UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

MARLESSON RODRIGUES OLIVEIRA DE SANTANA

Framework para Sistemas de Recomendação Baseados em *Neural Contextual Bandits* com Restrição de Justiça

Goiânia
2024

SIBI
SISTEMA DE BIBLIOTECAS UFG

PRPG
PRÓ-REITORIA DE PÓS-GRADUAÇÃO

UFG

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o(a) autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico: **[ ] Dissertação** **[X] Tese**

2. Identificação da Tese ou Dissertação:

Nome completo do(a) autor(a): Marlesson Rodrigues Oliveira de Santana

Título do trabalho: Framework para Sistemas de Recomendação Baseados em Neural Contextual Bandits com Restrição de Justiça

3. Informações de acesso ao documento:

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

Independente da concordância com a disponibilização eletrônica, é imprescindível o envio do(s) arquivo(s) em formato digital PDF da tese ou dissertação.

DocuSigned by:
Marlesson Rodrigues Oliveira de Santana
02B45B5B41014BA...
Assinatura do(a) autor(a)[2]

Ciente e de acordo:
DocuSigned by:
Anderson da Silva Soares
F3E6D79F3544454...
Assinatura do(a) orientador(a)[2] Data: 01 / 04 / 2024

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante: a) consulta ao(à) autor(a) e ao(à) orientador(a); b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação. O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

[2] As assinaturas devem ser originais sendo assinadas no próprio documento. Imagens coladas não serão aceitas.

MARLESSON RODRIGUES OLIVEIRA DE SANTANA

Framework para Sistemas de Recomendação Baseados em *Neural Contextual Bandits* com Restrição de Justiça

Tese apresentada ao Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Doutor em Ciência da Computação.

Área de concentração: Inteligência Artifical.

Orientador: Prof. Anderson da Silva Soares

Goiânia
2024

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Santana, Marlesson Rodrigues Oliveira de
Framework para Sistemas de Recomendação Baseados em Neural Contextual Bandits com Restrição de Justiça [manuscrito] / Marlesson Rodrigues Oliveira de Santana. - 2024.
XCIX, 99 f.: il.

Orientador: Prof. Dr. Anderson da Silva Soares.
Tese (Doutorado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação, Goiânia, 2024.
Apêndice.
Inclui siglas, tabelas, lista de figuras, lista de tabelas.

1. recomendação multistakeholder. 2. justiça na recomendação. 3. aprendizado por reforço. I. Soares, Anderson da Silva , orient. II. Título.

CDU 004

MARLESSON RODRIGUES OLIVEIRA DE SANTANA

Framework para Sistemas de Recomendação Baseados em *Neural Contextual Bandits* com Restrição de Justiça

Tese defendida no Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás como requisito parcial para obtenção do título de Doutor em Ciência da Computação, aprovada em 27 de Maio de 2024, pela Banca Examinadora constituída pelos professores:

Prof. Anderson da Silva Soares
Instituto de Informática – UFG
Presidente da Banca

Prof. Thierson Couto Rosa
Instituto de Informática – UFG

Prof. Cedric Luiz de Carvalho
Instituto de Informática – UFG

Prof. Aluizio Fausto Ribeiro Araujo
CIN-UFPE

Prof. Adriano Alonso Veloso
DCC-UFMG

<Dedico este trabalho ao meu filho Miguel e minha esposa Milena, por incentivar, apoiar e compreender quando necessário, e a todos os amigos que fiz nessa jornada que contribuíram para o resultado final.>

Agradecimentos

Gostaríamos de agradecer ao iFood, Rurax, Moblix e BettrAds pelo apoio financeiro e pela disponibilidade de suas equipes e infraestrutura na avaliação de componentes aqui presentes.

Publicações

Publicações realizadas em eventos durante o desenvolvimento desta tese:

1. Marlesson R. O. Santana et al. "***MARS-Gym: Offline Reinforcement Learning for Recommender Systems in Marketplaces***". Em: Offline Reinforcement Learning Workshop at Neural Information Processing Systems (NeurIPS). Spotlight Paper. Virtual Event, Canada: 34th Conference on Neural Information Processing Systems (NeurIPS 2020), 2020. URL:https://offline-rl-neurips.github.io/program/offrl_21.html [135].
2. Marlesson R. O. Santana et al. "***MARS-gym: A Gym Framework to Model, Train, and Evaluate Recommender Systems for Marketplaces***". Em: Workshop on Advanced Neural Algorithms and Theories for Recommender Systems (NeuRec). Virtual Event, Italy: 20th Industrial Conference on Data Mining, 2020 [134].
3. Marlesson R. O. Santana et al. "***Contextual Meta-Bandit for Recommender Systems Selection***". Em: Fourteenth ACM Conference on Recommender Systems (RecSys). RecSys '20. Virtual Event, Brazil: Association for Computing Machinery, 2020, pp. 444–449. ISBN: 9781450375832 [133].
4. Marlesson R. O. Santana e Anderson Soares. "***Hybrid Model with Time Modeling for Sequential Recommender Systems***". Em: ACM WSDM Workshop on Web Tourism (WSDM WebTour'21) (2021) [132].
5. Luana G. B. Martins, Marlesson R. O. Santana et al. "***Achieving Fairness in Personalized Recommendations through Reinforcement Learning***". Submetido a RecSys '24 (2024).
6. Marlesson R. O. Santana, Anderson Soares, Luana G. B. Martins, Anderson Soares et al. "***Framework for Recommender Systems based on Neural Contextual bandits with Fairness-Constrained***". Submetido a RecSys '24 (2024).

Resumo

Santana, Marlesson. **Framework para Sistemas de Recomendação Baseados em *Neural Contextual Bandits* com Restrição de Justiça**. Goiânia, 2024. 103p. Tese de Doutorado Relatório de Graduação. Instituto de Informática, Universidade Federal de Goiás.

O advento dos negócios digitais como *marketplaces*, em que uma empresa intermedeia uma transação comercial entre diferentes atores, apresenta desafios aos sistemas de recomendação por se tratar de um cenário *multistakeholders*. Nesse cenário, a recomendação deve atender a objetivos conflitantes entre as partes, como relevância *versus* exposição, por exemplo. Modelos estado da arte que tratam o problema de forma supervisionada, não apenas assumem que a recomendação é um problema estacionário, mas também são centradas no usuário, o que leva à degradação do sistema em longo prazo. Esta tese foca em modelar o sistema de recomendação como um problema de aprendizado por reforço, por um processo markoviano de tomada de decisão com incerteza onde seja possível modelar os diferentes interesses dos *stakeholders* em um ambiente com restrições de justiça. Os principais desafios estão na necessidade de interações reais entre os *stakeholders* e o sistema de recomendação em um ciclo de eventos contínuo que possibilite o cenário para o aprendizado *online*. Para o desenvolvimento deste trabalho, apresentamos uma proposta de modelo, baseado em *Neural Contextual Bandits* com restrição de justiça para cenários *multistakeholders*. Como resultados, apresentamos a construção do MARS-Gym, um framework para modelagem, treinamento e avaliação de sistemas de recomendação baseados em aprendizado por reforço, e o desenvolvimento de diferentes políticas de recomendação com controle de justiça adaptáveis aos modelos de *Neural Contextual Bandits*, o que levou a um aumento nas métricas de justiça para todos os cenários apresentados enquanto controla a redução nas métricas de relevância.

Palavras–chave

recomendação *multistakeholder*, justiça na recomendação, aprendizado por reforço

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1

Introdução

O advento dos negócios digitais, principalmente aqueles baseados em *e-commerce*, demanda uma nova categoria de plataforma tecnológica que atinge praticamente todos os negócios. Essa nova categoria de plataforma pode ser resumida ao conceito de *marketplaces* que constituem um sistema colaborativo de vendas mediado por uma empresa em que diferentes atores interagem comercialmente, tais como lojistas, clientes, fornecedores, transportadores, dentre outros. O grande benefício do *marketplace* refere-se à centralização das operações de compra em um ambiente seguro. Por meio deste, é possível oferecer maior catálogo ao cliente final e maior exposição dos fornecedores a esse mercado, centralizando a logística da transação em um único ponto ou facilitando o pagamento [101, 138, 90].

Um componente fundamental em um *marketplace* é o sistema de recomendação [110, 145] (RecSys). Considerando o fato de que a quantidade de produtos e usuários, em um *marketplace*, são relativamente maiores que outros modelos de negócio, sistemas de recomendação eficientes são uma componente tecnológica essencial para esse tipo de negócio. Os sistemas de recomendação conectam produtos com potenciais usuários de forma a personalizar a experiência por meio da simplificação da jornada de compra. A Amazon.com, um dos maiores *marketplaces* do mundo, relata a importância do algoritmo de recomendação para o crescimento da plataforma [61], e, segundo relatório da McKinsey [99], cerca de 35% das vendas realizadas na plataforma originam-se da assertividade do algoritmo de recomendação.

Os sistemas de recomendação têm sido pesquisados e aplicados em serviços *online* de diferentes domínios além do *e-commerce* [88], por exemplo, serviços de *streaming* de músicas e vídeos como *Spotify*, *Pandora*, *Last.fm*, *YouTube* e *Netflix* [29, 159, 42, 173], redes sociais como *Facebook*, *Twitter* e *TikTok* [15, 95], plataformas de busca de empregos como *LinkedIn* e *Xing* [80, 8], pesquisa acadêmica como o *Mendeley* [86] e no setor do turismo com plataformas de busca de passagens e hospedagens como *Booking.com*, *Airbnb* e *Trivago* [56, 81, 57, 59].

Enquanto os sistemas de recomendação tradicionais se concentram, especificamente, em aumentar a satisfação do usuário fornecendo conteúdo relevante, os *market-*

places e plataformas de economia compartilhada enfrentam um problema particular em virtude de constituírem um ambiente *multistakeholders*, em que a recomendação deve atender a objetivos conflitantes entre as partes, como relevância *versus* exposição, diversidade *versus* relevância, adaptabilidade *versus* satisfação do usuário, entre outras variáveis de interesse [30, 3, 7, 106]. Os algoritmos devem estar preparados para um cenário com maior estocasticidade em ambientes *multistakeholders* ao mesmo tempo em que devem ser tolerantes a vieses como o de popularidade e injustiças em grupos específicos. Por fim, os algoritmos não devem discriminar indivíduos ou grupos, dando-lhes oportunidades iguais perante ao sistema sob pena de prejudicar a experiência de uma das partes e degradar o desempenho do sistema [111, 120].

Há diversos algoritmos e abordagens supervisionadas que podem ser utilizadas para recomendação personalizada, as mais populares são a filtragem baseada em conteúdo e a filtragem colaborativa [12, 60]. Ambas utilizam o histórico de interações dos usuários para prever preferências futuras. No entanto, este tipo de abordagem assume que a recomendação é um problema estacionário. Outra limitação é que ignora-se o efeito do *feedback loop* quando a própria existência do modelo induz diferentes vieses que afetam os dados obtidos. Esses efeitos são particularmente potencializados em ambientes *multistakeholders* em virtude da maior complexidade nas interações.

Em trabalhos mais recentes [170, 36, 171, 165], visando abordar as limitações apresentadas, o problema da recomendação tem sido modelado como um problema de aprendizado por reforço como um processo Markoviano, também conhecido como aprendizado *online* ou por ambiente em que o agente está constantemente explorando ações e coletando *feedbacks* em um processo de aprendizagem contínua. Segundo Xin et al. [165], as abordagens supervisionadas falham em modelar adequadamente esse processo de interação sequencial, sendo o aprendizado por reforço uma abordagem promissora em virtude de apresentar alto poder de adaptabilidade em cenários complexos com estocasticidade e não estacionários.

A característica mais importante que distingue as duas formas de aprendizado é que a abordagem por reforço utiliza as informações coletadas para avaliar as ações tomadas em um processo contínuo, utilizando o *feedback* de forma avaliativa, enquanto o supervisionado instrui dando ações “corretas” para o aprendizado com base pressupostos de que os dados de treinamento são representativos o suficiente para generalizar o problema, utilizando o *feedback* de forma instrutiva no caso [149]. Um subgrupo dos algoritmos de reforço que se destacam na modelagem de algoritmos de recomendação são os *Multi-Armed Bandit*, que embora não seja uma modelagem de aprendizado por reforço completa por não considerar *feedbacks* atrasados, fornecerem uma modelagem simples para o processo de tomada de decisão com incerteza, ao treinar agentes para aprender a escolher itens por meio da exploração e intensificação na escolha de ações em busca de

maximizar a recompensa cumulativa no longo prazo.

Recentemente, as *Deep RecSys* [170, 49, 48], sistemas de recomendação baseado em arquiteturas de *Deep Learning*, estão avançando no estado-da-arte na área de RecSys ao resolver problemas específicos do domínio. Segundo Zhang et al. [170], os métodos que utilizam arquiteturas de *Deep Learning* resolvem diversos problemas encontrados nos métodos clássicos, como: (1) fácil extração de características em diferentes fontes de dados; (2) facilidade em processar e extrair padrões em abundância de dados; (3) modelagem sequencial e dinâmica por meio de redes recorrentes; (4) diversidade e flexibilidade de arquiteturas para modelos híbridos e multiobjetivos. Zhang et al. [170] também ressaltam que tanto *Deep Learning* quanto RecSys são tópicos de pesquisa em andamento nas últimas décadas com muito potencial para inovação.

As abordagens que utilizam aprendizado por reforço e as *Deep RecSys* podem ser combinadas com o objetivo de usar o melhor das duas áreas, as *Deep Reinforcement Learning* usam uma estratégia de aprendizado contínuo, com alta adaptabilidade para cenários complexos e estocásticos mantendo a flexibilidade das arquiteturas híbridas. As *Deep Reinforcement Learning*, com foco em recomendação [171], abriram espaço pouco explorado do ponto de vista de modelagem de solução, e são ideais para o ambiente *multistakeholders* em que há uma maior necessidade em relação à adaptabilidade do sistema assim como à otimização multiobjetivo para atender interesses conflitantes.

Por ser um campo em expansão, modelos de recomendação baseados em aprendizado por reforço que lidam com ambientes *multistakeholders*, em especial com modelagens que garantam uma definição de justiça para o ambiente, ainda são novidades e devido à complexidade de serem avaliados em ambiente de produção acabam por não serem a primeira opção de solução. Pode-se concluir que, no contexto de pesquisa, ainda existem diversas oportunidades no desenvolvimento de soluções baseadas em *Deep Learning* para recomendação. Embora, no cenário da indústria a predominância sejam dos modelos supervisionados clássicos, há uma real demanda em abordagens mais próximas das necessidades das plataformas *multistakeholders*.

Conforme elucidado por Bernardi, Batra e Bruscantini [23], os sistemas de recomendação são sistemas complexos, que, geralmente são compostos de algoritmos estatísticos avançados, baseados em enormes conjuntos de dados e sistemas de *software* distribuídos em grande escala que precisam operar em um regime de baixa latência. Além disso, destaca-se a necessidade do modelo interagir com o usuário em um ambiente próximo do real, o que torna o desenvolvimento de novas abordagens inviável em muitos casos, principalmente em cenários de aprendizado *online* que necessitam realizar exploração, pois o custo de uma recomendação errada pode ser impeditivo à utilização do modelo. Desse modo, uma barreira para o aperfeiçoamento de sistemas de recomendação pode, em parte, ser atribuída a uma possível falta de colaboração entre a indústria e a

academia pois, o ambiente de interações dinâmicas apresentado na indústria muitas vezes é simplificado a um ambiente estacionário na academia devido a limitações do próprio ambiente de pesquisa. Recentemente, abordagens utilizando ambientes de simulação para criação de modelos pré-treinados e transferência de conhecimento em diferentes contextos tem surgido como solução aos problemas apontados. A indústria e a academia carecem de ferramentas que possam tornar esse processo de validação de novos modelos menos custosos em produção, mais próximo do ambiente dinâmico e com um menor tempo de validação.

1.1 Proposta de trabalho

Os modelos estado-da-arte atuais apresentam pouca adaptabilidade a cenários não estacionários além de considerarem apenas a ótica do usuário na recomendação. Esses modelos falham em modelar corretamente todas as características presentes no cenário proposto de ambientes dinâmicos e *multistakeholders*. Nesse contexto, este trabalho objetiva a criação de um framework para sistemas de recomendação com aprendizado *online* e contínuo. Parte-se da hipótese de que esse tipo de problema geralmente não pode ser modelado e não apresentará resultados competitivos em situações reais com base no paradigma de aprendizado supervisionado tradicional. A proposta apresentada sugere que esse tipo de problema deve ser formulado com base no aprendizado não supervisionado e com modelagem que permita as restrições de justiça para ambientes *multistakeholders*. Como forma de implementar esse tipo de proposta, este trabalho faz uso de arquiteturas de aprendizado profundo (DL) e aprendizado por reforço para interação com o ambiente do sistema de recomendação.

As características da proposta e do cenário apresentado demandam interações reais entre os *stakeholders* e o sistema de recomendação em um ciclo de eventos contínuo que possibilite o cenário para o aprendizado *online*. Torna-se necessário considerar ambientes reais acessados por meio de parcerias com empresas privadas, cenários de *marketplaces* com interações em grande escala que podem prover a dinâmica necessária para o desenvolvimento dessa pesquisa. Para cobrir essa lacuna, do aprendizado e avaliação *online*, e para minimizar os riscos de utilização de modelos subótimos em produção, e aumentar a velocidade no desenvolvimento de melhorias, nossa proposta inclui o desenvolvimento de um ambiente de simulação de sistema de recomendação voltado para o cenário de *marketplace* real, onde, será possível modelar, treinar e avaliar os algorítimos propostos.

Além disso, ao analisar os trabalhos de recomendação, percebe-se que há grande dificuldade em modelar múltiplos objetivos ou restrições nas recomendações, o que, distancia os modelos estado-da-arte do cenário *multistackholder* real. Iremos abordar

essas limitações ao utilizar arquiteturas próprias de *Deep Learning* para modelar a otimização multiobjetiva e as características que definem o conceito de justiça seguindo as tendências dos trabalhos [97, 111, 105, 166].

Por fim, como a definição de justiça em recomendação é um tema abrangente, e está ligado às necessidades e contexto do negócio, desenvolveremos métricas objetivas de modo a mensurar o nível de justiça na plataforma em diferentes perspectivas além do usuário. O estudo de caso proposto como ambiente *multistakeholder*, explorado nesta tese, será o cenário de *marketplace*, em que modelaremos os interesses dos fornecedores e dos usuários do sistema.

1.2 Organização desta tese

Estruturou-se esta tese em 6 capítulos. No capítulo (2), apresentam-se os conceitos básicos de sistemas de recomendação com foco em abordagens *multistakeholders*; no capítulo (3), esclarecem-se alguns conceitos de aprendizado por ambiente e *Deep Learning* para RecSys, em que, se expõem diferentes abordagens de recomendação e discute-se como as redes neurais profundas podem ser utilizadas no controle de justiça, em que, algumas das pesquisas apresentadas constituem o estado-da-arte na área; no capítulo (4), identifica-se a metodologia utilizada e detalham-se as propostas para o desenvolvimento deste trabalho; no capítulo (5), apresentam-se os resultados. Por fim, no capítulo (6), expõem-se algumas conclusões deste trabalho.

CAPÍTULO 2

Sistemas de Recomendação *Multistakeholder*

Neste capítulo, apresentam-se alguns aspectos gerais de sistemas de recomendação e as principais particularidades encontradas em ambientes *multistakeholders*. Na seção (2.1), introduzem-se os sistemas de recomendação como um problema de aprendizado de máquina e os principais métodos de avaliação; na seção (2.3), explana-se sobre as particularidades de um ambiente *multistakeholder* e como os sistemas de recomendação têm um papel fundamental para a qualidade da plataforma. Explora-se também as áreas de pesquisa em RecSys relacionadas ao tema e expõem-se as métricas objetivas e subjetivas para avaliação de sistemas de recomendação *multistakeholder*.

2.1 Introdução a sistemas de recomendação

Os sistemas de recomendação são usados como uma tecnologia de filtragem de informações personalizadas que visa prever e identificar um conjunto de itens de interesse para usuários de acordo com suas necessidades e preferências pessoais [12, 139, 50]. Segundo Falk [50], em uma definição mais direta, um sistema de recomendação calcula e fornece conteúdo relevante para o usuário com base no conhecimento do usuário, conteúdo e interações entre o usuário e o conteúdo. Para Aggarwal et al. [12], um sistema de recomendação consiste em uma ferramenta do comerciante para aumentar as vendas de produtos ao recomendar itens cuidadosamente selecionados para os usuários, o que eleva o volume de vendas e os lucros.

Como formulação de problema há duas principais metodologias que são a base dos algoritmos de sistemas de recomendação, a abordagem como *problema de previsão* e o *problema de ranqueamento*. A primeira busca prevê o valor de escore para uma combinação usuário-item, geralmente associada a uma preferência explícita que o usuário informou para o item, como uma nota ou um "*like/deslike*"; a segunda abordagem visa à ordenação de uma lista de itens, não se preocupando com o valor do escore, apenas em como determinar uma top-k lista de itens relevantes para um determinado usuário. A primeira formulação é mais generalista. A segunda formulação pode ser derivada da

primeira por meio de várias combinações de usuário-item para ranqueá-las em uma lista [12].

Há diversas categorias de métodos que abordam as diferentes modelagens do problema, dentre as mais conhecidas estão a filtragem colaborativa e a filtragem baseada em conteúdo [60]. Na filtragem colaborativa, os algoritmos tentam prever o grau de interesse de um indivíduo em determinados produtos com base em correlações entre as avaliações feitas por estes e as avaliações fornecidas por outros clientes. A hipótese assumida nesse tipo de técnica refere-se ao fato de que pessoas que avaliaram um grande conjunto de produtos de maneira semelhante, pelo menos, num futuro próximo, devem continuar avaliando de maneira semelhante novos produtos. Nos algoritmos baseados em filtragem por conteúdo as implementações são utilizadas para induzir um perfil das preferências de um usuário com base em exemplos, considerando-se uma descrição das características dos conteúdos [155].

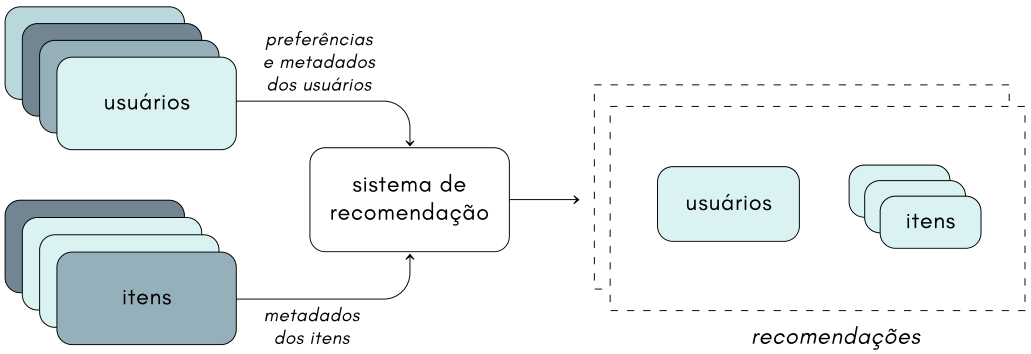


Figura 2.1: Processo geral de recomendação

Cada categoria de algoritmo apresenta prós e contras, e as soluções mais completas tendem a combinar as duas metodologias em algoritmos híbridos. Mas, de forma geral, conforme a Fig. 2.1, os sistemas de recomendação precisam conter pelo menos os seguintes itens para serem criados:

- **informações de preferências dos usuários**: as preferências podem ser explícitas ou implícitas. As explícitas referem-se ao momento em que o usuário indica, espontaneamente, o que lhe é importante e/ou o grau dessa importância, um exemplo seria o usuário adicionar o conteúdo aos favoritos ou dar uma nota qualitativa. As preferências implícitas são coletadas com base no comportamento do usuário no sistema. Essas informações podem indicar as preferências quando não é possível obtê-las de forma explícita, um exemplo poderia ser as matérias visualizadas em um site de notícias (*pageviews*), tempo de leitura, porcentagem do vídeo assistido, termos de buscas etc. A decidir a respeito de qual informação coletar como preferência do usuário, dependerá do próprio sistema e do domínio da recomendação.

- **Metadados dos usuários e itens**: tanto os usuários quanto os itens a serem recomendados têm informações que o caracterizam como conteúdo, no caso do usuário, compreendem informações pessoais (idade, gênero, localidade etc.) ou dados de cadastro com preferências já informadas. Para os itens, depende do conteúdo a ser recomendado, pois, cada domínio (livros, filmes, música, produtos etc.) tem sua particularidade sobre os atributos que caracterizam os itens. Na prática, a tarefa de escolher os atributos certos para caracterizar o conteúdo não é nada trivial. A escolha indevida pode levar o sistema a recomendar conteúdo não relevante para o usuário.
- **Um método para determinar se/ou quanto um item é relevante para um usuário**: metodologia, modelo ou algoritmo a ser executado que, dadas as informações de preferências e/ou metadados dos itens, consiga predizer um escore de preferência usuário-item ou consiga ordenar uma lista de itens relevantes para o usuário.

2.1.1 Filtragem colaborativa

O Princípio da Filtragem Colaborativa é utilizar a informação das interações que ocorrem entre os usuários e os conteúdos para que, de forma coletiva, essa informação seja útil para inferir as preferências dos indivíduos [155]. Segundo Adomavicius e Tuzhilin [10], sistemas de recomendação baseados em filtragem colaborativa estimam a função utilidade $s(i,u)$ de um item i para um usuário u baseado na função utilidade $s(i,u_j)$ para o mesmo item i e para os usuários $u_j \subset U$ que são similares ao usuário u. Geralmente o conjunto UxI, que representa a relação dos usuários e itens é representada por uma matriz de interação como na Fig. 2.2.

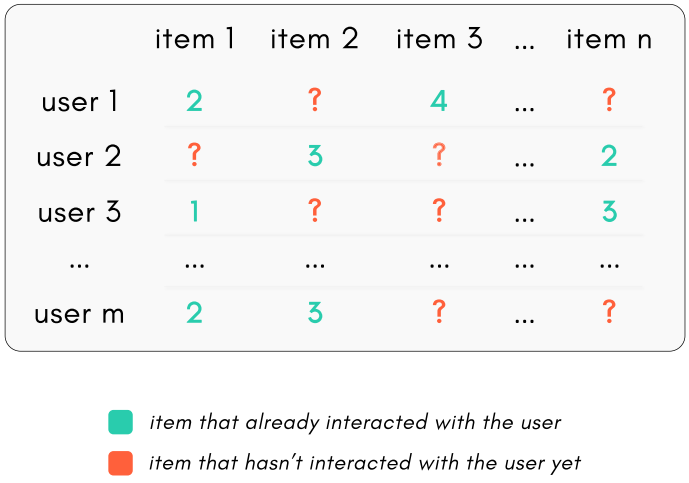


Figura 2.2: Matriz de interação. Cada registro é o valor associado à interação do usuário com o conteúdo

Há diferentes abordagens que podem ser utilizadas para estimar a função utili-

dade de uma filtragem colaborativa [60, 155]. As mais comuns baseiam-se em vizinhança, em que a ideia é usar a similaridade usuário-usuário ou item-item para fazer recomendações com base na matriz de interação; e as baseadas em modelo, em que se espera criar um estimador da função utilidade $s(i,u)$ utilizando técnicas de fatoração de matrizes ou aprendizado de máquina.

Ressalta-se que o método de filtragem colaborativa é de fácil aplicação em diferentes domínios em virtude de usar, apenas, a informação da matriz de interação. Essa abordagem enfrenta alguns problemas com a esparsidade dos dados e como tratar novos usuários e itens. Segundo [136], mesmo em cenários com usuários altamente ativos, eles podem ter interagido com menos de 1% dos itens. Outro ponto negativo refere-se aos novos usuários e itens, pois não haverá dados suficientes de interação. Essa situação é referida como problema de inicialização a frio (*cold start*) e pode prejudicar o desempenho do sistema.

Método de vizinhança baseado em usuário

A filtragem colaborativa de vizinhança, baseada em usuário parte da ideia de que usuários similares avaliam, similarmente, os mesmos itens. Metaforicamente, pode-se exemplificá-la em relação ao pedido de uma recomendação de algo para alguém que tenha os mesmos gostos [12, 104].

Para fazer uma previsão para o usuário u, em um determinado item i, obtém-se uma média ponderada por um coeficiente de similaridade entre os usuários que tenham avaliações naquele item segundo a seguinte fórmula [12]:

$$s(i,u) = \bar{r_u} + \frac{\sum_{s=1}^{k} w_{u,s}(r_{s,i} - \bar{r_s})}{\sum_{s=1}^{k} w_{u,s}} \tag{2-1}$$

Em que, $\bar{r_u}$ e $\bar{r_i}$ constituem as médias das notas dadas pelos usuários u e s, $r_{s,i}$ é a nota dada pelo usuário s ao item i, $w_{u,s}$ é a similaridade entre os usuários u e s, e k é o número de vizinhos considerados na equação.

Medidas de similaridade, baseadas em correlação podem ser usadas para calcular a similaridade entre dois usuários u e s. A Correlação de Pearson mede a extensão em que duas variáveis se relacionam linearmente uma com a outra [126]. As medidas de correlação de Pearson para usuários são fornecidas como:

$$w_{u,s} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r_u})(r_{s,i} - \bar{r_s})}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r_u})^2}\sqrt{\sum_{i \in I}(r_{s,i} - \bar{r_s})^2}} \tag{2-2}$$

Método baseado em modelo

A filtragem colaborativa, baseada em modelo parte do princípio de que é possível criar um estimador capaz de predizer a função utilidade. Nessa categoria, os algoritmos fazem aproximação probabilística e supervisionam o processo calculando o valor esperado de um escore dado os valores conhecidos na matriz de interação [60, 10].

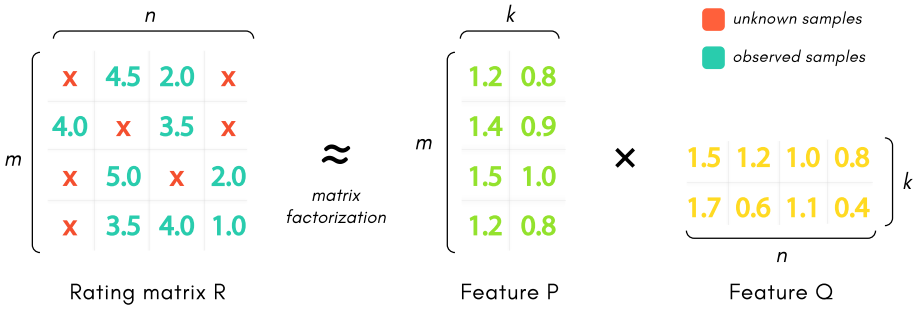


Figura 2.3: fatoração de matrizes

Os modelos mais utilizados nessa categoria são os modelos baseados em fatoração ou decomposição de matrizes, em que são utilizados técnicas como SVD [119], ALS [68] e Redes Neurais [63], para aprender os *fatores latentes* dos usuários e dos itens em um espaço de vetorial de dimensionalidade k. As interações usuário-item são modeladas como produtos internos nesse espaço vetorial. Dessa forma, cada item i está associado a um vetor $q_i \in R^k$, e cada usuário u está associado a um vetor $p_u \in R^k$ de tal modo que o produto vetorial entre os dois vetores capture a interação entre o usuário u e o item i [27]. Nesse sentido, uma estimativa de score é dada por:

$$s(i,u) = q_i^T p_u \tag{2-3}$$

Uma forma de aprender os fatores latentes (usuários e itens) é minimizar *Mean Squared Error* (MSE), o que ocorre por meio da Eq. 2-4, em relação ao conjunto de dados conhecido.

$$min, \sum_{(u,i)\in k} (r_{u,i} - q_i^T p_u)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \tag{2-4}$$

Em que, a constante λ controla o nível de regularização e k é o conjunto de pares (u,i) da matriz de interação, em que, o valor de $r_{u,i}$ é conhecido.

2.1.2 Filtragem baseada em conteúdo

A filtragem baseada em conteúdo considera características do item para recomendar outros itens semelhantes aos que o usuário gosta com base em suas ações anteriores ou *feedbacks* explícitos. Segundo Falk [50], essa abordagem pode ser mais complexa

do que a filtragem colaborativa em virtude de se tratar de extração de conhecimento do conteúdo, e de extrações de definições precisas de imagem, texto, áudio, ou outro tipo de dado bruto, o que, nem sempre é uma tarefa trivial.

O processo de recomendação, baseado em conteúdo é executado em três etapas, cada uma das quais pode ser tratada por um componente específico [14, 50, 12], são elas:

- **Analisador de conteúdo** – processo cujo objetivo é extrair um vetor de características para representação do item em um espaço de informação que possa ser usado pelos outros processos. Geralmente, essa é a etapa mais complexa, pois, a depender das informações e metadados disponíveis dos itens alguma etapa de pré-processamento se torna necessária para extrair informações relevantes de forma estruturada. Por exemplo, descrições de texto e imagens de produtos são naturalmente dados não estruturados e necessitam de uma etapa não trivial para serem transformados em vetores de características. Abordagens mais recentes utilizam de redes neurais como extratores de características em um processo de geração de *embeddings* [33]. Estas, por sua vez, criam vetores densos de k dimensões que representam os itens em um espaço vetorial.
- **Perfil do usuário** – processo destinado a criar um perfil do usuário com base nas informações de interação e metadados do usuário disponível na plataforma. Esse processo pode ser simples como uma lista dos itens consumidos ou usar de técnicas de aprendizado de máquina para modelar o perfil do usuário por meio de *feedbacks*.
- **Filtragem** – processo de filtragem ou ranqueamento dos itens relevantes para o usuário dado o perfil aprendido e as características dos itens extraídos na etapa anterior. O resultado dessa etapa pode consistir em uma relevância binária ou contínua dos itens para com o usuário. Quando o perfil do usuário é construído utilizando uma lista de itens, o processo de recomendação se resume a buscar itens similares que o usuário ainda não interagiu.

Assim como no método de filtragem colaborativa, a similaridade dos itens podem ser calculadas por meio da correlação de Pearson descrita na Eq. 2-2 ou similaridade de cosseno dado pela Eq. 2-5, em que i e j são os vetores de características extraídos da etapa de “analisador de conteúdo”.

$$w_{i,j} = cos(\bar{i}, \bar{j}) = \frac{\bar{i} * \bar{j}}{||\bar{i}|| * ||\bar{j}||} \tag{2-5}$$

Em comparação com os métodos de filtragem colaborativa, a filtragem baseada em conteúdo apresenta vantagens e desvantagens. Segundo Amatriain et al. e Falk [14, 50], é fácil adicionar novos itens ao sistema, pois é necessária, apenas, a extração das características para o cálculo de similaridade e não a obtenção de avaliações prévias como ocorre em relação à filtragem colaborativa. Além disso, segundo os autores, geralmente,

as recomendações são fáceis de explicar e não sofrem muito com vieses de popularidade dado que esta não é uma informação relevante para o método. Como ponto negativo, Aggarwal et al. [12] ressaltam que as recomendações oferecidas por essa categoria de método tendem a ser menos diversificadas, o que insere o usuário em uma certa "bolha de conteúdo", e pode, inclusive, ser prejudicial em longo prazo.

2.2 Avaliação de sistemas de recomendação

Uma parte fundamental para a construção dos sistemas de recomendação consiste no processo de avaliação de novos modelos, em que, o objetivo é avaliar e comparar a qualidade das recomendações geradas por diferentes modelos e parametrizações para que seja possível selecionar aquela com o melhor desempenho para um determinado cenário. Segundo Aggarwal et al., Shani e Gunawardana [12, 139], há três principais categorias de avaliação de sistemas de recomendação:

- Avaliação *offline*, com conjunto de dados históricos;
- Avaliação *online*, com interações reais de usuários;
- Estudo de usuário.

2.2.1 Avaliação *offline*

A avaliação *offline* é a metodologia mais mencionada na literatura para avaliação de sistemas de recomendação. Ela é realizada de forma similar a problemas de classificação/regressão em aprendizado de máquina, em que se utiliza uma base de dados pré-coletados de usuários, itens e interações ou avaliações realizadas pelos usuários para avaliar as recomendações sugeridas pelo modelo [2].

Abdollahpouri [2] ressalta que a principal vantagem dessa abordagem é sua eficiência em testar diversos algoritmos com baixo custo, visto que não é necessário realizar incursões *online* em um ambiente de produção. Para Aggarwal et al. [12], a principal desvantagem é que elas não medem a propensão real do usuário de reagir ao sistema de recomendação no futuro, sendo, apenas, um retrato do momento em que os dados foram coletados. Nessa perspectiva, a desvantagem pode ser ainda mais impactante em um ambiente dinâmico e não estacionário como um marketplace, em que as interações entre os *stakeholders* e o sistema de recomendação impactam na própria distribuição dos itens a serem recomendados no futuro, levando assim a uma avaliação enviesada e distante do cenário real.

Há diversas métricas que podem ser utilizadas para a qualificação do resultado das recomendações utilizando dados históricos, e a melhor escolha depende de como o problema foi formulado. Se a abordagem for similar ao problema de previsão, em que

é possível comparar o escore dado ao modelo com a preferência do usuário em relação aos dados coletados, a avaliação da recomendação pode ser realizada de forma similar a um problema de regressão e utilizar métricas de acurácias como RMSE e MAE [32] para chegar em uma estimativa de erro médio. Por outro lado, em uma formulação como problema de ranqueamento, em que a avaliação *offline* pretende estimar a qualidade de uma lista de recomendação ranqueada, as métricas mais comuns são as utilizadas na área de recuperação da informação [102], tais quais o MAP e nDCG [139, 160] apresentados nas Eq. 2-6 e Eq. 2-7. Essas duas métricas mensuram a qualidade de uma lista ranqueada com base na ideia de que os itens importantes, aqueles que o usuário interagiu na base histórica, estejam no começo da lista.

O *Mean Average Precision* (MAP) apresenta a média dos valores de precisão para os itens relevantes de 1 a k, em que k é o tamanho da lista ranqueada, Q é a quantidade de itens relevantes na lista, P_i a precisão do top_k e rel_i é uma função indicadora igual a 1, se o item, relativo à classificação, for um documento relevante; e zero, caso contrário.

$$MAP_k = \frac{\sum_{i=1}^{k}(P_i * rel_i)}{Q} \tag{2-6}$$

O *Normalized Discounted Cumulative Gain* (nDCG) é semelhante ao MAP, exceto que nDCG também funciona para relevância não binária e suaviza o desconto ao longo da lista. A métrica é dada pela Eq 2-7, em que rel_i é a relevância do item, k o tamanho da lista de recomendação e $IDCG_k$ é o DCG ideal.

$$\begin{aligned} nDCG_k &= \frac{DCG_k}{IDCG_k} \\ DCG_k &= \sum_{i=1}^{k} \frac{rel_i}{log_2(i+1)} \end{aligned} \tag{2-7}$$

2.2.2 Avaliação *online*

Em um ambiente não estacionário, os usuários estão interagindo com o sistema de recomendação a todo momento, influenciando e sendo influenciados, resultando em uma alteração da distribuição dos dados coletados ao longo do tempo. Portanto, a avaliação *online* está interessada em medir a mudança no comportamento do usuário ao interagir com diferentes sistemas de recomendação temporalmente. Dessa forma, mecanismos de teste *online* são implementados e, por meio destes, os experimentos são executados em um ambiente real, onde os algoritmos são comparados durante a execução do sistema em ambiente de produção [178, 2, 139].

Enquanto os pesquisadores concentram-se em cenários de avaliação *offline* com base nos dados históricos por uma série de limitações, tais quais terem acesso aos sistemas

online com a escala necessária, os profissionais da indústria valorizam os experimentos *online* em sistemas ativos em virtude de refletirem melhor os objetivos do negócio [122, 91], como aumentar a Taxa de Cliques (CTR) em determinado site, ou o aumento da conversão de clientes, por exemplo. O impacto de tais métricas de negócio somente podem ser calculadas *online* pois dependem de um efeito causal a recomendação apresentada pelo sistema. E, embora as métricas *offline* e *online* possam ser correlacionadas, trabalhos como os de Ji et al., Krauth et al., Garcin et al. [76, 83, 52] demostram que essa relação pode variar a tal ponto de inutilizá-la como referência a depender de diversos aspectos, tais como a taxa de novos usuários e itens no sistema, possíveis *data leakage* no treinamento *offline*, mudança de comportamento no ambiente *online* (*data drift* e *concept drift*), *feedback-loop* entre outros.

A avaliação *online* pode ser realizada por teste A/B [82] entre dois modelos para medir o impacto direto do sistema de recomendação em relação ao usuário final. A ideia é comparar dois algoritmos, a estratégia atual e a que está sendo testada em um ambiente de produção, controlando os grupos impactados pela recomendação de cada modelo. Dessa forma, os usuários são separados em dois grupos, A e B, em que cada grupo tem acesso, apenas, às recomendações de um dos modelos. Ao final de um período, calcula-se a métrica objetivo como o CTR ou a conversão para os dois grupos e define-se a melhor estratégia. Essa avaliação só é viável de ser realizada em um ecossistema tecnológico em pleno funcionamento, uma plataforma com interações reais de usuários, itens e demais *stakeholders*.

Segundo Abdollahpouri, Peska e Vojtas [2, 122], a avaliação *online* de um novo sistema de recomendação traz riscos ao negócio, pois, no caso de o algoritmo não funcionar como esperado, a experiência do usuário é prejudicada. Na prática, antes de conduzir um experimento *online*, uma avaliação *offline* é realizada e, se os resultados forem estáveis e razoáveis, o teste *online* pode ser executado a seguir.

2.2.3 Estudos de usuário

O estudo de usuário constitui uma alternativa à avaliação *online*. Nesse tipo de avaliação, os usuários de teste são ativamente recrutados e solicitados a interagirem com o sistema de recomendação, ou seja, a interação é *online* mas o ambiente de teste é controlado. O estudo pode ir além da avaliação da interação dos usuários com um sistema de recomendação. Outras fontes de avaliação, em que apenas essa metodologia permite, são os formulários de pesquisa e entrevistas dos candidatos para a coleta de informações relevantes ao processo [12, 2].

Embora essa categoria de avaliação possa ser complementar às demais, e trazer aspectos que não podem ser coletados tanto na avaliação *offline* quanto na *online*,

Aggarwal et al. [12] ressaltam que o fato de os usuários estarem ativamente cientes de seu recrutamento para um determinado estudo, provavelmente, afetará suas respostas. Portanto, os resultados das avaliações do usuário não podem ser totalmente confiáveis. Outro ponto é descrito por Abdollahpouri [2], em que, no entanto, tal abordagem tenha custo elevado, pois requer a busca de colaboradores dispostos a poupar seu tempo e a participar do estudo.

2.2.4 Aspectos gerais e desafios

Há outras formas relevantes de qualificar uma lista ranqueada além das focadas em acurácia. Herlocker et al. [64] afirmaram que a alta precisão por si só pode não ser suficiente para fornecer recomendações úteis e envolventes, e que outras propriedades devem ser consideradas simultaneamente. Há três importantes características que as listas recomendadas devem ter, além de uma boa acurácia [78, 108], são elas:

- **novidade** – a novidade de um sistema de recomendação está ligado à capacidade de recomendação de itens aos usuários, dos quais ele não tem conhecimento, ou um item que seja diferente do que o usuário costuma ver. Um sistema de recomendação enviesado para itens populares, geralmente, apresenta pouca novidade aos usuários.
- **cobertura** – a cobertura está na capacidade de o sistema recomendar diferentes itens do catálogo. A cobertura de catálogo é definida pela fração dos itens recomendados para, pelo menos, um usuário, sendo essa uma medida importante para identificar se o sistema está de fato explorando diferentes grupos de itens nas recomendações.
- **diversidade** – a diversidade é referente ao quanto os itens presentes em uma lista de recomendação são semelhantes entre si. De forma geral, espera-se que uma lista de recomendação seja a mais diversa quanto possível para que o usuário tenha escolhas suficientes.

É importante ressaltar que os aspectos de novidade, cobertura e diversidade são, geralmente, medidas contrárias às medidas de acurácia nos testes *offline*. Um balanço entre essas características na lista de recomendação é fundamental para a qualidade do sistema em diferentes aspectos. Outro ponto refere-se ao fato de que, embora as métricas muitas vezes sejam vistas como secundárias, no ambiente de recomendação *multistakeholder*, elas ganham um peso maior por estarem intrinsecamente ligadas aos objetivos dos demais parceiros do negócio. Um sistema que garanta a novidade na recomendação para o usuário está, de certa forma, reduzindo os vieses de popularidade que podem prejudicar provedores menos conhecidos na plataforma, enquanto a garantia de cobertura e diversidade impactam na taxa de exposição dos provedores, que geralmente é um dos principais objetivos desse grupo.

2.3 Recomendação *multistakeholder*

Os sistemas de recomendação são ferramentas que fornecem personalização e filtro do conteúdo em ambientes em que o volume de informação oferecida para o usuário final, é maior que a capacidade de consumo, sendo uma solução agnóstica a diferentes mídias e segmentos, como comércio eletrônico, redes sociais, notícias, *streaming* de vídeos e músicas, entre outros. As pesquisas em sistemas de recomendação estão voltadas, principalmente, para a personalização desse conteúdo voltado ao usuário final, mas, embora os usuários sejam uma das partes interessadas mais importantes em qualquer plataforma de recomendação, eles não são os únicos [6, 2]. Há inúmeros exemplos de ambientes em que há outras partes interessadas nas recomendações, geralmente com objetivos próprios e conflitantes entre-si. Abdollahpouri [2] define como "*stakeholder* da recomendação" qualquer grupo ou indivíduo que pode afetar ou é afetado pela entrega de recomendações aos usuários.

2.3.1 Dinâmica da recomendação

Embora a definição de "*stakeholder* da recomendação" possa incluir diversos grupos, [7, 3, 30, 19] consideram-se três grupos principais, pois, estes interagem diretamente com as recomendações. O primeiro grupo são os consumidores ou usuários finais, são aqueles que recebem as recomendações do sistema e interagem com elas. O segundo grupo compreende os provedores, aqueles que fornecem os itens que serão recomendados na plataforma, e, geralmente, utilizam a plataforma para alcançar o público desejado. O último grupo refere-se à própria plataforma que criou o sistema de recomendação e é um intermediário entre o consumidor e o fornecedor. Na Fig. 2.4 exemplificam-se os *stakeholders* da recomendação em um marketplace como Uber Eats ou iFood, em que, os consumidores são os usuários dos aplicativos que buscam comida, os provedores são os restaurantes na plataforma que oferecem seus serviços e estão em busca de exposição, e a plataforma precisa garantir a carga de trabalho dado aos parceiros que cuidam da logística e manter a qualidade do serviço entre as partes.

A respeito do conceito de *stakeholder*, ainda é possível subdividir cada grupo em sub-grupos de interesse, em que é possível caracterizar o grupo de forma homogênea ou heterogênea a depender dos interesses que se buscam atender com as recomendações [3]. Em um grupo homogêneo, tanto os usuários quanto os provedores recebem o mesmo tratamento de interesse definido no sistema de recomendação, mas há cenários em que, mesmo em relação ao grupo de usuários ou provedores, é importante segmentá-los em diferentes aspectos para obtenção de um sistema justo. Por exemplo, embora os usuários busquem a qualidade nas recomendações, é possível que devido a vieses de coleta, o sistema se comporte diferente para grupos distintos de gênero, localidade, classe social,

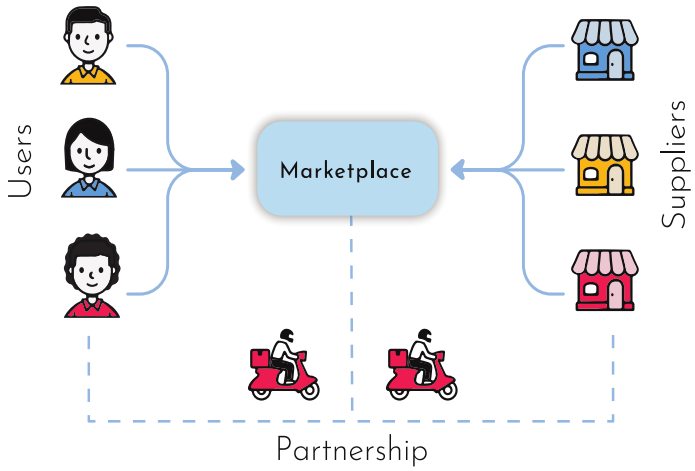


Figura 2.4: *Marketplace* como um sistema *multistakeholder*

ou qualquer outro agrupamento natural ou não [34, 116, 20]; no caso dos provedores que buscam um certo nível de exposição na plataforma, é possível, que a plataforma tenha níveis diferentes de exposição para diferentes grupos, seja devido a vieses de popularidade ou mesmo a uma estratégia de negócio [5, 4]. Controlar e garantir certos níveis de "justiça"intragrupo é uma ferramenta útil para elevar a qualidade geral do sistema de recomendação em cenários *multistakeholders* em longo prazo, por balancear os diferentes interesses.

Devido à necessidade de atender a diferentes grupos de interesse, a pesquisa e o desenvolvimento de sistemas de recomendação devem considerar os objetivos conflitantes entre as partes, desde a definição clara dos objetivos e expectativas de cada *stakeholder* em utilizar o sistema, ao *designer* de modelos específicos para atender a esses objetivos e à criação de métricas que possam avaliar esses modelos em diferentes perspectivas.

2.3.2 Áreas de pesquisa relacionadas

A recomendação *multistakeholder* pode ser entendida como uma generalização de diversos conceitos e esforços recentes para adequar as considerações envolvidas nos projetos e na avaliação de sistemas de recomendação além do foco na acurácia para o usuário final. Diferentes áreas de pesquisa podem se articular a essa temática, tais como: recomendação em ambiente de economia compartilhada, como os *marketplaces* [124, 138]; problemas multiobjetivos, com recomendação e otimização em diferentes métricas como diversidade, cobertura, novidade etc. [111, 97, 158]; personalização de *match* ou recomendação recíproca [164, 118, 113]; e justiça a respeito da recomendação para grupos distintos [30, 89, 19].

Economia compartilhada

A economia compartilhada é um termo abrangente para uma grande variedade de modelos econômicos organizacionais e um fenômeno em que novos modelos de negócios estão surgindo. São enquadrados como mediados por tecnologia em que é facilitado o acesso a bens ou serviços subutilizados e potencialmente reduzindo o consumo líquido. Plataformas de tecnologia que conectam diferentes grupos com interesses complementares são o centro dessa nova economia, em que é possível alugar (Airbnb, moObie), negociar (OLX, Enjoei, Mercado Livre), trocar (Bliive, skoob), compartilhar (hotmart), dar empréstimo (ulend, Bullla), prover serviços (Uber, GetNinjas) etc [43, 115].

Os sistemas de recomendação *multistakeholders* têm papel fundamental a respeito do sucesso das plataformas de economia compartilhada por fazer parte da estratégia para atrair e reter participantes de todos os lados do negócio. Para manter os níveis de qualidade das recomendações nesse cenário, é necessário que os desenvolvedores modelem e avaliem a utilidade do sistema para todos os interessados.

Recomendação multiobjetiva

Há um recente esforço relativo à incorporação de objetivos adicionais à modelagem e à avaliação de sistemas de recomendação além da precisão das recomendações. Esses objetivos, independentemente da quantidade de grupos de interesse, são potencializados no cenário *multistakeholder*. Conceitos de diversidade, novidade, cobertura e tratamento da calda longa [78] se misturam às necessidades e interesses de diferentes grupos em ambientes *multistakeholders*, como exposição e restrições de justiça na plataforma. Há um crescente interesse nos trabalhos que combinam os vários objetivos por meio de uso de técnicas de otimização de restrição, programação linear e aprendizado multi-task [71, 98, 35], mas as mais recentes e promissoras são as de aprendizado por reforço [111, 97, 37] por se adaptarem melhor ao cenário estocástico de uma ambiente dinâmico como os de economia compartilhada.

Recomendação recíproca

A recomendação recíproca examina as considerações em ambientes bilaterais para garantir que as recomendações sejam de qualidade para ambas as partes interessadas. Os exemplos mais clássicos são o do *match* de relacionamentos [123, 176], em que ambas as partes têm o mesmo interesse de "encontrar um parceiro ideal". Dessa forma, a recomendação de *match* tem de balancear o interesse das duas partes para que uma correspondência seja bem-sucedida. A outra é a recomendação de vagas de emprego, em que não é suficiente, apenas, recomendar a melhor vaga para um determinado currículo, pois deve existir uma reciprocidade do melhor candidato para uma vaga para que as

chances de sucesso na contratação sejam elevadas, atendendo assim, às expectativas de ambas as partes.

Essa área pode ser vista como uma especialização da recomendação *multistakeholders* para cenários de, apenas, dois interessados, que podem (ou não) ter os mesmos objetivos com relação à utilização do sistema, mas que compartilham da mesma importância para a plataforma.

2.3.3 Justiça em recomendação

A justiça em aprendizado de máquina é um conceito amplo e tem ganhado destaque nos últimos anos [109, 44, 26, 51]. Em conceito abstrato, significa não discriminar indivíduos ou grupos, dando-lhes oportunidades iguais perante ao sistema. Para aprendizado de máquina, o significado está relacionado ao fato de a performance do modelo ser a mesma entre diferentes grupos [62]. Essa discriminação pode ser de tratamento, quando o sistema usa de atributos sensíveis (como gênero, raça, idade etc) para gerar alguma disparidade no tratamento, ou de impacto, quando há disparidade nos resultados entre diferentes grupos [51, 31].

A definição do conceito de disparidade de tratamento e resultados é algo que depende do contexto de utilização do modelo e da estratégia de negócio adotada. No cenário de recomendação *multistakeholders*, a justiça na plataforma tem sua definição ancorada nos grupos e objetivos de cada *stakeholder*. Por exemplo, em um ambiente de *marketplace*, em que os provedores podem pagar por um plano de destaque na plataforma, a justiça está em garantir os diferentes níveis de exposição conforme o plano pago. Por outro lado, a qualidade das recomendações para os usuários não podem ser prejudicadas, em especial, de forma discriminada por atributos sensíveis.

Dessa forma, podemos usar a definição apresentada em Zafar et al. [168], para mensurar a justiça do sistema do ponto de vista de acurácia nas recomendações entre os grupos, e a definição apresentada por Patro et al. [120] para mensurar a utilidade da lista de recomendação para o usuário e a exposição do fornecedor. Definições similares podem ser encontradas em [143, 111, 30].

Em Zafar et al. [168], consideram-se a noção de justiça em três perspectivas. Primeiramente, na perspectiva de ***tratamento díspar***, que surge quando um sistema de recomendação fornece saídas diferentes para grupos de usuários ou provedores com atributos não sensíveis iguais, ou semelhantes, mas valores diferentes de atributos sensíveis. Matematicamente, para evitá-lo, o sistema precisa garantir a Eq. 2-8, em que z é o atributo sensível na análise, x um vetor de características compartilhado entre os grupos, a o item a ser recomendado e π uma política de recomendação.

$$\pi(a \mid x) = \pi(a \mid x, z). \tag{2-8}$$

A segunda é sobre ***impacto díspar***, que ocorre quando as saídas de decisão beneficiam ou prejudicam desproporcionalmente um grupo de usuários que compartilham um valor de atributo sensível específico. Para evitá-lo, precisamos garantir a Eq. 2-9, em que $\mathcal{Z}$ representa o conjunto com todos os valores possíveis de um atributo sensível z:

$$\pi(a = a_k \mid z_i) = \pi(a = a_k \mid z_j), \forall a_k \in \mathcal{A}, \forall z_i, z_j \in \mathcal{Z}.$$

A terceira definição apresentada por Zafar et al. [168] é a de ***maus tratos díspares***. Esta surge quando as taxas de erros diferem para grupos de usuários com valores diferentes para um determinado atributo sensível. Para calcular essas taxas, é preciso calcular o erro de classificação para diferentes grupos, dado pela Eq. 2-9, em que, $a*$ é a ação verdadeira de cada interação.

$$\pi(a = a^* \mid z_i, a^* = a_k) = \pi(a = a^* \mid z_j, a^* = a_k), \forall a_k \in \mathcal{A}, \forall z_i, z_j \in \mathcal{Z}.$$

Patro et al. [120] apresentam duas outras medidas que complementam a definição de Zafar et al. [168]. Dada uma lista de recomendação R_u para o usuário u, a utilidade dessa lista para o usuário corresponde à soma das pontuações de relevância dos itens presentes em R_u, pois, recomendar os k produtos mais relevantes proporcionará a máxima relevância. A **utilidade da lista** é definida na Eq. 2-9, em que, a definição de relevância de um produto pode ser definida como a probabilidade de que o usuário u goste do item p, a R_u^* é a lista com os *top-k* itens mais relevantes.

$$U(R_u) = \frac{\sum_{p \in R_u} V_u(p)}{\sum_{p \in R_u^*} V_u(p)} \tag{2-9}$$

A **métrica de exposição** dos provedores está relacionada aos interesses desses *stakeholders* em utilizar a plataforma. Quanto maior a exposição, maiores são as chances de bons negócios. Patro et al. [120] definem que a exposição do provedor p é a quantidade total de atenção que p recebe de todos os clientes para os quais p foi recomendado. O autor assume um modelo de atenção uniforme, em que os clientes prestam atenção semelhante a todos os k itens recomendados e expressão à exposição de um provedor como $E_p = \sum_{u \in U} R_u(p)$, onde $R_u(p)$ é 1 se $p \in R_u$, ou 0 caso contrário. Uma sugestão de melhoria dessa métrica é não utilizar uma distribuição uniforme como definida por Patro et al., pois a posição que o item é apresentado na lista influencia na chance dos usuários percebê-lo, o que por definição impacta na métrica de exposição.

2.4 Considerações finais

Nesse capítulo, foram apresentados alguns aspectos gerais de sistemas de recomendação e as principais particularidades em ambientes *multistakeholders*. Foi realizada uma introdução das principais categorias de métodos de recomendação como filtragem colaborativa e baseada em conteúdo, bem como os aspectos gerais da avaliação de modelos de recomendação centrados no usuário.

Por fim, foram expostas as principais particularidades e a dinâmica de recomendação quando existem múltiplos interessado na recomendação, realizada pelo sistema, principalmente quando esses interesses se mostram conflitantes. Foram descritas algumas áreas de pesquisa com temáticas relacionadas à recomendação *multistackholder* e como elas se complementam. Também foi abordado o conceito de justiça na recomendação e como essa área está relacionada com o ambiente *multistackholder*. Por fim, discutiram-se algumas métricas objetivas para avaliar a justiça em diferentes perspectivas.

CAPÍTULO 3

Aprendizado por Ambiente e *Deep Learning*

Neste capítulo, apresentam-se alguns aspectos gerais do aprendizado de máquina por ambiente, a importância dessa metodologia para a recomendação em ambientes *multistakeholders* e as principais técnicas e modelagens de *deep learning* para sistemas de recomendação. Na seção (3.1), introduzem-se o conceito de aprendizado de máquina por ambiente e as principais motivações para utilizá-lo em um ambiente de recomendação *multistakeholders*, bem como algumas técnicas que podem ser utilizadas nesse contexto; Por fim, na seção (3.2), expõem-se as técnicas de *Deep Learning* que podem ser utilizadas na área de sistemas de recomendação e discute-se como abordá-las em um cenário de aprendizado por ambiente.

3.1 Aprendizado por ambiente

Abordagens tradicionais para sistemas de recomendação consideram esse problema em um ambiente de aprendizagem supervisionado em que o modelo é treinado calculando-se a diferença entre a resposta esperada e a resposta obtida pela solução. No entanto, usando essa abordagem, assume-se que a recomendação seja um problema estacionário. Outra limitação está na visão incompleta de interesses dos usuários uma vez que o *feedback* é apenas parcialmente observável pois somente é possível conhecer os eventos em que o usuário ativamente interagiu. Como reforçam Mansoury et al., Huleihel, Pal e Shayevitz, Schmit e Riquelme [103, 69, 137], a própria existência do sistema de recomendação também induz diferentes vieses que afetam os dados obtidos de futuros usuários. Esse efeito é chamado de *feedback loop*, fenômeno potencialmente degradante em modelagens supervisionadas que no longo prazo amplifica comportamentos inesperados como viés de popularidade, viés de seleção e exposição, dentre outros [34].

Os sistemas de recomendação possuem como característica, a alta complexidade, característica não estacionária e de autorreforço por meio do *feedback loop*, ambiente *multistakeholders* e por fim a existência de objetivos conflitantes e um maior dinamismo nas interações. Estas características fazem com que abordagens clássicas sejam limitadas em relação à modelagem correta do cenário real. Primeiramente, elas são projetadas para

maximizar a recompensa imediata a respeito do usuário ao focar no aumento da taxa de cliques e compras, e não consideram recompensas de longo prazo, como a atuação efetiva do usuário ou questões de justiça. Além disso, elas tendem a adotar uma abordagem que enfatiza demasiadamente a popularidade dos itens, negligenciam a exploração de novos itens e apresentam baixa performance em cenários de *cold start* que acontece quando usuários e itens são adicionados com frequência, ou até mesmo, quando a dinâmica do sistema muda rapidamente como é o caso de ambientes *multistakeholders* com restrições de justiça.

Abordagens mais recentes [140, 171, 96, 172, 174, 105] modelam o sistema de recomendação adaptável ao ambiente e com aprendizado *online* por meio do Aprendizado por Reforço (AR). Com o emprego de AR, pode-se otimizar recompensas de longo prazo por meio do equilíbrio entre exploração e intensificação, ou seja, um balanço entre explorar recomendações não ótimas com base em uma política para adquirir conhecimento versus a utilização do conhecimento acumulado com base na exploração para dar boas recomendações em uma estratégia sequencial.

Pode-se formalizar o sistema de recomendação com base nos principais aspectos de sua composição. Chamamos esse processo de política π, que constitui o principal meio de tomada de decisão. Assim, no caso em que um conjunto de dados $\mathcal{D}$ está disponível, a política usada para adquiri-lo é chamada de política de coleção π_c. Para fazer recomendações, a política considera $x \in \mathcal{X} : \mathbb{U} \times \mathcal{C}$, composto pelo usuário atual $u \in \mathbb{U}$ e informação contextual $c \in \mathcal{C}$. Finalmente, a política seleciona uma ação $a \in \mathcal{A}$ de acordo com $\pi_d(a \mid x)$ como uma recomendação ao usuário.

Dessa forma, é possível descrever o processo de recomendação como um Processo de Decisão de Markov (PDM) dado pela tupla $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, T)$, em que, na etapa de tempo $t \in \mathcal{T}$, um recomendador recebe informações contextuais que compõem o estado do ambiente $s_t \in \mathcal{S}$ e seleciona uma recomendação $a_t \in \mathcal{A}$ amostrada com base em uma política π, ou seja, $a \sim \pi(\cdot \mid s_t)$. Como consequência de tal ação, o ambiente emite uma recompensa de *feedback* escalar r, de acordo com $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \{0, 1\}$ que simboliza se a recomendação foi bem sucedida. O ambiente também muda de estado seguindo uma dinâmica de transição $\mathcal{P} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$, que representa a densidade de probabilidade do próximo estado s_{t+1}. Na Fig. 3.1, demonstra a dinâmica de interações entre o agente, que representa o sistema de recomendação, e o ambiente, que interage com as recomendações e sinaliza a recompensa. O objetivo final de um agente de reforço é maximizar a recompensa cumulativa, ou seja, $\max \mathbb{E}[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)]$, em que γ é um fator de desconto para contabilizar recompensas atrasadas [149].

Nesse contexto, o aprendizado por reforço, por ser uma técnica de aprendizado de máquina baseada em tentativa, erro e *feedback*, e, por ter a característica de aprender por meio de interações com o ambiente, se torna uma alternativa interessante para

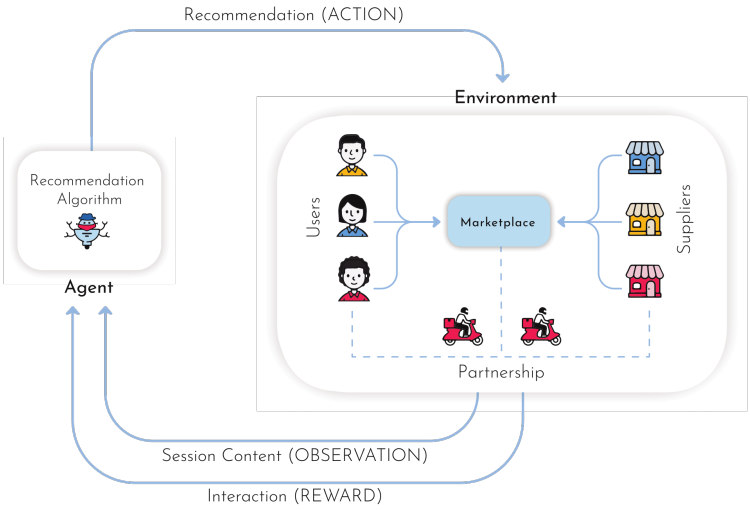


Figura 3.1: A interação entre o agente e o ambiente como um processo de decisão de Markov (PDM).

modelagem de sistemas de recomendação no cenário proposto. Ao tratar o sistema de recomendação como um PDM, os estados passados ao agente são limitados pela *propriedade de Markov* a qual as informações atuais são suficientes para tomar a melhor decisão possível. Portanto, os estados passado e futuro são irrelevantes para o processo de tomada de decisão, embora, alteram o histórico do usuário e o estado do ambiente [149, 140]. Na área de aprendizado por reforço, há diversas abordagens que modelam os sistemas de recomendação como um processo de tomada de decisão sequencial. Entre as abordagens mais conhecidas estão os modelos baseados em *Multi-Armed Bandits* e *Contextual Bandits* que simplificam a formulação apresentada ao considerar as recompensas imediatas, ou seja, não é necessário considerar *feedbacks* atrasados.

3.1.1 Multi-Armed Bandit

Embora a modelagem de *bandits* tenha sido introduzida por William R. Thompson, em 1933 [154], quando esteve interessando em testes médicos e na adaptação do tratamento em caso de certa eficácia de um droga, o nome de *multi-armed bandit* somente ficou conhecido a partir da década de 50 com estudos sobre “aprendizado” realizados por Frederick Mosteller e Robert Bush em que se apresentava uma máquina com “dois braços” e os participantes poderiam escolher puxar o braço esquerdo ou direito da máquina. Cada braço dava uma recompensa aleatória com a distribuição de recompensas desconhecidas para o jogador. O nome faz homenagem às máquinas de *caça-níquel*, metáfora utilizada para denotar um “bandido de um braço” (“bandido” porque eles roubam seu dinheiro) [85, 162, 144].

Multi-Armed Bandits (MAB) se tornaram muito comuns em sistemas de recomendação [91, 94, 141, 157] em virtude de fornecerem uma modelagem simples para o processo de tomada de decisão com incerteza, porém, bem adaptável a diferentes cenários. A configuração usual refere-se aos itens passíveis de recomendação como *arms* e o objetivo é aprender as recompensas esperadas para cada um deles por meio da exploração e intensificação da recomendação desses itens visando aprender qual é a recompensa potencial e, em seguida, explorar a melhor recomendação para maximizar a recompensa acumulada. O objetivo é aprender a escolher itens e realizar ações de recomendação que maximizem a recompensa total (também conhecida como minimizar o arrependimento) ao longo do tempo [94, 156, 157, 84].

A definição de um problema de MAB é dada por uma coleção de distribuições de probabilidade $v = (P_a : a \in A)$ em que A é o conjunto de ações disponíveis. O agente interage com o ambiente sequencialmente por meio de n rodadas. Cada rodada $t \in \{1, ..., n\}$. E, a cada rodada, uma política π é utilizada para escolher uma ação $a_t \in A$ enviada para o ambiente, que, por sua vez, apresenta uma recompensa $r_t \in R$ com base em uma distribuição P_{A_t} inicialmente desconhecida pelo agente. O objetivo é atualizar a política para maximizar a recompensa total dada por $S_n = \sum_{t=1}^{n} r_t$, que é a quantidade aleatória que depende das ações escolhidas pela política e as recompensas amostradas pelo ambiente. Tanto a política de exploração π quanto a recompensa r_t podem ser adaptadas para modelar diferentes cenários em um ambiente de recomendação *multistakeholder*, tais quais dar suporte a modelagem multiobjetiva ou restrições de justiça em relação às escolhas dos itens a serem recomendados.

ϵ-Greedy

Dentre as diversas políticas de exploração consolidadas, o ϵ-Greedy é uma das mais simples de implementar e entender. Basicamente, é definido um valor de *epsilon* qualquer que constitui a probabilidade de exploração aleatória das ações conhecidas, sendo um *threshold* do algorítimo que parametriza o dilema da exploração e intensificação [162]. O algoritmo ϵ-Greedy funciona oscilando, aleatoriamente, entre a escolha puramente aleatória e a escolha que maximiza a recompensa esperada segundo a Eq. 3-1, em que, p_t é uma probabilidade aleatória e $Q_t(a)$ é a recompensa esperada do item a no tempo t.

$$a_t = \begin{cases} \text{argmax} \quad Q_t(A), & se \quad \epsilon > p_t \\ \text{random} \quad a \in A, & se \quad \epsilon \leq p_t \end{cases} \tag{3-1}$$

No contexto de recomendação, a cada rodada t, o recomendador com a política do ϵ-Greedy decide, com base no ϵ definido, se escolhe um item aleatório dentre os disponíveis para recomendar em uma estratégia de exploração ou se escolhe o item com

uma estimativa de recompensa alta em relação à estratégia de intensificação. Em uma modelagem mais simples, a estimativa de recompensa de um item pode ser a recompensa média ao longo das rodadas.

Upper Confidence Bound (UCB)

Por ser uma modelagem simples, o ϵ-Greedy apresenta diversos problemas com relação à exploração realizada de forma aleatória, principalmente quando o espaço de busca é grande como no contexto de sistemas de recomendação. Uma alternativa é a política de exploração UCB, que se baseia no Princípio do Otimismo em Face da Incerteza em que as ações pouco exploradas são escolhidas com maior frequência para que as incertezas com relação às suas recompensas sejam reduzidas [85].

O UCB usa dados observados até uma rodada t para atribuir a cada ação um limite de confiança superior, uma superestimativa da média da recompensa esperada para a ação que reduz ao longo das rodadas. A Eq. 3-2 rege a escolha da ação em que $Q(a)$ consiste na recompensa média dada pela ação ao longo do tempo (ou uma estimativa dela), c é um parâmetro de exploração, quanto maior menos confiança terá nas estimativas, e $\sqrt{(\frac{log(t)}{N_t(a)})}$ é o que modela a incerteza como um limiar superior, quanto mais escolhas forem feitas por meio dessa ação, menor esse limiar.

$$a_t = argmax\left[Q_t(A) + c\sqrt{(\frac{log(t)}{N_t(A)})}\right] \tag{3-2}$$

Na prática, o UCB é uma família de algoritmos que modela a confiança para a estimativa de recompensa de diferentes formas [53].

3.1.2 Contextual Bandits

Em muitos problemas de *bandits*, incluindo sistemas de recomendação, os agentes têm acesso a informações adicionais além do histórico de recompensas obtidos em cada rodada. Essas informações podem ser utilizadas para gerar estimativas de recompensa esperada com maior qualidade ao selecionar uma determinada ação. Por exemplo, ao recomendar um filme, o sistema tem informações sobre o usuário, metadados do filme e informações contextuais como histórico de interações e dados da sessão do usuário, entre outras informações. Todas essas informações compõem o contexto da recomendação, e para esse cenário, o mais indicado é utilizar o *Contextual Bandits* [41], também conhecido como aprendizagem por reforço associativo [16] ou MAB com covariáveis [121].

Para *contextual bandits*, a recompensa r_t da rodada t depende do contexto x_t e da ação a_t escolhida. Dessa forma, a estimativa de recompensa na rodada t é dada por $\pi(a_t|x_t)$. Há diferentes abordagens presentes na literatura para *contextual bandits*, desde a discretização do espaço de contexto e utilização de um *bandit* por contexto [100], a

utilização de *bandits* com "conselho de especialista" [17, 18] e as mais populares que são as funções geradoras de recompensas lineares como LinUCB [39] e LinTS [13].

Para Cortes [41], os métodos de MAB podem ser adaptados para utilização de contexto por meio da construção de um "oráculo", modelo de aprendizado supervisionado treinado com base nos dados coletados das políticas existentes, sendo uma caixa-preta ao processo e um estimador de recompensa na forma $r_t = \hat{\rho}(a_t, x_t)$. O oráculo pode substituir os estimadores paramétricos dos modelos de MAB, e desde que haja uma estratégia de exploração definida, o oráculo se torna um estimador não enviesado da política presente nos dados. Outros problemas relacionados à construção de um oráculo ou aprendizado com base em dados coletados de uma política anterior são apresentados nos trabalhos [11, 25, 47].

O pseudocódigo do algoritmo de ϵ-Greedy adaptado para utilização de contexto com oráculo é apresentado no algoritmo 3.1. Onde o parâmetro p define a probabilidade de acontecer uma recomendação aleatória (fase de exploração), caso contrário o item a ser recomendado é o que apresenta a maior estimativa do "oráculo" (intensificação), e para cada rodada t, a observação $\{x^t, r_a^t\}$ contendo o contexto e a recompensa da rodada é adicionada ao registro de *logs*, por fim o "oráculo" é retreinado com mais essa informação. Uma revisão deste e de outros algoritmos pode ser analisado no trabalho de Cortes [41].

Algoritmo 3.1: Algoritmo ϵ-Greedy para *contextual bandits*

Input: probabilidade $p \in (0, 1]$, oráculos $\hat{f}_{1:k}$
for *cada rodada sucessiva t com contexto* x^t **do**
  $d \leftarrow$ numero randômico $\in (0, 1)$;
  if $d < (1 - p)$ **then**
    seleciona uma ação dada por $a = argmax_k \hat{f}_k(x^t)$
  else
    seleciona uma ação a uniformemente ao acaso de 1 a k
  Obtém a recompensa r_a^t
  Adiciona a observação $\{x^t, r_a^t\}$ ao histórico do *arm* a
  Atualiza o oráculo $\hat{f}_a$ com o novo histórico

3.1.3 Algoritmos com restrição de justiça

O aprendizado por ambiente com métodos de *bandits* apresenta diversos aspectos úteis à modelagem de sistemas de recomendação *multistakeholders* com restrições de justiça. Por ser definido como um processo de decisão de markov, o aprendizado se adapta rapidamente a cenários não estacionários por meio de *feedbacks* e aprendizado contínuo, mas, além disso, a própria definição da recompensa e a estratégia de exploração são componentes úteis para modelar os múltiplos objetivos dos *stakeholders* e as restrições

de justiça do sistema. É possível encontrar diversos trabalhos na literatura que mesclam esses conceitos [37, 77, 112, 111, 73, 97, 161, 105, 120].

Jeunen e Goethals [73] modelam uma heurística que adapta a política de exploração dos *bandits* para balancear a relevância das recomendações em uma lista *top-k* com restrições de justiça ligadas à exposição dos itens. O balanceamento é realizado ao nível de usuário, visto que a receptividade à randomização pode variar muito na população geral. O método apresentado, independentemente do algoritmo de *bandit* utilizado, se mostrou eficaz em reduzir a disparidade de exposição com impacto insignificante na utilidade da lista.

Já na pesquisa desenvolvida por Mehrotra et al. [111], os autores modelaram as questões de relevância, justiça e satisfação a respeito do contexto de recomendação de músicas (Spotify) usando o sinal de recompensa. Os autores metrificam cada um desses conceitos e apresentam diferentes combinações dessas métricas em um único sinal de recompensa combinada de forma ponderada. Também reforçam a importância de considerar a disposição do usuário em relação ao conteúdo recomendado pela justiça, o que pode impactar nas métricas de acurácia. Em outro trabalho do mesmo grupo, McInerney et al. [105] formularam um problema de recomendação com restrição de explicabilidade utilizando *contextual bandits*, em que, o sistema aprende a quais explicações (agrupamentos de músicas com um conceito central) cada usuário responde e o quanto deve explorar ou intensificar dentro de cada explicação.

Por fim, o Wang et al. [161] propôs um modelo baseado em *bandits* com política de exploração UCB treinado por meio de otimização multiobjetiva para abordar a exploração e a diversidade de recomendações em um mercado de *delivery* de alimentos. O sistema é integrado com uma camada de predição de demanda cujo objetivo é contextualizar melhor a recomendação para atender as restrições de justiça entre os entregadores.

Os trabalhos apresentados mostram o quanto a modelagem de *bandits* é adaptável a diferentes necessidades, em especial a forma de exploração e a modelagem de recompensa, o que é ideal para o propósito do trabalho.

3.1.4 Aprendizado e avaliação *off-policy*

Treinar novas políticas e avaliar hipóteses em um cenário de aprendizado por ambiente, que necessita de interações *online*, pode implicar um tempo elevado para obtenção de resultados. No cenário de sistemas de recomendação, adotar modelos que ainda não convergiram acima de um *baseline* gera perdas financeiras e podem depreciar a qualidade das recomendações e a experiência do usuário na plataforma. Dessa forma, mesmo abordagens focadas no aprendizado *online* utilizam de registros de *logs*, contendo

as interações do agente com o ambiente, contexto, ação e recompensa, para treinar e avaliar novas políticas em um ambiente *offline*.

No entanto, o aprendizado e avaliação *off-policy* [130, 93, 55, 47] são reconhecidos como um problema difícil, em virtude de tais dados serem tendenciosos para a política de coleta (as previsões fornecidas pelo algoritmo histórico serão sobre-representadas) e estarem incompletos (o *feedback* para outras previsões não estarão disponíveis) [150]. Para resolver esse problema, precisamos de estimadores contrafactuais [28], de modo a estimar como uma nova política teria performado se estivesse recomendado itens em vez da política de coleta. Alguns dos estimadores incluem o *Direct Method* (DM) [25], *Inverse Propensity Score* (IPS) e suas variações [147, 151] e o *Doubly Robust* (DR) [47]. Vários trabalhos aplicaram esses estimadores para avaliação [92, 46] e aprendizado [105, 150] *off-policy*.

O *Direct Method* (DM) forma uma estimativa $\hat{\rho}(a, x)$ da recompensa esperada condicionada ao contexto e à ação. Estimamos o valor da política usando o seguinte estimador de Monte-Carlo (Equação 3-3) sobre o conjunto de dados históricos.

$$\hat{v}_{DM}^{\pi_e} = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \sum_{a \in \mathcal{A}} \pi_e(a \mid x) \hat{\rho}(a, x), \tag{3-3}$$

em que, $\rho(x, a) = \mathbb{E}_{(x,a,r)\sim\pi_c}[r|x, a]$, π_e a política avaliada e π_c a política de coleta. A vantagem desse estimador refere-se à sua simplicidade, e à baixa variância. No entanto, não considera a distribuição da política avaliada. Além disso, sua aproximação dependerá diretamente dos dados da política de coleta, que muitas vezes é enviesada.

Por outro lado, o *Inverse Propensity Score* (IPS) forma uma estimativa da política de coleta π_c e usa a amostragem de importância para rebalancear as recompensas geradas por ela, de modo que, sejam estimativas imparciais da política avaliada π_e. Portanto, o valor da política estimada pela Eq. 3-4 é menos propensa à indução de viés em comparação ao DM, mas tem uma variância muito maior, especialmente quando a política avaliada é muito diferente da política de coleta. Há heurísticas para controlar esse problema, introduzindo uma compensação de viés-variância ou limitando e normalizando os pesos de importância [46].

$$\hat{v}_{IPS}^{\pi} = \frac{1}{|\mathcal{D}|} \sum_{(x,a,r) \in \mathcal{D}} \frac{\pi_e(a \mid x)}{\hat{\pi_c}(a \mid x)} r. \tag{3-4}$$

Enquanto isso, o *Doubly Robust* (DR), apresentado na Eq. 3-5, combina as duas estimativas, a da recompensa e a da política de coleta, por meio do uso da primeira como linha de base e a última para correção. Portanto, se pelo menos, um deles for preciso, o DR também será preciso.

$$\hat{v}^{\pi}_{DR} = \frac{1}{|\mathcal{D}|} \sum_{(x,a,r)\in\mathcal{D}} \left[\frac{\pi_e(a \mid x)}{\hat{\pi_c}(a \mid x)} (r - \hat{\rho}(x,a)) + \sum_{a\in\mathcal{A}} \pi_e(a \mid x)\hat{\rho}(x,a) \right]. \tag{3-5}$$

Além da avaliação, o treinamento de modelos baseados em aprendizado por reforço podem ser realizados de forma *offline* por meio de métodos de *off-policy learning*. No caso do *contextual bandits* com abordagem de *oráculo*, em que o modelo é treinados de forma supervisionada a partir dos dados de coleta, é importante garantir o não enviesamento do modelo quando treinados de forma *offline*. Para isso, podemos usar a *Counterfactual Risk Minimization* (CRM) [150], apresentada na Eq. 3-6, como uma função objetivo que pode ser adaptada para estimadores treinados por gradiente descendente. Esse método de aprendizado usa um dos estimadores contrafactuais para modificar o *log* da verossimilhança, e, em vez de política de avaliação, consideramos uma distribuição uniforme sobre as ações $\mathcal{U}(a)$, em virtude de os dados articularem-se à política de produção π_c e não de um experimento aleatório uniforme [105].

$$\mathcal{L}(\theta) = -\frac{1}{|\mathcal{D}|} \sum_{(x,a,r)\in\mathcal{D}} \frac{\mathcal{U}(a)}{\hat{\pi_c}(a \mid x)} \log p_\theta(r \mid x, a). \tag{3-6}$$

3.1.5 Ambientes de simulação

O desenvolvimento de sistemas de recomendação de ponta a ponta é um processo complexo por envolver sistemas de *softwares* de grande ponte, grande quantidade de dados e interações humanas. O processo iterativo de melhoria constante dos sistemas de recomendação envolve um cuidado a respeito da avaliação destes e dos impactos das mudanças em várias variáveis de interesse. Dessa forma, os ambientes de simulação, comuns em aprendizado por reforço, são uma alternativa viável para minimizar os riscos e aumentar a velocidade relativa ao desenvolvimento de melhorias. Recentemente, há um interesse significativo nas plataformas de simulação de RecSys pela indústria, o que tem permitido aos desenvolvedores a criação de ambientes simulados em que seus modelos podem ser analisados antes de entrarem em produção [23].

Em termos de ambientes simulados com o objetivo de modelar e avaliar os sistemas de recomendação, identificamos diferentes trabalhos na literatura que também são suportados pela indústria, tais quais o *RecoGym* [129] e *PyRecGym* [142], plataformas criadas com base no simulador do *OpenAI's Gym* para interações sequenciais que possibilitam o desenvolvimento de agentes de reforço para RecSys usando um ambiente de simulação bem estabelecido em outras áreas. O *RecoGym* usa de dados sintéticos para avaliar a interação sequencial do usuário, combinando *feedback* orgânico e exibição de anúncio intermitente. O *PyRecGym* estende a ideia do *RecoGym* com maior flexibilidade

nos tipos de dados de entrada e na função de recompensa, mas, por outro lado, usa dados de *logs* reais para simular a interação do usuário com o ambiente, o que pode introduzir um viés em relação à política de coleta em comparação à utilização de dados sintéticos.

Outros trabalhos mais recentes, como o *RecSim* [70, 114] e *OBP* [131], divergem a respeito do propósito da simulação. O primeiro fornece mais flexibilidade para a definição do ambiente que os demais, mas requerer a implementação de camadas de abstração para emular aspectos específicos do comportamento do usuário e da geração dos dados sintéticos, o que por sua vez, pode criar uma lacuna entre a simulação e a realidade. O *OBP* é focado na avaliação *off-policy* de problemas de *bandits* para RecSys em ambiente simulado com base em dados reais e sintéticos e embora tenha menor flexibilidade de configuração que o *RecSim* apresenta um catálogo grande de métodos *off-policy* e modelos implementados.

No entanto, essas plataformas assumem uma perspectiva centrada no usuário similar à apresentada na Fig. 3.2. Os episódios constituem sessões dos usuários e o objetivo é maximizar a satisfação do usuário. Em uma perspectiva de ambiente de recomendação *multistackholder*, como um *marketplace*, faz-se necessário simular diversos usuários atuando em cada episódio visando maximizar a satisfação de todas as partes envolvidas (usuários, fornecedores, terceiros etc). Além de restrições de justiça gerais do ecossistema.

Embora os ambientes de simulação tenham ganhado espaço nos últimos anos, principalmente pelo suporte dado por grandes empresas, ainda é um ambiente pouco explorado e com grande espaço para melhorias, em especial para cenários *multistakeholders*, devido à complexidade já apresentada.

3.2 *Deep Learning* para sistemas de recomendação

Dentre as técnicas de aprendizado e extração de padrões, os métodos de Aprendizado Profundo (*Deep Learning* – DL) têm se destacado com avanços significativos em diversas áreas do conhecimento. Uma de suas principais vantagens está na possibilidade de extração de padrões complexos em dados estruturados e não-estruturados. Problemas como classificação de imagens, reconhecimento da fala, tradução de textos, entre outras, obtiveram resultados próximos ou melhores que *benchmarks* humanos com a utilização de modelos de DL. Na área de sistemas recomendação, embora existam diversas técnicas consolidadas, que podemos chamar de clássicas, os métodos de DL apresentam abordagens mais eficazes para modelar a relação complexa de preferência que há entre os usuários e os conteúdos, além de apresentarem a flexibilidade necessária para resolver problemas de forma supervisionada, não supervisionada e com aprendizado por ambiente [170].

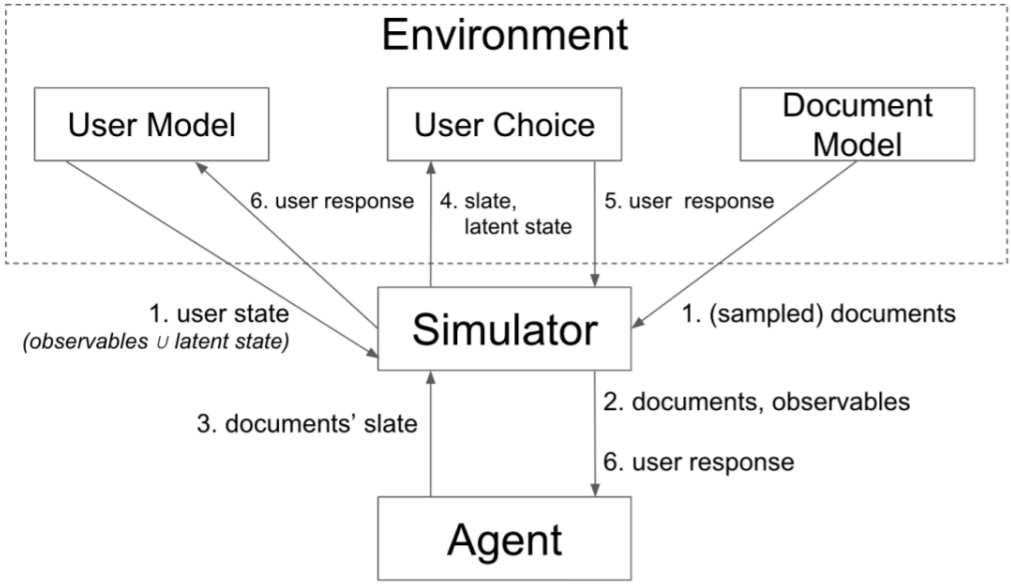


Figura 3.2: Fluxo de controle (único usuário) na arquitetura RecSim [70]. O ambiente consiste em um modelo de usuário focado em modelar e amostrar os usuários, um modelo de documento com mesmo objetivo para os documentos recomendáveis e um modelo de escolha do usuário que determina a resposta do usuário ao documento recomendado pelo agente, enquanto o simulador serve como interface entre o ambiente e o agente e gerência as interações entre os dois usando as seis etapas descritas na imagem. Imagem retirada de [70]

Segundo LeCun, Bengio e Hinton [87], os métodos de *Deep Learning* são métodos de aprendizagem que utilizam múltiplas camadas, obtidas pela composição de módulos simples, não-lineares e que podem ser combinados de diferentes maneiras. A aprendizagem profunda permite aprender representações de dados com múltiplos níveis de abstração e, embora pesquisas nessa temática já tenham produzido avanços relevantes em áreas como visão computacional, reconhecimento da fala e processamento de linguagem natural [66, 67, 22, 21], a aceitação de *Deep Learning* pela comunidade de sistemas de recomendação foi relativamente lenta, pois popularizou-se, apenas, a partir de 2016 com "*Workshop on Deep Learning for Recommender Systems*" no ACM RecSys 2016 [45].

As técnicas de *Deep Learning* apresentam uma série de características úteis para diferentes aplicações na área de sistemas de recomendação. Zhang et al. [170] elencam esses pontos da seguinte forma:

- *modelagem não-linear* – a relação de preferência entre os usuários e os conteúdos é uma relação complexa, modelos clássicos como fatoração de matrizes perdem capacidade de generalização da solução por serem abordagens lineares. DL apresenta a capacidade de modelar a não linearidade dos dados em diferentes camadas;

- *representação do aprendizado* – os tipos de dados disponíveis sobre o conteúdo são variados, como informações categóricas, numéricas, textos, imagens, áudios etc. Com DL é possível processar e extrair padrões de forma multimodal por meio da criação de *embeddings* [33], em que é possível obter uma representação vetorial do conteúdo, inclusive com característica semântica;
- *modelagem sequencial* – a recomendação pode ser vista como um problema sequencial de tomada de decisão, em que a série de eventos de longo (histórico dos dados) ou curto prazo (sessão do usuário) devem ser modeladas como contexto para recomendação. Arquiteturas de DL como RNNs [65], e, mais recentemente as *Transformers* [148, 146], podem ser utilizadas para, efetivamente, modelar dados em sequências;
- *flexibilidade* – no problema de recomendação existem diferentes sinais, métricas, e restrições ligadas ao negócio que requerem diferentes abordagens. As redes neurais podem ser modularizadas e combinadas para formular modelos de recomendação híbridos, que atendam características específicas do negócio.

Outras características relevantes em sistemas de recomendação referem-se à necessidade de se trabalhar em escala, com grandes quantidades de dados e com conjuntos dinâmicos de itens e usuários. Outra vantagem de DL refere-se ao fato de o treinamento ser realizado em *mini-batches* e quando necessário de forma incremental. Além disso, é possível reduzir a dimensionalidade sem perder a representatividade da informação original com arquiteturas como CNNs [152] e AE [163], e, devido à utilização de *embeddings* indexados, é possível atualizá-los de forma independente dos dados históricos [175].

Nesse contexto, em virtude das arquiteturas de DL apresentarem flexibilidade na modelagem e por ser uma área promissora, diversas arquiteturas específicas para sistemas de recomendação foram propostas nos últimos anos. De forma geral, o foco das pesquisas tem sido na modelagem supervisionada, como a predição de CTR [38], filtragem colaborativa para estimativa de *feedbacks* explícitos e implícitos [63] e predição de próxima interação [146] para problemas de *session-based*, entre outras. Recentemente, alguns trabalhos vêm adequando a utilização de modelos de DL para problemas de *bandits* com modelagem de reforço [40].

3.2.1 Arquiteturas de *Deep Learning* para RecSys

Nesta seção apresenta-se brevemente as arquiteturas que procuram predizer o valor da preferência implícita, como CTR, enfocando os trabalhos que são mais relevantes para esta tese. Do ponto de vista de problemas de *bandits* em aprendizado por ambiente, as arquiteturas apresentadas com esse foco podem ser utilizadas como estimadores de

recompensas complexas, não lineares e não enviesados desde que treinadas com base em um ciclo de reforço.

He et al. [63] apresentam uma arquitetura simples para recomendação usando o conceito de filtragem colaborativa treinada em dados de *feedbacks* explícitos ou implícitos, a *Neural Collaborative Filtering* (NCF) é apresentada da Fig. 3.3. A arquitetura modela diretamente a interação entre usuários e itens utilizando uma representação vetorial (*embeddings*) para ambos com base na identificação *one-hot* do usuário u e do item i, seguindo de múltiplas camadas de *multi-layer neural network* que visam extrair a relação não-linear entre o usuário e o item, por fim, a camada de saída é um *score* previsto para essa interação. A função de escore é definida como $\hat{r}_{ui} = f(U^T.u, V^T.i|U, V, \theta)$, em que, $f(.)$ representa as camadas da rede neural, e θ os parâmetros aprendidos dessa rede.

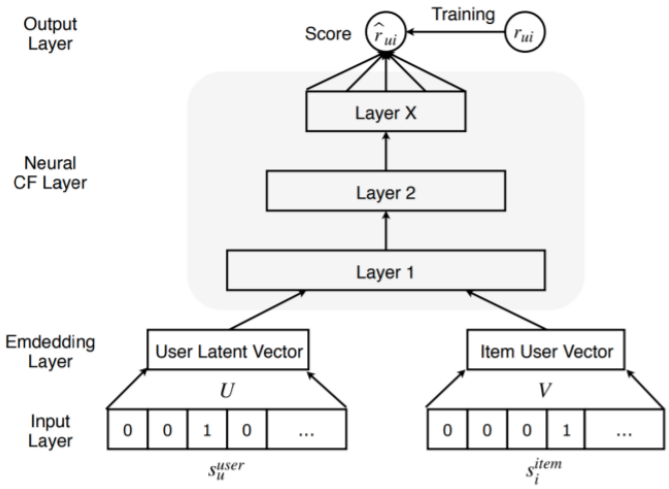


Figura 3.3: Arquitetura da *Neural Collaborative Filtering* (NCF). Imagem retirada de [170]

Toda a rede pode ser treinada com base em dados de preferências explícitas, como um problema de regressão usando a função de custo de erro quadrático médio (MSE) ou a partir de preferências implícitas, para estimar a probabilidade da interação, por meio da *binary cross-entropy*. Segundo Zhang et al. [170], a NCF pode ser vista como uma generalização da fatoração de matrizes tradicionais. Portanto, é conveniente fundir a interpretação neural da fatoração de matrizes com redes neurais para formular um modelo mais geral que use a linearidade da fatoração de matrizes tradicionais e a não linearidade das redes neurais para melhorar a qualidade da recomendação.

A arquitetura *Wide & Deep* [38], conforme na Fig. 3.4, foi introduzida pelo Google em 2016 e foi utilizada para recomendações de aplicativos no Google Play. Um diferencial de sua arquitetura foi a separação da rede em uma camada *wide*, que compreendia todas as *features* numéricas e transformações realizadas, bem como a composição de novas *features* com base em *cross-product transformation*, e uma camada *deep*, fo-

cada na extração de padrões de alto nível em dados categóricos. Todas essas *features* são combinadas por um componente linear generalizado como uma regressão logística para estimar um escore, seja para problemas de preferência explícitas ou implícitas.

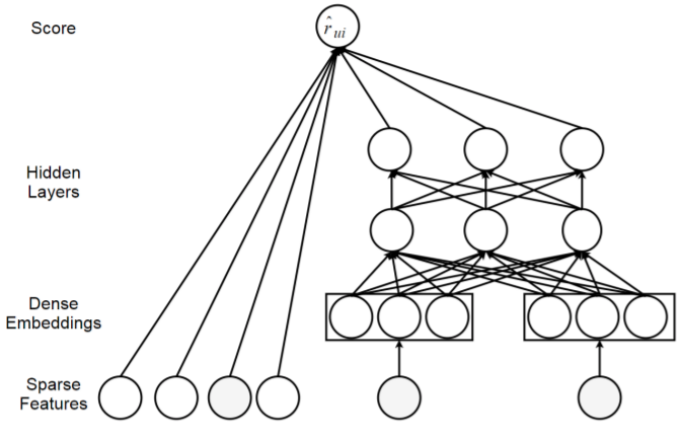


Figura 3.4: Arquitetura da *Wide & Deep*. Imagem retirada de [170]

Em um trabalho mais recente, Zhao et al. [173] estendem o conceito apresentado pela *Wide & Deep* e apresenta uma arquitetura *multi-modal* e *multi-task* para recomendação de vídeos do YouTube. A *Multi-gate Mixture-of-Experts* (MMoE), exibida na Fig. 3.5, tem como entrada diferentes tipos de dados (texto, dados categóricos, imagens, numéricos etc.) sendo treinada para predizer múltiplos objetivos separados em duas categorias, os objetivos de engajamento como *clicks* do usuário, grau de engajamento com o vídeo recomendado etc., e os objetivos de satisfação, como *likes* e pontuação da recomendação.

Todas essas arquiteturas mostram a flexibilidade em modelar o problema de recomendação, desde a inclusão de diferentes metadados como entradas do modelo e a padronização como vetores de *embeddins* até a modelagem de diferentes objetivos. Essas características corroboram a utilidade dessas técnicas no cenário apresentado nesta tese de recomendação em um ambiente *multistackholders*, em que uma grande variedade de sinais que podem ser utilizados como entrada assim como a possibilidade de desenvolvimento de modelos multiobjetivos com restrições de justiça. Por outro lado, devido à complexidade desses modelos ser um contraponto, a simplicidade das estratégias de exploração apresentadas nos problemas de *bandits* geralmente mesclá-los não é a primeira opção.

3.2.2 Neural Contextual Bandits

As chamadas *Neural Contextual Bandits* [40, 166, 128, 127, 58], algoritmos de *contextual bandits* com componentes de *redes neurais*, são recentes na literatura. Geralmente, os componentes adaptados são o estimador de recompensas visando gerar

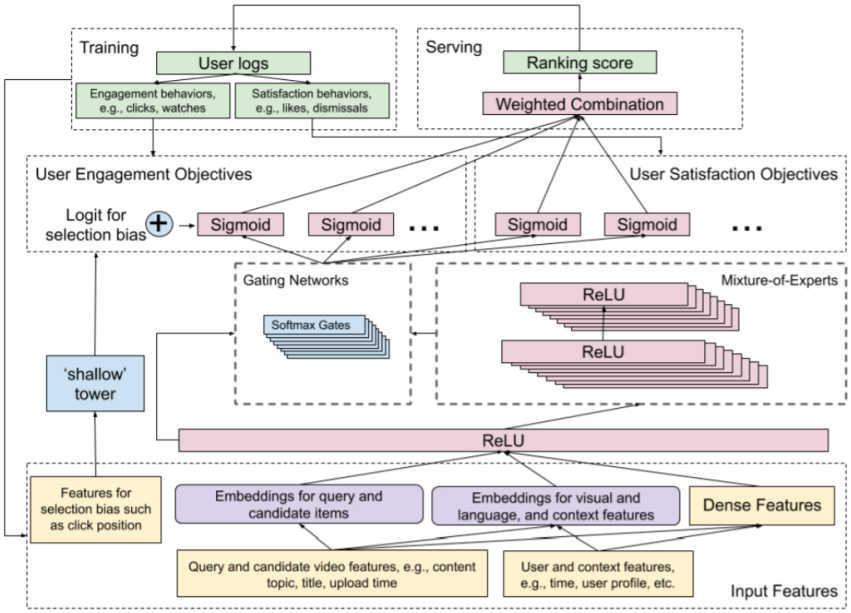


Figura 3.5: Arquitetura *Multi-gate Mixture-of-Experts (MMoE)*. Imagem retirada de [173]

estimativas não lineares dado o contexto e a ação assim como os extratores de *features* visando processar e extrair vetores de características com base nos dados originais que definem o contexto.

Zhou, Li e Gu, Xu et al. apresentam o *NeuralUCB* [177] e *Neural-LinUCB* [166] em abordagens similares, ambos estendem a modelagem do *LinUCB* [1], mas de formas diferentes. Enquanto Zhou, Li e Gu usam uma rede neural $f(x;\theta)$ para prever a recompensa do contexto x e limites de confiança superiores calculados a partir da rede para guiar a exploração, Xu et al. assumem que a função geradora de recompensa $r(.)$ pode ser expressa pelo produto interno entre uma representação das *features* e parâmetros de pesos de exploração dada por $r(.) = \langle \theta^*, \psi(.) \rangle$, em que $\theta^* \in \mathbb{R}^d$ é algum parâmetro de peso e $\psi(.)$ é o mapeamento de *features* do contexto. Xu et al. sugerem usar uma rede neural para fazer o mapeamento das *features* ao usar a última camada oculta da rede para transformar os vetores de característica original em um *embedding* e realizar a exploração do tipo UCB usando com base esse vetor, o que, segundo os autores, tem um ganho de desempenho e eficiência computacional em comparação à abordagem de Zhou, Li e Gu.

Em outra abordagem, Guo et al. [58] utilizam a arquitetura *Deep & Wide* como um preditor de CTR, algo como o oráculo de um *contextual bandits*, ajustado com um algoritmo de aproximação a posteriori para obter a incerteza do modelo. Dessa forma, considerando-se uma amostragem da distribuição de CTR para cada item, e então, um algoritmo de exploração, nesse caso, os ϵ-greedy, Thompson Smapling e UCB, seleciona k itens para exibir para um determinado usuário. Nesse trabalho, os autores utilizaram um modelo de DL como estimador da recompensa esperada, embora tenham usado o *Deep*

& Wide como arquitetura padrão. Qualquer outra arquitetura destinada a estimar CTR poderia ser utilizada nesse contexto.

Em uma outra pespectiva, Rigotti e Zhu [127] apresentaram um método novo de exploração, chamado *Sample Average Uncertainty (SAU)*, que mede, diretamente, a incerteza das recompensas médias da amostra ao longo do tempo. É uma abordagem frequentista em comparação às baseadas em aproximação a posteriori. Os autores apresentam duas versões desse método, uma que pode ser usada como um limite superior de confiança, similar ao UCB, e outra que amostra valores de uma distribuição gaussiana paramétrica com uma média dada pelo estimador de recompensas. Os autores reforçam que o estimador de recompensas pode ser obtido por uma rede neural $r = f(x; \theta)$ ou por um estimador linear qualquer $r = x^T\theta$.

Em trabalho mais recente, Zhu e Van Roy [178] da META exploram a escalabilidade dos métodos de *Neural Contextual Bandits* e propõem o Epistemic Neural Recommendation (ENR). Os autores reforçam que uma das limitações na utilização dos *Neural Contextual Bandits* tem sido a demanda computacional necessária para otimização desses métodos, e enfatizam a importância de uma amostragem eficiente.

3.3 Considerações finais

Neste capítulo foram apresentados aspectos gerais de aprendizado de máquina por ambiente e como esse conceito se relaciona com ambientes de recomendação *multistackholders*. Foi detalhado uma subcategoria de métodos de aprendizado por reforço, os *bandits*, e como esses algoritmos têm sido utilizados para modelar o problema de recomendação como um processo de tomada de decisão sequencial bem como eles são adaptados para um cenário de recomendação com restrições de justiça. Em seguida, foi resumido os principais pontos do aprendizado e avaliação *off-policy* e discutido como esses conceitos podem ser utilizados para treinar modelos por meio dos *logs* de interação reais de *marketplaces*.

Apresentamos e discutimos a utilidade dos ambientes de simulação utilizados pela indústria para o desenvolvimento de novas abordagens de recomendação, ressaltando que, por meio dos *frameworks* de simulação, é possível reduzir os riscos para validação de novos algoritmos e dar velocidade ao possibilitar interações consecutivas.

Por fim, reforçamos os principais benefícios ao utilizar redes neurais para sistemas de recomendação. Apresentamos algumas arquiteturas relevantes e como adaptá-las para o cenário de aprendizado por ambiente com *bandits*. Finalizamos o capítulo descrevendo as *Neural Contextual Bandits* e como essas novas categorias de algoritmos podem ser utilizadas no cenário de recomendação *multistackholder* com restrições de justiça.

CAPÍTULO 4

Metodologia

Neste capítulo, apresenta-se a metodologia que será utilizada na presente tese, a fim de desenvolver um modelo sistema de recomendação baseado em aprendizado por ambiente destinado a ambientes *multistackholder*s com modelagem de justiça. O modelo é baseado nos modelos estado-da-arte que utilizam *Deep Learning* (DL) e modelagem de *Contextual Bandits*.

A proposta do *Neural Contextual Bandits* com restrições de justiça é apresentada em duas etapas. A primeira, apresentada na seção (4.1) e (4.1.3), baseia-se no desenvolvimento de um ambiente de simulação contextualizado para o cenário *multistackholder* onde expomos detalhes da implementação do simulador *MARS-Gym*, onde iremos desenvolver e avaliar o modelo proposto em um ambiente controlado seguindo os trabalhos de [70, 142, 129]. Na segunda, apresentada na seção (4.2), descrevemos o método proposto para modelar a restrição de justiça, bem como os interesses dos múltiplos *stackholders* por meio de métodos de exploração e sinal de recompensa seguindo os trabalhos [111, 106, 58], mas adicionando uma modelagem de contexto baseada em *Deep Learning*.

4.1 Desenvolvimento do ambiente de simulação

Nesta seção, apresentamos todas as etapas para a construção de um ambiente de simulação com propósito de ser uma ferramenta utilizada nesta tese e posteriormente pela indústria, visando dar suporte ao desenvolvimento de novos modelos de sistemas de recomendação baseados em aprendizado por ambiente. O objetivo do simulador proposto é simular as interações realizadas em um ambiente *multistackholder* como um *marketplace*, provendo as ferramentas necessárias para preparar os dados, treinar os modelos e avaliar os resultados em diferentes perspectivas.

O MARS-Gym (Marketplace Recommender System Gym), detalhado nas seções seguintes, é um *framework* para modelagem, treinamento e avaliação de sistemas de recomendação baseados em aprendizado por reforço para *marketplaces*. O código fonte do MARS-Gym está disponível online em https://github.com/deeplearningbrasil/mars-gym.

4.1.1 Simulação do PDM

A principal característica do MARS-Gym é a modelagem dinâmica de um *marketplace* como um Processo de Decisão de Markov (PDM). O *framework* processa um *ground-truth dataset* com os *logs* de interação de uma *marketplace* real para realizar uma simulação realista. Em seguida, gera um ambiente *Gym* da OpenAI de modo que os dados direcionam as transições internas resultantes da interação do recomendador com o ambiente.

O ambiente fornece um protocolo com os requisitos necessários para que os dados sejam utilizados como fonte de simulação pelo *framework*, com dois requisitos principais. Primeiro, uma lista de interações entre usuários e fornecedores, contendo todas as informações contextuais e uma variável binária que explicita se cada interação foi bem-sucedida (por exemplo, uma compra ou um clique). Segundo, os metadados que descrevem os usuários e os fornecedores.

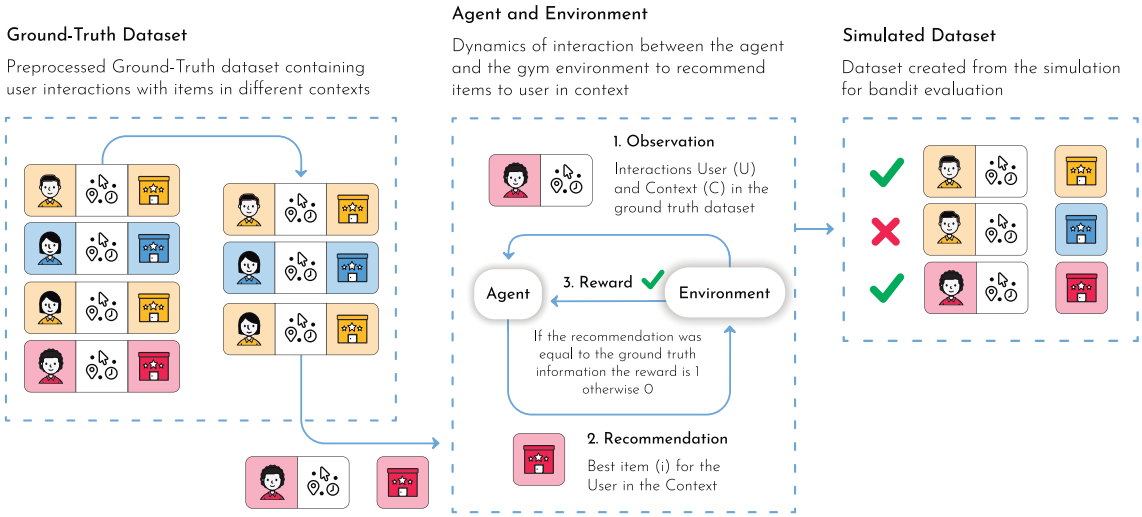


Figura 4.1: Diagrama de fluxo *MARS-Gym*, desde a ingestão do conjunto de dados para geração do ambiente até a simulação do PDM.

Na Fig. 4.1, apresenta-se como o MARS-Gym simula a dinâmica do *marketplace*. Em primeiro lugar, a estrutura filtra apenas as interações bem-sucedidas, uma vez que elas são a fonte disponível da verdadeira distribuição de recompensas. Em seguida, o ambiente *Gym* envolve os dados resultantes, em que cada etapa é uma interação do usuário com o fornecedor. A observação fornecida contém o usuário associado e seus metadados, bem como as informações contextuais do *log*. Com a ação selecionada, o agente deve retornar a recomendação de um fornecedor. O ambiente também fornece dados informativos adicionais por meio de um dicionário (por exemplo, uma lista dos itens disponíveis para recomendação, de modo a restringir o espaço de ação).

Calculamos a recompensa comparando a ação selecionada pelo agente com a fornecida pelo *log*. O agente recebe uma recompensa positiva se corresponder. Portanto,

o agente só descobre o fornecedor de destino no cenário de uma recomendação bem-sucedida. Caso contrário, deve explorar as ações para construir seu conhecimento.

A sequência de etapas segue a sequência de interações no *ground-truth dataset* filtrado para manter a dinâmica temporal. Definimos um episódio como uma iteração através de todos os *logs*, em vez da sessão do usuário. Esse comportamento pretende aproximar o cenário multiagente e manter a perspectiva no *marketplace*, não apenas no usuário. Por fim, as interações entre o agente proposto e o ambiente geram novos *logs* de interação. Esses dados, provenientes da simulação, são utilizados para treinar a política do agente e também fornecem a curva de recompensa cumulativa como a primeira fonte de avaliação. Na próxima subseção, descrevemos o projeto do *MARS-Gym* para realizar esta simulação.

4.1.2 Design do sistema

Compomos o *MARS-Gym* com três componentes internos principais: O Módulo de Engenharia de Dados, o Módulo de Simulação e o Módulo de Avaliação. A Fig. 4.2 mostra uma representação visual de nossa implementação. Para executá-los todos juntos, usamos o *Luigi* [24], que é uma biblioteca Python para gerenciamento de fluxo de trabalho. Esta biblioteca permite a criação de um grafo de tarefas, em que a saída de uma tarefa é a entrada da próxima. Usamos para compor o grafo de tarefas desde o processamento dos dados até a simulação e avaliação. No restante desta subseção, descrevemos cada módulo separadamente descrito na Fig. 4.2.

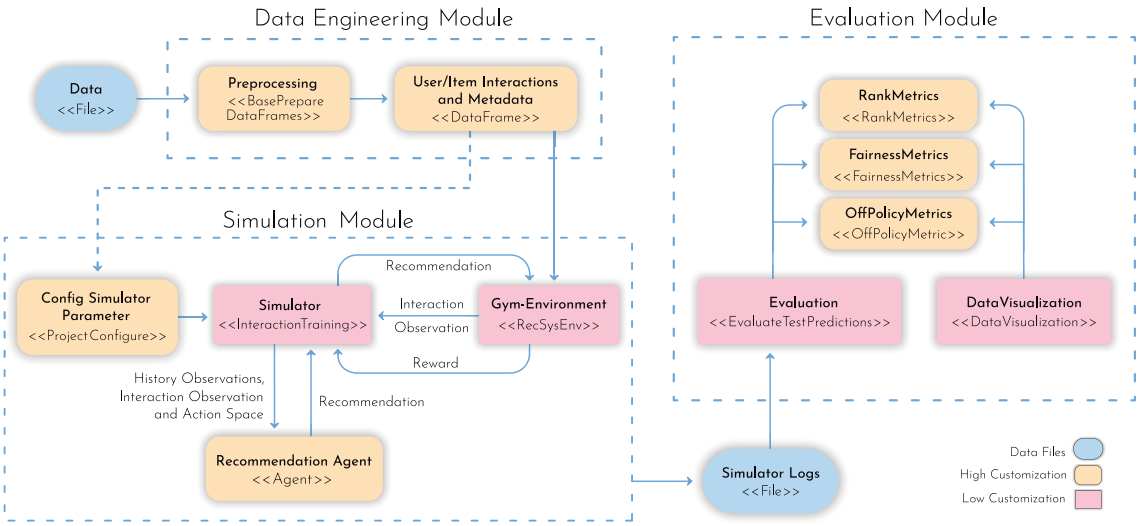


Figura 4.2: Arquitetura do *framework* MARS-Gym e seus três módulos internos.

Engenharia de Dados

Este módulo tem o objetivo de preparar os dados para a simulação, em que é possível limpar o conjunto de dados e aplicar todas as transformações necessárias para fornecer a lista de interações e os metadados associados aos usuários e fornecedores como saída. Ao final do processo, espera-se satisfazer todos os requisitos para criar o ambiente de *Gym* do OpenAI.

O *dataset* é ingerido pelo módulo como um Pandas DataFrames [153, 107]. Cada linha deve conter o identificador do usuário, os campos contendo informações contextuais, a ação (fornecedor) apresentada ao usuário e a recompensa esperada (se uma interação foi bem-sucedida no cenário de marketplace). Todos os campos devem estar no formato numérico, categórico ou array NumPy [117]. Implementamos essa limitação para estar em conformidade com a API OpenAI, mas podemos resolvê-la facilmente por meio das etapas de pré-processamento do conjunto de dados.

Para processar o conjunto de dados, usamos o Apache Spark [169], que é um poderoso mecanismo de computação distribuída para processamento de big data com suporte ao Luigi. Este nos permite ter um desempenho muito maior, mesmo quando executado em um único computador. Descobrimos que essa estrutura melhora consideravelmente o desempenho para processar grandes conjuntos de dados dentro do *MARS-Gym*.

Naturalmente, *marketplaces* distintos coletam dados de muitas maneiras diferentes. Portanto, é impossível fornecer uma única tarefa de processamento de dados que lide com todos os cenários. Por esse motivo, esse módulo é altamente personalizável e fornece abstrações e ferramentas para auxiliar na criação de tarefas de processamento de dados.

Módulo de Simulação

Esse módulo é o núcleo do MARS-Gym e implementa a simulação PDM descrita na Subseção (4.1.1). Além disso, também é o módulo para projetar e treinar agentes de recomendação. Compomos o MARS-Gym com três componentes principais: o Agente de Recomendação, o Ambiente *Gym* e o Simulador.

Conforme descrito anteriormente, o Ambiente *Gym* implementa a interface do *Gym* da OpenAI. Ele consome o conjunto de dados processado para criar uma representação da dinâmica do *maketplace* usando as mesmas interações bem-sucedidas, e, computa as recompensas usando as ações correspondentes como verdade. Por outro lado, o módulo do Agente de Recomendação implementa a interface do agente, que expõe os métodos *act* e *fit*. Especificamente, MARS-Gym requer que o método *act* retorne não apenas a ação, mas também as respectivas probabilidades de cada ação disponível. Portanto, qualquer sistema de tomada de decisão que satisfaça esses requisitos pode executar uma simulação usando o *framework*.

Para o método *fit*, MARS-Gym espera treinar o agente usando os dados de *logs* gerados pela simulação. Já fornecemos módulos PyTorch de alto nível com arquiteturas parametrizadas (incluindo regressão logística, *factorization machines* e redes neurais), algoritmos de otimização baseados em gradiente e métodos de regularização. Como também implementamos todo o procedimento de otimização do PyTorch (com suporte a GPU), o único trabalho que deixamos para o usuário do *framework* é combinar esses blocos de construção e selecionar os hiperparâmetros apropriados.

O terceiro componente é o Simulador, que desempenha o papel central do módulo. Ele gerencia o agente e o ambiente para conduzir a simulação entre os episódios, acumulando os dados de *logs* para treinamento e as recompensas para avaliação. A arquitetura proposta assume baixa customização neste módulo e no Ambiente de Gym. Ou seja, não esperamos nenhuma modificação nesses módulos para simular novos agentes ou *marketplaces*.

O Simulador é agnóstico à natureza do treinamento, suportando aprendizado *online* e em lote, que é facilmente configurável. No entanto, optamos por fazer o aprendizado em lote em nossas implementações por estar mais próximo do ambiente real de produção. Em tais sistemas, muitas vezes é inviável fazer o aprendizado *online* a cada interação. Assim, é comum treinar novamente o modelo com uma determinada programação (por exemplo, todos os dias ou semanas). Para dar suporte a esse tipo de aprendizado fora da política, aplicamos a correção nas funções de custo por meio de estimadores contrafactuais, conforme descrito na Eq. 3-6.

Módulo de Avaliação

Este módulo final utiliza os *logs* gerados com base na simulação do PDM para realizar uma avaliação multifacetada. A tarefa de avaliação calcula métricas de recomendação, avaliação *off-policy* e métricas de justiça. Em seguida, o componente de visualização de dados fornece uma interface amigável para apresentar e comparar essas métricas entre diferentes agentes, conforme apresentado na Fig. 4.3.

O Módulo de Simulação gera dois tipos diferentes de *logs* a serem utilizados pelo Módulo de Avaliação. Durante a simulação, ele divide o conjunto de dados processado em subconjuntos de treinamento e teste. Naturalmente, o *framework* usa o subconjunto de treinamento para conduzir o processo de treinamento na simulação, e o Módulo de Avaliação usa os *logs* resultantes para calcular e plotar a recompensa média cumulativa. Essas curvas avaliam a adaptabilidade do agente, o desempenho assintótico e a eficiência da amostra, fatores cruciais para estimar o custo de implantação em produção. Por outro lado, o subconjunto de teste fornece uma nova perspectiva do mesmo PDM, mas o agente interage com ele apenas após o treinamento. Dessa forma, usamos para medir a generalização, como insumos para a tarefa de avaliação.

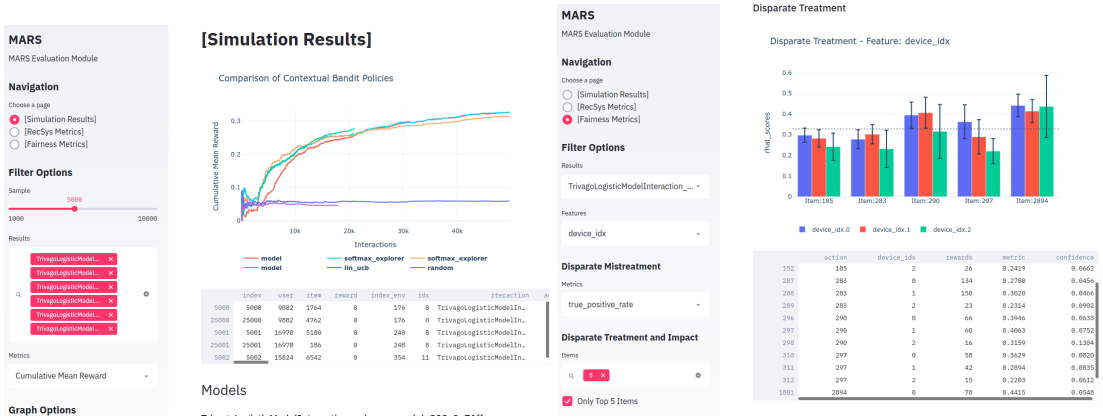


Figura 4.3: MARS-Gym fornece uma interface amigável para facilitar a análise dos atributos e a comparação entre os agentes. É possível avaliar os resultados da simulação, das métricas *off-policy* e das métricas de justiça diretamente na interface gráfica criada no Framework

Esses *logs* de avaliação salvam as informações necessárias para calcular todas as métricas. Para métricas de ranqueamento tradicionais para sistemas de recomendação, usamos listas de relevância para calcular a Precisão, MaP [167] e nDCG [72], bem como a lista ranqueada de ações para calcular a Cobertura [54] e Personalização. Para isso, usar os *logs* provenientes de interações bem-sucedidas no subconjunto de teste.

Em termos de avaliação *off-policy*, implementamos os estimadores de valor da política listados na Subseção (3.1.4): *Direct Method* (DM), *Inverse Propensity Score* (IPS) (e suas variações CIPS e SNIPS) e *Doubly Robust* (DR). Essas métricas são especialmente importantes para comparar com as métricas tradicionais, uma vez que, aplicam correções para lidar com o viés originado da política de coleta. Por exemplo, se um agente A tem um nDCG semelhante, mas uma estimativa de *Doubly Robust* mais baixa do que o agente B, significa que A está mais inclinado para sua política de coleta do que B.

Para essas métricas, primeiramente, precisamos calcular o estimador de recompensa esperado, $\hat{\rho}(x, a)$, e o estimador de política de coleta, $\hat{\pi_c}$, conforme descrito na Subseção (3.1.4). Eles são treinados automaticamente usando todo o conjunto de dados processado (ou seja, a saída do Módulo de Engenharia de Dados) e é o mesmo modelo base do agente. Em seguida, usamos as ações do *ground-truth* e a lista de probabilidades para calcular as métricas.

Por fim, para métricas de justiça, propomos implementações para as equações 2-8, 2-9 e 2-9, para analisar as noções de tratamento díspar, impacto e maus tratos, respectivamente. Para analisar o tratamento díspar, comparamos a distribuição da política sobre todas as ações e a mesma distribuição condicionada ao atributo sensível. Nesse sentido, marginalizamos as interações, em vez de considerá-las separadamente, o que consideramos inviável em larga escala.

Em termos de impacto díspar, nossa implementação escolhe um conjunto limi-

tado de ações populares e, para cada uma delas, traça as pontuações associadas a cada valor possível de um atributo sensível. Dessa forma, podemos comparar como os valores dos atributos impactam uma ação de forma diferente. Por fim, para maus tratos, consideramos as ações *ground-truth* nos *logs* de interações para calcular as métricas de classificação (como precisão, taxa de verdadeiro positivo e outras), agrupadas pelos valores do atributo sensível. Para ilustrar nossas implementações, nos referimos à Subseção (4.1.3).

4.1.3 Implementação de modelos *baselines*

Nesta seção, apresentamos todas as etapas para a implementação dos modelos de *contextual bandits* baselines para o projeto e o *designer* de experimentos para validação da plataforma de simulação proposta. O objetivo dessa etapa é modelar experimentos com diferentes graus de dificuldades utilizando um *dataset* com *logs* reais, bem como o desenvolvimento de algoritmos de *contextual bandits* que servirão como modelos *baselines* ao projeto.

Trivago Marketplace Dataset

Trivago é uma plataforma global de busca de hotéis localizada em mais de 190 países e oferece acesso a mais de dois milhões de hotéis para viajantes. O Trivago centraliza as ofertas hoteleiras em uma única plataforma. Assim, a plataforma precisa dar boas recomendações aos usuários e garantir que os hotéis afiliados sejam tratados de forma justa pela plataforma em diferentes cenários. Além disso, as preferências do usuário mudam ao longo do tempo e dependem do objetivo da viagem, bem como dos preços, tipo e disponibilidade da acomodação. Por fim, há interesse de todos os parceiros (viajante, site de reservas publicitárias e da plataforma Trivago) em sugerir boas recomendações em diferentes aspectos [9].

A Trivago organizou o *ACM RecSys Challenge* em 2019 [125]. Para essa competição, forneceu um conjunto de dados que consiste em *logs* de sessão com 910k amostras. Cada sessão contém uma sequência de interações entre um usuário e a plataforma. Eles podem representar diferentes ações do usuário, como dar um escore para o hotel, visualizar os metadados de itens (informações, imagens e ofertas), ordenar lista exibida, pesquisa de um destino ou ponto de interesse, etc. Além das informações da sessão do usuário, o conjunto de dados também fornece metadados de itens diferentes que caracterizam os hotéis. Na Tabela 4.1, mostram-se as estatísticas gerais do conjunto de dados do desafio Trivago.

Tabela 4.1: Trivago marketplace – estatísticas do conjunto de dados e tarefas de referência

Tarefa	Cidade	Interações	Cliques	Usuários	Sessões	Itens
Trivago Challenge	34,752	15,932,992	1,586,586	730,803	910,683	853,540
RecSys Cities	12	761,702	62,168	31,075	36,846	12,002
New York, USA	1	223,320	18,160	9,158	10,869	1,961
Rio de Janeiro, Brazil	1	161,973	9,122	4,429	5,563	2,080
Chicago, USA	1	22,939	1,890	1,155	1,347	662
Como, Italy	1	1,718	155	112	122	328

Designer de Experimentos

Organizamos o conjunto de dados em cinco tarefas, identificadas principalmente pela cidade de interesse. Na Tabela 4.1, apresentamos as estatísticas de cada tarefa proposta. A tarefa "RecSys Cities" contém um conjunto de 12 cidades presentes no conjunto de dados. Escolhemos as cidades com base em vários fatores além do agrupamento natural: o **tamanho do episódio**, que representa quanto tempo é a sequência de ações que o agente manipula; o **tamanho do espaço de estado**, representado pelo número de sessões únicas e de usuários únicos usuários, que expõe a variabilidade das informações contextuais; o **tamanho do espaço de ação** (ou seja, o número de parceiros disponíveis para recomendação), que indica o quão complexo é o problema de exploração e o **número de cliques**, que dá uma ideia de quantas interações bem-sucedidas à cidade teve para ser usada como ações reais durante a simulação. Para mais informações, consulte a página do conjunto de dados [125] e nosso código-fonte.

Contextual Bandits

Implementamos um conjunto diversificado de modelos *Contextual Bandits* como *baselines* para fácil extensão e comparação com outros agentes. Eles aplicam uma variedade de estratégias de exploração, a maioria delas inspirada em Cortes [41]. Também implementamos a *NeuralUCB* [177], que estende o LinUCB, mas apresenta uma modelagem de recompensa não-linear por redes neurais. Uma descrição breve dos métodos implementados segue abaixo:

- **Random** – política de recomendação aleatória. Escolhe a ação a partir de uma distribuição uniforme dos itens disponíveis.
- **PopularItem** – recomenda sempre o item mais popular até o momento. A popularidade do item é definida com base na recompensa acumulada para o item.
- ϵ**-Greedy** – política que consiste em escolher o melhor item com base na estimativa de recompensa com alguma alta probabilidade $(1 - \epsilon)$ ou um item aleatório com base na distribuição uniforme caso contrário.

- **LinTS** – generalização do algoritmo Thompson Sampling, que assume uma distribuição prévia de recompensa de cada ação ajustada a cada rodada, para o problema de *contextual bandits* com funções de compensação linear [13].
- **LinUCB** – utiliza uma função linear como estimador de recompensas e um limite superior que reduz a medida que a incerteza sobre a ação é reduzida [39].
- **NeuralUCB** – adaptação do LinUCB que utiliza uma rede neural como estimador de recompensa não-linear [177].
- **ExploreThenExploit** – política que alternar entre períodos de escolha de ações aleatórias e períodos de execução das ações com a maior recompensa esperada, estimada a partir do conhecimento adquirido na fase de exploração.
- **SoftmaxExplorer** – política que escolhe a ação a partir de uma distribuição de probabilidade proporcional, estimada a partir de uma função *softmax* da estimativa de recompensa esperada.
- **Adaptative-Greedy** – método similar ao ϵ-Greedy, mas em vez de utilizar um *threshold* fixo de probabilidade, utiliza um *threshold* da recompensa esperada com uma taxa de decaimento d. Escolhe o melhor item quando a estativa de recompensa está acima desse *threshold* ou uma escolha aleatória dos itens.
- **Percentile-Adaptive-Greedy** – política de exploração similar ao Adaptative-Greedy, mas em vez de utilizar on decaimento para reduzir um *threshold*, utiliza para reduzir o percentil das recompensas calculadas em uma janela móvel para definir o *threshold* da exploração.

A maioria dos agentes utiliza um oráculo, que faz estimativas de recompensa baseada no contexto e no item. Dessa forma, as estratégias de exploração usam essas estimativas para calcular probabilidades e escolher a ação a ser recomendada. Treinamos os agentes por meio de aprendizado *off-policy* em lote aplicando a função de custo CRM. O procedimento de treinamento aconteceu em épocas de um número fixo de interações, utilizando todos os *logs* adquiridos ao longo da simulação. Por fim, também realizamos o ajuste de hiperparâmetros em todos os agentes e apresentamos o melhor conjunto encontrado para cada um deles.

Foram realizados diferentes experimentos, utilizando as tarefas definidas em (4.1.3) e os modelos de *bandits* aqui definidos. Os detalhes dos experimentos e os seus resultados são expostos no capítulo (5).

4.2 *Neural Contextual Bandits* com restrições de justiça

Nesta seção, apresentamos a formulação do problema e as definições que serão utilizadas para descrever a modelagem do método proposto. O objetivo do *Neural Contextual Bandits* com restrições de justiça é modelar a definição de justiça apresentada e

balancear as recomendações para atingir um equilíbrio entre relevância e justiça no sistema sempre que possível.

Com o propósito de desenvolver um sistema de recomendação que equilibre a relevância do conteúdo recomendado para os usuários e a definição de justiça, que aqui utilizaremos a equidade na exposição dos fornecedores, propomos que o *Neural Contextual Bandits* possa ser adaptado de três formas diferentes para que o conceito de justiça seja inserido no sistema. A primeira é em relação a política de recomendação, onde iremos induzir uma exploração que beneficie o controle de justiça sob certas circunstâncias; a segunda é baseada na modelagem de recompensa, que contabiliza o ganho no balanceamento de justiça do sistema além da relevância da recomendação; a ultima é destina a propor um módulo de representação de *fairness* a ser utilizando no *oráculo*, habilitando o modelo a mapear melhor o estado atual do sistema.

Para avaliar os métodos propostos descrevemos diferentes formas de medir o ganho do sistema em relação à relevância e justiça.

4.2.1 Definição de Justiça

Nossa definição de justiça que utilizaremos nessa seção está centrada na exposição dos itens de um grupo predefinido. Esperamos que o sistema de recomendação possa equilibrar, até certo ponto, uma restrição de justiça ou taxa de exposição ideal dos itens que possuem um valor de atributo específico, independentemente de ser sensível ou não. Escolhemos a métrica de exposição dos provedores como fator principal na definição de justiça por ser relacionada aos interesses desses *stakeholders* em utilizar a plataforma.

Desse modo, cada item está associado a um grupo $G \in \{g_1, ..., g_l\}$, e $\mathcal{A}_g = \{a | G = g, a \in \mathcal{A}\}$ denota o grupo de itens com um valor de atributo g. Por exemplo, dada uma recomendação de hotéis, se o atributo protegido for "Quantidade de Estrelas", então $\mathcal{A}_g$ com g = "5"contém todos os hotéis com cinco estrelas. Usamos a exposição para definir a justiça entre diferentes grupos de itens, denotada por $x_t \in \mathbb{R}_+^l$, onde x_t^i representa a exposição do grupo i até o momento t.

$$x_t^i = \frac{\sum_{k=1}^{t} 1_{\mathcal{A}_{g_i}}(a_k)}{\sum_{i'=1}^{l} \sum_{k=1}^{t} 1_{\mathcal{A}_{g_{i'}}}(a_k)} \tag{4-1}$$

Em que, $1_A(x)$ é igual a 1 se $x \in A$, e 0 caso contrário. Como restrição de justiça ou exposição ideal, definimos pesos w_i para cada grupo de modo que $\sum_{i=1}^{l} w^i = 1$. Dessa forma, caso $(w^i - x^i)$ for negativo, o grupo i estaria super-representado, e caso positivo o grupo estaria sub-representado, encontrando o equilíbrio de justiça apenas se $\sum_{i=1}^{l}(w^i - x^i) = 0$.

Essa definição de justiça pode ser otimizada através do problema de otimização apresentado por Kelly, Maulloo e Tan, o *Weighted Proportional Fairness* (ou PropFair)

[79] é uma solução generalizada de Nash para vários grupos. Em que, uma alocação de atividades desejadas é ponderado proporcionalmente justo se for a solução do seguinte problema de otimização:

$$\max_{x_t} = \sum_{i=1}^{I} w_i \log(x_t^i), \qquad \text{s.t.} \sum_{i=1}^{I} x_t^i = 1, x_t^i \geq 0, i = 1, ..., I \tag{4-2}$$

O coeficiente $w_i \in \mathbb{R}_+$ é um parâmetro predefinido que pondera a importância de cada grupo. A solução ideal pode ser facilmente resolvido por métodos Multiplicadores de Lagrange,

$$x_*^i = \frac{w_i}{\sum_{i'=1}^{I} w_{i'}} \tag{4-3}$$

Desse modo, nas duas definições, o w_i controlaria o viés de exposição de cada grupo de itens no sistema de recomendação.

4.2.2 Afinidade do usuário a exploração

Outro conceito importante para definição desse trabalho é o de afinidade do usuário a exploração. Assumimos que os usuários têm vários graus de sensibilidade aos conteúdos recomendados, com alguns usuários interessados em um grupo ou tipo de conteúdo específico, enquanto outros são mais receptivos a exposição de outros fornecedores no momento da busca [111, 133], e que essa afinidade de exploração pode ser modelada a partir do histórico de iterações do usuário.

Dado um usuário u e seu histórico de itens interagidos positivamente $H_u = \{i_1, i_2, ..., i_m\}$, com $m \leq T$, onde i é a incorporação do item com o qual u interage (por exemplo, hotéis que pesquisou), nosso objetivo é definir a preferência do usuário u por itens semelhantes. A afinidade de exploração a novos itens do usuário é calculada com base na formulação:

$$\upsilon = 1 - \eta_{\zeta_u} \tag{4-4}$$

Em que, η_{ζ_u} é a média da similaridade de cosseno de cada item em H_u em relação ao centroide do espaço vetorial de itens para esse histórico de usuário, onde $\upsilon \in [0, 1]$. Uma afinidade de exploração do usuário igual a 0 implica que o usuário gosta de itens muito semelhantes e estaria pouco disposto a receber recomendações mais exploratórias; por outro lado, uma afinidade de exploração próximo de 1 não implicaria semelhanças entre os itens em H_u, indicando que o usuário tem um gosto mais diversificado e consequentemente estaria receptivo a receber recomendações diversificadas.

4.2.3 Trade-off entre relevância e justiça

Para balancear as necessidades dos usuários e dos fornecedores em um ambiente *multistakholder*, os sistemas de recomendação precisam encontrar um equilíbrio em termos da relevância do conteúdo recomendado para os clientes e sua justiça em termos de oportunidades de negócios para os fornecedores. Nesta seção, apresentamos uma série de políticas em que o sistema pode equilibrar a relevância e a justiça. Começamos considerando apenas a relevância e apenas a justiça como critério de otimização e, em seguida, apresentaremos outras políticas intercaladas.

Otimização por relevância

A política padrão de otimização por relevância considera apenas a estimativa de relevância de um conjunto para um determinado usuário fornecidas pelo *oráculo* do *Neural Contextual Bandit*. Dada uma um conjunto de itens a serem recomendados (S), um determinado usuário u e o contexto x, o item a ser recomendado é o que maximiza a estimativa de recompensa. Especificamente, nos definimos a *Relevance-Policy* como:

$$s_u^* = argmax_{s \in S} \ \Phi(u, s) \tag{4-5}$$

O $\Phi(.)$ representa a estimativa do *oráculo* para a tupla de usuário e contexto. O *oráculo* é treinado com os *feedbacks* fornecidos pelo usuário a partir das recomendações realizadas. Após o sistema realizar uma ação a_t, ou seja, recomendar um item ao usuário, o usuário interage com o ambiente e fornece o *feedback*. O sistema recebe uma recompensa imediata r_t de acordo com o *feedback* do usuário, com $r_t = 1$ significando que o usuário realizou uma interação com o item recomendado e $r_t = 0$ caso contrário. A recomendação de conteúdo relevante impacta positivamente a satisfação do usuário, mas esperamos maior injustiça para o conteúdo recomendado usando esta otimização e métrica como recompensa [111].

Otimização por justiça

A otimização por justiçam considera apenas o aspecto de balanceamento da exposição dos grupos previamente definidos afim de recomendar os itens que levem o sistema ao equilíbrio de justiça independente da satisfação do usuário. Para alcançar justiça de exposição entre os grupos de itens, tomamos o *PropFair* (Equação 4-2) como função de otimização. Dessa forma, dada uma um conjunto de itens a serem recomendados (S) e um parâmetro predefinido que pondera a importância de cada grupo ($w_g \in \mathbb{R}_+$), o item a ser recomendado é o que maximiza a estimativa de *PropFair*. Especificamente, nos definimos a *Fainess-Policy* como:

$$s_u^* = argmax_{s \in S} \ \Omega(s, w) \tag{4-6}$$

O $\Omega(.)$ representa a estimativa de *fairness* do sistema ao escolher o item s para ser exposto na recomendação, sua função não depende do usuário e por isso prevemos uma estimativa de relevância dos itens recomendados baixa, mas por outro lado equilibrando a exposição ideal dos itens dentro do sistema.

Combinando relevância e justiça por meio da afinidade de exploração

Partimos das definições de otimização por relevância e justiça para criar uma politica de recomendação interpolada, que considere conjuntamente a relevância de um conjunto para o usuário e o quanto o item contribui para a justiça do sistema balanceando essa importância por meio da afinidade do usuário a exploração (4-4). Espera-se que essa seja uma relação positiva, em que o sistema possa utilizar a propensão do usuário a explorar para equilibrar a exposição dos fornecedores sem prejudicar a relevância média esperada. Especificamente, nos definimos a *User-Affinity-Policy-I* como:

$$s_u^* = argmax_{s \in S} \ ((1 - \beta)\Phi(u, s) + \beta\Omega(s, w)) \tag{4-7}$$

Para um determinado usuário e conjunto, calculamos uma pontuação combinada por uma combinação ponderada de suas estimativas de relevância e justiça, em que $\beta = \upsilon\varphi$ e o parâmetro $\varphi \in [0, 1]$ sendo um parâmetro de controle dessa relação. Veremos o impacto desse parâmetro na seção 5.2.2.

Uma segunda adaptação dessa política de recomendação é usar a propensão do usuário a exploração como um *threshold* na decisão entre recomendação baseado em relevância ou justiça. Especificamente, nos definimos a *User-Affinity-Policy-II* como:

$$s_u^* = \begin{cases} argmax_{s \in S} & \Phi(u, s), \quad se \quad \upsilon < (1 - \varphi) \\ argmax_{s \in S} & \Omega(s, w), \quad se \quad \upsilon >= (1 - \varphi) \end{cases} \tag{4-8}$$

em que o parâmetro φ novamente é um parâmetro de controle dessa relação e define um *threshold* na propensão do usuário a exploração para definir se a política irá recomendar baseado na relevância ou na justiça. Espera-se que diferentes limiares possam apresentar resultados com diferentes balanceamentos. Veremos o impacto desse parâmetro na seção 5.2.2.

As duas políticas anteriores adaptam a política de recomendação e exploração do *Neural Contextual Bandits* a partir da definição de afinidade do usuário a exploração 4.2.2, mas mantêm a mesma modelagem de recompensa focada na relevância do item ao usuário. Em outras palavra, o *oráculo* do *Neural Contextual Bandits* continua sendo otimizado a partir da recompensa baseada na relevância do item para o usuário.

Combinando relevância e justiça por meio da recompensa

Modelos de *Multi-Armed Bandits* geralmente utilizam uma recompensa binária, onde $r = 1$ se o usuário realizar a interação com o item recomendado e $r = 0$ caso contrário. É possível partir do mesmo princípio de interpolação da relevância e da justiça para remodelar a função de recompensa de modo a induzir o modelo de *oráculo* à aprender o contexto ideal para recomendar itens baseados na relevância e/ou justiça enquanto mantém uma política de exploração não enviesada e particular de cada método.

No método proposta, a adaptação é realizada apenas na modelagem da função de recompensa, onde utilizaremos uma recompensa dupla que primeiro avalia a relevância do item pela interação do usuário, em seguida, avaliar o ganho de justiça no sistema em realizar a recomendação do item. Desse modo a função de recompensa passa a incorporar o sinal de justiça além do de relevância. Especificamente, nos definimos a *Reward-Policy* como:

$$r_t = (1 - \varphi)1_{\mathcal{A}_r}(a_t) + 1_{\mathcal{A}_g}(a_t) \tag{4-9}$$

Em que, $1_{\mathcal{A}_r}(x)$ é uma função indicadora que assume o valor de 1 se houve iteração com o item recomendado a_t, caso contrário é 0. Enquanto $1_{\mathcal{A}_g}(x)$ assume o valor de 1 se a exposição do item recomendando levar a melhoria no *PropFair* (Equação 4-2) do sistema, caso contrário é 0.

Combinando relevância e justiça por meio da representação de *fairness*

Uma componente fundamental do *Neural Contextual Bandits* são os extratores de *features*, que visam processar e extrair vetores de características com base nos dados que definem o contexto. Desse modo, propomos um módulo de representação do estado de *fairness* do sistema no *oráculo* para atender a três principais necessidades: (I) que o sistema de recomendação possa entender às relações de preferencia do usuário no contexto apresentado, (II) para que as informações de relação entre os grupos dos itens sejam melhor mapeadas nesse contexto e (III) para que o estado de justiça atual do sistema seja encodado de modo a ajudar o agente a promover os itens menos representados sem alterar a política de exploração.

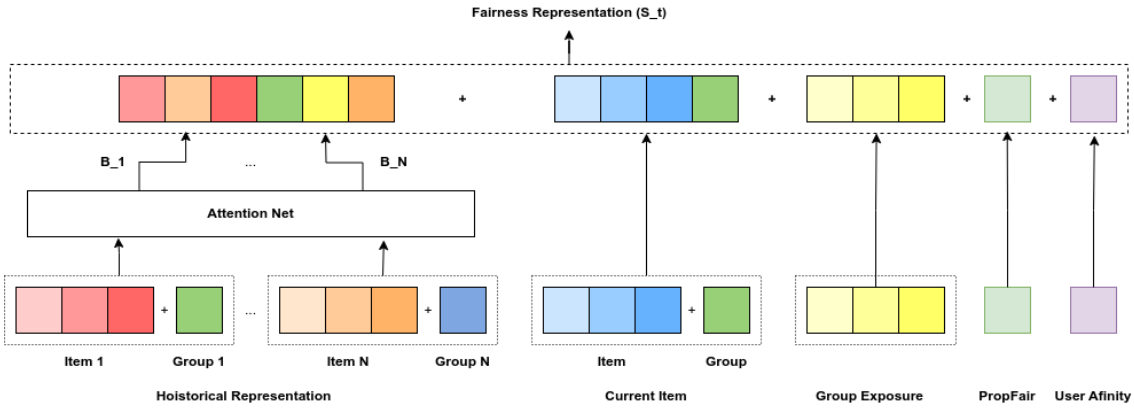


Figura 4.4: Modulo de Representação de Justiça

O módulo de representação do estado de justiça apresentado na Fig. 4.4 recebe os últimos N itens com interação positiva combinado com a codificação do grupo correspondente. Cada item a_i e g_i é mapeado e combinado em um vetor de *embedding* e_i. A seguir, definimos uma camada de atenção cujo objetivo é capturar dependências sequenciais entre os itens históricos, aprendendo um vetor de peso β de tamanho N, $\beta = \text{Softmax}(\omega^1 \sigma(\omega^2[e_1, ..., e_N] + b^2) + b^1$, onde ω^1, b^1, ω^2 , b^2 são os parâmetros de rede e $\sigma(.)$ é a função de ativação ReLU. O vetor de interação é obtido multiplicando os pesos de atenção pelas representações de itens correspondentes como $m_t = [\beta_1 e_1, ..., \beta_N e_N]$. Por fim, concatenamos essa representação com o *embedding* e_t do item candidato a ser recomendado, a diferença do vetor de exposição atual com a solução ideal (Equação (4-3)) x_t, o *PropFair* (Equação 4-2) do sistema p_t e a afinidade do usuário a exploração υ_u (Equação 4-4) para formar a representação do estado de justiça do sistema em um determinado tempo t.

$$s_t = [m_t || e_t || x_t || p_t || \upsilon_u] \tag{4-10}$$

Essa representação é utilizada como *feature* adicional de contexto para o *Neural Contextual Bandits* de modo a direcionar o aprendizado do modelo. E assim como a anterior, espera-se induzir o modelo a recomendar itens baseados na relevância e/ou justiça enquanto mantém uma política de exploração não enviesada. Esse módulo em conjunto com a modelagem de recompensa apresentado a seção anterior define o método *Fair-Feature-Policy*.

4.2.4 Designer de Experimentos

Avaliaremos as cinco políticas de recomendação propostas e o efeito no controle de justiça utilizando o MARS-Gym como simulador e o Trivago Marketplace Dataset (4.1.3) como conjunto de dados. Espera-se comparar os métodos em termos de acurácia

e controle de justiça baseado em grupos de itens que compartilham da mesma *feature* sensível.

Dataset

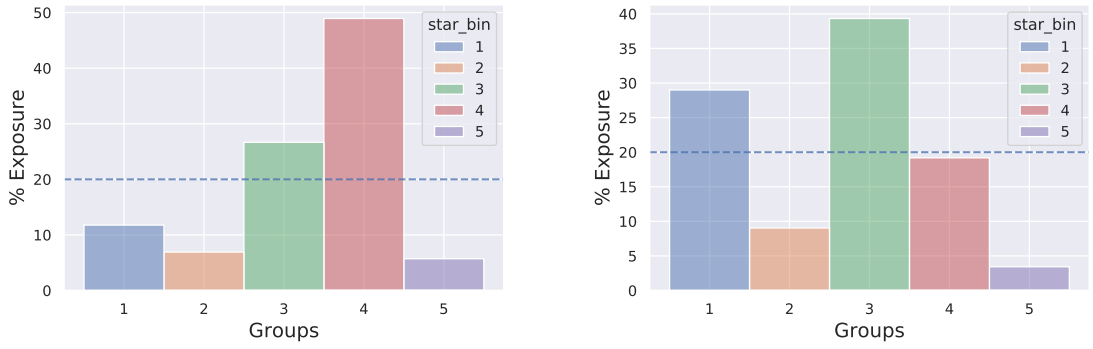
No Trivago Marketplace Dataset (4.1.3) os itens podem ser caracterizados e agrupados a partir das *features* de "estrelas", que assumem os valores de ‘1 star‘, ‘2 star‘, ‘3 stars‘, ‘4 star‘ e ‘5 star‘, e representam a avaliação dos hotéis e hospedagens disponíveis para recomendação. Utilizaremos essa *feature* para criar 5 grupos distintos em que cada item pertence exclusivamente a um grupo. A tabela 4.2 apresenta a distribuição dos itens em cada grupo para cada tarefa do conjunto de dados.

Tabela 4.2: Trivago marketplace – distribuição dos itens entre os grupos no conjunto de dados para a *feature* "estrelas"

Tarefa	Itens	1 star	2 star	3 star	4 star	5 star
RecSys Cities	12.358	66,7%	6,4%	13,4%	11,2%	2,3%
New York, USA	2.036	58,9%	9,6%	16,3%	12,8%	2,4%
Rio de Janeiro, Brazil	2.192	79,4%	5,2%	10,12%	4,1%	1,2%
Chicago, USA	670	50,0%	12,0%	21,3%	14,3%	2,4%
Como, Italy	343	68,2%	2,3%	14,0%	14,0%	1,5%

A distribuição da exposição dos grupos baseado na lista de itens disponíveis para recomendação podem não seguir a mesma distribuição apresentada na tabela 4.2 devido à característica do dataset ser voltado para tarefas de re-ranking, em que os itens recomendáveis no tempo t não seguem uma distribuição aleatória do total de itens. Mas, ainda assim, apresenta uma grande disparidade na exposição dos itens recomendáveis.

Figura 4.5: Distribuição da exposição dos itens da lista disponível para recomendação na tarefa "Chicago, USA"e "Rio de Janeiro, Brazil"



(a) Chicago, USA

(b) Rio de Janeiro, Brazil

Por fim, neste trabalho, utilizaremos como coeficientes de exposição ideal da recomendação uma exposição igualitária entre os grupos de 20%. A Fig. 4.5 exibe a exposição esperada em uma recomendação aleatória (sem o viés do sistema de recomendação) e o limiar da exposição ideal utilizada nesse trabalho.

Métricas de Avaliação

Avaliaremos as políticas de recomendação com base nas simulações realizadas pelo MARS-Gym utilizando três principais métricas, são elas:

Cumulative Mean Reward, onde avaliaremos a relevância das recomendações para o usuário. Definida por:

$$CMR = \frac{\sum_{k=1}^{T} y_{ak}}{T}, \tag{4-11}$$

em que T é a iteração máxima da simulação e y_{ak} é a recompensa da recomendação no tempo k. É esperado que os modelos com foco em acurácia das recomendações para o usuário apresentem métricas maiores de CMR.

PropFair, onde avaliaremos o equilíbrio de justiça do sistema e é definida por:

$$PropFair = \sum_{i=1}^{I} w_i \log(1 + x_T^i), \tag{4-12}$$

em que W define a importância de cada grupo, que aqui definiremos como igualitária tomando o valor de $W = \{0.2, 0.2, 0.2, 0.2, 0.2\}$ e x_T é a exposição dos grupos no tempo T defido em 4-1. É esperado que os modelos com foco em equilíbrio da exposição apresentem métricas maiores de *PropFair*.

E por último a *Unit Fairness Gain* (**UFG**) [97], onde avaliaremos o equilíbrio entre relevância e justiça.

$$UFG = \frac{PropFair}{CMR_{max} - CMR} = \frac{PropFair}{1 - CMR}, \tag{4-13}$$

A métrica UFG pode ser interpretado como o ganho de justiça *versus* a relevância das recomendações. Ou seja, o quanto o sistema sacrifica em termos de relevância para otimizar o equilíbrio do sistema. Um UFG maior indica uma relação melhor entre relevância e justiça.

4.3 Considerações finais

Neste capítulo, foram apresentados detalhes de implementação de cada etapa do projeto. Inicialmente, detalhamos o desenvolvimento do MARS-Gym, um simulador de sistemas de recomendação com foco em ambientes *multistakeholders* produtivos

ao modelar e utilizar *logs* de interações reais, bem como o *designer* de experimentos realizados. Para os experimentos foram definidos cinco cenários de recomendação com diferentes características para simulação pelo *framework*. A escolha do *dataset* e dos cenários se deu principalmente pela sua natureza produtiva e similaridade com os dados privados que avaliamos junto aos projetos de P&D com as empresas iFood [133], Rurax, Moblix e BettrAds.

Também, foi detalhada a modelagem de justiça utilizada no método proposto de *Neural Contextual Bandits* com restrições de justiça, bem como as métricas utilizadas para sua avaliação.

CAPÍTULO 5

Resultados

Nesse capítulo, são apresentados os resultados obtidos experimentalmente utilizando o MARS-Gym como simulador de ambiente *multistackholder* com base em um *dataset* com interações reais. Apresentamos também, os resultados de cada *bandit* implementado como modelo *baseline* e discutimos os resultados em diferentes perspectivavas. A primeira é a simulação e como esse ambiente pode ser útil para validar novos algoritmos, a segunda perspectiva é a de recomendação e avaliação *off-policy* e por fim a perspectiva de justiça nos algoritmos de recomendação em que apresentaremos os resultados obtidos com os experimentos do *Neural Contextual Bandits* com restrições de justiça e as políticas propostas, onde iremos comparar os métodos e a sensibilidade da parametrização na otimização multiobjetiva entre relevância e justiça.

Inicia-se este capítulo com a seção (5.1.1), apresentando detalhes dos resultados obtidos a partir da simulação dos modelos utilizando o MARS-Gym. Na seção (5.1.2), discutem-se resultados com base em uma perspectiva de sistemas de recomendação e como as abordagens de aprendizado e avaliação *off-policy* impactam nos resultados. Em seguida, na seção (5.1.3) realiza-se uma análise dos resultados de justiça no cenário geral. Por fim, na seção (5.2), apresentamos os resultados obtidos a partir da simulação utilizando o método de *Neural Contextual Bandits* com restrições de justiça e as políticas propostas para esse método.

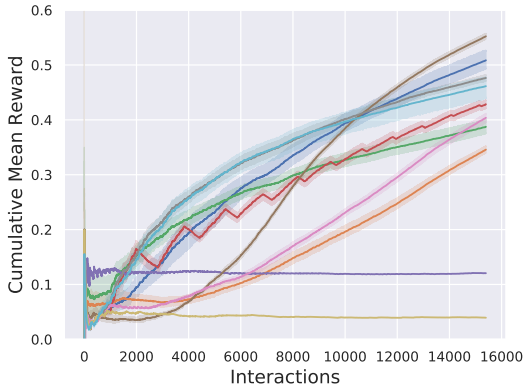
5.1 Resultados da simulação com o MARS-Gym

5.1.1 Resultados da simulação

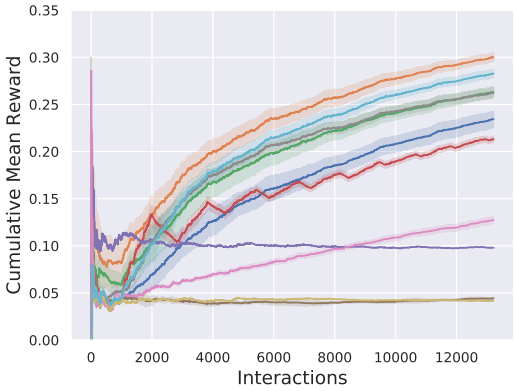
Realizamos a simulação de cada tarefa proposta até a convergência da maioria dos métodos para observar a recompensa média acumulada ao longo da simulação. É essencial entender o comportamento dos *bandits* e fazer uma comparação entre eles. Para significância estatística, representamos cada curva pela média e intervalo de confiança em cinco execuções do mesmo experimento. Na Fig. 5.1, apresentam-se os resultados das simulações para diferentes tarefas e métodos.

Figura 5.1: resultados da simulação dos *bandits*

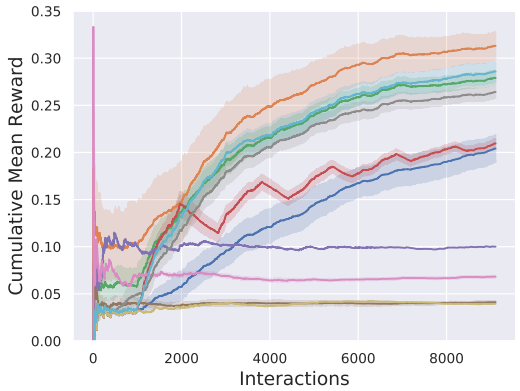
(a) Como, Italy



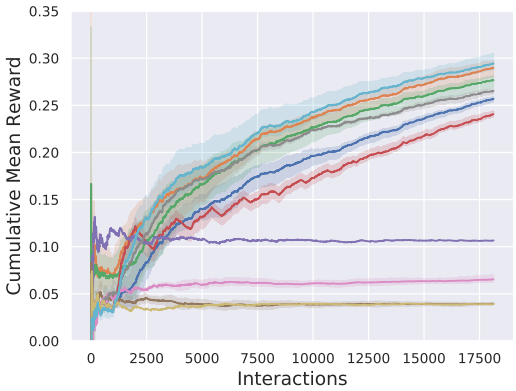
(b) Chicago, USA



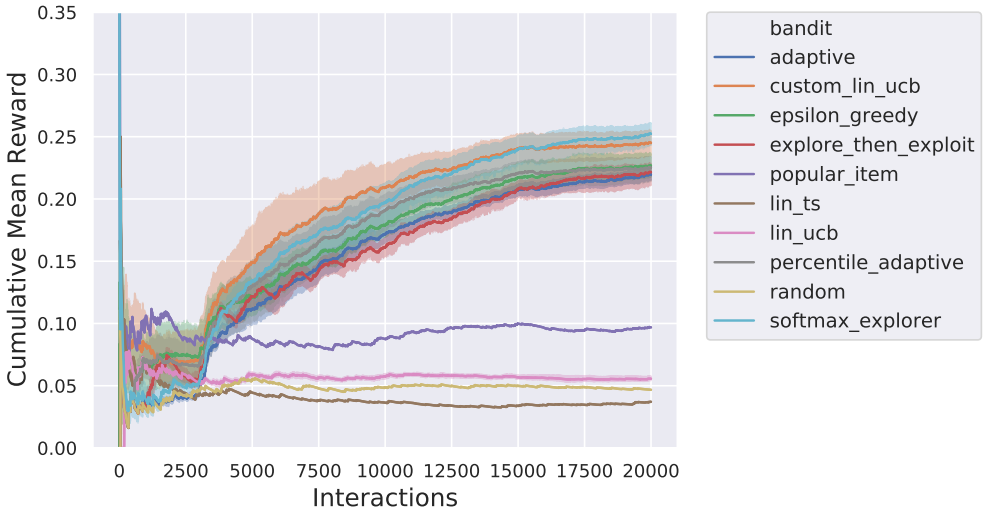
(c) Rio de Janeiro, Brazil



(d) New York, USA



(e) RecSys Cities



Em geral, os métodos começam com uma recompensa média abaixo do agente *PopularItem*, que em todos os experimentos se mantem constante ao longo das interações por não haver aprendizado nesse método. À medida que exploram novas recomendações e contextos, cada método apresenta um caminho de aprendizado diferente, mas a maioria deles ultrapassa esse *baseline*. Além disso, o desempenho desses métodos variam drasticamente entre as tarefas, o que reforça que não há melhor estratégia de exploração para todos os cenários. De fato, seu desempenho está intrinsecamente relacionado às restrições do PDM.

No entanto, os experimentos também mostram alguns padrões entre as tarefas. Observamos que métodos lineares como *LinTS* e *LinUCB* obtêm melhores resultados quando o espaço de busca é menor, mas não escalam bem à medida que a dimensionalidade do PDM aumenta. Por outro lado, o *NeuralUCB* apresenta melhores resultados nesses cenários, mas tem pior eficiência amostral no cenário de poucos dados. Nossa hipótese é que a introdução de não linearidades no oráculo aumentam a capacidade de representação de informações contextuais da política, à custa de um problema de otimização mais difícil.

É importante destacar a curva do agente *ExploreThenExploit*, que intercala localmente picos e vales à medida que o método muda de exploração total para intensificação total. Sugere uma sensibilidade na troca dos hiperparâmetros, o que é indesejável para ambientes de produção. Por fim, também apontamos que métodos simples (em uma perspectiva de implementação) como ϵ-*Greedy* e *SoftmaxExplorer*, apresentam resultados satisfatórios, sugerindo ser ideais para compor agentes e algoritmos mais complexos como configuração inicial.

5.1.2 Métricas de recomendação e avaliação *off-policy*

Avaliamos os *bandits* de acordo com métricas tradicionais de recomendação e métricas *off-policy* no subconjunto de teste da tarefa "Chicago, EUA". Na tabela 5.1, apresenta os resultados médios em cinco execuções.

Em primeiro lugar, confirmamos a hipótese de que a métrica de precisão está altamente correlacionada com os resultados das simulações na Fig. 5.1(b). O *NeuralUCB* apresentou os melhores resultados na tarefa, assim como na simulação. Comparativamente, o *AdaptiveGreedy* e *ExploreThenExploit* tiveram um desempenho melhor na avaliação do que na simulação, apresentando uma precisão de 0.331 e 0.315, ao contrário do ϵ-*Greedy* que obteve uma precisão de 0.311 mas apresentou melhores resultados que os anteriores na simulação. Nossa hipótese é que essas mudanças estão relacionadas à eficiência amostral de cada estratégia de exploração, bem como às propriedades de generalização das políticas aprendidas. Além disso, observamos que o *SoftmaxExplorer* supera

Tabela 5.1: Métricas de recomendação para a tarefa "Chicago, EUA"

	Métricas de recomendação clássicas				Avaliação *Off-Policy*		
	Precision	NDCG@5	Cove@5	Per@5	IPS	DE	DR
NeuralUCB	**0.338**	0.443	0.371	0.724	**0.324**	0.180	**0.299**
Adaptive-Greedy	0.331	0.415	0.379	0.770	0.314	**0.181**	0.291
P. Adaptive-Greedy	0.319	0.426	0.364	0.750	0.306	0.171	0.279
SoftmaxExplorer	0.316	**0.446**	0.328	0.727	0.308	0.171	0.281
ExploreThenExploit	0.315	0.423	0.313	0.737	0.308	0.167	0.280
ϵ-Greedy	0.311	0.441	0.343	0.737	0.305	0.165	0.278
LinUCB	0.065	0.183	0.297	0.721	0.047	0.033	0.043
LinTS	0.029	0.112	**0.419**	**0.778**	0.031	0.018	0.031
PopularItem	0.074	0.199	0.153	0.592	0.063	0.046	0.061
Random	0.042	0.145	0.392	0.777	0.028	0.024	0.029
Policy without CRM	**0.354**	**0.457**	**0.303**	0.717	0.340	**0.199**	0.312
Policy with CRM	0.344	0.434	0.300	**0.730**	**0.350**	0.195	**0.320**

todos os outros métodos na métrica nDCG, o que sugere um bom desempenho em cenários de recomendação de listas. Por fim, destacamos o desempenho do *AdaptiveGreedy* para métricas de diversidade (Cobertura e Personalização), mantendo boas métricas de precisão.

Observamos uma queda consistente da precisão para as métricas *off-policy*, sugerindo que todos os métodos exploram naturalmente o viés de amostragem. Comparativamente, vemos uma alta correlação entre essas métricas. No entanto, a classificação dos *bandits* não é a mesma, o que sugere que alguns *bandits* sofrem mais com o viés de amostragem. Essas evidências e análises são importantes para a implantação do modelo, pois, as métricas *off-policy* geralmente se correlacionam melhor com os experimentos *on-line* [74, 75].

Por fim, realizamos um estudo ablativo sobre a função de custo CRM 3-6. Para isso, treinamos duas políticas usando os dados de treinamento na configuração *full offline* (ou seja, usando diretamente os *logs* de interação para otimização, sem simulação de PDM), variando a função de custo. Na Tabela 5.1, observamos que a política com CRM (Policy with CRM) troca a precisão nas métricas de recomendação (Precision e NDCG@5) para melhorar as métricas *off-policy* (IPS e DR). Esse resultado valida que essa técnica é essencial para reduzir o efeito do viés de amostragem durante o treinamento e, portanto, melhorar a avaliação *off-policy*.

5.1.3 Resultados de justiça

Avaliamos o *bandit* do *SoftmaxExplorer* na tarefa "RecSys Cities", na perspectiva de maus tratos díspares 2-9 (Fig. 5.2(a), 5.2(b), 5.2(c) e 5.2(d)) e tratamento díspar 2-9 (Figura 5.2(e)). Selecionamos alguns atributos que consideramos sensíveis para todos os parceiros do *marketplace*.

Na Fig. 5.2(a), apresentamos taxas de verdadeiro positivo relacionada ao atributo de acessibilidade, que é um requisito importante para um sub-grupo de usuários que acessam a plataforma. Observamos que o modelo oferece recomendações mais precárias para acomodações com acessibilidade do que aquelas que não sugerem maus tratos díspares, o que é indesejável para a experiência dos usuários que exige esse recurso. Na Fig. 5.2(b), por outro lado, apresentam-se características relacionadas ao tipo de acomodação como hotéis, hostels e apartamentos. Observamos que os hotéis recebem uma recomendação muito melhor do agente do que os outros tipos, como hostels ou apartamentos. Da mesma forma, na Fig. 5.2(c), também apresentam-se maus tratos díspares para acomodações que são familiares ou oferecem creche. Esses atributos além de serem características do negócio dos fornecedores, são também os requisitos dos usuários ao buscar algo na plataforma, o que impacta a experiência tanto do usuário quanto dos fornecedores.

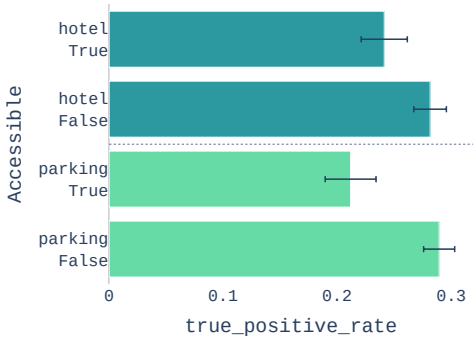
Em outra perspectiva, na Fig. 5.2(d), mostram-se as taxas de verdadeiros positivos para cada cidade analisada. Em geral, vemos resultados semelhantes para todas elas. As métricas para algumas cidades apresentam maiores intervalos de confiança, o que está diretamente relacionado a menor quantidade de interações disponíveis para ajustar o modelo. No entanto, diagnosticamos diferenças entre algumas cidades (por exemplo, comparando-se Nova York e Dublin), que poderíamos explorar para entender a origem da injustiça e melhorar o sistema de recomendação.

Por fim, na Fig. 5.2(e) apresentam-se as estimativas do mesmo modelo para cinco hotéis diferentes agrupados pelo dispositivo do usuário. Embora muitos deles apresentem o mesmo tratamento para todos os dispositivos, verificamos que o hotel com *ID 371* apresenta pontuações consideravelmente baixas para o dispositivo do tipo *desktop*. Nossa hipótese é que isso esteja relacionado à experiência do usuário usando esse dispositivo. Este é um caso em que surge um tratamento díspar, e a plataforma deve abordá-lo para garantir que o comportamento do modelo não prejudique sub-grupos de fornecedores por algum atributo não-sensível.

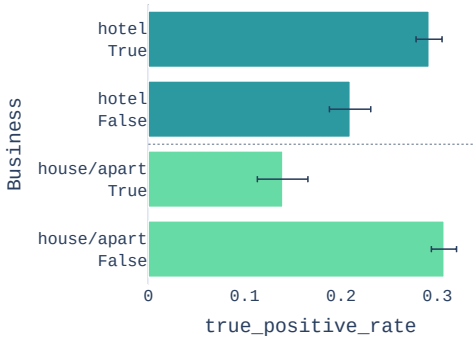
Em última análise, todos os gráficos na Fig. 5.2 diagnosticam diferentes cenários em que o sistema de recomendação não satisfaz as noções de justiça definidas a priori. Eles são a reprodução de muitos vieses no conjunto de dados. O pipeline e a plataforma de aprendizado de máquina devem abordá-los para garantir uma experiência justa a todos os *stakeholders* da plataforma. Por exemplo, podemos alterar a interface do usuário para

Figura 5.2: Análise de justiça para o *bandit SoftmaxExplorer*

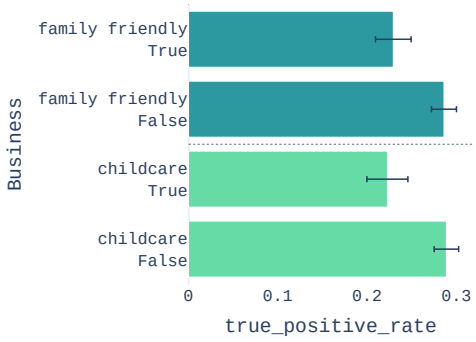
(a) Attributo: Acessibilidade



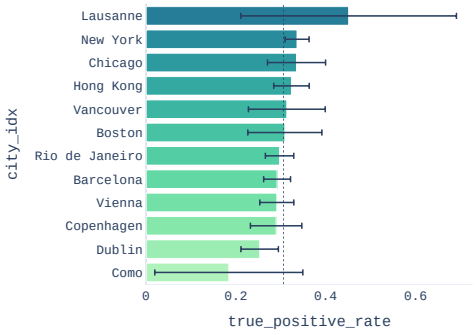
(b) Atributo: Tipo de Acomodação



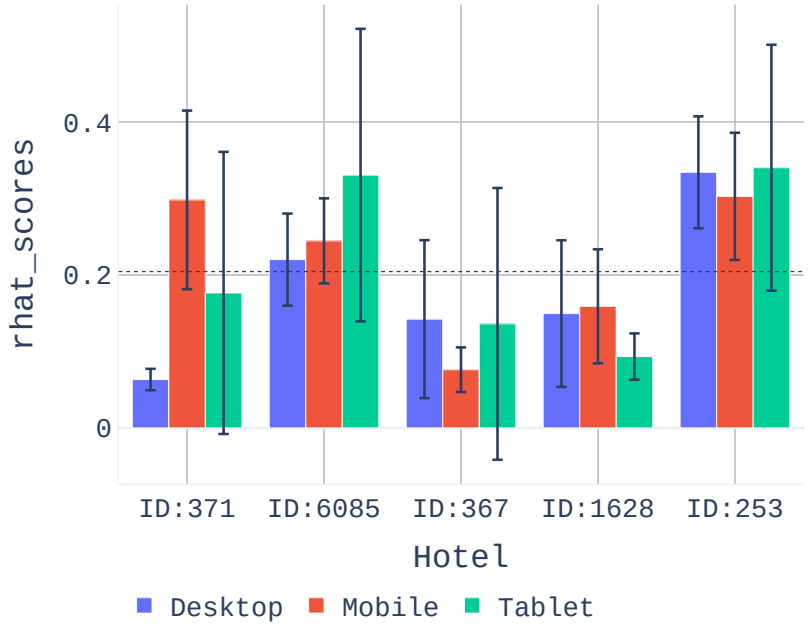
(c) Atributo: Afinidade do Negócio



(d) Atributo: Cidade



(e) Atributo: Dispositivo



tablet na plataforma ou balancear o conjunto de dados para as diferentes acomodações. De qualquer forma, essas métricas fornecem *insights* sobre onde o sistema precisa melhorar para manter a justiça e a integridade do *marketplace*.

5.2 Resultados do *Neural Contextual Bandits* com restrições de justiça

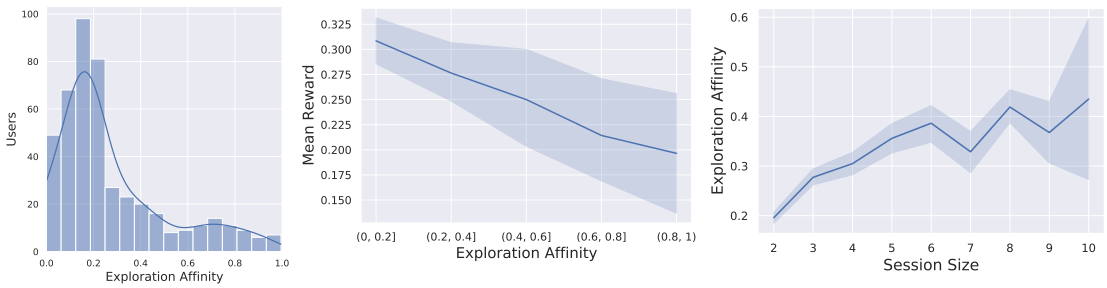
Nessa seção são apresentados os resultados das políticas propostas para o controle de justiça utilizando o *Neural Contextual Bandit*. Inicia-se a seção (5.2.1) apresentando os resultados da modelagem de afinidade do usuário a exploração e como essa métrica está relacionada com a relevância das recomendações, em seguida na seção (5.2.2) discutiremos sobre a parametrização do controle de justiça em cada política proposta, por fim, nas seções (5.2.3) e (5.2.4) realiza-se uma análise dos resultados do método proposto.

5.2.1 Influencia da afinidade do usuário a exploração na relevância

Ao avaliar a métrica de afinidade do usuário a exploração, o mesmo padrão apresentado na Fig. 5.3 aparece em todos as tarefas e *bandits* treinados. Em primeiro lugar, confirmamos a hipótese de que diferentes usuários podem apresentar diferentes níveis de afinidade a exploração e essa distribuição pode variar a depender do dataset (vide Apêndice A). Na Fig. 5.3(b), apresentamos a relação dessa métrica com a performance do modelo, onde observamos uma relação inversa que evidencia que mesmo um modelo otimizado para relevância perde acurácia a medida que o usuário está mais aberto a explorar novos conteúdos.

Figura 5.3: análise da afinidade do usuário a exploração baseado no tarefa "Chicago, USA"e no *bandit* NeuralUCB

(a) Distribuição da afinidade de exploração (b) Relação com a performance do *bandit* (c) Relação com o tamanho do histórico de interações do usuário



Em outra perspectiva, na Fig. 5.3(c), apresentamos a relação dessa métrica com o tamanho do histórico de interações do usuário, onde fica evidente que quanto maior esse

histórico é esperado que o usuário esteja mais aberto a novos conteúdos, possivelmente evidenciando que ainda não encontrou o conteúdo que está buscando no sistema. O que de certa forma, reforça a hipótese principal de que mesmo com mais dados disponíveis para otimizar o modelo baseado em relevância a depender da afinidade do usuário a exploração esse acréscimo não é refletido em uma melhor recomendação, sendo mais útil ao sistema como um todo otimizar outra métrica além da relevância para esses casos.

5.2.2 Efeito do parâmetro de controle φ nos métodos de justiça

O parâmetro φ foi introduzido em cada método que interpola políticas de relevância e justiça com o objetivo de ser um parâmetro de *tuning* que controla o peso a ser dado à política de justiça utilizada em cada método. Na Fig. 5.4, apresentam-se os resultados da simulação variando o φ em cada método em questão.

Em geral, os métodos respondem como esperado ao parâmetro φ, em que obtêm-se uma relação inversa com a relevância enquanto observa-se uma relação positiva na otimização de justiça do sistema. Observamos também que para os métodos *Reward-Policy*, *Fair-Feature-Policy* e *User-Affinity-Policy-II* o PropFair converge para seu ponto máximo com $\varphi = 0.2$ e $\varphi = 0.4$, não obtendo ganho adicional na justiça a partir desse valor mas reduzindo a performance do modelo de forma linear. Por fim, para o método *User-Affinity-Policy-I* o comportamento diverge dos demais devido a sua formulação ser baseada na relevância do parâmetro de afinidade do usuário a exploração e não necessariamente no peso da política de justiça, desde modo, é observado um ganho na justiça do sistema enquanto a métrica de acurácia relacionada a relevância das recomendações mantém-se estável com o acréscimo do φ até $\varphi < 0.8$ o que reforça a utilidade da modelagem de afinidade de exploração do usuário em contrabalancear esses objetivos.

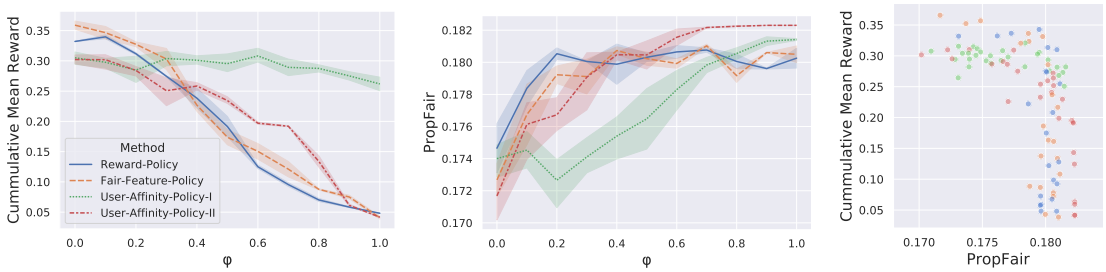


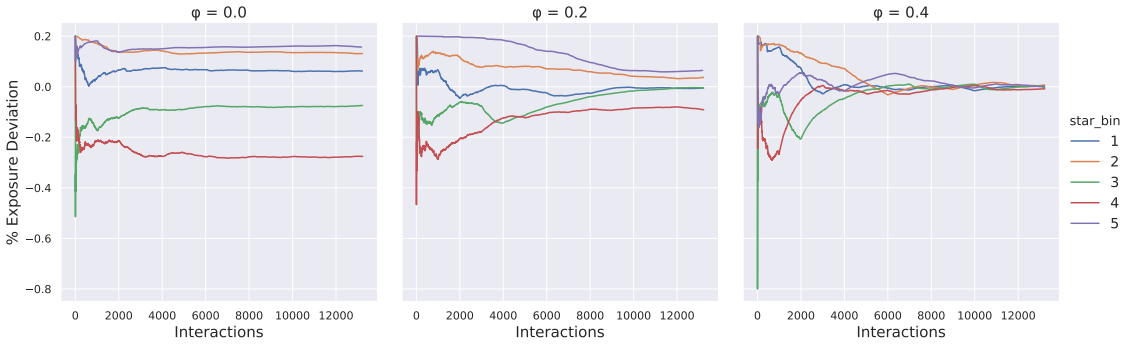
Figura 5.4: Tuning do parâmetro φ dos métodos que interpolam as políticas de relevância e justiça na tarefa [Chicago, USA]

Em última análise, avaliamos o método *Fair-Feature-Policy* em termo de exposição ideal esperada para diferentes valores de φ (Fig. 5.5). Na Fig. 5.5(a), apresentamos o desvio da exposição ideal e fica evidente que para $\varphi = 0$ o método mantém uma exposição constante para os grupos seguindo a mesma distribuição apresentada no dataset

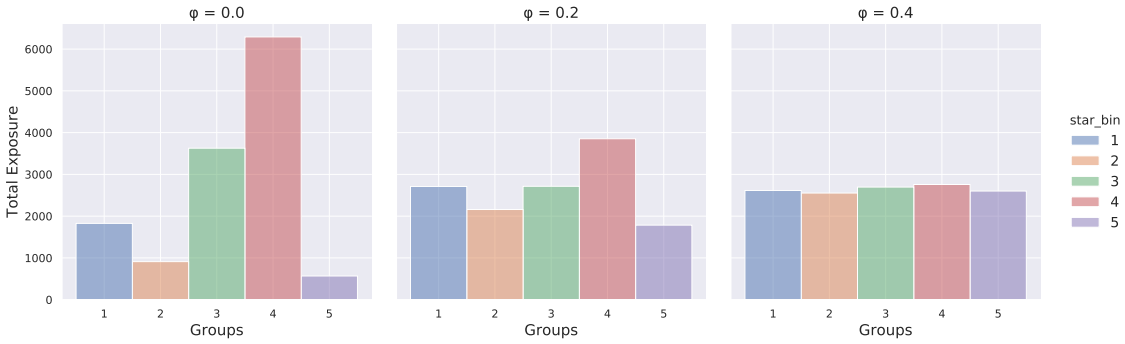
original (Fig. 5.5(b)), por outro lado, para $\varphi = 0.4$ o método converge para a solução ideal de exposição mantendo a justiça no sistema sob controle a partir desse ponto.

Figura 5.5: resultados da simulação de controle de justiça do *Fair-Feature-Policy* para diferentes valores de φ na tarefa "Chicago, USA"

(a) Simulação do desvio de exposição ideal dos grupos para diferentes valores de φ



(b) Exposição total dos itens de cada grupo para diferentes valores de φ



Por fim, o valor de φ ideal de cada método foi escolhido a partir dos experimentos que compõem a Fronteira de Pareto (Fig. (c) 5.4) entre as métricas *Cumulative Mean Reward* e PropFair que maximizam a UFG. Dessa forma, entende-se que cada política de recomendação a ser avaliado otimize o mesmo objetivo final a nível de parametrização.

5.2.3 Avaliação das políticas de recomendação com restrição de justiça

Avaliamos as seis políticas de recomendação com restrição de justiça e dois *bandits* baselines nas tarefas "Chicago, USA"e "RecSys Cities"para medir o ganho do sistema em relação à relevância e justiça. Na Tabela 5.2, apresentamos os resultados médio para cinco execuções da simulação em sua melhor parametrização do φ.

Em primeiro lugar, confirmamos a hipótese de que uma política focada apenas na relevância como a *Relevance-Policy* ou mesmo a *NeuralUCB*, que aqui representa o *Neural Contextual Bandit* com melhor resultado nas simulações anteriores, apresentam os

piores resultados do *PropFair*, assemelhando-se a uma política randômica como o *Random* que apresenta 0.174 (Chicago, USA) e 0.179 (Rio de Janeiro, Brazil). Por outro lado, uma política focada apenas na justiça como o *Fairness-Policy* embora apresente métricas de *PropFair* superiores de 0.182 (Chicago, USA e Rio de Janeiro, Brazil) também apresentem métricas de relevância similares a política *Random* de aproximadamente 0.040 de *CRM*, reforçando a hipótese de que relevância e justiça são conceitos conflitantes em termos de otimização.

Tabela 5.2: Métricas de relevância e justiça para as tarefas "Chicago, EUA"e "Rio de Janeiro, Brazil"

	Chicago, USA		
	CMR	PropFair	UFG
Relevance-Policy	$0.313_{\pm 0.012}$	$0.174_{\pm 0.001}$	$0.253_{\pm 0.005}$
Fainess-Policy	$0.041_{\pm 0.002}$	$\mathbf{0.182}_{\pm 0.000}$	$0.190_{\pm 0.000}$
User-Affinity-Policy-I ($\varphi = 0.6$)	$0.288_{\pm 0.005}$	$0.179_{\pm 0.000}$	$0.252_{\pm 0.001}$
User-Affinity-Policy-II ($\varphi = 0.1$)	$0.301_{\pm 0.011}$	$0.176_{\pm 0.001}$	$0.252_{\pm 0.003}$
Reward-Policy ($\varphi = 0.2$)	$0.319_{\pm 0.005}$	$0.179_{\pm 0.001}$	$0.263_{\pm 0.003}$
Fair-Feature-Policy ($\varphi = 0.2$)	$0.325_{\pm 0.008}$	$0.180_{\pm 0.001}$	$\mathbf{0.266}_{\pm 0.004}$
NeuralUCB	$\mathbf{0.335}_{\pm 0.012}$	$0.174_{\pm 0.001}$	$0.262_{\pm 0.006}$
Random	$0.044_{\pm 0.001}$	$0.174_{\pm 0.000}$	$0.182_{\pm 0.000}$
	Rio de Janeiro, Brazil		
	CMR	PropFair	UFG
Relevance-Policy	$0.301_{\pm 0.015}$	$0.178_{\pm 0.001}$	$0.256_{\pm 0.015}$
Fainess-Policy	$0.044_{\pm 0.001}$	$\mathbf{0.182}_{\pm 0.001}$	$0.191_{\pm 0.000}$
User-Affinity-Policy-I ($\varphi = 0.4$)	$0.312_{\pm 0.010}$	$0.181_{\pm 0.001}$	$0.263_{\pm 0.004}$
User-Affinity-Policy-II ($\varphi = 0.2$)	$0.258_{\pm 0.017}$	$0.181_{\pm 0.001}$	$0.243_{\pm 0.005}$
Reward-Policy ($\varphi = 0.1$)	$\mathbf{0.331}_{\pm 0.006}$	$0.179_{\pm 0.002}$	$0.267_{\pm 0.004}$
Fair-Feature-Policy ($\varphi = 0.1$)	$0.329_{\pm 0.012}$	$0.180_{\pm 0.001}$	$\mathbf{0.268}_{\pm 0.006}$
NeuralUCB	$0.326_{\pm 0.007}$	$0.177_{\pm 0.002}$	$0.262_{\pm 0.004}$
Random	$0.042_{\pm 0.000}$	$0.179_{\pm 0.000}$	$0.187_{\pm 0.000}$

Em geral, as políticas *User-Affinity-Policy-I*, *User-Affinity-Policy-II*, *Reward-Policy* e *Fair-Feature-Policy* com restrição de justiça apresentam um PropFair maior do que os demais, indicando a utilidade do controle de justiça implementado na política. Observamos que a política que melhor equilibra a relevância e justiça é a *Fair-Feature-Policy*, em que apresenta os melhores resultados em termos de UFG, enquanto mantém a CMR similar ao *NeuralUCB* e um PropFair superior. Com relação aos métodos *User-Affinity-Policy I e II* nossa hipótese é que apenas uma política de recomendação que beneficie a exploração com viés de controle de justiça não é suficiente para manter o

equilíbrio entre relevância e justiça sem perdas, os resultados de UFG desses métodos serem abaixo do método *NeuralUCB* reforçam que o ganho em justiça em utiliza-los não compensaria a perda na relevância. O mesmo não acontece para os métodos *Reward-Policy* e *Fair-Feature-Policy*, o que reforça a necessidade da modelagem de recompensa dupla para contabilizar o ganho em manter o equilíbrio da justiça do sistema otimizando o *oráculo* do *Neural Contextual Bandit* com ambos os sinais.

Em ultima análise, realizamos um estudo ablativo sobre o método *Fair-Feature-Policy*. É possível observar que o ganho apresentado pelo *Fair-Feature-Policy*, onde implementamos o módulo de representação de *fairness*, em comparação com o *Reward-Policy* é em termos de ganho tanto em relevância das recomendações quanto em justiça, o que se reflete na melhoria de todas as métricas. Esse resultado reforça a hipótese de que uma melhor representação do estado de justiça atual do sistema no *oráculo* leva de fato a melhores recomendações como um todo, induzindo o *oráculo* a tomar melhores decisões em termos de relevância e justiça sem alterar a política de exploração do *bandit* ou a definição de recompensa. Esse resultado valida a utilidade do módulo não apenas para o controle de justiça, mas também para a melhoria da relevância das recomendações.

Por fim, é importante destacar que o método *Fair-Feature-Policy* pode ser adaptado a qualquer outro *Neural Contextual Bandit* ao utilizar o módulo de representação do estado de *fainess* como *features* do *oráculo* e a modelagem multiobjetiva da recompensa, mantendo as características da política de exploração do *bandit*.

5.2.4 Avaliação dos métodos *baselines* com adaptação da restrição de justiça

Realizamos a simulação de cada tarefa proposta nos *bandits baselines* em sua versão adaptada com o método *Fair-Feature-Policy* para observar a convergência e o impacto nas métricas de relevância e justiça. Na tabela B.1, apresentamos os resultados em termos de ganho em relevância e justiça comparados as versões *vanilla* do mesmo método.

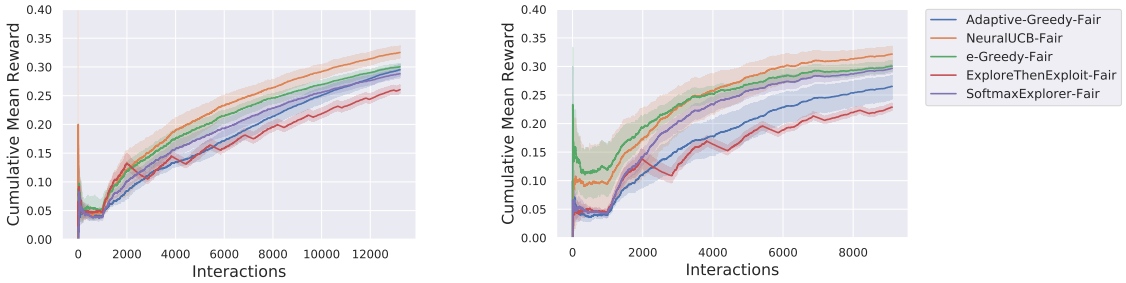
Em primeiro lugar, confirmamos a hipótese de que a convergência dos métodos e a particularidade da política de exploração não foi alterada pela inclusão do módulo de representação do estado de *fainess* e da modelagem multiobjetiva da recompensa, características essas necessárias para utilização da política *Fair-Feature-Policy*. Na Fig. 5.6 apresentam-se as convergências dos *bandits* com restrição de justiça e é possível observar o mesmo comportamento que a versão *vanilla* apresentada na seção 5.1.1 na Fig 5.1.

Na Tabela B.1, apresentamos o ganho relativo em relevância (CMR) e justiça (PropFair) dos *bandits* com controle de justiça em relação a sua versão *vanilla* para as

Figura 5.6: resultados da simulação dos *bandits* com adaptação da restrição de justiça

(a) Chicago, USA

(b) Rio de Janeiro, Brazil



tarefas "Chicago, USA"e "Rio de Janeiro, Brazil". Observamos um aumento consistente no controle de justiça em todos os métodos enquanto é observado uma redução na relevância na maioria dos métodos, sugerindo que todos os casos houve uma troca entre relevância e justiça, sendo essa troca positiva na maioria dos casos exceto para o método *ExploreThenExploit-Fair* onde observamos um UFG negativo nos dois experimentos. É importante ressaltar que devido a sensibilidade da métrica de PropFair, um aumento de 1% pode ser equivalente a ter um sistema próximo do ideal de exposição esperado, a Fig 5.2.4 apresenta o total de exposição de cada grupo para os métodos ϵ-*Greedy-Fair* e *SoftmaxExplorer-Fair* e sua versão *vanilla* onde podemos constatar o equilíbrio na exposição dado pelo aumento relativo do PropFair.

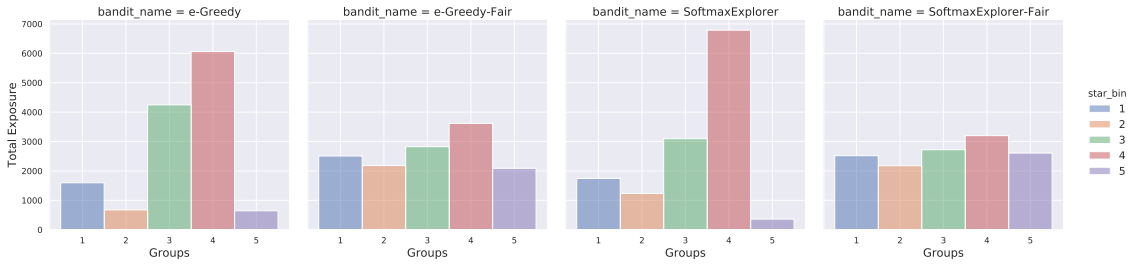
Tabela 5.3: comparação do ganho em relevância e justiça dos *bandits* com controle de justiça em relação a sua versão *vanilla* para as tarefas "Chicago, EUA"e "Rio de Janeiro, Brazil". Outros resultados no Apêndice B

	Chicago, USA			Rio de Janeiro, Brazil		
	CMR	PropFair	UFG	CMR	PropFair	UFG
NeuralUCB-Fair	−2.4%	3.5%	2.3%	−1.2%	1.1%	0.4%
Adaptive-Greedy-Fair	−1.7%	0.6%	1.6%	−2.2%	0.6%	−0.4%
SoftmaxExplorer-Fair	**-11.7%**	**5.2%**	−0.4%	**-4.5%**	1.1%	−0.8%
ExploreThenExploit-Fair	−3.7%	3.5%	2.1%	−3.8%	**1.7%**	0.4%
ϵ-Greedy-Fair	−4.8%	4.1%	**2.4%**	6.0%	1.1%	**3.2%**

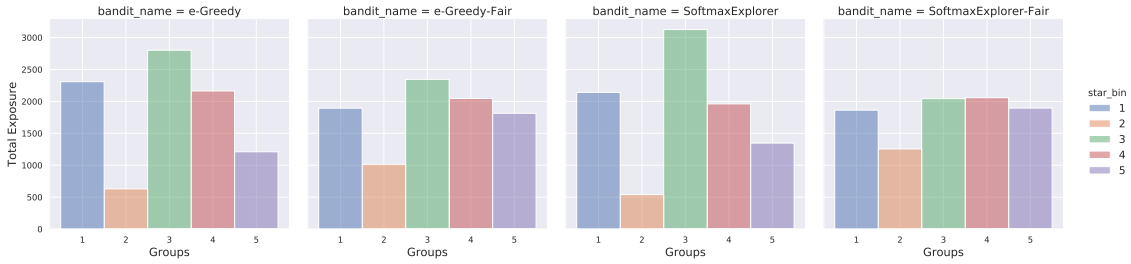
No entanto, os experimentos também mostram uma variância no ganho em utilizar o método *Fair-Feature-Policy* em cada *bandit*, sugerindo que a política de exploração do *bandit* impacta na otimização do *oráculo* em termos de relevância e justiça, e que a parametrização do φ é sensível ao método, sendo necessário ajustar a depender do método de exploração do *bandit*. Esse efeito fica evidente nos resultados do método *SoftmaxExplorer-Fair* no experimento "Chicago, USA", onde observamos uma maior discrepância em relação a sua versão *vanilla*, e ao apresentar um PropFair de 0.181 sugere

que o valor de $\varphi = 0.2$ para esse método está acima do ideal. Comparativamente, esse valor de PropFair é próximo de uma otimização focada na justiça como a do método *Fainess-Policy* (ver Tabela 5.2).

Figura 5.7: resultados da simulação de controle de justiça na exposição dos grupos do ϵ-*Greedy* e *SoftmaxExplorer* na versão *vanilla* e *Fair* para as tarefas "Chicago, USA"e "Rio de Janeiro, Brazil"



(a) Chicago, USA



(b) Rio de Janeiro, Brazil

5.3 Considerações finais

Neste capítulo, foram apresentados detalhes dos experimentos realizados neste trabalho. O primeiro objetivo foi a validação do *framework* de simulação, o MARS-Gym, e, o desenvolvimento e avaliação dos modelos *baselines* de *contextual bandits*.

Abordamos os resultados em três categorias, a primeira voltada em analisar a simulação, em que apresentamos as curvas de recompensa média acumulada de cada modelo. Dessa forma, foi possível discutir o impacto de cada estratégia de exploração nos diferentes cenários propostos. A segunda categoria de resultados validou os modelos na perspectiva de qualidade da lista recomendada usando as métricas de *nDCG*, *Precisão*, *Cobertura* e *Personalização* e como as métricas *off-policy* podem ser usadas para analisar o viés do modelo. Por fim, analisamos os modelos do ponto de vista de justiça, mensuramos as métricas de *maus tratos díspares* e *tratamento díspar* para diferentes atributos tanto dos usuários quanto dos fornecedores, em que ficou claro que a não preocupação

com esses vieses podem prejudicar a experiência mútuo dos interessados em utilizar a plataforma.

Em um segundo momento, foram apresentados os detalhes dos experimentos destinados as políticas que compõem o *Neural Contextual Bandit* com restrição de justiça onde avaliamos o impacto nas métricas de relevância e justiça em diferentes cenários. Em primeira análise, validamos a utilidade da modelagem de afinidade do usuário a exploração e como utiliza-la para o equilíbrio das recomendações com a justiça, em seguida avaliamos o impacto do parâmetro φ em cada política e definimos um *threshold* ideal para maximizar a métrica de UFG. Por fim, analisamos as políticas do ponto de vista de relevância e justiça, onde mensuramos as métricas de CMR, PropFair e UFG, em que ficou evidente que todas as políticas com controle de justiça apresentadas levaram a um aumento significativo no PropFair, e que diferentes métodos de *Neural Contextual Bandits* podem ser adaptados a essa política.

CAPÍTULO 6

Conclusões e Trabalhos Futuros

Neste trabalho, propusemos o *MARS-Gym*, um *framework* de código aberto para modelar, treinar e avaliar sistemas de recomendação baseados em RL para marketplaces. Apresentamos seus componentes internos e fornecemos implementações e análise de *baselines* para servir de ponto de partida para outros trabalhos. Ressaltamos também, que os resultados apresentados são uma contribuição extra do nosso trabalho na tarefa de *benchmarking* de *contextual bandits*, complementando os resultados de trabalhos anteriores. Os resultados são abordados no capítulo (5), em que, além de validarmos a utilidade do *framework* de simulação desenvolvido nesta tese, podemos discutir as particularidades de cada *bandit* implementado e avaliado nos diferentes cenários propostos como experimentos.

Pode-se concluir que a relevância das recomendações para o usuário e o controle de justiça como exposição para os fornecedores são de fato conceitos conflitantes e que ao otimizar um dos objetivos irá essencialmente prejudicar o outro, embora não seja uma relação proporcional devido a características do próprio usuário, como a afinidade a exploração por exemplo. Deste modo, as políticas de recomendação com restrição de justiça conseguem explorar e balancear as recomendações entre relevância e justiça de uma maneira otimizada.

Dentre as políticas de controle e justiça propostas, é possível concluir que uma modelagem voltada para recompensa multiobjetiva leva a um melhor equilíbrio entre relevância e justiça pois impacta diretamente na otimização do *oráculo* utilizando no *Neural Contextual Bandit* e evita uma exploração enviesada por esses conceitos. É importante ressaltar também que uma modelagem do estado atual de justiça do sistema como *features* do modelo impactam positivamente nas métricas de relevância e justiça, sendo um módulo importante no *Neural Contextual Bandit* com restrições de justiça. Por fim, é possível concluir que o método proposto (Fair-Feature-Policy) pode ser adaptado a qualquer *Neural Contextual Bandit*, dando características de controle de justiça ao método sem alterar fundamentalmente as particularidades do mesmo.

Por fim, também esperamos receber contribuições da comunidade para o MARS-gym por ser um framework *open source*, não apenas com solicitações de recursos e novos

códigos, mas também com novas tarefas de avaliação, competições e datasets integrados. Dessa forma, esperamos popularizar esse projeto e fornecer as ferramentas necessárias para acelerar a pesquisa e o desenvolvimento de agentes de aprendizado por reforço para recomendação em *marketplaces*.

Como trabalhos futuros, esperamos explorar outros modelos de *Neural Contextual Bandits* como o *Neural-LinUCB* [166] e o *Epistemic Neural Recommendation (ENR)* [178] e avaliar o impacto de outras formas de exploração como a *Sample Average Uncertanty (SAU)* [127] com o objetivo de adapta-los a uma modelagem com restrição de justiça. Esperamos também, assim como o apresentado no trabalho do Zhu e Van Roy [178], explorar a escalabilidade dos métodos propostos, visto que um dos principais objetivos é que viabilizar a implementação desses algorítimos pela indústria.

Referências Bibliográficas

[1] Yasin Abbasi-Yadkori, Dávid Pál e Csaba Szepesvári. "Improved algorithms for linear stochastic bandits". Em: *Advances in neural information processing systems* 24 (2011), pp. 2312–2320.

[2] Himan Abdollahpouri. "Popularity bias in recommendation: A multi-stakeholder perspective". Tese de dout. University of Colorado at Boulder, 2020.

[3] Himan Abdollahpouri e Robin Burke. "Multi-stakeholder recommendation and its connection to multi-sided fairness". Em: *arXiv preprint arXiv:1907.13158* (2019).

[4] Himan Abdollahpouri, Robin Burke e Bamshad Mobasher. "Controlling popularity bias in learning-to-rank recommendation". Em: *Proceedings of the eleventh ACM conference on recommender systems*. 2017, pp. 42–46.

[5] Himan Abdollahpouri, Robin Burke e Bamshad Mobasher. "Managing popularity bias in recommender systems with personalized re-ranking". Em: *The thirty-second international flairs conference*. 2019.

[6] Himan Abdollahpouri et al. "Beyond personalization: Research directions in multistakeholder recommendation". Em: *arXiv preprint arXiv:1905.01986* (2019).

[7] Himan Abdollahpouri et al. "Multistakeholder recommendation: Survey and research directions". Em: *User Modeling and User-Adapted Interaction* 30.1 (2020), pp. 127–158.

[8] Fabian Abel et al. "Recsys challenge 2016: Job recommendations". Em: *Proceedings of the 10th ACM conference on recommender systems*. 2016, pp. 425–426.

[9] Jens Adamczak. *RecSys Challenge 2019 · trivago tech blog*. TRIVAGO. Mar. de 2019. URL: https://tech.trivago.com/2019/03/11/recsys-challenge-2019.

[10] Gediminas Adomavicius e Alexander Tuzhilin. "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions". Em: *IEEE transactions on knowledge and data engineering* 17.6 (2005), pp. 734–749.

[11] Alekh Agarwal et al. “Taming the monster: A fast and simple algorithm for contextual bandits”. Em: *International Conference on Machine Learning*. PMLR. 2014, pp. 1638–1646.

[12] Charu C Aggarwal et al. *Recommender systems*. Vol. 1. Springer, 2016.

[13] Shipra Agrawal e Navin Goyal. “Thompson sampling for contextual bandits with linear payoffs”. Em: *International Conference on Machine Learning*. PMLR. 2013, pp. 127–135.

[14] Xavier Amatriain et al. “Recommender systems handbook”. Em: *Data Mining Methods for Recommender Systems. United State: Springer* (2011).

[15] Vito Walter Anelli et al. “RecSys 2021 Challenge Workshop: Fairness-aware engagement prediction at scale on Twitter’s Home Timeline”. Em: *Fifteenth ACM Conference on Recommender Systems*. 2021, pp. 819–824.

[16] Peter Auer. “Using confidence bounds for exploitation-exploration trade-offs”. Em: *Journal of Machine Learning Research* 3.Nov (2002), pp. 397–422.

[17] Peter Auer et al. “Gambling in a rigged casino: The adversarial multi-armed bandit problem”. Em: *Proceedings of IEEE 36th Annual Foundations of Computer Science*. IEEE. 1995, pp. 322–331.

[18] Peter Auer et al. “The nonstochastic multiarmed bandit problem”. Em: *SIAM journal on computing* 32.1 (2002), pp. 48–77.

[19] Kinjal Basu et al. “A Framework for Fairness in Two-Sided Marketplaces”. Em: *arXiv preprint arXiv:2006.12756* (2020).

[20] Joeran Beel et al. “The impact of demographics (age and gender) and other user-characteristics on evaluating recommender systems”. Em: *International Conference on Theory and Practice of Digital Libraries*. Springer. 2013, pp. 396–400.

[21] Yoshua Bengio. *Learning deep architectures for AI*. Now Publishers Inc, 2009.

[22] Yoshua Bengio, Yann LeCun et al. “Scaling learning algorithms towards AI”. Em: *Large-scale kernel machines* 34.5 (2007), pp. 1–41.

[23] Lucas Bernardi, Sakshi Batra e Cintia Alicia Bruscantini. “Simulations in Recommender Systems: An industry perspective”. Em: *arXiv preprint arXiv:2109.06723* (2021).

[24] Erik Bernhardsson e Elias Freider. *spotify/luigi*. Versão latest. Mai. de 2014. URL: https://github.com/spotify.

[25] Alina Beygelzimer e John Langford. “The offset tree for learning with partial labels”. Em: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2009, pp. 129–138.

[26] Reuben Binns. “Fairness in machine learning: Lessons from political philosophy”. Em: *Conference on Fairness, Accountability and Transparency*. PMLR. 2018, pp. 149–159.

[27] Dheeraj Bokde, Sheetal Girase e Debajyoti Mukhopadhyay. “Matrix factorization model in collaborative filtering algorithms: A survey”. Em: *Procedia Computer Science* 49 (2015), pp. 136–146.

[28] Léon Bottou et al. “Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising”. Em: *Journal of Machine Learning Research* 14.65 (2013), pp. 3207–3260. URL: http://jmlr.org/papers/v14/bottou13a.html.

[29] Jiajun Bu et al. “Music recommendation by unified hypergraph: combining social media information and music content”. Em: *Proceedings of the 18th ACM international conference on Multimedia*. 2010, pp. 391–400.

[30] Robin Burke. “Multisided fairness for recommendation”. Em: *arXiv preprint arXiv:1707.00093* (2017).

[31] Simon Caton e Christian Haas. “Fairness in machine learning: A survey”. Em: *arXiv preprint arXiv:2010.04053* (2020).

[32] Tianfeng Chai e Roland R Draxler. “Root mean square error (RMSE) or mean absolute error (MAE)”. Em: *Geoscientific Model Development Discussions* 7.1 (2014), pp. 1525–1534.

[33] Haochen Chen et al. “A tutorial on network embeddings”. Em: *arXiv preprint arXiv:1808.02590* (2018).

[34] Jiawei Chen et al. “Bias and debias in recommender system: A survey and future directions”. Em: *arXiv preprint arXiv:2010.03240* (2020).

[35] Lei Chen et al. “A multi-task learning approach for improving travel recommendation with keywords generation”. Em: *Knowledge-Based Systems* 233 (2021), p. 107521.

[36] Xiaocong Chen et al. “A survey of deep reinforcement learning in recommender systems: A systematic review and future directions”. Em: *arXiv preprint arXiv:2109.03540* (2021).

[37] Yifang Chen et al. “Fair contextual multi-armed bandits: Theory and experiments”. Em: *Conference on Uncertainty in Artificial Intelligence*. PMLR. 2020, pp. 181–190.

[38] Heng-Tze Cheng et al. “Wide & deep learning for recommender systems”. Em: *Proceedings of the 1st workshop on deep learning for recommender systems*. 2016, pp. 7–10.

[39] Wei Chu et al. “Contextual bandits with linear payoff functions”. Em: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop e Conference Proceedings. 2011, pp. 208–214.

[40] Mark Collier e Hector Urdiales Llorens. “Deep contextual multi-armed bandits”. Em: *arXiv preprint arXiv:1807.09809* (2018).

[41] David Cortes. “Adapting multi-armed bandits policies to contextual bandits scenarios”. Em: *CoRR* abs/1811.04383 (2018). arXiv: 1811.04383. URL: http://arxiv.org/abs/1811.04383.

[42] Paul Covington, Jay Adams e Emre Sargin. “Deep neural networks for youtube recommendations”. Em: *Proceedings of the 10th ACM conference on recommender systems*. 2016, pp. 191–198.

[43] Steven Kane Curtis e Oksana Mont. “Sharing economy business models for sustainability”. Em: *Journal of Cleaner Production* 266 (2020), p. 121519.

[44] Alexander D’Amour et al. “Fairness is not static: deeper understanding of long term fairness via simulation studies”. Em: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 2020, pp. 525–534.

[45] *DLRS 2017: Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*. Como, Italy: Association for Computing Machinery, 2017. ISBN: 9781450353533.

[46] Miroslav Dudík, John Langford e Lihong Li. “Doubly Robust Policy Evaluation and Learning”. Em: *Proceedings of the 28th International Conference on International Conference on Machine Learning*. ICML’11. Bellevue, Washington, USA: Omnipress, 2011, pp. 1097–1104. ISBN: 9781450306195.

[47] Miroslav Dudík et al. “Doubly robust policy evaluation and optimization”. Em: *Statistical Science* 29.4 (2014), pp. 485–511.

[48] Simen Eide, Audun M Øygard e Ning Zhou. “Five lessons from building a deep neural network recommender for marketplaces”. Em: *ACM KDD*. Vol. 18. 2018.

[49] Simen Eide e Ning Zhou. “Deep neural network marketplace recommenders in online experiments”. Em: *Proceedings of the 12th ACM Conference on Recommender Systems*. 2018, pp. 387–391.

[50] Kim Falk. *Practical recommender systems*. Simon e Schuster, 2019.

[51] Pratik Gajane e Mykola Pechenizkiy. “On formalizing fairness in prediction with machine learning”. Em: *arXiv preprint arXiv:1710.03184* (2017).

[52] Florent Garcin et al. “Offline and online evaluation of news recommender systems at swissinfo. ch”. Em: *Proceedings of the 8th ACM Conference on Recommender systems*. 2014, pp. 169–176.

[53] Aurélien Garivier e Eric Moulines. “On upper-confidence bound policies for non-stationary bandit problems”. Em: *arXiv preprint arXiv:0805.3415* (2008).

[54] Mouzhi Ge, Carla Delgado-Battenfeld e Dietmar Jannach. “Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity”. Em: *Proceedings of the Fourth ACM Conference on Recommender Systems*. RecSys ’10. Barcelona, Spain: Association for Computing Machinery, 2010, pp. 257–260. ISBN: 9781605589060. DOI: 10.1145/1864708.1864761. URL: https://doi.org/10.1145/1864708.1864761.

[55] Alexandre Gilotte et al. “Offline a/b testing for recommender systems”. Em: *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 2018, pp. 198–206.

[56] Dmitri Goldenberg e Pavel Levin. “Booking. com Multi-Destination Trips Dataset”. Em: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2021, pp. 2457–2462.

[57] Mihajlo Grbovic. “Search ranking and personalization at Airbnb”. Em: *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 2017, pp. 339–340.

[58] Dalin Guo et al. “Deep Bayesian Bandits: Exploring in Online Personalized Recommendations”. Em: *Fourteenth ACM Conference on Recommender Systems*. 2020, pp. 456–461.

[59] Malay Haldar et al. “Applying deep learning to airbnb search”. Em: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 1927–1935.

[60] Mohd Abdul Hameed, Omar Al Jadaan e Sirandas Ramachandram. “Collaborative filtering based recommendation system: A survey”. Em: *International Journal on Computer Science and Engineering* 4.5 (2012), p. 859.

[61] Larry Hardesty. *The history of Amazon’s recommendation algorithm*. Ago. de 2020. URL: https://www.amazon.science/the-history-of-amazons-recommendation-algorithm.

[62] Moritz Hardt, Eric Price e Nati Srebro. “Equality of opportunity in supervised learning”. Em: *Advances in neural information processing systems* 29 (2016), pp. 3315–3323.

[63] Xiangnan He et al. “Neural collaborative filtering”. Em: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 173–182.

[64] Jonathan L Herlocker et al. “Evaluating collaborative filtering recommender systems”. Em: *ACM Transactions on Information Systems (TOIS)* 22.1 (2004), pp. 5–53.

[65] Balázs Hidasi et al. “Session-based recommendations with recurrent neural networks”. Em: *arXiv preprint arXiv:1511.06939* (2015).

[66] Geoffrey E Hinton, Simon Osindero e Yee-Whye Teh. “A fast learning algorithm for deep belief nets”. Em: *Neural computation* 18.7 (2006), pp. 1527–1554.

[67] Geoffrey E Hinton e Ruslan R Salakhutdinov. “Reducing the dimensionality of data with neural networks”. Em: *science* 313.5786 (2006), pp. 504–507.

[68] Yifan Hu, Yehuda Koren e Chris Volinsky. “Collaborative filtering for implicit feedback datasets”. Em: *2008 Eighth IEEE International Conference on Data Mining*. Ieee. 2008, pp. 263–272.

[69] Wasim Huleihel, Soumyabrata Pal e Ofer Shayevitz. “Learning User Preferences in Non-Stationary Environments”. Em: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 1432–1440.

[70] Eugene Ie et al. *RecSim: A Configurable Simulation Platform for Recommender Systems*. 2019. arXiv: 1909.04847 [cs.LG].

[71] Tamas Jambor e Jun Wang. “Optimizing multiple objectives in collaborative filtering”. Em: *Proceedings of the fourth ACM conference on Recommender systems*. 2010, pp. 55–62.

[72] Kalervo Järvelin e Jaana Kekäläinen. “IR Evaluation Methods for Retrieving Highly Relevant Documents”. Em: *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’00. Athens, Greece: Association for Computing Machinery, 2000, pp. 41–48. ISBN: 1581132263. DOI: 10.1145/345508.345545. URL: https://doi.org/10.1145/345508.345545.

[73] Olivier Jeunen e Bart Goethals. “Top-k contextual bandits with equity of exposure”. Em: *Fifteenth ACM Conference on Recommender Systems*. 2021, pp. 310–320.

[74] Olivier Jeunen, David Rohde e Flavian Vasile. *On the Value of Bandit Feedback for Offline Recommender System Evaluation*. 2019. arXiv: 1907.12384 [cs.IR].

[75] Olivier Jeunen et al. *Learning from Bandit Feedback: An Overview of the State-of-the-art*. 2019. arXiv: 1909.08471 [cs.IR].

[76] Yitong Ji et al. "A critical study on data leakage in recommender system offline evaluation". Em: *ACM Transactions on Information Systems* 41.3 (2023), pp. 1–27.

[77] Matthew Joseph et al. "Fairness in learning: Classic and contextual bandits". Em: *arXiv preprint arXiv:1605.07139* (2016).

[78] Marius Kaminskas e Derek Bridge. "Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems". Em: *ACM Transactions on Interactive Intelligent Systems (TiiS)* 7.1 (2016), pp. 1–42.

[79] Frank P Kelly, Aman K Maulloo e David Kim Hong Tan. "Rate control for communication networks: shadow prices, proportional fairness and stability". Em: *Journal of the Operational Research society* 49 (1998), pp. 237–252.

[80] Krishnaram Kenthapadi, Benjamin Le e Ganesh Venkataraman. "Personalized job recommendation system at linkedin: Practical challenges and lessons learned". Em: *Proceedings of the eleventh ACM conference on recommender systems*. 2017, pp. 346–347.

[81] Peter Knees et al. "Recsys challenge 2019: Session-based hotel recommendations". Em: *Proceedings of the 13th ACM Conference on Recommender Systems*. 2019, pp. 570–571.

[82] Ron Kohavi e Roger Longbotham. "Online Controlled Experiments and A/B Testing." Em: *Encyclopedia of machine learning and data mining* 7.8 (2017), pp. 922–929.

[83] Karl Krauth et al. "Do offline metrics predict online performance in recommender systems?" Em: *arXiv preprint arXiv:2011.07931* (2020).

[84] John Langford e Tong Zhang. "The Epoch-Greedy Algorithm for Contextual Multi-armed Bandits". en. Em: (), p. 8.

[85] Tor Lattimore e Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[86] Minh Le, Subhradeep Kayal e Andrew Douglas. "The Impact of Recommenders on Scientific Article Discovery: The Case of Mendeley Suggest." Em: *ImpactRS@ RecSys*. 2019.

[87] Yann LeCun, Yoshua Bengio e Geoffrey Hinton. “Deep learning”. Em: *nature* 521.7553 (2015), pp. 436–444.

[88] Dokyun Lee e Kartik Hosanagar. “Impact of recommender systems on sales volume and diversity”. Em: (2014).

[89] Jurek Leonhardt, Avishek Anand e Megha Khosla. “User fairness in recommender systems”. Em: *Companion Proceedings of the The Web Conference 2018*. 2018, pp. 101–102.

[90] Boying Li et al. “Predicting online e-marketplace sales performances: A big data approach”. Em: *Computers & Industrial Engineering* 101 (2016), pp. 565–571.

[91] Lihong Li et al. “A contextual-bandit approach to personalized news article recommendation”. en. Em: *Proceedings of the 19th international conference on World wide web - WWW ’10*. Raleigh, North Carolina, USA: ACM Press, 2010, p. 661. ISBN: 978-1-60558-799-8. DOI: 10.1145/1772690.1772758. URL: http://portal.acm.org/citation.cfm?doid=1772690.1772758 (acesso em 26/05/2020).

[92] Lihong Li et al. “Counterfactual Estimation and Optimization of Click Metrics in Search Engines: A Case Study”. Em: *Proceedings of the 24th International Conference on World Wide Web*. WWW ’15 Companion. Florence, Italy: Association for Computing Machinery, 2015, pp. 929–934. ISBN: 9781450334730. DOI: 10.1145/2740908.2742562. URL: https://doi.org/10.1145/2740908.2742562.

[93] Lihong Li et al. “Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms”. Em: *Proceedings of the fourth ACM international conference on Web search and data mining*. 2011, pp. 297–306.

[94] Shuai Li, Alexandros Karatzoglou e Claudio Gentile. “Collaborative Filtering Bandits”. en. Em: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval - SIGIR ’16*. Pisa, Italy: ACM Press, 2016, pp. 539–548. ISBN: 978-1-4503-4069-4. DOI: 10.1145/2911451.2911548. URL: http://dl.acm.org/citation.cfm?doid=2911451.2911548 (acesso em 26/05/2020).

[95] R. Liao, V. Segovia e M. Brennan. *Attention Factory: The Story of TikTok and China’s ByteDance*. Independently Published, 2020. ISBN: 9798694483292. URL: https://books.google.com.br/books?id=BewAzgEACAAJ.

[96] Feng Liu et al. “Deep reinforcement learning based recommendation with explicit user-item interactions modeling”. Em: *arXiv preprint arXiv:1810.12027* (2018).

[97] Weiwen Liu et al. “Balancing between accuracy and fairness for interactive recommendation with reinforcement learning”. Em: *Advances in Knowledge Discovery and Data Mining* 12084 (2021), p. 155.

[98] Yichao Lu, Ruihai Dong e Barry Smyth. “Why I like it: multi-task learning for recommendation and explanation”. Em: *Proceedings of the 12th ACM Conference on Recommender Systems*. 2018, pp. 4–12.

[99] Ian MacKenzie, Chris Meyer e Steve Noble. *How retailers can keep up with consumers*. Fev. de 2018. URL: https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers#.

[100] Stefan Magureanu, Richard Combes e Alexandre Proutiere. “Lipschitz bandits: Regret lower bound and optimal algorithms”. Em: *Conference on Learning Theory*. PMLR. 2014, pp. 975–999.

[101] Juho Makkonen e Cristóbald Gracia. *The Lean Market Place: A Practical Guide to Building a Successful Online Marketplace*. Sharetribe, 2018.

[102] Christopher D Manning, Prabhakar Raghavan e Hinrich Schutze. “Introduction to information retrieval? cambridge university press 2008”. Em: *Ch* 20 (), pp. 405–416.

[103] Masoud Mansoury et al. “Feedback loop and bias amplification in recommender systems”. Em: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 2145–2148.

[104] Paolo Massa e Paolo Avesani. “Trust-aware recommender systems”. Em: *Proceedings of the 2007 ACM conference on Recommender systems*. 2007, pp. 17–24.

[105] James McInerney et al. “Explore, Exploit, and Explain: Personalizing Explainable Recommendations with Bandits”. Em: *Proceedings of the 12th ACM Conference on Recommender Systems*. RecSys ’18. Vancouver, British Columbia, Canada: Association for Computing Machinery, 2018, pp. 31–39. ISBN: 9781450359016. DOI: 10.1145/3240323.3240354. URL: https://doi.org/10.1145/3240323.3240354.

[106] James McInerney et al. “Explore, exploit, and explain: personalizing explainable recommendations with bandits”. Em: *Proceedings of the 12th ACM conference on recommender systems*. 2018, pp. 31–39.

[107] Wes McKinney. “Data Structures for Statistical Computing in Python”. Em: *Proceedings of the 9th Python in Science Conference*. Ed. por Stéfan van der Walt e Jarrod Millman. 2010, pp. 56–61. DOI: 10.25080/Majora-92bf1922-00a.

[108] Sean M McNee, John Riedl e Joseph A Konstan. “Being accurate is not enough: how accuracy metrics have hurt recommender systems”. Em: *CHI’06 extended abstracts on Human factors in computing systems*. 2006, pp. 1097–1101.

[109] Ninareh Mehrabi et al. “A survey on bias and fairness in machine learning”. Em: *ACM Computing Surveys (CSUR)* 54.6 (2021), pp. 1–35.

[110] Rishabh Mehrotra e Benjamin Carterette. “Recommendations in a marketplace”. Em: *Proceedings of the 13th ACM Conference on Recommender Systems*. 2019, pp. 580–581.

[111] Rishabh Mehrotra et al. “Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness & satisfaction in recommendation systems”. Em: *Proceedings of the 27th acm international conference on information and knowledge management*. 2018, pp. 2243–2251.

[112] Blossom Metevier et al. “Offline contextual bandits with high probability fairness guarantees”. Em: *Advances in neural information processing systems* 32 (2019).

[113] Tsunenori Mine, Tomoyuki Kakuta e Akira Ono. “Reciprocal recommendation for job matching with bidirectional feedback”. Em: *2013 Second IIAI International Conference on Advanced Applied Informatics*. IEEE. 2013, pp. 39–44.

[114] Martin Mladenov et al. “Recsim ng: Toward principled uncertainty modeling for recommender ecosystems”. Em: *arXiv preprint arXiv:2103.08057* (2021).

[115] Oksana Mont et al. “A decade of the sharing economy: Concepts, users, business and governance perspectives”. Em: *Journal of Cleaner Production* 269 (2020), p. 122215.

[116] Fakhroddin Noorbehbahani e Zeinab Zarein. “The impact of demographic factors on persuasion strategies in personalized recommender system”. Em: *2018 8th International Conference on Computer and Knowledge Engineering (ICCKE)*. IEEE. 2018, pp. 104–109.

[117] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[118] Iván Palomares et al. “Reciprocal Recommender Systems: Analysis of state-of-art literature, challenges and opportunities towards social recommendation”. Em: *Information Fusion* 69 (2021), pp. 103–127.

[119] Arkadiusz Paterek. “Improving regularized singular value decomposition for collaborative filtering”. Em: *Proceedings of KDD cup and workshop*. Vol. 2007. 2007, pp. 5–8.

[120] Gourab K Patro et al. “Fairrec: Two-sided fairness for personalized recommendations in two-sided platforms”. Em: *Proceedings of The Web Conference 2020*. 2020, pp. 1194–1204.

[121] Vianney Perchet e Philippe Rigollet. “The multi-armed bandit problem with covariates”. Em: *The Annals of Statistics* 41.2 (2013), pp. 693–721.

[122] Ladislav Peska e Peter Vojtas. “Off-line vs. On-line Evaluation of Recommender Systems in Small E-commerce”. Em: *Proceedings of the 31st ACM Conference on Hypertext and Social Media*. 2020, pp. 291–300.

[123] Luiz Pizzato et al. “Reciprocal recommender system for online dating”. Em: *Proceedings of the fourth ACM conference on Recommender systems*. 2010, pp. 353–354.

[124] Thomas Puschmann e Rainer Alt. “Sharing economy”. Em: *Business & Information Systems Engineering* 58.1 (2016), pp. 93–99.

[125] The ACM Conference Series on Recommender Systems. *ACM RecSys challenge 2019 | dataset*. ACM. 2019. URL: https://recsys.trivago.cloud/challenge/dataset.

[126] Paul Resnick et al. “Grouplens: An open architecture for collaborative filtering of netnews”. Em: *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 1994, pp. 175–186.

[127] Mattia Rigotti e Rong Zhu. “Deep Bandits Show-Off: Simple and Efficient Exploration with Deep Networks”. Em: *arXiv preprint arXiv:2105.04683* (2021).

[128] Carlos Riquelme, George Tucker e Jasper Snoek. “Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling”. Em: *arXiv preprint arXiv:1802.09127* (2018).

[129] David Rohde et al. *RecoGym: A Reinforcement Learning Environment for the problem of Product Recommendation in Online Advertising*. 2018. arXiv: 1808.00720 [cs.IR].

[130] Yuta Saito e Thorsten Joachims. “Counterfactual Learning and Evaluation for Recommender Systems: Foundations, Implementations, and Recent Advances”. Em: *Fifteenth ACM Conference on Recommender Systems*. 2021, pp. 828–830.

[131] Yuta Saito et al. “Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation”. Em: *arXiv preprint arXiv:2008.07146* (2020).

[132] Marlesson R. O. Santana e Anderson Soares. “Hybrid Model with Time Modeling for Sequential Recommender Systems”. Em: *ACM WSDM Workshop on Web Tourism (WSDM WebTour’21)* (2021).

[133] Marlesson R. O. Santana et al. “Contextual Meta-Bandit for Recommender Systems Selection”. Em: *Fourteenth ACM Conference on Recommender Systems (RecSys)*. RecSys ’20. Virtual Event, Brazil: Association for Computing Machinery, 2020, pp. 444–449. ISBN: 9781450375832. DOI: 10.1145/3383313.3412209. URL: https://doi.org/10.1145/3383313.3412209.

[134] Marlesson R. O. Santana et al. “MARS-gym: A Gym Framework to Model, Train, and Evaluate Recommender Systems for Marketplaces”. Em: *Workshop on Advanced Neural Algorithms and Theories for Recommender Systems (NeuRec)*. Virtual Event, Italy: 20th Industrial Conference on Data Mining, 2020. URL: https://arxiv.org/abs/2010.07035.

[135] Marlesson R. O. Santana et al. “MARS-Gym: Offline Reinforcement Learning for Recommender Systems in Marketplaces”. Em: *Offline Reinforcement Learning at Neural Information Processing Systems (NeurIPS)*. Spotlight Paper. Virtual Event, Canada: 34th Conference on Neural Information Processing Systems (NeurIPS 2020), 2020. URL: https://offline-rl-neurips.github.io/program/offrl_21.html.

[136] Badrul Sarwar et al. *Application of dimensionality reduction in recommender system-a case study*. Rel. técn. Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[137] Sven Schmit e Carlos Riquelme. “Human interaction with recommendation systems”. Em: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2018, pp. 862–870.

[138] M Scrimitore. *The economics of E-commerce. A strategic guide to understanding and designing the online marketplace.* 2006.

[139] Guy Shani e Asela Gunawardana. “Evaluating recommendation systems”. Em: *Recommender systems handbook*. Springer, 2011, pp. 257–297.

[140] Guy Shani, David Heckerman e Ronen I. Brafman. “An MDP-Based Recommender System”. Em: *J. Mach. Learn. Res.* 6 (dez. de 2005), pp. 1265–1295. ISSN: 1532-4435.

[141] Guy Shani, David Heckerman e Ronen I. Brafman. “An MDP-Based Recommender System”. Em: *J. Mach. Learn. Res.* 6 (dez. de 2005). Publisher: JMLR.org, pp. 1265–1295. ISSN: 1532-4435.

[142] Bichen Shi et al. “PyRecGym: A Reinforcement Learning Gym for Recommender Systems”. Em: *Proceedings of the 13th ACM Conference on Recommender Systems*. RecSys ’19. Copenhagen, Denmark: Association for Computing Machinery,

2019, pp. 491–495. ISBN: 9781450362436. DOI: 10.1145/3298689.3346981. URL: https://doi.org/10.1145/3298689.3346981.

[143] Ashudeep Singh e Thorsten Joachims. “Fairness of exposure in rankings”. Em: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 2219–2228.

[144] Aleksandrs Slivkins. “Introduction to multi-armed bandits”. Em: *arXiv preprint arXiv:1904.07272* (2019).

[145] Brent Smith e Greg Linden. “Two decades of recommender systems at Amazon. com”. Em: *Ieee internet computing* 21.3 (2017), pp. 12–18.

[146] Gabriel de Souza Pereira Moreira et al. “Transformers4Rec: Bridging the Gap between NLP and Sequential/Session-Based Recommendation”. Em: *Fifteenth ACM Conference on Recommender Systems*. 2021, pp. 143–153.

[147] Alex Strehl et al. “Learning from logged implicit exploration data”. Em: *arXiv preprint arXiv:1003.0120* (2010).

[148] Fei Sun et al. “BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer”. Em: *Proceedings of the 28th ACM international conference on information and knowledge management*. 2019, pp. 1441–1450.

[149] Richard S Sutton e Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[150] Adith Swaminathan e Thorsten Joachims. “Counterfactual Risk Minimization: Learning from Logged Bandit Feedback”. Em: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML’15. Lille, France: JMLR.org, 2015, pp. 814–823.

[151] Adith Swaminathan e Thorsten Joachims. “The self-normalized estimator for counterfactual learning”. Em: *advances in neural information processing systems* 28 (2015).

[152] Jiaxi Tang e Ke Wang. “Personalized top-n sequential recommendation via convolutional sequence embedding”. Em: *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 2018, pp. 565–573.

[153] The pandas development team. *pandas-dev/pandas: Pandas*. Versão latest. Fev. de 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[154] William R Thompson. “Biometrika trust”. Em: *Biometrika* 25.3/4 (1933), pp. 285–294.

[155] Poonam B Thorat, RM Goudar e Sunita Barve. “Survey on collaborative filtering, content-based filtering and hybrid recommendation system”. Em: *International Journal of Computer Applications* 110.4 (2015), pp. 31–36.

[156] Michel Tokic. “Adaptive ϵ-Greedy Exploration in Reinforcement Learning Based on Value Differences”. en. Em: *KI 2010: Advances in Artificial Intelligence*. Ed. por Rüdiger Dillmann et al. Vol. 6359. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 203–210. ISBN: 978-3-642-16110-0 978-3-642-16111-7. DOI: 10.1007/978-3-642-16111-7_23. URL: http://link.springer.com/10.1007/978-3-642-16111-7_23 (acesso em 26/05/2020).

[157] Q. Wang et al. “Online Interactive Collaborative Filtering Using Multi-Armed Bandit with Dependent Arms”. Em: *IEEE Transactions on Knowledge and Data Engineering* 31.8 (ago. de 2019), pp. 1569–1580. ISSN: 1558-2191. DOI: 10.1109/TKDE.2018.2866041.

[158] Shanfeng Wang et al. “Multi-objective optimization for long tail recommendation”. Em: *Knowledge-Based Systems* 104 (2016), pp. 145–155.

[159] Xinxi Wang e Ye Wang. “Improving content-based and hybrid music recommendation using deep learning”. Em: *Proceedings of the 22nd ACM international conference on Multimedia*. 2014, pp. 627–636.

[160] Yining Wang et al. “A theoretical analysis of NDCG ranking measures”. Em: *Proceedings of the 26th annual conference on learning theory (COLT 2013)*. Vol. 8. Citeseer. 2013, p. 6.

[161] Yuyan Wang et al. *Food Discovery with Uber Eats: Recommending for the Marketplace*. 2018. URL: https://eng.uber.com/uber-eats-recommending-marketplace/.

[162] John White. *Bandit algorithms for website optimization*. "O’Reilly Media, Inc.", 2012.

[163] Yao Wu et al. “Collaborative denoising auto-encoders for top-n recommender systems”. Em: *Proceedings of the ninth ACM international conference on web search and data mining*. 2016, pp. 153–162.

[164] Peng Xia et al. “Reciprocal recommendation system for online dating”. Em: *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE. 2015, pp. 234–241.

[165] Xin Xin et al. “Self-supervised reinforcement learning for recommender systems”. Em: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2020, pp. 931–940.

[166] Pan Xu et al. “Neural contextual bandits with deep representation and shallow exploration”. Em: *arXiv preprint arXiv:2012.01780* (2020).

[167] Yisong Yue et al. “A Support Vector Method for Optimizing Average Precision”. Em: *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’07. Amsterdam, The Netherlands: Association for Computing Machinery, 2007, pp. 271–278. ISBN: 9781595935977. DOI: 10.1145/1277741.1277790. URL: https://doi.org/10.1145/1277741.1277790.

[168] Muhammad Bilal Zafar et al. “Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment”. Em: *Proceedings of the 26th International Conference on World Wide Web*. WWW ’17. Perth, Australia: International World Wide Web Conferences Steering Committee, 2017, pp. 1171–1180. ISBN: 9781450349130. DOI: 10.1145/3038912.3052660. URL: https://doi.org/10.1145/3038912.3052660.

[169] Matei Zaharia et al. “Apache Spark: A Unified Engine for Big Data Processing”. Em: *Commun. ACM* 59.11 (out. de 2016), pp. 56–65. ISSN: 0001-0782. DOI: 10.1145/2934664. URL: https://doi.org/10.1145/2934664.

[170] Shuai Zhang et al. “Deep learning based recommender system: A survey and new perspectives”. Em: *ACM Computing Surveys (CSUR)* 52.1 (2019), pp. 1–38.

[171] Weinan Zhang et al. “Deep reinforcement learning for information retrieval: Fundamentals and advances”. Em: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2020, pp. 2468–2471.

[172] Xiangyu Zhao et al. “Deep reinforcement learning for list-wise recommendations”. Em: *arXiv preprint arXiv:1801.00209* (2017).

[173] Zhe Zhao et al. “Recommending what video to watch next: a multitask ranking system”. Em: *Proceedings of the 13th ACM Conference on Recommender Systems*. 2019, pp. 43–51.

[174] Guanjie Zheng et al. “DRN: A deep reinforcement learning framework for news recommendation”. Em: *Proceedings of the 2018 World Wide Web Conference*. 2018, pp. 167–176.

[175] Lei Zheng, Vahid Noroozi e Philip S Yu. “Joint deep modeling of users and items using reviews for recommendation”. Em: *Proceedings of the tenth ACM international conference on web search and data mining*. 2017, pp. 425–434.

[176] Yong Zheng et al. “Fairness in reciprocal recommendations: A speed-dating study”. Em: *Adjunct publication of the 26th conference on user modeling, adaptation and personalization*. 2018, pp. 29–34.

[177] Dongruo Zhou, Lihong Li e Quanquan Gu. “Neural contextual bandits with ucb-based exploration”. Em: *International Conference on Machine Learning*. PMLR. 2020, pp. 11492–11502.

[178] Zheqing Zhu e Benjamin Van Roy. “Scalable Neural Contextual Bandit for Recommender Systems”. Em: *arXiv preprint arXiv:2306.14834* (2023).

APÊNDICE A

Influencia da afinidade do usuário a exploração na relevância

Influencia da afinidade do usuário a exploração nas métricas de relevância do *bandit* "NeuralUCB"e relação com o tamanho de sessão de cada tarefa.

Figura A.1: Métricas da afinidade de exploração para tarefa "Como, Italy"

(a) Distribuição da afinidade de exploração (b) Relação com a performance do *bandit* (c) Relação com o tamanho do histórico de interações do usuário

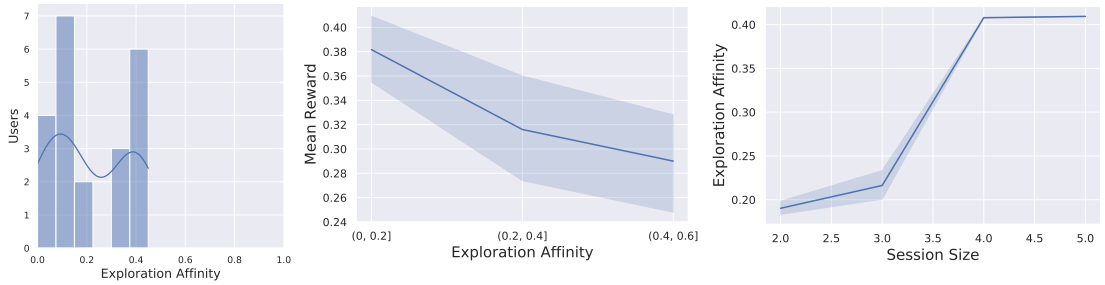


Figura A.2: Métricas da afinidade de exploração para tarefa "Chicago, USA"

(a) Distribuição da afinidade de exploração (b) Relação com a performance do *bandit* (c) Relação com o tamanho do histórico de interações do usuário

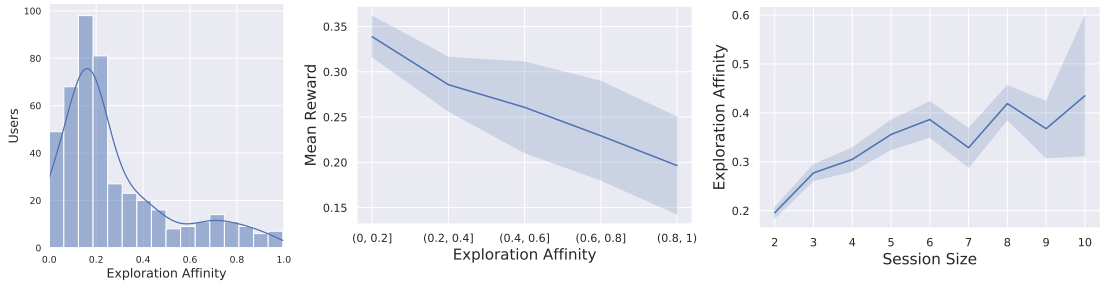


Figura A.3: Métricas da afinidade de exploração para tarefa "Rio de Janeiro, Brazil"

(a) Distribuição da afinidade de exploração

(b) Relação com a performance do *bandit*

(c) Relação com o tamanho do histórico de interações do usuário

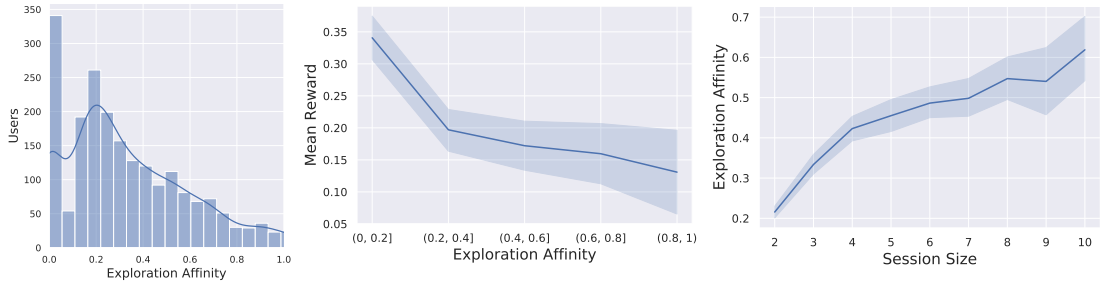


Figura A.4: Métricas da afinidade de exploração para tarefa "New York, USA"

(a) Distribuição da afinidade de exploração

(b) Relação com a performance do *bandit*

(c) Relação com o tamanho do histórico de interações do usuário

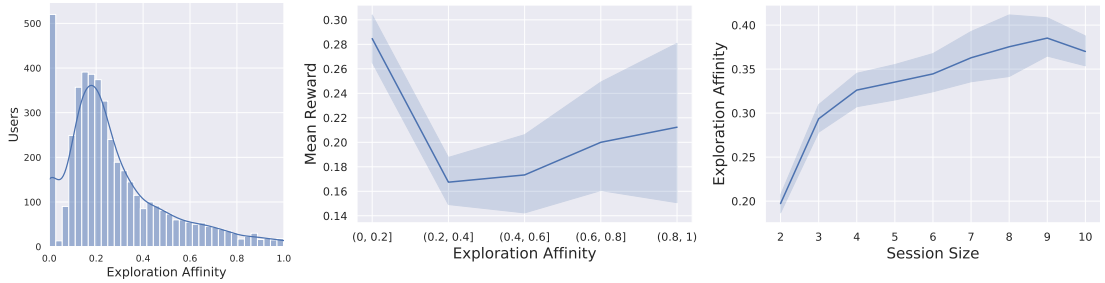
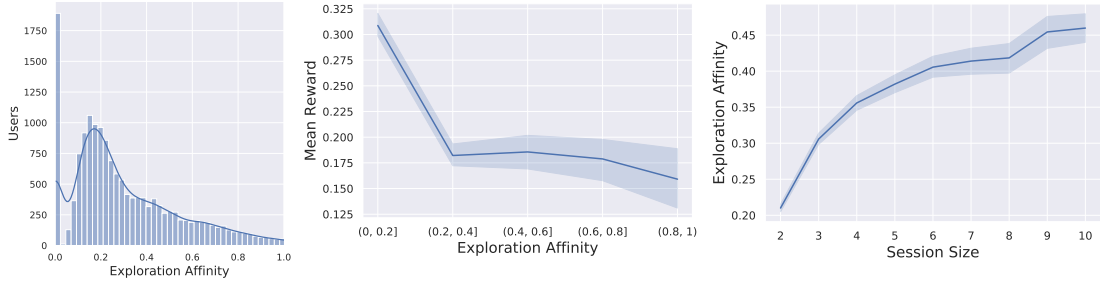


Figura A.5: Métricas da afinidade de exploração para tarefa "RecSys Cities"

(a) Distribuição da afinidade de exploração

(b) Relação com a performance do *bandit*

(c) Relação com o tamanho do histórico de interações do usuário



APÊNDICE B

Avaliação dos métodos baselines com adaptação da restrição de justiça

Para efeito de comparação, os resultados apresentados na Tabela B.1 foram gerados com o φ fixo para todos os *bandits* e tarefas.

Tabela B.1: comparação da média de cinco execuções do ganho em relevância e justiça dos *bandits* com controle de justiça utilizando o $\varphi = 0.2$ em relação a sua versão *vanilla*

	Como, Italy			Chicago, USA		
	CMR	PropFair	UFG	CMR	PropFair	UFG
NeuralUCB-Fair	−48.6%	0.0%	−20.8%	−1.5%	4.1%	3.1%
Adaptive-Greedy-Fair	−5.9%	0.6%	−9.0%	4.9%	1.2%	3.3%
SoftmaxExplorer-Fair	−1.6%	1.7%	−0.2%	−10.8%	4.6%	−0.4%
ExploreThenExploit-Fair	−7.0%	1.2%	−6.4%	−4.4%	2.9%	1.3%
ϵ-Greedy-Fair	−0.6%	1.8%	1.5%	−1.9%	4.0%	3.2%
	Rio de Janeiro, Brazil			New York, USA		
	CMR	PropFair	UFG	CMR	PropFair	UFG
NeuralUCB-Fair	−4.7%	1.1%	−0.4%	1.2%	2.3%	3.0%
Adaptive-Greedy-Fair	−11.6%	0.6%	−3.3%	2.7%	1.7%	2.8%
SoftmaxExplorer-Fair	−9.9%	2.2%	−2.7%	−6.5%	2.8%	−0.8%
ExploreThenExploit-Fair	−5.4%	1.7%	−0.4%	−1.4%	2.2%	1.6%
ϵ-Greedy-Fair	−12.3%	1.1%	−4.3%	−4.6%	2.8%	0.4%
	RecSys Cities					
	CMR	PropFair	UFG			
NeuralUCB-Fair	−16.0%	2.2%	−5.0%			
Adaptive-Greedy-Fair	−33.5%	2.2%	−12.2%			
SoftmaxExplorer-Fair	−34.0%	2.2%	−12.2%			
ExploreThenExploit-Fair	−23.8%	1.7%	−8.3%			
ϵ-Greedy-Fair	−16.8%	1.7%	−5.1%			