



# THE DEVELOPER'S CONFERENCE

Trilha – Python

## **Criando sistemas de recomendação com Python**

**José Fernando Tavares**

Booknando Livros



**JOSÉ FERNANDO TAVARES**



JFTavares



fernando@booknando.com.br



booknando.com.br



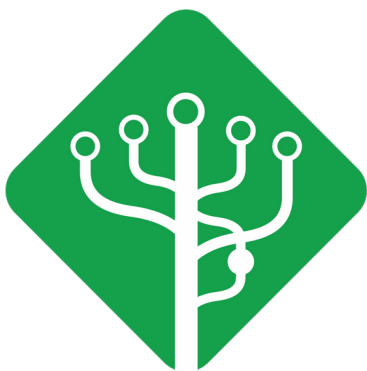
THE  
DEVELOPER'S  
CONFERENCE

- Formação humanista (filosofia e Teologia)
- Especialista em livros digitais
- Consultor de acessibilidade
- Cursando mestrado Ciência da Computação
- Cursando MBA em tecnologia para negócios
- **Apaixonado por Python**

# A força da comunidade



Apoio



DEVPARANÁ



## Python Maringá #1

*Venha participar conosco em  
nossa primeira edição!*

**Data:** 08/08/2019  
**Local:** EVOA Aceleradora  
**Horário:** 19:00 ~ 21:00

Apoio:



DEVPARANÁ



evoa



THE  
DEVELOPER'S  
CONFERENCE





# surprise

A Python scikit for  
recommender systems.

## Home

Documentation

 GitHub page

 Star  Fork

Maintained by Nicolas Hug

Page built with Jekyll and Hyde

# SCIKIT-SURPRISE

## Overview

Surprise is a Python [scikit](#) building and analyzing recommender systems.

Surprise was designed with the following purposes in mind:

- Give users perfect control over their experiments. To this end, a strong emphasis is laid on [documentation](#), which we have tried to make as clear and precise as possible by pointing out every detail of the algorithms.
- Alleviate the pain of [Dataset handling](#). Users can use both *built-in* datasets ([Movielens](#), [Jester](#)), and their own *custom* datasets.
- Provide various ready-to-use [prediction algorithms](#) such as [baseline algorithms](#), [neighborhood methods](#), matrix factorization-based ( [SVD](#), [PMF](#), [SVD++](#), [NMF](#)), and [many others](#). Also, various [similarity measures](#) (cosine, MSD, pearson...) are built-in.
- Make it easy to implement [new algorithm ideas](#).
- Provide tools to [evaluate](#), [analyse](#) and [compare](#) the algorithms performance. Cross-validation procedures can be run very easily using powerful CV iterators (inspired by [scikit-learn](#) excellent tools), as well as [exhaustive search over a set of parameters](#).



# UM MUNDO DE ESCOLHAS



# O que escolher?





# Customization



THE  
DEVELOPER'S  
CONFERENCE



O Santo Graal  
da internet



**amazon**

The Amazon logo features the word "amazon" in a bold, black, lowercase sans-serif font. Below the text is a curved orange arrow that starts under the letter 'a' and points towards the letter 'z'.

THE  
DEVELOPER'S  
CONFERENCE



**NETFLIX**

**Linked in**

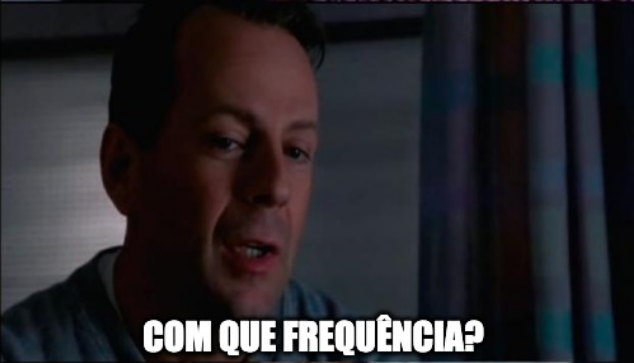
The LinkedIn logo features the word "Linked" in a bold, black, lowercase sans-serif font, followed by the word "in" in a white, lowercase sans-serif font inside a blue square.



THE  
DEVELOPER'S  
CONFERENCE



**EU VEJO SISTEMAS DE RECOMENDAÇÃO...**



**COM QUE FREQUÊNCIA?**



**O TEMPO TODO...**

imgflip.com

# Everywhere

# Modelos tradicionais



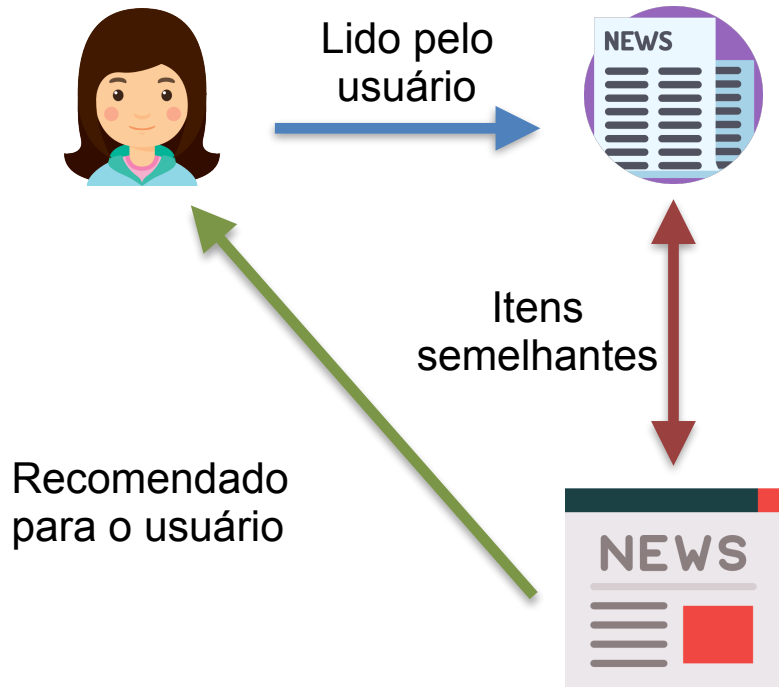
## SISTEMAS NÃO PERSONALIZADOS



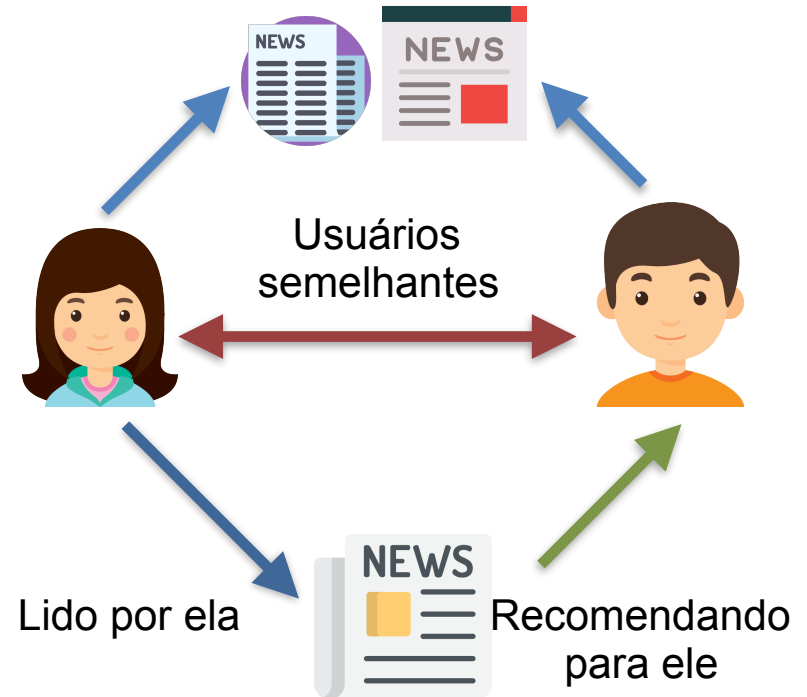
- Listas de mais vendidos
- Listas de melhor avaliados
- Listas de lançamentos
- etc...



## BASEADOS EM CONTEÚDO



## FILTRAGEM COLABORATIVA







## RECOMENDADOR INTERATIVO

Sabemos que pode ser difícil encontrar o jogo perfeito para você, então usamos o poder da aprendizagem de máquina (é o que a galera descolada anda fazendo) para ajudar a explorar a loja Steam.

Para este experimento, **analisamos o seu histórico para gerar uma lista de títulos feita para você**. Priorize títulos conhecidos ou jogos obscuros com o ajuste de popularidade. Caso tenha preferência por lançamentos recentes, temos um ajuste para isso também. Por fim, se estiver atrás de um gênero específico, filtre a lista por marcador.

Para conhecer melhor o recomendador, leia o [artigo no blog](#).

### TEMPO DE JOGO



josefernando.lavares  
1 jogos  
8 horas no total



8 horas  
última sessão 24 de jun de 2014

### SUAS RECOMENDAÇÕES

Priorizar jogos...

POPULARES

DE NICHU

Incluir apenas...

JOGOS ANTIGOS

LANÇAMENTOS

10 anos

Só exibir jogos com o marcador:

No Filter

Excluir jogos com o marcador:

No Filter

Excluir jogos na lista de desejos



### COUNTER-STRIKE: GLOBAL OFFENSIVE

LANÇADO EM 21 DE AGO DE 2012

Estratégia Ação FPS Guerra Cooperativo



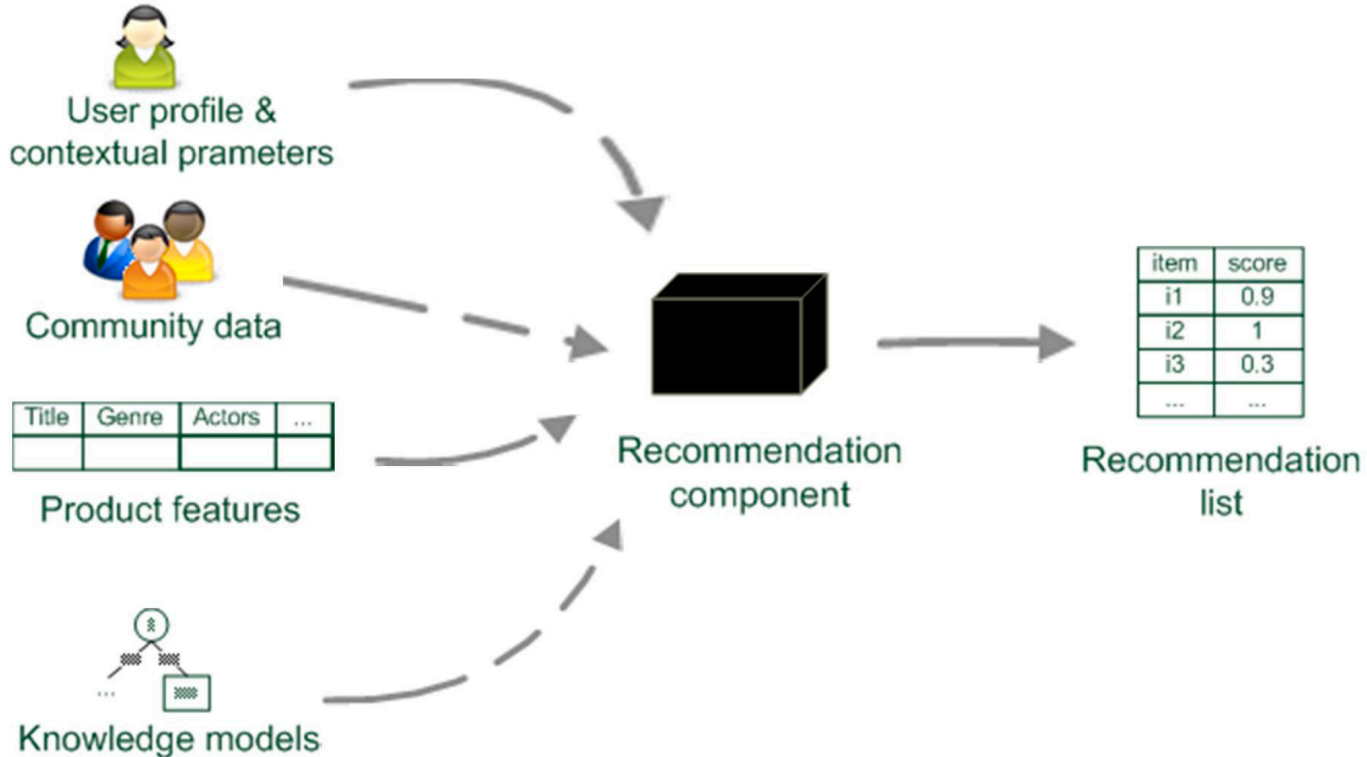
### DOOM 3: BFG EDITION

LANÇADO EM 16 DE OUT DE 2012

# SISTEMAS HÍBRIDOS



THE  
DEVELOPER'S  
CONFERENCE



# Scikit Surprise



Surprise é um add-on SciPy para construir e analisar sistemas de recomendação baseados em filtragem colaborativa

© Copyright 2015, Nicolas Hug.

Comunidade ativa no Github

Licença BSD 3-Clause

# Paradigmas suportados



THE  
DEVELOPER'S  
CONFERENCE



## Filtragem colaborativa

Sistemas de  
Recomendação

AMERICANAS .com

 Spotify

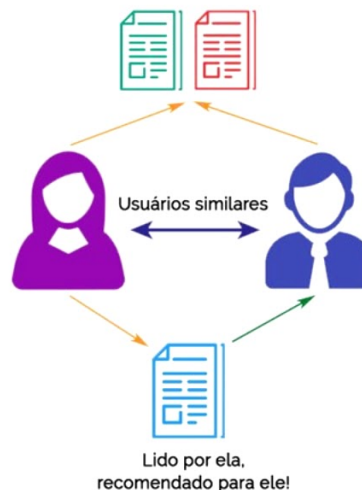
NETFLIX



You Tube 

FILTRAGEM COLABORATIVA

Lido pelos dois usuários







## Documentação muito completa



THE  
DEVELOPER'S  
CONFERENCE

### Matrix Factorization-based algorithms

```
class surprise.prediction_algorithms.matrix_factorization.SVD
```

Bases: `surprise.prediction_algorithms.algo_base.AlgoBase`

The famous SVD algorithm, as popularized by [Simon Funk](#) during the Netflix Prize. When baselines are not used, this is equivalent to Probabilistic Matrix Factorization [\[SM08\]](#) (see [note](#) below).

The prediction  $\hat{r}_{ui}$  is set as:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

If user  $u$  is unknown, then the bias  $b_u$  and the factors  $p_u$  are assumed to be zero. The same applies for item  $i$  with  $b_i$  and  $q_i$ .

For details, see equation (5) from [\[KBV09\]](#). See also [\[RRSK10\]](#), section 5.3.1.

To estimate all the unknown, we minimize the following regularized squared error:

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda (b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$



# Excelente para uso acadêmico ou em âmbitos de estudos dos algoritmos



THE  
DEVELOPER'S  
CONFERENCE

The screenshot shows a Kaggle notebook page. At the top, the Kaggle logo and navigation menu are visible. The notebook title is "Sistemas de Recomendación con Surprise y Lightfm". Below the title, it says "Python notebook using data from multiple data sources" and "401 views · 4mo ago". The main content area has a heading "Sistemas de Recomendación" and a paragraph explaining the goal: "Busca identificar las preferencias de los usuarios a partir de información de sus selecciones, de los items, y de la población. Un sistema de recomendación o motor compara el perfil de usuarios con características de los items y predictivamente intenta identificar la calificación que este le daría. Estas características pueden ser dadas por el mismo item o inferidas implícitamente a través del contexto que dan otros usuarios al mismo." Below this, there are two bullet points: "More than 80 per cent of the TV shows people watch on Netflix are discovered through the platform's recommendation system" and "35% of Amazon.com's revenue is generated by its recommendation engine". The text continues: "Usualmente se usan en estrategias de venta cruzada, lealtad, awerness, targeting e innovación. Estos son algunos ejemplos:". At the bottom, there is a code cell with the output "Customers who viewed this item also viewed" and a "Code" button. The page also shows a sidebar with navigation options like "Version 4", "Notebook", "Sistemas De Recomendación", "Aplicaciones", "Hibrido", "Data", "Log", and "Comments".

**Version 4**  
4 commits

**Notebook**

**Sistemas De Recomendación**

Aplicaciones

Hibrido

Data

Log

Comments

## Sistemas de Recomendación

Busca identificar las preferencias de los usuarios a partir de información de sus selecciones, de los items, y de la población. Un sistema de recomendación o motor compara el perfil de usuarios con características de los items y predictivamente intenta identificar la calificación que este le daría. Estas características pueden ser dadas por el mismo item o inferidas implícitamente a través del contexto que dan otros usuarios al mismo.

- "More than 80 per cent of the TV shows people watch on Netflix are discovered through the platform's recommendation system"
- "35% of Amazon.com's revenue is generated by its recommendation engine"

Usualmente se usan en estrategias de venta cruzada, lealtad, awerness, targeting e innovación. Estos son algunos ejemplos:

Out[1]:

Customers who viewed this item also viewed

Code



THE  
DEVELOPER'S  
CONFERENCE



## Fácil de Instalar

```
$ pip install scikit-surprise
```

The screenshot shows the Anaconda Navigator application window. The title bar reads 'Anaconda Navigator'. The main header features the 'ANACONDA NAVIGATOR' logo and a 'Sign in' link. A left-hand sidebar contains navigation options: 'Home', 'Environments', 'Learning', and 'Community'. The 'Environments' section is active, displaying a search bar and a list of environments: 'base (root)', 'Booknando', 'Bot', 'Surprise', 'gl-env', and 'microservice'. The 'Surprise' environment is highlighted with a green bar and a play button icon. On the right, a table shows the details for the selected environment:

| Name  | Description |
|---|-------------|
| <input checked="" type="checkbox"/> scikit-surprise |             |



```
from __future__ import (absolute_import, division,
print_function, unicode_literals)
from surprise import SVD
from surprise import Dataset
from surprise import accuracy
from surprise.model_selection import train_test_split

data = Dataset.load_builtin('ml-1m')
trainset, testset = train_test_split(data, test_size=.15)
algoritmo = SVD(n_epochs=20)
algoritmo.fit(trainset)

uid = str(103)
iid = str(1499)

pred = algoritmo.predict(uid, iid, r_ui=1, verbose=True)
test_pred = algoritmo.test(testset)

print("Avaliação RMSE: ")
accuracy.rmse(test_pred, verbose=True)

print("Avaliação MAE: ")
accuracy.mae(test_pred, verbose=True)
```



**Fácil de Usar**





## **Matrix factorization-based**

SVD

SVD++

NMF

## **Neighborhood methods (KNN)**

KNNBasic

KNNWithMeans

KNNWithZScore

**SlopeOne**

**Co-Clustering**



THE  
DEVELOPER'S  
CONFERENCE



## Cosine

$$\text{cosine\_sim}(i, j) = \frac{\sum_{u \in U_{ij}} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \cdot \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}$$

THE  
DEVELOPER'S  
CONFERENCE

## MSD

$$\text{msd}(i, j) = \frac{1}{|U_{ij}|} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2$$

## Pearson

$$\text{pearson\_sim}(i, j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \mu_i) \cdot (r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \mu_i)^2} \cdot \sqrt{\sum_{u \in U_{ij}} (r_{uj} - \mu_j)^2}}$$

# Flexibilidade dos dados de entrada



THE  
DEVELOPER'S  
CONFERENCE



**Possui base de dados já implementado  
(MovieLens, Jester)**

**Muito simples adicionar ou usar outra  
entrada de dados**



**Não possui ferramentas para preparar os dados  
de entrada**



```
1 from surprise import SVD
2 from surprise import Dataset
3 from surprise.model_selection import cross_validate
4
5
6 # Load the movielens-100k dataset (download it if needed),
7 data = Dataset.load_builtin('ml-100k')
8
9 # We'll use the famous SVD algorithm.
10 algo = SVD()
11
12 # Run 5-fold cross-validation and print results
13 cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5,
    verbose=True)
```

In [1]: (executing file "svd\_exemplo.py")

Dataset ml-100k could not be found. Do you want to download it? [Y/n] Y

Trying to download dataset from <http://files.grouplens.org/datasets/movielens/ml-100k.zip>...

Done! Dataset ml-100k has been saved to C:\WinPy3702\settings\.surprise\_data/ml-100k

Evaluating RMSE MAE of algorithm SVD on 5 splits (a)



## Possui um módulo que facilita a criação de novos algoritmos



THE  
DEVELOPER'S  
CONFERENCE

From file `examples/building_custom_algorithms/most_basic_algorithm.py`

```
from surprise import AlgoBase
from surprise import Dataset
from surprise.model_selection import cross_validate

class MyOwnAlgorithm(AlgoBase):

    def __init__(self):

        # Always call base method before doing anything.
        AlgoBase.__init__(self)

    def estimate(self, u, i):

        return 3

data = Dataset.load_builtin('ml-100k')
algo = MyOwnAlgorithm()

cross_validate(algo, data, verbose=True)
```



## Possui três metodologias de avaliação a RMSE, a MAE e a FCP



THE  
DEVELOPER'S  
CONFERENCE

```
# Run 5-fold cross-validation and print results
cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5,
               verbose=True)
```



Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

|                | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5 | Mean   | Std    |
|----------------|--------|--------|--------|--------|--------|--------|--------|
| RMSE (testset) | 0.9408 | 0.9403 | 0.9275 | 0.9279 | 0.9426 | 0.9358 | 0.0067 |
| MAE (testset)  | 0.7439 | 0.7400 | 0.7281 | 0.7358 | 0.7401 | 0.7376 | 0.0054 |
| Fit time       | 4.35   | 4.40   | 4.36   | 4.36   | 4.35   | 4.37   | 0.02   |
| Test time      | 0.14   | 0.12   | 0.16   | 0.11   | 0.11   | 0.13   | 0.02   |







## Não possui ferramenta específica para esparsidade de dados

- In Netflix: **98.8 %** of the ratings are unknown
- In Movielens: **95.7 %** of the ratings are unknown

Items

|   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 |   |   |   |   |   |   | 2 |   |   |
| 2 |   | 5 |   |   |   |   |   | 3 | 1 |
| 3 |   |   |   | 1 | 5 |   |   |   |   |
| 4 | 5 |   |   |   |   |   |   | 5 |   |
| 5 |   |   |   |   |   |   |   |   |   |
| 6 | 4 |   |   | 1 |   |   | 4 |   |   |
| 7 |   |   |   |   |   |   |   |   |   |
| 8 |   | 5 | 4 |   |   |   |   |   |   |
| 9 |   |   |   |   |   |   | 5 |   |   |

Users

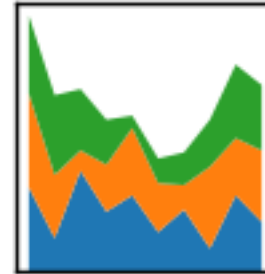
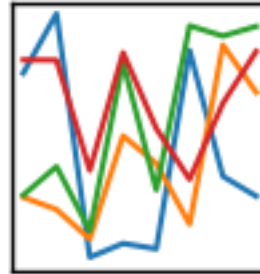
Ratings  
★★★★☆



**Não possuí ferramenta específica para lidar com grande quantidade de dados.**

# pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$





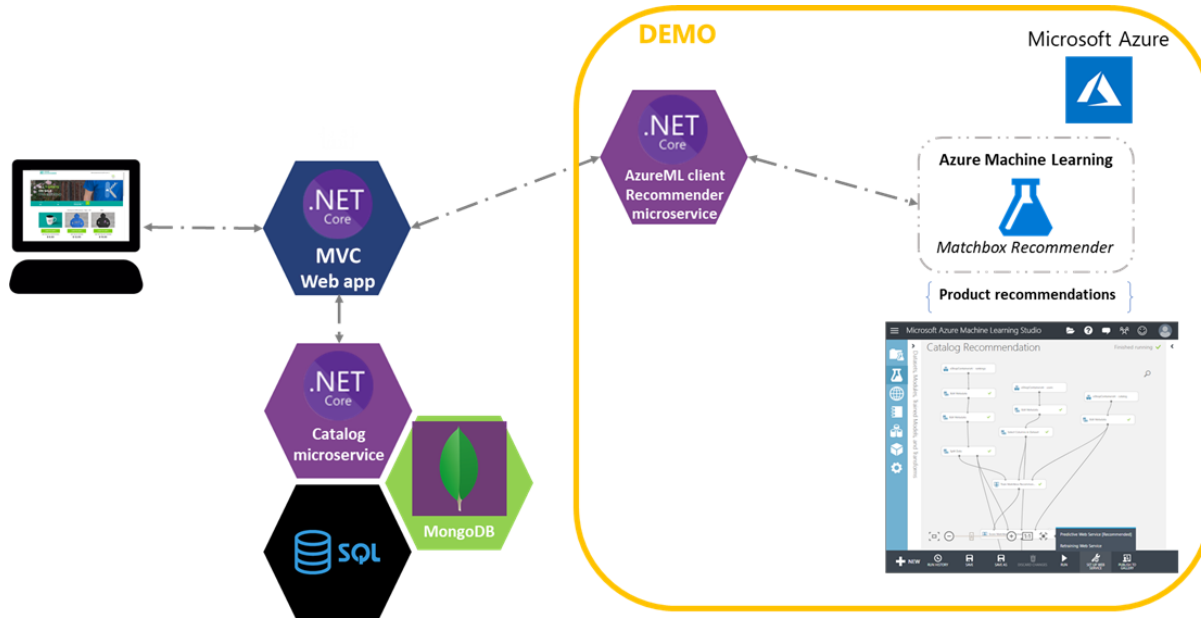
**Alguns algoritmos possuem um desempenho baixo**

**SVD++**





## Não foi pensado para ambiente de produção





THE  
DEVELOPER'S  
CONFERENCE

# SCIKIT-SURPRISE

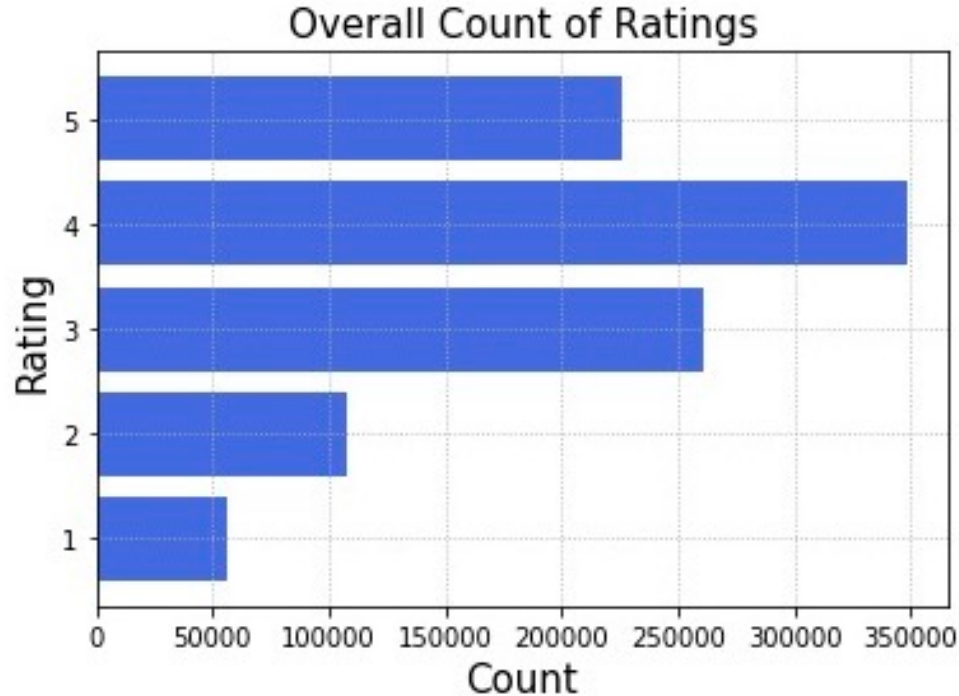
Brincando com o Movielens



# Movie Lens 1M



**6.040** usuários    **3.706** filmes    **1.000.209** avaliações



| <b>Movielens 1M</b> | <b>RMSE</b> | <b>MAE</b> | <b>Time</b> |
|---------------------|-------------|------------|-------------|
| SVD++               | 0,862       | 0,672      | 6:12:38     |
| SVD                 | 0,874       | 0,686      | 0:05:28     |
| k-NN Baseline       | 0,895       | 0,706      | 0:16:39     |
| Slope One           | 0,907       | 0,715      | 0:08:23     |
| Co-Clustering       | 0,916       | 0,718      | 0:01:59     |
| Baseline            | 0,909       | 0,719      | 0:00:38     |
| NMF                 | 0,917       | 0,725      | 0:05:31     |
| k-NN                | 0,923       | 0,727      | 0:15:11     |
| k-NN With Means     | 0,929       | 0,738      | 0:16:02     |
| Random              | 1.506       | 1.207      | 0:00:36     |



THE  
DEVELOPER'S  
CONFERENCE



**Rating Alto:** ID 49,  
idade entre 18 e 24 anos,  
programador e mora em Huston no Texas (USA).  
Ele deu um rating **alto (4)** para o filme  
**The Negotiator (1998)**, um filme de ação Ação e Thriller



**Rating Baixo:** ID 103  
idade entre 45 e 49 anos  
um executivo/manager com que mora em San Diego.  
Ele avaliou com um **rating baixo (1)** o filme  
**Anaconda (1997)**, um filme de ação/aventura/Thriller



```
from __future__ import (absolute_import, division, print_function,  
                        unicode_literals)  
  
from surprise import KNNBasic  
from surprise import Dataset  
from surprise import accuracy  
from surprise.model_selection import train_test_split  
  
# Load the movielens-100k dataset UserID::MovieID::Rating::Timestamp  
data = Dataset.load_builtin('ml-1m')  
trainset, testset = train_test_split(data, test_size=.5)
```



```
algoritmo = KNNBasic(k=50, sim_options={'name': 'pearson',  
'user_based': True, 'verbose' : True})
```

```
algoritmo.fit(trainset)
```

```
uid = str(49)
```

```
iid = str(2058)
```

```
pred = algoritmo.predict(uid, iid, r_ui=4, verbose=True)
```



```
# run the trained model against the testset
```

```
test_pred = algoritmo.test(testset)
```

```
# Avalia RMSE
```

```
print("Avaliação RMSE: ")
```

```
accuracy.rmse(test_pred, verbose=True)
```

```
# Avalia MAE
```

```
print("Avaliação MAE: ")
```

```
accuracy.mae(test_pred, verbose=True)
```





Predição de avaliação:

user: 49

item: 2058

$r_{ui} = 4.00$

est = 3.55

```
(Booknando) MacBookBooknando:recommenderSystem josefernandotavares$ python KNNWithMeans_single.py
Usando o algoritmo KNNWithMeans com 50 vizinhos
Algoritmo de similaridade: Pearson
Computing the pearson similarity matrix...
Done computing similarity matrix.
Predição de avaliação:
user: 49          item: 2058           $r_{ui} = 4.00$     est = 3.55    {'actual_k': 50, 'was_impossible': False}
Avaliação RMSE:
RMSE: 0.9736
Avaliação MAE:
MAE: 0.7751
(Booknando) MacBookBooknando:recommenderSystem josefernandotavares$
```

# SVD++



THE  
DEVELOPER'S  
CONFERENCE

## Para Rating ALTO:

| User      | Item        | r_ui     | est         | Epochs    | Test        | RMSE          | MAE           |
|-----------|-------------|----------|-------------|-----------|-------------|---------------|---------------|
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,52</b> | <b>20</b> | <b>0,15</b> | <b>0,8583</b> | <b>0,6693</b> |
| 49        | 2058        | 4        | 3,68        | 5         | 0,15        | 0,8854        | 0,6974        |
| 49        | 2058        | 4        | 3,62        | 5         | 0,15        | 0,888         | 0,7002        |

# SVD++



THE  
DEVELOPER'S  
CONFERENCE

## Para Rating Baixo:

| User       | Item        | r_ui     | est         | Epochs    | Test        | RMSE          | MAE           |
|------------|-------------|----------|-------------|-----------|-------------|---------------|---------------|
| <b>103</b> | <b>1499</b> | <b>1</b> | <b>2,62</b> | <b>20</b> | <b>0,15</b> | <b>0,8583</b> | <b>0,6693</b> |
| 103        | 1499        | 1        | 3,06        | 5         | 0,15        | 0,867         | 0,6806        |
| 103        | 1499        | 1        | 3,08        | 5         | 0,15        | 0,8878        | 0,7           |

# KNNWithMeans



THE  
DEVELOPER'S  
CONFERENCE

## Para Rating alto:

| Use<br>r  | Item        | r_ui     | est         | actual_k  | Test        | Av.RMSE       | Av.MAE        |
|-----------|-------------|----------|-------------|-----------|-------------|---------------|---------------|
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,76</b> | <b>40</b> | <b>0,15</b> | <b>0,9132</b> | <b>0,7184</b> |
| 49        | 2058        | 4        | 3,66        | 40        | 0,15        | 0,9159        | 0,7208        |
| 49        | 2058        | 4        | 3,77        | 30        | 0,15        | 0,9215        | 0,7263        |
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,9</b>  | <b>30</b> | <b>0,15</b> | <b>0,9237</b> | <b>0,728</b>  |
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,9</b>  | <b>10</b> | <b>0,15</b> | <b>0,957</b>  | <b>0,7548</b> |
| 49        | 2058        | 4        | 3,69        | 10        | 0,15        | 0,9575        | 0,7549        |
| 49        | 2058        | 4        | 3,39        | 10        | 0,15        | 0,9578        | 0,7563        |

# KNNWithMeans



THE  
DEVELOPER'S  
CONFERENCE

## Para Rating Baixo:

| User       | Item        | r_ui     | est         | actual_k  | Test        | Av.RMSE       | Av.MAE        |
|------------|-------------|----------|-------------|-----------|-------------|---------------|---------------|
| 103        | 1499        | 1        | 1,63        | 10        | 0,15        | 0,9592        | 0,7564        |
| <b>103</b> | <b>1499</b> | <b>1</b> | <b>1,27</b> | <b>10</b> | <b>0,15</b> | <b>0,9587</b> | <b>0,7564</b> |
| 103        | 1499        | 1        | 1,84        | 10        | 0,15        | 0,959         | 0,757         |

# KNNWithMeans X SVDpp



THE  
DEVELOPER'S  
CONFERENCE

para Rating Baixo:

estimativa distante

MAE BAIXO

|                     |     |      |   |      |    |      |        |        |
|---------------------|-----|------|---|------|----|------|--------|--------|
| <b>SVDpp</b>        | 103 | 1499 | 1 | 2,62 | 20 | 0,15 | 0,8583 | 0,6693 |
| <b>KNNWithMeans</b> | 103 | 1499 | 1 | 1,27 | 10 | 0,15 | 0,9587 | 0,7564 |

estimativa próxima

MAE ALTO

# SVD



THE  
DEVELOPER'S  
CONFERENCE

## Para Rating alto:

| User      | Item        | r_ui     | est         | Epochs    | Test        | Av.RMS<br>E   | Av.MAE        |
|-----------|-------------|----------|-------------|-----------|-------------|---------------|---------------|
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,48</b> | <b>10</b> | <b>0,15</b> | <b>0,8874</b> | <b>0,7003</b> |
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,72</b> | <b>10</b> | <b>0,15</b> | <b>0,8874</b> | <b>0,7015</b> |
| 49        | 2058        | 4        | 3,53        | 10        | 0,15        | 0,8883        | 0,7017        |
| 49        | 2058        | 4        | 3,6         | 5         | 0,15        | 0,9115        | 0,7233        |
| 49        | 2058        | 4        | 3,56        | 5         | 0,15        | 0,9128        | 0,7237        |
| 49        | 2058        | 4        | 3,49        | 5         | 0,15        | 0,9145        | 0,7246        |

# SlopOne:



## Para Rating alto:

| User      | Item        | r_ui     | est         | Test        | Av.RMSE       | Av.MAE        |
|-----------|-------------|----------|-------------|-------------|---------------|---------------|
| <b>49</b> | <b>2058</b> | <b>4</b> | <b>3,54</b> | <b>0,15</b> | <b>0,9029</b> | <b>0,7126</b> |
| 49        | 2058        | 4        | 3,48        | 0,15        | 0,9064        | 0,7146        |
| 49        | 2058        | 4        | 3,55        | 0,15        | 0,9079        | 0,716         |





## SlopOne:

### Para Rating Baixo:

|            |             |          |             |             |               |               |
|------------|-------------|----------|-------------|-------------|---------------|---------------|
| <b>103</b> | <b>1499</b> | <b>1</b> | <b>3,09</b> | <b>0,15</b> | <b>0,9047</b> | <b>0,7128</b> |
| 103        | 1499        | 1        | 2,99        | 0,15        | 0,906         | 0,7146        |
| 103        | 1499        | 1        | 3,01        | 0,15        | 0,9049        | 0,7131        |

É importante entender bem o conjunto de dados e como cada algoritmo interage com eles. É provável que um mesmo algoritmo que deu bons resultados neste conjunto de dados pode não dar os mesmos resultados com outro.

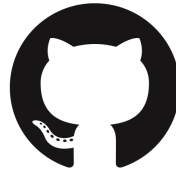
O **scikit-surprise** se demonstrou uma boa ferramenta para o estudo dos algoritmos e estudos off-line de dataset.

Mesmo com poucos conhecimento de Python é possível iniciar a estudar os algoritmos por trás dos sistemas de recomendação.



THE  
DEVELOPER'S  
CONFERENCE

**OBRIGADO!**



<https://github.com/JFTavares/recommenderSystem>

 **Booknando**