Guilherme Müller Moreira

Apresentação de Proposta: Abordagem Genérica para Sistemas de Recomendações

Brasil

13 de novembro de 2018

Guilherme Müller Moreira

Apresentação de Proposta: Abordagem Genérica para Sistemas de Recomendações

Apresentação da proposta de projeto para trabalho de conclusão do curso de Sistemas de Informação do CEFET Nova Friburgo. O projeto propõe uma abordagem genérica para sitemas de recomendação.

Centro de Ensino Técnico Federal Celso Suckow da Fonseca

Uned Nova Friburgo

Sistemas de Informação

Orientador: Marco André Abud Kappel

Coorientador: Luis Claudio Batista da Silva

Brasil

13 de novembro de 2018

Resumo

FAZER

Palavras-chaves: Recomendações. Machine Learning.

Sumário

1 Introdução

O crescimento da internet e dos meios digitais popularizou plataformas digitais de serviços e comércio de produtos. O número de itens nestes sistemas podem atingir quantidade e variedades grandes o suficiente para que seja difícil para o usuário encontrar o que busca sem o auxílio de mecanismos de filtragem. Um desses mecanismos são os sitemas de recomendação, que através de técnicas de aprendizado de máquina e análise de dados buscam entregar conteúdo personalizado para aos usuários. Em modelos de negócio digitais a eficiência em oferecer informações relevantes pode ter grande impacto no sucesso do empreendimento.

Os sistemas de recomendação funcionam através da coleta de informações dos usuários ou dos elementos de um sistema, processando-as e identificando similaridades. Estas informações podem ser de diferentes tipos, alguns exemplos são: dados comportamentais, avaliações de usuário, características textuais e dados audiovisuais. O processo de identificação das similaridades assim como o método de seleção das Recomendações podem ser implementados com diferentes abordagens. Este trabalho propõe uma abordagem genérica aos SRs utilizando técnicas de filtragem colaborativa.

Algoritmos de filtragem colaborativa funcionam organizando os usuários ou itens de um sistema em grupos com base na similaridade que possuem entre si. Os índices de similaridade são extraídos analisando os dados de cada usuário e item do sistema. As recomendações para um usuário são geradas observando os usuários que mais se assemelham a ele, partindo do pressuposto de que os padrões comportamentais permanecem numa tendência.

1.1 Problemas e Desafios

A construção de um sistema de recomendações não é trivial e apresenta alguns desafios. Dentre as dificuldades recorrentes da implementação de SRs Xiaoyuan e Khoshgoftaar destacam:

- **Esparsidade dos Dados:** Refere-se a necessidade do sistema de recomendações de lidar com itens cujos dados coletados ainda não são suficientes para estabelecer as similaridades com outros itens eficientemente. Este problema é comum quando um novo elemento é adicionado ao sistema. Em cenários como este, é improvável que o SR forneça boas recomendações.
- **Escalabilidade:** Conforme o número de usuários e itens de um sistema de recomen-

dações cresce, o custo computacional para a identificação das similaridades pode se tornar impraticável. Com uma base de milhões de usuários e itens, um algoritmo de filtragem colaborativa com complexidade O(n) já se torna pesado demais (XIAOYUAN; KHOSHGOFTAAR, 2009).

- **Sinonímia:** Sistemas de recomendações precisam lidar com a existência de itens muito semelhantes porém com nomes ligeiramente diferentes.
- **Ovelhas Cinzas:** Este termo se refere aos usuários com gostos que não são suficientemente semelhantes aos de nenhum outro grupo de usuários. O sistema deve ser capaz de identificar este tipo de usuário de forma que ele não exerça tanta influência sobre as recomendações.
- ***Shilling Attack*:** Neste tipo de ataque os usuários tentam manipular as recomendações parar benefício próprio. Isto é feito inserindo dados propositalmente no sistema, aumentando ou diminuindo a probabilidade de um item ser recomendado. É importante que existam mecanismos que evitem este tipo de manipulação de resultados num SR.

Os problemas acima identificados são exemplos que demonstram a não trivialidade de implementação de um SR. Além disto existem outros desafios que precisam ser superados neste tipo de sistema, relativos a escolhas de tecnologia, arquitetura, modelagem e de abordagens técnicas.

1.2 Proposta

Este trabalho propõe o desenvolvimento de um sistema genérico de recomendações baseado na técnica "Filtragem Colaborativa". Este sistema será utilizado como uma ferramenta consumida por aplicações de terceiros, recebendo dados e entregando recomendações. Para satisfazer este requisito, o projeto precisa ser elaborado sobre uma abstração que o torne independente de domínio e de tipos de cliente. A plataforma, conforme concebida, irá ser executada paralelamente às aplicações clientes, encapsulando e solucionando os desafios inerentes aos SRs.

Outra característica importante do projeto é ser capaz de intercambear diferentes algorítmos de filtragem colaborativa, a fim de se adequar às diferentes necessidades das aplicações clientes. Esta flexibilidade tem ainda outras motivações, que serão detalhadas ao longo deste documento.

O sistema proposto será uma aplicação de *CLI* e utilizará dois tipos de interface para se comunicar com os clientes. A primeira delas, utilizada para coleta de dados, será a leitura de logs de atividades gerados pela aplicação cliente e que seguem a abstração

definida pela ferramenta. Esta leitura será executada em períodos de tempo pré configurados. Já o consumo das recomendações será feito pela segunda interface de comunicação: uma API Rest. O papel desta API, além de entregar as recomendações é também o de permitir inserir dados específicos, descritos ao longo deste documento.

1.3 Motivações

O desenvolvimento da ferramenta proposta possui duas principais motivações. A primeira delas, relacionada às dificuldades de se desenvolver um SR, é facilitar o uso de recomendações em qualquer tipo de aplicação, com o menor custo de implementação possível. A ferramenta proposta permitirá que as aplicações clientes lidem apenas com a inserção de dados e consumo das recomendações. Tornar a implementação da funcionalidade de recomendações mais trivial permitirá que sistemas de menor porte possam oferecer este tipo de recurso.

A segunda motivação esta relacionada a capacidade de troca dos algorítmos utilizados pela ferramenta. Este recurso permitirá a análise da eficiência dos algoritmos em diferentes cenários.

1.4 Objetivos

Os objetivos do projeto foram divididos em macro objetivos, que caracterizam os principais requisitos do projeto, e micro objetivos, que caracterizam as principais entregas que compõem os macro objetivos.

1.4.1 Macro Objetivos

- Ser capaz de coletar dados padronizados de uma fonte preenchida pelos clientes do sistema.
- Ser aplicável a qualquer tipo de aplicação cliente.
- Permitir a escolha dos seguintes componentes do sistema via configuração: Índice de Similaridade, Técnica de Cluster e o Algoritmo de Seleção das Recomendações.
- Estabelecer indicadores de eficiência para diferentes configurações do item anterior e desenvolver uma análise comparatória entre elas.

1.4.2 Micro Objetivos

- Estabelecer uma abstração dos elementos comuns em sistemas de recomendação.

- Definir uma interface de dados a ser utilizada pelo cliente ao fornecer dados para a ferramenta.
- Estabelecer regras de leitura dos dados fornecidos pelo cliente.
- Projetar o sistema de maneira modular.
- Implementar ao menos duas opções para cada componente cambiável do sistema.
- Implementar uma API Rest com autenticação para o consumo das recomendações.

1.5 Estrutura do Documento

Este documento está dividido da seguinte maneira: O capítulo "Revisão Bibliográfica"apresenta uma revisão das principais referencias da bibliografia utilizada na montagem da proposta. Cada seção deste capítulo se refere a um artigo ou livro utilizado. Ainda neste capítulo a seção "Ferramentas Relacionadas"apresenta os resultados da pesquisa por software existentes que tenham objetivos semelhantes aos estabelecidos nesta proposta.

O Capítulo "Metodologia"contém detalhes a cerca do que foi planejado para a parte técnica do projeto, detalhendo e justificando as técnicas utilizadas para a construção da ferramenta projetada.

2 Revisão Bibliográfica

Neste capítulo serão revisados os materiais utilizados como fonte de pesquisa e referência para a elaboração desta proposta. Serão analisados também softwares existentes no mercado e suas diferenças para a ferramenta proposta neste documento.

2.1 Segaran (2007)

O livro de Segaran aborda teoria e prática de diversos tópicos relacionados a sistemas de recomendações. Dentre os temas tratados a Filtragem Colaborativa é definida como uma aplicação de um conceito conhecido como Inteligência Coletiva. Este termo se refere a informações que são derivadas de um coletivo de indivíduos. No seu conceito mais básico, técnicas de CF funciona identificando usuários com característica ou gostos semelhantes entre uma base de usuários (SEGARAN, 2007).

Entre exemplos e definições Segaran estabelece um fluxo básico para o processo de recomendação, decorrido em três etapas. A primeira etapa consiste em coletar os dados que relacionam os usuários aos itens do sistema. Estes dados representam padrões comportamentais ou perfis de interesse. A segunda fase do processo consiste em estabelecer um índice de similaridade utilizado para agrupar os perfis de usuários. Este agrupamento é feito através do calculo de similaridade entre cada usuário do sistema, estabelecendo uma matriz "de-para". Com estas informações, a terceira etapa do processo consiste em identificar os usuários mais similares e deles extrair como recomendações os itens com maior relevância. A relevância de um item pode ser calculada de diferentes maneiras.

Outro assunto abordado pelo livro é o uso de *clusters* para estabelecer os grupos de usuários e extrair os indivíduos de maior semelhança. Esta técnica foi escolhida para o projeto como forma de persistir as relações entre usuários identificadas nas aplicações clientes.

2.2 Xiaoyuan e Khoshgoftaar (2009)

O artigo de Xiaoyuan e Khoshgoftaar apresenta uma revisão das principais tecnologias e métodos utilizados em sistemas de recomendação. O trabalho foi importante para introduzir os principais conceitos que cercam o tema da filtragem colaborativa. O trabalho apresenta também os principais problemas que que sistemas de recomendações enfrentam.

Xiaoyuan e Khoshgoftaar dividem os algoritmos de filtragem colaborativa em três

grupos: Baseados em Memória, Baseados em Modelo e Híbridos. O primeiro tipo se caracterízam por serem simples e rápidos de implementar, considerando toda a base de dados para calcular similaridades e então oferecer recomendações. Geralmente, esta abordagem relaciona a similaridade dos usuários com base nos dados em comum que possuem a cerca dos itens do sistema. Estes dados podem ser avaliações, contagens de acesso etc. Das características negativas deste grupo de sistemas estão a ineficácia em lidar com dados esparços e o custo computacional elevado derivado da escala das bases de dados.

O segundo grupo de sistemas de recomendações, os sistemas baseados em modelo, busca superar os pontos negativos do primeiro. Esta abordagem utiliza técnicas de análise de dados e aprendizado de máquina para montar um modelo que permita fornecer recomendações.

Os sistemas de recomendação híbridos, categoria da ferramenta proposta neste documento, combínam características de filtragem colaborativa com filtragem baseada em conteúdo. Neste grupo, os algoritmos utilizam os dados que coleta dos usuários em relação aos itens assim como dados das características dos itens. combinando estes dois tipos de informações o sistema se torna mais eficiente contra a esparcidade de dados e as Ovelhas Cinzas.

O artigo estabelece uma introdução à diferentes algoritmos de cada grupo citado acima. Introduz também métricas de eficiência para SRs como erro absoluto médio e *Root Mean Squared Error*, que serão utilizadas para elaborar a análise de eficiência dos algoritimos implementados no projeto.

2.3 Das et al. (2007)

Este artigo apresenta uma proposta de solução para a sugestão de notícias do *website* Google News. A solução precisa satisfazer dois requisitos principais: Escalabilidade e Velocidade de Atualização do Modelo. Além disto é interessante que o algoritmo seja agnóstico e reutilizável em aplicações de diferentes domínios da empresa.

A solução proposta consiste em mesclar técnicas baseadas em memória e de modelo para gerar as recomendações. Da parte baseada em modelo são utilizadas duas técnicas de *cluster* - *PLSI* e MinHash - enquanto que dos métdos de memória foi utilizada a *item covisitation*. No modelo proposto, cada acesso a um *link* da plataforma *Google News* é registrado no histórico do usuário. O índice de similaridade dos usuários é calculado através da intersecção dos históricos de acesso enquanto que o algorítmo *PLSI* classifica os usuários em grupos e os links em gêneros.

A solução descrita no artigo de Das et al. (2007) se difere do projeto proposto neste documento em alguns pontos:

1. **Categorização dos Itens:** No modelo proposto por Das et al. (2007) os itens (notícias) são categorizados conforme os acessos que recebe. Visando a solucionar o problema da esparsidade de dados o projeto proposto possibilita a categorização dos itens adicionados. Esta categoria terá influência nas recomendações de forma que mesmo itens recém adicionados possam ser recomendados.

2. **Formato dos Dados:** Os dados utilizados na solução utilizada para a ferramenta do Google baseiam-se no histórico de acesso às notícias. A unidade escolhida foi a booleana - se um um usuário acessou uma notícia o valor é 1, do contrário é 0. A ferramenta descrita nesta proposta se difere em dois pontos: 1. Adota um sistema de verbos para indicar uma ação de um usuário; 2. Contabiliza as ações dos usuários de maneira incremental. Estas diferenças permitem identificar quais itens são mais relevantes assim como qual operação sobre o item é mais relevante para o usuário.

3. **Configuração dos Algoritmos:** A ferramenta génerica de recomendações conforme concebida se propõe a ser capaz de implementar e utilizar diferentes tipos de algorítmos de filtragem colaborativa, diferentemente da solução do referido artigo que se foca em apenas uma abordagem.

2.4 Greenacre (2008)

O material acadêmico do professor e autor Michael J. Greenacre apresenta as variações dos métodos euclidianos de cálculo de distância entre amostras de dados. Os algoritmos apresentados são:

- **Distância Euclidiana:** Utiliza o teorema de pitágoras para calcular a distância absoluta entre dois vetores n-dimensionais. Este método pode ser utilizado adicionando os dados extraídos dos clientes minimamente tratados como vetores para cálculo de distância. Esta distância pode ser interpretada como um índice de similaridade.

- **Distância Euclidiana Normalizada:** Utilizada para vetores cujas coordenadas não estão na mesma escala e precisam ser normalizadas antes do cálculo de distância.

- **Distância Euclidiana em Amostras de Contagem:** Apesar de estarem na mesma escala, a natureza do item contabilizado pode significar maior frequência de ocorrência (GREENACRE, 2008). Isto significa que os dados também precisam ser normalizados antes do cálculo de distância.

2.5 Ferramentas Relacionadas

A partir de buscas na internet diferentes ferramentas com propostas a cerca de recomendações foram encontradas. Esta seção irá elencar e comparar estes softwares com o projeto proposto.

- **Episerver:**[1] Enquadra-se no modelo de softwares como serviços. Possui foco em ecommerce e oferece diferentes pacotes com diferente funcionalidades. O *website* do produto não revela muitos detalhes, sugerindo que o Episerver é um conjunto de soluções para *e-commerce* que vão além das recomendações. Estas soluções são acopladas em sistemas de venda digital e executam uma série de atividades. O Módulo Episerver Perform se propõe a coletar dados dos usuários, importar o catálogo de produtos e entregar recomendações diretamente ao usuário. Se comparado com a proposta genérica apresentada por este documento, notam-se algumas diferenças. A primeira delas está na coleta de dados, que no sistema Episerver é uma responsabilidade do sistema. A ferramenta proposta aborda passivamente esta questão, apenas recebendo as informações dos sistemas clientes. Além disso, a ferramenta não é responsável por exibir as recomendações ao usuário e nem conhece a estrutura dos dados que recomenda. Desta maneira mantém-se a independencia do domínio, característica aparentemente não presente no software Episerver.

- **Strands:**[2] Semelhantemente ao software anterior, este produto tem foco em *e-commerce* e assume o pepel de coletor e exibidor das recomendações. Mais uma vez, devido ao fato de ser um sistema proprietário no modelo SaS, pouca informação sobre o funcionamento da plataforma é disponibilizada. Aparentemente, scripts de coleta de dados comportamentais são inseridos na loja virtual para identificar os padrões de navegação dos clientes. A entrega das recomendações é feita através da inserção de widgets que exibem produtos préviamente importados. Se comparado com a proposta nota-se que o sistema Strands não tem a pretenção de se manter abstrato ao domínio nem de oferecer diferentes implementações de algorítmos para escolha. Estas características são o que distinguem o produto citado do projeto proposto.

- **SLI Learning Recommendations:**[3] Semelhantemente aos outros sistemas apresentados, o *SLI Learning Recomendations* tem por foco *e-commerces* e funciona coletando dados através da inserção de scripts de coleta de dados na loja. Este sistema difere-se em considerar questões inerentes ao domínio das vendas digitais para efetuar recomendações. Os itens recomendados consideram fatores como venda

[1] https://www.episerver.com
[2] http://retail.strands.com/products/product-recommendations/
[3] https://www.sli-systems.com/solutions/product-recommendation

cruzada, aumento da média de valor dos carrinhos, etapas do processo de compra e conversão de possíveis clientes. Além disto este sistema possui uma API que permite adicionar recomendações em outros elementos como emails. O projeto proposto se difere por considerar exclusivamente os padrões comportamentais dos usuários e a categorização dos itens para efetuar as recomendações. Outro ponto de divergência é a abstração genérica da ferramenta projetada.

- **ActionML Universal Recommender:**[4] O projeto *Universal Recommender* se assemelha à ferramenta proposta em manter-se abstraído do domínio a fim de se tornar genérico. Este sistema funciona com base no conceito de eventos, ocorrencias na aplicação cliente que podem ser utilizadas como indicadores de preferência. Os eventos são classificados em dois tipos de acordo com uma modelagem prévia: Eventos primários e secundários. Um evento primário é o indicador de interesse mais forte na aplicação cliente e tem mais influência nas recomendações. Os eventos secundários podem ser quaisquer outras ações registradas no cliente que indiquem interesse do usuário. O sistema utiliza o algorítmo *Correlation Cross-Ocorrence* para gerar as recomendações. Das ferramentas apresentadas é a que mais se aproxima da proposta apresentada. Dentre as diferenças se destacam a necessidade de se modelar minimamente os eventos que serão inseridos no sistema, o consumo das recomendações, feito através de queries e não de uma API, e a abordagem híbrida utilizada. Sobre este último item, o *Universal Recommeder* permite que dados dos itens sejam inseridos no sistema enquanto que a ferramenta proposta permite apenas a categorização dos itens com o objetivo de melhorar a eficiência em cenários de dados esparços.

[4] https://github.com/actionml/universal-recommender

3 Metodologia

Neste capítulo serão detalhados o planejamento para o decorrer do projeto nos seguintes assuntos: Abstração do Modelo, Técnicas e Algoritmos, Tecnologias e Arquitetura do Sistema.

3.1 Abstração do Modelo

O desenvolvimento do projeto será baseado na construção de uma abstração que possa representar os elementos mais comuns dos sistemas de recomendações. Esta abstração deve ser capaz de representar elementos de qualquer domínio, mantendo a capacidade de identifica-los na aplicação cliente. Destes pressupostos estabeleceu-se o seguinte protótipo de abstração:

Tabela 1: Descrição dos Elementos da Abstração

Elemento	**Descrição**	**Formato**	**Escopo**
Ator	Representa algo ou alguém que executa atividades no cliente	ID Entidade + ID Elemento	Cliente
Ação	Representa uma funcionalidade ou atividade que atores podem executar no cliente	ID Único	Sistema
Objeto	Alvo de uma Ação executada por um Ator no cliente	ID Entidade + ID Elemento	Cliente
Categoria	Permite categorizar os objetos	ID Único	Sistema

Para todos os elementos da abstração descrita na tabela acima a ferramenta recebe um identificador, sem saber se o elemento existe ou não no cliente. Para os elementos pertencentes ao escopo dos clientes os identificadores são formados por dois valores, o id da entidade e o id do elemento. Esta chave composta é utilizada apenas para que o cliente identificar os itens quando consumirem as recomendações.

Os logs de atividade preenchidos pelos clientes devem conter, num formato fixo, os dados acima para representar um histórico de atividades de seus atores. Para as Categorias o sistema irá permitir o cadastro prévio via API. Isto será permitido para que, também através da API, seja possível informar índices de proxímidade entre categorias de objetos.

Em cenários de dados esparsos, as recomendações serão feitas considerando os itens das categorias mais próximas.

3.2 Técnicas e Algoritmos

Esta seção destina-se a detalhar os algorítmos que serão implementados no projeto assim como justificar suas ecolhas.

3.2.1 Algoritmos de Similaridade

Os dados utilizados pela ferramenta para identificar padrões comportamentais serão dados de contagem. A ferramenta irá contabilizar a quantidade de ocorrencias de cada combinação Ator x Ação x Objeto, identificando os comportamentos mais recorrentes dos atores nos sistemas clientes. Para identificar as similaridades a partir destes dados serão implementados os algoritmos:

- **Distância Euclidiana:** Calcula a distância absoluta entre dois vetores, quanto mais próximos maiores as semelhanças entre os dados. Por ser o algorítmo mais simples será utiliado como base para as comparações.
- **Distância Euclidiana Normalizada:** Normaliza os dados antes de efetuar o cálculo da distância. A normalização consiste em reduzir a influência das taxas de ocorrencia dos itens contabilizados, isto por que a contagem de um item pode ser mais alta por possuir uma taxa de ocorrencia maior que a dos outros. Outra etapa da normalizacão é efetuar o cálculo euclidiano não sobre valores absolutos, mas sobre a porcentagem de ocorrencias de um item em relação ao total de ocorrencias.
- **Distância Euclidiana Normalizada com Relevância:**Este algorítmo, além de efetuar as normalizações citadas no item anterior, leva em consideração o índice de relevância do objeto para o ator. Este índice varia entre 0 e 1 e é extraído a partir da recorrencia do objeto no histórico do ator assim como a recorrencia da categoria do objeto em questão.
- **Algorítmos de Pearson**: Diferentemente da distãncia euclidiana que identifica a distância entre dois vetores, o índice de pearson índica a tendencia de variação entre duas variáveis. Quanto mais próximas e proporcionais são as flutuações das variáveis, maior a similaridade entre elas. Serão implementadas cada variação da distância euclidiana citadas acima porém utilizando o índice de Pearson.

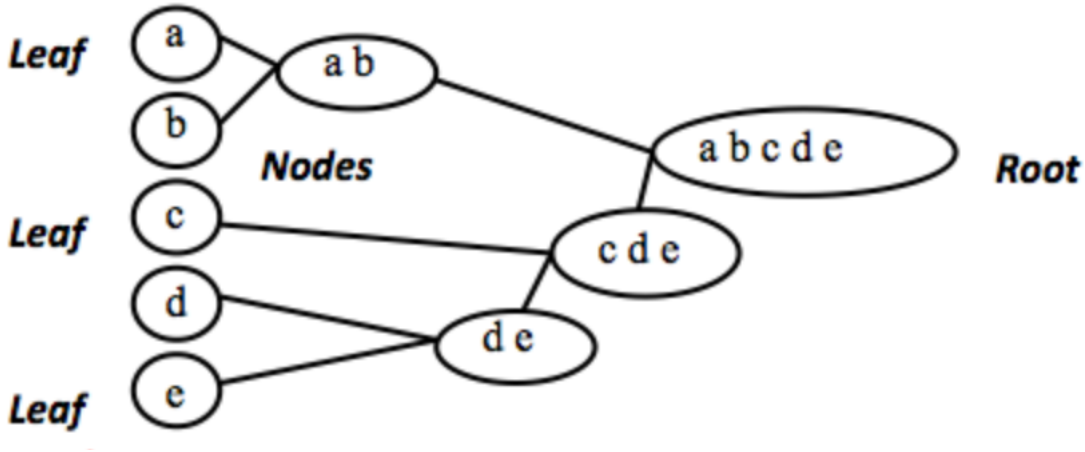
Através dos índices de similaridade a ferramenta estabelece uma matriz que contém a similaridade entre cada ator do sistema.

3.2.2 Algoritmo de Clusterização

A partir de uma matriz de similaridades entre atores é gerado um modelo de *clusters* que será utilizado para persistência e para consulta dos usuários com base na proximidade. Os algorítmos escolhidos para montagem dos clusters foram: Clusterização Hierárquica e Clusterização em Grafos.

Algorítmos de Clusterização Hierárquica se caracterizam por montar uma estrutura de árvore em que as folhas são os itens (Atores) e cada nível superior da hierarquia é formado por um cluster da junção dos dois nós diretamente abaixo. Nestes algorítmos a raíz da árvore é um único cluster contendo todos os outros nós da árvore.

Figura 1: Árvore de Clusters Hierárquicos



O segundo algoritmo planejado para ser desenvolvido é o de clusterização em grafos. Neste método cada ator é representado como um nó da estrutura. Os nós mais próximos são conectados entre si com arestas que possuem pesos. Para o projeto, o peso das arestas pode ser baseado no índice de similaridade dos atores. O grafo resultante deste método possui grupos de nós com várias conexões entre si, porém com poucas conexões com nós de outras seções. Estes grupos caracterizam os clusters.

A partir das estruturas apresentadas acima, algorítmos de seleção são utilizados para se encontrar os usuários mais próximos.

3.3 Tecnologias

Partindo dos objetivos do projeto a seguinte *stack* de desenvolvimento será utilizada:

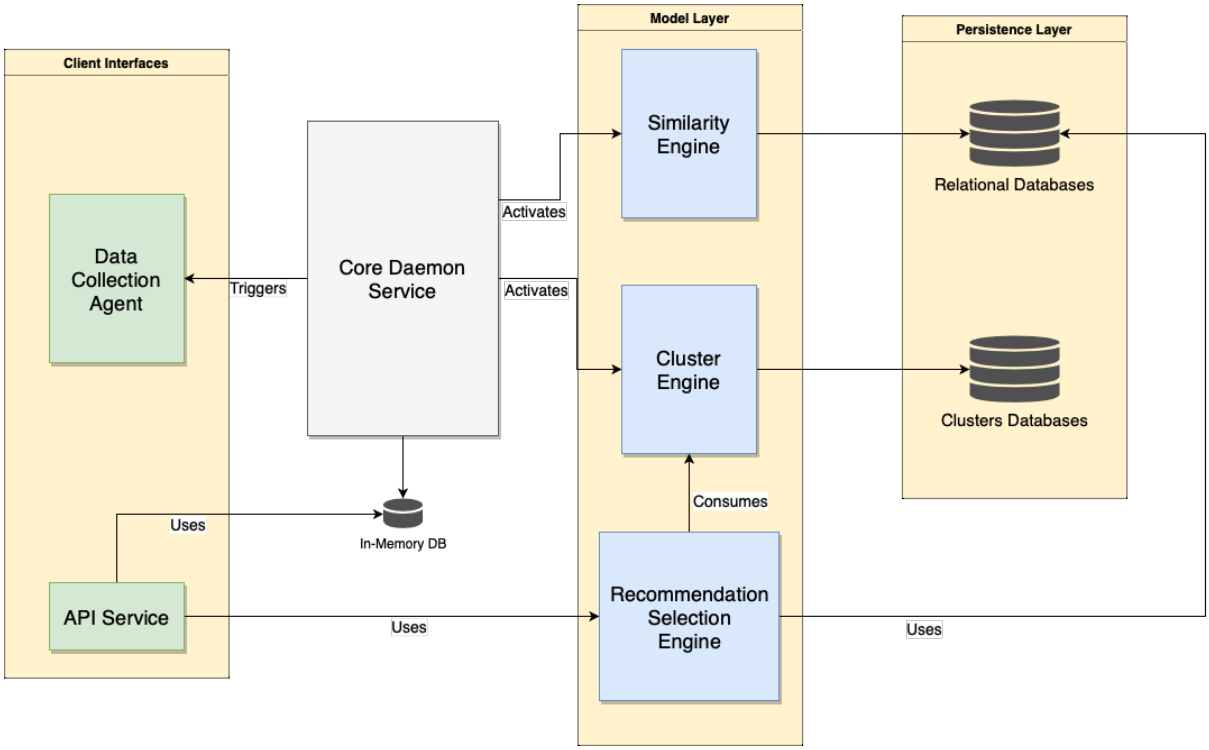
- **Python:** A linguagem será utilizada para desenvolver o coletor de dados, a API Rest e os algorítmos de similaridade.

- **MySQL:** O SGBD será utilizado para armazenamento dos históricos comportamentais extraídos dos logs dos clientes.
- **Redis:** Banco de dados em memória utilizado para cache de operações em prol de performance.
- **Neo4J:** Base de dados para armazenamento de grafos.
- **Docker:** Sistema de conteiners para gerenciamento de infraestrutura.
- **Git:** Sistema de controle de versão do projeto.

3.4 Arquitetura do Sistema

Esta seção apresenta o protótipo inicial da arquitetura do sistema. Serão descritos os papéis de cada componente do modelo.

Figura 2: Arquitetura de Componentes do Sistema



Na imagem acima os componentes na cor verde são os módulos e serviços que estabelecem interfaces de comunicação com as aplicações clientes. O agente de coleta de dados (*Data Collection Agent*) é o módulo responsável por iterar sobre os logs extraindo os dados. Este processo de leitura é disparado pelo núcleo do sistema (*Core Service*). O segundo elemento das interfaces de comunicação é a API que é responsável por retornar ao cliente as recomendações. A Api extrai as recomendações através do motor de seleção

de recomendações. Além disto, a API também se relaciona com o banco de dados em memória que será utilizado para disponibilizar configurações e cache de operações.

O componente central do sistema, o serviço daemon central (*Core Daemon Service*), carrega a responsabilidade de manter o processo de atualização de dados em execução. Este serviço oferece ainda uma interface em CLI para configurações de operação. Este componente dispara o agente de coleta de dados, ativa o motor de similaridade sobre os dados coletados e repassa os índices extraídos para o motor de clusterização. Assim como a API, este componente acessa a base de dados em memória para ler e inserir configurações.

Na camada de modelo estão presentes três motores: Motor de Similaridade, Motor de Clusterização e Motor de Seleção de Recomendações. O primeiro deles é responsável por estabelecer os índices de similaridade dos atores. O segundo, de clusterização, recebe os dados gerados no motor de similaridade por intermédio do *Core* e monta/atualiza a estrutura de clusters. Este componente é responsável ainda por extrair os atores por proximidade dos clusters. O último motor, de seleção de recomendações, seleciona objetos e ações com base nos atores retornados pelo motor de clusters. Cada um destes motores deve ser parametrizável de maneira que seja possível escolher qual algorítmo o motor irá utilizar.

3.5 Determinando Eficiência dos Algoritmos

Visto que os componentes do sistema podem ser parametrizados de forma que utilizem diferentes algorítmos um dos objetivos deste projeto é estabelecer métricas de eficiência e comparar os algorítmos.

4 Cronograma

A tabela abaixo apresenta o cronograma planejado para o desenvolvimento do projeto.

Atividade	Mar	Abr	Mai	Jun	Jul	Set	Out	Nov
Documentação de Requisitos	X							
Modelagem do Sistema	X	X						
Desenvolver Agente de Coleta de Dados		X						
Desenvolver Núcleo da Aplicação			X					
Desenvolver Motor de Similaridade			X	X				
Desenvolver Motor de Cluster				X	X			
Desenvolver Motor de Coleta de Recom.						X		
Coletar Dados de Eficiência							X	
Documentação e Monografia	X	X	X	X	X	X	X	X

Referências

DAS, A. et al. Google news personalization: Scalable online collaborative filtering. *Proceedings of the 16th International Conference on World Wide Web*, p. 9, 2007. Citado 3 vezes nas páginas 3, 9 e 10.

GREENACRE, M. *Measures of distance between samples: Euclidean*. 2008. Disponível em: <http://www.econ.upf.edu/~michael/stanford/maeb4.pdf>. Acesso em: 06.11.2018. Citado 2 vezes nas páginas 3 e 10.

SEGARAN, T. *Programming Collective Intelligence*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2007. Citado 2 vezes nas páginas 3 e 8.

XIAOYUAN, S.; KHOSHGOFTAAR, T. M. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, v. 2009, p. 19, 2009. Citado 4 vezes nas páginas 3, 4, 5 e 8.