

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Nícollas de Campos Silva

**ACTIVE LEARNING IN CONTEXTUAL BANDITS: HANDLING THE
UNCERTAINTY ABOUT THE USER'S PREFERENCES IN
INTERACTIVE RECOMMENDATION SYSTEMS**

Belo Horizonte
2023

Nícollas de Campos Silva

**ACTIVE LEARNING IN CONTEXTUAL BANDITS: HANDLING THE
UNCERTAINTY ABOUT THE USER'S PREFERENCES IN
INTERACTIVE RECOMMENDATION SYSTEMS**

Final Version

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfilment of the requirements for the degree of Doctor in Computer Science.

Advisor: Adriano César Machado Pereira
Co-Advisor: Leonardo Chaves Dutra da Rocha

Belo Horizonte
2023

Silva, Nícollas de Campos.

S586a Active learning in contextual bandits: [recurso eletrônico]
handling the uncertainty about the user's preferences in
interactive recommendation systems / Nícollas de Campos
Silva. 2023.

1 recurso online (150 f. il, color.): pdf.

Orientador: Adriano César Machado Pereira.

Coorientador: Leonardo Chaves Dutra da Rocha

Tese (Doutorado) - Universidade Federal de Minas
Gerais, Instituto de Ciências Exatas, Departamento de Ciência
da Computação.

Referências: f. 87 -114.

1. Computação – Teses. 2. Sistemas de Recomendação,
Multi-Armed Bandits. I. Adriano César Machado Pereira. II.
Leonardo Chaves Dutra da Rocha. III. Universidade Federal
de Minas Gerais, Instituto de Ciências Exatas, Departamento de
Ciência da Computação. IV. Título.

CDU 519.6*73(043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

ACTIVE LEARNING IN CONTEXTUAL BANDITS: HANDLING THE UNCERTAINTY ABOUT THE USER'S PREFERENCES IN LINTERACTIVE RECOMMENDATION SYSTEMS

NÍCOLLAS DE CAMPOS SILVA

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:

Prof. Adriano César Machado Pereira - Orientador
Departamento de Ciência da Computação - UFMG

Prof. Leonardo Chaves Dutra da Rocha - Coorientador
Departamento de Ciência da Computação - UFSJ

Prof. Rodrygo Luis Teodoro Santos
Departamento de Ciência da Computação - UFMG

Prof. Anisio Mendes Lacerda
Departamento de Ciência da Computação - UFMG

Dr. Fernando Henrique Jesus Mourão
SEEK AI Labs

Prof. Marcelo Garcia Manzato
Departamento de Ciências da Computação - USP

Belo Horizonte, 03 de julho de 2023.



Documento assinado eletronicamente por **Adriano Cesar Machado Pereira, Professor do Magistério Superior**, em 26/09/2023, às 11:22, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Rodrygo Luis Teodoro Santos, Professor do Magistério Superior**, em 26/09/2023, às 12:23, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Anisio Mendes Lacerda, Professor do Magistério Superior**, em 02/10/2023, às 07:33, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Leonardo Chaves Dutra da Rocha, Usuário Externo**, em 04/10/2023, às 10:24, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Marcelo Garcia Manzato, Usuário Externo**, em 09/10/2023, às 11:54, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **FERNANDO HENRIQUE DE JESUS MOURAO, Usuário Externo**, em 19/10/2023, às 09:09, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2659449** e o código CRC **DA18B7C6**.

*To all my family, especially my mother and wife.
To my dear uncle, whose influence has profoundly shaped my
journey until here.*

Acknowledgments

First of all, I want to express my heartfelt gratitude to God for being my support throughout this challenging journey. There have been countless moments in my life when His presence has brought me comfort and strength, enabling me to overcome every obstacle in my path. I am profoundly thankful for His divine guidance and for the countless blessings often through the intercession of the Virgin Mary.

Then, I would like to thank all my family and friends. You are the roots that ground me and the branches that lift me higher. My mother's love and unwavering support have shaped me into the person I am today. My grandmother's faith and uncle's willpower taught me to fight every day and never give up on my dreams. Especially, I also would like to express my gratitude to my beloved wife, Bárbara. With you, life is a beautiful journey filled with love, laughter, and endless possibilities. You are not just my partner but also my best friend, and I cherish every moment we share together. Moreover, to my cherished friends, many thanks for all the courage that you have inspired me. Our adventures and countless memories are treasures I dearly hold. Your friendship is a priceless gift that adds richness to my life, and I am grateful for every moment we have spent together.

Finally, I would like to express my acknowledgement to my esteemed professors, Adriano and Leonardo. Your guidance and mentorship have been invaluable. You have instilled in me a passion for learning that will continue to guide my path forward. Moreover, many thanks to my wonderful college friends that I met during this journey. The late-night coding and writing articles have created bonds that I will forever cherish. The RecLab that we have founded will be always one of my greatest achievements and I will hold it with pride and prestige. Our achievements in this research group have been memorable and recognised all over the world. Thank you for being an essential part of this experience.

Warmest regards,
Nícollas Silva.

“Non abbiate paura.”
(Johannes Paul II)

Resumo

Atualmente, Sistemas de Recomendação (SsR) têm se preocupado com o ambiente online de aplicações do mundo real, onde o sistema deve continuamente aprender e prever novas recomendações. Trabalhos atuais têm abordado essa tarefa como um problema de *Multi-Armed Bandit* (MAB) ao propor modelos de *Contextual Bandit* (CB). A ideia é aplicar técnicas de recomendação usuais para explorar as preferências do usuário, enquanto o sistema também tenta aprender novas informações sobre seus gostos. Contudo, o nível de personalização desses modelos ainda está diretamente relacionado às informações previamente disponíveis sobre os usuários. Após uma extensa revisão da literatura sobre o assunto, observamos que os algoritmos atuais têm negligenciado o impacto de cenários de incerteza sobre as preferências do usuário. Assumindo que o modelo *bandit* pode aprender independentemente do item recomendado, tais modelos estão perdendo uma oportunidade de obter mais informações sobre os usuários. Nesse sentido, esta dissertação aborda o desafio de lidar com cenários de incerteza em modelos de *Contextual Bandit*. Em particular, investigamos dois cenários comuns em sistemas interativos: (1) quando o usuário entra pela primeira vez e (2) quando o sistema continua fazendo recomendações incorretas devido a suposições enganosas anteriores. Em ambos os cenários, propomos introduzir conceitos de *Reinforcement Learning* para representar o *trade-off* entre *exploitation* e *exploration* nos modelos *bandit*. Nossa solução consiste em recomendar itens não personalizados com base na entropia e na popularidade para obter mais informações sobre o usuário sem diminuir a precisão do modelo quando um cenário de incerteza é observado. Essa solução é então instanciada em três algoritmos *bandit* tradicionais, criando novas versões de cada um deles. Experimentos em domínios de recomendação distintos mostram que essas versões modificadas superam suas versões originais e todas as demais linhas de base, aumentando a acurácia a longo prazo. Além disso, uma avaliação contrafactual válida que tais melhorias não foram simplesmente alcançadas devido ao viés de conjuntos de dados offline.

Palavras-chave: Sistemas de Recomendação, Multi-Armed Bandits

Abstract

Nowadays, Recommendation Systems (RSs) have been concerned about the online environment of real-world applications where the system should continually learn and predict new recommendations. Current works have addressed this task as a Multi-Armed Bandit (MAB) problem by proposing Contextual Bandit (CB) models. The idea is to apply usual recommendation techniques to exploit the user's preferences while the system also addresses some exploration to learn new information about their tastes. The personalisation level of such models is still directly related to the information previously available about the users. However, after an extensive literature review on this topic, we observe that current algorithms have neglected the impact of scenarios of uncertainty about the user's preferences. Assuming that the bandit model can learn regardless of the recommended item, such models are wasting an opportunity to get more information about the users. In this sense, this dissertation addresses the challenge of handling scenarios of uncertainty in Contextual Bandit models. In particular, we investigate two usual scenarios in interactive systems: (1) when the user joins for the first time and (2) when the system continually makes wrong recommendations because of prior misleading assumptions. In both scenarios, we propose to introduce concepts from the Active Learning theory to represent the usual trade-off between exploration and exploitation in the bandit models. Our solution consists of recommending non-personalised items based on entropy and popularity to get more information about the user without decreasing the model's accuracy when an uncertain scenario is observed. This solution is then instantiated into three traditional bandit algorithms, creating new versions of each of them. Experiments in distinct recommendation domains show that these modified versions outperform their original ones and all baselines by increasing the cumulative reward in the long run. Moreover, a counterfactual evaluation validates that such improvements were not simply achieved due to the bias of offline datasets.

Keywords: Recommendation Systems, Multi-Armed Bandits


List of Figures

1.1	The feedback loop between a user and the recommendation system [50]. At each user’s interaction, the system recommends an item(s) and receives feedback from a user. The recommendation and the user’s choice are made according to the model implemented and based on the user’s preferences.	18
2.1	An example of a recommendation scenario with ratings from 1 to 5.	28
2.2	Standard model defined to mitigate the recommendation problem.	29
2.3	Recommendation system generations.	30
2.4	Standard recommendation engine of Content-based methods.	31
2.5	Standard definition of Hybrid methods.	34
2.6	The traditional framework of Multi-Armed Bandits.	38
2.7	Distinct exploration strategies applied in each MAB algorithm.	38
3.1	Distinct challenges about the user’s preferences in their journey.	45
3.2	Popularity versus the items features vector importance measured by $q_i \cdot q_i^\top$. There is a significant correlation between the items features vector of Matrix Factorisation models and the most popular items.	49
3.3	An illustration of our methodology created to identify the problem’s impact on contextual bandit algorithms.	50
3.4	Impact of a naive initialisation in Contextual Bandit. While pure exploration requires more interactions to achieve higher recall, pure exploitation does it so fast. However, this fast learning has a considerable impact on the model’s performance as a whole. In turn, balancing exploration and exploitation improves the system’s precision without requiring many initial interactions.	53
4.1	Active Learning approaches in recommendation systems [75].	57
5.1	Global timestamp cut to select new users in offline datasets.	65
A.1	Systematic Literature Review protocol.	116
A.2	Annual publications about MAB in the recommendation field.	120
A.3	The main domains used in MAB research in the recommendation field.	121
A.4	MAB algorithms usually applied in the recommendation field.	122
A.5	RSs techniques usually combined with MAB algorithms.	123
A.6	An updated picture of the main RS techniques applied with MAB algorithms according to the works selected by our SLR.	124

A.7	Frequency of works which implemented each evaluation metric.	128
B.1	An overview of the iRec framework. It is composed of three main components that allow a researcher to simulate an interactive recommendation scenario and compare distinct algorithms by a fair evaluation of their quality.	136
B.2	iRec-cmdline: an application of the framework.	142

List of Tables

2.1	Classes of collaborative filtering methods.	32
3.1	The typical linear implementation of the three main MAB algorithms.	44
3.2	Datasets from three different recommendation domains.	49
5.1	An overview of the datasets applied in this work.	65
5.2	Parameters identified after a Grid Search tuning using 20% of the training set.	69
5.3	Identified parameters for our modified version of the algorithms.	70
5.4	A comparison between original Contextual Bandits and their modified versions for each recommendation domain. While algorithms labelled with $t = 0$ only address the pure cold-start, those labelled with $t \geq 0$ mitigate both scenarios of uncertainty. The results show a significant improvement achieved by the modified algorithms based on the Wilcoxon test with a p-value = 0.05. The symbol \blacktriangle represents significant gains, \bullet represents statistical draws, and \blacktriangledown represents significant losses.	71
5.5	Synthetic Dataset statistics.	73
5.6	Estimated policy values of the modified and original versions in a synthetic dataset. All modified strategies have outperformed the original ones. LinUCB has demonstrated the highest performance across all other algorithms.	74
5.7	The relative value achieved by each algorithm considering the original policy π_0 used to create the synthetic dataset. Such gains represent if the new algorithm would perform better or not than the original recommender used to collect the offline dataset.	75
5.8	Cumulative reward of all baselines and the three modified versions made by this work. Results show a statistical improvement in the modified algorithms by applying the Wilcoxon test with a p-value = 0.05. The symbol \blacktriangle represents significant gains, \bullet represents statistical draws, and \blacktriangledown represents significant losses.	77
A.1	Inclusion and exclusion criteria of our SLR.	117
A.2	Data extraction form.	118
A.3	Conferences and journals that have accepted more papers about multi-armed bandits in the recommendation field.	120
A.4	The main repositories with the datasets most applied by researchers to study MAB proposals in the recommendation field.	126

B.1	An overview of each framework proposed in the literature and our iRec.	134
B.2	Datasets currently accepted by the iRec.	137
B.3	The main scripts	143
B.4	Experimental results obtained using all configurations files described in Section 4. The results were compared based on Wilcoxon Test with p-value=0.05. The  symbol indicates statistical best results.	148

Contents

1	Introduction	17
1.1	Background	17
1.2	Thesis Statement	19
1.3	Motivations	22
1.4	Main Contributions	24
1.5	Outline	25
2	Background Concepts	27
2.1	Recommendation Systems	27
2.1.1	Recommendation Problem	27
2.1.2	Recommendation Methods	30
2.1.2.1	Content-Based	31
2.1.2.2	Collaborative Filtering	32
2.1.2.3	Hybrid Methods	33
2.2	Recommendation Challenges	34
2.2.1	Cold-Start problem	34
2.2.2	Misleading Assumptions	35
2.3	Multi-Armed Bandits	37
2.3.1	ϵ -Greedy	39
2.3.2	Upper Confidence Bound (UCB)	39
2.3.3	Thompson Sampling (TS)	40
2.4	Summary	41
3	Contextual Bandits & the user's preferences uncertainty	42
3.1	Contextual Bandits	42
3.2	User uncertainty in Contextual Bandits	44
3.2.1	Pure Cold-Start	45
3.2.2	Misleading Assumptions	46
3.3	Practical Outgrowths	47
3.3.1	Algorithms implication	47
3.3.2	Item features bias	48
3.4	Impact on Recommendation Systems	50

3.4.1	Evaluation Methodology	50
3.4.2	Experimental Setup	51
3.4.3	Results & Discussion	52
3.5	Summary	54
4	An Active Learning approach in Contextual Bandits	56
4.1	Active Learning	56
4.2	Addressing scenarios of uncertainty	58
4.2.1	Mitigating the pure cold-start problem	59
4.2.2	Mitigating misleading assumptions	60
4.3	Modified Contextual Bandits	60
4.3.1	Contextual Linear ϵ -Greedy	61
4.3.2	Contextual LinUCB	61
4.3.3	Contextual PTS	62
4.4	Summary	63
5	Experiments: Results & Discussions	64
5.1	Experimental Setup	64
5.2	Modified vs. Original Algorithms	68
5.3	Counterfactual Evaluation	72
5.4	Baselines Comparison	76
5.5	Summary	79
6	Conclusions & Future Work	80
6.1	Restatement of Thesis	80
6.2	Empirical Findings	81
6.3	Summary of Contributions	82
6.4	Limitations of the Work	83
6.5	Future Research	84
6.6	Final Remarks	85
	Bibliography	87
	Appendix A A Systematic Literature Review about Multi-Armed Bandits in Recommendation Systems	115
A.1	Systematic Literature Review Protocol	115
A.1.1	Phase 1: Research questions, search strings and sources	116
A.1.2	Phase 2: Selection of papers	117
A.1.3	Phase 3: Data extraction	118
A.2	MAB in Recommendation Systems	119
A.2.1	Works developed so far	119

A.2.2	The Evaluation Criteria	125
A.2.3	Facing RS challenges with MAB algorithms	128
A.3	Summary	131
Appendix B iRec: An Interactive Recommendation Framework		133
B.1	Libraries & Frameworks	133
B.2	The iRec Framework	135
B.2.1	Environment Setting	136
B.2.2	Recommendation Agent	138
B.2.3	Experimental Evaluation	140
B.3	Framework	142
B.3.1	Environment Setting	144
B.3.2	Recommendation Agent	145
B.3.3	Experimental Evaluation	146
B.4	Summary	149

Chapter 1

Introduction

This chapter introduces the dissertation presented in this work. The first section describes the research context by explaining the user’s feedback problem with contextual bandits, highlighting its challenges, and emphasising how it has been addressed in the last few years. Then, the second section points out the statement of the thesis and the main research questions raised in order to support our study. We intend to study how the uncertainty scenarios around the user’s preferences can influence the user’s long-term experience in the system. The main motivation is described in more detail in the third section. Finally, we present the main contributions and the complete outline of this work in the last two sections.

1.1 Background

Recommendation Systems (RSs) emerged at the beginning of the century to handle the amount of information available on online platforms of e-commerce and entertainment. Initially, such systems were proposed as simple filtering tools to identify what is more likely to be of user interest [181]. Then, after the years, the collective effort of industry and academia has resulted in an impressive number of algorithms, approaches, and advances in this field [67, 118, 263]. RSs became more complex approaches to make personalised recommendations in batches with their prior knowledge about user behaviours and/or relevant characteristics of the items. However, nowadays, RSs are no longer straightforward offline algorithms that make batch predictions according to business requirements. In current online systems, RSs have become responsible for guiding the entire user’s experience as a sequential decision model where the user continually interacts with the system in a feedback loop [261, 293].

The user feedback loop is an interactive scenario where the system recommends items to individual users while feedback on these recommendations is continuously observed. As illustrated in Figure 1.1, at each user’s interaction, the system should recommend one or more items, receive the user’s feedback (i.e., a click, a like, a play, and others), and then update its knowledge to the next trial [259]. The main goal is to learn

at each interaction in order to increase the system knowledge and maximise the user’s satisfaction in the long run. In this sense, some critical challenges must be addressed. First, how to rapidly learn a new user’s interest while not compromising his/her recommendation experience? Then, how to identify changes in the user’s preferences and relearn them over the interactions?

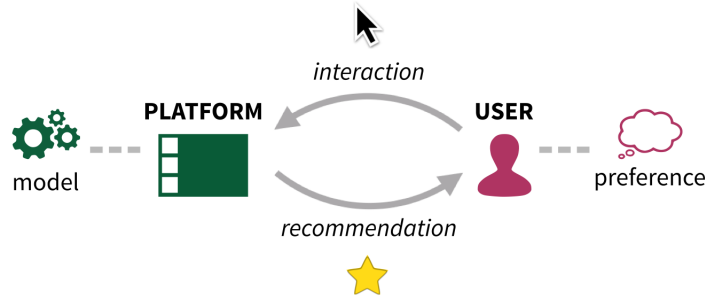


Figure 1.1: The feedback loop between a user and the recommendation system [50]. At each user’s interaction, the system recommends an item(s) and receives feedback from a user. The recommendation and the user’s choice are made according to the model implemented and based on the user’s preferences.

In the last years, researchers have answered such questions by balancing the goals of learning the user profile and providing accurate predictions as a *Multi-Armed Bandit* (MAB) problem [18, 85, 245, 257]. MAB is a classic problem of the Reinforcement Learning theory in which a fixed limited set of resources (arms) must be selected to maximise the expected gain (reward). Basically, at each trial, the system has to decide which arms to select, which order to select, and whether to continue with it or try a different option. In general, MAB models usually face the traditional dilemma between:

- (1) *exploiting* the arm that seems the best option so far; or
- (2) *exploring* an arm not yet tried out (or not tried enough).

Exploitation has notions of *being greedy* and a high probability of bringing a faster payback but can introduce a regret of missing unexplored opportunities. In turn, **exploration** usually has notions of *gaining info*, but it can take a long time to ensure enough knowledge for the model and may introduce a regret of wasting time on failures. The optimal solution is achieved by combining *information* and *greedy* gains.

In the recommendation field, items are usually modelled as the arms to be pulled and selecting an arm is equivalent to recommending an item [192]. The reward is the user’s feedback on that recommendation (e.g., clicks, acceptance, satisfaction, etc). Then, while exploitation means selecting items with a high probability of being rated, exploration means recommending different items in an attempt to gain more information about users [288, 192]. Similar to traditional recommendation scenarios, the item probability to be rated (a.k.a, the item utility) is defined by traditional RSs (like Collaborative Filtering) according to the user’s history of actions [133, 40, 18]. In turn, the exploration step is often

performed by random or uncertain choices like usual MAB solutions, such as ϵ -Greedy, Upper Confidence Bound (UCB), and Thompson Sampling (TS). Some of them occasionally explore items randomly (ϵ -Greedy algorithm). Other ones choose which options to explore by favouring actions with higher uncertainty because they can provide more information gains (UCB and Thompson Sampling algorithms) [133, 239, 281]. However, addressing the exp-exp dilemma while providing personalised recommendations remains an open challenge in the literature.

Current solutions have provided personalised recommendations in the interactive scenario by implementing Contextual Bandit models [266, 259]. In this case, a context (in the form of a feature vector) is revealed at each user's interaction and an arm must be selected based on it. The idea is similar to that used by traditional context-aware RSs but implemented in an online environment where the user will evaluate the recommendation and provide feedback (i.e., the reward) [133, 239]. Thus, the bandit model has worked for each user individually and provided personalised recommendations over the entire journey of each of them [129, 288, 257, 239]. However, despite the current advances, the personalisation level of such models still is directly related to the amount of information collected to represent the user's preferences. Moreover, unfortunately, there are several scenarios where the model may be **uncertainty about the users' preferences** over their journey in the system. A first example is the pure cold-start scenario where the user is completely new in the system and no information is previously provided [200, 17]. Another example happens when the system is not confident enough about its current knowledge of the user due to a preference change or even because the system has not defined the user's profile correctly [260, 174]. In both, the utility of any item (i.e., arm) for this user is not reliable enough to make good recommendations for them.

1.2 Thesis Statement

In light of this context, we aim to investigate the limitations of Contextual Bandits through the following thesis statement:

Recommending non-personalised items that enhance exploration and exploitation in scenarios of uncertainty where the information about the user is not straightforward or even does not exist increases the accuracy of Contextual Bandit models.

A closer look at the preceding statement introduces the main assumptions hereby adopted. It refers to the quality of bandit models in scenarios of uncertainty about the user's preferences. We hypothesise that Contextual Bandit models fail to handle the exploration and exploitation trade-off required to maximise the long-term rewards when

they have faced such scenarios. For this reason, such bandit models are limited to achieving per-user boundaries of accuracy in the long term. Especially in this work, we study two scenarios of uncertainty about the user’s preferences in the feedback loop problem: the first interactions (i.e., the pure cold-start problem) and the interactions after consecutive mismatches (i.e., when the system does not hit any recommendation for a while). In the first one, the total absence of user information creates a clear scenario of uncertainty where the system is not able to exploit or explore his/her unknown preferences. The second one refers to scenarios where the information about the user is not straightforward and the system keeps exploiting wrong assumptions that ultimately lead to unsatisfactory recommendations. In both, by assuming that bandit algorithms can always learn regardless of the items recommended, current Contextual Bandits have neglected the impact of such scenarios on the user’s journey. We believe these algorithms are wasting an opportunity to use such scenarios to gain more insights into the users’ preferences. In the first interactions or after successive mismatches, the users may not expect fully personalised recommendations. By implementing approaches that can acquire more knowledge about the users’ preferences in such scenarios of uncertainty, the accuracy of bandit models can be significantly improved over the long term.

In this sense, we raise three main research questions (RQ) to guide this work:

- **RQ1:** *How have current Contextual Bandit assumptions about uncertainty scenarios impacted the personalised recommendations?*
- **RQ2:** *How can exploration and exploitation be ensured in Contextual Bandit models when user preferences are uncertain?*
- **RQ3:** *What is the impact of enhancing exploration and exploitation when the Contextual Bandit model is uncertain about the user preferences?*

The first question (RQ1) aims to investigate the bandit models in the recommendation field in two different aspects: how they have been applied so far and what is the impact of their assumptions. In order to answer the first part of RQ1, we first perform a Systematic Literature Review (SLR) over the last 20 years of research in the field. A total of 1327 articles are examined and 230 are selected as the most relevant for this work (see Appendix A). Reading these articles, we notice that most state-of-the-art Contextual Bandit algorithms still define the utility of an item for a user by a linear combination of their features vector. The idea is similar to traditional Matrix Factorisation algorithms where users and items are represented in linear features to provide personalised recommendations [288]. However, such works do not address scenarios involving uncertainty around the user’s preferences. Researchers have assumed that only the exploration step of bandit models can deal with such scenarios. This work, in turn, studies the implementation details of current bandit algorithms and identifies a critical limitation to scenarios

of uncertainty. When bandit models face the scenario with the highest uncertainty (the pure cold-start problem), such models have performed as naive non-personalised RSs by:

- (1) selecting items randomly in an attempt to guess the users' preferences – a strategy of pure exploration; or
- (2) selecting the best items defined by the information known prior (similar to choosing the most popular items) – a strategy of pure exploitation; or
- (3) making a simple random exploration of the previous knowledge inferred by a Bayesian model – in an attempt to combine exploration and exploitation.

Then, by answering the second aspect of RQ1, this work studies the impact of the non-personalised approach in Contextual Bandit models. In this step, we only focus on the pure cold-start scenario where there is a clear scenario of uncertainty about the user preferences. Our analyses show that strategies based on pure exploration (like random ones) can get more knowledge into the model but it often requires many interactions until the users receive useful recommendations. On the other hand, pure-exploitation approaches often recommend potentially relevant items for users. However, these items do not add much knowledge to the model because most people usually like them (since they are the most popular). The result is a bandit model with high precision in the first user's interactions that cannot maximise long-term satisfaction. Thus, it is expected that a combination of the exploration and exploitation strategies may achieve better results by optimising both goals. However, a simple random strategy over the most popular items (as made by some of the current models) is also not enough to maximise long-term satisfaction.

Hence, the second research question aims to investigate how to ensure an effective combination of exploration and exploitation in such scenarios of uncertainty. Answering RQ2, we propose to apply concepts from the Active Learning (AL) theory. Traditionally, AL is applied to improve machine learning models by adding more feedback (i.e., true labels) [188]. In this sense, we propose to apply an AL strategy that combines the popularity and entropy of the items to get more information about the users without decreasing the model's accuracy. This strategy is selected since it does not use any information about the users and it might resemble the combination of exploration and exploitation even in scenarios of uncertainty. While using popularity means increasing the probability that a user will rate an item (i.e., exploitation), applying entropy means increasing the amount of possible information to be achieved if the user rates an item (i.e., exploration). In the first scenario of uncertainty, we create an initial context for new users based on the combination of popularity and entropy. The idea is to compel the bandit models to choose items based on this AL strategy in the first interaction. Thus, new users will start their journey by facing potentially relevant items that can also add some knowledge to the system. In turn, in the scenario of misleading assumptions, we introduce this AL strategy every time the recommender does not find a relevant item for consecutive times, such as 3, 5, 10,

or even 20 consecutive unsuccessful recommendations. The idea is to stop exploiting the wrong information for that user and try to learn their preferences again with a proper strategy. Since such strategies are not dependent on any specific information about the users, both modifications can be applied together in any Contextual Bandit model. In this work, we propose to apply our AL strategy in three distinct bandit algorithms – one for each class of the traditional MAB problem – Linear ϵ -Greedy [288], LinUCB [129], and the Particle Thompson Sampling [114].

Therefore, the third research question (RQ3) aims to measure the impact of this non-personalised Active Learning strategy in current Contextual Bandit models. To answer RQ3, we perform distinct experiments on three datasets from different recommendation domains (i.e., movies, songs, and books). First, we compare those three original bandit algorithms with their modified versions that add our AL approach in the first user’s interactions and after consecutive mismatches. This experiment highlights that applying AL in scenarios of uncertainty improves the quality of recommendations in the long run. The knowledge acquired by the model creates a chain reaction that results in our modified approaches outperforming the original ones. In the second experiment, we ensure that the existing bias of offline datasets has not caused this potential improvement. We perform a counterfactual evaluation using the Open Bandit Pipeline [189] to mitigate the exposure and selection bias from the user feedback problem [55, 165]. Results from different counterfactual estimators support our previous conclusions on an unbiased dataset. Then, in a third experiment, we compare our modified algorithms with baselines from three classes of algorithms related to this dissertation: (1) *State-of-the-art Contextual Bandits*, like LinUCB [129], Linear UCB, and GLM-UCB [288]; (2) *Meta-Learning approaches*, like NICF [297]; and (3) *Bayesian Inference methods*, like PTS [114], ICTRTS [245], and Cluster-Bandit [192]. Results show that our algorithms significantly outperform all baselines in the cumulative reward in the long run. All experiments were executed using the iRec, another contribution of this work (see Appendix B), a framework that enables the reproducibility of our entire evaluation process [209].

1.3 Motivations

First of all, the main motivation of this work consists in filling the lack of approaches to mitigate scenarios of uncertainty about the user’s preferences in Contextual Bandits. As aforementioned, we noticed that some scenarios of uncertainty have forced current bandit models to work as simple non-personalised algorithms that exploit the most liked items or even random items for all new users. Indeed, this idea of applying non-personalised approaches to make the first recommendations when the system does

not know anything about the target user is plausible. Over the years, these approaches have been applied in several e-commerce services and have achieved great results handling the user cold-start problem [203]. However, we assume that this naive representation may negatively influence the performance of such bandit models in the long run. In general, non-personalised recommendation systems are based on the same premise: *in the absence of prior information, the behaviour of a subset of new users tends to approximate the expected behaviour of its underlying population*. Although this assumption is valid for most users, other ones may have specific preferences and they may be not only interested in the most popular items. Nowadays, the diversity of users has become a striking feature in the current systems, since they house men, women, young people, adults or the elderly, without distinctions between age group, gender, social class, religion or any other [94]. Thus, items that appeal to a large portion of the population will not always satisfy all the preferences of the various existing users. Presenting unattractive items to new users makes these new users end up leaving the system without providing an expected financial return [170].

The second motivation is related to mitigating the user cold-start problem. Despite more than twenty years of academic research, it remains one of the hardest challenges of the field [22]. As all personalised algorithms have to correlate the user’s preferences to the items’ characteristics, the recommendation will not be successful if the system does not know any information about them [186, 26]. Similarly, most recommendation systems still fail to provide useful recommendations to new users in the industry. Indeed, this problem is usually recognised as one of the hardest challenges to achieve the desirable KPIs (Key Performance Indicators) for the business. This problem represents a crucial stage for the business since it provides the first impression for users and it is responsible for converting them into customers. In particular, in interactive systems, the cold-start problem might create even more impact on the recommendation system since the next interactions are predicted based on the first ones. As with any sequential model, the first recommendations might lead to wrong conclusions about the users’ preferences. Despite the most recent advances, most users keep leaving the systems after the first interactions [179]. The percentage of new users that become customers is low. And, unfortunately, current interactive models have neglected this challenge by assuming that systems would always learn the user’s preferences independently of the first items recommended.

Furthermore, our third motivation is related to the user’s short and long-term experience. We know that an unpleasing experience during the first user’s interactions in the system can lead to two other problems: (1) the user may have an unpleasant experience, being disappointed with the system; or (2) the system may model an incorrect profile about the user and start to recommend items that do not fit his/her real preferences. However, we do not know the impact of changing the first users’ interactions in the long-term user experience. As aforementioned, most Contextual Bandits usually show random items until they get all the user’s information required to make personalised recommendations.

It usually leads to a slow learning process, taking so many interactions to reflect positive rewards. Nevertheless, *what may happen if we change the first recommended items? If the learning stage in the initial interactions is faster than usual, what would happen with the user's long-term experience?* The studies published so far have been only concerned with new prediction rules or new exploration approaches. Since the first stage, neither work has been specifically concerned with the first recommendations (when the user is completely new to the system). In our opinion, the challenge should be related to selecting a small subset of potentially useful items to: (1) please the users at the beginning of their experience with relevant items; and (2) ensure enough knowledge about the user's preferences.

1.4 Main Contributions

The main contribution of this work is inherent to the user feedback problem and the contextual bandits. We identify room for exploration in the current representation of Contextual Bandit algorithms when facing scenarios of uncertainty about the user's preferences. Then, we propose a new solution that applies concepts from the Active Learning theory in this interactive recommendation field. This new approach outperforms all baselines and significantly impacts the model's accuracy in the long run. Hence, as main contributions of this dissertation, we highlight:

1. An extensive literature review about Multi-Armed Bandits in the recommendation field to provide an updated picture of the last 20 years.
2. The formulation of a problem in the way that current Contextual Bandit algorithms have handled scenarios of uncertainty about the user's preferences;
3. An empirical methodology that demonstrates the impact of naive non-personalised assumptions in the user's experience on current bandit models;
4. The proposal of applying Active Learning to properly address exploration and exploitation in scenarios of uncertainty about the user's preferences;
5. New algorithms based on Active Learning to enhance the user's long-term experience that outperforms strong baselines in the literature;
6. An empirical evaluation of bandit algorithms that simulate two scenarios of uncertainty – the pure cold-start and misleading assumptions;
7. A new framework for evaluating and implementing bandit models, named iRec, with all algorithms and methodologies used in this dissertation;

8. A counterfactual evaluation of our approach with a simulated dataset that confirms the usefulness of our approaches in an unbiased domain.

Furthermore, this work has resulted in six papers: two national papers, two international papers, and two international journals. The first papers were two national papers published in one of the most important Brazilian conferences on the Web – WebMedia (assigned as B1 by CAPES). In the first one, [208] have demonstrated the problem of Contextual Bandit algorithms in the pure cold-start problem. In the second one, [204] have presented a first approach to mitigate this problem – the basis of our current solutions. The first international one was published as a resource paper at the SIGIR conference. [209] introduce and explain the iRec framework created to allow the reproducibility of the dissertation experiments. Then, the other two international papers were published in two journals. The first one was published on the Expert Systems with Applications (assigned as A1 by CAPES) and [205] have presented our Systematic Literature Review. The other international journal was published in the first volume of the new ACM Transactions on Recommender Systems. [207] describe all the details about the Active Learning application in the first scenario of uncertainty – the pure cold-start problem. Finally, the second international paper [206] was published at the main track of the SIGIR conference (assigned as A1 by CAPES). This is the most recent paper about this work and contains the core of this dissertation, presenting our new solutions in both scenarios of uncertainty about user preferences.

1.5 Outline

The remaining of this work is organised as follows. In Chapter 2 we present the background concepts about Recommendation Systems and Multi-Armed Bandits. This chapter includes the main concepts, definitions, and approaches to both topics. It contains all the information required to understand this work and where it is located in the literature. Moreover, it also presents and discusses the two challenges related to the uncertainty about the user’s preferences. Then, each new chapter addresses and discusses one of the three research questions raised by this work.

First, in Chapter 3, we answer RQ1 by describing how the current bandit algorithms have been applied in the recommendation field. This discussion highlights the potential problem of current assumptions when the bandit model is facing a scenario of uncertainty about the user’s preferences. Then, in Chapter 4, we answer RQ2 by proposing a new approach to mitigate such challenges existing in current Contextual Bandit models. Our proposal consists of the application of Active Learning to create modified versions of three distinct bandit algorithms. Contrasting their original versions, these

new algorithms are able to address exploration and exploitation even in scenarios of uncertainty about the users. Finally, in Chapter 5, we answer RQ3 by demonstrating the potential gains of these modified versions in relation to their original versions and other strong baselines of the literature. In this chapter, we also make a counterfactual evaluation of our proposal to demonstrate that our gains are not biased by the current offline datasets from the literature. Chapter 6 presents the main conclusions, limitations, and future works related to this topic.

As additional chapters, we also present two appendices at the end of this work. Appendix A describes the Systematic Literature Review performed by this work to study and understand how the literature has discussed both topics together in the last 20 years. This work allowed us to consolidate an updated picture of the main research, highlighting the most used concepts and methods, their core characteristics, and their main limitations. In turn, Appendix B presents details about the iRec, an evaluation framework for recommendation systems in the interactive environment. It contains all the details required to understand this framework and the main guidelines to teach anyone how to apply it in their own research.

Chapter 2

Background Concepts

This chapter introduces the background concepts around the main topics discussed in this dissertation. First, we describe the general recommendation problem, the usual approaches applied to mitigate it, and the main challenges around them. Then, we present a short review of the main concepts of the Reinforcement Learning theory by formally describing the Multi-Armed Bandit (MAB) problem. This section highlights the main strategies applied to mitigate the MAB problem and simulate the online learning environment. Finally, we describe how such strategies have been applied to recommendation systems in recent years and the remaining open challenges. It contains all the information required to understand this dissertation and how this work fits in the literature.

2.1 Recommendation Systems

In the last three decades, the exponential growth of digital information on the Web has induced users into a stressful situation in which they do not know what to buy, listen to, or watch. Unintentionally, several e-commerce services have created hard scenarios for finding a relevant option just by offering thousands of distinct products at the same time [109]. Due to its drawbacks for their customers, this *information overload* has influenced several works to create a search mechanism to mitigate its impact since the 90s. One of the greatest examples is the Recommendation Systems (RSs), proposed as a new tool to provide personalised suggestions of items (e.g., movies, books, songs, etc) as output and guide users through the huge variety of options [202]. The idea is to identify the most relevant items for each user to improve their experience and also increase the business-related key performance indicators (KPIs), such as sales numbers or customer retention [45].

2.1.1 Recommendation Problem

The recommendation problem is defined as finding, among a potentially large number of items, those that better suit each user's individual interests in the system [26]. In this sense, at the most general level, any recommender system is designed to create a

certain *value* or *utility* for each user. Formally, given a set of users $U = \{u_1, u_2, \dots, u_m\}$, a set of items $I = \{i_1, i_2, \dots, i_n\}$ (e.g., books, movies, or songs), and the utility function $f(u, i)$ that maps the user u interest on the item i , the goal is to find an item $i^* \in I$ that maximises the utility function:

$$\forall u \in U, i_u^* \leftarrow \arg \max_{i \in I} f(u, i) \quad (2.1)$$

Initially, the utility function f was designed to return a *rating prediction*. This prediction refers to how much a user u should like an item i . Figure 2.1 shows an example of such a task, where the system must estimate a rating for the unrated items of the target user (the one circled by blue). User ratings are triples $\langle u, i, r_{ui} \rangle$ where r_{ui} is the value assigned – explicit or implicitly – by the user u to an item i . Usually, this value is a real number (e.g., from 0 to 1), a binary variable (e.g., like/dislike), or a value in a discrete range (e.g., from 1 to 5). Then, the prediction task consists of extrapolating the function f to the entire user and item space: $f : U \times I \rightarrow R$. Nowadays, however, researchers are more focused on creating a ranked list of items through a *top- k recommendation task*. In this task, the system must select a list of k items i_u^* to recommend for the user u . Thus, the function f is defined as $f : U \times L^* \rightarrow R$, where L^* is the set of all permutations up to the length k . Technically, this one can also use the same algorithms that were designed for the first task and then rank the items based on the predicted rating. Alternatively, other algorithms that do not consider the recommendable items individually but directly aim to optimise the ranking, can be used in this case – approaches of learning-to-rank or diversity-based methods.



			1	3		2	
	1	2	5		4		1
	4			3			
	?	3	?	?	5	4	?
		3	4		5		3

Figure 2.1: An example of a recommendation scenario with ratings from 1 to 5.

However, in both representations, recommendation still is a singular task in the field due to the human factor at the core of the recommendation. In these tasks, both the input signal and the prediction target consist of or involve user behaviour at their core. This brings about a specific level of complexity compared to, for instance, recognising shapes in an image or diagnosing a medical condition from medical tests [45]. Furthermore, recommendation is often not just about predicting people’s actions, but about enhancing (and hence changing) such actions by bringing awareness about potentially

better choices. In this sense, the recommendation problem is usually faced by the model illustrated in Figure 2.2. Similar to a usual search engine where the challenge is centred on the correlation between queries and documents, users and items are the main elements of a recommendation system [202]. While each user represents the query that guides the search mechanism, the items are the documents that should be retrieved. And, as a document must match the query searched, the items must match the user’s profile. This matching represents the correlation between the items and each target user. The system’s performance still directly depends on the available knowledge about users and items. Even the best matching approach will not be effective if the users and items are indexed differently.

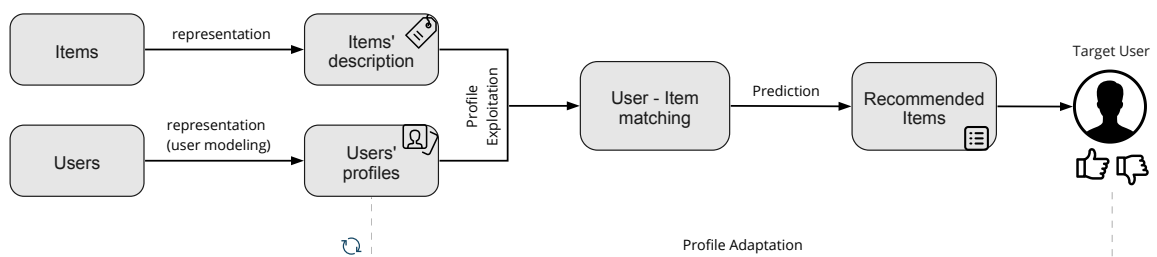


Figure 2.2: Standard model defined to mitigate the recommendation problem.

In general, the items’ representation is usually associated with their characteristics, like description, their target audience or even an embedding with all possible attributes. The item’s description is related to every single detail assigned by the system for that item in its domain. It can be textual as a movie’s synopsis and a song lyric, or categorical as the movie’s genres, the song categories, the product’s size, colour, or shape, and other examples. Despite their simplicity, these characteristics are plentiful in a real system. In scenarios where a product has just been released and the system does not know which users will like/dislike it, the best approach is to exploit its characteristics and recommend it with similar items [294].

On the other hand, there are two key elements when describing users’ preferences: the representation of their profiles and their maintenance. In the literature, several approaches represent user preferences for distinct domains. For instance, using the history of purchases in an e-commerce website, web usage mining (analysis of the links and time spent on a web page), listening habits (songs that a user listens to), and others. However, once the profile has been created, it does not remain static because the user’s interests might (and probably will) change. In this sense, a recommendation system should update the available knowledge about users according to their feedback for each recommendation. This feedback can be explicit when the system knows the user’s opinion or implicit when the system can only infer the user’s opinion. The first one usually comes in the form of positive or negative ratings in a discrete scale (e.g. from 0 to N) or a binary value

(like/dislike). Another way to gather explicit feedback is to allow users to write comments and opinions (i.e., reviews) about the items. In turn, the second type of feedback is usually gathered by monitoring the user's actions like the history of purchases, the time spent on a web page, the links followed by the user, the mouse movements, or analysing a media player usage (tracking the play, pause, skip and stop buttons). This one is more abundant than the explicit feedback but less accurate in representing the user's interests.

2.1.2 Recommendation Methods

In the literature, several works have proposed distinct approaches to handle the user-item matching process over the years. Especially, there are three main generations of recommendation systems, as illustrated in Figure 2.3.

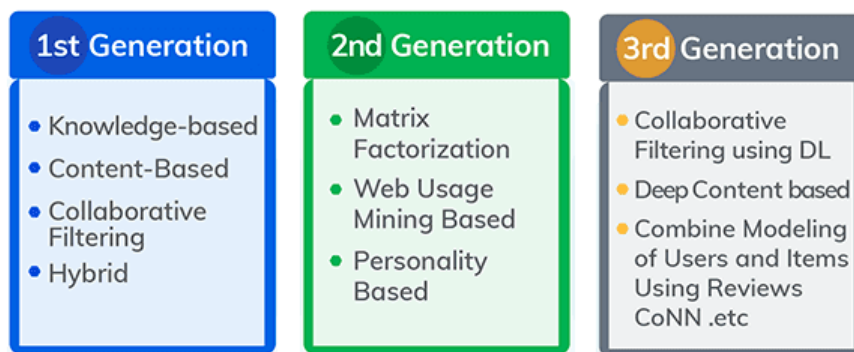


Figure 2.3: Recommendation system generations.

The first one emerged in the mid of 90s and it refers to the first strategies proposed in the recommendation field [233]. In general, these strategies assume a correlation between users and items with similar preferences and characteristics, creating the basis for all other generations of methods. The second generation emerged around 2005 when the researchers were especially motivated by the Netflix competition¹ to find new approaches and solutions for the recommendation problem. This generation was recognised by the matrix factorisation algorithms, opening new research directions for other personalised approaches that still are applied in real-world scenarios [5, 119]. Just recently, in the middle of the last decade, with the exponential growth of data around user behaviour by tracking their path in e-commerce and collecting impressions around each interaction, these algorithms were replaced with new approaches based on Deep Learning, Neural Networks, and others. This change marked the third generation of algorithms [74, 54, 97].

However, even the recent approaches are still based on the concepts defined in the first generation by the correlations between users and items with similar characteristics.

¹In October 2006, Netflix released a recommendation challenge by offering \$1,000,000 to those who improve at least 10% their movie recommendation system.

In this sense, we propose to review concepts around these main classes of recommendation systems that are the basis of all other generations of algorithms. In general, they are split into two classes of methods: personalised and non-personalised [202, 26]. While personalised methods usually exploit the available information about users to make recommendations that suit their profile, non-personalised methods only exploit the users' global opinion or items' global popularity to make the same recommendation for every user. The personalised ones are split into four main sub-classes: Content-Based (CB), Collaborative Filtering (CF), Demographic Filtering (DF), and Hybrid methods [26].

2.1.2.1 Content-Based

Formally, Content-Based methods estimate the utility function $f(u, i)$ for the item i to user u through a known function $f(u, j)$ defined by the same user u to an item j similar to i [233]. In other words, to estimate the user u rating to the item i , these methods first find similar items to i and then use this previous information to make the recommendation. Differently from other approaches, however, the similarity of items is estimated according to their characteristics, such as their description, category, gender, and others. It assumes that items with similar attributes will be evaluated similarly since users usually exhibit a correlated preference with the items' attributes [196, 186].

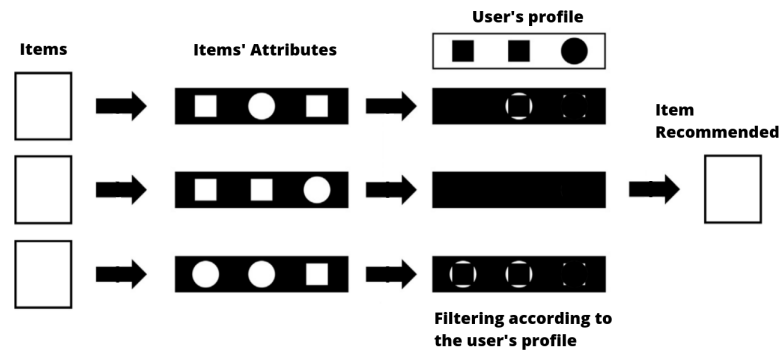


Figure 2.4: Standard recommendation engine of Content-based methods.

In these approaches, an item i is described as a vector $X_i = (x_1, x_2, \dots, x_n)$ of n attributes, which can be binary, nominal or numeric. In a movie scenario, for example, these attributes can be the actors, directors, genres, year of release, synopsis and others. Similarly, the user profile is modelled based on the attributes of items attributes rated by u in the past. It is formally described as a vector $Y_u = (y_1, y_2, \dots, y_n)$ where each element is a combination of the items' attributes. Thus, the recommendation is made by filtering all items in the domain to find the item whose X attributes match the user's Y profile. This process is illustrated in [26] as shown in Figure 2.4.

2.1.2.2 Collaborative Filtering

Collaborative Filtering methods assume that users (or items) with similar consumption histories would share common interests (or a common target audience). In this case, the utility function $f(u, i)$ is estimated based on (1) *users*: by the function $f(v, i)$ previously defined by another user v that shares the same interests than u ; or (2) *items*: by the function $f(u, j)$ previous defined by the same user u to another item j since j has the same attributes or share the same target audience than i [196, 26]. Besides this, these methods are also split into two subgroups according to the strategy used as shown in Table 2.1.

	Classes Methodology	
	Memory-based	Model-based
User-oriented	Combines the preferences of the k most like-minded users, with similar or correlated behaviour.	Correlates the user history of actions to predict new recommendations for the target user.
Item-oriented	Combines the ratings of the k most similar items, considering all users.	Correlates the past item ratings to predict new recommendations for the target user.

Table 2.1: Classes of collaborative filtering methods.

Memory-based methods explicitly measure the user or item similarity by using a correlation metric to make the recommendation [26]. The usual methods are often *item-based*, recommending based on the similarity of items (e.g., Item-kNN), or *user-based*, recommending based on the similarity of the target-user to other users (e.g., User-kNN). User-oriented Memory-based CF methods determine for each user a group of nearest neighbour users whose past ratings are similar, or highly correlated, with the user ratings. Scores for unseen items are predicted based on a combination of the scores known from the nearest neighbours. As these Memory-based CF methods are based primarily on clusters of users, their effectiveness depends on the generated clusters expressing high correlations between users. In the same way, Item-oriented Memory-based CF defines, for each item, a group of the most similarly evaluated items, considering all users in the domain. Later, scores for unseen items are derived from scores given for similar items by the target user of the recommendations.

On the other hand, Model-based CF methods learn a descriptive model of user preferences and then use it to generate ratings [26]. Many of these methods are inspired by machine learning, algebraic representation, statistical inference, deep learning approaches, neural networks or other techniques [245, 297]. Latent factor methods represent one of the most efficient and popular approaches in Model-based CF since they are generally effective at estimating overall structure that relates, simultaneously, most or all items [119]. These techniques have proven to be efficient in recommendation systems when predicting user preferences from known user-item ratings [60, 54]. Especially, in model-based approaches, users and items are considered an ensemble of multiple interests or aspects. Each user

$u \in U$ has a probabilistic membership in each of the aspects $z \in Z$ so that users in the same interest groups have similar tastes. Similarly, each item $i \in I$ has a probability to be interesting for the users in each aspect $z \in Z$. Thus, common model-based approaches usually represent the interest of u about $i \in I$ by the mix of these probabilities as follows.

$$P(i|u) = \sum_{z \in Z} P(i|z) \cdot P(z|u) \quad (2.2)$$

The first term $P(i|z)$ does not depend on the target user and represents how relevant the item i is for the group z . The second term $P(z|u)$ is the user-penalisation term also known as the user model. In current years, both have been addressed by latent feature vectors extracted from Probabilistic Matrix Factorisation (PMF) methods [184, 60, 54]. It factorises the rating matrix $M^{m \times n}$ into the product of two low-rank matrices $P \in \mathbb{R}^{m \times z}$ and $Q \in \mathbb{R}^{z \times n}$. While the matrix $P^{m \times z}$ contains the user-model θ_u , representing the multiple interests of each user u in the z groups, the matrix $Q^{z \times n}$ represents how relevant is the item i for the z groups. The recommendation is then redefined as the association of the users' features $p_u \in P$ and the items' features $q_i \in Q$ as follows: $s(i, u) = p_u^\top \cdot q_i$.

2.1.2.3 Hybrid Methods

In general, hybrid methods aim to explore the advantages of each traditional approach [5]. Most existing techniques can be classified as proposed by [150] and illustrated by [26] in Figure 2.5:

- (A) **Combining distinct approaches:** CF and CB methods are implemented separately and their predictions are combined by a quality function that determines the best recommendation to use [24, 116].
- (B) **Adding CB in CF:** an attempt is made to incorporate some characteristics of a CB into CF methods, such as by integrating content attributes into the user's profile (or items) in calculating the similarity function between users (or items) [157, 132, 104].
- (C) **Unified Model:** a model that incorporates both CB and CF strategies in a single model by combining content information with the users' history of ratings and their preferences [171, 62].
- (D) **Adding CF in CB:** it aims to incorporate some characteristics of CF in CB methods, such as dimensional reduction strategies applied to a large collection of existing content [159].

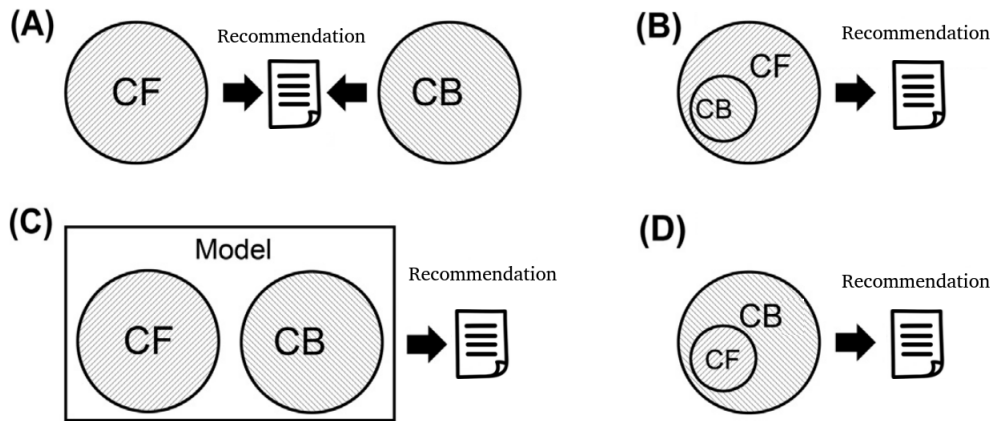


Figure 2.5: Standard definition of Hybrid methods.

2.2 Recommendation Challenges

Currently, most of the Recommendation Systems are semi-supervised strategies that explore the prior knowledge of users and items to generate personalised recommendations. Both Collaborative Filtering and Content-Based methods use the users' consumption history (i.e., past information) to model the profile of the target user. For this reason, the most critical challenges are related to the absence of information or the assumption of wrong preferences about users/items. Such scenarios still prefigure as outstanding problems for current models.

2.2.1 Cold-Start problem

The Cold-Start Problem is one of the most common issues that negatively affect the key performance indicators (KPIs) of many companies [25, 101]. It refers to the challenge of making personalised recommendations for new users or items that have few (or none) historical data [196, 5]. In the literature, there are three distinct perspectives on this problem. First, when a new user signs up for a service or they are an inactive user who has not interacted for so long time, the system may not have enough data to establish their preferences. This problem is known as *user cold-start problem* and it refers to the challenge of providing relevant recommendations when the user's profile is not well defined. Similarly, when a new item is added to the system, the data amount to understand its characteristics and how it relates to other items in the system is limited. This problem is known as *item cold-start problem* and it refers to the challenge of recommending items with few ratings. Then, a third problem happens when the system has only a set of items with few ratings to recommend for an inactive/new user. This problem is known as *user-item cold-start problem* [254] but this only happens in new systems and applications. However, the other two problems usually happen in most systems at some point in the user's journey and an

inaccurate approach to mitigate them may create a negative impact on the business.

In this sense, recent works have addressed the Cold-Start problem using several approaches [136]. The first proposals were based on Content-Based (CB) methods to replace the absence of ratings by the existing characteristics of the items/users [157, 137]. This approach is effective because the system usually has detailed information about the items, such as their genre, keywords, or attributes. Even the newest items released in the system contain such information that can be correlated with the other existing items [168]. Collaborative Filtering (CF) approaches, in turn, usually require a significant amount of historical data to correlate items/users with the same rating information. This may not be available to new items/users. In this sense, hybrid approaches that combine both CB and CF were also proposed to address the cold-start problem [197, 69, 195]. In this case, the system tries to combine the few ratings available with personal information about the users/items [218, 195]. Other approaches include knowledge-based and contextual-based recommendation systems. Both rely on user input to identify their preferences, constraints, location, or even their current mood or activities to make recommendations [76, 7, 110].

However, such methods still suffer from the **pure** cold-start problem. The terms *pure cold-start* and *cold-start* are often interchangeably used, but they can refer to slightly different aspects of the recommendation problem [203]. The *pure cold-start* problem typically refers to a scenario where a new user/item has no historical data available in the system – i.e., there are no past interactions or preferences for the user, or no metadata or other information about the item. The system has no information to learn the user’s preferences or the item’s quality and providing accurate recommendations is extremely hard. In turn, the term *cold-start* refers to a broader situation where the recommendation system has a limited amount of data to work with. It occurs in scenarios where there is some data available but it is sparse or incomplete, or where the system only has some indications about which items may suit the user’s preferences [267]. In general, the pure cold-start problem is considered the most challenging aspect of the recommendation problem, as it requires the system to make predictions based on very limited data. There is no perfect solution for this and researchers continue to explore new approaches to address these issues.

2.2.2 Misleading Assumptions

Misleading assumptions occur when the system has learned the wrong preferences about a user/item and it is unable to make satisfactory recommendations. This can happen in at least three distinct cases throughout the user’s journey. First, this problem may be associated with the traditional cold-start problem [25]. When a user only has a few interactions, the recommendation model may learn wrong preferences just because there is not enough information. Second, this problem may also happen when addressing users with dynamic preferences where their taste changes frequently or suddenly [260, 53].

In this scenario, the system may not rely on its past information or general user preferences. Then, the third scenario may happen when the system faces a shared account problem [236, 87]. This problem happens when more than one person uses the same account in the system because they share the same login information or when they have a family plan in the system. In this case, the preferences learned from one person may not be suitable to make recommendations for another one.

In the literature, such problems have been handled individually. The pure cold-start problem has been handled by knowledge-based and non-personalised methods as mentioned in the last section (see it for more details). In turn, the other two problems are discussed in this section. User Dynamic Preferences have been addressed as time or contextual-aware recommendation systems [141, 124]. Recently, researchers have addressed this problem by Deep Learning (DL) models [166, 252]. Wang et al. [246] apply a recurrent neural network to capture the session information in user interactions and predict the user's preferences in real-time. Similarly, Baral et al. [16] applies DL to capture the user's preferences changes and uses it as a context for the recommendation engine. Finally, other works also propose to explore the potential of using social network connections to capture user dynamic preferences [175].

On the other hand, the shared account problem has been addressed by distinct approaches related to the user's profile. In real-world systems, the most usual approach consists of allowing users to create multiple profiles within the same account [70]. This allows each user to have their own preferences and history, which can be used to provide personalised recommendations. Netflix is a good example of this approach. Other solutions explore different approaches related to user profiles. Some works have proposed to identify users who share the same account and treat them as a group [151, 152]. This has been done by analysing patterns in user behaviour, such as the items they interact with and the time of day they use the account. Since identified, each group can be treated as a single user for recommendation purposes. Other similar works have proposed to segment users based on their interests and preferences [236, 167]. This has been done by clustering users based on the items they interact with and the ratings they give. Since segmented, recommendations are tailored to each group. Finally, most recent approaches have proposed to rely on user feedback to improve their recommendations [224, 59]. In these cases, the recommendation system asks users to rate items or provide feedback on whether a recommendation was helpful. This feedback is then used to adjust the recommendations for each user over time.

In this work, however, these problems are summarised as the misleading problem. In our opinion, these three scenarios may happen when the system has made some wrong assumptions about a user and keeps exploiting it. If the current model deployed in a system leads to consecutive mistakes for one particular user, it may represent that this model is uncertain about the user's preferences. In this sense, a simple indicator for the misleading problem consists of counting the number of cumulative misses achieved by a

model for a user. Then, when this counter reaches a threshold previously defined, it should indicate that the model is not exploiting the correct preferences and that exploration is not helping the system to find better options. To the best of our knowledge, neither other work has proposed to handle this problem in this simple abstraction. This is a brief simplification of the aforementioned problems but our intention is not to solve them. They are only examples of how misleading assumptions may harm recommendation models.

2.3 Multi-Armed Bandits

The Multi-Armed Bandit (MAB) problem, sometimes called the K -armed bandit problem [287], is a classic problem in which a fixed limited set of resources (arms) must be selected between competing choices to maximise their expected gain (reward). The name ‘bandit’ comes from imagining a gambler at a row of slot machines in a casino, who has to improve his/her profit by maximising the sum of rewards earned through a sequence of lever pulls. Basically, at each trial, the gambler has to decide which machines to play, how many times to play each machine, in which order to play them, and whether to continue with the current machine or try a different machine. However, each machine provides a random reward according to its probability distribution. Then, the best way to solve this problem is to handle a crucial dilemma, deciding for the *exploitation* of the machine that has the highest expected payoff and the *exploration* of other machines to get more information about the expected payoffs.

Formally, the MAB is a sequential decision model represented by $\langle \mathcal{A}, \mathcal{R}, \mathcal{Q} \rangle$ where an agent has to continually choose an action $a \in \mathcal{A}$ for T trials among a set of actions \mathcal{A} ($|\mathcal{A}| = K$) in order to maximise the cumulative reward $\sum_{t=1}^T r_t$. In this case, $r_t = \mathcal{R}_t(a_t)$ is the reward achieved when an action a is performed. An MAB algorithm performs an action a at each trial t according to an action selection policy π . This policy follows a probability distribution, usually called the value function \mathcal{Q} , over each possible action a . The function \mathcal{Q} defines if an action a must be selected (or not) by measuring the expected reward – $\mathcal{Q}_t(a) = \mathbb{E}[r_t|a]$. Figure 2.6 illustrates this MAB definition by showing the agent and the environment continually interacting in a sequence of discrete time steps t . At each trial t , the agent samples an arm a_t and receives a reward r_t . The history of actions and rewards guides the agent selection for the time $t + 1$ and the other trials.

In a similar definition, it is possible to redefine the MAB main goal to minimise the regret associated with each action a chosen, as shown in Equation 2.3. Calling the arm with the highest expected reward at time t as the best arm, denoted as a_t^* , and its expected reward as the optimal reward r_t^* , the regret can be defined as the difference between r_t^* and the reward r_t achieved by the agent. Sometimes, it is also possible to

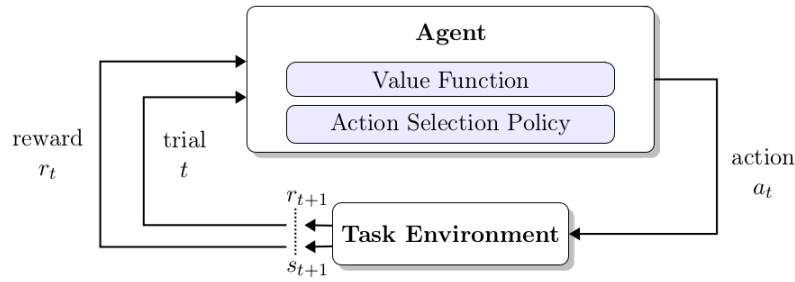


Figure 2.6: The traditional framework of Multi-Armed Bandits.

redefine the objective function according to the work goal by maximising the average reward returned, maximising the percentage of optimal action selection, minimising the number of trials without rewards, and so many others.

$$\text{Maximise } \underbrace{\sum_{t=1}^T r_t}_{\text{Reward}} \quad \equiv \quad \text{Minimise } \underbrace{\sum_{t=1}^T (r_t^* - r_t)}_{\text{Regret}} \quad (2.3)$$

In general, the quality of any MAB algorithm is usually related to the way it handles the exploration and exploitation dilemma in its action selection policy π . If the agent only performs the exploration, it will choose the actions randomly by ignoring all the knowledge achieved in the earlier steps. On the other hand, if the agent only performs exploitation, it will choose the actions according to the short-term reward similar to greedy approaches and, perhaps, never find the long-term reward. After all, each agent's decision has long-term consequences: each action influences the environment and determines what type of information the agent can observe to update its policy going forward [19]. In this sense, many MAB algorithms have been proposed in the literature with different properties [219]. The main algorithms are called as ϵ -Greedy [13], Upper Confidence Bounds (UCB) [13, 12], and Thompson Sampling (TS) [51]. As these algorithms can exploit the same value function $Q_t(a) = \mathbb{E}[r_t|a]$, the main difference between them is related to the exploration step. In terms of the exploration strategies, these algorithms can: (1) do no exploration at all, focusing on the short-term returns; (2) occasionally explore at random; or (3) choose which options to explore by favouring actions with higher uncertainty because they can provide higher information gain. These types of strategies can classify these MAB algorithms, as shown in Figure 2.7. The next subsections show more details about each algorithm.

<i>No exploration!</i> - Greedy algorithm	<i>Random exploration</i> - ϵ -Greedy algorithm	<i>Smart exploration</i> - Upper Confidence Bound (UCB) - Thompson Sampling
---	--	--

Figure 2.7: Distinct exploration strategies applied in each MAB algorithm.

2.3.1 ϵ -Greedy

The ϵ -Greedy algorithm handles the exp-exp dilemma by selecting the best action most of the time and doing a random exploration occasionally. The best action is usually estimated according to the experience by averaging the rewards associated with the target action a that was observed so far, as defined by Equation 2.4. The value function Q for each arm a in the trial t usually considers the mean of rewards achieved in the earlier trials $\tau < t$. $N_t(a)$ is the number of times that the action a was taken before the trial t . This exploitation step is performed with probability $(1 - \epsilon)$.

$$a_t^* = \arg \max_{a \in \mathcal{A}} Q_t(a) = \arg \max_{a \in \mathcal{A}} \frac{1}{N_t(a)} \sum_{\tau=1}^{t-1} r_\tau \quad (2.4)$$

Otherwise, the algorithm performs the random exploration uniformly with probability ϵ . The main idea behind this step is to guarantee the algorithm to not get stuck in a suboptimal reward forever. However, the challenge is to define the ϵ . In general, this parameter gives a poor performance at the extremes. If it is too small, the learning ability defined by the exploration is slow at the start, and the algorithm will be slow to react to changes. In turn, if it is too big, the algorithm will waste many trials pulling random arms without gaining much. The best parameter should allow the algorithm to choose the best action for a large proportion of the time. Unfortunately, due to the randomness, the algorithm may end up exploring a bad action which the agent has already confirmed in the past. To avoid such inefficient exploration, there are two main approaches available. The first one is to decrease the parameter ϵ in time. The second and most usual one is to be optimistic about options with high *uncertainty* and, thus, to prefer actions for which the agent has not had a confident value estimation yet. This kind of exploration is considered smarter than the other because it favours the exploration of actions with a strong potential to have an optimal value. It is the main base of the other two strategies usually applied by MAB algorithms.

2.3.2 Upper Confidence Bound (UCB)

The UCB algorithm measures the function value $Q_t(a)$ by considering the confidence bound of the reward value, called $\mathcal{C}_t(a)$, so that the true value is below with bound $Q_t(a) \leq \mathcal{Q}_t(a) + \mathcal{C}_t(a)$ with high probability. The main idea is to apply the upper bound to measure the potential of each action (arm) according to the uncertainty about its quality. Then, the agent always selects the greediest action with the highest UCB:

$$a_t^* = \arg \max_{a \in \mathcal{A}} \mathcal{Q}_t(a) + \mathcal{C}_t(a) \quad (2.5)$$

In the literature, there are several approaches to measure the confidence bound $\mathcal{C}_t(a)$ [163, 129, 133, 272]. In general, $\mathcal{C}_t(a)$ is defined as a function of $N_t(a)$, where a larger number of trials $N_t(a)$ is directly related to a smaller bound $\mathcal{C}_t(a)$. The traditional UCB algorithm measures $\mathcal{C}_t(a)$ by Hoeffding's Inequality [72], a theorem to any bounded distribution. Applying it, these works found that the probability of the expected reward being greater than the confidence bound is very small: $\mathbb{P}[Q(a_t) > \widehat{Q}_t(a) + \mathcal{C}_t(a)] \leq e^{-2t\mathcal{C}_t(a)^2}$. In this sense, the main works define a very tiny threshold p and apply it to measure the confidence bound, as follows: $\mathcal{C}_t(a) = \sqrt{-\log p / 2N_t(a)}$. One current heuristic aims to reduce the threshold p in time t to propose a parameter-free algorithm. The famous UCB1 algorithm [13] is proposed setting $p = (t - 4)$ to make a more confident bound estimation with more rewards observed. Its algorithm performs as follow:

$$\mathcal{C}_t(a) = \sqrt{\frac{2 \log t}{N_t(a)}} \quad \text{and} \quad a_t^* = \arg \max_{a \in \mathcal{A}} Q_t(a) + \sqrt{\frac{2 \log t}{N_t(a)}} \quad (2.6)$$

Furthermore, other works have also explored the prior distribution of rewards by modelling the expected mean reward as a Gaussian and setting the upper confidence bound \mathcal{C} based on the standard deviation. These approaches are called Bayesian UCB and they are able to make better-bound estimation [145, 249]. A traditional implementation draws the reward distribution according to a prior distribution Beta with parameter θ and sets $\mathcal{C}_t(a) = \alpha \cdot \sigma_t(a)$, where α is an adjustable hyperparameter and σ is the standard deviation of how many times a was chosen.

2.3.3 Thompson Sampling (TS)

Distinct from the other approaches, the Thompson Sampling algorithm implements the idea of probability matching. At each time step, the main idea is to select action a according to the probability that a is optimal according to the history of actions h_t already known by the agent:

$$\pi(a|h_t) = \mathbb{P}[Q(a) > Q(a'), \forall a' \neq a|h_t] = \mathbb{E}[a = \arg \max_{a \in \mathcal{A}} Q(a)] \quad (2.7)$$

Nowadays, the main TS algorithms are based on the Bernoulli bandit where they naturally assume that $Q(a)$ follows a Beta distribution by defining $Q(a)$ as the success probability θ in Bernoulli distribution. In general, the value of $\text{Beta}(\alpha, \beta)$ is within the interval $[0, 1]$ and the parameters α and β correspond, respectively, to the counts when the agent achieves success or failure to get a reward. Then, at each time t , the agent must sample an expected reward, $\tilde{Q}(a)$, from the prior distribution $\text{Beta}(\alpha_i, \beta_i)$ for every action. The best action is selected among the samples: $a_t^* = \arg \max_{a \in \mathcal{A}} \tilde{Q}(a)$. After the true reward is observed, the Beta distribution is updated accordingly by doing Bayesian inference. If

the action selected is correct (i.e., $a_t^* = a_i$), then α is updated. Otherwise, the β is incremented. This simple idea has worked very well in many scenarios [51]. However, for several practical and complex problems, estimating the posterior distributions with observed true rewards can be computationally expensive. In this case, it is necessary to approximate the posterior distributions using methods like Gibbs sampling, Laplace, and Bootstraps.

2.4 Summary

In this chapter, we present all the information required to understand this dissertation in three main sections. In the first one, we discuss the background concepts of recommendation systems. The recommendation problem is defined as a prediction and/or ranking task where the system should maximise a utility function related to the user's preferences. Then, we present the usual recommendation modelling to address the item's characteristics and the user's preferences in the system. Their representation is essential to ensure the best performance of the user-item matching process – the core of the recommendation problem. In the literature, there are distinct methods and strategies to make this matching. The following subsection groups these methods by their traditional taxonomy of Content-Based Filtering, Collaborative Filtering, and Hybrid.

Then, in the second section, we discuss the usual challenges in the recommendation task. Especially, this work highlights two of them: the cold-start and the misleading assumptions. The first one happens when there is not enough (or even no) information to represent the users and items. The second one happens when the recommendation model learns wrong assumptions about the user's preferences and exploits them without success for a while. Such problems are the core of this work where we discuss their impact on interactive scenarios represented by Multi-Armed Bandit models.

Finally, in the third section, we discuss the main concepts around the Reinforcement Learning theory. First, we highlight the Multi-Armed Bandit problem by formally describing it as an interactive scenario where the system must maximise the reward expected (or minimise the regret). Then, we present the three main approaches usually designed to mitigate this problem: ϵ -Greedy, UCB, and Thompson Sampling. In general, they are different in terms of the exploration stage. Some do not perform any exploration, others occasionally explore randomly, and others choose to explore by favouring actions with higher uncertainty. In the next chapter, we discuss how these approaches are applied in the recommendation field through Contextual Bandits and what issues are related to their assumptions.

Chapter 3

Contextual Bandits & the user's preferences uncertainty

This chapter investigates the first question raised by this work: *How have current Contextual Bandit assumptions about uncertainty scenarios impacted the personalised recommendations?* First, we present the background of Contextual Bandits and their assumptions in the users, items, and rewards representation. Then, we discuss the potential effects of these assumptions in scenarios of uncertainty about the user's preferences. To support this discussion, we perform an experiment with three bandit algorithms in distinct domains. This experiment simulates the behaviour of such algorithms when facing the scenario with the highest uncertainty: the pure cold-start problem.

3.1 Contextual Bandits

In the literature, contextual-aware systems consist of algorithms that usually apply some information about items (e.g., categories, genres, etc.) or users (e.g., age, gender, etc.) to improve the quality of the recommendations [26]. The idea is to take different contextual attributes to capture user preferences more correctly than only rating-based algorithms. These attributes can be related to the user, item, season, weather, or others and they can directly guide the system recommendations [6]. In general, authors have presented three different approaches to include context:

- (1) **prefiltering**: context is used to select some set of data and then predict as usual;
- (2) **posfiltering**: ratings are predicted and the results are filtered using the context;
- (3) **modelling**: the context is used inside the RS by an embedding representation.

In general, the modelling approaches remain the best ones to provide contextual recommendations because they can identify non-trivial relations between users and items [26]. In this case, users and items are modelled as vectors of features according to an

algebraic or probabilistic representation [133, 239, 291]. Despite the recent advances, the most traditional methods explored over the years still are based on Matrix Factorisation and Probabilistic Matrix Factorisation [26, 271, 44]. In this case, a rating matrix $M^{m \times n}$ is decomposed into two low-rank matrices: $P \in \mathbb{R}^{m \times z}$ and $Q \in \mathbb{R}^{z \times n}$. While $P^{m \times z}$ contains the user-model p_u , representing the multiple interests of each user u in z features, the matrix $Q^{z \times n}$ represents how relevant an item i is for the z features. Then, the relevance \tilde{r} of an item i for a user u is estimated by the combination of the users' features $p_u \in P$ and the items' features $q_i \in Q$ as follow:

$$\tilde{r}(i, u) = p_u^\top \cdot q_i \quad (3.1)$$

Contextual bandit algorithms have applied this modelling approach to represent the context and applied traditional concepts of MAB in recommendation systems. Items are typically modelled as arms to be pulled and selecting an arm is equivalent to recommending an item in a specific context [192]. The reward is the user's feedback on that recommendation (e.g., clicks, acceptance, satisfaction, etc.). At each trial, the model has to decide which arms to select and whether to continue with it or try a different option. Classical examples of such approach are *LinUCB* [129], *FactorUCB* [239], *hLinUCB* [238], and *CoLin* [257]. *LinUCB* was proposed to personalise some news to users by exploring the context represented by demographic, geographical, and behavioural information about them. Similarly, *FactorUCB* also adds users' social influence to improve the algorithm's convergence rate. In turn, *CoLin* [257] is a linear bandit model based on Collaborative Filtering concepts that explores user dependencies. This dependency is represented by a graph that keeps the affinity between users to estimate the parameters of the bandit model. More recently, *hLinUCB* [238] was proposed to overcome the previous one by aggregating the user's information in latent factors associated with some observable contextual information.

However, in several real-world scenarios, contextual information is not always available to the system due to the users' privacy concerns. In recent years, several countries have approved a law of data protection that forbids systems to collect any personal information about users without their previous authorisation. Moreover, users may log into the system from incognito navigation or even without allowing the cookies, and forget (or ignore) to fill all forms about their personal information. In this sense, such factors have limited the applicability of these methods over the years and highlighted other approaches that do not apply sensible information. Algorithms like *Linear UCB* [288] and *GLM-UCB* [288] are more applicable since they construct the context only based on the latent factors of a Probabilistic Matrix Factorisation formulation. Both are UCB-based algorithms from the original *LinUCB*. Their difference is in the prediction rule. Furthermore, *GLM-UCB* also enables the applicability of non-linear representations to estimate the reward and performs a time-dependent exploration process.

Thus, even these recent algorithms remain following the usual prediction rule defined in Equation 3.1 to combine the users and items feature vectors [266, 239]. Traditional MAB approaches, such as the ϵ -Greedy, UCB, and Thompson Sampling, have been adapted for this objective function [129, 51, 288, 98, 239]. Indeed, after conducting a detailed reading of the works related to contextual bandit algorithms in our SLR (see Appendix A), we identified the main patterns usually applied to implement a linear representation of each MAB algorithm. These implementations are illustrated in Table 3.1. All algorithms usually start by representing users and items into a linear features vector from the historic of actions h and finish by updating these representations at each iteration t . The difference between them is related to the way they control the exploration-exploitation dilemma. While the ϵ -Greedy performs a random choice with probability ϵ , UCB and Thompson Sampling perform exploration by measuring the uncertainty Σ around the information available about users and items.

ϵ -Greedy	UCB	Thompson Sampling
<ul style="list-style-type: none"> - Estimate $p_{u,t}$ based on h_t - With probability $(1 - \epsilon)$: $i_t^* = \arg \max_{i \in \mathcal{I}} (p_{u,t}^\top \cdot q_i)$ - Otherwise: pick i_t^* randomly - Receive the reward $r_{u,i}$ - Update h_t based on $r_{u,i}$ 	<ul style="list-style-type: none"> - Estimate $p_{u,t}$ based on h_t - Estimate $\Sigma_{u,i}$ by h_t and $\{q_i i \in \mathcal{I}\}$ - Choose the item: $i_t^* = \arg \max_{i \in \mathcal{I}} (p_{u,i}^\top q_i + \Sigma_{u,i})$ - Receive the reward $r_{u,i}$ - Update h_t based on $r_{u,i}$ 	<ul style="list-style-type: none"> - Estimate $(\mu_{u,t}, \Sigma_{u,t})$ from h_t - Estimate $(\nu_{i,t}, \Psi_i, t)$ from h_t - Sample $\tilde{p}_{u,t} \sim \mathcal{N}(p_u \mu_u, \Sigma_u)$ - Sample $\tilde{q}_{i,t} \sim \mathcal{N}(q_i \nu_i, \Psi_i)$ - Choose the item: $i_t^* = \arg \max_{i \in \mathcal{I}} (\tilde{p}_{u,t}^\top \cdot \tilde{q}_{i,t})$ - Receive the reward $r_{u,i}$ - Update h_t based on $r_{u,i}$

Table 3.1: The typical linear implementation of the three main MAB algorithms.

3.2 User uncertainty in Contextual Bandits

Contextual Bandits have been extremely effective in the recommendation task since they can self-learn according to each context at each user’s interaction [36, 107]. As aforementioned, even the most recent approaches have explored the linear representation of users and items to capture non-trivial relations between them. The disadvantage, however, is in the absence of data, where the system would not have enough instances to represent these features ideally. Especially after the recent advances in textual processing and categorical interpretation to better represent the item’s metadata, the challenge has been centred on the absence of information about the users [77, 241, 101].

During the user’s journey into an interactive system, there are at least two critical challenges related to the lack of information. These challenges are illustrated in Figure 3.1

regarding the user’s temporal line in the system. The first one is known as Cold-Start and it happens two times in the first user’s interactions. It is hardest in the first interaction ($T = 0$) when the user is completely new or when they log in without identifying themselves. In this case, the literature refers to it as a Pure Cold-Start problem. The system does not have any information about the user and it cannot provide personalised recommendations [101, 203, 200]. After some interactions, the system knows a little bit more about the user but it is not enough to define their profile ($T > 0$). This scenario is named Cold-Start [26, 219, 102]. In turn, the second critical challenge happens when the system has already learned a profile ($T \gg 0$) for the user but its modelling is not well defined. In some cases, the system could have made wrong assumptions about the user’s preferences and it is now exploiting a wrong profile. In another situation, the user’s preferences may have suddenly changed and their profile is not correct anymore [260, 174].

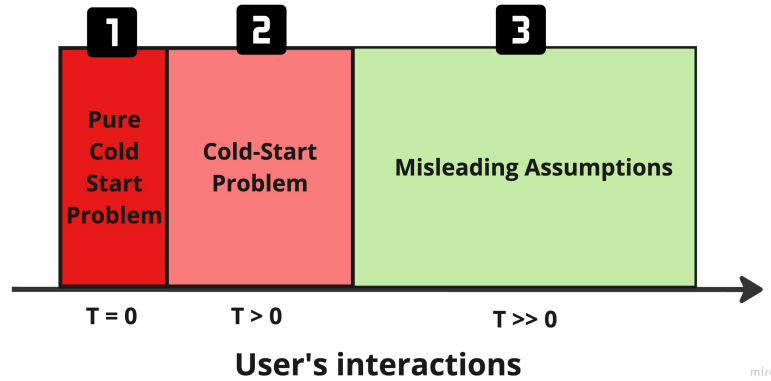


Figure 3.1: Distinct challenges about the user’s preferences in their journey.

In both challenges, the utility of any item (i.e., arm) for this user is not reliable enough to make personalised recommendations for them. Thus, mitigating such challenges is crucial for the success of any interactive RSs [219, 102]. However, our extensive literature review (see Appendix A) revealed that recent works in interactive scenarios have underestimated such challenges. By assuming that the bandit model can always learn regardless of the items recommended for each user, neither of them has proposed approaches to mitigate such challenges.

3.2.1 Pure Cold-Start

In the *pure* cold-start problem, bandit models are unable to define the contextual features vector for each user. They have assumed that the new user has no preferences for any item or feature. In practice, it means that such models have been initialising the contextual vector with a constant $c \geq 0$ for all features available. Thus, an item is predicted by using this contextual vector to represent the user’s features preference p_u and following the traditional rule of $p_u^\top q_i$, where q_i represents the features of each item i . In

this sense, we figured out that a simple initialisation of p_u with constant values may lead to an undesirable bias in the recommendations. If this value is 0, a usual approach in current implementations, the user feature vector p_u will lead the prediction rule to be equal to 0 regardless of the item feature vector. It results in a uniform selection of items (i.e., random or based on the items' IDs) with the same relevance score for this user – $\{s(I, u) = 0 : \forall i \in I\}$. In turn, if this value c is a constant $c > 0$, the items' features vector will become the sole relevant term of the objective function and the relevance score will follow its distribution – $\{s(I, u) \sim q_i : \forall i \in I\}$. It results in a fully biased recommendation towards the most representative items previously identified through the linear representation.

$$\begin{aligned} & \text{PREDICTION RULE: } i^*(t) = \arg \max_{i \in I} p_u^\top q_i \\ (t = 0) \quad & \left\{ \begin{array}{l} p_u = \{0 : z \in Z\} \rightarrow i^*(0) = \arg \max_{i \in I} \mathbf{0} \cdot q_i \quad (\text{random}) \\ p_u = \{c : z \in Z\} \rightarrow i^*(0) = \arg \max_{i \in I} \mathbf{c} \cdot q_i \quad (\text{biased}) \end{array} \right. \end{aligned}$$

In other words, most Contextual Bandits are usually compelled to take one of these two options when facing a new user:

- (1) a pure exploration – recommending items randomly; or
- (2) a pure exploitation – recommending items biased by the current knowledge.

The first option assumes that users are willing to interact for a long time and the algorithm is concerned with learning as much about the users' preferences. In turn, the second one assumes that users can leave the system after a few interactions and, thus, the system has to recommend items potentially relevant for users as soon as possible. Indeed, both assumptions are highly relevant. Nevertheless, here, they represent the well-known dilemma between exploration and exploitation from a new point of view: in the first users' interactions. Should we allow users to make their own choices and thus possibly lose the opportunity to sell something? Or, should we assume that they are impatient and recommend the best options, losing the opportunity to learn what they like? In this work, we believe that both options must be addressed to improve the user's experience in the system.

3.2.2 Misleading Assumptions

Misleading assumptions occur when the system has learned the wrong preferences about a user and it is unable to make satisfactory recommendations. This problem may happen in at least three distinct situations throughout the user's journey after the first interactions. First, the model faces the traditional cold-start problem where a user only has a few interactions and the model is unable to learn their preferences [9]. Second, when the user has dynamic preferences where their taste changes frequently or suddenly [260, 53].

And, third, when facing the shared account problem whereby multiple users use the same account and their behaviour is considerably different [236]. The absence of data or distinct behaviours may lead the model to fail the next relevant items for the user.

In general, such uncertainty scenarios have been underestimated in Contextual Bandits due to the model’s capacity to handle them by exploring new items and enhancing its known preferences. Researchers have assumed that by utilising user feedback at each interaction, any misleading assumptions would eventually be rectified. Although this is indeed true, the relearning process for individual user preferences is often time-consuming. Furthermore, the learning process of most models is primarily focused on identifying which arms have more potential to be relevant, rather than on understanding the user’s preferences. While this approach is generally effective in situations where the model can attempt different actions repeatedly, it may not be suitable for interactive recommendation systems where time is crucial in preventing user churn. In this work, we believe that addressing such scenarios may help bandit models respond more quickly in these uncertain situations.

3.3 Practical Outgrowths

In this section, we measure the impact of scenarios of uncertainty in the behaviour of the three traditional models presented in Table 3.1. First, we inspect the usual implementation of each algorithm when they have to handle one scenario of uncertainty. Then, we demonstrate the impact of such naive choices on algorithm behaviour. For this investigation, we only selected the Pure Cold-Start problem since this scenario represents the moment with the highest uncertainty about the user’s preferences. Moreover, it is guaranteed that this scenario will always happen regardless of the recommendation domain. In short, we have noticed that the same algorithm may perform completely differently according to the way they were implemented to handle this scenario of uncertainty.

3.3.1 Algorithms implication

First, we analyse the prediction rule of traditional Contextual Bandit algorithms:

(1) **ϵ -greedy algorithms** usually measures the item’s relevance by the product of features vectors p_u and q_i with a probability $(1 - \epsilon)$: $i_t^* = \arg \max_{i \in \mathcal{I}} (p_{u,t}^\top \cdot q_i)$. Thus, its predictions would change according to the constant value c used to initialise the user’s features vector p_u . If $c = 0$, the algorithm will perform a pure exploration of the items by recommending them randomly. Otherwise, when $c > 0$, it will perform pure exploitation of the items’ features (i.e., the vector q). The score of each item will be similar to its q_i value, biasing the entire prediction rule.

(2) In turn, **UCB algorithms** implement a confidence bound to represent the system uncertainty over the items and users. This confidence bound is usually measured by the combination of the item’s features vector q_i with uncertainty over the user at that trial t , named $\Sigma_{u,t}$. This component Σ is initialised as a simple identity matrix $I^{z \times z}$ and it is updated at each trial t . It sets the uncertainty component as:

$$i^*(t) = \arg \max_{i \in I} p_{u,t}^\top q_i + \sqrt{\mathbf{q}_i \cdot \Sigma_{u,t} \cdot \mathbf{q}_i^\top} \quad (3.2)$$

Thus, even if $c = 0$, the vector q_i will guide the recommendation due to the confidence bound associated with the prediction rule. In other words, these algorithms will always perform a pure exploitation of the previous item’s knowledge:

$$\text{if } p_u = \{0\}, \text{ then: } i^*(t) = \arg \max_{i \in I} \sqrt{q_i \cdot \Sigma_{u,t} \cdot q_i^\top} \quad (3.3)$$

(3) On the other hand, **TS algorithms** usually estimate the features vectors by sampling $\tilde{p}_{u,t} \sim \mathcal{N}(p_u | \mu_u, \Sigma_u)$ and $\tilde{q}_{i,t} \sim \mathcal{N}(q_i | \nu_i, \Psi_i)$ from the feature distributions usually extracted by a Probabilistic Matrix Factorisation (PMF) method. However, when the user is new and there is no information about their preferences, the system must initialise $\mu_{u,t}$ and $\Sigma_{u,t}$. The variance Σ is the uncertainty around that user and it starts from an identity matrix I . In turn, the mean μ is measured based on the items rated by that user and it is initialised by the constant values c . In this case, regardless of the value assumed by c , the algorithm will sample p_u from a normal distribution centred in c . Thus, at each iteration, TS algorithms will estimate p_u by sampling it from the prior distribution. As p_u can assume any value around c , the prediction will be defined by the linear product of these ‘random’ values and the q_i vector. Thus, these algorithms can, in a certain way, combine exploration and exploitation to perform the first recommendations. However, we assume that is not the best way to perform this combination since it will be created by random samples combined with the current system bias (the q_i vector).

In this sense, the behaviour of each bandit algorithm is contingent on how it is implemented to handle scenarios of uncertainty. A naive initialisation of the user’s feature vector can have a significant deviation from their expected results.

3.3.2 Item features bias

As aforementioned, when facing scenarios of uncertainty, Contextual Bandit models are compelled to choose between two main options: exploiting the current system knowledge or making an exploration of random items. In this sense, the idea of performing recommendations based on the previous system knowledge is not bad. However, this knowledge may be biased by the user’s selection and the most popular items. There-

fore, we select and analyse the correlation between these items’ feature vectors and the most popular bias in three distinct domains: movies, books, and songs. Each domain is represented by one of the datasets highlighted in Table 3.2¹.

Datasets	# Users	# Items	Sparsity
Netflix	10,000	17,372	98.69%
GoodBooks	53,423	10,000	98.88%
Yahoo Music R1	10,000	13,214	99.22%

Table 3.2: Datasets from three different recommendation domains.

In this experiment, we aim to investigate whether the current knowledge exploited by current Contextual Bandits is biased or not. First, for each dataset, we select all the data available and measure the items’ popularity and the items’ features vector. Popularity is the number of distinct users who rated the item. In turn, the item features vector is extracted from a Singular Value Decomposition (SVD) – an approach usually applied in Contextual Bandits. The SVD is applied over the entire dataset by using 10 eigenvalues and it returns the factorised matrices: (1) $P^{m \times z}$, representing the users’ features in z latent features; (2) $S^{z \times z}$ with the eigenvalues extracted; and (3) $Q^{z \times n}$, representing the items features in z features. Then, to compute the correlation between the popularity and the features vectors, we aggregate the vector q_i into one single number for each item. This value represents the item’s importance overall features by performing $q_i \cdot q_i^\top$. After that, we plot the values achieved for each item in Figure 3.2. This result highlights that the item features vector q_i is extremely biased to the most popular items. In other words, when the recommendations are based on the items feature vector q_i , the recommended items will be similar to the most popular ones.

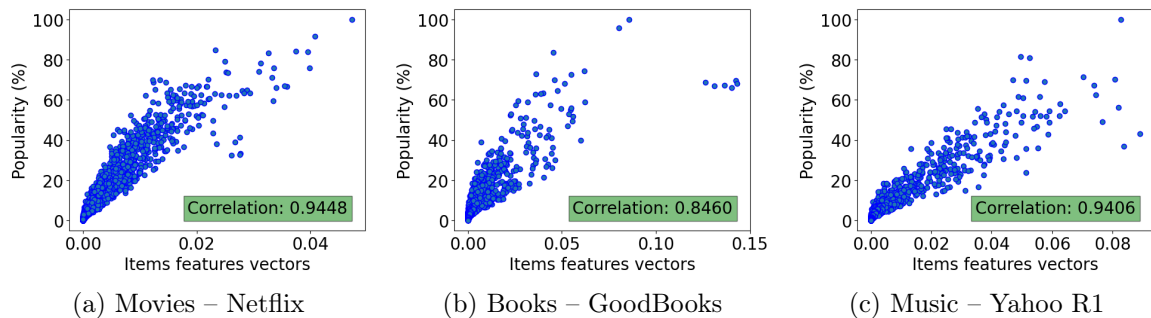


Figure 3.2: Popularity versus the items features vector importance measured by $q_i \cdot q_i^\top$. There is a significant correlation between the items features vector of Matrix Factorisation models and the most popular items.

¹They are available at: <https://www.kaggle.com/datasets/zygmunt/goodbooks-10k>, <https://webscope.sandbox.yahoo.com/catalog.php?datatype=r>

3.4 Impact on Recommendation Systems

The idea of exploiting non-personalised methods to face scenarios of uncertainty about the user’s preferences is plausible. Indeed, such approaches have been applied in different e-commerce [203]. However, to the best of our knowledge, there is not any work that has studied the impact of such methods in interactive recommendation systems. In this sense, we propose an evaluation methodology to highlight this impact in three domains and point out potential new approaches. This experiment is mainly focused on the pure cold-start problem where there is a clear scenario of uncertainty that is easier to measure.

3.4.1 Evaluation Methodology

In order to measure the impact of interactions performed over naive methods in the performance of an interactive recommendation system, we created an evaluation methodology as illustrated in Figure 3.3. This methodology consists of two interactive systems (or two stages) – one on the left and another one on the right. The first one contains a non-personalised recommender that will interact with the user for some time by providing popular or random items. Then, the second one contains a bandit algorithm that will apply the knowledge created by the first system (the grey area in the middle of the Figure) to make personalised recommendations. The main idea is to analyse the bandit performance (second system) based on the knowledge achieved by the first system (the non-personalised one). In this sense, the performance of the second system will reflect the impact of the first recommendations over the entire user’s journey.

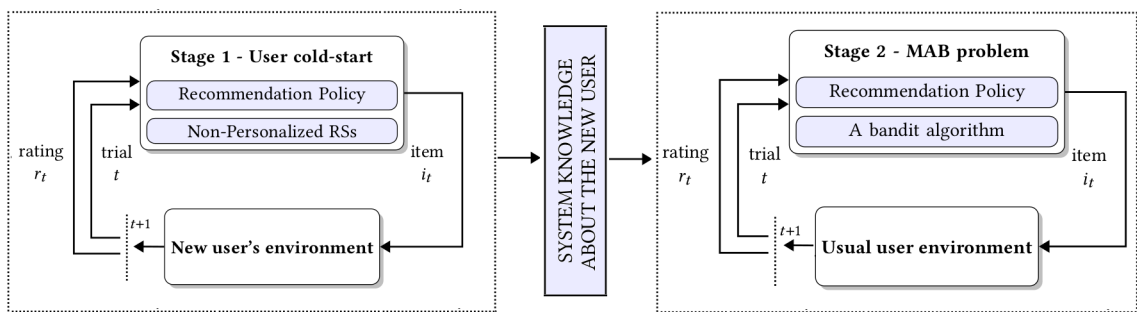


Figure 3.3: An illustration of our methodology created to identify the problem’s impact on contextual bandit algorithms.

Stage 1: First, we simulate an interactive scenario where the new user will face non-personalised recommendations for t times while the system learns with each feedback. As there is no consensus in the literature on how many items are needed to reach the desirable knowledge, we create this experiment to be parameter-free. This means that instead of defining a value for t , the interactive scenario will run until the system hits a percentage of the relevant items for each user. This percentage is defined in slots of 10% like 10%,

20%, 30%, ..., and 80%. Thus, for each user, the system will create eight distinct slots with the user's interactions made until hits that percentage of relevant items. In a simple example, if user A has 10 relevant items while another user B has 100 in the groundtruth, the interactive system will make as many as necessary recommendations until it achieves 1, 2, 3, ..., 8 relevant items for A and 10, 20, 30, ... 80 items for B. Probably, the system will perform fewer recommendations for user A than user B, but both will have 8 distinct slots of knowledge. To achieve distinct learning scenarios, we use a random approach to simulate pure exploration and the most popular items to simulate pure exploitation. In this stage, we recommend 5 items at each interaction.

Stage 2: It is another interactive system where an agent continuously recommends a list of items to the target user. But, in this stage, the user is no longer completely new. After stage 1, this user has already interacted with some items and there are distinct slots of knowledge about their preferences. Thus, the idea is to make more recommendations based on each slot of knowledge – i.e., with the hits and misses made until it gets each percentage of relevant items. These new recommendations are made according to the usual prediction rule of the bandit algorithm implemented at this stage. Therefore, the effectiveness of the bandit model in stage 2 will be directly related to the amount of knowledge achieved by each approach of stage 1. In this experiment, we select the traditional ϵ -Greedy algorithm by recommending 1 item per interaction.

This methodology enables the possibility to measure the impact of naive approaches to handle uncertainty scenarios as discussed in this work. Based on the number of interactions required in the stage 1 to hit each percentage of relevant items, we can measure how long the new user had to wait until the system started to apply a personalised method. Moreover, by measuring the precision-recall curve of the bandit algorithm in stage 2, we can also measure the impact of the first recommendations on the user's experience.

3.4.2 Experimental Setup

Then, in order to apply our evaluation methodology, we created an experimental setup according to the information below.

Datasets. First, we select the three traditional datasets from distinct domains described in Table 3.2. As we want to perform our evaluation methodology in the pure cold-start problem since it has the highest uncertainty about the user's preferences, we simulate some new users in these offline datasets. Basically, we select the last 20% of users that joined the system (i.e., users with the highest timestamps) to represent the new users. It means 2,000 users on Netflix, 10,648 users on GoodBooks, and 2,000 users on Yahoo. The other 80% of the data is used to train the algorithms, representing the information already in a system. All information around these new users (ratings, demographic infor-

mation, and others) is removed from the training set. To make possible our evaluation methodology, we filtered all datasets to ensure that users will have at least 20 items rated because we want to evaluate the user’s long-term preference similar to other works [77]. For those with many users (Netflix and Yahoo), we randomly selected 10,000 users in the dataset to facilitate our exhaustive experiments on the selected bandit algorithms.

Approaches. For this experiment, we select the ϵ -Greedy algorithm (based on SVD formulation) as the interactive RS to recommend until 10 items after each *slot of knowledge*. To simulate exploration and exploitation for new users, we select as:

- (1) **pure-exploration:** *Random* and *Entropy-based* algorithms;
- (2) **pure-exploitation:** *Most Popular* and *Best-Rated* algorithms.

In addition, to endorse our hypothesis about the impact of a smarter strategy in the candidate items selection, we also evaluate the performance of another non-personalised strategy that can achieve the trade-off between **exploration & exploitation**. This strategy combines the **log of popularity** and the **entropy** of the items. In this work, both concepts are defined as follow:

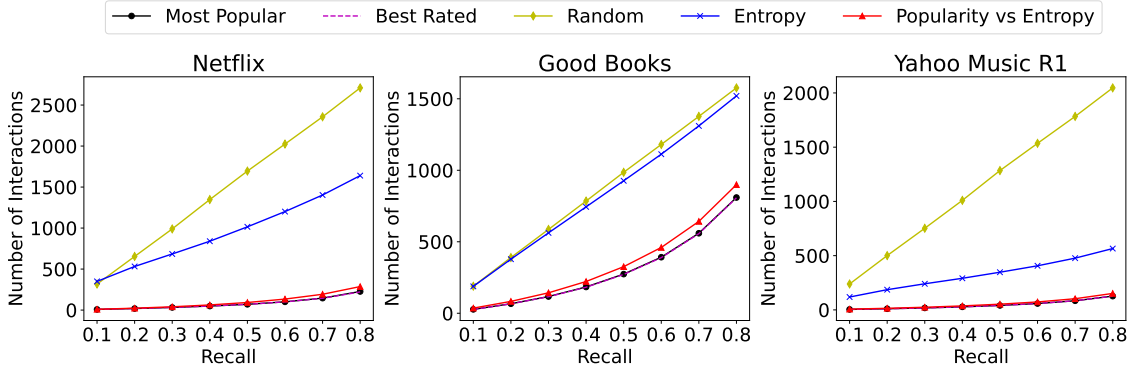
- popularity is the number of users who have rated the item;
- entropy is measured by: $\sum_r -P(i|r) \cdot \log P(i|r)$, where $P(i|r)$ is the probability of one item being rating by a value r .

Introducing popularity we want to increase the probability that a user will rate an item (i.e., a notion of exploitation). On the other hand, by applying entropy we intend to increase the amount of information that is possible to be achieved if the user rates the item (i.e., a notion of exploration). Both concepts have been mentioned as one of the most promising techniques of Active Learning [181, 75].

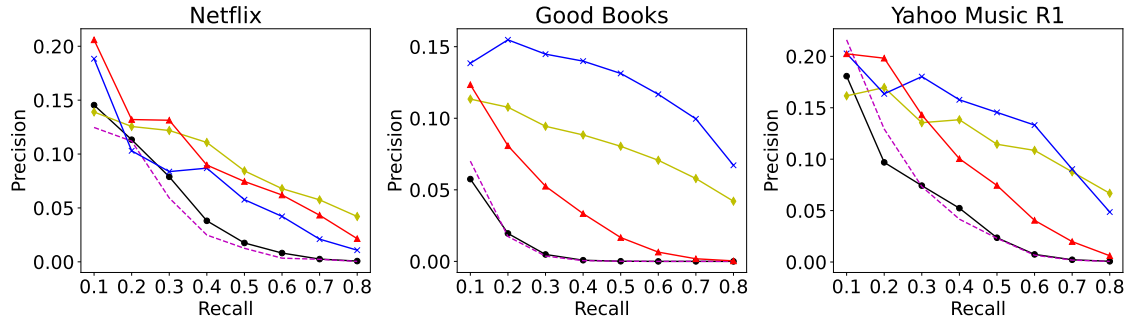
3.4.3 Results & Discussion

The methodology created to simulate the impact of naive methods to make the first recommendations for new users is then applied to over 20% of the users from the three selected datasets. Results are highlighted in Figure 3.4. Stage 1 is represented by the non-personalised approaches highlighted above. It is individually applied to each user until it hits a percentage of relevant items for each user – a slot of knowledge about their preferences. As aforementioned, each slot of knowledge is represented by the distinct levels of recall in the x-axis of both figures. Thus, each y-point is the result obtained according to each slot x of knowledge. The first one, Figure 3.4a, shows the number of interactions required to achieve this level of recall (resulting from stage 1). The second one, Figure 3.4b, represents the results from stage 2. It is the precision achieved by using the linear ϵ -Greedy

algorithm applied with the knowledge created so far. The idea is to validate which non-personalised approach from stage 1 can create better knowledge about the system and then produce more effective recommendations in the next stage, with the bandit algorithm.



(a) Average of interactions performed by non-personalised RSs.



(b) Precision@10 achieved by linear ϵ -Greedy with the previous knowledge.

Figure 3.4: Impact of a naive initialisation in Contextual Bandit. While pure exploration requires more interactions to achieve higher recall, pure exploitation does it so fast. However, this fast learning has a considerable impact on the model’s performance as a whole. In turn, balancing exploration and exploitation improves the system’s precision without requiring many initial interactions.

In a simple observation, pure exploration seems the best choice for new users because it guides the bandit algorithm to achieve the highest level of precision (Fig. 3.4b). However, to ensure this performance, these approaches require that many items be presented to the users. In our experiments, for instance, Random or Entropy-based approaches require that users analyse more than 8,000 items ($\sim 2,000$ interactions \times 5 items) to reach 80% of the user’s history (i.e., recall) – Fig. 3.4a. This is not feasible in practice since the users may not be so patient. Indeed, current systems usually choose pure-exploitation approaches (like Most Popular and Best-Rated) to make the first recommendations. On average, they often require less than 500 iterations ($\sim 2,500$ items) to achieve 80% of relevant items for each user. However, our analysis shows these approaches do not have the same impact on the bandit’s performance – Fig. 3.4b. These non-personalised RSs are based on global preferences and they do not add so much knowledge about the users. Thus, applying a contextual bandit algorithm after approaches

purely based on exploitation does not achieve high levels of precision.

In turn, our suggestion based on a learning process that combines exploration and exploitation seems more effective. By weighing items' popularity with their entropy, the system can identify interesting items for users and increase its own knowledge. In practice, it results in an approach that requires only a few interactions (quite similar to pure exploitation approaches) but achieves more precision gains in the user's long-term run. Although this combination does not bring the same knowledge achieved by pure exploration approaches, it also does not require many interactions to identify the user's taste. Thus, evaluating both results of Figure 3.4a and Figure 3.4b together, we can support the assumption that such a combination of exploration and exploitation since the beginning of the user's journey is more effective than other approaches. It also endorses our thesis statement by suggesting that such scenarios of uncertainty may determine per-user boundaries of accuracy in Contextual Bandit models.

3.5 Summary

In this chapter, we have investigated the first research question raised by this work around the impact of some scenarios of uncertainty in Contextual Bandits. First, we present a complete background of Contextual Bandit algorithms and their main assumptions to represent users and items in the recommendation domain. By inspecting the selected works in our Systematic Literature Review (see Appendix A), we could highlight the main approaches applied and define the guidelines usually followed by their implementations. It shows that even ϵ -Greedy, UCB, and TS usually follow the same prediction rule of traditional recommendation algorithms.

Then, we discuss and study the implication of uncertainty scenarios in their main prediction rule. As shown by this work, the absence of user information in the pure cold-start problem is the scenario with the highest uncertainty and it leads the bandit algorithms to perform as naive non-personalised recommenders in the first iterations. Some of them randomly select items in an attempt to guess the users' preferences. Others select the best items defined by the information known prior (choosing the most popular items). Other approaches perform a random exploration of the existing knowledge. These differences are related to the values used to represent the user's features vector.

Moreover, to measure the impact of such naive assumptions in the learning process of current Contextual Bandits, we create an evaluation methodology and perform a complete experiment in three distinct recommendation scenarios. Our observations show that naive approaches can delay the system's learning or even result in a bad experience for the user. It indicates that some scenarios of uncertainty determine per-user bound-

aries of accuracy for Contextual Bandit models. However, an approach based on both concepts of exploration and exploitation seems reasonable to mitigate such challenges. This approach consists of an attempt to increase the current knowledge about the user's preferences while also exploiting the existing information in the system. It results in a viable option for real-world scenarios by requiring only a few interactions to improve the levels of accuracy. This approach is then explored in the next chapters of this work.

Chapter 4

An Active Learning approach in Contextual Bandits

This chapter addresses the second question raised by this work: *How can exploration and exploitation be ensured in Contextual Bandit models when user preferences are uncertain?* The proposed solution consists of applying concepts from the Active Learning theory to the user feedback loop problem. Our idea is to explore such concepts in order to enhance current bandit models. We assume that by achieving more information about the user's preferences in scenarios of uncertainty, such contextual models may maximise their cumulative reward in the long run.

4.1 Active Learning

In the literature, Active Learning approaches have been proposed to improve the training processes of several Machine Learning algorithms that often require a considerable amount of high-quality data [199]. Their idea is to select the best candidate data points by querying for certain types of instances based on the data that the system has seen so far. Formally, the process is described by Elahi et al. [75] as follows. Given a training set of N input-output pairs $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where $x_i \in X$ is an instance, and $y_i \in Y$ is a label, we assume that there is a model M that maps input to output $M : X \rightarrow Y$, and another function $Loss(M)$ which measures the error of the model (the smaller the better). At every iteration j of the active learning process, the learner selects a candidate $x_j \in potentialCandidates \subset X$, requests the label of it, and obtains the label y_j . The instance x_j and its unknown label have the property that M' , which is the model M re-trained by adding the new pair (x_j, y_j) to the set of previously labelled instances, has the lowest loss.

In recommendation systems, Active Learning has been constantly motivated by the need to implement more effective sign-up processes [71]. In the sign-up stage, the system actively selects and proposes individual items or groups of items to be rated by the users [187, 188, 9]. For that, the system evaluates the entire set of items and selects the items that are estimated to be the most useful ones. The idea is to select the

items that may improve the accuracy of the system. Elahi et al. [75] classified the main strategies into two main categories: (1) non-personalised when the user information is not used to determine the candidate items; and (2) personalised, otherwise. Each of them is further partitioned into two subcategories: (1) single-heuristic when only one algorithm is applied; and (2) combined-heuristic when more than one technique is applied. Figure 4.1 illustrates the main approaches of each class.

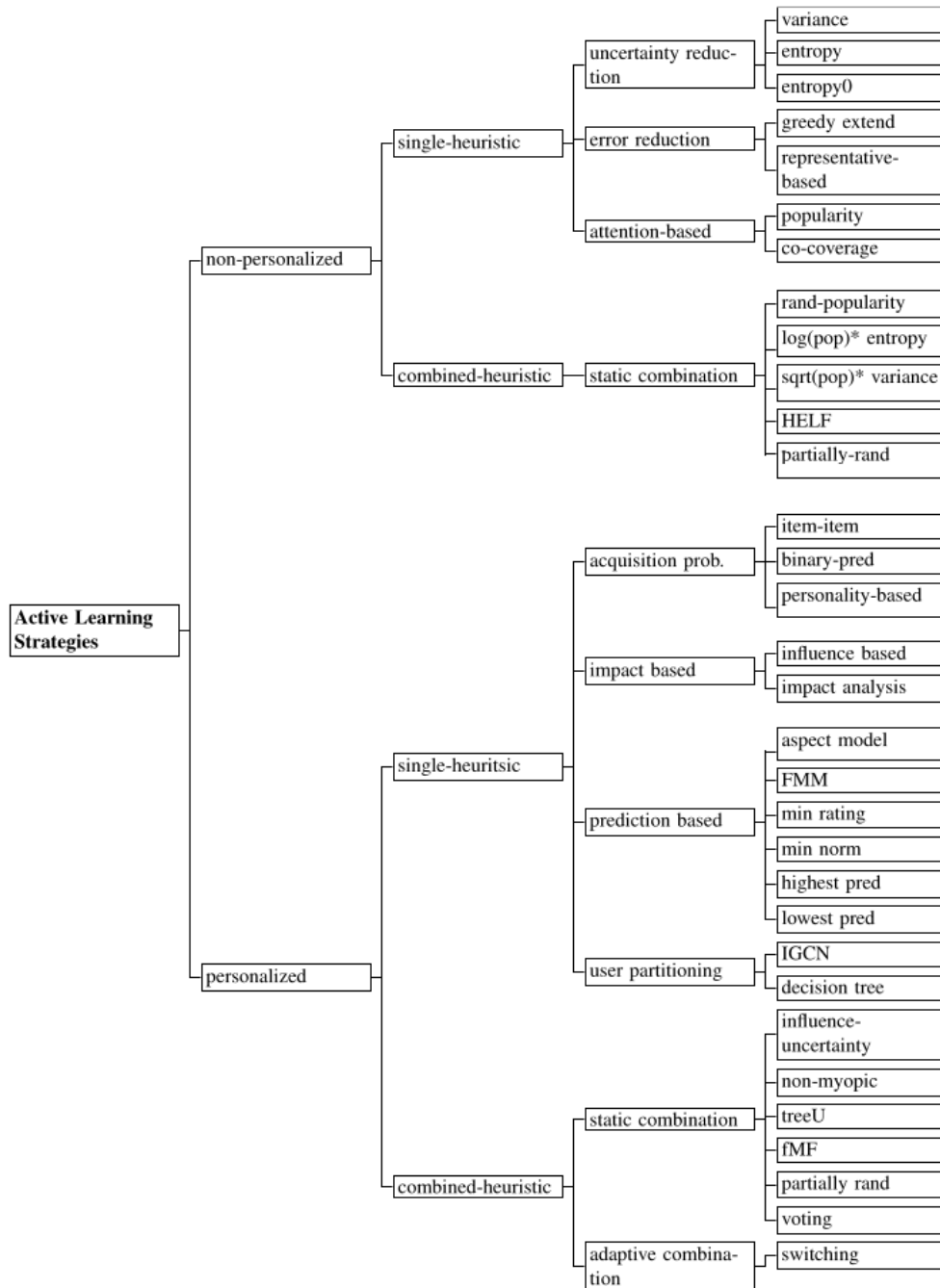


Figure 4.1: Active Learning approaches in recommendation systems [75].

However, the incorporation of these approaches into Contextual Bandit algorithms for interactive recommendation scenarios has not been thoroughly explored yet. Previous

works that deal with Active Learning in recommendation systems do not specifically address its application within the interactive scenario [34, 75]. These works propose a search for the best items among a set of potential candidates by minimising the loss function related to the system’s uncertainty. Nevertheless, this approach carries inherent risks as the items resulting from Active Learning are often controversial and may not align with the optimal choices for personalising the items recommended. In contrast, this work introduces a novel approach that combines one of the most promising Active Learning strategies with Contextual Bandits. Our objective is to leverage these approaches when the system encounters uncertainty regarding the user’s preferences.

4.2 Addressing scenarios of uncertainty

Ensuring the trade-off between exploration and exploitation in current Contextual Bandits is challenging when the system is uncertain about the user’s preferences. As previously discussed, the problem is usually on their prediction rule $p_u^\top q_i$ where p_u and q_i are, respectively, the user’s and item’s features vector represented by z features. Any significant alteration of this rule could result in a wrong combination of user and item features. Therefore, this work does not intend to modify how these algorithms usually work. Instead, we propose to introduce more information about the users to help the algorithms overcome such scenarios. The idea is to use the information available about the items to warm-start the user’s profile whenever it is necessary.

In this sense, we address two concepts around the items to collect more information about the system: (1) popularity – to maximise the probability of an item being rated; and (2) entropy – to increase the amount of information that is possible to obtain if the user rates the recommended item. While the first one introduces a notion of exploitation because it suggests in general the best items in the system, the second represents a notion of exploration since it is interested in the potential knowledge of each item. In this work, the popularity ρ of an item i is measured by the number of distinct users who rated i . In turn, the entropy ϕ is: $\sum_r -P(i|r) \cdot \log P(i|r)$, where $P(i|r)$ is the probability of an item i being rated with the rating r (usually defined in a range of 1 – 5). This added information is not associated with any sensitive information about the new users, such as social or demographic data. For this reason, it can be adapted for any other Contextual Bandit algorithm.

4.2.1 Mitigating the pure cold-start problem

When users have recently joined or they are not logged into the system, the interactive model will understand them as new users. It means that there is no information available about those users and it is not possible to make personalised recommendations so far – the hardest scenario of uncertainty. In this sense, we propose to make the first recommendation (when $t = 0$) equivalent to the non-personalised Active Learning strategy that combines popularity and entropy as illustrated in Equation 4.1. Our idea is to explore this moment when the user is not willing personalised recommendations to show items with the potential to get as much information as possible.

$$i_{(t=0)}^* = \arg \max_{i \in I} p_u^\top q_i \quad \equiv \quad i_{(t=0)}^* = \arg \max_{i \in I} \log \rho_i \cdot \phi_i \quad (4.1)$$

In practice, it means changing the output of the current prediction rule to be the same as the combination of popularity and entropy. In the current Contextual Bandit approaches, we should approximate the rule $p_u^\top \cdot q_i$ of the values achieved by multiplying popularity and entropy. As the item features vector $q_i \in Q$ can be measured even on the user cold-start scenario, the challenge is to estimate the user's feature vector p_u . This vector is then called as \mathbf{x} and the goal is to minimise the difference to the set \mathbf{y} (popularity vs. entropy) by minimising the Equation 4.2:

$$f(x) = \sum_{i \in I} (y_i - \mathbf{x}^\top \cdot q_i)^2, \quad \text{where: } y_i = \log \rho_i \cdot \phi_i \quad (4.2)$$

In this sense, we apply a quasi-Newton method of BFGS [164] to estimate values for \mathbf{x} . BFGS aims to gradually minimise the loss function $f(x)$ obtained through a gradient evaluation method. Starting from a vector of constants $\vec{\mathbf{x}}_0 = \{1\}_z$, for n iterations, the method estimates the next vector minimising the difference for the previous one: $\mathbf{x}_{n+1} = \mathbf{x}_n - [H(\mathbf{x}_n)]^{-1} \nabla f(\mathbf{x}_n)$. The Hessian matrix H is a square matrix of second-order partial derivatives of $f(x)$. This method is then applied for each item i to build the new vector \mathbf{x} .

$$n \text{ iterations} \left\{ \begin{array}{l} \vec{\mathbf{x}}_1 \leftarrow \vec{\mathbf{x}}_0 - [H(\vec{\mathbf{x}}_0)]^{-1} \nabla f(\vec{\mathbf{x}}_0) \\ \vec{\mathbf{x}}_2 \leftarrow \vec{\mathbf{x}}_1 - [H(\vec{\mathbf{x}}_1)]^{-1} \nabla f(\vec{\mathbf{x}}_1) \\ \vdots \\ \mathbf{X} \leftarrow \vec{\mathbf{x}}_{n-1} - [H(\vec{\mathbf{x}}_{n-1})]^{-1} \nabla f(\vec{\mathbf{x}}_{n-1}) \end{array} \right.$$

This approach is used to compute the initial user's features vector p_u . After the first interactions, the model should update the user's vector as usual.

4.2.2 Mitigating misleading assumptions

When the system has learned wrong assumptions for a user, the interactive model will exploit misleading assumptions until it relearns the user’s preferences. Thus, our approach to handling this scenario consists of using the Active Learning theory to recover the user’s interest after consecutive mismatching. In practice, this problem can be identified when the system has not provided at least one successful recommendation for a while. In this case, we assume that the user does not trust the personalised recommendations anymore. Consequently, we can replace the entire prediction rule (i.e., the original policy) with a recommendation based on the selected AL approach. Our proposal is to make this replacement when the number of consecutive mismatches, denoted as m , reaches a threshold $\mathcal{T} > 0$ specified as a parameter. In this way, the algorithm will have two value functions: one based on the AL approach, combining popularity and entropy; and another consisting of the original value function of the model.

$$\begin{array}{c} \text{PREDICTION RULE} \\ i^*(t) = \begin{cases} \text{original policy } \mathcal{Q}_t, & \text{if } m < \mathcal{T} \\ \arg \max_{i \in I} \log \rho_i \cdot \phi_i, & \text{otherwise} \end{cases} \end{array}$$

This novel approach offers an option to reduce the potential for misleading results with any Contextual Bandit algorithm. The extent of this change depends on the number of times the model chooses to use the AL approach. If \mathcal{T} is too low (e.g., 1 or 2), the AL approach will likely provide more recommendations than the original Contextual Bandit model. Conversely, if \mathcal{T} is too high (e.g., 20 or 30), more items will be recommended based on the original value function. The ideal value depends on the Contextual Bandit model and the desired outcome of the recommendations.

4.3 Modified Contextual Bandits

The Active Learning approach proposed by this work does not rely on sensitive information of new users, such as demographic or social and it can be adapted to any other bandit algorithm. In the first scenario, the idea is to modify the starting point ($t = 0$) of the interactive process. This captures the context x and then uses it to generate the user’s features. In the second scenario, we only need to create a mismatch counter m to assess how often the user does not rate the proposed item. If this counter reaches a threshold, the prediction rule is switched to the AL approach.

In this sense, we select three distinct contextual bandit models to be adapted by our new approach – one from each bandit class. Algorithms 1, 2, and 3 illustrate their adaptations. For the first scenario, we compute the context \mathbf{x} before the interactive process of

each method (on line 1). This context is then used to estimate the user’s feature vector p_u and updated with each item rated by the user. For the second scenario, we implemented a conditional state in the prediction rule of each algorithm. If the number of mismatches reaches the threshold \mathcal{T} (for $\mathcal{T} > 0$), the AL prediction rule will be triggered, and the item will then be selected based on the combination of its popularity and entropy. The remaining lines of the algorithms are the same as their original versions [288, 129, 114].

4.3.1 Contextual Linear ϵ -Greedy

The original version consists of a linear representation of the traditional ϵ -Greedy based on a Probabilistic Matrix Factorisation (PMF) to extract the features [288]. It exploits the usual rule ($p_u^\top \cdot q_i$) with probability ϵ and explores a random item with probability $1 - \epsilon$. The Algorithm 1 illustrates the modified version.

Algorithm 1 CONTEXTUAL ϵ -GREEDY

Require: features $Q = \{q_1, \dots, q_n\}$ from PMF, popularity ρ , entropy ϕ , variance λ_p , ϵ , and the threshold $\mathcal{T} > 0$

```

1:  $\mathbf{X} \leftarrow BFGS_x \sum_i (\log \rho_i \cdot \phi_i - \mathbf{x}^\top Q_i)^2$ 
2:  $\Sigma_{u,t} \leftarrow \lambda_p I_d$ 
3:  $m \leftarrow 0$ 
4: for  $t = 1, 2, \dots, T$  do
5:   Estimates  $p_{u,t} \leftarrow \Sigma_{u,t}^{-1} \cdot \mathbf{X}$ 
6:   With probability  $1 - \epsilon$ :
7:     If  $m < \mathcal{T}$ :
8:        $i_t^* \leftarrow \arg \max_{i \in I \setminus R} p_{u,t}^\top q_i$ 
9:     Otherwise:
10:       $i_t^* \leftarrow \arg \max_{i \in I} \log \rho_i \cdot \phi_i$ 
11:       $m \leftarrow 0$ 
12:     Otherwise: selects  $i_t^*$  randomly
13:     Receives the reward  $r_{u,i^*(t)}$ 
14:     If  $r_{u,i^*(t)} = 0$ :  $m \leftarrow m + 1$ 
15:     Updates  $\Sigma_{u,t} \leftarrow \Sigma_{u,t} + q_{i^*(t)} \cdot q_{i^*(t)}^\top$ 
16:     Updates  $\mathbf{X} \leftarrow \mathbf{X} + r_{u,i^*(t)} \cdot q_{i^*(t)}$ 
17: end for

```

4.3.2 Contextual LinUCB

The original version consists of a linear representation of the UCB defining the contexts as latent factors from the SVD [129]. It also exploits the usual rule ($p_u^\top \cdot q_i$) but adds an uncertainty over through the confidence interval $\|q_i\|$. Algorithm 2 illustrates it.

Algorithm 2 CONTEXTUAL *LinUCB***Require:** features Q from SVD, popularity ρ , entropy ϕ , the value α , threshold $\mathcal{T} > 0$

- 1: $\mathbf{X} \leftarrow BFGS_x \sum_i (\log \rho_i \cdot \phi_i - \mathbf{x}^\top Q_i)^2$
- 2: Initialise $\Sigma_{u,t} \leftarrow I_d$; Initialise $m \leftarrow 0$
- 3: **for** $t = 1, 2, \dots, T$ **do**
- 4: Estimates $p_{u,t} \leftarrow \Sigma_{u,t}^{-1} \cdot \mathbf{X}$
- 5: If $m < \mathcal{T}$:
- 6: $i_t^* \leftarrow \arg \max_{i \in I \setminus R} p_{u,t}^\top q_i + \alpha \|q_i\|_{2, \Sigma_{u,t}}$ where $\|q_i\|_{2, \Sigma_{u,t}} = \sqrt{q_i^\top \Sigma_{u,t} q_i}$
- 7: Otherwise:
- 8: $i_t^* \leftarrow \arg \max_{i \in I} \log \rho_i \cdot \phi_i$
- 9: $m \leftarrow 0$
- 10: Receives the reward $r_{u, i^*(t)}$
- 11: If $r_{u, i^*(t)} = 0$: $m \leftarrow m + 1$
- 12: Updates $\Sigma_{u,t} \leftarrow \Sigma_{u,t} + q_{i^*(t)} \cdot q_{i^*(t)}^\top$; Updates $\mathbf{X} \leftarrow \mathbf{X} + r_{u, i^*(t)} \cdot q_{i^*(t)}$
- 13: **end for**

4.3.3 Contextual PTS

The original version consists of an adaptation of the traditional TS model using PMF and Bayesian inference around the items [114]. It also adds a particle filtering approach for exploration. The modified version is illustrated in Algorithm 3.

Algorithm 3 CONTEXTUAL *PTS***Require:** features from PMF, variances σ , K particles, popularity ρ , entropy ϕ , and threshold $\mathcal{T} > 0$

- 1: $\mathbf{X} \leftarrow BFGS_x \sum_i (\log \rho_i \cdot \phi_i - \mathbf{x}^\top Q_i)^2$
- 2: Initialise particles: $[d_{X_u} \leftarrow \mathbf{X}] \forall K$; Initialise $m \leftarrow 0$
- 3: **for** $t = 1, 2, \dots, T$ **do**
- 4: $d' \sim d_w$
- 5: $\tilde{Q} \leftarrow d'_Q$
- 6: $\tilde{p}_u \sim P(p_u | \tilde{Q}, d'_{\sigma_P}, \sigma, r_{1:t-1}^o)$
- 7: If $m < \mathcal{T}$:
- 8: $i_t^* \leftarrow \arg \max_{i \in I \setminus R} \tilde{p}_u \tilde{q}_i$
- 9: Otherwise:
- 10: $i_t^* \leftarrow \arg \max_{i \in I} \log \rho_i \cdot \phi_i$
- 11: $m \leftarrow 0$
- 12: Receives the reward $r_{u, i^*(t)}$
- 13: If $r_{u, i^*(t)} = 0$: $m \leftarrow m + 1$
- 14: $r_t^o \leftarrow (u, i_t^*, r_{u, i^*(t)})$
- 15: Updates d based on [114]
- 16: **end for**

4.4 Summary

In this chapter, we answered the second research question by presenting a novel approach that applies concepts from Active Learning to mitigate scenarios of uncertainty about the user’s preferences. First, we present the background concepts around Active Learning and how this theory has been applied in traditional recommendation systems. In short, Active Learning aims to select items with the strongest potential to collect as much information as possible about the users. In this work, we propose to apply this concept from their non-personalised strategies since we intend to mitigate scenarios of uncertainty about the users. The selected strategy is based on the entropy and popularity of each item to balance the probability of increasing the system knowledge and the probability of being rated by the user.

Then, we present our approach that combines this non-personalised method with current Contextual Bandit models. Our idea is to introduce this Active Learning approach in two scenarios when the system is uncertain about the user’s preferences: the first interaction (pure cold-start problem), and after some misleading assumptions. For the first scenario, we propose to add an initial feature vector to make the first recommendation exclusively based on the Active Learning strategy. In turn, for the second one, we propose to use this non-personalised strategy every time the user dislikes the recommendation for a consecutive number of times. In these ways, we ensure that the Contextual Bandit model still applies the trade-off between exploration and exploitation.

Finally, we select three traditional Contextual Bandits – Linear ϵ – Greedy, Lin-UCB, and PTS (one from each class of MAB), and demonstrate how our approach could be applied to each algorithm. In short, we add two additional steps. The first is placed at the beginning of each algorithm and it aims to select the contextual vector for the new users. This contextual vector will represent the first user’s features. In turn, the second step is a conditional procedure that proposes to change the prediction rule. This approach aims to replace the current rule with one based on Active Learning when the condition is applied. The condition is a simple counter of the number of consecutive mismatches reached by the bandit algorithm. In the next chapter, we aim to answer our third question by measuring the impact of such changes on the models’ performance.

Chapter 5

Experiments: Results & Discussions

As aforementioned, this work investigates whether mitigating scenarios of uncertainty about the user’s preferences may improve the quality of current Contextual Bandits. In this sense, our proposal consists of applying an Active Learning approach in two of these scenarios: pure cold-start and misleading assumptions. As this approach is not dependent on the base algorithm, we apply this approach to three distinct bandit models and create their modified versions. Our idea is to introduce more information about the users in such scenarios of uncertainty to improve the next recommendations made by the interactive model. Thus, this chapter aims to measure the quality of our solutions through the third research question raised by this work: *What is the impact of enhancing exploration and exploitation when the Contextual Bandit model is uncertain about the user preferences?*

First, we present our experimental setup designed to answer this question. It follows the best practices in the literature about Multi-Armed Bandits and contains distinct evaluation metrics for three real-world domains. Then, we perform distinct analyses of the performance of our modified algorithms. Such analyses aim to answer the third research question by answering three smaller questions about our proposed solution based on Active Learning theory:

- **Q1:** *Are the modified versions of the bandit algorithms statistically superior to the original ones?*
- **Q2:** *Has the improvement of these modified versions been caused by the usual popularity bias of offline datasets?*
- **Q3:** *What is the effect of addressing uncertainty scenarios compared to existing state-of-the-art baselines?*

5.1 Experimental Setup

In order to validate our proposal, we define an experimental setup that simulates the user cold-start problem in offline datasets. All details are explained as follows.

Datasets. First, we select three recommendation datasets from three distinct domains: movies, books, and songs. They are all described in Table 5.1. To facilitate our interactive experiment, we randomly filter 10,000 users from datasets with many interactions (i.e., Netflix and Yahoo), and ensure that each user has at least 20 items rated.

Datasets	# Users	# Items	Sparsity
Netflix	10,000	17,372	98.67%
GoodBooks	53,423	10,000	98.88%
Yahoo Music R1	10,000	13,214	99.22%

Table 5.1: An overview of the datasets applied in this work.

Then, to simulate the pure cold-start scenario, we select the last 20% users who joined the system as the new users. These users are selected by defining a global timestamp to cut the dataset into training and test sets as illustrated in Figure 5.1. The cut time of each dataset consists of the first timestamp of the last 20% of new users. All data prior to the first interaction of the new users (coloured green in the figure) are used to train the algorithms, representing the information already within the system. In turn, all data from new users are removed from the training set (coloured yellow). This approach ensures that there is no data leakage in the experiment and it results in 2,000 users on Netflix, 10,648 on GoodBooks, and 2,000 on Yahoo. The only drawback is that we have to remove the extra ratings that happened before the cut point (those coloured red) from users that are in the training set.

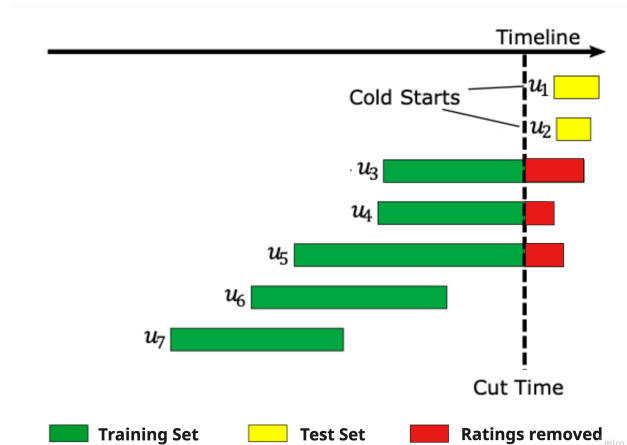


Figure 5.1: Global timestamp cut to select new users in offline datasets.

Evaluation Policy. As the interactive scenario has been recently studied in the literature, there is no consensus about the best practices to evaluate bandit algorithms. In this sense, we searched for all policies available on the articles from our SLR (see Appendix A) and, then, propose the evaluation policy described on the Algorithm 4. This evaluation is quite similar to a recent one proposed by Sanz-Cruzado et al. [192]. Each trial represents a user’s interaction with the system when one item is recommended and the user will

rate or not it. At each trial $t \in T$, the new user u is selected uniformly to simulate the random accessing pattern of users in the real world. Then, the system will recommend 1 item i for this user according to the method Π . This method Π will recommend the new item based on the training set \mathcal{D} , the available items (i.e., the items that have not been recommended yet – $I \setminus R_u$), and the knowledge Δ achieved until the trial t . After that, the system will update the current knowledge for the next iterations. Differently for other works, the system is not compelled to only recommend previously rated items (i.e., those from the test) to avoid a biased solution. All items can be recommended at each trial as long as it was not recommended in another trial before. Moreover, in our case, we compel the system to always make T iterations for each new user u .

Algorithm 4 Evaluation Policy

Require: Training set \mathcal{D} , testing set \mathcal{T} , number of trials T , and the number of items k to be recommended for each user per trial

- 1: $R_u \leftarrow \emptyset_{\forall u \in \mathcal{U}}$ // Recommendation list
- 2: $\Delta \leftarrow \emptyset$ // Knowledge around the users
- 3: $N \leftarrow |U| \times T$ // Total number of iterations to play
- 4: **for** $t = 1, 2, 3, \dots, N$ **do**
- 5: $\Delta' \leftarrow \emptyset$ // Saves the current knowledge
- 6: $u \leftarrow |U|$ // Random selects a user u (limited to T trials)
- 7: $i_t \leftarrow \Pi(u, \mathcal{D}, I \setminus R_u, \Delta)$ // Gets the item recommended
- 8: $R_u \leftarrow R_u \cup \{i_t\}$ // Saves this item
- 9: $\Delta \leftarrow \Delta \cup \{\mathcal{T}(u, i_t)\}$ // Updates the system knowledge
- 10: **end for**

After that, the recommendations are then evaluated based on the Cumulative Reward [192]. A reward is 1 when the recommended item was rated by the target user with a value $r \geq 4$ in the test set. For the experiments, we perform an extensive grid search for the best parameters of all algorithms by using 20% of the training as validation. This search includes the parameter τ used in our approach. All these evaluation policies and metrics applied in this work were published in an open-source framework named iRec [209]. The iRec is a complete framework that enables reproducible experiments in the interactive recommendation field with several methodologies, algorithms, and datasets (see Appendix B for more details).

Baselines. Mitigating the lack of information about users is one of the hardest challenges of the literature and it has attracted the attention of a huge number of researchers [17, 77, 241, 203]. However, its impact on the user’s journey has not been deeply studied in an interactive scenario modelled by Contextual Bandits. Moreover, most of the recent works do not address such challenges in the user’s journey since his first interaction, when no item has been rated yet [101, 200]. In this work, in turn, one of the uncertainty scenarios consists of the pure cold-start problem, where there is not any previous knowledge available (e.g., not even 1 item rated or no social information). Furthermore, to the

best of our knowledge, neither other work has proposed to mitigate these challenges by compelling the Contextual Bandits to address exploration and exploitation since the first recommendations.

Therefore, we only identified three classes of algorithms related to this challenge:

(1) ***State-of-the-art Contextual Bandits***. Context-aware recommendation systems can mitigate the lack of information about the users and items by exploring their external attributes (i.e., context) [123]. In general, this context can be:

- *fully observable*: when the contextual factors are known explicitly;
- *partially observable*: when only some information is known explicitly; or
- *unobservable*: when there is not any explicit information and the system needs to model it by latent variables.

In the Multi-Armed Bandit field, for instance, there are several algorithms to work with partially observable or unobservable contextual factors. Algorithms like *LinUCB* [129], *FactorUCB* [239], *hLinUCB* [238], and *CoLin* [257] explore explicit information about the user (demographic, geographical, and others) and the items (descriptions, categories, and others). Other ones, like *FactorUCB*, *CoLin* [257], **Linear UCB** [288] and **GLM-UCB** [288] explore unobservable factors by modelling users and items with a Probabilistic Matrix Factorisation approach. And, more recently, new algorithms like the *hLinUCB* [238] have been proposed to aggregate the partially observable information about the users with the latent factors usually applied in the field. However, as this work proposes to study the scenario with the total absence of user information, we do not apply any strategy that requires partially observable factors. We decided to use the algorithms highlighted in bold because they are competitive baselines for our modified algorithms.

(2) ***Meta-Learning approaches***. Similar to the Active Learning theory, Meta-learning is a class of Reinforcement Learning algorithms recently popularised for training easily generalised machine learning models. The idea is to create models that can rapidly adapt to a new task that is not used during the training with only a few examples [127]. It is inspired by the human learning process, which can quickly learn new tasks based on a small number of examples. In the recommendation field, it has been adapted to help the system deal with the huge amount of users that must receive a personalised recommendation. The idea is to consider each user as a single task and create a learning environment with a set of users to teach algorithms how to deal with new tasks (i.e., new users). The first work proposed in this sense was MeLU [127], a meta-learning strategy to identify the specific items that can be used to analyse the individual preferences of new users quickly. Then, the most recent work has proposed the **NICF** [297] to use Meta-Learning to guide a neural network model in an interactive environment. NICF outperformed MeLU.

(3) **Bayesian Inference methods.** These algorithms are based on probabilistic distributions to make inferences when very little evidence is available. Thus, in the recommendation field, they have been directly applied to deal with the cold-start scenario. They usually are variations of the traditional **Thompson Sampling** by applying a Beta distribution to fit the probability of success and failure of each arm with two positive parameters: α and β . They can be used to learn the arms' relevance before the interactive recommendation, through a training sample, or even learn it over the user's interactions. If they choose to learn on live, they will *explore* different arms in the first recommendations until they learn the best options. In turn, if they choose to learn before the recommendations, they will *exploit* the most successful arms (i.e., the most popular options). In this class of algorithms, we can highlight **PTS** [114], **ICTRTS** [245], and **Cluster-Bandit** [192]. PTS introduces a particle filtering process to guide the recommendations made by a PMF formulation. In turn, ICTRTS applies a TS and particle filtering approach combined with a topic regression model to handle the dilemma of exploration and exploitation. Finally, Cluster-Bandit is a variant of the nearest-neighbours collaborative filtering algorithm but endowed with a controlled stochastic exploration from a Beta distribution. All of these three algorithms are baselines for this work.

In this sense, we compare our modified algorithms with two non-personalised algorithms; three traditional MAB algorithms adapted to the recommendation field; and six competitive baselines. All baselines are highlighted in bold above. For all baselines, we perform an extensive grid search for the best parameters in a validation set (20% of the training). In most cases, we could follow the range of parameters searched in their original papers. However, some authors have not described such information and the authors have not answered our emails. In these cases, we inferred the range of parameters based on our own knowledge of such algorithms. The best parameters identified are described in Table 5.2. All the code is available at our GitHub repository¹ and split by the main experiments of our most recent paper [206].

5.2 Modified vs. Original Algorithms

This section aims to answer the first sub-question: *Are the modified versions of the bandit algorithms statistically superior to the original ones?* Basically, we perform a comparison of the modified Linear ϵ -Greedy, LinUCB, and PTS algorithms created in this work (see Algorithms 1, 2, 3) with their original versions. For this experiment, we split our modified versions into two versions for each bandit algorithm. The first one proposes to only mitigate the pure cold-start problem. Thus, each algorithm contains the context

¹<https://github.com/ncsilvaa/bandit-uncertainty>

	Netflix	Good Books	Yahoo Music
UCB	$c=0.01$	$c=0.25$	$c=0.01$
TS	$\alpha_0=1, \beta_0=100$	$\alpha_0=1, \beta_0=100$	$\alpha_0=1, \beta_0=100$
ϵ -Greedy	$\epsilon=0.001$	$\epsilon=0.1$	$\epsilon=0.0001$
Linear UCB	<ul style="list-style-type: none"> $\alpha=0.5, T=20, q_p^2=0.01$ $\sigma_q^2=0.01, \sigma^2=0.05$ num_lat=20 stop_criteria=0.0009 	<ul style="list-style-type: none"> $\alpha=1, T=20, q_p^2=0.01$ $\sigma_q^2=0.01, \sigma^2=0.05$ num_lat=15 stop_criteria=0.0009 	<ul style="list-style-type: none"> $\alpha=1, T=20, q_p^2=0.01$ $\sigma_q^2=0.01, \sigma^2=0.05$ num_lat=15 stop_criteria=0.0009
Linear ϵ -Greedy	<ul style="list-style-type: none"> num_lat=5, $\epsilon=0.001$ stop_criteria=0.0009 $\sigma=0.05, T=20$ $\sigma_q^2=0.001, \sigma_u^2=0.001$ 	<ul style="list-style-type: none"> num_lat=20, $\epsilon=0.01$ stop_criteria=0.0009 $\sigma=0.05, T=20$ $\sigma_q^2=0.01, \sigma_u^2=0.01$ 	<ul style="list-style-type: none"> $\sigma=0.05, \text{num_lat}=20$ stop_criteria=0.0009 $T=20, \epsilon=0.5$ $\sigma_q^2=0.01, \sigma_u^2=0.01$
GLM-UCB	<ul style="list-style-type: none"> $c=4, T=20, q_p^2=0.01$ $\sigma_q^2=0.01, \sigma^2=0.05$ num_lat=10 stop_criteria=0.0009 	<ul style="list-style-type: none"> $c=4, T=20, q_p^2=0.01$ $\sigma_q^2=0.01, \sigma^2=0.05$ num_lat=20 stop_criteria=0.0009 	<ul style="list-style-type: none"> $c=4, T=20, q_p^2=0.01$ $\sigma_q^2=0.01, \sigma^2=0.05$ num_lat=10 stop_criteria=0.0009
NICF	<ul style="list-style-type: none"> batch=256 clip_param=0.2 dropout=0.01 gamma=0.0 inner_epoch=50 latent_factor=10 learning_rate=0.001 num_blocks=1 num_heads=2 restore_model=False rnn_layer=2 time_step=100 training_epoch=4000 	<ul style="list-style-type: none"> batch=256 clip_param=0.2 dropout=0.01 gamma=0.0 inner_epoch=50 latent_factor=10 learning_rate=0.001 num_blocks=1 num_heads=2 restore_model=False rnn_layer=2 time_step=100 training_epoch=4000 	<ul style="list-style-type: none"> batch=256 clip_param=0.2 dropout=0.01 gamma=0.0 inner_epoch=50 latent_factor=10 learning_rate=0.001, num_blocks=1 num_heads=2 restore_model=False rnn_layer=2 time_step=100 training_epoch=4000
PTS	<ul style="list-style-type: none"> num_lat=15, D=8 $\sigma_B^2=0.6, \sigma_V^2=0.6$ $\sigma^2=0.3$ 	<ul style="list-style-type: none"> num_lat=5, D=2 $\sigma_B^2=1.0, \sigma_V^2=1.0$ $\sigma^2=0.5$ 	<ul style="list-style-type: none"> num_lat=10, D=6 $\sigma_B^2=0.6, \sigma_V^2=0.6$ $\sigma^2=0.3$
ICTRTS	<ul style="list-style-type: none"> num_lat=2, B=5 	<ul style="list-style-type: none"> num_lat=2, B=5 	<ul style="list-style-type: none"> num_lat=2, B=5
Cluster-Bandit	<ul style="list-style-type: none"> B=5, C=0.5, D=3 num_clusters=8 num_lat=20 	<ul style="list-style-type: none"> B=5, C=0.5, D=3 num_clusters=8 num_lat=20 	<ul style="list-style-type: none"> B=5, C=0.5, D=3 num_clusters=4 num_lat=20
LinUCB	<ul style="list-style-type: none"> $\alpha=0.9, \text{num_lat}=5$ 	<ul style="list-style-type: none"> $\alpha=0.25, \text{num_lat}=10$ 	<ul style="list-style-type: none"> $\alpha=1.0, \text{num_lat}=10$

Table 5.2: Parameters identified after a Grid Search tuning using 20% of the training set.

created in line 1, but they do not have the conditional statement based on the number of consecutive mistakes (i.e., $m < \mathcal{T}$). This version is tagged as $t = 0$ since it only mitigates the first user’s interaction in the system. The second version, in turn, proposes to mitigate both scenarios of uncertainty – the pure cold-start and the misleading assumptions. These versions are identical to the Algorithms 1, 2, 3 presented in the last chapter. They are tagged as $t \geq 0$ for working on the entire user journey since the first interactions. The identified parameters of each version are described in Table 5.3.

Therefore, Table 5.4 highlights the Cumulative Reward achieved after T interactions evaluating (or not) one recommended item. Each value is an average of the reward achieved for all users in this experiment. In this sense, we applied the Wilcoxon test for non-parametric distributions with a p-value equal to 0.05.

	Netflix	Good Books	Yahoo Music
LinUCB $_{t=0}$	<ul style="list-style-type: none"> $\alpha=0.9$ num_lat=5, $\tau=0$ 	<ul style="list-style-type: none"> $\alpha=0.25$ num_lat=10, $\tau=0$ 	<ul style="list-style-type: none"> $\alpha=0.5$ num_lat=20, $\tau=0$
LinUCB $_{t \geq 0}$	<ul style="list-style-type: none"> $\alpha=0.9$, num_lat=5, $\tau=20$ 	<ul style="list-style-type: none"> $\alpha=0.25$, num_lat=10, $\tau=20$ 	<ul style="list-style-type: none"> $\alpha=0.3$, num_lat=10, $\tau=10$
PTS $_{t=0}$	<ul style="list-style-type: none"> num_lat=10, D=10 $\tau=0$, $\sigma^2=0.3$ $\sigma_U^2=0.6$, $\sigma_V^2=0.6$ 	<ul style="list-style-type: none"> num_lat=5, D=2 $\tau=0$, $\sigma^2=0.5$ $\sigma_U^2=1.0$, $\sigma_V^2=1.0$ 	<ul style="list-style-type: none"> num_lat=10, D=10 $\tau=0$, $\sigma^2=0.3$ $\sigma_U^2=0.6$, $\sigma_V^2=0.6$
PTS $_{t \geq 0}$	<ul style="list-style-type: none"> num_lat=10, D=4 $\tau=5$, $\sigma^2=0.3$ $\sigma_U^2=0.6$, $\sigma_V^2=0.6$ 	<ul style="list-style-type: none"> num_lat=5, D=2 $\tau=5$, $\sigma^2=0.5$ $\sigma_U^2=1.0$, $\sigma_V^2=1.0$ 	<ul style="list-style-type: none"> num_lat=10, D=10 $\tau=5$, $\sigma^2=0.3$ $\sigma_U^2=0.6$, $\sigma_V^2=0.6$
Linear ϵ -Greedy $_{t=0}$	<ul style="list-style-type: none"> $\tau=0$, $\epsilon=0.001$ num_lat=5 stop_criteria=0.0009 $T=20$, $\sigma=0.05$ $\sigma_q^2=0.01$, $\sigma_u^2=0.01$ 	<ul style="list-style-type: none"> $\tau=0$, $\epsilon=0.01$ num_lat=20 stop_criteria=0.0009 $T=20$, $\sigma=0.05$ $\sigma_q^2=0.01$, $\sigma_u^2=0.01$ 	<ul style="list-style-type: none"> $\tau=0$, $\epsilon=0.01$ num_lat=20 stop_criteria=0.0009 $T=20$, $\sigma=0.05$ $\sigma_q^2=0.01$, $\sigma_u^2=0.01$
Linear ϵ -Greedy $_{t \geq 0}$	<ul style="list-style-type: none"> $\tau=5$, $\epsilon=0.001$ num_lat=5 stop_criteria=0.0009 $T=20$, $\sigma_q^2=0.001$ $\sigma_u^2=0.001$, $\sigma=0.01$ 	<ul style="list-style-type: none"> $\tau=5$, $\epsilon=0.01$ num_lat=20 stop_criteria=0.0009 $T=20$, $\sigma_q^2=0.01$ $\sigma_u^2=0.01$, $\sigma=0.05$ 	<ul style="list-style-type: none"> $\tau=5$, $\epsilon=0.01$ num_lat=20 stop_criteria=0.0009 $T=20$, $\sigma_q^2=0.01$ $\sigma_u^2=0.01$, $\sigma=0.05$

Table 5.3: Identified parameters for our modified version of the algorithms.

In general, we can observe significant improvements in both modified versions for each algorithm throughout all interactions. Obviously, the results of each modified approach still depend on the quality of the original bandit algorithm. In the Linear ϵ -Greedy (the simplest of these three models), we notice several improvements across all user interactions. In the original version, the model starts from a completely random solution (a pure-exploration approach) and cannot get satisfactory results in 100 interactions. Otherwise, in the modified versions, the model begins by recommending popular items with high entropy – a better approach than the original one, producing short-term improvements as well. Moreover, the modified version that addresses both uncertainty scenarios ($t \geq 0$) produces even better results than the others. It happens because the model can also recalculate its behaviour if it experiences consecutive mismatches. Similar behaviour can be observed in the LinUCB algorithm. While there may not be a substantial difference among the three variants, our modified versions have demonstrated statistically superior performance compared to the original implementations.

On the other hand, when analysing the contextual change in the PTS algorithm, we notice that our improvement is not as expressive as in the others. It happens because this model already performs a combination of exploration and exploitation in the first interactions by making a Bayesian inference of the distribution known prior. However, in practice, this approach consists of a random sampling of the most popular items and still is strongly biased by the exploitation goal. In turn, as our approach offers a smarter way to combine exploration and exploitation, our contextual changes make the model able to

Dataset	Netflix				
Measure	Cumulative Reward				
T	5	10	20	50	100
Linear ϵ -Greedy	0.002	0.189	0.735	2.104	5.479
Linear ϵ -Greedy $_{t=0}$	0.382▲	0.817▲	2.229▲	6.245▲	12.617▲
Linear ϵ -Greedy $_{t\geq 0}$	0.330▲	0.971▲	2.985▲	10.274▲	20.004▲
PTS	0.412	0.918	2.190	7.639	14.846
PTS $_{t=0}$	0.493▲	1.034▲	2.583▲	8.218▲	15.379▲
PTS $_{t\geq 0}$	0.469▲	1.028▲	2.691▲	8.276▲	15.312▲
LinUCB	0.677	2.225	5.535	13.567	22.980
LinUCB $_{t=0}$	1.065▲	2.777▲	6.111▲	14.159▲	23.410▲
LinUCB $_{t\geq 0}$	1.065▲	2.777▲	6.111▲	14.161▲	23.409▲
Dataset	Yahoo Music				
Measure	Cumulative Reward				
T	5	10	20	50	100
Linear ϵ -Greedy	0.021	0.059	0.187	0.985	3.278
Linear ϵ -Greedy $_{t=0}$	1.090▲	2.189▲	4.497▲	11.122▲	19.857▲
Linear ϵ -Greedy $_{t\geq 0}$	1.190▲	2.459▲	5.355▲	13.208▲	22.867▲
PTS	0.902	2.267	5.649	15.263	23.333
PTS $_{t=0}$	0.637▼	1.655▼	4.804▼	14.998▼	24.023▲
PTS $_{t\geq 0}$	0.657▼	1.722▼	4.857▼	14.994▼	24.112▲
LinUCB	0.746	1.475	5.702	16.662	26.538
LinUCB $_{t=0}$	1.410▲	2.641▲	4.191▼	16.780●	27.349▲
LinUCB $_{t\geq 0}$	1.411▲	2.509▲	7.395▲	18.057▲	27.425▲
Dataset	Good Books				
Measure	Cumulative Reward				
T	5	10	20	50	100
Linear ϵ -Greedy	0.026	0.039	0.092	0.282	0.723
Linear ϵ -Greedy $_{t=0}$	0.403▲	1.016▲	2.036▲	5.871▲	11.726▲
Linear ϵ -Greedy $_{t\geq 0}$	0.404▲	0.997▲	2.026▲	6.164▲	12.794▲
PTS	0.841	2.047	4.159	7.669	12.121
PTS $_{t=0}$	0.969▲	2.256▲	4.331▲	8.233▲	12.825▲
PTS $_{t\geq 0}$	0.993▲	2.324▲	4.507▲	8.492▲	13.095▲
LinUCB	0.651	0.889	3.572	9.541	15.758
LinUCB $_{t=0}$	1.174▲	1.893▲	5.435▲	11.252▲	17.522▲
LinUCB $_{t\geq 0}$	1.174▲	1.893▲	5.435▲	11.258▲	17.548▲

Table 5.4: A comparison between original Contextual Bandits and their modified versions for each recommendation domain. While algorithms labelled with $t = 0$ only address the pure cold-start, those labelled with $t \geq 0$ mitigate both scenarios of uncertainty. The results show a significant improvement achieved by the modified algorithms based on the Wilcoxon test with a p-value = 0.05. The symbol ▲ represents significant gains, ● represents statistical draws, and ▼ represents significant losses.

surpass its original version in most of the datasets. Only in the Yahoo dataset that such gain is mainly focused on the long term. In our opinion, it happens because this dataset is

part of an experiment made by Yahoo where they included some random recommendations during the user’s journey. In this sense, the popularity bias is not so strong and the combination of exploration and exploitation made by the original PTS is already enough to ensure a good understanding of the user’s preferences. In this case, our modified algorithms require more interactions to learn the user’s preferences and surpass the original versions.

Therefore, these empirical findings strongly reinforce our thesis statement by demonstrating the statistical superiority of all modified algorithms when compared to their original ones. However, these results also raise a subsequent question to be further investigated. As most of the recommendations’ datasets are biased by the items’ popularity and our approach takes advantage of this characteristic, it is crucial to explore the possibility that the observed performance gains are solely attributed to this inherent bias. Consequently, it becomes imperative to investigate the presence of any potential biases in these results in order to ascertain their validity.

5.3 Counterfactual Evaluation

This section aims to investigate our second sub-question: *Has the improvement of these modified versions been caused by the usual popularity bias of offline datasets?* In this sense, we perform a counterfactual evaluation of the modified Contextual Bandits to evaluate them without the selection and exposure bias [55, 165]. Such biases usually occur in offline datasets because all user ratings were collected from the user interactions with items recommended by a specific recommendation policy (i.e., the model in production at the time of the user’s interaction). Counterfactual estimators enable using existing log data to estimate how some new target recommendation policy (i.e., a new approach) would have performed if it had been used instead of the policy that logged the data. It enables an Off-Policy Evaluation (OPE) akin to an unbiased offline A/B test [142].

However, the current counterfactual estimators require that we know the production policy used to create the dataset – which is not available for offline datasets. In this sense, we create a synthetic recommendation dataset based on a prior known policy to produce ratings from 1 to 5. For this experiment, we adapt the recently published Open Bandit Pipeline [189] to create a dataset following the same settings of the traditional MovieLens 100k. It means that we create 100k synthetic ratings for the same amount of users. In this experiment, each context contains the user-id and their features (i.e., genre, age, and occupation). The context is selected according to the user-id based on the same order as the MovieLens dataset. However, at each interaction, the items are selected based on a simple Linear Regression (the recommendation policy π_0) that performs a weighted selection according to a random sample of items. In this sense, the recommendations

policy is now known by our system and the recommended items are not fully biased by their popularity. The ratings from each user to the recommended item are simulated by also using a Linear policy. Table 5.5 highlights some statistics of this new dataset. It is limited to 1,000 items due to the current limitations of the OBP framework.

Synthetic Dataset	
# Users	943
# Items	1,000
Sparsity	93.75%
Rating mean	3.25 (std. 1.27)
Avg. ratings per user	62 (std. 60)
Avg. ratings per item	59 (std. 26)

Table 5.5: Synthetic Dataset statistics.

For the counterfactual evaluation, we get the synthetic dataset and split it into train and test by selecting the last 20% of users as the new ones (i.e., test set). Both original and modified methods are trained and executed with this synthetic dataset, making 100 recommendations of one item for each new user. Then, each policy π_e (i.e., a recommender) is evaluated by three offline performance estimators. These OPEs estimate how these other policies (i.e., recommendation systems) would have performed if they had been used instead of the original policy π_0 . All of them aim to mitigate the absence of rewards for each possible action. In the equations below, D represents the synthetic dataset, a is the action selected (i.e., the recommended item), x is the context of each action (i.e., the user identifier and their features), and n refers to the amount of data recorded in the dataset.

- (1) **Direct Method (DM)**: learns a model to estimate all missing rewards \hat{r} for every single action a in each context x . Its goal is to approximate the estimated reward $\hat{r}(x_i, a)$ by exploiting the logged dataset D as a traditional supervised machine learning problem. Then, it uses all imputed rewards for weighting the actions selected by the new policy as follows. The result is an estimator with a low variance but a high bias for the current data.

$$\hat{V}_{DM}(\pi_e; D, \hat{r}) := \frac{1}{n} \sum_{i=1}^n \sum_{a \in \mathcal{A}} \pi_e(a|x_i) \hat{r}(x_i, a)$$

- (2) **Inverse Propensity Score (IPS)**: weights the new policy with the known value of the original policy π_0 used to create the synthetic dataset as follows. If for all action a and context x it holds that $\pi_e(a|x_a) > 0 \Rightarrow \pi_0(a_i|x_i) > 0$, the result is unbiased and $\hat{V}_{IPS}(\pi_e; D) \rightarrow V(\pi_e)$. However, it also results in a high variance estimator due to the variability of this propensity score.

$$\hat{V}_{IPS}(\pi_e; D) := \frac{1}{n} \sum_{i=1}^n \frac{\pi_e(a|x_i)}{\pi_0(a_i|x_i)} \cdot r_i$$

- (3) **Doubly Robust (DR)**: combines the DM and IPS estimators to reduce the variance and perform better than both. It is unbiased and consistent once it applies a correction term in the propensity score. Moreover, it also has the potential to achieve an optimal variance once it applies the DM estimator.

$$\hat{V}_{DR}(\pi_e; D, \hat{r}) := \hat{V}_{DM}(\pi_e; D, \hat{r}) + \frac{1}{n} \sum_{i=1}^n \frac{\pi_e(a|x_a)}{\pi_0(a_i|x_i)} \cdot (r_i - \hat{r}(x_i, a_i))$$

Results of each estimator are presented in Table 5.6. The first three columns (named IPS, DM, and DR) contain the average policy value made by each estimator for each user. These values are the expected reward measured by the counterfactual estimators for each algorithm after recommending 100 items. Such values represent the algorithm performance without the position and selection bias removed by each estimator. All of these three estimators can be used to measure the quality of these bandit algorithms – the higher value the better performance. Complementary, the other three columns contain the confidence interval around the average value in the first three columns. These values are used to represent the distribution of the estimated rewards made for each user. As expected, the DM estimator is biased by the original policy π_0 used to create the dataset and it presents the smallest variance for all algorithms. On the other hand, the IPS estimator is not biased by the original policy π_0 but it has the highest variance for all recommendation policies. DR is the most unbiased and consistent value. The highlighted results show when the estimated value of our modified versions is higher than the original algorithm’s standard deviation upper bound (e.g., the value 3.911 for the Linear ϵ -Greedy $_{t=0}$ made by the DR estimator is higher than the upper bound of its original version: 3.793).

Dataset	Synthetic Dataset					
Measure	Estimated policy value			95.0% CI (lower) – 95.0% CI (upper)		
Estimators	IPS	DM	DR	IPS	DM	DR
Random	0.656	3.253	3.245	0.000 - 1.791	3.226 - 3.281	3.214 - 3.276
Most Popular	3.643	3.394	3.438	2.113 - 5.476	3.370 - 3.420	3.373 - 3.526
Linear ϵ -Greedy	2.685	3.775	3.768	0.838 - 4.894	3.732 - 3.816	3.743 - 3.793
Linear ϵ -Greedy $_{t=0}$	3.058	3.924	3.911	0.978 - 5.886	3.883 - 3.968	3.883 - 3.937
Linear ϵ -Greedy $_{t \geq 0}$	3.470	4.042	4.161	1.32 - 5.908	3.845 - 4.18	4.135 - 4.187
PTS	2.468	4.370	4.392	0.765 - 4.59	4.31 - 4.417	4.367 - 4.418
PTS $_{t=0}$	5.588	4.350	4.398	2.733 - 8.802	4.304 - 4.391	4.374 - 4.423
PTS $_{t \geq 0}$	5.802	4.320	4.365	3.117 - 8.666	4.276 - 4.357	4.34 - 4.393
LinUCB	2.651	4.352	4.371	0.939 - 4.635	4.33 - 4.373	4.356 - 4.386
LinUCB $_{t=0}$	3.392	4.627	4.620	1.331 - 5.857	4.593 - 4.673	4.611 - 4.629
LinUCB $_{t \geq 0}$	4.849	4.500	4.508	2.572 - 7.72	4.475 - 4.524	4.499 - 4.518

Table 5.6: Estimated policy values of the modified and original versions in a synthetic dataset. All modified strategies have outperformed the original ones. LinUCB has demonstrated the highest performance across all other algorithms.

Comparing the original algorithms with their modified versions (i.e., three by three in this table), our modified ones exhibit superior performance across all counterfactual estimators. Particularly, both Linear ϵ -Greedy and LinUCB algorithms demonstrate significant improvement when evaluated using the DR estimator. The only exception is the PTS algorithm. This outcome could be attributed to the synthetic dataset utilised for evaluating these estimators. As previously described, this dataset was generated by selecting the item with the highest probability from a randomly sampled set of items for each user during each interaction. This selection process closely resembles the PTS algorithm, which randomly chooses items based on their probability to maximise the expected rewards. Consequently, the original algorithm already performs well on this dataset. Furthermore, when considering all algorithms at the same time, the modified versions of LinUCB continue to exhibit the highest estimated policy value. These values underscore the quality of this algorithm as a strong candidate for real-world applications.

Dataset	Synthetic Dataset				
Measure	Relative policy value			Average Gains	
Estimators	IPS	DM	DR	All Estimators	DR
Random	0.199	0.990	0.987	▼ 28%	▼ 2%
Most Popular	1.109	1.033	1.046	▲ 5.5%	▲ 4.6%
Linear ϵ -Greedy	0.817	1.149	1.147	▲ 3%	▲ 14%
Linear ϵ -Greedy _{$t=0$}	0.931	1.194	1.190	▲ 10.5%	▲ 19%
Linear ϵ -Greedy _{$t \geq 0$}	1.056	1.230	1.266	▲ 18%	▲ 26%
PTS	0.751	1.330	1.337	▲ 13.5%	▲ 33%
PTS _{$t=0$}	1.701	1.324	1.338	▲ 45%	▲ 33%
PTS _{$t \geq 0$}	1.766	1.315	1.329	▲ 46%	▲ 32%
LinUCB	0.807	1.325	1.331	▲ 15%	▲ 33%
LinUCB _{$t=0$}	1.033	1.408	1.406	▲ 27.5%	▲ 40%
LinUCB _{$t \geq 0$}	1.476	1.370	1.372	▲ 40.5%	▲ 37%

Table 5.7: The relative value achieved by each algorithm considering the original policy π_0 used to create the synthetic dataset. Such gains represent if the new algorithm would perform better or not than the original recommender used to collect the offline dataset.

Complementary, Table 5.7 presents the relative value of each recommendation policy – how much this new policy π_e improved the original policy π_0 . Values higher than 1 mean that the policy would have performed better if it had replaced the original policy. The last two columns highlight the percentage of improvement achieved by each algorithm considering an average of all estimators and the DR estimator. Simple approaches like Random and Most Popular are not able to outperform the original policy – they would not have improved the model. This result reflects the meaning of a counterfactual evaluation. Even though the Most Popular approach performs satisfactorily in the traditional offline evaluation (see Table 5.4), it does not mean it would have performed better in the real world. The Most Popular recommendations are fully biased by the exposure bias

of the popular items on the traditional offline datasets. On the other hand, all modified Contextual Bandits would have improved the original policy. In this Table, we highlight all approaches that have significantly improved by more than 20% of the original recommendation policy. Most of these results refer to our modified approaches. It emphasises that our AL approach is really able to learn the user’s preferences and increase the gains of a bandit algorithm. Moreover, such results also answer our second subquestion by showing that the improvements achieved by our AL approach are not related to the usual popularity bias of offline datasets.

5.4 Baselines Comparison

Finally, this section aims to answer the third subquestion: *What is the effect of addressing uncertainty scenarios compared to existing state-of-the-art baselines?* In this sense, we compare our modified versions with strong baselines from the literature. We select all algorithms listed in bold in Section 5.1. They are:

- **Non-personalised algorithms:** Random and Most Popular;
- **MAB algorithms:** ϵ -Greedy, UCB, and TS;
- **Contextual Bandit Algorithms:** Linear ϵ -Greedy, Linear UCB, GLM-UCB, Lin-UCB, and PTS;
- **Interactive models:** NICF, Cluster-Bandit, and ICTRTS.

Table 5.8 shows the results of each algorithm for each recommendation domain. The first group of columns contain the average of the accumulated number of hits to represent the Cumulative Reward of each bandit algorithm. The second group of columns contain the ratio of relevant items recommended to represent the Recall of each bandit model. In this work, an item is relevant for a user if this item has received a rating higher than 4. Both metrics show the values achieved by each algorithm in different stages of the user journey – represented by the number T of interactions. A result in the column of $T = 20$, for instance, represents the cumulative reward or the corresponding recall after 20 recommendations of 1 item (i.e., 20 items recommended). While the first rows in this table refer to the baselines, the last three algorithms refer to the modified versions. All these algorithms are executed using the iRec framework [209], another marginal contribution of this work to the literature – see Appendix B for more details. The iRec enables the reproducibility of our entire evaluation process and it contains all the algorithms implemented in Python.

Dataset	Netflix									
Measure	Cumulative Reward					Recall				
T	5	10	20	50	100	5	10	20	50	100
Random	0.029	0.057	0.104	0.247	0.484	0.000	0.001	0.001	0.003	0.006
Popular	1.581▲	2.771	4.915	10.203	17.776	0.036▲	0.058▲	0.096	0.185	0.304
ϵ -Greedy	0.671	1.292	2.417	5.304	9.791	0.009	0.017	0.032	0.074	0.142
UCB	0.558	1.176	2.274	5.222	9.660	0.007	0.016	0.031	0.073	0.138
TS	1.006	1.944	3.535	7.502	12.981	0.014	0.027	0.049	0.103	0.180
LinearUCB	0.659	1.718	4.105	11.104	21.308	0.009	0.027	0.065	0.178	0.335
GLM-UCB	0.649	1.277	3.711	11.580	21.835	0.010	0.021	0.060	0.194	0.352
L. ϵ -Greedy	0.002	0.189	0.735	2.104	5.479	0.000	0.001	0.004	0.012	0.036
LinUCB	0.677	2.225	5.535	13.567	22.980	0.008	0.039	0.108	0.254	0.395
PTS	0.412	0.918	2.190	7.639	14.846	0.006	0.016	0.038	0.150	0.274
NICF	1.429	2.446	4.508	9.248	13.983	0.027	0.044	0.078	0.148	0.212
Cluster Bandit	0.571	1.230	3.132	7.420	13.882	0.007	0.016	0.053	0.119	0.209
ICTRTS	0.016	0.052	0.339	2.148	5.091	0.000	0.001	0.004	0.027	0.067
L. ϵ -Greedy $_{t \geq 0}$	0.330	0.971	2.985	10.274	20.004	0.004	0.012	0.043	0.175	0.332
PTS $_{t \geq 0}$	0.469	1.028	2.691	8.276	15.312	0.008	0.018	0.053	0.164	0.278
LinUCB $_{t \geq 0}$	1.065	2.777●	6.111▲	14.161▲	23.409▲	0.017	0.053	0.121▲	0.265▲	0.400▲
Dataset	Yahoo Music R1									
Measure	Cumulative Reward					Recall				
T	5	10	20	50	100	5	10	20	50	100
Random	0.015	0.032	0.072	0.192	0.390	0.000	0.001	0.001	0.004	0.008
Popular	1.593	2.902	5.080	10.581	17.432	0.043	0.076	0.131	0.266	0.425
ϵ -Greedy	0.602	1.460	3.080	7.424	13.360	0.014	0.036	0.076	0.183	0.323
UCB	0.514	1.358	3.018	7.330	13.277	0.013	0.034	0.075	0.179	0.321
TS	0.957	1.907	3.697	8.356	14.720	0.025	0.048	0.093	0.204	0.354
LinearUCB	1.571	3.265●	6.604	15.208	24.906	0.044●	0.091●	0.177	0.380	0.588
GLM-UCB	0.910	1.634	4.955	14.197	24.328	0.026	0.045	0.133	0.358	0.576
L. ϵ -Greedy	0.021	0.059	0.187	0.985	3.278	0.001	0.001	0.004	0.018	0.057
LinUCB	0.746	1.475	5.702	16.662	26.538	0.018	0.035	0.149	0.419	0.625
PTS	0.902	2.267	5.649	15.263	23.333	0.021	0.055	0.141	0.378	0.555
NICF	1.661●	3.192	5.795	11.218	14.954	0.043	0.084	0.150	0.281	0.362
Cluster Bandit	0.996	2.428	4.704	10.095	16.777	0.028	0.064	0.123	0.254	0.413
ICTRTS	0.011	0.138	1.167	6.149	13.446	0.000	0.003	0.028	0.153	0.327
L. ϵ -Greedy $_{t \geq 0}$	1.190	2.459	5.355	13.208	22.867	0.029	0.059	0.129	0.302	0.504
PTS $_{t \geq 0}$	0.657	1.722	4.857	14.994	24.112	0.015	0.041	0.119	0.372	0.57
LinUCB $_{t \geq 0}$	1.411	2.509	7.395●	18.057▲	27.425▲	0.036	0.063	0.196●	0.454▲	0.649▲
Dataset	Good Books									
Measure	Cumulative Reward					Recall				
T	5	10	20	50	100	5	10	20	50	100
Random	0.039	0.077	0.155	0.375	0.757	0.001	0.001	0.002	0.005	0.010
Popular	1.238	2.231	4.246	7.919	11.993	0.017	0.030	0.057	0.105	0.157
ϵ -Greedy	0.445	0.792	1.424	3.121	5.721	0.006	0.011	0.019	0.041	0.075
UCB	0.404	0.742	1.342	2.990	5.579	0.006	0.010	0.018	0.040	0.073
TS	0.792	1.393	2.354	4.605	7.297	0.011	0.019	0.031	0.061	0.096
LinearUCB	0.451	0.976	2.741	7.467	13.557	0.006	0.013	0.037	0.098	0.175
GLM-UCB	0.347	0.757	1.600	7.184	13.519	0.005	0.010	0.021	0.094	0.176
L. ϵ -Greedy	0.026	0.039	0.092	0.282	0.723	0.001	0.001	0.002	0.004	0.011
LinUCB	0.651	0.889	3.572	9.541	15.758	0.009	0.012	0.048	0.124	0.202
PTS	0.841	2.047	4.159	7.669	12.121	0.011	0.027	0.056	0.102	0.160
NICF	1.379▲	2.547▲	4.362	7.488	10.400	0.019▲	0.035▲	0.059	0.099	0.136
Cluster Bandit	0.944	1.792	4.001	7.577	11.835	0.013	0.024	0.054	0.101	0.155
ICTRTS	0.330	1.054	2.723	6.897	11.251	0.004	0.014	0.036	0.091	0.148
L. ϵ -Greedy $_{t \geq 0}$	0.404	0.997	2.026	6.164	12.794	0.005	0.013	0.026	0.079	0.165
PTS $_{t \geq 0}$	0.993	2.324	4.507	8.492	13.095	0.014	0.031	0.061	0.113	0.172
LinUCB $_{t \geq 0}$	1.174	1.893	5.435▲	11.258▲	17.548▲	0.016	0.025	0.072▲	0.147▲	0.226▲

Table 5.8: Cumulative reward of all baselines and the three modified versions made by this work. Results show a statistical improvement in the modified algorithms by applying the Wilcoxon test with a p-value = 0.05. The symbol ▲ represents significant gains, ● represents statistical draws, and ▼ represents significant losses.

The results of our experiments reveal a statistically significant superiority of at least one of our modified versions over all other baseline models after the initial interactions. As our new Active Learning approach introduces some level of entropy in the initial

recommendations, it was expected that other models would outperform our modified versions in the short term. Besides that, even during these initial interactions, our algorithms demonstrate a cumulative reward that remains relatively close to the fully biased models, such as the Most Popular or NICF. Such performance highlights the fact that our approach seeks exploration in the first interactions but does not forget to exploit the information previously available. Moreover, by balancing this trade-off between exploration and exploitation, our approach is also responsible for an expressive improvement in the modified versions after the 20th interaction across all datasets. The main reason for such gain is related to the knowledge achieved in the first user’s interactions. In other words, our modified versions are able to learn more about the users’ preferences while they are still recommending relevant items for the users. Thus, our modified versions outperform all other baseline models in the user long run.

Moreover, considering each one of the modified versions proposed by this work, we can also highlight that:

1. The Linear ϵ -Greedy algorithm has been transformed into a highly competitive option. Before our modifications, the Linear ϵ -Greedy was one of the worst models, taking many interactions ($T \gg 100$) to learn user preferences. After it, this simple algorithm outperforms other approaches that take so much time to execute, like the Cluster-Bandit; and approaches that require so much effort to calibrate their parameters, like the NICF, ICTRTS, and PTS.
2. The modified PTS model becomes more competitive with the other algorithms in the long run. With the addition of entropy to the already biased popularity model, it learns more about user preferences and thus maximises their experience.
3. Especially, the modified LinUCB outperforms all of the state-of-the-art algorithms in the long run by achieving a bigger cumulative reward. After mitigating both scenarios of uncertainty, this method achieves better results even than strategies developed with similar assumptions, like Linear UCB and GLM-UCB.

Such results answer our third research question by highlighting the effect of addressing scenarios of uncertainty about the user’s preferences in Contextual Bandits. The modified versions of traditional algorithms improve the user boundaries of accuracy in the long run.

5.5 Summary

This chapter aimed to answer the third research question related to the impact of adding more knowledge about the users when the Contextual Bandit is uncertain about their preferences. Then, we propose an experimental setup based on three distinct recommendation domains: movies, songs, and books. For each dataset, we simulate both scenarios of uncertainty by selecting the last users that joined our system as new users. All information about these new users was removed from the dataset and the algorithms were trained with the remaining data collected before the first interaction of these new users. After that, we raised three subquestions to guide our experiments.

Answering the first subquestion we highlighted that our modified versions proposed in the last chapter are statistically superior to their original versions. In all datasets, addressing the pure cold-start and/or the misleading assumptions, the bandit models have become better than their original versions. Then, answering the second subquestion we perform a counterfactual evaluation in a synthetic and unbiased dataset. The idea of such analysis was to prove that the aforementioned gains of our modified versions were not only caused by the usual bias of the offline datasets. Experiments with distinct counterfactual estimators have supported the previous experiments and highlighted the potential of the modified LinUCB algorithm. Indeed, answering our third subquestion by comparing the impact of the modified versions with existing state-of-the-art baselines, we notice that LinUCB has outperformed all baselines for all datasets. Such results support our assumption that addressing scenarios of uncertainty can contribute to improving the user's experience in the long run.

Chapter 6

Conclusions & Future Work

This chapter starts by restating our statement of thesis, its relevance, and the research questions raised in this study. Then, we provide a synthesis of empirical findings with respect to the underlying questions. Next, we summarise the main contributions and the limitations of this dissertation. Then, we present promising research directions to be explored in the future. The chapter ends with our final remarks.

6.1 Restatement of Thesis

Recent works have addressed Recommendation Systems (RSs) as an interactive scenario through a *Contextual Bandit* model. In this case, items are modelled as arms to be pulled and selecting an arm is equivalent to recommending an item. The reward is the user's feedback on that recommendation (e.g., clicks, acceptance, satisfaction, etc). At each iteration, the model should mitigate the dilemma between (1) *exploiting* the arm that seems the best option so far; and (2) *exploring* an arm not yet tried out (or not tried enough). However, the personalisation level of such bandit models still is dependent on the amount of information available to represent the users. Scenarios where the model is uncertain of the users' preferences may introduce some bias in the balance of exploration and exploitation. In this work, we studied two of these scenarios. The first one happens when a user joins for the first time. The system is not able to exploit or explore their unknown preferences. The second one happens when the current knowledge about the users is unreliable because their profiles were not correctly defined or their preferences suddenly changed. In this case, the system keeps exploiting wrong assumptions that lead to unsatisfactory recommendations and a simple exploration strategy is not enough to relearn the user's profile in an acceptable time. In light of this context, we explore the following statement:

Exploring scenarios of uncertainty where the information about the user is not straightforward or even does not exist increases the accuracy of Contextual Bandit models in the long term.

Especially, we raised three main research questions that guided our study. The empirical findings related to each question are discussed in the next section.

6.2 Empirical Findings

In order to handle the complexity inherent to the above statement, we split our investigation into three main research questions (RQ). We extensively investigated each of these questions in each one of the previous chapters. This section synthesises these findings to validate our study.

- **RQ1:** *How have current Contextual Bandit assumptions about uncertainty scenarios impacted the users' experience?*
 - a) A Systematic Literature Review on all journals and conference papers published from 2000 to 2020 highlighted that most of the approaches are still assuming a linear assumption to represent users and items in Contextual Bandit models. By inspecting 1327 papers and deeply studying 230 of them, we identified the usual implementation pattern of the traditional MAB approaches in Recommender Systems (see Table 3.1).
 - b) Studying such assumptions in Section 3.2, we identified a limitation from these Contextual Bandit models to handle scenarios of uncertainty about the user's preferences. When facing the pure cold-start problem or misleading assumptions, such models are not able to correctly balance the trade-off between exploration and exploitation (see Section 3.3).
 - c) An empirical methodology created to simulate such limitations has demonstrated the long-term impact of such scenarios (see Section 3.4). On average, pure exploration approaches have required that users rate more than 8,000 items ($\sim 2,000$ interactions \times 5 items) for hitting 80% of the user's history. In turn, pure exploitation approaches have limited the accuracy of bandit models because they are not able to get the required knowledge in the first interactions.
- **RQ2:** *How can exploration and exploitation be addressed in Contextual Bandit models when user preferences are uncertain?*
 - a) This work proposes to address exploration and exploitation through Active Learning approaches. Such approaches aim to ask for the user's feedback on data points that are the most informative to the Machine Learning model. Our objective is to leverage this AL concept when the Contextual Bandit model encounters some uncertainty regarding the user's preferences.

- b) This novel approach address exploration and exploitation by balancing the item’s popularity and its entropy. While using popularity means increasing the probability that a user will rate an item (i.e., exploitation), applying entropy means increasing the amount of possible information to be achieved if the user rates an item (i.e., exploration).
 - c) Our proposal does not intend to change the successful linear assumption made by current Contextual Bandits. It aims to introduce Active Learning in an initial context to mitigate the pure cold-start problem and change the recommendation engine when the system has made misleading assumptions. We demonstrated how to apply our approach in three distinct bandit algorithms (see Algorithms in Section 4.3).
- **RQ3:** *What is the impact of adding more knowledge about the users when the Contextual Bandit model is uncertain of their preferences?*
- a) First, we measured the impact of adding Active Learning to Contextual Bandits by comparing the new algorithms with the original ones. Results demonstrate that our new approach is able to provide significant improvements achieving statistical gains over all algorithms (see Table 5.4).
 - b) Then, we demonstrated that such improvements are not related to the popularity bias of offline datasets. Applying a counterfactual evaluation in the results of our algorithms in a synthetic and unbiased dataset highlighted gains of over 25% on average (see Table 5.7).
 - c) Finally, we contrasted our modified algorithms to strong baselines in the literature and demonstrated significant gains in the user’s long run. In particular, our approach has made even simple algorithms, such as Linear ϵ -Greedy, achieve similar or better results than complex and powerful techniques of neural networks, such as NICF. Moreover, the modified version of the LinUCB has outperformed all baselines analysed.

6.3 Summary of Contributions

The main contributions could be classified into three main groups.

- **Concepts and Problems:** This study introduces novel concepts and investigates emerging challenges intrinsic to the recommendation task within Contextual Bandits models. Firstly, we elucidate two well-known hurdles from the literature on Recommender Systems (RSs) – pure cold-start and misleading assumptions – through the

lens of current bandit models. Then, we integrate some concepts from the Active Learning theory into Contextual Bandit algorithms to address the trade-off between exploration and exploitation in recommendation domains.

- **Algorithms:** We propose a new approach to be applied in current Contextual Bandits based on Active Learning concepts to effectively mitigate these two aforementioned scenarios of uncertainty. This approach is not dependent on any personal information about the users and can be instantiated for most of the Contextual Bandit models. In this work, we create new versions for three distinct bandit algorithms.
- **Understanding and Knowledge:** We acquired further knowledge about the importance of addressing the user’s preferences correctly to mitigate the system uncertainty in Contextual Bandit models. Our modified versions of traditional bandit algorithms have improved the user’s experience in the long run in three distinct domains. Moreover, we also confirmed the value of a counterfactual evaluation to ensure that our conclusions are not biased by the offline datasets.

Furthermore, we point out a new and relevant constraint to be addressed for other researchers in the recommendation field within Contextual Bandits. To the best of our knowledge, this is the first effort to mitigate scenarios of user uncertainty in these interactive systems. We reported all of these findings along distinct publications [208, 204, 209, 205, 207, 206].

6.4 Limitations of the Work

The study has offered an evaluative perspective on the algorithmic limitations of Contextual Bandits unaddressed by the literature. As a direct consequence of this methodology, the study faced a number of limitations, which need to be discussed.

- **Scope of this work:** Our discussions are related to scenarios of uncertainty about the user’s preferences in recommendation systems. However, this work only addresses the pure cold-start problem and the misleading assumptions. Certainly, there are other scenarios of uncertainty where we are unable to claim if our approach will mitigate them. Moreover, the scenario of misleading assumptions can be represented by other different strategies. In this work, for instance, we have mentioned that this concept may also be related to the cold-start, dynamic user preferences, or even the shared account problem. Finally, we can also mention the absence of personalised Active Learning strategies due to the scope of uncertainty that is mostly related to the absence of reliable user information.

- **Extent of results:** Our findings are purely based on heuristic assessments. Hence, we cannot make strong claims about the best approach to address the two scenarios of uncertainty studied in this work. Moreover, despite testing in three distinct domains, we cannot ensure that the results are extensible to all other domains where recommendation systems can be applied. Finally, despite the easy applicability of our new contextual approach, we are unable to ensure that it will work for non-linear algorithms that were not tested in this dissertation.
- **Implementation decisions:** For the sake of efficiency, we adopted several simplistic decisions that should be refined to handle the actual conditions of recommendation domains. For instance, we have studied the impact of scenarios of uncertainty in linear bandit models. Despite our SLR highlighting them as the most common assumption for Contextual Bandits in recent years, we are not sure about this impact on algorithms with other assumptions.
- **Experimental design:** For the sake of efficiency and to ensure the long-term experience of each user, we have sampled some of the datasets to only include 10,000 users with more than 20 items rated. Based on our tests, it does not have any implication for the algorithms' performance, but we should properly test them in huge datasets to make strong claims about our conclusions. Moreover, it is still unclear the necessary conditions to achieve the same results in an online environment. Despite our effort to perform a counterfactual evaluation of our new algorithms, such methodology is just an approximation of online experiments. Sometimes, this work uses the term "user's experience" to refer to the performance of an algorithm for a specific user, but we are aware that other factors besides the recommendation engine are also essential to ensure the best experience for a user in real-world applications.

6.5 Future Research

Aligned with the foregoing discussion about the limitations of this dissertation, we highlight as possible directions of future work three main branches.

1. **Online Experiment:** This study investigates the potential of Active Learning in enhancing the user experience in interactive recommendation systems. To advance this research, the next logical step involves conducting an online evaluation to monitor user choices from initial interactions to subsequent ones. Our idea is to select a conventional dataset and conduct an experiment that closely aligns with the methodology described in Section 5.1. In order to investigate scenarios of uncertainty, such as the pure cold-start and the misleading assumptions, the entire dataset can be

used as the training set for our algorithms that will act by just incorporating new users for each person who participates in the experiment. As our strategies do not rely on any prior knowledge about the users, any person is available to participate in this experiment. The huge challenges will be: (1) developing a real-time responsive system capable of generating new recommendations at each interaction and dynamically incorporating user feedback; and (2) ensuring that each user interacts for as many interactions as possible to enable a comprehensive evaluation of the long-term user's preferences.

2. **Personalised Active Learning:** In Section 4.1, we provided a comprehensive overview of the most popular strategies in the Active Learning theory, categorising them as either personalised or non-personalised methods. As the main objective of this work is on scenarios of uncertainty, where reliable user information is scarce, we have specifically chosen non-personalised algorithms. However, it is also important to acknowledge the potential of personalised strategies in Contextual Bandits models. Therefore, we strongly encourage other researchers to explore this direction by integrating Active Learning algorithms into the interactive recommendation scenario. The modifications we have made through the adoption of non-personalised strategies have already demonstrated the promising potential of such approaches.
3. **Scenarios of uncertainty:** This study highlights the significance of effectively handling the uncertainty about the user in Contextual Bandit models. Given the limited scope and time constraints of this research, we have primarily focused on two specific scenarios: the pure cold-start problem and the presence of misleading assumptions about users. However, it is important to acknowledge the existence of additional scenarios documented in the literature that warrant exploration. Specifically, we have touched upon the Dynamic User Preferences and the Shared Account problem, both of which hold considerable relevance in contemporary real-world entertainment applications. In this sense, we encourage other researchers to delve into these issues by employing alternative strategies derived from the Active Learning theory. To the best of our knowledge, the exploration of such problems in this particular context remains relatively uncharted territory.

6.6 Final Remarks

In summary, this work showed that the uncertainty about the user's preferences must be addressed in Contextual Bandit models when applied to the feedback problem in Recommendation Systems. By incorporating concepts from the Active Learning the-

ory in such scenarios, we can attain a more comprehensive understanding of the user that leads to more relevant items being recommended during the user’s journey. Indeed, offline experiments in distinct domains and a counterfactual evaluation in an unbiased dataset have highlighted the potential enhancements achievable through these strategies. It is worth emphasising that even straightforward strategies, when appropriately applied at the opportune moment, can introduce substantial improvements to existing models.

Bibliography

- [1] Personalized pricing recommender system: Multi-stage epsilon-greedy approach. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 57–64, 2011. URL <https://dl.acm.org/doi/abs/10.1145/2039320.2039329>.
- [2] Multi-armed bandits to recommend for cold-start user. In *Proceedings of the 4rd Symposium on Knowledge Discovery, Mining and Learning*, 2016. URL <http://www.academia.edu/download/53742655/mabrec.pdf>.
- [3] Selecting multiple web adverts: A contextual multi-armed bandit with state uncertainty. *Journal of the Operational Research Society*, 71(1):100–116, 2020. URL <https://www.tandfonline.com/doi/abs/10.1080/01605682.2018.1546650>.
- [4] Marc Abeille and Alessandro Lazaric. Linear thompson sampling revisited. In *Artificial Intelligence and Statistics*, pages 176–184. PMLR, 2017.
- [5] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.
- [6] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.
- [7] Charu C Aggarwal and Charu C Aggarwal. Knowledge-based recommender systems. *Recommender Systems: The Textbook*, pages 167–197, 2016.
- [8] Michal Aharon, Amit Kagian, Yohay Kaplan, Raz Nissim, and Oren Somekh. Serving ads to " yahoo answers" occasional visitors. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1257–1262, 2015.
- [9] Rabaa Alabdulrahman, Herna Viktor, and Eric Paquet. Active learning and deep learning for the cold-start problem in recommendation system: A comparative study. In *International Joint Conference on Knowledge Discovery, Knowledge Engineering, and Knowledge Management*, pages 24–53. Springer, 2019.
- [10] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Azzurra Ragone, and Joseph Trotta. Local popularity and time in top-n recommendation. In *European Conference on Information Retrieval*, pages 861–868. Springer, 2019.

- [11] Vito Walter Anelli, Alejandro Bellogín, Antonio Ferrara, Daniele Malitesta, Felice Antonio Merra, Claudio Pomo, Francesco Maria Donini, and Tommaso Di Noia. Elliot: a comprehensive and rigorous framework for reproducible recommender systems evaluation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [12] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.
- [13] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
- [14] Vivek Bagaria, Govinda Kamath, Vasilis Ntranos, Martin Zhang, and David Tse. Medoids in almost-linear time via multi-armed bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 500–509. PMLR, 2018.
- [15] Avinash Balakrishnan, Djallel Bouneffouf, Nicholas Mattei, and Francesca Rossi. Using multi-armed bandits to learn ethical priorities for online ai systems. *IBM Journal of Research and Development*, 63(4/5):1–1, 2019.
- [16] Ramesh Baral, S. S. Iyengar, Xiaolong Zhu, Tao Li, and Pawel Sniatala. Hires: A hierarchical contextual location recommendation system. volume 6, pages 1020–1037, 2019. doi: 10.1109/TCSS.2019.2938239.
- [17] Iman Barjasteh, Rana Forsati, Farzan Masrour, Abdol-Hossein Esfahanian, and Hayder Radha. Cold-start item and user recommendation with decoupled completion and transduction. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 91–98. ACM, 2015. doi: <https://doi.org/10.1145/2792838.2800196>.
- [18] Andrea Barraza-Urbina and Dorota Glowacka. Introduction to bandits in recommender systems. In *Fourteenth ACM RecSys*, pages 748–750, 2020.
- [19] Andrea Barraza-Urbina, Georgia Koutrika, Mathieu d’Aquin, and Conor Hayes. Bears: Towards an evaluation framework for bandit-based interactive recommender systems. *REVEAL 18, October 6-7, 2018, Vancouver, Canada*, 2018.
- [20] Soumya Basu, Rajat Sen, Sujay Sanghavi, and Sanjay Shakkottai. Blocking bandits. In *Advances in Neural Information Processing Systems*, pages 4784–4793, 2019. URL <http://papers.nips.cc/paper/8725-blocking-bandits>.
- [21] Alejandro Bellogín and Pablo Sánchez. Revisiting neighbourhood-based recommenders for temporal scenarios. In *RecTemp RecSys*, pages 40–44, 2017.

- [22] Lucas Bernardi, Jaap Kamps, Julia Kiseleva, and Melanie JI Mueller. The continuous cold start problem in e-commerce recommender systems. *arXiv preprint arXiv:1508.01177*, 2015.
- [23] Lucas Bernardi, Pablo Estevez, Matias Eidis, and Eqbal Osama. Recommending accommodation filters with online learning. 2715, 2020. URL <http://ceur-ws.org/Vol-2715/paper3.pdf>.
- [24] Daniel Billsus and Michael J Pazzani. User modeling for adaptive news access. *User modeling and user-adapted interaction*, 10(2-3):147–180, 2000.
- [25] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Jesús Bernal. A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-based systems*, 26:225–238, 2012.
- [26] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-Based Systems*, 2013.
- [27] Svetlin Bostandjiev, John O’Donovan, and Tobias Höllerer. Tasteweights: a visual interactive hybrid recommender system. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 35–42, 2012.
- [28] Djallel Bouneffouf. Freshness-aware thompson sampling. In *International Conference on Neural Information Processing*, pages 373–380. Springer, 2014. URL https://link.springer.com/chapter/10.1007/978-3-319-12643-2_46.
- [29] Djallel Bouneffouf. Contextual bandit algorithm for risk-aware recommender systems. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 4667–4674. IEEE, 2016.
- [30] Djallel Bouneffouf and Emmanuelle Claeys. Learning exploration for contextual bandit. In *AutoML ICML 2019: 6th ICML Workshop on Automated Machine Learning*, 2016.
- [31] Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gançarski. A contextual-bandit algorithm for mobile context-aware recommender system. In *International conference on neural information processing*, pages 324–331. Springer, 2012.
- [32] Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gançarski. Hybrid- ϵ -greedy for mobile context-aware recommender system. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 468–479. Springer, 2012.
- [33] Djallel Bouneffouf, Amel Bouzeghoub, and Alda Gançarski. Following the user’s interests in mobile context-aware recommender systems: The hybrid- ϵ -greedy algorithm. pages 657–662, 03 2012. doi: 10.1109/WAINA.2012.200.

- [34] Djallel Bouneffouf, Romain Laroche, Tanguy Urvoy, Raphael Féraud, and Robin Allesiardo. Contextual bandit for active learning: Active thompson sampling. In *International Conference on Neural Information Processing*, pages 405–412. Springer, 2014.
- [35] Djallel Bouneffouf, Irina Rish, Guillermo A Cecchi, and Raphaël Féraud. Context attentive bandits: Contextual bandit with restricted context. *arXiv preprint arXiv:1705.03821*, 2017.
- [36] Djallel Bouneffouf, Irina Rish, and Charu Aggarwal. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.
- [37] Guy Bresler, George Chen, and Devavrat Shah. A latent source model for online collaborative filtering. *Advances in Neural Information Processing Systems*, 4, 10 2014.
- [38] Björn Brodén, Mikael Hammar, Bengt J Nilsson, and Dimitris Paraschakis. Ensemble recommendations via thompson sampling: an experimental study within e-commerce. In *23rd International Conference on Intelligent User Interfaces*, pages 19–29, 2018. URL <https://dl.acm.org/doi/abs/10.1145/3172944.3172967>.
- [39] Björn Brodén, Mikael Hammar, Bengt J Nilsson, and Dimitris Paraschakis. A bandit-based ensemble framework for exploration/exploitation of diverse recommendation components: An experimental study within e-commerce. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 10(1):1–32, 2019.
- [40] Rocío Cañamares, Marcos Redondo, and Pablo Castells. Multi-armed recommender system bandit ensembles. In *Proceedings of the 13th ACM RecSys*, pages 432–436, 2019. URL <https://dl.acm.org/doi/abs/10.1145/3298689.3346984>.
- [41] Erion Çano and Maurizio Morisio. Hybrid recommender systems: A systematic literature review. *Intelligent Data Analysis*, 21(6):1487–1524, 2017.
- [42] Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie. Nearly optimal adaptive procedure with change detection for piecewise-stationary bandit. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 418–427, 2019. URL <http://proceedings.mlr.press/v89/cao19a.html>.
- [43] Stéphane Caron and Smriti Bhagat. Mixing bandits: A recipe for improved cold-start recommendations in a social network. In *Proceedings of the 7th Workshop on Social Network Mining and Analysis*, pages 1–9, 2013. URL <https://dl.acm.org/doi/abs/10.1145/2501025.2501029>.

- [44] Mario Casillo, Brij B Gupta, Marco Lombardi, Angelo Lorusso, Domenico Santaniello, and Carmine Valentino. Context aware recommender systems: A novel approach based on matrix factorization and contextual bias. *Electronics*, 11(7):1003, 2022.
- [45] Pablo Castells and Dietmar Jannach. Recommender systems: A primer. *arXiv preprint arXiv:2302.02579*, 2023.
- [46] Pablo Castells, Saúl Vargas, and Jun Wang. Novelty and diversity metrics for recommender systems: Choice, discovery and relevance. *Proceedings of International Workshop on Diversity in Document Retrieval (DDR)*, 01 2011.
- [47] Pablo Castells, Neil J Hurley, and Saul Vargas. Novelty and diversity in recommender systems. In *Recommender systems handbook*, pages 881–918. Springer, 2015.
- [48] L Elisa Celis, Sayash Kapoor, Farnood Salehi, and Nisheeth Vishnoi. Controlling polarization in personalization: An algorithmic framework. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 160–169, 2019.
- [49] Nicolo Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. A gang of bandits. In *Advances in Neural Information Processing Systems*, pages 737–745, 2013.
- [50] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. In *Proceedings of the 12th ACM conference on recommender systems*, pages 224–232, 2018.
- [51] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257, 2011.
- [52] Niladri Chatterji, Vidya Muthukumar, and Peter Bartlett. Osom: A simultaneously optimal algorithm for multi-armed and linear contextual bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 1844–1854, 2020.
- [53] Chao Chen, Dongsheng Li, Junchi Yan, and Xiaokang Yang. Modeling dynamic user preference via dictionary learning for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 34(11):5446–5458, 2022. doi:10.1109/TKDE.2021.3050407.
- [54] Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. Efficient neural matrix factorization without sampling for recommendation. *ACM Transactions on Information Systems (TOIS)*, 38(2):1–28, 2020.

- [55] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240*, 2020.
- [56] Lin Chen, Andreas Krause, and Amin Karbasi. Interactive submodular bandit. In *Advances in Neural Information Processing Systems*, pages 141–152, 2017.
- [57] Lixing Chen, Jie Xu, and Zhuo Lu. Contextual combinatorial multi-armed bandits with volatile arms and submodular reward. In *Advances in Neural Information Processing Systems*, pages 3247–3256, 2018.
- [58] Mingang Chen and Pan Liu. Performance evaluation of recommender systems. *International Journal of Performability Engineering*, 13(8), 2017.
- [59] Minmin Chen, Bo Chang, Can Xu, and Ed H Chi. User response models to improve a reinforce recommender system. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 121–129, 2021.
- [60] Shulong Chen and Yuxing Peng. Matrix factorization for recommendation with explicit and implicit feedback. *Knowledge-Based Systems*, 158:109–117, 2018.
- [61] Chien Ming Chi, Hsuan Tien Lin, and Ching Kang Ing. Online clustering of bandits with high-dimensional sparse relevant user features. 2019.
- [62] Keunho Choi, Donghee Yoo, Gunwoo Kim, and Yongmoo Suh. A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis. *Electronic Commerce Research and Applications*, 11(4):309–317, 2012.
- [63] Akshay Chougule, Sushant Gawade, Nimit Dave, and Aruna Pavate. Fitness solution using hybrid algorithm. 03 2018.
- [64] Konstantina Christakopoulou and Arindam Banerjee. Learning to interact with users: A collaborative-bandit approach. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 612–620. SIAM, 2018. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611975321.69>.
- [65] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 815–824, 2016.
- [66] Koby Crammer and Claudio Gentile. Multiclass classification with bandit feedback using adaptive regularization. volume 90, pages 273–280, 01 2011. doi: 10.1007/s10994-012-5321-8.

- [67] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46, 2010.
- [68] Paolo Cremonesi, Franca Garzotto, Sara Negro, Alessandro Vittorio Papadopoulos, and Roberto Turrin. Looking for good recommendations: A comparative evaluation of recommender systems. In *IFIP Conference on Human-Computer Interaction*, pages 152–168. Springer, 2011.
- [69] Luis M De Campos, Juan M Fernández-Luna, Juan F Huete, and Miguel A Rueda-Morales. Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks. *International Journal of Approximate Reasoning*, 51(7):785–799, 2010.
- [70] Yashar Deldjoo, Markus Schedl, Balázs Hidasi, Yinwei Wei, and Xiangnan He. Multimedia recommender systems: Algorithms and challenges. In *Recommender systems handbook*, pages 973–1014. Springer, 2021.
- [71] Christian Desrosiers and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. *Recommender systems handbook*, pages 107–144, 2011.
- [72] John Duchi. Cs229 supplemental lecture notes hoeffding’s inequality, 2017.
- [73] Bianca Dumitrascu, Karen Feng, and Barbara Engelhardt. Pg-ts: Improved thompson sampling for logistic contextual bandits. *Advances in neural information processing systems*, 31:4624–4633, 2018.
- [74] Simen Eide and Ning Zhou. Deep neural network marketplace recommenders in online experiments. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 387–391, 2018.
- [75] Mehdi Elahi, Francesco Ricci, and Neil Rubens. A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, 20:29–50, 2016.
- [76] Alexander Felfernig and Robin Burke. Constraint-based recommender systems: technologies and research issues. In *Proceedings of the 10th international conference on Electronic commerce*, pages 1–10, 2008.
- [77] Crícia Felício, Klérisson Paixão, Celia Barcelos, and Philippe Preux. A multi-armed bandit model selection for cold-start user recommendation. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, 2017.

- [78] Gerhard Friedrich and Markus Zanker. A taxonomy for generating explanations in recommender systems. *AI Magazine*, 32(3):90–98, 2011.
- [79] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 257–260, 2010.
- [80] Johannes Gehrke, Michael Hay, Edward Lui, and Rafael Pass. Crowd-blending privacy. In *Annual Cryptology Conference*, pages 479–496. Springer, 2012.
- [81] Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *International Conference on Machine Learning*, pages 757–765, 2014.
- [82] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. On context-dependent clustering of bandits. In *International Conference on Machine Learning*, pages 1253–1262. PMLR, 2017.
- [83] Sahin Cem Geyik, Vijay Dialani, Meng Meng, and Ryan Smith. In-session personalization for talent search. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 2107–2115, 2018.
- [84] Luis Gomes, Carlos Almeida, and Zita Vale. Recommendation of workplaces in a coworking building: A cyber-physical approach supported by a context-aware multi-agent system. *Sensors*, 20(12):3597, 2020.
- [85] Anjan Goswami, Chengxiang Zhai, and Prasant Mohapatra. Learning to diversify for e-commerce search with multi-armed bandit. In *eCOMSIGIR*, 2019. URL <http://ceur-ws.org/Vol-2410/paper18.pdf>.
- [86] Frédéric Guillou, Romaric Gaudel, and Philippe Preux. Scalable explore-exploit collaborative filtering. In *Pacific Asia Conference On Information Systems (PACIS)*. Association For Information System, 2016.
- [87] Lei Guo, Jinyu Zhang, Tong Chen, Xinhua Wang, and Hongzhi Yin. Reinforcement learning-enhanced shared-account cross-domain sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–14, 2022. doi: 10.1109/TKDE.2022.3185101.
- [88] Saurabh Gupta, Bharathan Balaji, and Runfei Luo. CPR: collaborative pairwise ranking for online list recommendations. 2715, 2020. URL <http://ceur-ws.org/Vol-2715/paper9.pdf>.
- [89] Nicolas Gutowski, Tassadit Amghar, Olivier Camp, and Slimane Hammoudi. A framework for context-aware service recommendation for mobile users: A focus on mobility in smart cities. *From Data To Decision*, 2017.

- [90] Nicolas Gutowski, Tassadit Amghar, Olivier Camp, and Fabien Chhel. Global versus individual accuracy in contextual multi-armed bandit. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 1647–1654, 2019.
- [91] Nicolas Gutowski, Tassadit Amghar, Olivier Camp, and Fabien Chhel. Gorthaur: A portfolio approach for dynamic selection of multi-armed bandit algorithms for recommendation. pages 1164–1171, 2019.
- [92] Nicolas Gutowski, Olivier Camp, Tassadit Amghar, and Fabien Chhel. Using individual accuracy to create context for non-contextual multi-armed bandit problems. In *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–6. IEEE, 2019.
- [93] Nicolas Gutowski, Olivier Camp, Fabien Chhel, Tassadit Amghar, and Patrick Albers. Improving bandit-based recommendations with spatial context reasoning: An online evaluation. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1366–1373. IEEE, 2019.
- [94] Mark Isaac Hammond, Keen McEwan Browne, Mike Estee, and Clara Kliman-Silver. Multiple user interfaces of an artificial intelligence system to accommodate different types of users solving different types of problems with artificial intelligence, January 26 2017. US Patent App. 15/417,033.
- [95] Awni Hannun, Brian Knott, Shubho Sengupta, and Laurens van der Maaten. Privacy-preserving contextual bandits. *arXiv preprint arXiv:1910.05299*, 2019.
- [96] Botao Hao, Yasin Abbasi Yadkori, Zheng Wen, and Guang Cheng. Bootstrapping upper confidence bound. In *Advances in Neural Information Processing Systems*, pages 12123–12133, 2019. URL <http://papers.nips.cc/paper/9382-bootstrapping-upper-confidence-bound>.
- [97] Bowen Hao, Jing Zhang, Hongzhi Yin, Cuiping Li, and Hong Chen. Pre-training graph neural networks for cold-start users and items representation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 265–273, 2021.
- [98] Negar Hariri, Bamshad Mobasher, and Robin Burke. Context adaptation in interactive recommender systems. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 41–48, 2014.
- [99] Negar Hariri, Bamshad Mobasher, and Robin Burke. Adapting to user preference changes in interactive recommendation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

- [100] Reinhard Heckel and Kannan Ramchandran. The sample complexity of online one-class collaborative filtering. *Proceedings of the 34th International Conference on Machine Learning JMLR.*, 2017.
- [101] Antonio Hernando, Jesús Bobadilla, Fernando Ortega, and Abraham Gutiérrez. A probabilistic model for recommending to new cold-start non-registered users. *Information Sciences*, 376:216–232, 2017.
- [102] Steven Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. 02 2018.
- [103] Chu-Cheng Hsieh, James Neufeld, Tracy King, and Junghoo Cho. Efficient approximate thompson sampling for search query recommendation. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 740–746, 2015.
- [104] Rong Hu and Pearl Pu. Using personality information in collaborative filtering for new users. *Recommender Systems and the Social Web*, 17, 2010.
- [105] Wen Huang, Kevin Labille, Xintao Wu, Dongwon Lee, and Neil Heffernan. Achieving user-side fairness in contextual bandits. 10 2020.
- [106] Nicole Immorlica, Jieming Mao, Aleksandrs Slivkins, and Zhiwei Steven Wu. Bayesian exploration with heterogeneous agents. In *The World Wide Web Conference*, pages 751–761, 2019.
- [107] Wacharawan Intayoad, Chayapol Kamyod, and Punnarumol Temdee. Reinforcement learning based on contextual bandits for personalized online learning recommendation systems. *Wireless Personal Communications*, 115(4):2917–2932, 2020.
- [108] Rolf Jagerman, Ilya Markov, and Maarten de Rijke. When people change their mind: Off-policy evaluation in non-stationary recommendation environments. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 447–455, 2019.
- [109] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender systems: an introduction*. Cambridge University Press, 2010.
- [110] Umair Javed, Kamran Shaukat, Ibrahim A Hameed, Farhat Iqbal, Talha Mahboob Alam, and Suhuai Luo. A review of content-based and context-based recommendation systems. *International Journal of Emerging Technologies in Learning (iJET)*, 16(3):274–306, 2021.
- [111] Matthieu Jedor, Vianney Perchet, and Jonathan Louedec. Categorized bandits. In *Advances in Neural Information Processing Systems*, pages 14422–14432, 2019.

- [112] Antti Kangasrääsio, Dorota Glowacka, and Samuel Kaski. Improving controllability and predictability of interactive recommendation interfaces for exploratory search. In *Proceedings of the 20th international conference on intelligent user interfaces*, pages 247–251, 2015.
- [113] Sumeet Katariya, Branislav Kveton, Zheng Wen, and Vamsi K Potluru. Conservative exploration using interleaving. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 954–963, 2019. URL <http://proceedings.mlr.press/v89/katariya19a.html>.
- [114] Jaya Kawale, Hung H Bui, Branislav Kveton, Long T Thanh, and Sanjay Chawla. Efficient thompson sampling for online matrix-factorization recommendation. *Advances in Neural Information Processing Systems*, 28:1297–1305, 2015.
- [115] Sami Khenissi, Boujelbene Mariem, and Olfa Nasraoui. Theoretical modeling of the iterative properties of user discovery in a collaborative filtering recommender system. In *Fourteenth ACM Conference on Recommender Systems*, pages 348–357, 2020.
- [116] Heung-Nam Kim, Abdulmajeed Alkhaldi, Abdulmotaleb El Saddik, and Geun-Sik Jo. Collaborative user modeling with user-generated tags for social recommender systems. *Expert Systems with Applications*, 38(7):8488–8496, 2011.
- [117] Weihao Kong, Emma Brunskill, and Gregory Valiant. Sublinear optimal policy value estimation in contextual bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 4377–4387. PMLR, 2020.
- [118] Yehuda Koren and Robert Bell. Advances in collaborative filtering. *Recommender systems handbook*, pages 77–118, 2015.
- [119] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.
- [120] Pigi Kouki, James Schaffer, Jay Pujara, John O’Donovan, and Lise Getoor. User preferences for hybrid explanations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 84–88, 2017.
- [121] Artus Krohn-Grimberghe, Alexandros Nanopoulos, and Lars Schmidt-Thieme. A novel multidimensional framework for evaluating recommender systems. In *LWA*, pages 113–120, 2010.
- [122] Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems. *arXiv preprint arXiv:1402.6028*, 2014.

- [123] Saurabh Kulkarni and Sunil F Rodd. Context aware recommendation systems: A review of the state of the art techniques. *Computer Science Review*, 37:100255, 2020.
- [124] Benjamin A Kwapong, Richard Anarfi, and Kenneth K Fletcher. Personalized service recommendation based on user dynamic preferences. In *Services Computing–SCC 2019: 16th International Conference, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 16*, pages 77–91. Springer, 2019.
- [125] Anisio Lacerda. Multi-objective ranked bandits for recommender systems. *Neurocomputing*, 246:12–24, 2017. URL <https://www.sciencedirect.com/science/article/pii/S092523121730228X>.
- [126] Anisio Lacerda, Adriano Veloso, and Nivio Ziviani. Adding value to daily-deals recommendation: Multi-armed bandits to match customers and deals. In *2015 Brazilian Conference on Intelligent Systems (BRACIS)*, pages 216–221. IEEE, 2015.
- [127] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. Melu: meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM SIGKDD*, pages 1073–1082, 2019.
- [128] Husheng Li. A recommendation system in cognitive radio networks with random data traffic. *IEEE Transactions on Vehicular Technology*, 60(4):1352–1364, 2011.
- [129] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.
- [130] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306, 2011.
- [131] Mengxian Li, Wenjun Jiang, and Kenli Li. When and what music will you listen to? fine-grained time-aware music recommendation. In *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/I-UCC)*, pages 1091–1098. IEEE, 2017.
- [132] Qing Li and Byeong Man Kim. Clustering approach for hybrid recommender system. In *Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on*, pages 33–38. IEEE, 2003.

- [133] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548, 2016.
- [134] Shuai Li, Baoxiang Wang, Shengyu Zhang, and Wei Chen. Contextual combinatorial cascading bandits. In *ICML*, volume 16, pages 1245–1253, 2016. URL <http://www.jmlr.org/proceedings/papers/v48/lif16-suppl.pdf>.
- [135] Yu Liang, Babak Loni, and Martha A Larson. Clef newsreel 2017: Contextual bandit news recommendation. In *CLEF (Working Notes)*, 2017. URL <https://pure.tudelft.nl/portal/files/35743499/35743393.pdf>.
- [136] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert systems with applications*, 41(4): 2065–2073, 2014.
- [137] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.
- [138] Bo Liu, Ying Wei, Yu Zhang, Zhixian Yan, and Qiang Yang. Transferable contextual bandit for cross-domain recommendation. In *AAAI*, 2018. URL <https://openreview.net/forum?id=r1-g8CxdWB¬eId=r1-g8CxdWB>.
- [139] Chang Liu, Qingpeng Cai, and Yukui Zhang. Multi-armed bandit mechanism with private histories. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 1607–1609, 2017.
- [140] Weiwen Liu, Shuai Li, and Shengyu Zhang. Contextual dependent click bandit algorithm for web recommendation. In *International Computing and Combinatorics Conference*, pages 39–50. Springer, 2018. URL https://link.springer.com/chapter/10.1007/978-3-319-94776-1_4.
- [141] Xin Liu. Modeling users’ dynamic preference for personalized recommendation. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.
- [142] Yaxu Liu, Jui-Nan Yen, Bowen Yuan, Rundong Shi, Peng Yan, and Chih-Jen Lin. Practical counterfactual policy learning for top-k recommendations. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1141–1151, 2022.
- [143] Jonathan Lou  dec, Max Chevalier, Josiane Mothe, Aur  lien Garivier, and S  bastien Gerchinovitz. A multiple-play bandit algorithm applied to recommender systems. In *FLAIRS Conference*, pages 67–72, 2015.

- [144] Xiuyuan Lu, Zheng Wen, and Branislav Kveton. Efficient online recommendation via low-rank ensemble sampling. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 460–464, 2018. URL <https://dl.acm.org/doi/abs/10.1145/3240323.3240408>.
- [145] Dhruv Kumar Mahajan, Rajeev Rastogi, Charu Tiwari, and Adway Mitra. Logucb: an explore-exploit algorithm for comments recommendation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 6–15, 2012. URL <https://dl.acm.org/doi/abs/10.1145/2396761.2396767>.
- [146] Mohammad Malekzadeh, Dimitrios Athanasakis, Hamed Haddadi, and Benjamin Livshits. Privacy-preserving bandits. 2019.
- [147] Indu Manickam, Andrew S Lan, and Richard G Baraniuk. Contextual multi-armed bandit algorithms for personalized learning action selection. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6344–6348. IEEE, 2017.
- [148] Silviu Maniu, Stratis Ioannidis, and Bogdan Cautis. Bandits under the influence (extended version). *arXiv preprint arXiv:2009.10135*, 2020.
- [149] Miguel Martín, Antonio Jiménez-Martín, and Alfonso Mateos. A numerical analysis of allocation strategies for the multi-armed bandit problem under delayed rewards conditions in digital campaign management. *Neurocomputing*, 363:99–113, 2019.
- [150] Robillard Martin, Maalej Walid, Walker Robert, and Zimmermann Thomas. *Recommendation Systems in Software Engineering*. Springer, 2014.
- [151] Judith Masthoff. Group recommender systems: Combining individual models. In *Recommender systems handbook*, pages 677–702. Springer, 2010.
- [152] Judith Masthoff. Group recommender systems: aggregation, satisfaction and group attributes. *recommender systems handbook*, pages 743–776, 2015.
- [153] Pyy Matikainen, P. Furlong, Rahul Sukthankar, and Martial Hebert. Multi-armed recommendation bandits for selecting state machine policies for robotic systems. pages 4545–4551, 05 2013. ISBN 978-1-4673-5641-1. doi: 10.1109/ICRA.2013.6631223.
- [154] Benedict C May, Nathan Korda, Anthony Lee, and David S Leslie. Optimistic bayesian sampling in contextual-bandit problems. *The Journal of Machine Learning Research*, 13(1):2069–2106, 2012. URL <https://dl.acm.org/doi/abs/10.5555/2503308.2343711>.

- [155] James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM RecSys*, pages 31–39, 2018.
- [156] Rishabh Mehrotra, Niannan Xue, and Mounia Lalmas. Bandit based optimization of multiple objectives on a music streaming platform. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3224–3233, 2020.
- [157] Prem Melville, Raymond J Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Aaai/iaai*, pages 187–192, 2002.
- [158] Nikita Mishra and Abhradeep Thakurta. Private stochastic multi-arm bandits: From theory to practice. In *ICML Workshop on Learning, Security, and Privacy*, 2014.
- [159] Raymond J Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM, 2000.
- [160] Subhojyoti Mukherjee, Branislav Kveton, and Anup B Rao. Latent ranked bandits. *ICML 2019 Workshop RL4RealLife*, 2019.
- [161] Hai Thanh Nguyen and Anders Kofod-Petersen. Using multi-armed bandit to solve cold-start problems in recommender systems at telco. In *Mining Intelligence and Knowledge Exploration*, pages 21–30. Springer, 2014.
- [162] Minh NH Nguyen, Chuan Pham, Jaehyeok Son, and Choong Seon Hong. Online learning-based clustering approach for news recommendation systems. In *2016 18TH Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 1–4. IEEE, 2016.
- [163] Trong T Nguyen and Hady W Lauw. Dynamic clustering of contextual multi-armed bandits. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1959–1962, 2014.
- [164] Jorge Nocedal and Stephen J Wright. Sequential quadratic programming. *Numerical optimization*, pages 529–562, 2006.
- [165] Weishen Pan, Sen Cui, Hongyi Wen, Kun Chen, Changshui Zhang, and Fei Wang. Correcting the user feedback-loop bias for recommendation systems. *arXiv preprint arXiv:2109.06037*, 2021.

- [166] Luca Pantea. Adapting to dynamic user preferences in recommendation systems via deep reinforcement learning. 2022.
- [167] Ilona Pawełszek. Customer segmentation based on activity monitoring applications for the recommendation system. *Procedia Computer Science*, 192:4751–4761, 2021.
- [168] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. *The adaptive web: methods and strategies of web personalization*, pages 325–341, 2007.
- [169] Yi Peng, Miao Xie, Jiahao Liu, Xuying Meng, Nan Li, Cheng Yang, Tao Yao, and Rong Jin. A practical semi-parametric contextual bandit. In *IJCAI*, pages 3246–3252, 2019. URL <https://www.ijcai.org/Proceedings/2019/0450.pdf>.
- [170] Andre Luiz Vizine Pereira and Eduardo Raul Hruschka. Simultaneous co-clustering and learning to address the cold start problem in recommender systems. *Knowledge-Based Systems*, 82:11–19, 2015.
- [171] Alexandrin Popescul, David M Pennock, and Steve Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 437–444. Morgan Kaufmann Publishers Inc., 2001.
- [172] Runtao Qiao, Shuhan Yan, and Beijun Shen. A reinforcement learning solution to cold-start problem in software crowdsourcing recommendations. In *IEEE International Conference on Progress in Informatics and Computing*, pages 8–14, 2018.
- [173] Lijing Qin, Shouyuan Chen, and Xiaoyan Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 461–469. SIAM, 2014. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611973440.53>.
- [174] Dimitrios Rafailidis and Alexandros Nanopoulos. Modeling users preference dynamics and side information in recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(6):782–792, 2015.
- [175] Dimitrios Rafailidis, Pavlos Kefalas, and Yannis Manolopoulos. Preference dynamics with multimodal user-item interactions in social media recommendation. *Expert Systems with Applications*, 74:11–18, 2017.
- [176] Mahmuda Rahman and Jae C Oh. Fast online learning to recommend a diverse set from big data. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 361–370. Springer, 2015.

- [177] Mahmuda Rahman and Jae C Oh. Parallel and synchronized ucb2 for online recommendation systems. In *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, volume 1, pages 413–416. IEEE, 2015.
- [178] Mahmuda Rahman and Jae C Oh. Graph bandit for diverse user coverage in online recommendation. *Applied Intelligence*, 48(8):1979–1995, 2018.
- [179] Eduardo Ramos. *E-commerce*. Editora FGV, 2015.
- [180] Dattaraj Rao. Contextual bandits for adapting to changing user preferences over time. *arXiv preprint arXiv:2009.10073*, 2020.
- [181] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K Lam, Sean M McNee, Joseph A Konstan, and John Riedl. Getting to know you: learning new user preferences in recommender systems. In *Proceedings of the 7th international conference on Intelligent user interfaces*, pages 127–134, 2002.
- [182] Aditya Narayan Ravi, Pranav Poduval, and Sharayu Moharir. Unreliable multi-armed bandits: A novel approach to recommendation systems. In *2020 International Conference on COMMunication Systems & NETWORKS (COMSNETS)*, pages 650–653. IEEE, 2020.
- [183] Wenbo Ren, Xingyu Zhou, Jia Liu, and Ness B Shroff. Multi-armed bandits with local differential privacy. *arXiv preprint arXiv:2007.03121*, 2020.
- [184] Xingyi Ren, Meina Song, E Haihong, and Junde Song. Context-aware probabilistic matrix factorization modeling for point-of-interest recommendation. *Neurocomputing*, 241:38–55, 2017.
- [185] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. Neural collaborative filtering vs. matrix factorization revisited. In *Fourteenth ACM Conference on Recommender Systems*, pages 240–248, 2020.
- [186] Francesco Ricci, Lior Rokach, and Bracha Shapira. *Introduction to recommender systems handbook*. Springer, 2011.
- [187] Neil Rubens and Masashi Sugiyama. Influence-based collaborative active learning. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 145–148, 2007.
- [188] Neil Rubens, Mehdi Elahi, Masashi Sugiyama, and Dain Kaplan. Active learning in recommender systems. In *Recommender systems handbook*, pages 809–846. Springer, 2015.

- [189] Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation. *arXiv preprint arXiv:2008.07146*, 2020.
- [190] Aghiles Salah, Quoc-Tuan Truong, and Hady W Lauw. Cornac: A comparative framework for multimodal recommender systems. *J. Mach. Learn. Res.*, 21:95–1, 2020.
- [191] Marlesson RO Santana, Luckeciano C Melo, Fernando HF Camargo, Bruno Brandão, Anderson Soares, Renan M Oliveira, and Sandor Caetano. Contextual meta-bandit for recommender systems selection. In *Fourteenth ACM Conference on Recommender Systems*, pages 444–449, 2020.
- [192] Javier Sanz-Cruzado, Pablo Castells, and Esther López. A simple multi-armed nearest-neighbor bandit for interactive recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 358–362, 2019.
- [193] A Ömer Saritaç and Cem Tekin. Combinatorial multi-armed bandit problem with probabilistically triggered arms: A case with bounded regret. In *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 111–115. IEEE, 2017.
- [194] Masahiro Sato, Koki Nagatani, and Takuji Tahara. Exploring an optimal online model for new job recommendation: Solution for recsys challenge 2017. In *Proceedings of the Recommender Systems Challenge 2017*, pages 1–5. 2017.
- [195] Martin Saveski and Amin Mantrach. Item cold-start recommendations: learning local collective embeddings. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 89–96, 2014.
- [196] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.
- [197] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260, 2002.
- [198] Gunnar Schröder, Maik Thiele, and Wolfgang Lehner. Setting goals and choosing metrics for recommender system evaluations. In *UCERSTI2 workshop at the 5th ACM conference on recommender systems, Chicago, USA*, volume 23, page 53, 2011.
- [199] Burr Settles. Active learning literature survey. 2009.

- [200] Sulthana Shams, Daron Anderson, and Douglas Leith. Cluster-based bandits: Fast cold-start for recommender system new users. 2021.
- [201] Guy Shani and Asela Gunawardana. Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer, 2011.
- [202] Bracha Shapira, Francesco Ricci, Paul B Kantor, and Lior Rokach. Recommender systems handbook., 2011.
- [203] Nícollas Silva, Diego Carvalho, Adriano CM Pereira, Fernando Mourão, and Leonardo Rocha. The pure cold-start problem: A deep study about how to conquer first-time users in recommendations domains. *Information Systems*, 80:1–12, 2019.
- [204] Nícollas Silva, Heitor Werneck, Thiago Silva, Adriano CM Pereira, and Leonardo Rocha. A contextual approach to improve the user’s experience in interactive recommendation systems. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, pages 89–96, 2021.
- [205] Nícollas Silva, Heitor Werneck, Thiago Silva, Adriano CM Pereira, and Leonardo Rocha. Multi-armed bandits in recommendation systems: A survey of the state-of-the-art and future directions. *Expert Systems with Applications*, 197:116669, 2022.
- [206] Nícollas Silva, Thiago Silva, Henrique Hott, Yan Ribeiro, Adriano Pereira, and Leonardo Rocha. Exploring scenarios of uncertainty about the users’ preferences in interactive recommendation systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023.
- [207] Nicollas Silva, Thiago Silva, Heitor Werneck, Leonardo Rocha, and Adriano Pereira. User cold-start problem in multi-armed bandits: when the first recommendations guide the user’s experience. *ACM Transactions on Recommender Systems*, 1(1): 1–24, 2023.
- [208] Thiago Silva, Nícollas Silva, Heitor Werneck, Adriano CM Pereira, and Leonardo Rocha. The impact of first recommendations based on exploration or exploitation approaches in recommender systems’ learning. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, pages 173–180, 2020.
- [209] Thiago Silva, Nícollas Silva, Heitor Werneck, Carlos Mito, Adriano CM Pereira, and Leonardo Rocha. irec: An interactive recommendation framework. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3165–3175, 2022.
- [210] Thiago Silveira, Min Zhang, Xiao Lin, Yiqun Liu, and Shaoping Ma. How good your recommender system is? a survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics*, 10(5):813–831, 2019.

- [211] Linqi Song, Cem Tekin, and Mihaela Van Der Schaar. Clustering based online learning in recommender systems: A bandit approach. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4528–4532. IEEE, 2014.
- [212] Linqi Song, Cem Tekin, and Mihaela Van Der Schaar. Online learning in large-scale contextual recommender systems. *IEEE Transactions on Services Computing*, 9(3):433–445, 2014. URL <https://ieeexplore.ieee.org/abstract/document/6940318/>.
- [213] Linqi Song, Christina Fragouli, and Devavrat Shah. Recommender systems over wireless: Challenges and opportunities. In *2018 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE, 2018.
- [214] Linqi Song, Christina Fragouli, and Devavrat Shah. Interactions between learning and broadcasting in wireless recommendation systems. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 2549–2553. IEEE, 2019.
- [215] Emily Strong, Bernard Kleynhans, and Serdar Kadioglu. Mabwiser: A parallelizable contextual multi-armed bandit library for python. In *31st IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2019, Portland, OR, USA, November 4-6, 2019*, pages 909–914. IEEE, 2019. doi: 10.1109/ICTAI.2019.00129. URL <https://doi.org/10.1109/ICTAI.2019.00129>.
- [216] Emily Strong, Bernard Kleynhans, and Serdar Kadioglu. MABWiser: parallelizable contextual multi-armed bandits. *Int. J. Artif. Intell. Tools*, 30(4), 2021. doi: 10.1142/S0218213021500214.
- [217] Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with gaussian processes. In *International Conference on Machine Learning*, pages 997–1005. PMLR, 2015. URL <http://proceedings.mlr.press/v37/sui15.html>.
- [218] Dongting Sun, Zhigang Luo, and Fuhai Zhang. A novel approach for collaborative filtering to alleviate the new item cold-start problem. In *2011 11th International Symposium on Communications & Information Technologies (ISCIT)*, pages 402–406. IEEE, 2011.
- [219] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [220] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Damien Jose, and Imed Zitouni. Off-policy evaluation for slate recommendation. 05 2016.

- [221] Sho Takemori, Masahiro Sato, Takashi Sonoda, Janmajay Singh, and Tomoko Ohkuma. Submodular bandit problem under multiple constraints. *arXiv preprint arXiv:2006.00661*, 2020.
- [222] Liang Tang, Yexi Jiang, Lei Li, and Tao Li. Ensemble contextual bandits for personalized recommendation. In *Proceedings of the 8th ACM RecSys*, pages 73–80, 2014.
- [223] Liang Tang, Yexi Jiang, Lei Li, Chunqiu Zeng, and Tao Li. Personalized recommendation via parameter-free contextual bandits. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 323–332, 2015.
- [224] Kubra Tas, Eyup Onder, and Mehmet S Aktas. On the implicit feedback based data modeling approaches for recommendation systems. In *2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pages 1–6. IEEE, 2021.
- [225] Maryam Tavakol and Ulf Brefeld. A unified contextual bandit framework for long- and short-term recommendations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 269–284. Springer, 2017.
- [226] Cem Tekin and Eralp Turğay. Multi-objective contextual multi-armed bandit with a dominant objective. *IEEE Transactions on Signal Processing*, 66(14):3799–3813, 2018.
- [227] Cem Tekin and Mihaela Van Der Schaar. Releaf: An algorithm for learning and exploiting relevance. *IEEE Journal of Selected Topics in Signal Processing*, 9(4):716–727, 2015.
- [228] Cem Tekin, Simpson Zhang, and Mihaela van der Schaar. Distributed online learning in social recommender systems. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):638–652, 2014.
- [229] Choon Hui Teo, Houssam Nassif, Daniel Hill, Sriram Srinivasan, Mitchell Goodman, Vijai Mohan, and SVN Vishwanathan. Adaptive, personalized diversity for visual discovery. In *Proceedings of the 10th ACM conference on recommender systems*, pages 35–38, 2016. URL <https://dl.acm.org/doi/abs/10.1145/2959100.2959171>.
- [230] Georgios Theocharous, Nikos Vlassis, and Zheng Wen. An interactive points of interest guidance system. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces Companion*, pages 49–52, 2017.

- [231] Stefano Tracà, Cynthia Rudin, and Weiyu Yan. Reducing exploration of dying arms in mortal bandits. In *Uncertainty in Artificial Intelligence*, pages 156–163. PMLR, 2020.
- [232] Abhishek Tripathi, TS Ashwin, and Ram Mohana Reddy Guddeti. A reinforcement learning and recurrent neural network based dynamic user modeling system. In *2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT)*, pages 411–415. IEEE, 2018.
- [233] Robin Van Meteren and Maarten Van Someren. Using content-based filtering for recommendation. In *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*, pages 47–56, 2000.
- [234] Saúl Vargas and Pablo Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 109–116, 2011.
- [235] Saúl Vargas. New approaches to diversity and novelty in recommender systems. In *Fourth BCS-IRSG Symposium on Future Directions in Information Access (FDIA 2011) 4*, pages 8–13, 2011.
- [236] Koen Verstrepen and Bart Goethals. Top-n recommendation for shared accounts. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 59–66, 2015.
- [237] Aleksandr Vorobev, Damien Lefortier, Gleb Gusev, and Pavel Serdyukov. Gathering additional feedback on search results by multi-armed bandits with respect to production ranking. pages 1177–1187, 05 2015. doi: 10.1145/2736277.2741104.
- [238] Huazheng Wang, Qingyun Wu, and Hongning Wang. Learning hidden features for contextual bandits. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1633–1642, 2016.
- [239] Huazheng Wang, Qingyun Wu, and Hongning Wang. Factorization bandits for interactive recommendation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [240] Huazheng Wang, Qian Zhao, Qingyun Wu, Shubham Chopra, Abhinav Khaitan, and Hongning Wang. Global and local differential privacy for collaborative bandits. In *Fourteenth ACM Conference on Recommender Systems*, pages 150–159, 2020.
- [241] Lu Wang, Chengyu Wang, Keqiang Wang, and Xiaofeng He. Biucb: A contextual bandit algorithm for cold-start and diversified recommendation. In *2017 IEEE International Conference on Big Knowledge (ICBK)*, pages 248–253. IEEE, 2017.

- [242] Pengyang Wang, Kunpeng Liu, Lu Jiang, Xiaolin Li, and Yanjie Fu. Incremental mobile user profiling: Reinforcement learning with spatial knowledge graph for modeling event streams. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 853–861, 2020.
- [243] Qing Wang, Tao Li, SS Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. Online it ticket automation recommendation using hierarchical multi-armed bandit algorithms. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 657–665. SIAM, 2018.
- [244] Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S Sitharama Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *IEEE Transactions on Knowledge and Data Engineering*, 31(8):1569–1580, 2018.
- [245] Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S Sitharama Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *IEEE Transactions on Knowledge and Data Engineering*, 31(8):1569–1580, 2018.
- [246] Ruiqin Wang, Jungang Lou, and Yunliang Jiang. Session-based recommendation with time-aware neural attention network. *Expert Systems with Applications*, 210: 118395, 2022.
- [247] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. Tem: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pages 1543–1552, 2018.
- [248] Xin Wang, Steven CH Hoi, Chenghao Liu, and Martin Ester. Interactive social recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 357–366, 2017.
- [249] Xinxi Wang, Yi Wang, David Hsu, and Ye Wang. Exploration in interactive personalized music recommendation: a reinforcement learning approach. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11(1):1–22, 2014.
- [250] Yingfei Wang, Hua Ouyang, Chu Wang, Jianhui Chen, Tsvetan Asamov, and Yi Chang. Efficient ordered combinatorial semi-bandits for whole-page recommendation. In *AAAI*, pages 2746–2753, 2017. URL http://www.yichang-cs.com/yahoo/AAAI17_SemiBandits.pdf.

- [251] Romain Warlop, Alessandro Lazaric, and Jérémie Mary. Fighting boredom in recommender systems with linear reinforcement learning. *Advances in Neural Information Processing Systems*, 31:1757–1768, 2018.
- [252] Chunting Wei, Jiwei Qin, and Qiulin Ren. A ranking recommendation algorithm based on dynamic user preference. *Sensors*, 22(22):8683, 2022.
- [253] Yiping Wen, Feiran Wang, Rui Wu, Jianxun Liu, and Buqing Cao. Improving the novelty of retail commodity recommendations using multiarmed bandit and gradient boosting decision tree. *Concurrency and Computation: Practice and Experience*, page e5703, 2020.
- [254] Chirayu Wongchokprasitti, Jaakko Peltonen, Tuukka Ruotsalo, Payel Bandyopadhyay, Giulio Jacucci, and Peter Brusilovsky. User model in a box: Cross-system user model transfer for resolving cold start problems. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 289–301. Springer, 2015. doi: https://doi.org/10.1007/978-3-319-20267-9_24.
- [255] Robert F Woolson. Wilcoxon signed-rank test. *Wiley encyclopedia of clinical trials*, pages 1–3, 2007.
- [256] Meng Wu, Ying Zhu, Qilian Yu, Bhargav Rajendra, Yunqi Zhao, Navid Aghdaie, and Kazi A Zaman. A recommender system for heterogeneous and time sensitive environment. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 210–218, 2019.
- [257] Qingyun Wu, Huazheng Wang, Quanquan Gu, and Hongning Wang. Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR Conference on Development in Information Retrieval*, 2016.
- [258] Qingyun Wu, Hongning Wang, Liangjie Hong, and Yue Shi. Returning is believing: Optimizing long-term user engagement in recommender systems. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1927–1936, 2017. URL <https://dl.acm.org/doi/abs/10.1145/3132847.3133025>.
- [259] Qingyun Wu, Naveen Iyer, and Hongning Wang. Learning contextual bandits in a non-stationary environment. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 495–504, 2018.
- [260] Qingyun Wu, Huazheng Wang, Yanen Li, and Hongning Wang. Dynamic ensemble of contextual bandits to satisfy users’ changing interests. pages 2080–2090, 05 2019. ISBN 978-1-4503-6674-8. doi: 10.1145/3308558.3313727.

- [261] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In *The World Wide Web Conference*, pages 2091–2102, 2019. URL <https://dl.acm.org/doi/abs/10.1145/3308558.3313442>.
- [262] Xian Wu, Suleyman Cetintas, Deguang Kong, Miao Lu, Jian Yang, and Nitesh Chawla. Learning from cross-modal behavior dynamics with graph-regularized neural contextual bandit. In *Proceedings of The Web Conference 2020*, pages 995–1005, 2020.
- [263] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the ninth ACM international conference on web search and data mining*, pages 153–162, 2016.
- [264] Yu Xin et al. *Challenges in recommender systems: scalability, privacy, and structured recommendations*. PhD thesis, Massachusetts Institute of Technology, 2015.
- [265] Xiao Xu, Sattar Vakili, Qing Zhao, and Ananthram Swami. Online learning with side information. In *MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM)*, pages 303–308. IEEE, 2017.
- [266] Xiao Xu, Fang Dong, Yanghua Li, Shaojian He, and Xin Li. Contextual-bandit based personalized recommendation with time-varying user interests. In *AAAI*, pages 6518–6525, 2020.
- [267] Usha Yadav, Neelam Duhan, and Komal Kumar Bhatia. Dealing with pure new user cold-start problem in recommendation system based on linked open data and social network features. *Mobile Information Systems*, 2020:1–20, 2020.
- [268] Yan Yan, Zitao Liu, Meng Zhao, Wentao Guo, Weipeng P Yan, and Yongjun Bao. A practical deep online ranking system in e-commerce recommendation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 186–201. Springer, 2018.
- [269] Kaige Yang and Laura Toni. Graph-based recommendation system. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 798–802. IEEE, 2018.
- [270] Mengyue Yang, Qingyang Li, Zhiwei Qin, and Jieping Ye. Hierarchical adaptive contextual bandits for resource constraint based recommendation. In *Proceedings of The Web Conference 2020*, pages 292–302, 2020.

- [271] Baolin Yi, Xiaoxuan Shen, Hai Liu, Zhaoli Zhang, Wei Zhang, Sannyuya Liu, and Naixue Xiong. Deep matrix factorization with implicit feedback embedding for recommendation system. *IEEE Transactions on Industrial Informatics*, 15(8):4591–4601, 2019.
- [272] Baosheng Yu, Meng Fang, and Dacheng Tao. Linear submodular bandits with a knapsack constraint. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 1380–1386, 2016.
- [273] Tong Yu, Ole Mengshoel, Dominique Meroux, and Zhen Jiang. Machine learning with decision trees and multi-armed bandits: An interactive vehicle recommender system. Technical report, SAE Technical Paper, 2019. URL <https://www.sae.org/publications/technical-papers/content/2019-01-1079/>.
- [274] Tong Yu, Yilin Shen, and Hongxia Jin. A visual dialog augmented interactive recommender system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 157–165, 2019.
- [275] Tong Yu, Yilin Shen, and Hongxia Jin. Towards hands-free visual dialog interactive recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1137–1144, 2020.
- [276] Yisong Yue and Carlos Guestrin. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems*, pages 2483–2491, 2011.
- [277] Chunqiu Zeng, Qing Wang, Shekoofeh Mokhtari, and Tao Li. Online context-aware recommendation with time varying multi-armed bandit. In *Proceedings of the 22nd ACM SIGKDD*, pages 2025–2034, 2016.
- [278] ChengXiang Zhai, William W Cohen, and John Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *ACM SIGIR Forum*, volume 49, pages 2–9. ACM New York, NY, USA, 2015.
- [279] Haifei Zhang, Yiyang Sun, Jianlin Qiu, Changyong Zhu, Junsong Zhao, and Haoyu Wang. A music recommendation system based on reinforcement learning. *Design Engineering*, pages 331–342, 2020.
- [280] Xiaofang Zhang, Qian Zhou, Tieke He, and Bin Liang. Con-cname: A contextual multi-armed bandit algorithm for personalized recommendations. In *International Conference on Artificial Neural Networks*, pages 326–336. Springer, 2018.
- [281] Xiaoying Zhang, Hong Xie, Hang Li, and John CS Lui. Conversational contextual bandit: Algorithm and application. In *Proceedings of The Web Conference 2020*, pages 662–672, 2020.

- [282] Yongfeng Zhang and Xu Chen. Explainable recommendation: A survey and new perspectives. 01 2020. doi: 10.1561/9781680836592.
- [283] Chen Zhao, Kohei Watanabe, Bin Yang, and Yu Hirate. Fast converging multi-armed bandit optimization using probabilistic graphical model. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 115–127. Springer, 2018.
- [284] Chen Zhao, Bin Yang, and Yu Hirate. A reward optimization model for decision-making under budget constraint. *Journal of Information Processing*, 27:190–200, 2019. URL https://www.jstage.jst.go.jp/article/ipsjjip/27/0/27_190/_article/-char/ja/.
- [285] Tong Zhao and Irwin King. Locality-sensitive linear bandit model for online social recommendation. In *International Conference on Neural Information Processing*, pages 80–90. Springer, 2016.
- [286] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4653–4664, 2021.
- [287] Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin. Deep reinforcement learning for search, recommendation, and online advertising: a survey. *ACM SIGWEB Newsletter*, (Spring):1–15, 2019.
- [288] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. Interactive collaborative filtering. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1411–1420, 2013.
- [289] Shan Zhong, Wenhao Ying, Xuemei Chen, and Qiming Fu. An adaptive similarity-measuring-based cmab model for recommendation system. *IEEE Access*, 8:42550–42561, 2020.
- [290] Chunyi Zhou, Yuanyuan Jin, Xiaoling Wang, and Yingjie Zhang. Conversational music recommendation based on bandits. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, pages 41–48. IEEE, 2020.
- [291] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1059–1068, 2018.

-
- [292] Pan Zhou, Kehao Wang, Linke Guo, Shimin Gong, and Bolong Zheng. A privacy-preserving distributed contextual federated online learning framework with big data support in social recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 2019.
- [293] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 179–188, 2020.
- [294] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Transactions on Knowledge and Data Engineering*, 32(4):631–644, 2019.
- [295] Zhenyu Zhu, Liusheng Huang, and Hongli Xu. Self-accelerated thompson sampling with near-optimal regret upper bound. *Neurocomputing*, 2020.
- [296] Shi Zong, Hao Ni, Kenny Sung, Nan Rosemary Ke, Zheng Wen, and Branislav Kveton. Cascading bandits for large-scale recommendation problems. *32nd Conf. on Uncertainty in Artificial Intelligence (UAI), 2016*, 03 2016.
- [297] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangji Huang, and Dawei Yin. Neural interactive collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 749–758, 2020.

Appendix A

A Systematic Literature Review about Multi-Armed Bandits in Recommendation Systems

This appendix represents an extra part of this thesis that was developed to identify the main guidelines around our research topic. It was published in a journal format on Expert Systems With Applications [205]. Basically, this study aims to review, organise, and present all works around Multi-Armed Bandits applied in the recommendation field. In this sense, we perform a Systematic Literature Review (SLR) protocol and search for works published in the main conferences and journals from the first studies (2000) until the last years (2020). Our research is guided by three main questions in order to: (1) synthesize the knowledge available in the literature around this topic; (2) highlight the best practices to construct and evaluate a bandit interactive model; and (3) analyse the current efforts to handle the traditional recommendation challenges.

A.1 Systematic Literature Review Protocol

A systematic literature review (SLR) is a scientific methodology designed to answer some well-formulated research questions. It aims to identify and synthesise all of the scholarly research on a particular topic by applying a rigorous, unbiased, and reproducible protocol. In general, there is a standard protocol usually defined by several steps at a high level to not consider the influence of research question type on the review procedures. Here, we design a protocol inspired by Çano and Morisio [41] that manages the review process in three main phases. First, we perform the paper collection by defining the search strings and searching in the main digital sources. Then, in phase 2, we perform the selection of papers in two steps: a coarse selection by analysing the title, year, conference, and abstract; and a detailed selection of the publications, by reading the full paper. Finally, in phase 3, we extract the main information of the papers according to our

criteria. Figure A.1 presents an overview of each step performed by this SLR protocol, representing a clear set of steps that are further discussed in the next sections.

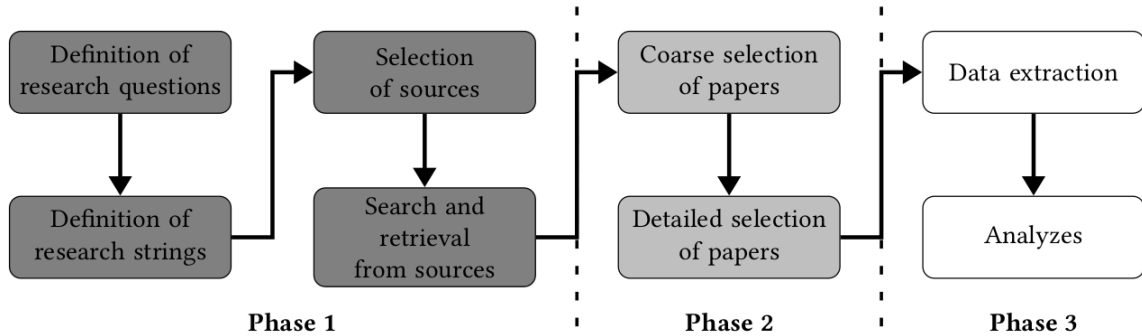


Figure A.1: Systematic Literature Review protocol.

A.1.1 Phase 1: Research questions, search strings and sources

First of all, this SLR defines the research questions to be applied to the main digital sources to identify and collect the most relevant papers in the literature. These questions are responsible to guide all the processes of our systematic literature review. Here, as this work is concerned about the application of Multi-Armed Bandits in the recommendation field, we design three main research questions to be answered by this SLR:

- **Q1:** *How have MAB algorithms been used in the RS field?*
- **Q2:** *How have MAB algorithms been empirically evaluated in the RS field?*
- **Q3:** *How have MAB algorithms dealt with classic challenges of the RSs literature, such as the cold-start problem?*

The first question is more inclusive and it may refer to most papers in the recommendation field related to MAB. Answering Q1, it is possible to consolidate a summary of the main research conducted in the last years. In turn, the second one refers to papers of MAB in the recommendation field that has performed an experimental evaluation. By analysing these papers we can identify how some MAB concepts are applied, what are the main adopted methods, what are their characteristics, and even how they are evaluated in this scenario. These analyses can also drive several future directions for the literature. Finally, the third question allows us to provide insights into how MAB has been applied to face some classical RSs' challenges. In Q3, we inspect papers related to bandit algorithms concerned with the main scenario of uncertainty studied in this work (the cold-start problem), and other relevant challenges from the literature.

In order to answer these questions, we propose to search for publications written in English and published in the last two decades, from 2000 until 2020 into the main

well-known repositories, such as IEEE Explorer, ACM Digital Library, Springer, Scopus, Scielo, and others. We advocate the use of the Google Scholar search engine because it automatically searches in these repositories. There, we use three search strings (SS) driven by the research questions as follows:

```
SS-Q1: ("multi-armed bandit" OR "multi-armed bandits") AND
("recommender system" OR "recommender systems" OR "recommendation")
SS-Q2: ("multi-armed bandit" OR "multi-armed bandits") AND
("recommender system" OR "recommender systems" OR "recommendation") AND
("experimental" OR "experiments" OR "evaluation" OR "datasets")
SS-Q3: ("multi-armed bandit" OR "multi-armed bandits") AND
("recommender system" OR "recommender systems" OR "recommendation") AND
("CF" OR "collaborative filtering") AND ("cold-start")
```

A.1.2 Phase 2: Selection of papers

Applying our search strings in the main sources, we identified 1327 articles by eliminating duplicates – those with the same title and publication link. While the first and second strings (SS-Q1 and SS-Q2) retrieved 930 and 936 papers respectively, the third string (SS-Q3) retrieved only 365 papers. To objectively decide whether to select each preliminary study for further processing or not, we defined a set of inclusion and exclusion criteria listed in Table A.1.

Inclusion criteria
<ul style="list-style-type: none"> - Papers presenting MAB algorithms in the context of recommendation systems - Papers that even though do not specifically propose a new MAB algorithm, but contrast those MAB with other methods to perform online recommendations - Papers from conferences and journals - Papers published from 2000 to 2020 - Papers written in English
Exclusion criteria
<ul style="list-style-type: none"> - Papers not addressing recommendation systems at all - Papers addressing RSs but not considering a MAB modelling - Papers that report only abstracts or presentation slides - Gray literature

Table A.1: Inclusion and exclusion criteria of our SLR.

In general, we only include journal and conference papers, leaving out grey literature, workshop presentations, abstract papers or slide presentations. Then, the selection of the most relevant studies is performed as follows. We initially analyse the title, publication year, and publication type (i.e., journal, conference, workshop, etc.), dropping out every paper that does not perform a recommendation study or does not address a bandit

model at all. After this step, we reach a list of 408 papers (30.75%). These papers are then deeply examined by analysing their introduction, experimental setup, and conclusion. From the 408 papers, 178 papers were rejected due to the same criteria listed, and 230 papers were selected as relevant studies for our SLR. These studies represent the most relevant works about MAB in the recommendation field and the remaining discussion of this chapter is focused on them.

A.1.3 Phase 3: Data extraction

In order to achieve our three main goals, we also propose to collect the main information for each paper. Then, we build a data extraction form to collect both paper metadata (i.e., author, title, year, etc.) and the relevant content data as described in Table A.2.

Data	Explanation	Examples
Title	-	-
Authors	-	-
Publication year	-	-
Conference	-	-
Source	Digital library where the paper is available	-
Domain	Domains studied in the paper	News, movies, songs, ads, etc.
Dataset	Public or private dataset used to train the algorithm	Yahoo, MovieLens, Netflix, etc.
Method	The bandit method applied	ϵ -Greedy, UCB, TS, etc.
RS Characteristics	The recommendation concept applied to guide the bandit algorithm	Graphs, matrix factorisation, probabilistic, etc.
Context	A context applied to improve the recommendation	User profile, items' characteristics, cookies, etc.
Data Type	The type of feedback provided by the user	Rating, clicks, likes, etc.
Data Processing	A processing stage performed to normalise the data	Mean-centering, Z-score, etc.
Metrics	The main metrics applied to measure the recommendation's quality	Precision, recall, CTR, etc.
Evaluation	The methodology applied to simulate the online user's interaction	Trials, interactions, etc.

Table A.2: Data extraction form.

While the metadata allows us to understand whereby is each work, the relevant content is used to answer our research questions and provide an overview of the main published works. It is defined based on the needed information to develop, apply, and analyse any MAB method in the field. The extraction process happens during the second step of reading when we examine the introduction, experimental setup, and conclusion of each paper. Here, we focused on 190 papers from the 230 works previously selected due to their experimental nature (i.e., those that performed experimental analyses).

A.2 MAB in Recommendation Systems

Nowadays, several works have modelled the online recommendation task as a Multi-Armed Bandit problem [241, 77, 244]. In most of the bandit representations, the items to be recommended are modelled as the arms to be pulled. Selecting an arm a is equivalent to recommending an item i and the reward is the user response to this recommendation (e.g., clicks, ratings, acceptance, etc.) [192]. Thus, the main goal is also to maximise the expected reward achieved after T times, as shown in Equation A.1. Here, however, the difference from traditional learning scenarios is the goal, which must be related to the user's satisfaction with the system. It requires a personalised action selection policy π to the users' preferences and tastes identified by the historic of user's actions h . For this reason, the item i_t^* should be chosen according to a prediction rule π , which is defined as a function to exploit and explore the current known information about the user until now: $i_t^* \equiv \pi(h_t)$. In the literature, there are a lot of distinct strategies to define and improve this action selection policy π .

$$i_{(\cdot)}^* = \arg \max_{i_{(\cdot)}} \sum_{t=1}^T \mathbb{E}[r_{u,i_t}|t] \quad (\text{A.1})$$

In this sense, the primary goal of our systematic literature review is to shed light on the current advances of MAB in the recommendation field. We intend to provide an overview of the most used concepts and methods, their characteristics, and also how they can be applied, evaluated, or improved. We guide our analysis in the 183 papers previously selected to answer the three research questions raised before. First, we highlight what the current works have proposed in the last few years by exploring some data extracted, such as the domain, dataset, method, RS characteristics, and context. Next, we analyse the evaluation criteria of MAB in the recommendation field by inspecting the experimental setup of the selected publications. Here, we investigate specific characteristics of the papers, such as the data type, the data processing approach, the evaluation metrics applied, and also the methodology chosen by the authors. And, finally, we discuss the application of MAB in traditional RSs' challenges by highlighting the number of works concerned with usual problems and their assumptions to handle each of them.

A.2.1 Works developed so far

The application of MAB in the recommendation field has received more attention recently. Despite the recommendation field had emerged in the mid of 90s, the first studies about this topic started in 2005 and it has become more relevant after 2010. In Figure A.2, it is possible to notice that more than 50% of all publications about this topic was only proposed in the last five years (2016 to 2020). The main explanation for this current

interest of the literature may be related to two main factors: the saturation of methods and concepts in the traditional scenario due to the great advances made in the last decade; and the increasing investment of several companies in RSs by requiring models capable to deal with the users’ needs without having to retrain the prediction model at each interaction.

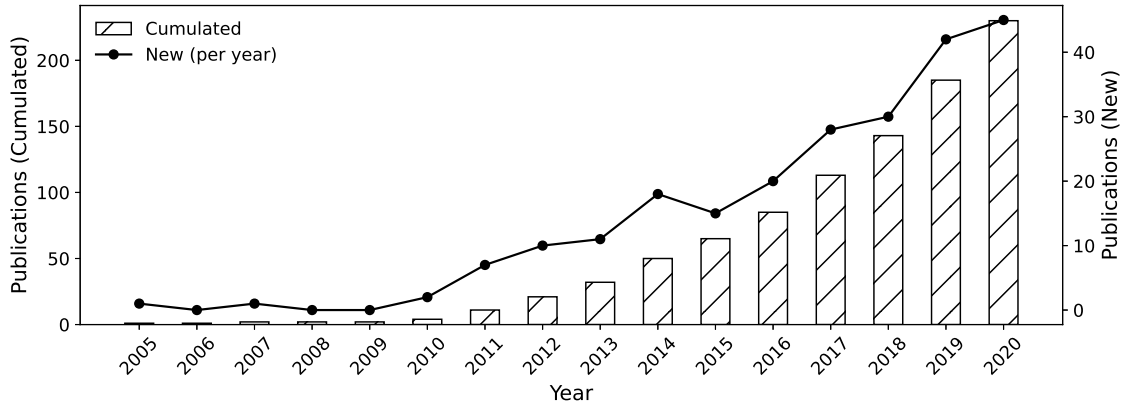


Figure A.2: Annual publications about MAB in the recommendation field.

Both interests of industry and academic have influenced several conferences to accept papers about the MAB application in the recommendation field and propose interesting talks between researchers and business leaders of big companies such as Netflix, Spotify, Google, Pandora, and others. Here, in our SLR, we identified 112 distinct conferences/journals that have accepted at least one paper about this topic. Table A.3 shows an overview of these conferences and journals by highlighting the top-20 that have accepted more papers. Therefore, the first insight we found is that MAB has been extensively discussed in the main conferences around the world.

#	Conference/Journal	Papers	#	Conference/Journal	Papers
1	RecSys	21	11	IJCAI	4
2	WWW	12	12	KDD	4
3	NIPS	11	13	TKDE	3
4	ICML	10	14	SAC	3
5	CIKM	9	15	University Lib	3
6	SIGIR	8	16	ICTAI	3
7	AAAI	7	17	UMAP	3
8	AISTATS	6	18	UAI	3
9	ICONIP	4	19	JMLR	3
10	WSDM	4	20	Neurocomputing	3

Table A.3: Conferences and journals that have accepted more papers about multi-armed bandits in the recommendation field.

Moreover, we identified that several approaches have been proposed for distinct domains by simulating a recommendation of movies to watch, news to read, products to buy, songs to hear, and others. Specifically, in the 190 papers with experimental evaluations

previously selected by our SLR, we identified 50 distinct domains highlighted according to the number of times each one is applied in a paper (i.e., their frequency) in Figure A.3.

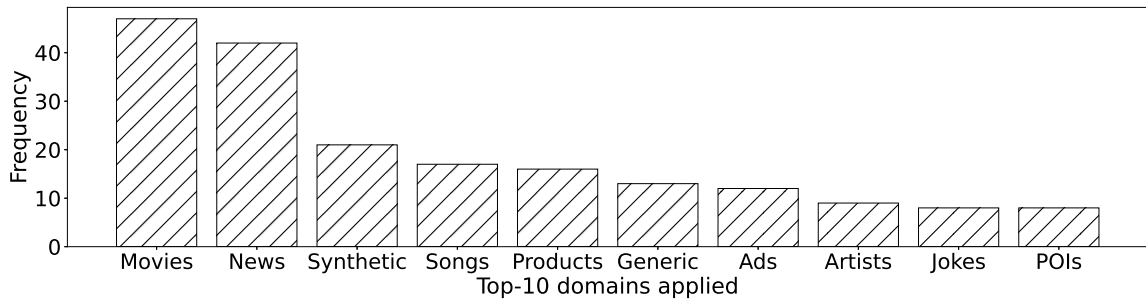


Figure A.3: The main domains used in MAB research in the recommendation field.

In general, the recommendation of Movies and News are the most usual by appearing in 24.7% and 22.1% of the papers [180, 208, 293, 281, 48, 231, 214, 138, 258]. The main works usually simulate these domains with traditional datasets available in the literature, such as the MovieLens library, the collection of news from the Yahoo front page, and the Amazon repository of users' purchases. Moreover, it is also a usual practice to evaluate MAB algorithms in synthetic or generic domains. While the synthetic domain is usually designed to simulate random (i.e., without bias) interactions of users with items, the generic domain refers to traditional datasets often applied to evaluate some Machine Learning algorithms (e.g., Iris Flower Species¹, CNAE-9², and others). In addition, we also noticed: (1) works proposed to other traditional recommendation domains, like Songs [257, 239, 155, 290, 108, 221, 262, 232], Artists [163, 98, 81, 43, 266, 285], Ads [82, 133, 222, 8, 223, 251, 149], Points-of-Interest (POIs) [57, 192, 296, 242, 230, 89, 213, 93], Products [85, 268, 38, 275, 253, 169], and Bookmarking [82, 81, 49, 262, 240]; (2) works that apply recommendation MAB algorithms to offer personalised rankings in the traditional Information Retrieval task [276, 237, 220]; and (3) works especially focused on non-usual scenarios like Jokes [177, 178, 176, 117, 265], Food [91, 106, 63, 92], and Jobs [194, 84].

However, despite the huge number of distinct domains currently used, the algorithms applied to handle the MAB problem are often related to the same approaches. Extracting the data related to 'Method' of each paper previously selected by our SLR, we calculate in our analysis the percentages of works related to each MAB approach in Figure A.4a. As we can see, the most used MAB algorithms are based on UCB, Thompson Sampling and ϵ -Greedy approaches, with 45.8%, 18.9%, and 16.8% of works, respectively. Other approaches, such as P2EE [182] and EXP3 [161, 128, 143] are unusual and less effectively. The works that are not concerned with a specific MAB algorithm, giving the option to apply any learning algorithm, are classified as independent from MAB ('i.i.d. MAB'). These works were not excluded by our SLR since we are also interested in all spectrums

¹<http://archive.ics.uci.edu/ml/datasets/Iris>

²<https://archive.ics.uci.edu/ml/datasets/CNAE-9>

related to applications of MAB. Our SLR also found works focused on designing other learning algorithms for specific scenarios that are classified here as ‘Others’. Besides, Figure A.4b highlights the application evolution of the three main MAB approaches over the last five years. Despite the greater number of publications based on the UCB approach, we can notice a growing interest among researchers in Thompson Sampling (TS) recently. The main explanation can be associated with the lack of theoretical analysis about TS before the publication of the paper named ‘An Empirical Evaluation of Thompson Sampling’ [51]. After this relevant work, more researchers could propose new MAB algorithms based on TS approaches, reflecting it in the number of publications in the next years.

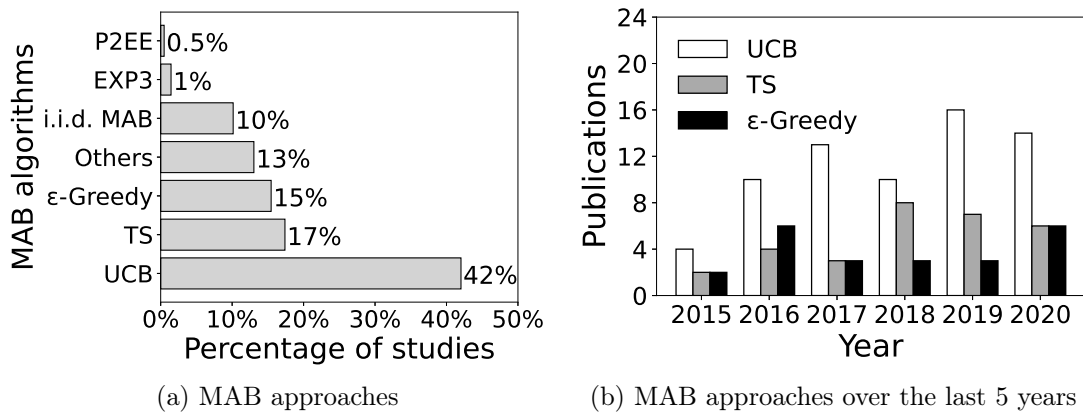


Figure A.4: MAB algorithms usually applied in the recommendation field.

Similarly, extracting the data related to ‘RS Characteristic’ of each paper previously selected by our SLR, our analyses also find the percentages of works where the RSs techniques are applied and also the number of techniques by year. A picture of this information is shown in Figure A.5. Here, these techniques are strongly relevant for the MAB algorithms because they must be combined to improve the recommendation’s quality [287]. The most usual RSs techniques aim to identify relevant items with distinct assumptions. Some techniques have assumed that users will like items that were liked by other users with the same preferences and tastes, designing algorithms based on Clusters [56, 82, 133, 81, 115, 270, 292, 61, 30, 111, 83, 14, 211], Graphs [31, 133, 49, 257, 283, 131, 29], and Matrix Factorisation [129, 239, 288, 229, 208, 156, 88, 148, 160, 39, 105, 225, 86, 227]. Other ones have considered some statistical fundamentals to model the user’s behaviour and, then, developed algorithms based on Bayesian [249, 145, 65, 3, 284, 83, 103] or Probabilistic concepts [277, 98, 250, 261, 222, 279, 23, 295, 274, 15, 273, 73, 280, 147, 35, 139, 193, 158]. Our SLR also identifies other technique, related to memory-based [31, 153, 37, 33, 289, 192, 100, 29, 227] and hybrid approaches [260, 77, 2, 191, 144, 135, 125, 126, 222, 92]. Moreover, it is important to observe that the three most applied techniques are based on Collaborative Filtering (CF) concepts, the most popular class of RSs in the literature [26]. Therefore, we also analyse the total of works that consider these techniques over the years in Fig-

ures A.5b. This figure shows the current growth of interest in Probabilistic and Matrix Factorisation. Indeed, these techniques have achieved the best results in the last few years.

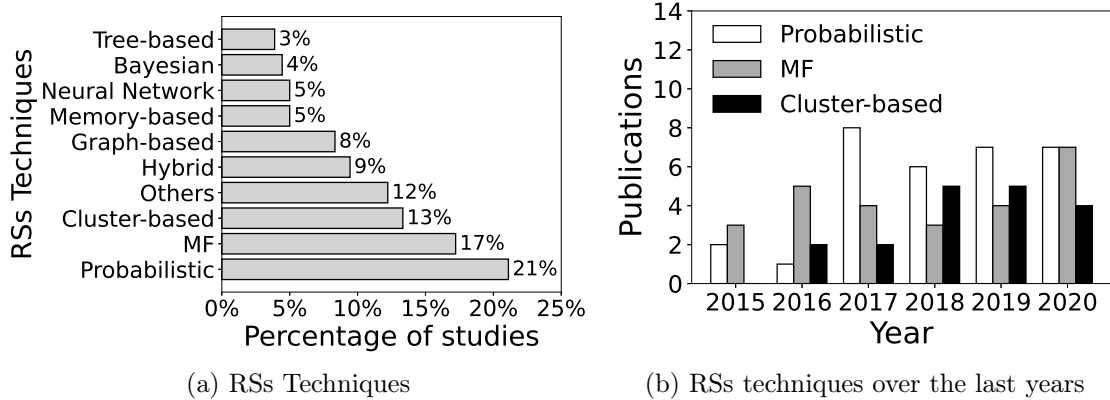


Figure A.5: RSs techniques usually combined with MAB algorithms.

Moreover, Figure A.6 highlights an updated picture of how MAB algorithms have been developed in the literature. This grid basically correlates the MAB approaches with the top-10 RSs techniques most applied in order to indicate how the action selection policy π has been implemented by the most relevant works in the field. The numbers show how many works were published with a MAB approach combined with an RSs strategy. In addition to the observations made before, we can also notice the influence of Collaborative Filtering concepts in MAB algorithms. In the top-10 RSs techniques applied, six of them are memory-based or model-based approaches (typical classes of CF algorithms). And, for the best of our knowledge, even the hybrid and heuristic approaches may usually apply a collaborative filtering representation combined with other information.

Then, studying this majority set of MAB algorithms (those CF-based), we identified two main groups of works with the same assumption of similarity between arms (i.e., playing one arm will give you information about similar arms). The first one has described the reward structure in terms of clusters of users and/or items and defined it similarly to memory-based methods (e.g., the k-Nearest Neighbours) [81, 133, 82]. In turn, the other one has assumed a similar structure between arms by constructing a model for each arm similar to CF model-based methods [288, 239, 276]. While some works have assumed a non-linear model applying a deep learning algorithm, other works have assumed a linear model and represented items/users probabilistically by vectors of features extracted from MF approaches [129, 288, 239, 244]. The linear bandits are most frequently applied and they are usually represented in two ways. Some works represent the reward by a Bernoulli [42] or Gaussian [98, 99, 243] distribution, applying (or not) a Bayesian Inference. They model an unknown probability distribution over the rewards ($\mathcal{R}^i(r) = \mathbb{P}[r|i]$) and try to identify the arm that will maximise the expected reward returned. In turn, other linear works usually represent each item i and user u by their features vector ex-

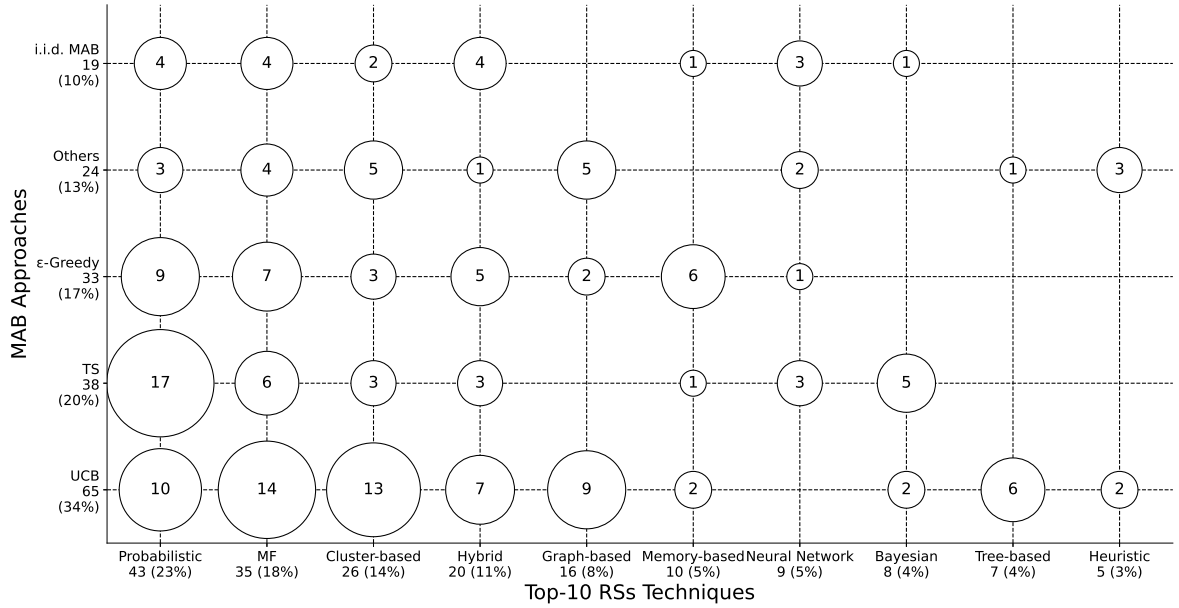


Figure A.6: An updated picture of the main RS techniques applied with MAB algorithms according to the works selected by our SLR.

tracted from an MF application called as \mathbf{q}_i and \mathbf{p}_u respectively. Here, the rating matrix $M^{m \times n}$ is estimated according to the product of two low-rank matrices $P \in \mathbb{R}^{m \times z}$ and $Q \in \mathbb{R}^{z \times n}$. While the matrix $P^{m \times z}$ contains the user-model p_u , representing the multiple interests of each user u in the z features, the matrix $Q^{z \times n}$ represents how relevant the item i is for the z groups. In this case, the expected reward is reformulated for the product of user and item features vectors:

$$i_{(\cdot)}^* = \arg \max_{i_{(\cdot)}} \sum_{t=1}^T \mathbb{E}[r_{u,i_t}|t] = \arg \max_{i_{(\cdot)}} \sum_{t=1}^T \mathbb{E}[\mathbf{p}_u^\top \mathbf{q}_{i_t}|t] \quad (\text{A.2})$$

Furthermore, current approaches usually introduce the user/item context available in the system to provide more effective recommendations [256, 226, 289]. Such approaches are named contextual Multi-Armed Bandits (CMAB) and they have been studied in around 43% of works selected by our SLR. Basically, in a typical contextual bandit setting, each arm a is associated with a d -dimensional context vector x_a , and its expected reward is guided by a conjecture of the context vector and an unknown bandit model, named θ^* . Here, in the online recommendation field, the unknown bandit model parameter θ^* is usually attached to each user to reflect their corresponding personalised preferences where θ_u^* for $u \in \mathcal{U}$ are independently estimated based on the observations from the corresponding users. Then, in a linear contextual bandit setting, it is assumed that $r_a \sim \mathcal{N}(x_a^\top \theta^*, \sigma^2)$ and the expected reward r_t achieved in a specific context x_a is defined as $\mathbb{E}[r_t|x_a] = x_a^\top \theta^*$. One of the most famous CMAB algorithms was designed by following these assumptions to maximise the total reward (i.e., the total user clicks). It is named LinUCB [129] and it measures the upper confidence bound for each item according

to the context x for choosing one of them at every trial t :

$$i_t^* = \arg \max_{i \in \mathcal{I}} (x_{t,i}^\top \theta^* + \alpha \sqrt{x_{t,i}^\top \mathbf{A}_i^{-1} x_{t,i}}) \quad (\text{A.3})$$

In general, the context x often summarises information of both the user u_t and the item i [1, 135]. However, our SLR also finds several definitions of context in the selected works. Some works simply model the d -dimensional vector x as a traditional embedding of TF-IDF values, measuring it by the items and users' tags [81, 259]. Currently, the works have modelled the context as users/items feature vectors extracted from an MF formulation [288, 98, 99, 248, 245, 3]. Other works, in turn, define the context with other information available in the dataset. Song et al. [212], Bouneffouf et al. [32], and Gutowski et al. [91], for example, use all information available about the users as context. They propose an embedding with the cookies of the users' last accesses to the system, their demographic information (e.g., gender, age, etc), and any other information available in their profile (e.g., preferences, social connection, etc.). Other works, like [133, 155, 28], apply explicit information about items to model the context. This type of data is very informative in specific domains, such as news [133, 129, 222], e-commerce [294, 261], and POIs [74, 296]. Other works are focused on extracting the context of users/items according to their cluster of interests [34, 269].

A.2.2 The Evaluation Criteria

In the literature, the evaluation criteria is a protocol defined to measure if a recommendation technique is (or is not) effective in a domain [68, 58]. Throughout the decades of recommendation systems research, it has been continually adapted and improved for several works [234, 46, 235] by performing two distinct criteria to measure the recommendation quality: (1) an offline experimentation; and (2) an online user's study. While offline experiments are often concerned about the prediction power – their ability to accurately predict the user's choices, the online assessments aim to measure the real business value – the actual value gain that a recommendation system can achieve to the system. However, due to performing AB tests with actual users, an online evaluation is more expensive than an offline and it usually requires a real system to collect the users' actions and opinions [121, 201]. Hence, it is a common practice to identify more offline experimentation than online user studies, especially when it is studying new scenarios and applications. Indeed, in our systematic review, studying the applicability of MAB in the recommendation field, we identified only 5 works (2.6%) that performed an online evaluation of their bandit algorithms. Once our study is really new, most of the works are still concerned with analysing their algorithms by offline experiments. By reviewing their evaluation criteria we identified three main steps to perform an offline evaluation:

Usual Datasets. In the MAB application in the recommendation field, we identified dis-

inct datasets that are selected or even created to measure the bandit performance. One usual practice is to create synthetic datasets to be applied specifically for some experimental analyses. However, it is not encouraged because these datasets are not reproducible and cannot provide confidence in the results achieved. The applicability of real-world datasets is undoubtedly the best practice for analysing the performance of any new recommendation technique. Fortunately, from the papers selected by our SLR, 186 works (80.9%) apply at least one real dataset in their experimentation. Our SLR highlights the datasets most applied to evaluate distinct bandit algorithms in Table A.4 and also provides how other researchers can find them.

Repository	Domain	Available at
MovieLens	Movies	https://grouplens.org/datasets/movielens/
Yahoo! News	News	https://webscope.sandbox.yahoo.com/
LastFM	Music	https://grouplens.org/datasets/hetrec-2011/
Delicious	Bookmarking	https://grouplens.org/datasets/hetrec-2011/
Netflix	Movies	https://kaggle.com/netflix-inc/netflix-prize-data
Jester	Jokes	https://goldberg.berkeley.edu/jester-data/
KDD - Online Ads	Advertising	https://www.kaggle.com/c/kddcup2012-track2/overview
Avazu	Advertising	https://kaggle.com/c/avazu-ctr-prediction
Amazon	Products	http://jmcauley.ucsd.edu/data/amazon/links.html
Yahoo! Music	Music	https://webscope.sandbox.yahoo.com/catalog.php?datatype=r
Million song	Music	http://millionsongdataset.com/
Yelp	Restaurant	https://yelp.com/dataset_challenge
Epinions	Products	http://alchemy.cs.washington.edu/data/epinions/

Table A.4: The main repositories with the datasets most applied by researchers to study MAB proposals in the recommendation field.

Data Processing. In most of these datasets previously mentioned, there is not only the information available about the users and items of the system but also the feedback that was given by a specific user to an item of the catalogue. This feedback may be explicit if the user explicitly says what they like and dislike, or implicit if it remains not clear if the user liked or not that item. For this reason, similar to traditional recommendation scenarios, 20% of works selected by our SLR have performed a data processing step to clean the data available by removing incomplete data, noises, and redundant records. These works usually perform a normalisation of their rating data by mapping each rating according to static scales or specific functions. Some works, like [40], simply defined that ratings less or equal to 3 mean the user did not like the item (mapping as 0) and, otherwise, ratings greater than 3 mean the user likes this item (mapping as 1). Other works, such as the one proposed in [20], define a specific function to normalise their ratings by converting each value to a $[0, 1]$ range. There is no consensus about the better practice behind the normalisation. However, it has become even more popular after the growth of UCB and Thompson Sampling applications due to their statistical assumptions of distributions such

as the Beta and Bayesian [284].

Methodologies & Metrics. As the bandit algorithm is an online learning method (see Fig. 2.6), most of the offline experiments must define an approach to simulate the user interaction with a real system. In the papers selected by our SLR, we identified five main approaches applied during the experimental evaluation:

- **Trials:** when a rating is estimated for one user-item pair in each interaction [284, 113, 52, 96, 134, 154].
- **Interactions:** when more than one item is recommended for the user [241, 140, 229, 217, 64, 237].
- **Replayer:** when the system tries to predict the items consumed chronologically according to the user’s consumption historic [277, 243, 244, 130, 222].
- **Leave-one-out:** when the system left one item out of the training step and tries to predict this specific item for a given user [77].
- **Cross-validation:** when the system performs the same user interaction many times by applying distinct data at the training [98, 249].

Based on our analyses, the methodology most applied is the *trials*, by covering more than 45.3% of the works selected. The second one is the *Interactions*, being applied in 8.9% of the works. *Replayer*, *Leave-one-out*, and *Cross-validation* are only applied in 2.6%, 0.5%, and 1.1% respectively. Possibly, the popularity of *trials* is related to a traditional bias that still exists in the evaluation criteria of several works. Similar to the first decade of researchers in the recommendation field, most bandit algorithms have been evaluated only according to their prediction power – their ability to accurately predict the user’s choices. Indeed, most evaluation metrics applied are to measure traditional prediction power. Especially, the researchers brought the classical metrics of reward and regret to the recommendation field and focused their bandit algorithms to maximise (minimise) it. In this case, the reward is interpreted as the number of hits achieved by such a method. On the other hand, regret is directly associated with the traditional error metrics used in the recommendation field, such as the MAE, RMSE, and others. Basically, these metrics represent the error in estimating a rating for a specific user-item pair.

However, it is now widely agreed that accurate predictions are crucial but insufficient to deploy a good recommendation engine [47, 46]. In many applications, people use a recommendation system for more than exact anticipation of their tastes. Users may also be interested in discovering new items, rapidly exploring diverse items, preserving their privacy, fast responses from the system, and many more properties of the interaction with the recommendation engine. Unfortunately, as shown in Figure A.7, metrics such as *diversity*, *novelty*, and *unexpectedness* have been applied only for less than 20% of the papers.

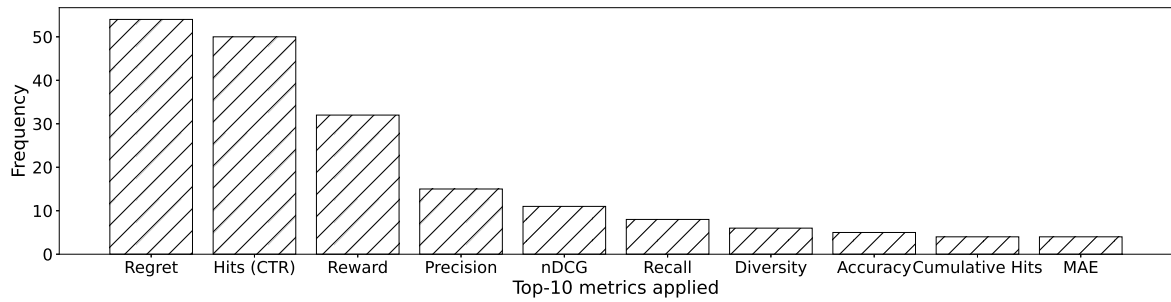


Figure A.7: Frequency of works which implemented each evaluation metric.

A.2.3 Facing RS challenges with MAB algorithms

Once the current works have combined MAB approaches with the main recommendation techniques, their solutions are susceptible to facing real challenges in the recommendation field. In this sense, our SLR also found many works concerned with addressing some challenges, such as data sparsity, cold-start, privacy, or explainability. In this section, we are concerned with explaining these challenges and understanding how MAB are handling them. Especially, we have a particular interest in the cold-start problem because it is the most similar challenge to the one described in this dissertation.

Data Sparsity & Scalability. Traditionally, while sparsity refers to the absence of data in a dataset, scalability refers to the difficulty of proposing an algorithm capable of presenting quick responses. In particular, in the recommendation field, both are strictly related because they are a consequence of the quick-growing of datasets according to more users and items added in real-world systems. With more items in a system, the difficulty for users to rate all of them also increases, worsening the data sparsity problem. Similarly, with more users in a system, it is more expensive to process all data and provide recommendations for all of them. Then, developing a scalable algorithm to handle the huge data sparsity is one of the great challenges of recommendation systems.

In the literature, many works have been proposed to face this problem in traditional scenarios, but only a few of them have also been concerned about it in the bandit domain. In general, we identified scalability and sparsity as being handled in distinct ways. Some works have proposed a cluster-based algorithm to address (or even summarise) the huge number of distinct dimensions (i.e., preferences) in a manageable number. The most relevant works in this sense are CLUB [66], DynUCB [163], and COFIBA [133]. In particular, Nguyen and Lauw [163] present studies of how to alleviate the sparsity problem with clusters of users. Other works, like [261] propose to deal with sparse scenarios through social information available from items and users. Indeed, all of these techniques help to improve the scalability of MAB algorithms by training their models only for groups of users. On the other hand, there are other works focused exclusively on such a scalability problem. Rahman and Oh [177], for instance, proposes a parallel version to the

classic UCB algorithm, proposed in [13]. And, Tekin et al. [228] and Nguyen et al. [162] present distributed MAB algorithms for the recommendation. Despite that, we did not find proposals that had goals to deal with both challenges simultaneously.

Cold-Start problem. The cold start problem is a well-known problem for recommendation systems in the literature. It usually refers to the absence of information that makes it more difficult to provide personalised recommendations. In general, there are three cases of cold start that may affect the system [25]:

- **New community:** refers to the start-up of the recommendation, when, although a catalogue of items might exist, almost no users are present and the lack of user interaction makes it very hard to provide reliable recommendations.
- **New item:** a new item is added to the system with some content information, but it has not received any review yet.
- **New user:** a new user makes their registration in the system and accesses it for the first time (they have not provided any interaction yet).

In the literature, all of these cases have been explored by using MAB algorithms. In our SLR we identified that most of the works are related to the *new community* problem due to the evaluation policy applied. As the researchers have usually evaluated the learning quality of bandit algorithms, they simulate a start-up scenario without any information about users and items [208, 77, 172]. However, this is an unrealistic scenario because even if a system is released right now, this system may get more information about its products on the big web. For these reasons, other works have been proposed exclusively to face the *new item* or *new user* problems. Indeed, it is more likely that such scenarios happen in practice where new products are recently released and new users are usually joining the systems.

In general, the new item problem (or item cold-start) often is faced by a hybrid approach that combines the bandit algorithms with content-based (CB) concepts [222, 241]. Basically, these approaches use the items' characteristics to correlate items and user preferences when there is no information about the items. In turn, we found two classes of bandit algorithms that are concerned with the *new user* problem (or user cold-start): (1) the Contextual MAB (CMAB); and (2) the hybrid approaches. The first one applies the contextual information about items or users (e.g., age, gender, contents, etc.) to allow users to explore the system's options [163, 173, 277]. The second one proposes to apply a model selection to handle the first two stages of the user's experience, changing from one model to another when the system has enough data about the user [126, 77, 125, 144, 91, 90].

Privacy Concerns. The growing concern about the privacy of users has affected several e-commerce systems and created many discussions in the recommendation field. The

success of a recommendation is directly related to its personalising ability. Consequently, many algorithms try to collect as much data about the user as possible. In particular, this is even more evident in contextual algorithms [163, 173, 277]. In order to improve the quality of the recommendations, such algorithms dynamically adapt to the users' interests and requirements based on the information collected by the recommendation agent. However, such information is often potentially confidential since it is associated with the users' particular interests in their previous interactions they may prefer to keep secret due to the content accessed, or even their addresses. In this sense, several privacy policies have been discussed over the years, leading many recommendation algorithms to change itself [146, 95, 183]. Even the multi-armed bandits' algorithms should be considered.

In the literature, we identified many proposals that can be applied to bandit algorithms. A first proposal is performing the recommendations in a particular device, like the users' own devices, in order to keep their personal information in their hands [264]. However, a local recommendation does not present promise since the best personalising methods are often related to incorporating useful information from other users who have similar preferences. So, Malekzadeh et al. [146] have proposed to apply a privacy model of *crowd-blending* [80], where a user mixes himself with a number $l \geq 1$ of other users in a way that replacing him data with any other user does not change the results of the recommendation. So, their privacy is protected by this group of other users with the same interests. Another approach, proposed by Zhou et al. [292], suggests storing the central recommendation model on a cloud server, only updating the learning parameters by each local recommendation agent. Then, all training data is kept on the local user agent and no updates are stored on the cloud server for privacy protection. Other works have discussed the privacy problem specifically for bandit algorithms by proposing a system that updates local agents by collecting feedback from other agents in a private manner [95, 183].

Explainable Recommendations. Nowadays, there is a growing consensus that explanations have helped the user to better understand and interpret the rationale of the recommendation system, thereby making it more trustworthy and engaging [282]. Famous platforms like Netflix, Amazon, and Spotify have labelled their recommendations with explanations such as 'because you watched it, you can like this'. Traditionally, their proposals usually follow one of two orthogonal dimensions: (1) the information source or display style of the explanations (e.g., textual sentence explanation, or visual explanation), which represents the human-computer interaction (HCI) perspective; and (2) model to generate such explanations, which represents the machine learning (ML) perspective. At the first one, various visualisation techniques for explaining recommendations have been proposed, including interfaces with concentric circles [112] and pathways between columns [27]. One of the most relevant is the taxonomy proposed by Friedrich and Zanker [78] taking into account the style (e.g., collaborative, knowledge, utility or social expla-

nation style) and paradigm (e.g. content-based, knowledge or collaborative-based) and type of preference model. In turn, from the second perspective, other approaches have been proposed. Kouki et al. [120], for instance, propose a hybrid recommendation system built on a probabilistic programming language and show that explanations improve the user experience of recommendation systems.

In our SLR, we identify two works concerned with explaining the recommendations by applying a reinforcement learning algorithm. First, McInerney et al. [155] proposed that users would respond to explanations differently and dynamically, and thus, a bandit-based approach for exploitation-exploration trade-off would help to find the best explanation orderings for each user. In particular, they proposed methods to jointly learn which explanations each user responds to, which are the best contents to recommend for each user, and how to balance exploration with exploitation to deal with uncertainty. This work shows that just as exploitation-exploration is beneficial to recommendation tasks, it is also beneficial to explanation tasks. Similarly, Wang et al. [247] proposed a model agnostic to a reinforcement learning framework to generate explained sentences for any recommendation model. In this design, the recommendation model to be explained is a part of the environment, while the agents are responsible for generating explanations and predicting the output ratings of the recommendation model based on these explanations. In this case, the agents will learn how to generate sentences with good explainability and correct grammar by trying to optimise the expected reward of the user's ratings for these sentences.

A.3 Summary

In this appendix, we have presented a systematic literature review of Multi-Armed Bandits in the recommendation field to shed light on their applicability and open challenges. By inspecting 1327 articles published from the last twenty years (2000 - 2020), we identified 230 works as the most relevant studies about MAB in the field. These articles were read in detail and analysed to fill a specific data extraction form. This form guides us to answer three main questions by: (1) consolidating an updated picture of the main research conducted in this area so far; (2) highlighting the most used concepts and methods, their core characteristics, and their main limitations; and (3) evaluating the applicability of MAB-based recommendation approaches in some traditional RSs' challenges, such as data sparsity, scalability, cold-start, privacy, and explainability.

The discussions indicate that several advances were already achieved by the existing works. In general, most existing works are focused on UCB or Thompson Sampling algorithms combined with concepts of collaborative filtering, such as Matrix Factorisation, clustering approaches, and others. Especially, by inspecting these works in detail, we

also presented a standard implementation for each traditional MAB algorithm (ϵ -Greedy, UCB, and TS) when they are combined with matrix factorisation concepts. Moreover, the analyses of this work also identify the main methodologies and metrics applied to simulate the online environment. In the inspected works, we observe a simple abstraction of the traditional evaluation criteria of reinforcement learning scenarios. In these cases, the system is only concerned about the model's learning by measuring the reward or regret assigned. In other words, current methodologies have only been concerned with the prediction power of the recommendation – an outdated concept in the current recommendation literature that is more worried about the user's satisfaction or engagement. Moreover, based on our knowledge achieved by reading the works identified by our SLR, we believe that the traditional challenges of the recommendation domain should be more studied. Most of them, like the cold-start problem, can directly affect the performance of current bandit algorithms.

Appendix B

iRec: An Interactive Recommendation Framework

Despite the relevancy of Multi-Armed Bandits for both academia and industry, their applicability in the online recommendation task is still recent and new. There still is a lack of consensus about the best evaluation practices to address them. Only a few libraries and one framework were created to store bandit-based algorithms and provide some evaluation metrics. Moreover, most of them are completely focused on the MAB problem and not well designed for a recommendation scenario. In this sense, during the development of this work, we built a new framework that contains several algorithms, evaluation methodologies, evaluation metrics, and a compatible infrastructure with the main recommendation datasets in the literature. This framework is named iRec and it is deeply explained in this appendix. It represents another extra part of this thesis and it was published at the SIGIR Conference [209].

B.1 Libraries & Frameworks

Despite the recent advances and the new bandit algorithms published, only a few works have been concerned about proposing or identifying a reproducible and fair experimental evaluation. Even with the growing consensus about reproducibility, current works about Multi-Armed Bandits in Recommendation Systems do not provide complete details about their experiments and even their source codes. Moreover, it is common that each article often applies its specific methodology and does not follow the same evaluation pattern. And, unfortunately, contrasting the current consensus of the RS community to measure the user's experience and engagement, most of the research papers about bandit in recommendation systems have only been concerned about the prediction power of each method. In general, they usually evaluate the traditional reward/regret metric from the reinforcement learning theory. Other metrics related to diversity, novelty, serendipity, unexpectedness, or even coverage have not been included in mostly of the recent works on this field.

In a huge inspection of the literature, we only found three recent works that tried to handle the evaluation of bandit algorithms in the recommendation field.

- **MABWiser** [216, 215] is an open-source library implemented in Python to address Multi-Armed Bandit algorithms with rapid prototyping and hyperparameter tuning. It provides batch and online simulations for traditional MAB solutions and Contextual Bandits (i.e., adaptations for the recommendation field). Despite the classical algorithms, like the ϵ -Greedy, UCB, TS, and Softmax [122], it also contains Parametric and Non-Parametric bandits. The parametric models include regression-based techniques such as LinUCB and LinTS. In terms of non-parametric ones, they implement some clustering approaches, like k-means and KNN algorithms.
- **Open Bandit Pipeline (OBP)** [189] is an open-source library containing both Multi-Armed Bandit algorithms and a series of evaluation policies. In short, this library can be divided into four main modules: a dataset module, which provides mechanisms to work on the datasets; a module where the implemented models are described; a simulation module, which allows to contrast and evaluate the performance of bandits algorithms; and finally an evaluation policy module, which has generic abstract interfaces ideal for custom implementations in which researchers can add and evaluate their algorithms.
- **BEARS** [19] is the unique framework proposed so far to mitigate the lack of pattern in bandit evaluations and support reproducible offline assessments. It is implemented in Python and it consists of simple building blocks to configure agents (solution approaches) and environments (problem settings). However, this framework only contains a few classic algorithms and is still limited to the traditional accuracy metrics from the RL scenario. Moreover, BEARS does not provide any data preparation strategy, and it does not allow any hyperparameter tuning for the algorithms.

Framework	Data Preparation							Recommendation Models													Evaluation												
	Filtering			Split		Tuning		Multi-Armed Bandits						Non-personalised							Metrics												
	By rating	By users	By items	Temporal	Random	Fixed	Grid Search	Random	ϵ -Greedy	UCB	TS	Softmax	ICTR	PTS	kNN Bandit	Linear ϵ -Greedy	Linear TS	Linear UCB	NICF	COFIBA	GLM-UCB	Random	Popular	Best Rated	Entropy	log(pop)*ent	Accuracy	Novelty	Diversity	Coverage	Error		
BEARS									✓	✓	✓																						✓
OBP									✓		✓					✓	✓	✓						✓									
MABWiser							✓		✓	✓	✓	✓	✓			✓	✓	✓						✓	✓								
iRec	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table B.1: An overview of each framework proposed in the literature and our iRec.

Contrasting these proposals, the *iRec* fully encompasses the experimental process of evaluating a recommendation system from the data entry until the evaluation methodology. It allows users to evaluate a complete study case from the perspective expected

in a recommendation system. Table B.1 presents an overview of all points covered by the libraries and the framework identified so far. It compares everything expectable in an evaluation framework from the data preparation to the evaluation process. As we can notice, the *iRec* has the highest coverage of methodologies and methods in the literature.

B.2 The iRec Framework

In this section, we detail our iRec framework¹ for an interactive recommendation scenario. It is structured in three main components as usually made by classical frameworks in the RS field [11, 190, 286]. These main components are:

- **Environment Setting:** responsible for loading, preprocessing, and splitting the dataset into train and test sets (when required) to create the task environment for the pipeline;
- **Recommendation Agent:** responsible for implementing the recommendation model required as an interactive algorithm that will interact with the environment;
- **Experimental Evaluation:** responsible for defining how the agent will interact with the environment to simulate the interactive scenario and get the logs required for a complete evaluation.

In short, the framework application starts with the Environment Setting component. By this component, a researcher can choose a database to be loaded for the experiment and define and execute all the data preparation modules to be applied to it, such as prefiltering and the data splitting strategy to create the training and testing sets. In the iRec, while the training set is used to tune the recommendation algorithms, the testing set represents the target audience of the recommendations in the interactive scenario (task environment). In this work, the iRec is focused on interactive personalised recommendations. Consequently, all models aim to identify the best item(s) for each user at each iteration. The Recommendation Agent chooses the item(s) according to the implemented model. The agent's interaction with the environment occurs through the Evaluation Policy defined by the researcher. This policy is similar to classical Reinforcement Learning approaches, where one item (or a set of items) is recommended at each algorithm iteration (i.e., trial). For a predetermined time (number of trials), the agent performs the recommendations within the task environment, receiving a positive or negative reward and updating its knowledge (if necessary). All logs (i.e., actions taken by

¹The iRec is available on <https://github.com/irec-org/irec>

the agent and rewards provided by the target audience) are then recorded by the framework to allow a proper evaluation. The Experimental Evaluation component analyses these logs, applies traditional recommendation evaluation metrics, and performs all the statistical tests required to conduct a fair evaluation. These components are illustrated in Figure B.1 by different colours that are discussed in the next sections.

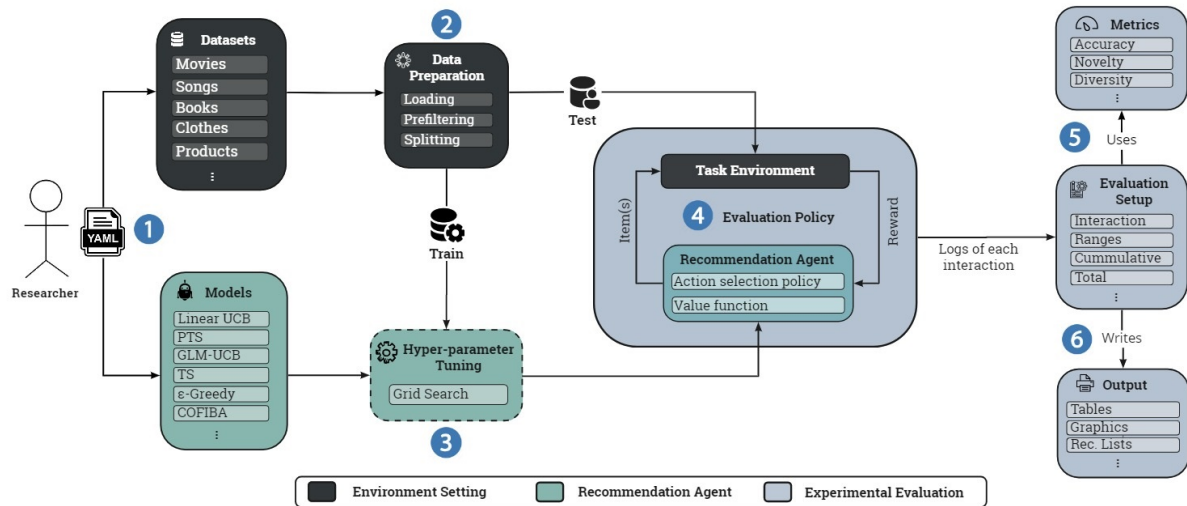


Figure B.1: An overview of the iRec framework. It is composed of three main components that allow a researcher to simulate an interactive recommendation scenario and compare distinct algorithms by a fair evaluation of their quality.

B.2.1 Environment Setting

The first component aims to create the entire environment for the recommendation scenario by preparing the databases for the framework’s execution. It is managed by the Data Preparation operator which identifies the selected database and applies (if required) three main operations: loading, filtering, and splitting. The available strategies for each module are described below. They will create the desirable Task Environment that the researcher wants to simulate. This environment is an abstraction of an e-commerce system like any selling platform such as Amazon, Farfetch, and others, or even a scenario of entertainment like Netflix, Spotify, or Podimo.

Loading. The iRec framework has two different loading strategies:

- *Full Data*: determines that the system must read the entire database and apply a splitting strategy;
- *Split Data*: determines that the data is already split into training and testing and the system should read these two data.

In both, the research should set the dataset path and its format. Currently, the iRec follows the usual pattern in recommendation datasets where each row is formatted as *userId*, *itemId*, *rating*, *timestamp*. If the researcher would like to change it for other datasets, s/he should only create a new operator that extends the default *Loader* and sets its new format definition. Table B.2 lists the current datasets available to iRec.

Datasets	Domain	Users	Items
MovieLens	Movies	138,493	26,744
Netflix	Movies	429,584	17,770
Yahoo Music	Music	10,000	13,214
LastFM	Music	5,000	36,718
Good Books	Books	53,424	10,000
Good Reads	Books	14,639	9,999
Amazon Store	Products	14,776	68,921
Clothing Fit	Clothes	105,508	5,850

Table B.2: Datasets currently accepted by the iRec.

Prefiltering. This module aims to apply all preprocessing selected by a researcher. In general, preprocessing is a common task in the recommendation field since the datasets usually have too much data and may directly increase an experiment’s execution time. Moreover, to avoid a huge sparsity in the data, many works also decide to remove some data by following certain criteria. In this way, the preprocessing stage applies filters in the original dataset to remove users, items, or even data outliers. Currently, the iRec has three approaches implemented to filter the data by:

- *Users*: selecting a predefined number random of users or those with fewer interactions than a certain threshold;
- *Items*: selecting a predefined number of items or those with fewer interactions than a certain threshold;
- *Ratings*: filtering all user-item pairs that have resulted in ratings lower than a certain threshold.

Splitting. This module aims to split the data into train, test, and validation sets. These approaches were created to simulate the scenario of new users joining the system. Thus, every split approach selects a group of users (and their ratings) for one set and the remaining group of users for the other set. In this module, the researcher can also define some parameters to create the validation sets used in the hyperparameter tuning. The three implemented approaches are:

- *Random*: a strategy that randomly selects the users to the sets;

- *Temporal*: a strategy that selects the last users to join the system based on their first timestamp. The idea is to avoid a bias of breaking the chronological order of the user's actions [10, 21].
- *Global Timestamp*: a strategy that selects a global timestamp of the dataset to create the train and test sets. All users' interactions that happened before this timestamp are set in the train. The remaining of their interactions are discarded. The users who just joined the system before this temporal cut are set in the test set. In contrast to the other one, in this way, future users' actions will never be used in the training set.

B.2.2 Recommendation Agent

This component aims to define an agent to interact with the previously defined environment by making recommendations and receiving the user's feedback. Similar to some approaches used in the Reinforcement Learning theory, an agent is represented by two main components: a **Value Function** and an **Action Selection Policy**. The *value function* represents the agent's goals, quantifying the expected consequences of its decisions. In this case, where the agent is a recommendation system, the *value function* represents the utility of each item for a user according to the algorithm prediction. The reward usually consists of a scalar value representative of the user's explicit/implicit feedback. In turn, the *action selection policy* represents the policy used by the agent to choose one (or more) items to be recommended. In general, such policies have two competing objectives: *exploiting* the items with the highest rewards in the past; or *exploring* unknown items to improve the system's knowledge. Thus, the definition of an agent that implements the ϵ -greedy, for example, consists of defining the *value function* as its original objective function and implementing its own *selection policy* that randomly explores new arms with probability ϵ . In the UCB and TS cases, when the objective function is already responsible for the exploration part, the implementation consists of only defining the *value function*. In these cases, the *selection policy* performs a greedy selection by returning the arm with the highest assigned value.

Recommendation Models. Currently, the iRec provides several state-of-the-art bandit algorithms from the literature:

- **ϵ -Greedy** [13]: classical bandit model that random explores other arms with probability ϵ .
- **UCB** [13]: it calculates a confidence interval for each item and tries to shrink the confidence bounds at each iteration.
- **Thompson Sampling (TS)** [51]: it follows a Gaussian distribution of items and users to predict based on samples.

- **Linear TS** [4]: a linear adaptation of the original TS to measure latent dimensions using Probabilistic Matrix Factorization (PMF).
- **Linear ϵ -Greedy** [288]: linear exploitation of the latent factors defined by the PMF of the classical ϵ -Greedy.
- **Linear UCB** [288]: an adaptation of the original LinUCB Li et al. [129] to measure the latent dimensions by a PMF formulation.
- **GLM-UCB** [288]: an adaptation of the Linear UCB with a sigmoid form that makes a time-dependent exploration.
- **ICTR** [244]: it is a topic regression model that utilises the TS and controls the items' dependency by a particle learning strategy.
- **kNN Bandit** [192]: a variant of the nearest-neighbours applied a classical parameter-free TS algorithm.
- **NICF** [297]: a combination of neural networks and collaborative filtering that performs a meta-learning of the user's preferences.
- **COFIBA** [133]: it defines an upper-confidence-based to combine it in an adaptive clustering of users and items.
- **PTS** [114]: it is a PMF formulation for the original TS that applies particle filtering to guide the exploration of items over time.
- **Cluster-Bandit (CB)** [200]: it is a new bandit algorithm based on clusters to face the cold-start problem.

In addition to the models mentioned above, other non-personalised recommendation systems are also implemented in the iRec. Some examples are the *Most Popular*, *Random*, *Entropy*, and *Best-Rated* [75]. They are extremely useful as baselines, as they can be combined with other reinforcement learning models.

Hyperparameter Tuning. Contrasting the current Reinforcement Learning approaches, the iRec allows researchers to make a hyperparameter optimisation in their algorithms. For complex models, like those based on a neural network, this module is essential to ensure the quality of such recommendations [185]. Currently, our framework provides a *Grid-Search* strategy to perform an exhaustive search for the best combination of hyperparameters. This search is done according to a range of values defined by the researcher as a parameter for the *Tuning* module. According to the parameters required for the value function selected for the experiment, the researcher can define fixed values or even a search interval to be explored by the tuner. Based on these definitions, the iRec will automatically run the recommender agent (with its value function and action selection

policy) for all parameters in the search. This process is done in parallel to avoid an expensive running time. In the end, this module stores the best parameters found. This is an optional step, represented by a dashed box in Figure B.1. The models are optimised according to the evaluation policy and evaluation setup detailed in the next section.

B.2.3 Experimental Evaluation

This component provides the evaluation policy used to define how the recommendation agent will interact with the previously defined environment. Its goal is to provide a fair and reproducible evaluation for several algorithms in the interactive recommendation scenario. Moreover, contrasting all other frameworks available in the literature, the iRec evaluation is centred on metrics and methodologies specifically designed by the RS community in the last years [186, 234, 210]. Its process is split into four main modules.

Evaluation Policy. This module is responsible for determining how the recommendation agent will interact with the previously defined environment. Basically, it implements the classical reinforcement learning algorithm where the system:

- (1) selects the target user;
- (2) gets the action from the recommendation model;
- (3) receives the feedback from the user to that specific action;
- (4) updates the model’s knowledge with this reward.

Currently, the iRec contains two main evaluation policies already defined: *Fixed Interactions* and *User-Driven Interactions*. In both, each user is randomly selected and each action will not be performed more than once for him/her. Their main difference is in the stop criteria of the interactive scenario. In the first one, each user will be selected for T times. Thus, the system will perform $T \times |U|$ iterations, where $|U|$ is the number of distinct users available for the evaluation. The number T is predefined by the researcher as a parameter. In turn, in the second one, the system will perform new actions until it hits all items registered in the user historical. The idea is to make an exhaustive experiment to observe which algorithm takes more time to reach all items previously rated by each user.

Evaluation Setup. This module aims to guide the entire evaluation process over the logs from each iteration of the Evaluation Policy. As the iRec stores each execution log, the researcher can define how s/he would like to evaluate the actions selected by the recommendation model after all interactions. The iRec already has some evaluation setups previously defined:

- *Interaction*: it evaluates the selected metrics’ overall interactions registered during the recommendation process. Given a scenario in which 100 interactions were performed, for instance, this strategy would evaluate each one separately.

- *Intervals*: it first aggregates some consecutive interactions in an interval to then evaluate the selected metrics over each group. For instance, in the execution of 10 interactions, the researcher can define two intervals to be evaluated by the system. Then, it will create two groups of 5 interactions each (one from the 1st to the 5th interaction and another one from the 6th to 10th interaction) to evaluate the selected metrics.
- *Cumulative*: it evaluates the interactions cumulatively from the first one until a specific value. In this way, the researcher can evaluate the accumulated result from the 1st to the 10th interaction, then from the 1st to the 15th interaction, and so on.
- *Total*: it evaluates the whole recommendation process as one unique procedure. For example, if certain items were recommended during 100 interactions, the metric will be calculated only at the 100th interaction.

Metrics: This module contains all evaluation metrics available in the iRec. Its goal is to provide distinct options to be selected during the previous setup. These metrics are suitable to the recommendation scenario and are usually split into a few groups: Accuracy [198, 291], Coverage [79], Novelty [234] and Diversity [278]. In our architecture, they follow an implementation pattern where each metric has two methods: (1) **compute**, in which the entire calculation is performed for a given user; and (2) **update**, which updates the historic of items in each user during the interactive scenario. Again, the metrics to be used in the evaluation process are defined in a configuration file. The metrics already implemented in iRec are:

- **Precision** [186]: it is the percentage of relevant items recommended considering the number of recommended items;
- **Recall** [186]: it the percentage of relevant items recommended considering the entire set of relevant items;
- **Hits** [186]: it is the number of recommendations that hits the history of each user;
- **EPC** [234]: it is measured by the expected number of relevant items not previously seen by the user (novelty);
- **ILD** [234]: it is measured by the Pearson correlation of the item's features vector between the list of items recommended (diversity);
- **EPD** [234]: it is a novelty that measures the distance between the items in the user's profile and the recommended ones;
- **Gini Coefficient** [192]: it is measured as the inverse of cumulative frequency that each item is recommended;

- **Users Coverage** [210]: it represents the percentage of distinct users that are interested in the recommended items.

Statistical Tests. In the iRec framework, we designed a specific module to make all statistical tests over the evaluation metrics. The proper use of statistical tests allows the researchers to feel more confident about the results. The idea is to compare the results obtained by each method and identify which of them statistically outperforms the others. In iRec, we implemented the Wilcoxon test [255] for non-parametric distributions, usual studied in the RS community.

Output: This is a specific module that offers several methods to visualise the results obtained. The iRec offers different ways of looking at the results through: (1) tables, in which the user can select different methods and metrics; and (2) graphics, in which researchers can find different insights from the data. If desired, the researcher can still access the entire experiment log, which contains information about the lists of recommendations, informing how many and which items were recommended for a given user.

B.3 Framework

This section presents a practical evaluation of our framework, detailing how to configure and execute each of the aforementioned steps. In this sense, we designed a command-line front-end named iRec-cmdline² that relies on iRec and illustrate how to prepare, execute and evaluate a practical experiment with our framework. This application is designed based on the MLOps concept, a set of practices that aims to deploy and maintain reliably and efficiently machine learning models in production. Its structure is illustrated in Figure B.2. The iRec-cmdline contains a repository to: (1) **settings**, where it records all configurations (i.e., parameters) required by each component; and (2) **scripts**, where several scripts provide a command-line interface to trigger the entire framework.

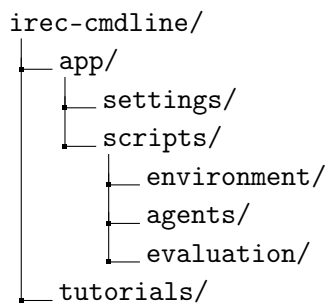


Figure B.2: iRec-cmdline: an application of the framework.

²Available in: <https://github.com/irec-org/irec-cmdline>

Script	Description	Parameters
<code>download_data.py</code>	It downloads the datasets listed in the framework.	<code>--dataset_name</code> <one or more datasets>
<code>generate_dataset.py</code>	It reads the datasets, and applies prefiltering and splitting strategies.	<code>--dataset_loaders</code> <one or more datasets> <code>--agents</code> <agents>
<code>run_agent_search.py</code>	It executes the parameter search routine based on the specified agents and datasets.	<code>--agents</code> <agents> <code>--dataset_loaders</code> <datasets> <code>--evaluation_policy</code> <eval policy> <code>--tasks</code> <num tasks (optional)>
<code>eval_agent_search.py</code>	It evaluates the parameter search results.	<code>--agents</code> <agents> <code>--dataset_loaders</code> <datasets> <code>--evaluation_policy</code> <eval policy> <code>--metric_evaluator</code> <metric eval> <code>--metrics</code> <metrics>
<code>print_search.py</code>	It displays and updates the settings with the best search parameters.	<code>--agents</code> <agents> <code>--dataset_loaders</code> <datasets> <code>--evaluation_policy</code> <eval policy> <code>--metric_evaluator</code> <metric eval> <code>--metrics</code> <metrics> <code>-t</code> (print only the best parameter) <code>-d</code> (save best parameter)
<code>run_agent_best.py</code>	It runs agents on datasets based on the best tuning parameters.	<code>--agents</code> <agents> <code>--dataset_loaders</code> <datasets> <code>--evaluation_policy</code> <eval policy> <code>--tasks</code> <num tasks (optional)>
<code>eval_agent_best.py</code>	It evaluates the results obtained after running the agents.	<code>--agents</code> <agents> <code>--dataset_loaders</code> <datasets> <code>--evaluation_policy</code> <eval policy> <code>--metric_evaluator</code> <metric eval> <code>--metrics</code> <metrics>
<code>print_latex_htable.py</code>	It generates Table B.4 with results.	<code>--agents</code> <agents> <code>--dataset_loaders</code> <datasets> <code>--evaluation_policy</code> <eval policy> <code>--metric_evaluator</code> <metric eval> <code>--metrics</code> <metrics>

Table B.3: The main scripts

The `settings` is composed of several *yaml* files responsible to set up all parameters from the modules of the framework. A *yaml* is a human-readable data-serialisation language commonly used for configuration files. Thus, a researcher can set the parameters of the agents, the hyperparameters tuning ranges, the functions of the preparation of the dataset, the required evaluation policies, all evaluation metrics to be applied, and others. In turn, the `scripts` contains several command-line interfaces to start operations of the *iRec*. The proper use of these scripts allows the researcher to create an execution pipeline according to their experimental setting. They are related to each one of the three main components of the *iRec*. Moreover, these scripts are integrated into MLflow, an open-source platform that manages the life cycle of machine learning models, recording their execution logs. It means that all framework modules are integrated into this platform, facilitating the management of experiments. [Table B.3](#) presents the main scripts that deal with the execution of common tasks for evaluating interactive recommender systems, as well as the other scripts responsible for hyperparameter setting, results' visualisation, and others. The next sections guide this experimentation process by showing each configuration performed to run the *iRec* by command lines.

B.3.1 Environment Setting

As previously explained, this component is responsible for preparing the databases for the complete experimental process. In the *irec-cmdline*, this component is executed by the script named *generate_dataset.py*. It performs the loading, prefiltering, and splitting steps of the *iRec* according to the parameters configured in the file *dataset_loaders.yaml*. The Configuration B.1 presents an example of how to configure these steps by considering the *MovieLens 1M* dataset. The dataset name is the first parameter to be entered. After that, the second parameter to be defined is which loader method will be used to load the database. Currently, *iRec* has two methods implemented, as aforementioned. For the *MovieLens* dataset, we defined the *Full Data* to perform the splitting with the approaches implemented in the *iRec*.

Configuration B.1: dataset_loaders.yaml

MovieLens 1M:

```
FullDataLoader:
  dataset:
    path: ./data/datasets/MovieLens 1M/ratings.csv
    random_seed: 0
    file_delimiter: ","
    skip_head: true
  # prefiltering: None
  prefiltering:
    filter_users:
      min_consumption: 50
      num_users: 10
    filter_items:
      min_ratings: 1
      num_items: 100
  splitting:
    strategy: temporal
    train_size: 0.8
    test_consumes: 5
  # validation: None
  validation:
    validation_size: 0.2
```

The *Loader* requires the researcher to set information about the dataset, the prefiltering, and the splitting approach. In the first one, the researcher should define *path* where the data is located, the *file_delimiter* that splits each row of the dataset, and if exists any *head* to be skipped. By default, *iRec* will read the datasets available in *app/data/datasets/*. Then, in the second one, the researcher must define the parameters

for the prefiltering. In our example are applied the two possible filters are to select 10 users and 100 items with certain criteria informed by the other parameters. If the researcher wants to avoid this step, s/he can define the *prefiltering* as *None* as commented out in the configuration file example. Finally, the researcher should also define which *splitting* strategy will be used by setting the option *strategy*. We use the temporal strategy in this configuration file, with 80% of the data composing the training and the rest for the test. Again, we left comments on how the configuration would be if we opted for the random strategy. Moreover, in this example, we also add the optional setting for the validation set. Setting this option and defining the *size* of the validation set, the *iRec* will perform the hyperparameter tuning in the next steps.

B.3.2 Recommendation Agent

As aforementioned, every recommendation system is an agent with an *action selection policy* and a *value function*. While the value function determines the importance of each item to a user (i.e., a prediction), the selection policy determines which criteria should be followed to select an action (i.e., make a recommendation). Thus, in the *irec-mdline*, the researcher must follow a template to define the recommender system in the file *dataset_agents.yaml* as illustrated in the Configuration B.2. For each dataset, the researcher should define distinct agents to be executed by our framework. In our example, we selected three classic MAB models to be evaluated in the *MovieLens 1M* dataset: *UCB*, *ϵ -Greedy* and *Thompson Sampling*. Each agent is configured with a tag related to the agent implementation that the algorithm should follow.

In this example, we set the tag as *SimpleAgent*, which means we must define only one action selection policy and one value function for each one of them. *iRec* also allows agents based on ensemble strategies [222] by using the tag *EnsembleAgent* and, in this case, more than one action selection policy and/or more than one value function can be set. After informing the agent type, we must provide the action selection policy and which value function each agent will use. As adopted by BEARS [19], the selection policy and the value function are implemented separately in our framework. Thus, after defining the type of agent, we must inform them in sequence. In terms of the action selection policy, the ϵ -Greedy traditionally uses the *egreedy* policy which selects the best items based on the diversification parameter ϵ to execute random recommendations [13]. In turn, UCB and TS use another policy, called *greedy*, that selects the best items at that moment [13, 51]. Regarding the value function, each agent has a value function that refers to itself. If the researcher wants to extend these value functions with new methods, we strongly recommended to call them by the same name as the new method.

Configuration B.2: dataset_agents.yaml

```
MovieLens 1M:
  EGreedy:
    SimpleAgent:
      action_selection_policy:
        ASPEGreedy:
          epsilon: 0.1
      value_function:
        EGreedy: {}
  UCB:
    SimpleAgent:
      action_selection_policy:
        ASPGreedy: {}
      value_function:
        UCB:
          c: 0.1
  ThompsonSampling:
    SimpleAgent:
      action_selection_policy:
        ASPGreedy: {}
      value_function:
        ThompsonSampling:
          alpha_0: 1
          beta_0: 100
```

B.3.3 Experimental Evaluation

The next step is to set the iterative process of recommending items, by setting how the agent should interact with the environment. First, it is necessary to set up an *evaluation policy* using the file *evaluation_policies.yaml*. In the example presented in Configuration B.3, we selected the policy *Fixed Interactions* and set its three parameters: the number of interactions; the size of the recommended list of items in each interaction; and a variable that informs if (or not) the researcher would like to store the entire log of the process. As discussed earlier, other policies can be applied. We present the other possibility implemented by the *iRec* commented out in the configuration file example. The execution of the experimental evaluation is performed by the *run_agent_best.py* script. This script performs according to the parameters configured in the files *dataset_loaders.yaml*, *dataset_agents.yaml* and *evaluation_policies.yaml*. The results of all interactions are recorded in log files for the evaluation setup phase.

Configuration B.3: evaluation_policies.yaml

```
FixedInteraction:
  num_interactions: 100
  interaction_size: 1
  save_info: True
#UserDrivenInteractions:
# interaction_size: 1
# recommend_test_data_rate_limit: 0.1
```

After that, the evaluation of the results is performed by the script *eval_agent_best.py*. In addition to the *dataset_loaders.yaml*, *dataset_agents.yaml* and *evaluation_policies.yaml* files, this script also uses the parameters configured in the *metric_evaluators.yaml* file, which defines how the researcher wants to evaluate the actions performed by the recommendation agents after all interactions. The list of metrics to be considered is informed to the script as a parameter— see Table B.3. The *irec-cmdline* provides different evaluation setups. The Configuration B.4 presents an example, considering a cumulative strategy, defined by the tag *Interaction*. For this strategy, the researcher must set up four main parameters:

1. the number of items to be selected per interaction (e.g., 1);
2. the interval of interactions to measure the evaluation metrics (e.g., at the interactions number 10, 50, and 100);
3. the total number of interactions (e.g., 100);
4. the minimum threshold for an item to be considered relevant to a user – an important threshold for the evaluation metrics, such as Recall, EPD and EPC.

The other options are commented out in the example illustrated on Configuration B.4. All of them are quite similar in terms of their configuration, but they can perform distinct analyses for the researcher.

Configuration B.4: metric_evaluators.yaml

```
Interaction:
  interaction_size: 1
  interactions_to_evaluate: [10, 50, 100]
  num_interactions: 100
  relevance_evaluator_threshold: 3.999
#Cumulative:
# interaction_size: 1
# interactions_to_evaluate: [5, 10, 20, 50, 100]
# num_interactions: 100
```

```

# relevance_evaluator_threshold: 3.999
#Total:
# interaction_size: 1
# interactions_to_evaluate: [5, 10, 20, 50, 100]
# num_interactions: 100
# relevance_evaluator_threshold: 3.999
#Intervals:
# interaction_size: 1
# interactions_to_evaluate: [5, 10, 20, 50, 100]
# num_interactions: 100
# relevance_evaluator_threshold: 3.999

```

After setting the configuration files and running the scripts, the researcher can choose how to view the final results. Currently, `irec-cmdline` offers scripts that automatically generate different forms of visualisation in graphics and tables. The script `print_latex_htable.py` provides a table to visualise the results – see Table 3. As the name implies, this file automatically generates a horizontal table with all models, datasets, and metrics informed by the execution parameters. This script also performs the statistical tests implemented in the *iRec* (the Wilcoxon test) over the results identified, and it also highlights significant gains and losses in the printed table.

Dataset Measure	MovieLens 1M														
	Hits			Recall			EPC			ILD			UsersCoverage		
	10	50	100	10	50	100	10	50	100	10	50	100	10	50	100
Popular	3.806▲	13.978	22.198	0.062▲	0.204	0.301	0.536	0.637	0.697	0.349	0.381	0.387	0.893	0.993	0.995
UCB	2.419	9.433	16.231	0.035	0.130	0.217	0.713	0.761	0.786	0.400	0.412	0.418	0.789	0.967	0.987
TS	2.960	11.620	19.061	0.045	0.164	0.257	0.645	0.713	0.755	0.381	0.395	0.404	0.852	0.981	0.997
ϵ -Greedy	2.405	9.445	16.328	0.036	0.132	0.219	0.711	0.761	0.786	0.400	0.412	0.418	0.988	0.991	0.995
Linear UCB	3.115	15.025	26.887▲	0.050	0.203	0.346▲	0.682	0.755	0.790	0.375	0.395	0.404	0.816	0.955	0.987
GLM-UCB	2.009	14.108	25.485	0.031	0.200	0.341	0.736	0.750	0.781	0.424	0.403	0.407	0.821	0.978	0.995
NICF	3.364	8.182	11.687	0.054	0.119	0.162	0.604	0.780	0.836▲	0.380	0.428	0.441▲	0.890	0.983	0.988
PTS	3.685	15.639▲	24.466	0.058	0.229▲	0.329	0.601	0.667	0.731	0.368	0.382	0.402	0.995▲	0.998▲	0.998▲
ICTRS	0.661	6.770	14.556	0.008	0.091	0.195	0.899▲	0.808▲	0.794	0.456▲	0.428▲	0.421	0.422	0.949	0.987
CB	3.145	13.687	22.041	0.051	0.200	0.298	0.636	0.654	0.709	0.411	0.388	0.394	0.874	0.991	0.995

Table B.4: Experimental results obtained using all configurations files described in Section 4. The results were compared based on Wilcoxon Test with p-value=0.05. The ▲ symbol indicates statistical best results.

Besides that, as mentioned in Section 3.1, *iRec* provides an optional module to perform a hyperparameter tuning. The researcher should select the tuning function and define the range of parameter(s) of the agent(s). In the `irec-cmdline`, it can be done by defining this information in the file `agents_variables.yaml` as illustrated in the Configuration B.5. The first name should refer to the tuning approach selected (e.g., the Grid Search). Then, the remaining setting file is quite similar to the one that represents the agent templates illustrated in the Configuration B.2. However, instead of passing only one parameter, the researcher must set a list of parameters to be explored. This example presents the configuration for the classical bandit algorithms of ϵ -Greedy ($\epsilon \in [0.1, 1]$), UCB ($c \in [0.1, 1]$) and Thompson Sampling ($\alpha \in [0.1, 1]$; $\beta \in [1, 100]$). Then, running the script `eval_agent_search.py` offered by the `irec-cmdline`, it will execute the

agent recommendation with the previously selected evaluation policy to search between the parameters. It considers the files *dataset_loaders.yaml*, *evaluation_policies.yaml* and *metric_evaluators.yaml* previously defined. In the end, the researcher can also run the script *print_search.py* to visualise the best parameters found and automatically update the file *dataset_agents.yaml* with them. Thus, in the next executions, the *irec-cmdline* will run all the algorithms with the best parameter identified.

Configuration B.5: agents_variables.yaml

```

GridSearch:
  EGreedy:
    SimpleAgent:
      action_selection_policy:
        ASPEGreedy:
          epsilon: linspace(0.1, 1, 5)
      value_function:
        EGreedy: {}
  UCB:
    SimpleAgent:
      action_selection_policy:
        ASPGreedy: {}
      value_function:
        UCB:
          c: [0.1, 0.5, 1]
  ThompsonSampling:
    SimpleAgent:
      action_selection_policy:
        ASPGreedy: {}
      value_function:
        ThompsonSampling:
          alpha_0: linspace(0.1, 1, 5)
          beta_0: linspace(1, 100, 10)

```

B.4 Summary

In this appendix, we present the iRec, a complete framework for evaluating interactive recommendation systems (RS). The iRec aims to deal with the lack of consensus about the best evaluation practices in this area, providing a complete environment for a reproducible evaluation and fair comparisons of recommendation systems. Its structure provides compatibility with comprehensive datasets adopted in the literature, different data preprocessing strategies, seventeen recommendation models, a complete hyperparam-

eter optimisation, and nine evaluation metrics with distinct goals (i.e., accuracy, novelty, diversity, coverage, etc). Moreover, iRec provides multiple evaluation policies currently used in the literature, statistical tests to compare the algorithms' performance, and different results' visualisations. Indeed, this framework was used to perform all experiments in this thesis.