Universidade de São Paulo

Instituto de Matemática e Estatística

Bacharelado em Ciências da Computação

Estudo comparativo de Algoritmos de Recomendação

Aluno:
Marcos M. Takahashi

Orientador:
Roberto Hirata Jr

Fevereiro, 2015

Resumo

Com o advento das mídias socias, tecnologia RFID, computação em nuvem entre outras, é possível ter acesso a dados digitais em um volume, velocidade e variedade inéditos e extrair informação desses dados são um grande diferencial para as empresas. Com base nisto surgiu o Big Data, que é o conjunto de soluções tecnológicas capazes de lidar com essa imensa gama de dados estruturados e não-estruturados.

Uma das aplicações de Big Data são os Sistemas de Recomendação que utilizam diversos dados de interação do cliente com o site e outras ferramentas e com isso personalizam os produtos exibidos ao cliente. O Comércio Eletrônico em especial utiliza muito as recomendações para se destacar em meio aos concorrentes, tentando assim exibir produtos de interesse dos clientes que gastam cada vez menos tempo nos sites.

Este trabalho apresenta a implementação e testes de dois algoritmos de Sistemas de Recomendação, comparando-os e avaliando os retornos financeiros gerados.

Sumário

Lista de Figuras

Capítulo 1

Introdução

1.1 Motivação

Os comércios eletrônicos em geral estão constantemente em busca de novas formas de engajar os clientes já que existem diversos concorrentes oferecendo os mesmos produtos a preços semelhantes que fazem com que o tempo que cada cliente passa nos sites seja cada vez menor, com média de 4 a 5 visualizações de página por visita. Para que o cliente encontre o produto desejado nesse curto intervalo de tempo em que acessa o site é que são utilizados algoritmos de recomendação poderosos que abrangem dados de compra, navegação e até interações com mídias sociais.

Mas um Sistema de Recomendação eficiente não consiste somente em utilizar todos os dados disponíveis. É preciso utilizar os tipos de dados certos com a metodologia que mais se adequa ao contexto e por isso a recomendação é personalizada a cada empresa, levando algumas a até promover concursos para encontrar a com melhor desempenho.

Aplicando algoritmos eficientes e modelando e utilizando corretamente os dados é possível ter ganhos reais com os Sistemas de Recomendação, tanto economicamente (melhorando o desempenho do site e consequentemente vendendo mais) quanto socialmente (as pessoas economizam tempo para encontrar os produtos desejados).

1.2 Sistemas de Recomendação

Um Sistema de Recomendação é um conjunto de técnicas e algoritmos que seleciona itens tendo como base os dados de interação e interesses dos usuários. Esses itens recomendados podem ser de diversos tipos como livros, filmes, música, vídeos, produtos em loja de comércio eletrônico, etc.

Muitas empresas utilizam sistemas poderosos para obter vantagem competitiva e, entre eles, casos como Amazon, Netflix e Last.fm são amplamente reconhecidos.

1.2.1 Amazon

A Amazon é referência em se tratando de serviços de internet e possui um sistema de recomendação de produtos muito poderoso, considerado um dos mais efetivos. Esse sistema é um dos responsáveis pela alta Conversão do site (transações dividido por visitas) que já chegou a ser de 10% e se baseia em diversas fontes de dados tais como: produtos comprados pelo usuário no passado, quais itens estão no carrinho de compras virtual, itens curtidos ou avaliados, o que outros consumidores viram ou compraram, entre outros [1]. O site da Amazon é completamente personalizado de acordo com o cliente e conforme se navega e interage com o site, os produtos exibidos se alteram.

Uma das recomendações utilizadas é o *Item-to-Item Collaborative Filtering* que é descrito no artigo [22] e por ser de fácil implementação, pode ser utilizado por qualquer tipo de site.

1.2.2 Netflix

A Netflix é uma empresa que utiliza sistemas de recomendação muito poderosos devido a enorme base de dados que possui. Cada filme visto e avaliado pelo usuário é utilizado como base para as próximas recomendações e isso faz com que 75 a 80% do que é assistido no site provenha das recomendações ao invés da busca, tanto que fala-se em futuramente só sugerir de 3 a 4 filmes ao usuário, sendo que estes serão exatamente aqueles que o usuário quer assistir [6].

De 2006 a 2009 promoveram o Netflix Prize, uma competição aberta a todos e que tinha como objetivo desenvolver o melhor algoritmo para prever notas de usuários em filmes. Diversos algoritmos surgiram com essa competição e a equipe vencedora recebeu um prêmio de 1 milhão de dólares por melhorar as recomendações da Netflix em 10,06%.

1.2.3 Last.fm

Last.fm é um site de músicas que utiliza um sistema recomendador que cria o perfil detalhado do gosto musical do usuário a partir das músicas escutadas tanto no próprio site quanto em estações de rádio, computador ou dispositivos portáteis de música. A informação é transferida pelo próprio

software que toca as músicas ou por plugins e então são utilizadas para criar uma seleção de músicas ao gosto do usuário [5].

1.3 Objetivos

- Estudar e analisar algoritmos e Sistemas de Recomendação
- Implementar os algoritmos *Large-scale Parallel Collaborative Filtering* e *Fast Context-aware Recommendations with Factorization Machines*;
- Testar, validar e comparar esses algoritmos em um contexto restrito do mundo real (e-commerce de Móveis e Utilidades Domésticas);
- Avaliar o retorno financeiro de cada algoritmo.

Capítulo 2

Conceitos Estudados

2.1 Aprendizagem Computacional

Aprendizagem Computacional (Machine Learning) é uma subárea da Ciência da Computação e da Estatística que estuda e desenvolve algoritmos e técnicas que aprimoram o seu desempenho e acurácia segundo uma medida de erro e com a experiência que é dada por conjuntos de dados de treinamento [34].

Os algoritmos envolvem diversos conceitos e técnicas de Inteligência Artificial e Otimização e por isso muitas vezes os temas são relacionadas na literatura.

Algumas das aplicações de aprendizagem computacional são ferramentas de busca, detecção de fraudes em cartões de crédito, segmentação de clientes (usuários) entre outros. Em suma pode ser utilizada em qualquer meios em que um algoritmo estático baseado em regras não atende as necessidades.

A tarefa de aprendizagem computacional pode ser dividida em [30]:

- **Aprendizagem Supervisionada:** o algoritmo recebe exemplos de entradas e saídas e a partir disso "aprende"uma regra que mapeia as entradas em saídas;
- **Aprendizagem Não Superviosionada:** não são fornecidos exemplos de saída, somente de entrada, então é o próprio algoritmo que define os agrupamentos e os tipos de saída a serem devolvidos.

Existem diversos tipos de algoritmos de aprendizagem computacional cada um com uma metodologia e finalidade diferente e temos como exemplos Aprendizagem por Árvore de Decisão, Redes Bayesianas e Análise de Agrupamentos.

2.1.1 Aprendizado por Árvore de Decisão

É um algoritmo de aprendizagem supervisionada que utiliza uma árvore de decisão como modelo preditivo que mapeia as observações de um item para concluir sobre o valor desejado acerca do item.

As árvores de decisão recebem como entrada um conjunto de atributos e retorna uma decisão que é o valor predito para a entrada. Sua estrutura é formada por nós e ramos, sendo que cada nó contém um teste em um atributo e cada ramo representa um possível valor do atributo. Os nós que não possuem nenhum ramo saindo dele são chamados folhas e especifica o valor de retorno se a folha for atingida [28].

Como exemplo de árvore de decisão podemos considerar o problema de esperar para jantar em um restaurante. O objetivo é definir se deve esperar ou não e são considerados os seguintes atributos: alternar de restaurante, ir para um bar, dia semana, estar com fome, número de fregueses, preço da comida, clima, se foi feita reserva, tipo do restaurante, estimativa de espera.

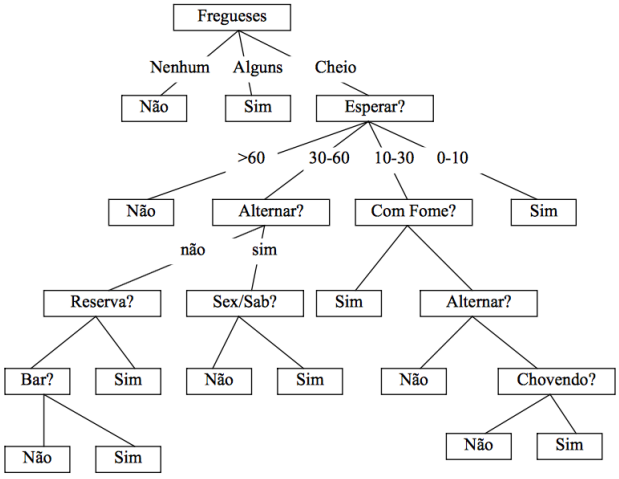

Figura 2.1: Exemplo de Árvore de Decisão para o problema de espera para jantar no Restaurante [13]

2.1.2 Redes Bayesianas

Redes bayesianas são modelos baseados em grafos para tomada de decisão baseado na incerteza, onde os nós representam as variáveis (discreta ou contínua), e os arcos representam a conexão direta entre eles. Possui esse nome pois utiliza probabilidade bayesiana para determinar as chances de ocorrência das diversas possibilidades [29].

Cada nó é associado com uma função de probabilidade que recebe como entrada um conjunto de valores para as variáveis pais do nó e retorna a probabilidade (ou distribuição de probabilidade) das variáveis representadas pelo nó.

Um exemplo de rede Bayesiana pode ser a relação de probabilidade entre doenças e sintomas. Dados os sintomas, a rede pode ser usada para calcular as probabilidades de presença das doenças.

2.1.3 Análise de Agrupamentos

Análise de Agrupamentos (ou Clustering) é a tarefa de agrupar um conjunto de dados de forma que os objetos que pertencem ao mesmo grupo (chamado de cluster) sejam mais semelhantes (de acordo com alguns critérios) entre eles do que aos que estão em outros grupos (clusters) [21].

Existem diversos tipos de algoritmos para agrupamento dos objetos, seja por distância entre os pontos ou por aproximação dos centróides, e cada um pode retornar um resultado diferente.

Um exemplo de Agrupamento seria a divisão das pessoas de uma Rede Social em grupos, identificando assim comunidades de determinados temas.

2.2 Sistemas de Recomendação

Diante da cada vez maior exposição a diferentes tipos de informações, faz-se necessário ajudar as pessoas a encontrarem aquilo que procuram e nesse contexto é que surgiram os sistemas de recomendação. Estes através de modelagem de dados e aplicação de algoritmos tentam prever a nota (rating) ou preferência de um usuário a um determinado item [26].

Alguns exemplos de recomendações são: um filme de romance a um casal ao invés de um filme de ação, um artigo de Sistemas de Recomendação ao invés de um artigo de Moda a um estudante de Ciências da Computação, uma viagem a Disney a uma família ao invés de uma viagem a Veneza (cidade do Amor) entre outros.

Em um problema de recomendação, as principais entidades são o usuário e o item e o objetivo é recomendar os itens com melhores notas aos usuários.

Os sistemas de recomendação são um tipo de Sistemas de Aprendizagem Computacional e fazem uso de algoritmos que são divididos em 3 grupos de acordo com [24]: Collaborative filtering, Content-based filtering e Hybrid Recommender Systems.

2.2.1 Collaborative Filtering

Muito utilizadas no comércio eletrônico em geral no formato "Quem comprou X também comprou Y", recomendam items que usuários semelhantes compraram ou interagiram.

Através dos items já comprados ou interagidos pelo usuário A, procura por usuários semelhantes que tiveram um comportamento igual ou similar (interagiram com os mesmos items) e seleciona itens que o usuário A ainda não interagiu. Então, recomenda aqueles com maior nota entre os usuários [18].

Por ser um tipo de recomendação baseado exclusivamente nos dados relacionados à venda (*Quem* comprou *qual* produto), é o mais utilizado nos sites em geral e possui uma implementação fácil e rápida.

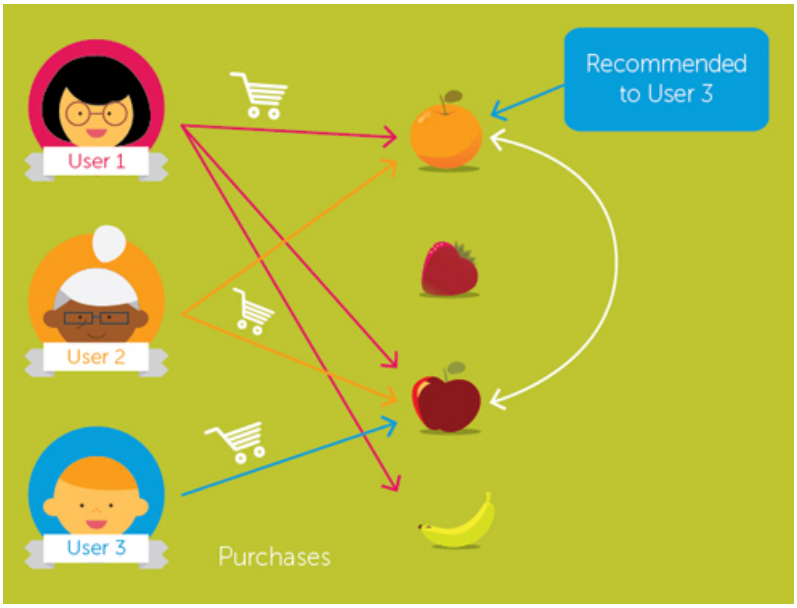

Figura 2.2: Como podemos ver na figura, o Usuário 3 comprou maçã como os outros usuários e então procura-se uma fruta que os outros Usuários compraram e o Usuário 3 ainda não comprou. Pelo exemplo pode-se ver que esta fruta é a laranja e então ela é recomendada. (Imagem obtida em [3])

2.2.2 Content-based Filtering

Recomendações do tipo Content-based recomendam itens similares àqueles que o usuário comprou ou interagiu no passado. De acordo com [23], o

processo básico consiste em cruzar os atributos do perfil do usuário (interesses e preferências) com os atributos dos itens, para recomendar ao usuário novos itens.

O sistema analisa os documentos, descrições e características dos itens interagidos anteriormente pelo usuário e com isso cria o perfil dos interesses do usuário. Dessa forma, recomenda novos itens com os atributos semelhantes ao perfil do usuário.

A maioria dos sistemas Content-based utilizam o texto ou descrição do item. Então os atributos normalmente são um conjunto de palavras.

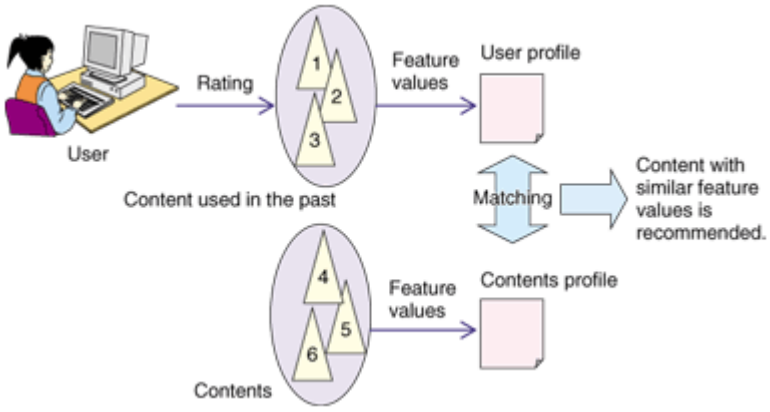

Figura 2.3: Através dos items avaliados pelo usuário, cria-se o perfil do mesmo e recomenda itens similares ao perfil [2]

2.2.3 Hybrid Recommender Systems

São algoritmos que combinam Collaborative com Content-based e podem ser feitos de diversas formas diferentes: aplicando os dois separados e juntando os resultados depois, adicionando a capacidade de content-based a Collaborative Filtering (ou vice-versa) ou unificando as duas abordagens em um único modelo.

Em [27] encontram-se alguns exemplos de combinação entre os algoritmos e os ganhos obtidos com eles.

2.2.4 Context-aware Recommender Systems

Ao contrário dos algoritmos tradicionais, incorpora além de informações do item e do usuário, dados do contexto que levaram ao acontecimento do evento como tempo, clima, local, pessoas que estavam juntas entre outras.

Existem muitos casos de recomendações que não podem levar em consideração somente os dados do item ou do usuário, como conteúdo personalizado de um site de filmes, sites de viagens e até sites de notícias. Como exemplo, podemos considerar uma pessoa que assiste televisão. Ela terá um comportamento diferente durante a semana, quando assiste ao noticiário à noite e aos finais de semana, quando assiste ao jogo de futebol ou a um filme. (Exemplo extraído de [15])

A incorporação do contexto permite personalizar ainda mais a recomendação e criar experiências realmente válidas ao usuário, assim como expressou C.K.Prahalad em [19] - "atingir e tocar clientes em qualquer lugar e a qualquer momento significa que as empresas precisam entregar não somente produtos competitivos, mas também experiências únicas e reais lapidadas pelo contexto do cliente."

A formulação da recomendação se dá através da seguinte função:

$$f : User \times Item \times Context \rightarrow Rating \tag{2.1}$$

onde $User$ e $Item$ são os domínios de usuário e item respectivamente. $Rating$ é o domínio dos ratings (notas atribuídas aos itens) e $Context$ são as informações de contexto associadas ao evento.

2.3 Mineração de Dados

Mineração de dados (Data Mining) é o processo de explorar grandes volumes de dados à procura de padrões para detectar relacionamentos entre variáveis, detectando assim novos subconjuntos de dados [17].

O objetivo principal do processo é extrair informação de um conjunto de dados estruturando-a para que possa ser utilizada posteriormente, mas também envolve manipulação, pré-processamento de dados, modelagem e inferência estatística, métricas entre outros.

É através da mineração de dados que as empresas extraem as informações relevantes do grande volume de dados que possuem e por isso também é utilizada na área de recomendação para preparar os dados que servem de entrada nos algoritmos.

A aplicação específica de Mineração de Dados a textos é chamada de Mineração de Textos (Text Mining) e é muito utilizada para identificação e indexação de palavras chaves de textos, que são de uso em sistemas de busca e também em Sistemas de Recomendação Content-based [12].

2.4 Análise de Redes Sociais

Do ponto de vista da computação, Análise de Redes Sociais (Social Network Analysis) é a análise (mapeamento e métricas) de relacionamentos e fluxos entre pessoas, grupos, organizações e quaisquer outras entidades de informação e ou conhecimento. Ela fornece uma análise visual e outra matemática dos relacionamentos entre as pessoas e é uma área que vem se evidenciando mais e mais com o cada vez maior uso das redes sociais em todo o mundo [11].

Esses relacionamentos são armazenados em estruturas chamadas Grafos Sociais (Social Graph) que contém as entidades em nós, e seus relacionamentos com outras entidades nas arestas.

A Análise de Redes Sociais possui diversas aplicações sendo algumas delas em agregação e mineração de dados, análise de atributos e comportamentos de usuários, sistemas de recomendação entre outros.

2.5 E-mail Marketing

É uma forma de utilizar o e-mail como ferramenta de marketing direto, respeitando normas e procedimentos pré-definidos.

Diferente das mensagens indesejadas (spams), deve-se ter o consentimento do cliente acerca do recebimento destas mensagens e é isso que torna o E-mail Marketing uma das melhores ferramentas de comunicação direta com o cliente, já que só atinge aqueles que já tiveram algum contato com a empresa [16].

Este é um ramo do marketing on-line que possibilita a aplicação de muita inteligência através do uso de diversos dados dos usuários e permite a aplicação de sistemas de recomendação.

Capítulo 3

Materiais e Métodos

3.1 Facebook Graph API

O Facebook é a maior rede social existente atualmente e possui uma grande base de dados com um crescimento de 500 Tera Bytes por dia (dados de 2012) [4].

A equipe do Facebook recentemente desenvolveu o Graph API que é uma interface fornecida pelo Facebook que permite o acesso a todos os dados públicos dos usuários e páginas, além do acesso a dados restritos de entidades (usuário ou página) que possuem relação com o usuário logado, através de um protocolo HTTP.

Os dados são armazenados em um Grafo Social e a estrutura do grafo e consequentemente dos dados se dá da seguinte forma:

- nós (são "coisas"como usuário, foto, página, comentário)
- arestas (as conexões entre as "coisas", como Fotos das Páginas ou Comentários das Fotos)
- campos (informações das "coisas", como o aniversário do Usuário ou o nome da Página).

Todas as requisições são feitas através de métodos GET e POST em graph.facebook.com, mas para que sejam válidas é necessário ter um Token de acesso que pode ser obtido manualmente ou através de ferramenta.

É possível acessar os dados facilmente pelo Graph API Explorer que é uma plataforma web desenvolvida pelo próprio Facebook para facilitar o acesso aos dados (https://developers.facebook.com/tools/explorer/). Para a automatização das extrações dos dados é necessário implementar um código próprio ou utilizar pacotes (bibliotecas) já prontos.

3.2 TF-IDF

TF-IDF é uma métrica comumente utilizada em Text-Mining e expressa a importância da palavra dentro de um conjunto de documentos (corpus). Conforme o número de vezes que a palavra aparece no documento, passa a ter maior importância dentro desse documento. Mas se essa mesma palavra aparece diversas vezes em vários documentos do conjunto, passa a ter menor importância dentro de todo o conjunto.

Assim, dá-se importância àquelas palavras que aparecem diversas vezes no documento e que podem ser consideradas palavras-chaves (como "recomendação" em um artigo sobre sistemas de recomendação) e penaliza as palavras que aparecem em diversos documentos do conjunto (como 'computação' em um conjunto de artigos de computação onde o artigo de sistemas de recomendação está inserido) [9].

Para calcular o TF-IDF, calcula-se a Frequência do Termo (Term Frequency ou TF) e a Frequência Inversa ao Documento (Inverse Document Frequency ou IDF) separadamente e então são multiplicados entre si para chegar ao valor final.

TF mensura a frequência de um termo em um documento. Como o termo pode aparecer conforme o tamanho do documento, é normalizado em relação ao número de termos do mesmo.

$$tf = \frac{\#\, ocorrencias\, termo\, t\, no\, documento}{\#\, total\, termos\, do\, documento} \tag{3.1}$$

IDF mensura a importância do termo no corpus. Através de IDF atribui-se um peso menor aos termos frequentes e então valoriza-se os termos raros.

$$idf = log\left(\frac{\#\, total\, de\, documentos}{\#\, documentos\, com\, termo\, t}\right) \tag{3.2}$$

E então, tf-idf pode ser expresso como:

$$tfidf = tf * idf \tag{3.3}$$

3.3 Base de dados - Loja

No mundo online muitos dados podem ser obtidos através de diversas ferramentas de Web Analytics como Google Analytics, Webtrends, KissMetrics entre outros, que rastreiam todo o fluxo de navegação do usuário a partir do momento em que este entra no site.

O fluxo normal de um usuário que efetua uma compra em um site de Comércio Eletrônico é:

- Visualização do produto (página de catálogo ou busca)
- Visualização de detalhe do produto
- Adição do produto ao carrinho
- Checkout - Finalização de compra (Cadastro, dados de pagamento e entrega do produto)
- Confirmação de compra

Este fluxo também é conhecido como o Funil de Conversão (Conversion Funnel) e tem esse nome pelo fato dos números diminuírem em cada etapa, como em um funil.

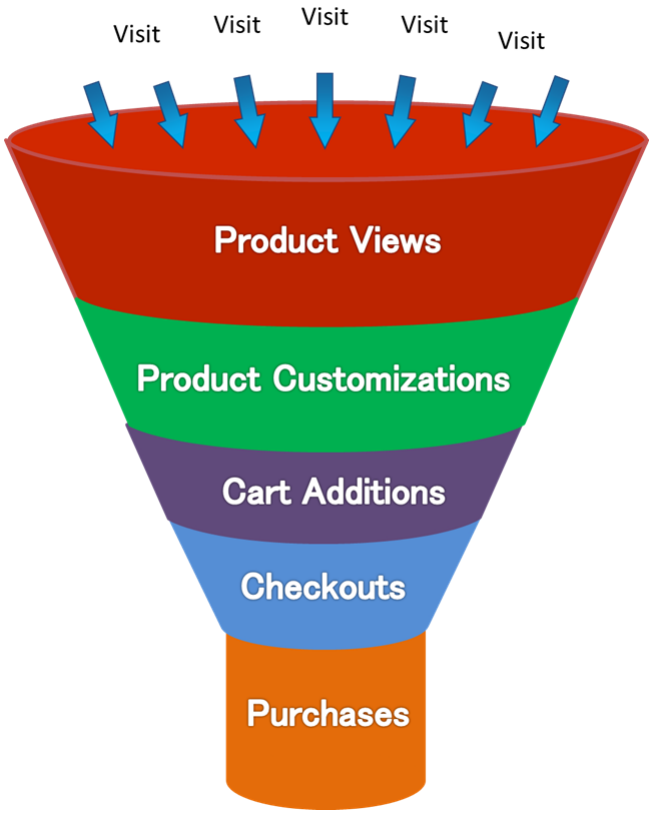

Figura 3.1: Funil de Conversão do E-commerce [25]

As principais métricas do processo de vendas no e-commerce estão representadas nesse funil, sendo algumas delas: Visualização do Produto, Visualização de Página do Produto, Adição ao Carrinho, Compra.

3.3.1 Visualização do Produto

Esta métrica representa a exibição do produto ao usuário, seja em uma página de catálogo ou como resultado de uma busca interna no site.

Um produto com alta visualização é um produto que está sempre em evidência.

3.3.2 Visualização de Página do Produto

Muito importante pois representa o interesse dos usuários em determinado produto.

Um alto valor representa que muitas pessoas se interessaram por esse produto e caso isto não reflita no número de vendas é ponto de atenção.

3.3.3 Adição ao Carrinho

Mensura o quanto o produto já foi adicionado ao carrinho e com isso a intenção de compra.

Existem casos em que o produto é adicionado ao carrinho mas não resulta em compra e esses são chamados de Abandono de Carrinho.

Abandono de carrinho é quando um usuário já demonstrou intenção de adquirir o mesmo mas por alguma razão não finalizou a compra. Esse abandono pode acontecer por diversas razões como preço do frete, tempo de entrega, falta de tempo para finalizar a compra, dificuldade na finalização da compra entre outros.

Existem lojas que enviam um lembrete ao usuário sobre o abandono do produto no carrinho e cerca de 1% dos usuários ao lembrarem do abandono compram os produtos que haviam abandonado. Dessa forma, muitas lojas revertem uma métrica negativa em algo posivito.

3.4 Large-scale Parallel Collaborative Filtering - CF

O algoritmo utiliza uma abordagem de Filtro Colaborativo (Collaborative Filtering) e foi desenvolvido por cientistas da IBM para a competição Netflix Prize e melhorou o desempenho do algoritmo que era utilizado pela Netflix

(CineMatch) em 5,91%. Apesar de não ser o que teve a melhor performance utiliza um método simples e escalável para grandes volumes de dados, o que o torna muito bom no contexto do E-commerce. Os detalhes do algoritmo podem ser encontrados no artigo [31].

3.4.1 Formulação do Problema

A maioria dos problemas de recomendação implica em utilizar uma função f, tal que:

$$f : User \times Item \rightarrow Rating \tag{3.4}$$

e utilizando os dados de ratings (nota) conhecidos, estimar os ratings dos pares $(user, item)$ que não são conhecidos.

Considere $R = \{r_{jk}\}_{n_u \times n_i}$ a matriz usuário-item onde cada elemento r_{jk} representa a nota (rating) atribuída pelo usuário j ao item k, podendo este ser um valor real ou inexistente; n_u denota o número de usuários e n_i denota o número de itens.

Assim como a maioria dos algoritmos, o objetivo é estimar os valores faltantes em R tendo como base os valores conhecidos.

A abordagem deste algoritmo consiste em decompor a matriz R em duas outras matrizes $U \subseteq \mathbb{R}^{n_u \times n_f}$ de características do usuário e $I \subseteq \mathbb{R}^{n_i \times n_f}$ de características do item, tal que $U \times I = \bar{R}$ e $\bar{R}$ é uma aproximação de R onde $\bar{r}_{jk} \approx r_{jk}$ para os valores conhecidos de R. n_f é o número de características (features) consideradas no modelo.

A partir desta decomposição, espera-se que $r_{jk} =< u_j; i_k >, \forall j, k$, mas na prática, minimiza-se uma função de perda para obter as matrizes U e I. A função utilizada é a de perda quadrática média (mean-square loss function) e a perda de um rating é definida por:

$$\mathcal{L}^2(r, u, i) = (r - < u, i >)^2 \tag{3.5}$$

e a perda total é a soma das perdas relativas a todos os ratings conhecidos de R.

$$\mathcal{L}^{emp}(R, U, I) = \frac{1}{n} \sum_{(j,k) \in N} \mathcal{L}^2(r_{jk}, u_j, i_k) \tag{3.6}$$

onde N é o conjunto de índices de todos os ratings conhecidos e n é o tamanho de N.

O problema de recomendação pode então ser formulado como o problema de encontrar os valores em que $\mathcal{L}^{emp}(R, U, I)$ é mínimo. Como R comumente

é uma matriz muito esparsa e pelo método em questão pode haver overfitting, aplica-se a regularização de Tikhonov [10] ao método e então, tem-se a equação:

$$\mathcal{L}_{\lambda}^{reg} = \mathcal{L}^{emp}(R, U, I) + \lambda(\|U\Gamma_U\|^2 + \|I\Gamma_I\|^2) \tag{3.7}$$

com Γ_U e Γ_I matrizes de Tikhonov selecionadas para o problema.

3.4.2 Cálculo das matrizes

Para se decompor a matriz R em U e I, os cientistas da IBM utilizam uma abordagem de Alternating-Least-Squares with Weighted-λ-Regulariation (ALS-WR), que pode ser encontrada em detalhes no artigo original deste algoritmo, em que se calcula cada matriz U e I separadamente, fixando uma delas para o cálculo da outra e vice-versa.

Adaptando a função 3.7, tem-se a função:

$$f(U, I) = \sum_{(j,k)\in N} (r_{jk} - u_j^T i_k)^2 + \lambda\left(\sum_j n_{u_j}\|u_j\|^2 + \sum_k n_{i_k}\|i_k\|^2\right) \tag{3.8}$$

onde n_{u_j} e n_{i_k} denotam o número de ratings do usuário j e item k respectivamente.

E para resolver U dado I, utiliza-se:

$$u_j = A_j^{-1}V_j, \quad \forall j \tag{3.9}$$

onde $A_j = I_{N_j}I_{N_j}^T + \lambda_{u_j}E$, $V_j = I_{N_j}R^T(j, N_j)$, E é a matriz identidade de $n_f \times n_f$, I_{N_j} denota a submatriz de I onde as colunas $k \in N_j$ são selecionadas e $R(j, N_j)$ é a matriz onde as colunas $k \in N_j$ da linha j é selecionada.

Da mema forma, para resolver I dado U:

$$i_k = A_k^{-1}V_k, \quad \forall j \tag{3.10}$$

onde $A_k = U_{N_k}U_{N_k}^T + \lambda_{i_k}E$, $V_k = U_{N_k}R^T(k, N_k)$, E é a matriz identidade de $n_f \times n_f$, U_{N_k} denota a submatriz de U onde as colunas $j \in N_k$ são selecionadas e $R(k, N_k)$ é a matriz onde as colunas $j \in N_k$ da linha k é selecionada.

3.4.3 Abordagem do algoritmo

Com base nos cálculos expressos anteriormente, o algoritmo pode ser resumido nos seguintes passos:

Passo 1 Inicializar a matriz I atribuindo o rating médio do item na primeira coluna e valores aleatórios próximos a 0 nas colunas restantes;

Passo 2 Fixar I e solucionar U minimizando a função objetivo através da função 3.9;

Passo 3 Fixar U e solucionar I minimizando a função objetivo através da função 3.10;

Passo 4 Repetir os passos 2 e 3 até satisfazer o critério de parada.

O critério de parada é baseado na Raiz Quadrada do Erro Quadrático Médio (Root Mean Squared Error - RMSE) dos ratings conhecidos. Após uma atualização dos valores de U e I, se a diferença entre o RMSE entre um teste e outro for menor do que $0,0001$, a iteração termina e considera-se as matrizes U e I obtidas.

3.5 Fast Context-aware Recommendations with Factorization Machines - FM

A formulação do problema de recomendação com a incorporação de contexto se dá através da seguinte função:

$$y : U \times I \times C_3 \times C_4 \ldots C_m \rightarrow \mathbb{R} \tag{3.11}$$

onde U são os usuários, I os items e $C_3, C_4, \ldots, C_m$ os diferentes contextos que compõem o problema. A abordagem tradicional modela o problema em m dimensões sendo que cada dimensão representa U, I ou $C_3, C_4, \ldots, C_m$ e cada um deles é um contexto (Usuário U e Item I podem ser considerados dimensões de contexto também) e k é a dimensão utilizada na fatoração.

O algoritmo *Multiverse Recommendation* [20] utiliza uma abordagem que pode ser processada com complexidade $O(k^m)$, sendo m o número de contextos e k o número de características (features). Este é um dos algoritmos com melhor desempenho neste tipo de recomendação, mas ainda assim apresenta uma complexidade computacional que a torna inviável a medida que se aumenta o tamanho de m ou k.

Uma das formas de resolver o problema da complexidade computacional é através de $FactorizationMachine$ que é utilizado neste algoritmo desenvolvido por Steffen Rendle. Os detalhes do algoritmo podem ser encontrados em seu artigo em [32].

3.5.1 Formulação do Problema

$FactorizationMachine(FMs)$ são modelos de classes genéricas que agrupam e podem imitar diversos tipos de sistemas de recomendação. Eles modelam todas interações entre pares de variáveis com a variável destino através de parâmetros de fatoração:

$$\hat{y}(x) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \hat{w}_{i,j} x_i x_j \tag{3.12}$$

onde $\hat{w}_{i,j}$ são as interações entre pares de variáveis

$$\hat{w}_{i,j} :=< v_i, v_j >= \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f} \tag{3.13}$$

e os parâmetros a serem estimados são:

$$w_0 \in \mathbb{R}, \quad W \in \mathbb{R}^n, \quad V \in \mathbb{R}^{n \times k} \tag{3.14}$$

Então, w_0 é o viés do modelo, w_i é a interação da i-ésima variável com a variável destino e $\hat{w}_{i,j}$ é a interação de pares de variáveis (i e j) com a variável destino.

Em [33] foi mostrado que 3.12 é equivalentes a:

$$\hat{y}(x) := w_0 + \sum_{i=1}^{n} w_i x_i + \frac{1}{2} \sum_{f=1}^{k} \left(\left(\sum_{i=1}^{n} v_{i,f} x_i \right)^2 - \sum_{i=1}^{n} {v_{i,f}}^2 {x_i}^2 \right) \tag{3.15}$$

e podem ser computadas eficientemente em $O(km)$.

3.5.2 Modelagem de dados

Uma grande variedade de dados de contexto pode ser utilizada com FMs e para isso, basta transformá-las como no exemplo da figura 3.2. Desta forma não é necessário armazenar os dados de forma diferente para cada tipo de domínio (User, Item ou Context).

O vetor final x é obtido simplesmente concatenando os diferentes domínios e esta é a entrada para a FM.

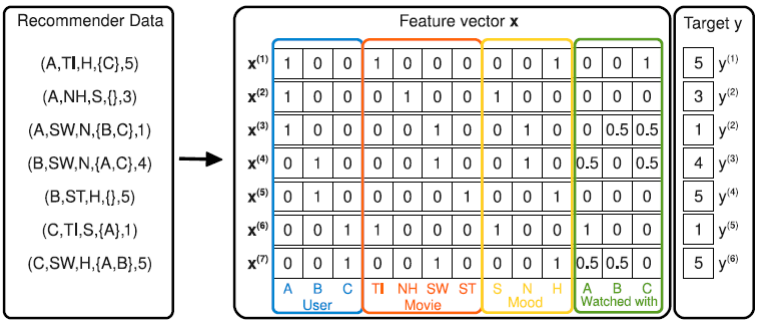

Figura 3.2: Exemplo de modelagem de dados para FMs

3.5.3 Cálculo dos parâmetros

Os parâmetros do problema são calculados através da função:

$$\theta = -\frac{\sum_{(x,y)\in S}(g_{(\theta)}(x) - y)h_\theta(x)}{\sum_{(x,y)\in S} {h_{(\theta)}}^2(x) + \lambda_{(\theta)}} \tag{3.16}$$

sendo que para cada parâmetro w_0, W e V, h_θ e $g_{(\theta)}$ são diferentes. Como forma de cálculo eficiente dos parâmetros, são computados o erro $e(x, y|\theta)$ e $q(x, f|\theta)$ para cada linha da base, tal que:

$$e(x, y|\theta) := \hat{y}(x|\theta) - y \tag{3.17}$$

$$q(x, f|\theta) := \sum_{i=1}^{n} v_{i,f} x_i \tag{3.18}$$

$$h_{(v_{l,f})} = x_l q(x, f|\theta) - {x_l}^2 v_{l,f} \tag{3.19}$$

e então, tem-se os parâmetros como:

$$w_0 = -\frac{\sum_{(x,y)\in S}(e(x, y|\theta) - w_0)}{|S| + \lambda_{(w_0)}} \tag{3.20}$$

$$w_i = -\frac{\sum_{(x,y)\in S}(e(x, y|\theta) - w_i x_i)x_i}{\sum_{(x,y)\in S} {x_i}^2 + \lambda_{(w_i)}} \quad i \in (1 \dots n) \tag{3.21}$$

$$v_{l,f} = -\frac{\sum_{(x,y)\in S}(e(x, y|\theta) - v_{l,f} h_{v_l,f}(x))h_{v_l,f}(x)}{\sum_{(x,y)\in S} {h_{v_l,f}}^2 + \lambda_{(v_l,f)}} \quad i \in (1 \dots n), f \in (1 \dots k) \tag{3.22}$$

3.5.4 Abordagem do algoritmo

O algoritmo pode ser expresso nos seguintes passos:

Passo 1 Inicializa os parâmetros com os seguintes valores: $w0 = 0$, $W = (0, \ldots, 0)$ e $V \sim \mathbb{N}(0; \sigma)$;

Passo 2 Computa os valores das matrizes $e(x, y|\theta)$ e $q(x, f|\theta)$ para cada $(x, y) \in S$ através das funções 3.17 e 3.18 respectivamente;

Passo 3 Computa o valor de w_0 com a função 3.20 e atualiza-se o valor de $e(x, y|\theta)$;

Passo 4 Computa os valores de W com a função 3.21 e atualiza-se o valor de $e(x, y|\theta)$;

Passo 5 Computa o valor de V com a função 3.22 e atualiza-se o valor de $e(x, y|\theta)$ e $q(x, f|\theta)$;

Passo 6 Repete os passos 3, 4 e 5 até atingir o critério de parada.

É utilizado como critério de parada o número de iterações do algoritmo.

Capítulo 4

Implementação

A implementação dos dois algoritmos, API de conexão com o Facebook e tratamento de dados, foram feitos em linguagem R por ser muito utilizada para Aprendizagem Computacional e Mineração de Dados, além de possuir muitos pacotes já desenvolvidos pela comunidade.

4.1 Facebook Graph API em R

A implementação do Facebook Graph API é uma adaptação do pacote RFacebook (https://github.com/pablobarbera/Rfacebook), em que foram modificadas as funções getPosts (retorna os posts de uma dada página) e getPostsLikes (retorna as pessoas que deram likes no dado post).

Com a função getPosts, obtém-se o id do post, data de criação e conteúdo (mensagem) dos posts de uma página e com a função getPostsLikes, retorna-se as pessoas que deram likes no post especificado.

Obteve-se dessa forma uma base de dados através da página da empresa analisada de todos os posts criados e para cada post, o conjunto das pessoas que deram likes.

4.1.1 Tratamento dos dados

O cruzamento dos dados foi feito utilizando o nome completo do usuário e isto resultou em um matching único de aproximadamente 10% dos usuários encontrados no Facebook com a base de dados da empresa analisada.

Foi aplicado um tratamento no conteúdo dos posts, extraindo somente as palavras que representassem alguma categoria de produtos da empresa e desta forma obteve-se uma base de clientes que curtiram categorias (palavras). Cruzou se cada categoria (palavra) curtida pelo cliente com os pedidos

de compra (ordens) posteriores a esse fato e chegou-se em uma base de pedidos dos clientes com as categorias curtidas antes de realizar a compra.

Nessa base aplicou-se o cálculo de Text Mining TF-IDF (Term Frequency - Inverse Document Frequency) obtendo-se assim um valor para cada categoria do pedido de compra.

Para os dados do facebook, considerou-se como documento o usuário e os termos as palavras (categorias) curtidas. O Corpus (conjunto de documentos) é o conjunto de todos usuários, então temos:

$$tf = \frac{\#\, ocorrencias\, categoria\, c\, com\, usuario\, u}{\#\, total\, categorias\, do\, usuario\, u} \tag{4.1}$$

$$idf = log\left(\frac{\#\, total\, usuarios}{\#\, usuarios\, com\, categoria\, c}\right) \tag{4.2}$$

Desta forma valoriza-se as categorias com maiores ocorrências com dado usuário, mas também valoriza-se categorias raras que tem poucas ocorrências no conjunto de todos usuários.

4.2 Dados da Loja

4.2.1 Surf History (histórico de navegação)

O Surf History é o nome dado ao histórico de navegação mas foi considerado somente o histórico de navegação nos produtos da loja (Visualização em Página de Produto).

Para cada produto visualizado (visto na página específica do produto) é computada uma visita desse usuário àquele produto e então, tem-se o número de visualizações de todos os produtos do usuário naquela visita.

Os produtos mais visualizados são aqueles que o usuário demonstrou maior interesse e dessa forma, esses são os casos considerados.

Cada visualização de produto da visita é vinculada à ordem posterior a essa visita e então obteve-se uma base de dados de visualizações de produtos e suas respectivas ordens.

4.2.2 Abandoned Cart (carrinho abandonado)

Cada produto abandonado pelo usuário foi computado e vinculado à ordem posterior ao abandono e então, obteve-se uma base de dados de abandono de produtos e suas respectivas ordens.

As bases de Abandoned Cart e Surf History foram normalizadas de forma que o número máximo de visualizações de um produto equivale a 50% de

um abandono desse mesmo produto e então obteve-se uma base final com a intenção de compra desse produto vinculada a uma ordem posterior.

4.2.3 Dados da compra

Alguns outros dados também foram considerados e estes estão diretamente vinculados à compra realizada.

- A data da compra e sua proximidade com datas especiais - existem datas especiais em que alguns produtos específicos são mais vendidos (produtos sazonais) e isto é uma forma de incorporar estes casos. Exemplos de datas especiais: Natal, Dia das Crianças, Dia dos Pais, Dia das Mães entre outros.

- Cupom de desconto - uma prática muito comum no comércio eletrônico é o oferecimento de cupons de desconto e estes alavancam as vendas em 20% ou mais.

- Recompra - é interessante saber se a ordem realizada pelo cliente é a primeira compra ou não e desta forma utiliza-se este parâmetro para saber se é uma compra ou recompra.

Os dados de compra foram extraídos de um banco de dados MySQL que é utilizado pela área de Business Intelligence da empresa.

4.3 Large-scale Parallel Collaborative Filtering - CF

O algoritmo foi desenvolvido para ser paralelizável e é isto que o torna tão eficiente. Mas, por ser um algoritmo de Aprendizagem Computacional para Recomendação, faz-se necessário ler e processar um grande volume de dados (matriz de mais 300.000 linhas e 50.000 colunas) e por isso, algumas medidas foram tomadas (seguindo [7] e [8]) para viabilizar a implementação.

Foram utilizados os seguintes pacotes do R para uma implementação eficiente: *data.table*, *bigmemory*, *bigalgebra*, *foreach* e *doParallel*.

- data.table: Este é um pacote muito utilizado pela comunidade, pois permite manipular dados como um banco de dados relacional através de chaves realizando cruzamentos (joins) de forma fácil e muito rápida. Além de ter um desempenho muito bom, permite o armazenamento de grande volume de dados e por isso todas as bases de dados manipuladas foram declaradas como data.table.

- bigmemory: Pacote próprio para uso de Bigdata que permite criar e manipular matrizes enormes, e além disso, possibilita o armazenamento das matrizes em arquivos o que o torna ideal para uso com paralelismo. [14]

- bigalgebra: É um pacote auxiliar ao bigmemory e permite realizar operações matriciais com matrizes do formato big.matrix.

- foreach e doParallel: Esses dois pacotes juntos permitem loops de execução paralela aumentando a velocidade de processamento de dados do R e pode ser utilizada tanto com multi-core (múltiplos processadores) quanto com nós de multi-cluster (múltiplos clusteres de computadores).

Para treinar a base (calcular as matrizes U e I):

Utilizou-se como dados de entrada do algoritmo o id do usuário, id do item comprado e o número de vezes que o item foi comprado pelo usuário.

Apesar do algoritmo exigir uma matriz $usuario \times item$, utilizou-se o armazenamento em tuplas (usuário, item, número de itens) e transformou-se em matriz somente aquelas necessárias para o cálculo de determinado valor, economizando-se assim espaço em memória.

Com a base de dados e os parâmetros λ e n_f fornecidos, inicializa-se as matrizes U e I como variáveis big.matrix de memória compartilhada e então, inicia-se um loop até que o critério de parada seja atingido.

Dentro do loop principal tem-se 3 loops que são executados sequencialmente para o cálculo da matriz U, matriz I e do erro quadrático médio nesta ordem. As iterações dentro desses 3 loops são executadas paralelamente por múltiplos processadores, sendo que todos processadores apontam para as mesmas matrizes (que são compartilhadas) e não há atualização simultânea das mesmas linhas, já que o pacote *foreach* e *doParallel* fazem a gestão dos processos simultâneos.

Para fazer as recomendações:

Recebe-se o id do usuário e o número de recomendações a serem devolvidas e a partir das matrizes U e I (após treinar a base, as matrizes ficam armazenadas em arquivos) calcula-se o rating desse usuário para todos os itens.

Após ordenar o rating do maior para o menor, retorna-se o número de recomendações solicitadas.

4.4 Fast Context-aware Recommendations with Factorization Machines - FM

Assim como no algoritmo anterior, foram utilizados os pacotes *data.table*, *bigmemory*, *big.algebra*, *foreach* e *doParallel*, mas além deles também foi utilizado o *Rcpp* e através do mesmo, criou-se uma nova função na linguagem C++ chamada *BigColSums*.

- Rcpp: é uma extensão do R que permite a integração com a linguagem C++, melhorando assim o desempenho de algumas funções e permitindo até conectar com projetos de C++;
- BigColSums: função criada para big.matrix, que permite realizar a soma dos elementos da coluna. O R já possui uma função para a soma de colunas de objetos nativos do R, mas como o big.matrix foi desenvolvido à parte e é escrito em C++, não é compatível com a função nativa e então faz-se necessário utilizar uma nova função.

Para treinar a base (calcular os parâmetros w_0, w e V):

Os seguintes dados foram utilizados como entrada do algoritmo:

- **User** (id_user, sexo, idade, estado geográfico);
- **Item** (id_item, categoria, cor);
- **Context** (proximidade da data com datas especiais, uso de cupom de desconto, recompra, likes em tags do facebook e agrupamento de dados de navegação com abandono de carrinho);
- **Rating** (quantidade de itens comprados);

e todos eles foram previamente modelados, mapeados e concatenados atendendo ao formato especificado na figura 3.2.

Mas ao invés de armazená-la como uma matriz única (algo em torno de $1.000.000 \times 400.000$), utilizou-se o armazenamento em tuplas e somente transformou-se em matriz as colunas exigidas no cálculo daquele parâmetro, economizando-se assim espaço em memória.

Através dessa base de dados de entrada, o número de fatores k e os parâmetros de regularização λ fornecidos, inicializa-se o processo de cálculo dos parâmetros.

As variáveis w_0, W e V são inicializadas como objetos big.matrix armazenados em arquivos, para que possam ser acessados por todas as threads que

efetuarem os cálculos em paralelo. Também são inicializadas as variáveis e e q como big.matrix e itera-se um loop que computa os valores iniciais de e e q em paralelo.

Com todas as variáveis iniciais computadas, itera-se um loop principal até que o critério de parada estabelecido seja atingido. Dentro desse loop principal são computados os valores de w_0, W e V sequencialmente e para cada execução é atualizado o valor de e que é o erro. Em cada iteração é executado um loop de cálculo dos valores de W e um outro loop para cada $V_i, i \in k$, sendo todos eles internamente paralelizados. Foi necessário efetuar o cálculo desta forma, pois os valores de V são dependentes de W e vice-versa, já que ambos utilizam valores da variável e.

Os pacotes *foreach*, *doParallel* e *bigmemory* garantem que todas as linhas serão computadas paralelamente e sem sobreposição de valores e nem atualização simultânea de campos iguais.

Para fazer as recomendações:

É realizado o mapeamento da entrada para que fiquem no formato específico do algoritmo e então a partir dos parâmetros w_0, W e V já calculados, retorna-se o valor de rating para aquele usuário, com aquele item, naquele contexto.

Verifica-se quais os itens com maior rating dado essas condições e então retorna o número de recomendações solicitadas.

Capítulo 5

Testes e Resultados

5.1 Testes realizados

Os testes realizados utilizaram dados de uma empresa de Comércio Eletrônico do setor de Móveis e Utilidades Domésticas e foram executados em um servidor Linux (Ubuntu Server 64bits, 24 processadores, 128GB RAM, 2 TB Disco).

O tempo de execução de cada um dos algoritmos para "treinar a base"(calcular as matrizes e os parâmetros) foram:

- CF: para $n_f = 100$, aproximadamente 10h
- FM: para $n_f = 5$ aproximadamente 40h

e a partir dos parâmetros calculados, foram realizadas recomendações de 6 items a aproximadamente 200.000 usuários para cada algoritmo. Estes foram recomendados através de Campanhas de E-mail Marketing.

Campanha de E-mail Marketing

Uma campanha de E-mail Marketing consiste normalmente em uma comunicação anunciando uma campanha do site da empresa (seleção de produtos que atendem a um critério específico. Exemplo: Campanha de Natal com produtos relacionados ao Natal).

Esta comunicação é feita através de uma chamada no e-mail com um banner que leva a página da Campanha dentro do site seguido de alguns produtos que ou são selecionados manualmente (produtos que estão em destaque, seja com preços especiais ou por serem novidades) ou utilizando algum tipo de inteligência para recomendação a cada cliente.

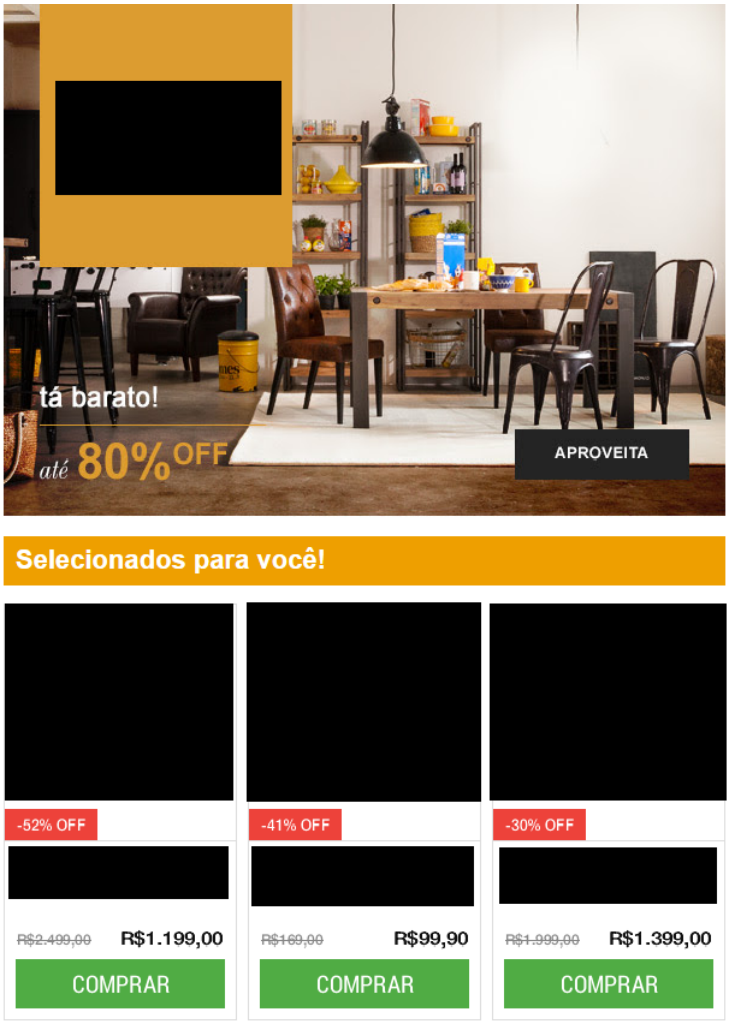

Figura 5.1: Exemplo de Campanha de E-mail Marketing com recomendações na seção Selecionados para você

As recomendações geradas pelos algoritmos deste trabalho foram enviadas através de campanhas de E-mail Marketing em 5 dias diferentes (29, 30 e 31 de outubro de 2014 e 5 e 6 de novembro de 2014), sendo que para cada envio de campanha de E-mail Marketing foram selecionados 22 grupos de aproximadamente 10.000 usuários, com 20 grupos escolhidos aleatoriamente, um grupo com usuários selecionados pelo algoritmo CF e outro grupo selecionado pelo algoritmo FM.

Não houve sobreposição de usuários nos mesmos algoritmos em campanhas diferentes, ou seja, um usuário não recebeu mais de uma campanha do mesmo algoritmo, mas pode ter recebido campanhas dos dois algoritmos.

5.2 Métricas utilizadas

Existem muitas métricas utilizadas no mercado para mensurar campanhas de E-mail Marketing sendo algumas delas:

- Quantidade de e-mails enviados (sent)
- Quantidade de e-mails abertos (open)
- Quantidade de e-mails clicados (click)
- Quantidade de pedidos de compra gerados (orders)
- Receita gerada (revenue)
- Taxa de abertura (open rate): $\frac{e-mails\,abertos}{e-mails\,enviados}$
- Taxa de cliques (CTR): $\frac{e-mails\,clicados}{e-mails\,enviados}$
- Cliques Sobre Aberturas (click-to-open): $\frac{e-mails\,clicados}{e-mails\,abertos}$
- Taxa de Conversão: $\frac{nmero\,de\,compras}{e-mails\,clicados}$
- Receita por E-mail: $\frac{receita\,gerada}{e-mails\,enviados}$

Como o grande diferencial da recomendação são os produtos exibidos no corpo do e-mail, foram consideradas somente as métricas de *Taxa de cliques (CTR)*, *Cliques Sobre Aberturas (Click/Open)* e *Receita por E-mail (Rev/Email)* como as relevantes para análise, por estarem diretamente relacionadas aos produtos exibidos.

5.3 Resultados Obtidos

Devido a sigilosidade dos dados, todos os números estão expressos como aumento ou queda dos Algoritmos em relação à base de usuários Normal (sem o uso de algoritmo). Para os casos de *CTR* e *Click/Open*, o cálculo se deu pela divisão dos dados dos Algoritmos pelo envio sem Algoritmo e no caso de *Rev/Email* se deu pela subtração dos dados dos Algoritmos pelo sem Algoritmo.

A seguir estão os resultados dos envios diários e o resultado consolidado (final):

Tabela 5.1: Resultado - Diário

Data	Base	CTR	Click/Open	Rev/Email (R$)
29/10/2014	Normal	0%	0%	0
	FM	1.94%	6.83%	-0.06
	CF	15.92%	14.59%	0.04
30/10/2014	Normal	0%	0%	0
	FM	5.68%	5.58%	-0.02
	CF	8.54%	6.75%	-0.03
31/10/2014	Normal	0%	0%	0
	FM	4.04%	11.91%	0.04
	CF	6.65%	9.35%	0.02
05/11/2014	Normal	0%	0%	0
	FM	24.51%	22.25%	0.08
	CF	31.27%	25.17%	0.12
06/11/2014	Normal	0%	0%	0
	FM	18.39%	14.28%	0.05
	CF	16.85%	25.84%	0.09

Tabela 5.2: Resultado - Consolidado

Base	CTR	Click/Open	Rev/Email (R$)
Normal	0%	0%	0
FM	8.39%	9.67%	0.01
CF	13.65%	14.06%	0.04

5.4 Análise Individual das Campanhas

A seguir uma análise detalhada de cada Campanha de E-mail Marketing com sua respectiva imagem (por questões de sigilo não foram exibidos os produtos recomendados), comparando os resultados obtidos pelos 3 grupos de usuários.

Os 3 grupos de usuários receberam as mesmas campanha nos respectivos dias, apenas com a diferença na seção de "Produtos Selecionados". A base dos algoritmos recebeu os produtos selecionados por cada algoritmo e a base normal recebeu produtos selecionados conforme melhor oferta e melhor destaque.

5.4.1 Campanha do dia 29/10/2014

Figura 5.2: Campanha de E-mail Marketing do dia 29/10/2014

Envio realizado em uma quarta-feira a uma base de 249.266 pessoas, sendo 9.873 à base FM, 9.913 à base CF e 229.480 a base normal.

Por ser final de mês e pela necessidade de atingir a meta de vendas da empresa, havia uma campanha muito forte no site com descontos bem agressivos (de até 80%) em diversas categorias de produtos como pode ser visto na figura 5.2. Isto fez com que o resultado financeiro não fosse expressivo (pior no caso do FM) em relação à base normal, já que com uma campanha bem atrativa mais produtos são vendidos. Desta forma a base normal teve um bom desempenho que fez com que não houvesse ganho financeiro utilizando os algoritmos.

Em relação a *CTR* e *Click/Open*, as bases dos algoritmos apresentaram melhora em comparação a base normal e o algoritmo *CF* teve um desempenho muito bom.

5.4.2 Campanha do dia 30/10/2014

Figura 5.3: Campanha de E-mail Marketing do dia 30/10/2014

Envio realizado em uma quinta-feira a uma base de 244.056 pessoas, sendo 9.941 à base FM, 9.932 à base CF e 224.183 a base normal.

Ainda estava em vigor a campanha de final do mês que também foi utilizada no dia anterior (29/10/2014) e por isso os algoritmos tiveram um desem-

penho um pouco melhor que a base normal em relação a *CTR* e *Click/Open* mas isso não resultou em ganhos financeiros.

5.4.3 Campanha do dia 31/10/2014

Figura 5.4: Campanha de E-mail Marketing do dia 31/10/2014

Envio realizado em uma sexta-feira a uma base de 231.306 pessoas, sendo 9.893 à base FM, 9.910 à base CF e 211.503 a base normal.

Apesar de a campanha de final do mês ainda estar em vigor, o desempenho dos algoritmos foi melhor que os dias anteriores, mantendo um ganho em *CTR* e *Click/Open*, mas desta vez impactando também financeiramente. Isso se deve ao fato de uma mesma campanha do site não trazer os mesmos retornos durante um longo período (os primeiros dias são os que desempenham melhor) e isso se reflete também no E-mail Marketing.

5.4.4 Campanha do dia 05/11/2014

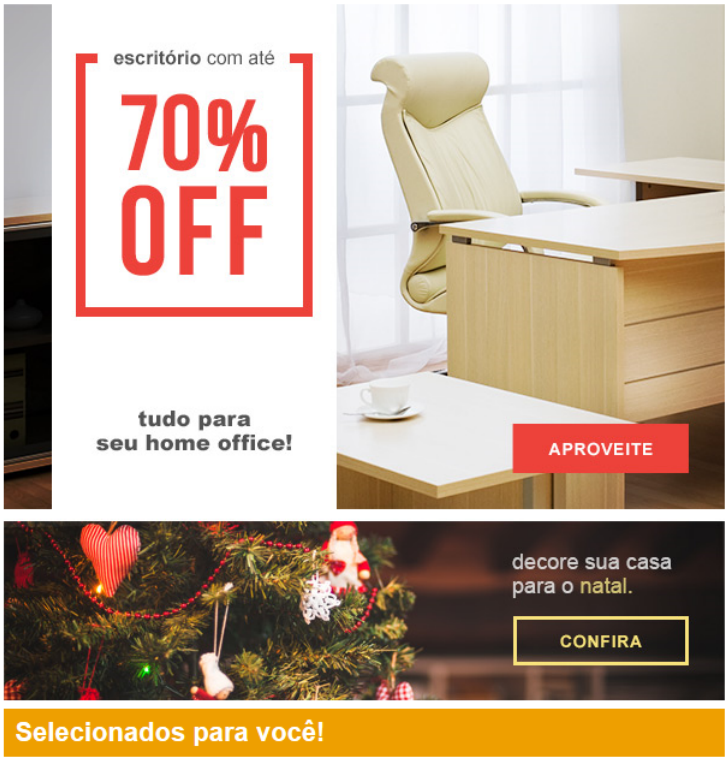

Figura 5.5: Campanha de E-mail Marketing do dia 05/11/2014

Envio realizado em uma quarta-feira a uma base de 243.658 pessoas, sendo 9.907 à base FM, 9.890 à base CF e 223.861 a base normal.

Com a virada do mês e nova meta mensal de vendas, deixa de existir uma campanha forte no site e com isso os algoritmos ganham muito mais força. Apesar da campanha geral do site ter 70% de desconto, esta é só para determinada categoria de produtos e então o desempenho não foi tão bom quanto nos outros dias.

Os algoritmos superaram e muito a base normal tanto em *CTR* e *Click/Open* quanto em receita *Rev/Email*, sendo que o *CF* desempenhou melhor.

5.4.5 Campanha do dia 06/11/2014

Figura 5.6: Campanha de E-mail Marketing do dia 06/11/2014

Envio realizado em uma quinta-feira a uma base de 206.131 pessoas, sendo 9.903 à base FM, 9.853 à base CF e 186.375 a base normal.

Os algoritmos novamente tiveram um desempenho muito superior em relação a base normal, com ganhos de 15 a 25% em *CTR* e *Click/Open* e até R$0.09 por e-mail enviado. Isso representa um maior acesso aos produtos ofertados e consequentemente maior acesso ao site que resultou em aumento nas vendas.

Capítulo 6

Conclusão

6.1 Considerações Finais

Após análise dos resultados obtidos nos testes chegou-se a conclusão de que os sistemas de recomendação através do uso de algoritmos de recomendação melhoraram tanto o fluxo de usuários como a receita proveniente de campanhas de E-mail Marketing.

Os maiores ganhos nos resultados foram observados em dias que não havia uma campanha forte no site. Como através das recomendações oferta-se produtos pontuais que o cliente pode se interessar, a utilização de recomendação pode ser um grande diferencial em dias sem grandes campanhas.

A melhora foi mais acentuada para o *Large-scale Parallel Collaborative Filtering - CF* e isso pode ser explicado pelo fato deste algoritmo utilizar muito mais características implícitas do problema (features) do que o *Fast Context-aware Recommendations with Factorization Machines - FM*. Os algoritmos que utilizam contexto ainda são computacionalmente ineficientes comparados aos algoritmos convencionais e por isso ainda deixam a desejar em tempo de execução e consequentemente na qualidade das recomendações para o Comércio Eletrônico.

Também pesa negativamente a escolha dos Contextos para o *FM*, pois a escolha e modelagem dos mesmos são muito trabalhosos e custosos sendo que nem sempre trazem o resultado esperado.

Então, para uso no Comércio Eletrônico, os Sistemas de Recomendações tradicionais continuam sendo mais interessantes e o algoritmo *CF* é uma boa alternativa de implementação fácil e eficiente. Mas os sistemas que utilizam o Contexto não podem ser descartados, necessitando de uma boa modelagem de dados com um algoritmo muito eficiente e que permita incorporar muitas características.

6.2 Trabalhos Futuros

Para a continuidade do trabalho seria interessante incorporar dados de navegação e abandono de carrinho ao *CF* para abranger mais usuários e possivelmente melhorar a recomendação.

Também seria válido testar novos tipos de contexto, aumentando o número de características e buscar novos tipos de algoritmos de Context Recommendation que sejam mais eficientes.

Outra possível melhora seria testar abordagens que unam os dois conceitos de Context e Collaborative, seja através de uma modelagem de dados que incorpore dados de Context ao Collaborative, seja pela unificação dos dois métodos.

Capítulo 7

Parte Subjetiva

Por trabalhar em uma empresa de Comércio Eletrônico (E-commerce), estou diariamente em contato com diferentes formas de engajar o cliente e uma das principais formas são as recomendações.

Tive o primeiro contato com os Sistemas de Recomendação no final de 2013 ao implementar uma recomendação bem simples e que trouxe um retorno financeiro muito satisfatório e então me interessei pelo assunto até que decidi como o tema de meu TCC.

7.1 Desafios e Frustrações

O maior desafio foi conciliar o projeto do TCC com o trabalho. Como não tinha mais nenhuma disciplina para fazer neste ano achei que conseguiria dedicar boa parte do meu tempo ao desenvolvimento do projeto e ele até teve um bom andamento até meados de junho, época em que fui promovido no trabalho. Desde então o trabalho consumiu grande parte do meu tempo e mesmo que tentasse não conseguia focar 100% no TCC nas horas livres por estar muito cansado e não querer "pensar em nada". Só voltei a me dedicar ao TCC na metade de setembro e por isso muitas coisas acabaram ficando em cima da hora, diferente do que foi previsto.

Um problema encontrado durante o desenvolvimento do projeto foi na implementação dos algoritmos. Já havia implementado os algoritmos antes de maio e por isso estava bem tranquilo quanto a isso, mas quando terminei o tratamento dos dados e testei, vi que o tempo de execução estava extremamente alto e então tive que refazer os códigos utilizando uma abordagem muito mais eficiente.

Fiquei frustrado por não conseguir usar muito os dados do Facebook, por não ter a implementação do Facebook Connect na empresa e dessa forma

não conseguir cruzar corretamente os dados do Facebook com os da empresa. Foram poucos dados comparado aos outros utilizados e por isso acredito que não foram muito relevantes.

Também me deixou frustrado não conseguir implementar um algoritmo Content-based, mas pelo tempo hábil acho que não seria possível.

7.2 Disciplinas Relevantes para o Trabalho

A seguir a lista de disciplinas que foram importantes para a realização do trabalho:

- MAE0121 e MAE0212 - Introdução a Probabilidade e a Estatística I e II: estas disciplinas me deram a base estatística para entender conceitos utilizados nos algoritmos e principalmente nas áreas de Mineração de Dados e Aprendizagem Computacional. Também foram nestas disciplinas que tive o primeiro contato com a linguagem R.

- MAC0426 - Sistemas de Banco de Dados: foi onde tive o primeiro contato com Bancos de Dados e hoje é a minha área de trabalho. Os conhecimentos adquiridos nesta disciplina me ajudaram no cruzamento e tratamento dos dados e também na extração dos dados da empresa.

- MAC0323 - Estrutura de Dados: foi a disciplina em que realmente aprendi a programar e foram ensinadas as primeiras estruturas de dados.

- MAC0300 - Métodos Numéricos da Álgebra Linear: nunca imaginei que fosse utilizar esta disciplina para o meu TCC, mas foi muito importante no entendimento e implementação dos algoritmos, já que todas os dados foram utilizados em forma matricial.

- MAC0438 - Programação Concorrente: foi a disciplina em que tive contato com concorrência e um pouco com paralelismo, o que me facilitou no trabalho.

- CRP0297 - Análise e Planejamento Mercadológico: disciplina da ECA-USP, que me ajudou muito com conceitos de marketing e análise de mercado e algumas métricas aprendidas foram utilizadas no trabalho.

7.3 Agradecimentos

Agradeço primeiramente ao professor Roberto Hirata por aceitar ser meu orientador mesmo sendo um trabalho de uma área desconhecida e que foi sempre muito solícito e acreditou e confiou em mim. Cheguei a achar que meu trabalho estava perdido em um momento, mas com sua orientação consegui chegar à etapa final. Com certeza não teria pessoa melhor para ser meu orientador.

Agradeço aos meus pais e irmãos por me apoiarem e serem tão pacientes, além de entenderem meu período de ausência nos afazeres diários e também nos eventos familiares.

Agradeço de coração a minha namorada Karina que me apoiou e incentivou num momento tão difícil e atribulado de minha vida e mesmo não tendo sido o melhor dos namorados, ainda assim não desistiu de mim. Amor, te amo muito!

Ao meu gestor de trabalho Ricardo Bechara e colegas de trabalho Maurício, Felipe, João, Fernando, Ricardo e Gabriel, que entenderam meu momento e cobriram minhas ausências em alguns dias de trabalho e ainda me aturaram em dias de mau humor devido a noites mal dormidas.

Aos meus amigos de IME Lucas, Paulo, Luciano, Nelson, Alexandre, Alex, Nobu, Marcão, Bode, Edsão, Lafond, Jéssika, Steven, Yin, Karina, Dani e outros que não me lembro, que além das aulas juntos e me ajudarem nas matérias, tornaram meus dias de IME tão divertidos.

E a todos os funcionários e professores que tive contato durante esses anos e principalmente ao IME, muito obrigado! Posso não ter gostado de algumas coisas, mas valorizo do fundo do coração cada experiência que tive nesta instituição. Hoje sinto que sou uma pessoa melhor e mais preparada para a vida.

Referências Bibliográficas

[1] Amazon's recommendation secret. http://fortune.com/2012/07/30/amazons-recommendation-secret/. [Online; acessado em 01-outubro-2014].

[2] Content-based filtering. http://findoutyourfavorite.blogspot.com.br/2012/04/content-based-filtering.html.

[3] Data science: Recommender engines with collaborative filtering. http://blog.operasolutions.com/bid/387700/Data-Science-Recommender-Engines-with-Collaborative-Filtering.

[4] How big is facebook's data? 2.5 billion pieces of content and 500+ terabytes ingested every day. http://tcrn.ch/1nKWiHM. [Online; acessado em 28-outubro-2014].

[5] Last.fm. http://en.wikipedia.org/wiki/Last.fm. [Online; acessado em 24-outubro-2014].

[6] Netflix may only give you 3 or 4 recommendations in the future. http://www.huffingtonpost.com/2014/05/20/netflix-recommendations_n_5357619.html. [Online; acessado em 24-outubro-2014].

[7] Taking r to the limit, part i - parallelization in r. http://www.bytemining.com/2010/07/taking-r-to-the-limit-part-i-parallelization-in-r/. [Online; acessado em 06-novembro-2014].

[8] Taking r to the limit, part ii - large datasets in r. http://www.bytemining.com/2010/08/taking-r-to-the-limit-part-ii-large-datasets-in-r/. [Online; acessado em 06-novembro-2014].

[9] Tf-idf. http://www.tfidf.com/. [Online; acessado em 30-outubro-2014].

[10] Tikhonov regularization and total least squares. http://drum.lib.umd.edu/bitstream/1903/913/2/CS-TR-3829.pdf.

[11] SNÁSEL Vaclav ABRAHAM Ajith, HASSANIEN Aboul-Ella. *Computational Social Network Analysis: Trends, Tools and Research Advances*. Springer, 2009.

[12] TAN Ah-hwee. Text mining: The state of the art and the challenges. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.132.6973. [Online; acessado em 26-novembro-2014].

[13] POZZER Cezar. Aprendizado por Árvores de decisão. http://www-usr.inf.ufsm.br/~pozzer/disciplinas/pj3d_decisionTrees.pdf, 2006.

[14] KANE Michael J EMERSON John W. The r package bigmemory: Supporting efficient computation and concurrent programming with large data sets. http://www.stat.yale.edu/~mjk56/temp/bigmemory-vignette.pdf. [Online; acessado em 07-novembro-2014].

[15] ADOMAVICIUS Gediminas and ALEXANDER Tuzhilin. Context-aware recommender systems. http://ids.csom.umn.edu/faculty/gedas/nsfcareer/CARS-chapter-2010.pdf. [Online; acessado em 08-agosto-2014].

[16] QUEIROZ Bruno GUN Murilo. *Estratégias de E-mail Marketing - Como obter resultados através do marketing direto na Internet*. Brasport, 2008.

[17] FRIEDMAN Jerome HASTIE Trevor, TIBSHIRANI Robert. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

[18] TERVEEN Loren RIEDL John HERLOCKER Jonathan, KONSTAN Joseph. Evaluating collaborative filtering recommender systems. http://dl.acm.org/citation.cfm?id=963772, 2004.

[19] PRAHALAD C. K. Beyond crm: Ck prahalad predicts customer context is the next big thing. American Management Association MwWorld, 2004.

[20] OLIVER Nuria KARATZOGLOU Alexandros, AMATRIAIN Xavier and BALTRUNAS Linas. Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering. http://www.ci.tuwien.ac.at/~alexis/Publications_files/tensorcofi-recsys10.pdf. [Online; acessado em 08-agosto-2014].

[21] BAILEY Kenneth. *Numerical Taxonomy and Cluster Analysis*. SAGE, 1994.

[22] YORK Jeremy LINDEN Greg, SMITH Brent. Amazon.com recommendations item-to-item collaborative filtering. http://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf. [Online; acessado em 08-agosto-2014].

[23] DE GEMMIS Marco LOPS Pasquale and SEMERARO Giovanni. Content-based recommender systems: State of the art and trends. http://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/handbook/ContentBasedRS.pdf. [Online; acessado em 08-agosto-2014].

[24] SINDHWANI Vikas MELVILLE Prem. Recommender systems. http://www.prem-melville.com/publications/recommender-systems-eml2010.pdf. [Online; acessado em 01-fevereiro-2015].

[25] POLLACK Michael. Mobile funnel optimization 101: Quantify, manage, and reduce user drop-off. https://www.vessel.io/mobile-funnel-optimization/.

[26] SHAPIRA Bracha KANTOR Paul B. RICCI Francesco, ROKACH Lior. *Recommender Systems Handbook*. Springer, 2011.

[27] BURKE Robin. Hybrid recommender systems: Survey and experiments. http://josquin.cs.depaul.edu/~rburke/pubs/burke-umuai02.pdf.

[28] MAIMON Oded ROKACH Lior. *Data mining with decision trees: theory and applications*. World Scientific Pub Co Inc, 2008.

[29] FALTIN Frederick RUGGERI Fabrizio, KENNETT Ron. Bayesian networks. http://www.eng.tau.ac.il/~bengal/BN.pdf, 2007.

[30] NORVIG Peter RUSSELL Stuart. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.

[31] WILKINSON Dennis SCHREIBER Robert, ZHOU Yunhong and PAN Rong. Large-scale parallel collaborative filtering for the netflix prize. http://www.hpl.hp.com/personal/Robert_Schreiber/papers/2008%20AAIM%20Netflix/netflix_aaim08(submitted).pdf. [Online; acessado em 08-agosto-2014].

[32] RENDLE Steffen. Fast context-aware recommendations with factorization machines. `http://www.ismll.uni-hildesheim.de/pub/pdfs/Rendle_et_al2011-Context_Aware.pdf`. [Online; acessado em 08-agosto-2014].

[33] RENDLE Steffen. Factorization machines. Proceedings of the 10th IEEE International Conference on Data Mining, 2010.

[34] MITCHELL Tom. *Machine Learning*. McGraw Hill, 1997.