
Sistema de Recomendação baseado em Reinforcement Learning: uma prova de conceito aplicada ao Video on Demand

Daniel Carvalho Marques

Dissertação
Mestrado em Gestão Comercial

Orientado por
Professor Doutor Pedro José Ramos Moreira de Campos

2021

Agradecimentos

Em primeiro lugar, gostaria de expressar a minha mais sincera gratidão ao meu orientador, Professor Doutor Pedro José Ramos Moreira de Campos, por acreditar desde o início no sucesso deste projeto, por todo o apoio, disponibilidade, encorajamento e partilha de conhecimento, fatores que foram fundamentais para a realização desta dissertação.

Gostaria também de agradecer a todos os meus amigos que me acompanharam nesta etapa por todo o seu apoio e motivação.

Um agradecimento à minha família, em especial aos meus pais, pela oportunidade que me forneceram de realizar este mestrado, mas também por todo o apoio incondicional, por todos os valores que me transmitiram e por sempre me motivarem a acreditar em mim.

Por último, mas não menos importante, quero expressar o meu mais profundo agradecimento, à minha namorada, Sílvia, por me motivar e ajudar a superar todos os desafios académicos, pessoais e profissionais, sem ela esta dissertação não teria sido possível.

Resumo

Com a presença de consumidores nos meios digitais, existe um crescente excesso de informação disponível e, atualmente, os consumidores não querem simplesmente opções, mas sim sugestões adequadas ao seu perfil. Os sistemas de recomendação desempenham um papel fundamental na compra de produtos/serviços, uma vez que procuram de entre um grande conjunto de dados quais os que mais se adequam às necessidades e interesses de um dado utilizador em específico. Um bom sistema de recomendação, ao sugerir conteúdos interessantes, ajuda a melhorar a experiência do utilizador e o seu interesse, levando a uma maior retenção e lealdade, o que pode significar um aumento das receitas para a empresa.

Nesta dissertação, como forma de complementar os trabalhos existentes sobre este tema, foi criado um sistema de recomendação adaptado ao mercado do *video on demand* recorrendo para isso a uma área do *Machine Learning* denominada *Reinforcement Learning*.

O objetivo principal da criação deste sistema é provar que o uso de um algoritmo *Q-Learning* baseado numa cadeia de Markov tabular tem a capacidade de realizar recomendações eficientes e personalizadas com base nos interesses específicos de um dado utilizador.

Esta solução é inovadora, pois ao contrário de outros métodos mais sofisticados, permite oferecer recomendações de forma simples e controlada.

Como forma de atingir esse fim, nesta dissertação foi desenvolvido não só o algoritmo que permite obter as recomendações a efetuar, mas também uma interface que permite apresentar os filmes ao utilizador e obter o *feedback* por parte do mesmo.

Para comprovar a premissa em estudo, foram realizados 100 testes com diferentes utilizadores sendo estes divididos em duas fases. Foram usadas várias métricas para analisar os resultados obtidos. De modo a comprovar a melhoria criada pelo sistema de recomendação em estudo foi usado na última fase, como termo de comparação, um sistema de recomendação baseado em *Collaborative Filtering*.

Palavras-chave: Sistema de Recomendação, Recomendação de Filmes, Aprendizagem por Reforço, *Q-learning*, *Machine learning*, Filtragem colaborativa

Abstract

With the presence of consumers in digital media, there is an increasing excess of information available and consumers do not just want options, but rather suggestions suited to their profile. Recommendation systems play a fundamental role in the purchase of products/services, as they search from a large set of data for those that best suit the needs and interests of a specific user. A good recommendation system by suggesting interesting content helps not only to improve the user experience but also their interest, leading to greater retention and loyalty, which can mean an increase in revenue for the company.

To complement the existing works on this topic, it was created a recommendation system adapted to the video on demand market, using a field of Machine Learning called Reinforcement Learning.

The main purpose of creating this system is to prove that a Q-learning algorithm based on a tabular Markov chain can make efficient and personalized recommendations based on the specific interests of a given user.

This solution is innovative because, unlike other more sophisticated methods, it allows to offer recommendations in a simple and controlled way.

In this dissertation it was developed an algorithm that allows us to obtain recommendations and an interface to present the films to the user and obtain feedback from them.

To prove the premise under study, 100 tests were performed with different users, which were divided into two phases. In addition, several metrics were used to analyze the results obtained. Furthermore, to prove the improvement generated by the recommendation system in the study, a recommendation system based on Collaborative Filtering was used in the last phase, as a term of comparison.

Keywords: Recommendation System, Movie Recommendation, Reinforcement Learning, Q-learning, Machine learning, Collaborative Filtering

Índice

Capítulo 1	1
Introdução.....	1
Capítulo 2	5
Revisão de Literatura	5
2.1. Inteligência Artificial e Machine Learning.....	5
2.1.1. Inteligência artificial	5
2.1.2. Machine Learning.....	5
2.2. Tipos de Machine Learning.....	6
2.2.1. Aprendizagem Supervisionada	7
2.2.2. Aprendizagem Não Supervisionada.....	7
2.2.3. Aprendizagem por Reforço/Reinforcement Learning	8
2.2.3.1. Q-Learning.....	10
2.2.3.2. Markov Decision Process	11
2.3. Sistemas de Recomendação.....	11
2.3.1. Fases do Processo de recomendação.....	12
Fase da Recolha de Informação.....	13
Fase de Aprendizagem	13
Fase de Previsão/ Recomendação.....	13
2.4. Técnicas de filtragem de recomendação.....	14
2.4.1. Collaborative Filtering	14
2.4.1.1. Item-based.....	14
2.4.1.2. User-based.....	15
2.4.2. Content-based Filtering.....	15
2.4.3. Hybrid Filtering	15

2.5. Trabalho Relacionado.....	15
2.6. Análise de resultados do algoritmo Q-Learning.....	16
2.7. Avaliação de Sistemas de Recomendação	17
2.7.1. Avaliação Offline	17
2.7.2. Prediction Accuracy	18
2.7.2.1. MAE.....	18
2.7.2.2. MSE.....	19
2.7.2.3. RMSE.....	19
2.7.3. Decision Support Metrics.....	19
2.7.4. Coverage.....	20
2.7.5. Popularity	21
2.7.6. Diversity Measures	21
2.7.6.1. ILS	21
2.7.6.2. Category Diversity	22
2.7.7. Outros métodos de avaliação.....	22
2.7.7.1. Ranking Based Methods.....	22
2.7.7.2. Temporal Evaluation	22
2.7.7.3. Percieved Quality.....	22
Capítulo 3.....	24
Metodologia e Dados: Descrição do Sistema.....	24
3.1. Definição do Problema	24
3.2. Solução Proposta.....	25
3.3. Arquitetura do Sistema.....	28
3.3.1. Q-Learning.....	28
3.3.2. Desenvolvimento do Sistema de recomendação	29
3.3.3. Cold Start	34
3.3.4. Base de Dados.....	35

3.3.5. Interface Gráfica.....	35
3.3.6. Esquema representativo do funcionamento do sistema de recomendação desenvolvido	37
3.4. Sistema de recomendação usado para comparação	39
Capítulo 4	42
Resultados e Avaliação	42
4.1. Dados recolhidos para avaliação.....	42
4.2. Análise dos resultados do algoritmo Q-Learning.....	43
4.3. Medidas de Avaliação do Sistema de Recomendação	44
4.3.1. Prediction Accuracy	44
4.3.1.1. MAE.....	44
4.3.1.2. MSE.....	45
4.3.1.3. RMSE.....	45
4.3.2. Decision Support Metrics.....	46
4.3.3. Coverage	48
4.3.4. Popularity.....	48
4.3.5. Diversity Measures	49
4.3.5.1. ILS	49
4.3.5.2. Category Diversity.....	50
Capítulo 5	52
Conclusões	52
5.1. Sugestões de melhoria para trabalhos futuros e limitações.....	53
Referências	56
Anexos	62
Anexo 1 - Exemplificação do funcionamento do sistema de recomendação em estudo ..	62
Anexo 2 - Exemplificação do funcionamento do sistema de recomendação usado como termo de comparação	62

Anexo 3 – Código do sistema de recomendação em estudo	62
Anexo 4 – Fluxograma	63
Anexo 5 - Base de dados de Filmes.....	63
Anexo 6 – Resultados das iterações com utilizadores.....	64
Anexo 7- Código do sistema de recomendação usado como termo de comparação.....	64
Anexo 8 - Recompensa e Arrependimento de cada utilizador	64
Anexo 9 – Resultados da Popularidade de cada filme para os dados da fase de pré-teste e teste.....	64
Anexo 9b - Resultados da Popularidade de cada filme para os dados da fase de teste.....	65
Anexo 10 – Combinações do grau de similaridade entre itens.....	65
Anexo 11 – Número de filmes recomendados por categoria a cada utilizador da fase de teste.....	65
Anexo 12 – Conversor de resultados do algoritmo.....	66

Índice de figuras

Figura 1: Enquadramento típico de um cenário de Reinforcement Learning.....	8
Figura 2: Fase do Processo de Recomendação	12
Figura 3: Modelo Q-Learning em Excel	29
Figura 4: Primeiro resultado obtido por parte do algoritmo.....	33
Figura 5: Introdução das preferências do utilizador face à recomendação apresentada	33
Figura 6: Conversor das recomendações fornecidas pelo algoritmo	36
Figura 7: Interface gráfica usada para a apresentação de filmes	37
Figura 8: Esquema representativo do funcionamento do sistema de recomendação desenvolvido	38

Índice de tabelas

Tabela 1: Classificação dos possíveis resultados da recomendação de um item a um utilizador	20
Tabela 2: Conversão do valor da recompensa.....	30
Tabela 3: Resultados da recompensa e arrependimento médio para os dados de pré-teste e teste.....	44
Tabela 4: Resultado do Erro Absoluto Médio para os dados de pré-teste e de teste.....	44
Tabela 5: Resultado do Erro Médio Quadrático para os dados de pré-teste e de teste	45
Tabela 6: Resultado da Raiz do Erro Quadrático para os dados de pré-teste e de teste.....	45
Tabela 7: Resultado da Precisão para os dados de pré-teste e de teste.....	46
Tabela 8: Evolução da Precisão ao longo das 6 diferentes slates	47
Tabela 9: Resultado da Diversidade para os dados de teste	49
Tabela 10: Resultado da Diversidade de categoria para os dados de teste.....	50

Índice de gráficos

Gráfico 1: Evolução da Precisão ao longo das 6 diferentes slates	47
Gráfico 2: Percentagem do número de vezes que cada filme foi recomendado face ao total de recomendações realizadas	48

Lista de acrónimos

BD – Base de Dados

CF - *Collaborative Filtering*

CTR – *Click Through Rate*

ILS - *Intra-List Similarity*

KNN - *K-Nearest Neighbours*

MAE – *Mean Absolute Error*

MDP - *Markov Decision Process*

ML - *Machine Learning*

MSE - *Mean Squared Error*

RL - *Reinforcement Learning*

RMSE - *Root Mean Squared Error*

SR - Sistema de Recomendação

Capítulo 1

Introdução

Ao longo dos últimos anos, o mercado tradicional tem vindo a sofrer diversas mudanças, principalmente com a rápida evolução das tecnologias da informação que têm vindo a revolucionar tanto os comportamentos do consumidor como os comportamentos das empresas.

De facto, algo que a pandemia atual veio intensificar foi, precisamente, o forte papel que a tecnologia possui na sociedade. Empresas que nunca ponderaram adotar canais digitais veem-se agora sem outra alternativa e o mesmo se passa com os consumidores.

Com a crescente presença de consumidores nos meios digitais, existe cada vez mais um excesso de informação disponível (Rathore & Farooq, 2020) e, tendo em conta o escasso tempo do consumidor, torna-se fundamental que lhe sejam apresentados produtos/serviços ideais de forma personalizada ao seu perfil.

Considerando esta problemática, surgem os sistemas de recomendação como forma de filtrar esse excesso de informação (Esmacili et al., 2020) e apresentar ao consumidor apenas os produtos/serviços que ele provavelmente está interessado em adquirir. Estes sistemas permitem um maior envolvimento do consumidor com a plataforma, o que permite gerar um maior *Click Through Rate* (CTR) e conversões, que são medidas standard usadas para medir o desempenho dos sistemas de comércio eletrónico.

Os sistemas de recomendação já são utilizados de forma bastante abrangente em diferentes setores, que vão desde o entretenimento, como é o caso da Netflix¹, até o retalho como é o caso da Amazon.

De acordo com um estudo realizado pela McKinsey em 2013 (MacKenzie et al., 2013), 75% do que é assistido na Netflix provém de recomendações por parte da plataforma, bem como 35% do que os consumidores compram na Amazon.

¹ “Os algoritmos de recomendação são a base do produto Netflix” (Netflix Research, 2021): <https://research.netflix.com/research-area/recommendations>

Contudo, muito frequentemente os sistemas tradicionalmente usados possuem uma falha ao considerar o consumidor como alguém que possui preferências estáticas (Poriya et al., 2014), algo que certamente não se verifica no mundo real. O consumidor evolui e, naturalmente, também os seus gostos e desejos. Considerando esta premissa, prova ser importante a existência de algum sistema que seja adaptável a essas mudanças.

De facto, sistemas de recomendação que efetuam recomendações personalizadas para cada um dos utilizadores de acordo com as suas preferências são fulcrais para uma boa experiência *online*, pois permitem recomendar os melhores itens e ajudam o cliente a tomar melhores decisões, o que em última análise aumenta a sua satisfação, algo que facilmente se pode converter em receitas. Por exemplo, um canal de *e-commerce* que consiga melhorar as suas recomendações potencia um melhor *up-selling* e *cross-selling*.

Como forma de solucionar esta limitação, muitos sistemas de recomendação são mais personalizados (Rashid et al., 2002), mas surge a oportunidade de usar novas técnicas que permitam melhorar o objetivo da recomendação personalizada. É precisamente aí que entra a relevância de aliar a aprendizagem por reforço (*Reinforcement Learning*) aos sistemas de recomendação. Este tipo de *machine learning* (ML) funciona com base no *feedback* recebido através de uma abordagem de tentativa e erro, na qual um *feedback* positivo confere uma recompensa e um *feedback* negativo uma penalização, sendo que objetivo principal é maximizar as recompensas recebidas (Sutton & Barto, 1998). Desta forma, o algoritmo evolui sistematicamente ao longo do tempo de modo a concluir de forma personalizada e em tempo real quais os produtos ou serviços que um consumidor com dadas características está interessado.

Nesta dissertação será criado um sistema de recomendação (SR) baseado em *Reinforcement Learning* usando para tal um algoritmo *Q-Learning* baseado numa cadeia de *Markov* tabular, que permite encontrar itens adequados a recomendar aos clientes. Este sistema desenvolvido foi nomeado **RQ-Markov - Recommender System based on a Q-learning Tabular Markov Chain**. Apesar de existirem diferentes abordagens para desenvolver este algoritmo, optou-se por esta em detrimento de outras, tais como o *deep reinforcement learning*, pois são essencialmente baseadas em modelos *black box* e com muitas limitações de interpretabilidade, apesar de serem muito rápidas.

Como forma de avaliar a qualidade do sistema desenvolvido, foram realizados testes num ambiente controlado no qual participaram 100 utilizadores. Este ambiente terá como objetivo ser usado como prova de conceito face à utilidade que o sistema de recomendação desenvolvido tem para um utilizador real e concluir se as recomendações apresentadas são adequadas ao perfil do consumidor em questão, potenciando assim uma melhor experiência para o mesmo.

Testou-se o sistema para um mercado específico, o da visualização de filmes (*video on demand*), onde o *streaming* (ex.: Netflix) tem assumido um papel importante. Este mercado foi selecionado devido às suas especificidades tal como será explicado no capítulo 3, no entanto o SR desenvolvido pode facilmente ser adaptado para outros setores.

Foi criado um sistema de recomendação completo, sendo que isto irá implicar a criação de um código em R que permitirá obter a recomendação, mas também foi desenvolvida uma interface que vai permitir não só converter o output gerado pelo código em recomendações concretas, mas também apresentar os filmes e captar as preferências do consumidor.

Um outro ponto de enorme relevo desta dissertação diz respeito ao uso de dados reais desde o momento zero, sendo que tanto na fase de pré-teste como na fase de teste todos os dados serão obtidos através de iterações com indivíduos reais evitando assim o uso de ambientes simulados ou bases de dados previamente recolhidas. Entenda-se por pré-teste o conjunto de dados que serviram para “alimentar” os algoritmos com informações reais de utilizadores e a fase de teste a fase que já possui incorporado esses dados.

Como forma de fortalecer esta análise e retirar conclusões sobre uma possível melhoria gerada pelo SR em estudo, será também usado um sistema de recomendação baseado em *collaborative filtering* (CF) que tem como função ser usado como termo de comparação com o sistema de recomendação em análise aquando da análise da sua performance.

Por fim, é espectável que após todo o trabalho desenvolvido seja possível não só provar a utilidade da aprendizagem por reforço em sistemas de recomendação, mas também provar que o sistema de recomendação desenvolvido tem o potencial de ser um fator chave para uma maior vantagem competitiva para qualquer empresa que queira recomendar itens

de forma eficiente aos seus clientes. O sistema desenvolvido pelo autor é inovador e pode ser implementado nos sistemas de comércio eletrônico das empresas.

Esta dissertação é composta por cinco capítulos, incluindo a introdução. No capítulo 2 é feita uma revisão de literatura detalhada que abrange todos os conceitos teóricos necessários à compreensão da solução proposta, bem como uma análise da literatura existente acerca do tema. De seguida, no capítulo 3, são referidos os métodos usados que permitiram dar resposta ao problema proposto e será apresentada a arquitetura do sistema, sendo também descritas todas as fases da construção do sistema de recomendação, bem como os softwares e linguagens de programação usados. No capítulo 4, serão apresentados os resultados obtidos e a sua avaliação. Por fim, no capítulo 5 serão apresentadas as principais conclusões, as limitações do estudo e possíveis sugestões de melhoria.

Capítulo 2

Revisão de Literatura

2.1. Inteligência Artificial e *Machine Learning*

2.1.1. Inteligência artificial

Uma das primeiras e mais conhecidas definições de Inteligência Artificial (IA) é fornecida por Alan Turing que refere o seguinte: *“If there is a machine behind a curtain and a human is interacting with it (by whatever means, e.g., audio or via typing etc.) and if the human feels like he/she is interacting with another human, then the machine is artificially intelligent”* citado por Joshi (2020).

Apesar desta ser uma versão bastante inovadora para a época, a verdade é que no mundo atual um sistema que consiga simular o comportamento humano não é suficiente e graças ao crescente volume de dados, evolução dos algoritmos e aumento da capacidade de armazenamento e processamento a IA atual vai para além disso. Atualmente, fala-se em Inteligência Artificial se esta for capaz de realizar uma ou mais destas tarefas: perceber linguagem humana, desempenharem tarefas mecânicas envolvendo manobras complexas, resolver problemas computacionais complexos que envolvam um grande conjunto de dados num curto espaço de tempo e devolver respostas humanas entre outras (Joshi, 2020).

2.1.2. *Machine Learning*

Machine learning (ML) é um ramo da Inteligência Artificial que foi definido por Mitchell (1997) como *“study of computer algorithms that allow computer programs to automatically improve through experience”*. No fundo, de acordo com Portugal et al. (2018), *machine learning* usa computadores para simular a aprendizagem humana e permite a esses computadores identificar e adquirir conhecimentos sobre o mundo real, e melhorar a performance de algumas tarefas baseadas nesse novo conhecimento.

Os humanos aprendem naturalmente com a experiência por causa da sua capacidade de raciocinar. Em contraste, os computadores não aprendem raciocinando, mas sim com algoritmos (Portugal et *al.*, 2018).

Um algoritmo de ML usa *inputs* para alcançar uma tarefa desejada sem ser literalmente programado para produzir um resultado específico. Esses algoritmos alteram ou adaptam automaticamente a sua arquitetura por meio da repetição, ou seja, da experiência, para que se tornem cada vez melhores na realização da tarefa desejada. O processo de adaptação é designado de treino, no qual amostras de *inputs* são fornecidas juntamente com os resultados desejados. O algoritmo configura-se então otimamente para que possa não apenas produzir o resultado desejado quando apresentado com os *inputs* de treino, mas pode generalizar para produzir o resultado desejado a partir de dados novos, anteriormente não vistos. Este treino é a parte de "aprendizagem" da *machine learning*. O treino não precisa de limitar-se a uma adaptação inicial durante um intervalo finito, pelo contrário o algoritmo vai aprendendo à medida que processa novos dados e aprende com os seus erros (El Naqa & Murphy, 2015).

Apesar do *machine learning* possuir bastantes utilidades no mundo moderno, tais como detecção de fraude, reconhecimento de objetos, tradução de texto, entre outros, uma das principais é o auxílio na recomendação de conteúdo, seja a nível de *e-commerce*, anúncios, conteúdo a apresentar em redes sociais ou até mesmo a recomendação de filmes.

Antes de abordar de que modo é possível melhorar um sistema de recomendação é importante perceber os diferentes tipos de métodos existentes que permitem atingir uma melhor e mais eficiente recomendação.

2.2. Tipos de *Machine Learning*

De acordo com a literatura analisada (Celebi & Aydin, 2016; Joshi, 2020; Kotsiantis, 2007; Maglogiannis, 2007; Mitchell, 1997; Zhang, 2010), existem vários tipos de algoritmos de aprendizagem, no entanto os três mais relevantes são: *Supervised Learning* (aprendizagem supervisionada), *Unsupervised Learning* (aprendizagem não supervisionada) e o *Reinforcement Learning*.

2.2.1. Aprendizagem Supervisionada

A Aprendizagem Supervisionada é certamente um dos principais e mais comumente usados ramos do ML. De forma simples a aprendizagem supervisionada tem como objetivo a criação de um modelo capaz de entender as relações existentes entre o output que se procura prever com base em dados rotulados e as características do input (Maglogiannis, 2007). Os dados rotulados são pares com uma entrada e um valor de saída também chamado output, resposta ou rótulo.

De acordo com Kotsiantis (2007), a aprendizagem supervisionada acontece quando são fornecidos aos algoritmos dados de treino para efeitos de aprendizagem do algoritmo e respostas corretas. A tarefa do algoritmo de *machine learning* é aprender com base nos dados de treino e aplicar o conhecimento que foi adquirindo usando dados reais.

O primeiro passo para a criação de um modelo de *supervised learning* passa por treinar o algoritmo e isto consiste em juntar os inputs aos outputs corretos e durante esta fase o algoritmo vai procurar identificar padrões e relações entre estes dados. Após esta primeira fase será realizada uma fase de teste em que se procura determinar se fornecendo inputs desconhecidos ao algoritmo este vai ser capaz de classificar corretamente esses dados.

Os algoritmos supervisionados mais conhecidos são a classificação e a regressão. Na classificação o objetivo é prever valores de variáveis qualitativas, ou discretas, por exemplo: {1,0}, {Verdadeiro, Falso}, {compra, não compra}. Na regressão o objetivo é prever valores contínuos, como por exemplo o preço de uma casa.

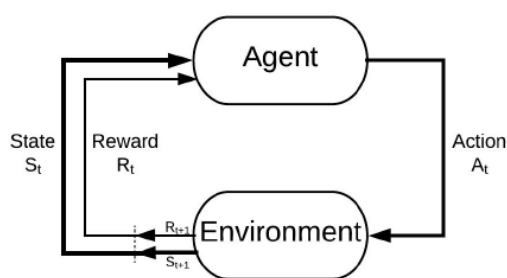
2.2.2. Aprendizagem Não Supervisionada

Contrariamente à aprendizagem apresentada anteriormente, no caso da *unsupervised machine learning* não existem dados rotulados e não existe *feedback* adquirido com o processo (JoshI, 2020). Na Aprendizagem Não Supervisionada, os algoritmos de *machine learning* não passam pela fase de treino. Alguns dados são apresentados ao algoritmo e o algoritmo aprende com esses dados por conta própria (Celebi & Aydin, 2016). Os algoritmos de Aprendizagem Não Supervisionados focam-se principalmente em encontrar padrões ocultos nos dados.

Na realidade, este tipo de aprendizagem centra-se em encontrar padrões e relações nos dados sem que lhe seja ensinado como atingir esse fim (Zhang, 2010). Um dos algoritmos mais importantes do *Unsupervised Learning* é o *Clustering* que tem como foco principal o agrupamento de dados em diferentes classes. Essencialmente, um cluster é um conjunto de dados semelhantes entre si.

2.2.3. Aprendizagem por Reforço/ *Reinforcement Learning*

O *Reinforcement Learning* (RL) foi introduzido por Sutton e Barto (1998) e tem sido aplicado de forma bem-sucedida em vários domínios e aplicações. Ao contrário da aprendizagem supervisionada, que apenas considera dados de treino rotulados, um algoritmo de aprendizagem por reforço explora ativamente o seu ambiente para reunir informações e explora o conhecimento adquirido para tomar decisões ou fazer previsões (Wang et al., 2014). Este método tem como objetivo adquirir observações do ambiente com o objetivo de tomar decisões que consigam maximizar a recompensa e diminuir os riscos, recorrendo para isso à tentativa e erro e utilizando recompensas e penalizações para treinar o comportamento (Wiering & Van Otterlo, 2012). Por outras palavras, o objetivo do *Reinforcement Learning* é maximizar a recompensa de um agente, realizando uma série de ações em resposta a um ambiente dinâmico (Shyalika, 2019). A Figura 1 mostra a interação entre as quatro componentes do *Reinforcement Learning*: agente, ambiente, recompensa e ação.



Fonte: Towards Data Science (2018)

Figura 1: Enquadramento típico de um cenário de *Reinforcement Learning*

De facto, existe um paralelismo muito interessante entre RL e a aprendizagem humana, pois tal como o ser humano adquire competências e as melhora através da

experiência, nomeadamente com o sucesso e o fracasso, também o *Reinforcement Learning* adquire informação das previsões corretas e incorretas sendo ambas fundamentais para a evolução do modelo.

No caso concreto dos sistemas de recomendação o RL tem o potencial de melhorar as recomendações apresentadas a um dado utilizador podendo assim maximizar as vendas, por exemplo.

Muitos sistemas de recomendação seguem uma abordagem estática ao seguir uma estratégia fixa ignorando a constante dinâmica existente entre sistemas de recomendação e os utilizadores. Para além disso, falham também ao focarem-se no *feedback* imediato negligenciando as recompensas de longo prazo (Liu et al., 2018). É exatamente aqui que o RL tem a capacidade de melhorar os sistemas de recomendação pois, não só tem o potencial de melhorar continuamente as estratégias através das interações com os utilizadores (Zhao et al., 2017), como consegue identificar alterações das suas preferências, bem como identificar estratégias que maximizem ganhos futuros.

Existem diversos algoritmos de *Reinforcement Learning*, tais como, *Q-Learning*, State-Action-Reward-State-Action (SARSA), Deep *Q* Network (SQN) e Deep Deterministic Policy Gradient (DDPG).

Enquanto que o *Q-Learning*, assim como será visto na secção 2.2.3.1, é uma técnica *off-policy*, isto é, o *Q-learning* define a recompensa por ter realizado uma ação num estado com base na ação de maior recompensa disponível dentro do novo estado resultante, o *State-Action-Reward-State-Action* (SARSA) é uma técnica *on-policy*, ou seja, realiza uma segunda ação a partir do segundo estado de acordo com o que aprendeu e define a recompensa para o primeiro par de estado-ação com base no que acontece depois (Afsar et al., 2021). A diferença entre os dois algoritmos, prende-se com o facto de o SARSA nem sempre escolherá a ação disponível mais promissora.

Mnih et al. (2015) introduziram *Deep Q-Network* (DQN) e iniciaram o campo de *deep RL*. Desenvolveram um novo agente capaz de combinar a aprendizagem por reforço com as redes neurais profundas, uma classe de rede neural artificial. Estas redes neurais profundas são capazes de aprender conceitos como categorias de objetos diretamente através de dados sensoriais brutos.

Porém, o DQN é destinado a problemas com algumas ações possíveis e, portanto, não são apropriados para ações espaciais contínuas (Casas, 2017). O algoritmo *Deep Deterministic Policy Gradient* (DDPG), desenvolvido por Lillicrap et al. (2015), colmata esta limitação, adaptando o *Deep Q-Learning* a domínios de ação contínuos.

O *Deep Learning* tem prevalecido na aprendizagem por reforço nos últimos anos, em jogos, robótica, entre outros, uma vez que tem beneficiado de *big data*, computação poderosa, novas técnicas algorítmicas e forte suporte financeiro (Li, 2017). No entanto, sendo uma classe específica de *machine learning*, sofre também de algumas limitações. Por exemplo, a maioria das abordagens de *Deep Learning* é baseada, essencialmente, em modelos *black box* (caixa negra), ou seja, de baixa interpretabilidade, sem princípios científicos claros e suficientes, na qual os investigadores não podem inspecionar e controlar a forma como o algoritmo está a funcionar (Li, 2017).

Assim sendo, nesta fase interessa descrever o *Q-Learning* e os processos de decisão de *Markov* (*Markov Decision Process*), pois o sistema foi desenvolvido com base nestes métodos.

2.2.3.1. *Q-Learning*

De acordo com Watkins (1989) o *Q-Learning* é um algoritmo de *Reinforcement Learning* que fornece aos agentes a capacidade de aprender a agir de forma otimizada em domínios *markovianos* (secção 2.2.3.2), experimentando as consequências das ações, sem exigir que eles construam mapas dos domínios. *Q-Learning* é um dos algoritmos mais usados devido à sua habilidade para comparar a utilidade esperada das ações disponíveis sem exigir um modelo do ambiente. Graças a este algoritmo, é possível encontrar uma ação ótima para qualquer estado acabado do *Markov Decision Process* (MDP) (Ciaburri, 2019).

O *Reinforcement Learning* envolve um agente, um conjunto de estados e um conjunto de ações por estado. O *Q-learning* atua através de uma matriz Q , na qual as linhas correspondem ao conjunto de estados possíveis e as colunas correspondem ao conjunto de ações disponíveis. Ao praticar determinada ação, o agente transita de estado para estado e a execução de uma ação num estado específico proporciona ao agente uma recompensa.

A fórmula de atualização dos valores da matriz Q utiliza a equação de Bellman (Shyalika, 2019), a qual se apoia em dois inputs: estado (s) e ação(a):

$$Q^{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1} + a) - Q(s_t, a_t)]$$

onde s_t corresponde ao estado e s_{t+1} representa o estado seguinte; a_t corresponde à ação atual e a_t representa a ação seguinte; α é a taxa de aprendizagem, sendo que esta é definida entre 0 e 1; r_t é a recompensa recebida quando o agente se move de s_t para s_{t+1} ; e γ é o fator de desconto, também ele definido entre 0 e 1.

O fator de desconto γ determina a importância de futuras recompensas. Um fator de desconto menor dá menos importância a recompensas futuras, focando-se no presente, enquanto um fator de desconto maior coloca mais ênfase nas recompensas futuras.

A taxa de aprendizagem α determina em que medida as informações adquiridas substituem as antigas. Assim, um fator mais baixo significa que o agente aprende mais lentamente, enquanto um fator igual mais próximo de um significa que o agente aprende mais rápido.

A cada momento t que um agente seleciona uma ação a_t , obtém uma recompensa r_t e entra num novo estado s_{t+1} e, portanto, Q é atualizado. Este processo é repetido várias vezes até que a aprendizagem seja interrompida. Desta forma, a Matriz Q é atualizada e a função de valor Q é maximizada.

Importa referir que o *Q-learning* é um algoritmo iterativo e, como tal, assume-se uma condição inicial antes da primeira atualização.

2.2.3.2. Markov Decision Process

O *Markov Decision Process* utiliza estados, ações, transições entre estados e uma definição de função de recompensa (Wiering & Van Otterlo, 2012). O agente é aquele aprende e que toma as decisões. Este interage com tudo aquilo que o rodeia, ou seja, o ambiente. O agente escolhe as ações seguintes e, com base nisso, o ambiente responde a essas ações apresentando novas situações ao agente. Tendo em conta as decisões do agente, são-lhe atribuídas recompensas que o agente tem por objetivo maximizar ao longo do tempo por meio da sua escolha de ações (Sutton & Barto, 1998).

2.3. Sistemas de Recomendação

Nos últimos anos, a Internet tem-se tornado num sistema global de elevada importância na vida de qualquer ser-humano e, como consequência as pessoas veem-se confrontadas com um elevado volume de informação, proveniente de diversas fontes. Como

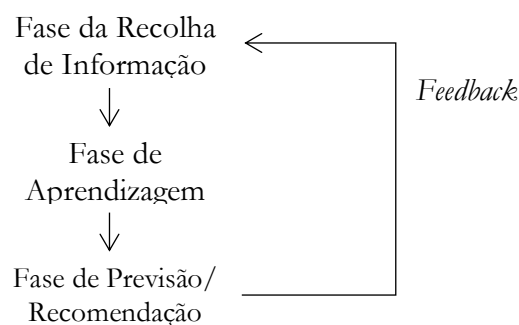
tal, as pessoas têm dificuldade em filtrar os fragmentos de informação e selecionar os dados que melhor se adequam às suas necessidades.

Os sistemas de recomendação têm vindo a tornar-se uma parte fulcral na maioria das plataformas *online*, nomeadamente no comércio eletrónico e na mitigação do problema da sobrecarga de informações por meio da sugestão de itens ou serviços personalizados ao utilizador.

Segundo Rashid et *al.* (2002), um sistema de recomendação é caracterizado como um sistema que auxilia as pessoas na tomada de decisão em espaços de informação complexos. Os sistemas de recomendação são sistemas de filtragem de informações que lidam com o problema de sobrecarga de informações (Konstan & Riedl, 2012) filtrando fragmentos de informações vitais de uma grande quantidade de informações geradas dinamicamente de acordo com as preferências, interesses ou comportamento observado do utilizador sobre o item (Pan & Li, 2010). O sistema de recomendação tem a capacidade de prever se um determinado utilizador preferiria um item ou não com base no seu perfil.

2.3.1. Fases do Processo de recomendação

Existem três fases principais no processo de recomendação: a fase de recolha de informações, a fase de aprendizagem e a fase de previsão/recomendação (Isinkaye et *al.*, 2015). A Figura 2 mostra as fases do sistema de recomendação.



Fonte: Isinkaye et *al.*, 2015, adaptado pelo autor

Figura 2: Fase do Processo de Recomendação

Fase da Recolha de Informação

Nesta fase, tal com o nome indica, é feita a recolha de informações relevantes sobre os utilizadores, tais como as características do utilizador, os seus comportamentos e os conteúdos das fontes que o utilizador pesquisa, de modo a gerar um perfil ou modelo do utilizador para realizar a previsão ou fazer uma recomendação. A recomendação só será confiável se o perfil do utilizador for bem construído. Deste modo, o sistema precisa de obter o máximo de informação possível sobre o utilizador para fazer uma recomendação adequada. Os sistemas de recomendação dependem de diferentes tipos de *inputs*: *feedback* explícito, *feedback* implícito e *feedback* híbrido. No *feedback* explícito o *input* é o interesse do utilizador no item. O sistema normalmente solicita que o utilizador, por meio da interface do sistema, forneça classificações aos itens de modo a construir e a melhorar o seu modelo. Ainda que o *feedback* explícito seja o que exige maior esforço por parte do utilizador, é o que fornece dados mais fiáveis. Já o *feedback* implícito conclui as preferências do utilizador de forma indireta, analisando o comportamento do mesmo. O sistema infere automaticamente as preferências do utilizador monitorizando diferentes ações dos mesmos, tais como o histórico de compras, o histórico de navegação, o tempo gasto em algumas páginas da web, etc. O *feedback* implícito reduz a carga de esforço sobre os utilizadores ao inferir as preferências do utilizador a partir do seu comportamento com o sistema. Existe ainda o chamado *feedback* híbrido, que é obtido através da combinação dos dois anteriores, o que permite minimizar os pontos fracos de ambos e obter assim um sistema com melhor desempenho.

Fase de Aprendizagem

Nesta fase são aplicados algoritmos de aprendizagem nos dados do utilizador que foram obtidos a partir dos *feedbacks* na fase de recolha de informação (Isinkaye et al., 2015).

Fase de Previsão/ Recomendação

Ao analisar os padrões obtidos na fase de aprendizagem, esta fase fornece as recomendações ou previsões dos itens que o utilizador poderá preferir. Os dados analisados na fase de aprendizagem fornecem certos padrões que, em seguida, são utilizados para prever o percurso do utilizador ou os seus interesses futuros (Kumar & Kumar, 2019).

2.4. Técnicas de filtragem de recomendação

Recentemente, várias abordagens para a construção de sistemas de recomendação foram desenvolvidas recorrendo a métodos de *collaborative filtering*, *content-based filtering* ou *hybrid filtering* (Liu et al., 2018).

Naturalmente existem outros sistemas tais como os *knowledge-based*, onde as recomendações são efetuadas com base em requisitos específicos do utilizador ou os *Demographics-Based* em que, tal como o nome indica, a informação demográfica do utilizador é usada como fator chave para identificar as preferências do utilizador (Isinkaye et al., 2015).

No entanto, será dada ênfase aos três primeiros, pois são os métodos mais comuns e onde estão focados os desenvolvimentos mais recentes.

2.4.1. Collaborative Filtering

A técnica de filtragem colaborativa recomenda um item a um utilizador específico com base na avaliação/opinião de outro utilizador (Kumar & Thakur, 2018).

A filtragem colaborativa é uma técnica de previsão de domínio independente para conteúdo que não pode ser descrito de forma fácil e adequada por meta-dados, tais como filmes e música (Isinkaye et al., 2015). Dado um domínio de escolhas (itens), os utilizadores podem expressar as suas opiniões (avaliações) acerca dos itens que experimentaram anteriormente (Rashid et al., 2002). As recomendações são feitas de acordo com o perfil dos outros utilizadores e as suas similaridades com as preferências e interesses do utilizador em análise. Por exemplo, numa aplicação de recomendação de filmes, a técnica de filtragem colaborativa tenta encontrar outros utilizadores com interesses e preferências semelhantes às do utilizador em análise, de modo a recomendar os filmes que eles mais gostam (Sharma & Gera, 2013).

2.4.1.1. Item-based

As técnicas baseadas em itens analisam primeiro a matriz item-utilizador para identificar as relações entre os diferentes itens e, em seguida, usam as semelhanças para calcular indiretamente as recomendações para os utilizadores (Sarwar et al., 2001)

2.4.1.2. *User-based*

A recomendação baseada no utilizador consiste em seleccionar um conjunto de utilizadores semelhante ao utilizador em análise, analisar os seus gostos e, depois, fazer as sugestões de itens ao utilizador com base nessas semelhanças (Koochi & Kiano, 2016).

2.4.2. *Content-based Filtering*

A técnica baseada em conteúdo é um algoritmo de domínio dependente e enfatiza mais a análise dos atributos dos itens para gerar previsões (Isinkaye et al., 2015). A recomendação é feita baseando-se no perfil do utilizador e utilizando os atributos dos itens que o utilizador avaliou no passado. A ideia geral por trás desta técnica é que se o utilizador gostou de um item específico, então também poderá gostar de um item semelhante (Patel et al., 2017). A técnica de filtragem baseada em conteúdo não precisa do perfil de outros utilizadores, uma vez que não influenciam a recomendação.

2.4.3. *Hybrid Filtering*

A técnica de filtragem híbrida combina diferentes técnicas de recomendação com o objetivo de obter uma melhor otimização do sistema de modo a evitar algumas limitações e problemas de sistemas de recomendação puros (Adomavicius & Zhang, 2012). A ideia central para a utilização da técnica de filtragem híbrida é o facto de que a combinação de diferentes algoritmos fornecerá recomendações mais precisas e eficazes do que um único algoritmo, uma vez que as desvantagens de um algoritmo podem ser ultrapassadas por outro algoritmo.

2.5. Trabalho Relacionado

Os sistemas de recomendação têm sido alvo de grandes pesquisas e, por conseguinte, de descobertas, no que respeita a novas técnicas, e de melhoramento dos sistemas (Bobadilla et al., 2013). Os sistemas de recomendação têm-se tornado cada vez mais populares nos últimos anos e têm sido utilizados numa variedade de domínios, incluindo filmes, música, livros, consultas de pesquisa e redes sociais (Resnick & Varian, 1997). Inúmeras técnicas foram propostas para lidar com o problema de recomendação, sendo que as técnicas

tradicionais incluem filtragem colaborativa, filtragem baseada em conteúdo e métodos híbridos (Afsar et al., 2021). Recentemente, o *Deep Learning* também ganhou popularidade no campo do *Recommendation System* (RS) devido às suas capacidades de encontrar relacionamentos complexos e não lineares entre utilizadores e itens e também devido ao seu alto desempenho na recomendação (Afsar et al., 2021). Liu et al. (2018) apontaram as limitações das técnicas tradicionais, como por exemplo da filtragem colaborativa, filtragem baseada em conteúdo e propuseram uma nova técnica de recomendação baseada no *Deep Reinforcement Learning*. A técnica de *Deep Reinforcement Learning* trata a recomendação como um procedimento de tomada de decisão sequencial e adota um esquema de aprendizagem por reforço para modelar as interações entre os utilizadores e os sistemas de recomendação, que podem considerar tanto a adaptação dinâmica quando as recompensas de longo prazo. Também Zhao et al. (2018) perceberam as limitações das técnicas tradicionais e propuseram um novo sistema de recomendação com a capacidade de melhorar continuamente suas estratégias durante as interações com os utilizadores. Modelaram as interações sequenciais entre os utilizadores e um sistema de recomendação como um MDP e aproveitaram o *Reinforcement Learning* para aprender automaticamente as estratégias ideais por meio da recomendação de itens por tentativa e erro e por via do recebimento de reforços desses itens através do *feedback* dos utilizadores. Mooney & Roy (2000) desenvolveram um sistema de recomendação de livros baseado no conteúdo que utiliza extração de informações e um algoritmo de *machine learning* para a categorização de texto. Os resultados experimentais iniciais demonstram que esta abordagem pode produzir recomendações precisas.

No que respeita aos sistemas de recomendação aliados a filmes, existem também já algumas pesquisas efetuadas (Halder et al., 2012). Halder et al. (2012) propuseram um SR que tem a capacidade de lidar com novos utilizadores e itens. Em primeiro lugar, é criado um conjunto de filmes por géneros, característica essa que permite criar o sistema de recomendação baseada no conteúdo. Para demonstrar a eficácia do modelo proposto usaram um conjunto de dados da *MovieLens* (Movilens, 2021).

2.6. Análise de resultados do algoritmo *Q-Learning*

Uma das melhores formas de perceber a evolução do *Q-Learning* ao longo do tempo é recorrendo à análise da recompensa cumulativa (Sidana, 2018), pois o objetivo do algoritmo em cada iteração é maximizar esse valor.

O arrependimento é uma outra forma de analisar a evolução do algoritmo. Uma ação da qual haja arrependimento significa que é aquela que deveria ter sido praticada. O arrependimento é então definido como a diferença entre o *payoff* incorrido pelo algoritmo e o *payoff* ótimo (Yang & Toni, 2018). Quer isto dizer que o payoff ótimo terá que ser maior ou igual ao payoff incorrido pelo algoritmo, de forma a obter arrependimento positivo ou nulo para analisar a evolução do algoritmo.

2.7. Avaliação de Sistemas de Recomendação

De acordo com Herlocker et al. (2004), avaliar os sistemas de recomendação e os seus algoritmos é inerentemente difícil por, essencialmente, três motivos. Primeiro, diferentes algoritmos podem ser melhores ou piores consoante diferentes conjuntos de dados. Segundo, os objetivos para os quais uma avaliação é realizada podem ser diferentes. E terceiro, há um desafio significativo em decidir qual a combinação de medidas a usar na avaliação comparativa.

Sempre que é desenvolvido um sistema de recomendação é útil e necessário avaliar se o sistema funciona bem. Os sistemas de recomendação podem ser avaliados por meio de várias métricas e testes *offline*. As métricas de avaliação são essenciais para avaliar a qualidade e desempenho do sistema de recomendação (Ziegler et al., 2005). Sempre que um sistema de recomendação é avaliado, a métrica usada baseia-se em certas premissas (Del Olmo & Gaudioso, 2008).

2.7.1. Avaliação *Offline*

Grande parte do trabalho na avaliação de algoritmos tem-se concentrado na análise *offline* de precisão preditiva (Herlocker et al., 2004).

No que respeita à avaliação dos sistemas de recomendação é comum usar a avaliação *offline* para pré-testar algoritmos de modo a entender o seu comportamento antes do teste efetuado pelo utilizador, dado que este último pode ser dispendioso. A avaliação *offline* também é benéfica para realizar comparações diretas e objetivas de diferentes algoritmos.

Esta avaliação é baseada essencialmente na configuração de treino e teste de ML. Inicia-se com um conjunto de dados, normalmente classificações atribuídas pelos utilizadores ou dados históricos de utilizadores. Os utilizadores são então divididos em dois grupos: o conjunto de treino e o conjunto de teste. Um modelo de recomendação é construído em relação ao conjunto de treino.

Contudo, as análises *offline* têm duas limitações (Herlocker et al., 2004). A primeira prende-se com o facto de a dispersão natural dos conjuntos de dados de classificações atribuídas pelos utilizadores limitar o conjunto de itens que podem ser avaliados, ou seja, não podemos avaliar a adequação da recomendação de um item a um utilizador se o utilizador não tiver atribuído uma classificação àquele item no conjunto de dados. Em segundo lugar, existe uma limitação à avaliação objetiva dos resultados da previsão, uma vez que nenhuma análise *offline* pode determinar se os utilizadores preferirão um determinado sistema, seja por causa das suas previsões ou por causa de outros critérios menos objetivo, como a aparência da interface.

2.7.2. Prediction Accuracy

As métricas de precisão preditiva medem o quão precisa foi a previsão das classificações em relação às verdadeiras classificações feitas pelos utilizadores (Ziegler et al., 2005).

2.7.2.1. MAE

Um método simples de medir a qualidade da recomendação é medir o erro absoluto médio (MAE) (Herlocker et al., 2002, 2004). O erro médio absoluto é a média da diferença entre a classificação prevista pelo sistema de recomendação e a classificação real fornecida pelo utilizador. Seja p a classificação prevista e r a classificação real, a MAE é dada pela seguinte fórmula:

$$MAE = \frac{\sum_{i=1}^n |p_i - r_i|}{n}$$

Um resultado zero significa que não existiu nenhuma diferença entre o resultado previsto e o resultado obtido pelo utilizador, logo quanto menor o valor do MAE melhor é a precisão da previsão.

2.7.2.2. MSE

O erro médio quadrático (MSE) é semelhante ao erro médio absoluto, com a diferença de que em vez de usar o erro absoluto para cancelar o sinal negativo, eleva-se ao quadrado. Quanto menor o MSE, mais próximas as previsões dos valores reais e melhor será o desempenho do sistema (Mehrijoo et al., 2019). Assumindo as mesmas variáveis anteriores, MSE é dado por:

$$MSE = \frac{\sum_{i=1}^n (p_i - r_i)^2}{n}$$

2.7.2.3. RMSE

De acordo com Gunawardana e Shani (2009), a raiz do erro quadrático médio (RMSE) é uma métrica popular para avaliar um algoritmo. Se $p_{i,j}$, é a classificação prevista para o utilizador i acerca do item j , e $r_{i,j}$ é a classificação real, e $K = \{(i, j)\}$ é o conjunto de classificações ocultas de itens do utilizador, então o RMSE é definido como:

$$\sqrt{\frac{\sum_{i=1}^n (p_{i,j} - r_{i,j})^2}{n}}$$

No fundo trata-se da raiz quadrada do MSE.

O RMSE é adequado para a tarefa de previsão, porque mede imprecisões em todas as classificações, sejam elas negativas ou positivas, além de que tem o efeito de colocar maior ênfase em erros grandes (Herlocker et al., 2004). Também aqui, um baixo valor indica que o sistema de recomendação é mais adequado (Mehrijoo et al., 2019).

2.7.3. *Decision Support Metrics*

As métricas de apoio à decisão avaliam se o sistema de recomendação é capaz de fazer recomendações relevantes. Estas métricas comparam a lista de itens recomendados pelo sistema de recomendação com as verdadeiras preferências do utilizador (Sidana, 2018). Ou seja, estas não consideram as previsões e seus desvios das classificações reais (Ziegler et al., 2005).

	Relevante	Não Relevante
Recomendado	Verdadeiro positivo (VP)	Falso positivo (FP)
Não Recomendado	Falso negativo (FN)	Verdadeiro negativo (VN)

Tabela 1: Classificação dos possíveis resultados da recomendação de um item a um utilizador

A Precisão e a *Recall* são duas métricas de qualidade de recomendação importantes e comumente usadas (Mehrjoo et al., 2019). A Precisão mede a fração de itens relevantes para o utilizador recomendados na lista. A *Recall* mede qual é fração dos itens relevantes que foram selecionados do conjunto de recomendações. Seja $L(s)$ a lista de recomendações e R os itens relevantes para um utilizador, as duas métricas podem ser definidas (Sidana, 2018):

$$Precision = \frac{|R \cap L(s)|}{|L(s)|} \equiv \frac{VP}{VP + FP}$$

$$Recall = \frac{|R \cap L(s)|}{|R|} \equiv \frac{VP}{VP + FN}$$

Contudo, estas duas métricas podem entrar em conflito. Se se aumentar o tamanho da lista de recomendações, por um lado a *Recall* aumentará, mas por outro a Precisão irá diminuir. Este conflito pode ser resolvido usando a métrica *F1 Score*, que é a média harmônica de Precisão e *Recall* e é dada por:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

2.7.4. Coverage

Embora as métricas de precisão sejam importantes, existem características de satisfação do utilizador que estas não conseguem capturar (Ziegler et al., 2005).

Coverage ou métrica de Cobertura é uma das métricas de não-precisão mais utilizadas e mede a percentagem de dados que o modelo utiliza de entre o total de dados que podiam ter sido utilizados (Ziegler et al., 2005).

Esta métrica tem uma limitação importante. Para um sistema de recomendação altamente impreciso, isto é, quando as recomendações são feitas aleatoriamente, a cobertura pode ser muito alta, enquanto a precisão da recomendação é extremamente baixa. (Liu et al., 2019).

2.7.5. Popularity

Um obstáculo à eficácia dos sistemas de recomendação está no facto de estes sofrerem de *popularity bias* nos seus resultados, isto é, os itens populares, ou seja, os itens com mais avaliações, são recomendados com maior frequência do que os menos populares (Park & Tuzhilin, 2008). Os itens menos populares, ou seja, que tem menos avaliações ou cliques, seja porque são novos ou simplesmente impopulares são então representados por uma cauda longa e, por essa razão a popularity bias é também conhecida como o problema da cauda longa (*long tail problem*) (Sidana, 2018).

No entanto, os itens menos populares e de cauda longa são precisamente aqueles que costumam ser recomendações desejáveis (Abdollahpouri et al., 2017). Podemos então aqui verificar o Princípio de Pareto cujo princípio advoga que 80% das consequências de inúmeros fenómenos provêm de 20% das causas.

2.7.6. Diversity Measures

A Diversidade é a inclusão de diferentes tipos de itens na recomendação para o utilizador, diferentes das suas preferências anteriores (Yadav et al., 2011). Existem, portanto, métricas que têm como objetivo capturar a diversidade entre os itens de uma lista.

2.7.6.1. ILS

Uma das métricas para calcular a diversidade é a similaridade intralista (ILS). Para calcular este critério, é necessária uma definição de similaridade entre os itens (Mehrijo et al., 2019). Esta métrica usa características dos itens recomendados para calcular a semelhança, tais como género, autor, entre outras (Ziegler et al., 2005). Quanto maior a similaridade entre os itens, menor será a diversidade intralista.

A similaridade intralista é obtida a partir da seguinte fórmula (Mehrijo et al., 2019):

$$ILS = \frac{\sum_{k \in L_i} \sum_{j \in L_j, j \neq k} c(k, j)}{2}$$

onde $c(k, j)$ é o grau de similaridade entre dois itens da lista com base numa característica específica predefinida que possui um valor entre 0 e 1. Neste caso em concreto a similaridade foi calculada recorrendo ao *cosine similarity*.

A diversidade intralista é calculada da seguinte forma:

$$D = 1 - \left(\frac{ILS}{N}\right)$$

2.7.6.2. Category Diversity

Um dos problemas das métricas de precisão prende-se com a sua incapacidade de capturar os aspetos mais amplos da satisfação do utilizador, que vão além da precisão da lista de itens apresentada ao utilizador. A diversificação por categoria passa então por tornar as listas de recomendações mais diversificadas e, portanto, mais úteis (Ziegler et al., 2005).

2.7.7. Outros métodos de avaliação

2.7.7.1. Ranking Based Methods

Nos sistemas de recomendação, geralmente é apresentada ao utilizador uma lista de recomendações contendo os itens principais e, idealmente, por ordem decrescente de preferência. As recomendações podem, portanto, ser estudadas como um problema de *ranking*, usando-se medidas de *ranking* para avaliar quão boa é a avaliação (Sidana, 2018).

2.7.7.2. Temporal Evaluation

É importante considerar que as preferências dos utilizadores se alteram ao longo do tempo. Hoje um utilizador pode avaliar um item com melhor classificação do que daqui a um ano, por exemplo. Por essa razão deve considerar-se a dimensão temporal ao avaliar um sistema de recomendação.

2.7.7.3. Percieved Quality

A qualidade percebida diz respeito à avaliação dos utilizadores acerca da recomendação feita pelo sistema. Esta também pode ser uma métrica útil, no sentido em que

permite definir o que torna uma recomendação “boa”. No entanto, exige esforço por parte dos utilizadores e, por isso, pode ser difícil conseguir a participação dos mesmos nessa avaliação.

Capítulo 3

Metodologia e Dados: Descrição do Sistema

Este capítulo tem como principal objetivo referir os métodos usados neste estudo que permitiram responder ao problema em análise, e será dividido em três partes principais, começando com a definição do problema e a respetiva solução proposta, onde será realizada uma apresentação da metodologia e todas as suas características e especificidades.

Numa segunda parte vai ser apresentada a arquitetura do sistema onde serão descritas todas as fases usadas na construção do sistema de recomendação e vão também ser referidos todos os softwares e linguagens de programação usadas.

No anexo 1, de modo a facilitar a compreensão de todo o sistema de recomendação e de todas as ferramentas usadas, é possível encontrar um exemplo do processo de recomendação que inclui todas as fases envolvidas.

Por último, neste capítulo, vai ser apresentado um sistema de recomendação baseado em CF que será usado na fase de teste, isto é, nos últimos 30 utilizadores, e que tem como principal objetivo ser usado como termo de comparação no próximo capítulo, permitindo assim retirar melhores conclusões da análise efetuada ao sistema de recomendação em estudo. No anexo 2, é possível encontrar um exemplo de iterações de um utilizador com o sistema de recomendação referido anteriormente.

3.1. Definição do Problema

Como consequência do aumento da presença de consumidores nos meios digitais, surge um crescente aumento da informação disponível e, tendo em conta o escasso tempo de que o consumidor dispõe, torna-se fundamental que lhe sejam apresentados produtos/serviços ideais de forma personalizada de acordo com o seu perfil.

Tendo isto em conta, o problema em questão centra-se na criação de um sistema de recomendação capaz de filtrar esse excesso de informação e apresentar ao consumidor apenas os produtos/serviços que ele provavelmente está interessado em adquirir. E apesar

dos sistemas de recomendação serem algo comum no meio digital, muito frequentemente os sistemas tradicionalmente usados possuem uma falha ao considerar o consumidor como alguém que possui preferências estáticas, algo que certamente não se verifica no mundo real.

Posto isto, surge a oportunidade de usar novas técnicas que permitam efetuar recomendações mais personalizadas para cada um dos utilizadores de acordo com as suas preferências, pois cada vez mais o utilizador não procura simplesmente opções, mas sim sugestões adequadas ao seu perfil.

Para além disso e como forma de comprovar a usabilidade em ambiente real do sistema de recomendação é importante testá-lo junto de utilizadores reais num ambiente controlado relacionado com comércio eletrónico na área do *video on demand*.

Por último, e como forma de fortalecer as conclusões retiradas e justificar a melhoria gerada pelo sistema de recomendação desenvolvido é importante a realização de uma comparação dos resultados obtidos face a outro sistema de recomendação amplamente usado tanto a nível académico como a nível comercial.

3.2. Solução Proposta

Após a identificação do problema em estudo foi necessária a criação de um modelo que tivesse em vista a sua solução. Posto isto, numa primeira fase foram analisados vários tipos de métodos que provassem ter algum contributo para o estudo dos sistemas de recomendação.

Após esta análise inicial foi escolhido o *Reinforcement Learning*, mais propriamente o algoritmo *Q-Learning* baseado em cadeias de *Markov* tabulares, uma vez que tem a capacidade de detetar estratégias otimizadas de recomendação, usando para isso uma função *Q*. Esta função é maximizada, e está dependente de recompensas e penalizações. Uma outra vantagem face a sistemas tradicionais é a capacidade de captar recompensas no longo prazo de forma controlada e interpretável, tornando o sistema muito mais interessante para o utilizador, na medida em que o sistema evolui a sua recomendação de acordo com a evolução do perfil do utilizador.

Depois de uma longa pesquisa foi concluído que a melhor forma de desenvolver o algoritmo seria criando um ambiente controlado, realizando uma prova de conceito face à

utilidade que um sistema de recomendação baseado num algoritmo de *Q-Learning* tem para o utilizador. Trata-se de um algoritmo de RL baseado no método *Markov Chain* tabular, que é um método adequado para resolver problemas de RL em que as funções de valor podem ser representadas em tabelas, uma vez que o tamanho das ações e do espaço de estados são pequenos (Afsar et al., 2021). A este sistema desenvolvido deu-se o nome de **RQ-Markov - Recommender System based on a Q-learning Tabular Markov Chain**.

Não se optou por um método baseado em *deep reinforcement learning*, pois as abordagens de *deep learning* são essencialmente baseadas em modelos *black box* (caixa negra), ou seja, de baixa interpretabilidade, nos quais os investigadores não podem realmente inspecionar a forma como o algoritmo está a realizar e controlar o seu funcionamento (Li, 2017).

Relativamente ao mercado ao qual se irá aplicar o sistema de recomendação desenvolvido, numa primeira abordagem o objetivo seria realizar o estudo com produtos do retalho, nomeadamente com produtos de retalho da moda. No entanto, devido à complexidade que existe em definir categorias neste tipo de setor urgiu a necessidade de realizar o estudo para um tipo de mercado para o qual fosse não só simples a criação de uma base de dados (BD) com itens, como também a criação de categorias adequadas. Tendo em conta esta premissa foi concluído que o mercado do *video on demand* seria a escolha mais indicada para a realização deste estudo, uma vez que permite retirar conclusões que podem facilmente ser transpostas para outro tipo de mercados, pois o sistema de recomendação pode ser adaptado a diversos ambientes.

Tendo em conta que o objetivo deste estudo é analisar o sistema de recomendação, foi usada a técnica de *backpropagation*, que é descrita como possuindo a capacidade de direcionar o vetor de pesquisa para qualquer direção. (Ackley & Littman, 1989). Quer isto dizer, neste caso concreto, que através de *backpropagation* foi possível calcular a matriz Q e todos os seus valores apenas após a obtenção de todas recompensas/penalizações.

Esta técnica, apesar de ter algumas limitações a nível prático, provou ser a melhor opção para a realização de uma prova de conceito. Uma das limitações que pode ser imputada a esta técnica é a necessidade de armazenamento de várias matrizes Q .

De facto, à primeira vista poderia até parecer que este método seria semelhante a um algoritmo de min-max, no entanto a abordagem adotada tem um fator muito importante de distinção, o facto de ao serem realizados testes com utilizadores reais permitir a criação de

uma base de dados mais reduzida face ao cenário em que todas as matrizes possíveis fossem armazenadas.

Este tópico leva ao próximo ponto-chave do sistema de recomendação desenvolvido neste estudo, que se centra no uso de dados reais tanto para a fase de pré-teste como para a fase de teste. Face aos estudos analisados, este ponto acaba por ser um fator de distinção muito importante, na medida em que não são usados *datasets* com o *MovieLens* (Movilens, 2021), que é amplamente usado na bibliografia analisada (Liu et al., 2019; Rashid et al., 2002). A vantagem proveniente desta metodologia é que para além do controlo total do modelo existe também uma maior confiança nos dados obtidos. Foi procurado ir mais além e não usar apenas um ambiente simulado, mas sim criar um ambiente onde existiu iteração real desde o momento zero com utilizadores reais.

No entanto, é natural que, de modo a interagir com os utilizadores de pré-teste, fossem usadas matrizes pré-existent. Contudo, este tópico vai ser abordado de forma mais aprofundada nas próximas secções.

Após esta pesquisa e definição de caminho a seguir, surgiu a necessidade da criação do algoritmo e de todo o sistema que permitisse a realização de testes a utilizadores. Inicialmente, foi usado o *Excel* (Microsoft, 2021) para definir de que forma seria calculado o *Q-Learning*, bem como todos os seus parâmetros.

Posteriormente, foi utilizada linguagem de programação R auxiliada pelo software *R Studio* (R Core Team, 2021) como forma de automatizar o algoritmo, uma vez que em *Excel* tal tarefa seria dificilmente exequível. Por último, provou ser importante a criação de uma interface onde seriam apresentados ao utilizador os diferentes filmes em cada *slate* e onde seria obtido o *feedback* por parte do mesmo. Entenda-se por *feedback* o interesse ou não nos filmes apresentados ao utilizador e por *slate* o conjunto de filmes que são apresentados em cada iteração com o utilizador.

3.3. Arquitetura do Sistema

3.3.1. *Q-Learning*

O primeiro passo na criação de um sistema de recomendação é naturalmente a criação do algoritmo. Nesta dissertação, tal como já foi referido, o algoritmo a ser usado será o *Q-Learning*.

Tal como seria de esperar, também aqui o *Q-Learning* vai atuar através de uma matriz na qual as linhas correspondem ao conjunto de estados possíveis, que neste caso diz respeito às seis diferentes categorias de filmes disponíveis, e as colunas dizem respeito ao conjunto de ações disponíveis, que neste caso são as seis diferentes *slates*. Ao ser realizada uma determinada ação, o agente vai transitar de categoria em categoria (estados) e a execução de uma ação numa categoria específica vai proporcionar ao agente uma recompensa ou penalização.

No modelo desenvolvido, as recompensas, r_t , correspondem ao *feedback* positivo e são representadas pelo valor 1. O valor -1 representa uma penalização e o consequente desinteresse do utilizador pelo filme apresentado. O 0 refere-se à situação em que em determinado estado não pode ser tomada nenhuma ação.

$$Q^{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1} + a) - Q(s_t, a_t)]$$

Na equação anterior pode observar-se a forma de atualização dos valores da matriz Q , onde s_t corresponde ao estado e s_{t+1} representa o estado seguinte; a_t corresponde à ação atual e a_t representa a ação seguinte; α é a taxa de aprendizagem, sendo que esta é definida entre 0 e 1; r_t é a recompensa recebida quando o agente se move de s_t para s_{t+1} ; e γ é o fator de desconto, também ele definido entre 0 e 1.

Relativamente à taxa de aprendizagem esta determina em que medida as informações adquiridas substituem as antigas. Posto isto, o valor considerado foi de 0,7, uma vez que desta forma é dado um peso maior a informações mais recentes, mas também é um valor adequado o suficiente para a consideração de informações anteriores.

Por último, o fator de desconto, γ , neste estudo é igual a 1, assumindo desta forma uma grande importância de futuras recompensas, pois foi assumido que um sistema de recompensa será tanto melhor quanto a sua capacidade de prever recompensas futuras, uma vez que um sistema que considere apenas ganhos no curto prazo não traria nenhum tipo de vantagem face a sistemas já existentes.

3.3.2. Desenvolvimento do Sistema de recomendação

Após a definição dos diferentes parâmetros do algoritmo foi criado um modelo auxiliar que permite obter os valores necessários para atingir o valor de Q final (Q^{new}).

		Ações					
		Slate 1	Slate 2	Slate 3	Slate 4	Slate 5	Slate 6
Estados	Categoria 1	1	0	1	0	1	0
	Categoria 2	1	0	1	0	1	0
	Categoria 3	1	0	1	0	1	0
	Categoria 4	0	1	0	1	0	1
	Categoria 5	0	1	0	1	0	1
	Categoria 6	0	1	0	1	0	1

Recompensa	Recompensa corrigida
3	7
3	7
3	7
3	7
3	7
3	7
3	7

R	42
Y	1
Alpha	0,7

Q final	3,458881	6,423635	9,811927	14,289312	20,529991	29,378567	41,990814
R	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000	42,000000
Q Old	0,000000	4,941258	9,176622	14,017040	20,413302	29,328558	41,969380
max Q	4,941258	7,058940	10,084200	14,406000	20,580000	29,400000	0,000000

Fonte: Autor

Figura 3: Modelo Q -Learning em Excel

Na Figura 3 pode observar-se um exemplo ilustrativo de uma iteração com o modelo desenvolvido. Este modelo foi criado em *Excel* e tem como objetivo apenas demonstrar de forma visual o funcionamento do algoritmo e representar todos os passos envolvidos para o cálculo do Q final, no entanto não possui utilidade prática na sugestão das recomendações, essa funcionalidade encontra-se presente no código R desenvolvido.

Um dos primeiros pontos que é importante referir diz respeito ao número ações e estados, representados nas colunas e linhas, respetivamente. Este número foi decidido e fixado com base no poder computacional disponível para a realização deste estudo, uma vez que um maior número de ações ou estados iria exigir uma capacidade de processamento maior. Tendo isto em conta, foi também definido 3 como o número máximo de estados para cada ação.

Este modelo auxiliar tem início com a introdução de todas as recompensas ou penalizações obtidas. Após essa introdução é realizada a soma das recompensas/penalizações obtidas para cada categoria e esse valor vai então ser convertido, estando esse resultado presente na coluna denominada “Recompensa corrigida”. Por último, foi realizado o somatório das recompensas corrigidas de cada estado obtendo assim o valor final da recompensa total, que neste caso assume o valor 42, que representa a melhor recompensa total que é possível de atingir.

Valor da Recompensa	Valor da recompensa Corrigida
1	1
2	5
3	7
4	7
5	8
6	9
0	0
-1	-1
-2	-5
-3	-15
-4	-30
-5	-60
-6	-90

Tabela 2: Conversão do valor da recompensa

Na tabela 2 pode observar-se a conversão da recompensa para recompensa corrigida. Este processo foi criado com o objetivo não só de penalizar de elevada forma as situações em que o algoritmo recomenda bastantes vezes itens para os quais se obtém *feedback* negativo, mas também penalizar quando uma categoria é demasiadas vezes selecionada. A título de exemplo, num cenário em que a categoria 1 é recomendada seis vezes obtendo *feedback* negativo em todas elas, o somatório da recompensa seria de -6. No entanto, de modo a incentivar a não escolha dessa categoria, a recompensa corrigida possui um valor de -90. Assim, o algoritmo recebe um incentivo para não recomendar de forma elevada categorias

para as quais se recebeu diversos *feedbacks* negativos, pois isso vai penalizar fortemente o valor da recompensa final dando assim preferência à análise de outras categorias.

Para além disso, é importante também condicionar o algoritmo para que este não selecione apenas categorias para as quais obtém inicialmente *feedback* positivo. Posto isto, é possível observar na tabela 2 que a partir da recompensa que possui o valor 3 o incremento da recompensa corrigida é cada vez menor. Por conseguinte, o algoritmo tende a seleccionar mais categorias, uma vez que isso lhe irá fornecer uma recompensa total superior. Por exemplo, uma categoria que obtenha três ou quatro *feedback* positivos vai levar à mesma recompensa total (7), logo é mais benéfico obter três *feedback* positivos numa dada categoria e um numa outra, pois a recompensa final será superior (7+1). Um outro exemplo interessante é o cenário em que são recomendados 6 filmes da categoria 1 e os mesmos receberem um *feedback* positivo por parte do utilizador obtendo uma recompensa total de 9. No entanto, caso sejam recomendados três filmes da categoria 1 e três da categoria 2 e se obtiver um *feedback* positivo em todos eles a recompensa total será de 14 (7+7), incentivando assim o algoritmo a pesquisar interesses em novas categorias, uma vez que permite a obtenção de uma maior recompensa total. Este é de facto um dos fatores chave diferenciador face a sistemas tradicionais, uma vez que procura ao máximo pesquisar novos interesses do utilizador.

Ainda no tópico da recompensa é importante denotar que a recompensa total só é utilizada uma única vez para o cálculo do Q . Isto ocorre porque este problema está desenhado com base na *backpropagation*, logo só após todas as ações estarem realizadas é que é calculado o Q para cada momento anterior. Naturalmente que o Q poderia ser calculado à medida que se ia obtendo uma recompensa ou penalização, mas foi considerado que para este estudo em concreto esta abordagem faria mais sentido.

Em relação aos valores de $\max Q$ e Q_{Old} , estes são obtidos com base na equação referida na secção 3.3.1 seguindo, no entanto, o conceito de *backpropagation* que implica que o cálculo dos mesmos bem como o Q final é realizado no sentido inverso, iniciando no último momento após o cálculo da recompensa final e evoluindo até o momento inicial, ou seja, o momento que corresponde à primeira recomendação realizada.

Apesar do modelo desenvolvido em *Excel* ser útil na medida em que auxilia no desenvolvimento inicial do algoritmo *Q-learning*, naturalmente foi necessário criar um sistema

que permite não só armazenar todas as matrizes, como também obter a recomendação a efetuar em cada momento ao utilizador. Com esta premissa em mente foi desenvolvido pelo autor um código de programação em R (R Core Team, 2021) que se encontra presente de forma integral no anexo 3.

Esse código teve não só que incorporar o cálculo de Q e todos os seus componentes, como também armazenar todos esses resultados numa matriz e realizar posteriormente a comparação entre elas, de modo a determinar qual a recomendação ideal a realizar a cada utilizador em cada momento com base nas suas preferências específicas.

Tal como referido anteriormente, existe a necessidade de armazenar matrizes e realizar a comparação entre elas, de modo a perceber para cada momento quais são as recomendações que devem ser apresentadas ao utilizador. Estas recomendações são seleccionadas de acordo com as preferências anteriores do utilizador até um dado momento e com base no maior valor de Q final entre todas elas. Ou seja, o objetivo principal das matrizes armazenadas é, numa primeira fase, armazenar os *feedbacks* dos utilizadores e, deste modo, identificar utilizadores com as mesmas preferências. De seguida, o algoritmo irá procurar de entre essas matrizes qual é que possui o maior valor de Q para o momento seguinte ao que se encontra, e deste modo é possível apresentar a próxima recomendação, repetindo este processo até atingir a última *slate*.

Posto isto, foram geradas de forma aleatória seis milhões de matrizes, matrizes essas imprescindíveis numa fase inicial, pois não tendo iterações anteriores com os utilizadores esta foi a forma seleccionada de interagir com os mesmos e de realizar recomendações.

Dessas seis milhões foram substituídas as últimas 125, sendo que 25 foram adicionadas artificialmente com o objetivo de fornecer dados que seriam espectáveis de ser utilizados nas iterações com utilizadores.

As restantes 100 matrizes dizem respeito às iterações de pré-teste e de teste que foram realizadas com os utilizadores, das quais 70 correspondentes à fase de pré-teste e 30 à fase de teste. Um facto relevante é que nas iterações de teste algumas das matrizes utilizadas foram correspondentes às 70 iterações com os utilizadores na fase de pré-teste, o que pode indicar que com o aumento de iterações e a consequente substituição de matrizes aleatórias o sistema de recomendação vai sendo melhorado.

A análise da primeira fase do algoritmo dá-se assim por terminada, a qual permite gerar, armazenar as matrizes e realizar todos os cálculos necessários para atingir o resultado Q para cada uma delas.

Após essa fase, foi necessário dotar o algoritmo da habilidade para interagir com o utilizador, ou seja, recomendar ao fim de cada iteração os próximos filmes a recomendar. Tal como referido anteriormente, esta recomendação baseia-se nos valores das matrizes Q para os respetivos estados e ações em que o utilizador se encontra, sendo que o que vai ser realizado é uma pesquisa que permite encontrar qual das matrizes proporciona a maximização do valor de Q de entre os dados já calculados e armazenados anteriormente, sendo dada dessa forma a melhor solução possível para cada um dos momentos.

Nas últimas linhas de código é realizada, primeiramente, uma pesquisa pelos resultados obtidos na primeira *slate*, que neste caso simbolizam que o utilizador apresentou interesse pelo primeiro filme da primeira categoria, mas não demonstrou pelos outros filmes apresentados, correspondentes a filmes da categoria 2 e 3, respetivamente, tal como pode ser observado nos primeiros 6 dígitos da figura seguinte. A conversão destes valores em filmes concretos é abordada na secção 3.2.5.

1 -1 -1 0 0 0 1 0 1 0 0 1

Fonte: Autor

Figura 4: Primeiro resultado obtido por parte do algoritmo

Na figura anterior é possível também observar pelos últimos seis dígitos que se obtém a primeira recomendação por parte do algoritmo, que indica que devem ser recomendados ao utilizador filmes da categoria 1, 3 e 6 e esta recomendação é a que maximiza o valor Q para o momento em que se encontra.

Após essa recomendação é necessário introduzir a resposta do utilizador à sugestão realizada pelo algoritmo, sendo os dados introduzido da seguinte forma:

[1] 1 -1 -1 0 0 0 1 0 1 0 0 1
1,0,1,0,0,-1

Fonte: Autor

Figura 5: Introdução das preferências do utilizador face à recomendação apresentada

Neste caso o utilizador teve interesse por todos os filmes recomendados com exceção do filme pertencente à categoria 6.

Após a introdução do *feedback* do utilizador, o algoritmo vai realizar uma nova pesquisa tendo em conta todos os dados que obteve do utilizador até ao momento e que maximiza o valor de Q . Com isto, vai realizar uma nova recomendação e este processo vai ser repetido até atingir as últimas três recomendações ou até o algoritmo não ter em memória nenhum tipo de resposta armazenada face às preferências daquele utilizador em concreto.

No cenário em que o algoritmo não tem resposta, foi criado um fluxograma que permite realizar a recomendação ao utilizador de modo a maximizar o valor de Q . Esse fluxograma pode ser encontrado no anexo 4, e é espectável que com o aumento de iterações com diferentes utilizadores o uso deste fluxograma seja desnecessário, uma vez que a ausência de resposta por parte do algoritmo é diminuta.

3.3.3. *Cold Start*

O problema de *cold-start* ocorre quando um sistema de recomendação não tem informações suficientes sobre o utilizador ou item (Lika et al., 2014).

Neste caso em concreto, o problema ocorre no início de cada iteração com um novo utilizador, uma vez que não existem 3 filmes ideais a recomendar na primeira iteração. Posto isto, e de forma a tornar o sistema o mais justo a nível de comparações futuras, foi decidido que todas as iterações com novos utilizadores seriam iniciadas com os 3 primeiros filmes das três primeiras categorias sendo o Id desses filmes 1, 7 e 13 respetivamente. Desta forma existe um início justo em todas as iterações com todos os utilizadores garantindo uma maior fiabilidade nas análises realizadas no próximo capítulo.

Este método pode não ser o mais eficiente e uma solução possível seria recorrendo a análise de dados sociodemográficos como forma de auxílio na primeira recomendação. No entanto, tendo em conta que a base de dados deste estudo é criada única e exclusivamente com dados de iterações com os utilizadores e não se recorre a nenhuma BD existente, esta solução não iria ser útil para os primeiros utilizadores devido à inexistência de dados pelo

que foi decidido não usar e, deste modo, não existiu qualquer tipo de influência nos resultados.

3.3.4. Base de Dados

Tendo em conta que vão ser apresentados diversos filmes aos utilizares foi necessária a criação de uma base de dados com filmes, que se encontra presente no anexo 5. Essa BD é composta por 36 filmes divididos por 6 categorias que correspondem aos estados do *Q-Learning*. Cada categoria é composta por 6 filmes, pois este é o número máximo de filmes da mesma categoria que pode ser apresentado ao utilizador. Em relação à escolha dos filmes em concreto, esta foi baseada no IMDB (IMBD, 2021) (IMDB é uma das fontes mais confiáveis do mundo para conteúdo cinematográfico) e na avaliação e popularidade de cada um dos filmes pertencentes às seis diferentes categorias.

Foram armazenados em *Excel* todos os resultados obtidos através da iteração com o algoritmo de *Q-Learning* tanto para a fase de pré-teste, como para a fase de teste. Para além disso, também foram armazenados os dados obtidos na fase de teste para o sistema de recomendação baseado na técnica de *collaborative filtering*, que vai ser abordado na secção 3.4. Estes resultados podem ser encontrados no anexo 6.

3.3.5. Interface Gráfica

Tal como é possível observar nas imagens dos resultados obtidos pelo algoritmo (secção 3.3.2), estes são apenas valores iguais a 0, 1 e -1, que correspondem à execução de uma ação numa categoria específica e que proporciona uma recompensa ou penalização.

Posto isto, surge a necessidade de não só converter estes valores em filmes em concreto, mas também a criação de uma interface que permita a apresentação dos filmes e a obtenção do *feedback* do utilizador face aos filmes recomendados.

De modo a converter os inputs fornecidos pelo sistema de recomendação em filmes foi criado o seguinte modelo em *Excel* (presente no anexo 12) para efetuar a conversão:

	Slate 1	Slate 2	Slate 3	Slate 4	Slate 5	Slate 6		Soma	Soma	Soma	Soma	Soma	Soma
Categoria 1	1	0	0	1	0	1		1	1	1	2	2	3
Categoria 2	1	0	0	1	0	1		1	1	1	2	2	3
Categoria 3	1	0	0	1	0	1		1	1	1	2	2	3
Categoria 4	0	1	1	0	1	0		0	1	2	2	3	3
Categoria 5	0	1	1	0	1	0		0	1	2	2	3	3
Categoria 6	0	1	1	0	1	0		0	1	2	2	3	3

Id dos Filmes a Recomendar					
1	0	0	2	0	3
7	0	0	8	0	9
13	0	0	14	0	15
0	19	20	0	21	0
0	25	26	0	27	0
0	31	32	0	33	0

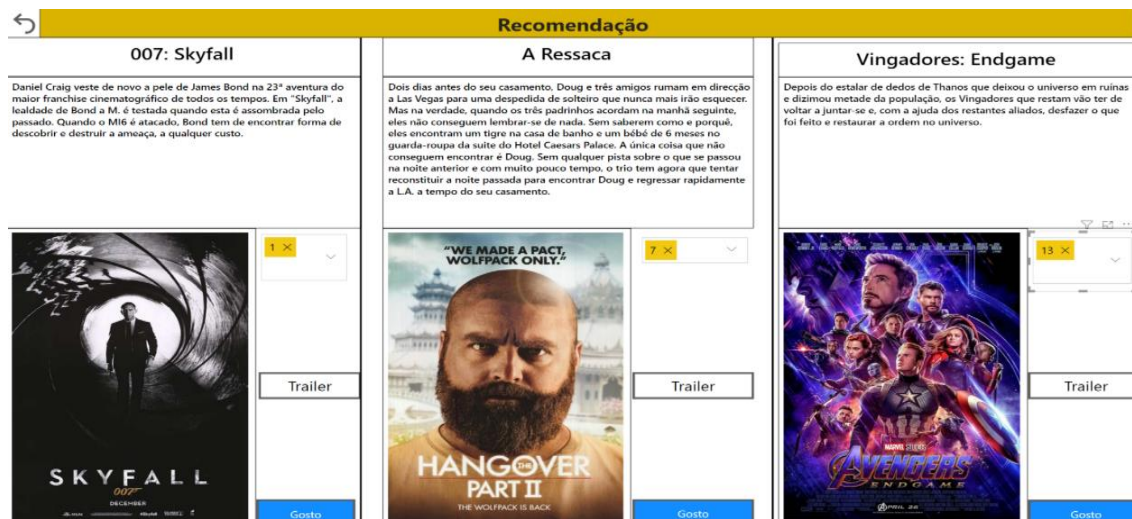
Fonte: Autor

Figura 6: Conversor das recomendações fornecidas pelo algoritmo

Na figura anterior está presente um exemplo de iteração entre o algoritmo e o modelo de conversão. Na primeira tabela são introduzidos os resultados obtidos em R, sendo que esses resultados são introduzidos faseadamente após cada iteração com o utilizador. Após essa introdução vai ser somado o número de vezes que uma dada categoria já foi selecionada sendo que a seleção dos filmes vai ser por ordem crescente do Id do filme numa dada categoria e esse mesmo Id vai ser apresentado na última tabela.

Por exemplo, na *slate 2* verifica-se que existe uma primeira interseção com a categoria 4, sendo fornecido o ID primeiro filme dessa mesma categoria, que neste caso possui ID=19. Na *slate 3* existe novamente uma interseção com a categoria 4, logo vai ser selecionado o próximo filme dessa categoria que possui ID=20.

Após a obtenção dos ID dos filmes a recomendar é necessário apresentar ao utilizador os mesmos e com esse objetivo em mente foi criado, recorrendo à ferramenta *PowerBi* (Microsoft, 2021), uma interface gráfica que permite apresentar ao utilizador os filmes recomendados pelo algoritmo.



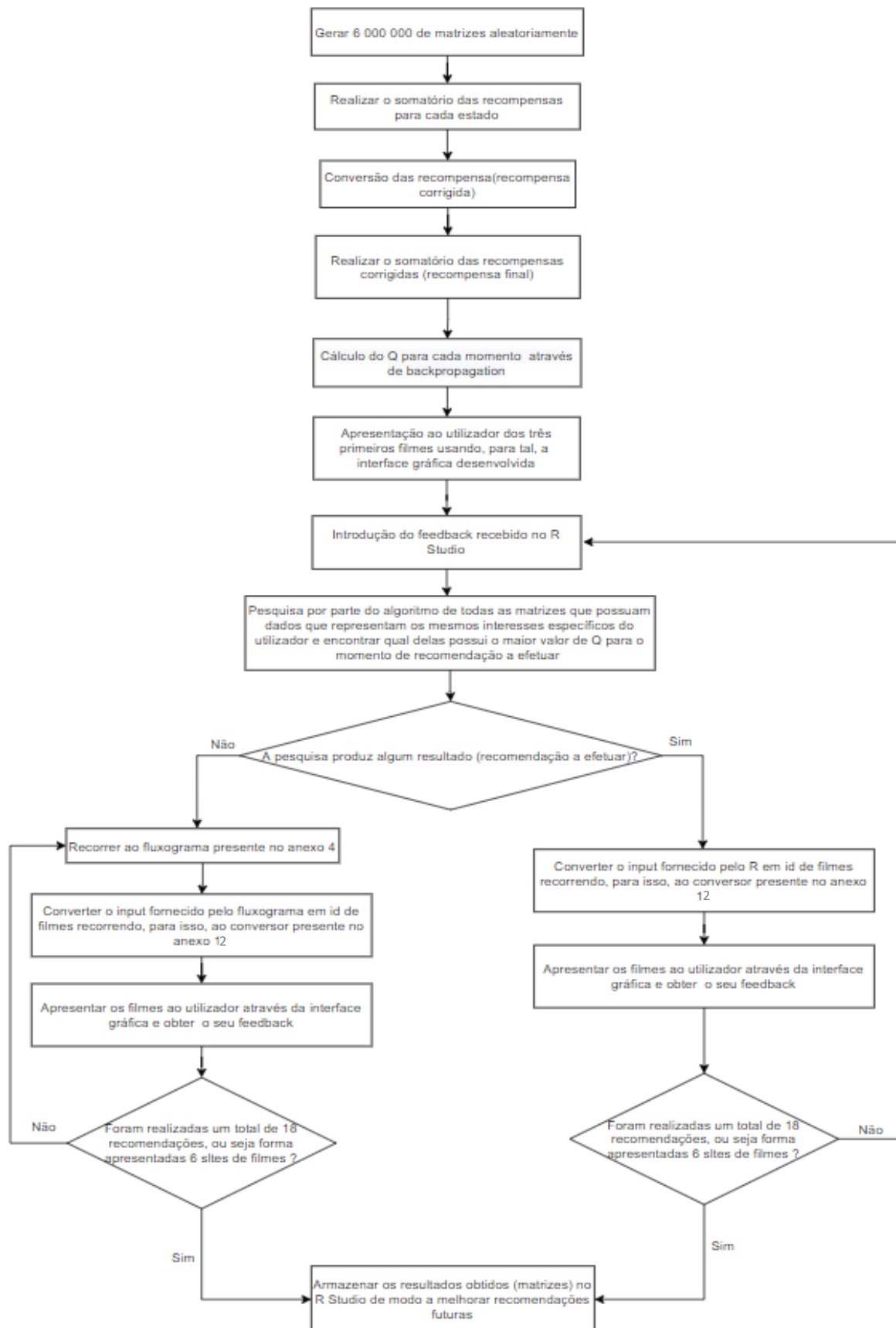
Fonte: Autor

Figura 7: Interface gráfica usada para a apresentação de filmes

Nesta plataforma foi apresentado ao utilizador cada um dos filmes recomendados pelo algoritmo num dado momento. De modo a obter a melhor resposta possível por parte do utilizador foi apresentada uma pequena sinopse para cada filme, assim como uma hiperligação que permite ao utilizador visualizar um curto *trailer*. Deste modo, o utilizador tinha à sua disposição todas as informações necessárias para tomar uma decisão bem fundamentada face ao seu interesse ou falta dele em determinado filme.

3.3.6. Esquema representativo do funcionamento do sistema de recomendação desenvolvido

Como forma de auxiliar a compreensão do sistema desenvolvido, bem como o seu funcionamento, foi criado um esquema, presente na Figura 8, que descreve todos os passos necessários para atingir todas as recomendações personalizadas a apresentar ao utilizador.



Fonte: Autor

Figura 8: Esquema representativo do funcionamento do sistema de recomendação desenvolvido

3.4. Sistema de recomendação usado para comparação

Com objetivo de analisar o sistema de recomendação criado surgiu a necessidade de comparação com um outro sistema. Tendo isto em conta, foi usado um sistema de recomendação *item-based collaborative filtering* (Agrawal, 2021), tendo sido realizadas alterações que permitiram adaptar esse sistema a este estudo em concreto. Este sistema vai apenas ser usado junto dos últimos 30 utilizadores e, tendo como dados de pré-teste os mesmos dados que o algoritmo *Q-Learning* desta forma é possível garantir uma igualdade de cenários e tornar a comparação entre SR o mais justa possível.

O algoritmo usado foi o *K-Nearest Neighbours* (KNN), que é dos algoritmos mais usados para implementar sistemas de recomendação baseados na filtragem colaborativa.

As técnicas baseadas em itens analisam em primeiro lugar a matriz de item-utilizador para identificar as relações entre os diferentes itens e, em seguida, usam essas relações para calcular indiretamente as recomendações para os utilizadores (Sarwar et al., 2001).

Para determinar a similaridade entre as avaliações dos itens pelos utilizadores, isto é, utilizar o KNN, é necessário definir a métrica de similaridade. Uma das métricas mais usadas é o cosseno, e é aquela que irei utilizar nesta análise. A fórmula do cosseno (Lekakos & Caravelas, 2008) é a seguinte:

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}|| * ||\vec{b}||} = \frac{\sum_i a_i b_i}{\sqrt{\sum_i a_i^2} \sqrt{\sum_i b_i^2}}$$

Esta fórmula informa-nos sobre a similaridade entre os dois vetores e $1 - \cos(\vec{a}, \vec{b})$ dá-nos a distância cosseno (Chomboon et al., 2015). Conforme a métrica da similaridade do cosseno se aproxima de 1, o ângulo entre os dois vetores a e b torna-se menor, ou seja, a e b são mais semelhantes entre si. Assim, quando o valor de cosseno é 1 existem semelhanças entre os dados. Se o cosseno for zero os dados não estão relacionados e se o valor for -1 não há similaridade.

Como forma de aplicar este algoritmo foi usada a linguagem de programação *Python* (Van Rossum & Drake Jr, 2009) sendo usado o *Jupyter Notebook* para correr o código.

No anexo 7 encontra-se o código completo, no entanto é relevante explicar nesta secção algumas partes do código, de modo a auxiliar no entendimento do seu funcionamento.

Numa primeira fase são importadas todas as bibliotecas necessárias bem como as bases de dados no formato *csv*. Estas duas bases de dados podem ser vistas através do código *movies.head* (36) e *ratings.head()* (estão presentes no anexo 5 e 6 respetivamente), e dizem respeito à base de dados de filmes e dos dados das preferências dos utilizadores dados estes que foram obtidos na fase de pré-teste.

De seguida, vai ser criada uma matriz que junta o *feedback* dos utilizadores para todos os filmes, contudo cada utilizador avalia um máximo de 18 filmes, pelo que vai ser necessário converter a ausência de *feedback* para o valor 0 e foi exatamente isso que foi efetuado.

A próxima parte do código de elevado interesse para a compreensão do sistema de recomendação, diz respeito ao uso do algoritmo KNN, recorrendo à métrica da distância do cosseno para o cálculo da similaridade.

Por último, será criada a função de recomendação, que começa por procurar se o nome do filme se encontra na base de dados e caso esteja presente são encontrados 15 filmes semelhantes com base na máxima similaridade existente entre eles.

Após a explicação do funcionamento do código resta agora explicar como é realizada a recomendação ao utilizador. Tal como no SR em estudo também este sistema vai ser iniciado com os primeiros filmes da categoria 1,2 e 3, sendo os ids desses filmes 1,7 e 13 respetivamente. Também aqui os filmes vão ser apresentados ao utilizador usando a interface gráfica exposta anteriormente.

Após a obtenção do *feedback* em relação a esses filmes vão ser selecionados os que possuem a maior proximidade em relação aos filmes para os quais se obteve *feedback* positivo. Caso o utilizador não tenha interesse em nenhum dos três primeiros vão ser apresentados os primeiros filmes das categorias seguintes repetindo este processo até ser encontrado um filme que obtenha um *feedback* positivo por parte do utilizador.

Depois de obter três nomes de filmes mais próximos, estes vão ser novamente apresentados ao utilizador. Caso não exista *feedback* positivo em nenhuma das sugestões vão ser sugeridos de acordo com os últimos filmes apreciados na iteração anterior. Este processo

é repetido até ser realizado um total de 18 recomendações. Um ponto importante a referir deste sistema é que nenhum filme pode ser recomendado mais de uma vez, tal como acontece no sistema de recomendação em estudo.

Em caso de empate, isto é, quando as distâncias são iguais, a escolha é feita por ordem crescente do Id da categoria (1-2-3-4-5-6).

De modo a enriquecer este SR com um maior número de dados, tal como foi realizado no sistema de recomendação em estudo, também aqui, após ser finalizado o contacto com o utilizador, as suas escolhas serão colocadas na base de dados que vai ser usada numa nova iteração com um novo utilizador.

Capítulo 4

Resultados e Avaliação

Neste capítulo, numa primeira fase será realizada uma análise de todos os dados recolhidos de modo a facilitar a compreensão dos restantes pontos.

Posteriormente vai ser efetuada uma breve análise dos resultados obtidos pelo algoritmo *Q-Learning*, uma vez que permite retirar indícios acerca do desempenho do sistema de recomendação.

De seguida será realizada uma avaliação *offline* do sistema de recomendação, onde vão ser empregues os principais métodos usados na bibliografia analisada, sendo que em cada caso encontram-se explicados no capítulo 2 os conceitos teóricos que permitem realizar neste capítulo a análise aplicada ao estudo em concreto.

Alguns investigadores (Herlocker et *al.*, 2004) sugerem que novos algoritmos de recomendação devem ser comparados com algoritmos existentes, de modo a comparar a sua performance. Assim, neste estudo vai ser usado um sistema de recomendação baseado em CF e, sempre que se provar relevante, será realizada uma comparação entre ambos os algoritmos.

4.1. Dados recolhidos para avaliação

Antes de ser realizada a avaliação do sistema de recomendação é importante apresentar os dados obtidos e realizar uma análise inicial de modo a facilitar a compreensão de todos os próximos pontos que vão ser abordados.

Tal como já foi referido no capítulo anterior, a base de dados foi construída com base em 100 iterações com 100 pessoas diferentes, sendo que as 70 primeiras foram usadas numa fase de pré-teste e as últimas 30 para teste. Importante lembrar que se entende por pré-teste o conjunto de dados que serviram para “alimentar” os algoritmos com informações reais de utilizadores e a fase de teste a fase que já possui incorporado esses dados.

Nas iterações de pré-teste foi apenas usado o algoritmo *Q-Learning*, no entanto nas últimas trinta foi também usado um algoritmo *K-Nearest Neighbours*, que foi treinado com os mesmos dados obtidos na fase de pré-teste de modo a não influenciar os resultados obtidos. O uso de um sistema de recomendação de *collaborative filtering* com base no algoritmo *KNN* tem como objetivo servir de termo de comparação aquando da avaliação do sistema de recomendação em estudo e foi usada esta técnica em concreto porque é a mais utilizada na elaboração académica e comercial de sistemas de recomendação (Liu et al., 2019). A título de exemplo, a Amazon tem usado a filtragem colaborativa há mais de uma década para recomendar produtos aos seus clientes (Linden et al., 2003).

Em relação à base de dados esta pode ser consultada no anexo 6 e nela podem ser encontrados os dados de pré-teste e de teste. Por linha podemos encontrar a avaliação de um dado utilizador a um determinado filme. Na coluna da avaliação apenas estão presentes 2 valores: 1 e -1, sendo que o 1 representa o interesse no filme e o -1 o desinteresse nesse mesmo filme. De modo a auxiliar a compreensão das restantes colunas, no anexo 5 encontra-se a BD de filmes que permite ligar a categoria bem como o título do filme aos seus respetivos Id's.

4.2. Análise dos resultados do algoritmo *Q-Learning*

Tal como referido na secção 2.6 as melhores formas de perceber a evolução do *Q-Learning* ao longo do tempo é recorrendo à análise da recompensa cumulativa e ao arrependimento que é definido como a diferença entre o *payoff* incorrido pelo algoritmo e o *payoff* ótimo (Yang & Toni, 2018).

Posto isto, foi calculado para cada um dos utilizadores a sua respetiva recompensa e arrependimento (anexo 8), e com esses dados foi então calculada a média para os dados de pré-teste e de teste.

	Dados de Pré-Teste	Dados de Teste
Recompensa Média	12,5	14,3
Arrependimento Médio	29,5	27,7

Tabela 3: Resultados da recompensa e arrependimento médio para os dados de pré-teste e teste

Tal como podemos confirmar na tabela anterior, a recompensa média sofreu um aumento de 1,8 e o arrependimento médio um decréscimo de 1,9. Naturalmente, a análise destes valores de forma isolada não permite retirar conclusões face ao desempenho do sistema de recomendação, no entanto a alteração dos resultados é um indicador da evolução positiva do algoritmo.

4.3. Medidas de Avaliação do Sistema de Recomendação

4.3.1. *Prediction Accuracy*

4.3.1.1. MAE

Para cada um dos utilizadores foi calculado o valor absoluto da diferença entre o *rating* previsto pelo modelo e o *rating* obtido. Após este passo foi então obtida a MAE, que foi calculada separadamente para o grupo de pré-teste e de teste, sendo que neste último foi também feita a separação entre o algoritmo de *Q-Learning* e o *K-Nearest Neighbours*.

	Dados de Pré-Teste	Dados de Teste	
		<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
MAE	0,66	0,74	0,60

Tabela 4: Resultado do Erro Absoluto Médio para os dados de pré-teste e de teste

Através desta métrica pode-se observar que o sistema de recomendação em estudo sofreu uma melhoria na recomendação efetuada. Quer isto dizer que, em média, o sistema conseguiu melhorar as sugestões a apresentar ao utilizador. Por outro lado, quando comparados os dados de pré-teste com os dados de teste obtidos pelo sistema de

recomendação baseado no algoritmo *K-Nearest Neighbours* é possível perceber um aumento do valor de MAE e, consequentemente, um decréscimo na qualidade da recomendação.

4.3.1.2. MSE

Seguindo o mesmo procedimento usado para a métrica anterior, mas aplicado ao MSE foi possível obter os seguintes resultados:

	Dados de Pré-Teste	Dados de Teste	
		<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
MSE	1,33	1,48	1,21

Tabela 5: Resultado do Erro Médio Quadrático para os dados de pré-teste e de teste

A diferença entre esta métrica e a anterior é que penaliza os resultados de forma mais elevada, logo uma pequena diferença faz com que o resultado do MSE sofra uma maior variação.

Como seria de esperar, é possível observar, assim como na métrica anterior, uma melhoria entre o momento de pré-teste e o momento de teste em relação ao sistema de recomendação em estudo e, naturalmente, um decréscimo na qualidade do sistema de recomendação usado como termo de comparação.

4.3.1.3. RMSE

À semelhança do que foi realizado para as duas métricas anteriores também aqui foi realizado o cálculo da diferença entre o rating previsto pelo modelo e o rating obtido, sendo este elevado ao quadrado, permitindo assim o cálculo do RMSE que apresenta os seguintes valores:

	Dados de Pré-Teste	Dados de Teste	
		<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
RMSE	1,15	1,22	1,10

Tabela 6: Resultado da Raiz do Erro Quadrático para os dados de pré-teste e de teste

Estes dados vêm uma vez mais corroborar a conclusão obtida nas duas métricas anteriores, ou seja, o sistema de recomendação em estudo não só teve uma melhoria na qualidade da recomendação como também teve um melhor desempenho quando comparado com o sistema de recomendação que recorre ao algoritmo *K-Nearest Neighbours*.

4.3.2. *Decision Support Metrics*

Apesar de tanto o *Recall* como o F1 serem métricas bastante interessantes do ponto de vista teórico neste caso em concreto não foram usadas, uma vez que para realizar o cálculo do *Recall* é necessário conhecer quais os itens relevantes que não foram recomendados, algo que seria impossível saber a menos que fossem assumidos certos pressupostos, como por exemplo que todos os itens não recomendados eram relevantes ou que nenhum o era. Posto isto, e correndo o risco de influenciar os resultados com informações possivelmente erradas, não foram consideradas essas métricas.

Em relação à precisão essa sim foi calculada sendo que o primeiro passo foi a obtenção de todos os verdadeiros positivos, bem como dos falsos positivos para cada um dos utilizadores que constituem a fase de pré-teste e de teste, o que permitiu obter os seguintes resultados:

	Dados de Pré-Teste	Dados de Teste	
		<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
Verdadeiro positivo (VP)	842	340	377
Falso positivo (FP)	418	200	163
Precisão	67%	63%	70%

Tabela 7: Resultado da Precisão para os dados de pré-teste e de teste

Pode observar-se que existiu um aumento da precisão de 4% para o caso do sistema de recomendação baseado em RL, indicando que o número de itens relevantes face aos itens apresentados aumentou na fase de teste, mas para além disso também é possível denotar que 70% é um valor bastante positivo e que os filmes apresentados pelo algoritmo são relevantes para os diferentes utilizadores. No que diz respeito ao CF aí é possível observar o fenómeno oposto.

Uma outra análise interessante que pode ser efetuada recorrendo à precisão é o estudo da evolução da precisão ao longo das 6 diferentes *slates* para a fase de teste.

Slate	<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
1	68%	68%
2	64%	67%
3	63%	68%
4	63%	69%
5	62%	70%
6	63%	70%

Tabela 8: Evolução da Precisão ao longo das 6 diferentes *slates*

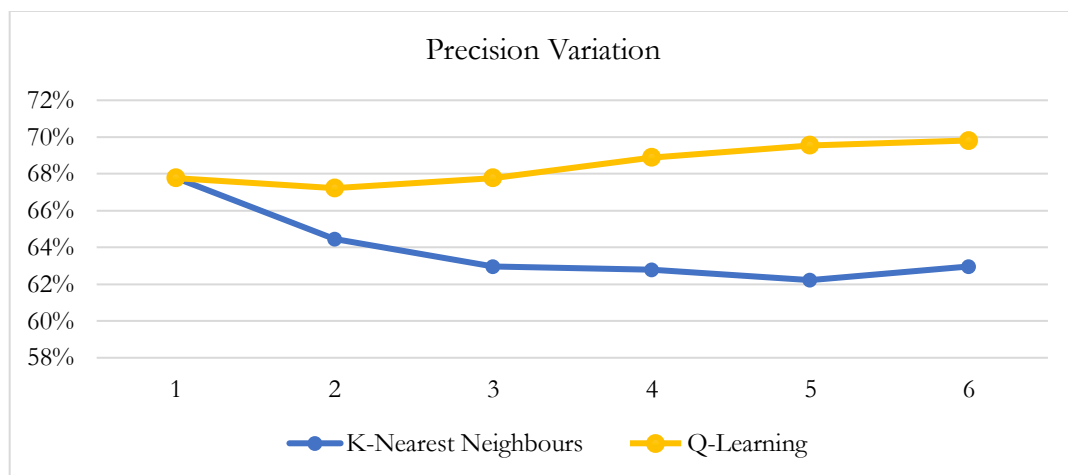


Gráfico 1: Evolução da Precisão ao longo das 6 diferentes *slates*

Os resultados obtidos permitem concluir que os sistemas de recomendação evoluem em sentidos opostos, enquanto o sistema de recomendação em análise (RL) tem um aumento de precisão com o passar das iterações com o utilizador o *collaborative filtering* tem o comportamento oposto. Isto pode ser um indicador de que de facto o algoritmo de RL consegue melhorar as recomendações efetuadas com o aumento das iterações dos utilizadores.

4.3.3. Coverage

No caso concreto em estudo e tendo em conta que se trata de uma prova de conceito para todos os utilizadores a cobertura possui um valor de aproximadamente 50%, uma vez que são sempre apresentados 18 filmes de um total de 36 filmes que podem ser recomendados. O número de filmes apresentados é estático pois com este estudo procurou manter-se sempre que possível uma igualdade de cenários para todos os utilizadores e sistemas de recomendação de modo que a análise não fosse influenciada.

4.3.4. Popularity

No anexo 9 é possível encontrar uma tabela que possui, por ordem decrescente, o número de vezes que cada um dos filmes foi recomendado para todos os dados (pré-teste e teste), no anexo 9b encontra-se uma tabela com os dados também ordenados por ordem decrescente, mas neste caso relativos apenas aos dados de teste e é possível observar que as percentagens bem como a ordem de filmes são bastante semelhantes em ambas as tabelas. Posto isto vai ser analisado o caso que incorpora todos os dados pois, uma vez que são ambos semelhantes, faz todo sentido analisar o maior conjunto de dados possível.

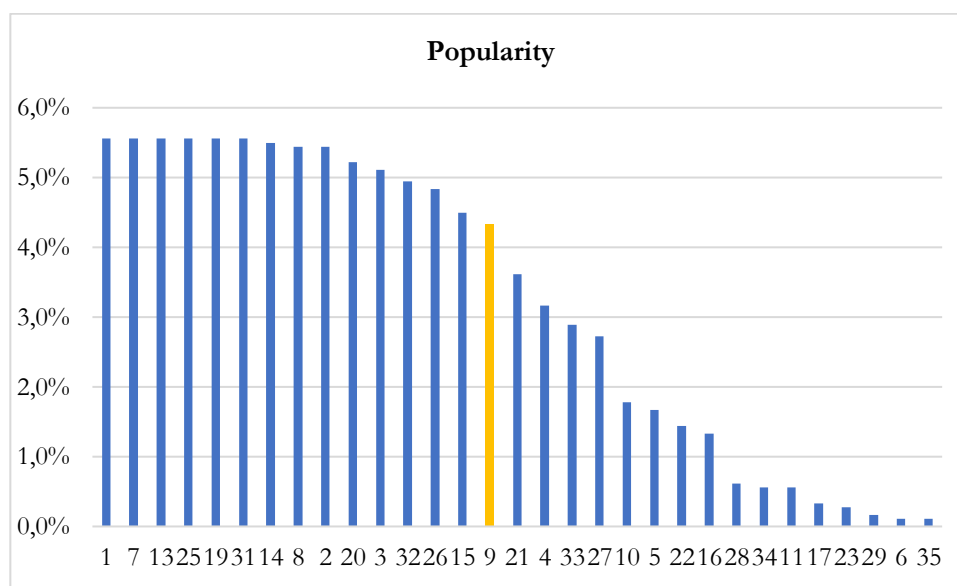


Gráfico 2: Percentagem do número de vezes que cada filme foi recomendado face ao total de recomendações realizadas

O gráfico anterior foi criado com base nos dados do anexo 9 e é possível observar no eixo vertical a percentagem do número de vezes que cada filme foi recomendado face ao total e no eixo horizontal o Id dos filmes.

Os filmes com Id: 1,7,13,25,19,31 são sempre recomendados e isso é perfeitamente compreensível uma vez que estes são os primeiros filmes de cada uma das 6 categorias de filmes disponíveis, o que prova que em todos os casos o algoritmo recomendou pelo menos uma vez um filme de cada categoria.

Com base no anexo 9 e no gráfico anterior é possível concluir que se atinge aproximadamente 80% do número total de recomendações no filme que possui o ID: 9. Esse filme é também o décimo quinto no total de filmes recomendados pelo sistema de recomendação em análise (RL). Tendo isto em conta é possível afirmar que cerca de 80% das recomendações se encontram em cerca de 42% dos filmes disponíveis ($\frac{15}{36}$), provando assim que o sistema de recomendação que recorre ao algoritmo *Q-Learning* possui uma aceitável capacidade de resistir à recomendação de apenas itens populares.

4.3.5. *Diversity Measures*

4.3.5.1. ILS

	Dados de Teste	
	<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
Diversidade	0,45331	0,45076

Tabela 9: Resultado da Diversidade para os dados de teste

No anexo 10 pode encontrar-se o cálculo de todas as combinações de $c(k,j)$ para ambos os sistemas de recomendação referentes à fase de teste. Após esse cálculo pode ser calculado tanto o ILS como o D. Para efeitos de análise a métrica que faz mais sentido analisar é precisamente a diversidade intralista.

Pode observar-se na tabela anterior que apesar do sistema de recomendação usado como termo de comparação possuir um valor ligeiramente superior, na prática ambos os algoritmos possuem valores semelhantes e tendo em conta esses valores é possível afirmar que ambos possuem uma diversidade relativamente aceitável e apesar do valor da diversidade

poder ser maior isso poderia levar a uma diminuição da qualidade das recomendações tornando o sistema de recomendação menos eficiente.

No entanto, uma das razões que pode condicionar a diversidade é o número de filmes presentes na base de dados pelo que seria espectável que com o aumento dessa base de dados a diversidade sofre-se um aumento significativo.

4.3.5.2. *Category Diversity*

Como referido na secção 2.7.6.2, a análise de novas categorias para além das habitualmente recomendadas ao consumidor é um fator bastante importante na medida em que recomendações que possuem apenas itens de algumas categorias específicas são bastante semelhantes e a recomendação muito menos empolgante podendo não só levar ao aborrecimento do utilizador como também correr o risco de não serem apresentados itens interessantes para o consumidor.

Um exemplo prático desta premissa é o caso de um utilizador que gosta da categoria comédia, no entanto este pode ter outros interesses, como por exemplo animação, e caso lhe sejam apenas recomendados filmes do género comédia isso pode levar ao seu desinteresse pelo sistema de recomendação.

Posto isto, e de forma a perceber a *category diversity* foi calculado para ambos os algoritmos na fase de teste o número de categorias que foram abordadas em cada uma das iterações com os diferentes utilizadores (anexo 11). Foi também assumido que uma categoria apenas seria considerada como representada caso tivessem sido recomendados pelo menos dois filmes.

Tendo isto em conta foi possível obter os seguintes resultados:

	Dados de Teste	
	<i>K-Nearest Neighbours</i>	<i>Q-Learning</i>
Diversidade de Categoria	5	6

Tabela 10: Resultado da Diversidade de categoria para os dados de teste

Estes valores resultam da média arredondada às unidades do número de categorias abordadas para cada um dos utilizadores.

Pode observar-se nesses resultados que para a fase de teste em média o sistema de recomendação baseado em *collaborative filtering* recomenda em média 5 categorias a cada utilizador enquanto o SR em estudo recomenda cerca de 6 categorias. Relembrando que o número máximo possível de categorias recomendadas é 6 é possível afirmar que apesar do sistema de recomendação em análise ter um melhor desempenho ambos os sistemas de recomendação apresentam bons resultados. Significando isto que tentam analisar várias categorias evitando dessa forma o desinteresse do utilizador.

No entanto, de modo a analisar de forma mais aprofundada esta métrica, seria interessante no futuro analisar não só para um maior número de categorias, mas também para um maior número de utilizadores pois seria esperado que aí houvesse uma maior divergência entre ambos os sistemas de recomendação uma vez que o algoritmo KNN iria tender a recomendar um menor número de categorias e focar-se nas categorias reconhecidas como seguras.

Capítulo 5

Conclusões

No presente capítulo vai ser realizada, numa primeira fase, uma análise crítica da concretização das questões inicialmente propostas, tentando provar as contribuições desta dissertação para o conhecimento científico já existente. De seguida, será realizada uma análise das principais limitações, bem como de possíveis sugestões para a melhoria de trabalhos futuros.

Nesta dissertação foi criado um sistema de recomendação RQ-Markov completo baseado no algoritmo *Q-Learning* e na metodologia *Markov Chain* tabular que possui a capacidade de filtrar e recomendar itens de forma eficiente com base no perfil de cada utilizador em concreto.

De facto, o sistema criado inclui não só a recomendação e o código que a permite obter, mas também toda a interface gráfica que permite a sua apresentação ao utilizador e isto foi um dos pontos chave definidos inicialmente que foi concretizado com sucesso.

Naturalmente, existem algumas limitações face ao uso deste sistema em ambiente real, tais como o elevado número de dados de pré-teste que iriam ser necessários face ao aumento das bases de dados de itens o que levaria a um aumento do tempo de processamento da recomendação nestas condições. No entanto, com esta dissertação o principal objetivo foi a criação de uma prova de conceito face à utilidade que um sistema de recomendação baseado no algoritmo *Q-Learning* tem para a sugestão eficiente de recomendações ao utilizador e esse objetivo foi concretizado. Para além disso, na secção 5.1 da dissertação vão ser referidas algumas alterações passíveis de realizar no SR criado e que vão abrir portas a avanços futuros neste tópico permitindo a sua adaptação ao uso em ambiente real.

Como forma de comprovar a eficiência do sistema de recomendação foi realizada uma avaliação através de diversas métricas, que de forma isoladas podem ser redutoras, no entanto a sua análise conjunta permite retirar várias conclusões e demonstrar a eficiência do sistema. Em primeiro lugar é possível detetar uma melhoria dos resultados em relação à fase de teste face à fase de pré-teste, o que indica uma evolução do sistema de recomendação. A relevância das recomendações efetuadas também é bastante positiva, para além disso as

recomendações apresentam não só uma elevada resistência à sugestão de apenas itens populares, como também apresentam uma diversidade bastante aceitável seja a nível de itens individuais, como na categoria em que se inserem.

Para além disso, os resultados desta avaliação foram comparados com um sistema de recomendação baseado em *collaborative filtering* e este é um outro ponto chave do estudo realizado, uma vez que permite denotar a melhoria que é gerada pelo sistema de recomendação em estudo face a um sistema que é amplamente usado seja a nível académico, seja a nível comercial, provando assim a pertinência do estudo desenvolvido.

Com o intuito de acrescentar uma contribuição aos estudos já realizados foi desenvolvido um ambiente no qual participaram 100 utilizadores, sendo que tanto para a fase de pré-teste, como para a fase de teste foram sempre usados dados adquiridos com o contacto com utilizadores reais em oposição ao uso ou de bases de dados já existentes ou ambientes virtuais simulados, desta forma os resultados e avaliações obtidas foram muito mais realistas conferindo assim um maior suporte ao estudo realizado.

Em suma com os resultados desta dissertação é possível afirmar que o sistema de recomendação desenvolvido tem um grande potencial de realizar recomendações eficientes ao utilizador, pois as mesmas não só são adequadas ao seu perfil específico como evoluem ao longo das iterações com o mesmo potenciando assim uma melhor experiência.

5.1. Sugestões de melhoria para trabalhos futuros e limitações

Na elaboração desta dissertação foi possível perceber que existem algumas limitações e nesta secção não vão apenas ser enumeradas como também vai ser realizada uma sugestão de possíveis melhorias, bem como novos pontos de estudo que permitam uma melhoria de trabalhos futuros.

Tal como referido anteriormente, uma das principais limitações deste estudo diz respeito à usabilidade do sistema desenvolvido num ambiente real, no entanto é importante realçar que o objetivo inicial deste estudo foi provar a utilidade de um sistema de recomendação baseado no algoritmo *Q-Learning* e este foi concluído com sucesso, uma vez que os resultados obtidos apontam não só para uma eficiência das recomendações realizadas

aos utilizadores mas também uma melhoria face ao sistema de recomendação usado como termo de comparação.

Posto isto, e de modo a adaptar o sistema desenvolvido a um ambiente real, seria importante a utilização de um maior número de estados e ações, uma vez que a base de dados de itens teria que ser superior. Naturalmente que este aumento iria requerer um maior número de dados de pré-teste o que levaria a um aumento do tempo de processamento de cada recomendação, pelo que seria necessário um maior poder computacional. Em alternativa e de modo a reduzir o tempo de processamento seria também possível o cálculo em tempo real do valor de Q face ao cálculo através de *backpropagation*, no entanto isso poderia implicar uma redução da qualidade da recomendação, pelo que uma análise comparativa entre as duas formas de cálculo seria uma excelente contribuição para os estudos científicos já existentes.

Um outro ponto de enorme relevo que é importante referir diz respeito à criação da base de dados de filmes que apesar de ter sido selecionada da melhor forma identificada pode condicionar os resultados obtidos uma vez que tiveram que ser selecionados 36 filmes de um grande conjunto de possibilidades. Posto isto, e de forma a que a base de dados usada não seja condicionante dos resultados, seria interessante, por exemplo, a adaptação do sistema desenvolvido a um *e-commerce* real, em que seria utilizado toda a lista de produtos para realizar a recomendação e aí seria também possível realizar uma comparação face aos resultados obtidos no passado com o sistema de recomendação anteriormente usado. Para além disso, poderia também ser realizada uma avaliação *online*, algo que sem sombra de dúvida é um complemento importante à avaliação *offline*.

No futuro e de modo a analisar a mudança do comportamento e interesses das pessoas, assim como a adaptação do algoritmo no longo prazo, pode ser interessante ter duas fases de teste de modo a detetar se de facto é percecionado pelo sistema de recomendação a mudança de preferências, no entanto seria interessante realizar o estudo para um mercado em que seja mais propensa a alteração de interesses como é o caso do retalho.

Uma limitação que foi encontrada neste estudo diz respeito à falta de dados em determinados momentos da recomendação e apesar de ter sido criada uma forma de solucionar essa questão esta pode não ser a mais eficiente e isso pode condicionar os resultados obtidos, pelo que é importante em estudos futuros realizar uma fase de pré-teste mais extensa e aí possivelmente os resultados irão sofrer uma melhoria.

Neste estudo, não foi considerado como tópico de estudo a ordem através da qual as recomendações são efetuadas, uma vez que este fator não era relevante para o trabalho desenvolvido, no entanto em certos sistemas de recomendação a ordem pode ser um fator importante pois a não recomendação de itens interessantes o mais cedo possível pode levar ao desinteresse do utilizador. Assim, poderá ser relevante no futuro a análise da ordem das recomendações efetuadas.

Nesta dissertação, devido a limitações temporais, não foi possível realizar um maior número de iterações com utilizadores, mas um maior número de iterações de teste seria bastante relevante, pois iria permitir não só fortificar as conclusões retiradas na avaliação, mas também iria levar a uma maior diferença nos resultados entre o sistema de recomendação em estudo face ao sistema de recomendação baseado em *collaborative filtering* permitindo assim comprovar a melhoria criada pelo sistema de recomendação desenvolvido.

Referências

- Abdollahpouri, H., Burke, R., e Mobasher B. (2017). Controlling Popularity Bias in Learning-to-Rank Recommendation. Em *Proceedings of the Eleventh ACM Conference on Recommender Systems*. New York, Agosto 2017, Association for Computing Machinery, 42–46, <https://doi.org/10.1145/3109859.3109912>
- Ackley D. H. e Littman M. L. (1989). Generalization and scaling in reinforcement learning. Em *Proceedings of the 2nd International Conference on Neural Information Processing Systems*. Janeiro 1989. Cognitive Science Research Group, 550-557.
- Adomavicius G. e Zhang J. (2012). Impact of data characteristics on recommender systems performance. *ACM Transactions on Management Information Systems*, **3**(1): 1-17.
- Afsar, M., Crump, T. e Far, B. (2021). Reinforcement Learning based Recommender Systems: A Survey. University of Calgary.
- Agrawal, R. (2021). How to Build a Book Recommendation System. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2021/06/build-book-recommendation-system-unsupervised-learning-project/>
- Bobadilla, J., Ortega, F., Hernando, A. e A. Gutiérrez (2013). Recommender systems survey. *Knowledge-Based Systems*, **46**: 109 – 132.
- Casas, N. (2017). Deep Deterministic Policy Gradient for Urban Traffic Light Control. Cornell University.
- Celebi, M. E. e Aydin, K. (2016). Unsupervised learning algorithms. Spring.
- Chomboon, K., Chujai, P., Teerarassammee, P., Kerdprasop, K. e Kerdprasop, N. (2015). An Empirical Study of Distance Metrics for k-Nearest Neighbor Algorithm. Em *Proceedings of the 3rd International Conference on Industrial Application Engineering 2015*. 280-285.
- Ciaburro, G. (2019). Hands-on Reinforcement Learning with R. Birmingham, UK.
- Del Olmo, F. H. e Gaudioso, E. (2008). Evaluation of recommender systems: A new approach. *Expert Systems with Applications*. **35**(3): 790-804.

- Ekstrand, M. D., Riedl, J. T. e Konstan, J. A. (2010). Collaborative Filtering Recommender Systems, *4*(2): 81-173.
- El Naqa, I., e Murphy M.J. (2015). What Is Machine Learning? Em *Machine Learning in Radiation Oncology*. Springer, Cham, https://doi.org/10.1007/978-3-319-18305-3_1
- Esmaeili, L., Mardani, S., Golpayegani, S. e Madar, Z. (2020). A novel tourism recommender system in the context of social commerce. *Expert Systems with Applications*, 149, https://www.sciencedirect.com/science/article/pii/S0957417420301263?casa_token=YgIloH8NilMAAAAA:T83AxH6FbkHPau7DNzBIdo0VyXk50U9UVSl5D9ZvGhrqVCCtPzzvM6yZD7Q8dp2SAQQsrtiSK5Y
- Gunawardana, A. e Shani, G. (2009). A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *Journal of Machine Learning Research*, **10**: 2935-2962
- Halder, S., Sarkar, A. e Lee, Y. (2012). Movie Recommendation System Based on Movie Swarm. *2012 Second International Conference on Cloud and Green Computing*.
- Herlocker, J., Konstan, J. A., e Riedl, J., (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, **5**(4): 287–310.
- Herlocker, J., Konstan, J. A., Terveen, L. G. e Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, **22**(1): 5–53.
- IMDb (2021). “Ratings, Reviews, and Where to Watch the Best Movies & Tv Shows.”, <https://www.imdb.com/>
- Isinkaye, F. O., Folajimi, Y. O. e Ojokoh, B. A. (2015) Recommendation Systems: Principles, Methods and Evaluation. *Egyptian Informatics Journal*, **16**(3): 261–273, [10.1016/j.eij.2015.06.005](https://doi.org/10.1016/j.eij.2015.06.005).
- Joshi, A. V. (2020). Introduction to AI e ML. Em *Machine Learning and Artificial Intelligence*. Springer, Cham, https://doi.org/10.1007/978-3-030-26622-6_1
- Konstan J. A. e Riedl J. (2012). Recommender systems: from algorithms to user experience. *User Model User-Adapt Interact*, **22**: 101–23, [https://doi.org/10.1007/s11257-011-9112-](https://doi.org/10.1007/s11257-011-9112-x)

- Koohi, H. e Kiani, K. (2016). User based Collaborative Filtering using fuzzy C-means. *Measurement*, **91**: 134–139.
- Kotsiantis, S. (2007). Supervised Machine Learning: A review of classification techniques. *Informatica*. **31**(3): 249-268.
- Kumar, M. V. e Kumar, P. N. (2019). A study on different phases and various recommendation system techniques. *International Journal of Recent Technology and Engineering*, **7**(5C): 38-41.
- Kumar, P. e Thakur, R. S. (2018). Recommendation system techniques and related issues: a survey. *International Journal of Information Technology*, **10**: 495–501, <https://doi.org/10.1007/s41870-018-0138-8>
- Lekakos, G. e Caravelas, P. (2008). A hybrid approach for movie recommendation. *Multimedia Tools and Applications*, **36**: 55–70, <https://doi.org/10.1007/s11042-006-0082-7>
- Li, Y. (2017). Deep Reinforcement Learning: An Overview. Cornell University.
- Lika, B., Kolomvatsos, K. e Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*. **41**(4): 2065-2073, <https://doi.org/10.1016/j.eswa.2013.09.005>.
- Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. e Wierstra, D. (2015). Continuous control with deep reinforcement learning. Cornell University.
- Linden, G., Smith, B., e York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* **4**: 1
- Liu, F., Tang, R., Li, X., Zhang, W., Ye, Y., Chen, H., Guo, H. e Zhang, Y. (2018). Deep reinforcement learning based recommendation with explicit user-item interactions modeling.
- Liu, Q., Reiner, A. H., Frigessi, A. e Scheel, I. (2019). Diverse personalized recommendations with uncertainty from implicit preference data with the Bayesian Mallows model. *Knowledge-Based Systems*, **186**

- MacKenzie, I., Meyer, C. e Noble S. (2013). How retailers can keep up with consumers. <https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers>
- Maglogiannis, I. G. (2007). Emerging artificial intelligence applications in computer engineering. Ios Press.
- Mehrjoo, S., Hajipour, F. e Mehrjoo, M. (2019). Optimal Diversity of Recommendation List for Recommender Systems based on the Users' Desire Diversity. *Journal of Information Science Theory and Practice*, **7**(3): 31-39
- Microsoft (2021). Crie uma cultura baseada em dados com business intelligence para todos, <https://powerbi.microsoft.com/pt-pt/>
- Microsoft (2021). Microsoft Excel. <https://www.microsoft.com/pt-pt/microsoft-365/excel>
- Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill Education.
- Mooney, R. J. e Roy, L. (2000). Content-based book recommending using learning for text categorization, 195–204.
- Movielens (2021). Non-commercial, personalized movie recommendations. <https://movielens.org/>
- Netflix Research (2021). Recommendations. <https://research.netflix.com/research-area/recommendations>
- Pan, C. e Li, W. (2010). Research paper recommendation with topic analysis. *International Conference on Computer Design and Applications*, **4**: 4-264, 10.1109/ICCDA.2010.5541170.
- Park, Y. e Tuzhilin, A. (2008). The long tail of recommender systems and how to leverage it. Em Proceedings of the 2008 ACM conference on Recommender systems. ACM, 11–18
- Patel, B., Desai, P. e Panchal, U. (2017). Methods of Recommender System: A Review. *2017 International Conference on Innovations in Information, Embedded and Communication Systems*, 1-4, 10.1109/iciiecs.2017.8275856
- Portugal, I., Alencar, P. e Cowan, D. (2018). The use of machine learning algorithms in recommender systems: A systematic review. *Expert Systems with Applications*, **97**: 205-227

- Poriya, A., Patel, N., Bhagat, T. e Sharma, R. (2014). Non-Personalized Recommender Systems and Userbased Collaborative Recommender Systems. *International Journal of Applied Information Systems*. **6**(9)
- R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. <https://www.R-project.org/>.
- Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A. e Riedl, J. (2002). Getting to know you: learning new user preferences in recommender systems. *Em Proceedings of the 7th international conference on Intelligent user interfaces*, 127-134, <https://doi.org/10.1145/502716.502737>
- Rathore, F., e Farooq, F. (2020). Information Overload and Infodemic in the COVID-19 Pandemic. *JPMMA. The Journal of the Pakistan Medical Association*, **70**(Suppl 3)(5): 162-165, <https://pubmed.ncbi.nlm.nih.gov/32515403/>
- Resnick, P. e Varian, H. (1997). Recommender systems. *Communications of the ACM*, 56–58.
- Sarwar, B., Karypis, G., Konstan, J., e Riedl, J. (2001). Item-Based Collaborative Filtering Recommendation Algorithms. University of Minnesota, Minneapolis.
- Sharma, L. e Gera, A. (2013). A Survey of Recommendation System: Research Challenges. *International Journal of Engineering Trends and Technology*, **4**(5): 1989-1992
- Shyalika, C. (2019). A Beginners Guide to Q-Learning. Towards Data Science. <https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c>
- Sidana, S. (2018). Recommendation systems for online advertising. Computers and Society. Université Grenoble Alpes
- Sutton, R. S. e Barto, A. G. (1998). Reinforcement learning: An introduction. MIT press. <http://incompleteideas.net/book/bookdraft2017nov5.pdf>
- Taghipour, N., Kardan, A. e Ghirdary, S. (2007). Usage-Based Web Recommendations: A Reinforcement Learning Approach. In *Proceedings of the 2007 ACM Conference on Recommender Systems*, 113-120.
- Wang, X., Wang, Y., Hsu, D. e Wang, Y. (2014). Exploration in Interactive Personalized Music Recommendation: A Reinforcement Learning Approach. *Association for Computing Machinery*, **11**(1): 1-22 <https://doi.org/10.1145/2623372>

- Watkins, C. e Dayan, P. (1992). *Q-learning*. Springer
- Watkins, C.J.C.H. (1989). Learning from delayed rewards. PhD Thesis, University of Cambridge, England.
- Wiering, M. e Van Otterlo, M. (2012). Reinforcement Learning. Springer, **12**
<https://link.springer.com/content/pdf/10.1007%2F978-3-642-27645-3.pdf>
- Yadav, N., Kumar, R., Singh, A. e Pal, S. (2011). Diversity in Recommendation System: A Cluster Based Approach? Indian Institute of Technology (BHU).
- Yang, K. e Toni, L. (2018). Graph-Based Recommendation System. 2018 IEEE Global Conference on Signal and Information Processing, 798-802
- Zhang, Y. (2010). New advances in machine learning. BoD—Books on Demand.
- Zhao, X., Zhang, L., Xia, L., Ding, Z., Yin, D. e Tang, J. (2017). Deep reinforcement learning for list-wise recommendations.
- Zhao, X., Zhang, L., Xia, L., Ding, Z., Yin, D., e Tang, J. (2018). Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning.
- Ziegler, Cai-Nicolas, McNee, S., Konstan, J. e Lausen G. (2005). Improving recommendation lists through topic diversification. Em *Proceedings of the 14th international conference on World Wide Web*. Association for Computing Machinery, 22-32

Anexos

Anexo 1 - Exemplificação do funcionamento do sistema de recomendação em estudo

Neste anexo encontra-se um vídeo que demonstra uma iteração completa entre um utilizador e o sistema de recomendação desenvolvido. Com o objetivo de encurtar o tempo de vídeo foram removidos os trechos em que o algoritmo processava a recomendação a efetuar.

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 2 - Exemplificação do funcionamento do sistema de recomendação usado como termo de comparação

Neste anexo encontra-se um vídeo que demonstra uma iteração completa entre um utilizador e o sistema de recomendação baseado em *Collaborative Filtering*.

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 3 – Código do sistema de recomendação em estudo

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 4 – Fluxograma

Este anexo procura dar continuidade de resposta ao algoritmo nas situações em que o mesmo não tem resposta armazenada, para tal vão ser indicadas quais as categorias que devem ser recomendadas a cada momento S_t e que vão maximizar a função Q .

De modo a atingir esse fim este fluxograma vai ser usado para cada um dos estados a partir do momento em que o algoritmo deixou de proporcionar resposta e vai ser utilizado até se atingir as 18 recomendações.

O método de funcionamento deste fluxograma centra-se em decidir se para cada momento o estado X deve ser ou não ser selecionado para ser apresentado ao utilizador.

Para aceder ao fluxograma é necessário recorrer ao website: <https://app.diagrams.net/> e introduzir o ficheiro denominado Anexo 4.

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 5 - Base de dados de Filmes

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 6 – Resultados das iterações com utilizadores

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 7- Código do sistema de recomendação usado como termo de comparação

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 8 - Recompensa e Arrependimento de cada utilizador

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 9 – Resultados da Popularidade de cada filme para os dados da fase de pré-teste e teste

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 9b - Resultados da Popularidade de cada filme para os dados da fase de teste

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 10 – Combinações do grau de similaridade entre itens

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 11 – Número de filmes recomendados por categoria a cada utilizador da fase de teste

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>

Anexo 12 – Conversor de resultados do algoritmo

O anexo encontra-se disponível online no seguinte endereço Web (no caso de a hiperligação não funcionar é recomendado que seja colada diretamente a referência do navegador):

<https://drive.google.com/drive/folders/1Be4W3wu1Msg0d1fQwnYFNac0l55pC8A4?usp=sharing>