Association Analysis

Vishal Patel

Spring 2024

Course Outline

1. ~~Introduction~~
2. ~~The Data Science Process~~
3. ~~Supervised Learning~~
4. **Unsupervised Learning**
5. The Grunt Work
6. Closing Thoughts

https://www.linkedin.com/pulse/tale-beer-diapers-ibeukp/

Association Analysis

Association Analysis

Also known as:

1. Association Rule Mining
2. Frequent Itemset Mining
3. Link Analysis
4. Market Basket Analysis

Product Associations

If someone did X,

does that make the customer

more likely to do Y?

Examples: Product bundle analysis, web log analysis, catalogue design, cross-sell, etc.

Basic Concept

Given a dataset of transactions,

where each transaction

is **a list of items**

purchased by a customer…

… **find** all rules

that correlate one item (or itemset)

with another item (or itemset).

ANTECEDENT CONSEQUENT

{Coffee, ...} → {Bagels}

- **Bagels as the consequent:** Can be used to determine what should be done to boost its sales
- **Coffee as the antecedent:** Can be used to see which products would be affected if the store discontinues selling coffee
- **Antecedent + Consequent:** Can be used to see which items can be bundled together to create a combo offer

Table 6.4. Association rules extracted from the 1984 United States Congressional Voting Records.

Association Rule	Confidence
{budget resolution = no, MX-missile=no, aid to El Salvador = yes } $\longrightarrow$ {Republican}	91.0%
{budget resolution = yes, MX-missile=yes, aid to El Salvador = no } $\longrightarrow$ {Democrat}	97.5%
{crime = yes, right-to-sue = yes, physician fee freeze = yes} $\longrightarrow$ {Republican}	93.5%
{crime = no, right-to-sue = no, physician fee freeze = no} $\longrightarrow$ {Democrat}	100%

Two Primary Measures

1. Support
2. Confidence

Support

$$support\ (A \rightarrow B) = support\ (A \cup B)$$

Proportion of transactions that contain the item or itemset

Typically, $support$ is used to measure the abundance or frequency of an itemset.

An itemset is referred to as a "**frequent itemset**" if its $support$ is larger than a specified minimum-$support$ threshold.

Loosely answers the question: **Does the rule occur by chance?**

Confidence

$$confidence\ (A \rightarrow B) = \frac{support\ (A\ \cup B)}{support\ (A)}$$

The probability of seeing the consequent in a transaction
given that it contains the antecedent.

Notes: **(1)** The $confidence$ for an itemset is equal to 1 if those items always occur together.

(2) $confidence\ (A \rightarrow B) \neq confidence\ (B \rightarrow A)$

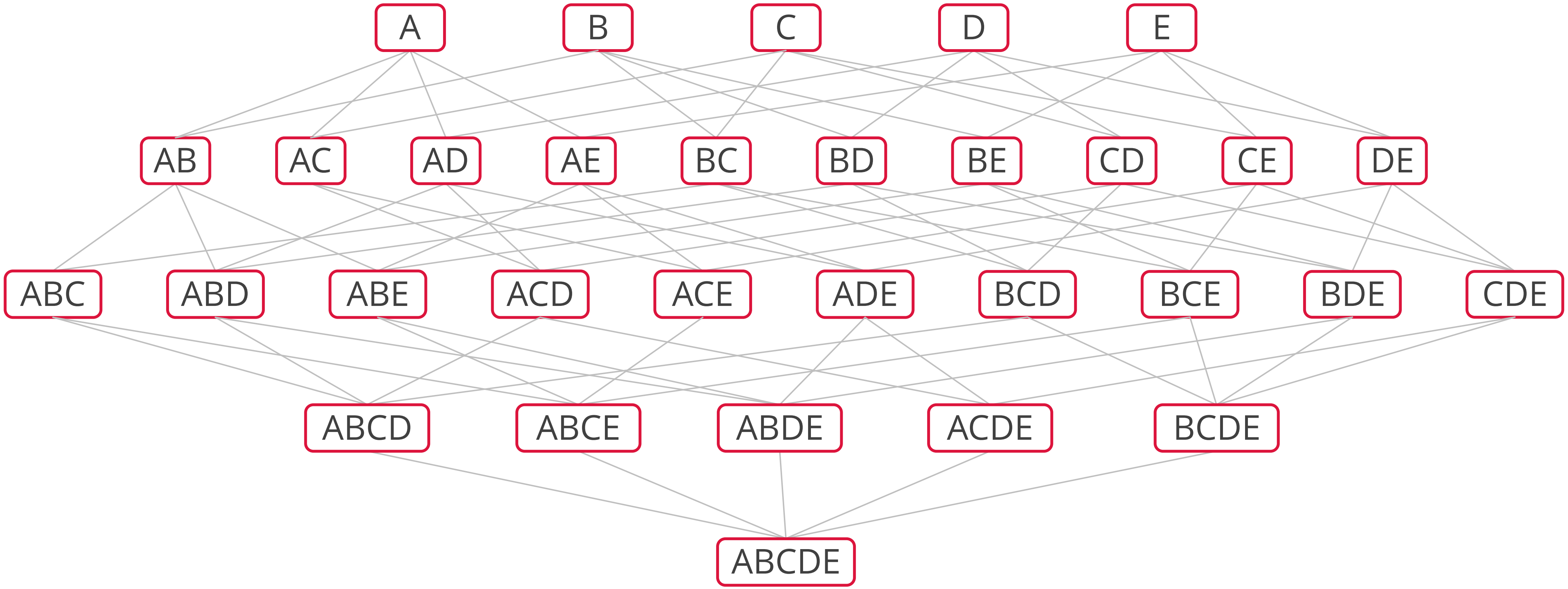
Loosely answers the question: **How strong is the association?**
Provides an estimate of $p(B \mid A)$.

Binary Representation

Order ID	Items Purchased
1	Milk, Onion, Nutmeg, Kidney Beans, Eggs, Yogurt
2	Dill, Onion, Nutmeg, Kidney Beans, Eggs, Yogurt
3	Milk, Apple, Kidney Beans, Eggs
4	Milk, Unicorn, Corn, Kidney Beans, Yogurt
5	Corn, Onion, Onion, Kidney Beans, Ice cream, Eggs

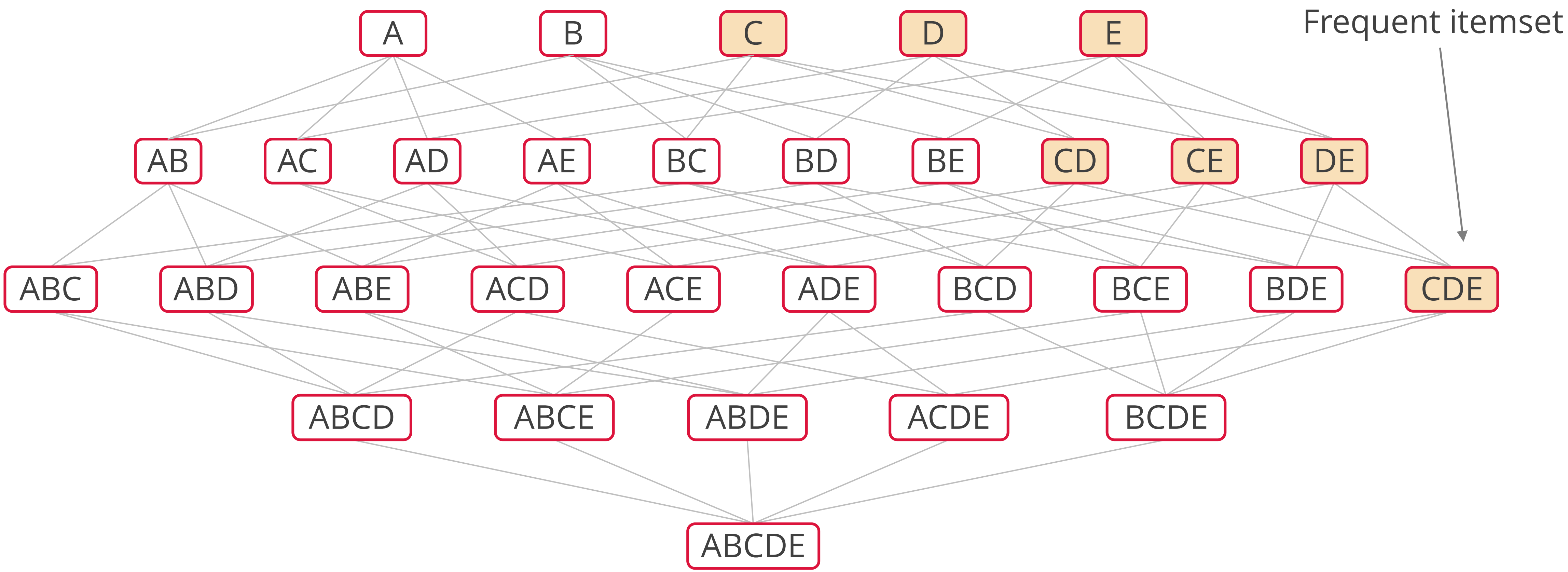
Order ID	Apple	Corn	Dill	Eggs	Ice Cream	Kidney Beans	Milk	Nutmeg	Onion	Unicorn	Yogurt
1	0	0	0	1	0	1	1	1	1	0	1
2	0	0	1	1	0	1	0	1	1	0	1
3	1	0	0	1	0	1	1	0	0	1	0
4	0	1	0	0	1	1	1	0	0	0	1
5	0	1	0	1	1	1	0	0	1	0	0

Brute-force Approach



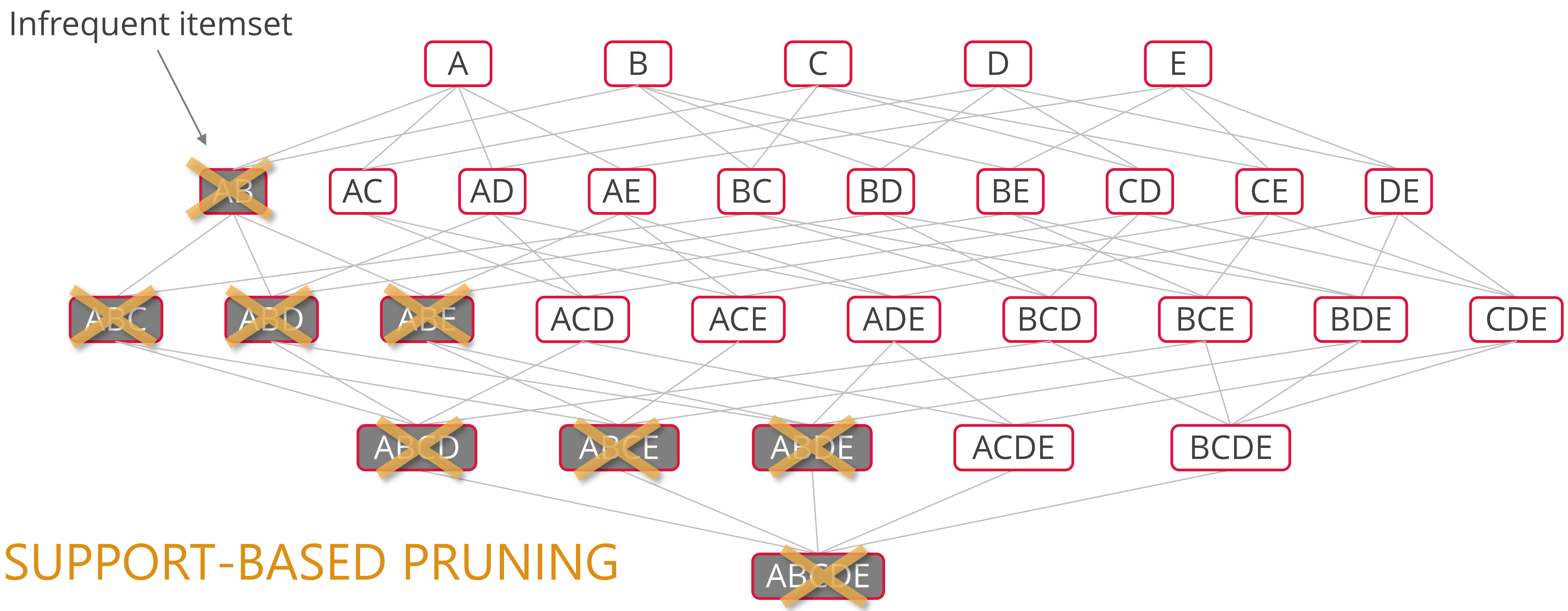
$$C(5,1) + C(5,2) + C(5,3) + C(5,4) + C(5,5) = 5 + 10 + 10 + 5 + 1 = 31$$

Apriori Principle #1



Apriori Principle #1: If an itemset is frequent (e.g., CDE), then all subsets of this itemset are also frequent.

Apriori Principle #2



Apriori Principle #2: If an itemset is infrequent (e.g., AB), then all subsets of this itemset are also infrequent.

TID	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Soda}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Soda}

Minimum $support$ = 60% (i.e., count=3)

If every subset is considered: $C(6,1) + C(6,2) + C(6,3) = 41$

With $support$-based pruning: $6 + 6 + 1 = 13$

Candidate 1-Itemsets

	Item	Count
	Beer	**3**
	Bread	**4**
✗	Soda	2
	Diapers	**4**
	Milk	**4**
✗	Eggs	1

Candidate 2-Itemsets

	Item	Count
✗	{Beer, Bread}	2
	{Beer, Diapers}	**3**
✗	{Beer, Milk}	2
	{Bread, Diapers}	**3**
	{Bread, Milk}	**3**
	{Diapers, Milk}	**3**

Candidate 3-Itemsets

	Item	Count
✗	{Bread, Diapers, Milk}	2

✗ Itemsets removed because of low $support$

Frequent itemset generation using the Apriori algorithm

Why Support and Confidence Are not Enough

$$support\;(\;\rightarrow\;) = \frac{15}{100} = 15\%$$

$$confidence\;(\;\rightarrow\;) = \frac{15}{20} = 75\%$$

If we know that someone drinks tea, then we have 75% confidence that she also drinks coffee.

However, 90% of all people drink coffee *regardless* of whether they drink tea of not!

Hence, knowing that someone drinks tea should *lower* our expectation that she also drinks coffee. This is why we need to calculate the **lift**.

			Total
	15	5	20
	75	5	80
Total	90	10	100

Evaluating Association Rules

$$lift(A \rightarrow B) = \frac{confidence\ (A\ \rightarrow B)}{support\ (B)}$$

The $lift$ metric is commonly used to measure how much more often the antecedent and consequent of a rule occur together than we would expect if they were statistically independent.

Lift	Interpretation
$= \mathbf{1}$	The items are **independent**
$< \mathbf{1}$	The items have an **inverse** relationship
$> \mathbf{1}$	The items have a **positive** relationship

Two Challenges

1. Computationally expensive
2. Potentially spurious patterns

Factors to Consider

1. ***Support* threshold (*support*-based pruning):** Lowering the *support* threshold results in more itemsets being declared as "frequent". Start with a higher *support* threshold and then lower if necessary.

2. **Number of items (dimensionality):** Number of total items under consideration directly impacts the computational requirement. Consider excluding some items that might be irrelevant. Also, consider **product hierarchies** to reduce the number of combinations.

3. **Number of transactions**: The Apriori algorithm makes repeated passes over the dataset. Consider dropping transactions either prior to the analysis or during the analysis based on initial results.

4. **Transaction width (average basket size):** The maximum number of items in a transaction also impacts the computational complexity. Discarding some items or some transactions can help.

```
class mlxtend.frequent_patterns.
association_rules (
    df,
    metric='confidence',
    min_threshold=0.8,
    support_only=False)
```

Generates a DataFrame of association rules including the metrics 'score', 'confidence', and 'lift'

http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/#api

```
class mlxtend.frequent_patterns.
association_rules (
    df,
    metric='confidence’,
    min_threshold=0.8,
    support_only=False)
```

Metric to evaluate if a rule is of interest.

Automatically set to 'support' if support_only=True.

Otherwise, supported metrics are 'support', 'confidence', 'lift’, 'leverage', and 'conviction’.

```
class mlxtend.frequent_patterns.
association_rules (
    df,
    metric='confidence',
    min_threshold=0.8,
    support_only=False)
```

Minimal threshold for the evaluation metric, via the metric parameter, to decide whether a candidate rule is of interest.

```
class mlxtend.frequent_patterns.
association_rules (
    df,
    metric='confidence',
    min_threshold=0.8,
    support_only=False)
```

Only computes the rule support and fills the other metric columns with NaNs.

This is useful if:

a) the input DataFrame is incomplete, e.g., does not contain support values for all rule antecedents and consequents

b) you simply want to speed up the computation because you don't need the other metrics.

→ → → Association Analysis Tutorial

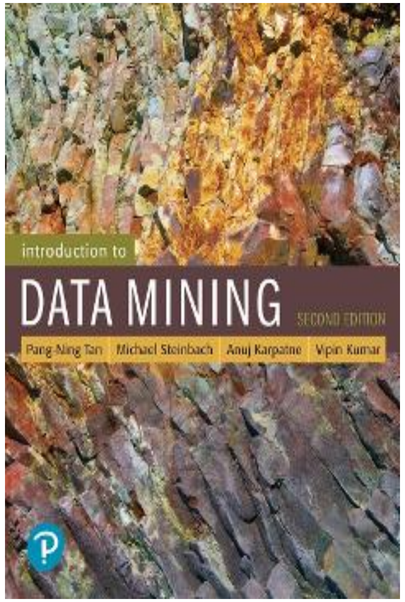
16_apriori.ipynb

Install mlxtend package first.

Installing Python Packages

- Open Anaconda Prompt (or Power Prompt) and install a conda package using this command: `conda install package-name`
- If needed, install a specific version of a conda package using this command: `conda install package-name=2.3.4`
- For installing Python packages, `conda` and `pip` are considered nearly identical. (More on that here.)
- To view all packages installed on your computer: `conda list`
- To check the version of a specific package: `pip show package-name`. This will also display the location of that package on your computer.
- To remove a previously installed package: `conda remove package-name`
- To check Anaconda or Python version: `python -V` or `conda -V`.

Further Reading



Chapter 5. **Association Analysis**: *Basic Concepts and Algorithms*

mlxtend Documentation:
http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/#overview

Recommender Systems

Recommender Systems: Applications

1. Movies
2. Videos
3. Music
4. Apps
5. News
6. Books
7. Friends
8. …

Recommender Systems

The goal is to **generate** useful **recommendations** to users for items that might interest them.

"I don't know what I am looking for."

Information Retrieval

The goal is to **obtain** relevant **information** based on the requirements (query) described by users.

"I know what I am looking for."

The Goals of a Recommender System

Products

Customers	A	B	C
1			✓
2			✓
3			✓
4		✓	

Recommend the best product for each customer

Products

Customers

	A	B	C
1			
2			✓
3	✓	✓	✓
4	✓	✓	

Select the best customers for each product

Basic Models

1

Based on

user-item **interactions**

E.g., ratings or buying behavior

COLLABORATIVE FILTERING (CF) METHODS

2

Based on

user and item **attributes**

E.g., textual profiles, keywords

CONTENT-BASED RECOMMENDER METHODS

Collaborative Filtering

- One of the most widely used techniques for recommender systems
- Applicable across a wide range of industries
- Does not require much domain knowledge
- Recommender Systems rely on **user ratings**
- Ratings are not always available
- Ratings can be **explicit** (likes, star ratings) or **implicit** (clicks, views, spend)

Collaborative Filtering

1

User based

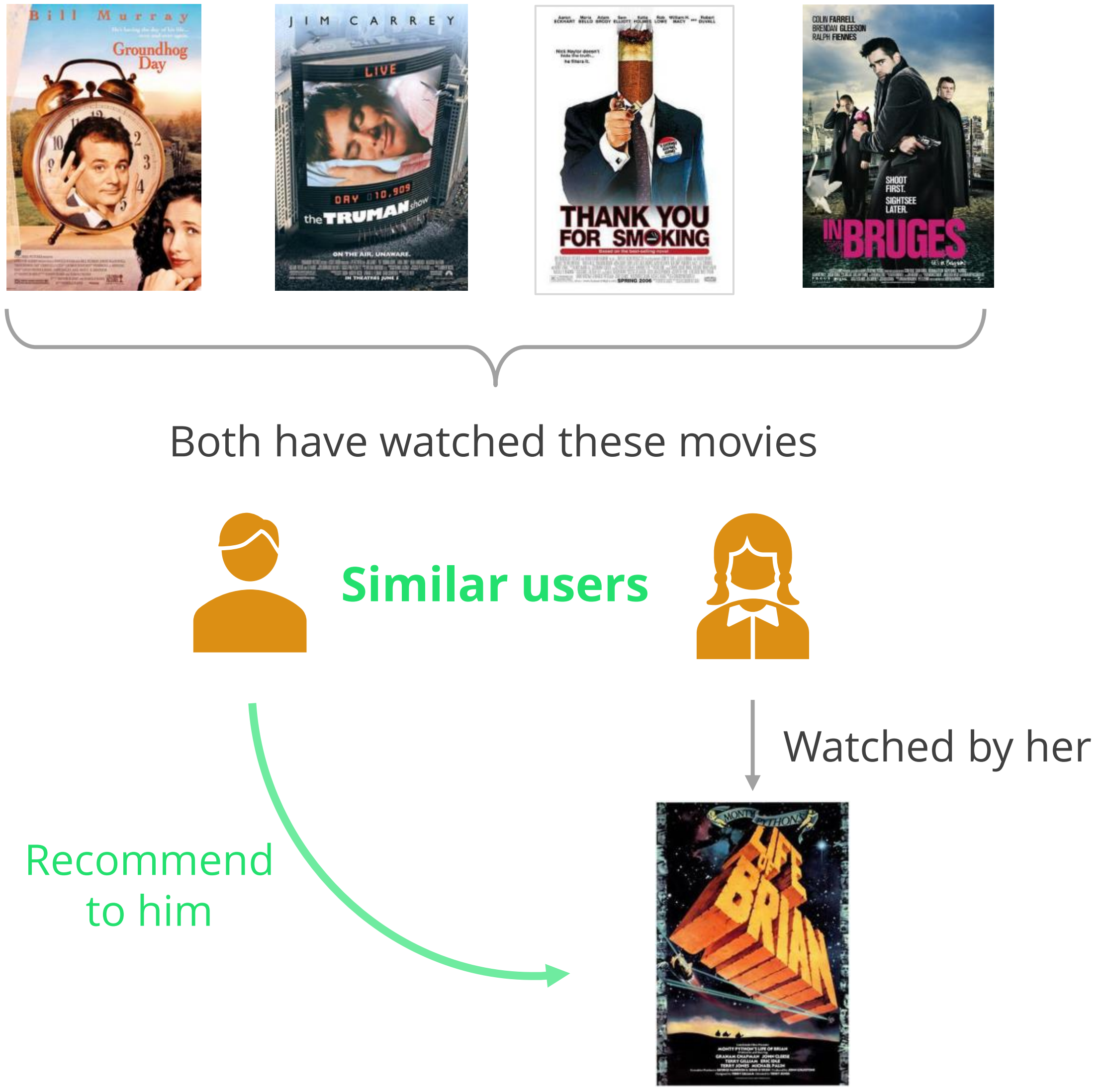
2

Item based

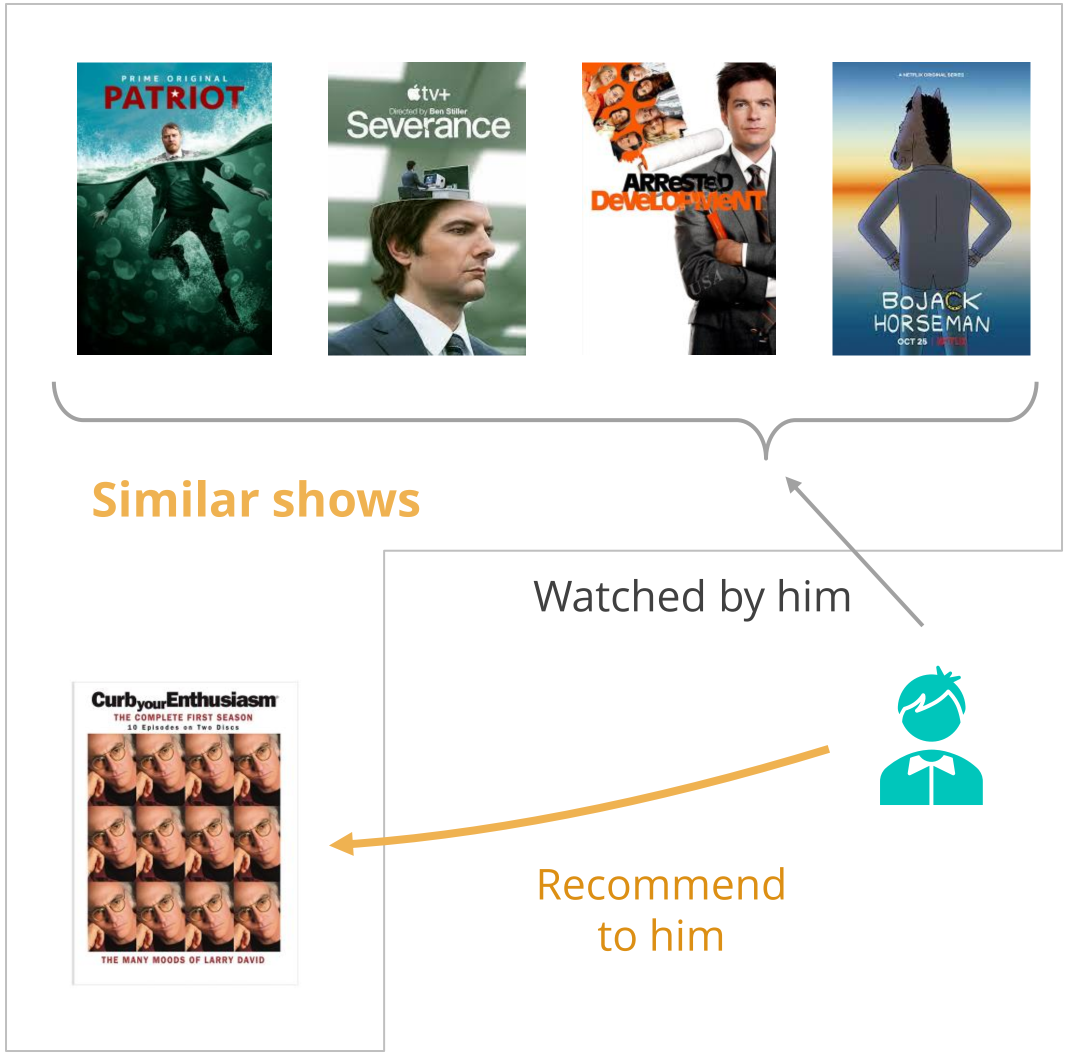
Collaborative Filtering

1

User based



Collaborative Filtering



2

Item based

Collaborative Filtering

1

User based

Uses the similarity between target users and other users

2

Item based

Uses the similarity between the target items and other items

Collectively, these two approaches are known as **memory-based methods.**

User-based

Item Ratings

Users

	A	B	C	D	E
1	5	3	4	4	?
2	3	1	2	3	3
3	4	3	4	3	5
4	3	3	1	5	4
5	1	5	5	2	1

Let's use Pearson's correlation coefficient to measure *similarity*.

$similarity(1,2) = 0.85$

$similarity(1,3) = 0.71$

$similarity(1,4) = 0.00$

$similarity(1,5) = -0.80$

Naïve approach: Find k most similar users and use the average from their ratings.

User-based

Item Ratings

Users

	A	B	C	D	E
1	5	3	4	4	**?**
2	3	1	2	3	3
3	4	3	4	3	5
4	3	3	1	5	4
5	1	5	5	2	1

$$\hat{r}(a,j) = \frac{\sum_{b \in N} r(b,i)}{|N|}$$

$N = k \text{ most similar neighbors}$

$r(a,i) = Rating\ for\ item\ i\ by\ user\ a$

Naïve approach: Find k most similar users and use the average from their ratings.

User-based

Item Ratings

Users	A	B	C	D	E
1	5	3	4	4	**4**
2	3	1	2	3	3
3	4	3	4	3	5
4	3	3	1	5	4
5	1	5	5	2	1

$$\hat{r}(1,E) = \frac{3+5}{2} = 4$$

$similarity(1,2) = 0.85$

$similarity(1,3) = 0.71$

$similarity(1,4) = 0.00$

$similarity(1,5) = -0.80$

Naïve approach: Find k most similar users and use the average from their ratings.

Improvements

- Use weighted averages.
  - Assign more weights to more similar neighbors.
- Use other similarity measures, e.g., cosine similarity.
- Consider the number of co-rated items.
- Consider agreement on rare items as more important.
- Etc.

Item-based

Item Ratings

Users

	A	B	C	D	E
1	5	3	4	4	**4.5**
2	3	1	2	3	3
3	4	3	4	3	5
4	3	3	1	5	4
5	1	5	5	2	1

$$\hat{r}(1, E) = \frac{5+4}{2} = 4.5$$

Item-based approach is computationally more efficient because

$$number\ of\ items\ \ll\ number\ of\ users$$

Naïve approach: Find k most similar **items** and use the average from their ratings.

Issues with user or item-based approaches

1 Scalability

2 Sparsity

Latent Factor Model

Singular Value Decomposition

Create latent factors of users and items and use those to predict ratings.

Other issues (with CF)

1 Cold Start

- How to recommend new items?
- What to recommend to new users?
- There needs to be a sufficient number of users to be able to make predictions based on similarity.

2 Popularity Bias

- Collaborative Filtering (CF) tends to recommend popular items.

Alternatives to CF

Clustering

1. Cluster similar users
2. Non-personalized predictions ("popularity") for each cluster

Association Rules

"Customers who bought X also bought..."

Classifiers

Multi-class / Multi-label classification models

→→→ Collaborative Filtering Tutorial

17_collab_filtering.ipynb