Association Rule Learning

Learning Algorithm Implementations

Mining association rules

- **Standard separate-and-conquer method**
- **Looking at every possible combination of attributes, every combination of values on right-hand side**
- **Problems:**
  - Computational complexity
  - Resulting in enormous number of rules
    - pruned based on support and confidence

Association Rule Learning

- **Popular and well researched method for discovering interesting relations between variables in large datasets**
- **It is intended to identify strong rules discovered in databases using different measures of interestingness**
- **Market Basket Analyses**
    - Promotional pricing, product placement, web usage mining, intrusion detection, bioinformatics
- **Does not consider order (sequence mining)**

Item sets

- **Coverage = Support**
  - Number of instances rule predicts correctly
- **Accuracy = Confidence**
  - proportion of the number of instances that the rule applies to
- **Item: one attribute-value pair**
- **Item set: all items occurring in a rule**

Goal

- **Produce only rules that exceed pre-defined support**
  - Find all item sets with the given minimum support
  - generating rules from these item sets
- **Generate one item sets, two item sets, etc.**

Weather data example

One-item sets	Two-item sets	Three-item sets	Four-item sets
Outlook = Sunny (5)	Outlook = Sunny Temperature = Mild (2)	Outlook = Sunny Temperature = Hot Humidity = High (2)	Outlook = Sunny Temperature = Hot Humidity = High Play = No (2)
Temperature = Cool (4)	Outlook = Sunny Humidity = High (3)	Outlook = Sunny Humidity = High Windy = False (2)	Outlook = Rainy Temperature = Mild Windy = False Play = Yes (2)
...	...	...	...

Total number of item sets

- **With minimum support = 2**
  - 12 one-item sets
  - 47 two-item sets
  - 39 three-item sets
  - 6 four-item sets
  - 0 five-item sets
- **Once all item sets with minimum support have been generated they are turned into association rules**

Association rules

- **Example: 3 item set with coverage=4**

  Humidity = Normal, Windy = False, Play = Yes (4)

- **Produces seven (2N-1) potential rules:**

If Humidity=Normal and Windy=False then Play=Yes	4/4
If Humidity=Normal and Play=Yes then Windy=False	4/6
If Windy=False and Play=Yes then Humidity=Normal	4/6
If Humidity=Normal then Windy=False and Play=Yes	4/7
If Windy=False then Humidity=Normal and Play=Yes	4/8
If Play=Yes then Humidity=Normal and Windy=False	4/9
If True then Humidity=Normal and Windy=False and Play=Yes	4/12

Rules with support > 1 and confidence = 100%

	Association rule		Sup.	Conf.
1	Humidity=Normal Windy=False	⇒Play=Yes	4	100%
2	Temperature=Cool	⇒Humidity=Normal	4	100%
3	Outlook=Overcast	⇒Play=Yes	4	100%
4	Temperature=Cold Play=Yes	⇒Humidity=Normal	3	100%
...	...	...	...	...
58	Outlook=Sunny Temperature=Hot	⇒Humidity=High	2	100%

- **Total**
  - 3 rules with support four
  - 5 with support three
  - 50 with support two

Generating rules from the same item set

- **Item set**
  - Temperature = Cool, Humidity = Normal, Windy = False, Play = Yes (2)

- **Sub-sets with coverage of (2):**

Temperature = Cool, Windy = False	**(2)**
Temperature = Cool, Humidity = Normal, Windy = False	**(2)**
Temperature = Cool, Windy = False, Play = Yes	**(2)**

- **Resulting rules (coverage=2 & confidence=100%):**

Temperature = Cool, Windy = False Than Humidity = Normal, Play = Yes
Temperature = Cool, Windy = False, Humidity = Normal Than Play = Yes
Temperature = Cool, Windy = False, Play = Yes Than Humidity = Normal

How to efficiently find all frequent item sets?

- **First find one-item sets**
  - Use them to generate two-item sets
  - use two-item sets to generate three-item sets …
- **If (A B) is frequent item set then**
  - (A) and (B) have to be frequent item sets as well
- **if X is frequent k-item set than**
  - all (k-1)- item subsets of X are also frequent
  - compute k-item set by merging (k-1)-item sets

Efficient item set generation

- **Given: five three-item sets**

`(A B C), (A B D), (A C D), (A C E), (B C D)`

- **Candidate four-item sets:**
- **`(A B C D) OK because of (B C D)`**
- **`(A C D E) Not OK because of (C D E)`**
- **Second stage:**
  - take each item and generate rules - checking minimum accuracy

Summary

- **Practical issue need to generate a certain number of rules**
  - by incrementally reducing min. support required
- **ARFF format very inefficient for typical market basket data**
  - Attributes represent items in a basket and most items are usually missing