COMPSCI762: Introduction to Machine Learning

Association Rules

Jörg Simon Wicker and Katerina Taškova

The University of Auckland

THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

SCIENCE
SCHOOL OF COMPUTER SCIENCE

Motivation

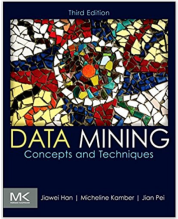
Motivation – Product Recommendation

Data Mining: Concepts and Techniques, Third Edition (The Morgan Kaufmann Series in Data Management Systems) 3rd Edition

by Jiawei Han (Author), Micheline Kamber (Author), Jian Pei (Author)

48 customer reviews

Look inside



eTextbook $22.81 - $47.12 | Hardcover $39.37 - $49.60 | Other Sellers See all 2 versions

Buy used $39.37

Buy new **$49.60**

In Stock.

Ships from and sold by Amazon.com. Gift-wrap available.

List Price: ~~$74.95~~ Save: $25.35 (34%)

13 New from $49.60

This item ships to **New Zealand**. **Want it Friday, March 16?** Order within **13 hrs 44 mins** and choose **AmazonGlobal Priority Shipping** at checkout. Learn more

Qty: 1

Add to Cart

Turn on 1-Click ordering

Ship to:

New Zealand

ISBN-13: 978-9380931913

ISBN-10: 0123814790

Why is ISBN important?

More Buying Choices

41 used & new from $39.37

13 New from $49.60 | 28 Used from $39.37

See All Buying Options

Sell yours for a Gift Card

We'll buy it for **$21.86**

Learn More

Trade in now

Have one to sell? Sell on Amazon

Add to List

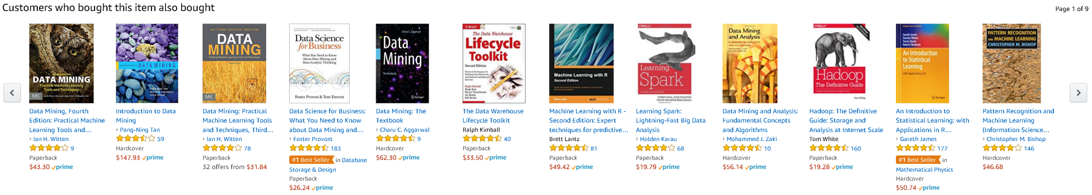
Share <Embed>

prime student College student? Get FREE shipping and exclusive deals LEARN MORE

The increasing volume of data in modern business and science calls for more complex and sophisticated tools. Although advances in data mining technology have made extensive data collection much easier, itâs still always evolving and there is a constant need for new techniques and tools that can help us transform this data into useful information and knowledge.

Motivation – Product Recommendation

Frequent itemsets: sets of items frequently 'bought' together



With this information, you could:

- Put them close to each other in the store
- Make suggestions/bundles on a website

User-Product Matrix

Column X^j gives all users that bought product j

$x_{ij} = 0$ means user i has not bought item j

$$X = \begin{bmatrix} \cdots & \cdots & \cdots & \ldots 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \ldots X_i & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \ldots 1 & \cdots \end{bmatrix}$$

Row X_i gives all items bought by user i

$x_{ij} = 1$ means user i bought item j

Clustering vs. Frequent Itemset Mining

- Clustering
  - Which **examples** are related?
  - **Grouping rows** together
- Frequent Itemset Mining
  - Which **features** “are 1” together?
  - Relating **groups of columns**

Sunglasses	Sandals	Sunscreen	Snorkel
1	1	1	0
0	0	1	0
1	1	1	0
1	1	1	1
1	0	0	0
1	1	1	1
0	0	0	0

This lecture will cover

Motivation

Association Rules

- Support and Confidence
- Upper Bound of Joint Probabilities
- Apriori

Summary

Partly based on the lecture slides from University of British Columbia CPSC340

Association Rules

Applications

- Which foods are frequently eaten together?
- Which genes are turned on at the same time?
- Which traits occur together in animals?
- Where do secondary cancers develop?
- Which traffic intersections are busy/closed at the same time?
- Which players outscore opponents together?

Support and Confidence

Itemsets

- Itemset is a collection of one or more items. E.g., {Milk, Bread, Diaper}
- K-itemset is an itemset that contains k items. E.g., 2-itemset {Milk, Bread}
- Support of an itemset S, $p(S = 1)$, is the proportion of instances that have all items in S
- How do we compute $p(S = 1)$?
  - If $S = \{bread, milk\}$, we count proportion of times they are both "1".

Bread	Eggs	Milk	Oranges
1	1	1	0
0	0	1	0
1	0	1	0
0	1	0	1

$$p(S = 1) = \frac{\#\text{times all elements of S are 1}}{n}$$

$$p(Bread = 1, Milk = 1) = \frac{2}{4}$$

Association Rules

- Consider two sets of items S and T: E.g., $S =$ milk, eggs and T $=$ bread.
- We can also consider association rules $S \rightarrow T$:
  If you buy all items in S, you are likely to also buy all items in T
  E.g., if you buy milk and eggs, you are likely to buy bread.
  - Interpretation in terms of conditional probability:
    - The rule $S \rightarrow T$ means that
      $p(T = 1|S = 1) = p(T1 = 1, T2 = 1, \ldots, Tt = 1|S1 = 1, S2 = 1, \ldots Sc = 1)$ is 'high'
    - Association rules are **directed**: $p(T|S) \neq p(S|T)$
    - Association rules are **not necessarily causal**
      E.g., buying bread doesn't necessarily imply buying milk/eggs:
    - The correlation could be due to a **common cause**
      E.g., the common cause is that you are going to cook breakfast

Scoring and Learning Associations Rules

We "score" rule $S \to T$ by "support" and "confidence".

- Support
  - How often does S happen?
  - How often were milk and eggs bought together?
  - **Marginal probability:** $p(S = 1)$
- Confidence
  - When S happens, how often does T happen?
  - When milk and eggs were bought, how often was bread bought?
  - **Conditional probability:** $p(T = 1|S = 1) = \frac{p(S,T)}{p(S)}$
- Association rule learning problem

  Given support 's' and confidence 'c', output all rules with support at least 's' and confidence at least 'c'.

Example

Customer	Fantastic Mister Fox	Fight Club	Oldboy	Lady Vengeance	...
#1	1	1	1	0	...
#2	1	1	1	1	...
#3	0	1	1	1	...
#4	0	1	0	1	...
#5	1	1	1	0	...
#6	0	0	1	1	...

- What is the support of $X = \{FightClub\}$?
  - $p(X) = 5/6$
- What is the support of $X \to Y$ with $X = \{Oldboy, LadyVengeance\}$, $Y = \{FightClub\}$? What is the confidence?
  - $p(X, Y) = 2/6$
  - $p(X|Y) = \frac{p(X,Y)}{p(Y)} = \frac{2/6}{5/6} = \frac{2}{5}$

Challenge in Learning Association Rules

- Frequent itemset goal (given a support threshold s)
  - Find all sets S with $p(S = 1) \geq s$
  - And/or all rules with minimum confidence c
- **Challenge**: with d features there are $2^d - 1$ possible sets
  - For $d = 4$
    $\{1\}\{2\}\{3\}\{4\}\{1,2\}\{1,3\}\{1,4\}\{2,3\}\{2,4\}\{3,4\}\{1,2,3\}\{1,2,4\}\{1,3,4\}\{2,3,4\}\{1,2,3,4\}$
- It takes too long to even write all sets unless d is tiny
- Can we avoid testing all sets?
  - Yes, using a basic property of probabilities
    - downward-closure/anti-monotonicity

Upper Bound of Joint Probabilities

Upper Bound on Joint Probabilities

- Suppose we know that $p(S = 1) \geq s$
- Can we say anything about $p(S = 1, A = 1)$?
  - Probability of buying all items in S, plus another item A
- Yes, $p(S = 1, A = 1)$ cannot be bigger than $p(S = 1)$
  - By the product rule we have $p(S = 1, A = 1) = p(A = 1|S = 1)p(S = 1) \leq p(S = 1)$
- E.g., probability of rolling 2 sixes on 2 dice ($1/36$) is less than 1 six on one die ($1/6$)

Support Set Pruning

- This property means that $p(S = 1) < s$ implies $p(S = 1, A = 1) < s$
  - If $p(milk = 1) < 0.1$, then $p(milk = 1, eggs = 1) < 0.1$
  - We **never consider** $p(S = 1, A = 1)$ if $p(S = 1)$ has **low support**
- Anti-monotonicity[1] property: Any non-empty subset of a frequent itemset must be frequent
  - If $\{FightClub, Oldboy, LadyVengeance\}$ is frequent, so is $\{FightClub, Oldboy\}$
  - That is, every instance having $\{FightClub, Oldboy, LadyVengeance\}$ also contains $\{FightClub, Oldboy\}$

[1] Monotic in the context of failing the frequency(support) test: if an itemset S fails the test then all supersets (itemset that contain all items in S plus additional items not in S) will fail the test as well

Apriori

Apriori Algorithm

- Apriori algorithm for finding all subsets with $p(S = 1) \geq s$
  1. Generate list of all sets S that have a size of 1
  2. Set $k = 1$
  3. Prune candidates S of size k where $p(S = 1) < s$
  4. Add all sets of size $(k + 1)$ that have all subsets of size k in current list
  5. Set $k = k + 1$ and go to 3

Apriori – Example with $min\_support = 2$

Data

Tid	Items
1	A, C, D
2	B, C, E
3	A, B, C, E
4	B, E

$\xrightarrow{\text{1st scan}}$ C_1

Itemset	support
$\{A\}$	2
$\{B\}$	3
$\{C\}$	3
$\{D\}$	1
$\{E\}$	3

$\longrightarrow$ L_1

Itemset	support
$\{A\}$	2
$\{B\}$	3
$\{C\}$	3
$\{E\}$	3

$\longrightarrow$ C_2

Itemset
$\{A, B\}$
$\{A, C\}$
$\{A, E\}$
$\{B, C\}$
$\{B, E\}$
$\{C, E\}$

2nd scan

C_2

Itemset	support
$\{A, B\}$	1
$\{A, C\}$	2
$\{A, E\}$	1
$\{B, C\}$	2
$\{B, E\}$	3
$\{C, E\}$	2

$\longleftarrow$ L_2

Itemset	support
$\{A, C\}$	2
$\{B, C\}$	2
$\{B, E\}$	3
$\{C, E\}$	2

$\longleftarrow$ C_3

Itemset
$\{B, C, E\}$[2]

$\xleftarrow{\text{3rd scan}}$ L_3

Itemset	support
$\{B, C, E\}$	2

[2] $C_3 = \{\{B, C, E\}, \{A, B, C\}, \{A, B, E\}, \{A, C, E\}\}$, but as "all nonemty subsets of a frequent itemset must also be frequent", we filter the last 3 itemsets (containg non-frequent subsets $\{A, B\}$ and $\{A, E\}$) before the 3rd scan.

Pseudo-Code

C_k: Candidate itemset of size k
L_k: frequent itemset of size k
$L_1 = \{\text{frequent 1-items}\}$

for $(k = 1; L_k \neq \{\}; k++)$ **do**
  C_{k+1} =candidates generated from L_k
  foreach transaction $t \in database$ **do**
    increment the count of all candidates in C_{k+1} that are contained in t
  end
  L_{k+1} = candidates in C_{k+1} with $support \geq min\_support$
end
return $\cup_k L_k$

Apriori

- How to generate candidates?
  - Step 1: self-joining L_k
  - Step 2: pruning
- Example of Candidate-generation
  - $L_3 = \{abc, abd, acd, ace, bcd\}$
  - Self-joining: $L_3 * L_3$
  - $abcd$ from abc and abd
  - $acde$ from acd and ace
- Pruning:
  - $acde$ is removed because?
    - ade is not in L_3
  - $C_4 = \{abcd\}$

Main Ideas

- Each iteration consists of two phases
  - Candidate formation
  - Candidate testing (database scan)
- Minimize database scans
- Avoid unnecessary tests on the database (test only those patterns that can, knowing the previous levels, be frequent)

Aprioiri Discussion (bonus slide)

- Some implementations prune the output
  - Maximal frequent subsets
    - Only return sets S with $p(S = 1) \geq s$ where no superset S' has $p(S' = 1) \geq s$
    - E.g., don't return $\{break, milk\}$ if $\{bread, milk, diapers\}$ also has high support
- Number of rules we need to test is hard to quantify
  - Need to test more rules for small s
  - Need to test more rules as average $\#$items per example increase
- Computing $p(S = 1)$ if S has k elements costs $O(nk)$
  - But there is some redundancy
    - Computing $p(\{1, 2, 3\})$ and $p(\{1, 2, 4\})$ can re-use some computation
  - Hashing can be used to speed up various computations

Summary

Summary

- Association rule mining searches relationships among the features
- Support: measure of how often we see an item.
- Frequent itemsets: sets of items with sufficient support.
- Apriori algorithm: finds itemsets by exploiting the anti-monotonicity property of minimum support data sets

Literature

- Chapter 6 of Han's *Data Mining: Concepts and Techniques*

THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

SCIENCE
SCHOOL OF COMPUTER SCIENCE

Thank you for your attention!

https://ml.auckland.ac.nz