Statistical Machine Learning with Python Week #2

Ching-Shih Tsou (Ph.D.), Distinguished Prof. at the Department of Mechanical Engineering/Director of the Center for Artifical Intelligence & Data Science, Ming Chi University of Technology

August/2020 at MCUT

Association Rules Mining or Frequent Pattern Mining

- Online radio keeps track of everything you play. It uses this information for recommending music you are likely to enjoy and supports focused marketing that sends you advertisements for music you are likely to buy. Why waste scarce advertising dollars on items that customers are unlikely to purchase.
- Suppose you were given data from a music community site. For each user you may have a log of every artist he/she had downloaded to their computer. You may even have demographic information on the user (such as age, sex, location, occupation, and interests). Your objective is to build a system that recommends new music to users in this community.
- From the available information, it is usually quite easy to determine the support for (i.e., the frequencies of listening to) various individual artists, as well as the joint support for pairs (or larger groupings) of artists. All you have to do is count the incidences (0/1) across all members of your network and divide those frequencies by the number of your members. From the support we can calculate the confidence and the lift.
- For illustration we use a large data set with close to 300,000 records of song (artist) selections made by 15,000 users. Even larger data sets are available on the web (see, e.g., Celma (2010), and the data sets on his web page http://ocelma.net/MusicRecommendationDataset (http://ocelma.net/MusicRecommendationDataset)). Each row of our data set contains the name of the artist the user has listened to. Our first user, a woman from Germany, has listened to 16 artists, resulting in the first 16 rows of the data matrix. The two demographic variables listed here (gender and country) are not used in our analysis. However, it would be straightforward to stratify the following market basket analysis on gender and country of origin, and investigate whether findings change (we recommend that you do this as an exercise).
- The first thing we need to accomplish is to transform the data as given here into an incidence matrix where each listener represents a row, with 0 and 1s across the columns indicating whether or not he or she has played a certain artist.
- The last step in the program involves the construction of the association rules. We look for artists (or groups of artists) who have support larger than 0.01 (1%) and who give confidence to another artist that is larger than 0.50 (50%). These requirements rule out rare artists.

```
import pandas as pd
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
```

```
lastfm = pd.read_csv("lastfm.csv")
print(lastfm.head())
```

```
##    user                 artist sex  country
## 0     1    red hot chili peppers   f  Germany
## 1     1  the black dahlia murder   f  Germany
## 2     1                goldfrapp   f  Germany
## 3     1         dropkick murphys   f  Germany
## 4     1                 le tigre   f  Germany
```

```
print(lastfm.dtypes)
```

```
## user        int64
## artist     object
## sex        object
## country    object
## dtype: object
```

```
print(lastfm.user.value_counts()[:5])
```

```
## 17681    76
## 15057    63
## 1208     55
## 19558    55
## 13424    54
## Name: user, dtype: int64
```

```
# 15,000 users
print(lastfm.user.unique().shape)
```

```
## (15000,)
```

```
print(lastfm.artist.value_counts()[:5])
```

```
## radiohead                2704
## the beatles              2668
## coldplay                 2378
## red hot chili peppers    1786
## muse                     1711
## Name: artist, dtype: int64
```

```
# 1,004 artists
print(lastfm.artist.unique().shape)
```

```
## (1004,)
```

```
# group by user
grouped = lastfm.groupby('user')
```

```
print(list(grouped)[:2])
```

```
## [(1,     user                   artist sex  country
## 0      1    red hot chili peppers   f  Germany
## 1      1  the black dahlia murder   f  Germany
## 2      1                goldfrapp   f  Germany
## 3      1         dropkick murphys   f  Germany
## 4      1                 le tigre   f  Germany
## 5      1               schandmaul   f  Germany
## 6      1                    edguy   f  Germany
## 7      1             jack johnson   f  Germany
## 8      1                eluveitie   f  Germany
## 9      1              the killers   f  Germany
## 10     1             judas priest   f  Germany
## 11     1               rob zombie   f  Germany
## 12     1               john mayer   f  Germany
## 13     1                  the who   f  Germany
## 14     1               guano apes   f  Germany
## 15     1       the rolling stones   f  Germany), (3,     user
artist sex        country
## 16     3     devendra banhart   m  United States
## 17     3     boards of canada   m  United States
## 18     3           cocorosie   m  United States
## 19     3          aphex twin   m  United States
## 20     3    animal collective   m  United States
## 21     3           atmosphere   m  United States
## 22     3         joanna newsom   m  United States
## 23     3                  air   m  United States
## 24     3           portishead   m  United States
## 25     3       massive attack   m  United States
## 26     3  broken social scene   m  United States
## 27     3          arcade fire   m  United States
## 28     3                plaid   m  United States
## 29     3          prefuse 73   m  United States
## 30     3                  m83   m  United States
## 31     3        the flashbulb   m  United States
## 32     3             pavement   m  United States
## 33     3            goldfrapp   m  United States
## 34     3           amon tobin   m  United States
## 35     3         sage francis   m  United States
## 36     3             four tet   m  United States
## 37     3          max richter   m  United States
## 38     3             autechre   m  United States
## 39     3            radiohead   m  United States
## 40     3   neutral milk hotel   m  United States
## 41     3         beastie boys   m  United States
## 42     3           aesop rock   m  United States
## 43     3              mf doom   m  United States
## 44     3            the books   m  United States)]
```

```
#have skip number
print(list(grouped.groups.keys())[:10])
```

```
## [1, 3, 4, 5, 6, 7, 9, 12, 13, 14]
```

```
# Count the number of listeners of each user
numArt = grouped.agg({'artist': "count"})
print(numArt[5:10])
```

```
##       artist
## user
## 7         22
## 9         19
## 12        30
## 13         7
## 14         8
```

```
grouped = grouped['artist']
music = [list(artist) for (user, artist) in grouped]
print([x for x in music if len(x) < 3][:2])
```

```
## [['michael jackson', 'a tribe called quest'], ['bob marley & the wailers']]
```

```
from mlxtend.preprocessing import TransactionEncoder
te = TransactionEncoder()
txn_binary = te.fit(music).transform(music)
print(txn_binary.shape)
```

```
## (15000, 1004)
```

```
print(te.columns_[15:20])
```

```
## ['abba', 'above & beyond', 'ac/dc', 'adam green', 'adele']
```

```
df = pd.DataFrame(txn_binary, columns=te.columns_)
print(df.iloc[:5, 15:20])
```

```
##     abba  above & beyond  ac/dc  adam green  adele
## 0  False           False  False       False  False
## 1  False           False  False       False  False
## 2  False           False  False       False  False
## 3  False           False   True       False  False
## 4  False           False  False       False  False
```

```
# apriori
from mlxtend.frequent_patterns import apriori
```

```
import time
start = time.time()
freq_itemsets = apriori(df, min_support=0.01,
use_colnames=True)
end = time.time()
print(end - start)
```

```
## 25.102942943572998
```

```
freq_itemsets['length'] = freq_itemsets['itemsets'].apply(lambda x: len(x))
# support,itemsets,length
print(freq_itemsets.head())
```

```
##     support              itemsets  length
## 0  0.022733                (2pac)       1
## 1  0.030933        (3 doors down)       1
## 2  0.032800  (30 seconds to mars)       1
## 3  0.021800             (50 cent)       1
## 4  0.013667      (65daysofstatic)       1
```

```
print(freq_itemsets.dtypes)
```

```
## support     float64
## itemsets     object
## length        int64
## dtype: object
```

```
print(freq_itemsets[(freq_itemsets['length'] == 2)
& (freq_itemsets['support'] >= 0.05)])
```

```
##       support                  itemsets  length
## 921    0.0546     (coldplay, radiohead)       2
## 1503   0.0582  (the beatles, radiohead)       2
```

```
# association_rules
from mlxtend.frequent_patterns import association_rules
# confidence >= 0.5
musicrules = association_rules(freq_itemsets,
metric="confidence", min_threshold=0.5)
print(musicrules.head())
```

```
##              antecedents  consequents  antecedent support  consequent suppo
rt  \
## 0                (beck)  (radiohead)            0.057467            0.1802
67
## 1                (blur)  (radiohead)            0.033533            0.1802
67
## 2  (broken social scene)  (radiohead)            0.027533            0.1802
67
## 3               (keane)   (coldplay)            0.034933            0.1585
33
## 4         (snow patrol)   (coldplay)            0.050400            0.1585
33
##
##     support  confidence      lift  leverage  conviction
## 0  0.029267    0.509281  2.825152  0.018907    1.670473
## 1  0.017533    0.522863  2.900496  0.011488    1.718024
## 2  0.015067    0.547215  3.035589  0.010103    1.810427
## 3  0.022267    0.637405  4.020634  0.016729    2.320676
## 4  0.026467    0.525132  3.312441  0.018477    1.772002
```

```
musicrules['antecedent_len'] = musicrules['antecedents'].apply(lambda x: len
(x))
print(musicrules.head())
```

```
##              antecedents  consequents  antecedent support  consequent suppo
rt  \
## 0                (beck)  (radiohead)            0.057467            0.1802
67
## 1                (blur)  (radiohead)            0.033533            0.1802
67
## 2  (broken social scene)  (radiohead)            0.027533            0.1802
67
## 3               (keane)   (coldplay)            0.034933            0.1585
33
## 4         (snow patrol)   (coldplay)            0.050400            0.1585
33
##
##     support  confidence      lift  leverage  conviction  antecedent_len
## 0  0.029267    0.509281  2.825152  0.018907    1.670473               1
## 1  0.017533    0.522863  2.900496  0.011488    1.718024               1
## 2  0.015067    0.547215  3.035589  0.010103    1.810427               1
## 3  0.022267    0.637405  4.020634  0.016729    2.320676               1
## 4  0.026467    0.525132  3.312441  0.018477    1.772002               1
```

```
print(musicrules[(musicrules['antecedent_len'] > 0) &
(musicrules['confidence'] > 0.55)&(musicrules['lift'] > 5)])
```

```
##                   antecedents   consequents  antecedent support  \
## 8                      (t.i.)  (kanye west)            0.018333
## 12      (the pussycat dolls)     (rihanna)            0.018000
## 38  (led zeppelin, the doors)  (pink floyd)            0.017867
##
##     consequent support   support  confidence       lift  leverage  convicti
on  \
## 8             0.064067  0.010400    0.567273   8.854413  0.009225    2.1628
71
## 12            0.043067  0.010400    0.577778  13.415893  0.009625    2.2664
21
## 38            0.104933  0.010667    0.597015   5.689469  0.008792    2.2210
91
##
##     antecedent_len
## 8                1
## 12               1
## 38               2
```