Day 55

DIY Solution

Q1. What is a recommendation system? How does it work?

Answer: A recommendation engine is a data filtering tool using machine learning algorithms to recommend the most relevant items to a particular user or customer. It operates on the print patterns in consumer behavior data, which can be collected implicitly or explicitly.

One of the crucial components behind the working of a recommendation engine is the recommender function, which considers specific information about the user and predicts the rating that the user might assign to a product/component.

It uses specialized algorithms and techniques to support even the largest product catalogs. Driven by an orchestration layer, the recommendation engine can intelligently select which filters and algorithms to apply in any given situation for a specific customer. It allows marketers to maximize conversions and also their average order value.

A recommendation engine processes data through the below four phases:

Collection: Data collected here can be explicit, such as data fed by users (ratings and comments on products), or implicit, such as page views, order history/return history, and cart events.

Storing: The type of data you use to create recommendations can help you decide the kind of storage you should use, like the NoSQL database, a standard SQL database, or object storage.

Analyzing: The recommender system analyzes and finds items with similar user engagement data by filtering it using different methods such as batch analysis, real-time analysis, or near-real-time system analysis.

Filtering: The last step is to filter the data to get the relevant information required to provide recommendations to the user. And to enable this, you will need to choose an algorithm suiting the recommendation engine (choose the appropriate algorithm).

Q2. What is the difference between Collaborative and Content-Based Recommender Systems?

Answer: Collaborative methods for recommender systems are based solely on the past interactions recorded between users and items to produce new recommendations.

Unlike collaborative methods, content-based approaches use additional information about users and items.

A real-world example of a company using collaborative filtering is song suggestion. They would create a 'station' of recommended songs by observing what bands and individual tracks the user has listened to regularly and comparing those against the listening behavior of other users. This approach leverages the behavior of users.

Many other uses the content-based approach of recommender systems. It uses the properties of a song or artist to see a 'station' that plays music with similar properties. User feedback is used to refine the station's results, deemphasizing specific attributes when a user 'dislikes' a particular song and other attributes details 'likes' a song.

Q3. What are similarity measures? Explain their types.

Answer: Similarity is measured using the distance metric. The nearest points are the most similar, and the farthest points are the least relevant. The similarity is subjective and is highly dependent on the domain and application. For example, two movies are similar because of genre, length, or cast. Care should be taken when calculating distance across dimensions/features that are unrelated. The relative values of each element must be normalized, or one-part could end up dominating the distance calculation.

Minkowski Distance: When the dimension of a data point is numeric, the general form is called the Minkowski distance.

$\mathrm{d}(x, y) = (\sum_i^n (|x_i - y_i|)^q)^{\frac{1}{q}}$

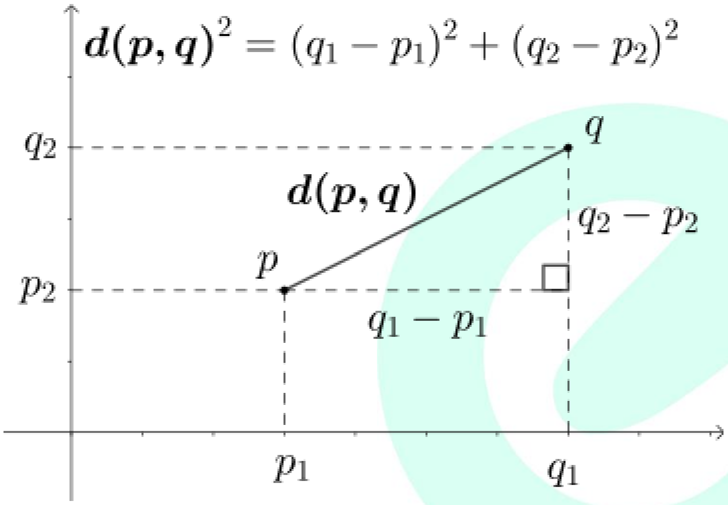
It is a generic distance metric where Manhattan (r = 1) or Euclidean (r = 2) distance measures are generalizations of it.

Manhattan Distance: The distance between two points measured along axes at right angles.

$\mathrm{d}(x, y) = \sum_i^n |x_i - y_i|$

It is also called rectilinear distance, L1-distance/L1-norm, Minkowski's L1-distance, city block distance, and taxicab distance.

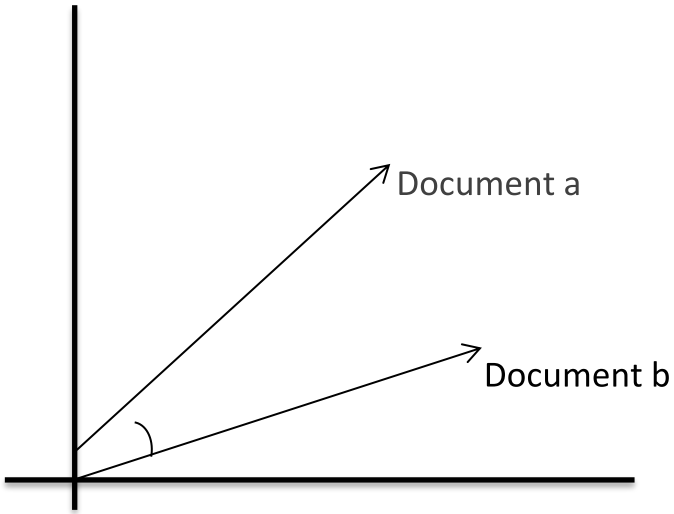
Euclidean Distance: The square root of the sum of squares of the difference between the coordinates and is given by the Pythagorean theorem.



$\mathrm{d}(x, y) = \sqrt{\sum_i^n (x_i - y_i)^2}$

It is also called as L2 norm or ruler distance. In general, the default distance is considered Euclidean distance.

Cosine Similarity: Measures the cosine of the angle between two vectors. It is a judgment of orientation rather than magnitude between two vectors with respect to the origin. The cosine of 0 degrees is 1, which means the data points are similar, and the cosine of 90 degrees is 0, which means data points are dissimilar.



$$\sin(x, y) = cos\ \theta = \frac{x.y}{||x||.||y||}$$

Cosine similarity is subjective to the domain and application and is not an actual distance metric. For example, data points [1,2] and [100,200], are shown as similar with cosine similarity, whereas the Euclidean distance measure shows them as being far away from each other (i.e., they are dissimilar).

Pearson Coefficient: It measures the correlation between two random variables and ranges between [−1, 1].

$$r = \frac{\sum(x-x^{\wedge})(y-y^{\wedge})}{\sqrt{\sum(x-x^{\wedge})^2 \sum(y-y^{\wedge})^2}}$$

If the value is 1, it is a positive correlation, and if −1, there is a negative correlation among variables.

Jaccard Similarity: In the other similarity metrics, we discussed some ways to find the similarity between objects, where the objects are points or vectors. We use Jaccard similarity to find similarities between finite sets. It is defined as the cardinality of the intersection of sets divided by the cardinality of the union of the sample sets.

$$d(x, y) = \frac{|A \cap B|}{|A \cup B|}$$

Hamming Distance: All the similarities we discussed were distance measures for continuous variables. In the case of categorical variables, Hamming distance must be used.

Hamming distance

$$D_H = \sum_{i=1}^{k} |x_i - y_i|$$

x = y → D = 0

x ≠ y → D = 1

X	Y	Distance
Male	Male	0
Male	Female	1

If the value (x) and the value (y) are the same, the distance D will be equal to 0; otherwise, D = 1. If we have data that is binary (i.e., classification), one will go for Hamming distance. The lower value means high similarity, and the higher value means less similarity between variables. For example, the Hamming distance between 1101111 and 1001001 is 3, while the Hamming distance between 'Batman' and 'Antman' is 2.

Q4. What Challenges are involved with recommendation engines?

Answer:

There are various challenges involved with recommendation engines, such as,

Synonymous Names: The challenge of synonymy arises when a single product or item is represented with two or more different names or listings of items (for instance, action movies or action films) having a similar meaning. In such a case, the recommendation system is not capable of recognizing whether the terms show various items or the same item.

Scalability: One of the other issues with recommendation systems is the scalability of algorithms having real-world datasets. In most cases, the traditional approach has become overwhelmed by the multiplicity of products and clients, leading to dataset challenges and performance reduction.

Latency Challenges: Latency issues arise when new products are added more frequently to the database of a recommendation engine. Still, already existing products are recommended to users since newly added products are not rated. Companies can use either a collaborative filtering method or the category-based approach in combination with user-item interaction to deal with the issue.

Privacy: In most cases, customers need to feed their personal information to the recommendation system for tailor-made and beneficial services. However, it causes various data privacy and security issues, making the customers feel hesitant to feed their personal data into recommendation systems.

But since the recommendation system is bound to have the customer's personal information and use it to the fullest to offer personalized recommendation services, they must navigate the situation with extra care and ensure trust among their users.

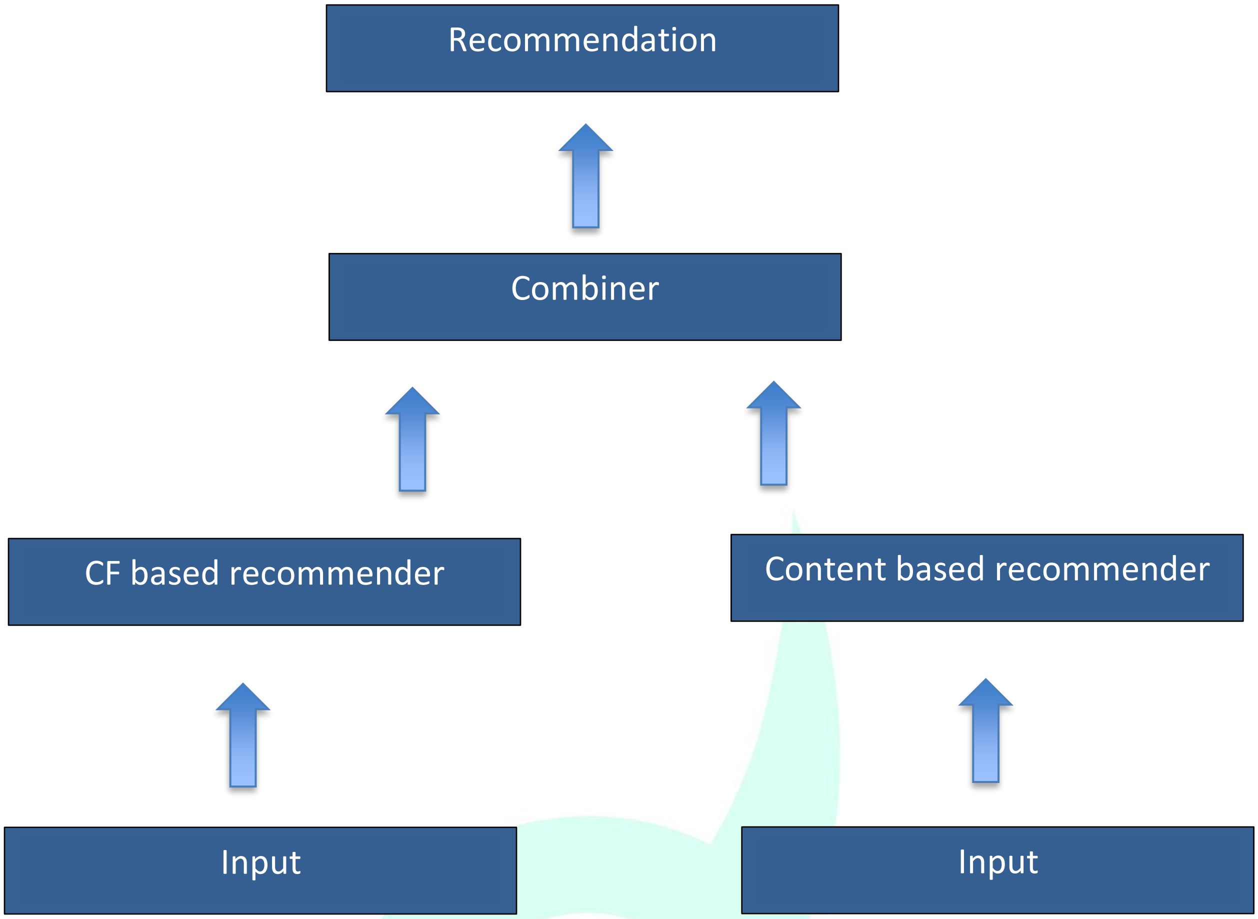
Issue of Sparsity: There are instances when users do not give ratings or reviews to the purchased products, making the rating and review model relatively sparse and leading to data sparsity issues. It leads to a decrease in the possibility of finding a set of customers with similar ratings or interests.

Now that we have discussed the challenges involved in recommended systems, let us come to the advantages and applications.

Q5. What is a hybrid recommendation system?

Answer:

Hybrid Recommendation engines are essentially the combination of diverse rating and sorting algorithms. For instance, a hybrid recommendation engine could use collaborative filtering and product-based filtering in tandem to recommend a broader range of products to customers with accurate precision.



Netflix is an excellent example of a hybrid recommendation system as they make recommendations by:

- Comparing the watching and searching habits of users and finding similar users on that platform, thus making use of collaborative filtering

- Recommending such shows/movies which share common characteristics with the ones rated highly by the user. It is how they make use of content-based filtering.

Compared to pure collaborative and content-based methods, hybrid methods can provide more accurate recommendations. They can also overcome the common issues in recommendation systems, such as cold start and data paucity troubles.