# AML

Part - 1 ← → Part - 2 ✓

(90%)

↳ Recommender Systems

↳ Time - Series Analysis

## * Recommender Systems:

Netflix

Youtube

Amazon → $\boxed{\text{Laptop}}$ → Added to Cart

Instagram

↓ Cart Page

Bumble

Frequently Bought Together 'Keyboard' 'Mouse'

(Recommendations)

• R.S. predicts what a user might be interested in by:

[FRIENDS] ✓

[Jolly LLB - 3] ✓

} Comedy Movies / Web - Series

① past purchasing / browsing history

② similar users behaviour.

③ similar products.

Why companies need R.S. ?

① Improve User Experience
② Increase Revenue / Profit
③ Reduce Search Time

USER
ACTIVITY - FUNNEL :

Opens App

↓

View / Browse

↓

Add to Cart

↓

Make Payment

↓

Order Placed

$\boxed{\text{New User}}$

↓ Playstore Download / Install

$\boxed{\text{BLINKIT}}$

↓ Register / Sign-up

Mob. No ⟶ OTP ⟶ Details.

$\boxed{\text{New User}}$

↓ 1st order

Database

→ 1 row added

Jan 2025 ⟶ $\boxed{\text{New User Acquisition}}$

Feb 2025 | Feb 2025

Place 2nd order | Place NO order

→ Retained User | → CHURN

(Retention)

↓ March 2025

2nd order [Re-activated Customer]

* Types of Recommender Systems:

① Popularity-based:

Trending on Insta ⟶ SPOTIFY

- Recommends 'most bought' or 'most-viewed' item.
- No personalization
- Used for new users.

② Content-based Filtering:

- Recommends items / products as per what user liked/watched earlier.

Ex: Comedy shows ⟶ Comedy MOVIES

③ Collaborative Filtering:

- People like you bought this / watched this.

- Learns from user behaviour, ratings, purchase history

SPOTIFY
NETFLIX
YOUTUBE
INSTAGRAM
AMAZON
} Hybrid Recommender System

- Market - Basket Analysis
  ↳ Improves:

UPSELL :
CROSS - SELL :

1. Product Placement / Inventory
2. Combo offers and bundles
3. Cross - selling recommendations

GRAB → zomato
↳ ola / uber
↳ zepto
↳ paytm

→ SUPER APP

upsell
- user → 100 Grabtaxi rides $\xrightarrow{\text{UPSELL}}$ 200 G.T. rides

cross-sell
- user → 100 Grabtaxi rides → 5 GRAB Food orders

60 Rs/-

onion - 30/-
tomato - 20/-
potato - 10/-

[Market - Basket Analysis]

combo (onion, tomato, potato)
-55/-

→ APRIORI ALGORITM :

Core Idea :

(Milk, Bread, Butter)

| Transaction ID | Items Purchased |
|---|---|
| 1 | Milk, Bread, Eggs |
| 2 | Bread, Butter |
| 3 | Milk, Bread, Butter |
| 4 | Bread, Eggs |
| 5 | Milk, Bread, Butter, Cookies |
| 6 | Milk, Cookies |
| 7 | Bread, Butter, Eggs |
| 8 | Milk, Bread |
| 9 | Bread, Cookies |
| 10 | Milk, Butter, Bread |

Steps :

① Calculates Support for each item.

② Form 2-item combination and calculate support again.

③ Generate Few Rules

- Confidence
- Lift → >1 (Strongly Associated)
  → <1 (Weakly Associated)

$$\boxed{\text{Support} = \frac{\text{count of item}}{\text{transactions count}}}$$

① Calculate Support

| Item | Count | Support |
|---|---|---|
| Milk | 6 | 0.6 ✓ |
| Bread | 9 | 0.9 ✓ |
| Butter | 0.5 | 0.5 ✓ |
| Eggs | 4 | 0.4 ✓ |
| Cookies | 3 | 0.3 ✓ |
| ✗ Curd | 2 | 0.2 ✗ |

Assume min. support as 0.3 (30%)

② Form 2-item sets & calculate support

| | | |
|---|---|---|
| Milk, Bread | 5 | 0.5 ✓ |
| Milk, Butter | 3 | 0.3 ✓ |
| Milk, Eggs | 2 | 0.2 ✗ |
| Bread, Butter | 5 | 0.5 ✓ |

Bread, Eggs 2 0.2 ✗

$D = \{1, 2, 3, 4, 5, \ldots\ldots n\}$

No. of subsets $= 2^n$

3 Confidence. $(X \rightarrow Y)$

Support (Milk, Bread)

Support (Milk)

Confidence $\simeq (0, 1)$

$$\text{Confidence } (X \rightarrow Y) = \frac{\text{Support } (X \cap Y)}{\text{Support}(X)}$$

$$= \frac{0.5}{0.6} \simeq 0.83$$

4 LIFT

$$\text{Lift} = \frac{\text{Confidence } (X \rightarrow Y)}{\text{Support } (Y)}$$

$$= \frac{0.83}{0.9} < 1$$

(Weakly Associated)

Y , X
(2) (Bread, Butter) $\longrightarrow$ 0.5

(Butter) $\longrightarrow$ 0.5
X

$$\text{Confidence} = \frac{0.5}{0.5} = 1.$$

$$\text{Lift} = \frac{\text{Confidence}}{\text{Support (Bread)}} = \frac{1}{0.9} > 1$$

(Strongly Associated)

Apriori Algorithm Output tells –

1. Frequent itemsets
2. Association Rules $(A \longrightarrow B)$

Limitations :

1. Slow with large datasets
2. Not suitable with continous data

• ASSOCIATION RULES :

↳ finding relations between frequent item-sets

Notation : $\boxed{X \longrightarrow Y}$

→ People who buy 'X' have a very high likelihood to buy 'Y'.

Ex :

$X \rightarrow \{\text{milk, bread}\}$

$Y \rightarrow \{\text{jam, eggs}\}$

{milk, bread} $\longrightarrow$ {jam, eggs}

(Antecedent) (Consequent)

if then

Pizza $\longrightarrow$ coke Y

X $\longrightarrow$ garlic bread Y

## FORMULAS

① $$\boxed{\text{Support}(X) = \frac{\text{No. of transactions in which } X \text{ is present}}{\text{Total transactions}}}$$

$$\text{Support} = \frac{x}{N}$$

② Confidence

$X \rightarrow Y$ ?

$$\boxed{\text{Confidence}(X \rightarrow Y) = \frac{\text{Support}(X \text{ and } Y)}{\text{Support}(X)}}$$

$= P(Y \mid X)$ (logically)

## ③ Lift :

$$\boxed{\text{Lift}(X \rightarrow Y) = \frac{\text{Confidence}(X \rightarrow Y)}{\text{Support}(Y)}}$$

$$= \frac{P(X \text{ and } Y)}{P(X) \cdot P(Y)} \quad \text{(intuitively)}$$

- Lift $(X \rightarrow Y) = 1$ , X and Y are independent
- Lift $(X \rightarrow Y) < 1$ , unlikely to be bought together
- Lift $(X \rightarrow Y) > 1$ , likely to be bought together

Lift varies from (0 , infinity)

Original Data:

Invoice    Stock Code    Description    Quantity

↓ To prepare data for Apriori algorithm

① Filtered OK Data

② Grouping on invoice No. and Description → Quantity Sum

Transaction 1 Mug 6 ←

Transaction 1 Mug 4 ←

[Invoice, Description] . quantity. sum()

→ unstack()

| invoices | product | total quantity |
|---|---|---|
| 1 | Milk | 6 |
| 1 | Bread | 10 |
| 1 | Butter | 2 |

unstack()

↓ pivot table (rows → columns)

| invoices | Milk | Bread | Butter |
|---|---|---|---|
| 1 | 6 | 10 | 0 |

" Each product becomes a separate column "

↓

| invoice No | Milk | Bread | Butter |
| --- | --- | --- | --- |
| 1 | 1 | 1 | 0 |

→ Association Rules :

→ Co-occurence

If any transaction / invoice have 1 single item, remove those

→ Invoices → at least 2 distinct items are present