Content-based recommendation systems
(based on chapter 9 of Mining of Massive Datasets, a book by Rajaraman, Leskovec, and Ullman’s book)

Fernando Lobo

Data mining
Content-based Recommendation Systems

- Focus on properties of items.
- Similarity of items is determined by measuring the similarity in their properties.
Item profiles

- Need to construct a profile for each item.

- A profile is a collection of important characteristics about the item.

- Example for item = movie. Profile can be:
  - set of actors
  - director
  - year the movie was made
  - genre
Discovering features

- Features can be obvious and immediately available (as in the movie example).

- But many times they are not. Examples:
  - document collections
  - images
Discovering features of documents

- Documents can be news articles, blog posts, webpages, research papers, etc.

- Identify a set of words that characterize the topic of a document.

- Need a way to find the importance of a word in a document.

- We can pick the $n$ most important words of that document as the set of words that characterize the document.
Finding the importance of a word in a document

Common approach:

- Remove stop words — the most common words of a language that tend to say nothing about the topic of a document (examples from English: the, and, of, but, …)

- For the remaining words compute their TF.IDF score

- TF.IDF stands for *Term Frequency times Inverse Document Frequency*
First compute the *Term Frequency* (TF):

- Given a collection of \( N \) documents.

- Let \( f_{ij} = \) number of times word \( i \) appears in document \( j \).

- Then the term (word) frequency \( TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \)

- Term frequency is \( f_{ij} \) normalized by dividing it by the maximum number of occurrences of any term in the same document (excluding stop words)
TF.IDF score

Then compute the *Inverse Document Frequency* (IDF):

- IDF for a term (word) is defined as follows. Suppose word $i$ appears in $n_i$ of the $N$ documents.

- The $IDF_i = \log(N/n_i)$

- TF.IDF for term $i$ in document $j = TF_{ij} \times IDF_i$
TF.IDF score example

- Suppose we have $2^{20} = 1048576$ documents. Suppose word $w$ appears in $2^{10} = 1024$ of them.

- The $IDF_w = \log(2^{20}/2^{10}) = 10$

- Suppose that in a document $k$, word $w$ appears one time and the maximum number of occurrences of any word in this document is 20. Then,
  - $TF_{wk} = 1/20$.
  - TF.IDF for word $w$ in document $k$ is $1/20 \times 10 = 1/2$. 

Finding similar items

- Find a similar item by using a distance measure.

- For documents, two popular distance measures are:
  - Jaccard distance between sets of words
  - Cosine distance between sets, treated as vectors
Jaccard Similarity and Jaccard Distance of Sets

- The Jaccard similarity (SIM) of sets $S$ and $T$ is 
  \[ \frac{|S \cap T|}{|S \cup T|} \]

- Example: $\text{SIM}(S, T) = \frac{3}{8}$

- Jaccard distance of $S$ and $T$ is $1 - \text{SIM}(S, T)$
Cosine Distance of sets

- Compute the dot product of the sets (treated as vectors) and divide by their Euclidean distance from the origin.

- Example: \( x = [1, 2, -1], \ y = [2, 1, 1] \)

  Dot product \( x \cdot y = 1 \cdot 2 + 2 \cdot 1 + (-1) \cdot 1 = 3 \)

  Euclidean distance of \( x \) to the origin
  
  \[
  = \sqrt{1^2 + 2^2 + (-1)^2} = \sqrt{6}
  \]
  
  (same thing for \( y \))

  Cosine distance between \( x \) and \( y = \frac{3}{\sqrt{6} \sqrt{6}} = 1/2 \)
Sets of words as bit vectors

- Think of a set of words as a bit vector, one bit position for each possible word.
- Position has 1 if the word is in the set, and has 0 if not.
- Only need to take care of words that exist in both documents. (0’s don’t affect the calculations)
User profiles

- Weighted average of rated item profiles

- Example: items = movies represented by boolean profiles.

Utility matrix has a 1 if the user has seen a movie and is blank otherwise

If 20% of the movies that user $U$ likes have Julia Roberts as one of the actors, then user profile for $U$ will have 0.2 in the component for Julia Roberts.
User profiles

- If utility matrix is not boolean, e.g., ratings 1–5, then weight the vectors by the utility value and normalize by subtracting the average value for a user.

- This way we get negative weights for items with below average ratings, and positive weights for items with above average ratings.
Recommending items to users based on content

- Compute cosine distance between user’s and item’s vectors

- Movie example:

  - highest recommendations (lowest cosine distance) belong to movies with lots of actors that appear in many of the movies the user likes.