Machine Learning para Sistemas de Recomendação

Gabriel Moreira • Lead Data Scientist • gabrielpm@ciandt.com

sli.do: #ciandttechsummit

Gabriel Moreira

@gspmoreira

- Lead Data Scientist na CI&T
- Aluno de Doutorado no ITA - pesquisa com ênfase em *Deep Learning* e Sistemas de Recomendação
- Mais de quinze anos de experiência em desenvolvimento de software

https://about.me/gspmoreira

A vida é curta!

Recomendações sociais

Recomendações por interação

“Muitas vezes, as pessoas não sabem o que elas querem até que você mostre a elas.”

Steve Jobs
Apple

“Estamos saindo da Era da Informação e entrando na Era da Recomendação.”

Chris Anderson
Autor do Livro “A cauda longa”

Sistemas de recomendação são responsáveis por...

amazon.com

70% da home page da Amazon é dedicada a recomendações

NETFLIX

⅔ dos filmes assistidos

Google news

Recomendações geram 38% mais clicks

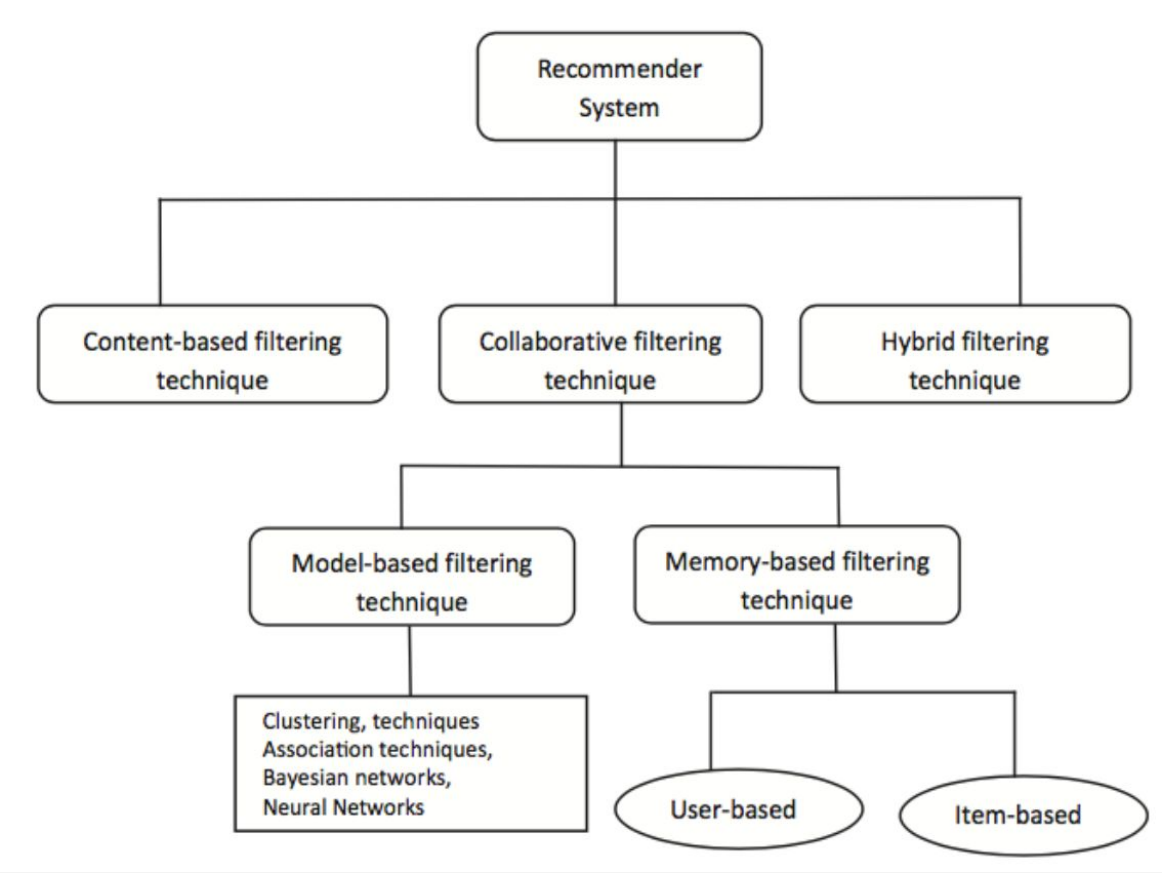
O que mais eu posso recomendar?

O que um sistema de recomendação pode fazer?

1 - Recomendação
Dado um usuário, produza uma lista ordenada de itens relevantes.

2 - Predição
Dado um item, calcule sua relevância
para cada usuário

Abordagens de recomendação tradicionais

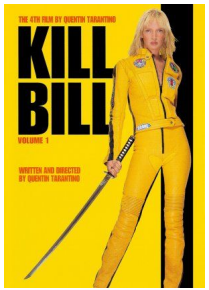


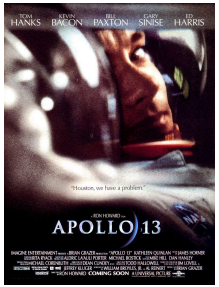
Fonte: "Recommendation systems: Principles, methods and evaluation"

Filtragem Colaborativa

Filtragem Colaborativa baseada em Memória

Similaridade entre Usuários *(k-nearest neighbors)*





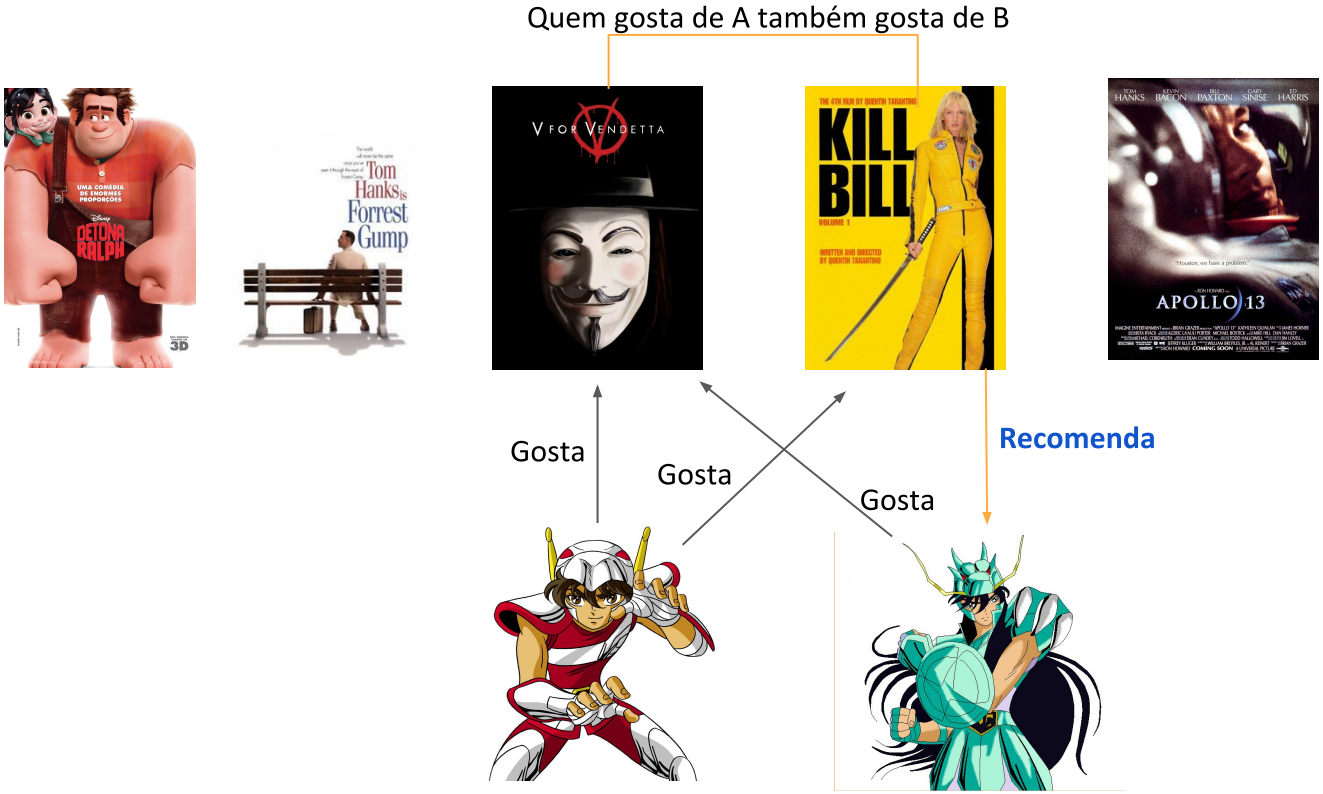
Gosta

Recomenda

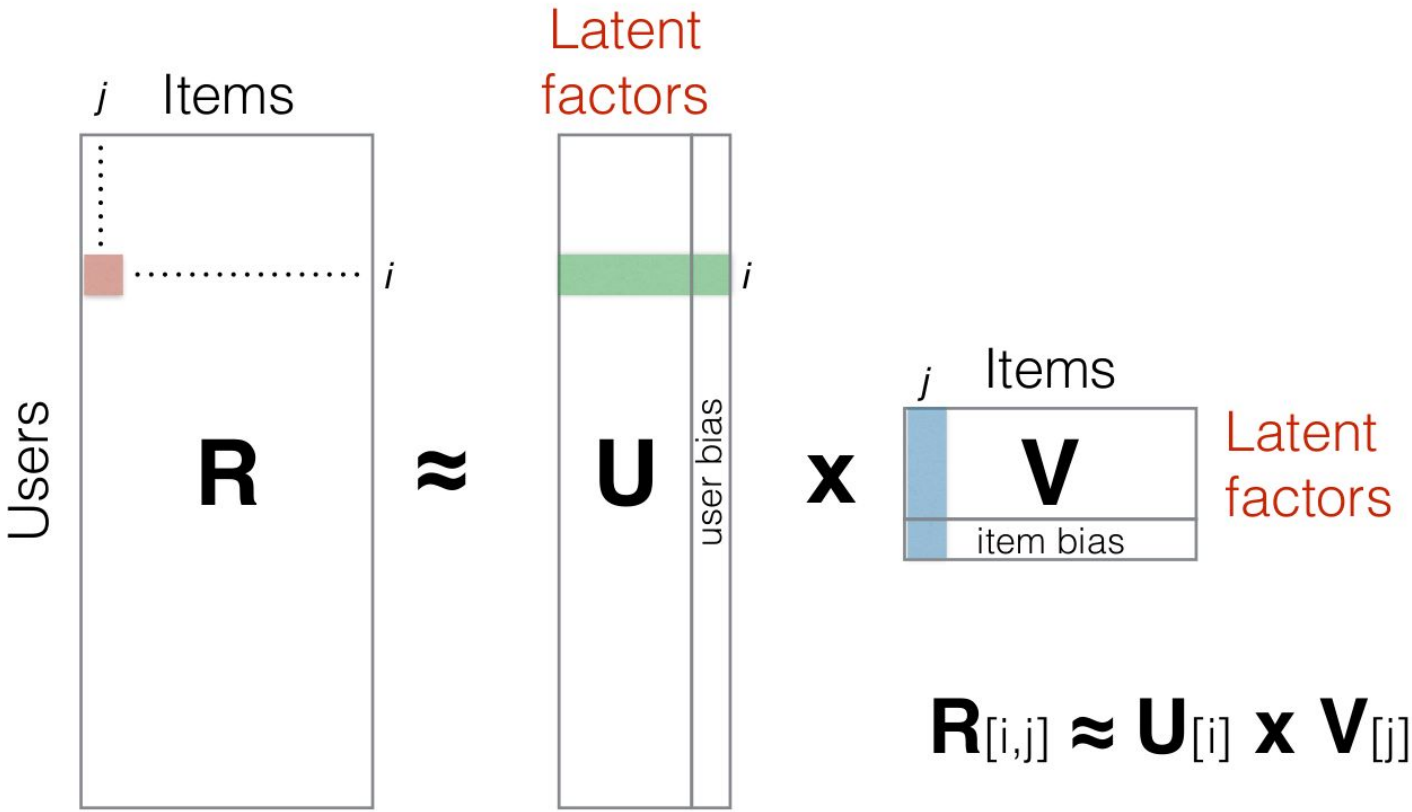
Interesses similares

Filtragem Colaborativa baseada em Memória

Similaridade entre Itens *(k-nearest neighbors)*



Filtragem Colaborativa baseada em Fatoração de Matrizes



$$R_{[i,j]} \approx U_{[i]} \times V_{[j]}$$

Filtragem Colaborativa

Vantagens

- Funciona para qualquer tipo de item, pois ignora atributos

Desvantagens

- Precisa de um mínimo de usuários e itens para calcular similaridades
- Não pode recomendar itens que ainda não foram consumidos pelos usuários

Frameworks de Filtragem Colaborativa

Java

Python / Scala

Python

Java

.NET

Exemplo de Filtragem Colaborativa usando Mahout

```
User,Item,Rating1,
15,4.0
1,16,5.0
1,17,1.0
1,18,5.0
2,10,1.0
2,11,2.0
2,15,5.0
2,16,4.5
2,17,1.0
2,18,5.0
3,11,2.5
```

input.csv

```
// Loads user-item ratings
1 DataModel model = new FileDataModel(new File("input.csv"));
// Defines a similarity metric to compare users (Person's correlation coefficient)
2 UserSimilarity similarity = new PearsonCorrelationSimilarity(model);
// Threshold the minimum similarity to consider two users similar
3 UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity,
model);
// Create a User-Based Collaborative Filtering recommender
4 UserBasedRecommender recommender = new
GenericUserBasedRecommender(model, neighborhood, similarity);
// Return the top 3 recommendations for userId=2
5 List recommendations = recommender.recommend(2, 3);
```

https://mahout.apache.org/users/recommender/userbased-5-minutes.html

Filtragem baseada em Conteúdo

Filtragem baseada em Conteúdo

Filtragem Baseada em Conteúdo

Vantagens

- Não depende de outros usuários
- Podem recomendar itens novos
- Recomendações são facilmente explicáveis

Desvantagens

- Super-especialização
- Extrair atributos de áudio, filmes e imagens não é trivial

Vetorização de Texto

Representa cada documento como um vetor de *features*. Cada posição no vetor representa a relevância de uma palavra (token) no documento.

- *BoW* (*Bag of words*)
- *TF-IDF* (*Term Frequency - Inverse Document Frequency*)

	D1	D2	D3	D4
linux	3	4	1	0
modem	4	3	0	1
the	3	4	4	3
clutch	0	1	4	3
steering	2	0	3	3
petrol	0	1	3	4

Matriz Documento-Token - *Bag of Words*

Vetorização de Texto

TF-IDF - Termos mais “relevantes” em um documento consistem em termos frequentes em um documento que são raros em outros documentos.

f_{ij} = frequency of term (feature) ***i*** in doc (item) ***j***

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

n_i = number of docs that mention term ***i***

N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Exemplo de TF-IDF com scikit-learn

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_df=0.5, max_features=1000,
                             min_df=2, stop_words='portuguese')
tfidf_corpus = vectorizer.fit_transform(text_corpus)
```

tokens

documentos	face	pessoa	guia	faca	gato	peixe	durma	micro	festa	cento	...
	0	1	2	3	4	5	6	7	8	9	
D1		0.05					0.25				
D2	0.02			0.32					0.45		
...											

Exemplo de uma matriz esparsa de TF-IDF

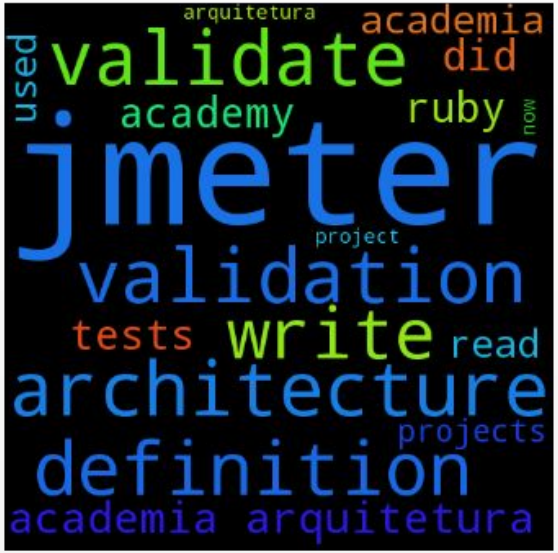
Vetorização de Texto

Exemplo:

"Did you ever wonder how great it would be if you could **write** your **jmeter tests** in **ruby** ? This projects aims to do so. If you use it on your project just let me now. On the Architecture Academy you can read how jmeter can be used to **validate** your Architecture. **definition** | architecture **validation** | **academia** de **arquitetura"**

Termos relevantes (TF-IDF)

Termos (*unigrams* e *bigrams*)	**peso**
jmeter	0.466
architecture	0.380
validate	0.243
validation	0.242
definition	0.239
write	0.225
academia arquitetura	0.218
academy	0.216
ruby	0.213
tests	0.209



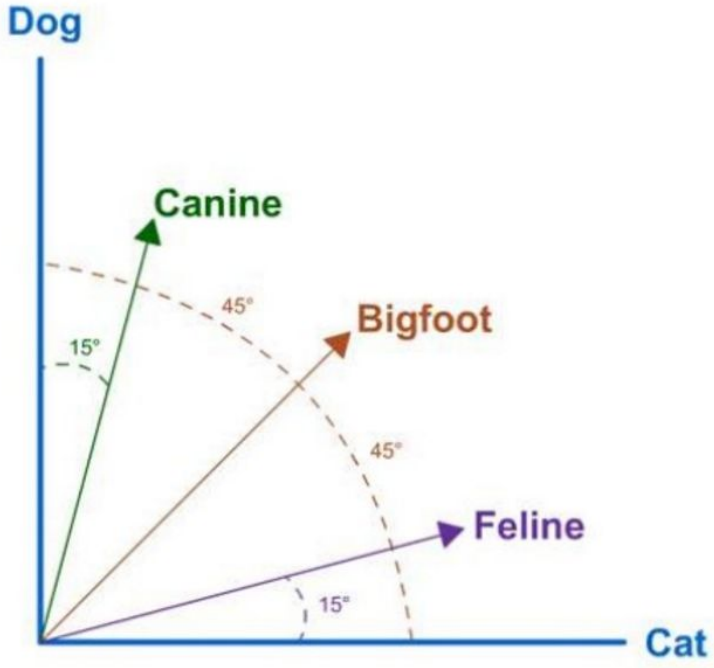
Vetorização de Texto

Visualização da média dos vetores TF-IDF dos posts no Google+ for Work (CI&T)

Mais detalhes sobre social e textual analytics em http://bit.ly/python4ds_nb

Cosine Similarity

Métrica de similaridade calculada como o cosseno do ângulo entre os vetores




```
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(tfidf_matrix[0:1], tfidf_matrix)
```

Exemplo com scikit-learn

$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

Similaridade entre pessoas

Exemplo de tokens relevantes de uma pessoa e similaridade com outras pessoas

Filtragem Híbrida

Sistemas de Recomendação Híbridos

Algumas abordagens...

Composto
Agrega recomendações de uma cadeia de algoritmos.

Ponderado
Cada algoritmo tem um peso e a ordenação final das recomendações é definida por uma média ponderada

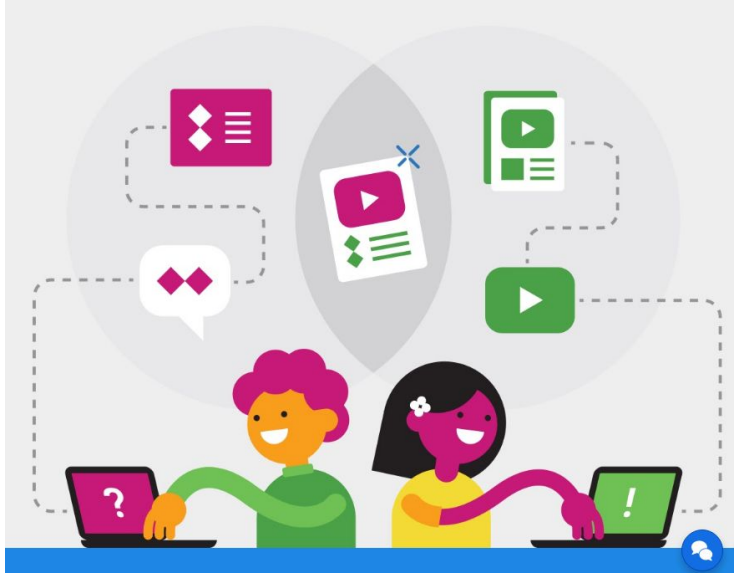
Integrados
Filtragem Colaborativa e Filtragem Baseada em Conteúdo integradas em uma função objetivo (loss function). Ex:

- Matrix Factorization + Topic Modeling
- Multi-view Deep neural networks to model user interactions and textual items content (word embeddings)

Case

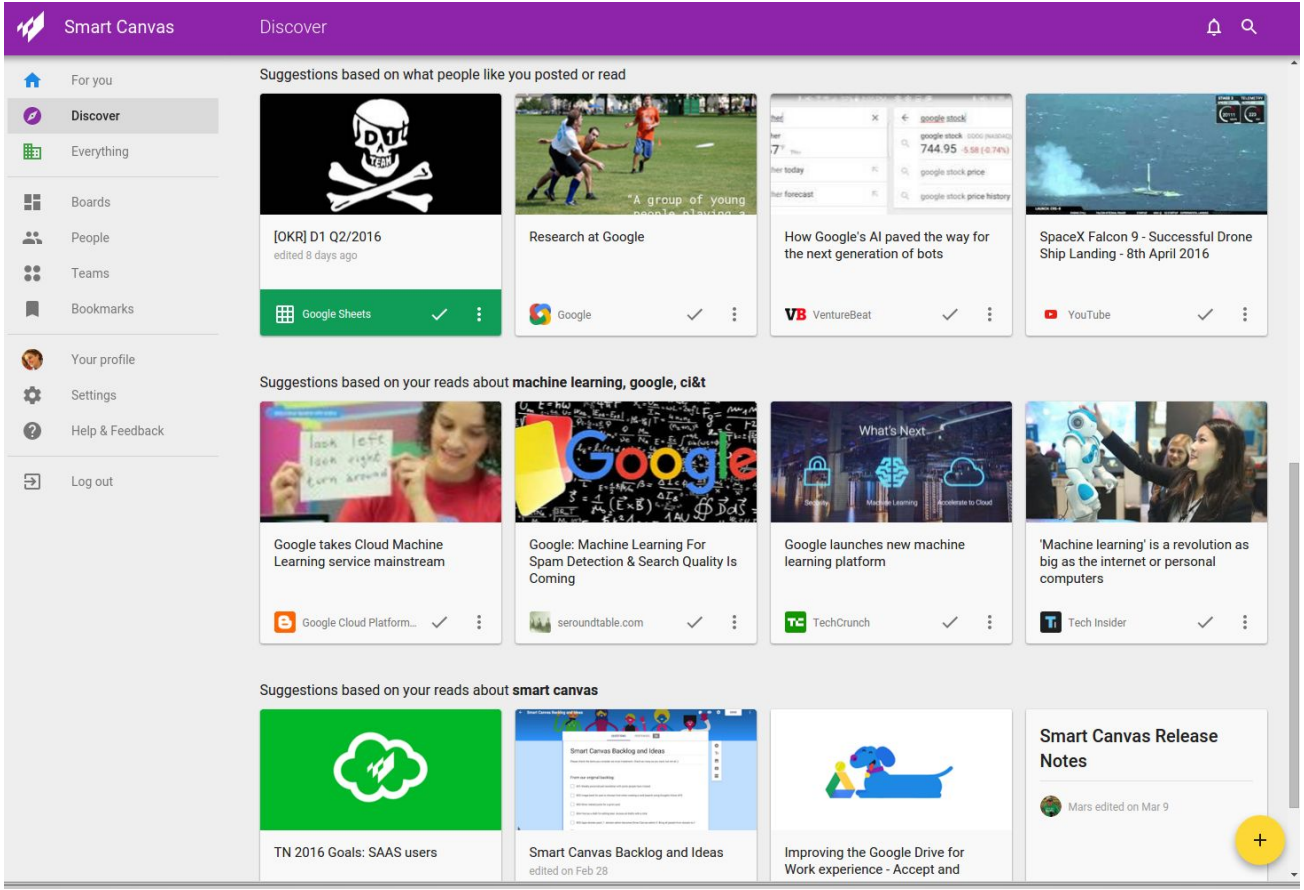
Smart Canvas©

Colaboração Corporativa



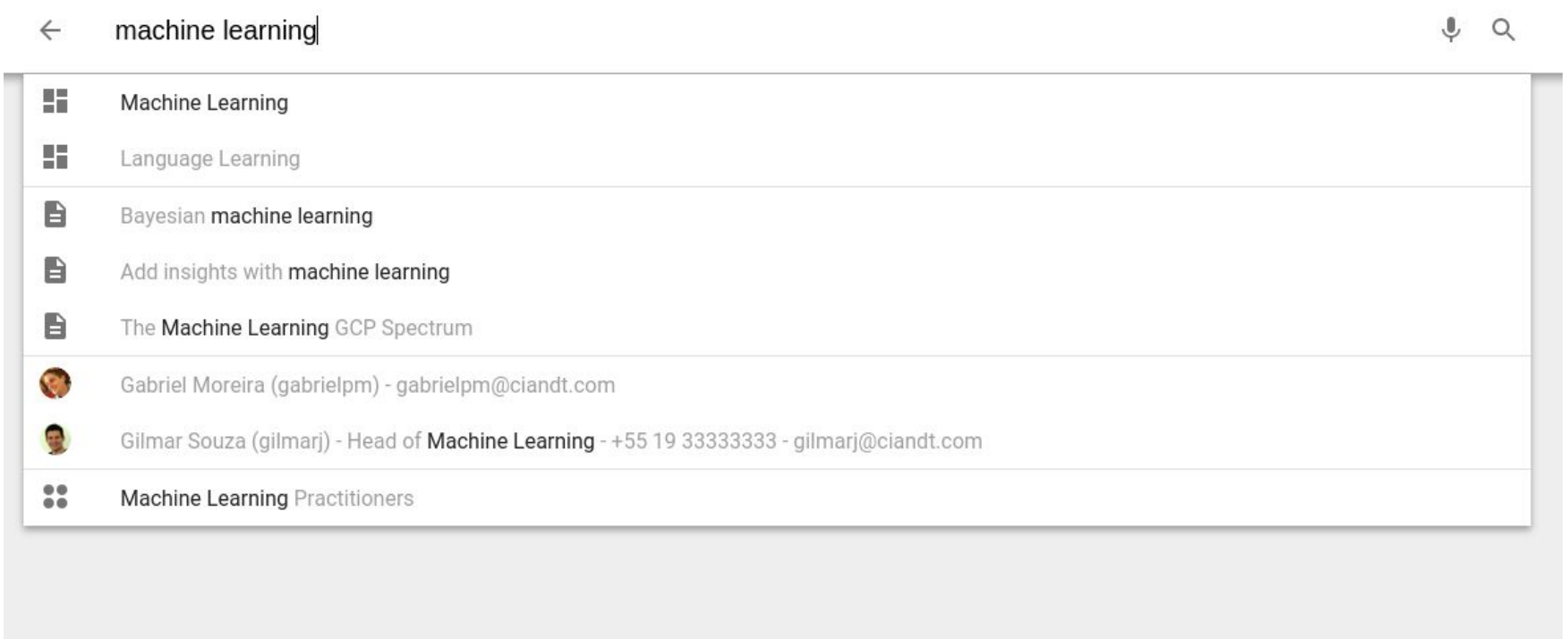
http://www.smartcanvas.com/

Recomendação de conteúdo

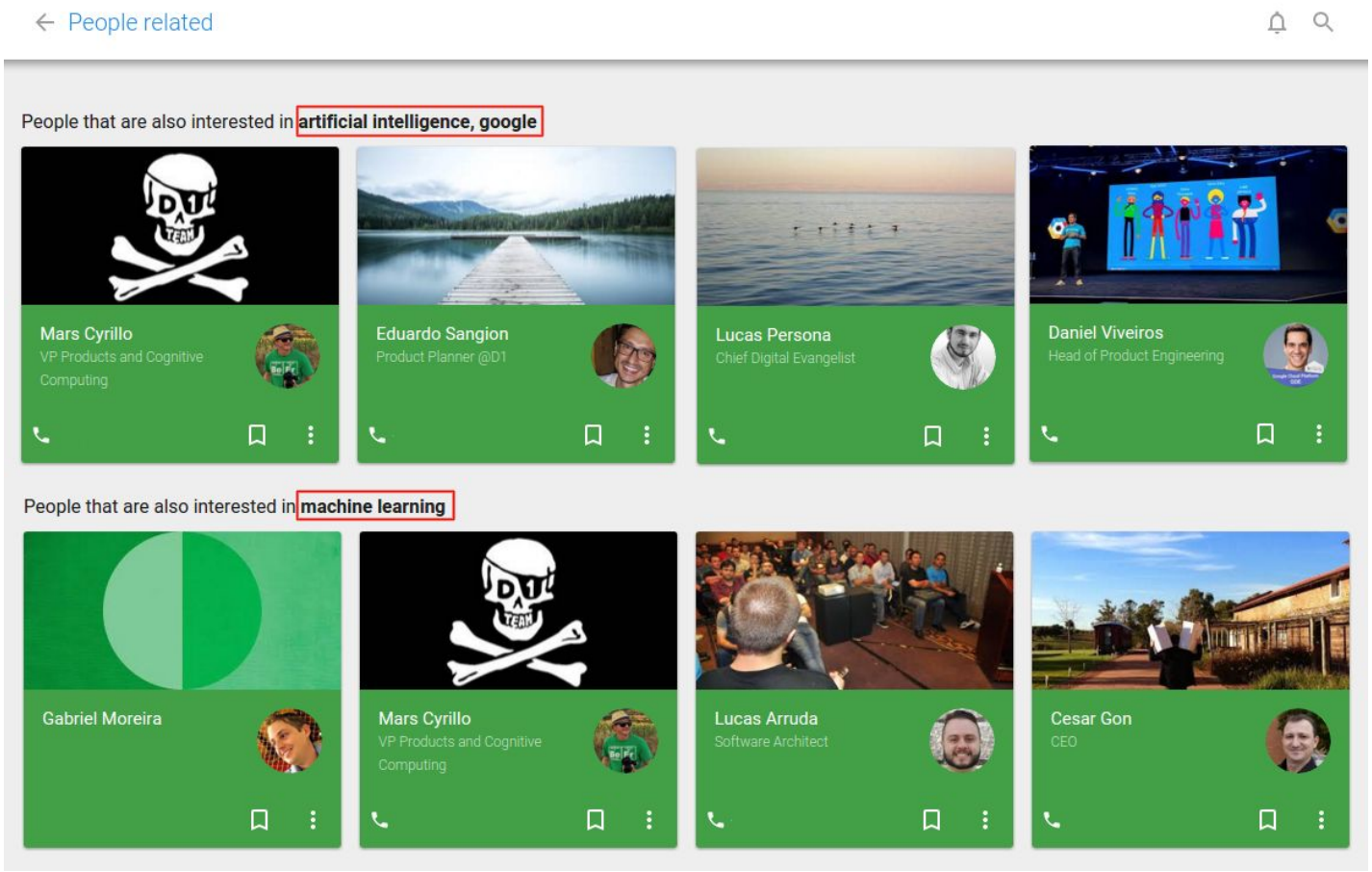


Tópicos de interesse

Busca por pessoas interessadas ou experts

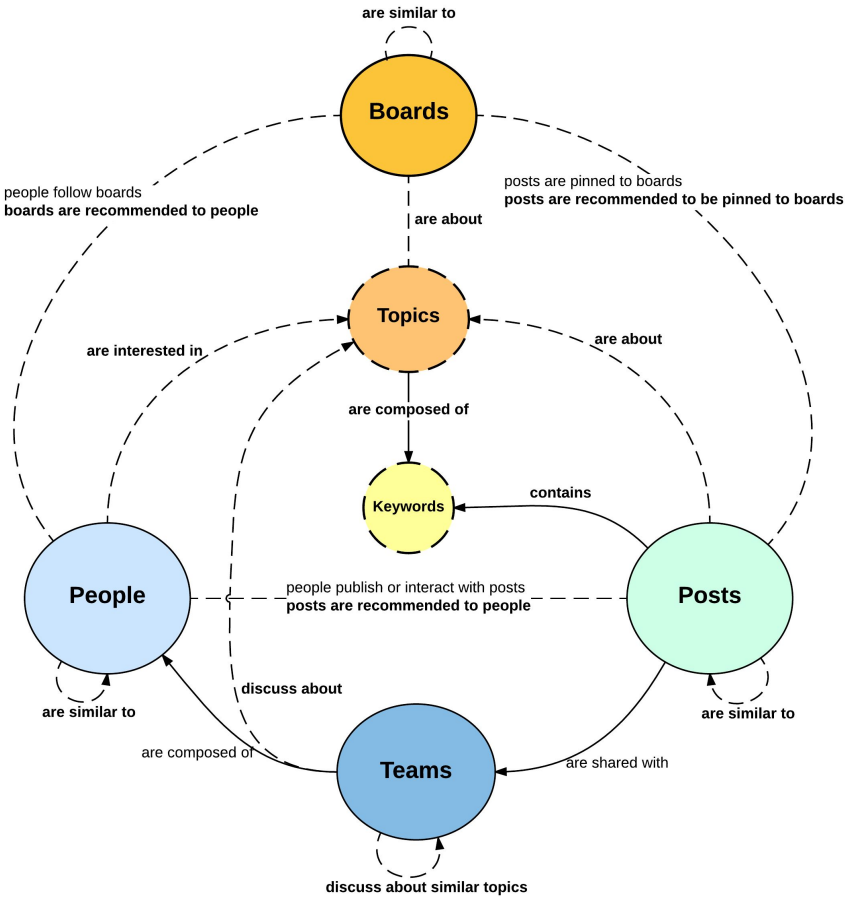


Pessoas similares



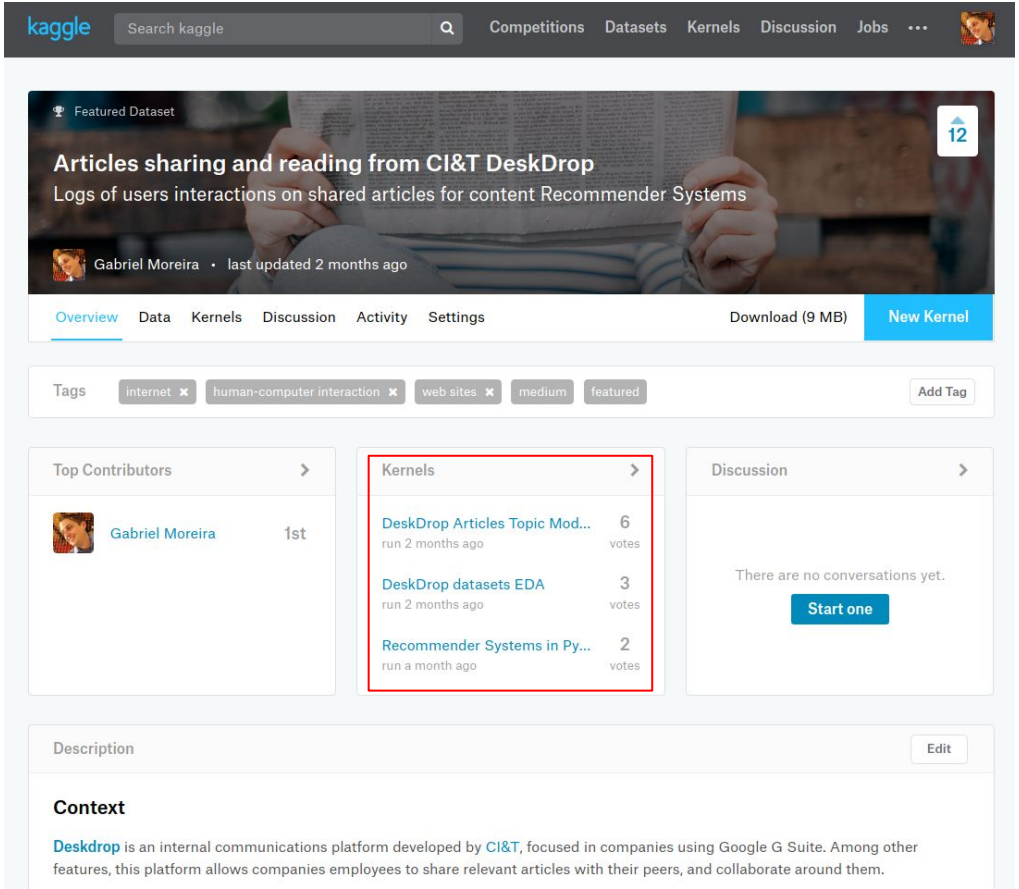
Como funciona?

Modelo de Grafo



Smart Canvas © Graph Model:
Linhas tracejadas e rótulos em negrito são os relacionamentos inferidos por Machine Learning, usando técnicas de Sistemas de Recomendação e Modelagem de Tópicos

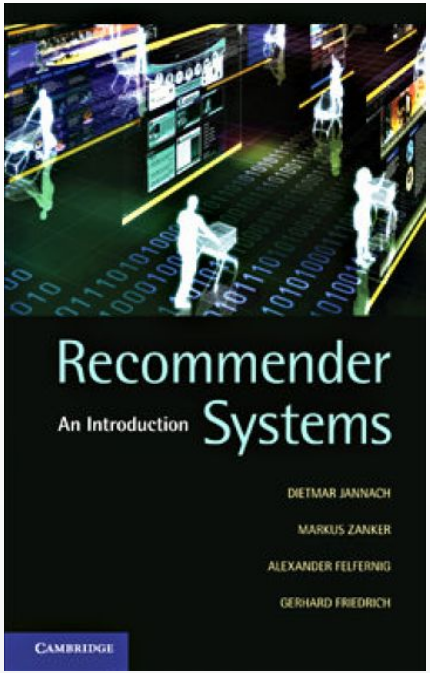
Dataset do CI&T Deskdrop no Kaggle!

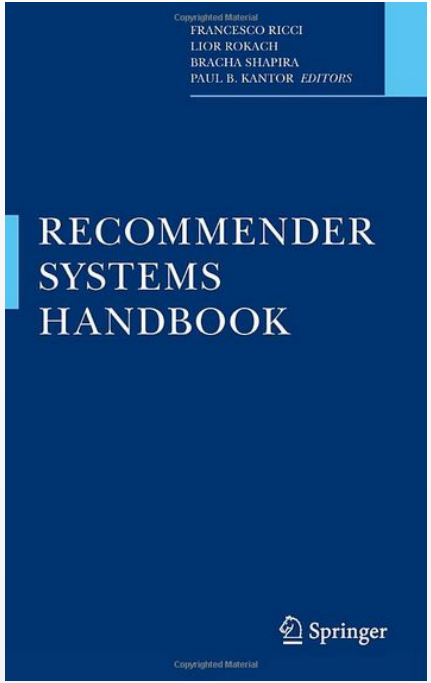


- 12 meses de logs (Mar. 2016 - Feb. 2017)
- ~ 73k interações de usuários
- ~ 3k artigos públicos compartilhados na plataforma

bit.ly/dataset4recsys

Referências





The ACM Conference Series on **Recommender Systems**

recsys.acm.org/

MACHINE
LEARNING
C I & T

Perguntas?

Gabriel Moreira
Lead Data Scientist
gabrielpm@ciandt.com
@gspmoreira

Obrigado!