Association Rule mining
OR
Association Rule Learning

• Type of unsupervised learning that checks for dependency of one data item on another data item and maps accordingly so that it can be more profitable

If A then B [A ⇒ B]
- Product — A
- B — Consequent
- A — antecedent

For ex → Buying Bread [A] — milk / egg / Butter [B] — 3 options a Person might By (Buy) after Buying Bread — maps A to Possible B's To make most Profit

[If customer Buys Bread, he's likely 70% to Buy milk]

THE WHOLE IDEA

ARL
- Apriori [BFS]
- Eclat [DFS]
- F-p Growth Algo [Advance Apriori]

Working 1. **Support** → Frequency of A → $Supp(x) = \frac{Freq(x)}{T}$ ← T = Transaction

OR — Relative frequency that the items in the rule appear together in the dataset.

[0.5] = A and B occur together in half of the Transaction

$$* \ Support(A \Rightarrow B) = \frac{(T \text{ containing Both A and B})}{\text{Total transaction}} \rightarrow P(A \cap B) \quad \frac{freq(A,B)}{N}$$

• **Confidence** → Cond^n Probab of an itemset occuring given that another itemset has already occured. → Ratio of the no of Transaction contain Both itemsets to the Total no. of Transactions containing the First one $\rightarrow \frac{P(A \cap B)}{P(A)}$

[no. of times A ⇒ B rein have found to be TRUE]

$$* \left[A \to B = \frac{Supp(AB)}{Support(A)} \right]$$

Lift → Strength of Association B/w two items in a dataset.

↳ Dividing the Probability of finding the two items together in a Transaction by the Probability of finding each item independently in a Transaction.

↳ $Lift(A \Rightarrow B) = \frac{P(A \cap B)}{P(A) * P(B)}$ ↓ $\frac{Support}{P(A) * P(B)}$

>1 → Strong association	<1 → weak association (less)	To see if we need to spend Time for that Rule

Apriori Algorithm

- Uses frequent item sets to generate 'association Rule'.
- **concept** → Subset of a Frequent itemset must also be a frequent Itemset
- * **Frequent itemset** is an Itemset whose **support value > Threshold value**
- **Bottom-up Approach** → Identify individual items that appear in at least a minimum no. of T, and then Extends these items into larger and larger itemsets, using **JOIN and Prune** strategy.

1] Set a minimum support and confidence threshold for the desired association Rules.

2] Identify all individual Items that appear in atleast minimum no of T
↳ Frequent Itemsets $[S_1, S_2, S_3, S_4]$ [> Threshold value]

3] Use the frequent itemsets to generate candidate Rules.
[Subsets] [$X \Rightarrow Y$] [X, Y both of sets of items]

4] For each candidate rule → calculate the support and confidence of the Rule. [verifying if those subsets on the Basis of confidence level]

5] Select the Rules that meet the min support and confidence Threshold and [output these as the Resulting Association Rules.]

6] Repeat the process, using the Resulting association Rules as the new set of frequent itemsets and generate new candidate rules from these.

↳ [Process continues until no more association rules can be found]

* confidence Threshold = 60%

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

* Support threshold = 20% (> 20%)

Items	Freq	Support	
1	3	60%	→ S_1
2	3	60%	
3	4	80%	
4	1	20%	— X
5	4	80%	

Rem elements → 1, 2, 3, 5

Set of 2 → S_2

Items	Freq	Support	
1,2	1	20%	— X
1,3	3	60%	
1,5	2	40%	
2,3	2	40%	
2,5	3	60%	
3,5	3	60%	

Rem elements → 1, 2, 3, 5 → using S_2 create S_3

Set of 3 → S_3

Items	Freq	Support
1,2,3 X		
1,2,5 X		
1,3,5 →	2	40%
2,3,5 →	2	40%

[At S_3 we can check for Pruning → If subsets are not Frequent Itemsets so that set is not frequent Itemset]

[1,2,3 → {1}, {2}, {3}, {1,2}, {1,3}, {2,3}] from S_2 so (1,2,3) → X

[1,2,5 → {1}, {2}, {3}, {1,2}, {2,5}, {1,5}] so (1,2,5) → X

Set of 4 → S_4

Items	Freq	Support
(1,2,3,5)	1	20% X

Revert Back to S_3

Ans 1,3,5 and 2,3,5

↳ Frequency Itemsets are identified on the basis of support Now find rules from freq itemsets on the basis of confidence.

→ (1,3,5) → (1,3), (3,5), (1,5), (1), (3), (5)

→ (2,3,5) → (2,3), (2,5), (3,5), (2), (3), (5)

verify these subsets using confidence

1,3,5 → (1,3) | (1,5) | (3,5) | (1) | (3) | (5)

1⇒3 (1,3) Role 1 →

S → (1-S) (S recommends 1-S)
Subset Proper

Conf-level → support(I) / support(S)

R1
(1,3) → 2/5 / 3/5 = 2/3 > 60% — ✓

R2
(1,5) → 2/5 / 2/5 = 100% > 60% ✓

R3
(3,5) → 2/3 > 60% ✓

R4
(1) ⇒ 2/5 / 3/5 > 60% ✓

R5
(3) = 2/5 / 4/5 = 50% < 60% ✗

R6
(5) = 2/4 = 50% < 60% ✗

R

library(arules) // apriori

library(arulesviz) // plot(rules)

data(Groceries)

- data_as_list ← as(Groceries, 'list')
- data_as_transaction ← as(Groceries, 'transactions')

rules ← apriori(data = Groceries, parameters = list(support = 0.020, confidence = 0.15))

inspect(rules) OR inspect(sort(rules, by = 'lift'))

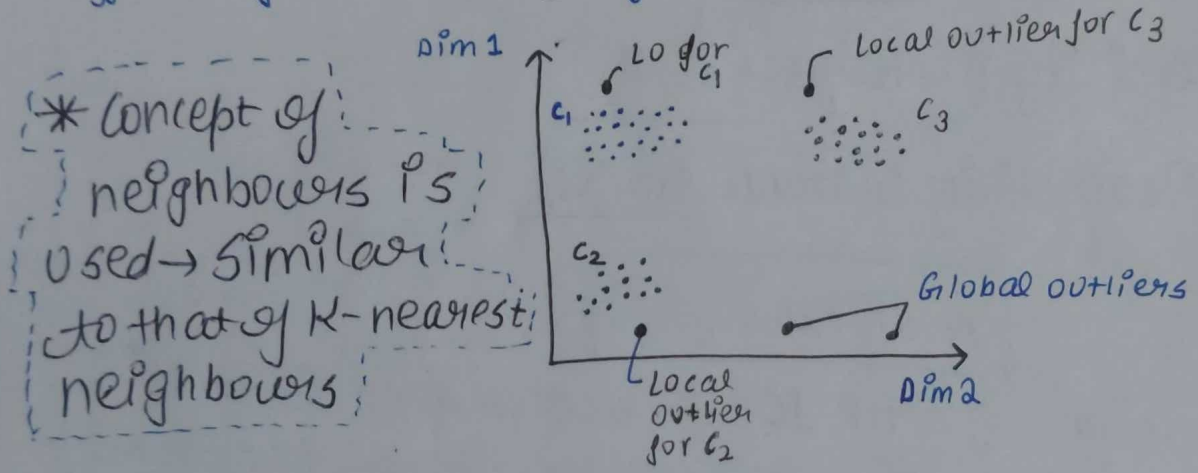
plot(rules)

• Local outlier factor Anamoly Detection → LOF

- Z-score or IQR
- Boxplots
- LOF

Global outliers → Datapoints which are 'significantly' 'different' from the rest of the Dataset.

Local outliers → Datapoints which are significantly different from their [neighbours in the dataset]



* Concept of neighbours is used → similar to that of K-nearest neighbours

LOF → • A score (scalar value) = LOF is the Deciding factor

For a given Dataset

$$D_n = \{(x_i, y_i) \mid x_i \in R^2,\ y_i \in \{xyz\}\}$$

$$* \quad LOF(x_i) = \frac{\sum_{x_j \in N(x_i)} lrd(x_j)}{|N(x_i)|} \times \frac{1}{lrd(x_i)}$$

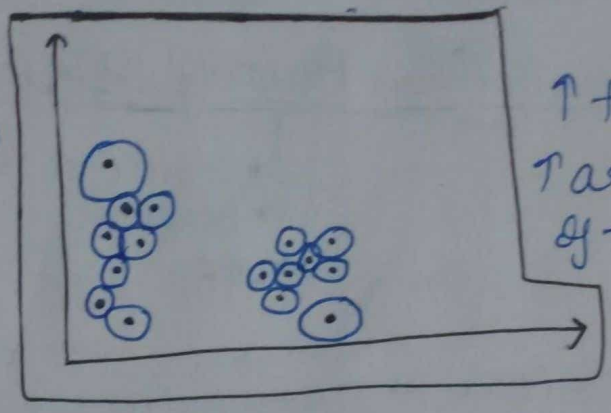
$|N(x_i)|$ = No. of elements in the neighbourhood of x_i

$lrd(x_i)$ = Local reachability Density of x_i

• WORKING → (1) LOF is assigned to each Data points. These assigned LOF scores of each data points are compared to find outliers.

(2) ↑ value of LOF of any DP → ↑ chances of that DP being an outlier

Radius of circle ∝ (=) to the LOF scores of each DataPoint.



↑ the Radius, ↑ are the chances of that DP being an Outlier.

2] Parameters for calculating LOF

i) k-distance (x_i)

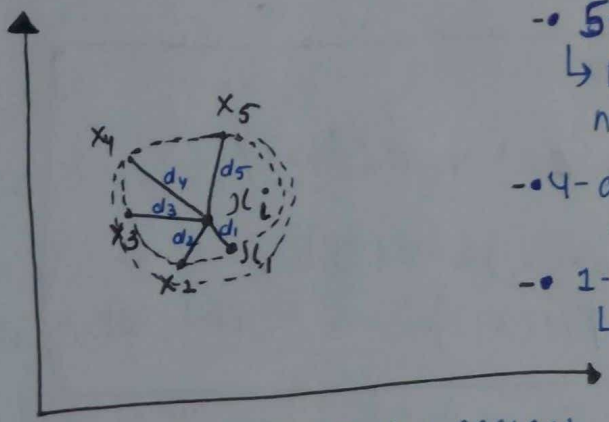
ii] Nearest Neighbor $N_k(x_i)$

iii] Reachability Distance (x_i, x_j)

iv] Local Reachability Distance (lrd(x_i))

k-distance (x_i)

The k-Distance of a DP x_i in a dataset is the Distance of the k^{th} nearest neighbour of x_i from x_i



- 5-distance (x_i)
  - ↳ Distance B/w x_i and the 5th nearest neighbour of x_i i.e d_5
- 4-distance (x_i)
  - ↳ d_4
- 1-distance (x_i)
  - ↳ d_1

→ x_1, x_2, x_3, x_4, x_5 are nearest neighbour of x_i

→ d_i is the distance b/w x_i and x_j where $x_j \in (x_1, x_2, x_3, x_4, x_5)$

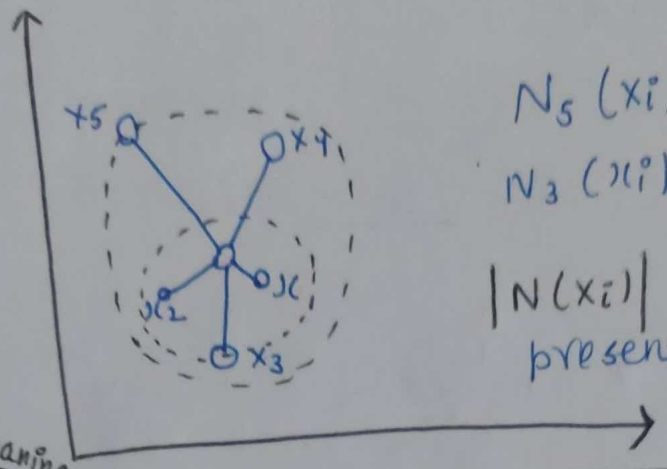
→ $d_1 < d_2 < d_3 < d_4 < d_5$

X-data

```
sklearn.neighbors
import LocalOutlierFactor

lof = local outlier factor (n_neighbor=20, contamination=0.1)

outlier_scores = lof.fit_predict(X)
    ↳ 1 → Inliner (+ve)
    ↳ -1 → Outlier (-ve)

outlier_indices = np.where(outlier_scores == -1)
    ↳ np.delete(X, outlier_indices, axis=0)
```

Nearest Neighbour $N_K(x_i)$

- Just like **K-Nearest Neighbours** of D.P x_i denoted by $N_K(x_i)$ is a SET of all Points that belong to the k^{th} nearest neighbour of x_i (i.e → points in the k^{th} neighborhood of x_i)



$N_5(x_i) = \{x_1, x_2, x_3, x_4, x_5\}$

$N_3(x_i) = \{x_1, x_2, x_3\}$

$|N(x_i)|$ = Total no. of data points present in the neighbor of x_i (5) for given K value.

actual meaning

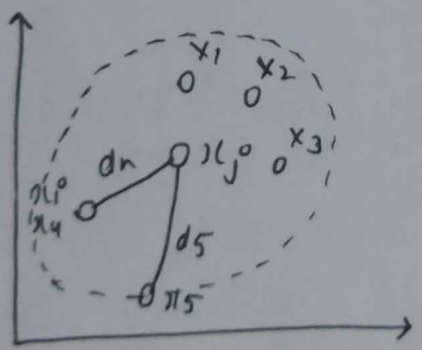
Reachability - Distance (x_i, x_j)

2 points K Beech ka actual distance ya source wale ka K-distance (Total neighbours)

→ The R-D of a DP x_j from x_i is the max of K-distance of x_j and the actual Distance B/w x_i and x_j

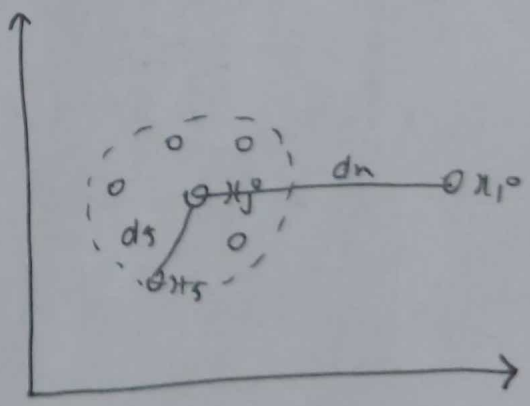
$$R\text{-}D(x_i, x_j) = \max(K\text{-distance}(x_j), dist(x_i, x_j))$$

Case 1 (x_i lies within the cluster)



$\max(d_5, d_n) = d_5$

Case 2 (x_i lies Outside the cluster)



$\max(d_5, d_n) = d_n$

Local Reachability Density (lrd(xi))

- LRD of a Datapoint x_i is the Inverse of the [Avg RD of x_i from its neighboorhood.

* In short → It measures How CLOSE the neighborhood of at a points of x_i are from it.

* If lrd(x_i) is HIGH → x_i is in dense neighboorhood
* If lrd(x_i) is Low → x_i is in sparse neighborhood

$$lrd(x_i) = \frac{1}{\sum_{x_j \in N(x_i)} \left\{ \frac{R.D(x_i, x_j)}{|N(x_i)|} \right\}}$$

(set; Total Number)

$$LOF(x_i) \rightarrow \underbrace{\frac{\sum_{x_j \in N(x_i)} lrd(x_j)}{|N(x_i)|}}_{A} \times \underbrace{\frac{1}{lrd(x_i)}}_{B}$$

a) LOF(x_i) is large when A is large and B is small
   → [Outlier]

b) LOF(x_i) is small when A will be small and B will be large → [Not Outlier]

- [~1] Similar Density as neighbors,
- [<1] Higher Density than neighbors (Inliner)
- [>1] Lower Density than neighbors (outliers)

Imp → we can either visualize the circles to show Outliers or we can show red Dots.