Web Science: Assignment #9

Alexander Nwala

Puneeth Bikkasandra

Tuesday, May 1, 2018

Contents

Problem 1

Using the data from A7:

1. Consider each row in the blog-term matrix as a 1000 dimension vector, corresponding to a blog.
2. 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

SOLUTION :

I have solved the problem as described in the below steps :

1. I have used the "blogData.txt" blog metric from A7 and code files from **Programming Collective Intelligence** text book.
2. Modified the "numpredict" file to add the cosine similarity function
3. Created a new file ,"nearestEstimate"
4. Passed the title of the two blogs, whose nearest neighbours need to be determined

Listing 1: numpredict.py

```
from random import random,randint
import math

def wineprice(rating,age):
  peak_age=rating-50

  # Calculate price based on rating
  price=rating/2
  if age>peak_age:
    # Past its peak, goes bad in 10 years
    price=price*(5-(age-peak_age)/2)
  else:
    # Increases to 5x original value as it
    # approaches its peak
    price=price*(5*((age+1)/peak_age))
  if price<0: price=0
  return price


def wineset1():
  rows=[]
  for i in range(300):
    # Create a random age and rating
```

```
    rating=random()*50+50
    age=random()*50

    # Get reference price
    price=wineprice(rating,age)

    # Add some noise
    price*=(random()*0.2+0.9)

    # Add to the dataset
    rows.append({'input':(rating,age),
                 'result':price})
  return rows

def euclidean(v1,v2):
  d=0.0
  for i in range(len(v1)):
    d+=(v1[i]-v2[i])**2
  return math.sqrt(d)


def cosineDistance(x, y):
    return np.dot(x, y) / (np.sqrt(np.dot(x, x)) * np.sqrt(np.dot(y, y)))


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

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

    # Add the distance and the index
    distancelist.append((cosineDistance(vec1,vec2),i))

  # Sort by distance
  distancelist.sort()
  return distancelist

def knnestimate(data,vec1,k=5):
  # Get sorted distances
  dlist=getdistances(data,vec1)
  neighbors = dlist[-k-1:-1]
  return neighbors

def inverseweight(dist,num=1.0,const=0.1):
  return num/(dist+const)

def subtractweight(dist,const=1.0):
  if dist>const:
    return 0
  else:
    return const-dist
```

```
def gaussian(dist,sigma=5.0):
  return math.e**(-dist**2/(2*sigma**2))

def weightedknn(data,vec1,k=5,weightf=gaussian):
  # Get distances
  dlist=getdistances(data,vec1)
  avg=0.0
  totalweight=0.0

  # Get weighted average
  for i in range(k):
    dist=dlist[i][0]
    idx=dlist[i][1]
    weight=weightf(dist)
    avg+=weight*data[idx]['result']
    totalweight+=weight
  if totalweight==0: return 0
  avg=avg/totalweight
  return avg

def dividedata(data,test=0.05):
  trainset=[]
  testset=[]
  for row in data:
    if random()<test:
      testset.append(row)
    else:
      trainset.append(row)
  return trainset,testset

def testalgorithm(algf,trainset,testset):
  error=0.0
  for row in testset:
    guess=algf(trainset,row['input'])
    error+=(row['result']-guess)**2
    #print row['result'],guess
  #print error/len(testset)
  return error/len(testset)

def crossvalidate(algf,data,trials=100,test=0.1):
  error=0.0
  for i in range(trials):
    trainset,testset=dividedata(data,test)
    error+=testalgorithm(algf,trainset,testset)
  return error/trials

def wineset2():
  rows=[]
  for i in range(300):
    rating=random()*50+50
    age=random()*50
    aisle=float(randint(1,20))
```

```
    bottlesize=[375.0,750.0,1500.0][randint(0,2)]
    price=wineprice(rating,age)
    price*=(bottlesize/750)
    price*=(random()*0.2+0.9)
    rows.append({'input':(rating,age,aisle,bottlesize),
                 'result':price})
  return rows

def rescale(data,scale):
  scaleddata=[]
  for row in data:
    scaled=[scale[i]*row['input'][i] for i in range(len(scale))]
    scaleddata.append({'input':scaled,'result':row['result']})
  return scaleddata

def createcostfunction(algf,data):
  def costf(scale):
    sdata=rescale(data,scale)
    return crossvalidate(algf,sdata,trials=20)
  return costf

weightdomain=[(0,10)]*4

def wineset3():
  rows=wineset1()
  for row in rows:
    if random()<0.5:
      # Wine was bought at a discount store
      row['result']*=0.6
  return rows

def probguess(data,vec1,low,high,k=5,weightf=gaussian):
  dlist=getdistances(data,vec1)
  nweight=0.0
  tweight=0.0

  for i in range(k):
    dist=dlist[i][0]
    idx=dlist[i][1]
    weight=weightf(dist)
    v=data[idx]['result']

    # Is this point in the range?
    if v>=low and v<=high:
      nweight+=weight
    tweight+=weight
  if tweight==0: return 0

  # The probability is the weights in the range
  # divided by all the weights
  return nweight/tweight

from pylab import *
```

```
def cumulativegraph(data,vec1,high,k=5,weightf=gaussian):
  t1=arange(0.0,high,0.1)
  cprob=array([probguess(data,vec1,0,v,k,weightf) for v in t1])
  plot(t1,cprob)
  show()


def probabilitygraph(data,vec1,high,k=5,weightf=gaussian,ss=5.0):
  # Make a range for the prices
  t1=arange(0.0,high,0.1)

  # Get the probabilities for the entire range
  probs=[probguess(data,vec1,v,v+0.1,k,weightf) for v in t1]

  # Smooth them by adding the gaussian of the nearby probabilites
  smoothed=[]
  for i in range(len(probs)):
    sv=0.0
    for j in range(0,len(probs)):
      dist=abs(i-j)*0.1
      weight=gaussian(dist,sigma=ss)
      sv+=weight*probs[j]
    smoothed.append(sv)
  smoothed=array(smoothed)

  plot(t1,smoothed)
  show()
```

Below code determines the nearest distance through cosine similarity :

Listing 2: nearestEstimate.py

```
import clusters
import numpredict
def findNearestNeighbour(i, data, k):
    testing = data[i]
    neighbors = numpredict.knnestimate(data, testing, k)
    for i in neighbors:
        print(blogs[i])


blogs, text, data = clusters.readfile("blogDataForknn.txt")
for name in "F-Measure", "Web Science and Digital Libraries Research Group":
    for k in 1, 2, 5, 10, 20:
        print("Blog Name", name)
        print("For K", k)
        findNearestNeighbour(blogs.index(name), data, k)
        print("\n\n")
```

Listing 3: knn Result

```
 ('Blog Name', 'F-Measure')
('For K', 1)
SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA



('Blog Name', 'F-Measure')
('For K', 2)
Subterranean Noise
SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA



('Blog Name', 'F-Measure')
('For K', 5)
PSI LAB
Some Call It Noise....
The Jeopardy of Contentment
Subterranean Noise
SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA



('Blog Name', 'F-Measure')
('For K', 10)
Sonology
i'm in too truthful a mood
the fast break of champions
The Music Binge
A layer of chips
PSI LAB
Some Call It Noise....
The Jeopardy of Contentment
Subterranean Noise
SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA



('Blog Name', 'F-Measure')
('For K', 20)
Steel City Rust
Captain Panda's Local & Independent Music Showcase
The Slow Music Movement
DaveCromwell Writes
Abu Everyday
The Run-Out Groove
Rants from the Pants
Broken Biscuit Records
My Life From A to Z
from a voice plantation
Sonology
i'm in too truthful a mood
```

```
the fast break of champions
The Music Binge
A layer of chips
PSI LAB
Some Call It Noise....
The Jeopardy of Contentment
Subterranean Noise
SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA



('Blog Name', 'Web Science and Digital Libraries Research Group')
('For K', 1)
Subterranean Noise



('Blog Name', 'Web Science and Digital Libraries Research Group')
('For K', 2)
Sonology
Subterranean Noise


('Blog Name', 'Web Science and Digital Libraries Research Group')
('For K', 5)
Hip In Detroit
PSI LAB
macthemost
Sonology
Subterranean Noise


('Blog Name', 'Web Science and Digital Libraries Research Group')
('For K', 10)
She May Be Naked
ELLIA TOWNSEND A2
Diagnosis: No Radio
Abu Everyday
Rants from the Pants
Hip In Detroit
PSI LAB
macthemost
Sonology
Subterranean Noise


('Blog Name', 'Web Science and Digital Libraries Research Group')
('For K', 20)
DaveCromwell Writes
Playing Favorites
```

Problem 1 continued on next page...

```
My Life From A to Z
ORGANMYTH
Nothing But Ordinary Glances At Extraordinary Things
Avidd Wallows' Blog
from a voice plantation
Mile In Mine
A2 MEDIA COURSEWORK JOINT BLOG
Alex Denney
She May Be Naked
ELLIA TOWNSEND A2
Diagnosis: No Radio
Abu Everyday
Rants from the Pants
Hip In Detroit
PSI LAB
macthemost
Sonology
Subterranean Noise
```

References

1. https://cmry.github.io/notes/euclidean-v-cosine
2. http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.DistanceMetric.html