ELEN 4903: Machine Learning

Columbia University, Spring 2018

Homework 5: Due April 30, 2017 by 11:59pm

Please read these instructions to ensure you receive full credit on your homework. 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 (e.g., .m, .r, .py, .c). Any coding language is acceptable, but do not submit notebooks, do not wrap your files in .rar, .zip, .tar and do not submit your write-up in .doc or other file type. Your grade will be based on the contents of one PDF file and the original source code. Additional files will be ignored. We will not run your code, so everything you are asked to show should be put in the PDF file. Show all work for full credit.

Late submission policy: Late homeworks will have 0.1% deducted from the final grade for each minute late. *Your homework submission time will be based on the time of your **last** submission to Courseworks. I will not revert to an earlier submission!* Therefore, do not re-submit after midnight on the due date unless you are confident the new submission is significantly better to overcompensate for the points lost. Submission time is non-negotiable and will be based on the time you submitted your last file to Courseworks. The number of points deducted will be rounded to the nearest integer.

Problem 1 (Markov chains) – 30 points

In this problem, you will rank 763 college football teams based on the scores of every game in the 2017 season. The data provided in `CFB2017_scores.csv` contains the result of one game on each line in the format

`Team A index, Team A points, Team B index, Team B points.`

If Team A has more points than Team B, then Team A wins, and vice versa. The index of a team refers to the row of "TeamNames.txt" where that team's name can be found.

Construct a 763×763 random walk matrix M on the college football teams. First construct the unnormalized matrix $\widehat{M}$ with values initialized to zeros. For one particular game, let i be the index of Team A and j the index of Team B. Then update

$$\widehat{M}_{ii} \leftarrow \widehat{M}_{ii} + \mathbb{1}\{\texttt{Team A wins}\} + \frac{\texttt{points}_i}{\texttt{points}_i+\texttt{points}_j},$$

$$\widehat{M}_{jj} \leftarrow \widehat{M}_{jj} + \mathbb{1}\{\texttt{Team B wins}\} + \frac{\texttt{points}_j}{\texttt{points}_i+\texttt{points}_j},$$

$$\widehat{M}_{ij} \leftarrow \widehat{M}_{ij} + \mathbb{1}\{\texttt{Team B wins}\} + \frac{\texttt{points}_j}{\texttt{points}_i+\texttt{points}_j},$$

$$\widehat{M}_{ji} \leftarrow \widehat{M}_{ji} + \mathbb{1}\{\texttt{Team A wins}\} + \frac{\texttt{points}_i}{\texttt{points}_i+\texttt{points}_j}.$$

After processing all games, let M be the matrix formed by normalizing the rows of $\widehat{M}$ so they sum to one. Let w_t be the 1×763 state vector at step t. Set w_0 to the uniform distribution. Therefore, w_t is the marginal distribution on each state after t steps given that the starting state is chosen uniformly at random.

a) Use w_t to rank the teams by sorting in decreasing value according to this vector. List the top 25 teams and their corresponding values in w_t for $t = 10, 100, 1000, 10000$.

b) We saw that w_∞ is related to the first eigenvector of M^T. That is, we can find w_∞ by getting the first eigenvector and eigenvalue of M^T and post-processing:

$$M^T u_1 = \lambda_1 u_1, \quad w_\infty = u_1^T / \left[\sum_j u_1(j)\right]$$

This is because $u_1^T u_1 = 1$ by convention. Also, we observe that $\lambda_1 = 1$ for this specific matrix. Plot $\|w_t - w_\infty\|_1$ as a function of t for $t = 1, \ldots, 10000$.

Problem 2 (Nonnegative matrix factorization) – 30 points

In this problem you will factorize an $N \times M$ 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 to a positive number, e.g., from a Uniform(1,2) distribution.

The data to be used for this problem 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 construct 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.

a) Implement and run the NMF algorithm on this data using the *divergence penalty*. Set the rank to 25 and run for 100 iterations. This corresponds to learning 25 topics. Plot the objective as a function of iteration.

b) After running the algorithm, normalize the columns of W so they sum to one. For each column of W, list the 10 words having the largest weight and show the weight. The ith row of W corresponds to the ith word in the "dictionary" provided with the data. Organize these lists in a 5×5 table.

Comments about Problem 2:

- When dividing, you may get 0/0 = NaN. This will cause the algorithm to return all NaN values. You can add a very small number (e.g., 10^{-16}) to the denominator to avoid this.
- 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. Any index that doesn't appear in a row has a count of zero for that word. The vocabulary word corresponding to each index is given in the corresponding row of `nyt_vocab.dat`.