Data Mining Principles

Dr. Gizem Agar

Instructor, MS-ADS, University of Chicago

Analytics Manager, Caterpillar Inc.

Lecture 1

- Course Overview/Meeting
- Python and Project discussion
- Introduction to Data Mining Principles

Lecture 2

- Clustering
  - Intro (C205)
  - Calculating distance
  - KMeans (example: Utilities dataset, C220)
    - Elbow method for finding k
    - Cluster validation metrics (within-cluster dispersion, cluster separation, silhouette coefficient)
- In-class coding challenge: KMeans on Digits dataset (C225)

Lecture 3

- Review Student Work: KMeans on Digits dataset (C225)
- Clustering
  - Hierarchical Clustering (example: Utilities dataset, C230)
    - Dendrograms
    - Cluster maps
  - Density-based clustering: DBSCAN (C320, synthetic blobs and moons data)
  - Clustering validity

Lecture 4

- Dimension Reduction
  - PCA (C340 Cereal dataset)
    - Normalization BONUS: C350, Center Scale code examples
- In-class coding challenge:
  - PCA (C345 Cancer dataset)
- Supervised Learning: Regression

- Due:
  - Assignment 1B, Hierarchical Clustering
  - 1-pager (linear regression)
  - Project submission 1

Lecture 5

- Linear, Multiple Linear Regression, and Regularization
- Classification and Logistic Regression discussion

- Due:
  - Assignment 2
  - 1-pager (logistic regression)
  - Project submission 1 – any updates as needed

Lecture 6

- Continue on classification discussion
- Decision and Regression Trees
  - Decision Trees – C520 DecisionTrees Examples jupyter notebook

Due:

- 1-pager cheat sheet on ensemble methods
- Assignment 3, Feb 19

Lecture 7

- Ensemble Methods: Random Forest and Boosted Trees
  - Code examples: Ensemble Methods, Ensemble_PersonalLoanExample
  - Compare with DecisionTree_PersonalLoanExample
- Support Vector Machines (SVM)

Due:

- 1-pager cheat sheet on SVM (including definition and cost function)
- Assignment 3, Feb 19

Lecture 8

- Neural Networks
  - C810_NN_TinyData.ipynb
  - C815_NN_Accidents.ipynb
- Association Rule Mining
- Recommendation Systems

Neural Networks

What are neural networks?

- Flexible data-driven (blackbox) method
- Classification, regression, and feature extraction
- Basis of deep learning (image and voice recognition applications)
- Have capacity to generalize
- High predictive performance
- Danger of overfitting

The network

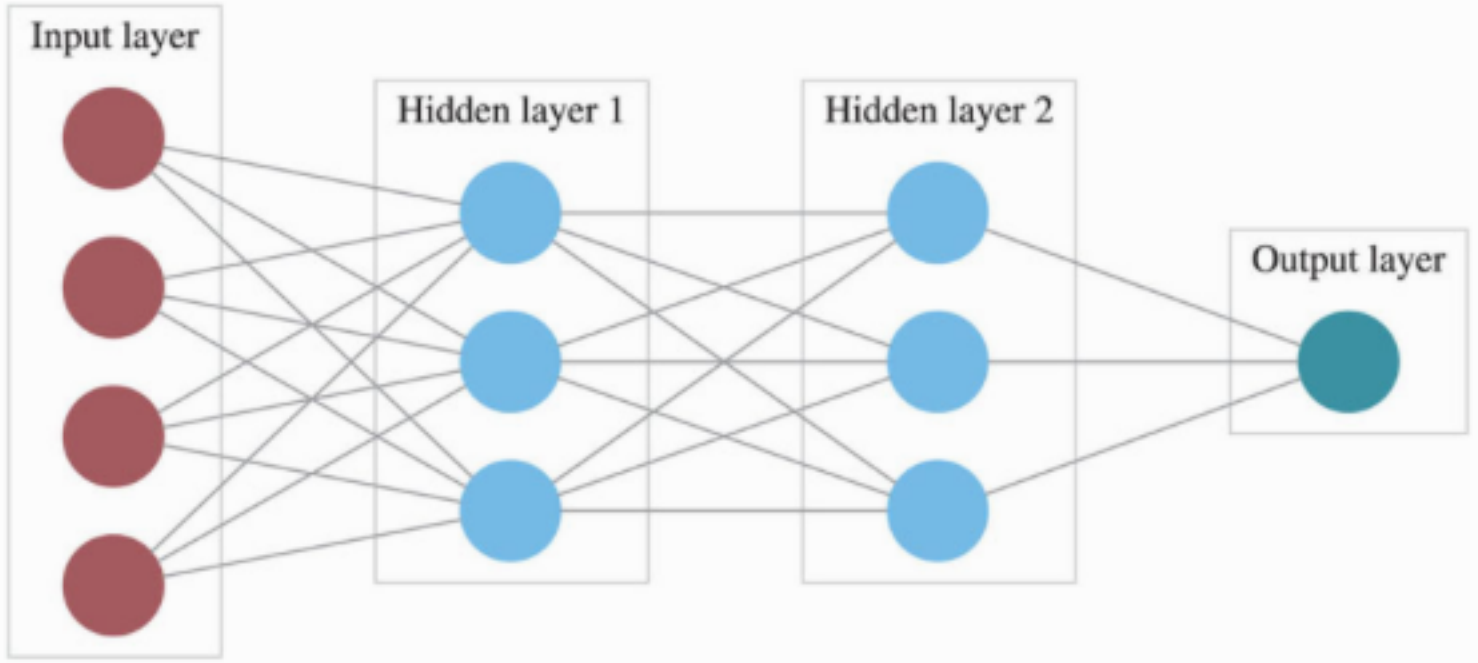


Figure 11.1 Multilayer feedforward neural network

Tiny example

Table 11.1 Tiny Example on Tasting Scores for Six Consumers with Two Predictors

Obs.	Fat score	Salt score	Acceptance
1	0.2	0.9	like
2	0.1	0.1	dislike
3	0.2	0.4	dislike
4	0.2	0.5	dislike
5	0.4	0.5	like
6	0.3	0.8	like

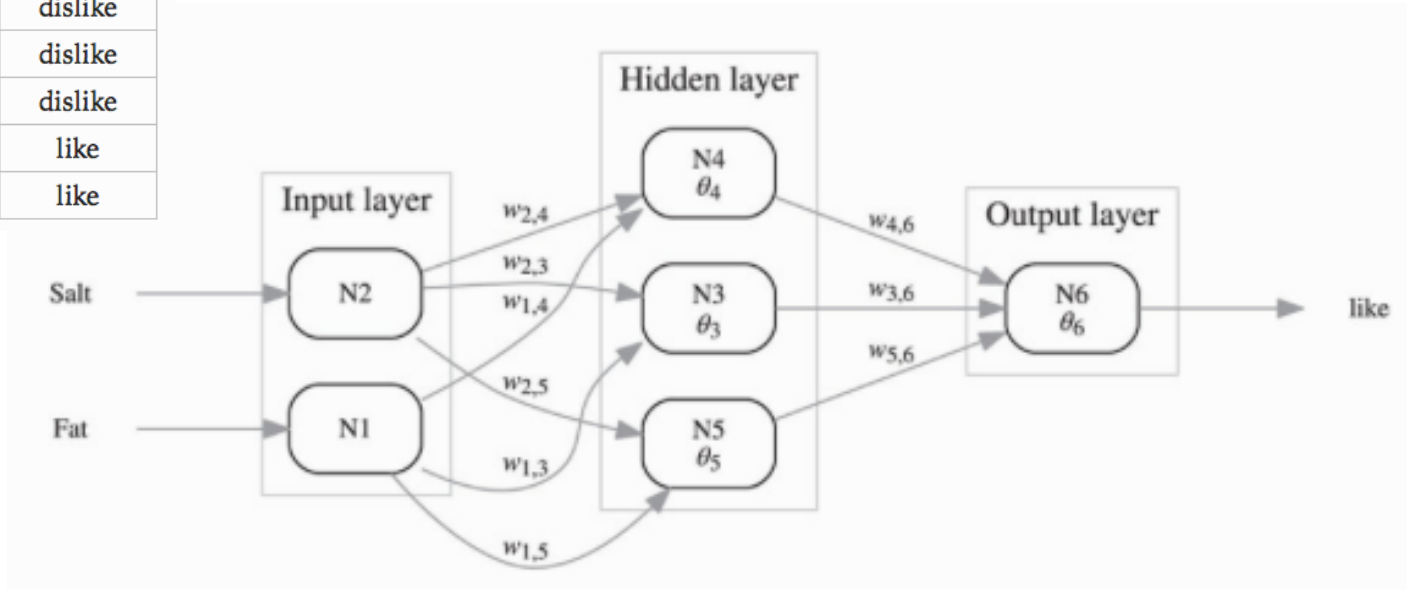


Figure 11.2 Neural network for the tiny example. Rectangles represent nodes ("neurons"), $w_{i,j}$ on arrows are weights, and θ_j inside nodes are bias values

Figure 11.2 describes an example of a typical neural net that could be used for predicting cheese preference (*like/dislike*) by new consumers, based on these data. We numbered the nodes in the example from N1 to N6. Nodes N1 and N2 belong to the input layer, nodes N3 to N5 belong to the hidden layer, and node N6 belongs to the output layer. The values on the connecting arrows are called *weights*, and the weight on the arrow from node i to node j is denoted by $w_{i,\,j}$. The additional *bias* parameters, denoted by θ_j (inside each node), serve as an intercept for the output from node j. These are all explained in further detail below.

1- Input Layer

- No bias in the input layer.
- Output of input layer is the same as its input.

2 – Hidden Layer

- Hidden layer nodes take as input the output values from the input layer
- To compute the output of a hidden layer node, compute a weighted sum of the inputs and apply a certain function to it (transfer function / activation function)

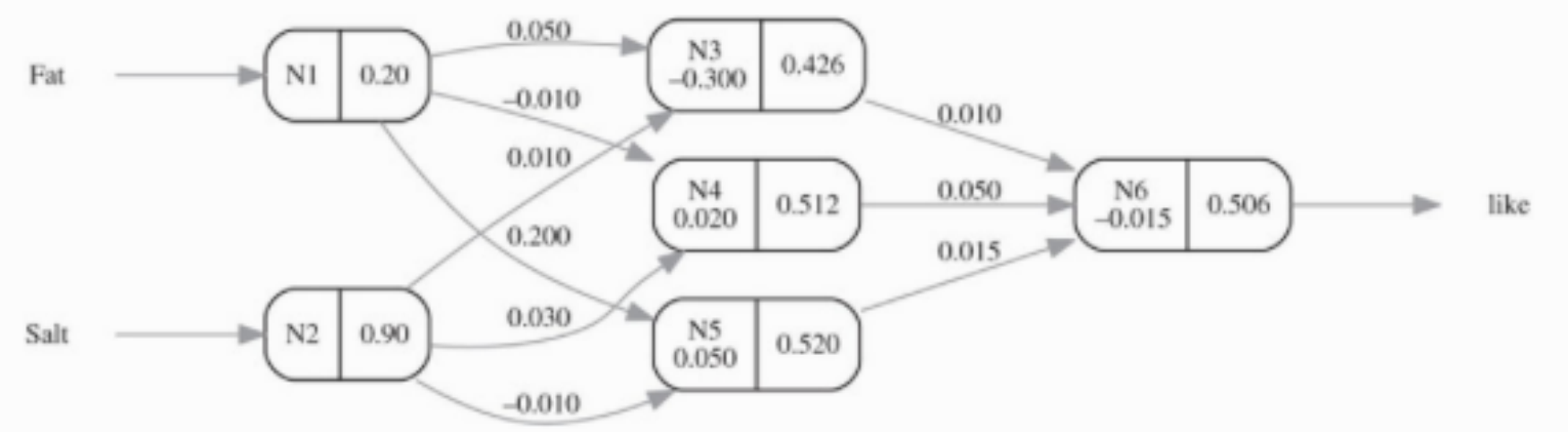


Figure 11.3 Computing node outputs (values are on right side within each node) using the first record in the tiny example and a logistic function

Figure 11.3 shows the initial weights, bias, inputs, and outputs for the first record in our tiny example. If there is more than one hidden layer, the same calculation applies, except that the input values for the second, third, and so on, hidden layers would be the output of the preceding hidden layer. This means that the number of input values into a certain node is equal to the number of nodes in the preceding layer. (If there was an additional hidden layer in our example, its nodes would receive input from the three nodes in the first hidden layer.)

Finally, the output layer obtains input values from the (last) hidden layer. It applies the same function as above to create the output. In other words, it takes a weighted sum of its input values and then applies the function g. In our example, output node N6 receives input from the three hidden layer nodes. We can compute the output of this node by

$$\text{Output}_{\text{N6}} = \frac{1}{1 + e^{-[-0.015 + (0.01)(0.43) + (0.05)(0.51) + (0.015)(0.52)]}} = 0.506. \tag{11.2}$$

How do NNs learn?

- Training a NN means finding the bias and weights that leads to best predictive results
- Activation functions
  - Logistic function
  - Hyperbolic tangent
  - …

Conclusion on Neural Networks

- Good predictive performance
- High tolerance to noisy data
- Capable of capturing highly complicated relationships
- Blackbox
- Overfitting
- Data-dependent
- No built-in variable selection
- Requires high computational time

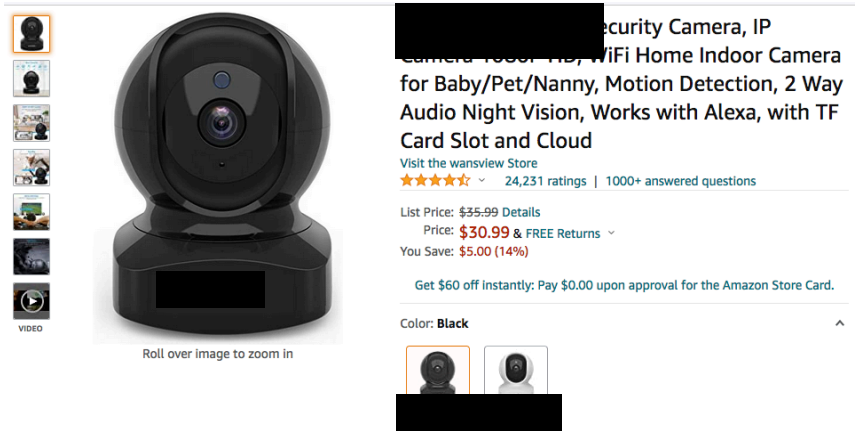
NN Examples:

Review Jupyter Notebook “C810_NN_TinyData”
“C815_NN_Accidents”

Association Rule Mining

Association Rules

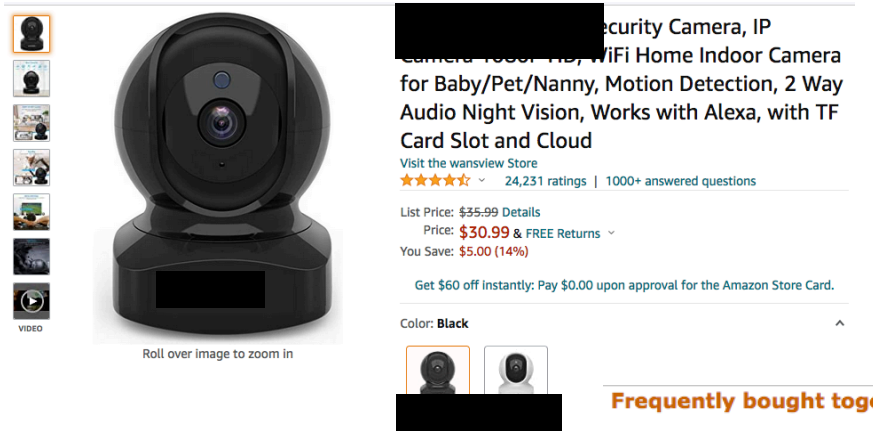
- What goes with what?



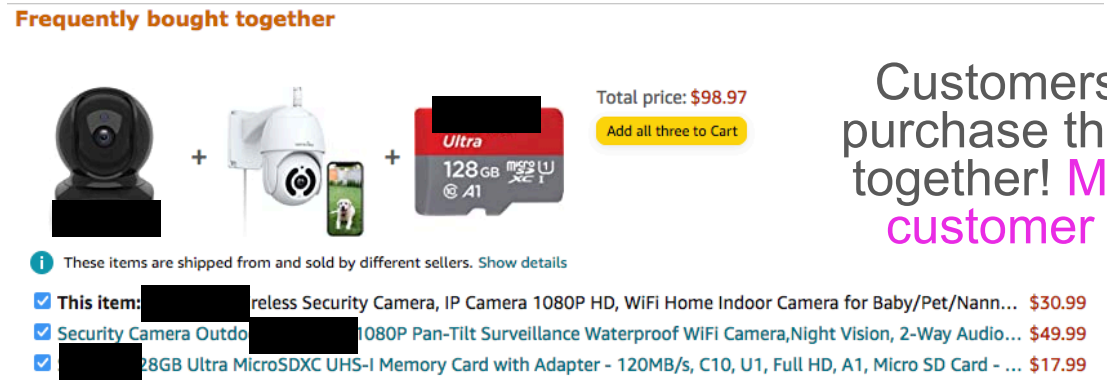
What can I also sell with a security camera?

Association Rules

- What goes with what?



What can I also sell with a security camera?



Customers tend to purchase these items together! Maybe this customer will too!

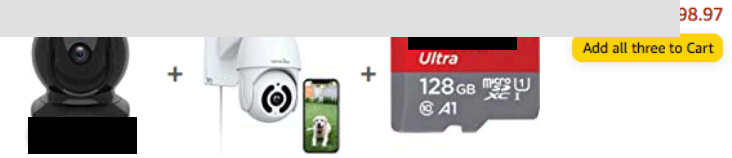
Association Rules

- What goes with what?

What can I also sell with a security camera?

R1: {security camera} → {water proof outdoor camera}

R2: {security camera} → {memory card}



Customers tend to purchase these items together! Maybe this customer will too!

Association Rules

- What goes with what? the goal is to identify item clusters in transaction-type databases (aka market basket analysis, affinity analysis)
- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

TID	*Items*
1	**Bread, Milk**
2	**Bread, Diaper, Beer, Eggs**
3	**Milk, Diaper, Beer, Coke**
4	**Bread, Milk, Diaper, Beer**
5	**Bread, Milk, Diaper, Coke**

Example association rules:

{Diaper} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

Important: Implication means co-occurrence, not causality!

Examples of Association Rules

Transactional Data

TID	*Items*
1	**Bread, Milk**
2	**Bread, Diaper, Beer, Eggs**
3	**Milk, Diaper, Beer, Coke**
4	**Bread, Milk, Diaper, Beer**
5	**Bread, Milk, Diaper, Coke**

Medical Data

	X1 (Pain)	X2 (Headache)	X3 (Weakness)
A (Diabetes)	19	52	20
B (Obesity)	20	52	20
C (Anemia)	14	50	24
D(Tuberculosis)	15	51	24
E (Nephritis)	18	47	18

Security Data

Examples of Association Rules

- Transactional Data: how to optimize store layouts and item placement, for cross-selling, for promotions, for catalog design, and how to identify customer segments based on buying patterns
- Medical: which symptoms appear together?
- Security: which combinations of words indicate a phishing attempt?

Building Blocks of Association Rules: Itemsets and Support

{Beer, Diapers, Milk}:

{Beer, Diapers} ⟶ {Milk}, {Beer, Milk} ⟶ {Diapers},
{Diapers, Milk} ⟶ {Beer}, {Beer} ⟶ {Diapers, Milk},
{Milk} ⟶ {Beer,Diapers}, {Diapers} ⟶ {Beer,Milk}.

Itemsets

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Which transactions contain an itemset?

> Support count:
σ({Milk, Bread,Diaper}) = 2

> Support:
s({Milk, Bread, Diaper}) = 2/5
Transactions 4 and 5, out of 5 total

Building Blocks of Association Rules: Itemsets and Support

{Beer, Diapers, Milk}:

{Beer, Diapers} ⟶ {Milk}, {Beer, Milk} ⟶ {Diapers},
{Diapers, Milk} ⟶ {Beer}, {Beer} ⟶ {Diapers, Milk},
{Milk} ⟶ {Beer,Diapers}, {Diapers} ⟶ {Beer,Milk}.

Itemsets

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Which transactions contain an itemset?

> Support count:
σ({Milk, Bread,Diaper}) = 2

> Support:
s({Milk, Bread, Diaper}) = 2/5
Transactions 4 and 5, out of 5 total

Building Blocks of Association Rules: Support and Confidence

TID	*Items*
1	**Bread, Milk**
2	**Bread, Diaper, Beer, Eggs**
3	**Milk, Diaper, Beer, Coke**
4	**Bread, Milk, Diaper, Beer**
5	**Bread, Milk, Diaper, Coke**

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \{\text{Beer}\}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Building Blocks of Association Rules: Support and Confidence

TID	Items
1	**Bread, Milk**
2	**Bread, Diaper, Beer, Eggs**
3	**Milk, Diaper, Beer, Coke**
4	**Bread, Milk, Diaper, Beer**
5	**Bread, Milk, Diaper, Coke**

$$\{\text{Milk, Diaper}\} \Rightarrow \{\text{Beer}\}$$ Association Rule

$$s = \frac{\sigma(\text{Milk, Diaper, Beer})}{|T|} = \frac{2}{5} = 0.4$$

Support(s): Fraction of transactions that contain both X and Y

$$c = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67$$

Confidence(c): Measures how often items in Y appear in transactions that contain X

Building Blocks of Association Rules: Itemsets and Support Count

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

Building Blocks of Association Rules: Support and Confidence

- Association Rule
  - An implication expression of the form X → Y, where X and Y are itemsets
  - Example: {Milk, Diaper} → {Beer}
- Rule Evaluation Metrics
  - Support (s): Fraction of transactions that contain both X and Y
  - Confidence (c): Measures how often items in Y appear in transactions that contain X

Mining Association Rules

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Mining Association Rules

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

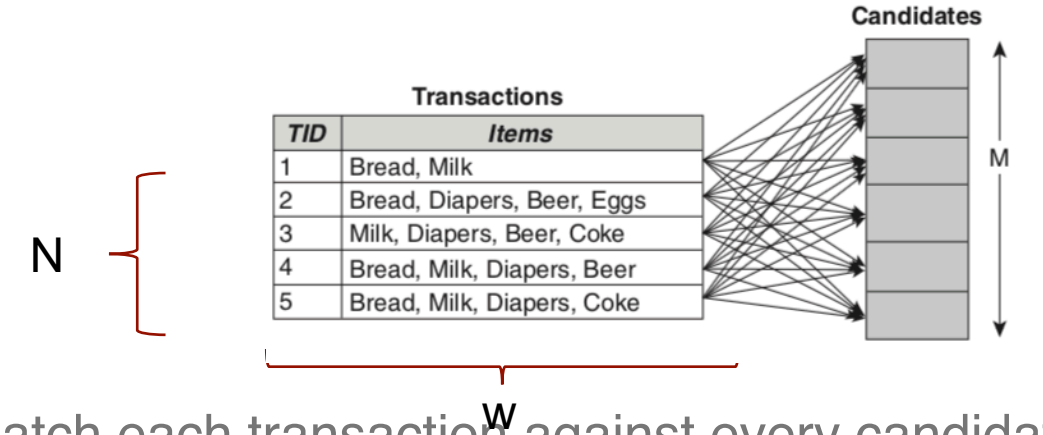
- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Two-step approach

- Frequent Itemset Generation: generate all itemsets whose support >= minsup
- Rule Generation: generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of frequent itemset

Frequent Itemset Generation

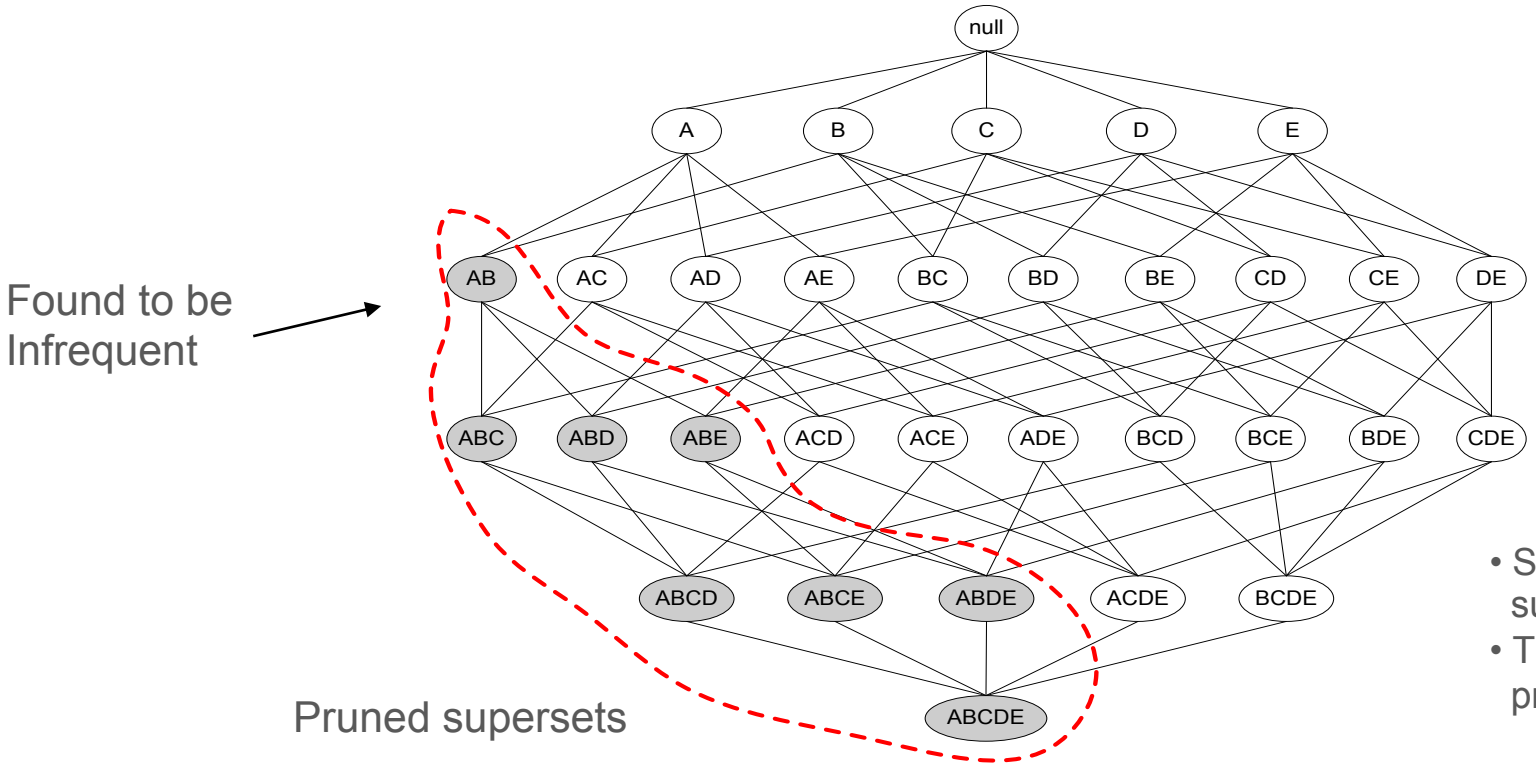
- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database



  - Match each transaction against every candidate
  - Complexity ~ O(NMw) => Expensive since M = 2^d !!!

Apriori Principle

- If an itemset is frequent, then all of its subsets must also be frequent



- Support of an itemset never exceeds the support of its subsets
- This is known as the anti-monotone property of support

Brute-force generation

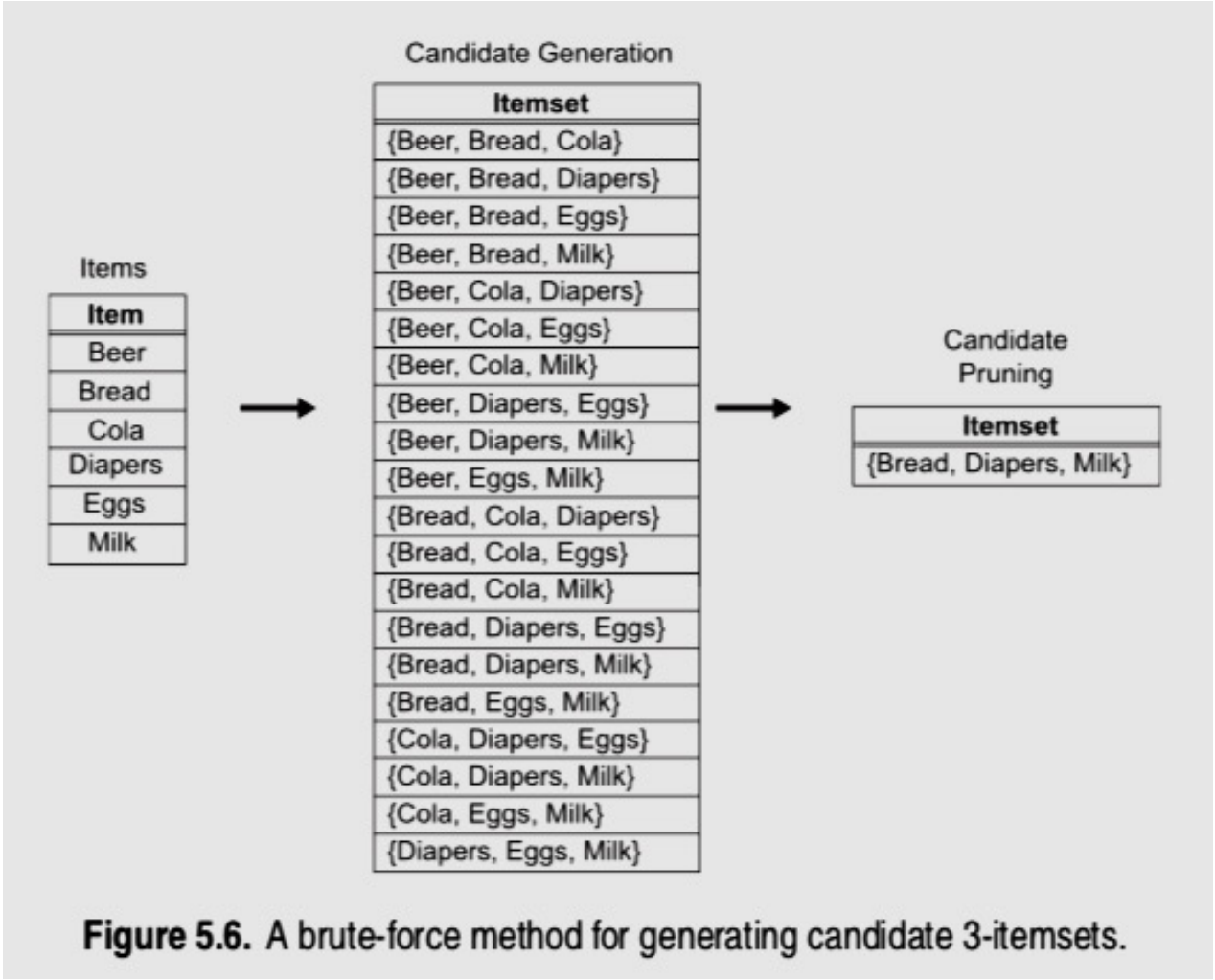


Figure 5.6. A brute-force method for generating candidate 3-itemsets.

Candidate generation how many?

$$C(n, r) = ?$$

$$C(n, r) = C(6, 3)$$

$$= \frac{6!}{(3!(6-3)!)}$$

$$= \frac{6!}{3! \times 3!}$$

$$= 20$$

Illustrating Apriori Principle

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Items (1-itemsets)

Item	Count
Bread	4
~~Coke~~	~~2~~
Milk	4
Beer	3
Diaper	4
~~Eggs~~	~~1~~

Minimum Support = 3

If every subset is considered,

$^{6}C_{1} + {}^{6}C_{2} + {}^{6}C_{3}$

$6 + 15 + 20 = 41$

With support-based pruning,

$6 + 6 + 4 = 16$

Illustrating Apriori Principle

Items (1-itemsets)

Item	Count
Bread	4
~~Coke~~	~~2~~
Milk	4
Beer	3
Diaper	4
~~Eggs~~	~~1~~

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Minimum Support = 3

If every subset is considered,

$^{6}C_{1} + {}^{6}C_{2} + {}^{6}C_{3}$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 4 = 16

Illustrating Apriori Principle

Items (1-itemsets)

Item	Count
Bread	4
~~Coke~~	~~2~~
Milk	4
Beer	3
Diaper	4
~~Eggs~~	~~1~~

Itemset	Count
{Bread,Milk}	3
~~{Beer, Bread}~~	~~2~~
{Bread,Diaper}	3
~~{Beer,Milk}~~	~~2~~
{Diaper,Milk}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Minimum Support = 3

If every subset is considered,

$^{6}C_{1} + {}^{6}C_{2} + {}^{6}C_{3}$

$6 + 15 + 20 = 41$

With support-based pruning,

$6 + 6 + 4 = 16$

Illustrating Apriori Principle

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Itemset	Count
{Bread,Milk}	3
{Beer, Bread}	2
{Bread,Diaper}	3
{Beer,Milk}	2
{Diaper,Milk}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Minimum Support = 3

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 4 = 16

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread,Diaper,Milk}
{ Beer, Bread, Milk}

Items (3-itemsets)

Illustrating Apriori Principle

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Itemset	Count
{Bread,Milk}	3
{Beer, Bread}	2
{Bread,Diaper}	3
{Beer,Milk}	2
{Diaper,Milk}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Coke
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Coke

Minimum Support = 3

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 4 = 16

6 + 6 + 1 = 13

Itemset	Count
{ Beer, Diaper, Milk}	2
{ Beer,Bread, Diaper}	2
{Bread, Diaper, Milk}	2
{Beer, Bread, Milk}	1

Items (3-itemsets)

$$\binom{6}{1} + \binom{4}{2} + 1 = 6 + 6 + 1 = 13$$

68% reduction in the number of candidates achieved using Apriori principle!

Explanation:

Initially, every item is considered as a candidate 1-itemset. After counting their supports, the candidate itemsets {`Cola`} and {`Eggs`} are discarded because they appear in fewer than three transactions. In the next iteration, candidate 2-itemsets are generated using only the frequent 1-itemsets because the *Apriori* principle ensures that all supersets of the infrequent 1-itemsets must be infrequent. Because there are only four frequent 1-itemsets, the number of candidate 2-itemsets generated by the algorithm is $\binom{4}{2} = 6$. Two of these six candidates, {`Beer`, `Bread`} and {`Beer`, `Milk`}, are subsequently found to be infrequent after computing their support values. The remaining four candidates are frequent, and thus will be used to generate candidate 3-itemsets. Without support-based pruning, there are $\binom{6}{3} = 20$ candidate 3-itemsets that can be formed using the six items given in this example. With the *Apriori* principle, we only need to keep candidate 3-itemsets whose subsets are frequent. The only candidate that has this property is {`Bread`, `Diapers`, `Milk`}. However, even though the subsets of {`Bread`, `Diapers`, `Milk`} are frequent, the itemset itself is not.

Apriori Algorithm

Algorithm 5.1 Frequent itemset generation of the *Apriori* algorithm.

1: $k = 1$.
2: $F_k = \{ i \mid i \in I \wedge \sigma(\{i\}) \geq N \times minsup \}$. {Find all frequent 1-itemsets}
3: **repeat**
4: $k = k + 1$.
5: C_k = candidate-gen(F_{k-1}). {Generate candidate itemsets.}
6: C_k = candidate-prune(C_k, F_{k-1}). {Prune candidate itemsets.}
7: **for** each transaction $t \in T$ **do**
8: C_t = subset(C_k, t). {Identify all candidates that belong to t.}
9: **for** each candidate itemset $c \in C_t$ **do**
10: $\sigma(c) = \sigma(c) + 1$. {Increment support count.}
11: **end for**
12: **end for**
13: $F_k = \{ c \mid c \in C_k \wedge \sigma(c) \geq N \times minsup \}$. {Extract the frequent k-itemsets.}
14: **until** $F_k = \emptyset$
15: Result = $\bigcup F_k$.

Review Jupyter Notebooks

Recommender Systems

Collaborative Filtering

What are recommender systems?

- Goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them.
- Suggestions for products on Amazon, or movies on Netflix, are real-world examples of the operation of industry strength recommender systems.
- The design of such recommendation engines depends on the domain and the particular characteristics of the data available

Categorizing approaches to recommender systems

- **Collaborative Filtering (CF):** In CF systems, a user is recommended items based on the past ratings of all users collectively.
- **Content-based recommending:** Recommend items that are similar in content to items the user has liked in the past, or matched to predefined attributes of the user.
- **Hybrid approaches:** These methods combine both collaborative and content-based approaches.

Collaborative filtering: many to one

- Collaborative filtering is a popular technique in identifying relevant items **for a specific user** from the very large set of items (“filtering”) by **considering preferences of many users** (“collaboration”).
- “People Like You”, “Similar Pages”
- Amazon, Netflix, Pandora, Spotify, and many other websites/services
- Covert browsers into buyers, increase cross-selling, and build loyalty!

Ratings Data

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

- n users (u1, u2, …, un) and p items (i1, i2, …, ip), then (ru, i) is the user rating of u for item i, n x p table.
- When both n and p are large, the data can be stored in many rows of triplets of the form (Uu, Ii, ru, i) (more practical)

Example: Netflix Prize Contest

- www.netflixprize.com
- US$ 1 million contest
- Purpose was to improve its recommendation system called Cinematch.
- Winning team used the information on which movies a customer decided rate turned out to be critically informative of customers' preferences, more than simply considering the 1–5 rating information

					Movie ID				
Customer 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								

“People Like You”

- finding users with similar preferences, and recommending items that they liked but the user hasn’t purchased.

  1. Find users who are most similar to the user of interest (neighbors). This is done by comparing the preference of our user to the preferences of other users.

  2. Considering only the items that the user has not yet purchased, recommend the ones that are most preferred by the user’s neighbors.

“People Like You”

- finding users with similar preferences, and recommending items that they liked but the user hasn’t purchased.

  1. Find users who are most similar to the user of interest (neighbors). This is done by comparing the preference of our user to the preferences of other users. → use similarity metrics

  2. Considering only the items that the user has not yet purchased, recommend the ones that are most preferred by the user’s neighbors.→ user-based top-N recommendation

Similarity Metrics

- Cosine Similarity
- Pearson Correlation
- Euclidian Distance
- Manhattan Distance
- Spearman rank correlation
- Kendall's τ correlation
- Mean squared differences,
- Entropy, and adjusted cosine similarity (Herlocker, Konstan, Borchers, & Riedl, 1999; Su & Khoshgoftaar, 2009).

User- or item-based filtering

A subset of users are chosen based on their similarity to the active user, and a weighted combination of their ratings is used to produce predictions for this user

> user-based filtering

> item-based filtering

Challenges and Limitations

- Sparsity
- Cold-start
- Fraud

Cold-start issue

- Collaborative filtering suffers from what is called a cold start: it cannot be used as is to create recommendations for new users or new items. For a user who rated a single item, the correlation coefficient between this and other users (in user-generated collaborative filtering) will have a denominator of zero and the cosine proximity will be 1 regardless of the rating. In a similar vein, users with just one item, and items with just one user, do not qualify as candidates for nearby neighbors.

Resources

- Shmueli et al, 2021, CH14
- Tan et al, 2019, CH5
- Bell, R. M., Koren, Y., and Volinsky, C., “The BellKor 2008 Solution to the Netflix Prize”, www.netflixprize.com/assets/ProgressPrize2008_BellKor.pdf.

Review Jupyter Notebooks