**Trabalhos de Conclusão de Curso**

28 de novembro, São Paulo-SP, Brasil

**PROPOSTA DE ALGORITMO E DESENVOLVIMENTO DE BIBLIOTECA PARA SISTEMAS DE RECOMENDAÇÃO DE PRODUTOS DE LOJAS DE COMÉRCIO ONLINE**

|  |  |
| --- | --- |
| Antonio Guilherme Ferreira Viggiano  agfviggiano@gmail.com | Fernando Fochi Silveira de Araújo  fernando.fochi@gmail.com |
| Orientador: Prof. Dr. Fábio Gagliardi Cozman  fgcozman@usp.br | |

### Resumo

*O objetivo deste trabalho é projetar e avaliar o desempenho de uma biblioteca computacional para sistemas de recomendação de produtos de lojas de comércio online. Essa biblioteca tem finalidade de permitir a fácil implementação de um sistema de recomendação genérico para ser utilizado por acadêmicos e e-commerces que desejem automatizar o processo de sugestão de itens, tal como em email marketing. A biblioteca foi desenvolvida utilizando-se três diferentes algoritmos de recomendação. O algoritmo baseado na ponderação de atributos, que trata-se de um método híbrido entre filtragem colaborativa e filtragem baseada em conteúdo, onde a partir da regressão linear de dados de uma rede social, extrai os pesos que determinam a importância de cada atributo dos itens. O segundo método, baseado em perfil de usuários, leva em consideração o interesse dos usuários por features, indiretamente calculado a partir de seu interesse pelos*

*itens. O terceiro método, baseado na correlação usuário-item, é uma variante do método baseado no perfil de usuários. Este método busca os itens com features mais similares aos atributos pelos quais o usuário se interessa.*

*A avaliação comparativa dos métodos mostrou a superioridade do algoritmo de perfil de usuários em quase todos os aspectos, e avaliou os principais parâmetros de influência na qualidade da recomendação. A partir dos resultados empíricos mostrados neste trabalho, é possível estabelecer diretrizes para a elaboração de um sistema de recomendação próprio com base na biblioteca elaborada pela dupla.*

**Palavras chave:** *Inteligência artificial, Aprendizado computacional, Comercio eletrônico, Produtos.*

**1. Introdução**

O comércio on-line se torna cada vez mais importante na vida das pessoas, de forma que a adoção deste método de compra é cada vez mais comum. Estima-se que em 2013 um bilhão de pessoas compraram online (1), gerando uma receita anual de 1,25 trilhão de dólares com expectativas de crescer 17% ao ano até 2017 (2). Como o mercado de varejo online é novo como um todo, este ainda passará por algumas mudanças drásticas em um curto espaço de tempo. Um dos itens-chave destas mudança é a capacidade de se analisar os dados gerados pelos consumidores. Com estes dados será possível segmentar os clientes mais facilmente, dando possibilidade às empresas de direcionar suas estratégias de forma mais eficiente, até o ponto em que campanhas de marketing e precificação serão totalmente personalizadas (3). Uma das maneiras de se usar estes dados é através dos sistemas de recomendação. Sistemas de recomendação são “ferramentas e técnicas de software destinadas a prover sugestões de itens para usuários” (4). Estes sistemas são utilizados por diversos serviços online e geram um grande impacto quando utilizados corretamente. Em 2012, cerca de 75% dos vídeos assistidos através do site NetFlix foram acessados por meio de recomendações (5). Em 2006, as recomendações representaram 35% dos livros vendidos pela Amazon (6), enquanto em 2007 cerca de 38% das notícias lidas no Google News foram sugeridas por um sistema de recomendação (7). Para atingir uma alta qualidade de sugestões, é imprescindível que o sistema seja altamente personalizado. Essa idéia se resume ao comentário do diretor executivo do e-commerce Amazon.com: “se você possui 2 milhões de clientes na web, você precisa ter 2 milhões de lojas na web”.

Motivados pela importância econômica dos e-commerces, bem como pela possibilidade de criar um conjunto de ferramentas open source que possam ser utilizadas pela comunidade acadêmica e empresarial, propomos como Trabalho de Conclusão de Curso o desenvolvimento de uma biblioteca computacional para sistemas de recomendação de produtos de lojas de comércio online. Esse pacote computacional é composto de métodos de leitura de dados de histórico de compras e de informações de clientes e produtos, de cálculo de sugestões de itens com base em algoritmos de recomendação e de análise de desempenho das recomendações. A motivação de se criar uma biblioteca de software decorre principalmente da sua abrangência e capacidade de adaptação, visto que é possível atender a mais casos de uso que um sistema de recomendação completo. De um lado, um sistema de recomendação possui uma finalidade específica – como por exemplo de sugerir notícias para usuários de internet – e uma entrada e saída de dados específica – como por exemplo o fato de as notícias sempre estarem ordenada pelas mais recentes em uma tabela de sugestões. De outro lado, uma biblioteca computacional pode receber qualquer tipo de dados e gerar qualquer saída de dados. Caso uma empresa ou um acadêmico queira construir seu próprio sistema de

recomendação, basta elaborar a conexão entre o pacote apresentado pela dupla, seu banco de dados e a interface gráfica de apresentação de resultados. As contribuições científica e tecnológica deste trabalho para a Engenharia Mecatrônica estão sobretudo nos campos de inteligência artificial, de sistemas de informação e de automação de processos.

**2. Metodologia**

Por se tratar primordialmente de um projeto de Engenharia de Software, a metodologia de projeto deste trabalho é fundamentalmente iterativa. A primeira etapa, de definição de necessidade, decorreu da constatação de que o crescente número de lojas de comércio online impulsiona a criação de sistemas de recomendação gratuitos, de fácil integração e de código aberto. Em seguida, quanto à definição dos parâmetros de sucesso, estabelecemos que deveria ser feita uma veriﬁcação entre as sugestões do sistema e os itens comprados pelos usuários. Posteriormente, foram propostas diversas soluções para o problema. Foram escolhidos dois métodos-base, provenientes de artigos de referência, e um terceiro método proposto pela dupla. Após o primeiro semestre de trabalho, os algoritmos foram implementados e testados em bancos de dados simples, com menos de dez usuários e ítens. Após essa primeira etapa de projeto, passamos à fase de aplicação das soluções em bancos de dados reais. O primeiro conjunto de dados abertos é proveniente do sistema de recomendações de filmes MovieLens (<http://movielens.umn.edu>), e é composto de 100 000 avaliações (valores inteiros de 1 a 5) de 943 usuários para 1682 filmes. Além disso, cada usuário avaliou pelo menos 20 filmes (categoria, ano de publicação). Nessa base de dados, chamada de 100k, o catálogo de filme faz o papel de catálogo de produtos, e o histórico de compras se refere à avaliação dos filmes feita por cada usuário. O segundo banco de dados é extraído do Internet Movie Database (IMDB), e possui 28 819 filmes. Esse banco está presente na biblioteca *ggplot2* da linguagem de programação R. Na nossa análise, os bancos de dados 100k e IMDB foram utilizados complementarmente. A união desses dois conjuntos deu origem à base 100k-IMDB, composta por 943 usuários, 1682 itens e 25 atributos. A fim de realizar um estudo comparativo (*benchmarking*) com os artigos de referência, mantivemos a mesma metodologia de avaliação de qualidade dos artigos-base. Em particular, implementamos uma validação cruzada considerando T = 75% do banco de dados como base de treinamento ou aprendizado e os 25% restantes como base de testes. Em seguida, mascaramos H = 75% das avaliações dos usuários-teste, de modo a medir a qualidade do sistema de recomendação em prever os itens positivamente avaliados. Cerca de uma dezena de parâmetros de interesse foram avaliados para cada um dos métodos (Tabela 2). Como a complexidade dos algoritmos excede os limites dos computadores pessoais da dupla, foi necessário contratar o serviço de computação nas nuvens Amazon Web Services. Alugamos duas máquinas virtuais do tipo r3.large, otimizadas para memória. As máquinas, de especificação 2 vCPU, 15 GB de memória RAM e sistema operacional Amazon Linux AMI release 2014.09 x86\_64, baseado em RHEL Fedora, custaram USD 0,175 por hora de uso. Todos os testes foram realizados em aproximadamente 12 horas, custando apenas USD 4,20 (menos de R$ 12,00).

**3. Requisitos do Projeto**

Os requisitos funcionais de um sistema de recomendação ditam principalmente sobre a escalabilidade e o desempenho das recomendações do sistema (8). Como as sugestões serão calculadas de maneira *offline*, não há necessidade para uma elevada taxa de recomendações por período de tempo (*throughput*). A ﬁm de poder estabelecer uma base comparativa entre o sistema proposto pela dupla UI e os sistemas de referência FW e UP, serão utilizados os mesmos indicadores de desempenho dos artigos-base: precisão, abrangência e medida F1 (9, 10). Precisão é a porcentagem de casos corretamente preditos em relação ao tamanho da lista de recomendações. Abrangência é a razão entre o número de itens corretamente preditos e aqueles que foram efetivamente avaliados pelo usuário. A medida F1, por sua vez, é a média harmônica entre precisão e abrangência. Todas essas métricas são dependentes dos diversos parâmetros do problema, como do tamanho da lista de recomendações N, da quantidade de vizinhos mais próximos k, e principalmente do banco de dados utilizado. Como os artigos de referência não o disponibilizaram integralmente, os valores de precisão, abrangência e medida F1 serão estimados para o banco de dados da dupla. Espera-se que a precisão, abrangência e consequentemente a medida F1 sejam maiores que 20%. Esses valores foram escolhidos por serem superiores aos de algoritmos puramente baseados em conteúdo ou em ﬁltragem colaborativa (9, 10). Na prática, o resultado mais importante é a comparação entre os três métodos para um mesmo banco de dados de referência.

Tabela 1: Requisitos para o sistema de recomendação

|  |  |
| --- | --- |
| **Requisito** | **Valor** |
| Precisão | 20 % |
| Abrangência | 20 % |

**4. Síntese de Soluções**

**4.1. Algoritmo baseado na ponderação de atributos (FW)**

O primeiro algoritmo que utilizaremos no sistema de recomendação, adaptado de (9) e denominado ponderação de atributos, *feature weighting* ou FW, trata-se de um híbrido entre ﬁltragem colaborativa e ﬁltragem baseada em conteúdo. A partir da regressão linear de dados de uma rede social (*Internet Movie Database, IMDB*), extraem-se os pesos que determinam a importância de cada atributo dos itens, onde ocorre a ﬁltragem colaborativa dos usuários. Após obtenção dos pesos, realiza-se a ﬁltragem baseada em conteúdo para determinar os itens com maior similaridade, que são ﬁnalmente recomendados. Na ﬁltragem baseada em conteúdo, “cada item é representado por um vetor de

atributos ou features”. A similaridade sij entre dois itens i e j é dada pela média ponderada das distâncias entre as *features* dos itens:

sij = ∑f wf (1 − df ij)

As distâncias entre os atributos df são determinadas conforme o tipo de dado avaliado e seu domínio, normalizadas no intervalo [0, 1]. Os pesos wf são a priori desconhecidos. A Referência (9) os determina a partir de uma regressão linear, onde se determina o número de usuários que se interessam tanto por i quanto por j. Esses valores permitem determinar “o julgamento humano de similaridade entre itens”, e pode ser calculado a partir da matriz de avaliações. Desta forma, os pesos wf são determinados e calcula-se a matriz de similaridade **S**. Recomendam-se, por fim, os itens similares àqueles já comprados.

**4.2. Algoritmo baseado no perﬁl de usuários (UP)**

O segundo algoritmo, adaptado de (10), é um hibrido entre ﬁltragem colaborativa e ﬁltragem baseada em conteúdo. Os atributos dos itens são ponderados no cálculo de similaridade, com pesos extraídos de um modelo de perﬁl de usuários, denominado *user proﬁle* ou UP. Esse perﬁl leva em consideração o interesse dos usuários por features, indiretamente calculado a partir de seu interesse pelos itens. Para se determinar a relevância de f para u, deve-se levar em conta não somente a frequência com a qual uma característica aparece, mas também o fato de algumas características estarem contidas na maioria dos itens. Determina-se, então, os pesos wuf, que mostram a relevância de f para u, a partir da medida estatística TF-IDF (*term frequency–inverse document frequency*), presente em formulações de recuperação de informação e mineração de dados.

wuf = TFuf IDFf

Os pesos wuf, obtidos na TF-IDF, são utilizados para calcular a similaridade suv entre dois usuários u e v. Dispondo-se de **S**, selecionam-se os k vizinhos mais próximos vku com maior similaridade suv, dentre todos v ≠ u. Posteriormente, determina-se o conjunto Ivku de itens i avaliados positivamente por vku. Avalia-se a frequência total fuf dos atributos f para os itens de Ivku. Por ﬁm, calcula-se o peso ωui de cada item, a partir da matriz de avaliações aif e de frequências fuf, e gera-se a lista dos top-N produtos a serem recomendados para o usuário u.

ωui = ∑f aif fuf

**4.3. Algoritmo baseado na correlação usuário-item (UI)**

Este método é uma variante da solução UP, e também está embasado no cálculo da preferência do usuário por features, medida através do seu interesse pelos itens. O algoritmo UI utiliza as matrizes de correlação ponderada entre usuários e atributos **W** e a matriz de atributos dos itens **A** no cálculo da correlação usuário-item. A lista dos N produtos a serem recomendados decorre portanto do cálculo de ωui e da escolha dos itens que maximizem essa variável para cada usuário.

ωui = ∑f wuf aif

Ao passo que o método UP recomenda itens a partir dos k vizinhos mais próximos, o algoritmo UI busca os itens com features mais similares aos atributos pelos quais u se interessa, diretamente através da matriz de atributos.

Em princípio, esse tipo de recomendação deve fornecer sugestões de qualidade similar ao algoritmo original, pois os dois tem a mesma fundamentação teórica.

**5. Desenvolvimento da biblioteca**

**5.1 Ferramentas utilizadas**A programação da biblioteca computacional se deu por meio do ambiente de desenvolvimento integrado RStudio versão 0.98.953 (<http://www.rstudio.com/>). Esse IDE inclui console, editor de texto e corretor de sintaxe que suporta a execução direta de código, bem como ferramentas para visualização de gráficos, depuração de erros e gerenciamento de espaço de trabalho. Além disso, o RStudio está disponível via licença de código aberto AGPLv3 (Affero General Public License version 3) para os principais sistemas operacionais (Windows, Mac e Linux).

**5.2 Estrutura da biblioteca**A biblioteca está estruturada em quatro seções principais: *db*, onde está o banco de dados MovieLens 100k, *methods*, onde estão os algoritmos de recomendação, *results*, onde estão os métodos de avaliação de qualidade e *setup*, onde estão codificadas funções diversas, tais como leitura de banco de dados e cálculo de medidas de distância. As principais funções da biblioteca estão descritas na documentação do projeto, juntamente com o código, que pode ser obtido através do endereço <https://github.com/aviggiano/tcc/tree/master/recsys>.

**5.3 Ambiente de testes**Inicialmente, realizamos os testes de qualidade nos nossos próprios computadores pessoais. Todavia, a execução de testes sucessivos exigia muita capacidade computacional, principalmente quanto a memória virtual. A alocação de objetos e matrizes na memória RAM é muito custosa, principalmente na etapa de determinação de medidas de distância df ij para o algoritmo FW. Por essa razão, realizamos todas as etapas de recomendação e avaliação de qualidade em máquinas memory-optimized nos servidores da Amazon Web Services (<http://aws.amazon.com/>). Visto que o serviço é cobrado por hora-máquina, desenvolvemos um script de inicialização para instalar todos os pacotes de programação e execução imediata dos testes, permitindo assim reduzir os custos da análise. As etapas de configuração do ambiente de testes envolvem o cadastro na Amazon Web Services, a criação de uma máquina virtual, a instalação das ferramentas de programação e o descarregamento do código de testes.

**6. Resultados**

Além da biblioteca computacional para sistemas de recomendação, este trabalho tem como resultado as análises de desempenho dos algoritmos propostos, em termos de precisão, abrangência e tempo computacional, mediante a mudanças em suas variáveis de importância (Tabela 2).

Tabela 2: – Parâmetros de influência no desempenho dos algoritmos de recomendação

|  |  |  |
| --- | --- | --- |
| **Variável** | **Descrição** | **Valor padrão** |
| N | Tamanho da lista de recomendação | 20 |
| T | Percentual da base de aprendizado na  validação cruzada | 75% |
| H | Percentual de avaliações “escondidas”  dos usuários-teste na validação cruzada | 75% |
| M | Valor mínimo para avaliações positivas | 2 |
| k | Número de vizinhos mais próximos | 10 |
| F | Conjunto de atributos dos itens | Todos atributos |
| df | Medida de distância entre atributos | Distância L1 |
| W | Quantidade de pesos dos atributos | Todo wf > 0 |

**6.1. Tamanho da lista de recomendações**

A medida que o tamanho da lista de recomendações aumenta, a precisão cai e a abrangência cresce (Figuras 1 e 2). A primeira decresce com N porque a quantidade de itens sugeridos se torna excessivamente maior que a quantidade de itens positivamente avaliados pelos usuários-teste. A segunda, por sua vez, cresce com N porque a probabilidade de sugerirmos itens relevantes para o usuário aumenta quando sugerimos mais itens. Para igual ao total de itens , a abrangência atinge 100%, pois todos os itens teriam sido recomendados.

O método UP supera os outros algoritmos para todos os valores de , tanto em precisão quanto em abrangência.

Figura 1 - Precisão por tamanho de N

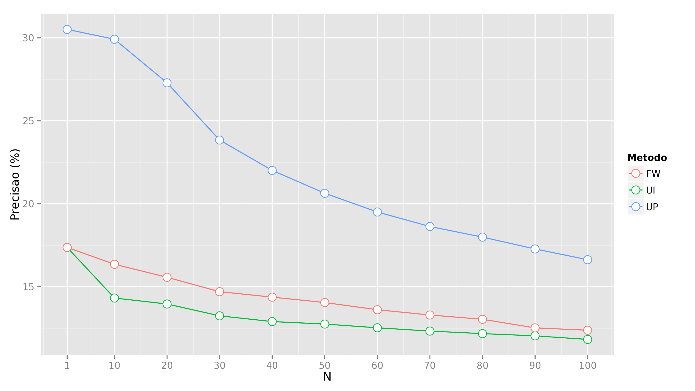
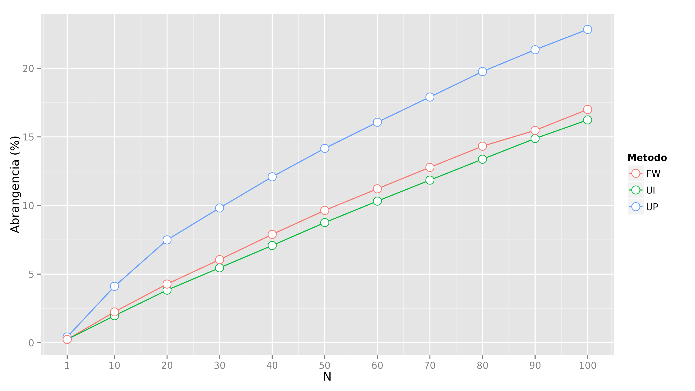
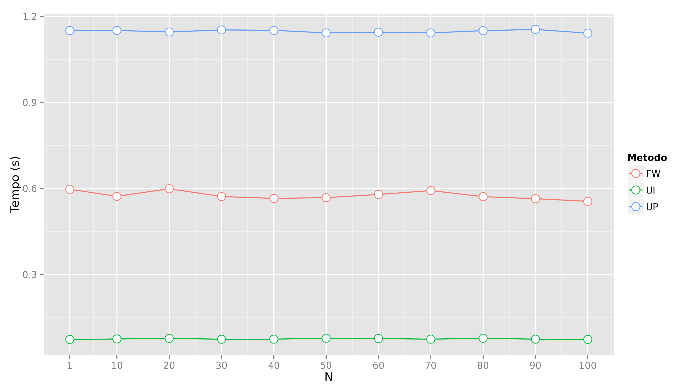


Figura 2 - Abrangência por tamanho de N



Apesar de alta qualidade das recomendações do método UP, este possui também a maior complexidade computacional. Seu tempo de execução é 2 vezes maior que o do método FW e 4 vezes maior que o do método UI, como podemos ver na figura 3.

Figura 3 - Tempo de execução por tamanho de N



**6.2. Percentual da base de aprendizado**

A medida que o percentual da base de aprendizados aumenta, a precisão de todos os métodos cresce ligeiramente. Isso é consequência do caráter colaborativo dos algoritmos, já que a qualidade da recomendação depende da quantidade total de dados. Entretanto, pôde-se observar que a abrangência e a medida são praticamente constantes para valores crescentes de de modo que esse parâmetro não tem grande relevância para o sucesso do sistema de recomendação.

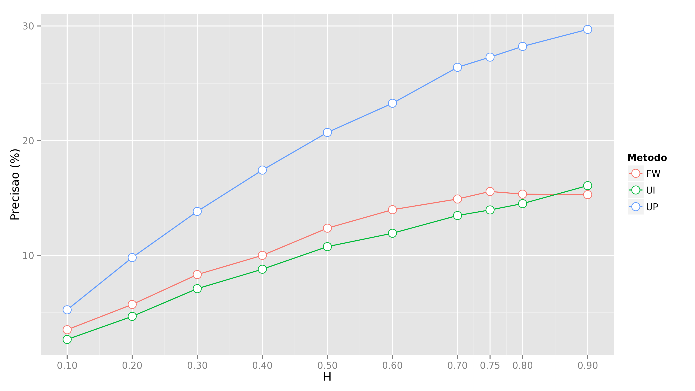
O parâmetro não exerce nenhuma influência sobre o tempo de execução dos métodos UP e UI, mas apenas sobre o método FW. Isso ocorre porque a etapa de maior custo computacional é linearmente dependente da quantidade de usuários . Quanto menos usuários-teste, mais veloz é o algoritmo.

**6.3. Percentual de avaliações “escondidas” dos usuários-teste na validação cruzada H**

Quanto maior o número de avaliações “escondidas”, mais fácil é acertar os itens dos usuários-teste, pois a lista de recomendação é pequena em relação ao total de itens positivamente avaliados pelo usuário. Por esse motivo, a precisão cresce com para todos os métodos.

Para o algoritmo FW, a precisão atinge seu máximo em e depois decresce ligeiramente (Figura 4).

Figura 4 - Precisão por percentual de avaliações escondidas

 Isso ocorre porque o cálculo dos pesos depende da quantidade de avaliações . Existe, pois, um compromisso (*tradeoff* ) entre facilidade de se acertar

itens avaliados quando há muitas avaliações escondidas e a dificuldade de se estimar quando não há muitos dados de avaliações.

Ao passo que a precisão dos métodos aumenta com , a abrangência diminui. Visto que a quantidade de itens da lista *top-N* é fixa, quanto maior o número de itens “escondidos”, mais difícil é de se retornar todos os itens relevantes.

O resultado de uma precisão crescente em função de e uma abrangência decrescente é que a medida possui um ponto de máximo. Para todos os métodos, o valor máximo é tal que .

Quanto ao tempo de execução, a influência é a mesma do parâmetro : para o método FW, a etapa de maior custo computacional é linearmente dependente da quantidade de itens Quanto menos avaliações de itens dos usuários-teste, mais veloz é o algoritmo.

**6.4. Valor mínimo para avaliações positivas**

Contrariamente ao que esperávamos, tornar o algoritmo mais “seletivo” não melhora sua precisão. Apesar de o valor mínimo estar intimamente ligado com a noção de “avaliação positiva” e de entrar no cálculo de parâmetros importantes dos métodos, esse parâmetro pouco influencia a precisão para .

Esse resultado pode ser explicado porque a maioria das avaliações são positivas, e portanto tem quase o mesmo efeito de . Isso não ocorre somente pelo fato de os clientes comprarem itens similares a seus gostos, e portanto de raramente se decepcionarem, mas também pelo fato de os usuários terem menos disposição para dar avaliações negativas. Esse fenômeno se chama *hidden* *feedback*, e se caracteriza pelo fato de que os itens avaliados não são escolhidos ao acaso, mas sim por despertarem aspectos de interesse das preferências do usuário, indo além dos valores numéricos das avaliações (11).

Ao se analisar a abrangência dos métodos, a seletividade influencia na recomendação. Quanto maior , menor é a quantidade de itens muito bem avaliados. Estes possuem elevada ponderação/correlação e são facilmente escolhidos pelos algoritmos. Por esse motivo, o desempenho do sistema é melhor.

A complexidade computacional dos algoritmos também depende de , já que mais ou menos parâmetros são analisados no cálculo da TF-IDF (métodos UI e UP) e dos pesos dos atributos (método FW).

Um detalhe a se observar é que a precisão é nula e a abrangência é inexistente para , já que todas as avaliações pertencem ao conjunto {1, 2, 3, 4, 5}. Para o algoritmo UI, tanto a precisão quanto a abrangência são nulas para M = 4.

**6.5. Número de vizinhos mais próximos**

O único método que recomenda itens com base nos vizinhos mais próximos é o UP. Percebe-se que com o aumento de , a precisão e a abrangência caem, pois a vizinhança se torna excessivamente grande e repleta de usuários sem muita similaridade com o usuário-teste. Pôde-se observar que o valor máximo de precisão e abrangência ocorre para , onde .

**6.6. Conjunto de atributos dos itens**

Para o banco de dados 100k-IMDB, o conjunto de atributos dos itens é ={data de lançamento, gênero, ano, duração, orçamento, avaliação, votos}. A fim de se avaliar a performance dos algoritmos mediante a remoção em determinados atributos, decidimos excluir do conjunto as features {data de lançamento, ano}, pois julgamos que elas não eram tratadas corretamente pelos métodos UI e UP.

O resultado desse teste se observa por exemplo na Figura 5, em que a precisão de todos os métodos melhora substancialmente. Da mesma forma, a abrangência também aumenta para todos os métodos, como se vê na Figura 6.

A conclusão desse experimento é que aumentar a quantidade de atributos dos itens não aumenta necessariamente a qualidade do algoritmo de recomendação. De fato, o algoritmo deve estar preparado para “aprender” quais são as features relevantes para cada usuário e eventualmente descartar automaticamente os atributos desnecessários.

Figura 5 - Precisão por tamanho da lista de recomendações excluindo-se alguns atributos

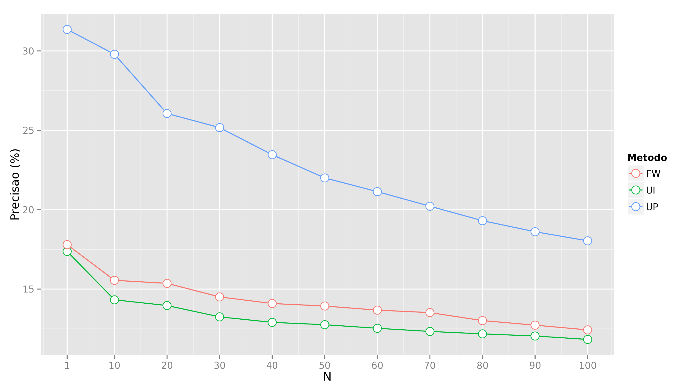
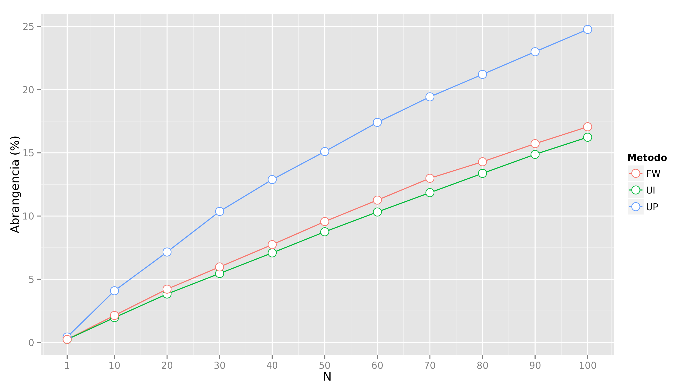


Figura 6 - Abrangência por tamanho da lista de recomendações excluindo-se alguns atributos



**6.7. Medida de distância entre atributos**

Contrariamente ao esperado, o emprego de medidas de distância específicas para o método FW não melhorou a qualidade da recomendação. O uso do índice de Jaccard para os atributos do tipo gênero, em vez da distância absoluta, decresceu a precisão e abrangência do método. Nota-se a importância da determinação de uma medida de distância correta, pois a influência no desempenho do método é drástica.

**6.8. Pesos dos atributos**

A fim de avaliar a influência da quantidade de pesos utilizados na recomendação do método FW, realizamos os testes selecionando apenas os maiores elementos

Observou-se que a qualidade da recomendação, tanto em termos de precisão e abrangência são aproximadamente independentes da quantidade de pesos . Para , as medidas de desempenho se tornam constantes.

**7. Conclusão**

Este Trabalho de Conclusão de Curso cumpriu seus objetivos e antedeu aos requisitos estabelecidos no início do projeto. Foi elaborada uma biblioteca para sistemas de recomendação de produtos de comércios on-line e foi estabelecida uma respectiva análise de desempenho dos algoritmos de recomendação.

As avaliações de desempenho dos métodos propostos na biblioteca deste trabalho verificaram resultados já conhecidos no meio acadêmico. Em particular, a dependência entre qualidade de recomendação e tamanho da lista de sugestões se comprovou (impacto de ).

Além disso, mostramos que um banco de dados com maior quantidade de avaliações (impacto de ) tem mais relevância que um banco de dados com mais usuários (impacto de ).

Outro resultado do trabalho foi a comprovação do fenômeno de *hidden feedback* (impacto de ). Mesmo que construamos métodos embasados na “avaliação positiva” dos usuários, esse parâmetro pode não ter tanta influência, visto que a maioria das avaliações dos clientes já são de fato positivas.

Também foi verificada a influência da quantidade de vizinhos mais próximos em algoritmos que usam essa metodologia colaborativa (impacto de ). Apesar de influenciar na qualidade da recomendação, esse parâmetro desempenha papel secundário.

Uma outra conclusão importante deste trabalho foi da importância de se escolher a priori o conjunto de atributos dos itens (impacto de ). A categorização excessiva dos itens pode ser maléfica para a recomendação, caso as *features* não tenham relevância para os usuários.

Avaliamos também diferentes medidas de distância entre os atributos (impacto de ). A medida da diferença em valor absoluto foi comparada com outros índices, como o índice Jaccard, para uma lista de gêneros, e verificou-se que a distância resulta em melhor qualidade de recomendação. Vale ressaltar a importância da escolha das medidas de distância, visto seu impacto no desempenho do sistema.

Por fim, avaliamos também a quantidade de pesos dos atributos no método FW (impacto de ). Vimos que a quantidade de não tem grande impacto na recomendação, visto que o valor dos pesos, em si, já é suficiente para alterar a qualidade das sugestões.

Apenas o método UP atingiu os requisitos funcionais em termos de precisão e abrangência, para uma combinação específica de parâmetros, como valores pequenos de . Tanto para esse algoritmo quanto para o método FW, o desempenho é sensivelmente inferior ao relatado nos artigos de referência. O motivo por trás disso é a dissimilaridade entre os bancos de dados. Assim como foi confirmado, o emprego de bases com mais recomendações influencia grandemente na qualidade das recomendações. Para se obter um benchmarking mais fiel, seria necessário utilizar o banco de dados dos autores de referência.

**8. Trabalhos Futuros**

A extensão desse Trabalho de Conclusão de Curso pode se dar de diversas maneiras, tanto na área acadêmica quanto na área empresarial. Seguindo o atual encaminhamento do projeto, a principal oportunidade do nosso trabalho é a criação de um serviço de um “Sistema de Recomendação nas Nuvens”.

Desejamos eliminar as restrições quanto a entrada e saída de dados, de forma que elas fossem completamente arbitrárias. O objetivo é que o usuário possa informar ao sistema como é formado sua base, e que todo o tratamento preliminar seja feito automaticamente.

É possível explorar também a construção de um driver que possibilite a conexão entre o sistema de recomendação e um banco de dados SQL, sem que seja necessária a etapa intermediária de arquivos CSV para aquisição de dados. Em seguida, é importante elaborar um website para o sistema de recomendação e exportar toda a lógica para um servidor dedicado.

Outra melhoria desejada é a reconstrução dos métodos na linguagem de programação C, a fim de melhorar a performance computacional. Dessa forma, o serviço de “sistema de recomendação nas nuvens” estaria completo e poderia ser utilizado por comércios on-line reais.

Também seria desejável, para uma avaliação mais completa do trabalho, o emprego dos métodos computacionais em um banco de dados de um e-commerce real. Apesar de termos contatado diversas lojas de comércio online, devido a impedimentos administrativos, não obtivemos sucesso em firmar uma parceria com essas lojas.

No campo acadêmico, há muito espaço para melhorias nos algoritmos de recomendação. As metodologias de solução de cada um dos sistemas deveriam ser debatidas ao máximo, de modo a explorar casos de uso particulares e a propor mudanças e otimizações.

Faz-se necessário responder a perguntas como “O que acontece com itens ou usuários sem nenhuma avaliação?” e “Qual o desempenho dos métodos para outros bancos de dados?”.

**Referências Bibliográficas**

[1] EMARKETER. B2C Ecommerce Climbs Worldwide, as Emerging Markets Drive Sales Higher. 2013. Disponível em: <http://www.emarketer.com/Article/B2C-Ecommerce-Climbs-Worldwide-Emerging-Markets-Drive-Sales-Higher/1010004>.

[2] EMARKETER. Global B2C Ecommerce Sales to Hit $1.5 Trillion This Year Driven by Growth in Emerging Markets. 2014. Disponível em: <http://www.emarketer.com/Article/Global-B2C-Ecommerce-Sales-Hit-15-Trillion-This-Year-Driven-by-Growth-Emerging-Markets/1010575>.

[3] COOPERS, P. W. The Go-to-Market Revolution - Igniting Growth with Marketing, Sales, and Pricing. 2014. Disponível em: <https://www.bcgperspectives.com/content/articles/go\_to\_market\_strategy\_growth\_go\_to\_market\_revolution\_igniting\_growth\_marketing\_sales\_pricing>.

[4] RICCI, L. R. F.; SHAPIRA, B. Introduction to recommender systems handbook. In: Recommender Systems Handbook. [S.l.]: Springer, 2011. p. 1–35.

[5] AMATRIAIN, X. Netflix Recommendations: Beyond the 5 stars. 2012. Disponível em: <http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html>.

[6] MARSHALL, M. Aggregate Knowledge raises $5M from Kleiner, on a roll. 2006. Disponível em: <http://venturebeat.com/2006/12/10/aggregate-knowledge-raises-5m-from-kleiner-on-a-roll/>.

[7] DAS, A. S. et al. Google news personalization: scalable online collaborative filtering. In: ACM. Proceedings of the 16th international conference on World Wide Web. [S.l.], 2007. p. 271–280.

[8] SARWAR, B. et al. Item-based collaborative filtering recommendation algorithms. In: ACM. Proceedings of the 10th international conference on World Wide Web. [S.l.], 2001. p. 285–29.

[9] SYMEONIDIS, P.; NANOPOULOS, A.; MANOLOPOULOS, Y. Feature-weighted user model for recommender systems. In: User Modeling 2007. [S.l.]: Springer, 2007. p. 97–106

[10] DEBNATH, S.; GANGULY, N.; MITRA, P. Feature weighting in content based recommendation system using social network analysis. In: ACM. Proceedings of the 17th international conference on World Wide Web. [S.l.], 2008. p. 1041–1042.

[11]LOPS, P.; GEMMIS, M. de; SEMERARO, G. Advances in collaborative filtering. In: Recommender Systems Handbook. [S.l.]: Springer, 2011. p. 145–184.