Project 4 Code

Charley Ferrari

July 11, 2016

```
library(dplyr)
library(reshape2)
library(plyr)

setwd('/Users/Charley/Downloads/cuny/IS 643 Recommender Systems/Week 4/hetrec2011-delicious-2k')
```

```
bookmark_tags <- read.delim('bookmark_tags.dat', header=TRUE)

bookmarks <- read.delim('bookmarks.dat', sep='\t', header=TRUE)

tags <- read.delim('tags.dat', sep='\t', header=TRUE)

user_taggedbookmarks_timestamps <- read.delim('user_taggedbookmarks-timestamps.dat',
                                              sep='\t', header=TRUE)
```

Adding context to a recommendation engine

This dataset comes from a collaborative bookmark tagging platform called del.icio.ous. In this platform, users are given the ability to store bookmarks and tag them. Our main data matrix, user_taggedbookmarks, includes a collection of timestamped tags. For this system, I will be treating the tags as extra context, and leaving timestamps out for the moment.

I tried to apply the models put forth in chapter 10 of Statistical Methods for Recommender Systems, but believe that a simpler method will be possible for this dataset, and will only be taking the core concept of thinking of this recommendation system as a three dimensional problem.

Rather than just dealing with users and items, we will consider tags to be our third dimension. In two dimensional versions of this problem, we would compute the cosine similarity between users and/or items. In this case, we are computing a version of the cosine similarity (based on the Frobenius norm) between vectors, and can judge similarities between users, bookmarks, or tags.

The objective of this recommendation engine will be to compute the probability of a user tagging a bookmark with a particular tag. By computing distance in this way, we should be ensuring that both the tag and the bookmark will be relevent to the user.

I couldn't find any great solutions to dealing with sparse three dimensional arrays, but because we are dealing with binary data (0's for no interaction or 1 for a successful tag), the calculations should be relatively straight forward. The similarity measure I'm using is shown below:

$$\frac{\sum_{i=1}^{n_i}\sum_{j=1}^{n_j} A_{ij}B_{ij}}{\sqrt{\sum_{i=1}^{n_i}\sum_{j=1}^{n_j} A_{ij}}\sqrt{\sum_{i=1}^{n_i}\sum_{j=1}^{n_j} B_{ij}}}$$

In our problem, for computing two users similarities, this simplifies. Lets consider a to be the number of tags by user A, b to be the number of tags by user B, and s to be the number of tag-bookmark pairs that are shared. We end up with

$$\frac{s}{a \times b}$$

So, I will write my similarity functions to take advantage of this directly from the data frame. Here is what this function would look like for a user:

```
userSimCalc <- function(u1, u2){
  slice <- user_taggedbookmarks_timestamps %>%
    filter(userID %in% c(u1, u2))
  numdistinct <- nrow(
    slice %>% distinct(bookmarkID, tagID)
  )
  s <- nrow(slice) - numdistinct
  u1nr <- nrow(
    slice %>% filter(userID == u1)
  )
  u2nr <- nrow(
    slice %>% filter(userID == u2)
  )
  return(s / (u1nr*u2nr))
}
```