CS432 Spring 2018

Assignment 9

Jonathan Kruszewski

04/29/18

CS 432/532 Web Science

Spring 2018
http://anwala.github.io/lectures/cs532-s18/

Assignment #9
Due: 11:59pm April 21

Support your answer: include all relevant discussion, assumptions, examples, etc.

(10 points)
1. Using the data from A7:

- Consider each row in the blog-term matrix as a 1000 dimension vector, corresponding to a blog.

- Use knnestimate() to compute the nearest neighbors for both:
    http://f-measure.blogspot.com/
    http://ws-dl.blogspot.com/

    for k={1,2,5,10,20}.

    Use cosine distance metric (chapter 8) not euclidean distance. So you have to implement numpredict.cosine() instead of using numpredict.euclidean() in:
    https://github.com/arthur-e/Programming-Collective-Intelligence/blob/master/chapter8/numpredict.py

==========
========The questions below is for 3 points extra credit==========
==========

3. Re-download the 1000 TimeMaps from A2, Q2. Create a graph where the x-axis represents the 1000 TimeMaps. If a TimeMap has "shrunk", it will have a negative value below the x-axis corresponding to the size difference between the two TimeMaps. If it has stayed the same, it will have a "0" value. If it has grown, the value will be positive and correspond to the increase in size between the two TimeMaps.

As always, upload all the TimeMap data. If the A2 github has the original TimeMaps, then you can just point to where they are in the report.

Part 1:

1. Using the data from A7:

- Consider each row in the blog-term matrix as a 1000 dimension vector, corresponding to a blog.

- Use knnestimate() to compute the nearest neighbors for both:
  http://f-measure.blogspot.com/
  http://ws-dl.blogspot.com/

  for k={1,2,5,10,20}.

  Use cosine distance metric (chapter 8) not euclidean distance. So you have to implement numpredict.cosine() instead of using numpredict.euclidean() in:
  https://github.com/arthur-e/Programming-Collective-Intelligence/blob/master/chapter8/numpredict.py

To accomplish this task, “clusters.py” and “blogdata1 (copy).txt” from assignment 7 were used, in addition to a modified “numpredict.py” provided from https://github.com/arthur-e/Programming-Collective-Intelligence/blob/master/chapter8/numpredict.py , and a python program “FindkNN.py”. First “numpredict.py” was modified to add a cosine distance function to be used in the place of the preexisting euclidean distance function. “FindkNN.py” calls the readfile function in “clusters.py”to read the “blogdata1 (copy).txt” file from the previous assignment. It then reads separate files for the data for the f-measure and ws-dl blogs “fmesureData.txt” and “ws-dlData.txt”, these files contain only the data and no non-integer content for their respective blogs. “FindkNN.py” then calls the knnestimate function in “numpredict.py” which has been modified to return a list of the k closest neighbors. Output for k = 1,2,5 for F-Measure can be found below, full output can be found in “F-Measure.txt” and “Web Science and Digital Libraries Research Group.txt”

```
Nearest Neighbors For  F-Measure (k =  1 )
        F-Measure       ---         1.0

Nearest Neighbors For  F-Measure (k =  2 )
        F-Measure       ---         1.0
        SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA      ---         0.5488766405456125

Nearest Neighbors For  F-Measure (k =  5 )
        F-Measure       ---         1.0
        SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA      ---         0.5488766405456125
        CardrossManiac2         ---         0.5364194539091759
        Aiming to misbehave     ---         0.5093344205413922
        DaveCromwell Writes     ---         0.5082421990442522
```

Part 2:

3. Re-download the 1000 TimeMaps from A2, Q2. Create a graph where the x-axis represents the 1000 TimeMaps. If a TimeMap has "shrunk", it will have a negative value below the x-axis corresponding to the size difference between the two TimeMaps. If it has stayed the same, it will have a "0" value. If it has grown, the value will be positive and correspond to the increase in size between the two TimeMaps.

As always, upload all the TimeMap data. If the A2 github has the original TimeMaps, then you can just point to where they are in the report.

This task was accomplished by using the Timestamp files and “GetCarbonDate.py” program from Assignment 2, a python program, “TimeMapsCount.py”, written to calculate the change in number of time maps, and an R script, “TimestampChangeGraphRCode.txt”, written to graph the change. Below are the two graphs produced by the R script, the top graph had the maximum y value set to 28 so the negative changes would be visible, the bottom graph is the full view of the graphed data.