# Lecture 2: 8 April, 2021

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Data Mining and Machine Learning
April–July 2021

# Market-Basket Analysis

- Set of items $I = \{i_1, i_2, \ldots, i_N\}$
- Set of transactions $T = \{t_1, t_2, \ldots, t_M\}$
  - A transaction is a set $t \subseteq I$ of items
- Identify association rules $X \rightarrow Y$
  - $X, Y \subseteq I$, $X \cap Y = \emptyset$
  - If $X \subseteq t_j$ then it is likely that $Y \subseteq t_j$
- Two thresholds
  - How frequently does $X \subseteq t_j$ imply $Y \subseteq t_j$?
  - How significant is this pattern overall?

# Setting thresholds

- For $Z \subseteq I$, $Z.\text{count} = |\{t_j \mid Z \subseteq t_j\}|$
- How frequently does $X \subseteq t_j$ imply $Y \subseteq t_j$?
  - Fix a confidence level $\chi$
  - Want $\frac{(X \cup Y).count}{X.count} \geq \chi$
- How significant is this pattern overall?
  - Fix a support level $\sigma$
  - Want $\frac{(X \cup Y).count}{M = |T|} \geq \sigma$
- Given sets of items $I$ and transactions $T$, with confidence $\chi$ and support $\sigma$, find all valid association rules $X \rightarrow Y$

"Exact" solution

# Frequent itemsets

- $X \to Y$ is interesting only if $(X \cup Y).\text{count} \geq \sigma \cdot M$
- First identify all frequent itemsets
  - $Z \subseteq I$ such that $Z.\text{count} \geq \sigma \cdot M$
- Naïve strategy: maintain a counter for each $Z$
  - For each $t_j \in T$
    For each $Z \subseteq t_j$
    Increment the counter for $Z$
  - After scanning all transactions, keep $Z$ with $Z.\text{count} \geq \sigma \cdot M$
- Need to maintain $2^{|I|}$ counters
  - Infeasible amount of memory
  - Can we do better?

$\frac{Z.\text{count}}{M} \geq \sigma$

Set $X$, $|X| = n$
Subsets $2^n$

# Sample calculation

- Let's assume a bound on each $t_i \in T$
  - No transaction has more than 10 items
- Say $N = |I| = 10^6$, $M = |T| = 10^9$, $\sigma = 0.01$
  - Number of possible subsets to count is $\sum_{i=1}^{10} \binom{10^6}{i}$

Fix T - must be a bound

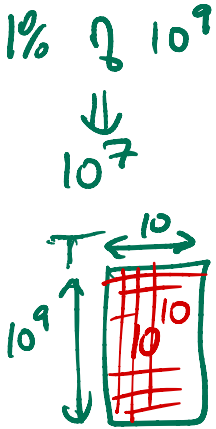
1 million

1 billion

1%

# Sample calculation

- Let's assume a bound on each $t_i \in T$
  - No transaction has more than 10 items
- Say $N = |I| = 10^6$, $M = |T| = 10^9$, $\sigma = 0.01$
  - Number of possible subsets to count is $\sum_{i=1}^{10} \binom{10^6}{i}$
- A singleton subset $\{x\}$ that is frequent is an item $x$ that appears in at least $10^7$ transactions



# Sample calculation

- Let's assume a bound on each $t_i \in T$
  - No transaction has more than 10 items
- Say $N = |I| = 10^6$, $M = |T| = 10^9$, $\sigma = 0.01$
  - Number of possible subsets to count is $\sum_{i=1}^{10} \binom{10^6}{i}$
- A singleton subset $\{x\}$ that is frequent is an item $x$ that appears in at least $10^7$ transactions
- Totally, $T$ contains at most $10^{10}$ items
- At most $10^{10}/10^7 = 1000$ items are frequent!
- How can we exploit this?

$10^6$ items

1000 can be frequent

$10 \times 10^9$

# Apriori

- Clearly, if $Z$ is frequent, so is every subset $Y \subseteq Z$

# Apriori

- Clearly, if $Z$ is frequent, so is every subset $Y \subseteq Z$
- We exploit the contrapositive

If A then B

If ¬B then ¬A

**Apriori observation**

If $Z$ is not a frequent itemset, no superset $Y \supseteq Z$ can be frequent

# Apriori

- Clearly, if $Z$ is frequent, so is every subset $Y \subseteq Z$
- We exploit the contrapositive

**Apriori observation**

If $Z$ is not a frequent itemset, no superset $Y \supseteq Z$ can be frequent

- For instance, in our earlier example, every frequent itemset must be built from the 1000 frequent items
- In particular, for any frequent pair $\{x, y\}$, both $\{x\}$ and $\{y\}$ must be frequent
- Build frequent itemsets bottom up, size 1,2,...

# Apriori algorithm

- $F_i$ : frequent itemsets of size $i$ — Level $i$

# Apriori algorithm

- $F_i$ : frequent itemsets of size $i$ — Level $i$
- $F_1$: Scan $T$, maintain a counter for each $x \in I$

$|I| = 10^6$

# Apriori algorithm

- $F_i$ : frequent itemsets of size $i$ — Level $i$
- $F_1$: Scan $T$, maintain a counter for each $x \in I$
- $C_2 = \{\{x, y\} \mid x, y \in F_1\}$: Candidates in level 2

$1000 \times 1000 \Rightarrow 10^6$
$F_1 \quad F_1$
1MB

$|I| \times |I|$
$10^{12}$ – 1000 GB = 1TB

# Apriori algorithm

- $F_i$ : frequent itemsets of size $i$ — Level $i$
- $F_1$: Scan $T$, maintain a counter for each $x \in I$
- $C_2 = \{\{x, y\} \mid x, y \in F_1\}$: Candidates in level 2
- $F_2$: Scan $T$, maintain a counter for each $X \in C_2$

# Apriori algorithm

- $F_i$ : frequent itemsets of size $i$ — Level $i$
- $F_1$: Scan $T$, maintain a counter for each $x \in I$
- $C_2 = \{\{x, y\} \mid x, y \in F_1\}$: Candidates in level 2
- $F_2$: Scan $T$, maintain a counter for each $X \in C_2$
- $C_3 = \{\{x, y, z\} \mid \{x, y\}, \{x, z\}, \{y, z\} \in F_2\}$
- $F_3$: Scan $T$, maintain a counter for each $X \in C_3$

Also $\{x\}, \{y\}, \{z\} \in F_1$

Subsumed by Apriori on $F_2$

# Apriori algorithm

- $F_i$ : frequent itemsets of size $i$ — Level $i$
- $F_1$: Scan $T$, maintain a counter for each $x \in I$
- $C_2 = \{\{x, y\} \mid x, y \in F_1\}$: Candidates in level 2
- $F_2$: Scan $T$, maintain a counter for each $X \in C_2$
- $C_3 = \{\{x, y, z\} \mid \{x, y\}, \{x, z\}, \{y, z\} \in F_2\}$ — $k=3$
- $F_3$: Scan $T$, maintain a counter for each $X \in C_3$
- ...
- $C_k$ = subsets of size $k$, every $(k-1)$-subset is in $F_{k-1}$
- $F_k$: Scan $T$, maintain a counter for each $X \in C_k$ —
- ...

# Apriori algorithm

- $C_k$ = subsets of size $k$, every $(k-1)$-subset is in $F_{k-1}$
- How do we generate $C_k$?

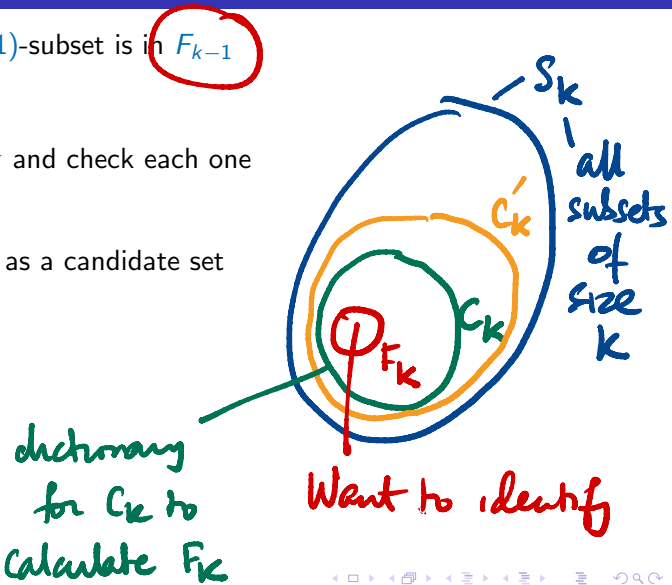
$\binom{N}{k}$

# Apriori algorithm

- $C_k$ = subsets of size $k$, every $(k-1)$-subset is in $F_{k-1}$
- How do we generate $C_k$?
- Naïve: enumerate subsets of size $k$ and check each one
  - Expensive!

# Apriori algorithm

- $C_k$ = subsets of size $k$, every $(k-1)$-subset is in $F_{k-1}$
- How do we generate $C_k$?
- Naïve: enumerate subsets of size $k$ and check each one
  - Expensive!
- Observation: Any $C'_k \supseteq C_k$ will do as a candidate set



Find frequent subsets, maintain a counter for each subset

- All possible frequent subsets
- Instead level by level
  - Frequent subset of size 1
  - Frequent subsets of size 2
  - :

$M$ transactions

$\sigma$ support threshold

$$\frac{\Sigma.\text{count}}{M} \geq \sigma$$

$$\Sigma.\text{count} \geq \sigma \cdot M$$

Frequent sets of size 1 — Counter for every $\{x\}$

Frequent sets of size 2 = Counter for every $\{x,y\}$

Frequent sets of size $k$ = Counter for every $\binom{N}{k}$ $\{x_1, x_2, \ldots, x_k\}$

Count subsets of size k

- Naive: counter for every set $X \in I_k$ (↑ k-size subsets of $I$)
- Instead: find a smaller set of "candidates" to count

$$F_k \subseteq C_k \subseteq I_k$$

- $F_k$: Frequent sets of size k
- $C_k$: candidate set of size k
- $I_k$: All sets of size k

For X to be in $C_k$

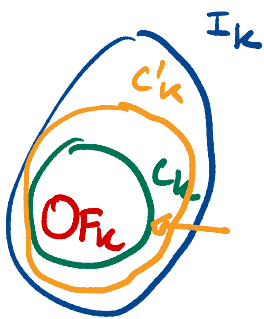
Apriori → every subset of X must be frequent

- Every 1-subset is in $F_1$
- Every 2-subset is in $F_2$
- ⋮
- Every k-1 subset is in $F_{k-1}$ → Implies all the lines above

↳ sufficient to check this alone

# Apriori algorithm

- $C_k$ = subsets of size $k$, every $(k-1)$-subset is in $F_{k-1}$
- How do we generate $C_k$?
- Naïve: enumerate subsets of size $k$ and check each one
  - Expensive!
- Observation: Any $C'_k \supseteq C_k$ will do as a candidate set
- Items are ordered: $i_1 < i_2 < \cdots < i_N$
- List each itemset in ascending order
- Merge two $(k-1)$-subsets if they differ in last element
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\}$
  - $X' = \{i_1, i_2, \ldots, i_{k-2}, i'_{k-1}\}$
  - $\text{Merge}(X, X') = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\}$




# Apriori algorithm

- $\text{Merge}(X, X') = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\}$
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\}$
  - $X' = \{i_1, i_2, \ldots, i_{k-2}, i'_{k-1}\}$
- $C'_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$

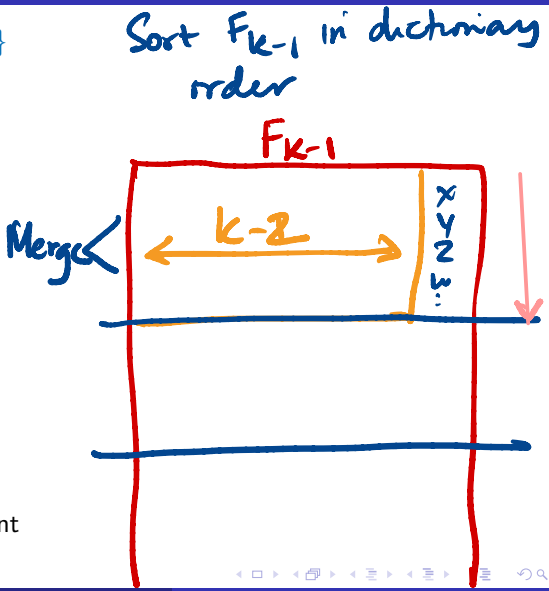
To show
$C'_k \supseteq C_k$

# Apriori algorithm

- $\text{Merge}(X, X') = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\}$
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\}$
  - $X' = \{i_1, i_2, \ldots, i_{k-2}, i'_{k-1}\}$
- $C'_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
- Claim $C_k \subseteq C'_k$
  - Suppose $Y = \{i_1, i_2, \ldots, i_{k-1}, i_k\} \in C_k$
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\} \in F_{k-1}$ and $X' = \{i_1, i_2, \ldots, i_{k-2}, i_k\} \in F_{k-1}$
  - $Y = \text{Merge}(X, X') \in C'_k$

# Apriori algorithm

- $\text{Merge}(X, X') = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\}$
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\}$
  - $X' = \{i_1, i_2, \ldots, i_{k-2}, i'_{k-1}\}$
- $C'_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
- Claim $C_k \subseteq C'_k$
  - Suppose $Y = \{i_1, i_2, \ldots, i_{k-1}, i_k\} \in C_k$
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\} \in F_{k-1}$ and $X' = \{i_1, i_2, \ldots, i_{k-2}, i_k\} \in F_{k-1}$
  - $Y = \text{Merge}(X, X') \in C'_k$
- Can generate $C'_k$ efficiently
  - Arrange $F_{k-1}$ in dictionary order
  - Split into blocks that differ on last element
  - Merge all pairs within each block



$Y \in C_k$ — Merge $(X_1, X_2)$ ($k-1$, $k-1$)

$[i_1, i_2, \ldots i_{k-2}, i_{k-1}, i_k]$

Drop

Upto k-2, every subset will be frequent

At k-1 – no justification $i_1, i_3, i_2 \ldots i_{k-1}, i_k \in F_{k-1}$

# Apriori algorithm

- $C_1 = \{\{x\} \mid x \in I\}$
- $F_1 = \{Z \mid Z \in C_1, Z.\text{count} \geq \sigma \cdot M\}$
- For $k \in \{2, 3, \ldots\}$
  - $C_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
  - $F_k = \{Z \mid Z \in C_k, Z.\text{count} \geq \sigma \cdot M\}$

Sets of size 1

Prefix of length 0

{x} {y} → {x,y}

$C_k$ $C'_k$

$C_3$ : {x,y,z}

$C'_3$ : {x,y,z}

xy, xz, yz ∈ $F_2$

xy, xz ∈ $F_2$

# Apriori algorithm

- $C_1 = \{\{x\} \mid x \in I\}$
- $F_1 = \{Z \mid Z \in C_1, Z.\text{count} \geq \sigma \cdot M\}$
- For $k \in \{2, 3, \ldots\}$
  - $C_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
  - $F_k = \{Z \mid Z \in C_k, Z.\text{count} \geq \sigma \cdot M\}$
- When do we stop?

# Apriori algorithm

- $C_1 = \{\{x\} \mid x \in I\}$
- $F_1 = \{Z \mid Z \in C_1, Z.\text{count} \geq \sigma \cdot M\}$
- For $k \in \{2, 3, \ldots\}$
  - $C_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
  - $F_k = \{Z \mid Z \in C_k, Z.\text{count} \geq \sigma \cdot M\}$
- When do we stop?
- $k$ exceeds the size of the largest transaction
- $F_k$ is empty

Binning
Make blocks
Pairwise within blocks

# Apriori algorithm

- $C_1 = \{\{x\} \mid x \in I\}$
- $F_1 = \{Z \mid Z \in C_1, Z.\text{count} \geq \sigma \cdot M\}$
- For $k \in \{2, 3, \ldots\}$
  - $C_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
  - $F_k = \{Z \mid Z \in C_k, Z.\text{count} \geq \sigma \cdot M\}$
- When do we stop?
- $k$ exceeds the size of the largest transaction
- $F_k$ is empty

Next step: From frequent itemsets to association rules

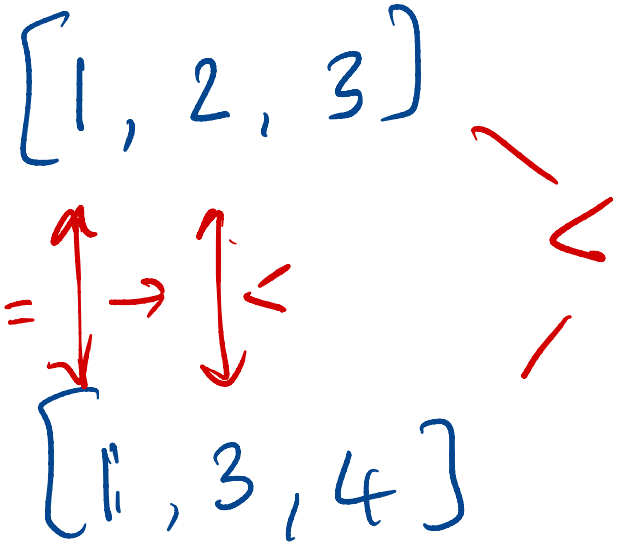
Identified $Z$

$\frac{X \cup Y.\text{count}}{X.\text{count}} \geq \chi$ s.t. $Z = X \cup Y$

$\frac{Z.\text{count}}{M} \geq \sigma$

Split $Z$ as $X, Y$ s.t.

$X \to Y$ matches confidence $\chi$




an
<
ant

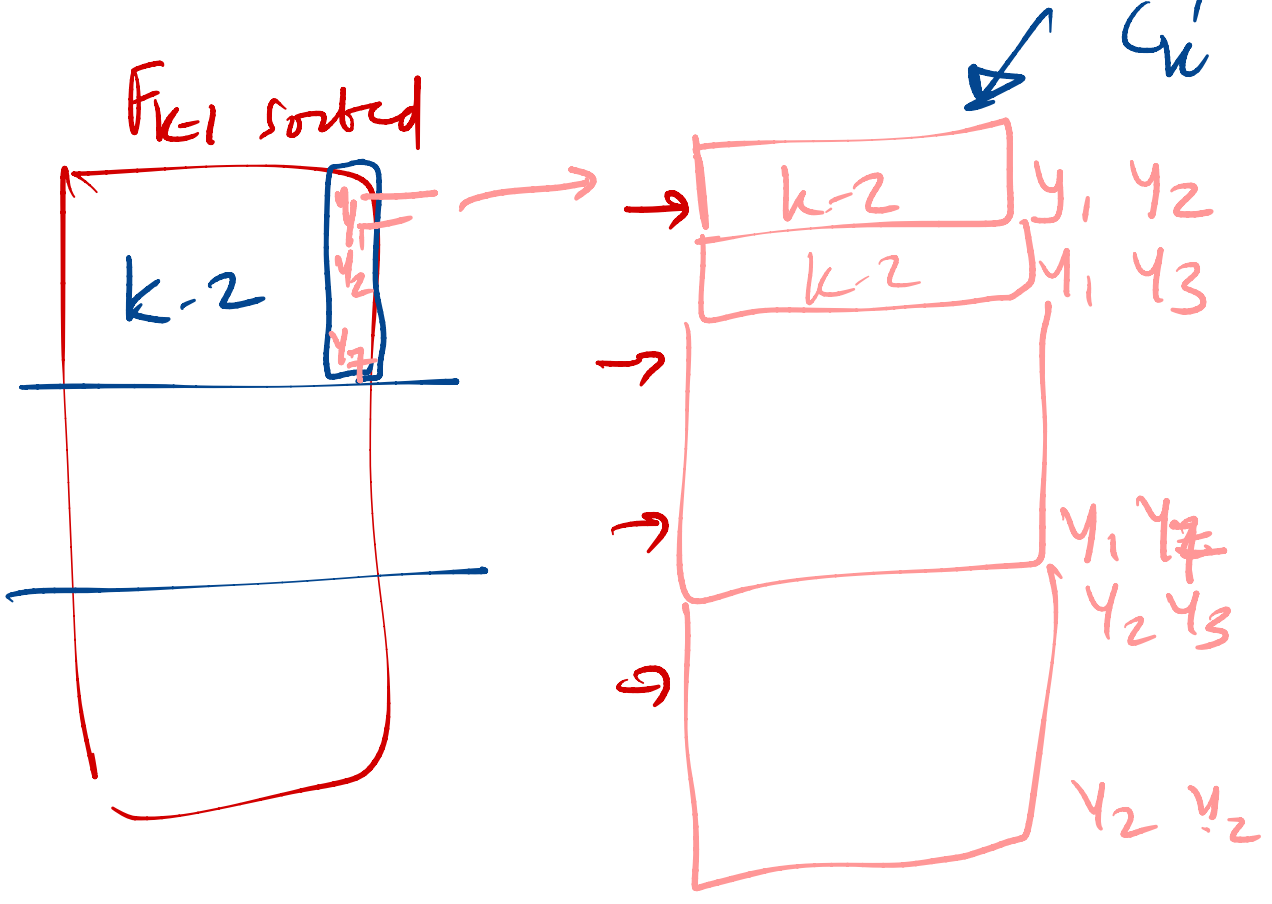
$(x_1, \ldots, x_{k-1})$

vs

$(y_1, \ldots y_{k-1})$

ant

art


$F_{k-1}$ sorted
k-2
$y_1$
$y_2$
$y_3$
$G_k'$ in lexicographic order
k-2
$y_1$ $y_2$
k-2
$y_1$ $y_3$
$y_1$ $y_3$
$y_2$ $y_3$
$y_2$ $y_2$