

09RecommendationSystem

November 4, 2018

1. 推薦系統的種類

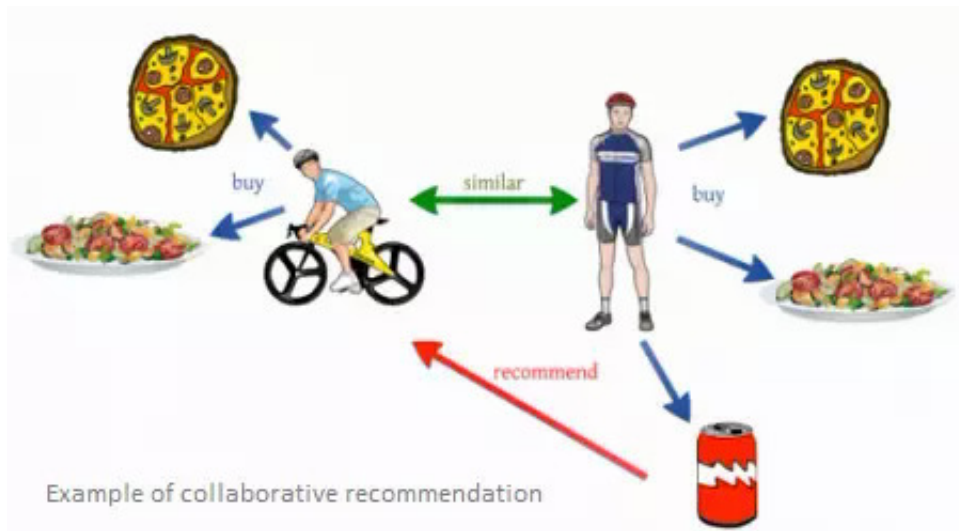
- collabrative(協同推薦)
- content-based(內容推薦)

from quora

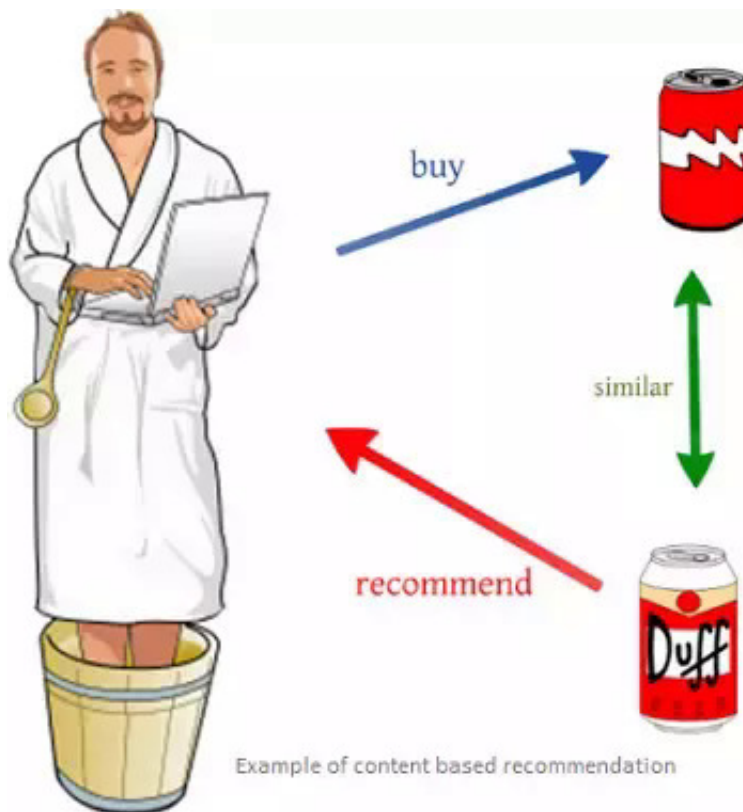
```
In [32]: import re
import pandas as pd
import numpy as np
from collections import Counter
from sklearn import preprocessing
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from matplotlib import pyplot as plt
import xgboost

In [2]: df_articles = pd.read_csv('shared_articles.csv')
df_articles = df_articles[df_articles['eventType'] == 'CONTENT SHARED']
df_articles.info()
df_articles.head(5)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3047 entries, 1 to 3121
Data columns (total 13 columns):
timestamp          3047 non-null int64
eventType          3047 non-null object
contentId          3047 non-null int64
authorPersonId     3047 non-null int64
authorSessionId    3047 non-null int64
authorUserAgent    669 non-null object
authorRegion       669 non-null object
authorCountry      669 non-null object
contentType        3047 non-null object
url                3047 non-null object
```



collabrative



content-based

```

title          3047 non-null object
text           3047 non-null object
lang           3047 non-null object
dtypes: int64(4), object(9)
memory usage: 333.3+ KB

```

```

Out[2]:
   timestamp      eventType      contentId      authorPersonId \
1  1459193988  CONTENT SHARED -4110354420726924665  4340306774493623681
2  1459194146  CONTENT SHARED -7292285110016212249  4340306774493623681
3  1459194474  CONTENT SHARED -6151852268067518688  3891637997717104548
4  1459194497  CONTENT SHARED  2448026894306402386  4340306774493623681
5  1459194522  CONTENT SHARED -2826566343807132236  4340306774493623681

      authorSessionId authorUserAgent authorRegion authorCountry contentType \
1  8940341205206233829             NaN             NaN             NaN      HTML
2  8940341205206233829             NaN             NaN             NaN      HTML
3 -1457532940883382585             NaN             NaN             NaN      HTML
4  8940341205206233829             NaN             NaN             NaN      HTML
5  8940341205206233829             NaN             NaN             NaN      HTML

                                     url \
1  http://www.nytimes.com/2016/03/28/business/dea...
2  http://cointelegraph.com/news/bitcoin-future-w...
3  https://cloudplatform.googleblog.com/2016/03/G...
4  https://bitcoinmagazine.com/articles/ibm-wants...
5  http://www.coindesk.com/ieee-blockchain-oxford...

                                     title \
1  Ethereum, a Virtual Currency, Enables Transact...
2  Bitcoin Future: When GBPcoin of Branson Wins O...
3                                     Google Data Center 360ř Tour
4  IBM Wants to "Evolve the Internet" With Blockc...
5  IEEE to Talk Blockchain at Cloud Computing Oxf...

                                     text lang
1  All of this work is still very early. The firs...  en
2  The alarm clock wakes me at 8:00 with stream o...  en
3  We're excited to share the Google Data Center ...  en
4  The Aite Group projects the blockchain market ...  en
5  One of the largest and oldest organizations fo...  en

```

```

In [3]: def process_Agent(Agent):
         if "windows" in Agent:
             return 0
         elif "macintosh" in Agent:
             return 1
         elif "linux" in Agent:

```

```

        return 2
    else:
        return 3

le_authorUserAgent = preprocessing.LabelEncoder()
df_articles['authorUserAgent'] = le_authorUserAgent.fit_transform(df_articles['authorUserAgent'])
le_authorRegion = preprocessing.LabelEncoder()
df_articles['authorRegion'] = le_authorRegion.fit_transform(df_articles['authorRegion'])
le_authorCountry = preprocessing.LabelEncoder()
df_articles['authorCountry'] = le_authorCountry.fit_transform(df_articles['authorCountry'])
le_contentType = preprocessing.LabelEncoder()
df_articles['contentType'] = le_contentType.fit_transform(df_articles['contentType']).astype(str)
le_url = preprocessing.LabelEncoder()
df_articles['url'] = le_url.fit_transform(df_articles['url']).apply(lambda x: re.findall(r'http://', x))
le_lang = preprocessing.LabelEncoder()
df_articles['lang'] = le_lang.fit_transform(df_articles['lang']).astype(str).apply(lambda x: re.findall(r'\b([a-z]{2,3})\b', x))

tfidf = TfidfVectorizer()
tfidf.fit(np.hstack([df_articles['title'].values, df_articles['text'].values]))
df_articles['title'] = list(tfidf.transform(df_articles['title'].values).toarray())
df_articles['text'] = list(tfidf.transform(df_articles['text'].values).toarray())

df_articles = df_articles[["contentId", "authorUserAgent", "authorRegion", "authorCountry", "contentType", "url", "lang", "title", "text"]]
df_articles.head(5)

```

```

Out[3]:
   contentId  authorUserAgent  authorRegion  authorCountry  \
1 -4110354420726924665         3           9             3
2 -7292285110016212249         3           9             3
3 -6151852268067518688         3           9             3
4  2448026894306402386         3           9             3
5 -2826566343807132236         3           9             3

   contentType  url  lang  title \
1           0  896    0  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
2           0  182    0  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
3           0  176    0  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
4           0   50    0  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
5           0  706    0  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...

   text
1  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
2  [0.048182400537760725, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
3  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
4  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
5  [0.0, 0.03408482227280377, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...

```

```

In [5]: X_nlp_forPCA = np.concatenate([

```

```

np.hstack(df_articles['title'].values).reshape(len(df_articles),-1),
np.hstack(df_articles['text'].values).reshape(len(df_articles),-1)
], axis=1)

pca = PCA(n_components=400, random_state=1212)
df_articles['title'] = list(pca.fit_transform(X_nlp_forPCA[:, :int(X_nlp_forPCA.shape[1])].values).reshape(len(df_articles),-1))
df_articles['text'] = list(pca.fit_transform(X_nlp_forPCA[:, int(X_nlp_forPCA.shape[1]):].values).reshape(len(df_articles),-1))
df_articles.head(5)

```

```

Out[5]:
   contentId  authorUserAgent  authorRegion  authorCountry  \
1 -4110354420726924665          3           9             3
2 -7292285110016212249          3           9             3
3 -6151852268067518688          3           9             3
4  2448026894306402386          3           9             3
5 -2826566343807132236          3           9             3

   contentType  url  lang  title  \
1           0  896    0  [-0.04075413153557972, -0.015353894409987887, ...
2           0  182    0  [-0.0012556766428318809, -0.06646168057041796, ...
3           0  176    0  [0.11418731903476237, 0.17239345455221505, -0...
4           0   50    0  [0.08234964109707694, -0.13114805530673168, -0...
5           0  706    0  [0.04849331403920229, 0.012213744929441139, -0...

   text
1  [-0.25381908135912207, -0.03928945983356414, -...
2  [-0.10704562138903156, 0.0692688189810105, -0...
3  [-0.20080173580474742, 0.0334745050091199, 0.0...
4  [-0.29032820837142886, -0.08506507624478626, -...
5  [-0.16041997828540744, 0.057870128420256864, -...

```

```

In [6]: df_interactions = pd.read_csv('users_interactions.csv')
df_interactions.info()
df_interactions.head(5)

```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72312 entries, 0 to 72311
Data columns (total 8 columns):
timestamp      72312 non-null int64
eventType      72312 non-null object
contentId      72312 non-null int64
personId       72312 non-null int64
sessionId      72312 non-null int64
userAgent      56918 non-null object
userRegion     56907 non-null object
userCountry    56918 non-null object
dtypes: int64(4), object(4)
memory usage: 4.4+ MB

```

```

Out [6]:      timestamp eventType      contentId      personId \
0  1465413032      VIEW -3499919498720038879 -8845298781299428018
1  1465412560      VIEW  8890720798209849691 -1032019229384696495
2  1465416190      VIEW  310515487419366995 -1130272294246983140
3  1465413895  FOLLOW  310515487419366995   344280948527967603
4  1465412290      VIEW -7820640624231356730 -445337111692715325

      sessionId      userAgent \
0  1264196770339959068      NaN
1  3621737643587579081  Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2...
2  2631864456530402479      NaN
3  -3167637573980064150      NaN
4  5611481178424124714      NaN

      userRegion userCountry
0      NaN      NaN
1      NY      US
2      NaN      NaN
3      NaN      NaN
4      NaN      NaN

```

```

In [7]: score_mapping = {'BOOKMARK':5,
                        'COMMENT CREATED':4,
                        'FOLLOW':3,
                        'LIKE':2,
                        'VIEW':1
                        }

df_interactions['eventType'] = df_interactions['eventType'].apply(score_mapping.get)

le_userAgent = preprocessing.LabelEncoder()
df_interactions['userAgent'] = le_userAgent.fit_transform(df_interactions['userAgent'])
le_userRegion = preprocessing.LabelEncoder()
df_interactions['userRegion'] = le_userRegion.fit_transform(df_interactions['userRegion'])
le_userCountry = preprocessing.LabelEncoder()
df_interactions['userCountry'] = le_userCountry.fit_transform(df_interactions['userCountry'])

df_interactions = df_interactions[["contentId", "userAgent", "userRegion", "userCountry", "eventType"]]
df_interactions.head(5)

```

```

Out [7]:      contentId  userAgent  userRegion  userCountry  eventType
0 -3499919498720038879         3         43         18         1
1  8890720798209849691         1         50         22         1
2   310515487419366995         3         43         18         1
3   310515487419366995         3         43         18         3
4 -7820640624231356730         3         43         18         1

```

```

In [8]: df_merge = pd.merge(df_interactions, df_articles, on='contentId')
df_merge.head(5)

```

```

Out [8]:
      contentId  userAgent  userRegion  userCountry  eventType  \
0 -3499919498720038879      3         43          18          1
1 -3499919498720038879      1         66           2          1
2 -3499919498720038879      0         66           2          1
3 -3499919498720038879      2         66           2          1
4 -3499919498720038879      2         66           2          1

      authorUserAgent  authorRegion  authorCountry  contentType  url  lang  \
0              3              9              3              0  569    0
1              3              9              3              0  569    0
2              3              9              3              0  569    0
3              3              9              3              0  569    0
4              3              9              3              0  569    0

      title  \
0 [0.04436414967487065, -0.0828542054558259, -0...
1 [0.04436414967487065, -0.0828542054558259, -0...
2 [0.04436414967487065, -0.0828542054558259, -0...
3 [0.04436414967487065, -0.0828542054558259, -0...
4 [0.04436414967487065, -0.0828542054558259, -0...

      text
0 [-0.2306608936690724, -0.04000880102084441, 0...
1 [-0.2306608936690724, -0.04000880102084441, 0...
2 [-0.2306608936690724, -0.04000880102084441, 0...
3 [-0.2306608936690724, -0.04000880102084441, 0...
4 [-0.2306608936690724, -0.04000880102084441, 0...

```

```

In [18]: X_part1 = df_merge[["userAgent", "userRegion", "userCountry", "authorUserAgent", "authorRegion", "authorCountry", "contentType", "url", "lang", "title", "text"]]
X_part2 = np.concatenate([
    np.hstack(df_merge['title'].values).reshape(len(df_merge),-1),
    np.hstack(df_merge['text'].values).reshape(len(df_merge),-1)
], axis=1)
X = np.concatenate([X_part1, X_part2], axis=1)
Y = df_merge[['eventType']].values
Y = np.log(Y)

```

```

In [24]: X_train, X_valid, Y_train, Y_valid = train_test_split(X, Y, test_size=1500, random_state=42)
print(X_train.shape)
print(X_valid.shape)
print(Y_train.shape)
print(Y_valid.shape)

```

```

(70769, 809)
(1500, 809)
(70769, 1)
(1500, 1)

```

```

In [27]: # Set our parameters for xgboost
        params = {}

        # 請填入以下參數:
        # 目標函數: 線性回歸
        # 評價函數: rmse
        # 學習速度: 0.01
        # 最大深度: 5
        # bst = xgboost.train(params, d_train, 3000, watchlist, early_stopping_rounds=50, verbose=1)
        #=====your works starts=====#
        params['objective'] =
        params['eval_metric'] =
        params['eta'] =
        params['max_depth'] =
        d_train =
        d_valid =
        watchlist =
        bst =
        Y_pred =
        #=====your works ends=====#

```

```

[08:14:58] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[0]      train-rmse:0.519924      valid-rmse:0.529289
Multiple eval metrics have been passed: 'valid-rmse' will be used for early stopping.

```

Will train until valid-rmse hasn't improved in 10 rounds.

```

[08:14:59] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:01] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:01] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:02] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:03] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:04] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:05] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:06] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:07] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:08] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[10]      train-rmse:0.444217      valid-rmse:0.455455
[08:15:09] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:10] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:11] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:12] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:14] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:15] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:16] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:17] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:18] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[08:15:20] C:\Users\Administrator\Desktop\xgboost\src\tree\updater_prune.cc:74: tree pruning error
[20]      train-rmse:0.396852      valid-rmse:0.409591

```


[illegible]

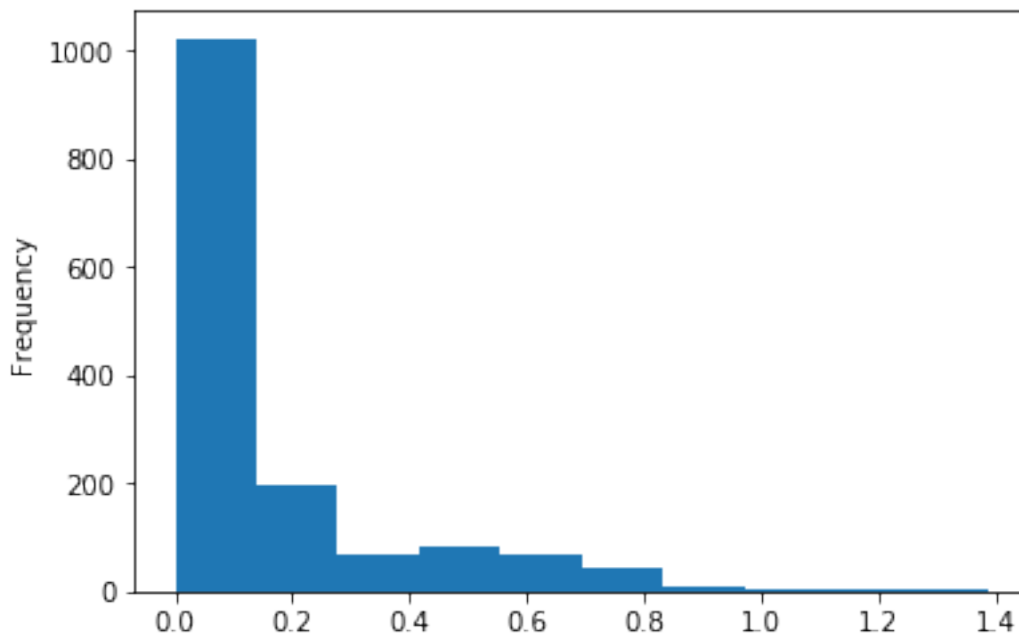
[illegible]


```
In [30]: df_result = pd.DataFrame()
Y_pred = bst.predict(xgboost.DMatrix(X_valid))
df_result['predict'] = np.exp(Y_pred)
df_result['truth'] = np.exp((list(Y_valid)))
df_result['error'] = df_result.apply(lambda x: np.abs(x['predict'] - x['truth']) / x['truth'], axis=1)
df_result_sort = df_result.sort_values('error')
df_result.head()
```

```
Out[30]:
```

	predict	truth	error
0	1.124131	1.0	0.124131
1	1.018890	1.0	0.018890
2	1.061923	1.0	0.061923
3	1.198980	1.0	0.198980
4	1.017537	1.0	0.017537

```
In [35]: df_result_sort['error'].plot('hist')
plt.show()
```



```
In [36]: # 請使用 plt.scatter, 以 0~len(df_result) 作為 x, 預測值 (黑色) 與實際值 (紅色) 作為 y。
#!=====your works starts=====!#

#!=====your works ends=====!#

plt.show()
```

