Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Unidade Acadêmica de Sistemas e Computação
Programa de Pós-Graduação em Ciência da Computação

Caio Santos Bezerra Nóbrega

Geração de Explicações para Sistemas de Recomendação via Modelos Locais Interpretáveis

CAMPINA GRANDE - PB
2021

Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Geração de Explicações para Sistemas de Recomendação via Modelos Locais Interpretáveis

Caio Santos Bezerra Nóbrega

Tese submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Doutor em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Metodologia e Técnicas da Computação

Leandro Balby Marinho
(Orientador)

Campina Grande, Paraíba, Brasil

N754g Nóbrega, Caio Santos Bezerra.

Geração de explicações para sistemas de recomendação via modelos locais interpretáveis / Caio Santos Bezerra Nóbrega. – Campina Grande, 2021.

123 f. : il. color.

Tese (Doutorado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2021.

"Orientação: Prof. Dr. Leandro Balby Marinho".

Referências.

1. Sistemas de Recomendação. 2. Explicações de Recomendações. 3. Interpretabilidade. 4. Fatoração de Matrizes. 5. Aprendizagem Profunda. I. Marinho, Leandro Balby. II. Título.

CDU 004(043)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECÁRIA MARIA ANTONIA DE SOUSA CRB 15/398

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
POS-GRADUACAO CIENCIAS DA COMPUTACAO
Rua Aprígio Veloso, 882, - Bairro Universitário, Campina Grande/PB, CEP 58429-900

FOLHA DE ASSINATURA PARA TESES E DISSERTAÇÕES

CAIO SANTOS BEZERRA NÓBREGA

GERAÇÃO DE EXPLICAÇÕES PARA SISTEMAS DE RECOMENDAÇÃO VIA MODELOS LOCAIS INTERPRETÁVEIS

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação como pré-requisito para obtenção do título de Doutor em Ciência da Computação.

Aprovada em: 05/02/2021

Prof. Dr. LEANDRO BALBY MARINHO**,** Orientador, UFCG

Prof. Dr. CLÁUDIO ELÍZIO CALAZANS CAMPELO**,** Examinador Interno, UFCG

Prof. Dr. NAZARENO FERREIRA DE ANDRADE, Examinador Interno, UFCG

Prof. Dr. EDLENO SILVA DE MOURA**,** Examinador Externo, UFAM

Prof. Dr. MARCELO GARCIA MANZATO, Examinador Externo, USP

Documento assinado eletronicamente por **LEANDRO BALBY MARINHO**, **PROFESSOR 3 GRAU**, em 08/03/2021, às 10:17, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Marcelo Garcia Manzato**, **Usuário**

Externo, em 09/03/2021, às 11:34, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Edleno Silva de Moura**, **Usuário Externo**, em 09/03/2021, às 11:40, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **NAZARENO FERREIRA DE ANDRADE**, **PROFESSOR(A) DO MAGISTERIO SUPERIOR**, em 09/03/2021, às 12:07, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **CLAUDIO ELIZIO CALAZANS CAMPELO**, **PROFESSOR(A) DO MAGISTERIO SUPERIOR**, em 05/04/2021, às 11:09, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

A autenticidade deste documento pode ser conferida no site https://sei.ufcg.edu.br/autenticidade, informando o código verificador **1263991** e o código CRC **17710EE7**.

Referência: Processo nº 23096.003274/2021-77 SEI nº 1263991

Resumo

O aumento da sofisticação e da complexidade dos modelos de aprendizado de máquina os transformou em caixas-pretas, onde o raciocínio por trás de suas predições é difícil de entender por humanos. Mesmo para tarefas de baixo risco, como recomendação de filmes, mostrar porque uma recomendação foi feita se tornou um requisito importante, dado que pode aumentar a confiança dos usuários e ajudá-los a tomar decisões melhores. Nesse sentido, existe uma crescente necessidade de dotar tais modelos de interpretabilidade, de forma a torná-los claros ou de fácil compreensão. Uma abordagem popular para atingir esse objetivo é conhecida como *post-hoc interpretability*, a qual consiste em separar o processo de geração de explicações do modelo complexo, ou seja, adicionar uma camada de interpretabilidade sobre ele. Nesta tese, propomos investigar métodos *post-hoc interpretability* para sistemas de recomendação complexos. Em particular, propomos uma adaptação do LIME (*Local Interpretable Model-agnostic Explanations*), um método popular de interpretabilidade post-hoc, cujo objetivo é aprender um modelo interpretável, sob um espaço de características interpretáveis na vizinhança da instância que se deseja explicar. Além disso, investigamos como diferentes estratégias de geração de vizinhanças podem impactar a qualidade das explicações. Conduzimos experimentos offline e mostramos que nosso método proposto é uma alternativa promissora, pois é comparável em termos de fidelidade, ou seja, pode imitar localmente o comportamento de um recomendador complexo e tem a vantagem adicional de possibilitar diferentes estilos de explicação. Por fim, mostramos que o histórico de consumo do usuário é a estratégia de vizinhança que melhor se adequa ao nosso método.

Palavras-chave: Sistemas de Recomendação, Explicações de Recomendações, Interpretabilidade, Fatoração de Matrizes, Apredizabem Profunda.

Abstract

The increase in sophistication and complexity of Machine Learning (ML) models has turned them into black boxes where the reasoning behind their predictions is hard to understand by humans. Even for low-risk tasks such as movie recommendation, show why a recommendation was made has become a key requirement since it can potentially improve users' trust and help them to make better decisions. In this sense, there is a growing need to equip such models with interpretability in order to make them clear or easy to understand. A popular approach to achieve this goal is known as *post-hoc* interpretability, which consists of separating the process of generating explanations from the complex model, i.e, adding a layer of interpretability on top of it. In this thesis, we propose to investigate post-hoc interpretability methods for complex recommender systems. In particular, we propose an adaptation of LIME (Local Interpretable Model-agnostic Explanations), a popular *post-hoc* interpretability method, whose objective is to learn an interpretable model, under a space of interpretable features in the neighborhood of the instance being predicted. In addition, we investigate how different neighborhood generation strategies can impact the quality of the explanations. We conduct offline experiments and show that our proposed adaptation is a promising alternative since it is comparable in terms of fidelity, i.e., can locally mimic the behavior of a complex recommender, and has the additional advantage of enabling different styles of explanations. Finally, we show that the user's consumption history is the neighborhood strategy that best suits our approach.

Keywords: Recommender Systems, Recommendation Explanations, Interpretability, Matrix Factorization, Deep Learning.

Agradecimentos

À minha família, aos amigos e ao Prof. Leandro Balby por todo o suporte durante a realização deste trabalho.

Conteúdo

Lista de Siglas

AR Association Rules. 40, 67

DL Deep Learning. 22, 52, 53, 65, 85

DNN Deep Neural Network. 3

FC Filtragem Colaborativa. 5, 14, 15, 51, 59, 80, 87

FM Factorization Machines. 21, 65, 71, 85

GD Gradient Descent. 20

GDPR General Data Protection Regulation. 4

IBCF Item-based Collaborative Filtering. 16

MF Matrix Factorization. 18, 23, 24, 85

ML Machine Learning. 2, 17, 50, 57

SR Sistemas de Recomendação. 1, 12, 13, 35, 46, 54, 61, 71, 74, 80, 86, 90

SSE Sum of Squared Errors. 20

UBCF User-based Collaborative Filtering. xiv, 16, 17

Lista de Figuras

Lista de Tabelas

Lista de Quadros

Capítulo 1

Introdução

Neste capítulo, apresentamos o contexto e a motivação da pesquisa realizada nesta tese. Primeiramente, introduzimos os principais desafios dos Sistemas de Recomendação no que diz respeito à interpretabilidade de suas predições. Em seguida, as questões de pesquisa e as principais contribuições são apresentadas. Por fim, o capítulo é encerrado com uma descrição da organização dos demais capítulos.

1.1 Motivação

Sites de comércio eletrônico, redes sociais, entre outros serviços online, oferecem uma enorme quantidade de opções, com o intuito de satisfazer as necessidades ou preferências dos diversos tipos de consumidores existentes. Logo, sugerir opções relevantes tornou-se um ponto chave para aumentar a satisfação dos usuários, fidelizá-los ao serviço e, consequentemente, impulsionar os lucros nos negócios. Foi esse contexto que fomentou o desenvolvimento de técnicas e ferramentas que filtram as opções disponíveis de forma a expor, aos usuários, as mais relevantes, sendo Sistemas de Recomendação (SR) uma das que obteve maior sucesso e adoção (GOLDBERG et al., 1992; KOREN; BELL; VOLINSKY, 2009).

Sistemas de Recomendação auxiliam os usuários sugerindo itens que eles provavelmente gostariam de consumir. A finalidade é facilitar os processos de decisão que os usuários enfrentam, tais como, qual produto comprar, a qual filme assistir ou qual música ouvir (RICCI; ROKACH; SHAPIRA, 2011). Inicialmente, as pesquisas na área de SR focaram principalmente em predizer de forma acurada os interesses dos usuários, ou seja, recomendar itens

que fossem os mais adequados baseados nas suas preferências. No entanto, posteriormente, estudos observaram que outras dimensões das recomendações devem ser exploradas para alcançar a satisfação do usuário de forma mais ampla. Prover diversidade nas recomendações (MCNEE; RIEDL; KONSTAN, 2006; ZIEGLER et al., 2005) e gerar explicações das recomendações (TINTAREV; MASTHOFF, 2007) são dimensões bem estudadas nesse sentido.

Geralmente, as recomendações são personalizadas pelo fato de que os usuários podem considerar características ou atributos distintos para formar suas preferências pessoais. Por exemplo, em um cenário de recomendação de restaurantes, dois usuários podem ter avaliado o mesmo restaurante com 4 estrelas (em uma escala de 1 a 5). No entanto, a característica que motivou o primeiro pode ter sido a localização, enquanto o preço pode ter motivado o segundo. Sendo assim, quando esses usuários se depararem com uma recomendação, eles, muito provavelmente, usarão diferentes razões para decidir se a aceitam ou não.

Uma forma de ajudar o usuário a decidir se aceita ou não uma recomendação é deixar explícito, transparente e fácil de entender o processo pelo qual o recomendador decidiu sugerir o item em questão. Por esses motivos, sentiu-se a necessidade de equipar recomendadores com mecanismos de explicação. Vários estudos (HERLOCKER; KONSTAN; RIEDL, 2000; TINTAREV; MASTHOFF, 2007, 2012) já mostraram que gerar explicações afeta o grau de transparência e, consequentemente, de interpretabilidade de um SR assim como sua a aceitação e confiança pelos usuários.

De fato, há uma crescente necessidade de interpretabilidade de modelos que vai muito além da área de SR, atingindo a área de Aprendizado de Máquina (em inglês, *Machine Learning (ML)*) como um todo (NUNES; JANNACH, 2017). Essa tendência pode ser justificada pela ampla e bem-sucedida adoção de modelos de aprendizado de máquina para resolver os mais variados tipos de problemas (incluindo SR), que, cada vez mais, afetam diretamente a vida humana. Logo, quanto mais as decisões de uma máquina afetarem a vida de um humano, mais importante será que este seja capaz de interpretar as razões dessas decisões.

Um modelo de aprendizado de máquina é considerado interpretável quando é capaz de explicar ou apresentar suas decisões de forma compreensível a um ser humano (DOSHI-VELEZ; KIM, 2017a). Normalmente, os modelos intrinsecamente interpretáveis são mais simples, logo, tendem a não capturar os padrões mais complexos presentes em muitos problemas do mundo real. Para contornar essa questão, estão sendo desenvolvidos modelos

cada vez mais complexos, que acabam sendo considerados caixas-pretas dada a dificuldade em interpretá-los. Sendo assim, o seguinte *trade-off* se apresenta: normalmente, quanto mais complexos os modelos, mais acurados e menos interpretáveis eles são (SARKAR et al., 2016).

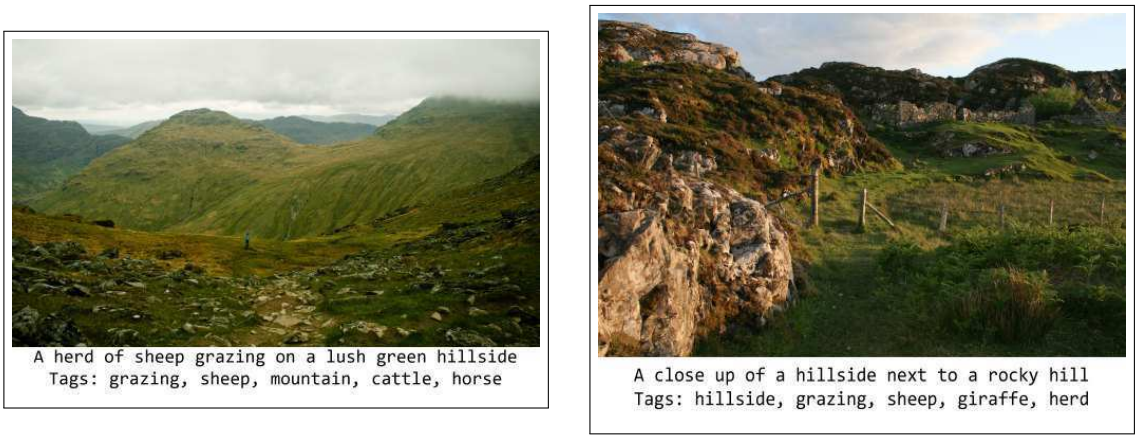


Figura 1.1: Exemplo de classificação equivocada de um modelo complexo. Para ambas as imagens, o modelo associou a classe "ovelha" (em inglês, "*sheep*") mesmo quando nenhuma ovelha está presente.

Um exemplo de problema complexo do mundo real é a classificação de imagens. Os resultados mais acurados para este problema são obtidos por modelos baseados em redes neurais profundas (em inglês, *Deep Neural Network (DNN)*). Normalmente, esses modelos apresentam centenas de camadas e de unidades escondidas, tornando-os caixas-pretas difíceis de interpretar (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

A Figura 1.1 ilustra uma aplicação de um modelo DNN para uma tarefa de associação de *tags* a uma imagem. O modelo em questão é desenvolvido pela Microsoft com acurácia de até 92%[1]. Nesse exemplo, o classificador prediz erroneamente que as imagens de entrada contém a presença de *"ovelhas"*, uma das classes/*tags* do problema. Porém, uma rápida inspeção manual mostra que o classificador errou na sua predição. Possivelmente, o modelo aprendeu alguma correlação espúria, por exemplo, a correlação entre a presença de grama na imagem e a classe *"ovelha"*. Portanto, mesmo apresentando bom desempenho em acurácia, se o usuário não entende exatamente as razões da decisão do modelo, ele pode perder confiança nas suas predições.

[1]Mais detalhes em: https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/concept-tagging-images

A criação de iniciativas governamentais é outro indicativo que comprova esse renovado interesse em promover interpretabilidade em modelos de aprendizado de máquina. Em 2016, a União Europeia lançou a *General Data Protection Regulation (GDPR)*[2], regulamentação que efetivamente cria um "direito à explicação", por meio da qual os seus cidadãos podem pedir uma explicação de uma decisão feita por algum algoritmo que utilizou seus dados pessoais (GOODMAN; FLAXMAN, 2016). O Brasil também criou sua legislação, chamada de Lei Geral de Proteção de Dados Pessoais (LGPD)[3], Lei nº 13.709/2018 — por enquanto, esta foca somente nas questões de privacidade dos dados.

Além da GDPR, em Maio de 2017, foi lançado o programa *DARPA Explainable Artificial Intelligence (XAI)*[4], cujo objetivo é fomentar a pesquisa para desenvolvimento de modelos interpretáveis e ao mesmo tempo acurados. A Figura 1.2 ilustra um exemplo da proposta idealizada por esse programa. Nesse exemplo, predizer corretamente que *"gato"* é o rótulo da imagem de entrada não é suficiente. A saída adequada para um modelo considerado *XAI* também deve incluir as características ou atributos que o modelo levou em consideração para classificar a imagem como *"gato"*. Aqui, os atributos destacados foram pele, bigodes e orelhas, inclusive ilustrados com outras imagens.

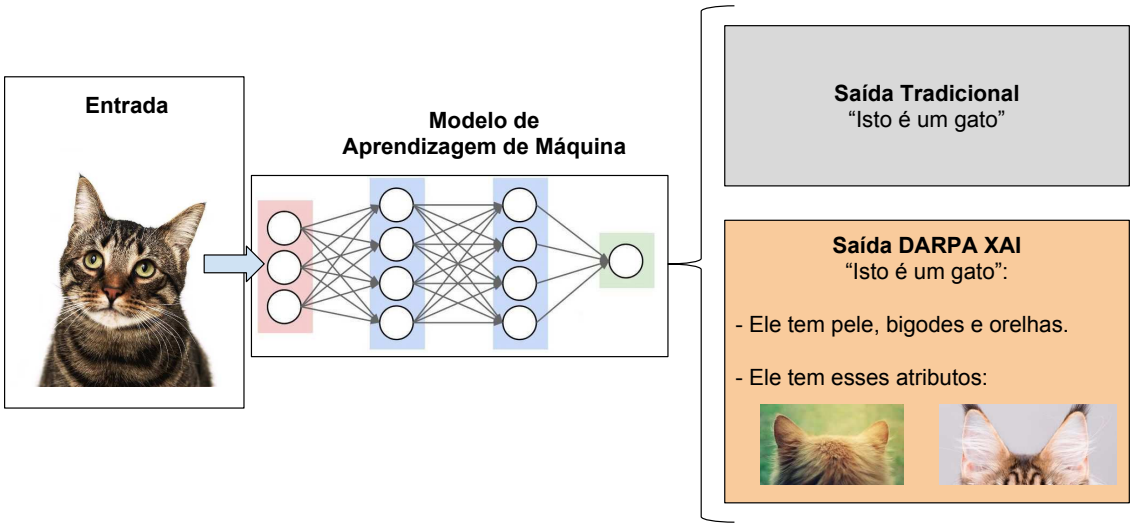


Figura 1.2: Explicações para uma classificação provida por um algoritmo de aprendizado de máquina. Não basta acertar a classe, é preciso prover as explicações que levaram a tomar essa decisão. Exemplo extraído e adaptado do site do programa DARPA XAI[4].

[2]Regulação (EU) 2016/679 : https://www.eugdpr.org/the-regulation.html
[3]https://lgpd-brasil.info/
[4] https://www.darpa.mil/program/explainable-artificial-intelligence

Duas alternativas despontam como solução para garantir interpretabilidade de modelos de aprendizado de máquina. A primeira, já mencionada anteriormente, é a utilização de modelos que são intrinsecamente (ou reconhecidamente) interpretáveis, tais como Regressão Linear e Árvores de Decisão. Nesse caso, a geração de explicações está diretamente ligada à estrutura simples que o modelo de predição escolhido geralmente apresenta e, teoricamente, deve ser uma tarefa trivial. Por exemplo, a interpretação de modelos de regressão linear se resume, tipicamente, à importância dos atributos segundo a magnitude dos pesos aprendidos pelo modelo. Por outro lado, uma comum desvantagem dessa alternativa é a possibilidade de abrir mão da acurácia em troca da facilidade de interpretação. A Seção 2.4.1 detalha essa abordagem.

A segunda alternativa é desassociar o processo de geração de explicações do modelo que gera as predições. Basicamente, a ideia é criar uma camada de interpretabilidade sobre um modelo, por meio da qual são extraídas as explicações. Sendo assim, ganha-se em flexibilidade, pois, à princípio, qualquer modelo pode ser utilizado. Além disso, não é preciso abrir mão de modelos complexos de alta acurácia e difíceis de interpretar. Na literatura, tais técnicas são denominadas de *post-hoc interpretablity* ou *model-agnostic interpretability* (MOLNAR, 2019; GUIDOTTI et al., 2018). A Seção 2.4.2 detalha essa abordagem.

No que diz respeito aos SR, existem diferentes algoritmos e, para muitos deles, não é óbvio interpretar ou explicar a recomendação sugerida. Dentre os algoritmos presentes na literatura, estão os modelos de Filtragem Colaborativa (FC), os quais alcançaram um grande sucesso devido ao bom desempenho em termos de acurácia em diversos domínios (KOREN; BELL; VOLINSKY, 2009). Os dois principais tipos de FC são os algoritmos baseados em memória e os baseados em modelos. Os modelos baseados em memória computam similaridades entre usuários (ou itens) para obter uma vizinhança de usuários que compartilham preferências similares e, a partir delas, realizar recomendações. Por exemplo, na abordagem baseada no usuário, o valor da nota que o usuário u daria ao item i é calculado como uma agregação das notas que alguns usuários similares a u deram ao item i. Modelos baseados em memórias são abordagens *white boxes*, ou seja, podem ser considerados interpretáveis, pois explicações podem ser feitas a partir da vizinhança calculada. Por exemplo, "Sugerimos esse item, pois ele é muito bem avaliado entre pessoas que gostam dos mesmos itens que você".

A abordagem baseada em modelos, por sua vez, tem como principal representante os modelos de fatores latentes. Esses modelos mapeiam conjuntamente usuários e itens para um mesmo espaço euclidiano, onde os fatores são as dimensões, e realizam a recomendação por meio da similaridade entre as representações vetoriais dos usuários e itens nesse novo espaço. Não se sabe o que realmente esses fatores representam, mas supostamente descrevem atributos dos itens, nos quais as preferências dos usuários são baseadas (KOREN; BELL; VOLINSKY, 2009). Por não terem um significado claro, as explicações baseadas nesses fatores são opacas e, portanto, essa família de modelos pode ser considerada *black box*.

Geralmente, produzir explicações compreensíveis por humanos requer informações extras (em inglês, *side information*), tais como atributos que descrevem os itens (MUSTO et al., 2016) ou revisões textuais feitas pelos usuários (CHANG; HARPER; TERVEEN, 2016). Informações extras podem trazer mais detalhes às explicações, tornando mais explícito e fácil de interpretar o processo de decisão do recomendador. De forma semelhante às abordagens mencionadas acima, as informações extras podem ser utilizadas nas formas abaixo:

(a) geração de explicações após o processo de recomendação. Aqui o gerador de explicações é independente do modelo de recomendação subjacente, ou seja, as explicações são geradas de maneira agnóstica ao recomendador subjacente. Sendo assim, há liberdade de escolha tanto do modelo de recomendação quanto do modelo gerador de explicações. A parte a da Figura 1.3 ilustra essa abordagem.

(b) criação de um modelo de recomendação interpretável por definição, também chamado de *interpretable-aware*. Nesses casos, funções de custo que incluem componentes de interpretabilidade são inseridas no processo de treinamento do modelo de recomendação. Dessa forma, as recomendações são geradas acompanhadas de suas explicações. A parte b da Figura 1.3 ilustra essa abordagem.

Há diversas formas de se implementar a abordagem de interpretabilidade *post-hoc*. Uma delas consiste no aprendizado de modelos interpretáveis, tais que estes sejam fiéis aos modelos originais que são eventualmente complexos e difíceis de explicar. A intuição é que, ao utilizar um modelo interpretável para aproximar o comportamento de um modelo complexo, é possível extrair explicações do primeiro para as predições do segundo.

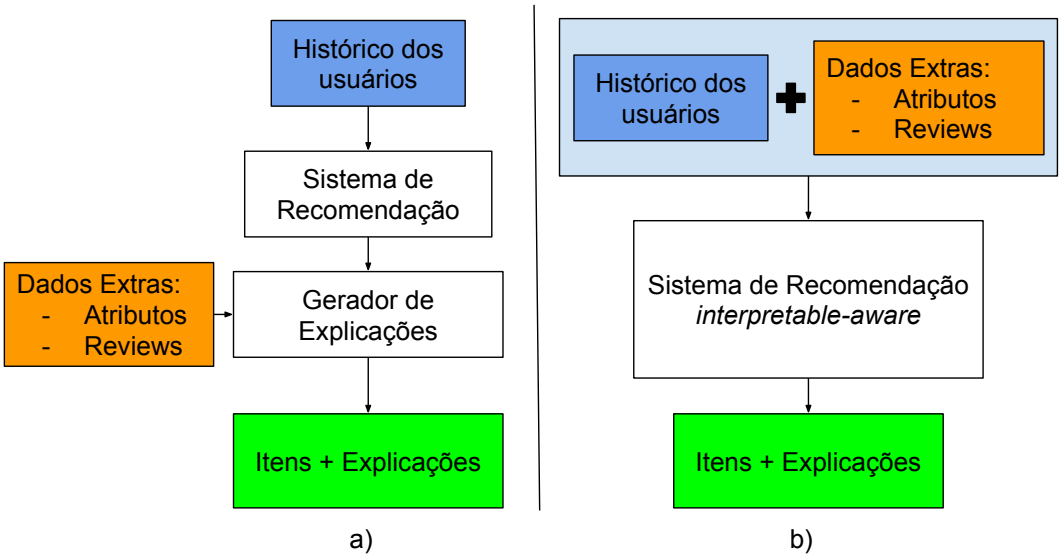


Figura 1.3: Estratégias para geração de explicações de uma recomendação: a) Utilização de informações extras após o treinamento do recomendador; b) Utilização de informações extras durante o treinamento do recomendador. Figura modificada de (ABDOLLAHI; NASRAOUI, 2016a).

Um método popular nessa direção é o *Local Interpretable Model-agnostic Explanations* (LIME) (RIBEIRO; SINGH; GUESTRIN, 2016b), que propõe aprender um modelo interpretável, em um espaço de atributos interpretáveis, na vizinhança de uma dada instância que está sendo predita. Nesse caso, o modelo interpretável pode ser entendido como uma aproximação local do modelo original *black-box*.

Vale ressaltar que os dados e os algoritmos de sistemas de recomendação podem ser muito diferentes dos métodos tradicionais de aprendizagem supervisionada, para os quais o LIME foi originalmente projetado. Enquanto a maioria das abordagens de aprendizado de máquina tomam como entrada vetores de características, muitos algoritmos de recomendação tomam como entrada triplas (usuário, item, avaliação). Sendo assim, para aplicar o LIME em sistemas de recomendação, são necessárias adaptações. Por exemplo, fornecer uma representação adequada para os dados de entrada, assim como fornecer atributos que possam ser usados para explicar uma recomendação. Com o melhor do nosso conhecimento, o LIME ainda não foi aplicado ao domínio de SR.

Em resumo, o problema de pesquisa que esta tese visa investigar é o seguinte: *Como tornar modelos complexos e de difícil interpretação de Sistemas de Recomendação interpretáveis?*. Mais especificamente, o problema é extrair explicações de uma dada recomendação

proveniente de qualquer modelo de recomendação baseados em aprendizado de máquina.

1.2 Objetivos

O objetivo geral desta tese é propor abordagens que promovam a interpretação de modelos *black-boxes* no contexto de sistemas de recomendação. De fato, interpretabilidade de modelos baseados em aprendizado de máquina se tornou um linha de pesquisa bastante ativa, principalmente pelo fato de os modelos estarem cada vez mais complexos e menos interpretáveis. Em um cenário ideal, seria desejável interpretar as predições de quaisquer modelo. Nessa direção, o LIME é um método de interpretabilidade agnóstico ao modelo bastante popular, porém ainda não aplicado para o domínio de sistemas de recomendação. Sendo assim, o objetivo geral pode ser entendido por meio das seguintes questões de pesquisa:

- ***Questão de Pesquisa 1 (QP1): O método agnóstico ao modelo, LIME, é efetivo ao gerar explicações para o domínio de Sistemas de Recomendação?***
  É sabido que há avanços relacionados ao problema de interpretação de modelos *black-box* na área de aprendizado de máquina, principalmente para tarefas de classificação (RIBEIRO; SINGH; GUESTRIN, 2016b; LAKKARAJU et al., 2017). O LIME é um dos métodos mais usados nesse sentido. Entretanto, ele não foi projetado para SR e não há iniciativas ou propostas que tratem desse assunto específico na literatura. Logo, essa questão de pesquisa visa atacar essa lacuna. Vale salientar também que essa questão de pesquisa está relacionada à estratégia da Figura 1.3.

- ***Questão de Pesquisa 2 (QP2): Qual é o impacto da formação da vizinhança no método LIME na qualidade das suas explicações?***
  O método LIME baseia-se na abordagem de modelos locais interpretáveis. Nessa abordagem, um modelo intrinsecamente interpretável tenta imitar o comportamento de um modelo complexo subjacente nas vizinhanças de uma dada instância que se deseja explicar. A literatura aponta que uma de suas fragilidades é a forte dependência na etapa de criação da vizinhança, a qual pode ocasionar explicações que não necessariamente refletem o modelo subjacente (TAN et al., 2019; ALVAREZ-MELIS; JAAKKOLA, 2018). Nosso objetivo é investigar e expandir esse achados para o contexto de siste-

mas de recomendação.

1.3 Contribuições

Para alcançar os objetivos propostos na seção anterior, as seguintes contribuições foram realizadas:

1. Uma adaptação do algoritmo do LIME no trabalho (RIBEIRO; SINGH; GUESTRIN, 2016b). Como dito anteriormente, o LIME não foi projetado para sistemas de recomendação. Basicamente, os principais motivos envolvem a representação dos dados de entrada e etapa da criação da vizinhança. A adaptação proposta atacou esses pontos da seguinte forma. Para a representação dos dados, a ideia foi considerar informações extras sobre os itens como atributos explicáveis para aprender o modelo local interpretável. Para a criação da vizinhança, propomos uma definição de localidade baseada no usuário da instância alvo que se deseja interpretar. A estratégia do LIME original envolve geração de dados sintéticos, o que se provou problemático, como argumentamos em mais detalhes no Capítulo 4, para o cenário de sistemas de recomendação.

2. Uma avaliação experimental offline comparando nossa adaptação descrita na primeira contribuição com uma abordagem do estado da arte baseado em regras de associação. Os principais resultados mostram que nosso método proposto é uma alternativa promissora, pois é comparável em termos de fidelidade, ou seja, pode imitar localmente o comportamento de um recomendador complexo e tem a vantagem adicional de possibilitar diferentes estilos de explicação.

3. Proposta de novas estratégias para geração de vizinhança. Como dito anteriormente, trabalhos já apontaram deficiências no LIME original. O principal fator envolve a etapa da criação da vizinhança. Logo, esse problema foi atacado ao expandir a proposta do item 1, já que esta apresenta uma única forma de geração da vizinhança, a seleção aleatória. Por exemplo, uma das estratégias seleciona itens que sejam similares ao item da instância alvo que se deseja interpretar.

4. Uma avaliação experimental offline comparando as diferentes estratégias da contribuição 3, com o objetivo de analisar o impacto na qualidade das explicações. Os principais

resultados mostram que o histórico de consumo do usuário é a estratégia de vizinhança que apresenta os resultados mais promissores.

As contribuições 1 e 2 foram publicadas no artigo, cujo título é *"Towards explaining recommendations through local surrogate models"* (SAC, 2019) (NóBREGA; MARINHO, 2019).

1.4 Estrutura

O conteúdo restante desta tese está organizado da seguinte maneira:

- **Capítulo 2 - Fundamentação Teórica.** Esse capítulo versa sobre os principais conceitos que embasam essa tese.
- **Capítulo 3 - Trabalhos Relacionados.** Esse capítulo aborda os trabalhos relacionados à geração de explicações em Sistemas de Recomendação.
- **Capítulo 4 - Explicações para uma Recomendação via aprendizado de Modelos Locais Interpretáveis.** Esse capítulo detalha a proposta para explicar uma dada recomendação via aprendizado de modelos locais interpretáveis, além de uma avaliação experimental comparando a proposta com uma alternativa do estado da arte.
- **Capítulo 5 - Análise de Estratégias de Localidade para Modelos Locais Interpretáveis.** Esse capítulo investiga o impacto das novas estratégias propostas para geração de vizinhança, passo fundamental na abordagem de modelos locais. Também foi realizada uma avaliação experimental offline. Esse Capítulo pode ser visto como uma expansão do Capítulo 4.
- **Capítulo 6 - Conclusões e Trabalhos Futuros.** Esse capítulo apresenta as conclusões e limitações desta tese e discute as possibilidades de trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Neste capítulo, apresentamos a fundamentação teórica necessária para o entendimento do nosso trabalho. Inicialmente, sumarizamos a notação utilizada nesta tese. Em seguida, mostramos os conceitos básicos relativos aos Sistemas de Recomendação e uma visão geral das estratégias existentes de implementação (Seção 2.2). Em seguida, apresentamos os estilos e tipos de explicações presentes no contexto de Sistemas de Recomendação (Seção 2.3). Por fim, também são apresentados modelos que são reconhecidamente interpretáveis e métodos que promovem interpretabilidade de forma agnóstica ao recomendador (Seção 2.4).

2.1 Definições e Notações

As principais definições e notações utilizadas em todo o restante desta tese estão resumidas a seguir. Quando necessário, definições ou notações adicionais serão descritas no capítulo onde forem aplicadas.

- U, o conjunto de usuários;
- I, o conjunto de itens;
- I_u, o conjunto de itens avaliados/acessados pelo usuário u;
- U_i, o conjunto de usuários que avaliaram o item i;
- A, o conjunto de atributos (em inglês, *features*) dos itens;
- $r_{u,i}$, avaliação/nota do usuário u ao item i, onde $u \in U$, $i \in I$;
- R, o conjunto de avaliações presentes no históricos de consumo de todos os usuários, tal que $R = \{r_{u,i} \in \mathbb{R}^+ \mid u = 1, 2, ..., |U|,\ i = 1, 2, ..., |I|\}$;

- $\hat{R}$, o conjunto de predições realizadas por modelo de recomendação, tal que $\hat{R} = \{\hat{r}_{u,i} \in \mathbb{R}^+ \mid u = 1, 2, ..., |U|,\ i = 1, 2, ..., |I|\}$;

2.2 Sistemas de Recomendação

Sistemas de Recomendação (SR) são ferramentas de software e algoritmos que sugerem itens que provavelmente o usuário gostaria de consumir (RICCI; ROKACH; SHAPIRA, 2015; RESNICK; VARIAN, 1997). Os itens recomendados podem ser de quaisquer domínios, tais como, filmes, músicas, notícias, produtos, entre outros.

Como dito na Seção 1.1, os SR são especialmente úteis em cenários onde há uma enorme quantidade de informações. O "Paradoxo da Escolha" (SCHWARTZ, 2005) e a "Sobrecarga de Informações" (ROETZEL, 2019) são exemplos de efeitos negativos bem conhecidos associados a esses cenários. Ambos estão relacionados e, basicamente, afirmam que à medida que o indivíduo recebe mais opções de escolha, sua capacidade de tomar decisões se deteriora, ficando menos satisfeito com a decisão tomada. Sendo assim, os recomendadores se tornaram uma maneira efetiva de mitigar tais efeitos negativos, aumentando a satisfação dos usuários, bem como, melhorando a receita de negócios e, por este motivo, são atualmente são peças fundamentais em empresas como Amazon (LINDEN; SMITH; YORK, 2003), Google (DAS et al., 2007) e Netflix (GOMEZ-URIBE; HUNT, 2016).

Basicamente, os SR abordam dois tipos de problemas: predição de avaliações (ou notas) e recomendação de itens. No primeiro caso, é realizada uma predição de um valor numérico, que representa a preferência do usuário por um determinado item. Por exemplo, em um serviço de filmes, no qual o usuário dá notas de 1 a 5 aos filmes vistos, o sistema irá predizer notas a filmes ainda não vistos pelo usuário, baseado no seu histórico de notas. Portanto, nesses casos, as recomendações são baseadas no *feedback* explícito dos usuários, expressado por suas avaliações/notas.

Para o segundo tipo de problema, em vez de predição de avaliações, uma lista de itens é recomendada de acordo com os graus de relevância dos itens nessa lista. Por exemplo, a recomendação de uma lista com os 10 filmes (top-10) com maior chance de interesse, com base no histórico de filmes já assistidos pelo usuário. Nos casos em que as avaliações dos usuários não estão disponíveis, o sistema de recomendação infere as preferências por meio

de *feedback* implícito, o qual é extraído a partir de iterações dos usuários com o sistema, tais como itens acessados, histórico de compra, histórico de pesquisa, entre outros.

Em algumas aplicações, além de fornecer *feedback*, seja de maneira explícita ou implícita, os usuários também podem fornecer comentários (também chamados de *reviews*) ou tags descrevendo o item ou sua interação com o item. As *reviews* textuais podem fornecer informações úteis e enriquecer a construção do perfil de preferência do usuário, mas seu processamento pode exigir técnicas custosas de processamento de linguagem natural.

2.2.1 Classificação de Sistemas de Recomendação

Há dois paradigmas principais na área de Sistemas de Recomendação (ADOMAVICIUS; TUZHILIN, 2005):

- SR baseados em conteúdo;
- SR baseados em Filtragem Colaborativa;

Na abordagem baseada em conteúdo, o SR extrai os atributos (conteúdos) dos itens presentes no histórico dos usuários e constrói um perfil para cada item e para cada usuário em função desses atributos (BALABANOVIć; SHOHAM, 1997). A partir disso, os itens com os perfis mais similares aos perfis dos usuários são recomendados. Por exemplo, no contexto de filmes, os atributos podem ser: atores, diretor e gênero. Se o usuário dá notas altas a filmes de ação, muito provavelmente mais filmes desse gênero serão recomendados. Tags e *reviews* textuais são fontes comuns para extração de atributos dos itens.

Mais formalmente, seja A o conjunto de todos os atributos que os itens podem ter, o perfil $\mathbf{i}$ de um dado item i pode ser descrito como um vetor de pesos da seguinte forma: $\mathbf{i} \in \mathbb{R}^{|A|}$, tal que $\mathbf{i} = (w_1, w_2, ..., w_{|A|})$, onde cada valor w é um peso que representa a relevância de cada atributo $a \in A$ em relação a i. Os valores de w podem ser binários indicando presença/ausência de atributos, valores de frequência, tais como *term frequency/inverse document frequency (TF-IDF)* (BAEZA-YATES; RIBEIRO-NETO, 1999), ou outra estratégia que meça a força da relação entre o item e seus atributos. De forma semelhante, o perfil $\mathbf{u}$ de um dado usuário u também pode ser descrito como um vetor de $|A|$ dimensões. Umas das formas de construção de $\mathbf{u}$ consiste em somar os perfis/vetores de cada item presente no

seu histórico. A intuição é que se um certo atributo aparece em muitos itens no histórico do usuário, esse atributo vai ser mais importante do aqueles que aparecem em poucos itens.

Para realizar a recomendação para um dado usuário u, avalia-se a similaridade entre o seu perfil $\mathbf{u}$ com o perfil de cada item ainda não consumido por u, de forma que os itens cujos valores de similaridade foram mais altos são recomendados. Uma das formas de computar a similaridade entre vetores é a chamada similaridade do cosseno, descrita na Equação 2.1, onde $||.||_2$ representa a norma L2 (HORN; JOHNSON, 2012).

$$cos(u, i) = \frac{\mathbf{u} \cdot \mathbf{i}}{||\mathbf{u}||_2||\mathbf{i}||_2} \tag{2.1}$$

Essa abordagem não necessita de uma grande base de usuários para funcionar, pois basta que esses tenham alguns itens em seus históricos de consumo e que os atributos dos itens estejam disponíveis. Logo, essa abordagem tem a vantagem de lidar com o problema do item novo (em inglês, *item cold-start problem*) (SCHEIN et al., 2002), pois é capaz de recomendar novos itens mesmo que nenhum usuário os tenha consumido. Por outro lado, ela também apresenta desvantagens conhecidas. Uma delas é que os atributos dos itens podem estar indisponíveis ou a sua extração pode não ser trivial, por exemplo, a extração automática de atributos de *streams* de áudio ou vídeo. Outra desvantagem é a super especialização, que representa o efeito colateral do sistema ficar muito especializado no perfil do usuário, de forma que falha ao não recomendar itens mais diversos ao já consumidos no passado, mas que o usuário gostaria de consumir. Por exemplo, um dado usuário costuma assistir à filmes de ação e aventura, mas também poderia se interessar em recomendações de filmes de suspense.

Diferentemente da abordagem baseada em conteúdo, a Filtragem Colaborativa (FC) baseia-se na identificação de usuários com preferências similares a um dado usuário alvo. O termo colaborativo decorre justamente do fato de não ser focada em informações externas, mas somente nos itens presentes no histórico de consumo dos usuários. A ideia central é que usuários que gostaram de itens semelhantes no passado, gostarão de itens semelhantes no futuro (DESHPANDE; KARYPIS, 2004). Em outras palavras, se dois usuários deram notas similares aos mesmos itens no passado, então provavelmente continuarão a dar notas similares a outros itens no futuro.

O ponto forte da FC é a capacidade de sugerir itens que surpreendam os usuários. Por

exemplo, ainda no contexto de filmes, um usuário que tem um perfil voltado a filmes de ação pode receber uma recomendação de um filme de suspense. Isso se deve ao fato de que outros usuários com perfil similar, ou seja, que gostam de filmes de ação, também podem ter gostado do filme de suspense.

Por outro lado, um problema comum à FC é a esparsidade dos dados, devido ao fato dos usuários normalmente avaliarem/acessarem apenas uma pequena fração dos itens disponíveis. Em muitos SR reais, é comum encontrar altos níveis de esparsidade, onde mesmo usuários ativos avaliam bem menos que 1% dos itens (SARWAR et al., 2000). Outro problema relacionado à esparsidade dos dados é o chamado *cold-start*, que ocorre quando há pouco ou nenhum registro de transações de um dado usuário (*user cold-start*) ou de um dado item (*item cold-start*). Esses cenários dificultam a captura das preferências e ocasionam recomendações pouco acuradas (ou até mesmo a incapacidade de gerar recomendações).

A Filtragem Colaborativa pode ser dividida em algoritmos baseados em memória e algoritmos baseados em modelos. Essas subcategorias são detalhadas nas subseções a seguir. A Figura 2.1 resume os dois paradigmas principais em Sistemas de Recomendação.

Por fim, a literatura ainda cita abordagens híbridas, formuladas com o intuito de combinar as vantagens das abordagens baseadas em conteúdo e filtragem colaborativa (BASILICO; HOFMANN, 2004; LI; KIM, 2003; BURKE, 2007). Por exemplo, o problema *item cold-start* presente nos métodos baseados em FC pode ser mitigado com o auxílio das abordagens baseadas em conteúdo. Ainda, os sistemas de recomendação *context-aware*, assim chamados porque utilizam informações tais como contexto temporal (além de atributos do itens), também são comumente categorizados como abordagens híbridas.

FC baseada em Memória

Essa abordagem é focada no cálculo de correlações entre os usuários (ou entre itens), de forma que a relevância de um item i para um usuário alvo u é baseada na agregação das interações dos n usuários mais similares ao usuário alvo u e que já consumiram i (ADOMAVICIUS; TUZHILIN, 2005). Mais formalmente, temos:

$$t_{u,i} = \underset{v \in V_u}{\operatorname{aggr}}\, t_{v,i} \tag{2.2}$$

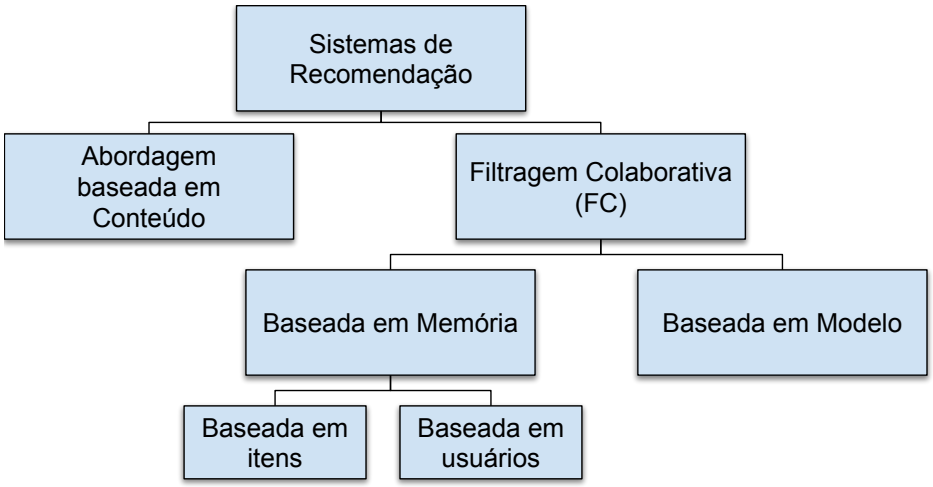


Figura 2.1: Classificação resumida dos principais paradigmas em Sistemas de Recomendação.

onde $t_{u,i}$ é o valor que representa a relevância de i para u, e V_u representa o conjunto de usuários mais similares a u. Num cenário de predição de avaliações, a Equação 2.2 pode ser instanciada por meio da Equação 2.3, na qual a nota que usuário u avaliaria o item i, i.e, $\hat{r}_{u,i}$ é uma média ponderada das notas que os n usuários mais similares a u deram a i.

$$\hat{r}_{u,i} = \frac{\sum_{v \in V_u}^{n} sim(u,v) * r_{v,i}}{\sum_{v \in V_u}^{n} sim(u,v)} \tag{2.3}$$

Aqui a função sim calcula a similaridade entre o usuário alvo u e o seu vizinho v. Uma possível instanciação da função sim é a função de similaridade do cosseno, descrita na Equação 2.1.

Essa abordagem também é conhecida como Filtragem Colaborativa baseado no usuário (em inglês, *User-based Collaborative Filtering (UBCF)*), pois a computação foi baseada numa vizinhança de usuários similares a u. Contudo, a literatura também apresenta a abordagem Filtragem Colaborativa baseado no item (em inglês, *Item-based Collaborative Filtering (IBCF)*), na qual a vizinhança é baseada nas interações de u com os itens mais similares a i, ou seja, a vizinhança é construída em torno de i. Aqui não há fase de treinamento, porém todos os dados devem estar na memória (RAM) para, de forma rápida, realizar a computação da agregação.

O Quadro 2.1 demonstra o pseudocódigo do algoritmo de Filtragem Colaborativa baseada no usuário para predição de uma nota. A primeira etapa consiste em obter as similari-

dades entre os usuários, calculada par a par e é baseada nas avaliações dos seus respectivos históricos de consumo presentes na matriz R. Em seguida, é obtida a lista dos n usuários mais similares a um dado usuário alvo u (linha 5 do pseudocódigo). Essa lista é comumente chamada de vizinhança. Por fim, a predição da nota $\hat{r}_{u,i}$ que o usuário alvo u daria a um dado item i é computada como uma média ponderada baseada nas avaliações dos usuários da vizinhança e nas respectivas similaridades.

Quadro 2.1: Pseudocódigo do algoritmo de Filtragem Colaborativa baseada em Usuário (UBCF) para predição de uma nota.

Input: Matriz R, número de vizinhos n, usuário alvo u, item i
Output: Predição $\hat{r}_{u,i}$
01: similarities = []
02: weight, norm = 0
03: **foreach** $v \in U$ u:
04: similarities[v] = $sim(u, v)$
05: $V_u = \arg\max^n_{v \in V}$ similarities[v]
06: **foreach** $v \in V_u$:
07: weight = weight + $r_{v,i} * sim(u, v)$
08: norm = norm + $sim(u, v)$
09: $\hat{r}_{u,i} = weight/norm$

Um ponto positivo é a facilidade que esta estratégia possibilita ao explicar as predições/recomendações. As explicações podem ser feitas a partir da vizinhança calculada. Por exemplo, "Sugerimos esse item, pois ele é muito bem avaliado entre pessoas que gostam dos mesmos itens que você".

FC baseada em Modelos

Na abordagem chamada Filtragem Colaborativa baseada em modelos, métodos de ML e mineração de dados são utilizados no contexto de aprendizado de modelos preditivos (AGGARWAL, 2016). Árvores de decisão, modelos baseados em regras, métodos Bayesianos, redes neurais e modelos de fatores latentes são exemplos de métodos de aprendizado de máquina utilizados.

Diferentemente da abordagem baseada em memória, a fase de treinamento (ou construção) do modelo é claramente separada da fase de predição. Normalmente, para se obter o modelo, há uma função de custo que precisa ser otimizada a partir de um treinamento reali-

zado em cima dos dados de treino disponíveis. A vantagem é que, uma vez que o modelo é aprendido, as predições podem ser feitas mais rapidamente. A seguir, detalhamos os modelos baseados em fatores latentes e modelos baseados em técnicas de *Deep Learning*. Esses são os modelos usados como instâncias de modelos caixa-preta nos experimentos dos Capítulos 4 e 5.

Modelos baseados em Fatores Latentes

Dentre os modelos listados acima, a família de modelos baseados em fatores latentes se tornou bastante popular pois geralmente apresenta uma melhor performance quando comparada aos métodos baseados puramente em vizinhança ou conteúdo (KOREN; BELL; VOLINSKY, 2009). Destaque para os métodos baseados em Fatoração de Matrizes (em inglês, *Matrix Factorization (MF)*).

As técnicas baseadas em MF assumem que apenas um pequeno número de fatores (características) dos itens influenciam as preferências dos usuários. Sendo assim, elas consistem em mapear conjuntamente usuários e itens para esse espaço de fatores de dimensionalidade reduzida. Por exemplo, num cenário de filmes, os itens podem ser mapeados nos seguintes fatores latentes: romance e ação. Logo, os valores dos fatores representam o grau de preferência (por parte do usuário) ou o grau de associação (por parte do item) aos fatores. Vale salientar que, na prática, esses fatores são opacos, ou seja, não possuem interpretação clara. O exemplo mencionado é meramente ilustrativo.

Dado que um Sistema de Recomendação pode ser visto como um problema de completar os valores ausentes de uma matriz que armazena interações entre usuários e itens, por exemplo, suas notas a filmes assistidos, a MF atua da seguinte forma. A matriz Usuário-Item denotada por R, a qual representa as interações do conjunto R em forma matricial, é decomposta no produto de duas matrizes densas $P \in \mathbb{R}^{|U| \times k}$ e $Q \in \mathbb{R}^{|I| \times k}$, onde k indica o número de fatores latentes e, tipicamente, $k \ll |U|$ e $k \ll |I|$. Logo, a matriz R indica o espaço original de interações entre usuários e itens. Consequentemente, as matrizes P e Q indicam o mapeamento de R no espaço de fatores latentes.

Após a fatoração de R, cada usuário u estará associado a um vetor $p_u \in \mathbb{R}^k$ e cada item i estará associado a um vetor $q_i \in \mathbb{R}^k$. Note que cada elemento dos vetores corresponderá a um fator latente. Uma vez mapeados para o mesmo espaço de k fatores, a proximidade de

itens a usuários indica a relevância desses itens para esses usuários. Essa proximidade pode ser capturada por meio do produto escalar entre os respectivos vetores **p** e **q**, de forma que $R \approx P \times Q^T$. Então, a relevância de um item i para um usuário alvo u pode ser formulada da seguinte maneira:

$$t_{u,i} = p_u \cdot q_i^T = \sum_{f=1}^{k} p_{u,f} \times q_{f,i} \tag{2.4}$$

A Figura 2.2 ilustra um exemplo simplificado da fatoração de matrizes para um espaço de fatores latentes de dimensão 3. A matriz original contém as avaliações dos usuários (representadas pelas células verdes) e em destaque está o valor que será predito (célula vermelha). As linhas em destaque representam como seria calcular a predição para o usuário 1 (representado pela primeira linha da matriz P) em relação ao item 3 (representado pela terceira coluna da matriz Q), ou seja, calcular o $\hat{r}_{1,3}$.

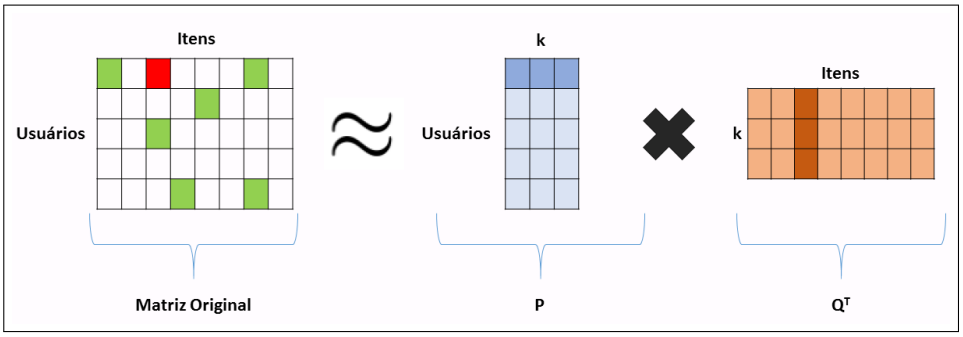


Figura 2.2: Ilustração da Fatoração de Matrizes como solução para completar valores ausentes na matriz original.

Além de ser possível encontrar novas interações entre usuários e itens, como mostrado no exemplo da Figura 2.2, relações mais significativas podem ser descobertas, tais como, relações entre dois usuários, embora eles tenham histórico de consumo totalmente diferentes (DESROSIERS; KARYPIS, 2011).

Dada a finalidade de aproximar a matriz original R à matriz resultante do produto de P e Q, a Fatoração de Matrizes pode ser formalizada como um problema de otimização da seguinte forma:

$$\arg\min_{P,\ Q} erro(t, R) \tag{2.5}$$

A formulação acima responde a seguinte questão: "Quais são os melhores valores de P e Q, tal que o erro entre os valores observados R e suas respectivas predições calculadas por $P \times Q^T$ seja o menor possível?". Portanto, o treinamento do modelo da fatoração de matrizes consiste em achar os parâmetros P e Q, de acordo com a Fórmula 2.5. Para o problema de predição de avaliações, a função $erro$ normalmente utilizada é a soma dos erros quadráticos (em inglês, *Sum of Squared Errors (SSE)*), descrita na Equação 2.6.

se

$$erro(t, R) = SSE = \sum_{r_{u,i} \in R} e_{u,i}^2 = \sum_{r_{u,i} \in R} (r_{u,i} - p_u \cdot q_i^T)^2 \tag{2.6}$$

Uma vantagem das técnicas baseadas em Fatoração de Matrizes é sua flexibilidade em lidar com requisitos que dependem da aplicação em questão. Por exemplo, tipicamente, em dados de aplicações baseadas em Filtragem Colaborativa, usuários e itens apresentam vieses, ou seja, alguns usuários tendem a avaliar com notas mais altas que outros, e itens tendem a receber notas mais altas que outros. Essa particularidade dos dados pode não ser capturada pela interação usuário-item, descrita na Equação 2.4. Uma das extensões possibilitada pela flexibilidade desses modelos é a inclusão desses vieses na estimativa da relevância de um item i para um usuário alvo u. Portanto, a Equação 2.4 pode ser adaptada da seguinte maneira para o problema de predição de notas

$$\hat{r}_{u,i} = \mu + b_u + b_i + p_u \cdot q_i^T \tag{2.7}$$

onde μ representa uma média global de todas as avaliações presentes em R, b_u e b_i representam os vieses do usuário e do item, respectivamente.

Técnicas de otimização são utilizadas para encontrar os valores que compõem P e Q. Uma técnica bastante utilizada é a chamada gradiente descendente (em inglês, *Gradient Descent (GD)*), popularizada para o contexto de recomendação por Simon Funk (FUNK,) e aplicada com sucesso em vários outros trabalhos (KOREN, 2008; PATEREK, 2007; SALAKHUTDINOV; MNIH; HINTON, 2007; TAKáCS et al., 2007).

O algoritmo GD consiste em percorrer todas as instâncias no conjunto de treinamento, calculando o erro associado, por exemplo, $e_{u,i} := r_{u,i} - \hat{r}_{u,i}$. Então, modificam-se os vetores p_u e q_i por um valor proporcional e na direção oposta de seus gradientes. O Quadro 2.2 demonstra o pseudocódigo do algoritmo gradiente descendente.

Quadro 2.2: Pseudocódigo do algoritmo Gradiente Descendente

Input: Matriz R, número de fatores latentes k, taxa de aprendizagem α
Output: Matrizes P e Q otimizadas
01: inicializa matrizes P e Q
02: **repeat**
03: **For each** $r_{u,i} \in R$
04: $p_u = p_u - \alpha(e_{u,i}q_i)$
05: $q_i = q_i - \alpha(e_{u,i}p_u)$
06: **Until** atingir critério de parada

Como mencionado anteriormente, devido à sua flexibilidade, técnicas baseadas em Fatoração de Matrizes se tornaram ponto de partida para extensões e adaptações. Máquinas de Fatoração (em inglês, *Factorization Machines (FM)*) é uma das extensões destacadas na literatura por se tratar de um modelo mais genérico de fatoração que inclui a MF, como aqui apresentada, como caso especial (RENDLE, 2010).

As principais vantagens do modelo de Máquinas de Fatoração são: permitir incluir informações extras dos usuários ou itens (ex., atributos) e permitir níveis de interação entre essas variáveis (usuários, itens e atributos) mais sofisticadas. A MF, descrita neste capítulo, só permite interações de ordem 2 (usuários e itens).

Formalmente, dado um vetor de entrada $\mathbf{x} \in \mathbb{R}^n$, a FM prediz um valor para a variável resposta y ao modelar interações esparsas entre todas as combinações possíveis de atributos presentes nos dados de treino. A Equação 2.8 define uma FM de ordem 2:

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \mathbf{v}_i^T \mathbf{v}_j \cdot x_i x_j \tag{2.8}$$

onde $w_0 \in \mathbb{R}$ é um viés global e pode ser comparado ao μ da Equação 2.7; $\mathbf{w} = (w_i, \ldots, w_n) \in \mathbb{R}^n$ representa os pesos de cada atributo e atuam de maneira similar aos vieses b_u e b_i da Equação 2.7; e o termo $\mathbf{v}_i^T \mathbf{v}_i$ modela a interação entre o atributo i e o atributo j, aprendido via a fatoração, de forma que $v_i \in \mathbb{R}^k$ é vetor com k dimensões de fatores latentes do atributo i. Os parâmetros w_0, $\mathbf{w}$ e $\mathbf{V} \in \mathbb{R}^{n \times k}$ podem ser aprendidos por meio de métodos de otimização, tais como o Gradiente Descendente. Note que a formulação acima é de ordem 2, pois considera apenas interações entre pares de atributos.

A Figura 2.3 ilustra os dados de entrada para um modelo de Máquinas de Fatoração. Aqui as tuplas apresentam as seguintes informações: a interação usuário-item por meio do

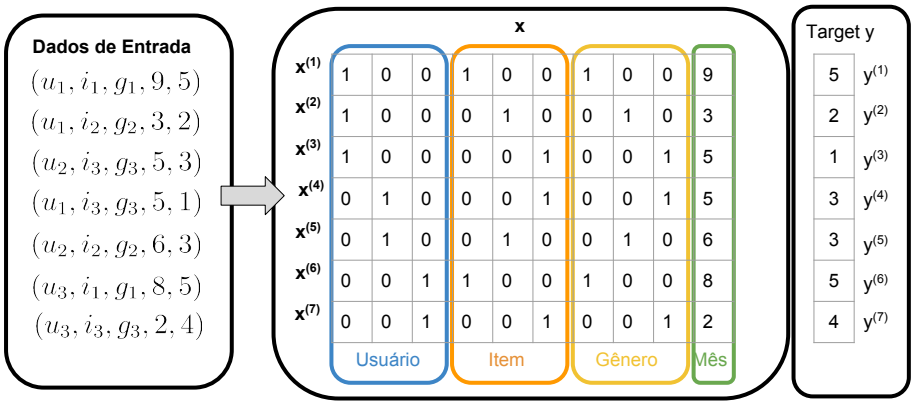


Figura 2.3: Exemplo de dados de entrada para um modelo de Máquinas de Fatoração. Figura extraída e adaptada de (RENDLE, 2010)

.

par (u, i); o gênero $g_i \in F$ do item da interação; o instante que aconteceu a interação usuário-item (dado numérico que corresponde ao número de meses após Janeiro de 2010); e a nota que o usuário avaliou o item. Note que todas as entradas foram mapeadas num vetor de atributos $\mathbf{x}^{(*)}$ que contém todos os dados da transação. As variáveis categóricas (as entidades usuário, item, gênero) foram convertidas para variáveis indicadores binárias. Por fim, a nota que usuário avaliou foi separada como variável resposta que o modelo deve tentar aprender.

Modelos baseados em *Deep Learning*

Devido ao grande sucesso de técnicas de Deep Learning (DL) em diversas áreas, tais como visão computacional (VOULODIMOS et al., 2018) e processamento de linguagem natural (OTTER; MEDINA; KALITA, 2018), o interesse tanto da academia quanto da indústria em aplicar tais técnicas no domínio de Sistemas de Recomendação tem aumentado com o passar do anos.

No que diz respeito à academia, desde 2016, a principal conferência internacional na área de recomendação, RecSys[1], tem organizado um *workshop*[2] dedicado exclusivamente à SR baseados em DL. Da mesma forma, muitas empresas têm adotado esses modelos para elevar a qualidade dos seus serviços de recomendação. Por exemplo, o Yahoo utilizou métodos de DL e conseguiu aumentar a taxa de cliques (em inglês, *click-through rate (CTR)*) em

[1] https://recsys.acm.org/
[2] https://recsys.acm.org/recsys16/dlrs/

23% no seu sistema de distribuição de notícias (OKURA et al., 2017). O Youtube também relatou que uma proposta de uma arquitetura baseada em redes neurais profundas superou os resultados obtidos com abordagens baseadas em MF, principalmente em termos de tempo assistido de vídeo (COVINGTON; ADAMS; SARGIN, 2016).

Mais recentemente, (ZHANG et al., 2019) fez uma revisão da literatura sobre modelos de recomendação baseados em DL. Também é proposto uma forma de categorizar os trabalhos em duas dimensões: "Modelo baseado em redes neurais" e "Modelo integrado". A primeira é a dimensão que distingue modelos que utilizam apenas uma única técnica de DL, por exemplo, redes neurais recorrentes (HIDASI et al., 2016) ou redes neurais convolucionais (WANG et al., 2017), daqueles que utilizam mais de uma (LEI et al., 2016). A segunda dimensão considera se o modelo integra técnicas de DL com técnicas tradicionais, tais como MF (WANG; WANG; YEUNG, 2015), ou se o modelo recomenda utilizando exclusivamente técnicas de DL.

De acordo com as dimensões descritas acima, o trabalho de (CHENG et al., 2016) pode ser categorizado como um modelo baseado exclusivamente em uma única técnica de DL, neste caso, a *Multilayer Perceptron (MLP)* (GOODFELLOW; BENGIO; COURVILLE, 2016). Inicialmente aplicado para o domínio de recomendação de aplicativos na Google Play[3], esse trabalho é considerado pioneiro e influente, pois introduziu a ideia *Wide & Deep Learning*, cuja proposta é combinar, em um treinamento conjunto, um componente baseado em modelos lineares e um componente baseado em redes neurais profundas. O componente *wide* consiste em uma rede neural de uma única camada e pode ser entendido como uma modelo linear generalizado, que é capaz de capturar padrões diretamente dos atributos originais presentes nos dados de treinamento ou daqueles obtidos por transformações desses. Por exemplo, atributos originais podem ser a categoria do aplicativo, por exemplo "jogo", e o idioma. Logo, um atributo derivado seria a combinação entre eles. Já o componente *deep* é uma MLP, que mapeia os atributos originais em vetores densos de valores reais e de baixa dimensão, também chamado de *embedding*. Os *embeddings* podem capturar representações mais abstratas e gerais, aumentando assim o poder de generalização do modelo. A Figura 2.4 ilustra ambos os componentes presentes nesse modelo.

Independentemente de como são posicionados nessas dimensões, pode-se assumir que

[3]https://play.google.com/store

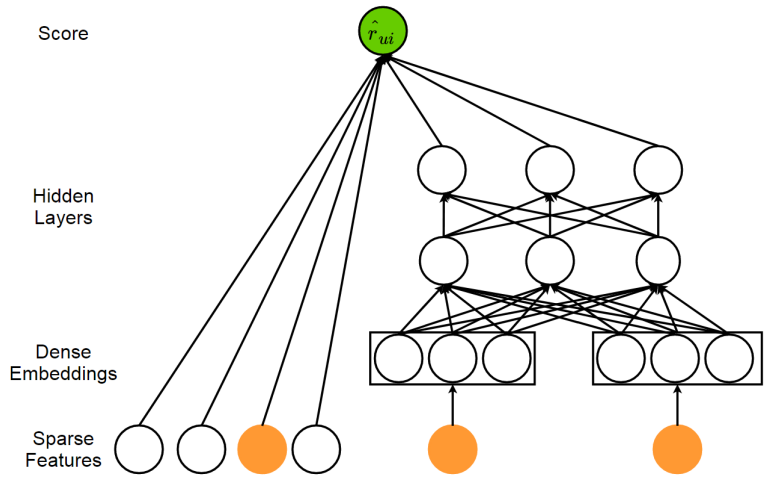


Figura 2.4: Ilustração do modelo de recomendação baseado em Deep Learning Wide&Deep. A parte da esquerda representa o componente *wide*, enquanto a parte da direita o *deep*. Figura extraída de (ZHANG et al., 2019).

recomendadores baseados em DL não são interpretáveis por definição. O Capítulo 3 discute trabalhos que tentam minimizar esse problema.

Ademais, também vale destacar que outras análises (DACREMA; CREMONESI; JANNACH, 2019; LIN, 2019) indicam que algumas abordagens de recomendadores baseados em DL não superam o desempenho de outros métodos mais simples, porém bem-sucedidos, e estabelecidos há mais tempo na literatura, por exemplo a MF, quando estes são cuidadosamente bem ajustados. Há, portanto, uma crítica e alerta para que experimentos sejam realizados com um maior rigor, de forma que o progresso de DL na área de sistemas de recomendação seja mais consistente e claro.

2.3 Explicações em Sistemas de Recomendação

É sabido que fornecer explicações para recomendações pode aumentar a confiança e aceitação dos usuários nos Sistemas de Recomendação (BILGIC; MOONEY, 2005; PU; CHEN, 2006). De fato, muitos SR comerciais fornecem explicações simples para os usuários com o intuito de atingir esses objetivos. O SR da Amazon mostra itens semelhantes que o usuário (ou outros usuários semelhantes) compraram ou visualizaram ao recomendar um novo item. A explicação é sucinta: "Clientes que compraram este item também compraram os itens X, Y e Z". De maneira similar, o SR da Netflix também justifica suas sugestões de filmes listando filmes similares, como mostra a Figura 2.5.

Uma série de diferentes taxonomias de explicações já foram propostas na literatura (PAPADIMITRIOU; SYMEONIDIS; MANOLOPOULOS, 2012; VIG; SEN; RIEDL, 2009; NUNES; JANNACH, 2017). Essas taxonomias abrangem diversos aspectos diferentes relacionados às explicações. A seguir, detalhamos três deles: estilos, tipos e objetivos de explicações.

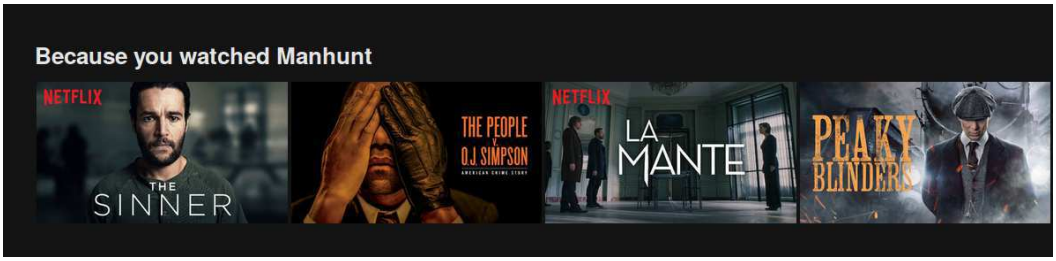


Figura 2.5: Exemplo de uma explicação no serviço Netflix.

2.3.1 Estilos de Explicações

Segundo a taxonomia proposta por (PAPADIMITRIOU; SYMEONIDIS; MANOLOPOULOS, 2012), as explicações de recomendações podem ser categorizadas, basicamente, em quatro estilos diferentes: Humano, que provê explicações com base em usuários similares; Item, que se baseia nas escolhas feitas por um usuário em itens similares; Atributo, que explica a recomendação com base nos atributos de um item avaliado pelo usuário previamente; e Híbrido, que é uma combinação qualquer dos outros estilos. A Tabela 2.1 lista exemplos para cada um dos diferentes estilos.

Tabela 2.1: Estilos de explicações

#	Estilo	Exemplo
1	Humano	Este livro foi recomendado porque clientes com perfis similares ao seu também compraram.
2	Item	Este livro foi recomendado porque você comprou o livro X.
3	Atributo	Este item foi recomendado porque trata de história e guerra, atributos que você geralmente está interessado.
4	Híbrido	Este livro foi recomendado porque você aprovou livros similares e outros clientes com perfis similares ao seu também compraram.

Para cada um dos estilos, é possível criar diferentes designs de interfaces com o usuário (HERLOCKER; KONSTAN; RIEDL, 2000). As Figuras 2.6, 2.7 e 2.8 exemplificam designs

de interface para os estilos humano, atributo e híbrido, respectivamente, para a recomendação do filme *"Star Wars: The Force Awakens"*.

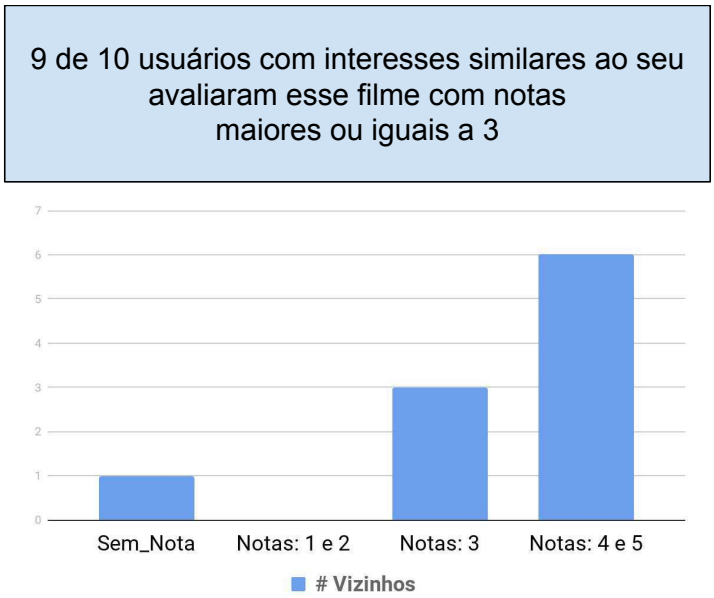


Figura 2.6: Exemplo de interface para o estilo "Humano". O histograma resume o comportamento dos vizinhos mais similares do usuário alvo. Figura adaptada de (ABDOLLAHI; NASRAOUI, 2016b).

Figura 2.7: Exemplo de interface para o estilo "Atributo". A nuvem de *tags* exibe os atributos que o usuário alvo tem mais e menos interesse em azul e vermelho, respectivamente. Figura adaptada de (GEDIKLI; JANNACH; GE, 2014).

2.3.2 Tipos de Explicações

A literatura também categoriza as explicações em dois tipos: *white-box* e *black-box* (FRIEDRICH; ZANKER, 2011). As explicações *white-box* descrevem como o recomendador funciona, ou seja, descrevem as entradas do recomendador e os passos que o levaram a uma

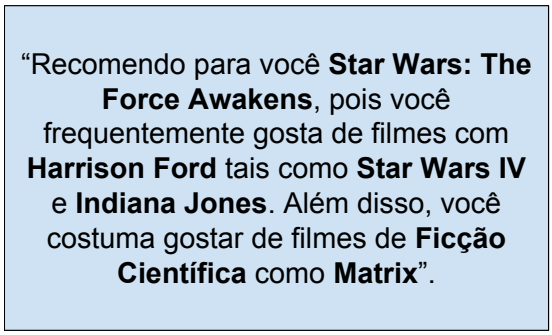


Figura 2.8: Exemplo de interface para o estilo "Híbrido". O template é preenchido tanto com itens quanto com atributos. Figura inspira em um exemplo de explicação de (MUSTO et al., 2016).

determinada saída. Logo, esse tipo de explicação torna o usuário ciente do funcionamento interno do algoritmo. Por exemplo, a Figura 2.6 pode ser considerada um exemplo de explicação *white-box*, pois torna explícita a vizinhança que gerou a recomendação/predição da nota. Note que, mesmo sem detalhes de como a vizinhança foi formada, ao saber quantos e quais as notas dos usuários similares, o usuário-alvo tem uma noção mais clara de como foi computada a predição e, portanto, uma ideia de como um recomendador baseado em vizinhança funciona.

Por outro lado, explicações *black-box* proveem justificativas para as saídas dos algoritmos sem entrar em detalhes do seu funcionamento. Logo, esse tipo de explicação serve para que os usuários verifiquem se o algoritmo está computando as saídas de acordo com seus objetivos ou necessidades. Nesse caso, os usuários ficam mais confortáveis e satisfeitos (SINHA; SWEARINGEN, 2002). Por exemplo, a Figura 2.7 pode ser considerada um exemplo de explicação *black-box*, pois mostra que os perfis de usuário e item são próximos quando se considera um espaço de atributos, muito embora um recomendador possa utilizar outros mecanismos para gerar as recomendações.

2.3.3 Objetivos de Explicações

Dada a diversidade de critérios para avaliar uma explicação, Tintarev e Masthoff (TINTAREV; MASTHOFF, 2007) propuseram sete diferentes objetivos para o uso de explicações em SR. Eles estão detalhados a seguir:

- **Transparência:** prover ao usuário a lógica por trás da recomendação, ou seja, prover **como** a recomendação foi escolhida. Por exemplo, para um recomendador cuja abordagem é baseada em conteúdo, a sentença *"Recomendamos o filme Dunkirk porque este contém a tag guerra e você gostou de outros filmes com essa mesma tag"* explica, em alto nível, que o atributo guerra foi o principal fator por trás da recomendação.

- **Escrutinabilidade:** permitir que o usuário possa alterar o comportamento do recomendador. Nesse caso, as explicações fazem parte de um ciclo, onde o usuário tem uma noção de como o recomendador se comporta e tem a oportunidade de controlar e corrigir as suas premissas. Por exemplo, em cenário de recomendações de hotéis/acomodações, considere a seguinte explicação *"Recomendamos este hotel porque você pode gostar da localização e do preço."*. Em um recomendador que provê escrutinabilidade, graças à explicação, o usuário poderia configurá-lo de forma a dar mais importância ao preço do que à localização.

- **Confiança:** aumentar a confiança do usuário pelo recomendador. Normalmente, a confiança está relacionada à transparência e a possibilidade de interação com o sistema, além da acurácia das suas recomendações. Por exemplo, um usuário pode ficar mais confiante nas recomendações, se entender por que uma recomendação ruim (ou com baixa confiança) foi feita e conseguir assim evitar que ela ocorra novamente.

- **Efetividade:** Ajudar usuários a tomar boas decisões, ou seja, ajudá-los a aceitar recomendações acuradas e descartar recomendações irrelevantes. Note que, por definição, a efetividade está fortemente relacionada à acurácia do recomendador, uma vez que este precisa aprender as preferências do usuário para dar suporte a sua tomada de decisão. A efetividade de uma explicação pode ser medida como a diferença entre a relação antes e depois do consumo de uma recomendação. Por exemplo, em um cenário de recomendação de livros, suponha que um usuário avalia uma recomendação em dois momentos, uma ao receber a explicação da recomendação e outra após ler o livro. Se a opinião/avaliação do usuário não for mudar muito, pode-se afirmar que a explicação foi efetiva, já que o usuário consumiu um item que realmente acabou gostando.

- **Persuasão:** persuadir o usuário a comprar ou experimentar uma recomendação, sendo esta acurada ou não. Geralmente, a persuasão tende a beneficiar mais o sistema em si do que os usuários, o que pode acabar sendo um efeito negativo no longo prazo, caso os usuários percebam que estão sendo manipulados. A persuasão de uma explicação pode ser medida ao comparar a diferença entre as avaliações de um item recomendado com e sem explicação. Se avaliação ao prover uma explicação for maior que a avaliação sem explicação, indica que o objetivo da persuasão foi atingido.

- **Eficiência:** Ajudar usuários a tomar decisões mais rapidamente. A eficiência pode ser obtida ou melhorada quando o usuário entende a relação entre os itens presentes em lista de recomendação. Por exemplo, em um cenário de recomendação de celulares, a explicação *"Mais memória RAM e uma maior resolução da câmera, porém mais caro"* pode ajudar um usuário a achar mais rapidamente um celular que atenda seus critérios. Uma forma de avaliar a eficiência consiste em medir o tempo que um usuário levou a achar o produto desejado.

- **Satisfação:** Aumentar satisfação do usuário de forma geral com o recomendador. Uma forma de avaliar satisfação é medir a lealdade ou engajamento dos usuários. Se os usuários utilizam o sistema com maior frequência, pode-se assumir que o uso do sistema é uma experiência agradável, portanto, tem um efeito positivo na satisfação do usuário. No entanto, diferentes fatores podem impactar de maneira diferente os usuários. Por exemplo, prover explicações efetivas pode ajudar os usuários por um lado, mas também pode aumentar o esforço cognitivo, diminuindo a satisfação geral.

2.4 Interpretabilidade de Modelos de Aprendizado de Máquina

Interpretabilidade, no contexto de modelos de aprendizado de máquina, pode ser definida como a capacidade de explicar as causas de uma predição de forma inteligível para um humano (DOSHI-VELEZ; KIM, 2017b). Dessa forma, quanto mais interpretável um modelo for, mais fácil será para um humano compreender as suas decisões. Logo, gerar explicações para as decisões/predições de um modelo é uma maneira de prover interpretabilidade.

Como mencionado na Seção 1.1, é possível obter interpretabilidade de modelos de aprendizado de máquina basicamente de duas maneiras: utilizando modelos que são intrinsecamente interpretáveis, isto é, modelos cuja estrutura é facil de entender e a extração de explicações é uma tarefa trivial, realizada direta e exclusivamente neles; ou por meio de métodos que adicionam uma camada de interpretabilidade a um modelo qualquer, isto é, métodos que analisam um modelo após o seu treinamento e utilizam artifícios para extrair explicações. Nessa segunda maneira, as explicações não são obtidas diretamente do modelo que se deseja entender as predições. As seções seguintes detalham essas alternativas.

2.4.1 Modelos Interpretáveis

A maneira mais fácil e óbvia de se obter interpretabilidade é a utilização de modelos que são intrinsecamente interpretáveis. Esse subconjunto de modelos pode combinar acurácia satisfatória com a vantagem de serem facilmente interpretáveis. Contudo, normalmente, acurácia e interpretabilidade são objetivos conflitantes, ou seja, em muitos cenários é necessária a escolha de um modelo complexo para obter uma acurácia melhor, sacrificando a interpretabilidade das suas predições (SARKAR et al., 2016).

A literatura em interpretabilidade de modelos de aprendizado de máquina normalmente considera os modelos de Regressão Linear e Árvore de Decisão como interpretáveis (MOLNAR, 2019). A seguir, detalhamos como funcionam esses modelos e como é possível interpretá-los.

Regressão Linear

Regressão linear (HASTIE; TIBSHIRANI; FRIEDMAN, 2001) é uma abordagem simples, oriunda da área de estatística, cujo objetivo é predizer uma variável resposta numérica. Assume-se que existe uma relação linear entre a variável alvo que se deseja prever e as variáveis independentes, chamadas de preditores. Entender os fundamentos da regressão linear é essencial para entender outros métodos mais complexos de aprendizado de máquina, pois muitos destes são generalizações ou extensões de uma regressão linear simples.

Formalmente, a regressão linear pode ser definida da seguinte maneira. Sejam y a variável resposta alvo e X o conjunto com p variáveis preditoras, temos que a relação entre X e

y pode ser descrita pela Equação 2.9.

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \epsilon \tag{2.9}$$

Logo, a predição se dá como uma soma ponderada entre as p variáveis preditoras. β_i são pesos associados a cada preditor $x_i \in X$ e são chamados de coeficientes ou parâmetros da regressão linear. O β_0 é chamado de coeficiente linear e não está associado a nenhum preditor. Por fim, o termo ϵ representa o erro irredutível do modelo, ou seja, a porção dos dados que não pode ser modelada pelo modelo de regressão.

O treinamento de um modelo de regressão linear consiste em, dado um conjunto de dados, estimar valores de $\beta_0, \ldots, \beta_p$ tal que a modelo se ajuste bem a esse conjunto de dados. A forma mais usual para realizar esse treinamento utiliza o algoritmo dos mínimos quadrados (semelhante à Equação 2.5). A Figura 2.9 ilustra uma regressão linear.

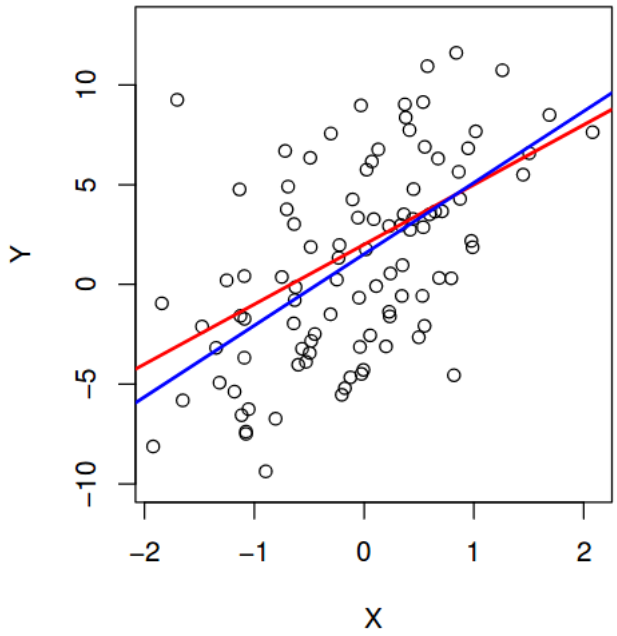


Figura 2.9: Ilustração de uma regressão linear. A reta vermelha indica função linear verdadeira (e usualmente desconhecida) entre X e y. Enquanto a reta azul é a estimativa usando o algoritmo de mínimos quadrados a partir dos dados observados, pontos pretos. Figura extraída do livro (JAMES et al., 2014).

A interpretação de um coeficiente β_i é a seguinte: considerando todos os demais coeficientes fixos, um aumento de uma unidade num preditor x_i aumenta a estimativa de y em β_i unidades, caso este seja positivo, ou decresce caso seja negativo.

Dados os coeficientes $\beta_0, \ldots, \beta_p$ aprendidos após o treinamento do modelo, obter uma explicação para um predição se torna uma tarefa trivial. A explicação é dada pela importância que cada preditor e seu respectivo coeficiente teve para a computação da predição e pode ser medida por métricas como *p-valor*. Então, é possível ordenar os preditores em uma ordem decrescente de importância, tornando explícito quais aqueles que são mais correlacionados com a predição. Para a explicação do modelo como um todo, basta observar os coeficientes isoladamente. Vale salientar que essa interpretação é válida quando todos preditores estão numa mesma escala (ou normalizados), pois isso afetará o peso aprendido para cada coeficiente.

Caso o problema seja predizer uma variável categórica, ou seja, caso a tarefa seja uma classificação, existe uma variação chamada regressão logística, por meio da qual é modelada a probabilidade de y pertencer a uma dada categoria.

Árvore de Decisão

Árvore de decisão é um dos algoritmos mais populares em aprendizado de máquina (BREIMAN et al., 1984). Na verdade, esse algoritmo é base para uma família de modelos baseados em árvores, tais como *random forests* (HO, 1995; BREIMAN, 2001) e *gradient boosted trees* (FRIEDMAN, 2001).

O treinamento de uma árvore de decisão segue uma abordagem gulosa de divisão e conquista. Novamente, sejam y a variável resposta alvo e X o conjunto de variáveis preditoras, tal que $|X| = p$. Dado um conjunto de entrada, é feita uma pesquisa entre os preditores X de tal forma que é escolhido o preditor que, naquele instante, melhor particione os dados de entrada nas respectivas classes (no caso de um problema de classificação). Existem diferentes critérios para avaliar a escolha do preditor, tais como, *Information Gain* e *Gini Impurity*. Esse processo de divisão é repetido recursivamente nas partições geradas até atingir um critério de parada predeterminado.

A estrutura final do modelo é uma árvore, na qual existem dois tipos de nó: nó interno, que representam uma decisão baseado em um preditor; e nó folha, que representa a predição para a variável resposta.

A Figura 2.10 exemplifica uma estrutura de árvore de decisão, treinada a partir de um conjunto com 17 instâncias descritas em duas *features*: x_1 e x_2. Nesse exemplo, há duas

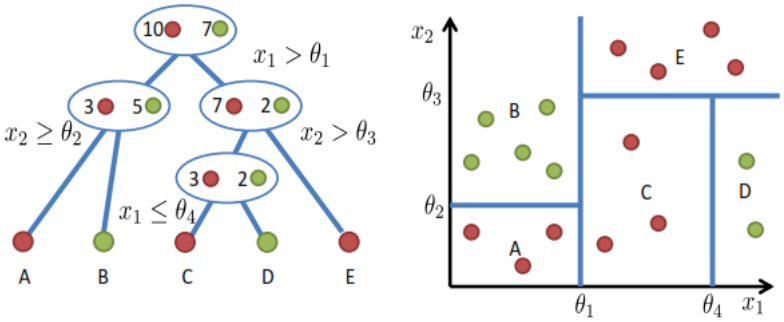


Figura 2.10: Ilustração de uma árvore de decisão. A imagem da esquerda mostra a estrutura da árvore após o processo recursivo de particionamento binário. A imagem da direita mostra as regiões e fronteiras de decisão correspondentes do mesmo particionamento.

classes representadas pelos pontos verdes e vermelhas. O primeiro preditor escolhido foi x_1 e definido o seu respectivo *threshold* θ_1, logo se tornou o nó raiz. Então, os dados de treino são particionados em dois sub-grupos disjuntos, com 8 e 9 instâncias. Esse processo de particionamento binário é repetido até atingir um critério de parada. Nesse exemplo hipotético, foi possível atingir um particionamento perfeito, ou seja, os nós folhas A, B, C, D e E contém apenas instâncias de uma única classe. Ainda, as árvores de decisão podem ser utilizadas tanto para a tarefa de classificação quanto para regressão (HASTIE; TIBSHIRANI; FRIEDMAN, 2001).

A interpretação para uma predição se dá por meio da navegação nos nós internos da árvore. Logo, a explicação é uma concatenação das decisões em cada nó interno. Por exemplo, considere o problema de decisão *"hoje é um bom dia para brincar fora de casa?"*. Aqui, os atributos são *Clima*, *Umidade* e *Vento*. Se um determinado dia foi ensolarado e com alta umidade, então não é um bom dia para sair. Note que podemos chegar a essa decisão (ponto vermelho) pela navegação dos nós internos da árvore, ilustrada na Figura 2.11.

2.4.2 Métodos Agnósticos ao Modelo

Para muitas tarefas, modelos intrinsecamente interpretáveis podem não ser expressivos o suficiente para capturar padrões complexos, impactando negativamente sua performance em termos de acurácia. Por essa razão, modelos complexos (mas opacos) tendem a ser escolhidos. Como dito anteriormente, nesses casos, o ganho em acurácia vem acompanhado da perda de interpretabilidade.

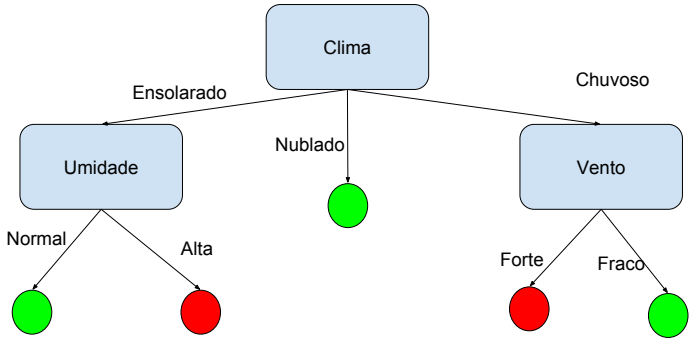


Figura 2.11: Ilustração de uma árvore de decisão para julgar se é um dia bom para brincar fora de casa.

Para contornar esse problema, a literatura apresenta como alternativa tratar a interpretabilidade de maneira desassociada do modelo de predição (LIPTON, 2018; MOLNAR, 2019). Dessa forma, o modelo de predição é tratado como caixa-preta e métodos à parte são aplicados para explicar as predições produzidas por esse modelo caixa-preta. Em outras palavras, as explicações são geradas de maneira independente, ou seja, agnóstica ao modelo de predição, que foi treinado anteriormente (por isso, essa abordagem também é chamada de *post-hoc interpretability*).

A característica de ser agnóstica ao modelo faz com que essa abordagem tenha a grande vantagem da flexibilidade: quaisquer modelos, complexos ou não, podem ser adotados para a tarefa de predição. Essa característica é especialmente atrativa para empresas/serviços, pois o custo de atualização ou mudança se seus modelos, que tendem a ser sofisticados e complexos, é minimizado. Além disso, modelos com estruturas e abordagens completamente diferentes podem ter suas explicações comparadas, pois estas podem ser geradas pelo mesmo método *post-hoc*. Note que também há flexibilidade por parte dos métodos *post-hoc* de geração de explicações.

De fator, há vários e diferentes métodos de interpretabilidade *post-hoc* (GUIDOTTI et al., 2018; MOLNAR, 2019; CARVALHO; PEREIRA; CARDOSO, 2019). Dentre eles, a abordagem baseada em modelos locais se tornou bastante popular. Ela consiste em utilizar um modelo intrinsecamente interpretável como aproximação local e fiel (também chamado na literatura de *local surrogate*) do modelo de predição subjacente que se deseja interpretar. Os métodos *LIME* (*Local Interpretable Model-agnostic Explanations*) e *Explanation Mining* seguem essa abordagem, sendo o último um dos pioneiros a aplicá-la no contexto de SR.

Esses métodos, detalhados a seguir, são relevantes para esta tese, uma vez que propomos uma adaptação do LIME para o contexto de SR e o comparamos com o *Explanation Mining*.

***Local Interpretable Model-agnostic Explanations* (LIME)**

O método *LIME*, proposto por (RIBEIRO; SINGH; GUESTRIN, 2016b), gera explicações de forma agnóstica para qualquer modelo, seja classificador ou regressor. A sua intuição é, dada uma instância que se deseja explicar, aproximar um modelo simples e interpretável usando as predições realizadas por um modelo assumidamente complexo para a vizinhança dessa instância. Argumenta-se que, embora o modelo interpretável possa não representar o modelo original e complexo de maneira global, aproximá-lo localmente, ou seja, tentar imitar o seu comportamento nas proximidades de uma dada instância alvo é uma tarefa viável.

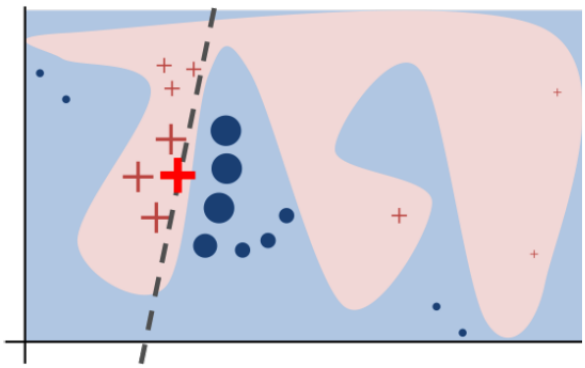


Figura 2.12: Exemplo da intuição do LIME. Um modelo interpretável, representado pela reta, é localmente fiel à fronteira de decisão (regiões azul e rosa) aprendida por um modelo mais complexo. Figura extraída do artigo do LIME (RIBEIRO; SINGH; GUESTRIN, 2016b).

A Figura 2.12 ilustra a intuição do LIME em alto nível. As regiões azul e rosa representam a fronteira de decisão de um classificador binário não linear e, portanto, complexo. O ponto vermelho "+" em destaque é a instância alvo que se deseja explicar. Os demais pontos são amostrados aleatoriamente e suas respectivas classes preditas, representadas pelos formatos "+" em tom rosa e "o" em tom azul, formam a vizinhança da instância alvo. Por fim, a linha tracejada representa o modelo interpretável, treinado com os dados da vizinhança, visando aproximar o comportamento local do classificador binário.

Mais formalmente, temos que, a partir de uma instância alvo $x \in \mathbb{R}^d$ de d dimensões, construímos um conjunto de dados de treinamento nas redondezas de x, porém considerando

um espaço de atributos que sejam interpretáveis ou que seja uma versão simplificada de R^d. Em seguida, aprendemos um classificador ou regressor sobre esse conjunto de dados interpretável e o usamos para explicar por que a instância x foi predita da maneira como foi pelo modelo complexo original $f : \mathbb{R}^d \rightarrow \mathbb{R}$. Note que, como detalhado na Seção 2.4.1, para extrair os top-n atributos de um modelo tal como um modelo de regressão logística, precisamos, a princípio, recuperar as atributos com os top-n pesos em ordem decrescente de magnitude. Os detalhes de cada passo do LIME estão listados a seguir:

1. Geração de uma representação interpretável $x' \in \{0,1\}^{|A|}$ da instância alvo x, utilizando um conjunto A de atributos interpretáveis. Esse passo visa satisfazer o requisito de que as explicações sejam inteligíveis por humanos. Por exemplo, para a tarefa de classificação de imagens, uma representação interpretável de uma imagem seria um vetor binário, indicando se uma determinada região da imagem (conjunto de pixels) está presente ou ausente. Enquanto que o classificador subjacente pode trabalhar com informações mais complexas, como um vetor que representa os canais RGB de cada pixel. Logo, explicações em termos de componentes interpretáveis, como regiões da imagem, são mais fáceis de entender do que em termos de intensidades de RGB.

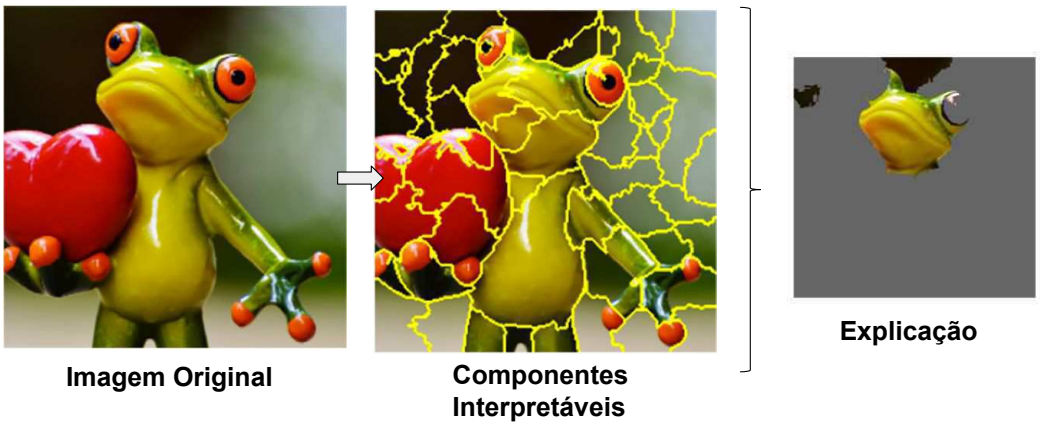


Figura 2.13: Exemplo do mapeamento de uma instância em uma sua representação interpretável. A explicação consiste em um subconjunto dos componentes interpretáveis.[4]

A Figura 2.13 ilustra esse exemplo de mapeamento. Aqui a imagem original (instância alvo) é particionada em várias áreas, onde cada área é considerada um componente interpretável. A explicação extraída de um modelo interpretável, treinado a partir desse

[4]Figura adaptada de https://www.oreilly.com/learning/introduction-to-local-interpretable-model-agnostic-explanations-lime

conjunto de componentes interpretáveis, será um subconjunto desses componentes, de forma que um humano identifique facilmente as razões por trás da decisão do classificador. Aqui uma possível explicação é formada por componentes perto da região da face do sapo.

Logo, antes de construir o modelo interpretável propriamente dito, precisamos definir uma função de mapeamento $h : \mathbb{R}^d \rightarrow \{0, 1\}^{|A|}$ utilizada para transformar x em x'.

2. Geração de uma vizinhança ao redor da instância alvo x por meio de perturbações em sua representação interpretável x'. Cada perturbação $z' \in \{0, 1\}^{|A|}$ é uma amostra ao redor de x' e é definida por meio de elementos não-nulos de x' amostrados aleatoriamente. Cada perturbação recebe um peso proporcionalmente à sua similaridade com x'. O conjunto de perturbações é denotado por Z.

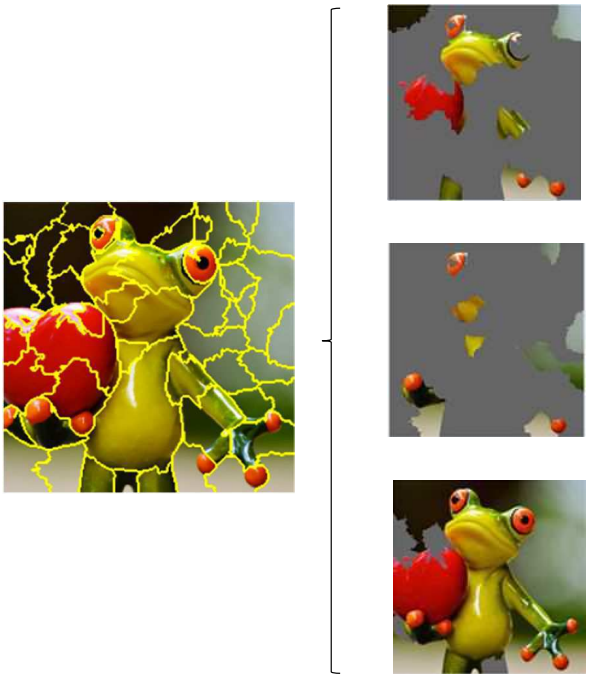


Figura 2.14: Exemplo de perturbações em uma representação interpretável de uma imagem. Cada perturbação contém um subconjunto de componentes interpretáveis.

A Figura 2.14 ilustra uma geração de perturbações para a tarefa de classificação de imagens. Aqui cada perturbação corresponde a uma nova imagem, contendo um subconjunto de componentes interpretáveis.

3. Gerar predições para x e para cada perturbação $z' \in Z$, utilizado o modelo original

f. Para tanto, é preciso mapear cada $z' \in Z$ de volta a sua respectiva representação original por meio de h^{-1}, uma vez que o modelo subjacente f não foi treinado baseado no conjunto de representações interpretáveis. Esse passo é feito com o objetivo de coletar as predições para as instâncias que compõem a vizinhança de x. Portanto, agora temos um conjunto de treinamento, no qual as instâncias são compostas por atributos interpretáveis e suas predições, e que pode ser utilizado para construir um modelo localmente próximo g.

4. Para capturar a noção de localidade, cada instância $z \in Z$ é ponderada por uma função de proximidade $\pi_x(z)$ (por exemplo, distância euclidiana ou do cosseno) que mede a proximidade entre z e x. Na verdade, no artigo original do LIME, a função de proximidade é aplicada juntamente com uma função de *kernel* exponencial com o intuito de suavizá-la. A Equação 2.10 define $\pi_x(z)$. A intuição é que agora aprenderemos a prever da mesma forma que f, mas usando apenas informações locais e interpretáveis em torno de x.

$$\pi_x(z) = exp(-D(x,z)^2/\sigma^2) \tag{2.10}$$

onde D é uma função de distância, por exemplo a distância do cosseno, e σ é um parâmetro chamado *kernel width*, que controla o quanto a suavidade que a função kernel deve aplicar, de forma que quanto maior σ for, mais suave será.

5. Por fim, é possível definir um modelo fácil de interpretar que seja uma função linear de atributos binários (ou interpretáveis) da seguinte forma:

$$g(x') = \theta_0 + \sum_{i=1}^{|A|} \theta_i x'_i \tag{2.11}$$

onde $\theta_i \in \mathbb{R}$ e x'_i é o i-ésimo atributo interpretável de x'. Dessa forma, os atributos mais relevantes são os que apresentam os maiores valores absolutos de θ, assumindo que o LIME tenta garantir que $g(x') \approx f(x)$. Os parâmetros θ são aprendidos por

meio do processo de minimização da seguinte função objetivo:

$$\arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g) \tag{2.12}$$

onde G é o conjunto de modelos interpretáveis possíveis e Ω controla a complexidade do modelo interpretável (por exemplo, via regularização ridge or lasso).

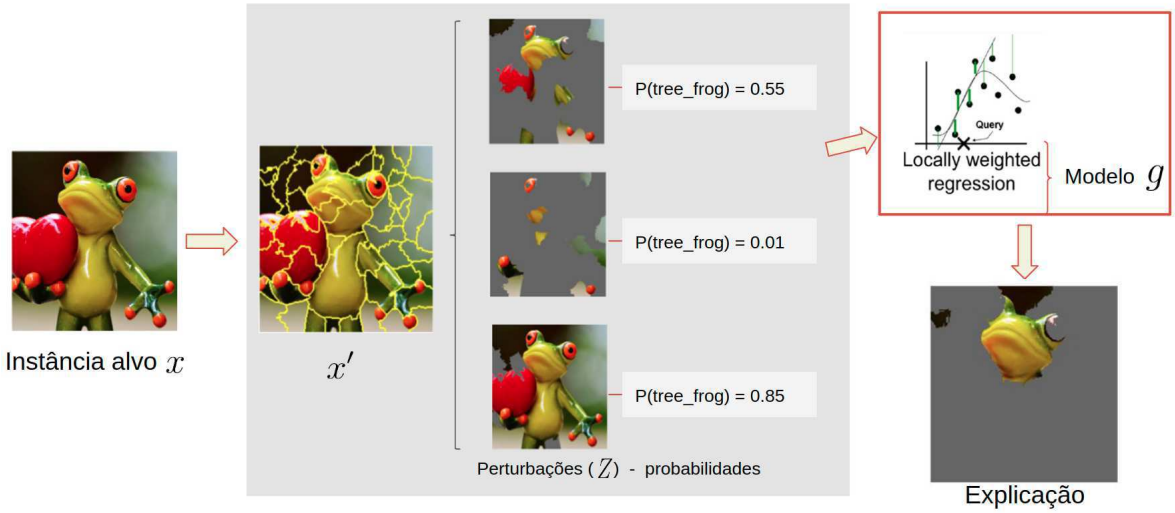


Figura 2.15: Exemplo dos passos do LIME[5]. Deseja-se obter uma explicação para a classificação da imagem de entrada como *"tree frog"*. São mostrados exemplos de perturbações que vão compor o conjunto de treinamento para uma modelo local interpretável, por meio do qual é extraída a explicação.

A Figura 2.15 resume, em alto nível, os passos listados acima. Nesse exemplo, deseja-se saber quais as razões que fizeram o modelo subjacente classificar a imagem como *"tree frog"* com probabilidade de 0,54. Baseado na imagem de entrada, outras imagens são geradas, de forma que sejam perturbações daquela. Então, utiliza-se o modelo subjacente para obter a classificação de cada perturbação. Por fim, o comportamento do modelo subjacente é aproximado por um modelo intrinsecamente interpretável, o qual é treinado com a imagem de entrada, as perturbações e suas respectivas classificações. Aqui foi escolhida uma regressão logística, já que a tarefa é uma classificação. Como dito na Seção 2.4.1, é possível interpretar uma regressão observando os pesos. Nesse exemplo, o componente com maior peso corresponde à área da face do sapo, o que dá indícios que o classificador subjacente está

[5]Figura adaptada de https://www.oreilly.com/learning/introduction-to-local-interpretable-model-agnostic-explanations-lime

de fato aprendendo uma característica relevante na hora de classificar uma imagem como "*tree frog*". Se o componente com maior peso correspondesse à uma área de background da imagem, daria indícios que o classificador não aprendeu o que realmente caracteriza um sapo.

Vale salientar que o LIME se aproxima do conceito proposto no programa *DARPA Explainable Artificial Intelligence (XAI)* (vide Seção 1.1) e é a base da solução proposta por esta tese no Capítulo 4.

Explanation Mining

Em se tratando de métodos *post-hoc interpretability* aplicados no contexto de Sistemas de Recomendação, o trabalho (PEAKE; WANG, 2018) pode ser considerado um dos pioneiros, com o melhor do nosso conhecimento. Nesse trabalho, os autores utilizam uma abordagem baseada em modelos locais, onde um modelo baseado em regras de associação (em inglês, *Association Rules (AR)*) foi escolhido modelo intrinsecamente interpretável para aproximar o comportamento de um modelo de Fatoração de Matrizes.

Mineração de regras de associação (AGRAWAL; IMIELIńSKI; SWAMI, 1993) é um método utilizado para descobrir relações ou padrões frequentes entre elementos presentes em grandes bases de dados. Mais especificamente, o objetivo é encontrar elementos que implicam na presença de outros elementos em uma mesma transação. Por exemplo, a regra $\{cebolas, batatas\} \Rightarrow \{carne\}$, possivelmente encontrada em dados de vendas em supermercados, indica que se um consumidor comprou cebolas e batatas, é provável que também irá comprar carne.

Existem diferentes formas para avaliar se uma dada regra de associação é mais interessante que outra. A medida suporte indica quão frequente é uma dada transação $X \subseteq T$ no conjunto de todas as transações T. Por exemplo, seja 5 o número de transações do conjunto T, se a transação $\{cebolas, batatas\}$ aparece uma vez, então $suporte(\{cebolas, batatas\}) = 1/5 = 0,2$. Já a medida confiança indica quão frequente uma dada regra $X \Rightarrow Y$ foi considerada verdadeira. Mais formalmente, a confiança pode ser computada como $suporte(X \cup Y)/suporte(X)$. Por exemplo, o valor de confiança da regra $\{cebolas, batatas\} \Rightarrow \{carne\}$ é 1 se toda vez que um consumidor comprou cebolas e batatas, também comprou *carne*.

No contexto de SR, regras de associação podem ser utilizadas como uma abordagem de

filtragem colaborativa da seguinte forma. Dado o conjunto de avaliações R formatado numa matriz de usuários por itens, cada linha de R representa o conjunto de itens que um usuário avaliou e também pode ser interpretada como uma transação. Logo, as regras computadas por um algoritmo de regras de associação representam a relação de itens que coocorrem nos históricos dos usuários e podem ser utilizadas para gerar recomendações para um dado usuário $u \in U$ seguindo os passos abaixo:

1. Dado o conjunto de todas as regras $X \Rightarrow Y$, filtrar aquelas onde o antecedente X representa um conjunto de itens que está presente no histórico de consumo de u, enquanto Y representa um conjunto de itens que u ainda não consumiu;

2. Ordenar as regras filtradas no passo 1, utilizando alguma estatística, tais como, suporte ou confiança.

3. Recomendar os top-n itens presentes como Y nas regras filtradas.

Quadro 2.3: Pseudocódigo do algoritmo *Explanation Mining*.

Input: Dados de treinamento (Matriz R), Predições (Matriz $\hat{R}$), número de transações d, número de recomendações n
Output: recomendações e suas respectivas explicações
$T = \{\}$
foreach usuário $u \in U$:
    gerar a lista de transações T_u baseada nas top-d predições presentes em $\hat{R}$
    $T = T \cup T_u$
Gerar o conjunto de regras Z no formato $X \Rightarrow Y$ baseado nas transações de T
foreach usuário $u \in U$:
    obter regras $Z_u \subseteq Z$ tal que $X \subseteq R$ e $Y \subsetneq R$
    ordenar as regras baseado em algum critério (ex. suporte)
    recomendar os top-n itens presentes em Y com as suas explicações (regras $X \Rightarrow Y$)

No trabalho (PEAKE; WANG, 2018), o conjunto de transações T para treinar um modelo de regras de associação é gerado com as top-d predições para cada usuário, incluindo predições de itens do seu próprio histórico de consumo. Então, aplica-se um algoritmo de mineração de regras de associação em T para obter as regras $X \Rightarrow Y$. Por fim, para obter explicações para as recomendações para um dado usuário u, novamente as regras são filtradas para que o antecedente X esteja presente no histórico do usuário u, e o consequente Y

não esteja presente. Dessa forma, a regra $X \Rightarrow Y$ provê a explicação *"como você assistiu X, então recomendamos Y"*. O Quadro 2.3 detalha um pseudocódigo dessa proposta.

Perceba que o conjunto de regras de associação Z é gerado baseado nas predições de todos os usuários (linhas 1 a 5 do pseudocódigo acima). Os autores argumentam que essa ideia pode não capturar fielmente a ideia de localidade por tratar com todos usuários de uma vez e por isso a chamam de regras globais. Sendo assim, a ideia de localidade foi de fato implementada ao utilizar os usuários mais similares, baseado nas predições $\hat{R}$, ao usuário alvo u. Ou seja, diferentes conjuntos Z são gerados, um para cada usuário $u \in U$, onde apenas as transações dos usuários mais similares a u são levadas em consideração.

2.4.3 Considerações Finais

Neste capítulo, detalhamos os principais conceitos abordados nesta tese. Apresentamos os fundamentos dos modelos clássicos de recomendação e também como a interpretabilidade é promovida por meio das explicações das recomendações. Aliás, foi demonstrado que, segundo a literatura, uma explicação pode ter diferentes objetivos e que isso contribui para o fato de não haver um consenso sobre o que é uma explicação ideal.

Ao final, foram detalhados dois métodos que promovem interpretabilidade para modelos de aprendizado de máquina: o LIME e o *Explanation Mining*. Ambos os métodos são agnósticos ao modelo e seguem a abordagem baseada em modelos locais interpretáveis. A diferença é que o LIME não foi projetado para o (e aplicado ao) domínio de recomendação, fato que motivou o início desta pesquisa.

Capítulo 3

Trabalhos Relacionados

Neste capítulo, são abordados diversos trabalhos que tratam do problema de tornar modelos complexos e de difícil interpretação de Sistemas de Recomendação mais interpretáveis. A literatura apresenta diferentes formas de atacar esse problema. Podemos agrupar esses trabalhos em duas grandes categorias: Design de Interface para Explicações em Sistemas de Recomendação (Seção 3.1); e Geração de Explicações para Recomendações (Seção 3.2). Enquanto a primeira trata de design e princípios de interface para exibição de explicações, a segunda trata das duas principais formas de gerar explicações para um SR: gerar explicações após o treinamento do modelo recomendador; e a incluir, no treinamento do modelo, funções de custo que contém componentes de interpretabilidade. Analisamos esses trabalhos com o objetivo de destacar as semelhanças e diferenças em relação ao trabalho apresentado nesta tese.

Os trabalhos apresentados neste capítulo foram encontrados por meio de pesquisas nas principais conferências de Aprendizagem de Máquina e Sistemas de Recomendação (RecSys, ICML, NIPS, ECML, AAAI, SIGIR, WWW), além das bibliotecas digitais da IEEE[1] e ACM[2], e sites de busca como Google Scholar[3].

[1] https://ieeexplore.ieee.org/Xplore/home.jsp
[2] https://dl.acm.org/
[3] https://scholar.google.com/

3.1 Design de Interface para Explicações em Sistemas de Recomendação

Diversos trabalhos têm atacado o problema de gerar explicações na área de Sistemas de Recomendação. Um estudo pioneiro que destacou o uso de explicações para elucidar recomendações foi (HERLOCKER; KONSTAN; RIEDL, 2000), que propôs 21 tipos diferentes de interfaces para apresentar explicações. De fato, gerar explicações não é um requisito recente, uma vez que já tem sido considerado como questão de pesquisa em áreas correlatas, tais como, sistemas especialistas (BUCHANAN; SHORTLIFFE, 1985; HUNT; PRICE, 1988; LOPEZ-SUAREZ; KAMEL, 1994).

Assim como (HERLOCKER; KONSTAN; RIEDL, 2000), vários trabalhos seguiram a linha de pesquisa cujo objetivo é propor diferentes formas de apresentação das explicações (vide Seção 2.3.1). Por exemplo, (HERNANDO et al., 2013) utiliza um grafo para conectar os itens recomendados com os itens consumidos por um determinado usuário alvo. Logo, como o usuário se depara com explicações sendo itens relacionados, as explicações deste trabalho seguem o estilo *item* (ver Seção 2.3.1). Além disso, o seu diferencial é prover não apenas uma explicação individual, pois o grafo explica todos os itens recomendados para um dado usuário. Em outro exemplo, (VIG; SEN; RIEDL, 2009) utiliza *tags*, rótulos que ajudam a descrever uma determinada entidade, para explicar recomendações. Aqui são utilizadas *tags* dos itens e a ideia é exibir aquelas que são relevantes seguindo dois critérios: descrever bem o item e que também esteja alinhada com as preferências do usuário alvo, ou seja, baseada nas tags dos itens avaliados positivamente no seu histórico de consumo.

O trabalho (GEDIKLI; JANNACH; GE, 2014) estendeu o trabalho pioneiro de (HERLOCKER; KONSTAN; RIEDL, 2000) ao comparar, em um estudo com usuários, um subconjunto dos diferentes estilos propostos por este com sua solução de explicações baseadas em nuvem de *tags*. Além disso, diferentes designs de interfaces foram analisadas sob os diferentes objetivos que uma explicação pode ter. A propósito, não há uma forma de explicação que seja a melhor em todos os cenários. Estudos têm mostrado que diferentes formas de explicação podem ser exitosas dependendo do objetivo que se deseja atingir. Por exemplo, explicações baseadas nas preferências de usuários similares são exitosas quando o objetivo é persuasão: os usuários tendem a superestimar a qualidade das recomendações explicadas

desta maneira (BILGIC; MOONEY, 2005). Enquanto que explicações baseadas em itens e *tags* são bem-sucedidas no objetivo satisfação (Symeonidis; Nanopoulos; Manolopoulos, 2008). Esses objetivos foram descritos no trabalho de (TINTAREV; MASTHOFF, 2012) e estão detalhados na Seção 2.3.3.

Similarmente, Pu e Chen (PU; CHEN, 2007) propõe princípios de design para construção de interfaces que exibam explicações. Um dos cinco princípios propostos declara que a lista de itens recomendados deve ser apresentada junto com informações sobre o *trade-off* dos atributos desses itens. Por exemplo, supondo que o item top-1 da lista de recomendação foi um laptop, a ideia é organizar a exibição dos demais itens, destacando o *trade-off* entre cada um deles e o item top-1. Nesse caso, poderíamos ter: o item top-2 tem um *maior preço e é mais pesado*; o top-3 é *mais barato, mas o processador é mais lento*. O argumento desse princípio é que, ao exibir a lista de recomendação dessa forma com as explicações realçando o *trade-off*, pode exigir menos esforço cognitivo do usuário ao tentar entender a lista.

As explicações dos trabalhos mencionados acima geralmente são visualizações (ou textos) estáticas. Uma linha de pesquisa relacionada foca em prover transparência às recomendações por meio de visualizações interativas. Por exemplo, o TasteWeights (BOSTANDJIEV; O'DONOVAN; HöLLERER, 2012) é um ferramenta que gera recomendações de forma híbrida. Ela permite que o usuário possa ajustar o recomendador ao exibir a ligação entre as suas avaliações, as preferências aprendidas pelo recomendador e as recomendações propriamente ditas. TalkExplorer (VERBERT et al., 2013) e SetFusion (PARRA; BRUSILOVSKY; TRATTNER, 2014) são ferramentas similares. Elas também permitem visualizar relações entre recomendações e múltiplas técnicas de recomendação, porém com propostas diferentes para atingir esse objetivo. O trabalho (HE; PARRA; VERBERT, 2016) faz uma extensa comparação entre essas 3 e outras propostas dessa linha de pesquisa.

Há trabalhos nessa direção voltados para visualizações de outros modelos de aprendizado de máquina. Por exemplo, o trabalho de (JEYAKUMAR et al., 2020) foca em modelo de redes neurais profundas e mostra, basicamente, que pessoas preferem explicações diferentes para tipos diferentes de dados. Por exemplo, se o domínio do problema envolve texto, então métodos que geram explicações em termos de atributos dos dados, como o proposto nesta tese, é o mais indicado, enquanto que problemas com imagens, o indicado é exibir explicações com exemplos de outras imagens.

Diferentemente dos estudos listados nesta seção, o nosso trabalho não foca em desenvolver ou comparar novas interfaces ou estilos de apresentação de explicações. Por outro lado, esses estudos estão relacionados com o trabalho desta tese de duas maneiras: (1) confirmam que exibir explicações pode melhorar a aceitação de um SR, o que ratifica a relevância do nosso problema; e (2) informações extras (atributos de usuários ou itens) são formas bastante comuns de apresentar as explicações. Note que, a nossa proposta (vide Capítulo 4) permite explicações tanto no estilo item quanto no estilo atributo (vide Seção 2.3.1).

3.2 Geração de Explicações para Recomendações

Na Seção 2.4, foi discutido que há duas formas principais de obter interpretabilidade em modelos de aprendizado de máquina: utilizando modelos intrinsecamente interpretáveis e por meio de métodos que geram explicações de forma agnóstica ao modelo. Vários trabalhos adaptaram e aplicaram ambas as ideias para o contexto de Sistemas de Recomendação.

3.2.1 Métodos Agnósticos ao Modelo aplicados a Sistemas de Recomendação

Basicamente, métodos agnósticos ao modelo adicionam uma camada de interpretabilidade por meio da qual são extraídas as explicações. Dessa forma, as predições de um modelo, geralmente *black-box*, e as explicações para essas predições são geradas em processos desassociados.

O trabalho de (MUSTO et al., 2016) segue essa linha de pesquisa e propõe a utilização de linguagem natural para gerar explicações personalizadas. A ideia é fugir da forma resumida de apresentar explicações apenas como uma lista de atributos. Nesse sentido, os autores propõem gerar explicações em linguagem natural ao preencher um *template* que menciona tanto itens do histórico do usuário quanto os seus atributos. Por exemplo, *"Recomendo a você <Filme> uma vez que você geralmente gosta de filmes estrelados por <Ator> e que contenham <Gênero>"* (vide Figura 2.8 para exemplo mais detalhado). O propósito é enriquecer a explicação, dando mais detalhes ao conectar o item recomendado com os itens do

histórico por meio de atributos extraídos da *Linked Open Data (LOD) Cloud*[4] (base de dados que relaciona diversas entidades e lista suas propriedades).

O trabalho de (CHANG; HARPER; TERVEEN, 2016) também gera explicações usando linguagem natural, porém há dois aspectos que o diferem do anterior: a fonte de dados das explicações são *tags* fornecidas pelos usuários ou extraídas de *reviews* textuais dos itens; e utiliza o fator humano (em inglês, *crowd-sourcing*) ao contratar pessoas para refinar *tags* e sintetizar *reviews*. Aqui o diferencial é justamente o fator humano na tarefa de gerar explicações mais elaboradas e convincentes. Mesmo que serviços de *crowd-sourcing* estejam ficando cada vez mais acessíveis, essa característica pode ser entendida como desvantagem, pois tende a aumentar o custo do processo de geração de explicações. Apesar de ter sido implantado e avaliado apenas em um serviço de filmes, a ideia pode ser aplicada em diferentes domínios/algoritmos.

Os métodos dos dois últimos trabalhos citados seguem a abordagem *post-hoc* (ou *model-agnostic*) *interpretability*, pois são capazes de gerar explicações de maneira independente do modelo recomendador, e por isso estão relacionados ao trabalho desta tese. Contudo, essas abordagens diferem da nossa no sentido de que elas não se concentram em aprender modelos locais para explicar porque as recomendações individuais foram feitas. Logo, a forma de gerar explicações proposta nesta tese tem a característica de imitar o modelo de recomendação subjacente, o que pode tornar a geração de explicações mais fiéis quando comparada com essas outras propostas. Outra diferença que podemos destacar é o estilo das explicações. Enquanto eles seguem o estilo *híbrido* e usam linguagem natural, a nossa proposta permite os estilos *item* e *atributo*. Nesse caso, para determinadas aplicações, pode ser uma desvantagem da nossa proposta.

De fato, aplicar modelos locais é uma forma proeminente de métodos de interpretabilidade agnóstico ao modelo. O trabalho de Peake & Wang (PEAKE; WANG, 2018) pode ser considerado o pioneiro, com o melhor do nosso conhecimento, em aplicar tal abordagem aos SR. Nesse trabalho, os autores utilizam uma abordagem baseada em regras de associação como modelo intrinsecamente interpretável para aproximar o comportamento de um modelo de recomendação baseado em Fatoração de Matrizes. Basicamente, a ideia é construir um conjunto com itens relevantes de cada usuário, baseado nas predições do recomendador, para

[4] http://linkeddata.org/

treinar um modelo de regras de associação. Em outras palavras, a lista de itens relevantes de cada usuário representa uma transação e o conjunto de todas as transações forma a entrada para o modelo de regras de associação. Dessa forma, se um dado item Y é recomendado e há uma regra de associação, tal que $X \Rightarrow Y$, a seguinte explicação do estilo item é gerada: "como você assistiu X, então recomendamos Y". Esse é o trabalho relacionado mais próximo da pesquisa conduzida nesta tese (vide Capítulo 4). Aqui, nos referimos a esse método como EM-RS, que significa *Explanation Mining for Recommender Systems*, inspirado no título do próprio artigo.

Na verdade, a abordagem baseada em modelos locais interpretáveis é mais adotada no contexto de aprendizado de máquina em geral. Nessa direção, um dos trabalhos mais populares é de (RIBEIRO; SINGH; GUESTRIN, 2016b). Nesse trabalho, foi proposto o método *Local Interpretable Model-Agnostic Explanations* (LIME), o qual consiste em construir modelos lineares locais ao redor de uma instância que se deseja entender a classificação. Esse método, detalhado na Seção 2.4.2, é relevante, pois foi inspiração para esta pesquisa e diversos outros trabalhos. Por exemplo, (MISHRA; STURM; DIXON, 2017) propõe 3 formas diferentes de aplicar o LIME para o contexto de análise de conteúdo musical. Já o trabalho (SINGH; ANAND, 2019) expande a ideia do LIME para lidar com modelos de ranking no contexto de recuperação da informação, visando responder questões como *"Por que documento x está acima do documento y no ranking?"* ou *"Por que documento x é relevante para uma consulta q?"*. Ainda, o trabalho (RIBEIRO; SINGH; GUESTRIN, 2016a), dos mesmos autores do LIME, também propõe uma extensão chamada aLIME, que explica uma dada predição com regras *if-then*. Esses trabalhos estão relacionados com a pesquisa desta tese, pois todos tratam do método LIME, no entanto, até onde sabemos, a sua aplicação em Sistemas de Recomendação ainda não foi investigada.

Há trabalhos na literatura que identificaram e investigaram deficiências tanto do LIME como em outras abordagens baseadas em modelos locais. O trabalho de (ALVAREZ-MELIS; JAAKKOLA, 2018) define que explicações devem ser robustas, ou seja, instâncias similares devem ter explicações similares. Ele aponta que as explicações oriundas do LIME podem ser robustas/estáveis quando o modelo subjacente é linear, porém para modelos não lineares isso nem sempre acontece. (TAN et al., 2019) também aponta essas mesmas deficiências. Vale ressaltar que ambos os trabalhos experimentam apenas para problemas/modelos de classifi-

cação e, não, sistemas de recomendação.

Cientes das deficiências do LIME, pesquisadores já desenvolveram diferentes propostas para mitigá-las. O trabalho de (LAUGEL et al., 2018) observa que a qualidade das explicações para uma dada instância alvo está fortemente dependente da etapa de formação da vizinhança local dessa instância. Dessa forma, um método baseado em vizinhos mais próximos é proposto, tal que instâncias sintéticas são geradas dentro de uma região esférica cujo centro é a instância alvo. A intuição por trás dessa proposta é controlar a definição de localidade por meio do raio da região esférica. O trabalho de (JIA et al., 2019) também ataca esse problema de aprimoramento da geração de vizinhança. Os autores propõem a utilização de técnicas de redução de dimensionalidade, tais como *Principal Component Analysis (PCA)*, para obter dados sintéticos mais fiéis à instância alvo. A intuição é que, ao mapear os dados originais em um espaço de menor dimensão, os dados sintéticos que vão compor a vizinhança preservem as características principais da localidade da instância alvo. O Capítulo 5 investiga esse problema e explora essa linha de pesquisa ainda não investigada no domínio de SR.

Como mencionado anteriormente, o uso de modelos locais interpretáveis é apenas uma das abordagens para métodos de interpretabilidade *post-hoc*. Uma outra abordagem envolve técnicas de relevância de atributos. Essa família de técnicas provê interpretabilidade para o funcionamento de modelo *black-box* por meio da medição e ordenação da influência, relevância ou importância que cada atributo tem sobre uma dada predição. *SHapley Additive exPlanations* (SHAP) (LUNDBERG; LEE, 2017) é uma proposta que implementa essa abordagem com base no conceito da teoria dos jogos chamado de valores Shapley. Do ponto de vista da teoria do jogos, as variáveis preditoras de um dado modelo são os "jogadores" e eles competem no jogo de influenciar a saída do modelo. Dessa forma, quanto maior a relevância da variável em uma dada predição, maior é seu valor Shapley. Em um estudo com usuários, os autores mostram que as explicações providas pelo SHAP estão mais de acordo com as explicações oriundas da intuição dos usuários quando comparadas com o LIME e outros métodos relacionados. Por outro lado, uma desvantagem desse método é o seu custo, pois, para obter os valores Shapley para um modelo com F atributos, é preciso treinar $2^{|}F|$ modelos, um para cada combinação de atributos. Isso é impraticável em bases de dados do mundo real e, por essa razão, esse método está fora do escopo deste trabalho.

Vale ressaltar que tanto o SHAP quanto o LIME podem ser aplicados para o problema de classificação de imagens. (BUHRMESTER; MÜNCH; ARENS, 2019) faz um revisão de literatura de métodos voltados para esse linha de pesquisa.

Normalmente, a avaliação de métodos que geram explicações envolve experimentos com humanos, sejam experimentos online de testes A/B ou estudos com usuários. Contudo, muitos trabalhos realizam experimentos offline. Para esses casos, é preciso gerar um conjunto *ground truth* de explicações, com o qual as explicações providas pelos métodos são comparadas. A forma mais adotada de realizar isso é utilizar um modelo intrinsecamente interpretável. Por exemplo, em um dos experimentos do artigo original do LIME, é utilizado um modelo de árvores de decisão (vide Seção 2.4.1) para gerar o conjunto *ground truth* de explicações, que seriam as explicações ideais. A avaliação consiste, normalmente, em computar a fração das explicações ideais que o método consegue recuperar. Vale salientar que essa é a metodologia de avaliação adotada nos experimentos desta tese. Recentemente, (GUIDOTTI, 2021) argumenta que, geralmente, os trabalhos que seguem essa metodologia escolhem o modelo do *ground truth* de forma *ad-hoc*. Logo, é proposto uma forma de gerar classificadores interpretáveis de forma sintética, que pode ser uma forma de padronizar experimentos offline para avaliações de métodos baseados em modelos locais interpretáveis.

Como discutido acima, há diversos métodos para gerar explicações tanto no contexto de SR quanto no de ML em geral. O trabalho de (NUNES; JANNACH, 2017) faz uma ampla revisão sistemática – revisão de literatura caracterizada pela exatidão e transparência do processo de procura, avaliação e síntese (KITCHENHAM; CHARTERS, 2007) – envolvendo 217 artigos científicos, principalmente, da área de sistemas de recomendação que, de alguma forma, estão relacionados com explicações. Esse trabalho ressalta a importância das explicações como forma de viabilizar a transparência e, ainda, propõe uma nova taxonomia que trata as diferentes facetas como, por exemplo, o objetivo da explicação, conteúdo e apresentação. Mais recentemente, (VILONE; LONGO, 2020) também fez uma revisão sistemática, porém, com escopo maior, abrangendo pesquisas de diferentes áreas sobre *Explainable AI*. Por fim, vale destacar os trabalhos (Gilpin et al., 2018; Barredo Arrieta et al., 2020) que são surveys sobre interpretabilidade voltados para ML em geral, e o trabalho de (ZHANG; CHEN, 2018) que é um survey voltado para SR.

3.2.2 Recomendações Explicáveis

Outra linha de pesquisa ataca o problema da dificuldade de interpretação de modelos black-box de recomendação com um viés mais intrínseco, ou seja, está diretamente conectada ao modelo de recomendação em si. O principal objetivo dos trabalhos que seguem essa linha é desenvolver métodos cujos itens recomendados sejam tanto acurados quanto fáceis de explicar. Em outras palavras, essa linha de pesquisa propõe desenvolver funções de custo que incluem componentes de interpretabilidade e incluí-las no treinamento do modelo de recomendação. Comumente, a literatura se refere a essa linha de pesquisa como recomendações explicáveis.

Nesse sentido, vários trabalhos foram propostos. (ZHANG et al., 2014) assume que recomendações explicáveis são itens que estão alinhados com os atributos favoritos do usuário. A ideia principal é extrair atributos e sentimento/opinião sobre esses atributos, chamados de fatores explícitos, a partir das *reviews* dos usuários e inseri-los numa nova função de custo para modelos de Fatoração de Matrizes. Dessa forma, os fatores latentes podem ser alinhados com os fatores explícitos e as recomendações podem ser apresentadas com explicações, por exemplo *"Este produto é recomendado porque você está interessado em um atributo específico e este produto é bem avaliado quanto a esse atributo"*. Os mesmos autores estenderam esse trabalho para lidar com a questão de preferências dinâmicas, que é o fato de as preferências dos usuários poderem mudar com o passar do tempo (ZHANG et al., 2015). Ainda, o trabalho de (CHEN et al., 2016) é uma outra extensão, porém, a mesma ideia é adaptada e aplicada para fatoração de tensores.

De forma semelhante, o trabalho de (HE et al., 2015) também utiliza reviews como fonte de informações extras, porém atua em um framework de grafos ao invés da Fatoração de Matrizes. Nesse caso, a relação usuário-item-atributo é modelada como um grafo tripartido e uma função de custo é proposta para ranquear os vértices do grafo de forma que tanto o aspecto de filtragem colaborativa FC quanto de abordagens baseada em conteúdo sejam alcançados. (Heckel et al., 2017) também utiliza a modelagem com grafos, porém, grafos bipartidos, pois informações extras não são utilizadas. Nesse caso, os autores aplicam técnicas de identificação de *co-clusters*, que são grupos de usuários e itens que compartilham padrões similares, e as explicações podem ser geradas no estilo híbrido, por exemplo *"Item x foi recomendado para usuário u com confiança c, pois usuário u já consumiu os itens a,*

b e usuários similares a u também consumiram x". Ainda, o trabalho de (RANA; BRIDGE, 2017) também utiliza grafos para geração de recomendações explicáveis. Especificamente, a recomendação é modelada como um problema de localização de caminho em um grafo de similaridade item-item, onde cada item é um nó do grafo. Uma vez que uma cadeia (sequência de nós) foi construída para cada item candidato, as cadeias top-n são selecionadas iterativamente com base em sua cobertura total dos atributos do item candidato e sua dissimilaridade com outras cadeias no top-n. Os atributos dos itens presentes na cadeia formam as explicações.

De fato, a literatura apresenta uma série de trabalhos que focam em modelos baseados em Fatoração de Matrizes. Alguns autores tentam dar um significado aos fatores latentes do modelo (vide Seção 2.2.1). (GANTNER et al., 2010) propõe um método para mapear atributos de usuários ou itens aos fatores latentes. Embora o seu propósito original seja atacar o problema de *cold-start*, a estratégia também serve para promover interpretabilidade aos fatores. Outro trabalho nesse sentido foi proposto por McAuley & Leskovec (MCAULEY; LESKOVEC, 2013), no qual os fatores latentes são mapeados para tópicos extraídos dos reviews dos usuários. Ainda de maneira similar, (BRUN; ALEKSANDROVA; BOYER, 2014; ALEKSANDROVA et al., 2017) propõe mapear cada fator latente em um usuário, na tentativa de tornar o entendimento da Fatoração de Matrizes similar aos métodos baseados na vizinhança. De forma semelhante, há trabalhos que focam em modelos baseados em *Deep Learning* (DL), por exemplo, tornando as camadas da rede visualmente inteligíveis (Khalifa Bashier Babiker; Goebel, 2017). Ao contrário dos trabalhos acima mencionados, o foco do nosso trabalho é elucidar uma previsão/recomendação por meio da geração de explicações e não dar significado às estruturas ou parâmetros do modelo.

Uma linha de pesquisa relacionada envolve integrar modelos baseados em fatores latentes com outros modelos intrinsecamente interpretáveis, tais como árvores de decisão (vide Seção 2.4.1). A ideia é combinar o poder preditivo dos modelos baseados em fatores latentes com o poder de interpretabilidade de outros modelos. Por exemplo, (WANG et al., 2018) empregaram um modelo baseado em árvores para aprender regras de decisão explícitas (caminhos pelos nós da árvore), que são baseadas em atributos extraídos de informações extras. Em seguida, os autores incorporam tais regras no treinamento do modelo de fatoração, resultando em um processo de recomendação transparente e fácil de interpretar. O trabalho

de (TAO et al., 2019) utiliza árvores de decisão para guiar o treinamento de um modelo de fatoração de matrizes. Especificamente, treinam-se árvores de decisão para usuários e itens, cujos atributos são extraídos de *reviews*, de forma que cada nó esteja associada a um fator latente. Por exemplo, se dois usuários compartilham *reviews* positivas para cerveja, então eles devem estar associados ao mesmo nó numa árvore de decisão, pois compartilham os mesmo fatores. Dessa forma, explicar uma recomendação se resume a percorrer os nós das árvores.

Outras estratégias geram recomendações explicáveis baseadas somente na matriz usuário-item, sem utilizar informações extras. Abdollahi & Nasraoui (ABDOLLAHI; NASRAOUI, 2016a) definem uma medida de explicabilidade baseada na distribuição das avaliações dos usuários e a inserem no aprendizado de modelos baseados em Fatoração de Matrizes. A intuição da métrica é fazer com que o vetor de fatores latentes do usuário seja próximo do vetor de fatores latentes do item, se muitos usuários similares também consumiram o mesmo item. Dessa forma, explicações como *"muitos usuários similares a você também consumiram este item"* são geradas. O trabalho de (LIU et al., 2019) estende essa ideia e aplica em um modelo de fatoração probabilístico. Os autores empregam um mecanismo para avaliar a influência dos dados dos históricos dos usuários, de forma que, para explicar uma predição, usuários e itens mais relacionados a esta são utilizados.

Por fim, como mencionado na Seção 2.2.1, muitos modelos de recomendação são baseados em técnicas de *Deep Learning* (DL).Para cada técnica de DL, há diferentes modelos de recomendação e, consequentemente, diferentes propostas para gerar recomendações explicáveis. O trabalho de (SEO et al., 2017) propõe modelar as preferências dos usuários com uma rede neural convolucional sobre dados de *reviews*. Mecanismos de atenção são utilizados para extrair representações dos usuários e dos itens. O modelo gera uma recomendação de uma *review*, destacando as palavras importantes de seu texto como explicações para ajudar a entender a recomendação. Inspirado no processamento de informações por humanos da psicologia cognitiva, (CHEN et al., 2019) propõe uma arquitetura *enconder-selector-decoder* para gerar recomendações explicáveis. Já o trabalho de (CHEN et al., 2019) propõe uma abordagem para recomendações explicáveis visuais e textuais para o domínio de moda. Baseado em diferentes técnicas de DL, o modelo destaca regiões da imagem do item de moda, descrevendo-as com um texto em linguagem natural.

Todos os trabalhos desta subseção estão relacionado com a pesquisa desta tese, pois abordagem o tema de interpretabilidade em modelos de recomendação e de aprendizado de máquina. Contudo, o foco é diferente, uma vez que eles visam desenvolver métodos que são interpretáveis por definição.

3.3 Posicionamento em Relação aos Trabalhos Relacionados

Diversos trabalhos lidam com o problema de interpretabilidade em modelos de Sistemas de Recomendação e Aprendizado de Máquina em geral. Como mencionado anteriormente, a literatura apresenta diferentes abordagens de solução.

A abordagem de gerar recomendações explicáveis está principalmente voltada para modelos de fatores latentes e baseados em *Deep Learning*. Os trabalhos implementam esssa abordagem, propondo novas arquiteturas/modelos ou combinação de modelos com o objetivo de obter soluções intrinsecamente interpretáveis. As soluções são diversas e aplicadas em diferentes tarefas de recomendação.

A abordagem de métodos de interpretabilidade *post-hoc* é bastante popular devido ao fator flexibilidade que ela provê. Os trabalhos relacionados que seguem essa abordagem estão mais próximos desta tese. Especificamente, destaca-se o trabalho de (PEAKE; WANG, 2018) por ser um dos pioneiros a aplicar a abordagem *post-hoc*, modelos locais interpretáveis, no contexto de SR. Ele faz parte da nossa avaliação, pois nossa proposta também é baseado na mesma abordagem. Vale ressaltar que nossa proposta é uma adaptação do método LIME (RIBEIRO; SINGH; GUESTRIN, 2016b) ainda não aplicado, com o melhor do nosso conhecimento, para o contexto de SR.

A literatura lista trabalhos (ALVAREZ-MELIS; JAAKKOLA, 2018; TAN et al., 2019) que evidenciaram deficiências do LIME e também trabalhos (LAUGEL et al., 2018; JIA et al., 2019) que propuseram soluções para essas deficiências. Inspirados nessas soluções, também investigamos como adaptá-las para a nossa proposta.

A Tabela 3.1 sumariza o posicionamento deste trabalho em relação aos principais trabalhos relacionados do estado da arte.

	Estilos			Abordagem		Aplicação			Relação - LIME	
Trabalhos	Humano	Item	Atributo	Recomendações Explicáveis	Métodos de interpretab. *post-hoc*	Sistemas de Recomendação	Problemas de Classificação	Outro	Adaptação	Avaliação
(RIBEIRO; SINGH; GUESTRIN, 2016b)					X		X			
(MUSTO et al., 2016)		X	X		X	X				
(CHANG; HARPER; TERVEEN, 2016)			X		X	X				
(RIBEIRO; SINGH; GUESTRIN, 2016a)					X		X			X
(MISHRA; STURM; DIXON, 2017)					X			X	X	
(SINGH; ANAND, 2019)					X			X	X	
(ALVAREZ-MELIS; JAAKKOLA, 2018)					X		X			X
(TAN et al., 2019)					X		X			X
(LAUGEL et al., 2018)					X		X		X	X
(JIA et al., 2019)					X		X		X	X
(LUNDBERG; LEE, 2017)					X		X			X
(GUIDOTTI, 2021)					X		X			X
(PEAKE; WANG, 2018)		X			X	X				
(ZHANG et al., 2014)			X	X		X				
(ZHANG et al., 2015)			X	X		X				
(CHEN et al., 2016)			X	X		X				
(HE et al., 2015)			X	X		X				
(Heckel et al., 2017)	X	X		X		X				
(WANG et al., 2018)			X	X		X				
(TAO et al., 2019)			X	X		X				
(ABDOLLAHI; NASRAOUI, 2016a)		X		X		X				
(LIU et al., 2019)		X		X		X				
(SEO et al., 2017)			X	X		X				
(CHEN et al., 2019)			X	X		X				
Este trabalho		X	X		X	X			X	

Tabela 3.1: Tabela Comparativa de Trabalhos Relacionados.

Capítulo 4

Explicações para uma Recomendação via aprendizado de Modelos Locais Interpretáveis

Este capítulo formaliza o problema da extração de explicações para uma dada recomendação de forma agnóstica ao recomendador. Em seguida, detalha uma proposta de solução para esse problema. O capítulo ainda apresenta uma comparação da nossa proposta com uma alternativa do estado da arte. Por fim, são detalhados alguns exemplos de explicações providas por nossa proposta.

4.1 Formalização do Problema

Como mencionado no Capítulo 1, o problema técnico da nossa pesquisa consiste em extrair explicações de uma dada recomendação proveniente de qualquer modelo de Sistema de Recomendação. Consideramos como explicações as características/atributos do item que são fáceis de entender por humanos. Por exemplo, em um cenário de recomendação de filmes, na sentença "estamos recomendando este filme para você porque ele tem *ação* e *comédia*", os gêneros do item compõem a explicação. Logo, o problema se resume a encontrar os atributos do item que melhor explicam uma determinada recomendação de acordo com as preferências do usuário.

Ademais, de acordo com as definições do Capítulo 2, a explicação que estamos interes-

sados segue o estilo *Atributo* (vide Seção 2.3.1) e o tipo *black-box*, pois dão justificativas para as saídas do recomendador sem entrar em detalhes do seu funcionamento (vide Seção 2.3.2).

O problema pode ser formalmente definido da seguinte maneira. Sejam um conjunto de usuários U, um conjunto de itens I, um conjunto de atributos dos itens F e $\hat{r}(u, i)$ a predição de uma avaliação realizada por um modelo de recomendação $r : U \times I \rightarrow \mathbb{R}$, tal que $u \in U$ and $i \in I$. Se r leva em consideração informações extras acerca de usuários ou itens, é conhecido como *context-aware* ou híbrido. Queremos encontrar um subconjunto com os top-n atributos $E \subseteq A$ que melhor expliquem uma dada recomendação, ou seja:

$$E = \arg\max_{a \in A}^{n} s(a, \hat{r}(u, i)) \tag{4.1}$$

onde $s(\cdot, \cdot)$ é uma função que mede quão bom é o atributo $a \in A$ para explicar a predição $\hat{r}(u, i)$ e o n acima do $\arg\max$ representa os top-n atributos que maximizam essa função.

Note que a definição da função $s(\cdot, \cdot)$ não é uma tarefa trivial, uma vez que, idealmente, depende do julgamento humano. Na seção a seguir, apresentamos uma alternativa para definição da função $s(\cdot, \cdot)$ que não depende necessariamente do julgamento humano, mas de um modelo intrinsecamente interpretável que seja uma aproximação, pelo menos local, do modelo original.

Vale ressaltar que, embora o problema esteja formalizado em termos de predição de avaliações, o mesmo raciocínio pode ser considerado para o problema de recomendação de itens. Neste caso, um *score* $\in \mathbb{R}$ é computado e utilizado para ordenar a lista de recomendações. Portanto, de forma semelhante a $\hat{r}(u, i)$, ele também pode ser entrada para a função $s(\cdot, \cdot)$.

4.2 Adaptação do LIME para Sistemas de Recomendação

O LIME é um método agnóstico ao modelo (vide Seção 2.4.2), cujo objetivo é explicar as predições de qualquer classificador ou regressor. A ideia é aprender um modelo interpretável que imite o comportamento do modelo original nas proximidades de uma dada instância alvo que está sendo predita. Apesar das deficiências eventuais apontadas na literatura e discutidas no Capítulo 3, o LIME causou impacto na área de interpretabilidade de modelos de ML, principalmente modelos de classificação.

No entanto, os dados e métodos dos SR podem ser muito diferentes das configurações de métodos supervisionados tradicionais de ML, para as quais o LIME foi originalmente projetado. Enquanto a maioria das abordagens de ML tomam como entrada vetores de características, muitos modelos de recomendação usam a tripla: (usuário, item, avaliação). Logo, além de fornecer uma representação adequada para o LIME, também é preciso prover atributos que possam ser usados para formar uma explicação. Uma ideia é considerar as informações extras acerca do usuário e/ou do item como atributos explicáveis e então utilizá-las para aprender o modelo local.

Nas subseções a seguir, detalhamos as adaptações realizadas de modo a ser possível aplicar o método LIME em modelos de recomendação black-box.

4.2.1 Representação Interpretável dos Dados

Como dito anteriormente, na maioria das configurações de modelos supervisionados de aprendizado de máquina, as instâncias de entrada são representadas como vetores de características ou dados tabulares. Já em sistemas de recomendação simples (ou seja, sem informações extras ou secundárias) as instâncias de treinamento geralmente assumem a forma de pares/tuplas (usuário, item), no caso de *feedback* implícito, ou triplas no formato (usuário, item, avaliação), em caso de *feedback* explícito. Como explicado na Seção 4.1, nesta tese, consideramos apenas dados de feedback explícito.

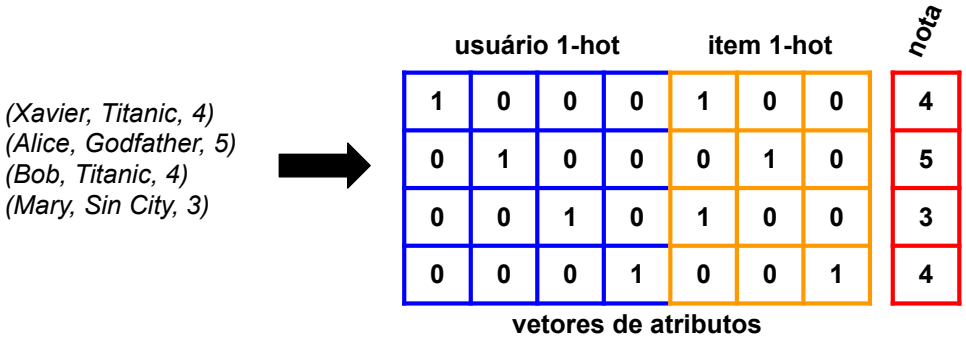


Figura 4.1: Representação tabular dos dados de avaliações em um sistema de recomendação de filmes. Por exemplo, o usuário *'Xavier'* deu nota 4 ao filme *'Titanic'*.

Portanto, o primeiro passo da nossa proposta de adaptação consiste em transformar instâncias do formato (usuário, item, avaliação) em vetores de atributos. Usuários e itens podem

ser vistos como atributos categóricos de muitos níveis e, portanto, podemos representá-los trivialmente como vetores de atributos usando o *multi-hot encoding*. O *one-hot encoding* é uma estratégia para transformar variáveis categóricas em uma representação de vetores numéricos binários com o objetivo de distinguir cada valor que a variável categórica pode assumir. Dessa forma, o vetor consiste em vários valores 0 e um único valor 1. No exemplo da Figura 4.1, temos 4 usuários. Logo, cada usuário foi representando em um vetor binário, onde o valor 1 representa o seu ID. *Multi-hot encoding* é uma variação do *one-hot encoding*, onde há mais de um valor 1. Isso ocorre no nosso caso, pois a representação individual de uma instância contém os ids do usuário e do item.

Após o mapeamento de representações, os dados do recomendador são agora representados por uma matriz, contendo um par (usuário, item) por linha e um vetor coluna que representa as avaliações correspondentes.

Observe que, ao aplicar tal mapeamento de representações, os únicos atributos disponíveis para gerar explicações são os IDs dos usuários e dos itens. Aplicar o LIME a essa representação provavelmente levará a explicações no estilo FC baseada no usuário (estilo humano) ou FC baseada no item, por exemplo, *"Estou recomendando este item para você porque você gostou de itens semelhantes no passado"* (vide Seção 2.3.1).

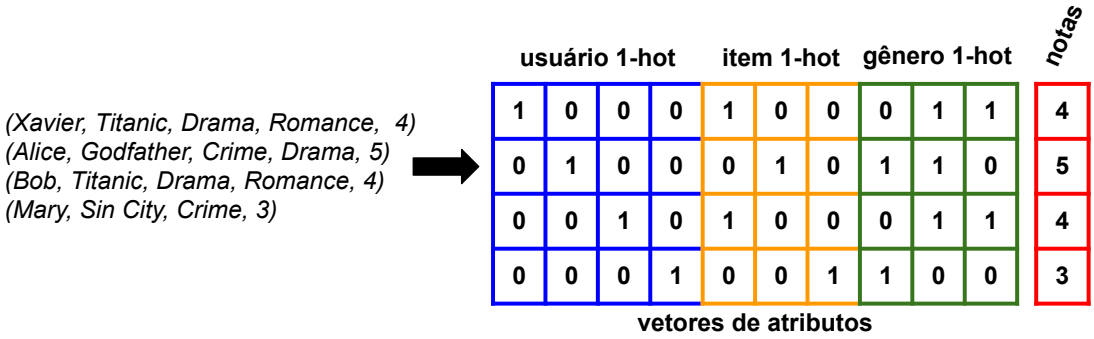


Figura 4.2: Representação tabular de dados de avaliações expandidos com informações secundárias em um sistema de recomendação de filmes. Neste exemplo, as informações secundárias são os gêneros de filmes.

Para explorar outros atributos possivelmente mais informativos, também podemos considerar as informações secundárias de itens/usuários, a partir das quais atributos explicáveis podem ser extraídos. Neste caso, faz mais sentido considerar algoritmos de recomendação que utilizam tais informações secundárias, chamados de *context-aware* ou híbridos (vide Se-

ção 2.2.1). A justificativa é que, dado que o algoritmo considera, de alguma forma, atributos explicáveis em sua lógica interna, as explicações tendem a refletir com mais fidelidade essa lógica, afinal vez também utilizam os mesmos atributos.

A Figura 4.2 ilustra os dados de avaliações expandidos com informações secundárias dos itens. Como todos os atributos neste exemplo são categóricos, consideramos representações usando o *multi-hot encoding*, embora outros tipos de atributos possam ser facilmente incorporados, tais como valores numéricos reais e datas. Essa etapa é adaptação do passo 1 detalhado na Seção 2.4.2.

4.2.2 Geração dos Dados de Treinamento

Para obter os dados de treinamento para aplicação do LIME, é preciso gerar um conjunto de dados sintéticos, chamadas perturbações, tomando a instância alvo como entrada desse processo. No algoritmo original do LIME, essa etapa significa amostrar valores booleanos para cada atributo de acordo com a distribuição empírica dos atributos no conjunto de dados originais. Por exemplo, se drama é o gênero mais popular, então a atributo *one-hot* correspondente terá uma maior probabilidade de ser amostrada. Contudo, considerando a representação tal como ilustrada na Figura 4.2, podemos obter amostras espúrias se aplicarmos diretamente a ideia original.

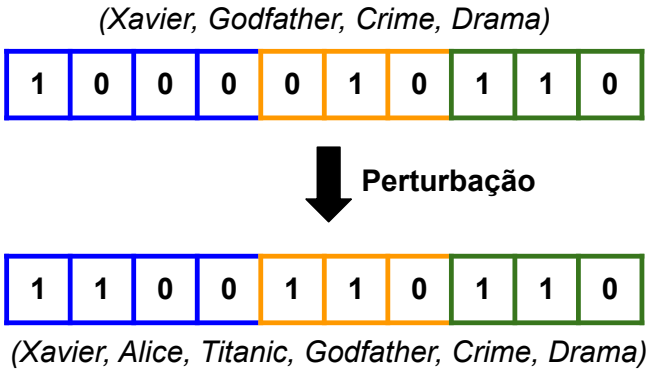


Figura 4.3: Exemplo do processo de geração de uma perturbação resultando em uma amostra espúria contendo dois usuários e dois itens para o mesmo vetor.

Observe que apenas um usuário e um item devem estar presentes por linha, ou seja, haverá exatamente um elemento diferente de zero para os componentes do usuário e do item, respectivamente. Se executarmos as perturbações inadvertidamente, teremos como resultado

amostras com mais de um elemento diferente de zero por usuário e item, o que não faz sentido, pois cada linha representa uma interação entre um único usuário e um único item. A Figura 4.3 ilustra esse problema. Nesse exemplo, o vetor que representa a perturbação indica que há dois usuários, *'Xavier'* e *'Alice'* e dois itens, *'Titanic'* e *'Godfather'*. Perceba que, mesmo sendo uma representação *multi-hot*, a instância gerada não é possível em dados de SR.

Além disso, outro problema é que algumas perturbações podem gerar amostras que nunca ocorreriam no conjunto de dados. Por exemplo, suponha que a instância que se deseja explicar seja *(Mary, Titanic, Drama)* e alguma perturbação possa ter gerado a instância *(Mary, Titanic, Animation)*. É muito improvável que isso seja observado no conjunto de dados originais, já que *Titanic* não é uma animação.

Dada a instância alvo (u, i), tal que usuário $u \in U$ e um item $i \in I$, sendo predita pela função $r(u, i)$, propomos uma abordagem alternativa que usa dados reais em vez de dados sintéticos para treinar o modelo interpretável. A ideia é fixar o usuário u da instância alvo e amostrar itens de acordo com sua distribuição empírica, ou seja, itens mais populares têm maior chance de serem selecionados. Sendo assim, formalmente, temos que a vizinha Z pode ser definida como $Z = \{(u, z) | z \in I, z \neq i\}$.

A intuição por trás dessa ideia de composição de vizinhança é que o comportamento do recomendador em outros itens considerando o mesmo usuário pode ajudar a explicar seu comportamento na instância alvo. Essa intuição é suportada pelo algoritmo de filtragem colaborativa baseada no item. Por exemplo, suponha que a instância alvo seja $(u,$ *'Toy Story'*$)$ e a predição do recomendador diz que o usuário u provavelmente gostará desse filme. Ainda, suponha que o filme *'Star Wars'* seja similar ao *'Toy Story'* em termos de avaliações dos usuários, ou seja, quando um usuário gosta de um, tende a gostar do outro. Logo, utilizar a predição do recomendador para a instância $(u,$ *'Star Wars'*$)$ pode ajudar a entender a predição para a instância alvo $(u,$ *'Toy Story'*$)$. Ademais, fixar o usuário já captura alguma noção de localidade em torno da instância alvo, uma vez que todas as perturbações geradas compartilharão o mesmo usuário.

Note que também poderíamos tentar fixar itens e amostrar usuários, mas, neste caso, a diferença de uma perturbação para outra seria de apenas 1 bit (ou seja, o usuário), gerando assim um conjunto de dados onde todas as amostras teriam a mesma distância para o ins-

tância sendo predita. Esse caso não é desejável, uma vez essa questão das distâncias entre as instâncias $z \in Z$ e a instância alvo também faz parte da noção de localidade do LIME, pois instâncias mais próximas devem ter maior importância no treinamento do modelo local interpretável (vide passo 4 do LIME original, detalhado na Seção 2.4.2). O efeito prático é que as explicações geradas tendem a perder fidelidade com a instância alvo. Vale lembrar que essa etapa de seleção de vizinhança está relacionada ao passo 2 do LIME original.

4.2.3 Geração de Explicações

Após a geração de perturbações, utilizamos r para obter o rótulo de cada uma delas e um modelo de regressão linear será treinado para aprender os pesos das atributos. Para extrair explicações, o procedimento é exatamente o mesmo que a abordagem LIME original.

A Figura 4.4 ilustra de maneira geral todo o procedimento da adaptação proposta para o LIME, que está sumarizado nas etapas abaixo.

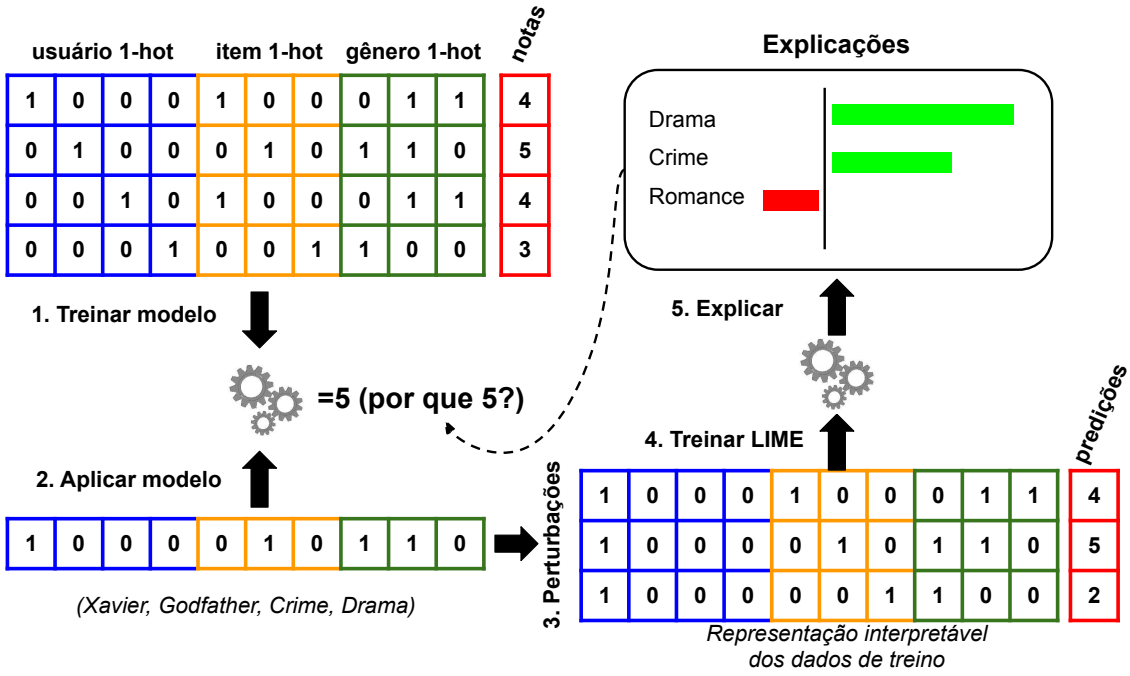


Figura 4.4: Exemplo do workflow da adaptação do método LIME para sistemas de recomendação.

1. A primeira etapa é treinar um recomendador nos dados históricos dos usuários. Neste exemplo, temos dados de avaliações de filmes com informações secundárias dos itens

(ou seja, gênero), portanto, um algoritmo *context-aware* pode ser utilizado. Vale salientar que o recomendador não precisa ser necessariamente treinado na representação de dados tabulares descritos neste capítulo.

2. Assim que o recomendador for treinado, podemos fazer predições. Neste exemplo, a instância alvo é *(Xavier, Godfather, Drama, Action)* e a avaliação predita é 5. Neste ponto, o usuário pode não saber por que essa predição foi computada, especialmente se o algoritmo subjacente for complexo.

3. A próxima etapa é gerar amostras em torno dessa instância alvo. Neste ponto, a representação tabular é necessária. Geramos amostras fixando o usuário e amostrando itens para o formar o conjunto de perturbações Z. Em seguida, predições são realizadas usando o modelo de recomendação subjacente, treinado no passo 1, para definir o rótulo de cada perturbação. Neste caso, o rótulo é uma nota, mas também poderia ser uma classe, por exemplo, o usuário vai gostar ou não do item. Observe que, neste exemplo, não há transformação de atributos, ou seja, os atributos ditos como interpretáveis são iguais aos originais, logo, a função h é a função de identidade.

4. A próxima etapa é treinar uma regressão Lasso (ou Ridge) no conjunto de dados Z. Com a regressão lasso, podemos controlar a complexidade do modelo de regressão linear deixando apenas os n atributos mais importantes. Além disso, no treinamento do modelo de regressão, é levada em consideração uma função de proximidade $\pi_x(z)$ (distância euclidiana ou distância do cosseno) para ponderar as instâncias $z \in Z$, de forma que instâncias mais similares à instância alvo tenham um maior peso. Note que essa regressão corresponde à uma instanciação de um modelo $g \in G$ do conjunto de modelos intrinsecamente interpretáveis G (vide Equação 2.5).

5. Por fim, os top-n atributos juntamente com seus respectivos pesos são apresentados para o usuário. Neste exemplo, teríamos que *Drama* e *Crime* contribuíram positivamente para a recomendação enquanto *Romance* contribuiu negativamente. Eventualmente, a explicação pode apresentar gêneros que não estão associados ao item da instância alvo que está sendo predit. Isso ocorre pelo fato de que o conjunto Z, isto é, a vizinhança da instância alvo, é formado por itens diversos, que podem estar asso-

ciados a gêneros diferentes dos da instância alvo, mas também podem ser de alguma forma relevantes para o usuário.

Dessa forma, é possível redefinir o problema formulado na Seção 4.1 da seguinte maneira:

$$E = \arg\max_{a \in A}^{n} s'(a, g) \tag{4.2}$$

onde s' é uma função substituta tal que $s'(a, g) \approx s(a, \hat{r}(u, i))$. Em outras palavras, a importância de um atributo $a \in A$ para explicar a predição $\hat{r}(u, i)$ é capturada por meio de sua relevância para o modelo fácil de interpretar g.

4.3 Avaliação Experimental

Nesta seção, primeiramente, investigamos em detalhes as relações entre a nossa proposta de adaptação do LIME para sistemas de recomendação, chamada de LIME-RS, e o algoritmo EM-RS, a abordagem proposta por Peake & Wang (PEAKE; WANG, 2018) e detalhada na Seção 2.4.2. A seguir, comparamos diretamente as explicações do LIME-RS com o EM-RS.

Os resultados discutidos nesta seção foram publicados no artigo intitulado *Towards explaining recommendations through local surrogate models* (NóBREGA; MARINHO, 2019). Por fim, o código fonte e os dados para replicar experimentos estão disponíveis em https://github.com/caionobrega/explaining-recommendations.

4.3.1 Base de Dados

Como conjunto de dados de recomendação, foi utilizada a base de dados MovieLens 20M[1], uma das mais populares na comunidade de SR, que consiste em avaliações de usuários, em uma escala de 1-5 estrelas, em filmes (HARPER; KONSTAN, 2015). Além disso, esse dataset apresenta informações extras, tais como gêneros de filmes e tags, que podem ser usadas tanto como atributos de treinamento do modelo de recomendação como atributos de explicação.

[1] https://grouplens.org/datasets/movielens/20m/

Em nossos experimentos, usamos o histórico de 10.000 usuários selecionados aleatoriamente. O importante é que o modelo subjacente esteja bem treinado e que para isso uma amostra com 10.000 usuários é suficiente, já que muitos trabalhos usam amostras menores e os modelos de recomendação apresentam bons resultados. Também usamos gêneros de filmes como atributos de explicação, embora nossa abordagem seja flexível para lidar com atributos adicionais. No final, nosso conjunto de dados experimentais compreende 10.000 usuários, 14.359 filmes, 1.598.315 avaliações e 18 gêneros.

Os conjuntos de treinamento e teste foram gerados por meio de uma divisão temporal no conjunto de dados de forma que as avaliações registradas antes de 2010 formem o conjunto de treinamento, representando aproximadamente 85 % do total de avaliações.

4.3.2 Modelo Recomendador

Como representante dos recomendadores complexos e sabidamente classificados como caixa-preta, escolhemos o modelo *Factorization Machines* (FM) (RENDLE, 2010), um algoritmo *context-aware* baseado em fatores latentes, que modela interações esparsas entre todos os pares possíveis de atributos disponíveis. Além disso, devido à sua natureza genérica, o FM pode incluir vários modelos de fatoração específicos como casos especiais. O FM também é usado como a base de outros modelos baseados em *Deep Learning* (DL) mais recentes e mais complexos, como DeepFM (GUO et al., 2017). Os detalhes do modelo FM estão descritos na Seção 2.2.1.

Treinamos o modelo FM usando o pacote pyFM[2] com 50 fatores latentes e usando 10 iterações como critérios de parada. Os outros parâmetros são definidos com os valores padrão. Os resultados de acurácia desse modelo FM estão compatíveis, em termos de RMSE, com os *benchmarks* conhecidos (HUG, 2020), indicando que obtivemos um bom modelo. A Tabela 4.1 descreve os resultados de acurácia, calculados em cima dos dados de teste. Note que a FM foi treinada com e sem atributos de gênero.

[2]https://github.com/coreylynch/pyFM

Tabela 4.1: Valores de acurácia para o modelo FM com e sem atributos dos itens

Modelo	Métrica	Resultado
FM	Precision@10	0.2052
	Recall@10	0.027
	RMSE	0.92
FM_genre	Precision@10	0.12
	Recall@10	0.015
	RMSE	0.92

4.3.3 Relação entre LIME-RS e EM-RS

Nesta seção, discutimos as relações entre o método LIME-RS e o EM-RS. Ambos são métodos de interpretabilide baseados em modelos locais e agnósticos ao modelo subjacente, porém tratam a noção de localidade de forma diferente.

Para garantir a localidade no LIME-RS, fixamos o usuário da instância a ser explicada e amostramos outros itens. Note que os itens amostrados não pertencem necessariamente ao histórico do usuário da instância alvo. Assim, este usuário, juntamente com as predições para estes itens amostrados, formam o conjunto de treinamento para LIME-RS (vide Seção 4.2.2). Já para a localidade do EM-RS, as melhores predições, geradas pelo modelo caixa-preta para um determinado usuário alvo e seus vizinhos, formam o conjunto de treinamento para o modelo local interpretável. Assim, a localidade é capturada pela vizinhança, ou seja, usuários mais similares, do usuário alvo, para o qual as recomendações serão explicadas.

Note que, apesar da diferença na definição de localidade, os conjuntos de treinamento tanto do LIME-RS quanto do EM-RS são formados por tuplas (usuário, item). Dessa forma, se trocarmos os dados de treinamento utilizados pelo LIME-RS pelos dados de treinamento utilizado pelo EM-RS (e vice-versa), acabaríamos por ter a mesma abordagem na sua essência. Em outras palavras, seria possível treinar o modelo local interpretável do LIME-RS, usando o critério de seleção de vizinhança do EM-RS.

Ambas as abordagens têm também a noção de explicabilidade global, que visa compreender um modelo de caixa-preta como um todo. EM-RS aborda essa noção de globalidade, considerando as top predições para todos os usuários ao mesmo tempo, e não apenas para os usuários mais similares, como um conjunto de treinamento para o modelo interpretável. No LIME original, isto pode ser alcançado explicando várias instâncias individuais representa-

tivas de um modelo (RIBEIRO; SINGH; GUESTRIN, 2016b).

Além das diferentes definições de localidade e globalidade, estas abordagens adotam diferentes opções para seus modelos interpretáveis locais. Como mencionado anteriormente, o EM-RS utiliza regras de associação AR, enquanto a opção padrão do LIME-RS é o modelo de regressão, *Ridge Regression (RR)*.

Dadas as diferenças mencionadas acima, a comparação direta entre esses métodos não é óbvia. No entanto, é possível investigar se o RR é comparável ao AR no que diz respeito à sua fidelidade ao modelo de recomendação subjacente. A ideia é verificar se RR, que é o modelo padrão do LIME original e também da nossa adaptação, funciona tão bem quanto o AR. Para isso, assumimos a perspectiva de localidade (ou globalidade) do EM-RS.

Seguimos o mesmo protocolo descrito no trabalho de (PEAKE; WANG, 2018) e consideramos os cenários global e local. Para o cenário global, descrito no Quadro 2.3, o número de transações d é 30, isto é, as trinta maiores predições de cada usuário da base são usadas para treinar um único modelo interpretável. Por exemplo, se a base de dados tiver 1.000 usuários, então um conjunto com 30.000 predições vai formar o conjunto de treinamento de um modelo interpretável, que será usado para explicar todas as instâncias alvo.

Já para o cenário local, um modelo interpretável é gerado para cada usuário das instâncias alvo. Nesse caso, são levadas em consideração as 30 maiores predições dos 10 usuários mais similares ao usuário da instância alvo. Por exemplo, considerando novamente um total de 1.000 usuários, teremos 1.000 modelos, onde cada modelo é treinado com um conjunto de 300 predições. Para obter o conjunto dos 10 usuários similares de um dado usuário alvo, cada usuário é representado como um vetor de predições para os 200 itens mais populares. Em seguida, é aplicada a similaridade do cosseno para obter os 10 usuários mais similares. O Quadro 4.1 descreve a lógica do cenário local.

Para avaliar a fidelidade dos modelos interpretáveis ao modelo de recomendação caixa-preta, usamos a métrica *Fidelidade ao Modelo* (em inglês, *Model Fidelity*), também usada no trabalho de (PEAKE; WANG, 2018) e descrita na Equação 4.3. Ela é definida como a proporção de itens explicáveis (*Explainable Items*), ou seja, o conjunto de itens recuperados pelo modelo interpretável sobre os itens recomendados, conjunto de itens recuperados pelo modelo de recomendação complexo. Em outras palavras, representa a porcentagem de *overlap* entre as recomendações do modelo local e as do recomendador original. Portanto, um

Quadro 4.1: Pseudocódigo do algoritmo *Explanation Mining* com regras locais

Input: Dados de treinamento (Matriz R), Predições (Matriz $\hat{R}$), número de transações d, número de recomendações n, número de vizinhos similares k
Output: recomendações e suas respectivas explicações
01: **foreach** usuário $u \in U$:
02: obter conjunto V_u dos k vizinhos mais próximos, baseado em $\hat{R}$
03: $T = \{\}$
04: **foreach** vizinho $v \in V_u$:
05: gerar lista de transações T_v baseada nas top-d predições presentes em $\hat{R}$
06: $T = T \cup T_v$
07: gerar o conjunto de regras Z_u no formato $X \Rightarrow Y$ baseado nas transações de T
08: filtrar regras Z_u tal que $X \subseteq R$ e $Y \subsetneq R$
09: ordenar as regras baseado em algum critério (ex. suporte)
10: recomendar os top-n itens presentes em Y com as suas explicações (regras $X \Rightarrow Y$)

valor de 0,5 de *Model Fidelity* significa que o modelo local foi capaz de recuperar metade dos itens sugeridos pelo recomendador original subjacente, ou seja, se usássemos o modelo local em vez do recomendador original, seria possível recuperar metade dos itens. Nos experimentos, tanto os modelos interpretáveis quanto os de recomendação geram uma lista com as top-10 recomendações.

$$\text{Fidelidade ao Modelo} = \frac{|\text{Itens Explicáveis} \cap \text{Itens Recomendados}|}{|\text{Itens Recomendados}|} \tag{4.3}$$

Para o treinamento do modelo baseado em regras de associação (AR), usamos o algoritmo de regras de associação *apriori* com a mesma implementação e hiperparâmetros descritos no artigo original, pois o conjunto de dados do artigo original também foi o MovieLens. Além disso, também mantivemos a decisão de manter regras com tamanho 2, ou seja, um item como antecedente e um item como consequente, que visa obter explicações mais simples e diminuir o esforço cognitivo para interpretá-las.

Reportamos a fidelidade do modelo média para cada modelo interpretável na Tabela 4.2 (todos os resultados são estatisticamente significativos com $p = 0,01$). Em todos os casos, os modelos interpretáveis são capazes de recuperar pelo menos 50% dos itens. Isso significa que esses modelos podem explicar pelo menos metade da lista de recomendações. Além disso, os resultados indicam que o RR possui desempenho equivalente ao AR no cenário global, mas é melhor no cenário local. Na verdade, ambos os modelos apresentam melhor

desempenho quando consideramos o cenário local.

Tabela 4.2: Resultados para a métrica Model Fidelity ao comparar Regras de Associação (AR) Ridge Regression (RR) como modelos intrinsecamente interpretáveis.

Modelo Interpretável	Tipo	Resultado
AR	Global	0.531
	Local	0.5820
RR	Global	0.522
	Local	0.668

4.3.4 Comparando as Explicações do LIME-RS vs EM-RS

Neste experimento, comparamos as explicações de EM-RS com LIME-RS. Para executar o LIME-RS, usamos a seguinte configuração: um Ridge Regression (RR) como modelo interpretável; 1.000 amostras (ou perturbações) para treinar o modelo interpretável e a distância d cosseno, equivalente a 1 - similaridade do cosseno, como $\pi_x(z)$ para ponderar essas amostras. Para executar o EM-RS, adotamos a abordagem local conforme descrito na seção anterior.

Para explicar um item recomendado Y com o EM-RS, itens do histórico de consumo do usuário são selecionados como antecedentes X da regra de associação. Em outras palavras, apenas as regras em que o antecedente está no histórico de consumo do usuário são levadas em consideração. Observe que o EM-RS gera apenas explicações no estilo item (vide Seção 2.3.1), ou seja, ”Como você assistiu a X, recomendamos Y”.

Para comparar o LIME-RS com o EM-RS sob o mesmo aspecto, não consideramos informações extras e extraímos explicações no estilo item diretamente do LIME-RS. Isso é possível, pois o conjunto de treinamento da RR (vide Seção 4.2.1) considera usuários, itens e gêneros. Dessa forma, há um peso associado a cada item também. Em seguida, filtramos os itens para obter apenas aqueles presentes no histórico de consumo do usuário. Esse passo é realizado para satisfazer à condição do EM-RS, no qual o antecedente X da regra de associação é um item do histórico de consumo do usuário. Finalmente, recuperamos da regressão os itens associados aos top-k pesos positivos em ordem decrescente de magnitude. Ao fazer isso, estamos, essencialmente, imitando o estilo de explicação do EM-RS. Agora, para um

determinado item recomendado Y, basicamente, precisamos comparar o X do EM-RS com o X do LIME-RS.

Para selecionar os itens a serem explicados, selecionamos apenas aqueles que aparecem na mesma posição das listas de recomendações (top-10 itens) de FM e AR. Isso garante que os itens explicados são aqueles que o AR reproduziu com mais fidelidade o comportamento do FM. Terminamos com 451 itens para comparação de explicações.

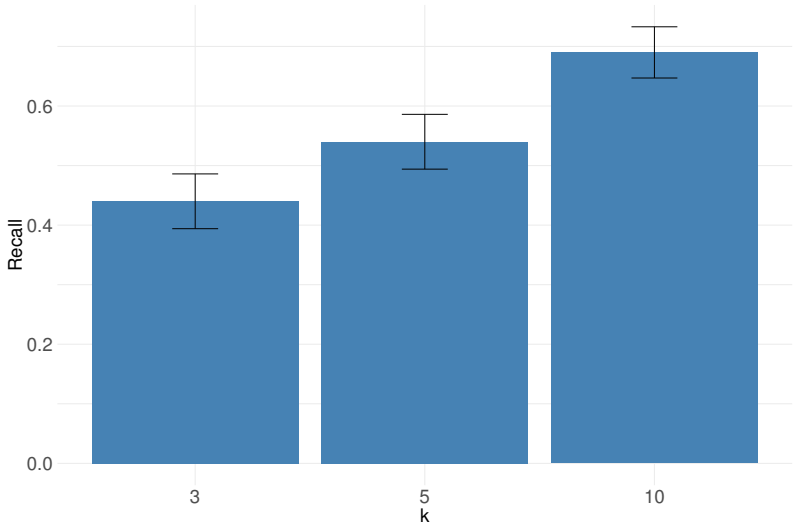


Figura 4.5: Recall do LIME-RS sobre as explicações do EM-RS. As *error bars* representam um intervalo de confiança 95%.

Para cada um dos itens selecionados para avaliação, é computada a proporção de explicações do EM-RS que foram recuperadas pelo LIME-RS. Em outras palavras, é a proporção dos itens X presentes nas regras $X \Rightarrow Y$ do EM-RS que foram recuperados pelas explicações estilo item do LIME-RS itens. Na Figura 4.5, mostramos a média de *recall* (com intervalos de confiança 95%) de todas as instâncias selecionadas, considerando as 3, 5 e 10 melhores explicações do LIME-RS.

Esse resultado mostra que, com as dez melhores explicações, o LIME-RS foi capaz de recuperar até 69 % das explicações EM-RS, o que representa uma boa concordância entre os dois métodos. Em outras palavras, apesar da diferença no que diz respeito à definição de localidade, os métodos podem ser considerados semelhantes, pois, em geral, podem gerar explicações similares quando o estilo adotado é o estilo item.

Aliás, o EM-RS, por definição, só gera explicações no estilo item. Disto isso, também é importante destacar que, diferentemente do EM-RS, o LIME-RS pode facilmente gerar

outros estilos de explicação, como o estilo atributo, que considera atributos/conteúdo do item nas explicações.

4.4 Exemplos de Explicações usando LIME-RS

Nesta seção, apresentamos alguns exemplos e ideias de como aplicar a nossa adaptação do LIME para sistemas de recomendação, LIME-RS, e interpretar suas explicações. Assim como nos experimentos da Seção 4.3, utilizamos um modelo FM, treinado com gêneros de filmes como informações extras, como modelo caixa-preta de recomendação, e o LIME-RS para explicar as predições providas por esse modelo. Dessa forma, é possível gerar explicações no estilo atributo (vide Seção 2.3.1), por exemplo, *"Eu estou recomendando esse item para você, porque os atributos 'a' e 'b' tiveram uma grande influência na formação dessa predição"*.

Vale ressaltar que, assim como o LIME, o método LIME-RS tem como objetivo explicar por que uma determinada predição foi realizada, seja ela uma predição boa ou ruim. No contexto do problema de predição de avaliações/notas em SR, predições boas e ruins são aquelas que geram, respectivamente, erros baixos e altos nas instâncias do conjunto de teste. Dessa forma, se uma dada predição for ruim, o LIME-RS pode até ajudar a depurar o modelo de recomendação, tornando explícitas suas suposições equivocadas. Tendo isso em mente, são apresentados a seguir, dois exemplos como representantes de predições boas e ruins, respectivamente, e aplicamos o LIME-RS para explicá-los. Com relação às explicações geradas, consideramos os top-10 gêneros mais importantes.

Primeiramente, analisamos as explicações para o caso de uma predição boa. A Figura 4.6 ilustra a explicação para a predição feita para um dado usuário alvo e o filme *"O Poderoso Chefão"*. Neste exemplo, a avaliação predita é cinco e a avaliação real presente no conjunto de teste também é cinco, ou seja, é uma predição perfeita. O primeiro ponto importante a destacar é que *Crime* e *Drama*, que são os gêneros do filme, estão entre os 6 gêneros mais relevantes de acordo com LIME-RS. Contudo, o gênero mais relevante para essa predição foi *Children* (infantil), que está forte e negativamente correlacionado com a avaliação predita. Isso pode ser explicado pelo fato de o usuário alvo dessa recomendação aparentemente não gostar desse gênero, pois dos 40 filmes do gênero *Children* que ele avaliou, 38 receberam

avaliações baixas. Assim sendo, a explicação gerada para esse exemplo pode ser interpretado como *"você pode gostar deste filme, uma vez que está forte e negativamente correlacionado com o 'Children', um gênero que você parece não gostar.*"

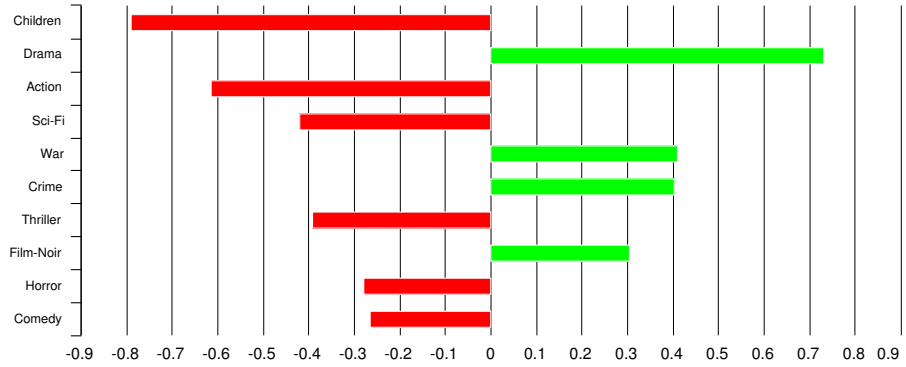


Figura 4.6: Exemplo das top-10 explicações geradas pelo LIME-RS para uma predição do filme "O Poderoso Chefão". O tamanho da barra representa a magnitude do peso associado ao gênero e a cor representa se este peso está positivamente (verde) ou negativamente (vermelho) correlacionado à predição.

A Figura 4.7 descreve a distribuição de avaliações por gênero do usuário alvo do exemplo de uma boa predição. Aqui, uma avaliação é considerada boa ou ruim se for maior ou menor do que a mediana das avaliações do usuário, respectivamente. É possível perceber que o gênero *Drama* parece ser o gênero favorito, enquanto que o gênero *Comédia* para ser o gênero que este usuário menos gosta, assim como, o gênero *Comédia*.

Como exemplo de predição ruim, foi selecionada uma predição feita para um outro usuário alvo e o filme *"O Exorcista"*. Nesse caso, a avaliação prevista e real foi 4 e 0,5, respectivamente, o que pode ser considerado uma predição claramente errada. Similarmente, analisamos este exemplo, observando primeiro as explicações geradas LIME-RS, ilustradas na Figura 4.8.

Observe que *Horror* e *Mystery*, que são os gêneros do filme, estão entre os 4 gêneros mais relevantes de acordo com LIME-RS. Neste caso, *Horror* está associado ao maior peso negativo, enquanto o gênero *Mystery* está associado ao segundo maior peso positivo. Do ponto de vista do filme, *Horror* deveria ter uma correlação forte e positiva, já que este filme realmente tem muito terror. Com relação à perspectiva do usuário, observe que entre os poucos filmes de *Horror*, aos quais ele assistiu, a maioria foi avaliada negativamente (vide Figura 4.9). Uma possível interpretação para esse resultado é que o recomendador falhou em

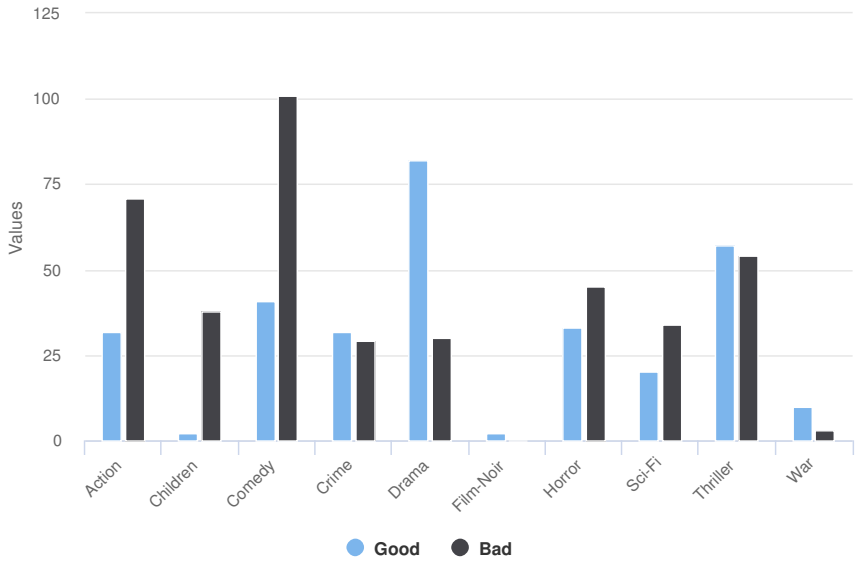


Figura 4.7: Distribuição de avaliações por gênero de filmes do usuário alvo do exemplo da Figura 4.6. Uma avaliação é considerada boa (barra azul) ou ruim (barra preta) se esta for maior ou menor do que a mediana das avaliações do usuário, respectivamente.

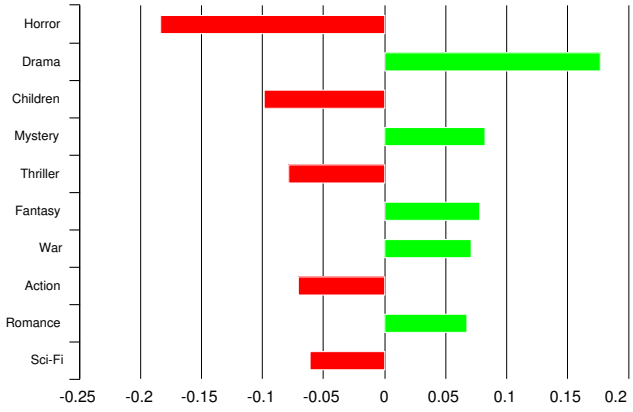


Figura 4.8: Exemplo das top-10 explicações geradas pelo LIME-RS para uma predição do "O Exorcista".

modelar a relação entre *Horror* e o filme *"O Exorcista"* e também em vincular isso ao fato verdadeiro, porém pouco frequente, de que este usuário alvo não gosta de filmes *Horror*. Já com relação ao gênero *Mystery*, do ponto de vista do filme, a explicação acertou, ou seja, o filme deve estar positivamente correlacionado com *Mystery*. Do ponto de vista do usuário alvo, por sua vez, a explicação está equivocada, este usuário parece não gostar de filmes *Mystery*. Isso pode indicar que o recomendador não foi capaz de modelar as preferências negativas do usuário em relação a *Mystery*, o que é compreensível dado o pequeno número de avaliações e a distribuição bastante semelhante de avaliações boas e ruins para filmes

deste gênero. Nesse caso, o LIME-RS pode fornecer pistas importantes sobre por que o recomendador não está funcionando corretamente.

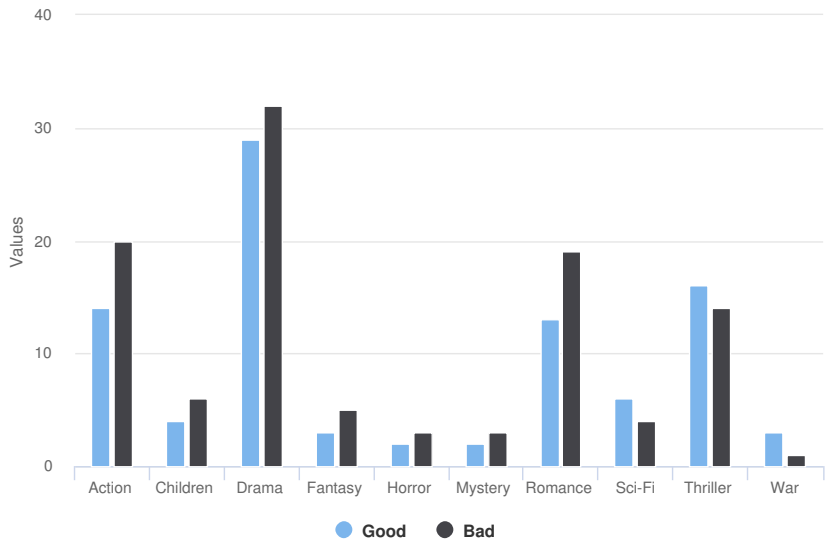


Figura 4.9: Distribuição de avaliações por gênero de filmes do usuário alvo do exemplo da Figura 4.8.

4.5 Considerações Finais

Neste capítulo, detalhamos a proposta, LIME-RS, uma adaptação do método LIME para sistemas de recomendação. Destacamos as peculiaridades que impediam o LIME de ser aplicado diretamente em SR e justificamos as mudanças implementadas para contorná-las.

Realizamos uma avaliação offline, comparando o LIME-RS com o EM-RS, método de interpretabilidade *post-hoc* baseado em regras de associação, proposto por (WANG et al., 2018). Investigamos o grau em que ambas as abordagens geram as mesmas explicações para um dado conjunto de predições que se deseja interpretar. Observamos que, em geral, esses modelos são semelhantes entre si, pois geram explicações similares, onde a principal diferença é definição de localidade. Porém, nossa proposta tem a vantagem adicional de gerar explicações em diferentes estilos.

Ao final, discutimos por meio de exemplos como aplicar e interpretar as explicações providas pelo LIME-RS.

Capítulo 5

Análise de Estratégias de Localidade para Modelos Locais Interpretáveis

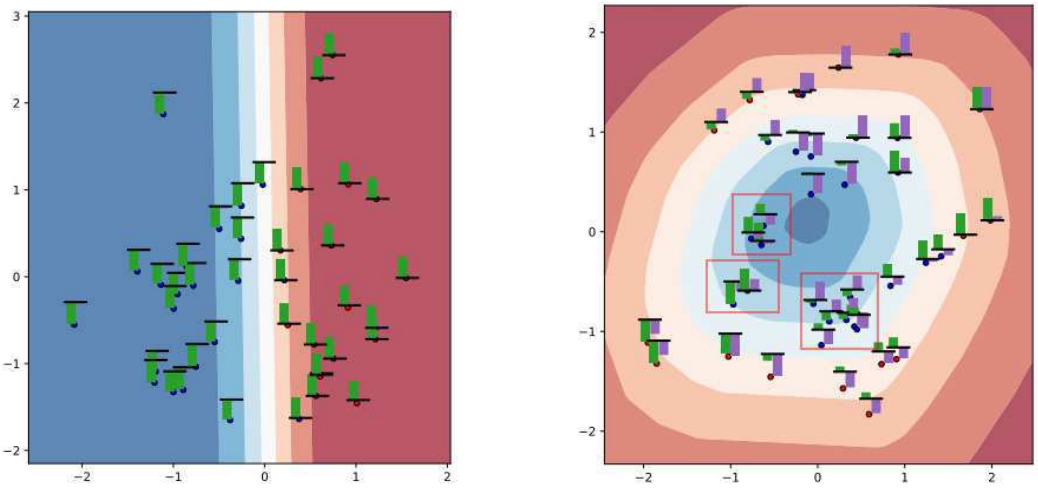
Neste capítulo, expandimos a investigação do LIME-RS, detalhada no Capítulo 4. Primeiramente, discutimos como a definição de localidade impacta a abordagem baseada em modelos locais interpretáveis e, consequentemente, as explicações extraídas deles (Seção 5.1). Então, propomos novas estratégias para definição de localidade, ou seja, novas formas de gerar vizinhanças para o contexto de recomendações (Seção 5.2). Por fim, descrevemos uma avaliação offline para comparar as novas estratégias de localidade propostas (Seção 5.3).

5.1 Problemas Relacionados à Abordagem de Modelos Locais Interpretáveis

O aprendizado de modelos locais interpretáveis é apenas uma das abordagens de interpretabilidade *post-hoc* (MOLNAR, 2019). Os métodos que seguem essa abordagem são caracterizados por construir um conjunto de treinamento nas redondezas de uma dada instância alvo para a qual a predição se deseja explicar. A intuição é que um modelo interpretável, treinado com esse conjunto de dados, imite localmente o comportamento de um modelo *black-box* que gerou uma predição para a instância alvo. Sendo assim, a definição de localidade está fortemente relacionada à fidelidade do modelo interpretável ao modelo *black-box*.

A estratégia de localidade do método LIME (RIBEIRO; SINGH; GUESTRIN, 2016b)

envolve a criação de um conjunto de dados sintéticos por meio de perturbações na representação interpretável da instância alvo. Note que, nesse passo, as perturbações são amostradas aleatoriamente e, para capturar, de fato, a noção de localidade, é utilizada uma função $\pi_x(z)$ para ponderar a proximidade entre cada perturbação sintética z e a instância alvo x. Vale lembrar que uma forma de ajustar a definição de localidade consiste em mudar a configuração da função de proximidade $\pi_x(z)$, por exemplo alterando o chamado parâmetro *kernel width*. A Seção 2.4.2 dá mais detalhes sobre esse processo do LIME.



(a) Predições e suas explicações para um SVM linear.

(b) Predições e suas explicações para um classificador baseado em redes neurais.

Figura 5.1: Exemplo de explicações providas pelo LIME para dois classificadores binários. O *heatmap* (*background* nas cores azul e rosa) retrata a fronteira de decisão dos classificadores. Cada ponto está associado à predição do classificador (cores azuis e rosa dos pontos), bem como, à explicação dessa predição (gráfico de barras, que indica a força que os atributos x_1 (verde) e x_2 (roxo) representam para a predição). As explicações para o modelo não-linear (imagem b) sofrem variações (regiões destacadas), e, portanto, são exemplos de explicações instáveis. Figura adaptada de (ALVAREZ-MELIS; JAAKKOLA, 2018).

Trabalhos relacionados exploraram essa característica de localidade e identificaram deficiências tanto do LIME como em outros métodos baseados em modelos locais. Por exemplo, o trabalho de (ALVAREZ-MELIS; JAAKKOLA, 2018) argumenta que tais métodos devem gerar explicações robustas, isto é, para instâncias similares, ou seja, instâncias que estão próximas entre si numa vizinhança local, devem ser geradas explicações similares. Esse trabalho aponta que as explicações oriundas do LIME podem ser robustas/estáveis quando o modelo subjacente é linear, porém para modelos não lineares isso nem sempre acontece.

A Figura 5.1 ilustra o problema mencionado acima. Nesse exemplo, o LIME gera explicações para dois classificadores binários, um SVM linear (imagem a) e uma rede neural com duas camadas (b), treinados em uma base de dados sintética de duas dimensões (atributos x_1 e x_2). Nas imagens, estão espalhados pontos, que representam instâncias de teste. A cor do ponto indica a predição do classificador subjacente, enquanto que as explicações para as predições estão representadas em um gráfico de barras. Esse tipo de visualização de explicações é similar ao adotado nos exemplos da Seção 4.4 e pode ser interpretado da seguinte maneira. Aqui, a cor da barra indica o atributo: verde para x_1 e roxo para x_2; a altura da barra indica a força da correlação entre o atributo e predição do classificador; e o sentido da barra indica se o atributo está correlacionado à classe rosa. Por exemplo, na imagem a, as explicações mostram que o atributo relevante é o x_1, pois, para todas as predições, a barra verde é a que se destaca. Além disso, para predizer as instâncias como a classe azul, as barras estão apontadas para baixo, enquanto que, para a classe rosa, elas estão apontadas para cima. Essas explicações fazem sentido, uma vez que o problema da imagem a é linearmente separável, bastando apenas considerar uma dimensão, a do atributo x_1, para realizar a classificação. Portanto, as explicações para o modelo linear são estáveis, ou seja, para instâncias próximas/similares, os gráficos de barras são similares. Por outro lado, as explicações para o modelo não-linear (imagem b) são mais instáveis, ou seja, para instâncias próximas, as explicações sofrem variações. Isso pode ser percebido nas regiões destacadas.

Ainda sobre o problema da instabilidade, (TAN et al., 2019) alerta para o fato de que, se um método produz explicações não confiáveis para as predições de um modelo *black-box* subjacente, também surge a desconfiança sobre o modelo *black-box* em si. Os autores também apontam deficiências do LIME e destacam que a aleatoriedade no processo de amostragem da vizinhança da instância alvo é uma das fontes das inconsistências/instabilidades.

A literatura apresenta propostas para contornar esse problema relacionado à definição de localidade. Por exemplo, o trabalho de (LAUGEL et al., 2018) propõe uma estratégia para geração de vizinhança sintética local, de forma que as instâncias sejam sintetizadas dentro de uma região esférica, cujo centro é a instância alvo. O princípio dessa proposta é que o raio dessa região esférica seja determinado pela distância entre a instância alvo e a instância real mais próxima, cuja predição de classe seja diferente da predição de classe da instância alvo. Mais formalmente, temos que, dados a instância alvo x e o modelo classificador f, se deseja

encontrar a instância real mais próxima b, tal que $f(x) \neq f(b)$. A intuição por trás dessa proposta é que b vai pertencer à fronteira de decisão de f mais próxima a x, logo, amostrar instâncias (sintéticas) nessa região, baseada em x e b, tende a capturar melhor essa fronteira de decisão local. Dessa forma, o problema da aleatoriedade na geração da vizinhança do LIME original pode ser minimizado, pois a vizinhança agora tende a ser mais coesa em termos de distância para a instância alvo. Vale ressaltar que ainda se mantém a aplicação da função de proximidade $\pi_x(z)$.

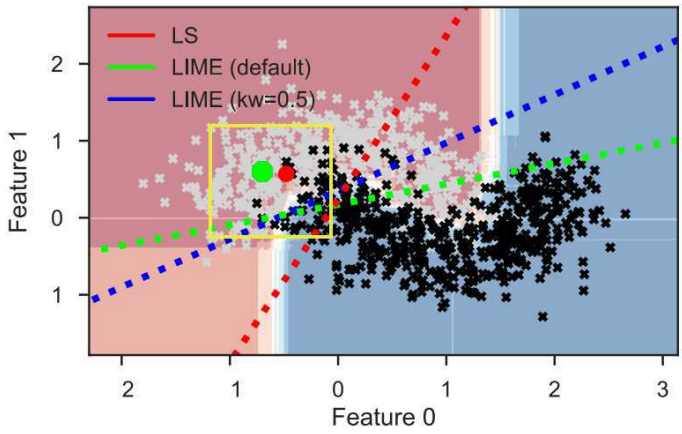


Figura 5.2: Exemplo de três modelos locais interpretáveis, aproximando, de forma local, a fronteira de decisão (áreas azul e rosa) de um classificador binário complexo ao predizer uma instância (ponto verde). A linha verde representa o modelo linear aprendido pelo LIME original com a configuração padrão. A linha azul representa o modelo linear aprendido pelo LIME original com *kernel width = 0,5*. A linha vermelha é o modelo aprendido por uma modificação do LIME, que cria dados sintéticos numa região esférica delimitada pela instância real mais próxima (ponto vermelho), porém com predição diferente da instância alvo. A linha vermelha é o modelo que melhor captura a fronteira de decisão local, área destacada em amarelo. Figura adaptada de (LAUGEL et al., 2018).

.

A Figura 5.2 ilustra a aplicação da solução proposta por (LAUGEL et al., 2018). Nesse exemplo, três modelos locais interpretáveis tentam aproximar o comportamento local de um classificador binário complexo ao predizer uma instância alvo (ponto verde). Primeiramente, note que uma aproximação da fronteira de decisão global seria uma linha horizontal na altura do valor 0 da *Feature* 1, pois separaria os pontos pretos dos pontos cinza. Essa seria uma aproximação grosseira, uma vez que os dados não são lineares e, por isso, não podem ser perfeitamente separados por uma reta. Note também que, nas redondezas da instância alvo, uma linha horizontal não separaria bem os dados, pois a fronteira de decisão tende a ser

mais inclinada no sentido vertical, como a região em amarelo destaca. Dessa forma, essa é a fronteira de decisão local que um modelo treinando nas vizinhanças da instância alvo deveria capturar. A fronteira de decisão aprendida pelo LIME original com a configuração padrão (linha verde) é mais horizontal e parece ser uma aproximação mais global do que local. Uma variação do LIME original com *kernel width = 0,5*, definido no passo 4 do LIME original (vide Seção 2.4.2), se aproxima mais da fronteira de decisão local, mas não o suficiente. A solução de (LAUGEL et al., 2018) é a que captura a fronteira de decisão local mais fielmente, onde o ponto vermelho delimita o raio da região esférica.

O trabalho de (JIA et al., 2019) também propõe outra forma de geração de vizinhança para métodos baseados em modelos locais interpretáveis. A solução utiliza técnicas de redução de dimensionalidade, tais como *Principal Component Analysis (PCA)*, para obter dados sintéticos mais fiéis à instância alvo. A intuição é que, ao mapear os dados originais em um espaço de menor dimensão, os dados sintéticos que vão compor a vizinhança preservem as características principais da localidade da instância alvo.

Por fim, o trabalho de (PEAKE; WANG, 2018), utilizado nos experimentos do Capítulo 4, também trata desse assunto de forma indireta, ao usar uma técnica de clustering hierárquico para tentar aprimorar, seguindo sua definição de localidade, a forma de selecionar usuários mais próximos (vide Seção 2.4.2 para mais detalhes desse trabalho). Essa ideia, porém, obteve piores resultados que a ideia descrita na Seção 4.3.3 e, portanto, descartada dos experimentos deste capítulo.

5.2 Propostas de Novas Estratégias de Localidade para o Contexto de Recomendações

Na seção anterior, foram discutidos problemas acerca da definição de localidade em métodos de modelos locais interpretáveis. Especificamente, foi mostrado que a definição de localidade adotada pelo LIME pode impactar negativamente as suas explicações, tornando-as instáveis.

Os experimentos avaliaram essa questão em tarefas de classificação, mas, essencialmente, tais problemas também podem ocorrer no contexto de sistemas de recomendação, uma vez que estes envolvem dados e modelos tão complexos quanto os presentes nas tarefas

de classificação.

As propostas descritas para contornar o problema de explicações instáveis do LIME focam em estratégias de criação de dados sintéticos para formar a vizinhança local da instância alvo. Essas estratégias foram aplicadas e fazem mais sentido em dados tabulares com atributos numéricos reais, pois basta usar uma distribuição de probabilidade para gerar novas instâncias sintéticas. Essa estratégia não é diretamente aplicável ao contexto de recomendação, por causa das peculiaridades das entidades usuários e itens. Em dados de SR, os usuários e itens costumam apresentar perfis bem específicos baseados nos históricos de consumo dos usuários, o que faz com que a tarefa geração de dados sintéticos nesses casos não seja trivial.

Na proposta do LIME-RS, descrita no Capítulo 4, uma das adaptações envolve justamente a etapa de geração da vizinhança. A ideia naquele momento partia da seguinte premissa: dada a instância alvo (u, i), a noção de localidade é capturada ao fixar o usuário u. Os itens, necessários para completar a tupla, são amostrados de maneira aleatória, seguindo a mesma lógica do LIME original. Note que, no LIME, a aleatoriedade no processo de amostragem da vizinhança foi destacado como fonte das inconsistências/instabilidades de suas explicações.

Considerando as evidências da literatura, discutidas na seção anterior, e dado que as soluções, com o melhor do nosso conhecimento, não foram aplicadas para o contexto de recomendação, surgiu a necessidade de investigar essa linha de pesquisa e propor novas estratégias de geração de vizinhança para o LIME-RS. É importante destacar que as estratégias estão relacionadas com os trabalhos da seção anterior, no entanto são empíricas, ou seja, não há uma contribuição teórica como nesses trabalhos.

Inspirados pelas abordagens de recomendação, Filtragem Colaborativa (FC) e a baseada em conteúdo, propomos, a seguir, novas estratégias para composição da vizinhança local do LIME-RS. Vale ressaltar que todas as novas estratégias mantêm a premissa original do LIME-RS: todas as instâncias que vão compor a vizinhança local Z têm o mesmo usuário u da instância alvo (u, i) para a qual a predição se deseja explicar. Assim, a ideia é que o comportamento do recomendador em outros itens considerando o mesmo usuário pode ajudar a explicar seu comportamento na instância alvo. Além disso, há também uma função de proximidade $\pi_x(z)$ para ponderar as instâncias $z \in Z$.

Para todos os exemplos abaixo, considere que queremos explicar a predição, provida por

um recomendador r, para o filme *'Toy Story'*, cujos gêneros são *'Animação', 'Children' e 'Comédia'* ou seja, a instância alvo é $(u$, *'Toy Story'*$)$.

- **random:** os itens são selecionados de forma aleatória. Essa é a estratégia padrão de seleção do LIME-RS, que, por sua vez, foi inspirada na estratégia do LIME original. Como discutido anteriormente, a noção de localidade é basicamente capturada pela função de proximidade $\pi_x(z)$ (distância do cosseno), já que, pelo fato da aleatoriedade, é possível selecionar itens bastante diferentes do item i, o que, intuitivamente, não está alinhado com a ideia de localidade/vizinhança. Por exemplo, suponha que a seleção aleatória trouxe, por exemplo, 50% de filmes de terror e 50% de drama. Como esses são gêneros bastante diferentes de animação, o explicador não terá como prover uma boa explicação nesse caso. Em outras palavras, não há garantia de que filmes de animação (ou com gêneros em comum à instância alvo) serão selecionados. Portanto, teoricamente, segundo essa estratégia, a vizinhança Z pode ser composta de tal forma que a noção de localidade provida pela função $\pi_x(z)$, não seja robusta. Essa estratégia está formalizada na Equação 5.1:

$$Z_{random} = random(I \setminus \{i\}, k) \tag{5.1}$$

  onde $I \setminus \{i\}$ é o conjunto de todos os itens, exceto o item i da instância alvo; $random$ é uma função de amostragem que segue uma distribuição uniforme; e k é o tamanho da vizinhança.

- **content-based:** os itens são selecionados considerando o espaço de atributos A. Essa estratégia é inspirada na abordagem de SR baseado em conteúdo, detalhada na Seção 2.2.1, onde os itens são representados por vetores da seguinte forma. Um dado item j pode ser descrito como um vetor de pesos $\mathbf{j}_A = (w_1, w_2, ..., w_{|A|})$, onde cada valor w é um peso que representa a relevância de cada atributo $a \in A$ em relação a j. Supondo $A = \{$*'Animação'*, *'Children'*, *'Comédia'*, *'Drama'*, *'Romance'*$\}$, o item i da instância alvo pode ser representado usando o *multi-hot encoding*, de forma que temos $\mathbf{i}_A = (1, 1, 1, 0, 0)$. O princípio dessa estratégia é compor a vizinhança Z com itens similares ao item i no que diz respeito aos atributos que eles contêm. A intuição é definir a

localidade de forma que o comportamento do recomendador r em outros itens com atributos similares pode ajudar a explicar seu comportamento na instância alvo. Por exemplo, considere os seguintes filmes: *'O Rei Leão'*, cujos gêneros são *'Animação', 'Children' e 'Musical'*, e *'Titanic'*, cujos gêneros são *'Romance', 'Drama'*. Nesse caso, *'O Rei Leão'* vai estar mais próximo a *'Toy Story'*, uma vez que compartilham atributos em comum, e, portanto, terá mais chances de fazer parte da vizinhança local Z da instância alvo. A Equação 5.2 formaliza essa estratégia.

$$Z_{content\_based} = \underset{z \in I \setminus \{i\}}{\arg\max^{k}} sim(\mathbf{z}_A, \mathbf{i}_A) \tag{5.2}$$

onde $I \setminus \{i\}$ é o conjunto de todo os itens, exceto o item i da instância alvo; $\mathbf{z}_A \in \{0,1\}^{|A|}$ e $\mathbf{i}_A \in \{0,1\}^{|A|}$ representam os vetores de atributos, considerando um espaço de atributos A, presentes nos item z e i, respectivamente; sim é uma função de similaridade, por exemplo, a similaridade do cosseno (Equação 2.1), e k é o tamanho da vizinhança.

- **rating-based:** os itens são selecionados considerando o espaço de avaliações R. Essa estratégia é inspirada na abordagem FC baseado no item, detalhada na Seção 2.2.1, onde os itens são representados por vetores de avaliações. Especificamente, um dado item j pode ser representado pelo vetor $\mathbf{j}_R = (r_{u_1,j}, r_{u_2,j}, ..., r_{u_{|U|},j})$, onde $r \in R$, ou seja, um vetor que contêm as avaliações que usuários deram a j. O princípio dessa estratégia é compor a vizinhança Z com itens similares ao item i no que diz respeito às avaliações que receberam. A intuição é definir a localidade de forma que o comportamento do recomendador r em outros itens com histórico de avaliações similares pode ajudar a explicar seu comportamento na instância alvo. Note que essa estratégia está relacionada à estratégia de vizinhança *content-based*, porém, aqui a noção de localidade se baseia no caráter colaborativo das preferências dos usuários, que um recomendador baseado em FC tenta aprender. Em outras palavras, a noção de localidade consiste em itens que têm performances parecidas com o item i em termos de aceitação por parte dos usuários. Por exemplo, supondo que os usuário que gostaram do filme *'Toy Story'*, também gostaram do filme *'Star Wars'*, logo, *'Star Wars'*, apesar de não ter gênero em comum com filme da instância alvo, é um candidato com mais chances

de ser selecionado para compor a vizinhança de *'Toy Story'*. A Equação 5.3 formaliza essa estratégia.

$$Z_{rating\_based} = \underset{z \in I \setminus \{i\}}{\arg\max^{k}} sim(\mathbf{z}_R, \mathbf{i}_R) \tag{5.3}$$

onde $\mathbf{z}_R \in \mathbb{R}^{|U|}$ representa o vetor de avaliações que o item z recebeu, sim é uma função de similaridade, por exemplo, a similaridade do cosseno (Equação 2.1); e k é o tamanho da vizinhança.

- **user-content_based:** variação da estratégia *content-based*, onde é dada prioridade aos itens do histórico do usuário u da instância alvo. A ideia é dar mais força às preferências do usuário na hora de construir a vizinhança local Z. Por exemplo, suponha dois filmes *'O Rei Leão'*, cujos gêneros são *'Animação', 'Children' e 'Musical'*, e *'Peter Pan'*, cujos gêneros são *'Animação', 'Children', 'Fantasia' e 'Musical'*, onde apenas o segundo pertence ao histórico do usuário. Note que ambos os filmes têm dois gêneros em comum com *'Toy Story'*, *'Animação', 'Children'*, mas, seguindo a lógica dessa estratégia, o filme *'Peter Pan'* terá prioridade na formação da vizinhança Z. Note também que, não necessariamente, haverá itens do histórico do usuário em Z, pois o item da instância alvo pode ter atributos que nenhum filme do histórico do usuário tem. Por exemplo, supondo que o histórico do usuário u seja formado apenas por filmes de *'Ação'* e *'Crime'*, essa estratégia equivaleria à estratégia *content-based*. A Equação 5.4 formaliza essa estratégia.

$$Z_{user-content\_based} = \underset{z \in I_u}{\arg\max^{m}} sim(\mathbf{z}_A, \mathbf{i}_A) \cup Z_{content\_based} \tag{5.4}$$

onde I_u representa o histórico de itens do usuário u; $\mathbf{z}_A \in \{0,1\}^{|A|}$ e $\mathbf{i}_A \in \{0,1\}^{|A|}$ representam os vetores de atributos, considerando um espaço de atributos A, presentes nos item z e i; sim é uma função de similaridade; e m + $|Z_{content\_based}|$ é o tamanho da vizinhança.

- **user-predicted_ratings:** apenas itens do histórico do usuário u da instância alvo. A ideia consiste em restringir a vizinha exclusivamente ao usuário u, de forma que, a no-

ção de localidade considere apenas as preferências do usuário e o que o recomendador r aprendeu sobre elas. Em outras palavras, os usuários levam em consideração diferentes aspectos ao aceitar uma recomendação, ou seja, apresentam preferências particulares. De maneira geral, um recomendador é considerado acurado quando consegue aprender corretamente as preferências dos usuários de forma personalizada. Sendo assim, considerar exclusivamente o histórico dos usuários é uma maneira de capturar as preferências individuais mais fielmente. A Equação 5.5 formaliza essa estratégia.

$$Z_{user-predicted\_ratings} = \{(u, i)|i \in I_u\} \tag{5.5}$$

onde I_u representa o histórico de itens do usuário u;

5.3 Avaliação Experimental

Nesta seção, comparamos as diferentes estratégias de geração de vizinhança, descritas na Seção 5.2. Baseado nos protocolos dos experimentos presentes nos trabalhos relacionados, é realizada uma avaliação offline com o objetivo de responder a questão de pesquisa (QP2): *Qual é o impacto da formação da vizinhança no método LIME na qualidade de suas explicações?*

5.3.1 Base de Dados

Assim como nos experimentos do Capítulo 4, também utilizamos o MovieLens, disponibilizada publicamente pelo grupo de pesquisa GroupLens da Universidade de Minnesota[1]. Dessa vez, porém, experimentamos com a versão 1M. Ela apresenta as mesmas caraterísticas dos dados da versão 20M: avaliações de filmes em uma escala de 5 estrelas, e informações sobre o gêneros dos filmes. As diferenças são que ela contém apenas avaliações com estrelas inteiras e não há informações sobre *tags*. Aliás, o fato de realizar experimentos com um universo mais variado de atributos, além dos gêneros, foi o principal motivo para a escolha da versão 20M nos experimentos do Capítulo 4. Contudo, ao inspecionar os dados mais

[1]<https://grouplens.org/datasets/movielens/>

detalhadamente, foi percebido que as *tags* apresentavam bastante ruído, principalmente, por serem criadas de forma colaborativa, e por essa razão não foram incluídas nos experimentos.

Foram removidos 224 filmes, e suas avaliações, do conjunto de dados original. Esse passo foi realizado para atender a um requisito de uma ferramenta desenvolvida nesta tese para analisar os resultados. O requisito consistia em manter apenas filmes que possuíssem seus ids mapeados para o TMDB[2], base de onde são extraídas outras informações necessárias para a nossa ferramenta, por exemplo, os posteres dos filmes. Além disso, 12 perfis de usuários fictícios foram incluídos com o propósito de servirem para testes de sanidade, explicados na Seção 5.3.3. A Tabela 5.1 sumariza os dados após essas modificações.

Tabela 5.1: Descrição da base de dados utilizada nos experimentos.

Base de Dados	**|U|**	**|I|**	**|A|**	**#Avaliações**	**Tipo**	**Domínio**
ML-1M	6.052	3.676	18	998.505	Ratings	Filmes

5.3.2 Modelo Recomendador

Como representante de recomendadores complexos e sabidamente classificados como caixa-preta, escolhemos o modelo Wide&Deep (CHENG et al., 2016), um algoritmo que combina as abordagens baseadas em *Deep Learning* (DL) e *Matrix Factorization* (MF). Esse modelo foi escolhido por ser mais recente e ser ainda mais complexo que o FM, utilizado nos experimentos do Capítulo 4. Os detalhes do modelo Wide&Deep estão descritos na Seção 2.2.1.

5.3.3 Protocolo de Avaliação

Como dito anteriormente, neste experimento realizamos uma avaliação offline. Nesse tipo de avaliação, é preciso definir um *ground truth*, conjunto que, teoricamente, deve ser o resultado esperado ideal para uma dada tarefa. Assim sendo, é possível comparar resultados providos por diferentes abordagens de forma que, quanto mais próximo do *ground truth* o resultado de uma dada abordagem for, melhor será o desempenho dessa abordagem na tarefa em questão.

Como formalizado na Seção 4.1, o nosso problema pode ser resumido em encontrar um subconjunto de atributos que melhor expliquem uma dada recomendação. Por exemplo, em

[2]https://www.themoviedb.org/

um cenário de recomendação de filmes, na sentença *'recomendamos este filme para você porque ele tem **ação** e **comédia**"*, os gêneros em destaque são os atributos que explicam a recomendação. Dessa forma, dado o nosso problema, o *ground truth* consiste justamente no conjunto de atributos mais relevantes que, idealmente, formam a melhor explicação para uma predição de uma recomendação.

Nos trabalhos relacionados, a construção do *ground truth* para métodos de interpretabilidade *post-hoc* é baseada no uso de um modelo intrinsecamente interpretável, treinado diretamente nos dados de treino do modelo complexo subjacente. Por exemplo, nos experimentos offline do artigo do LIME original, um dos modelos escolhidos para formar o *ground truth* é uma árvore de decisão, detalhada na Seção 2.4.1. Os atributos extraídos da árvore de decisão formam o conjunto de explicações consideradas ideais. A premissa é que um modelo intrinsecamente interpretável, treinado diretamente nos dados de treino pode, provavelmente, recuperar os atributos mais relevantes (o *ground truth*) para o problema em questão. Assim, é assumido que as explicações providas por eles podem servir como referência de qualidade (ou fidelidade) para as explicações extraídas por métodos de interpretabilidade *post-hoc*.

No contexto de sistemas de recomendação, essa ideia também pode ser implementada. Por exemplo, um modelo de recomendação baseado em conteúdo (vide Seção 2.2.1) é considerado *white box*, logo, interpretável por definição, e portanto, pode ser utilizado para gerar um *ground truth* para as predições de um modelo de recomendação *black box* e complexo. A seguir, é detalhada uma proposta de geração de *ground truth* que usa um modelo baseado em conteúdo.

Proposta de Geração de Explicações Ideais no Contexto de Recomendações

A literatura define diferentes estilos de como apresentar uma explicação de uma recomendação (vide Seção 2.3.1). Utilizar informações extras, tais como atributos dos usuários ou dos itens, é um dos estilos mais populares em SR.

O estilo baseado em atributos dos itens é a forma natural de explicar modelos de recomendação baseados em conteúdo, pois, na lógica desse modelos, usuários e itens são representados como vetores de atributos. Dessa forma, um exemplo de explicação para um modelo de recomendação baseado em conteúdo seria: *"O filme 'Procurando Nemo' foi recomendado porque é uma animação, gênero que você frequentemente está interessado"*. Note

que, implicitamente, essa explicação está levando em consideração a perspectiva do usuário quando indica a frequência dessa preferência específica, e a perspectiva do item, pois cita o gênero do filme.

De fato, é possível definir que uma explicação "ideal" deveria levar em consideração tanto a perspectiva do usuário quanto a do item recomendado (o termo ideal está entre aspas, pois essa é uma definição particular desta tese). Essa definição pode ser justificada pelos seguintes fatores. A literatura mostra que um recomendador personalizado acurado aprende as preferências do usuário, que estão obviamente descritas no histórico dele. Logo, uma explicação "ideal" deveria refletir esse fato. Além disso, a literatura mostra que a relevância do item para um dado usuário depende da vizinhança de usuários similares (para a abordagem baseada em FC) ou dos atributos (para a abordagem baseada em conteúdo). Logo, uma explicação "ideal" deveria também capturar essa ideia.

Na Seção 2.3.3, foi discutido que uma explicação pode ter diferentes objetivos. Um dos objetivos é a transparência, que visa prover ao usuário a lógica por trás da recomendação, ou no nosso caso, prover como a predição de uma avaliação foi obtida. A nossa proposta de explicação "ideal", ao tentar deixar claro quais são atributos mais relevantes para uma dada predição de avaliação, joga luz sobre a lógica do recomendador e, portanto, está relacionada ao objetivo transparência.

Para satisfazer os requisitos de capturar as perspectivas do usuário e do item, a nossa proposta de uma explicação "ideal" $\mathbf{e}_{u,i}$ para uma dada instância alvo (u, i) é obtida com os seguintes passos. Como explicado na Seção 2.2.1, dado um conjunto de atributos A, o perfil $\mathbf{i}$ do item i pode ser descrito como um vetor de pesos $\mathbf{i} = (w_1, w_2, ..., w_{|A|})$. Nesta proposta, os pesos são computados via TF-IDF, pois por meio dessa técnica é possível obter uma representação mais informativa do que a representação *one-hot encoding*, por exemplo. O *corpus* utilizado para obter os valores de TF-IDF, é o conjunto I composto por todos os itens da base de dados. A ideia é obter, de forma geral, a importância dos atributos para o item i. Esse passo está ilustrado na parte superior da Figura 5.3 e representa a captura da perspectiva do item.

Para capturar a perspectiva do usuário, utilizamos o seu histórico de avaliações. Novamente, cada item do histórico é representado por vetores com valores TF-IDF. Porém, dessa vez, o *corpus* utilizado para obter esse valores é apenas o histórico do usuário, ou seja, o

conjunto I_u. Note que, dessa forma, um mesmo item poderá ter uma representação diferente para cada usuário. A intuição é que, ao diminuir o escopo do *corpus*, seja possível capturar as preferências do usuário u de forma mais fiel. Os itens do histórico, representados em vetores, e suas respectivas avaliações (conjunto R_u) formam o conjunto de treinamente de uma regressão linear, usada para aprender o perfil $\mathbf{u}$ do usuário u. Especificamente, o propósito da regressão é aprender a preferência do usuário por cada atributo $a \in A$. Dessa forma, o usuário u também pode ser representado por um vetor $\mathbf{u} = (v_1, v_2, ..., v_{|A|})$, onde cada valor $v \in \mathbb{R}$ é um coeficiente da regressão e indica a importância de v para u. Esse passo está ilustrado na parte inferior da Figura 5.3.

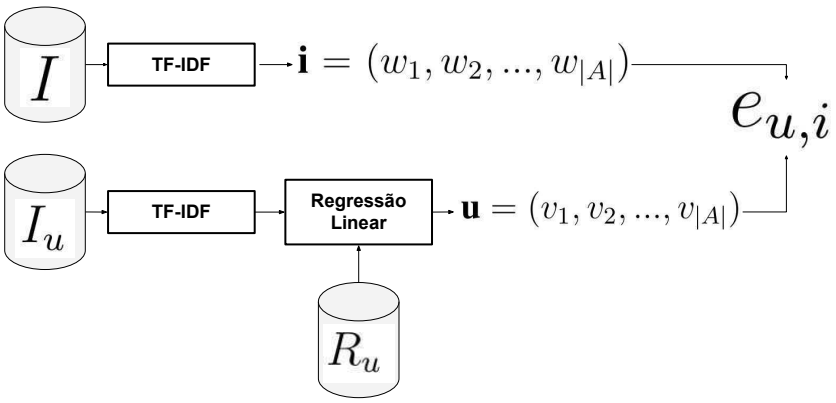


Figura 5.3: Ilustração dos passos para a geração de uma explicação "ideal" $e_{u,i}$, segundo a proposta desta tese, para a instância alvo (u, i). Definição baseada na abordagem de recomendação baseada em conteúdo.

Por fim, os perfis do usuário e do item são agregados, calculando a média dos respectivos pesos w_j e v_j, como mostra a Equação 5.6. A ideia é que a combinação dos perfis seja a explicação "ideal" para (u, i).

$$e_{u,i} = ((w_1 + v1)/2, (w_2 + v2)/2, ..., (w_{|A|} + v_{|A|})/2) \tag{5.6}$$

A Figura 5.4 ilustra um exemplo de explicação "ideal" para um usuário, cujo perfil foi criado artificialmente com a seguinte lógica para avaliar filmes: avaliações com 5 estrelas são atribuídas a filmes de um dado gênero configurado como favorito; avaliações com 1 e 2 a filmes de um dado gênero configurado malquisto; e avaliações com 3 e 4 para filmes neutros. Os filmes são amostrados aleatoriamente. A necessidade de perfis artificiais é justificada

para realização de testes de sanidade, isto é, verificar se a explicação "ideal" faz sentido em um cenário controlado.

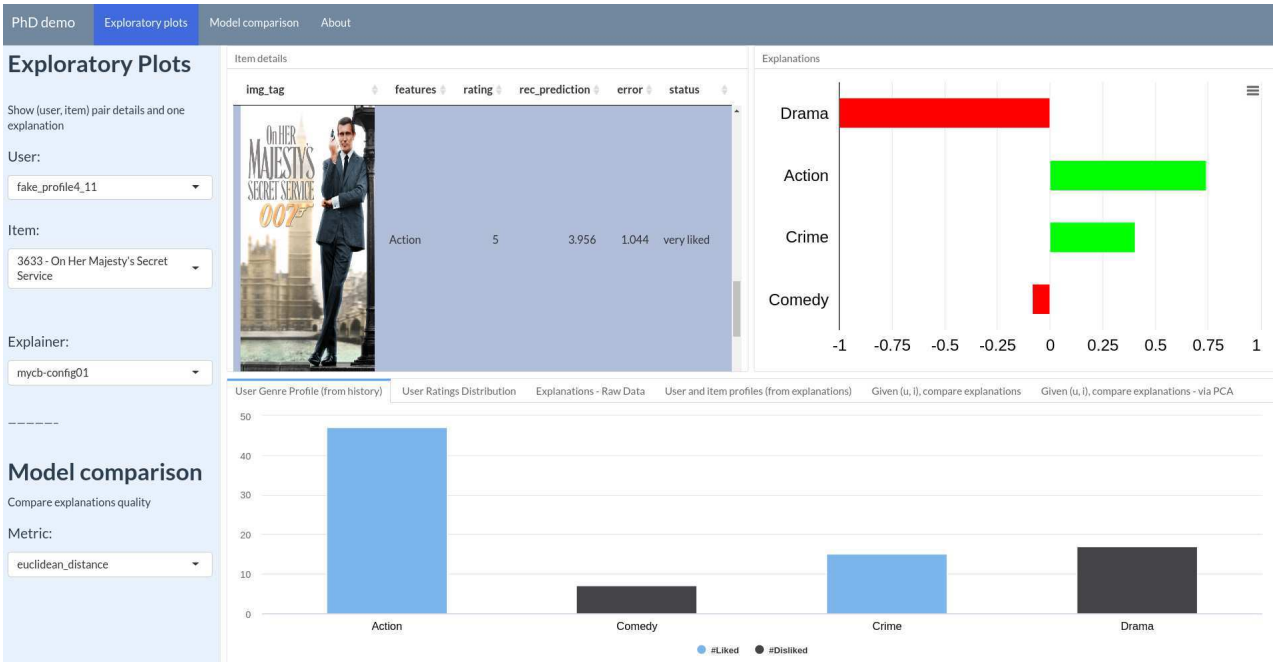


Figura 5.4: Exemplo de uma explicação "ideal" para um usuário criado artificialmente. Figura extraída de uma ferramenta desenvolvida nesta tese para realizar experimentos e analisar resultados.

Nesse exemplo, o usuário alvo apresenta 80 filmes no seu histórico de consumo e uma média de avaliações de 4,17. O perfil desse usuário fictício foi configurado da seguinte maneira: *"Ação"* e *"Crime"* são os gêneros favoritos; *"Drama"* é o gênero que ele não gosta; e *"Comédia"* foi configurado como neutro. A guia *'User Genre Profile (from history)'*, parte inferior da Figura 5.4, ilustra essas informações acerca do perfil desse usuário. Observe que, como esperado, as barras que representam *"Ação"* e *"Crime"* são azuis, indicando que todos os filmes desses gêneros foram avaliados como acima da média do usuário. Por outro lado, *"Comédia"* e *"Drama"* são barras pretas, pois todas as avaliações dos filmes com esses gêneros ficaram abaixo da média. Sendo assim, uma explicação "ideal" para esse usuário alvo deveria capturar tais preferências, bem como dar importância para os atributos do item recomendado, que nesse exemplo é também o gênero *"Ação"*.

Seguindo a proposta definida acima, obtemos a explicação "ideal" para o filme *"On Her Majesty's Secret Service"*. Nesse exemplo, *"Drama"* é o atributo mais relevante, sendo forte e negativamente correlacionada com a avaliação predita. Isso pode ser explicado pelo fato de

o usuário não gostar desse gênero, já que todos os 17 filmes *"Drama"* receberam avaliações baixas. Por outro lado, *"Ação"* e *"Crime"* aparecem em seguida e são os gêneros favoritos do usuário, sendo *"Ação"* o único atributo presente nesse filme. Por fim, o atributo menos relevante é *"Comédia"*, o que pode ser explicado por ser o gênero menos frequente. Vale ressaltar que os pesos da explicação são normalizados de forma que a magnitude do peso associado ao atributo mais relevante seja igual a 1. Essa decisão foi tomada para facilitar a comparação de diferentes explicações.

Métricas

Para avaliar a qualidade das explicações, escolhemos duas métricas: *recall* e distância euclidiana. O *recall* é uma métrica clássica da área de Recuperação de Informação (em inglês, Information Retrieval (IR)). Normalmente, em SR, o *recall* é utilizado como métrica de relevância de uma lista de recomendações, especificamente, para medir se o recomendador está de fato recuperando itens relevantes para os usuários. Os resultados da avaliação de uma lista de recomendação (ou de um modelo de classificação) podem ser agrupados em uma tabela, chamada de matriz de confusão, por meio da qual é possível computar diferentes métricas. A Tabela 5.2 descreve a matriz de confusão. Note que o *recall*, definido na Equação 5.7, é a fração de itens considerados relevantes sobre o total de itens relevantes.

Neste experimento, o *recall* é usado para avaliar explicações, comparando os conjuntos dos atributos presentes nas explicações providas pelo LIME-RS e nas explicações do *ground truth*. Um valor de 0,5 para o *recall* significa que o LIME-RS foi capaz de recuperar metade dos atributos mais relevantes do *ground truth*, a explicação "ideal" para uma dada instância alvo. Aliás, o *recall* é usado dessa mesma maneira nos experimentos offline do artigo do LIME original, bem como, nos experimentos da Seção 4.2.4.

Tabela 5.2: Matriz de confusão: tabela que agrupa os valores de verdadeiros positivos (tp), verdadeiros negativos (tf), falsos positivos (fp) e falsos negativos(fn).

	Relevante	**Irrelevante**	**Total**
Classificado como Relevante	tp	fp	tp+fp
Classificado como Irrelevante	fn	tn	fn +tn
Total	tp+fn	fp+tn	N

$$recall = \frac{tp}{tp + fn} \tag{5.7}$$

A distância euclidiana, definida na Equação 5.8, é utilizada para medir a proximidade entre dois vetores. Note que as explicações do *ground truth* e as providas pelo LIME-RS podem ser representadas por vetores, onde cada peso do vetor (valores reais) representa a força de um dado atributo no item da instância alvo. Note que essa métrica, ao comparar vetores de valores reais, considera tanto os sinais quanto a magnitude dos pesos dos atributos. Sendo assim, essa métrica serve como um complemento ao *recall*, uma vez que este é computado com conjuntos, formados considerando apenas a magnitude dos pesos.

$$euclidean\_distance = d(p, q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{5.8}$$

Reprodutibilidade dos Experimentos

Para fins de reprodutibilidade dos experimentos, foram escolhidas base de dados públicas e disponíveis online (vide Seção 5.3.1), bem como as seguintes bibliotecas de código aberto: Microsoft Recommenders[3] e LIME[4]. O código dos algoritmos e da avaliação experimental está disponível online[5].

Vale salientar que o foco destes experimentos é avaliar as explicações das recomendações e não as recomendações propriamente ditas. Mesmo assim, foi feita uma busca para encontrar o melhor ajuste e configuração dos parâmetros para os recomendadores. Utilizamos o pacote *Hyperdrive*[6] da AzureML também da Microsoft para realizar um *grid-search* dos hiperparâmetros usando a sua infra-estrutura de nuvem. A Tabela 5.3 descreve os valores ótimos encontrados para o modelo de recomendação Wide&Deep.

Por fim, a Tabela 5.4 detalha os resultados de acurácia. Os resultados de acurácia estão compatíveis, em termos de RMSE, com os benchmarks conhecidos para essa base de dados (HUG, 2020). Além disso, a Figura 5.5 mostra as curvas do RMSE nos dados de treino e teste ao longo das iterações de treinamento. É possível observar que, após 15.000 iterações,

[3]https://github.com/microsoft/recommenders

[4]https://github.com/marcotcr/lime

[5]https://github.com/caionobrega/movielens-explanations

[6]https://docs.microsoft.com/en-us/python/api/azureml-train-core/azureml.train.hyperdrive?view=azure-ml-py

o modelo converge.

Tabela 5.3: Configuração de hiperpârametros dos recomendadores

Model de Recomendação	Configuração
Wide&Deep	"linear_optimizer": "ftrl", "linear_optimizer_lr": 0.09727, "linear_l1_reg": 0.25378, "linear_l2_reg": 0.29009, "linear_momentum": 0.02058, "dnn_layer_sizes": [64, 64], "dnn_optimizer": "adagrad", "dnn_optimizer_lr": 0.08148, "dnn_l1_reg": 0.91548, "dnn_l2_reg": 0.93721, "dnn_momentum": 0.03003, "dnn_user_dim": 16, "dnn_item_dim": 8, "crossed_feat_dim": 1000, "dnn_dropout": 0.30855, "dnn_batch_norm": 0, "batch_size": 64, "steps": 50000

5.3.4 Resultados

Para responder a questão de pesquisa *QP2*, executamos dois experimentos. Primeiramente, avaliamos o desempenho de cada estratégia de vizinhança, descritas na Seção 5.2, quando comparadas com a nossa proposta de explicação ideal, elaborada em 5.3.3. Em um segundo momento, investigamos o efeito da alteração da função de proximidade $\pi_x(z)$ presente no LIME.

Para ambos experimentos, foram geradas explicações para 500 instâncias alvo selecionadas aleatoriamente. Acreditamos que é uma amostra com tamanho suficientemente grande

Tabela 5.4: Resultados de acurácia

Modelo Recomendador	Métrica	Resultado
Wide&Deep	Precision@10	0,07
	Recall@10	0,02
	RMSE	Treino: 0,87 - Teste: 0,95

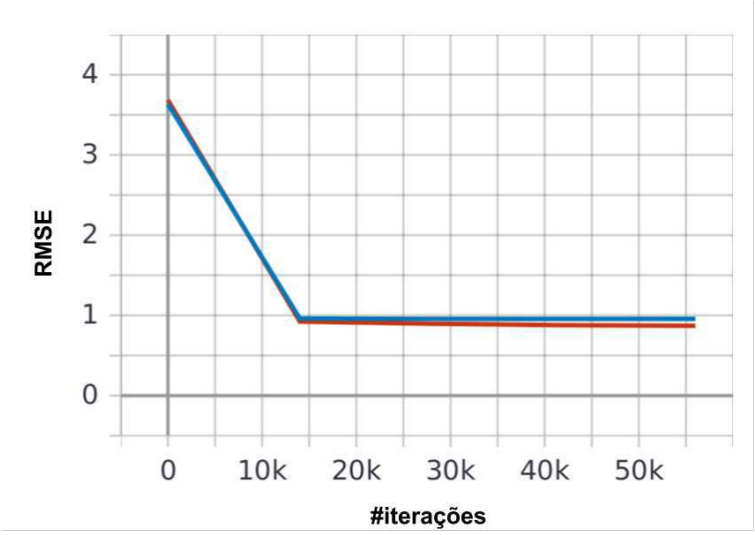


Figura 5.5: Curvas do RMSE nos dados de treino e teste ao longo das iterações de treinamento. É possível observar que, após 15.000 iterações, o modelo converge.

e diverso para obter resultados significativos. Além disso, consideramos os top-10 atributos mais relevantes.

Outra decisão metodológica envolve o tamanho das vizinhanças geradas. No LIME original, essa quantidade é definida por um número fixo arbitrário, por exemplo, 1.000 ou 5.000 itens sintéticos criados. Nos experimentos do Capítulo 4, também foi seguida essa lógica ao criar vizinhanças de mesmo tamanho, independemente da instância alvo. Contudo, dado o objetivo dos experimentos deste capítulo, foi observado que definir um número fixo para quaisquer instâncias alvo seria inviável. Isso ocorre devido à estratégia *"user-predicted_ratings"*, que, por definição, tem um tamanho pré-estabelecido de vizinhança. Sendo assim, visando realizar uma comparação justa, todas as demais estratégias devem produzir vizinhanças com o mesmo tamanho que o da *"user-predicted_ratings"*, ou seja, todas as vizinhanças têm o tamanho do histórico do usuário alvo. Dessa forma, todas as estratégias acabam adotando uma segunda intuição de localidade que é justamente o tamanho do histórico do usuário.

Vale lembrar que a função de proximidade $\pi_x(z)$, neste caso, a similaridade do cosseno, também é aplicada em todos os casos.

Análise da Qualidade das Explicações do LIME-RS considerando Novas Estratégias de Localidade

Como dito acima, os experimentos desta seção visam analisar a qualidade das explicações, levando em consideração a nossa proposta de explicação ideal.

A Figura 5.6 mostra a média com o intervalo de confiança (*alpha = 5%*) para as duas métricas: *recall* e distância euclidiana. Vale destacar que a distância euclidiana está normalizada para que os valores fiquem entre 0 e 1, e assim a comparação seja direta.

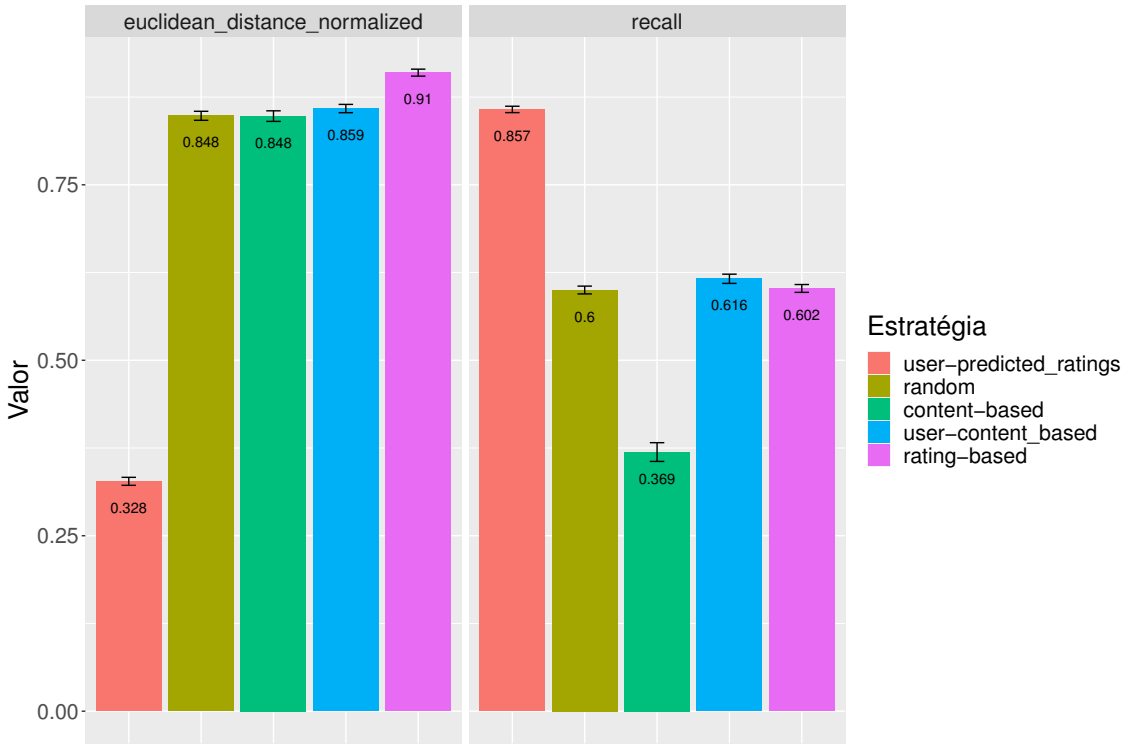


Figura 5.6: Avaliação offline da qualidade das explicações para diferentes estratégias de vizinhança aplicadas ao LIME-RS. Instâncias alvo selecionadas aleatoriamente.

Os resultados mostram que, para ambas as métricas, a estratégia *"user-predicted_ratings"* é a que apresenta melhor desempenho, sendo bem superior que as demais estratégias. Isso pode ser explicado pelo fato de que os itens que compõem a vizinhança são exclusivamente os itens do histórico do usuário da instância alvo, o que indica que a intuição da utilização do histórico do usuário parece ser a opção mais promissora para capturar a noção de localidade.

O desempenho da estratégia *"rating-based"* mostrou-se não ser adequado para capturar os atributos relevantes do *ground truth*. De fato, essa estratégia, juntamente com a *"ran-*

dom", é a que, teoricamente, pode selecionar itens mais diversos no que diz respeito aos atributos dos itens. Logo, dada a nossa definição de *ground truth*, era esperado que essas estratégias gerassem explicações mais distantes da explicação "ideal". Em termos de distância euclidiana, a *"rating-based"* apresentou o pior desempenho, enquanto que, em termos de *recall* está mais próxima das demais estratégias.

A estratégia *"random"* está com desempenho similar às estratégias *"user-content_based"* e *"content_based"*, de forma que não há diferença estatisticamente significativa entre elas, pelo menos no que se refere à distância euclidiana. Intuitivamente, esperava-se que a estratégia *random*, por ser a mais trivial e ingênua, tivesse o pior desempenho. Por outro lado, era esperado que as estratégias inspiradas na abordagem baseada em conteúdo tivessem desempenho melhor, uma vez que o *ground truth* também é inspirado nessa abordagem. Uma hipótese para justificar esse resultado pode estar relacionada aos históricos dos usuários, pois estes geralmente apresentam uma variância nas avaliações, especialmente para gêneros populares. Por exemplo, suponha que um dado usuário consumiu cem filmes de *"Drama"*, porém avaliou positivamente apenas metade deles. Essa variância pode indicar que há outros fatores, além do gênero do filme, envolvidos nas preferências desse usuário, pois, do contrário, haveria uma maior quantidade de avaliações positivas. Sendo assim, nesses casos, essas abordagens podem falhar em capturar esses fatores. Outro motivo pode estar relacionado à representação interpretável das instâncias (vide Seção 4.2.1), pois, ao considerar apenas instâncias do mesmo gênero, a diferença entre elas se resume ao item em si. Em outras palavras, as instâncias $z \in Z$ são equidistantes à instância alvo, fazendo com que a função de de proximidade $\pi_x(z)$ seja irrelevante. Dessa forma, o LIME-RS aprende que os itens das instâncias $z \in Z$ são os atributos mais importantes.

Nos resultados discutidos acima, as instâncias alvo foram selecionadas aleatoriamente. Contudo, é sabido que o modelo interpretável do LIME-RS é treinado baseado nas predições providas pelo recomendador subjacente. Sendo assim, aplicamos o LIME-RS, com as diferentes estratégias de vizinhança, para explicar instâncias alvo, cujas predições foram mais acuradas. Em outras palavras, queremos investigar a relação entre a qualidade do recomendador, medida pelo erro nas predições de avaliações, e as explicações providas pelo LIME-RS. Para avaliar essa questão, foram selecionadas outras 500 instâncias alvo, considerando tanto o erro de predição no item quanto o RMSE do usuário computado nos dados de

treino do seu histórico. A Figura 5.7 mostra os resultados desse novo experimento. É possível perceber que há um impacto positivo, porém discreto, em todas as estratégias. Podemos destacar a melhoria, em termos de *recall*, para as abordagens baseadas em conteúdo. Isso pode indicar que, para os usuários dessas instâncias, o recomendador aprendeu melhor as suas preferências quanto aos atributos. Além disso, a estratégia baseada exclusivamente no histórico ainda é a que apresentou melhor resultado. Por fim, de forma geral, esse resultado indica há uma correlação entre o desempenho do recomendador e as explicações geradas.

Vale lembrar que a função de proximidade $\pi_x(z)$ é aplicada em todas as estratégias nos dois experimentos e pode estar afetando de alguma forma os resultados. Na Seção 5.3.4, discutimos mais sobre esse efeito.

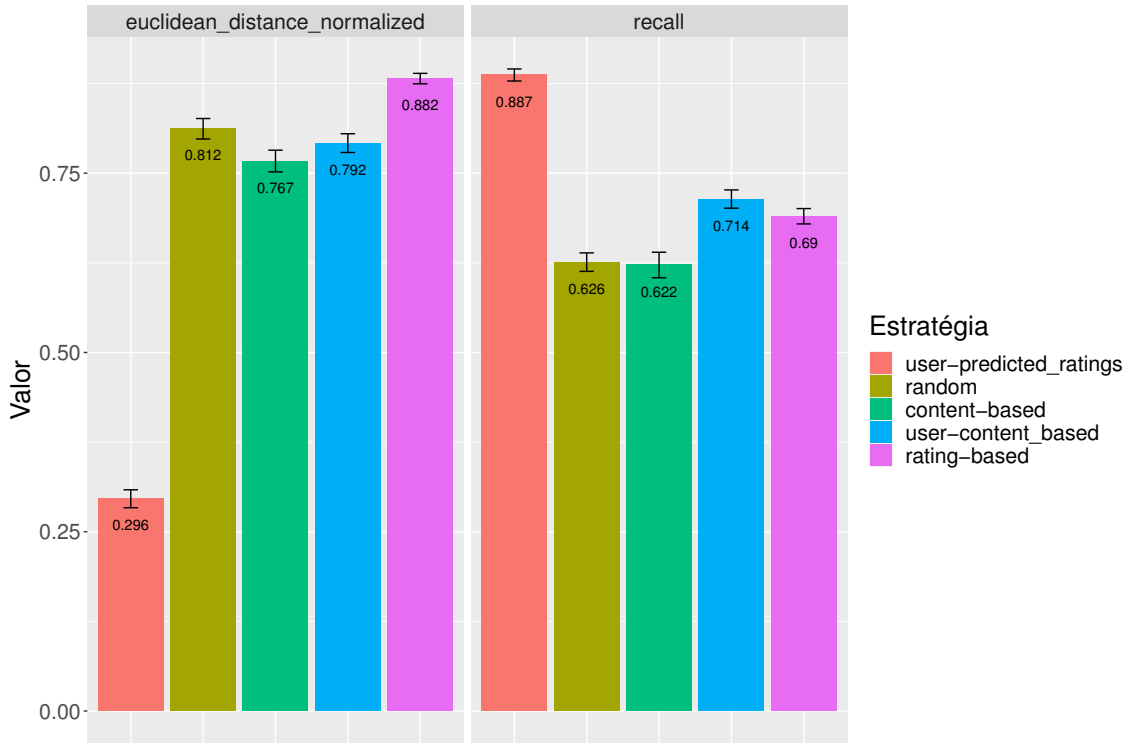


Figura 5.7: Avaliação offline da qualidade das explicações para diferentes estratégias de vizinhança aplicadas ao LIME-RS. Instâncias alvo selecionadas baseado no erro de predição do modelo de recomendação.

A seguir, ilustramos as explicações geradas pelas diferentes estratégias de vizinhança em dois exemplos. Além disso, incluímos as explicações obtidas ao treinar o LIME-RS com as predições fornecidas por um recomendador subjacente mal treinado. A ideia é também mostrar em exemplos a relação entre a qualidade do modelo de recomendação e as explicações. Nesse caso, o modelo de recomendação é também o Wide&Deep com a mesma configura-

ção da Tabela 5.3, porém treinado por apenas 50 épocas. O RMSE nos dados de teste desse modelo é 1,3.

A Figura 5.8 mostra as explicações para o usuário fictício, descrito na Seção 5.3.3. Primeiramente, considere as explicações geradas pelo LIME-RS treinado com um recomendador acurado (coluna a). Note que a explicação gerada pela estratégia *user-predicted_ratings* é a que mais se aproxima da explicação "ideal", com atributos seguindo a mesma ordem de importância. As estratégias baseadas em conteúdo são, como esperado, similares entre si. Elas conseguem capturar a perspectiva do usuário com relação ao gênero *"Drama"*, porém não capturam para o gênero *"Ação"*. Já as estratégias *random* e *rating_based* capturam o gênero *"Ação"* mas não atribuem grande importância a esse gênero. O gênero *"Thriller"* aparece nas explicações porque alguns itens selecionados na composição da vizinhança tem como atributos tanto *"Ação"* quanto *"Thriller"*. A sua relevância pode indicar que o recomendador aprendeu a relação entre esses gêneros. Já as explicações obtidas com o recomendador mal treinado (coluna b) são bastante diferentes quando comparadas às respectivas explicações geradas com a versão mais acurada do recomendador. A exceção é a estratégia *user-predicted_ratings*. Isso pode ser explicado pelo fato do perfil desse usuário fictício ter um padrão bem definido.

A Figura 5.9 mostra um exemplo para um usuário real. Dessa vez, o usuário alvo tem um perfil mais diverso, porém com perfil mais voltado para drama e comédia. A explicação "ideal" para o filme *"Armageddon"* considera três dos gêneros do filme, *"Sci Fi"*, *"Aventura"*, *"Thriller"*, como atributos entre os top-6 mais relevantes. A Figura 5.10 mostra as explicações geradas pelas diferentes estratégias para esse exemplo. Primeiramente, considere as explicações geradas pelo LIME-RS treinado com um recomendador acurado (coluna a). Note que a explicação gerada pela estratégia *user-predicted_ratings* mostra que *"Ação"* está forte e negativamente relacionado ao filme. Isso faz sentido porque o usuário geralmente não gosta de filmes desse gênero. As estratégias baseada em conteúdo dessa vez são menos similares entre si. Apesar de atribuírem o mesmo sinal, a ordem de importância varia bastante. Isso pode ser explicado pela diversidade da vizinhança, uma vez que o item apresenta quatro gêneros. Por fim, as estratégias *random* e *rating_based* apresentam explicações mais aleatórias, sem muita ligação com a perspectiva do usuário ou do item. Já as explicações obtidas com o recomendador mal treinado (coluna b) são ainda mais distantes da explicação

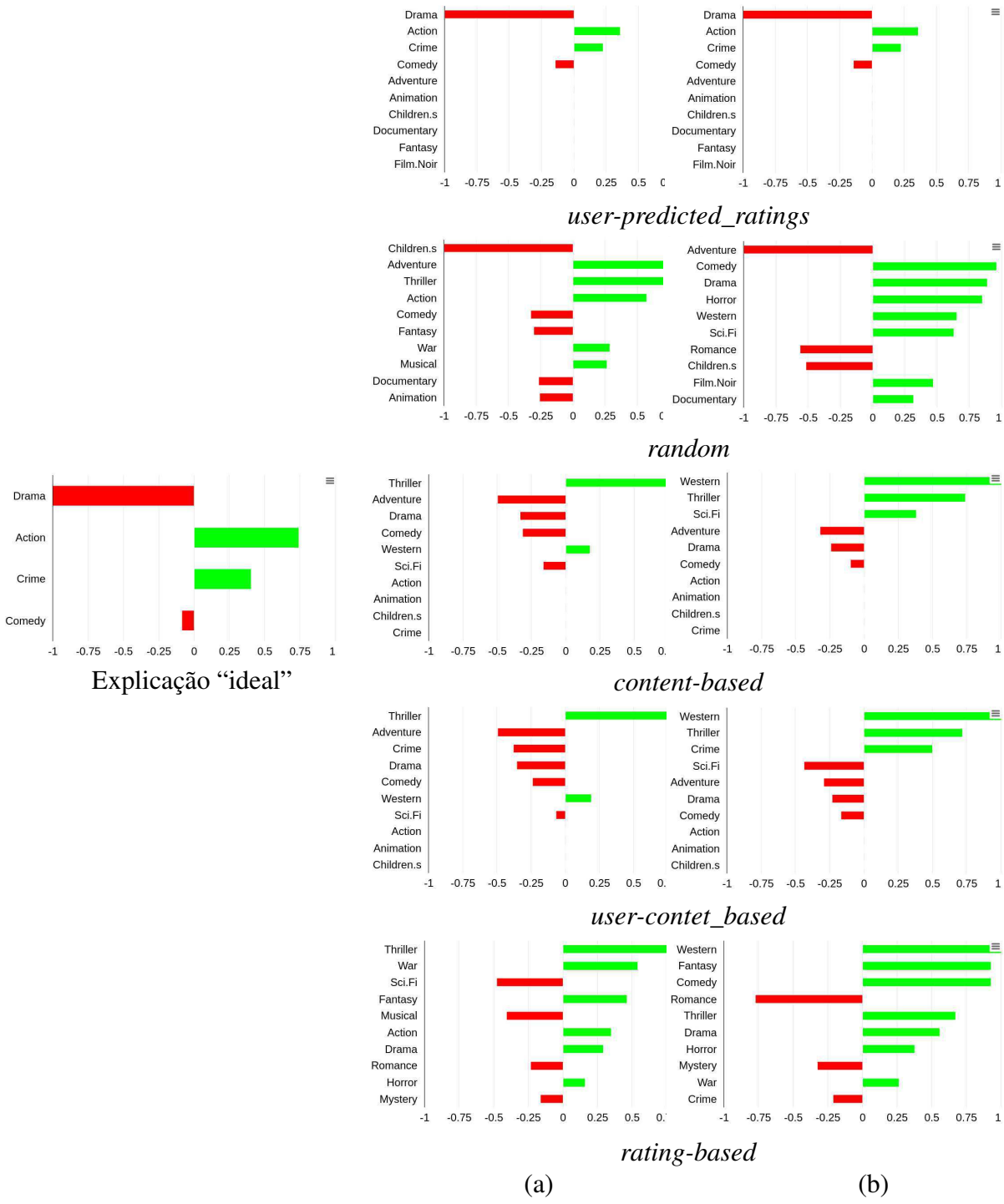


Figura 5.8: Exemplo de explicações geradas por diferentes estratégias de vizinhança aplicadas ao LIME-RS. A instância alvo é a mesma da Figura 5.4, composta por um usuário fictício e o filme *"On Her Majesty's Secret Service"*. As explicações foram extraídas de um recomendador mais acurado (coluna a) e um menos acurado b).

"ideal", inclusive a estratégia *user-predicted_ratings*. Esse resultado mostra que o fato de o recomendador estar mal treinado impactou ainda mais a qualidade das explicações.

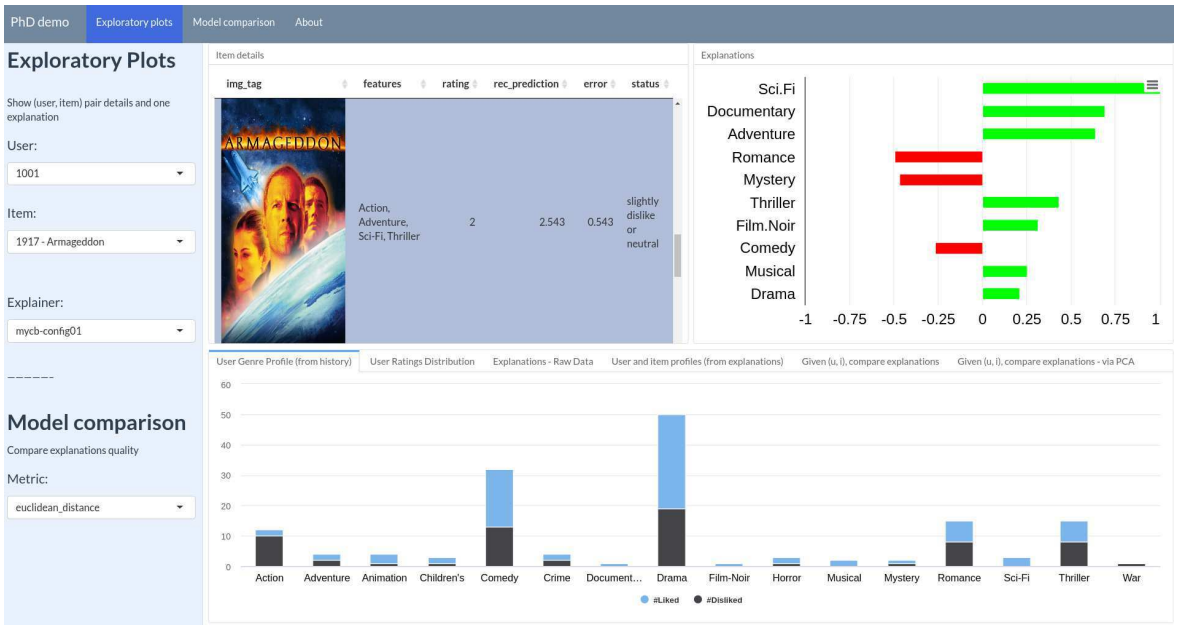


Figura 5.9: Exemplo de uma explicação "ideal" para um usuário real. Figura extraída de uma ferramenta desenvolvida nesta tese para realizar experimentos e analisar resultados.

A questão de pesquisa 2 (QP2) que estes experimentos visam investigar é: *"Qual é o impacto da formação da vizinhança no método LIME na qualidade de suas explicações?"*. Os resultados mostraram que a estratégia de vizinhança baseada nos histórico do usuário é a mais promissora para gerar explicações usando o LIME-RS. Também vale destacar que as explicações das demais estratégias podem variar bastante e, geralmente, são mais distantes da proposta de explicação "ideal". Além disso, a qualidade do modelo de recomendação também tem um papel importante, pois, se o modelo não for bom, as explicações tendem a ser ruins, independentemente da estratégia de vizinhança adotada.

Análise da Qualidade das Explicações do LIME-RS considerando Diferentes Funções de Proximidade

Como discutido anteriormente, um fator importante na ideia de localidade do LIME original é a função de proximidade $\pi_x(z)$, pois por meio dela as instâncias da vizinhança são ponderadas no treinamento do modelo local interpretável.

Essa questão foi brevemente discutida no trabalho de (LAUGEL et al., 2018). Os autores mostraram que o LIME é sensível à configuração da função de proximidade $\pi_x(z)$, o que pode impactar negativamente as suas explicações. A Figura 5.2 ilustra essa discussão, onde

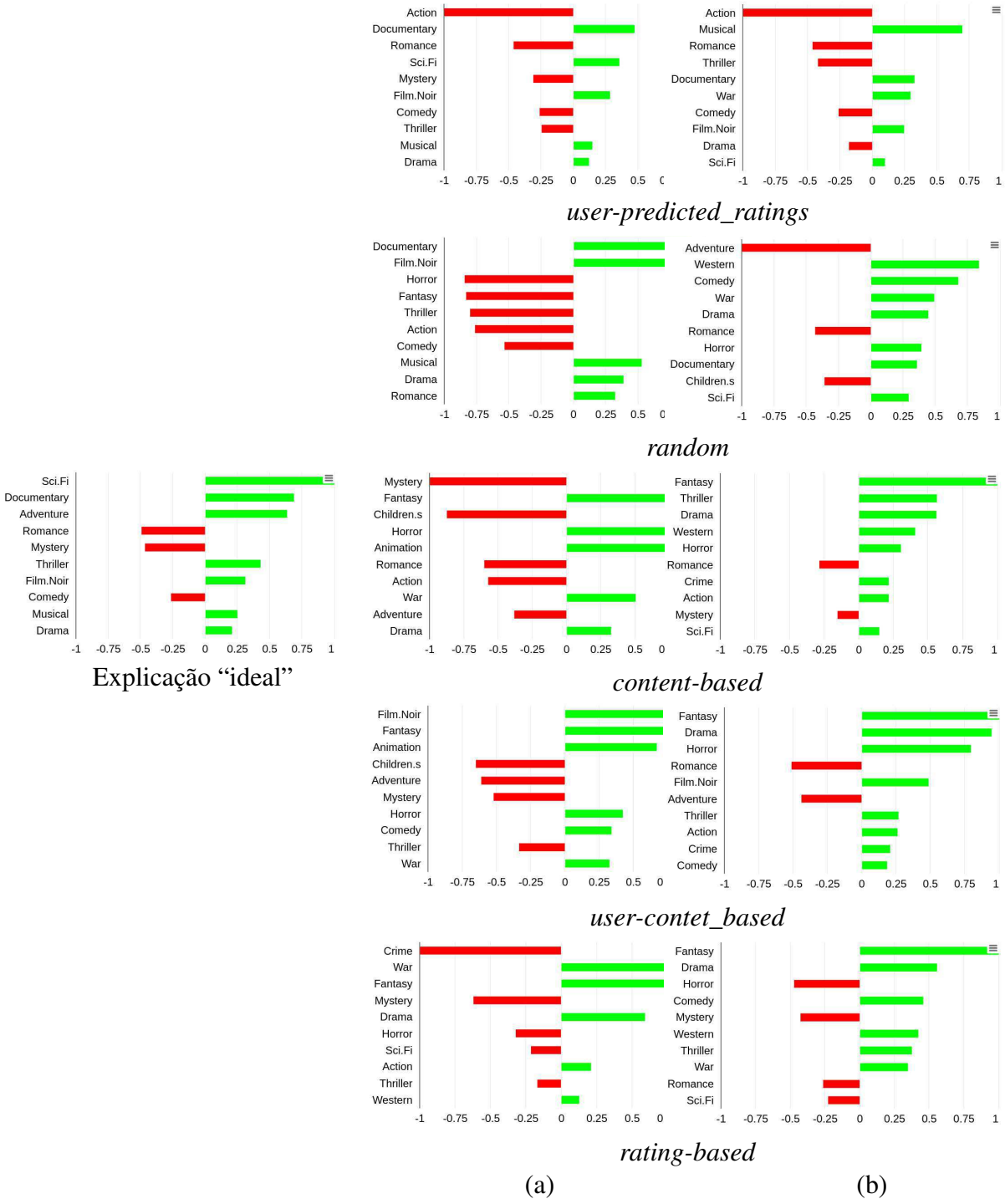


Figura 5.10: Exemplo de explicações geradas por diferentes estratégias de vizinhança aplicadas ao LIME-RS. A instância alvo é a mesma da Figura 5.9, composta por um usuário real e o filme *"Armageddon"*. As explicações foram extraídas de um recomendador mais acurado (coluna a) e um menos acurado (coluna b).

o LIME com configuração diferente para $\pi_x(z)$ (curva azul), é melhor que o LIME original (curva verde) para aproximar o comportamento local do modelo subjacente. Sendo assim,

investigar essa direção é um complemento à questão de pesquisa 2.

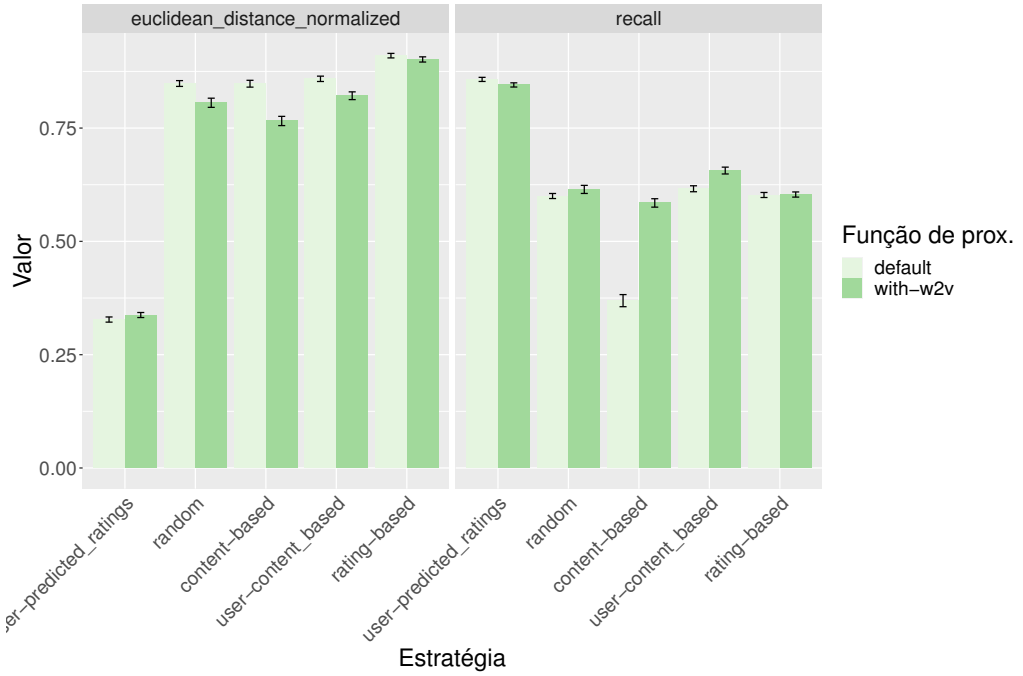


Figura 5.11: Avaliação offline da qualidade das explicações para diferentes estratégias de vizinhança aplicadas ao LIME-RS e diferentes funções de proximidade.

Aqui, comparamos duas funções de proximidade: a abordagem *default* com a abordagem *with-w2v*. A abordagem *default* é que aplica a função de proximidade do LIME original, ou seja, distância do cosseno baseada na representação *multi-hot* da instância alvo. Já a abordagem *with-w2v* também está baseada na distância do cosseno, porém utilizando a representação Word2Vec adaptada para sistemas de recomendação (OZSOY, 2016). A ideia é que, ao utilizar representações mais sofisticadas para os itens, tais como os *embeddings* providos pelo algoritmo Word2Vec, seja possível achar a relação mais complexa entre os itens. Por exemplo, ao utilizar a representação *multi-hot*, se uma dada perturbação for do gênero *"Animação"* e o item da instância for *"Ação"*, a distância entre eles será máxima e, portanto, essa perturbação teria pouca relevância no treinamento do modelo local interpretável. Já com a representação Word2Vec desses mesmos itens, seria possível obter uma relação implícita, por exemplo, ambos os filmes têm o mesmo diretor, ou foram lançados na mesma época. Dessa forma, essa perturbação teria alguma relevância no treinamento do modelo local interpretável.

A Figura 5.11 detalha os resultados da comparação entre a função de proximidade *de-*

fault e *with-w2v*. Observe que, excetuando-se a estratégia *"user-predicted_ratings"*, onde os resultados não são significativamente diferentes, há uma discreta melhora, porém significativa nas estratégias *"user-content_based"* , *"random"* e *"content_based"*. Destaque para a *"content_based"* que, em termos de *recall*, melhorou seu desempenho em aproximadamente 60%.

Esses resultados indicam que investigar a função de proximidade é um caminho promissor para obter explicações mais fiéis ao recomendador subjacente.

5.4 Considerações Finais

Neste capítulo, expandimos a proposta LIME-RS, visando investigar novas e diferentes maneiras de geração da vizinhança local, passo fundamental do LIME e do LIME-RS.

As abordagens para geração de uma vizinhança local para uma dada instância alvo foram inspiradas nas abordagens de recomendação baseada em filtragem colaborativa e baseada em conteúdo. Entre as abordagens propostas, a abordagem que leva em consideração somente os itens do histórico do usuário alvo, foi a que apresentou os resultados mais fiéis à nossa proposta de explicação ideal. A propósito, a definição proposta para explicação ideal também é uma contribuição deste trabalho. Também verificamos que há uma correlação entre a qualidade do modelo de recomendação e das explicações geradas pelo LIME-RS, de forma que, não importa a estratégia de vizinhança, se o modelo for mal treinado, isto é com predições ruins, as explicações também serão ruins.

Por fim, também analisamos o impacto que a função de proximidade tem no LIME-RS. A função de proximidade é uma forma de ponderar a importância das instâncias que compoẽm a vizinhança local para com a instância alvo. Os resultados indicaram que utilizar métodos para extração de representações mais sofisticadas e associá-las à função de proximidade pode melhorar o desempenho do LIME-RS no que diz respeito à qualidade das explicações.

Capítulo 6

Conclusões e Trabalhos Futuros

Neste capítulo, sumarizamos os principais achados desta tese e apresentamos as nossas conclusões. Também listamos o que consideramos ser os pontos a serem atacados como trabalhos futuros desta pesquisa.

6.1 Conclusões

Explicar predições de modelos de aprendizado de máquina tem se tornado uma questão cada vez mais sensível, dada a ampla adoção para resolver as mais variadas tarefas. Neste trabalho, lidamos com o problema de gerar explicações para uma recomendação qualquer, visando promover interpretabilidade a sistemas de recomendação sabidamente complexos e de difícil interpretação.

Para atacar tal problema, investigamos a aplicação de métodos agnósticos ao modelo, os quais separam o processo de geração de explicações do modelo subjacente que se deseja interpretar. Especificamente, identificamos as características intrínsecas que impediam o LIME (RIBEIRO; SINGH; GUESTRIN, 2016b) de ser aplicado diretamente aos modelos de sistemas de recomendação e propusemos alternativas que tornassem isso possível.

Também comparamos nossa adaptação do LIME com o método agnóstico ao modelo proposto por Peake & Wang (PEAKE; WANG, 2018). Primeiro estudamos em detalhes as diferenças e relações entre essas abordagens. Em seguida, investigamos o grau em que ambas as abordagens geram as mesmas explicações para os mesmos itens previstos. Observamos que, em geral, esses modelos são muito semelhantes entre si, onde a principal diferença é

como a noção de localidade é garantida.

Inspirados por trabalhos relacionados, também investigamos o problema da geração da vizinhança em métodos *post-hoc interpretability*, cuja abordagem é baseada em treinar modelos locais interpretáveis. Propomos quatro diferentes abordagens para geração de explicações e uma ideia de explicação "ideal", que busca capturar tanto a perspectiva do usuário quanto a do item. Os resultados mostraram que a vizinhança baseada exclusivamente no histórico do usuário é a que gerou explicações que melhor refletiam a combinação entre perfil do usuário e item que se deseja explicar. Por outro lado, as demais abordagens não foram superiores à abordagem aleatória, baseline do experimento. As possíveis razões para essas limitações também foram discutidas. Ainda, verificamos que a qualidade do modelo de recomendação pode impactar a qualidade das explicações, de forma que um modelo não treinado tende a gerar explicações ruins. Por fim, também foi investigado o impacto que a função de proximidade usada para ponderar a importância das instâncias da vizinhança local em métodos baseados em modelos locais interpretáveis. Especificamente, aplicamos a técnica Word2Vec para obter representações abstratas das instâncias da vizinhança local e usá-las na computação da função de proximidade. Foi observado que houve uma discreta melhora na qualidade das explicações quando comparadas com a versão padrão de aplicação da função de proximidade, o que indica uma direção de pesquisa a ser explorada no futuro.

Por fim, é importante destacar algumas limitações deste trabalho. Os experimentos foram executados em uma única base de dados, MovieLens, cujo domínio é avaliações de filmes. Apesar de ser bastante utilizada em trabalhos relacionados, o ideal seria expandir os experimentos para outras bases de dados. Outra limitação diz respeito à avaliação experimental. Neste trabalho, realizamos uma avaliação offline com o objetivo de analisar a qualidade das explicações. Vale destacar que conduzimos essa avaliação offline, seguindo a mesma metodologia de trabalhos relacionados. Contudo, uma avaliação mais completa exige estudos com humanos para analisar o efeito prático das explicações. Na verdade, chegamos a realizar experimentos nesse sentido usando a própria plataforma do MovieLens. Na primeira versão desse experimento, houve um número pequeno de usuários, o que não permitiu tirar conclusões confiáveis. Devido ao tempo, não conseguimos executar uma segunda versão com participação de mais usuários. O apêndice A descreve, resumidamente, como esse estudo foi realizado.

6.2 Trabalhos Futuros

Nesta seção, descrevemos as próximas atividades que podem ser realizadas a fim de generalizar e expandir os achados e contribuições desta teste. Os caminhos que vislumbramos como futuro da pesquisa estão listados abaixo:

- Adicionar mais modelos de recomendação e bases de dados aos experimentos. Neste trabalho, utilizamos um representante de modelos baseado em Fatoração de Matrizes e um representante de modelos baseado em Deep Learning. Contudo, para generalizar os resultados e tornar as evidências mais fortes é necessário expandir os experimentos nesse sentido;

- Analisar o impacto de diferentes representações na computação da função de proximidade. Na Seção 5.3.4, foi discutido que os passos que envolvem a aplicação da função de proximidade pode ser uma vertente para melhorar a qualidade das explicações do LIME-RS. Aqui, podemos explorar outras funções e representações visando capturar padrões mais complexos e melhorar o ajuste do modelo local interpretável;

- Investigar a aplicabilidade de métodos de interpretabilidade *post hoc* para geração de explicações globais para modelos de recomendação. Essa tarefa é desafiadora, pois os modelos de recomendação geralmente são personalizados e, portanto, extrair um entendimento global não é trivial;

- Adaptar o LIME-RS, ou outros métodos de interpretabilidade *post hoc* para a tarefa de ranking, ou seja, gerar explicações para listas de recomendação. Esse ponto seria um complemento da pesquisa, pois este trabalho aborda a geração de explicações para instâncias individuais e, não, para conjuntos ou listas;

- Realizar mais experimentos com usuários. Dado que um dos objetivos da pesquisa é avaliar a qualidade das explicações, então o mais adequado é complementar os achados deste trabalho por meio de uma avaliação com usuários, uma vez que interpretar explicações é uma atividade subjetiva. De fato, realizamos um experimento com usuários, porém este foi uma versão preliminar com pequena participação de usuários, o que não permitiu tirar conclusões confiáveis.

- Realizar experimentos para analisar causalidade. Utilizar ou adaptar métodos de causalidade presentes na literatura e compará-los com os métodos de interpretabilidade *post-hoc*. Os resultados desses experimentos podem esclarecer em quais situações os métodos *post-hoc* capturam causalidade de forma mais efetiva.

Referências Bibliográficas

ABDOLLAHI, B.; NASRAOUI, O. Explainable matrix factorization for collaborative filtering. In: **Proceedings of the 25th International Conference Companion on World Wide Web**. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2016. (WWW '16 Companion), p. 5–6. ISBN 978-1-4503-4144-8. Disponível em: <https://doi.org/10.1145/2872518.2889405>.

ABDOLLAHI, B.; NASRAOUI, O. Explainable restricted boltzmann machines for collaborative filtering. **CoRR**, abs/1606.07129, 2016. Disponível em: <http://dblp.uni-trier.de/db/journals/corr/corr1606.html\#AbdollahiN16>.

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. **IEEE Trans. on Knowl. and Data Eng.**, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 17, n. 6, p. 734–749, jun. 2005. ISSN 1041-4347. Disponível em: <http://dx.doi.org/10.1109/TKDE.2005.99>.

AGGARWAL, C. C. **Recommender Systems: The Textbook**. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2016. ISBN 3319296574.

AGRAWAL, R.; IMIELIńSKI, T.; SWAMI, A. Mining association rules between sets of items in large databases. **SIGMOD Rec.**, Association for Computing Machinery, New York, NY, USA, v. 22, n. 2, p. 207–216, jun. 1993. ISSN 0163-5808. Disponível em: <https://doi.org/10.1145/170036.170072>.

ALEKSANDROVA, M.; BRUN, A.; BOYER, A.; CHERTOV, O. Identifying representative users in matrix factorization-based recommender systems: Application to solving the content-less new item cold-start problem. **J. Intell. Inf. Syst.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 48, n. 2, p. 365–397, abr. 2017. ISSN 0925-9902. Disponível em: <https://doi.org/10.1007/s10844-016-0418-3>.

ALVAREZ-MELIS, D.; JAAKKOLA, T. S. On the robustness of interpretability methods. **CoRR**, abs/1806.08049, 2018. Disponível em: <http://arxiv.org/abs/1806.08049>.

BAEZA-YATES, R. A.; RIBEIRO-NETO, B. **Modern Information Retrieval**. USA: Addison-Wesley Longman Publishing Co., Inc., 1999. ISBN 020139829X.

BALABANOVIć, M.; SHOHAM, Y. Fab: Content-based, collaborative recommendation. **Commun. ACM**, ACM, New York, NY, USA, v. 40, n. 3, p. 66–72, mar. 1997. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/245108.245124>.

Barredo Arrieta, A.; DíAZ-RODRíGUEZ, N.; Del Ser, J.; BENNETOT, A.; TABIK, S.; BARBADO, A.; GARCIA, S.; GIL-LOPEZ, S.; MOLINA, D.; BENJAMINS, R.; CHATILA, R.; HERRERA, F. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. **Information Fusion**, v. 58, p. 82 – 115, 2020. ISSN 1566-2535. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1566253519308103>.

BASILICO, J.; HOFMANN, T. Unifying collaborative and content-based filtering. In: **Proceedings of the Twenty-first International Conference on Machine Learning**. New York, NY, USA: ACM, 2004. (ICML '04), p. 9–. ISBN 1-58113-838-5. Disponível em: <http://doi.acm.org/10.1145/1015330.1015394>.

BILGIC, M.; MOONEY, R. Explaining recommendations: Satisfaction vs. promotion. In: **Proceedings of Beyond Personalization 2005: A Workshop on the Next Stage of Recommender Systems Research at the 2005 International Conference on Intelligent User Interfaces**. [s.n.], 2005. Disponível em: <http://www.cs.iit.edu/~ml/pdfs/bilgic-iui05-wkshp.pdf>.

BOSTANDJIEV, S.; O'DONOVAN, J.; HöLLERER, T. Tasteweights: A visual interactive hybrid recommender system. In: **Proceedings of the Sixth ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2012. (RecSys '12), p. 35–42. ISBN 9781450312707. Disponível em: <https://doi.org/10.1145/2365952.2365964>.

BREIMAN, L. Random forests. **Machine Learning**, v. 45, n. 1, p. 5–32, Oct 2001. ISSN 1573-0565. Disponível em: <https://doi.org/10.1023/A:1010933404324>.

BREIMAN, L.; FRIEDMAN, J. H.; OLSHEN, R. A.; STONE, C. J. **Classification and Regression Trees**. Monterey, CA: Wadsworth and Brooks, 1984.

BRUN, A.; ALEKSANDROVA, M.; BOYER, A. Can latent features be interpreted as users in matrix factorization-based recommender systems? In: **Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 02**. Washington, DC, USA: IEEE Computer Society, 2014. (WI-IAT '14), p. 226–233. ISBN 978-1-4799-4143-8. Disponível em: <http://dx.doi.org/10.1109/WI-IAT.2014.102>.

BUCHANAN, B. G.; SHORTLIFFE, E. H. (Ed.). **Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project**. Reading, MA: Addison-Wesley, 1985.

BUHRMESTER, V.; MÜNCH, D.; ARENS, M. Analysis of explainers of black box deep neural networks for computer vision: A survey. **CoRR**, abs/1911.12116, 2019. Disponível em: <http://arxiv.org/abs/1911.12116>.

BURKE, R. Hybrid web recommender systems. In: ____. **The Adaptive Web: Methods and Strategies of Web Personalization**. Berlin, Heidelberg: Springer-Verlag, 2007. p. 377–408. ISBN 9783540720782.

CARVALHO, D. V.; PEREIRA, E. M.; CARDOSO, J. S. Machine learning interpretability: A survey on methods and metrics. **Electronics (Switzerland)**, v. 8, n. 8, p. 1–34, 2019. ISSN 20799292.

CHANG, S.; HARPER, F. M.; TERVEEN, L. G. Crowd-based personalized natural language explanations for recommendations. In: **Proceedings of the 10th ACM Conference on Recommender Systems**. New York, NY, USA: ACM, 2016. (RecSys '16), p. 175–182. ISBN 978-1-4503-4035-9. Disponível em: <http://doi.acm.org/10.1145/2959100.2959153>.

CHEN, X.; CHEN, H.; XU, H.; ZHANG, Y.; CAO, Y.; QIN, Z.; ZHA, H. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In: **Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: Association for Computing Machinery, 2019. (SIGIR'19), p. 765–774. ISBN 9781450361729. Disponível em: <https://doi.org/10.1145/3331184.3331254>.

CHEN, X.; QIN, Z.; ZHANG, Y.; XU, T. Learning to rank features for recommendation over multiple categories. In: **Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: Association for Computing Machinery, 2016. (SIGIR '16), p. 305–314. ISBN 9781450340694. Disponível em: <https://doi.org/10.1145/2911451.2911549>.

CHEN, Z.; WANG, X.; XIE, X.; WU, T.; BU, G.; WANG, Y.; CHEN, E. Co-attentive multi-task learning for explainable recommendation. In: **Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19**. International Joint Conferences on Artificial Intelligence Organization, 2019. p. 2137–2143. Disponível em: <https://doi.org/10.24963/ijcai.2019/296>.

CHENG, H.-T.; KOC, L.; HARMSEN, J.; SHAKED, T.; CHANDRA, T.; ARADHYE, H.; ANDERSON, G.; CORRADO, G.; CHAI, W.; ISPIR, M.; ANIL, R.; HAQUE, Z.; HONG, L.; JAIN, V.; LIU, X.; SHAH, H. Wide amp; deep learning for recommender systems. In: **Proceedings of the 1st Workshop on Deep Learning for Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2016. (DLRS 2016), p. 7–10. ISBN 9781450347952. Disponível em: <https://doi.org/10.1145/2988450.2988454>.

COVINGTON, P.; ADAMS, J.; SARGIN, E. Deep neural networks for youtube recommendations. In: **Proceedings of the 10th ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2016. (RecSys '16), p. 191–198. ISBN 9781450340359. Disponível em: <https://doi.org/10.1145/2959100.2959190>.

DACREMA, M. F.; CREMONESI, P.; JANNACH, D. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In: **Proceedings of the 13th ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2019. (RecSys '19), p. 101–109. ISBN 9781450362436. Disponível em: <https://doi.org/10.1145/3298689.3347058>.

DAS, A. S.; DATAR, M.; GARG, A.; RAJARAM, S. Google news personalization: Scalable online collaborative filtering. In: **Proceedings of the 16th International Conference on**

World Wide Web. New York, NY, USA: ACM, 2007. (WWW '07), p. 271–280. ISBN 978-1-59593-654-7. Disponível em: <http://doi.acm.org/10.1145/1242572.1242610>.

DESHPANDE, M.; KARYPIS, G. Item-based top-n recommendation algorithms. **ACM Trans. Inf. Syst.**, ACM, New York, NY, USA, v. 22, n. 1, p. 143–177, jan. 2004. ISSN 1046-8188. Disponível em: <http://doi.acm.org/10.1145/963770.963776>.

DESROSIERS, C.; KARYPIS, G. A comprehensive survey of neighborhood-based recommendation methods. In: RICCI, F.; ROKACH, L.; SHAPIRA, B.; KANTOR, P. B. (Ed.). **Recommender Systems Handbook**. [S.l.]: Springer, 2011. p. 107–144. ISBN 978-0-387-85819-7.

DOSHI-VELEZ, F.; KIM, B. Towards a rigorous science of interpretable machine learning. 2017.

DOSHI-VELEZ, F.; KIM, B. Towards a rigorous science of interpretable machine learning. **arXiv**, 2017. Disponível em: <https://arxiv.org/abs/1702.08608>.

FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. **Ann. Statist.**, The Institute of Mathematical Statistics, v. 29, n. 5, p. 1189–1232, 10 2001. Disponível em: <https://doi.org/10.1214/aos/1013203451>.

FRIEDRICH, G.; ZANKER, M. M.: A taxonomy for generating explanations in recommender systems. **AI Magazine**, p. 90–98, 2011.

FUNK, S. **Netflix Update: Try This at Home.** Disponível em: <http://sifter.org/~simon/journal/20061211.html>. Último acesso em Abril de 2014.

GANTNER, Z.; DRUMOND, L.; FREUDENTHALER, C.; RENDLE, S.; SCHMIDT-THIEME, L. Learning attribute-to-feature mappings for cold-start recommendations. In: **Proceedings of the 2010 IEEE International Conference on Data Mining**. Washington, DC, USA: IEEE Computer Society, 2010. (ICDM '10), p. 176–185. ISBN 978-0-7695-4256-0. Disponível em: <http://dx.doi.org/10.1109/ICDM.2010.129>.

GEDIKLI, F.; JANNACH, D.; GE, M. How should i explain? a comparison of different explanation types for recommender systems. **Int. J. Hum.-Comput. Stud.**, Academic Press, Inc., Duluth, MN, USA, v. 72, n. 4, p. 367–382, abr. 2014. ISSN 1071-5819. Disponível em: <http://dx.doi.org/10.1016/j.ijhcs.2013.12.007>.

Gilpin, L. H.; Bau, D.; Yuan, B. Z.; Bajwa, A.; Specter, M.; Kagal, L. Explaining explanations: An overview of interpretability of machine learning. In: **2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)**. [S.l.: s.n.], 2018. p. 80–89.

GOLDBERG, D.; NICHOLS, D.; OKI, B. M.; TERRY, D. Using collaborative filtering to weave an information tapestry. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 35, n. 12, p. 61–70, dez. 1992. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/138859.138867>.

GOMEZ-URIBE, C. A.; HUNT, N. The netflix recommender system: Algorithms, business value, and innovation. **ACM Trans. Manage. Inf. Syst.**, Association for Computing Machinery, New York, NY, USA, v. 6, n. 4, dez. 2016. ISSN 2158-656X. Disponível em: <https://doi.org/10.1145/2843948>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GOODMAN, B.; FLAXMAN, S. **EU regulations on algorithmic decision-making and a "right to explanation"**. 2016. Cite arxiv:1606.08813Comment: presented at 2016 ICML Workshop on Human Interpretability in Machine Learning (WHI 2016), New York, NY. Disponível em: <http://arxiv.org/abs/1606.08813>.

GUIDOTTI, R. Evaluating local explanation methods on ground truth. **Artificial Intelligence**, v. 291, p. 103428, 2021. ISSN 0004-3702. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0004370220301776>.

GUIDOTTI, R.; MONREALE, A.; RUGGIERI, S.; TURINI, F.; GIANNOTTI, F.; PEDRESCHI, D. A survey of methods for explaining black box models. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 51, n. 5, ago. 2018. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3236009>.

GUO, H.; TANG, R.; YE, Y.; LI, Z.; HE, X. Deepfm: A factorization-machine based neural network for ctr prediction. In: **Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17**. [s.n.], 2017. p. 1725–1731. Disponível em: <https://doi.org/10.24963/ijcai.2017/239>.

HARPER, F. M.; KONSTAN, J. A. The movielens datasets: History and context. **ACM Trans. Interact. Intell. Syst.**, ACM, New York, NY, USA, v. 5, n. 4, p. 19:1–19:19, dez. 2015. ISSN 2160-6455. Disponível em: <http://doi.acm.org/10.1145/2827872>.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning**. New York, NY, USA: Springer New York Inc., 2001. (Springer Series in Statistics).

HE, C.; PARRA, D.; VERBERT, K. Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities. **Expert Systems with Applications**, v. 56, p. 9 – 27, 2016. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417416300367>.

HE, X.; CHEN, T.; KAN, M.-Y.; CHEN, X. Trirank: Review-aware explainable recommendation by modeling aspects. In: **Proceedings of the 24th ACM International on Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2015. (CIKM '15), p. 1661–1670. ISBN 978-1-4503-3794-6. Disponível em: <http://doi.acm.org/10.1145/2806416.2806504>.

Heckel, R.; Vlachos, M.; Parnell, T.; Duenner, C. Scalable and interpretable product recommendations via overlapping co-clustering. In: **2017 IEEE 33rd International Conference on Data Engineering (ICDE)**. [S.l.: s.n.], 2017. p. 1033–1044.

HERLOCKER, J. L.; KONSTAN, J. A.; RIEDL, J. Explaining collaborative filtering recommendations. In: **Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work**. New York, NY, USA: ACM, 2000. (CSCW '00), p. 241–250. ISBN 1-58113-222-0. Disponível em: <http://doi.acm.org/10.1145/358916.358995>.

HERNANDO, A.; BOBADILLA, J.; ORTEGA, F.; GUTIÉRREZ, A. Trees for explaining recommendations made through collaborative filtering. **Inf. Sci.**, Elsevier Science Inc., New York, NY, USA, v. 239, p. 1–17, ago. 2013. ISSN 0020-0255. Disponível em: <http://dx.doi.org/10.1016/j.ins.2013.03.018>.

HIDASI, B.; QUADRANA, M.; KARATZOGLOU, A.; TIKK, D. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In: **Proceedings of the 10th ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2016. (RecSys '16), p. 241–248. ISBN 9781450340359. Disponível em: <https://doi.org/10.1145/2959100.2959167>.

HO, T. K. Random decision forests. In: **Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1**. Washington, DC, USA: IEEE Computer Society, 1995. (ICDAR '95), p. 278–. ISBN 0-8186-7128-9. Disponível em: <http://dl.acm.org/citation.cfm?id=844379.844681>.

HORN, R. A.; JOHNSON, C. R. Norms for vectors and matrices. In: ____. **Matrix Analysis**. 2. ed. [S.l.]: Cambridge University Press, 2012. p. 313–386.

HUG, N. Surprise: A python library for recommender systems. **Journal of Open Source Software**, The Open Journal, v. 5, n. 52, p. 2174, 2020. Disponível em: <https://doi.org/10.21105/joss.02174>.

HUNT, J.; PRICE, C. Explaining qualitative diagnosis. **Engineering Applications of Artificial Intelligence**, v. 1, n. 3, p. 161 – 169, 1988. ISSN 0952-1976. Disponível em: <http://www.sciencedirect.com/science/article/pii/0952197688900024>.

JAMES, G.; WITTEN, D.; HASTIE, T.; TIBSHIRANI, R. **An Introduction to Statistical Learning: With Applications in R**. [S.l.]: Springer Publishing Company, Incorporated, 2014. ISBN 1461471370, 9781461471370.

JEYAKUMAR, J. V.; NOOR, J.; CHENG, Y.-H.; GARCIA, L.; SRIVASTAVA, M. How can i explain this to you? an empirical study of deep neural network explanation methods. **Advances in Neural Information Processing Systems**, v. 33, 2020.

JIA, Y.; BAILEY, J.; RAMAMOHANARAO, K.; LECKIE, C.; HOULE, M. E. Improving the quality of explanations with local embedding perturbations. In: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining**. New York, NY, USA: Association for Computing Machinery, 2019. (KDD '19), p. 875–884. ISBN 9781450362016. Disponível em: <https://doi.org/10.1145/3292500.3330930>.

Khalifa Bashier Babiker, H.; Goebel, R. An Introduction to Deep Visual Explanation. **ArXiv e-prints**, nov. 2017.

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing Systematic Literature Reviews in Software Engineering**. 2007.

KOREN, Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: **Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2008. (KDD '08), p. 426–434. ISBN 978-1-60558-193-4. Disponível em: <http://doi.acm.org/10.1145/1401890.1401944>.

KOREN, Y.; BELL, R.; VOLINSKY, C. Matrix factorization techniques for recommender systems. **Computer**, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 42, n. 8, p. 30–37, ago. 2009. ISSN 0018-9162. Disponível em: <http://dx.doi.org/10.1109/MC.2009.263>.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: **Proceedings of the 25th International Conference on Neural Information Processing Systems**. USA: Curran Associates Inc., 2012. (NIPS'12), p. 1097–1105. Disponível em: <http://dl.acm.org/citation.cfm?id=2999134.2999257>.

LAKKARAJU, H.; KAMAR, E.; CARUANA, R.; LESKOVEC, J. Interpretable & explorable approximations of black box models. **CoRR**, abs/1707.01154, 2017. Disponível em: <http://arxiv.org/abs/1707.01154>.

LAUGEL, T.; RENARD, X.; LESOT, M.; MARSALA, C.; DETYNIECKI, M. Defining locality for surrogates in post-hoc interpretablity. **CoRR**, abs/1806.07498, 2018. Disponível em: <http://arxiv.org/abs/1806.07498>.

LEI, C.; LIU, D.; LI, W.; ZHA, Z.; LI, H. Comparative deep learning of hybrid representations for image recommendations. **CoRR**, abs/1604.01252, 2016. Disponível em: <http://arxiv.org/abs/1604.01252>.

LI, Q.; KIM, B. M. An approach for combining content-based and collaborative filters. In: **Proceedings of the Sixth International Workshop on Information Retrieval with Asian Languages - Volume 11**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003. (AsianIR '03), p. 17–24. Disponível em: <http://dx.doi.org/10.3115/1118935.1118938>.

LIN, J. The neural hype and comparisons against weak baselines. **SIGIR Forum**, Association for Computing Machinery, New York, NY, USA, v. 52, n. 2, p. 40–51, jan. 2019. ISSN 0163-5840. Disponível em: <https://doi.org/10.1145/3308774.3308781>.

LINDEN, G.; SMITH, B.; YORK, J. Amazon.com recommendations: Item-to-item collaborative filtering. **IEEE Internet Computing**, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 7, n. 1, p. 76–80, jan. 2003. ISSN 1089-7801. Disponível em: <http://dx.doi.org/10.1109/MIC.2003.1167344>.

LIPTON, Z. C. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. **Queue**, Association for Computing Machinery, New York, NY, USA, v. 16, n. 3, p. 31–57, jun. 2018. ISSN 1542-7730. Disponível em: <https://doi.org/10.1145/3236386.3241340>.

LIU, H.; WEN, J.; JING, L.; YU, J.; ZHANG, X.; ZHANG, M. In2rec: Influence-based interpretable recommendation. In: **Proceedings of the 28th ACM International Conference on Information and Knowledge Management**. New York, NY, USA: Association for Computing Machinery, 2019. (CIKM '19), p. 1803–1812. ISBN 9781450369763. Disponível em: <https://doi.org/10.1145/3357384.3358017>.

LOPEZ-SUAREZ, A.; KAMEL, M. Dykor: A method for generating the content of explanations in knowledge systems. **Know.-Based Syst.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 7, n. 3, p. 177–188, set. 1994. ISSN 0950-7051. Disponível em: <https://doi.org/10.1016/0950-7051(94)90004-3>.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2017. v. 30, p. 4765–4774. Disponível em: <https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf>.

MCAULEY, J.; LESKOVEC, J. Hidden factors and hidden topics: Understanding rating dimensions with review text. In: **Proceedings of the 7th ACM Conference on Recommender Systems**. New York, NY, USA: ACM, 2013. (RecSys '13), p. 165–172. ISBN 978-1-4503-2409-0. Disponível em: <http://doi.acm.org/10.1145/2507157.2507163>.

MCNEE, S. M.; RIEDL, J.; KONSTAN, J. A. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In: **CHI '06 Extended Abstracts on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 2006. (CHI EA '06), p. 1097–1101. ISBN 1595932984. Disponível em: <https://doi.org/10.1145/1125451.1125659>.

MISHRA, S.; STURM, B. L.; DIXON, S. Local interpretable model-agnostic explanations for music content analysis. In: **ISMIR**. [S.l.: s.n.], 2017.

MOLNAR, C. **Interpretable Machine Learning**: A guide for making black box models explainable. [S.l.: s.n.], 2019. <https://christophm.github.io/interpretable-ml-book/>.

MUSTO, C.; NARDUCCI, F.; LOPS, P.; GEMMIS, M. D.; SEMERARO, G. Explod: A framework for explaining recommendations based on the linked open data cloud. In: **Proceedings of the 10th ACM Conference on Recommender Systems**. New York, NY, USA: ACM, 2016. (RecSys '16), p. 151–154. ISBN 978-1-4503-4035-9. Disponível em: <http://doi.acm.org/10.1145/2959100.2959173>.

NóBREGA, C.; MARINHO, L. Towards explaining recommendations through local surrogate models. In: **Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing**. New York, NY, USA: Association for Computing Machinery, 2019. (SAC '19), p. 1671–1678. ISBN 9781450359337. Disponível em: <https://doi.org/10.1145/3297280.3297443>.

NUNES, I.; JANNACH, D. A systematic review and taxonomy of explanations in decision support and recommender systems. **User Modeling and User-Adapted Interaction**, Kluwer Academic Publishers, Hingham, MA, USA, v. 27, n. 3-5, p. 393–444, dez. 2017. ISSN 0924-1868. Disponível em: <https://doi.org/10.1007/s11257-017-9195-0>.

OKURA, S.; TAGAMI, Y.; ONO, S.; TAJIMA, A. Embedding-based news recommendation for millions of users. In: **Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2017. (KDD '17), p. 1933–1942. ISBN 9781450348874. Disponível em: <https://doi.org/10.1145/3097983.3098108>.

OTTER, D. W.; MEDINA, J. R.; KALITA, J. K. A survey of the usages of deep learning in natural language processing. **CoRR**, abs/1807.10854, 2018. Disponível em: <http://arxiv.org/abs/1807.10854>.

OZSOY, M. G. From word embeddings to item recommendation. **CoRR**, abs/1601.01356, 2016. Disponível em: <http://arxiv.org/abs/1601.01356>.

PAPADIMITRIOU, A.; SYMEONIDIS, P.; MANOLOPOULOS, Y. A generalized taxonomy of explanations styles for traditional and social recommender systems. **Data Min. Knowl. Discov.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 24, n. 3, p. 555–583, maio 2012. ISSN 1384-5810. Disponível em: <http://dx.doi.org/10.1007/s10618-011-0215-0>.

PARRA, D.; BRUSILOVSKY, P.; TRATTNER, C. See what you want to see: Visual user-driven approach for hybrid recommendation. In: **Proceedings of the 19th International Conference on Intelligent User Interfaces**. New York, NY, USA: Association for Computing Machinery, 2014. (IUI '14), p. 235–240. ISBN 9781450321846. Disponível em: <https://doi.org/10.1145/2557500.2557542>.

PATEREK, A. Improving regularized singular value decomposition for collaborative filtering. In: **Proceedings of the KDD Cup Workshop at the 13th ACM International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2007. (SIGKDD '07), p. 39–42. ISBN 1-59593-180-5. Disponível em: <http://doi.acm.org/10.1145/1102351.1102441>.

PEAKE, G.; WANG, J. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In: **Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. New York, NY, USA: ACM, 2018. (KDD '18), p. 2060–2069. ISBN 978-1-4503-5552-0. Disponível em: <http://doi.acm.org/10.1145/3219819.3220072>.

PU, P.; CHEN, L. Trust building with explanation interfaces. In: **Proceedings of the 11th International Conference on Intelligent User Interfaces**. New York, NY, USA: ACM, 2006. (IUI '06), p. 93–100. ISBN 1-59593-287-9. Disponível em: <http://doi.acm.org/10.1145/1111449.1111475>.

PU, P.; CHEN, L. Trust-inspiring explanation interfaces for recommender systems. **Know.-Based Syst.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 20, n. 6, p. 542–556, ago. 2007. ISSN 0950-7051. Disponível em: <http://dx.doi.org/10.1016/j.knosys.2007.04.004>.

RANA, A.; BRIDGE, D. Explanation chains: Recommendations by explanation. In: TIKK, D.; PU, P. (Ed.). **Procs. of the Poster Track of the 11th ACM Conference on Recommender Systems**. [S.l.]: CEUR Workshop Proceedings, vol-1905, 2017.

RENDLE, S. Factorization machines. In: **Proceedings of the 2010 IEEE International Conference on Data Mining**. Washington, DC, USA: IEEE Computer Society, 2010. (ICDM '10), p. 995–1000. ISBN 978-0-7695-4256-0. Disponível em: <http://dx.doi.org/10.1109/ICDM.2010.127>.

RESNICK, P.; VARIAN, H. R. Recommender systems. **Communications of the ACM**, ACM, v. 40, n. 3, p. 56–58, 1997.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. Nothing else matters: Model-agnostic explanations by identifying prediction invariance. **CoRR**, abs/1611.05817, 2016. Disponível em: <http://arxiv.org/abs/1611.05817>.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "why should i trust you?": Explaining the predictions of any classifier. In: **Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2016. (KDD '16), p. 1135–1144. ISBN 978-1-4503-4232-2. Disponível em: <http://doi.acm.org/10.1145/2939672.2939778>.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: RICCI, F.; ROKACH, L.; SHAPIRA, B.; KANTOR, P. B. (Ed.). **Recommender Systems Handbook**. Springer, 2011. p. 1–35. ISBN 978-0-387-85819-7. Disponível em: <http://dblp.uni-trier.de/db/reference/rsh/rsh2011.html\#RicciRS11>.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Recommender systems handbook. In: **Recommender systems handbook**. [S.l.]: Springer, 2015.

ROETZEL, P. G. Information overload in the information age: a review of the literature from business administration, business psychology, and related disciplines with a bibliometric approach and framework development. **Business Research**, v. 12, n. 2, p. 479–522, 2019. Disponível em: <https://EconPapers.repec.org/RePEc:spr:busres:v:12:y:2019:i:2:d:10.1007_s40685-018-0069-z>.

SALAKHUTDINOV, R.; MNIH, A.; HINTON, G. Restricted boltzmann machines for collaborative filtering. In: **Proceedings of the 24th International Conference on Machine Learning**. New York, NY, USA: ACM, 2007. (ICML '07), p. 791–798. ISBN 978-1-59593-793-3. Disponível em: <http://doi.acm.org/10.1145/1273496.1273596>.

SARKAR, S.; WEYDE, T.; GARCEZ, A. S. d'Avila; SLABAUGH, G. G.; DRAGICEVIC, S.; PERCY, C. Accuracy and interpretability trade-offs in machine learning applied to safer gambling. In: **CoCo@NIPS**. [S.l.: s.n.], 2016.

SARWAR, B. M.; KARYPIS, G.; KONSTAN, J. A.; RIEDL, J. T. Application of dimensionality reduction in recommender systems: A case study. In: **WebKDD Workshop at the ACM SIGKKD**. [S.l.: s.n.], 2000.

SCHEIN, A. I.; POPESCUL, A.; UNGAR, L. H.; PENNOCK, D. M. Methods and metrics for cold-start recommendations. In: **Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: Association for Computing Machinery, 2002. (SIGIR '02), p. 253–260. ISBN 1581135610. Disponível em: <https://doi.org/10.1145/564376.564421>.

SCHWARTZ, B. **The Paradox of Choice: Why More Is Less**. Harper Perennial, 2005. Paperback. ISBN 0060005696. Disponível em: <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&path=ASIN/0060005696>.

SEO, S.; HUANG, J.; YANG, H.; LIU, Y. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In: **Proceedings of the Eleventh ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2017. (RecSys '17), p. 297–305. ISBN 9781450346528. Disponível em: <https://doi.org/10.1145/3109859.3109890>.

SINGH, J.; ANAND, A. Exs: Explainable search using local model agnostic interpretability. In: **Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2019. (WSDM '19), p. 770–773. ISBN 9781450359405. Disponível em: <https://doi.org/10.1145/3289600.3290620>.

SINHA, R.; SWEARINGEN, K. The role of transparency in recommender systems. In: **CHI '02 Extended Abstracts on Human Factors in Computing Systems**. New York, NY, USA: ACM, 2002. (CHI EA '02), p. 830–831. ISBN 1-58113-454-1. Disponível em: <http://doi.acm.org/10.1145/506443.506619>.

Symeonidis, P.; Nanopoulos, A.; Manolopoulos, Y. Providing justifications in recommender systems. **IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans**, v. 38, n. 6, p. 1262–1272, 2008.

TAKáCS, G.; PILáSZY, I.; NÉMETH, B.; TIKK, D. Major components of the gravity recommendation system. **SIGKDD Explor. Newsl.**, ACM, New York, NY, USA, v. 9, n. 2, p. 80–83, dez. 2007. ISSN 1931-0145. Disponível em: <http://doi.acm.org/10.1145/1345448.1345466>.

TAN, H. F.; SONG, K.; UDELL, M.; SUN, Y.; ZHANG, Y. Why should you trust my interpretation? understanding uncertainty in LIME predictions. **CoRR**, abs/1904.12991, 2019. Disponível em: <http://arxiv.org/abs/1904.12991>.

TAO, Y.; JIA, Y.; WANG, N.; WANG, H. The fact: Taming latent factor models for explainability with factorization trees. In: **Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: Association for Computing Machinery, 2019. (SIGIR'19), p. 295–304. ISBN 9781450361729. Disponível em: <https://doi.org/10.1145/3331184.3331244>.

TINTAREV, N.; MASTHOFF, J. A survey of explanations in recommender systems. In: **Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering Workshop**. Washington, DC, USA: IEEE Computer Society, 2007. (ICDEW '07), p. 801–810. ISBN 978-1-4244-0831-3. Disponível em: <http://dx.doi.org/10.1109/ICDEW.2007.4401070>.

TINTAREV, N.; MASTHOFF, J. Evaluating the effectiveness of explanations for recommender systems. **User Modeling and User-Adapted Interaction**, Kluwer Academic Publishers, Hingham, MA, USA, v. 22, n. 4-5, p. 399–439, out. 2012. ISSN 0924-1868. Disponível em: <http://dx.doi.org/10.1007/s11257-011-9117-5>.

VERBERT, K.; PARRA, D.; BRUSILOVSKY, P.; DUVAL, E. Visualizing recommendations to support exploration, transparency and controllability. In: **Proceedings of the 2013 International Conference on Intelligent User Interfaces**. New York, NY, USA: Association for Computing Machinery, 2013. (IUI '13), p. 351–362. ISBN 9781450319652. Disponível em: <https://doi.org/10.1145/2449396.2449442>.

VIG, J.; SEN, S.; RIEDL, J. Tagsplanations: Explaining recommendations using tags. In: **Proceedings of the 14th International Conference on Intelligent User Interfaces**. New York, NY, USA: ACM, 2009. (IUI '09), p. 47–56. ISBN 978-1-60558-168-2. Disponível em: <http://doi.acm.org/10.1145/1502650.1502661>.

VILONE, G.; LONGO, L. **Explainable Artificial Intelligence: a Systematic Review**. 2020.

VOULODIMOS, A.; DOULAMIS, N.; DOULAMIS, A.; PROTOPAPADAKIS, E.; ANDINA, D. Deep learning for computer vision: A brief review. **Intell. Neuroscience**, Hindawi Limited, London, GBR, v. 2018, jan. 2018. ISSN 1687-5265. Disponível em: <https://doi.org/10.1155/2018/7068349>.

WANG, H.; WANG, N.; YEUNG, D.-Y. Collaborative deep learning for recommender systems. In: **Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2015. (KDD '15), p. 1235–1244. ISBN 9781450336642. Disponível em: <https://doi.org/10.1145/2783258.2783273>.

WANG, X.; HE, X.; FENG, F.; NIE, L.; CHUA, T.-S. Tem: Tree-enhanced embedding model for explainable recommendation. In: **Proceedings of the 2018 World Wide Web Conference**. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2018. (WWW '18), p. 1543–1552. ISBN 978-1-4503-5639-8. Disponível em: <https://doi.org/10.1145/3178876.3186066>.

WANG, X.; YU, L.; REN, K.; TAO, G.; ZHANG, W.; YU, Y.; WANG, J. Dynamic attention deep model for article recommendation by learning human editors' demonstration. In: **Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2017. (KDD '17), p. 2051–2059. ISBN 9781450348874. Disponível em: <https://doi.org/10.1145/3097983.3098096>.

ZHANG, S.; YAO, L.; SUN, A.; TAY, Y. Deep learning based recommender system: A survey and new perspectives. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 52, n. 1, fev. 2019. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3285029>.

ZHANG, Y.; CHEN, X. Explainable recommendation: A survey and new perspectives. **CoRR**, abs/1804.11192, 2018. Disponível em: <http://arxiv.org/abs/1804.11192>.

ZHANG, Y.; LAI, G.; ZHANG, M.; ZHANG, Y.; LIU, Y.; MA, S. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In: **Proceedings of the 37th International ACM SIGIR Conference on Research & Development in**

Information Retrieval. New York, NY, USA: ACM, 2014. (SIGIR '14), p. 83–92. ISBN 978-1-4503-2257-7. Disponível em: <http://doi.acm.org/10.1145/2600428.2609579>.

ZHANG, Y.; ZHANG, M.; ZHANG, Y.; LAI, G.; LIU, Y.; ZHANG, H.; MA, S. Daily-aware personalized recommendation based on feature-level time series analysis. In: **Proceedings of the 24th International Conference on World Wide Web**. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2015. (WWW '15), p. 1373–1383. ISBN 9781450334693. Disponível em: <https://doi.org/10.1145/2736277.2741087>.

ZIEGLER, C.-N.; MCNEE, S. M.; KONSTAN, J. A.; LAUSEN, G. Improving recommendation lists through topic diversification. In: **Proceedings of the 14th International Conference on World Wide Web**. New York, NY, USA: Association for Computing Machinery, 2005. (WWW '05), p. 22–32. ISBN 1595930469. Disponível em: <https://doi.org/10.1145/1060745.1060754>.

Apêndice A

Estudo Preliminar com Usuários

Durante a execução deste trabalho, foi elaborado e realizado um experimento com usuários. Mais especificamente, convidamos, via e-email, usuários ativos da plataforma MovieLens[1] para um estudo, cujo objetivo era avaliar se as explicações geradas ajudavam os usuários a tomar melhores decisões. A Seção A.1 descreve em mais detalhes a metodologia desse estudo.

A.1 Metodologia do Estudo com Usuários

Uma explicação pode ter diferentes objetivos (vide Seção 2.3.3). No nosso estudo com usuários, decidimos abordar o objetivo efetividade, que consiste em ajudar usuários a tomar boas decisões, ou seja, ajudá-los a aceitar recomendações acuradas e descartar recomendações irrelevantes. A Figura A.1 mostra o e-mail que convidava os usuários a participar do estudo.

Dado o nosso objetivo, o estudo simulava o processo de tomada de decisão com os seguintes passos:

1. Mostrar uma lista de filmes (do histórico do usuário) acompanhados com as respectivas predições e explicações. O poster e título do filme eram omitidos para que o usuário não identificasse o filme. A lista era composta por 10 filmes: 3 avaliados positivamente, 3 avaliados negativamente e 4 filmes considerados neutros. O critério da escolha era baseado na média de avaliações do usuário. Então, pedir aos usuários que

[1] https://movielens.org/

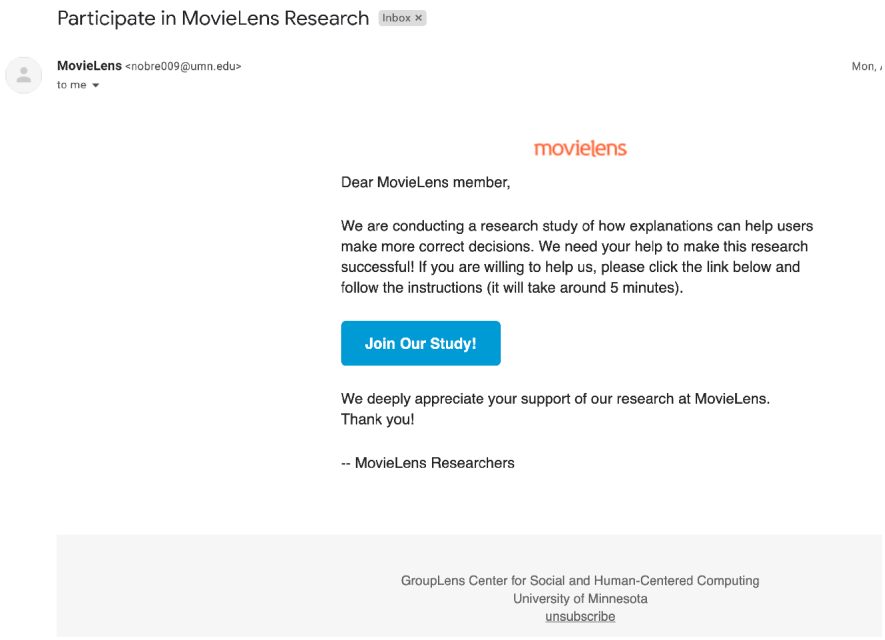


Figura A.1: Exemplo do email que convidava usuários a participarem do nosso estudo com usuários.

tomem a decisão de escolher os 3 filmes que mais gostaria de consumir. Note que, nesse momento, a escolha é baseada apenas nas predições e explicações.

2. Obter um feedback explícito acerca das expliações exibidos, usuários que tomem a decisão de escolher o itens que mais gostariam de consumir. Note que a escolha é baseada apenas nas predições e explicações.

3. Mostrar novamente a mesma lista de filmes do passo 1, porém exibindo o poster e título dos filmes e omitindo as explicações. Então, pedir para que o usuário novamente escolha os itens que mais gostaria de consumir. Aqui, a lista é embaralhada, ou seja, os itens são exibidos numa ordem diferente do passo 1, com o intuito de evitar que usuário determine sua escolha baseada na seleção anterior.

As Figuras A.2, A.3 A.4 representam os passos 1, 2 e 3, respectivamente.

A avaliação da efetividade das explicações consistia, basicamente, em comparar os itens escolhidos no passo 1 e no passo 3, de forma a verificar se o itens escolhidos são, de fato, os itens avaliados positivamente. Poucos usuários participaram desse estudo e, por isso, não foi possível tirar conclusões confiáveis sobre os resultados obtidos.

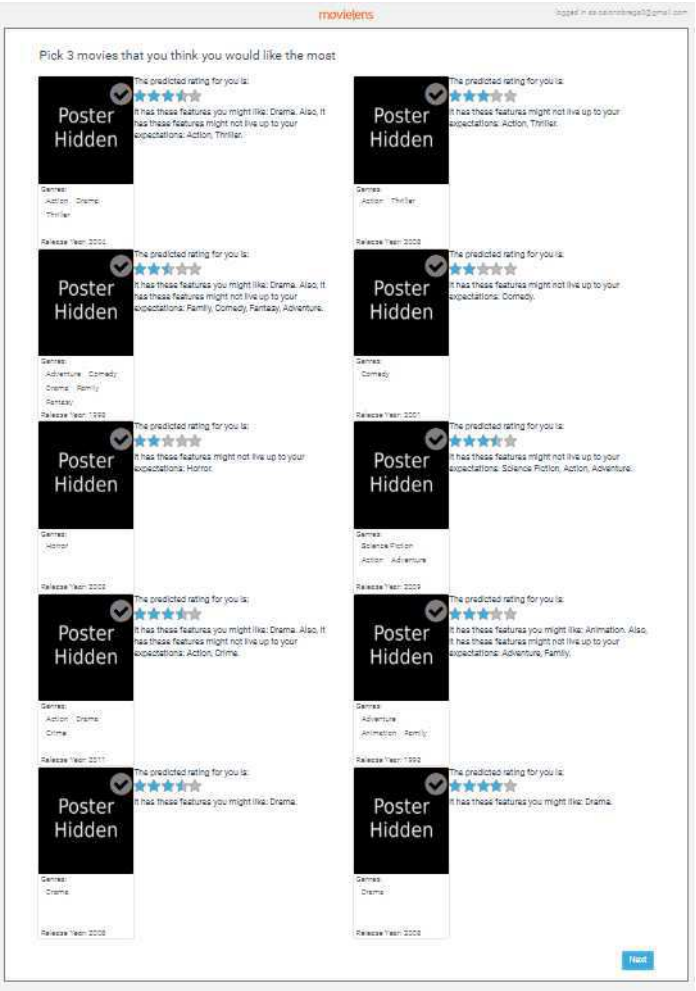


Figura A.2: Exemplo do passo 1 do estudo preliminar com usuários. É exibida uma lista com 10 filmes com as respectivas predições e explicações. O poster e o título do filme são omitidos.

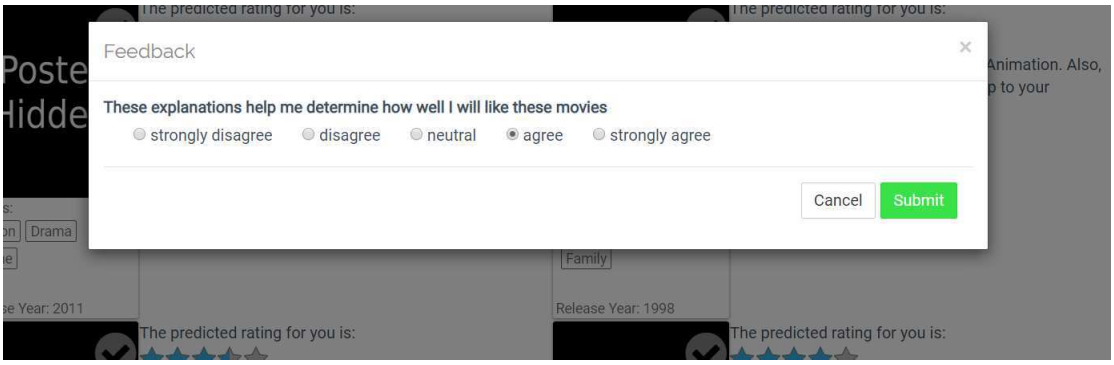


Figura A.3: Passo 2 do estudo preliminar com usuários, que consiste em o feedback do usuário acerca das explicações.

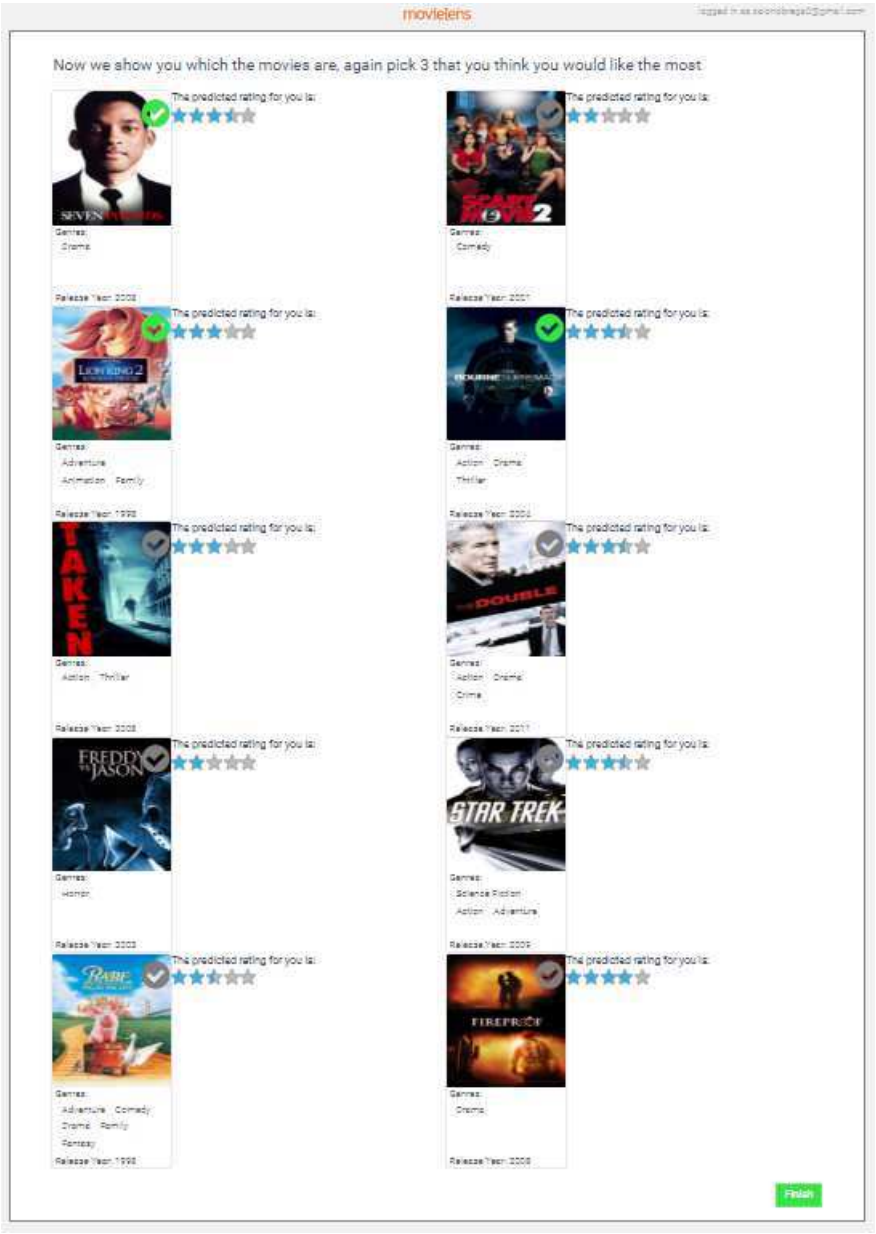


Figura A.4: Exemplo do passo 3 do estudo preliminar com usuários. A lista de filmes do passo 1 é exibida em uma ordem aleatória, de forma que o poster e o título do filme são revelados e as explicações omitidas.