Recommender Systems

Recommender systems are systems that are designed to recommend things to the user based on many different factors. These systems predict the most likely product that the users are most likely to purchase and are of interest to.

Companies like Netflix, Amazon, Instagram, etc. use recommender systems to help their users to identify the correct product or movies for them.

**Why recommender systems?**

* Increase in revenue based on personalization.
* Better user experience.
* More time spent on the platform
* Help websites improve user engagement.

**Recommender systems applications**

* Netflix to recommend movies
* E-commerce websites to recommend the products.
* Social media platforms to recommend feeds/ blogs/news/songs…etc. Ex: Instagram, Facebook.
* Food recommendations by Zomato, Swiggy.
* Songs recommendations by Spotify and Wynk music.
* Dating apps recommend people.

A few examples of how the recommendation system works

* Amazon Recommender System
  + Amazon sells virtually all categories of products such as books, CDs, software, electronics, and so on.
  + The recommendations on Amazon are provided on the basis of explicitly provided

ratings, buying behavior, and browsing behavior.

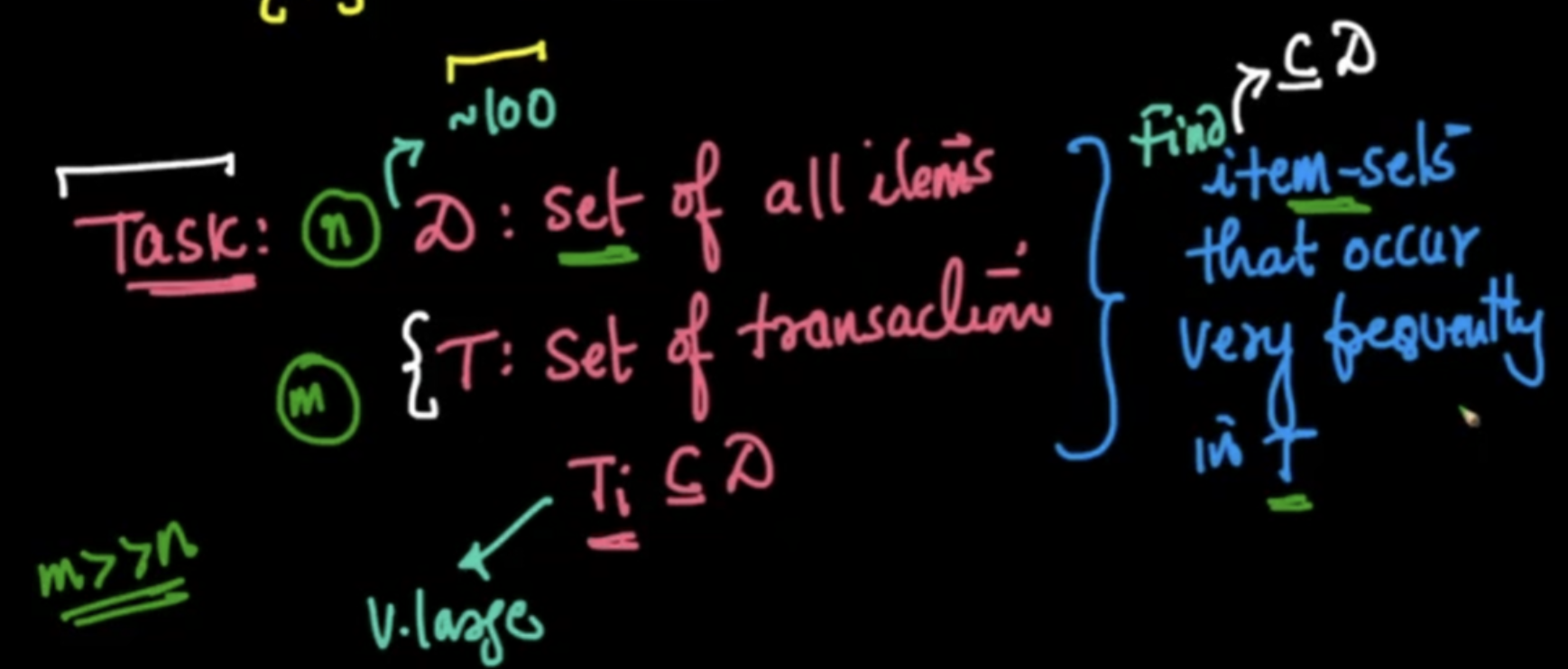
* + The ratings on Amazon are speciﬁed on a 5-point scale, with the lowest rating being 1-star, and the highest rating being 5-star.
  + The customer-speciﬁc buying and browsing data can be easily collected when users are logged in with an account authentication mechanism supported by Amazon.
  + Recommendations are also provided to users on the main Web page of the site, whenever they log into their accounts.
  + Based on the ratings given for each item amazon finds the similarity between items and recommends the items using collaborative filtering.
* Netflix Movie Recommender System
  + The recommendations system estimates the probability of a user watching a particular title based on the following factors:
    - Viewer interactions with Netflix services like viewer ratings, viewing history, etc.
    - Information about the categories, year of release, title, genres, and more.
    - Other viewers with similar watching preferences and tastes.
    - Time duration of a viewer watching a show
    - The device on which a viewer is watching.
    - The time of the day a viewer watches -This is because Netflix has the data that there is different viewing behavior based on the time of the day, the day of the week, the location, and the device on which a show or movie is viewed.
  + For every new subscriber, Netflix asks them to choose titles they would like to watch. These titles are used as the first step for personalized recommendations.
  + Later as viewers continue to watch over time the recommendations are powered by the titles they watched more recently along with other factors mentioned above.
  + To recommend titles for the users Netflix uses a various number of algorithms along with a content-based recommendation system.
* Google News Recommender System
  + The Google News Recommender system is able to recommend news to users based on their history of clicks. The clicks are associated with speciﬁc users based on identiﬁcation mechanisms enabled by Gmail accounts.
  + The act of a user clicking on a news article can be viewed as a positive rating for that article.
  + Collaborative recommendation algorithms are applied to the collected ratings so that inferences can be made about the personalized articles for speciﬁc users.

**Types of recommender systems**

1. Apriori Algorithm
2. Content-based filtering system.
3. Collaborative-based filtering system.
4. Similarity-based filtering system.
5. Matrix factorization
6. Popularity-based recommender system.
7. Regression-based recommender system
8. Group-based recommender system

## **Market-Basket Analysis**

* Market basket analysis is used to analyze the combination of products that have been bought together.
* This is a technique used for purchases done by a customer. This identifies the pattern of frequent purchases of items by customers.
* This analysis can help to promote deals, offers, sale by the companies, and data mining techniques helps to achieve this analysis task.
* For Example, At huge superstores like Walmart, we will have products across many categories: Daliy essentials, Food products (like Milk, Butter, jam, bread, etc), Beauty Products, Toys, etc.
* Even then, we're looking at a **few hundreds** of products, or at the max, a few thousand, as opposed to the world of e-commerce where there may be lakhs or millions of products.
  + - Let n be the Total No of distinct products
    - Let's define D as the set of all the products we have, then, D={1, 2, 3, ..., n}
* Consider a customer that is done selecting the items they want, and are proceeding towards the billing counter. They present their **basket** full of products and the cashier scans the product bought and its quantity. This is called a **transaction**, denoted by T.
  + No of transactions m >> No of distinct items n



* This technique of analyzing transaction data to give recommendations to the customer is called **Market Basket Analysis**.
* Other applications of Market Basket Analysis:
  + **Bio-informatics**
    - If two chemical components c1 and c3 occur frequently within different proteins, then we can find out that perhaps there is some relation between these components.
    - If two gene sequences ATTC and AGTC occur frequently, in the sequence of some mammals, then we can find out that perhaps there is some relation between them.
  + **Medicine**
    - If we find that according to a doctor's prescription, medicines m1, m2, and m3 are being prescribed frequently, then it means that together they form some combination drug, which can cure a certain ailment.
  + **Finding similar webpages/web usage mining**
    - If in a single session many users are visiting the same webpages (w1, w2, w3), then perhaps they are related in nature

**Apriori Algorithm**

* Apriori algorithm refers to the algorithm which is used to calculate the association rules between objects. It means how two or more objects are related to one another. we can say that the apriori algorithm is an association rule learning that analyzes that people who bought product A also bought product B.
* It is a Frequency-based algorithm. Generally, the apriori algorithm operates on a database containing a huge number of transactions.

Ex: People who bought iPhones also bought Airpods.

**Examples** where the apriori algorithm can be used:

* Telecommunications
* Banking / Insurance
* Medical
* E-Commerce
* Retail

**To implement Apriori Algorithm in code:**

* We have a built-in function that implements apriori for us, under the `mlxtend.frequent\_patterns` library.
* We need to specify a minimum support threshold value, as a parameter of this function.
* Since our column names represent the items, we use `use\_colnames=True`
* This will give us the most frequent item sets, so we sort them in descending order also, using `sort\_values()`
* Using the Apriori algorithm, we get a sense of which item-sets are frequent.
* We have another tool in the Market Basket Analysis, called the **Association rule** using which we can find relations between these frequent item sets.

### **Association rules**

### Association Rules are widely used to analyze retail basket or transaction data and are intended to identify strong rules discovered in transaction data using measures of patterns, based on the concept of strong rules.

### There are various metrics in place to help us understand the strength of the association between an antecedent and consequent:

* **Working of association rule:**
  + Consider we have our set of items: D ={ 1, 2, 3, ..., n}
  + Let's define X and Y as another set of items, as follows:
    - X= {1,2,3}
    - Y = {4,6}
  + If the item-set {1, 2, 3, 4, 6} is a **frequent itemset**, then according to the **Association Rule** of Market Basket Analysis, we can say that people who buy X, have a very high likelihood to buy Y also.
  + This can be written as X -> Y and **read as "If X, then Y".**
* For example, **If** a person buys beer, **then** there is a high tendency of buying diapers.
* X → Y is not the same as Y → X
* When we say, People buying beer, have a high tendency of buying diapers, that **does not imply** that people buying diapers have a high tendency of buying beers.
* To set these apart, we have the following terminologies in place:
  + **Antecedent (If):** The items on the LEFT ie., the item which the customer buys.
  + **Consequent (Then):** The items on the RIGHT ie., the item which the customer follows to buy.
* There are a couple of different metrics to know how strongly two item sets are associated:
  + Support
  + Confidence
  + Lift
  + Leverage
  + Conviction

**1. Support:**

It is calculated to check how popular a given item is. It is measured by the proportion of transactions in which an item set appears. It is also used to measure abundance or frequency.

Drawback: If ‘x’ is very popular then many items will have high support w.r.t to x.

**2. Confidence:**

It is calculated to check how likely item X is purchased when item Y is purchased. This is measured by the proportion of transactions with item X, in which item Y also appears.

Drawback: If ‘y’ is very popular then it will have high confidence w.r.t to many items.

**3. Lift:**

It is calculated to measure how likely item Y is purchased when item X is purchased while controlling for how popular item Y is.

* lift(X→Y) = 1 if X and Y are independent
* lift(X→Y) < 1, unlikely to be bought together**.**
* lift(X → Y) > 1, likely to be bought together
* lift value ranges from 0 to infinity.
* Gives Bi-directional recommendations.

**Steps to implement the apriori algorithm:**

### Create a frequency table of all the items that occur in all transactions.

1. Create a pivot matrix representing 1 if the item is present and 0 if the item is not present.
2. Encode the matrix, return 0 for all the items that have the value less than or equal to 0 and return 1 for all the values that are greater than or equal to 1.
3. Use the library mlxtend.frequent\_patterns and import apriori.
4. Calculate frequent itemsets using the metric min\_support.
5. Use the library mlxtend.frequent\_patterns and import association\_rules
6. With the help of association\_rules select the appropriate metric (ex: support, confidence, lift … default = confidence) to recommend the items.

**Advantages:**

1. It is the most simple and easy to understand and implement.
2. It doesn’t require labeled data as it is completely unsupervised and hence this can be used for many different situations as we can find unlabeled data quite often.
3. It is used to calculate large item sets.

**Disadvantages:**

1. Apriori algorithm is an expensive method to find support since the calculation has to pass through the whole database.
2. It is computationally expensive.
3. Complexity grows exponentially.
4. Cold start problem.