ALGONQUIN COLLEGE

CST8390 BUSINESS INTELLIGENCE & DATA ANALYTICS

Week 7

Association Rule

Association Rule Mining

- a **rule**-based machine learning method
- Objective is to discover interesting relations between variables in large databases.
- intended to identify strong **rules** discovered in databases using some measures of interestingness.

Association Rule Mining (Cont'd….)

- important data mining model studied extensively by the database and data mining community.
- Assume all data are categorical.
- No good algorithm for numeric data.
- Initially used for Market Analysis to find how items purchased by customers are related.

Example

- analyzes and predicts customer behavior
- if a customer buys bread, there is an 80% chance to buy butter too. bread → butter
- buys {onions, potatoes} → buys{tomatoes} (likely to buy tomatoes also)
- useful for marketing activities like product promotion or product pricing

Example (Cont'd…)

Bread → Butter[20%, 45%]

If then statement ... if Bread is purchased, then Butter is purchased

- LHS is antecedent
- RHS is consequent

- Bread : Antecedent
- Butter : Consequent
- 20% : Support
- 45% : Confidence

Example (Cont'd…)

A → B[Support, Confidence]

- Support denotes probability that contains A
- Confidence denotes probability that a transaction containing A also contains B

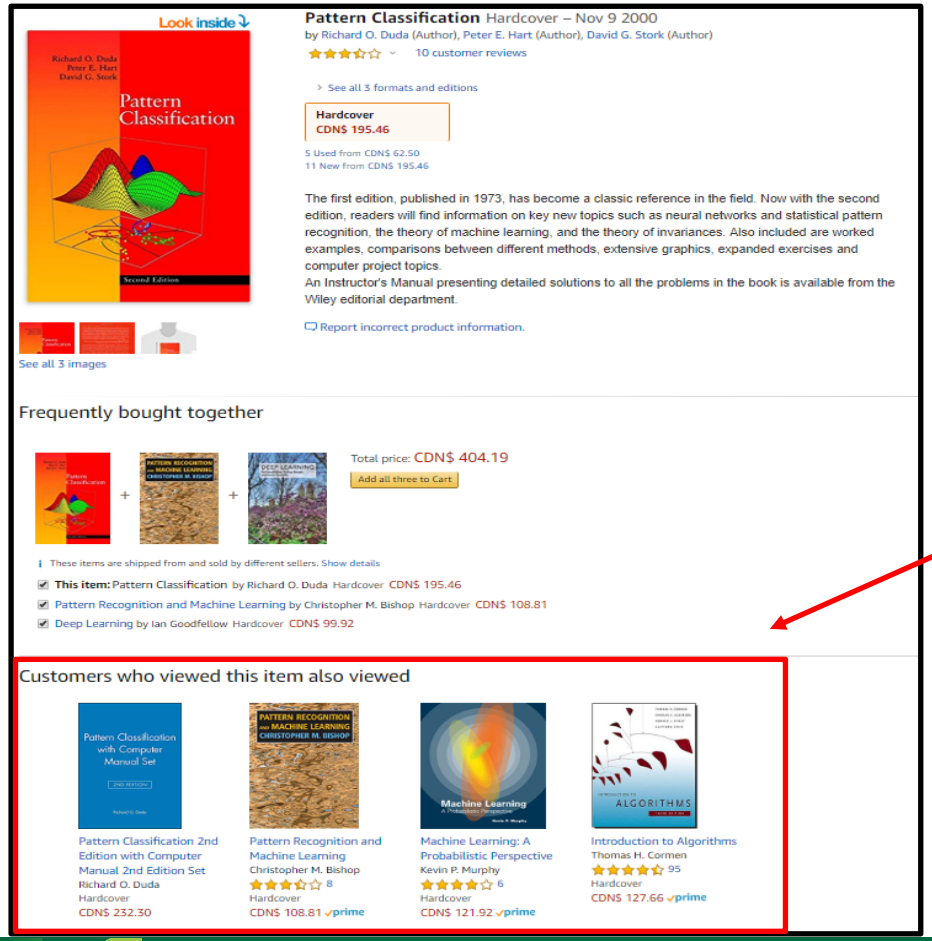
- Total Transactions in a store: 100
- Out of this 100, in 20 transactions, bread is present

- so 20/100 = 0.2 which is 20% is the **Support**

- Out of this 20 transactions, 9 has butter too.
- So, 9/20 = 0.45 = 45% which is the **Confidence**

Applications

- Web usage
- Banking
- Bioinformatics
- Market based analysis
- Credit/debit card analysis
- Product clustering
- Catalog design



Used in many recommender systems

Example 2

Transaction	Onion	Potato	Burger	Milk	Pepsi
T1	1	1	1	0	0
T2	0	1	1	1	0
T3	0	0	0	1	1
T4	1	1	0	1	0
T5	1	1	1	0	1
T6	1	1	1	1	0

Itemset I = {Onion, Potato, Burger, Milk, Pepsi}

Dataset has 6 transactions

Example rule:
{Onion, Potato} → {Burger}

Frequent Item Sets

- Ideally, we want to create all possible combinations of items
- Problem: When number of items increases, computational time increases exponentially
- Solution: Consider only “frequent item sets”
- Criteria to identify frequent items: Support

Algorithms for Association Rule

- Apriori Algorithm (the best)
- Elcat Algorithm
- F.P. Growth Algorithm

Apriori algorithm

Generating Frequent Item Sets:

- For k products:
  - User sets a minimum support criterion
  - Generate a list of one-item sets that meet the support criterion
  - Use the list of one-item sets to generate list of two-item sets that meet the support criterion
  - Use the list of two-item sets to generate list of three-item sets that meet the support criterion
  - Continue up through k-item sets

Support

$$\text{Support}(X) = \frac{\textit{Number of transactions in which X appears}}{\textit{Total number of transactions}}$$

Item	Frequency of transactions
Onion	4
Potato	5
Burger	4
Milk	4
Pepsi	2

Less than 50% which is 3.

Only 4 items are significant, which occurred greater than 50%

$$\text{Support(Onion)} = \frac{4}{6} = 0.6667$$

$$\text{Support(Potato)} = \frac{5}{6} = 0.8333$$

$$\text{Support(Burger)} = \frac{4}{6} = 0.6667$$

$$\text{Support(Milk)} = \frac{4}{6} = 0.6667$$

Support Threshold is set as 50%

Item	Frequency of transactions	Support
Onion	4	0.666666667
Potato	5	0.833333333
Burger	4	0.666666667
Milk	4	0.666666667
Pepsi	2	

Creating 2-itemset

Itemset	Frequency of transactions	Support
Onion, Potato	4	0.666666667
Onion, Burger	3	0.5
Onion, Milk	2	
Potato, Burger	4	0.666666667
Potato, Milk	3	0.5
Burger, Milk	2	

Creating 3-itemset

Itemset	Frequency of transactions	Support
Onion, Potato, Burger	3	0.5
Potato, Burger, Milk	2	

Apply self-join to create 3-itemset

→ from the 2-itemset, find two pairs with the same first item, so we get

[Onion, Potato] & [Onion Burger]. From this, we make [Onion, Potato, Burger]

Confidence

$$Conf(X \rightarrow Y) \frac{Support(X \cup Y)}{Support(X)}$$

- The confidence is 1 (maximal) for a rule X->Y if the consequent and antecedent always occur together.
- Range: [0, 1]

Lift

- $Lift(X \rightarrow Y) \frac{Support(X \cup Y)}{Support(X) * Support(Y)}$
- It measures how often X and Y happen together versus happen independently.
- If the lift is greater than 1, then there is a positive correlation.
- If the lift is less than 1, then it implies there is a negative correlation.
- If the lift is 1 then there is no relationship.
- Range: $[0, \infty]$

Results (Partial)

	Confidence	Lift
Onion --> Potato	1.00	1.2
Onion --> Burger	0.75	1.125
Potato --> Burger	0.80	1.2
Potato --> Milk	0.60	0.9
Potato --> Onion	0.80	1.2
Burger --> Onion	0.75	1.125
Burger --> Potato	1	1.2
Milk --> Potato	0.75	0.9

Demo in Weka to get full results!!!

Summary

- 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 is measured by *confidence* and *lift*

References

- http://www.saedsayad.com/association_rules.htm
- https://www.kdnuggets.com/2016/04/association-rules-apriori-algorithm-tutorial.html