Session – Summary

Recommendation System

When you go to an ecommerce platform, one of the most important factors that induces you to make the actual purchase is how quickly are you able to locate the item you want to buy. One way in which ecommerce platforms achieve is through recommendation systems (also known as recommender systems), e.g. the 'customers who bought this item also bought…' section on Amazon.

Types of Recommendation Systems

One popular technique of recommendation/recommender systems is **content-based filtering.** Content here refers to the content or attributes of the products you like. So, the idea in content-based filtering is to tag products using certain keywords, understand what the user likes, look up those keywords in the database and recommend different products with same attributes.

Also, note that some features are obviously more important than others. For example, in books, the author name may be a more important feature than genre and price, because people tend to read books by the same author. This suggests that the system needs to assign weights to the features.

This technique is called content-based filtering because you need to understand the content descriptions very well to make good recommendations. The **basic idea of content-based filters** is to map relevant content based on certain **features**. For example, books may be tagged by features such as author name, price, genre, etc. Another example is movies which may be tagged by attributes such as actors, genre, box office collection, etc. Another example of content-based systems is YouTube. When you watch videos of a certain band, say Maroon 5 or Red Hot Chilli Peppers, the system finds other videos which are tagged by the band name. Note that, in this case, there is one feature which is more important than all others, i.e. the band name. There may be other features like genre, video length, musical instruments, etc., but the band name is probably the most important.

The other extremely popular technique is **collaborative filtering.** The basic idea of collaborative filters is that similar users tend to like similar items. It is based on the assumption that, if some users have had similar interests in the past, they will also have similar tastes in the future.

So, what is remarkable about this technique? In collaborative systems, the main advantage is that you do not need to know the content in detail. Amazon's product '*Customers who bought this item also bought…*' is a popular example of collaborative filtering. When you buy *Hamlet* by Shakespeare, it recommends other books which were bought by other people who also bought *Hamlet.* In this case, the system may not even care about the genre or the price of the books being recommended; it simply recommends them to you because other people similar to you bought them.

Content-based Filtering

So, in content-based systems, you need a way to store two things — the description of the features of the item, and the interest of the user with respect to that feature. In this case, the name of the author is one of the features. The other features can be the genre of the book, price, the number of pages, etc.

To solve the first problem, each item is described through an item vector. Thus, if there are n attributes that can be used to describe any item set, the item vector for an item will be a vector of size n having 0 or 1 corresponding to each characteristic. Similarly, the taste and preferences of a user is represented by a user vector where rating points are assigned corresponding to each characteristic.

The dot product of the two vectors depicts the compatibility of the item with the user. The higher the value of this dot product, the more is the chance of the user liking the particular product.

Collaborative Filtering

User-based Collaborative Filtering

In user-based collaborative filtering, you try to find users who have similar tastes and preferences, based on the past ratings they have given to the same items. A similar correlation measure or cosine similarity measure may be applied to find this similarity. Finally, you can use the weighted sum of the ratings provided by like-minded users for an item to predict what rating may be provided by a user.

Item-based Collaborative Filtering

Another extremely successful type of collaborative filtering algorithm is item-based collaborative filtering. Item-based systems are heavily used by companies such as Amazon and various movie recommendation sites.

In user-based collaborative filters, you measure the similarity between users. The basic idea is that similar users are likely have similar tastes. In contrast, item-based systems calculate the similarity between various items or products. However, you need a mathematical way to measure the similarity between movies. The most popular metric in item-based systems takes the dot product of the item vectors, exactly like in content-based filters. The similarity measure between two items a and b is defined as:

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

Where a.b is the usual dot product and the denominator contains the length of the vectors.

Issues in Recommendation Systems

The limitations of the two types of recommender systems are as follows:

1. The cold start problem: How will you provide recommendations when the website is just starting up and you have no previous data about products or users? How can the system know what a new user will like?
2. Explicit and implicit feedback: How can you collect information on what users like?
3. Recommending only a narrow range of items: How can you ensure that recommendations provided are not all similar to each other and have sufficient variation among them?

Let’s start with what is known as the cold start problem. In the ratings matrix you saw in the session, we assumed that most of the ratings exist. In reality, this is not the case because many users may not have watched most of the movies, or the platform is new and doesn’t have many users. For example, Amazon has millions of products and users, and most users will have rated only a few of them. The challenge is to generate useful recommendations using only a few data points.

This is especially a problem in collaborative filtering where recommendations are based on ratings given by other users. In this regard, content-based systems are better because recommendations are based on the content descriptions and not on other users.

One way to solve the cold start problem in content-based filtering is to explicitly ask the users what they like. For example, you may have noticed on some news sites that they ask you what kind of content you like to read. You can choose topics like sports, politics, technology, etc.

In collaborative filtering, explicit feedback is given in the form of ratings. When only a few users have rated the items, you can use alternative methods to infer ratings. One such way is called **implicit feedback**. In implicit feedback, you observe the behaviour of the users, such as their browsing history, items they have searched for in the past, etc., and use this data to predict ratings.

Another problem that arises in content-based systems is that new and unrelated items are rarely recommended. For example, in news recommender systems, if you have only read politics and cricket-related articles in the past, the system will keep recommending those articles in the future. It will probably not recommend you to read technology even if you like reading about it.

This is not a problem with collaborative systems. They will recommend other topics if similar users like them, and hence they recommend diverse items to you.