CS6301: R For Data Scientists

LECTURE 25: RECOMMENDATION ENGINES

Recommendation Engine

A tool that will suggest items which are considered *statistically relevant* for the user

You have experienced these many times – Facebook recommending friends, shopping online, Tinder, …

- In some sense, all dating sites!

There are two basic kinds:

- Content Based Recommendation: Identify key aspects of an item a customer is interested in, find other items that have the same (or similar) characteristics
- Collaborative Filtering: Look at all similar users that purchased the item the user is looking at, create a list of items purchased by those users, and present the most commonly occurring item

recommenderlab

Seems to be the popular choice for doing recommendation engines in R

The package uses a ratingMatrix data structure for rating data

Many standard matrix operations can be done on these objects

Common notation: UBCF (User Based Collaborative Filtering), IBCF (Item Based Collaborative Filtering)

Most of the work involves processing data

Collaborative Filtering

We will try to build a recommendation engine that will suggest movies to a user

We will do this by trying to predict what rating a user would give to a movie that the user has not seen yet

- Then we can just recommend the top 10 choices …

Given a unviewed movie our user has not seen yet …

- UBCF: Find users that give similar ratings to our user, use their ratings to estimate ratings for the unviewed movie.
- IBCF: Find movies that are similar to the new movie (in terms of ratings), use those movies to estimate a rating for the unviewed movie

Either way, we need a matrix of movies versus users to capture the ratings – and a way to measure "similarity"

Measuring Similarity

How do we measure if two users are “similar”?

Note we can use the distance functions we saw earlier (*dist(), daisy()*)

In addition to the metrics we’ve seen, there are also Cosine and Jaccard

Cosine: Think of vectors in space, and compute the cosine of the angle between them

$$\cos(\theta) = \frac{\vec{v}_1 \cdot \vec{v}_2}{||\vec{v}_1||||\vec{v}_2||}$$

If the vectors are “going in the same direction”, this should be close to ‘1’

If the vectors are orthogonal, it is ‘0’

If the vectors are going in opposite directions, this is ‘-1’

Measuring Similarity(2)

If the data is centered, the cosine similarity is equivalent to the Pearson similarity, which is the usual correlation coefficient

Jaccard: An index that measures similarity by comparing the number of common elements to the total number of elements

Example: A = {1,2,3}, B = {2,3,4,5}. Then

$$J(A,B) = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} = \frac{2}{3+4-2} = .4$$

If the sets are identical, this is ‘1’, while if they have noting in common, this is ‘0’

Content Based Filtering

Here we need more information about the items we are searching for an also what the user is looking for

Basic idea – get descriptive profile for all items, get descriptive profile for the item the user is interested in, and then try to match user to items that best fit

We may have a particular item a user is currently looking at (simplest case)

We may also need to create a user profile based upon items the user has viewed/purchased/reviewed

Does not seem to be a good package for this in R …

Usually involves some careful analysis of the available data

Links

https://muffynomster.wordpress.com/2015/06/07/building-a-movie-recommendation-engine-with-r/

https://www.r-bloggers.com/recommender-systems-101-a-step-by-step-practical-example-in-r/

https://ashokharnal.wordpress.com/2014/12/18/using-recommenderlab-for-predicting-ratings-for-movielens-data/