APLICAÇÃO DE REDE NEURAL SEM PESO E LOCALITY-SENSITIVE HASHING PARA O PROBLEMA DE COLD-START ITEM EM FILTRAGEM COLABORATIVA

Kleyton Pontes Cotta

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Geraldo Zimbrão da Silva

Rio de Janeiro
Setembro de 2018

APLICAÇÃO DE REDE NEURAL SEM PESO E LOCALITY-SENSITIVE HASHING PARA O PROBLEMA DE COLD-START ITEM EM FILTRAGEM COLABORATIVA

Kleyton Pontes Cotta

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Geraldo Zimbrão da Silva, D.Sc.

Prof. Alexandre de Assis Bento Lima, D.Sc.

Prof. Leandro Guimarães Marques Alvim, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2018

Cotta, Kleyton Pontes

Aplicação de Rede Neural Sem Peso e Locality-Sensitive Hashing para o Problema de Cold-Start Item em Filtragem Colaborativa/Kleyton Pontes Cotta. – Rio de Janeiro: UFRJ/COPPE, 2018.

XV, 92 p.: il.; 29, 7cm.

Orientador: Geraldo Zimbrão da Silva

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2018.

Referências Bibliográficas: p. 81 – 92.

1. Sistema de Recomendação. 2. Cold-start. 3. Deep Learning. 4. WiSARD. I. Silva, Geraldo Zimbrão da. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Aos meus filhos.

Agradecimentos

Quero agradecer aos meus pais, Sebastião Cotta e Maria do Rosário Pontes por toda educação que me ofereceram e pelo apoio em todos os momentos da minha vida. A minha irmã Kelly e minha tia Carmo, que deram forças para continuar meus estudos, contribuindo para superar os obstáculos.

Agradecer minha "Eterna Namorada", que me apoiou dando-me força nos momentos difíceis, sempre dizendo palavras positivas e tendo paciência nas minhas ausências. Portanto, agora só me resta dizer: "Quero vê-la sorrir, quero vê-la cantar...".

Agradecimento especial aos meus filhos Nicolas e Maria Lis, que são minha inspiração para seguir em frente e superar os desafios impostos pela vida, dedico todas as conquistas para essas crianças que eu amo.

Agradeço aos professores do PESC pelos ensinamentos passados, dos quais contribuíram na minha formação acadêmica durante esta jornada, em especial ao professor Geraldo Zimbrão, pelo apoio e orientação para realização deste trabalho.

Aos meus colegas da UFRRJ, UFRJ e COPPETEC, que não se negaram a ajudar quando solicitados, em especial, Hugo, Gustavo, Victor, Júlio e Raul, que contribuíram com discussões acaloradas sobre o tema e diversos outros assuntos.

Agradeço ao CNPq pelo apoio financeiro que viabilizou esta pesquisa e à COPPE/UFRJ pelo prazer de ter sido aluno desta instituição de ensino tão respeitada.

E por fim, agradecer todos meus amigos e familiares que acreditaram, incentivaram e souberam entender as minhas ausências.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

APLICAÇÃO DE REDE NEURAL SEM PESO E LOCALITY-SENSITIVE HASHING PARA O PROBLEMA DE COLD-START ITEM EM FILTRAGEM COLABORATIVA

Kleyton Pontes Cotta

Setembro/2018

Orientador: Geraldo Zimbrão da Silva

Programa: Engenharia de Sistemas e Computação

Sistemas de recomendação são técnicas e ferramentas usadas para sugerir itens personalizados baseados no perfil do usuário. A filtragem colaborativa é uma das abordagens mais bem sucedidas em sistemas de recomendação. Essa abordagem visa a recomendar um item para um usuário baseado em itens previamente avaliados por outros usuários do sistema. Entretanto, é amplamente sabido que as abordagens dos métodos de filtragem colaborativa sofrem de problemas como, *cold-start*, escalabilidade e dispersão. O problema de *cold-start* é um dilema de longa data em sistemas de recomendação. Ocorre quando existem a indisponibilidade de informações adequadas sobre os itens ou usuários disponíveis no sistema, e desta forma não é possível fazer recomendações relevantes. Neste contexto, o trabalho apresenta uma abordagem híbrida para o problema de *cold-start itens* em filtragem colaborativa para sistemas de recomendação. Utilizamos a técnica de *Locality-Sensitive Hashing* (LSH) com o método *MinMaxwise* para processamento de textos com intuito de encontrar as similaridades entre os itens. O LSH foi incorporado nas abordagens tradicionais da literatura e demais métodos de aprendizagem de máquina, gerando bons resultados para esses métodos. A WiSARD obteve resultado melhores em relação ao tempo de treinamento e para bases com a presença de itens com completo *cold-start*.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

APPLICATION OF WEIGHTLESS NEURAL NETWORK AND LOCALITY-SENSITIVE HASHING FOR THE COLD-START ITEM PROBLEM IN COLLABORATIVE FILTERING

Kleyton Pontes Cotta

September/2018

Advisor: Geraldo Zimbrão da Silva

Department: Systems Engineering and Computer Science

Recommender systems are techniques and tools used to suggest customized items based on the user profile. Collaborative filtering is one of the most successful approaches in recommender systems. This approach aims to recommend an item to a user based on items previously evaluated by other system users. However, it is widely known that approaches to collaborative filtering methods suffer from problems such as cold-start, scalability and scatter. The cold-start problem is a longstanding dilemma in recommendation systems. Occurs when there is unavailability of adequate information about the items or users available in the system and therefore it is not possible to make relevant recommendations. In this context, the paper presents a hybrid approach to the problem of cold-start items in collaborative filtering for recommender systems. We used the Locality-Sensitive Hashing (LSH) technique with the MinMaxwise method for word processing in order to find similarities among the items. LSH was incorporated into traditional literature approaches and other machine learning methods, generating good results for these methods. The WiSARD obtained better results regarding training time and for bases with the presence of items with complete cold-start.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Capítulo 1

Introdução

Neste capítulo, apresentamos uma breve introdução, acompanhada da motivação e os principais objetivos do nosso trabalho. Mostramos também as contribuições ocasionadas, e por fim, definimos a estrutura de leitura do trabalho.

1.1 Motivação

Sistemas de Recomendação obtiveram uma importância maior nos últimos anos, principalmente pelo fato da sobrecarga de informação, devido ao crescimento de novos negócios que oferecem milhares de possibilidades de bens e serviços aos seus usuários. O objetivo da recomendação é ajudar usuários a encontrar itens que sejam potencialmente interessantes a partir de um grande repositório de itens. Sistemas de recomendação são amplamente utilizados por aplicativos e serviços de *e-Commerce*, como redes sociais, Amazon, Netflix, Deezer, Youtube, Google News entre outros. Através da recomendação é possível minimizar a possibilidade de ocorrer um abandono no processo de compra, uma vez que fazer escolhas com muitas opções pode ser incômodo e difícil SCHWARTZ (2005).

Em vista deste problema, os sistemas de recomendação surgiram com o foco na busca por informações relevantes de acordo com características do próprio usuário, bem como em determinados requisitos relacionados aos itens que se quer encontrar. Em geral são utilizadas as três técnicas de filtragem citadas a seguir: filtragem baseada em conteúdo (CB), filtragem colaborativa (CF), também conhecida como filtragem social e filtragem híbrida.

A filtragem baseada em conteúdo constrói um perfil de usuário baseado nas características dos itens na qual o usuário teve alguma interação. Para recomendar um item, CB faz a correlação entre o conteúdo deste item e os interesses do usuário extraídos de seu perfil.

A filtragem colaborativa é uma das abordagens mais bem sucedidas em sistemas de recomendação. Essa abordagem visa recomendar um item para um usuário ba-

seado em itens previamente avaliados por outros usuários do sistema. Uma possível abordagem seria a modelagem de uma matriz R de dimensão N x M, onde N representa os usuários e M os itens, e cada posição de R[i,j] contém uma nota ou vazio BRAIDA *et al.* (2015).

Entretanto, é amplamente sabido que a abordagem da CF sofre com a esparsidade e com *cold-start* (CS). A matriz de *rating* possui apenas uma pequena porcentagem de itens classificados, até os itens mais populares podem ter apenas algumas classificações.

Dado que a matriz de *rating* esparsa é muito desafiador estimar a relação entre os itens e usuários para realizar recomendações eficazes. Outro problema para a abordagem de CF é o problema de *cold-start*, que ocorre para novos usuários ou novos itens. A abordagem CF requer um grande número de classificações de um usuário ou classificações em um item para que uma recomendação seja eficaz, porém não funcionará para novos usuários, novos itens ou ambos devido a poucas classificações disponíveis no sistema. Além disso, o problema de CS pode ser dividido em completo *cold-start* (CCS) no qual o número de registros de classificação para um item seja igual a zero e em incompleto *cold-start* (ICS) onde existem poucos registros de classificação para o item.

A abordagem híbrida é aquela que combina a abordagem de filtragem CB e a abordagem CF, tentando amenizar as dificuldades e fornecer um resultado mais eficiente (AGARWAL e CHEN, 2009, CHEN *et al.*, 2012, HU *et al.*, 2013). Note-se que a maioria dos trabalhos sobre o problema de recomendação para CS é fornecer recomendações de itens que podem ser interessantes para determinados usuários. Embora muitos trabalhos tenham sido feitos com a abordagem híbrida para resolver os problemas de esparsidade e *cold-start*, a recomendação de itens para CS ainda é uma questão de pesquisa em aberta.

Entretanto, existem alguns métodos que não foram testados para esse tipo de problema, por exemplo, redes neurais sem peso, também conhecida como Wisard. Diferentemente das redes tradicionais, o processo de aprendizagem das redes neurais sem pesos é realizado através da modificação das palavras armazenadas na memória, permitindo a construção de algoritmos flexíveis e o aprendizado rápido. As entradas e a saída dos neurônios são números binários (0 ou 1) e não existem pesos entre os neurônios.

O presente trabalho propõe uma abordagem híbrida para o problema de *cold-start* itens em filtragem colaborativa para sistemas de recomendação. Utilizamos a técnica de *Locality-Sensitive Hashing* (LSH) com o método *MinMaxwise* para processamento de textos com intuído de encontrar as similaridades entre os itens. O LSH foi incorporado nas abordagens tradicionais da literatura e demais métodos de aprendizagem de máquina, gerando bons resultados para esses métodos.

1.2 Objetivos

Os objetivos deste trabalho são os seguintes:

- Apresentar uma abordagem híbrida, utilizando Wisard e Rede Neurais acrescidos de *Locality-Sensitive Hashing*, para ser aplicada ao problema de *cold-start* em filtragem colaborativa para sistema de recomendação;
- Analisar os atributos dos itens e usuários para definir quais são os mais relevantes no problema de *cold-start*. Aplicar *Locality-Sensitive Hashing* para o processamento das similaridades dos atributos adquiridos externamente;
- Comparar e analisar o desempenho do método proposto em relação as principais abordagens da literatura para filtragem colaborativa.

1.3 Contribuições

As contribuições deste trabalhos podem ser resumidas aos itens abaixo:

- Apresentar uma abordagem que permite a construção de algoritmos flexíveis, com o baixo custo computacional e de aprendizagem rápida;
- Apresentar uma abordagem com resultados significativos para o problema de *cold-start*;
- Criar uma alternativa para normalização de notas com melhor resultado, utilizando técnicas de *Locality-Sensitive Hashing*;
- Apresentar uma técnica de pré-processamento de texto eficiente para encontrar similaridade entre documentos para ser aplicado em uma abordagem híbrida.

1.4 Organização

Esta dissertação está organizada em 5 capítulos, dos quais este é o primeiro. No capítulo 2, descreve os principais conceitos teóricos envolvidos, além de mencionar os trabalhos relacionados ao tema. No capítulo 3, detalha como funciona a nossa proposta. No capítulo 4, apresenta a metodologia que guiou nossos experimentos, bem como as demais estratégias comparadas e expomos os resultados obtidos e suas análises. Finalmente, no capítulo 5, tiramos as devidas conclusões e apontamos possíveis trabalhos futuros.

Capítulo 2

Fundamentação teórica

Neste capítulo apresenta uma breve revisão bibliográfica sobre sistema de recomendação, aprendizado de máquina e *Locality-Sensitive Hashing*. Logo após, uma contextualização do estudo de redes neurais na perspectiva conhecida como Wisard. Essas revisões fornece uma base para o entendimento da metodologia usada e os experimentos realizados. Por fim, apresenta os principais trabalhos da literatura que utilizam as técnicas mencionadas no contexto dos sistemas de recomendação.

2.1 Sistema de recomendação

A origem dos sistemas de recomendação (RS) pode ser encontrada em diversas áreas de pesquisas em meados da década de 1990. Entretanto tornou-se uma importante área de pesquisa desde o surgimento dos primeiros trabalhos sobre filtragem colaborativa (GOLDBERG *et al.*, 1992, HILL *et al.*, 1995, RESNICK *et al.*, 1994, SHARDANAND e MAES, 1995). A dificuldade das pessoas em realizar uma escolha entre as várias alternativas que lhe são apresentadas dentro de uma gama de produtos e serviços ofertados foi agravada ainda mais devido ao fato de que a informação tornou-se disponível e difundida de maneira mais rápida com a evolução da internet.

Nesse contexto, o indivíduo encontra-se imerso à essa situação de excesso de informação e diversidade de opções. Muitas vezes uma pessoa possui pouca ou quase nenhuma experiência pessoal para realizar algumas escolhas entre as várias alternativas que lhe são apresentadas. Para minimizar as dúvidas e necessidades que se tem frente à escolha entre alternativas, geralmente confia-se nas recomendações, que podem ser passadas por outras pessoas, as quais podem chegar de forma direta ou por opiniões de especialistas, jornais com boa reputação, entre outros (SHARDANAND e MAES, 1995). Portanto, a evolução dos sistemas de recomendação partiu do fato de utilizarem grandes bases de informações, dos quais permitiram que as recomendações pudessem ser alcançadas de forma satisfatória, desta forma proporcionando maior credibilidade que uma recomendação humana.

Sistemas de recomendação são técnicas e ferramentas usadas para sugerir itens personalizados baseados nos interesses do usuário. Em um sistema comum, geralmente os usuários fornecem as recomendações. Essas informações capturadas são utilizadas pelo sistema para apresentá-las para os grupos de indivíduos considerados potenciais interessados para esse tipo de recomendação (RICCI *et al.*, 2015). A Figura 2.1 mostra o modelo conceitual de um RS.

Figura 2.1: Modelo conceitual de um sistema de recomendação.

O interesse nesta área ainda permanece alto, pois constitui uma área de pesquisa rica em problemas, devido à abundância de aplicações práticas que ajudam os usuários a lidar com a sobrecarga de informações e as recomendações especializadas, de conteúdo e serviços. Tanto na indústria quanto na academia há muito trabalho sendo feito no desenvolvimento de novas abordagens para recomendação. Exemplos de aplicações desenvolvidas que utilizam recomendação em seu sistema pode ser vista tanto na Netflix[1] para recomendar filmes, na Amazon[2] para produtos, no Spotify[3] para música, no Trivago[4] para hotéis, entre outros.

Os sistemas de recomendação possuem como objetivo a filtragem de informações de acordo com o perfil de interesse dos usuários a fim de recomendar itens que atendam às expectativas e necessidades dos mesmos quanto a informações. Porém, é comum que estas aplicações tragam muito conteúdo irrelevante (ADOMAVICIUS e TUZHILIN, 2005).

Em vista deste problema, os sistemas de recomendação surgiram com o foco na busca por informações relevantes de acordo com características do próprio usuário, bem como nos determinados requisitos relacionados aos itens que se quer encontrar.

Em sistemas de recomendação são utilizadas em geral uma das três técnicas de filtragem de informação citadas a seguir: filtragem baseada em conteúdo (CB), filtragem colaborativa (CF), também conhecida como filtragem social ou filtragem híbrida (FH).

[1]https://netflix.com/
[2]https://amazon.com/
[3]https://spotify.com/
[4]https://trivago.com/

2.1.1 Filtragem baseada em conteúdo

A abordagem baseada em conteúdo surgiu da área de recuperação de informação e busca analisar descrições de itens para identificar itens que possam ser, potencialmente, interessantes ao usuário em questão. Dessa forma, os itens podem ser representados de forma estruturada, com diversos atributos nomeados de diferentes tipos, ou não estruturada, com apenas uma descrição. Há também uma forma semi-estruturada, onde existem tantos atributos nomeados, como textos livres (PAZZANI e BILLSUS, 2007).

Devido aos significativos avanços feitos pelas comunidades de filtragem de informação e filtragem de conteúdo, muitos sistemas baseados em filtragem de conteúdo focam na recomendação de itens com informações textuais, como documentos e websites. As melhorias sobre os sistemas tradicionais de recuperação de informação vieram com a utilização do perfil do usuário, que contém suas preferências e necessidades.

A filtragem baseada em conteúdo, então, presume que o usuário se interesse por itens similares ao que já se interessou no passado, o que nos faz definir a similaridade entre os itens. Desta forma, a filtragem baseada em conteúdo parte do princípio de que os usuários tendem a interessar-se por itens similares aos que demonstraram interesse no passado, definindo então, a similaridade entre os itens (HERLOCKER, 2000).

Para realizar a recomendação, os recomendadores baseados em conteúdo costumam recomendar itens similares aos itens que o usuário preferiu no passado, vários itens serão comparados aos itens pelos quais o usuário avaliou positivamente e os que possuírem o maior nível de similaridade com estes serão recomendados (CAZELLA *et al.*, 2010).

Conforme mencionado anteriormente, Sistemas de Recomendação baseados em conteúdo podem recomendar itens similares a itens que o usuário gostou no passado. Deste modo, vários itens são comparados com itens que foram avaliados positivamente e os mais similares serão recomendados.

2.1.2 Medidas de Similaridade

Esta seção descreve sobre as técnicas de medição de similaridade, que são utilizados em métodos de filtragem colaborativa. A medida de similaridade ou semelhança, é uma função de valor real que quantifica a semelhança entre dois objetos. Embora não exista uma definição única de uma medida de similaridade, geralmente medidas de similaridade são, em certo sentido o inverso de métricas de distância, isto quer dizer que, elas assumem valores altos para objetos semelhantes e zero ou negativo para objetos muito diferentes.

2.1.2.1 Distância Euclidiana

Na matemática, a distância Euclidiana é a distância entre dois pontos, X e Y, que pode ser provada pela aplicação repetida do teorema de Pitágoras (DEZA e DEZA, 2009). Tendo como definição a Equação 2.1:

$$d(X,Y) = \sqrt{\sum_{i=1}^{N}(X_i - Y_i)^2} \tag{2.1}$$

onde N é o número de dimensões e i é os atributos de X e Y.

2.1.2.2 Distância de Minkowski

A distância de Minkowski é uma métrica num espaço vetorial normalizado que pode ser considerado como uma generalização da distância euclidiana e da distância de Manhattan (BAGCHI, 2015), sendo representada Equação2.2 a seguir:

$$d(X,Y) = (\sum_{i=1}^{N} |X_i - Y_i|^p)^{\frac{1}{p}} \tag{2.2}$$

onde p é o grau de distância. Dependendo do valor de p, a distância Minkowski é determinada com nomes específicos.

2.1.2.3 Similaridade do Cosseno

Similaridade do Cosseno é a medida de similaridade entre dois vetores de um espaço com produto interno que mede o cosseno do ângulo entre elas (LINDEN *et al.*, 2003). Dado dois vetores de atributos, X e Y, a similaridade é representada usando um produto escalar, conforme mostra a Equação 2.3:

$$\cos(X,Y) = \frac{(X \bullet Y)}{||X|| \cdot ||Y||} \tag{2.3}$$

onde $\bullet$ indica produto escalar dos vetores analisados e $||X||$ é a norma do vetor X.

2.1.2.4 Correlação de Pearson

O coeficiente de correlação de Pearson mede o grau da correlação e a direção dessa correlação, positiva ou negativa, entre duas variáveis de escala métrica (SHARDANAND e MAES, 1995). Este coeficiente, normalmente representado por ρ assume apenas valores entre -1 e 1.

- $\rho = 1$ Significa uma correlação perfeita positiva entre as duas variáveis.

- $\rho = -1$ Significa uma correlação negativa perfeita entre as duas variáveis - Isto é, se uma aumenta, a outra sempre diminui.
- $\rho = 0$ Significa que as duas variáveis não dependem linearmente uma da outra. No entanto, pode existir uma dependência não linear. Assim, o resultado $\rho = 0$ deve ser investigado por outros meios.

A formula é representada pela Equação 2.4 a seguir:

$$\rho(X, Y) = \frac{\sum(X, Y)}{\sigma_X \times \sigma_Y} \tag{2.4}$$

onde $\sum$ é a covariância de pontos dos dados X e Y e σ é o desvio padrão.

2.1.2.5 Correlação de Spearman

Correlação de Spearman é uma medida de correlação não-paramétrica, isto é, ele avalia uma função monótona arbitrária que pode ser a descrição da relação entre duas variáveis, sem fazer suposições sobre a distribuição de frequências das variáveis. Não requer a suposição que a relação entre as variáveis é linear, nem requer que as variáveis sejam medidas em intervalo de classe. Pode ser usado para as variáveis medidas no nível ordinal (DEZA e DEZA, 2009). Sua formula é definida pela Equação 2.5.

$$\mathcal{P} = 1 - \frac{6 \sum d_i^2}{(n^3 - n)} \tag{2.5}$$

onde d_i é igual a diferença entre cada posto de valores correspondentes de X e Y, e n é igual ao número dos pares dos valores.

2.1.2.6 Coeficiente de Tanimoto

Uma relação de similaridade é dada sobre *bitmaps*, em que cada bit de uma matriz de tamanho fixo representa a presença ou ausência de uma característica no modelo. A definição da taxa é o número de bits comuns, dividido pelo número de bits definidos (BAGCHI, 2015).

Apresentado em termos matemáticos, se as amostras de X e Y são mapas de bits, X_i é o i-ésimo bit de X, $\wedge$, $\vee$ são os operadores de bit a bit "e", "ou"respectivamente, então a relação de similaridade T_s é dada a seguir pela Equação 2.6:

$$T_s(X, Y) = \frac{\sum_i (X_i \wedge Y_i)}{\sum_i (X_i \vee Y_i)} \tag{2.6}$$

Portanto o coeficiente de Tanimoto é definido pela Equação 2.7:

$$T_d(X, Y) = -log_2(T_s(X, Y)) \tag{2.7}$$

2.1.2.7 Similaridade de Verossimilhança

Similaridade de Verossimilhança é um teste estatístico utilizado para comparar e ajustar dois modelos. Baseia-se na razão de probabilidade, que expressa o número de vezes mais provável que os dados estão sob um modelo do que o outro (ARSAN *et al.*, 2016). É definida pela Equação 2.8.

$$D = -2\ln(\frac{modelo1}{modelo2}) = -2\ln(modelo1) + 2\ln(modelo2) \tag{2.8}$$

2.1.3 Filtragem colaborativa

Ela se diferencia da filtragem baseada em conteúdo exatamente por não exigir a compreensão ou reconhecimento do conteúdo dos itens. Constitui-se em uma das mais populares técnicas de recomendação, sendo utilizada em muitos sistemas existentes na Internet (SCHAFER *et al.*, 2001). Nos sistemas colaborativos, a essência está na troca de experiências entre as pessoas que possuem interesses comuns. Nestes sistemas, os itens são filtrados baseada nas avaliações feitas pelos usuários.

Métodos de filtragem colaborativa analisam grande quantidades de informações sobre as preferências dos utilizadores e preveem as preferências de usuários semelhantes para recomendar itens. Sendo assim, é possível realizar uma previsão precisa das preferências de um usuário e realizar recomendações de itens sem qualquer necessidade de uma análise detalhada das características do item. A técnica se baseia na análise de preferências comuns em um grupo de pessoas (CAZELLA *et al.*, 2009) e sua essência é a troca de experiências entre pessoas que tem interesses em comum e possuem escolhas semelhantes por itens (REATEGUI e CAZELLA, 2005).

Uma das vantagens apresentadas por esse método é a possibilidade de apresentar aos usuários recomendações de itens que não estavam sendo pesquisados de forma ativa, o que torna possível que os mesmos possam receber recomendações inesperadas. Outra contribuição importante dos sistemas de filtragem colaborativa se refere à possibilidade de formação de comunidades de usuários pela identificação de seus gostos e interesses similares (REATEGUI e CAZELLA, 2005).

Entretanto, os métodos de filtragem colaborativa sofrem de problemas como *cold-start*, escalabilidade e dispersão, visto que na matriz de *ratings*, apenas uma pequena porcentagem de elementos obtém valores. Mesmo os itens mais populares podem ter apenas algumas avaliações. Com uma matriz de *ratings* esparsa, é muito difícil estimar a relação entre itens e usuários para realizar uma recomendação eficaz. A abordagem de CF exige um grande número de avaliações de um usuário ou de um item, para obter uma predição eficaz, o que não funcionará para novos usuários, novos itens ou ambos devido a poucas classificações disponíveis no sistema.

Existem três classes de CF que são: filtragem colaborativa baseada em memória,

filtragem colaborativa baseada em modelo e filtragem colaborativa híbrida apresentadas a seguir.

2.1.3.1 Filtragem colaborativa baseada em memória

A filtragem colaborativa baseada em memória utiliza toda a base de dados que contém a relação entre usuários u e itens i para calcular a similaridade entre usuários ou itens e assim fazer as predições ou recomendações. Por causa da utilização da similaridade como heurística esse algoritmo também é chamado na literatura como algoritmo baseado em heurística (ADOMAVICIUS e TUZHILIN, 2005). Exemplos típicos dessa abordagem são as recomendações top-N baseadas em vizinhança e baseadas em itens ou usuários.

As abordagens baseadas no usuário foi usada inicialmente em muitos sistemas comerciais. Uma matriz usuário-item R de dimensão n x m pode ser utilizada para representar a relação dos n usuários sobre os m itens. Cada campo, $r_{i,j}$ da matriz R pode conter: valores de avaliações, ou os valores zero ou um. São métodos que tentam prever a preferência de um usuário u para um item i, ou seja r'_{ui} o valor do *rating* que o usuário u dá ao item i é calculado como um somatório dos *ratings* de alguns usuários semelhantes do item, conforme a Equação 2.9. Um exemplo deste somatório pode ser dada pela Equação 2.10.

$$r_{u,i} = \sum\nolimits_{u' \in U} r_{u',i} \tag{2.9}$$

Para cálculo da similaridade são atribuídos pesos a todos os usuários indicando seu grau de similaridade com o usuário ativo. A similaridade é baseada nas notas que foram avaliadas em comum, ou seja, co-avaliadas entre dois usuários ou dois itens (ADOMAVICIUS e TUZHILIN, 2005). As medidas de similaridades mais conhecidas são: coeficiente de correlação de *Pearson* e a similaridade do cosseno, cujo as fórmulas são mostrada pelas Equações 2.4 e 2.3, respectivamente.

Vale ressaltar que, na recomendação utilizando a abordagem de vizinhos mais próximos, a similaridade é a principal técnica a ser analisada, sendo assim, responsável pela seleção dos vizinhos mais confiáveis que serão utilizados para a previsão e desta forma fornecer o grau de importância desses vizinhos (RICCI *et al.*, 2015). Devido a esses fatores, a escolha do método de similaridade impacta no desempenho do algoritmo de recomendação e nas suas predições.

Após a definição da similaridade de dois usuários u_1 e u_2 como sim_{u_1,u_2} e supondo que o item i já foi avaliado por um conjunto de usuários é definido um $K_i(u_1)$ como k vizinhos mais próximos do usuário u_1 que avaliaram o item i. Desta forma, a previsão será dada pela média ponderada das notas dos usuários similares a u_1 conforme mostra a Equação 2.10.

$$r'_{u_1,i} = \frac{\sum\limits_{u_2 \in K_i(u_1)} \text{sim}(u_1, u_2) r_{u_2 i}}{\sum\limits_{u_2 \in K_i(u_1)} |\,\text{sim}(u_1, u_2)|} \tag{2.10}$$

As abordagens baseadas nos itens são métodos que tentam prever a preferência de um item i para usuário u, ou seja, r'_{ui}. Definindo a similaridade de dois itens i_1 e i_2 como sim_{i_1,i_2} e supondo que o usuário u já avaliou diversos itens, é definido um $K_u(i_1)$ como k vizinhos mais próximos do item i_1 que foram avaliados pelo usuário u. Desta forma, a previsão será dada pela média ponderada das notas dos itens similares a i_1 como mostra a Equação 2.11.

$$r'_{u,i_1} = \frac{\sum\limits_{i_2 \in K_u(i_1)} \text{sim}(i_1, i_2) r_{u,i_2}}{\sum\limits_{i_2 \in K_u(i_1)} |\,\text{sim}(i_1, i_2)|} \tag{2.11}$$

Um problema encontrado está relacionado aos vários critérios adotados pelos usuários para realizar as avaliações dos itens, ou seja, o significado do conjunto de preferência possui relações diferentes para cada usuário. Desta forma, prejudica a previsão tanto nas Equações 2.10 e 2.11. Para solucionar esse problema utiliza as notas normalizadas $h(r_{i_1 i_2})$ para realizar a previsão (RICCI *et al.*, 2015).

A forma de normalização mais comum relatada na literatura é a média central. Tem como objetivo determinar se o *rating* é positivo ou negativo em relação à avaliação média do conjunto de *ratings*, ou seja, o algoritmo irá prever esse desvio em vez da própria classificação (RICCI *et al.*, 2015). As Equações 2.12 e 2.13 apresentam a normalização pela média central baseados nos usuários e itens, respectivamente.

$$r'_{u_1,i} = \bar{r}_{u_1} + \frac{\sum\limits_{u_2 \in K_i(u_1)} \text{sim}(u_1, u_2)(r_{u_2 i} - \bar{r}_{u_2})}{\sum\limits_{u_2 \in K_i(u_1)} |\,\text{sim}(u_1, u_2)|} \tag{2.12}$$

$$r'_{u,i_1} = \bar{r}_{i_1} + \frac{\sum\limits_{i_2 \in K_u(i_1)} \text{sim}(i_1, i_2)(r_{u,i_2} - \bar{r}_{i_2})}{\sum\limits_{i_2 \in K_u(i_1)} |\,\text{sim}(i_1, i_2)|} \tag{2.13}$$

A CF baseada em memória possui algumas limitações como o problema da esparsidade, pois diante da grande quantidade de produtos disponíveis principalmente em grandes lojas como a Amazon[5], Netflix[6] entre outras, usuários ativos compram ou avaliam poucos itens. Logo, a matriz usuário-item que representa as transações dos clientes, possuirá muitas células vazias, dificultando assim fazer as associações entre os usuários ou itens e as recomendações produzidas não serão tão precisas.

[5]https://amazon.com/
[6]https://netflix.com/

Outra limitação é a escalabilidade, pois algoritmos baseados no usuário requerem a computação de milhares de produtos e clientes, que crescem o tempo todo. Assim, ao ter que responder às solicitações de milhares de usuários ao mesmo tempo, um sistema de recomendação pode sofrer sérios problemas de escalabilidade.

2.1.3.2 Filtragem colaborativa baseada em modelo

Nos algoritmos de CF baseados em modelo, os dados sobre os usuários e os itens, o conjunto de *ratings*, são utilizados para criar um modelo matemático, que por sua vez é usado para prever a nota que um usuário u daria a um item i. Assim, o aprendizado de máquina como: redes bayesianas, que formulam um modelo probabilístico para CF; clusterização, que trata a CF como um problema de classificação; métodos baseados em regras, que aplicam algoritmos que descobrem regras de associação entre itens previamente adquiridos gerando a recomendação baseada na força da associação entre os itens, redução de dimensionalidade, entre outras técnicas, são normalmente aplicados para criar um modelo que melhore a precisão sobre os métodos regulares baseados em memória (SARWAR *et al.*, 2001).

Assim como a filtragem colaborativa baseada em memória, a filtragem baseada em modelo também pode ser divida em abordagem baseada no usuário e abordagem baseada no item. O método baseado no usuário constrói um modelo com base no usuário, e para isso pode utilizar diferentes técnicas, como por exemplo clusterização, onde usuários similares são agrupados em um mesmo *cluster*. Este modelo é então utilizado para estimar a probabilidade de que o usuário alvo pertença a determinado *cluster* X, que pode então ser usado para fazer predições de avaliações para o usuário alvo. Já no método baseada no item, a construção do modelo é com base nos itens, utilizando medidas de similaridade como a probabilidade condicional BREESE *et al.* (1998), conforme mostra a Equação 2.14.

$$p_{u,i} = E(r'_{u,i}) = \sum_{r=0}^{m} Pr(r'_{u,i} = r | r'_{u,k}, k \in I_u) r \tag{2.14}$$

onde r é uma variável aleatória inteira no intervalo entre 0 e m e $p_{u,i}$ é a probabilidades condicionais de um usuário u dar uma nota a um item i.

Em (BREESE *et al.*, 1998), os autores desenvolveram dois modelos probabilísticos para filtros colaborativos baseados em modelos. O primeiro utiliza clusterização, onde os usuários são agrupados em classes e suas avaliações são independentes, e o segundo modelo utiliza redes de Bayes, onde o estado de cada nodo corresponde a possíveis valores de avaliação para cada item.

O modelo de rede bayesiana é uma rede na qual cada nó corresponde a cada item no domínio e seu estado representa os diferentes valores que a classificação atribuída pelo usuário pode assumir (BREESE *et al.*, 1998).

O conceito de clusterização continuou a ser amplamente explorado. Em UNGAR e FOSTER (1998) foi proposto um modelo estatístico onde cada usuário pertence a uma dada classe e vários algoritmos, como K-means e Gibbs, foram comparados para estimar os parâmetros do modelo. No trabalho de (CHIEN *et al.*, 1999) a proposta era um modelo Bayesiano que buscava agrupar os usuários do sistema em grupos onde as classificações seguem distribuições de probabilidade similares. Para estimar as classificações ausentes neste modelo, foi utilizado um algoritmo de busca híbrida em conjunto com uma Cadeia de Markov (SHANI *et al.*, 2005).

A recomendação baseada em modelos é basicamente voltada para a personalização do conteúdo a ser entregue, pois antes da recomendação é feita uma análise do comportamento do usuário para aprender o modelo do seu perfil. Desta forma, recomendação baseada em modelo consegue superar o problema da escalabilidade presente na CF baseada em memória, pois quando uma recomendação é requisitada, as relações de similaridade entre os usuários ou entre os itens podem ser encontradas através do modelo previamente construído e não calculada em tempo real, como ocorre na CF baseada em memória.

Exitem alguns algoritmos que utilizam ambas as filtragem colaborativas, baseadas em memória e baseadas em modelo denominada Filtragem Colaborativa Híbrida, podem ser encontrados em (PENNOCK *et al.*, 2000, XUE *et al.*, 2005). A combinação consegue superar o problema da esparsidade, conforme mostrado em (XUE *et al.*, 2005) aumentando a eficiência e precisão das recomendações. Porém o sistema de recomendação que utiliza algoritmos baseado em memória e algoritmos baseado em modelo possuem um problema denominado *Cold-Start*.

2.1.4 Filtragem híbrida

As técnicas híbridas apresentam em sua implementação o uso de dois ou mais algoritmos de recomendação trabalhando em conjunto para gerar predições ou recomendações. Esta junção tem como objetivo realçar o potencial de cada algoritmo e desta forma, compensar as possíveis deficiências quando utilizados sozinhos. Geralmente, essas técnicas contam com uso de recomendações baseadas em conteúdo para reduzir as limitações das técnicas puramente baseadas em filtragem colaborativa. Porém existem casos do uso de diferentes técnicas de filtragem colaborativa em conjunto, como por exemplo a técnica de CF baseada em memória junta com a técnica de CF baseada em modelo (ADOMAVICIUS e TUZHILIN, 2005).

Segundo ADOMAVICIUS e TUZHILIN (2005) existem diferentes maneiras de combinar o método baseado em conteúdo e de filtragem colaborativa em um sistema de recomendação híbrido, podendo ser classificados da seguinte forma:

- Implementar os métodos CB e CF separadamente e combinar suas previsões;

- Combinar algumas características do método CB em uma abordagem CF;
- Combinar algumas características do método CF em uma abordagem CB;
- Construir um modelo unificado que incorpore características das duas abordagens.

Desta forma, podemos ter dois cenários. Primeiro, podemos combinar as saídas das classificações obtidas pelas abordagens individuais em uma recomendação final usando uma combinação linear de classificações (CLAYPOOL *et al.*, 1999) ou utilizar uma estrutura de votação (PAZZANI, 1999). Entretanto, existem outras possibilidades, como por exemplo, usar uma das abordagens individuais, a qualquer momento, escolhendo usar aquela que contenha a melhor predição com base em alguma métrica de avaliação.

Muitos artigos comparam o desempenho de uma recomendação com abordagem híbrida que utiliza-se de filtragem colaborativa e métodos baseados em conteúdo e mostram que os recomendadores híbridos podem levar a melhor acurácia nas recomendações, especialmente para situações de novo usuário ou novo item onde a filtragem colaborativa não traz recomendações satisfatórias (BURKE, 2002, KHAN *et al.*, 2017, KIM *et al.*, 2006, SOBOROFF e NICHOLAS, 1999). Entretanto, recomendadores híbridos dependem de informação externa que usualmente não está disponível, e possuem geralmente maior dificuldade e complexidade para a implementação (ADOMAVICIUS e TUZHILIN, 2005).

2.1.5 Problema de *Cold-Start*

O problema *Cold-Start* (CS) é um dilema de longa data em sistemas de recomendação. Ocorre quando existem a indisponibilidade de informações adequadas sobre os itens ou usuários disponíveis no sistema, e desta forma não é possível fazer recomendações relevantes (PARK e CHU, 2009). Esse problema pode ser dividido em *Cold-Start User* e *Cold-Start Item* descritos a seguir.

O *Cold-Start User* acontece quando não há nenhuma classificação do usuário. Por exemplo, a partir da matriz de *ratings* conforme mostra a Figura 2.2a, a abordagem de CF consegue identificar que os usuários u_1 e u_4 avaliaram os itens i_1, i_2 e i_4, logo esses usuários tem preferências em comum. Através desse conhecimento obtido, é possível recomendar o item i_5 para o usuário u_4, pois este item já foi avaliado por um usuário com gostos semelhantes ao dele. Já a Figura 2.2b mostra que o usuário u_3 não avaliou nenhum item, portanto a abordagem de CF não consegue encontrar relações ente os demais usuários, ocorrendo então o problema de *Cold-Start User*.

O problema *Cold-Start Item* ocorre quando pouco ou nenhum usuário realiza uma avaliação, para determinado item. Por exemplo, a partir da matriz de *ratings*

	i_1	i_2	i_3	i_4	i_5
u_1	3	5	?	3	4
u_2	?	3	?	?	5
u_3	?	?	?	?	?
u_4	5	5	?	3	?
u_5	4	?	?	4	?

(a) u_1 e u_4

	i_1	i_2	i_3	i_4	i_5
u_1	3	5	?	3	4
u_2	?	3	?	?	5
u_3	?	?	?	?	?
u_4	5	5	?	3	?
u_5	4	?	?	4	?

(b) *Cold-Start User*

Figura 2.2: Relação entre os usuários u_1 e u_4 e o problema de *Cold-Start User*

conforme mostra a Figura 2.3a, a abordagem de FC consegue identificar a relação entre os itens i_1 e i_4, pois os usuários u_1, u_4 e u_5 sempre que avaliaram o item i_1 também avaliaram o item i_4. Porém se o item i_4 nunca tivesse sido avaliado não seria possível encontrar essa relação do item i_1 com o item i_4. Esse exemplo mostra que o item i_4, juntamente com o item i_3 sofre do problema *Cold-Start Item*, visto na Figura 2.3b.

	i_1	i_2	i_3	i_4	i_5
u_1	3	5	?	3	4
u_2	?	3	?	?	5
u_3	?	?	?	?	?
u_4	5	5	?	3	?
u_5	4	?	?	4	?

(a) i_1 e i_4

	i_1	i_2	i_3	i_4	i_5
u_1	3	5	?	?	4
u_2	?	3	?	?	5
u_3	?	?	?	?	?
u_4	5	5	?	?	?
u_5	4	?	?	?	?

(b) *Cold-Start Item*

Figura 2.3: Relação entre os itens i_1 e i_4 e o problema de *Cold-Start Item*

Os problemas citados anteriormente podem ser subdivididos ainda mais. Desta forma propicia então uma definição do completo *cold-start* (CCS), que representa os itens ou usuários que não receberam ou fizeram nenhuma avaliação, e o incompleto *cold-start* (ICS), quando o número de registros das avaliações tanto dos itens e/ou usuários são baixas. A Figura 2.4 apresenta uma ilustração simplificada da classificação dos itens entre CCS, ICS e não CS em sistemas de recomendação.

Na literatura, vários pesquisadores abordaram maneiras de como coletar informações ausentes para o problema de CS (ELAHI *et al.*, 2014, FERNÁNDEZ-TOBÍAS *et al.*, 2016, ZHANG *et al.*, 2016). Estas soluções, baseadas na natureza da sua coleta de dados ausentes, podem ser classificadas em duas classe: soluções explícitas e soluções implícitas (GOPE e JAIN, 2017). A maneira implícita de coletar informações inclui a navegação e os históricos da compra dos usuários, seus

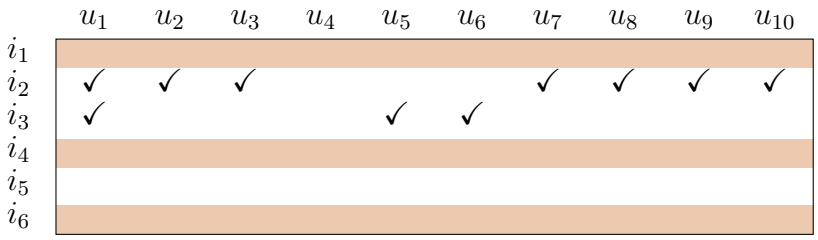


Figura 2.4: Representação dos itens com problema de CS, onde ✓representa itens avaliados. Item i_2 não sofre com CS, item i_3 ICS e item i_5 CCS.

dados demográficos entre outros. A maneira explícita pede aos usuários que avaliem itens ou preencham questionários (GOPE e JAIN, 2017).

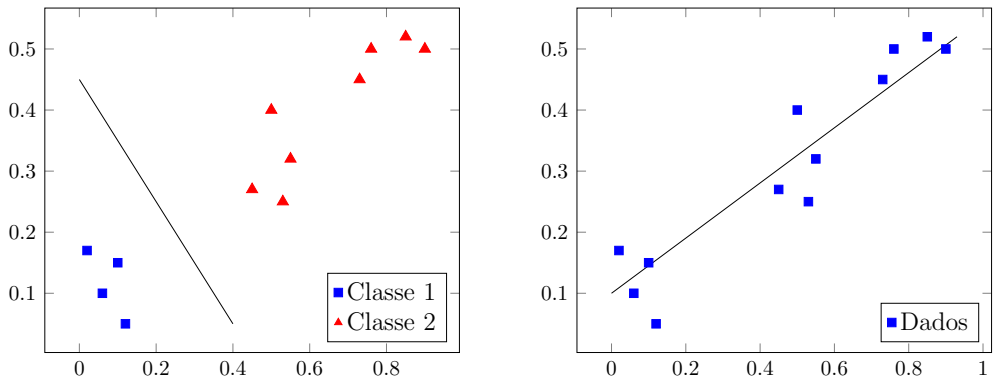
2.2 Aprendizado de máquina

O aprendizado de máquina cresceu como um sub-campo em Inteligência Artificial, no qual o objetivo principal é fazer com que as máquinas aprendam com os dados. As primeiras ideias surgiram na década de 50, quando TURING (1950) descreve o conceito de máquinas de aprendizagem, embora na época não tivesse sido proposto nenhum modelo de como construir tais máquinas. A ideia geral era que essas máquinas teriam algumas regras básicas de operação que diriam como reagiriam a situações específicas. Portanto, essas máquinas devem ser capazes de aprender (MITCHELL, 1997). Isso é um pouco semelhante ao que existe hoje em relação aos modelos treinados construídos com os atuais algoritmos de aprendizado de máquina.

Os algoritmos que são conhecidos hoje na comunidade de aprendizado de máquina teveram sua origem nos anos setenta e oitenta com a ascensão de sistemas especialistas e o algoritmo de *back propagation*. Estes algoritmos podem ser agrupados de acordo com o seu método de aprendizagem: aprendizado supervisionado, aprendizado não supervisionado, aprendizado semi-supervisionado e aprendizado por reforço (RUSSELL e NORVIG, 1995).

No aprendizado supervisionado, a máquina é apresentada com alguns exemplos do que deve ser aprendido, consiste basicamente de pares de entrada e saída. Em seguida, aprende uma função $y = h(x)$ que mapeia valores da entrada x para a saída y. Se a função $h(x)$ é discreta, então a tarefa de aprendizado supervisionado é chamada de classificação, conforme mostra a Figura 2.5a, caso contrário, se for contínua, a tarefa de aprendizado é chamada de regressão, conforme mostra a Figura 2.5b (MITCHELL, 1997).

No aprendizado não supervisionado, também conhecido como aprendizado por observação e descoberta, desde o início nenhum exemplo rotulado é fornecido e o algoritmo deve descobrir os padrões dentro do conjunto de dados fornecido, agrupando-



(a) Classificação (b) Regressão

Figura 2.5: Exemplo de aprendizado para classificação e regressão.

os de acordo com suas características. Os algoritmos mais comuns para aprendizado não supervisionado são o de clusterização, hierárquico e mapas auto-organizadores, onde os dados de entrada são agrupados por similaridade de uma determinada propriedade (DECKER e FOCARDI, 1995).

O aprendizado semi-supervisionado fica em uma região entre o aprendizado supervisionado e o não supervisionado. Alguns exemplos rotulados são fornecidos, mas a máquina deve descobrir por si própria quando novas etiquetas devem ser criadas, considerando a similaridade de algumas propriedades (ZHU e GOLDBERG, 2009). Esse tipo de aprendizado é útil quando a lista de rótulos para as instâncias não é completa ou não é totalmente conhecida.

O aprendizado por reforço é o tipo de aprendizado em que uma máquina é recompensada ou punida pelas decisões tomadas durante o processo de aprendizado.

2.2.1 *k-Nearest Neighbors*

O método *k-Nearest Neighbors* (KNN, k-vizinhos mais próximos) é considerado um dos métodos de classificação mais antigos e simples. Consiste em atribuir uma classe ao elemento desconhecido usando as informações dos vizinhos mais próximos (COVER e HART, 1967). Para calcular os vizinhos mais próximos de um elemento é necessário definir uma medida de similaridade. Existem diferentes funções para o cálculo desta similaridade, como foi mostrada na Seção 2.1.2. A similaridade é um ponto que possui grande importância para o método KNN, pois ela está diretamente relacionada ao desempenho deste método (HASTIE *et al.*, 2009).

O KNN pode ser usado tanto para classificação como regressão. Na classificação, o elemento é classificado por uma votação majoritária de seus vizinhos, com o elemento sendo atribuído à classe mais comum entre seus vizinhos mais próximos. Já na regressão, o valor previsto é a média dos valores de seus k vizinhos mais próximos (HASTIE *et al.*, 2009). A Figura 2.6 mostra um exemplo de KNN.

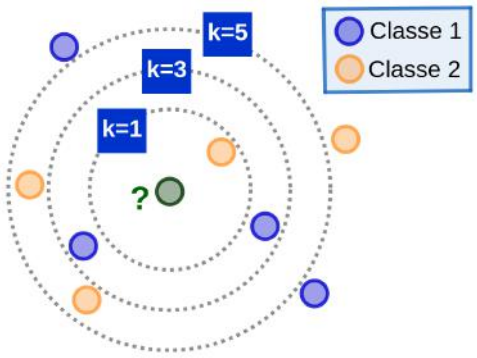


Figura 2.6: Exemplo de KNN com k valendo 1, 3 e 5.

Como pode ser visto pela Figura 2.6, a escolha do tamanho do k tem influência no desempenho do método KNN, portanto a definição do valor mais adequado para k é fundamental. De um modo em geral, para determinar o valor de k onde o método KNN atinja o melhor desempenho é necessário realizar diversos experimentos escolhendo diferentes valores para k.

2.2.2 Rede neurais

Redes neurais artificiais são um modelo computacional composto por interconexões chamadas de neurônios. Representa uma analogia às redes neurais biológicas e tem como principal função simular computacionalmente o funcionamento destas, onde cada neurônio gera a entrada para o outro. Basicamente, os neurônios são agrupados, constituindo assim uma rede neuronal cuja complexidade estrutural é variável, de acordo com a arquitetura utilizada. Dessa forma, tarefas dos mais diversos graus de complexidade podem ser executadas por uma rede neural.

A origem deste modelo foi através dos trabalhos de MCCULLOCH e PITTS (1943) que tentaram modelar um neurônio artificial. Em seguida, o Perceptron, um modelo simples de neurônio artificial para classificação binária, capaz de expressar classes linearmente separáveis, foi criado por ROSENBLATT (1958). A Figura 2.7 mostra o modelo de um neurônio, que forma a base para o projeto de redes neurais artificiais.

A arquitetura de uma rede neural desse tipo é definida pelo padrão de ligações entre seus neurônios, enquanto que a determinação dos pesos sinápticos fica a cargo do algoritmo de treinamento ou aprendizado utilizado (HAYKIN, 2008). Assim podemos dizer que um perceptron é um modelo que recebe um vetor de entrada no valor de x_1 a x_n e calcula uma combinação linear dessas entradas. Desta forma, a saída $u_k(x_1, ..., x_n)$ será 1 se o valor resultante estiver acima de um certo limite e -1 caso contrário, onde cada w_{ki} é o valor do peso determinado para cada *input* x_i, interferindo no valor obtido na saída do algoritmo.

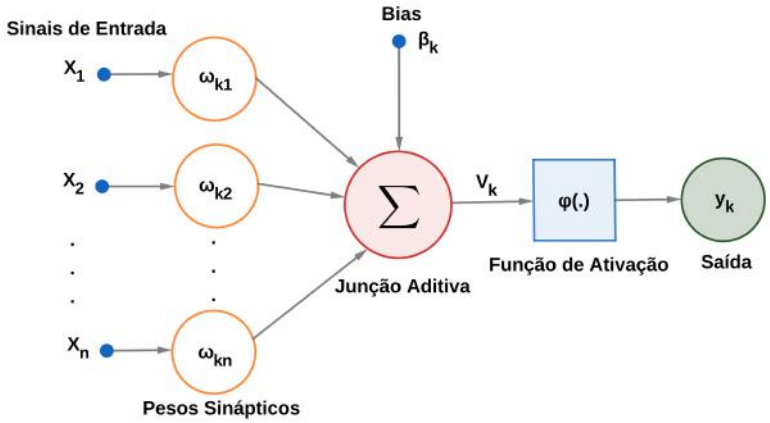


Figura 2.7: Modelo de um neurônio.

O modelo neural da Figura 2.7 inclui um *bias*, representado por β_k, têm como objetivo aumentar ou diminuir a entrada líquida da função de ativação $\varphi(.)$, em termos matemáticos podemos descrever um neurônio k pelas Equações 2.15 e 2.16.

$$u_k = \sum_{i=1}^{n} \omega_{ki} x_i \tag{2.15}$$

$$y_k = \varphi(u_k + \beta_k) \tag{2.16}$$

O uso do bias b_k tem o efeito de aplicar uma transformação afim à saída u_k, dada pela Equação 2.17, como mostra a Figura 2.7.

$$V_k = \varphi(u_k + \beta_k) \tag{2.17}$$

A Figura 2.8 apresenta a estrutura básica de um *perceptron* de múltiplas camadas (MLP). Dado que a capacidade computacional de um neurônio é limitada foi pensado que um conjunto de neurônios artificiais interligados em forma de uma rede é capaz de resolver problemas de alta complexidade. Os *perceptrons* de múltiplas camadas foram aplicados com sucesso para solucionar diversos tipos de problemas, através de seu treinamento de forma supervisionada utilizando-se da retropropagaçao de erro (*error back-propagation*). Segundo HAYKIN (2008) esta abordagem com múltiplas camadas detêm três características distintas:

- Possui uma função de ativação não-linear, podendo ser definida pela *sigmóide* apresentada pela Equação 2.18.

$$y_j = \frac{1}{1 + exp(-v_j)} \tag{2.18}$$

  onde o v_j é o campo local induzido do neurônio j e y_j é a saída do neurônio. A função logística procura levar em conta fase refratária de neurônios reais.

- A rede contém uma ou mais camadas ocultas, das quais não pertence as camadas de entrada e saída da rede. Esses neurônios ocultos fazem a rede a aprender tarefas complexas extraindo as características mais significativas dos padrões de entrada.

- A rede possui um alto grau de conectividade, determinada pelas sinapses. Dado uma alteração nesta conectividade consequentemente ocorre uma mudança na população das conexões sinápticas ou de seus pesos.

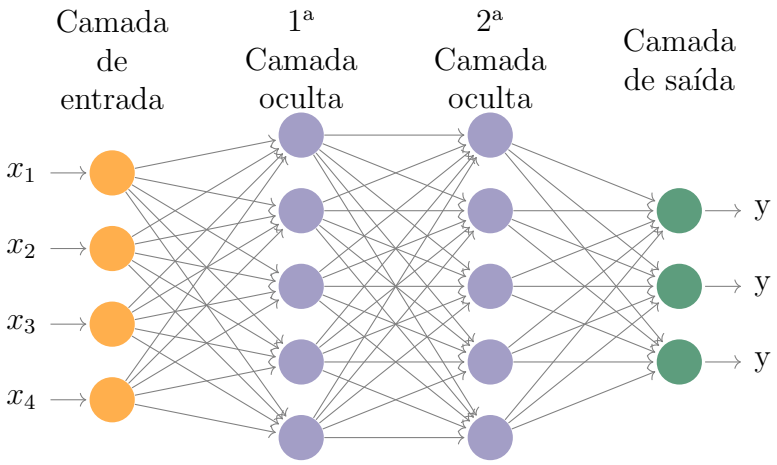


Figura 2.8: Arquitetura de um *perceptron* de múltiplas camadas.

2.2.3 *Deep learning*

O aprendizado de representações complexas de alto nível a partir de dados brutos é um problema não trivial e desafiador nos dias atuais (LEE *et al.*, 2009). Com base nesse problema, surgiram diversos métodos baseados em *deep learning*, que buscam aprender características hierárquicas, onde características de níveis mais altos são formadas a partir de uma composição de características de níveis mais baixos, de dados brutos. Ou seja, esse tipo de aprendizado automático em diferentes níveis de abstração permite o aprendizado de funções complexas que mapeiam os dados brutos diretamente sem a necessidade de um especialista humano (BENGIO, 2009).

O conceito de *deep learning* se originou de diversas tentativas falhas de treinar Perceptrons Multicamadas (MLP) com diversas camadas ocultas, denominadas Redes Neurais Multi-Camada Profundas, tendo como base a profundidade arquitetural do cérebro humano (BENGIO, 2009). Entretanto, sozinho, o algoritmo *Backpropagation*, comumente utilizado para treinar esse tipo de rede, não funciona bem para o aprendizado de redes com mais do que um número pequeno de camadas ocultas. A presença generalizada de ótimos locais e desafios de otimização na função objetivo

não-convexa tornaram-se os entraves para o uso desse tipo de aprendizado (DENG e YU, 2014).

Somente em 2006, com a introdução das *Deep Beliefs Networks* (HINTON *et al.*, 2006), composta do empilhamento de *Restricted Boltzmann Machines* (RBM) e utilizando um algoritmo de aprendizagem que treina uma camada de cada vez, bons resultados começaram a ser obtidos. Posteriormente, explorando o mesmo princípio de aprendizagem, foram propostos algoritmos baseados em Autoencoders esparsos (BENGIO, 2009).

Desde então, técnicas de *deep learning* têm sido utilizadas com sucesso em diferentes tipos de tarefas, como classificação (VINCENT *et al.*, 2008), regressão (SALAKHUTDINOV e HINTON, 2008), redução de dimensionalidade (HINTON e SALAKHUTDINOV, 2006), detecção de rostos (LE *et al.*, 2013), processamento de linguagem natural (VISHNUBHOTLA *et al.*, 2010), recuperação de informação (SALAKHUTDINOV e HINTON, 2009) e filtragem colaborativa (SEDHAIN *et al.*, 2015b).

O aprendizado baseado em *deep learning* está em interseções entre as áreas de redes neurais, inteligência artificial, modelagem gráfica, otimização, reconhecimento de padrões de pesquisa e processamento de sinais (DENG e YU, 2014).

2.2.3.1 *Dropout*

O Dropout serve como uma forma de regularização, ou seja, ele intencionalmente desativa neurônios da rede com o intuito de reajustar as saídas das camadas prevenindo assim o overfitting. Podem ser utilizados após qualquer camada da rede. Com isso ele permite efetuar de forma simplificada uma média de valores regularizando a saída da camada. A utilização da camada de dropout ganhou popularidade pela sua utilização em Rede Neurais Convolucionais.

2.2.3.2 *Dense*

Nas Camadas Densas são aplicados o conceito de Backpropagation. Nessas camadas o principal objetivo é relacionar os parâmetros aprendidos pela rede com a saída esperada e ajustar os parâmetros para corrigir erros. Funcionam como uma função onde existe uma entrada que são os parâmetros da Rede e uma Saída que deve ser a saída esperada inicialmente. Por ser supervisionado ele compara a saída dessa função com a esperada e corrige os parâmetros da Rede para que se adequem.

2.2.3.3 *Batch normalisation*

Batch Normalisation é uma técnica para melhorar o desempenho e a estabilidade das redes neurais, além de tornar as arquiteturas de aprendizado profundo mais so-

fisticadas na prática. Tem dois benefícios potenciais: pode acelerar a aprendizagem, pois permite que você empregue taxas de aprendizado mais elevadas e regulariza a aprendizagem. A ideia foi proposta por (IOFFE e SZEGEDY, 2015), ao invés de simplesmente normalizar as entradas para a rede, normalizava as entradas para as camadas internas da rede.

2.2.3.4 *Embedding*

Embedding é uma representação da entrada de dados em um espaço vetorial. Por exemplo, mapeando imagens e suas descrições textuais em um espaço comum de incorporação e minimizando a distância entre elas, podemos combinar etiquetas com imagens.

2.3 Rede neurais sem peso

Diferentemente das rede tradicionais, o processo de aprendizagem das redes neurais sem pesos é realizado através da modificação das palavras armazenadas na memória, permitindo a construção de algoritmos flexíveis e o aprendizado rápido. As entradas e a saída dos neurônios são números binários (0 ou 1) e não existem pesos entre os neurônios. Para o treinamento de uma rede neural sem peso é preciso apenas incrementar o valor do conteúdo de cada RAM, endereçado pelos n-bits da entrada. Em uma rede sem treinamento, todos os endereços de cada RAM, são inicializadas com zeros.

2.3.1 WiSARD

A rede neural WiSARD (Wilkie, Stonham & Aleksander's Recognition Device) foi criada na década de 80 por ALEKSANDER *et al.* (1984). É uma rede neural sem pesos onde os neurônios são implementados em memórias (RAM). Teve como finalidade o reconhecimento de imagens. A forma de implementação consegue apresentar uma vantagem de ser treinada em um tempo muito curto, essencial para as aplicações que necessitam de escalabilidade.

WiSARD é uma rede neural destinada para o reconhecimento de padrões, originalmente concebido para implementação de hardware, mas também pode ser utilizada com outros fins, como por exemplo o controle automático dos movimentos de uma plataforma offshore (FRANÇA *et al.*, 2010), a minimização do problema de saturação dos neurônios (TARLING e ROHWER, 1993), o rastreamento de alvos (FRANCISCO e EBECKEN, 2013), o reconhecimento de padrões e a construção de modelos mais representativos para padrões previamente treinados (GRIECO *et al.*, 2010).

Podemos afirmar que a WiSARD é um sistema formado por vários discriminadores. Cada um desses discriminadores está relacionado a uma das possíveis classes do conjunto de padrões inseridas. Um discriminador é composto por um grupo de RAM (*Random Access Memory*) ou neurônios, e um dispositivo de soma $\sum$ (ALEKSANDER *et al.*, 2009).

Portanto uma arquitetura WiSARD é composta por um grupo de discriminadores, cada um deles correspondem à uma classe do problema a ser mapeada. Durante a fase de treinamento, o padrão de entrada é apresentado apenas para o discriminador com a sua classe de destino correspondente. A classificação é obtida mediante a apresentação do padrão de entrada para todos os discriminadores selecionando a classe com o maior valor de ativação. A Figura 2.9 mostra essa arquitetura.

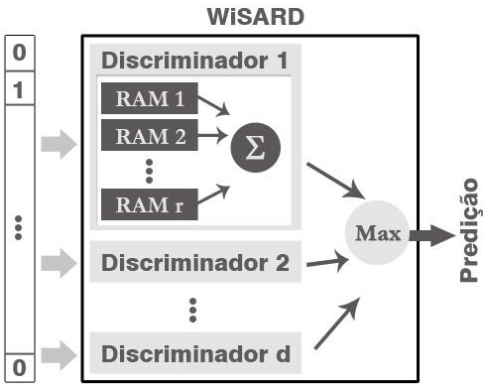


Figura 2.9: Representação de uma WiSARD, onde r é a quantidade de RAM por discriminador e d é a quantidade de discriminadores.

2.3.1.1 Fase de treinamento

Ao longo da fase de treinamento da WiSARD ocorre o aprendizado dos padrões associados às classes existentes no conjunto de entradas. Para o treinamento dos discriminadores, inicialmente é preciso que todas as posições de escrita de todos os seus neurônios estejam com o valor 0. Em seguida um conjunto de dados é fornecido e mapeado de forma aleatória, ou seja, cada neurônio de um discriminador receberá um grupo de bits específico para cada entrada de dados submetida à rede neural.

Os dados de treinamento da classe somente ativa os neurônios do discriminador corresponde à sua classe. Desta forma, o discriminador aprende as características das entradas da classe a qual ele representa, permitindo assim que ele possa reconhecer novas entradas como sendo ou não pertencentes a sua classe.

A Figura 2.10 apresenta um exemplo de treinamento de um discriminador que segue os seguintes passos: o padrão a ser treinado é convertido em binário, em seguida são selecionados N bits de forma aleatória, onde N é arbitrário e deve ser definido empiricamente. Os bits selecionados são usados para ativar uma das

suas posições de memória 2^N. A formação é executada diretamente através do armazenamento 1 na posição endereçada. A classificação é alcançada através da recuperação do valor armazenado.

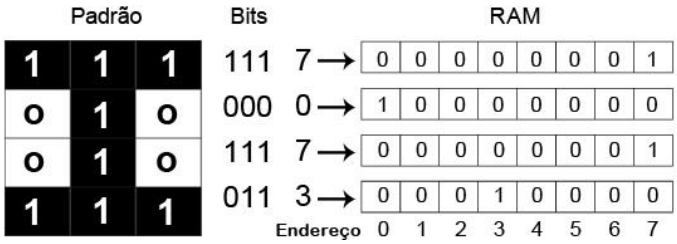


Figura 2.10: Treinamento de um discriminador da categoria I de uma WiSARD com memórias de 8 bits e entrada de 12 bits.

2.3.1.2 Fase de teste

A fase de teste tem como objetivo verificar a acurácia da rede para novas entradas. Cada entrada é mapeada aleatoriamente em grupo de bits, como já mostrado anteriormente. Em seguida o grupo de bits é endereçados a uma RAM específica. Este procedimento é aplicado a todos discriminadores. Desta forma, cada um dos discriminadores produzirá uma medida de similaridade correspondente ao valor da entrada para teste. Este valor é a quantidade de número de RAM que tiveram como saída o valor 1.

A medida de similaridade de um discriminador da WiSARD após a apuração dos dados de entrada na rede é apresentada conforme mostra a Figura 2.11.

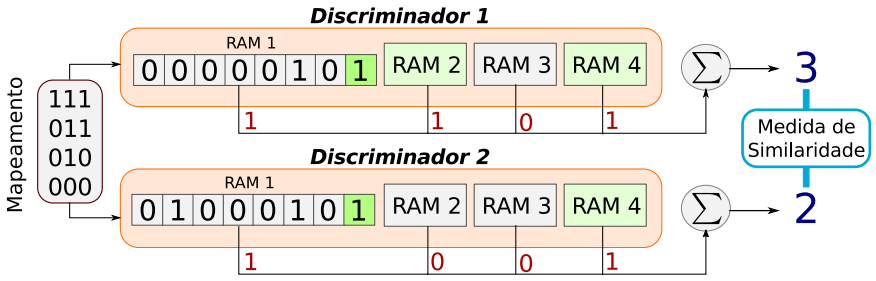


Figura 2.11: Medida de similaridade de um discriminador na fase de teste.

Para determinar a classe do elemento da base de teste, é só verificar qual discriminador obteve a maior medida de similaridade r_{max}, e assim rotular o elemento em questão com a classe deste discriminador. Após a verificação de todo conjunto de teste, calcula-se a acurácia do método WiSARD verificando o percentual de acertos.

2.3.1.3 *Bleaching*

Uma das principais desvantagens da WiSARD é o fato de realizar escolha aleatória entre os discriminadores de maior pontuação quando ocorre empate.

A técnica do *Bleaching* foi elaborada de modo a minimizar esse problema, buscando uma resposta determinística para os casos de empate, aperfeiçoando-se assim esta rede neural. O método consiste em armazenar nas memórias da WiSARD o número de vezes que o padrão foi apresentado, para que na etapa de reconhecimento, essa informação possa ser usada como critério de desempate. Com isso, ao invés de serem armazenados apenas valores binários (0 e 1) dentro das memórias, também terá os valores relacionado à quantidade de vezes que o processo de escrita foi realizado para a unidade. A figura 2.12 mostra um exemplo de classificação utilizando *bleaching*, onde b representa o valor aplicado à técnica.

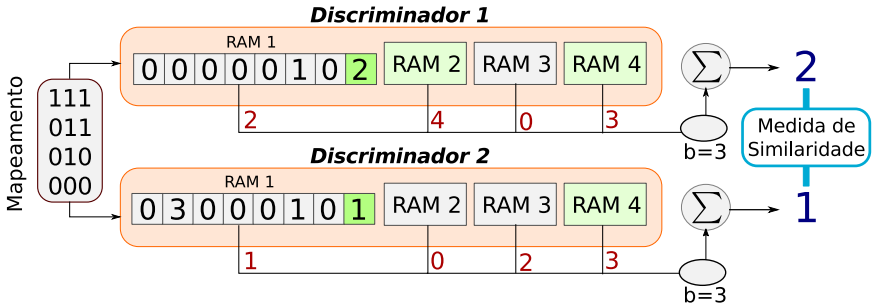


Figura 2.12: Exemplo usando a técnica de *bleaching*.

A escolha do valor de b pode variar de acordo com o que pretende-se alcançar com a rede, ocasionando uma maior generalização ou especificidade. Quanto maior for valor de b, menos generalista a rede será.

Um ponto negativo dessa técnica é a possibilidade do aumento no tempo de resposta do reconhecimento. Entretanto, não altera a complexidade, nem o tempo de resposta da etapa de treinamento.

2.4 *Fingerprints*

A técnica *fingerprints* vem sendo bastante utilizada para caracterizar e comparar documentos de acordo com o seu conteúdo. Essa abordagem é voltada para encontrar documentos similares em grandes sistemas. Originalmente foi desenvolvida por MANBER (1994). A técnica era aplicada através da seleção de trechos de um documento e da subsequente utilização de funções *hash* para produzir as *fingerprints* (HEINTZE, 1996).

As *fingerprints* funcionam como assinaturas de um documento. Portanto, uma característica dessa abordagem é que, dada uma alteração no documento, existe uma alta probabilidade de que seja produzida uma *fingerprints* diferente (MANBER, 1994). Através dessa técnica é possível fazer a comparação de *fingerprint* de diferentes documentos para determinar se elas são ou não similares (HOAD e ZOBEL, 2003, MANBER, 1994).

Basicamente a ideia por trás de uma *fingerprint* é fazer o mapeamento de uma *string* qualquer utilizando uma função matemática para um número inteiro aleatório, usando uma determinada chave (MANBER, 1994). Ou seja, uma *fingerprint* $f(d)$ de um documento d, pode ser considerada um conjunto de *substrings* extraídas de d e posteriormente codificadas, que servem para identificar d de forma única (STEIN e ZU EISSEN, 2006). As *substrings* a serem mapeadas podem ser compostas de caracteres, palavras ou mesmo sentenças (BARRÓN-CEDEÑO, 2010).

2.5 *Locality-Sensitive Hashing* para documentos

A técnica de *Locality-Sensitive Hashing* (LSH) foi desenvolvida por INDYK e MOTWD (1998), e aprimorada posteriormente por GIONIS *et al.* (1999) para resolver problemas de alta dimensionalidade computacional, como na busca do vizinho mais próximo em base de dados muito extensa ou quando a avaliação de similaridade entre pares de objetos é computacionalmente custosa (KULIS e GRAUMAN, 2012). Para essas pesquisas o tempo computacional é extremamente reduzido, com um pequeno custo de não encontrar o vizinho mais próximo ao item avaliado (SLANEY e CASEY, 2008). Essa técnica é baseada na premissa de que objetos semelhantes estão muito mais próximos do que aqueles que não são similares (GIONIS *et al.*, 1999).

A abordagem usada na técnica do LSH parte do princípio onde é gerado valores de *hashes* para diferentes objetos usando várias funções *hash*, de modo a assegurar que, para cada função, a probabilidade de colisão seja muito maior para objetos próximos do que para aqueles que estão distantes (GIONIS *et al.*, 1999). Ou seja, essa técnica remete ao um simples conceito: se dois objetos são próximos, a sua projeção vetorial em um subespaço de menor dimensionalidade leva a dois pontos adjuntos. Desta forma pode-se determinar os vizinhos mais próximos a um objeto calculando os *hashes* para o objeto e para os outros itens da base de dados e comparando os valores encontrados (GIONIS *et al.*, 1999, SLANEY e CASEY, 2008). A Figura 2.13 apresenta um exemplo de objetos mapeados com LSH.

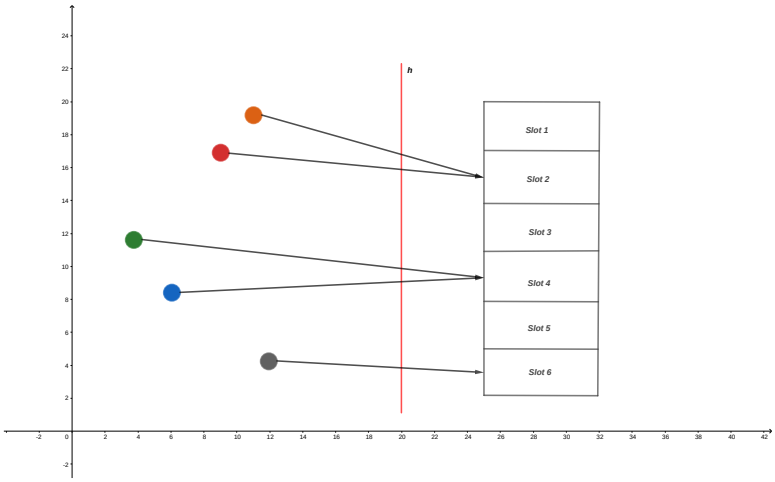


Figura 2.13: Objetos próximos são mapeados para o mesmo *slot*.

O algoritmo usado pela técnica LSH é dividido em duas etapas, primeiro é o pré-processamento e o segundo é a consulta para encontrar os vizinhos mais próximos. O pré-processamento tem como entrada um conjunto Z de I objetos e uma variável N, que representa o número de funções *hash*. As funções *hash* são responsáveis por fazer o mapeamento dos objetos para os *slots* em tabelas *hash*. Cada *slot* irá agrupar um conjunto de itens com propriedades semelhantes, isto é, o mesmo valor de *hash*. A primeira etapa do LSH é descrito pelo algoritmo 1.

Algoritmo 1: Etapa de pré-processamento do LSH

Entrada: N
início
  para *cada* $n \in N$ **faça**
    Iniciar uma tabela *hash* T_n
    Gerar uma função *hash* aleatória $h(.)$
  fim
  para *cada* $n \in N$ **faça**
    para *cada* $i \in I$ **faça**
      Mapear o objeto o_i no *slot* $h(o_i)$ da tabela T_n
    fim
  fim
fim
retorna *Tabela* T_n *contendo os itens mapeados*

Na segunda etapa, consulta dos vizinhos mais próximos, deve-se buscar os K itens que mais se aproximam do nosso objeto de busca b. É nesta fase que são utilizadas as tabelas *hash* criadas anteriormente pela etapa de pré-processamento. Veja o algoritmo 2 a seguir.

Algoritmo 2: Consulta dos vizinhos mais próximos do LSH

Entrada: T_n, N, b

início

  $V_k \longleftarrow \emptyset$

  para *cada* $n \in N$ **faça**

    $V_k \longleftarrow V_k \cup$ {objetos encontrados em $h(b)$ da tabela T_n}

  fim

fim

retorna *K vizinhos mais próximos de b encontrados no conjunto* V_k

2.5.1 MinMaxwise hashing

A abordagem de *MinMaxwise Hashing* (JI *et al.*, 2013) é uma variante do método *Minwise Hashing* proposto por BRODER (1998), são uma instância particular da técnica LSH.

A ideia do algoritmo *Minwise Hashing* é descobrir a similaridade entre documentos por meio de um problema de interseção de conjuntos, que pode ser facilmente resolvido através de um processo de amostragem aleatória realizada de forma independente para cada documento (BRODER, 1998).

A criação do método *Minwise Hashing* teve como motivação o aumento do conteúdo disponível para internet, documentos online duplicados ou até mesmo plagiado. Para atacar esse problema, o *MinMaxwise Hashing* pretende descobrir a similaridade entre documentos observando o máximo e o mínimo de um conjunto. Segundo a literatura os resultados obtidos com essa abordagem são melhores que aqueles produzidos pelo *Minwise Hashing*, já que o método *MinMaxwise Hashing*, estabelece simultaneamente as fronteiras inferiores e superiores dos conjuntos analisados.

O algoritmo *min-Hash* utiliza a permutação dos grupos de objetos pertencentes a cada conjunto Z_i do universo Z por meio de uma função *hash* h. A partir de cada permutação, o valor da assinatura mínima de cada conjunto Z_i será igual ao primeiro menor elemento pertencente a Z_i, tendo em vista a ordem da permutação fornecida por h (CHUM *et al.*, 2008).

O *MinMaxwise Hashing*, ao invés de manter apenas o menor valor de *hash* de cada permutação aleatória, como é realizado pelo *Minwise Hashing*, mantém os menores e os maiores valores de cada permutação. A grande vantagem do *MinMaxwise Hashing* é a capacidade de captar mais informações que represente o conjunto com a mesma quantidade de permutações. Portanto, para dois conjuntos serem considerados semelhantes os seus limites devem ser parecidos.

O *MinMaxwise Hashing* proporciona um método simples para estimar a simila-

ridade de documentos, permitindo que os documentos originais sejam descartados, reduzindo significativamente o tamanho da entrada (CHARIKAR, 2002). Para encontrar a similaridade entre os conjuntos e utilizar os documentos como conjuntos de características é usado o índice de Jaccard (JACCARD, 1901), conhecido também como coeficiente de similaridade de Jaccard. O índice de Jaccard é definido pela equação 2.19 (BRODER *et al.*, 2000).

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{2.19}$$

onde A é o conjunto de termos presentes no documento a e B é o conjunto de termos presentes no documento b. Esta medida aplicada a uma representação *bag-of-words* produz um cálculo de similaridade onde os termos têm pesos binários, ou seja, há o termo ou não no documento e todos possuem a mesma importância.

A similaridade de Jaccard se destina a encontrar documentos textuais em grandes coleções de dados. Avaliar se dois documentos são cópias exatas, basta comparar-los caractere a caractere, e se houver alguma diferença, pode-se concluir que eles não são os mesmos.

2.5.2 Implementação do MinMaxwise hashing

Para iniciar a implementação do método *MinMaxwise*, devemos observar a interação entre os conjuntos e os elementos através de uma representação matricial, a matriz característica M, na qual as colunas de M correspondem aos conjuntos e as linhas, a cada elemento do conjunto universo Z_i. A posição $M(j, i)$ de M é dada por 1 se o elemento j estiver presente no conjunto i. Caso contrário, $M(j, i)$ será 0.

A implementação do método *MinMaxwise Hashing* constitui em gerar duas assinaturas *minmax-Hash* referentes à permutação dada por uma função *hash* h, sendo uma delas a assinatura mínima do conjunto, correspondente ao *min-Hash* dada pela equação 2.20, e outra assinatura, correspondente ao máximo do conjunto, o *max-Hash* dada pela equação 2.21.

$$min(Z_i, h) = \arg\min_{t \in Z_i} h(t) \tag{2.20}$$

$$max(Z_i, h) = \arg\max_{t \in Z_i} h(t) \tag{2.21}$$

O método *min-Hash* é baseado na probabilidade de $min(Z_1, h) = min(Z_2, h)$ seja dado pela equação 2.22. O método *max-Hash* é análogo ao anterior como mostra a equação 2.23 (BRODER *et al.*, 2000, CHUM *et al.*, 2008).

$$Pr(min(Z_1, h) = min(Z_2, h)) = \frac{|Z_1 \cap Z_2|}{|Z_1 \cup Z_2|} = J(Z_1, Z_2) \tag{2.22}$$

$$Pr(max(Z_1, h) = max(Z_2, h)) = \frac{|Z_1 \cap Z_2|}{|Z_1 \cup Z_2|} = J(Z_1, Z_2) \tag{2.23}$$

As assinaturas *minmax-Hash* são obtidas através de N funções *hash* independentes $(h1, h2, ..., h_n)$. Os planos contêm o dobro da informação em relação às abordagens *Minwise Hashing* e *Maxwise Hashing*. Cada conjunto Z_i é caracterizado por um plano $C_{minmax,i}$. A equação 2.24 apresenta os planos gerados para o *MinMaxwise Hashing*.

$$\bar{Z}_{minmax,i} = \bar{Z}_{min,i} \;||\; \bar{Z}_{max,i} \tag{2.24}$$

A similaridade par a par entre as assinaturas dos conjuntos Z_1 e Z_2 é calculada através da equação 2.25

$$J(\bar{Z}_{minmax,1}, \bar{Z}_{minmax,2}) = \frac{1}{N} \sum_{n=1}^{N} \gamma(\bar{Z}_{minmax,1}(n), \bar{Z}_{minmax,2}(n)) \tag{2.25}$$

onde $\bar{Z}_{minmax,i}(n)$ é a assinatura *minmax-Hash* gerada para o conjunto Z_i por meio da permutação dada por h_n e pela equação 2.26

$$\gamma(A, B) = \begin{cases} 1, & \text{se A = B} \\ 0, & \text{caso contrário} \end{cases} \tag{2.26}$$

O cálculo do *minmax-Hash* para um conjunto Z_i pertencente a M é dado pelo seguinte algoritmo 3 (JI *et al.*, 2013, LESKOVEC *et al.*, 2014).

Algoritmo 3: Cálculo da assinatura *MinMaxwise Hashing* do conjunto $Z_i \in M$

Entrada: M, N
início
  $slot_{min}(h_n, Z_i)$ como ∞
  $slot_{max}(h_n, Z_i)$ como $-\infty$
  para *cada permutação dada por* h_n *com* n *de* 1 *até* N **faça**
    para *cada* L_j *de* M **faça**
      se $M(L_j, Z_i)$ *igual a* 1 **então**
        se $h_n(j) < slot_{min}(fh_n, Z_i)$ **então**
          $slot_{min}(h_n, Z_i) \longleftarrow h(j)$
        fim
        se $h_n(j) > slot_{max}(h_n, Z_i)$ **então**
          $slot_{max}(h_n, Z_i) \longleftarrow h(j)$
        fim
      fim
    fim
  fim
fim
retorna *Assinatura MinMaxwise Hashing do conjunto* Z_i

2.6 Trabalhos relacionados

A importância de técnicas de recomendação precisas, motivadas pela aplicação em vários setores, tem criado um grande número de pesquisas acadêmicas nesta área (RICCI *et al.*, 2015). Sistemas de recomendação são mais frequentemente baseados em filtragem colaborativa. Esta abordagem pretende extrapolar previsões de classificação a partir de uma matriz preenchida com as classificações correspondentes dadas pelos usuários aos itens.

Um dos primeiros a trabalhar com filtragem colaborativa lidando com uma grande quantidade de itens foi do grupo Movie Lens[7] no sistema *Usenet news* RESNICK *et al.* (1994), o trabalho consistiu em usar a correlação de Pearson para identificar similaridade entre os usuários.

Outra abordagem de filtragem colaborativa foi explorada em O'CONNOR e HERLOCKER (2001), onde foi usada clusterização dos itens contidos também na base de dados do Movie Lens. Porém algumas restrições foram feitas neste trabalho, como descartar qualquer filme correlacionado com menos de cinco outros filmes ou descartar qualquer filme com menos de dez votos.

Normalmente, há duas abordagens que são amplamente utilizadas. Por exemplo, o método de vizinhos mais próximos no qual utiliza a semelhança entre usuários através da base do conteúdo que tenham avaliação em comum, desta forma, criar uma relação para uma recomendação (ADOMAVICIUS e TUZHILIN, 2005). A

[7] https://movielens.org/

outra abordagem que se destaca por ser relacionalmente mais poderosa tem sido o uso de modelos de fatores latentes. Fatoração de Matriz (MF) é a técnica mais popular para derivar modelos de fatores latentes. Obteve sucesso na competição Netflix[8] mostrando sua força BENNETT *et al.* (2007), KOREN *et al.* (2009).

Abordagem de recomendação usando o algoritmo IRSVD (*Improved Regularized Singular Value Decomposition*), foi utilizada em PATEREK (2007). Neste trabalho a clusterização foi feita usando o K-Means, houve também um pós-processamento com SVD(Singular Value Decomposition) e K-NN juntos com outros algoritmos baseados no SVD. Os resultados apresentados utilizaram como métrica de avaliação o MAE (*Mean Absolute Error*) e o RMSE (*Root Mean Squared Error*).

Ultimamente, os métodos baseados em *deep learning* surgiram como uma ferramenta poderosa para a representação de aprendizagem e são amplamente utilizados em muitas aplicações, que vão desde o reconhecimento de voz e o reconhecimento de imagem. A aplicação desses modelos para a tarefa de filtragem colaborativa é recente e não há muitas tentativas nesse sentido. SALAKHUTDINOV *et al.* (2007) foram os primeiros a aplicar *deep learning* para a tarefa de filtragem colaborativa.

Mais recentemente, WANG *et al.* (2014) propôs um Modelo hierárquico bayesiano chamado de *collaborative deep learning* (CDL), que utiliza de *Stacked denoising Autoencoders* (SDA). Visa extrair características para itens para serem utilizadas para recomendação. LI *et al.* (2015) apresentou um trabalho semelhante ao anterior, porém calcula os parâmetros de forma fechada e extrai características dos itens e usuários.

Já no trabalho de BRAIDA *et al.* (2015), propôs uma metodologia para transformar o problema da filtragem colaborativa em um problema de aprendizado supervisionado sendo nomeado de COFILS. No qual pode ser dividida em 3 etapas: pré-processamento, extração de variáveis latentes e regressão/classificação. A matriz de *ratings* é a entrada para extração das variáveis latentes. Estas variáveis latentes são então utilizadas na segunda etapa que constrói um espaço de recurso da entrada para representar os *ratings*. O trabalho de BARBIERI *et al.* (2017), deu continuidade ao COFILS, os autores apresentaram uma alternativa para primeira fase, no qual utiliza o *Stacked Denoising Autoencoder* (SDA) para extração de variáveis latentes, onde conseguindo bons resultados. Entretanto as duas propostas utilizando a abordagem COFILS não se preocuparam em analisar o problema de *cold-star* e a dinâmica temporal que os dados sofrem podendo afetar o modelo.

A incorporação de informações secundárias melhora a precisão da previsão, acelera o processo de treinamento e permite que o modelo seja mais robusto. Em vista disso, os trabalhos de STRUB e MARY (2015), STRUB *et al.* (2016) criaram um modelo chamado *Collaborative Filtering Neural network* (CFN), que é uma extensão

[8]https://www.netflix.com

do trabalho de SEDHAIN *et al.* (2015a), no qual implantam as técnicas de *denoising*, o que torna o CFN mais robusto incorporando as informações secundárias, como perfis de usuário e descrição de itens, para atenuar a influência da dispersão e de *cold-start*.

Já o trabalho de WEI *et al.* (2017), propôs um modelo colaborativo híbrido baseado em *autoencoder* e timeSVD++. É um modelo que utiliza informações temporais e o SDAE para aprender representações de itens a partir de recursos brutos e tem como objetivo resolver o problema de *cold-start*.

No entanto, os métodos baseado em *deep learning* para encontrar similaridade com intuito de recomendação de itens que sofrem com *cold-start* são computacionalmente caro. No trabalho de GOPE e JAIN (2017) foram apresentadas diversas soluções existentes mostrando como as abordagens coletam as informações ausentes. Segundo o autor, o problema de *cold-start* ainda está em aberto. Portanto, para uma melhor solução, os pesquisadores interessados podem partir nesta direção para pesquisa.

Percebe-se o uso de várias abordagens e pesquisas para tentar resolver o problema de recomendação, porém até o presente momento da execução deste trabalho, não foi encontrado nenhum artigo ou pesquisa relacionado ao nosso trabalho, no qual utiliza de rede neurais sem peso WiSARD como abordagem para classificação de notas em sistema de recomendação em *cold-start*.

A WiSARD é assunto de pesquisa intensiva em vários domínios por exemplo: problema de detecção de mudança no campo de visão de uma câmera GREGORIO *et al.* (2012), predição de crises epilépticas DE AGUIAR (2014), previsão de umidade de solos BRANDT (2015), rastreamento de objetos em vídeos BERGER *et al.* (2016), estimativa de fundo para aplicações de cena de vídeo DE GREGORIO e GIORDANO (2017), detecção de falhas e diagnóstico em sistemas dinâmicos univariados e multivariados OLIVEIRA *et al.* (2017), reconhecimento de conjunto aberto CARDOSO *et al.* (2017), e assim por diante. Esses trabalhos em geral obtiveram em suas abordagens resultados significativos.

Desta forma visamos analisar a rede neural sem peso WiSARD como uma alternativa para solução do problema de recomendação em cold-start, juntamente com a utilização de métodos menos custosos para coleta de conteúdo para aplicação em aprendizado supervisionado e *deep learning*.

Capítulo 3

Proposta

Neste capítulo descrevemos a proposta do nosso trabalho para atingir os objetivos apresentados na Seção 1.2. A princípio apresentamos uma breve motivação para a proposta de criação de uma abordagem híbrida para o problema de *cold-start* itens em filtragem colaborativa utilizando técnicas de pré-processamento de textos, em seguida definimos o problema que iremos atacar. Por fim vamos discutir os passos para esse processo.

3.1 Motivação e definição do problema

Sistemas de recomendação são técnicas e ferramentas usadas para sugerir itens personalizados baseados no perfil do usuário. O interesse nesta área ainda permanece alto, pois constitui uma linha de pesquisa rica em problemas, devido à abundância de aplicações práticas que ajudam os usuários a lidar com a sobrecarga de informações para as recomendações.

Portanto, graças à recomendação é possível minimizar a possibilidade de ocorrer um abandono no processo de compra, uma vez que fazer escolhas com muitas opções pode ser incômodo e difícil SCHWARTZ (2005).

Em vista deste problema, os sistemas de recomendação surgiram com o foco na busca por informações relevantes de acordo com características do próprio usuário, bem como em determinados requisitos relacionados aos itens que se quer encontrar. A filtragem colaborativa é uma das abordagens mais bem sucedidas em sistemas de recomendação. A abordagem visa recomendar um item para um usuário baseado em itens previamente avaliados por outros usuários do sistema.

Entretanto, é amplamente sabido que as abordagens dos métodos de filtragem colaborativa sofrem de problemas como: *cold-start*, escalabilidade e dispersão. O problema *cold-start* é um dilema de longa data em sistemas de recomendação. Ocorre quando existem a indisponibilidade de informações adequadas sobre os itens ou

usuários disponíveis no sistema, e desta forma não é possível fazer recomendações relevantes (PARK e CHU, 2009).

A abordagem híbrida é aquela que combina a abordagem de filtragem baseada em conteúdo e a filtragem colaborativa, tentando amenizar os dificuldades e fornecer um resultado mais eficiente. Embora muitos trabalhos, tenham sido feitos com a abordagem híbrida para resolver os problemas de esparsidade e *cold-start*, a recomendação de itens para CS ainda é uma questão de pesquisa em aberta.

O problema de *cold-start* pode ser dividido em CS usuário e CS item, podendo ter subcategorias como: completo *cold-start*, que representa os itens ou usuários que não receberam ou fizeram nenhuma avaliação, e o incompleto *cold-start*, quando o número de registros das avaliações tanto dos itens ou usuários são baixas.

3.2 Definição da proposta

O presente trabalho propõe uma abordagem híbrida para o problema de *cold-start* itens na filtragem colaborativa em sistemas de recomendação. Utilizamos a técnica de *Locality-Sensitive Hashing* com o método *MinMaxwise* para processamento de textos com intuído de encontrar as similaridades entre os itens.

O LSH foi incorporado nas abordagens tradicionais da literatura como rede neurais sem peso e demais métodos de aprendizagem de máquina. Desta forma, podemos comparar a abordagem WiSARD, que permite a construção de algoritmos flexíveis com o baixo custo computacional e de aprendizagem rápida, com técnicas de *deep learning*, que têm sido utilizadas com sucesso em diferentes tipos de tarefas de classificação, e como *baseline* usaremos a abordagem de k-vizinhos mais próximos. A figura 3.1 apresenta o fluxo para implementação da nossa proposta.

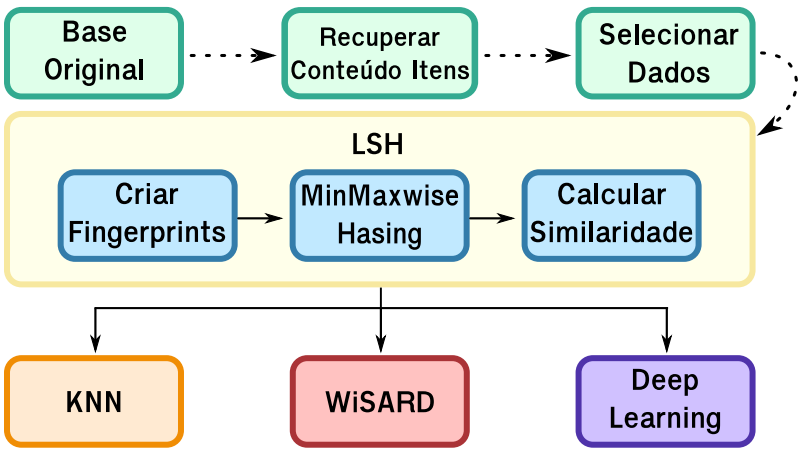


Figura 3.1: Fluxo para implementação da proposta.

O objetivo é utilizar informações sobre os itens para ser usados nos algorítimos de recomendação. Os dados que coletamos estão relacionado as características dos

itens como por exemplo: atores, escritores, diretores, país de origem, idioma, gênero, indicações e/ou premiações e a sinopse do filme. Esses dados serviram de entrada para o LSH que retorna uma matriz de similaridade entre os itens, que por sua vez, será utilizada em conjunto na implementação das abordagens.

Conforme mostrado anteriormente pela Figura 3.1, o primeiro passo será recuperar os dados de cada item e armazenar em um banco de dados, em seguida aplicar o método *MinMaxwise* para encontrar a matriz de similaridade.

Para cada abordagem selecionada para os experimentos, a matriz de similaridade será adicionada ao método. Segundo passo é elaborar de qual forma será a utilização desta similaridade. E por fim vamos comparar e analisar os resultados obtidos.

Com objetivos secundários, iremos aplicar o LSH em um modelo proposto por BRAIDA *et al.* (2015), que transforma o problema da filtragem colaborativa em um problema de aprendizado supervisionado (COFILS). No qual utiliza técnica de decomposição de valores singulares (SVD) para diminuição da dimensionalidade e recuperação das variáveis latentes. Desta forma, será definido um novo espaço utilizando a matriz de preferências das características dos usuários e dos itens. Logo a previsão de uma nota utilizando esse novo espaço, torna-se um problema de reconhecimento de padrão, então é possível aplicar as técnicas de aprendizado de máquina.

Para as abordagens com *deep learning*, vamos variar o tamanho e a quantidade das camadas ocultas e a escolha dos atributos de entradas. Desta forma é possível analisar o método com diversos tipos de modelagens.

Por fim, vamos analisar todos os métodos propostos em condições de completo *Cold-Start* e incompleto *Cold-Start*. Desta forma vamos verificar o desempenho do algoritmo em relação a variação da quantidade de itens que sofrem com o problema de *Cold-Start* dentro da base de dados.

No próximo capítulo mostraremos com mais detalhes a metodologia adotada e seus devidos experimentos.

Capítulo 4

Avaliação experimental

Nesse capítulo descrevemos a metodologia adotada para este trabalho, apresentamos a base de dados utilizada nos experimentos e o processo de recuperação das informações para os itens. Definimos e discriminamos os experimentos, mencionamos as métricas adotadas para avaliação. Por fim, apresentamos os resultados obtidos e suas análises.

4.1 Objetivos dos experimentos

Nesta trabalho foi realizado experimentos para as abordagens *k-nearest neighbors* (KNN) que serviu de *baseline*, usamos uma rede neural sem peso (WiSARD) e uma rede neural *perceptron* multicamadas (RN). Os métodos foram avaliados no seu estado da arte e também acrescido do *Locality-Sensitive Hashing* (LSH) na sua implementação, tendo como objetivo compará-los em um ambiente com o problema de *cold-start*.

As descrições dos métodos utilizados nos experimentos são mostradas a seguir:

- **WiSARD (Wd)**
  Contém 5 discriminadores. Usa para treinamento os dados da idade, sexo e ocupação do usuário mais o gênero do filme.

- **WiSARD Gêneros (Wd-G)**
  Contém 5 discriminadores para cada gênero do filme. Usa para treinamento os dados da idade, sexo e ocupação do usuário mais o gênero do filme.

- **WiSARD Média (Wd M)**
  Contém 5 discriminadores. Calcula a nota média em relação ao usuário, filme, sexo, ocupação e grupos de idades, para utilizar no treinamento junto com os dados da idade, sexo e ocupação do usuário mais o gênero do filme.

- **WiSARD Gêneros Média (Wd-G M)**
  Contém 5 discriminadores para cada gênero do filme. Calcula a nota média em relação ao usuário, filme, sexo, ocupação e grupos de idades, para utilizar no treinamento junto com os dados da idade, sexo e ocupação do usuário mais o gênero do filme.

- **WiSARD Média LSH (Wd M-LSH)**
  Contém 5 discriminadores. A partir da matriz de similaridade gerada pelo LSH, calcula-se a nota média em relação ao usuário, filme, sexo, ocupação e grupos de idades, para utilizar no treinamento junto com os dados da idade, sexo e ocupação do usuário mais o gênero do filme.

- **WiSARD Gêneros Média LSH (Wd-G M-LSH)**
  Contém 5 discriminadores para cada gênero do filme. A partir da matriz de similaridade gerada pelo LSH, calcula-se a nota média em relação ao usuário, filme, sexo, ocupação e grupos de idades, para utilizar no treinamento junto com os dados da idade, sexo e ocupação do usuário mais o gênero do filme.

- **KNN Média Item (Knn Mi)**
  Realiza a normalização pela média das notas do item.

- **KNN Média Usuário (Knn Mu)**
  Realiza a normalização pela média das notas do usuário.

- **KNN Média Global (Knn Mg)**
  Realiza a normalização pela média das notas globais.

- **KNN LSH (Knn LSH)**
  Realiza a normalização pela média das notas dos itens similares.

- **Rede Neural Média (RN M)**
  A rede utiliza como entrada para treinamento os id dos itens e usuários, esses dados são agregados às notas médias relacionada ao item, usuário, grupos de idades, sexo e ocupação.

- **Rede Neural Média LSH (RN M-LSH)**
  A rede utiliza como entrada para treinamento os ids do item, usuário e dos itens similares, esses dados são agregados as notas médias relacionada ao item, usuário, grupos de idades, sexo e ocupação.

- **Rede Neural (RN)**
  A rede utiliza como entrada para treinamento os ids do item e usuário.

- **Rede Neural LSH (RN LSH)**
  A rede utiliza como entrada para treinamento os id dos itens, usuários e dos itens similares.

Além dos métodos citados anteriormente, foram investigados também os algoritmos floresta aleatória (RF) e variações de rede neurais, que obtiveram de entrada os dados vindo da decomposição de matriz. A Figura 4.1 mostra o diagrama dos métodos analisados em nosso trabalho.

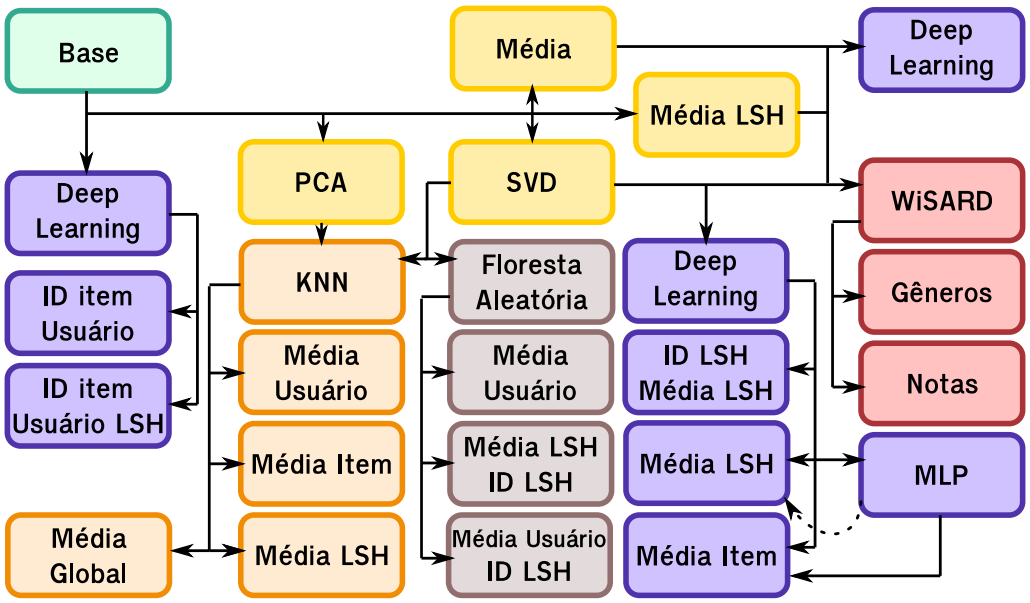


Figura 4.1: Diagrama dos métodos analisados.

Para um melhor entendimento nomeamos os experimentos de I a IX, cujos objetivos individuais são discriminados abaixo:

Experimento I

Neste experimento nosso objetivo é encontrar o melhor k_lsh para a abordagem que utiliza o LSH junto com Knn e compará-lo com abordagens tradicionais utilizadas no método Knn.

Experimento II

Neste experimento vamos comparar os valores obtidos considerando a variação dos pesos dos atributos de idade, sexo, ocupação e gênero para os métodos Wd e Wd-G.

Experimento III

Neste experimento vamos comparar os valores obtidos considerando a variação dos pesos dos atributos de nota média do filme, nota média do usuário, nota média do sexo, nota média da ocupação, nota média para idade por grupo para os métodos Wd M, Wd M-LSH, Wd-G M e Wd-G M-LSH.

Experimento IV
Neste experimento nosso objetivo é aplicar a abordagem LSH junto com uma rede Neural e comparar os métodos testados anteriormente.

Experimento V
Neste experimento nosso objetivo é encontrar o valor para o *bleaching* para os métodos Wd M, Wd M-LSH, Wd-G M e Wd-G M-LSH.

Experimento VI
Neste experimento nosso objetivo é encontrar o melhor k_lsh para a rede neural com a utilização do LSH.

Experimento VII
Neste experimento vamos comparar os métodos considerando a variação dos itens para completo *cold-start*.

Experimento VIII
Neste experimento vamos comparar os métodos para a variação dos itens para incompleto *cold-start*.

Experimento IX
Neste experimento vamos apresentar a performance dos algoritmos, mostrando os tempos praticados para treinamento, teste e pré-processamento para os métodos utilizados.

4.2 Metodologia

Vamos descrever nesta seção os passos e soluções tomadas que deram as diretrizes para a realização dos experimentos do trabalho.

4.2.1 Base de dados

A base de dados usada neste trabalho foi a coletada do recomendador online do MovieLens[1] (ml-100k), contendo 100 mil notas dadas por 943 usuários para 1682 filmes, onde os usuários poderiam variar de 1 a 5, onde 1 é ruim e 5 é excelente. Este conjunto de dados foi gerado em abril de 1998.

Como a base de dados selecionada não contém informações detalhada dos filmes, dados fundamentais que são utilizados para melhorar as previsões das classificações

[1] https://movielens.org/

dos itens que sofrem com o problema de *cold-start*. Desta forma, foi necessário criar uma metodologia para extrair os dados. A Figura 4.2 mostra o diagrama desta metodologia.

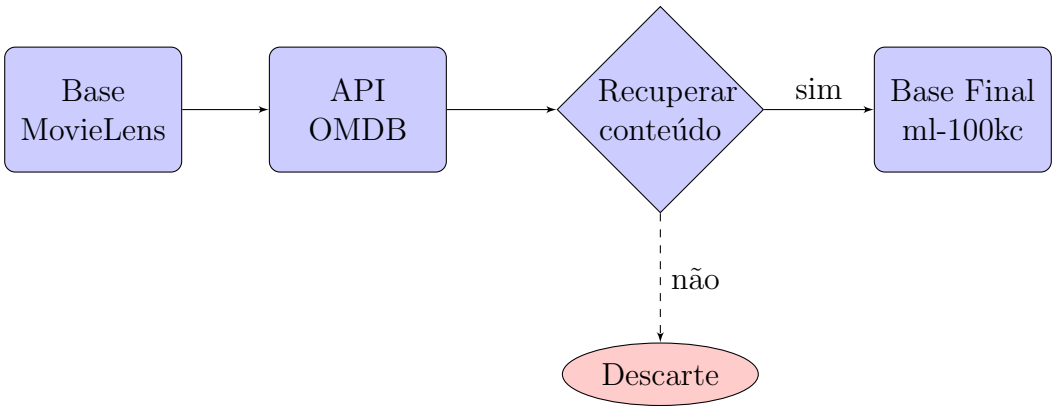


Figura 4.2: Diagrama de recuperação dos conteúdos dos filmes.

Como pode ser visto pelo diagrama anterior, usamos a API do OMDb[2], que é um serviço da Web gratuito para obter informações sobre os filmes, para extrair os dados dos filmes do site IMDB[3].

Portanto, um programa em Python foi criado para realizar a extração destes conteúdos. O programa tem como objetivo percorrer os filmes do conjunto de dados do MovieLens e enviar solicitações de busca utilizando as variáveis de título do filme, ano do filme e a identificação do IMDB para a API OMDb. Se o resumo do filme estivesse preenchido as respostas das solicitações são salvas em um banco de dados.

Em seguida foi realizada a remoção da base de dados dos itens que não foram encontrados ou que tiveram a sinopse não preenchida. Portanto o conjunto de dados final possui 943 usuários, 1159 filmes e 73117 avaliações apresentada na tabela 4.1.

Tabela 4.1: Base de dados após tratamento.

Usuários	Itens	Avaliações	Grau de Esparsidade
943	1159	73117	93.31%

O gráfico 4.3 apresenta a distribuição das avaliações por *rating* após o tratamento da base. Já os gráficos 4.4a e 4.4b mostram a distribuição das avaliações ao longo do tempo separada por mês e dia, respectivamente.

[2]http://www.omdbapi.com
[3]http://www.imdb.com/

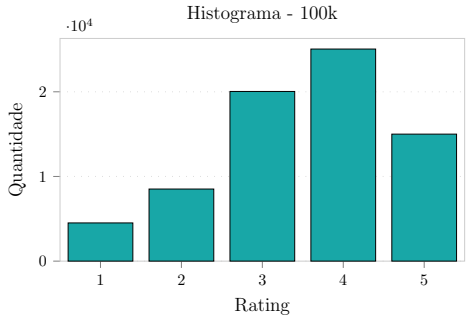
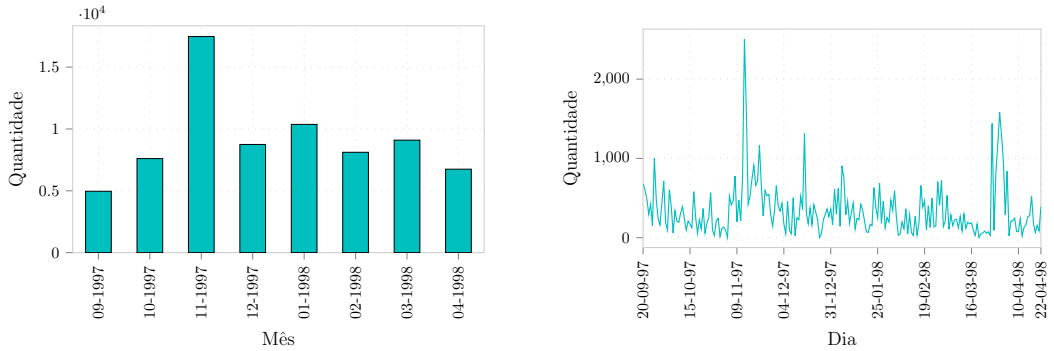


Figura 4.3: Histograma MovieLens 100kc.



(a) Rating por mês

(b) Rating por dia

Figura 4.4: Distribuição das avaliações do MovieLens 100kc em relação ao tempo.

Na fase de recuperação do conteúdo coletamos as seguintes dados: atores, escritores, diretores, país de origem, idioma, gênero, indicações e/ou premiações e a sinopse do filme. Os dados coletados juntamente com o título do filme servem de entrada para a técnica LSH, que por sua vez é responsável em criar a matriz de similaridade entre os itens. Na Tabela 4.2 apresentamos um resumo dos dados selecionados.

Tabela 4.2: Relação dos conteúdos recuperados.

Ator	País	Idioma	Gênero	Diretor	Escritor
2900	54	82	24	860	1999

Note que apesar da base original do MovieLens (ml-100k) conter apenas 19 tipos de gêneros, após a execução do algoritmo de recuperação do conteúdo para os itens a quantidade subiu para 24 gêneros.

4.2.2 Métricas de avaliação

Existem diversas métricas utilizadas na literatura para comparar sistemas de recomendação. Podem ser encontradas em HERLOCKER *et al.* (2004) e em HERLOCKER *et al.* (1999). Segundo os autores, ao invés de usar a acurácia da classificação ou o erro da classificação, as métricas mais usada em filtragem colaborativa são o *mean absolute error* (MAE) e *root mean squared error* (RMSE). Portanto, neste trabalho serão utilizadas as métricas MAE e RMSE que são normalmente usadas para comparar as técnicas de recomendação.

O MAE calcula a média da diferença absoluta entre as predições e as avaliações reais, como mostra a Equação 4.1.

$$MAE = \frac{\sum_{u,i} |p_{ui} - r_{ui}|}{n} \tag{4.1}$$

onde n é o total de avaliações sobre todos os usuários, p_{ui} é a avaliação prevista para o usuário u sobre o item i e r_{ui} é a real avaliação. Quanto menor o valor do MAE melhor é a predição.

O RMSE diferencia-se do MAE devido ao fato de penalizar os erros maiores em comparação aos erros menores. Desta forma a métrica RMSE amplifica as contribuições de erros absolutos entre predições e valores reais, conforme mostra a Equação 4.2.

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

onde n é o total de avaliações sobre todos os usuários, p_{ui} é a avaliação prevista para o usuário u sobre o item i e r_{ui} é a real avaliação.

4.3 Organização dos experimentos

Os experimentos I, II, III e IV foram executados utilizando o método de validação cruzada, técnica amplamente empregada em problemas onde o objetivo da modelagem é a predição. Adotamos o conceito de *k-fold*, que consiste em dividir a base de dados aleatoriamente em k sub-bases e para cada uma dessas divisões é aplicado o algoritmo utilizando essas sub-bases como teste e os demais como treinamento, portanto foi adotado o valor 10 para k.

E importante ressaltar que os experimentos V, VI, VII e $VIII$ foram baseados no modelo de experimento proposto por WEI *et al.* (2017). No qual separa a base de acordo com a quantidade de itens que sofrem com completo *cold-start* (CCS) e incompleto *cold-start* (ICS).

Primeiramente a base é ordenada de acordo com a data da inserção do *rating* dado pelo usuário a um item. Em seguida é escolhida a quantidade L de itens que pretende sofrer com o problema de *cold-start*, foram escolhidos os valores 100, 200 e 300 para L, tanto para CCS e ICS. Vale lembrar que CCS está relacionado a itens que não tiveram nenhum *rating* e o ICS são itens que receberam no máximo 5 *ratings*. As tabelas 4.3 e 4.4 apresentam a relação das amostras selecionadas para cada L do CCS e ICS respectivamente.

Tabela 4.3: Estatísticas das bases de treinamento e teste para CCS.

	L=100		L=200		L=300	
	Treino	Teste	Treino	Teste	Treino	Teste
Número de Itens	1059	1118	959	1140	859	1154
Número de Usuários	608	467	278	772	152	868
Número de *Ratings*	45521	27596	18221	54896	10003	63114
Porcentagem	62,26%	37,74%	24,92%	75,08%	13,68%	86,32%

Tabela 4.4: Estatísticas das bases de treinamento e teste para ICS.

	L=100		L=200		L=300	
	Treino	Teste	Treino	Teste	Treino	Teste
Número de Itens	1132	931	1110	1052	1068	1114
Número de Usuários	886	141	701	370	619	457
Número de *Ratings*	67203	5914	54247	18870	46135	26982
Porcentagem	91,91%	8,09%	74,19%	25,81%	76,77%	23,23%

A tabela 4.5 mostra a quantidade de itens e avaliações com CCS para as amostras de ICS 100, 200 e 300.

Tabela 4.5: Relação de itens com CCS na base ICS.

Quantidade	ICS 100	ICS 200	ICS 300
Itens	27	49	91
Avaliações	85	148	460

Já nos experimentos VII e $VIII$, foram também analisadas as entradas para os métodos utilizando a técnica de decomposição de valores singulares, baseada

no COFILS, modelo proposto por BRAIDA *et al.* (2015). No qual transformar o problema da filtragem colaborativa em um problema de aprendizado supervisionado.

Por fim, no experimento IX apresentamos os resultados para os tempos de treinamento para os métodos utilizados.

4.3.1 *Locality-Sensitive Hashing*

Para utilizar o LSH, previamente foram criadas as *fingerprints* para cada atributo recuperado na fase de pré-processamento. Foi utilizado o TF-IDF para encontrar as frequências dos termos, tendo os parâmetros definidos para *stop words* sendo *english* e o tamanho do *ngran* igual a 2.

Através de testes empíricos, foi definido o uso de 50 permutações para encontrar duas assinaturas, sendo uma mínima e outra máxima de cada documento. Em seguida foi calculada a matriz de distancia através do índice de Jaccard.

Portanto, para cada atributo recuperado, existe uma matriz de similaridade. Sendo assim, foi calculada a média das matrizes destes atributos, resultando então, em uma matriz de similaridade com todos atributos, sendo esta utilizada no trabalho.

4.3.2 Implementação da WiSARD

Como citado anteriormente, analisamos diferentes formas de abordagens para a WiSARD. O modelo proposto da implementação é representado pela Figura 4.5, onde podemos ver as fases para o desenvolvimento.

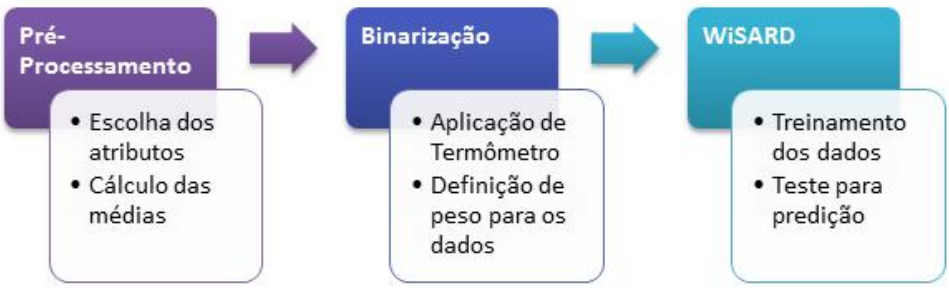


Figura 4.5: Modelo de implementação da WiSARD.

Na fase de binarização dos dados escolhemos a técnica de termômetro para dados normalizados. Assim, proporcionalmente transformamos uma variável quantitativa em valores compostos por uma sequência de 0 e 1. O termômetro tem como objetivo ordenar sequencialmente as unidades de acordo com os dados originais. A Tabela 4.6 mostra um exemplo de utilização do termômetro.

Tabela 4.6: Codificação do termômetro para variáveis quantitativas.

Idade	**Codificação**
0-9	0000000
10-19	1000000
20-29	1100000
30-39	1110000
40-49	1111000
50-59	1111100
60-69	1111110
70-79	1111111

Para atributos categóricos, aplicamos uma técnica que indica a presença ou ausência de um atributo. Esses atributos assumem valores binários: 0 significa ausência e 1 significa presença. A Tabela 4.7 mostra um exemplo desta codificação. Vale a pena mencionar que um filme pode ter mais de um gênero, por exemplo, ação e ficção científica. A abordagem é uma solução razoável, pois ativa os bits relacionados a esses gêneros.

Tabela 4.7: Codificação para atributos categóricos.

Gênero	**Codificação**
Ação	1000
Comédia	0100
Drama	0010
Ficção	0001

4.3.3 Modelo *deep learning*

Como foi visto na Figura 4.1, criamos diferentes formas de abordagens utilizando técnicas de *deep learning*. Vamos detalhar os modelos que adotamos para as análises realizadas em nosso trabalho.

Através de testes preliminares foram escolhidos os atributos para rede neural. As Figuras 4.6 e 4.7, mostram o fluxo da implantação dos modelos para as abordagens de deep learning e indicam os tamanhos das entradas e das camadas ocultas utilizadas.

(a) DL-RN-id (b) DL-RN-svd lsh/mi (c) DL-RN-mlp lsh/mi

Figura 4.6: Fluxo da implementação dos modelos de deep learning.

(a) DL-RN-LSH-id (b) DL-RN-LSH-svd-lsh

Figura 4.7: Fluxo da implementação dos modelos de deep learning com a inserção dos itens similares gerados por LSH.

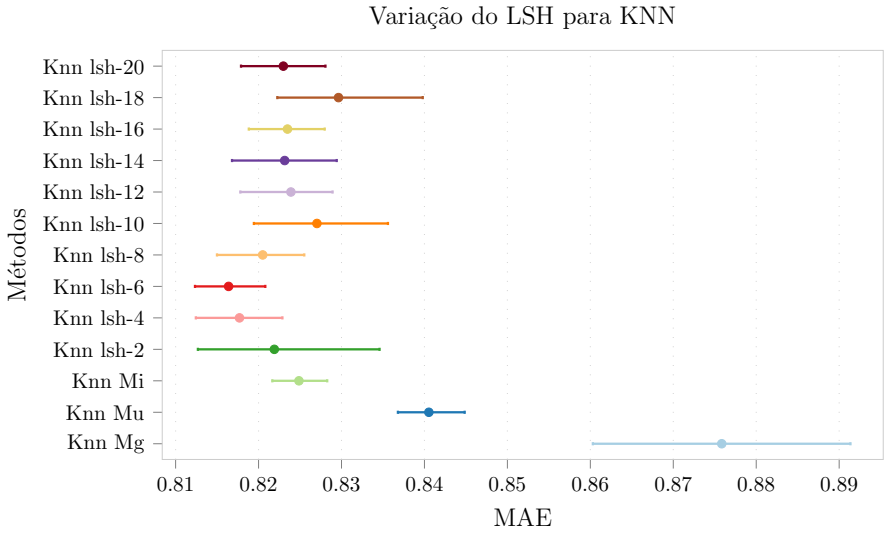
Nota-se que após a concatenação as abordagens 4.6a, 4.6b, 4.7a e 4.7b seguem os mesmos fluxo, desta forma é possível verificar a influência da entrada, sem que os parâmetros de regulagem dos métodos interfiram. Já a abordagem 4.6c apresenta um modelo mais simplificado, com intuito de comparação aos modelos sofisticados.

4.4 Resultados e discussão

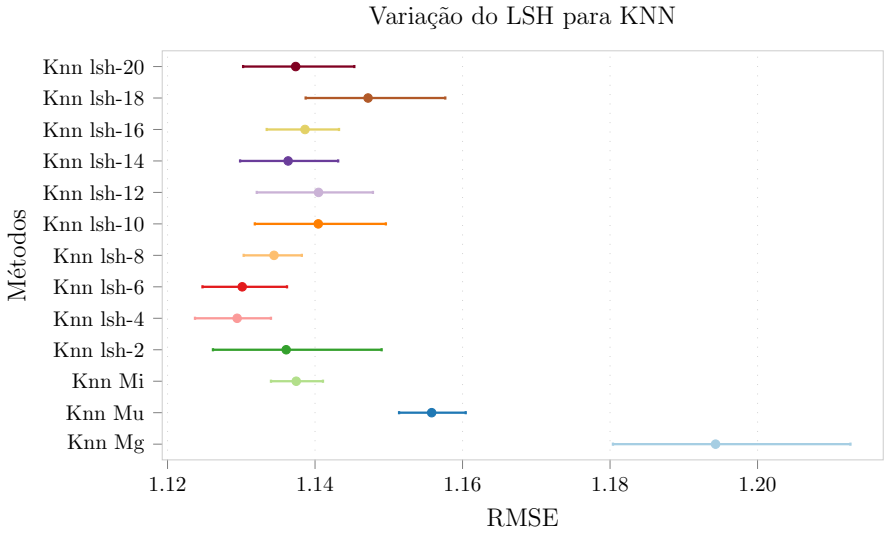
Nesta seção apresentamos os resultados dos experimentos citados na Seção 4.1 e realizamos as análises dos resultados.

4.4.1 Experimento I

Neste experimento realizamos a variação do *k_lsh* para o método Knn LSH, fixando para k vizinhos igual 100 e comparamos com abordagens tradicionais utilizados no método Knn. A Figura 4.8a e 4.8b apresenta o valor para o MAE e RMSE, respectivamente. Nota-se que após a execução, os métodos com os parâmetros de *k_lsh_6* e *k_lsh_4* obtiveram os melhores resultados tanto para o MAE e RMSE.



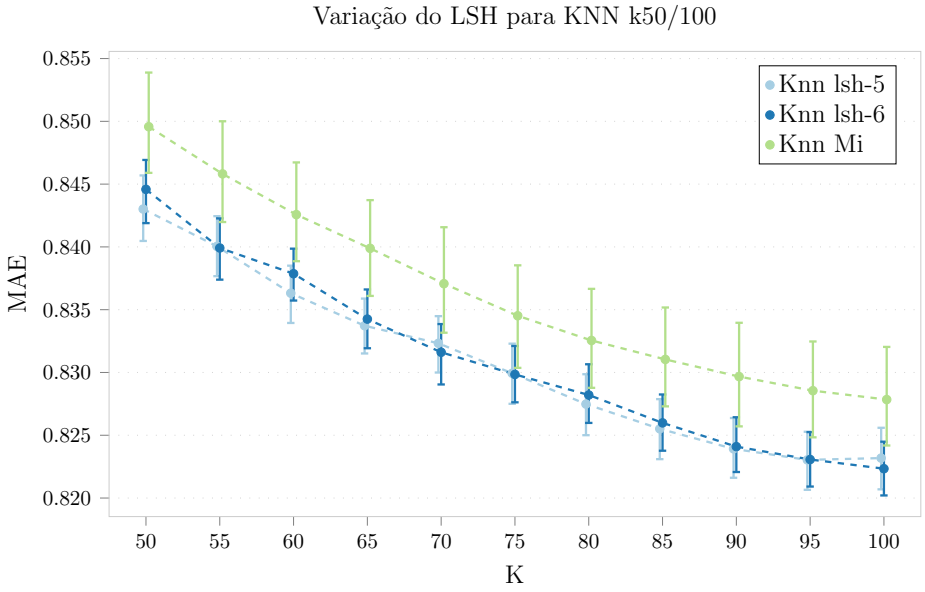
(a) Valor do MAE



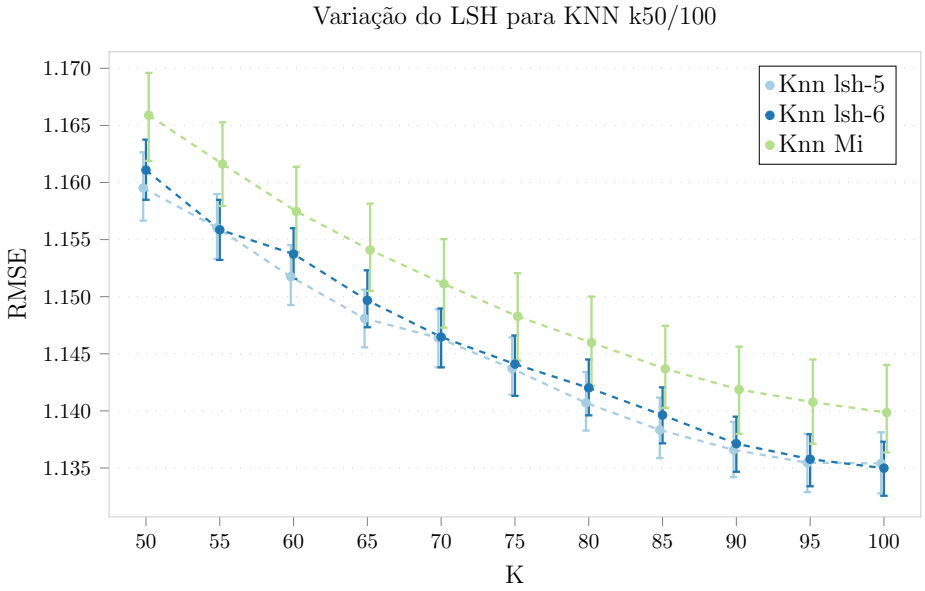
(b) Valor do RMSE

Figura 4.8: Resultado do MAE e RMSE para Knn com k=100.

Com intuito de validar os resultados anteriores, a Figura 4.9 mostra a relação da variação do k do método Knn com a variação do k_lsh_5 e k_lsh_6.



(a) MAE



(b) RMSE

Figura 4.9: Resultado do MAE e RMSE para Knn com k entre 50-100.

Através destes experimentos podemos inferir que a utilização do método LSH junto com o Knn obteve melhores resultados que o knn com abordagens tradicionais. Porém, o valor para o k_lsh deve ser inferior a 10, sendo assim adotamos para o k_lsh o valor 5.

4.4.2 Experimento II

Neste experimento comparamos os valores obtidos do MAE para a variação dos pesos dos atributos de idade, sexo, ocupação e gênero, para os métodos Wd e Wd-G. A Figura 4.10 apresenta os resultados.

Percebe-se que ao aumentar o peso dos atributos, aumenta a possibilidade de variação na quantidade de memória RAM do método WiSARD. Ocorre também o aumento do tempo de treinamento e teste, que pode ser verificado no experimento IX, na seção 4.4.9, podendo ser vista na Figura 4.33.

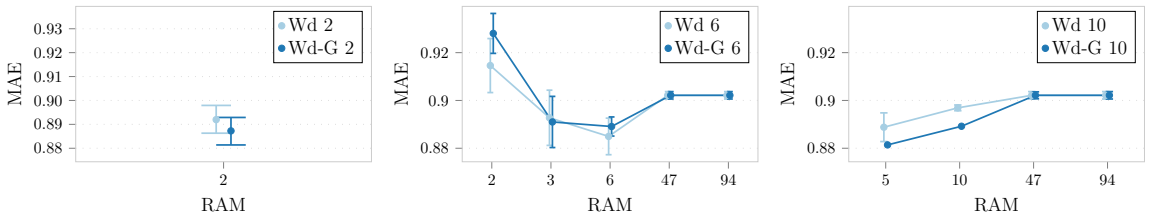


Figura 4.10: WISARD variação dos pesos para (Idade, Sexo, Ocupação, Gênero, Ano).

Entretanto, o valor do MAE diminui de 0.88726 para 0.8814 se compararmos as abordagens Wd-G 2 e Wd-G 10, respectivamente. Portanto, a escolha do peso dos atributos impacta no tempo de execução do algoritmo. Desta forma a escolha dos pesos vai levar em consideração se deseja-se ganho computacional ou de acurácia. Outra observação que podemos fazer é em relação a abordagem WiSARD utilizando gêneros. O método obteve, na maioria dos casos, um melhor resultado.

4.4.3 Experimento III

A Figura 4.11 apresenta os resultados para este experimento, no qual comparamos os valores obtidos do MAE para a variação dos pesos para os atributos de nota média do filme, nota média do usuário, nota média do sexo, nota média da ocupação, nota média para idade por grupo para os métodos Wd M, Wd M-LSH, Wd-G M e Wd-G M-LSH.

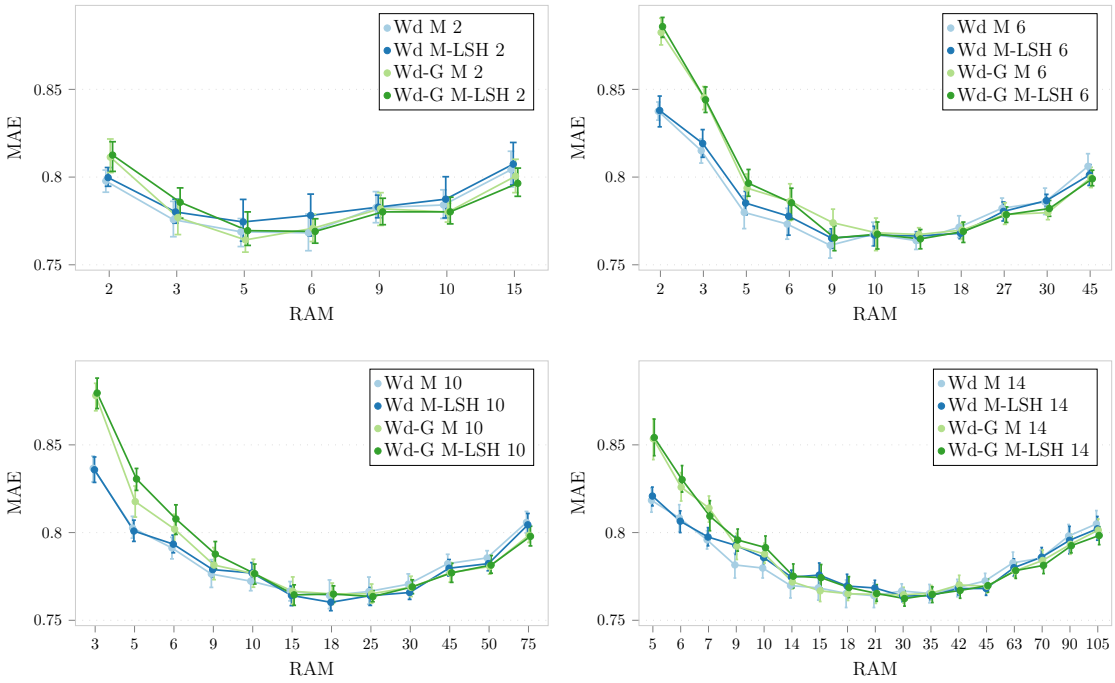


Figura 4.11: Variação dos pesos para (Nota Média Filme, Nota Média Usuário, Nota Média Sexo, Nota Média Ocupação, Nota Média idade por grupo)

Neste experimento, nota-se que foi utilizado a técnica do LSH para o cálculo da média. Deste modo verificamos a interferência da variação do peso para os métodos com a presença do LSH. Contudo, as abordagens com LSH seguiram de certa forma, o padrão dos resultados dos métodos sem o LSH. Os melhores resultados podem ser vistos na Tabela 4.8.

Tabela 4.8: Melhores resultados para o experimento III.

RAM	Método	Desvio Padrão	MAE
5	Wd-G M 2	0.010582	0.76420 (0,52%)
9	Wd M 6	0.013491	0.76112 (0,11%)
18	Wd M-LSH 10	0.007616	0.76026 (0,00%)
30	Wd-G M-LSH 14	0.007613	0.76239 (0,28%)

Outro experimento realizado é mostrado na Figura 4.12, onde comparamos os métodos da WiSARD utilizando todos os atributos, à direita, e utilizando somente os atributos de médias, à esquerda. Neste caso, podemos concluir que a utilização integral dos dados, ao que tudo indica é melhor para o algoritmo do WiSARD.

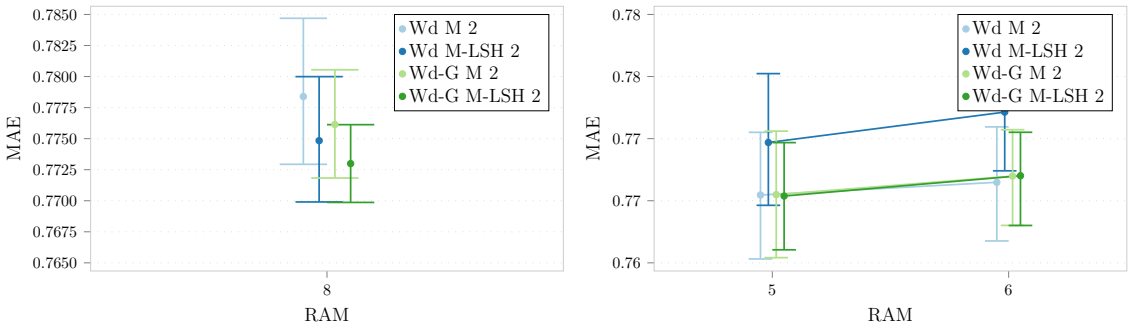


Figura 4.12: WISARD atributos notas médias x WISARD todos atributos.

4.4.4 Experimento IV

Neste experimento aplicamos a abordagem LSH junto com uma rede Neural de aprendizagem profunda e comparamos os métodos testados anteriormente. Sendo assim, vamos verificar o quanto genérico são os métodos abordados para a variação dos dados da base. Foi utilizado o k-fold-10 executado 5 vezes. A Figura 4.13 apresenta os resultados para o MAE e RMSE.

Analisando primeiramente os resultados para a Figura 4.13a, percebe-se que os algoritmos com a inclusão da abordagem LSH obtiveram melhores resultados ao compará-los com os mesmo sem a inclusão. A tabela 4.9 apresenta a porcentagem desta melhora.

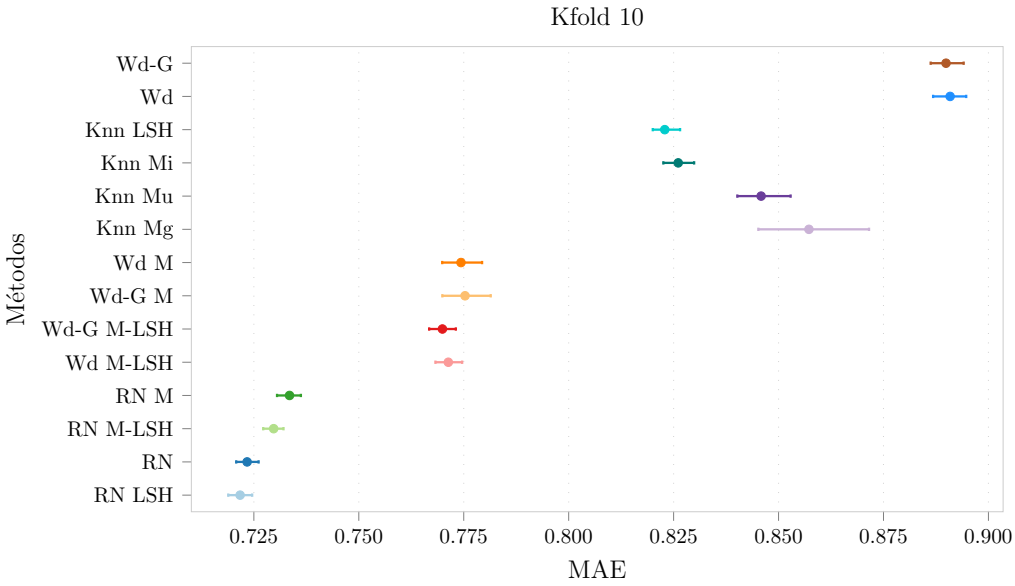
Tabela 4.9: Relação entre os métodos para Kfold-10.

Métodos LSH/Padrão	MAE	RMSE
RN LSH e RN	-0,23%	-0,11%
Wd M-LSH e Wd M	-0,39%	-0,32%
Wd-G M-LSH e Wd-G M	-0,70%	-0,45%
Knn LSH e Knn Mi	-0,39%	-0,32%
Wd-G M-LSH e Knn Mi	-6,80%	-5,88%

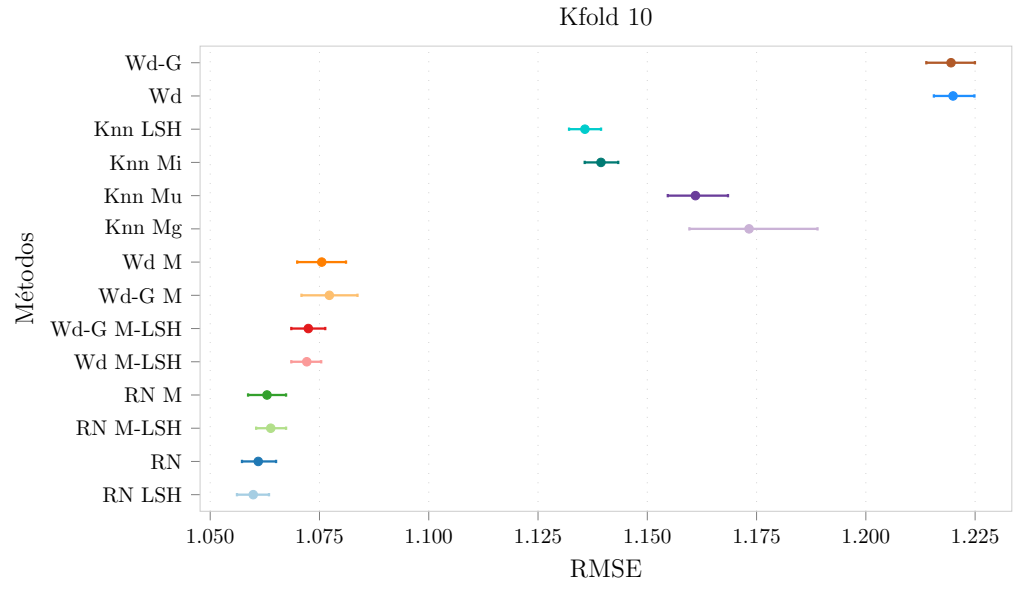
Na Tabela 4.10, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH. A rede neural *deep learning* com LSH atingiu o melhor resultado.

Tabela 4.10: Relação dos melhores métodos para Kfold-10.

Métodos	MAE	RMSE
RN	0,72339119 (0,00%)	1,0609884 (0,00%)
RN LSH	**0,7217254 (-0,23%)**	**1,0598396 (-0,11%)**
Wd-G M-LSH	0,769919 (6,43%)	1,072446 (1,08%)



(a) MAE



(b) RMSE

Figura 4.13: Resultado do MAE e RMSE para kfold-10.

Para o resultado do RMSE, Figura 4.13b, consta-se a permanência dos métodos com LSH sendo melhores que o sem LSH. A abordagem RN LSH obteve o RMSE

no valor de 1.05983 enquanto a RN atingiu 1.06098.

4.4.5 Experimento V

Para esse experimento utilizamos a base ajustada para CCS-100. Nosso objetivo é encontrar o valor para o *bleaching* para os métodos Wd M, Wd M-LSH, Wd-G M e Wd-G M-LSH. A Figura 4.14 mostra a variação do *bleaching* em comparação ao método de desempate utilizado em nosso trabalho.

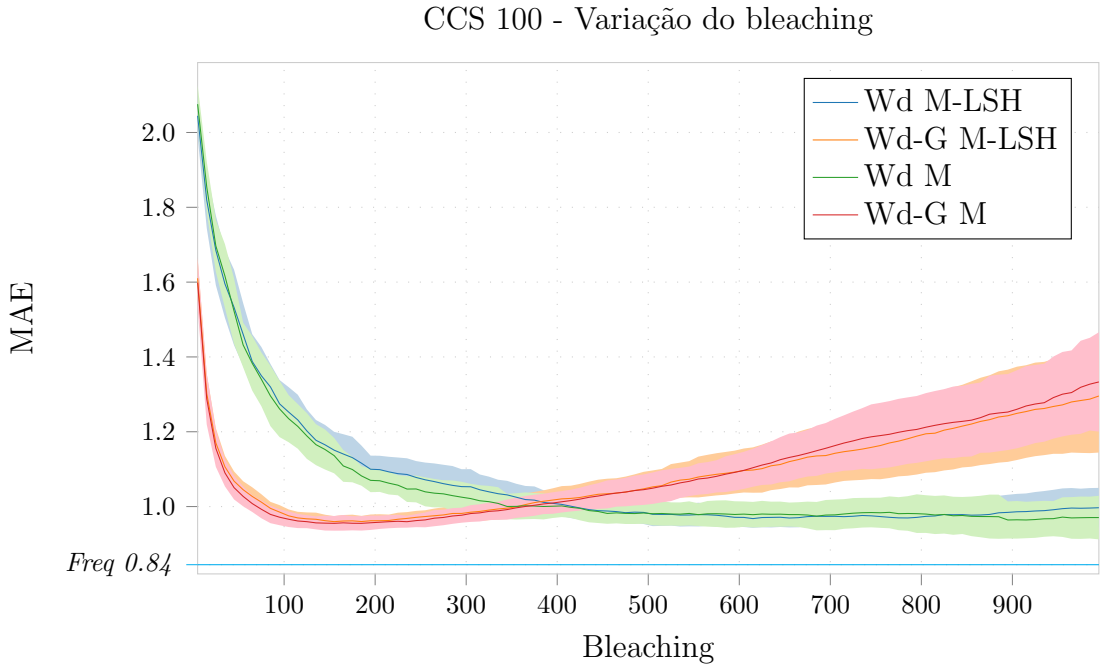


Figura 4.14: Variação do valor do bleaching para o WiSARD.

Podemos constatar que, independente do valor para o *bleaching*, o método baseado na frequência conseguiu melhor resultado com MAE de 0.8446. Um ponto que podemos destacar é a mudança de desempenho da WiSARD, ocorrida por volta do valor 200 para o *bleaching* nas abordagens baseadas em gêneros.

4.4.6 Experimento VI

Neste experimento utilizamos a base ajustada para CCS-100. Nosso objetivo é encontrar o melhor k_lsh para a rede neural com LSH. A Figura 4.15 apresenta os resultados. Nota-se que o RN LSH 5 obteve melhor resultado, portanto 5 é o valor do k_lsh.

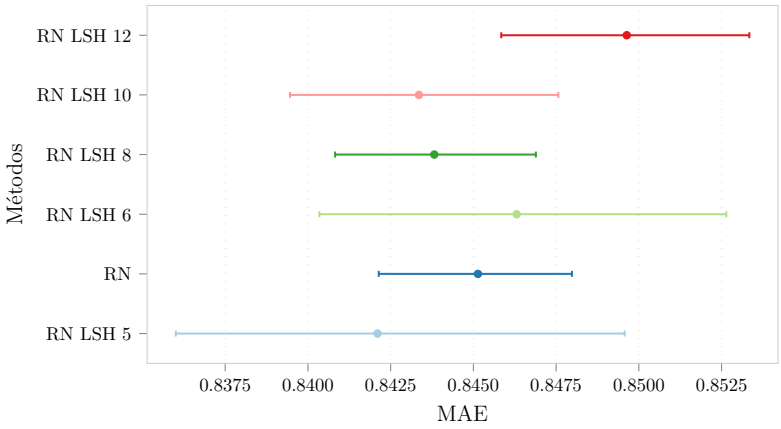
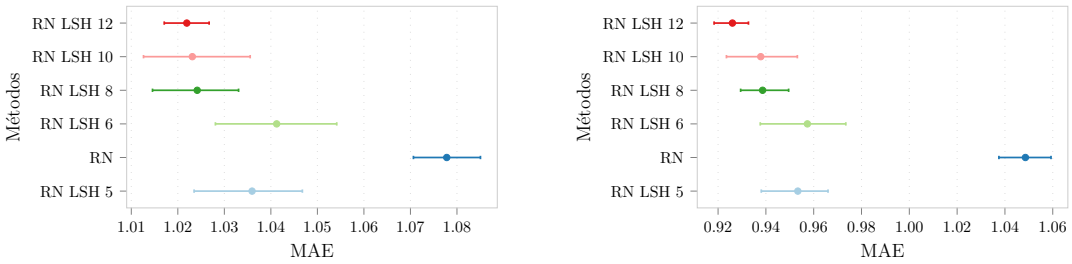


Figura 4.15: Resultado do MAE para variação do k vizinhos LSH para Rede Neural.

Na Figura 4.16a, mostramos os resultados para o MAE onde somente os itens avaliados com no máximo 9 notas são levados em consideração para o cálculo. Já na Figura 4.16b, apresentamos os resultados para o MAE dos itens que não receberam avaliações.



(a) CCS 100 - Intervalo 0 a 9

(b) CCS 100 - Itens com 0 rating

Figura 4.16: Resultado do MAE para variação do k vizinhos do LSH em relação a quantidade de avaliações dos itens.

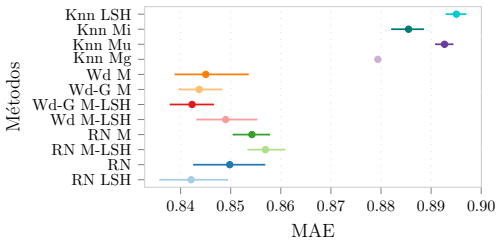
Percebe-se que a abordagem RN difere dos demais quando analisamos somente os itens com poucas avaliações. Por outro lado, os métodos com maiores k_lsh destacaram-se. Sendo assim, é possível perceber que aumentando a quantidade de itens similares no algoritmo melhora a previsão das notas dos itens com poucas avaliações.

4.4.7 Experimento VII

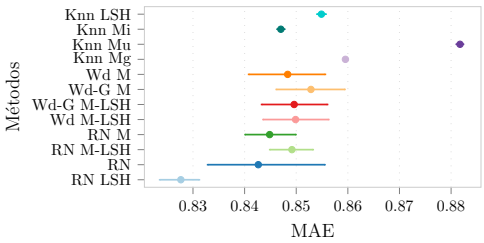
Após selecionar os dados da base, determinando uma quantidade L de itens com nenhuma avaliação, ou seja, selecionar itens que sofrem com o problema de *cold-start*. Comparamos as abordagens considerando a variação de itens para completo *cold-start*. A Figura 4.17 mostra os resultados para os 3 valores escolhidos para L.

Percebe-se que na Figura 4.17a, onde foram selecionado 100 itens com CCS, as abordagens utilizando WiSARD conseguiram superar a rede neural *deep learning*

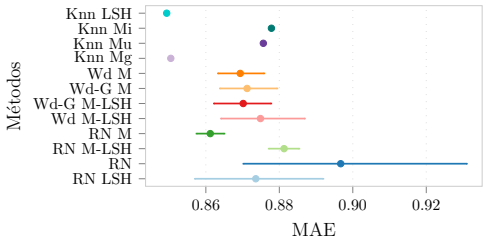
sem LSH, algo que não ocorreu no experimento 4.4.4 utilizando kfold-10.



(a) CCS 100



(b) CCS 200

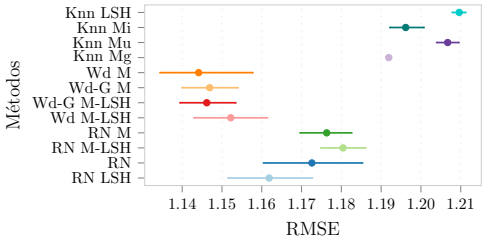


(c) CCS 300

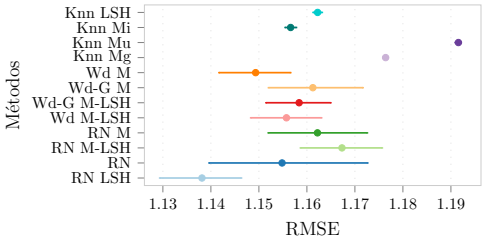
Figura 4.17: Resultado do MAE para completo cold-start (CCS).

Desta forma, a abordagem Wd-G M-LSH obteve um MAE de 0.84229 enquanto o RN ficou com 0.84981. Entretanto, o RN com LSH superou todos os outros métodos, obtendo o valor de 0.84210.

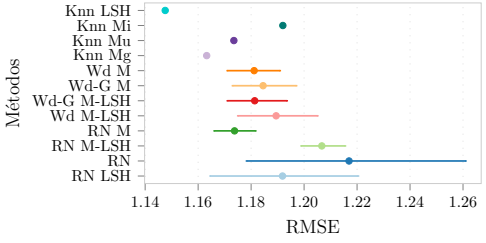
A Figura 4.18 apresenta os valores do RMSE obtidos pelos métodos em relação à variação da base de acordo com CCS.



(a) CCS 100



(b) CCS 200



(c) CCS 300

Figura 4.18: Resultado do RMSE para completo cold-start (CCS).

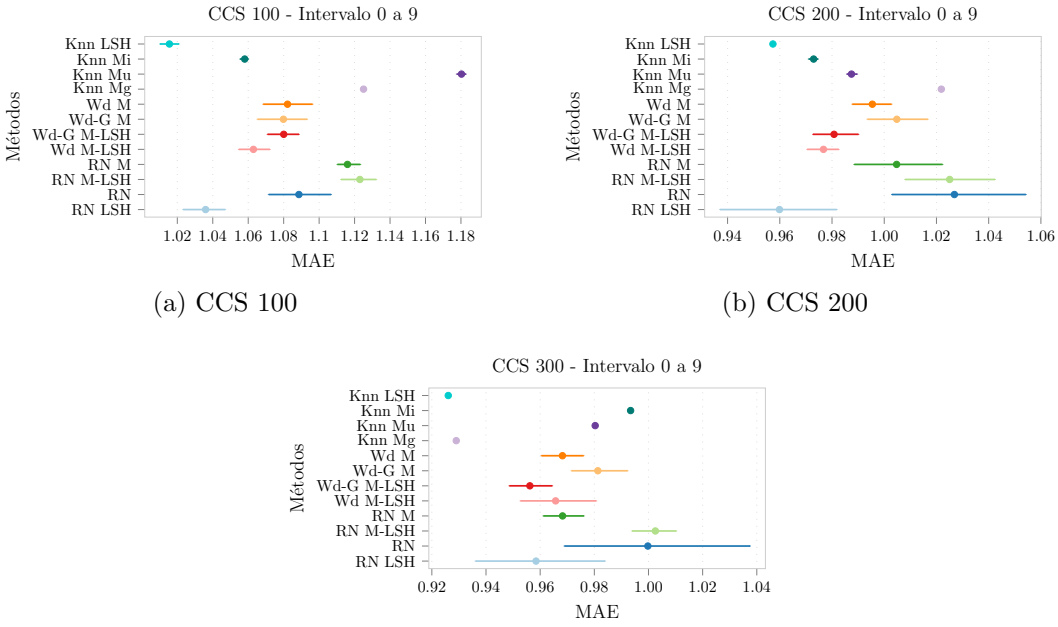
Na Tabela 4.11, destacamos os melhores resultados alcançados pelas abordagens

que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.11: Relação dos melhores métodos para CCS.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
RN	0,849811 (0,72%)	0,842654 (0,00%)	0,896706 (5,44%)	1,172601 (2,49%)	1,154828 (0,48%)	1,216967 (4,62%)
Wd-G M	0,84371 (0,00%)	0,852849 (1,21%)	0,87122 (2,44%)	1,14691 (0,24%)	1,16124 (1,04%)	1,18452 (1,83%)
Wd M	0,84502 (0,16%)	0,84834 (0,67%)	0,86936 (2,22%)	**1,14415 (0,00%)**	1,14932 (0,00%)	1,18115 (1,54%)
Knn Mg	0,87937 (4,23%)	0,85953 (2,00%)	0,85046 (0,00%)	1,19191 (4,17%)	1,1764 (2,36%)	1,16323 (0,00%)
RN LSH	**0,84210 (-0,19%)**	**0,827649 (-1,78%)**	0,873592 (2,72%)	1,161859 (1,55%)	**1,138145 (-0,97%)**	1,191858 (2,46%)
Wd-G M-LSH	0,84229 (-0,17%)	0,8496 (0,82%)	0,87017 (2,32%)	1,14618 (0,18%)	1,15838 (0,79%)	1,18136 (1,56%)
Knn LSH	0,89503 (6,08%)	0,85488 (1,45%)	**0,84934 (-0,13%)**	1,20963 (5,72%)	1,16223 (1,12%)	**1,14757 (-1,35%)**

A Figura 4.19 apresenta os valores do MAE obtidos pelos métodos em relação à variação da base de acordo com CCS, considerando apenas os itens que contém no máximo 9 avaliações.



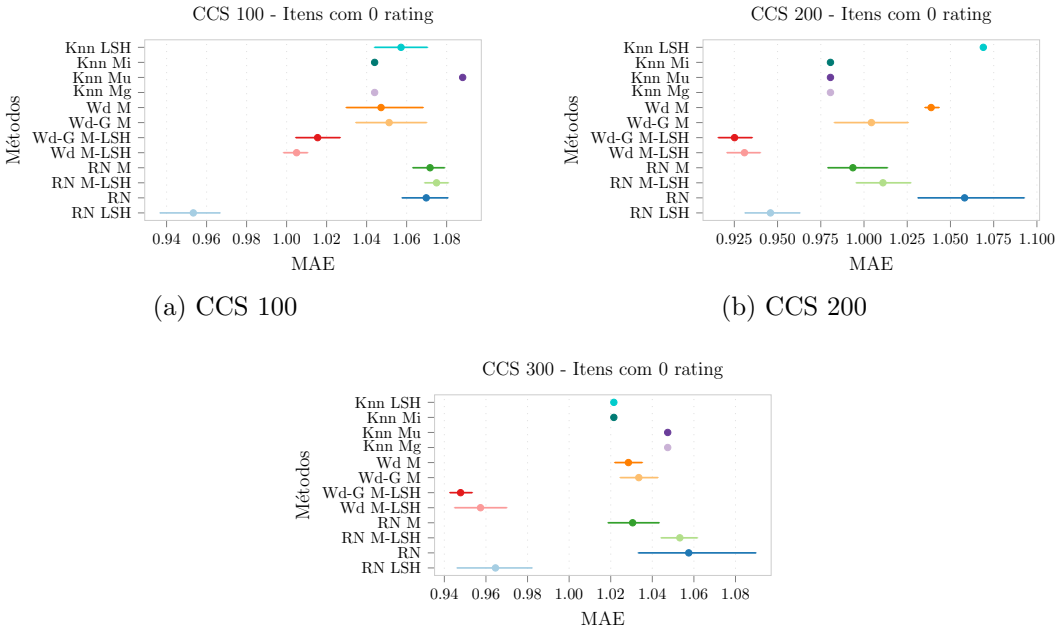
(a) CCS 100

(b) CCS 200

(c) CCS 300

Figura 4.19: Resultado do MAE para itens no intervalo de 0 a 9.

A Figura 4.20 apresenta os valores do MAE obtidos pelos métodos em relação à variação da base de acordo com CCS, considerando apenas os itens que não foram avaliados, ou seja, itens com completo *cold-start* presente na base das amostras.



(a) CCS 100

(b) CCS 200

(c) CCS 300

Figura 4.20: Resultado do MAE para itens com 0 rating.

Na Tabela 4.12, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.12: Relação dos melhores métodos para itens com 0 *ratings* e no intervalo de 0 a 9 *ratings* para ICS.

	MAE - 0 *ratings*			MAE - 0 a 9 *ratings*		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
Knn Mg	1,044 (0,00%)	0,9806 (0,00%)	1,0474 (2,54%)	1,12505 (6,33%)	1,02187 (5,03%)	0,9290099 (0,00%)
Knn Mi	1,044 (0,00%)	0,9806 (0,00%)	1,0215 (0,00%)	1,05803 (0,00%)	0,97296 (0,00%)	0,9933999 (6,93%)
RN LSH	**0,95336 (-8,68%)**	0,9459099 (-3,54%)	0,96458 (-5,57%)	1,03596 (-2,09%)	0,95988 (-1,34%)	0,9584699 (3,17%)
Wd M-LSH	1,00498 (-3,74%)	0,93087 (-5,07%)	0,95739 (-6,28%)	1,06294 (0,46%)	0,976729 (0,39%)	0,96571 (3,95%)
Wd-G M-LSH	1,0155 (-2,73%)	**0,92513 (-5,66%)**	**0,9478 (-7,21%)**	1,08003 (2,08%)	0,98078 (0,80%)	0,9561999 (2,93%)
Knn LSH	1,0572 (1,26%)	1,069 (9,01%)	1,0215 (0,00%)	**1,01555 (-4,02%)**	**0,95737 (-1,60%)**	**0,92606 (-0,32%)**

Aplicação do SVD com LSH

Agora vamos comparar os métodos após a aplicação da técnica de SVD para diminuição da dimensionalidade e recuperação das variáveis latentes. Desta forma, será definido um novo espaço utilizando a matriz de preferências das características dos usuários e dos itens.

Na figura 4.21, apresentamos os resultados para o MAE e RMSE para as abordagens com *deep learning* em diferentes cenários com variação na quantidade de itens em completo *cold-start*. Neste experimento implementamos uma rede neural para receber as variáveis latentes geradas pelo SVD, em conjunto com a lista de itens similares gerada pelo LSH.

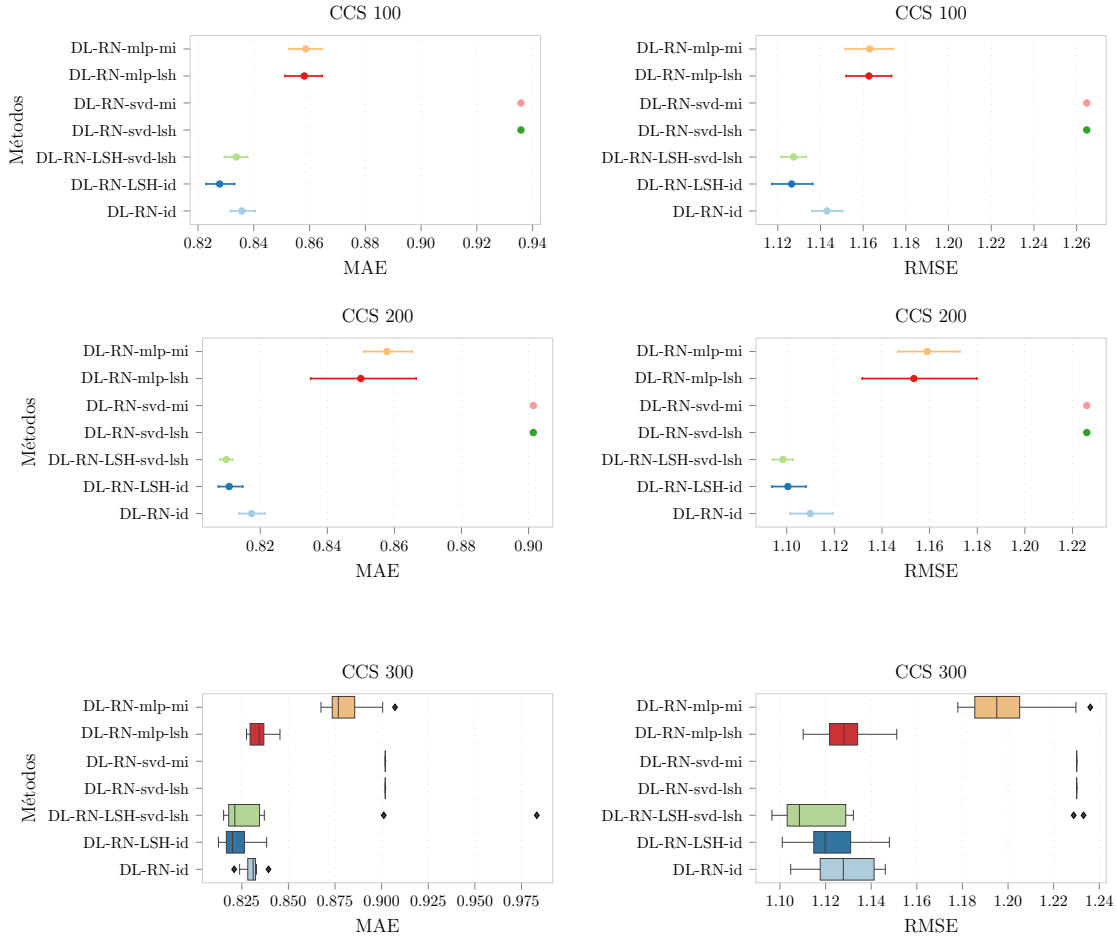


Figura 4.21: Resultado do MAE e RMSE para o método *deep learning* variando à base em CCS 100, 200 e 300.

Nota-se que as abordagens que utilizaram o uso da média dos itens similares, geradas pelo LSH, para o preenchimento das notas vazias da matriz de *ratings*, antes da aplicação do SVD, obtiveram melhores resultados tanto para o MAE e RMSE.

Verificamos que no método DL-RN-LSH-svd-lsh para o CCS 300, a média foi afetada por alguns resultados discrepantes.

Agora na figura 4.22, apresentamos os resultados para o MAE e RMSE para as abordagens com WiSARD. O método foi analisado em diferentes cenários, através da variação na quantidade de itens em completo *cold-start* e quantidade de memória (RAM) para os discriminadores. Neste experimento implementamos a WiSARD para receber as variáveis latentes geradas pelo SVD, utilizando a matriz normalizada pela média LSH.

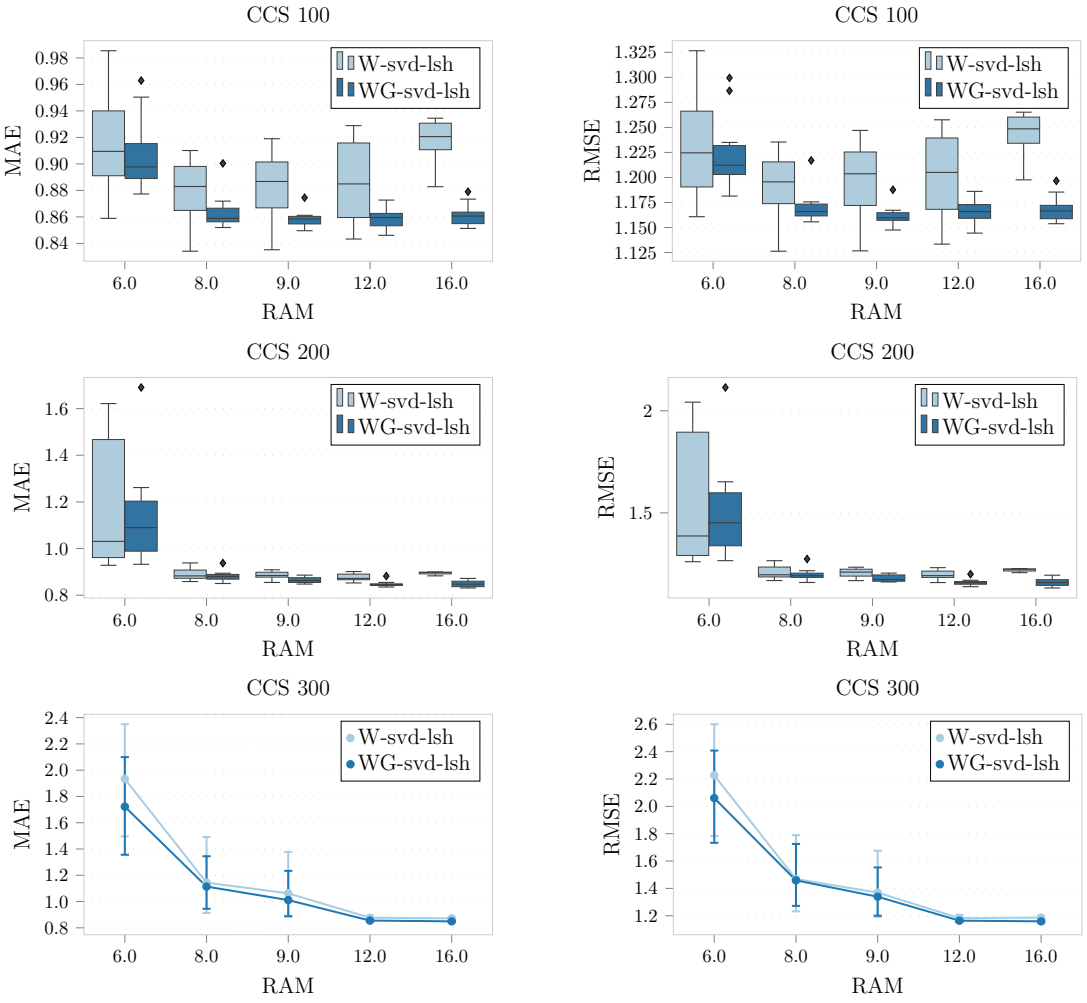


Figura 4.22: Resultado do MAE e RMSE para o método WiSARD variando à base em CCS 100, 200 e 300.

Percebe-se que a abordagem da WiSARD por gêneros (WG) obteve melhores resultados utilizando 12 memórias RAM para CCS 100 e 200 e 16 RAM para CCS 300, enquanto que a WiSARD por notas (W), atingiu seu melhor resultado usando 8, 12 e 16 RAM para CCS 100, 200 e 300, respectivamente.

É notável a eficiência da WG em relação ao W. Entretanto, é válido ressaltar que a WG tem um tempo de treinamento maior que o W, podendo ser visto nos experimentos IX, na seção 4.4.9.

Os resultados para as abordagens do algoritmo de Floresta Aleatória (RF) podem ser visto na Figura 4.23. Neste experimento preparamos a RF para receber as variáveis latentes da normalização pela média do usuário (mu) e pela média LSH (lsh). Também agregamos no experimento uma abordagem que utiliza a lista de similaridade LSH (LSH-mu e LSH-lsh). Por fim, variamos a quantidade de números de árvores geradas.

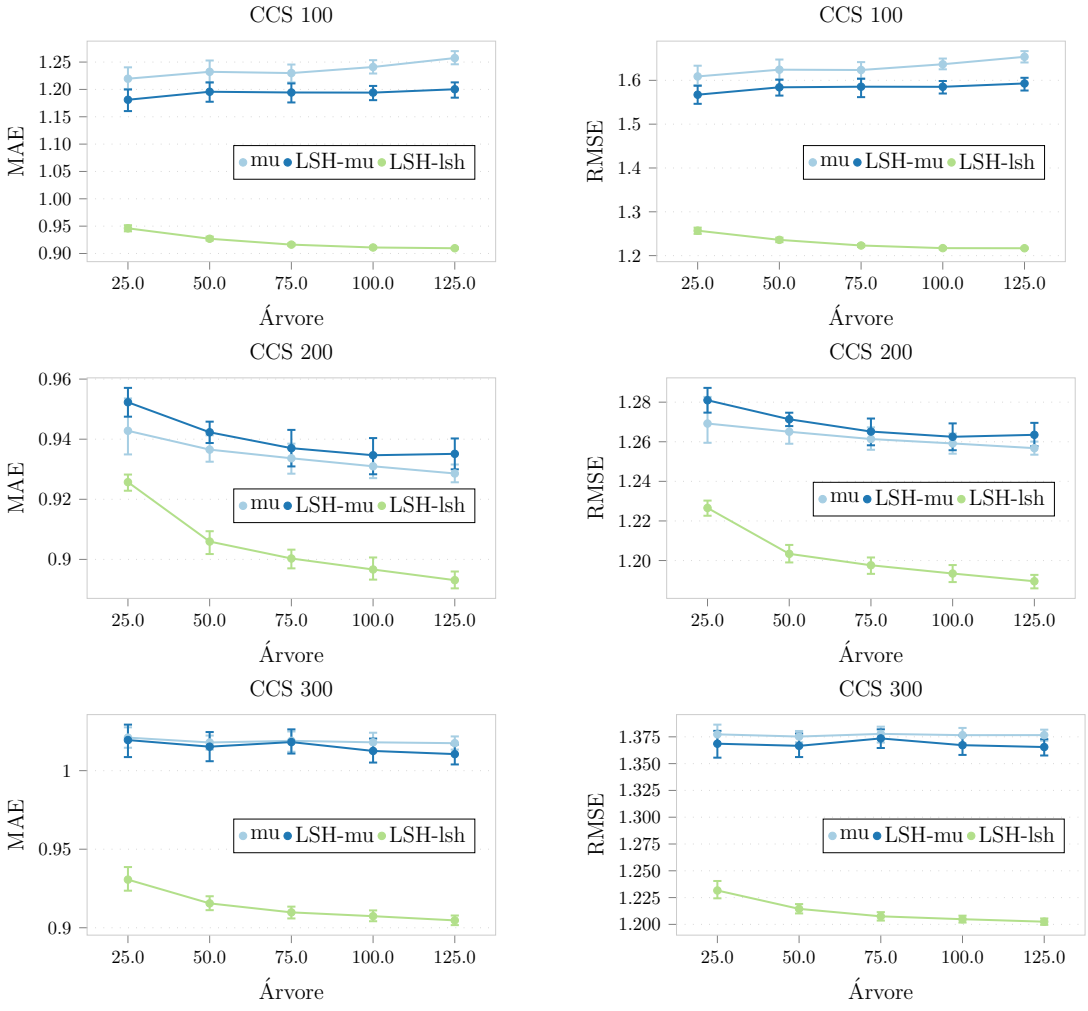


Figura 4.23: Resultado do MAE e RMSE para o método Floresta Aleatória variando à base em CCS 100, 200 e 300.

Em todos cenários analisados neste experimento, foi constatado a eficiência da abordagem de RF LSH-lsh. Este método utilizou a matriz de similaridade LSH acoplado com as variáveis latentes, geradas pelo SVD com a normalização da matriz pela média dos itens similares.

A Figura 4.24 mostra os resultados para o MAE e RMSE dos experimentos para as abordagens com KNN utilizando as variáveis latentes para diferentes valores de k. As variáveis foram geradas a partir da normalização da matriz de preferências pela média do item, média usuário, média LSH e média global. Testamos também a combinação da lista de similaridade LSH agregada as variáveis latentes da normalização pela média LSH.

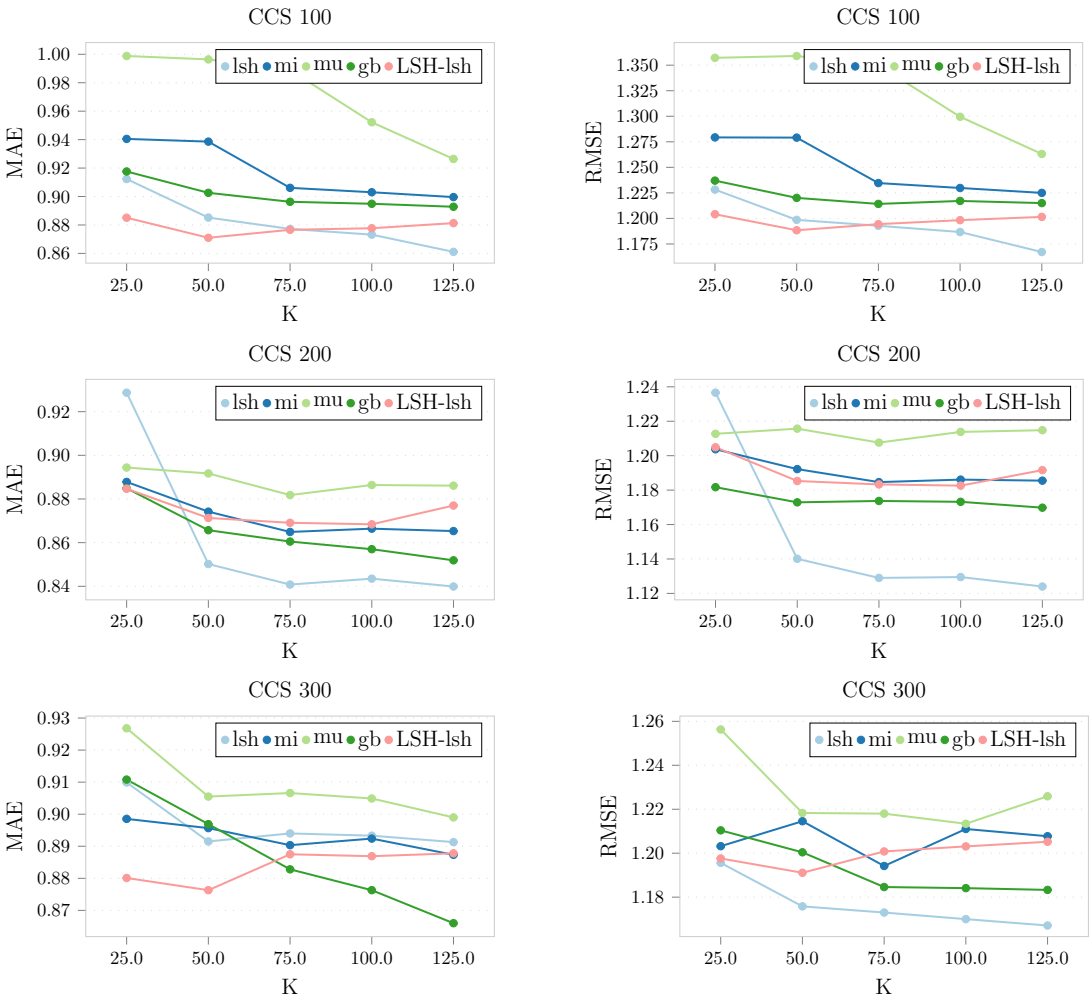


Figura 4.24: Resultado do MAE e RMSE para o método knn variando à base em CCS 100, 200 e 300.

A abordagem do KNN foi muito sensível a trocas de parâmetros. Foi o método que mais sofreu variações nos resultados, por causa destas parametrizações inicias.

Por outro lado, percebe-se que o KNN mu, não obteve bons resultados perante aos outros métodos. A abordagem usando média global, melhora o resultado quando aumenta o valor do K, podendo ser visto mais claramente na base CCS 300. Já na abordagem LSH-lsh, os resultados aproximadamente manteve-se nas 3 bases analisadas. Além disso podemos concluir que o valor do k para esse método deve ser

menor que 75. A abordagem usando média lsh na maioria dos casos obteve o melhor resultado, tanto para o MAE e o RMSE.

A figura 4.25, apresenta os melhores resultados para o MAE e RMSE alcançados pelos métodos que utilizaram as técnicas de SVD para recuperar as variáveis latentes.

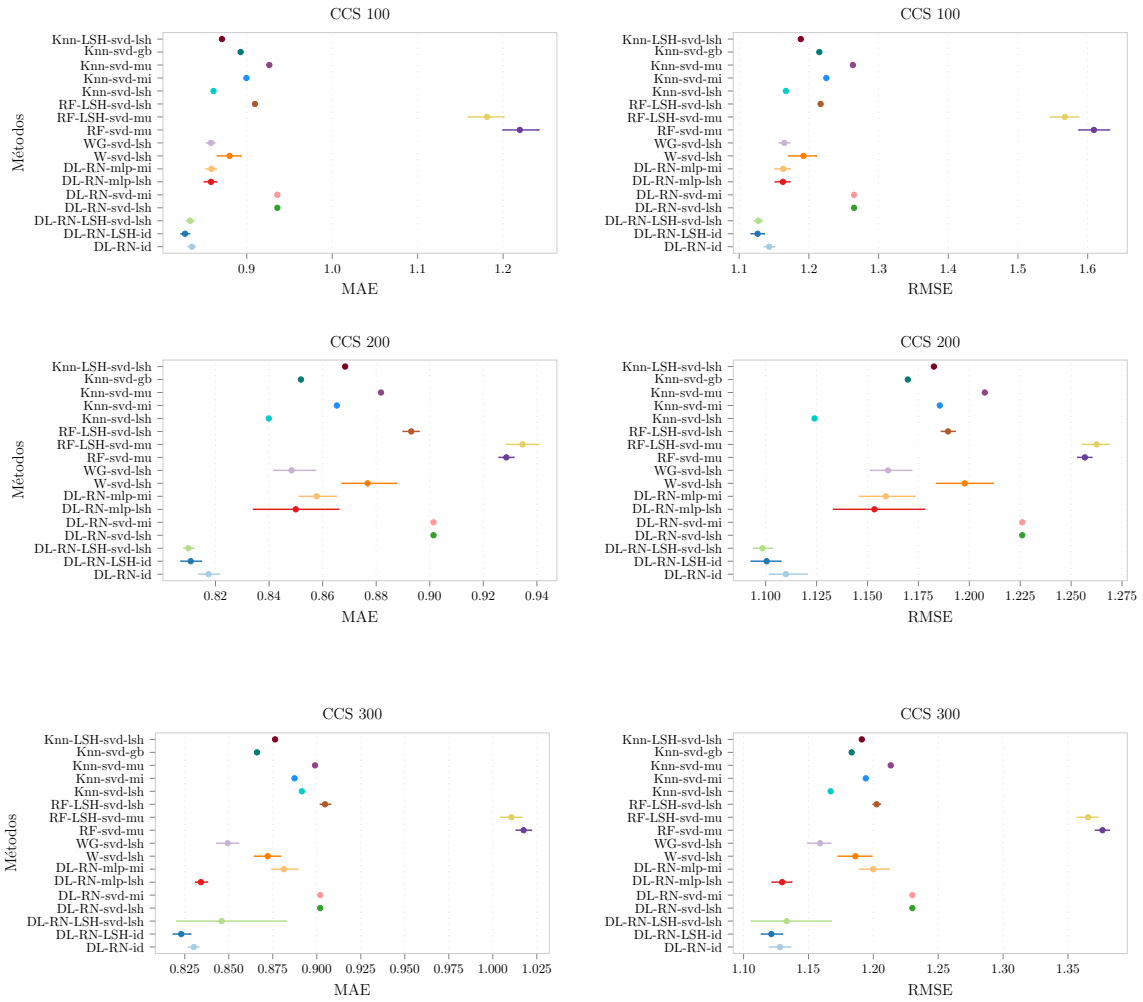


Figura 4.25: Comparação do MAE e RMSE para os melhores resultados alcançados pelos métodos analisados.

Pela análise dos gráficos é possível afirmar que os métodos dos quais utilizaram a matriz de similaridade, gerada pelo LSH, atingiu melhores resultados tanto no MAE e RMSE. As abordagens com *deep learning* foram superiores aos demais, já que os 3 primeiros resultados foram métodos que utilizaram esta técnica.

Vale ressaltar que a WiSARD por gêneros conseguiu resultados melhores que algumas abordagens tradicionais da literatura. Outra abordagem que merece destaque é o knn utilizando a média por LSH, que ficou entre os melhores das abordagens para o knn.

Na Tabela 4.13, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.13: Relação dos melhores métodos para CCS com SVD.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
DL-RN-id	0,835719 (0,0%)	0,81741 (0,0%)	0,830128 (0,0%)	1,143025 (0,0%)	1,109887 (0,0%)	1,128042 (0,0%)
DL-RN-LSH-id	**0,827747 (-0,95%)**	0,810741 (-0,82%)	**0,822942 (-0,87%)**	**1,126462 (-1,45%)**	1,10047 (-0,85%)	**1,121411 (-0,59%)**
DL-RN-LSH-svd-lsh	0,833724 (-0,24%)	**0,809876 (-0,92%)**	0,845966 (1,91%)	1,127429 (-1,36%)	**1,098388 (-1,04%)**	1,133233 (0,46%)

Agora vamos mostrar através da Figura 4.26 os resultados obtidos para o MAE e RMSE, levando em consideração somente os itens que não receberam avalizações, ou seja, itens que na fase de treinamento continham 0 *ratings*, sendo assim um item com completo *cold-start*.

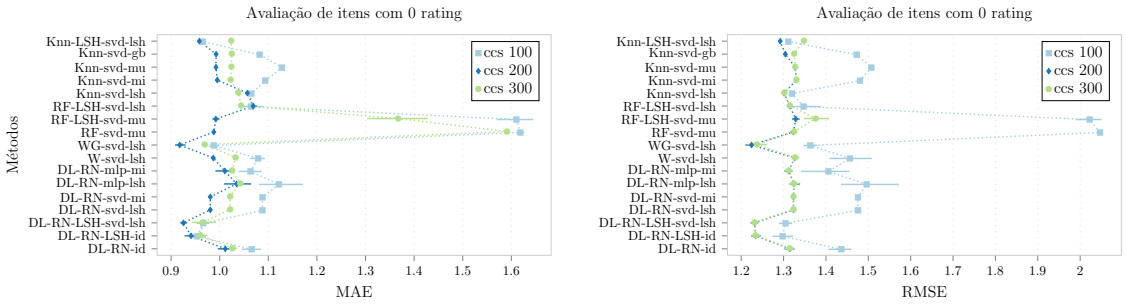


Figura 4.26: Resultado do MAE e RMSE para itens sem avaliação com SVD.

Ao analisar somente os itens que sofrem de *cold-start*, o método WG conseguiu melhores resultados para a base de CCS 200 para o MAE e RMSE. Enquanto os métodos DL-RN-LSH-id e DL-RN-LSH-svd-lsh ficaram com os melhores resultados nas demais bases.

Na Tabela 4.14, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.14: Relação dos melhores métodos para itens com 0 *ratings* para CCS com SVD.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
DL-RN-mlp-mi	1,06328 (0,00%)	1,01059 (3,06%)	1,0254 (0,38%)	1,40581 (0,00%)	1,31189 (0,60%)	1,38117 (8,16%)
DL-RN-svd-mi	1,088 (2,32%)	0,9806 (0,00%)	1,0215 (0,00%)	1,4757 (4,97%)	1,3234 (1,48%)	1,3781(7,92%)
Knn-svd-gb	1,0822 (1,78%)	0,9926 (1,22%)	1,0248 (0,32%)	1,4725 (4,74%)	1,3041 (0,00%)	1,277(0,00%)
DL-RN-LSH-id	**0,95296 (-10,38%)**	0,94113 (-4,03%)	**0,96013 (-6,01%)**	**1,29774 (-7,69%)**	1,23419 (-5,36%)	1,28179 (0,38%)
DL-RN-LSH-svd-lsh	0,96615 (-9,13%)	0,9253 (-5,64%)	0,96569 (-5,46%)	1,30459 (-7,20%)	1,23147 (-5,57%)	**1,26797 (-0,71%)**
WG-svd-lsh	0,98776 (-7,10%)	**0,91775 (-6,41%)**	0,96918 (-5,12%)	1,36308 (-3,04%)	**1,22442 (-6,11%)**	1,28972 (1,00%)

Por fim vamos mostrar através da Figura 4.27 os resultados obtidos para o MAE e RMSE, levando em consideração os itens que receberam entre 0 a 9 avalizações, ou seja, itens que na fase de treinamento continham no máximo 9 *ratings*.

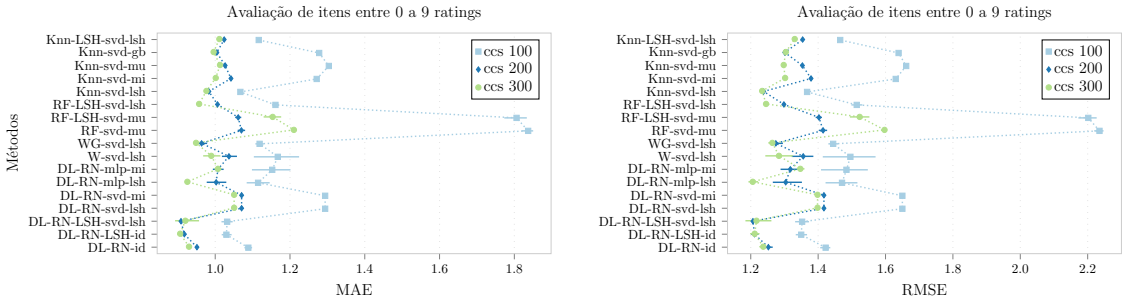


Figura 4.27: Resultado do MAE e RMSE para itens com 0 a 9 *ratings*.

Nota-se que as abordagens com LSH, obtiveram bons resultados para os itens com 0 a 9 *ratings*, confirmando então, os resultados obtidos anteriormente ao analisar todo o conjunto de teste.

Na Tabela 4.15, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.15: Relação dos melhores métodos para itens com 0 a 9 *ratings* para CCS com SVD.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
DL-RN-id	1,08858 (0,00%)	0,95113 (0,00%)	0,92995 (0,00%)	1,4223 (0,00%)	1,25226 (0,00%)	1,23715 (0,00%)
DL-RN-LSH-id	**1,0299 (-5,39%)**	0,91742 (-3,54%)	**0,90623 (-2,55%)**	**1,34973 (-5,10%)**	1,21135 (-3,27%)	1,21163 (-2,06%)
DL-RN-LSH-svd-lsh	1,03188 (-5,21%)	**0,90847 (-4,49%)**	0,92023 (-1,05%)	1,35247 (-4,91%)	**1,20648 (-3,66%)**	1,21706 (-1,62%)
DL-RN-mlp-lsh	1,11467 (2,40%)	1,00253 (5,40%)	0,9255 (-0,48%)	1,47089 (3,42%)	1,30376 (4,11%)	**1,20589 (-2,53%)**

4.4.8 Experimento VIII

Neste experimento vamos comparar os métodos para a variação dos itens para incompleto *cold-start* (ICS).

Para selecionar os dados da base, determinamos uma quantidade L de itens com poucas avaliações, definimos que itens com até 5 *ratings* seriam selecionados até completar a marca determinada por L. Em seguida comparamos as abordagens considerando a variação de itens para incompleto *cold-start*.

A Figura 4.28 mostra os resultados do MAE e RMSE obtidos pelos métodos em relação à variação da base de acordo com os valores determinado para L.

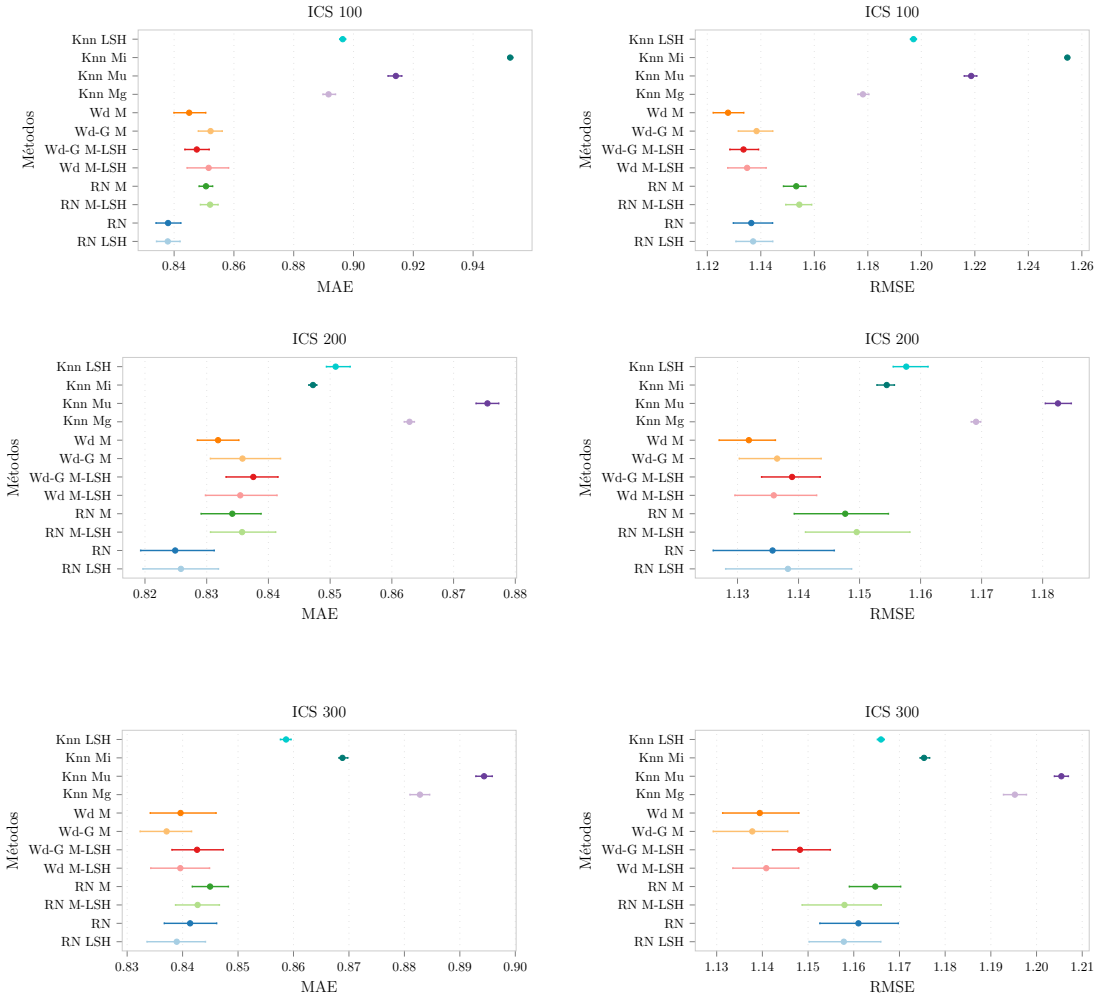


Figura 4.28: Comparação do MAE e RMSE para os melhores resultados alcançados pelos métodos analisados.

Percebe-se que a rede neural obteve em geral os melhores resultados para o MAE, sendo superado somente pela abordagem WiSARD por gêneros usando as médias

das notas no teste para ICS 300. Entretanto, as abordagens com WiSARD foram melhores em relação ao RMSE, principalmente nos teste com ICS 300.

Na Tabela 4.16, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.16: Relação dos melhores métodos para ICS.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
RN	0,837976 (0,00%)	**0,824896 (0,00%)**	0,841357 (0,51%)	1,136446 (0,77%)	1,135743 (0,34%)	1,161024 (2,04%)
Wd-G M	0,85219 (1,70%)	0,8358 (1,32%)	**0,837109 (0,00%)**	1,13841 (0,95%)	1,13649 (0,41%)	1,13779 (0,00%)
Wd M	0,84504 (0,84%)	0,83184 (0,84%)	0,83963 (0,30%)	**1,12775 (0,00%)**	**1,13186 (0,00%)**	1,13943 (0,14%)
RN LSH	**0,837878 (-0,01%)**	0,82582399 (0,11%)	0,83894 (0,22%)	1,137131 (0,83%)	1,13826 (0,57%)	1,157821 (1,76%)
Wd M-LSH	0,8515499 (1,62%)	0,8354399 (1,28%)	0,8395899 (0,30%)	1,13488 (0,63%)	1,13594 (0,36%)	1,14085 (0,27%)
Wd-G M-LSH	0,84758 (1,15%)	0,83755 (1,53%)	0,84261 (0,66%)	1,13354 (0,51%)	1,13894 (0,63%)	1,14823 (0,92%)

A Figura 4.29 apresenta na coluna da direita os valores do MAE obtidos pelos métodos em relação à variação da base de acordo com ICS, considerando apenas os itens que não foram avaliados, ou seja, itens com completo *cold-start* presente na base das amostras ICS. Já na coluna da esquerda mostra os valores do MAE obtidos pelos métodos em relação à variação da base de acordo com ICS, considerando apenas os itens que contém no máximo 9 avaliações.

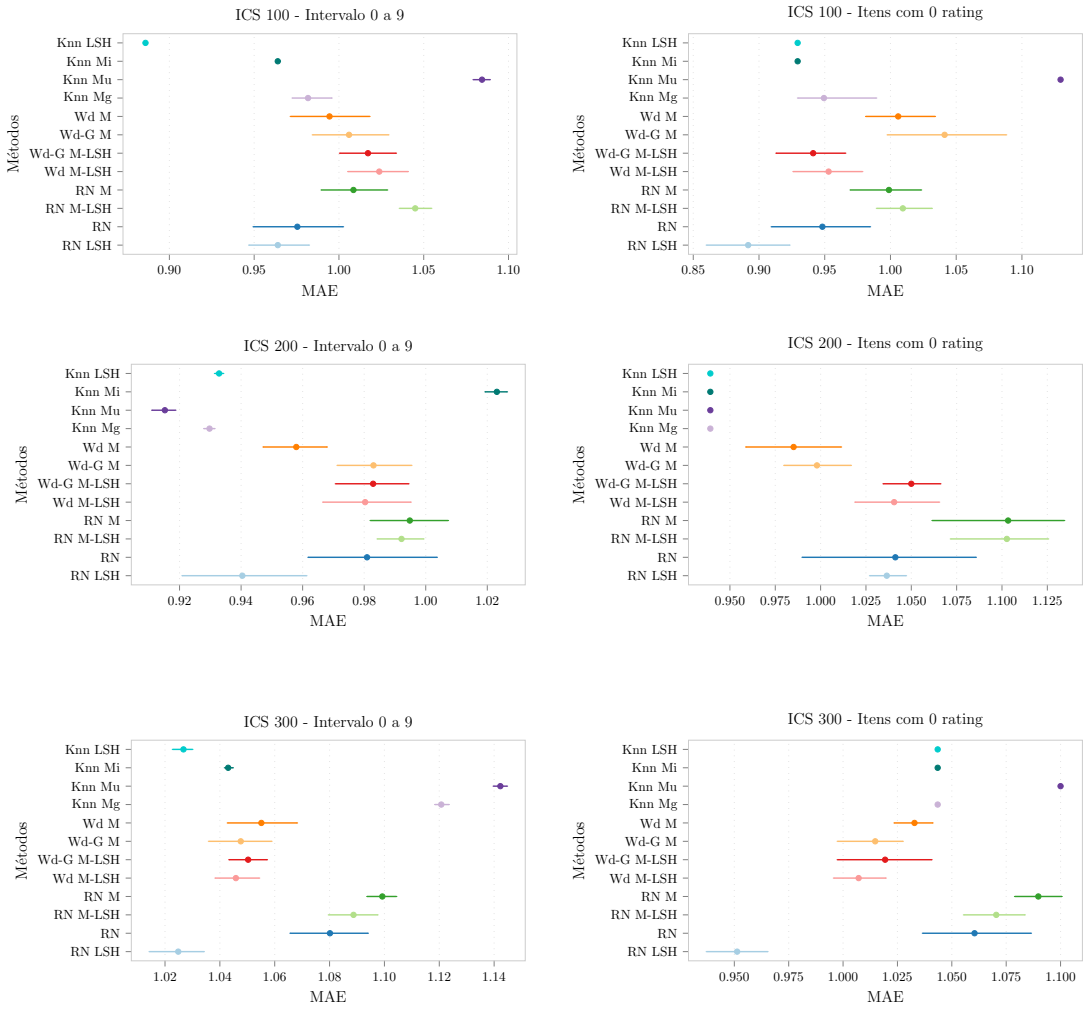


Figura 4.29: Comparação do MAE para itens no intervalo de 0 a 9 *ratings* e para itens com 0 *rating*.

Na Tabela 4.17, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.17: Relação dos melhores métodos para itens com 0 *ratings* e no intervalo de 0 a 9 *ratings* para ICS.

	MAE - 0 *ratings*			MAE - 0 a 9 *ratings*		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
Knn Mu	1,1294 (21,52%)	0,939199 (0,00%)	1,1 (8,40%)	1,08438 (12,50%)	**0,91523 (0,00%)**	1,14227 (9,51%)
Knn Mi	0,929399 (0,00%)	0,939199 (0,00%)	1,0435 (2,83%)	0,963919 (0,00%)	1,02311 (11,79%)	1,04303 (0,00%)
RN LSH	**0,89176 (-4,05%)**	1,03648 (10,36%)	**0,9513 (-6,26%)**	0,96395 (0,003%)	0,94041 (2,75%)	**1,02482 (-1,75%)**
Knn LSH	0,929399 (0,00%)	0,939199 (0,00%)	1,0435 (2,83%)	**0,88591 (-8,09%)**	0,93285 (1,93%)	1,02673 (-1,56%)

Aplicação do SVD com LSH

Agora vamos comparar os métodos após a aplicação da técnica de SVD para o problema de ICS.

A figura 4.30, apresenta os melhores resultados para o MAE e RMSE alcançados para os métodos que utilizaram as variáveis latentes e suas combinações como entradas. Neste experimento mostra a eficácia dos métodos em relação à variação das amostras de ICS 100, 200 e 300.

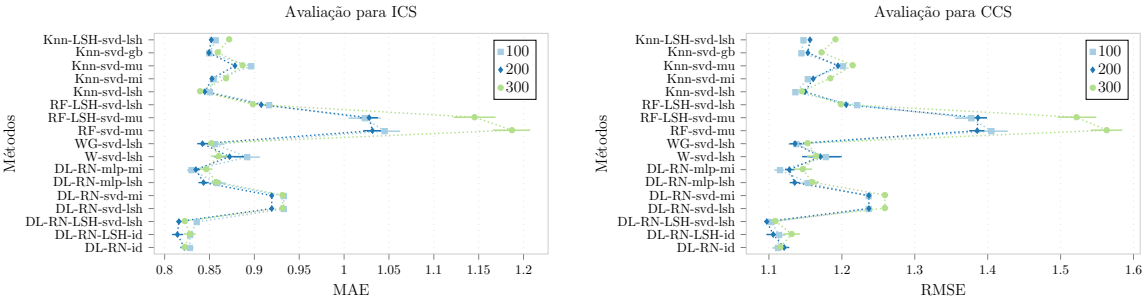


Figura 4.30: Comparação do MAE e RMSE para os melhores resultados alcançados pelos métodos analisados.

Na Tabela 4.18, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.18: Relação dos melhores métodos para ICS com SVD.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
DL-RN-id	**0,82807 (0,00%)**	0,82187 (0,00%)	**0,822467 (0,00%)**	1,112513 (0,00%)	1,120867 (0,00%)	1,115725 (0,00%)
DL-RN-LSH-id	0,828476 (0,05%)	**0,814187 (-0,93%)**	0,827977 (0,67%)	1,113861 (0,12%)	1,105859 (-1,34%)	1,131158 (1,38%)
DL-RN-LSH-svd-lsh	0,835746 (0,93%)	0,815866 (-0,73%)	0,822473 (0,001%)	**1,1034 (-0,82%)**	**1,097062 (-2,12%)**	**1,108728 (-0,63%)**

Agora vamos mostrar através da Figura 4.31 os resultados obtidos para o MAE e RMSE, levando em consideração somente os itens que não receberam avalizações, ou seja, itens que na fase de treinamento continham 0 *ratings*, sendo assim um item com completo *cold-start*. A quantidade de itens que sofrem com CCS pode ser vista na tabela 4.5.

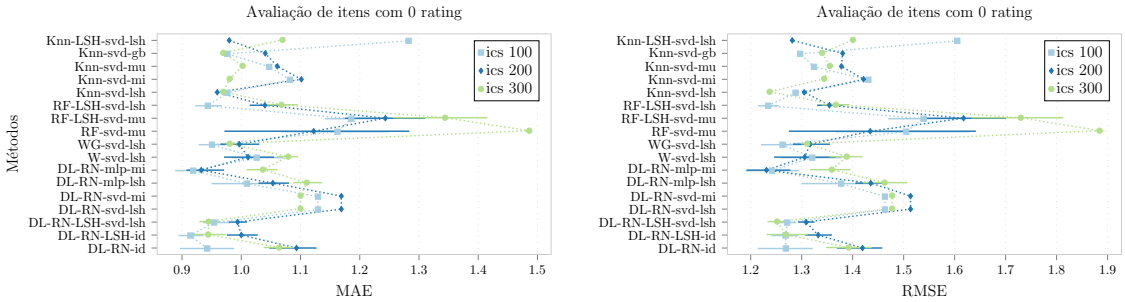


Figura 4.31: Resultado do MAE e RMSE para itens sem avaliação.

Na Tabela 4.19, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.19: Relação dos melhores métodos para itens com 0 a 9 *ratings* para ICS com SVD.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
DL-RN-id	0,92528 (2,81%)	0,96642 (1,80%)	1,0695 (1,56%)	1,21286 (0,05%)	1,28459 (2,75%)	1,38802 (0,00%)
DL-RN-mlp-mi	**0,9 (0,00%)**	0,94932 (0,00%)	1,08258 (2,80%)	**1,21227 (0,00%)**	1,25016 (0,00%)	1,40847 (1,47%)
Knn-svd-gb	0,9628 (6,98%)	1,0425 (9,82%)	1,0531 (0,00%)	1,257 (3,69%)	1,3808 (10,45%)	1,4105 (1,62%)
DL-RN-LSH-svd-lsh	0,95131 (5,70%)	**0,88745 (-6,52%)**	1,01437 (-3,68%)	1,23528 (1,90%)	**1,18524 (-5,19%)**	1,32476 (-4,56%)
Knn-svd-lsh	0,9665 (7,39%)	1,0114 (6,54%)	**0,9001 (-14,53%)**	1,2991 (7,16%)	1,3343 (6,73%)	**1,2184 (-12,22%)**

Por fim vamos mostrar através da Figura 4.32 os resultados obtidos para o MAE e RMSE, levando em consideração os itens que receberam entre 0 a 9 avalizações, ou seja, itens que na fase de treinamento continham no máximo 9 *ratings*.

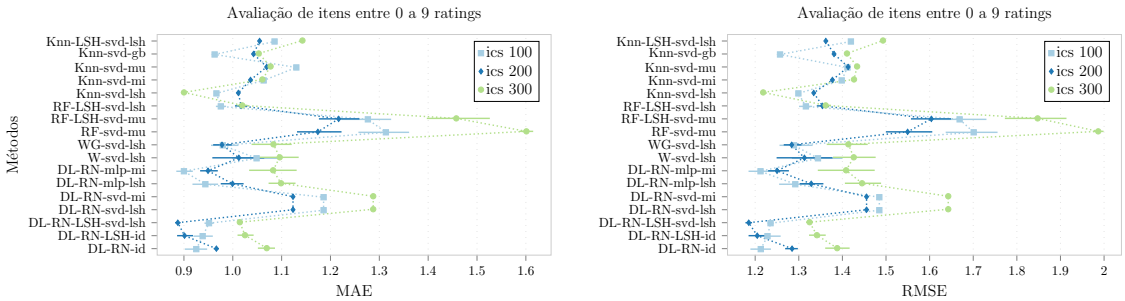


Figura 4.32: Resultado do MAE e RMSE para itens com 0 a 9 *ratings*.

Na Tabela 4.20, destacamos os melhores resultados alcançados pelas abordagens que utilizaram LSH em comparação aos melhores resultados obtidos pelos métodos tradicionais sem o LSH.

Tabela 4.20: Relação dos melhores métodos para itens com 0 *ratings* para ICS com SVD.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
DL-RN-mlp-mi	0,91884 (0,00%)	**0,93241 (0,00%)**	1,03652 (6,90%)	1,24179 (0,00%)	**1,23089 (0,00%)**	1,35916 (1,43%)
Knn-svd-gb	0,9765 (6,28%)	1,0405 (11,59%)	0,9696 (0,00%)	1,2971 (4,45%)	1,3804 (12,15%)	1,34 (0,00%)
DL-RN-LSH-id	**0,91412 (-0,51%)**	1,00001 (7,25%)	**0,94413 (-2,63%)**	1,26831 (2,14%)	1,33236 (8,24%)	1,26884 (-5,31%)
Knn-svd-lsh	0,9765 (6,28%)	0,9595 (2,91%)	0,9696 (0,00%)	1,288 (3,72%)	1,3049 (6,01%)	**1,2371 (-7,68%)**
RF-LSH-svd-lsh	0,94353 (2,69%)	1,03988 (11,53%)	1,06783 (10,13%)	**1,23436 (-0,60%)**	1,35434 (10,03%)	1,36677 (2,00%)

4.4.9 Experimento IX

Nesta seção vamos apresentar as performances dos algoritmos, mostrando os tempos praticados para treinamento e pré-processamento para os métodos utilizados. Desta forma possibilita analisá-los em situações onde teriam o melhor resultado.

A Figura 4.33, mostra o tempo de treinamento da WiSARD em relação à variação do peso dos atributos, realizado nos experimentos II e III, na seção 4.4.2.

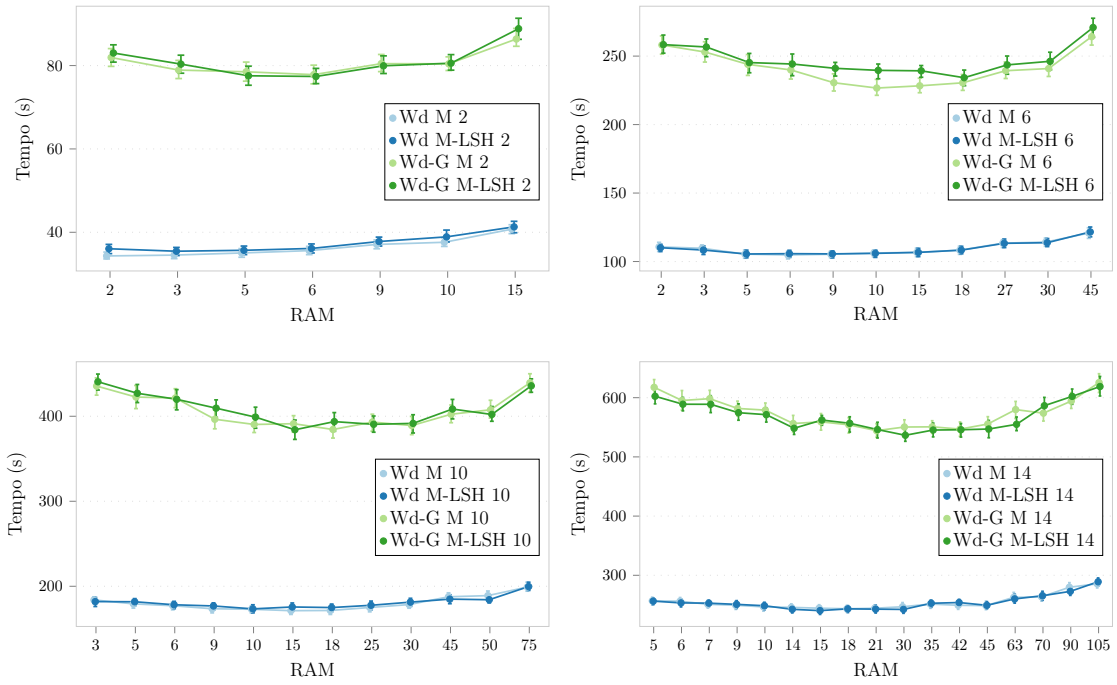


Figura 4.33: Tempo de Treinamento da WiSARD para Kfold-10 em relação a variação do peso dos atributos.

Percebe-se que quando aumentamos o peso dos atributos o tempo de treinamento também sofre um aumento. Como por exemplo, a abordagem Wd M 14, que ficou aproximadamente 6 vezes mais lento em comparação ao Wd M 2.

Analisamos os tempos de treinamento para as abordagens que utilizaram as variáveis latentes extraídas pelo SVD. As Figuras 4.34 e 4.35 mostram os resultados para CCS 100 e CCS 300, respectivamente.

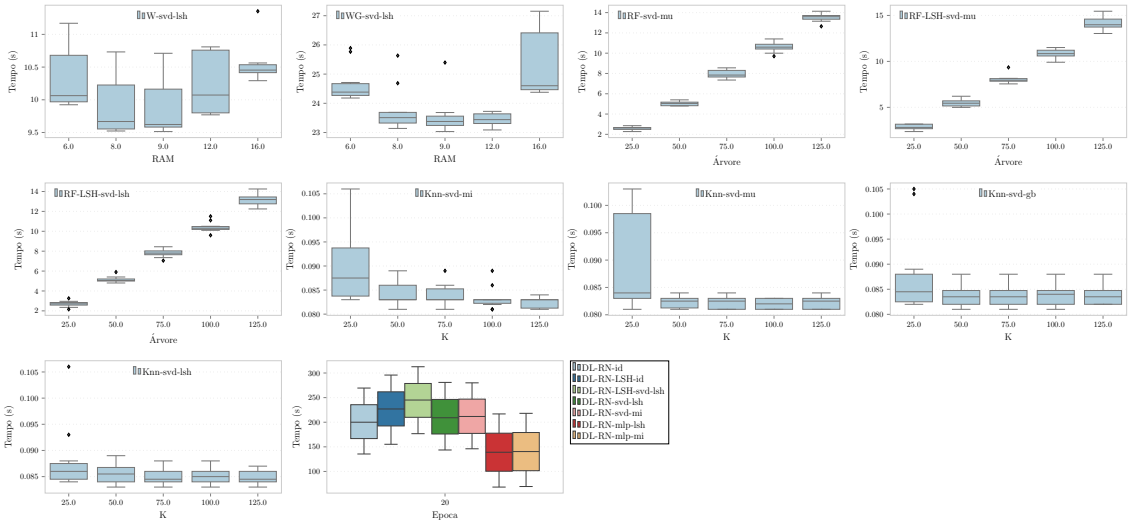


Figura 4.34: Tempo de treino para CCS 100.

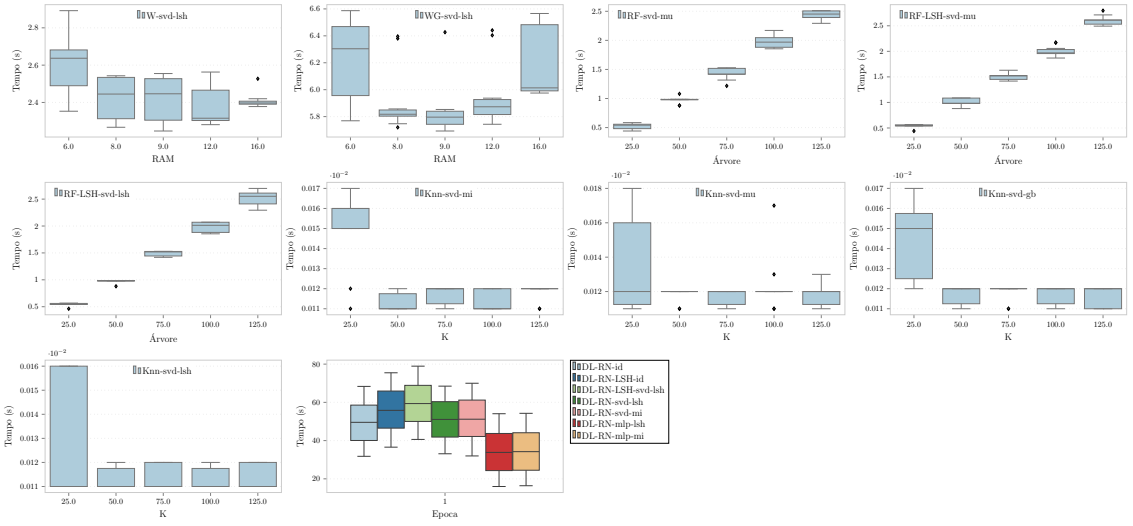


Figura 4.35: Tempo de treino para CCS 300.

Como esperado, o tempo de treinamento dos métodos reduzem quando diminuímos a base, devido ao aumento de itens com *cold-start*. Nota-se também uma variação no tempo devido à escolha dos parâmetros.

As Figuras 4.36 e 4.37, mostram os tempos das abordagens incluindo o tempo de pré-processamento da matriz de similaridade LSH e no caso da WiSARD o tempo de binarização dos dados, para as bases selecionadas em CCS 100 e 300.

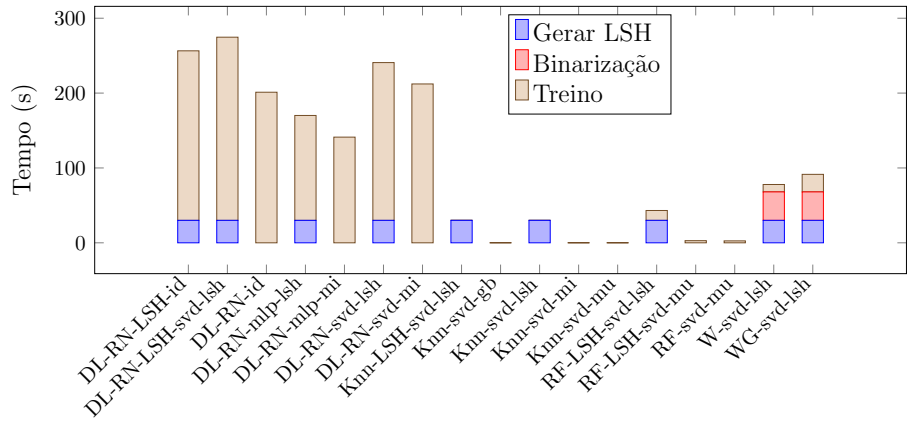


Figura 4.36: Tempo de total considerando pré-processamento para CCS 100.

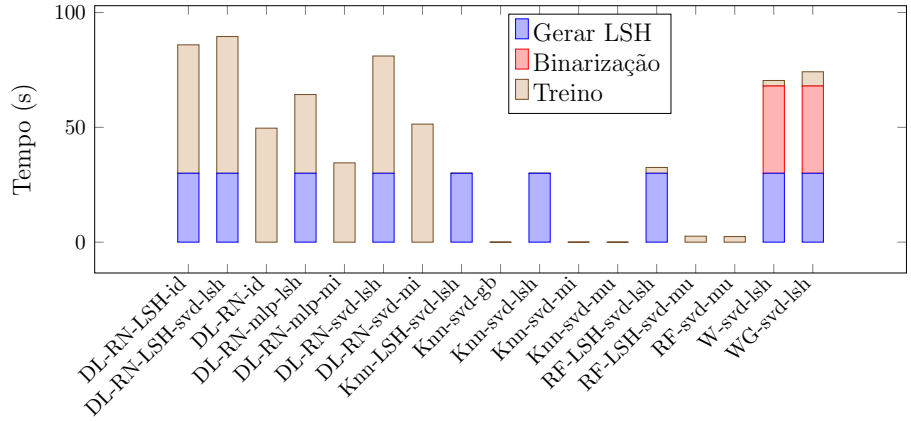


Figura 4.37: Tempo de total considerando pré-processamento para CCS 300.

Percebe-se que os ao utilizar as abordagens com LSH, acarreta um aumento no tempo, como já esperado. Entretanto este aumento é menos percebido quando utilizamos bases maiores. Em relação à binarização dos dados, o método WiSARD é afetado por esse fator. Porém este tempo extra, pode ser reduzido, se esta transformação for realizada de forma paralela. Por fim, o método WiSARD (WG-svd-lsh) foi superior em $89,28\%$ e $87,92\%$ na média para o treinamento com a base CCS 100 e 300 em comparação aos métodos *deep learning* (DL-RN-LSH-id e DL-RN-id), respectivamente.

4.4.10 Análises dos dados

Através dos experimentos anteriores, nota-se uma melhora nos resultados do MAE e RMSE ao utilizar o SVD para selecionar as variáveis latentes nas bases de CCS e ICS. Portanto, nesta seção serão discutidos os resultados apresentados para os experimentos que utilizaram o SVD, evidenciando os principais resultados obtidos. Por fim, serão comparados as técnicas clássicas de sistema de recomendação com as

configurações da proposta. Separamos algumas abordagens para uma análise mais detalhada.

A Tabela 4.21, mostra a porcentagem da redução do erro entre o métodos, separando-os par a par, para cada valor L de CCS de acordo com as métricas MAE e RMSE.

Tabela 4.21: Relação entre os métodos para CCS.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
DL-RN-LSH-id e DL-RN-id	-0,95%	-0,82%	-0,87%	-1,45%	-0,85%	-0,59%
DL-RN-LSH-svd-lsh e DL-RN-id	-0,24%	-0,92%	1,91%	-1,36%	-1,04%	0,46%
RF-LSH-svd-lsh e RF-LSH-svd-mu	-22,99%	-4,45%	-10,47%	-22,36%	-5,78%	-11,93%
Knn-svd-lsh e Knn-svd-mi	-4,28%	-2,94%	0,47%	-4,73%	-5,19%	-2,27%
DL-RN-svd-lsh e DL-RN-svd-mi	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
DL-RN-mlp-lsh e DL-RN-mlp-mi	-0,07%	-0,91%	-5,35%	-0,03%	-0,48%	-5,85%
WG-svd-lsh e DL-RN-id	2,68%	3,79%	2,32%	1,91%	4,52%	2,74%
WG-svd-lsh e Knn-svd-mi	-8,30%	-5,88%	-5,83%	-7,90%	-5,38%	-5,78%

Nota-se que os métodos com a utilização do LSH em sua implementação foram geralmente melhores comparados aos próprios métodos sem a inserção do LSH. Os métodos DL-RN-svd-lsh e DL-RN-svd-mi não houveram variação nos resultados. Já o método Wg-svd-lsh foi superior ao knn que é uma abordagem clássica da literatura, entretanto não conseguiu alcançar os algoritmos de *deep learning*.

A Tabela 4.22, mostra a porcentagem da redução do erro entre o métodos, separando-os par a par, para cada valor L de CCS de acordo com as métricas MAE e RMSE analisando somente os itens que não receberam *ratings* na fase de treinamento.

Tabela 4.22: Relação entre os métodos para itens com 0 *ratings* para CCS.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
DL-RN-LSH-id e DL-RN-id	-10,59%	-6,95%	-6,45%	-9,65%	-6,12%	-5,13%
DL-RN-LSH-svd-lsh e DL-RN-id	-9,35%	-8,52%	-5,91%	-9,17%	-6,33%	-6,15%
RF-LSH-svd-lsh e RF-LSH-svd-mu	-33,87%	7,74%	-23,64%	-33,39%	-1,05%	-27,09%
Knn-svd-lsh e Knn-svd-mi	-2,62%	6,23%	1,56%	-10,84%	-2,17%	0,02%
DL-RN-svd-lsh e DL-RN-svd-mi	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
DL-RN-mlp-lsh e DL-RN-mlp-mi	5,50%	2,40%	1,63%	6,42%	0,95%	-5,74%
WG-svd-lsh e DL-RN-id	-7,32%	-9,26%	-5,57%	-5,10%	-6,86%	-4,54%
WG-svd-lsh e Knn-svd-mi	-9,21%	-6,41%	-5,12%	-7,63%	-7,48%	-6,41%

Nota-se que os métodos com a utilização do LSH acoplado conseguiram melhorar a previsão para os itens em *cold-start*. Já o método Wg-svd-lsh foi superior a todos os métodos analisados neste quesito.

A Tabela 4.23, mostra a porcentagem da redução do erro entre o métodos, separando-os par a par, para cada valor L de CCS de acordo com as métricas MAE e RMSE analisando somente os itens que receberam até 9 *ratings* na fase de treinamento.

Tabela 4.23: Relação entre os métodos para itens com 0 a 9 *ratings* para CCS.

	MAE			RMSE		
	CCS 100	CCS 200	CCS 300	CCS 100	CCS 200	CCS 300
DL-RN-LSH-id e DL-RN-id	-5,39%	-3,54%	-2,55%	-5,10%	-3,27%	-2,06%
DL-RN-LSH-svd-lsh e DL-RN-id	-5,21%	-4,49%	-1,05%	-4,91%	-3,66%	-1,62%
RF-LSH-svd-lsh e RF-LSH-svd-mu	-35,71%	-5,22%	-17,06%	-31,21%	-7,42%	-18,24%
Knn-svd-lsh e Knn-svd-mi	-16,05%	-5,67%	-2,49%	-16,14%	-10,20%	-5,23%
DL-RN-svd-lsh e DL-RN-svd-mi	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
DL-RN-mlp-lsh e DL-RN-mlp-mi	-3,28%	-0,46%	-8,13%	-0,89%	-1,04%	-10,49%
WG-svd-lsh e DL-RN-id	2,77%	1,31%	2,02%	1,56%	1,77%	2,18%
WG-svd-lsh e Knn-svd-mi	-13,54%	-10,00%	-9,70%	-12,45%	-10,07%	-9,59%

Nota-se que a superioridade dos métodos com LSH se mantém na previsão dos itens com poucas avaliações.

Dando continuidade às análises, a tabela 4.24, mostra a porcentagem da redução do erro entre o métodos, separando-os par a par, para cada valor L de ICS de acordo com as métricas MAE e RMSE.

Tabela 4.24: Relação entre os métodos para ICS.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
DL-RN-LSH-id e DL-RN-id	0,05%	-0,93%	0,67%	0,12%	-1,34%	1,38%
DL-RN-LSH-svd-lsh e DL-RN-id	0,93%	-0,73%	0,00%	-0,82%	-2,12%	-0,63%
RF-LSH-svd-lsh e RF-LSH-svd-mu	-10,43%	-11,70%	-21,58%	-11,37%	-13,03%	-21,23%
Knn-svd-lsh e Knn-svd-mi	-0,48%	-0,93%	-3,33%	-1,47%	-0,96%	-3,29%
DL-RN-svd-lsh e DL-RN-svd-mi	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
DL-RN-mlp-lsh e DL-RN-mlp-mi	3,45%	1,03%	1,33%	3,49%	0,65%	1,13%
WG-svd-lsh e DL-RN-id	3,38%	2,47%	3,59%	2,23%	1,33%	3,34%
WG-svd-lsh e Knn-svd-mi	0,17%	-1,23%	-1,89%	-1,39%	-2,14%	-2,63%

Percebe-se uma diminuição de ganhos dos métodos com LSH perante aos tradi-

cionais, principalmente em relação ao MAE para base com poucos itens em ICS.

Agora a Tabela 4.25, mostra a porcentagem da redução do erro entre o métodos, separando-os par a par, para cada valor L de ICS de acordo com as métricas MAE e RMSE analisando somente os itens que não receberam *ratings* na fase de treinamento.

Tabela 4.25: Relação entre os métodos para itens com 0 *ratings* para ICS.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
DL-RN-LSH-id e DL-RN-id	-3,00%	-8,53%	-11,26%	-0,05%	-6,12%	-8,86%
DL-RN-LSH-svd-lsh e DL-RN-id	1,25%	-9,09%	-11,20%	0,21%	-7,81%	-10,08%
RF-LSH-svd-lsh e RF-LSH-svd-mu	-20,44%	-16,36%	-20,17%	-19,80%	-16,27%	-20,99%
Knn-svd-lsh e Knn-svd-mi	-8,79%	-12,88%	-1,10%	-9,98%	-8,20%	-7,96%
DL-RN-svd-lsh e DL-RN-svd-mi	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
DL-RN-mlp-lsh e DL-RN-mlp-mi	9,86%	12,97%	7,13%	10,94%	16,62%	7,62%
WG-svd-lsh e DL-RN-id	0,87%	-8,84%	-7,85%	-0,52%	-7,23%	-5,83%
WG-svd-lsh e Knn-svd-mi	-11,21%	-9,51%	0,00%	-11,77%	-7,37%	-2,46%

Nota-se que os métodos dos quais utilizaram o LSH conseguiram melhorar a previsão para os itens em *cold-start*. Já o método Wg-svd-lsh na maioria dos casos conseguiu se manter como um bom método de previsão para itens com completo *cold-start*.

A Tabela 4.26, mostra a porcentagem da redução do erro entre o métodos, separando-os par a par, para cada valor L de ICS de acordo com as métricas MAE e RMSE analisando somente os itens que receberam até 9 *ratings* na fase de treinamento.

Tabela 4.26: Relação entre os métodos para itens com 0 a 9 *ratings* para ICS.

	MAE			RMSE		
	ICS 100	ICS 200	ICS 300	ICS 100	ICS 200	ICS 300
DL-RN-LSH-id e DL-RN-id	1,41%	-6,74%	-4,17%	1,27%	-6,21%	-3,35%
DL-RN-LSH-svd-lsh e DL-RN-id	2,81%	-8,17%	-5,15%	1,85%	-7,73%	-4,56%
RF-LSH-svd-lsh e RF-LSH-svd-mu	-23,59%	-16,42%	-30,07%	-21,13%	-15,55%	-26,31%
Knn-svd-lsh e Knn-svd-mi	-9,10%	-2,40%	-15,13%	-7,10%	-3,11%	-14,59%
DL-RN-svd-lsh e DL-RN-svd-mi	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
DL-RN-mlp-lsh e DL-RN-mlp-mi	4,88%	5,28%	1,50%	6,58%	6,27%	2,60%
WG-svd-lsh e DL-RN-id	5,87%	1,19%	1,29%	6,27%	-0,06%	1,85%
WG-svd-lsh e Knn-svd-mi	-7,87%	-5,63%	2,14%	-7,83%	-6,78%	-0,90%

Nota-se que a superioridade dos métodos com LSH se mantém na previsão dos itens com poucas avaliações, principalmente nas bases ICS 200 e 300.

De modo geral, a proposta utilizando a matriz de similaridade gerada pelo LSH junto aos métodos clássicos obteve resultados satisfatórios, principalmente para itens com *cold-start*, foco do nosso trabalho.

Os algoritmos baseados em *deep learning*, obtiveram melhores resultados entre todos. Porém sofrem com alto tempo para treinamento. Já a proposta utilizando WiSARD por gêneros obteve um resultado melhor do que o algoritmo dos vizinhos mais próximos. A única desvantagem foi no caso da base ICS 100. O algoritmo proposto classificou pior comparada com o clássico para a avaliação geral. Porém continuou sendo melhor para classificar itens com *cold-start*.

Uma das soluções geradas deste trabalho, é a junção dos algoritmos baseados em *deep learning* com LSH, DL-RN-LSH-id, com a WiSARD por gêneros. Desta forma, podemos prever de forma seletiva, ou seja, itens com 0 avaliações na fase de treinamento seriam previsto pela WiSARD, enquanto os demais pela *deep learning*.

Através desses resultados pode-se concluir que existe uma diminuição considerável tanto no MAE quanto no RMSE utilizando a proposta comparada com os algoritmos clássicos da filtragem colaborativa.

Capítulo 5

Conclusões

Neste capítulo apresentamos as conclusões do trabalho, dissertando brevemente sobre o que foi abordado e como foi elucidado o problema proposto. Em seguida mostramos as contribuições e limitações da proposta juntamente com os trabalhos futuros.

5.1 Considerações sobre o trabalho

Neste trabalho abordamos o problema de *cold-start* em filtragem colaborativa. Um dilema de longa data em sistemas de recomendação. Ocorre quando existe à indisponibilidade de informações adequadas para itens ou usuários disponíveis no sistema. Desta forma, não é possível realizar recomendações relevantes aos usuários. Recentemente métodos de aprendizagem profunda surgiram como ferramentas úteis para representações complexas, sendo possível utilizar mais dados para amenizar este problema. Entretanto o custo computacional é elevado.

Portanto, propomos utilizar uma abordagem híbrida para o problema de *cold-start* itens, através de um algoritmo de baixo custo computacional e de aprendizagem rápida. Para isso, usamos a rede neural sem peso (WiSARD) e o *Locality-Sensitive Hashing*, com o método *MinMaxwise*, para definir as similaridades entre os itens.

A fim de avaliar a nossa proposta, nós fizemos experiências utilizando a base MovieLens 100k e adicionamos informações sobre itens. Selecionamos da base, conjuntos para teste e treinamento, nos quais simulam o completo *cold-start* e incompleto *cold-start*. Foram analisado os parâmetros para a WiSARD imerso ao problema de *cold-start*, já que este método, até o presente trabalho, não tem citações na literatura do seu uso.

Além disso, também incorporamos o *Locality-Sensitive Hashing* nas abordagens tradicionais da literatura e demais métodos de aprendizagem de máquina, gerando bons resultados para esses métodos.

Nossos resultados indicam que ao utilizar a matriz de similaridade gerada pelo LSH acoplado aos métodos tradicionais, melhora o MAE e RMSE para as análises realizadas para as bases CCS 100, 200 e 300 em comparação aos próprios métodos tradicionais sem adição do LSH. Desta forma melhora os métodos de rede neural (DL-RN-ID), k vizinhos mais próximos (knn-mi), floresta aleatória (RF-LSH-svd-mu) e rede neural (DL-RN-MLP-mi) na maioria dos casos.

Foi visto também que a utilização do SVD para selecionar as variáveis latentes conseguiu melhorar as abordagens testadas. Sendo assim, mostraremos os principais resultados a seguir.

A abordagem de rede neural (DL-RN-LSH-id) foi superior em $0,95\%$ e $0,87\%$ no MAE e $1,45\%$ e $0,59\%$ no RMSE para as bases CCS 100 e 300 respectivamente. Já a rede neural (DL-RN-LSH-svd-lsh) foi superior em $0,92\%$ e $1,04\%$ na base CCS 200 para o MAE e RMSE, nesta ordem.

Sobre as análises feitas para itens com 0 *ratings*, a WiSARD por gêneros (WG-svd-lsh) reduziu o MAE em $6,41\%$ e o RMSE $6,11\%$ na base de CCS 200. Já a rede neural (DL-RN-LSH-id) reduziu o MAE em $10,38\%$ e o RMSE em $7,69\%$ para a base CCS 100 e diminuiu em $6,01\%$ o MAE para a base CCS 300. Vale constar que a rede neural (DL-RN-LSH-svd-lsh) obteve o melhor RMSE para a base CCS 300 no qual reduziu em $0,71\%$.

Sobre as análises feitas para os itens com 0 a 9 *ratings*, a rede neural (DL-RN-LSH-id) reduziu o MAE em $5,39\%$ e $2,55\%$ para as bases CCS 100 e 300, respectivamente e diminuiu em $5,1\%$ o RMSE para a base CCS 100. Já a rede neural (DL-RN-LSH-svd-lsh) foi superior em $4,49\%$ para o MAE e $3,66\%$ para o RMSE na base CCS 200. Vale constar que a rede neural (DL-RN-mlp-lsh) obteve o melhor RMSE para a base CCS 300 no qual reduziu em $2,53\%$.

Por fim, o método WiSARD (WG-svd-lsh) foi superior em $89,28\%$ e $87,92\%$ na média para o treinamento com a base CCS 100 e 300 em comparação aos métodos *deep learning* (DL-RN-LSH-id e DL-RN-id), respectivamente.

5.2 Contribuições

Dado o problema de *cold-start* em filtragem colaborativa para sistema de recomendação, nosso trabalho consiste em apresentar uma abordagem que permite a construção de algoritmos flexíveis, com o baixo custo computacional e de aprendizagem rápida. Para tal, foi proposto utilizar a técnica de *Locality-Sensitive Hashing* com o método *MinMaxwise* para processamento de textos com intuído de encontrar as similaridades entre os itens para ser incorporado nas abordagens tradicionais da literatura como rede neurais sem peso e demais métodos de aprendizagem de máquina.

Desta forma, comparamos a abordagem WiSARD, que permite a construção de algoritmos de aprendizagem rápida com baixo custo computacional, com técnicas de *deep learning*, que tem sido utilizadas com sucesso em diferentes tipos de tarefas de classificação.

Além disso, utilizamos a matriz de similaridade dos itens, gerada pelo LSH, para criar uma alternativa para normalização das notas e preenchimento dos dados faltantes da matriz de avaliações.

Com o objetivo de validar a utilização do LSH nas abordagens escolhidas, foram realizados diversos experimentos visando à avaliação do desempenho da técnica com diversas combinações dos parâmetros, principalmente relacionadas às variações de itens com completo *cold-start* e incompleto *cold-start*.

Os experimentos realizados utilizando as técnicas de LSH incorporadas obtiveram resultados satisfatórios e superiores aos algoritmos clássicos. De acordo com as métricas analisadas, MAE e o RMSE, os métodos obtiveram uma redução nas previsões de itens com completo *cold-start*, ou seja, itens que na fase de treinamento continham nenhuma avaliação.

5.3 Limitações e trabalhos futuros

Os experimentos mostram que a utilização da técnica de *Locality-Sensitive Hashing* com o método *MinMaxwise* incorporado aos algoritmos é válida. Entretanto, este processamento extra pode afetar no tempo de execução do algoritmo, tal situação é mais sentida quando a base de treinamento é pequena. Porém no trabalho de DUARTE *et al.* (2017) conseguiram reduzir o tempo de processamento pela metade em relação ao do *MinMaxwise*, tendo em troca uma pequena imprecisão ao se medir o Jaccard. Desta forma, seria possível diminuir o tempo da criação da matriz de similaridade do itens.

Para validar a proposta e simplificar os experimentos, foi utilizado somente uma base para treinar e testar. Mesmo utilizando a separação de dados por tamanhos de itens com completo *cold-start* e incompleto *cold-start* para verificar a performance dos algoritmos propostos em ambientes diferenciados, na literatura existem diversas bases que poderiam ter sido utilizadas. Dependendo das características destas bases os algoritmos propostos podem obter um melhor ou pior resultado. Um trabalho futuro seria avaliar a proposta nessas outras bases de dados.

Outra situação a ser verificada é sobre as mudanças contínuas na distribuição dos dados, principalmente em sistema de recomendação, onde as preferências dos usuários e inserções de itens, ao longo do tempo sofrem essas variações.

Referências Bibliográficas

ADOMAVICIUS, G., TUZHILIN, A., 2005, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions", *IEEE Trans. on Knowl. and Data Eng.*, v. 17, n. 6 (June), pp. 734–749. ISSN: 1041-4347. doi: 10.1109/TKDE.2005.99. Disponível em: <http://dx.doi.org/10.1109/TKDE.2005.99>. Acesso em: 2014-06-15.

AGARWAL, D., CHEN, B.-C., 2009, "Regression-based Latent Factor Models", *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining*, pp. 19–28. ISSN: 1605584959. doi: 10.1145/1557019.1557029. Disponível em: <http://doi.acm.org/10.1145/1557019.1557029>.

ALEKSANDER, I., THOMAS, W., BOWDEN, P., 1984, "WISARD: a radical step forward in image recognition", *Sensor review*, v. 4, n. 3, pp. 120–124.

ALEKSANDER, M. D. G., FRANÇA, F. M. G., LIMA, P. M. V., et al., 2009, "A brief introduction to Weightless Neural Systems", *ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning*, , n. April, pp. 22–24.

ARSAN, T., KOKSAL, E., BOZKUS, Z., 2016, "Comparison of Collaborative Filtering Algorithms with Various Similarity Measures for Movie Recommendation", *International Journal of Computer Science, Engineering and Applications*, v. 6, n. 3, pp. 1–20. ISSN: 22310088. doi: 10.5121/ijcsea.2016.6301. Disponível em: <http://aircconline.com/ijcsea/V6N3/6316ijcsea01.pdf>.

BAGCHI, S., 2015, "Performance and quality assessment of similarity measures in collaborative filtering using mahout", *Procedia Computer Science*, v. 50, pp. 229–234. ISSN: 18770509. doi: 10.1016/j.procs.2015.04.055.

BARBIERI, J., ALVIM, L. G., BRAIDA, F., et al., 2017, "Autoencoders and recommender systems: COFILS approach", *Expert Systems with Applica-*

tions, v. 89, pp. 81–90. ISSN: 09574174. doi: 10.1016/j.eswa.2017.07.030. Disponível em: <http://dx.doi.org/10.1016/j.eswa.2017.07.030>.

BARRÓN-CEDEÑO, A., 2010, "On the Mono- and Cross-language Detection of Text Reuse and Plagiarism", *Procesamiento de Lenguaje Natural*, pp. 914–914. doi: 10.1145/1835449.1835687. Disponível em: <http://doi.acm.org/10.1145/1835449.1835687>.

BENGIO, Y., 2009, "Learning deep architectures for AI", *Foundations and trends® in Machine Learning*, v. 2, n. 1, pp. 1–127.

BENNETT, J., ELKAN, C., LIU, B., et al., 2007, "KDD Cup and Workshop 2007", *SIGKDD Explorations*, v. 9, n. 2, pp. 51–52. doi: 10.1145/1345448.1345459.

BERGER, M., DE SOUZA, A. F., NETO, J. D. O., et al., 2016, "Visual tracking with VG-RAM Weightless Neural Networks", *Neurocomputing*, v. 183, pp. 90–105. ISSN: 18728286. doi: 10.1016/j.neucom.2015.04.127.

BRAIDA, F., MELLO, C. E., PASINATO, M. B., et al., 2015, "Transforming collaborative filtering into supervised learning", *Expert Systems with Applications*, v. 42, n. 10, pp. 4733–4742. ISSN: 09574174. doi: 10.1016/j.eswa.2015.01.023.

BRANDT, P. F. S., 2015, "PREVISAO DE UMIDADE DE SOLOS ATRAVES DE REDES NEURAIS SEM PESO", .

BREESE, J. S., HECKERMAN, D., KADIE, C., 1998, "Empirical analysis of predictive algorithms for collaborative filtering", *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, pp. 43—-52. ISSN: 15532712. doi: 10.1111/j.1553-2712.2011.01172.x.

BRODER, A. Z., 1998, "On the resemblance and containment of documents", *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pp. 21–29. ISSN: 0818681322. doi: 10.1109/SEQUEN.1997.666900. Disponível em: <http://ieeexplore.ieee.org/document/666900/>.

BRODER, A. Z., CHARIKAR, M., FRIEZE, A. M., et al., 2000, "Min-wise independent permutations", *Journal of Computer and System Sciences*, v. 60, n. 3, pp. 630–659.

BURKE, R., 2002, "Hybrid Recommender Systems : Survey and", *User Modeling and UserAdapted Interaction*, v. 12, n. 4, pp. 331–370. ISSN:

09241868. doi: 10.1023/A:1021240730564]. Disponível em: <http://www.springerlink.com/index/N881136032U8K111.pdf>.

CARDOSO, D. O., GAMA, J., FRANÇA, F. M., 2017, "Weightless neural networks for open set recognition", *Machine Learning*, v. 106, n. 9-10, pp. 1547–1567. ISSN: 15730565. doi: 10.1007/s10994-017-5646-4.

CAZELLA, S. C., REATEGUI, E., MACHADO, M., et al., 2009, "Recomendação de Objetos de Aprendizagem Empregando Filtragem Colaborativa e Competências", *Simpósio Brasileiro de Informática na Educação (SBIE)*.

CAZELLA, S. C., DRUMM, J. V., BARBOSA, J. L. V., 2010, "Um serviço para recomendação de artigos científicos baseado em filtragem de conteúdo aplicado a dispositivos móveis", *RENOTE - Novas tecnologias na Educação*, v. 8, n. 3, pp. 1–11. ISSN: 1679-1916. Disponível em: <http://seer.ufrgs.br/index.php/renote/article/view/18057/10645>.

CHARIKAR, M. S., 2002, "Similarity estimation techniques from rounding algorithms", *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, p. 380. ISSN: 07349025. doi: 10.1145/509961.509965. Disponível em: <http://portal.acm.org/citation.cfm?doid=509907.509965>.

CHEN, T., ZHANG, W., LU, Q., et al., 2012, "SVDFeature: a toolkit for feature-based collaborative filtering", *Journal of Machine Learning Research*, v. 13, n. Dec, pp. 3619–3622.

CHIEN, Y. H., GEORGE, E. I., CHEN, Y.-H., et al., 1999, "A bayesian model for collaborative filtering", *Direct*, , n. 1. Disponível em: <http://www-stat.wharton.upenn.edu/{~}edgeorge/Research{_}papers/Bcollab.pdf>.

CHUM, O., PHILBIN, J., ZISSERMAN, A., 2008, "Near Duplicate Image Detection: min-Hash and tf-idf Weighting", *Procedings of the British Machine Vision Conference*, pp. 50.1–50.10. ISSN: 10959203. doi: 10.5244/C.22.50. Disponível em: <http://www.bmva.org/bmvc/2008/papers/119.html>.

CLAYPOOL, M., MIRANDA, T., GOKHALE, A., et al., 1999, "Combining content-based and collaborative filters in an online newspaper", *Proceedings of Recommender Systems Workshop at ACM SIGIR*, pp. 40–48. doi: 10.1.1.46.3659. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.5230{&}rep=rep1{&}type=pdf>.

COVER, T., HART, P., 1967, "Nearest neighbor pattern classification", *Information Theory, IEEE Transactions on*, v. 13, n. 1, pp. 21–27. ISSN: 0018-9448. doi: 10.1109/TIT.1967.1053964.

DE AGUIAR, K., 2014, "Predição de Crises Epilépticas Utilizando Rede Neural Sem Peso WiSARD", .

DE GREGORIO, M., GIORDANO, M., 2017, "Background estimation by weightless neural networks", *Pattern Recognition Letters*, v. 96, pp. 55–65. ISSN: 01678655. doi: 10.1016/j.patrec.2017.05.029.

DECKER, K. M., FOCARDI, S., 1995, "Technology Overview: A Report on Data Mining", *Swiss Scientific Computing Center*. doi: 10.1.1.50.6800. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.50.6800>.

DENG, L., YU, D., 2014, "Deep learning: methods and applications", *Foundations and Trends in Signal Processing*, v. 7, n. 3–4, pp. 197–387.

DEZA, M. M., DEZA, E., 2009, *Encyclopedia of distances*. ISBN: 9783642002335. doi: 10.1007/978-3-642-00234-2.

DUARTE, F., CALED, D., XEXÉO, G., 2017, "Minmax Circular Sector Arc for External Plagiarism's Heuristic Retrieval stage", *Knowledge-Based Systems*. ISSN: 09507051. doi: 10.1016/j.knosys.2017.08.013.

ELAHI, M., RICCI, F., RUBENS, N., 2014, "Active Learning in Collaborative Filtering Recommender Systems", *International Conference on Electronic Commerce and Web Technologies*, pp. 113–124. ISSN: 15740137. doi: 10.1007/978-3-319-10491-1_12.

FERNÁNDEZ-TOBÍAS, I., BRAUNHOFER, M., ELAHI, M., et al., 2016, "Alleviating the new user problem in collaborative filtering by exploiting personality information", *User Modelling and User-Adapted Interaction*, v. 26, n. 2-3, pp. 221–255. ISSN: 15731391. doi: 10.1007/s11257-016-9172-z.

FRANÇA, H. L., DA SILVA, J. C. P., DE GREGORIO, M., et al., 2010, "Movement persuit control of an offshore automated platform via a RAM-based neural network", *11th International Conference on Control, Automation, Robotics and Vision*, pp. 2437–2441. doi: 10.1109/ICARCV.2010.5707913.

FRANCISCO, N., EBECKEN, F., 2013, "Aplicação da rede neural sem peso wisard para o rastreamento de alvos de superfície no mar", *Journal of the Brazilian Computational Intelligence Society*, v. 11, pp. 103–122.

GIONIS, A., INDYK, P., MOTWANI, R., 1999, "Similarity Search in High Dimensions via Hashing", *VLDB '99 Proceedings of the 25th International Conference on Very Large Data Bases*, v. 99, n. 1, pp. 518–529. ISSN: 08941912. doi: 10.1.1.41.4809. Disponível em: <http://www.cs.princeton.edu/courses/archive/spring13/cos598C/Gionis.pdf{%}5Cnhttp://portal.acm.org/citation.cfm?id=671516>.

GOLDBERG, D., NICHOLS, D., OKI, B. M., et al., 1992, "Using Collaborative Filtering to Weave an Information Tapestry", *Commun. ACM*, v. 35, n. 12 (December), pp. 61–70. ISSN: 0001-0782. doi: 10.1145/138859.138867. Disponível em: <http://doi.acm.org/10.1145/138859.138867>. Acesso em: 2014-06-15.

GOPE, J., JAIN, S. K., 2017, "A Survey on Solving Cold-start Problem in Recommender Systems", *International Conference on Computing, Communication and Automation (ICCCA2017)*, pp. 133–138. doi: 10.1109/CCAA.2017.8229786.

GREGORIO, M. D., CIBERNETICA, I., CNR, E. C. I., et al., 2012, "Change Detection with Weightless Neural Networks", , n. 318786, pp. 403–407.

GRIECO, B. P., LIMA, P. M., DE GREGORIO, M., et al., 2010, "Producing pattern examples from "mental" images", *Neurocomputing*, v. 73, n. 7, pp. 1057–1064.

HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J., 2009, "The Elements of Statistical Learning", *Bayesian Forecasting and Dynamic Models*, v. 1, pp. 1–694. ISSN: 0172-7397. doi: 10.1007/b94608. Disponível em: <http://www.springerlink.com/index/10.1007/b94608>.

HAYKIN, S., 2008, *Redes neurais: princípios e prática*. Bookman Editora. ISBN: 9788573077186. doi: 8573077182.

HEINTZE, N., 1996, "Scalable Document Fingerprinting". In: *1996 USENIX Workshop on Electronic Commerce*, pp. 191–200, 1996. ISBN: 1-880446-83-9. doi: 10.1.1.38.8072. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.38.8072>.

HERLOCKER, J. L., KONSTAN, J. A., BORCHERS, A., et al., 1999, "An algorithmic framework for performing collaborative filtering", *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '99*, pp. 230–

237. ISSN: 09540121. doi: 10.1145/312624.312682. Disponível em: <http://portal.acm.org/citation.cfm?doid=312624.312682>.

HERLOCKER, J. L., KONSTAN, J. A., TERVEEN, L. G., et al., 2004, "Evaluating collaborative filtering recommender systems", *ACM Transactions on Information Systems*, v. 22, n. 1, pp. 5–53. ISSN: 10468188. doi: 10.1145/963770.963772. Disponível em: <http://portal.acm.org/citation.cfm?doid=963770.963772>.

HERLOCKER, J. L., 2000, *Understanding and Improving Automated Collaborative Filtering Systems. 2000. 220 f.* Tese de Doutorado, Tese (Doutorado em Ciência da Computação)-University of Minnesota, Minnesota.

HILL, W., STEAD, L., ROSENSTEIN, M., et al., 1995, "Recommending and evaluating choices in a virtual community of use". In: *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95*, pp. 194–201. ISBN: 0201847051. doi: 10.1145/223904.223929. Disponível em: <http://portal.acm.org/citation.cfm?doid=223904.223929>.

HINTON, G. E., SALAKHUTDINOV, R. R., 2006, "Reducing the dimensionality of data with neural networks", *Science*, v. 313, n. 5786, pp. 504–507.

HINTON, G. E., OSINDERO, S., TEH, Y.-W., 2006, "A fast learning algorithm for deep belief nets", *Neural computation*, v. 18, n. 7, pp. 1527–1554.

HOAD, T. C., ZOBEL, J., 2003, "Methods for identifying versioned and plagiarized documents", *Journal of the Association for Information Science and Technology*, v. 54, n. 3, pp. 203–215.

HU, L., CAO, J., XU, G., et al., 2013, "Personalized recommendation via cross-domain triadic factorization". In: *Proceedings of the 22nd international conference on World Wide Web - WWW '13*, pp. 595–606, 2013. ISBN: 9781450320351. doi: 10.1145/2488388.2488441. Disponível em: <http://dl.acm.org/citation.cfm?doid=2488388.2488441>.

INDYK, P., MOTWD, R., 1998, "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality", *Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM*, pp. 604–613. ISSN: 00123692. doi: 10.4086/toc.2012.v008a014.

IOFFE, S., SZEGEDY, C., 2015, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *arXiv preprint arXiv:1502.03167*.

JACCARD, P., 1901, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura", *Bulletin del la Société Vaudoise des Sciences Naturelles*, v. 37, n. JANUARY 1901, pp. 547–579. doi: http://dx.doi.org/10.5169/seals-266450.

JI, J., LI, J., YAN, S., et al., 2013, "Min-max hash for jaccard similarity". In: *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 301–309, 2013. ISBN: 978-0-7695-5108-1. doi: 10.1109/ICDM.2013.119.

KHAN, B. M., MANSHA, A., KHAN, F. H., et al., 2017, "Collaborative filtering based online recommendation systems: A survey", *2017 International Conference on Information and Communication Technologies (ICICT)*, pp. 125–130. doi: 10.1109/ICICT.2017.8320176.

KIM, B. M., LI, Q., PARK, C. S., et al., 2006, "A new approach for combining content-based and collaborative filters", *Journal of Intelligent Information Systems*, v. 27, n. 1, pp. 79–91. ISSN: 09259902. doi: 10.1007/s10844-006-8771-2.

KOREN, Y., BELL, R., VOLINSKY, C., 2009, "Matrix factorization techniques for recommender systems", *Computer*, v. 42, n. 8, pp. 30–37. ISSN: 00189162. doi: 10.1109/MC.2009.263.

KULIS, B., GRAUMAN, K., 2012, "Kernelized locality-sensitive hashing", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 34, n. 6, pp. 1092–1104. ISSN: 01628828. doi: 10.1109/TPAMI.2011.219.

LE, Q. V., RANZATO, M., MONGA, R., et al., 2013, "Building high-level features using large scale unsupervised learning", *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8595–8598. ISSN: 1520-6149. doi: 10.1109/ICASSP.2013.6639343. Disponível em: <`http://ieeexplore.ieee.org/document/6639343/`>.

LEE, H., PHAM, P., LARGMAN, Y., et al., 2009, "Unsupervised feature learning for audio classification using convolutional deep belief networks." *Nips*, pp. 1–9. ISSN: 02643294. doi: 10.1145/1553374.1553453.

LESKOVEC, J., RAJARAMAN, A., ULLMAN, J. D., 2014, *Mining of massive datasets: Second edition*. Cambridge university press. ISBN: 9781139924801. doi: 10.1017/CBO9781139924801.

LI, S., KAWALE, J., FU, Y., 2015, "Deep Collaborative Filtering via Marginalized Denoising Auto-encoder". In: *Proceedings of the 24th ACM*

International on Conference on Information and Knowledge Management - CIKM '15, pp. 811–820. ACM, 2015. ISBN: 9781450337946. doi: 10.1145/2806416.2806527. Disponível em: <http://dl.acm.org/citation.cfm?doid=2806416.2806527>.

LINDEN, G., SMITH, B., YORK, J., 2003, "Amazon. com recommendations: Item-to-item collaborative filtering", *Internet Computing, IEEE*, v. 7, n. 1, pp. 76–80. ISSN: 10897801. doi: 10.1109/MIC.2003.1167344. Disponível em: <http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=1167344>.

MANBER, U., 1994, "Finding Similar Files in a Large File System". In: *Proceedings of the USENIX Winter Technical Conference*, pp. 1–10, 1994. ISBN: 1880446588. doi: 10.1.1.12.3222. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.12.3222{&}rep=rep1{&}type=pdf>.

MCCULLOCH, W. S., PITTS, W., 1943, "A logical calculus of the ideas immanent in nervous activity", *The Bulletin of Mathematical Biophysics*, v. 5, n. 4, pp. 115–133. ISSN: 00074985. doi: 10.1007/BF02478259.

MITCHELL, T. M., 1997, *Machine Learning*. N. 1. ISBN: 0070428077. doi: 10.1145/242224.242229. Disponível em: <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20{&}path=ASIN/0070428077>.

O'CONNOR, M., HERLOCKER, J., 2001, "Clustering items for collaborative filtering", *the Proceedings of SIGIR-2001 Workshop on*.

OLIVEIRA, J. C. M., PONTES, K. V., SARTORI, I., et al., 2017, "Fault Detection and Diagnosis in dynamic systems using Weightless Neural Networks", *Expert Systems with Applications*, v. 84, pp. 200–219. ISSN: 09574174. doi: 10.1016/j.eswa.2017.05.020.

PARK, S.-T., CHU, W., 2009, "Pairwise preference regression for cold-start recommendation". In: *Proceedings of the third ACM conference on Recommender systems - RecSys '09*, p. 21, 2009. ISBN: 9781605584355. doi: 10.1145/1639714.1639720. Disponível em: <http://portal.acm.org/citation.cfm?doid=1639714.1639720>.

PATEREK, A., 2007, "Improving regularized singular value decomposition for collaborative filtering", *KDD Cup and Workshop*, pp. 2–5. ISSN: 00121606. doi: 10.1145/1557019.1557072.

PAZZANI, M. J., 1999, "A Framework for Collaborative, Content-Based and Demographic Filtering", *Artificial Intelligence Review*, v. 13, n. 5-6 (December), pp. 393–408. ISSN: 0269-2821. doi: 10.1023/A:1006544522159. Disponível em: <http://dx.doi.org/10.1023/A:1006544522159>. Acesso em: 2014-07-03.

PAZZANI, M. J., BILLSUS, D., 2007, "Content-based recommendation systems". In: *The adaptive web*, Springer, pp. 325–341.

PENNOCK, D. M., LAWRENCE, S., GILES, C. L., 2000, "Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-Based Approach", *Uncertainty in Artificial Intelligence Proceedings*, pp. 473–480. ISSN: 00992240.

REATEGUI, E., CAZELLA, S. C., 2005, "Sistemas de Recomendação", *XXV Congresso da Sociedade Brasileira de Computação A Universalidade da Computação Um Agente de Inovação e Conhecimento*, pp. 306–348. Disponível em: <http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Sistemas+de+Recomenda??o{#}4>.

RESNICK, P., IACOVOU, N., SUCHAK, M., et al., 1994, "GroupLens : An Open Architecture for Collaborative Filtering of Netnews", *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pp. 175–186. ISSN: 00027863. doi: 10.1145/192844.192905.

RICCI, F., ROKACH, L., SHAPIRA, B., 2015, *Introduction to Recommender Systems Handbook*, v. 54. New York, Springer. ISBN: 978-0-387-85819-7. doi: 10.1007/978-0-387-85820-3. Disponível em: <http://www.springerlink.com/index/10.1007/978-0-387-85820-3{%}5Cnhttp://link.springer.com/10.1007/978-0-387-85820-3{_}1>.

ROSENBLATT, F., 1958, "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, v. 65, n. 6, pp. 386–408. ISSN: 0033295X. doi: 10.1037/h0042519.

RUSSELL, S. J., NORVIG, P., 1995, *Artificial Intelligence: A Modern Approach*, v. 9. ISBN: 9780131038059. doi: 10.1016/0925-2312(95)90020-9. Disponível em: <http://portal.acm.org/citation.cfm?id=773294>.

SALAKHUTDINOV, R., HINTON, G., 2008, "Using Deep Belief Nets to Learn Covariance Kernels for Gaussian Processes", *Advances in Neural Information Processing Systems 20*, v. 20, pp. 1–8. Disponível em: <http://web.mit.edu/{~}rsalakhu/www/papers/nips07{_}gp.pdf>.

SALAKHUTDINOV, R., HINTON, G., 2009, "Semantic hashing", *International Journal of Approximate Reasoning*, v. 50, n. 7, pp. 969–978.

SALAKHUTDINOV, R., MNIH, A., HINTON, G., 2007, "Restricted Boltzmann machines for collaborative filtering". In: *Proceedings of the 24th international conference on Machine learning - ICML '07*, pp. 791–798, 2007. ISBN: 9781595937933. doi: 10.1145/1273496.1273596. Disponível em: <http://portal.acm.org/citation.cfm?doid=1273496.1273596>.

SARWAR, B., KARYPIS, G., KONSTAN, J., et al., 2001, "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the tenth international conference on World Wide Web - WWW '01*, pp. 285–295. ISBN: 1581133480. doi: 10.1145/371920.372071. Disponível em: <http://portal.acm.org/citation.cfm?doid=371920.372071>.

SCHAFER, J. B., KONSTAN, J. A., RIEDL, J., 2001, "E-commerce recommendation applications". In: *Applications of Data Mining to Electronic Commerce*, Springer, pp. 115–153.

SCHWARTZ, B., 2005, *The paradox of choice*. New York, ECCO. ISBN: 0060005688 9780060005689.

SEDHAIN, S., MENON, A. K., SANNER, S., et al., 2015a, "AutoRec : Autoencoders Meet Collaborative Filtering", *WWW 2015 Companion: Proceedings of the 24th International Conference on World Wide Web*, pp. 111–112. doi: 10.1145/2740908.2742726.

SEDHAIN, S., MENON, A. K., SANNER, S., et al., 2015b, "AutoRec : Autoencoders Meet Collaborative Filtering", *WWW 2015 Companion: Proceedings of the 24th International Conference on World Wide Web*, pp. 111–112. doi: 10.1145/2740908.2742726.

SHANI, G., HECKERMAN, D., BRAFMAN, R. I., 2005, "An MDP-based recommender system", *Journal of Machine Learning Research*, v. 6, n. Sep, pp. 1265–1295.

SHARDANAND, U., MAES, P., 1995, "Social Information Filtering: Algorithms for Automating &Ldquo;Word of Mouth&Rdquo;", *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95*, pp. 210–217. ISSN: 10414347. doi: 10.1145/223904.223931. Disponível em: <http://dx.doi.org/10.1145/223904.223931>. Acesso em: 2014-06-15.

SLANEY, M., CASEY, M., 2008, "Locality-sensitive hashing for finding nearest neighbors [lecture notes]", *IEEE Signal processing magazine*, v. 25, n. 2, pp. 128–131.

SOBOROFF, I., NICHOLAS, C., 1999, "Combining content and collaboration in text filtering", *International Joint Conferences on Artificial Intelligence*, v. 99, pp. 86—-91. ISSN: 03640213. doi: 10.3115/1118935.1118938. Disponível em: <http://www.csee.umbc.edu/csee/research/cadip/1999Symposium/mlif.pdf>.

STEIN, B., ZU EISSEN, S. M., 2006, "Near similarity search and plagiarism analysis". In: *From data and information analysis to knowledge engineering*, Springer, pp. 430–437.

STRUB, F., MARY, J., 2015, "Collaborative Filtering with Stacked Denoising AutoEncoders and Sparse Inputs", *Nipsw*, pp. 1–8.

STRUB, F., GAUDEL, R., MARY, J., 2016, "Hybrid Recommender System based on Autoencoders", doi: 10.1145/2988450.2988456. Disponível em: <http://arxiv.org/abs/1606.07659{%}0Ahttp://dx.doi.org/10.1145/2988450.2988456>.

TARLING, R., ROHWER, R., 1993, "Efficient use of training data in the n-tuple recognition method", *Electronics letters*, v. 29, n. 24, pp. 2093–2094.

TURING, A. M., 1950, "Computing machinery and intelligence", *Mind*, v. 59, n. 236, pp. 433–460. ISSN: 0026-4423. doi: http://dx.doi.org/10.1007/978-1-4020-6710-5_3.

UNGAR, L. H., FOSTER, D. P., 1998, "Clustering methods for collaborative filtering", *AAAI Workshop on Recommendation Systems*, pp. 114–129. ISSN: 10897798. doi: 10.1.1.33.4026. Disponível em: <http://www.aaai.org/Papers/Workshops/1998/WS-98-08/WS98-08-029.pdf>.

VINCENT, P., LAROCHELLE, H., BENGIO, Y., et al., 2008, "Extracting and composing robust features with denoising autoencoders". In: *Proceedings of the 25th international conference on Machine learning - ICML '08*, pp. 1096–1103, 2008. ISBN: 9781605582054. doi: 10.1145/1390156.1390294. Disponível em: <http://portal.acm.org/citation.cfm?doid=1390156.1390294>.

VISHNUBHOTLA, S., FERNANDEZ, R., RAMABHADRAN, B., 2010, "An auto-encoder neural-network based low-dimensionality approach to excitation

modeling for HMM-based text-to-speech". In: *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 4614–4617. IEEE, March. doi: 10.1109/ICASSP.2010.5495546.

WANG, H., WANG, N., YEUNG, D.-Y., 2014, "Collaborative Deep Learning for Recommender Systems", *arXiv preprint arXiv:1409.2944*.

WEI, J., HE, J., CHEN, K., et al., 2017, "Collaborative filtering and deep learning based recommendation system for cold start items", *Expert Systems with Applications*, v. 69, pp. 1339–1351. ISSN: 09574174. doi: 10.1016/j.eswa.2016.09.040.

XUE, G.-R., LIN, C., YANG, Q., et al., 2005, "Scalable collaborative filtering using cluster-based smoothing". In: *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 114–121. ACM.

ZHANG, Z., JIN, X., LI, L., et al., 2016, "Multi-Domain Active Learning for Recommendation", *[AAAI2016]Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2358–2364.

ZHU, X., GOLDBERG, A. B., 2009, "Introduction to Semi-Supervised Learning", *Synthesis Lectures on Artificial Intelligence and Machine Learning*, v. 3, n. 1, pp. 1–130. ISSN: 1939-4608. doi: 10.2200/S00196ED1V01Y200906AIM006. Disponível em: <http://www.morganclaypool.com/doi/abs/10.2200/S00196ED1V01Y200906AIM006>.