Haensel AMS Homework

Paul Teehan

June 6, 2016

We are asked to solve the following task:

- Two listings of product-session pairs are provided; one for search results, and one for viewings.
- For each product that was viewed, find which three products are most often viewed or displayed in the same session.

I interpret this as follows. Suppose a product appears in m sessions across both datasets. For each session, we should collect the list of other products that also appeared in that session, either viewed or displayed, so that we have a full set of products associated with that session. Then, across each of these m sessions, we should identify the three products that appeared in the most number of sessions.

I approach this problem by constructing a similarity matrix where the value of each cell (i, j) is the number of times product i appeared in a session that also included product j. The matrix is square with each dimension equal to the total number of unique products that were either viewed or displayed. For example, consider a scenario with four products 'a', 'b', 'c', and 'd', across three sessions, as follows:

```
Session 1: 'a', 'b', 'c'
Session 2: 'b', 'c', 'd'
Session 3: 'c', 'd'
```

The similarity matrix is 4 x 4 and has the following values:

```
[1  1  1  0
 1  2  2  1
 1  2  3  2
 0  1  2  2]
```

The diagonal entries record the total number of times each product appears. The most frequently co-occuring products can be identified by reading off each row. For example, the third row corresponds to product 'c' and has entires [1, 2, 3, 2]. This indicates that across all sessions that included 'c', 'a' appeared once, 'b' twice, 'c' three times, and 'd' twice. To solve the problem, we need to construct the full similarity matrix, and then for each product, identify the three highest occuring products.

I wrote a class to handle constructing and populating the matrix and returning the output in a useful form. Here is the example output for product 12:

In [9]:
```
import pickle
import product_similarity
from product_similarity import SimilarityTable

table = pickle.load(open('similarity_table.p', 'rb'))
table._export_product(16, max_rank=3)
```

Out[9]:

	appearances	product	reference_product	opportunities	appearances_pct	rank
1282	4.0	3014	16	7.0	0.571429	1
1366	3.0	3100	16	7.0	0.428571	2
1302	3.0	3034	16	7.0	0.428571	2
1390	3.0	3124	16	7.0	0.428571	2
1383	3.0	3117	16	7.0	0.428571	2
1459	3.0	3194	16	7.0	0.428571	2
1270	3.0	3002	16	7.0	0.428571	2

In this case, product 16 appeared in seven sessions. The most similar product, 3014, appeared in four of those sessions. There were six products that tied for second with three appearances. Ties are common, especially for less common products with small numbers of appearances. Where ties result in more than three products occupying the top three ranks, I have reported the complete set including ties; whoever consumes the output can reduce to three products using the method of their choice. The full results for all products are in task3_output.csv.

As a final brief analysis, here are the ten most similar product pairs among all products that appeared at least ten times. For reference product 171, there were five products that appeared in a remarkable 90 of the 91 sessions, indicating very high similarity.

In [16]:
```
import pandas as pd
output = pd.read_csv('task3_output.csv')
output[output['appearances'] > 10].sort_values('appearances_pct', ascending=False
)[0:10]
```

Out[16]:

	Unnamed: 0	appearances	product	reference_product	opportunities	appearances_pct	rank
10429	333	90.0	587	171	91.0	0.989011	1.0
10427	176	90.0	301	171	91.0	0.989011	1.0
10431	762	90.0	1437	171	91.0	0.989011	1.0
10430	658	90.0	1216	171	91.0	0.989011	1.0
10428	200	90.0	344	171	91.0	0.989011	1.0
14398	200	118.0	344	301	122.0	0.967213	1.0
14397	333	118.0	587	301	122.0	0.967213	1.0
14399	762	117.0	1437	301	122.0	0.959016	3.0
2006	1284	15.0	3016	952	16.0	0.937500	1.0
2005	1412	15.0	3146	952	16.0	0.937500	1.0