

**FACULDADE METROPOLITANA DA GRANDE FORTALEZA**

**CURSO DE SISTEMAS DE INFORMAÇÃO**

**ANTONIO JACKSON DOS SANTOS PASSOS**

**FILTRAGEM COLABORATIVA DE SISTEMA DE RECOMENDAÇÃO – Uma abordagem unificada de Interface Homem-Máquina com Machine Learning usando Apache Mahout**

FORTALEZA - CE

2017

ANTONIO JACKSON DOS SANTOS PASSOS

**FILTRAGEM COLABORATIVA DE SISTEMA DE RECOMENDAÇÃO – Uma abordagem unificada de Interface Homem-Máquina com Machine Learning usando Apache Mahout**

Artigo científico apresentado ao curso de Sistema de Informação da Faculdade Metropolitana da Grande Fortaleza – FAMETRO – como requisito para obtenção do grau de bacharel, sob a orientação do profº Francisco Nauber Bernardo Gois

Fortaleza - CE

2017

ANTONIO JACKSON DOS SANTOS PASSOS

**FILTRAGEM COLABORATIVA DE SISTEMA DE RECOMENDAÇÃO – Uma abordagem unificada de Interface Homem-Máquina com Machine Learning usando Apache Mahout**

Este artigo científico foi apresentado no dia 00 de mês de 2017 como requisito para obtenção do grau de bacharel em Sistemas de Informação da Faculdade Metropolitana da Grande Fortaleza – FAMETRO – tendo sido aprovada pela banca examinadora composta pelos professores abaixo:

BANCA EXAMINADORA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Profº. M.e Francisco Nauber Bernardo Gois

Orientador – Faculdade Metropolitana da Grande Fortaleza

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Profº. M.e Daniel Almeida Chagas

Doutorando e Mestre em Informática Aplicada

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Profº M.e Júlio Alcântara Tavares

Membro – Faculdade Metropolitana da Grande Fortaleza

**AGRADECIMENTOS**

A Deus, por acreditar que nossa existência pressupõe outra infinitamente superior, dando-me o discernimento e sabedoria para alcançar meus objetivos.

Aos familiares e amigos, que com muito carinho e apoio, contribuíram de forma indireta e diretamente para que eu chegasse a esta etapa de minha vida.

Ao Coordenador, Júlio Alcântara Tavares, e todos os professores do Curso de Sistema de Informação da Fametro, em especial ao Doutorando Nauber Gois, pelo convívio, pelo apoio, por aceitar me orientar neste artigo de fundamental importância em minha vida acadêmica, pela compreensão e pela amizade.

O melhor resultado acontece quando todos em um grupo fazem o melhor por si próprios e pelo grupo.

John Nash

**FILTRAGEM COLABORATIVA DE SISTEMA DE RECOMENDAÇÃO – Uma abordagem unificada de Interface Homem-Máquina com Machine Learning usando Apache Mahout**

**Antonio Jackson dos Santos Passos1**

**RESUMO**

Este artigo refere-se a como ajudar as pessoas a viverem melhor, economizando tempo e elevando o nível de satisfação da experiência do usuário, auxiliando-o na tomada de decisão em seu cotidiano com acesso a sistemas de recomendações, bem como associar esse tipo de sistema com a performance da interação humano-computador e seus principais benefícios no nível de satisfação.

Para fundamentar a problematização deste documento, foi escolhido a filtragem colaborativa em sistemas de recomendações usando a plataforma apache mahout, um ambiente para criação rápida de aplicações escaláveis e automáticas para machine learning, com implementação na linguagem java, onde o mesmo busca criar recomendações para um usuário alvo, baseado em experiências e declarações correlacionadas de outros usuários de mesma similaridade com o usuário alvo. Ao fim da execução desse tipo de sistema, é possível avaliar o grau de importância no impacto da experiência do usuário, bem como as estratégias de mercado para explorar.

Objetiva-se embasar a implementação da simulação de um sistema de recomendação de professores para alunos do curso de sistema de informação da Faculdade Metropolitana da Grande Fortaleza – Fametro, através de suas similaridades, usando a biblioteca apache mahout.

Palavras-chave: Filtragem Colaborativa. Aprendizado de máquina. Inteligência Artificial. Sistema de Recomendação. Mahout.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

¹ Discente do curso de Sistemas de Informação da Faculdade Metropolitana da Grande Fortaleza (FAMETRO).

# 1. INTRODUÇÃO

A partir de pesquisas militares no auge da Guerra Fria, na década de 1960, surge a rede mundial dos computadores, ou internet. Desde então, com os avanços tecnológicos, surge o princípio da era da informação, porém pode-se afirmar que já vivemos a chamada, era da recomendação (FERREIRA, 2007); Nesse momento, as exigências de mercado provém de usuários e consumidores que julgarão a qualidade e satisfação da pesquisa feita pelos mesmos em sites de busca na internet, como por exemplo o Google¹, que facilita diversas ações no cotidiano das pessoas que consequentemente encontram uma diversidade muito grande de conteúdo, dando origem a um fenômeno chamado sobrecarga de informação (KIELGAST et.al, 1995).

No mundo pós-internet surge a necessidade de criação de novas ferramentas que auxiliem os usuários e projetistas de sistemas, a nortear a navegação para o destino procurado sem causar frustações ou arrependimentos. Em mecanismos de busca, usuários precisam ter uma ideia mínima do que deseja encontrar (*“push” de informação)*, porém essa nova área de pesquisa despertou interesse de pesquisadores independentes em meados de 1990 surgindo assim os Sistemas de Recomendação (SR´s) que trabalham com a perspectiva de descoberta, onde a informação é servida ao usuário (“*pull” de informação),* possibilitando-o sensações inesperadas e experiência personalizada.

O tempo é cada vez mais precioso para pessoas do século XXI, que buscam otimizar o tempo de suas atividades cotidianas e consequentemente deparam-se com o paradoxo da escolha (SCHWARTZ et.al, 2007), gerando ansiedade e confusão, dificultando a escolha da melhor opção. Perguntas tais como: Como investir meu dinheiro para garantir um futuro melhor para meus filhos? Qual o melhor destino de férias? Que filme posso escolher para assistir com meus amigos? Que livro comprar? Que curso devo escolher para minha carreira? Enfim, a todo instante é preciso efetuar decisões, buscando sempre a melhor escolha, aquela que satisfaça o usuário, daí encontra-se a fundamentação para utilização dos SR´s. Tal descoberta, almeja permitir ao usuário desse sistema, ter contato com uma gama de informações não convencionais porém com uma grande relevância para o indivíduo. Por exemplo, ao fazer uma pesquisa na internet, os resultados obtidos seguem o requisito de popularidade, mas nada garante que essa característica seja relevante para o usuário, que dificilmente irá ter acesso às informações menos populares na rede.

Sistemas de recomendação são amplamente usados por lojas virtuais, sites de filmes e livros e todo tipo de comércio eletrônico na web bem como redes sociais com recomendações de pessoas, amizades, grupos de trabalho, empregos entre outros. O primeiro web site a utilizar o SR em grandes proporções foi o My Yahoo, em julho de 1996 (MANBER et.al, 2000) Atualmente, um grande número de websites detém de vantagens dos SR´s, utilizando estratégias de customização para seus clientes como por exemplo, a Amazon.com, Netflix, iTunes Store, Youtube, Facebook, Twitter e Linkedin, tendo por finalidade realizar predições quanto à preferência dos seus usuários, aumentando suas margens de vendas e criando um diferencial competitivo no mercado em relação aos concorrentes.

Para fazer recomendações, o que é preciso é encontrar pessoas com preferências e gostos parecidos e cruzar as escolhas para fazer recomendações baseando-se nessas informações. Não se resume apenas a mostrar mais do que você já gosta, mas oferecer a melhor experiência com base em quem aquela pessoa é. Ou seja, as recomendações devem ser similares, mas não iguais. (Antonio Molins - Engenheiro espanhol que atuou no sistema de recomendação da NetFlix)

Atualmente existem três técnicas principais para implementação de um sistema de recomendação. A primeira é baseada em filtragem de conteúdo, que utiliza a correlação entre o perfil do usuário, sendo necessário adquirir dados pessoais do mesmo, e os itens recomendados. A segunda, baseada na filtragem colaborativa, analisa a classificação prévia dos utilizadores para estabelecer uma associação entre itens e usuários, que ao contrário da primeira, poderá recomendar itens de difícil acesso ou mesmo indisponíveis. A terceira, também denominada de filtragem híbrida, é o resultado da combinação de duas ou mais técnicas com objetivo de otimizar seus desempenhos. A utilização de SR tem sido uma abordagem eficiente para reduzir o esforço do usuário em encontrar informações de seu interesse.

Este documento foi estruturado da seguinte forma: na seção 2, começa-se por introduzir os principais conceitos dos sistemas de recomendação, fundamentando a problematização; na seção 3, apresenta-se a implementação do sistema utilizando a biblioteca mahout com dados reais; na seção 4, mostra-se os resultados da aplicação com uma pesquisa experimental qualitativa e por fim na seção 5, aborda-se a conclusão do trabalho.

# 2. SISTEMAS DE RECOMENDAÇÃO E SEUS CONCEITOS

Esta seção aborda os principais conceitos mais implementados em sistemas de recomendação (SR), bem como a função adotada para possibilitar a recomendação ao usuário. Ao fim da mesma é fundamentada a problematização e as técnicas existentes para classificar a informação.

## 2.1 Introdução aos Sistemas de Recomendação

A predição das preferências do usuário ou clientes, gerou uma disputa muito ampla para a maioria dos web sites na rede, significando um diferencial competitivo e agregando valor considerável para ambas as partes do processo. Isso é possível por causa da utilização dos sistemas de recomendação que tem como principal objetivo analisar dados dos utilizadores, extraindo informações preciosas e relevantes, para recomendar itens, produtos, filmes, músicas, pessoas, carreiras profissionais, cursos, profissões, viagens, enfim tudo que é possível recomendar para uma pessoa.

Para realizar predições precisas, esse tipo de sistema utiliza diversas técnicas baseadas em algoritmos da inteligência artificial chamados de *machine learning* ou aprendizado de máquina. Estes podem levar em consideração dois principais aspectos de análise dos dados: o perfil de cada usuário ou o item que deseja recomendar. Dentre estas considerações, destaca-se três principais técnicas de filtragem sendo elas de conteúdo, colaborativa e híbrida das quais adiante neste documento é explanado suas peculiaridades.

Pelo fato desse tipo sistema auxiliar empresas do comércio *online* a estabelecerem uma relação íntima e personalizada para seus clientes, é que se explica a popularidade e crescente interesse de acadêmicos em incentivar pesquisas na área a fim de melhorias dos algoritmos e suas precisões, tanto é que o início do termo, Sistema de Recomendação ou *Recommender Systems,* surge em meados dos anos 1990 como área de pesquisa independente com a criação do sistema *Tapestry*, que foi o primeiro sistema de recomendação comercial, onde classificava como “bom” ou “mal” as anotações feitas pelos usuários de seus próprios e-mails. Este sistema utilizava algoritmos de colaboração e de conteúdo, tendo dessa forma diversas desvantagens como por exemplo, a falta de capacidade de agrupamento de usuários por interesses similares, pois o mesmo não teria um método capaz de tal feito (GOLDBERG et.al, 1992).

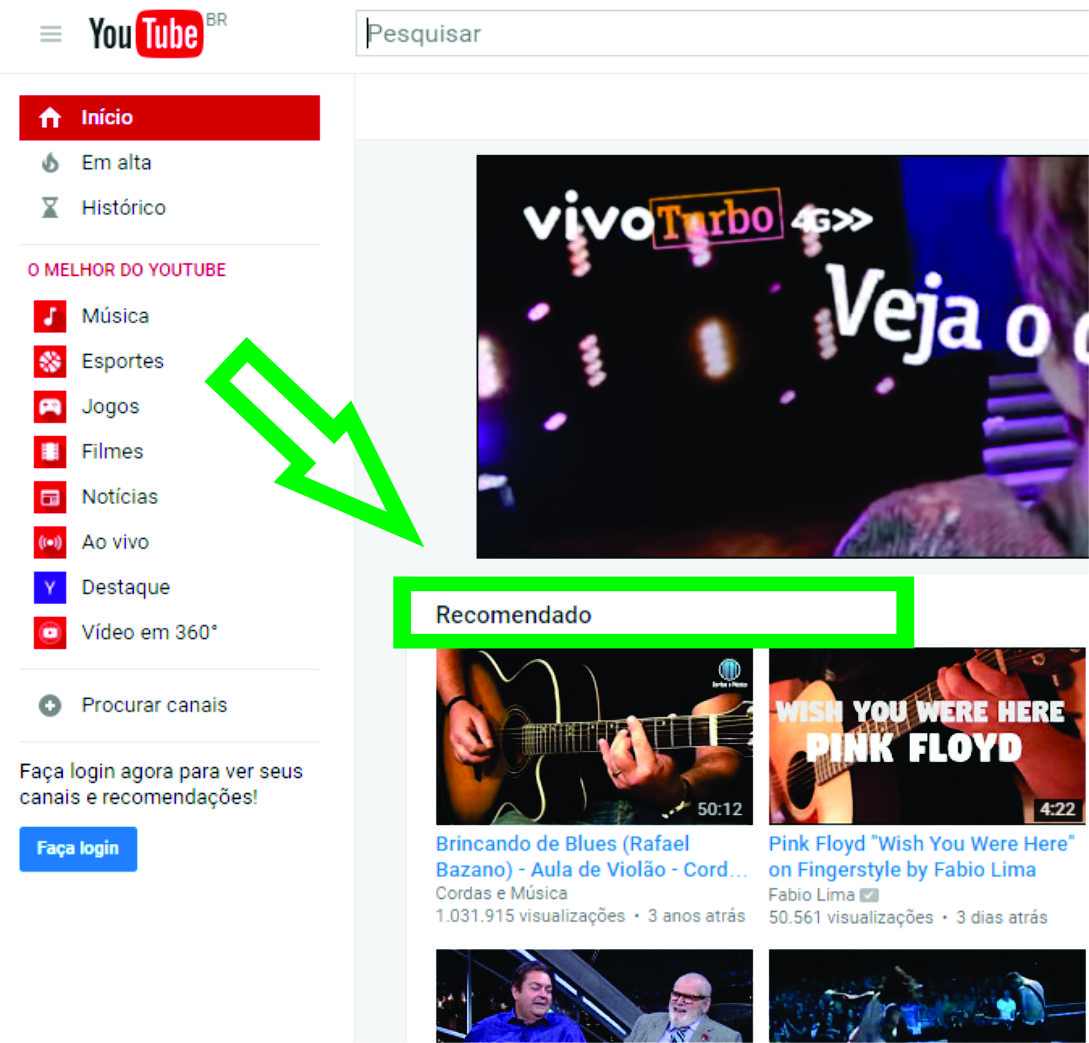
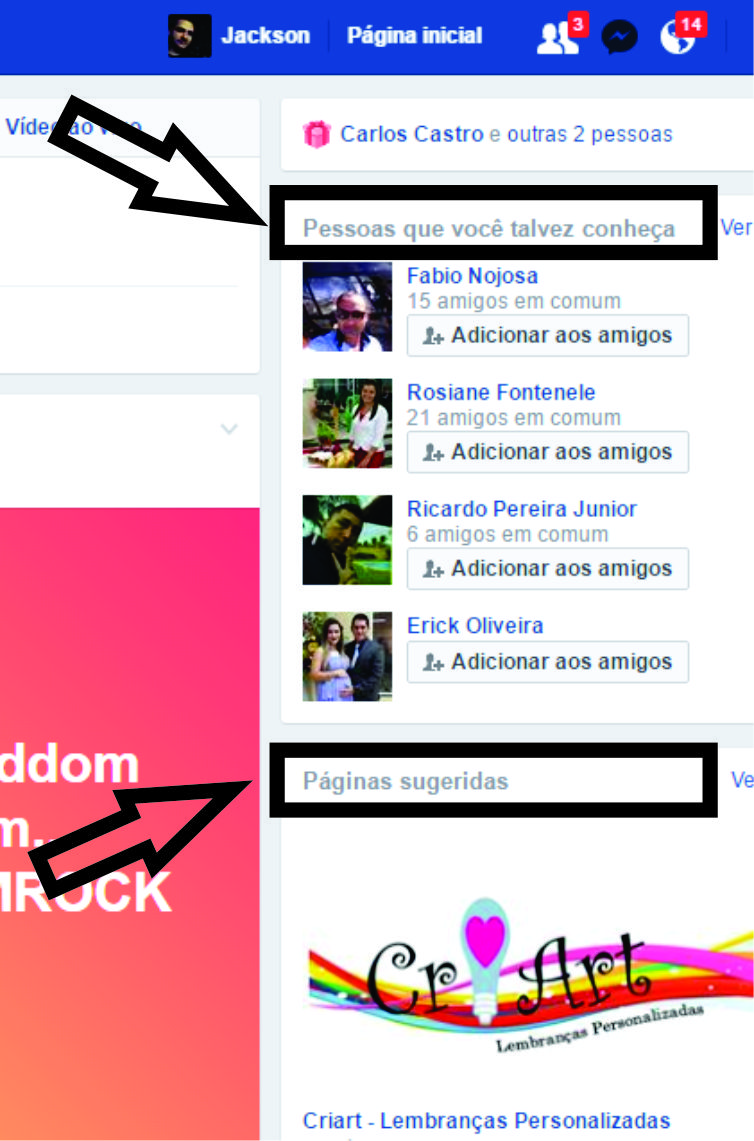
Com o crescimento perceptível de entretenimento e redes sociais na rede mundial de computadores, é mais fácil identificar no dia-a-dia o sistema de recomendação na prática como é o caso das imagens a seguir, que servem como reforço para o entendimento da aplicação dos SR´s. Na figura 1, é representado como o *Youtube.com* recomenda vídeos para o usuário com base em categorias que o mesmo acessa ou que pessoas similares a ele também acessam. De forma semelhante trabalha o *Facebook.com*, apenas com a diferença que ao invés de recomendar vídeos, este o faz com pessoas, sugestões de novas amizades, ou até mesmo páginas web como mostra a figura 2.

Figura 1 (esquerda) – Exemplo de recomendação feita pelo serviço do *Youtube.com;*

Figura 2 (direita) – Exemplo de recomendações da página do *Facebook.com*

## 2.2 Definição da Filtragem Colaborativa

Para fundamentação da filtragem colaborativa, é preciso analisar melhor o problema geral dos sistemas de recomendação, considerando 𝐔 como o conjunto de todos os utilizadores de um sistema, e 𝐈 como o conjunto de todos os itens passivos de recomendações, tais como filmes, pessoas, ou músicas. Com o propósito de possibilitar uma recomendação, é considerado 𝑟 como a função medidora da utilidade/relevância de um item para um determinado usuário, representada por: 𝑟:𝑼×𝑰 →𝑹, onde 𝑹 é o conjunto real (MARCELINO, 2014).

Com esses conceitos embasados, pode-se entender melhor a forma utilizada pela técnica de recomendação por filtragem colaborativa que foi projetada para ter a capacidade de recomendar itens indisponíveis ou de difícil acesso aos usuários, pois toma como base a classificação prévia desses itens por outros usuários cujo perfil é similar ao do usuário ativo do qual se quer efetuar alguma recomendação. A tabela 1 apresenta o funcionamento da filtragem colaborativa, com um exemplo fictício de recomendação de um produto para o usuário Nauber; para isso procura-se outros usuários com gostos semelhantes, no caso, Daniel e Júlio já compraram produtos que Nauber também comprou (PROD2). Logo após recomenda-se a Nauber produtos que estes dois usuários (Daniel e Júlio) possuem, porém que Nauber ainda não adquiriu como o PROD1 e PROD5. O resultado final da recomendação toma como base o histórico de avaliações comuns entre os usuários e o valor de predição calculado.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USUÁRIO** | **PROD1** | **PROD2** | **PROD3** | **PROD4** | **PROD5** | **PROD6** |
| DANIEL |  | **X** |  |  | **X** |  |
| JÚLIO | **X** | **X** |  |  |  |  |
| ANELINE |  |  | X | X | X |  |
| JACKSON |  |  | X |  |  |  |
| CONCEIÇÃO | X |  |  | X |  |  |
| NAUBER | **?** | **X** |  |  | **?** |  |

**Tabela 1 – Recomendação baseada em filtragem colaborativa. Fonte: elaborada pelo autor.**

A técnica de Filtragem colaborativa segue três passos: (1) calcular a similaridade ao usuários alvo; (2) selecionar os vizinhos mais próximos para considerar a predição; (3) ponderar as avaliações dos vizinhos com seus pesos. Esta técnica é conhecida como “*k-nearest-neighbor*” ou “*user-based*”. Existem alguns coeficientes de similaridade como *Coseno, Pearson*, entre outros. A equação a seguir mostra o coeficiente de *Pearson*, pois é amplamente aplicado em literaturas de sistemas de recomendação (CAZELLA et.al, 2005), sendo a função utilizada pela biblioteca mahout, que será implementada mais à frente.

onde é a correlação do usuário ***a***alvo com um usuário ***b***; e são avaliações atribuídas a um item de cada usuário ***a*** e ***b*** respectivamente; e são as médias de todas as avaliações dos usuários a e b respectivamente; É importante destacar que é necessária mais de uma avaliação em comum para que a recomendação seja bem efetivada (CAZELLA et.al, 2015). A média ponderada das avaliações dos vizinhos com coeficiente de similaridade desejado, passado por parâmetro, resulta na predição de um item para o usuário alvo.

## 2.3 Classificação da Informação

Existem dois modos de extrair, do usuário, as informações necessárias para realizar uma recomendação: explicitamente e implicitamente (PAPAGELIS et.al, 2005). Estes métodos acabam provendo a discussão da questão da privacidade do usuário que pode sentir-se invadido e enfadado dependendo a abordagem utilizada. A seguir é explicado as duas técnicas de mineração das informações.

### **2.3.1 Classificação Explícita**

A obtenção dos dados, realizada de forma explícita, pode ser feita através de questionários ou da classificação por intervalo fixo (exemplo: de 1 a 5). Essa é a forma na qual o usuário optam por expor suas recomendações de um determinado item, de forma natural, por vontade própria. Esse tipo de abordagem é o que melhor se adequa aos sistemas de filtragem colaborativa (PAPAGELIS et.al, 2005), porém exige do usuário uma participação ativa no sistema onde quanto mais ele interagir, mais o sistema vai conhecê-lo, fator que nem sempre é visto de forma positiva pelos mesmos. Para minimizar o impacto negativo ao usuário, são utilizadas estratégias de atribuições de pontos para adquirir produtos grátis, ou participação de sorteios para acesso a conteúdo privilegiado (EKSTRAND, 2010).

### **2.3.2 Classificação Implícita**

Esse tipo de abordagem vem suprir o problema do usuário não querer classificar ou recomendar nada. A estratégia infere os gostos dos usuários por meio de histórico da interação do mesmo com o sistema, como por exemplo, histórico de compras, movimento e cliques de mouses, pesquisas realizadas, entre outros. Porém para essa estratégia há um risco grande de imprecisão, pelo fato dessas ações poderem não ser fiéis ao desejo do utilizador (OARD et.al, 1998), por exemplo, o fato de alguém clicar em uma notícia, não significa dizer que vá ler, ou o fato de fazer uma pesquisa, não quer dizer que represente o interesse real daquele utilizador, ele pode estar realizando pesquisas para outra pessoa.

# 3. IMPLEMENTAÇÃO DE UM RECOMENDADOR USANDO DADOS REAIS NO MAHOUT

Nesta seção será demonstrado como desenvolver uma simples aplicação de um sistema de recomendação por filtragem colaborativa baseada no usuário, que irá recomendar professores para alunos do curso de graduação de sistema de informação da Faculdade Metropolitana da Grande Fortaleza – Fametro. Logo abaixo está a lista de ferramentas utilizadas no projeto recomendador:

* *Eclipse Java EE IDE for Web Developers, versão Mars Release (4.5.0)* (ECLIPSE, 2017);
* *Apache-maven versão 3.5.0* (MAVEN, 2017).
* *Java versão 1.8.0\_131* (ORACLE, 2017).
* *Sistema Operacional Windows 10 Pro 64bits*.

O mahout é uma biblioteca que tem por objetivo criar rapidamente um ambiente de aplicativos de aprendizagem de máquina, ou *machine learning*, com performance escalável, oferecendo três principais características: a primeira é um ambiente e estrutura de programação simples e reutilizável; a segunda com uma grande variedade de algoritmos já implementados pronto para uso; e a última o Samsara, ambiente de experimentação de matemática vetorial com sintaxe R-like que funciona em escala. (APACHE MAHOUT, 2017).

Para o bom funcionamento da aplicação com mahout, é preciso obter os dados de entrada com a extensão .csv, ou seja, arquivos de valores separados por vírgula com três valores, onde o primeiro número representa o usuário, o segundo número, o item, e o terceiro, a nota que o usuário associou para o item, criando dessa maneira uma matriz de valores com estrutura semelhante a: *IDusuário,IDitem,Notaitem*.

Foi realizado uma enquete não oficial, através do aplicativo whatsapp, por meio de dois grupos de alunos, a maioria destes são de turmas formadas a partir de 2014.1 do corrente curso. Foi informado o objetivo da pesquisa e esclarecido a finalidade da mesma, deixando evidente para os participantes que os dados são apenas demonstrativos e servirão somente para fundamentação.

Através do site institucional da faculdade (FAMETRO, 2017) obteve-se a lista de docentes de 2017.1 e criada uma tabela com ID´s dos professores e para cada um foi atribuída a nota que representa o quanto o aluno tem afinidade com aquele professor. Como a pesquisa é de cunho experimental, o tratamento dos dados submeteu-se a sanitização dos dados, a fim de privar identidades de ambas as partes envolvidas no processo.

A pergunta consiste em saber que nota de relevância você (*IDaluno*) avalia para um determinado professor (*IDprofessor*). Ao fim da pesquisa obteve-se os seguintes dados abaixo que foram salvos no arquivo chamado ***ProfNotasFametro.csv***:

|  |
| --- |
| 1,1,9.0 |
| 1,2,8.0 |
| 1,3,7.0 |
| 1,4,9.0 |
| 1,5,9.0 |
| 1,6,7.0 |
| 1,7,7.0 |
| 1,8,10.0 |
| 1,9,5.0 |
| 1,10,6.0 |
| 1,11,10.0 |
| 1,12,5.0 |
| 1,13,8.0 |
| 1,14,8.0 |
| 1,15,8.0 |
| 1,16,7.0 |
| 1,17,10.0 |
| 1,18,9.0 |
| 1,19,9.0 |
| 2,1,10.0 |
| 2,2,8.0 |
| 2,3,7.0 |
| 2,4,9.0 |
| 2,5,9.0 |
| 2,6,10.0 |
| 2,7,7.0 |
| 2,8,10.0 |
| 2,9,5.0 |
| 2,10,6.0 |
| 2,11,10.0 |
| 2,12,5.0 |
| 2,13,8.0 |
| 2,14,8.0 |
| 2,15,8.0 |
| 2,16,7.0 |
| 2,17,10.0 |
| 2,18,9.0 |
| 2,19,9.0 |
| 3,2,8.0 |
| 3,3,7.0 |
| 3,8,10.0 |
| 3,9,0.0 |
| 3,11,9.0 |
| 3,12,10.0 |
| 3,15,9.0 |
| 3,17,10.0 |
| 3,18,6.0 |
| 4,1,9.0 |
| 4,2,8.0 |
| 4,4,8.0 |
| 4,5,9.0 |
| 4,8,10.0 |
| 4,9,4.0 |
| 4,11,10.0 |
| 4,12,5.0 |
| 4,17,9.0 |
| 4,18,6.0 |
| 4,19,9,0 |
| 5,2,9.0 |
| 5,3,1.0 |
| 5,4,8.0 |
| 5,8,10.0 |
| 5,9,2.0 |
| 5,11,10.0 |
| 5,12,1.0 |
| 5,14,6.0 |
| 5,17,9.0 |
| 5,18,5.0 |
| 6,2,9.0 |
| 6,5,9.0 |
| 6,8,10.0 |
| 6,9,4.0 |
| 6,11,10.0 |
| 6,14,6.0 |
| 6,17,9.0 |
| 6,18,6.0 |
| 6,19,8.0 |
| 7,1,8.0 |
| 7,4,8.0 |
| 7,8,10.0 |
| 7,9,5.0 |
| 7,11,10.0 |
| 7,14,7.0 |
| 7,17,10.0 |
| 7,18,7.0 |
| 7,19,8.0 |
| 8,1,9.0 |
| 8,2,9.0 |
| 8,3,6.0 |
| 8,4,8.0 |
| 8,5,10.0 |
| 8,8,10.0 |
| 8,9,1.0 |
| 8,11,10.0 |
| 8,12,5.0 |
| 8,14,2.0 |
| 8,15,10.0 |
| 8,17,10.0 |
| 8,18,7.0 |
| 8,19,10 |
| 9,1,10.0 |
| 9,9,8.0 |
| 9,10,8.0 |
| 9,11,9.0 |
| 9,17,8.0 |
| 9,19,10 |
| 10,1,10.0 |
| 10,9,5.0 |
| 10,11,9.0 |
| 10,14,7.0 |
| 10,15,4.0 |
| 10,17,8.0 |
| 10,18,6.0 |
| 10,19,8.0 |
| 11,1,9.0 |
| 11,3,7.0 |
| 11,10,7.0 |
| 11,12,8.0 |
| 11,15,10.0 |
| 11,16,0.0 |
| 11,17,9.0 |
| 11,19,8.0 |
| 12,1,10.0 |
| 12,8,10.0 |
| 12,9,6.0 |
| 12,10,8.0 |
| 12,11,10.0 |
| 12,14,8.0 |
| 12,15,5.0 |
| 12,17,10.0 |
| 12,18,6.0 |
| 12,19,10.0 |
| 13,1,8.0 |
| 13,2,9.0 |
| 13,3,4.0 |
| 13,4,9.0 |
| 13,5,10.0 |
| 13,6,5.0 |
| 13,7,5.0 |
| 13,8,10.0 |
| 13,9,0.0 |
| 13,10,5.0 |
| 13,11,10.0 |
| 13,12,4.0 |
| 13,13,9.0 |
| 13,14,4.0 |
| 13,15,4.0 |
| 13,16,5.0 |
| 13,17,10.0 |
| 13,18,8.0 |
| 13,19,9.0 |

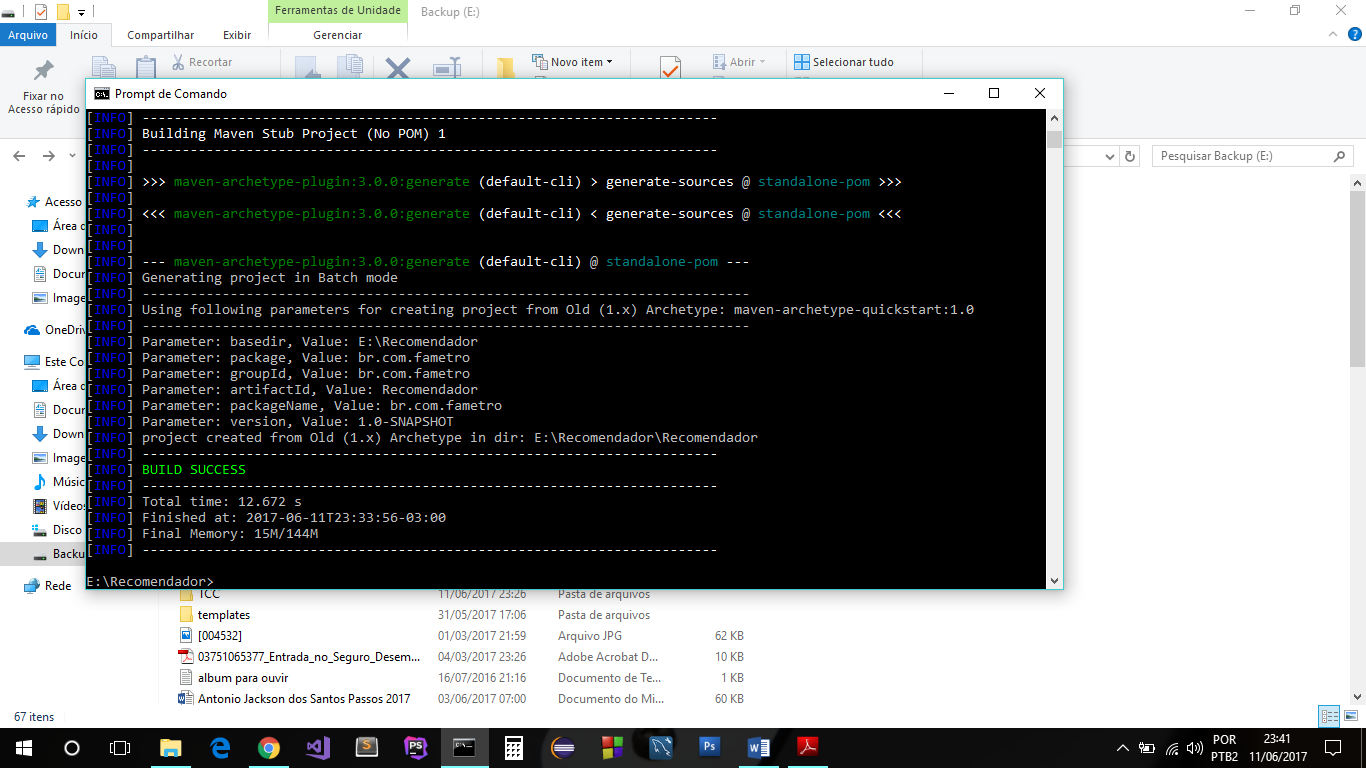
Cuja estrutura representa o ***IDaluno,IDProfessor,NotaProfessor***.

**3.1 Preparação do ambiente de desenvolvimento**

A implementação poderá ser feita em seu IDE java favorito, porém neste projeto utilizou-se o eclipse mars. Após realizado o download e instalação das ferramentas necessárias, cria-se um projeto simples com o Maven através do terminal do sistema operacional. O nome do projeto será Recomendador e o pacote principal será br.com.fametro. Para isso é necessário verificar se o maven está no classpath do sistema operacional, para a partir dessa condição criar um quickstart, ou seja, um projeto rápido do maven, sendo necessário navegar pelo terminal do sistema até a estrutura de pasta do projeto escolhido e dentro da mesma, executa-se o código.

mvn archetype:generate –DgroupId=br.com.fametro –DartifactId=Recomendador –DarchetypeArtifactId=maven-archetype-quickstart –DinteractiveMode=false

A execução deverá obter como resultado uma mensagem de *Build Success* como mostra a imagem a seguir.



Agora com o projeto criado pelo maven, é necessário realizar o import no eclipse, *file>import>Existing Maven Projects>,* informando o caminho do diretório onde anteriormente foi criado o projeto. Em seguida declara-se as dependências do mahout no projeto, editando o arquivo pom.xml, adicionando a linha de código a seguir.

<dependency>

<groupId>org.apache.mahout</groupId>

<artifactId>mahout-mr</artifactId>

<version>0.12.2</version>

</dependency>

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-client</artifactId>

<version>2.7.2</version>

</dependency>

Por padrão o eclipse não possui uma atualização de todas as bibliotecas existentes, por isso é necessário navegar até as preferências e alterar essa opção na IDE *Window>Preferences>Maven,* marcando a opção *Download repository index updates on startup*. Em seguida, apaga-se o arquivo App.java dentro de src/main/java br.com.fametro e fecha-se a IDE e inicia-se novamente para que, ao iniciar, ela faça o download necessário de todas as bibliotecas do projeto. Esta etapa poderá demorar um pouco, porém ao final da atualização o maven será capaz de usar as funções necessárias para criar um recomendador com o mahout.

**3.2 Criando um recomendador de professores**

O primeiro passo é colocar o arquivo .csv que contém a estrutura dos dados coletados na pesquisa, dentro do diretório src do projeto. Em seguida será necessário ler este arquivo criando um modelo de dados DataModel model = new FileDataModel(file) e implementar três funções simples. A primeira para calcular a similaridade entre os alunos com a função PearsonCorrelationSimilarity. A segunda é necessária para informar a distância dos vizinhos mais próximos ThresholdUserNeighborhood. Por fim cria-se o recomendador baseado nos alunos com a função GenericUserBasedRecommender, que receberá por parâmetro o modelo de dados, os vizinhos mais próximos e a similaridade dos alunos. Esta implementação está descrita na implementação da classe RecomendadorBuilder.java, porém antes foi criada a classe RecomendaProfessor.java dentro do diretório br.com.fametro do projeto.

**public** **class** RecomendaProfessor {

**public** **static** **void** main(String[] args) **throws** IOException, TasteException {

File file = **new** File("src/ProfNotasFametro.csv");

DataModel model = **new** FileDataModel(file);

//chama o criador de recomendações

Recommender recommender = **new** RecomendadorBuilder().buildRecommender(model);

//recomendações

List<RecommendedItem> recommendations = recommender.recommend(9, 3);

**for** (RecommendedItem recommendation : recommendations) {

System.***out***.println(recommendation);

}

}

}

Em seguida é necessário criar no mesmo pacote a classe RecomendadorBuilder.java que retornará um recomendador baseado no modelo de dados obtidos na pesquisa experimental, para que o avaliador funcione corretamente.

**public** **class** RecomendadorBuilder **implements** RecommenderBuilder {

**public** Recommender buildRecommender(DataModel model) **throws** TasteException {

//função de similaridade

UserSimilarity similarity = **new** PearsonCorrelationSimilarity(model);

//calcula proximidade dos vizinhos mais próximos

UserNeighborhood neighborhood = **new** ThresholdUserNeighborhood(0.1, similarity, model);

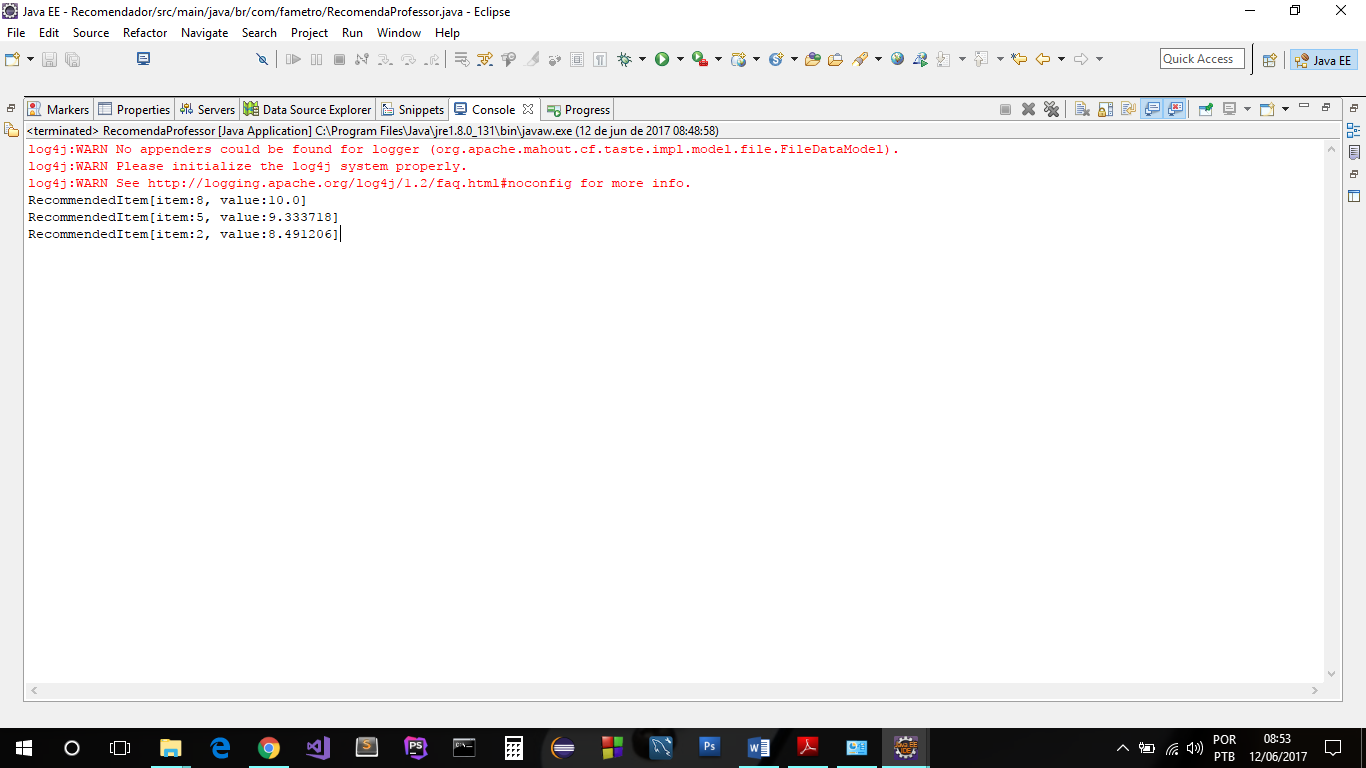
//cria recomendador

UserBasedRecommender recommender = **new** GenericUserBasedRecommender(model, neighborhood, similarity);

**return** recommender;

}

}

A executar esta classe já é possível analisar o funcionando do recomendador porém o eclipse ainda mostra umas mensagens de log como mostra a imagem abaixo.

Para configura-las basta criar um novo diretório no projeto *File>New>Folder*, folder name: src/main/resources clicar com o botão direito do mouse em cima da pasta e escolher *Build Path>Use as Source Folder* com isso criará um novo diretório no build e neste clica-se com o botão direito e cria-se um novo arquivo *File>New>Other>General>File* com o File name: *log4j.xml* adicionando o trecho de código abaixo para configurar as mensagens do loj4J.

<?xml version=*"1.0"* encoding=*"UTF-8"* ?>

<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<log4j:configuration xmlns:log4j=*"http://jakarta.apache.org/log4j/"*>

<appender name=*"main"* class=*"org.apache.log4j.ConsoleAppender"*>

<param name=*"Target"* value=*"System.out"*/>

<layout class=*"org.apache.log4j.PatternLayout"*>

<param name=*"ConversionPattern"* value=*"[%t] %d{HH:mm:ss,SSS} %5p [%-20c{1}] %m%n"* />

</layout>

</appender>

<category name=*"org.apache"*>

<priority value=*"DEBUG"*/>

<appender-ref ref=*"main"* />

</category>

</log4j:configuration>

## 

## 3.3 Treinando o algoritmo para avaliar o recomendador

Na seção anterior, implementou-se um recomendador de professores, porém é necessário automatizar uma avaliação dos algoritmos usados a fim de mensurar a qualidade das recomendações realizadas pelo mahout, baseado na análise do arquivo do Data Model.

Para efetivar o aprendizado de máquina, será imposto ao avaliador que separe 10% (dez por cento) dos dados para realizar testes e 90% (noventa por cento) para realizar o treino do algoritmo avaliador (AverageAbsoluteDifferenceRecommenderEvaluator). Este avaliador será baseado na média absoluta dos erros encontrados que solicitará como argumento um construtor de recomendações (que será implementado), um data model, e dois argumentos, o primeiro indicando a taxa de treinamento e o segundo a taxa de teste. A implementação do avaliador pode ser conferida a seguir, onde criou-se uma nova classe dentro do pacote br.com.fametro chamada Avaliador.java. que imprimirá a margem de erro.

**public** **class** Avaliador {

**public** **static** **void** main(String[] args) **throws** IOException, TasteException {

RandomUtils.*useTestSeed*();

File file = **new** File("src/ProfNotasFametro.csv");

DataModel model = **new** FileDataModel(file);

RecommenderEvaluator evaluator = **new** AverageAbsoluteDifferenceRecommenderEvaluator();

RecommenderBuilder builder = **new** RecomendadorBuilder();

**double** erro = evaluator.evaluate(builder , **null**, model, 0.9, 1.0);

System.***out***.println(erro);

}

}

## 3.4 Pergunta de pesquisa experimental

A seguinte questão de pesquisa é abordada:

Como avaliar realmente se a margem de erro encontrada pelo avaliador é fiel a realidade? E como garantir uma recomendação quando se tem pouco ou nada de informação sobre algum aluno?

## 3.5 Variáveis

A variável independente é a implementação da classe Avaliador.java que baseia na classe RecomendadorBuilder.java, onde encontram-se as funções de similaridade e vizinhos mais próximos utilizadas para garantir que o aprendizado de máquina seja bem sucedido, porém a avaliação não seria fiel a realidade se não houvesse a variável dependente, que é a melhor solução para que algoritmo não tome decisões viciadas, trocando de margem de erro a cada execução. Isso foi possível devido ao trecho código RandomUtils.*useTestSeed*(); que garantiu, de forma randômica, deixando fixa a margem de erro e aumentando a eficácia em qualquer modelagem de dados.

## 

## 3.6 Hipóteses

É notório na base de dados ProfNotasFametro.csv que, alguns alunos fizeram recomendação para todos os professores enquanto outros avaliaram pouco mais de 3 professores. Isso ocorre em todo tipo de base de dados, onde se tem mais informações sobre alguns usuários e menos informações quanto a outros. Porém o sistema de recomendação não será capaz de fazê-la quando não se tem informação nenhuma do aluno ou quando ele não avaliou a quantidade mínima de professores para tornar o algoritmo capaz de recomendar. A solução para esta hipótese é criar métodos extras para conseguir informações pertinentes a realidade do aluno, afim de alcançar condições necessárias para efetuar uma recomendação, como por exemplo, a criação de formulário com perguntas específicas para avaliar qual área ou disciplina que o aluno mais se identifica.

## 3.7 Ameaças a validade do experimento

A questão temporal também influencia bastante da qualidade da recomendação, dependendo da especificidade da mesma, pois avaliações feitas a muito tempo podem não ter relevância para o aluno no qual se pretende recomendar um professor. Além disso deve-se ter bastante cuidado com as escalas distintas, pois as mesmas podem influenciar negativamente no resultado final tanto da recomendação quanto da avaliação.

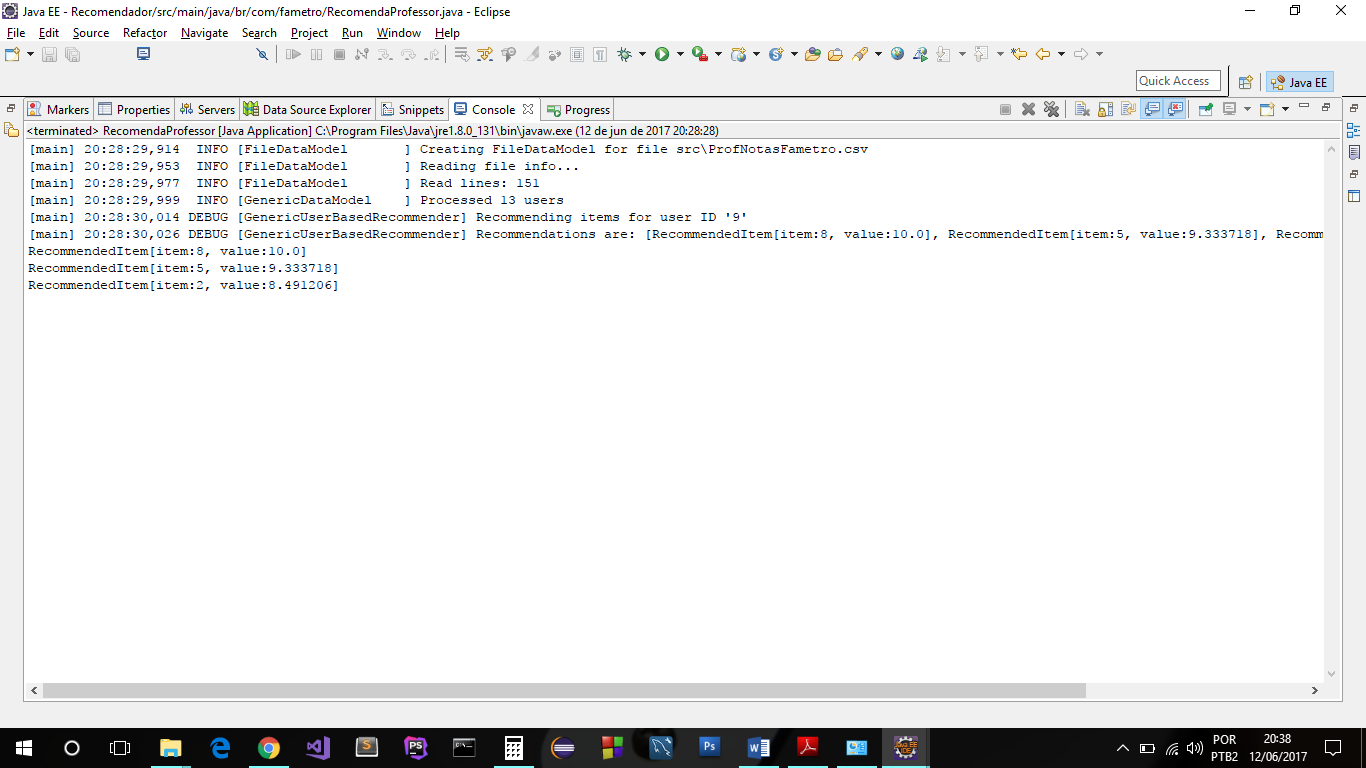
O algoritmo de similaridade e de avaliação, esperam escalas semelhantes, como por exemplo de 0 a 10, ou 0 a 1. Uma vez escolhida o tipo de escala a ser utilizada é recomendável que se mantenha o padrão para garantir uma boa qualidade no final da aplicação.

A recomendação para um usuário que avaliou todos os professores será inviável, pois o algoritmo não terá opções de recomendação, limitando a aplicação. Isso pode ocorrer simplesmente por vontade própria do aluno ou mesmo por inconsciência. A nota avaliada para um professor deverá ser digna da realidade, se em algum momento alunos não forem fiéis as suas avaliações, o algoritmo também seguirá a mesma sina, afinal ele é a representação de recomendações humanas reais.

Outra questão está associada ao anonimato de ambas as partes envolvidas na aplicação, pois o sistema poderá ser confundido com uma breve invasão de privacidade ou mesmo incômodo para os usuários no que diz respeito a usabilidade e interação. Daí a importância do uso de técnicas e esclarecimento explícito ao usuário, deixando claro o objetivo da aplicação e seus benefícios, a fim de mitigar possíveis insatisfação.

# 4. RESULTADOS

Na execução da classe RecomendaProfessor.java obtêm-se toda a saída da aplicação. Como este artigo tem por finalidade apenas embasar um sistema, a recomendação está sendo feita de forma estática, passando por parâmetro o usuário e a quantidade de recomendação desejada. A imagem abaixo mostra a saída de recomendações para o aluno de Id 9 além das notas que o algoritmo julga que o aluno dará a um professor.



O projeto completo poderá ser baixado pelo github através do link <https://github.com/JacksonPassos/RecomendaProfessor>. O resultado da taxa de erro encontrada no avaliador foi uma média de 1.34, o que representa que, por exemplo, na imagem acima que o algoritmo recomendou o professor 2 dando uma nota 8.49, implica-se que na realidade o aluno dará uma nota aproximadamente entre 7,15 e 9,83.

# 5. CONCLUSÃO

A interação humano-computador busca melhorar a experiência percebida do usuário, papel este que a recomendação de algo relevantes ao mesmo, ajuda a difundir. Este artigo abordou os principais conceitos existentes para recomendar usando o aprendizado de máquina para treinar algoritmos realizando de forma simples uma implementação de um recomendador de professores, que poderá ser utilizada para outros tipos de modelagens, bastando apenas carregar o arquivo em formato .csv. Explorou-se as funções utilizadas pelo mahout e realizou-se testes de validação para saber quão mensurável é a qualidade da aplicação. A partir de então fica embasado um possível sistema de recomendação que a faculdade em questão poderá implementar futuramente, apenas integrando com sistemas existentes ou criando outra plataforma.

A biblioteca mahout foi escolhida para implementação deste artigo devido sua alta escalabilidade e praticidade, provando que é possível aplicar uma inteligência artificial usando java, que é utilizado amplamente por grandes empresas. Com o fundamentação deste trabalho é possível imaginar e implementar diversos tipos de sistemas que pretende-se recomendar algum item para algum usuário de forma colaborativa e não necessariamente deverá ser lido um arquivo de texto, pois o mahout suporta a leitura de arquivos de um banco de dados, mantê-los em memória ou até mesmo grava-los em um arquivo temporário, rodar a aplicação de forma distribuída. Existem outros tipos de algoritmos e vale a pena aprofundar-se para saber qual deles é o mais adequado para a realidade que surgirá.

A pesquisa para obtenção dos dados, apesar de ser obtida por redes sociais de troca de mensagens, não diminui a importância do trabalho, pois toda a base de dados poderá ser importada de um banco de dados qualquer, havendo somente a necessidade de minerar e modelar os dados para o formato explicado. Ressalta-se que esta pesquisa poderá ser utilizada pela faculdade mencionada no trabalho, a fim de melhorar a interação dos alunos para o curso de graduação que eles cursam.

Após toda abordagem de conceitos e objetivos do sistema de recomendação por filtragem colaborativa, facilita-se o entendimento de como os algoritmos funcionam, abstraindo a administração da integração e funcionalidades através da biblioteca mahout e do apache maven, deixando claro a alta escalabilidade de um sistema implementado com estas tecnologias. Alguns cuidados na implementação também foram abordados, como por exemplo a questão do teste ser feito de forma randômica para não prejudicar a qualidade, pois no caso de ser realizado vários testes na mesma base de dados, chegará um determinado ponto que o algoritmo será muito bom e a margem de erro chegará próximo a zero, o que é muito preocupante, quando leva-se em consideração o aprendizado de máquina em outra base de dados.

Todo esforço que é pensado e arquitetado para facilitar a vida do usuário sempre será encarado com uma ótica positiva, pois todos desejam ter pouco trabalho e satisfazer-se rapidamente em qualquer sistema. A proposta almejada foi satisfeita, podendo recomendar professores para aqueles alunos que não sabem ao certo o que seguir profissionalmente, ou mesmo não sabe que professor poderá orienta-lo em seus trabalhos acadêmicos. Visto dessa forma que há inúmeras aplicações a serem projetadas, sempre buscando melhorar o índice de satisfação dos alunos quanto a sua instituição de ensino.

**RECOMMENDATION SYSTEM COLLABORATIVE FILTERING - A UNIFIED HUMAN MACHINE INTERFACE APPROACH WITH MACHINE LEARNING USING APACHE MAHOUT**

**ABSTRACT**

This article refers to how to help people live better, saving time and raising the level of satisfaction of the user experience, assisting them in making decisions in their daily lives with access to referral systems, as well as associating this type of System with a performance of human interaction and computer and its main benefits, is not of satisfaction. In order to substantiate a problematization of this document, a collaborative filtering in requirements systems was chosen using an apache mahout platform, an environment for the rapid creation of scalable and automatic applications for machine learning, with implementation in the Java language, where the same search Create for an e-mail Goal, based on experiences and correlated statements of other users of the same sympathy with the target user. At the end of the execution of the system type, it is possible to assess the degree of importance without impact of the user experience, as well as how to market strategy to explore. The objective of this work is to base an implementation of the simulation of a teacher recommendation system for students of the information system course of the Metropolitan College of the Great Fortress - Fametro, through its similarities, using an apache mahout library.

Keywords: Collaborative Filtration. Machine learning. Artificial intelligence. System of recommendation. Mahout.

# 6. REFERÊNCIAS

FERREIRA, Aletéia, **“Cybermarketing e a moda: segmentação através dos blogs e orkut”,** 2007.

KIELGAST, Soeren, HUBBARD, Bruce A., **“Valor agregado à informação – da teoria à prática”**, 1995.

SCHWARTZ, BARRY. **"O paradoxo da escolha: por que mais é menos."** São Paulo: A Girafa Editora, 2007.

MANBER, U. PATEL, A. ROBISON, J. **Experience with Personalization on Yahoo!** Communication of the ACM, New York, 2000.

GOLDBERG, D., NICHOLS, D., B. M. Oki, and D. Terry, **“Using Collaborative Filtering to Weave an Information Tapestry”,** 1992.

MARCELINO, Vasco Ferreira. **“Sistema de Recomendação - Filtragem Colaborativa”**, 2014.

REATEGUI, E.; CAZELLA, S. **Sistemas de Recomendação. Minicurso do IV Encontro Nacional de Inteligência Artificial (ENIA)**. São Leopoldo, p. 306-348, 2005.

REATEGUI, E.; CAZELLA, S. “**A ciência da opinião: estado da arte em sistemas de recomendação”**, 2015.

PAPAGELIS, M., & PLEXOUSAKIS, D. **Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents**, 2005.

EKSTRAND, M. D. **Collaborative Filtering Recommender Systems**, 2010.

DOUGLAS W. OARD and JINMOOK Kim, **“Implicit Feedback for a Recommender Systems”**, 1998.

APACHE MAHOUT. **O que é o Apache Mahout?** disponível em <http://mahout.apache.org/> acesso em abril de 2017.

MAHOUT. **Criando um recomendador com base em usuário em 5 minutos**, disponível em <https://mahout.apache.org/users/recommender/userbased-5-minutes.html> acesso em abril de 2017.

FAMETRO. **Corpo Docente**, disponível em <http://www.fametro.com.br/graduacao/bacharelado/sistema-de-informacao> acesso em maio de 2017.

ORACLE. **Java™ Runtime Environment**, disponível em <http://www.oracle.com/technetwork/java/javase/8u131-relnotes-3565278.html> acesso em fevereiro de 2017.

ECLIPSE. **Eclipse IDE for Java EE Developers**, disponível em <http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/marsr> acesso em março de 2017.

MAVEN. **Apache-maven-3.5.0**, disponível em < https://maven.apache.org/download.cgi> acesso em março de 2017.