Project 3

Max Wagner

June 25, 2016

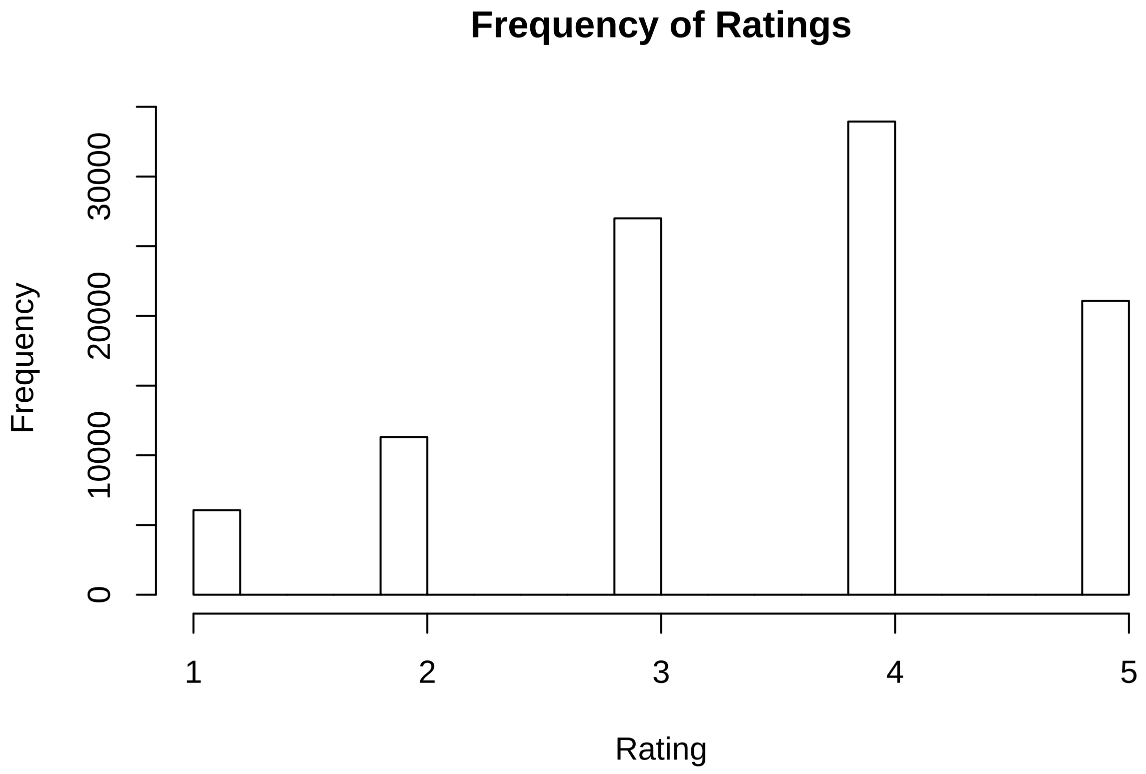
Data

I've decided to switch from the limited data set I used for the first two projects to the more diverse MovieLense set. This allows for a better representation of what the methods are actually doing, and differences in efficiency can actually be seen.

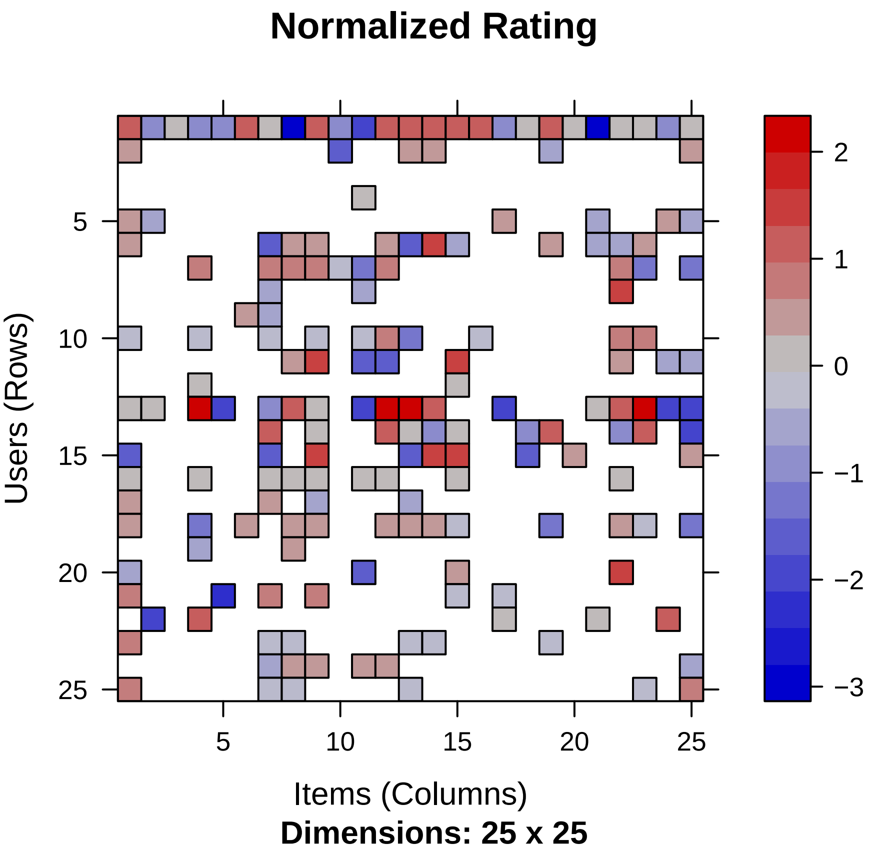
I'll begin by just loading the data, and visualize the ratings in the set. We can also see it as a normalized set in the second plot. Importantly, this second graph shows the high amount of NA's in the data set.

```r
library("recommenderlab")
data(MovieLense)

ml.matrix <- as(MovieLense, "matrix")
hist(ml.matrix, main = "Frequency of Ratings", xlab = "Rating")
```



```
ml.norm <- normalize(MovieLense[c(1:25), c(1:25)])
plot(image(ml.norm, main = "Normalized Rating"))
```

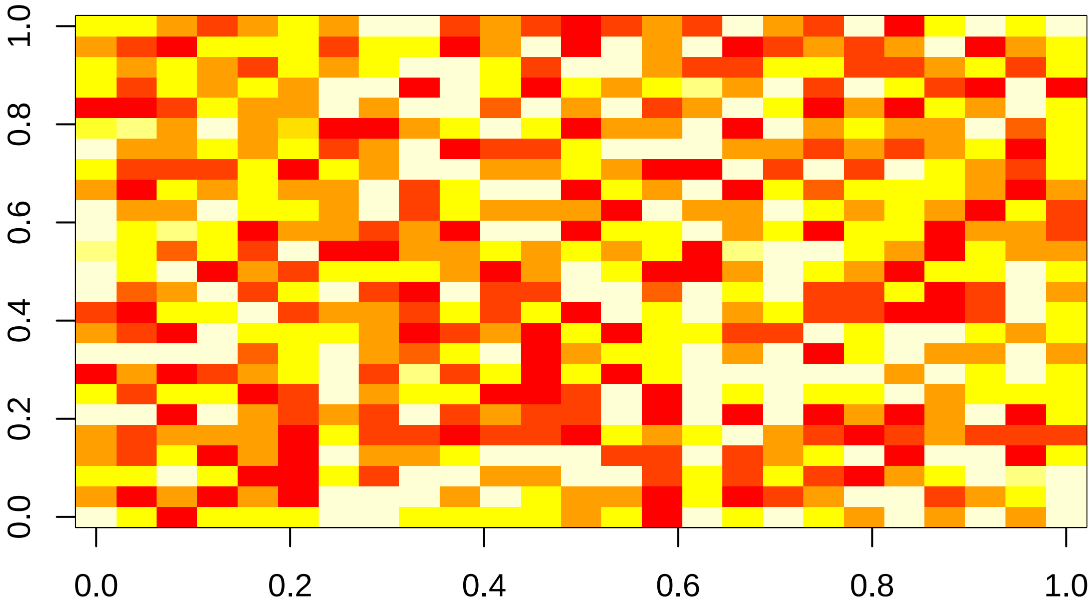


NMF Method

The first method to try is Non-Zero Matrix Factorization. To do this, the `NMF` package will be used. The NA's must be removed for the method to work. I'll just make all NA values in the MovieLense set 0. The graph shows a much different story than the orignal values in the matrix from above. There are very few extreme values because of the removal of NA from the color scale, and no distict patterns which would infer that the method did not work correctly.

```
library("NMF")

ml.matrix[is.na(ml.matrix)] <- sample(1:5, length(ml.matrix[is.na(ml.matrix)]),replace = T)
ml.mf <- nmf(ml.matrix[c(1:25), c(1:25)],25)
image(fitted(ml.mf))
```



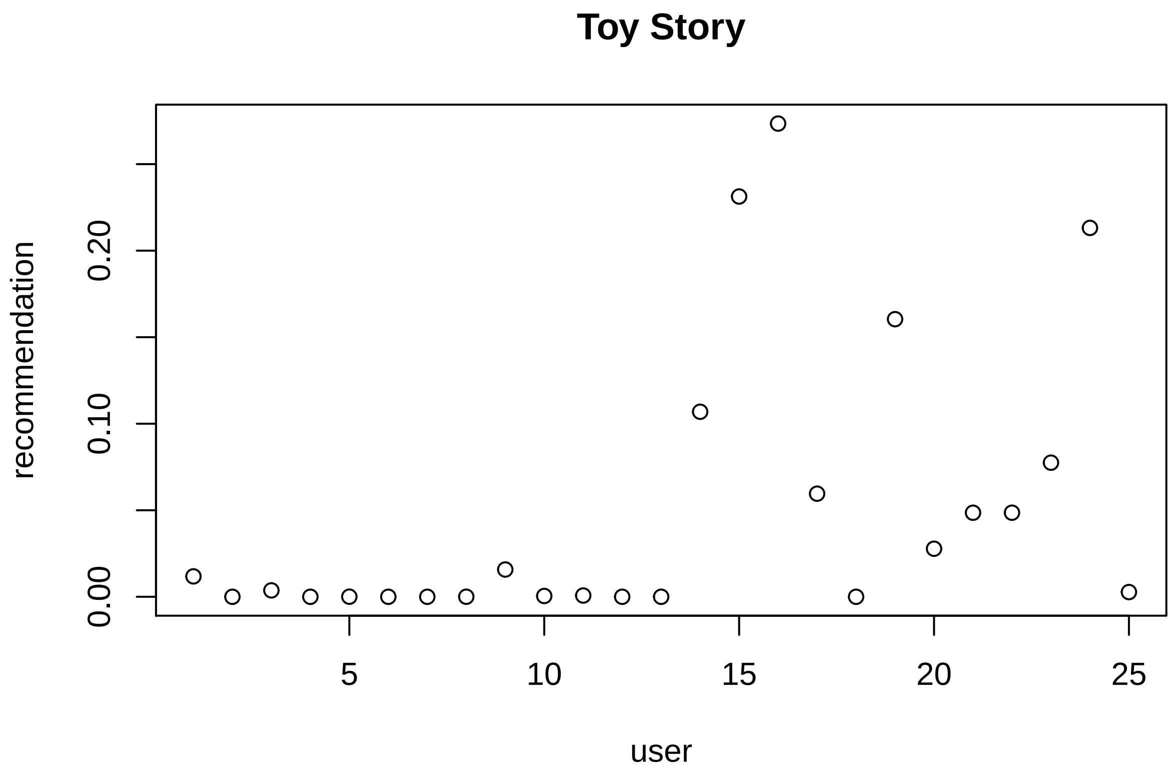
If interested, the details about the models can be seen below:

```
ml.mf
```

```
## <Object of class: NMFfit>
##  # Model:
##   <Object of class:NMFstd>
##   features: 25
##   basis/rank: 25
##   samples: 25
##  # Details:
##   algorithm:  brunet
##   seed:  random
##   RNG: 403L, 241L, ..., -795384973L [e64112521a756faef0c89ca9ff5341a1]
##   distance metric:  'KL'
##   residuals:  0.183719
##   Iterations: 1750
##   Timing:
##      user  system elapsed
##      0.22    0.00    0.22
```

The above information gives us enough to go on to make some recommendations to a user. For example in the graph below, the first movie, Toy Story, is shown for each of the 25 users the NMF was run for. A number closer to 0 represents a movie the user would be more likely to want to see. This is at a slight discrepancy due to the random values being filled in. Unfortunately NMF does not support, at least to my knowledge, NA values.

```
ml.cf <- data.frame(t(coef(ml.mf)))
plot(ml.cf$X1, main = "Toy Story", xlab = "user", ylab = "recommendation")
```



SVD Method

The second method, similar to the above, is with SVD. One difference is that SVD can work with negatives, while NMF cannot. In this set, there are no negatives, so there should not be any obvious advantage to one over the other.

I'll reload the data fresh, so there's no overlap between the two methods. It then needs to be normalized, and each segment of the SVD needs to be saved.

```
ml.matrix <- as(MovieLense, "matrix")
ml.matrix[is.na(ml.matrix)] <- sample(1:5, length(ml.matrix[is.na(ml.matrix)]),replace = T)
ml.svd <- svd(scale(ml.matrix))
```

The next step here is finding

Citations

http://econometricsense.blogspot.com/2012/10/nonnegative-matrix-factorization-and.html

http://www.r-bloggers.com/using-the-svd-to-find-the-needle-in-the-haystack/