Exercise 9 Sparse Matrix Factorization **Cenying(Tracy) Yang**
SDS 385 Due Date: 11/16/2016
Prof. James Scott

Algorithm

Witten et al. (2009) proposes a penalized matrix decomposition, with applications to sparse principal component and canonical correlation analysis. Here, I summarize how the algorithm works.
The algorithm attempts to solve the following problem

$$\begin{aligned} \underset{u\in R^N, v\in R^p}{\text{minimize}} \quad & \frac{1}{2}\|X - duv^T\|_F^2 \\ \text{subject to} \quad & \|u\|_2^2 = 1, \quad \|v\|_2^2 = 1 \\ & \|u\|_1 \le c_1, \quad \|v\|_1 \le c_2 \end{aligned}$$

The problem is equivalent to

$$\begin{aligned} \underset{u\in R^N, v\in R^p}{\text{maximize}} \quad & u^T X v \\ \text{subject to} \quad & \|u\|_2^2 = 1, \quad \|v\|_2^2 = 1 \\ & \|u\|_1 \le c_1, \quad \|v\|_1 \le c_2 \end{aligned}$$

The ***rank 1 matrix approximation*** algorithm runs as follows:

1. Initialize v with ℓ_2 norm being 1
2. Update $u = \frac{S_{\Delta_1}(Xv)}{\|S_{\Delta_1}(Xv)\|_2}$
3. Update $v = \frac{S_{\Delta_2}(X^T u)}{\|S_{\Delta_2}(X^T u)\|_2}$

where Δ_1 and Δ_2 are chosen based on binary search such that $\|u\|_1 = c_1$ and $\|v\|_1 = c_2$ respectively. And S is the soft thresholding operator.
The ***rank k matrix approximation*** algorithm is built based on rank 1 algorithm.

1. Initialized $X^1 \leftarrow X$
2. For $k \in 1, ..., K$:
   (a) Find u_k, v_k and d_k by applying the rank 1 matrix approximation algorithm to data X^k
   (b) $X^{T+1} \leftarrow X^k - d_k u_k v_k^T$

Application to social marketing data

I transform the data using square root so that standard deviation of count data is approximately constant.
A quick look into what topics are discussed most frequently on the Twitter:

```
                 colmeans
chatter          1.9094277
photo_sharing    1.3681041
current_events   1.0501033
health_nutrition 1.0167170
travel           0.9549531
cooking          0.9548596
```

```
sports_fandom    0.9266684
politics         0.9127590
food             0.8753729
shopping         0.8478124
college_uni      0.7991081
personal_fitness 0.7801828
tv_film          0.6993931
news             0.6925771
uncategorized    0.6505390
religion         0.6451401
family           0.6420915
online_gaming    0.6414276
parenting        0.6044721
fashion          0.6030449
automotive       0.5741795
outdoors         0.5594291
school           0.5515864
music            0.5246168
sports_playing   0.4988884
beauty           0.4791737
computers        0.4741713
art              0.4619582
home_and_garden  0.4469229
dating           0.4349158
eco              0.4313205
crafts           0.4241252
business         0.3661789
small_business   0.2972368
adult            0.1598987
spam             0.0063218
```

I implement ***rank 1*** algorithm to find out the major topics with different threshold c_1 and c_2. **Results with $c_1 = 5$ and $c_2 = 5$:**

```
                       [,1]
chatter          0.33546158
current_events   0.14109283
travel           0.23827754
photo_sharing    0.28766436
uncategorized    0.08844721
tv_film          0.09532899
sports_fandom    0.17428162
politics         0.29400356
food             0.20316637
family           0.12305913
home_and_garden  0.02597041
music            0.07265437
news             0.18950121
online_gaming    0.02882779
shopping         0.17889782
health_nutrition 0.34411310
college_uni      0.11984188
sports_playing   0.06698127
```

```
cooking          0.29931693
eco              0.05654087
computers        0.11904043
business         0.04288818
outdoors         0.11606007
crafts           0.03991565
automotive       0.09678883
art              0.06787109
religion         0.18920898
beauty           0.14555424
parenting        0.16155267
dating           0.11070105
school           0.13720559
personal_fitness 0.21847360
fashion          0.15848661
small_business   0.03282414
spam             0.00000000
adult            0.00000000
```

Results with $c_1 = 3$ and $c_2 = 3$:

```
                       [,1]
chatter          0.44257137
current_events   0.06261090
travel           0.19925966
photo_sharing    0.18299416
uncategorized    0.00000000
tv_film          0.00000000
sports_fandom    0.03580287
politics         0.28373996
food             0.13758287
family           0.00000000
home_and_garden  0.00000000
music            0.00000000
news             0.14357340
online_gaming    0.00000000
shopping         0.07275263
health_nutrition 0.66699643
college_uni      0.00000000
sports_playing   0.00000000
cooking          0.11345524
eco              0.00000000
computers        0.00000000
business         0.00000000
outdoors         0.05049464
crafts           0.00000000
automotive       0.00000000
art              0.00000000
religion         0.09896652
beauty           0.00000000
parenting        0.05501594
dating           0.06175267
school           0.04381438
```

```
personal_fitness 0.34861642
fashion          0.00000000
small_business   0.00000000
spam             0.00000000
adult            0.00000000
```

Results with $c_1 = 1.5$ and $c_2 = 1.5$:

```
                       [,1]
chatter          0.50167170
current_events   0.00000000
travel           0.00000000
photo_sharing    0.06085227
uncategorized    0.00000000
tv_film          0.00000000
sports_fandom    0.00000000
politics         0.00000000
food             0.00000000
family           0.00000000
home_and_garden  0.00000000
music            0.00000000
news             0.00000000
online_gaming    0.00000000
shopping         0.00000000
health_nutrition 0.85937328
college_uni      0.00000000
sports_playing   0.00000000
cooking          0.00000000
eco              0.00000000
computers        0.00000000
business         0.00000000
outdoors         0.00000000
crafts           0.00000000
automotive       0.00000000
art              0.00000000
religion         0.00000000
beauty           0.00000000
parenting        0.00000000
dating           0.00000000
school           0.00000000
personal_fitness 0.07810288
fashion          0.00000000
small_business   0.00000000
spam             0.00000000
adult            0.00000000
```

As we can see, **as penalty c_1 and c_2 shrinks, fewer topics will be chosen**.

Class Notes

Q1: LDA v.s. matrix factorization
LDA provides the distribution of topics. It is a unsupervised learning algorithm, but it is less scaleble and takes more time.
Matrix factorization depends on penalty parameters and gives a specific results based on the choice of penalty parameters. But it is more scaleble since it deals well with sparse matrix and runs much faster.

Q2: When to use covariance matrix and when to use original matrix when doing matrix factorization?
PCA uses covariance matrix, which is

$$S = \frac{1}{n} X^T X \tag{1}$$

where $X \in R^{n \times D}$, and $S \in R^{D \times D}$. Factorize it, we get $S = UDU^T$.
If we use the original matrix, we have

$$X = U_L \Omega U_R^T \tag{2}$$

where $U_L \in R^{n \times D}$, and $\Omega_L = diag(\lambda_1, ..., \lambda_D)$, and $U_R \in R^{D \times D}$, which are orthogonal.
Then, S takes the form of

$$\frac{1}{n} X^T X = \frac{1}{n} U_R \Omega U_L^T U_L \Omega U_R^T = U_R (\frac{1}{n} \Omega^2) U_R^T \tag{3}$$

Denote $D = \frac{1}{n}\Omega^2$. $S_1 = X_1^T U_R^{(1)}$, and $XU_R = U_L \Omega$.