CS 432: Homework Number Nine

Due on April 30, 2019 at 11:59 PM

Alexander Nwala

Tim Bruce

Problem 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()

Solution

This is it... the final question.
It was actually fairly difficult to make the most popular term 1000 item matrices. I adapted a function that went unused in A7 to work with clusters.readfile data, and then I made a nifty script to make the matrices for a single website, and then for all of the websites.

Listing 1: Functions for creating a blog-term matrix.

```
def popularTerms(data, wordorder):
    # Gets the most popular 1000 words from the dataset (or all of the words).
    words = {}

    for wordKey in range(len(wordorder)):
        word = wordorder[wordKey]
        wordcount = 0
        for line in data:
            wordcount += int(line[wordKey])
        if word in words.keys():
            words[word] += wordcount
        else:
            words[word] = wordcount
    if len(words) > 1000:
        return sorted(words.items(), key=lambda x: -x[1])[:1000]
    else:
        return sorted(words.items(), key=lambda x: -x[1])[:len(words)-1]


def blog_term_matrix(popTerms, data, words, blogNames):
    # Creates the blog-term matrix for all of the blogs.
    website_matricies = []
    for site in blogNames:
        website_matricies.append([site, get_blog_term_matrix(popterms, data[blogNames.
    index(site)], words)])

    return website_matricies


def get_blog_term_matrix(popTerms, site_data, words):
    # Creates the blog term matrix for a single website.
    matrix = []
    for item in popTerms:
        word_index = words.index(item[0])
        matrix.append(site_data[word_index])

    return matrix
```

The popularTerms function is the adapted function from A7. It selects the most popular 1000 words in the dataset from clusters.readfile in order. If there are not 1000 words, it does as many as it can. blog_term_matrix creates the blog-term matrix by getting count of the word in each site, ordered by word in the popularTerms output.

I found the cosine function in scipy.spatial.distance, and it was simple to implement. The use of knnestimate is something I've posted a question about in slack a day ago and haven't gotten a response about. The function knnestimate is designed for estimating the price of wine as an average based on price and age. We have words and for this question we a looking for similar vectors. We can use part of knnestimate, but we need to modify it. I have modified knnestimate as shown below.

Listing 2: Modified knnestimate for use in this assignment.

```python
def knnestimate(data, vec1, k=5):
    # Get sorted distances
    dlist = getdistances(data, vec1)
    avg = []

    # Take the average of the top k results
    for i in range(k):
        avg.append(data[dlist[i][1]][0])

    return avg
```

Note that I have modified it to just return the top k results. If this has been completed incorrectly, I'd be happy to fix it with some explanation as to what value I'm supposed to be averaging in knnestimate. Here are the top 20 results it returned for each website in order.

```
F-Measure
["Sophie's Floorboard", 'film babble blog', "Henry's Western Round-up", 'Embracing Hope', 'Anonymous Lawyer', 'Althouse', 'Pesach Sheini', 'Musings on the Muse', "Let's Backtrack", 'Kitchen Witch Blog', 'Good Research Papers', 'Inexplicata-The Journal of Hispanic Ufology', 'Onofframp', 'Bootpacks and Skintracks', 'Words Worth', "Aile Reve's Blog", 'Mandela Effect Proof', 'In From the Cold', 'Realm of a Green Witch', "Rebecca Tushnet's 43(B)log"]

ws-dl

['A Capital Idea', 'Tanfield Railway Blog', 'English Words ending in aa, ah...zy', '::: Back Tracking :::', 'BackTrack', "Julia's coding blog - Practice makes perfect", 'Ubuntu Tips And Tricks', 'Comments on Radical Honey: Bewitched by the Blackthorn Being, part II ~ sacrifice & service', 'Onofframp', 'Rent a Carry Daba 7 Seater on Cheap Rates At Lahore', 'Ghana Consumer Watch', 'Index to The Recipe file', "Quiltville's Quips & Snips!!", 'Shoulder Arthritis / Rotator Cuff Tears: causes of shoulder pain', "Comments on Ms. Yingling Reads: The Assassin's Curse (Blackthorn Key #3)", 'asphyxia', 'Words Going Wild', 'Words, words, words (and phrases)', "Blaugustine's Other Blog", 'Wild Plants Post']

```

It's been an amazing semester. Have a wonderful summer!