Universidade do Minho
Departamento de Informática

Mestrado Integrado em Engenharia Informática

Sistemas Baseados em Similaridade

Conceção e Implementação de um Sistema de Recomendacão

Grupo 6
A80524 André Viveiros
A80426 Guilherme Andrade
A81765 Joana Matos
A80624 Sofia Teixeira

Braga
Janeiro, 2020

Conteúdo

1 Introdução

Este relatório apresenta e documenta o terceiro trabalho prático desenvolvido no âmbito da Unidade Curricular Sistemas Baseados em Similaridade, do curso Mestrado em Engenharia Informática da Universidade do Minho, no ano letivo de 2019/2020.

Este trabalho tem como objetivo explorar e aperfeiçoar conhecimentos em *Machine Learning*, mais concretamente, em sistemas de recomendação, através da plataforma *Knime* e da utilização da linguagem de programação *Python* juntamente com todas as suas bibliotecas. Em suma, consiste na implementação de um sistema de recomendação no contexto de **Filmes**, que por sua vez deve abordar vários tópicos estudados anteriormente, respetivamente sistemas híbridos, sistemas baseados em conteúdo e sistemas colaborativos, etc. Para além disso, deve ser apresentada uma *interface* amigável, onde nesta possa ser perceptível o vantagem dos sistemas de recomendação e a avaliação dos algoritmos implementados, neste caso, na precisão da recomendação.

1.1 Contextualização e caso de estudo

Numa fase prévia, foi pedido que se explorasse os sistemas de recomendação, quais os algoritmos utilizados, tais como as suas utilizações e vantagens. Tendo isto em conta, foi agora requerido a realização de uma aplicação prática dos conhecimentos adquiridos. O caso de estudo considerado centra-se no desenvolvimento de um sistema de recomendação de filmes utilizando *KNIME* e *Python* e de uma merda interface que demonstre o seu potencial.

1.2 Estrutura do relatório

- No capítulo Metodologia explica-se sucintamente os passos realizados para o desenvolvimento deste sistema de recomendação, tal como todas as plataformas utilizas;
- No capítulo *Dataset* explica-se os diferentes tipos de base de dados utilizados e os seus respetivos atributos e como se procedeu no tratamento de dados;
- No capítulo Técnicas explica-se os diferentes tipos de métodos utilizados para fazer a recomendação, dando mais ênfase nos algoritmos e ideias para a implementação dos mesmos.
- Na Interface explica-se como esta foi desenvolvida e mostram-se algumas imagens dos resultados finais;
- O relatório é concluído na Conclusão com observações relevantes.

2 Metodologia

Este projeto teve início com a procura de um dataset acessível relativo a este tema. Com efeito, esta fase foi concluída através da plataforma Kaggle onde foram encontrados vários possíveis *datasets*. O *dataset* escolhido é denominado como *The Movie Dataset*.

De seguida, realizou-se o tratamento e manipulação dos dados em KNIME de maneira a que estes fossem legíveis, coerentes e fosse possível obter o máximo de informação possível dos mesmos.

Tendo os dados prontos a serem utilizados, procedeu-se à concepção das técnicas de recomendação. Foram desenvolvidas várias técnicas relativas a filtragem colaborativa e sistemas baseados em conteúdo, que foram utilizadas na elaboração de uma técnica híbrida em conjunto com alguns parâmetros adicionais. Todo o processo relativo a tratamento de dados foi por sua vez, desenvolvido em KNIME, mas em relação às técnicas de recomendação, algumas foram desenvolvidas em Python, onde posteriormente eram todas combinadas num único algoritmo, também em Python.

Por fim, foi concebida uma pequena interface utilizando Django, de modo a ser possível observar o resultado de todo este processo.

3 *Dataset*

O *dataset* escolhido proveniente da plataforma Kaggle, denominado *"The Movies Dataset"* trata-se de uma amostra de, aproximadamente, quarenta e cinco mil filmes. É apresentado um conjunto de ficheiros representativos do *dataset*, nos quais foi necessária alguma manipulação, quer em termos da estrutura dos dados, quer em termos da informação contida propriamente dita.

Numa fase inicial, foram selecionados os ficheiros considerados mais representativos do *dataset*, nomeadamente: "movies_metada.csv", "ratings_small.csv", "credits.csv"e "keywords.csv".

3.1 Descrição do *Dataset*

3.1.1 *Movies*

O ficheiro "movies_metada.csv"contém características diretamente relacionadas com o filme, tais como o ano de estreia, título, língua falada, género e faturação de bilheteira. Este vai ter um papel central no trabalho e será alvo de alguma manipulação de valores para um melhor entendimento e interpretação do autor. Faz-se uma breve descrição de todos os atributos do *dataset*.

- **adult** - identifica se o filme é caracterizado ou não apenas para adultos;
- **belongs_to_collection** - afirma ou não se o filme pertence a uma trilogia/coleção;
- **budget** - despesas associadas ao filme;
- **genres** - géneros do filme;
- **homepage** - página onde o filme se encontra;
- **id** - identificação do filme;
- **imdb**$_id$ $-identificação do filme na página do imdb$; **original_language** $-linguagem nativa do filme$;
- **original_title** - título original do filme;
- **overview** - breve descrição do filme;
- **popularity** - popularidade do filme;
- **production_companies** - companhias que o produziram;
- **production_contries** - países de produção do filme;
- **release_date** - identifica a data de lançamento;
- **revenue** - receita associada ao filme;
- **status** - estado do filme(se foi lançado ou não);
- **runtime** - apresenta a duração do filme;
- **spoken_languages** - linguagens faladas no filme;
- **title** - título do filme que pode não ser necessariamente o seu título originial;

- **vote_average** - média de classificações feitas ao filme;
- **vote_count** - quantidade de votos feitos ao filme.

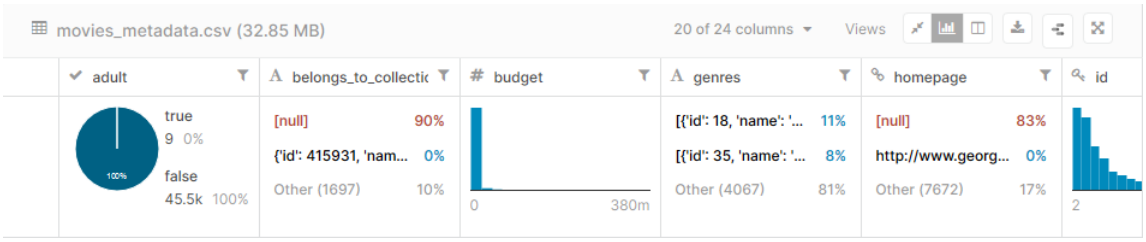


Figura 1: Ficheiro "movies_metada.csv".

3.1.2 *Ratings*

O ficheiro "ratings_small.csv"é composto por linhas que fazem a correspondência entre um utilizador e um filme através de uma classificação. Este ficheiro corresponderá à base de dados de utilizadores neste sistema e será através deste que se calculará, por exemplo, recomendações por filtragem colaborativa baseadas em utilizadores.

A partir da figura 2, é possível observar os atributos:

- **userId** - identifica a identificação de um utilizador;
- **movieId** - identifica a identificação do filme;
- **rating** - representa a classificação que um utilizador deu a um filme de zero a cinco;
- **timestamp** - representam o tempo desde uma determinada hora até a avaliação dos utilizadores.

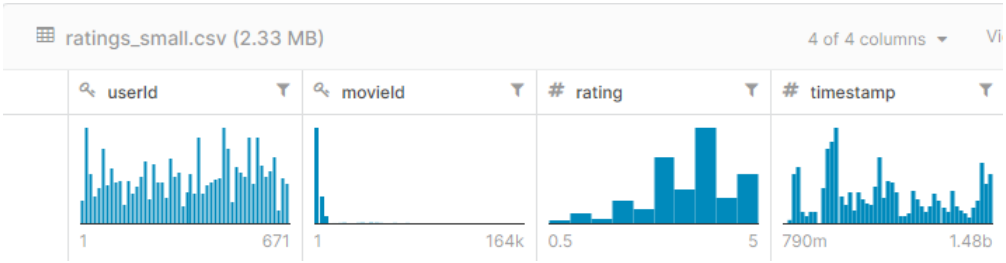


Figura 2: Ficheiro "ratings_small.csv".

3.1.3 *Credits* e *Keywords*

Os ficheiros "credits.csv"e "keywords.csv"são complementares ao primeiro indicado nesta lista ("movies_metada.csv"). O ficheiro relacionado com créditos dará informações relacionadas com o *staff* e *casting* de um dado filme. O ficheiro relacionado com palavras-chave dará informações relacionadas com o conteúdo de um dado filme, representando o enredo de um filme através de determinadas palavras ilustradas e explicativas.

A partir da imagem 3, referente ao primeiro ficheiro mencionado, observam-se os atributos:

- **cast** - atrizes e atores pertencentes ao filme;
- **crew** - equipa responsável pela realização do filme;
- **id** - filme associado a cada um dos atributos em cima referidos.

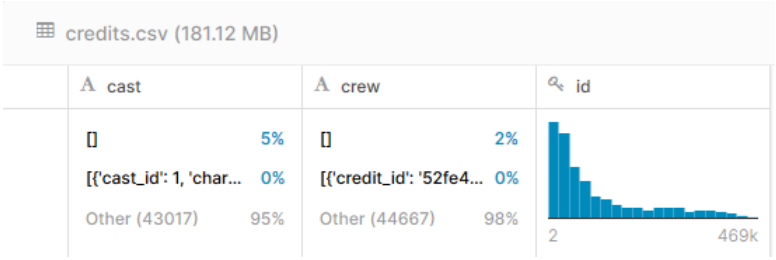


Figura 3: Ficheiro "credits.csv".

Finalmente, a partir da imagem 4, observam-se os atributos do último ficheiro:

- **id** - identifica o filme em causa;
- **keywords** - refere possíveis palavras-chave que identifiquem o decorrer do filme.

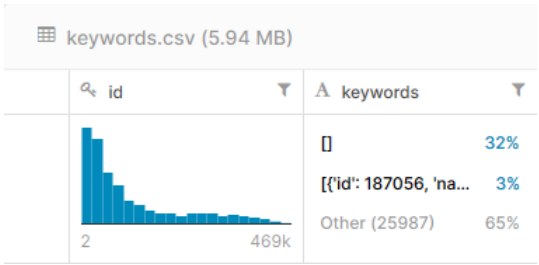


Figura 4: Ficheiro "keywords.csv".

3.2 Tratamento de dados

Para uma representação da informação contida em alguns dos ficheiros referidos acima, nomeadamente "movies_metada.csv", "credits.csv"e "keywords.csv", foi necessário um pré-processamento, dado que alguns destes apresentavam um formato em JSON para cada linha. Foram definidos alguns *scripts* em Python para a leitura, extração e exposição dos dados em formato *csv*. Não foi utilizada toda a informação contida nas *strings* JSON, pois foi considerado que alguns dados não seriam relevantes para o trabalho proposto, como por exemplo, no ficheiro "moviesdataset.csv", o campo da *homepage*. Após isto, procedeu-se à manipulação e limpeza do *dataset*. Cada um dos ficheiros foi tratado individualmente pois, mais tarde, representariam critérios diferentes no que toca ao sistema de recomendação em si.

3.2.1 Filmes

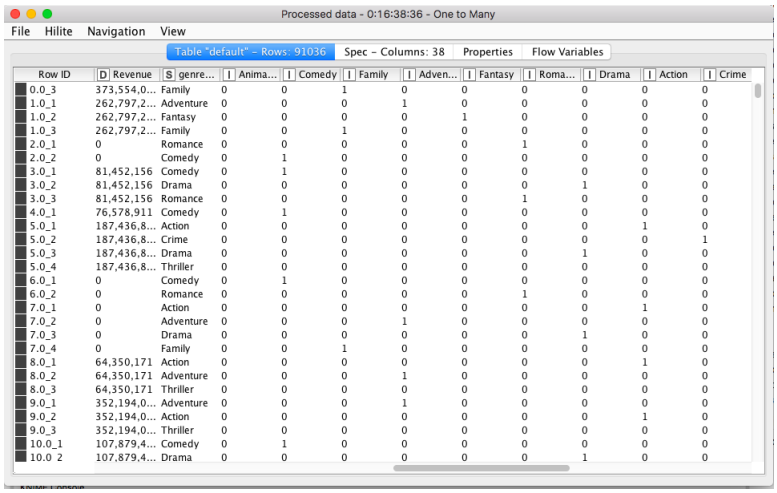
O ficheiro dos filmes era composto inicialmente por imensas colunas distintas, então foi necessário um processo de seleção de acordo com a sua importância para o problema. No final desta fase, foi criado um conjunto composto por dez colunas que se achou mais relevantes:

- **id** - identificador único de cada filme, necessário para referenciar cada instância na tabela.
- **title** - título de um filme traduzido para inglês.
- **original title** - título original de um filme, ou seja, na sua língua nativa.
- **year** - ano da estreia do filme, este foi retirado após a formatação de uma coluna denominada por release_date que continha uma data com dia, mês e ano; foi decidido que o dia e o mês não apresentavam importância para o problema em questão e a coluna foi decomposta.
- **vote_average** - classificação média de um filme.
- **popularity** - índice de popularidade de um filme
- **genres** - lista de géneros a que um filme pertence.
- **runtime** - duração de um filme em minutos.
- **vote_count** - número de votos registados para um filme.
- **revenue** - receitas de um filme.

Das colunas que foram excluídas, existem algumas que devem ser mencionadas, pois noutro contexto ou noutra dimensão de trabalho, poderiam ser consideradas importantes. A coluna *budget* é uma coluna referente ao orçamento para cada filme e poderia ser um indicador útil para encontrar relações de lucro através da combinação desta coluna com *revenue*. As colunas *production_companies* e *production_countries* referem-se a detalhes de produção do filme e poderia ser mais um critério considerado na altura de fazer recomendações a utilizadores. Ligado a este critério, estaria também a utilização da coluna *spoken_languages* que se refere à língua falada no filme, coluna também considerada irrelevante nesta questão. Decidiu-se retirar estas colunas, pois deu-se mais ênfase a técnicas baseadas em filtragem colaborativa. Devido a uma grande dimensão de dados, estas colunas são as mais relevantes para os objetivos que queremos atingir, que serão mencionados posteriormente.

No que toca à formatação, reparou-se que alguns atributos numéricos eram apresentados como *strings*, nomeadamente, os valores da coluna *id*, *revenue* e *popularity*, e procedeu-se à transformação destes. Os valores de *id* e *revenue* em interos e os valores de *popularity* em *dobule*.

Na exploração de dados, foi notado que algumas das entradas da tabela apresentavam *missing values* nas colunas *release_date*, *id* e *title* e foi decidido remover essas ditas entradas devido a uma grande dimensão de dados. É de notar que filmes com *runtime* a zero também foram removidos. Nas colunas *revenue* e *budget*, observou-se que havia valores a zero também. Interpretou-se os valores a zero de *revenue* como sendo credíveis. Quanto aos valores de *budget*, valores a zero não seriam credíveis neste contexto, sendo esta uma das razões pela qual esta coluna acabou excluída. Na variável *genres*, continha-se uma lista de géneros associados ao filme, isto também teve que ser tratado para que se pode-se utilizar o atributo na própria recomendação. Decidiu-se utilizar o nodo *OneToMany* no *Knime*, que separou os géneros em categorias binárias como podemos observar na imagem 5. Todos os dados referentes a *release_date* estão em *string*, por isso também ocorreu uma transformação de *string to date*, através do *knime* onde posteriormente se retirou o ano do respetivo filme.



Row ID	Revenue	genre...	Anima...	Comedy	Family	Adven...	Fantasy	Roma...	Drama	Action	Crime
0.0_3	373,554,0...	Family	0	0	1	0	0	0	0	0	0
1.0_1	262,797,2...	Adventure	0	0	0	1	0	0	0	0	0
1.0_2	262,797,2...	Fantasy	0	0	0	0	1	0	0	0	0
1.0_3	262,797,2...	Family	0	0	1	0	0	0	0	0	0
2.0_1	0	Romance	0	0	0	0	0	1	0	0	0
2.0_2	0	Comedy	0	1	0	0	0	0	0	0	0
3.0_1	81,452,156	Comedy	0	1	0	0	0	0	0	0	0
3.0_2	81,452,156	Drama	0	0	0	0	0	0	1	0	0
3.0_3	81,452,156	Romance	0	0	0	0	0	1	0	0	0
4.0_1	76,578,911	Comedy	0	1	0	0	0	0	0	0	0
5.0_1	187,436,8...	Action	0	0	0	0	0	0	0	1	0
5.0_2	187,436,8...	Crime	0	0	0	0	0	0	0	0	1
5.0_3	187,436,8...	Drama	0	0	0	0	0	0	1	0	0
5.0_4	187,436,8...	Thriller	0	0	0	0	0	0	0	0	0
6.0_1	0	Comedy	0	1	0	0	0	0	0	0	0
6.0_2	0	Romance	0	0	0	0	0	1	0	0	0
7.0_1	0	Action	0	0	0	0	0	0	0	1	0
7.0_2	0	Adventure	0	0	0	1	0	0	0	0	0
7.0_3	0	Drama	0	0	0	0	0	0	1	0	0
7.0_4	0	Family	0	0	1	0	0	0	0	0	0
8.0_1	64,350,171	Action	0	0	0	0	0	0	0	1	0
8.0_2	64,350,171	Adventure	0	0	0	1	0	0	0	0	0
8.0_3	64,350,171	Thriller	0	0	0	0	0	0	0	0	0
9.0_1	352,194,0...	Adventure	0	0	0	1	0	0	0	0	0
9.0_2	352,194,0...	Action	0	0	0	0	0	0	0	1	0
9.0_3	352,194,0...	Thriller	0	0	0	0	0	0	0	0	0
10.0_1	107,879,4...	Comedy	0	1	0	0	0	0	0	0	0
10.0_2	107,879,4...	Drama	0	0	0	0	0	0	1	0	0

Figura 5: *One To Many on Genres*

Todo o *workflow* utilizado para o tratamento deste ficheiro é representado na imagem 6

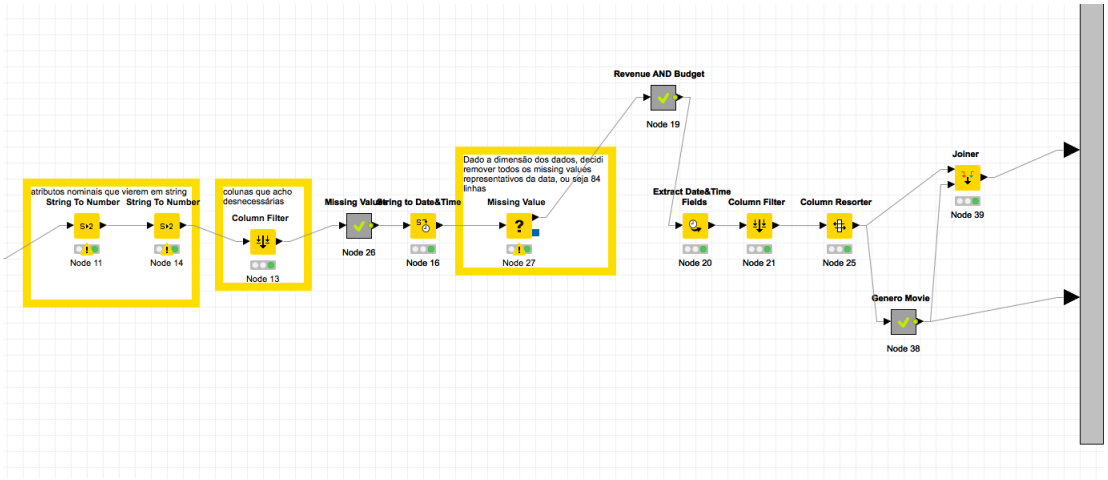


Figura 6: *Workflow*

3.2.2 *Ratings*

O ficheiro relativo a classificações dadas por certos utilizadores não continha *strings* em *JSON* e não precisou de qualquer formatação ou manipulação de dados. Foi possível aplicar de seguida os procedimentos em relação à recomendação em si.

- **id** - identificador único de cada utilizador, necessário para referenciar cada instância na tabela.
- **movie_id** - identificador único com a função de referenciar filmes.
- **rating** - classificação definida por um determinado utilizador a um determinado filme.

3.2.3 *Keywords* e *Credits*

A razão pela qual ambos os ficheiros são mencionados nesta mesma secção deve-se a um tratamento muito semelhante dos dois ficheiros. O ficheiro relativo a palavras chaves descritoras de um filme apresenta duas colunas apenas, ambas essenciais para o critério de comparação em questão:

- **id** - identificador único de cada filme, necessário para referenciar cada instância na tabela.
- **keywords** - lista totalmente formatada em *string* do conjunto de palavras descritoras.

O ficheiro relativo a créditos apresentava três colunas mas foi reduzida, por decisão de grupo, para duas colunas:

- **id** - identificador único de cada filme, necessário para referenciar cada instância na tabela.
- **cast** - lista totalmente formatada em *string* dos atores participantes num filme.

A coluna excluída dizia respeita ao *staff* do filme, tal como diretores, encenadores e técnicos. A decisão da exclusão desta coluna deveu-se à grande variedade de combinações de nomes na lista

que levaria a uma separação muito heterogénea de todas as instâncias. Uma abordagem poderia ter sido a filtragem dos nomes mais relevantes, como de realizadores e produtores, que mantêm um papel muito ativo no filme, mas os autores decidiram que não teria um papel crucial nesta questão.

As listas apresentadas em formato *string* dos dois ficheiros apresentavam a mesma estrutura: primeiro e último caracteres coincidiam com os símbolos "[" e "]", respetivamente; todos os elementos da lista apresentavam aspas à sua volta; havia uma vírgula de separação entre cada dois elementos, formato de *array*. Procedeu-se à remoção de casos que apresentavam listas vazias. De seguida, a remoção de quaisquer símbolos de pontuação estranhos, ou seja, todos os símbolos de pontuação à excepção das vírgulas delimitadoras. Estas vírgulas iam servir para, com cada entrada da coluna, criar um objeto do tipo coleção, objeto de KNIME. que mais tarde iria ser necessária para aplicar certos algoritmos para atingir o objetivo proposto, algoritmos estes discutidos mais à frente. Os *inputs* recebidos nos atributos *cast* e *keywords* tinham o formato apresentado na figura 7, e após todo este processo que acabou de ser mencionado e pode ser observado na imagem 8 obtemos o resultado referido na imagem 9

Row ID	I id	S keywords
0.0	862	['jealousy', 'toy', 'boy', 'friendship', 'friends', 'rivalry', 'boy next door', 'new toy', 'toy comes to life']
1.0	8844	['board game', 'disappearance', "based on children's book", 'new home', 'recluse', 'giant insect']
2.0	15602	['fishing', 'best friend', 'duringcreditsstinger', 'old men']
3.0	31357	['based on novel', 'interracial relationship', 'single mother', 'divorce', 'chick flick']
4.0	11862	['baby', 'midlife crisis', 'confidence', 'aging', 'daughter', 'mother daughter relationship', 'pregnan...
5.0	949	['robbery', 'detective', 'bank', 'obsession', 'chase', 'shooting', 'thief', 'honor', 'murder', 'suspense...
6.0	11860	['paris', 'brother brother relationship', 'chauffeur', 'long island', 'fusion', 'millionaire']
7.0	45325	[]
8.0	9091	['terrorist', 'hostage', 'explosive', 'vice president']
9.0	710	['cuba', 'falsely accused', 'secret identity', 'computer virus', 'secret base', 'secret intelligence ser...
10.0	9087	['white house', 'usa president', 'new love', 'widower', 'wildlife conservation']
11.0	12110	['dracula', 'spoof']
12.0	21032	['wolf', 'dog-sledding race', 'alaska', 'dog', 'goose', 'bear attack', 'dog sled', 'frozen lake']
13.0	10858	['usa president', 'presidential election', 'watergate scandal', 'biography', 'government', 'historical...
14.0	1408	['exotic island', 'treasure', 'map', 'ship', 'scalp', 'pirate']
15.0	524	['poker', 'drug abuse', '1970s', 'overdose', 'illegal prostitution']
16.0	4584	['bowling', 'based on novel', 'servant', 'country life', 'jane austen', 'inheritance', 'military officer', '...
17.0	5	['hotel', "new year's eve", 'witch', 'bet', 'hotel room', 'sperm', 'los angeles', 'hoodlum', 'woman di...
18.0	9273	['africa', 'indigenous', 'human animal relationship', 'bat']
19.0	11517	['brother brother relationship', 'subway', 'new york city', 'new york subway', 'train robbery']
20.0	8012	['gambling', 'miami', 'based on novel', 'job', 'murder', 'travel', 'mafia', 'money', 'debt', 'mobster',...

Figura 7: Formato dos atributos

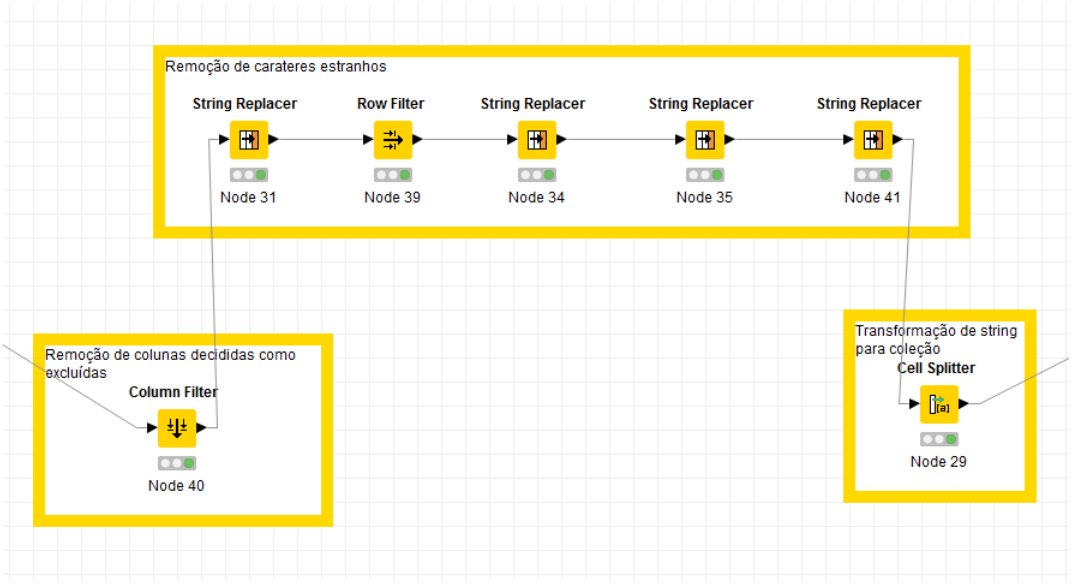


Figura 8: Nodo "Format to collection"responsável pela formatação da informação em *string*

Row ID	id	keywords_SplitResultList
0.0	862	[jealousy,toy,boy,...]
1.0	8844	[board game,disappearance,based on children s book,...]
2.0	15602	[fishing,best friend,duringcreditsstinger,...]
3.0	31357	[based on novel,interracial relationship,single mother,...]
4.0	11862	[baby,midlife crisis,confidence,...]
5.0	949	[robbery,detective,bank,...]
6.0	11860	[paris,brother brother relationship,chauffeur,...]
8.0	9091	[terrorist,hostage,explosive,...]
9.0	710	[cuba,falsely accused,secret identity,...]
10.0	9087	[white house,usa president,new love,...]
11.0	12110	[dracula,spoof]
12.0	21032	[wolf,dog-sledding race,alaska,...]
13.0	10858	[usa president,presidential election,watergate scandal,...]
14.0	1408	[exotic island,treasure,map,...]
15.0	524	[poker,drug abuse,1970s,...]
16.0	4584	[bowling,based on novel,servant,...]
17.0	5	[hotel,new year s eve,witch,...]

Figura 9: Resultado do processo

4 Técnicas

Após um pré-processamento para tornar os dados legíveis e coerentes procedeu-se as técnicas de sistemas de recomendação, onde foi dado mais foco em filtragem colaborativa.

4.1 Sistemas colaborativos

4.1.1 User-based

A primeira técnica que se apresenta é designada como *user-based nearest neighbor recommendation*. Para esta técnica, necessita-se da base de dados *ratings.csv* e do ID de um determinado utilizador que se quer fazer a recomendação; através disso identifica outros utilizadores que têm gostos semelhantes ao gostos prévios do utilizador. Após isto, para cada filme **p** que o utilizador ainda não tenha visto, uma previsão é computada pelas avaliações p feitas pelos *peers* do utilizador. Este método assume que, se os utilizadores tiverem gostos semelhantes no passado, terão gostos semelhantes no futuro e as preferências de cada utilizador mantêm-se estáveis e constantes ao longo do tempo.

	Filme1	Filme2	Filme3	Filme4	Filme5
Carl	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Tabela 1: Avaliação da base de dados para recomendação colaborativa

Uma breve ilustração da técnica pode ser feita da seguinte forma. Examine-se a tabela 1, onde se encontra uma base de dados com avaliações entre 0 a 5 sobre determinados filmes. O objetivo do sistema de recomendação é saber se o Carl irá, ou não gostar do filme5, que este ainda não avaliou ou viu. Antes que se apresentar as fórmulas matemáticas, definiu-se primeiro os termos matemáticos, usa-se $U = u_1, ..., u_n$ para o conjunto de utilizadores, $p = p_1, ..., p_m$ para o conjunto de produtos e $R = n * m$, a matriz de avaliação r_{i_j}, com $i \in 1...n$, $j \in 1...m$.

Para determinar a similaridade entre utilizadores, uma das medidas utilizadas foi a *Pearson's correlation measure*. Após calcular a similaridade entre *Carl* e os restantes utilizadores, do User1 ao User4, obtém-se, respetivamente, 0.85,0.7, 0.0 e -0.67. Com base nestes cálculos é lógico assumir que o User1 e o User2 são os candidatos que apresentam maior similaridade. Para prever a avaliação do Carl (utilizador x) para o Filme5 (produto y), através dos *peers* (p) mais próximos calculados anteriormente, uma possível fórmula é a seguinte, em que $\bar{r_x}$ representa a média de avaliação do utilizador x:

$$pred(x, y) = \bar{rx} + \frac{\sum_{p \in N} sim(x, p) * (rp, y - \bar{rp})}{\sum_{p \in N} sim(a, p)}$$

$$\frac{4 + 1}{(0.85 + 0.7)(0.85(32.4) + 0.70(53.8))} = 4.87$$

Dado este resultado, sendo a escala de 0 a 5, é possível assumir que a decisão a tomar será adicionar o Filme5 na lista do Carl.

Esta foi a técnica implementada relativamente a sistemas colaborativos. Algumas complicações com esta técnica incluem o facto de ser computacionalmente pesada e, apesar de ser uma técnica poderosa, precisa de uma matriz de *ratings* suficientemente grande para que dê bons resultados, por outras palavras, esta técnica é proporcional à esparsidade da matriz de avaliações. Sendo assim, para superar este desafio decidiu-se apresentar inicialmente a um utilizador,enquanto não existem avaliações, os **top N** filmes mais populares e apenas utilizar esta técnica quando o utilizador avalia pelo menos 20 filmes diferentes e que a matriz de *ratings* seja suficientemente grande.

4.1.2 KNN

Esta técnica é designada como *K-Nearest-Neighbor* e é um algoritmo de *Machine Learning* utilizado em problemas de regressão e classificação. Não foi implementada esta técnica de origem, utilizou-se a biblioteca *sklearn* que já continha este método implementado.

Primeiramente, inicializou-se o *trainset* com 80 por cento dos dados e o *testset* com os restantes, em que o *rating* é a variável de classe e *id_user* e *id_movie* são as variáveis independentes. Após a preparação dos dados fez-se um ciclo até 15, para se obter o número ideal de K, utilizando o *mean square errors* (MSE) como *loss function*.

O melhor parâmetro foi para K = 14, como se pode observar pela seguinte imagem.

```
RMSE value for k=  7 is: 1.0853030236119425
RMSE value for k=  8 is: 1.069115434600025
RMSE value for k=  9 is: 1.0675054572387759
RMSE value for k=  10 is: 1.0572014945127537
RMSE value for k=  11 is: 1.0602656380079927
RMSE value for k=  12 is: 1.0485936348801241
RMSE value for k=  13 is: 1.04922837243128
RMSE value for k=  14 is: 1.0364529385734553
RMSE value for k=  15 is: 1.0362619788880074
RMSE value for k=  16 is: 1.0416755207957036
RMSE value for k=  17 is: 1.0375685244390245
```

Figura 10: Número ideal de K

Após o cálculo do número ideal de vizinhos, define-se o modelo com K=14, e obtêm-se assim uma nova técnica que, através de um identificador do utilizador e um identificador de um filme, prevê a avaliação desse utilizador ao filme. Apesar desta técnica não ser muito precisa, decidiu-se utilizá-la devido à sua simplicidade.

4.2 Sistemas baseados em conteúdo

Para que o utilizador não tenha que esperar muito para que uma recomendação seja feita, decidiu-se implementar outra técnica não tão poderosa como a *User-based* mas que é capaz de apresentar resultados satisfatórios. Serão explicados, sucintamente, os passos na implementação desta técnica mas, resumidamente, este método utiliza *clusters* para fazer previsões, ou seja, dado um conjunto de características de um determinado filme, o método determina o *cluster* em que o filme está. Caso o utilizador avalie positivamente este filme, então são-lhe sugeridos novos filmes dentro do mesmo *cluster*.

4.3 *Clustering* sobre características do filme

Como mencionado no capítulo 3, tem-se vários ficheiros relativos aos filmes. Esta técnica é utilizada no *movie_dataset.csv*, dado as suas características. Esta técnica recebe atributos específicos como parâmetro para fazer os *clusters*, respetivamente, *year*, *voteaverage*, *popularity*, *runtime*, *revenue* e *tags*. Primeiramente, calcula-se o número ideal de *clusters* para este respetivo *dataset* através de um ciclo, como se pode observar na imagem 11.

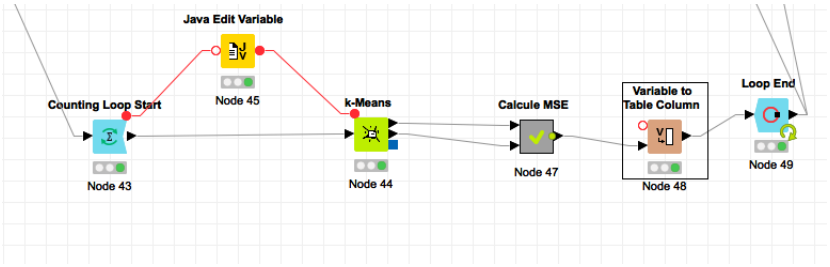


Figura 11: Determinação do número ideal de *clusters*

Este ciclo, começa com uma variável correspondente ao número de *clusters*. Neste caso começa a 1 e, em cada iteração, calcula o MSE (*mean square error*) associado a essa variável e incrementa uma unidade. Este ciclo acaba quando esta variável chega aos 15. O resultado obtido foi o seguinte:

Row ID	Mean(...	k	Iteration
Row0#0	0.255	1	0
Row0#1	0.232	2	1
Row0#2	0.238	3	2
Row0#3	0.215	4	3
Row0#4	0.207	5	4
Row0#5	0.199	6	5
Row0#6	0.191	7	6
Row0#7	0.196	8	7
Row0#8	0.19	9	8
Row0#9	0.186	10	9
Row0#10	0.18	11	10
Row0#11	0.178	12	11
Row0#12	0.176	13	12
Row0#13	0.17	14	13
Row0#14	0.169	15	14

Figura 12: Resultados do ciclo

Como se pode observar pela imagem 13, o maior salto foi entre a iteração 3 e 4. Segundo o método do cotovelo, o melhor número para a variável representativa é, respetivamente quatro. Sendo assim, utilizaram-se quatro *clusters* e procedeu-se com a recomendação. Finalizando, após calculado o número ideal de *clusters*, ou seja, aquele que minimiza a função objetivo, neste caso, *mean square errors*, é utilizado o *K-means* com quatro *clusters* e os atributos referidos anteriormente. Após todos os filmes estarem no seu respetivo *cluster*, procede-se à recomendação. Para os filmes com uma melhor avaliação do utilizador, encontram-se pelos menos três filmes do mesmo *cluster* com a maior popularidade e são recomendados todos os encontrados. Apesar de ser um método rápido e escalável pois todo este processamento é feito *offline*, ou seja, não ocorre *delay* na resposta, esta

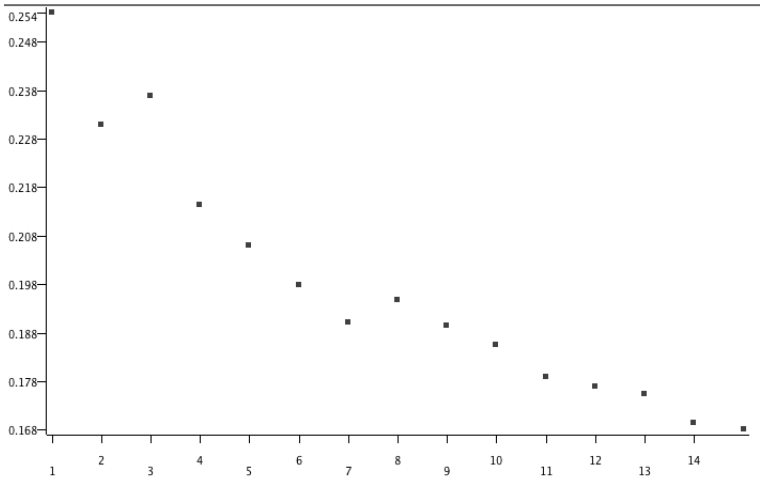


Figura 13: Análise de resultados

técnica apresenta alguns problemas quando a similaridade entre filmes não é claramente definida. Porém, mesmo quando a similaridade é clara, os resultados do sistema baseados em conteúdo tendem a ser muito homogéneos. Isso faz com que os filmes recomendados nunca caiam fora da zona de conforto definida no início do registo dos utilizadores. "As pessoas mudam com o tempo, assim como mudam suas preferências. Sistemas de conteúdo simples têm dificuldade para acompanhar essas mudanças".

4.4 Pesquisa por similaridade

Uma das principais vertentes dos sistemas baseados em conteúdo baseia-se na busca de objetos semelhantes. Este conceito é aplicado aqui através da utilização da fórmula de Dice:

$$\frac{2 * |keywords(x) \cap keywords(y)|}{|keywords(x)| + |keywords(y)|}.$$

Esta fórmula foi utilizada para dois ficheiros em separado, nomeadamente, o ficheiro de créditos e palavras chave, de forma a produzir uma maior variedade de resultados que permitissem maiores potenciais vertentes de interpretação da base de dados de objetos em questão.

Como referido anteriormente na secção do tratamento de dados, ambas as colunas de maior interesse destes ficheiros foram transformadas para objetos KNIME do tipo coleção. Isto foi feito para permitir a aplicação desta buscar de similaridade. Através da transformação destas coleções para vetores de *bits*, *input* necessário no programa para que fosse efetuada a buscar, foi possível comparar as diferentes instâncias em termos da similaridade e retirar os vinte mais parecidos para cada filme (número decidido pelos autores como suficiente para a o problema em questão). Existiria, então, uma tabela que contivesse, para cada filme, os vinte filmes mais semelhantes referidos pelo seu ID. Esta foi a operação efetuada para ambos os ficheiros.

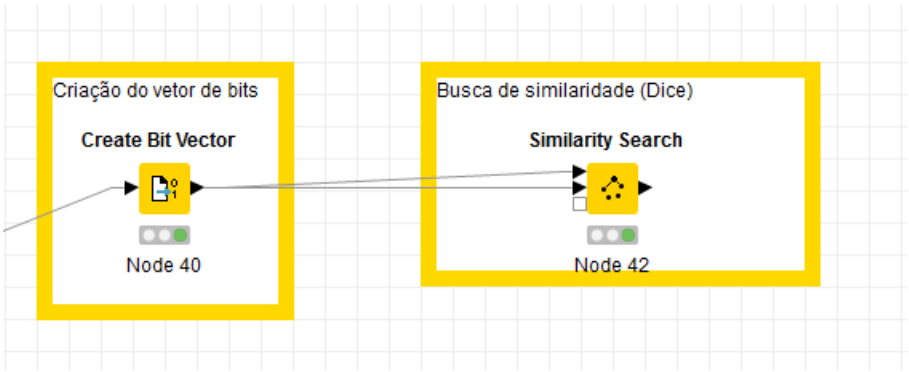


Figura 14: Nodo "Similarity Search"responsável pelo *output* de uma tabela de N instâncias mais semelhantes em relação a uma estabelecida previamente (N = 20).

Row ID	id	nearest neighbor - index	nearest neighbor - id	distance
24403.0_1	204762	0	44284	0
24403.0_2	204762	1	52856	0
24403.0_3	204762	2	11017	0
24403.0_4	204762	3	218473	0
24403.0_5	204762	4	23637	0
24403.0_6	204762	5	124472	0
24403.0_7	204762	6	85778	0
24403.0_8	204762	7	123763	0
24403.0_9	204762	8	172198	0
24403.0_10	204762	9	213917	0
24403.0_11	204762	10	117036	0
24403.0_12	204762	11	124676	0
24403.0_13	204762	12	41326	0
24403.0_14	204762	13	96288	0
24403.0_15	204762	14	41240	0
24403.0_16	204762	15	124843	0
24403.0_17	204762	16	201445	0
24403.0_18	204762	17	124642	0
24403.0_19	204762	18	106129	0
24403.0_20	204762	19	65046	0

Figura 15: *Output* da *Similarity Search* relativo a *keywords*, para um filme com ID 204762; o *output* relativo ao elenco é muito semelhante, pois apenas mudam os argumentos de procura.

4.5 Regras de associação

O conceito de regras de associação está intimamente ligado com a associação da presença de produtos ao longo de um *dataset* e vai aqui ser aplicado ao ficheiro referente a classificações dadas pelos utilizadores, nomeadamente "ratings.csv".

Para a aplicação desta noção, foi necessário determinar os parâmetros denominados por *support* e *confidence* que refletem as proporções de presenças de ambos os predicados (antecedente e consequente) nos dados. Os valores optados foram 0,1 para *support* e 0,4 *confidence*. Devido ao tema deste *dataset*, filmes, os autores reconheciam uma grande esparsidade no que toca a gostos, ou seja,

existem muitos filmes de categorias e características diferentes assim como muitas variedades e combinações de gostos por parte dos utilizadores. Para os valores de ambos os parâmetros, considerou-se esta conclusão, o número de entradas no ficheiro ao qual seria aplicado este método, o ficheiro de *ratings* e o número de filmes existentes na base de dados, ou seja, o número de entradas no ficheiro de filmes. A relação entre o número de entradas nos dois ficheiros está intrinsecamente relacionada com a esparsidade de dados e é determinante. No caso em questão, apresenta-se aproximadamente quarenta e cinco mil filmes num e seiscentos e cinquenta filmes noutro, o que consiste numa relação considerada desequilibrada pois, para uma grande variedade de filmes, havendo um relativamente pequeno número de utilizadores, em condições normais, apenas uma relativamente pequena fracção das transações (par antecedente-consequente) possíveis vai ser exibida, o que diminui drasticamente as probabilidades de transações comuns entre utilizadores. A partir deste princípio, definiu-se um valor de *support* relativamente baixo para ilustrar a esparsidade da amostra. Para o valor de *confidence*, a decisão foi tomada baseada na tolerância a coincidências relativas à esparsidade: para um potencial pequeno número de amostras, define-se que constitui uma regra de associação se a relação dos predicados se verificar em grande número, tendo em conta a variedade de opções.

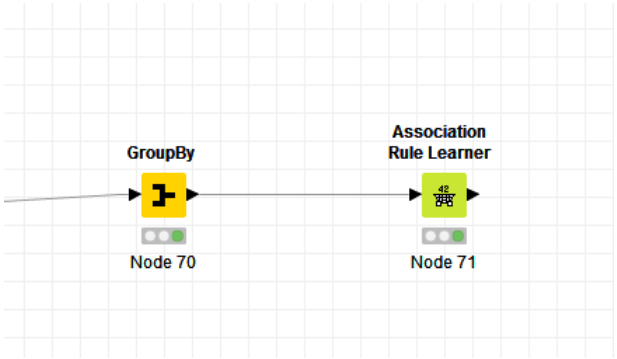


Figura 16: Nodo "Association Rules"responsável pelo *output* de uma tabela contendo as regras de associação segundo os parâmetros *support* e *confidence*.

Row ID	D Support	D Confide...	D Lift	I Conseq...	S implies	[...] Items
rule0	0.101	0.642	3.528	10	<---	[434]
rule1	0.101	0.557	3.528	434	<---	[10]
rule2	0.101	0.567	1.892	47	<---	[39]
rule3	0.101	0.567	1.558	527	<---	[39]
rule4	0.101	0.459	2.569	39	<---	[34]
rule5	0.101	0.567	2.569	34	<---	[39]
rule6	0.101	0.43	2.407	39	<---	[231]
rule7	0.101	0.567	2.407	231	<---	[39]
rule8	0.101	0.567	1.539	1	<---	[39]
rule9	0.101	0.624	2.083	47	<---	[223]
rule10	0.101	0.654	2.183	47	<---	[1222]
rule11	0.101	0.493	1.645	47	<---	[2628]
rule12	0.101	0.472	1.576	47	<---	[1193]
rule13	0.101	0.553	1.846	47	<---	[1923]
rule14	0.101	0.482	1.61	47	<---	[6539]
rule15	0.101	0.548	1.831	50	<---	[778]
rule16	0.101	0.463	1.544	50	<---	[2716]
rule17	0.101	0.511	1.707	50	<---	[2918]
rule18	0.101	0.511	1.505	110	<---	[2918]
rule19	0.101	0.673	2.259	150	<---	[339]

Figura 17: *Output* parcial do *Association Rule Learner*

4.6 Sistemas Híbridos

Dado que todas as técnicas contêm os seus problemas, uma maneira de superar este desafio foi o desenvolvimento de sistemas híbridos que combinam os melhores aspetos de todas estas técnicas conseguindo assim uma melhor precisão na recomendação. Dito isto, foi desenvolvida uma técnica que tira partido de todos os métodos de recomendação desenvolvidos, três relacionados com a filtragem colaborativa, *User-based*, *KNN* e *Association Rules* e dois relativos a sistemas baseados em conteúdo, *content-based* e *online similarity search*. Apenas esta técnica, *hybrid system*, é utilizada na recomendação, já que é um método que envolve todos os outros.

A ideia no desenvolvimento deste trabalho foi sem dúvida, ter mais foco na filtragem colaborativa. Portanto, também é de esperar que os sistemas híbridos tenham mais em conta a recomendação feita por técnicas deste tipo. A arquitetura utilizada nesta técnica híbrida foi do tipo *monolithic*, representada na imagem 18.

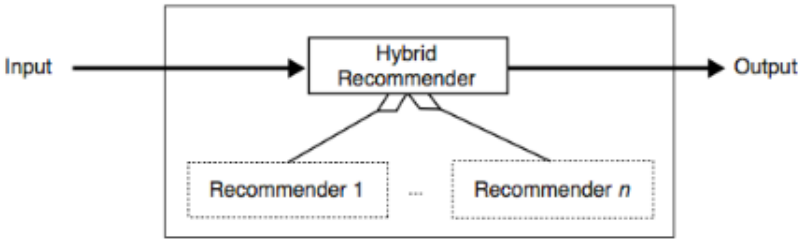


Figura 18: *Monolithic System*

Numa fase inicial, todas as técnicas, exceto *User-based* são utilizadas e após retornarem a previsão de específicos filmes, observam-se quais os filmes que foram recomendados em todas as técnicas, pois só esta coordenação, já evidencia que será vantajoso recomendar o filme. Seguidamente, é feita a média das previsões para todos os filmes que apareçam simultaneamente nas recomendações de todas as técnicas. A maior previsão obtida pela média deste conjunto de técnicas é a recomendada. Caso não ocorra a previsão do mesmo filme por todas as técnicas, observa-se se acontece pelo menos em 4 técnicas e assim sucessivamente. No caso de nenhum filme aparecer em pelo menos duas técnicas, devolve-se os filmes que obtiveram a melhor previsão.

No caso de o utilizador já ter algumas avaliações, podendo já ser utilizado o *user-based*, então este entra para a conta com um peso de três, ou seja, o resultado desta técnica é contado três vezes no cálculo da média, tendo um peso maior que os outros métodos.

5 Interface

Na realização da *Interface*, utilizamos o **Django**. O Django é uma *framework* de alto nível para criar aplicações *Web* através de Python. Utilizamos esta *framework* pois incentiva um desenvolvimento rápido e o *design* é limpo e pragmático. Uma das grandes razões da utilização desta ferramenta é que, apesar de ser *open-source*, cuida de parte "chata"do desenvolvimento *Web*, para que os criadores se possam concentrar em construir a aplicação sem precisar de pensar muito acerca dos detalhes do *design* como acontece, por exemplo, em HTML. Para além disso, todas as técnicas foram desenvolvidas em Python, por isso, uma *framework* que a utilize só facilita o processo.

Por exemplo, todo o processo administrativo foi tratado pelo Django como podemos observar nas imagens 19 e 20.

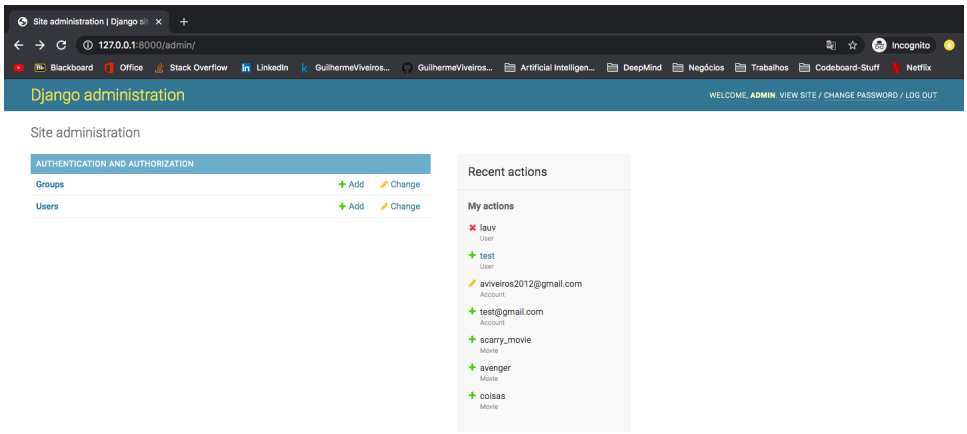


Figura 19: Página do administrador

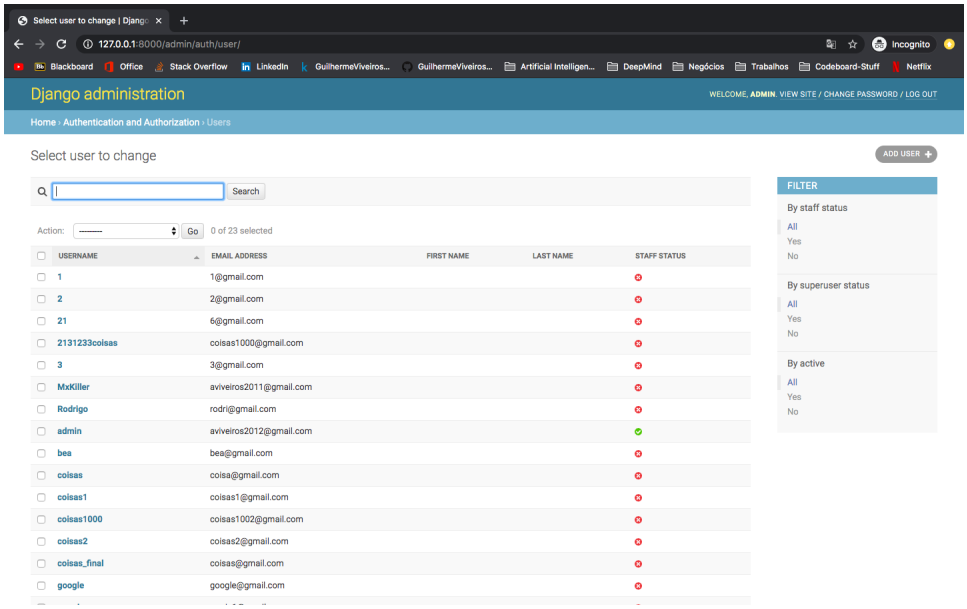


Figura 20: Vista do administrador sobre os utilizadores

Tanto o *sing up* como o *login* são apresentados, como podemos observar na imagem 21.

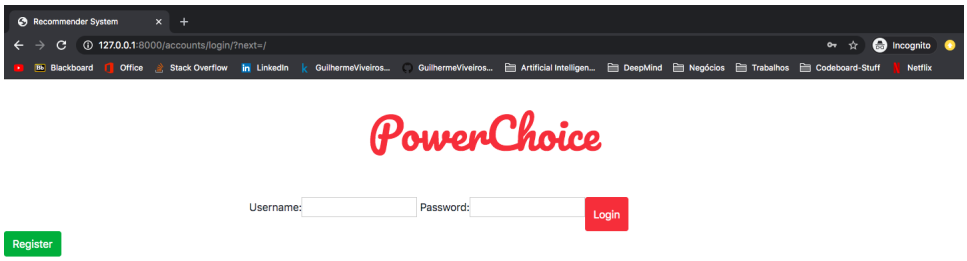


Figura 21: Página inicial

Após o registo de um utilizador no *site*, é lhe atribuído um determinado identificador, que vai ser utilizado, por exemplo, quando o utilizador avaliar um determinado filme. Inicialmente, quando o utilizador ainda não avaliou nenhum filme, vai ser direcionado, após a sua autenticação, para uma página onde são exibidos os doze filmes mais vistos, como se pode observar na imagem 22.

Dado a dimensão do *dataset*, em vez de se guardar todas as imagens relativas aos filmes, espera-se o resultado das técnicas de recomendação, ou seja, os título dos filmes e, após isto, fez-se um pequeno *script* em Python que retira as imagens relativas aos filmes da Internet e as redimensiona para que tenham a mesma dimensão. Não houve muito cuidado na resolução da imagem pois o objetivo desta interface é a exploração dos algoritmos desenvolvidos para as técnicas de recomendação. Agora imagine-se que o utilizador quer avaliar um filme. Suponha-se que este gostou de ver *Hunger*

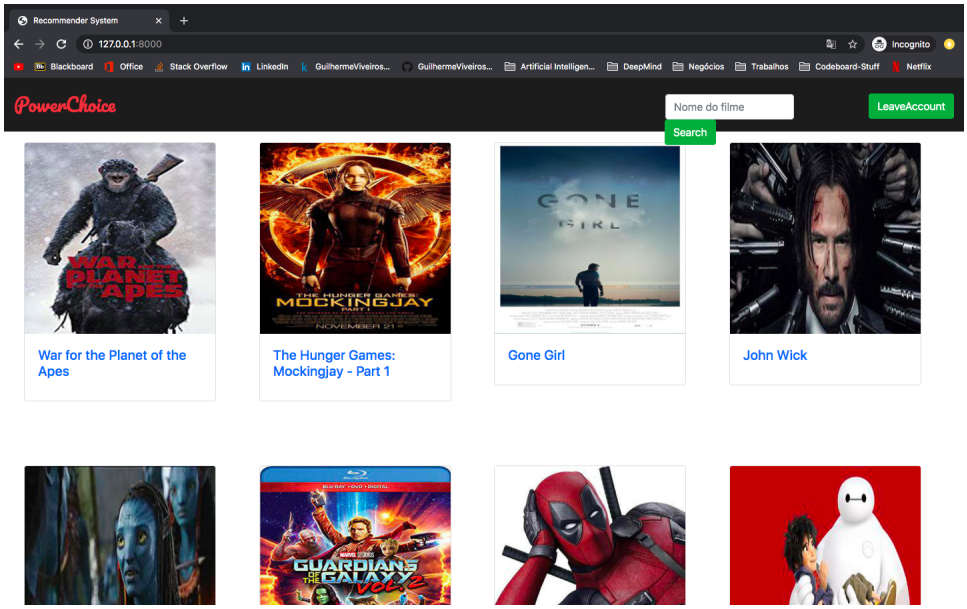


Figura 22: Filmes mais vistos

Games e irá o avaliar com uma *rate* de 5. Esta função está disponível no *site* como podemos observar na imagem 23.

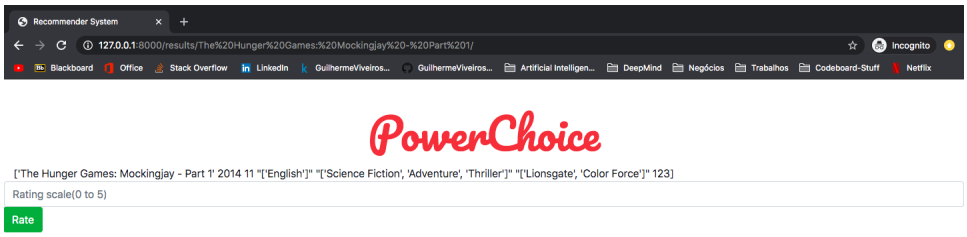


Figura 23: Avaliação de *Hunger games*

Após o utilizador avaliar o filme, os sistemas de recomendação entram em ação. Como explicado anteriormente, para este caso apenas algumas técnicas entram em funcionamento dentro dos sistemas híbridos devido ao número de avaliações do utilizador ser muito pequeno. Após as operações dos sistemas híbridos estarem concluídas, um possível resultado seria o apresentado na imagem 24.

Como podemos observar, entraram filmes novos para a página principal, representantes do resultado dos sistemas de recomendação. Salienta-se novamente que, quanto mais avaliações existirem, melhores serão as recomendações de novos filmes.

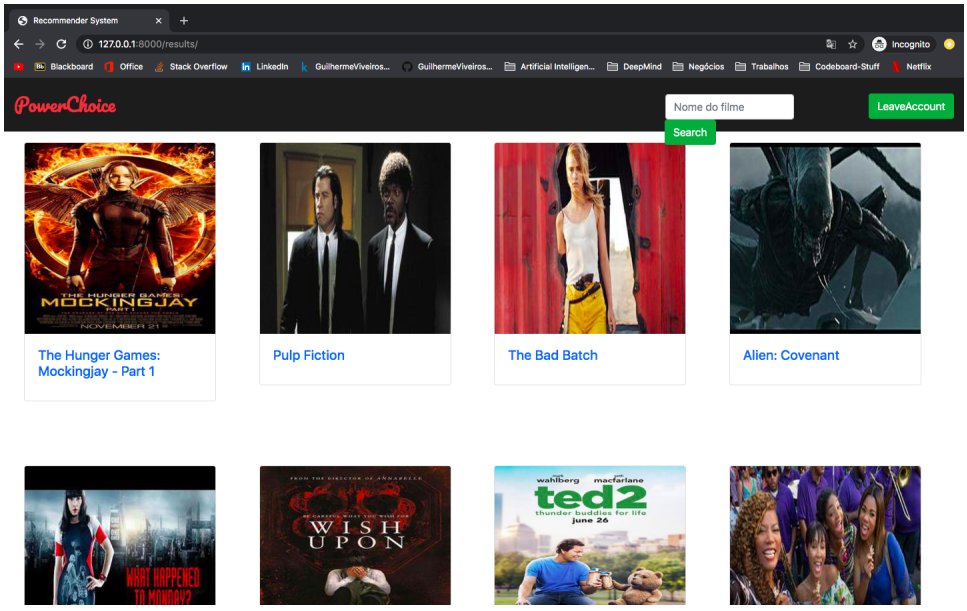


Figura 24: Recomendações

6 Conclusão

Após a realização deste trabalho, o grupo ficou mais familiarizado com a ferramenta KNIME e do seu poder em relação à manipulação de dados. Para além disso percebeu-se melhor como as técnicas de recomendação funcionam e os algoritmos utilizados. Conclui-se que o nosso sistema de recomendação ainda está longe de estar finalizado, pois apesar de serem utilizadas técnicas bastantes poderosas, não se teve tempo de as afinar devido ao tempo de trabalho. Também não se conseguiu investir em técnicas mais poderosas e engraçadas, como por exemplo, redes neuronais, que poderiam dar um ótimo resultado dado à dimensão dos dados existentes.