	0.0	0.00	0.000000	0	0.000000	0	...	0.00
520	0.000000	0.0	0.00	0.000000	0	0.000000	0	...	0.00
522	0.071429	0.0	0.00	0.333333	0	0.000000	0	...	0.00
523	0.119048	0.0	0.00	0.000000	0	0.000000	0	...	0.00
524	0.000000	0.0	0.00	0.166667	0	0.000000	0	...	0.00
525	0.000000	0.0	0.25	0.000000	0	0.000000	1	...	0.00
526	0.023810	0.0	0.00	0.000000	0	0.000000	0	...	0.00

Figura 3.4 - Matriz Perfis-Interesses (Exemplo)

3.2.2 Correlação entre Interesses

Para que fosse possível identificar as correlações entre os diversos interesses, ou seja, responder à questão de "Se o utilizador mostrou preferência por um determinado interesse A, que outro ou outros interesses poderão ser do seu agrado?", foi criada uma matriz (denominada matriz Interesses -Interesses). Um exemplo da matriz de correlação entre os interesses pode ser vista na Figura 3.6.

Utilizando os valores da matriz Perfis-Interesses são determinados os valores de correlação dos pares das colunas, uma vez que estas representam os interesses. Para determinar o valor da correlação é utilizado o método de correlação de *Pearson*, que será definido em seguida.

Coeficiente de correlação linear de *Pearson*

A intensidade da associação linear existente entre as variáveis pode ser quantificada através do chamado coeficiente de correlação linear de *Pearson*.

Sejam $\boldsymbol{x}$ e $\boldsymbol{y}$ dois vetores e $\bar{x}$ e $\bar{y}$ as respetivas médias. Então, o coeficiente de correlação de *Pearson* é definido como:

$$Corr(\boldsymbol{x},\boldsymbol{y}) = \frac{\sum_i (x_i-\bar{x})\,(y_i-\bar{y})}{\sqrt{\sum (x_i-\bar{x})^2}\sqrt{\sum (y_i-\bar{y})^2}} = \frac{\langle \boldsymbol{x}-\bar{\boldsymbol{x}},\boldsymbol{y}-\bar{\boldsymbol{y}}\rangle}{\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|\|\boldsymbol{y}-\bar{\boldsymbol{y}}\|}, \quad (1)$$

onde $-1 \leq Corr(\boldsymbol{x},\boldsymbol{y}) \leq 1$.

Neste contexto, os vetores $\boldsymbol{x}$ e $\boldsymbol{y}$ representam dois interesses e o i um perfil.

Na Figura 3.5, é possível verificar que os valores positivos e próximos de 1 são positivamente correlacionados enquanto valores próximos de -1 indicam correlação negativa. Ou seja, no exemplo em que $r = 1$, caso o perfil esteja interessado no Interesse $\boldsymbol{x}$ é bastante provável que também goste do Interesse $\boldsymbol{y}$. No caso em que $r = 0$, não é possível retirar nenhuma conclusão entre os interesses $\boldsymbol{x}$ e $\boldsymbol{y}$.

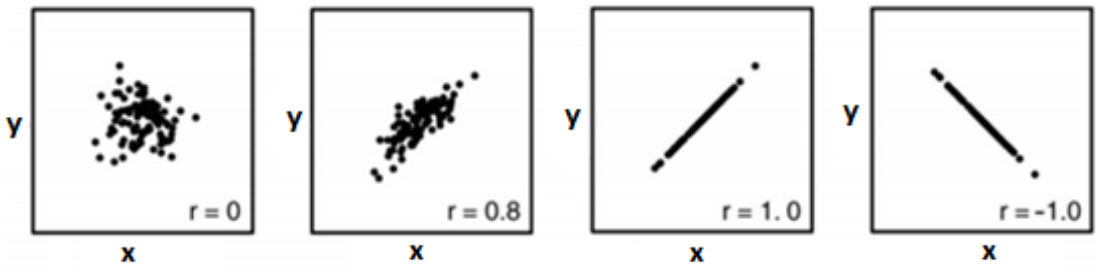


*Figura 3.5 - Representações do coeficiente de Pearson em diversos cenários entre dois interesses (**x** e **y**).*

Na Tabela 3.6 - Tipos de correlação para o intervalo de valores possível *[52]*, constam os diversos tipos de correlação existentes nos diversos intervalos possíveis.

Coeficiente de Correlação	Tipo de Correlação
-1	**Perfeita Negativa**
-0.8	**Fortemente negativa**
-0.5	**Moderadamente Negativa**
-0.2	**Ligeiramente Negativa**
0	**Sem associação**
0.2	**Ligeiramente Positiva**
0.5	**Moderadamente Positiva**
0.8	**Fortemente Positiva**
1	**Perfeita Positiva**

Tabela 3.6 - Tipos de correlação para o intervalo de valores possível [52].

É necessário observar que não se verificar correlação linear, não significa que não se verifique outro tipo de correlação, por exemplo, exponencial. E, também, qualquer que seja a correlação verificada, correlação não significa causalidade [53].

	actor/director	aerospace/defense	album	amateur sports team	animal	animal breed	app
actor/director	1.000000	0.007394	0.083749	0.056807	0.226311	0.050650	-3.284891e-03
aerospace/defense	0.007394	1.000000	0.079110	-0.016017	-0.006131	0.067409	-1.748102e-02
album	0.083749	0.079110	1.000000	0.036566	0.075580	-0.029157	-1.496518e-02
amateur sports team	0.056807	-0.016017	0.036566	1.000000	0.182041	0.024636	-3.132545e-02
animal	0.226311	-0.006131	0.075580	0.182041	1.000000	-0.003954	-6.494876e-03
animal breed	0.050650	0.067409	-0.029157	0.024636	-0.003954	1.000000	-1.127517e-02
app	-0.003285	-0.017481	-0.014965	-0.031325	-0.006495	-0.011275	1.000000e+00

Figura 3.6 - Matriz de correlação entre interesses (exemplo)

Através desta matriz é possível retirar informações úteis entre as relações dos interesses, que serão utilizados no capítulo referente aos resultados.

3.3 Ranking de perfis similares a um dado perfil

Um dos objetivos deste trabalho consiste em determinar os perfis similares dado um perfil de utilizador. Esta secção descreve os passos efetuados para atingir esse objetivo.

Considerando que a matriz Perfis-Interesses já se encontra criada, esta fase consiste em determinar, através de diferentes métricas, as distâncias entre os diversos pares de perfis, ou seja, calcular todas as distâncias entre os vetores de interesses do perfil escolhido e todos os restantes vetores existentes na matriz.

Um passo importante a referir é que, antes do cálculo da similaridade entre os perfis, apenas se consideram os utilizadores que tenham os mesmos interesses, ou seja, as colunas preenchidas correspondentes aos interesses do perfil a comparar. Considere-se que no exemplo da Figura 3.4, o perfil 517 tinha apenas os interesses *actor/director*, *album* e *animal*. Então, a matriz da qual se iam considerar os dados para determinar a similaridade de perfis seria uma sub-matriz que continha apenas essas colunas. Através desta técnica, o custo de processamento é reduzido significativamente além de garantir que os valores que correspondiam a zero do perfil a comparar não influenciariam na fórmula de cálculo. Para determinar os valores de similaridade entre os vetores que representam os perfis, podem ser utilizadas diversas métricas, das quais se destacam: distância euclidiana, distância de *Manhattan*, coeficiente de *Jaccard*, coeficiente de correlação de *Pearson* e cosseno do ângulo.

Finalmente, tendo a lista com os valores da similaridade entre os perfis, basta escolher o número de perfis similares pretendidos (considere-se n), ordenar a lista com as distâncias e devolver os n perfis.

Em seguida, serão definidas as métricas utilizadas.

Medida de similaridade

Considere-se um conjunto de pontos, chamado espaço. Uma medida de distância $d(\boldsymbol{x}, \boldsymbol{y})$ neste espaço considera dois pontos como argumentos e devolve um número real. Também terá que satisfazer os seguintes axiomas [47]:

1. $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$ (distâncias não-negativas).
2. $d(\boldsymbol{x}, \boldsymbol{y}) = 0$ *se e só se* $\boldsymbol{x} = \boldsymbol{y}$ (as distâncias são positivas, exceto para a distância com um ponto e ele próprio).
3. $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$ (distância simétrica).
4. $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{z}, \boldsymbol{y})$ (desigualdade triangular).

Contextualizando ao tema do projeto, $\boldsymbol{x}$ e $\boldsymbol{y}$ representam dois perfis (conjunto de interesses).

Distância Euclidiana

É a medida de distância mais comum e também é conhecida como L2-*Norm*. Pode ser definida como:

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{2}$$

onde $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ e $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$.

Distância de *Manhattan*

Outra medida de distância comum é a *Manhattan* ou também conhecida como *L1-Norm*.

A distância entre dois pontos é a soma das diferenças em cada dimensão. É chamada de distância *Manhattan* porque é a distância que um indivíduo teria que viajar entre pontos nas ruas da cidade tal como em *Manhattan*. Pode ser definida como:

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n}|x_i - y_i|, \tag{3}$$

onde $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ e $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$.

Distância do Cosseno

Esta métrica visa considerar as direções representadas pelos pontos num espaço multidimensional, pelo que não há distinção do comprimento entre o par de vetores. Então, a distância entre dois pontos é o valor do cosseno do ângulo entre os dois vetores. Este ângulo será um valor entre 0 e 180 graus.

Sendo dados dois vetores $\boldsymbol{x}$ e $\boldsymbol{y}$, o cosseno do ângulo entre eles é produto interno $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ dividido pelas *L2-Norm* de $\boldsymbol{x}$ e $\boldsymbol{y}$ (isto é, as suas distâncias Euclidianas a partir da origem), como se pode visualizar na fórmula seguinte:

$$CosSim(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} = \frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} \qquad (4)$$

Coeficiente de *Jaccard*

O coeficiente de similaridade de *Jaccard* é uma medida estatística utilizada para comparar a similaridade e diversidade em conjuntos. Este coeficiente mede a similaridade entre conjuntos finitos e é definido como o resultado da interseção dividido pelo resultado da união dos conjuntos.

Sejam A e B dois conjuntos que representam os interesses de dois perfis, respetivamente. O coeficiente de similaridade é definido por:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \qquad (5)$$

Se A e B forem ambos vazios, define-se que $J(A, B) = 1$.

E o intervalo do coeficiente de *Jaccard* está compreendido ente $0 \leq J(A, B) \leq 1$.

A **distância *Jaccard***, que mede a dissimilaridade entre dois conjuntos, é complementar ao coeficiente *Jaccard* e é obtido subtraindo o coeficiente *Jaccard* ao valor 1, ou equivalente, através da divisão da diferença entre o resultado da união e interseção dos dois conjuntos pela reunião, como se pode observar na fórmula seguinte:

$$d_J(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \qquad (6)$$

Coeficiente de correlação linear de *Pearson*

Outra métrica passível de ser utilizada no cálculo da similaridade entre perfis consiste no coeficiente de correlação linear de *Pearson* (*vide* Equação 1) onde $\boldsymbol{x}$ e $\boldsymbol{y}$ representam perfis e i os diferentes interesses.

3.4 Clustering

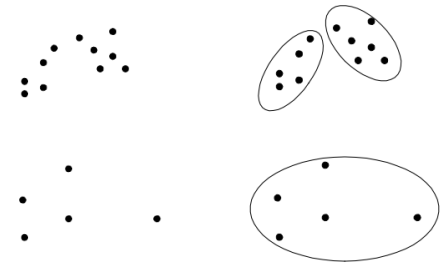
Um dos objetivos deste trabalho é a possibilidade de criar *clusters*, seja de perfis ou de interesses.

Clustering [47] é o processo de análise de uma coleção de pontos, e agrupá-los em *clusters* de acordo com alguma métrica de distância. O objetivo é que esses pontos contidos no mesmo *cluster* tenham uma pequena distância uns dos outros (ou seja, sejam de algum modo relacionados); pelo lado oposto, os pontos em diferentes *clusters* estejam a uma distância significativa.

Segundo [54], a utilização de técnicas de *clustering* melhoram o processo de criar grupos de utilizadores em redes sociais uma vez que reduzem o tamanho do conjunto de dados pelo que a complexidade destes processos é também diminuído. Acrescentam, ainda, que a utilização de *clustering* nos processos de recomendação resolvem problemas como o "início gelado" (*cold start)*[25] e o facto dos dados serem esparsos.

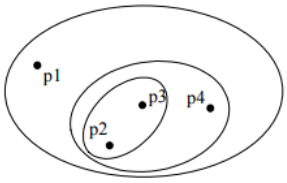
Existem diversos tipos de *clustering* [55]:

- **Hierárquico** ou de **Partição** – um *clustering* do tipo partição consiste em dividir o conjunto de elementos em subconjuntos (*clusters*) que não se sobrepõem de modo que cada elemento pertença apenas a um subconjunto (por exemplo, Figura 3.7). Caso seja permitido que os *clusters* tenham *sub-clusters*, então significa que o tipo de *clustering* é hierárquico, que é um conjunto de *clusters* "aninhados" que estão organizados segundo uma árvore. Cada nó (*cluster*) na árvore (exceto para os nós finais) é a união dos seus filhos (*sub-clusters*) e o nó que se encontra na raiz representa o *cluster* que contém todos os outros *sub-clusters*. Um exemplo pode ser visto na Figura 3.8.



Pontos originais *Clustering* de partição

Figura 3.7 - Clustering de partição



Clustering hierárquico (aninhado)

Figura 3.8 - Clustering hierárquico

[25] *Cold Start* – ocorre quando um novo utilizador se regista na rede social em que esta não tem dados suficientes para efetuar recomendações.

- **Exclusivo** ou **Sobreposto** ou **Impreciso (*fuzzy*)** – Em *clusters* sobrepostos (não-exclusivos), alguns elementos podem pertencer a múltiplos *clusters*. Já nos *fuzzy*, um elemento pertence a todos os *clusters* com um certo grau (entre 0 e 1 em que 0 não corresponde e o 1 pertence totalmente). Finalmente e, como o nome indica, exclusivo indica que os elementos pertencem apenas a um *cluster*. Exemplos dos dois tipos podem ser visto nas Figura 3.9 e Figura 3.10.

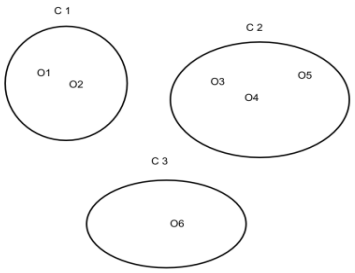


Figura 3.9 - Clustering exclusivo

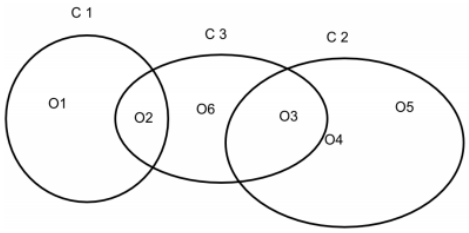


Figura 3.10 - Clustering com sobreposição (não-exclusivo)

- **Completo** ou **Parcial** – efetuar *clustering* completo significa que todos os elementos do conjunto são utilizados. No sentido contrário, quando é parcial podem existir elementos que não são agrupados, por exemplo, *outliers* ou elementos que causam ruído. Por exemplo, uma aplicação que utiliza *clustering* para organizar documentos para navegação terá que garantir que todos os documentos possam ser navegados.

Na secção seguinte, irão ser descritos os algoritmos que a aplicação (Capítulo 4.) suporta quando a meta é criar diversos *clusters*, seja de perfis ou interesses e extrair informação sobre os mesmos.

Em [56], o objetivo dos autores consiste em criar *clusters* de utilizadores na rede social *Twitter*[26]. Os autores referem que os algoritmos *K-Means* e *Clustering* Hierárquico são dois dos mais populares e utilizados na área. No entanto, afirmam que o segundo torna-se lento quando lida com conjuntos de dados grandes como o caso de utilizadores de redes sociais. Outros algoritmos passíveis de escolha são o DBSCAN e o BIRCH, que serão descritos adiante.

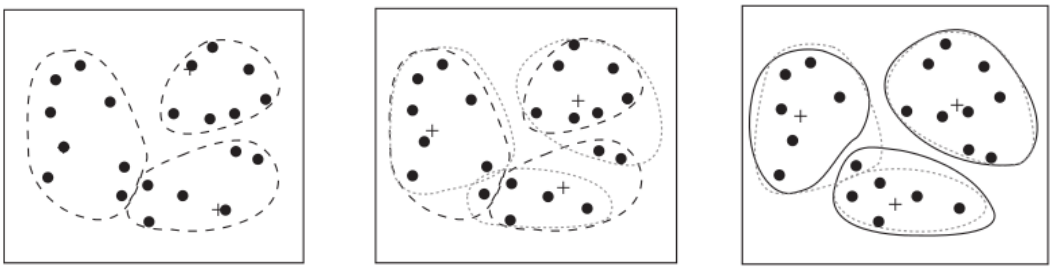
[26] https://twitter.com/

3.4.1 Algoritmo K-*Means*

Este algoritmo pertence ao grupo do tipo partição e define o centróide[27] de um *cluster* como o valor médio dos elementos pertencentes ao *cluster* [57].

O procedimento consiste nos seguintes passos: em primeiro, são escolhidos aleatoriamente k elementos de um conjunto D, cada um como representantes da média ou centro do *cluster*. Para cada um dos restantes elementos é determinado qual o *cluster* correspondente, utilizando uma métrica (por exemplo, a distância Euclidiana) entre o elemento e a centróide. Então, o algoritmo iterativamente melhora a variação intra-*cluster*. Para cada *cluster* é calculado a nova média (centróide) utilizando os elementos de cada *cluster* da última iteração. Seguidamente, todos os elementos são novamente realocados nos *clusters* utilizando os novos valores dos centróides. As iterações continuam até que não haja alterações de elementos em *clusters*.

Um exemplo da utilização deste algoritmo pode ser visto na Figura 3.11 e um resumo pode ser visto no Algoritmo 3.1.



(a) *Clustering* Inicial (*Nota*: dois símbolos + estão sobrepostos por pontos) (b) Iteração (c) *Clustering* Final

Figura 3.11 - Exemplo de clustering utilizando o algoritmo K-Means (o centróide de cada cluster é designado com o símbolo +)

Os resultados podem depender da seleção inicial dos centróides (geralmente aleatórios). Para combater este problema na prática, é comum correr o algoritmo múltiplas vezes com diferentes valores dos centróides.

A ordem de complexidade é $O(n \times k \times t)$, onde n é o número total de elementos, k representa o número de *clusters* e t o número de iterações. Normalmente, k é significativamente menor que n e t é significativamente menor que n. Portanto, o algoritmo é relativamente escalável e eficiente a processar conjuntos de dados grandes.

Existem diversas variantes do método K-*Means*. Estas podem diferir na seleção inicial dos centróides, o cálculo da dissimilaridade ou então estratégias para determinar os centróides dos *clusters*.

[27] Centróide representa o centro ou o ponto representativo do *cluster*.

A necessidade de especificar previamente o número de *clusters* pode ser visto como uma desvantagem. Tem havido estudos para resolver este problema, tal como determinar uma lista de possíveis valores para, posteriormente, utilizar técnicas analíticas com vista a escolher o melhor valor comparativamente aos outros.

Este algoritmo não deve ser utilizado para conjuntos de dados com formas não-convexas ou para *clusters* de diferentes tamanhos. Adicionalmente, é sensível a ruído e elementos que sejam *outliers* uma vez que mesmo um número reduzido desses elementos pode influenciar o valor da média.

Algoritmo K-*Means*

Entrada:

- k: número de *clusters*,
- D: conjunto de dados contendo n elementos.

Saída: Um conjunto de *clusters*.

Método:

1) Escolher aleatoriamente k elementos do conjunto D como centróides
2) **Repetir**
3) (Re)alocar cada elemento ao *cluster* para o qual o elemento é mais similar tendo como base o valor médio dos elementos do *cluster*
4) Atualizar as médias dos *clusters*, ou seja, determinar o valor médio dos elementos para cada *cluster*
5) **Até** não haver mudanças de elementos em *clusters*.

Algoritmo 3.1 - Algoritmo K-Means

3.4.2 Algoritmo *Clustering* Hierárquico (*Hierarchical Clustering)*

Técnicas que utilizam este algoritmo são consideradas como sendo a segunda categoria mais importante no que diz respeito a métodos de *clustering* [55]. Tal como o K-*Means*, estas abordagens são relativamente antigas comparando com outros algoritmos de *clustering*, mas continuam a ser bastante utilizadas. Existem, basicamente, duas abordagens quanto à utilização deste algoritmo: aglomerativo ou divisivo, dependendo se a estratégia adotada quando se procede à decomposição hierárquica for de "baixo-para-cima" (agrupando) ou "cima-para-baixo" (separando).

- **Aglomerativo** (agrupamentos sucessivos) – é utilizado quando a estratégia é de "baixo-para-cima". Tipicamente, começa com os elementos como *clusters* individuais e, em cada iteração, agrupa o par mais próximo de *clusters* até que reste apenas um único *cluster* ou então alguma condição seja atingida. É necessário existir uma noção de proximidade dos *clusters*.
- **Divisivo** (divisões de elementos) – é utilizado na estratégia "cima-para-baixo". Começa com um único *cluster* onde estão contidos todos os elementos e, em cada iteração, os *clusters* vão sendo separados até que cada *cluster* contenha apenas um elemento ou então alguma condição seja atingida. Neste caso, é necessário decidir que *cluster* irá ser dividido e como é que se processa essa separação.

As técnicas que utilizam a abordagem aglomerativa são mais utilizadas, pelo que nesta secção irá ser dado um resumo do processo do algoritmo relativamente a este tipo.

Graficamente, o resultado deste algoritmo costuma ser traduzido num diagrama denominado **dendrograma** ou diagrama de árvore. Como se trata de uma representação baseada em árvore, os ramos representam os elementos e a raiz o agrupamento de todos eles, como se pode observar na Figura 3.12.

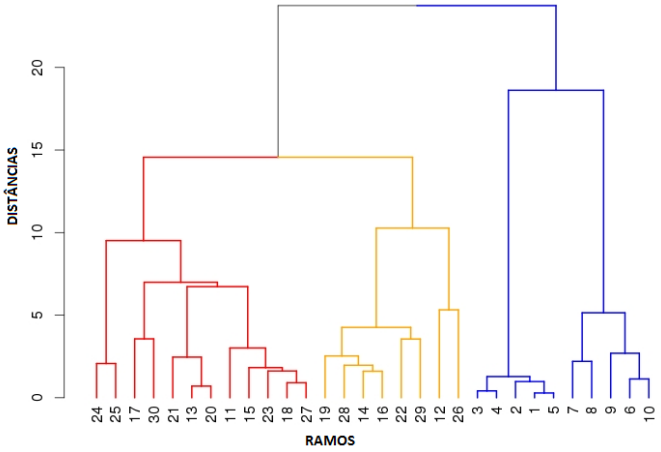


Figura 3.12 - Exemplo de dendrograma

Tendo acesso ao dendrograma, uma opção possível é definir uma distância de corte, ou seja, atribuir um limite para o número de *clusters*. Por exemplo, na Figura 3.13 pode-se observar diversas possibilidades de corte. O "Corte 3" corresponderia a dois *clusters*, ou seja, os elementos {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} num *cluster* e os restantes noutro. O "Corte 2" iria resultar em 4 *clusters* e, finalmente, o "Corte 1" iria resultar em 10 *clusters*.

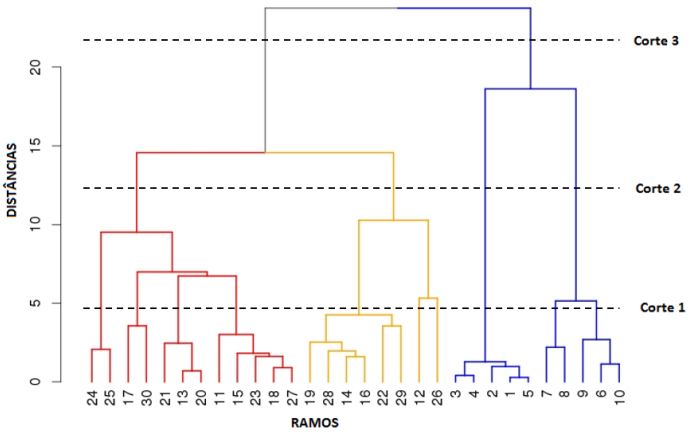


Figura 3.13 - Exemplo de cortes num dendrograma

Em ambos os casos (aglomerativo e divisivo), pode-se definir o número de *clusters* como uma condição de paragem. Uma ilustração das duas estratégias relativamente ao dendrograma pode ser vista na Figura 3.14.

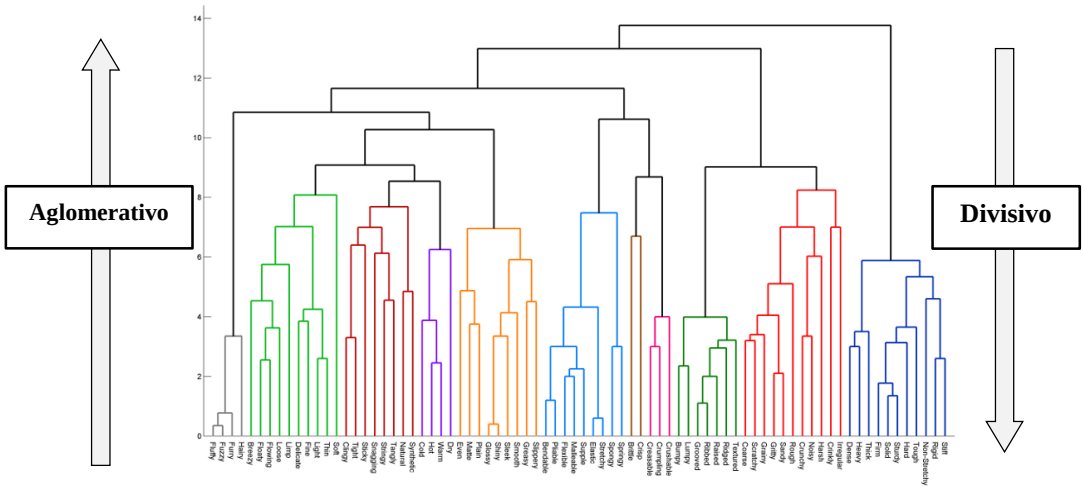


Figura 3.14 – Ilustração das estratégias Aglomerativa e Divisiva

Algoritmo Aglomerativo de *Clustering* Hierárquico

Como referido anteriormente, este algoritmo começa por considerar todos os elementos como *clusters* individuais e, iterativamente, agrupa os dois *clusters* mais próximo segundo uma medida de proximidade predefinida. O processo termina quando apenas restar um *cluster* ou então uma condição seja atingida. Esta abordagem está formalmente descrita no Algoritmo 3.2.

Algoritmo Aglomerativo de *Clustering* Hierárquico

Entrada:

- ***k***: número de *clusters* (opcional como condição de paragem).
- ***D***: conjunto de dados contendo n elementos.

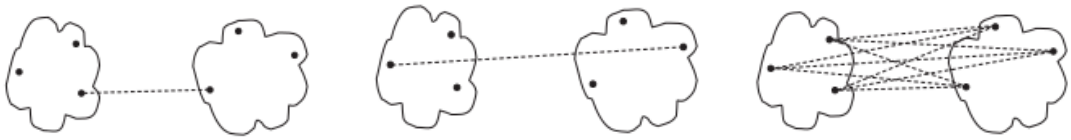
Saída: Árvore com níveis hierárquicos dos *clusters*.

Método:

1) Computar a matriz de proximidade, caso necessário.
2) **Repetir**
3) Agrupar os dois *clusters* mais próximos.
4) Atualizar a matriz de proximidade de modo a refletir a proximidade entre o novo *cluster* e os originais.
5) **Até** apenas existir um *cluster* ou condição de paragem atingida.

Algoritmo 3.2 - Algoritmo Aglomerativo de Clustering Hierárquico

Uma operação chave deste algoritmo é o cálculo da proximidade entre dois *clusters*. Podem ser utilizadas diversas métricas, tais como **Mínimo**, **Máximo** ou **Média**. Mínimo define que a proximidade é o valor mínimo das distâncias entre dois elementos de *clusters* diferentes. Máximo define que a proximidade deve ser calculada através da distância dos dois elementos mais afastados dos dois *clusters*. A terceira alternativa considera a utilização do valor médio de todas as distâncias dos pares dos elementos dos diferentes *clusters*. A Figura 3.15 ilustra as três métricas.



(a) Mínimo (b) Máximo (c) Média

Figura 3.15 - Métricas para definir a proximidade entre dois clusters [55]

Outra alternativa que também pode ser utilizada como métrica de proximidade é o método de *Ward* [55]. Este método considera que os *clusters* são representados pelos seus centróides, e mede a proximidade entre dois *clusters* baseando-se na otimização dos erros quadráticos que resultam quando se agrupam dois *clusters*. Tal como no algoritmo K-*Means*, este método tenta minimizar a soma das distâncias quadráticas dos elementos dos centróides dos *clusters*. Este método é menos suscetível a ruído e *outliers*.

Considerando n como o número de elementos de um conjunto, a ordem de complexidade deste algoritmo é, no caso geral, $O(n^3)$ o que faz com que seja lento para conjuntos de dados grandes.

Geralmente, este tipo de algoritmo é utilizado quando o problema consiste na criação de uma taxonomia que requer uma hierarquia. No entanto, o processamento tem um custo elevado. O facto de todos os agrupamentos serem finais, ou seja, é impossível retroceder, pode causar ruído e prejudicar os resultados obtidos. Para resolver estes dois problemas, uma técnica que pode ser utilizada consiste em separar inicialmente os dados em *clusters* utilizando outras técnicas, tais como o K-*Means* [57].

3.4.3 Algoritmo DBSCAN

Uma outra vertente para criar agrupamentos de dados utiliza o conceito de densidade [55]. Deste modo, no espaço multidimensional regiões com grande concentração de pontos pertencem ao mesmo *cluster*. O algoritmo DBSCAN (*Density-based Spatial Clustering of Applications with Noise*) é um exemplo deste tipo.

Apesar de não existirem tantas abordagens para definir a densidade como existem na definição de similaridade, existem diversos métodos distintos. Nesta secção apenas se considera a abordagem baseado no centro.

Densidade tradicional: abordagem baseada no centro

Nesta abordagem, a densidade é estimada para um elemento específico através da contagem do número de elementos que estejam dentro de um determinado raio (denominado Eps). Esta técnica encontra-se ilustrada na Figura 3.16. Neste exemplo, o número de elementos (pontos) dentro de um raio (Eps) a partir do elemento A é de 7, incluindo o próprio A.

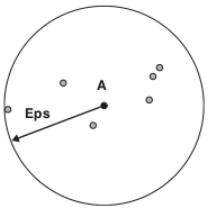


Figura 3.16 - Exemplo de densidade baseada no centro

Trata-se de um método simples de implementar, mas a densidade de qualquer elemento depende da especificação do raio. Por exemplo, se o raio for suficientemente grande, então todos os elementos terão densidade m, onde m é o número de elementos no conjunto de dados. Analogamente, se o raio for demasiado pequeno, então todos os elementos terão densidade 1, ou seja, o próprio elemento. Uma estratégia para decidir qual o raio apropriado para um conjunto de dados com baixa dimensão será visto ainda nesta secção.

A abordagem de densidade baseada no centro permite classificar os elementos em diversos tipos, dos quais se destacam:

- **Elemento do núcleo** (*Core point*) – elementos que se encontram no interior do *cluster*. Um elemento pertence ao núcleo caso haja um número suficiente de pontos na sua vizinhança correspondente ao argumento $MinPts$ (este argumento é definido previamente) e que se encontrem dentro do raio Eps (outro argumento também definido previamente). Na Figura 3.17, um elemento deste tipo é o ponto A para o Eps indicado e $MinPts \leq 7$.
- **Elemento dos limites** (*Border point*) – é um elemento que apesar de não pertencer ao núcleo, ainda se encontra na vizinhança do elemento do núcleo. Na Figura 3.17, um exemplo deste tipo de elemento é o ponto B.
- **Elemento de ruído** (*Noise point*) – é um elemento que não se enquadra nos dois anteriores. Corresponde ao ponto C na Figura 3.17.

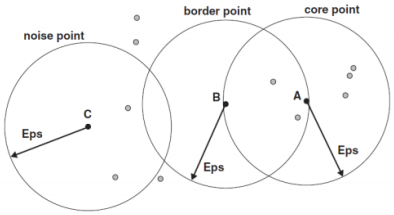


Figura 3.17 - Classificação dos elementos de acordo com a densidade baseada no centro [55]

Algoritmo DBSCAN

Considerando as definições anteriores, o algoritmo DBSCAN pode ser descrito como: quaisquer dois elementos que se encontrem suficientemente pertos – com uma distância Eps entre eles – são colocados no mesmo *cluster*. Da mesma forma, qualquer elemento dos limites que esteja suficientemente perto a um elemento do núcleo é colocado no mesmo *cluster* como um elemento do núcleo. Os pontos de ruído são descartados. Um resumo do funcionamento deste algoritmo pode ser visto no Algoritmo 3.3.

Uma das tarefas necessárias antes de executar este algoritmo é a escolha dos parâmetros Eps e $MinPts$. A abordagem básica consiste em observar o comportamento da distância de um dado elemento aos seus k vizinhos, que pode ser denominada por *k-dist*. Para elementos que pertençam ao mesmo *cluster*, o valor de *k-dist* será pequeno se k não for maior que o tamanho do *cluster*. É necessário notar que poderá existir alguma variação, dependendo na densidade do *cluster* e na distribuição aleatória dos elementos mas, em média, o intervalo da variação não será grande caso as densidades do *cluster* não forem extremamente diferentes. No entanto, para elementos que não estejam no *cluster*, tais como os elementos de ruído, o valor de *k-dist* será relativamente grande. Portanto, caso o valor *k-dist* seja calculado para todos os elementos para um determinado k, ordená-los por ordem crescente e listar os valores ordenados, é esperado visualizar uma mudança radical no valor de *k-dist* que corresponde a um valor aceitável para o parâmetro Eps. Caso se selecione este valor como o parâmetro Eps e considere o valor de k como o parâmetro $MinPts$, então os elementos para os quais *k-dist* é menor que Eps serão categorizados como elementos do núcleo, enquanto os outros elementos serão categorizados como ruído ou então elementos dos limites.

A Figura 3.18 (Pontos originais) mostra um exemplo de um conjunto de dados, enquanto o gráfico *k-dist* para os dados é dado na Figura 3.19. O valor de Eps que é determinado através desta forma depende de k, mas não se altera drasticamente quando há alterações de k. Se o valor de k for muito pequeno, então mesmo um número pequeno de elementos próximos que sejam ruído ou *outliers* serão incorretamente categorizados em *clusters*. Se o valor de k for demasiado grande, então *clusters* pequenos (de tamanho menor que k) serão categorizados como ruído.

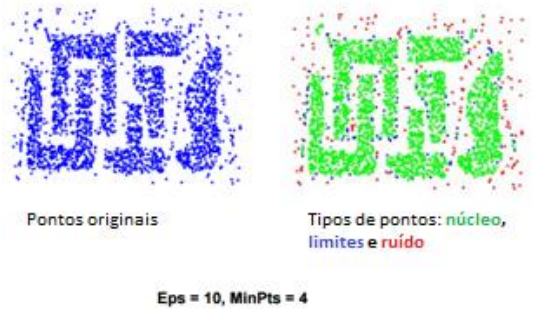


Figura 3.18 - Exemplo de utilização do algoritmo DBSCAN [55]

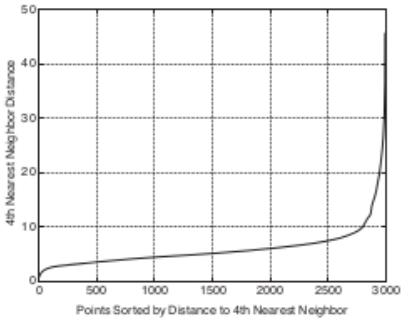


Figura 3.19 - Gráfico k-dist [55]

Este algoritmo apresenta como vantagens a relativa resistência a ruído e é capaz de lidar com *clusters* de formas e tamanhos arbitrários. Adicionalmente, o DBSCAN poderá encontrar *clusters* que não sejam identificados utilizando, por exemplo, o algoritmo K-*Means*, como pode ser visto na Figura 3.18.

No entanto, este algoritmo poderá ter problemas quando os *clusters* têm densidades muito variadas. Outro problema que poderá, eventualmente, ocorrer é na presença de dados com um número grande de dimensões uma vez que neste caso a determinação da densidade torna-se difícil. Finalmente, o DBSCAN pode ter um custo computacional elevado quando é necessária determinar todas as proximidades para todos os pares, o que é habitual em conjuntos de dados com um número grande de dimensões.

Algoritmo DBSCAN

Entrada:

- $\boldsymbol{D}$: conjunto de dados contendo n elementos.
- $\boldsymbol{Eps}$: valor do raio
- $\boldsymbol{MinPts}$: número mínimo de densidade na vizinhança.

Saída: Conjunto de *clusters* baseados na densidade.

Método:

```
1)  Marcar todos os elementos como "não visitado"
2)  Fazer
3)      Escolher aleatoriamente um elemento "não visitado" p
4)      Marcar como "visitado"
5)      Se o Eps de p tem no mínimo MinPts elementos
6)          Criar um novo cluster C e adicionar p a C
7)          Seja N o conjunto de elementos dentro de Eps de p:
8)          Para cada ponto p' em N:
9)              Se p' é "não-visitado":
10)                 Marcar p' como "visitado"
11)                 Se o Eps de p' tem no mínimo MinPts elementos
12)                     Adicionar esses pontos a N
13)             Se p' não é membro de nenhum cluster, adicionar p' a C
14)         Terminar o ciclo
15)         Devolver C
16)     Senão marcar p como ruído
17) Até que não haja elementos "não visitado"
```

Algoritmo 3.3 - Algoritmo DBSCAN

3.4.4 Algoritmo BIRCH

O algoritmo BIRCH (*Balanced Iterative Reducing and Clustering using Hierarchies*) é do tipo hierárquico e é utilizado, geralmente, para lidar com conjuntos de dados grandes [58].

Uma das vantagens deste algoritmo é a possibilidade de ir agrupando os dados de entrada de uma forma incremental e dinâmica, ou seja, não necessita da presença do conjunto de dados completo desde o início. Tem em conta as limitações de memória e tempo, pelo que tenta encontrar o melhor processo de *clustering* possível

para esse conjunto de recursos. Na maioria das vezes, este algoritmo precisas apenas de uma passagem pelos dados.

O algoritmo BIRCH é baseado em diversas fases, das quais duas são as principais: percorrer o conjunto de dados de modo a criar uma árvore em memória (ou seja, cria um dendrograma denominado *Clustering Feature Tree* ou CF-*tree*); seguidamente, aplica o algoritmo de *clustering* para agrupar os diversos nós (folhas). Opcionalmente, pode haver mais duas fases, que neste documento não são descritas, no entanto podem ser consultadas em [58].

Fase 1: Armazenamento dos dados na CF-*tree*

Começando com um valor de limiar inicial, percorrem-se os dados inserindo-os na CF-*tree*. Caso a memória se esgote antes que os dados sejam todos percorridos, o valor do limiar é aumentado e a CF-*tree* é reconstruída, mas desta vez com um menor número de nós, reinserindo as folhas da CF-*tree* antiga. Após a finalização deste processo, ou seja, todos os registos das folhas terem sido reinseridos, o passo onde se percorriam todos os dados é reiniciado do ponto onde havia parado.

Fase 2: Formação global dos agrupamentos

Dependendo da ordem que os dados vão sendo são adicionados à CF-*tree* ou caso o conjunto de dados possua uma distribuição assimétrica, pode acontecer que alguns *clusters* não correspondam totalmente à realidade. Esta fase surge para minimizar esse problema.

Com os vetores $\boldsymbol{CF}$ conhecidos, cada *subcluster* pode ser tratado como um simples registo, através do cálculo do seu centróide, representando o *subcluster*. Ou seja, os *subclusters* podem ser utilizados sem nenhuma modificação, já que a informação contida nos vetores $\boldsymbol{CF}$ torna-se suficiente. O algoritmo BIRCH proposto em [58] utiliza um algoritmo hierárquico aglomerativo diretamente nos *subclusters* encontrados na fase anterior, representados pelos seus $\boldsymbol{CFs}$. O algoritmo define uma métrica de distância para aglomerar os *subclusters* até que se obtenha o número final de *clusters* pretendido.

Este algoritmo torna-se facilmente escalável. A complexidade da fase 1 do algoritmo é linear de acordo com tamanho do conjunto de dados [59].

Algoritmo BIRCH

Entrada:

- k: número de *clusters*.
- D: conjunto de dados contendo n elementos.

Saída: Conjunto de *clusters*.

Método:

```
1)  #Fase 1
2)  Inicializar a CF-tree vazia, N = {}
3)  Para cada ponto p de D fazer:
4)      Escolher o nó folha (M) em N que está mais próximo de p
5)      Adicionar p a m.
6)      Calcular o diâmetro M de m
7)      Se M>k:
8)          Separar m
9)          #poderá ser necessário separar os antecessores de m
10) #Fase 2
11) Aplicar outro algoritmo de clustering para criar os clusters das folhas de N
```

Algoritmo 3.4 - Algoritmo BIRCH

3.5 Considerações finais

Uma fase importante para se decidir qual a estratégia a seguir é a exploração dos dados. Neste capítulo foi possível entender a proveniência e tipologia dos mesmos, assim como o processo de limpeza necessário a que foram submetidos. Tendo em conta os valores estatísticos obtidos, decidiu-se utilizar as categorias dos "Likes" para perfilar os utilizadores, ou seja, representar os perfis como vetores de interesses. Foi, então, criada uma matriz onde as linhas representavam os perfis e as colunas os respetivos interesses. A fase seguinte consistiu em determinar a correlação entre os interesses, utilizando a correlação de *Pearson*. Seguidamente, definiu-se o conceito de *clustering* e detalharam-se diversas métricas de distâncias entre vetores, tais como Euclidiana, *Manhathan*, *Jaccard*, entre outras. Estas métricas são utilizadas para determinar os perfis similares. Finalmente, foram descritos quatro algoritmos diferentes que podem ser utilizados em problemas de *clustering*: K-*Means*, *Clustering* Hierárquico, DBSCAN e BIRCH.

4. *matchly* (protótipo)

"If the facts don't fit the theory, change the facts."
Albert Einstein

No presente capítulo expõem-se os requisitos que foram surgindo ao longo do desenvolvimento da aplicação, assim como uma breve explicação da sua arquitetura, implementação e descrição funcional.

4.1 Requisitos

De um modo geral, o conjunto de requisitos de um sistema é definido durante as fases iniciais do processo de desenvolvimento. Tal conjunto de requisitos é visto como uma especificação das funcionalidades a implementar. Os requisitos são, portanto, descrições de como o sistema deverá funcionar além de conterem informações do domínio da aplicação e restrições sobre a operação do sistema.

4.1.1 Requisitos funcionais

Um requisito funcional pode ser definido como uma função de um sistema de *software* ou um seu componente. Uma função, por sua vez, é descrita como um conjunto de entradas, respetivo processamento e as saídas.

Para especificar esta aplicação, os requisitos funcionais podem ser subdivididos em 3 categorias:

- **Fontes de dados**
  - ✓ Importar ficheiros de dados provenientes de uma rede social
  - ✓ Visualizar a lista atual de ficheiros de dados
  - ✓ Processar o(s) ficheiro(s) de modo a criar perfis e guardar em base de dados
- **Interesses**
  - ✓ Visualizar e explorar a matriz de correlação entre interesses
  - ✓ Criar um dendrograma que representa a estrutura hierárquica de *clustering* dos interesses
  - ✓ Criar e explorar *clusters* de interesses
- **Perfis**
  - ✓ Visualizar e explorar a tabela de perfis presentes na base de dados
  - ✓ Visualizar e explorar a matriz de perfis-interesses
  - ✓ Determinar os perfis mais similares a um dado perfil
  - ✓ Criar e gravar modelos com possibilidade de utilização de diversos algoritmos de *clustering*
  - ✓ Criar e explorar *clusters* de perfis

4.1.2 Requisitos não-funcionais

Enquanto os requisitos funcionais especificam resultados particulares de um sistema, estes devem ser complementados com requisitos não-funcionais, os quais especificam caraterísticas gerais no que diz respeito ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, disponibilidade, segurança e tecnologias envolvidas. Por vezes, estes requisitos não-funcionais provocam mesmo restrições aos funcionais.

Deste modo, os requisitos não-funcionais para esta aplicação são os seguintes:

- **Privacidade** – Os dados não deverão conter qualquer informação que seja passível de identificação pessoal (por exemplo, nome, *email*, etc.). A base de dados deve ser protegida para acesso apenas de utilizadores autorizados e devidamente autenticados.
- **Desempenho** – O sistema deverá devolver um *feedback* do estado de processamento da tarefa que o utilizador ativou. Em caso de erro deverá também alertar o utilizador.
- **Usabilidade** – este requisito está associado à facilidade de uso da aplicação. A *interface* deverá ser de fácil navegação e, sempre que possível, conter secções de ajuda com descrições das funcionalidades.

4.2 Modelação

Esta secção visa descrever as diferentes tarefas de modelação que fazem parte da fase inicial do desenvolvimento da aplicação. Será especificada a arquitetura geral, os diversos casos de utilização e o diagrama de classes.

4.2.1 Arquitetura geral

A aplicação baseia-se numa aplicação *web*, constituída por um conjunto de camadas: **Visualização**, **Processamento** e **Persistência de Dados** (Figura 4.1).

A camada principal (Processamento), consiste em todas as operações desde a limpeza, transformação e processamento de dados provenientes das fontes de dados como a manipulação das matrizes referidas anteriormente e a manipulação dos modelos de *clustering*.

A camada inferior consiste na persistência de dados onde é utilizada uma base de dados ou então utiliza o sistema de ficheiros (texto e binários) para manipular imagens e ficheiros dos diversos modelos de *clustering*.

Finalmente, a camada superior engloba as funcionalidades de visualização dos dados processados tais como matrizes, *clusters* ou figuras.

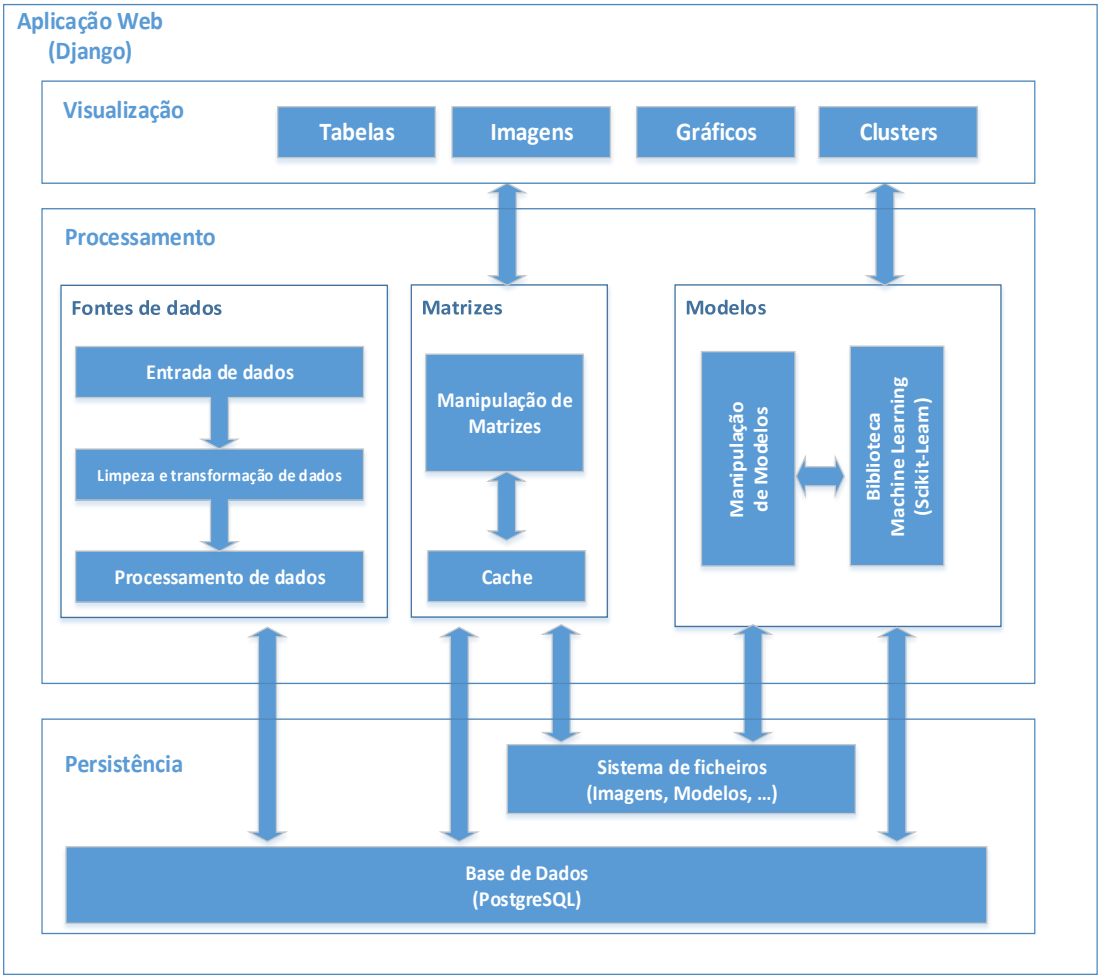


Figura 4.1 - Arquitetura geral

4.2.2 Casos de utilização

Para simplificar o planeamento da aplicação recorreu-se a diagramas de casos de utilização, em notação *Unified Modeling Language* (UML) [60], que define os requisitos básicos da aplicação.

A Figura 4.2 representa o caso de utilização do cenário "Fontes de dados". Este cenário carateriza-se pela manipulação dos ficheiros de dados, desde ao seu carregamento na aplicação até ao processamento dos mesmos por forma a criar perfis na base de dados. O utilizador também poderá, caso pretenda, eliminar os dados presentes na base de dados.

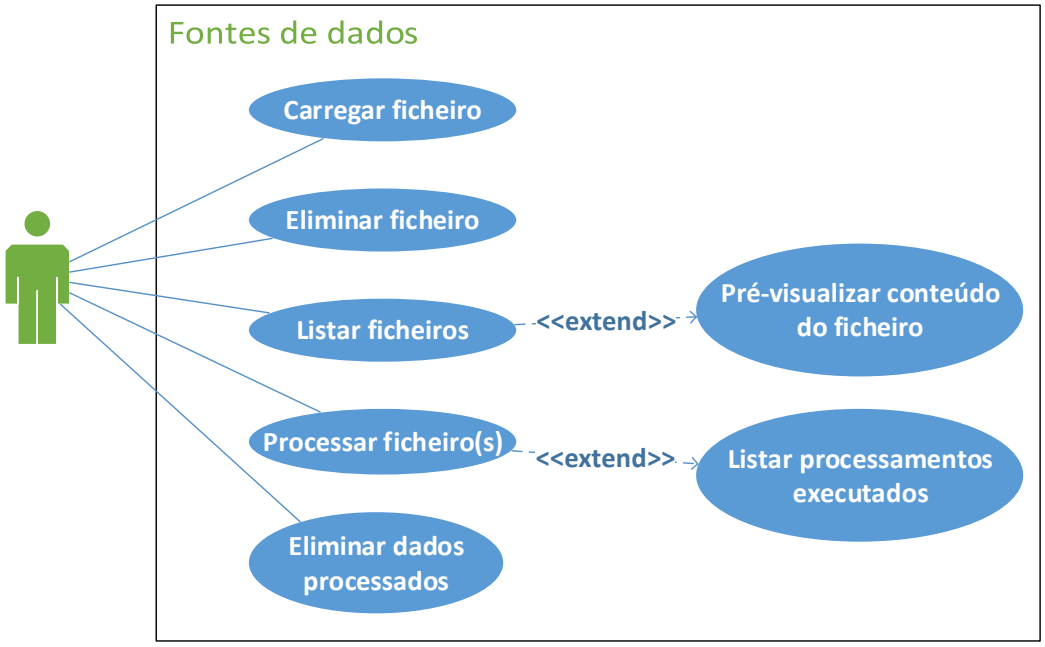


Figura 4.2 - Caso de utilização (Fontes de dados)

Na Figura 4.3 estão representadas as principais funcionalidades que o utilizador tem acesso respetivamente ao cenário "Interesses", das quais se destacam:

- **Explorar a matriz de correlação de interesses** – como referido anteriormente, esta matriz representa os valores de correlação entre os interesses extraídos dos perfis sociais. O utilizador pode explorá-la através de duas opções: listar em tabela ou através da matriz interativa.
- **Atualizar a matriz de correlação** – sempre que o utilizador desejar (por exemplo, quando existem dados novos na base de dados), tem à sua disposição a possibilidade de atualizar esta matriz.
- **Visualizar dendrograma** – o dendrograma representa os níveis hierárquicos dos diversos *clusters* dos interesses.
- **Atualizar dendrograma** – analogamente à opção de atualização supracitada, o utilizador pode, sempre que achar oportuno, atualizar o ficheiro do dendrograma.
- **Criar *clusters*** – uma das opções mais importantes neste cenário consiste em criar os diversos *clusters* de interesses para ser possível a sua visualização e, desse modo, retirar informações. O utilizador pode, também, escolher o número total de *clusters* que irão ser criados.

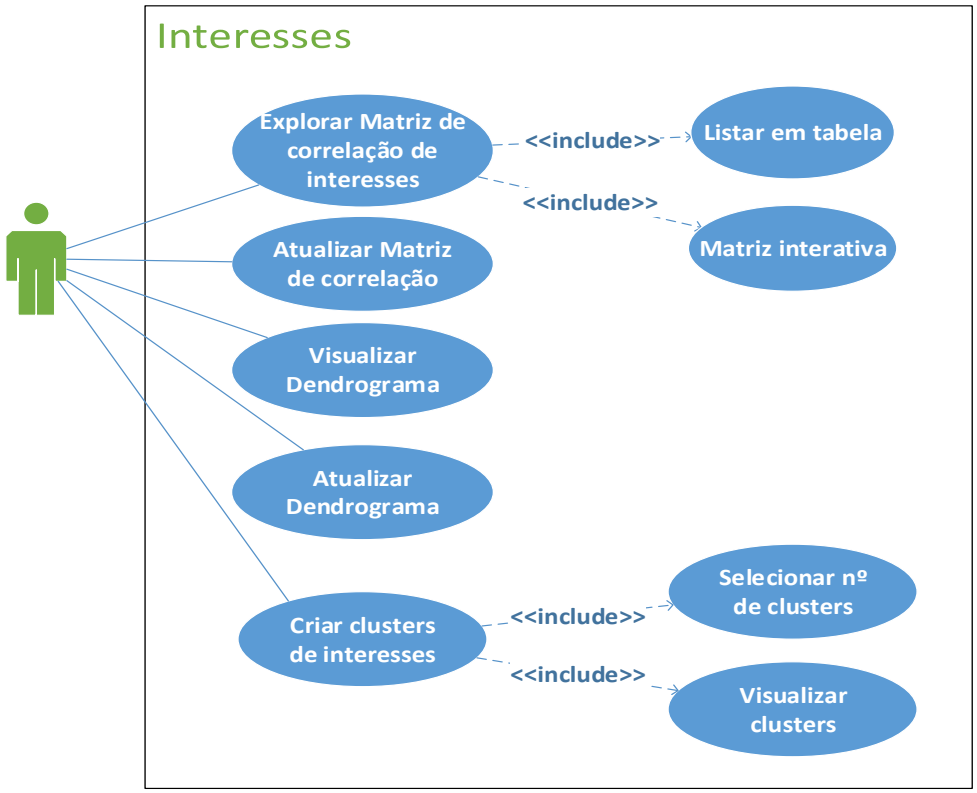


Figura 4.3 - Caso de utilização (Interesses)

No que diz respeito ao cenário "Perfis" (Figura 4.4), as principais funcionalidades consistem em:

- **Explorar matriz de perfis-interesses** – esta opção consiste em explorar a matriz de perfis-interesses em forma de tabela.
- **Atualizar matriz de perfis-interesses** – sempre que pretenda ou achar necessário (por exemplo, entrada de novos dados no sistema), o utilizador tem à sua disposição a opção de atualizar os valores da matriz.
- **Criar *Ranking* de perfis similares** – uma das funcionalidades principais da aplicação consiste em criar uma lista dos perfis mais similares a um dado perfil.
- **Criar modelo (*clustering*)** – a aplicação deverá suportar a possibilidade de criar modelos utilizando diferentes algoritmos (configuráveis, ou seja, com a alternativa de escolha de determinados parâmetros) de *clustering*. Sempre que o utilizador pretenda, a aplicação permite gravar o modelo criado.
- **Listar modelos** – Opção que permite listar todos os modelos gravados previamente. O utilizador pode, também, eliminar modelos ou então criar e visualizar os *clusters* de perfis utilizando um determinado modelo.

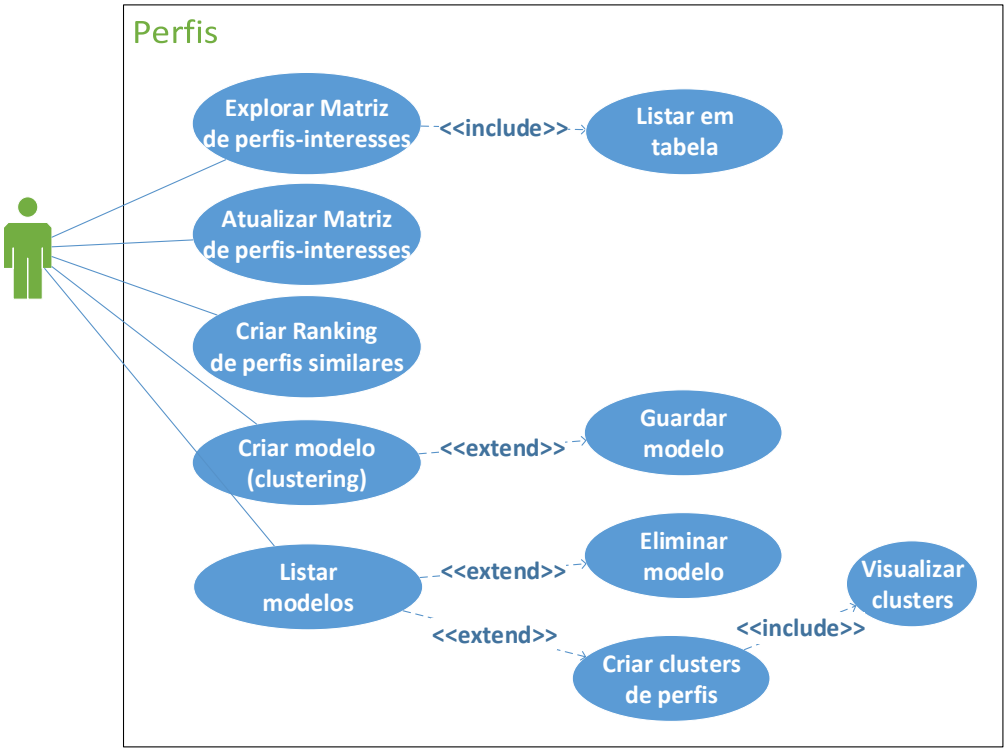


Figura 4.4 - Caso de utilização (Perfis)

4.2.3 Diagrama de classes

Os casos de utilização fornecem uma perspetiva do sistema de um ponto de vista externo, ou seja, do ator. Por outro lado, o diagrama de classes é uma representação da estrutura e relações entre as classes do próprio sistema. Na Figura 4.5 pode ser visto o diagrama de classes da aplicação *matchly*.

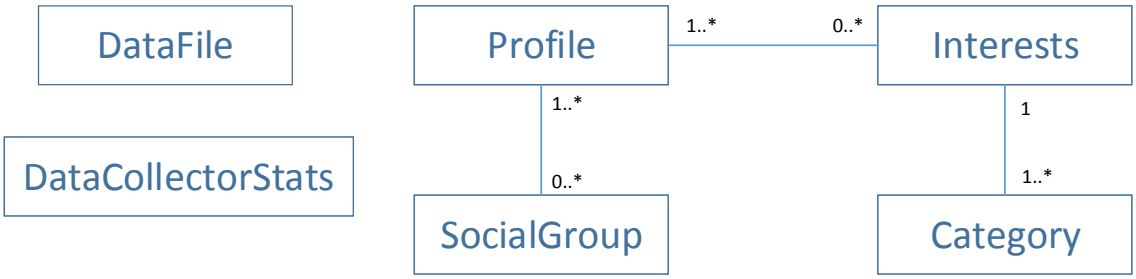


Figura 4.5 - Diagrama de Classes (versão resumida)

- ***DataFile*** – Esta classe representa os ficheiros que são carregados na aplicação. Ou seja, as fontes de dados com dados provenientes de redes sociais.
- ***Profile*** – A classe mais importante que representa os perfis previamente processados a partir dos dados originais.
- ***Category*** – Esta classe representa as diferentes categorias dos interesses dos diversos perfis. Existem dois níveis hierárquicos de categorias.
- ***Interests*** – Esta classe representa a relação entre os interesses e os perfis. Ou seja, dispõe informação do número de "Likes" que cada utilizador realizou numa dada categoria.
- ***SocialGroup*** – A classe consiste na representação dos grupos sociais obtidos a partir dos ficheiros provenientes das redes sociais.
- ***DataCollectorStats*** – Consiste em representar estatísticas sobre os ficheiros (fontes de dados) processados (número de perfis criados, tempo total de processamento, número de ficheiros processados com sucesso, número de ficheiros processados com erro, etc.).

Para uma descrição mais detalhada dos atributos associados a cada classe deve-se consultar o anexo.

4.3 Implementação

Uma vez definida a modelação da aplicação, procedeu-se à sua implementação. Nas secções seguintes irão ser descritas as tecnologias utilizadas assim como irão ser disponibilizadas diversas imagens das funcionalidades disponíveis.

4.3.1 Tecnologias utilizadas

Django[28] foi a tecnologia utilizada para criar a aplicação *web*. Trata-se de uma *framework* que utiliza a linguagem de programação *Python* e tem como foco o rápido desenvolvimento de aplicações não descurando um *design* pragmático e limpo. Foi construído por programadores experientes, pelo que se trata de uma *framework* robusta no que diz respeito a desenvolvimento *web*. É gratuita e *open-source*[29].

A justificação para a escolha da tecnologia supracitada deve-se ao facto de já existir alguma experiência no desenvolvimento *web* através desta *framework* e, também, facilitar a utilização da biblioteca de algoritmos de *machine learning* (utiliza também a linguagem *Python*) que irá ser descrita ainda nesta secção.

[28] https://www.djangoproject.com/

[29] *Open source* é um termo em inglês que significa código aberto. Refere-se ao código-fonte de um programa, que pode ser adaptado para diferentes fins. O termo foi criado pela OSI (*Open Source Initiative*) que o utiliza sob um ponto de vista essencialmente técnico.

Como sistema de gestão de base de dados (SGBD), a tecnologia escolhida foi **PostgreSQL**[30]. Trata-se de um sistema que pode ser utilizado como solução de gestão de dados de alto rendimento além de ser agnóstico em relação à plataforma de sistema operativo, funcionando tanto em *Windows*, *Linux*, *Unix* ou *Mac OS* [61]. Trata-se, ainda, de um sistema capaz de rivalizar com outros sistemas com reconhecimento no mercado (por exemplo, *Oracle* ou *SQL Server*).

Scikit-Learn[31] agrega ferramentas de suporte a *data mining* e análise de dados (tais como *NumPy*, *SciPy* e *matplotlib*). É escrita na linguagem *Python* e contém uma vasta lista de algoritmos utilizados em contexto de *machine learning* (*Support Vector Machines*, *Logistic Regression*, *K-Means*, etc.). Está acessível a todas as pessoas que a pretendam utilizar, é *open-source* e pode fazer parte, inclusive, de produtos em contexto comercial. É utilizada, inclusive, pelo *Facebook* em projetos ligados à avaliação da sua rede social.

Outra ferramenta utilizada foi a biblioteca **Pandas**[32], escrita em *Python* que fornece alta *performance*, facilita a utilização de estruturas de dados e ferramentas de análise de dados. Nesta aplicação, foi essencialmente utilizada na fase de transformação e limpeza de dados e para manipulação de estruturas de dados, geralmente, com grandes quantidades (por exemplo, matrizes).

HTML, **CSS** e ***JavaScript*** – como é comum em qualquer aplicação *web*, estas três tecnologias fazem parte da lista utilizadas neste protótipo.

A nível de *design*, foi utilizada a *framework* **Bootstrap**[33] – conjunto gratuito e *open-source* que agrega ferramentas de criação de aplicações *web*. Contém modelos de *design* baseados em CSS e HTML com tipografia, formulários, botões, navegação e outros componentes de *interface* assim como extensões de *JavaScript*. Facilita e acelera o desenvolvimento *web* conseguindo obter um *design* simples com aspeto profissional.

Para visualizar os diversos gráficos estatísticos e diversos *clusters*, foi utilizada a biblioteca **d3.js**[34]. É uma biblioteca escrita em *JavaScript* com o objetivo de manipulação de documentos baseados em dados. Utiliza tecnologias, tais como HTML, SVG e CSS. As suas classes podem ser reutilizadas ou mesmo alteradas de modo a serem adaptadas a um problema específico.

Foi, ainda, utilizado o *software* de controlo de versões de código **Git**[35], o qual traz muitas vantagens uma vez que serve como segurança, tanto de cópias antigas, como de permitir retroceder para versões antigas ao longo do desenvolvimento.

[30] http://www.postgresql.org/
[31] http://scikit-learn.org/
[32] http://pandas.pydata.org/
[33] http://getbootstrap.com/
[34] http://d3js.org/
[35] https://git-scm.com/

4.3.2 Funcionalidades

Nesta secção, irão ser listadas as principais funcionalidades da aplicação *matchly*. As restantes funcionalidades podem ser consultadas em anexo.

Ecrã de autenticação – a página inicial permite ao utilizador autenticar-se para ter acesso ao *back-office*.

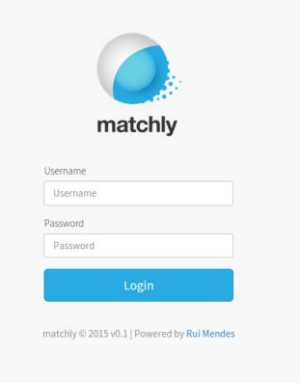


Figura 4.6 - Página de autenticação

Profiler (Fontes de dados) – As principais opções consistem em manipular os ficheiros com dados provenientes de redes sociais: carregar, processar (criar perfis), eliminar ou listar.

Na Figura 4.7, do lado esquerdo estão representadas as opções do *menu* principal que dá acesso ao *Data Collector* e *List*. A primeira pode ser vista na parte direita da figura e é onde é possível carregar ficheiros, processá-los ou ainda visualizar estatísticas sobre os processamentos anteriores.

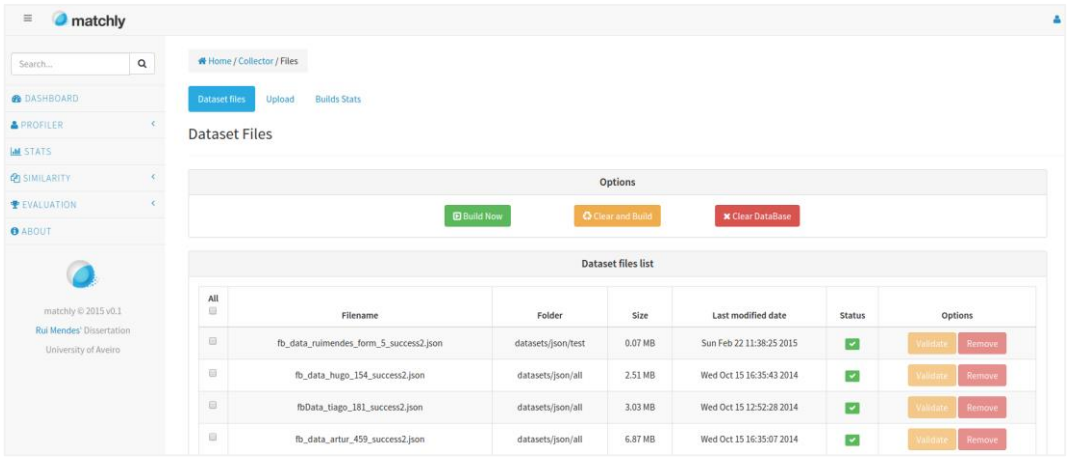


Figura 4.7 - Menu Profiler e Lista de ficheiros carregados

Stats – É possível visualizar diversos gráficos e estatísticas dos dados dos perfis presentes na base de dados.

Dentro da opção "Stats", é possível navegar por diversos campos com vista a visualizar gráficos dos dados assim como alguns dados estatísticos (Figura 4.8).

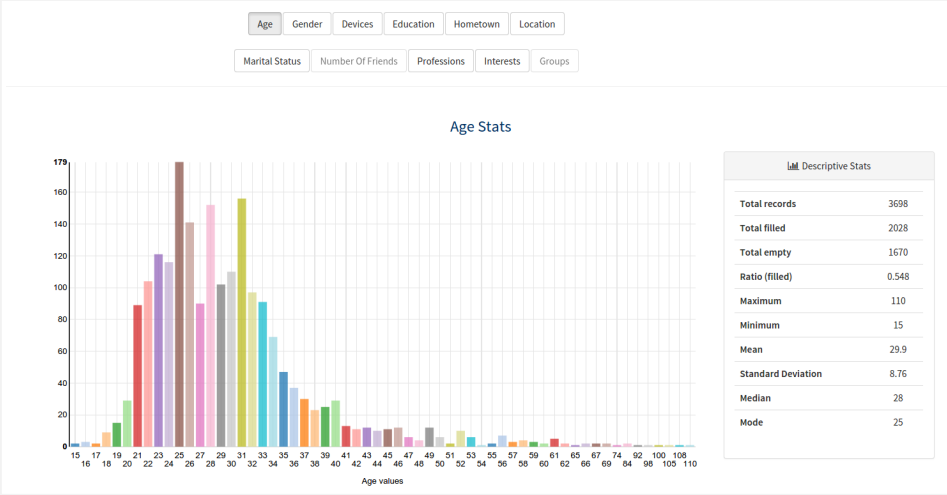


Figura 4.8 - Gráfico e estatísticas do campo Age

Similarity – Área onde é possível consultar as matrizes de correlação de interesses, perfis-interesses, criar *clusters* de interesses, criar modelos ou ainda criar *clusters* de perfis.

A opção *Similarity* > *Interests* disponibiliza diversas opções, das quais se destacam: visualizar a matriz de correlação entre interesses, explorar o conjunto de dados da matriz anterior numa matriz interativa (Figura 4.9), visualizar um dendrograma ou criar *clusters* de interesses (Figura 4.10).

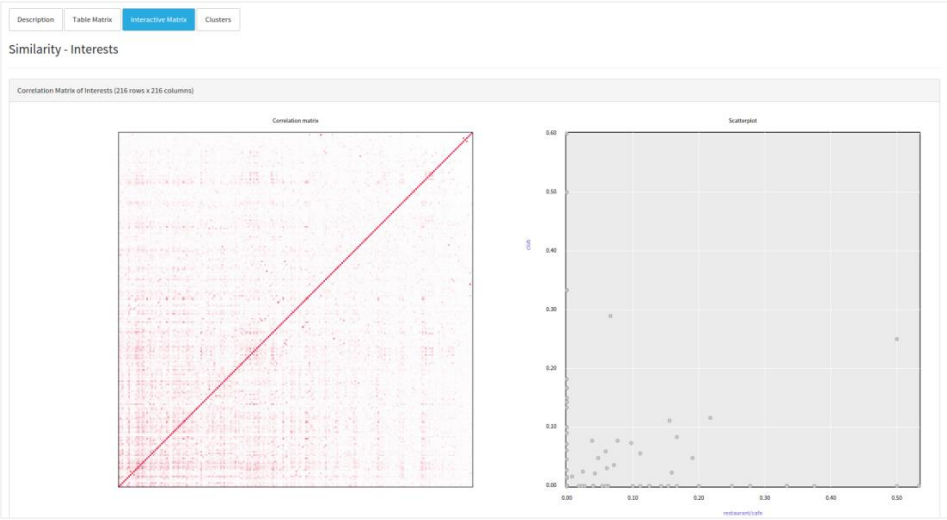


Figura 4.9 - Matriz interativa de correlação entre interesses

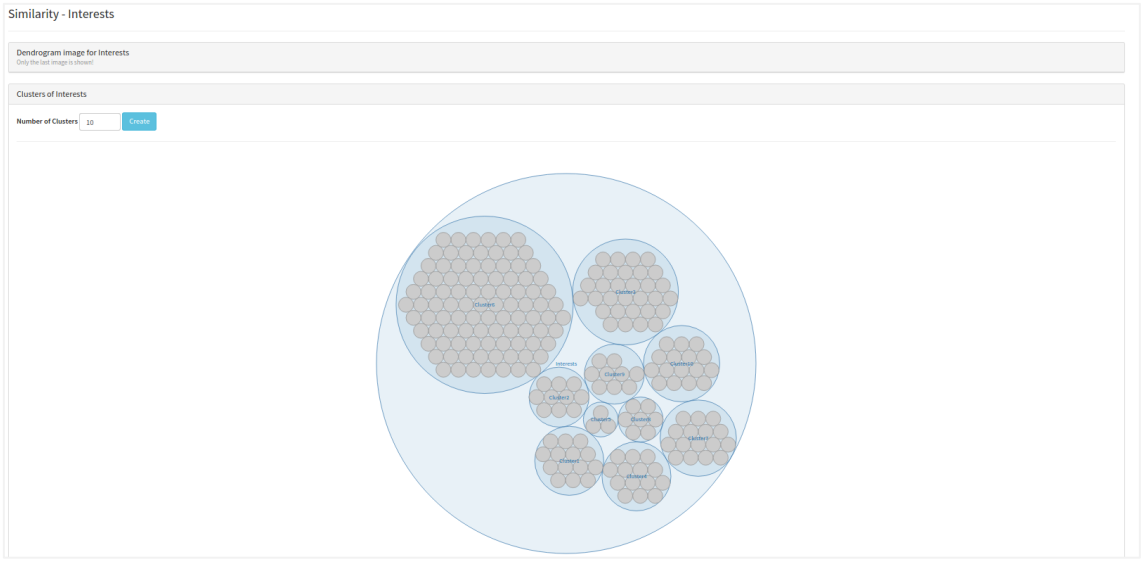


Figura 4.10 - Clusters de interesses

Existe, também, uma zona de resultados (*Evaluation*) de modo a avaliar os modelos criados assim como verificar a performance do sistema de predição de *ratings* baseado num sistema de recomendação (assuntos que serão especificados na secção 5.).

Algorithm	CreatedAt	Name	Params	Evaluation	Time to process *(seconds)*	Silhouette analysis *(image)*
hierarclust	June 25, 2015, 7:27 p.m.	hierarclustModel__20150625-192723-546055	**reduced_matrix_nclusters:** 10 **algorithm:** hierarclust **reduced_matrix:** False **create_silhouettes:** True **affinity:** euclidean **nClusters:** 5 **linkage:** ward	**euclidean:** 0.318 **cosine:** 0.251 **manhattan:** 0.116	0.004	
kmeans	June 25, 2015, 7:27 p.m.	kmeansModel__20150625-192712-631778	**reduced_matrix_nclusters:** 10 **nInit:** 10 **algorithm:** kmeans **reduced_matrix:** True **create_silhouettes:** True **nClusters:** 5 **maxIter:** 300	**euclidean:** 0.265 **cosine:** 0.405 **manhattan:** 0.227	0.04	
birch	June 25, 2015, 1:24 a.m.	birchModel__20150625-012407-575119	**reduced_matrix_nclusters:** 15 **compute_labels:** 1 **algorithm:** birch **reduced_matrix:** True **create_silhouettes:** False **branching_factor:** 50 **nClusters:** 25 **threshold:** 0.2	**euclidean:** 0.136 **cosine:** -0.027 **manhattan:** 0.106	0.167	

Figura 4.11 - Avaliação dos modelos criados

4.4 Considerações finais

Uma vez a estratégia definida, a fase seguinte foi criar uma aplicação para facilitar a obtenção de resultados para responder às perguntas inicialmente propostas. Para isso, inicialmente foi efetuada a modelação do sistema, ou seja, criada a arquitetura geral, os casos de utilização e o diagrama de classes. Como se trata de um protótipo, o foco baseou-se em criar um sistema o mais simples e intuitivo possível. Finalmente, foi possível visualizar algumas das funcionalidades criadas presentes na aplicação através de algumas imagens.

5. Resultados e discussão

"Torture the data, and it will confess to anything."
Ronald Coase
(Economics, Nobel Prize Laureate)

Neste capítulo irão constar os diversos resultados obtidos ao longo do desenvolvimento do projeto assim como uma discussão dos mesmos. Serão, também, descritos os processos de obtenção dos resultados.

5.1 Avaliação de modelos

Um dos principais desafios neste tipo de problemas não supervisionados[36] consiste em efetuar a sua avaliação. A escolha de métricas de forma a ser possível comparar *performances* dos diversos algoritmos que a aplicação suporta não se torna uma tarefa trivial. Deste modo, a estratégia consistiu em criar e testar diversos cenários, diferenciados pela escolha de diferentes parâmetros; seguidamente efetuar o levantamento de diversas métricas, das quais se destacam o valor da silhueta (*vide* Equação 7) e tempo de processamento. Para determinar os valores das métricas foram executados três testes para cada cenário e considerou-se a média dos valores resultante de cada iteração de teste.

Como referido anteriormente, a matriz que representa perfis e interesses é, geralmente, esparsa. Com vista a solucionar este problema optou-se por agrupar interesses baseados nas correlações. Os interesses com maior correlação são inseridos no mesmo grupo, designado por tópico. Cada tópico resulta da soma de todos os valores correspondentes aos interesses. Por exemplo, considere-se que os interesses *musician/band*, *music chart* e *concert tour* têm uma correlação forte, as colunas referentes a estes interesses iriam ser transformadas apenas numa, a qual se chamaria Tópico1. Define-se **matriz completa** como sendo a matriz perfis-interesses e **matriz reduzida** a matriz perfis-tópicos.

As tabelas e gráficos presentes nesta secção representam os resultados de cada cenário para cada um dos algoritmos que a aplicação suporta: K-*Means*, *Clustering* hierárquico, DBSCAN e BIRCH.

Coeficiente da silhueta

A silhueta de um *cluster* foi introduzida como resultado de uma técnica para avaliar visualmente quais os pontos corretamente classificados no *cluster* certo ou não [62], baseando-se no valor da silhueta desses pontos. O valor da silhueta de um dado ponto define a sua proximidade ao seu *cluster* relativamente à proximidade aos outros *clusters*.

[36] O conjunto de dados carece de atributos que permitam aferir a validade do modelo ou modelos criados.

Matematicamente, o valor da silhueta para cada ponto é definido por:

$$S(\boldsymbol{x}) = \frac{b(\boldsymbol{x}) - a(\boldsymbol{x})}{max[a(\boldsymbol{x}), b(\boldsymbol{x})]} \quad (7)$$

onde $a(\boldsymbol{x})$ é a distância média entre o ponto $\boldsymbol{x}$ e todos os outros pontos no seu *cluster* e $b(\boldsymbol{x})$ é o valor mínimo das distâncias médias entre $\boldsymbol{x}$ e os pontos de outros *clusters*.

Para um dado ponto x, o intervalo de valores possíveis da sua silhueta varia entre -1 e 1. Se o valor for próximo de -1, o ponto encontra-se mais próximo de outro *cluster* que aquele a que pertence. Caso o valor é próximo de 1, então a distância média ao seu *cluster* é significativamente mais pequena do que a qualquer outro. Então, quanto maior for o coeficiente da silhueta, mais compactos e separados se encontram os *clusters*.

O coeficiente da silhueta de um *cluster* é definido como a média dos valores da silhueta dos seus pontos. Finalmente, agregando a informação de todos os pontos, o coeficiente da silhueta global é a média dos valores das silhuetas dos *clusters*. [63]

5.1.1 Resultados dos testes

Estipulou-se que, para todos os testes, o número de colunas da matriz reduzida seria 15. Ou seja, os interesses são agrupados através da correlação entre eles de modo a transformar a matriz completa apenas em 15 colunas. Note-se que a matriz completa contém 214 colunas.

Nº total Perfis	3702
Nº perfis repetidos	544
Nº perfis distintos	3158
Nº total Interesses	214
Nº de iterações em cada teste	3
Nº de colunas da matriz reduzida	15

Tabela 5.1 - Parâmetros gerais dos testes

Algoritmo K-Means

Nº máximo de iterações	300
Nº de ciclos com inicialização dos centróides	10

Tabela 5.2 - Parâmetros gerais dos testes para o algoritmo K-Means

Este teste visa testar o algoritmo K-*Means* alterando o número de *clusters* iniciais e o tipo de matriz (completa ou reduzida). Para o primeiro cenário, a métrica para comparar a performance é o valor da silhueta. No segundo cenário, a métrica corresponde ao tempo de processamento (em segundos).

Através da Figura 5.1, que corresponde ao primeiro cenário, é possível observar que, para todos os casos relativamente ao número de *clusters*, o algoritmo apresenta melhores resultados para o caso em que se utilizou a matriz reduzida. Relativamente aos valores da silhueta comparativamente ao número de *clusters*, os valores não sofrem grandes alterações (entre 0.04 e 0.06 para o caso da matriz completa, e entre 0.12 e 0.16 no caso da matriz reduzida).

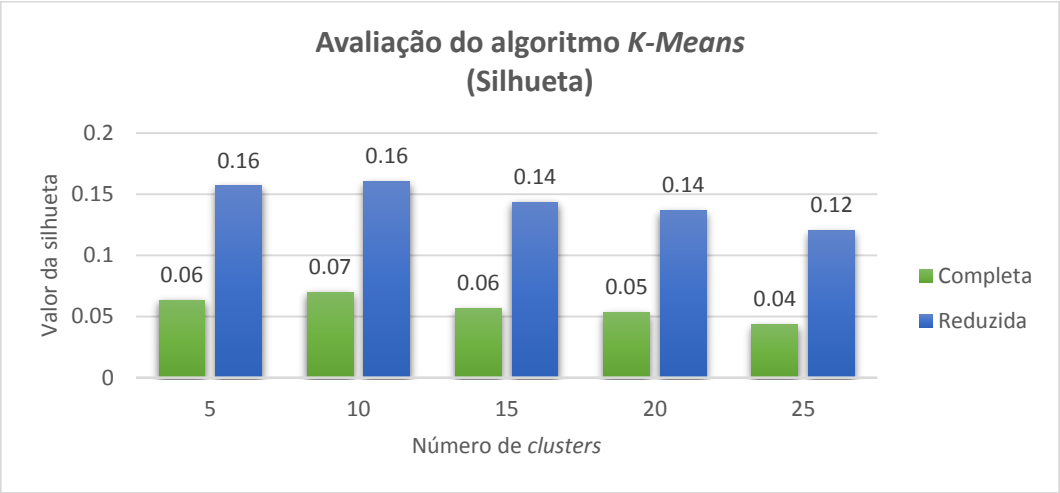


Figura 5.1 - Avaliação do algoritmo K-Means (Silhueta) - Cenário 1

Quanto ao tempo de processamento (Figura 5.2), é possível observar que, para todos os casos relativamente ao número de *clusters*, o algoritmo demora muito menos tempo a ser processado no caso em que se utilizou a matriz reduzida. Se compararmos relativamente ao número de *clusters* formados, o tempo de processamento é (quase) diretamente proporcional ao número de clusters. No caso da matriz completa os valores variam entre 4.02 segundos quando o número de *clusters* é 4 até ao valor máximo de 27.02 segundos no caso em que o número de clusters é 25, ou seja, o tempo de processamento aumenta 672%. Para a matriz reduzida, os valores variam entre 0.24 e 1.55 para o número de *clusters* 5 e 25, respetivamente. Foi um aumento de 646% apesar da grandeza dos valores ser muito inferior ao caso da matriz completa.

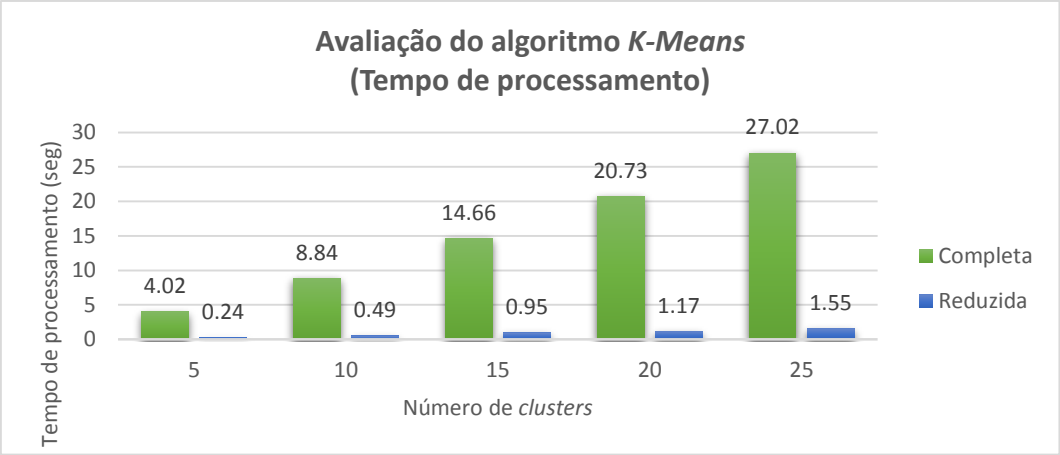


Figura 5.2 - Avaliação do algoritmo K-Means (Tempo de processamento) - Cenário 2

Algoritmo *Clustering* Hierárquico

Ligação	Completo
Métrica	Euclidiana

Tabela 5.3 - Parâmetros gerais dos testes para o algoritmo Clustering Hierárquico

No primeiro cenário (Figura 5.3), nota-se claramente que o algoritmo *clustering* hierárquico perde bastante performance (silhueta) quando a matriz é reduzida. Por exemplo, quando o número de *clusters* é 5, a silhueta desce de 0.69 para 0.3, no entanto o decréscimo ainda é mais acentuado nas outras possibilidades.

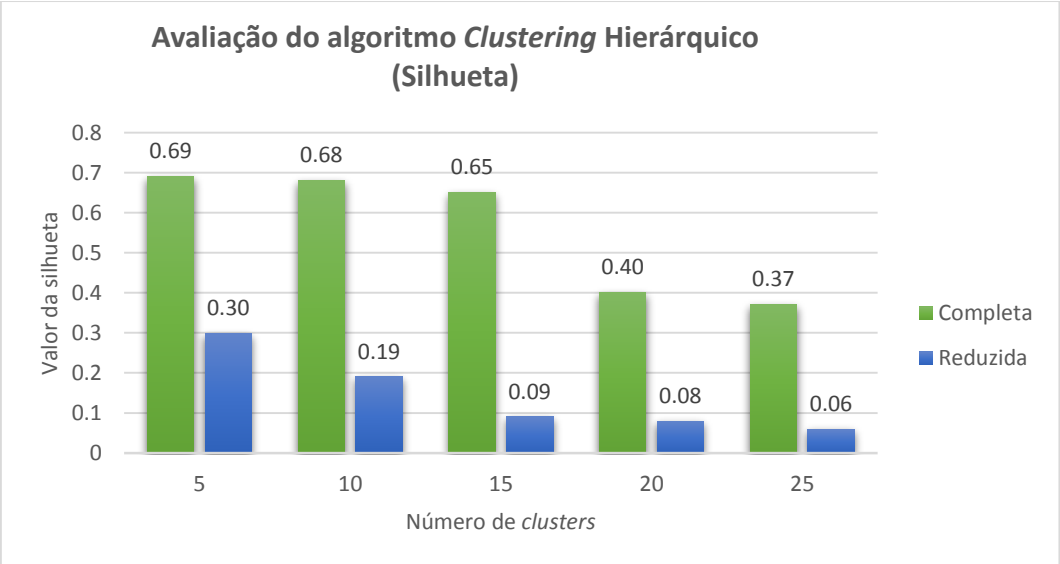


Figura 5.3 - Avaliação do algoritmo Clustering Hierárquico (Silhueta) - Cenário 1

Quanto ao segundo cenário (tempo de processamento), o caso em que se utiliza a matriz reduzida, exceto para o caso do número de *clusters* ser 15, apresenta valores inferiores. No entanto, esses valores não são significativos, como se pode observar na Figura 5.4.

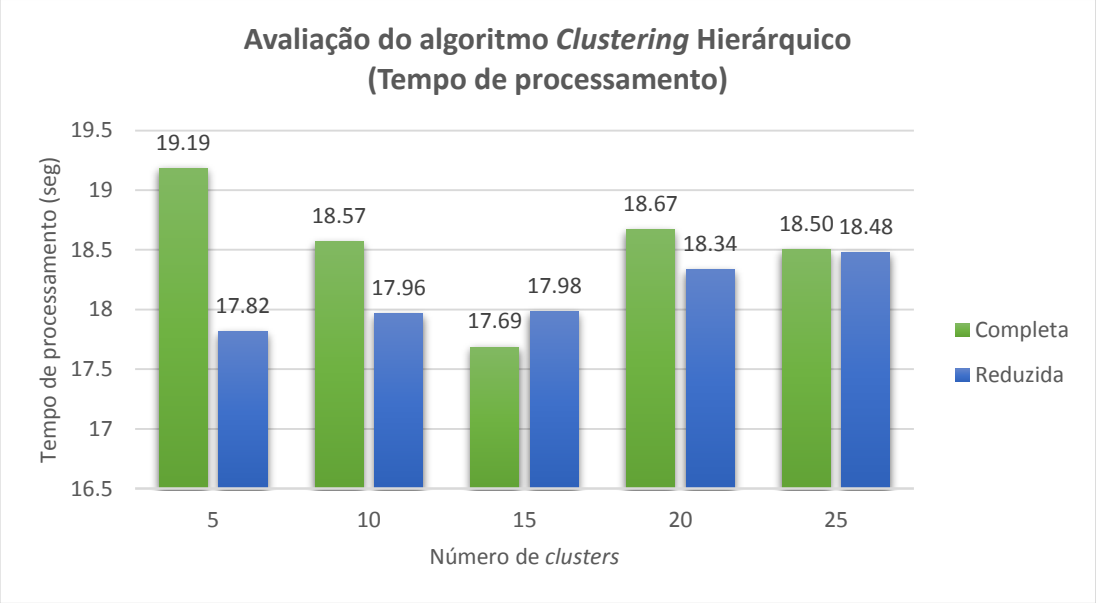


Figura 5.4 - Avaliação do algoritmo Clustering Hierárquico (Tempo de processamento) - Cenário 2

Algoritmo DBSCAN

Nº mínimo de amostras	3
Eps (raio)	0.4
Métrica	Euclidiana

Tabela 5.4 - Parâmetros gerais dos testes para o algoritmo DBSCAN

No primeiro cenário (silhueta), pode-se observar que não houve qualquer diferença utilizando a matriz reduzida ou a completa.

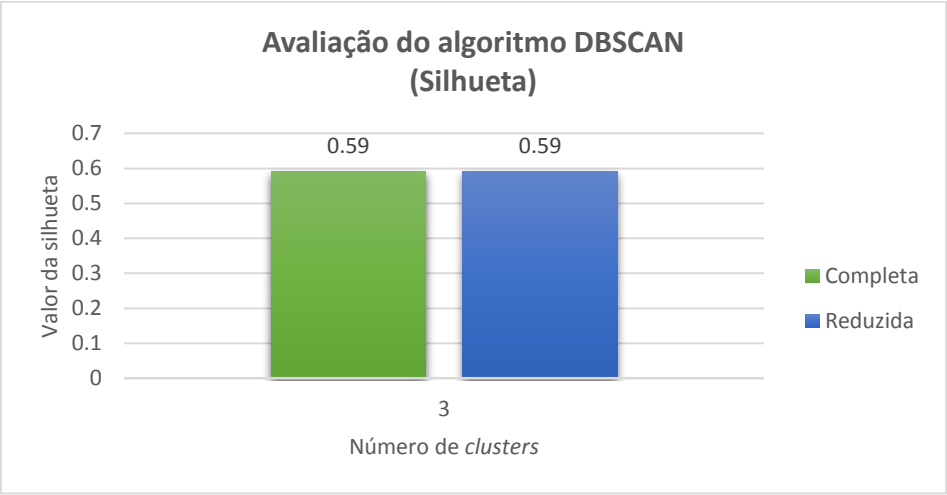


Figura 5.5 - Avaliação do algoritmo DBSCAN (Silhueta) - Cenário 1

Quanto ao segundo cenário (tempo de processamento), os resultados são bastantes diferentes. Utilizando a matriz reduzida, o tempo de processamento é drasticamente reduzido (de 6.15 segundos para 0.78 segundos), como se pode observar na Figura 5.6.

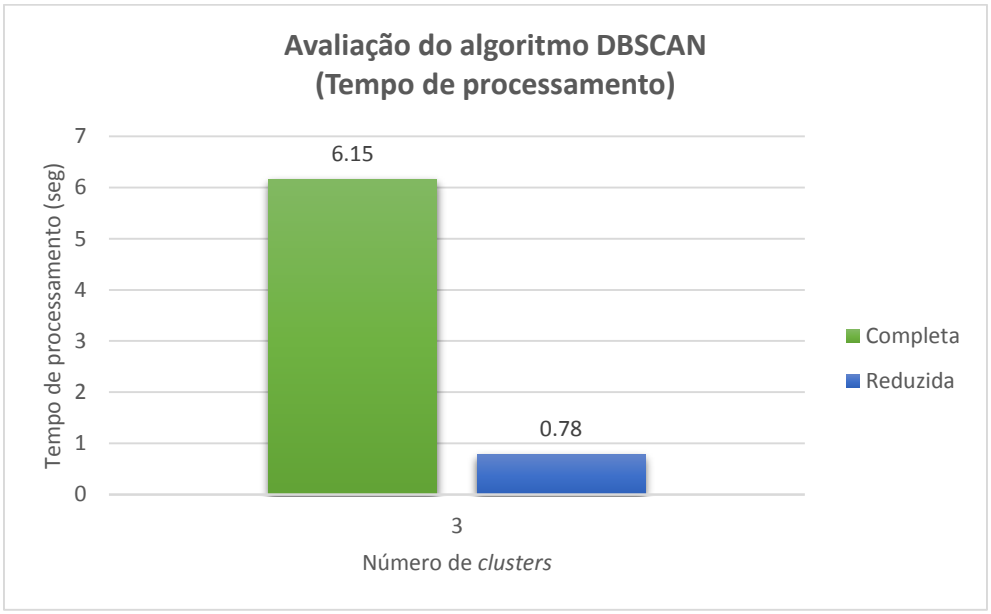


Figura 5.6 - Avaliação do algoritmo DBSCAN (Tempo de processamento) - Cenário 2

Algoritmo BIRCH

Limiar (*Treshold)*	0.2

Tabela 5.5 - Parâmetros gerais dos testes para o algoritmo BIRCH

Considerando o primeiro cenário (silhueta), o valor da silhueta quando se utiliza a matriz reduzida apenas é superior no caso do número de *clusters* ser 10. Nos restantes casos, o valor utilizando a matriz completa é superior. No entanto, o único caso em que existe uma grande diferença é no caso do número de *clusters* ser 5 (0.54 para 0.29), já que nos restantes casos a diferença não é muito grande como se pode observar na Figura 5.7.

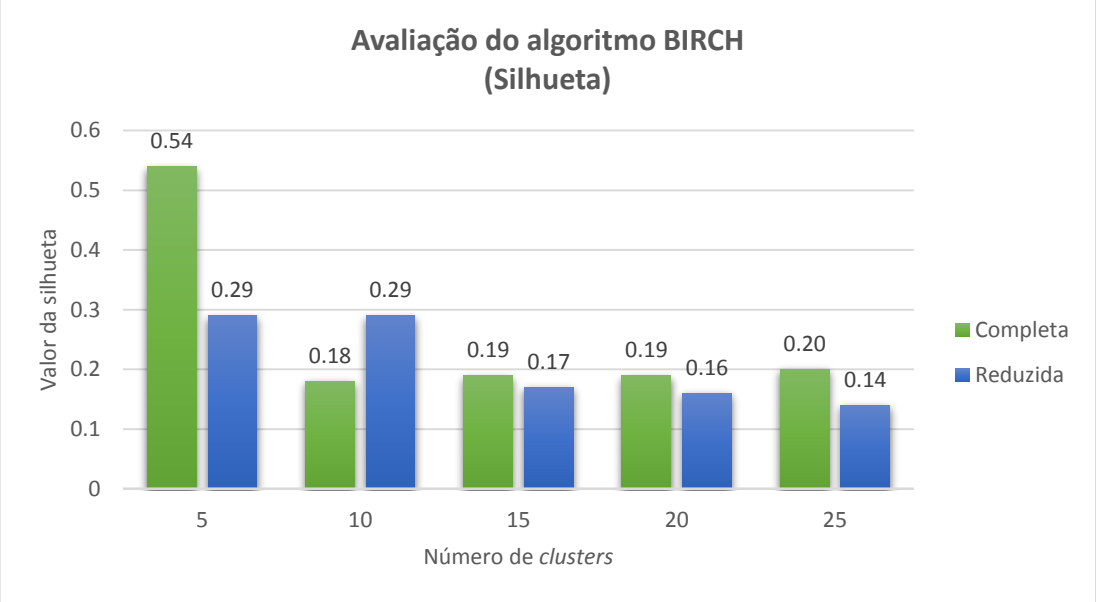


Figura 5.7 - Avaliação do algoritmo BIRCH (Silhueta) - Cenário 1

Quanto ao outro cenário (tempo de processamento), os valores quando se utiliza a matriz completa para todos as hipóteses do número de *clusters* são, sensivelmente, o dobro do caso em que se utiliza a matriz reduzida, como indicam os valores da Figura 5.8.

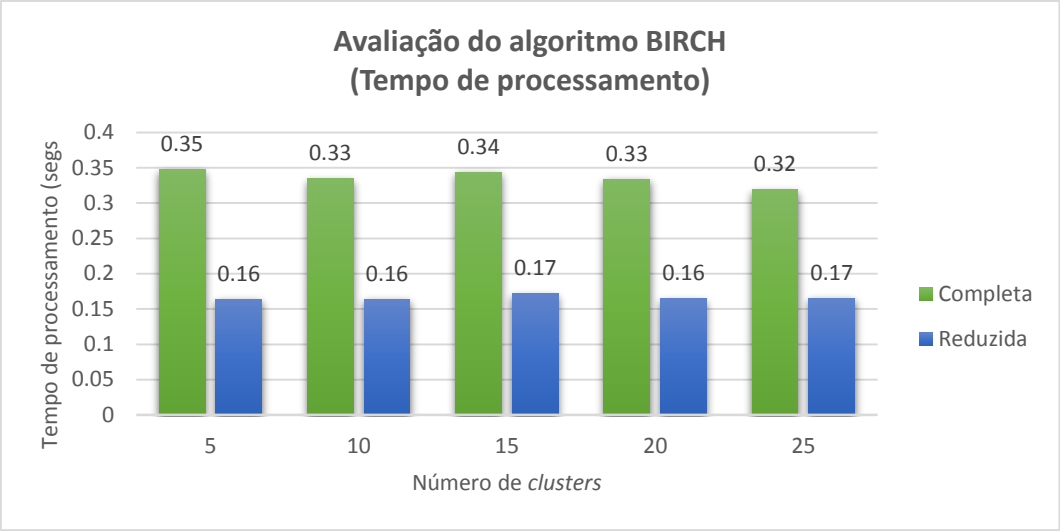


Figura 5.8 - Avaliação do algoritmo BIRCH (Tempo de processamento) - Cenário 2

Comparação dos algoritmos

Caso se faça uma comparação lado a lado de todos os algoritmos relativamente ao cenário em que a métrica de avaliação é o valor da silhueta, pode-se observar que, utilizando a matriz completa, o algoritmo *Clustering* Hierárquico é o que apresenta melhores valores, seguido do DBSCAN, depois o BIRCH e em último o K-*Means*. Quanto ao caso da matriz reduzida, a ordem da lista alterou-se onde o algoritmo com melhor performance foi o DBSCAN, seguido do BIRCH, depois o K-*Means* e, finalmente, o *Clustering* Hierárquico, como se pode observar na Figura 5.9. No entanto, é importante salientar que o número de *clusters* que resultou do DBSCAN foi 3, pelo que este facto é importante na tomada de decisão da escolha do algoritmo.

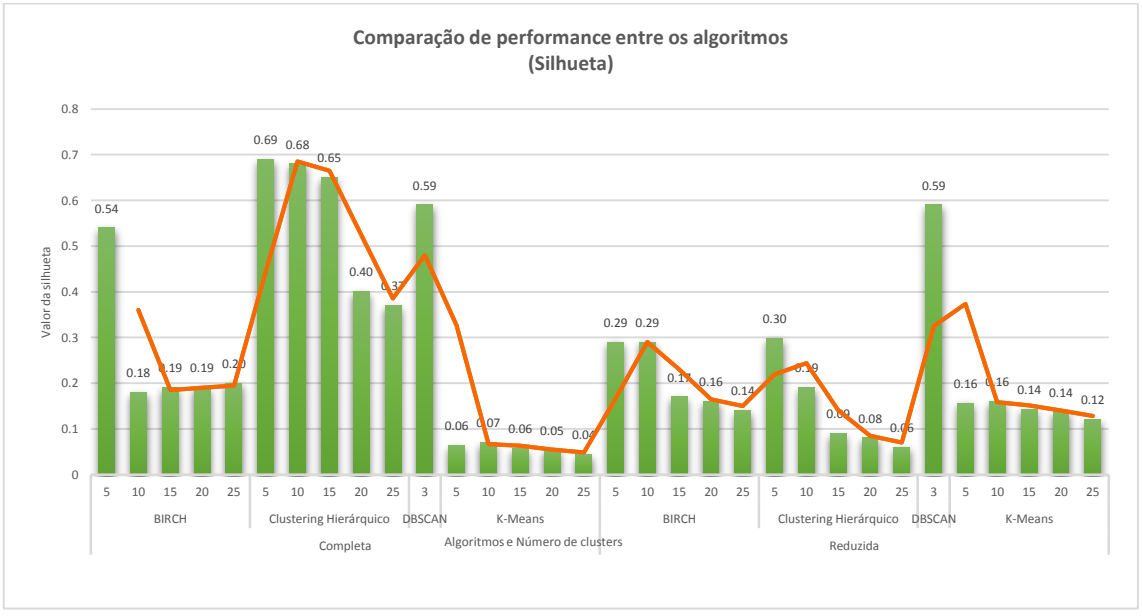


Figura 5.9 - Gráfico de comparação de performance entre algoritmos (silhueta)

No que diz respeito ao segundo cenário (tempo de processamento), é possível observar na Figura 5.10 que, relativamente ao caso da matriz completa, o algoritmo com o tempo mais baixo é o BIRCH, seguido do DBSCAN e depois os outros dois em que o *Clustering* Hierárquico apresenta tempos sem grande alteração enquanto que o tempo de processamento do K-*Means* aumenta progressivamente quando o número de clusters também aumenta. Quando a matriz utilizada é a reduzida então o algoritmo com menor tempo continua a ser o BIRCH, seguido do DBSCAN e K-*Means* e, finalmente, o *Clustering* Hierárquico. É importante referir que o tempo de processamento do algoritmo K-*Means* foi aquele que mais se alterou relativamente à utilização das duas matrizes.

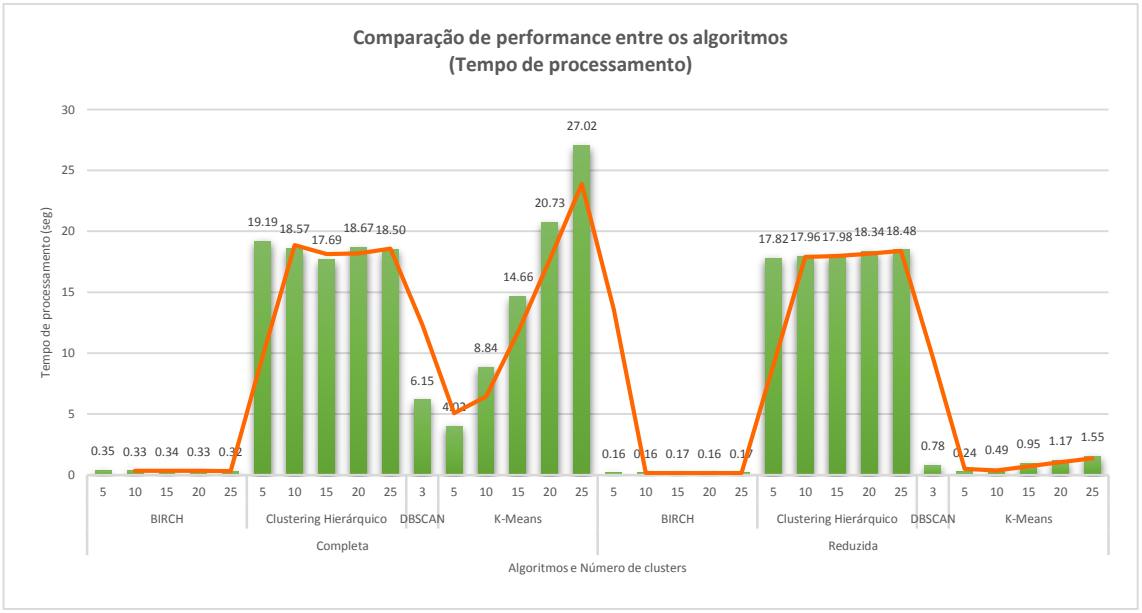


Figura 5.10 - Gráfico de comparação de performance entre algoritmos (tempo de processamento)

5.2 Predição de ratings de interesses

Uma outra abordagem de modo a ser possível testar a aplicação baseou-se em [48], onde são utilizadas técnicas de avaliação de sistemas de recomendação. O objetivo consiste em aferir o grau de eficácia na predição de *ratings* do *items* que não estejam preenchidos pelo perfil (ou cliente) utilizando os valores dos *ratings* dos perfis similares. A estratégia seguida consiste no conjunto dos seguintes passos:

1. **Redução da matriz de perfis**
   É utilizada a matriz perfis-tópicos. O número de colunas deve ser previamente escolhido.
2. **Seleção de um perfil e remoção de diversos valores correspondentes aos interesses**
   Os valores retirados são guardados para serem utilizados no último passo de avaliação da performance. O número de valores a retirar deve ser previamente escolhido.

3. **Cálculo das similaridades entre o perfil escolhido no passo 2 e todos os outros perfis**
   A métrica de distância utilizada é a correlação de *Pearson*.
4. **Filtragem dos perfis em que o valor da similaridade calculado no passo 3 seja superior ao valor zero**
   Consideram-se apenas estes perfis uma vez que o objetivo passa por encontrar os perfis de alguma forma similares. Valores negativos ou zero não indicam qualquer similaridade.
5. **Predição dos diversos valores retirados no passo 2**.
   A fórmula de cálculo para estes valores consistiu na média ponderada entre os valores preenchidos pelos outros perfis e o valor da similaridade. Ou seja, os valores do interesse dos outros perfis similares são multiplicados por um peso (o valor da similaridade) e o valor final é a média destes valores.
6. **Avaliação das predições**
   Utilizando os valores guardados no passo 2 (*ground-truth*) e os valores das predições, são determinadas diversas métricas de avaliação (Equações 8, 9 e 10).

Erro Absoluto Médio (MAE)

Esta métrica corresponde ao valor médio dos erros entre os valores estimados (f) e os esperados (y). O resultado devolvido é um valor real positivo e a solução ótima é o valor 0 (zero). Então, o MAE pode ser definido como:

$$MAE(f, y) = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| \qquad (8)$$

onde f_i é o valor estimado, y_i é o valor esperado e n o número total de valores.

Erro Quadrático Médio (MSE)

Esta métrica corresponde ao valor médio dos erros quadráticos entre os valores estimados e os esperados. O resultado devolvido é um valor real positivo e a solução ótima é o valor 0 (zero). Então, o MSE pode ser definido como:

$$MSE(f, y) = \frac{1}{n}\sum_{i=1}^{n}(f_i - y_i)^2 \qquad (9)$$

onde f_i é o valor estimado, y_i é o valor esperado e n o número total de valores.

Mediana do Erro Absoluto (MedAE)

Esta métrica corresponde ao valor da mediana dos erros entre os valores estimados e os esperados. O resultado devolvido é um valor real positivo e a solução ótima é o valor 0 (zero). Então, o MedAE pode ser definido como:

$$MedAE(f, y) = mediana(|f_1 - y_1|, \ldots, |f_n - y_n| \qquad (10)$$

onde f_i é o valor estimado, y_i é o valor esperado e n o número total de valores.

5.2.1 Resultados das predições

Para realizar os testes, definiu-se que o número de *clusters* deveria variar entre 10 e 15 e a percentagem do número de *items* a prever seria de 30%, 50% e 70%. Ou seja, caso um perfil tenha 10 interesses, caso este parâmetro seja 30% significa que irão ser inferidos os valores de 3 *items*.

Como se pode observar nas Figura 5.11, Figura 5.12 e Figura 5.13, a predição de *ratings* de *items* de um dado perfil a partir dos *ratings* de perfis similares obteve valores de erro bastante próximos de zero para os casos de predição de 30% e 70% do número de *items*.

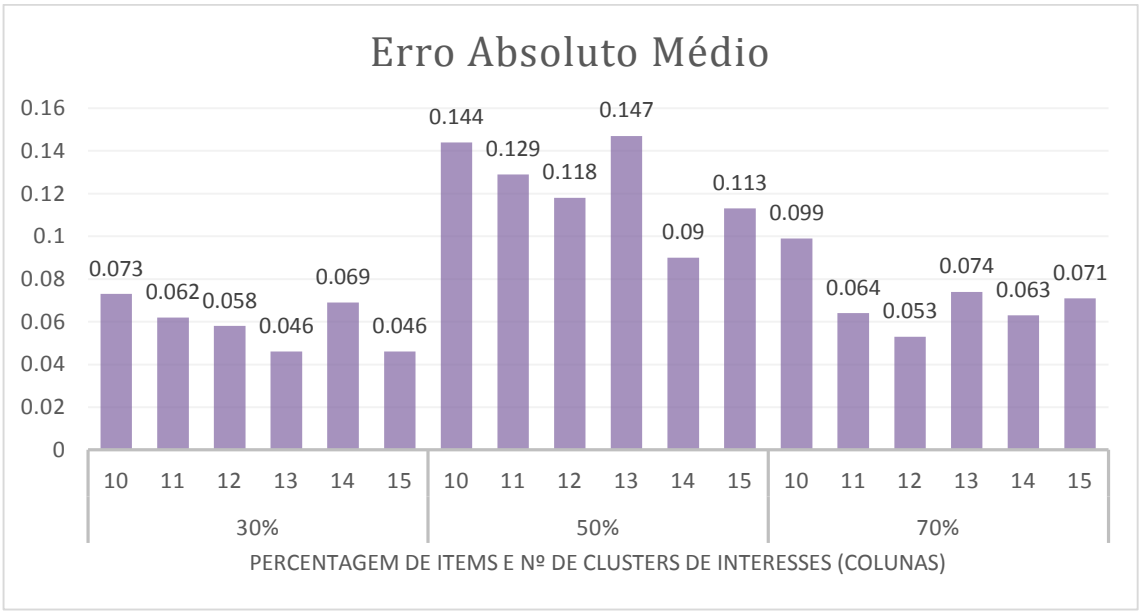


Figura 5.11 - Resultados da predição (Erro Absoluto Médio)

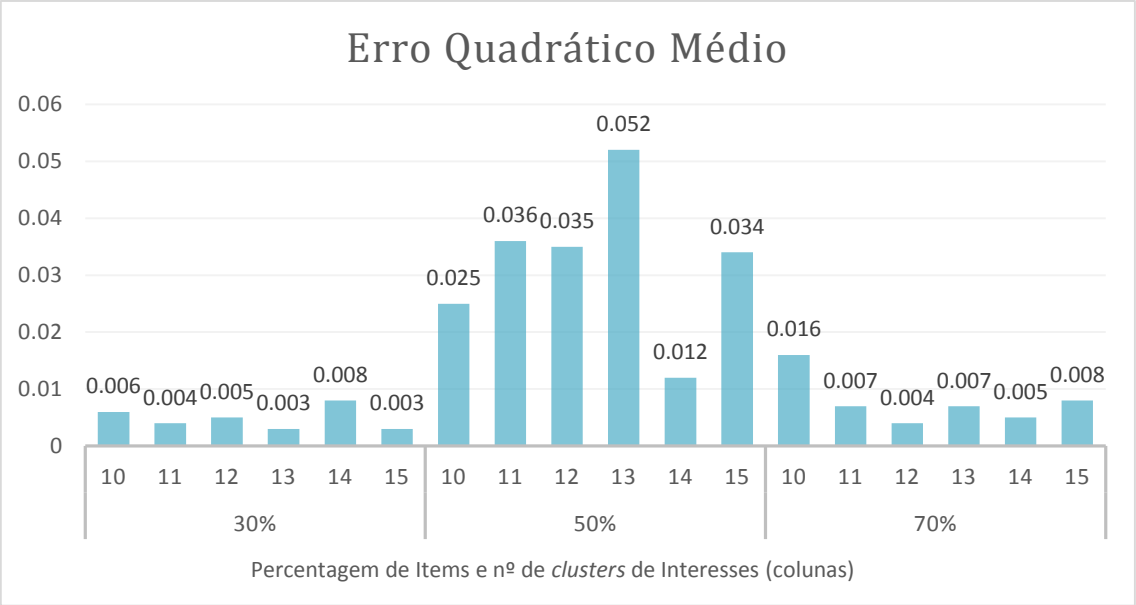


Figura 5.12 - Resultados da predição (Erro Quadrático Médio)

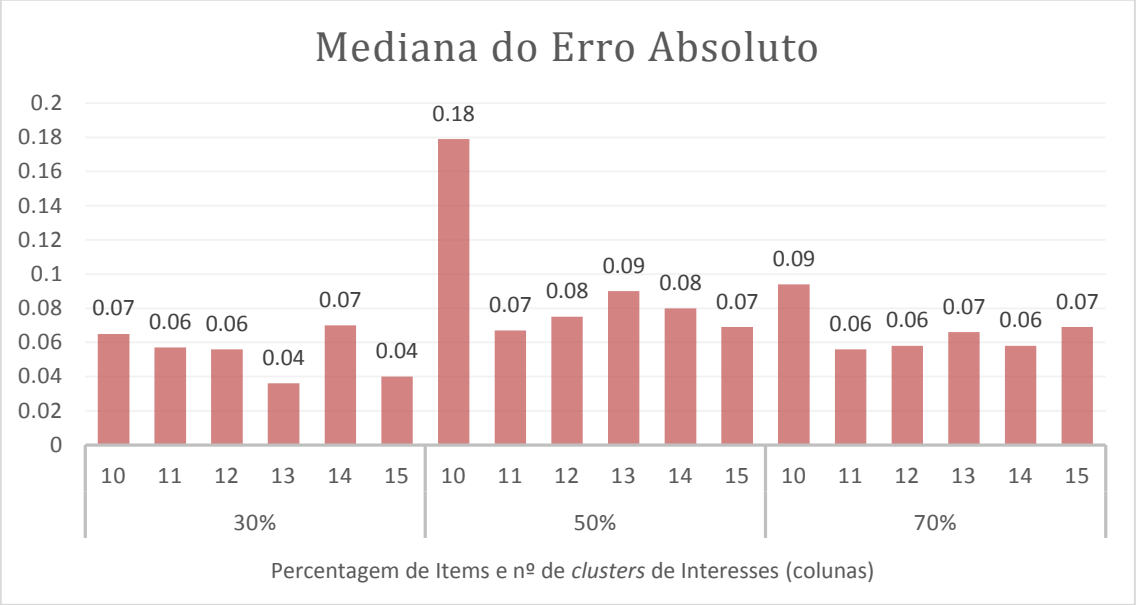


Figura 5.13 - Resultados da predição (Mediana do Erro Absoluto)

5.3 Outros resultados

Como referido anteriormente, a aplicação permite obter *clusters* tanto de perfis como interesses.

Clusters de perfis

Para determinar os *clusters* de perfis é necessário a criação do modelo. Esse modelo pode ser obtido através da utilização de diversos algoritmos de *clustering*, nomeadamente *K-Means*, *Clustering* Hierárquico, DBSCAN e BIRCH, podendo ainda serem escolhidos diferentes argumentos para cada um deles.

Uma vez criado o modelo, é possível visualizar os *clusters* com os diversos perfis. Um exemplo do resultado pode ser visto na Figura 5.14.

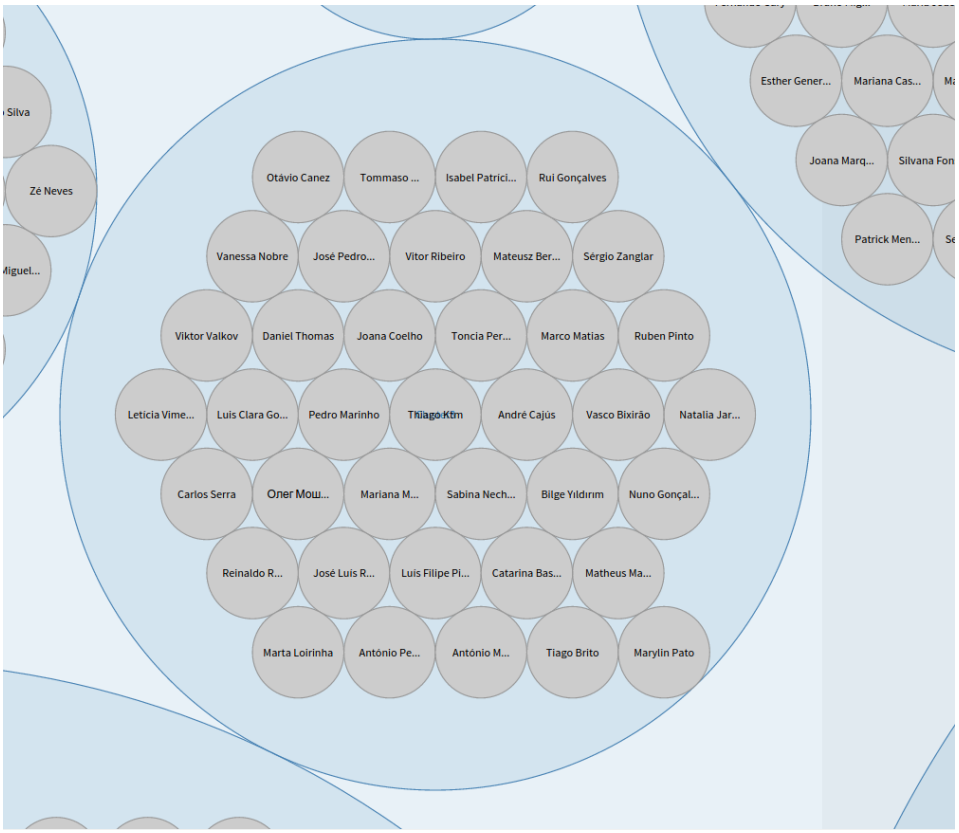


Figura 5.14 –Visualização de clusters de perfis utilizando um modelo baseado em K-Means

Na Figura 5.15 está representado o resultado da funcionalidade onde é possível visualizar as estatísticas de cada *cluster*, ou seja, a soma de cada interesse de todos os perfis pertencentes ao *cluster* respetivo.

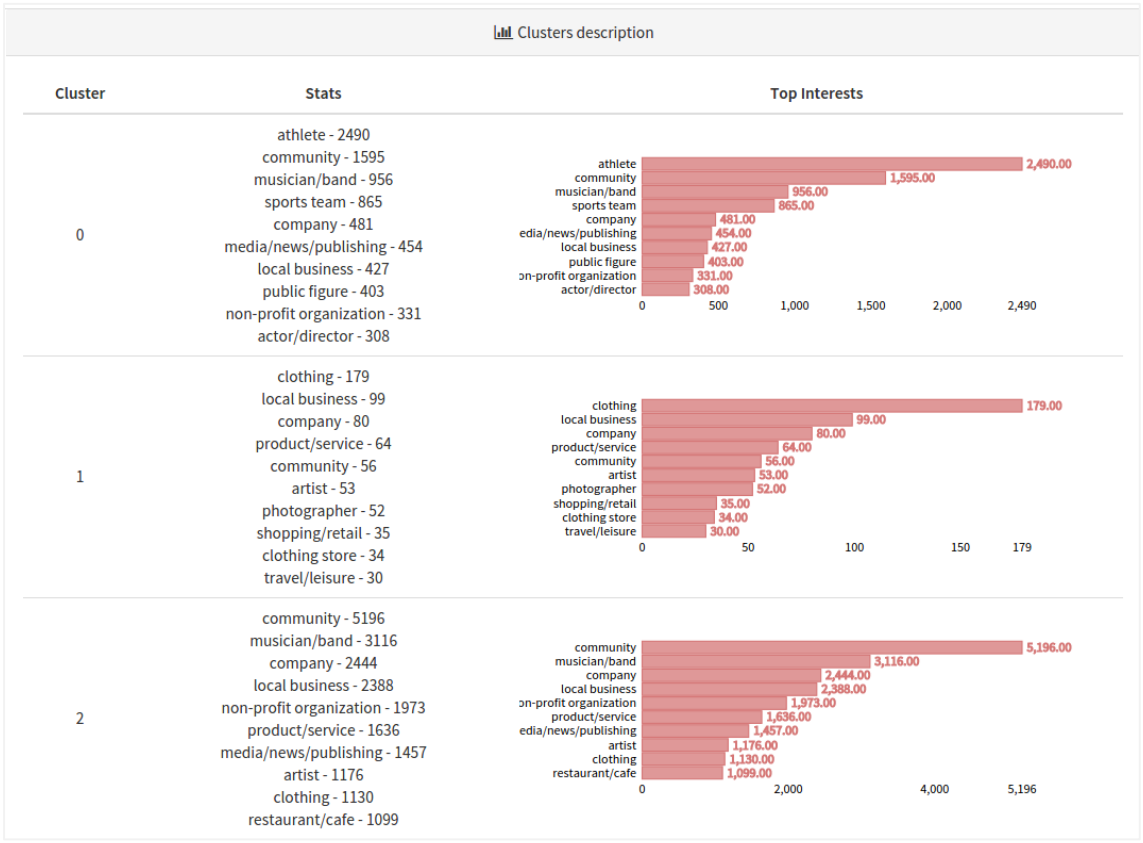


Figura 5.15 – Visualização de estatísticas de clusters de perfis utilizando um modelo baseado em K-Means

5.4 Considerações finais

Neste capítulo, os algoritmos de *clustering* foram comparados em diferentes cenários, isto é, com diferentes valores dos parâmetros iniciais (número de *clusters*, etc.). Sendo um problema esparso utilizou-se uma forma simples de reduzir o número de interesses. Foram criados grupos de interesses (designados por tópicos) e a dimensão da matriz onde foi aplicado este processo foi bastante reduzida. Para avaliar a performance dos diversos modelos de *clustering* foi utilizado o conceito da silhueta e tempos de processamento. Posteriormente, foram efetuados testes comparativos entre os quatro algoritmos. Finalmente, foi adotada uma estratégia baseada em sistemas de recomendação de modo a avaliar a predição de *ratings* de um perfil escolhido aleatoriamente.

6. Conclusões

"An approximate answer to the right problem is worth a good deal more than an exact answer to an approximate problem."
John Tukey

Neste capítulo irão ser descritas as conclusões e, finalmente, irão ser propostas novas funcionalidades ou sugestões como futuro trabalho.

Com o aumento da popularidade das redes sociais *online*, o número de utilizadores e a informação que elas produzem está a crescer a cada minuto. Devido a este facto, diversas áreas, tanto comerciais como científicas, consideram que as redes sociais são uma fonte rica de informação, pelo que são foco de múltiplos estudos. Um desses estudos trata-se da similaridade entre perfis, que não se trata de uma tarefa trivial e pode ser uma ferramenta eficaz para diversos sistemas, tais como de recomendação de produtos ou mesmo identificar casos de doença com os mesmos sintomas.

Neste projeto foi possível, tendo acesso a um conjunto de dados previamente, determinar grupos (*clusters*) de perfis através da sua similaridade. Optou-se por definir um perfil como um conjunto de interesses, pelo que foi possível, também, verificar os interesses correlacionados. Ou seja, foi possível responder à pergunta "Se o perfil X mostrou interesse em A, então será que estará interessado em B?". Por exemplo, foi possível concluir que os interesses *bar* e *arts/entertainment/nightlife* estavam constantemente presentes no mesmo *cluster*, ou analogamente para *public figure* e *actor/director*.

De acordo com os resultados obtidos em 5.1.1, conclui-se que quando o algoritmo é o K-*Means*, é vantajoso utilizar a matriz reduzida, tanto na avaliação do valor da silhueta como no tempo de processamento. No algoritmo de *clustering* hierárquico, utilizar a matriz reduzida diminui a performance relativamente ao valor da silhueta, o tempo de processamento apesar de também ser inferior, não é uma redução significativa. Neste caso específico não se pode concluir que não é vantajoso utilizar a matriz reduzida uma vez que a matriz completa apresenta muitos zeros, e influenciam os resultados. Quanto ao DBSCAN, este apresentou o mesmo valor da silhueta em ambos os casos, mas o tempo de processamento diminui drasticamente quando é utilizada a matriz reduzida. No entanto, como este se trata de um algoritmo que não se escolhe, à partida, o número de *clusters*, a sua utilização deverá ter este fator em conta, o que em alguns casos poderá ser uma vantagem e em outros uma desvantagem. Por exemplo, nos testes efetuados, este algoritmo devolveu sempre 3 *clusters*. Finalmente, quando o algoritmo BIRCH utiliza a matriz reduzida diminui tanto o valor da silhueta como o tempo de processamento, mas a diferença da primeira ia sendo cada vez mais pequena à medida que o número de *clusters* aumentava. Pelo que este algoritmo poderá fazer sentido ser utilizado quando o número de *clusters* pretendido é elevado.

Foi, também, efetuado um teste de performance entre os quatro algoritmos. Utilizando a matriz completa, o algoritmo *Clustering* Hierárquico obteve valores bastante superiores relativamente aos outros, pelo que este

será a escolha acertada. Caso se utilize a matriz reduzida, não existe nenhum que se evidencie quanto aos valores das silhuetas, exceto o DBSCAN que se mantém como no caso anterior. Já nos tempos de processamento, caso este seja um fator determinante, para a matriz completa apenas o BIRCH obteve bons resultados, e na matriz reduzida tanto o BIRCH como o K-*Means* foram os melhores.

Outro teste efetuado diz respeito à predição de *ratings* de items de um dado utilizador. Para isso foram utilizadas técnicas utilizadas em sistemas de recomendação. Os testes mostraram que é possível utilizar os valores estimados para um tópico de um utilizador a partir dos valores para o mesmo tópico de utilizadores semelhantes.

Finalmente, conclui-se que, apesar de todas as dificuldades ao longo do projeto, os objetivos inicialmente propostos foram atingidos sabendo-se que, na era atual da informação, o que hoje é dado como certo ótimo poderá não o ser amanhã. Pelo que o "caminho faz-se caminhando" e o sucesso passa por nunca parar de aprender.

6.1 Sugestões para trabalho futuro

De modo a melhorar a aplicação atual, os seguintes parágrafos servirão como sugestões para futuro trabalho.

No projeto atual apenas se consideram os interesses como representantes do perfil, pelo que uma sugestão passa por incluir, no cálculo da similaridade, outros atributos tais como amizades, eventos ou atividades.

Uma funcionalidade interessante passa por permitir a criação de cenários definidos por conjuntos de atributos do perfil para comparar. Por exemplo, no primeiro cenário poderiam ser utilizados a profissão, estado civil e interesses dos perfis, num segundo cenário podia ser considerado a idade, habilitações literárias, sexo e interesses ou um terceiro apenas com os interesses. O sistema tinha que ser dinâmico na escolha dos algoritmos de *clustering* para construir os modelos (meta-aprendizagem [7]), ou seja, decidir qual o melhor algoritmo perante o cenário em causa.

Outra sugestão passa por utilizar a técnica supracitada em 2.1.3, ou seja, a utilização dos algoritmos ***Min-Hash*** e ***Locality-Sensitive Hashing*** para determinar os perfis candidatos a similares. Com esta técnica o tempo e custo de processamento iria ser reduzido uma vez que apenas era efetuado o cálculo de similaridade entre o perfil e os candidatos. Permitiria, também, escalar a aplicação. Diversos estudos [64] [65] utilizam algoritmos de *clustering* suportados por esta aplicação (DBSCAN e *Hierarchical Clustering*) juntamente com o LSH, pelo que poderiam servir de apoio na implementação desta técnica.

Disponibilizar uma API onde fosse possível comunicar com aplicações externas tornar-se-ia, certamente, vantajoso para alargar o leque de opções do projeto. Essas aplicações poderiam, por exemplo, inserir um perfil para que o sistema devolvesse um conjunto de perfis similares, possibilidade de exportar ou importar modelos, entre outras funcionalidades.

Outra sugestão passa pela criação de um módulo que fornecesse funcionalidades de ligação a APIs de redes sociais para obter dados diretamente dessas redes e não apenas através de ficheiros *Yml* ou *Json*.

Finalmente, utilizando a funcionalidade de predição de *ratings* de interesses, a sugestão passa por adicionar a funcionalidade que iria permitir a recomendação automática de interesses, ou seja, entraria um dado perfil, o resultado seria uma lista de interesses baseada no *top* dos ratings que foram inferidos. Adicionalmente, disponibilizando uma API, esta funcionalidade poderia servir como motor de recomendação de conteúdos a diversas plataformas.

Bibliografia

[1] P. Resnick, H. R. Varian, and G. Editors, “Recommender Systems mmende tems,” *Commun. ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[2] a. Kobsa, “Generic user modeling systems,” pp. 136–154, 2007.

[3] P. De Meo, D. Rosaci, and G. M. L. Sarn, “An XML-Based Adaptive Multi-agent System for Handling E-commerce Activities,” pp. 152–166.

[4] S. Klenk, J. Dippon, P. Fritz, and G. Heidemann, “Determining patient similarity in medical social networks,” in *CEUR Workshop Proceedings*, 2010, vol. 572, pp. 6–13.

[5] A. M. Kaplan and M. Haenlein, “Users of the world, unite! The challenges and opportunities of Social Media,” *Bus. Horiz.*, vol. 53, no. 1, pp. 59–68, 2010.

[6] R. D. S. Villaca, L. B. De Paula, R. Pasquini, and M. F. Magalhaes, “A Similarity Search System Based on the Hamming Distance of Social Profiles,” *2013 IEEE Seventh Int. Conf. Semant. Comput.*, pp. 90–93, Sep. 2013.

[7] J. Gama, A. P. de L. Carvalho, K. Facelli, L. Márcia, and A. C. Lorena, “Extração de Conhecimento de Base de dados - Data Mining,” in *Extração de Conhecimento de Base de dados - Data Mining*, 1ª ed., M. Robalo, Ed. Lisboa: Sílabo, Edições, 2012, pp. 236–242;285–311.

[8] C. Tanner, I. Litvin, and A. Joshi, “Social Networks: Finding Highly Similar Users and Their Inherent Patterns,” *Soc. Networks*, 2008.

[9] B. Bringmann, M. Berlingerio, F. Bonchi, and A. Gionis, “Learning and predicting the evolution of social networks,” in *IEEE Intelligent Systems*, 2010, vol. 25, no. 4, pp. 26–34.

[10] M. Wischenbart, S. Lechner, S. Mitsch, E. Kapsammer, A. Kusel, B. Pröll, W. Retschitzegger, W. Schwinger, J. Schönböck, and M. Wimmer, “User profile integration made easy,” *21St Int. Conf. Companion*, pp. 939–948, 2012.

[11] F. Orlandi, J. Breslin, and A. Passant, “Aggregated, interoperable and multi-domain user profiles for the social web,” *Proc. 8th Int. Conf. Semant. Syst. - I-SEMANTICS ’12*, p. 41, 2012.

[12] T. Vander Wal, “Folksonomy. Technical Report.” [Online]. Available: http://vanderwal.net/folksonomy.html. [Accessed: 14-Apr-2015].

[13] F. Abel, E. Herder, G.-J. Houben, N. Henze, and D. Krause, “Cross-system user modeling and personalization on the Social Web,” *User Model. User-adapt. Interact.*, vol. 23, no. 2–3, pp. 169–209, Nov. 2012.

[14] C. S. Firan, W. Nejdl, and R. Paiu, “The benefit of using tag-based profiles,” *Proc. - 2007 Lat. Am. Web Conf. LA-WEB 2007*, no. Section 6, pp. 32–41, 2007.

[15] F. Carmagnola, F. Cena, L. Console, O. Cortassa, C. Gena, A. Goy, I. Torre, A. Toso, and F. Vernero, “Tag-based user modeling for social multi-device adaptive guides,” *User Model. User-Adapted Interact.*, vol. 18, no. 5, pp. 497–538, 2008.

[16] L. Specia and E. Motta, “Integrating folksonomies with the semantic web,” no. September 2006, 2007.

[17] M. Szomszor, I. Cantador, and H. Alani, “Correlating user profiles from multiple folksonomies,” 2008.

[18] S. Noor and K. Martinez, “Using Social Data as Context for Making Cultural Heritage Recommendations: An Ontology based Approach,” pp. 2–3, 2009.

[19] G. Srinivas, N. Tandon, and V. Varma, “A weighted tag similarity measure based on a collaborative weight model,” *Proc. 2nd Int. Work. Search Min. user-generated contents - SMUC ’10*, p. 79, 2010.

[20] D. Brickley and L. Miller, “FOAF Vocabulary Specification,” *Namesp. Doc.*, vol. 3, p. http://xmlns.com/foaf/spec/, 2010.

[21] U. Bojars, J. Breslin, D. Berrueta, and D. Brickley, “SIOC core ontology specification,” *WC Memb.*, pp. 1–15, 2007.

[22] D. Heckmann, T. Schwartz, B. Brandherm, M. Schmitz, and M. von Wilamowitz-Moellendorff, “Gumo - The General User Model Ontology,” *Proc. {UM} 2005 10th {I}nternational {C}onference {U}ser Model.*, vol. 3538, pp. 428–432, 2005.

[23] P. De Meo, A. Nocera, G. Terracina, and D. Ursino, “Recommendation of similar users, resources and social networks in a Social Internetworking Scenario,” *Inf. Sci. (Ny).*, vol. 181, no. 7, pp. 1285–1305, Apr. 2011.

[24] M. T. Group, M. T. Group, U. P. Fabra, and U. P. Fabra, “Foa ng the Music: Bridging the Semantic Gap in Music Recommendation `,” pp. 927–934, 2006.

[25] P. Bhattacharyya, A. Garg, and S. F. Wu, “Analysis of User Keyword in Online Social Networks,” *Soc. Networks Anal. Min. J. (by Springer)*, 2010.

[26] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel, “You are who you know,” *Proc. third ACM Int. Conf. Web search data Min. - WSDM ’10*, pp. 251–260, 2010.

[27] M. Mcpherson, L. Smith-lovin, and J. M. Cook, “Homophily in Social Networks,” *Phaedrus*, vol. 27, pp. 415–444, 2001.

[28] A. Fiore and J. Donath, “Homophily in online dating: when do you like someone like yourself?,” *CHI’05 Ext. Abstr. Hum. Factors …*, pp. 1–4, 2005.

[29] J. Lindamood, R. Heatherly, and M. Kantarcioglu, “Inferring Private Information Using Social Network Data,” *Acm*, pp. 1145–1146, 2009.

[30] C. G. Akcora, B. Carminati, and E. Ferrari, “User similarities on social networks,” *Soc. Netw. Anal. Min.*, vol. 3, no. 3, pp. 475–495, Jan. 2013.

[31] J. Kim, D. Choi, B. Ko, E. Lee, and P. Kim, “Extracting User Interests on Facebook,” *Int. J. Distrib. Sens. Networks*, vol. 2014, pp. 1–5, 2014.

[32] P. De Meo, E. Ferrara, and G. Fiumara, “Finding Similar Users in Facebook,” M. Safar and K. Mahdi, Eds. IGI Global, 2011.

[33] W. Geyer, C. Dugan, D. R. Millen, M. Muller, and J. Freyne, “Recommending topics for self-descriptions in online user profiles,” *Proc. 2008 ACM Conf. Recomm. Syst. - RecSys ’08*, pp. 59–66, 2008.

[34] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri, “Feedback effects between similarity and social influence in online communities,” *Proceeding 14th ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD 08*, p. 160, 2008.

[35] S. Knecht, J. Reinholz, and P. Kenning, “The spread of obesity in a social network.,” *N. Engl. J. Med.*, vol. 357, no. 18, pp. 1866–1867; author reply 1867–1868, 2007.

[36] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins, “Geographic routing in social networks.,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 102, no. 33, pp. 11623–11628, 2005.

[37] M. de Gemmis, P. Lops, G. Semeraro, P. Basile, and M. Degemmis, “Integrating tags in a semantic content-based recommender,” *Proc. 2008 ACM Conf. Recomm. Syst. - RecSys ’08*, p. 163, 2008.

[38] V. Zanardi and L. Capra, “Social Ranking: Uncovering Relevant Content Using Tag-based Recommender Systems,” pp. 51–58, 2008.

[39] J. Vosecky, D. Hong, and V. Shen, “User identification across multiple social networks,” *Networked Digit. Technol. …*, pp. 360–365, 2009.

[40] V. A. Dabeeru, “User Profile Relationships using String Similarity Metrics in Social Networks,” *CoRR*, vol. abs/1408.3, 2014.

[41] C. Stanfill and D. Waltz, “Toward memory-based reasoning,” *Commun. ACM*, vol. 29, no. 12, pp. 1213–1228, 1986.

[42] S. Boriah, “Similarity Measures for Categorical Data : A Comparative Evaluation,” *Distribution*, 2008.

[43] H. Xie, X. Li, J. Wang, Q. Li, and Y. Cai, “The Collaborative Search by Tag-Based User Profile in Social Media,” *Sci. World J.*, vol. 2014, no. iii, pp. 1–7, 2014.

[44] A. Savvopoulos, M. Virvou, D. N. Sotiropoulos, and G. A. Tsihrintzis, “Clustering for user modeling in recommender e-commerce application: {A} RUP-based intelligent software life-cycle,” in *Knowledge-Based Software Engineering, Proceedings of the Eighth Joint Conference on Knowledge-Based Software Engineering, {JCKBSE} 2008, August 25-28, 2008, University of Piraeus, Piraeus, Greece*, 2008, pp. 295–304.

[45] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang, “Finding interesting associations without support pruning,” *Knowl. Data Eng. IEEE Trans.*, vol. 13, no. 1, pp. 64–78, Jan. 2001.

[46] P. Bonhard, C. Harries, J. McCarthy, and M. Sasse, “Accounting for taste: using profile similarity to improve recommender systems.,” 2006.

[47] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*, vol. 67. 2011.

[48] X. Su and T. M. Khoshgoftaar, “A Survey of Collaborative Filtering Techniques,” *Adv. Artif. Intell.*, vol. 2009, no. Section 3, pp. 1–19, 2009.

[49] D. Billsus, D. Billsus, M. J. Pazzani, and M. J. Pazzani, “Learning collaborative information filters,” *Proc. Fifteenth Int. Conf. Mach. Learn.*, vol. 54, p. 47, 1998.

[50] A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke, “Personalized recommendation in social tagging systems using hierarchical clustering,” *Proc. 2008 ACM Conf. Recomm. Syst. - RecSys ’08*, p. 259, 2008.

[51] C. Hung, Y. Huang, J. Hsu, and D. Wu, “Tag-based user profiling for social media recommendation,” *Work. Intell. …*, pp. 49–55, 2008.

[52] K. H. Zou, K. Tuncali, and S. G. Silverman, “Correlation and simple linear regression.,” *Radiology*, vol. 227, no. 3, pp. 617–622, 2003.

[53] J. Aldrich, “Correlations Genuine and Spurious in Pearson and Yule.” 1995.

[54] S. Alsaleh, R. Nayak, and Y. Xu, “Grouping people in social networks using a weighted multi-constraints clustering method,” *2012 IEEE Int. Conf. Fuzzy Syst.*, pp. 1–8, 2012.

[55] P.-N. Tan, M. Steinbach, and V. Kumar, “Chap 8 : Cluster Analysis: Basic Concepts and Algorithms,” *Introd. to Data Min.*, p. Chapter 8, 2005.

[56] V. Eck, N. Jan, and L. Waltman, “Clustering Users in Twitter Based on Interests,” *J. Am. Soc. Inf. Sci. Technol.*, vol. 59, no. 10, pp. 1653–1661, 2008.

[57] H. Jiawei, K. Micheline, and P. Jian, *Data mining : concepts and techniques*, 3rd ed. Wyman Street,Waltham, 2012.

[58] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: An Efficient Data Clustering Method for Very Large Databases,” *Proc. 1996 ACM SIGMOD Int. Conf. Manag. Data*, vol. 1, pp. 103–114, 1996.

[59] H. Bauke and S. Mertens, “Cluster Computing,” *Vasa*, vol. 2, pp. 7–9, 2006.

[60] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, vol. 3. 1998.

[61] C. P. Caldeira, *PostgreSQL – Guia Fundamental*. Lisboa: Sílabo, Edições, 2015.

[62] A. Struyf, M. Hubert, and P. J. Rousseeuw, “Integrating robust clustering techniques in S-PLUS,” *Comput. Stat. Data Anal.*, vol. 26, no. 1, pp. 17–37, 1997.

[63] L. Dalton, V. Ballarin, and M. Brun, “Clustering algorithms: on learning, validation, performance, and applications to genomics.,” *Curr. Genomics*, vol. 10, no. 6, pp. 430–445, 2009.

[64] Y. P. Wu, J. J. Guo, and X. J. Zhang, “A linear DBSCAN algorithm based on LSH,” *Proc. Sixth Int. Conf. Mach. Learn. Cybern. ICMLC 2007*, vol. 5, no. August, pp. 2608–2614, 2007.

[65] H. Koga, T. Ishibashi, and T. Watanabe, “Using Locality-Sensitive Hashing,” pp. 114–128, 2004.

[66] International Standards Organization, “ISO 8601:2004(E) Data elements and interchange formats - Information interchange - Representation of dates and times,” vol. 2004, p. 40, 2004.

Anexos

A) Lista completa de todos os atributos presentes nos ficheiros cedidos pela empresa Ubiprism.

Atributo	Descrição
facebookId	Identificador único na rede social *Facebook*.
fbName	Nome do utilizador.
gender	Género do utilizador.
birthday	Data de nascimento do utilizador.
fbFriends	Lista de amigos que o utilizador possui.
fbNumberOfFriends	Número de amigos que o utilizador possui na rede social.
fbGroups	Grupos que o utilizador pertence.
fbDevices	Dispositivos móveis associados ao perfil do *Facebook*.
fbPages	Páginas que o utilizador fez *Like*.
fbMaritalStatus	Representa o estado civil do utilizador.
fbInterests	Representa os interesses do utilizador categorizados por diversos temas (filmes, música, livros, etc.).
fbEducation	Campo onde o utilizador pode colocar o histórico a nível académico, desde o secundário, a curso superior ou outros cursos que tenha realizado.
fbWorkHistory	Histórico profissional do utilizador.
_id	Identificador do elemento.
createDate	Data de criação do elemento.
updateDate	Data de última atualização do elemento.
email	Endereço eletrónico do utilizador.
access_token	*Token* de acesso da aplicação ao perfil do utilizador.
fbUsername	"Alcunha" do utilizador na rede social *Facebook*.
fbLocation	Localização do utilizador. Geralmente, o local onde vive.
fbFamily	Representa as associações familiares com outros utilizadores da rede social.
fbHometown	Campo que representa a localidade de nascimento.
fbSubscribers	Representa o conjunto de subscritores que subscreveram o utilizador.
fbNumberOfSubscribers	Representa o número de subscritores que subscreveram o utilizador.
fbSubscribedTo	Representa o conjunto de subscritores que o utilizador subscreveu.
fbNumberOfSubscribedTo	Representa o número de subscritores que o utilizador subscreveu.
fbPolitical	Corresponde aos ideais políticos do utilizador.
fbReligion	Identifica a religião correspondente ao utilizador.

Legenda: Tabela completa dos atributos presentes no conjunto de dados

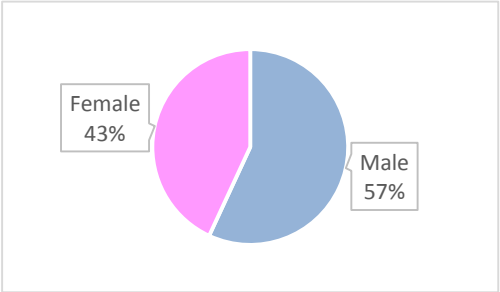
B) Lista completa das estatísticas dos atributos do perfil

gender

Este campo representa o género do utilizador. A taxa de preenchimento é bastante considerável (95,79%).

Total de dados	3702
Dados em falta	156
Taxa de preenchimento	95.79%
Moda	*male* (2018)

Legenda: Estatísticas do atributo "gender"



Legenda: Gráfico do atributo "gender"

fbFriends

Lista de amigos que o utilizador possui na rede social *Facebook*.

Total de dados	3702
Dados em falta	3686
Taxa de preenchimento	0.38%

Legenda: Estatísticas do atributo "fbFriends"

fbPages

Páginas que o utilizador fez "Like", ou seja, são páginas das quais o utilizador demonstrou interesse. Estas páginas encontram-se associadas a uma categoria (previamente definida pela rede social *Facebook*).

Total de dados	3702
Dados em falta	542
Taxa de preenchimento	85.36%

Legenda: Estatísticas do atributo "fbPages"

fbInterests

Este campo representa os interesses do utilizador categorizados por diversos temas (filmes, música, livros, etc.).

Total de dados	3702
Dados em falta	3364
Taxa de preenchimento	9.13%

Legenda: Estatísticas do atributo "fbInterests"

birthday

Este campo representa a data de nascimento do utilizador.

Total de dados	3702
Dados em falta	772
Taxa de preenchimento	79.15%

Legenda: Estatísticas do atributo "birthday"

age

Este campo representa a idade do utilizador.

Total de dados	3702
Dados em falta	1679
Taxa de preenchimento	54.65%
Máximo	110
Mínimo	15
Média	30.18
Desvio Padrão	8.77
Mediana	29
Moda	25

Legenda: Estatísticas do atributo "age"

fbUsername

Este campo representa o nome de utilizador.

Total de dados	3702
Dados em falta	267
Taxa de preenchimento	92.79%

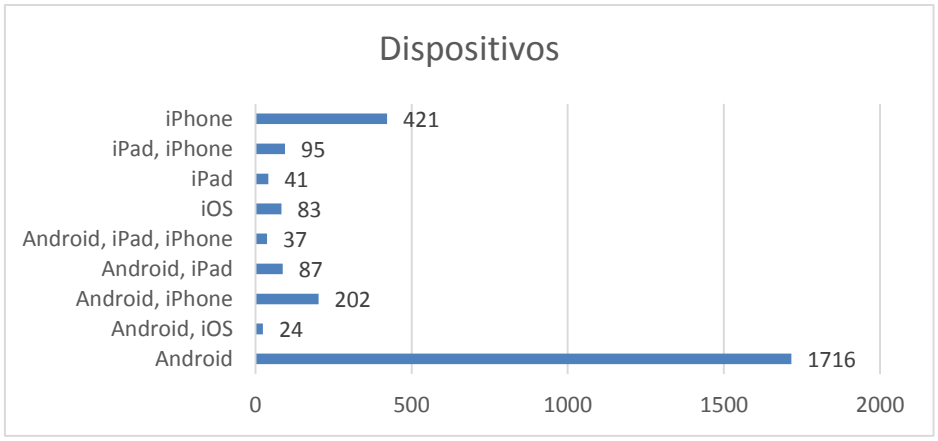
Legenda: Estatísticas do atributo "fbUsername"

fbDevices

Este campo representa os dispositivos associados à conta do utilizador.

Total de dados	3702
Dados em falta	996
Taxa de preenchimento	33.10%
Moda	Android (1716)

Legenda: Estatísticas do atributo "fbDevices"



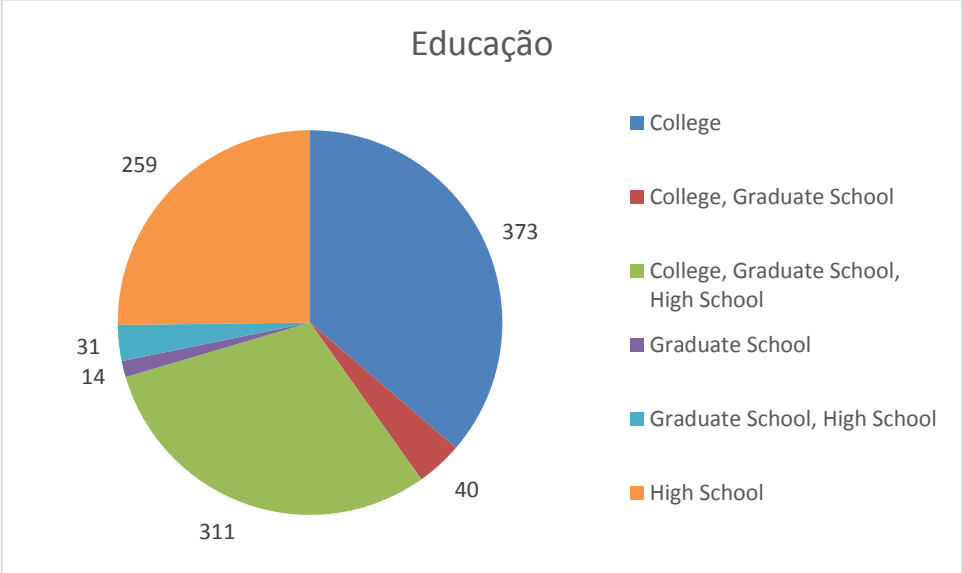
Legenda: Gráfico do atributo "fbDevices"

fbEducation

Este campo representa o histórico de estudos do utilizador.

Total de dados	3702
Dados em falta	758
Taxa de preenchimento	79.52%

Legenda: Estatísticas do atributo "fbEducation"



Legenda: Gráfico do atributo "fbEducation"

fbName

Este campo representa o nome completo do utilizador.

Total de dados	3702
Dados em falta	3702
Taxa de preenchimento	100%

Legenda: Estatísticas do atributo "fbName"

fbFirstName

Este campo representa o primeiro nome do utilizador.

Total de dados	3702
Dados em falta	3383
Taxa de preenchimento	8.62%

Legenda: Estatísticas do atributo "fbFirstName"

fbLastName

Este campo representa o apelido do utilizador.

Total de dados	3702
Dados em falta	3383
Taxa de preenchimento	8.62

Legenda: Estatísticas do atributo "fbLastName"

email

Este campo representa o endereço de correio eletrónico do utilizador.

Total de dados	3702
Dados em falta	3691
Taxa de preenchimento	0.3%

Legenda: Estatísticas do atributo "email"

fbFriends

Este campo representa os perfis dos amigos do utilizador.

Total de dados	3702
Dados em falta	3702
Taxa de preenchimento	0%

Legenda: Estatísticas do atributo "fbFriends"

fbNumberOfFriends

Este campo representa o número total de amigos do utilizador.

Total de dados	3702
Dados em falta	747
Taxa de preenchimento	79.82%
Média	758.23
Desvio Padrão	674.83
Mínimo	1
Percentil 25%	328.5
Percentil 50%	608
Percentil 75%	985
Máximo	5000

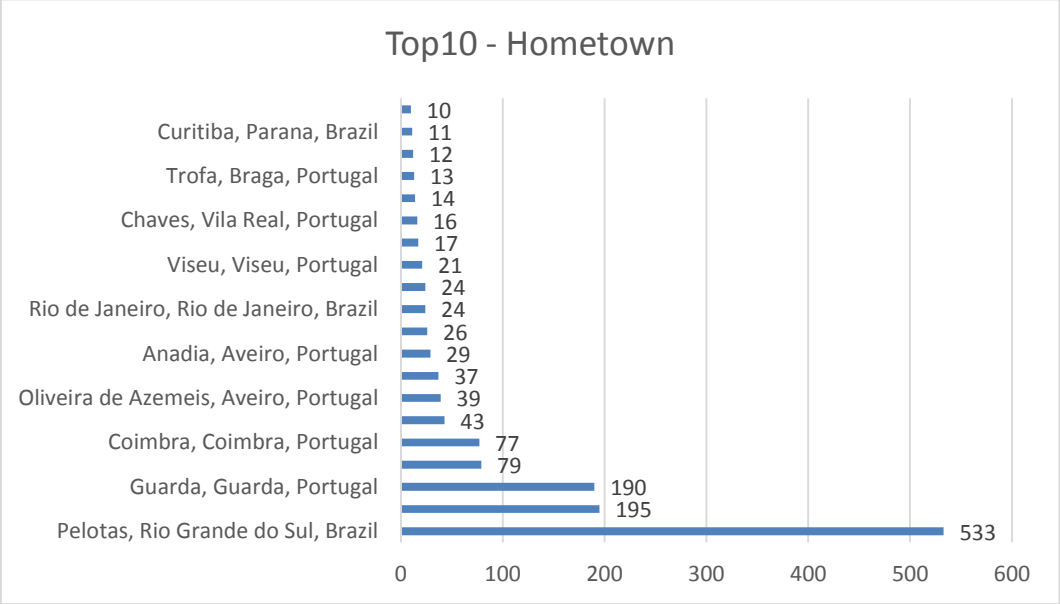
Legenda: Estatísticas do atributo "fbNumberOfFriends"

fbHometown

Este campo representa o local de nascimento (ou local onde passou a adolescência) do utilizador.

Total de dados	3702
Dados em falta	1606
Taxa de preenchimento	56.62%

Legenda: Estatísticas do atributo "fbHometown"



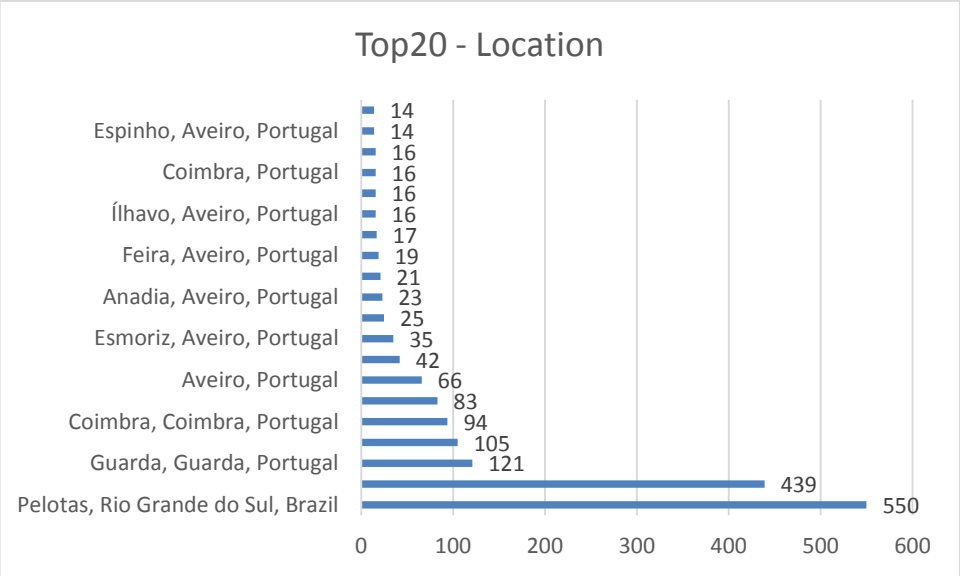
Legenda: Gráfico do atributo "fbHometown"

fbLocation

Este campo representa o local de habitação.

Total de dados	3702
Dados em falta	1540
Taxa de preenchimento	58.4%

Legenda: Estatísticas do atributo "fbLocation"



Legenda: Gráfico do atributo "fbLocation"

fbNumberOfSubscribedTo

Este campo representa o número de perfis que o utilizador subscreve.

Total de dados	3702
Dados em falta	0
Taxa de preenchimento	100%
Média	13.45
Desvio Padrão	75.48
Mínimo	0
Percentil 25%	0
Percentil 50%	0
Percentil 75%	0
Máximo	2042

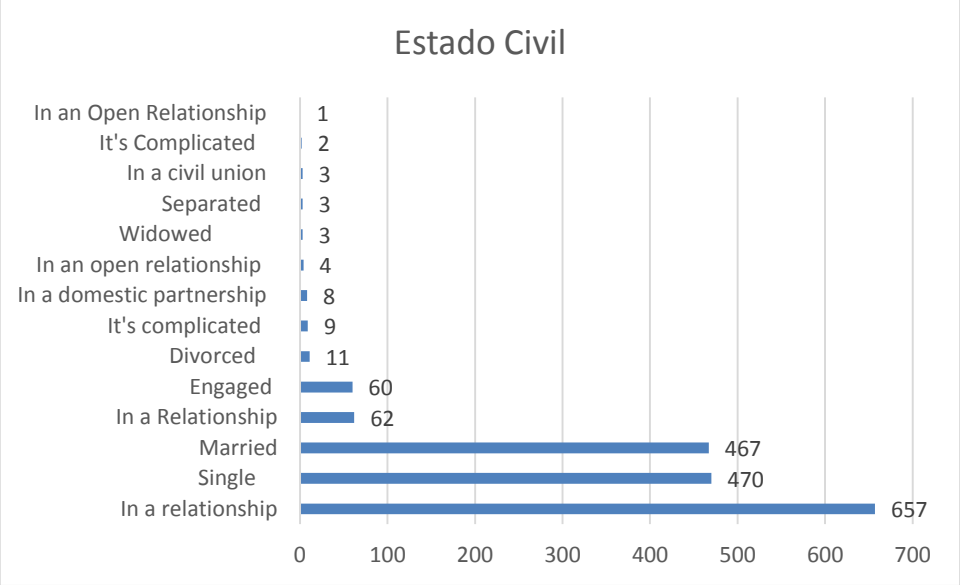
Legenda: Estatísticas do atributo "fbNumberOfSubscribedTo"

fbMaritalStatus

Este campo representa o estado civil do utilizador.

Total de dados	3702
Dados em falta	1942
Taxa de preenchimento	47.54%

Legenda: Estatísticas do atributo "fbMaritalStatus"



Legenda: Gráfico do atributo "fbMaritalStatus"

fbNumberOfSubscribedTo

Este campo representa o número de perfis que subscrevem o utilizador.

Total de dados	3702
Dados em falta	328
Taxa de preenchimento	91.14%
Média	14.57
Desvio Padrão	78.96
Mínimo	0
Percentil 25%	0
Percentil 50%	0
Percentil 75%	0
Máximo	258

Legenda: Estatísticas do atributo "fbNumberOfSubscribers"

fbPolitical

Este campo representa os ideais políticos do utilizador.

Total de dados	3702
Dados em falta	3695
Taxa de preenchimento	0.19%

Legenda: Estatísticas do atributo "fbPolitical"

fbReligion

Este campo representa a religião do utilizador.

Total de dados	3702
Dados em falta	3690
Taxa de preenchimento	0.19%

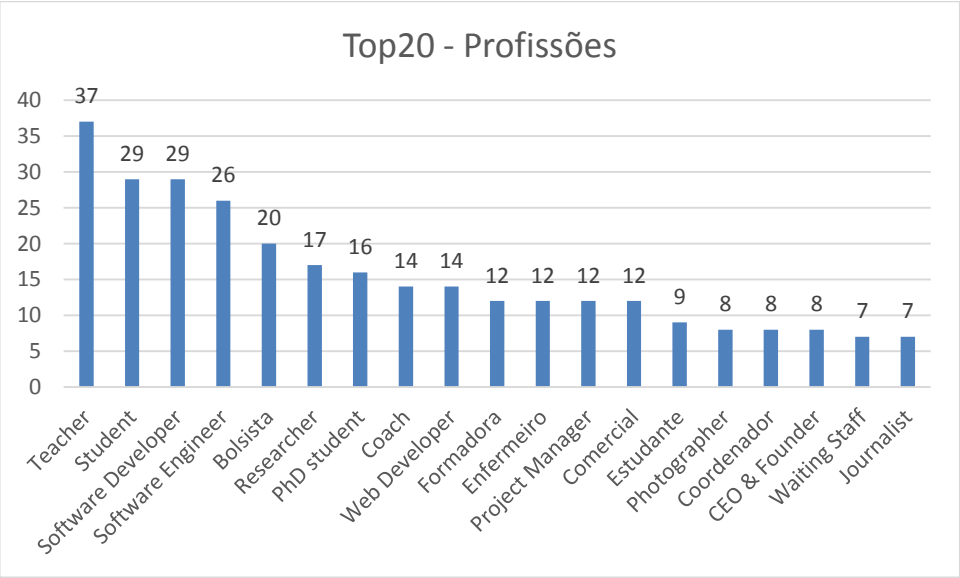
Legenda: Estatísticas do atributo "fbReligion"

fbWorkHistory

Este campo representa o histórico profissional do utilizador.

Total de dados	3702
Dados em falta	2436
Taxa de preenchimento	34.2%

Legenda: Estatísticas do atributo "fbWorkHistory"



Legenda: Gráfico do atributo "fbWorkHistory"

C) Lista completa de categorias dos *Likes* das páginas (*Facebook*)

Categorias (1º nível)			
actor/director	computers/technology	language	radio station
aerospace/defense	concentration or major	lawyer	real estate
airport	concert tour	legal/law	record label
album	concert venue	library	recreation/sports website
amateur sports team	consulting/business services	literary editor	reference website
animal	course	local business	regional website
animal breed	dancer	local/travel website	religion
app	designer	magazine	restaurant/cafe
app page	diseases	media/news/publishing	retail and consumer merchandise
appliances	drink	medical procedure	school sports team
art	drugs	mining/materials	science website
article	editor	monarch	shopping/retail
artist	education	movie	small business
arts/entertainment/ nightlife	education website	movie character	society/culture website
arts/humanities website	education/work status	movie general	software
athlete	electronics	movie genre	song
attractions/things to do	energy/utility	movie theater	spas/beauty/personal care
author	engineering/construction	movies/music	sport
automobiles and parts	entertainer	museum/art gallery	sports event
automotive	entertainment website	music	sports league
baby goods/kids goods	entrepreneur	music award	sports team
bags/luggage	episode	music chart	sports venue
bank/financial institution	event planning/event services	music video	sports/recreation/activities
bank/financial services	farming/agriculture	musical genre	state/province/region
bar	fictional character	musical instrument	studio
biotechnology	field of study	musician/band	teacher
board game	food	news personality	teens/kids website
book	food/beverages	news/media website	telecommunication
book genre	food/grocery	non-governmental organization (ngo)	tools/equipment
book series	furniture	non-profit organization	topic
book store	games/toys	office supplies	tours/sightseeing
building materials	geographical feature	other	transit stop
business person	government official	outdoor gear/sporting goods	transport/freight
business/economy website	government organization	patio/garden	transportation
camera/photo	government website	personal blog	travel/leisure
cars	health/beauty	personal website	tv
cause	health/medical/ pharmaceuticals	pet	tv channel
chef	health/medical/ pharmacy	pet services	tv genre

chemicals	health/wellness website	pet supplies	tv network
church/religious organization	high school status	phone/tablet	tv season
city	home decor	photographer	tv show
clothing	home improvement	playlist	tv/movie award
club	home/garden website	political ideology	university
coach	hospital/clinic	political organization	video
color	household supplies	political party	video game
comedian	industrials	politician	vitamins/supplements
commercial equipment	insurance company	producer	website
community	interest	product/service	wine/spirits
community organization	internet/software	profession	work position
community/government	jewelry/watches	professional services	work project
company	journalist	professional sports team	work status
competition	just for fun	profile	writer
computers	kitchen/cooking	public figure	
computers/internet website	landmark	public places	

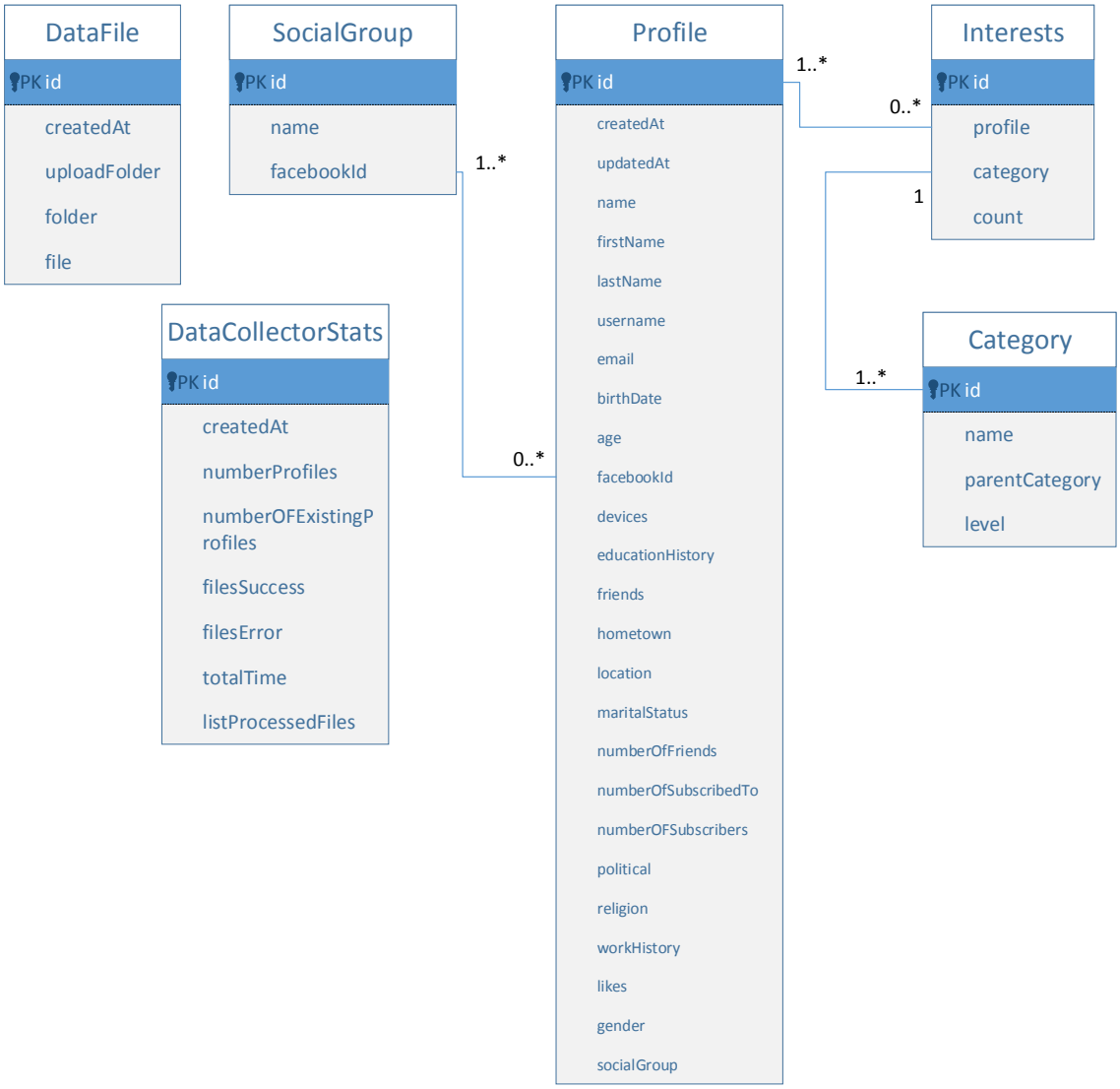
Legenda: Lista completa de categorias dos Likes (Facebook) – 1ºNível

D) Tabela de correlação entre Interesses (exemplo com 25 colunas, no total são 214)

	community	comedian	clothing	album	city	tainment w	local busines	other	electronics	cause	usician/bar	website	company	oduct/servi	rofit organi	onsumer m	bar	sports team	public figure	opping/net	ctor/direct	welry/watc	tv network	politician	artist
community	1	0.057469	-0.07209	0.02126	0.025206	0.089423	-0.00243	-0.01048	0.028182	0.027866	-0.25472	-0.07989	-0.08002	-0.05662	0.068201	-0.03917	-0.00228	-0.07862	0.00877	0.033341	0.002879	0.044628	-0.08219	-0.03625	-0.0042
comedian	0.057469	1	-0.07483	0.021687	-0.01574	0.108848	-0.05857	-0.01784	0.007424	-0.00982	-0.05299	-0.02903	-0.01037	-0.01399	-0.07574	0.004639	-0.04576	0.002803	0.116632	-0.0286	0.138579	-0.03691	0.058679	0.010228	-0.0034
clothing	-0.07209	-0.07483	1	0.00101	-0.06085	-0.06562	0.040685	-0.03292	-0.04495	-0.04888	-0.07685	0.155488	0.007758	0.03134	-0.10986	0.129579	-0.03062	-0.0502	-0.04617	0.158692	-0.00643	0.172311	-0.05276	-0.03987	0.01727
album	0.02126	0.021687	0.00101	1	0.017927	0.034984	-0.00546	-0.00716	-0.01781	-0.02814	0.061011	-0.02235	-0.06697	-0.04698	-0.05269	-0.02422	0.019469	-0.01202	0.00148	-0.01869	0.009571	-0.03306	0.012719	-0.01826	-0.01514
city	0.025206	-0.01574	-0.06085	0.017927	1	-0.02885	0.007922	-0.00866	0.00424	0.001966	-0.04084	-0.02242	-0.02828	-0.01588	-0.00324	-0.01652	0.028507	-0.01383	0.028213	-0.05612	-0.00565	-0.00722	0.02563	-0.00556	-0.01222
tainment w	0.089423	0.108848	-0.06562	0.034984	-0.02885	1	-0.09273	0.032029	0.031711	-0.01245	-0.06372	-0.00154	-0.0655	0.033355	-0.08208	0.020936	-0.01655	-0.01491	0.036438	-0.07258	0.050498	-0.00706	-0.01451	-0.06307	-0.03377
cal busines	-0.00243	-0.05857	0.040685	-0.00546	0.007922	-0.09273	1	-0.00627	-0.03329	0.006086	-0.19544	-0.09485	0.113494	0.030897	-0.01967	0.065722	0.024169	-0.07571	-0.06609	0.169922	-0.07623	0.081215	-0.08887	0.010418	0.002691
other	-0.01048	-0.01784	-0.03292	-0.00716	-0.00866	0.032029	-0.00627	1	-0.00999	0.190314	-0.02713	0.040988	-0.01864	0.005708	0.016287	0.009001	-0.03207	0.003174	0.009749	-0.01478	-0.01305	-0.0004	0.001367	0.001257	-0.03394
electronics	0.028182	0.007424	-0.04495	-0.01781	0.00424	0.031711	-0.03329	-0.00999	1	0.018006	-0.0466	-0.00214	0.036777	0.051211	0.03716	0.026466	-0.02487	-0.01552	-0.03201	-0.0411	-0.01511	-0.01589	0.046818	0.02478	-0.02882
cause	0.027866	-0.00982	-0.04888	-0.02814	0.001966	-0.01245	0.006086	0.190314	0.018006	1	-0.05585	-0.03131	0.03397	0.02245	0.141128	0.023674	-0.04381	-0.03919	-0.02381	-0.03809	-0.03269	-0.04426	-0.03476	0.026426	-0.03216
usician/bar	-0.25472	-0.05299	-0.07685	0.061011	-0.04084	-0.06372	-0.19544	-0.02713	-0.0466	-0.05585	1	-0.06227	-0.18882	-0.14082	-0.13093	-0.08915	-0.01122	-0.04941	-0.04513	-0.15245	-0.01697	-0.08858	0.065554	-0.05962	-0.00431
website	-0.07989	-0.02903	0.155488	-0.02235	-0.02242	-0.00154	-0.09485	0.040988	-0.00214	-0.03131	-0.06227	1	-0.0719	-0.01227	-0.04194	-0.01699	-0.07202	0.001704	-0.05931	-0.0635	-0.04199	-0.00724	-0.00942	-0.0515	-0.05669
company	-0.08002	-0.01037	0.007758	-0.06697	-0.02828	-0.0655	0.113494	-0.01864	0.036777	-0.03397	-0.18882	-0.0719	1	0.073623	-0.01576	0.092424	-0.0366	-0.00489	-0.03388	0.087063	-0.08005	0.022346	-0.03732	-0.01999	-0.02712
oduct/servi	-0.05662	-0.01399	0.03134	-0.04698	-0.01588	0.033355	0.030897	0.005708	0.051211	0.02245	-0.14082	-0.01227	0.073623	1	0.053981	0.027185	-0.07601	-0.06975	-0.02229	-0.00139	-0.01701	0.076825	-0.04426	-0.05473	-0.01535
rofit organi	0.068201	-0.07574	-0.10986	-0.05269	-0.00324	-0.08208	-0.01967	0.016287	0.03716	0.141128	-0.13093	-0.04194	-0.01576	0.053981	1	-0.05847	0.030316	-0.08507	-0.07261	-0.08401	-0.05559	0.021367	-0.04605	0.002701	-0.0307
onsumer m	-0.03917	0.004639	0.129579	-0.02422	-0.01652	0.020936	0.065722	0.009001	0.026466	0.023674	-0.08915	-0.01699	0.092424	0.027185	-0.05847	1	-0.02793	-0.03722	-0.01823	0.108819	-0.01192	0.087835	-0.02941	-0.00011	-0.02263
bar	-0.00228	-0.04576	-0.03062	0.019469	0.028507	-0.01655	0.024169	-0.03207	-0.02487	-0.04381	-0.01122	-0.07202	-0.0366	-0.07601	0.030316	-0.02793	1	-0.06249	-0.04025	-0.05077	-0.04023	-0.02045	-0.02941	-0.02045	0.026001
sports team	-0.07862	0.002803	-0.0502	-0.01202	-0.01383	-0.01491	-0.07571	0.003174	-0.01552	-0.03919	-0.04941	0.001704	-0.00489	-0.06975	-0.08507	-0.03722	-0.06249	1	-0.05077	-0.03212	-0.04271	-0.08145	0.031941	0.011741	-0.07671
public figure	0.00877	0.116632	-0.04617	0.00148	0.028213	0.036438	-0.06609	0.009749	-0.03201	-0.02381	-0.04513	-0.05931	0.03388	-0.02229	-0.07261	-0.01823	0.108819	-0.04025	-0.05077	1	-0.03342	-0.0153	0.021039	0.106587	0.042742
opping/net	0.033341	-0.0286	0.158692	-0.01869	-0.05612	-0.07258	0.169922	-0.01478	-0.0411	-0.03809	-0.15245	-0.0635	0.087063	-0.00139	-0.08401	0.108819	-0.01105	-0.03212	-0.03342	1	-0.04642	0.142044	-0.04789	0.009187	-0.01236
ctor/direct	0.002879	0.138579	-0.00643	0.009571	-0.00565	0.050498	-0.07623	-0.01305	-0.01511	-0.05697	-0.01697	-0.04199	-0.08005	-0.01701	-0.05559	-0.01192	-0.0075	-0.04271	0.239818	-0.04642	1	0.024378	0.039096	-0.04032	0.033836
welry/watc	0.044628	-0.03691	0.172311	0.03306	-0.00722	-0.00706	0.081215	-0.0004	-0.01589	0.031625	-0.08858	-0.00724	0.022346	0.076825	0.021367	0.087835	-0.04023	-0.08145	-0.0153	0.142044	0.024378	1	-0.03896	-0.04799	-0.00128
tv network	-0.08219	0.058679	-0.05276	0.012719	0.02563	-0.01451	-0.08887	0.001367	0.046818	-0.03476	0.065554	-0.00942	-0.03732	-0.04426	-0.04605	0.021367	-0.02045	0.031941	0.021039	-0.04789	0.039096	-0.03896	1	-0.01398	-0.0265
politician	-0.03625	0.010228	-0.03987	-0.01826	-0.00556	-0.06307	0.010418	0.001257	0.02478	0.026426	-0.05962	-0.0515	-0.01999	-0.05473	0.002701	-0.00011	-0.03216	0.011741	0.106587	0.009187	-0.04032	-0.04799	-0.01398	1	-0.02661
artist	-0.0042	-0.0034	0.01727	-0.01514	-0.01222	-0.03377	0.002691	-0.03394	-0.02882	-0.03216	-0.00431	-0.05669	-0.02712	-0.01535	-0.0307	-0.02263	0.026001	-0.07671	0.042742	-0.01236	0.033836	-0.00128	-0.0265	-0.02661	1

Legenda: Tabela de correlação de 25 Interesses

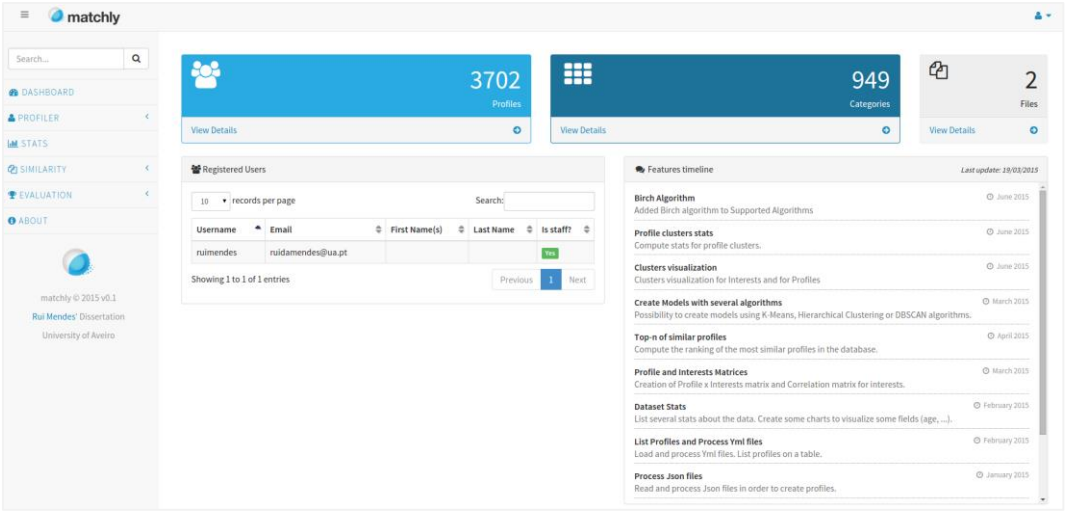
E) Diagrama de Classes detalhado



Legenda: Diagrama de Classes detalhado

F) Outras funcionalidades da aplicação *matchly*

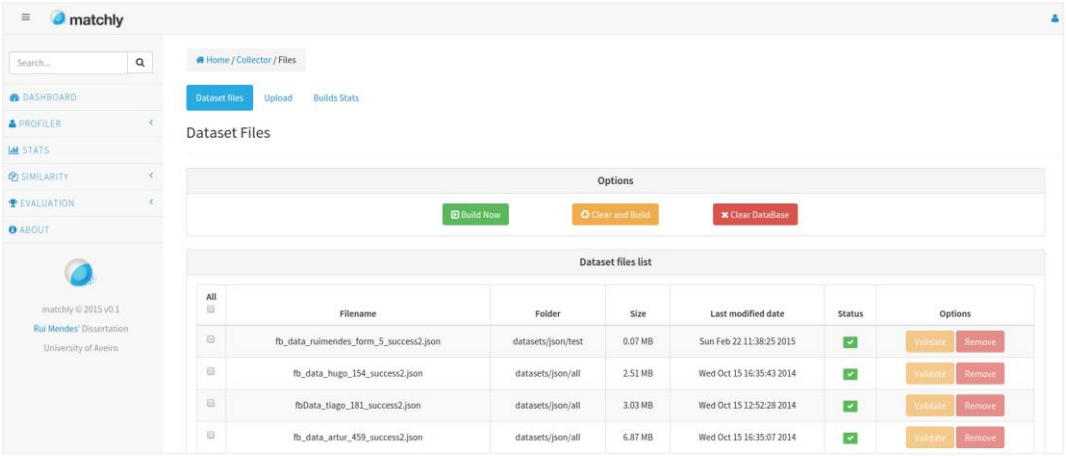
Dashboard – Página inicial do *back-office*. Contém diversas informações tais como o número de perfis presentes na base de dados, número de categorias de interesses processadas, o número atual de ficheiros com dados em "bruto", entre outras. Uma visão geral desta página pode ser vista na imagem abaixo.



Legenda: Página inicial do back-office (Dashboard)

Profiler (Fontes de dados) – As principais opções consistem em manipular os ficheiros com dados provenientes de redes sociais: carregar, processar (criar perfis), eliminar ou listar.

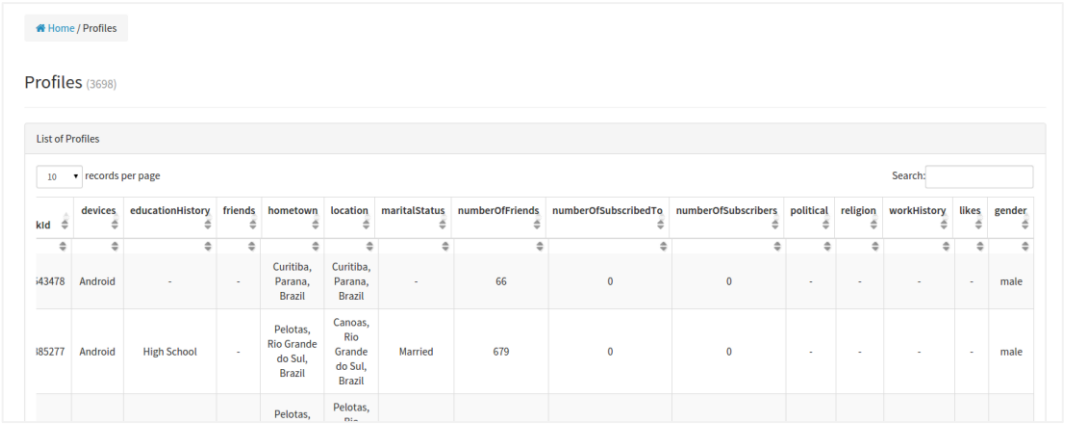
Na imagem abaixo, do lado esquerdo estão representadas as opções do *menu* principal que dá acesso ao *Data Collector* e *List*. A primeira pode ser vista na parte direita da figura e é onde é possível carregar ficheiros, processá-los ou ainda visualizar estatísticas sobre os processamentos anteriores.



Legenda: Menu Profiler e Lista de ficheiros carregados

A opção de listar os perfis atuais da base de dados pode ser vista na imagem seguinte.

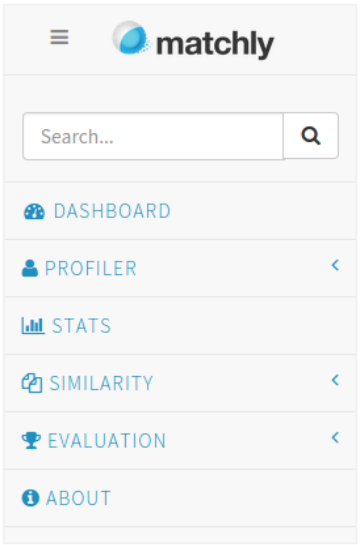
Uma vez que esta lista pode ter um tamanho consideravelmente grande, a tabela permite paginar os resultados de modo a facilitar a navegação e tempo de carregamento dos dados para o ecrã.



Legenda: Listar perfis (Profiler > List)

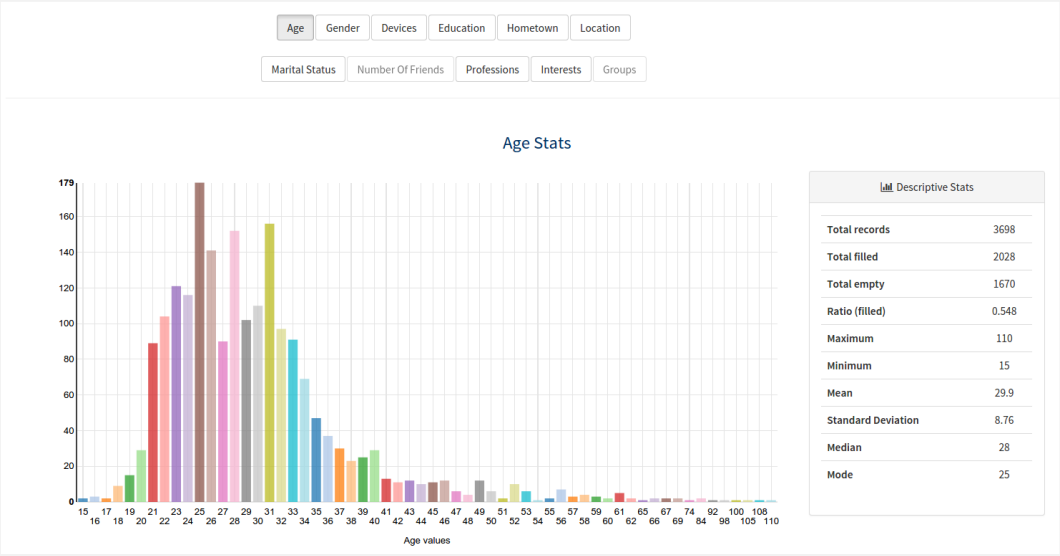
Stats – É possível visualizar diversos gráficos e estatísticas dos dados dos perfis presentes na base de dados.

Para aceder a opção basta carregar em “Stats” no menu principal.



Legenda: Opção Stats (Menu principal)

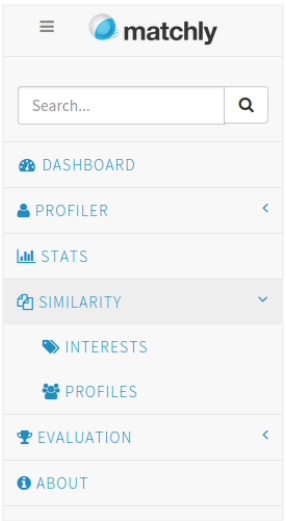
Uma vez dentro da opção “Stats”, é possível navegar por diversos campos com vista a visualizar gráficos dos dados assim como alguns dados estatísticos (imagem abaixo).



Legenda: Gráfico e estatísticas do campo Age

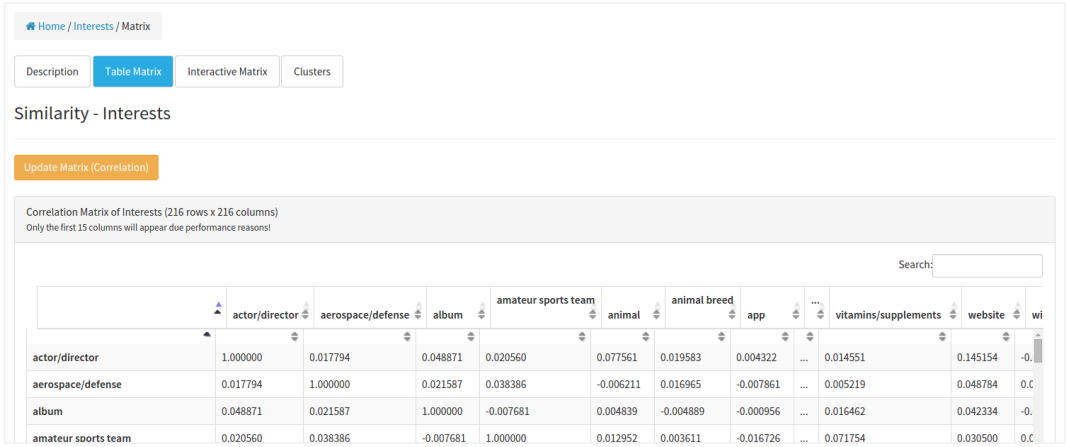
Similarity – Área onde é possível consultar as matrizes de correlação de interesses, perfis-interesses, criar *clusters* de interesses, criar modelos ou ainda criar *clusters* de perfis.

Como se pode observar na imagem abaixo, basta selecionar a opção *Similarity* > *Interests* ou *Similarity* > *Profiles* para aceder a uma das opções disponíveis.

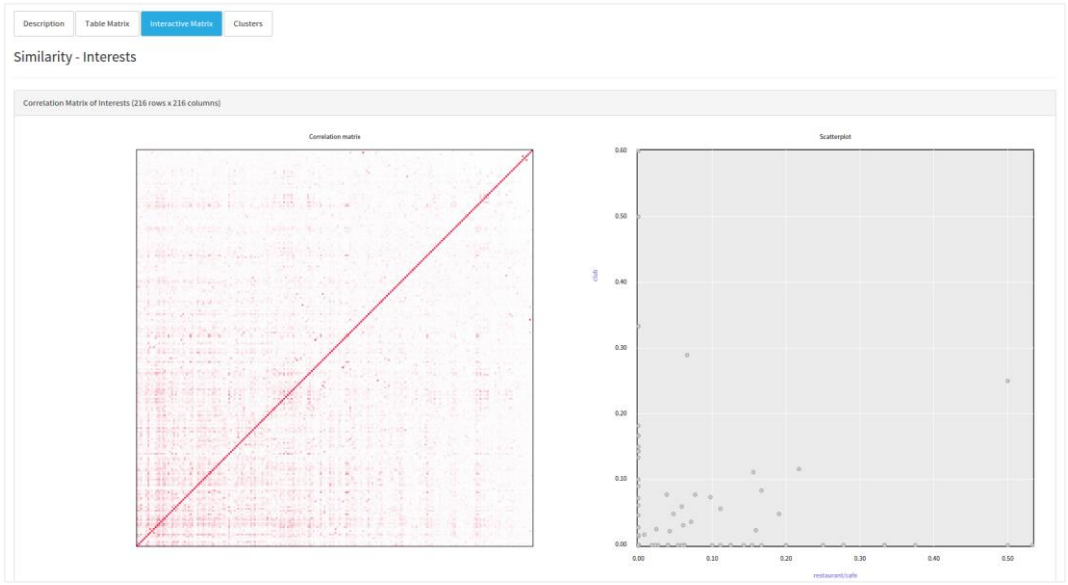


Legenda: Opção Similarity (Menu principal)

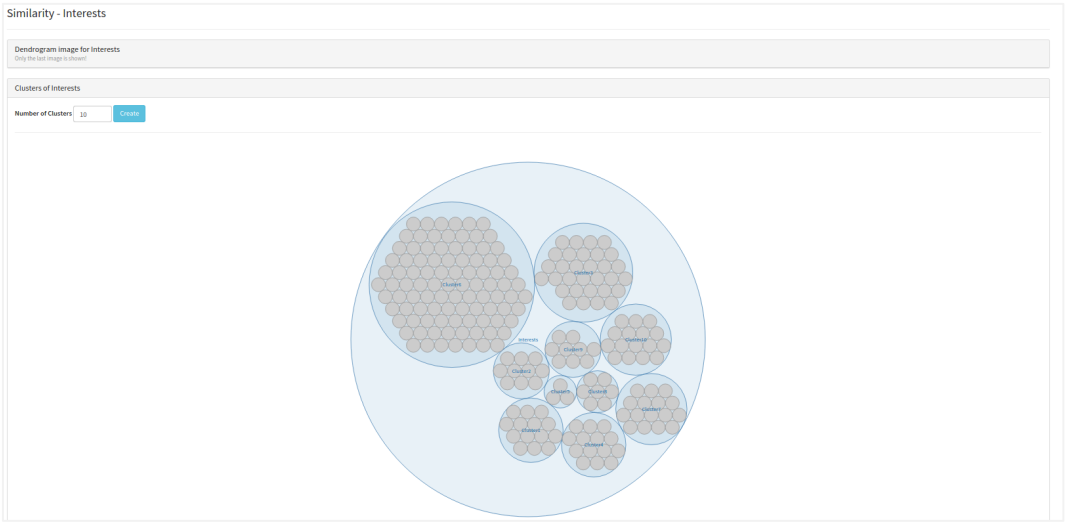
A opção *Similarity* > *Interests* disponibiliza diversas opções, das quais se destacam: visualizar a matriz de correlação entre interesses, explorar o conjunto de dados da matriz anterior numa matriz interativa, visualizar um dendrograma ou criar *clusters* de interesses. Estas opções podem ser vistas nas imagens seguintes.



Legenda: Matriz de correlação entre interesses

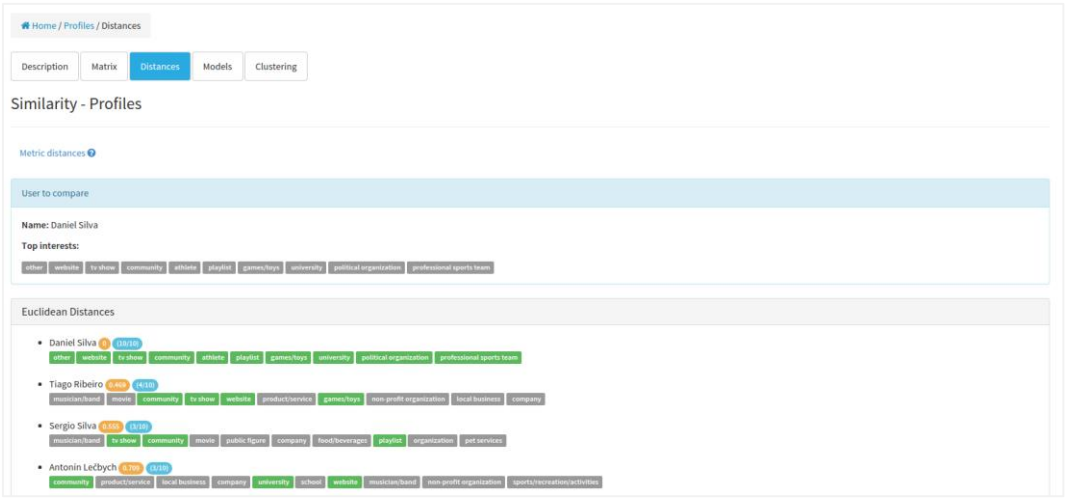


Legenda:Matriz interativa de correlação entre interesses

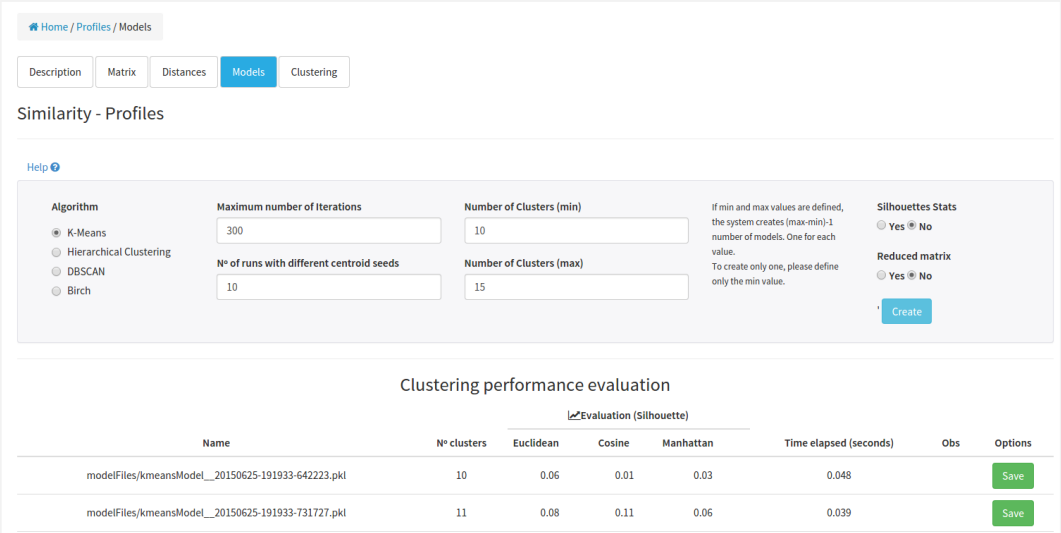


Legenda: Clusters de interesses

A opção *Similarity* > *Profiles* disponibiliza opções, tais como: listar matriz de perfis-similares, criar uma lista de perfis similares a um dado perfil, criar modelos utilizando algoritmos de *clustering*, listar modelos ou criar *clusters* de perfis.

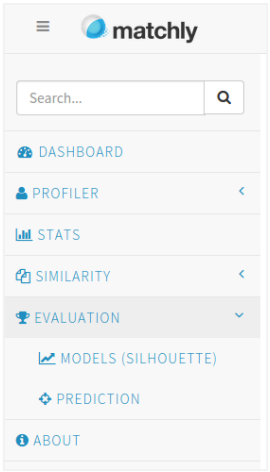


Legenda: Ranking de perfis similares a um dado perfil



Legenda: Criar modelos utilizando algoritmos de clustering

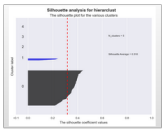
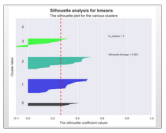
Finalmente, existe uma zona de resultados (*Evaluation*) de modo a avaliar os modelos criados assim como verificar a performance do sistema de predição de *ratings* baseado num sistema de recomendação (assuntos especificados na secção 5.).



Legenda: Opção Evaluation (Menu Principal)

Home / Evaluation / Models

Help

Algorithm	CreatedAt	Name	Params	Evaluation	Time to process *(seconds)*	Silhouette analysis *(image)*
hierarclust	June 25, 2015, 7:27 p.m.	hierarclustModel__20150625-192723-546055	**reduced_matrix_nclusters:** 10 **algorithm:** hierarclust **reduced_matrix:** False **create_silhouettes:** True **affinity:** euclidean **nClusters:** 5 **linkage:** ward	**euclidean:** 0.318 **cosine:** 0.251 **manhattan:** 0.116	0.004	
kmeans	June 25, 2015, 7:27 p.m.	kmeansModel__20150625-192712-631778	**reduced_matrix_nclusters:** 10 **nInit:** 10 **algorithm:** kmeans **reduced_matrix:** True **create_silhouettes:** True **nClusters:** 5 **maxIter:** 300	**euclidean:** 0.265 **cosine:** 0.405 **manhattan:** 0.227	0.04	
birch	June 25, 2015, 1:24 a.m.	birchModel__20150625-012407-575119	**reduced_matrix_nclusters:** 15 **compute_labels:** 1 **algorithm:** birch **reduced_matrix:** True **create_silhouettes:** False **branching_factor:** 50 **nClusters:** 25 **threshold:** 0.2	**euclidean:** 0.136 **cosine:** -0.027 **manhattan:** 0.106	0.167	

Legenda: Avaliação dos modelos criados