LEARNING TO RANK PARA BUSCA EM COMÉRCIO ELETRÔNICO

ROBERTO CIDADE FONSECA

LEARNING TO RANK PARA BUSCA EM COMÉRCIO ELETRÔNICO

Dissertação apresentada ao Programa de Pós-Graduação em Informáticado Instituto de Computação da Universidade Federal do Amazonas como requisito parcial para a obtenção do grau de Mestre em Informática.

Orientador: Edleno Silva de Moura

Manaus

Agosto de 2018

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Fonseca, Roberto Cidade
F676l Learning to Rank para Busca em Comércio Eletrônico / Roberto Cidade Fonseca. 2018
44 f.: il. color; 31 cm.

Orientador: Edleno Silva de Moura
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Learning to Rank. 2. Machine Learning. 3. Recuperação de Informação. 4. Comércio Eletrônico. 5. Teste A/B. I. Moura, Edleno Silva de II. Universidade Federal do Amazonas III. Título

PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
INSTITUTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

UFAM

FOLHA DE APROVAÇÃO

"Learning to Rank para Busca em Comércio Eletrônico"

ROBERTO CIDADE FONSECA

Dissertação de Mestrado defendida e aprovada pela banca examinadora constituída pelos Professores:

Prof. Edleno Silva de Moura - PRESIDENTE

Prof. Marco Antonio Pinheiro de Cristo - MEMBRO INTERNO

Prof. Rafael Giusti - MEMBRO EXTERNO

Prof. Thierson Couto Rosa - MEMBRO EXTERNO

Manaus, 28 de Agosto de 2018

À minha mãe, que sempre acreditou na importância de uma boa educação.

Agradecimentos

Agradeço à Deus, porque sem Ele nada é possível.

À minha esposa Myrian Rodrigues da Silva, por todo amor e apoio que me deu todos esses anos.

Ao meu orientador Edleno, por ter me guiado e deixado todo o processo mais simples, com sugestões e dicas importantes.

"A = A"
(John Galt)

Resumo

Métodos que geram funções de ordenação de resultados baseadas em aprendizagem de máquina têm sido amplamente utilizados em sistemas de busca para a web, como as utilizadas em motores de busca como o Google e Bing. No entanto, esses recursos não têm sido muito empregados ou estudados em outros contextos. É o caso, por exemplo, do comércio eletrônico, no qual, a interação de usuários com lojas virtuais produz dados como: quando um usuário acessou a página de uma loja pela primeira vez, que consultas realizou, quais produtos clicou, e o que comprou. Neste trabalho, propomos a utilização de métodos de aprendizagem de máquina para aprender funções de ordenação de resultados no contexto de comércio eletrônico. Estudamos formas alternativas de estimar a relevância de um resultado para uma dada consulta e realizamos experimentos utilizando dados extraídos de lojas de comércio eletrônico. Realizamos experimentos tanto com ambientes que denominamos *offline*, onde uma base de dados é montada com a abordagem tradicional de separa-la em treino, validação e teste, quanto em ambientes que denominamos *online*, onde pusemos versões distintas dos sistemas para funcionar em lojas com usuários em situações reais de compra. Apresentamos no estudo nossas conclusões a respeito dos experimentos realizados.

Abstract

Machine learning (ML) based ranking functions generating methods have been broadly used on web search systems, such as the utilized by Google and Bing. Nonetheless, such methods have not been employed or studied in other contexts. It is the case, to cite an example, of electronic commerce (e-commerce), on which the user interaction with virtual stores produces data as: when an user landed on a page for the first time, queries submitted, products clicked and what she bought. In this work, we propose to leverage ML to learn ranking functions for the e-commerce context. We studied alternatives to estimate the relevance of a result for a given query and deployed experiments using data mined from e-commerce shops. We ran experiments in setups we denominated *offline*, where a dataset was created the traditional way by separating it in three subsets of training, validation and test, as well as in setups we denominated *online*, where distinct versions of the system were deployed to shops facing users in a real purchase situation. We present in the study our conclusions regarding the performed experiments.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Lojas de comércio eletrônico têm tipicamente, centenas, milhares ou até milhões de produtos em seus catálogos. Cada loja tem uma coleção de dados que precisa ser organizada e sobre a qual é necessário realizar operações de pesquisa. E isso vai além de permitir ao dono da loja uma fácil organização de seus produtos para fins de controle de estoque, mas, permitir que seus clientes procurem, comparem e comprem seus produtos. As lojas e marcas expõem seus catálogos a seus clientes de forma instantânea através de suas páginas web ou por meio de aplicativos para celular. A todo momento seus catálogos estão sendo vistos por milhares de usuários, cada um com objetivos diferentes: pesquisa de preço, compra, comparação entre produtos similares.

Esse tipo de interação tem crescido a cada ano. O comércio eletrônico é um dos tipos de varejo que mais cresce mundialmente. Nos Estados Unidos isso levou ao esvaziamento de *shopping centers*. Marcas e lojistas substituem catálogos físicos de venda por suas versões virtuais disponíveis em seus sites. É possível encomendar quase tudo pela internet. Marcas pequenas compram espaço em grandes varejistas eletrônicos através de *Marketplaces*, ganhando presença online a um custo reduzido.

Por conta da crescente demanda de espaço no varejo online, cresce também a necessidade de ferramentas capazes de ajudar seus clientes a encontrar o que procuram, em especial as ferramentas expostas aos consumidores finais.

Cada loja de comércio eletrônico é formada por diversos componentes que atuam juntos, para facilitar e melhorar a experiência de seus clientes. Dentre esses componentes se encontra a busca. Ela permite que os consumidores realizem consultas utilizando palavras que tentam traduzir aquilo que procuram. A busca então mostra um conjunto de produtos que julgou atender à necessidade do usuário, permitindo-o navegar e explorar um subconjunto do catálogo de produtos de uma loja ou marca. Ao contrário de quando fazemos uma consulta na web, onde a experiência normalmente é compa-

rada à de um usuário de biblioteca que tem uma necessidade de informação e procura documentos para satisfazer tal necessidade, a experiência de busca em catálogos de produtos do varejo eletrônico se assemelha àquela de paginar um catálogo de compras, pulando de vez em quando entre as páginas para tentar encontrar os produtos que mais lhe interessam.

Essa busca deve ser rápida, pois a maioria dos usuários perde a paciência com mais de alguns segundos de carregamento; e inteligente, de forma a ordenar os produtos de acordo com o perfil de cada usuário a cada momento, e de acordo com as características de cada produto. Possuir ou não essas características faz a diferença entre uma boa experiência de compra e a frustração e consequente abandono da página da loja pelo usuário.

A busca é formada por diversos componentes. Dentre eles está o que chamamos de *função de ordenação de resultados*, ou *função de ordenação*, que pontua os produtos no resultado de uma consulta. Ao fazê-lo, a função tenta traduzir aquilo que é relevante para o usuário para a ordem em que os produtos são apresentados, deixando aqueles com maior pontuação no topo. Mas afinal, o que é relevante para um usuário em particular? O que ele está procurando quando faz uma consulta? E como podemos ordenar os produtos maximizando a chance de compra do produto que ele procura, ou que ainda não sabe que procura?

A migração do catálogo das lojas para um ambiente virtual público permite aos consumidores explorar os produtos de uma loja e deixar seus interesses implícitos no padrão com que interagem com a página do lojista. Durante a navegação pelo catálogo de uma loja, os usuários escrevem consultas, clicam em produtos desejados, adicionam produtos ao carrinho, removem produtos do carrinho, navegam pelo resultado de sua busca várias vezes, filtram o resultado, gastam mais ou menos tempo antes de tomarem diferentes ações, e finalmente, compram. O fato de termos esses registros possibilita a extração de dados que podem ser utilizados para a otimizar a ordem dos resultados.

Uma das maneiras de lidarmos com esse problema é utilizar técnicas de aprendizagem de máquina para estimarmos a relevância do produto para um usuário dada uma consulta, usando exemplos criados a partir da mineração dos registros de acesso às lojas. Na literatura, a área que abrange técnicas que aprendem funções de ordenação, é conhecida como *learning to rank*.

1.1 Problema

Este trabalho aborda o problema de desenvolver funções de ordenação para o varejo eletrônico, levando em conta o viés posicional, que consiste na tendência de usuários clicarem em resultados nas primeiras posições [13].

O problema de aprendizado de funções de ordenação é enfrentado aplicando métodos já presentes na literatura, mas com o diferencial de se utilizar essas técnicas no comércio eletrônico, em um sistema comercial real através de testes A/B. Utilizou-se *features* entre as consultas realizadas em lojas do varejo eletrônico, extraídas de dados comportamentais de compradores. Em seguida, para treinar modelos definiu-se conceitos de relevância, utilizados como objetivo de otimização dos algoritmos de aprendizagem de máquina, sendo necessário experimentar diversas definições e escolher a mais útil aos propósitos dos varejistas eletrônicos. Possibilitando assim substituir modelos configurados manualmente, com suas desvantagens inerentes, como: não escalabilidade e difícil configuração.

1.2 Contribuições

Aqui resumimos as contribuições desta dissertação.

- Reportamos o primeiro resultado em teste A/B do uso de técnicas de *learning to rank* para a ordenação de resultados de busca em catálogos de produtos em lojas de comércio eletrônico. Demonstrando o possível impacto financeiro real que o uso dessas abordagens pode acarretar.
- Comparamos o desempenho de diferentes modelos utilizando diversas definições de relevância, buscando alternativas ao paradigma de Cranfield e ao uso direto de *feedback* implícito com o uso de cliques para tal.
- Comparação entre funções aprendidas dos dados comportamentais de usuários de varejo eletrônico e uma função manualmente otimizada por um time de especialistas de domínio em varejo eletrônico.
- Reportamos o desempenho do uso de um modelo de clique para tentar sanar o problema do viés posicional criando definições de relevância que o levam em conta.

Esta dissertação está dividida em 5 capítulos. O primeiro capítulo trata da motivação deste trabalho e resume as contribuições desta dissertação. O segundo capítulo

introduz conceitos necessários à sua compreensão e discute os trabalhos relacionados. No terceiro capítulo entra-se em detalhe nos experimentos *offline* realizados, discutindo-se os resultados ao aprender funções de ordenação com diferentes métodos de *learning to rank* e diversas definições de relevância, comparando-as. O quarto descreve os testes A/B, sua duração, domínio da loja utilizada e seus resultados. O quinto capítulo trata de próximos passos e sugestões para trabalhos futuros.

Capítulo 2

Conceitos Básicos e Trabalhos Relacionados

2.1 Recuperação de Informação

Um sistema de RI permite que seus usuários busquem informações relevantes em bases de dados. A interação entre usuários e sistemas de recuperação é iniciada pelo usuário com a finalidade de satisfazer alguma (mais ou menos explícita) necessidade de informação [1]. O usuário expressa essa necessidade de informação numa consulta (e.g., como uma sequência de palavras-chave, mas outras formas também são possíveis) e a submete ao sistema. Baseado na consulta, a tarefa do sistema é selecionar informações que tem maior chance de serem relevantes para a necessidade do usuário (o conceito de relevância é central para RI, mas difícil de definir; discute-se esse conceito em detalhe até o final desta seção). Mais frequentemente isso toma a forma de seleção de documentos do catálogo do sistema. O resultado apresentado é tipicamente na forma de uma lista ordenada pela probabilidade do documento atender à necessidade de informação do usuário [33].

Um dos principais focos da pesquisa em RI é o desenvolvimento de modelos de recuperação que capturem a relação entre uma consulta e um documento. Um dos primeiros modelos a se destacar foi o modelo vetorial (VSM de *vector space model*, em inglês), no qual consultas e documentos são representados como vetores em algum espaço de termos [34]. Ao invés de retornar o conjunto de documentos que casassem com a consulta, o sistema de recuperação baseado no VSM retornava listas ordenadas de documentos pela sua similaridade à consulta do usuário no espaço vetorial. O conceito de listas ordenadas de documentos recebeu maior detalhamento e forma-

lização pelo princípio da probabilidade de ordenação de resultados, que afirma que o desempenho da recuperação é otimizado quando sistemas ordenam seus documentos pela probabilidade da sua relevância [33] (assumindo um usuário e independência entre os documentos). Consequentemente, abordagens probabilísticas para RI foram desenvolvidas, com o BM25 sendo uma de suas variantes mais famosas [39]. As mais recentes e principais abordagens desenvolvidas se baseiam em modelos estatísticos de linguagem, onde documentos são modelados como sequências de palavras retiradas de uma distribuição subjacente. Assim, a pontuação de um documento para uma consulta é implementada como a estimativa da probabilidade de que uma consulta e um documento sejam amostras de uma mesma distribuição [19, 32].

Recentemente, uma ampla variedade de novos fatores contextuais, que serão chamadas a partir de agora de *features*, têm sido levados em consideração e integrados à modelos de recuperação. Por exemplo, termos extraídos de consultas passadas do usuário e páginas anteriormente visitadas podem capturar os interesses gerais e repertório cognitivo de um usuário [30, 37]. Similar a isso, detecção dos objetivos da busca e a intenção do usuário podem ser usados para melhorar o desempenho da recuperação [4, 41]. Num estudo de busca por especialistas, constatou-se que o uso de *features* contextuais, dependentes da tarefa sendo executada, como a mídia sendo usada, a estrutura organizacional e a posição de um especialista dentro da organização, poderiam melhorar o desempenho do sistema de recuperação [21, 20]. Finalmente, o uso da localização [2] e a informação temporal [3] mostrou melhorar os resultados de busca em buscas web e por notícias, por exemplo.

Novos modelos de RI são avaliados seguindo a tradição de pesquisa empírica da área. O paradigma de Cranfield, que é uma das peças fundamentais da *Text Retrieval Conference* (TREC - o maior esforço de padronização de avaliação em RI) [42], permitiu o rápido progresso na área. Esta configuração concentra-se nos elementos básicos de avaliação em RI. Dado um par documento-consulta, marcadores especialistas (também chamados de juízes de relevância) devem julgar manualmente, i.e., marcar se ou quão relevante o documento é considerado para uma dada consulta [43].

A partir dessa marcação é possível comparar funções de ordenação. Dentre as várias métricas desenvolvidas na área, a mais comumente utilizada é o NDCG e suas variações. Na sua forma mais geral essa métrica é dada pela fórmula abaixo:

$$NDCG = \sum \frac{2^{rel(l[i])} - 1}{log_2 i + 1} iDCG^{-1}$$

Dado um resultado ordenado de tamanho l, com i sendo a posição de algum documento neste resultado, esta métrica soma o ganho baseado na relevância atribuída

($rel(l[i])$) a cada documento (seja por um juiz ou *feedback*), e a divide por um fator de desconto (baseado no log da posição em que o documento foi apresentado). Este fator é o Ganho Cumulativo Descontado (DCG, do inglês - Discounted Cumulative Gain). O DCG é então divido pelo DCG ideal ($iDCG$), ou seja, o DCG obtido dos documentos em ordem decrescente de sua relevância real conhecida. Temos assim o NDCG, o DCG normalizado pelo DCG ideal. A métrica é descrita com mais detalhes por Jarvelin et al. [22].

Posto que a abundância de informações presentes facilita a extração de dados que podem ser úteis para capturar relações entre consultas, documentos e usuários. A combinação desses dados em *features*, de modo a permitir a captura de padrões de interesse, depende da sua adequada combinação, uma tarefa cuja dificuldade é proporcional à quantidade de dados. Uma solução é derivar automaticamente as funções de ordenação; o foco de estudo da área de *learning to rank* (L2R) para RI.

2.2 *Learning to rank*

Sistemas de busca modernos levam em consideração centenas de *features* de ordenação. Para enfrentar o problema de combinar o grande conjunto de *features* de tais sistemas, métodos de L2R foram desenvolvidos. Esses métodos utilizam técnicas de aprendizagem de máquina para combinar tais *features* automaticamente. L2R é uma área de pesquisa ativa e muitas abordagens foram propostas e refinadas nos últimos anos [27].

Assumi-se para os propósitos deste trabalho que abordagens de L2R para RI requerem uma representação baseada em *features*, onde vetores de *features* codificam características de uma consulta, um documento e a relação entre a consulta e o documento. Tais representações baseadas em *features* permitem generalizações entre documentos e consultas. O objetivo do algoritmo de aprendizagem é, então, encontrar padrões gerais e combinar *features* para melhorar o resultado das consultas (e.g., de acordo com alguma métrica de avaliação). Isso leva à seguinte formulação do problema de aprendizado supervisionado. A entrada fornecida são amostras na forma $(\boldsymbol{x}, y, q)$. Aqui, $\boldsymbol{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ é um vetor de *features* n-dimensional, que representa a relação entre um documento e uma consulta; y denota o gabarito, a verdadeira relevância do documento para a consulta; Finalmente, $q \in \mathbb{Z}$ indica a qual consulta a amostra pertence. O gabarito y pode ser obtido de um marcador treinado no domínio dos documentos, mas também pode ser inferido através do comportamento de usuários.

A grande maioria das abordagens de L2R para RI são desenvolvidas para métodos supervisionados. Nesses métodos assume-se dados de treino na forma de uma amostra

representativa de consultas, documentos e julgamentos de relevância. O formato e significado da entrada (vetores de*features*) e saída (rótulos de relevância) se modificam entre as abordagens supervisionadas. De acordo com Cao et al. [7], três abordagens se destacam, a saber, *pointwise*, *pairwise* e *listwise* [7, 29] (do Inglês: baseadas em pontos, pares e listas, respectivamente). Distinguir entre as abordagens é importante, pois permite visualizar melhor suas características, como o tipo de função de perda ou objetivo de otimização que pode ser formulado.

A aprendizagem *pointwise* para ordenação de resultados recebe como entrada um vetor de *features* $\boldsymbol{x}$ para documentos [29] e aprende um mapeamento do gabarito para os rótulos y. Dependendo do domínio de y, métodos padrão de aprendizado supervisionado podem ser utilizados. Por exemplo, pontuação binária de relevância (i.e., prever se um documento é relevante ou não) pode ser aprendida usando abordagens de classificação [31]. Abordagens de regressão podem ser utilizadas para aprender pontuações de relevância contínua (i.e. o grau de relevância de um documento) [11, 16]. A função de perda depende de que uma abordagem específica seja escolhida, podendo ser a perda um-zero no caso de classificação ou o erro quadrático no caso da regressão. A desvantagem de ambas formulações está no fato delas não corresponderem bem ao problema de ordenação de resultados em RI. Em L2R, a ordem na qual os documentos é posta é crucial, enquanto na predição exata de uma relevância não é. É possível termos um desempenho perfeito na ordenação de resultados, mesmo com altas taxas de erro na classificação ou regressão. Além disso, os conjuntos de treino em RI frequentemente são desbalanceados (pode haver ordens de magnitude mais documentos não relevantes do que relevantes), tornando a aprendizagem difícil. As vantagens da abordagem *pointwise* estão na sua complexidade, que é baixa, comparada às abordagens *pairwise* e *listwise*, além dos métodos de classificação e regressão serem aplicáveis diretamente. Extensões desta abordagem incluem regressão ordinal, onde o mapeamento da pontuação de saída e limiares, para distinguir diferentes níveis de relevância, são aprendidos simultaneamente. Por exemplo, Prank é popular e eficaz em configurações onde predições de relevância absolutas são necessárias [12].

Abordagens *pairwise* para ordenação de resultados operam sobre pares de documentos, i.e., eles recebem como entrada pares de vetores de documentos para uma dada consulta $(\boldsymbol{x}_1, \boldsymbol{x}_2, q) \in \mathbb{R}^n \times \mathbb{R}$ [29]. Esses pares são mapeados para rótulos binários, e.g., $y \in \{-1, +1\}$, que indicam se os dois documentos estão sendo apresentados na ordem correta ou se deveriam ser trocados. Esse problema pode ser reduzido a um problema de classificação binária, transformando a entrada em um único vetor, na combinação $\boldsymbol{x} = \boldsymbol{x}_{1,q} - \boldsymbol{x}_{2,q}$. Nesse caso, a função de perda poderia ser baseada nos erros de classificação sobre todos os pares de documentos. Otimizar para esta função de perda

pode, como nas abordagens *pointwise*, resultar em erro com relação ao desempenho de ordenação de resultados, porque métricas de avaliação de RI são muito mais sensíveis a mudanças na ordenação de resultados no topo do resultado, do que em mudanças em seu final. Entretanto, contar os erros de classificação consideraria todas as trocas de ordem dos documentos igualmente importantes. Outro problema reside no fato de consultas com muitos documentos candidatos associados interferirem no resultado, já que nas métricas em RI tipicamente calculamos a média, dando o mesmo peso para todas as consultas. A complexidade da abordagem é maior do que na *pointwise* (quadrática no número de documentos se todos os possíveis pares forem considerados), mas métodos de amostragem se mostraram eficazes e eficientes [36]. Uma das principais vantagens sobre abordagens *pointwise* está em abstrair pontuações específicas de relevância. Ao invés disso, foca-se na ordem relativa dos pares de documentos. Uma abordagem amplamente conhecida e efetiva de aprendizado *pairwise* para ordenação de resultados é o rankSVM. Esta abordagem utiliza SVMs para minimizar a perda *hinge* entre os pares [18, 23]. Abordagens relacionadas foram desenvolvidas na área de aprendizado por preferência [17].

Listwise é um grupo de abordagens que opera sobre a lista ordenada completa [29]. Estas abordagens recebem como entrada o vetor de *features* n-dimensional de todos os m candidatos para uma consulta $\boldsymbol{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$, e aprendem a prever, a pontuação para todos os candidatos, ou a permutação completa dos documentos no resultado. A função de perda para tais abordagens pode ser a métrica de RI, apesar de tais funções poderem ser difíceis de otimizar diretamente, por não serem suaves ou diferenciáveis. Alternativas incluem aproximações suaves de tais métricas (e.g., Softrank [40]), ou, quando o gabarito da ordenação dos resultados é dado, medidas de diferença entre a ordenação predita e a ordenação do gabarito (e.g., ListNet [7]). Abordagens *listwise* têm a vantagem de serem otimizáveis diretamente para altos desempenhos na métrica escolhida. No entanto, sua complexidade pode ser alta. Elas são consideradas métodos do estado da arte, como evidenciado pelo método vencedor da Yahoo! Learning to rank Challenge 3, e de Yandex Personalized Web Search Challenge [6]. O melhor desempenho foi alcançado por um comitê que combinava modelos *listwise*, incluindo Lambdarank [5, 15] e LambMART [6], o qual otimiza métricas *listwise* diretamente, usando gradiente descendente e árvores de decisão impulsionadas (boosted decision trees, do inglês). No entanto, Wu et al. [44] utilizaram várias técnicas de combinação de modelos com abordagens *pointwise* e *pairwise* e mais redes neurais profundas e ganharam a competição CIKM Cup 2016 Track 2: Personalized E-Commerce

Search Challenge[1]. Assim como em outras configurações de aprendizado supervisionado, L2R tipicamente assume que um conjunto representativo de dados (incluindo gabarito) está disponível em tempo de treino, para que então características dos dados possam ser estimadas. Os rótulos dos dados são normalmente obtidos de juízes especialistas por marcação manual. Um processo caro, porque os especialistas podem entender as consultas de modo diferente do usuário leigo do sistema. Portanto, corre-se o risco de não capturar precisamente as preferências dos usuários [35].

No contexto de comércio eletrônico, é possível obter rótulos de forma barata através do registro comportamental dos usuários. Ao acessar as páginas de lojas eletrônicas, guarda-se cada interação dos consumidores com o sistema: cada clique, cada paginação de resultado, cada consulta realizada. Tem-se assim o *feedback* implícito dos usuários do sistema, que podem ser combinados de diversas formas. Em contrapartida, o *feedback* obtido desta maneira é ruidoso e de baixa qualidade, pois insere o que chamamos de viés posicional [23, 24, 13]. Esse consiste na propensão do consumidor clicar em produtos que apareçam no topo do resultado, mesmo que o resultado não seja o melhor para a sua intenção. A definição do que será o rótulo, conceito de relevância, pode ser modificado de acordo com os interesses da aplicação [25, 28].

Neste trabalho testou-se vários conceitos de relevância (os termos "conceito de relevância", "esquema de relevância" e "definição de relevância" serão utilizados de forma intercambiável) e os modelos gerados a partir deles buscando o melhor objetivo para os algoritmos de aprendizagem no contexto de comércio eletrônico. Procurou-se melhorar o desempenho da função de ordenação usada por uma solução comercial de busca para comércio eletrônico. Realizou-se testes A/B, verificando-se ou não a concordância entre seus resultados e os resultados obtidos a partir de avaliações feitas sobre os registros de interações dos usuários com a métrica NDCG.

2.3 Trabalhos Relacionados

Diversos autores publicaram trabalhos sobre L2R para comércio eletrônico. Não na mesma proporção que para L2R para busca na web, é certo. E mesmo os que publicaram trabalhos se utilizando de L2R o fizeram num contexto diferente, *Marketplace*, que se assemelha mais a um sistema de classificados [10], plataformas como eBay e OLX por exemplo, do que a uma busca em catálogo de produtos. No primeiro podemos ter vários anúncios para um mesmo produto de diferentes vendedores com qualidades de anúncio diferentes. Enquanto no segundo todos os anúncios são padronizados, com o

[1] `https://competitions.codalab.org/competitions/11161`

mesmo nível de qualidade e não há diferentes anúncios para o mesmo produto. Esses trabalhos não investigam o problema de definir conceitos de relevância e nem do viés posicional. Também não realizaram testes A/B. O trabalho de Karmaker et al. [25] é o que mais se assemelha ao mostrado nesta dissertação. Os autores realizaram diversos experimentos com o método LambdaMart, comparando o NDCG de vários esquemas de relevância. No entanto, também não realizaram testes A/B, ou seja, não comprovam ganhos ou melhorias em cenários reais.

Para avaliar funções de ordenação de um sistema de comércio eletrônico é impraticável adotar a abordagem clássica, que consiste em manualmente rotular um conjunto de consultas e seus resultados [42]. A solução é utilizar os dados ruidosos dos registros das interações dos usuários. Meios de lidar com o ruído inerente a esse tipo de dado foram vastamente investigados. Joachims et al. [23] utilizaram dados de *feedback* implícito dos cliques para treinar modelos de aprendizagem. No entanto, não lidou com o ruído dos dados, o que convencionou-se chamar de viés posicional (do inglês, *positional bias*). Para lidar com o problema do viés posicional, vários autores surgiram com estratégias, por exemplo: Joachims et al. [24] adicionaram novos exemplos de treino quando certas condições ocorriam. A partir de então tentou-se lidar com o problema utilizando modelos de clique para estimar de fato a relevância dos resultados para uma dada consulta. O livro por Chuklin et al. [9] resume os últimos trabalhos na área. O que apresenta-se neste trabalho difere dos anteriores principalmente por seu contexto de comércio eletrônico, com o uso dos registros de vendas de clientes de varejo eletrônico, mesmo utilizando o modelo gráfico probabilístico DBN proposto por Chapelle [8], para lidar com o problema do viés posicional. Além de propor novas definições de relevância utilizando variadas combinações dos registros de clique e compra dos usuários.

Neste trabalho investiga-se o modo como os métodos já existentes na literatura lidam com os dados de comércio eletrônico. E também diferentes conceitos para o que chamamos de relevância e como eles podem ser usados como o objetivo de treino dos algoritmos de L2R.

Capítulo 3

Estimativas de Relevância

A fim de melhorar o desempenho da função de ordenação usada por uma solução proprietária de busca para comércio eletrônico, com o uso de técnicas de L2R, explorou-se a criação de diferentes formas de se estimar a relevância. Combinou-se dados de registros comportamentais de usuários e comparou-se os modelos criados a partir dos conceitos desenvolvidos usando diferentes métodos de aprendizagem.

3.1 Esquemas de Relevância

O problema que se quer resolver é a criação de uma função de ordenação que, tendo como entrada um produto, uma consulta e um usuário, quantifique a probabilidade do usuário comprar o produto, ordenando os produtos da maior probabilidade para a menor no resultado da consulta.

Os esquemas de relevância serão utilizados como o gabarito dos exemplos de treino para os métodos de L2R, que serão otimizados baseados neles.

Um esquema de relevância é uma formulação que tenta estimar essa probabilidade. A tabela 3.1 lista os esquemas propostos e estudados neste capítulo.

3.1.1 *Feedback* Simples

Nesse esquema de relevância, atribuímos a estimativa de relevância de acordo com a relação de um produto para a consulta. Se ele foi comprado, clicado ou se é da mesma categoria que um dos produtos comprados ou clicados. Ele considera que uma compra é mais importante que um clique, que por sua vez é mais importante que um produto que apenas foi mostrado no resultado, que por sua vez é mais importante que um produto que não divide nenhuma categoria com um dos produtos clicados ou comprados. Esta

Tabela 3.1: Esquemas de relevância.

Esquema
Feedback Simples
Feedback Normalizado
Frequência de Clique por Consulta
Frequência de Venda por Consulta
Frequência de Venda
Probabilidade de Satisfação
Taxa de Conversão
Taxa de Cliques

ordem de importância vem da intuição dada pelo contexto de comércio eletrônico. O sinal de compra é mais importante que o sinal de clique para o lojista. A relevância atribuída por esse esquema de relevância entre uma consulta e os produtos em seu resultado é constituída conforme mostrado na Tabela 3.2. Ele atribui valor de 3 para produtos comprados, 2 para produtos clicados, 1 para produtos da mesma categoria de algum dos produtos comprados ou clicados e 0 para o restante.

Tabela 3.2: Esquema de relevância *Feedback* Simples.

Tipo de *feedback*	Relevância
Compra	3
Clique	2
Mesma categoria de produto comprado/clicado	1
Outro	0

3.1.2 *Feedback* Normalizado

Um dos problemas do *Feedback* Simples é que ele não leva em consideração a diferença entre produtos com o mesmo tipo de fonte para estimar a relevância. Entre dois produtos comprados, qual é mais relevante? Um que foi vendido 1000 vezes ou um que foi vendido 10? O *Feedback* Normalizado tenta resolver esse problema definindo relevância como:

$$FS = R(c,p) = 3 * NVendas(c,p) + 2 * NCliques(c,p) + RCat(c, Cat(p))$$

Onde:

$$NVendas(c,p) = \frac{Vendas(c,p)}{MaxVendas(c)}$$

$$NCliques(c,p) = \frac{Cliques(c,p)}{MaxCliques(c)}$$

$$RCat(c,cat) = \frac{PVendidos(c,cat) + PClicados(c,cat)}{2}$$

c é uma consulta qualquer.

p é um produto qualquer.

$R(c,p)$ é a relevância entre a consulta c e o produto p.

$Cat(p)$ é uma função que retorna a categoria de um produto p qualquer.

$Cliques(c,p)$ é uma função que retorna o número de cliques que p teve para a consulta c.

$MaxCliques(c)$ é uma função que retorna o número de cliques do produto mais clicado entre os retornados no resultado para a consulta c.

$NCliques(c,p)$ é a função $Cliques(c,p)$ normalizada pelo seu máximo entre c e todos os possíveis valores para p.

$Vendas(c,p)$ é uma função que retorna o número de vendas que p teve para a consulta c.

$MaxVendas(c)$ é uma função que retorna o número de vendas do produto mais vendido entre os retornados no resultado para a consulta c.

$NVendas(c,p)$ é a função $Vendas(c,p)$ normalizada pelo seu máximo entre c e todos os possíveis valores para p.

$PVendidos(c,cat)$ é o percentual de produtos vendidos para a consulta c com uma categoria cat qualquer.

$PClicados(c,cat)$ é o percentual de produtos clicados para a consulta c com uma categoria cat qualquer.

$RCat(c,cat)$ é a média entre o $PVendidos(c,cat)$ e o $PClicados(c,cat)$ e é como o *Feedback* Normalizado define a relevância entre uma consulta c e uma categoria cat.

Esse esquema leva em conta a frequência de cliques e compras dos resultados, resolvendo o problema do esquema anterior em decidir qual produto deve ficar à frente

do outro, se ambos tiverem sido clicados/comprados. Ele mantém a hierarquia de importância *compra* > *clique* > *categoria* , por meio dos fatores 3 e 2, que multiplicam os componentes de venda e clique, respectivamente. Eles foram escolhidos arbitrariamente, com o objetivo de dar mais importância a interações envolvendo compras do que as que envolviam cliques. O componente de categoria gradua a importância de um produto de acordo com os produtos clicados e comprados que compartilham a sua categoria.

Quando o produto ou a consulta não tem registros históricos, a relevância entre eles é nula.

3.1.3 *Feedback* Baseado em Estatísticas

Várias das *features* que descrevem a relação entre produtos e consultas são estatísticas, por exemplo: a frequência com que um produto foi clicado ou comprado, a frequência com que ele foi clicado ou comprado para uma consulta, a frequência com que ele foi clicado na última semana, ou no último dia. Todas são estatísticas que podem ser obtidas através dos registros de uma loja. Tomando-se individualmente qualquer uma dessas estatísticas é possível criar um esquema de relevância baseado em seu valor e, como ela descreve a interação dos usuários com os produtos, tem-se uma definição de relevância baseada no *feedback* dado por ela. Esquemas definidos dessa maneira são muito simples, quando comparados com os outros definidos neste capítulo, sendo utilizados apenas para fins de comparação nos experimentos posteriores.

Genericamente, esta família de esquemas de relevância pode ser definida como:

$$R_{Est}(c, p) = Est(c, p)$$

Onde:

$Est(c, p)$ é o valor de uma estatística qualquer entre a consulta c e o produto p.

Um exemplo é a frequência de clique de um produto para uma consulta. Se nos registros disponíveis sabe-se que, para a consulta c, o produto p foi clicado n vezes, então $R_{Est}(c, p) = n$.

Considerando estatísticas com respeito apenas aos produtos:

$$R_{Est}(p) = Est(p)$$

Onde:

$Est(p)$ é uma estatística qualquer para o produto p, e.g., o número de vezes que ele foi comprado.

3.1.4 Probabilidade de Satisfação

A Probabilidade de Satisfação (PS) é um esquema de relevância proposto por Chapelle e Zhang [8], que desenvolveram um modelo de previsão de cliques. O modelo consiste numa rede Bayesiana que é capaz de estimar a relevância entre uma URL e uma consulta. Como o trabalho inicial lidava com URLs em resultados de buscas na web, o contexto foi adaptado para o de comércio eletrônico. Assim, o modelo adaptado estima a relevância de um produto para uma consulta. A estimativa de relevância é dada por:

$$r_p = P(S_i = 1|C_i = 1)P(C_i = 1|E_i = 1)$$

Onde:

i é a posição do produto p no resultado.

$P(S_i = 1|C_i = 1)$ é a probabilidade de satisfação do usuário com o produto p dado que ele foi clicado.

$P(C_i = 1|E_i = 1)$ é a probabilidade de clicar no produto p dado que ele foi examinado.

Nessa classe de modelos, examinar significa que o cliente viu um dos produtos no resultado. E_i é o evento onde o usuário examina o produto na posição i do resultado.

S_i é o evento onde o usuário se satisfez com a página apresentada (o conteúdo dado pela URL ou a página de produto neste caso) e parar a busca.

A relação entre esses dois eventos é dado pela chance do usuário clicar no resultado dada a relevância percebida por ele após examinar o produto da página de resultado, o componente $P(C_i = 1|E_i = 1)$. E a chance do usuário ficar satisfeito com o produto após entrar na sua página, aqui a verdadeira relevância: $P(S_i = 1|C_i = 1)$. Temos assim a razão entre a relevância que ele percebeu na página de resultado e a relevância que real após visitar a página do produto.

Além disso, ao invés de levar em consideração apenas dados de clique para treinar o modelo, levou-se em conta também dados de compras e treinou-se um segundo modelo, o qual estima a relevância de um produto de forma análoga à definida acima, porém, troca o evento C_i, que é o clique num produto na posição i, por um novo evento Cr_i, que é a compra do produto na posição i. Dessa forma, define-se outra estimativa de relevância, levando em conta o evento de compra:

$$r_{p,compra} = P(S_i = 1|Cr_i = 1)P(Cr_i = 1|E_i = 1)$$

Combinou-se então os dois modelos treinados, um com dados de clique e outro com dados de compra e definiu-se a relevância de um produto p para a consulta c,

como:

$$PS = R_S(c, p) = 3r_{p,compra} + 2r_p$$

Onde se dá mais importância para o fator relacionado a compras, numa razão de 3 : 2. De novo, esses valores foram definidos arbitrariamente, com o fim de dar maior importância ao fator relacionado com compras do que com cliques.

Este esquema de relevância é o primeiro dos apresentados até agora a levar em conta o problema do viés posicional, pois considera a posição em que um produto é clicado/comprado durante o seu processo de treino.

3.1.5 Taxas de Métricas de Interesse

Também foram realizados experimentos levando em conta algumas das definições de relevância descritas por Karmaker et al.[25]. A saber, taxa de conversão (TxC) e taxa de cliques (TxCl). A taxa de conversão é definida pelos autores como, dado um conjunto de produtos D_c para a consulta c, então, a taxa de conversão é computada da seguinte maneira, para $p \in D_c$:

$$\text{conv}(c, p) = \frac{\text{pedidos}(c, p)}{\text{visualizações}(c, p)}$$

Então, a relevância baseada na taxa de conversão é computada como:

$$rel_{conv} = teto\left(4.\frac{conv(c, p)}{max_{p \in D_c} conv(c, p)}\right)$$

Onde:

visualizações(c, p) é o número de vezes que o produto foi mostrado para a consulta.

pedidos(c, p) é o número de vezes que o produto foi comprado para a consulta.

A relevância baseada na taxa de cliques é computada de maneira similar, substituindo *pedidos* por *cliques*. Dessa forma, temos dois esquemas de relevância, um para compras e outro para cliques.

3.2 Experimentos

Realizou-se quatro rodadas de experimentos. Duas dessas rodadas são o que foi denominado de experimentos ***offline***, pois foram realizados utilizando-se modelos treinados e testados em dados de produção obtidos com registros de consultas submetidas às lojas. Nesse caso, os registros foram divididos em treino, validação e teste. As outras

duas rodadas consistiram em realizar testes A/B, colocando-se uma das funções de ordenação em produção competindo com nosso *baseline*. Essa segunda forma de experimento foi denominada ***online***. Nos testes *offline*, não há como medir o impacto potencial das melhorias dos métodos em fatores como cliques ou vendas para o lojista, e portanto utilizou-se como métrica de avaliação o NDCG, uma métrica amplamente utilizada na literatura. Nas avaliações *online*, pudemos avaliar o impacto real da mudança de sistemas nas vendas realizadas pelas lojas ou ainda o aumento na taxa de cliques nos resultados de consultas.

As rodadas de experimentos foram realizadas na ordem: 1) experimento *offline*, 2) experimento *online*, 3) experimento *offline*, 4) experimento *online*. A primeira rodada teve o objetivo de escolher um método de geração de modelos, de funções de ordenação, para a realização do teste A/B. Na segunda rodada usou-se o método escolhido em produção (cf. Capítulo 4). A terceira rodada foi para experimentar outras definições de relevância publicadas após a realização do teste A/B e comparar com o esquema de Probabilidade de Satisfação, verificando-se o desempenho de modelos de clique adaptados ao contexto de comércio eletrônico e considerando-se também dados de registros de compras. A última rodada consistiu em utilizar o conceito de Probabilidade de Satisfação em um teste A/B (cf. Capítulo 4).

Para as rodadas *offline* gerou-se dados de treino para três lojas de ramos distintos. Uma loja de móveis, uma de esportes e uma de cosméticos. Os dados das lojas de móveis e de esportes foram usados na primeira rodada *offline* e os da loja de cosméticos na segunda rodada *offline*.

Na solução comercial de busca utilizada nos experimentos, a função de ordenação *baseline* consistiu numa combinação linear entre a representação vetorial de cada produto e a consulta que o tem em seu resultado, e um vetor de pesos configurado manualmente por uma equipe de especialistas em busca para comércio eletrônico. A função foi configurada de acordo com a experiência desses especialistas e mostrou ganhos em diversas disputas comerciais com outras soluções comerciais de busca. O processo de decisão pelos pesos utilizados pela função foi configurado num processo iterativo através da observação dos efeitos que cada mudança no vetor de pesos ocasionava na ordem em que os produtos eram mostrados no resultado de várias consultas de grandes varejistas do comércio eletrônico.

Os métodos testados competiram com o *baseline* utilizando o mesmo conjunto de *features*: estatísticas de clique, venda e proximidade textual entre a consulta e os termos que descrevem o produto. O conjunto exato de *features* utilizadas é mantida em segredo pela empresa detenedora da solução de busca utilizada. É importante dizer que normalmente seriam utilizadas várias outras *features* durante o treino das funções

de ordenação. No entanto, seria injusto com o *baseline*, pois como é configurado manualmente, torna-se cada vez mais difícil e caro configurar cada peso de forma manual. Esse é outro problema que a criação automatizada de funções resolve: possibilita a fácil experimentação com diferentes conjuntos de *features*, sem a necessidade de um processo custoso de configuração manual de uma quantidade crescente de pesos.

Em todos os experimentos, o teste de significância utilizado foi o *bootstrap*.

3.2.1 Conjunto de Dados

Os dados utilizados nos experimentos e avaliações foram criados a partir de uma solução proprietária de busca.

Para os experimentos *offline* e o primeiro teste A/B, utilizou-se um período de 45 dias de registros para a criação dos dados de treino, validação e teste dos métodos. O período de treino composto de 30 dias e o de validação e teste por 8 e 7 dias, respectivamente.

Para o último teste A/B, utilizou-se uma janela de 30 dias a partir da data atual para treino, validação e teste. Realizou-se validação cruzada de 10 conjuntos.

As consultas de cada período eram submetidas à solução de busca, a qual criava os exemplos de treino para o método de aprendizagem escolhido, num formato amplamente utilizado na literatura de L2R.

Esse processo pode ser utilizado para qualquer loja cujos dados estejam disponíveis para a solução de busca. Por conta disso, a depender do domínio da loja de interesse algumas consultas possuem milhões de resultados. Isso é um problema pela complexidade de alguns métodos, como: LambdaMART, RankNet e métodos de programação genética. O tempo de criação de um modelo duraria mais que um dia.

O problema em treinos que durem mais de um dia consiste na utilidade dos modelos criados a partir dos métodos. No contexto de comércio eletrônico os dados utilizados para treino chegam numa velocidade tal que modelos mais antigos que um dia são modelos que não levam em conta as informações mais atuais disponíveis. Além disso, os experimentos realizados nesta dissertação indicam que os métodos mais demorados não justificavam a demora de tempo de treino com melhores desempenhos, ficando muitas vezes atrás de outros métodos com menor tempo de treino. Por essa razão abandonarem-se métodos baseados em programação genética nos experimentos realizados nessa dissertação.

Durante a construção do conjunto de dados, os resultados de consultas com mais de 200 produtos foram podados. Considerou-se apenas os 200 primeiros produtos do resultado, incluindo além deles qualquer produto que houvesse sido comprado no

período sendo considerado. Por exemplo, se para a consulta c fossem apresentados 500 resultados, apenas os primeiros 200 produtos no resultado seriam considerados e mais qualquer outro produto comprado ou clicado, independente de estar ou não entre os primeiros 200 resultados. Ou seja, mesmo podando o resultado para a criação dos dados de treino, alguns ainda seriam maiores que 200 produtos.

3.2.2 Métodos de Aprendizagem de Máquina

Os métodos de aprendizagem de máquina escolhidos para os experimentos foram os seguintes:

- SVM
- RankNet
- LambdaMART

A escolha dos métodos foi devido à diversidade de abordagens. Tem-se modelos de diferentes tipos: um SVM, uma rede neural e uma floresta. Dois métodos do tipo *pairwise*, SVM e RankNet, e um *listwise*, LambdaMART.

A implementação dos métodos usada para o SVM foi a encontrada no pacote sofia-ml[1], com a configuração utilizada no apêndice A. As dos métodos RankNet e LambdaMART foi a do pacote RankLib[2], com configurações padrão para ambos.

3.2.3 Primeira rodada

Na primeira rodada de experimentos utilizou-se dados de duas lojas, uma do setor de esportes e outra do setor de móveis. Para avaliar os métodos, nestes experimentos *offline*, utilizou-se a métrica NDCG@n, uma variação da descrita no Capítulo 2, na seção 2.1, para $n \in \{1, 3, 5, 10, 20\}$. Nesta variação, computamos a métricas apenas até a posição n, ao invés de para todo o resultado.

A intuição ao realizar esses experimentos para a escolha de um método está em considerar que um método é melhor que o *baseline* se ele for consistentemente melhor em várias definições de relevância.

[1] https://code.google.com/archive/p/sofia-ml/
[2] https://sourceforge.net/p/lemur/wiki/RankLib/

Esquemas de Relevância Utilizados

Abaixo segue a lista de esquemas utilizados nos experimentos: o *Feedback* Simples, que categoriza cada produto no resultado; o *Feedback* Normalizado que categoriza de forma a diferenciar dois produtos numa mesma categoria; três esquemas que utilizam estatísticas diretamente para atribuir a relevância; número de cliques do produto para consulta; número de vendas do produto para consulta; e número de vendas do produto no site inteiro.

Os últimos três esquemas de relevância: Probabilidade de Satisfação, Taxa de Conversão e Taxa de Cliques, não foram utilizados porque não haviam sido concedidos até a realização desta primeira rodada de experimentos.

Tabela 3.3: Esquemas de relevância utilizados em primeira rodada de experimentos *offline*.

Esquema
Feedback Simples
Feedback Normalizado
$R_{\text{Cliques por Consulta}}$
$R_{\text{Vendas por Consulta}}$
R_{Vendas}

Por conta de alguns métodos de L2R, nomeadamente LambdaMART e RankNet, requererem valores $\mathbb{Z}_{\geq 0}$ como rótulos de relevância foi necessário discretizar o valor da relevância no conjunto de dados. Dada uma definição de relevância *defrev* qualquer, um conjunto $p \in D_c$ de produtos no resultado de uma consulta c, o seu valor normalizado é dado por:

$$rel_{defrev} = teto\left(5.\frac{defrev(c,p)}{max_{p \in D_c} defrev(c,p)}\right)$$

Onde:

defrev(c,p) é o valor dado pela definição de relevância entre a consulta c e o produto p.

Discussão dos Resultados

Os resultados com melhores desempenho e com diferença significativa ($p < 0{,}05$) sobre o *baseline* estão em negrito.

Para melhor apresentar os resultados o nome dos esquemas de relevância foram abreviados, conforme a tabela 3.4.

Tabela 3.4: Abreviações dos nomes de esquemas de relevância

Esquema	Abreviação
Feedback Simples	FS
Feedback Normalizado	FN
R_{Vendas}	V
$R_{VendasporConsulta}$	VC
$R_{CliquesporConsulta}$	CC

Tabela 3.5: Resultados das avaliações com diferentes esquemas de relevância - Esportes.

Loja	**Esquema**	**NDCG@k**	**Baseline**	**LMART**	**RankNet**	**SVM**
Esportes	V	1	0.0259	**0.0849**	0.0565	0.0565
		3	0.0366	**0.0849**	0.0565	0.0564
		5	0.0388	**0.0849**	0.0565	0.0564
		10	0.0416	**0.0849**	0.0565	0.0564
		20	0.0426	**0.0849**	0.0565	0.0564
	CC	1	0.0056	0.0065	0.0050	**0.0065**
		3	0.0057	0.0064	0.0053	**0.0065**
		5	0.0060	0.0064	0.0055	**0.0065**
		10	0.0060	0.0064	0.0057	**0.0065**
		20	0.0060	0.0064	0.0057	**0.0065**
	VC	1	0.0920	0.0919	0.0916	0.0917
		3	0.0981	0.0944	0.0951	0.0967
		5	0.1004	0.0965	0.0975	0.0989
		10	0.1029	0.0995	0.1004	0.1009
		20	0.1048	0.1015	0.1026	0.1026
	FN	1	0.8361	0.8686	0.8453	**0.9582**
		3	0.8626	0.8855	0.8776	**0.9604**
		5	0.8703	0.8891	0.8858	**0.9564**
		10	0.8815	0.8948	0.8936	**0.9528**
		20	0.8895	0.8995	0.8991	**0.9513**
	FS	1	0.7728	0.8427	**0.9784**	0.8528
		3	0.8305	0.8806	**0.9580**	0.8903
		5	0.8449	0.8938	**0.9514**	0.8993
		10	0.8625	0.91	**0.9483**	0.9110

Tabela 3.5 – continuação

Loja	Esquema	NDCG@k	Baseline	LMART	RankNet	SVM
		20	0.8788	0.9206	**0.9501**	0.9211

No conjunto de dados da loja de esportes, com exceção do VC, em todos os outros esquemas os métodos de L2R tiveram resultados melhores. O LambdaMART foi em média 136% superior ao *baseline* no esquema V. No esquema CC, o SVM venceu o LambdaMART, sem diferenças significativas, sendo 11% superior ao *baseline*. No FN o SVM também venceu os outros métodos, com diferença de 10% sobre o *baseline*. O RankNet foi o vencedor no esquema FS, com 14% sobre o *baseline*. O SVM foi superior ao *baseline* em 4 de 5 esquemas testados, sendo que no único esquema onde perdeu, junto com os demais métodos, foi o que perdeu por menor percentagem (1,5%). O RankNet perdeu em 2 ocasiões, no esquema VC e no CC, por 2% e 7%, respectivamente. O LambdaMART, assim como o SVM perdeu apenas no esquema VC.

Tabela 3.6: Resultados das avaliações com diferentes esquemas de relevância - Móveis.

Loja	Esquema	NDCG@k	Baseline	LMART	RankNet	SVM
Móveis	V	1	0.0498	**0.0849**	0.0799	0.0825
		3	0.0649	**0.0849**	0.0814	0.0837
		5	0.0686	**0.0849**	0.0819	0.0838
		10	0.0707	**0.0849**	0.0822	0.0841
		20	0.0714	**0.0849**	0.0822	0.0841
	CC	1	0.0011	0.0009	0.0011	0.0011
		3	0.0010	0.0009	0.0011	0.0011
		5	0.0010	0.0009	0.0011	0.0011
		10	0.0011	0.001	0.0011	0.0011
		20	0.0011	0.001	0.0011	0.0011
	VC	1	0.0832	**0.0936**	0.0891	0.0825
		3	0.0861	**0.0923**	0.0892	0.0841
		5	0.0868	**0.0921**	0.0893	0.0846
		10	0.0879	**0.092**	0.0896	0.0851
		20	0.0887	**0.0921**	0.09	0.0857
	FN	1	0.8294	**0.9501**	0.852	0.9388
		3	0.8713	**0.9488**	0.8954	0.9446

Tabela 3.6 – continuação

Loja	Esquema	NDCG@k	Baseline	LMART	RankNet	SVM
		5	0.8818	**0.9485**	0.905	0.9445
		10	0.8926	**0.9478**	0.9121	0.9441
		20	0.9005	**0.9476**	0.9165	0.9440
	FS	1	0.8057	0.8892	**0.9876**	0.9079
		3	0.8723	0.9231	**0.9776**	0.9373
		5	0.8877	0.9329	**0.9744**	0.9442
		10	0.9052	0.9421	**0.9732**	0.9521
		20	0.9197	0.9502	**0.9742**	0.9587

Nos resultados da loja de móveis, ao contrário do primeiro conjunto de dados, em nenhum dos esquemas o *baseline* ganhou de todos os modelos aprendidos. O LambdaMART foi superior 32%, em média, no esquema V, sendo acompanhado de perto dos outros métodos, que tiveram ganho de 27% e 30% sobre o *baseline*, o RankNet e o SVM, respectivamente. No esquema CC, o LambdaMART foi o único a perder do *baseline*. No esquema VC, o método a ser derrotado pelo *baseline* dessa vez foi o SVM, sendo que os outros dois métodos venceram. O LambdaMART supera o *baseline* por 8,5% no esquema FN, seguido de pelo SVM que supera o *baseline* por 8%. Finalmente, de novo, o RankNet derrota todos os outros métodos no esquema FS. Neste conjunto de dados, o único método a perder para o *baseline* em algum esquema foi o SVM, no esquema VC, por 2,5%.

Houve um empate entre os métodos de L2R. Todos eles obtiveram ganhos sobre o *baseline* em 8 de 10 configurações de experimentos (a média da variações do NDCG@k, 5 esquemas de relevância para cada loja).

Levando em conta esses resultados foi decidido que o método SVM seria o utilizado no teste A/B. Apesar de não gerar o melhor modelo em todos os casos ele possui um modelo de mais fácil compreensão. Dado que o método gera um modelo $\omega, \omega \in \mathbb{R}^n$, que pode ser entendido como a importância dada a cada *feature*. Se uma *feature* f_i recebeu um peso $2x$, ela é 2 vezes mais importante que um *feature* f_j que recebeu peso x. Além disso, o tempo de treinamento é bem menor em relação aos outros. Mesmo não tendo sempre o melhor resultado em alguns esquemas. As redes neurais e florestas geradas pelos outros métodos são opacas e seus tempos de treino são bem maiores que os obtidos com o SVM.

3.3 Terceira rodada

Nesta rodada de experimentos testou-se dois esquemas definidos por Karmaker et al. [25]: a taxa de cliques (TxCl), a taxa análoga para conversão (TxC) e a Probabilidade de Satisfação (PS) definida nas seções anteriores deste capítulo. Não utilizou-se os esquemas de relevância anteriores porque conta do resultado obtido na segunda rodada de experimentos e primeiro teste A/B (Cf. Capítulo 4).

Nessa rodada utilizou-se um conjunto de dados criado a partir dos registros de uma loja de cosméticos. A troca da loja utilizada para o teste A/B se deu pela impossibilidade técnicas e desinteresse por parte do lojista dono da loja móveis. Avaliou-se os mesmos métodos da primeira rodada, porém utilizando-se apenas a métrica NDCG@10, por conta dos resultados com maior frequência de interação serem os apresentados no topo da página de busca. Os resultados podem ser vistos na tabela 3.7.

Tabela 3.7: Resultados de avaliação com esquemas TxCl, TxC e PS.

Esquema	**Método**	**NDCG@10**
TxCl	*Baseline*	0.84947
	LambdaMART	0.88849
	RankNet	0.86060
	SVM	0.87438
TxC	*Baseline*	0.88928
	LambdaMART	0.93797
	RankNet	0.89180
	SVM	0.92280
PS	*Baseline*	0.75056
	LambdaMART	0.85930
	RankNet	0.85775
	SVM	0.86271

De novo, observa-se que o métodos de aprendizagem de máquina vencem o *baseline* seja qual for o esquema de relevância utilizado. No entanto, em alguns casos, a diferença não é significativa, como no esquema TxC entre o *baseline* e RankNet.

Utilizando-se o esquema TxCl, o vencedor entre os métodos é o LambdaMART. Que também vence no TxC, porém, sem diferença significativa com o segundo colocado, o SVM. Finalmente, utilizando-se o esquema que leva em conta o viés posicional dos dados, o PS, verifica-se que nenhum método de aprendizagem de máquina é significativamente melhor avaliado que o outro, no entanto, todos mostram-se melhores que o *baseline*.

Avaliação Cruzada

Com os conjuntos de dados criados nesta rodada, cruzou-se a avaliação dos modelos, analisando o resultado de modelos treinados em dados com esquemas de relevância diferentes dos dados de teste. Os resultados estão dispostos na tabela 3.8. Por exemplo, na primeira linha treinou-se um modelo do método LambdaMART com o esquema de relevância PS e ele foi avaliado em dados com os esquemas TxC e TxCl.

Tabela 3.8: Resultados de avaliação cruzada com esquemas TxCl, TxC e PS.

		Conjunto de Teste		
Conjunto de Treino	Método	PS	TxC	TxCl
PS	LambdaMART	–	0.51292	0.63337
	RankNet	–	0.50523	**0.68584**
	SVM	–	0.51923	0.62859
TxC	LambdaMART	0.79936	–	0.78936
	RankNet	0.75833	–	0.78468
	SVM	**0.8182**	–	**0.87133**
TxCl	LambdaMART	**0.7928**	0.78491	–
	RankNet	0.76216	0.7861	–
	SVM	0.76472	**0.84408**	–

Com modelos treinados no PS, todos os métodos ficaram com resultados muito próximos quando testados no TxC, sem diferença significativa entre eles. Por outro lado, quando testados no TxCl, o RankNet foi o melhor avaliado. No TxC, as diferenças entre os algoritmos foi mais pronunciada, com o SVM vencendo os outros métodos tanto em dados do PS quanto do TxCl.

O SVM foi o melhor modelo quando treinado em dados TxCl e avaliado no TxC, enquanto quando os modelos foram avaliados no PS o melhor foi o LambdaMART.

Nota-se que o resultado dos modelos treinados em dados que utilizavam o PS sempre tiveram o pior desempenho na avaliação cruzada. Isso evidencia a distância entre os conceitos de relevância. Um conceito que leva em consideração o viés posicional não é um bom preditor de relevância quando o outro conceito de relevância não leva o viés posicional em consideração.

Comparando os resultados da avaliação cruzada com a não cruzada há uma distância média de 20% entre os resultados. Utilizando o conjunto de treino TxCl e testando o modelo assim gerado no conjunto de teste TxC, o *baseline* é superior aos modelos do esquema PS em alguns casos, na avaliação cruzada. O mesmo também ocorre em quase todos os modelos menos o SVM quando treinados no TxC e testados no TxCl.

Capítulo 4

Testes A/B

Testes A/B são experimentos controlados. No contexto deste trabalho trata-se de dividir o tráfego de usuários acessando uma aplicação aleatoriamente, no caso o site de comércio eletrônico de algum lojista. Parte do tráfego acessará a versão de controle, que é o site sem alterações, enquanto outra parte, normalmente metade do tráfego, acessa uma versão com alguma modificação a fim de testar alguma hipótese: se botões vermelhos são mais clicados, se alterar a caixa de busca aumenta o seu uso pelos clientes, se o novo algoritmo de busca e ordenação de resultados aumenta vendas da loja. Aquilo que se está comparando entre os grupos é chamado de Métrica de Avaliação Geral (do Inglês, OEC - Overall Evaluation Criterion) e ela deve ser definida de acordo com os interesses do domínio onde o teste está sendo realizado. Neste trabalho usa-se o termo "métricas de interesse" com o mesmo propósito. O teste é colocado no ar e assim fica por um período pré-estabelecido de tempo ou até que se perceba uma diferença estatisticamente significante para a métrica de interesse. Para uma introdução mais profunda no assunto de testes A/B: Kohavi em [26].

Ao total dois testes A/B foram realizados em períodos e lojas de ramos diferentes. O primeiro e o segundo teste ocorreram após a primeira e segunda rodada de testes *offline*, respectivamente. No segundo teste tentou-se resolver problemas percebidos durante o primeiro. Em ambos, buscou-se verificar a correlação dos resultados de experimentos *offline*, avaliados com o NDCG@n, com o resultado dos experimentos online, avaliados com métricas de impacto nas vendas. Colocou-se em produção um método de aprendizagem de máquina treinado com registros da loja em questão.

4.1 Loja de Móveis

Esta foi a segunda rodada de experimentos. Neste teste foram utilizados dados de treino de uma loja de móveis construídos com o esquema de relevância *Feedback* Normalizado (FN), competindo com o *baseline* configurado manualmente. O FN foi escolhido porque até a data da realização deste teste A/B ele havia sido o esquema de relevância concebido que levava mais fatores em consideração em sua formulação.

Configuração

A tabela 4.1 resume as configurações do teste.

Tabela 4.1: Configuração de primeiro teste A/B

Tempo de teste (dias)	52
Tipo de divisão	70% controle e 30% experimento
Controle	*Baseline*
Experimento	SVM, FN como objetivo
Métrica de Interesse 1	RPV
Métrica de Interesse 2	Conversão

O teste durou 52 dias. Separou-se o tráfego que acessava a busca da loja eletrônica em dois grupos. O grupo controle tinha 70% do tráfego e o de experimento 30%.

O grupo de controle utilizou a função de ordenação configurada manualmente por um grupo de especialistas em comércio eletrônico. O grupo de experimento utilizou uma função treinada a partir de um método de L2R *pairwise*, um SVM, que utilizou o período imediatamente anterior de 30 dias ao teste para aprender um modelo, tendo a definição de relevância FN como objetivo.

As métricas de interesse foram RPV e conversão.

RPV, do inglês *Revenue per View*, significa receita por visualização, i.e., a soma total de receita bruta da loja, vindo das vendas de produtos, dividida pelo número de visualizações. Para os fins deste teste definiu-se a receita como o total faturado pelo grupo no período do teste, e uma visualização como um conjunto de acessos com intervalos menores que 30 minutos entre um e outro, sendo um acesso toda ação que recarrega a página web da loja: realizar uma consulta, paginar, aplicar um filtro nos resultados.

Conversão é o número de compras dividido pelo número de visualizações. Usou-se a definição de visualização já mencionada.

Resultados

Avaliou-se os resultados do teste durante toda sua execução. Ao final têm-se o que é mostrado na tabela 4.2. Os resultados tem significância estatística com $p < 0{,}05$, testadas via bootstrap.

Os dados do teste mostram que o SVM perdeu do *baseline* manualmente configurado pelo grupo de especialistas por 4,7% em conversão e por 2,26% em RPV.

Tabela 4.2: Teste A/B entre *baseline* e SVM (FN).

Métrica	Diferença
RPV	−2,26 %
Conversão	−4,7 %

Os resultados demonstram que resultados positivos em experimentos *offline* não correlacionam necessariamente com resultados positivos quando colocados em produção. A razão no entanto, não fica clara.

Teria sido a escolha do método? Os experimentos *offline* evidenciam o contrário. Em quase todos os experimentos o SVM venceu o *baseline* e venceu ou empatou com os outros métodos de aprendizagem automática.

Teria sido a definição de relevância escolhida? Talvez. O esquema utilizado foi o julgado mais abrangente até o momento da realização do teste. O segundo teste foi realizado só depois de ter-se aprendido sobre novas definições, como as citadas por Karmaker et al. e a Probabilidade de Satisfação.

4.2 Loja de Cosméticos

O segundo teste A/B, e quarta rodada de experimentos, foi realizado numa loja de cosméticos. Como dito anteriormente, não foi possível realizar um novo teste na loja de Móveis, o que teria sido o ideal. Os motivos para isso vão desde a vontade do lojista em participar do teste, quanto a variáveis internas da firma que fornece a solução de busca. Sendo assim, a loja de cosméticos era a que estava disponível no momento.

Este teste utilizou o esquema de relevância PS na construção de seus dados de treino. Além disso, o modelo era gerado diariamente, considerando-se os últimos 30 dias de registros comportamentais, conforme a figura 4.1. Assim, ele tenta sanar os prováveis problemas do primeiro teste. 1) Não considerar o viés posicional na formulação do conceito de relevância utilizado. 2) Não atualizar o modelo durante a execução do teste.

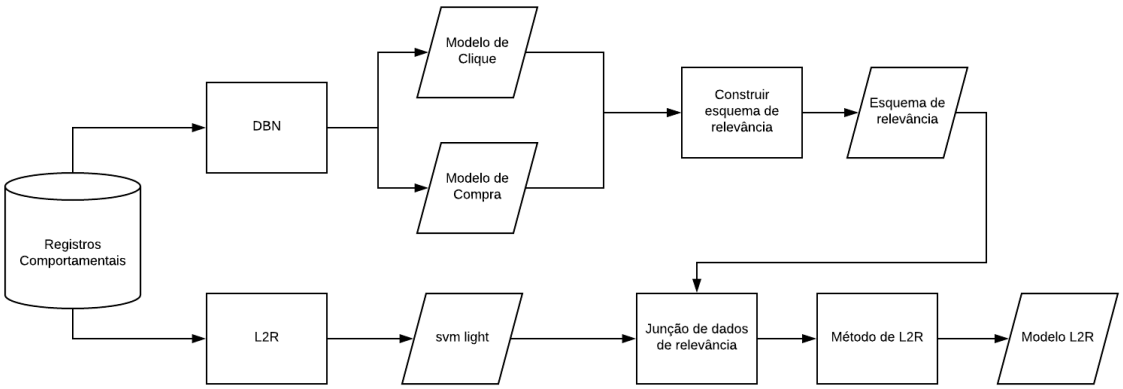

Figura 4.1: *Pipeline* de processamento de registros comportamentais para geração de modelo SVM, tendo PS como objetivo. Executado diariamente.

Configuração

A tabela 4.3 resume as configurações do teste.

Tabela 4.3: Configuração de segundo teste A/B

Tempo de teste (dias)	37
Tipo de divisão	50% controle e 50% experimento
Controle	*Baseline*
Experimento	SVM, PS como objetivo
Métrica de Interesse 1	RPV
Métrica de Interesse 2	Conversão

O teste durou 37 dias. As mesmas métricas de interesse do primeiro teste foram utilizadas. E, assim como no primeiro teste o grupo de controle utilizou o *baseline* configurado por especialistas.

Resultados

Avaliou-se os resultados do teste durante toda sua execução. Ao final tem-se o que é mostrado na tabela 4.4. Os resultados têm significância estatística com $p < 0{,}05$ testada via *bootstrap*.

Durante o teste ocorreram variações drásticas de até 30 vezes entre o valor mínimo e valor máximo dos pesos em um modelo SVM, o que indica a mudança no comportando dos usuários. Fica claro que é necessário a atualização do modelo com o tempo.

Computou-se o NDCG@10 durante a execução do teste, como medida de controle de qualidade do modelo. O gráfico 4.3 demonstra que durante quase todo o período de teste os modelos treinados são superiores ao modelo manual.

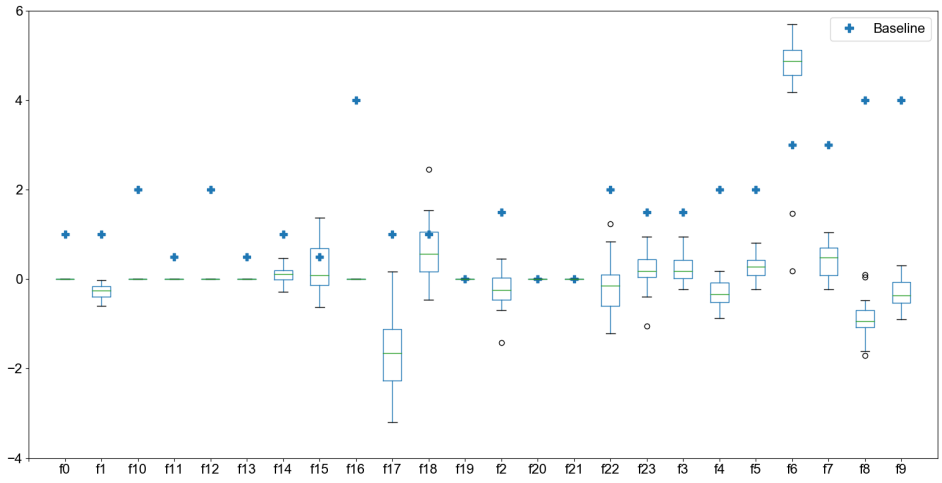

Figura 4.2: Comparação entre modelo manual e a distribuição dos pesos e suas médias aprendidos por SVM durante período de teste.

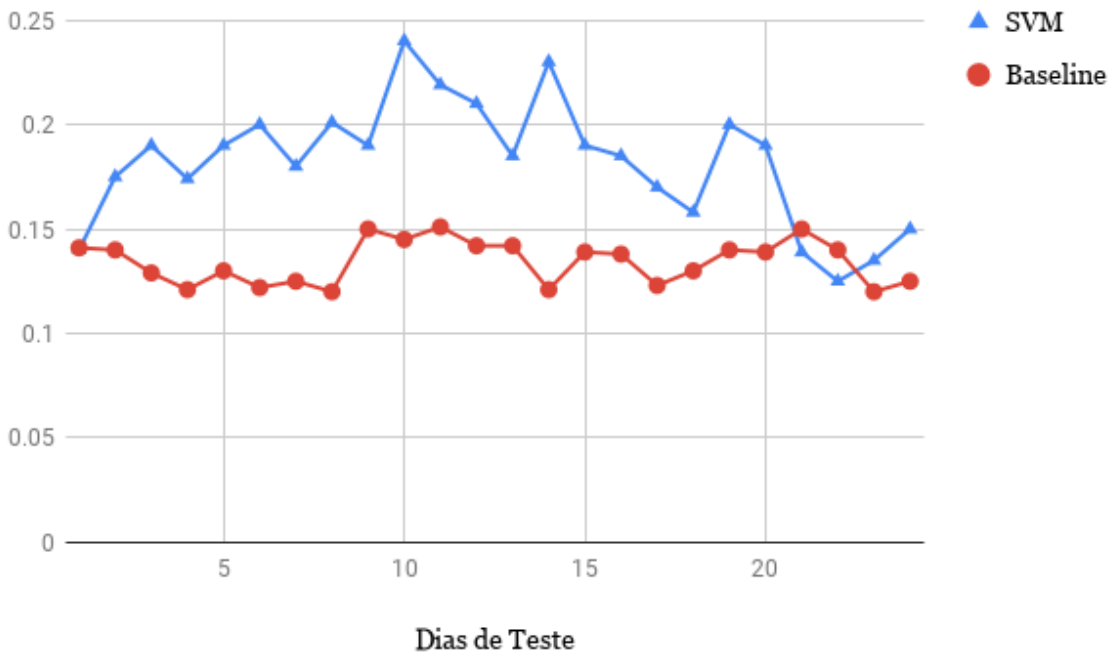

Figura 4.3: Comparação entre NDCG@10 de modelo treinado e *baseline*.

Neste teste, ao contrário do primeiro, houve um ganho 2,04% em RPV e 0,85% em conversão, do grupo de experimento em relação ao controle. Sendo a principal diferença entre os dois testes o modelo utilizado e como ele foi criado. O objetivo do método de L2R foi modificado. O PS considera o viés posicional em sua formulação. O

Tabela 4.4: Teste A/B entre *baseline* e SVM (PS).

Métrica	Diferença
RPV	2,04%
Conversão	0,85%

modelo foi atualizado diariamente, mudando em conjunto com o padrão dos registros comportamentais da loja.

4.3 Discussão

No primeiro teste A/B o uso de um modelo treinado foi inferior ao modelo configurado manualmente. Por conta disso novos experimentos *offline* foram realizados afim de tentar solucionar os possíveis problemas que ocasionaram o desempenho pobre (cf. Capítulo 3).

No segundo teste, utilizando um modelo com um objetivo que levava em conta o viés posicional, e que era atualizado diariamente, o resultado foi positivo, evidência de que o principal problema era a definição de relevância utilizada e não a escolha de abordagem ou algoritmo de aprendizagem, que poderia ter sido qualquer um: LambdaMART, SVM ou RankNet.

Portanto, o uso de métodos de L2R pode trazer ganhos financeiros para sites de varejo eletrônico. No entanto, é necessário realizar mais experimentos para que fique claro se o ganho obtido se verificaria em lojas de outro setor por exemplo.

Capítulo 5

Conclusão

Este trabalho buscou utilizar métodos de L2R disponíveis na literatura em um contexto de comércio eletrônico. Os experimentos foram realizados utilizando-se testes do tipo A/B para comparar sistemas. Até onde se sabe, este foi o primeiro trabalho a reportar resultados da realização de testes A/B utilizando técnicas de L2R para uma solução de busca em comércio eletrônico.

Uma conclusão importante é que ganhos obtidos na avaliação de sistemas utilizando-se a abordagem tradicional de dividir a coleção em treino, validação e teste não necessariamente se traduzem em ganho quando se aplica a nova função de ordenação aprendida em um teste A/B. Por exemplo, verificou-se que utilizar esquemas de relevância que não levam em conta o fator de viés posicional em sua formulação é uma estratégia que leva a ganhos durante na abordagem de treino e teste, mas que não leva necessariamente a ganhos reais na hora de realizar um teste A/B. Por exemplo, no teste A/B realizado na loja de móveis o modelo sem viés posicional treinado foi inferior ao *baseline* construído por um time de especialistas, mesmo tendo sido superior nos experimentos de treino, validação e teste feitos com uma coleção montada a partir de consultas passadas, causando uma perda de $-4{,}7\%$ em rpv e $-2{,}26\%$ em conversão, enquanto os ganhos em NDCG foram em média de 13%.

Comparou-se também o resultado de diversos métodos de L2R em registros de lojas de comércio eletrônico, utilizando-se diferentes definições de relevância criadas a partir da intuição trazida pelos registros de interação dos usuários. Os modelos também foram comparados com um modelo configurado manualmente por um time de especialistas em busca para comércio eletrônico. Os modelos treinados geralmente se saíram melhores que o treinado pelo time de especialistas, em 8 das 10 configurações utilizadas na primeira rodada de experimentos. Entre os métodos de aprendizagem, nenhum se sobressaiu, com diferentes métodos se saindo melhor que os outros a depen-

der das configurações do experimento. Esse resultado é similar ao que se tem obtido na comparação entre métodos de L2R feita em trabalhos da literatura como Costa et al.[38] e Silva et al.[14].

Outra contribuição foi a adaptação e extensão de um modelo gráfico probabilístico de cliques usado em busca na web para busca em comércio eletrônico, considerando também eventos de compra, como uma tentativa de sanar o problema do viés posicional. De acordo com o resultado obtido no teste A/B realizado na loja de cosméticos, considerar o viés na formulação da definição de relevância tornou os modelos aprendidos melhores que os manuais, dando um ganho comercial real para a loja onde o teste foi realizado. Por exemplo, no experimento realizado, houve um aumento de 2,04% em RPV.

Realizou-se dois testes A/B durante o trabalho. O primeiro em uma loja de comércio eletrônico do ramo de móveis. A partir dos resultados dos primeiros experimentos escolheu-se o método SVM como o gerador de funções de ordenação do grupo experimental, por seu tempo de treino menor e a inteligibilidade dos modelos criados. Usou-se o esquema de relevância *Feedback* Normalizado. Objetivava-se verificar se o desempenho *offline* se traduziria em ganhos comerciais para a loja testada. O que ocorreu foi que o modelo treinado se saiu pior que o *baseline*, havendo uma piora no desempenho comercial no grupo que o utilizou, indicando que as métricas *offline* não traduziram bem o desempenho em ambiente de produção, causando perdas em número de vendas, já mencionadas. Acredita-se que os motivos para o desempenho inferior do modelo frente ao *baseline* foi a) o modelo não foi atualizado desde o início do teste. Ele deveria ter sido atualizado constantemente, conforme novos registros de usuários ficassem disponíveis. b) a definição de relevância utilizada para otimizar o modelo não levou em conta o viés posicional.

Executou-se uma segunda rodada de experimentos *offline*. Dessa vez utilizando um novo conceito de relevância, que leva em consideração o viés posicional em sua formulação. O novo esquema, o PS, foi adaptado a partir de um modelo gráfico probabilístico de cliques. Que passou a levar em conta eventos de compra. Ele foi comparado com novas definições de relevância, vindas de um trabalho na área de busca para comércio eletrônico, e com o *baseline* . Essas novas definições foram apresentadas em um dos únicos trabalhos similares a este. Nele, as definições são baseadas em taxas por visualização de cada produto para uma consulta. Novamente, verificou-se que os modelos treinados eram melhor avaliados que o *baseline*, por em média 6,9% no NDCG. Realizou-se dessa vez uma avaliação cruzada para verificar quão bem modelos treinados com diferentes definições de relevância se saem em conjuntos de teste com definições diferentes das dos conjuntos de treino. Percebeu-se uma distância significativa entre

o resultado da avaliação dos modelos da literatura e o PS, cerca de 20%, o que pode ser explicado pela fato do PS levar em conta o viés posicional diretamente em sua definição, ao contrário das outras definições.

No segundo teste A/B colocou-se essas hipóteses à prova. Dessa vez, o teste foi realizado em uma loja do ramo de cosméticos. De novo, utilizando-se o método SVM. Com a diferença que o SVM era atualizado diariamente, levando-se em conta os registros mais recentes dos usuários, e tendo-se um objetivo de treino diferente. Utilizado-se a Probabilidade de Satisfação. O resultado deste teste foi positivo, com o grupo experimental tendo um aumento estatisticamente significativo nas métricas de interesse, de 2,04% em RPV e 0,85% em conversão, acompanhando o resultado também positivo do NDCG nos experimentos *offline* de 14%.

Por fim, nos testes realizados não houve correlação direta entre os ganhos de NDCG obtidos nos testes *offline* e os ganhos de aumento de vendas obtidos nos testes online. Para ambos os testes A/B, os experimentos *offline* realizados antes de cada um indicavam que os modelos treinados são superiores aos configurados manualmente. Isso se verificou em apenas um dos testes A/B. Portanto, o desempenho obtido nos testes *offline*, da forma como têm sido realizados em geral na literatura, não está correlacionado necessariamente com a variação nas vendas obtidas na loja de comércio eletrônico ao utilizar o modelo num ambiente de produção.

5.1 Trabalhos futuros

Os resultados deste trabalho abrem possibilidades de estudo. Uma maneira de continuar o trabalho iniciado aqui é realizar novos testes A/B, procurando-se uma definição de relevância que se traduza no sucesso comercial da loja, sanando alguns pontos, como: 1) atualizar o modelo mais frequentemente, conforme novos dados fiquem disponíveis, seja várias vezes ao dia, seja de forma *online*, a partir de cada interação do usuário durante a sua sessão de busca. 2) Utilizar novas *features*. Ao contrário do *baseline* que é limitado pela disponibilidade de especialistas em comércio eletrônico, os métodos de L2R conseguem lidar com várias *features* se forma automática, graduando-se sua importância conforme a definição de relevância utilizada. 3) Realizar novos testes A/B, escolhendo lojas de departamentos diferentes dos já testados, afim de verificar se o comportamento observado é específico de um ramo e se alguns domínios são mais ou menos difíceis de obter resultados positivos.

Outra maneira de dar continuidade ao trabalho aqui iniciado seria explorar mais o uso de L2R. Utilizou-se neste trabalho 45 dias para o treino, validação e teste dos

modelos. Verificar se a utilização de mais dados resultaria necessariamente em melhor desempenho dos modelos. Como o ambiente de comércio eletrônico é rico em dados e informações das interações dos usuários, a criação de perfis de usuários, possibilitaria a sua segmentação, permitindo que cada usuário tivesse um resultado de busca conforme a sua necessidade e intenção. Poderia-se utilizar tais perfis durante o treino dos modelos, ou treinar diversos modelos, um para cada perfil de usuário, a cada momento. Ou seja, personalizar o resultado individualmente.

Apêndice A

Configuração de Métodos de learning to rank

A.0.1 Configuração de SVM

No quadro abaixo vê-se os parãmetros utilizados nos experimentos realizados neste trabalho, com o pacote sofia-ml.

```
sofia-ml
--iterations 1000000
--lambda C
--loop_type rank
```

Os outros parâmetros foram deixados com seus valores padrão.

Referências Bibliográficas

[1] Belkin, N. J.; Oddy, R. N. & Brooks, H. M. (1982). Ask for information retrieval: Part i. background and theory. *Journal of documentation*, 38(2):61--71.

[2] Bennett, P. N.; Radlinski, F.; White, R. W. & Yilmaz, E. (2011). Inferring and using location metadata to personalize web search. Em *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 135--144. ACM.

[3] Berberich, K.; Bedathur, S.; Alonso, O. & Weikum, G. (2010). A language modeling approach for temporal information needs. Em *European Conference on Information Retrieval*, pp. 13--25. Springer.

[4] Besser, J.; Larson, M. & Hofmann, K. (2010). Podcast search: User goals and retrieval technologies. *Online Information Review*, 34(3):395--419.

[5] Burges, C.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N. & Hullender, G. (2005). Learning to rank using gradient descent. Em *Proceedings of the 22nd international conference on Machine learning*, pp. 89--96. ACM.

[6] Burges, C.; Svore, K.; Bennett, P.; Pastusiak, A. & Wu, Q. (2011). Learning to rank using an ensemble of lambda-gradient models. Em *Proceedings of the learning to rank Challenge*, pp. 25--35.

[7] Cao, Z.; Qin, T.; Liu, T.-Y.; Tsai, M.-F. & Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. Em *Proceedings of the 24th international conference on Machine learning*, pp. 129--136. ACM.

[8] Chapelle, O. & Zhang, Y. (2009). A dynamic bayesian network click model for web search ranking. Em *Proceedings of the 18th international conference on World wide web*, pp. 1--10. ACM.

[9] Chuklin, A.; Markov, I. & Rijke, M. d. (2015). Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3):1--115.

[10] Chung, S. H.; Goswami, A.; Lee, H. & Hu, J. (2012). The impact of images on user clicks in product search. Em *Proceedings of the Twelfth International Workshop on Multimedia Data Mining*, pp. 25--33. ACM.

[11] Cossock, D. & Zhang, T. (2006). Subset ranking using regression. Em *International Conference on Computational Learning Theory*, pp. 605--619. Springer.

[12] Crammer, K.; Singer, Y. et al. (2001). Pranking with ranking. Em *Nips*, volume 1, pp. 641--647.

[13] Craswell, N.; Zoeter, O.; Taylor, M. & Ramsey, B. (2008). An Experimental Comparison of Click Position-Bias Models. Em *WSDM'08*.

[14] da Costa Carvalho, A. L.; Rossi, C.; de Moura, E. S.; da Silva, A. S. & Fernandes, D. (2012). Lepref: Learn to precompute evidence fusion for efficient query evaluation. *Journal of the American Society for Information Science and Technology*, 63(7):1383--1397.

[15] Donmez, P.; Svore, K. M. & Burges, C. J. (2009). On the local optimality of lambdarank. Em *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 460--467. ACM.

[16] Fuhr, N. (1989). Optimum polynomial retrieval functions based on the probability ranking principle. *ACM Transactions on Information Systems (TOIS)*, 7(3):183--204.

[17] Fürnkranz, J. & Hüllermeier, E. (2010). Preference learning: An introduction. Em *Preference learning*, pp. 1--17. Springer.

[18] Herbrich, R.; Graepel, T. & Obermayer, K. (1999). Support vector learning for ordinal regression. Em *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)*, volume 1, pp. 97--102. IET.

[19] Hiemstra, D. (1998). A linguistically motivated probabilistic model of information retrieval. Em *International Conference on Theory and Practice of Digital Libraries*, pp. 569--584. Springer.

[20] Hofmann, K.; Balog, K.; Bogers, T. & De Rijke, M. (2010). Contextual factors for finding similar experts. *Journal of the American Society for Information Science and Technology*, 61(5):994--1014.

[21] Hofmann, K.; Balog, K.; Bogers, T.; Rijke, M. d. et al. (2008). Integrating contextual factors into topic-centric retrieval models for finding similar experts.

[22] Järvelin, K. & Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422--446.

[23] Joachims, T. (2002). Optimizing search engines using clickthrough data. Em *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 133--142. ACM.

[24] Joachims, T.; Granka, L.; Pan, B.; Hembrooke, H. & Gay, G. (2005). Accurately Interpreting Clickthrough Data as Implicit Feedback.

[25] Karmaker Santu, S. K.; Sondhi, P. & Zhai, C. (2017). On application of learning to rank for e-commerce search. Em *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 475--484. ACM.

[26] Kohavi, R. & Longbotham, R. (2017). *Online Controlled Experiments and A/B Testing*, pp. 922--929. Springer US, Boston, MA.

[27] Li, H. (2014). Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 7(3):1--121.

[28] Liu, S.; Xiao, F.; Ou, W. & Si, L. (2017). Cascade ranking for operational e-commerce search. *arXiv preprint arXiv:1706.02093*.

[29] Liu, T.-Y. (2009). Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225--331.

[30] Matthijs, N. & Radlinski, F. (2011). Personalizing web search using long term browsing history. Em *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 25--34. ACM.

[31] Nallapati, R. (2004). Discriminative models for information retrieval. Em *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 64--71. ACM.

[32] Ponte, J. M. & Croft, W. B. (1998). A language modeling approach to information retrieval. Em *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 275--281. ACM.

[33] Robertson, S. E. (1977). The probability ranking principle in ir. *Journal of documentation*, 33(4):294--304.

[34] Salton, G. (1979). Mathematics and information retrieval. *Journal of Documentation*, 35(1):1--29.

[35] Sanderson, M. (2010). *Test collection based evaluation of information retrieval systems*. Now Publishers Inc.

[36] Sculley, D. (2009). Large scale learning to rank. Em *NIPS Workshop on Advances in Ranking*, pp. 58--63.

[37] Shen, X.; Tan, B. & Zhai, C. (2005). Context-sensitive information retrieval using implicit feedback. Em *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 43--50. ACM.

[38] Silva, T. P. C.; de Moura, E. S.; Cavalcanti, J. M. B.; da Silva, A. S.; de Carvalho, M. G. & Gonçalves, M. A. (2009). An evolutionary approach for combining different sources of evidence in search engines. *Information Systems*, 34(2):276--289.

[39] Spark-Jones, K.; Walker, S. & Robertson, S. (2000). A probabilistic model of information retrieval: Development and comparative experiments-part 1. *Information Processing and Management*, 36(6):779--840.

[40] Taylor, M.; Guiver, J.; Robertson, S. & Minka, T. (2008). Softrank: Optimising non-smooth rank metrics.

[41] Teevan, J.; Dumais, S. T. & Liebling, D. J. (2008). To personalize or not to personalize: modeling queries with variation in user intent. Em *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 163--170. ACM.

[42] Voorhees, E. M. (2001). The philosophy of information retrieval evaluation. Em *Workshop of the Cross-Language Evaluation Forum for European Languages*, pp. 355--370. Springer.

[43] Voorhees, E. M.; Harman, D. K. et al. (2005). *TREC: Experiment and evaluation in information retrieval*, volume 1. MIT press Cambridge.

[44] Wu, C.; Yan, M. & Si, L. (2017). Ensemble methods for personalized e-commerce search challenge at CIKM cup 2016. *CoRR*, abs/1708.04479.