- Netflix challenge: An open competition for the best collaborative filtering algorithm to predict user ratings for films.
- Oct 2nd, 2006, Netflix began a contest with a million dollar prize to anyone who could improveover their current system by at least 10%in RMSE.
- Data set includes 5-star ratings of 17770 movies and 480,189 anonymous users collected by Netflix. In total there are more 100 million ratings.
- The training set has around 99 million ratings. The goal is to obtain the highest RMSE on a testing test of 3 million ratings.

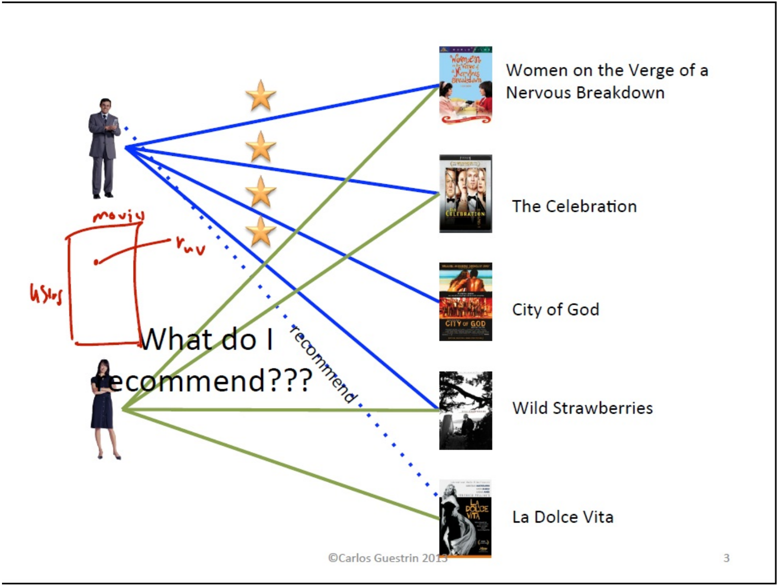

Figure: Carlos Guestrin, University of Washington Stat592

- Recommendation systems are information filtering systems that seek to predict the rating or preference that a user would give to an item.
- Many E-commerce websites are already using recommendation systems to help their customers find products to purchase.
- A recommendation system learns from a customer and recommends products among the available ones based on users' habits and profiles.
- Even online advertising companies can take advantage of these tools to make profits based on users' search and purchase.

Users	Titanic	Batman	Inception	SuperMar	Spiderma	matrix
Michele	2.5		3	3.5	2.5	3
SATYA	3	3.5		5	3	3.5
PRANAV	2.5	3		3.5		4
SURESH		3.5	3	4	2.5	
TOM		4	2	3	2	3
LEO	3	4		5	3.5	3
CHAN		4.5		4	1	

Figure: www.analyticbridge.com

- The most general setting in which recommendation systems are studied: a data matrix of n users and m items, each cells $r_{u,i}$ corresponds to the rating given to item i by the user u.
- User rating matrix is usually sparse as most users don't rate most items.
- The recommendation task is to predict what rating a user would give to a previously unrated item.
- Item that gets highest predicted rating will be recommended to the user.
- The user under current consideration for recommendations is called the active user.

- Two main strategies to build recommendation systems.
- Content-based filtering (CT): recommend items that are similar in content to items the user has liked in the past (or matched to attributes of the user).
- Collaborate based filtering (CF): a user is recommended items based on past ratings of all users collectively.
- Example: CT filtering: the active user likes Starwar I,II, and III, recommend The force awakens.
- CF filtering: user A likes Starwars movies. User B,C, and D like Starwars and Startrek. Recommendation to A based on preferences of people that watch similar movies to A.

- Collaborative filtering: Neighborhood-based and model-based approaches.
- In neighborhood-based methods, a subset of users are chosen based on their similarity to the active user. A weighted combination of their ratings is used to produce predictions for the user.
    1. Assign weights to all users with respect to similarity with the active user.
    2. Choose k users with highest similarity.
    3. Compute a prediction from a weighted combination of the selected neighbors' ratings.

- Model-based collaborative filtering: provide recommendations by estimating parameters of statistical models for user ratings.
- Latent factor and matrix factorization models have emerged as the state of the art methodology in this class of techniques.
- These methods were important part of the winning strategy of Netflix prize.
- Latent factor models assume that the similarity between users and items are induced by some hidden lower-dimensional structure in the data.

- Latent factor models try to explain ratings by characterizing on both users and items based on a number of factors inferred from the rating patterns.
- Example: For movies, factors could cover unseen dimensions such as action movies vs sci fi vs comedies, even less well defined aspects such as character development or some inexplicable dimension.
- From users' side, each factor will measure how much the user like movies that score high on those corresponding factor.

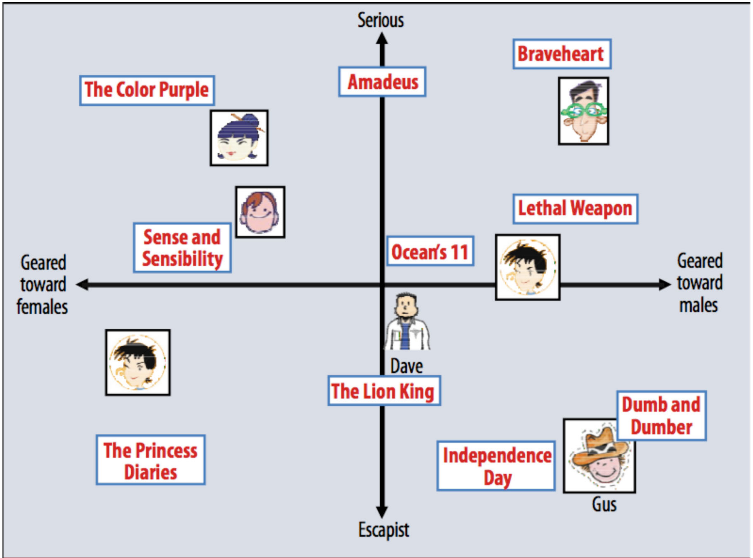

Figure 2. A simplified illustration of the latent factor approach, which characterizes both users and movies using two axes—male versus female and serious versus escapist.

- A realization of latent factor model is based on matrix factorization.
- These types of methods can scale to high dimension data sets and have good prediction accuracy.
- A strength of matrix factorization is that it allows the model to infer *implicit feedbacks* by observing user behavior including purchase history, browsing history, search patterns.

- Matrix factorization models map both users and items to a joint latent factor space of dimension f.
- User item interaction is modeled as inner products of vector in that space.
- Each item i has a corresponding vector $q_i \in \mathcal{R}^f$. Each user u has a vector $p_u \in \mathcal{R}^f$.
- Each entry q_i measure the extent to which the item possesses those factors.
- Each entry in p_u measure the interest of the user has in items that score high on corressponding factors.

- The dot product $\langle q_i, p_u \rangle = q_i^T p_u$ measures the interaction between user u and item i (or preference).
- This is used to model the rating of item i from user u.
- $\hat{r}_{u,i} = \langle q_i, p_u \rangle = q_i^T p_u$.
- To learn the parameters of the model, solve the following problem:

$$\min_{p,q} \sum_{(u,i)\in K} (r_{u,i} - q_i^T p_u)^2$$

,where K is the set of pairs (u,i) that are ratings from user u of item i.

- To avoid over fitting and commputation purpose, a regularized problem is solved:

$$\min_{p,q} \sum_{(u,i)\in K} (r_{u,i} - q_i^T p_u)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2), \lambda \geq 0.$$

- The problem is actually not convex in terms of p_u and q_i, fortunately it is still differentiable.
- Approximation algorithm: Define the predictin error (or residual at the current solution)

$$e_{u,i} = r_{u,i} - q_i^T p_u.$$

Then the next solution estimate is updated by the following rule:

$$\begin{aligned} q_i &\leftarrow q_i + \alpha(e_{u,i} p_u - \lambda q_i) \\ p_u &\leftarrow p_u + \alpha(e_{u,i} q_i - \lambda p_u). \end{aligned}$$

- This resembles the gradient descent method that we have seen.

- The matrix factorization model is flexible in dealing with different data aspects and application specific requirements.
- Adding bias to the models: Some users tend to give higher ratings to items than others. Some items also tend to get higer ratings than similar items.
- A simple model for bias associated with $r_{u,i}$:

$$b_{u,i} = \mu + b_i + b_u.$$
$$r_{u,i} = \mu + b_u + b_i + q_i^T p_u.$$

, where μ is the overall average rating. b_u and b_i models the deviation from the average of ratings from user u and i tem i.

- The estimation comes from solving the problem:

$$\min_{p,q,b} \sum_{(u,i)\in K} (r_{u,i} - \mu - b_u - b_i q_i^T p_u)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2), \lambda \geq$$

- The matrix factorization model can be modified to incorporate time effects on ratings.

$$r_{u,i}(t) = \mu + b_u(t) + b_i(t) + q_i^T p_u(t).$$

- Matrix factorization model allows incorporation of additional information.
- An user u express implicit preference for a set of items denoted by N(u). This is characterized by: $\sum_{i \in N(u)} x_i$.
- Similar attributes can be added in this way: demographic, income, age group...
- $$\hat{r}_{u,i} = \mu + b_u + b_i + q_i^T(p_u + \sum_{i \in N(u)} x_i)$$