IPG Politécnico da Guarda
Polytechnic of Guarda

RELATÓRIO DE PROJETO

Licenciatura em Engenharia Informática

João Manuel Perereira Rodrigues Coelho

dezembrol | 2015

Instituto Politécnico da Guarda

Escola Superior de Tecnologia e Gestão

Sugestão de conteúdos

João Manuel Pereira Rodrigues Coelho – nº 1010832

Projeto aplicado no Curso
De Engenharia Informática

Dezembro de 2015

Resumo

Este documento descreve o trabalho realizado no âmbito da Unidade Curricular do Projeto de Informática da Licenciatura em Engenharia Informática, na Escola superior de Tecnologia e Gestão do Instituto Politécnico da Guarda.

Com a evolução tecnológica a disponibilidade de informação cientifica e o facto da criação de conteúdo ser tão fácil, com o aumento de jogos de computador criados por empresas independentes e muitos outros fatores. Existe tanta informação e conteúdo disponível, que encontrar o que queremos é difícil, e encontrar o que nos interessa é ainda mais difícil.

O trabalho consistiu em desenvolver uma aplicação, para efeitos de demonstração, que permita ver um exemplo de sugestão de conteúdos em funcionamento.

Abstract

This document describe the work done under the discipline Projeto de Informática *in the graduation in* Engenharia Informática *from* Escola Superior de Tecnologia e Gestão *in the Instituto Politécnico da Guarda.*

With the evolution of technology the availability of scientific information and the fact that creating content is so easy, with the increase in computer games created by independent companies and many other facts. There is so much information and content available, that to find what we want is hard, to find what we like is harder.

The work consisted in developing an application, for testing and demonstration purposes, with the objective to allow to see an example of content suggestion at work.

Índice

Glossário

Amazon AWS – Amazon Web Services

EMR – *Elastic MapReduce*

ESTG – Escola Superior de Tecnologia e Gestão

Java EE – *Java Enterprise Edition*

1. Introdução

O projeto consiste na criação de uma aplicação, que pretende demonstrar um exemplo prático de sugestão automática de conteúdos, neste caso de filmes. Os utilizadores vêm filmes e pontuam o quanto gostaram (de 0 a 5). Quando pela ordem de um administrador é iniciado o trabalho de sugestão de conteúdos, que vão ser mostrados aos utilizadores quando terminado.

Nos últimos anos com a evolução da internet a quantidade de informação disponível tem crescido de maneira quase exponencial, encontrar o que queremos é cada vez mais difícil. Tanto que num grupo de pessoas o mais comum é ouvir "descobri", é um trabalho difícil, e só conseguimos assimilar uma quantidade de informação muito limitada por dia. Tem que existir uma solução.

Machine Learning (máquina aprendiz) é usado para prever o futuro usando informação passada, parece simples, mas na verdade é muito complexo. É possível prever o que cada pessoa é capaz de gostar de várias maneiras:

Filtro Colaborativo - Usando opiniões de vários utilizadores é possível criar grupos de interesse, e depois sugerir conteúdos dentro desse grupo que o utilizador se encontra.

Métodos de filtro colaborativo são baseados na recolha e análise de uma grande quantidade de informação do comportamento dos utilizadores e prever o que os utilizadores gostam baseando-se na similaridade com outros utilizadores. Como não analisa o conteúdo em si é possível sugerir itens complexos como filmes sem ter a "compreensão" do próprio item que está a sugerir.

O filtro colaborativo foi feito com a ideologia : quem é similar no passado vai ser similar no futuro.

Exemplos de uso:

Amazon,

Facebook,

Twitter.

Filtro baseado em conteúdo - Baseado na descrição do item e no histórico do utilizador, funciona com palavras chave que são usadas para descrever cada item. Também é criado um profile de cada utilizador usando interações passadas.

Este método sugere itens similares aos anteriormente vistos, pelo que a mudança

permanente do ser humano não é tida em conta. Não são necessárias grandes quantidades de dados, visto que cada utilizador é tratado de maneira completamente separada.

Híbrido - Pesquisa recente tem demonstrado que a sugestão híbrida, combinando filtro baseado em conteúdo e colaborativo, pode ser mais eficiente em alguns casos. Existem várias maneiras de implementar, pelo que não vou especificar visto que não foi usado [1].

Exemplo de utilização: Netflix

O projeto enquadra-se no âmbito e complexidade adequada às competências adquiridas no curso:

• Autonomia e capacidade de definir objetivos;

• Capacidade de modelação de problemas;

• Saber elaborar relatórios de análise de soluções;

• Gestão de tempo, cumprimento de prazos;

O projeto realizado obedeceu as seguintes condições:

• Ter um orientador docente da Unidade Técnico-Científica de informática da ESTG do Instituto Politécnico da Guarda;

• Ter um plano de desenvolvimento aprovado pelo docente.

1.1 Motivação

A principal motivação para o desenvolvimento deste projeto é a possibilidade de aprender mais sobre inteligência artificial, mais especificamente *machine learning* (máquina aprendiz), aprender *python* com *klein*, aperfeiçoar conhecimentos de matemática, obter competências nas tecnologias *cloud* (com *amazon aws*) e aprender *mahout* com *spark* e *hadoop* de forma a desenvolver a aplicação de demonstração.

A aplicação serve para demonstrar, em ambiente de testes, o que é possível fazer com *machine learning* (máquina aprendiz).

1.2 Solução

A solução consiste na elaboração de uma aplicação desenvolvida em *python* com a *framework klein* e a biblioteca sqlite como base de dados, utilizando PyCharm para edição de código e *fossil* para gestão de versões. Para a sugestão de conteúdos foi usado EMR da *amazon aws*.

1.3 Contribuição

A contribuição principal deste projeto é a pesquisa, desenvolvimento, implementação e teste de um website para efeitos de demonstração.

1.4 Definição do problema

Demonstrar sugestão de conteúdos em funcionamento, para isso vai ser desenvolvida uma aplicação simples que permita pontear filmes com vários utilizadores, ver histórico, pesquisar por filmes, e executar a sugestão de conteúdos.

1.5 Objetivos previstos

Os objetivos previstos são:

• Conhecer os principais algoritmos de recomendação automática de conteúdos;

• Saber escolher o algoritmo mais apropriado segundo diversos casos de uso;

• Implementar e otimizar um sistema de recomendação de conteúdos usando serviço *Machine Learning* da *Amazon Web Services*;

• Codificar diversas aplicações de *Machine Learning*;

• Integração do sistema e/ou aplicações com o *software ardina.customer* baseado em Salesforce.

2. Metodologia e resultados esperados

2.1 Metodologia

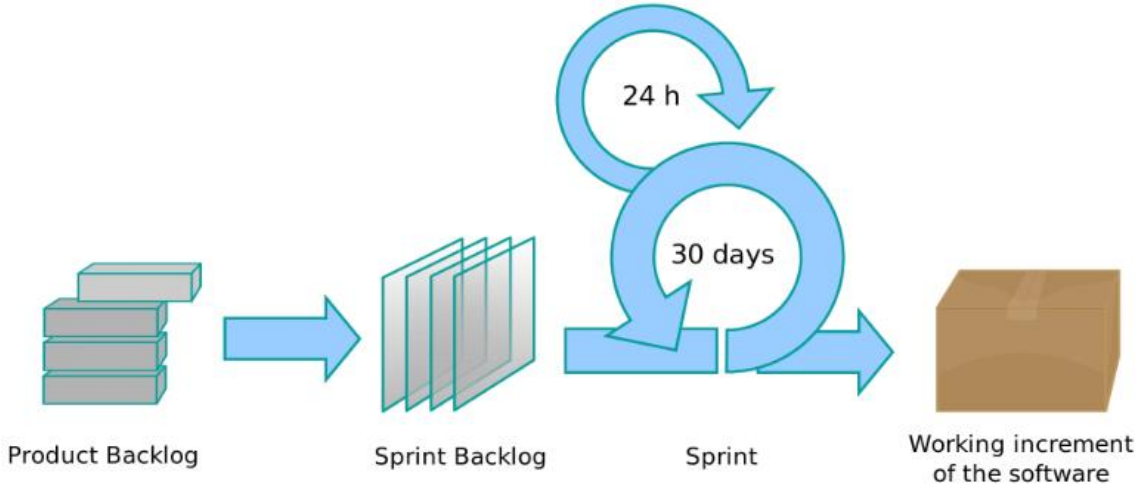


Figura 1 Metodologia Scrum

A metodologia usada foi o scrum, uma metodologia ágil com o defenição "a flexibilidade e integridade da equipa de desenvolvimento, aonde a equipa trabalha como uma unidade para chegar a um objetivo comum"[9].

O "*product Backlog*" é tudo o que ainda falta ser elaborado.

O *sprint* é um bocado de trabalho a ser elaborado num período de tempo, normalmente 1 semana a 30 dias.

Durante o sprint, todos os dias (normalmente) é debatido o que foi feito, e repartido o trabalho para o resto do dia. No fim de cada sprint o bocado funcional do programa é adicionado ou pacote já feito.

Visto que o projeto foi feito por uma só pessoa, os tempos de sprint foram variando muito.

2.2 Descrição das tarefas

As principais tarefas foram:

• Tarefa 1 – Estudo sobre *machine learning*;

• Tarefa 2 – Estudo sobre EMR da *amazon AWS* e *mahout*;

• Tarefa 3 – Desenho da base de dados e interfaces;

• Tarefa 4 – Implementação e testes da aplicação;

• Tarefa 5 – Elaboração do relatório.

O agendamento das tarefas é apresentado na Figura 2.

Tarefa	Nome	Data inicio	Data Fim
1	Estudo sobre machine learning, data mining e sugestão de conteúdos	25-05-2015	30-06-2015
2	Estudo sobre EMR da amazon aws e mahout	01-07-2015	13-07-2015
3	Criar a base de dados, definir e desenhar as páginas web	16-07-2015	17-07-2015
4	Criar a aplicação e testes	20-07-2015	30-07-2015
5	Elaboração do relatório	03-08-2015	04-12-2015

Figura 2 Tarefas

O respetivo mapa de Gantt é representado na figura 3.

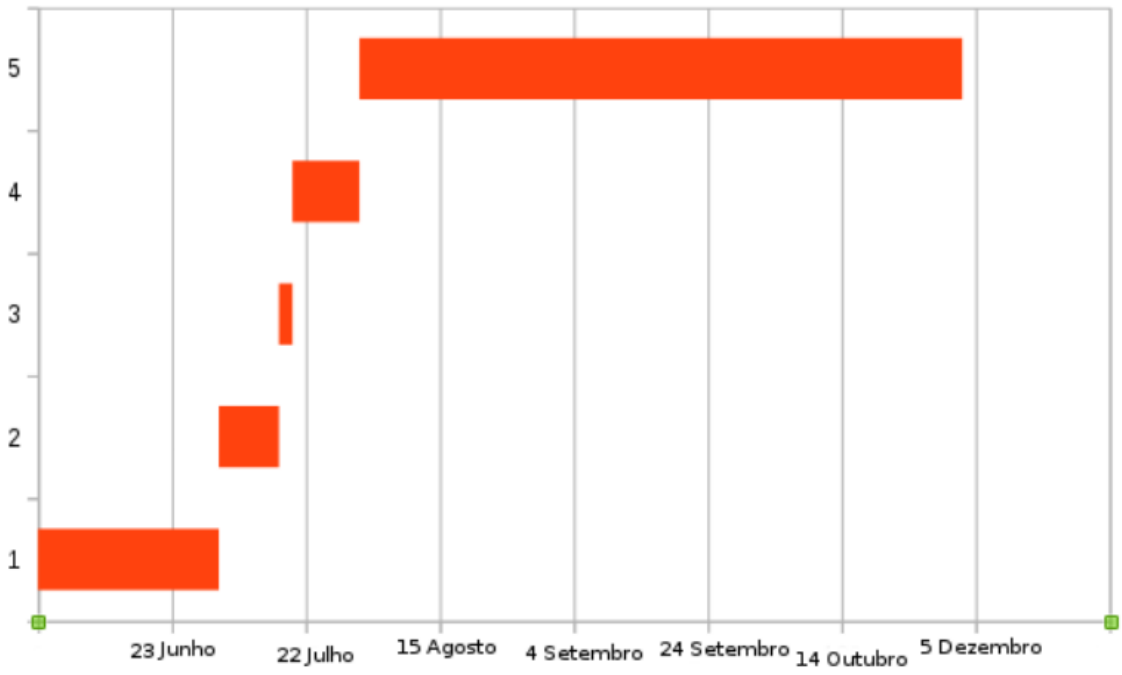


Figura 3 Mapa de Gantt

2.2 Resultados esperados

No final do projeto espera-se que:

• o autor e a Dom Digital tenham aprendido como funciona a sugestão de conteúdos;

• se implemente várias aplicações de *machine learning*;

• se integre um sistema de sugestão de conteúdos no *ardina.customer*.

3. Tecnologias utilizadas

3.1 Tecnologias Web

3.1.1 HTML

HTML (abreviação para a expressão inglesa *HyperText Markup Language*, quesignifica Linguagem de Marcação de Hipertexto) é uma linguagem de marcação utilizada para produzir páginas web. Documentos HTML podem ser interpretados por navegadores. HTML é o bloco básico de qualquer página web. Ela determina a formatação e conteúdo, mas não a sua funcionalidade. Também é usado para definir a estrutura da página, no entanto, tem limitações[2].

3.1.2 CSS

CSS (abreviação para a expressão inglesa *Cascading Style Sheets*). É uma linguagem que define estilos das páginas web: controla fontes, cores, margens, linhas, alturas, imagens de fundo, entre outras.

O HTML pode ser usado para definir a estrutura da página, no entanto o CSS é mais preciso e sofisticado. É suportado em todos os navegadores e é usado para formatar conteúdos estruturados, enquanto o HTML é usado para estruturar conteúdos[3].

3.1.3 Python

Python é uma linguagem interpretada e de código aberto. É interpretada, ou seja : Só no momento de execução é traduzida para código-máquina. É legível e flexível por

desenho, pelo que normalmente demora menos tempo a implementar e manter um programa[4].

Foi escolhida pelo autor, pela sua vontade de aprender, e pela simplicidade e flexibilidade da própria.

3.1.4 SQLite

SQLite é uma base de dados simples e rápida, que não requer configuração para estar pronta a funcionar. Assegura a Integridade dos dados, é usada em aplicações móveis e sites com pouca ou média popularidade. Não foi, no entanto, desenhada para suportar aplicações de grande volume[5].

Foi escolhida por dispensar configuração, e por se enquadrar nas necessidades previstas no projeto.

3.1.5 Klein

Klein é uma pequena *framework* desenvolvida para simplificar a criação de webservices. É muito simples de manusear[6].

3.1.6 Hadoop

O *Hadoop* não foi usado diretamente no projeto. Devemos, no entanto, notar que é uma tecnologia importante para a distribuição de trabalho em larga escala. Foi usado com spark e mahout.

3.1.7 Spark

O *Spark* é uma *framework* que pode funcionar cooperativa-mente com o *hadoop*. Neste caso, foi essa a escolha. E pode acelerar as pesquisas por 100 vezes em várias circunstancias[7].

3.1.8 Mahout

Mahout é uma biblioteca e ou *framework* para criar aplicações de máquinaaprendiz (*machine learning*) de forma facilmente escalável e em pouco tempo[8].

3.2 Tecnologias alternativas

3.2.1 Asp .Net

Asp dot Net ou simplesmente Asp Net, é uma tecnologia muito automática, facilita imenso o desenvolvimento de uma aplicação web. No entanto é obrigatório utilizar o sistema operativo windows, que não foi usado. Mas, tem um grande problema, quando algo para de funcionar, pode-se facilmente gastar um dia inteiro para resolver, e quando resolvido não se sabe o que foi feito, pelo que a aprendizagem desta tecnologia é lenta. E já caiu em desuso.

3.2.2 JSP

O JSP (*JavaServer Pages*) é uma tecnologia criada pela *Sun Microsystems*, simples de usar, mas menos legível que o python, funciona em todos os sistemas operativos que o java EE corre.

3.3 Ferramentas utilizadas

3.3.2 PyCharm

Um editor de texto e ambiente de desenvolvimento integrado muito completo que facilita o desenvolvimento e teste.

3.3.1 SSH

SSH (abreviação para *Secure Shell*) é um protocolo e uma interface por linha de comandos para aceder a um servidor remotamente com segurança.

Foi usado para aceder aos servidores da *amazon aws*.

4 Implementação da solução

4.1 Implementação da base de dados

4.1.1 Modelo relacional

Não foram usados utilizadores no modelo relacional desenvolvido. A fonte de dados não refere utilizador destino, mas apenas identificador único. O modelo relacional não precisa de utilizadores, visto que a fonte de dados não diz quem são os utilizadores e também porque essa informação não é usada na própria sugestão.

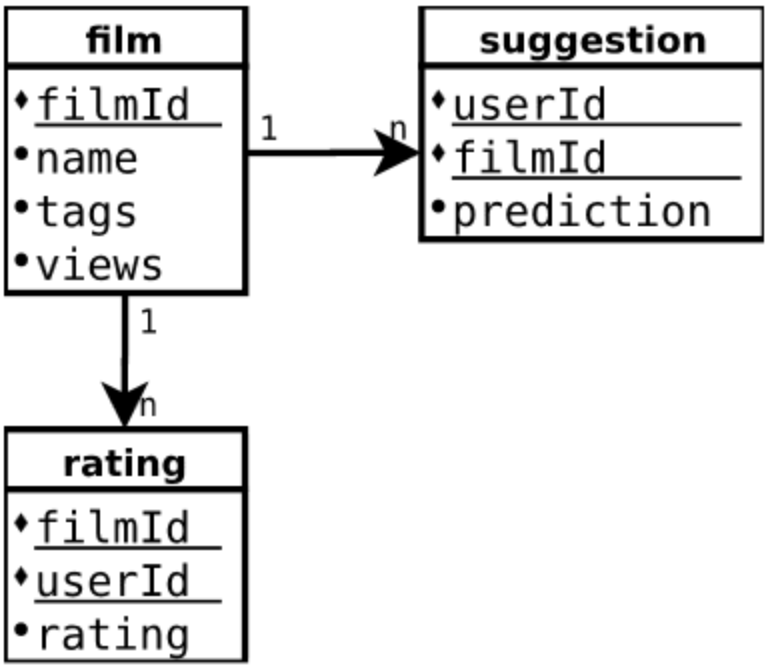


Figura 4 Modelo relacional

4.1.2 Descrição das tabelas

Descrevo, de seguida, as tabelas, com o respectivo dicionário de dados.

4.1.2.1 Tabela *Film*

Na Tabela "1: Estrutura da tabela Film" é definido o nome do filme. O campo "tags", é um campo que contém todas as tags do filme. Não foi separado por questões de simplicidade.

Campo	Tipo	*Tamanho*	*Validações*	Descrição
filmId (P)	Inteiro	20	Maior que 0	Número sequencial que identifica inequivocamente cada filme.
name	String	255		Nome do filme.
tags	String	255		Géneros do filme, separados por virgulas.
views	Inteiro	20	Maior que 0	Número total de visualizações do filme

Tabela 1 Estrutura da tabela Film

4.1.2.2 Tabela *suggestion*

Na tabela "2: Estrutura da tabela suggestion", é definida o utilizador a quempretence a sugestão, o filme sugerido, e a previsão da máquina.

Campo	Tipo	Tamanho	Validações	Descrição
userId (P)	Inteiro	20	Maior que 0	Número sequencial que identifica inequivocamente cada utilizador.
filmId (PF)	Inteiro	20	Maior que 0	Número sequencial que identifica inequivocamente cada filme.
prediction	real	5	Entre 0 e 5	Previsão da pontuação do utilizador.

Tabela 2 Estrutura da tabela suggestion

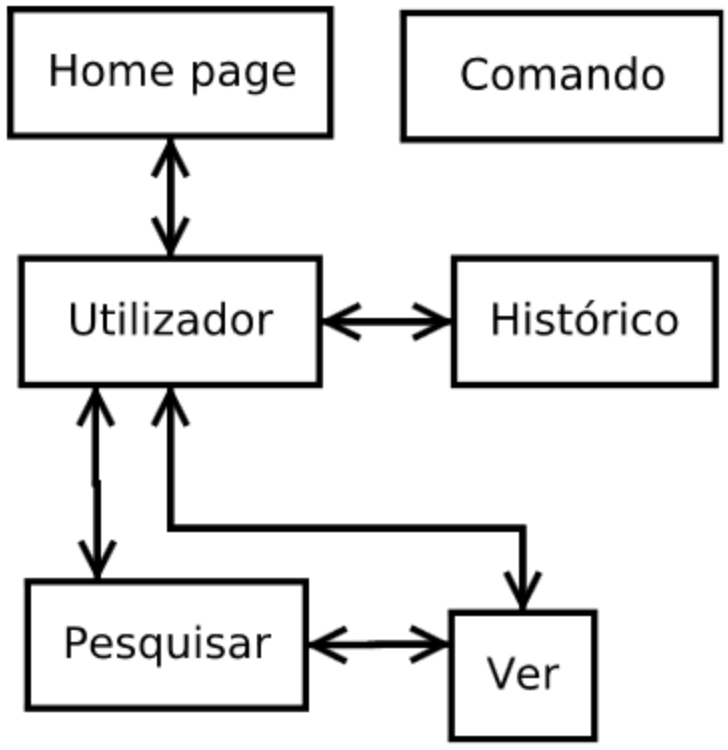
4.1.2.3 Tabela *rating*

Na tabela "3: Estrutura da tabela *rating*" é definida a pontuação de cada utilizador para cada filme.

Campo	Tipo	Tamanho	Validações	Descrição
filmId (PF)	Inteiro	20	Maior que 0	Número sequencial que identifica inequívocamente cada filme.
userId (PF)	Inteiro	20	Maior que 0	Número sequencial que identifica inequívocamente cada utilizador.
rating	real	2	Entre 0 a 5	Pontuação dada pelo utilizador.

Tabela 3 Estrutura da tabela rating

4.2 Mapa do site



4.2.1 *Home page*

O "*Home page*" é onde se seleciona o identificador (id) de utilizador a utilizar.

4.2.2 Utilizador

Na página "utilizador" podemos ver as sugestões, os filmes das sugestões, iniciar uma pesquisa e ver o histórico.

4.2.3 Pesquisar

Na página "pesquisar" podemos pesquisar por:

-id do filme

-por nome/tags

No segundo caso, se o programa não encontrar o número esperado de resultados, que pode ser configurável, mas assume o valor 10 por defeito, pesquisa os que faltam por tags.

4.2.4 Ver

Na página 'Ver' podemos pontuar o filme e submeter.

4.2.5 Histórico

Na página do histórico podemos ver o histórico dos filmes pontuados.

4.2.6 Comando

A página comando destina-se a facilitar a execução de testes de demonstração. Num ambiente de produção estes testes seriam automáticos. Nesta página, podemos iniciar o processo de sugestão de conteúdos, neste caso filmes, para todos os utilizadores, assim como visualizar o respetivo progresso.

5 Conclusões e trabalho futuro

5.1 Conclusão

Durante a elaboração do projeto e relatório foram surgindo contra tempos que atrasaram bastante a sua elaboração. O facto de no inicio o autor não ter conhecimentos sobre *machine learning* e, especialmente sugestão de conteúdos, e no fim, o seu atraso e objetivos não compridos. Foi muito gratificante aprender python com klein, tecnologias que não são ensinadas no curso de Engenharia Informática da Guarda.

Inicialmente estava planeado o desenvolvimento e otimização de um algoritmo de sugestão de conteúdos, para o fazer é preciso ter conhecimentos de *machine learning*, *data mining* e bom conhecimento de álgebra linear e geometria analítica. Pelo que no projeto foi usado um algoritmo já desenvolvido. O algoritmo de *machine learning* usado foi *Clustering*, para fazer o calculo de similaridade entre os utilizadores foi usado similaridade por cosseno, que não é dos melhores, mas era o melhor já existente no *mahout*. A linguagem escolhida devia ter sido o java, o *mahout* tem uma biblioteca de funcionalidades extensa, para o maior controlo e aproveitamento da própria. No entanto existe pouco documentação, o que dificultou as escolhas.

Durante o projeto foram encontradas vários problemas. A alteração de ermissões do hadoop, nos serviços da amazon, que obrigou a alterar a maneira como as sugestões são criadas, lidas e guardadas entre outros.

Não foi possível a integração do sistema de sugestões com o software ardina.customer por tempo insuficiente.

5.2 Trabalho futuro

Os objetivos previstos para o projeto não foram compridos, mas o objetivo principal foi: aprender, errar, corrigir erros e sobre tudo, resolver problemas. Existe muito trabalho por fazer, criar um site para testes, ou mesmo alterar um já em produção, de modo a automaticamente requisitar os recursos da *amazon* AWS, executar a sugestão e devolver ao site que por sua vez vai utilizar as sugestões, também é preciso saber aprecisão dos algoritmos, melhorar na arte de *machine learning*.Estudar e criar web services com *twisted*, uma *framework* que é usado pela *framework* usada no projeto, o *klein*. Esta *framework* é mais complicada de manejar, no entanto é muito mais poderosa e com muito mais funcionalidades.

Bibliografia

[1] Wikipedia. Recommender system approaches
https://en.wikipedia.org/wiki/Recommender_system#Approaches.

[2] Wikiversity. HTML Introduction
https://en.wikiversity.org/wiki/HTML/Introduction.

[3] HTML.NET. O que é css.
http://pt-br.html.net/tutorials/css/lesson1.php.

[4] Margaret Rouse. Python definition
http://searchenterpriselinux.techtarget.com/definition/Python.

[5] Sqlite. About sqlite
https://www.sqlite.org/about.html.

[6] Twisted team, Klein, a Web Micro-Framework
https://github.com/twisted/klein.

[7] Wikipedia. Apache Spark
https://en.wikipedia.org/wiki/Apache_Spark.

[8] Apache. What is Apache Mahout?
http://mahout.apache.org/.

[9] Wikipidia, Scrum (software development)
https://en.wikipedia.org/wiki/Scrum_%28software_development%29.