DATA MINING
SIMILARITY & DISTANCE

Similarity and Distance

Recommender Systems

SIMILARITY AND DISTANCE

Thanks to:

Tan, Steinbach, and Kumar, “Introduction to Data Mining”

Rajaraman and Ullman, “Mining Massive Datasets”

Similarity and Distance

- For many different problems we need to quantify how close two objects are.
- Examples:
  - For an item bought by a customer, find other similar items
  - Group together the customers of a site so that similar customers are shown the same ad.
  - Group together web documents so that you can separate the ones that talk about politics and the ones that talk about sports.
  - Find all the near-duplicate mirrored web documents.
  - Find credit card transactions that are very different from previous transactions.
- To solve these problems we need a definition of similarity, or distance.
  - The definition depends on the type of data that we have

Similarity

- Numerical measure of how alike two data objects are.
  - A function that maps pairs of objects to real values
  - Higher when objects are more alike.
- Often falls in the range [0,1], sometimes in [-1,1]

- Desirable properties for similarity
  1. $s(p, q) = 1$ (or maximum similarity) only if $p = q$. (Identity)
  2. $s(p, q) = s(q, p)$ for all p and q. (Symmetry)

Similarity between sets

- Consider the following documents

apple releases new ipod

apple releases new ipad

new apple pie recipe

- Which ones are more similar?

- How would you quantify their similarity?

Similarity: Intersection

- Number of words in common

apple
releases
new ipod

apple
releases
new ipad

new
apple pie
recipe

- Sim(D,D) = 3, Sim(D,D) = Sim(D,D) =2
- What about this document?

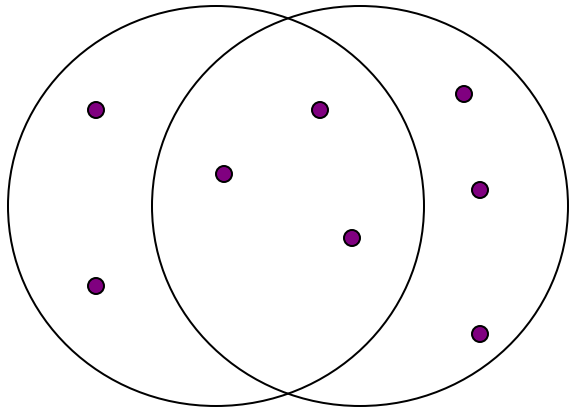
Akis releases new book
with apple pie recipes

- Sim(D,D) = Sim(D,D) = 3

Jaccard Similarity

- The Jaccard similarity (Jaccard coefficient) of two sets S_1, S_2 is the size of their intersection divided by the size of their union.

$$JSim\ (S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$



3 in intersection.
8 in union.
Jaccard similarity = 3/8

- Extreme behavior:
  - $JSim(X, Y) = 1$, iff $X = Y$
  - $JSim(X, Y) = 0$ iff X, Y have no elements in common
- $JSim$ is symmetric

Probabilistic interpretation:
The Jaccard similarity of two sets is the probability that a randomly selected element from the union is in the intersection

Jaccard Similarity between sets

- The distance for the documents

apple releases new ipod

apple releases new ipad

new apple pie recipe

Akis releases new book with apple pie recipes

- JSim(D,D) = 3/5
- JSim(D,D) = JSim(D,D) = 2/6
- JSim(D,D) = JSim(D,D) = 3/9

Similarity between vectors

Documents (and sets in general) can also be represented as vectors

document	Apple	Microsoft	Obama	Election
D1	10	20	0	0
D2	30	60	0	0
D3	60	30	0	0
D4	0	0	10	20

How do we measure the similarity of two vectors?

- We could view them as sets of words. Jaccard Similarity will show that D4 is different form the rest
- But all pairs of the other three documents are equally similar

We want to capture how well the two vectors are aligned

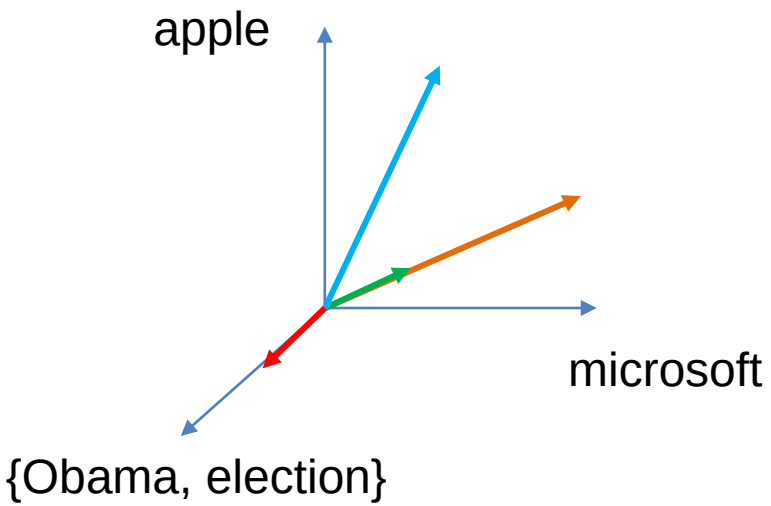
Example

document	Apple	Microsoft	Obama	Election
D1	10	20	0	0
D2	30	60	0	0
D3	60	30	0	0
D4	0	0	10	20

Documents D1, D2 are in the “same direction”

Document D3 is on the same plane as D1, D2

Document D4 is orthogonal to the rest



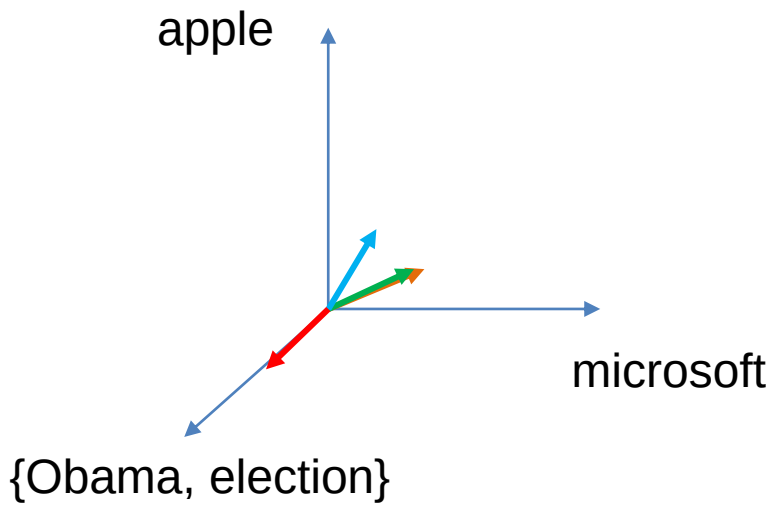
Example

document	Apple	Microsoft	Obama	Election
D1	10	20	0	0
D2	30	60	0	0
D3	60	30	0	0
D4	0	0	10	20

Documents D1, D2 are in the “same direction”

Document D3 is on the same plane as D1, D2

Document D4 is orthogonal to the rest



Cosine Similarity

- $Sim(X, Y) = \cos(X, Y)$
  - The cosine of the angle between X and Y

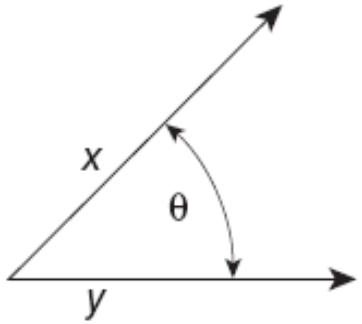


Figure 2.16. Geometric illustration of the cosine measure.

- If the vectors are aligned (correlated) angle is zero degrees and $\cos(X, Y) = 1$
- If the vectors are orthogonal (no common coordinates) angle is 90 degrees and $\cos(X, Y) = 0$

- Cosine is commonly used for comparing documents, where we assume that the vectors are normalized by the document length, or words are weighted by tf-idf.

Cosine Similarity – Math

- If x and y are two vectors, then

$$\cos(x, y) = \frac{x \cdot y}{\|x\|\|y\|}$$

where $x \cdot y = \sum_{i=1}^{n} x_i y_i$ is the dot product of x and y

Note: We only need to consider the non-zero entries of the vectors

Example:

$$x = 3\ 2\ 0\ 5\ 0\ 0\ 2\ 0\ 0$$

$$y = 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2$$

$$x \cdot y = 3 \cdot 1 + 2 \cdot 0 + 0 \cdot 0 + 5 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 + 2 \cdot 1 + 0 \cdot 0 + 0 \cdot 0 = 12$$

$$\|x\| = \sqrt{3^2 + 2^2 + 0^2 + 5^2 + 0^2 + 0^2 + 2^2 + 0^2 + 0^2} = \sqrt{42} = 6.481$$

$$\|y\| = \sqrt{1^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 1^2 + 0^2 + 2^2} = \sqrt{6} = 2.245$$

$$\cos(x, y) = 0.315$$

What if we have 0/1 vectors?

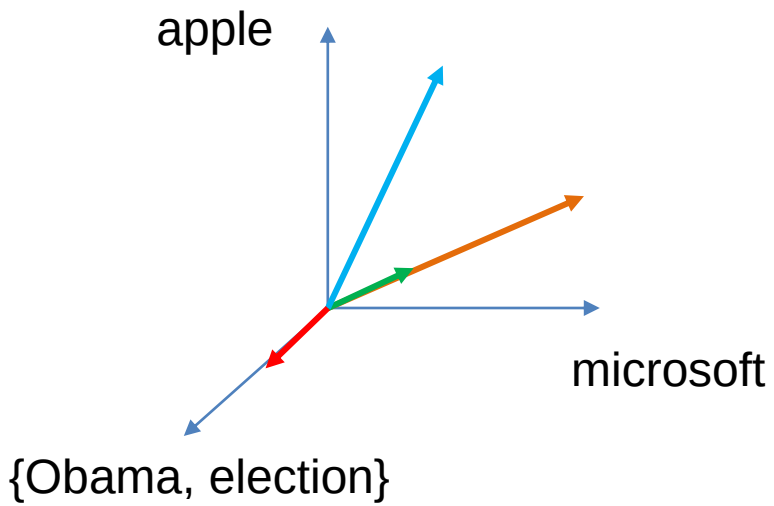
Example

document	Apple	Microsoft	Obama	Election
D1	10	20	0	0
D2	30	60	0	0
D3	60	30	0	0
D4	0	0	10	20

Cos(D1,D2) = 1

Cos (D3,D1) = Cos(D3,D2) = 4/5

Cos(D4,D1) = Cos(D4,D2) = Cos(D4,D3) = 0



Correlation Coefficient

- The correlation coefficient measures correlation between two random variables.
- If we have observations (vectors) $X = (x_1, \dots, x_n)$ and $Y = (y_1, \dots, y_n)$ is defined as

$$CorrCoeff(X,Y) = \frac{\sum_i (x_i - \mu_X)(y_i - \mu_Y)}{\sqrt{\sum_i (x_i - \mu_X)^2}\sqrt{\sum_i (y_i - \mu_Y)^2}}$$

- This is essentially the cosine similarity between the centered vectors (where from each entry we remove the mean value of the vector.
- The correlation coefficient takes values in [-1,1]
  - -1 negative correlation, +1 positive correlation, 0 no correlation.
- Most statistical packages also compute a p-value that measures the statistical importance of the correlation
  - Lower value – higher statistical importance

Correlation Coefficient

Normalized vectors

document	Apple	Microsoft	Obama	Election
D1	-5	+5	0	0
D2	-15	+15	0	0
D3	+15	-15	0	0
D4	0	0	-5	+5

$$CorrCoeff(X,Y) = \frac{\sum_i (x_i - \mu_X)(y_i - \mu_Y)}{\sqrt{\sum_i (x_i - \mu_X)^2}\sqrt{\sum_i (y_i - \mu_Y)^2}}$$

CorrCoeff(D1,D2) = 1

CorrCoeff(D1,D3) = CorrCoeff(D2,D3) = -1

CorrCoeff(D1,D4) = CorrCoeff(D2,D4) = CorrCoeff(D3,D4) = 0

Distance

- Numerical measure of how different two data objects are
  - A function that maps pairs of objects to real values
  - Lower when objects are more alike
  - Higher when two objects are different
- Minimum distance is 0, when comparing an object with itself.
- Upper limit varies

Distance Metric

- A distance function d is a distance metric if it is a function from pairs of objects to real numbers such that:
  1. $d(x, y) \geq 0$. (non-negativity)
  2. $d(x, y) = 0$ iff $x = y$. (identity)
  3. $d(x, y) = d(y, x)$. (symmetry)
  4. $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality).

Triangle Inequality

- Triangle inequality guarantees that the distance function is well-behaved.
  - The direct connection is the shortest distance

- It is useful also for proving useful properties about the data.

Example

- We have a set of objects $X = \{x_1, \dots, x_n\}$ of a universe U (e.g., $U = \mathbb{R}^d$), and a distance function d that is a metric.
- We want to find the object $z \in U$ that minimizes the sum of distances $\sum_{x \in X} d(x, z)$ from the objects in X.
  - For some distance metrics this is easy, for some it is an NP-hard problem.
- It is easy to find the object $x^* \in X$ that minimizes the distances from all the objects in X (simply examine each point)
- But how good is this? We can prove that

$$\sum_{x \in X} d(x, x^*) \leq 2 \sum_{x \in X} d(x, z)$$

  - We are a factor 2 away from the best solution.

Distances for real vectors

- Vectors $x = (x_1, \dots, x_d)$ and $y = (y_1, \dots, y_d)$

L_p norms are known to be distance metrics

- $\boldsymbol{L_p}$-norms or Minkowski distance:

$$L_p(x, y) = [|x_1 - y_1|^p + \cdots + |x_d - y_d|^p]^{1/p}$$

- $\boldsymbol{L_2}$-norm: Euclidean distance:

$$L_2(x, y) = \sqrt{|x_1 - y_1|^2 + \cdots + |x_d - y_d|^2}$$

- $\boldsymbol{L_1}$-norm: Manhattan distance:

$$L_1(x, y) = |x_1 - y_1| + \cdots + |x_d - y_d|$$

- $\boldsymbol{L_\infty}$-norm:

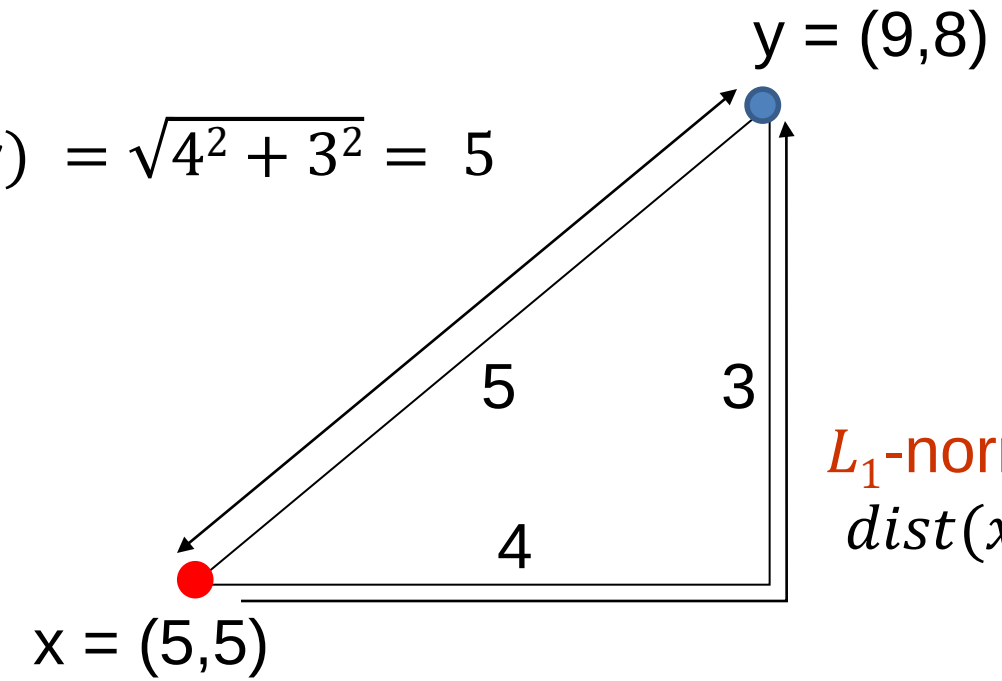
$$L_\infty(x, y) = \max\{|x_1 - y_1|, \dots, |x_d - y_d|\}$$

  - The limit of $\boldsymbol{L_p}$ as p goes to infinity.

Example of Distances

L_2-norm:

$$dist(x, y) = \sqrt{4^2 + 3^2} = 5$$



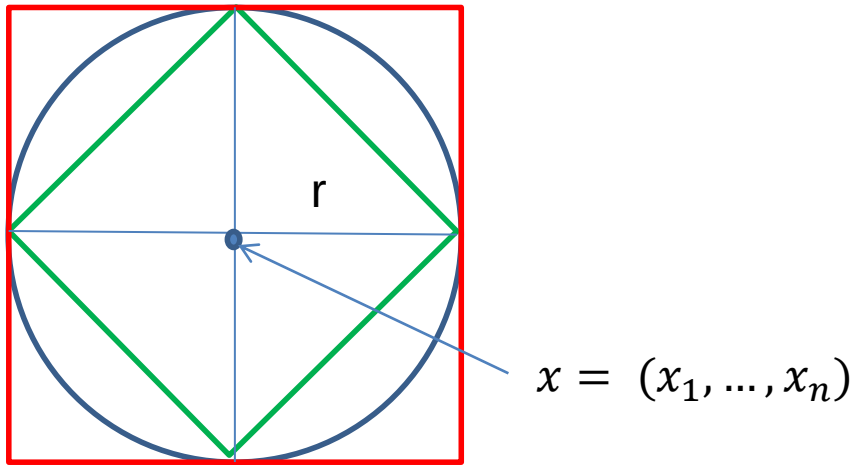
L_1-norm:

$$dist(x, y) = 4 + 3 = 7$$

L_∞-norm:

$$dist(x, y) = \max\{3,4\} = 4$$

Example



Green: All points y at distance $L_1(x, y) = r$ from point x

Blue: All points y at distance $L_2(x, y) = r$ from point x

Red: All points y at distance $L_\infty(x, y) = r$ from point x

L_p distances for sets

- We can apply all the L_p distances to the cases of sets of attributes, with or without counts, if we represent the sets as vectors
  - E.g., a transaction is a 0/1 vector
  - E.g., a document is a vector of counts.

Similarities into distances

- Jaccard distance:

$$JDist(X, Y) = 1 - JSim(X, Y)$$

- Jaccard Distance is a metric

- Cosine distance:

$$Dist(X, Y) = 1 - \cos(X, Y)$$

- Cosine distance is a metric

Hamming Distance

- Hamming distance is the number of positions in which bit-vectors differ.
  - Example:
    - p_1 = 10101
    - p_2 = 10011.
    - $d(p_1, p_2) = 2$ because the bit-vectors differ in the 3rd and 4th positions.
    - The L_1 norm for the binary vectors

- Hamming distance between two vectors of categorical attributes is the number of positions in which they differ.
  - Example:
    - x = (married, low income, cheat)
    - y = (single, low income, not cheat)
    - $d(x, y) = 2$

Why Hamming Distance Is a Distance Metric

- d(x,x) = 0 since no positions differ.
- d(x,y) = d(y,x) by symmetry of “different from.”
- d(x,y) ≥ 0 since strings cannot differ in a negative number of positions.
- Triangle inequality: changing *x* to *z* and then to *y* is one way to change *x* to *y*.

- For binary vectors if follows from the fact that L_1 norm is a metric

Distance between strings

- How do we define similarity between strings?

weird	wierd
intelligent	unintelligent
Athena	Athina

- Important for recognizing and correcting typing errors and analyzing DNA sequences.

Edit Distance for strings

- The edit distance of two strings is the number of inserts and deletes of characters needed to turn one into the other.
- Example: x = abcde ; y = bcduve.
  - Turn *x* into *y* by deleting a, then inserting u and v after d.
  - Edit distance = 3.
- Minimum number of operations can be computed using dynamic programming
- Common distance measure for comparing DNA sequences

Why Edit Distance Is a Distance Metric

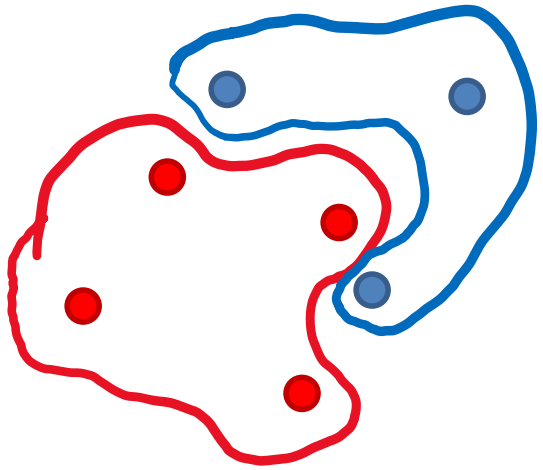
- d(x,x) = 0 because 0 edits suffice.
- d(x,y) = d(y,x) because insert/delete are inverses of each other.
- d(x,y) ≥ 0: no notion of negative edits.
- Triangle inequality: changing *x* to *z* and then to *y* is one way to change *x* to *y*. The minimum is no more than that

Variant Edit Distances

- Allow insert, delete, and mutate.
  - Change one character into another.
- Minimum number of inserts, deletes, and mutates also forms a distance measure.

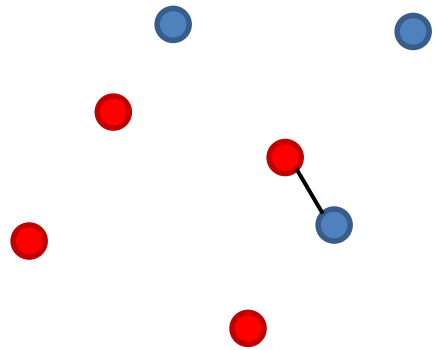
- Same for any set of operations on strings.
  - Example: substring reversal or block transposition is used for DNA sequences
  - Example: character transposition is used for spelling

Distance between sets of points



How do we measure the distance between the two sets?

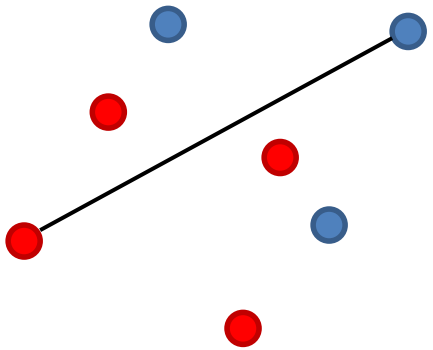
Distance between sets of points



How do we measure the distance between the two sets?

Minimum distance over all pairs

Distance between sets of points

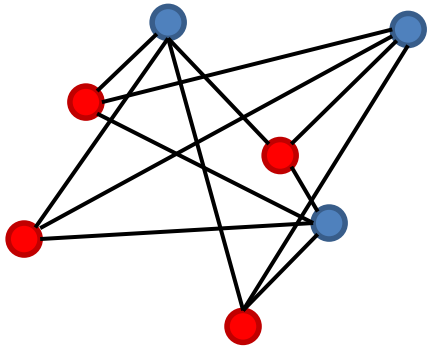


How do we measure the distance between the two sets?

Minimum distance over all pairs

Maximum distance over all pairs

Distance between sets of points



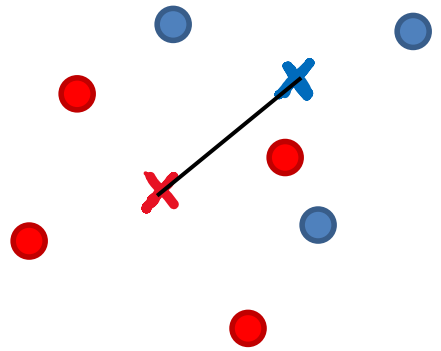
How do we measure the distance between the two sets?

Minimum distance over all pairs

Maximum distance over all pairs

Average distance over all pairs

Distance between sets of points



How do we measure the distance between the two sets?

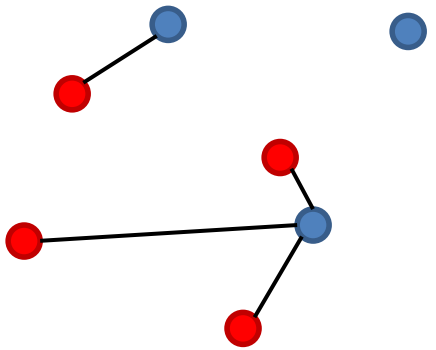
Minimum distance over all pairs

Maximum distance over all pairs

Average distance over all pairs

Distance between averages

Distance between sets of points



How do we measure the distance between the two sets?

Minimum distance over all pairs

Maximum distance over all pairs

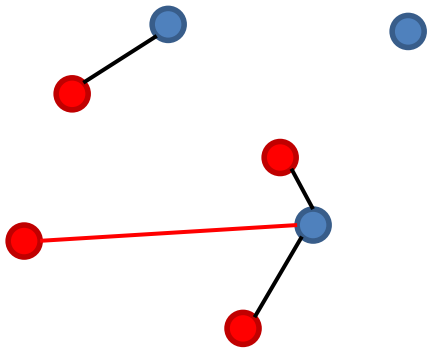
Average distance over all pairs

Distance between averages

Hausdorff distance:

- For each red point x compute the distance to the closest Blue point: $d(x, Blue) = \min_{y \in Blue} d(x, y)$

Distance between sets of points



How do we measure the distance between the two sets?

Minimum distance over all pairs

Maximum distance over all pairs

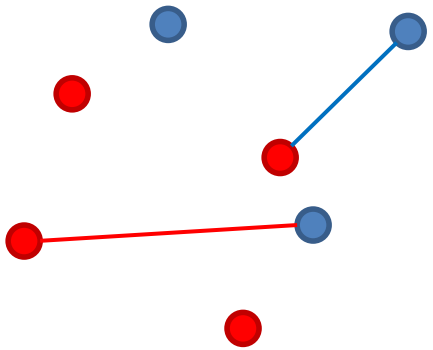
Average distance over all pairs

Distance between averages

Hausdorff distance:

- For each red point x compute the distance to the closest Blue point: $d(x, Blue) = \min_{y \in Blue} d(x, y)$
- Find the maximum: this is the distance from Red to Blue: $d(Red, Blue) = \max_{x \in Red} d(x, Blue)$

Distance between sets of points



How do we measure the distance between the two sets?

Minimum distance over all pairs

Maximum distance over all pairs

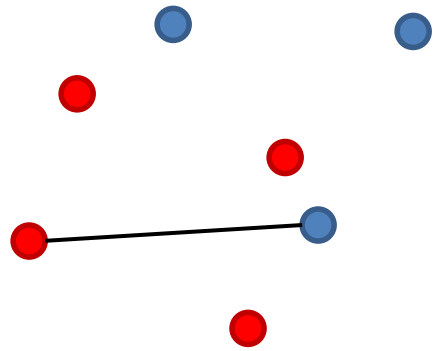
Average distance over all pairs

Distance between averages

Hausdorff distance:

- For each red point x compute the distance to the closest Blue point: $d(x, Blue) = \min_{y \in Blue} d(x, y)$
- Find the maximum: this is the distance from Red to Blue: $d(Red, Blue) = \max_{x \in Red} d(x, Blue)$
- Compute the $d(Blue, Red)$

Distance between sets of points



How do we measure the distance between the two sets?

Minimum distance over all pairs

Maximum distance over all pairs

Average distance over all pairs

Distance between averages

Hausdorff distance:

- For each red point x compute the distance to the closest Blue point: $d(x, Blue) = \min_{y \in Blue} d(x, y)$
- Find the maximum: this is the distance from Red to Blue: $d(Red, Blue) = \max_{x \in Red} d(x, Blue)$
- Compute the $d(Blue, Red)$
- Take the maximum of the two

$$d_H(Red, Blue) = \max\left\{ \max_{x \in Red} \min_{y \in Blue} d(x, y), \max_{x \in Red} \min_{y \in Blue} d(x, y) \right\}$$

Distances between distributions

- Sometimes data can be represented as a distribution (e.g., a document is a distribution over the words)

document	Apple	Microsoft	Obama	Election
D1	0.35	0.5	0.1	0.05
D2	0.4	0.4	0.1	0.1
D3	0.05	0.05	0.6	0.3

- How do we measure distance between distributions?

Variational distance

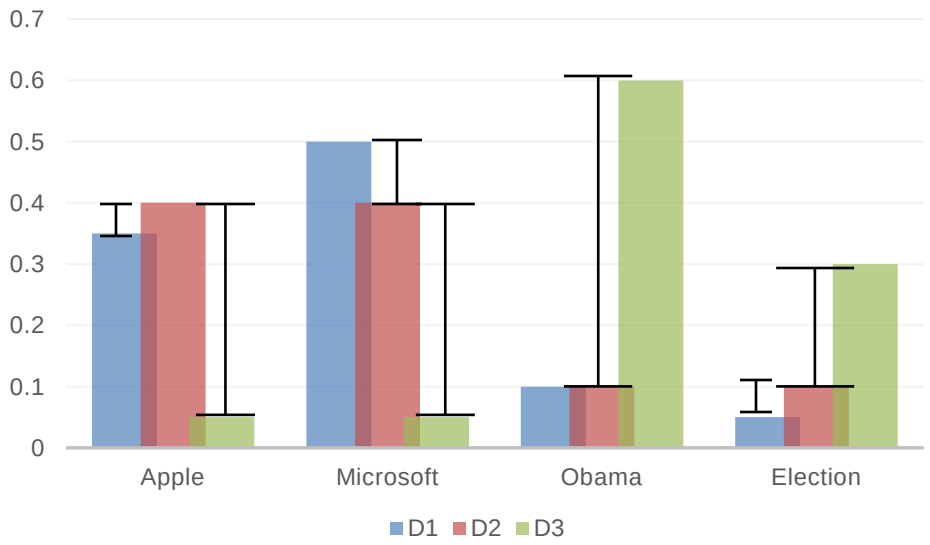
- Variational distance: The L_1 distance between the distribution vectors

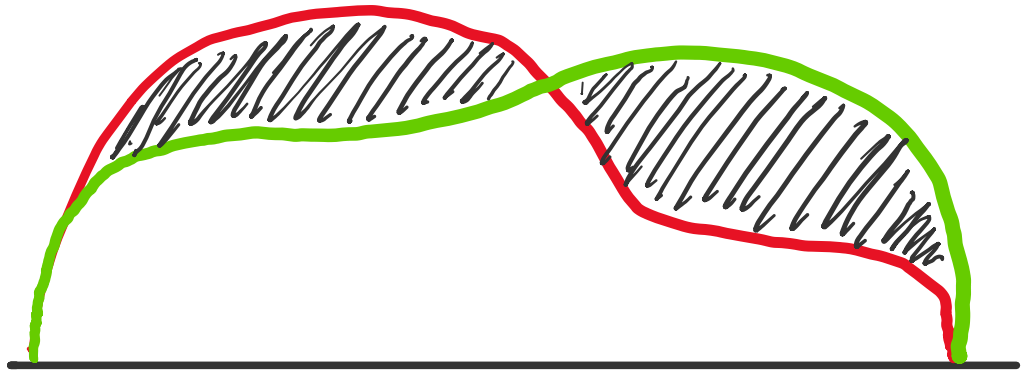
document	Apple	Microsoft	Obama	Election
D1	0.35	0.5	0.1	0.05
D2	0.4	0.4	0.1	0.1
D3	0.05	0.05	0.6	0.3

Dist(D1,D2) = 0.05+0.1+0.05 = 0.2

Dist(D2,D3) = 0.35+0.35+0.5+ 0.2 = 1.4

Dist(D1,D3) = 0.3+0.45+0.5+ 0.25 = 1.5





Information theoretic distances

document	Apple	Microsoft	Obama	Election
D1	0.35	0.5	0.1	0.05
D2	0.4	0.4	0.1	0.1
D3	0.05	0.05	0.6	0.3

- KL-divergence (Kullback-Leibler) for distributions P,Q

$$D_{KL}(P\|Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

- KL-divergence is asymmetric. We can make it symmetric by taking the average of both sides

$$\frac{1}{2}\left(D_{KL}(P\|Q) + D_{KL}(Q\|P)\right)$$

- JS-divergence (Jensen-Shannon)

$$JS(P,Q) = \frac{1}{2} D_{KL}(P\|M) + \frac{1}{2} D_{KL}(Q\|M)$$

$$M = \frac{1}{2}(P+Q)$$

Average distribution

Ranking distances

- The input in this case is two rankings/orderings of the same n items. For example:

$$R_1 = \langle x, y, z, w \rangle$$
$$R_2 = \langle y, w, z, x \rangle$$

  - How do we define distance in this case?
- Kendal's tau: Number of pairs of items that are in different order:

$$|\{(x, y), (x, z), (x, w), (z, w)\}| = 4$$

  - Defines a metric.
  - Maximum: $\frac{n(n-1)}{2}$ when rankings are reversed.
- Spearman rank distance: L_1 distance between the ranks
  - $SR(R_1, R_2) = |1 - 4| + |2 - 1| + |3 - 3| + |4 - 2| = 6$

	x	y	z	w
R_1	1	2	3	4
R_2	4	1	3	2

Why is similarity important?

- We saw many definitions of similarity and distance
- How do we make use of similarity in practice?
- What issues do we have to deal with?

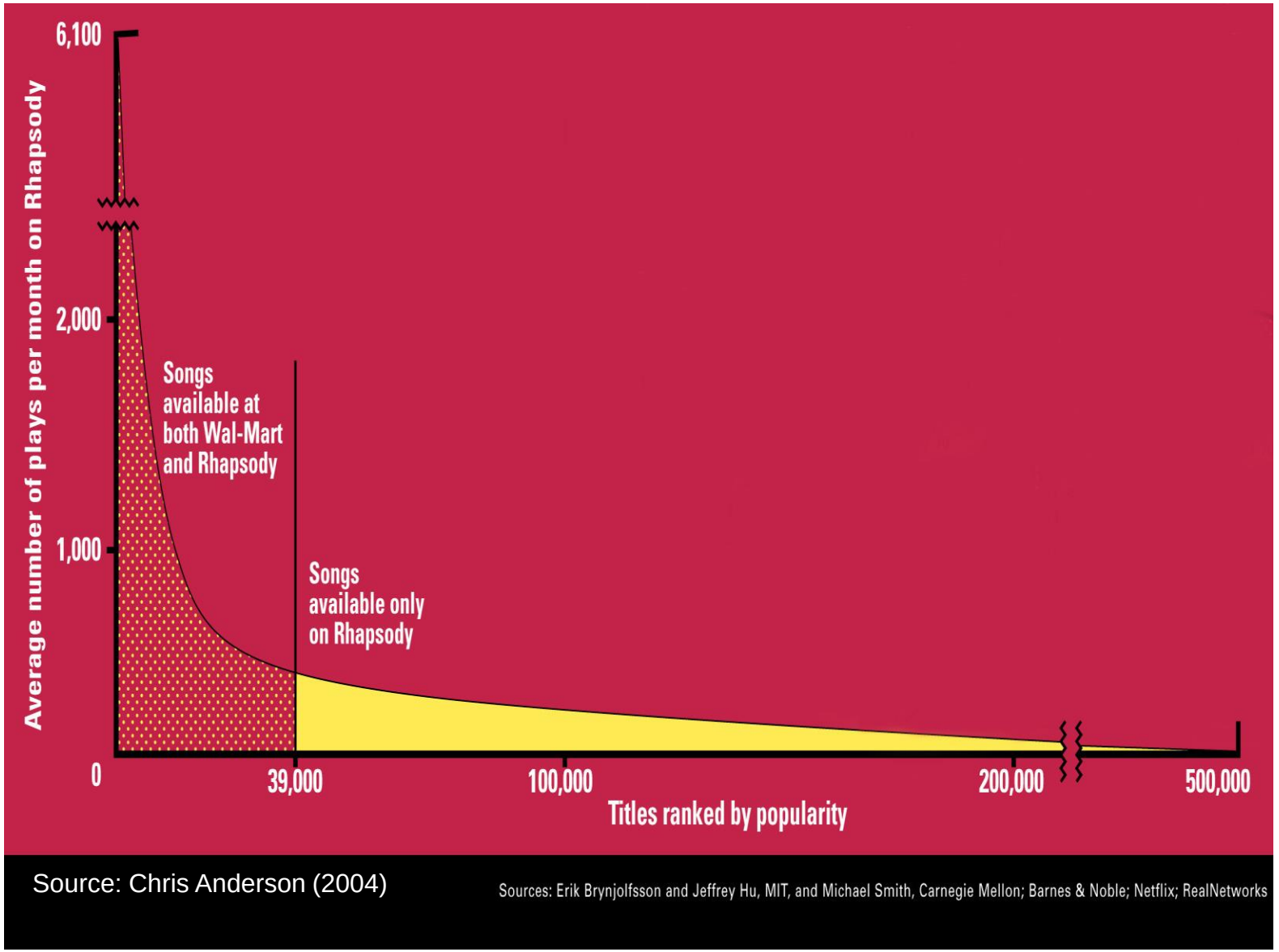
APPLICATIONS OF SIMILARITY: RECOMMENDATION SYSTEMS

An important problem

- Recommendation systems
  - When a user buys an item (initially books) we want to recommend other items that the user may like
  - When a user rates a movie, we want to recommend movies that the user may like
  - When a user likes a song, we want to recommend other songs that they may like

- A big success story for data mining
- Exploits the long tail
  - How Into Thin Air made Touching the Void popular

The Long Tail



Source: Chris Anderson (2004)

Utility (Preference) Matrix

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Rows: Users
Columns: Movies (in general Items)
Values: The rating of the user for the movie

How can we fill the empty entries of the matrix?

Recommendation Systems

- Content-based:
  - Represent the items into a feature space and
  - Recommend items to customer C similar to previous items rated highly by C

- Examples
  - Movie recommendations:
    - recommend movies with same actor(s), director, genre, …
  - Documents: websites, blogs, news:
    - recommend other documents with “similar” content

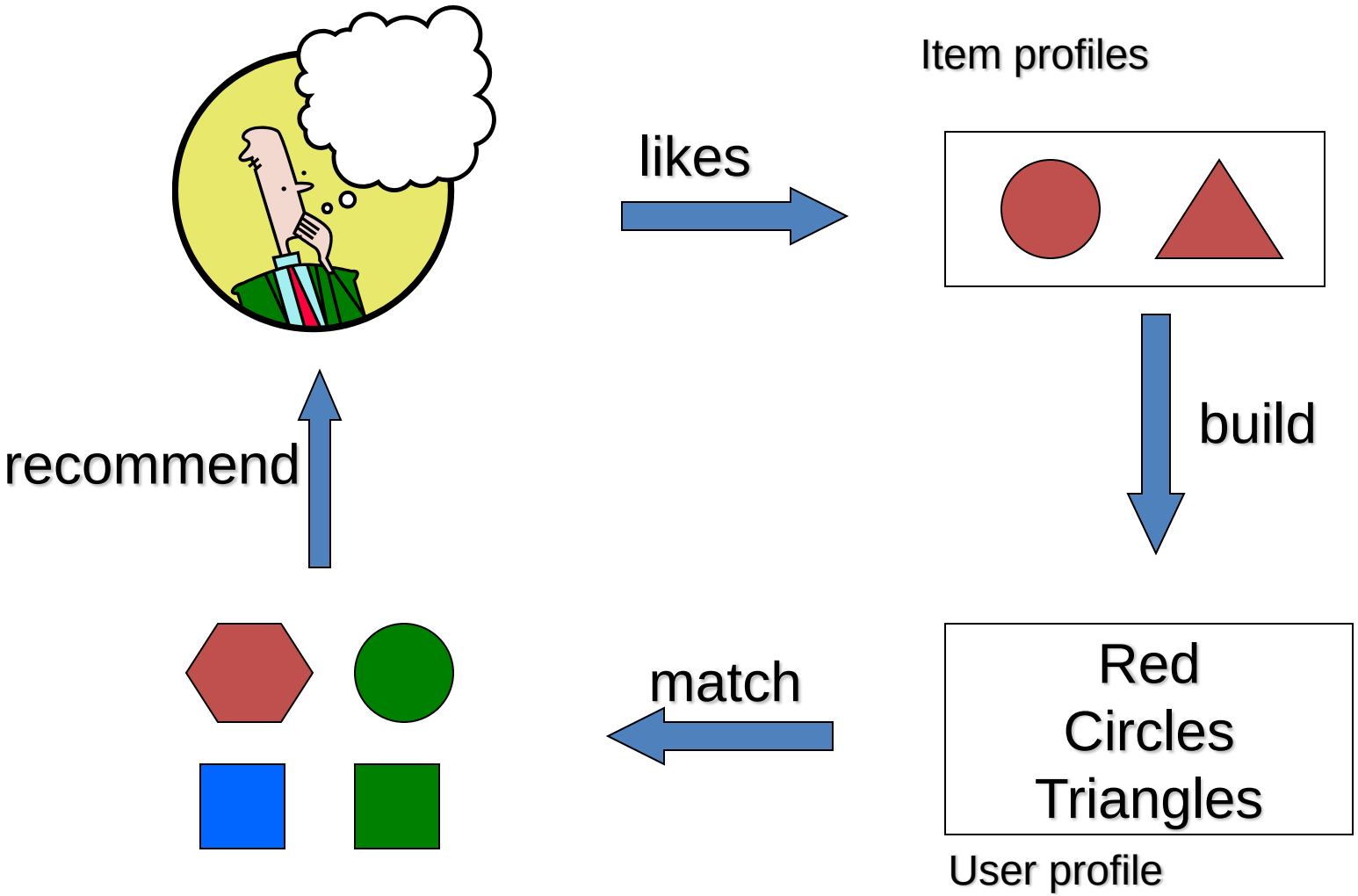
Content-based prediction

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Someone who likes one of the Harry Potter (or Star Wars) movies is likely to like the rest

- Same actors, similar story, same genre

Intuition


Item profiles
likes
build
recommend
match
Red
Circles
Triangles
User profile

Approach

- Map items into a feature space:
  - For movies:
    - Actors, directors, genre, rating, year,…
    - Challenge: make all features compatible.
  - For documents?

- To compare items with users we need to map users to the same feature space. How?
  - Take all the movies that the user has seen and take the average vector
    - Other aggregation functions are also possible.

- Recommend to user C the most similar item i computing similarity in the common feature space
  - Distributional distance measures also work well.

Limitations of content-based approach

- Finding the appropriate features
  - e.g., images, movies, music
  - Embeddings and deep learning can help
- Overspecialization
  - Never recommends items outside user's content profile
  - People might have multiple interests
- Recommendations for new users: cold-start problem
  - How to build a profile?

Collaborative filtering

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Two users are similar if they rate the same items in a similar way

Recommend to user C, the items liked by many of the most similar users.

User Similarity

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Which pair of users do you consider as the most similar?

What is the right definition of similarity?

User Similarity

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	1			1	1		
B	1	1	1				
C				1	1	1	
D		1					1

Jaccard Similarity: users are sets of movies

Disregards the ratings.

$$Jsim(A, B) = 1/5$$
$$Jsim(A, C) = 1/2$$
$$Jsim(B, D) = 1/4$$

User Similarity

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Cosine Similarity:

Assumes zero entries are negatives, non-zeros are positive:

$$Cos(A, B) = 0.38$$
$$Cos(A, C) = 0.32$$

User Similarity

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	2/3			5/3	-7/3		
B	1/3	1/3	-2/3				
C				-5/3	1/3	4/3	
D		0					0

Normalized Cosine Similarity:

- Subtract the mean rating per user (without the zeros) and then compute Cosine (correlation coefficient)

$$Corr(A, B) = 0.0920$$
$$Corr(A, C) = -0.559$$

User-based Collaborative Filtering

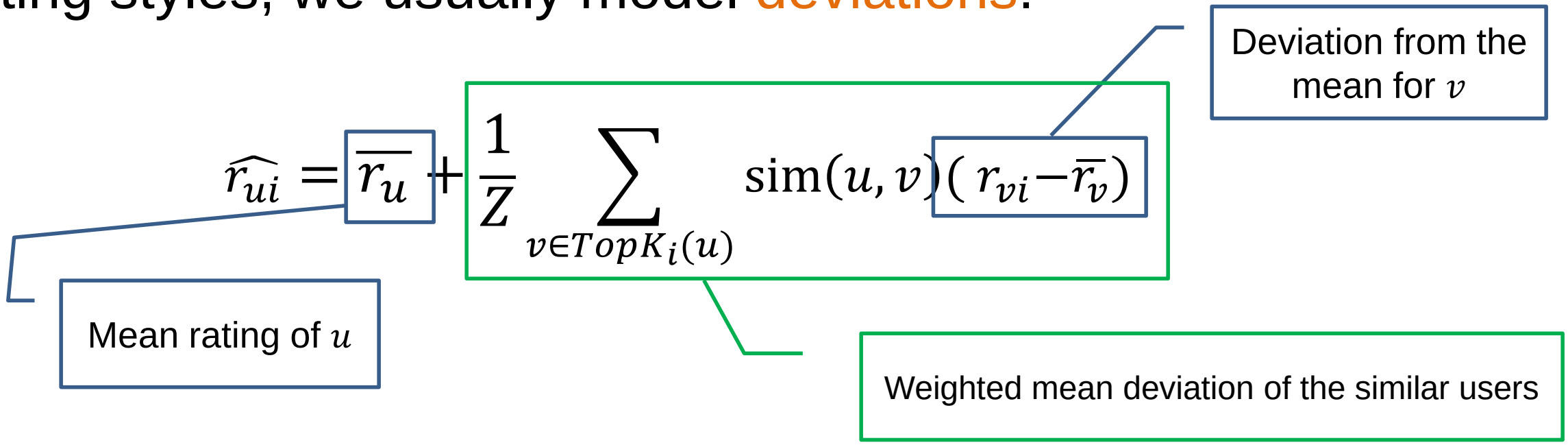
- To estimate the rating for a user-item pair (u, i):
  - Find the set $TopK_i(u)$ of the K most similar users to u who have rated item i.
  - Estimate u's ratings for item i, by aggregating the ratings of users in $TopK$:

$$\widehat{r_{ui}} = \frac{1}{Z} \sum_{v \in TopK_i(u)} \text{sim}(u, v) r_{vi}$$

$$Z = \sum_{v \in TopK_i(u)} \text{sim}(u, v)$$

User-based Collaborative Filtering

- To account for the fact that different users have different rating styles, we usually model deviations:



Note: Similarity can be computed with or without centering (subtracting the mean)

Item-based Collaborative Filtering

- There is a duality in the use of the preference matrix. In the same way we define user similarity (rows), we can also define item similarity (columns)
  - Intuition: Two items are similar if they are rated in the same way by many users.
  - Better defined similarity since it captures the notion of genre of an item: Items rated by the same users define a genre
    - Better since items usually (but not always) have a single genre, while users may have multiple interests.

Item-based Collaborative Filtering

- To estimate the rating for a user-item pair (u,i):
  - Find the set $TopK_u(i)$ of most similar items to item i that have been rated by user u.
  - Aggregate their ratings to predict the rating for item i.

$$\widehat{r_{ui}} = \frac{1}{Z} \sum_{j \in TopK_u(i)} \text{sim}(i,j) r_{uj}$$

$$Z = \sum_{j \in TopK_u(i)} \text{sim}(i,j)$$

Item-based Collaborative Filtering

- Again, we want to model deviations in rating behavior.
- Approach 1: Do exactly the same as for users.
  - Normalize the columns and compute Pearson correlation.

$$\widehat{r_{ui}} = \overline{r_i} + \frac{1}{Z} \sum_{j \in TopK_u(i)} \text{sim}(i,j)(r_{uj} - \overline{r_j})$$

  Assumes that different items are rated differently

- Approach 2: Normalize again the rows of the matrix

$$\widehat{r_{ui}} = \overline{r_u} + \frac{1}{Z} \sum_{j \in TopK_u(i)} \text{sim}(i,j)(r_{uj} - \overline{r_u})$$

  - Note that we add to the mean rating of the user

Implementation details

- When removing the mean rating make sure to take into account only the rated (non-zero) entries
- What if we cannot find k similar users/items?
  - Use as many as you can find
- For efficiency, when looking for the k most similar users (items) we can take the k most similar users (items) regardless if they have rated the item i
  - We assume missing ratings are zero.
  - More efficient but not a good idea.
- Pearson correlation can be negative, which complicates the formula and the normalization
  - We usually assume that the k most similar entries do not have negative similarities
  - If we have negative similarities, we should take absolute values when computing normalizing factor Z

Evaluation

- Split the data into train and test set (e.g., 80%,20%)
  - The train set will be used to estimate the similarities or the user and item profiles
  - The test will be used to evaluate the accuracy of the predictions
- Data usually means the ratings $r(u, i)$
  - We randomly hide 20% (or more, or less) of the ratings and we try to predict them
- We could do the split in different ways
  - E.g. randomly select a subset of users to predict for (and delete more ratings)
  - Split based on time of the ratings: Keep all the ratings up to a certain time, and predict the ones in a later time

Evaluation

- Metrics: how do we evaluate the prediction?

- Evaluate our ability to predict the numeric rating of the item
  - The output of the algorithm is a numeric value for each item

- Root Mean Square Error (RMSE) :

$$RMSE = \sqrt{\frac{1}{n}\sum_{i,j}\left(\widehat{r_{ij}} - r_{ij}\right)^2}$$

Evaluation

- Evaluate our ability to predict binary (action/no action) event:
  - The output of the algorithm is also a yes/no value

- Precision/Recall :
  - Precision = fraction of predicted positive actions that were correct

$$Precision = \frac{Correct\ Positive\ Decisions}{Positive\ Decisions}$$

  - Recall = fraction of positive actions that were predicted correctly

$$Recall = \frac{Correct\ Positive\ Decisions}{Positive\ Items}$$

Evaluation

- Evaluate our ability to rank the items correctly:
  - The output of the algorithm is a ranking of the items, we want the most relevant items to be ranked higher
- AUC (Area Under the ROC Curve) [binary data]
  - The fraction of (relevant, non-relevant) pairs where the relevant item is ranked higher than the non-relevant item.
- Precision/Recall @ k [binary data]:
  - Look at the top-k recommendations and compute the precision and recall
- NDCG@k
  - [binary data]:

$$\frac{1}{Z}\sum_{i=1}^{k}\frac{\delta(item\ i\ is\ relevant)}{\log(i+1)}, \mathrm{Z} = \sum_{i=1}^{k}\frac{1}{\log(i+1)}$$

  - [rated data]:

$$\frac{1}{Z}\sum_{i=1}^{k}\frac{\exp(r_i)}{\log(i+1)}, \mathrm{Z} = \sum_{i=1}^{k}\frac{\exp(r_i^*)}{\log(i+1)}, \quad r_i = true\ rating\ of\ i^{th} item\ in\ the\ ranking$$
$$r_i^* = rating\ of\ i^{th} best\ item$$

- Kendal' tau metric [rated data]:
  - The fraction of pairs of items for a user that are ordered correctly (or incorrectly)

Pros and cons of collaborative filtering

- Works for any kind of items
  - No feature selection needed
- Cold-start problem: New user, or new item
- Sparsity of rating matrix
  - Cluster-based smoothing?

The Netflix Challenge

- 1M prize to improve the prediction accuracy by 10%