
Capítulo 5

Recomendador baseado em conhecimento Sistemas

"Conhecimento é saber que um tomate é uma fruta. Sabedoria é saber que não se deve colocá-lo em uma salada de frutas." – Brian O'Driscoll

5.1 Introdução

Tanto sistemas baseados em conteúdo quanto sistemas colaborativos exigem uma quantidade significativa de dados sobre experiências anteriores de compra e avaliação. Por exemplo, sistemas colaborativos exigem uma matriz de avaliações razoavelmente bem preenchida para fazer recomendações futuras. Nos casos em que a quantidade de dados disponíveis é limitada, as recomendações são ruins ou não abrangem todo o espectro de combinações de itens e usuários. Esse problema também é conhecido como problema da inicialização a frio. Diferentes sistemas apresentam diferentes níveis de suscetibilidade a esse problema.

Por exemplo, sistemas colaborativos são os mais suscetíveis e não conseguem lidar muito bem com novos itens ou novos usuários. Sistemas de recomendação baseados em conteúdo são um pouco melhores em lidar com novos itens, mas ainda assim não conseguem fornecer recomendações a novos usuários.

Além disso, esses métodos geralmente não são adequados para domínios nos quais o produto é altamente personalizado. Exemplos incluem itens como imóveis, automóveis, solicitações de turismo, serviços financeiros ou bens de luxo caros. Esses itens são comprados raramente e, muitas vezes, não há classificações suficientes disponíveis. Em muitos casos, o domínio do item pode ser complexo e pode haver poucas instâncias de um item específico com um conjunto específico de propriedades. Por exemplo, pode-se querer comprar uma casa com um número específico de quartos, gramado, localidade e assim por diante. Devido à complexidade na descrição do item, pode ser difícil obter um conjunto razoável de classificações que reflitam o histórico de um usuário em um item semelhante. Da mesma forma, uma classificação antiga para um carro com um conjunto específico de opções pode nem mesmo ser relevante no contexto atual.

Como lidar com tamanha personalização e escassez de classificações? Sistemas de recomendação baseados em conhecimento dependem da solicitação explícita dos requisitos do usuário para tais itens. No entanto, em domínios tão complexos, muitas vezes é difícil para os usuários enunciarem completamente ou mesmo entenderem como seus requisitos correspondem à disponibilidade do produto. Por exemplo, um usuário pode nem estar ciente de que um carro com uma determinada combinação de eficiência de combustível e potência está disponível. Portanto, tais sistemas utilizam feedback interativo, que permite ao usuário explorar o espaço inherentemente complexo do produto e aprender sobre as compensações disponíveis entre as várias opções. O processo de recuperação e exploração é facilitado por bases de conhecimento que descrevem as utilidades e/ou compensações de vários recursos no domínio do produto. O uso de bases de conhecimento é tão importante para um processo eficaz de recuperação e exploração que tais sistemas são chamados de sistemas de recomendação baseados em conhecimento.

Sistemas de recomendação baseados em conhecimento são adequados para a recomendação de itens que não são comprados regularmente. Além disso, nesses domínios de itens, os usuários geralmente são mais ativos em explicitar suas necessidades. Um usuário pode frequentemente estar disposto a aceitar uma recomendação de filme sem muita informação, mas não estaria disposto a aceitar recomendações sobre uma casa ou um carro sem ter informações detalhadas sobre as características específicas do item. Portanto, sistemas de recomendação baseados em conhecimento são adequados para tipos de domínios de itens diferentes daqueles de sistemas colaborativos e baseados em conteúdo. Em geral, sistemas de recomendação baseados em conhecimento são apropriados nas seguintes situações:

1. Os clientes desejam especificar explicitamente suas necessidades. Portanto, a interatividade é um componente crucial desses sistemas. Observe que sistemas colaborativos e baseados em conteúdo não permitem esse tipo de feedback detalhado.
2. É difícil obter classificações para um tipo específico de item devido à maior complexidade do domínio do produto em termos de tipos de itens e opções disponíveis.
3. Em alguns domínios, como computadores, as classificações podem ser sensíveis ao tempo. As classificações de um carro ou computador antigo não são muito úteis para recomendações, pois evoluem com a disponibilidade do produto e as necessidades correspondentes do usuário.

Uma parte crucial dos sistemas baseados em conhecimento é o maior controle que o usuário tem na condução do processo de recomendação. Esse maior controle é resultado direto da necessidade de especificar requisitos detalhados em um domínio de problema inherentemente complexo. Em um nível básico, as diferenças conceituais nas três categorias de recomendações são descritas na Tabela 5.1.

Observe que também há diferenças significativas nos dados de entrada usados por vários sistemas.

As recomendações de sistemas colaborativos e baseados em conteúdo baseiam-se principalmente em dados históricos, enquanto os sistemas baseados em conhecimento baseiam-se nas especificações diretas dos usuários sobre o que eles desejam. Uma importante característica distinta dos sistemas baseados em conhecimento é um alto nível de personalização para o domínio específico. Essa personalização é alcançada por meio do uso de uma base de conhecimento que codifica o conhecimento de domínio relevante na forma de restrições ou métricas de similaridade. Alguns sistemas baseados em conhecimento também podem usar atributos do usuário (por exemplo, atributos demográficos) além dos atributos do item, que são especificados no momento da consulta. Nesses casos, o conhecimento do domínio também pode codificar relacionamentos entre atributos do usuário e atributos do item. O uso de tais atributos, no entanto, não é universal para sistemas baseados em conhecimento, nos quais o foco maior está nos requisitos do usuário.

Os sistemas de recomendação baseados em conhecimento podem ser categorizados com base na metodologia interativa do usuário e nas bases de conhecimento correspondentes usadas para facilitar a interação.

Existem dois tipos principais de sistemas de recomendação baseados em conhecimento:

1. Sistemas de recomendação baseados em restrições: Em sistemas baseados em restrições [196, 197], os usuários normalmente especificam requisitos ou restrições (por exemplo, limites inferiores ou superiores) no item

Tabela 5.1: Os objetivos conceituais de vários sistemas de recomendação

Abordagem	Objetivo Conceitual	Entrada
Colaborativo	Dê-me recomendações com base em uma abordagem colaborativa que alavanca as avaliações e ações dos meus colegas/minhas.	Avaliações do usuário + classificações da comunidade
Conteúdo-baseado	Dê-me recomendações com base no conteúdo (atributos) Eu tenho favorecido em minhas avaliações e ações passadas.	Avaliações do usuário + atributos do item
Baseado no conhecimento	Dê-me recomendações com base em minhas especificações explícitas Especificação do tipo de conteúdo (atributos) que desejo.	usuário + atributos do item + conhecimento de domínio

atributos. Além disso, regras específicas de domínio são usadas para atender aos requisitos do usuário ou atributos para atributos de itens. Essas regras representam o conhecimento específico do domínio usadas pelo sistema. Essas regras podem assumir a forma de restrições específicas de domínio nos atributos do item (por exemplo, "Carros anteriores ao ano de 1970 não têm controle de cruzeiro"). Além disso, os sistemas baseados em restrições frequentemente criam regras que relacionam os atributos do usuário a atributos do item (por exemplo, "Investidores mais velhos não investem em produtos de risco ultra-alto."). Em Nesses casos, os atributos do usuário também podem ser especificados no processo de pesquisa. Dependendo com base no número e no tipo de resultados retornados, o usuário pode ter a oportunidade de modificar seus requisitos originais. Por exemplo, um usuário pode relaxar algumas restrições quando poucos resultados são retornados ou adicionar mais restrições quando muitos resultados são retornados. Este processo de busca é repetido interativamente até que o usuário chegue ao seu resultados desejados.

2. Sistemas de recomendação baseados em casos: Em sistemas de recomendação baseados em casos [102, 116, 377, 558], casos específicos são especificados pelo usuário como alvos ou pontos de ancoragem. Semelhança As métricas são definidas nos atributos do item para recuperar itens semelhantes a esses alvos. As métricas de similaridade são frequentemente definidas cuidadosamente de forma específica para cada domínio. Portanto, as métricas de similaridade formam o conhecimento de domínio que é usado em tais sistemas. os resultados retornados são frequentemente usados como novos casos de destino com algumas modificações interativas pelo usuário. Por exemplo, quando um usuário vê um resultado retornado que é quase semelhante para o que ela quer, ela pode reemitir uma consulta com esse alvo, mas com alguns dos atributos alterados a seu gosto. Alternativamente, uma crítica direcional pode ser especificada para podar itens com valores de atributos específicos maiores (ou menores) do que os de um determinado item de interesse. Este processo interativo é usado para guiar o usuário em direção ao item final recomendação.

Observe que, em ambos os casos, o sistema oferece a oportunidade para o usuário alterar seus requisitos especificados. No entanto, a maneira como isso é feito é diferente nos dois casos. sistemas baseados em casos, exemplos (ou casos) são usados como pontos de ancoragem para orientar a busca em conjunto com métricas de similaridade, enquanto em sistemas baseados em restrições, critérios/regras específicos (ou restrições) são usadas para orientar a busca. Em ambos os casos, os resultados apresentados são usados para modificar os critérios para encontrar recomendações adicionais. Os sistemas baseados em conhecimento derivam seu nome pelo fato de que eles codificam vários tipos de conhecimento de domínio na forma de restrições, regras, métricas de similaridade e funções de utilidade durante o processo de busca. Para Por exemplo, o design de uma métrica de similaridade ou de uma restrição específica requer uma abordagem específica do domínio conhecimento, que é crucial para o funcionamento eficaz do sistema de recomendação. Em geral, os sistemas baseados em conhecimento recorrem a fontes de informação altamente heterogéneas e específicas de cada domínio. conhecimento, em comparação com sistemas colaborativos e baseados em conteúdo, que trabalham com tipos de dados de entrada semelhantes em vários domínios. Como resultado, os sistemas baseados em conhecimento

Os sistemas são altamente personalizados e não são facilmente generalizáveis entre vários domínios. No entanto, os princípios mais amplos pelos quais essa personalização é feita são invariáveis entre os domínios. O objetivo deste capítulo é discutir esses princípios.

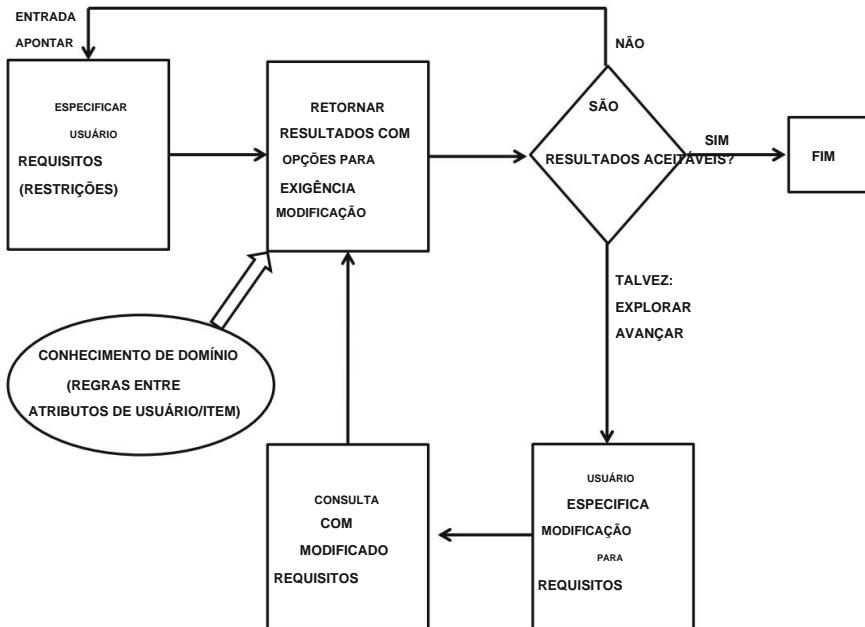
A interação entre o usuário e o recomendador pode assumir a forma de sistemas conversacionais, sistemas baseados em busca ou sistemas de navegação. Essas diferentes formas de orientação podem estar presentes isoladamente ou em combinação, e são definidas da seguinte forma:

1. Sistemas conversacionais: Neste caso, as preferências do usuário são determinadas no contexto de um ciclo de feedback. A principal razão para isso é que o domínio do item é complexo e as preferências do usuário só podem ser determinadas no contexto de um sistema conversacional iterativo.
2. Sistemas baseados em pesquisa: em sistemas baseados em pesquisa, as preferências do usuário são obtidas usando uma sequência predefinida de perguntas, como a seguinte: "Você prefere uma casa em uma área suburbana ou dentro da cidade?"
3. Recomendação baseada em navegação: na recomendação baseada em navegação, o usuário especifica uma série de solicitações de alteração para o item que está sendo recomendado no momento. Por meio de um conjunto iterativo de solicitações de mudança, é possível chegar a um item desejável. Um exemplo de solicitação de mudança especificada pelo usuário, quando uma casa específica está sendo recomendada, é o seguinte: "Gostaria de uma casa semelhante a cerca de 8 km a oeste da casa atualmente recomendada". Esses sistemas de recomendação também são chamados de sistemas de recomendação críticos [120, 121, 417].

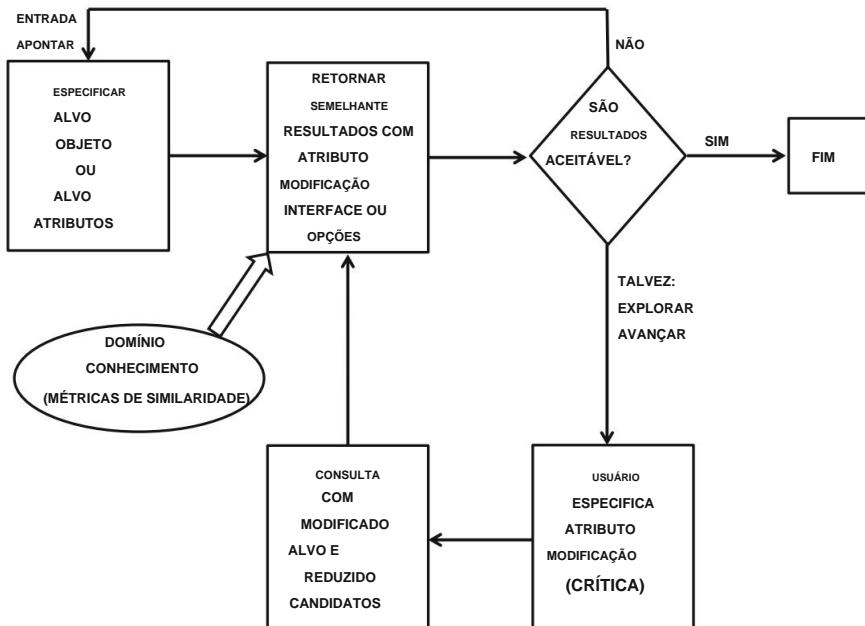
Essas diferentes formas de orientação são adequadas a diferentes tipos de sistemas de recomendação. Por exemplo, sistemas de crítica são naturalmente projetados para recomendadores baseados em casos, pois se critica um caso específico para chegar ao resultado desejado. Por outro lado, um sistema baseado em busca pode ser usado para definir requisitos do usuário para recomendadores baseados em restrições. Algumas formas de orientação podem ser usadas tanto com sistemas baseados em restrições quanto com sistemas baseados em casos. Além disso, diferentes formas de orientação também podem ser usadas em combinação em um sistema baseado em conhecimento. Não há regras rígidas sobre como se pode projetar a interface para um sistema baseado em conhecimento. O objetivo é sempre guiar o usuário por um espaço de produto complexo.

Exemplos típicos do processo interativo em recomendadores baseados em restrições e recomendadores baseados em casos são ilustrados nas Figuras 5.1(a) e (b), respectivamente. A abordagem interativa geral é bastante semelhante. A principal diferença entre os dois casos está em como o usuário especifica as consultas e interage com o sistema para refinamento subsequente.

Em sistemas baseados em restrições, requisitos específicos (ou restrições) são especificados pelo usuário, enquanto em sistemas baseados em casos, alvos específicos (ou casos) são especificados. Correspondentemente, diferentes tipos de processos interativos e conhecimento de domínio são utilizados nos dois sistemas. Em sistemas baseados em restrições, a consulta original é modificada pela adição, exclusão, modificação ou relaxamento do conjunto original de requisitos do usuário. Em sistemas baseados em casos, o alvo é modificado pela interação do usuário ou os resultados da pesquisa são podados por meio do uso de críticas direcionais. Em tais críticas, o usuário simplesmente declara se um atributo específico nos resultados da pesquisa precisa ser aumentado, diminuído ou alterado de uma determinada maneira. Tal abordagem representa um estilo mais coloquial do que simplesmente modificar o alvo. Em ambos os tipos de sistemas, uma motivação comum é que os usuários frequentemente não estão em condições de declarar exatamente seus requisitos antecipadamente em um domínio de produto complexo. Em sistemas baseados em restrições, esse problema é parcialmente resolvido por meio de uma base de conhecimento de regras, que mapeia os requisitos do usuário para os atributos do produto. Em sistemas baseados em casos, esse problema é resolvido



(a) Interação baseada em restrições



(b) Interação baseada em casos

Figura 5.1: Visão geral do processo interativo em recomendadores baseados em conhecimento

Tabela 5.2: Exemplos de atributos em um aplicativo de recomendação para compra de imóveis

ID do item: Camas. Banheiros. Localidade				Tipo Área do piso Preço		
1	3	2	Bronx	Moradia geminada	1600	220.000
2	5	2,5	Chappaqua em dpis níveis	973.000	3600	
	4		Yorktown	Rancho	630.000	2600
3 4	2	2 1,5	Yorktown	Condomínio	220.000	1500
5	4	2	Ossining	Colonial	430.000	2700

através de um estilo de crítica conversacional. O aspecto interativo é comum a ambos sistemas e é crucial para ajudar os usuários a descobrir como os itens em um produto complexo domínio atende às suas necessidades.

É de salientar que a maioria das formas de sistemas de recomendação baseados no conhecimento dependem fortemente nas descrições dos itens na forma de atributos relacionais em vez de tratando-os como palavras-chave de texto como sistemas baseados em conteúdo. Esta é uma consequência natural da complexidade inerente às recomendações baseadas no conhecimento em que domínios específicos o conhecimento pode ser mais facilmente codificado com atributos relacionais. Por exemplo, os atributos para um conjunto de casas em uma aplicação imobiliária é ilustrado na Tabela 5.2. Em casos baseados recomendadores, as métricas de similaridade são definidas em termos desses atributos para fornecer correspondências semelhantes às casas-alvo fornecidas pelo usuário. Observe que cada relacional atributo teria um significado e peso diferentes no processo de correspondência, dependendo em critérios específicos de domínio. Em sistemas baseados em restrições, as consultas são especificadas no forma de requisitos sobre esses atributos, como um preço máximo para a casa ou um localidade específica. Portanto, o problema se reduz a uma instância da satisfação de restrição problema, onde é preciso identificar o conjunto relevante de instâncias que satisfazem todas as restrições.

Este capítulo está organizado da seguinte forma. Recomendadores baseados em restrições são introduzidos em seção 5.2. Recomendadores baseados em casos são discutidos na seção 5.3. O uso de métodos persistentes A personalização em sistemas baseados em conhecimento é discutida na seção 5.4. Um resumo é fornecido na seção 5.5.

5.2 Sistemas de recomendação baseados em restrições

Os sistemas de recomendação baseados em restrições permitem que os usuários especifiquem requisitos rígidos ou restrições nos atributos do item. Além disso, um conjunto de regras é usado para corresponder os requisitos do cliente com atributos do item. No entanto, os clientes nem sempre podem especificar suas consultas em termos dos mesmos atributos que descrevem os itens. Portanto, um é necessário um conjunto adicional de regras que relate os requisitos do cliente com o produto atributos. Em relação ao exemplo anterior de compra de casa na Tabela 5.2, alguns exemplos de os atributos especificados pelo cliente são os seguintes:

Estado civil (categórico), Tamanho da família (numérico), subúrbio ou cidade (binário), Min-Quartos (numérico), Máximo de Quartos (numérico), Preço Máximo (numérico)

Esses atributos podem representar propriedades inerentes do cliente (por exemplo, dados demográficos), ou podem especificar os requisitos do cliente para o produto. Esses requisitos geralmente são

1 Os sistemas baseados em conteúdo são usados tanto na recuperação de informações quanto em ambientes relacionais, enquanto sistemas baseados em conhecimento são usados principalmente no ambiente relacional.

especificado interativamente durante o diálogo entre o cliente e o sistema de recomendação. Observe que muitos dos atributos dos requisitos não estão incluídos na Tabela 5.2. Embora o Embora os mapeamentos de alguns atributos de requisitos do cliente, como Preço Máximo, para atributos do produto sejam óbvios, os mapeamentos de outros, como Suburbano ou Rural, não são tão óbvios. Da mesma forma, em uma aplicação financeira, um cliente pode especificar um requisito de produto como como "investimentos conservadores", que precisam ser mapeados para atributos concretos do produto (por exemplo, Tipo de ativo = Tesouro) descrevendo diretamente os produtos. Claramente, é necessário de alguma forma ser capaz de mapear esses atributos/requisitos do cliente nos atributos do produto para para filtrar produtos para recomendação. Isso é feito por meio do uso de bases de conhecimento. As bases de conhecimento contêm regras adicionais que mapeiam atributos/requisitos do cliente aos atributos do produto:

Suburbano-ou-rural=Suburbano ÿ Localidade= Lista de localidades relevantes

Essas regras são chamadas de condições de filtro porque mapeiam os requisitos do usuário para o item atributos e usar esse mapeamento para filtrar os resultados recuperados. Observe que esses tipos de regras podem ser derivados do domínio do produto ou, mais raramente, podem ser derivados de mineração histórica de tais conjuntos de dados. Neste caso particular, é evidente que esta regra pode ser derivado diretamente usando informações geográficas publicamente disponíveis. Outro exemplo é o domínio do carro, onde certos pacotes opcionais podem ser válidos apenas com certos outros atributos. Por exemplo, um motor de alto torque pode estar disponível apenas em um modelo esportivo. Tais condições também são chamadas de condições de compatibilidade, porque podem ser usadas para descobrir rapidamente inconsistências nos requisitos especificados pelo usuário com o domínio do produto. Em muitos casos, tais restrições de compatibilidade podem ser integradas na interface do usuário. Por exemplo, o O site de preços de carros Edmunds.com impede que os usuários insiram requisitos mutuamente inconsistentes na interface do usuário. Em outros casos, onde a detecção de inconsistências não é possível na interface do usuário, tais inconsistências podem ser detectadas no momento do processamento da consulta por retornando conjuntos vazios de resultados.

Algumas das outras restrições de compatibilidade podem relacionar atributos do cliente entre si. Tais restrições são úteis quando os clientes especificam informações pessoais (por exemplo, informações demográficas) sobre si mesmos durante a sessão interativa. Por exemplo, atributos demográficos podem estar relacionados aos requisitos do produto do cliente com base em: restrições específicas de domínio ou experiência histórica. Um exemplo de tal restrição é como segue:

Estado civil = solteiro ÿ Quartos mínimos ÿ 5

Presumivelmente, por experiência específica de domínio ou por meio de mineração de dados históricos conjuntos, inferiu-se que indivíduos solteiros não preferem comprar casas muito grandes.

Da mesma forma, uma casa pequena pode não ser adequada para uma família muito grande. Essa restrição é modelado com a seguinte regra:

Tamanho família ÿ 5 ÿ Quartos mínimos ÿ 3

Portanto, existem três tipos principais de entrada no sistema de recomendação baseado em restrições:

1. A primeira classe de entradas é representada pelos atributos que descrevem as propriedades inerentes do usuário (por exemplo, dados demográficos, perfis de risco) e requisitos específicos no produto (por exemplo, Min-Bedrooms). Alguns desses atributos são fáceis de relacionar ao produto atributos, enquanto outros podem ser relacionados aos atributos do produto apenas por meio do uso de bases de conhecimento. Na maioria dos casos, as propriedades e os requisitos do cliente são especificados interativamente em uma sessão e não são persistentes em várias sessões.

Portanto, se outro usuário especificar o mesmo conjunto de requisitos em uma sessão, obterá o mesmo resultado. Isso é diferente de outros tipos de sistemas de recomendação, onde a personalização é persistente porque se baseia em dados históricos.

2. A segunda classe de entradas é representada por bases de conhecimento, que mapeiam atributos/requisitos do cliente para diversos atributos do produto. O mapeamento pode ser realizado direta ou indiretamente, da seguinte forma:

- Diretamente: Essas regras relacionam os requisitos do cliente aos requisitos rígidos sobre os atributos do produto. Um exemplo de tal regra é o seguinte:

Suburbano-ou-rural=Suburbano → Localidade= Lista de localidades relevantes

Mínimo de quartos ≥ 3 → Preço ≥ 100.000

Essas regras também são chamadas de condições de filtro.

- Indiretamente: Essas regras relacionam os atributos/requisitos do cliente aos requisitos normalmente esperados do produto. Portanto, essas regras também podem ser vistas como uma forma indireta de relacionar os atributos do cliente aos atributos do produto. Exemplos dessas regras são os seguintes:

Tamanho família ≥ 5 → Quartos mínimos ≥ 3

Tamanho família ≥ 5 → Banheiros mínimos ≥ 2

Observe que as condições em ambos os lados da regra representam atributos do cliente, embora as do lado direito sejam geralmente requisitos do cliente, que podem ser facilmente mapeados para atributos do produto. Essas restrições representam restrições de compatibilidade. Caso as restrições de compatibilidade ou as condições de filtro sejam inconsistentes com os requisitos especificados pelo cliente, a lista de itens recomendados estará vazia.

As bases de conhecimento mencionadas são derivadas de informações publicamente disponíveis, especialistas da área, experiências anteriores ou mineração de dados de conjuntos de dados históricos. Portanto, a construção das bases de conhecimento exige um esforço considerável.

3. Por fim, o catálogo de produtos contém uma lista de todos os produtos, juntamente com os atributos correspondentes. Um instantâneo de um catálogo de produtos para o exemplo de compra de imóvel é ilustrado na Tabela 5.2.

Portanto, o problema se resume a determinar todas as instâncias na lista de produtos disponíveis que atendem aos requisitos do cliente e às regras na base de conhecimento.

5.2.1 Retornando Resultados Relevantes

O problema de retornar resultados relevantes pode ser demonstrado como uma instância do problema de satisfação de restrições, visualizando cada item no catálogo como uma restrição aos atributos e expressando o catálogo na forma normal disjuntiva. Essa expressão é então combinada com as regras da base de conhecimento para determinar se existe uma região mutuamente consistente do espaço do produto.

De forma mais simples, o conjunto de regras e requisitos pode ser reduzido a uma tarefa de filtragem de dados no catálogo. Todos os requisitos do cliente e as regras ativas relevantes para o cliente são usados para construir uma consulta de seleção de banco de dados. As etapas para criar essa consulta de filtragem são as seguintes:

1. Para cada requisito (ou atributo pessoal) especificado pelo cliente em sua interface de usuário, é verificado se ele corresponde ao antecedente de uma regra na base de conhecimento. Se houver tal correspondência, o consequente dessa regra é tratado como uma condição de seleção válida. Por exemplo, considere o caso imobiliário mencionado anteriormente. Se o cliente tiver especificado Family-Size=6 e ZIP Code=10547 entre seus atributos pessoais e preferências na interface de usuário, será detectado que Family-Size=6 aciona as seguintes regras:

Tamanho família \geq 5 \Rightarrow Quartos mínimos \geq 3

Tamanho família \geq 5 \Rightarrow Banheiros mínimos \geq 2

Portanto, as consequências dessas condições são adicionadas aos requisitos do usuário. A base de regras é verificada novamente com esses requisitos expandidos, e observa-se que a restrição recém-adicionada Min-Bedrooms \geq 3 aciona as seguintes regras:

Mínimo de quartos \geq 3 \Rightarrow Preço \geq 100.000

Quartos mínimos \geq 3 \Rightarrow Quartos \geq 3

Min-Banheiros \geq 3 \Rightarrow Banheiros \geq 2

Portanto, as condições Preço \geq 100.000 e as restrições de intervalo nos atributos de requisito Quartos Mínimos e Banheiros Mínimos são substituídas pelas dos atributos de produto Quartos e Banheiros. Na próxima iteração, verifica-se que nenhuma outra condição pode ser adicionada aos requisitos do usuário.

2. Esses requisitos expandidos são usados para construir uma consulta de banco de dados na forma normal conjuntiva. Isso representa uma consulta de seleção de banco de dados tradicional, que calcula a interseção das seguintes restrições no catálogo de produtos:

(Quartos \geq 3) \wedge (Banheiros \geq 2) \wedge (Preço \geq 100.000) \wedge (CEP=10547)

Observe que a abordagem essencialmente mapeia todas as restrições de atributos do cliente e restrições de atributos de requisitos para restrições no domínio do produto.

3. Esta consulta de seleção é então usada para recuperar as instâncias no catálogo que são relevantes aos requisitos do usuário.

Vale ressaltar que a maioria dos sistemas baseados em restrições permite a especificação de todos os requisitos do usuário ou outros atributos (por exemplo, preferências, informações demográficas) durante a própria sessão. Em outras palavras, as informações especificadas normalmente não são persistentes; se um usuário diferente especificar a mesma entrada, obterá exatamente o mesmo resultado. Essa característica é comum à maioria dos sistemas baseados em conhecimento. A Seção 5.4 discutirá alguns avanços recentes na personalização persistente de sistemas baseados em conhecimento.

A lista resultante de itens que satisfazem as restrições é então apresentada ao usuário. A metodologia de classificação dos itens será discutida posteriormente nesta seção. O usuário pode então modificar ainda mais seus requisitos para obter recomendações mais refinadas. O processo geral de exploração e refinamento frequentemente leva o cliente a descobrir recomendações que, de outra forma, não teria conseguido obter sozinho.

5.2.2 Abordagem de interação

A interação entre o usuário e o sistema de recomendação geralmente ocorre em três fases.

1. Uma interface interativa é usada pelo usuário para especificar suas preferências iniciais. Uma abordagem comum é usar um formulário no estilo web, no qual os valores desejados dos atributos podem ser inseridos. Um exemplo de uma interface hipotética para compra de imóvel, que usaremos como exemplo prático, é fornecido na Figura 5.2. Alternativamente, o usuário pode ser questionado sobre uma série de perguntas para obter suas preferências iniciais. Por exemplo, o site de recomendação de carros Edmunds.com apresenta uma série de interfaces para que os usuários especifiquem suas preferências sobre os recursos específicos que desejam. As respostas às perguntas na primeira interface podem afetar as perguntas na próxima interface.
2. O usuário recebe uma lista classificada de itens correspondentes. Normalmente, é fornecida uma explicação para o motivo do retorno dos itens. Em alguns casos, nenhum item pode corresponder aos requisitos do usuário. Nesses casos, possíveis flexibilizações dos requisitos podem ser sugeridas. Por exemplo, na Figura 5.3, nenhum resultado é retornado pela consulta, e possíveis flexibilizações são sugeridas. Nos casos em que muitos itens são retornados, sugestões para possíveis restrições (requisitos do usuário) são incluídas. Por exemplo, na Figura 5.4, muitos resultados são retornados. Sugere-se que possíveis restrições sejam adicionadas à consulta.
3. O usuário então refina seus requisitos com base nos resultados retornados. Esse refinamento pode assumir a forma da adição de novos requisitos ou da remoção de alguns deles. Por exemplo, quando um conjunto vazio é retornado, fica evidente que alguns dos requisitos precisam ser flexibilizados. Métodos de satisfação de restrições são usados para identificar possíveis conjuntos de restrições candidatas que podem precisar ser flexibilizados. Portanto, o sistema geralmente auxilia o usuário a fazer suas modificações de forma mais inteligente e eficiente.

Assim, a abordagem geral utiliza um ciclo de feedback iterativo para auxiliar os usuários na tomada de decisões significativas. É crucial projetar um sistema que possa orientar o usuário em direção a requisitos que aumentem sua conscientização sobre as opções disponíveis.

Há vários aspectos dessa interação nos quais a computação explícita é necessária para auxiliar o usuário. Por exemplo, um usuário normalmente não consegue especificar os valores desejados para todos os atributos do produto. Por exemplo, em nosso exemplo de compra de imóvel, o usuário pode especificar restrições apenas quanto ao número de quartos e não especificar nenhuma restrição quanto ao preço. Várias soluções são possíveis neste cenário:

1. O sistema pode deixar os outros atributos sem restrições e recuperar os resultados com base apenas nas restrições especificadas. Por exemplo, todas as faixas possíveis de preços podem ser consideradas para fornecer o primeiro conjunto de respostas ao usuário. Embora essa possa ser a escolha mais razoável, quando a consulta do usuário é bem formulada, pode não ser uma solução eficaz em casos em que o número de respostas é grande.
2. Em alguns casos, valores padrão podem ser sugeridos ao usuário para fins de orientação. Os valores padrão podem ser usados apenas para orientar o usuário na seleção de valores ou podem ser incluídos na consulta caso o usuário não selecione nenhum valor (incluindo o padrão) para aquele atributo. Pode-se argumentar que incluir um valor padrão na consulta (sem especificação explícita) pode levar a um viés significativo no sistema de recomendação, especialmente quando os padrões não são muito bem pesquisados. Em geral,

**EXAMPLE OF HYPOTHETICAL CONSTRAINT-BASED INTERFACE
FOR HOME BUYING (constraint-example.com)**

[ENTRY POINT]

I WOULD LIKE TO BUY A HOUSE SATISFYING THE FOLLOWING REQUIREMENTS:

MIN. BR MAX. BR MIN. BATH MAX. BATH

MIN. PRICE MAX. PRICE HOME STYLE ZIP CODE

SUBMIT SEARCH

Figura 5.2: Um exemplo hipotético de uma interface de usuário inicial para um recomendador baseado em restrições (constraint-example.com)

Os valores padrão devem ser usados apenas como sugestão para o usuário. Isso ocorre porque o objetivo principal dos padrões deve ser guiar o usuário em direção a valores naturais, em vez de substituir opções não especificadas.

Como os valores padrão são determinados? Na maioria dos casos, é necessário escolher os valores padrão de acordo com o domínio. Além disso, alguns valores padrão podem ser afetados por outros.

Por exemplo, a potência de um modelo de carro selecionado pode frequentemente refletir a eficiência de combustível desejada. As bases de conhecimento precisam armazenar explicitamente os dados sobre esses valores padrão. Em alguns casos, onde os dados históricos das sessões do usuário estão disponíveis, é possível aprender os valores padrão. Para os vários usuários, seus valores de atributos especificados nas sessões de consulta podem estar disponíveis, incluindo os valores ausentes. Os valores médios em várias sessões podem ser usados como padrões. Considere uma sessão de consulta iniciada por Alice para comprar carros. Inicialmente, seus padrões são calculados com base nos valores médios em sessões históricas. No entanto, se ela especificar a potência desejada do carro, a interface ajusta automaticamente seu valor padrão de eficiência de combustível. Esse novo valor padrão é baseado na média de eficiência de combustível dos carros, que foram especificados em sessões históricas para carros com potência semelhante.

Em alguns casos, o sistema pode ajustar automaticamente os valores padrão com base em restrições de viabilidade em relação à base de conhecimento. À medida que os usuários especificam cada vez mais valores na interface, a média só pode ser calculada para as sessões dentro da vizinhança da especificação atual.

Após a consulta ser emitida, o sistema fornece uma lista ordenada de possíveis correspondências do catálogo. Portanto, é importante poder classificar as correspondências de forma significativa e também fornecer explicações para os resultados recomendados, se necessário. Nos casos em que o conjunto de correspondências retornado for muito pequeno ou muito grande, orientações adicionais podem ser fornecidas ao usuário sobre como flexibilizar ou restringir os requisitos. Vale ressaltar que o fornecimento de explicações também é uma maneira inteligente de orientar o usuário em direção a refinamentos de consulta mais significativos. A seguir, discutiremos esses vários aspectos da orientação interativa ao usuário.

EXAMPLE OF HYPOTHETICAL CONSTRAINT-BASED INTERFACE FOR HOME BUYING (constraint-example.com)

[CONSTRAINT MODIFICATION INTERFACE]



YOU SPECIFIED THE FOLLOWING REQUIREMENTS (CURRENT VALUES IN BRACKETS):

MIN. BR (5)	MAX. BR	MIN. BATH	MAX. BATH (1)
MIN. PRICE	MAX. PRICE (\$70K)	HOME STYLE	ZIP CODE (10547)

SUBMIT MODIFICATION GO BACK TO ENTRY POINT

YOUR QUERY RETURNED 0 RESULTS. MODIFY YOUR SEARCH ACCORDING TO THE SUGGESTIONS BELOW:

- EITHER REDUCE MIN. BR FROM 5 OR INCREASE MAX. PRICE FROM \$70K.
- EITHER REDUCE MIN. BR FROM 5 OR INCREASE MAX. BATH FROM 1.
- EITHER INCREASE MAX. PRICE OR CHANGE ZIP CODE FROM 10547.

MORE DETAILS

Figura 5.3: Um exemplo hipotético de uma interface de usuário para lidar com resultados de consulta vazios em um recomendador baseado em restrições (constraint-example.com)

5.2.3 Classificação dos itens correspondentes

Existem vários métodos naturais para classificar os itens de acordo com as necessidades do usuário.

A abordagem mais simples é permitir que o usuário especifique um único atributo numérico com base no qual classificar os itens. Por exemplo, no aplicativo de compra de imóvel, o sistema pode oferecer ao usuário a opção de classificar os itens com base em (qualquer um dos seguintes) preço do imóvel, número de quartos ou distância de um determinado CEP. Essa abordagem é, de fato, utilizada em muitas interfaces comerciais.

Usar um único atributo tem a desvantagem de que a importância de outros atributos é desconsiderada. Uma abordagem comum é usar funções de utilidade para classificar os itens correspondentes. Seja $V = (v_1 \dots v_d)$ o vetor de valores que definem os atributos dos produtos correspondentes. Portanto, a dimensionalidade do espaço de conteúdo é d . As funções de utilidade podem ser definidas como funções ponderadas das utilidades de atributos individuais. Cada atributo tem um peso w_j atribuído a ele e tem uma contribuição definida pela função $f_j(v_j)$ dependendo do valor v_j do atributo correspondente. Então, a utilidade $U(V)$ do item correspondente é dada por:

$$U(V) = \sum_{j=1}^d w_j \cdot f_j(v_j)$$
(5.1)

Claramente, é necessário instanciar os valores de w_j e $f_j(\cdot)$ para aprender a função de utilidade. O projeto de funções de utilidade eficazes frequentemente requer conhecimento específico do domínio ou dados de aprendizado de interações anteriores do usuário. Por exemplo, quando v_j é numérico, pode-se assumir que a função $f_j(v_j)$ é linear em v_j e, então, aprender os coeficientes da função linear.

EXAMPLE OF HYPOTHETICAL CONSTRAINT-BASED INTERFACE FOR HOME BUYING (constraint-example.com)

[CONSTRAINT MODIFICATION INTERFACE]



YOU SPECIFIED THE FOLLOWING REQUIREMENTS (CURRENT VALUES IN BRACKETS):

MIN. BR (2)	MAX. BR	MIN. BATH (1)	MAX. BATH (3)
MIN. PRICE (100K)	MAX. PRICE	HOME STYLE (CAPE)	ZIP CODE (10547)

SUBMIT MODIFICATION **GO BACK TO ENTRY POINT**

YOUR QUERY RETURNED **178 RESULTS**. MODIFY YOUR SEARCH ACCORDING TO THE SUGGESTIONS BELOW TO REDUCE MATCHES:

- ADD A VALUE FOR MAX. BR.
- ADD A VALUE FOR MAX. PRICE.
- CHANGE HOME STYLE FROM CAPE TO COLONIAL.

MORE DETAILS

Figura 5.4: Um exemplo hipotético de uma interface de usuário para lidar com muitos resultados de consulta em um recomendador baseado em restrições (constraint-example.com)

função, bem como w_j , obtendo feedback de vários usuários. Normalmente, os dados de treinamento são obtidos de alguns usuários que recebem a tarefa de classificar alguns itens de amostra. Essas classificações são então usadas para aprender o modelo mencionado acima com o uso de modelos de regressão. Essa abordagem está relacionada à metodologia de análise conjunta [155, 531]. A análise conjunta define métodos estatísticos para o estudo formal de como as pessoas valorizam os diferentes atributos que compõem um produto ou serviço individual. As notas bibliográficas contêm indicadores de alguns métodos comumente usados para o projeto de funções de utilidade.

5.2.4 Lidando com resultados inaceitáveis ou conjuntos vazios

Em muitos casos, uma consulta específica pode retornar um conjunto vazio de resultados. Em outros casos, o conjunto de resultados retornado pode não ser grande o suficiente para atender aos requisitos do usuário. Nesses casos, o usuário tem duas opções. Se for considerado que não existe uma maneira direta de reparar as restrições, ele pode optar por recomeçar do ponto de entrada. Alternativamente, ele pode decidir alterar ou flexibilizar as restrições para a próxima iteração interativa.

Como o usuário pode fazer uma escolha significativa sobre se deve ou não relaxar as restrições e de que maneira? Nesses casos, muitas vezes é útil fornecer ao usuário alguma orientação sobre como relaxar os requisitos atuais. Essas propostas são chamadas de propostas de reparo. A ideia é ser capaz de determinar conjuntos mínimos de restrições inconsistentes e apresentá-los ao usuário. É mais fácil para o usuário assimilar conjuntos mínimos de restrições inconsistentes e encontrar maneiras de relaxar uma ou mais restrições nesses conjuntos. Considere o exemplo da compra de uma casa, em que pode ser descoberto que o usuário especificou muitos requisitos, mas o único par de requisitos mutuamente inconsistente é Preço Máximo < 100.000 e Quartos Mínimos > 5.

Se esse par de restrições for apresentado ao usuário, ele poderá entender que precisa aumentar o preço máximo que está disposto a pagar ou se contentar com um número menor de quartos. Uma maneira ingênua de encontrar o conjunto mínimo de restrições inconsistentes é realizar uma busca de baixo para cima de todas as combinações de requisitos do usuário e determinar os menores conjuntos que são inviáveis. Em muitas interfaces interativas, o usuário pode especificar apenas um pequeno número de requisitos (digamos, de 5 a 10), e o número de restrições envolvendo esses atributos (no conhecimento do domínio) também pode ser pequeno. Nesses casos, a exploração exaustiva de todas as possibilidades não é uma abordagem irracional. Por sua própria natureza, a especificação interativa de requisitos frequentemente resulta na especificação de um número relativamente pequeno de restrições. É incomum que um usuário especifique 100 requisitos diferentes em uma consulta interativa. Em alguns casos, no entanto, quando o número de requisitos especificados pelo usuário é grande e o conhecimento do domínio é significativo, essa exploração exaustiva de baixo para cima pode não ser uma opção viável. Métodos mais sofisticados, como QUICKXPLAIN e MINRELAX, também foram propostos, os quais podem ser usados para descoberta rápida de pequenos conjuntos conflitantes e relaxamentos mínimos [198, 273, 274, 289, 419].

A maioria desses métodos utiliza princípios semelhantes; pequenos conjuntos de restrições violadoras são determinados e os relaxamentos mais apropriados são sugeridos com base em alguns critérios predefinidos. Em aplicações reais, no entanto, às vezes é difícil sugerir critérios concretos para o relaxamento de restrições. Portanto, uma alternativa simples é apresentar ao usuário pequenos conjuntos de restrições inconsistentes, o que muitas vezes pode fornecer intuição suficiente para o usuário formular restrições modificadas.

5.2.5 Adicionando Restrições

Em alguns casos, o número de resultados retornados pode ser muito grande, e o usuário pode precisar sugerir possíveis restrições a serem adicionadas à consulta. Nesses casos, uma variedade de métodos pode ser usada para sugerir restrições ao usuário, juntamente com possíveis valores padrão. Os atributos para tais restrições são frequentemente escolhidos pela mineração de logs históricos de sessão. Os logs históricos de sessão podem ser definidos para todos os usuários ou para o usuário específico em questão. Este último fornece resultados mais personalizados, mas pode frequentemente estar indisponível para itens comprados com pouca frequência (por exemplo, carros ou casas). Vale ressaltar que os sistemas baseados em conhecimento são geralmente projetados para não usar tais informações persistentes e históricas precisamente porque são projetados para funcionar em configurações de inicialização a frio; no entanto, tais informações podem frequentemente ser muito úteis para melhorar a experiência do usuário quando estão disponíveis.

Como os dados históricos de sessão podem ser usados? A ideia é selecionar restrições que sejam populares. Por exemplo, se um usuário especificou as restrições em um conjunto de atributos de item, outras sessões contendo um ou mais desses atributos são identificadas. Por exemplo, se um usuário especificou restrições sobre o número de quartos e o preço, sessões anteriores contendo restrições sobre o quarto e o preço são identificadas. Em particular, as k principais sessões de vizinho mais próximo em termos do número de atributos comuns são identificadas. Se for determinado que a restrição mais popular entre essas k principais sessões é sobre o número de banheiros, então esse atributo é sugerido pela interface como um candidato para adicionar restrições adicionais.

Em muitos casos, a ordem temporal na qual os usuários especificaram restrições no passado está disponível. Nesses casos, também é possível usar a ordem em que o cliente especificou as restrições, tratando-as como um conjunto ordenado, em vez de um conjunto não ordenado [389]. Uma maneira simples de atingir esse objetivo é determinar o atributo mais frequente que segue o conjunto especificado atual de atributos restritos em sessões anteriores.

A mineração de padrões sequenciais pode ser usada para determinar tais atributos frequentes. Os trabalhos

Em [389, 390], o problema de aprendizagem sequencial é modelado como um Processo de Decisão Markoviano (MDP) e técnicas de aprendizagem por reforço são utilizadas para mensurar o impacto de diversas escolhas. As restrições podem ser sugeridas com base em sua seletividade no banco de dados ou com base na especificação média do usuário em sessões anteriores.

5.3 Recomendadores baseados em casos

Em sistemas de recomendação baseados em casos, métricas de similaridade são usadas para recuperar exemplos semelhantes aos alvos (ou casos) especificados. Por exemplo, no exemplo imobiliário da Tabela 5.2, o usuário pode especificar uma localidade, o número de quartos e um preço desejado para especificar um conjunto de atributos alvo. Ao contrário dos sistemas baseados em restrições, nenhuma restrição rígida (por exemplo, valores mínimos ou máximos) é imposta a esses atributos. Também é possível projetar uma interface de consulta inicial na qual exemplos de itens relevantes sejam usados como alvos. No entanto, é mais natural especificar as propriedades desejadas na interface de consulta inicial. Uma função de similaridade é usada para recuperar os exemplos mais semelhantes ao alvo especificado pelo usuário. Por exemplo, se nenhuma casa for encontrada especificando exatamente os requisitos do usuário, a função de similaridade é usada para recuperar e classificar os itens que são o mais semelhantes possível à consulta do usuário. Portanto, diferentemente dos sistemas de recomendação baseados em restrições, o problema de recuperar conjuntos vazios não é um problema nos sistemas de recomendação baseados em casos.

Existem também diferenças substanciais entre um recomendador baseado em restrições e um recomendador baseado em casos em termos de como os resultados são refinados. Sistemas baseados em restrições utilizam relaxamento, modificação e estreitamento de requisitos para refinar os resultados. Os primeiros sistemas baseados em casos defendiam a modificação repetida dos requisitos de consulta do usuário até que uma solução adequada pudesse ser encontrada. Posteriormente, o método de crítica foi desenvolvido.

A ideia geral da crítica é que os usuários podem selecionar um ou mais resultados recuperados e especificar outras consultas do seguinte formato:

“Dê-me mais itens como X, mas eles são diferentes no(s) atributo(s) Y de acordo com a orientação Z.”

Existe uma variação significativa em termos de se um ou mais atributos são selecionados para modificação e como a orientação para a modificação dos atributos é especificada. O principal objetivo da análise crítica é oferecer suporte à navegação interativa no espaço de itens, onde o usuário gradualmente toma conhecimento de outras opções disponíveis por meio dos exemplos recuperados. A navegação interativa no espaço de itens tem a vantagem de ser um processo de aprendizagem para o usuário durante o processo de formulação iterativa da consulta. Muitas vezes, é possível que, por meio da exploração repetida e interativa, o usuário consiga chegar a itens que, de outra forma, não teriam sido alcançados logo no início.

Por exemplo, considere o exemplo de compra de imóvel da Tabela 5.2. O usuário pode ter especificado inicialmente um preço desejado, o número de quartos e uma localidade desejada. Alternativamente, o usuário pode especificar um endereço de destino para fornecer um exemplo de uma possível casa na qual ele possa estar interessado. Um exemplo de uma interface inicial na qual o usuário pode especificar o destino de duas maneiras diferentes é ilustrado na Figura 5.5. A parte superior da interface ilustra a especificação das características do destino, enquanto a parte inferior da interface ilustra a especificação de um endereço de destino. Esta última abordagem é útil em domínios onde os usuários têm maior dificuldade em especificar características tecnicamente enigmáticas. Um exemplo pode ser o caso de câmeras digitais, onde é mais difícil especificar todas as características técnicas exatamente para um não especialista em fotografia. Portanto, um usuário pode especificar a câmera de seu amigo como

**EXAMPLE OF HYPOTHETICAL CASE-BASED RECOMMENDATION
INTERFACE FOR HOME BUYING (critique-example.com)**

[ENTRY POINT]



I WOULD LIKE TO BUY A HOUSE SIMILAR TO ONE WITH THE FOLLOWING FEATURES:

NUMBER OF BR	▼	NUMBER OF BATH	▼	HOME STYLE	▼
PRICE RANGE	▼	ZIP CODE			
SUBMIT SEARCH					

I WOULD LIKE TO BUY AN HOUSE JUST LIKE THE ONE AT THE FOLLOWING ADDRESS:

812 SCENIC DRIVE	MOHEGAN LAKE	NY	▼
SUBMIT SEARCH			

Figura 5.5: Um exemplo hipotético de uma interface de usuário inicial em um recomendador baseado em casos (critique-example.com)

o caso-alvo, em vez de especificar todos os recursos técnicos. Observe que esta interface foi projetada hipoteticamente apenas para fins ilustrativos e não se baseia em um sistema de recomendação real.

O sistema utiliza a consulta alvo em conjunto com funções de similaridade ou utilidade para recuperar resultados correspondentes. Eventualmente, ao recuperar os resultados, o usuário pode decidir gostar de uma casa específica, exceto que suas especificações contêm características (por exemplo, uma casa colonial) das quais ele não gosta particularmente. Nesse ponto, o usuário pode utilizar esse exemplo como uma âncora e especificar os atributos específicos que deseja que sejam diferentes. Observe que o motivo pelo qual o usuário consegue fazer esse segundo conjunto de especificações de consulta criticada é que agora ele tem um exemplo concreto para trabalhar, do qual não tinha conhecimento anteriormente. As interfaces para crítica podem ser definidas de diversas maneiras diferentes e são discutidas em detalhes na seção 5.3.2. O sistema então emite uma nova consulta com o alvo modificado e com um conjunto reduzido de candidatos, que eram os resultados da consulta anterior. Em muitos casos, o efeito é simplesmente podar os resultados da pesquisa de casos que não são considerados relevantes, em vez de fornecer uma reclassificação dos resultados retornados.

Portanto, diferentemente de sistemas baseados em restrições, o número de respostas retornadas em iterações baseadas em casos geralmente reduz de um ciclo para o próximo. No entanto, também é possível projetar sistemas baseados em casos nos quais os candidatos nem sempre são reduzidos de uma iteração para a próxima, expandindo o escopo de cada consulta para todo o banco de dados, em vez do conjunto de resultados candidatos recuperados no momento. Esse tipo de escolha de projeto tem suas próprias desvantagens. Por exemplo, ao expandir o escopo de cada consulta, o usuário poderá navegar para um resultado final mais distante da consulta atual. Por outro lado, também é possível que os resultados se tornem cada vez mais irrelevantes em iterações posteriores. Para os fins deste capítulo, assumimos que os candidatos retornados sempre diminuem de uma iteração para a próxima.

Por meio de críticas repetidas, o usuário pode, às vezes, chegar a um resultado final bastante diferente da especificação da consulta inicial. Afinal, muitas vezes é difícil para um usuário articular todas as características desejadas logo no início. Por exemplo, o usuário pode não estar ciente de um preço aceitável para as características desejadas da casa no início do processo de consulta. Essa abordagem interativa preenche a lacuna entre sua compreensão inicial e a disponibilidade do item. É esse poder da navegação assistida que torna os métodos baseados em casos tão poderosos para aumentar a conscientização do usuário. Às vezes, também é possível que o usuário chegue a um conjunto vazio de candidatos por meio da redução repetida do conjunto de candidatos.

Tal sessão pode ser vista como infrutífera e, nesse caso, o usuário precisa reiniciar do zero no ponto de entrada. Observe que isso é diferente dos sistemas baseados em restrições, onde o usuário também tem a opção de relaxar seu conjunto atual de requisitos para ampliar o conjunto de resultados. A razão para essa diferença é que os sistemas baseados em casos geralmente reduzem o número de candidatos de um ciclo para o outro, enquanto os sistemas baseados em restrições não o fazem.

Para que um sistema de recomendação baseado em casos funcione eficazmente, existem dois fatores cruciais aspectos do sistema que devem ser projetados de forma eficaz:

1. Métricas de similaridade: O design eficaz de métricas de similaridade é muito importante em sistemas baseados em casos para obter resultados relevantes. A importância de vários atributos deve ser devidamente incorporada à função de similaridade para que o sistema funcione de forma eficaz.
2. Métodos de crítica: A exploração interativa do espaço do item é apoiada pelo uso de métodos de crítica. Uma variedade de métodos de crítica está disponível para apoiar diferentes objetivos de exploração.

Nesta seção, discutiremos esses dois aspectos importantes do design do sistema de recomendação baseado em casos.

5.3.1 Métricas de similaridade

O design adequado de métricas de similaridade é essencial para recuperar itens significativos em resposta a uma consulta específica. Os primeiros sistemas FindMe [121] ordenavam os atributos em nível decrescente de importância e primeiro classificavam pelo critério mais importante, depois pelo próximo mais importante e assim por diante. Por exemplo, no sistema de recomendação de restaurantes Entree, a primeira classificação pode ser baseada no tipo de culinária, a segunda no preço e assim por diante. Embora essa abordagem seja eficiente, seu uso pode não ser eficaz para todos os domínios. Em geral, é desejável desenvolver uma função de similaridade de forma fechada cujos parâmetros possam ser definidos por especialistas do domínio ou ajustados por um processo de aprendizagem.

Considere uma aplicação na qual o produto é descrito por d atributos. Gostaríamos de determinar os valores de similaridade entre dois vetores de atributos parciais definidos em um subconjunto S do universo de d atributos (ou seja, $|S| = s \leq d$). Sejam $X = (x_1 \dots x_d)$ e $T = (t_1 \dots t_d)$ dois vetores d -dimensionais, que podem ser parcialmente especificados. Aqui, T representa o alvo. Assume-se que pelo menos o subconjunto de atributos $S \subseteq \{1 \dots d\}$ seja especificado em ambos os vetores. Observe que estamos usando vetores de atributos parciais porque tais consultas são frequentemente definidas apenas em um pequeno subconjunto de atributos especificados pelo usuário. Por exemplo, no exemplo imobiliário mencionado anteriormente, o usuário pode especificar apenas um pequeno conjunto de recursos de consulta, como o número de quartos ou banheiros. Então, a função de similaridade $f(T, X)$ entre os dois conjuntos de vetores é definida da seguinte forma:

$$f(\bar{T}, \bar{X}) = \frac{\sum_{i=1}^S w_i \cdot \text{Sim}(t_i, x_i)}{\sum_{i=1}^S w_i} \quad (5.2)$$

Aqui, $\text{Sim}(t_i, x_i)$ representa a similaridade entre os valores x_i e t_i . O peso w_i representa o peso do i -ésimo atributo e regula a importância relativa desse atributo. Como as funções de similaridade $\text{Sim}(t_i, x_i)$ e a importância do atributo w_i podem ser aprendidas?

Primeiro, discutiremos a determinação da função de similaridade $\text{Sim}(t_i, x_i)$. Observe que esses atributos podem ser quantitativos ou categóricos, o que aumenta ainda mais a heterogeneidade e a complexidade de tal sistema. Além disso, os atributos podem ser simétricos ou assimétricos em termos de valores maiores ou menores [558]. Por exemplo, considere o atributo preço no exemplo de compra de imóvel da Tabela 5.2. Se um produto devolvido tiver um preço menor do que o valor-alvo, ele será mais facilmente aceitável do que um caso em que o produto devolvido tenha um preço maior do que o valor-alvo. O nível preciso de assimetria pode ser diferente para diferentes atributos. Por exemplo, para um atributo, como a resolução da câmera, o usuário pode achar resoluções maiores mais desejáveis, mas a preferência pode não ser tão forte quanto no caso do preço. Outros atributos podem ser completamente simétricos, caso em que o usuário desejará que o valor do atributo estivesse exatamente no valor-alvo t_i . Um exemplo de uma métrica simétrica é o seguinte:

$$\text{Sim}(t_i, x_i) = 1 - \frac{|t_i - x_i|}{\max_i - \min_i} \quad (5.3)$$

Aqui, \max_i e \min_i representam os valores máximos ou mínimos possíveis do atributo i . Alternativamente, pode-se usar o desvio padrão \bar{y}_i (em dados históricos) para definir a função de similaridade:

$$\text{Sim}(t_i, x_i) = \max(0, 1 - \frac{|t_i - x_i|}{3 \cdot \bar{y}_i}) \quad (5.4)$$

Observe que, no caso da métrica simétrica, a similaridade é inteiramente definida pela diferença entre os dois atributos. No caso de um atributo assimétrico, pode-se adicionar uma recompensa assimétrica adicional, que entra em vigor dependendo se o valor do atributo alvo é menor ou maior. Para o caso de atributos em que valores maiores são melhores, um exemplo de uma possível função de similaridade é o seguinte:

$$\text{Sim}(t_i, x_i) = 1 - \bar{y}_i \cdot I(x_i > t_i) \cdot \frac{\max_i - |t_i - x_i|}{\max_i - \min_i} \quad (5.5)$$

$I(x_i > t_i)$ é uma função indicadora que assume 1 se $x_i > t_i$, e 0 caso contrário.

Recompensa assimétrica

Aqui, $\bar{y}_i > 0$ é um parâmetro definido pelo usuário, e $I(x_i > t_i)$ é uma função indicadora que assume o valor 1 se $x_i > t_i$, e 0 caso contrário. Observe que a recompensa só entra em vigor quando o valor do atributo x_i (por exemplo, resolução da câmera) é maior que o valor alvo t_i . Para casos em que valores menores são melhores (por exemplo, preço), a função de recompensa é semelhante, exceto que valores menores são recompensados pela função indicadora:

$$\text{Sim}(t_i, x_i) = 1 - \bar{y}_i \cdot I(x_i < t_i) \cdot \frac{\max_i - |t_i - x_i|}{\max_i - \min_i} \quad (5.6)$$

$I(x_i < t_i)$ é uma função indicadora que assume 1 se $x_i < t_i$, e 0 caso contrário.

Recompensa assimétrica

Os valores de \bar{y}_i são escolhidos de forma altamente específica ao domínio. Para valores de $\bar{y}_i > 1$, a “similaridade” na verdade aumenta com a maior distância do alvo. Nesses casos, é útil

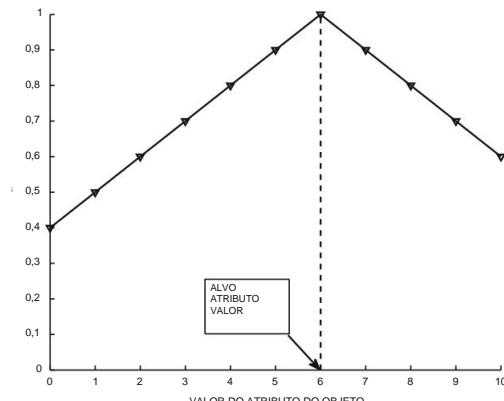
pensar em $\text{Sim}(t_i, x_i)$ como uma função de utilidade em vez de uma função de similaridade. Por exemplo, no caso do preço, sempre se preferiria um preço menor a um preço maior, embora o preço-alvo possa definir um ponto de inflexão na intensidade com que se prefere um preço menor a um preço maior. Quando o valor de γ_i é exatamente 1,0, isso implica que não se importa com mudanças adicionais em relação ao valor-alvo em uma das direções. Um exemplo pode ser o caso da resolução da câmera, onde não se importa com resoluções além de um certo ponto. Quando $\gamma_i \neq (0, 1)$, isso implica que o usuário prefere um valor no alvo sobre todos os outros valores, mas pode ter preferências assimétricas em ambos os lados do alvo. Por exemplo, a preferência de um usuário por potência pode aumentar fortemente até o alvo, e ele também pode ter uma leve aversão a uma potência maior que o alvo devido ao maior consumo de combustível. Esses exemplos sugerem que não há maneiras simples de predefinir tais métricas de similaridade; muito trabalho precisa ser feito pelo especialista no domínio.

Exemplos de funções de similaridade simétricas e assimétricas são ilustrados na Figura 5.6.

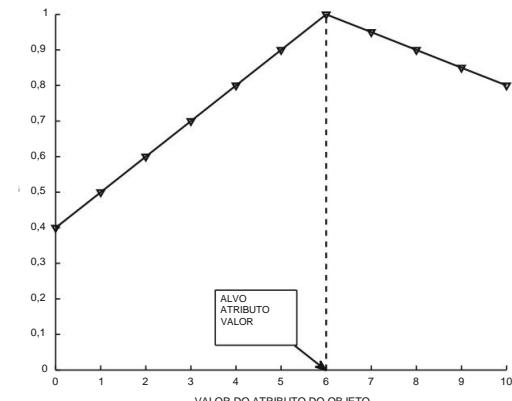
O intervalo do domínio é $[0, 10]$ e um valor alvo de 6 é usado. Uma função de similaridade simétrica é mostrada na Figura 5.6(a), onde a similaridade é linearmente dependente da distância do alvo. No entanto, no exemplo de potência discutido acima, a função de similaridade assimétrica da Figura 5.6(b) pode ser mais apropriada, onde $\gamma_i = 0,5$. Para um atributo como a resolução da câmera, pode-se decidir não alocar nenhuma utilidade além do alvo do usuário, como resultado da qual a função de similaridade pode ser plana além desse ponto. Tal caso é ilustrado na Figura 5.6(c), onde γ_i é definido como 1. Finalmente, no caso do preço, valores menores são recompensados, embora o preço alvo do usuário possa definir um ponto de inflexão na função de utilidade. Este caso é ilustrado na Figura 5.6(d), onde o valor de γ_i é definido como 1,3, com recompensas sendo concedidas por não atingir o alvo. Este caso em particular é digno de nota, pois a "similaridade" aumenta com a distância do alvo, desde que o valor seja o menor possível. Nesses casos, a interpretação de utilidade dessas funções faz muito mais sentido do que a interpretação de similaridade. Nessa interpretação, os valores dos atributos alvo representam apenas os principais pontos de inflexão da função de utilidade.

No caso de dados categóricos, a determinação de valores de similaridade costuma ser mais desafiadora. Normalmente, hierarquias de domínio são construídas para determinar os valores de similaridade. Dois objetos mais próximos um do outro dentro do contexto de uma hierarquia de domínio podem ser considerados mais semelhantes. Essa hierarquia de domínio às vezes está disponível diretamente em fontes como o Sistema de Classificação Industrial Norte-Americano (NAICS) e, em outros casos, precisa ser construída diretamente à mão. Por exemplo, um atributo como o gênero do filme pode ser classificado hierarquicamente, conforme mostrado na Figura 5.7. Observe que gêneros relacionados tendem a estar mais próximos uns dos outros na hierarquia. Por exemplo, filmes para crianças são considerados tão diferentes daqueles para o público em geral que se bifurcam na raiz da taxonomia. Essa hierarquia pode ser usada pelo especialista do domínio para codificar manualmente as similaridades.

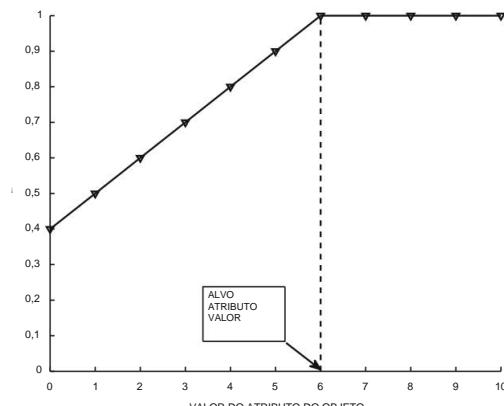
Em alguns casos, métodos de aprendizagem também podem ser usados para facilitar o cálculo de similaridade. Por exemplo, o feedback poderia ser obtido dos usuários sobre pares de gêneros, e métodos de aprendizagem poderiam ser usados para aprender a similaridade entre pares de itens [18]. A abordagem de aprendizagem mais ampla também pode ser usada para determinar outros parâmetros da função de similaridade, como o valor de γ_i nas Equações 5.5 e 5.6. Vale ressaltar que a forma específica da função de similaridade pode ser diferente daquela nas Equações 5.5 e 5.6, dependendo do domínio de dados. É aqui que o especialista no domínio tem que investir uma quantidade significativa de tempo para decidir como modelar a configuração específica do problema. Esse investimento é uma parte inerente do esforço específico do domínio que os sistemas de recomendação baseados em conhecimento exigem e também derivam seu nome.



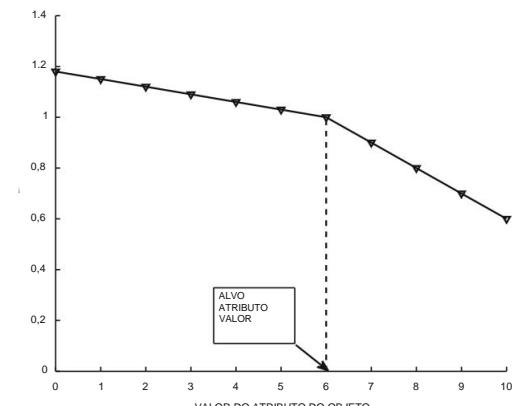
(a) Simétrico ($\bar{y}_i = 0$)
(penalidade pela distância absoluta)



(b) Assimétrico ($\bar{y}_i = 0,5$)
(penalidade mais suave por ultrapassagem)



(a) Assimétrico ($\bar{y}_i = 1,0$) (sem
penalidade por ultrapassagem)



(b) Assimétrico ($\bar{y}_i = 1,3$)
(menos é sempre melhor)

Figura 5.6: Exemplos de diferentes tipos de similaridade simétrica e assimétrica

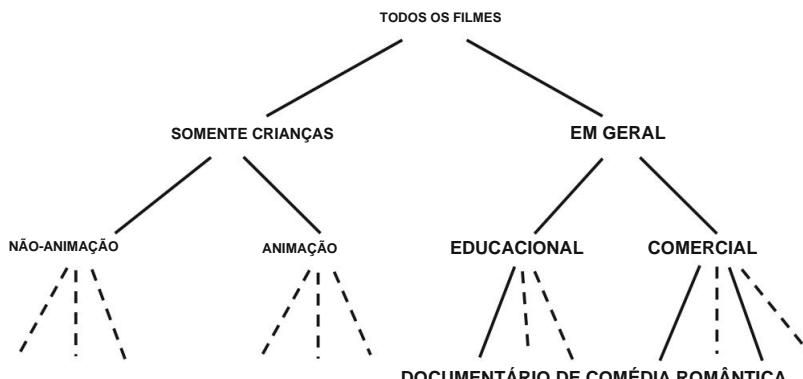


Figura 5.7: Um exemplo de classificação hierárquica de gêneros cinematográficos

Uma segunda questão no projeto de funções de similaridade é a determinação da importância relativa de vários atributos. A importância relativa do i-ésimo atributo é regulada pelo parâmetro w_i na Equação 5.2. Uma possibilidade é que um especialista no domínio codifique manualmente os valores de w_i por meio de tentativa e erro. A outra possibilidade é aprender os valores de w_i com o feedback do usuário. Pares de objetos-alvo podem ser apresentados aos usuários, e os usuários podem ser solicitados a classificar o quão semelhantes esses objetos-alvo são. Esse feedback pode ser usado em conjunto com um modelo de regressão linear para determinar o valor de w_i . Modelos de regressão linear são discutidos em detalhes na seção 4.4.5 do Capítulo 4, e seu uso para o aprendizado de funções de similaridade é discutido em [18]. Vários outros resultados [97, 163, 563, 627] discutem métodos de aprendizado com feedback do usuário no contexto específico de sistemas de recomendação. Muitos desses métodos, como os de [627], mostram como a ponderação de recursos pode ser alcançada com o feedback do usuário. O trabalho em [563] solicita feedback do usuário em termos da ordenação relativa dos casos retornados e o utiliza para aprender os pesos relativos das características. Muitas vezes, é mais fácil para o usuário especificar ordenações relativas do que especificar valores de similaridade explícitos para pares de objetos.

5.3.1.1 Incorporando Diversidade na Computação de Similaridade

Como os sistemas baseados em casos utilizam atributos de itens para recuperar produtos semelhantes, eles enfrentam muitos dos mesmos desafios que os sistemas baseados em conteúdo para retornar resultados diversos. Em muitos casos, os resultados retornados pelos sistemas baseados em casos são todos muito semelhantes. O problema com a falta de diversidade é que, se um usuário não gostar do resultado mais bem classificado, muitas vezes não gostará dos outros resultados, que são todos muito semelhantes. Por exemplo, no aplicativo de compra de imóveis, é possível que o sistema de recomendação retorne unidades de condomínio do mesmo complexo sob a mesma administração. Claramente, esse cenário reduz a real escolha disponível para o usuário entre os resultados mais bem classificados.

Considere um cenário em que se deseja recuperar os k principais resultados correspondentes a um caso específico. Uma possibilidade é recuperar os $b \cdot k$ principais resultados (para $b > 1$) e, em seguida, selecionar aleatoriamente k itens dessa lista. Essa estratégia também é chamada de estratégia de seleção aleatória limitada. No entanto, essa estratégia não parece funcionar muito bem na prática.

Uma abordagem mais eficaz é a estratégia de seleção gananciosa limitada [560]. Nessa estratégia, começamos com os principais $b \cdot k$ casos semelhantes ao alvo e construímos incrementalmente um conjunto diverso de k instâncias a partir desses $b \cdot k$ casos. Portanto, começamos com o conjunto vazio R e o construímos incrementalmente adicionando instâncias do conjunto base de $b \cdot k$ casos. O primeiro passo é criar uma métrica de qualidade que combine similaridade e diversidade. Assuma, sem perda de generalidade, que a função de similaridade $f(\underline{X}, \underline{Y})$ sempre mapeia para um valor em $(0, 1)$. Então, a diversidade $D(\underline{X}, \underline{Y})$ pode ser vista como a distância entre \underline{X} e \underline{Y} :

$$D(\underline{X}, \underline{Y}) = \overline{f(\underline{X}, \underline{Y})} \quad (5.7)$$

Então, a diversidade média entre o candidato X e um conjunto R de casos atualmente selecionados é definida como a diversidade média entre X e os casos em R :

$$\text{Média}(\underline{X}, \underline{R}) = \frac{\overline{\overline{Y} \in R} D(\underline{X}, \underline{Y})}{|R|} \quad (5.8)$$

Então, para o alvo T , a qualidade geral $Q(T, X, R)$ é calculada da seguinte forma:

$$Q(\underline{T}, \underline{X}, \underline{R}) = f(\underline{T}, \underline{X}) \cdot \overline{\text{Média}}(\underline{X}, \underline{R}) \quad (5.9)$$

O caso X com a maior qualidade é adicionado incrementalmente ao conjunto R até que a cardinalidade do conjunto R seja k. Este conjunto é apresentado ao usuário. Consulte as notas bibliográficas para outras técnicas de aumento de diversidade utilizadas na literatura.

5.3.2 Métodos de Crítica

As críticas são motivadas pelo fato de que os usuários muitas vezes não conseguem declarar seus requisitos com exatidão na consulta inicial. Em alguns domínios complexos, eles podem até ter dificuldade em traduzir suas necessidades de forma semanticamente significativa para os valores de atributos no domínio do produto. Somente após visualizar os resultados de uma consulta é que um usuário pode perceber que deveria ter formulado sua consulta de forma um pouco diferente. As críticas são projetadas para fornecer aos usuários essa capacidade posteriormente.

Após a apresentação dos resultados aos usuários, o feedback normalmente é possibilitado por meio de críticas. Em muitos casos, as interfaces são projetadas para criticar o item correspondente mais semelhante, embora seja tecnicamente possível para o usuário criticar qualquer um dos itens na lista de k itens recuperada. Nas críticas, os usuários especificam solicitações de alteração em um ou mais atributos de um item dos quais possam gostar. Por exemplo, no aplicativo de compra de imóveis da Figura 5.2, o usuário pode gostar de uma casa específica, mas pode querer a casa em um local diferente ou com um quarto a mais. Portanto, o usuário pode especificar as alterações nas características de um dos itens de que gosta. O usuário pode especificar uma crítica direcional (por exemplo, "mais barato") ou uma crítica de substituição (por exemplo, "cor diferente"). Nesses casos, os exemplos que não satisfazem as críticas especificadas pelo usuário são eliminados e os exemplos semelhantes ao item preferido pelo usuário (mas que satisfazem a sequência atual de críticas) são recuperados. Quando múltiplas críticas são especificadas em ciclos de recomendação sequenciais, a preferência é dada às críticas mais recentes.

Em um determinado momento, o usuário pode especificar um único recurso ou uma combinação de recursos para modificação. Nesse contexto, as críticas são de três tipos diferentes, correspondendo a críticas simples, críticas compostas e críticas dinâmicas. Discutiremos cada um desses tipos de críticas nas seções a seguir.

5.3.2.1 Críticas Simples

Em uma crítica simples, o usuário especifica uma única alteração em uma das características de um item recomendado. Na Figura 5.8, usamos nosso cenário anterior baseado em casos ([critique-example.com](#)) para mostrar um exemplo de uma interface de crítica simples. Observe que o usuário pode especificar uma alteração em apenas uma das características da casa recomendada nessa interface. Frequentemente, em muitos sistemas, como os sistemas FindMe, uma interface mais conversacional é usada, onde os usuários especificam se desejam aumentar ou diminuir um valor de atributo específico em vez de modificar explicitamente um dos valores de atributo alvo. Isso é chamado de crítica direcional. Nesses casos, a lista de candidatos é simplesmente removida dos objetos para os quais o atributo criticado está do lado errado da preferência declarada pelo usuário. A vantagem dessa abordagem é que o usuário pode declarar sua preferência e navegar pelo espaço do produto sem precisar especificar ou alterar os valores dos atributos de forma precisa. Essa abordagem é particularmente importante em domínios onde os usuários podem não saber o valor exato do atributo a ser usado (por exemplo, a potência de um motor). Outra vantagem de uma crítica direcional é que ela tem um estilo conversacional simples, o que pode ser mais intuitivo e atraente para o usuário. Nos casos em que o usuário não considera o conjunto atual de resultados recuperados útil, ele também tem a opção de retornar ao ponto de entrada. Isso representa um ciclo infrutífero no processo de crítica.

EXAMPLE OF HYPOTHETICAL CASE-BASED RECOMMENDATION INTERFACE FOR HOME BUYING (critique-example.com)

[SIMPLE CRITIQUING INTERFACE]



YOU SPECIFIED THE FOLLOWING TARGET:
812 SCENIC DRIVE, MOHEGAN LAKE, NY

YOUR TOP RECOMMENDATION IS:
742 SCENIC DRIVE, MOHEGAN LAKE, NY

WE RECOMMEND THIS HOUSE BECAUSE: IT HAS SIMILAR BEDROOMS, BATHROOMS, LOCALITY, PRICE RANGE, AND HOME STYLE AS YOUR TARGET

I WOULD LIKE TO BUY A HOUSE SIMILAR TO THE TOP RECOMMENDATION BUT WITH ONE OF THE FOLLOWING CHANGES:

NUMBER OF BR	SUBMIT CHANGE	NUMBER OF BATH	SUBMIT CHANGE
PRICE RANGE	SUBMIT CHANGE	ZIP CODE	SUBMIT CHANGE
HOME STYLE	SUBMIT CHANGE	SEE OTHER RESULTS	
		GO BACK TO ENTRY POINT	

(a) Simple critiquing by directly modifying feature values

EXAMPLE OF HYPOTHETICAL CASE-BASED RECOMMENDATION INTERFACE FOR HOME BUYING (critique-example.com)

[SIMPLE CRITIQUING INTERFACE]



YOU SPECIFIED THE FOLLOWING TARGET:
812 SCENIC DRIVE, MOHEGAN LAKE, NY

YOUR TOP RECOMMENDATION IS:
742 SCENIC DRIVE, MOHEGAN LAKE, NY

WE RECOMMEND THIS HOUSE BECAUSE: IT HAS SIMILAR BEDROOMS, BATHROOMS, LOCALITY, PRICE RANGE, AND HOME STYLE AS YOUR TARGET

I WOULD LIKE TO BUY A HOUSE SIMILAR TO THE TOP RECOMMENDATION BUT WITH ONE OF THE FOLLOWING CHANGES:

NUMBER OF BR	MORE	LESS	NUMBER OF BATH	MORE	LESS
PRICE RANGE	MORE	LESS	EXPLORE NEARBY ZIP CODES		
EXPLORE RELATED STYLES			SEE OTHER RESULTS		GO BACK TO ENTRY POINT

(b) The conversational style of directional critiques

Figura 5.8: Exemplos hipotéticos de interfaces de usuário para crítica simples em um caso baseado recomendador (critique-example.com)

O principal problema com a abordagem de crítica simples é sua navegação trabalhosa. Se o produto recomendado contiver muitos recursos que precisam ser alterados, isso levará a uma cadeia mais longa de críticas subsequentes. Além disso, quando um dos recursos é alterado, o sistema de recomendação pode precisar alterar automaticamente pelo menos alguns dos valores dos outros recursos, dependendo da disponibilidade do item. Na maioria dos casos, é impossível manter os valores dos outros recursos em valores exatamente constantes em um determinado ciclo. Como resultado, quando o usuário altera alguns recursos para os valores desejados, ele pode perceber que os valores dos outros recursos não são mais aceitáveis. Quanto maior o número de ciclos de recomendação, menor o controle que o usuário terá sobre as alterações nos valores dos outros recursos que eram aceitáveis em iterações anteriores. Esse problema geralmente resulta da falta de compreensão do usuário sobre as compensações naturais no domínio do problema. Por exemplo, um usuário pode não entender a compensação entre potência e eficiência de combustível e tentar navegar para um carro com alta potência e também uma alta eficiência de combustível de 50 milhas por galão [121]. Este problema de infrutífero em longos ciclos de recomendação é discutido em detalhes em [423]. O principal problema em muitas interfaces de crítica é que o próximo conjunto de itens recomendados se baseia nos itens mais recentes sendo criticados, e não há como navegar de volta aos itens anteriores. Como resultado, um longo ciclo de críticas simples pode, às vezes, levar a uma conclusão infrutífera.

5.3.2.2 Críticas Compostas

Críticas compostas foram desenvolvidas para reduzir a duração dos ciclos de recomendação [414]. Neste caso, o usuário é capaz de especificar múltiplas modificações de características em um único ciclo. Por exemplo, o sistema Car Navigator [120] permite que o usuário especifique múltiplas modificações, que estão ocultas por trás de descrições informais que o usuário pode entender (por exemplo, mais elegante, mais espaçoso, mais barato, mais esportivo). Por exemplo, o especialista do domínio pode codificar o fato de que "mais elegante" sugere um determinado subconjunto de modelos com preço mais alto e estrutura interna sofisticada. É claro que também é possível para o usuário modificar as características necessárias do produto diretamente, mas isso aumenta a carga sobre ele. O ponto na crítica conversacional é que quando um usuário pode desejar ter um carro "mais elegante", mas pode não ser facilmente capaz de expressar isso concretamente em termos das características do produto, como a estrutura interna do carro. Por outro lado, uma qualificação como "mais elegante" é mais intuitiva e pode ser codificada em termos das características do produto por um especialista do domínio. Este processo interativo é projetado para ajudá-los a aprender o complexo espaço do produto de forma intuitiva.

No exemplo de compra de imóvel da Tabela 5.2, o usuário pode especificar uma localidade diferente e uma alteração no preço em um único ciclo. Um exemplo de crítica composta para o exemplo de compra de imóvel é ilustrado na Figura 5.9(a). Para tornar a abordagem mais conversacional, uma interface como a da Figura 5.9(b) codificará automaticamente múltiplas alterações em uma única seleção. Por exemplo, se o usuário selecionar "mais espaçoso", isso implica que tanto o número de quartos quanto o número de banheiros podem precisar ser aumentados. Para o segundo tipo de interface, o especialista no domínio precisa despender um esforço significativo no design da interface relevante e na interpretação das escolhas do usuário em termos de alterações feitas em múltiplos recursos do produto. Essa codificação é estática e é feita antecipadamente.

A principal vantagem da crítica composta é que o usuário pode alterar vários recursos na recomendação alvo para emitir uma nova consulta ou remover os resultados da pesquisa da consulta anterior. Como resultado, essa abordagem permite grandes saltos no espaço de recursos do produto, e o usuário frequentemente tem melhor controle sobre o processo de crítica. Isso é útil para reduzir o número de ciclos de recomendação e tornar o processo de exploração mais eficiente. No entanto, não está claro se as críticas compostas sempre ajudam o usuário a aprender o

EXAMPLE OF HYPOTHETICAL CASE-BASED RECOMMENDATION INTERFACE FOR HOME BUYING (critique-example.com)

[COMPOUND CRITIQUING INTERFACE]



YOU SPECIFIED THE FOLLOWING TARGET:
812 SCENIC DRIVE, MOHEGAN LAKE, NY

YOUR TOP RECOMMENDATION IS:
742 SCENIC DRIVE, MOHEGAN LAKE, NY

WE RECOMMEND THIS HOUSE BECAUSE: IT HAS SIMILAR BEDROOMS, BATHROOMS, LOCALITY, PRICE RANGE, AND HOME STYLE AS YOUR TARGET

I WOULD LIKE TO BUY A HOUSE SIMILAR TO THE TOP RECOMMENDATION
BUT WITH **ONE OR MORE** OF THE FOLLOWING CHANGES:

NUMBER OF BR	▼	NUMBER OF BATH	▼	HOME STYLE	▼
PRICE RANGE	▼	ZIP CODE			
SUBMIT SEARCH			SEE OTHER RESULTS		GO BACK TO ENTRY POINT

(a) Compound critiquing by modifying multiple feature values

EXAMPLE OF HYPOTHETICAL CASE-BASED RECOMMENDATION INTERFACE FOR HOME BUYING (critique-example.com)

[COMPOUND CRITIQUING INTERFACE]



YOU SPECIFIED THE FOLLOWING TARGET:
812 SCENIC DRIVE, MOHEGAN LAKE, NY

YOUR TOP RECOMMENDATION IS:
742 SCENIC DRIVE, MOHEGAN LAKE, NY

WE RECOMMEND THIS HOUSE BECAUSE: IT HAS SIMILAR BEDROOMS, BATHROOMS, LOCALITY, PRICE RANGE, AND HOME STYLE AS YOUR TARGET

I WOULD LIKE TO BUY A HOUSE SIMILAR TO THE TOP RECOMMENDATION
BUT WITH THE FOLLOWING GENERAL GUIDANCE (THE SYSTEM WILL AUTOMATICALLY ADJUST ONE OR MORE PRODUCT REQUIREMENTS FOR YOU):

CLASSIER	CHEAPER	MORE SPACIOUS	BETTER SCHOOL DISTRICT
		SEE OTHER RESULTS	
		GO BACK TO ENTRY POINT	

(b) Reducing the user's burden of specifying multiple features with domain knowledge

Figura 5.9: Exemplos hipotéticos de interfaces de usuário para crítica composta em um caso baseado recomendador (critique-example.com)

EXAMPLE OF HYPOTHETICAL CASE-BASED RECOMMENDATION INTERFACE FOR HOME BUYING (critique-example.com)

[DYNAMIC CRITIQUING INTERFACE]



YOU SPECIFIED THE FOLLOWING TARGET:
812 SCENIC DRIVE, MOHEGAN LAKE, NY

YOUR TOP RECOMMENDATION IS:
742 SCENIC DRIVE, MOHEGAN LAKE, NY

WE RECOMMEND THIS HOUSE BECAUSE: IT HAS SIMILAR BEDROOMS, BATHROOMS, LOCALITY, PRICE RANGE, AND HOME STYLE AS YOUR TARGET

I WOULD LIKE TO BUY A HOUSE SIMILAR TO THE TOP RECOMMENDATION BUT WITH ONE OF THE FOLLOWING CHANGE COMBINATIONS :

DIFFERENT STYLE AT SMALLER PRICE (12)	SUBMIT CHANGE	MORE BEDROOMS AT GREATER PRICE (22)	SUBMIT CHANGE
FEWER BEDROOMS AT SMALLER PRICE (13)	SUBMIT CHANGE	DIFFERENT STYLE IN NEARBY LOCALITY (29)	SUBMIT CHANGE
MORE BEDROOMS IN NEARBY LOCALITY (15)	SUBMIT CHANGE	SEE OTHER RESULTS	GO BACK TO ENTRY POINT

Figura 5.10: Um exemplo hipotético de uma interface de usuário para crítica dinâmica em um recomendador baseado em casos (critique-example.com)

O espaço do produto é melhor do que críticas simples; ciclos curtos de crítica também reduzem a probabilidade de o usuário aprender diferentes compensações e correlações entre recursos no espaço do produto. Por outro lado, um usuário pode, às vezes, aprender muito sobre o espaço do produto ao passar pelo processo lento e trabalhoso de crítica simples.

5.3.2.3 Críticas Dinâmicas

Embora as críticas compostas permitam saltos maiores no espaço de navegação, elas apresentam a desvantagem de que as opções de crítica apresentadas ao usuário são estáticas, no sentido de que não dependem dos resultados recuperados. Por exemplo, se o usuário estiver navegando por carros e já estiver navegando pelo carro mais caro com a maior potência possível, a opção de aumentar a potência e o preço ainda será exibida na interface de crítica. Claramente, especificar essas opções levará a uma busca infrutífera. Isso ocorre porque os usuários muitas vezes não estão totalmente cientes das compensações inerentes ao complexo espaço de produtos.

Na crítica dinâmica, o objetivo é utilizar a mineração de dados nos resultados recuperados para determinar as vias de exploração mais frutíferas e apresentá-las ao usuário. Assim, as críticas dinâmicas são, por definição, críticas compostas, pois quase sempre representam combinações de mudanças apresentadas ao usuário. A principal diferença é que apenas o subconjunto das possibilidades mais relevantes é apresentado, com base nos resultados recuperados no momento. Portanto, as críticas dinâmicas são projetadas para fornecer melhor orientação ao usuário durante a busca.

processo.

Um aspecto importante da crítica dinâmica é a capacidade de descobrir combinações frequentes de mudanças nas características do produto. A noção de suporte é adaptada do padrão frequente

mineração [23] para determinar padrões de características de produtos frequentemente concomitantes nos resultados recuperados. O suporte de um padrão é definido como a fração dos resultados recuperados que satisfazem esse padrão. Consulte a Definição 3.3.1 no Capítulo 3 para uma definição formal de suporte. Portanto, essa abordagem determina todos os padrões de mudança que especificam um valor mínimo de suporte predefinido. Por exemplo, no exemplo de compra de imóvel da Tabela 5.2, o sistema pode determinar as seguintes críticas dinâmicas em ordem de suporte:

Mais quartos, melhor preço: Suporte = 25%

Mais quartos, mais banheiros, melhor preço: Suporte = 20%

Menos quartos, preço menor: Suporte = 20%

Mais quartos, localidade=Yonkers: Suporte= 15%

Observe que opções conflitantes, como "Mais Quartos, Preço Menor", têm menor chance de serem incluídas, pois podem ser eliminadas com base no critério de suporte mínimo. No entanto, padrões com baixo suporte não são necessariamente desinteressantes. De fato, uma vez determinados todos os padrões que atendem ao limite mínimo de suporte, muitos sistemas de recomendação ordenam as críticas ao usuário em ordem crescente de suporte.

A lógica por trás dessa abordagem é que críticas com baixo apoio são frequentemente padrões menos óbvios que podem ser usados para eliminar um número maior de itens da lista de candidatos. Um exemplo hipotético de uma interface de crítica dinâmica, baseada em nosso sistema anterior de compra de imóveis (critique-example.com), é ilustrado na Figura 5.10. Observe que uma quantidade numérica está associada a cada uma das opções apresentadas na interface. Esse número corresponde ao apoio bruto das opções apresentadas.

Um exemplo real de uma abordagem de crítica dinâmica que utiliza mineração frequente de padrões e regras de associação é o sistema Qwikshop discutido em [491]. Uma observação importante sobre sistemas de crítica dinâmica é que eles aumentam a carga cognitiva do usuário, quando vistos por ciclo, mas reduzem a carga cognitiva ao longo de toda a sessão devido à sua capacidade de chegar a recomendações aceitáveis mais rapidamente [416].

Esta é uma das razões pelas quais o design eficaz de processos explicativos no ciclo de crítica é mais importante em sistemas de crítica dinâmicos.

5.3.3 Explicação em Críticas

É sempre aconselhável incorporar poder explicativo ao processo de crítica, pois isso ajuda o usuário a compreender melhor o espaço informacional. Existem diversas formas de explicação que são utilizadas para melhorar a qualidade das críticas. Alguns exemplos dessas explicações são os seguintes:

1. Na crítica simples, é comum que o usuário navegue de forma infrutífera devido à falta de consciência das compensações inerentes ao espaço do produto. Por exemplo, um usuário pode aumentar sucessivamente a potência, aumentar a quilometragem por galão e, em seguida, tentar reduzir o preço desejado. Nesses casos, o sistema pode não ser capaz de mostrar um resultado aceitável para o usuário, que terá que reiniciar o processo de navegação.
Ao final de tal sessão, é desejável que o sistema determine automaticamente a natureza do trade-off que resultou em uma sessão infrutífera. Frequentemente, é possível determinar tais trade-offs com o uso de estatísticas de correlação e coocorrência. O usuário pode então obter insights sobre os conflitos nas críticas inseridas por ele na sessão anterior. Essa abordagem é utilizada em alguns sistemas FindMe [121].

2. Foi demonstrado em [492] como explicações podem ser usadas em conjunto com críticas compostas dinâmicas durante uma sessão. Por exemplo, o sistema Qwikshop fornece informações sobre a fração de instâncias que satisfazem cada crítica composta. Isso fornece ao usuário uma ideia clara do tamanho do espaço que está prestes a explorar antes de fazer uma escolha crítica. Fornecer ao usuário melhores explicações durante a sessão aumenta a probabilidade de que ela seja frutífera.

O principal perigo em sistemas baseados em crítica é a probabilidade de os usuários vagarem pelo espaço do conhecimento sem rumo, sem encontrar o que procuram. Adicionar explicações à interface reduz bastante essa probabilidade.

5.4 Personalização Persistente em Sistemas Baseados em Conhecimento

Embora sistemas baseados em conhecimento, como sistemas baseados em restrições, permitam a especificação de preferências, características e/ou atributos demográficos do usuário, as informações inseridas são tipicamente específicas da sessão e não são persistentes entre as sessões. Os únicos dados persistentes na maioria desses sistemas são o conhecimento do domínio na forma de vários bancos de dados específicos do sistema, como restrições ou métricas de similaridade. Essa falta de dados persistentes é uma consequência natural de como os sistemas baseados em conhecimento tendem a usar dados históricos apenas de forma limitada em comparação com sistemas baseados em conteúdo e colaborativos. Isso também é uma vantagem dos sistemas baseados em conhecimento, pois eles tendem a sofrer menos com problemas de inicialização a frio em comparação com outros sistemas que dependem de dados históricos. De fato, os sistemas de recomendação baseados em conhecimento são frequentemente projetados para itens mais caros e comprados ocasionalmente, que são altamente personalizados. Nesses casos, os dados históricos devem ser usados com alguma cautela, mesmo quando disponíveis. No entanto, alguns sistemas baseados em conhecimento também foram projetados para usar formas persistentes de personalização.

As ações do usuário em várias sessões podem ser usadas para construir um perfil persistente sobre o usuário em relação ao que ele gostou ou não gostou. Por exemplo, CASPER é um sistema de recrutamento online [95] no qual as ações do usuário em anúncios de emprego recuperados, como salvar o anúncio, enviá-lo por e-mail para si mesmo ou se candidatar ao anúncio, são salvas para referência futura. Além disso, os usuários podem classificar negativamente os anúncios quando eles são irrelevantes. Observe que esse processo resulta na construção de um perfil de feedback implícito. O processo de recomendação é uma abordagem de duas etapas. Na primeira etapa, os resultados são recuperados com base nos requisitos do usuário, como no caso de qualquer recomendador baseado em conhecimento. Subsequentemente, os resultados são classificados com base na similaridade com perfis anteriores que o usuário gostou. Também é possível incluir informações colaborativas identificando outros usuários com perfis semelhantes e usando suas informações de sessão no processo de aprendizagem.

Muitas etapas em sistemas baseados em conhecimento podem ser personalizadas quando os dados de interação do usuário está disponível. Os passos são os seguintes:

1. O aprendizado de funções de utilidade/similaridade sobre vários atributos pode ser personalizado tanto para recomendadores baseados em restrições (fase de classificação) quanto para recomendadores baseados em casos (fase de recuperação). Quando o feedback anterior de um usuário específico está disponível, é possível aprender a importância relativa de vários atributos para esse usuário na função de utilidade.
2. O processo de sugestão de restrições (cf. seção 5.2.5) para um usuário pode ser personalizado se um número significativo de sessões desse usuário estiver disponível.

3. As críticas dinâmicas para um usuário podem ser personalizadas se houver dados suficientes disponíveis sobre esse usuário para determinar padrões relevantes. A única diferença em relação à forma mais comum de crítica dinâmica é que dados específicos do usuário são utilizados, em vez de todos os dados, para determinar os padrões frequentes. Também é possível incluir as sessões de usuários com sessões semelhantes no processo de mineração para aumentar o poder de colaboração do recomendador.

Embora existam muitas maneiras pelas quais a personalização pode ser incorporada à estrutura da recomendação baseada em conhecimento, o maior desafio geralmente é a indisponibilidade de dados de sessão suficientes para um usuário específico. Sistemas baseados em conhecimento são inherentemente projetados para itens altamente personalizados em um espaço de domínio complexo. É por isso que o nível de personalização é geralmente limitado em domínios baseados em conhecimento.

5.5 Resumo

Sistemas de recomendação baseados em conhecimento são geralmente projetados para domínios nos quais os itens são altamente personalizados e é difícil que as informações de classificação reflitam diretamente preferências mais amplas. Nesses casos, é desejável dar ao usuário maior controle no processo de recomendação por meio da especificação de requisitos e interatividade. Sistemas de recomendação baseados em conhecimento podem ser sistemas baseados em restrições ou em casos. Em sistemas baseados em restrições, os usuários especificam seus requisitos, que são combinados com regras específicas do domínio para fornecer recomendações. Os usuários podem adicionar ou flexibilizar restrições dependendo do tamanho dos resultados. Em sistemas baseados em casos, os usuários trabalham com alvos e listas de candidatos que são modificados iterativamente por meio do processo de crítica.

Para recuperação, são utilizadas funções de similaridade dependentes de domínio, que também podem ser aprendidas. As modificações nas consultas são realizadas por meio da crítica. As críticas podem ser simples, compostas ou dinâmicas. Sistemas baseados em conhecimento baseiam-se amplamente nos requisitos do usuário e incorporam apenas uma quantidade limitada de dados históricos. Portanto, geralmente são eficazes no tratamento de problemas de inicialização a frio. A desvantagem dessa abordagem é que as informações históricas não são utilizadas para "preencher lacunas". Nos últimos anos, métodos também foram desenvolvidos para incorporar um maior grau de personalização com o uso de informações históricas das sessões do usuário.

5.6 Notas Bibliográficas

Pesquisas sobre vários sistemas de recomendação baseados em conhecimento e métodos de elicitación de preferências podem ser encontradas em [197, 417]. Sistemas de recomendação baseados em casos são revisados em [102, 116, 377, 558]. Pesquisas sobre métodos de elicitación de preferências e críticas podem ser encontradas em [148, 149]. Sistemas de recomendação baseados em restrições são discutidos em [196, 197]. Historicamente, sistemas de recomendação baseados em restrições foram propostos muito mais tarde do que os recomendadores baseados em casos. De fato, o artigo original de Burke [116] sobre sistemas de recomendação baseados em conhecimento descreve principalmente recomendadores baseados em casos. No entanto, alguns aspectos dos recomendadores baseados em restrições também são descritos neste trabalho. Métodos para aprender funções de utilidade no contexto de sistemas de recomendação baseados em restrições são discutidos em [155, 531]. Métodos para lidar com resultados vazios em sistemas baseados em restrições, como descoberta rápida de pequenos conjuntos conflitantes e relaxamentos mínimos são discutidos em [198, 199, 273, 274, 289, 419, 574].

Esses trabalhos também discutem como esses conjuntos conflitantes podem ser usados para fornecer explicações e diagnósticos de reparo das consultas do usuário. Métodos baseados em popularidade para selecionar o próximo atributo de restrição são discutidos em [196, 389]. A seleção de valores padrão para o atributo

restrições são discutidas em [483]. Um sistema de recomendação baseado em restrições bem conhecido é o recomendador VITA [201], que foi construído com base no sistema CWAdvisor [200].

O problema do aprendizado de funções de similaridade para recomendadores baseados em casos é discutido em [18, 97, 163, 563, 627]. O trabalho em [563] é notável por aprender pesos de vários recursos para computação de similaridade. Métodos de aprendizado por reforço para aprender funções de similaridade para sistemas baseados em casos são discutidos em [288, 506]. As estratégias de seleção aleatória limitada e seleção gananciosa limitada para aumentar a diversidade de sistemas de recomendação baseados em casos são discutidas em [560]. O trabalho em [550] também combina similaridade com diversidade como a abordagem gananciosa limitada, mas aplica apenas diversidade no conjunto recuperado de b-k casos, em vez de criar uma métrica de qualidade combinando similaridade e diversidade. As noções de camadas de similaridade e intervalos de similaridade para aprimoramento de diversidade são discutidas em [420]. Uma abordagem orientada a compromisso para aprimoramento de diversidade é discutida em [421].

O poder da recuperação baseada em ordem para diversificação de similaridade é discutido em [101]. Resultados experimentais [94, 560] mostram as vantagens de incorporar diversidade em sistemas de recomendação. A questão da crítica em sistemas de recomendação baseados em casos é discutida em detalhes em [417, 422, 423]. Críticas compostas foram discutidas pela primeira vez em [120], embora o termo tenha sido cunhado pela primeira vez em [414]. Um estudo comparativo de várias técnicas de crítica composta pode ser encontrado em [664]. O uso de explicações em críticas compostas é discutido em [492].

Os primeiros sistemas de recomendação baseados em casos foram propostos em [120, 121] no contexto do sistema de recomendação de restaurantes Entree. As primeiras formas desses sistemas também foram chamadas de sistemas FindMe [121], que se mostraram aplicáveis a uma ampla variedade de domínios. O personal shopper Wasabi é um sistema de recomendação baseado em casos e é discutido em [125]. Sistemas baseados em casos têm sido usados para serviços de consultoria de viagens [507], sistemas de recrutamento online [95], vendas de carros (Car Navigator) [120], vendas de vídeos (Video Navigator) [121], filmes (Pick A Flick) [121], recomendações de câmeras digitais (por exemplo, Qwikshop) [279, 491] e acomodações em imóveis para aluguel [263].

A maioria dos sistemas baseados em conhecimento alavanca os requisitos e preferências do usuário, conforme especificado em uma única sessão. Portanto, se um usuário diferente inserir a mesma entrada, obterá exatamente o mesmo resultado. Embora tal abordagem forneça melhor controle ao usuário e também não sofra de problemas de inicialização a frio, ela tende a ignorar dados históricos quando disponíveis. Nos últimos anos, também testemunhamos um aumento nas informações persistentes e de longo prazo sobre o usuário em sistemas de recomendação baseados em conhecimento [95, 454, 558]. Um exemplo de tal sistema é o sistema de recrutamento online CASPER [95], que constrói perfis de usuário persistentes para recomendações futuras. Um sistema de recomendação de viagens personalizado com o uso de perfis de usuário é discutido em [170]. As sessões de usuários semelhantes são alavancadas para recomendações de viagens personalizadas em [507]. Tal abordagem não apenas alavanca o comportamento do usuário-alvo, mas também as informações colaborativas disponíveis em uma comunidade de usuários.

O trabalho em [641] utiliza as informações de crítica em múltiplas sessões de forma colaborativa para construir perfis de usuários. Outro trabalho relevante é a abordagem MAUT [665], que se baseia na teoria da utilidade multiatributo. Essa abordagem aprende uma função de preferência de utilidade para cada usuário com base em suas críticas nas sessões anteriores. Outro exemplo de dados persistentes que podem ser usados efetivamente em tais sistemas são as informações demográficas. Embora os sistemas de recomendação demográfica variem amplamente em seu uso [117, 320], alguns dos sistemas demográficos também podem ser considerados sistemas baseados em conhecimento, quando regras de associação de perfil são usadas para sugerir interativamente preferências aos usuários de forma online [31, 32]. Tais sistemas permitem o refinamento progressivo das consultas a fim de derivar o conjunto de regras mais apropriado para um determinado grupo demográfico. Da mesma forma, vários tipos de técnicas de recomendação e classificação baseadas em utilidade são usados no contexto de sistemas baseados em conhecimento [74].

5.7 Exercícios

1. Implemente um algoritmo para determinar se um conjunto de requisitos especificados pelo cliente e um conjunto de regras em uma base de conhecimento recuperarão um conjunto vazio de um catálogo de produtos. Suponha que o antecedente e o consequente de cada regra contenham uma única restrição sobre as características do produto. As restrições sobre atributos numéricos estão na forma de desigualdades (por exemplo, Preço \geq 30), enquanto as restrições sobre atributos categóricos estão na forma de instanciações unitárias (por exemplo, Cor=Azul). Além disso, os requisitos do cliente também são expressos como restrições semelhantes no espaço de características.
2. Suponha que você tenha dados contendo informações sobre os valores de utilidade de um determinado cliente para um grande conjunto de itens em um domínio específico (por exemplo, carros). Suponha que o valor de utilidade do j-ésimo produto seja u_j ($j \in \{1 \dots n\}$). Os itens são descritos por um conjunto de d características numéricas. Discuta como você usará esses dados para classificar outros itens no mesmo domínio de produto para esse cliente.

Capítulo 6

Baseado em conjunto e híbrido Sistemas de Recomendação

"O que é melhor, uma intuição poética ou uma obra intelectual? Acho que se complementam." – Manuel Puig

6.1 Introdução

Nos capítulos anteriores, discutimos três classes diferentes de métodos de recomendação. Métodos colaborativos usam as avaliações de uma comunidade de usuários para fazer recomendações, enquanto métodos baseados em conteúdo usam as avaliações de um único usuário em conjunto com descrições de itens centradas em atributos para fazer recomendações. Métodos baseados em conhecimento exigem a especificação explícita dos requisitos do usuário para fazer recomendações e não exigem nenhuma avaliação histórica. Portanto, esses métodos usam diferentes fontes de dados e têm diferentes pontos fortes e fracos. Por exemplo, sistemas baseados em conhecimento podem lidar com problemas de inicialização a frio muito melhor do que sistemas baseados em conteúdo ou colaborativos, porque não exigem avaliações. Por outro lado, eles são mais fracos do que sistemas baseados em conteúdo e colaborativos em termos de uso de personalização persistente a partir de dados históricos. Se um usuário diferente inserir os mesmos requisitos e dados em uma interface interativa baseada em conhecimento, ele poderá obter exatamente o mesmo resultado.

Todos esses modelos parecem bastante restritivos isoladamente, especialmente quando múltiplas fontes de dados estão disponíveis. Em geral, seria desejável utilizar todo o conhecimento disponível em diferentes fontes de dados e também utilizar o poder algorítmico de vários sistemas de recomendação para fazer inferências robustas. Sistemas de recomendação híbridos foram projetados para explorar essas possibilidades. Existem três maneiras principais de criar sistemas de recomendação híbridos:

1. Design de conjunto: neste design, os resultados de algoritmos prontos para uso são combinados em uma única saída mais robusta. Por exemplo, pode-se combinar as saídas de classificação de um recomendador baseado em conteúdo e de um recomendador colaborativo em uma única saída.

Existe uma variação significativa em termos das metodologias específicas utilizadas para o processo de combinação. O princípio básico em vigor não é muito diferente do design de métodos de conjunto em muitas aplicações de mineração de dados, como agrupamento, classificação e análise de outliers.

O design do conjunto pode ser formalizado da seguinte forma: R^k seja uma matriz $m \times n$ contendo o

Sejam R^k as previsões dos m usuários para os n itens pelo k -ésimo algoritmo, onde $k \in \{1 \dots q\}$.

Portanto, um total de q algoritmos diferentes são usados para chegar a essas previsões.

A entrada (u, j)-ésima de R^k contém a avaliação prevista do usuário u para o item j pelo k -ésimo algoritmo.

Observe que as entradas observadas da matriz de avaliações original R são replicadas em cada R^k devido às diferentes previsões de R^k , e apenas as entradas não observadas de R variam em diferentes R^k diferentes algoritmos. O resultado final do algoritmo é obtido pela combinação das previsões $R^1 \dots R^q$ em uma única saída.

Essa combinação pode ser realizada de várias maneiras, como o cálculo da média ponderada das várias previsões. Além disso, em alguns algoritmos de conjunto sequencial, a previsão R^k pode depender dos resultados do componente anterior R^{k-1} . Em outros casos, as saídas podem não ser combinadas diretamente. Em vez disso, a saída de um sistema é usada como entrada para o próximo como um conjunto de recursos de conteúdo. As características comuns de todos esses sistemas são que (a) eles usam recomendadores existentes de forma pronta para uso e (b) eles produzem uma pontuação ou classificação unificada.

2. Design monolítico: Neste caso, um algoritmo de recomendação integrado é criado utilizando vários tipos de dados.

Às vezes, pode não haver uma distinção clara entre as várias partes (por exemplo, conteúdo e colaboração) do algoritmo. Em outros casos, algoritmos de recomendação colaborativos ou baseados em conteúdo existentes podem precisar ser modificados para serem usados dentro da abordagem geral, mesmo quando há distinções claras entre os estágios baseado em conteúdo e colaboração. Portanto, essa abordagem tende a integrar as várias fontes de dados de forma mais estreita, e não é fácil visualizar componentes individuais como caixas-pretas prontas para uso.

3. Sistemas mistos: Assim como os conjuntos, esses sistemas utilizam múltiplos algoritmos de recomendação como caixas-pretas, mas os itens recomendados pelos vários sistemas são apresentados lado a lado. Por exemplo, o programa de televisão de um dia inteiro é uma entidade composta contendo vários itens. Não faz sentido considerar a recomendação de um único item isoladamente; em vez disso, é a combinação dos itens que cria a recomendação.

Portanto, o termo “sistema híbrido” é usado em um contexto mais amplo do que o termo “sistema de conjunto”. Todos os sistemas de conjunto são, por definição, sistemas híbridos, mas o inverso não é necessariamente verdadeiro.

Embora sistemas de recomendação híbridos geralmente combinem o poder de diferentes tipos de recomendadores (por exemplo, baseados em conteúdo e conhecimento), não há razão para que tais sistemas não possam combinar modelos do mesmo tipo. Como os modelos baseados em conteúdo são essencialmente classificadores de texto, é sabido que existe uma ampla variedade de modelos de conjunto para melhorar a precisão da classificação. Portanto, qualquer sistema de conjunto baseado em classificação pode ser usado para melhorar a eficácia dos modelos baseados em conteúdo. Este argumento também se estende aos modelos de recomendação colaborativos. Por exemplo, pode-se facilmente combinar os resultados previstos de

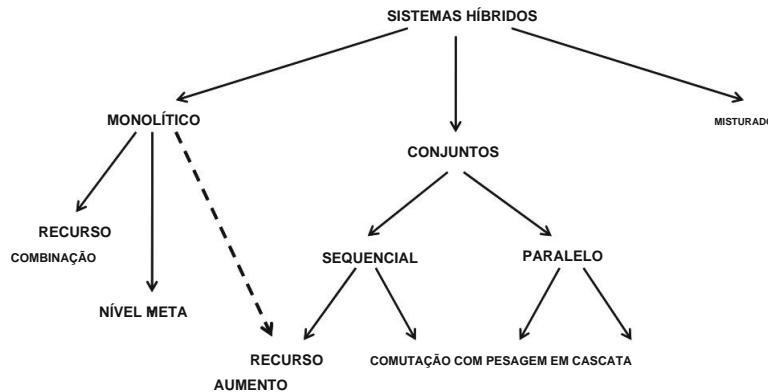


Figura 6.1: A taxonomia dos sistemas híbridos

um modelo de fator latente com aqueles de um modelo de vizinhança para obter recomendações mais precisas [266]. De fato, ambas as inscrições vencedoras no concurso Netflix Prize, chamadas de "Bellkor's Pragmatic Chaos" [311] e "The Ensemble" [704], eram sistemas de conjunto.

Em um nível mais amplo, os sistemas de recomendação híbridos estão intimamente relacionados ao campo da análise de conjunto em classificação. Por exemplo, modelos colaborativos são generalizações de modelos de classificação, conforme discutido na introdução do Capítulo 3. Como discutiremos na seção 6.2 deste capítulo, os fundamentos teóricos da análise de conjunto em classificação são semelhantes aos da filtragem colaborativa. Portanto, este capítulo também se concentrará em como a abordagem de recomendação pode ser usada para melhorar a eficácia dos sistemas de recomendação colaborativos, da mesma forma que se pode usar conjuntos no campo da classificação de dados.

De acordo com Burke [117], os sistemas de recomendação híbridos podem ser classificados nas seguintes categorias:

1. Ponderada: Neste caso, as pontuações de vários sistemas de recomendação são combinadas em uma única pontuação unificada, calculando-se os agregados ponderados das pontuações dos componentes individuais do conjunto. A metodologia para ponderar os componentes pode ser heurística ou pode utilizar modelos estatísticos formais.
2. Alternância: O algoritmo alterna entre vários sistemas de recomendação, dependendo das necessidades atuais. Por exemplo, em fases iniciais, pode-se usar um sistema de recomendação baseado em conhecimento para evitar problemas de inicialização a frio. Em fases posteriores, quando mais avaliações estão disponíveis, pode-se usar um sistema de recomendação baseado em conteúdo ou colaborativo. Alternativamente, o sistema pode selecionar de forma adaptativa o sistema de recomendação específico que fornece a recomendação mais precisa em um determinado momento.
3. Cascata: Neste caso, um sistema de recomendação refina as recomendações fornecidas por outro. Em formas generalizadas de cascata, como o boosting, o processo de treinamento de um sistema de recomendação é influenciado pela saída do sistema anterior, e os resultados gerais são combinados em uma única saída.

¹Ambas as inscrições empatarem na taxa de erro. O prêmio foi concedido à primeira por ter sido enviada 20 minutos antes.

4. Aumento de características: A saída de um sistema de recomendação é usada para criar características de entrada para o próximo. Enquanto o híbrido em cascata refina sucessivamente as recomendações do sistema anterior, a abordagem de aumento de características as trata como características de entrada para o próximo sistema. Essa abordagem compartilha uma série de semelhanças intuitivas com a noção de empilhamento, comumente usada em classificação. No empilhamento, as saídas de um classificador são usadas como características para o próximo. Como os diferentes recomendadores são (geralmente) usados como caixas-pretas prontas para uso, a abordagem ainda é um método de conjunto (na maioria dos casos) em vez de um método monolítico.
5. Combinação de recursos: neste caso, os recursos de diferentes fontes de dados são combinados e utilizados no contexto de um único sistema de recomendação. Essa abordagem pode ser vista como um sistema monolítico e, portanto, não é um método de conjunto.
6. Meta-nível: O modelo utilizado por um sistema de recomendação é usado como entrada para outro sistema. A combinação típica utilizada é a de um sistema baseado em conteúdo e um sistema colaborativo. O sistema colaborativo é modificado para usar os recursos de conteúdo para determinar grupos de pares. Em seguida, a matriz de classificações é usada em conjunto com esse grupo de pares para fazer previsões. Observe que essa abordagem precisa modificar o sistema colaborativo para usar uma matriz de conteúdo para encontrar grupos de pares, embora as previsões finais ainda sejam realizadas com a matriz de classificações. Portanto, o sistema colaborativo precisa ser modificado e não é possível usá-lo de forma padronizada. Isso torna a abordagem de meta-nível um sistema monolítico em vez de um sistema de conjunto. Alguns desses métodos também são chamados de "colaboração via conteúdo" devido à maneira como combinam informações colaborativas e de conteúdo.
7. Mista: Recomendações de vários mecanismos são apresentadas ao usuário simultaneamente. A rigor, essa abordagem não é um sistema de conjunto, pois não combina explicitamente as pontuações (de um item específico) dos vários componentes. Além disso, essa abordagem é frequentemente usada quando a recomendação é uma entidade composta na qual vários itens podem ser recomendados como um conjunto relacionado. Por exemplo, um programa de televisão composto pode ser construído a partir dos vários itens recomendados [559].
- Portanto, essa abordagem é bem diferente de todos os métodos mencionados anteriormente. Por um lado, ela utiliza outros recomendadores como caixas-pretas (como conjuntos), mas não combina as classificações previstas do mesmo item por diferentes recomendadores.
- Portanto, os recomendadores mistos não podem ser vistos como métodos monolíticos ou baseados em conjuntos, sendo classificados em uma categoria distinta. A abordagem é mais relevante em domínios de itens complexos e é frequentemente usada em conjunto com sistemas de recomendação baseados em conhecimento.

As quatro primeiras categorias mencionadas são sistemas de conjunto, as duas seguintes são sistemas monolíticos e a última é um sistema misto. A última categoria de sistemas mistos não pode ser categorizada de forma precisa como um sistema monolítico ou de conjunto, porque apresenta múltiplas recomendações como uma entidade composta. Uma categorização hierárquica desses vários tipos de sistemas é mostrada na Figura 6.1. Embora tenhamos usado a categorização de nível superior de sistemas paralelos e sequenciais², conforme introduzida por [275], enfatizamos que nossa categorização do conjunto original de seis categorias de Burke é ligeiramente diferente daquela de [275]. Ao contrário da taxonomia em [275], que classifica os sistemas de meta-nível como métodos sequenciais, vemos os sistemas de meta-nível como monolíticos porque não se pode usar algoritmos de recomendação prontos para uso, como no caso de um verdadeiro conjunto. Da mesma forma, o trabalho em [275]

²Isso também é conhecido como sistema de pipeline [275].

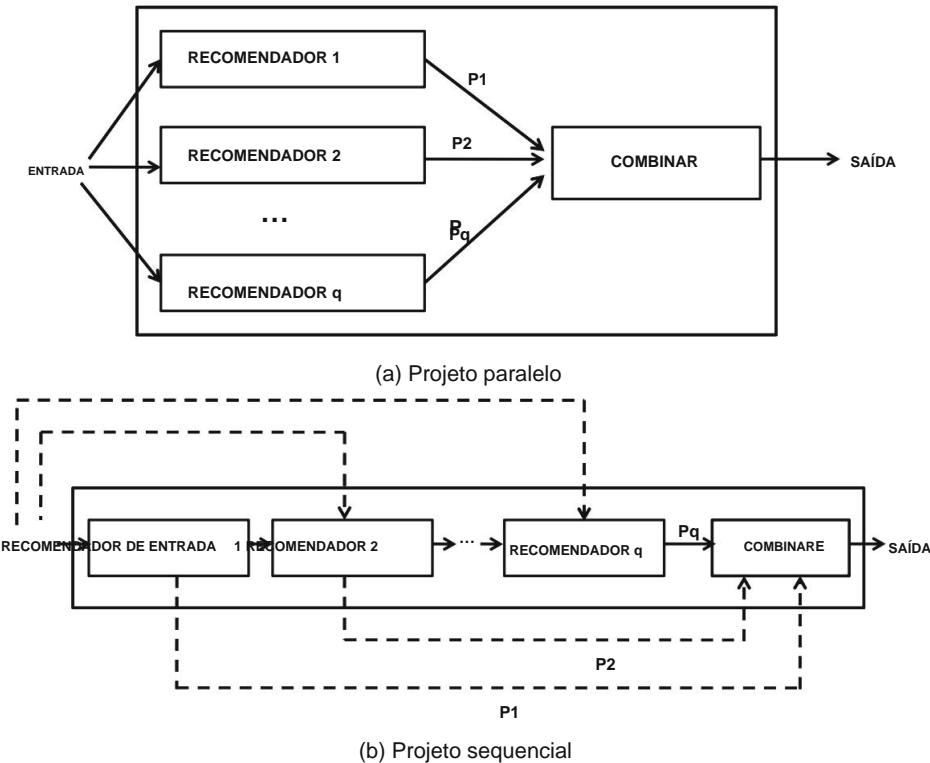


Figura 6.2: Conjuntos paralelos e sequenciais

As visualizações apresentam híbridos de aumento como sistemas monolíticos. Embora os recomendadores individuais sejam combinados de forma mais complexa em híbridos de aumento de recursos, recomendadores individuais ainda são usados como caixas-pretas prontas para uso em grande parte. Isto é a principal característica distintiva de um sistema de conjunto de um sistema monolítico, e a abordagem lembra muito os métodos de empilhamento na classificação. Portanto, nós visualizar híbridos de aumento de recursos como sistemas de conjunto em vez de sistemas monolíticos. No entanto, em alguns casos de híbridos de aumento de recursos, pequenas alterações são necessárias no recomendador pronto para uso. Nesses casos, esses sistemas podem ser tecnicamente considerados têm um design monolítico. Mostramos essa possibilidade com uma linha pontilhada na Figura 6.1.

Além dos métodos monolíticos e mistos, que não são verdadeiramente conjuntos, um pode visualizar todos os métodos de conjunto como tendo designs sequenciais ou paralelos [275]. Em projetos paralelos, os vários recomendadores funcionam independentemente uns dos outros e as previsões dos recomendadores individuais são combinadas no final. A ponderação e os métodos de comutação podem ser vistos como projetos paralelos. Em projetos sequenciais, a saída de um recomendador é usado como entrada para o outro. Os sistemas em cascata e meta-nível podem ser vistos como exemplos de métodos sequenciais. Uma ilustração pictórica da combinação O processo em sistemas sequenciais e paralelos é mostrado na Figura 6.2. Neste capítulo, iremos fornecer uma discussão detalhada de vários sistemas de recomendação em cada uma dessas categorias, embora usaremos a taxonomia de nível inferior de Burke [117] para organizar a discussão.

Este capítulo está organizado da seguinte forma. Na seção 6.2, discutimos a perspectiva de classificação dos sistemas de recomendação baseados em conjuntos. Também exploramos o nível em que a teorias e metodologias existentes para métodos de conjunto no campo da classificação também

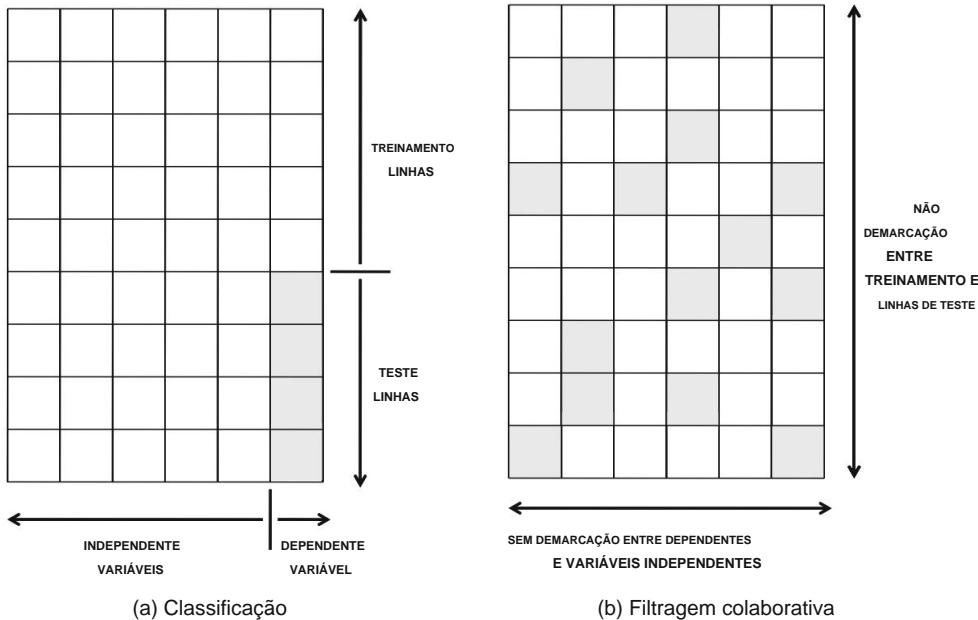


Figura 6.3: Revisando a Figura 1.4 do Capítulo 1. Comparando o problema de classificação tradicional com a filtragem colaborativa. Entradas sombreadas estão ausentes e precisam ser previstas.

Aplicar a sistemas de recomendação. Na seção 6.3, são discutidos diversos exemplos de híbridos ponderados. Na seção 6.4, são discutidos diversos híbridos de comutação. Híbridos em cascata são discutidos na seção 6.5, enquanto híbridos de aumento de características são discutidos na seção 6.6. Híbridos de meta-nível são discutidos na seção 6.7. Métodos de combinação de características são apresentados na seção 6.8. Sistemas mistos são discutidos na seção 6.9. Um resumo é fornecido na seção 6.10.

6.2 Métodos de conjunto da classificação

Perspectiva

Métodos de ensemble são comumente utilizados na área de classificação de dados para melhorar a robustez de algoritmos de aprendizagem. Como discutiremos a seguir, grande parte dessa teoria também se aplica a diversas formas de sistemas de recomendação. Por exemplo, sistemas de recomendação baseados em conteúdo são frequentemente aplicações diretas de algoritmos de classificação de texto. Portanto, uma aplicação direta dos métodos de ensemble existentes na classificação geralmente é suficiente para obter resultados de alta qualidade.

Conforme discutido no Capítulo 1, a filtragem colaborativa é uma generalização do problema de classificação de dados. Replicamos a Figura 1.4 do Capítulo 1 na Figura 6.3 para ilustrar a relação entre os dois problemas. É evidente a partir da Figura 6.3(a) que as variáveis de característica e a variável de classe são claramente demarcadas na classificação. As principais características distintivas da filtragem colaborativa da classificação são que as variáveis de característica e a variável de classe não são claramente demarcadas na primeira e que as entradas ausentes podem ocorrer em qualquer coluna ou linha. O fato de que entradas ausentes podem ocorrer em qualquer linha implica que as instâncias de treinamento e teste também não são claramente demarcadas. Uma questão importante surge quanto a

se a teoria de viés-variância desenvolvida no campo da classificação [242] também se aplica a sistemas de recomendação. Experimentos repetidos [266, 311] demonstraram que a combinação de múltiplos sistemas de recomendação colaborativos frequentemente leva a resultados mais precisos. Isso ocorre porque a teoria de viés-variância, projetada para classificação, também se aplica ao cenário de filtragem colaborativa. Isso significa que muitas técnicas tradicionais de conjunto, da classificação, também podem ser generalizadas para a filtragem colaborativa. No entanto, devido ao fato de que as entradas ausentes podem ocorrer em qualquer linha ou coluna dos dados, às vezes é algorítmicamente desafiador generalizar os algoritmos de conjunto para classificação para a filtragem colaborativa.

Primeiro, apresentamos o trade-off viés-variância aplicado ao campo de classificação de dados. Considere um modelo simplificado de classificação ou regressão, no qual um campo específico precisa ser previsto, conforme mostrado na Figura 6.3(a). Pode-se demonstrar que o erro de um classificador na previsão da variável dependente pode ser decomposto em três componentes:

1. Viés: Cada classificador faz suas próprias suposições de modelagem sobre a natureza da fronteira de decisão entre classes. Por exemplo, um classificador SVM linear assume que as duas classes podem ser separadas por uma fronteira de decisão linear. Isso, obviamente, não é verdade na prática. Em outras palavras, qualquer máquina de vetores de suporte linear terá um viés inherente. Quando um classificador tem alto viés, ele fará previsões consistentemente incorretas sobre escolhas específicas de instâncias de teste próximas à fronteira de decisão modelada incorretamente, mesmo quando diferentes amostras dos dados de treinamento são usadas para o processo de aprendizagem.
2. Variância: Variações aleatórias nas escolhas dos dados de treinamento levarão a modelos diferentes. Como resultado, a variável dependente para uma instância de teste pode ser prevista de forma inconsistente por diferentes escolhas de conjuntos de dados de treinamento. A variância do modelo está intimamente relacionada ao sobreajuste. Quando um classificador tem tendência ao sobreajuste, ele fará previsões inconsistentes para a mesma instância de teste em diferentes conjuntos de dados de treinamento.
3. Ruído: O ruído refere-se aos erros intrínsecos na rotulagem da classe-alvo. Como este é um aspecto intrínseco da qualidade dos dados, há pouco que se possa fazer para corrigi-lo.

Portanto, o foco da análise de conjunto geralmente está na redução do viés e da variância.

O erro quadrático médio esperado de um classificador em um conjunto de instâncias de teste pode ser demonstrado como a soma do viés, da variância e do ruído. Essa relação pode ser expressa da seguinte forma:

$$\text{Erro} = \text{Viés}^2 + \text{Variância} + \text{Ruído} \quad (6.1)$$

Vale ressaltar que, ao reduzir os componentes de viés ou variância, pode-se reduzir o erro geral de um classificador. Por exemplo, métodos de conjunto de classificação, como bagging [99], reduzem a variância, enquanto métodos como boosting [206] podem reduzir o viés. Vale ressaltar que a única diferença entre classificação e filtragem colaborativa é que entradas ausentes podem ocorrer em qualquer coluna, e não apenas na variável de classe. No entanto, o resultado de viés-variância ainda se mantém quando aplicado ao problema de previsão de uma coluna específica, independentemente de as outras colunas serem especificadas de forma incompleta ou não. Isso significa que os princípios básicos da análise de conjunto em classificação também são válidos para filtragem colaborativa.

De fato, como veremos mais adiante neste capítulo, muitos métodos clássicos de conjunto em classificação, como bagging e boosting, também foram adaptados à filtragem colaborativa.

6.3 Híbridos ponderados

Seja $R = [r_{uj}]$ uma matriz de classificações $m \times n$. Em híbridos ponderados, as saídas de vários sistemas de recomendação são combinadas usando um conjunto de pesos. Sejam $R^1 \dots R^q$ as matrizes de classificações completamente especificadas $m \times n$, nas quais as entradas não observadas de R são previstas por q algoritmos diferentes. Observe que as entradas r_{uj} que já são observadas na matriz de classificações $m \times n$ original R já estão fixadas em seus valores observados em cada matriz de previsão R^k . Então, para um conjunto de pesos $\hat{y}^1 \dots \hat{y}^q$, o híbrido ponderado cria uma matriz de previsão combinada $\hat{R} = [\hat{r}_{uj}]$ como segue:

$$\hat{R} = \sum_{i=1}^q \hat{y}^i R^i \quad (6.2)$$

ponderar os vários sistemas de forma diferencial, de modo a $\sum_i \hat{y}^i = 1$. No entanto, o ideal é dar maior importância aos sistemas mais precisos. Existem vários métodos para essa ponderação diferencial.

Também é possível escrever a equação acima mencionada em termos de entradas individuais da matriz:

$$\hat{r}_{uj} = \sum_{i=1}^q \hat{y}^i r_{uj} \quad (6.3)$$

Aqui \hat{r}_{uj} denota a previsão do i -ésimo componente do conjunto para o usuário u e o item j e \hat{r}_{uj} denota a previsão final.

Para determinar os pesos ótimos, é necessário ser capaz de avaliar a eficácia de uma combinação particular de pesos $\hat{y}^1 \dots \hat{y}^q$. Embora este tópico seja discutido em mais detalhes no Capítulo 7, forneceremos uma abordagem de avaliação simples aqui para fins de discussão. Uma abordagem simples é manter uma pequena fração (por exemplo, 25%) das entradas conhecidas na matriz de classificações $m \times n$ $R = [r_{uj}]$ e criar as matrizes de predição $R^1 \dots R^q$ aplicando os q algoritmos de base diferentes nos 75% restantes das entradas em R . As previsões resultantes $R^1 \dots R^q$ são então combinadas para criar a previsão baseada em conjunto \hat{R} de acordo com a Equação 6.2. Sejam os índices de itens do usuário (u, j) dessas entradas mantidas denotados por H . Então, para um dado vetor $\hat{y} = (\hat{y}^1 \dots \hat{y}^q)$ de pesos, a eficácia de um esquema particular pode ser avaliada usando o erro quadrático médio (MSE) ou o erro absoluto médio (MAE) da matriz prevista $\hat{R} = [\hat{r}_{uj}]_{m \times n}$ sobre as classificações mantidas em H :

$$\begin{aligned} \text{MSE}(\hat{y}) &= \frac{\sum_{(u,j) \in H} (\hat{r}_{uj} - r_{uj})^2}{|H|} \\ \text{MAE}(\hat{y}) &= \frac{\sum_{(u,j) \in H} |\hat{r}_{uj} - r_{uj}|}{|H|} \end{aligned}$$

Essas métricas fornecem uma avaliação de uma combinação particular de coeficientes $\hat{y}^1 \dots \hat{y}^q$. Como podemos determinar os valores ótimos de $\hat{y}^1 \dots \hat{y}^q$ para minimizar essas métricas? Uma abordagem simples, que funciona bem para o caso do MSE, é usar a regressão linear. Supõe-se que as classificações no conjunto H retido fornecem os valores de verdade da variável dependente, e os parâmetros $\hat{y}^1 \dots \hat{y}^q$ sejam as variáveis independentes. A ideia é selecionar as variáveis independentes de modo que o erro quadrático médio da combinação linear seja minimizado em relação às classificações conhecidas no conjunto retido. Os fundamentos do modelo de regressão linear são discutidos na seção 4.4.5 do Capítulo 4, embora em um contexto diferente.

A principal diferença aqui está em termos de como as variáveis dependentes e independentes são

definido e em termos de como o problema de regressão linear é formulado. Neste caso, as variáveis independentes correspondem às previsões de classificação de vários modelos para a entrada (u, j) , e a variável dependente corresponde ao valor de cada classificação prevista \hat{r}_{uj} da combinação do conjunto no conjunto mantido H . Portanto, cada classificação observada no conjunto mantido fornece um exemplo de treinamento para o modelo de regressão linear. Os coeficientes de regressão correspondem aos pesos de vários modelos de componentes e precisam ser aprendidos a partir dos exemplos de treinamento (mantidos). Após os pesos terem sido aprendidos usando regressão linear, os modelos de componentes individuais são retreinados em todo o conjunto de treinamento sem nenhuma entrada mantida. Os pesos, que foram aprendidos usando as entradas mantidas, são usados em conjunto com esses modelos q . É importante não esquecer esta etapa final para garantir que o aprendizado máximo seja obtido de todas as informações disponíveis nas classificações. A abordagem de regressão linear para combinação de modelos é discutida em [266].

Uma abordagem relacionada, que pode fazer bom uso de todo o conhecimento dos dados de treinamento, é a validação cruzada. Os métodos de validação cruzada são discutidos no Capítulo 7.

Embora muitos sistemas simplesmente calculem a média dos resultados de múltiplos modelos, o uso da regressão é importante para garantir que os diversos modelos sejam ponderados adequadamente. Esses algoritmos baseados em regressão foram incluídos entre muitas das inscrições de alto desempenho no concurso Netflix Prize [311, 554] e estão intimamente relacionados ao conceito de empilhamento na classificação de dados.

A abordagem de regressão linear é, no entanto, sensível à presença de ruído e valores discrepantes. Isso ocorre porque a função de erro quadrático é excessivamente influenciada pelos maiores erros nos dados. Uma variedade de métodos de regressão robustos está disponível, os quais são mais resistentes à presença de ruído e outliers. Um desses métodos usa o erro absoluto médio (MAE) como função objetivo, em oposição ao erro quadrático médio. O MAE é bem conhecido por ser mais robusto a ruído e outliers porque não enfatiza excessivamente grandes erros. Uma abordagem comum é usar o método de descida do gradiente para determinar o valor ótimo do vetor de parâmetros $(\hat{y}_1 \dots \hat{y}_q)$ da Equação 6.3. O algoritmo começa definindo $\hat{y}_1 = \hat{y}_2 = \dots = \hat{y}_q = 1/q$. Posteriormente, o gradiente é calculado sobre as entradas mantidas em H da seguinte forma:

$$\frac{\partial \text{MAE}(\hat{y})}{\partial \hat{y}_i} = \frac{\sum_{(u,j) \in H} \frac{\hat{y}_j (\hat{r}_{uj} - \hat{y}_i)}{|H|}}{\sum_{(u,j) \in H} \hat{y}_i} \quad (6.4)$$

O valor de \hat{r}_{uj} pode ser expandido usando a Equação 6.3, e a derivada parcial pode ser simplificada em termos das classificações dos componentes individuais do conjunto da seguinte forma:

$$\frac{\partial \text{MAE}(\hat{y})}{\partial \hat{y}_i} = \frac{\sum_{(u,j) \in H} \text{sinal}(\hat{r}_{uj} - \hat{y}_i) \cdot r_{uj}}{|H|} \quad (6.5)$$

O gradiente pode ser escrito em termos das derivadas parciais individuais:

$$\frac{\partial \text{MAE}}{\partial \hat{y}_i} = \frac{\partial \text{MAE}(\hat{y})}{\partial \hat{y}_1} \dots \frac{\partial \text{MAE}(\hat{y})}{\partial \hat{y}_q}$$

Este gradiente é então usado para descer através do espaço de parâmetros \hat{y} com uma abordagem de descida de gradiente iterativa como segue:

1. Initialize $\hat{y}(0) = (1/q \dots 1/q)$ e $t = 0$.
2. Etapa iterativa 1: Atualizar $\hat{y}(t+1) \leftarrow \hat{y}(t) - \eta \cdot \nabla \text{MAE}$. O valor de $\eta > 0$ pode ser determinado usando uma busca linear para que a melhoria máxima em MAE seja alcançada.

3. Etapa iterativa 2: atualize o índice de iteração como $t \leftarrow t + 1$.
4. Etapa iterativa 3 (verificação de convergência): se o MAE tiver melhorado em pelo menos um mini-valor máximo desde a última iteração e, em seguida, vá para a etapa iterativa 1.
5. Relate $\hat{y}(t)$.

Regularização pode ser adicionada para evitar overfitting. Também é possível adicionar outras restrições aos vários valores de y_i , como não negatividade ou garantir que a soma seja igual a 1.

Essas restrições naturais melhoram a generalização para entradas não identificadas. As equações de descida do gradiente podem ser modificadas com relativa facilidade para respeitar essas restrições. Após a determinação dos pesos ótimos, todos os modelos de conjunto são retreinados em toda a matriz de classificações, sem nenhuma entrada retida. As previsões desses modelos são combinadas com o uso do vetor de pesos descoberto pela abordagem iterativa.

Existem outras maneiras de realizar buscas de parâmetros. Uma abordagem mais simples é tentar diversas combinações de parâmetros criteriosamente escolhidas em um conjunto de classificações. Por exemplo, pode-se ajustar os vários valores de y_i em sucessão, tentando valores diferentes e mantendo os outros constantes. Essa abordagem é geralmente aplicada a vários tipos de ajuste de parâmetros [311] e pode frequentemente fornecer resultados razoavelmente precisos. Exemplos de várias técnicas de busca são fornecidos em [162, 659].

Esses métodos podem ser aprimorados ainda mais com diferentes tipos de recursos de conteúdo de meta-nível [65, 66, 554]. Esses métodos são discutidos na seção 6.8.2. Muitos dos métodos de conjunto existentes não utilizam esses esquemas sofisticados de combinação. Frequentemente, essas técnicas utilizam uma média simples das previsões de diferentes componentes. É particularmente importante ponderar os diferentes componentes quando os valores de utilidade previstos estão em escalas diferentes ou quando alguns dos componentes do conjunto são muito mais precisos do que outros. A seguir, forneceremos exemplos específicos de como diferentes tipos de modelos são frequentemente combinados.

6.3.1 Vários tipos de combinações de modelos

Em combinações de modelos ponderados, uma variedade de mecanismos de recomendação podem ser combinados. Normalmente, há duas formas de combinações de modelos:

1. Tipos de dados e classes de modelos homogêneos: Neste caso, diferentes modelos são aplicados aos mesmos dados. Por exemplo, pode-se aplicar vários mecanismos de filtragem colaborativa, como métodos baseados em vizinhança, SVD e técnicas de Bayes, em uma matriz de classificação. Os resultados são então agregados em um único valor previsto. Tal abordagem é robusta porque evita o viés específico de algoritmos específicos em um determinado conjunto de dados, mesmo que todos os modelos constituintes pertençam à mesma classe (por exemplo, métodos colaborativos). Um exemplo dessa combinação é fornecido em [266]. Foi demonstrado em [637] como a combinação de um conjunto de três métodos diferentes de fatoração de matrizes pode fornecer resultados de alta qualidade. Em particular, a fatoração de matriz regularizada, a fatoração de matriz não negativa e a fatoração de margem máxima foram usadas como componentes do conjunto, e os resultados correspondentes foram calculados em média. Um interessante conjunto de fusão, discutido em [67], usa o mesmo algoritmo de recomendação para vários componentes do conjunto, mas com diferentes escolhas de parâmetros ou escolhas de design algorítmico. Por exemplo, diferentes números de fatores latentes podem ser usados em um algoritmo SVD, diferentes números de vizinhos mais próximos podem ser usados em um algoritmo baseado em vizinhança, ou a escolha da métrica de similaridade pode ser variada. Uma média simples

das previsões de vários sistemas. Como demonstrado em [67], essa abordagem simples quase sempre melhorou o desempenho do modelo base. Uma variação anterior dessa abordagem [180] utiliza conjuntos de métodos de fatoração de matrizes de margem máxima, mas com diferentes configurações de parâmetros. O trabalho em [338] combina um algoritmo de vizinhança baseado em usuário e um algoritmo de vizinhança baseado em item.

2. Tipos de dados e classes de modelos heterogêneos: Nesses casos, diferentes classes de modelos são aplicadas a diferentes fontes de dados. Por exemplo, um componente do modelo pode ser um recomendador colaborativo que utiliza uma matriz de classificação, enquanto outro componente do modelo pode ser um recomendador baseado em conteúdo. Essa abordagem essencialmente funde o poder de múltiplas fontes de dados no processo de combinação. A ideia é alavancar o conhecimento complementar nas diversas fontes de dados para fornecer as recomendações mais precisas. Por exemplo, o trabalho em [659] combina um recomendador colaborativo e baseado em conhecimento, enquanto o trabalho em [162] combina um recomendador baseado em conteúdo e colaborativo. Ao trabalhar com diferentes tipos de dados, torna-se particularmente importante ponderar cuidadosamente as previsões dos vários componentes do conjunto.

Essas diferentes formas de modelos oferecem excelente flexibilidade na exploração de vários tipos de combinações de modelos.

6.3.2 Adaptação do ensacamento da classificação

Conforme discutido anteriormente neste capítulo, os resultados teóricos sobre o trade-off viés-variância também se aplicam ao problema de filtragem colaborativa, pois este último é uma generalização direta da classificação. Uma das técnicas comuns de combinação ponderada usadas no problema de classificação é a de bagging. Portanto, esse método também pode ser usado na filtragem colaborativa. No entanto, a abordagem de bagging precisa ser ligeiramente modificada para se adaptar ao fato de que o problema de filtragem colaborativa é formulado de forma um pouco diferente daquela da classificação. Primeiramente, discutiremos o bagging no contexto da classificação.

A ideia básica do bagging é reduzir o componente de variância do erro na classificação. No bagging, q conjuntos de dados de treinamento são criados com amostragem bootstrapped. Na amostragem bootstrapped, as linhas da matriz de dados são amostradas com substituição para criar um novo conjunto de dados de treinamento do mesmo tamanho que o conjunto de dados de treinamento original. Esse novo conjunto de dados de treinamento normalmente contém muitas duplicatas. Além disso, pode-se demonstrar que a fração esperada de linhas da matriz de dados original que não está representada em uma determinada amostra bootstrapped é dada por $1/e$, onde e é a base do logaritmo natural. Um total de q modelos de treinamento são criados com cada um dos conjuntos de dados de treinamento amostrados. Para uma determinada instância de teste, a previsão média desses q modelos é relatada. O bagging geralmente melhora a precisão da classificação porque reduz o componente de variância do erro.

Uma variante específica do bagging, conhecida como subagging [111, 112], subamostra as linhas, em vez de realizar a amostragem com substituição. Por exemplo, pode-se simplesmente usar todas as linhas distintas em uma amostra bootstrap para treinar os modelos. Os métodos de bagging e subagging podem ser generalizados para filtragem colaborativa da seguinte forma:

1. Bootstrapping por linha: Neste caso, as linhas da matriz de classificações R são amostradas com substituição para criar uma nova matriz de classificações com as mesmas dimensões. Um total de q matrizes de classificações $R_1 \dots R_q$ são criadas. Observe que as linhas podem ser duplicadas no processo de amostragem, embora sejam tratadas como linhas separadas. Um algoritmo de filtragem colaborativa existente (por exemplo, modelo de fator latente) é então aplicado a cada uma delas.

os conjuntos de dados de treinamento q. Para cada conjunto de dados de treinamento, uma classificação de item pode ser prevista para um usuário somente se esse usuário estiver representado pelo menos uma vez na matriz. Nesse caso, a classificação prevista desse componente do conjunto é a classificação média³ desse item sobre as ocorrências duplicadas desse usuário. A classificação prevista é então calculada como a média de todos os componentes do conjunto nos quais esse usuário está presente. Observe que, para valores razoavelmente grandes de q, cada usuário normalmente estará presente em pelo menos um componente do conjunto com um valor de alta probabilidade de $1/(1/e)^q$. Portanto, todos os usuários serão representados com alta probabilidade.

2. Subamostragem por linha: Esta abordagem é semelhante ao bootstrapping por linha, exceto que as linhas são amostradas sem reposição. A fração f de linhas amostradas é escolhida aleatoriamente entre (0,1, 0,5). O número de componentes do conjunto q deve ser significativamente maior que 10 para garantir que todas as linhas sejam representadas. O principal problema com essa abordagem é que é difícil prever todas as entradas nesse cenário e, portanto, é necessário calcular a média de um número menor de componentes. Portanto, os benefícios da redução da variância não são totalmente alcançados.
3. Ensacamento por entradas: Neste caso, as entradas da matriz de classificações original são amostradas com substituição para criar as q matrizes de classificações diferentes R1 ...Rq. Como muitas entradas podem ser amostradas repetidamente, elas agora são associadas a pesos. Portanto, é necessário um algoritmo de filtragem colaborativa básico que possa lidar com entradas com pesos. Tais algoritmos são discutidos na seção 6.5.2.1. Assim como no caso do ensacamento por linhas, as classificações previstas são calculadas como médias sobre os vários componentes do conjunto.
4. Subamostragem por entrada: Na subamostragem por entrada, uma fração das entradas é retida aleatoriamente da matriz de classificações R para criar um conjunto de dados de treinamento amostrado. Normalmente, um valor de f é amostrado de (0,1, 0,5) e, em seguida, uma fração f das entradas na matriz de classificações original é escolhida aleatoriamente e retida. Essa abordagem é repetida para criar q conjuntos de dados de treinamento R1 ...Rq. Assim, cada usuário e cada item são representados em cada matriz subamostrada, mas o número de entradas especificadas na matriz subamostrada é menor do que nos dados de treinamento originais. Um algoritmo de filtragem colaborativa (por exemplo, modelo de fator latente) é aplicado a cada matriz de classificações para criar uma matriz predita. A predição final é a média simples dessas q previsões diferentes.

Nos métodos mencionados, a etapa final do conjunto utiliza uma média simples das previsões, em vez de uma média ponderada. A razão para usar uma média simples é que todos os componentes do modelo são criados com uma abordagem probabilística idêntica e, portanto, devem ser ponderados igualmente. Em muitos desses casos, é importante escolher métodos de base instável para obter bons ganhos de desempenho.

Embora a discussão acima forneça uma visão geral das diversas possibilidades de redução de variância, apenas um pequeno subconjunto dessas possibilidades foi realmente explorado e avaliado na literatura de pesquisa. Por exemplo, não temos conhecimento de quaisquer resultados experimentais sobre a eficácia de métodos de subamostragem. Embora os métodos de subamostragem frequentemente forneçam resultados superiores ao bagging no domínio de classificação [658], seu efeito na filtragem colaborativa é difícil de prever em matrizes esparsas. Em matrizes esparsas, a eliminação de entradas pode levar à incapacidade de prever alguns usuários ou itens, o que pode, às vezes, piorar o desempenho geral. Uma discussão sobre algoritmos de bagging no contexto de métodos colaborativos

³É possível que valores não especificados em linhas duplicadas sejam previstos de forma diferente, embora isso seja relativamente incomum para a maioria dos algoritmos de filtragem colaborativa.

a filtragem pode ser encontrada em [67]. Neste trabalho, uma abordagem de bootstrapping linha a linha é usada, e linhas duplicadas são tratadas como linhas ponderadas. Portanto, a abordagem assume que o preditor base pode lidar com linhas ponderadas. Conforme discutido em [67], melhorias significativas no erro foram alcançadas com bagging, embora a abordagem parecesse ser um pouco sensível à escolha do preditor base. Em particular, de acordo com os resultados em [67], a abordagem bagging melhorou a precisão sobre a maioria dos preditores base, com exceção do modelo de vizinhança fatorada [72]. Isso pode ser possivelmente resultado de um alto nível de correlação entre as previsões dos vários modelos bagged, quando o método de vizinhança fatorada é usado. Em geral, é desejável usar modelos base não correlacionados com baixo viés e alta variância para extrair o máximo benefício do bagging. Nos casos em que o bagging não funciona devido às altas correlações entre os preditores base, pode ser útil usar explicitamente a injeção de aleatoriedade.

6.3.3 Injeção de Aleatoriedade

A injeção de aleatoriedade é uma abordagem que compartilha muitos princípios de florestas aleatórias na classificação [22]. A ideia básica é tomar um classificador base e injetar aleatoriedade explicitamente nele. Vários métodos podem ser usados para injetar aleatoriedade. Alguns exemplos [67] são os seguintes:

1. Injetando aleatoriedade em um modelo de vizinhança: Em vez de usar os k vizinhos mais próximos (usuários ou itens) em um modelo de vizinhança baseado em usuários ou itens, os vizinhos $\hat{y} \cdot k$ mais próximos são selecionados para $\hat{y} \neq 1$. Em seguida, k elementos são selecionados aleatoriamente desses $\hat{y} \cdot k$ vizinhos. Essa abordagem pode, no entanto, ser demonstrada como uma variante indireta da subamostragem por linha no fator $1/\hat{y}$. A previsão média dos vários componentes é retornada pela abordagem.
2. Injetando aleatoriedade em um modelo de fatoração matricial: Os métodos de fatoração matricial são inherentemente randomizados, pois realizam um gradiente descendente sobre o espaço de soluções após a inicialização aleatória das matrizes de fatores. Portanto, ao escolher diferentes inicializações, soluções diferentes são frequentemente obtidas. As combinações dessas diferentes soluções geralmente fornecem resultados mais precisos.

Uma média simples das previsões dos diferentes componentes é retornada pelo conjunto aleatório. Assim como as florestas aleatórias, essa abordagem pode reduzir a variância do conjunto sem afetar significativamente o viés. Em muitos casos, essa abordagem funciona muito bem quando o bagging não funciona devido ao alto nível de correlação entre os vários preditores.

Conforme demonstrado em [67], a abordagem de injeção de aleatoriedade funciona muito bem quando o modelo de vizinhança fatorado é usado como preditor base [72]. Vale ressaltar que a abordagem de ensacamento não funciona muito bem no caso do modelo de vizinhança fatorado.

6.4 Trocando Híbridos

Híbridos de comutação são usados mais comumente em sistemas de recomendação no contexto do problema de seleção de modelos, mas muitas vezes não são formalmente reconhecidos como sistemas híbridos.

A motivação original para a troca de sistemas [117] foi lidar com o problema de inicialização a frio, em que um modelo específico funciona melhor em estágios iniciais, quando há escassez de dados disponíveis. No entanto, em estágios posteriores, um modelo diferente é mais eficaz e, portanto, a troca é feita para o modelo mais eficaz.

Também é possível visualizar modelos de comutação no sentido mais geral de seleção de modelos. Por exemplo, até mesmo a etapa de seleção de parâmetros da maioria dos modelos de recomendação exige a execução do modelo em vários valores de parâmetros e, em seguida, a seleção do valor ideal. Essa forma específica de seleção de modelos é adaptada da literatura de classificação e também é chamada de "balde de modelos". A seguir, discutiremos esses dois tipos de híbridos.

6.4.1 Mecanismos de comutação para problemas de partida a frio

Mecanismos de alternância são frequentemente utilizados para lidar com o problema de inicialização a frio, no qual um recomendador tem melhor desempenho com menos dados, enquanto o outro recomendador tem melhor desempenho com mais dados. Pode-se utilizar um recomendador baseado em conhecimento quando há poucas classificações disponíveis, pois os sistemas de recomendação baseados em conhecimento podem funcionar sem nenhuma classificação e dependem das especificações do usuário quanto às suas necessidades. No entanto, à medida que mais classificações se tornam disponíveis, pode-se migrar para um recomendador colaborativo. Também é possível combinar recomendadores baseados em conteúdo e colaborativos dessa forma, pois os recomendadores baseados em conteúdo podem funcionar bem para novos itens, enquanto os recomendadores colaborativos não conseguem fornecer recomendações eficazes para novos itens.

O trabalho em [85] propõe o sistema Daily Learner no qual vários recomendadores são usados em uma estratégia ordenada. Se recomendações suficientes não forem encontradas por recomendadores anteriores, então recomendadores posteriores são usados. Em particular, o trabalho em [85] usa dois recomendadores baseados em conteúdo e um único recomendador colaborativo. Primeiro, um classificador de conteúdo vizinho mais próximo é usado, seguido por um sistema colaborativo e, finalmente, um classificador de conteúdo Bayes ingênuo é usado para corresponder ao perfil de longo prazo. Essa abordagem não aborda completamente o problema de inicialização a frio porque todos os alunos subjacentes precisam de alguma quantidade de dados. Outro trabalho [659] combina versões híbridas de sistemas colaborativos e baseados em conhecimento. O sistema baseado em conhecimento fornece resultados mais precisos durante a fase de inicialização a frio, enquanto o sistema colaborativo fornece resultados mais precisos em estágios posteriores.

A incorporação de sistemas baseados em conhecimento é geralmente mais desejável para lidar com o problema de inicialização a frio.

6.4.2 Balde de Modelos

Nesta abordagem, uma fração (por exemplo, 25% a 33%) das entradas especificadas na matriz de classificações são mantidas, e vários modelos são aplicados à matriz resultante. As entradas mantidas são então usadas para avaliar a eficácia do modelo em termos de uma medida padrão, como o MSE ou o MAE. O modelo que produz o menor MSE ou MAE é usado como o relevante. Esta abordagem também é comumente usada para ajuste de parâmetros. Por exemplo, cada modelo pode corresponder a um valor diferente do parâmetro do algoritmo, e o valor que fornece o melhor resultado é selecionado como o relevante. Uma vez que o modelo relevante tenha sido selecionado, ele é retreinado em toda a matriz de classificações, e os resultados são relatados. Em vez de usar uma abordagem de retenção, uma técnica diferente conhecida como validação cruzada também é usada. Você aprenderá mais sobre técnicas de validação cruzada e hold-out no Capítulo 7. O balde de modelos é a abordagem de conjunto mais útil em sistemas de recomendação, embora raramente seja reconhecido como um sistema de conjunto, a menos que os diferentes modelos sejam derivados de tipos de dados heterogêneos. Quando o balde de modelos é usado no contexto de uma matriz de classificações que muda dinamicamente, é possível que o sistema alterne de um componente para outro. No entanto, quando usado para dados estáticos, o sistema também pode ser visto como um caso especial de recomendadores ponderados, em que o peso de um componente é definido como 1 e os pesos dos componentes restantes são definidos como 0.

6.5 Híbridos em Cascata

No trabalho original de Burke [117], híbridos em cascata foram definidos de forma um tanto restrita, na qual cada recomendador refina ativamente as recomendações feitas pelo recomendador anterior. Aqui, adotamos uma visão mais ampla dos híbridos em cascata, na qual um recomendador pode usar as recomendações do recomendador anterior de qualquer forma (além do refinamento direto) e, então, combinar os resultados para elaborar a recomendação final. Essa definição mais ampla abrange classes maiores de híbridos importantes, como o reforço, que de outra forma não seriam incluídos em nenhuma das categorias de híbridos. Consequentemente, definimos duas categorias diferentes de recomendadores em cascata.

6.5.1 Refinamento sucessivo das recomendações

Nessa abordagem, um sistema de recomendação refina sucessivamente a saída de recomendações da iteração anterior. Por exemplo, o primeiro recomendador pode fornecer uma classificação aproximada e também eliminar muitos dos itens potenciais. O segundo nível de recomendação usa essa classificação aproximada para refiná-la ainda mais e desempatar. A classificação resultante é apresentada ao usuário. Um exemplo desse sistema de recomendação é o EntreeC [117], que usa o conhecimento dos interesses declarados do usuário para fornecer uma classificação aproximada. As recomendações resultantes são então divididas em grupos de preferência aproximadamente igual. As recomendações dentro de um grupo são, portanto, consideradas empatadas ao final do primeiro estágio.

Uma técnica colaborativa é usada para desempatar e classificar as recomendações dentro de cada grupo. O primeiro recomendador baseado em conhecimento tem claramente maior prioridade, pois o recomendador de segundo nível não pode alterar as recomendações feitas no primeiro nível.

A outra observação é que o recomendador de segundo nível é muito mais eficiente porque precisa se concentrar apenas nos vínculos dentro de cada grupo. Portanto, o espaço de itens de cada aplicação do recomendador de segundo nível é muito menor.

6.5.2 Reforço

O reforço tem sido usado popularmente no contexto de classificação [206] e regressão [207].

Um dos primeiros métodos de reforço foi o algoritmo AdaBoost [206]. A variante de regressão desse algoritmo é chamada de AdaBoost.RT [207]. A variante de regressão é mais relevante para a filtragem colaborativa, pois facilita o tratamento de classificações como atributos numéricos. No reforço tradicional, uma sequência de rodadas de treinamento é usada com exemplos de treinamento ponderados.

Os pesos em cada rodada são modificados dependendo do desempenho do classificador na rodada anterior. Especificamente, os pesos nos exemplos de treinamento com erro são aumentados, enquanto os pesos nos exemplos modelados corretamente são reduzidos. Como resultado, o classificador tende a classificar corretamente os exemplos que não conseguiu classificar corretamente na rodada anterior. Ao usar várias dessas rodadas, obtém-se uma sequência de modelos de classificação. Para uma determinada instância de teste, todos os modelos são aplicados a ela, e a previsão ponderada é relatada como a relevante.

O reforço precisa ser modificado para funcionar na filtragem colaborativa, na qual não há uma demarcação clara entre as linhas de treinamento e teste, e também não há uma distinção clara entre as colunas dependentes e independentes. Um método para modificar o reforço para filtragem colaborativa é proposto em [67]. Ao contrário da classificação e da modelagem de regressão, nas quais os pesos são associados às linhas, os pesos dos exemplos de treinamento na filtragem colaborativa são associados às classificações individuais. Portanto, se o conjunto S representa o conjunto de classificações observadas nos dados de treinamento, então um total de $|S|$ pesos é mantido. Observe que

S é um conjunto de posições (u, j) na matriz de classificações $m \times n R$, tal que r_{uj} é observado. Supõe-se também que o algoritmo de filtragem colaborativa base tenha capacidade para trabalhar com classificações ponderadas (cf. seção 6.3). Em cada iteração, os pesos de cada uma dessas classificações são modificados dependendo da capacidade do algoritmo de filtragem colaborativa de prever aquela entrada específica.

O algoritmo geral é aplicado para um total de T iterações. Na t -ésima iteração, o peso associado à entrada (u, j) -ésima da matriz de classificações é denotado por $W_t(u, j)$. O algoritmo começa ponderando igualmente cada entrada e prevê todas as classificações usando um modelo de base. A previsão de uma entrada $(u, j) \rightarrow S$ é considerada "incorrecta" se a classificação prevista \hat{r}_{uj} variar da classificação real r_{uj} em pelo menos um valor predefinido δ . A taxa de erro na t -ésima iteração é calculada como a fração de classificações especificadas em S para as quais o valor previsto está incorreto, de acordo com esta definição. Os pesos dos exemplos previstos corretamente, enquanto os pesos dos exemplos previstos incorretamente, são reduzidos multiplicando-os pelos α_t . Em cada iteração, os pesos são sempre normalizados para $\sum_t \alpha_t$. Exemplos previstos, permanecem inalterados. Somar 1. Portanto, os pesos relativos das entradas classificadas incorretamente sempre aumentam ao longo das várias iterações. O modelo de base é aplicado novamente aos dados reponderados.

Esta abordagem é repetida para T iterações, a fim de criar T previsões diferentes para as entradas não especificadas. A média ponderada dessas T previsões diferentes é usada como a previsão final de uma entrada, onde o peso da t -ésima previsão é $\log t$. Vale ressaltar que as regras de atualização de peso e combinação de modelos em [67] são ligeiramente diferentes daquelas usadas na modelagem de classificação e regressão. No entanto, há muito poucos estudos nesta área, além do trabalho em [67], sobre o uso de métodos de reforço para filtragem colaborativa. É concebível que as estratégias simples propostas em [67] possam ser ainda mais aprimoradas com experimentação.

6.5.2.1 Modelos de base ponderados

Os métodos de boosting e bagging exigem o uso de modelos base ponderados, nos quais as entradas são associadas a pesos. Nesta seção, mostramos como os modelos de filtragem colaborativa existentes podem ser modificados para que funcionem com pesos.

Vamos supor que o peso w_{uk} esteja associado a uma entrada específica na matriz de classificações para o usuário u e o item k . É relativamente simples modificar modelos existentes para trabalhar com pesos nas entradas:

1. Algoritmos baseados em vizinhança: A avaliação média de um usuário é calculada de forma ponderada para centralizar as avaliações na média. Tanto as medidas de Pearson quanto as medidas de cosseno podem ser modificadas para levar em conta os pesos. Portanto, a Equação 2.2 do Capítulo 2 pode ser modificada da seguinte forma para calcular o coeficiente de Pearson entre os usuários u e v :

$$\text{Pearson}(u, v) = \frac{\sum_{k \in \text{V}(u)} w_{uk} \cdot \sum_{k \in \text{V}(v)} w_{vk} - \sum_{k \in \text{V}(u)} w_{uk} \cdot \sum_{k \in \text{V}(v)} w_{vk}}{\sqrt{\sum_{k \in \text{V}(u)} w_{uk}^2} \sqrt{\sum_{k \in \text{V}(v)} w_{vk}^2}} \quad (6.6)$$

O leitor deve consultar a seção 2.3 do Capítulo 2 para obter detalhes das notações.

Uma maneira diferente⁴ de modificar a medida é a seguinte:

$$\text{Pearson}(u, v) = \frac{\sum_{k \in \text{V}(u)} w_{uk} \cdot \sum_{k \in \text{V}(v)} w_{vk} - \sum_{k \in \text{V}(u)} w_{uk} \cdot \sum_{k \in \text{V}(v)} w_{vk}}{\sqrt{\sum_{k \in \text{V}(u)} w_{uk}^2} \sqrt{\sum_{k \in \text{V}(v)} w_{vk}^2}} \quad (6.7)$$

⁴O trabalho em [67] propõe apenas a primeira técnica para calcular a similaridade.

Para medidas de similaridade item-item, a medida de cosseno ajustada pode ser modificada de forma semelhante. Essas medidas de similaridade ponderadas são usadas tanto para calcular os vizinhos mais próximos quanto para a média (ponderada) das avaliações no grupo de pares.

- Modelos de fatores latentes: Os modelos de fatores latentes são definidos como problemas de otimização nos quais a soma dos quadrados dos erros das entradas especificadas é minimizada.

Neste caso, a soma ponderada dos quadrados do problema de otimização deve ser minimizada. Portanto, a função objetivo na seção 3.6.4.2 do Capítulo 3 pode ser modificada da seguinte forma:

$$\text{Minimize } J = \frac{1}{(i,j) \in S} \sum_{i=1}^m \sum_{j=1}^k \left| \hat{y}_{ij} - \sum_{s=1}^n u_{is} v_{js} \right|^2 \quad (6.8)$$

Aqui, $U = [u_{is}]$ e $V = [v_{js}]$ são as matrizes $m \times k$ e $n \times k$ de fator de usuário e fator de item, respectivamente. Observe os pesos associados aos erros nas entradas.

A mudança correspondente no método de descida do gradiente é ponderar as atualizações relevantes:

$$\begin{aligned} & u_{is} \leftarrow u_{is} + \alpha (w_{ij} - \hat{y}_{ij}) v_{js} \\ & v_{js} \leftarrow v_{js} + \alpha (w_{ij} - \hat{y}_{ij}) u_{is} \end{aligned}$$

Muitos outros algoritmos básicos de filtragem colaborativa podem ser modificados para funcionar com entradas ponderadas. Esses tipos de algoritmos básicos ponderados são úteis para muitos conjuntos de filtragem colaborativa, como reforço e ensacamento.

6.6 Híbridos de Aumento de Recursos

O híbrido de aumento de características compartilha uma série de semelhanças intuitivas com o conjunto de empilhamento na classificação. No empilhamento [634], o classificador de primeiro nível é usado para criar ou aumentar um conjunto de características para o classificador de segundo nível. Em muitos casos, sistemas prontos para uso são usados como um conjunto. No entanto, em alguns casos, podem ser necessárias alterações no sistema de recomendação de componentes para funcionar com os dados modificados e, portanto, o sistema híbrido não é um verdadeiro conjunto de sistemas prontos para uso.

O sistema Libra [448] combina o sistema de recomendação da Amazon.com com seu próprio classificador Bayes. A abordagem usa os "autores relacionados" e "títulos relacionados" que a Amazon gera como características que descrevem os itens. Observe que a Amazon gera essas recomendações com o uso de um sistema de recomendação colaborativo. Esses dados são então usados em conjunto com um recomendador baseado em conteúdo para fazer as previsões finais. Observe que qualquer sistema baseado em conteúdo pronto para uso pode ser usado em princípio e, portanto, a abordagem pode ser vista como um sistema de conjunto. A abordagem em [448] opta por um classificador de texto Bayes ingênuo. Foi descoberto por meio de experimentos que as características geradas pelo sistema colaborativo da Amazon eram de alta qualidade e contribuíram significativamente para recomendações de melhor qualidade.

Em vez de usar um sistema colaborativo primeiro, também é possível usar o sistema baseado em conteúdo primeiro. A ideia básica é usar o sistema baseado em conteúdo para preencher as entradas ausentes na matriz de classificações, de modo que ela não seja mais esparsa. Assim, as entradas ausentes são estimadas pelo sistema baseado em conteúdo para criar uma matriz de classificações mais densa. Essas classificações recém-adicionadas são chamadas de pseudoclassificações. Em seguida, um recomendador colaborativo é usado na matriz de classificações densa para fazer previsões de classificação. Finalmente, a previsão colaborativa é combinada.

com a previsão original baseada no conteúdo de forma ponderada para produzir a previsão geral de as entradas ausentes na matriz [431]. A incorporação das classificações ausentes na primeira fase permite uma aplicação mais robusta da segunda fase em termos de computação de similaridade. No entanto, o cálculo de similaridade precisa ser modificado para dar menos peso às pseudoclassificações em comparação às classificações reais. Isso ocorre porque as pseudoclassificações foram inferidas e pode ser propenso a erros.

Como tais pesos podem ser determinados? O peso de uma pseudo-classificação representa intuitivamente a certeza do algoritmo na previsão da primeira fase, e é uma variável crescente. função do número de avaliações $|I|$ desse usuário. Uma série de funções heurísticas são usadas para ponderar várias classificações, e o leitor é remetido a [431] para mais detalhes. Observe que esta abordagem requer modificações na segunda fase da filtragem colaborativa e na implementação de soluções prontas para uso. algoritmos não podem ser utilizados. Tais métodos podem ser vistos como sistemas monolíticos.

O aumento de recursos tem uma longa história em sistemas de recomendação. Um dos primeiros exemplos de aumento de recursos foi implementado no contexto do GroupLens.

sistema [526], no qual um sistema baseado em conhecimento foi usado para criar um banco de dados de classificações artificiais. Os agentes, conhecidos como robôs de filtro, usaram critérios específicos, como o número de erros de ortografia ou o tamanho da mensagem para atribuir classificações aos itens, enquanto agem como usuários artificiais. Posteriormente, essas classificações foram utilizadas no contexto de um sistema colaborativo para fazer recomendações.

6.7 Híbridos de Meta-Nível

Em um híbrido de meta-nível, o modelo aprendido por um recomendador é usado como entrada para o próximo nível. Um exemplo importante de colaboração via conteúdo foi o trabalho inicial de Paz-zani [475]. Um modelo baseado em conteúdo [363] é construído que descreve a discriminação recursos que preveem restaurantes. Os recursos discriminativos podem ser determinados usando qualquer dos métodos de seleção de recursos discutidos na seção 4.3 do Capítulo 4. Cada usuário é definido por uma representação vetorial de palavras discriminativas. Um exemplo da possível palavra-usuário A matriz para um sistema de recomendação de restaurantes é mostrada abaixo:

Palavra ã carni Usuário ã	carne	bovina	cordeiro assado	ovos fritos	
Sayani	0	3	0	2,5	1.7
John	2,3	1,3	0,2	1.4	2.1
Mary	0	2,8	0,9	1.1	2.6
Peter	2,4	1,7	0	3,5	1.9
Jack	1,6	2,2	3,1	1,0	0

Os pesos da tabela acima mencionada podem ser obtidos utilizando as descrições dos itens que o usuário acessou. Observe que as palavras irrelevantes já foram removidas porque a seleção de recursos baseada em conteúdo na primeira fase cria uma representação de espaço vetorial discriminativa para cada usuário. Além disso, a representação é significativamente mais densa do que uma matriz de classificação típica. Portanto, é possível calcular de forma robusta as semelhanças entre usuários com esta nova representação. A ideia principal aqui é que o grupo de pares baseado em conteúdo é usado para determinar os usuários mais semelhantes ao usuário-alvo. Uma vez que o grupo de pares tenha sido determinado, então a média ponderada das avaliações do grupo de pares é usada para determinar as classificações previstas. Observe que esta abordagem requer uma certa quantidade de mudança para

o recomendador colaborativo original, pelo menos em termos de como a similaridade é computada.

A formação do grupo de pares deve utilizar a matriz usuário-palavra (que foi o modelo criado na primeira fase), enquanto a recomendação final utiliza a matriz de classificações. Isso difere de um sistema colaborativo em que ambas as etapas utilizam a mesma matriz. Além disso, a primeira fase da abordagem não pode utilizar modelos baseados em conteúdo prontos para uso em sua totalidade, pois é principalmente uma fase de seleção de recursos (pré-processamento). Portanto, em muitos casos, esses sistemas não podem ser considerados verdadeiros conjuntos, pois não utilizam métodos existentes como recomendadores prontos para uso.

Outro exemplo de sistema de meta-nível foi o LaboUr [534], no qual um modelo baseado em instâncias é usado para aprender o perfil do usuário baseado em conteúdo. Os perfis são então comparados usando uma abordagem colaborativa. Esses modelos são comparados entre usuários para fazer previsões.

Muitos desses métodos se enquadram na categoria de “colaboração via conteúdo”, embora essa não seja a única maneira pela qual tais híbridos podem ser construídos.

6.8 Híbridos de combinação de recursos

Em híbridos de combinação de características, a ideia é combinar os dados de entrada de várias fontes (por exemplo, conteúdo e colaboração) em uma representação unificada antes de aplicar um algoritmo preditivo. Na maioria dos casos, esse algoritmo preditivo é um algoritmo baseado em conteúdo que usa informações colaborativas como características adicionais. Um exemplo de tal abordagem foi apresentado em [69], onde o classificador RIPPER foi aplicado ao conjunto de dados aumentado. Foi demonstrado em [69] que a metodologia alcançou melhorias significativas em relação a uma abordagem puramente colaborativa. No entanto, as características de conteúdo precisam ser escolhidas manualmente para atingir esse resultado. Portanto, a abordagem pode ser sensível à escolha do conjunto de dados e da representação das características. A abordagem reduz a sensibilidade do sistema ao número de usuários que classificaram um item. Esta é, obviamente, a propriedade de qualquer sistema baseado em conteúdo, que é robusto ao problema de inicialização a frio da perspectiva de novos itens.

Observe que é possível realizar a combinação de diversas maneiras, com diferentes tipos de conhecimento prévio. Por exemplo, considere o caso em que cada item está associado a uma taxonomia de nível superior que representa os gêneros dos itens. O perfil de representação do usuário e dos itens pode ser ampliado em termos dos gêneros relevantes na hierarquia. A matriz de classificações pode então ser construída em termos de gêneros em vez de itens. Em matrizes esparsas, essa abordagem pode fornecer resultados mais eficazes porque reduz o número de colunas e porque a maioria das entradas provavelmente será preenchida na matriz compactada.

Outra abordagem é aumentar uma matriz de classificações e adicionar colunas para palavras-chave, além de itens. Portanto, a matriz de classificações se torna uma matriz $m \times (n + d)$, onde n é o número de itens e d é o número de palavras-chave. Os pesos dos "itens de palavras-chave" são baseados na agregação ponderada das descrições dos itens acessados, comprados ou avaliados pelo usuário. Uma abordagem tradicional de fatoração de vizinhança ou matriz pode ser usada com essa matriz aumentada. Os pesos relativos dos dois tipos de colunas podem ser aprendidos por meio de validação cruzada (consulte o Capítulo 7). Esse tipo de combinação de dois modelos de otimização é comum em ambientes híbridos, onde a função objetivo é configurada da seguinte forma em termos de um vetor de parâmetros \hat{y} :

$$J = \text{CollaborativeObjective}(\hat{y}) + \hat{y}^T \text{ContentObjective}(\hat{y}) + \text{Regularização} \quad (6.9)$$

A função objetivo é então otimizada sobre o vetor de parâmetros \hat{y} . Um exemplo específico, discutido abaixo, é a generalização de modelos lineares esparsos (cf. seção 2.6.5 do Capítulo 2) com informações secundárias.

6.8.1 Regressão e fatoração de matrizes

Seja R uma matriz de avaliações de feedback implícitas $m \times n$, e C uma matriz de conteúdo $ad \times n$, na qual cada item é descrito por frequências não negativas de d palavras. Exemplos incluem descrições de itens ou revisões curtas de itens. Como R é uma matriz de feedback implícita, as entradas ausentes são assumidas como 0s. Como na seção 2.6.5, seja W uma matriz de coeficientes item-item $n \times n$ na qual as avaliações são previstas como $R^* = RW$. No entanto, neste caso, também podemos prever as avaliações como $R^* = CW$. Portanto, em vez de otimizar apenas $\|R - RW\|^2$, adicionamos um termo adicional baseado em conteúdo $\|R - CW\|^2$. Juntamente com a regularização da rede elástica e as restrições de não negatividade/diagonais, o modelo de otimização aprimorado é declarado da seguinte forma [456]: Minimize $J = \|R - RW\|^2 + \gamma \cdot \|R - CW\|^2 + \gamma \|W\|^2 + \gamma_1 \cdot \|W\|_1$

sujeito a: $W \geq 0$ Diagonal(W)=0

O parâmetro de peso γ pode ser determinado em uma fase de ajuste. Embora as classificações possam ser previstas como $R^* = RW$ ou como $R^* = CW$, apenas a primeira função de previsão é utilizada.

Portanto, o termo $\|R - CW\|^2$ é usado apenas para refinar a função objetivo como um regularizador adicional. Em outras palavras, o objetivo do termo adicional é melhorar o poder de generalização do modelo para prever ações futuras (e ainda desconhecidas) do usuário. Algumas variações dessa função objetivo básica são discutidas em [456].

Este tipo de abordagem pode ser usado para combinar qualquer outro tipo de modelo de filtragem colaborativa (otimização) com métodos baseados em conteúdo. Por exemplo, no caso da fatoração matricial, pode-se usar uma matriz de fatores de usuário U $m \times k$, uma matriz de fatores de itens compartilhados $n \times k$ e uma matriz de fatores de conteúdo Z $ad \times k$ para configurar o modelo de otimização da seguinte forma [557]: $V^T \|U\|^2 + \gamma \|V\|^2 + \|Z\|^2$

$$\text{Minimize } J = \|R - UV^T\|^2 + \gamma \cdot \|C - ZV^T\|^2$$

Observe que a matriz fatorial do item V é compartilhada entre as fatorações da matriz de classificação e da matriz de conteúdo. Esses modelos de fatoração de matriz compartilhada também são usados para incorporar outros tipos de informações secundárias, como dados de confiança social (cf. seção 11.3.8 do Capítulo 11). Uma visão geral da combinação de métodos de fatoração de matrizes com modelos arbitrários é fornecida na seção 3.7.7 do Capítulo 3.

6.8.2 Recursos de nível meta

Não é necessário usar a combinação de recursos no contexto de múltiplos tipos de recomendadores (por exemplo, conteúdo e colaborativo). Novos meta-recursos podem ser extraídos de recursos de um tipo específico de recomendador e então combinados dentro do modelo de conjunto. Por exemplo, pode-se extrair recursos de nível meta de uma matriz de classificações correspondente ao número de avaliações dadas por vários usuários e itens. Quando um usuário avalia muitos filmes, ou quando um filme é avaliado por muitos usuários, isso afeta a precisão da recomendação dos vários algoritmos de maneiras diferentes. Diferentes sistemas de recomendação serão mais ou menos sensíveis a essas características e, portanto, terão um desempenho melhor ou pior para vários usuários e itens.

A ideia básica das características de meta-nível é levar em conta essas diferenças específicas de entrada no processo de combinação de modelos com o uso de meta-características. As meta-características resultantes podem ser combinadas com outros algoritmos de conjunto para criar um projeto de conjunto, que incorpora características de vários tipos de híbridos, mas não se enquadra perfeitamente em nenhuma das sete categorias originais de Burke [117]. No entanto, está mais intimamente relacionado aos híbridos de combinação de características, no sentido de que combina meta-características com classificações.

Id.	Descrição Valor
1	constante de 1 (usar apenas esse recurso equivale a usar o modelo de regressão linear global da seção 6.3)
2	Uma variável binária que indica se o usuário avaliou mais de 3 filmes nesta data específica O log do número de vezes
	que um filme foi avaliado O log do número de datas distintas em
3	que um usuário avaliou filmes
5	Uma estimativa bayesiana da classificação média do filme após ter subtraído a média estimada bayesiana do usuário 6
	O log do número de avaliações do usuário
16	O desvio padrão das avaliações do usuário
17	O desvio padrão das classificações de filmes
18	O registro de (Data de classificação - Data da primeira classificação do usuário +1)
19	O log do número de avaliações do usuário na data +1

Tabela 6.1: Um subconjunto das meta-características usadas em [554] para combinação de conjuntos no conjunto de dados do Prêmio Netflix

A abordagem de meta-recursos provou ser um método potencialmente poderoso para o projeto robusto de conjuntos. De fato, ambas as entradas vencedoras no conteúdo do Prêmio Netflix, correspondentes a Pragmatic Chaos [311] e The Ensemble [704], de Bellkor, utilizaram essa abordagem. Descreveremos o uso desses meta-recursos em algoritmos de filtragem colaborativa. Em particular, discutiremos a metodologia de empilhamento linear ponderado por recursos [554], que combina esses meta-recursos com os métodos de empilhamento discutidos anteriormente na seção 6.3.

Esta abordagem baseia-se na técnica de combinação utilizada em The Ensemble [704]. Um subconjunto das metacaracterísticas utilizadas em [554] para o processo de empilhamento no conjunto de dados do Prêmio Netflix é fornecido na Tabela 6.1 para fins ilustrativos. O identificador na coluna da esquerda corresponde ao identificador utilizado no artigo original [554]. Essas características são particularmente instrutivas, pois geralmente é possível extraír características análogas para outros conjuntos de dados de classificação. Observe que cada característica na Tabela 6.1 é específica para uma entrada na matriz de classificação.

Suponhamos que um total de I metacaracterísticas (numéricas) foram extraídas, e seus valores são $z_{ut} \dots z_{ut}$ para q par usuário-item (u, t). Portanto, as metacaracterísticas são específicas para cada entrada (u, t) na matriz de classificações, embora algumas características possam assumir os mesmos valores para valores variáveis de u ou valores variáveis de t . Por exemplo, a característica 3 na Tabela 6.1 não variará com o usuário u , mas variará com o item t .

Vamos supor que haja um total de q métodos de recomendação base, e os pesos associados aos q métodos de recomendação sejam denotados por $w_1 \dots w_q$. Então, para uma determinada entrada (u, t) na matriz de classificações, se as previsões dos q componentes forem $\hat{r}_1, \hat{r}_2, \dots, \hat{r}_q$, então a previsão \hat{r}_{ut} do conjunto geral é dada pelo seguinte:

$$\text{rota} = \sum_{i=1}^q w_i \hat{r}_i$$
(6.10)

Gostaríamos que a previsão estimada \hat{r}_{ut} do conjunto correspondesse o mais próximo possível à classificação observada r_{ut} . Observe que a abordagem na seção 6.3 usa um modelo de regressão linear para aprender os pesos $w_1 \dots w_q$, mantendo uma fração predefinida das entradas durante o processo de treinamento dos modelos q e, em seguida, usando as entradas mantidas como os valores observados no modelo de regressão linear. Tal abordagem é de empilhamento puro e pode ser considerada

um híbrido ponderado. No entanto, pode ser aprimorado ainda mais com meta-recursos. A ideia principal é que os pesos de regressão linear $w_1 \dots w_q$ são específicos para cada entrada na matriz de classificações e são eles próprios funções lineares das meta-características. Por outras palavras, os pesos agora precisam ser sobreescritos com (u, t) para levar em conta o fato de que são específicos para cada entrada (u, t) na matriz de classificações:

$$\text{rota} = \sum_{i=1}^q \text{o que } r_{\text{mas}}^i \quad (6.11)$$

Este é um modelo mais refinado porque a natureza da combinação é local para cada entrada na matriz de classificação, e não é globalmente visível em toda a matriz. O problema é que o número, $m \times n \times q$, de diferentes parâmetros w_{ut} torna-se grande demais para ser aprendido de forma robusta. Na verdade, o número de parâmetros (pesos) é maior que o número das classificações observadas, resultando em sobreajuste. Portanto, os pesos são assumidas como combinações lineares das meta-características sob a suposição de que estas As metacaracterísticas regulam a importância relativa dos vários modelos para as combinações individuais de itens-usuário. Portanto, introduzimos os parâmetros vij que regulam a importância da j -ésima meta-característica para o i -ésimo modelo. Os pesos para a entrada (u, t) agora podem ser expressos como uma combinação linear dos valores de meta-característica da entrada (u, t) da seguinte forma:

$$\text{o que } = \sum_{j=1}^l vij zut_{eu} \quad (6.12)$$

Agora podemos expressar o problema de modelagem de regressão em termos de um número menor, $q \times l$, de parâmetros vij , onde vij regula o impacto do j -ésimo meta-recurso no relativo importância do modelo de conjunto i -ésimo. Substituindo o valor de w_{ut} da Equação 6.12 em Equação 6.11, obtemos a relação entre a classificação do conjunto e o componente classificações da seguinte forma:

$$\text{rota} = \sum_{i=1}^q \sum_{j=1}^l vij zut_{eu} r_{\text{mas}}^i \quad (6.13)$$

Observe que este ainda é um problema de regressão linear em coeficientes $q \times l$ correspondentes ao variáveis vij . Um modelo de regressão de mínimos quadrados padrão pode ser usado para aprender os valores de vij nas classificações mantidas⁵. As variáveis independentes desta regressão são dadas por quantidades zut_{eu} . A regularização pode ser usada para reduzir o sobreajuste. Após os pesos terem aprendido usando regressão linear, os modelos de componentes individuais são retreinados no conjunto de treinamento completo sem nenhuma entrada retida. Os pesos, que foram aprendidos usando o entradas estendidas são usadas em conjunto com esses modelos q .

6.9 Híbridos Mistas

A principal característica dos sistemas de recomendação mistos é que eles combinam as pontuações de diferentes componentes em termos de apresentação, em vez de em termos de combinação as pontuações previstas. Em muitos casos, os itens recomendados são apresentados um ao lado do outro [121, 623]. Portanto, a principal característica distintiva de tais sistemas é a combinação de apresentação em vez da combinação de pontuações previstas.

⁵No contexto do concurso do Prémio Netflix, isto foi conseguido numa parte especial do conjunto de dados, referida como o conjunto de sondas. O conjunto de sondas não foi utilizado para construir os modelos de conjuntos de componentes.

A maioria dos outros sistemas híbridos concentra-se na criação de uma classificação unificada, extraída dos vários sistemas. Um exemplo clássico é ilustrado em [559], no qual uma listagem de televisão personalizada é criada usando um sistema misto. Normalmente, um programa composto é apresentado ao usuário. Esse programa composto é criado pela combinação de itens recomendados por diferentes sistemas. Esses programas compostos são típicos no uso de sistemas mistos, embora a aplicabilidade de sistemas mistos vá além desses cenários. Em muitos desses casos, a ideia básica é que a recomendação é projetada para um item relativamente complexo contendo muitos componentes, e não faz sentido recomendar os itens individualmente. O problema de inicialização de um novo item é frequentemente amenizado com um sistema de recomendação misto. Como um programa de televisão tem muitos espaços, o recomendador baseado em conteúdo ou colaborativo pode ser bem-sucedido em preencher os diferentes espaços. Em alguns casos, um número suficiente de recomendações para os espaços pode ser alcançado apenas com múltiplos recomendadores de diferentes tipos, especialmente no início, quando há escassez de dados disponíveis. Entretanto, a resolução de conflitos pode ser necessária em alguns casos em que há mais opções disponíveis do que vagas disponíveis.

Outro exemplo de um híbrido misto foi proposto no domínio do turismo [660, 661]. Nesse caso, são criados pacotes de recomendações, onde cada pacote contém diversas categorias de itens. Por exemplo, em um sistema de recomendação de turismo, as diferentes categorias podem corresponder a acomodações, atividades de lazer, passagens aéreas e assim por diante. Os turistas normalmente compram pacotes desses itens de diversas categorias para criar suas viagens. Para cada categoria, um sistema de recomendação diferente é empregado. A ideia básica aqui é que o sistema de recomendação mais apropriado para obter as melhores acomodações pode não ser o mais apropriado para recomendar atividades turísticas. Portanto, cada um desses diferentes aspectos é tratado como uma categoria diferente para a qual um sistema de recomendação diferente é empregado. Além disso, é importante recomendar pacotes nos quais os itens de várias categorias não sejam mutuamente inconsistentes. Por exemplo, se um turista recebe uma recomendação de uma atividade de lazer muito distante de seu local de acomodação, o pacote geral de recomendação não será muito conveniente para o turista. Portanto, uma base de conhecimento contendo um conjunto de restrições de domínio é incorporada ao processo de agrupamento. As restrições são projetadas para resolver inconsistências no domínio do produto. Um problema de satisfação de restrições é empregado para determinar um agrupamento mutuamente consistente. Mais detalhes da abordagem são discutidos em [660, 661].

Vale ressaltar que muitos dos híbridos mistos são frequentemente utilizados em conjunto com sistemas de recomendação baseados em conhecimento como um dos componentes [121, 660]. Isso não é coincidência. Híbridos mistos são geralmente projetados para domínios de produtos complexos com múltiplos componentes, como sistemas de recomendação baseados em conhecimento.

6.10 Resumo

Sistemas de recomendação híbridos são utilizados para alavancar o poder de múltiplas fontes de dados ou para melhorar o desempenho de sistemas de recomendação existentes dentro de uma modalidade de dados específica. Uma motivação importante para a construção de sistemas de recomendação híbridos é que diferentes tipos de sistemas de recomendação, como métodos colaborativos, baseados em conteúdo e baseados em conhecimento, apresentam diferentes pontos fortes e fracos. Alguns sistemas de recomendação funcionam mais eficazmente na inicialização a frio, enquanto outros funcionam mais eficazmente quando há dados suficientes disponíveis. Os sistemas de recomendação híbridos buscam alavancar os pontos fortes complementares desses sistemas para criar um sistema com maior robustez geral.

Métodos de conjunto também são utilizados para melhorar a precisão de métodos de filtragem colaborativa, nos quais os diferentes componentes utilizam a mesma matriz de classificação. Nesses casos, os modelos individuais utilizam os mesmos dados base, em vez de diferentes fontes de dados. Esses métodos estão muito mais próximos das ideias existentes sobre análise de conjunto no domínio da classificação.

A ideia básica é usar os vários modelos para incorporar a diversidade e reduzir o viés do modelo.

Muitos dos resultados teóricos existentes sobre o trade-off entre viés e variância na classificação também são aplicáveis a aplicações de filtragem colaborativa. Portanto, muitas técnicas, como bagging e boosting, podem ser adaptadas com modificações relativamente pequenas.

Os sistemas híbridos são projetados como sistemas monolíticos, sistemas de conjunto ou sistemas mistos. Os sistemas ensemble são normalmente projetados utilizando arranjos sequenciais ou paralelos de recomendadores. No design monolítico, os recomendadores existentes são modificados ou são criados recomendadores inteiramente novos, combinando recursos de diversas modalidades de dados. Em sistemas mistos, recomendações de vários mecanismos são apresentadas simultaneamente. Em muitos casos, meta-recursos também podem ser extraídos de uma modalidade de dados específica para combinar as previsões de vários recomendadores de forma específica para cada entrada. A grande vantagem dos sistemas híbridos e de conjunto decorre de sua capacidade de alavancar pontos fortes complementares em vários sistemas. As principais inscrições no concurso do Prêmio Netflix foram todas de sistemas de conjunto.

6.11 Notas Bibliográficas

Embora os sistemas híbridos tenham uma longa e rica história no desenvolvimento de sistemas de recomendação, uma categorização formal desses métodos não foi realizada até a pesquisa de Burke [117]. Uma discussão sobre sistemas de recomendação híbridos no contexto específico da Web é fornecida em [118]. Burke originalmente categorizou os sistemas de recomendação em sete categorias diferentes. Posteriormente, Jannach et al. [275] criaram uma categorização de nível superior dessas categorias de nível inferior em sistemas pipelined e paralelos. A taxonomia hierárquica neste livro segue aproximadamente o trabalho de [275] e [117], embora faça uma série de modificações para incluir vários métodos importantes, como boosting, em uma dessas categorias. É importante notar que esta taxonomia não é exaustiva porque muitos sistemas de conjunto, como aqueles vencedores do Prêmio Netflix, usam ideias de muitos tipos de híbridos. No entanto, a categorização original de Burke é muito instrutiva, porque abrange a maioria dos blocos de construção importantes. Recentemente, os métodos de conjunto têm recebido muita atenção, especialmente depois que as inscrições vencedoras no concurso Netflix Prize foram ambos sistemas de conjunto [311, 704].

Métodos de conjunto têm sido amplamente utilizados na literatura de classificação. Uma discussão detalhada sobre o trade-off viés-variância no contexto do problema de classificação é apresentada em [22]. Métodos de bagging e subamostragem para classificação são discutidos em [111–113].

Um trabalho recente [67] mostra como se pode alavancar métodos de conjunto da literatura de classificação para sistemas de recomendação, adaptando métodos como bagging e AdaBoost.RT.

Enquanto alguns sistemas de conjuntos são desenvolvidos com essa motivação, outros sistemas combinam o poder de diferentes tipos de dados. Modelos ponderados estão entre as classes mais populares de modelos. Alguns dos modelos combinam modelos construídos em tipos de dados homogêneos. Métodos para construir conjuntos ponderados homogêneos são discutidos em [67, 266]. Os vencedores [311, 704] do concurso Netflix Prize também usaram um sistema de conjuntos ponderados, embora a combinação use meta-características adicionais, o que o imbui com algumas das propriedades de uma abordagem de combinação de características. O trabalho em [180] usa conjuntos de métodos de fatoração de matriz de margem máxima com diferentes configurações de parâmetros. Baseado no usuário e baseado em item

algoritmos de vizinhança são combinados em [338]. Outros trabalhos recentes sobre modelos ponderados mostram como combinar sistemas construídos sobre diferentes tipos de dados. O trabalho em [659] combina um recomendador colaborativo e baseado em conhecimento, enquanto o trabalho em [162] combina um recomendador colaborativo e baseado em conteúdo.

Um híbrido de comutação baseado em desempenho é discutido em [601]. Uma abordagem interessante de aprendizado de máquina para mecanismos de comutação é discutida em [610]. Outros mecanismos de comutação para lidar com problemas de inicialização a frio são discutidos em [85]. Outra combinação de um sistema baseado em conhecimento e colaborativo para criar um híbrido de comutação é discutida em [659].

Os sistemas em cascata usam o processamento sequencial das classificações para fazer recomendações. Tais sistemas podem utilizar refinamentos ou métodos de reforço. O recomendador EntreeC [117] é o exemplo mais conhecido de um sistema em cascata que utiliza refinamentos. Um sistema em cascata que utiliza reforço é discutido em [67]. Este último método utiliza uma versão ponderada do algoritmo AdaBoost.RT para criar o recomendador híbrido.

Híbridos de aumento de características usam os recomendadores de um tipo para aumentar as características de outro. O sistema Libra [448] combina o sistema de recomendação da Amazon.com com seu próprio classificador Bayesiano. A saída do sistema da Amazon é usada para criar um recomendador baseado em conteúdo. O método em [431] usa um sistema baseado em conteúdo para estimar as entradas ausentes da matriz de classificações e usa os valores estimados no contexto de um sistema colaborativo. No sistema GroupLens [526], um sistema baseado em conhecimento foi usado para criar um banco de dados de classificações artificiais. Essas classificações foram usadas no contexto de um sistema colaborativo para fazer recomendações. O trabalho em [600] mostra como usar um híbrido de aumento de características para recomendar artigos de pesquisa.

Muitas técnicas têm sido usadas recentemente para criar espaços de características fundidas ou representações unificadas a partir de matrizes de classificação e matrizes de conteúdo. Essa representação unificada ou espaço de características forma a base sobre a qual ferramentas de aprendizado de máquina podem ser aplicadas. Um dos primeiros trabalhos nessa linha constrói mapas de características conjuntas [68] a partir de informações de classificação e conteúdo e, em seguida, usa modelos de aprendizado de máquina para realizar a previsão. Uma abordagem baseada em tensores é usada para atingir esse objetivo. Uma abordagem análoga também é usada em [557], que fatora conjuntamente a matriz de perfil de compra do usuário-item e a matriz de conteúdo do item-característica em um espaço latente comum. Essa representação latente é então usada para aprendizado. O trabalho em [411] usa um modelo de fator latente no qual o texto da avaliação é combinado com as classificações. Um modelo de fator latente baseado em regressão é proposto em [14] para previsão de classificação, que usa características de conteúdo para estimativa de fatores. Os fatores latentes do usuário e do item são estimados por meio de regressão independente nas características do usuário e do item. Em seguida, uma função multiplicativa é usada nos fatores do usuário e do item para previsão. Modelos de regressão esparsos também foram utilizados para predição fundida em [456]. Por fim, modelos baseados em grafos foram utilizados para criar representações unificadas. O trabalho em [238] analisa os pesos de interação entre ações do usuário e vários recursos, como informações de perfil do usuário-item e informações secundárias. Máquinas de Boltzmann unificadas são utilizadas para realizar a predição. Uma representação unificada baseada em grafos foi proposta em [129]. Uma rede bayesiana é criada contendo nós de itens, nós de usuários e nós de recursos de itens. Essa rede bayesiana é utilizada para realizar recomendações combinadas baseadas em conteúdo e colaborativas.

Em um híbrido de metanível, o modelo aprendido por um recomendador é usado como entrada para o próximo nível. No trabalho inicial de Pazzani [475], um modelo baseado em conteúdo [363] é construído para descrever as características discriminativas que predizem restaurantes. Cada usuário é definido por uma representação vetorial de palavras discriminativas. O modelo baseado em conteúdo é usado para determinar o grupo de pares, que é então usado para fins de recomendação. Combinações de metanível de sistemas baseados em conteúdo e colaborativos são discutidas em [475, 534]. Um híbrido de metanível bayesiano de dois estágios é discutido em [166]. Um tipo diferente de modelo hierárquico de Bayes que

combina sistemas colaborativos e baseados em conteúdo é apresentado em [652]. Métodos para empilhar sistemas de recomendação com meta-recursos são discutidos em [65, 66, 311, 554]. O sistema STREAM [65, 66] foi um dos primeiros sistemas a alavancar recursos de nível meta.

Vários sistemas de recomendação mistos foram propostos em [121, 559, 623, 660, 661]. Um sistema de recomendação misto para a criação de programas de televisão é discutido em [559], enquanto um sistema para o fornecimento de pacotes turísticos é discutido em [660]. Vale ressaltar que muitos sistemas de recomendação mistos são utilizados em domínios de produtos complexos, como sistemas de recomendação baseados em conhecimento [121, 660].

6.12 Exercícios

1. Como a classificação do modelo de fator latente afeta o trade-off entre viés e variância em um sistema de recomendação? Se você tivesse que usar um modelo de fator latente como modelo base para um conjunto de bagging, escolheria um modelo com classificação alta ou baixa?
2. A sua resposta ao Exercício 1 muda se você tiver que usar o reforço em conjunto com um modelo de fator latente?
3. Implementar um modelo de ensacamento de entrada usando um modelo de fator latente ponderado como o modelo base.
4. Suponha que você criou um sistema colaborativo no qual a matriz usuário-item continha frequências de palavras como linhas adicionais da matriz. Cada linha adicional é uma palavra, e o valor da combinação palavra-item é uma frequência. Um modelo de vizinhança baseado em itens é usado com essa representação aumentada. Que tipo de híbrido isso seria considerado? Discuta o possível impacto do uso de tal modelo na precisão e diversidade do sistema de recomendação.
5. Discuta como você controlaria a força relativa das partes colaborativas e baseadas em conteúdo no Exercício 4 com um único parâmetro de peso. Como você determinaria o valor ideal do parâmetro de peso de forma orientada por dados?

Capítulo 7

Avaliando Sistemas de Recomendação

“O verdadeiro gênio reside na capacidade de avaliação de informações incertas, perigosas e conflitantes.” – Winston Churchill

7.1 Introdução

A avaliação da filtragem colaborativa compartilha uma série de semelhanças com a da classificação. Essa semelhança se deve ao fato de que a filtragem colaborativa pode ser vista como uma generalização do problema de modelagem de classificação e regressão (cf. seção 1.3.1.3 do Capítulo 1). No entanto, há muitos aspectos do processo de avaliação que são exclusivos das aplicações de filtragem colaborativa. A avaliação de métodos baseados em conteúdo é ainda mais semelhante à da modelagem de classificação e regressão, pois os métodos baseados em conteúdo frequentemente utilizam métodos de classificação de texto em segundo plano. Este capítulo apresentará vários mecanismos para avaliar diversos algoritmos de recomendação e também relacionará essas técnicas aos métodos análogos usados na modelagem de classificação.

Um projeto adequado do sistema de avaliação é crucial para compreender a eficácia de diversos algoritmos de recomendação. Como veremos mais adiante neste capítulo, a avaliação de sistemas de recomendação é frequentemente multifacetada, e um único critério não consegue captar muitos dos objetivos do projetista. Um projeto incorreto da avaliação experimental pode levar a uma subestimação ou superestimação grosseira da real precisão de um determinado algoritmo ou modelo.

Os sistemas de recomendação podem ser avaliados por meio de métodos online ou offline. Em um sistema online, as reações do usuário são medidas em relação às recomendações apresentadas. Portanto, a participação do usuário é essencial em sistemas online. Por exemplo, em uma avaliação online de um sistema de recomendação de notícias, pode-se medir a taxa de conversão de usuários que clicam em artigos recomendados. Esses métodos de teste são chamados de testes A/B e medem o impacto direto do sistema de recomendação.

no usuário final. Em última análise, aumentar a taxa de conversão em itens rentáveis é o objetivo mais importante de um sistema de recomendação e pode fornecer uma medida real da eficácia do sistema. No entanto, como as avaliações online exigem a participação ativa do usuário, muitas vezes não é viável utilizá-las em benchmarking e pesquisas. Geralmente, há desafios significativos para obter acesso a dados de conversão de usuários de sistemas com participação de usuários em larga escala. Mesmo que tal acesso seja obtido, geralmente é específico para um único sistema de larga escala. Por outro lado, frequentemente se deseja usar conjuntos de dados de diferentes tipos e de múltiplos domínios. Testar em múltiplos conjuntos de dados é particularmente importante para garantir maior poder de generalização do sistema de recomendação, de modo que se possa ter certeza de que o algoritmo funciona em uma variedade de cenários. Nesses casos, são utilizadas avaliações offline com conjuntos de dados históricos. Os métodos offline são, de longe, os métodos mais comuns para avaliar sistemas de recomendação de uma perspectiva de pesquisa e prática. Portanto, a maior parte deste capítulo se concentrará em métodos offline, embora alguma discussão sobre métodos online também seja incluída para fins de completude.

Ao trabalhar com métodos offline, as medidas de precisão podem frequentemente fornecer uma imagem incompleta da verdadeira taxa de conversão de um sistema de recomendação. Diversas outras medidas secundárias também desempenham um papel. Portanto, é importante projetar o sistema de avaliação cuidadosamente para que as métricas medidas reflitam verdadeiramente a eficácia do sistema da perspectiva do usuário. Em particular, as seguintes questões são importantes da perspectiva do projeto de métodos de avaliação para sistemas de recomendação:

1. Objetivos da avaliação: Embora seja tentador usar métricas de precisão para avaliar sistemas de recomendação, essa abordagem pode frequentemente fornecer uma imagem incompleta da experiência do usuário. Embora as métricas de precisão sejam indiscutivelmente os componentes mais importantes da avaliação, muitos objetivos secundários, como novidade, confiança, cobertura e serendipidade, são importantes para a experiência do usuário. Isso ocorre porque essas métricas têm impactos importantes de curto e longo prazo nas taxas de conversão. No entanto, a quantificação real de alguns desses fatores costuma ser bastante subjetiva, e muitas vezes não há medidas concretas para fornecer uma métrica numérica.
2. Problemas de planejamento experimental: Mesmo quando a precisão é usada como métrica, é crucial planejar os experimentos de forma que ela não seja superestimada ou subestimada. Por exemplo, se o mesmo conjunto de classificações especificadas for usado tanto para a construção do modelo quanto para a avaliação da precisão, a precisão será grosseiramente superestimada. Nesse contexto, um planejamento experimental cuidadoso é importante.
3. Métricas de precisão: Apesar da importância de outras medidas secundárias, as métricas de precisão continuam sendo o componente mais importante na avaliação. Os sistemas de recomendação podem ser avaliados em termos da precisão da previsão de uma classificação ou da precisão da classificação dos itens. Portanto, diversas métricas comuns, como o erro absoluto médio e o erro quadrático médio, são utilizadas com frequência. A avaliação de classificações pode ser realizada com o uso de vários métodos, como cálculos baseados em utilidade, coeficientes de correlação de classificação e a curva característica de operação do receptor.

Neste capítulo, começaremos discutindo os objetivos gerais da avaliação de sistemas de recomendação, além do critério mais básico de precisão. Exemplos de tais objetivos incluem diversidade e novidade. O principal desafio na quantificação desses objetivos é que eles frequentemente são objetivos subjetivos baseados na experiência do usuário. De uma perspectiva de quantificação, a precisão é um objetivo concreto, relativamente fácil de mensurar e, portanto, usado com mais frequência para benchmarking e testes. Existem alguns métodos de quantificação para avaliar os objetivos secundários, como

diversidade e novidade. Embora a maior parte deste capítulo se concentre em métricas de precisão, diversas medidas de quantificação para os objetivos secundários também serão discutidas.

Este capítulo está organizado da seguinte forma. Uma visão geral dos diferentes tipos de sistemas de avaliação é apresentada na seção 7.2. A seção 7.3 estuda os objetivos gerais da avaliação de sistemas de recomendação. O projeto apropriado de métodos de teste de acurácia é discutido na seção 7.4. Métricas de acurácia para sistemas de recomendação são discutidas na seção 7.5. As limitações das medidas de avaliação são discutidas na seção 7.6. Um resumo é apresentado na seção 7.7.

7.2 Paradigmas de Avaliação

Existem três tipos principais de avaliação de sistemas de recomendação: estudos com usuários, avaliações online e avaliações offline com conjuntos de dados históricos. Os dois primeiros tipos envolvem usuários, embora sejam conduzidos de maneiras ligeiramente diferentes. As principais diferenças entre os dois primeiros cenários residem na forma como os usuários são recrutados para os estudos. Embora as avaliações online forneçam insights úteis sobre os verdadeiros efeitos de um algoritmo de recomendação, muitas vezes existem obstáculos práticos significativos à sua implementação. A seguir, apresentamos uma visão geral desses diferentes tipos de avaliação.

7.2.1 Estudos de Usuário

Em estudos com usuários, os participantes do teste são recrutados ativamente e solicitados a interagir com o sistema de recomendação para executar tarefas específicas. O feedback pode ser coletado do usuário antes e depois da interação, e o sistema também coleta informações sobre sua interação com o sistema de recomendação. Esses dados são então usados para fazer inferências sobre os gostos ou desgostos do usuário. Por exemplo, os usuários podem ser solicitados a interagir com as recomendações em um site de produto e fornecer seu feedback sobre a qualidade das recomendações. Essa abordagem pode então ser usada para avaliar a eficácia dos algoritmos subjacentes. Alternativamente, os usuários podem ser solicitados a ouvir várias músicas e, em seguida, fornecer seu feedback sobre essas músicas na forma de classificações.

Uma vantagem importante dos estudos de usuários é que eles permitem a coleta de informações sobre a interação do usuário com o sistema. Diversos cenários podem ser testados sobre o efeito da alteração do sistema de recomendação na interação do usuário, como o efeito da alteração de um algoritmo ou interface de usuário específicos. Por outro lado, a conscientização ativa do usuário sobre o teste do sistema de recomendação pode frequentemente influenciar suas escolhas e ações.

Também é difícil e caro recrutar grandes grupos de usuários para fins de avaliação.

Em muitos casos, os usuários recrutados não são representativos da população em geral, pois o próprio processo de recrutamento é um filtro centrado em vieses, que não pode ser totalmente controlado. Nem todos os usuários estariam dispostos a participar de tal estudo, e aqueles que concordam podem ter interesses não representativos em relação à população restante. Por exemplo, no caso da classificação de músicas, os participantes (voluntários) provavelmente são entusiastas de música. Além disso, o fato de os usuários estarem ativamente cientes de seu recrutamento para um estudo específico provavelmente afetará suas respostas. Portanto, os resultados das avaliações dos usuários não são totalmente confiáveis.

7.2.2 Avaliação Online

As avaliações online também alavancam estudos de usuários, exceto que os usuários geralmente são usuários reais em um sistema totalmente implantado ou comercial. Essa abordagem às vezes é menos suscetível a vieses no processo de recrutamento, pois os usuários costumam usar o sistema diretamente no curso natural das coisas. Esses sistemas podem frequentemente ser usados para avaliar a comparação

desempenho de vários algoritmos [305]. Normalmente, os usuários podem ser amostrados aleatoriamente, e os vários algoritmos podem ser testados com cada amostra de usuários. Um exemplo típico de uma métrica, que é usada para medir a eficácia do sistema de recomendação sobre os usuários, é a taxa de conversão. A taxa de conversão mede a frequência com que um usuário seleciona um item recomendado. Por exemplo, em um sistema de recomendação de notícias, pode-se calcular a fração de vezes que um usuário seleciona um artigo recomendado. Se desejado, os custos ou lucros esperados podem ser adicionados aos itens para tornar a medição sensível à importância do item. Esses métodos também são chamados de testes A/B e medem o impacto direto do sistema de recomendação sobre o usuário final. A ideia básica nesses métodos é comparar dois algoritmos da seguinte forma:

1. Segmente os usuários em dois grupos A e B.
2. Use um algoritmo para o grupo A e outro algoritmo para o grupo B por um período de tempo, mantendo todas as outras condições (por exemplo, processo de seleção de usuários) nos dois grupos o mais semelhantes possível.
3. No final do processo, compare a taxa de conversão (ou outra métrica de retorno) do dois grupos.

Essa abordagem é muito semelhante à utilizada em ensaios clínicos na medicina. É a mais precisa para testar o desempenho de longo prazo do sistema diretamente em termos de objetivos como lucro. Esses métodos também podem ser utilizados nos estudos com usuários discutidos na seção anterior.

Uma observação é que não é necessário segmentar estritamente os usuários em grupos nos casos em que o retorno de cada interação entre o usuário e o recomendador pode ser medido separadamente. Nesses casos, um dos algoritmos pode ser mostrado ao mesmo usuário aleatoriamente, e o retorno dessa interação específica pode ser medido. Esses métodos de avaliação de sistemas de recomendação também foram generalizados para o desenvolvimento de algoritmos de recomendação mais eficazes. Os algoritmos resultantes são chamados de algoritmos multi-arm bandit. A ideia básica é semelhante à de um jogador (sistema de recomendação) que se depara com a escolha de selecionar uma de um conjunto de máquinas caça-níqueis (algoritmos de recomendação) no cassino. O jogador suspeita que uma dessas máquinas tem um retorno (taxa de conversão) melhor do que as outras. Portanto, o jogador tenta uma máquina caça-níqueis aleatoriamente 10% das vezes para explorar os retornos relativos das máquinas. O jogador seleciona avidamente a máquina caça-níqueis com melhor pagamento nos 90% restantes do tempo, a fim de explorar o conhecimento adquirido nos testes exploratórios. O processo de exploração e aproveitamento é totalmente intercalado de forma aleatória. Além disso, o jogador pode optar por dar maior peso aos resultados recentes em comparação aos resultados mais antigos para avaliação. Essa abordagem geral está relacionada à noção de aprendizado por reforço, que frequentemente pode ser combinada com sistemas online. Embora o aprendizado por reforço tenha sido extensivamente estudado na literatura de classificação e modelagem de regressão [579], o trabalho correspondente no domínio da recomendação é bastante limitado [389, 390, 585]. Existe uma oportunidade significativa de pesquisa para o desenvolvimento adicional de tais algoritmos.

A principal desvantagem é que tais sistemas não podem ser implementados de forma realista, a menos que um grande número de usuários já esteja registrado. Portanto, é difícil usar esse método durante a fase de inicialização. Além disso, tais sistemas geralmente não são de acesso aberto e são acessíveis apenas ao proprietário do sistema comercial específico em questão. Portanto, tais testes podem ser realizados apenas pela entidade comercial e para o número limitado de cenários gerenciados por seu sistema. Isso significa que os testes muitas vezes não são generalizáveis.

para benchmarking independente de sistema por cientistas e profissionais. Em muitos casos, é desejável testar a robustez de um algoritmo de recomendação por meio de testes de estresse em uma variedade de configurações e domínios de dados. Utilizando múltiplas configurações, pode-se obter uma ideia da generalização do sistema. Infelizmente, os métodos online não são projetados para atender a essas necessidades. Parte do problema é que não se pode controlar totalmente as ações dos usuários de teste no processo de avaliação.

7.2.3 Avaliação offline com conjuntos de dados históricos

Em testes offline, dados históricos, como classificações, são usados. Em alguns casos, informações temporais também podem ser associadas às classificações, como o registro de data e hora em que cada usuário classificou o item. Um exemplo bem conhecido de um conjunto de dados históricos é o conjunto de dados do Prêmio Netflix [311]. Este conjunto de dados foi originalmente lançado no contexto de um concurso online e, desde então, tem sido usado como um benchmark padronizado para testar muitos algoritmos. A principal vantagem do uso de conjuntos de dados históricos é que eles não exigem acesso a uma grande base de usuários. Uma vez que um conjunto de dados tenha sido coletado, ele pode ser usado como um benchmark padronizado para comparar vários algoritmos em uma variedade de configurações. Além disso, vários conjuntos de dados de vários domínios (por exemplo, música, filmes, notícias) podem ser usados para testar a generalização do sistema de recomendação.

Métodos offline estão entre as técnicas mais populares para testar algoritmos de recomendação, pois estruturas e medidas de avaliação padronizadas foram desenvolvidas para esses casos. Portanto, grande parte deste capítulo será dedicada ao estudo da avaliação offline. A principal desvantagem das avaliações offline é que elas não medem a propensão real do usuário a reagir ao sistema de recomendação no futuro. Por exemplo, os dados podem evoluir ao longo do tempo e as previsões atuais podem não refletir as previsões mais adequadas para o futuro. Além disso, medidas como precisão não capturam características importantes das recomendações, como serendipidade e novidade.

Tais recomendações têm efeitos importantes a longo prazo na taxa de conversão das recomendações. No entanto, apesar dessas desvantagens, os métodos offline continuam sendo as técnicas mais amplamente aceitas para avaliação de sistemas de recomendação. Isso se deve às quantificações estatisticamente robustas e de fácil compreensão disponíveis por meio desses métodos de teste.

7.3 Objetivos Gerais do Design de Avaliação

Nesta seção, estudaremos alguns dos objetivos gerais na avaliação de sistemas de recomendação. Além do conhecido objetivo de precisão, outros objetivos gerais incluem fatores como diversidade, serendipidade, novidade, robustez e escalabilidade. Alguns desses objetivos podem ser quantificados concretamente, enquanto outros são objetivos subjetivos baseados na experiência do usuário. Nesses casos, a única maneira de mensurar tais objetivos é por meio de pesquisas com usuários. Nesta seção, estudaremos esses diferentes objetivos.

7.3.1 Precisão

A precisão é uma das medidas mais fundamentais pelas quais os sistemas de recomendação são avaliados. Nesta seção, fornecemos uma breve introdução a essa medida. Uma discussão detalhada é fornecida na seção 7.5 deste capítulo. No caso mais geral, as classificações são grandezas numéricas que precisam ser estimadas. Portanto, as métricas de precisão são frequentemente

semelhantes aos usados na modelagem de regressão. Seja R a matriz de classificações na qual r_{uj} é a classificação conhecida do usuário u para o item j . Considere o caso em que um algoritmo de recomendação estima essa classificação como \hat{r}_{uj} . Então, o erro específico de entrada da estimativa é dado pela quantidade $e_{uj} = \hat{r}_{uj} - r_{uj}$. O erro geral é calculado pela média dos erros específicos de entrada em termos de valores absolutos ou em termos de valores quadrados. Além disso, muitos sistemas não preveem classificações; em vez disso, eles apenas produzem classificações dos principais itens recomendados. Isso é particularmente comum em conjuntos de dados de feedback implícito. Diferentes métodos são usados para avaliar a precisão das previsões de classificações e a precisão das classificações.

Como os vários métodos para calcular a precisão são discutidos em detalhes na seção 7.5, eles não serão discutidos aqui. O objetivo desta breve seção é apresentar brevemente algumas medidas para garantir a continuidade em discussões futuras. Os principais componentes da avaliação da precisão são os seguintes:

1. Projetando a avaliação da precisão: Todas as entradas observadas de uma matriz de classificações não podem ser usadas tanto para treinar o modelo quanto para avaliar a precisão. Fazer isso superestimaria grosseiramente a precisão devido ao sobreajuste. É importante usar apenas um conjunto diferente de entradas para avaliação do que foi usado para treinamento. Se S são as entradas observadas na matriz de classificações, então um pequeno subconjunto $E \subseteq S$ é usado para avaliação, e o conjunto $S \setminus E$ é usado para treinamento. Este problema é idêntico ao encontrado na avaliação de algoritmos de classificação. Afinal, como discutido em capítulos anteriores, a filtragem colaborativa é uma generalização direta do problema de classificação e modelagem de regressão. Portanto, os métodos padrão que são usados em classificação e modelagem de regressão, como hold-out e validação cruzada, também são usados na avaliação de algoritmos de recomendação. Essas questões serão discutidas em mais detalhes na seção 7.4.

2. Métricas de precisão: Métricas de precisão são usadas para avaliar a precisão da previsão de estimativas de classificações de combinações específicas de usuário-item ou a precisão da classificação top-k prevista por um sistema de recomendação. Normalmente, as classificações de um conjunto E de entradas na matriz de classificações são ocultadas, e a precisão é avaliada com base nessas entradas ocultas. Diferentes classes de métodos são usadas para os dois casos:

- Precisão na estimativa de classificações: Conforme discutido acima, o erro específico da entrada é dado por $e_{uj} = \hat{r}_{uj} - r_{uj}$ para o usuário u e o item j . Esse erro pode ser aproveitado de várias maneiras para calcular o erro geral sobre o conjunto E de entradas na matriz de classificações na qual a avaliação é realizada. Um exemplo é o erro quadrático médio, que é denotado por MSE:

$$MSE = \frac{\sum_{(u,j) \in E} e_{uj}^2}{|E|} \quad (7.1)$$

A raiz quadrada da quantidade mencionada acima é chamada de erro quadrático médio, ou RMSE.

$$RMSE = \sqrt{\frac{\sum_{(u,j) \in E} e_{uj}^2}{|E|}} \quad (7.2)$$

A maioria dessas medidas são emprestadas da literatura sobre modelagem de regressão. Outras formas importantes de medir o erro, como o erro absoluto médio, são discutidas na seção 7.5.

- Precisão na estimativa de classificações: Muitos sistemas de recomendação não estimam classificações diretamente; em vez disso, fornecem estimativas das classificações subjacentes. Dependendo da natureza da verdade básica, podem-se utilizar medidas de correlação de classificação, medidas baseadas em utilidade ou a característica operacional do receptor. Os dois últimos métodos são projetados para conjuntos de dados unários (feedback implícito). Esses métodos são discutidos em detalhes na seção 7.5.

Algumas medidas de precisão também são projetadas para maximizar o lucro do comerciante, porque todos os itens não são igualmente importantes da perspectiva do processo de recomendação.

Essas métricas incorporam custos específicos de cada item ao cálculo. O principal problema com métricas de precisão é que elas frequentemente não medem a verdadeira eficácia de um sistema de recomendação em cenários reais. Por exemplo, uma recomendação óbvia pode ser precisa, mas um usuário pode ter comprado aquele item de qualquer maneira. Portanto, tal recomendação pode ter pouca utilidade em termos de melhoria da taxa de conversão do sistema. Uma discussão sobre os desafios associados ao uso de métricas de precisão pode ser encontrada em [418].

7.3.2 Cobertura

Mesmo quando um sistema de recomendação é altamente preciso, ele pode frequentemente não ser capaz de recomendar uma certa proporção dos itens, ou pode nunca ser capaz de recomendar a uma certa proporção dos usuários. Essa medida é chamada de cobertura. Essa limitação dos sistemas de recomendação é um artefato do fato de que as matrizes de classificação são esparsas. Por exemplo, se uma matriz de classificação contém uma única entrada para cada linha e cada coluna, nenhuma recomendação significativa pode ser feita por praticamente nenhum algoritmo. No entanto, diferentes sistemas de recomendação têm diferentes níveis de propensão em fornecer cobertura. Em cenários práticos, os sistemas frequentemente têm 100% de cobertura devido ao uso de padrões para classificações que não são possíveis de prever. Um exemplo desse padrão seria relatar a média de todas as classificações de um usuário para um item quando a classificação para uma combinação específica usuário-item não pode ser prevista. Portanto, o trade-off entre precisão e cobertura sempre precisa ser incorporado ao processo de avaliação. Existem dois tipos de cobertura, que são chamados de cobertura do espaço do usuário e cobertura do espaço do item, respectivamente.

A cobertura do espaço do usuário mede a fração de usuários para a qual pelo menos k avaliações podem ser previstas. O valor de k deve ser definido como o tamanho esperado da lista de recomendações.

Quando menos de k avaliações podem ser previstas para um usuário, não é mais possível apresentar uma lista de recomendações significativa de tamanho k ao usuário. Tal situação pode ocorrer quando um usuário especifica pouquíssimas avaliações em comum com outros usuários. Considere um algoritmo de vizinhança baseado em usuário. É difícil calcular de forma robusta os pares desse usuário, devido às pouquíssimas avaliações mutuamente especificadas com outros usuários. Portanto, muitas vezes é difícil fazer recomendações suficientes para esse usuário. Para níveis muito altos de dispersão, é possível que nenhum algoritmo seja capaz de prever sequer uma avaliação para esse usuário. No entanto, algoritmos diferentes podem ter diferentes níveis de cobertura, e a cobertura de um usuário pode ser estimada executando cada algoritmo e determinando o número de itens para os quais uma previsão é feita. Um aspecto complicado da cobertura do espaço do usuário é que qualquer algoritmo pode fornecer cobertura completa simplesmente prevendo avaliações aleatórias para combinações usuário-item, cujas avaliações ele não pode prever de forma confiável. Portanto, a cobertura do espaço do usuário deve sempre ser avaliada em termos do trade-off entre precisão e cobertura. Por exemplo, em um recomendador baseado em vizinhança, aumentar o tamanho da vizinhança fornece uma curva que mostra a compensação entre cobertura e precisão.

Uma definição alternativa de cobertura do espaço do usuário é em termos da quantidade mínima de perfil que deve ser construída para um usuário antes que seja possível fazer recomendações a ele. Para um algoritmo específico, é possível estimar, por meio de experimentos, o número mínimo de avaliações observadas de qualquer usuário para as quais uma recomendação poderia ser feita. No entanto, muitas vezes é difícil avaliar essa quantidade porque a métrica é sensível à identidade dos itens para os quais o usuário especifica avaliações.

A noção de cobertura do espaço do item é análoga à de cobertura do espaço do usuário. A cobertura do espaço do item mede a fração de itens para os quais as avaliações de pelo menos k usuários podem ser previstas. Na prática, no entanto, essa noção raramente é utilizada, pois os sistemas de recomendação geralmente fornecem listas de recomendações para os usuários e raramente são utilizados para gerar usuários recomendados para itens.

Uma forma diferente de avaliação da cobertura do espaço de itens é definida pela noção de cobertura de catálogo, que é especificamente adequada para listas de recomendação. Observe que a definição acima mencionada foi adaptada para a previsão dos valores das classificações. Imagine um cenário em que cada entrada na matriz de classificações pode ser prevista por um algoritmo, mas o mesmo conjunto de itens top- k é sempre recomendado a todos os usuários. Portanto, embora a definição acima mencionada de cobertura do espaço de itens sugira um bom desempenho, a cobertura real entre todos os usuários é muito limitada. Em outras palavras, as recomendações não são diversas entre os usuários e o catálogo de itens não é totalmente coberto. Seja T_u a lista dos principais itens recomendados ao usuário $u \in \{1 \dots m\}$. A cobertura do catálogo CC é definida como a fração de itens que são recomendados a pelo menos um usuário.

$$CC = \frac{|\{y_{mu}=1 \text{ } Tu\}|}{n} \quad (7.3)$$

Aqui, a notação n representa o número de itens. É fácil estimar essa fração por meio de experimentos.

7.3.3 Confiança e Fé

A estimativa de classificações é um processo inexato que pode variar significativamente com os dados de treinamento específicos disponíveis. Além disso, a metodologia algorítmica também pode ter um impacto significativo nas classificações previstas. Isso sempre gera incerteza no usuário quanto à precisão das previsões. Muitos sistemas de recomendação podem relatar classificações juntamente com estimativas de confiança. Por exemplo, um intervalo de confiança sobre o intervalo de classificações previstas pode ser fornecido. Em geral, sistemas de recomendação que podem recomendar com precisão intervalos de confiança menores são mais desejáveis, pois reforçam a confiança do usuário no sistema.

Para dois algoritmos que usam o mesmo método para relatar a confiança, é possível mensurar o quanto bem o erro previsto corresponde a esses intervalos de confiança. Por exemplo, se dois sistemas de recomendação fornecem intervalos de confiança de 95% para cada classificação, pode-se medir a largura absoluta dos intervalos relatados pelos dois algoritmos. O algoritmo com a menor largura de intervalo de confiança vencerá, desde que ambos os algoritmos estejam corretos (ou seja, dentro dos intervalos especificados) pelo menos 95% das vezes nas classificações ocultas. Se um dos algoritmos ficar abaixo da precisão necessária de 95%, ele perde automaticamente. Infelizmente, se um sistema usa intervalos de confiança de 95% e outro usa intervalos de confiança de 99%, não é possível compará-los significativamente. Portanto, é possível usar tais sistemas apenas definindo o mesmo nível de confiança em ambos os casos.

Enquanto a confiança mede a fé do sistema na recomendação, a confiança mede a fé do usuário na avaliação. O conceito de confiança social é discutido em mais detalhes no Capítulo 11. Em termos gerais, a confiança mede o nível de fé que o usuário tem na recomendação.

classificações relatadas. Mesmo que as classificações previstas sejam precisas, muitas vezes não são úteis se o usuário não confiar nas classificações fornecidas. A confiança está intimamente relacionada, mas não é exatamente a mesma, à precisão. Por exemplo, quando explicações são fornecidas pelo sistema de recomendação, é mais provável que o usuário confie no sistema, especialmente se as explicações forem lógicas.

A confiança frequentemente não atende aos mesmos objetivos que a utilidade (utilidade) de uma recomendação. Por exemplo, se um sistema de recomendação sugere alguns itens já apreciados e conhecidos pelo usuário, pode-se argumentar que há pouca utilidade fornecida ao usuário por tal recomendação. Por outro lado, tais itens podem aumentar a confiança do usuário no sistema. Este objetivo está em contradição direta com outros objetivos, como a novidade, em que recomendações já conhecidas pelo usuário são indesejáveis. É comum que os vários objetivos em sistemas de recomendação sejam compensados entre si. A maneira mais simples de medir a confiança é conduzir pesquisas com usuários durante os experimentos, nas quais os usuários são explicitamente questionados sobre sua confiança nos resultados. Tais experimentos também são chamados de experimentos online. Numerosos métodos online para avaliação de confiança são discutidos em [171, 175, 248, 486].

Geralmente, é difícil medir a confiança por meio de experimentos offline.

7.3.4 Novidade

A novidade de um sistema de recomendação avalia a probabilidade de um sistema de recomendação fornecer recomendações ao usuário das quais ele não tem conhecimento ou que não tenha visto antes. Uma discussão sobre a noção de novidade é fornecida em [308]. Recomendações não vistas frequentemente aumentam a capacidade do usuário de descobrir insights importantes sobre seus gostos e desgostos que ele não conhecia anteriormente. Isso é mais importante do que descobrir itens que ele já conhecia, mas não classificou. Em muitos tipos de sistemas de recomendação, como métodos baseados em conteúdo, as recomendações tendem a ser um tanto óbvias devido à propensão do sistema a recomendar itens esperados. Embora um pequeno número dessas recomendações possa aumentar a confiança do usuário final no sistema subjacente, elas nem sempre são úteis em termos de melhoria das taxas de conversão. A maneira mais natural de medir a novidade é por meio de experimentação online, na qual os usuários são explicitamente questionados se já conheciam um item anteriormente.

Conforme discutido na introdução, a experimentação online nem sempre é viável devido à falta de acesso a um sistema que suporte uma grande base de usuários online. Felizmente, é possível estimar aproximadamente a novidade com métodos offline, desde que os registros de tempo estejam disponíveis com as avaliações. A ideia básica é que sistemas inovadores são melhores em recomendar itens com maior probabilidade de serem selecionados pelo usuário no futuro, em vez de no momento presente. Portanto, todas as avaliações criadas após um determinado ponto no tempo t_0 são removidas dos dados de treinamento. Além disso, algumas das avaliações anteriores a t_0 também são removidas. O sistema é então treinado com essas avaliações removidas. Esses itens removidos são então usados para fins de pontuação. Para cada item avaliado antes do tempo t_0 e recomendado corretamente, a pontuação de avaliação de novidade é penalizada. Por outro lado, para cada item avaliado após o tempo t_0 e recomendado corretamente, a pontuação de avaliação de novidade é recompensada.

Portanto, esta avaliação mede um tipo de precisão diferencial entre previsões futuras e passadas. Em algumas medidas de novidade, presume-se que itens populares têm menor probabilidade de serem novos, e menos crédito é dado à recomendação de itens populares.

7.3.5 Serendipidade

A palavra “serendipidade” significa literalmente “descoberta afortunada”. Portanto, serendipidade é uma medida do nível de surpresa em recomendações bem-sucedidas. Em outras palavras, recomendações

precisa ser inesperado. Em contraste, a novidade requer apenas que o usuário não estivesse ciente da recomendação anteriormente. A serendipidade é uma condição mais forte do que a novidade. Todas as recomendações serendipitosa são novas, mas o inverso nem sempre é verdadeiro. Considere o caso em que um usuário específico come frequentemente em restaurantes indianos. A recomendação de um novo restaurante paquistanês para esse usuário pode ser nova se esse usuário não tiver comido naquele restaurante antes. No entanto, tal recomendação não é serendipitosa, porque é bem sabido que a comida Indiana e paquistanesa são quase idênticas. Por outro lado, se o sistema de recomendação sugere um novo restaurante etíope para o usuário, então tal recomendação é serendipitosa porque é menos óbvia. Portanto, uma maneira de ver a serendipidade é como um afastamento da "obviedade".

Existem diversas maneiras de medir a serendipidade em sistemas de recomendação. Essa noção também aparece no contexto de aplicações de recuperação de informação [670]. O trabalho em [214] propôs métodos online e offline para avaliar a serendipidade:

1. Métodos online: O sistema de recomendação coleta feedback do usuário sobre a utilidade de uma recomendação e sua obviedade. A fração de recomendações que são úteis e não óbvias é usada como medida da serendipidade.
2. Métodos offline: Também é possível usar um recomendador primitivo para gerar informações sobre a obviedade de uma recomendação de forma automatizada. O recomendador primitivo é normalmente selecionado como um recomendador baseado em conteúdo, que tem alta propensão a recomendar itens óbvios. Em seguida, determina-se a fração dos itens recomendados nas listas top-k que estão corretos (ou seja, com altos valores de avaliações ocultas) e que também não são recomendados pelo recomendador primitivo. Essa fração fornece uma medida da serendipidade.

Vale ressaltar que não basta mensurar a fração de itens não óbvios, pois um sistema pode recomendar itens não relacionados. Portanto, a utilidade dos itens é sempre incorporada à mensuração da serendipidade. A serendipidade tem efeitos importantes a longo prazo na melhoria da taxa de conversão de um sistema de recomendação, mesmo quando se opõe ao objetivo imediato de maximizar a precisão. Diversas métricas para avaliação da serendipidade são discutidas em [214, 450].

7.3.6 Diversidade

A noção de diversidade implica que o conjunto de recomendações propostas dentro de uma única lista de recomendações deve ser o mais diverso possível. Por exemplo, considere o caso em que três filmes são recomendados a um usuário na lista dos 3 principais itens. Se todos os três filmes forem de um gênero específico e contiverem atores semelhantes, haverá pouca diversidade nas recomendações. Se o usuário não gostar da primeira escolha, há uma boa chance de que ele não goste de todos eles. Apresentar diferentes tipos de filmes pode frequentemente aumentar a chance de o usuário selecionar um deles. Observe que a diversidade é sempre medida em um conjunto de recomendações e está intimamente relacionada à novidade e à serendipidade. Garantir maior diversidade pode frequentemente aumentar a novidade e a serendipidade das recomendações.

Além disso, uma maior diversidade de recomendações também pode aumentar a diversidade de vendas e a cobertura do catálogo do sistema.

A diversidade pode ser medida em termos da similaridade centrada no conteúdo entre pares de itens. A representação em espaço vetorial da descrição de cada item é usada para o cálculo da similaridade. Por exemplo, se um conjunto de k itens for recomendado ao usuário, a similaridade entre pares é calculada entre cada par de itens na lista. A similaridade média entre

Todos os pares podem ser relatados como diversidade. Valores mais baixos da similaridade média indicam maior diversidade. A diversidade pode frequentemente fornecer resultados muito diferentes daqueles das métricas de precisão. Uma discussão sobre a conexão entre diversidade e similaridade é fornecida em [560].

7.3.7 Robustez e Estabilidade

Um sistema de recomendação é estável e robusto quando as recomendações não são significativamente afetadas na presença de ataques como avaliações falsas ou quando os padrões nos dados evoluem significativamente ao longo do tempo. Em geral, existem motivações significativas, orientadas pelo lucro, para que alguns usuários insiram avaliações falsas [158, 329, 393, 444]. Por exemplo, o autor ou editor de um livro pode inserir avaliações positivas falsas sobre um livro na Amazon.com ou pode inserir avaliações negativas falsas sobre os livros de um concorrente. Modelos de ataque para sistemas de recomendação são discutidos no Capítulo 12. A avaliação de tais modelos também é estudada no mesmo capítulo. As medidas correspondentes podem ser usadas para estimar a robustez e a estabilidade de tais sistemas contra ataques.

7.3.8 Escalabilidade

Nos últimos anos, tornou-se cada vez mais fácil coletar um grande número de classificações e informações de feedback implícito de diversos usuários. Nesses casos, o tamanho dos conjuntos de dados continua a aumentar ao longo do tempo. Como resultado, tornou-se cada vez mais essencial projetar sistemas de recomendação que possam funcionar de forma eficaz e eficiente na presença de grandes quantidades de dados [527, 528, 587].

Diversas medidas são utilizadas para determinar a escalabilidade de um sistema:

1. Tempo de treinamento: A maioria dos sistemas de recomendação requer uma fase de treinamento, separada da fase de testes. Por exemplo, um algoritmo de filtragem colaborativa baseado em vizinhança pode exigir o pré-cálculo do grupo de pares de um usuário, e um sistema de fatoração de matrizes requer a determinação dos fatores latentes. O tempo total necessário para treinar um modelo é usado como uma das medidas. Na maioria dos casos, o treinamento é realizado offline. Portanto, desde que o tempo de treinamento seja da ordem de algumas horas, é bastante aceitável na maioria dos cenários reais.
2. Tempo de previsão: Após o treinamento do modelo, ele é usado para determinar as principais recomendações para um cliente específico. É crucial que o tempo de previsão seja baixo, pois ele determina a latência com que o usuário recebe as respostas.
3. Requisitos de memória: Quando as matrizes de classificação são grandes, às vezes é um desafio armazenar a matriz inteira na memória principal. Nesses casos, é essencial projetar o algoritmo para minimizar os requisitos de memória. Quando os requisitos de memória se tornam muito altos, torna-se difícil usar os sistemas em ambientes práticos e de larga escala.

A importância da escalabilidade tornou-se particularmente grande nos últimos anos devido à crescente importância do paradigma “big data”.

7.4 Problemas de design na avaliação de recomendação offline

Nesta seção, discutiremos a questão do design da avaliação de recomendações. As discussões nesta seção e na próxima dizem respeito à avaliação da precisão de conjuntos de dados offline e históricos.

É crucial projetar sistemas de recomendação de forma que a precisão não seja grosseiramente

superestimada ou subestimada. Por exemplo, não se pode usar o mesmo conjunto de classificações especificadas para treinamento e avaliação. Fazer isso superestimaria grosseiramente a precisão do algoritmo subjacente. Portanto, apenas uma parte dos dados é usada para treinamento, e o restante é frequentemente usado para testes. A matriz de classificações é tipicamente amostrada de forma entrada a entrada. Em outras palavras, um subconjunto das entradas é usado para treinamento, e as entradas restantes são usadas para avaliação de precisão. Observe que essa abordagem é semelhante à usada para testar algoritmos de classificação e modelagem de regressão. A principal diferença é que os métodos de classificação e modelagem de regressão amostram linhas dos dados rotulados, em vez de amostrar as entradas. Essa diferença ocorre porque as entradas não especificadas são sempre restritas à variável de classe na classificação, enquanto qualquer entrada da matriz de classificações pode ser não especificada. O projeto de sistemas de avaliação de recomendação é muito semelhante ao de sistemas de avaliação de classificadores devido à similaridade entre os problemas de recomendação e classificação.

Um erro comum cometido por analistas no benchmarking de sistemas de recomendação é usar os mesmos dados para ajuste de parâmetros e para testes. Essa abordagem superestima grosseiramente a precisão, pois o ajuste de parâmetros faz parte do treinamento, e o uso de dados de teste no processo de treinamento leva ao overfitting. Para evitar essa possibilidade, os dados são frequentemente divididos em três partes:

1. Dados de treinamento: Esta parte dos dados é usada para construir o modelo de treinamento. Por exemplo, em um modelo de fator latente, esta parte dos dados é usada para criar os fatores latentes a partir da matriz de classificações. Pode-se até usar esses dados para criar vários modelos, a fim de, eventualmente, selecionar o modelo que melhor se adapta ao conjunto de dados em questão.
2. Dados de validação: esta parte dos dados é usada para seleção de modelos e ajuste de parâmetros. Por exemplo, os parâmetros de regularização em um modelo de fator latente podem ser determinados testando a precisão dos dados de validação. Caso vários modelos tenham sido construídos a partir dos dados de treinamento, os dados de validação são usados para determinar a precisão de cada modelo e selecionar o melhor.
3. Dados de teste: Esta parte dos dados é usada para testar a precisão do modelo final (ajustado). É importante que os dados de teste nem sejam analisados durante o processo de ajuste de parâmetros e seleção do modelo para evitar overfitting. Os dados de teste são usados apenas uma vez, no final do processo. Além disso, se o analista usar os resultados dos dados de teste para ajustar o modelo de alguma forma, os resultados serão contaminados com o conhecimento dos dados de teste.

Um exemplo de divisão da matriz de classificações em dados de treinamento, validação e teste é ilustrado na Figura 7.1(a). Observe que os dados de validação também podem ser considerados parte dos dados de treinamento, pois são usados para criar o modelo final ajustado. A divisão da matriz de classificações nas proporções 2:1:1 é particularmente comum. Em outras palavras, metade das classificações especificadas é usada para a construção do modelo e um quarto pode ser usado para a seleção e o teste do modelo, respectivamente. No entanto, quando os tamanhos das matrizes de classificações são grandes, é possível usar proporções muito menores para validação e teste. Esse foi o caso do conjunto de dados do Prêmio Netflix.

7.4.1 Estudo de caso do conjunto de dados de prêmios da Netflix

Um exemplo particularmente instrutivo de um conjunto de dados bem conhecido usado em filtragem colaborativa é o conjunto de dados do Prêmio Netflix, pois demonstra os esforços extraordinários que a Netflix fez para evitar o overfitting no conjunto de teste dos participantes do concurso. Nos dados da Netflix

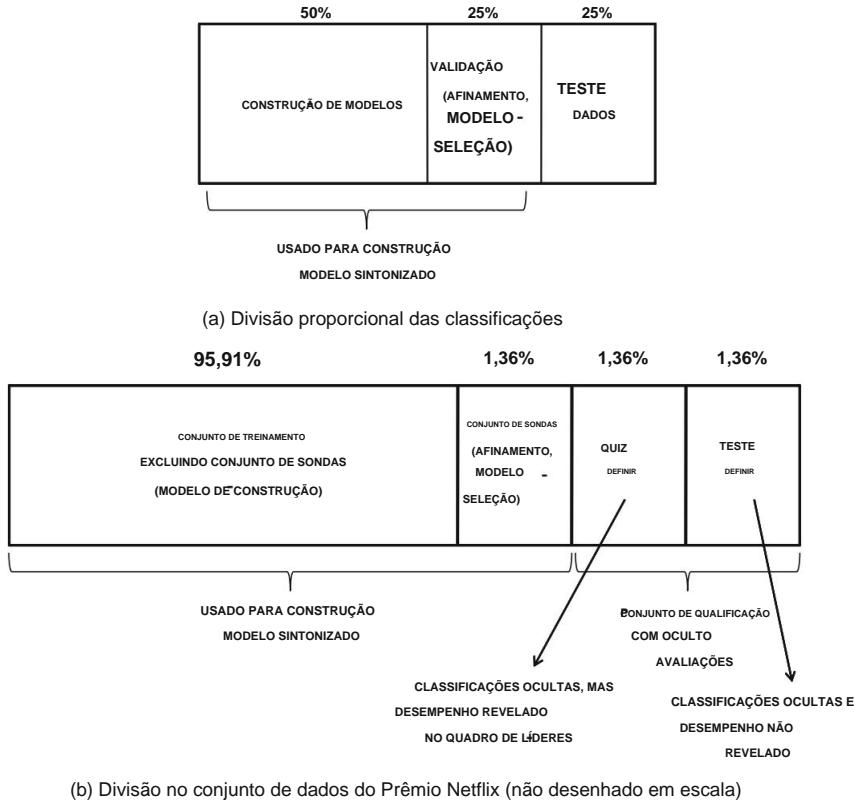


Figura 7.1: Particionando uma matriz de classificação para design de avaliação

conjunto, a maior parte do conjunto de dados continha 95,91% das classificações. Essa parte do conjunto de dados era normalmente usada pelos participantes do concurso para construção de modelos. Outros 1,36% do conjunto de dados eram revelados aos participantes como um conjunto de sondas. Portanto, a parte dos dados para construção de modelos e os dados da sonda juntos continham $95,91 + 1,36 = 97,27\%$ dos dados. O conjunto de sondas era normalmente usado por concursos para várias formas de ajuste de parâmetros e seleção de modelos e, portanto, servia a um propósito muito semelhante como um conjunto de validação. No entanto, diferentes concorrentes usavam o conjunto de sondas de várias maneiras, especialmente porque as classificações no conjunto de sondas eram mais recentes e a distribuição estatística das classificações nos conjuntos de treinamento e de sondas eram ligeiramente diferentes. Para o caso de métodos de conjunto [554], o conjunto de sondas era frequentemente usado para aprender os pesos de vários componentes do conjunto. O conjunto de dados combinado com classificações reveladas (incluindo o conjunto de sondas) corresponde aos dados de treinamento completos, porque foi usado para construir o modelo final ajustado. Uma peculiaridade importante dos dados de treinamento foi que as distribuições do conjunto de sondas e da parte de construção do modelo do conjunto de treinamento não eram exatamente idênticas, embora o conjunto de sondas refletisse as características estatísticas do conjunto qualificador com classificações ocultas. A razão para essa diferença era que a maioria dos dados de classificação eram frequentemente bastante antigos e não refletiam a verdadeira distribuição das classificações mais recentes ou futuras. Os conjuntos de sondas e de qualificação foram baseados em classificações mais recentes, em comparação com os 95,91% das classificações na primeira parte dos dados de treinamento.

As classificações dos 2,7% restantes dos dados foram ocultadas, e apenas trios do formato Usuário, Filme e Data de Classificação foram fornecidos sem as classificações reais. A principal diferença em relação a um conjunto de teste era que os participantes podiam enviar seu desempenho no conjunto de qualificação.

para a Netflix, e o desempenho em metade dos dados qualificadores, conhecido como conjunto de questionários, foi revelado aos participantes em um quadro de classificação. Embora revelar o desempenho no conjunto de questionários aos participantes fosse importante para dar a eles uma ideia da qualidade de seus resultados, o problema em fazer isso era que os participantes poderiam usar o conhecimento do desempenho de seu algoritmo no quadro de classificação para treinar excessivamente seu algoritmo no conjunto de questionários com envios repetidos. Claramente, fazer isso resulta na contaminação dos resultados pelo conhecimento do desempenho no conjunto de questionários, mesmo quando as classificações são ocultadas.

Portanto, a parte do conjunto de qualificação que não estava no conjunto de perguntas e respostas foi usada como conjunto de teste, e os resultados apenas dessa parte do conjunto de qualificação foram usados para determinar o desempenho final para fins de determinação do prêmio. O desempenho no conjunto de perguntas e respostas não teve influência na competição final, exceto para dar aos participantes uma ideia contínua de seu desempenho durante o período da competição. Além disso, os participantes não foram informados sobre qual parte do conjunto de qualificação era o conjunto de perguntas e respostas. Esse arranjo garantiu que um conjunto de dados verdadeiramente fora da amostra fosse usado para determinar os vencedores finais da competição.

A divisão geral do conjunto de dados da Netflix é mostrada na Figura 7.1(b). A única diferença em relação à divisão na Figura 7.1(a) é a presença de um conjunto de questionários adicional. De fato, é possível remover o conjunto de questionários completamente sem afetar o concurso da Netflix de forma significativa, exceto pelo fato de que os participantes não seriam mais capazes de obter uma ideia da qualidade de suas inscrições. De fato, o design da avaliação do Prêmio Netflix é um excelente exemplo da importância de não utilizar nenhum conhecimento do desempenho no conjunto de testes em nenhuma etapa do processo de treinamento até o final. O benchmarking em pesquisa e prática frequentemente falha em atender a esses padrões de uma forma ou de outra.

7.4.2 Segmentação das classificações para treinamento e teste

Na prática, conjuntos de dados reais não são pré-particionados em conjuntos de dados de treinamento, validação e teste. Portanto, é importante poder dividir as entradas de uma matriz de classificações nessas porções automaticamente. A maioria dos métodos de divisão disponíveis, como hold-out e validação cruzada, são usados para dividir o conjunto de dados em duas porções em vez de três. No entanto, é possível obter três porções da seguinte maneira. Primeiro, dividindo as entradas de classificação em porções de treinamento e teste e, em seguida, segmentando a porção de validação dos dados de treinamento, é possível obter os três segmentos necessários. Portanto, a seguir, discutiremos a segmentação da matriz de classificações em porções de treinamento e teste das entradas usando métodos como hold-out e validação cruzada. No entanto, esses métodos também são usados para dividir os dados de treinamento nas porções de construção do modelo e validação.

Essa divisão hierárquica é ilustrada na Figura 7.2. A seguir, usaremos consistentemente a terminologia do primeiro nível de divisão da Figura 7.2 para dados de "treinamento" e "teste", embora a mesma abordagem também possa ser usada para a divisão do segundo nível, para partes de construção e validação do modelo. Essa consistência na terminologia é seguida para evitar confusão.

7.4.2.1 Retenção

No método de retenção, uma fração das entradas na matriz de classificação é oculta e as entradas restantes são usadas para construir o modelo de treinamento. A precisão da previsão das entradas ocultas é então relatada como a precisão geral. Essa abordagem garante que a precisão relatada não seja resultado de sobreajuste ao conjunto de dados específico, pois as entradas

¹⁰O design real em métodos como validação cruzada é um pouco mais complexo porque os dados são segmentados de várias maneiras, embora sejam sempre divididos em duas partes durante uma fase específica de execução do treinamento.

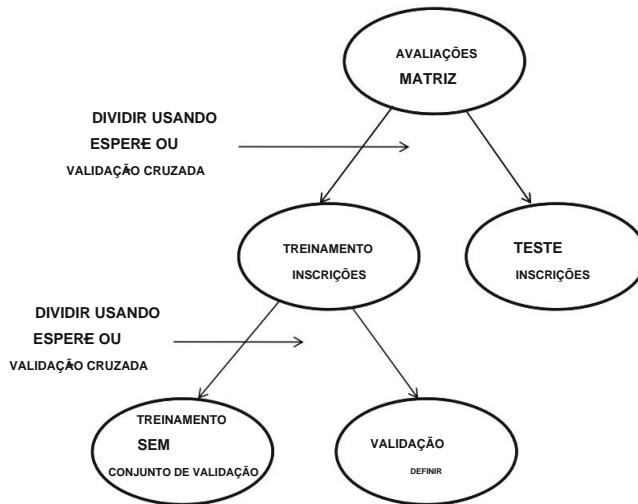


Figura 7.2: Divisão hierárquica de entradas classificadas em partes de treinamento, validação e teste

Os dados utilizados para avaliação são ocultados durante o treinamento. Essa abordagem, no entanto, subestima a precisão real. Primeiro, todas as entradas não são utilizadas no treinamento e, portanto, todo o poder dos dados não é utilizado. Segundo, considere o caso em que as entradas mantidas fora do contexto têm uma classificação média maior do que a matriz de classificações completa. Isso significa que as entradas mantidas dentro do contexto têm uma classificação média menor do que a matriz de classificações, e também as entradas mantidas fora do contexto. Isso levará a um viés pessimista na avaliação.

7.4.2.2 Validação Cruzada

No método de validação cruzada, as entradas de classificação são divididas em q conjuntos iguais. Portanto, se S for o conjunto de entradas especificadas na matriz de classificações R , então o tamanho de cada conjunto, em termos do número de entradas, é $|S|/q$. Um dos q segmentos é usado para teste, e os segmentos restantes ($q - 1$) são usados para treinamento. Em outras palavras, um total de $|S|/q$ entradas são ocultadas durante cada processo de treinamento, e a precisão é então avaliada sobre essas entradas. Esse processo é repetido q vezes usando cada um dos q segmentos como o conjunto de teste.

A precisão média sobre os q diferentes conjuntos de teste é relatada. Observe que essa abordagem pode estimar com precisão a precisão real quando o valor de q é grande. Um caso especial é aquele em que q é escolhido para ser igual ao número de entradas especificadas na matriz de classificações. Portanto, $|S| - 1$ entradas de classificação são usadas para treinamento, e a entrada um é usada para teste. Essa abordagem é chamada de validação cruzada leave-one-out. Embora essa abordagem possa se aproximar da precisão, geralmente é muito caro treinar o modelo $|S|$ vezes.

Na prática, o valor de q é fixado em um número como 10. No entanto, a validação cruzada leave-one-out não é muito difícil de implementar para o caso específico de algoritmos de filtragem colaborativa baseados em vizinhança.

7.4.3 Comparação com o Design de Classificação

O design de avaliação na filtragem colaborativa é muito semelhante ao da classificação. Isso não é uma coincidência. A filtragem colaborativa é uma generalização do problema de classificação, em que qualquer entrada ausente pode ser prevista, em vez de simplesmente uma variável específica, que

é designada como variável dependente. A principal diferença em relação à classificação é que, na classificação, os dados são segmentados por linha (entre as linhas de treinamento e de teste), enquanto na filtragem colaborativa, os dados são segmentados por entrada (entre as entradas de treinamento e de teste). Essa diferença reflete de perto a natureza da relação entre os problemas de classificação e de filtragem colaborativa. Discussões sobre delineamentos de avaliação no contexto do problema de classificação podem ser encontradas em [18, 22].

Uma diferença em relação ao design de classificação é que o desempenho em entradas ocultas frequentemente não reflete o desempenho real do sistema em ambientes reais. Isso ocorre porque as classificações ocultas não são escolhidas aleatoriamente na matriz. Em vez disso, as classificações ocultas são tipicamente itens que o usuário escolheu consumir. Portanto, tais entradas provavelmente apresentarão valores de classificação mais altos em comparação com valores realmente ausentes. Este é um problema de viés de seleção de amostra. Embora esse problema também possa surgir na classificação, ele é muito mais comum em aplicações de filtragem colaborativa. Uma breve discussão sobre essa questão é fornecida na seção 7.6.

7.5 Métricas de precisão na avaliação offline

A avaliação offline pode ser realizada medindo a precisão da previsão dos valores de classificação (por exemplo, com RMSE) ou medindo a precisão da classificação dos itens recomendados. A lógica para o último conjunto de medidas é que os sistemas de recomendação frequentemente fornecem listas classificadas de itens sem prever explicitamente as classificações. Medidas baseadas em classificação frequentemente focam na precisão apenas das classificações dos k itens mais populares, em vez de todos os itens. Isso é particularmente verdadeiro no caso de conjuntos de dados de feedback implícito. Mesmo no caso de classificações explícitas, as avaliações baseadas em classificação fornecem uma perspectiva mais realista da verdadeira utilidade do sistema de recomendação, porque o usuário visualiza apenas os k itens mais populares, em vez de todos os itens. No entanto, para benchmarking, a precisão das previsões de classificação é geralmente preferida devido à sua simplicidade. Na competição do Prêmio Netflix, a medida RMSE foi usada para a avaliação final. A seguir, ambas as formas de avaliação de precisão serão discutidas.

7.5.1 Medindo a precisão da previsão de classificações

Após a finalização do projeto de avaliação de um experimento offline, a precisão precisa ser medida no conjunto de teste. Como discutido anteriormente, seja S o conjunto de entradas especificadas (observadas) e $E \subseteq S$ o conjunto de entradas no conjunto de teste usado para avaliação. Cada entrada em E é um par de índices usuário-item da forma (u, j) correspondente a uma posição na matriz de classificações.

Observe que o conjunto E pode corresponder às entradas mantidas no método de retenção ou pode corresponder a uma das partições de tamanho $|S|/q$ durante a validação cruzada.

Seja r_{uj} o valor da classificação (oculta) da entrada $(u, j) \in E$, que é usada no conjunto de teste. Além disso, seja \hat{r}_{uj} a classificação prevista da entrada (u, j) pelo algoritmo de treinamento específico que está sendo usado. O erro específico da entrada é dado por $e_{uj} = \hat{r}_{uj} - r_{uj}$. Esse erro pode ser aproveitado de várias maneiras para calcular o erro geral sobre o conjunto E de entradas no qual a avaliação é realizada. Um exemplo é o erro quadrático médio, denotado por MSE:

$$\text{MSE} = \frac{\sum_{(u,j) \in E} e_{uj}^2}{|E|} \quad (7.4)$$

Claramente, valores menores do MSE indicam desempenho superior. A raiz quadrada desse valor é chamada de erro quadrático médio (RMSE) e é frequentemente usada no lugar do MSE:

$$\text{RMSE} = \sqrt{\frac{\sum_{(u,j) \in E} e_{uj}^2}{|E|}} \quad (7.5)$$

O RMSE é expresso em unidades de classificação, e não em unidades de classificação ao quadrado, como o MSE. O RMSE foi usado como métrica padrão para o concurso Netflix Prize. Uma característica do RMSE é que ele tende a penalizar desproporcionalmente erros maiores devido ao termo ao quadrado dentro da soma. Uma medida, conhecida como erro médio absoluto (MAE), não penaliza desproporcionalmente erros maiores:

$$\text{MAE} = \frac{\sum_{(u,j) \in E} |e_{uj}|}{|E|} \quad (7.6)$$

Outras medidas relacionadas, como o RMSE normalizado (NRMSE) e o MAE normalizado (NMAE) são definidos de forma semelhante, exceto que cada um deles é dividido pelo intervalo $r_{\max} - r_{\min}$ das classificações:

$$\begin{aligned} \text{NRMSE} &= \frac{\text{RMSE}}{r_{\max} - r_{\min}} \\ \text{NMAE} &= \frac{\text{MAE}}{r_{\max} - r_{\min}} \end{aligned}$$

Os valores normalizados do RMSE e do MAE sempre se situam no intervalo (0, 1) e, portanto, são mais interpretáveis de um ponto de vista intuitivo. Também é possível usar esses valores para comparar o desempenho de um algoritmo específico em diferentes conjuntos de dados com escalas de classificação variadas.

7.5.1.1 RMSE versus MAE

RMSE ou MAE são melhores como medida de avaliação? Não há uma resposta clara para essa pergunta, pois isso depende da aplicação em questão. Como o RMSE soma os erros quadráticos, ele é mais significativamente afetado por grandes valores de erro ou outliers. Algumas classificações mal previstas podem prejudicar significativamente a medida RMSE. Em aplicações onde a robustez da previsão em várias classificações é muito importante, o RMSE pode ser uma medida mais apropriada.

Por outro lado, o MAE reflete melhor a precisão quando a importância dos valores discrepantes na avaliação é limitada. O principal problema com o RMSE é que ele não reflete fielmente o erro médio e, às vezes, pode levar a resultados enganosos [632].

Claramente, a escolha específica deve depender da aplicação em questão. Uma discussão sobre os benefícios relativos dos dois tipos de medidas pode ser encontrada em [141].

7.5.1.2 Impacto da Cauda Longa

Um problema com essas métricas é que elas são fortemente influenciadas pelas classificações dos itens populares. Os itens que recebem pouquíssimas classificações são ignorados. Conforme discutido no Capítulo 2, as matrizes de classificação exibem uma propriedade de cauda longa, na qual a grande maioria dos itens é comprada (ou classificada) raramente. Replicamos a Figura 2.1 do Capítulo 2 na Figura 7.3. O eixo X representa os índices dos itens em ordem decrescente de popularidade, e o eixo Y indica a frequência de classificação. É evidente que apenas alguns itens recebem um grande número de classificações, enquanto a maioria dos itens restantes recebe poucas classificações. Estes últimos constituem a cauda longa. Infelizmente, os itens na cauda longa frequentemente contribuem para a grande maioria do lucro dos comerciantes [49]. Como resultado, os itens mais importantes geralmente recebem o menor peso no processo de avaliação. Além disso, muitas vezes é muito mais difícil prever os valores de

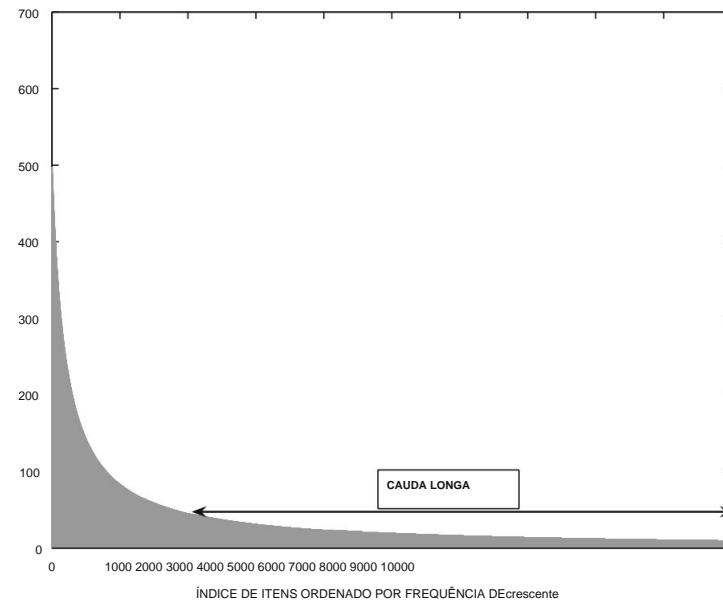


Figura 7.3: A longa cauda das frequências de classificação (Revisitando a Figura 2.1 do Capítulo 2)

as classificações na cauda longa devido à maior dispersão local [173]. Portanto, as precisões de previsão em itens esparsos serão tipicamente diferentes daquelas em itens populares. Uma maneira de lidar com esse problema é calcular o RMSE ou o MAE separadamente para todas as classificações ocultas associadas a cada item e, em seguida, calcular a média dos diferentes itens de forma ponderada. Em outras palavras, os cálculos de precisão das Equações 7.5 e 7.6 podem ser ponderados com um peso específico do item, dependendo da importância relativa, lucro ou utilidade para o comerciante. Também é possível realizar esses cálculos com ponderação específica do usuário (em vez de ponderação específica do item), embora a aplicabilidade prática da ponderação específica do usuário seja limitada.

7.5.2 Avaliando a Classificação por Correlação As medidas

mencionadas acima são projetadas para avaliar a precisão da previsão do valor real da classificação de uma combinação usuário-item. Na prática, o sistema de recomendação cria uma classificação de itens para um usuário, e os k itens mais recomendados são os mais recomendados. O valor de k pode variar de acordo com o sistema, o item e o usuário em questão. Em geral, é desejável que itens com classificação alta sejam classificados acima de itens com classificação baixa. Considere um usuário u , para o qual as classificações do conjunto I_u de itens foram ocultadas por uma estratégia de validação cruzada ou de retenção.

Por exemplo, se as classificações do primeiro, terceiro e quinto itens (colunas) do usuário (l_u) estiverem ocultas para fins de avaliação, então temos $I_u = \{1, 3, 5\}$.

Gostaríamos de mensurar o quanto bem as ordenações de verdade básica das classificações em I_u se relacionam com a ordenação prevista pelo sistema de recomendação para o conjunto I_u . Uma questão importante a ser lembrada é que as classificações são tipicamente escolhidas a partir de uma escala discreta, e existem muitos empates na verdade básica. Portanto, é importante que as medidas de classificação não penalizem o sistema por classificar um item acima de outro quando eles estão empatados na verdade básica. A classe mais comum de métodos é usar coeficientes de correlação de classificação. Os dois coeficientes de correlação de classificação mais comumente usados são os seguintes:

1. Coeficiente de correlação de Spearman: O primeiro passo é classificar todos os itens de 1 a $|u|$, tanto para a previsão do sistema de recomendação quanto para a verdade básica. O coeficiente de correlação de Spearman é simplesmente igual ao coeficiente de correlação de Pearson aplicado a essas classificações. O valor calculado sempre varia entre ($\hat{y}_1, +1$), e valores positivos maiores são mais desejáveis.

O coeficiente de correlação de Spearman é específico para o usuário u e pode ser calculado para todos os usuários, obtendo-se um valor global. Alternativamente, a correlação de classificação de Spearman pode ser calculada para todas as avaliações ocultas de todos os usuários de uma só vez, em vez de calcular valores específicos do usuário e calculá-los.

Um problema com essa abordagem é que a verdade básica conterá muitos empates e, portanto, o desempate aleatório pode levar a algum ruído na avaliação. Para esse propósito, utiliza-se uma abordagem conhecida como Spearman corrigido por empate. Uma maneira de realizar a correção é usar a classificação média de todos os empates, em vez de usar o desempate aleatório. Por exemplo, se a classificação da verdade básica das duas primeiras classificações for idêntica em uma lista de quatro itens, em vez de usar as classificações {1, 2, 3, 4}, pode-se usar as classificações {1.5, 1.5, 3, 4}.

2. Coeficiente de correlação de classificação de Kendall: Para cada par de itens $j, k \neq i$, o seguinte crédito $C(j, k)$ é calculado comparando a classificação prevista com a classificação da verdade básica desses itens:

$$C(j, k) = \begin{cases} +1 & \text{se os itens } j \text{ e } k \text{ estiverem na mesma ordem relativa em} \\ & \text{classificação de verdade básica e classificação prevista (concordante)} \\ -1 & \text{se os itens } j \text{ e } k \text{ estiverem em uma ordem relativa diferente em} \\ & \text{classificação de verdade básica e classificação prevista (discordantes)} \\ 0 & \text{se os itens } j \text{ e } k \text{ estiverem empatados em qualquer um dos} \\ & \text{classificações de verdade básica ou classificação prevista} \end{cases} \quad (7.7)$$

Em seguida, o coeficiente de correlação de classificação de Kendall \hat{y}_u , que é específico para o usuário u , é calculado como o valor médio de $C(j, k)$ sobre todos os pares $|u|(|u| - 1)/2$ de itens de teste para o usuário i :

$$\hat{y}_u = \frac{\sum_{j < k} C(j, k)}{|u| \cdot (|u| - 1)/2} \quad (7.8)$$

Uma maneira diferente de entender o coeficiente de correlação de classificação de Kendall é a seguinte:

$$\hat{y}_u = \frac{\text{Número de pares concordantes} - \text{Número de pares discordantes}}{\text{Número de pares em } |u|} \quad (7.9)$$

Observe que este valor é um valor específico do cliente do coeficiente de Kendall. O valor de \hat{y}_u pode ser calculado sobre todos os usuários u para obter uma medida global heurística. Alternativamente, pode-se realizar o cálculo do coeficiente de Kendall da Equação 7.8 sobre todos os pares ocultos usuário-item, em vez de apenas os pares do cliente u , a fim de obter um valor global \hat{y} .

Diversas outras medidas, como a medida de desempenho baseada na distância normalizada (NDPM), foram propostas na literatura. Consulte as notas bibliográficas.

7.5.3 Avaliando a classificação via utilidade

Na discussão anterior, a classificação da verdade básica é comparada à classificação do sistema de recomendação. Métodos baseados em utilidade usam a classificação da verdade básica em combinação com a classificação do sistema de recomendação. Para o caso de conjuntos de dados de feedback implícito, a classificação é substituída por um valor de 0 a 1, dependendo se o cliente consumiu ou não o item. O objetivo geral dos métodos baseados em utilidade é criar uma quantificação precisa de quão útil o cliente pode achar a classificação do sistema de recomendação. Um princípio importante subjacente a tais métodos é que as listas de recomendações são curtas em comparação com o número total de itens. Portanto, a maior parte da utilidade de uma classificação específica deve ser baseada na relevância dos itens que estão no topo da lista de recomendações. Nesse sentido, a medida RMSE tem uma fraqueza porque pondera igualmente os erros nos itens com classificação baixa em comparação com aqueles nos itens com classificação alta. Foi sugerido [713] que pequenas alterações no RMSE, como 1%, podem levar a grandes alterações de mais de 15% nas identidades dos itens mais bem avaliados. Esses são os únicos itens que o usuário final do sistema de recomendação realmente verá. Da mesma forma, medidas baseadas em utilidade quantificam a utilidade de uma lista de recomendações, atribuindo maior importância aos itens mais bem avaliados.

Como nas seções anteriores, assume-se que a classificação de cada item em I_u é ocultada do sistema de recomendação antes da avaliação. Aqui, I_u representa o conjunto de itens avaliados pelo usuário u , que são ocultados do sistema de recomendação antes da avaliação.

Desenvolveremos quantificações de utilidade globais e específicas do usuário.

Na classificação baseada em utilidade, a ideia básica é que cada item em I_u tem uma utilidade para o usuário, que depende tanto de sua posição na lista de recomendações quanto de sua classificação de verdade básica. Um item que tem uma classificação de verdade básica mais alta obviamente tem maior utilidade para o usuário. Além disso, itens com classificação mais alta na lista de recomendações têm maior utilidade para o usuário, pois têm maior probabilidade de serem notados (em virtude de sua posição) e, eventualmente, selecionados. O ideal seria que itens com classificação mais alta de veracidade fossem colocados o mais alto possível na lista de recomendações.

Como esses componentes baseados em classificação e em classificação são definidos? Para qualquer item $j \in I_u$, sua utilidade baseada em classificação para o usuário i é assumida como $\max\{r_{uj} / C_u, 0\}$, onde C_u é um valor de classificação de equilíbrio (neutro) para o usuário u . Por exemplo, C_u pode ser definido como a classificação média do usuário u . Por outro lado, a utilidade baseada em classificação do item é $2^{(v_j - \bar{v})/\bar{v}}$, onde v_j é a classificação do item j na lista de itens recomendados e \bar{v} é um parâmetro de meia-vida. Em outras palavras, a utilidade baseada em classificação decai exponencialmente com sua classificação, e descer nas classificações em \bar{v} reduz a utilidade por um fator de 2. A lógica do componente de classificação baseado em decaimento é garantir que a utilidade final de uma classificação específica seja regulada principalmente pelos primeiros itens. Afinal, o usuário raramente navega pelos itens que estão muito baixos na lista. A utilidade $F(u, j)$ do item $j \in I_u$ para o usuário u é definida como o produto dos valores de utilidade baseados na classificação e na classificação:

$$F(u, j) = \frac{\max\{r_{uj} / C_u, 0\}}{2^{(v_j - \bar{v})/\bar{v}}} \quad (7.10)$$

O R-score, que é específico para o usuário u , é a soma de $F(u, j)$ sobre todas as classificações ocultas em I_u :

$$R\text{-pontuação}(u) = \sum_{j \in I_u} F(u, j) \quad (7.11)$$

Observe que o valor de v_j pode assumir qualquer valor de 1 a n , onde n é o número total de itens. No entanto, na prática, muitas vezes, restringe-se o tamanho da lista recomendada a um

valor máximo de L. Portanto, é possível calcular o R-score sobre uma lista recomendada de tamanho específico L em vez de usar todos os itens, da seguinte forma:

$$R\text{-pontuação}(u) = \frac{F(u, j)}{\sum_{j=1}^L F(u, j)} \quad (7.12)$$

A ideia aqui é que classificações abaixo de L não têm utilidade para o usuário porque a lista de recomendações tem tamanho L. Essa variação se baseia no princípio de que as listas de recomendações costumam ser muito curtas em comparação com o número total de itens. O R-score geral pode ser calculado somando esse valor para todos os usuários.

$$R\text{-score} = \frac{1}{m} \sum_{u=1}^m R\text{-score}(u) \quad (7.13)$$

A queda exponencial na utilidade implica que os usuários estão interessados apenas nos itens mais bem classificados e não prestam muita atenção aos itens mais abaixo. Isso pode não ser verdade em todas as aplicações, especialmente em sistemas de recomendação de notícias, onde os usuários normalmente navegam por vários itens mais abaixo na lista de itens recomendados. Nesses casos, a taxa de desconto deve ser definida de forma mais suave. Um exemplo dessa medida é o ganho cumulativo descontado (GDC). Nesse caso, o fator de desconto do item j é definido como $\log_2(v_j + 1)$, onde v_j é a classificação do item j no conjunto de teste Iu. Então, o ganho cumulativo descontado é definido da seguinte forma:

$$DCG = \frac{1}{m} \sum_{u=1}^m \frac{gu_j}{\log_2(v_j + 1)} \quad (7.14)$$

Aqui, gu_j representa a utilidade (ou ganho) do usuário u ao consumir o item j. Normalmente, o valor de gu_j é definido como uma função exponencial da relevância (por exemplo, classificações não negativas ou taxas de acerto do usuário):

$$gu_j = 2^{reluj} \quad (7.15)$$

Aqui, $reluj$ é a relevância da verdade básica do item j para o usuário u, que é calculada como uma função heurística das avaliações ou acertos. Em muitos cenários, as avaliações brutas são utilizadas. É comum calcular o ganho cumulativo descontado sobre uma lista de recomendações de tamanho específico L, em vez de usar todos os itens:

$$DCG = \frac{1}{m} \sum_{u=1}^m \frac{gu_j}{\log_2(v_j + 1)} \quad (7.16)$$

A ideia básica é que as listas recomendadas não tenham tamanho maior que L.

Então, o ganho cumulativo descontado normalizado (NDCG) é definido como a razão entre o ganho cumulativo descontado e seu valor ideal, que também é chamado de ganho cumulativo descontado ideal (IDCG).

$$NDCG = \frac{DCG}{IDCG} \quad (7.17)$$

O ganho cumulativo descontado ideal é calculado repetindo o cálculo para DCG, exceto que as classificações de verdade básica são usadas no cálculo.

Outra medida comumente usada é a taxa média de acerto recíproco (ARHR) [181].

Esta medida foi projetada para conjuntos de dados de feedback implícito, nos quais cada valor de $ruj \in \{0, 1\}$. Portanto, um valor $ruj = 1$ representa um "acerto" em que um cliente comprou ou clicou em um item. Um valor $ruj = 0$ corresponde a uma situação em que um cliente não comprou.

ou clicou em um item. Nessa configuração de feedback implícito, os valores ausentes na matriz de classificação são considerados 0.

Neste caso, a taxa de desconto baseada na classificação é $1/v_j$, onde v_j é a classificação do item j na lista recomendada, e a utilidade do item é simplesmente o valor de "classificação" oculto $r_{uj} \in \{0, 1\}$.

Observe que a taxa de desconto não é tão rápida quanto a métrica R-score, mas é mais rápida que o DCG.

Portanto, a utilidade combinada de um item é dada por r_{uj}/v_j . Esta expressão representa a contribuição do item $j \in I_u$ para a utilidade. Então, a métrica ARHR para o usuário i é definida pela soma destes valores sobre todos os itens ocultos em I_u :

$$\text{ARHR}(u) = \frac{\sum_{j \in I_u} r_{uj}}{\sum_{j \in I_u} v_j} \quad (7.18)$$

Também é possível definir a taxa de acerto recíproca média para uma lista recomendada de tamanho L adicionando apenas os valores de utilidade para os quais $v_j \leq L$.

$$\text{ARHR}(u) = \frac{\sum_{j \in I_u, v_j \leq L} r_{uj}}{\sum_{j \in I_u, v_j \leq L} v_j} \quad (7.19)$$

Uma peculiaridade da taxa de acerto recíproca média é que ela é normalmente usada quando o valor de $|I_u|$ é exatamente 1, e quando o valor r_{uj} do item correspondente (oculto) $j \in I_u$ é sempre 1. Portanto, há exatamente um item oculto para cada usuário, e o usuário sempre comprou ou clicou neste item. Em outras palavras, a taxa de acerto recíproca média recompensa a utilidade (de forma recíproca de classificação) por recomendar a única resposta correta em uma posição alta na lista de recomendações. Este foi o cenário em que esta medida foi introduzida [181], embora seja possível generalizá-la para configurações arbitrárias em termos do número de itens ocultos e configurações de feedback explícito. A expressão acima mencionada fornece esta definição generalizada porque é possível usar um conjunto I_u de tamanho arbitrário em uma configuração de feedback explícito. O valor ARHR global é calculado pela média deste valor sobre os m usuários:

$$\text{ARHR} = \frac{\frac{1}{m} \sum_{u=1}^m \text{ARHR}(u)}{m} \quad (7.20)$$

A taxa de acerto recíproca média também é chamada de classificação recíproca média (MRR). Nos casos em que o valor de $|I_u|$ é 1, a taxa de acerto recíproca média sempre está no intervalo $(0, 1)$. Nesses casos, a entrada oculta geralmente é um item para o qual $r_{uj} = 1$ e o comprimento da lista de recomendações é restrito a L . Observe que apenas "acertos" contribuem para a utilidade nesses casos. Uma simplificação dessa medida é a taxa de acerto, na qual a ponderação de classificação recíproca não é usada, e o valor de $|I_u|$ é exatamente 1. Portanto, a taxa de acerto (HR) é simplesmente a fração de usuários para os quais a resposta correta está incluída na lista de recomendações de comprimento L . A desvantagem da taxa de acerto é que ela dá igual importância a um acerto, independentemente de sua posição na lista de recomendações.

ARHR e HR são quase sempre utilizados em conjuntos de dados de feedback implícito, nos quais os valores ausentes são tratados como 0. No entanto, a definição da Equação 7.19 é apresentada de forma mais geral. Tal definição também pode ser utilizada no contexto de conjuntos de dados de feedback explícito, nos quais os valores de r_{uj} não precisam ser extraídos de $\{0, 1\}$. Nesses casos, as classificações de qualquer número de itens de cada usuário são ocultadas, e os valores das classificações ocultas podem ser arbitrários. Além disso, os valores ausentes não precisam ser tratados como 0, e I_u é sempre selecionado dentre os itens observados.

Uma medida relacionada é a precisão média (MAP), que calcula a fração de itens relevantes em uma lista recomendada de comprimento L para um determinado usuário. Vários itens igualmente espaçados

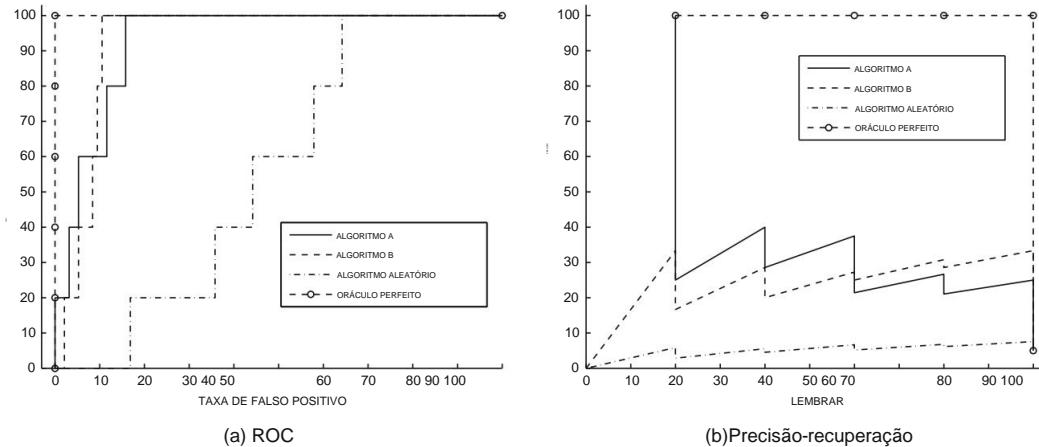


Figura 7.4: Curva ROC e curvas de precisão-recall

valores de L são usados, e a precisão é calculada sobre essas listas de recomendações de vários comprimentos. A precisão resultante é então calculada para todos os usuários.

Várias outras medidas foram propostas na literatura para avaliar a eficácia das classificações. Por exemplo, o índice de elevação [361] divide os itens classificados em decis para calcular uma pontuação de utilidade. Consulte as notas bibliográficas.

7.5.4 Avaliação da classificação por meio da característica operacional do receptor

Métodos de classificação são usados frequentemente na avaliação do consumo real de itens.

Por exemplo, a Netflix pode recomendar um conjunto de itens classificados para um usuário, e o usuário pode eventualmente consumir apenas um subconjunto desses itens. Portanto, esses métodos são bem adequados para conjuntos de dados de feedback implícito, como vendas, cliques ou visualizações de filmes. Essas ações podem ser representado na forma de matrizes de classificação unárias, nas quais os valores ausentes são considerados para ser equivalente a 0. Portanto, a verdade fundamental é de natureza binária.

Os itens que são eventualmente consumidos também são chamados de pontos positivos da verdade fundamental ou verdadeiros positivos. O algoritmo de recomendação pode fornecer uma lista classificada de qualquer número de itens. Qual a porcentagem desses itens é relevante? Uma questão fundamental aqui é que a resposta a esta questão depende do tamanho da lista recomendada. Alterar o número de itens recomendados na lista classificada têm um efeito direto na compensação entre a fração de itens recomendados que são realmente consumidos e a fração de itens consumidos que são capturados pelo sistema de recomendação. Este trade-off pode ser medido de duas maneiras diferentes maneiras com o uso de uma curva de precisão-recall ou de uma curva característica de operação do receptor (ROC). Esses gráficos de compensação são comumente usados na detecção de classes raras, avaliação de análise de outliers, e recuperação de informações. Na verdade, esses gráficos de compensação podem ser usados em qualquer aplicação onde uma verdade básica binária é comparada a uma lista classificada descoberta por um algoritmo.

A suposição básica é que é possível classificar todos os itens usando uma pontuação numérica, que é a saída do algoritmo em questão. Apenas os itens principais são recomendados. Por variando o tamanho da lista recomendada, pode-se então examinar a fração de itens relevantes (itens positivos de verdade fundamental) na lista e a fração de itens relevantes que são perdidos pela lista. Se a lista recomendada for muito pequena, o algoritmo perderá itens relevantes (falsos negativos). Por outro lado, se uma lista muito grande for recomendada, isso levará a muitas recomendações espúrias que nunca são utilizadas pelo usuário (falsos positivos).

Tabela 7.1: Algoritmo de classificação de instâncias positivas de

verdade fundamental	Classificação de itens que são realmente usados (verdades básicas positivas) 1, 5,
Algoritmo A	8, 15, 20 3, 7,
Algoritmo B	11, 13, 15 17, 36,
Algoritmo Aleatório	45, 59, 66 1, 2, 3, 4,
Oráculo Perfeito	5

Isso leva a um trade-off entre falsos positivos e falsos negativos. O problema é que o tamanho correto da lista de recomendações nunca é conhecido exatamente em um cenário real.

No entanto, toda a curva de compensação pode ser quantificada usando uma variedade de medidas, e dois algoritmos podem ser comparados em toda a curva de compensação. Dois exemplos dessas curvas são a curva de precisão-recall e a curva ROC (Receiver Operating Characteristic).

Suponha que se selecione o conjunto t mais alto de itens classificados para recomendar ao usuário. Para qualquer valor t do tamanho da lista de recomendações, o conjunto de itens recomendados é denotado por $S(t)$. Observe que $|S(t)| = t$. Portanto, à medida que t muda, o tamanho de $S(t)$ também muda. Seja G o verdadeiro conjunto de itens relevantes (positivos da verdade fundamental) que são consumidos pelo usuário. Então, para qualquer tamanho t da lista de recomendações, a precisão é definida como a porcentagem de itens recomendados que realmente se mostram relevantes (ou seja, consumidos pelo usuário).

$$P\text{ precisão}(t) = 100 \cdot \frac{|S(t) \cap G|}{|S(t)|}$$

O valor de $P\text{ precisão}(t)$ não é necessariamente monótono em t , pois tanto o numerador quanto o denominador podem mudar com t de forma diferente. A revocação é definida como a porcentagem de verdades básicas positivas que foram recomendadas como positivas para uma lista de tamanho t .

$$Recall(t) = 100 \cdot \frac{|S(t) \cap G|}{|G|}$$

Embora exista um equilíbrio natural entre precisão e recall, esse equilíbrio não é necessariamente monótono. Em outras palavras, um aumento no recall nem sempre leva a uma redução na precisão. Uma maneira de criar uma única medida que resuma precisão e recall é a medida F1, que é a média harmônica entre a precisão e o recall.

$$F1(t) = \frac{2 \cdot \text{Precisão}(t) \cdot \text{Recall}(t)}{\text{Precisão}(t) + \text{Recall}(t)} \quad (7.21)$$

Embora a medida $F1(t)$ forneça uma quantificação melhor do que a precisão ou a recuperação, ela ainda depende do tamanho t da lista recomendada e, portanto, ainda não é uma representação completa do trade-off entre precisão e recuperação. É possível examinar visualmente todo o trade-off entre precisão e recuperação variando o valor de t e plotando a precisão versus a recuperação. Como mostrado posteriormente com um exemplo, a falta de monotonicidade da precisão torna os resultados mais difíceis de interpretar intuitivamente.

Uma segunda maneira de gerar o trade-off de forma mais intuitiva é por meio do uso da curva ROC. A taxa de verdadeiros positivos, que é a mesma que a de recall, é definida como a porcentagem de positivos verdadeiros que foram incluídos na lista de recomendações de tamanho t .

$$TPR(t) = Recall(t) = 100 \cdot \frac{|S(t) \cap G|}{|G|}$$

A taxa de falsos positivos FPR(t) é a porcentagem de falsos positivos relatados na lista de recomendações em relação aos falsos negativos (ou seja, itens irrelevantes não consumidos pelo usuário). Portanto, se U representa o universo de todos os itens, o conjunto de falsos negativos é dado por $(U \setminus G)$, e a parte falsamente relatada na lista de recomendações é $(S(t) \setminus G)$.

Portanto, a taxa de falsos positivos é definida da seguinte forma:

$$FPR(t) = 100 \cdot \frac{|S(t) \setminus G|}{|U \setminus G|} \quad (7.22)$$

A taxa de falsos positivos pode ser vista como um tipo de recall "ruim", no qual a fração dos negativos da verdade fundamental (ou seja, itens não consumidos), que são incorretamente capturados na lista recomendada $S(t)$, é relatada. A curva ROC é definida plotando o $FPR(t)$ no eixo X e o $TPR(t)$ no eixo Y para valores variáveis de t . Em outras palavras, a curva ROC plota o recall "bom" em relação ao recall "ruim". Observe que ambas as formas de recall estarão em 100% quando $S(t)$ for definido para todo o universo de itens. Portanto, os pontos finais da curva ROC estão sempre em $(0, 0)$ e $(100, 100)$, e espera-se que um método aleatório exiba desempenho ao longo da linha diagonal que conecta esses pontos. O aumento obtido acima dessa linha diagonal fornece uma ideia da precisão da abordagem. A área sob a curva ROC fornece uma avaliação quantitativa concreta da eficácia de um método específico.

Embora seja possível usar diretamente a área mostrada na Figura 7.4(a), a curva ROC em forma de escada é frequentemente modificada para usar segmentos lineares locais que não são paralelos nem ao eixo X nem ao eixo Y. A regra trapezoidal [195] é então usada para calcular a área com um pouco mais de precisão. Do ponto de vista prático, essa mudança geralmente faz pouca diferença no cálculo final.

Para ilustrar os insights obtidos a partir dessas diferentes representações gráficas, considere um exemplo de cenário com 100 itens, no qual 5 itens são verdadeiramente relevantes. Dois algoritmos, A e B, são aplicados a esse conjunto de dados, que classificam todos os itens de 1 a 100, com as classificações mais baixas sendo selecionadas primeiro na lista recomendada. Assim, os valores da taxa de verdadeiro-positivo e da taxa de falso-positivo podem ser gerados a partir das classificações dos 5 itens relevantes. Na Tabela 7.1, algumas classificações hipotéticas para os 5 itens verdadeiramente relevantes foram ilustradas para os diferentes algoritmos. Além disso, as classificações dos itens positivos da verdade fundamental para um algoritmo aleatório foram indicadas. Esse algoritmo classifica todos os itens aleatoriamente. Da mesma forma, as classificações para um algoritmo de "oráculo perfeito", que classifica os 5 itens principais corretos na lista recomendada, também foram ilustradas na tabela. As curvas ROC resultantes são ilustradas na Figura 7.4(a). As curvas de precisão-recall correspondentes são ilustradas na Figura 7.4(b). Observe que as curvas ROC são sempre crescentes monotonicamente, enquanto as curvas de precisão-recall não são monotônicas. Embora as curvas de precisão-recall não sejam tão facilmente interpretáveis quanto as curvas ROC, é fácil observar que as tendências relativas entre diferentes algoritmos são as mesmas em ambos os casos. Em geral, as curvas ROC são usadas com mais frequência devido à maior facilidade de interpretação.

O que essas curvas realmente nos dizem? Para casos em que uma curva domina estritamente a outra, fica claro que o algoritmo para a primeira curva é superior. Por exemplo, fica imediatamente evidente que o algoritmo oráculo é superior a todos os algoritmos e que o algoritmo aleatório é inferior a todos os outros algoritmos. Por outro lado, os algoritmos A e B mostram dominância em diferentes partes da curva ROC. Nesses casos, é difícil dizer que um algoritmo é estritamente superior. A partir da Tabela 7.1, fica claro que o Algoritmo A classifica três itens relevantes muito bem, mas os dois itens restantes são mal classificados.

No caso do Algoritmo B, os itens com classificação mais alta não são tão bem classificados quanto o Algoritmo A, embora todos os cinco itens relevantes sejam determinados muito antes em termos de limite de classificação. Correspondentemente, o Algoritmo A domina na parte inicial da curva ROC, enquanto

O algoritmo B domina na última parte. É possível usar a área sob a curva ROC como um indicador da eficácia geral do algoritmo. No entanto, nem todas as partes da curva ROC são igualmente importantes, pois geralmente há limites práticos para o tamanho da lista recomendada.

A descrição acima ilustra a geração de curvas ROC específicas para cada cliente, pois as curvas ROC são específicas para cada usuário. Também é possível gerar curvas ROC globais classificando pares usuário-item e, em seguida, usando a mesma abordagem discutida acima. Para classificar pares usuário-item, presume-se que o algoritmo tenha um mecanismo para classificá-los usando valores de afinidade previstos. Por exemplo, as classificações previstas para pares usuário-item podem ser usadas para classificá-los.

7.5.5 Qual medida de classificação é melhor?

Embora as curvas ROC sejam frequentemente utilizadas para avaliar sistemas de recomendação, elas nem sempre refletem o desempenho da perspectiva do usuário final. Em muitos cenários, o usuário final vê apenas um pequeno subconjunto de itens com classificação superior. Medidas como o ROC e o coeficiente de Kendall, que tratam itens com classificação superior e inferior igualmente, são incapazes de capturar a maior importância de itens com classificação superior. Por exemplo, a classificação relativa entre dois itens classificados em primeiro e segundo lugar na lista de recomendações é muito mais importante do que a classificação relativa de dois itens, que estão classificados em 100º e 101º lugar na lista. Nesse contexto, medidas baseadas em utilidade, como o NDCG, fazem um trabalho muito melhor do que coeficientes de correlação de classificação ou medidas ROC na distinção entre itens com classificação superior e inferior.

7.6 Limitações das Medidas de Avaliação

Medidas de avaliação baseadas em precisão apresentam uma série de fragilidades decorrentes do viés de seleção em sistemas de recomendação. Em particular, as entradas ausentes em uma matriz de classificação não são aleatórias, pois os usuários tendem a classificar itens mais populares. Conforme mostrado na Figura 7.3, alguns itens são classificados por muitos usuários, enquanto a grande maioria dos itens pode ser encontrada na cauda longa. As distribuições das classificações de itens populares costumam ser diferentes daquelas dos itens na cauda longa. Quando um item é muito popular, é mais provável que seja devido ao seu conteúdo notável. Esse fator também afetará a classificação desse item. Como resultado, a precisão da maioria dos algoritmos de recomendação é diferente nos itens mais populares em comparação com os itens na cauda longa [564]. De forma mais geral, o fato de um determinado usuário ter optado por não classificar um item específico até o momento tem um impacto significativo em qual seria sua classificação se o usuário fosse forçado a classificar todos os itens. Essa questão é apresentada em [184] em um contexto um pouco diferente, da seguinte forma:

Intuitivamente, um processo simples poderia explicar os resultados: os usuários optaram por avaliar as músicas que ouvem e ouviram músicas que esperavam gostar, evitando gêneros dos quais não gostam. Portanto, a maioria das músicas que receberiam uma avaliação ruim não são avaliadas voluntariamente pelos usuários. Como as pessoas raramente ouvem músicas aleatórias ou assistem a filmes aleatórios, deveríamos esperar observar, em muitas áreas, uma diferença entre a distribuição de avaliações para itens aleatórios e a distribuição correspondente para os itens selecionados pelos usuários.

²Um efeito relacionado é que as classificações observadas provavelmente são especificadas por usuários que são avaliadores frequentes. Avaliadores frequentes podem mostrar padrões diferentes de valores de classificação em comparação a avaliadores pouco frequentes.

Esses fatores causam problemas de viés no processo de avaliação. Afinal, para realizar a avaliação em um determinado conjunto de dados, não se pode usar classificações realmente ausentes; em vez disso, é preciso simular itens ausentes com o uso de mecanismos de validação cruzada ou de retenção em classificações já especificadas. Portanto, os itens ausentes simulados podem não apresentar precisão semelhante à que se obteria teoricamente com os itens realmente consumidos no futuro.

Os itens que serão consumidos no futuro não serão selecionados aleatoriamente dentre as entradas ausentes pelos motivos discutidos acima. Essa propriedade das distribuições de classificação também é conhecida como Missing Not At Random (MNAR), ou viés de seleção [402, 565]. Essa propriedade pode levar a uma avaliação relativa incorreta dos algoritmos. Por exemplo, um modelo baseado em popularidade, no qual os itens com a classificação média mais alta são recomendados, pode ter um desempenho melhor em termos de geração de receita para o comerciante do que sua avaliação com base em classificações ausentes aleatoriamente pode sugerir. Esse problema é agravado pelo fato de que os itens na cauda longa são especialmente importantes para o sistema de recomendação, porque uma parcela desproporcional dos lucros em tais sistemas é realizada por meio desses itens.

Existem várias soluções para esse problema. A solução mais simples é não selecionar as classificações ausentes aleatoriamente, mas usar um modelo para selecionar as classificações de teste com base na probabilidade de serem classificadas no futuro. Outra solução é não dividir as classificações aleatoriamente entre treinamento e teste, mas dividi-las temporalmente, usando classificações mais recentes como parte dos dados de teste; de fato, o concurso Netflix Prize usou classificações mais recentes no conjunto de qualificação, embora algumas das classificações recentes também tenham sido fornecidas como parte do conjunto de sondagem. Uma abordagem que tem sido usada nos últimos anos é corrigir esse viés modelando o viés na distribuição de classificações ausente dentro das medidas de avaliação [565, 566]. Embora tal abordagem tenha alguns méritos, ela tem a desvantagem de que o próprio processo de avaliação agora pressupõe um modelo de como as classificações se comportam. Tal abordagem pode, inadvertidamente, favorecer algoritmos que usam um modelo semelhante ao usado para a previsão de classificações como para o processo de avaliação. É digno de nota que muitos algoritmos recentes [309] usam feedback implícito dentro do processo de previsão. Isso levanta a possibilidade de que um futuro algoritmo de previsão possa ser projetado para ser adaptado ao modelo usado para ajustar o efeito do viés de seleção do usuário dentro da avaliação. Embora as suposições em [565], que relacionam as classificações ausentes à sua relevância, sejam bastante razoáveis, a adição de mais suposições (ou complexidade) aos mecanismos de avaliação aumenta a possibilidade de "jogo" durante o benchmarking. No final das contas, é importante perceber que essas limitações na avaliação de filtragem colaborativa são inerentes; a qualidade de qualquer sistema de avaliação é fundamentalmente limitada pela qualidade da verdade básica disponível. Em geral, foi demonstrado por meio de experimentos com dados da Netflix [309] que o uso de medidas RMSE diretas nas classificações observadas geralmente se correlacionam muito bem com a precisão em todos os itens.

Outra fonte de viés de avaliação é o fato de que os interesses do usuário podem evoluir com o tempo. Como resultado, o desempenho em um conjunto de espera pode não refletir o desempenho futuro. Embora não seja uma solução perfeita, o uso de divisões temporais entre as classificações de treinamento e teste parece uma escolha razoável. Embora a divisão temporal resulte em testes de treinamento e teste com distribuições um tanto diferentes, ela também reflete o cenário do mundo real mais de perto. Nesse sentido, o concurso Netflix Prize novamente fornece um excelente modelo de design de avaliação realista. Diversas outras variações de métodos temporais no processo de avaliação são discutidas em [335].

7.6.1 Evitando jogos de avaliação

O fato de as classificações ausentes não serem aleatórias pode, às vezes, levar a uma manipulação não intencional (ou intencional) das avaliações em cenários onde os pares usuário-item das inscrições de teste são especificados. Por exemplo, no concurso Netflix Prize, as coordenadas dos pares usuário-item no conjunto de qualificação foram especificadas, embora os valores das classificações não tenham sido especificados. Ao incorporar as coordenadas dos pares usuário-item dentro do conjunto qualificador como feedback implícito (ou seja, matriz F na seção 3.6.4.6), pode-se melhorar a qualidade das recomendações. Pode-se argumentar que tal algoritmo teria uma vantagem injusta sobre um que não incluisse nenhuma informação sobre as identidades dos itens avaliados no conjunto qualificador. A razão é que, em cenários da vida real, nunca se teria qualquer informação sobre as coordenadas futuras dos itens avaliados, como as facilmente disponíveis no conjunto qualificador dos dados do Prêmio Netflix. Portanto, a vantagem adicional de incorporar tal feedback implícito desapareceria em cenários da vida real. Uma solução seria não especificar as coordenadas das entradas de teste e, assim, avaliar todas as entradas. No entanto, se a matriz de classificações tiver dimensões muito grandes (por exemplo, 107×105), pode ser impraticável realizar a previsão sobre todas as entradas. Além disso, seria difícil armazenar e carregar um número tão grande de previsões em um concurso online como o Prêmio Netflix. Nesses casos, uma alternativa seria incluir entradas não classificadas (espúrias) no conjunto de teste. Essas entradas não são usadas para avaliação, mas têm o efeito de impedir o uso das coordenadas das entradas de teste como feedback implícito.

7.7 Resumo

A avaliação de sistemas de recomendação é crucial para obter uma ideia clara sobre a qualidade de diferentes algoritmos. O método mais direto de medir a eficácia de um sistema de recomendação é calcular a taxa de conversão de itens recomendados em usos reais. Isso pode ser feito por meio de estudos com usuários ou estudos online.

Tais estudos são frequentemente difíceis para pesquisadores e profissionais devido à dificuldade de acesso à infraestrutura relevante com grandes grupos de usuários. Métodos offline têm a vantagem de poderem ser usados com múltiplos conjuntos de dados históricos. Nesses casos, é perigoso usar a precisão como único critério, pois maximizar a precisão nem sempre leva à maximização das taxas de conversão a longo prazo. Diversos critérios, como cobertura, novidade, serendipidade, estabilidade e escalabilidade, são utilizados para avaliar a eficácia dos sistemas de recomendação.

O design adequado dos sistemas de avaliação de recomendação é necessário para garantir que não haja vieses no processo de avaliação. Por exemplo, em uma aplicação de filtragem colaborativa, é importante garantir que todas as classificações sejam avaliadas com uma abordagem fora da amostra. Uma variedade de métodos, como hold-out e validação cruzada, são utilizados para garantir a avaliação fora da amostra. O erro é calculado com medidas como MAE, MSE e RMSE. Em algumas medidas, os itens são ponderados de forma diferente para levar em conta sua importância diferencial. Para avaliar a eficácia dos métodos de classificação, podem ser utilizadas a correlação de classificação, medidas baseadas em utilidade ou medidas baseadas em uso. Para medidas baseadas em uso, precisão e recall são utilizados para caracterizar o trade-off inerente à variação do tamanho da lista recomendada. A medida F1 também é utilizada, que é a média harmônica entre a precisão e o recall.

7.8 Notas Bibliográficas

Excelentes discussões sobre a avaliação de sistemas de recomendação podem ser encontradas em [246, 275, 538]. A avaliação pode ser realizada com estudos de usuários ou com conjuntos de dados históricos. Os primeiros trabalhos sobre avaliação com estudos de usuários podem ser encontrados em [339, 385, 433]. Um estudo anterior sobre avaliação de algoritmos de recomendação com conjuntos de dados históricos pode ser encontrado em [98]. Métricas para avaliar sistemas de recomendação na presença de inicialização a frio são discutidas em [533]. Experimentos controlados para avaliação online em aplicações Web são discutidos em [305]. Um estudo geral sobre o projeto de avaliação online é fornecido em [93]. A avaliação de sistemas multi-armed bandit é discutida em [349]. Uma comparação de sistemas de recomendação online em relação às decisões humanas é fornecida em [317].

O trabalho em [246] apresenta diversas variantes de métricas de precisão para avaliação. Este artigo é talvez uma das principais autoridades na avaliação de sistemas de recomendação.

As armadilhas do uso do RMSE como medida de avaliação são apresentadas em [632]. Uma breve nota técnica sobre os méritos relativos do uso do MAE e do RMSE como medidas de precisão pode ser encontrada em [141]. Os desafios e armadilhas no uso de métricas de precisão são discutidos em [418]. Métodos alternativos para avaliar sistemas de recomendação são fornecidos em [459].

Uma discussão sobre a importância da novidade é apresentada em [308]. Métodos online para medir a novidade de um sistema de recomendação são apresentados em [140, 286]. O uso da popularidade na medição da novidade é discutido em [140, 539, 680]. O trabalho em [670] mostrou que a serendipidade pode ser alcançada em um sistema de recomendação com a ajuda da rotulagem do usuário.

Métricas para avaliação de serendipidade são discutidas em [214, 450]. O trabalho em [214] também estuda o uso de métricas de cobertura. Métricas de diversidade são discutidas em [560]. O impacto dos sistemas de recomendação na diversidade de vendas é discutido em [203]. Métricas de robustez e estabilidade para sistemas de recomendação são discutidas em [158, 329, 393, 444]. Um estudo da avaliação de sistemas de classificação pode ser encontrado em [18, 22]. As discussões nesses livros fornecem uma compreensão das técnicas padrão usadas, como hold-out e validação cruzada.

Métodos de correlação de classificação são discutidos em [298, 299]. A medida de preferência de distância normalizada é discutida em [505]. O R-score para a avaliação de classificações baseada em utilidade é discutido em [98]. A medida NDCG é discutida em [59]. O índice de sustentação é discutido em [361], enquanto a taxa média de acerto recíproco (ARHR) é proposta em [181]. Uma discussão sobre curvas ROC no contexto de classificação pode ser encontrada em [195], embora as mesmas ideias também sejam aplicáveis ao caso de sistemas de recomendação. O uso de curvas ROC globais e específicas do cliente é discutido em [533].

Uma limitação dos sistemas de recomendação é que os valores nas classificações estão relacionados à sua frequência relativa e que os itens ausentes frequentemente estão na cauda longa. Portanto, o uso de mecanismos de validação cruzada ou de retenção leva a um viés de seleção contra itens menos frequentes. Vários métodos recentes para corrigir o viés de itens ausentes são discutidos em [402, 564–566]. A abordagem em [565] propõe o uso de diferentes suposições para os itens relevantes e não relevantes, em termos de decidir quais classificações estão ausentes. Um algoritmo de treinamento também é projetado em [565] com base nessas suposições. Uma estrutura temporal para avaliação realista é discutida em [335]. Os sistemas de recomendação também precisam ser avaliados de forma um pouco diferente em vários cenários, como na presença de contextos específicos. Esses contextos podem incluir tempo, localização ou informações sociais. Uma estrutura de avaliação para sistemas de recomendação no contexto de dados temporais é fornecida em [130]. Um workshop recente dedicado exclusivamente à avaliação de sistemas de recomendação pode ser encontrado em [4].

7.9 Exercícios

1. Suponha que um comerciante saiba o lucro q_i obtido na venda do item i . Projete uma métrica de erro para um sistema de filtragem colaborativa que pondere a importância de cada item com seu lucro.
2. Suponha que você projetou um algoritmo para filtragem colaborativa e descobriu que ele estava apresentando um desempenho ruim em avaliações com valor 5, mas estava apresentando um bom desempenho nas demais avaliações. Você usou essa informação para modificar seu algoritmo e testá-lo novamente. Discuta as armadilhas da segunda avaliação. Relacione sua resposta ao motivo pelo qual a Netflix optou por separar o conjunto de questionários e o conjunto de testes no conjunto de dados do Prêmio Netflix.
3. Implemente um algoritmo para construir o ROC e as curvas de precisão-recall.
4. Suponha que você tenha um conjunto de dados de feedback implícito em que as classificações são unárias. Uma curva ROC ou a métrica RMSE forneceriam resultados mais significativos?
5. Considere o usuário John, para quem você ocultou as avaliações de Aliens (5), Exterminador do Futuro (5), Nero (1) e Gladiador (6). Os valores entre parênteses representam suas avaliações ocultas, e valores mais altos são melhores. Agora, considere um cenário em que o sistema de recomendação classifica esses filmes na ordem: Exterminador do Futuro, Aliens, Gladiador e Nero.
 - (a) Calcule o coeficiente de correlação de postos de Spearman como uma medida da qualidade da classificação de recomendação. (b) Calcule o coeficiente de correlação de postos de Kendall como uma medida da qualidade da classificação de recomendação.
6. Para o problema do Exercício 5, a utilidade de John para um filme j é dada por $\max\{r_{ij} \geq 3, 0\}$, onde r_{ij} é sua classificação.
 - (a) Com base nessa suposição de utilidade, calcule o R-score específico para John. Assuma um valor de meia-vida de $\gamma = 1$.
 - (b) Para a mesma suposição de utilidade, calcule o componente do ganho cumulativo descontado (GDC) específico para John, se houver um total de 10 usuários no sistema.
7. Para o problema do Exercício 5, suponha que as únicas classificações ocultas pertençam a John, e as classificações previstas pelo sistema de recomendação sejam Aliens (4.3), Terminator (5.4), Nero (1.3) e Gladiador (5). Os valores entre parênteses representam as classificações previstas.
 - (a) Calcule o MSE das classificações previstas. (b) Calcule o MAE das classificações previstas. (c) Calcule o RMSE das classificações previstas. (d) Calcule o MAE e o RMSE normalizados, assumindo que todas as classificações estão no intervalo {1 ... 6}.