Old Dominion Univeristy

CS 495: Introduction to Web Science
Instructor: Michael L. Nelson, Ph.D
Fall 2014 4:20pm - 7:10pm R, ECSB 2120

Assignment # 11

George C. Micros UIN: 00757376

Honor Pledge

I pledge to support the Honor System of Old Dominion University. I will refrain from any form of academic dishonesty or deception, such as cheating or plagIiarism. I am aware that as a member of the academic community it is my responsibility to turn in all suspected violations of the Honor Code. I will report to a hearing if summoned.

Signed _____
December 11, 2014

George C. Micros

Written Assignment 11

Fall 2014
CS 495: Introduction to Web Science
Dr. Michael Nelson

December 11, 2014

Contents

1 Written Assignment 11 5
1.1 Question 1 6
1.1.1 The Question 6
1.1.2 The Answer 6

Chapter 1
Written Assignment 11

1.1 Question 1

1.1.1 The Question

Using the data from A9:
- Consider each row in the blog-term matrix as a 500 dimension vector, corresponding to a blog.
- From chapter 8, replace numpredict.euclidean() with cosine as the distance metric. In other words, you'll be computing the cosine between vectors of 500 dimensions.
- 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.

1.1.2 The Answer

A function for the cosine similarity was made based off the definition of the dot product. This was used with the nnestimate() function

```
#! /usr/bin/python

import math

def cos_sim(v1, v2):
  sumxx, sumyy, sumxy = 0 ,0 ,0
  for i in range(len(v1)):
    x = v1[i]; y = v2[i]
    sumxx += float(x)*float(x)
    sumyy += float(y)*float(y)
    sumxy += float(x)*float(y)
  return sumxy/math.sqrt(sumxx*sumyy)

def getdistances(data,vec1):
  distancelist=[]

  # Loop over every item in the dataset
  for i in range(len(data)):
    vec2=data[i]

    try:
      distancelist.append((cos_sim(vec1,vec2),i))
    except:
      pass

  # Sort by distance
  distancelist.sort()
  return distancelist

def knnestimate(data,vec1,k=5):
  # Get sorted distances
  dlist=getdistances(data,vec1)
  avg=0.0
  return dlist
vecs = {}

f = open("blogdata2.txt", "r")

for line in f:
  a = line.strip('\n').split('\t');
  b = a.pop(0)
  vecs[b] = a

print len(vecs)

fm = 'F-Measure'
ws = 'Web Science and Digital Libraries Research Group'

a = vecs[fm]
temp = vecs.values()
temp.pop(vecs.keys().index(fm))
```

```
53 a = knnestimate(temp,a,k=5)

55

k = [1, 2, 5, 10, 20]
57 print "--------F-Measure kNN--------"
for i in k:
59   print "---k = "+str(i)
   for j in range(i):
61     b = a[j][1]
     print vecs.keys()[b]
63

a = vecs[ws]
65 temp = vecs.values()
temp.pop(vecs.keys().index(ws))
67

a = knnestimate(temp,a,k=5)
69


71 k = [1, 2, 5, 10, 20]
print "--------WS-DL kNN--------"
73 for i in k:
   print "---k = "+str(i)
75   for j in range(i):
     b = a[j][1]
77     print vecs.keys()[b]
```

Listing 1.1: Python script that computes kNN based on cosine similarity

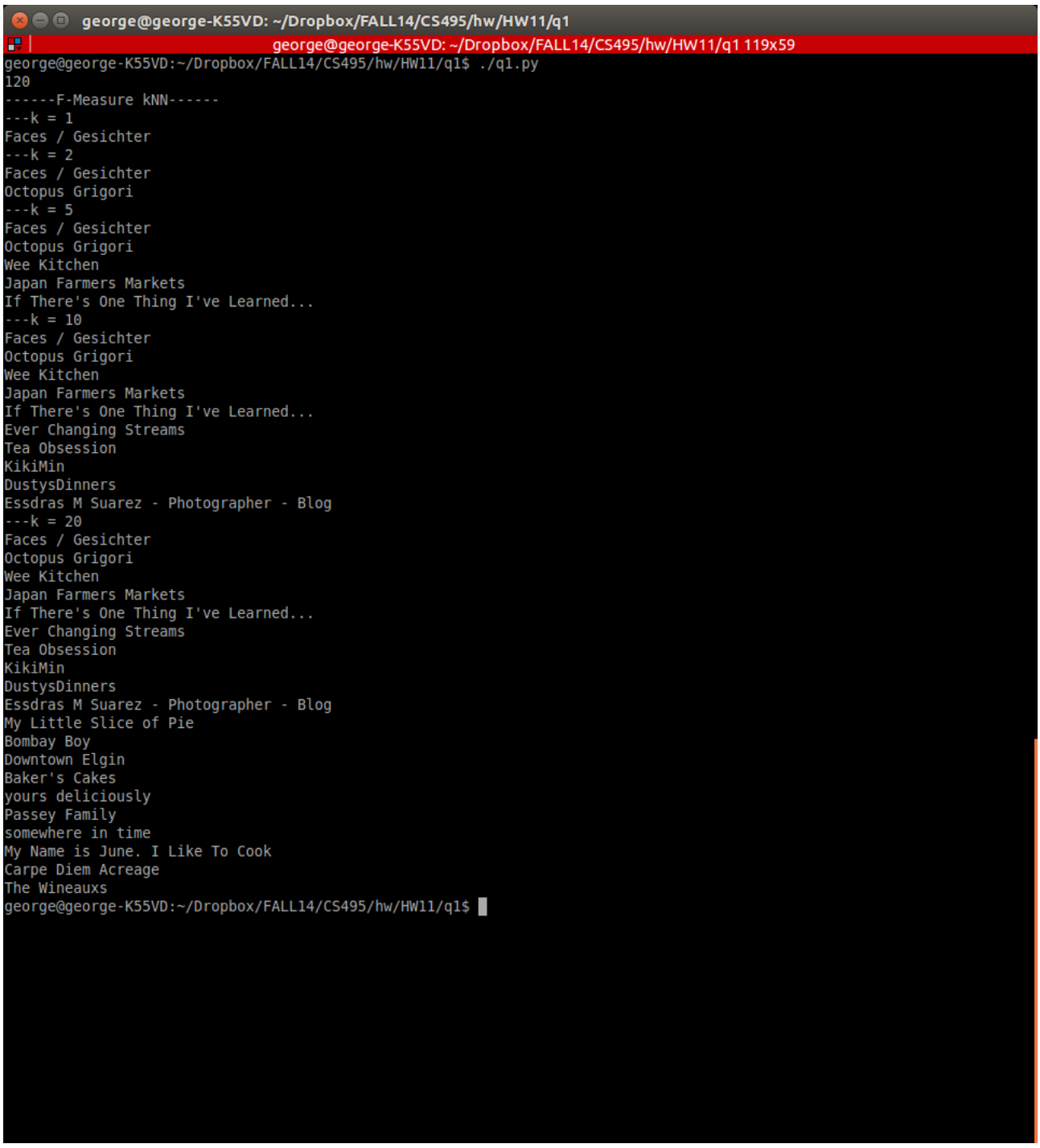

Fig. 1.1: Clustings for the F-Measure blog

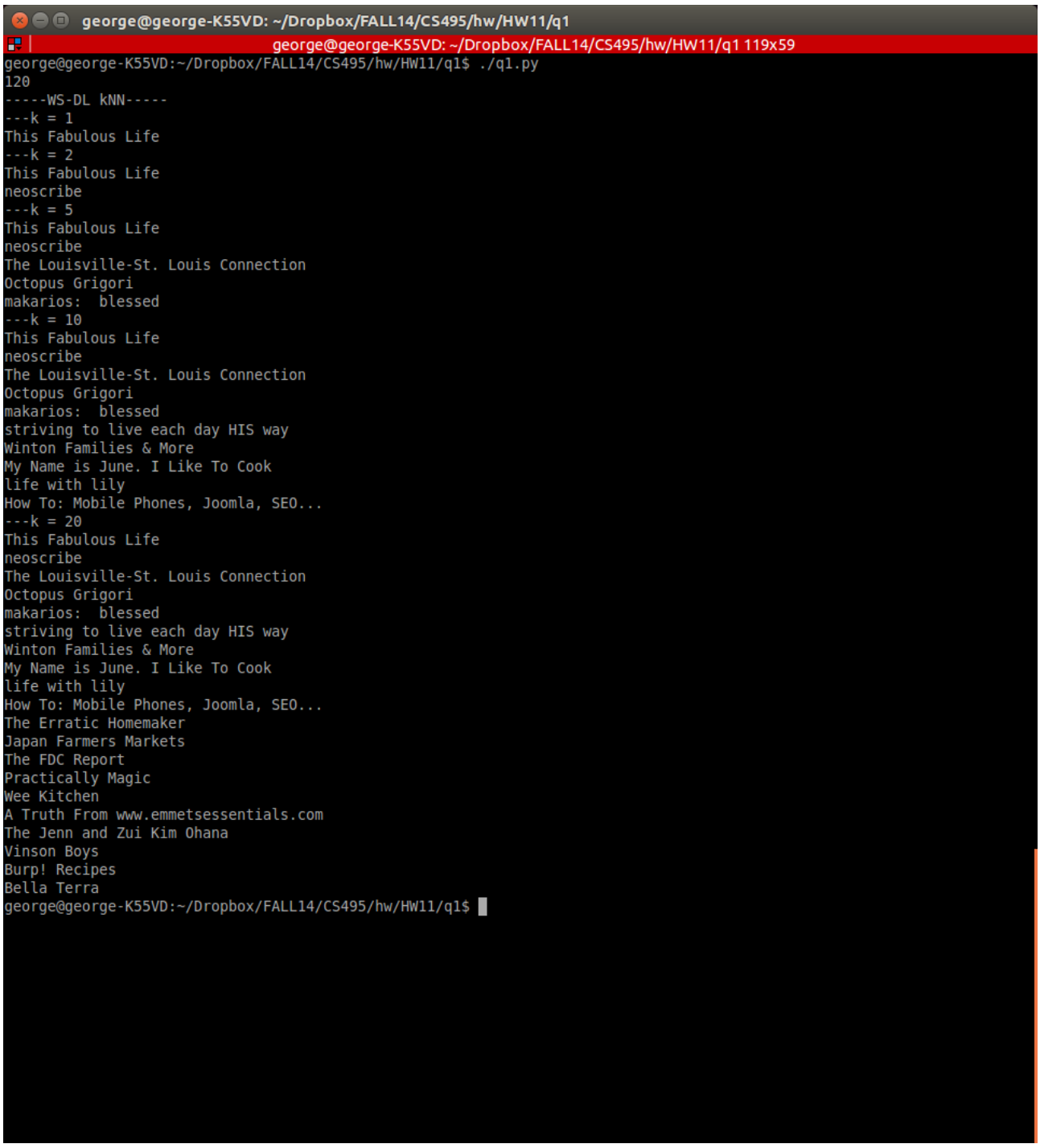

Fig. 1.2: CLusterings for the WS-DL blogs