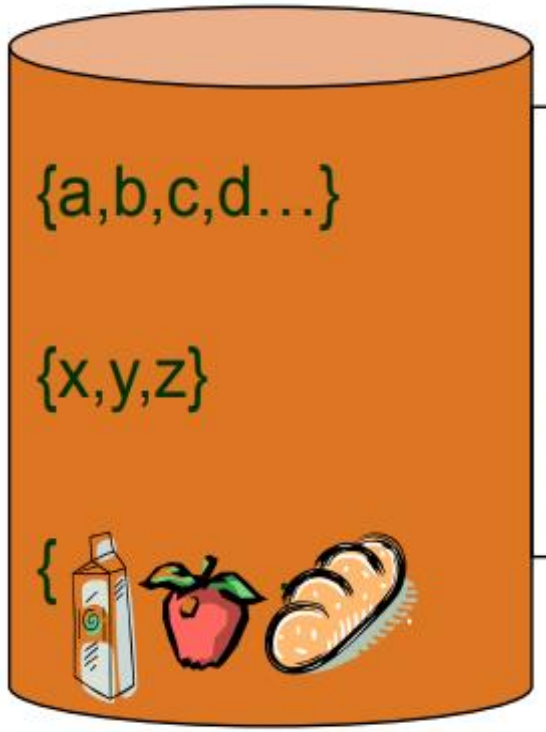

Organic
SEAFOOD
GROUND BEER
97¢
Rollback
Was $1.18
97¢
Low Price
$1.18
88¢
Rollback
Was 82¢
68¢
Rollback
Was 82¢
68¢
Everyday Low Price
98¢
Everyday Low Price
$1.42

Association Rules
&
Apriori Algorithm

aims at discovering associations between items in a transactional database.



find combinations of items that occur typically together

Association rule mining

Given a set of records each of which contain some number of items from a given collection of transactions. Aim is to produce the dependency rules which will predict occurrence of an item based on the occurrences of other items.

Transaction	ID	Items bought
T1	10	Beer, Nuts, Diaper
T2	20	Beer, Coke, Diaper
T3	30	Bread, Diaper, Eggs
T4	40	Nuts, Eggs, Milk
T5	50	Nuts, Coffee, Diaper, Eggs, Milk

Example of Association Rules

{Diaper} $\longrightarrow$ {Beer},

{Milk, Bread} $\longrightarrow$ {Eggs,Coke},

{Beer, Bread} $\longrightarrow$ {Milk}

Implication ("->") means co-occurrence, not causality!

- **The customers who purchase Bread have a chance to purchase Milk**
- **The retail organizations are trying to find the assocation between products which can be sold togother**

Note: Assume all data are **categorical**. However, no good algorithm for numeric data.

Example of Association rule

Let due to some observation it found that 90% of transactions that purchase bread and butter also purchase milk.

Then the rule discovered be :- **If(bread and butter) then milk**

"IF" part = antecedent bread and butter ⟶ Milk [with confidence level 90%]

"THEN" part = consequent (If: antecedent) (then: consequent)

Note: Antecedent and consequent are disjoint (i.e., have no items in common)

Milk is consequent => Can be used to determine what should be done to boost its sales.

Bread and butter in the antecedent => Can be used to see which products would be affected if the store discontinues selling bread and butter.

Bread and butter in antecedent and Milk in consequent => Can be used to see what products should be sold with bread and butter to promote sale of Milk!

Transaction data: Supermarket data

Concepts:

- **I:** the set of all items sold in the store. {Bread, Milk,...}
- **A transaction:** items purchased in a basket; it may have TID (transaction ID)
- **A transactional dataset:** A set of transactions
- **Itemset:** A collection of one or more items

  {Milk, Bread, Diaper}

  ➢k-itemset: An itemset that contains k items

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

- **Support count (σ):** Frequency of occurrence of an itemset

  ({Milk, Bread,Diaper}) = 2, ({Milk})= 4

- **Support:** Fraction of transactions that contain an itemset

  S({Milk, Bread, Diaper}) = 2/5, S({Milk})= 4/5

- **Frequent Itemset:** An itemset whose support is greater than or equal to a minsup threshold

Minimum support is always according to the choice. One can select any minimum support to decide that the itemset is frequent or not.

Class Exercise

Transaction data: Supermarket data

Items = {milk (m), coke (c), pepsi (p), beer (b), juice (j)}
Minimum support $s = 3$
Transactions

1. T1 = {m, c, b}
2. T2 = {m, p, j}
3. T3 = {m, b}
4. T4 = {c, j}
5. T5 = {m, p, b}
6. T6 = {m, c, b, j}
7. T7 = {c, b, j}
8. T8 = {b, c}

Determine the frequent Itemsets.

Association Rule Mining Task

Basket	Product 1	Product 2	Product 3
1	Milk	Cheese	
2	Milk	Apples	Cheese
3	Apples	Banana	
4	Milk	Cheese	
5	Apples	Banana	
6	Milk	Cheese	Banana
7	Milk	Cheese	
8	Cheese	Banana	
9	Cheese	Milk	

The association rule has three measures that express the **degree of confidence** in the rule, i.e.

Support, Confidence, and Lift

Given a set of transactions T, the goal of association rule mining is to find all rules having

- **support ≥ minsup threshold**
- **confidence ≥ minconf threshold**

- **Support**: The number of transactions that include both items {A} and {B} as a percentage of the total number of transactions. It is a measure of how frequently the collection of items (here {A} & {B}) occur together as a percentage of all transactions.
- probability that a transaction contains (A ∪ B)

Example: Support(milk) = 6/9, Support(cheese) = 7/9, Support(milk & cheese) = 6/9.

$$\text{Support} = \frac{(A + B)}{\text{Total}} = \frac{(\text{Milk} + \text{Cheese})}{\text{Total}} = \frac{6}{9} = .6666667$$

Association Rule Mining Task

- **Confidence**: It is the ratio of the no of transactions that includes {A} and {B} to the no of transactions that includes all items in {A}.
- Confidence is the conditional probability that a transaction having A also contains B:- Pr(B|A)

Example: , Confidence(milk => cheese) = (milk & cheese)/(milk) = 6/ 6.

$$\text{Confidence} = \frac{(A+B)}{A} = \frac{(\text{Milk}+\text{Cheese})}{\text{Milk}} = \frac{6}{6} = 1.000$$

Lift: The lift of the rule A=>B is the confidence of the rule divided by the expected confidence, assuming that the itemsets A and B are independent of each other.

Example: Lift(milk => cheese) = [(milk & cheese)/(milk)]/[cheese/Total] = [6/6] / [7/9] = 1/0.777.

$$\text{Lift} = \left(\frac{\left(\frac{(A+B)}{A}\right)}{\left(\frac{B}{\text{Total}}\right)}\right) = \left(\frac{\left(\frac{(\text{Milk}+\text{Cheese})}{\text{Milk}}\right)}{\left(\frac{\text{Cheese}}{\text{Total}}\right)}\right) = \left(\frac{\left(\frac{6}{6}\right)}{\left(\frac{7}{9}\right)}\right) = \left(\frac{1}{.7777778}\right) = 1.28$$

The support of an association rule is **the percentage of groups that contain all of the items listed in that association rule**. The percentage value is calculated from among all the groups that were considered.

A rule expresses a conditional probability. **Confidence** in a rule is calculated by dividing the probability of the items occurring together by the probability of the occurrence of the antecedent.
For example, if B (antecedent) is present, what is the chance that C (consequent) will also be present? What is the confidence for the rule "IF B, THEN C"?

A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.
Confidence(A->B)=Support_count(A∪B)/Support_count(A)

Apriori algorithm is given by R. Agrawal and R. Srikant in 1994 for finding frequent itemsets in a dataset for boolean association rule. Name of the algorithm is Apriori because it uses prior knowledge of frequent itemset properties.

We apply an iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.To improve the efficiency of level-wise generation of frequent itemsets, an important property is used called *Apriori property* which helps by reducing the search space.

Apriori Property

All non-empty subset of frequent itemset must be frequent. The key concept of Apriori algorithm is its anti-monotonicity.

All subsets of a frequent itemset must be frequent(Apriori property). If an itemset is infrequent, all its supersets will be infrequent.

Consider the following dataset and we will find frequent itemsets and generate association rules for them.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

minimum support count is 2
minimum confidence is 60%

Step-1: K=1

(I) Create a table containing support count of each item present in dataset – Called **C1(candidate set)**

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-1: K=1

(II) compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

C1(Candidate set)

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

itemset L1

Step-2: K=2

(I) Generate candidate set C2 using L1 (this is called join step). Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common.

Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent.Check for each itemset)

Now find support count of these itemsets by searching in dataset.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

itemset L1

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

candidate set C2

Step-2:

(II) compare candidate (C2) support count with minimum support count(here min_support=2 if support count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

itemset L1

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

candidate set C2

Step-2:

(II) compare candidate (C2) support count with minimum support count(here min_support=2 if support count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

candidate set C2

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

itemset L2

Step-3:

Generate candidate set C3 using L2 (join step). Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common. So here, for L2, first element should match.
So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}
Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset.(Here subset of {I1, I2, I3} are {I1, I2},{I2, I3},{I1, I3} which are frequent. For {I2, I3, I4}, subset {I3, I4} is not frequent so remove it. Similarly check for every itemset)
find support count of these remaining itemset by searching in dataset

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

itemset L2

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

candidate set C2

Step-3:
(II) Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

candidate set C2

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

itemset L3

Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence –

A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.

Confidence(A->B)=Support_count(A∪B)/Support_count(A)

So here, by taking an example of any frequent itemset, we will show the rule generation.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

itemset L3

Itemset {I1, I2, I3} //from L3

SO rules can be

[I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%

[I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%

[I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%

[I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6*100=33%

[I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7*100=28%

[I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6*100=33%

So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

What are the advantages of Apriori Algorithm?

The Apriori algorithm advantages are as follows:

1. This is the most simple and easy-to-understand algorithm among association rule learning algorithms
2. The resulting rules are intuitive and easy to communicate to an end-user
3. It doesn't require labeled data as it is fully unsupervised; as a result, you can use it in many different situations because unlabeled data is often more accessible
4. Many extensions were proposed for different use cases based on this implementation—for example, there are association learning algorithms that take into account the ordering of items, their number, and associated timestamps
5. The algorithm is exhaustive, so it finds all the rules with the specified support and confidence

What are the disadvantages of Apriori Algorithm?

One of the biggest limitations of the Apriori Algorithm is that it is slow. This is so because of the bare decided by the:

1. A large number of itemsets in the Apriori algorithm dataset.
2. Low minimum support in the data set for the Apriori algorithm.
3. The time needed to hold a large number of candidate sets with many frequent itemsets.
4. Thus it is inefficient when used with large volumes of datasets.

How can we improve the Apriori Algorithm's efficiency?

Many methods are available for improving the efficiency of the algorithm.

1. **Hash-Based Technique:** This method uses a hash-based structure called a hash table for generating the k-itemsets and their corresponding count. It uses a hash function for generating the table.
2. **Transaction Reduction:** This method reduces the number of transactions scanned in iterations. The transactions which do not contain frequent items are marked or removed.
3. **Partitioning:** This method requires only two database scans to mine the frequent itemsets. It says that for any itemset to be potentially frequent in the database, it should be frequent in at least one of the partitions of the database.
4. **Sampling:** This method picks a random sample S from Database D and then searches for frequent itemset in S. It may be possible to lose a global frequent itemset. This can be reduced by lowering the min_sup.
5. **Dynamic Itemset Counting:** This technique can add new candidate itemsets at any marked start point of the database during the scanning of the database.

The blow item are purchased in the Friday evening. Find the **frequent itemsets** and generate **association rules** on this. Assume that minimum support threshold (s = 33.33%) and minimum confident threshold (c = 60%).

Transaction Id	Time	Item
1	Fiday: 6PM	Beer ,Buns, Ketch up
2	Fiday: 7PM	Beer, Buns
3	Fiday: 8PM	Beer ,Coke, Chips
4	Fiday: 8.30PM	Chips, Coke
5	Friday: 9.00PM	Chips, Ketch up
6	Friday: 10.00PM	Beer ,Coke, Chips

The blow item are purchased in the Friday evening. Find the **frequent itemsets** and generate **association rules** on this. Assume that minimum support threshold (s = 33.33%) and minimum confident threshold (c = 60%).

Transaction Id	Time	Item
1	Fiday: 6PM	Beer ,Buns, Ketch up
2	Fiday: 7PM	Beer, Buns
3	Fiday: 8PM	Beer ,Coke, Chips
4	Fiday: 8.30PM	Chips, Coke
5	Friday: 9.00PM	Chips, Ketch up
6	Friday: 10.00PM	Beer ,Coke, Chips

Item	Frquency
Beer	4
Buns	2
Ketch up	2
Coke	3
Chips	4