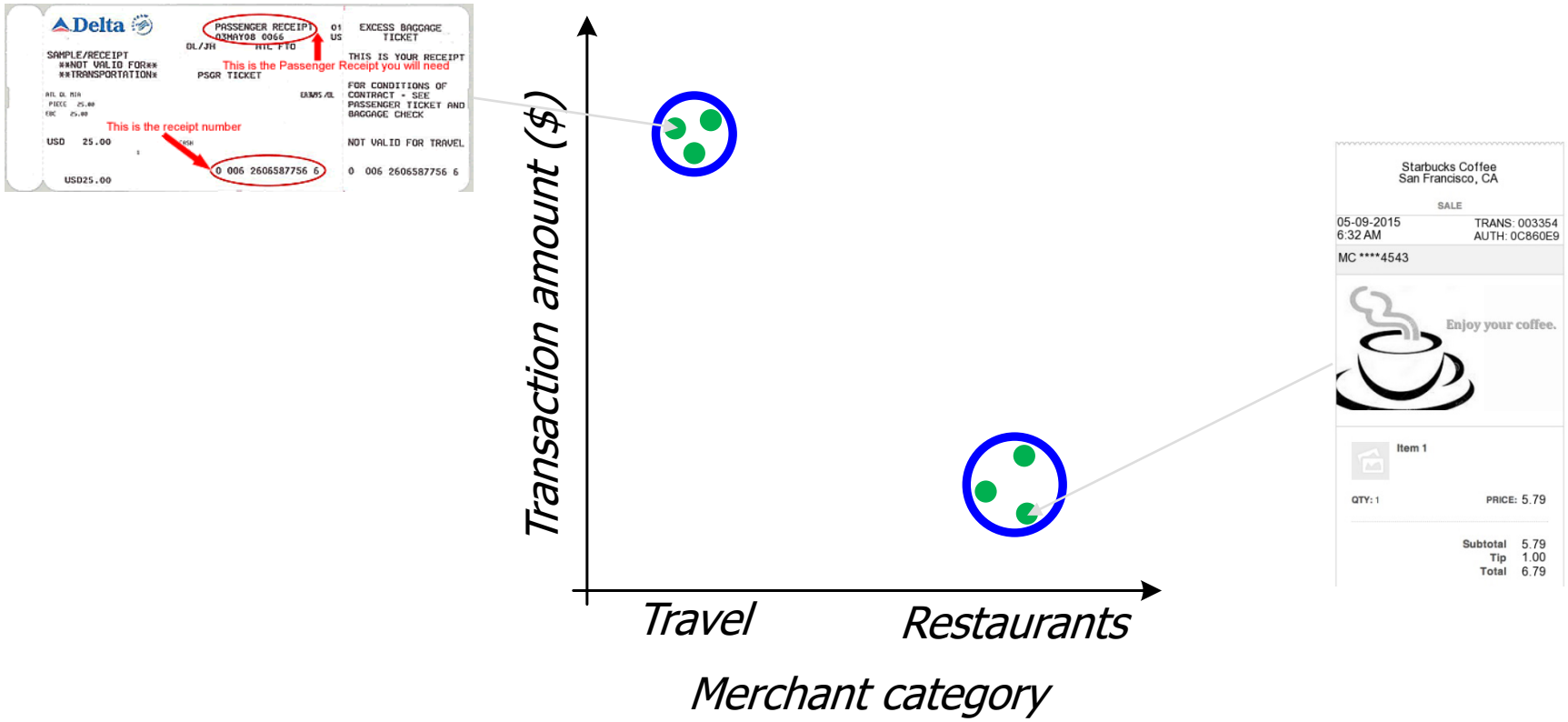
Recommender Systems

Review: Machine Learning

- **Machine Learning** applies statistics to **datasets** to predict **future data**
- **Supervised Learning** uses **labeled datasets**
  - Datapoints represented as vectors of **feature** values
    - e.g., height, weight, age, sex, income, blah blah blah
  - Datapoints each annotated with a **label**
    - "this person is cool or not cool"
  - Algorithm like SVM or regression can **derive a function** $\hat{f}(x)$ that produces a **label** that minimizes the **error** between $\hat{f}(x)$ and the intended labels
- **Unsupervised Learning** uses **unlabeled datasets**
  - We can use **K-means clustering** to automatically find **centroids**
- We often employ **k-fold cross validation** to detect *underfitting* or *overfitting* by splitting our dataset into **train** and **test** "folds"

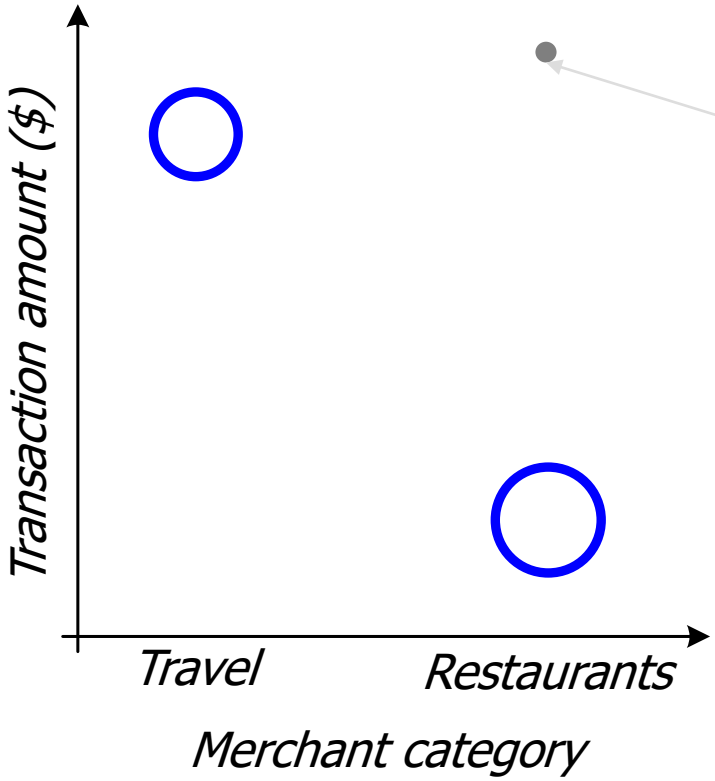
Unsupervised learning example

- Example: credit card fraud detection



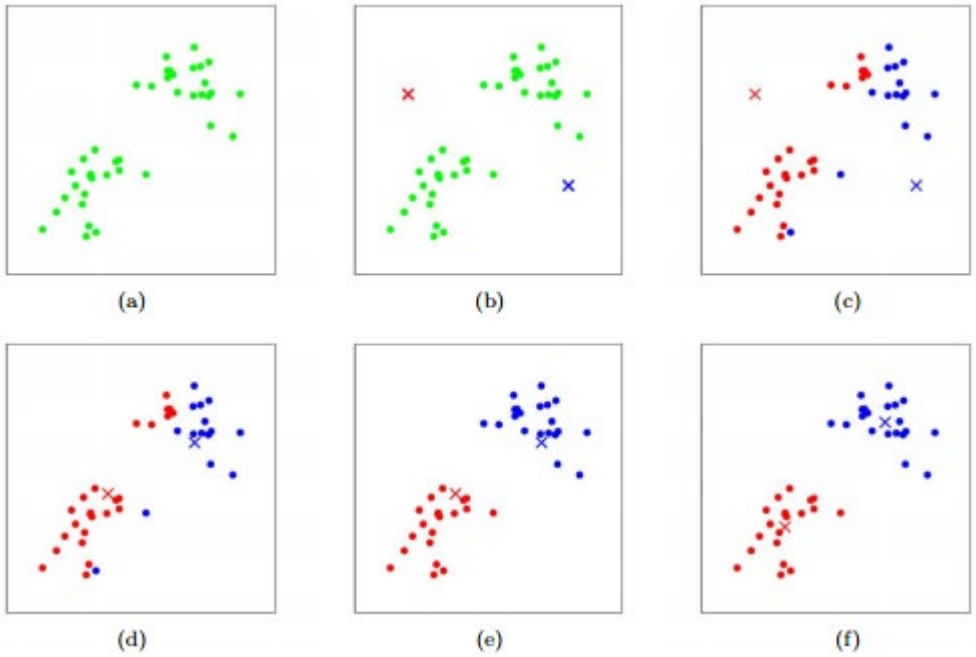
Unsupervised learning example

- Good for anomaly detection



anomaly

K-means clustering



- Very popular technique, as follows:
  - Grab a distance metric between two points
  - Choose the number of clusters = k
  - Generate k random **centroids**
- Repeat the following:
  - Assign each data item to the closest center
  - Choose new cluster centroids
  - Iterate until centroids converge (i.e., they don't change much)

- In practice, **distance metric** matters more than clustering algorithm

When does k-means fail?

- What if clusters are **oblongs**?
  - Rectangles?
  - Hourglasses?
- What if clusters **overlap**?
  - Document subsets?
  - Image closeups?
- What if clusters are different **sizes**?
  - People cloning wikipedia.org vs. people cloning cafarella.com
  - Consider both volume and # points

How to pick *k*?

- Difficult without domain knowledge
- Agglomerative clustering
  - Start one cluster per example
  - Merge the two closest clusters
  - Repeat until you've got one cluster
  - Output result

- How do you measure cluster closeness?
  - Distance between centroids?
  - Min distance between pairs? (or max?)

Cluster evaluation

- Need the "right" number of "good" clusters
- Correctness of a cluster is easy
  - Do members belong together?
  - Roughly similar to precision
- Testing whether clusters are "right" is harder
- Multiple good clusterings possible for a single dataset
- In general, evaluation is **much** harder than with supervised learning

Important questions

- How do you measure similarity?
- How do you construct the clustering?
- How do you evaluate the outcome?

Cluster similarity measurement

- Euclidean distance (for reals)
- Jaccard distance (for set overlap)
- Bit distance (for vectors of booleans)

- Many others possible, depending on your application
- How would you measure similarity when clustering:
  - Images?
  - Videos?
  - Schemas?

POLICY & LAW \ US & WORLD \ NET NEUTRALITY \

Congress just cleared the way for internet providers to sell your web browsing history

Resolution is now off to the president's desk

by Jacob Kastrenakes | Mar 28, 2017, 5:57pm EDT

By THE ASSOCIATED PRESS MARCH 23, 2017, 6:54 P.M. E.D.T.

NEW YORK — The Senate voted to kill Obama-era online privacy regulations , a first step toward allowing internet providers such as Comcast, AT&T and Verizon to sell your browsing habits and other personal information as they expand their own online ad businesses.

Ethics and machine learning

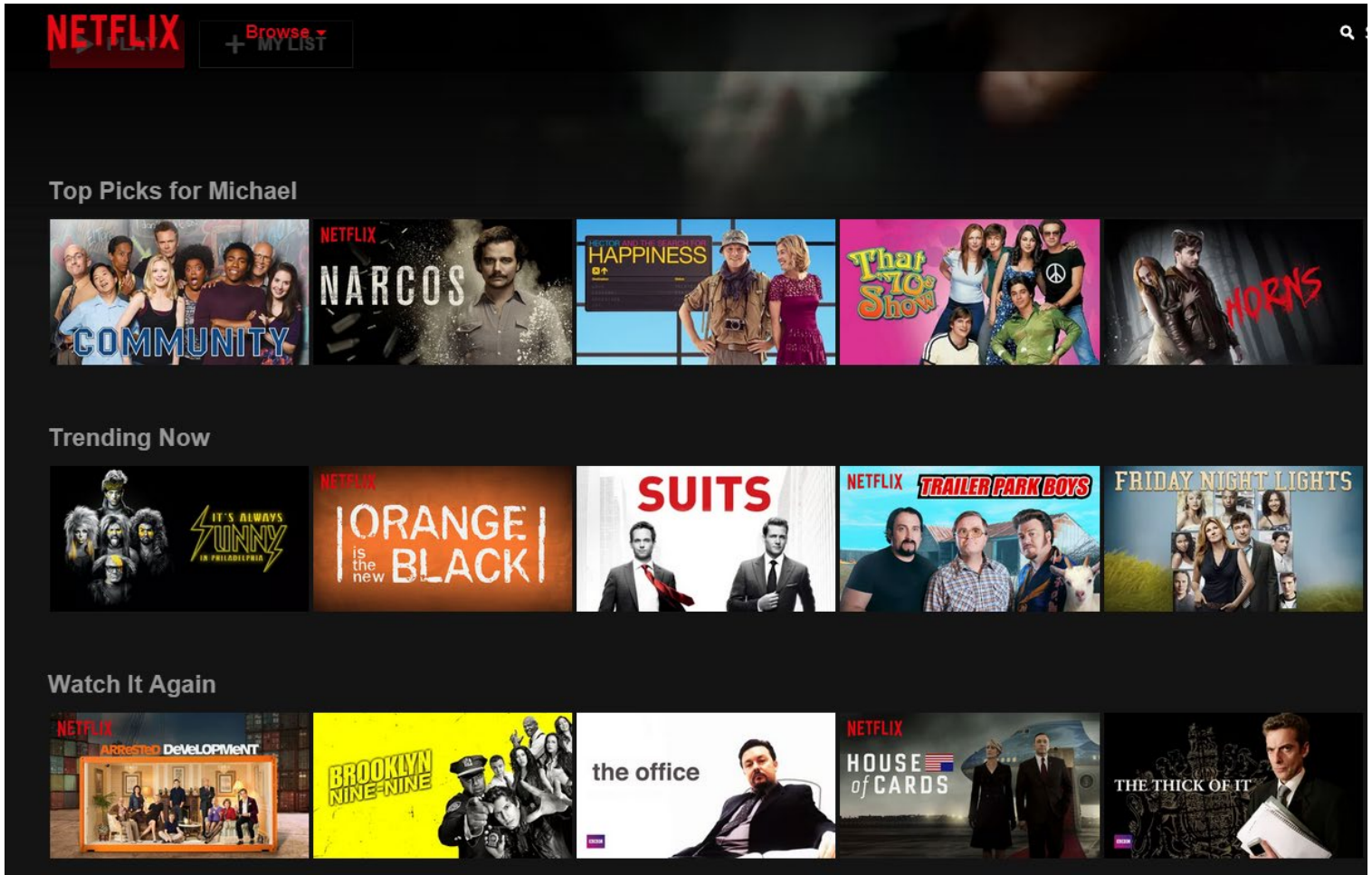
- Many data mining projects are ethically and politically contentious
  - Credit card offers
  - Financial trades
  - Total Information Awareness project (TIA)
- Many data-mining projects are ethically complicated because of the data used
  - Is the privacy-leaking AOL data OK?
  - What's so wrong about collecting WiFi info?

One Slide Summary: Recommendation Systems

- **Recommender systems** try to *predict what users like*
- **User-based Collaborative filtering** tries to predict recommendations based on *similarity between users*
  - "Since you're the same age, gender, income-level, etc. as these 1000 other people that bought bananas, here's a bunch of other things you'll like."
  - Considers *distance between users* when predicting recommendations
- **Content-based filtering** tries to predict recommendations based on *historical trends for one user*
  - "Since you've listened to 'Love Story' by T Swifty 60,000 times, here's some other songs you'd like based on the genre, FFT, etc."
  - Considers *distance between items* when predicting recommendations

Example

- Movies (e.g., Netflix, Amazon, Hulu, etc.)

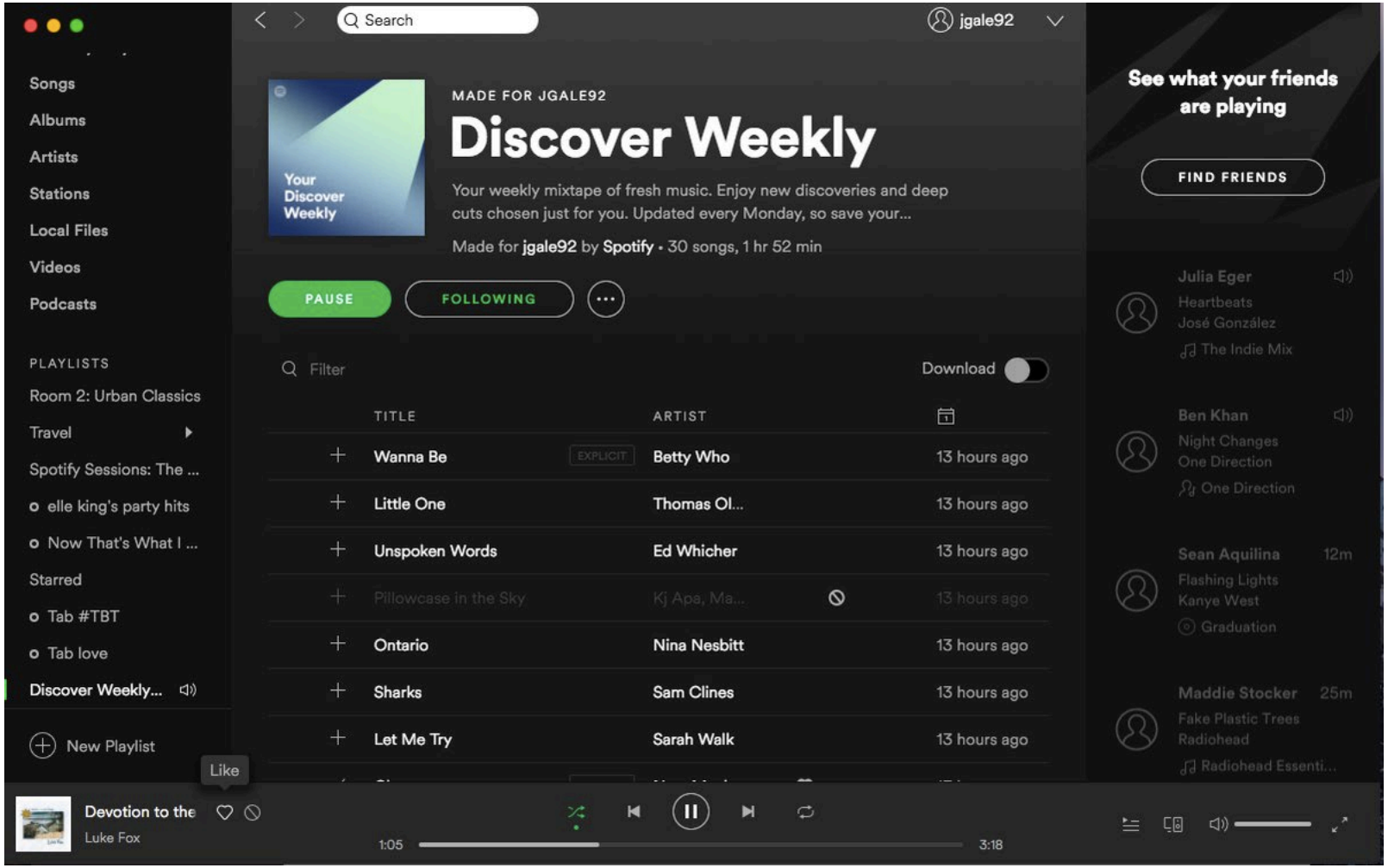


Example

- Products (e.g., Amazon, etc.)

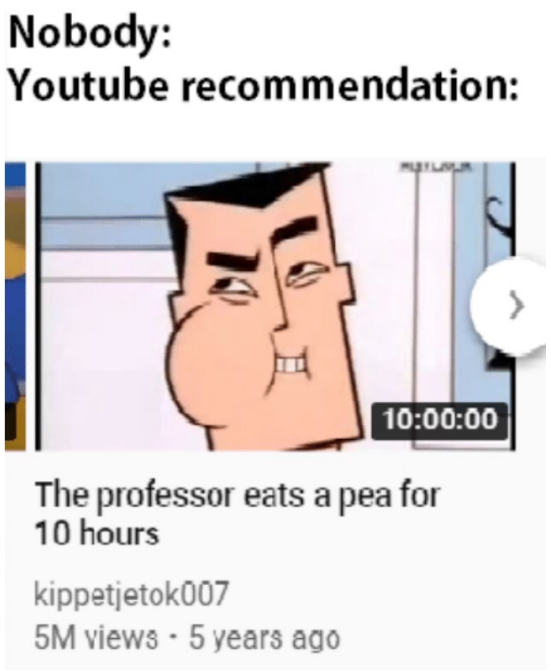
Example

- Music (e.g., Spotify, Apple Music, etc.)



Example

- Recommender Systems predict what you'll like
- Put another way: they predict your preferences
- More examples
  - Products
  - Movies
  - Music
  - Books
  - Video games
  - Colleagues
  - Friends



Recommendation challenges

- Recommendations must be made among thousands (millions?) of products or selections
  - *Representation:* what features do we use to capture meaningful information about users or selections?

- Users hate telling you preferences
  - They either can't be bothered
  - Or they're bad at knowing…
- Everyone is different (right?)
  - Alternatively: the better a recommendation system works for you, the more boring you are

Data collection

- **Data** drives all recommendation systems (and ML in general)

- **Explicit** data collection
  - Ask users to rate movies, products, whatever
  - Ask users for demographic information

- **Implicit** data collection
  - Web logs of past user activity
  - Timing information, e.g., how long you paused on a post before scrolling

Data collection

- How to predict what movies you like? Features?
- One approach: do it like Web pages
  - Collect data on my movie ratings and preferences

Rating (from user):

The Godfather	Year…	Genre….	4
Ernest Goes to Camp			3
Casablanca			2
36 Hours			5
Love and Death			4

  - Collect features: genre, length, year, etc.
  - Build score-predictor; recommend high-scorers
- Problems?

Collaborative filtering

- Recommend movies enjoyed by *people who are similar to you*
- Basic assumption: People who agreed in the past will agree in the future

- We can use **averaging** to derive **scores** across users to improve data sparsity
  - But this usually performs poorly where wide variance in preference exists
- We use **k-nearest neighbor** to find users that are **nearby** a given user before applying averaging

Averaging example

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	?	2	5	1
Chris	4	2	4	?
Donna	3	?	5	1

Averaging example

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	3.66	2	5	1
Chris	4	2	4	2
Donna	3	2	5	1

Averaging

- Ignores specific aspects of a user
- Terrible when there is large variation in interest
  - Music, movies are a few examples

- How can we take into account uniqueness of a user?

Nearest neighbor algorithm

- Find another user with similar ratings
- Use other user's rating to "fill in the blank"
- People who agreed in the past will agree in the future

Nearest neighbor example

- Who are the nearest neighbors?

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	?	2	5	1
Chris	4	2	4	?
Donna	3	?	5	1

Nearest neighbor example

- Bob's nearest neighbor is Donna

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	?	2	**5**	**1**
Chris	4	2	4	?
Donna	3	?	**5**	**1**

Nearest neighbor example

- Use Donna's rating to fill in Bob's

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	**3**	2	5	1
Chris	4	2	4	?
Donna	**3**	?	5	1

Nearest neighbor example

- Chris's nearest neighbor is Alice

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	3	2	5	1
Chris	4	2	4	**4**
Donna	3	?	5	1

Nearest neighbor example

- Donna's nearest neighbor is Bob

	W.	Xanadu	Youngblood	Zorro
Alice	4	2	4	4
Bob	3	2	5	1
Chris	4	2	4	4
Donna	3	**2**	5	1

Distance metric

- How to measure **distance** between neighbors?

- Many possibilities, including
  - Euclidean distance
  - Cosine similarity
  - Pearson correlation coefficient

- This should remind you of the IR lectures

Euclidean distance

- Each movie rated by both users is a dimension
- One user is one vector in multidimensional space
- Measure distance between the two vectors

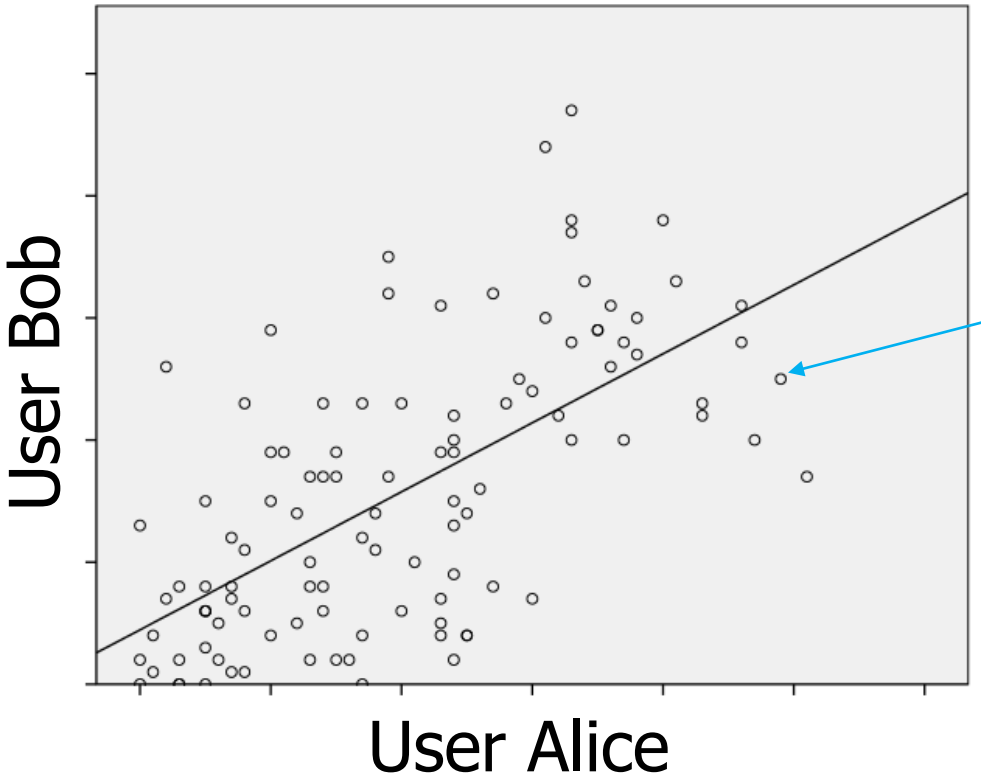
Cosine similarity

- Each movie rated by both users is a dimension
- One user is one vector in multidimensional space

- Cosine similarity = 1
  - Vectors "pointed in the same direction"
- Cosine similarity = 0
  - Vectors orthogonal
- Cosine similarity doesn't consider the length of the vector, only the angle

Pearson correlation coefficient

- Each movie rated by both users is a point



Rating of one movie by Bob and Alice

Pearson correlation coefficient

- Value between +1 and −1
  - 1 is total positive correlation
  - 0 is no correlation
  - −1 is total negative correlation

Pearson correlation coefficient

- S = set of movies
- $r_{u,i}$ = rating of user u on movie *i*
- S_u = {i ∈ S | *u* saw movie *i*}
- S_{uv} = {i ∈ S | both *u* and *v* saw movie *i*}

$$r_u = \sum_{i \in S_u} r_{u,i} / |S_u|$$

$$\frac{\sum_{i \in S_{uv}} (r_{u,i} - r_u)(r_{v,i} - r_v)}{\sqrt{\sum_{i \in S_{uv}} (r_{u,i} - r_u)^2 \sum_{i \in S_{uv}} (r_{v,i} - r_v)^2}}$$

K-nearest neighbors (k-NN)

- Can we do better than selecting just one nearest neighbor?
- Select several.

1. Find the *k* closest users
2. Select the most frequent score (mode)

k-NN problems

1. Find the k closest users
2. Select the most frequent score

- How would a really popular film affect recommendations?

k-NN problems

1. Find the k closest users
2. Select the most frequent score

- How would a really popular film affect recommendations?
- The popular film will be recommended most of the time because it will often be the most frequent score
- The popular choice might not always be the best choice for a user

k-NN with weights

- Solution: weight other users' scores by similarity

$$r_{u,i} = k \sum_{v \in Top-Sim(u)} sim(u,v) r_{v,i}$$

- Top-Sim(u) is the *n* most-similar user neighbors to *u*
- *k* is a normalizer

$$k = 1 / \sum_{v \in Top-Sim(u)} |sim(u,v)|$$

k-NN problems

- How would a "cult classic" film affect recommendations?
  - Enjoyed by few, but they REALLY like it
- Rare film less likely to be predicted by k-NN

- Can use *inverse-user-frequency*
  - Similar to TFxIDF's inverse document frequency
  - Rarely seen movies are more influential

Thought Questions

- Suppose this is a projection of different users that have rated books. What can you conclude about the point that is marked?

- What would be a good value for k if you wanted to use k-nearest neighbors to predict which books you would enjoy? Why would k = 5 not be a good choice?

Collaborative filtering weaknesses

- Cold start
  - Need user data before you can start making accurate recommendations
- Scalability
  - Millions of users, n-choose-2 pairs
  - Large amount of computation power
- Sparsity
  - Most users will only have rated a small subset of the overall database
  - Even the most popular items have very few ratings

Representation

- Each user is associated with a vector of items
  - Each item has a vector of features

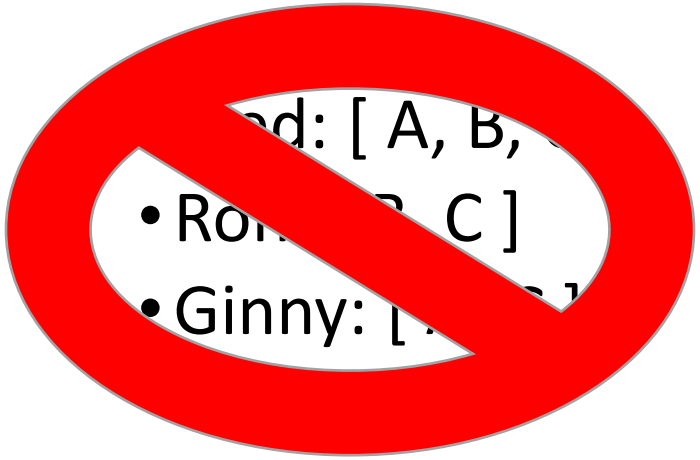
- Fred has items A, B, and C in shopping cart
- Ron has items B and C in shopping cart
- Ginny has items A, and C in shopping cart

- Fred: [A, B, C]
- Ron: [B, C]
- Ginny: [A, C]

Representation

- Each user is associated with a vector of items
  - Each item has a vector of features

- Fred has items A, B, and C in shopping cart
- Ron has items B and C in shopping cart
- Ginny has items A, and C in shopping cart



Fred: [1 1 1 0]
Ron: [0 1 1 0]
Ginny: [1 0 1 0]

Content-based filtering

- Content-based filtering: recommend items similar *to items the user has liked in the past*

- Basic assumption: People will like the same things in the future that they liked in the past

- One solution to collaborative filtering's problems with scalability and sparsity
  - Avoids nearest-neighbor operations on users

Content-based filtering

- For test item *i*, find *k* most-similar items the user has rated previously
  - *i*'s score is a weighted combination of user's ratings on those *k* items

- How can we compute item similarity?
  - Can use cosine or correlation
  - The vector for item *i* is the set of user-reviews associated with *i*

Target and pregnancy prediction

1. Predict pregnancy
2. Recommend pregnancy related products
3. Father angry when teen daughter receives ads
4. Father finds out daughter is pregnant

How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did

Kashmir Hill Forbes Staff

Welcome to The Not-So Private Parts where technology & privacy collide

Every time you go shopping, you share intimate details about your consumption patterns with retailers. And many of those retailers are studying those details to figure out what you like, what you need, and which coupons are most likely to make you happy. Target TGT +0%, for example, has figured out how to data-mine its way into your womb, to figure out whether you have a baby on the way long before you need to start buying diapers.

TARGET

Target has got you in its aim

Target and pregnancy prediction

- Features were products
  - Unscented lotion, calcium, magnesium and zinc supplements
- Sent ads through the mail
- Story from 2012

Netflix prize

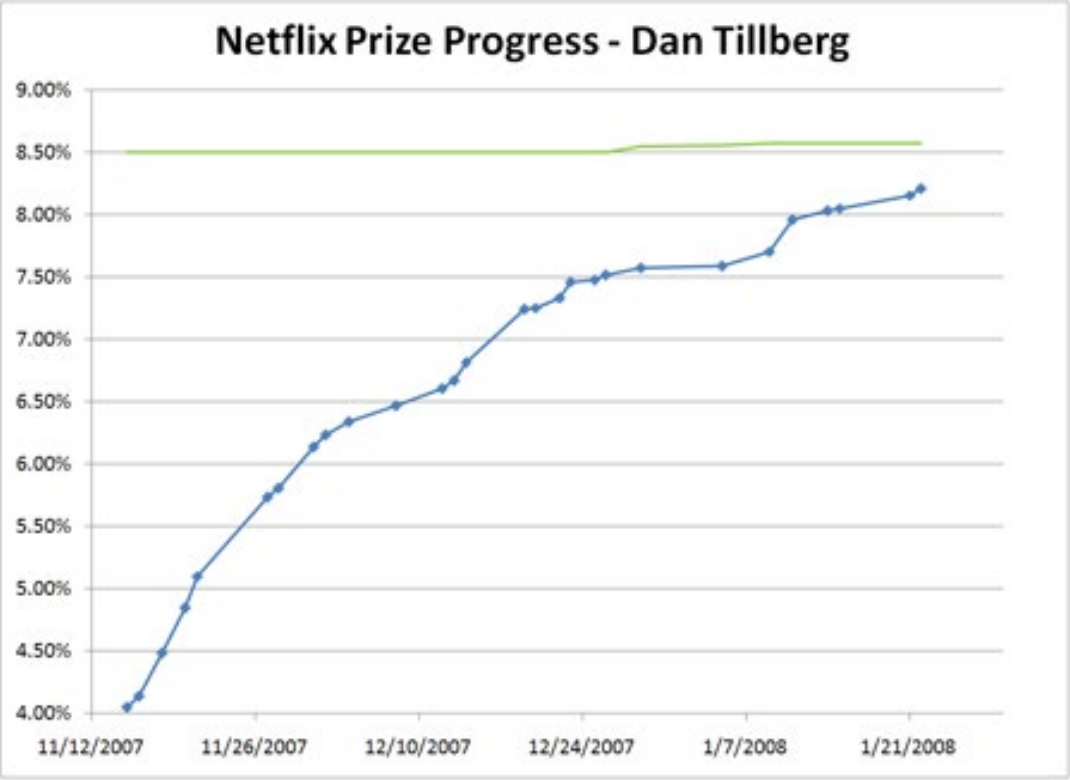
- Announced October 2, 2006
- $1M to whoever could improve NetFlix's own recommender by 10%
- Data from Netflix in the form:
  - <user, movie, date, grade>
  - Where grade is 1...5
  - That's it
- Training data: 100M examples from ~500k users on ~18k movies
- Ideas for how you might do this?

Netflix prize

- How did they do it?
- Among many ideas:
  - Use IMDB for director, genre, etc.
  - Some movies' grades change when reviewers grade them in a clump, suggesting a long time has passed since the movie was viewed
  - User grades depend on day of week

Netflix prize

- Dan Tillberg's progress…
- Dropped out in 2008 while in 5th place



Netflix prize

- Won on September 21, 2009

Summary

- User-based collaborative filtering: recommend items enjoyed by *users who are similar to you*

- Content-based filtering: recommend items *similar to items you have enjoyed in the past*