

MMAI 5040 – Winter 2022

Business Application of AI 1

Session 5: Association Rules & Collaborative Filtering

January 31, 2022

Divinus Oppong-Tawiah, PhD.

Today's Class ...

1. Announcements
2. Association Rules
3. Collaborative Filtering
4. Practice Lab

ASSOCIATION RULES

What are Association Rules?

- Study of “what goes with what”
 - “Customers who bought X also bought Y”
 - “What symptoms go with what diagnosis”
- Transaction-based or event-based
- Also called “market basket analysis” and “affinity analysis”
- Business applications:
 - retail: store layouts & item placement, cross-selling, promotions, etc
 - medical diagnosis: what symptoms appear together
 - forensics... etc

Used in many recommender systems

Bound Away
[Last Train Home](#)



List Price: \$16.98

Price: **\$16.98** and eligible for **FREE Super Saver Shipping** on orders over \$25. [See details.](#)

Availability: Usually ships within 24 hours

Want it delivered Tomorrow? Order it in the next 4 hours and 9 minutes, and choose **One-Day S** checkout. [See details.](#)

41 used & new from **\$6.99**

▶ [See more product details](#)

[Share your own customer images](#)



Based on customer purchases, this is the #82 [Early Adopter Product in Alternative Rock](#).

801x612

Buy this title for only \$.01 when you get a new Amazon Visa® Card

Apply now and if you're approved instantly, **save \$30** off your first purchase, earn **3% rewards**, get a **0% APR,*** and pay no



Amazon Visa discount: \$30.00

Applied to this item: - \$16.97

Discount remaining: \$13.03

[Find out how](#)

[\(Don't show again\)](#)

Customers who bought this title also bought:

- [Time and Water](#) ~ Last Train Home (👁 why?)
- [Cold Roses](#) ~ Ryan Adams & the Cardinals (👁 why?)
- [Tambourine](#) ~ Tift Merritt (👁 why?)
- [Last Train Home](#) ~ Last Train Home (👁 why?)
- [True North](#) ~ Last Train Home (👁 why?)
- [Universal United House of Prayer](#) ~ Buddy Miller (👁 why?)
- [Wicked Twisted Road \[ENHANCED\]](#) ~ Reckless Kelly (👁 why?)
- [Hacienda Brothers](#) ~ Hacienda Brothers (👁 why?)

Generating Rules

Terms

“IF” part = **antecedent**

“THEN” part = **consequent**

“Item set” = the items (e.g., products) comprising the antecedent or consequent

- Antecedent and consequent are *disjoint* (i.e., have no items in common)

Tiny Example: Phone Cases

Transaction	Color(s) purchased
1	red white green
2	white orange
3	white blue
4	red white orange
5	red blue
6	white blue
7	red blue
8	red white blue green
9	red white blue
10	yellow



Many Rules are Possible

For example: Transaction 1 supports several rules, such as

- “If red, then white” (“If a red faceplate is purchased, then so is a white one”)
- “If white, then red”
- “If red and white, then green”
- + several more

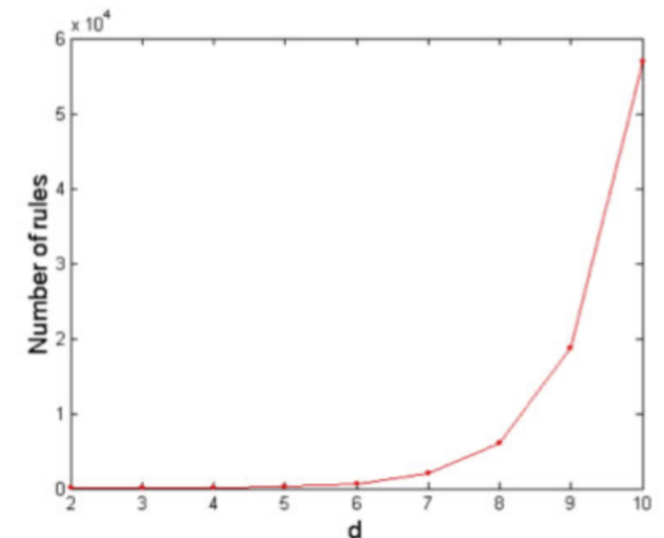
Frequent Item Sets

- Ideally, given a set of transactions, we want to create all possible combination of items
- **Problem:** computation time grows exponentially as # items increase :
 - exponential $O(3^n)$
- **Solution:** consider only “frequent item sets”
- Criterion for frequency: *support*

$$R = \sum_{k=1}^d \binom{d}{k} \sum_{i=1}^{d-k} \binom{d-k}{i}$$

$$= 3^d - 2^{d+1} + 1.$$

If $d = 6$, $R = 602$ rules
 $d = 7$, $R = 1932$ rules



Support

- *Support for an itemset* = # (or percent) of transactions that include an itemset
 - Example: support for the item set {red, white} is 4 out of 10 transactions, or 40%
- *Support for a rule* = # (or percent) of transactions that include both the antecedent and the consequent
- Any caveats with support?

Apriori Algorithm

Generating Frequent Item Sets

For k products...

1. User sets a minimum support criterion
2. Next, generate list of one-item sets that meet the support criterion
3. Use the list of one-item sets to generate list of two-item sets that meet the support criterion
4. Use list of two-item sets to generate list of three-item sets
5. Continue up through k -item sets

Measures of Rule Performance

1. **Confidence:** *the % of antecedent transactions that also have the consequent itemset*
2. **Benchmark confidence:** transactions with consequent as % of all transactions
3. **Lift Ratio** = *confidence / (benchmark confidence)*
 - ❑ *Lift > 1 indicates a rule that is useful in finding consequent items sets (i.e., more useful than just selecting transactions randomly)*
4. **Leverage** = $P(\text{antecedent AND consequent}) - P(\text{antecedent}) \times P(\text{consequent})$
 - ❑ Leverage = 0 when the two items are independent. It ranges from -1 (antecedent and consequent are antagonistic) to +1 (antecedent makes consequent more likely)

Another way to view the relationship between support and confidence:

$$\text{Support} = \hat{P}(\text{antecedent AND consequent})$$

Support: the *probability* that an antecedent and a consequent co-occur in a randomly selected transaction

$$\text{Confidence} = \frac{\hat{P}(\text{antecedent AND consequent})}{\hat{P}(\text{antecedent})} = \hat{P}(\text{consequent} \mid \text{antecedent})$$

Confidence: the *conditional probability* that a transaction selected randomly will include all the items in the consequent given that the transactions include all the items in the antecedent.

$$\text{Benchmark confidence} = \frac{\hat{P}(\text{antecedent}) \times \hat{P}(\text{consequent})}{\hat{P}(\text{antecedent})} = \hat{P}(\text{consequent})$$

Benchmark confidence: if antecedent and consequent itemsets are truly independent, what confidence values would we expect to see?

Alternate Data Format: Binary Matrix

Transaction	Red	White	Blue	Orange	Green	Yellow
1	1	1	0	0	1	0
2	0	1	0	1	0	0
3	0	1	1	0	0	0
4	1	1	0	1	0	0
5	1	0	1	0	0	0
6	0	1	1	0	0	0
7	1	0	1	0	0	0
8	1	1	1	0	1	0
9	1	1	1	0	0	0
10	0	0	0	0	0	1

Support for an itemset

Transaction	Color(s) purchased
-------------	--------------------

1	red white green
2	white orange
3	white blue
4	red white orange
5	red blue
6	white blue
7	red blue
8	red white blue green
9	red white blue
10	yellow

Support for {red, white} = 4

Support for Various Itemsets

Transaction	Color(s) purchased	{itemset} support (count)
1	red white green	{red} 6
2	white orange	{white} 7
3	white blue	{blue} 6
4	red white orange	{orange} 2
5	red blue	{green} 2
6	white blue	{red, white} 4
7	red blue	{red, blue} 4
8	red white blue green	{red, green} 2
9	red white blue	{white, blue} 4
10	yellow	{white, orange} 2
		{white, green} 2
		{red, white, blue} 2
		{red, white, green} 2

Process of Rule Selection

- Generate all rules that meet specified support & confidence
 - ❑ Find frequent item sets (those with sufficient support – see above)
 - ❑ From these item sets, generate rules with sufficient confidence

Example: Rules from {red, white, green}

{red, white} > {green} with confidence = $2/4 = 50\%$

● $[(\text{support } \{\text{red, white, green}\}) / (\text{support } \{\text{red, white}\})]$

{red, green} > {white} with confidence = $2/2 = 100\%$

● $[(\text{support } \{\text{red, white, green}\}) / (\text{support } \{\text{red, green}\})]$

Plus 4 more with confidence of 100%, 33%, 29% & 100%

If confidence criterion is 70%, report only rules 2, 3 and 6

Generating Rules in Python

Load and preprocess data set

```
fp_df = pd.read_csv('Faceplate.csv')
fp_df.set_index('Transaction', inplace=True)
print(fp_df)
```

create frequent itemsets

```
itemsets = apriori(fp_df, min_support=0.2, use_colnames=True)
```

convert into rules

```
rules = association_rules(itemsets, metric='confidence', min_threshold=0.5)
rules.sort_values(by=['lift'], ascending=False).head(6)
```

	antecedents	consequents	support	confidence	lift	leverage
14	(White, Red)	(Green)	0.2	0.5	2.500000	0.12
15	(Green)	(White, Red)	0.2	1.0	2.500000	0.12
4	(Green)	(Red)	0.2	1.0	1.666667	0.08
12	(Green, White)	(Red)	0.2	1.0	1.666667	0.08
7	(Orange)	(White)	0.2	1.0	1.428571	0.06
8	(Green)	(White)	0.2	1.0	1.428571	0.06

P(green) if you use the rule

How much better is the chance of getting a green if you use the rule than if you select randomly

Interpretation

- *Lift ratio* and *leverage* show how effective the rule is in finding consequents (useful if finding a particular consequent is important)
- *Confidence* shows the rate at which consequents will be found (useful in learning costs of promotion)
- *Support* measures overall impact
 - ❑ E.g., if support is low, it means the rule impacts a small number of transactions and may be of little use.


Caution: The Role of Chance

- Random data can generate apparently interesting association rules.
- The more rules you produce, the greater this danger.
- Rules based on large numbers of records are less subject to this danger.

Rules from some randomly-generated transactions:

	antecedents	consequents	support	confidence	lift	leverage
3	(8, 3)	(4)	0.04	1.0	4.545455	0.0312
1	(1, 5)	(8)	0.04	1.0	1.851852	0.0184
2	(2, 7)	(9)	0.04	1.0	1.851852	0.0184
4	(3, 4)	(8)	0.04	1.0	1.851852	0.0184
5	(3, 7)	(9)	0.04	1.0	1.851852	0.0184
6	(4, 5)	(9)	0.04	1.0	1.851852	0.0184

Even chance data can
produce high lift



Example: Charles Book Club

TABLE 14.7 SUBSET OF BOOK PURCHASE TRANSACTIONS IN BINARY MATRIX FORMAT

ChildBks	YouthBks	CookBks	DoItYBks	cefBks	ArtBks	GeogBks	ItalCook	ItalAtlas	ItalArt	Florence
0	1	0	1	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	1	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0
1	1	1	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Row 1, e.g., is a transaction in which books were bought in the following categories: Youth, Do it Yourself, Geography

Rules Produced by apriori

	antecedents	consequents	support	confidence	lift
64	(RefBks, YouthBks)	(ChildBks, CookBks)	0.05525	0.68000	2.80992
73	(RefBks, DoItYBks)	(ChildBks, CookBks)	0.06125	0.66216	2.73621
60	(YouthBks, DoItYBks)	(ChildBks, CookBks)	0.06700	0.64891	2.68145
80	(RefBks, GeogBks)	(ChildBks, CookBks)	0.05025	0.61468	2.54000
69	(YouthBks, GeogBks)	(ChildBks, CookBks)	0.06325	0.60526	2.50109
77	(GeogBks, DoItYBks)	(ChildBks, CookBks)	0.06050	0.59901	2.47525
67	(ChildBks, GeogBks, CookBks)	(YouthBks)	0.06325	0.57763	2.42445
70	(ChildBks, RefBks, CookBks)	(DoItYBks)	0.06125	0.59179	2.32301
49	(GeogBks, DoItYBks)	(YouthBks)	0.05450	0.53960	2.26486
62	(ChildBks, RefBks, CookBks)	(YouthBks)	0.05525	0.53382	2.24057
58	(ChildBks, CookBks, DoItYBks)	(YouthBks)	0.06700	0.52446	2.20131
56	(ChildBks, YouthBks, CookBks)	(DoItYBks)	0.06700	0.55833	2.19169
33	(ChildBks, RefBks)	(DoItYBks)	0.07100	0.55361	2.17314
74	(ChildBks, GeogBks, CookBks)	(DoItYBks)	0.06050	0.55251	2.16884
20	(ChildBks, GeogBks)	(YouthBks)	0.07550	0.51624	2.16680
46	(GeogBks, CookBks)	(YouthBks)	0.08025	0.51360	2.15572
61	(ChildBks, RefBks, YouthBks)	(CookBks)	0.05525	0.89113	2.14471
16	(ChildBks, YouthBks)	(DoItYBks)	0.08025	0.54407	2.13569
50	(RefBks, CookBks)	(DoItYBks)	0.07450	0.53309	2.09262
28	(RefBks)	(ChildBks, CookBks)	0.10350	0.50549	2.08882
72	(RefBks, CookBks, DoItYBks)	(ChildBks)	0.06125	0.82215	2.08667
15	(YouthBks)	(ChildBks, CookBks)	0.12000	0.50367	2.08129
71	(ChildBks, RefBks, DoItYBks)	(CookBks)	0.06125	0.86268	2.07624
22	(ChildBks, CookBks)	(DoItYBks)	0.12775	0.52789	2.07220
25	(DoItYBks)	(ChildBks, CookBks)	0.12775	0.50147	2.07220

Caution:

Duplication (same trio of books)

This does not mean that the rules are not useful, but rather, you can compress them and reduce the number of itemsets to be considered for possible action from a business perspective

Summary – Association Rules

- Association rules (or *affinity analysis*, or *market basket analysis*) produce rules on associations between items from a database of transactions
- Widely used in **recommender systems**
- Most popular method is **Apriori algorithm**
- To reduce computation, we consider only “frequent” item sets (=support)
- Performance of rules is measured by *confidence* and *lift*
- Can produce a profusion of rules; review is required to identify useful rules and to reduce redundancy

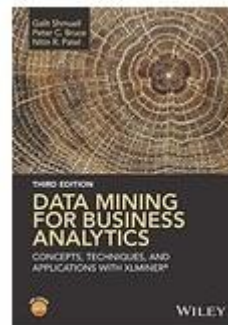
10 minutes break

COLLABORATIVE FILTERING

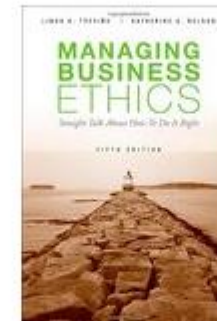
Collaborative Filtering – What is it?

- ‘Filtering’ relevant info about a specific user from the preferences of many users (‘collaborators’)
- Decision context?
- User based methods
- Item based methods

Customers Who Bought This Item Also Bought



Practical Management
Science (with Essential
Textbook Resources...
> Wayne L. Winston



Managing Business Ethics:
Straight Talk about How to
Do It Right
> Linda K. Trevino



Data Science for Business:
What You Need to Know
about Data Mining and...
> Foster Provost

Item-user matrix

- Cells are user preferences, r_{ij} , for items
- Preferences can be ratings, or binary (buy, click, like)

User ID	Item ID			
	I_1	I_2	\dots	I_p
U_1	$r_{1,1}$	$r_{1,2}$	\dots	$r_{1,p}$
U_2	$r_{2,1}$	$r_{2,2}$	\dots	$r_{2,p}$
\vdots				
U_n	$r_{n,1}$	$r_{n,2}$	\dots	$r_{n,p}$

More efficient to store as rows of triplets

- More efficient for very large transaction dataset
- Each row has the user ID, the item ID, and the user's rating of that item

$$(U_u, I_i, r_{ui})$$

User-based Collaborative Filtering

- Start with a single user who will be the target of the recommendations
- Find other users who are most similar, based on comparing preference vectors

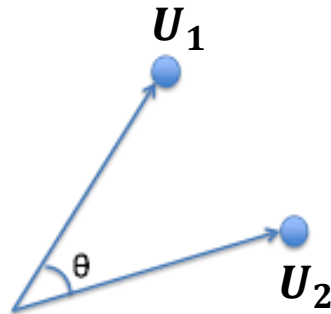
Measuring similarity – Pearson Correlation

- Like nearest-neighbor algorithm
- But Euclidean distance does not do well here
- Correlation proximity does better (Pearson)
- For each user pair, find the co-rated items, calculate correlation between the vectors of their ratings for those items
 - Note that the average ratings for each user are across all products, not just the co-rated ones

$$\text{Corr}(U_1, U_2) = \frac{\sum (r_{1,i} - \bar{r}_1)(r_{2,i} - \bar{r}_2)}{\sqrt{\sum (r_{1,i} - \bar{r}_1)^2} \sqrt{\sum (r_{2,i} - \bar{r}_2)^2}}$$

Or, ... cosine similarity

- Like correlation coefficient, except do not subtract the means



$$\text{Cosine Similarity}(\mathbf{U}_1, \mathbf{U}_2) = \cos(\theta) = \frac{\mathbf{U}_1 \cdot \mathbf{U}_2}{\|\mathbf{U}_1\| \|\mathbf{U}_2\|}$$

$$\text{Cosine Similarity}(\mathbf{U}_1, \mathbf{U}_2) = \frac{\sum(r_{1i})(r_{2i})}{\sqrt{\sum((r_{1i})^2)} \sqrt{\sum((r_{2i})^2)}}$$

Measuring similarity cont'd.

- **“Cold start” problem:** For users with just one item, or items with just one neighbor, neither cosine similarity nor correlation produces useful metric
- **Binary matrix?** Must use all the data, not just the co-rated items.
 - This can add useful info – in the Netflix contest, information about which movies users chose to rate was informative.

Example – Tiny Netflix subset customer ratings of movies

Customer ID	Movie ID									
	1	5	8	17	18	28	30	44	48	
30878	4	1			3	3	4	5		
124105	4									
822109	5									
823519	3		1	4		4	5			
885013	4	5								
893988	3						4	4		
1248029	3					2	4		3	
1503895	4									
1842128	4						3			
2238063	3									

Consider users 30878 and 823519

Correlation between users 30878 and 823519

- First find average ratings for each user:

$$\bar{r}_{30878} = (4 + 1 + 3 + 3 + 4 + 5)/6 = 3.333$$

$$\bar{r}_{823519} = (3 + 1 + 4 + 4 + 5)/5 = 3.4$$

- Find correlation using departure from avg. ratings for the co-rated movies (movies 1, 28 and 30):

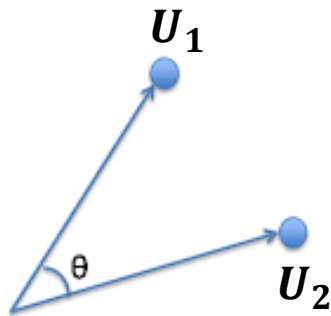
$$\text{Corr}(U_{30878}, U_{823519}) =$$

$$\frac{(4 - 3.333)(3 - 3.4) + (3 - 3.333)(4 - 3.4) + (4 - 3.333)(5 - 3.4)}{\sqrt{(4 - 3.333)^2 + (3 - 3.333)^2 + (4 - 3.333)^2} \sqrt{(3 - 3.4)^2 + (4 - 3.4)^2 + (5 - 3.4)^2}} \\ = 0.6/1.75 = 0.34$$

Cosine similarity for same users

- Use raw ratings instead of departures from averages:

$$\begin{aligned}\text{Cos Sim}(U_{30878}, U_{823519}) &= \frac{4 \times 3 + 3 \times 4 + 4 \times 5}{\sqrt{4^2 + 3^2 + 4^2} \sqrt{3^2 + 4^2 + 5^2}} \\ &= 44/45.277 = 0.972\end{aligned}$$



Ranges from 0 (no similarity) to 1 (perfect match)

Using the similarity info to make recommendations

- Given a new user, identify k-nearest users
- Consider all the items they rated/purchased, except for the co-rated ones
- Among these other items, what is the best one? “Best” could be
 - ☐ Most purchased
 - ☐ Highest rated
 - ☐ Most rated
- That “best” item is the recommendation for the new user

Item-based collaborative filtering

- When the number of users is huge, user-based calculations pose an obstacle (similarity measures cannot be calculated until user shows up)
- Alternative – when a user purchases an item, focus on similar items
 1. Find co-rated (co-purchased) items (by any user)
 2. Recommend the most popular or most correlated item
- See examples with surprise package in Python

Summary – Collaborative Filtering

- User-based – for a new user, find other users who share his/her preferences, recommend the highest-rated item that new user does not have.
 - ❑ User-user correlations cannot be calculated until new user appears on the scene... so it is slow if lots of users
- Item-based – for a new user considering an item, find other item that is most similar in terms of user preferences.
 - ❑ Ability to calculate item-item correlations in advance greatly speeds up the algorithm
 - ❑ Downside, less diversity in items than user tastes..

Association Rules vs. Collaborative Filtering

- *Frequent itemset vs personalized recommendations*
- *Transactional data vs user data*
- *Binary data and ratings data*

- ☐ AR: focus entirely on frequent (popular) item combinations. Data rows are single transactions. Ignores user dimension. Often used in displays (what goes with what).
- ☐ CF: focus is on user preferences. Data rows are user purchases or ratings over time. Can capture “long tail” of user preferences – useful for recommendations involving unusual items

Class 4 Exercise

(Submit on canvas in today's Class Participation folder. Due before next class):

- See end of Lab practice file: Class 4. Associative Rules & Collaborative Filtering

Next Class ...

- Clustering Analysis
- In-person at Room W1 33 SSB

Looking forward to meeting you!