COMS 4721: Machine Learning for Data Science

Columbia University, Spring 2015

Homework 5: Due April 28, 2015

Submit the written portion of your homework as a single PDF file through Courseworks (less than 5MB). In addition to your PDF write-up, submit all code written by you in their original extensions through Courseworks. Do not submit in .rar, .zip, .tar, .doc, or other file types. Your grade will be based on the contents of one PDF file and original source code. Everything resulting from the problems on this homework other than the raw code should be put in the PDF file.

Show all work for full credit. Late homeworks will not be accepted – i.e., homework submitted to Courseworks after midnight on the due date.

Problem 1 (Markov chains) – 40 points

In this problem, you will rank 759 college football teams based only on the scores of every game in the 2014 season. The data provided contains the result of one game on each line. For the ith line, the information contained in "scores" is

```
scores(i,1) = Team1 index,  scores(i,2) = Team1 points,
scores(i,3) = Team2 index,  scores(i,4) = Team2 points.
```

If `scores(i,2) > scores(i,4)` then Team1 wins and Team2 loses, and vice versa. The index of a team refers to the row of "legend" where that team's name can be found.

Construct a 759×759 random walk matrix M on the college football teams. First construct the unnormalized matrix $\hat{M}$, initially set to all zeros. For a particular game i, let j_1 be the index of Team1 and j_2 the index of Team2. Then update

$$\hat{M}_{j_1j_1} \leftarrow \hat{M}_{j_1j_1} + \mathbb{1}\{\texttt{Team1 wins}\} + \frac{\texttt{points}_{j_1}}{\texttt{points}_{j_1}+\texttt{points}_{j_2}},$$

$$\hat{M}_{j_2j_2} \leftarrow \hat{M}_{j_2j_2} + \mathbb{1}\{\texttt{Team2 wins}\} + \frac{\texttt{points}_{j_2}}{\texttt{points}_{j_1}+\texttt{points}_{j_2}},$$

$$\hat{M}_{j_1j_2} \leftarrow \hat{M}_{j_1j_2} + \mathbb{1}\{\texttt{Team2 wins}\} + \frac{\texttt{points}_{j_2}}{\texttt{points}_{j_1}+\texttt{points}_{j_2}},$$

$$\hat{M}_{j_2j_1} \leftarrow \hat{M}_{j_2j_1} + \mathbb{1}\{\texttt{Team1 wins}\} + \frac{\texttt{points}_{j_1}}{\texttt{points}_{j_1}+\texttt{points}_{j_2}}.$$

After processing all games, let M be the matrix formed by normalizing the rows of $\hat{M}$ so they sum to 1.

- Let w_t be the 759×1 state vector at step t. Set w_0 to the uniform distribution. Therefore, w_t is the distribution of the state after t steps given that the starting state at time 0 is uniformly distributed.
- Use w_t to rank the teams by sorting in decreasing value according to this vector. List the top 20 teams and their corresponding values in w_t for $t = \{10, 100, 200, 1000\}$.
- We saw that w_∞ corresponds to the first eigenvector of M^T: $M^T w_\infty = w_\infty \Rightarrow M^T u_1 = \lambda_1 u_1$, $\lambda_1 = 1$ and $w_\infty = u_1/\sum_j u_1(j)$ (since by convention $u_1^T u_1 = 1$). Plot $\|w_t - u_1/\sum_j u_1(j)\|_1$ as a function of t for $t = 1, \ldots, 1000$. What is $\|w_{1000} - u_1/\sum_j u_1(j)\|_1$?

Problem 2 (Nonnegative matrix factorization) – 60 points

In this problem you will factorize a $n \times m$ matrix X into a rank-K approximation WH, where W is $n \times K$, H is $K \times m$ and all values in the matrices are nonnegative. Each value in W and H can be initialized randomly, e.g., from a Uniform(0,1) distribution. (See a hint about the implementation below.)

<u>*Part 1*</u>: The data to be used for Part 1 consists of 1000 images of faces, each originally 32×32, but vectorized to length 1024. The data matrix is therefore 1024×1000.

- Implement and run the NMF algorithm on this data using the *Euclidean penalty*. Set the rank of the factorization to 25 and run for 200 iterations.
- Plot the objective as a function of iteration.
- For 3 images in the data set, show the original image and the column of W for which the corresponding weight in H is the largest. This should be shown as two 32×32 images. Select the three images so that the columns displayed from W are different.

<u>*Part 2*</u>: The data to be used for Part 2 consists of 8447 documents from *The New York Times*. (See below for how to process the data.) The vocabulary size is 3012 words. You will need to use this data to constitute the matrix X, where X_{ij} is the number of times word i appears in document j. Therefore, X is 3012×8447 and most values in X will equal zero.

- Implement and run the NMF algorithm on this data using the *divergence penalty*. Set the rank (i.e., number of topics) to 25 and run for 200 iterations.
- Plot the objective as a function of iteration.
- After running the algorithm, normalize the columns of W so they sum to one. Pick 5 columns of W. For each selected column show the 10 words having the largest probability according to the values in W and give the probabilities. The ith row of W corresponds to the ith word in the "dictionary" provided with the data.

HINT: When dividing, you may get 0/0 = NaN which will propagate, causing the algorithm to output only NaN's. In the division, add a very small number (e.g., 10^{-16}) to the denominator to avoid this.

```
About the text data used in Part 2:

MATLAB
- Xid is a cell array. Xid{d} is a vector giving the indexes of words
      appearing in document d. Any word not listed doesn't appear.
- Xcnt is a cell array. Xcnt{d} is a vector indicating that word
       Xid{d}(i) appears Xcnt{d}(i) times in document d.

CSV
- Each row in nyt_data.txt corresponds to a single document. It
  gives the index of words appearing in that document and the number
  of times they appear. It uses the format "idx:cnt" with commas
  separating each unique word in the document.
```