Photo credit: Paxabay

Building a Collaborative Filtering Recommender System with ClickStream Data

How to implement a recommendation algorithm based on prior implicit feedback.

Susan Li Follow
Apr 20, 2019 · 7 min read

Recommender systems are everywhere, helping you find everything from books to romantic dates, hotels to restaurants.

There are all kinds of recommender systems for all sorts of situations, depends on your needs and available data.

Explicit vs Implicit

Let's face it, **explicit** feedback is hard to collect as they require additional input from the users. The users give explicit feedback only when they choose to do so. As a result, most of the time, people don't provide ratings at all (I myself totally guilty of this on Amazon!). Therefore, the amount of explicit data collected are extremely scarce.

On the other hand, **implicit** data is easy to collect in large quantities without any effort from the users. The goal is to convert user behavior into user preferences which indirectly reflect opinion through observing user behavior. For example, a user that bookmarked many articles by the same author probably likes that author.

The Data

Our goal today is to develop a recommender system with implicit data collection which is clickstream data, in our case.

It is very hard to find public available data for this project. I am using data from Articles sharing and reading from CI&T DeskDrop. Deskdrop is an internal communications platform that allows companies employees to share relevant articles with their peers, and collaborate around them.

The data contains about 73k users interactions on more than 3k public articles shared in the platform, more importantly, it contains rich implicit feedback, different interaction types were logged, making it possible to infer the user's level of interest in the articles.
我们将使用隐式库，一个快速的Python协作过滤，用于我们的矩阵分解。
And we will be using Implicit Library, a Fast Python Collaborative Filtering for Implicit Datasets, for our matrix factorization.

Data Pre-processing

- Remove columns that we do not need.
- Remove `eventType == 'CONTENT REMOVED'` from `articles_df`.
- Merge `interactions_df` with `articles_df`.

```
import pandas as pd
import scipy.sparse as sparse
import numpy as np
import random
import implicit
from sklearn.preprocessing import MinMaxScaler

articles_df = pd.read_csv('shared_articles.csv')
interactions_df = pd.read_csv('users_interactions.csv')
articles_df.drop(['authorUserAgent', 'authorRegion', 'authorCountry'], axis=1, inplace=True)
```

```
interactions_df.drop(['userAgent', 'userRegion', 'userCountry'], axis=1, inplace=True)

articles_df = articles_df[articles_df['eventType'] == 'CONTENT SHARED']
articles_df.drop('eventType', axis=1, inplace=True)
df = pd.merge(interactions_df[['contentId','personId', 'eventType']], articles_df[['contentId',
```

This is the data set that will get us to start:

	contentId	personId	eventType	title
0	-3499919498720038879	-8845298781299428018	VIEW	Hiri wants to fix the workplace email problem
1	-3499919498720038879	-8845298781299428018	VIEW	Hiri wants to fix the workplace email problem
2	-3499919498720038879	-108842214936804958	VIEW	Hiri wants to fix the workplace email problem
3	-3499919498720038879	-1443636648652872475	VIEW	Hiri wants to fix the workplace email problem
4	-3499919498720038879	-1443636648652872475	VIEW	Hiri wants to fix the workplace email problem
5	-3499919498720038879	-1443636648652872475	VIEW	Hiri wants to fix the workplace email problem
6	-3499919498720038879	-8020832670974472349	VIEW	Hiri wants to fix the workplace email problem
7	-3499919498720038879	-8020832670974472349	VIEW	Hiri wants to fix the workplace email problem
8	-3499919498720038879	-9009798162809551896	LIKE	Hiri wants to fix the workplace email problem
9	-3499919498720038879	-9009798162809551896	VIEW	Hiri wants to fix the workplace email problem

Table 1

这告诉我们每个人对每种内容具有哪种事件类型。 有很多重复的记录，我们将在短期内删除它们。

This tells us what event type each person has with each content. There are many duplicated records and we will remove them shortly.

```
df['eventType'].value_counts()
```

```
VIEW                61043
LIKE                 5745
BOOKMARK             2463
COMMENT CREATED      1611
FOLLOW               1407
Name: eventType, dtype: int64
```

Name: eventType, dtype: int64

Figure 1

The eventType values are:

- **VIEW**: The user has opened the article. A page view in a content site can mean many things. It can mean that the user is interested, or maybe user is just lost or clicking randomly.
- **LIKE**: The user has liked the article.
- **BOOKMARK**: The user has bookmarked the article for easy return in the future. This is a strong indication that the user finds something of interest.
- **COMMENT CREATED**: The user left a comment on the article.
  用户选择收到有关该文章的任何新评论的通知。
- **FOLLOW**: The user chose to be notified on any new comment about the article.

We are going to associate each eventType with a weight or strength. It is reasonable to assume that for example, a bookmark on an article indicates a higher interest of the user on that article than a like.

```
event_type_strength = {
   'VIEW': 1.0,
   'LIKE': 2.0,
   'BOOKMARK': 3.0,
   'FOLLOW': 4.0,
   'COMMENT CREATED': 5.0,
}

df['eventStrength'] = df['eventType'].apply(lambda x:
event_type_strength[x])
```

	contentId	personId	eventType	title	eventStrength
0	-3499919498720038879	-8845298781299428018	VIEW	Hiri wants to fix the workplace email problem	1.0
2	-3499919498720038879	-108842214936804958	VIEW	Hiri wants to fix the workplace email problem	1.0
3	-3499919498720038879	-1443636648652872475	VIEW	Hiri wants to fix the workplace email problem	1.0
6	-3499919498720038879	-8020832670974472349	VIEW	Hiri wants to fix the workplace email problem	1.0
8	-3499919498720038879	-9009798162809551896	LIKE	Hiri wants to fix the workplace email problem	2.0
9	-3499919498720038879	-9009798162809551896	VIEW	Hiri wants to fix the workplace email problem	1.0

9	-3499919498720038879	-9009798162809551896	VIEW	Hiri wants to fix the workplace email problem	1.0
11	-3499919498720038879	-3596626804281480007	VIEW	Hiri wants to fix the workplace email problem	1.0
12	-3499919498720038879	-3596626804281480007	BOOKMARK	Hiri wants to fix the workplace email problem	3.0
15	-3499919498720038879	-108842214936804958	LIKE	Hiri wants to fix the workplace email problem	2.0
17	-3499919498720038879	-7487897518634781969	VIEW	Hiri wants to fix the workplace email problem	1.0

Table 2

- Drop duplicated records.
- Group eventStrength together with person and content.

```
df = df.drop_duplicates()
grouped_df = df.groupby(['personId', 'contentId',
'title']).sum().reset_index()
grouped_df.sample(10)
```

- We get the final result of grouped eventStrength.

	personId	contentId	title	eventStrength
11755	-3325685634318512394	2727743992157210358	The Future of Digital Banking is Now	1.0
14802	-2012808108604118403	-2314360211211303885	O meme 'Luiza, você está atenta?' explicou da ...	1.0
4152	-7381477755938439823	8690601908689146128	A New Number Format for Computers Could Nuke A...	1.0
27923	3302556033962996625	-4089455027188314262	Python Is Not Java (dirtSimple.org)	4.0
22423	881856221521045800	385212706171385893	Don Tapscott: How the blockchain is changing m...	10.0
29640	3617923130431400422	-6590819806697898649	Listas com RecyclerView - Android Dev BR	1.0
34929	6018247344671480458	-5848755753905481990	Incrível, Skype se lembrou que o Linux existe!	1.0
38661	8237333675050684529	2910288655685129221	Black Mirror (TV Series 2011-)	1.0
34309	5660542693104786364	3241240229241793423	Startup usa algoritmo e rede social para aprov...	6.0
1905	-8607239111818252463	1469580151036142903	Don't document your code. Code your documentat...	1.0

Table 3

Alternating Least Squares Recommender Model Fitting

eventStrength不是表示一个明确的评级，而是表示对交互作用强度的信心

Instead of representing an explicit rating, the eventStrength can represent a "confidence" in terms of how strong the interaction was. Articles with a larger number of eventStrength by a person can carry more weight in our ratings matrix of eventStrength.

eventStrength不是表示一个明确的评级，而是表示对交互作用强度的信心。一个人所写的含有较多eventStrength的文章在我们的eventStrength评级矩阵中会有更多的权重

- To get around “negative integer” warning, I will have to create numeric `person_id` and `content_id` columns.
- Create two matrices, one for fitting the model (content-person) and one for recommendations (person-content).
- Initialize the Alternating Least Squares (ALS) recommendation model.
- Fit the model using the sparse content-person matrix.
- We set the type of our matrix to double for the ALS function to run properly.

implicit_als_model.py

Finding the Similar Articles

We are going to find the top 10 most similar articles for content_id = 450, titled “***Google’s fair use victory is good for open source***”, this article seems talk about Google and open source.

从我们训练过的模型中获取人员和内容向量

- Get the person and content vectors from our trained model.

计算向量范数

- Calculate the vector norms.

计算相似性得分

- Calculate the similarity score.
- Get the top 10 contents.

用本文创建最相似文章的内容得分元组列表

- Create a list of content-score tuples of most similar articles with this article.

similar_content.py

Figure 2

The 1st article is itself. The other 9 articles are about Google, or opensource software, or cloud, or AI, or the other tech companies. I am sure you will agree with me that they are all some what similar with the first one!

Recommend Articles to Persons

The following function will return the top 10 recommendations chosen based on the person / content vectors for contents never interacted with for any given person.

从稀疏的人员内容矩阵中获得交互得分

- Get the interactions score from the sparse person content matrix.
- Add 1 to everything, so that articles with no interaction yet become equal to 1.
- Make articles already interacted zero.
- Get dot product of person vector and all content vectors.
- Scale this recommendation vector between 0 and 1.
- Content already interacted have their recommendation multiplied by zero.
- Sort the indices of the content into order of best recommendations.
- Start empty list to store titles and scores.
- Append titles and scores to the list.
- Get the trained person and content vectors. We convert them to csr matrices.
- Create recommendations for person with id 50.

implicit_rec_als_id_50.py

Figure 3

在这里，我们为person_id = 50提供了十大建议。它们有意义吗？ 让我们获得此人与之互动的十大文章

Here we have top 10 recommendations for person_id = 50. Do they make sense? Let's get top 10 articles this person has interacted with.

```
grouped_df.loc[grouped_df['person_id'] == 50].sort_values(by=
['eventStrength'], ascending=False)[['title', 'person_id',
'eventStrength']].head(10)
```


Table 4

显然，这个人对Drupal等开源CMS的文章感兴趣，她也阅读软件开发和商业相关的文章，如谷歌，Slack或Johnson Johnson。

Apparently, this person is interested in articles on open source CMS such as Drupal, she also reads software development and business related articles, namely "Google", "Slack" or "Johnson Johnson".

我们推荐给她的文章包括Drupal的数字体验、信息技术与人类、软件开发和关于谷歌的商业文章

The articles we recommended to her includes Drupal for digital experience, information technology vs. humanity, software development, and business articles about Google.

Pretty impressive! Let's try one more.

We recommended the following articles to person_id = 1:

```
person_id = 1

recommendations = recommend(person_id, sparse_person_content,
person_vecs, content_vecs)

print(recommendations)
```


Figure 4

The following are the articles person_id = 1 has interacted with:

```
grouped_df.loc[grouped_df['person_id'] == 1].sort_values(by=
['eventStrength'], ascending=False)[['title', 'eventStrength',
'person_id']]
```

Table 5

显然，此人仅与5篇文章进行了互动，并且她的兴趣似乎非常有限。 她互动的文章是关于学习日语和/或android开发的。

Apparently, this person has only interacted with 5 articles and she seems to have very limited interest. The articles she interacted was about learning Japanese language and / or android development.

The articles we recommended to her includes learning Japanese language, android development and user interface design. Cool!

Evaluation the Recommender System

The above spot checks look all good. But the best evaluation metrics for a recommender system is how much the system adds value to the end user and/or business, whether the system increase page views, likes, bookmarks, follows and comments. We will want to do some kind online A/B testing to evaluate these metrics.

However, before we push the recommender system online, there are other common metrics for evaluating the performance of a recommender in isolation. By following this tutorial, we were able to calculate the AUC for each person in our training set that had at least one article masked. And AUC for the most popular articles for the persons to compare.

Jupyter notebook can be found on Github. Happy Easter!

References:

AlternatingLeastSquares - Implicit 0.3.8 documentation

A Recommendation Model based off the algorithms described in the paper 'Collaborative Filtering for Implicit Feedback...

implicit.readthedocs.io

ALS Implicit Collaborative Filtering

Continuing on the collaborative filtering theme from my collaborative filtering with binary data example i'm going to...

medium.com

Recommender Systems in Python 101

Using data from Articles sharing and reading from CI&T DeskDrop

www.kaggle.com

Machine Learning Recommendation System Python Artificial Intelligence Implicit