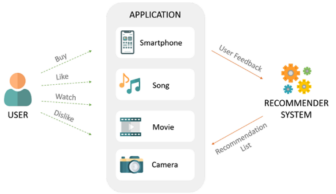
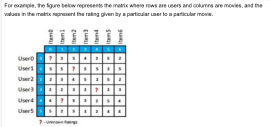
Basics of Recommendation systems

Description	What	Limitations/Assumptions	Example/Formula	Reference/Comments
Recommendation System/ Recommendation Engine/ Rec	is an automated system that provides preferable suggestions/recommendations of products to users	80% of viewer activity is driven by personalized recommendations	When u watch a movie of some particular genre on Netflix Watch some video on Youtube they start to give you suggestions/recommendations for similar types of content you may like Amazon recommendation system for online shoppers after you view some product it starts suggesting similar products for you to buy After you listen to songs on the app, the recommendation systems of music apps like Spotify or Apple Music start to suggest other similar songs you may like to listen to	
Usage of recommendation systems	essentially use the customer data (what you watch, what you buy, what music you listen to , etc) to find patterns in that data and based on these patterns, **provide personalized suggestions to improve customer experience**	Like every machine learning algorith,a recommendation systems makes predictions based on user's historical behavior. It tries to predict user preference for a set of items based on past experience		
Data Representation for Recommendation Systems	is represented as a matrix of users and items.			
User Item Interaction Matrix	The ? in the above matrix represents missing data i.e there is no interaction between that user-item pair, or in other words, that user has not rated that movie			
Sparse Matrix	**In reality, most of the data pounts in such a matrix would be** ? There would be a large number of users and movies, and the actual number of user-item interactions which took place would usually pale in comparison to the total number of possible user-item interactions because **only a few users rate a few movies.**			
Averaging	One simple way is to assume that every user is the same and **we predict values simply by the average of the column**, or in our case, the prediction is the average rating given to some movie by all the users who watch that movie			
Content Based Filtering	Content based approach uses additional information about users and/or items. Idea of content-based methods is to try to build a model, based on available "features", that explain the observed user-item interactions. If we manage to get such a model, then making new predictions for a user is pretty easy, we just need to look at the profile (age,sec, etc) of this user and, based on this information, determine relevant movies to suggest		IN movie recommender system, this additional info can be age of the user, gender , job or any other personal information of the user Model the fact that young women tend to better rate some movies, young men tend to rate some movies better, and so on.	

Recommendation systems

Description	What	Limitations/Assumptions	Example/Formula	Reference/Comments
Outline	Background Problem Statement Simple Solutions			
Background	Recommendation Systems: Why & What Example Datasets			
Problem Statement	Recommendation Systems: a prediction problem Model: from caricature to extremely complex			
Simple Solutions	1. Averaging 2. Content-based			
Yelp Data	Business: Attributes (locations, category), hours Users: attributes, friends Reviews: rating, description, time Check-ins: time Tip			https://www.kaggle.com/code/jagangupta/what-s-in-a-review-yelp-ratings-eda/notebook https://www.kaggle.com/code/jagangupta/what-s-in-a-review-yelp-ratings-eda/notebook
MovieLens Data	Movies: attributes including title, release date, genre, actors, director Users: demographics including age, gender, occupation, zip code Reviews: ratings, timestamp			https://grouplens.org/datasets/movielens/100k/
Problem Statement - What does yelp want to do	**Provide list of businesses** that satisfy your search criteria **ordered** as per user's interest or preference list at that moment advertisement revenue considerations so that ultimately **matching** happens within few clicks that provides instance gratification to both user and Yelp and continues bringing user back to Yelp			https://www.ijircst.org/DOC/3-major-challenges-of-recommender-system-and-related-solutions.pdf
Recommendation Problem Statements	Linkedin: Connect people professionally Facebook: filter friend's feed Poshmark,Etsy: organize content of display Amazon, Retail: display products and sellers Tinder, Match: find suitable partner Netflix, Youtube, Spotify: entertainment of interest			
So that ultimately **matching** within few clicks, providing instant gratification to **user, provider and platform** and **users** and **providers** continue engaging with the **platform**				
Prediction problem what is the **likelihood** of **matching** **user** **provider** at a given **time** in a given **context**	**Accurate solution** to this **prediction** problem provides essential **ingredient** for **connecting users**, **providers on the platform** while **respecting interests** of **users, providers and platform**			
Prediction problem **N** users **M** providers or items (or other users) Given user **i** and item **j** **Likelihood** if **i** matching with **j** L_{ij}				

Challenges that we will not discuss	user provided data not generated at random can be strategic can be driven by innate preference provider/ item can be systematically manipulated convenient location strategically modified content Side effects of recommendation systems Information bubble Feedback loops Too powerful platforms Regulation of platforms Interplay with recommendation systems			

Simple solutions for Recommendation systems

Description	What	Limitations/Assumptions	Example/Formula	Reference/Comments
Averaging	A simple solution to get started All users are identical That is, all rows of the matrix are the same	We wish to predice L_{ij} All rows in column **j** are the same Each observation in column **j** is outcome of a coin toss with bias L_{ij} Solution Estimate L_{ij} as the average of all observations in column **j**		
Averaging - How accurate	By law of large numbers But how large should it be By central limit theorem An improved estimate			
Averaging - By law of large numbers	as number of observations in column j grow the estimate converges to the true likelihood			
Averaging - But how large should it be?	one (or few) good rating does not mean the place is excellent			
Averaging - By central limit theorem	the estimation error scale as 1/squareroot(n) with n observations			
Averaging - an improved estimate	average + 1/squareroot(n)			
Averaging - instead we assume All items or providers are identical	estimate: row average (+ correction for number of observations)			
Challenge- content is not structured	How do we convert these attributes or content to features Age: It's a number. That's easy Gender: Two classes or binary. Convert into 0/1 Occupation: Treat as a class. Use one-hot encoding Tip --it has free form text Need an approach to convert text into number or vector of numbers			
Text to vector of number	Create word frequency in document Matrix M Perform PCA of M Each document received k coordinates via k principal components This is the vector representing the text features (restricted to data) Another (more classical) otpion: TF-IDF vector But it can be very large			

TF-IDF vectore	TF-IDF vectorization involves calculating the TF-IDF score for every word in your corpus relative to that document and then putting that information into a vevtor	https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/	