Lecture 2: 10 January, 2023

Madhavan Mukund

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

Data Mining and Machine Learning
January–April 2023

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 χ
  - Want $\dfrac{(X \cup Y).count}{X.count} \geq \chi$
- How significant is this pattern overall?
  - Fix a support level σ
  - Want $\dfrac{(X \cup Y).count}{M} \geq \sigma$
- Given sets of items I and transactions T, with confidence χ and support σ, find all valid association rules $X \rightarrow Y$

Frequent itemsets

- $X \rightarrow Y$ 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?

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 $\displaystyle\sum_{i=1}^{10} \binom{10^6}{i}$

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 $\displaystyle\sum_{i=1}^{10} \binom{10^6}{i}$
- 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 $\displaystyle\sum_{i=1}^{10} \binom{10^6}{i}$
- 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?

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

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 $x \in I$

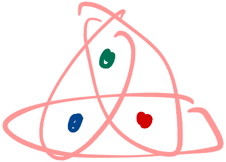
Count $\geq \sigma \cdot M$

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

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 $X \in C_3$

$I = C_1 \longrightarrow F_1$

$C_2 \approx F_1 \times F_1 \longrightarrow F_2$

$C_3 \longrightarrow F_3$

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 $X \in C_3$
- ...
- 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

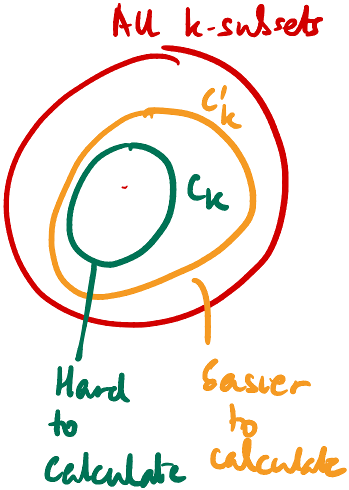
- C_k = subsets of size k, every $(k-1)$-subset is in F_{k-1}
- How do we generate C_k?

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

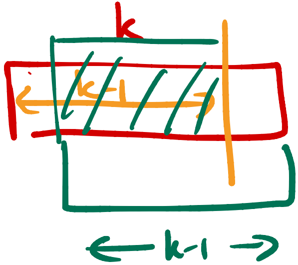


Apriori algorithm

- 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 — canonical representation

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 — canonical representation
- 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

- $\mathrm{Merge}(X, X') = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\}$
  - $X = \{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\}$
  - $X' = \{i_1, i_2, \ldots, i_{k-2}, i'_{k-1}\}$

Apriori algorithm

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

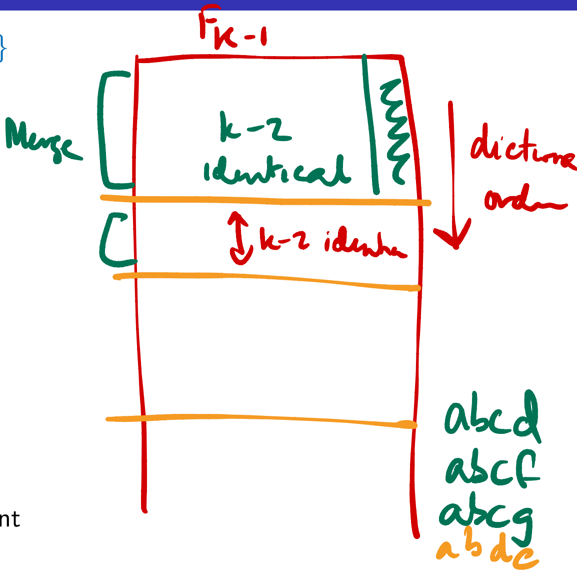
Apriori algorithm

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

$x_1, x_2, \ldots, x_{k-2}, \cancel{x_{k-1}}, \cancel{x_k}$

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



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 $k \in \{2, 3, \ldots\}$
    - $C'_k = \{\text{Merge}(X, X') \mid X, X' \in F_{k-1}\}$
    - $F_k = \{Z \mid Z \in C'_k, Z.\text{count} \geq \sigma \cdot M\}$

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   More generally, C_k is empty

C_2 F_1 $\sqcup\, x_1$ $\sqcup\, x_1'$

$C_2' = C_2 = F_1 \times F_1 \setminus I$

C'_3 (x, y, z)

Also: Applicability of insight

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

Association rules

- Given sets of items I and transactions T, with confidence χ and support σ, find all valid association rules $X \to Y$
  - $X, Y \subseteq I$, $X \cap Y = \emptyset$
  - $\dfrac{(X \cup Y).count}{X.count} \geq \chi$
  - $\dfrac{(X \cup Y).count}{M} \geq \sigma$

Association rules

- Given sets of items I and transactions T, with confidence χ and support σ, find all valid association rules $X \rightarrow Y$
  - $X, Y \subseteq I$, $X \cap Y = \emptyset$
  - $\dfrac{(X \cup Y).count}{X.count} \geq \chi$
  - $\dfrac{(X \cup Y).count}{M} \geq \sigma$
- For a rule $X \rightarrow Y$ to be valid, $X \cup Y$ should be a frequent itemset
- Apriori algorithm finds all $Z \subseteq I$ such that $Z.\text{count} \geq \sigma \cdot M$

Association rules

Naïve strategy

- For every frequent itemset Z
  - Enumerate all pairs $X, Y \subseteq Z$, $X \cap Y = \emptyset$
  - Check $\dfrac{(X \cup Y).count}{X.count} \geq \chi$

Association rules

Naïve strategy

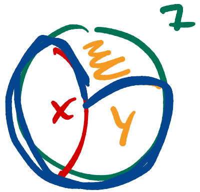
- For every frequent itemset Z
  - Enumerate all pairs $X, Y \subseteq Z$, $X \cap Y = \emptyset$
  - Check $\dfrac{(X \cup Y).count}{X.count} \geq \chi$

Can we do better?

Association rules

Naïve strategy

- For every frequent itemset Z
  - Enumerate all pairs $X, Y \subseteq Z$, $X \cap Y = \emptyset$
  - Check $\dfrac{(X \cup Y).count}{X.count} \geq \chi$



Can we do better?

- Sufficient to check all partitions of Z
  - Suppose $X, Y \subseteq Z$, $X \to Y$ is a valid association rule, but $X \cup Y$ is a proper subset of Z

Association rules

Naïve strategy

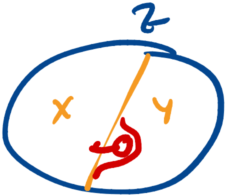
- For every frequent itemset Z
  - Enumerate all pairs $X, Y \subseteq Z$, $X \cap Y = \emptyset$
  - Check $\dfrac{(X \cup Y).count}{X.count} \geq \chi$

Can we do better?

- Sufficient to check all partitions of Z
  - Suppose $X, Y \subseteq Z$, $X \rightarrow Y$ is a valid association rule, but $X \cup Y$ is a proper subset of Z
  - $X \cup Y = Z' \subsetneq Z$
  - Z' is also a frequent itemset (a priori)
  - X, Y partitions Z'

Association rules

- Sufficient to check all partitions of Z
- Suppose $Z = X \uplus Y$, $X \to Y$ is a valid rule and $y \in Y$
- What about $(X \cup \{y\}) \to Y \setminus \{y\}$?



$X \to Y$

is valid

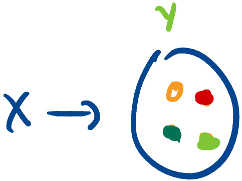
$$\frac{(X \cup Y).\text{count}}{X.\text{count}} \geq \chi$$

Association rules

- Sufficient to check all partitions of Z
- Suppose $Z = X \uplus Y$, $X \rightarrow Y$ is a valid rule and $y \in Y$
- What about $(X \cup \{y\}) \rightarrow Y \setminus \{y\}$?
  - Know $\dfrac{(X \cup Y).count}{X.count} \geq \chi$
  - Check $\dfrac{(X \cup Y).count}{(X \cup \{y\}).count} \geq \chi$

Association rules

- Sufficient to check all partitions of Z
- Suppose $Z = X \uplus Y$, $X \rightarrow Y$ is a valid rule and $y \in Y$
- What about $(X \cup \{y\}) \rightarrow Y \setminus \{y\}$?
  - Know $\dfrac{(X \cup Y).count}{X.count} \geq \chi$
  - Check $\dfrac{(X \cup Y).count}{(X \cup \{y\}).count} \geq \chi$
  - X.count $\geq (X \cup \{y\})$.count, always
  - Second fraction has smaller denominator, so $(X \cup \{y\}) \rightarrow Y \setminus \{y\}$ is also a valid rule



Association rules

- Sufficient to check all partitions of Z
- Suppose $Z = X \uplus Y$, $X \to Y$ is a valid rule and $y \in Y$
- What about $(X \cup \{y\}) \to Y \setminus \{y\}$?
  - Know $\dfrac{(X \cup Y).count}{X.count} \geq \chi$
  - Check $\dfrac{(X \cup Y).count}{(X \cup \{y\}).count} \geq \chi$
  - X.count $\geq (X \cup \{y\})$.count, always
  - Second fraction has smaller denominator, so $(X \cup \{y\}) \to Y \setminus \{y\}$ is also a valid rule

Observation: Can use apriori principle again!

Apriori for association rules

- If $X \to Y$ is a valid rule, and $y \in Y$,
  $(X \cup \{y\}) \to Y \setminus \{y\}$ must also be a valid rule

- If $X \to Y$ is not a valid rule, and $x \in X$,
  $(X \setminus \{x\}) \to Y \cup \{x\}$ cannot be a valid rule

Apriori for association rules

- If $X \to Y$ is a valid rule, and $y \in Y$, $(X \cup \{y\}) \to Y \setminus \{y\}$ must also be a valid rule
- If $X \to Y$ is not a valid rule, and $x \in X$, $(X \setminus \{x\}) \to Y \cup \{x\}$ cannot be a valid rule
- Start by checking rules with single element on the right
  - $Z \setminus \{z\} \to \{z\}$
- For $X \to \{x, y\}$ to be a valid rule, both $(X \cup \{x\}) \to \{y\}$ and $(X \cup \{y\}) \to \{x\}$ must be valid
- Explore partitions of each frequent itemset "level by level"

Association rules for classification

- Classify documents by topic
- Consider the table on the right

Words in document	**Topic**
student, teach, school	Education
student, school	Education
teach, school, city, game	Education
cricket, football	Sports
football, player, spectator	Sports
cricket, coach, game, team	Sports
football, team, city, game	Sports

Association rules for classification

- Classify documents by topic
- Consider the table on the right
- Items are regular words and topics
- Documents are transactions — set of words and one topic

Words in document	Topic
student, teach, school	Education
student, school	Education
teach, school, city, game	Education
cricket, football	Sports
football, player, spectator	Sports
cricket, coach, game, team	Sports
football, team, city, game	Sports

Association rules for classification

- Classify documents by topic
- Consider the table on the right
- Items are regular words and topics
- Documents are transactions — set of words and one topic
- Look for association rules of a special form
  - $\{\text{student, school}\} \rightarrow \{\text{Education}\}$
  - $\{\text{game, team}\} \rightarrow \{\text{Sports}\}$

Words in document	**Topic**
student, teach, school	Education
student, school	Education
teach, school, city, game	Education
cricket, football	Sports
football, player, spectator	Sports
cricket, coach, game, team	Sports
football, team, city, game	Sports

Association rules for classification

- Classify documents by topic
- Consider the table on the right
- Items are regular words and topics
- Documents are transactions — set of words and one topic
- Look for association rules of a special form
  - $\{$student, school$\} \rightarrow \{$Education$\}$
  - $\{$game, team$\} \rightarrow \{$Sports$\}$
- Right hand side always a single topic
- Class Association Rules

Words in document	**Topic**
student, teach, school	Education
student, school	Education
teach, school, city, game	Education
cricket, football	Sports
football, player, spectator	Sports
cricket, coach, game, team	Sports
football, team, city, game	Sports

Summary

- Market-basket analysis searches for correlated items across transactions
- Formalized as association rules
- Apriori principle helps us to efficiently
  - identify frequent itemsets, and
  - split these itemsets into valid rules
- Class association rules — simple supervised learning model