# BDA M5 - Big Data Analytics module 5 Notes

Big Data Analytics (University of Mumbai)

# MODULE V

# CHAPTER 5

# Real-Time Big Data Models

**University Prescribed Syllabus w.e.f Academic Year 2022-2023**

5.1 A Model for Recommendation Systems, Content-Based Recommendations, Collaborative Filtering

5.2 Case Study : Product Recommendation

5.3 Social Networks as Graphs, Clustering of Social-Network Graphs, Direct Discovery of Communities in a social graph


5.1 A Model for Recommendation Systems, Content-Based Recommendations, Collaborative Filtering ..... 5-2

**UQ.** How recommendation is done based on properties of product? Explain with suitable example. (MU - May 19, 10 Marks) ..... 5-2

**UQ.** What are Recommendation Systems? (MU - Dec. 17, 5 Marks) ..... 5-2

5.1.1 A Model for Recommendation Systems ..... 5-2

5.1.2 Applications for Recommendation System ..... 5-3

**UQ.** Clearly explain two applications for Recommendation system. (MU - Dec. 17, 5 Marks) ..... 5-3

5.1.3 Types of Recommender Systems ..... 5-5

5.1.4 Content based Recommendations ..... 5-5

5.1.5 Content-Based Recommendations Advantages and Disadvantages ..... 5-7

5.1.6 Collaborative Filtering ..... 5-7

5.2 Case study Product Recommendation ..... 5-10

5.3 Social Networks as Graphs, Clustering of Social-Network Graphs, Direct Discovery of Communities in a social graph ..... 5-14

5.3.1 Social Networks as Graphs ..... 5-14

5.3.2 Clustering of Social-Network Graphs ..... 5-17

5.3.3 Direct Discovery of Communities in a social graph ..... 5-21

**UQ.** Explain the Clique Percolation Method (CPM) used in direct discovery of communities in a social graph with example. (MU - Dec. 18, 10 Marks) ..... 5-21

**UQ.** What is a "Community" in a social network Graph? (MU - May 18, 5 Marks) ..... 5-21

**UQ.** What is a "Community" in a social Network Graph? Explain any one algorithm for finding communities in asocial graph. (MU - Dec. 17, 10 Marks) ..... 5-21

• **Chater Ends** ..... 5-22

## 5.1 A MODEL FOR RECOMMENDATION SYSTEMS, CONTENT-BASED RECOMMENDATIONS, COLLABORATIVE FILTERING

UQ. How recommendation is done based on properties of product? Explain with suitable example. **(MU - May 19, 10 Marks)**

UQ. What are Recommendation Systems? **(MU - Dec. 17, 5 Marks)**

### 5.1.1 A Model for Recommendation Systems

A recommendation system is also called as Recommender system. A recommender system is really an automated system to filter some entities can be any products, ads, people, movies or songs. And we see this from all over on a daily basis from Amzon, Netfix, Youtube, FilpCart etc.

A recommender system captures the pattern of people's behaviour and use it to predict what else they might want or like.

For example, we watch a movie and then later on we get a recommendation for a different movie based on the power of previous viewing history. It could also be a product that we bought and then we get a recommendation for another product based on the previous product viewing or purchase history. And recommender doesn't work only in what products we are being shown, but also in what order the products are being ranked.

For example, if you've recently purchased a book on Machine Learning in Python and you've enjoyed reading it, it's very likely that you'll also enjoy reading a book on Data Visualization. People also tend to have similar tastes to those of the people they're close to in their lives. Recommender systems try to capture these patterns and similar behaviours, to help predict what else you might like.

**Applications**

- What to buy?
  - E-commerce, books, movies, shoes, etc.
- Where to eat?
- Which job to apply to?
- Who you should be friends with?
  - LinkedIn, Facebook

Personalize your experience on the web

- News platforms, news personalization

Recommender systems function with two kinds of information:

- **Characteristic information**. This is information about items (keywords, categories, etc.) and users (preferences, profiles, etc.).
- **User-item interactions.** This is information such as ratings, number of purchases, likes, etc.

Why the Recommendation system?

- Benefits users in finding items of their interest.
- Help item providers in delivering their items to the right user.
- Identity products that are most relevant to users.
- Personalized content.
- Help websites to improve user engagement.

**Advantages of Recommender system**

1. Broder exposure
2. Possibility of continual usage or purchase of products
3. Provides better experience

### 5.1.2 Applications for Recommendation System

**UQ.** Clearly explain two applications for Recommendation system. **(MU - Dec. 17, 5 Marks)**

**(1) Entertainment** : recommendations for movies, music, and IPTV.

**(2) Content** : personalized newspapers, recommendation for documents, recommendations of Web pages, e-learning applications, and e-mail filters.

**(3) E-commerce** : recommendations for consumers of products to buy such as books, cameras, PCs etc.

**(4) Services** : recommendations of travel services, recommendation of experts for consultation, recommendation of houses to rent, or matchmaking services

## 5.1.3 Types of Recommender Systems

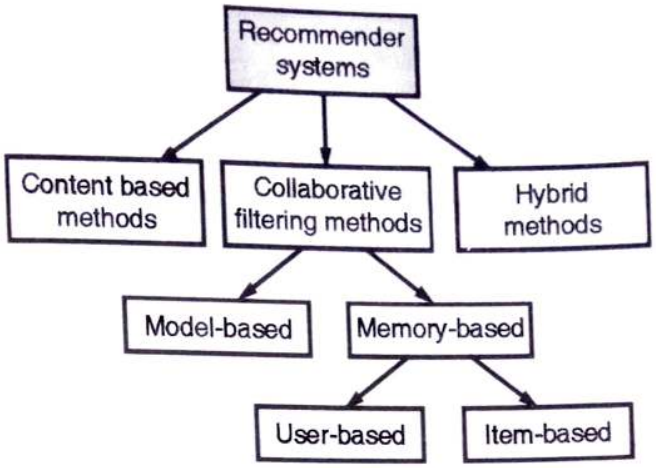


Fig. 5.1.1

### Collaborative filtering

- Collaborative filtering focuses on collecting and analyzing data on user behavior, activities, and preferences, to predict what a person will like, based on their similarity to other users.
- To plot and calculate these similarities, collaborative filtering uses a matrix style formula. An advantage of collaborative filtering is that it doesn't need to analyze or understand the content (products, films, books). It simply picks items to recommend based on what they know about the user.

### Content-based filtering

- Content-based filtering works on the principle that if you like a particular item, you will also like this other item.
- To make recommendations, algorithms use a profile of the customer's preferences and a description of an item (genre, product type, color, word length) to work out the similarity of items using cosine and Euclidean distances.
- The downside of content-based filtering is that the system is limited to recommending products or content similar to what the person is already buying or using. It can't go beyond this to recommend other types of products or content. For example, it couldn't recommend products beyond homeware if the customer had only brought homeware.

### brid model

A hybrid recommendation engine looks at both the meta (collaborative) data and the transactional (content-based) data. Because of this, it outperforms both.

- In a hybrid recommendation engine, natural language processing tags can be generated for each product or item (movie, song), and vector equations used to calculate the similarity of products.
- A collaborative filtering matrix can then be used to recommend items to users depending on their behaviors, activities, and preferences. Netflix is the perfect example of a hybrid recommendation engine. It takes into account both the interests of the user (collaborative) and the descriptions or features of the movie or show (content-based).

### 5.1.4 Content based Recommendations

- The gist of this approach is that we match users to the content or items they have liked or bought. Here the attributes of the users and the products are important.
- For example, for movie recommendations, we use features such as director, actors, movie length, genre, etc. to find similarity between movies. Furthermore, we can extract features like sentiment score and tf-idf scores from movie descriptions and reviews. (The tf-idf score of a word reflects how important a word is to a document in a collection of documents). **The aim of content-based recommendation is to create a 'profile' for each user and each item.**
- Consider an example of recommending news articles to users. Let's say we have 100 articles and a vocabulary of size N. We first compute the tf-idf score for each of the words for every article. Then we construct 2 vectors:
  1. **Item vector :** This is a vector of length N. It contains 1 for words that have a high tf-idf score in that article, otherwise 0.
  2. **User vector :** Again a 1xN vector. For every word, we store the probability of the word occurring (i.e. having a high tf-idf score) in articles that the user has consumed. Note here, that the user vector is based on the attributes of the item (tf-idf score of words in this case).
- Once we have these profiles, we compute similarities between the users and the items. The items that are recommended are the ones that 1) the user has the highest similarity with or 2) has the highest similarity with the other items the user has read. There are multiple ways of doing this. Let's look at 2 common methods:
  1. **Cosine similarity**
     - To compute similarity between the user and item, we simply take the cosine similarity between the user vector and the item vector. This gives us user-item similarity.

- To recommend items that are most similar to the items the user has bought, we compute cosine similarity between the articles the user has read and other articles. The ones that are most similar are recommended. Thus this is item-item similarity.

$$\text{cosine}(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}$$

- Cosine similarity is best suited when you have high dimensional features, especially in information retrieval and text mining.

**2. Jaccard similarity:**

- Also known as intersection over union, the formula is as follows:

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

- This is used for item-item similarity. We compare item vectors with each other and return the items that are most similar.
- Jaccard similarity is useful only when the vectors contain binary values. If they have rankings or ratings that can take on multiple values, Jaccard similarity is not applicable.
- This method is useful when we have a whole lot of 'external' features, like weather conditions, market factors, etc. which are not a property of the user or the product and can be highly variable. For example, the previous day's opening and closing price play an important role in determining the profitability of investing in a particular stock.
- This comes under the class of supervised problems where the label is whether the user liked/clicked on a product or not (0/1) or the rating the user gave that product or the number of units the user bought.
- For example, if a user likes movies such as 'Mission Impossible' then we can recommend him the movies of 'Tom Cruise' or movies with the genre 'Action'.

- In this filtering, two types of data are used. First, the likes of the user, the user's interest, user's personal information such as age or, sometimes the user's history too. This data is represented by the user vector. Second, information related to the product's known as an item vector. The item vector contains the features of all items based on which similarity between them can be calculated.

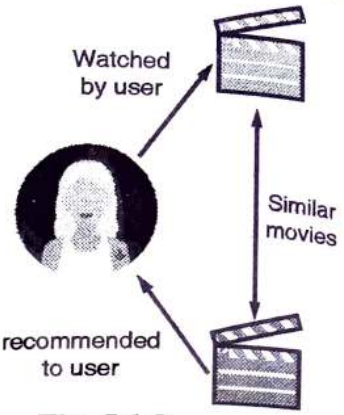


**Fig. 5.1.2**

- The recommendations are calculated using cosine similarity. If 'A' is the user vector and 'B' is an item vector then cosine similarity Values calculated in the cosine similarity matrix are sorted in descending order and the items at the top for that user are recommended.

### 5.1.5 Content-Based Recommendations Advantages and Disadvantages

| Sr. No. | Advantages | Disadvantages |
|---|---|---|
| 1 | Does not depend on data of other users. | When we have a new user, without much information about his transactions, we cannot make accurate recommendations |
| 2 | There is no cold start problem for new items. This is because, using the item features we can easily find items it is similar to. | Clear-cut groups of similar products may result in not recommending different products. E may end up recommending a small subset over and over again |
| 3 | Recommendation results are interpretable. | If there is limited information about the content, it is difficult to clearly discriminate between items and group recommendations |

### 5.1.6 Collaborative Filtering

- The recommendations are done based on the user's behavior. History of the user plays an important role. For example, if the user 'A' likes 'Coldplay', 'The Linkin Park' and 'Britney Spears' while the user 'B' likes 'Coldplay',

- 'The Linkin Park' and 'Taylor Swift' then they have similar interests. So, there is a huge probability that the user 'A' would like 'Taylor Swift' and the user 'B' would like 'Britney Spears'. This is the way collaborative filtering is done.

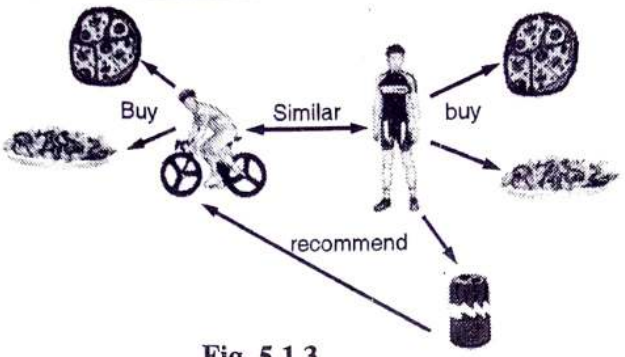


Fig. 5.1.3

- The underlying assumption of the collaborative filtering approach is that if A and B buy similar products, A is more likely to buy a product that B has bought than a product which a random person has bought.
- Unlike content based, there are no features corresponding to users or items here. All we have is the Utility Matrix. This is what it looks like:

| | HP1 | HP2 | HP3 | TW | SW1 | SW2 | SW3 |
|---|---|---|---|---|---|---|---|
| A | 4 | | | 5 | 1 | | |
| B | 5 | 5 | 4 | | | | |
| C | | | | 2 | 4 | 5 | |
| D | | 3 | | | | | 3 |

- A, B, C, D are the users, and the columns represent movies. The values represent ratings (1–5) a user has given a movie. In other cases, these values could be 0/1 depending on whether the user watched the movie or not. There are a 2 broad categories that collaborative filtering can be split into:

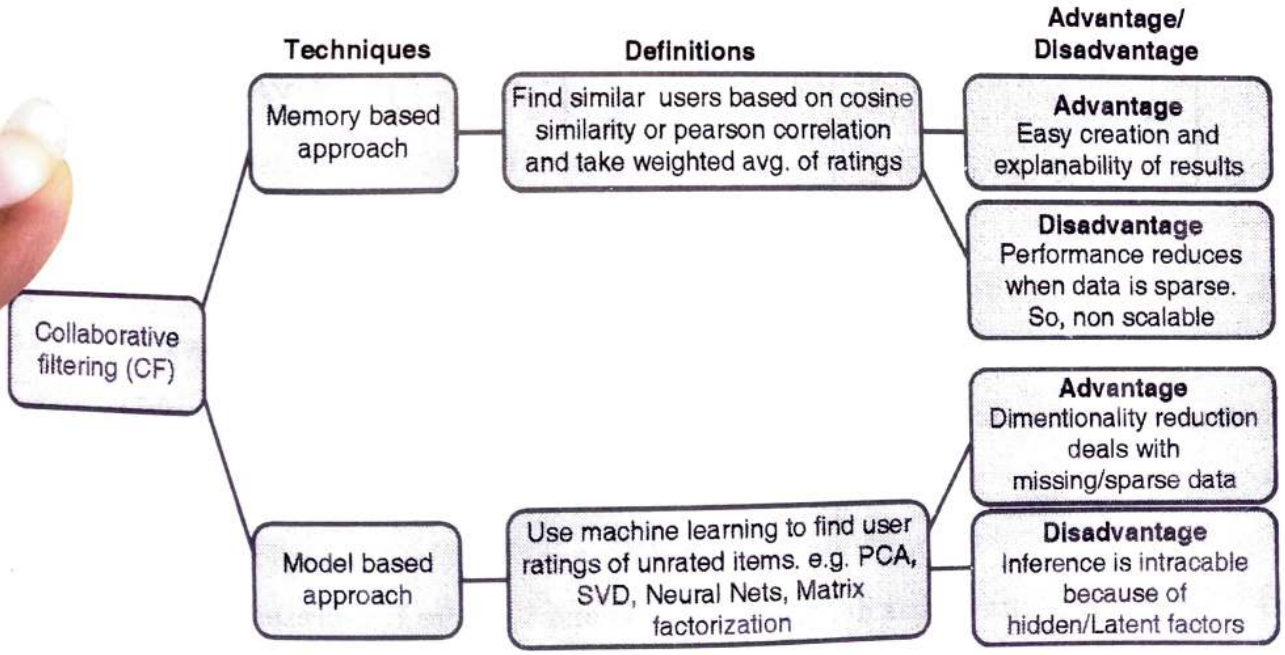


Fig. 5.1.4

**Memory based approach**

- For the memory based approach, the utility matrix is memorized and recommendations are made by querying the given user with the rest of the utility matrix. Let's consider an example of the same: If we have $m$ movies and $u$ users, we want to find out how much user $i$ likes movie $k$.

$$\bar{y}_i = \frac{1}{|I_i|} \sum_{j \in I_i} y_{ij}$$

This is the mean rating that user $i$ has given all the movies she/he has rated. Using this, we estimate his rating of movie k as follows:

Similarity between users a and i

$$\hat{y}_{ik} = \bar{y}_i + \frac{1}{\Sigma a \in U_k = |w_{ia}|} \sum_{a \in U_k} w_{ia}(y_{ak} - \bar{y}_a)$$

a's rating of k – a's average ratings

All users that have rated k

Similarity between users $a$ and $i$ can be computed using any methods like cosine similarity / Jaccard similarity / Pearson's correlation coefficient, etc. These results are very easy to create and interpret, but once the data becomes too sparse, performance becomes poor.

**Model based approach**

- One of the more prevalent implementations of model based approach is Matrix Factorization. In this, we create representations of the users and items from the utility matrix. This is what it looks like :

$$\begin{bmatrix} 5 & 1 & 4 & 5 & 1 \\ & 5 & 2 & 1 & 4 \\ 1 & 4 & 1 & 1 & 2 \\ 4 & 1 & 5 & 5 & 4 \\ 5 & 3 & 3 & & 4 \\ 1 & 5 & 1 & 1 & 1 \\ 5 & 1 & 5 & 5 & 4 \end{bmatrix} \approx \begin{bmatrix} \mu_{11} & \mu_{12} & \ldots & \mu_1 K \\ \mu_{21} & \mu_{22} & \ldots & \mu_2 K \\ \mu_{31} & \mu_{32} & \ldots & \mu_3 K \\ \mu_{41} & \mu_{42} & \ldots & \mu_4 K \\ \mu_{51} & \mu_{52} & \ldots & \mu_5 K \\ \mu_{61} & \mu_{62} & \ldots & \mu_6 K \\ \mu_{71} & \mu_{72} & \ldots & \mu_7 K \end{bmatrix} \times \begin{bmatrix} \upsilon_{11} & \upsilon_{21} & \upsilon_{31} & \upsilon_{41} & \upsilon_{61} \\ \upsilon_{12} & \upsilon_{22} & \upsilon_{32} & \upsilon_{42} & \upsilon_{62} \\ \vdots & & & & \\ \upsilon_1 k & \upsilon_2 k & \upsilon_3 k & \upsilon_4 k & \upsilon_6 k \end{bmatrix}$$

$$\approx \begin{bmatrix} 0.2 & 3.4 \\ 3.6 & 1.0 \\ 2.6 & 0.6 \\ 0.9 & 3.7 \\ 2.0 & 3.4 \\ 2.9 & 0.5 \\ 0.8 & 3.9 \end{bmatrix} \times \begin{bmatrix} 0.0 & 1.5 & 0.1 & 0.0 & 0.7 \\ 1.3 & 0.0 & 1.2 & 1.4 & 0.7 \end{bmatrix}$$

- Thus, our utility matrix decomposes into U and V where U represents the users and V represents the movies in a low dimensional space.

- This can be achieved by using matrix decomposition techniques like SVD or PCA or by learning the 2 embedding matrices using neural networks with the help of some optimizer like Adam, SGD etc.

$$\hat{y}_{ij} = u_i \cdot v_j$$

- For a user $i$ and every movie $j$ we just need to compute rating $y$ to and recommend the movies with the highest predicted rating. This approach is most useful when we have a ton of data and it has high sparsity.
- Matrix factorization helps by reducing the dimensionality, hence making computation faster. One disadvantage of this method is that we tend to lose interpretability as we do not know what exactly elements of the user/item vectors mean.

## 5.2 CASE STUDY PRODUCT RECOMMENDATION

- Systems that automatically recommend products and content may play a more significant role than you realize. With the number of options offered to the consumer continuing to explode, product recommendation systems embody a key way in which machine learning serves as an antidote to "information overload" and "analysis paralysis." Movies, music, books... there are just too many options!
- Netflix recommends movies by predicting which ones you'd rate highly. Spotify and Pandora recommend music. Amazon was a pioneer recommending books
- Recommending users the products based on various parameters inferred from the available dataset. The selection of parameters is a challenging task, since, based on the attribute, it might result in under fitted model or over fitted model.
- Models which are either under fitted or over fitted, gives poor result in accuracy on testing dataset, and thus are not preferred. This work implements the product recommendation based on collaborative filtering and the recommendation is highly focused on the influence of other users as well as the products itself.

### 1. Recommendation Based on Influence

- This type of recommendation will be based on how the particular node is influenced by the other nodes.
- In the case of product recommendation, this method will allow the user to see a list of recommended items that he or she can buy along with the item that he or she has already chosen.

- Suppose item X is bought together with item Y. If the item X is bought with several other items as well, then we say that X and Y don't influence each other whereas, if the item X is mostly bought along with item Y then we say that item X and Y influence each other. This is called _influence scoring'.

**2. Recommendation by Commonly Bought Products**

- This method is like the transitive property. If X is bought together with Y and most of the people buying Y have also bought Z along with it, then item Z will also be recommended when any user is buying item X. Also, if the item Z is bought commonly then its chances of being recommended increases.
- This system performs collaborative filtering by considering the recommendations based on both, influence and products that are widely bought together.

**3. Movie Recommendation**

- It uses both, collaborative filtering technique and content-based filtering technique for implementation of movie recommendation system. Collaborative-filtering techniques deal with data of the particular user along with the data of other users. On the contrary, content-based filtering techniques deal with the data of the user alone.
- The collaborative filtering technique is very simple where the requirement is just to find the neighbors that share similar interests with that of the user and then recommend the user the movies that the neighbors, having similar taste, like. The basic idea is to recommend similar items (movies in this case) to the similar end-users of the recommendation system based on the history of ratings by the user for the items .

**This collaborative filtering technique is divided into the following steps :**

(1) **Collaborative filtering learning algorithm :** The collaborative filtering algorithm is dependent on a few parameters based on the movie and the rating by users. With these rating on some movies, by some user, the system will begin by finding out those parameters that affect this prediction and find out the best fit movie. For the ease of calculating, a cost function is set up to unroll these parameters into single vector params;

(2) **Calculating collaborative filtering :** The collaborative filtering calculations are carried out using the cost functions and gradients of the values that are obtained .

(3) **Regularized cost function :** After the calculation of the collaborative filtering parameters, the calculated measures of cost function and the gradients are regularized to further work with it.

The content-based filtering technique is divided into the following steps – identifying the actors, genres, directors and plots of the movies that the user has watched; apply content-based filtering algorithm: to suggest movies based on calculations.

(4) **Amazon product recommendation :** In order to provide customers with accurate product recommendations, **Amazon's algorithm must analyze huge amounts of data**. In this way, it better understands the behavior of all users and the interests of each viewer.

**To do this, the recommendation engine collects two types of information**

- general data about products and users;
- data on relations and dependencies between them.
- Getting to know the existing relationships in the online store will provide the recommendation engine with an insight into the real mechanisms governing customers' purchasing decisions.
- Amazon's recommendation algorithm analyzes 3 main types of dependencies and relationships for its operation:

**User-product**

- This type of relationship occurs when some users with certain characteristics prefer products of a given type and buy them more often.
- A good example would be gamers buying expensive computer components or fans of various series and movies buying related gadgets and t-shirts.

**Product-product**

- Product-product relationships occur when the products offered in the store are similar in terms of both appearance and specification.
- Some examples include books, movies, series or music of the same genre, or dishes from the same cuisine.
- User-user it occurs when individual customers with certain characteristics have similar tastes or preferences for certain products.

- Examples of such relationships include teenagers massively buying merchandise from their favorite You Tuber or cooking fans who prefer a specific line of kitchen products.
- In addition to collecting information about relationships and connections Amazon's recommendation algorithm also uses different types of product and user data:

**User behaviour data**

- This type of data is useful information about the preferences of individual customers, their interaction with the products in question.
- Amazon uses cookies to collect data about your browsing history, likes, or session length.

**User Demographics data**

- User demographics data is linked to personal information about individual customers, such as age, education, income and location.
- In order to collect such data, the user's consent is required.

**Product Attribute Data**

- Product Attribute data is information related to the product itself, such as the specification of the computer, blouse size information, collection description.

**A method of filtering data by Amazon's recommendation algorithm**

Many of these systems can be classified as content-based filtering or collaborative filtering.

**Content-based filtering on Amazonie**

- Content-based filtering is one of the simplest systems and is constantly used by modern recommendation systems.
- The main idea behind content-based filtering is that if a customer likes a certain product, chances are they will like another product with a similar specification.

**Collaborative filtering**

- Unlike content-based filtering, group filtering uses the experience of other users to generate recommendations.
- Interestingly, Amazon pioneered this approach and published the article Recommendations: Item-to-Item Collaborative Filtering in 2003, which later won an award from the Institute of Electrical and Electronics Engineers (IEEE). ]

- The undoubted advantage of this method is that it allows the recommendation engine to generate proposals for relatively complex products, such as movies or music, without the need to have specialist knowledge about them.
- Compared to content-based filtering, team filtering produces better results in several key areas:
  - **diversity** : group filtering generates a more diverse list of recommended products, offering customers a wider choice,
  - **randomness :** recommendations generated using the collaborative filtering method are much more likely to positively surprise the client and show them a product of interest, which they might not otherwise discover,
  - **randomness :** the collaborative filtering method is able to more effectively present to customers novelties in the store's offer that users would most likely be interested in.
  - It is worth noting, however, that both collaborative and content-based methods offer the best and most effective results if they work together in one comprehensive recommendation engine.
  - This is how the Amazon algorithm works - by combining the best elements of both strategies, it offers its clients the highest quality recommendations.
  - Amazon's recommendation algorithm is probably the most complex and effective of the ecommerce market. The company has been steadily developing its recommendation system for almost 20 years and is now responsible for a large proportion of sales.
  - Amazon 's personalized recommendation engine improves customer experience and presents products in real time based on their browsing history

## 5.3 SOCIAL NETWORKS AS GRAPHS, CLUSTERING OF SOCIAL-NETWORK GRAPHS, DIRECT DISCOVERY OF COMMUNITIES IN A SOCIAL GRAPH

### 5.3.1 Social Networks as Graphs

- There is collection of entities that participate in network. Typically, people, but could be something else too. There is at least one relationship between entities of the network. For example: friends
- Sometimes Boolean : two people are either friends or they are not. They may have a Discrete degree. Eg. friends, family, acquaintances, or none. It could be real number : the fraction of the average day that two people. An example spends talking to each other.

- There is an assumption of non randomness or locality. This condition is the hardest to formalize, but the intuition is that relationships tend to cluster. That is, if entity A is related to both B and C, then there is a higher probability than average that B and C are related.
- Social networks are naturally modeled as graphs, which we sometimes refer to as a social graph. The entities are the nodes, and an edge connects two nodes if the nodes are related by the relationship that characterizes the network.
- If there is a degree associated with the relationship, this degree is represented by labeling the edges. Often, social graphs are undirected, as for the Face book friend's graph. But they can be directed graphs, as for example the graphs of followers on Twitter or Google+.

---

**Ex. 5.3.1 :** Fig. Ex. 5.3.1 is an example of a tiny social network. The entities are the nodes A through G. The relationship, which we might think of as "friends," is represented by the edges. For instance, B is friends with A, C, and D.Is this graph really typical of a social network, in the sense that it exhibits locality of relationships?

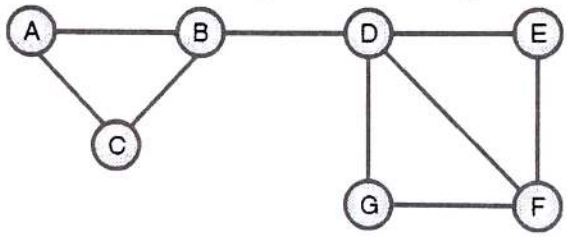


**Fig. Ex. 5.3.1 : Example of a small social network**

**Soln. :**

- Check for the non-randomness criterion. In a random graph (V, E) of 7 nodes and 9 edges, if XY is an edge, YZ is an edge, what is the probability that XZ is an edge?
- For a large random graph, it would be close to $\frac{|E|}{|V|C2} = \frac{9}{21} \sim 0.43$
- Small graph: XY and YZ are already edges, so compute within the rest. So the probability is $\frac{|E|-2}{(|V|C2-2)} = \frac{7}{19} = 0.37$.
- Now let's compute what is the probability for this graph in particular. For each X, check possible YZ and check if YZ is an edge or not Example : if $X$ = A, YZ = {BC}, it is an edge.

| X = | YZ = | Yes / Total |
|---|---|---|
| A | BC | $\frac{1}{1}$ |
| B | AC, AD, CD | $\frac{1}{3}$ |
| C | AB | $\frac{1}{1}$ |
| D | BE, BG, BF, EF, EG, FG | $\frac{2}{6}$ |

| X = | YZ = | Yes/ Total |
|---|---|---|
| E | DF | $\frac{1}{1}$ |
| F | DE, DG, EG | $\frac{2}{3}$ |
| G | DF | $\frac{1}{1}$ |
| Total | | $\frac{9}{16} \sim 0.56$ |

## Varieties of Social Networks

### Telephone networks

- Nodes are phone numbers.
- AB is an edge if A and B talked over phone within the last one week, or month, or ever.
- Edges could be weighted by the number of times phone calls were made, or total time of conversation.

### Email networks: nodes are email addresses

- AB is an edge if A and B sent mails to each other within the last one week, or month, or ever.
  - One directional edges would allow spammers to have edges.
- Edges could be weighted.
- Other networks: collaboration network – authors of papers, jointly written papers or not.
- Also networks exhibiting locality property

### Collaboration networks

- Nodes represent individuals who have published research papers. There is an edge between two individuals who published one or more papers jointly.
- Option-ally, we can label edges by the number of joint publications. The communities in this network are authors working on a particular topic.

An alternative view of the same data is as a graph in which the nodes are papers. Two papers are connected by an edge if they have at least one author in common. Now, we form communities that are collections of papers on the same topic.

### Other Examples of Social Graphs

- Many other phenomena give rise to graphs that look something like social graphs, especially exhibiting locality.
- Examples include: information networks(documents, web graphs, patents), infrastructure networks (roads, planes, water pipes, power grids), biological networks (genes, proteins, food-webs of animal seating each other), as well as other types, like product co-purchasing networks(e.g., Group on).

## 5.3.2 Clustering of Social-Network Graphs

**GQ.** Explain clustering of Social-Network Graphs with example. **(10 Marks)**

An important aspect of social networks is that they contain communities of entities that are connected by many edges. These typically correspond to groups of friends at school or groups of researchers interested in the same topic, for example.

### Distance Measures for Social-Network Graphs

- When the edges of the graph have labels, these labels might be usable as a distance measure, depending on what they represented. But when the edges are unlabeled, as in a "friends" graph, there is not much we can do to define a suitable distance.
- Our first instinct is to assume that nodes are close if they have an edge between them and distant if not. Thus, we could say that the distance d (x, y) is 0 if there is an edge (x, y) and 1 if there is no such edge. We could use any other two values, such as 1 and $\infty$, as long as the distance is closer when there is an edge.
- Neither of these two-valued "distance measures" – 0 and 1 or 1 and $\infty$– is a true distance measure.

### Applying Standard Clustering Methods

- In particular, suppose we use as the inter cluster distance the minimum distance between nodes of the two clusters. Hierarchical clustering of a social-network graph starts by combining some two nodes that are connected by an edge.
- Successively, edges that are not between two nodes of the same cluster would be chosen randomly to combine the clusters to which their two nodes belong. The choices would be random, because all distances represented by an edge are the same.

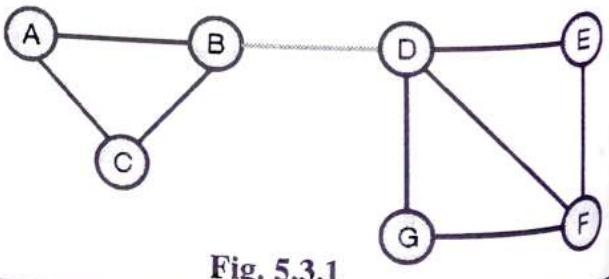


Fig. 5.3.1

### Point-assignment approach

- in k-means randomly picked nodes might be in the same cluster
- randomly chosen and another as far away as possible doesn't do much better (.i.e. E and G)
- even choosing B and F – problem with assignment of D

### Betweenness

This method based on finding the edges that are **least likely to be inside a community.** Betweenness of and edge (a, b) is the number of pairs of nodes x and y such that the edge (a, b) lies on the shortest path between x and y. High value suggests that (a, b) runs between two different communities – a and b do not belong to the same community

### The Girvan-Newman Algorithm

To exploit betweenness, we need to calculate the number of shortest paths going through each edge

Girvan-Newman Algorithm visits each node X once and computes the number of shortest paths from X to other nodes through each of the edges.

- **Step I :**

  **BFS :** Start at a node X, perform a BFS with X as root

  **Observe :** level of node Y = length of shortest path from X to Y

  Edges between level are called "DAG" edges

  Each DAG edge is part of at least one shortest path from X

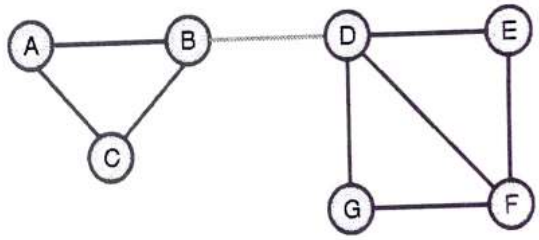


**Fig. 5.3.2**

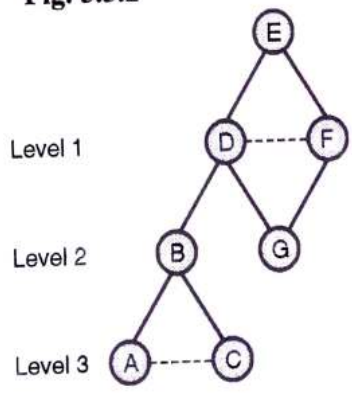


**Fig. 5.3.3 : Newman Algorithm**

### Step II :

Label each node by the number of shortest paths that reach it from the root the sum of the labels of its parents

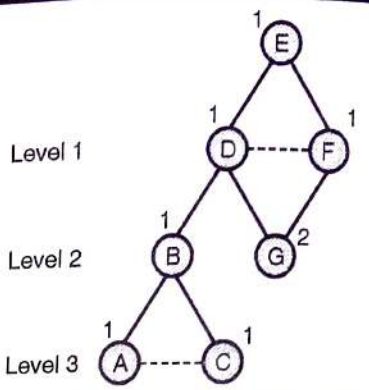


Fig. 5.3.4 : Newman Algorithm

### Step III :

- do the calculation starting from the bottom according to rules:
- leaf gets a credit of 1 node that is not a leaf gets a credit equal to 1 plus the sum of the credits of the edges to the level below edge entering node from the level above is given a proportional share of that node

Credit of DAG edges : Let $Y_i$ (I = 1, ....k) be parents of Z, $p_i$ = label ($Y_i$)

$$\text{credit (Y, Z)} = \frac{\text{credit (Z)} \times p_i}{(p_i + \ldots p_i)}$$

- **Intuition :** a DAG edge Y, Z gets the share of credit of Z proportional to the ≠ of shortest paths from X to Z going through Y,Z
- **Finally :** Repeat Steps 1, 2 and 2 with each node as root. For each edge, betweenness = sum credits obtained in all iterations / 2

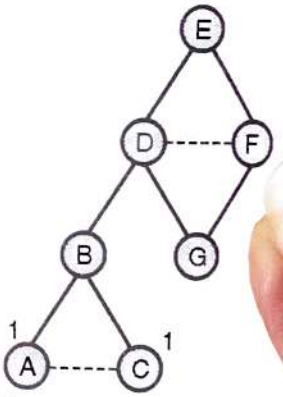


Fig. 5.3.5 : Newman Algorithm

- First, A and C, being leaves, get credit 1. Each of these nodes have only one parent, so their credit is given to the edges (B, A) and (B, C), respectively.
- At level 2, G is a leaf, so it gets credit 1. B is not a leaf, so it gets credit equal to 1 plus the credits on the DAG edges entering it from below. Since both these edges have credit 1, the credit of B is 3. Intuitively 3 represents the fact that all shortest paths from E to A, B, and C go through B. Fig. 5.3.6 shows the credits assigned so far.

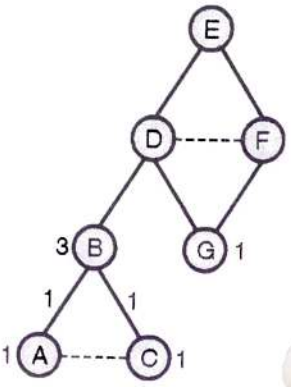


Fig. 5.3.6 : Newman Algorithm – levels 3 and 2

- Now, let us proceed to level 1. B has only one parent, D, so the edge (D,B) gets the entire credit of B, which is 3. However, G has two parents, D and F. We therefore need to divide the credit of 1 that G has between the edges (D, G) and (F, G). In what proportion do we divide? If you examine the labels of Fig. 5.3.4. you see that both D and F have label 1, representing the fact that there is one shortest path from E to each of these nodes.

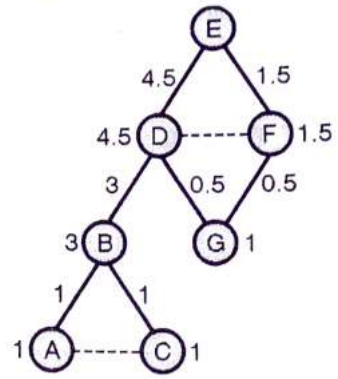


**Fig. 5.3.7 : Newman Algorithm**

- Thus, we give half the credit of G to each of these edges; i.e., their credit is each $1/(1+1) = 0.5$.
- Had the labels of D and F in Fig. 5.3.7 been 5 and 3, meaning there were five shortest paths to D and only three to F, then the credit of edge (D, G) would have been 5/8 and the credit of edge (F, G) would have been 3/8.
- Now, we can assign credits to the nodes at level 1. D gets 1 plus the credits of the edges entering it from below, which are 3 and 0.5. That is, the credit of D is 4.5. The credit of F is 1 plus the credit of the edge (F, G), or 1.5. Finally, the edges (E, D) and (E, F) receive the credit of D and F, respectively, since each of these nodes has only one parent. These credits are all shown in Fig. 5.3.7.
- The credit on each of the edges in Fig. 5.3.7 is the contribution to the betweenness of that edge due to shortest paths from E. For example, this contribution for the edge (E, D) is 4.5.
- To complete the betweenness calculation, we have to repeat this calculation for every node as the root and sum the contributions. Finally, we must divide by 2 to get the true betweenness, since every shortest path will be discovered twice, once for each of its endpoints.

**Using Betweenness to Find Communities**

- to complete betweenness calculation:
  - repeat these steps for every node as the root and sum the contributions
  - divide by 2, because every shortest path will be discovered twice, once for each endpoints
- betweenness may behave like a distance measure, but is not exactly a distance measure
- ordering edges by betweenness and removing / adding nodes from graph

**Ex. 5.3.1:** We see it with the betweenness for each edge in Fig. Ex.5.3.1. The calculation of the betweenness will be left to the reader. The only tricky part of the count is to observe that between E and G there are two shortest paths, one going through D and the other through F. Thus, each of the edges (D, E), (E, F), (D, G), and (G, F) are credited with half a shortest path.

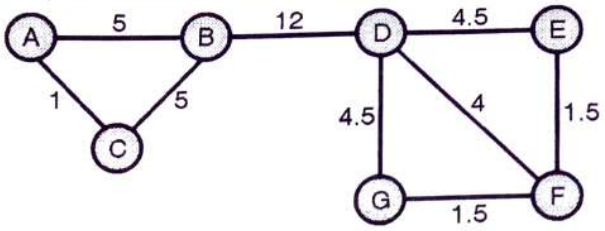


**Fig. 5.3.1 : Betweenness scores for the graph**

**Soln. :** Clearly, edge (B,D) has the highest betweenness, so it is removed first. That leaves us with exactly the communities we observed make the most sense, namely: {A, B, C} and {D, E, F, G}. However, we can continue to remove edges.

Next to leave are (A, B) and (B, C) with a score of 5, followed by (D, E) and (D, G) with a score of 4.5. Then, (D, F), whose score is 4, would leave the graph. We see in Fig. Ex. 5.3.1 (a) the graph that remains.

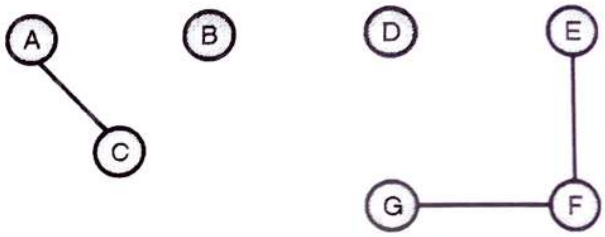


**Fig. Ex. 5.3.1 (a) : All the edges with betweenness 4 or more have been removed**

The "communities" of Fig. Ex. 5.3.1 (a) look strange. One implication is that A and C are more closely knit to each other than to B. That is, in some sense B is a "traitor" to the community {A, B, C} because he has a friend D outside that community. Likewise, D can be seen as a "traitor" to the group {D, E, F, G}, which is why in Fig. Fig. Ex. 5.3.1 (a) , only E, F, and G remain connected.

## 5.3.3 Direct Discovery of Communities in a social graph

**UQ.** Explain the Clique Percolation Method (CPM) used in direct discovery of communities in a social graph with example. **(MU - Dec. 18, 10 Marks)**

**UQ.** What is a "Community" in a social network Graph? **(MU - May 18, 5 Marks)**

**UQ.** What is a "Community" in a social Network Graph? Explain any one algorithm for finding communities in asocial graph. **(MU - Dec. 17, 10 Marks)**

Discovering communities directly by looking for subsets of the nodes that have a relatively large number of edges among them. Interestingly, the technique for doing this search on a large graph involves finding large frequent item sets

### Finding Cliques

We want to discover communities directly by looking for subsets of the nodes that have a relatively large number of edges among them

- first thought-finding a large clique
- NP-complete problem, even approximating the maximal clique is hard

It is possible to have a set of nodes with almost all edges between them, yet with relatively small cliques

### Complete Bipartite Graphs

- Graph that consists of 's' nodes on one side and 't' nodes on the other side with all 'st' possible edges between them
- It is possible to guarantee that a bipartite graph with many edges has a large complete bipartite sub graph (unlike cliques)
- It might be regarded as the nucleus of community

### Finding Complete Bipartite Sub graphs

we are given a large bipartite graph G , and we want to find instances of Ks,t within it. It is similar to finding frequent item sets. Items" on the left side (Ks,t – t nodes there) assumption that t ≤ s The baskets" on the right side. member of the basket are the nodes from left side connected to that node. support threshold s frequent item set of size t and s of the baskets, in which all those items appear, form an instance of Ks,t

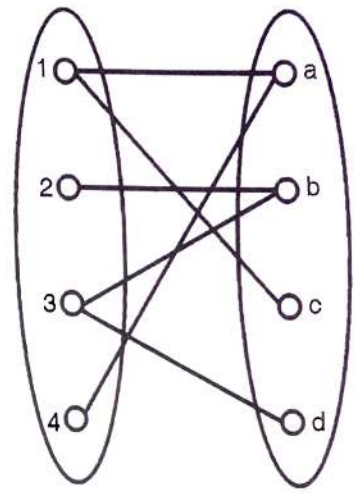


**Fig. 5.3.7 : The bipartite graph**

### Why Complete Bipartite Graphs Must Exist

If there is a community with n nodes and average degree d, then this community is guaranteed to have a complete bipartite sub graph Ks,t when:

$$n\frac{\binom{d}{t}}{\binom{n}{t}} \geq s$$

which approximately is :

$$n\left(\frac{d}{n}\right) \geq s$$

*Chapter Ends...*

❑❑❑