UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS
CURSO DE SISTEMAS DE INFORMAÇÃO

SUELEN SCARIOTT

DESENVOLVIMENTO DE UM RECOMENDADOR DE LIVROS COM BASE EM TÉCNICAS DE MINERAÇÃO DE DADOS

CAXIAS DO SUL - RS
2019

UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS
CURSO DE SISTEMAS DE INFORMAÇÃO

SUELEN SCARIOTT

DESENVOLVIMENTO DE UM RECOMENDADOR DE LIVROS COM BASE EM TÉCNICAS DE MINERAÇÃO DE DADOS

Trabalho de Conclusão de Curso para obtenção do título de Bacharel em Sistemas de Informação pela Universidade de Caxias do Sul, na Área do Conhecimento de Ciências Exatas e Engenharias.

Orientadora Profa. Dr. Helena Graziottin Ribeiro.

CAXIAS DO SUL - RS
2019

SUELEN SCARIOTT

DESENVOLVIMENTO DE UM RECOMENDADOR DE LIVROS COM BASE EM TÉCNICAS DE MINERAÇÃO DE DADOS

Trabalho de Conclusão de Curso para obtenção do título de Bacharel em Sistemas de Informação pela Universidade de Caxias do Sul, na Área do Conhecimento de Ciências Exatas e Engenharias.

Aprovada em: 28/11/2019

Banca Examinadora

_____

Prof. Dr. Helena Graziottin Ribeiro

Universidade de Caxias do Sul – UCS

_____

Prof. Dr. Carine Geltrudes Webber

Universidade de Caxias do Sul - UCS

_____

Prof. Dr. Elisa Boff

Universidade de Caxias do Sul – UCS

AGRADECIMENTOS

Agradeço a minha família, por todo o apoio e incentivo durante essa caminhada que é a graduação. Por acreditar em mim e não permitir que desistir fosse uma opção.

Ao meu namorado, Ronald, agradeço pela companhia para estudar e pelo incentivo nos momentos de incerteza.

A Laura Ceconi por toda a ajuda com o Django.

A minha orientadora, Helena, agradeço pela paciência, condução e conselhos ao longo do trabalho.

RESUMO

Com o grande volume de dados disponível hoje em dia existe dificuldade de encontrar conteúdo relevante. Os sistemas de recomendação vêm ao encontro a essa necessidade, fornecendo recomendação de conteúdo de maneira personalizada, buscando oferecer informação realmente útil para cada pessoa. O tema principal do trabalho é a implementação de um protótipo de sistema de recomendação, chamado de recomendador de livros, fazendo uso de técnicas da mineração de dados como a clusterização para definir grupos de usuários. Há uma quantidade enorme de livros disponíveis, assim como informações sobre eles. O protótipo implementa filtragem colaborativa com base na similaridade de usuários, e foi desenvolvido na linguagem Python, juntamente com o framework de desenvolvimento Django.

Palavras-chave: sistemas de recomendação, *big data*, mineração de dados, *k-means*.

ABSTRACT

Considering the large amount of data available today there is difficulty in finding relevant content. The recommendation systems meet this need, providing personalized recommendation of content, seeking to offer really useful information for each person. The main theme of the work is the implementation of a prototype recommendation system, called book recommendation, making use of data mining techniques such as clustering to define user groups. There are huge amount of books available as well as information about them. The prototype implements collaborative filtering based on user similarity, and was developed in the Python language, along with the Django development framework.

Keywords: recommendation systems, big data, data mining, k-means.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

Sigla	**Significado**
CSV	*Comma Separated Values*
FC	Filtragem Colaborativa
GUI	Graphical User Interface
KDD	*Knowledge Discovery in Databases*
KNN	*K Nearest Neighbor*
MAE	Mean Absolute Error
MD	Mineração de Dados
RecSys	Recommender Systems
RMSE	*Root Mean Squared Error*
SGBD	Sistema de Gerenciamento de Banco de Dados
SQL	*Structured Query Language*
SR	Sistemas de Recomendação
TCC	Trabalho de Conclusão de Curso

SUMÁRIO

1 INTRODUÇÃO

Em meio a tanta informação disponível hoje em dia, acaba sendo difícil encontrar aquela que seja de maior relevância de acordo com a necessidade de cada pessoa. Isso ocorre nos mais diversos ambientes, tanto profissional, quanto acadêmico ou pessoal.

No contexto de publicações literárias, há uma expressiva quantidade de títulos disponíveis no mercado. São livros que vão desde os clássicos, escritos há séculos atrás, até as publicações mais recentes, e que não param de crescer a cada dia. A *web* e as livrarias virtuais tornaram os livros mais acessíveis, pois eles estão a apenas alguns “cliques” de distância, o que é mais prático do que buscar livros de maneira presencial. Nessas plataformas *online*, geralmente há uma grande variedade de obras para o leitor escolher.

Antes do uso de recursos tecnológicos para realizar recomendações isso já era feito entre as pessoas, onde aquelas que possuíam gostos semelhantes indicavam leituras umas às outras. Contudo, com o uso de recursos computacionais e da internet, pessoas que não se conhecem e que são de países distintos, por exemplo, podem estar contribuindo para a recomendação umas das outras. Sistemas de recomendação são capazes de considerar, processar e relacionar uma quantidade tão elevada de informações que seria impossível, em termos temporais, um ser humano realizar tal tarefa.

O enfoque deste trabalho é em relação a sistemas de recomendação para livros. Esses sistemas são desenvolvidos a partir de muitos dados sobre livros (*Big Data*), utilizando métodos e técnicas de análise sobre esses dados (mineração de dados) de acordo com as preferências dos usuários.

1.1 OBJETIVOS

O objetivo deste trabalho é implementar um protótipo de sistema de recomendação voltado para a recomendação de livros, fazendo uso de técnicas de mineração de dados para a implementação.

Para atender ao objetivo geral, são propostos os seguintes objetivos específicos:

a) Estudar conceitos referentes a sistemas de recomendação e mineração

de dados;

b) Identificar técnicas e ferramentas que são utilizadas na concepção de soluções de recomendação;

1.2 ORGANIZAÇÃO DO TEXTO

Este trabalho se divide em seis capítulos que vão desde o entendimento de conceitos necessários até a proposta de solução e trabalhos futuros. No capítulo dois é apresentado o conceito de *Big Data,* sua exploração através da mineração de dados, a inserção da mineração no processo de descoberta do conhecimento, algumas das principais tarefas relacionadas a ela, como a classificação e o agrupamento.

Através do capítulo três são explicados os sistemas de recomendação, bem como os tipos existentes e as estratégias que podem ser adotadas na sua concepção. No decorrer do capítulo quatro são apresentados trabalhos relacionados, realizando um levantamento do contexto em que cada proposta foi elaborada, quais os métodos, e formas de avaliação utilizadas.

O capítulo cinco apresenta a modelagem e desenvolvimento do protótipo de sistema de recomendação, inicialmente descrevendo a metodologia utilizada e mostrando elementos da modelagem, como as histórias de usuário e os protótipos de tela. Após são apresentadas as etapas do processo de KDD que foram executadas e os detalhes do desenvolvimento do motor de recomendação e da aplicação *Web*.

No sexto capítulo é elaborada a conclusão do trabalho, levando em consideração os trabalhos futuros.

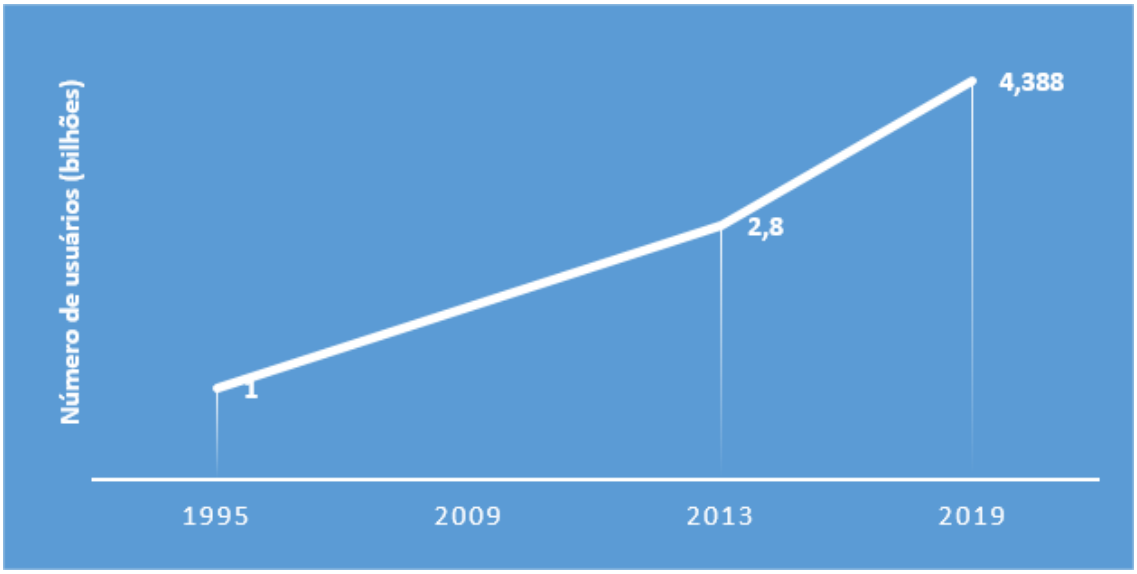
2 BIG DATA

Sistemas de recomendação se fazem necessários devido à dificuldade que as pessoas possuem de encontrar conteúdo relevante pra elas perante o oceano de informações disponíveis na atualidade. Através do entendimento do conceito e das características do *Big Data* se torna possível escolher e aplicar técnicas de Mineração de Dados, que farão com que ele deixe de ser um desafio, para se tornar um aliado na construção de soluções de recomendação.

O ser humano deixa vestígios por onde passa. Desde as pinturas rupestres até os registros escritos, sempre foram produzidas informações sobre a vida e a cultura das civilizações; mesmo que armazenadas em paredes de cavernas, potes de barro ou até mesmo, em papel. O tempo foi passando, as tecnologias se aperfeiçoando, e hoje em dia, foi atingido um ponto em que o tipo de informação que o homem produz não é mais apenas física, ela é também digital.

Principalmente a partir da década de 90, com a popularização da internet, aumentou o número de pessoas conectadas na web. Castro e Ferrari (2016) demonstram que enquanto que em 1995 a quantidade de usuários de internet era de em torno de 1 milhão de pessoas, em 2013 estava em 2,8 bilhões. Segundo Kemp (2019), em 2019 os usuários de internet já atingiram o número de 4,388 bilhões. A figura 1 representa o aumento desse número de usuários da internet ao longo dos anos.

Figura 1 - Usuários de internet ao longo dos anos.



Fonte: Próprio autor.

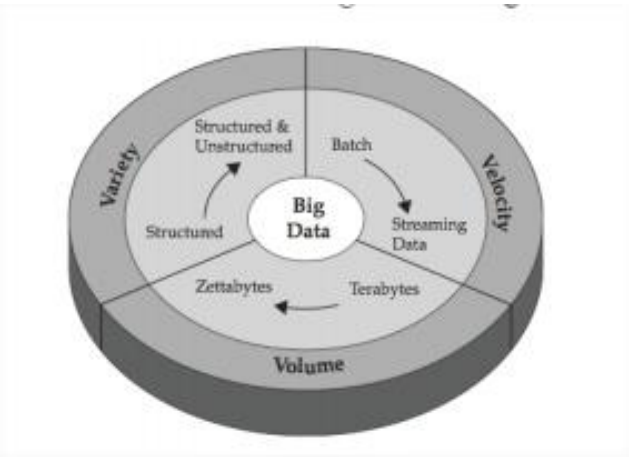
Zikopoulos (2012), relata que “a era do Big Data está em pleno vigor hoje porque o mundo está mudando. (..) Através de avanços na tecnologia de comunicação, pessoas e coisas estão se tornando cada vez mais interconectados - e não apenas parte do tempo, mas o tempo todo.”

Em consequência da popularização da internet e consequentemente, do aumento da interconexão entre pessoas e coisas, a quantidade de dados produzidos pelos usuários da rede veio crescendo rapidamente.

Castro e Ferrari (2016, p. 9) afirmam que “o volume de dados produzidos cresce exponencialmente, ao ponto de que em curtos períodos de tempo se geram mais dados do que em muitos séculos de história da humanidade.” Para referenciar esse grande volume de dados foi cunhado o termo *Big Data*.

Alves (2015, p. 18) afirma que *big data* “não se trata apenas uma questão de uma certa quantidade de bytes. Três características o distinguem: volume, variedade e velocidade”. A figura 2 demonstra essas três características.

Figura 2 - Caracterização do big data por seu volume, velocidade e variedade.



Fonte: ZIKOPOULOS, 2012, p. 33.

O volume refere-se basicamente da quantidade de informações; a variedade é relativa aos diferentes tipos de dados existentes, como, por exemplo: textos de uma rede social, metadados de imagens, dados de um sensor, GPS de um celular, vídeos de uma câmera de segurança; e a velocidade diz respeito ao retorno das análises da informação no menor tempo possível, às vezes até mesmo em tempo real (ALVES, 2015).

2.1 EXPLORAÇÃO DO BIG DATA ATRAVÉS DA MINERAÇÃO DE DADOS

Castro e Ferrari (2016) afirmam que o desafio dos tempos atuais em relação ao Big Data é conseguir gerenciar, armazenar, processar e extrair conhecimento a partir dele. É através do uso de técnicas e ferramentas que se torna possível aplicar métodos de análise de dados para enfrentar tal desafio.

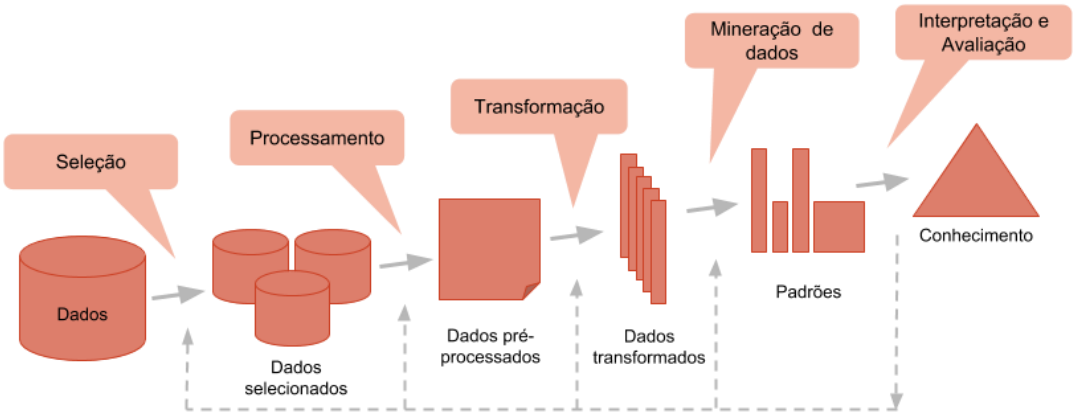
O processo de exploração de dados através do uso ferramentas e com o intuito de obtenção de conhecimento foi batizado de mineração de dados (MD). Esse nome faz referência ao processo de extração de minerais valiosos de uma mina. MD é um método de análise de dados que envolve a "seleção e integração das bases de dados, a limpeza da base, a seleção e transformação dos dados, a mineração e avaliação dos dados" (CASTRO; FERRARI, 2016, p. 26).

Segundo Fayyad at. All (1996), a mineração de dados é uma das etapas do processo da descoberta do conhecimento (*Knowledge Discovery)* que consiste em aplicar algoritmos de análise e descoberta de dados visando identificar padrões (ou modelos) sobre os dados. Braga (2005, p. 15) reforça essa definição afirmando que "a mineração de dados está inserida em um processo maior denominado descoberta de conhecimento em banco de dados, *Knowledge Discovery in Databases* (KDD)." Pode-se conceituar o termo KDD pela definição a seguir.

> "O termo *KDD-Knowledge Discovery in Databases* foi criado em 1995 para designar o conjunto de processos, técnicas e abordagens que propiciam o contexto no qual a mineração de dados terá lugar. Em suma é a aplicação do método científico moderno aos problemas do mundo dos negócios." (BRAGA, 2005, p.23)

Na figura 3, pode-se observar onde se encontra inserida a mineração de dados no processo de descoberta do conhecimento, sendo precedida pela etapa de formatação e sucedida pela interpretação, ou avaliação, dos resultados obtidos. Este processo pode ser considerado interativo e iterativo. Este último significa que o processo é cíclico, ou seja, ele pode se repetir até que os resultados esperados sejam atingidos. E ser interativo significa que ocorrem decisões humanas em cada etapa do processo, pois a pessoa que está executando o processo interage com mesmo.

Figura 3 - Fluxo do KDD



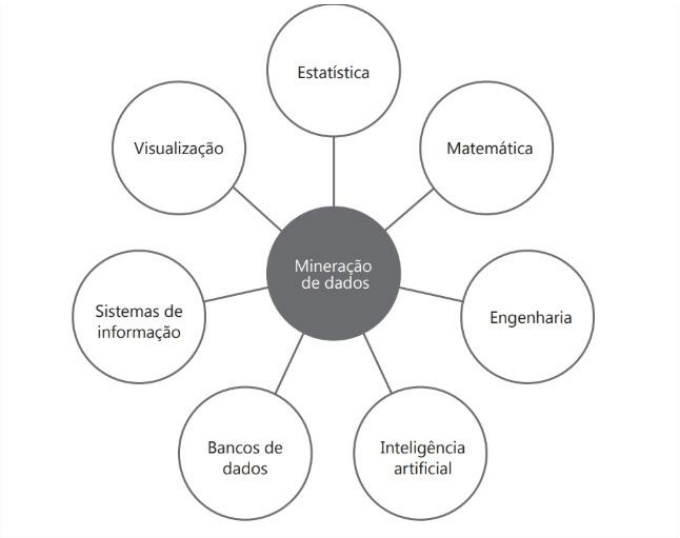
Fonte: Moura (2019).

Este fluxo do KDD pode ser explicado resumidamente pelas seguintes etapas, segundo Fayyad at. All (1996):

a) Seleção: Entendimento do problema e dos objetivos que se tem com o processo. Com base nesse entendimento é realizada a filtragem dos dados que serão usados nas etapas posteriores;
b) Pré-processamento: remoção de ruído (se existir) usando estratégias para resolver dados faltantes e remoção de irrelevâncias;
c) Transformação/Formatação: Conversão dos dados para tipos de dados adequados e que sejam suportados pelos algoritmos de mineração que se pretende utilizar.
d) Mineração de dados: Encontrar o método de mineração de dados que irá atender os objetivos propostos e aplicá-lo.
e) Interpretação e Avaliação: Compreender os padrões identificados e avaliar a efetividade do modelo escolhido. Neste momento pode-se decidir por executar novamente o processo, mudando a abordagem para comparar com os resultados obtidos até o momento.

Castro e Ferrari (2016), ilustram, através da figura 4, que a MD envolve conhecimento de diversas áreas. São elas: a estatística, a matemática, a engenharia, a IA, banco de dados, sistemas de informação, e visualização.

Figura 4 - Áreas envolvidas na MD.



Fonte: Castro e Ferrari (2016, p. 28).

Dentro da MD existem dois paradigmas de aprendizagem, sendo que o aprendizado pode ser dividido em supervisionado e não-supervisionado. Quando os rótulos das classes dos dados utilizados para treinamento de algum algoritmo já são conhecidos, chama-se isso de aprendizagem supervisionada. E quando esses rótulos não são conhecidos, chama-se de aprendizagem não supervisionada. Exemplo de tarefa que usa aprendizagem supervisionada: classificação; e de não supervisionada: *clustering* (agrupamentos).

Ao trabalhar com aprendizagem supervisionada é importante tomar cuidado com dois tipos de erro que podem ocorrer: erro de representação/aproximação e erro de generalização. Quando ocorre erro de representação significa que o modelo “não é suficientemente flexível para representar o conjunto de objetos” (CASTRO; FERRARI, 2016, p. 170). Erro de generalização ocorre por exemplo “quando o modelo é treinado em excesso e absorve os ruídos dos dados de treinamento” (CASTRO; FERRARI, 2016, p. 170).

Para evitar que tais erros ocorram é comum buscar interromper o treinamento no momento adequado. Para isso é utilizada a validação cruzada (*cross validation*). “Para alguns modelos, como as árvores de decisão e as regras de classificação, esse tipo de mecanismo não é necessário, mas, para outros, como as redes neurais, é muito usado para definir adequadamente o ponto de parada do treinamento.” (CASTRO; FERRARI, 2016, p. 171).

2.2 TAREFAS EXECUTADAS NA MINERAÇÃO DE DADOS

Na MD existem vários algoritmos que podem ser usados de acordo com o

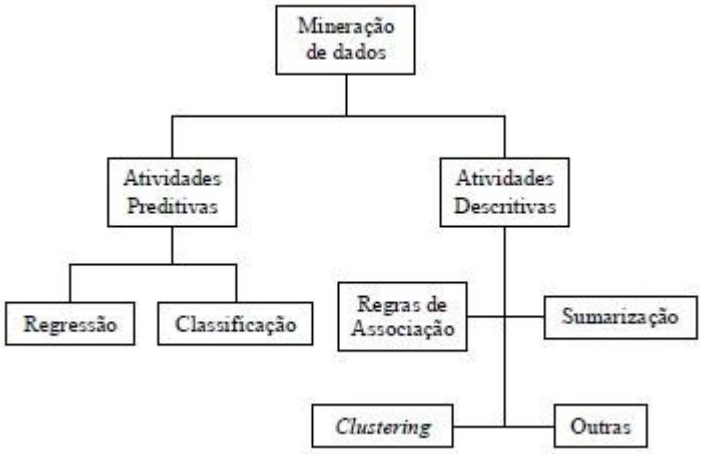
problema que se busca resolver. Eles podem ser subdivididos e dois grupos: preditivos e descritivos.

Os modelos descritivos auxiliam na identificação de relações entre os dados, de forma investigativa, permitindo entender melhor o que ocorreu no passado. Ele descreve os fatos ocorridos. Já os modelos preditivos detectam padrões nos dados, possibilitando que eles prevejam o que pode ocorrer no futuro.

> O Aprendizado preditivo analisa os dados que representam eventos passados buscando relações entre estes que permitam predizer situações em novos dados futuros, tais como: a classificação para predições de valores discretos e a regressão para predições de valores contínuos. (CARVALHO; DALLAGASSA, 2014)

A Mineração de Dados possui capacidade de realizar determinadas tarefas. Cada uma delas ajuda a realizar atividades que se enquadram em um dos grupos de algoritmos, que são os descritivos ou preditivos. A figura 5 demonstra, de forma esquematizada, a separação das tarefas de acordo com o grupo em que ela se enquadra, sendo que nas preditivas encontra-se a regressão e classificação; e nas descritivas, as regras de associação, a sumarização e o *clustering.*

Figura 5 - Separação de tarefas da MD



Fonte: TRONCHONI et al. (2010).

Quando se trabalha com atividades preditivas, Monard e Baranauskas (2003, p. 44), destacam pontos importantes em relação a escolha dos atributos conforme destacado no trecho a seguir:

> Um ponto importante a ser considerado é a escolha de atributos com boa capacidade preditiva. Não importa qual método seja empregado, os conceitos que podem ser aprendidos estão à mercê dos dados e da qualidade dos atributos. Por exemplo, para a tarefa de determinar se uma pessoa está ou não com gripe, pode-se escolher atributos com baixo poder

> preditivo, tais como (cor-do-cabelo, cor-do-olho, modelo-do-carro, número-de-filhos) ou atributos com alto poder preditivo, tais como (temperatura, resistência-da-pele, exame do pulmão). Para esta tarefa específica, no segundo caso, melhores previsões em exemplos não-rotulados provavelmente ocorrerão do que com o primeiro conjunto de atributos.

Algumas das tarefas da Mineração de Dados mais comumente utilizadas estão apresentadas a seguir, da seção 2.1.1 até a seção 2.2.5.

2.2.1 Classificação

Capaz de predizer a classe de um item. Na classificação, os dados de entrada (para treinamento) do algoritmo já se encontram rotulados, ou seja, já tem uma classe definida. O objetivo é que quando se tenha um novo dado, por exemplo, um novo leitor na questão da recomendação de livros, seja possível identificar em qual classe existente ele se encaixaria melhor. Isto ocorre, pois, ao realizar a etapa de treinamento, é criado um modelo que será capaz de identificar a qual classe novos leitores pertencerão.

2.2.2 Agrupamento (*Clustering*)

Identifica agrupamentos nos dados. Nesta tarefa os dados não estão previamente rotulados, cabendo ao algoritmo identificar grupos. Dentro de um mesmo grupo deve haver a maior homogeneidade possível nos dados e entre um grupo e outro deve haver heterogeneidade. Realizando análises sobre um conjunto de dados podem ser obtidos diferentes agrupamentos, dependendo de quais características se deseja considerar para fazer esse agrupamento.

> Um agrupamento (ou cluster) é uma coleção de registros similares entre si, porém diferentes dos outros registros nos demais agrupamentos (…) Geralmente a tarefa de agrupamento é combinada com outras tarefas, além de serem usadas na fase de preparação dos dados. (CAMILO, 2009, p. 10)

2.2.3 Associação

Encontra associações que ocorrem com frequência entre os atributos, permitindo identificar relações do tipo: quem leu A também leu B. Isso permite que, por exemplo, na elaboração de uma solução de recomendação, essas relações sejam utilizadas para sugerir B aos leitores que até o momento leram apenas A.

2.2.4 Algoritmos para realizar as tarefas de mineração de dados

Existem algoritmos que possibilitam a realização de cada uma das tarefas da mineração de dados. Há vários para cada tarefa e com diferentes implementações. Serão listados a seguir alguns exemplos de algoritmos para cada uma das tarefas.

- Classificação: *Naive Bayes, J48, MultilayerPerceptron;*
- Agrupamento: *Knn, k-means, EM, Cobweb, hierarchical clustering;*
- Associação: *Apriori*, *FP Growth.*

3 SISTEMAS DE RECOMENDAÇÃO

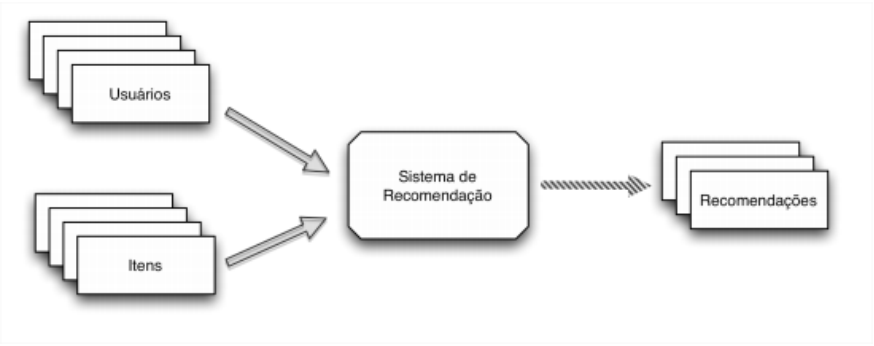
Devido ao grande volume de dados existentes (*Big Data)*, torna-se difícil aos usuários encontrarem informação realmente relevante através de buscadores na internet, como por exemplo o google.com, pois para tal é necessário que os mesmos forneçam palavras-chave na busca (CAZZELA, 2006).

Os sistemas de recomendação, também chamados apenas de SR, ou então RecSys, possibilitam encontrar informação sem que seja necessário o uso de palavra-chave, permitindo assim encontrar resultados relevantes para sugerir aos usuários. RecSys exigem menos experiência e esforço por parte de quem o utiliza, pois fornecem recomendações que foram reconhecidas como satisfatórias pelos seus utilizadores.

> "Um Sistema de Recomendação é um conjunto de técnicas e algoritmos que seleciona itens tendo como base os dados de interação e interesses dos usuários. Esses itens recomendados podem ser de diversos tipos como livros, filmes, música, vídeos, produtos em loja de comércio eletrônico, etc." (TAKAHASHI, 2015, p. 6).

Sendo mais conhecidos por sua utilização em *e-commerces*, esses sistemas não são restritos apenas a esse uso. Souza (2014, p.1), cita que "Sistemas de Recomendação podem ser vistos ao realizar buscas em sites de pesquisa da internet, em compras online, ou até mesmo ao visualizamos nossos e-mails". Seu funcionamento pode ter como base tanto o uso dados históricos dos usuários, buscando itens similares a esses dados; como também tomando como fundamento a preferência de pessoas com perfil parecido ao que se está fazendo a recomendação. A figura 6 demonstra que a fonte de informações que alimenta o SR é proveniente tanto dos usuários, como dos itens.

Figura 6 - Fontes de dados dos SR



Fonte: Souza (2011, p. 16).

Adomavicius e Tuzhilin (2005) citam que:

> “os sistemas de recomendação emergiram como uma área de pesquisa independente em meados da década de 1990, quando os pesquisadores começaram a se concentrar em problemas de recomendação que dependiam explicitamente da estrutura de pontuações (*ratings*).”

As propostas mais comuns de SR envolvem possibilitar a estimativa de itens que ainda não foram pontuados pelo usuário, buscando identificar qual o grau de relevância desses itens para ele.

Sistemas de Recomendação “são o mecanismo por trás da propaganda personalizada” (SOUZA, 2014, p.1), podendo ser aplicados apenas quando há grande volume de dados disponíveis. As preferências dos usuários, que irão compor essa massa de dados, podem ser recuperadas de duas maneiras: implícita ou explicitamente.

A coleta explícita de dados ocorre quando um usuário, voluntariamente, fornece seu feedback para o sistema, como, por exemplo, quando ele avalia um produto ou um vendedor. Já a coleta implícita ocorre ao armazenar dados sobre o comportamento do usuário, como por exemplo, onde ele clicou; ou então dados do seu histórico de compras e navegação no site; sua localização geográfica, dentre outras.

3.1 TIPOS DE SISTEMAS DE RECOMENDAÇÃO

Uma forma bastante utilizada para classificar os sistemas de recomendação é em três tipos. São elas: baseado em conteúdo, filtragem colaborativa e abordagens híbridas. Souza (2014, p.1), é um dos autores que utiliza essa separação.

Adomavicius e Tuzhilin (2005, p. 735) também realizam essa divisão em três categorias:

> “[...] sistemas de recomendação geralmente são classificados nas seguintes categorias, com base em como as recomendações são feitas:
>
> - Recomendações baseadas em conteúdo: Será recomendado ao usuário itens semelhantes aos que ele preferiu no passado;
> - Recomendações colaborativas: Será recomendado ao usuário itens que pessoas com gostos e preferências semelhantes gostaram no passado;

> - Abordagens híbridas: esses métodos combinam métodos colaborativos e baseados em conteúdo.”

Contudo, outros autores falam também em *Context-aware Recommender Systems* (Sistemas de Recomendação Cientes do Contexto). A seguir serão detalhadas e exemplificadas cada uma dessas formas de classificação.

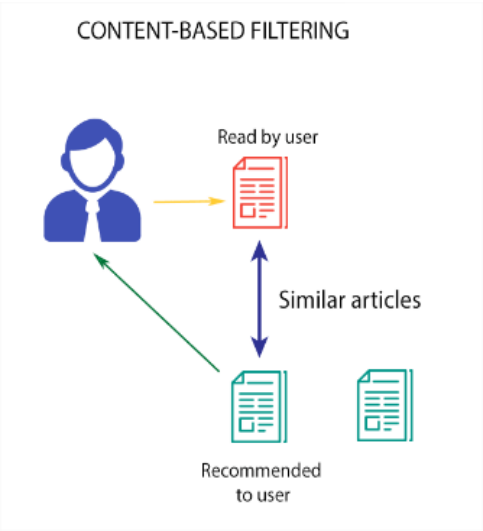
3.1.1 Sistemas de Recomendação baseados em conteúdo

Quando um SR é baseado em conteúdo significa que com base nas preferências do passado de determinado usuário serão indicados novos itens. Conforme pode ser observado na figura 7, o sistema encontra itens similares baseando-se em algo que o usuário já leu e os recomenda para ele. Souza (2011) cita que os sistemas de recomendação baseados em conteúdo são elaborados de forma que sejam sugeridos itens similares aos que os usuários já tenham avaliado positivamente anteriormente.

> “Algoritmos de recomendação baseados em conteúdo aprendem os perfis de interesse dos usuários com base nas características presentes em cada um dos itens que esses usuários avaliaram previamente. Sendo assim, eles constroem os perfis dos usuários a partir do perfil dos itens. O tipo de perfil derivado depende do método de aprendizado empregado. Árvores de decisão, redes neurais, e representações vetoriais são alguns exemplos destes métodos que tem sido comumente utilizados. Estes perfis de usuários são modelos que demandam algum tempo de observação e são atualizados a medida em que novas evidências relacionadas às preferências dos usuários são observadas.” (SOUZA, 2011, p. 20)

A construção do perfil dos itens envolve a extração de dados que possibilitem identificar ou categorizar os mesmos; são as características deles. Por exemplo: gênero do livro (horror, fantasia, autoajuda, ficção científica…), autor, data e país de publicação.

Figura 7 - Funcionamento da filtragem por conteúdo



Fonte: SANTANA (2019).

A vantagem de seu uso é que ele consegue começar a sugerir itens mesmo sem muitas informações iniciais do usuário, porém, é incapaz de sugerir novos assuntos, ficando sempre preso a temas que já são de conhecimento do utilizador. Isso é chamado de o*verspecialization,* que pode ser traduzido como superespecialização. Adomavicius e Tuzhilin (2005) afirmam que quando o sistema somente consegue recomendar itens que combinem com o perfil do usuário, este está fadado a ter apenas recomendações de itens parecidos com os que ele já avaliou positivamente.

3.1.2 Filtragem Colaborativa

O segundo tipo é chamado de filtragem colaborativa, onde o sistema busca itens de usuários parecidos para fazer as sugestões. Reategui e Cazella (2005) explicam que a expressão “filtragem colaborativa” foi criada para designar um tipo de sistema onde exista colaboração entre um grupo de interessados.

> “A filtragem colaborativa automatiza essencialmente o processo de recomendações "boca a boca", isto é, as informações são recomendadas a um utilizador baseado nos valores atribuídos por outras pessoas com interesses semelhantes.” (FERREIRA; OLIVEIRA, 2012)

Sistemas de filtragem colaborativa identificam automaticamente relações entre os usuários com base em similaridades não somente de seus perfis, mas também de seus comportamentos. A tabela 1 demonstra de maneira simples como

funciona na prática a filtragem colaborativa. Por exemplo, tanto Maria quanto José deram boas pontuações para os livros “Anjos e Demônios” e “Ponto de Impacto”. Supondo que quem leu algum desses títulos gosta também do outro, então pode-se recomendar o livro “Anjos e Demônios” para a Ema, pois ela tem o gosto parecido com Maria e José.

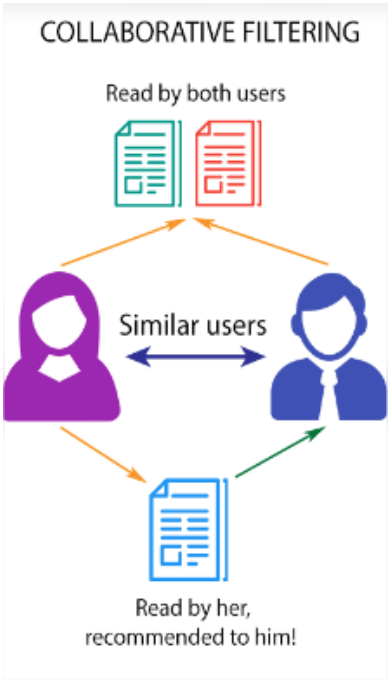
Tabela 1 - Matriz de usuários versus *livros*

	Pontuação por livro (de 1 a 5)		
Usuário	Anjos e Demônios	A Culpa é das Estrelas	Ponto de Impacto
Maria	5		4
Ema	?	1	5
Joana		4	
José	4		4

Fonte: Próprio autor.

E a figura 8 ilustra o funcionamento da filtragem colaborativa exemplificada com a tabela 1, em que o sistema encontra usuários similares para fazer recomendações.

Figura 8 - Funcionamento da filtragem colaborativa



Fonte: Santana (2019).

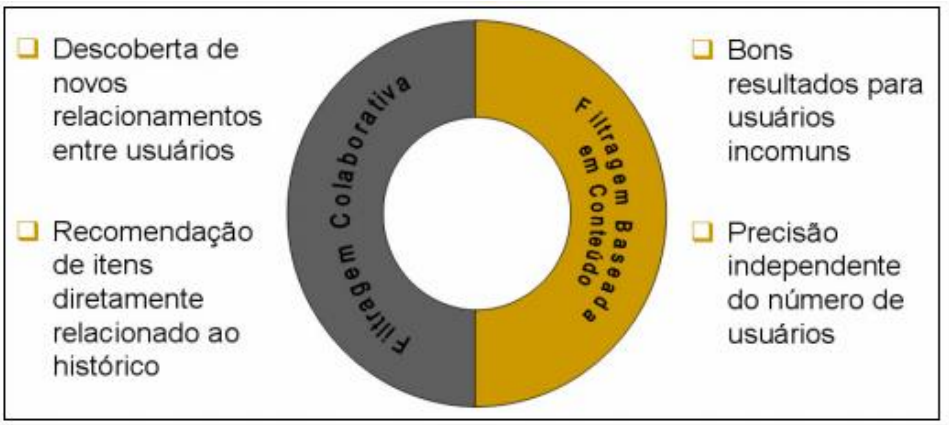
A vantagem do uso dessa abordagem é que evita recomendações repetitivas, porém, para um funcionamento adequado requer grande número de dados dos usuários. Se destacam alguns problemas como:

- O primeiro avaliador: quando um novo item surge o sistema não tem condição de recomendá-lo, pois ele ainda não foi avaliado por ninguém;
- As pontuações esparsas: quando há poucos utilizadores e muito volume de informação. Nesse cenário torna-se difícil recomendar, por causa do baixo nível de itens já pontuados;
- A similaridade: onde para alguém com gostos bem incomuns será difícil encontrar pessoas com interesses similares, tornando a recomendação pobre, ou fraca.

3.1.3 Sistemas híbridos

O terceiro tipo, por sua vez, é composto pela combinação dos dois métodos anteriores e chama-se de sistema híbrido. Ele busca somar os pontos fortes da filtragem colaborativa e da baseada em conteúdo. Como pode ser observado na figura 9 ele aproveita qualidades da filtragem baseada em conteúdo, como os bons resultados para usuários incomuns e a precisão, mesmo com poucos utilizadores, pois não depende de encontrar relações com pessoas parecidas. E juntamente a isso agrega coisas interessantes da filtragem colaborativa, como a descoberta de novos relacionamentos entre usuários e a recomendação de itens diretamente relacionada com o histórico de pontuações que o usuário já atribuiu no passado.

Figura 9 - Sistemas híbridos



Fonte: Reategui e Cazella (2005, p. 320).

3.1.4 Sistemas de Recomendação cientes do contexto

Alguns autores, como Hidasi (2014) e Takahashi (2015), em publicações mais recentes, expõem uma quarta abordagem, chamada de *Context-aware Recommender Systems*, que pode ser traduzida como Sistemas de Recomendação cientes do contexto. Este incorpora não somente informações referentes ao usuário e ao item, como também leva em consideração dados do contexto, observando "dados do contexto que levaram ao acontecimento do evento como tempo, clima, local, pessoas que estavam juntas entre outras" (TAKAHASHI, 2015, p. 13).

Ricci, Rokach e Shapira (2019, p. 13) exemplificam de maneira simples uma aplicação de SR cientes do contexto:

> "[...] o contexto do usuário pode ser usado para personalizar melhor os *outputs* do sistema. Por exemplo, em um contexto temporal, as recomendações de férias no inverno devem ser muito diferentes das fornecidas no verão."

Podendo ser considerados como uma ramificação dos SR cientes de contexto, existem também as abordagens demográficas. Este tipo de sistema recomenda itens baseado no perfil demográfico do usuário. Segundo (Krulwich, 1997), a abordagem utiliza as descrições das pessoas, como a idade, o gênero, a profissão, o endereço, etc. para compreender a relação entre uma simples informação e o tipo de pessoa que tem a mesma referência. "O perfil do usuário é criado pela classificação dos usuários em estereótipos que representam as características de uma classe de usuários." (REATEGUI; CAZELLA 2005, p. 307)

> Muitos sites adotam soluções de personalização simples e eficazes com base em dados demográficos. Por exemplo, os usuários são enviados para determinados sites com base em seu idioma ou país. Ou sugestões podem ser personalizadas de acordo com a idade do usuário. (RICCI; ROKACH; SHAPIRA, 2019)

Um exemplo de uso de abordagem demográfica está no sistema PRIZM, da *Claritas Corporation*. Ele enquadra cada residência em algum de seus 68 segmentos demográficos e comportamentais, permitindo que ações de marketing, sejam direcionadas ao público-alvo de acordo com sua localização. Isso pode, por exemplo, auxiliar na escolha dos melhores pontos para propaganda via *Outdoors* na cidade.

3.2 ESTRATÉGIAS DE RECOMENDAÇÃO

Podem ser empregadas diversas estratégias para realizar as recomendações aos usuários. Segundo Reategui e Cazella (2005), as mais utilizadas são as listas de recomendação, avaliação dos usuários, suas recomendações, usuários que se interessaram por X também se interessaram por Y, e associação por conteúdo.

- Listas de recomendação: São elaboradas listas de itens organizados por tipos de interesses. Essa estratégia não exige necessariamente que se tenha muito conhecimento sobre o perfil do usuário. Exemplos de lista: presentes para o dia dos namorados.
- Avaliação dos usuários. Ocorre quando o usuário pode avaliar o item. Existe tanto de forma qualitativa, através de comentários; como na forma quantitativa, quando uma pontuação é atribuída ao item como forma de avaliação. Depois de realizada a avaliação ela fica disponível para que os demais usuários possam ver a opinião dos outros a respeito de determinado item.
- Suas recomendações: Quando a recomendação é personalizada, elaborada com base em dados do perfil do usuário.
- Pessoas que se interessaram por X também se interessaram por Y: Esta estratégia busca encontrar padrões nos hábitos dos usuários, como por exemplo: quem leu “Orgulho e Preconceito” também leu “Persuasão”. Reategui e Cazella (2005, p. 314), explicam que “ela exige uma análise mais profunda dos hábitos do usuário para a identificação de padrões e recomendação de itens com base nestes padrões.”
- Associação por conteúdo: recomendação baseada no conteúdo dos itens. Por exemplo: o sistema identifica uma relação entre preferências por autores, como quem lê Meg Cabot também lê Cassandra Clare, então se uma pessoa leu algum livro de uma dessas autoras, será sugerido para ele também livros da outra autora.

4 TRABALHOS RELACIONADOS

No desenvolvimento de sistemas de recomendação podem ser utilizadas inúmeras abordagens e a escolha entre uma ou outra forma de implementação depende de fatores como o tipo de sistema que se deseja construir, qual, ou quais estratégias de recomendação serão priorizadas e quais dados estão disponíveis para poder basear o modelo.

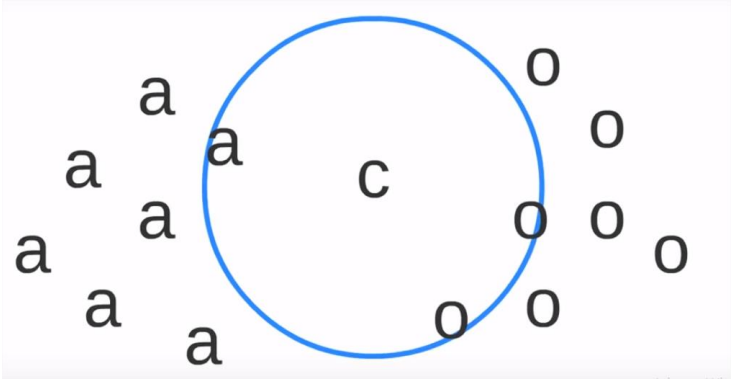
Este capítulo apresenta alguns artigos que tem relação com o presente estudo por abordarem a concepção de sistemas de recomendação ou estudos a respeito dos mesmos. Com ele se busca informações como o contexto em que cada proposta foi elaborada, quais os métodos utilizados, bem como as formas de avaliação desses trabalhos.

4.1 APLICAÇÃO DO ALGORITMO *K NEAREST NEIGHBOR* E SIMILARIDADE DO COSSENO

Cui (2017) apresenta o design e a implementação de um sistema de recomendação para filmes. O objetivo principal é ajudar usuários a obterem filmes de seu interesse perante os massivos dados de filme existentes. Nele é utilizado o algoritmo Knn para aplicar uma abordagem de filtragem colaborativa.

O Knn (*K Nearest Neighbor*) é um algoritmo de classificação. Ele identifica os k vizinhos mais próximos de determinado elemento. Por exemplo: se o k for igual a 3, então na situação ilustrada na figura 10, por exemplo, o Knn busca identificar quais os 3 os elementos que estão mais próximos do elemento c, como pode ser observado pelos elementos selecionados dentro do círculo azul. As letras “a” representam elementos de classe “a” e as leras “o”, elementos de classe “o”. Após essa identificação dos 3 elementos mais próximos é realizada a classificação do elemento “c”. Ele irá para a classe que for predominante dentre os vizinhos, que no caso é a classe “o”.

Figura 10 - Exemplo de busca dos 3 vizinhos mais próximos de c.



Fonte: Körting (2014).

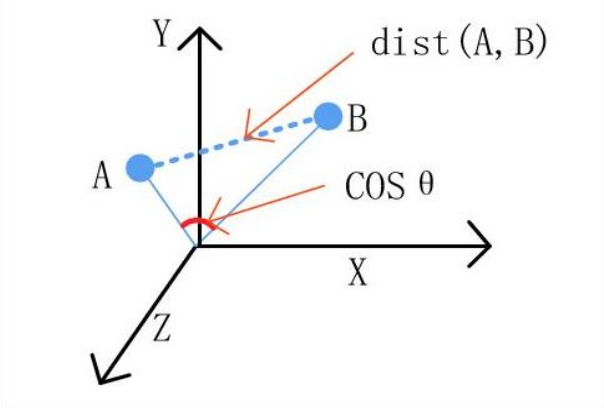
A similaridade entre os vizinhos é calculada com base nas pontuações que cada usuário atribuiu aos filmes. Cui (2017) cita ainda que geralmente a similaridade entre 2 usuários é calculada utilizando *Cosine Similarity* (similaridade de cosseno).

Na figura 11 há 2 retas identificadas por linhas azuis. Elas foram traçadas de cada ponto (“A” e “B”) até a intersecção dos eixos do gráfico. *Cosine Similarity* é um indicador referente ao ângulo formado entre tais retas. Quanto menor (mais próximo de zero) for o ângulo, maior a similaridade entre os dois pontos.

No gráfico da figura 11, as letras “A” e “B” representam vetores. Quando é realizada recomendação usando filtragem colaborativa, geralmente os dados de cada usuário versus suas avaliações sobre os itens são transformados em um vetor, sendo que os componentes do vetor seriam formados pela pontuação que o usuário atribuiu a cada item. Por exemplo: UsuárioA{5,4,3,2,3,5} e UsuárioB{4,4,1,1,2,4}. Cui (2017, p. 2) explica *Cosine Similarity* dizendo que “o método calcula a similaridade entre dois usuários calculando o cosseno do ângulo entre os dois vetores.”

O resultado do cálculo da similaridade de cosseno pode variar de -1 até +1, sendo que entre -1 e 0 (zero) significa que os pontos são praticamente opostos (muito dissimilares) e entre 0 e +1 há semelhança, sendo que quanto mais próximo de +1, maior a semelhança.

Figura 11 - Representação da semelhança de cosseno entre dois pontos.



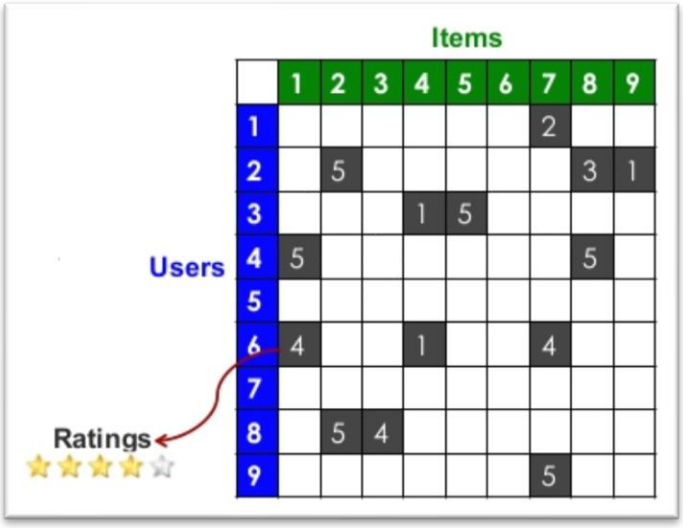
Fonte: Wang, Chen e Wu (2017, p. 146).

4.2 APLICAÇÃO DE FATORAÇÃO DE MATRIZES

Bokde, Girase e Mukhopadhyay (2015, p. 137) propõem um estudo compreensível sobre Matrix Factorization (Fatoração de Matrizes), para lidar com os desafios na área de algoritmos para Filtragem Colaborativa. Eles citam que “A abordagem de fatoração de matrizes é considerada mais eficaz para reduzir o problema de altos níveis de esparsidade nos dados de Sistemas de Recomendação”.

Uma matriz é considerada esparsa quando muitos de seus elementos possuem valor zero ou são nulos (não possuem valor algum especificado). Por exemplo, na matriz representada através da figura 12, todos os elementos que não possuem valores especificados contribuem para tornar a matriz esparsa.

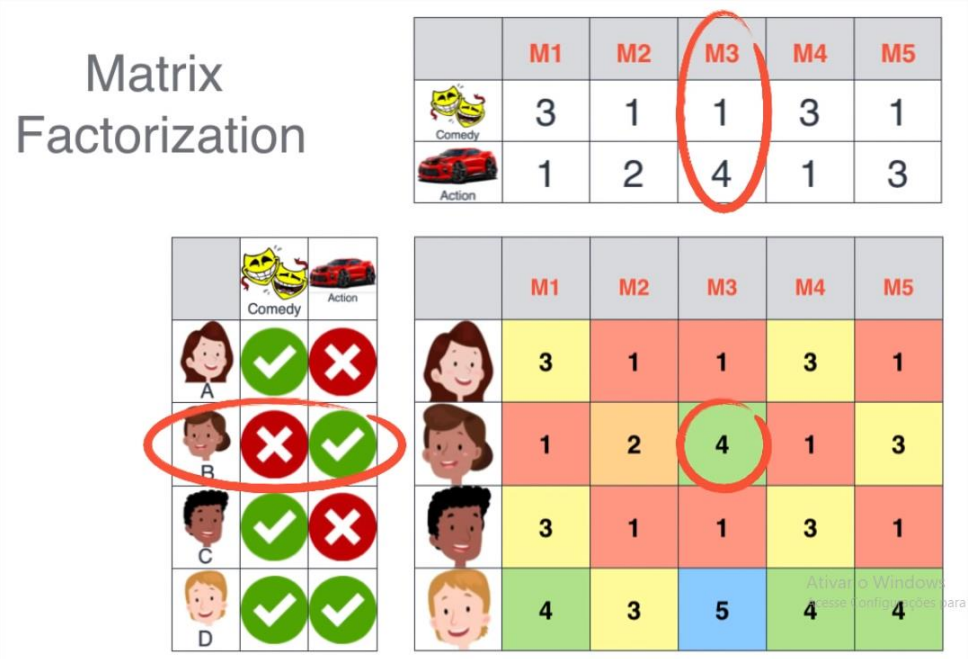
Figura 12 - Demonstração da esparsidade de dados em uma matriz de usuários x itens



Fonte: Serrano (2018).

Segundo Serrano (2018), Fatoração de Matrizes geralmente é utilizada em grandes bancos de dados, pois ela permite o uso de abordagens escaláveis. Com ela é possível elaborar perfis usuários e itens pelo uso de técnicas de álgebra linear. Essa técnica pode ser exemplificada pela figura 13. Nela é possível observar como a fatoração de matrizes pode ajudar a diminuir a esparsidade de dados, pois sua utilização permite preencher todas os elementos da matriz usuários *versus* itens com base em outras duas matrizes menores.

Figura 13 - Fatoração de Matrizes



Fonte: Serrano (2018).

Percebe-se que na matriz localizada mais à esquerda da figura 13 as linhas representam usuários; e as colunas, gêneros de filmes (comédia e ação). O símbolo verde significa que aquele usuário gosta daquele gênero e o símbolo com o “x” vermelho significa que ele não gosta. Já na matriz localizada na parte superior da imagem, os gêneros são as linhas e os filmes são as colunas. Ela representa o quanto cada filme se enquadra naquele gênero.

A matriz do meio é resultado da fatoração, sendo que para o usuário B, que gosta de ação, mas não gosta de comédia, o filme M3 recebe a pontuação 4, que é o indicador do gênero ação para este filme. Para o usuário D, que gosta tanto de ação como de comédia, o filme M3 recebe pontuação 5, que é resultado da soma dos indicadores do gênero ação e comédia desse filme, conforme pode ser observado na matriz superior.

Cerqueira e Coello (2013, p. 4) numa avaliação comparativa de algoritmos para sistemas de recomendação abordam o assunto de fatoração de matrizes conforme abaixo:

> Em contraste com algoritmos de filtragem colaborativa baseada em memória, os algoritmos baseados em semântica latente, constroem um modelo explícito durante uma fase de aprendizado, através do reconhecimento de padrões nos dados de treinamento. Após a fase de aprendizado, eles usam o modelo para prever avaliações. Muitos dos avanços recentes para aumentar a precisão em sistemas de recomendação utilizam esta abordagem. Especificamente, esses métodos fatoram a matriz de avaliações em duas matrizes de baixa ordem: uma para representar o perfil dos usuários e a outra o perfil dos itens.

Para realizar a avaliação comparativa de algoritmos de recomendação a métrica utilizado por Cerqueira e Coello (2013) foi o erro absoluto médio, ou MAE (Mean Absolute Error), enquanto que Bokde, Girase e Mukhopadhyay (2015) citam mais de uma forma de avaliação. Dentre elas estão:

4.2.1 Erro Absoluto Médio

Erro absoluto médio (*Mean Absolute Error* - MAE) é a medida capaz de mensurar a diferença entre a predição realizada pelo algoritmo e o valor que o usuário realmente atribuiria a ele. A figura 14 apresenta a fórmula do erro absoluto médio. Sendo que **pi** é a predição que o algoritmo fez para o usuário **i**; **ri** é o valor que o usuário realmente atribuiria ao item e **k** é o número de itens que o usuário **i** avaliou.

Figura 14 - Fórmula do Erro Absoluto Médio

$$MAE = \frac{(\sum(pi - ri))}{k}$$

Fonte: Bokde, Girase e Mukhopadhyay (2015, p. 144).

4.2.2 Raiz do Erro Quadrático Médio

Segundo Hallak e Pereira Filho (2011) a raiz do erro quadrático médio (*Root Mean Squared Error* - RMSE), é geralmente usada para expressar a acurácia dos resultados. Essa medida pune mais os erros, visto que eleva a diferença entre o

valor predito e o real ao quadrado. A figura 15 apresenta sua fórmula.

Figura 15 - Fórmula do erro quadrático médio

$$RMSE = \sqrt{\frac{\sum(pi - ri)^2}{k}}$$

Fonte: Bokde, Girase e Mukhopadhyay (2015, p. 145).

4.2.3 Precisão

Precisão (*Precision*) é parcela de pontuações que o algoritmo acertou dentre as predições realizadas. A sua fórmula está na figura 16, demonstrando que o cálculo é realizado pelo número de recomendações corretas/relevantes dividido pelo número total de itens que foram recomendados.

Figura 16 - Fórmula da precisão

$$Precisao = \frac{|ItensInteressantes \bigcap ItensRecomendados|}{|ItensRecomendados|}$$

Fonte: Bokde, Girase e Mukhopadhyay (2015, p. 145).

4.2.4 Revocação ou Sensitividade

Sensitividade (*Recall* ou *Sensitivity*) é a parcela de pontuações que o algoritmo acertou dentre a quantidade total de itens que são relevantes para o usuário. A sua fórmula está representada na figura 17, sendo que o cálculo é realizado pelo número de recomendações corretas/relevantes dividido pelo total de itens que seriam relevantes ao usuário.

Figura 17 - Fórmula da sensitividade

$$Sensitividade = \frac{|ItensInteressantes \bigcap ItensRecomendados|}{|ItensInteressantes|}$$

Fonte: Bokde, Girase e Mukhopadhyay (2015, p. 145).

4.3 APLICAÇÃO DO K-MEANS

Phorasim e Yu (2016) propõem um método de recomendação de filmes utilizando o algoritmo de agrupamento *k-means*. O objetivo do uso dele é particionar os dados em um número k de *clusters* que irão ajudar a identificar agrupamentos nos dados.

O *k-means* funciona de maneira não supervisionada, ou seja, o atributo classificador é desconhecido (os dados não são rotulados). O seu funcionamento ocorre da seguinte forma:

1 - É escolhido um número inicial de centroides, sendo que centroide é um ponto que fica no centro de um *cluster*.

2 - Cada um dos demais pontos, que no caso do recomendador de filmes, podem ser usuários, são atribuídos ao centroide mais próximo.

3 - A posição do centroide é recalculada de modo que fique no centro dos elementos vinculados a ele.

Os passos de número 2 e 3 descritos anteriormente são repetidos até que entre a execução de um ciclo e outro a posição do centroide não seja mais alterada.

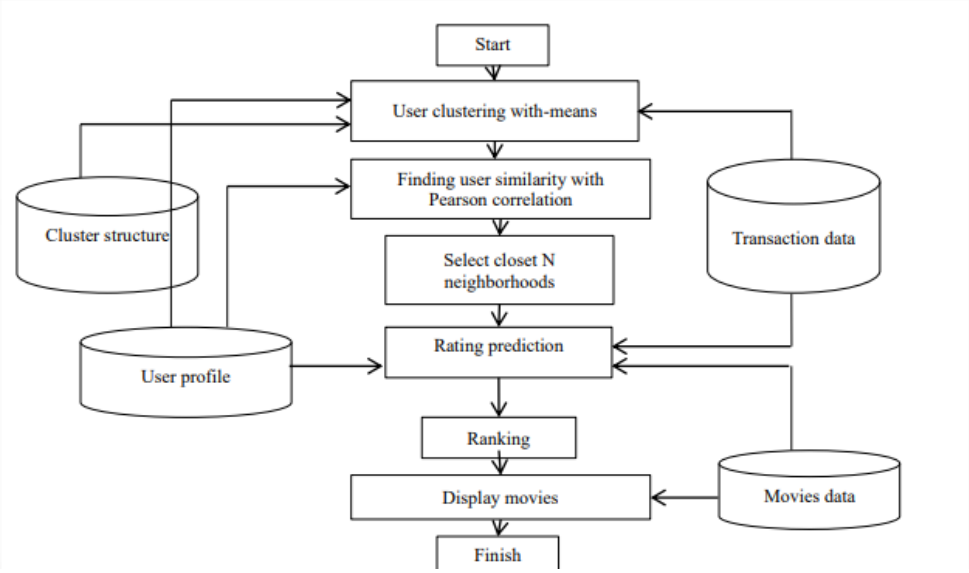
As etapas de implementação do trabalho de Phorasim e Yu (2016) consistem em:

1 - Estudar os problemas e necessidades do sistema;

2 - Busca de dados, que consiste em coletar informação necessária, dentre aquela disponível, para ajudar a atender as necessidades levantadas.

3 - Análise do sistema, que consiste na modelagem do sistema que será desenvolvido. A figura 18 demonstra o modelo desenhado por Phorasim e Yu (2016, p. 55), em que o sistema inicia realizando o agrupamento com k-means, depois mede a similaridade dos usuários que estão dentro de um mesmo cluster e, a partir dessa medida consegue selecionar os N vizinhos mais próximos do usuário em questão. Então é feita uma recomendação para esse usuário com base nos filmes com melhor ranking dentre os vizinhos mais próximos e que o usuário ainda não assistiu.

Figura 18 - Modelo do sistema de recomendação de Phorasim e Yu



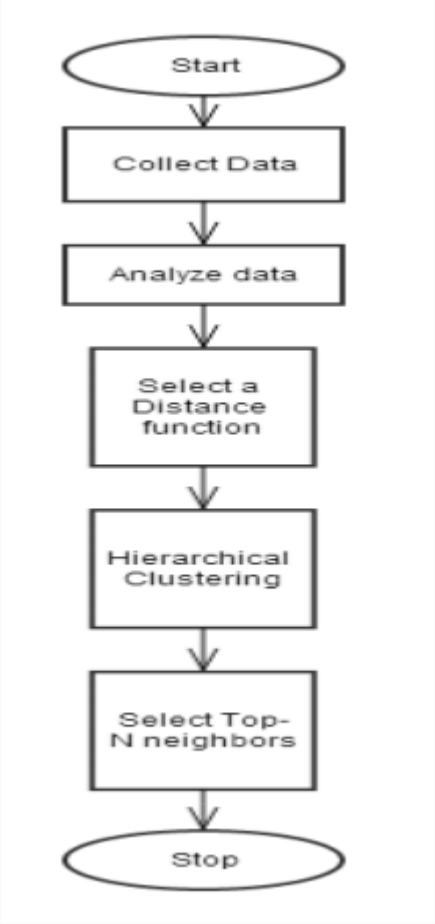
Fonte: Phorasim e Yu (2016, p.55).

4.4 APLICAÇÃO DE *HIERARQUICAL CLUSTERING*

Kumar e Chalotra (2014) afirmam que na literatura se encontram diversas propostas de uso e adaptação de algoritmos nos sistemas de recomendação, muitos deles buscam melhorar o uso do *k-means*. Porém, *k-means* demonstra resultados melhores quando a natureza do conjunto de dados é contínua ao invés de discreta. E nos problemas de recomendação, geralmente se trabalha com dados de natureza discreta.

Eles ainda citam que outras pesquisas fazem uso de medidas de distância entre os itens, como por exemplo, a distância euclidiana. Esta é computacionalmente onerosa, pois realiza potenciação e raiz quadrada sobre uma matriz de dados que geralmente possui milhares de registros. Foi considerando esses fatores que Kumar e Chalotra (2014) propõem o desenvolvimento de um sistema de recomendação usando filtragem colaborativa baseada na similaridade de usuário com usuário e formação de grupos com o uso de *hierarquical clustering*. Na figura 19 está desenhada estrutura desse sistema.

Figura 19 - Estrutura do sistema proposto por Kumar e Chalotra



Fonte: Kumar e Chalotra (2014, p.2).

A função de distância utilizada para medir a similaridade ou dissimilaridade entre dois usuários em comparação foi *Jaccard Similarity Coefficient.* E a partir da matriz de distâncias gerada aplicando essa função foram gerados os clusters.

A avaliação do trabalho de Kumar e Chalotra (2014) é feita em relação a acurácia das recomendações e do tempo de execução da solução proposta.

4.5 PANORAMA DOS TRABALHOS RELACIONADOS

Visando obter uma visão macro do que está presente em cada um dos trabalhos relacionados estudados foi elaborada a tabela 2, apresentando as principais contribuições de cada um deles. As linhas apresentam características e cada uma das colunas representa um dos trabalhos, sendo que T1 é o trabalho apresentado por Cui (2017), T2 o trabalho de Bokde, Girase e Mukhopadhyay (2015), T3 o trabalho de Phorasim e Yu (2016) e T4 o trabalho de Kumar e Chalotra (2014).

Tabela 2 – Contribuições dos trabalhos relacionados

Características	T1	T2	T3	T4
Tipo de SR	FC	FC	FC	FC
Algoritmo	*Knn*		*K-means*	*Hierarquical Clustering*
Medida de Similaridade	Similaridade do Cosseno			Coeficiente de Similaridade Jaccard
Como trata dados faltantes		Fatoração de Matrizes		
Formas de Avaliação		MAE, RMSE, precisão e sensitividade		Acurácia e tempo de execução

Fonte: próprio autor

5 MODELAGEM E DESENVOLVIMENTO DE UM RECOMENDADOR DE LIVROS

Durante o desenvolvimento do presente trabalho foi modelado e implementado o protótipo de um recomendador de livros que permite aos usuários obter recomendações de maneira personalizada.

O tipo de sistema de recomendação utilizado na implementação do protótipo foi filtragem colaborativa. A filtragem colaborativa pode ser feita com base na similaridade entre itens ou entre usuários. Neste trabalho é utilizada a similaridade entre usuários. A escolha da FC se deve ao fato de tal abordagem ser amplamente utilizada tanto em estudos acadêmicos como em aplicações comerciais. Mesmo que em sistemas comerciais ela geralmente venha acompanhada de outras técnicas que caracterizem o sistema como do tipo híbrido, ainda sim, fundamentalmente muitos trabalham com FC.

5.1 MÉTODO UTILIZADO

A primeira etapa se iniciou com estudo de *big data*, que permite compreender a necessidade de recomendação personalizada para auxiliar pessoas na escolha de livros perante a grande quantidade de dados disponíveis. Foi estudada também a exploração do *big data* através da mineração de dados, suas etapas e quais tarefas podem ser executadas através dela. Após isso foram detalhados os tipos de sistemas de recomendação e as estratégias de recomendação existentes.

Com a análise de trabalhos relacionados foram identificadas quais técnicas são geralmente utilizadas de acordo com cada contexto e necessidade de recomendação.

A segunda etapa consistiu na elaboração da modelagem de um protótipo, chamado de recomendador de livros. Para isso foi executado o processo de KDD com o auxílio do Software para mineração de dados Orange. O KDD é o processo proposto por Fayyad at. All (1996) que foi visto na seção 2.1 deste trabalho intitulada “exploração do *Big Data* através da mineração de dados”.

A terceira etapa tratou da implementação do recomendador. A partir das escolhas resultantes da experimentação com no Orange iniciou-se a implementação de um motor de recomendação. Durante esta etapa foi utilizado o ambiente

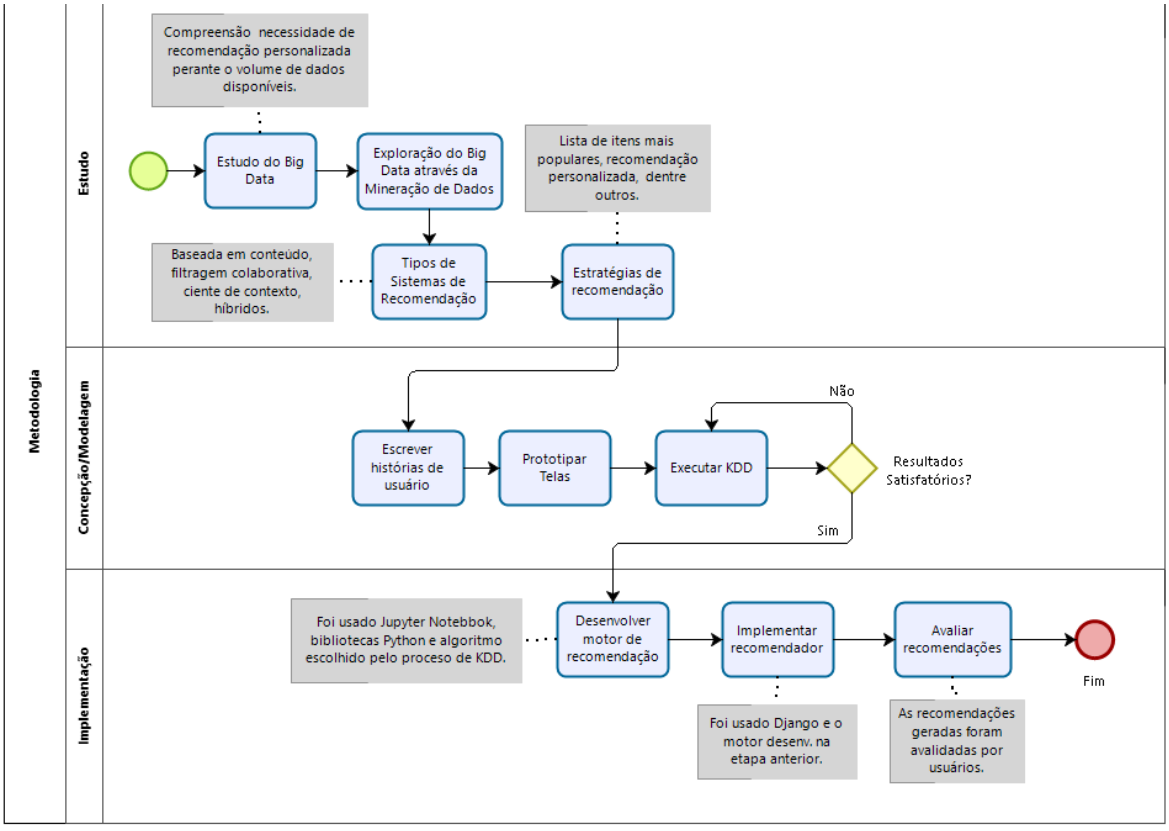
computacional Jupyter Notebook, onde foi escrito um script em Python que faz uso dos dados provenientes do *dataset* de livros, juntamente com o algoritmo escolhido no Orange, para prever leituras interessantes e recomendá-las aos usuários.

Porém, até esta etapa do desenvolvimento era possível obter recomendações apenas por linha de comando. E de nada adianta construir um motor de recomendação se não for possível disponibilizá-lo para ser utilizado por outras pessoas. Para atender essa necessidade, a forma de disponibilizar as recomendações foi projetada fazendo uso de histórias de usuários juntamente com uma proposta de interface gráfica, servindo de guia para a última fase de desenvolvimento.

Esta última fase consistiu em construir um protótipo de recomendador utilizando o *framework* de desenvolvimento Django. A escolha do Django se deve ao fato de ele utilizar a linguagem de desenvolvimento Python, que já havia sido utilizada para elaborar o motor de recomendação.

A figura 20 resume o processo utilizado do início do estudo até a finalização da implementação e a avaliação das recomendações.

Figura 20 - Desenho do processo



Fonte: Próprio autor

Nos próximos capítulos serão apresentadas com maior detalhe as etapas de Concepção/Metodologia e Implementação do recomendador.

5.2 HISTÓRIAS DE USUÁRIO

As histórias de usuário são utilizadas para entender as necessidades dos usuários. São como os requisitos do sistema vistos de maneira macro e na visão de do utilizador do sistema.

As histórias do Recomendador de Livros estão descritas nas tabelas 3 a 6.

Tabela 3 - História de usuário número 1.

História 1 – Obter recomendação personalizada
Maria, enquanto leitora, gostaria de obter recomendações personalizadas de livros que vão de encontro com o seu gosto literário, porque na falta de amigos com gosto parecido, precisa obter essas indicações de alguma outra forma.

Fonte: Próprio autor.

Tabela 4 - História de usuário número 2.

História 2 – Avaliar livros
Maria, enquanto leitora, entende que para que seja realizada a tarefa de recomendação será necessário demonstrar como é seu gosto literário de alguma forma. Ela sabe que um software não seria capaz adivinhar o que ela gosta e o que ela não gosta de ler. (Isso pode ser realizado através da avaliação de alguns títulos lidos por ela no passado.)

Fonte: Próprio autor.

Tabela 5 - História de usuário número 3.

História 3 – Pesquisar livros para avaliar
Maria, enquanto leitora, quando for realizar a avaliação de livros, gostaria de poder pesquisar livros que lembra de ter lido para então poder dar uma pontuação a eles.

Fonte: Próprio autor.

Tabela 6 - História de usuário número 4.

História 4 – Ter sugestões de livros para avaliar
Maria, enquanto leitora, quando for realizar a avaliação de livros, gostaria que fossem sugeridos alguns para ela poder dar uma pontuação. Isso se deve a nem sempre se lembrar com facilidade do título dos livros lidos, e, se houverem algumas sugestões, pode ser que ajude a lembrar, ou até mesmo que apareça algum livro que ela já leu. Isso facilitará a tarefa de avaliação.

Fonte: Próprio autor.

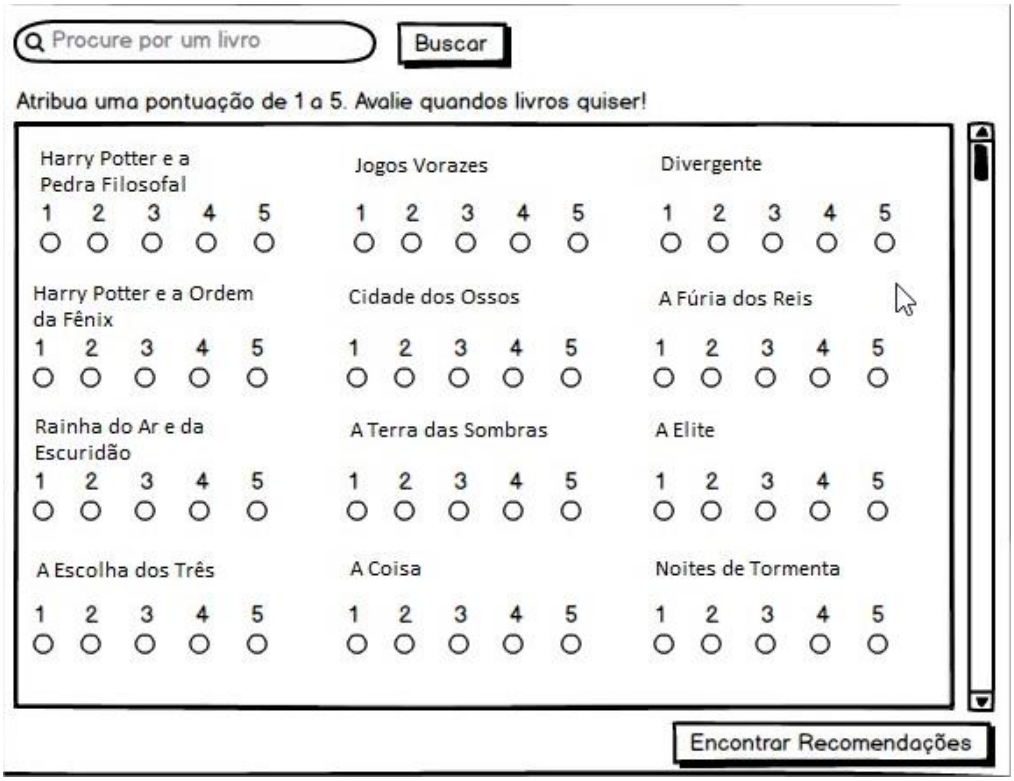
5.3 PROTÓTIPOS DE TELA

Para demonstrar como foi planejada a interface gráfica do recomendador de livros são apresentadas algumas propostas de telas. Elas não representam de maneira exata a versão final das interfaces de usuário da solução desenvolvida, mas servem para ter uma ideia de como elas irão se parecer e quais elementos estarão contidos nelas.

A figura 21 representa a tela para avaliação de livros e a figura 22 representa a tela de visualização de recomendações. A avaliação de livros deve ser realizada numa escala de 1 a 5, onde 1 representa a pior e 5 representa a melhor pontuação. Inicialmente será sugerida uma lista de livros para avaliação. Essa lista é composta pelos livros mais populares do *dataset* (ou seja, os que possuem maior número de avaliações). Foi modelada também a funcionalidade de pesquisa por livros (usando nome do livro ou do autor) que se deseja avaliar, no caso de ele não estar na lista inicial.

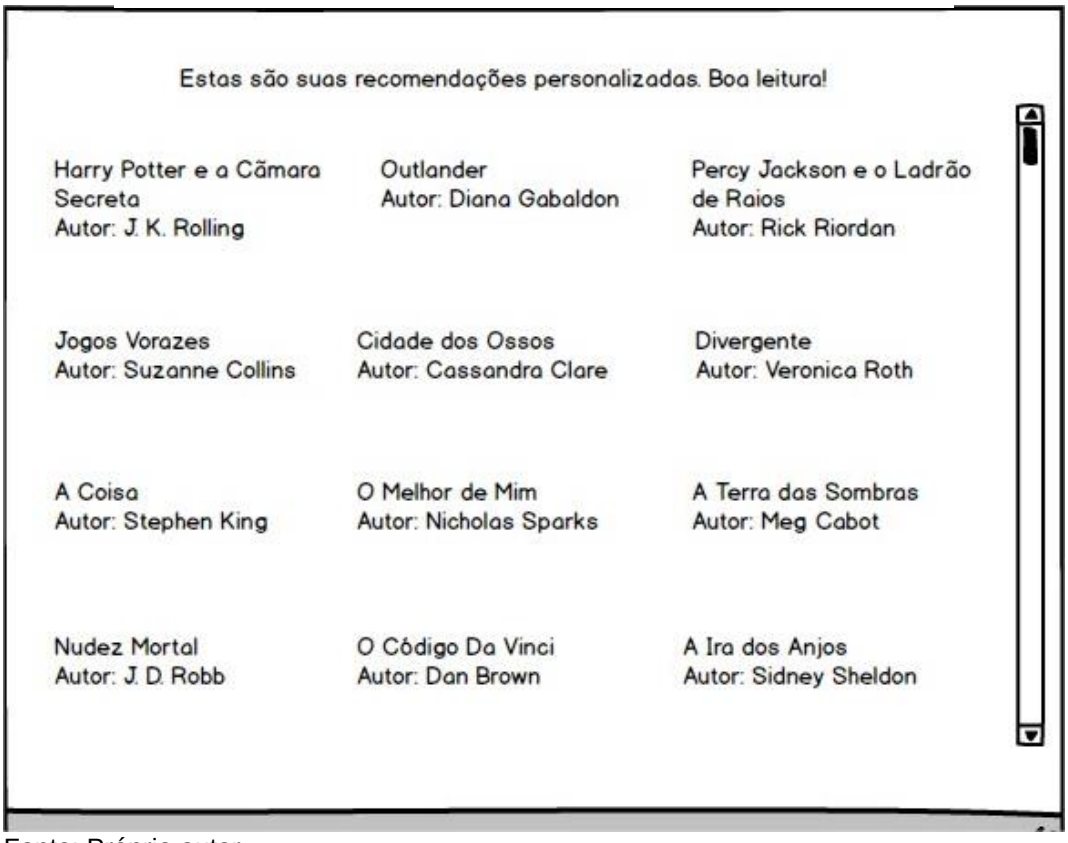
Após realizada a avaliação, o recomendador de livros irá retornar as recomendações personalizadas de acordo com o gosto literário que foi identificado para aquele usuário.

Figura 21 - Tela de avaliação de livros



Fonte: próprio autor.

Figura 22 - Tela de visualização de recomendações



Fonte: Próprio autor.

5.4 O CONJUNTO DE DADOS (*DATASET*)

O conjunto de dados (*dataset*) escolhido para ser utilizado neste trabalho chama-se *goodbooks-10k*. Ele contém informações a respeito de livros populares. Há dados descritivos a respeito do livro, como autor, ano de publicação, título e também pontuações (numa escala de 1 a 5, onde 1 é a pior avaliação e 5 é a melhor) que foram atribuídas pelos usuários para cada um desses livros.

Quando se pretende trabalhar minerando dados é importante ter compreensão sobre o conjunto de dados escolhido. Para isso podem ser realizadas análises iniciais simples, como conhecer os atributos existentes, os tipos de dados de cada um deles, a quantidade de registros presente nos arquivos e procurar por inconsistências (dados faltantes, duplicados ou até mesmo incorretos).

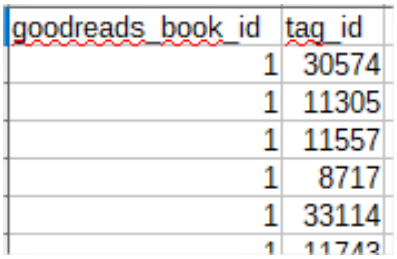
Depois dessa compreensão inicial pode-se começar análises um pouco mais detalhadas. Podem ser citadas: a análise da média de determinado atributo numérico ou então a verificação de quais valores distintos existem para uma coluna. Por exemplo: conferir se a coluna que contém as pontuações de 1 a 5 realmente só possui valores dentro dessa faixa numérica.

Alguns dados apurados a respeito do *dataset* utilizado constam a seguir: para a maioria dos títulos há pelo menos 100 avaliações (através das pontuações) realizadas, porém para alguns deles há um pouco menos do que isso. Todos os usuários realizaram, pelo menos, duas avaliações; e a quantidade média de avaliações é de 8 por usuário. Há também informações sobre livros que o usuário identificou que gostaria de ler, mas não leu ainda, e por isso, ainda não foi avaliado por ele.

Esse conjunto de dados está no formato .csv, e em arquivos distintos. A seguir há um resumo do que o que contém cada um dos arquivos.

- book_tags.csv: contém tags ou gêneros atribuídos pelos leitores aos livros. Tags neste arquivo são representadas por seus IDs. Os registros estão ordenados por pelo id do livro de maneira ascendente. A figura 23 demonstra as colunas desse arquivo.

Figura 23 - Colunas do arquivo book_tags.csv



goodreads_book_id	tag_id
1	30574
1	11305
1	11557
1	8717
1	33114
1	11743

Fonte: Próprio autor.

- books.csv: Possui informações a respeito de cada livro, como: código identificador, autor(es), ano de publicação, título, quantidade de vezes que foi avaliado através de pontuação, quantidade de resenhas escritas, quantidade de avaliações recebidas em cada uma das pontuações existentes. A figura 24 demonstra algumas colunas desse arquivo.

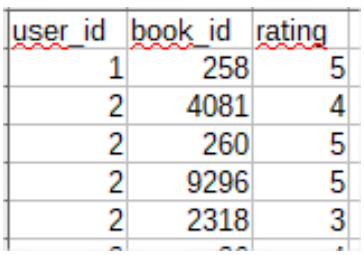
Figura 24 - Colunas do arquivo books.csv

authors	original_publication_year	original_title
Suzanne Collins	2008.0	The Hunger Games
J.K. Rowling, Mary GrandPrÃ©	1997.0	Harry Potter and the Philosopher's Stone
Stephenie Meyer	2005.0	Twilight
Harper Lee	1960.0	To Kill a Mockingbird
F. Scott Fitzgerald	1925.0	The Great Gatsby
John Green	2012.0	The Fault in Our Stars
J.R.R. Tolkien	1937.0	The Hobbit or There and Back Again
J.D. Salinger	1951.0	The Catcher in the Rye
Dan Brown	2000.0	Angels & Demons
Jane Austen	1813.0	Pride and Prejudice
Khaled Hosseini	2003.0	The Kite Runner

Fonte: Próprio autor.

- ratings.csv: cada registro representa uma avaliação de um usuário (representado pelo seu id) para determinado livro (também representado pelo seu id), numa pontuação de 1 a 5. A figura 25 demonstra as colunas desse arquivo.

Figura 25 - Colunas do arquivo ratings.csv



user_id	book_id	rating
1	258	5
2	4081	4
2	260	5
2	9296	5
2	2318	3
2	26	4

Fonte: Próprio autor.

- tags.csv: possui a relação do id da *tag* com o seu respectivo conteúdo (texto). A figura 26 demonstra as colunas desse arquivo.

Figura 26 - Colunas do arquivo tags.csv

tag_id	tag_name
22	-d-c--
23	-dean
24	-england-
25	-fiction
26	-fictional
27	-fictitious
28	-football-
29	-george

Fonte: Próprio autor.

- to_read.csv: contém as intenções de leitura dos usuários, ou seja, são livros que ainda não foram lidos por eles e, portanto, não tem pontuação ainda. Existe quase um milhão de registros. A figura 27 demonstra as colunas desse arquivo.

Figura 27 - Colunas do arquivo to_read.csv

user_id	book_id
9	8
15	398
15	275
37	7173
34	380
34	483
34	8598
34	3581
70	498
76	4250

Fonte: Próprio autor

5.5 EXECUÇÃO DO PROCESSO DE KDD

Conforme estudado na seção 2.1, “exploração do *big data* através da mineração de dados”, Fayyad at. All (1996) declaram que a mineração de dados é uma das etapas do processo da descoberta do conhecimento que consiste em aplicar algoritmos de análise e descoberta de dados visando identificar padrões sobre os dados.

A aplicação de tarefas de agrupamento (*clustering*) durante a Mineração de Dados sobre o *dataset* escolhido tem como objetivo encontrar técnicas que

permitem identificar grupos de usuários similares. No final do processo de KDD foi escolhida uma dessas técnicas para utilizar na implementação do recomendador de livros.

As etapas de seleção, preparação (ou pré-processamento) e mineração de dados, provenientes do processo de KDD, foram executadas no decorrer do trabalho e serão detalhadas a seguir.

5.5.1 Seleção, Pré-processamento e Transformação de dados

Com base no objetivo a ser alcançado com o processo de KDD, foi realizada a filtragem dos dados para serem usados nas etapas posteriores. Como o processo de KDD é cíclico, dependendo das necessidades encontradas no decorrer do processo foi necessário voltar a esta etapa mais de uma vez no decorrer do trabalho para realizar ajustes.

Considerando as informações disponíveis no conjunto de dados foram selecionados dois arquivos para serem trabalhados. São eles: books.csv (dados dos livros, como autor, título e ano) e ratings.csv (pontuações de 1 a 5 que os usuários atribuíram aos livros). O arquivo ratings.csv foi usado tanto no processo de KDD como posteriormente, na implementação do recomendador; enquanto que books.csv foi utilizado apenas na implementação do recomendador.

Em books.csv haviam algumas correções nos dados para tratar. As principais foram: livros sem o autor informado, livros sem o ano informado e a coluna referente ao ano estava com o formato incorreto (tinha uma casa decimal, ao invés de ser um inteiro). Os casos de informações faltantes foram em torno de 15 ocorrências e para resolvê-las foi pesquisado na internet a informação faltante e preenchido na planilha. E o tipo de dado incorreto também foi resolvido através do uso de recursos do próprio editor de planilha.

Inicialmente o arquivo ratings.csv possuía três colunas, que eram: “user_id”, “book_id” e “rating”. Para trabalhar com esse arquivo foi realizada uma alteração no formato de linhas e colunas dele de modo que se obtivesse uma matriz de usuários *versus* itens. Uma matriz nesse mesmo formato foi utilizada na solução de recomendação proposta pelo trabalho de Bokde, Girase e Mukhopadhyay (2015), que foi citado no capítulo 4, “trabalhos relacionados”.

Após a remodelagem do arquivo ratings.csv foi obtida uma tabela no formato

apresentado na tabela 3, onde cada linha passou a representar um usuário (*user_id*) e cada coluna, um livro (*book_id*). E os elementos no cruzamento das linhas com as colunas representem a pontuação (*rating*) que determinado usuário atribuiu a tal livro.

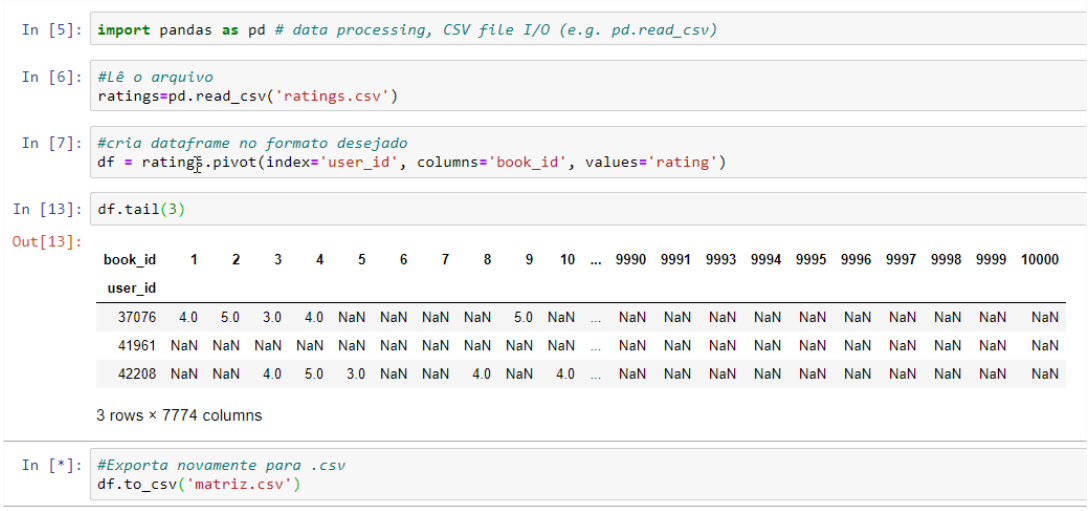
Tabela 7 - Formato dos dados após a seleção

Id do usuário	Id do livro 1	Id do livro 2	Id do livro 3	Id do livro 4	(...)
1		4	3	5	
2	2		4	3	
3		5			
4	1	2	3	5	

Fonte: Próprio autor.

Para realizar a remodelagem foi importado o arquivo com formato .csv para dentro de um script em Python utilizando a ferramenta Jupyter Notebbok. Dentro do script foi utilizado o comando “pivot” da biblioteca “Pandas”, que permitiu deixar a matriz exatamente no formato desejado. A figura 28 abaixo contém o *script* de geração dessa matriz.

Figura 28 - Script de remodelagem de ratings



```
In [5]: import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

In [6]: #Lê o arquivo
        ratings=pd.read_csv('ratings.csv')

In [7]: #cria dataframe no formato desejado
        df = ratings.pivot(index='user_id', columns='book_id', values='rating')

In [13]: df.tail(3)

Out[13]:
```

book_id	1	2	3	4	5	6	7	8	9	10	...	9990	9991	9993	9994	9995	9996	9997	9998	9999	10000
user_id																					
37076	4.0	5.0	3.0	4.0	NaN	NaN	NaN	NaN	5.0	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
41961	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
42208	NaN	NaN	4.0	5.0	3.0	NaN	NaN	4.0	NaN	4.0	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
3 rows × 7774 columns

In [*]: #Exporta novamente para .csv
        df.to_csv('matriz.csv')
```

Fonte: próprio autor.

Um dos problemas encontrados em relação a essa matriz de usuários *versus* itens é que ela ficou esparsa, pois geralmente um usuário vai ter pontuado apenas uma pequena fração de itens perante todos os disponíveis. E a maioria das técnicas disponíveis no Orange não aceitam valores de entrada (*imputs*) nulos.

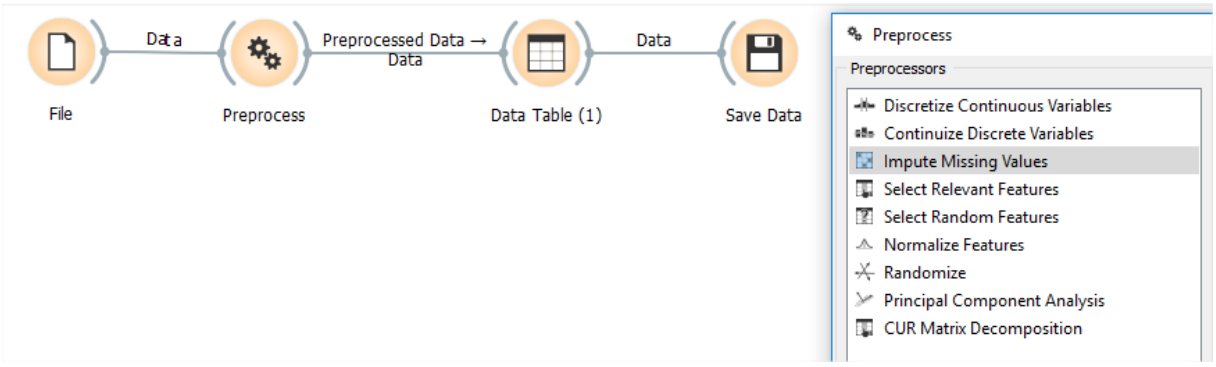
Quando se deseja resolver dados nulos existem diversas opções a serem escolhidas. Uma delas é excluir todas as linhas ou colunas que possuem algum valor nulo. Essa não é uma boa alternativa nesse caso, visto que quase todas as linhas ou colunas tem pelo menos algum valor nulo. Então se fosse usada essa alternativa acabaria excluindo praticamente toda a matriz.

Outras opções são: preencher os nulos com a média daquela coluna, ou seja, a média daquele livro; preencher com a média a linha (usuário) ou então substituir os nulos por zero.

A estratégia escolhida foi a substituição por zeros. Essa escolha foi levando em consideração o objetivo de encontrar grupos de usuários similares. A informação de uma pessoa não ter lido certo livro pode ser um indicador para encontrar usuários similares. Da mesma forma que ter lido e pontuado livros de maneira igual (ou similar) seja uma forma de encontrar usuários parecidos, o fato de um grupo de pessoas não ter lido determinados livros também é um indicador para auxiliar a encontrar *clusters* de usuários nos dados.

Para realizar esse tratamento dos nulos foi utilizado o *widget* "*Preprocess"* disponível no Orange. A figura 29 demonstra o *workflow* criado, onde "*File"* é responsável por importar o arquivo .csv, "*Preprocess"* foi usado para resolução dos nulos, "*Data Table*" meramente para visualização e "*Save Data*" para salvar o resultado no formato desejado. No canto direito da figura 29 é possível observar também a janela do "*Preprocess*", onde uma das opções disponíveis trata da inserção de valores faltantes ("*Impute Missing Values*").

Figura 29 - Workflow de pré-processamento



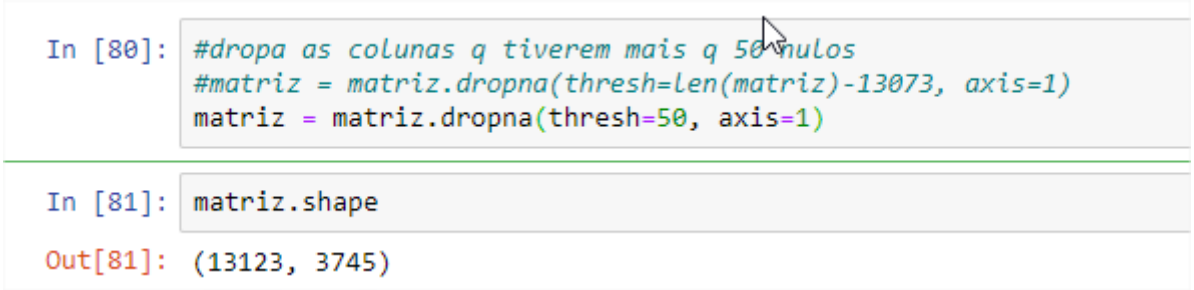
Fonte: Próprio autor

Outro problema encontrado nas experimentações com o Orange foi que o tempo de execução dos algoritmos estava muito grande (passando de duas horas) e

geralmente ocorria erro de falta de memória no computador antes de acabar a execução. Nesse momento não há como deixar de lembrar da colocação de Castro e Ferrari (2016), em que eles afirmam que um dos desafios em relação ao *Big Data* (dentre outras coisas) é conseguir processar e extrair conhecimento a partir dele. E esse problema do tempo de execução e falta de memória trata-se justamente do processamento dos dados para extração de conhecimento.

Para resolver essa questão foi necessário realizar a seleção de atributos (no caso, livros) mais relevantes. No Jupyter Notebook foi escrito um script em Python para manter apenas os livros que possuíam pelo menos 50 avaliações. Com isso a quantidade de livros foi de 7775 para 3745. A figura 30 contém um print do trecho de código em que as colunas são selecionadas.

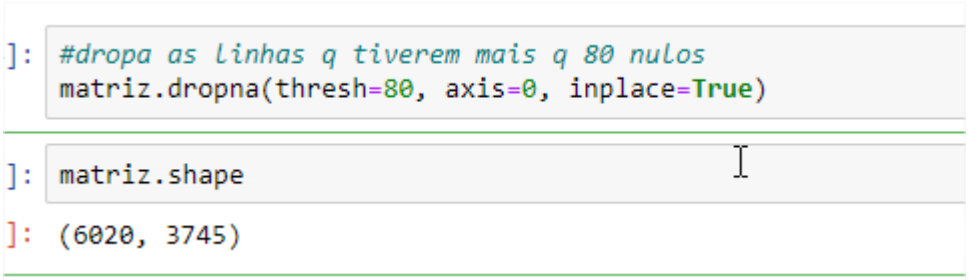
Figura 30 - Comando para seleção de atributos



Fonte: próprio autor.

Foi necessário também selecionar alguns registros (usuários). Foram escolhidos os usuários que avaliaram pelo menos 80 livros (por mais que não pareça provável, praticamente metade dos usuários tem pelo menos essa quantidade de avaliações). Com essa seleção o número de usuários caiu de 13123 para 6020. A figura 31 contém um print do trecho de código em que as linhas foram selecionadas:

Figura 31 - Comando para seleção de registros



Fonte: próprio autor.

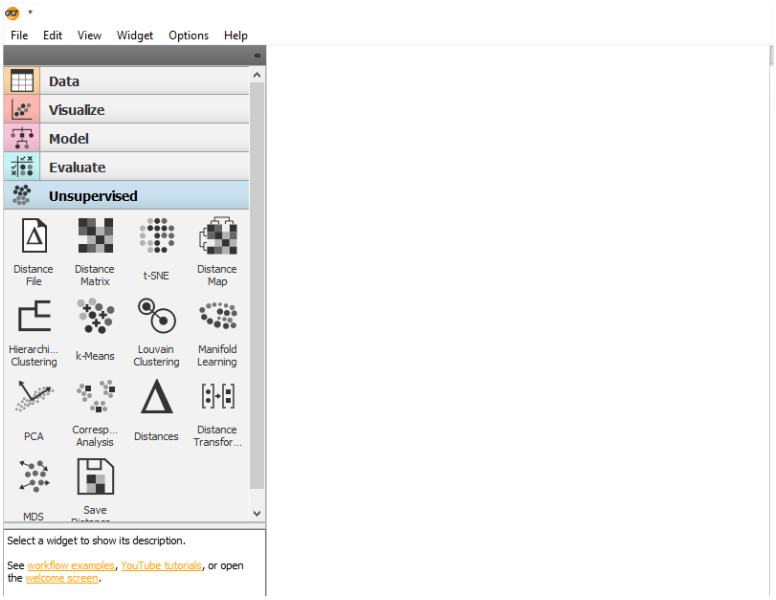
Finalmente, após o tratamento, correções, remodelagem e seleção dos dados iniciais, o arquivo está pronto para serem realizados os testes no Orange.

5.5.2 Mineração de dados com Orange

Esta etapa foi realizada com o auxílio do software Orange, que é um software de código aberto para Mineração de Dados desenvolvido pela universidade de Liubliana (Liubliana é a capital da Eslovênia). A figura 32 demonstra a tela inicial do software Orange.

O painel localizado na parte esquerda da tela contém ferramentas ou *widgets* que permitem trabalhar desde a importação, limpeza e visualização de dados até a aplicação de algoritmos como o *k-means* e *hierarquical clustering*.

Figura 32 - Tela inicial do Orange



Fonte: Próprio autor.

Com base nos estudos realizados, foram identificados alguns algoritmos para fazer a experimentação no Orange. Foram escolhidos: o *k-means,* que foi estudado ao analisar o trabalho relacionado de Phorasim e Yu (2016) e o *hierarquical clustering,* abordado no recomendador de Kumar e Chalotra (2014).

Recapitulando o trabalho de Phorasim e Yu (2016), verifica-se que nele foi desenvolvido um método de recomendação de filmes em que o sistema inicialmente realiza agrupamento nos dados com k-means, depois seleciona os usuários mais similares dentro de um mesmo cluster e, por fim, gera recomendações para esse usuário com base nos filmes com melhor ranking dentre os usuários mais próximos e

que a pessoa ainda não assistiu.

Já Kumar e Chalotra (2014) propõem o desenvolvimento de um sistema de recomendação usando filtragem colaborativa baseada na similaridade de usuário com usuário e formação de grupos com o uso de *hierarquical clustering*.

Foi nesse mesmo sentido de produzir agrupamentos por similaridade entre os usuários que foram aplicados os algoritmos de mineração no Orange. A figura 33 demonstra um exemplo de agrupamento de usuários para esclarecer o objetivo que se pretente alcançar nesta etapa de experimentação. Veja que os usuários 1, 2 e 3 pontuaram os livros A, B e C com notas elevadas (4 e 5), enquanto que aos livros D e E foram atribuídas notas baixas; e o livro F não foi lido por nenhum deles.

É evidente que existe uma similaridade na forma como os livros foram pontuados para tais usuários, o que indica que eles possuem gostos parecidos, e, portanto, pode ser formado um cluster com os mesmos. Da mesma forma, ao analisar os usuários 5 e 6, percebe-se que eles possuem uma opinião literária diferente dos usuários anteriores, visto que eles gostaram dos livros D e E e não gostaram dos livros A, B e C.

Figura 33 - Exemplo de formação de clusters de usuários por similaridade de pontuação

	livro A	livro B	livro C	livro D	livro E	livro F
usuario 1	5	4	5	3	2	nulo
usuario 2	5	5	5	3	3	nulo
usuario 3	4	5	5	2	3	nulo
usuario 4	2	1	1	4	5	2
usuario 5	1	1	2	4	5	1
usuario 6	2	1	2	5	5	2

Cluster 1

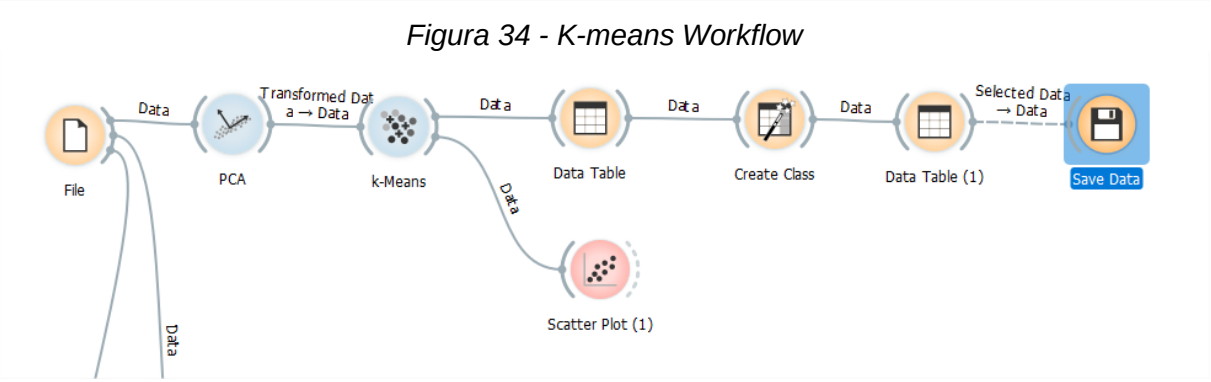
Cluster 2

Fonte: Próprio autor.

Tendo em mente o objetivo da mineração de dados sobre esse cenário apresentado foram construídos *workflows* no Orange para testar os algoritmos *kmeans* e *hierarquical clustetring* sobre a matriz de usuários *versus* livros que foi preparada na etapa de “Seleção, Pré-processamento e Transformação de dados”.

O primeiro teste foi com *k-means* e seu *workflow* consta na figura 34. Ele demonstra todas as etapas que foram realizadas desde a importação do arquivo através do widget "*File*", até o salvamento dos resultados em um novo arquivo, através do uso de "*Save Data*".

Figura 34 - K-means Workflow



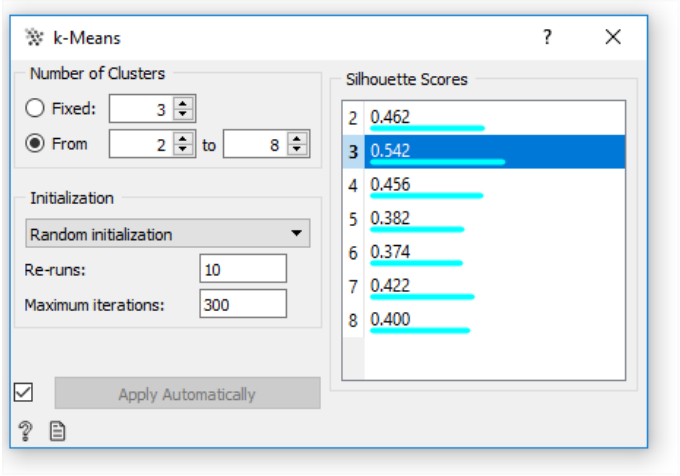
Fonte: próprio autor.

Neste *workflow* podem ser destacadas as seguintes etapas principais:

a) PCA: Como a matriz utilizada possui inúmeros atributos, acaba se tornando difícil visualizar os dados graficamente, pois nos gráficos geralmente são apresentadas apenas duas ou três dimensões por vez. A aplicação da Análise de Componentes Principais, ou "PCA" (sigla em inglês para o termo *Principal Component Analysis)*, permite gerar componentes principais baseando-se nos atributos da matriz, reduzindo assim a dimensionalidade da mesma. Os componentes principais são um conjunto reduzido de informações, mas que buscam representar ao máximo possível a totalidade dos dados.

b) *K-Means*: Esse algoritmo de agrupamento foi executado buscando encontrar agrupamentos interessantes de usuários com gosto literário parecido. Para auxiliar a encontrar o número ideal de grupos foi utilizado o *silhouette score* ou pontuação da silhueta. Esta medida indica, em média, o quão bem os *datapoints* se encaixam dentro do *cluster* designado. Quanto mais alta a medida da silhueta significa que os pontos estão melhores alocados. Na figura 35 pode ser observado que o melhor número de clusters segundo a pontuação da silhueta é 3.

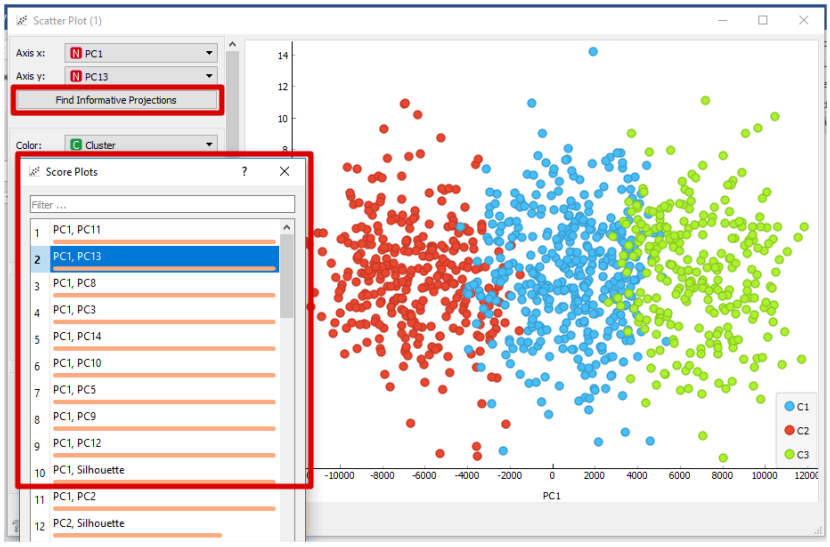
Figura 35 - Silhouette Score



Fonte: Próprio autor.

c) *Scatter Plot*: Cria um gráfico de disperção, permitindo ajustar alguns parâmetros de visualização. É possível pintar os pontos da tela com a cor do *cluster* que foi atribuido a cada ponto com o *k-means,* e também pode se escolher qual componente considerar no eixo x e no eixo y do gráfico. Além disso, existe a funcionalidade de encontrar projeções informativas. A mesma encontra-se disponível no botão "*Find Informative Projections*", destacado na figura 36. Ao clicar nesse botão abre a janela chamada "*Score Plots*", também destacada na figura 36. Nessa janela é possivel escolher entre as diferentes projeções previamente montadas e conforme o usuário seleciona as opções o gráfico de pontos (mais a direita na figura) vai muando a projeção.

Figura 36 - Scatter Plot



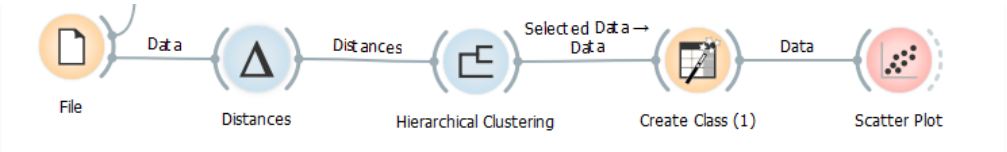
Fonte: Próprio autor.

d) *Create Class*: Serve para criar uma nova coluna contendo a classe de cada registro. Nesse caso a classe foi criada com base no grupo identificado

na execução do *k-means*.

O segundo teste foi com *Hierarquical Clustering* e seu *workflow* consta na figura 37. Ela demonstra todas as etapas que foram realizadas desde a importação do arquivo através do widget "*File*", até a visualização dos dados, através do uso de "*Scatter Plot*".

Figura 37 - Hierarquical Clustering Workflow

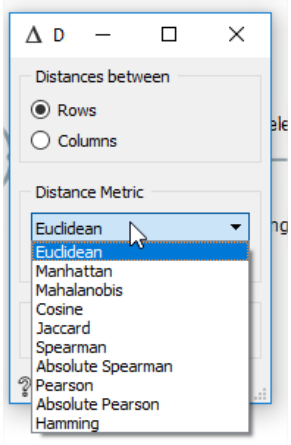


Fonte: Próprio autor.

Neste *workflow* podem ser destacadas as seguintes etapas principais:

a) *Distances*: Para poder aplicar o *hierarquical clustering* é necessário ter como parâmetro de entrada as distâncias entre os pontos computadas. Para isso foi utilizado o *widget distances*, que calcula a distância entre registros (linhas) ou atributos (colunas) do dataset, conforme for configurado. Veja na figura 38 que pode ser escolhida também qual das métricas de distância a utilizar. Foram realizados testes sempre com distâncias entre os *rows* (registros) e variando as medidas "*Euclidean*", "*Manhattan*" e "*Cosine*". Com as duas primeiras medidas foram obtidos resultados similares encontrando agrupamentos com instâncias bem distribuídas entre as classes, enquanto que nos testes com a última das medidas as instâncias ficaram praticamente todas no mesmo *cluster*. Mais a seguir serão apresentados esses resulados ao falar sobre o tópico *Hierarquical Clustering*.

Figura 38 - Distances Widget

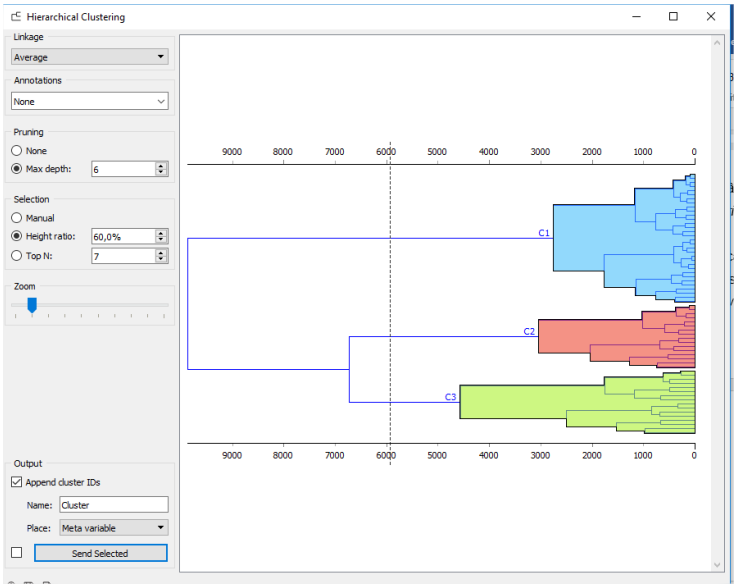


Fonte: Próprio autor.

b) *Hierarquical Clusering*: Esta técnica permite descobrir além de grupos, também subgrupos nos dados analizados. Para conseguir montar tais grupos ela requer como dado de entrada as distâncias entre os pontos já calculadas. Quanto menor for a distância, maior a similaridade entre as instâncias. Ao aplicar o *hierarquical clustering* no software Orange, os resultados são apresentados em um dendrograma, que se trata de um tipo de diagrama que representa uma estrutura hierárquica.

Na figura 39 consta o dendrograma resultante da execução do *workflow* no Orange com a métrica de distância do *widget* "*Distances*" igual a "*Euclidean*". Uma forma de validar a quantidade de clusters a ser formada é traçando uma linha na árvore representada, de modo que ela não intersecte nenhuma junção de grupos, ou então, pode ser escolhido um ponto manualmente, de acordo com o julgamento humano da melhor divisão. Na figura 39 essa linha está traçada de maneira pontilhada na vertical sobre o número 6000. Ela foi posicionada com o uso do parâmetro "*Heigh ratio*" igual a 60%.

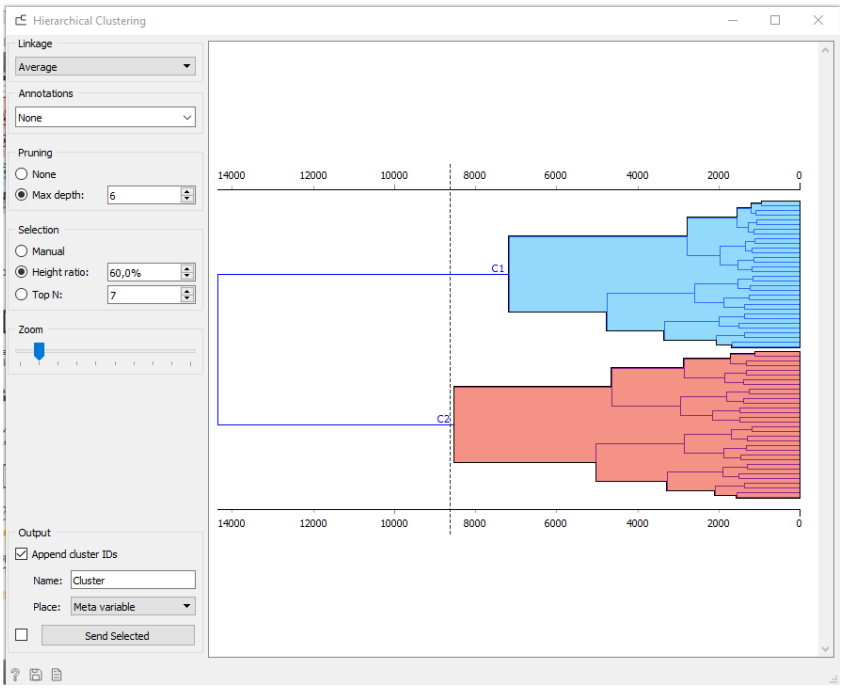
Figura 39 - Dendrograma com distância Euclideana



Fonte: Próprio autor.

Na execução do mesmo *workflow,* agora com a métrica de distância do *widget* "*Distances*" igual a "*Manhattan*", utilizando parâmetro de "*Heigh ratio*" com o mesmo valor, de 60%, passaram a ser gerados apenas 2 grupos. O dendrograma resultande dessa execução é apresentado figura 40.

Figura 40 - Dendrograma com distância de Manhattan

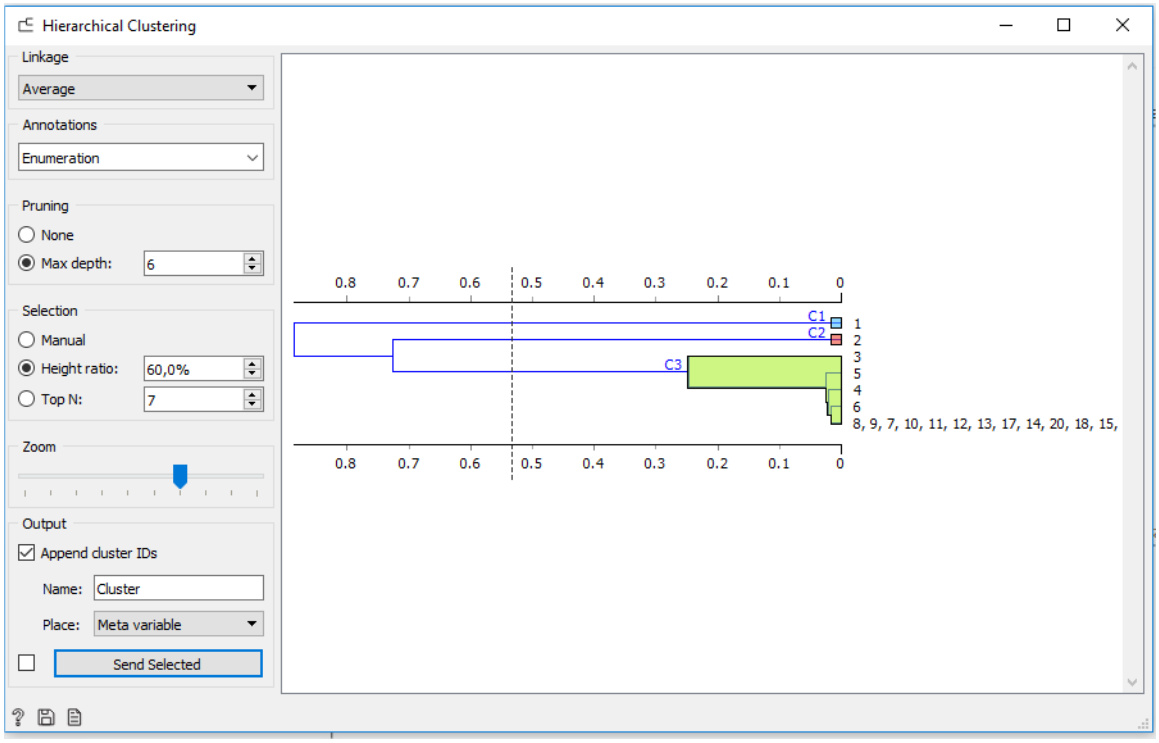


Fonte: Próprio autor.

O último teste foi realizado utilizando a métrica de distância igual a "*Cosine*". No dendrograma representado pela figura 41 pode ser observado que apenas uma instância de dados foi enquadrada no *cluster* 1, representado pela sigla C1 e uma instância no *cluster* 2 (C2), enquanto que todo o restante foi agrupado em C3.

Analisando o diagrama, percebe-se que se praticamente a totalidade dos dados está em um mesmo cluster, esse agrupamento não obteve sucesso, pois não conseguiu encontrar nem ao menos algumas instancias similares ao item que está em C1 e ao que está em C2. Esse resultado não atende ao objetivo de formar grupos de usuários com perfil literário similar, pois se praticamente todos os usuários foram colocados juntos, é como se não houvesse agrupamento algum.

Figura 41 - Dendrograma com distância do Cosseno



Fonte: Próprio autor.

Após serem finalizadas as experimentações no Orange foi escolhida uma das técnicas testadas para ser utilizada no desenvolvimento do motor de recomendação em Python.

A avaliação de técnicas de aprendizagem não supervisionada ocorre de forma mais subjetiva do que a supervisionada. Dabbura (2018) afirma que ao contrário do aprendizado supervisionado, onde existe um resultado conhecido (classe) para avaliar o desempenho do modelo, a análise de *clustering* não possui uma métrica de avaliação sólida para avaliar o resultado de diferentes algoritmos de *clustering*. O conhecimento de um ser humano sobre o domínio, aliados a intuição ajudam na validação.

Observando os gráficos e diagramas gerados com cada uma das técnicas que foram testadas é possível verificar que tanto o resultado da aplicação de *K-means* como o resultado da aplicação de *Hierarquical Clustering* (utilizando como medida a distância euclidiana) formaram agrupamentos interessantes nos dados e com as instâncias bem distribuídas perante os grupos formados. Qualquer um deles poderia ser escolhido para utilização na próxima fase de implementação. Contudo, como apenas um será aplicado, foi necessário escolher um deles e, devido a ter

obtido um tempo de execução um pouco menor nos testes realizados, foi selecionado o algoritmo *k-means*.

5.6 DESENVOLVIMENTO DO MOTOR DE RECOMENDAÇÃO

A partir da escolha resultante da experimentação com no Orange iniciou-se a implementação de um motor de recomendação. Durante esta etapa foi utilizado o ambiente computacional Jupyter Notebook, onde foi escrito um script em Python que faz uso dos dados provenientes do *dataset* de livros, juntamente com o algoritmo *k-means*, escolhido no decorrer do processo de KDD no Orange, para prever leituras interessantes e recomendá-las aos usuários.

As etapas que o motor de recomendação executa podem ser resumidas através dos seguintes passos:

a) Importação de bibliotecas: Foram necessárias para a implementação as bibliotecas “Sklearn” e “Pandas”. “Sklearn” contém o método de *clustering k-means,* e “Pandas” possui inúmeras funcionalidades para trabalhar com estruturas de dados tabulares, chamadas *dataframes*, e também métodos de análise de dados.

b) Criação de pandas *dataframes*: Com a função “read_csv” é possível importar os dados diretamente dos arquivos que estão no formato .csv. Ela também utiliza as colunas da primeira linha do arquivo (*header*) para nomear cada uma das colunas do *dataframe*. E a partir dessas estruturas criadas os dados já ficam disponíveis para serem acessados no restante do script.

Na figura 42 podem ser visualizados os trechos de código referentes a importação das bibliotecas e importação dos arquivos.

Figura 42 - Código inicial motor de recomendação

```
# Importa bibliotecas que serão itilizadas
from sklearn.cluster import KMeans
import pandas as pd

Ratings = pd.read_csv("ratings.csv")
Books = pd.read_csv("books.csv")
```

Fonte: Próprio autor.

c) Merge: Foi realizado um merge entre Books e Ratings para que, quando for

criada a matriz de usuários *versus* livros, se tenha disponível o título do livro para colocar nas colunas. O título era uma coluna disponível apenas em “Books”, e, como “Ratings” é a base para criação da matriz, foi necessário buscar essa informação em “Books”.

d) Pivot: Remodela os dados com base nos parâmetros de *index* (define o que vai compor as linhas), *columns* (define o que vai compor as colunas) e *values*, que são os valores que preencherão a intersecção de linhas com colunas. A aplicação deste comando foi o que possibilitou deixar os dados no formato desejado.

Observe a figura 43, que demonstra a realização do merge e criação da matriz de usuários *versus* livros, chamada de “user_book_ratings”. Nessa imagem também é possível visualizar como a matriz ficou, pois são apresentadas algumas colunas e registros da mesma.

Figura 43 - Merge e pivot table

```
#merge para depois enxergar cada coluna com o title ao invés do id do livro
ratings_title = pd.merge(Ratings, Books[['book', 'title']], on='title')
user_book_ratings = pd.pivot_table(ratings_title, index='user_id', columns='title', values='rating') #pandas puro
user_book_ratings
```

title	1984	Angels & Demons (Robert Langdon, #1)	Animal Farm	Catching Fire (The Hunger Games, #2)	Divergent (Divergent, #1)	Harry Potter and the Prisoner of Azkaban (Harry Potter, #3)	Harry Potter and the Sorcerer's Stone (Harry Potter, #1)	Pride and Prejudice	The Catcher in the Rye	The Diary of a Young Girl	The Fault in Our Stars	The Girl with the Dragon Tattoo (Millennium, #1)	The Great Gatsby	The Hobbit	The Hunger Games (The Hunger Games, #1)	The Kite Runner
user_id																
1	3.0	NaN	4.0	NaN	3.0	NaN	3.0	NaN	3.0	NaN	4.0	NaN	NaN	3.0	2.0	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN
3	NaN	2.0	NaN	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	3.0	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	4.0	NaN	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Fonte: Próprio autor.

e) Aplicar *k-means*: Cria três clusters e armazena o resultado na variável *predictions*. Ela é um *array* que contém o número do grupo que foi encontrado para cada usuário. A figura 44 contém o fragmento de código com a chamada do *k-means*.

Figura 44 - Inicialização K-means

```
:  # substitui nulls por zero
user_book_ratings = user_book_ratings.fillna(0)
#inicializa o k-means com os parâmetros desejados
kmeans = KMeans(n_clusters=3, init='k-means++')
# prediz em qual cluster cada usuário está inserido
predictions = kmeans.fit_predict(user_book_ratings)
```

Fonte: Próprio autor.

f) Criar classe: Cria uma coluna chamada “*cluster*” que contém o *cluster* referente a cada registro (usuário) do *dataframe*. Observe essa coluna destacada na figura 45.

Figura 45 - Criação do atributo classificador

```
# Cria uma coluna com o cluster correspondente a cada usuário
user_book_ratings['cluster'] = kmeans.labels_
user_book_ratings
```

	Catching Fire (The Hunger Games, #2)	Divergent (Divergent, #1)	Harry Potter and the Prisoner of Azkaban (Harry Potter, #3)	Harry Potter and the Sorcerer's Stone (Harry Potter, #1)	Pride and Prejudice	The Catcher in the Rye	The Diary of a Young Girl	The Fault in Our Stars	The Girl with the Dragon Tattoo (Millennium, #1)	The Great Gatsby	The Hobbit	The Hunger Games (The Hunger Games, #1)	The Kite Runner	To Kill a Mockingbird	Twilight (Twilight, #1)	cluster
0	0.0	3.0	0.0	3.0	0.0	3.0	0.0	4.0	0.0	0.0	3.0	2.0	0.0	0.0	0.0	1
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	0.0	3.0	4.0	2
0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	0
0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0
0	4.0	0.0	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0

Fonte: Próprio autor.

g) Cálculo da média de pontuação dos livros no cluster do usuário: Após a geração dos *clusters* é identificado em qual deles o usuário corrente (aquele que estiver logado no sistema) faz parte. São selecionados todos os usuários deste cluster e calculada a média das *ratings* deles apenas para os livros que o usuário corrente ainda não leu, ou seja, não avaliou ainda. Na figura 46 pode ser verificado com maior detalhe essa parte da implementação.

Figura 46 - Média pontuação livros no cluster do usuário corrente

```
# descobre o número do cluster do usuário atual (teste fixo com usuário 2)
cluster_number = user_book_ratings.loc[2, 'cluster']

#filtra o dataframe para manter apenas quem é do mesmo cluster q ele
cluster = user_book_ratings[user_book_ratings.cluster == cluster_number].drop(['cluster'], axis=1)

# Retorna todas as ratings do usuário atual
user_ratings = pd.DataFrame(user_book_ratings.loc[2])
user_ratings.columns = ['rating']

# Encontra os livros que o usuário não leu (não pontou)
user_unrated_books = user_ratings[user_ratings['rating'] == 0]

# Média das ratings dos membros desse cluster para os livros que o usuário ainda não leu
avg_ratings = pd.DataFrame(pd.concat([user_unrated_books, cluster.mean()], axis=1, join='inner').loc[:, 0])
avg_ratings.columns = ['mean_rate']
```

Fonte: Próprio autor.

h) Apresenta lista de recomendações: Após calculada a média, é realizada uma ordenação de maneira decrescente, de modo que os resultados mais relevantes (livros com maior média) fiquem no topo do resultado. E depois de realizada a ordenação foi passado um número limite de recomendações e realizado um laço de repetição para apresentar como resultado o título dos livros recomendados ao usuário. Veja na figura 47 essa parte da implementação.

Figura 47 - Recomendações

```
# Ordena pela pontuação de maneira decrescente (as maiores ratings ficam por cima)
avg_ratings = avg_ratings.sort_values(by='mean_rate', ascending=False)[:3]

# retorna só os títulos dos livros - cria um novo dataframe com os indexes
recomendations = pd.DataFrame(avg_ratings.index)
for index, row in recomendations.iterrows():
    print(row['title'])
```

```
Harry Potter and the Prisoner of Azkaban (Harry Potter, #3)
The Fault in Our Stars
Twilight (Twilight, #1)
```

Fonte: Próprio autor.

Com a funcionalidade principal implementada através desse motor de recomendação foi possível prosseguir para a próxima etapa deste trabalho, que é disponibilizar tal funcionalidade por meio de uma interface gráfica para que os usuários tenham acesso a ela não apenas por linha de comando, mas também de maneira mais visual. No próximo capítulo será tratada a implementação de uma aplicação web utilizando o Django *Framework* para atender a essa necessidade.

5.7 DESENVOLVIMENTO DA APLICAÇÃO WEB EM *DJANGO*

Para iniciar a implementação de uma aplicação *web,* que se trata do protótipo de um sistema de recomendação, é necessário conhecer os componentes que são parte desse sistema, bem como o fluxo da informação dentro dele e as tecnologias utilizadas na sua produção.

5.7.1 Componentes do Recomendador e tecnologias utilizadas

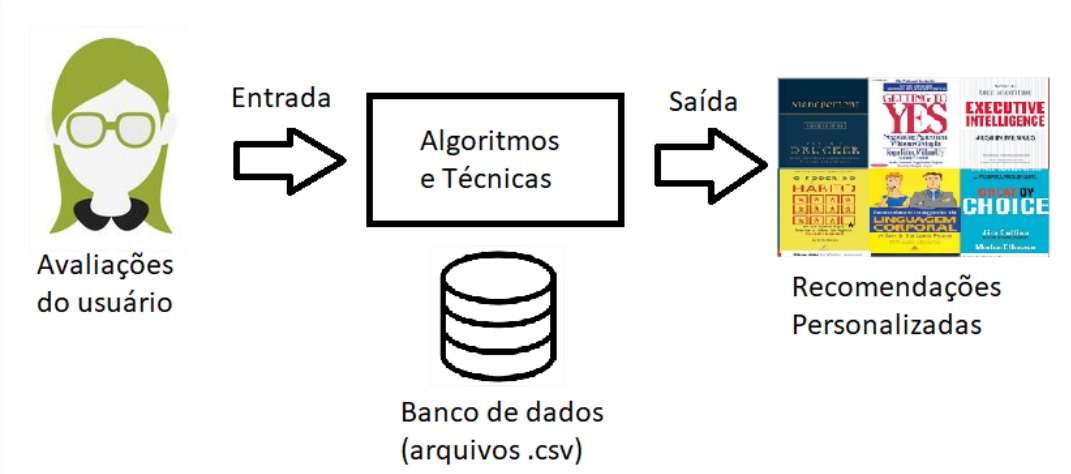
Segundo Phorasim e Yu (2016), os componentes dos sistemas de recomendação compõem quatro partes. São elas: o banco de dados, a interface

humano computador, o algoritmo e os componentes de recomendação. Na figura 48 estão representados tais componentes e a forma como eles se relacionam.

As avaliações do usuário consistem na pontuação (de 1 a 5) que serão preenchidas pelos utilizadores do sistema para os livros que elas já leram. O banco de dados padrão do Django é o sqlite3, e foi esse o utilizado nesta implementação.

Foram importados para esse banco os arquivos .csv provenientes do *dataset* escolhido, já com as informações completas e com os tipos de dados corrigidos, conforme etapas de seleção, pré-processamento e transformação provenientes do processo de KDD.

Figura 48 - Componentes do Recomendador de Livros



Fonte: Próprio autor.

O recomendador de livros foi desenvolvido em Python. Essa linguagem de programação foi escolhida, pois vem sendo amplamente utilizada em soluções que necessitam trabalhar com análise de dados. Mckinney (2018, p. 20) afirma que:

> Nos últimos dez anos, Python passou de uma linguagem de computação científica inovadora, ou para ser usada por 'sua própria conta e risco', para uma das linguagens mais importantes em ciência de dados, aprendizado de máquina (machine learning) e desenvolvimento de software em geral, no ambiente acadêmico e no mercado.

Segundo Mckinney (2018), existem bibliotecas específicas para se trabalhar com Python para análise de dados. Dentre elas podem ser citadas:

- A NumPy, que "oferece suporte a maioria das aplicações científicas que envolvem dados numéricos em Python". (MCKINNEY, 2018, p. 22)
- Pandas, que oferecem estruturas de dados e ferramentas para facilitar o trabalho envolvendo dados estruturados ou tabulares;

- Matplotlib, que possui funcionalidades que envolvem a visualização dos dados.
- SciPy, que é uma coleção de pacotes que possibilitam lidar com problemas comuns no processamento de dados. Dentre esses pacotes está o scipy.sparse que contém soluções para “matrizes esparsas e sistemas lineares esparsos” (MCKINNEY, 2018, p. 26);
- Scikit-learn, que é um kit de ferramentas para trabalhar com aprendizado de máquina. Ele possui capacidade de executar tarefas de agrupamento (*clustering*) e fatoração de matrizes, por exemplo.

Para a implementação da interface gráfica foi utilizado o *framework* de desenvolvimento Django. A IDE de desenvolvimento escolhida foi o PyCharm, mas existem inúmeras que poderiam ter sido escolhidas, como por exemplo, o Visual Studio Code, que atendem ao mesmo propósito.

5.7.2 Implementação

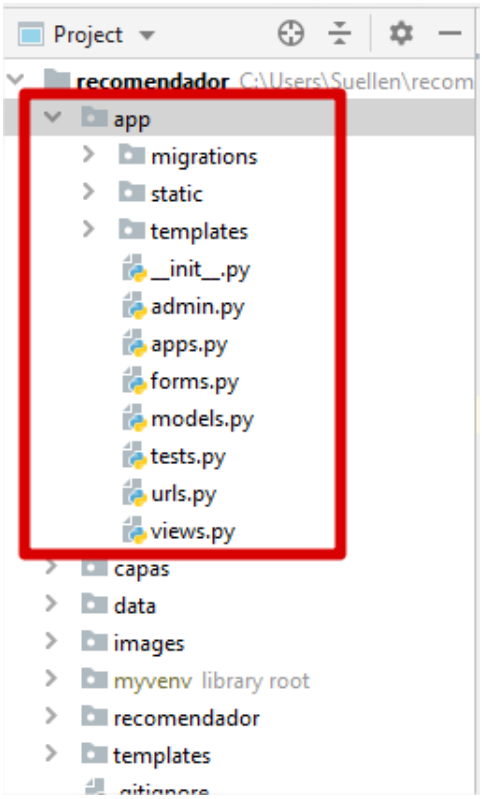
A primeira etapa para começar o desenvolvimento foi criar um *virtual environment* (ambiente virtual) para poder configurar as versões que se deseja usar do Python e do Django. A vantagem em trabalhar com ambiente virtual é que ele permite escolher qual versão do Python se deseja trabalhar, pois é comum ter mais de uma versão instalada no computador.

Por exemplo: para usar o Orange e o Jupyter Notebook (que vem com o Anaconda) podem ser necessárias versões específicas do Python que sejam compatíveis com essas aplicações, mas no momento do desenvolvimento pode-se querer trabalhar com outra versão (geralmente uma das mais recentes, que vem sendo aprimoradas e disponibilizando cada vez mais funcionalidades). Neste trabalho foi utilizado o Python 3.6.1 e Django 2.2.6.

A seguir foi criado um projeto do Django, com o comando “django-admin.exe startproject recomendador .“ (onde “recomendador” é o nome escolhido para o projeto), que cria a estrutura de pastas necessárias para trabalhar com esse *framework*. Todos esses comandos são executados pelo prompt de comando do computador, dentro do ambiente virtual criado no inicialmente.

O próximo passo foi a criação de uma aplicação dentro do projeto, com o objetivo de manter o código organizado. O comando para isso é “python manage.py startapp app”, sendo que “app” foi o nome escolhido para a aplicação. Após a criação da aplicação, a estrutura de pastas fica parecida com a figura 49, contendo arquivos como o “models.py”, “urls.py” e “views.py” que serão detalhados a seguir.

Figura 49 - Estrutura de pastas da aplicação



Fonte: Próprio autor.

Os modelos (*models*) representam a estrutura dos objetos que se quer definir. No Django, servem tanto como definição para criar e manter as tabelas do banco de dados, como também para utilizar os atributos criados na própria aplicação. Em “models.py” foram definidos dois modelos chamados de “Book” e “Rating”. Através da figura 50 podem ser observados também os atributos criados para cada um deles.

Figura 50 - Definição dos modelos

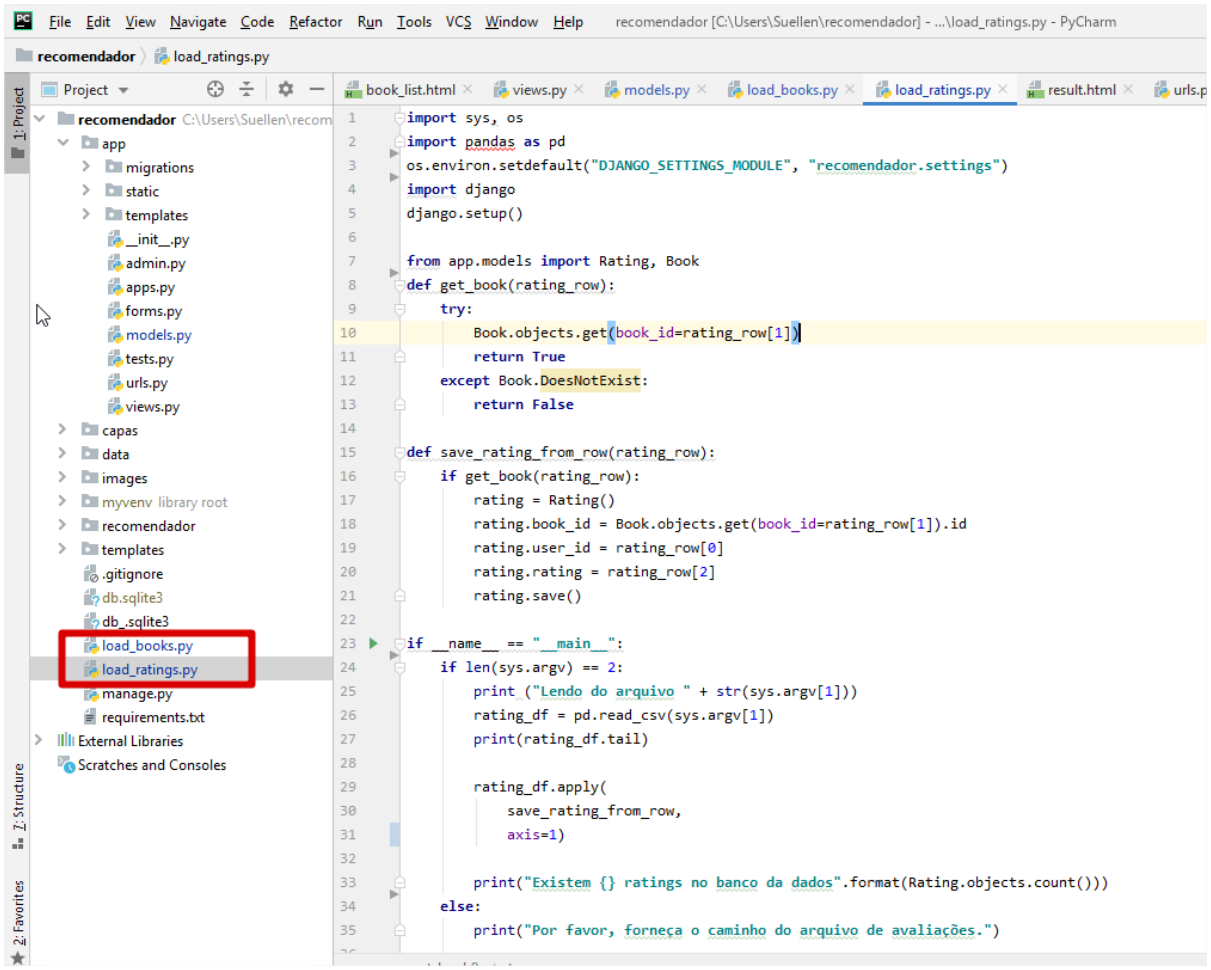
```
from django.conf import settings
from django.db import models
from django_pandas.managers import DataFrameManager

class Book(models.Model):
    book = models.IntegerField()
    authors = models.CharField (max_length=500)
    year = models.IntegerField()
    title = models.CharField(max_length=500)
    average_rating = models.FloatField()
    ratings_count = models.FloatField()
    image_url = models.URLField()

    objects = DataFrameManager()

    def __str__(self):
        return self.title

class Rating(models.Model):
    user_id = models.IntegerField()
    book = models.ForeignKey(Book, on_delete=models.CASCADE)
    rating = models.IntegerField()

    objects = DataFrameManager()
```

Fonte: Próprio autor

Após a definição dos modelos foi possível criar as tabelas referentes a eles no banco da dados. Para a criação utilizando o banco padrão do Django, que é sqlite3, basta executar o comando “python manage.py makemigrations app”, que cria um arquivo de migração. Depois é necessário aplicar esse arquivo criado através da execução de “python manage.py migrate app”.

Com as tabelas criadas falta apenas inserir informações nesse banco de dados. Para importar os arquivos .csv para dentro da estrutura de tabelas criadas foram escritos scripts que varrem os arquivos linha a linha e vão os inserindo os dados nas tabelas. Os nomes dos scripts são: “load_books.py” e “load_ratings.py”. A figura 51 destaca a localização deles na árvore do projeto e também contém o código fonte do script de importação de ratings.

Figura 51 - Scripts de importação de dados



Fonte: Próprio autor

A seguir foi realizada a configuração das urls e criação das *views*. As URLs no Django são o que direciona para as “views.py”. Por exemplo: no recomendador de livros, quando o usuário acessar url da página principal, é preciso que o sistema encontre a *view* referente a essa página.

Nas *views* é onde ocorre a comunicação entre o que vem da tela (mais adiante será visto a parte de implementação dos *templates* em html) e a lógica da aplicação. Essa “conversa” ocorre em dois sentidos: Tanto quando o usuário requisita algo para a aplicação, como, por exemplo, buscar recomendações; como quando a aplicação precisa enviar algo para a tela, como, por exemplo, retornar as recomendações ou então apresentar uma lista inicial de livros a serem avaliados.

As regras de negócio do sistema ficam implementadas nas “views.py”, portanto o motor de recomendação construído na etapa anterior de desenvolvimento deste trabalho foi colocado dentro de uma *view* desta aplicação construída em

Django. Foram implementadas as seguintes *views*:

a) “book_list”, que retorna uma lista de livros a serem avaliados pelos usuários. Essa lista é o que vai ser apresentado na tela inicial do recomendador.

b) “rating_new”, que é responsável por pegar a pontuação atribuída pelo usuário para determinado livro e devolver para a aplicação gravar no banco de dados

c) “result”, que é acionada quando o usuário clica no botão “buscar recomendações”. Nela há a lógica do motor de recomendação e o retorno da lista de livros recomendados para a tela. As figuras 52 e 53 contém o prints do código fonte da *view* “result”. A lógica é muito similar a aquela vista anteriormente no motor de recomendação, com algumas alterações apenas em relação a recuperação dos dados, que agora estão no banco de dados e não mais em arquivos .csv.

Figura 52 - View “result”, parte 1

```
@login_required
def result(request):
    #dados do usuário logado
    curr_user = request.user
    curr_user_id = curr_user.id
    curr_user_id = int(curr_user_id)

    #cria querysets de todos os registros das tabelas book e rating
    qs_Books = Book.objects.all()
    qs_Ratings1 = Rating.objects.values('user_id','book','rating').order_by('user_id')
    qs_Ratings2 = Rating.objects.values('user_id', 'book', 'rating').order_by('-rating')
    qs_Ratings_usu_atual = Rating.objects.values('user_id', 'book', 'rating').filter(user_id=curr_user_id)

    #cria pandas dataframes baseados nos querysets
    Books = qs_Books.to_dataframe()
    Ratings1 = qs_Ratings1.to_dataframe()
    Ratings2 = qs_Ratings2.to_dataframe()
    Ratings_usu_atual = qs_Ratings_usu_atual.to_dataframe()

    #junta os ratings gerais com os ratings do user_id -> concatena os dataframes
    frames = [Ratings1, Ratings2, Ratings_usu_atual]
    Ratings = pd.concat(frames)

    #transforma os dataframes para formato desejado (matriz users X books)
    Ratings.columns = ['user_id','title','rating'] #renomeia as colunas de ratings (book vira title)
    ratings_title = pd.merge(Ratings, Books[['book', 'title']], on='title')
    user_book_ratings = pd.pivot_table(ratings_title, index='user_id', columns='title', values='rating') #pandas puro

    # substitui nulls por zero
    user_book_ratings = user_book_ratings.fillna(0)
    kmeans = KMeans(n_clusters=3, init='k-means++')
    kmeans.fit(user_book_ratings);
```

Fonte: Próprio autor

Figura 53 - View “result”, parte 2

```
# Cria uma coluna com o cluster correspondente a cada usuário
user_book_ratings['cluster'] = kmeans.labels_

# descobre o número do cluster do usuário atual e filtra o dataframe para manter apenas quem é do mesmo cluster q ele
cluster_number = user_book_ratings.loc[curr_user_id, 'cluster']
cluster = user_book_ratings[user_book_ratings.cluster == cluster_number].drop(['cluster'], axis=1)

# Retorna todas as ratings do usuário atual
user_ratings = pd.DataFrame(user_book_ratings.loc[curr_user_id])
user_ratings.columns = ['rating']

# Quais livros o usuário não leu (não pontou)?
user_unrated_books = user_ratings[user_ratings['rating'] == 0]

# Média das ratings dos membros desse cluster para os livros que o usuário ainda não leu
avg_ratings = pd.DataFrame(pd.concat([user_unrated_books, cluster.mean()], axis=1, join='inner').loc[:, 0])
avg_ratings.columns = ['mean_rate']

# Ordena pela pontuação de maneira decrescente (as maiores ratings ficam por cima)
avg_ratings = avg_ratings.sort_values(by='mean_rate', ascending=False)[:12]

# retorna só os títulos dos livros - cria um novo dataframe com os indexes
recomendations = avg_ratings.index
recomendations = recomendations.values.tolist()
books_result = []

#vai incrementando a lista de resultados (book_result)1
for title in recomendations:
    books_result.append(Book.objects.filter(title=title).first())

#retorna resultado para a template de result
return render(request, 'app/result.html', {'recomendations': books_result})
```

Fonte: Próprio autor

Para criar aquilo que o usuário vê e disponibilizar o recomendador para ser utilizado, foram escritos arquivos html com o uso de templates do Django. Para definição de estilos para as páginas foi utilizado CSS e também o Bootstrap.

Os *templates* do Django permitem acessar atributos dos modelos criados de dentro da página html. Na figura 54 há um exemplo de uso de template para retornar as variáveis do *model* “Book” para a tela. O código destacado em vermelho busca informações como o título, autor e ano para apresentar na página *web*.

Figura 54 - Trecho de html usando Django template

```
{% for book in books %}
    <div class="book col-xs-12 col-md-3">
        <img src='{{ book.image_url }}' class='img-fluid' width="200" height="150"/>
        <h4>{{ book.title }}</h4>
        <p>{{ book.authors }}</p>
        <p>{{ book.year }}</p>
        <p>{{ book.text|linebreaksbr }}</p>
        <form action="." method="POST">
            {% csrf_token %}
            <input type="text" value="{{ book.id }}" id="{{ form.book.id_for_label }}_{{ book.id }}" name="{{ form.book.name }}">
            {{ form.rating }}
        </form>
    </div>
{% endfor %}
```

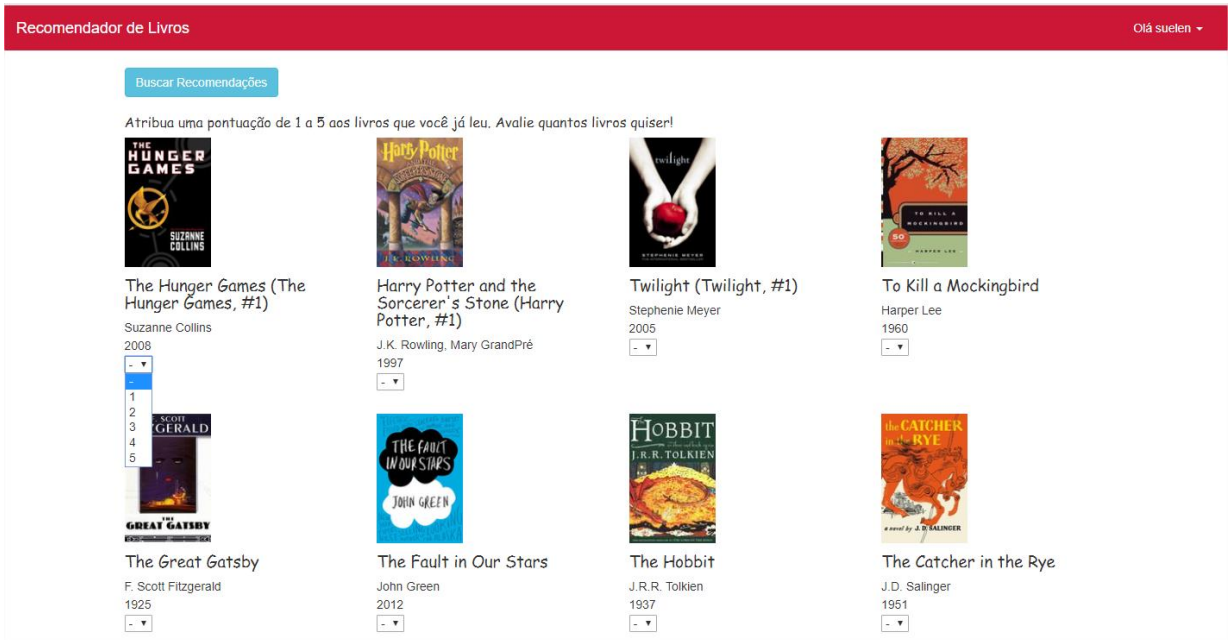
Fonte: Próprio autor.

Um artifício importante no desenvolvimento do recomendador foi o uso da ferramenta de administração do Django, que foi útil em diversos momentos. Naquela etapa de importação dos arquivos .csv para o banco de dados foram necessárias diversas tentativas até a importação completa acontecer com sucesso. A ferramenta de administração permitiu verificar de forma prática quais os dados que haviam inseridos e a quantidade de registros em cada tabela. Ela foi utilizada também para a criação dos usuários para as pessoas que fizeram a avaliação do recomendador.

Houveram mais pormenores que fizeram parte de implementação, como, por exemplo, autenticação de usuários e a disponibilização das imagens com as capas dos livros na interface. Porém, como essas partes não são tão relevantes quanto ao objetivo principal do presente trabalho, eles não serão detalhados da mesma forma como foi feito com os componentes apresentados até o momento.

Mediante todas as etapas de desenvolvimento concluídas, resta apresentar o resultado disto. Após o usuário (criado pela ferramenta de administração) ter logado no sistema, é apresentado a ele a página inicial do recomendador de livros, que pode ser visualizada na figura 55. Nesta tela cada livro possui uma lista com opções de 1 a 5 para que seja possível realizar a sua avaliação. É necessário que o usuário avalie alguns livros que ele tenha lido para que o sistema tenha informações para passar ao motor de recomendação gerar uma lista de sugestões de leitura.

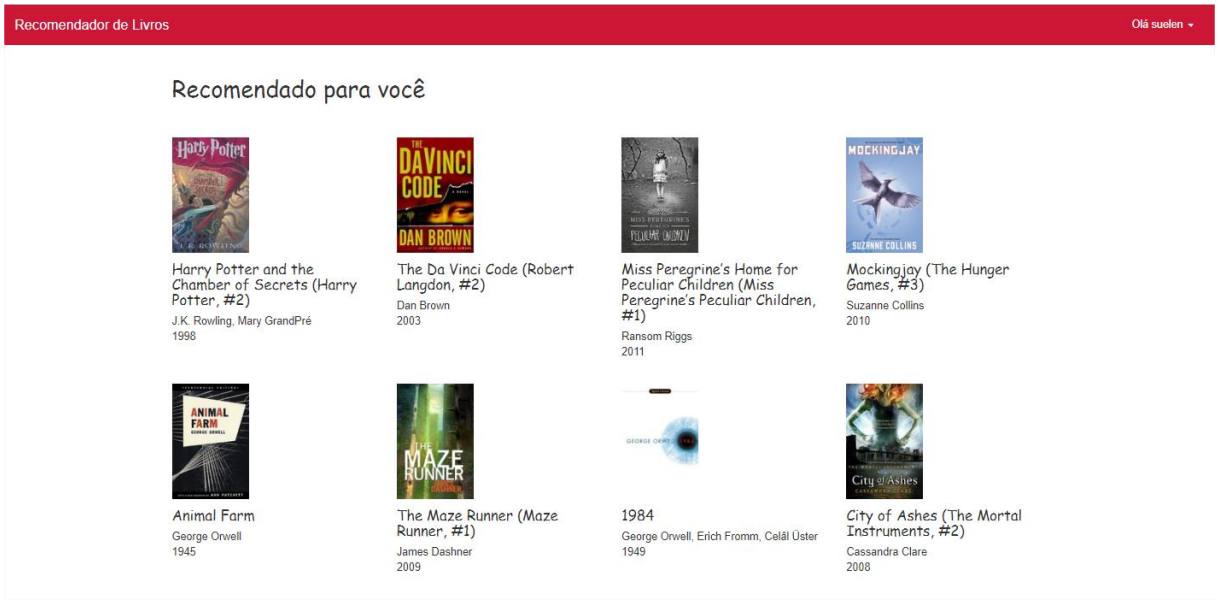
Figura 55 - Interface inicial do recomendador de livros



Fonte: Próprio autor.

Após finalizar a avaliação dos livros podem ser obtidas as recomendações através do botão “Buscar Recomendações”. Quando esse botão é acionado o recomendador redireciona para a tela de resultado das recomendações, que pode ser visualizada na figura 56.

Figura 56 - Interface de recomendações



Fonte: Próprio autor

5.8 AVALIAÇÃO DO RECOMENDADOR

Para identificar se as recomendações geradas pelo protótipo construído são relevantes aos usuários foi realizada uma avaliação baseada em um questionário respondido por 10 pessoas.

5.8.1 O questionário

O questionário foi aplicado presencialmente, pois ele é respondido após os usuários terem utilizado o recomendador de livros. Quando o usuário utilizou o recomendador de livros, foi solicitado que avaliasse pelo menos 5 livros, mas se quisesse avaliar mais era permitido. Após as avaliações foi utilizada a funcionalidade de buscar recomendações. O resultado da busca retornou 10 sugestões de leitura. E com base nelas foram realizadas as seguintes perguntas:

1) Você já leu algum dos títulos recomendados? (Sim ou Não)

Essa pergunta se deve ao fato de que geralmente os usuários não avaliam todos os livros que leram na vida então, consequentemente, os resultados da recomendação acabam sugerindo obras que já foram lidas pelas pessoas. Para o propósito de avaliar SR isso não é de todo ruim, pois permite que se obtenha a pontuação real que aquela pessoa atribuiria àquele livro que já foi lido.

2) Se sim, cite os títulos dos livros e quais pontuações você atribui a eles.

3) Sobre os livros que foram sugeridos e você ainda não leu. Pontue de 1 a 5 de acordo com a probabilidade de que você os leria algum dia. (sinta-se à vontade para pesquisar mais sobre o livro ou autor para responder a essa pergunta verdadeiramente).

1) Quantos livros há na sua lista de livros para os próximos 6 meses?

5) Cite um gênero, ou autor, ou tipo de livro que você gosta.

6) Cite 2 hobbies seus.

As perguntas 5 e 6 buscam compreender se há diversidade no perfil dos respondentes do questionário.

Baseado nessas seis perguntas foi preenchida uma tabela para cada um dos respondentes. Uma das respostas pode ser observada pela figura 57. As demais respostas estão disponíveis em apêndices.

Figura 57 - Resposta do questionário avaliativo

	Cleber		
1)	Você já leu algum dos títulos recomendados? Não		
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5		
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra		
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 2		
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.		
	Gêneros: terror, ficção científica, política e filosofia. Autores: Stephen King, Ayn Rand, R. R. Tolkien		
6)	Cite 2 hobbies seus: Violão, videogame, leitura		
	Livro	Já havia lido esse livro?	Pontuação/Interesse (1 a 5)
	The Hobbit	não	5
	The Catcher in the Rye	não	3
	Slaughterhouse-Five 4	não	2
	Angels & Demons (Robert Langdon, #1)	não	4
	The Lord of the Rings (The Lord of the Rings, #1-3)	não	5
	Foundation and Empire (Foundation #2)	não	4
	Pride and Prejudice	não	2
	Foundation (Foundation #1)	não	4
	Hamlet	não	3
	Harry Potter and the Half-Blood Prince (Harry Potter, #6)	não	2

Fonte: Próprio autor

5.8.2 A avaliação

Em “Trabalhos Relacionados” são citadas algumas medidas utilizadas na avaliação de sistemas de recomendação. Dentre aquelas estudadas foram selecionadas a MAE (*Mean Absolute Error*), que é utilizada por Cerqueira e Coello (2013), e a RMSE (*Root Mean Squared Error*), que é uma das medidas utilizadas por Bokde, Girase e Mukhopadhyay (2015). A principal diferença entre as duas é que RMSE pune mais os erros, pois eleva as diferenças entre o previsto e o real ao quadrado. Sendo assim, RMSE pode ser considerada uma medida mais rigorosa do que MAE.

Para cada um dos respondentes da pesquisa foi calculado o MAE e o RMSE utilizando a própria planilha eletrônica, onde haviam sido anotadas as respostas das 6 perguntas do questionário. Na figura 58 há uma imagem dos cálculos realizados para um dos respondentes; nela pode ser observado que o resultado de MAE foi 1,6 e RMSE foi de 1,9 para esse usuário. Os cálculos detalhados para cada uma das

pessoas que participou da avaliação estão nos apêndices deste trabalho.

Figura 58 - Cálculos realizados para avaliação do recomendador

Livro	Já havia lido esse livro?	Pontuação/Interesse (1 a 5)	Pontuação atribuída pelo recomendador	Diferença das Pontuações	Diferenças ao quadrado
The Hobbit	não	5	5	0	0
The Catcher in the Rye	não	3	5	2	4
Slaughterhouse-Five 4	não	2	5	3	9
Angels & Demons (Robert Langdon, #1)	não	4	5	1	1
The Lord of the Rings (The Lord of the Rings, #1-3)	não	5	5	0	0
Foundation and Empire (Foundation #2)	não	4	5	1	1
Pride and Prejudice	não	2	5	3	9
Foundation (Foundation #1)	não	4	5	1	1
Hamlet	não	3	5	2	4
Harry Potter and the Half-Blood Prince (Harry Potter, #6)	não	2	5	3	9
			MAE	**1,6**	
				RMSE	**1,9**

Fonte: Próprio autor

A apuração dos resultados das medidas calculadas se encontra na tabela 8. Lembrando que quanto menor o resultado das medidas de erro, melhor o desempenho do recomendador. Repare que no melhor resultado foi obtido MAE igual a 0,6 e RMSE igual a 0,9; enquanto que no pior resultado o MAE foi de 2 e o RMSE foi de 2,7. Ao final da tabela 8 foi calculada a média do MAE e RMSE.

Se fossem extraídos aleatoriamente livros de um banco de dados para sugerir leituras a uma pessoa, considera-se que a chance de que cada livro fosse de interesse daquela pessoa é de 50%. O recomendador de livros precisa, no mínimo, superar o acaso, ou seja, acertar pelo menos mais da metade das recomendações, senão ele não é efetivo.

Isso é mensurado da seguinte forma: numa escala de pontuação de 1 a 5, o número que representa 50% da pontuação máxima é 2,5. Então, para que o recomendador seja mais efetivo do que escolher um livro qualquer, é preciso que a sua taxa de acertos seja maior que 2,5.

O resultado das **médias** de MAE e RMSE é apresentado na tabela 8: são eles, 1,5 e 1,9, respectivamente. Como esses valores representam os erros médios, para encontrar a taxa de acertos é necessário diminuí-los da pontuação máxima, que é 5. Sendo assim, as taxas de acerto são: **3,5** (resultado de 5 diminuído de 1,5) e **3,1** (resultado de 5 diminuído de 1,9).

Como ambas as taxas são maiores que 2,5, o protótipo de sistema de recomendação construído pode ser considerado efetivo em sua tarefa de

recomendar livros.

Tabela 8 – Resultados dos cálculos de erro

	MAE	RMSE
Cleber	1,6	1,9
Aline	2	2,7
Morbini	1,8	2,0
Gamba	1,6	2,1
Samuel	1,4	1,8
Cassiano	1,7	2,0
Rodrigo	2,2	2,4
Bandiera	0,6	0,9
Nelci	1	1,4
Carlos	1,5	1,8
MÉDIA	**1,5**	**1,9**

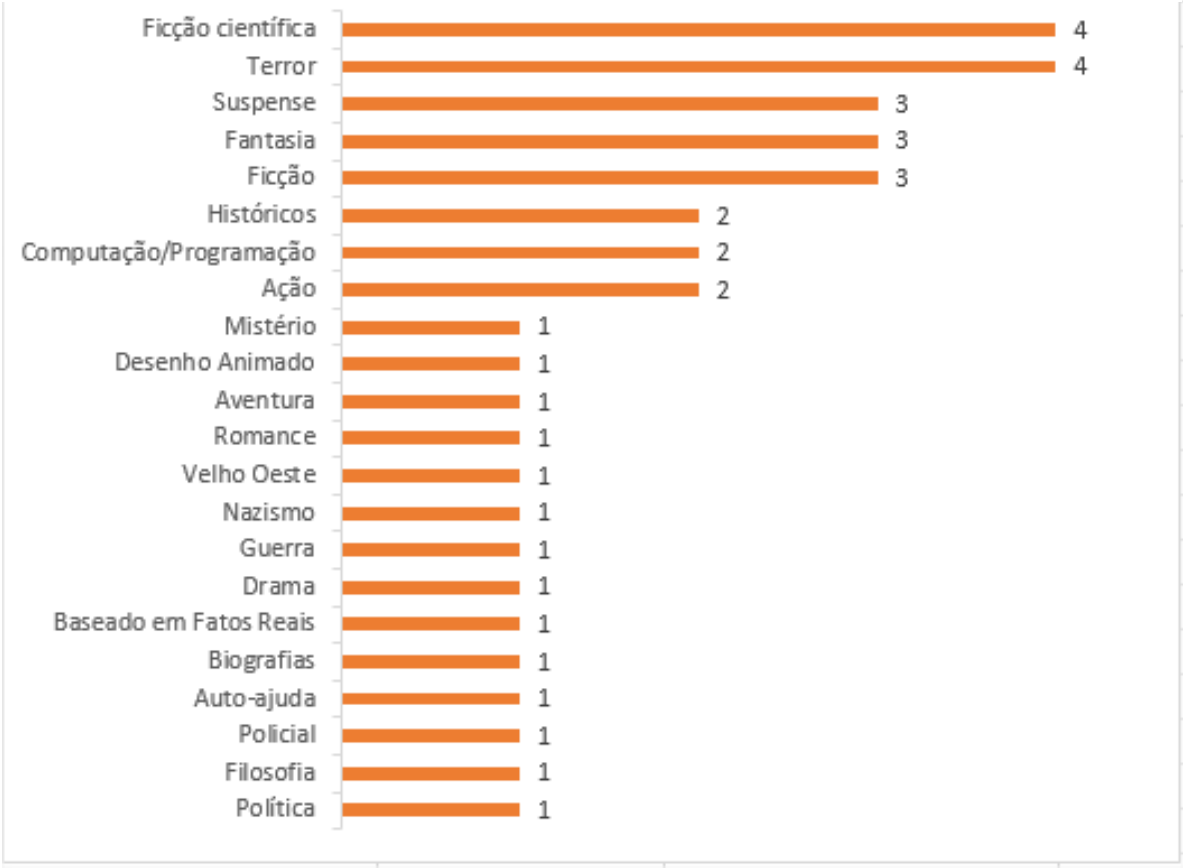
Fonte: Próprio autor

Outro aspecto analisado foi o perfil dos respondentes da pesquisa para identificar se ela foi aplicada em um público diversificado. As perguntas de número cinco e seis da pesquisa são em relação a esse aspecto.

Baseado nas respostas da pergunta cinco foi elaborado o gráfico constante na figura 59. Ele representa a quantidade de vezes que um termo foi usado como resposta por algum usuário. O termo “Terror”, por exemplo, foi citado por 4 pessoas, enquanto que “Velho Oeste” aparece apenas uma vez.

Veja que mais da metade dos termos utilizados foi citado apenas uma vez. Isso é um indicador de que as pessoas entrevistadas possuem interesses de certa forma diversificados. Porém, há alguma convergência de gostos similares quanto a “Ficção científica”, “Terror”, “Suspense”, “Fantasia” e “Ficção”, pois eles foram citados por, pelo menos, três leitores.

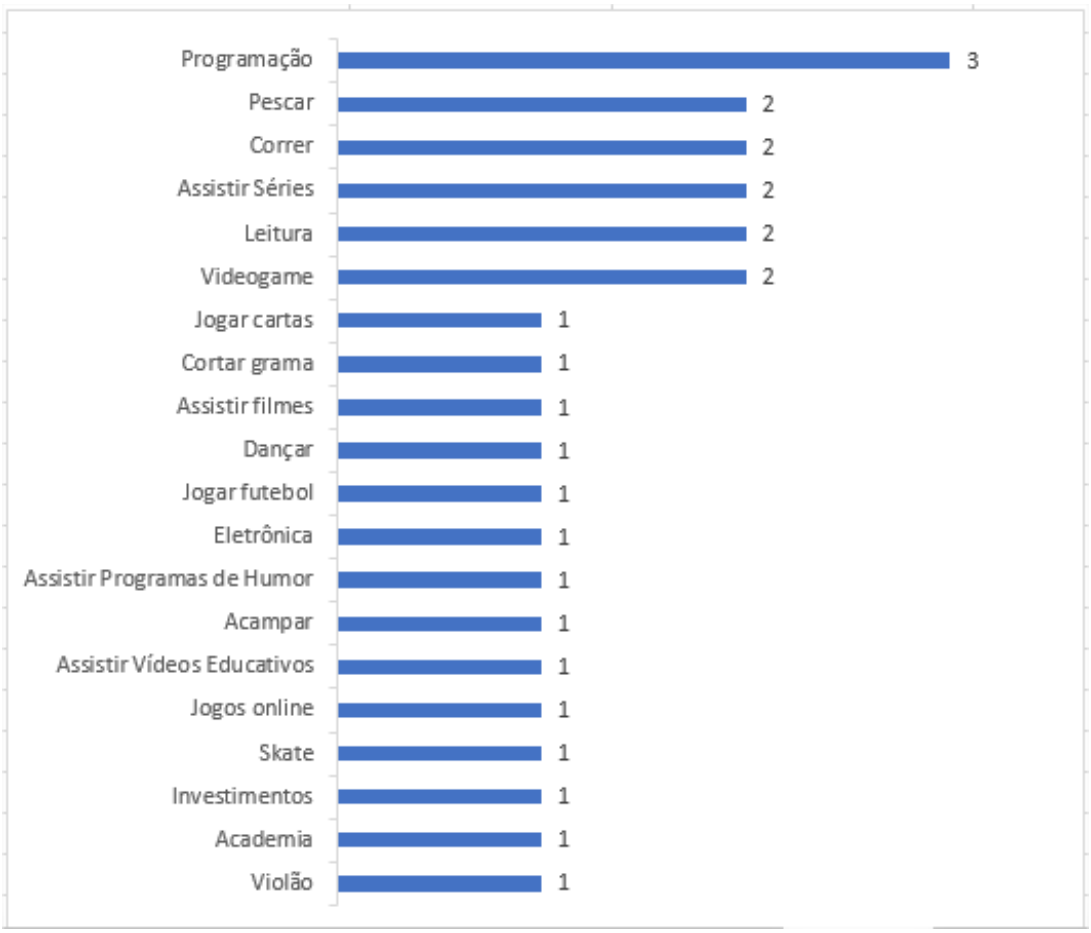
Figura 59 - Gráfico de gêneros de interesse dos leitores



Fonte: Próprio autor

Considerando as respostas da pergunta seis foi elaborado o gráfico apresentado na figura 60 que demonstra haver diversidade nos hobbies dos entrevistados. Os hobbies dizem um pouco mais sobre o perfil dos usuários e pelas respostas pode ser observado que eles variam desde atividades físicas, como “Corrida”, até atividades intelectuais, como “Programação” e “Leitura”, sendo que a maioria dos termos aparece apenas uma vez.

Figura 60 - Gráfico de hobbies dos leitores



Fonte: próprio autor

6 CONCLUSÃO

Nos dias atuais a internet possui um oceano de informações sobre praticamente qualquer assunto que se pense em pesquisar. Não seria diferente no campo das publicações literárias, onde surgem novos autores e obras a cada momento, e, com a facilidade que a tecnologia proporciona, esse conteúdo é disponibilizado rapidamente junto com o restante dessa vastidão de informações.

Nesse contexto de abundância de opções é que surge a necessidade de uso de sistemas de recomendação. Ao realizar buscas na internet com termos como “leituras para 2020”, serão retornados inúmeros resultados. Na verdade, haverá informação até demais nesse resultado. As pessoas tem cada vez menos tempo de navegar perante uma imensidão de informações até encontrar aquilo que seja de seu real interesse.

Os sistemas de recomendação de livros buscam minimizar esse esforço por parte do usuário para encontrar leituras, pois já sabem daquilo que o usuário vai gostar de ler antes mesmo dele saber disso.

Uma outra forma de encontrar recomendações personalizadas é perguntando para uma pessoa com gosto literário parecido com o seu. Porém, nem todos conhecem essa pessoa que tem interesses tão similares com os seus. Para resolver isso existe a filtragem colaborativa, que é uma técnica utilizada em SR para simular essa recomendação “boca a boca”. Ela funciona encontrando usuários com gostos similares aos seus e recomendando os livros que ele já leu e você ainda não (e vice-versa).

Para que os SR sejam efetivos é necessário que haja abundância de informação para ser analisada. Isso se deve ao fato de que a maioria deles trabalha aplicando algoritmos que possuem bons resultados quando existe uma quantidade elevada de dados (*big data*) para que eles identifiquem padrões nos mesmos.

O processo que busca encontrar conhecimento a partir da abundância de dados se chama KDD. Ele foi aplicado no decorrer deste trabalho para identificar técnicas capazes de identificar grupos de usuários similares em um determinado conjunto de dados.

Com a execução desse processo chegou-se ao algoritmo *K-means,* que descobre grupos (*clusters*) nos dados sobre os quais ele é aplicado. Ele foi utilizado no desenvolvimento de um motor de recomendação que se baseia no princípio da

filtragem colaborativa para sugerir leituras.

Foi implementada também uma aplicação web que disponibiliza as pessoas esses conceitos estudados através da funcionalidade de realizar recomendações literárias personalizadas.

Durante o desenvolvimento deste trabalho foi possível obter tanto conhecimento teórico dos conceitos necessários, como também colocá-los em prática. Uma das maiores dificuldades da fase prática foi conseguir montar a tabela de dados no formato desejado. Houve uma tentativa de fazer essa modelagem pelo uso de técnicas comuns de programação, mas durante o desenvolvimento descobriu-se que havia uma função pronta na biblioteca pandas que fazia exatamente o que se estava tentando implementar.

Depois de resolvido esse impedimento foi possível prosseguir com a execução, sendo que a implementação do motor de recomendação foi a parte mais interessante de fazer. Foi necessário estudar Python e, principalmente, as bibliotecas usadas em análise de dados para conseguir fazer essa etapa. Inicialmente o motor de recomendação teve alguns problemas de performance que precisaram ser resolvidos, mas no geral foi prazeroso realizar testes até deixá-lo pronto para ser usado.

A parte mais difícil do desenvolvimento foi a implementação em Django. Principalmente pois desenvolvimento *web* é um tópico desconhecido por parte de quem lhes escreve. E para conseguir tirar esse protótipo do papel foi preciso estudar Django (foi feito o tutorial “Django Girls”) e sofrer um pouco até conseguir um resultado apresentável.

No geral, estudar, modelar e implementar um protótipo de sistema de recomendação foi positivo e o sentimento é de dever cumprido. Existem alguns aspectos que poderiam ser ainda mais explorados, principalmente no sentido de agregar mais funcionalidades no recomendador, como a pesquisa de livros na tela inicial, e também disponibilizar resenhas das obras para os leitores.

O texto deste trabalho pode ser utilizado por pessoas que desejam conhecer mais sobre *Big Data*, Sistemas de Recomendação e os tipos existentes, Processo de Descoberta de Conhecimento, Mineração de Dados utilizando o software Orange, algoritmos de *Clustering e* implementação de Sistemas de Recomendação usando Python e Django.

Para avaliação do protótipo implementado foi aplicado um questionário com

dez pessoas que utilizaram o recomendador de livros. Os resultados indicam que o perfil das pessoas entrevistadas é diversificado e as taxas médias de acertos das recomendações, que são **3,5** (pelo cálculo de erro MAE) e **3,1** (pelo cálculo de erro RMSE), demonstram que as leituras recomendadas foram efetivas.

Inclusive, a maioria das pessoas que participou dos testes pediu que lhes fosse enviada a lista de leituras recomendadas, pois eles gostariam de adicionar as suas listas de leituras, ou até mesmo assistir aos filmes correspondentes a alguma obra que foi adaptada para o cinema.

6.1.1 Trabalhos Futuros

Tudo na vida pode, e deve, ser aprimorado. Não seria diferente com este trabalho de conclusão de curso.

Há uma funcionalidade que havia sido prevista na modelagem do recomendador que não foi implementada por falta de tempo. Se trata da busca de livros por nome do livro ou nome do autor.

Um trabalho que pode ser feito continuamente é a pesquisa sobre novidades nessa área de soluções de recomendação, pois há uma vastidão de técnicas que são utilizadas e em um trabalho só não é possível estudar ou testar todas elas. Vale encontrar algumas opções com as quais o pesquisador se identifique para conhece-las de maneira mais aprofundada.

E por fim, como o recomendador se trata de um protótipo e não um sistema completo, não foram implementadas funcionalidades que não fossem de atendimento ao objetivo principal de realizar recomendações.

Por exemplo, caso um usuário queira se cadastrar no sistema ele não consegue, pois essa opção está disponível apenas pela ferramenta de administração do Django. Então quem criou os usuários para a realização dos testes foi o administrador do sistema.

Outra funcionalidade que os usuários sentiram falta no momento de utilização da ferramenta foi não existir uma resenha ou resumo sobre os livros no próprio recomendador. Toda vez que algum livro que a pessoa não conhecia era sugerido, precisava-se ir pesquisar na internet um resumo sobre ele.

Entretanto, um dificultador para desenvolver essa funcionalidade é que o

dataset utilizado não contém resenhas dos livros, então haveria um certo trabalho para encontrar essas informações e importá-las no banco de dados.

REFERÊNCIAS

ADOMAVICIUS, G.; TUZHILIN, A.. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. **Ieee Transactions On Knowledge And Data Engineering**, [s.l.], v. 17, n. 6, p.734-749, jun. 2005. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/tkde.2005.99. Disponível em: <http://pages.stern.nyu.edu/~atuzhili/pdf/TKDE-Paper-as-Printed.pdf>. Acesso em: 29 abr. 2019.

ALVES, Carolina Limeira. **SISTEMAS DE RECOMENDAÇÃO DE CONTEÚDO**: UMA ANÁLISE SOBRE A EXPERIÊNCIA DO USUÁRIO EM PRODUTOS DIGITAIS. 2015. 179 f. Dissertação (Mestrado) - Curso de Programa de Pós-graduação em Design, PontifÍcia Universidade CatÓlica do Rio de Janeiro - Puc-rio, Rio de Janeiro, 2015. Disponível em: <https://www.maxwell.vrac.puc-rio.br/colecao.php?strSecao=resultado&nrSeq=25573@1>. Acesso em: 08 maio 2019.

BOKDE, Dheeraj; GIRASE, Sheetal; MUKHOPADHYAY, Debajyoti. Matrix Factorization Model in Collaborative Filtering Algorithms: A Survey. **Procedia Computer Science**, Pune, v. 49, p.136-146, 2015. Elsevier BV. http://dx.doi.org/10.1016/j.procs.2015.04.237. Disponível em: <https://reader.elsevier.com/reader/sd/pii/S1877050915007462?token=2333060245740D609A81E65D53B195204F8F4A1F89603D30A55E96BB5266C28470DFCFAE15A08920B3A7341FDB758CDE>. Acesso em: 09 jun. 2019.

BRAGA, Luis Paulo Vieira. **Introdução à mineração de dados**. 2. ed. Rio de Janeiro: E-papers, 2005.

CAMILO, Cássio Oliveira. **Mineração de Dados:** Conceitos, Tarefas, Métodos e Ferramentas. 2009. 29 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás, Goiás, 2009. Disponível em: <http://www.portal.inf.ufg.br/sites/default/files/uploads/relatorios-tecnicos/RT-

INF_001-09.pdf>. Acesso em: 31 maio 2019.

CARVALHO, D. R.; DALLAGASSA, M. Mineração de dados: aplicações, ferramentas, tipos de aprendizado e outros subtemas. **AtoZ**: novas práticas em informação e conhecimento, Curitiba, v. 3, n. 2, p. 82-86, jul./dez. 2014. Disponível em: <http://www.atoz.ufpr.br>. Acesso em: 25/05/2019.

CASTRO, Leandro Nunes de; FERRARI, Daniel Gomes. **Introdução à mineração de dados**: Conceitos básicos, algoritmos e aplicações. São Paulo: Saraiva, 2016. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/978-85-472-0100-5/cfi/2!/4/4@0.00:0.00>. Acesso em: 15 abr. 2019.

CAZELLA, Silvio César. **Aplicando a Relevância da Opinião de Usuários em Sistema de Recomendação para Pesquisadores.** 2006. 99 f. Tese (Doutorado) - Curso de Pós-graduação em Ciência da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2006.

CERQUEIRA, Luiz Henrique Marino; COELLO, Juan Manuel Adán. AVALIAÇÃO COMPARATIVA DE ALGORITMOS PARA SISTEMAS DE RECOMENDAÇÃO EM MÚLTIPLOS DOMÍNOS: PREVISÃO DO DESEMPENHO DE ESTUDANTES A PARTIR DE SUA INTERAÇÃO COM UM SISTEMA TUTOR. In: XVIII ENCONTRO DE INICIAÇÃO CIENTÍFICA, 18., 2013, Campinas. **Anais do III Encontro de Iniciação em Desenvolvimento Tecnológico e Inovação**. Campinas: PUC Campinas, 2013. p. 1 – 5

CUI, Bei-bei. Design and Implementation of Movie Recommendation System Based on Knn Collaborative Filtering Algorithm. **Itm Web Of Conferences**, Beijing, v. 12, p.1-5, 2017. EDP Sciences. http://dx.doi.org/10.1051/itmconf/20171204008. Disponível em: <https://www.researchgate.net/publication/319487277_Design_and_Implementation_of_Movie_Recommendation_System_Based_on_Knn_Collaborative_Filtering_Algorithm>. Acesso em: 06 jun. 2019.

DABBURA, Imad. **K-means Clustering**: Algorithm, Applications, Evaluation Methods, and Drawbacks. 2018. Disponível em: <https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a>. Acesso em: 05 nov. 2019.

FAYYAD, U. M., Piatetsky Shapiro, G., Smyth, P. & Uthurusamy, R. "**Advances in Knowledge Discovery and Data Mining**" 1996, AAAIPress, The Mit Press.

FERREIRA, Fernando Colmenero; OLIVEIRA, Adicinéia Aparecida. OS SISTEMAS DE RECOMENDAÇÃO NA WEB COMO DETERMINANTES PRESCRITIVOS NA TOMADA DE DECISÃO. **Journal Of Information Systems And Technology Management,** [s.l.], v. 9, n. 2, p.353-3668, 29 ago. 2012. TECSI. http://dx.doi.org/10.4301/s1807-17752012000200008. Disponível em: <http://www.jistem.fea.usp.br/index.php/jistem/article/view/10.4301%252FS1807-17752012000200008/313>. Acesso em: 08 maio 2019.

HALLAK, Ricardo; PEREIRA FILHO, Augusto José. Metodologia para análise de desempenho de simulações de sistemas convectivos na região metropolitana de São Paulo com o modelo ARPS: sensibilidade a variações com os esquemas de advecção e assimilação de dados. **Revista Brasileira de Meteorologia**, São Paulo, v. 26, n. 4, p.591-608, dez. 2011. FapUNIFESP (SciELO). http://dx.doi.org/10.1590/s0102-77862011000400009. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0102-77862011000400009>. Acesso em: 09 jun. 2019.

HIDASI, Balazs. Factorization models for context-aware recommendations. **Infocommunications Journal**, Budapest, v. , n. 4, p.27-344, dez. 2014. Disponível em: <http://www.hidasi.eu/content/italsx_infocomm.pdf>. Acesso em: 17 abr. 2019.
KÖRTING, Thales Sehn. **How kNN algorithm works**. 2014. Disponível em: <https://www.youtube.com/watch?v=UqYde-LULfs>. Acesso em: 08 jun. 2019.

KEMP, Simon. **DIGITAL 2019**: GLOBAL INTERNET USE ACCELERATES. 2019. Disponível em: <https://wearesocial.com/blog/2019/01/digital-2019-global-internet-use-accelerates>. Acesso em: 01 jul. 2019.

KUMAR, Prabhat; CHALOTRA, Sherry. An Efficient Recommender System using Hierarchical Clustering Algorithm. **International Journal Of Computer Science Trends And Technology (IJCST)**, Punjab-india, v. 2, n. 4, p.1-6, ago. 2014. Disponível em: <https://pdfs.semanticscholar.org/9f8e/1945eb105352cd84f9c2f33c444adc8f3b73.pdf?_ga=2.178853094.377995886.1572120983-57673534.1557964125>. Acesso em: 20 ago. 2019.

MCKINNEY, Wes. **Python para Análise de Dados**. 2. ed. São Paulo: Novatec, 2018. 615 p.

MONARD, Maria Carolina; BARANAUSKAS, José Augusto. **Sistemas Inteligentes**: Fundamentos e Aplicações. Barueri: Manole Ltda, 2003. 139 p. Disponível em: <http://dcm.ffclrp.usp.br/~augusto/publications/2003-sistemas-inteligentes-cap4.pdf>. Acesso em: 10 jun. 2019.

MOURA, Karina. **KDD Process**: Ciclo de vida dos dados #1. 2019. Disponível em: <https://medium.com/@kvmoura/kdd-process-9b8e3062142>. Acesso em: 25 set. 2019.

PHORASIM, Phongsavanh; YU, Lasheng. Movies recommendation system using collaborative filtering and k-means. **International Journal Of Advanced Computer Research**, Changsha, v. 7, n. 29, p.52-59, 1 nov. 2016. Disponível em: <https://pdfs.semanticscholar.org/1422/a2ff644eb3b91b1f2f9f8785083c53e87a40.pdf>. Acesso em: 10 jun. 2019.

REATEGUI, Eliseo Berni; CAZELLA, Sílvio César. Sistemas de Recomendação. In: XXV CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 25., 2005, São Leopoldo. **A Universalidade da Computação: Um agente de Inovação e Conhecimento.** São Leopoldo: Unissinos, 2005. p. 306 - 348. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.2811&rep=rep1&type=pdf>. Acesso em: 08 maio 2019.

RESNICK, Paul; VARIAN, Hal R.. Recommender Systems. **Communications Of The Acm**, [s.l.], v. 3, n. 40, p.56-58, mar. 1997. Disponível em: <http://www.inf.unibz.it/~ricci/ISR/papers/resnick-varian97.pdf>. Acesso em: 08 maio 2019.

RICCI, Francesco; ROKACH, Lior; SHAPIRA, Bracha. **Introduction to Recommender Systems Handbook**. Disponível em: <http://www.inf.unibz.it/~ricci/papers/intro-rec-sys-handbook.pdf>. Acesso em: 13 maio 2019.

SANTANA, Marlesson. **Deep Learning para Sistemas de Recomendação:** (Parte 1)—Introdução. Disponível em: <https://medium.com/data-hackers/deep-learning-para-sistemas-de-recomenda%C3%A7%C3%A3o-parte-1-introdu%C3%A7%C3%A3o-b19a896c471e>. Acesso em: 13 maio 2019.

SATPATHY, Tridibesh. **A Guide to the SCRUM BODY OF KNOWLEDGE**. Phoenix: Scrumstudy, 2016. 342 p. Disponível em: <https://www.scrumstudy.com/SBOK/SCRUMstudy-SBOK-Guide-2016.pdf>. Acesso em: 23 jun. 2019.

SERRANO, Luis. **How does Netflix recommend movies? Matrix Factorization**. 2018. Disponível em: <https://www.youtube.com/watch?v=ZspR5PZemcs>. Acesso em: 09 jun. 2019.

SOUZA, Bruno de Figueiredo Melo e. **Modelos de fatoração matricial para recomendação de vídeos**. 2011. 66 f. Dissertação (Mestrado) - Curso de Programa de Pós-graduação em Informática, Departamento de Informática da Puc-rio, PontifÍcia Universidade CatÓlica do Rio de Janeiro - Puc-rio, Rio de Janeiro, 2011. Disponível em: <https://www.maxwell.vrac.puc-rio.br/colecao.php?strSecao=resultado&nrSeq=19273@1>. Acesso em: 08 maio 2019.

SOUZA, Renata Ghisloti Duarte de. **Sistemas de Recomendação:** Aplicando

Sistemas de Recomendação em Situações Práticas. 2014. Artigo publicado por IBM Developer. Disponível em: <https://www.ibm.com/developerworks/br/local/data/sistemas_recomendacao/index.html>. Acesso em: 22 abr. 2019.

TAKAHASHI, Marcos M.. **Estudo comparativo de Algoritmos de Recomendação.** 2015. 49 f. TCC (Graduação) - Curso de Pós-graduação em Ciência da Computação, Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2015. Disponível em: <https://bcc.ime.usp.br/tccs/2014/marcost/monografia_final.pdf>. Acesso em: 01 maio 2019.

TRONCHONI, Alex B. et al. Descoberta de conhecimento em base de dados de eventos de desligamentos de empresas de distribuição. **Sba**: Controle & Automação Sociedade Brasileira de Automatica, Porto Alegre, v. 21, n. 2, p.185-200, abr. 2010. FapUNIFESP (SciELO). http://dx.doi.org/10.1590/s0103-17592010000200007. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-17592010000200007#fig5>. Acesso em: 25 maio 2019.

WANG, Leilei; CHEN, Zhigang; WU, Jia. **An Opportunistic Routing for Data Forwarding Based on Vehicle Mobility Association in Vehicular Ad Hoc Networks**. Information, Changsha, v. 8, n. 4, p.140-158, 7 nov. 2017. MDPI AG. http://dx.doi.org/10.3390/info8040140.

ZIKOPOULOS, Paul et al. **Understanding Big Data**: Analytics for Entreprise Class Hadoop and Streaming Data. Chicago: Mc Graw Hill, 2012. 166 p. Disponível em: <https://www.immagic.com/eLibrary/ARCHIVES/EBOOKS/I111025E.pdf>. Acesso em: 16 maio 2019.

APÊNDICE A – RESPOSTAS DA PESQUISA AVALIATIVA

	Cleber
1)	Você já leu algum dos títulos recomendados? Não
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 2
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: terror, ficção científica, política e filosofia. Autores: Stephen King, Ayn Rand, R. R. Tolkien
6)	Cite 2 hobbies seus: Violão, videogame, leitura

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
The Hobbit	não	5	5	0	0
The Catcher in the Rye	não	3	5	2	4
Slaughterhouse-Five 4	não	2	5	3	9
Angels & Demons (Robert Langdon, #1)	não	4	5	1	1
The Lord of the Rings (The Lord of the Rings, #1-3)	não	5	5	0	0
Foundation and Empire (Foundation #2)	não	4	5	1	1
Pride and Prejudice	não	2	5	3	9
Foundation (Foundation #1)	não	4	5	1	1
Hamlet	não	3	5	2	4
Harry Potter and the Half-Blood Prince (Harry Potter, #6)	não	2	5	3	9
			MAE	**1,6**	
				RMSE	**1,9**

	Aline
1)	Você já leu algum dos títulos recomendados? Sim
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 2
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Ficção cintífica, fantasia. Autores: J.D. Robb, Sidney Sheldon, Danielle Steel, Stephen King
6)	Cite 2 hobbies seus: Assistir séries, ler e correr.

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
The Catcher in the Rye	não	3	5	2	4

Slaughterhouse-Five	não	1	5	4	16
Pride and Prejudice	não	5	5	0	0
The Hitchhiker's Guide to the Galaxy (Hitchhiker's Guide to the Galaxy, #1)	não	1	5	4	16
Memoirs of a Geisha	não	5	5	0	0
Harry Potter and the Prisoner of Azkaban (Harry Potter, #3)	sim	5	5	0	0
Water for Elephants	não	3	5	2	4
The Kite Runner	sim	5	5	0	0
One Hundred Years of Solitude	não	1	5	4	16
Life of Pi	não	1	5	4	16
			MEA	**2**	
				RMSE	**2,7**

	Morbini
1)	Você já leu algum dos títulos recomendados? Não
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 1
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Terror, policial, ação, fantasia, ficção
6)	Cite 2 hobbies seus: Academia e investimentos.

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
Foundation and Empire (Foundation #2)	não	3	5	2	4
A Deepness in the Sky (Zones of Thought, #2)	não	3	5	2	4
Slaughterhouse-Five	não	4	5	1	1
Solaris	não	2	5	3	9
The Tombs of Atuan (Earthsea Cycle, #2)	não	2	5	3	9
The Time Machine	não	4	5	1	1
State of Wonder	não	3	5	2	4
Lord of Light	não	3	5	2	4
Anansi Boys	não	3	5	2	4
I, Robot (Robot #0.1)	não	5	5	0	0
			MEA	**1,8**	
				RMSE	**2**

	Gamba
1)	Você já leu algum dos títulos recomendados? Não
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 2
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Terror, autoajuda, computação, biografias. Autores: Dan Brown.
6)	Cite 2 hobbies seus: Skate, corrida, jogos online, assistir a vídeos educativos.

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
Angels & Demons (Robert Langdon, #1)	sim	5	5	0	0
Pride and Prejudice	sim	1	5	4	16
The Hitchhiker's Guide to the Galaxy (Hitchhiker's Guide to the Galaxy, #1)	não	5	5	0	0
Divergent (Divergent, #1)	sim	1	5	4	16
The Hobbit	sim	4	5	1	1
Miss Peregrine's Home for Peculiar Children (Miss Peregrine's Peculiar Children, #1)	não	4	5	1	1
Gone with the Wind	sim	3	5	2	4
Catching Fire (The Hunger Games, #2)	sim	4	5	1	1
Animal Farm	não	3	5	2	4
Mockingjay (The Hunger Games, #3)	sim	4	5	1	1
			MEA	1,6	
				RMSE	**2,1**

	Samuel
1)	Você já leu algum dos títulos recomendados? Sim
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 6
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Suspense, terror, drama, baseado em fatos reais, ficção.
6)	Cite 2 hobbies seus: Pescar, acampar, assistir seriados.

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
The Catcher in the Rye	não	3	5	2	4
To Kill a Mockingbird	não	4	5	1	1

Slaughterhouse-Five	não	3	5	2	4
The Kite Runner	não	4	5	1	1
Memoirs of a Geisha	sim	4	5	1	1
Harry Potter and the Half-Blood Prince (Harry Potter, #6)	sim	3	5	2	4
Harry Potter and the Order of the Phoenix (Harry Potter, #5)	sim	5	5	0	0
Water for Elephants	não	4	5	1	1
Harry Potter and the Prisoner of Azkaban (Harry Potter, #3)	sim	5	5	0	0
The Sun Also Rises	não	1	5	4	16
			MEA	1,4	
				RMSE	**1,8**

	Cassiano
1)	Você já leu algum dos títulos recomendados? Não
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 5
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Fantasia, ficção científica, históricos.
6)	Cite 2 hobbies seus: Programas de humor, programação e eletrônica.

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
The Catcher in the Rye	não	2	5	3	9
To Kill a Mockingbird	não	4	5	1	1
Slaughterhouse-Five	não	5	5	0	0
The Kite Runner	não	2	5	3	9
Pride and Prejudice	não	3	5	2	4
Memoirs of a Geisha	não	4	5	1	1
1984	não	4	5	1	1
The Hobbit	não	2	5	3	9
The Sun Also Rises	não	4	5	1	1
One Hundred Years of Solitude	não	3	5	2	4
			MEA	1,7	
				RMSE	**2,0**

	Rodrigo
1)	Você já leu algum dos títulos recomendados? Não
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 1

5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Ficção, suspense, livros históricos, livros de programação.
6)	Cite 2 hobbies seus: Cursos de programação e videogame

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
Pride and Prejudice	não	1	5	4	16
The Kite Runner	não	2	5	3	9
Slaughterhouse-Five	não	5	5	0	0
One Hundred Years of Solitude	não	3	5	2	4
Water for Elephants	não	3	5	2	4
The Hunger Games (The Hunger Games, #1)	não	3	5	2	4
Memoirs of a Geisha	não	2	5	3	9
The Sun Also Rises	não	2	5	3	9
Life of Pi	não	3	5	2	4
The Stand	não	4	5	1	1
			MEA	2,2	
				RMSE	**2,4**

	Bandiera
1)	Você já leu algum dos títulos recomendados? Sim
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 1
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Ficção, gerras, nazismo, velho oeste.
6)	Cite 2 hobbies seus: Correr, jogar futebol e programar.

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
Harry Potter and the Half-Blood Prince (Harry Potter, #6)	sim	5	5	0	0
Harry Potter and the Order of the Phoenix (Harry Potter, #5)	sim	5	5	0	0
The Hobbit	não	5	5	0	0
1984	não	3	5	2	4
Pride and Prejudice	sim	4	5	1	1
The Hitchhiker's Guide to the Galaxy (Hitchhiker's Guide to the Galaxy, #1)	não	5	5	0	0
The Kite Runner	não	4	5	1	1
One Hundred Years of Solitude	não	4	5	1	1

Harry Potter and the Chamber of Secrets (Harry Potter, #2)	sim	5	5	0	0
Slaughterhouse-Five	não	4	5	1	1
			MEA	0,6	
				RMSE	**0,9**

	Nelci
1)	Você já leu algum dos títulos recomendados? Sim
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 2
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Espiritas, romances, aventura, desenho animado, mistério.
6)	Cite 2 hobbies seus: Dançar, assistir filmes

Livro	Já havia lido?	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado
The Hobbit	sim	5	5	0	0
Divergent (Divergent, #1)	não	3	5	2	4
1984	não	2	5	3	9
Harry Potter and the Chamber of Secrets (Harry Potter, #2)	sim	5	5	0	0
Animal Farm	não	4	5	1	1
City of Glass (The Mortal Instruments, #3)	não	5	5	0	0
No Country for Old Men	não	4	5	1	1
Anna Karenina	não	5	5	0	0
Insurgent (Divergent, #2)	não	3	5	2	4
The Maze Runner (Maze Runner, #1)	não	4	5	1	1
			MEA	1	
				RMSE	**1,4**

	Carlos
1)	Você já leu algum dos títulos recomendados? Não
2)	Caso você já tenha lido algum dos livros recomendados, pontue-o de 1 a 5
3)	Para os livros que você ainda não leu classifique seu nível de interesse pela obra
4)	Quantos livros há na sua lista de leitura para os próximos 6 meses? 0
5)	Buscando compreender o perfil do leitor cite um gênero, ou autor, ou tipo de livro que você gosta.
	Gêneros: Ação, suspense, ficção científica, espirita.
6)	Cite 2 hobbies seus: Pescar, cortar grama, jogar cartas

Livro	Já havia	Pontos usuário	Pontos SR	Diferença Pontos	Diferenças ao quadrado

	lido?				
To Kill a Mockingbird	não	4	5	1	1
Atlas Shrugged	não	2	5	3	9
The Catcher in the Rye	não	3	5	2	4
The Maze Runner (Maze Runner, #1)	não	5	5	0	0
Insurgent (Divergent, #2)	não	5	5	0	0
Bridget Jones's Diary (Bridget Jones, #1)	não	4	5	1	1
The Other Boleyn Girl (The Plantagenet and Tudor Novels, #9)	não	2	5	3	9
Clockwork Angel (The Infernal Devices, #1)	não	3	5	2	4
City of Bones (The Mortal Instruments, #1)	não	3	5	2	4
The Third Twin	não	4	5	1	1
			MEA	1,5	
				RMSE	**1,8**