UNIVERSIDADE FEDERAL RURAL DO RIO DE JANEIRO
INSTITUTO MULTIDISCIPLINAR

RAÍZA BEATRIZ SILVA SANTANA

Particionamento Automático de Dados Utilizando Variáveis Latentes no Problema de Filtragem Colaborativa

Prof. Filipe Braida do Carmo, M.Sc.
Orientador

Nova Iguaçu, Janeiro de 2016

Particionamento Automático de Dados Utilizando Variáveis Latentes no Problema de Filtragem Colaborativa

Raíza Beatriz Silva Santana

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto Multidisciplinar da Universidade Federal Rural do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Apresentado por:

Raíza Beatriz Silva Santana

Aprovado por:

Prof. Filipe Braida do Carmo, M.Sc.

Prof. Carlos Eduardo Ribeiro de Mello, Ph.D.

Prof. Leandro Guimaraes Marques Alvim, D.Sc.

NOVA IGUAÇU, RJ - BRASIL

Janeiro de 2016

Agradecimentos

Quero agradecer, em primeiro lugar e com muito carinho, à minha mãe Dalva, que durante toda minha vida e principalmente durante a graduação me apoiou e incentivou.

Agradeço a todos com que eu convivi durante a graduação, aprendi algo com cada pessoa e experiência. Quero agradecer principalmente aos meus amigos Válber e Jefferson, que sempre me apoiaram dentro e fora da faculdade. Agradeço também ao meu orientador Filipe Braida, professor que eu respeito e admiro, e que teve paciência e fé em mim durante o projeto final.

Agradeço com muito carinho ao Hebert, que com toda paciência e amor, me auxiliou e apoiou nessa jornada.

RESUMO

Particionamento Automático de Dados Utilizando Variáveis Latentes no Problema de Filtragem Colaborativa

Raíza Beatriz Silva Santana

Janeiro/2016

Orientador: Filipe Braida do Carmo, M.Sc.

A filtragem colaborativa é uma das abordagens mais conhecidas em sistemas de recomendação, ela faz uso da matriz de avaliações para a tarefa de prever as notas do usuários sobre os itens, e a partir da similaridade entre os usuários ou itens realiza a sugestão. Todavia, essas informações geralmente são esparsas e como, atualmente, as bases dos sistemas de recomendação tem dimensões entre milhares e milhões de usuários e itens, a execução do algoritmo pode se tornar impraticável. Este trabalho baseia-se no particionamento da base de dados, representando as entidades da recomendação a partir da extração dos fatores latentes dos dados originais, e na utilização da matriz de avaliações para eliminar a necessidade de conhecimento *a priori* sobre o domínio do sistema. Foram propostas algumas técnicas para o particionamento da base de dados e estas foram comparadas entre si, sendo avaliadas sob um conjunto de dados reais. O trabalho propôs o particionamento dos dados e a recomendação de filmes a partir disso, para avaliar como seria a qualidade das recomendações nesse agrupamento. Foi mostrado que os fatores latentes são efetivos e representam as entidades dos sistemas de recomendações, podendo ser usados como base para os algoritmos da área.

ABSTRACT

Particionamento Automático de Dados Utilizando Variáveis Latentes no Problema de Filtragem Colaborativa

Raíza Beatriz Silva Santana

Janeiro/2016

Advisor: Filipe Braida do Carmo, M.Sc.

Collaborative filtering is one of the most popular approaches in recommender systems, it makes use of the array of reviews for the task of predicting the user's notes about the items, and from the similarity between users or items carries the suggestion.

This work is based on the partitioning of the database, representing the entities of the recommendation from the extraction of latent factors of the original data and the use of ratings matrix to eliminate the need for knowledge a priori on the field of the system. have been proposed some techniques for partitioning the database and these were compared and evaluated under a real data set. have been proposed some techniques for partitioning the database and these were compared and evaluated under a real data set. The work proposed the partitioning of data and recommending films from that, to assess what would be the quality of the recommendations in this group. It was shown that the latent factors are effective and represent entities of recommendation systems, can be used as the basis for algorithms area.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

SR	Sistema de Recomendação
FC	Filtragem Colaborativa
kNN	*k-Nearest Neighbors* (k-vizinhos mais próximos)

Sumário

Capítulo 1

Introdução

1.1 Motivação

O volume de informação produzido no mundo tem aumentado progressivamente, numa velocidade maior do que nossa capacidade de assimilá-lo (Turner, 2014). Somente em 2015, aproximadamente 2 milhões e meio de livros foram publicados[1]; a *Wikipedia*[2], enciclopédia com livre acesso pela *web*, dispõe no momento atual[3] de mais de cinco milhões de artigos escritos em inglês.

O avanço das tecnologias de informação e comunicação (TIC) popularizou o acesso a todo esse conhecimento, ampliando as opções de conteúdos, produtos e serviços disponíveis. Em virtude dessa facilidade, encontrar conteúdo relevante tornou-se uma atividade exaustiva, assim como o hábito de valer-se de sugestão de conhecidos, reportagens, artigos, guias de viagem, e outros.

Um sistema de recomendação desponta como facilitador nessa busca por informação útil, já que filtrar toda a *web* ou procurar um livro interessante em uma loja *online*, por exemplo, são tarefas por vezes impraticáveis. Em (Kantor, 2011) um sistema de recomendação (SR) é definido como um conjunto de técnicas e ferramentas de *software* que fornecem sugestões de itens que são úteis para um usuário . Existem

[1] http://www.worldometers.info/books/

[2] www.wikipedia.org

[3] 25 de janeiro de 2016

diversas maneiras para se efetuar a recomendação, utilizar informações do produto ou serviço a ser recomendado (recomendação baseada em conteúdo) (Adomavicius, 2005) é uma delas.

Essa área surgiu nos anos 90 e têm sido largamente utilizada por empresas como *Netflix*[4], *E-bay*[5], *Google*[6] e outras grandes varejistas no comércio eletrônico (Schafer, 1999). Na *Netflix*, por exemplo, a recomendação é a base da fidelização dos seus clientes onde a indicação de novos conteúdos, de forma personalizada, é uma forma de garantir a lealdade dos consumidores.

Todavia, existem algumas limitações na aplicação do algoritmo de recomendação, alguns deles são baseados nas avaliações dos itens pelos usuários, analisando os itens avaliados em comum entre usuários ou usuários comuns entre os itens. Geralmente, essas avaliações são poucas quando comparadas à quantidade de usuários e itens dentro do sistema (Sarwar, 2002). Esse problema se extende para os novos usuários ou itens, pois até eles possuírem um número mínimo de avaliações, não é possível considerá-los na recomendação (Papagelis, 2005).

Uma outra consideração importante é que as informações sobre os itens e/ou usuários não estão sempre disponíveis nos sitemas de recomendação (Koren, 2009), o que impossibilita a execução dos algoritmos dependentes dessas informações.

Esse trabalho tem como objetivo propor uma metodologia que identifique a similaridades nas entidades de recomendação de forma automática, sem conhecimento *a priori* do sistema, para, então, utilizar as técnicas de filtragem colaborativa na obtenção de recomendações independentes do domínio.

[4]www.netflix.com
[5]www.ebay.com
[6]www.google.com

1.2 Objetivos

Os objetivos desse trabalho são:

- Propor uma metodologia que encontre e agrupe itens similares de maneira automática e baseado apenas na matriz de avaliação;
- Aplicar os algoritmos de filtragem colaborativa a esse agrupamento;
- Comparar os desempenhos encontrados utilizando algumas variações entre os experimentos.

1.3 Organização

Esse trabalho se estrutura da seguinte forma:

O capítulo 2 discute conceitos referentes à fundamentação teórica do trabalho. São abordados os conceitos mais importantes da área de sistemas de recomendação.

O capítulo 3 apresenta a proposta do trabalho, que é realizar a recomendação independentemente do domíno do sistema, além de mostrar trabalhos relacionados na mesma área de pesquisa.

O capítulo 4 formaliza a metodologia para os experimentos, expondo qual a base de dados utilizada, as formas de avaliação, os experimentos realizados e traz uma análise dos resultados obtidos.

O capítulo 5 expõe as conclusões, limitações, possíveis trabalhos fututos.

Por fim, são apresentadas as referências bibiográficas deste trabalho.

Capítulo 2

Sistemas de Recomendação

A massa de informação disponível na *Internet*, juntamente com a sua natureza dinâmica e heterogênea, levou a um aumento na dificuldade de se encontrar conteúdo que seja relevante. Para (Bawden, 2001), a sobrecarga informacional - informação disponível e potencialmente útil - acaba desencadeando um obstáculo para o indivíduo que necessita dela, pois se apresenta de forma não estruturada e volumosa. Já (TERRA, 2003) diz que o excesso de informação está associado à perda de controle sobre ela e à inabilidade em usá-la efetivamente, resultando em ineficiência no trabalho e até em riscos para a saúde.

Portanto, o acesso à informação personalizada se tornou crucial: usuários precisam de uma estrutura customizada na hora de filtrar grandes quantidades de informação de acordo com seus interesses e preferências.

Os sistemas de recomendação surgem a partir dessa necessidade, auxiliando os usuários a encontrar produtos e serviços relacionados ao seu perfil analisando as experiências anteriores do próprio usuário ou de outros usuários semelhantes a ele, evitando assim, uma recomendação genérica que indica os itens mais consumidos ou, ainda, indicando um item pouco consumido, mas que se encaixa bem no perfil de determinado usuário e, no entanto, têm pouca visibilidade dentro do sistema.

2.1 Fundamentação Teórica

Podemos definir um sistema de recomendação (SR) como um conjunto de técnicas e ferramentas de software que fornecem sugestões de itens interessantes para um usuário (Mahmood, 2009).

O início do desenvolvimento na área se deu na década de 90, com a criação do *Tapestry* (Goldberg, 1992), que consistia num sistema de recomendação em listas de *e-mails* que permitia aos usuários avaliar as mensagens recebidas expressando sua opinião, possibilitando, assim, que os usuários filtrassem mensagens considerando não apenas seu conteúdo, mas também opiniões de outras pessoas sobre a mensagem. Este projeto lançou o conceito de filtragem colaborativa, já que nele a informação era filtrada levando-se em consideração a relação do usuário-alvo com os usuários semelhantes a ele.

Atualmente, a recomendação é peça chave em diversas empresas como *Amazon*[1], Youtube[2], *Netflix*[3] e *Ebay*[4], por exemplo. Em (Linden, 2003), é demonstrado que SR trazem três principais benefícios para o comércio eletrônico: o aumento das vendas, vendas cruzadas e uma maior lealdade dos seus consumidores.

Em (Adomavicius, 2005), encontramos uma descrição formal do problema de recomendação como: Seja C o conjunto de todos os usuários e S o conjunto de todos os itens (filmes, músicas, artigos) que podem ser recomendados. O conjunto S pode se aproximar da casa das centenas ou milhares de itens em algumas aplicações, assim como o conjunto dos usuários, que pode ser de milhões, em alguns casos. Seja u a função utilidade que informa a importância do item s ao usuário c, *i.e.*, $u : CxS \rightarrow R$, sendo R um conjunto ordenado. Então, para cada usuário $c \in C$ queremos escolher um item $s' \in S$ que maximize a função utilidade:

$$\forall\ c \in C, s'c = argmax\ u(c, s)$$

[1] www.amazon.com

[2] www.youtube.com

[3] www.netflix.com

[4] www.ebay.com

	Matrix	Eu, Robô	A procura da felicidade	O jogo da imitação
Hebert	5	⊘	⊘	5
Jefferson	3	3	4	⊘
Válber	⊘	1	⊘	⊘

Tabela 2.1: Fragmento da matriz de notas de um sistema de recomendação de filmes

Na maioria dos sistemas de recomendação, a função utilidade é representada por uma nota que indica a relevância de um item para um usuário e pode variar de acordo com o sistema. Temos, como exemplo, a Tabela 2.1, que simula as avaliações dos usuários em relação aos filmes, onde as notas variam de 1 a 5 e para os itens que o usuário não explicitou a preferência, é utilizado o símbolo "⊘". O problema principal de um SR é que a função utilidade não está, necessariamente, definida para todo o conjunto CxS, mas somente para um subconjunto deste (Adomavicius, 2005).

Portanto, o propósito do sistema de recomendação é utilizar técnicas para estimar os valores não explicitados pelos usuários e, assim, oferecer boas recomendações (Adomavicius, 2005). Essas técnicas podem se basear no conteúdo dos itens ou nos usuários, associando interesses de um usuário aos atributos de um item; ou podem se apoiar nas notas explicitadas pelos usuários para filtrar os itens mais relevantes para um determinado usuário similar ao grupo.

A estimativa desses valores não explicitados é utilizada para prever quão relevante um item será para um determinado usuário e gera uma nota estimada de um item i para um usuário u, caracterizando assim a predição (Sarwar, 2001). Após esse processo, os n itens mais significativos para o usuário u, ou seja, os que possuem as maiores notas, são indicados a ele, caractarizando assim a recomendação. É interessante que essa lista não contenha itens já avaliados pelo usuário u.

A preferência de um usuário por um item que não está especificado pode ser estimada de duas maneiras. A primeira utiliza heurísticas que definem a função utilidade u e empiricamente validam sua performance. A segunda, estima a função utilidade u que minimiza algum critério em especial, como a raiz quadrada da média

do erro (RMSE) (Adomavicius, 2005).

Dado que os itens não avaliados tenham seus valores (notas) calculados, utilizando alguma das abordagens citadas anteriormente, o sistema de recomendação selecionará o item que possuir o maior valor entre os estimados, representando maior relevância para o usuário. Ou, como segunda opção, é aceitável recomendar o conjunto de n itens de maior valor estimado para um usuário.

Inúmeras soluções de SR foram desenvolvidas e podemos classificá-las de acordo com sua abordagem e o conteúdo utilizados. Em (Burke, 2007), foi proposta uma sistemática de cinco diferentes classes, e o trabalho foi estendido por (Ricci, 2011) adicionando uma nova classe, como mostrado a seguir:

- Baseada em conteúdo: é realizada uma seleção baseada na análise do conteúdo dos itens e nos perfis dos usuários, onde somente os itens similares ao conjunto de preferências do usuário são recomendados;
- Filtragem colaborativa: nesta abordagem, os itens a serem recomendados são filtrados com base nas avaliações feitas por usuários similares, ou seja, com gostos e preferências parecidas ao usuário alvo;
- Demográfica: as informações demográficas do usuário são usadas para realizar a recomendação;
- Baseada em conhecimento: sistemas baseados em conhecimento recomendam itens que possuem atributos que vão de encontro às necessidades e preferências do usuário, de acordo com o conhecimento do domínio;
- Baseada em comunidade: este tipo se baseia nas preferências dos usuários relacionados. A ideia tem como premissa a inclinação das pessoas a dar mais importância à recomendação de amigos do que de pessoas desconhecidas porém com gostos similares;
- Híbrida: esta abordagem consiste na mistura das técnicas acima. O recomendador híbrido junta as abordagens A e B, aproveita suas vantagens e corrige as desvantagens de cada uma.

A seguir será apresentada a técnica de Filtragem Colaborativa, assim como seus algoritmos e problemas mais relevantes. Essa abordagem foi escolhida por ser uma das principais e mais bem conceituadas na área de sistemas de recomendação como mostrado em (Massa, 2004; Herlocker, 2004; Schafer, 2007; Sarwar, 2002; Schafer, 1999), e por isso, será o foco deste trabalho.

2.2 Filtragem Colaborativa

A essência dos sistemas baseados em filtragem colaborativa (*Collaborative Filtering - CF*) está na troca de experiências entre pessoas que possuem interesses comuns. Essa abordagem se aproxima da ideia de considerarmos a opinião de pessoas com gostos coincidentes aos nossos na hora de escolher um item, como um filme ou livro, por exemplo.

Formalmente, podemos dizer que uma função utilidade $u(c, s)$ de um um item s para um usuário c é estimada com base no valor da função utilidade $u(c_j, s)$ para o mesmo item s e para os usuários $c_j \in C$ que são similares ao usuário c (Adomavicius, 2005). O conjunto $C \times S$, que representa a relação dos itens e usuários, é apresentada em forma de matriz. O principal ponto desta abordagem é identificar os usuários c_j que possuem o mesmo perfil de comportamento que o usuário c e explicitaram a nota para o item s. Supondo que o usuário c tenda para o mesmo comportamento dos usuários c_j similares a este, podemos obter a sua nota para o item s a partir das notas dadas por c_j. Por suas características a filtragem colaborativa também é conhecida na literatura como abordagem social.

A filtragem colaborativa tem algumas vantagens sobre a baseada em conteúdo, que também é bem popular na área. Nela não é necessária a compreensão ou reconhecimento sobre o conteúdo dos itens. Já na abordagem baseada em conteúdo existe a necessidade de conhecimento sobre o item, e a recomendação está diretamente relacionada com a qualidade deste (Linden, 2003). Outra vantagem é a possibilidade de apresentar aos usuários recomendações inesperadas que não estavam sendo pesquisadas de forma ativa, e com isso reprimir o problema de recomendar "mais

do mesmo", aumentando as recomendações *serendipity* (Ge, 2010). Uma outra contribuição significativa dos SR de filtragem colaborativa se refere à possibilidade de formação de comunidades de usuários pela identificação de seus gostos e interestes similares.

Todavia, existem diversos desafios quando se trabalha com CF. Em geral, os sistemas de recomendação possuem uma base grande de produtos e a quantidade de dados total do sistema quando comparados à matriz usuário-item completa é cerca de 1% (Sarwar, 2001). Por conseguinte, a matriz usuários-itens se torna extremamente esparsa, o que reduz a qualidade e o desempenho da recomendação, já que estes estão diretamente relacionados ao tamanho e a esparsidade dos dados (Linden, 2003).

Outro problema oriundo da esparsidade é o surgimento de um item ou usuário novo. Dado a metodologia da filtragem colaborativa, um usuário ou item novo numa base esparsa dificulta a tarefa de encontrar o conjunto de usuários similares. Como consequência, o item tem pouca visibilidade até que uma certa quantidade de usuários o recomende e um usuário fica impossibilitado de receber recomendações até que ele avalie um certo número de itens. Este problema é conhecido na literatura como *cold-starter* (Schein, 2002).

A escalabilidade também é um fator considerável. Nos algoritmos de vizinhos mais próximos, por exemplo, os cálculos crescem conforme o número de itens e usuários, exigindo mais recursos computacionais. Com milhões de usuários e itens, um típico SR baseado em *web* executando os algoritmos existentes sofrerá sérios problemas de escalabilidade.

Diversas abordagens foram criadas para minimizar os problemas descritos acima. Grande parte destas usa técnicas de redução de dimensionalidade como a decomposição de valores singulares (SVD) ou análise das principais componentes (PCA) para lidar com a esparsidade e ainda assim prover boas recomendações (Billsus, 1998; Pelikan, 2001). A literatura divide os algoritmos de filtragem colaborativa em duas grandes classes: baseados em memória e baseados em modelo (Adomavicius, 2005), que serão detalhados a seguir.

2.2.0.1 Memória

Algoritmos baseados em memória realizam a recomendação com base em todos os itens já avaliados pelos usuários. Isto é, a avaliação desconhecida de um item $u_{c,s}$ para um usuário c e um item s é normalmente calculada agregando-se as preferências de alguns usuários (os n mais similares) para o mesmo item s (Adomavicius, 2005). Em casos mais simples, a agregação pode ser uma simples média entre as notas, no entanto a agregação mais comum é a média ponderada.

A similaridade entre dois usuários é, basicamente, a distância medida entre eles e é usada como peso na média ponderada, ou seja, as avaliações dos usuários mais próximos têm um peso maior do que a de usuários mais distantes. Um fato importante sobre a média ponderada é que esta técnica não considera o fato de que diferentes usuários podem usar a escala de avaliação de formas diferentes. Por exemplo, se temos uma escala de 1 a 5 para avaliação de filmes, pode ocorrer de dois usuários que gostaram realmente do filme darem notas diferentes para o mesmo. Dado que são usuários diferentes, podem ter formas diferentes de medir a escala de notas. A média ponderada ajustada tem sido amplamente utilizada para resolver esse problema. Nesta aproximação, em vez de usar o valor absoluto das avaliações, é usado o desvio padrão da média das avaliações de um determinado usuário.

É possível separar os usuários em níveis de similaridade através dessas técnicas, definindo, por exemplo, o conjunto de n usuários mais próximos de cada usuário alvo, simplificando o processo de recomendação.

Várias estratégias podem ser utilizadas para calcular a similaridade entre dois usuários ou itens nos sistemas de recomendação baseados em filtragem colaborativa. Na maioria dos casos, a semelhança entre dois usuário é baseada nos itens que os dois avaliaram. As duas aproximações mais populares são correlação (*correlation*) e baseada em cosseno (*cosine-based*) (Adomavicius, 2005).

A similaridade baseada na correlação – entre dois usuários u e v ou entre dois itens i e j – é medida calculando-se a correlação de *Pearson* ou outras similaridades baseadas em correlações. A correlação de *Pearson* mede a correlação estatística

(Pearson's r) entre duas variáveis linearmente dependentes (Resnick, 1994). A correlação de Pearson entre os usuários u e v é dada pela seguinte fórmula:

$$w_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I}(r_{v,i} - \bar{r}_v)^2}} \tag{2.1}$$

onde o somatório de $i \in I$ se refere aos itens que tanto o usuário u quanto o usuário v avaliaram e r_u é a avaliação média dos itens co-avaliados do usuário u. Para a correlação entre itens, a fórmula sofre algumas alterações:

$$w_{i,j} = \frac{\sum_{u \in U}(r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U}(r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_{u \in U}(r_{u,j} - \bar{r}_j)^2}} \tag{2.2}$$

onde o conjunto $u \in U$ representa os usuários que avaliaram ambos os itens i e j, $r_{u,i}$ é a avaliação de um usuário u a um item i e $\bar{r}_i$ é a média das avaliações do item i pelos outros usuários, conforme pode ser visto na Figura 2.1. Os algoritmos baseados na correlação de *Pearson* compõe uma parte representativa dos algoritmos de CF e são amplamente usados em pesquisas nessa área (Su, 2009).

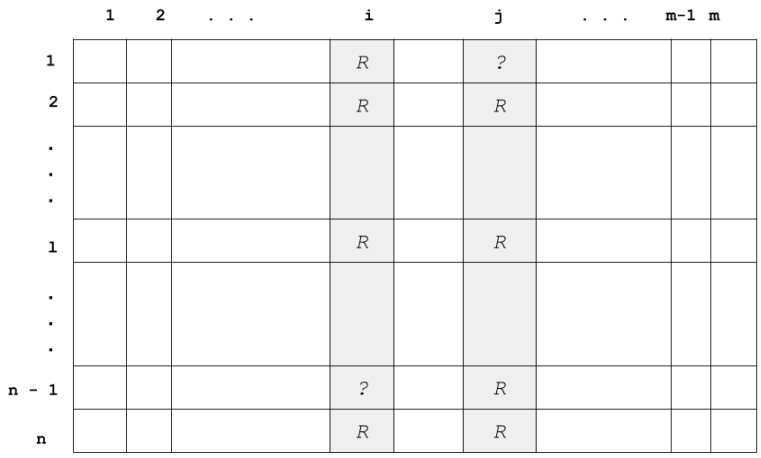


Figura 2.1: Similaridade entre itens ($w_{i,j}$) calculada com base nos itens avaliados pelos usuários 2, l e n.

Na similaridade baseada no cosseno, dois usuários u e v são tratados como vetores

(x_u, x_v) num espaço m-dimensional. Logo, a similaridade entre estes dois vetores é dada pelo cosseno do ângulo formado entre eles (Adomavicius, 2005).

$$cos(x_u, x_v) = \frac{x'_u x_v}{||x_u||||x_v||} \tag{2.3}$$

esse valor pode ser utilizado para expressar a similaridade entre dois usuários ou dois itens onde o espaço é formado pelas notas co-avaliadas entre dois objetos x_u e x_v, isto é, $S_{uv} = \{s \in S | r_{u,s} \neq \oslash \& r_{v,s} \neq \oslash\}$, onde S_{uv} é o conjunto de intercepção entre os dois e $sim(u, v)$ varia de [-1,1]. Por exemplo, para os vetores $\vec{A} = \{x_1, y_1\}$ e $\vec{B} = \{x_2, y_2\}$, o vetor de similaridade do cosseno é:

$$cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{||\vec{A}|| * ||\vec{B}||} = \frac{x_1 x_2 + y_1 y_2}{\sqrt{x_1^2 + y_1^2}\sqrt{x_2^2 + y_2^2}} \tag{2.4}$$

Nessa aproximação, a situação onde diferentes usuários usam a escala de avaliação de forma diferente não é tratada. Para ajustar esse valor, a média de notas do usuário é subtraída de cada par de itens co-avaliados. A similaridade baseada no cosseno ajustada tem a mesma fórmula que a correlação de *Pearson*. De fato, a correlação de *Pearson* executa a similaridade baseada no cosseno com uma normalização das notas do usuário de acordo com seu comportamento (Adomavicius, 2005).

2.2.0.2 Modelo

Diferente dos algoritmos baseados em memória, que utilizam diretamente as notas armazenadas para realizar a predição de uma nota por um usuário u para um item i, os algoritmos baseados em modelo desenvolvem um modelo matemático para realizar a previsão dos itens. Nessa categoria, os algoritmos fazem aproximação probabilística e supervisionam o processo calculando o valor esperado de uma nota, dado as notas reais deste usuário para outros itens (Adomavicius, 2005).

Técnicas de aprendizado de máquina e mineração de dados são utilizadas na construção dos modelos a fim de permitir que o sistema aprenda a reconhecer padrões complexos com base em dados de treino e, então, realizar predições inteligentes para os dados reais dos sistemas de recomendação (Su, 2009). Geralmente, os algoritmos de classificação podem ser usados como modelos de CF se as notas são categorizadas, ou, modelos de regressão e SVD para notas numéricas.

Em (Breese, 1998), é proposto um modelo probabilístico para calcular as notas não explícitadas pelos usuários:

$$r_{c,s} = E(r_{c,s}) = \sum_{i=0}^{n} r \times Pr(r_{c,s} = i | r_{c,s'}, s' \in S_c) \tag{2.5}$$

Assumindo que a nota é uma variável aleatória inteira que varia entre 0 e n, então, a expressão tem como objetivo calcular as probabilidades condicionais de um usuário c dar uma nota para um item s e, com isso, a nota prevista $r_{c,s}$ será a ponderação das probabilidades com o valor absoluto da nota. Para calcular essa probabilidade, (Breese, 1998) propõe duas alternativas: o modelo de agrupamento e redes Bayesianas. No modelo bayesiano, usuários com o mesmo padrão de notas são agrupados em classes (Chen, 1999). Os parâmetros do modelo são estimados utilizando uma Cadeia de Markov.

Uma limitação dessa abordagem é que cada usuário pode acabar sendo alocado em um único grupo, enquanto algumas aplicações podem se beneficiar da habilidade de alocar usuários em várias categorias ao mesmo tempo. Por exemplo, em uma aplicação de recomendação de livros, um usuário pode estar interessado pelo tópico "Programação"enquanto está no trabalho mas para lazer pode se interestar por um assunto totalmente diferente, como "Pesca"(Adomavicius, 2005).

Outra técnica existente são os algoritmos baseados em modelos de regressão (Su, 2009), que tentam aproximar a nota com base nesses modelos. Seja $X = (X_1, X_2, ..., X_n)$ a variável aleatória que representa as notas de um usuário sobre os itens, seja Υ uma matriz $n \times k$, sendo $R = (R_1, R_2, ..., R_n)$ a variável aleatória que representa o ruído das escolhas dos usuários e N é $i \times j$ onde $n_{u,i}$ é a nota do usuário u para o item i. O modelo de regressão linear pode ser representado por:

$$N = \Upsilon X + R \tag{2.6}$$

Existem, na literatura, diversas soluções para preencher os valores faltantes da matriz N, isto é, realizar a predição das notas faltantes. Em (Canny, 2002), é proposta a utilização de algoritmos de *Expectation Maximization* (EM) (Dempster, 1977) para preencher esses valores.

Em (Funk, 2015), foi proposta uma simplificação do modelo de regressão e a utilização da técnica de Decomposição em Variáveis Singulares (SVD) para a construção do modelo. Essa proposta incentivou o surgimento de uma classe de algoritmos baseados em modelo que utiliza o conceito de fatores latentes (Berry, 1995) para a construção do modelo de recomendação.

Os fatores latentes tentam descrever um objeto em fatores, prezando pelos que mais agregam informação ao mesmo. No caso dos itens, os fatores latentes podem definir o quanto ele corresponde a uma determinada categoria, por exemplo, o quanto um filme se encaixa no gênero ação ou comédia. Já para o usuário, elas podem definir qual característica de um item faz a sua avaliação ser melhor ou pior (Funk, 2015). Exemplificando, o filme Matrix pode ser descrito como (ação=1.5; ficção-científica=2), considerando uma escala de 0 a 2, e o usuário Hebert é descrito como (ação=1.2; ficção-científica=1.5). Combinando os dois, é possível descobrir que Hebert gostará do filme Matrix com $1.5 \times 1.2 + 2 \times 1.5 = 4.8$, considerando uma escala de 1 a 5.

Uma outra forma de decompor a matriz em fatores latentes é a Análise dos Componentes Principais *(PCA — Principal Component Analysis)* visto em (Kim, 2005; Gupta, 1999; Goldberg, 2001). Este método é associado à idéia de redução de massa de dados, com menor perda possível da informação (Varella, 2008). O *PCA* é uma técnica estatística que transforma um conjunto de variáveis em outro de mesma dimensão. No conjunto final, cada variável é independente em relação às outras e o método de extração faz com que estas retenham o máximo de informação em relação à variação contida nos dados originais.

O conjunto final é ordenado de acordo com a variação que representa nos dados, ou seja, a variação capturada pelo primeiro componente é maior que a quantidade de variação no segundo, e assim por diante. Dessa forma, a dimensão é reduzida ao desconsiderar os componentes que pouco contribuem para a variação (Dodorico, 2014).

O método pode ser utilizado no agrupamento de indivíduos de acordo com sua variação, ou seja, seu comportamento dentro da população. Os indivíduos são agrupados dada a similaridade das suas variâncias (Varella, 2008). Isso pode ser aplicado à matriz usuário-item, analisando o padrão de avaliação dos usuários e agrupando os mais similares, ou então, aplicando essa análise aos itens.

Uma limitação do PCA é que ele supõe que os dados são uma combinação linear e que foram elaborados a partir de uma distribuição estatística Gaussiana. Caso essas suposições não sejam verdadeiras, não há garantias do funcionamento efetivo da técnica (Dodorico, 2014).

No trabalho é utilizada a técnica de Decomposição em Valores Singulares para iniciar o modelo onde a matriz de notas $m \times n$ é decomposta em PSQ^T, sendo que P é $m \times n$, Q é $n \times n$ e S é a matriz singular $m \times n$. Com essa fatoração, para cada usuário c existirá um vetor $p_c \in R^k$ e para cada item s existirá um vetor $q_s \in R^k$ correspondente aos fatores latentes de cada um. Assim, a nota de um usuário para um item pode ser aproximada com a multiplicação dos fatores de cada um, como exemplificado anteriormente:

$$r_{c,s} = q_s^t p_c \tag{2.7}$$

É possível manter k colunas das matrizes P, S e Q para obter uma matriz original aproximada, como exemplificado abaixo, na Figura 2.2.

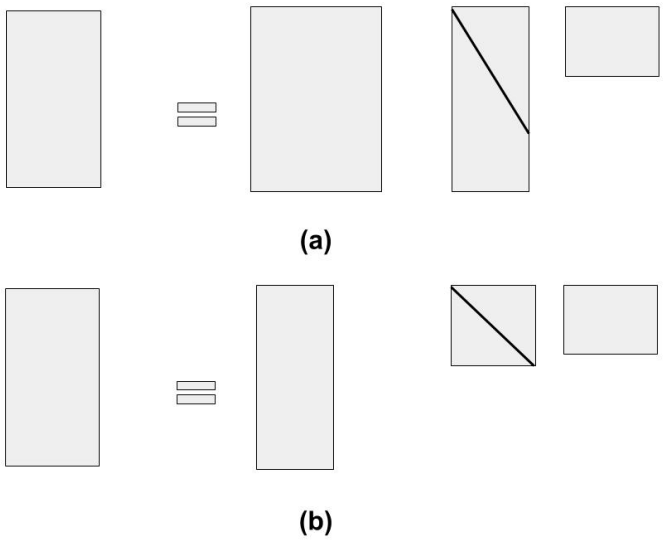


Figura 2.2: (a) - Decomposição SVD. (b) - Decomposição aproximada

A esparsidade da matriz de notas é o maior desafio na realização da fatoração (Adomavicius, 2005). A solução proposta por (Funk, 2015) utilizou a realização de um processo de aprendizado após a decomposição. Com o objetivo de minimizar o erro quadrático da soma, o algoritmo gradiente descendente foi usado para garantir a convergência do modelo. O trabalho propõe a utilização da constante com o valor de 0.02 para regularizar a função.

$$min(q_*, p_*) \sum_{(c,s)\in X} (r_{cs} - q_s^T p_c)^2 + \lambda(||q_s||^2 + ||p_c||^2) \tag{2.8}$$

O algoritmo do gradiente descendente irá analisar todas as avaliações $r_{c,s}$ dos dados de treino e para cada uma irá calcular o erro em relação à nota prevista $r'_{c,s}$. A cada avaliação dos dados de treino, os parâmetros q_s e p_c serão corigidos com relação ao erro $e_{c,s}$. No trabalho foi proposta a utilização da constante γ com intuito de controlar a taxa de aprendizado e o valor definido para γ foi 0.001.

$$e_{cs} = rcs - q_s^T p_c \tag{2.9}$$

$$q_s \leftarrow q_s + \gamma \times (e_{cs} p_c - \lambda q_s) \tag{2.10}$$

$$p_c \leftarrow p_c + \gamma \times (e_{cs} q_s - \lambda p_c) \tag{2.11}$$

Em muitos SR é possível notar que um item e um usuário possuem uma tendência (*bias*) independente de qualquer iteração, ou seja, um item que possui uma nota alta tenderá a manter essa nota para os demais usuários. Portanto, o objetivo não é expressar a nota como uma simples interação dos fatores latentes do usuário e do item, como visto na equação 2.7, pois parte da nota é tendência do usuário e do item sobre a média global naquele sistema de recomendação e a outra parte é dada pelos fatores latentes. Pode-se afirmar, então, que uma nota r_{cs} detém uma tendência b associada tanto ao item quanto ao usuário em relação à média global α.

$$b_{cs} = \alpha + b_c + b_s \tag{2.12}$$

Aplicando a tendência ao cálculo da nota baseado na decomposição de valores singulares temos:

$$r_{cs} = \alpha + b_c + b_s + q_s^t p_c \tag{2.13}$$

Acrescentando os novos parêmatros (b_c, b_s) ao modelo de aprendizado temos:

$$min(b_*, q_*, p_*) \sum_{(c,s) \in X} (r_{cs} - \alpha - b_c - b_s - q_s^T p_c)^2 + \lambda(\alpha + b_c + b_s + ||q_s||^2 + ||p_c||^2) \tag{2.14}$$

Adotando o conceito de tendência no método proposto por Funk em (Funk, 2015), que utiliza o gradiente descendente como função objetivo, temos:

$$b_s \leftarrow b_s + \gamma \times (e_{cs} p_c - \lambda b_s) \tag{2.15}$$

$$b_c \leftarrow b_c + \gamma \times (e_{cs} p_c - \lambda b_c) \tag{2.16}$$

$$q_s \leftarrow q_s + \gamma \times (e_{cs} p_c - \lambda q_s) \tag{2.17}$$

$$p_c \leftarrow p_c + \gamma \times (e_{cs} q_s - \lambda p_c) \tag{2.18}$$

Algumas extensões desse modelo foram propostas na literatura (Paterek, 2007; Ricci, 2011). Em todas elas, estende-se o modelo adicionando novos parâmetros ou criando novos conceitos, como tempo ou contexto, e com isso adicionam mais informação ao modelo para que as previsões sejam mais assertivas.

Capítulo 3

Proposta

Os sistemas de recomendação têm como objetivo principal resolver o problema de encontrar itens relevantes para um usuário (Adomavicius, 2005). Podemos citar como exemplo a *Amazon*[1], uma empresa norte americana de comércio eletrônico que desenvolveu seu próprio sistema de recomendação, personalizando totalmente a experiência de compra para cada consumidor. A empresa sugere, por exemplo, livros de programação a um engenheiro de *software* e brinquedos a uma mãe recente (Linden, 2003).

O problema de recomendação pode ter inúmeras configurações e existem diversas técnicas para resolvê-lo (Kantor, 2011), no entanto, a maioria delas é extremamente custosa. Na filtragem colaborativa, por exemplo, a quantidade de análises a ser realizada é da ordem de $O(MN)$ no pior caso, onde M é o número de usuários e N é o número de itens, já que é necessário analisar N itens para cada um dos M usuários. Como citado em (Sarwar, 2001) a matriz usuários-itens geralmente é esparsa. Isso faz com que o algoritmo tenda a realizar $(M + N)$ análises. O que em bases grandes – com milhões de itens e usuários – continua sendo problemático em relação à performance e a escala (Linden, 2003).

É possível ilustrar o problema de lidar com uma grande quantidade de dados com a seguinte situação hipotética, que na verdade é bem comum em lojas virtuais

[1] www.amazon.com

de médio a grande porte. Imagine uma loja de livros que possui 10.000 títulos diferentes à venda e que, nessa loja, existam dados de venda desses livros para 1.000 clientes. Então, temos uma matriz esparsa de 1.000 × 10.000 para examinar. Se os livros forem descritos com 10 atributos como gênero, data de lançamento, número de páginas, entre outros, essa matriz terá 10 dimensões, o que fará com que a quantidade de cálculos a serem feitos seja 1.000 × 10.000 elevado a 10.

A situação descrita acima, que é o exemplo de um caso simples, mostra como o processo de recomendação pode ser custoso. Para o caso da *Amazon* [2], onde o número de clientes e de itens é na casa dos milhões (Linden, 2003), a recomendação tradicional se torna impraticável e outras alternativas precisam ser consideradas para que recomendações de qualidade continuem sendo geradas em tempo aceitável.

Neste capítulo são apresentados os desafios e algumas soluções existentes para se trabalhar com matrizes de alta dimensionalidade. São expostos alguns trabalhos relacionados, o problema a ser resolvido é definido e a solução proposta é detalhada, passo a passo.

3.1 Trabalhos Relacionados

Reduzir a dimensionalidade da base de dados em um SR é uma opção para contornar o problema de performance e escalabidade (Sarwar, 2000). Na literatura, existem várias abordagens utilizando essa ideia. O sistema *Grundy* (Rich, 1979) é um dos primeiros a mostrar que descrever usuários através de esteriótipos baseados em algumas de suas características melhora a exatidão das recomendações. Esse trabalho incentivou vários outros (Burke, 2002; Balabanovic, 1998; Hooda, 2014; Victor, 2011) a buscarem formas de modelar usuários e itens a fim de aumentar a acurácia das recomendações.

A modelagem, ou redução, de itens ou usuários pode ser feita através da decomposição em fatores latentes, como mostrado em (Funk, 2015). Estratégias de agrupamento ou análise dos componentes principais (PCA) também podem ser uti-

[2] www.amazon.com

lizadas (Goldberg, 2001).

Uma das técnicas mais populares na filtragem colaborativa é a abordagem baseada nos k vizinhos mais próximos *(k-Nearest Neighbor ou k-NN)*. Esse método identifica pares de itens ou usuários que tendem a ter um comportamento similar, para então descobrir relações implícitas entre os usuários e itens (Bell, 2007). Este método é intuitivo e cada uma das recomendações geradas é facilmente explicada.

A técnica de *cluster* (agrupamento) tem o mesmo objetivo da técnica do k-NN: agrupar itens ou usuários com comportamento semelhante. Uma vez que a base de dados é agrupada, a recomendação pode ser feita pela média de opinião dos usuários de um grupo. Essa técnica pode produzir recomendações menos personalizadas e, consequentemente, com uma acurácia menor que o algoritmo dos k vizinhos mais próximos. Porém, quando o agrupamento está completo, a performance do algoritmo é ótima, já que o tamanho de cada grupo a ser analizado é menor (Bell, 2007).

Em (Sarwar, 2002), são unidas as técnicas de *cluster* e k-NN, explicado na Figura 3.1. Com base na similaridade entre os usuários são criados *clusters* e, para cada usuário u, os usuários pertencentes ao *cluster* em que ele está inserido são considerados sua vizinhança. O trabalho utiliza a matriz usuários-itens para gerar uma matriz que possui somente dados dos usuários.

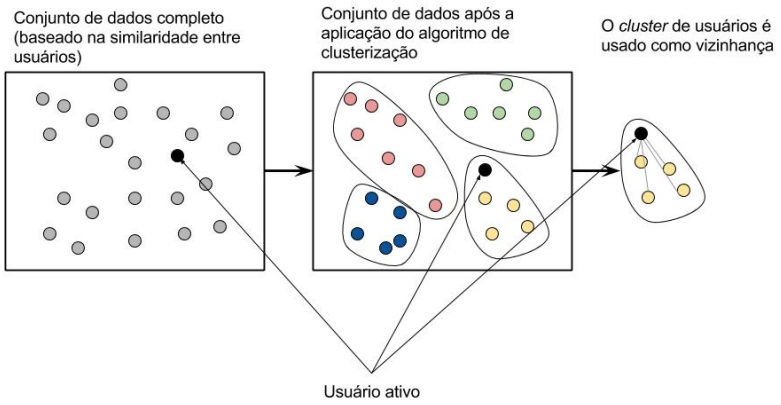
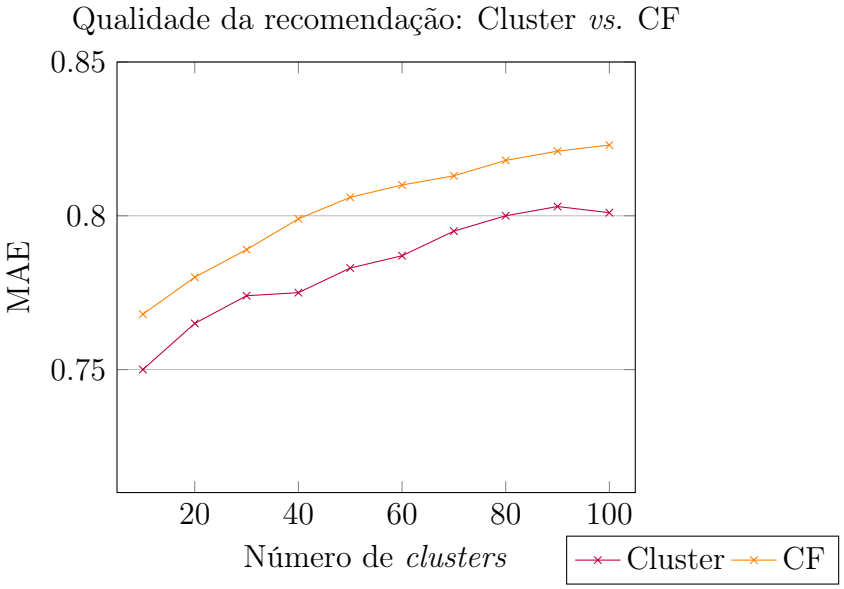


Figura 3.1: Formação de vizinhança a partir dos *clusters*. (Sarwar, 2002)

Nesse trabalho os dados de treinos são divididos de tal forma que a interseção entre dois *clusters* é vazia. Formalmente, o *dataset* A é particionado em $A_1, A_2, ..., A_p$,

onde $A_i \cap A_j = \phi$ para $1 \leq i, j \leq p$; e $A_1 \cup A_2 \cup ... \cup A_p = A$. Então, a vizinhança de um usuário u é definida como o *cluster* a qual esse usuário pertence.

Após a definição da vizinhança, o método clássico da filtragem colaborativa é utilizado para gerar as recomendações. O algoritmo de clusterização utilizado é uma adaptação do *k-means* chamado *bisecting k-means*. O trabalho mostra que a técnica produz recomendações com qualidade comparável a filtragem colaborativa básica e melhora significantemente a performance.



O algoritmo *k-means* original, que é um dos mais simples métodos de aprendizado de máquina não-supervisionado, também é uma alternativa para a redução de dimensionalidade da base. O método propõe uma forma fácil de classificar os dados a partir de um número k de classes definidas previamente. O objetivo desse algoritmo é minimizar uma função objetivo como a raiz quadrada do erro, por exemplo. Em (Kularbphettong, 2014), essa técnica é combinada com a filtragem colaborativa. A fórmula Euclidiana mostrada a seguir:

$$j = \sum_{j=1}^{x} \sum_{i=1}^{x} ||x_i^{(j)} - c_j||^2 \tag{3.1}$$

é usada para medir a distância entre um ponto $x_i^{(j)}$ e o centro da classe c_j, indicando a distância dos elementos da base aos centros dos suas respectivas classes.

O trabalho agrupa os 150 usuários da base em 3 grupos, baseados nos dados informados pelo usuário para a construção de seu perfil, como idade e gênero. O *k-means* foi utilizado para agrupar os usuários conforme sua similaridade e a filtragem colaborativa produziu resultados com base nas notas explicitadas pelos mesmos. Os resultados do trabalho mostram que essa aproximação gerou, com sucesso, recomendações que iam de acordo com as preferências dos usuários.

Analisando os trabalhos relevantes, é notável que um conhecimento extra, além das avaliações, sobre usuários ou itens, é essencial para a maior parte das técnicas de redução da dimensionalidade dos dados. Muitas delas utilizam informações como idade e gênero dos usuários ou, então, algumas características dos itens para agrupar os mesmos. No entanto, em alguns sistemas de recomendação como de vídeos ou notícias, por exemplo, pode ser difícil obter essas informações, o que impossibilitaria a execução desses algoritmos.

Um modelo independente do domínio do sistema de recomendação seria uma boa maneira de solucionar esse problema. Agrupar usuários ou itens similares, de forma a reduzir a quantidade de análises a serem feitas na recomendação, somente com base em suas avaliações, seria um avanço considerável. Além da redução de custos, já que serão analisados somente os fatores que influenciam a avaliação de um usuário u a um item i, contra todos as variáveis disponíveis, esse modelo poderia ser aplicado em qualquer sistema de recomendação, já que todos possuem uma matriz usuários-itens.

3.2 Definição do Problema

Como discutido, o objetivo é propor uma metodologia de particionamento da base de dados que mantenha a precisão das recomendações. A característica crucial dessa metodologia é agrupar itens ou usuários, similares simplesmente com base nos dados da matriz de avaliações, desconsiderando qualquer informação externa sobre eles.

A motivação principal dessa proposta é que, atualmente, é impraticável executar

as recomendações para toda uma base usuários-itens em tempo real, dada a dimensão da mesma. Além disso, informações sobre o domínio do sistema não são facilmente obtidas em todos os sistemas de recomendação.

No contexto dos sistemas de recomendação, temos um conjunto U de usuários e I de itens. Cada usuário u pode explicitar sua preferência para um subconjunto de itens i' através de uma nota r. Esta nota é definida dentro de um conjunto de preferência R, que demonstra o interesse do usuário u para cada item desse subconjunto.

Usuário semelhantes tendem a avaliar itens de forma similar, então, ao encontrar um subconjunto de usuários u' similar a um usuário u, podemos calcular a nota de um item i, ainda não avaliado por esse usuário, a partir das notas dos usuários similares a u.

O mesmo pode ser aplicado aos itens. Dado um subconjunto de itens i' similares a um item i, a nota desse item para um usuário v pode ser obtida através das notas recebidas pelos itens do subconjunto i'.

Os itens ou usuários similares podem ser encontrados de diversas formas, como discutido na seção anterior. O objetivo do trabalho é definir um processo que reúna itens similares de forma automática, independente do domínio e sem a necessidade de conhecimento *a priori* sobre os mesmos. Esses grupos serão utilizados para gerar modelos de previsão. As recomendações computadas nos grupos devem manter a mesma acurácia do que as recomendações considerando toda a base.

3.3 Proposta

O presente trabalho propõe uma forma alternativa de execução do algoritmo de filtragem colaborativa, com foco em uma maneira diferente de calcular a similaridade entre os itens, desconsiderando informações externas e de domínio. Essa alternativa visa particionar a base de dados a fim de diminuir o tempo computacional do algoritmo.

A similaridade entre os itens é uma parte crucial da filtragem colaborativa e, como fatores latentes são uma representação alternativa de um item (Funk, 2015), é crível que podemos calcular a similaridade entre eles a partir desses fatores.

Dado o padrão das avaliações dos usuários para os itens, é viável extrair os termos latentes de cada um deles fatorando a matriz usuários-itens (Koren, 2009). Existem diversos metódos na álgebra linear utilizados para a decomposição de matrizes, como *PCA*, *SVD* e QR, por exemplo (Koren, 2009). Estes métodos rezudem a dimensão extraindo variáveis latentes que contenham o máximo de informação possível sobre os dados originais (Varella, 2008). Os algoritmos permitem que a matriz original seja explicitada de outra forma, eliminando os componentes que não trazem informação relevante à ela.

As variáveis latentes tentam explicar de alguma maneira a relação entre os dados e seu padrão de variação. Ao aplicar a decomposição à matriz de usuários-itens, serão geradas duas novas matrizes (U e I), que são as matrizes dos fatores dos usuários e itens, respectivamente. Cada usuário u é definido como um vetor de fatores U_u; o mesmo vale para o item. Nesse exemplo, os fatores podem ser relacionados com a ocupação do usuário, como estudante, vendedor e outros.

A Figura 3.2 exemplifica uma forma de agrupamento, onde usuários são unidos de acordo com seus gostos por tipos de filmes.

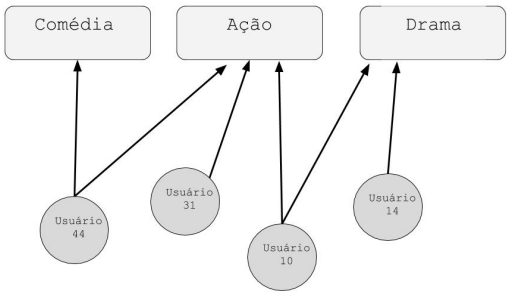


Figura 3.2: Agrupamento de usuários de acordo com suas preferências. (Mackey, 2014)

Um outro exemplo, mostrado na Figura 3.3 que utiliza a base dos filmes da

Netflix[3], mostra a aplicação do algoritmo $k - NN$ a uma parte dessa base de filmes. Esse exemplo mostra que filmes semelhantes, como os infantis ou os de ação, se agrupam no mesmo conjunto.

Nessa figura são mostrados dois dos *clusters* gerados: um que agrupa filmes infantis e outro com filmes de ação. É crível que dada essa organização dos itens, a previsão das notas para esses grupos tende a ser coerente, já que eles são agrupados de forma que faz sentido, de acordo com o gênero a qual cada um pertence.

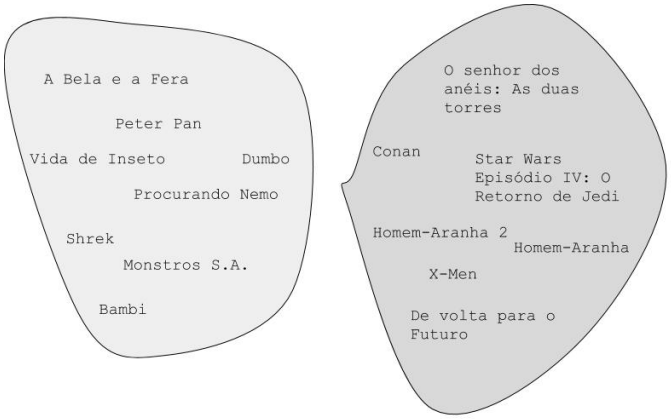


Figura 3.3: Amostra do resultado da execução do algoritmo k-NN na base de filmes do *Netflix*.

À vista disso, a proposta para esse trabalho é realizar recomendações baseadas em grupos de itens, de forma que menos análises sejam realizadas ao se executar essa recomendação.

Em um primeiro momento, os itens serão agrupados de acordo com as categorias em que se encontram e, então, serão executados métodos de predição para gerar as notas para esses grupos, esta configuração será a base dos experimentos.

Em um segundo momento, as variáveis latentes serão extraídas da matriz usuários-itens e um algoritmo de clusterização será aplicado para definir os grupos de itens similares. Nestes grupos serão executados modelos de previsão.

O resultado da clusterização será comparado à base do experimento, para avaliar se é possível organizar os itens sem conhecimento prévio e manter a qualidade e o

[3]www.netflix.com

tempo de execução das recomendações.

Capítulo 4

Avaliação Experimental

Neste capítulo são apresentados os objetivos dos experimentos e a metodologia utilizada. A organização dos expetimentos é descrita, assim como sua execução e resultados.

4.1 Objetivos dos experimentos

No capítulo anterior foi apresentada a proposta do presente trabalho, que é agrupar os itens similares com base nos seus fatores latentes, reduzindo a dimensionalidade da base de dados e obtendo performance e acurácia aceitáveis nos sistemas de recomendação. Para atingir esse objetivo, foi proposta a junção de técnicas de decomposição de matrizes com técnicas de inteligência artificial para o agrupamento dos dados, além de técnicas clássicas de filtragem colaborativa para realizar as recomendações a partir desses dados.

Esse trabalho foi dividido em três etapas: a extração de fatores latentes, ou decomposição de matrizes; os algoritmos de clusterização para agrupamento dos dados e os algoritmos de filtragem colaborativa para as recomendações.

Existem inúmeras configurações para a combinação dessas três fases, e a escolha da primeira parte pode influenciar nos algoritmos utilizados nas seguintes, uma determinada técnica de junção esparsa dos dados pode prejudicar um algoritmo que

necessite de certa proximidade ou semelhança entre eles, por exemplo.

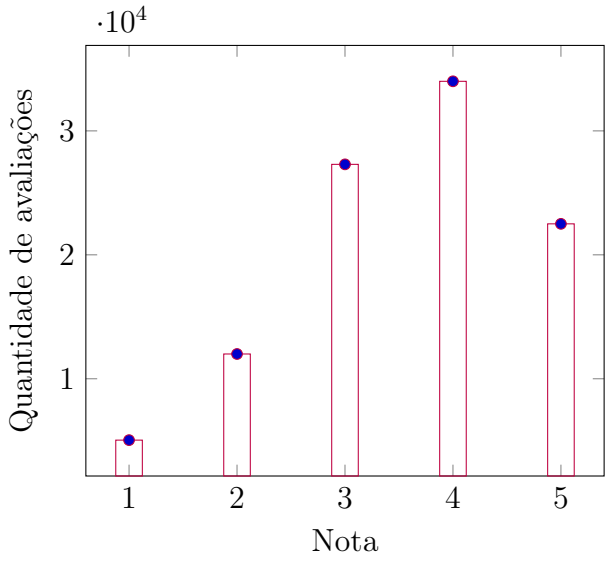
O objetivo dos experimentos é avaliar algumas dessas combinações e verificar seu desempenho e precisão nas recomendações, além de, comparar com a base dos experimentos.

4.2 Base de dados

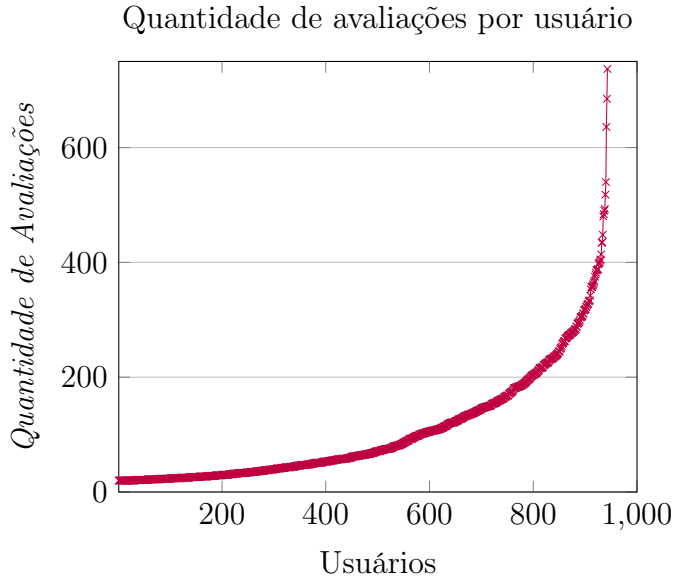
A base de dados utilizada nos experimentos foi o *MovieLens*[1], um *site* que realiza recomendação de filmes de forma não-comercial, desenvolvido pelo *GroupLens*[2]. O *site* possui 100.000 avaliações, 943 usuários e 1682 filmes, e seu conjunto de preferências é um número inteiro que varia de 1 a 5. Além das avaliações, existem outras informações sobre os usuários, como idade e sexo, e sobre os filmes, como título, data de lançamento e gêneros a qual um filme pertence.

O *MovieLens*[1] exige que cada usuário avalie pelo menos 20 itens. No entanto, a quantidade de avaliações existentes nesse sistema continua seguindo a tendência de outras bases de sistemas de recomendação, detendo poucas avaliações quando comparado a todas as possibilidades. A proporção de avaliações com relação ao total, ou seja, o índice de esparsidade é de 6.30%. A distribuição das notas se comporta como uma normal de média 3.5299 e desvio padrão de 1.1257, como mostrado abaixo:

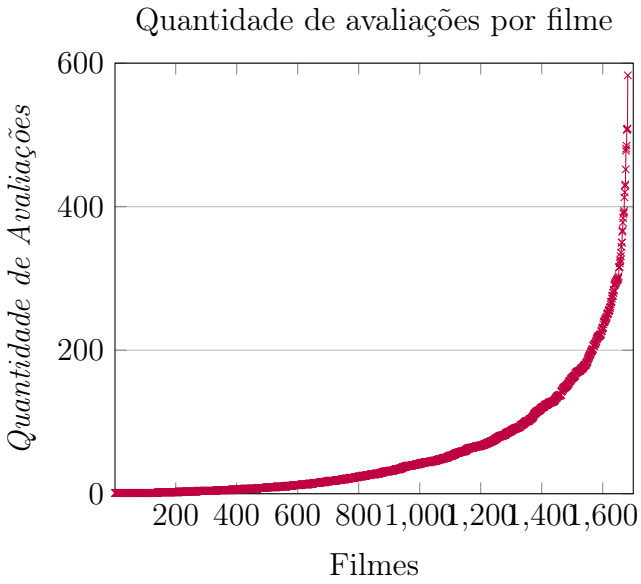
[1]http://www.movielens.org
[2]http://www.grouplens.org



A quantidade de avaliações feita por um usuário tem média de 106.04 e se caracteriza como uma distribuição exponencial, como mostra o gráfico abaixo, onde a quantidade mínima de avaliações é 20 e a máxima 737.



Os filmes possuem uma quantidade média de avaliações de 59.45 e esta também é disposta em uma distribuição exponencial, como ilustrado no gráfico abaixo, onde a quantidade máxima de avaliações para um filme é 583.



Abaixo segue um trecho dos dados utilizados na base dos experimentos, que consistiu em analisar as informações dos itens.

```
1 | Toy Story (1995) | 01-Jan-1995 | ... |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2 | GoldenEye (1995) | 01-Jan-1995 |...|0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3 | Four Rooms (1995) | 01-Jan-1995 | ... |0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4 | Get Shorty (1995) | 01-Jan-1995 |...|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
```

Figura 4.1: Trecho da base de dados dos itens

Essa base apresenta informações dos filmes como título, ano de lançamento, *link* para o IMDB[1] e uma relação de gêneros, representada por 0, caso um filme não pertenca a um gênero e 1, caso contrário.

Abaixo é apresentado um trecho da base de dados da matriz de avaliações, que foi usada em todos os experimentos para avaliar a acurária.

[1]http://www.imdb.com

User	Item	Rating	Time
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596
298	474	4	884182806
115	265	2	881171488
253	465	5	891628467
305	451	3	886324817
6	86	3	883603013

Figura 4.2: Trecho da base de dados das avaliações

4.3 Avaliação

Para analisar a exatidão dos algoritmos, serão consideradas duas métricas amplamente utilizadas para comparar técnicas de recomendação. São elas a *Mean Absolute Error* (MAE), que é a média da diferença absoluta entre as notas estimadas e as reais, e o *Root Mean Squared Error* (RMSE), que é a raiz quadrada da MAE ao quadrado.

O cálculo do MAE é detalhado na equação abaixo, onde k é o número total de avaliações de todos os usuários, p_{ui} é a previsão da nota de um usuário u para um item i e n_{ui} é a nota real.

$$MAE = \frac{\sum_{(u,i)} |p_{ui} - n_{ui}|}{k} \tag{4.1}$$

O RMSE se tornou popular após o *Netflix Prize*[2]. Essa medida penaliza erros grandes comparando-os aos pequenos. Sua equação é detalhada abaixo, onde k é o número total de avaliações de todos os usuários, p_{ui} é a previsão da nota de um usuário u para um item i e n_{ui} é a nota real.

$$RMSE = \sqrt{\frac{\sum_{(u,i)} |p_{ui} - n_{ui}|^2}{k}} \tag{4.2}$$

[2]http://www.netflixprize.com

4.4 Metodologia e Organização dos Experimentos

A metodologia desse trabalho será composta de diversos experimentos com o objetivo de avaliar o desempenho das técnicas de recomendação unidas a algumas combinações dos parâmetros, como o algoritmo escolhido para o modelo de previsão, o número de fatores latentes, a quantidade mínima de vizinhos e *clusters*.

A metodologia proposta possui duas etapas: união dos itens similares e aplicação das técnicas de filtragem colaborativa a esses grupos. Foram combinadas técnicas diferentes para a execução da primeira e segunda etapa. Para agrupar os itens similares, duas abordagens foram adotadas: análise dos dados da base *MovieLens* e clusterização a partir dos fatores latentes. A análise da base avaliou a matriz filmes-gêneros, que relaciona os gêneros aos filmes. Nessa abordagem os grupos foram criados analisando quais filmes pertenciam a determinado gênero.

Para a clusterização, foi aplicada a técnica de decomposição de valores singulares (SVD) à matriz de preferências, assumindo a matriz V como os fatores latentes do item e ignorando a matriz U dos usuários, assim como seus valores singulares.

A escolha do algoritmo SVD para extração dos fatores latentes se justifica pela pela sua importância na área de *Latent Semantic Analysis* (Landauer, 1998) e pela sua simplicidade. Essa etapa é responsável por reduzir a dimensionalidade da base, com objetivo de diminuir o custo do algoritmo a ser executado na fase seguinte.

Após a extração dos fatores, foi aplicado o algoritmo k-means à matriz V, para a geração dos *clusters*. O *k-means* foi escolhido por ser um dos mais simples métodos de aprendizado de máquina não-supervisionado além de produzir bons resultados com respeito a classificação. A centralidade dos elementos foi considerada na inicialização dos *clusters* pois, como analisado em (He, 2004) e constatado empiricamente durante o desenvolvimento desse trabalho, a inicialização aleatória prejudica bastante os resultados.

Finalmente, foram aplicados três algoritmos clássicos da filtragem colaborativa: k-NN, *Regulared SVD* e *Improved Regulared SVD*. Esses algoritmos foram escolhidos em função de representar as duas principais abordagens na filtragem colaborativa:

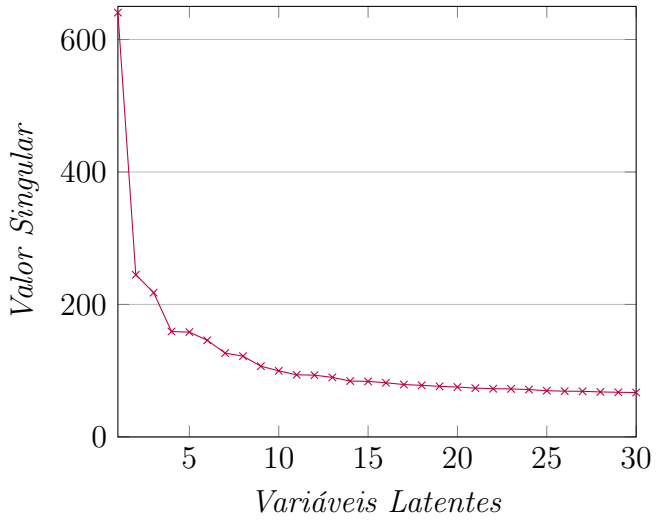
modelo e memória. Além disso, para garantir consistência nos resultados, cada configuração de experimento foi rodada aproximadamente 10 vezes.

Os parâmetros utilizados na proposta são:

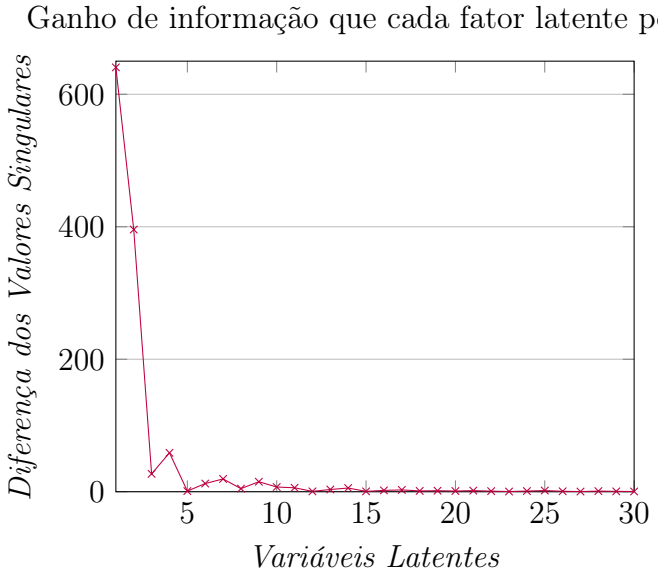
- Quantidade de fatores latentes;
- Quantidade de *folds*, para a validação cruzada;
- Quantidade de *clusters*;
- Algoritmos de filtragem colaborativa.

A quantidade de fatores latentes a ser utilizada foi determinada com a aplicação do algoritmo *SVD* na matriz usuários-itens da base *MovieLens* e os valores singulares extraídos de cada variável foram analisados, como mostrado abaixo.

Quantidade de informação que cada fator latente possui



Abaixo, é mostrada a diferença entre cada valor singular e seu antecessor.



Analisando os valores singulares, observa-se uma grande queda do valor na oitava variável latente e a partir da nona não existe variação considerável desse valor. Portanto, foi determinado que essa quantidade seria a máxima. Como os fatores anteriores possuem uma grande quantidade de informação, elas também foram consideradas e avaliadas nos experimentos.

Em todos os experimentos a validação cruzada foi utilizada para avaliar a variabilidade de cada algoritmo, que consistiu em utilizar o 10-*fold validation*, que divide a amostra original aleatoriamente em 10 sub-amostras e para cada uma dessas divisões é aplicado o algoritmo, utilizando uma dessas sub-amostras como teste e os demais como treinamento. A vantagem desse método é que todas as observações serão usadas tanto para treino quanto para a validação, e cada observação será utilizada para a validação no mínimo uma única vez.

Uma limitação encontrada na técnica dos vizinhos mais próximos foi a limitação com relação ao número mínimo de vizinhos para realizar a previsão, que afeta diretamente a cobertura. Para tal, foi estipulado um número mínimo de vizinhos, $k = 8$, que um item necessita para ser avaliado, a fim de garantir um melhor resultado para os modelos de previsão. Para efeito de análise, a cobertura dos dados de cada experimento foi também analisada.

Sumarizando, serão dois experimentos, com variações nos algoritmos a serem

analisados, com objetivo de avaliar a proposta do presente trabalho:

1. Avaliação do agrupamento de itens por categoria (ação, terror, aventura, infantil) e previsão das notas utilizando os algoritmos *k-NN*, *Regulared SVD* e *Improved Regulared SVD*, explicado abaixo na Figura 4.3.

Figura 4.3: *Workflow* da base dos experimentos

2. Avaliação da clusterização dos itens, a partir dos fatores latentes extraídas da matriz usuários-itens com o SVD; aplicação do algoritmo *k-NN*, *Regulared SVD* e *Improved Regulared SVD* para a previsão das notas. Todos com variação de 1 a 8 na quantidade de *features* a serem analisadas, explicado na Figura 4.4.

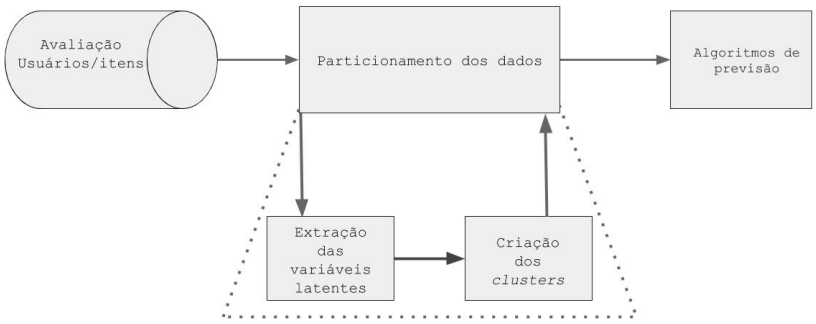


Figura 4.4: *Workflow* do experimentos principais

4.5 Resultados

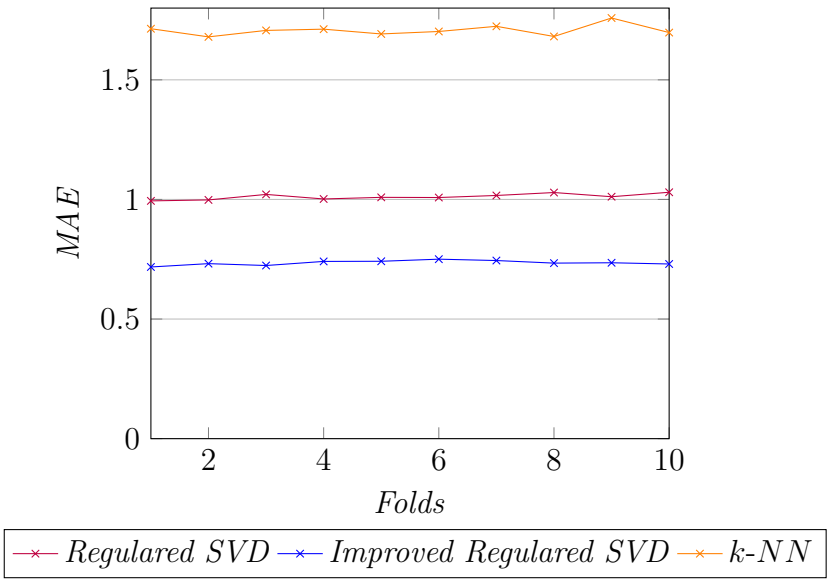
Nesta seção serão apresentados os resultados obtidos em cada experimento. Ao final, será feita uma análise conclusiva desses resultados, avaliando os ganhos obtidos com a utilização da proposta desse trabalho.

4.5.1 Experimento I

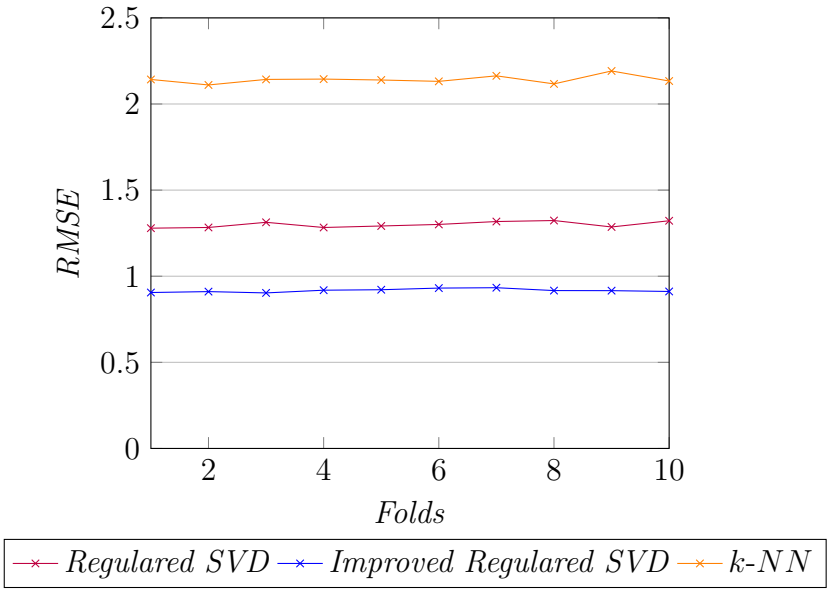
Este experimento é a *baseline* do trabalho. Nele, os itens são agrupados de acordo com informações de gênero contidas na base de itens. É executado a validação cruzada, onde o conjunto de dados é separado em *folds* para treino e teste, e esses conjuntos são treinados com os modelos de previsão de acordo com o gênero a qual cada item pertence.

As recomendações serão executadas com o algoritmo *Regulared SVD, Improved Regulared SVD* e k-NN, e para garantir a consistência dos resultados o experimento foi executado aproximadamente dez vezes. Seguem os resultados encontrados.

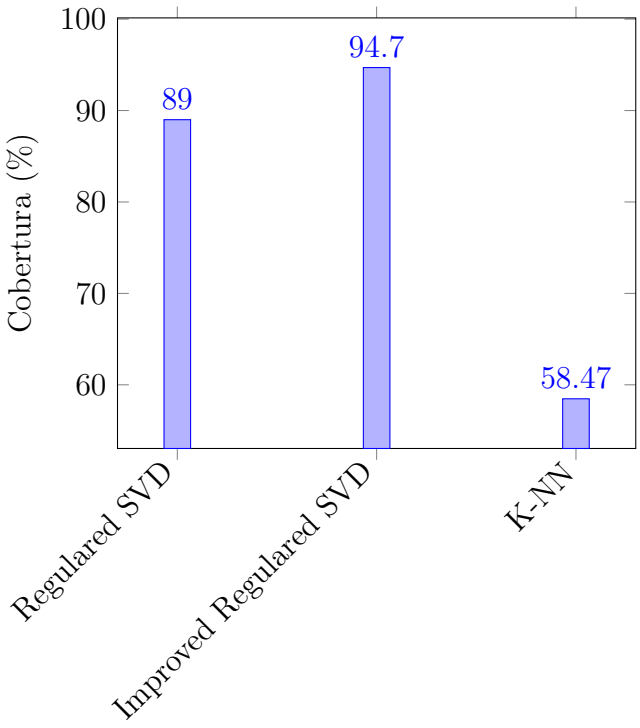
(1) - Avaliação do valor da MAE de cada algoritmo por *fold*



(2) - Avaliação do valor da RMSE de cada algoritmo por *fold*



(3) - Porcentagem de cobertura de cada algoritmo



O algoritmo com melhor resultado foi o *Improved Regulared SVD*, em todos os aspectos, tanto na MAE, RMSE quanto na cobertura. O seu desempenho se justifica em partes por sua cobertura, que se destacou, além de considerar os *bias* do usuário e do item. O *Regulared SVD* seguiu a mesma tendência, porém, com um resultado pior, quando comparado ao *Improved Regulared SVD*. O *k-NN* foi o algoritmo com

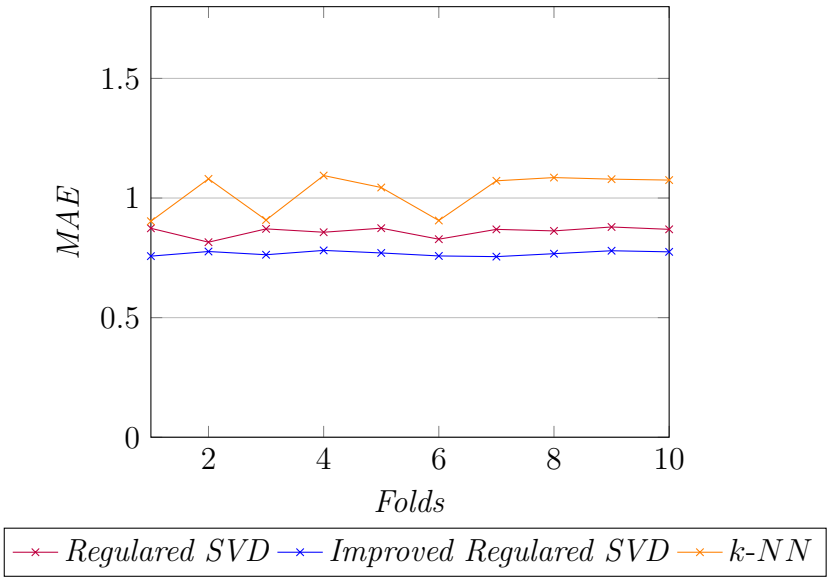
pior desempenho e cobertura, o que se justifica por não conseguir o número mínimo de vizinhos, definido no experimento como 8, nos agrupamentos realizados (por gênero).

4.5.2 Experimento II

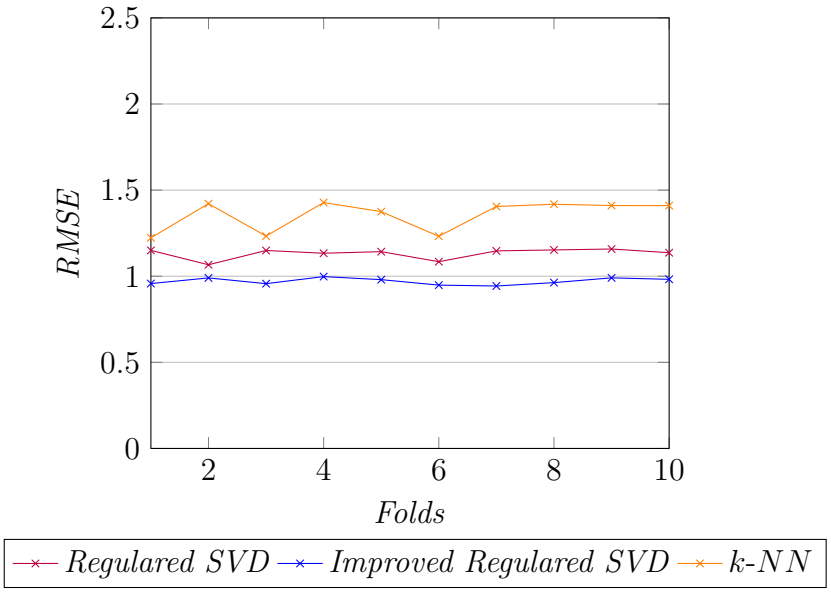
O segundo experimento usará a matriz de preferências para a extração dos fatores latentes dos itens, com a técnica SVD usando de um a oito fatores latentes para tal.

A validação cruzada é inserida neste momento do experimento, para cada *fold* são gerados 10 *clusters* com os dados de treino do *fold*, a melhor configuração de *clusters* é escolhida com base no menor valor da função objetivo obtida. Então, é realizado o teste com os dados do mesmo *fold*. Os *clusters* são gerados com o algoritmo *k-means* e sua inicialização leva em conta a centralidade dos elementos. Seguem os resultados encontrados.

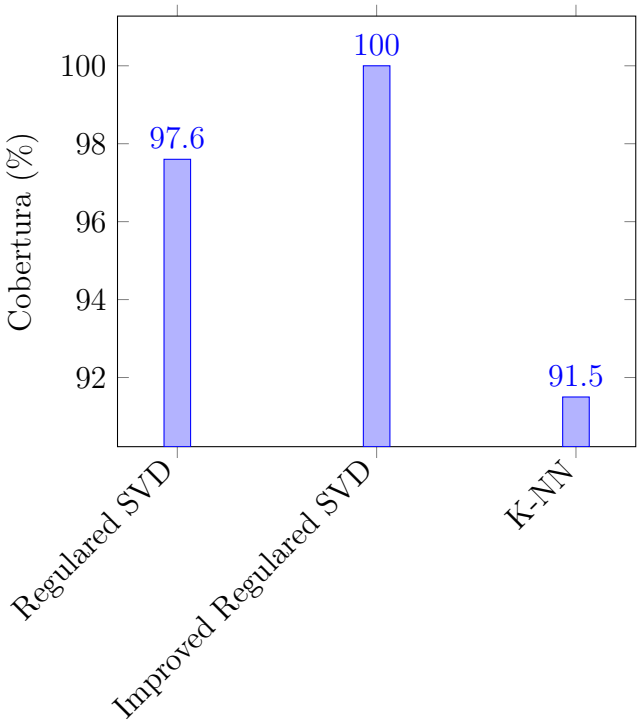
(1) - Avaliação do valor da MAE de cada algoritmo por *fold*



(2) - Avaliação do valor da RMSE de cada algoritmo por *fold*



(3) - Porcentagem de cobertura de cada algoritmo



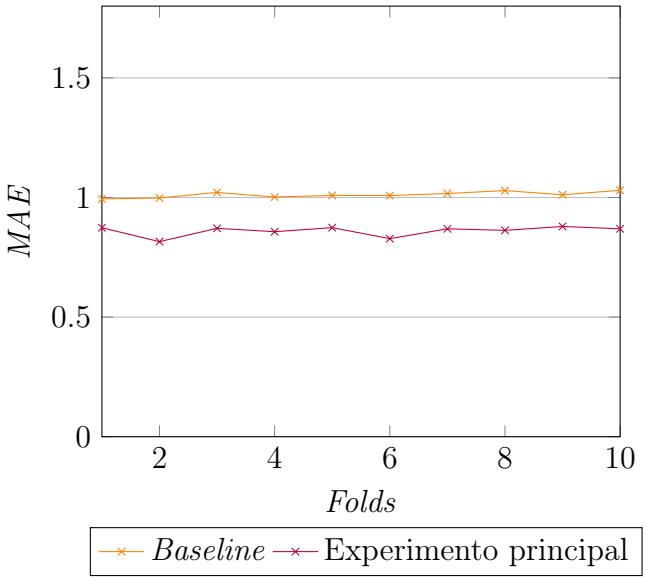
Neste experimento o melhor resultado foi obtido com o *Improved Regulared SVD*, que obteve 100% de cobertura. O *Regulared SVD* também obteve um resultado bom, seguindo a tendência do *Improved Regulared SVD*. O *k-NN* se destacou pois teve uma variância maior do que os outros no resultado e, seu desempenho continua

sendo o pior dos três algoritmos. Com a clusterização a cobertura dos algoritmos foi maior, no geral, o que melhorou os resultados como um todo.

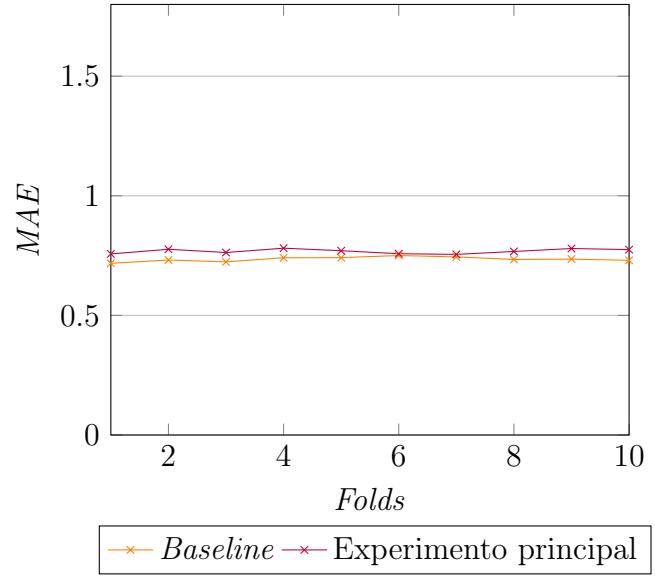
4.5.3 Experimento III

Este experimento é responsável por comparar os resultados do *baseline* e os resultados do experimento principal, com intuito de analisar o ganho obtido devido a técnica de particionamento proposta no trabalho. Seguem os resultados encontrados.

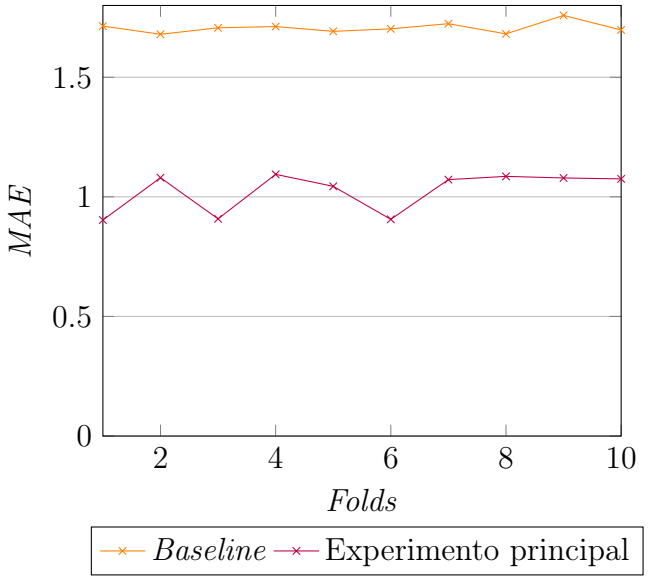
(1) - Avaliação do valor da MAE na execução do *Regulared SVD* por *fold*



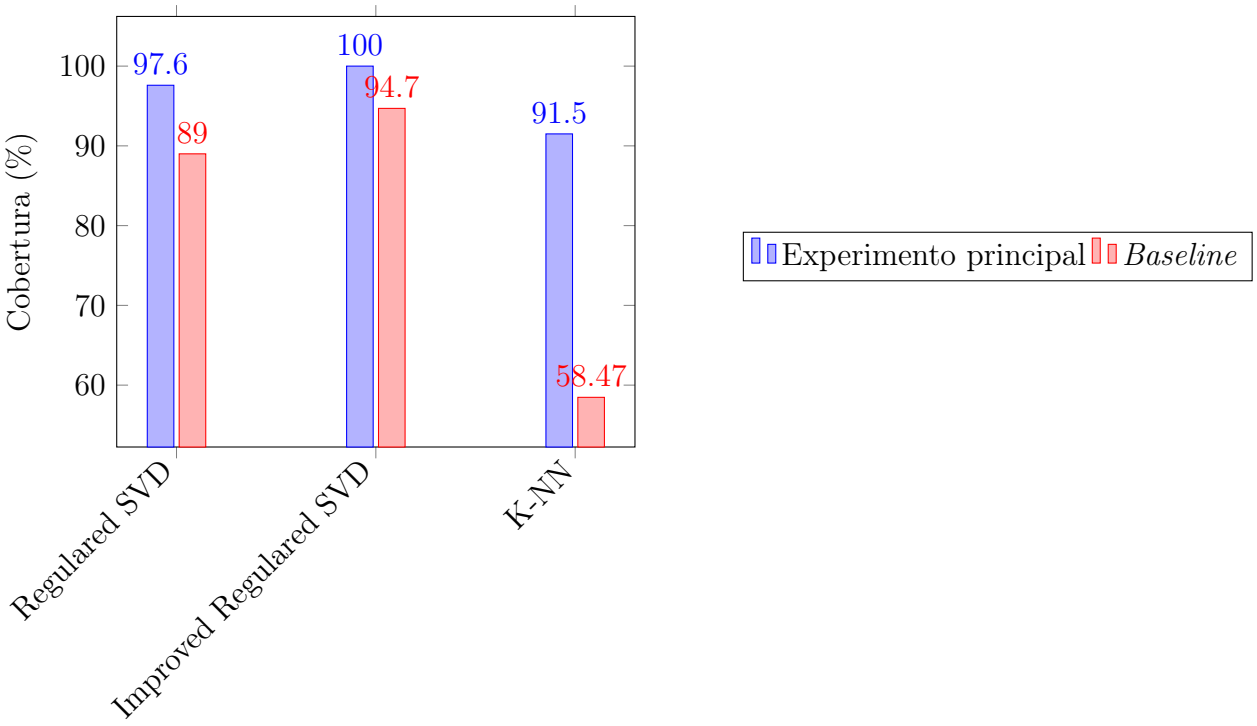
(2) - Avaliação do valor da MAE na execução do *Improved Regulared SVD* por *fold*



(3) - Avaliação do valor da MAE na execução do *k-NN* por *fold*



(4) - Porcentagem de cobertura de cada algoritmo



A cobertura dos algoritmos aumentou com a técnica de clusterização, o que afetou principalmente os algoritmos *k-NN* e *Regulared SVD*, os dois tiveram uma melhora no resultado, em comparação com o *baseline*. O aumento na cobertura afetou de forma mais intensa o *k-NN*, pois os *clusters* permitiram que mais vizinhos fossem encontrados, o que diminuiu consideravelmente o valor do erro encontrado.

Já o *Improved Regulared SVD* não foi tão influenciado pelo particionamento dos dados, o seu desempenho é muito similar nas duas abordagens.

4.5.4 Análise dos Dados

Nesta seção são discutidos os resultados apresentados até o momento, evidenciando os resultados obtidos com a configuração proposta.

O primeiro experimento tinha como objetivo avaliar o quanto as informações inerentes aos itens poderiam influenciar no particionamento e, consequentemente, no desempenho dos algoritmos. Para cada classe de algoritmo utilizado houve um comportamento diferente nesse experimento, os algoritmos baseados em modelo ob-

tiveram um desempenho melhor do que os baseados em memória, inclusive a variação a cada *fold* dos algoritmos de modelo também foi menor. Como as duas medidas de erro seguem a mesma tendência, foi verificado que para os maiores valores de MAE, são encontrados os maiores valores do RMSE.

O experimento mostrou que utilizar informações dos itens (como gênero dos filmes dessa base) não traz ganho no desempenho da recomendação e, para alguns até piora consideravelmente, como foi o caso do k-NN. A cobertura obtida no experimento não foi boa o suficiente, o que mostra, mais uma vez que esse particionamento não é o melhor a ser realizado.

O segundo experimento tinha como objetivo avaliar a proposta do trabalho, particionando os itens a partir dos seus fatores latentes. O experimento obteve um resultado consistentemente melhor do que o primeiro, o que mostra a validade da proposta deste trabalho. O erro dos algoritmos foi menor e a cobertura foi maior, em todos eles, mesmo que cada um tenha tido um comportamento diferente. Os algoritmos baseados em modelo tiveram um desempenho melhor com relação ao baseado em memória, no entanto, como dito anteriormente, todos os resultados tiveram uma melhora considerável. A cobertura dos algoritmos segui a tendência e chegou a 100% no *Improved Regulared SVD*, uma das causas do aperfeiçoamento do resultado.

Capítulo 5

Conclusões

Este capítulo reporta as considerações finais referentes a este trabalho, suas contribuições e perspectivas futuras.

5.1 Considerações finais

O presente trabalho propôs uma decomposição automática dos dados, com base nos fatores latentes utilizando a matriz de preferências do sistema de recomendação, partindo do ideia de que é possível realizar recomendações sem conhecimento prévio do domínio do sistema.

A filtragem colaborativa, uma das mais conhecidas abordagens de recomendação, utiliza as preferências explícitas dos usuários sobre os itens para a tarefa de recomendar. No entanto, ainda tem a necessidade de relacionar itens e/ou usuários similares a fim de direcionar as sugestões.

Portanto, esse trabalho propôs uma técnica de particionamento dos dados capaz de extrair a similaridade e um padrão de avaliação entre eles, para que então, apenas com as de notas explicitadas pelos usuários a cada item, fosse realizada a recomendação destes.

5.2 Contribuições

A principal colaboração desse trabalho é a confirmação de que é possível executar recomendações sem conhecimento *a priori* do domínio. Para tal, foi proposta uma metodologia que particiona os dados e a partir disso realiza as recomendações. Foi proposta a teoria de que existem fatores latentes que representam os usuários e os itens, e com isso seria possível, a partir da matriz de preferências, encontrar um padrão para representação das notas através desses fatores e assim aplicar os algoritmos de recomendação.

A fim de validar a propositura foram realizados alguns experimentos visando a avaliação do desempenho da técnica utilizando diversas combinações dos parâmetros, como exemplo, a quantidade de fatores latentes utilizados na clusterização, o algoritmo de recomendação e outros, e por fim foi comparado o desempenho da proposta com a base do trabalho, que utiliza informações dos itens para agregá-los.

Os experimentos realizados obtiveram resultados satisfatórios e superiores a *baseline* proposta. Além de ser superior nas métricas, como MAE e o RMSE, a proposta obteve uma melhora na cobertura, o que significa que a previsão das notas foi realizada para mais itens, ou seja, mais notas foram geradas e mais itens analisados, o que contribuiu para o desempenho superior da proposta.

5.3 Limitações e trabalhos futuros

Esta seção expõe potenciais pontos de melhoria e ampliação do particionamento automático dos dados e experimentos futuros, alguns dos quais são também soluções às limitações também descritas.

Os experimentos mostram que a solução proposta para a recomendação de notas dentro do problema de filtragem colaborativa é válida. Neste trabalho, foram realizados experimentos que demonstraram a eficiência da agregação com base nos fatores latentes e a aplicação dos algoritmos da filtragem colaborativa.

Um desafio é utilizar tal abordagem em um sistema de recomendação real. Todos

os experimentos foram realizados visando um sistema de recomendação *off-line* e com base fixa. Todavia, é necessário que além da baixa taxa de erro, o algoritmo esteja preparado para bases maiores e atualização dos dados, isto é, que seja escalável.

Uma limitação do trabalho é a utilização de apenas uma base para testar e validar a proposta. Na literatura existem diversas bases diferentes com características específicas, como o tamanho do conjunto de preferência, a distribuição das notas tanto para o usuário quanto para o item, o grau de esparsidade e outros. Essas características resultam em resultados melhores ou piores, de acordo com cada algoritmo utilizado. Um trabalho futuro seria analisar o desempenho da proposta em outras bases de dados.

Referências

Alexeer Adomavicius, Gediminas e Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

Marko Balabanovic. Learning to surf: multiagent systems for adaptive web page recommendation. 1998.

Bawden. Information and digital literacies: a review of concepts. *Journal of documentation*, 57(2):218–259, 2001.

Yehuda Bell, Robert M e Koren. Improved neighborhood-based collaborative filtering. In *KDD-Cup e Workshop*, pages 7–14. ACM press, 2007.

Susan T e O'Brien Gavin W Berry, Michael W e Dumais. Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595, 1995.

Michael J Billsus, Daniel e Pazzani. Learning collaborative information filters. In *ICML*, volume 98, pages 46–54, 1998.

David e Kadie Carl Breese, John S e Heckerman. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

Robin Burke. Hybrid web recommender systems, the adaptive web: methods and strategies of web personalization, 2007.

John Canny. Collaborative filtering with privacy. In *Security e Privacy, 2002. Proceedings. 2002 IEEE Symposium on*, pages 45–57. IEEE, 2002.

Joseph G e Yiannoutsos Constantin Chen, Ming-Hui e Ibrahim. Prior elicitation, variable selection and bayesian computation for logistic regression models. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, pages 223–242, 1999.

Nan M e Rubin Donald B Dempster, Arthur P e Laird. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Juan Manuel Adán Dodorico, Bruno Henrique Sebba e Coello. Previsão do desempenho de estudantes useo algoritmos de filtragem colaborativa baseados em fatoração de matrizes. 2014.

Simon Funk. Disponível em <http://sifter.org/simon/journal/20061211.html>, October 2015. URL `http://sifter.org/~simon/journal/20061211.html`.

Carla e Jannach Dietmar Ge, Mouzhi e Delgado-Battenfeld. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 257–260. ACM, 2010.

David e Oki Brian M e Terry Douglas Goldberg, David e Nichols. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12): 61–70, 1992.

Theresa e Gupta Dhruv e Perkins Chris Goldberg, Ken e Roeder. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

Mark e Narita Hiro e Goldberg Ken Gupta, Dhruv e Digiovanni. Jester 2.0 (poster abstract): evaluation of an new linear time collaborative filtering algorithm. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 291–292. ACM, 1999.

Man e Tan Chew-Lim e Sung Sam-Yuan e Low Hwee-Boon He, Ji e Lan. Initialization of cluster refinement algorithms: A review and comparative study. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 1. IEEE, 2004.

Joseph A e Terveen Loren G e Riedl John T Herlocker, Jonathan L e Konstan. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

Kulvinder e Dhawan Sanjeev Hooda, Rahul e Singh. A study of recommender systems on social networks and content-based web systems. *International Journal of Computer Applications*, 97(4):23–28, 2014.

Lior e Ricci Francesco e Shapira Bracha Kantor, Paul B e Rokach. *Recommender systems handbook*. Springer, 2011.

Bong-Jin Kim, Dohyun e Yum. Collaborative filtering based on iterative principal component analysis. *Expert Systems with Applications*, 28(4):823–830, 2005.

Robert e Volinsky Chris Koren, Yehuda e Bell. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

Sunisa e Tongsiri Cholticha e Roonrakwit Pattarapan Kularbphettong, Kunyanuth e Somngam. A recommender system using collaborative filtering and k-mean based on eroid application. *Journal of Theoretical & Applied Information Technology*, 70(1), 2014.

Peter W e Laham Darrell Landauer, Thomas K e Foltz. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

Brent e York Jeremy Linden, Greg e Smith. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

Francesco Mahmood, Tariq e Ricci. Improving recommender systems with adaptive conversational strategies. In *Proceedings of the 20th ACM conference on Hypertext e hypermedia*, pages 73–82. ACM, 2009.

Paolo Massa, Paolo e Avesani. Trust-aware collaborative filtering for recommender systems. In *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, pages 492–508. Springer, 2004.

Dimitris e Kutsuras Themistoklis Papagelis, Manos e Plexousakis. Alleviating the sparsity problem of collaborative filtering using trust inferences. In *Trust management*, pages 224–239. Springer, 2005.

Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup e workshop*, volume 2007, pages 5–8, 2007.

David E Pelikan, Martin e Goldberg. Hierarchical bayesian optimization algorithm= bayesian optimization algorithm+ niching+ local structures. 2001.

Neophytos e Suchak Mitesh e Bergstrom Peter e Riedl John Resnick, Paul e Iacovou. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM. ISBN 0-89791-689-1. doi: 10.1145/192844.192905. URL `http://doi.acm.org/10.1145/192844.192905`.

Lior e Shapira Bracha Ricci, Francesco e Rokach. *Introduction to recommender systems hebook*. Springer, 2011.

Elaine Rich. User modeling via stereotypes*. *Cognitive science*, 3(4):329–354, 1979.

George e Konstan Joseph e Riedl John Sarwar, Badrul e Karypis. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167. ACM, 2000.

George e Konstan Joseph e Riedl John Sarwar, Badrul e Karypis. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

George e Konstan Joseph e Riedl John Sarwar, Badrul M e Karypis. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using

clustering. In *Proceedings of the fifth international conference on computer e information technology*, volume 1. Citeseer, 2002.

Dan e Herlocker Jon e Sen Shilad Schafer, J Ben e Frankowski. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.

Joseph e Riedl John Schafer, J Ben e Konstan. Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166. ACM, 1999.

Alexerin e Ungar Lyle H e Pennock David M Schein, erew I e Popescul. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research e development in information retrieval*, pages 253–260. ACM, 2002.

Taghi M Su, Xiaoyuan e Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4, 2009.

Marcelo Peixoto TERRA, João Cláudio e BAX. Portais corporativos: instrumento de gestão de informação e de conhecimento. *A Gestão da Informação e do Conhecimento*, 1:33–53, 2003.

John F e Reinsel David e Minton Stephen Turner, Vernon e Gantz. The digital universe of opportunities: Rich data and the increasing value of the internet of things. *International Data Corporation, White Paper, IDC_1672*, 2014.

Carlos Alberto Alves Varella. Análise de componentes principais. *Universidade Federal Rural do Rio de Janeiro, Disponível em: http://www. ufrrj. br/institutos/it/deng/varella/Downloads. Acessado em*, 18, 2008.

Chris e De Cock Martine Victor, Patricia e Cornelis. *Trust networks for recommender systems*, volume 4. Springer Science & Business Media, 2011.