C.E.S.A.R – CENTRO DE ESTUDOS E SISTEMAS AVANÇADOS DO RECIFE

Lucas Vieira Galvão

Sistema de Recomendação para Vendas Online

Recife
2021

Lucas Vieira Galvão

Sistema de Recomendação para Vendas Online

Monografia apresentada ao programa de Especialização em análise e engenharia de dados do Centro de Estudos e Sistemas Educacionais do Recife – C.E.S.A.R, como requisito para a obtenção do título de Especialista em análise e engenharia de dados

Orientação: Prof.(a). Felipe Silva Ferraz e Prof.(a). Gustavo Henrique Alexandre da Silva

Recife
2021

Resumo

O objetivo deste trabalho é desenvolver um sistema de recomendações para maximização das vendas online, utilizando cinco modelos diferentes para a construção desse sistema.

Dado a vasta gama de produtos oferecidos na Internet, muitas vezes o consumidor se sente sobrecarregado com a quantidade de opções. Seja para escolher que filme assistir em um serviço de *streaming*, ou para definir que produto comprar em um site de e-commerce, existem cada vez mais opções. Nesse contexto, sistemas de recomendações são importantes para melhorar a experiência do usuário e aumentar as chances de retorno para a plataforma.

Após o desenvolvimento dos modelos, os resultados obtidos foram comparados para determinar qual melhor se adequa ao problema. Inicialmente é definido o que é um sistema de recomendação, quais são suas contribuições para uma empresa e quais são os diferentes modelos. Após as definições iniciais, são comparados os resultados apresentados pelos modelos desenvolvidos para maximização de vendas.

Sumário

1 – Introdução **5**

2 – Contexto do Trabalho **6**

2.1 - Sistemas de Recomendação 6

2.1.1 - Métodos Baseados em Conteúdo 6

2.1.2 - Filtragem Colaborativa 7

2.1.3 - Híbrido 8

2.2 - Baseline Predictors 8

2.3 - Singular Value Decomposition (SVD) 8

2.3.1 - SVD++ 9

2.4 - Non-Negative Matrix Factorization (NMF) 10

2.5 - Normal Predictor 10

3 – Trabalho Realizado **11**

3.1 - Apresentação dos Dados 11

3.2 - Transformação da Base 13

3.3 - Desenvolvimento dos Modelos 14

4 - Resultados **15**

5 – Conclusão **21**

Referências Bibliográficas **22**

1 – Introdução

O advento da internet revolucionou a maneira como as pessoas consomem produtos e serviços, desde como assistem filmes e séries por aplicativos de streaming, até como pedem comida de um restaurante através de serviços de delivery. Com isso os mercados foram ampliados, empresas são capazes de atingir uma quantidade maior de potenciais clientes e os consumidores possuem mais opções de produtos e serviços disponíveis.

Porém essa ampliação do mercado também traz desafios, pois com o incremento de opções para os clientes, também aumenta a concorrência por essa clientela. Esse não é um problema só para os vendedores, mas também para os consumidores. Ter uma vasta gama de opções, apesar de aparentar ser algo positivo, traz consigo uma série de problemas. Cerca de 39% dos clientes deixaram de comprar em um site, e compraram em outro, pois se sentiram sobrecarregados com a quantidade de opções [1].

Para aprimorar essa experiência, são implementados sistemas de recomendação, segundo a mesma pesquisa, 75% dos clientes tem maior probabilidade de realizar uma compra se o site oferece um das seguintes funcionalidades:

- Reconhece o cliente pelo nome
- Fornece sugestões de produtos baseado em compras passadas
- Conhece o seu histórico de compras [1]

Na seção 2 deste trabalho é definido o que é um sistema de recomendação e qual a sua importância para uma empresa, também é apresentado alguns dos algoritmos utilizados para construção desses sistemas.

Na seção 3 são apresentados os modelos desenvolvidos, incluindo a base utilizada, quais modificações foram feitas nesta e quais algoritmos foram escolhidos.

Na seção 4 os resultados de cada modelo desenvolvido na seção 3 são comparados para entender como cada um se adequa ao problema proposto e definir qual tem maior aderência.

2 – Contexto do Trabalho

Sistemas de recomendação são algoritmos voltados para gerar sugestões de itens relevantes para o usuário. Nesta seção são abordados os usos de sistemas de recomendações, quais ganhos estão relacionados com esse uso e como construir um sistema de recomendação [2].

2.1 - Sistemas de Recomendação

Com o crescimento do e-commerce, a variedade de produtos ofertados para os consumidores também aumentou. Dada a grande quantidade de opções disponíveis, muitos clientes se sentem sobrecarregados e não sabem o que escolher. Para solucionar esse problema muitos comércios utilizam sistemas de recomendação.

Sistemas de recomendação são uma linha de defesa intuitiva contra o *over-choice* do consumidor, construindo uma experiência personalizada para atender as preferências de cada cliente [3]. Com isso, o usuário tem uma melhor experiência na plataforma, aumentando a fidelização.

Existem várias maneiras de se construir um sistema de recomendação, porém, de forma geral, os algoritmos de recomendação podem ser divididos em dois grupos: baseado em conteúdo (content-based) e filtragem colaborativa (collaborative filtering). Os modelos baseados em conteúdo utilizam o perfil do usuário e as características dos itens para fazer as recomendações. Já os modelos de filtragem colaborativa utilizam o histórico de atividades dos usuários para encontrar o item a ser recomendado [4].

2.1.1 - Métodos Baseados em Conteúdo

Métodos baseados em conteúdo criam perfis para usuários, com intuito de mapear os interesses destes. Os itens também são categorizados, baseados nas suas características, de forma que o modelo consiga relacionar um item com o outro. Com esses perfis mapeados, o sistema é capaz de associar usuários com produtos que atendam seus interesses [5].

O primeiro passo para um modelo baseado em conteúdo é criar uma representação dos produtos de forma que possa ser processada pelo algoritmo [6]. Esses dados são a base do modelo, é baseado neles que o perfil do usuário será traçado, porém, em muitos casos, essas informações não estão disponíveis ou são de difícil coleta [5].

2.1.2 - Filtragem Colaborativa

Filtragem colaborativa é um método que analisa a relação entre os usuários e as interdependências de itens para prever uma nova relação entre usuário e item [5].

Por exemplo, para recomendar um filme para um determinado usuário, o sistema de recomendação que utiliza filtragem colaborativa analisará o histórico desse usuário e buscará na sua base de dados outros usuários que tenham históricos semelhantes para encontrar qual filmes estes assistiram que ainda não foi visto pelo usuário. Outra maneira que o modelo pode usar para encontrar o filme a ser recomendado é analisando a relação entre filmes, por exemplo, se o usuário em questão assistiu o primeiro filme de uma trilogia, pelo histórico de atividades o modelo pode aferir que grande parte dos usuários, após assistirem o primeiro filme, assistem o segundo em seguida, com essa relação entre itens determinada, o sistema pode recomendar o próximo filme para usuário.

Para determinar os interesses dos usuários, o modelo de filtragem colaborativa precisa do *feedback* desse, sendo este de uma entre três categorias distintas: *input* explícito, *feedback* explícito e *feedback* implícito. *Input* explícito ocorre quando é solicitado ao usuário que preencha um questionário no seu primeiro acesso, dessa forma é construído um perfil provisório. *Feedback* explícito é uma avaliação direta do usuário após consumo, por exemplo, após uma compra em um site de e-commerce é solicitada uma avaliação do item obtido. Tanto o *input* explícito quanto o *feedback* explícito necessitam da participação ativa do usuário [7], assim o volume desses dados é limitado.

Feedback implícito ocorre quando dados são extraídos através da análise do comportamento do usuário, podendo ser os clicks em para um site de notícias ou que músicas escuta ou que filmes assistem para uma plataforma de streaming [8]. A quantidade de dados de *feedback* implícito é, na maioria dos casos, imensa. Porém apenas uma pequena parcela desses dados são relevantes para recomendações e precisam ser processadas antes do uso [9]. Além disso, o uso de *feedback* implícito traz outro desafio, esse tipo de dado, em muitos casos, só é capaz de registrar avaliações positivas, não levando em consideração *feedbacks* negativos [10].

Os modelos de filtragem colaborativa, em suas aplicações mais bem sucedidas, apresentam resultados melhores que os baseados em conteúdo, pois são capazes de aprender fatores latentes do modelo através de técnicas de fatoração de matriz aplicadas a matriz usuário-item, sem precisar de dados adicionais além do histórico de compras dos usuários [4].

Apesar de apresentar bons resultados, quando comparados com métodos baseados em conteúdo, modelos de filtragem colaborativa possuem duas grandes limitações. A

primeira é a escassez de dados, considerando a matriz usuário-item, em que cada linha representa um usuário e cada coluna um item, somente uma pequena porcentagem do total de elementos terão valores [11].

A segunda limitação é conhecida como *cold start problem*, como filtragem colaborativa utiliza o histórico de um usuário para encontrar outros usuários semelhantes, é necessário que exista uma grande quantidade de avaliações pelo usuário para que seja possível relacionar este com os demais, esses dados não estão disponíveis para novos usuários [11]. Esse problema também ocorre para itens novos.

2.1.3 - Híbrido

Para superar as limitações da filtragem colaborativa é necessário o uso de fontes adicionais de dados dos usuários e itens, conhecidos como "side information" [4], assim se popularizou o uso de modelos híbridos.

Modelos híbridos combinam métodos baseados em conteúdo e filtragem colaborativa para superar as limitações de ambos os modelos e obter resultados mais eficientes [11].

2.2 - Baseline Predictors

Os modelos de filtragem colaborativa buscam capturar as interações dos usuários com os itens, que produzem diferentes ratings. Porém muitos desses valores são devido a características associadas ao usuário ou ao item, de forma independente da interação entre eles [12]. Por exemplo, um certo cliente u tende a dar notas mais altas que a média, ou um produto i recebe avaliações piores que os demais.

Dessa forma um baseline predictor é um modelo simples que busca realizar recomendações utilizando essas tendências, ou *bias*, dos usuários e itens. Utilizando este algoritmo, o rating do produto i para o cliente u, pode ser encontrado pela fórmula [12]:

$$b_{ui} = \mu + b_u + b_i$$

Onde b_{ui} representa o rating de i para u, μé a média global dos ratings e b_u e b_i são o *bias* do usuário e do item, respectivamente.

2.3 - Singular Value Decomposition (SVD)

SVD é um método de fatoração de matriz utilizado para gerar aproximações de baixa característica. Dada uma matriz A, com dimensões m $\times$ n, onde a $\geq$ b, e característica r, SVD(A) será:

$$SVD(A) = U \times S \times V^T$$

Onde U, S e V possuem dimensões m × m, m × n e n × n, respectivamente. As matrizes U e V são ortogonais, com suas colunas sendo autovetores de AA^T e A^TA, respectivamente. A matriz S é uma matriz diagonal, que contém apenas r valores não nulos, os elementos dessa diagonal tem a seguinte propriedade $\sigma 1 \geq \sigma 2 \geq ... \geq \sigma r > 0$ [13].

Uma propriedade importante do SVD, é que este é capaz de fornecer a aproximação ótima da matriz original utilizando a multiplicação de três matrizes menores [13]. Através da remoção de entradas de menor impacto, o SVD pode manter m << r valores singulares, considerando que os valores de S estão ordenados, são mantidos apenas os m primeiros valores. As matrizes U e V são reduzidas para U_m e V_m, respectivamente, pela remoção de (r-m) colunas da matriz U e (r-m) linhas da matriz V [14]. Multiplicando esses valores temos:

$$A_m = U_m \times S_m \times V_m^T$$

Dessa forma, o rating r_{ui}, para o usuário u e item i, pode ser calculado da seguinte maneira:

$$r_{ui} = b_{ui} + q_i^T p_u$$

Onde, q_i é o vetor com os fatores do item i, p_u é o vetor com fatores do usuário u e b_{ui} é o baseline predictor, que pode ser definido como $\mu + b_u + b_i + q_i$, sendoμa média dos ratings da base, b_u e b_i são os bias do usuário e item, respectivamente [14].

A redução de dimensionalidade é muito valiosa para métodos de filtragem colaborativa, pois gera um conjunto de autovetores não relacionados, sendo cada usuário e item representado por um autovetor. Dessa forma, usuários que avaliaram produtos semelhantes, porém não os mesmo, são mapeados para o mesmo espaço [14].

2.3.1 - SVD++

Além do modelo inicial do SVD, também foram desenvolvidas modificações, buscando melhorar a performance do algoritmo, uma dessas é o SVD++. Este é uma adaptação que considera informações adicionais para a redução de dimensionalidade, sendo essas o feedback implícito [15].

Assim, utilizando o SVD++, o rating r_{ui}, para o usuário u e item i, é calculado pela fórmula:

$$r_{ui} = b_{ui} + q_{if}^{T}(p_{uf} + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_{if})$$

Sendo p_{uf} o vetor referente ao feedback explícito e $|N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_{if}$ representa o vetor do feedback implícito [15].

O SVD++ apresenta melhor performance de predição quando comparado com o SVD, porém o processo de treinamento tem maior custo computacional e o processo é mais demorado [15].

2.4 - Non-Negative Matrix Factorization (NMF)

Assim como os modelos apresentados acima, o NMF é um método de fatoração matricial, dessa forma seu objetivo é encontrar uma aproximação da matriz original, de forma que [16]:

$$R \simeq X \times V$$

O NMF difere dos demais métodos, pois esse aplica uma restrição de não negatividade nas aproximações *X* e *V*. Esta limitação é aplicada através da adaptação da taxa de aprendizado, que é redimensionada para cancelar os componentes negativos a cada atualização [17]. A vantagem desse método é que a aproximação é apenas baseada na adição de fatores, de forma que o algoritmo está aprendendo as partes do objeto [18].

2.5 - Normal Predictor

O normal predictor é um algoritmo que atribui um rating aleatório para a interação de um usuário com um item, baseado na distribuição da base de dados, presumindo que esta seja normal. O rating r_{ui} é gerado através da distribuição normal $N(\mu, \sigma^2)$ são estimados por máxima verossimilhança [19].

3 – Trabalho Realizado

Para a construção do sistema de recomendação foi utilizada a base *Retailrocket recommender system dataset* [20], contendo dados de eventos realizados por um usuário em um site de e-commerce, podendo o evento ser um entre visualização de produto, adição ao carrinho ou compra.

Utilizando algoritmos de filtragem colaborativa foram construídos e comparados cinco modelos distintos, são eles: Normal Predictor, Baseline Only, SVD, SVD++ e NMF. Foi utilizado a biblioteca *Surprise* para a construção dos modelos. O desenvolvimento está disponível em: https://github.com/lucas-galvao/TCC-SitemaRecomendacao

3.1 - Apresentação dos Dados

A base de dados utilizada é organizada de forma que cada linha representa um evento ocorrido no site, ao todo são 2.756.101 eventos, registrados no período de 05/03/2015 até 18/09/2015. Do total de eventos registrados, estes estão divididos da seguinte forma: 2.664.312 (96,7%) dos registros são visualizações, 69.332 (2,5%) são adições ao carrinho e apenas 22.457 (0,8%) são compras. Na base estão registrados 1.407.580 usuários distintos interagindo com 235.061 produtos diferentes.

Agrupando os dados por usuário, temos que em média, cada usuário visualizou 1,89 produtos, adicionou 0,05 ao carrinho e comprou 0,02. A partir dessas análises é concluído que apenas uma pequena quantidade de usuários estão comprando produtos.

Filtrando esta base, selecionando os clientes que compraram pelo menos uma vez, temos que o total de usuários compradores é 11.719, 0,8% do total registrado. Esses, por vez, foram agrupados em função da quantidade de vezes que compraram, segue abaixo o agrupamento.

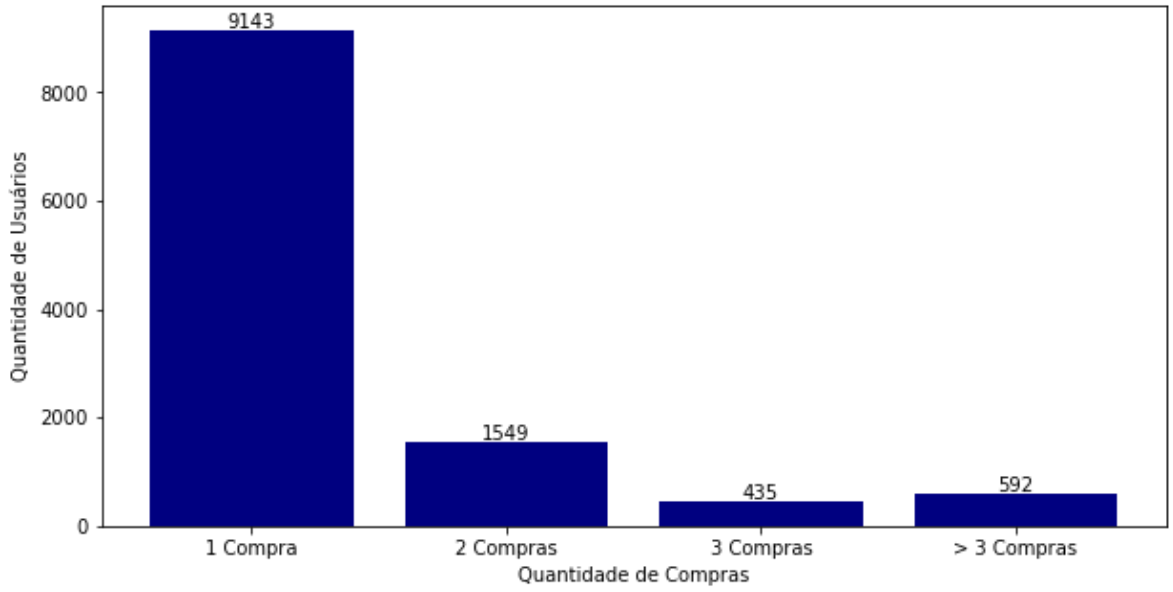

Figura 1 – Distribuição de usuários por volume de compras, fonte:O Autor

Baseado no gráfico acima, é possível concluir que a grande maioria, 78,02%, dos usuários que realizaram pelo menos uma compra, apenas compraram uma única vez no período.

Além da quantidade de usuários por grupo de compra, também foi analisado o comportamento desses usuários. O gráfico abaixo mostra a média de visualizações realizadas pelos clientes de cada faixa de compras:

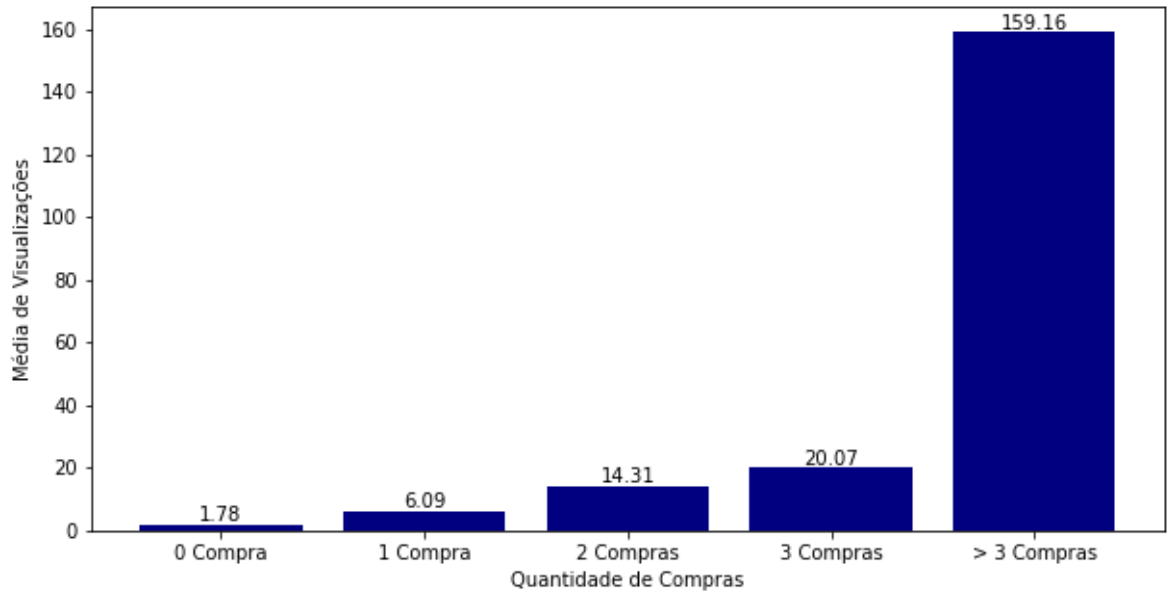

Figura 2 – Média de visualizações por volume de compras, fonte:O Autor

Analisando os dados, é possível inferir que a média de visualizações é proporcional à quantidade comprada. Temos que o coeficiente de correlação entre a coluna visualizações e a coluna compras é 0,78, indicando que o número de visualizações de um usuário poderá ser utilizado para inferir se este irá comprar um produto.

Também foi estudada a relação entre a quantidade de compras e a quantidade de adições ao carrinho, segue abaixo o gráfico descrevendo essa relação.

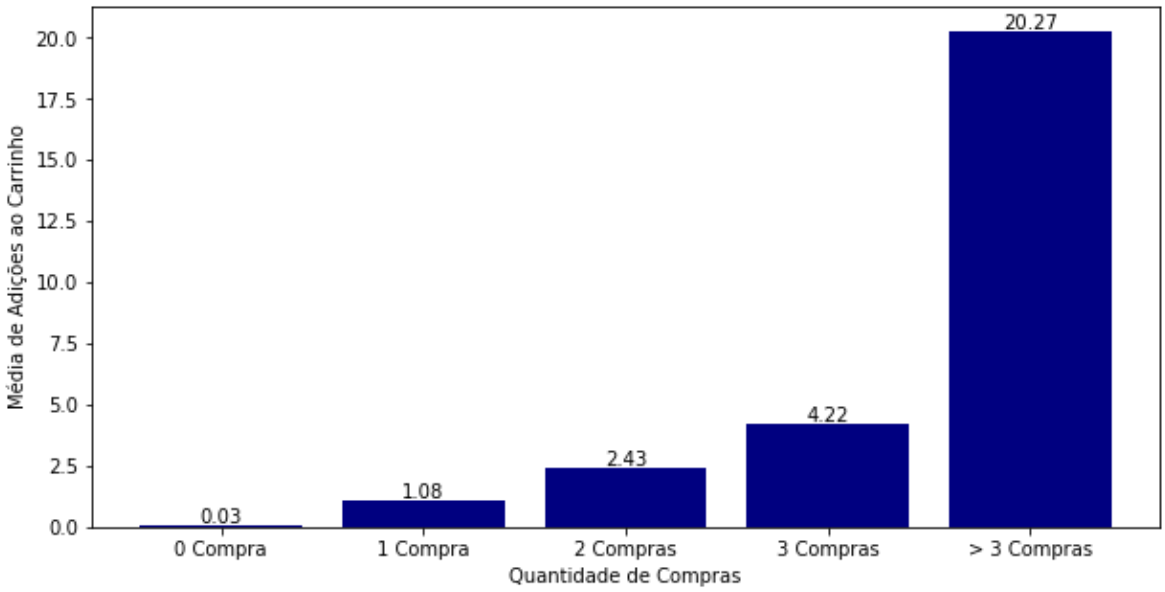

Figura 3 – Média de adições ao carrinho por volume de compras, fonte:O Autor

Temos que a quantidade de adições ao carrinho, similar a quantidade de visualizações, também é proporcional ao número de compras, porém os valores de adições ao carrinho são mais próximos do valor de compras, demonstrado pelo coeficiente de correlação que é 0,9.

3.2 - Transformação da Base

A base de dados utilizada oferece um problema, como cada linha dessa representa um evento, podemos ter vários casos do mesmo usuário interagindo com o mesmo item, resultando em mais de uma linha para cada iteração. Para a construção dos modelos, os dados foram agrupados em função dos valores de usuário e item, de forma que, dado um usuário u e um item i, a relação u-i aparece em apenas uma linha.

Com esse agrupamento, os valores de cada evento foram somados, de forma que a linha representando a relação u-i contém a quantidade de vezes que u visualizou i, quantas vezes foi adicionado ao carrinho e o total de compras. Como os modelos aceitam apenas três entradas, sendo elas: ids dos usuários, ids dos items e rating, é necessário reduzir o total de eventos para um único valor. Para encontrar a transformação que mais se adequa ao problema, foram construídas três versões da base, variando o cálculo do rating para cada uma.

Na primeira versão foi utilizado um rating binário, indicando se o cliente comprou ou não o produto, essa transformação utiliza apenas os valores do evento sale, descartando os demais. Para o cálculo do rating na segunda variação foi definido que o valor desse dependerá do evento de maior impacto, sendo compra o maior, seguido por adição ao carrinho e por último visualização. Caso o usuário tenha comprado o item, seu rating será 1, se não ocorreu compra, mas o produto foi adicionado ao carrinho, o valor será 0,9, se nenhum dos eventos citados for registrado, o rating será 0,7. Para a definição dos pesos dos eventos adição ao carrinho e visualização foram utilizados aproximações dos valores das correlações.

Para a última versão, assim como na segunda, o valor do rating será definido pelo evento de maior impacto, a diferença será nos valores, estes serão categorizados de acordo com a jornada do cliente. No caso, se este apenas visualizar o produto, o rating terá valor 1, se o item foi adicionado ao carrinho, o valor será 2 e caso a compra seja realizada o rating será 3.

3.3 - Desenvolvimento dos Modelos

Cada versão da base foi utilizada para construção de modelos de recomendação, ao todo foram criados cinco modelos para cada variação dos dados, sendo esses: Normal Predictor, Baseline Only, SVD, SVD++ e NMF.

4 - Resultados

Para definir qual modelo melhor se adequa ao problema em questão foram utilizadas métricas para compará-los. Essas métricas foram: RMSE, MAE e hit rate at k.

O RMSE (erro médio quadrático) e o MAE (erro médio absoluto) são medidas do erro entre o valor real do rating na base de teste e o valor calculado pelo modelo. Essas métricas se diferem no cálculo, o RMSE, como o nome indica, é o valor da raiz quadrada da soma das diferenças entre o real e o previsto elevado ao quadrado dividido pela quantidade de observações, como demonstrado na fórmula abaixo:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\widehat{y_i} - y_i)^2}{n}}$$

Onde y_i é o valor do rating para o elemento i, $\widehat{y_i}$ é o rating calculado pelo modelo para i e n é o total de elementos.

De forma similar, o MAE calcula a média dos erros entre o valor calculado e o valor real, porém esse cálculo utiliza o valor absoluto dos erros, representado abaixo:

$$MAE = \frac{\sum_{i=1}^{n} \left|\widehat{y_i} - y_i\right|}{n}$$

O hit rate at k consiste em gerar recomendações para o usuário usando todos os itens da base, ou uma parcela destes, e selecionar as k principais. Com as k recomendações em mãos, é verificado se o produto comprado pelo usuário contido na base de teste se encontra entre os selecionados, caso sim é adicionado 1 ao valor do hit, caso contrário o valor não é alterado. Ao fim o valor do hit é dividido pelo total de usuários.

Para a base em questão foi necessário uma adaptação no cálculo do hit rate, como muitos usuários não realizaram compras, apenas visualizaram produtos ou adicionaram esses ao carrinho, foram considerados apenas os clientes compradores para o cálculo dessa métrica. Outra adaptação foi o uso de uma parcela dos itens para gerar as recomendações, considerando que a base possui 235.061 produtos distintos, utilizar a base completa para o cálculo do hit rate teria um custo computacional muito grande, então as recomendações foram calculadas para os produtos que usuário interagiu na base de testes mais duzentos produtos selecionados aleatoriamente dentre o total contido nos dados, excluindo os que o usuário interagiu.

Considerando que o volume de compras por usuário é baixo, com a maioria dos compradores adquirindo apenas um item, o valor de k escolhido deve refletir essa característica da base, por isso foram considerados 5 e 10 como k.

O modelo NMF apresentou erros quando executado com a base binário, então as métricas desse estão com valores zerados.

Segue abaixo os valores encontrados para as métricas selecionadas, onde binário representa a primeira versão da base, pesos a segunda e categorias a terceira.

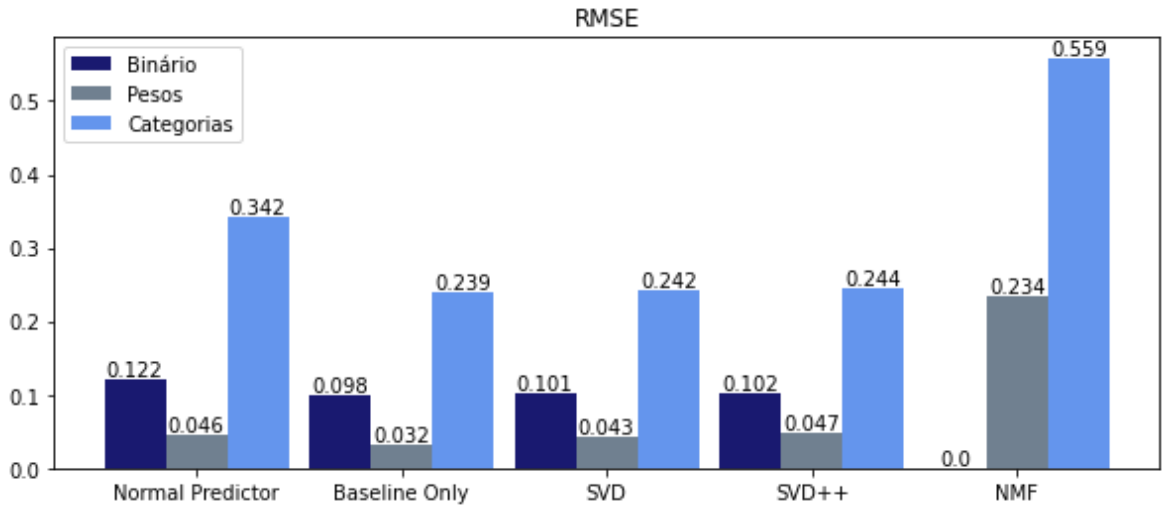

Figura 4 – RMSE, fonte:O Autor

O RMSE apresentou resultados bons para a maioria dos casos, tendo valores abaixo de 0,12 para quase todos os modelos considerando as bases binário e pesos, sendo o Normal Predictor e o NMF os únicos com valor acima. Os resultados para a base categorias foram mais elevados, essa diferença é devido a faixa dos ratings, para essa os valores são entre 0 e 3, já para as demais são entre 0 e 1, logo o erro quando se prever que o usuário não irá comprar um produto quando esta transação ocorreu, terá um peso maior na base categoria, o que causa o aumento no RMSE.

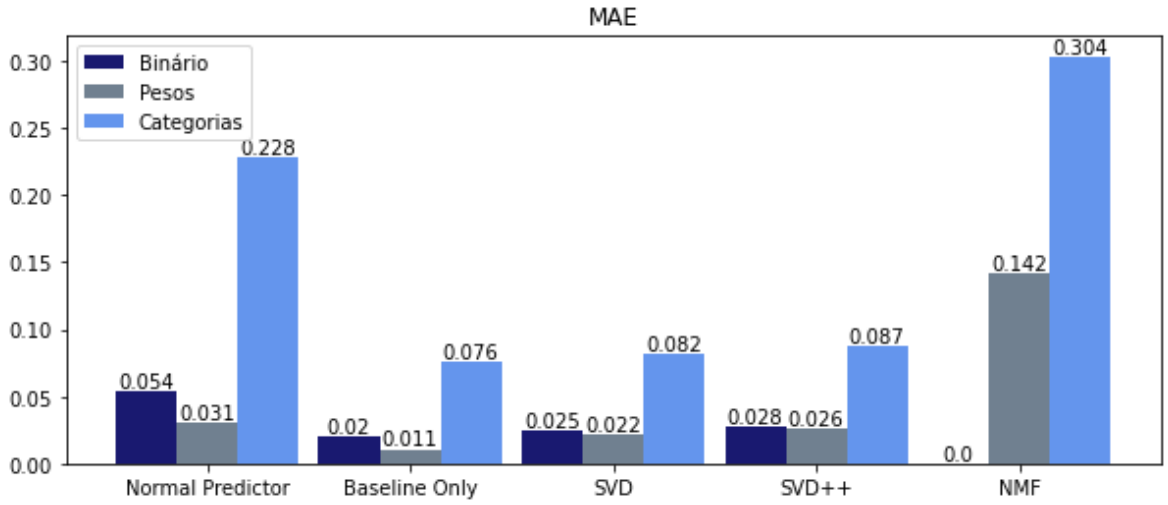

Figura 5 – MAE, fonte:O Autor

Os valores do MAE estão distribuídos de forma semelhante ao RMSE, os valores para as bases binário e pesos tiveram valores baixos, e a base categorias teve resultados mais elevados.

Considerando a distribuição da base, com a grande maioria dos eventos sendo visualizações (96,7%) e uma pequena parcela desses são compras (0,8%), a média dos ratings para as bases são muito mais próximas dos menores valores, associados a visualização, logo, algoritmos que utilizam este valor como principal indicador para o cálculo do rating, terão métricas de erro baixas, pois, para a grande maioria dos casos, o valor real será muito próximo da média, resultando em um erro pequeno, como é o caso para o Normal Predictor e Baseline Only. Da mesma forma, algoritmos que realizam a redução de dimensionalidade buscando reduzir as métricas de erro, como o SVD, SVD++ e NMF, podem otimizar esse resultado com valores próximos do evento visualização, pois os casos em que o cliente comprou o produto são tão pouco que o impacto do erro de não recomendar esse item não será expressivo nas métricas.

Além disso, como o menor valor observado na base de treinamento é 0,7, referente ao peso do evento visualização, reduzindo as medidas de erro, já que a média será muito próxima desse valor e o máximo possível é 1.

Por esses motivos a base categorias apresenta resultados mais elevados para os algoritmos utilizados. Para investigar esse comportamento, as métricas de erro foram calculadas considerando apenas os casos de venda, segue abaixo os resultados:

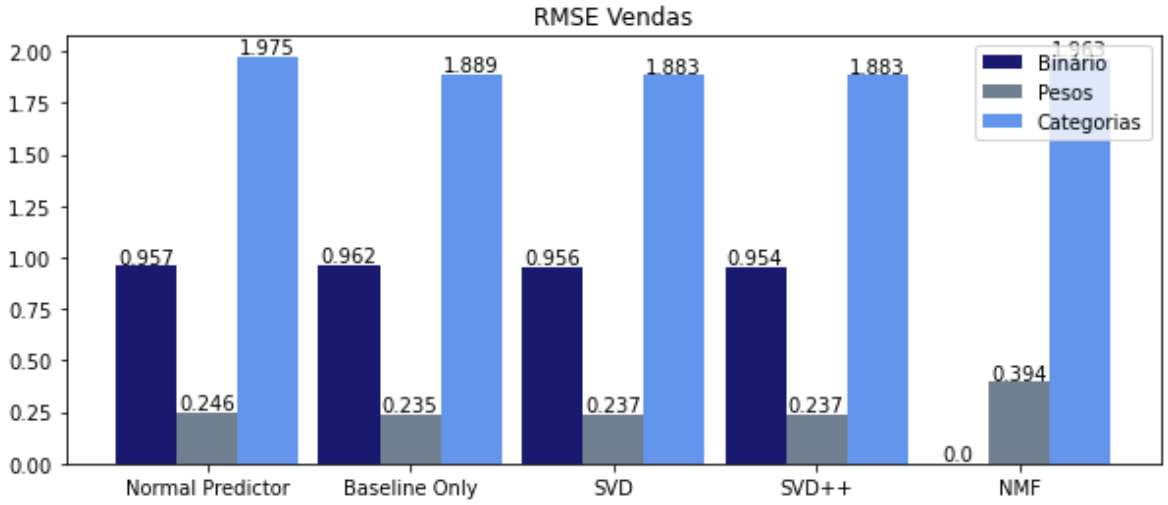

Figura 6 – RMSE Vendas, fonte:O Autor

Analisando o gráfico acima fica claro o comportamento descrito acima, considerando apenas os casos em que ocorreu a venda, o valor do RMSE fica acima de 0,95 para todos os algoritmos na base binário, exceto o NMF por questões de erro, sendo 1 o valor máximo para essa métrica. A base pesos possui resultados menores, como descrito anteriormente, o que afetará os algoritmos Normal Predictor e Baseline Only, pois seus cálculos dependem principalmente da média das observações. O NMF também será afetado pela média, pois a sua aplicação sofre do problema do cold start, de forma que se for avaliar um usuário ou produto que não tenha encontrado na base de treinamento, o valor do rating será a média. Para a base categorias o erro fica ainda mais elevado, pois para essa base o evento visualização possui rating 1, sendo 3 o valor para a compra, como observado anteriormente.

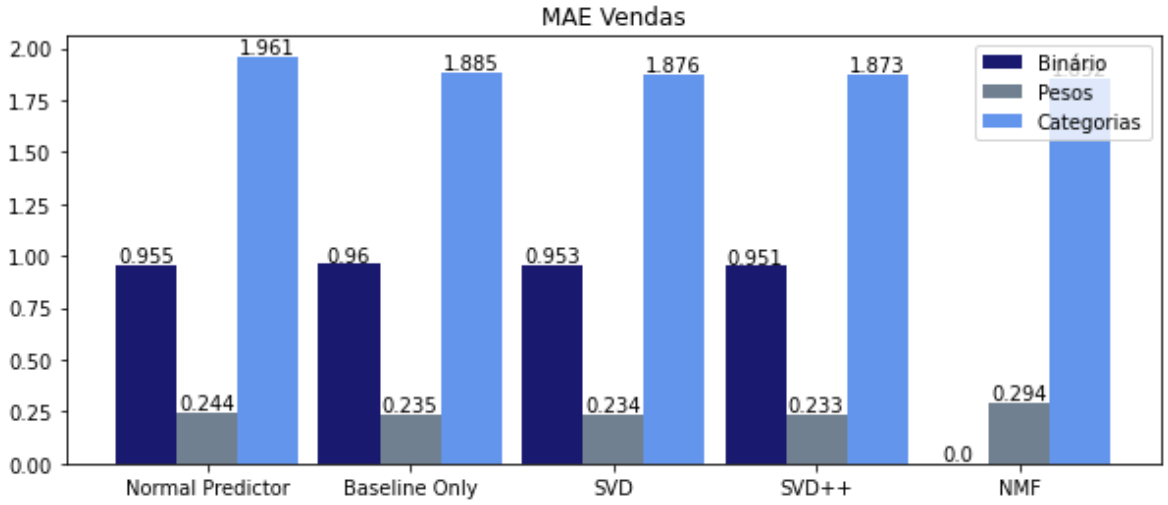

Figura 7 – MAE Vendas, fonte:O Autor

Os resultados da métrica MAE, novamente, são similares aos do RMSE, evidenciando os pontos levantados. Da mesma forma, o desbalanceamento da base causa problemas para os modelos, tendo em vista que estes dependem da média dos valores para seus cálculos ou buscam reduzir as medidas de erro.

Segue abaixo os resultados obtidos com o cálculo do hit rate, medida que indicará se os algoritmos são capazes de gerar recomendações relevantes para os clientes.

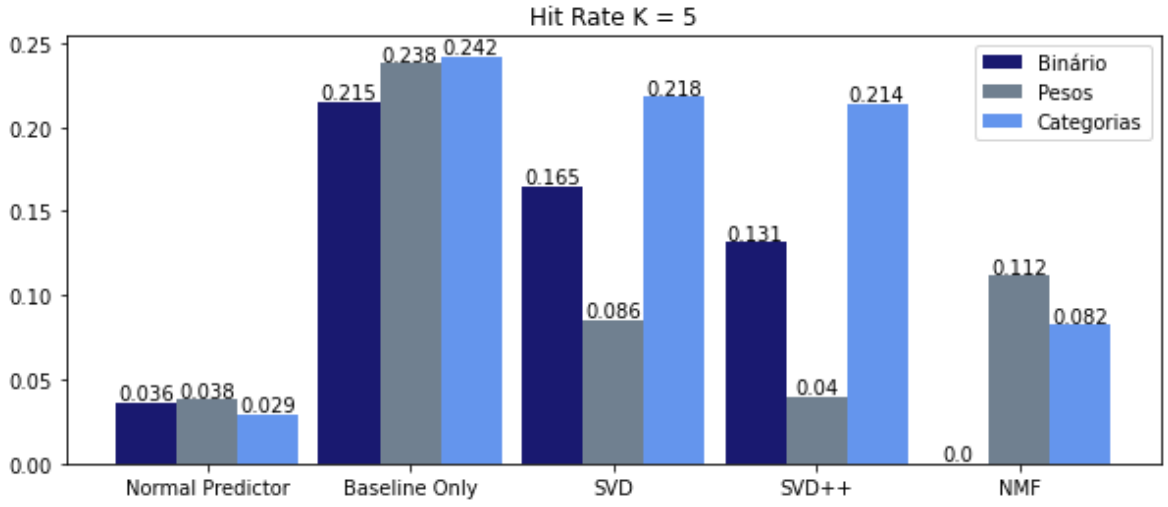

Figura 8 – Hit Rate K = 5, fonte:O Autor

Analisando os resultados do hit rate é evidente que os modelos não estão gerando recomendações adequadas para os clientes, sendo o Baseline Only o único que alcançou um valor superior a 22%, com 24,2%, para base categorias, sendo o melhor.

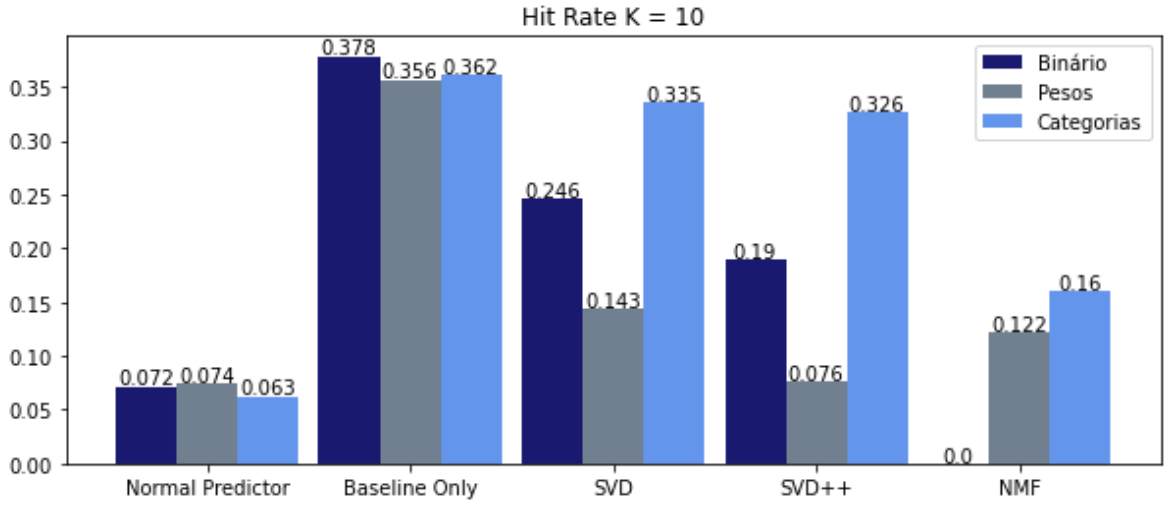

Figura 9 – Hit Rate K = 10, fonte:O Autor

Utilizando k = 10 os resultados melhoraram para a maioria dos modelos, com o Baseline Only novamente obtendo o maior valor com 37,8% para a base binário. Esse modelo teve esse resultado superior aos demais, pois cerca de 41% dos produtos envolvidos em uma compra na base de testes também foram comprados na base de treinamento, considerando que os componentes para o cálculo do rating para esse modelo são a média dos ratings e as tendências de usuários e itens, o valor do último tem um grande impacto nas recomendações, em especial na base categorias que os eventos compra tem um peso maior que nas demais.

Diferente dos produtos, apenas 3,8% dos usuários compradores da base de testes também realizaram uma compra na base de treinamento. Essa diferença entre as bases, além dos problemas de desbalanceamento, interferem na performance dos algoritmos que buscam encontrar semelhanças entre os usuários para gerar recomendações.

5 – Conclusão

Mediante aos resultados obtidos para os modelos construídos, conclui-se que nenhum se adaptou adequadamente ao problema, sendo o principal motivo a escassez de eventos de compra na base utilizada. Por causa do desbalanceamento entre os dados, os modelos se especializaram em prever casos de não compra, como ficou evidenciado pelas métricas RMSE e MAE. Além disso, os modelos não foram capazes de gerar recomendações satisfatórias, hipótese comprovada pela métrica hit rate at k, com maior valor sendo 37,8% para k = 10.

Entre as versões de cálculo do rating utilizadas os resultados variam a depender do modelo, considerando as diferenças devido a amplitude dos ratings, as métricas de erro são similares entre as bases, sem grandes diferenças. Porém o hit rate apresentou variações significativas para cada modelo, por exemplo os modelos SVD e SVD++ tiveram seus melhores resultados com a base categorias e o pior desempenho para a base pesos, a principal diferença entre os ratings dessas versões, com categorias tendo a maior amplitude e pesos a menor, dessa a não recomendação de uma venda observada tem um impacto maior nas métricas de erro, medida que esses algoritmos buscando otimizar no processo de treinamento. Por outro lado, algoritmos que utilizam a média das observações como principal componente para a predição, como Baseline Only e Normal Predictor, apresentaram hit rates próximos para as três bases utilizadas.

Com um volume maior de compras e a proporção desses casos aumentando, os modelos seriam forçados a identificar corretamente os casos de venda para otimizar as métricas de erro, sendo esse o objetivo do SVD, SVD++ e NMF. Da maneira que a base atual está constituída, esses algoritmos são construídos com o intuito de identificar casos em que o cliente não compra o produto, ignorando os casos em que a venda ocorre, pois o erro associado a esses eventos é mínimo comparado a alternativa.

Referências Bibliográficas

[1] Consumers Welcome Personalized Offerings but Businesses Are Struggling to Deliver, Finds Accenture Interactive Personalization Research - Online, acesso em 08/04/2021 na url https://newsroom.accenture.com/news/consumers-welcome-personalized-offerings-but-businesses-are-struggling-to-deliver-finds-accenture-interactive-personalization-research.htm

[2] Introduction to recommender systems - Online, acesso em 28/04/2021 na url https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

[3] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay, **"Deep Learning based Recommender System: A Survey and New Perspectives".** (2018) ACM Comput. Surv. 1, 1, Article 1

[4] Dong, Xin, L. Yu, Zhonghuo Wu, Yuxia Sun, L. Yuan and Fangxi Zhang. **"A Hybrid Collaborative Filtering Model with Deep Structure for Recommender Systems"**. (2017) AAAI.

[5] Hu, Yifan & Koren, Yehuda & Volinsky, Chris., **"Collaborative Filtering for Implicit Feedback Datasets. Proceedings".** (2008) IEEE International Conference on Data Mining, ICDM. 263-272. 10.1109/ICDM.2008.22.

[6] Zisopoulos, Charilaos & Karagiannidis, Savvas & Demirtsoglou, Georgios & Antaris, Stefanos, **"Content-Based Recommendation Systems".** (2008)

[7] Dooms, S., De Pessemier, T., & Martens, L., **"An online evaluation of explicit feedback mechanisms for recommender systems"**. (2011) 7th International Conference on Web Information Systems and Technologies (WEBIST-2011) (pp. 391-394). Ghent University, Department of Information technology.

[8] Jeunen, Olivier**, "Revisiting Offline Evaluation for Implicit-Feedback Recommender Systems"**. (2019) 10.1145/3298689.3347069.

[9] Lee, T. Q., Park, Y., & Park, Y. T., **"A time-based approach to effective recommender systems using implicit feedback"**. (2008) Expert systems with applications, 34(4), 3055-3062.

[10] Baltrunas, Linas & Amatriain, Xavier, "**Towards time-dependant recommendation based on implicit feedback"**. (2009) Proceedings of the Third ACM Conference on Recommender Systems

[11] Wei, Jian & He, Jianhua & Chen, Kai & Zhou, Yi & Tang, Zuoyin, **"Collaborative Filtering and Deep Learning Based Recommendation System For Cold Start Items"**. (2016) Expert Systems with Applications. 69. 10.1016/j.eswa.2016.09.040

[12] Koren, Y. **"The bellkor solution to the netflix grand prize"**. (2009) Netflix prize documentation, 81, 1-10.

[13] Zhou, Xun & He, Jing & Huang, Guangyan & Zhang, Yanchun, **"SVD-based incremental approaches**

[14] Kumar, Rajeev and Verma, BK and Rastogi, Shyam Sunder, "**Social popularity based SVD++ recommender system"**. (2014) International Journal of Computer Applications. 87

[15] J. Jiao, X. Zhang, F. Li and Y. Wang, **"A Novel Learning Rate Function and Its Application on the SVD++ Recommendation Algorithm"**. (2020) in IEEE Access, vol. 8, pp. 14112-14122

[16] Zhang, F., Lu, Y., Chen, J., Liu, S., & Ling, Z., **"Robust collaborative filtering based on non-negative matrix factorization and R 1 -norm"**. (2017) Knowledge-Based Systems, 118, 177–190.

[17] Xin Luo, Mengchu Zhou, Yunni Xia, & Qingsheng Zhu, **"An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems"**. (2014) IEEE Transactions on Industrial Informatics, 10(2), 1273–1284.

[18] Lee, D. D., & Seung, H. S., **"Learning the parts of objects by non-negative matrix factorization"**. (1999) Nature, 401(6755), 788–791.

[19] Basic algorithms — Surprise 1 documentation - Online, acesso em 30/04/2021 na url https://surprise.readthedocs.io/en/stable/basic_algorithms.html

[20] Retailrocket recommender system dataset | Kaggle - Online, acesso em 24/04/2021 na url https://www.kaggle.com/retailrocket/ecommerce-dataset