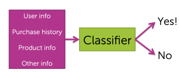
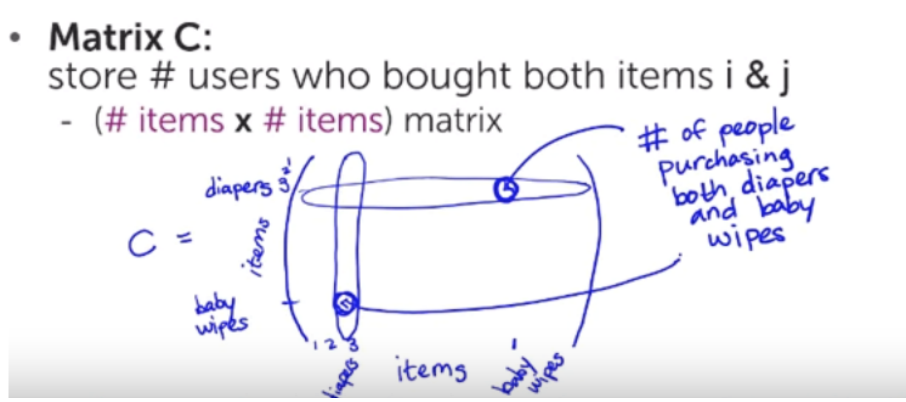
Recommending Products

Types to Recommend Products

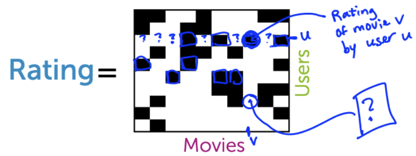
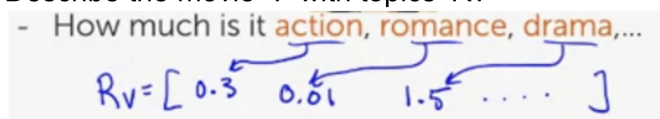
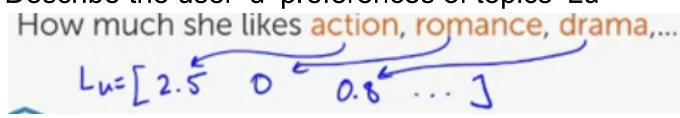
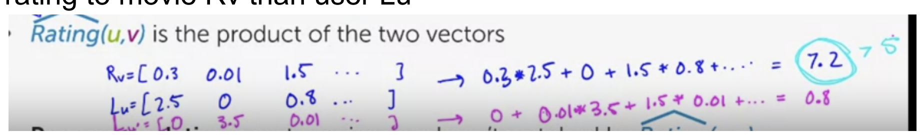
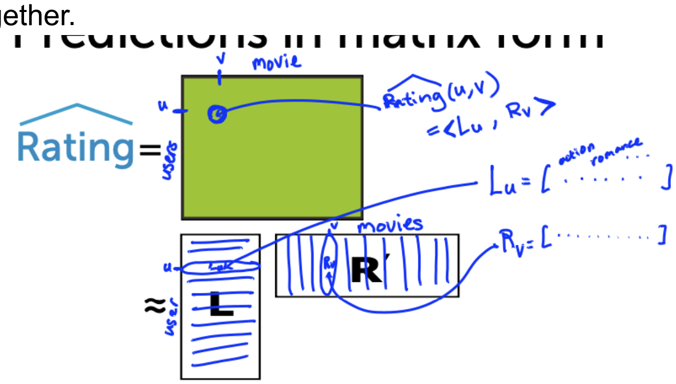
Solution	Pro	Limitation
Popularity	Simplest	No Personalization
Classification 	- Personalized - Can capture context Can handle limited user history	- Features may not be available - Doesn't work as well as collaborative filtering
Collaborative filtering (people who bought this also bought…)	- Personalized - Capture context - Normalized to provide equal weight between popular/unpopular products	- Recommend similar items to the one you bought - Recommendations based on purchase history.
Weighted Average Approach	- Accounts for history of purchases, simpler calculation	Does not utilize - Context (time of day) - User feature(age) - Product features (baby vs electronics) Cold start problem - What if a new user doesn't have a purchase history?
Matrix Factorization	- Provide personalization - Capture context	- Cold start (reliant on past history)

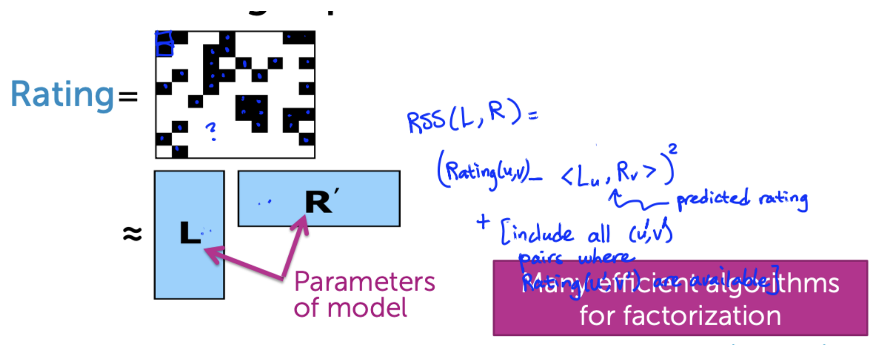
Co-occurrence matrix (for collaborative filtering)

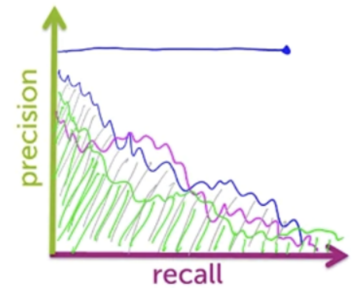
Symmetric matrix that has the same labels on the rows and columns and stores the count of the # of users who bought both items in the matrix



	Let's say the user bought diapers. Pull the diaper vector from the matrix and recommend the items with the largest counts. $\begin{array}{ccccccc} [0 & \cdots & 4 & \cdots & 100 & \cdots &] \\ \text{DVD} & & \text{pacifiers} & & \text{baby wipes} & & \end{array}$
Co-occurrence matrix must be normalized	Very popular items drown out the other effects. The matrix will be recommended based on popularity and not personalized. Let's say you bought several toys, since diapers still have the largest count, you'd get recommended diapers even though other toys might be more suitable. Jaccard Similarity: normalizes by popularity $\frac{\text{who purchased } i \text{ and } j}{\text{who purchased } i \text{ or } j}$ Other similarity metrics are cosine similarity
Weighted Average	User bought (diapers, milk) and we want to see if we should recommend baby wipes too. Score = ½(how many times people who bought diapers bought baby wipes) + ½ (how many times people who bought milk bought baby wipes) Can also weigh more heavily my purchase history to account for context. • User bought items {*diapers*, *milk*} – Compute user-specific score for each item ***j*** in inventory by combining similarities: ***Score*(, *baby wipes*) =** $\frac{1}{2}(S_{\text{baby wipes, diapers}} + S_{\text{baby wipes, milk}})$ – Could also weight recent purchases more • Sort ***Score*(, *j*)** and find item ***j*** with highest similarity
Matrix Factorization Approach	How do we give recommendations and guess what rating a person would give to a movie they've never watched? Matrix shows what rating the user (y-axis) gives to each movie (x-axis) where the black cells are known and white is unknown. We want to fill the white.

	
	Describe the movie ‘v’ with topics ‘Rv’  Describe the user ‘u’ preferences of topics ‘Lu’  Ratings(u,v)-hat is the predicted ratings the user would give to the movie. Done by multiplying the vectors together. User Lu gives higher rating to movie Rv than user Lu’ 
Predictions in matrix form	We can put the predictions into a matrix form with a little linear algebra by multiplying the user preferences and the movie topics together. 
Discovering topics from data	Taking the matrix and approximating it with the LR factorization. So the quality check is the ratings matric against the multiplication of LR

	
Combining Features and Discovered Topics	How to predict a cold start? (new user) Use just the features specified by that user (gender, age) and predict based on that. As we get more data, we can weigh more heavily on our matrix factorization approach and use those learned features in the next predictions. User-specified **feature based** model into **matrix factorization.** Toggle between the two depending on the information available.
Performance Metric	How to assess performance for different systems we might consider using? Don’t use classification accuracy - Classification accuracy is fraction of items correctly classified when we are focused on how quickly can we discover the few liked items? Use **Recall** - How well does the system cover the things i like - $\frac{\#\ liked\ and\ shown}{\#\ liked}$ Use **Precision** - How well does the system not show the things I don’t like - $\frac{\#\ liked\ and\ shown}{\#\ shown}$
Maximize Performance Metric	Maximize recall: - Recommend everything, if you recommend everything, then you definitely show all the liked items - BUT precision would be small Optimal Recommender: - Captures everything user likes and only shows the liked items
Precision-Recall Curves	Shows the tradeoff between precision and recall for different thresholds for a single classifier. For a given precision, want recall as large as possible - find the largest area under the curve (AUC)

	- Set desired recall and maximize precision (precision k) if you know exactly how many products you want to recommend 
Recommender Systems ML Block Diagram	**Training Data** consists of user, product, ratings table and we **feature extract** the user id and product id. Also have gender and age for feature models (for new users) as input x. The x gets put into a **ML model** such as matrix factorization which outputs **predicted ratings** y-hat. The ML model is defined by parameters x-hat of (Lu, Ru) which are features for users and features for products. We compare actual ratings (y) with predicted ratings (y-hat) and take the residual sum of squares which then updates the parameters w-hat.