6. Text Clustering using KNN

September 22, 2018

Saumay Agrawal
16BCE1151

In [1]:
```
from pprint import pprint
```

In [2]:
```
# Code for the K-Nearest Neighbours Clustering algorithm

        def vectorize(doc,terms):
            docvector=[]
            count=0
            for i in range(len(terms)):
                docvector.append(0)
                for syn in terms[i]:
                    docvector[i]+=doc.lower().split(" ",500).count(syn.lower())
            return docvector

        def calcDist(a,b):
            dist=0
            for x, y in zip(a,b):
                dist+=(x-y)**2
            return round(dist**0.5,4)

        def distInit(mat,n):
            for i in range(n):
                mat.append([0 for j in range(n)])
            return None

        def findMin(distMat):
            minval=float('inf')
            for i in range(len(distMat)):
                for j in range(len(distMat)):
                    if(i==j):
                        continue
                    if(minval>distMat[i][j] and distMat[i][j]!=0):
                        minval=distMat[i][j]
            return minval
```

```
def findMax(distMat):
    maxval=float('-inf')
    for i in range(len(distMat)):
        for j in range(len(distMat)):
            if(i==j):
                continue
            if(maxval<distMat[i][j] and distMat[i][j]!=0):
                maxval=distMat[i][j]
    return maxval

def cluster(index1,index2,nei,dlist,clist):
    found=0
    if nei[index1]!='':
        index2=nei[index1][0]
    else:
        addCluster(index1,dlist,clist)
        return None
    if(clist==[]):
        clist.append([index1,index2])
    else:
        for clus in clist:
            if(index1 in clus or index2 in clus):
                found=1
                if(index2 not in clus):
                    if(nei[index2][1]==nei[index1][1]):
                        clus.append(index2)
                    elif nei[index2][1]>nei[index1][1]:
                        addCluster(index2,dlist,clist)
                if(index1 not in clus):
                    if(nei[index1][1]==nei[index2][1]):
                        clus.append(index1)
                    elif nei[index1][1]>nei[index2][1]:
                        addCluster(index1,dlist,clist)
        if(found==0):
            clist.append([index1,index2])
    return None

def addCluster(index,dlist,clist):
    for clus in clist:
        if index in clus:
            return None
    clist.append([index])
    return None

def vectorAvg(a):
    center=[0 for i in range(len(a[0]))]
    for vec in a:
        center=list(map(sum,zip(center,vec)))
```

```
    n=len(a)
    center[:]=[round(x/n,3) for x in center]
    return center

def nearClustering(docvectors,ite=1):
    print("level: ",ite," clustering")
    distMat=[]
    n=len(docvectors)
    distInit(distMat,n)
    for i in range(n):
        for j in range(i+1,n):
            dist=calcDist(docvectors[i],docvectors[j])
            distMat[i][j]=dist
            distMat[j][i]=dist
    print("\nvectors to cluster:")
    pprint(docvectors)
    groups=[]
    d=(findMin(distMat)+findMax(distMat))/2
    print("\navg  dist=",d)
    for i in range(n):
        groups.append([i])
        for j in range(n):
            if i!=j:
                if distMat[i][j]<=d:
                    groups[i].append(j)
    nei=[]
    for i in range(len(groups)):
        nei.append("")
        mindist=float("inf")
        for j in groups[i]:
            if mindist>distMat[i][j] and i!=j and distMat[i][j]!=0:
                mindist=distMat[i][j]
                nei[i]=(j,mindist)
    clusterlist=[]
    for i in range(n):
        #print("i=",i)
        minval=float('inf')
        for j in range(n):
            if(i==j):
                continue
            if(distMat[i][j]<minval):
                minval=distMat[i][j]
                #print("minval:",minval)
        for j in range(n):
            if(i!=j):
                cluster(i,j,nei,doclist,clusterlist)
    print("\nclusters formed:")
    pprint(clusterlist)
```

```
            newlist=[]
            for clus in clusterlist:
                vec=[docvectors[i] for i in clus]
                center=vectorAvg(vec)
                newlist.append(list(center))
            print("\ncentroid of the new clusters:")
            pprint(newlist)
            print('*'*50)
            if(len(newlist)>1):
                nearClustering(newlist,ite+1)
            return None
```

In [3]:
```
# Code for part 1, clustering of given documents

terms=[['automotive'],['car','cars'],['motorcycles','motorcycle'],['self-drive'],['IoT

doc1='Electric automotive maker Tesla Inc. is likely to introduce its products in India
doc2='Automotive major Mahindra likely to introduce driverless cars'
doc3='BMW plans to introduce its own motorcycles in india'
doc4='Just drive, a self-drive car rental firm uses smart vehicle technology based on I
doc5='Automotive industry going to hire thousands in 2018'
doc6='Famous cricket player  Dhoni brought his priced car Hummer which is an SUV'
doc7='Dhoni led india to its second world cup victory'
doc8='IoT in cars will lead to more safety and make driverless vehicle revolution possi
doc9='Sachin recommended Dhoni for the indian skipper post'

docvectors=[]
doclist=[doc1,doc2,doc3,doc4,doc5,doc6,doc7,doc8,doc9]
for doc in doclist:
    docvectors.append(vectorize(doc,terms))
nearClustering(docvectors)
```

```
level:  1  clustering

vectors to cluster:
[[1, 0, 0, 0, 0, 0, 0],
 [1, 1, 0, 0, 0, 0, 0],
 [0, 0, 1, 0, 0, 0, 0],
 [0, 1, 0, 1, 1, 0, 0],
 [1, 0, 0, 0, 0, 1, 0],
 [0, 1, 0, 0, 0, 0, 1],
 [0, 0, 0, 0, 0, 0, 1],
 [0, 1, 0, 0, 1, 0, 0],
 [0, 0, 0, 0, 0, 0, 1]]

avg  dist= 1.61805

clusters formed:
```

```
[[0, 1, 4], [2], [3, 7], [5, 6, 8]]

centroid of the new clusters:
[[1.0, 0.333, 0.0, 0.0, 0.0, 0.333, 0.0],
 [0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 1.0, 0.0, 0.5, 1.0, 0.0, 0.0],
 [0.0, 0.333, 0.0, 0.0, 0.0, 0.0, 1.0]]
************************************************
level:  2  clustering

vectors to cluster:
[[1.0, 0.333, 0.0, 0.0, 0.0, 0.333, 0.0],
 [0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 1.0, 0.0, 0.5, 1.0, 0.0, 0.0],
 [0.0, 0.333, 0.0, 0.0, 0.0, 0.0, 1.0]]

avg  dist= 1.62785

clusters formed:
[[0, 3, 1], [2]]

centroid of the new clusters:
[[0.333, 0.222, 0.333, 0.0, 0.0, 0.111, 0.333],
 [0.0, 1.0, 0.0, 0.5, 1.0, 0.0, 0.0]]
************************************************
level:  3  clustering

vectors to cluster:
[[0.333, 0.222, 0.333, 0.0, 0.0, 0.111, 0.333],
 [0.0, 1.0, 0.0, 0.5, 1.0, 0.0, 0.0]]

avg  dist= 1.4833

clusters formed:
[[0, 1]]

centroid of the new clusters:
[[0.167, 0.611, 0.167, 0.25, 0.5, 0.056, 0.167]]
************************************************


In [4]: # Code for text minining from web for part 2

        from bs4 import BeautifulSoup
        import requests
        from string import punctuation

        doclist=['doc1.txt','doc2.txt','doc3.txt','doc4.txt','doc5.txt','doc6.txt','doc7.txt',
```

```
        links=['https://www.zigwheels.com/newcars/Tesla',' https://www.financialexpress.com/au
               ' https://en.wikipedia.org/wiki/Toyota_Prius','https://economictimes.indiatimes
               ' https://indianexpress.com/article/india/india-news-india/demonetisation-hits-
               ' https://www.livemint.com/Politics/ySbMKTIC4MINsz1btccBJO/How-demonetisation-a
               ' https://inc42.com/buzz/electric-vehicles-this-week-centre-reduces-gst-on-lith
               'https://www.youthkiawaaz.com/2017/12/impact-of-demonetisation-on-the-indian-ec
               ' https://www.news18.com/news/business/how-gst-will-curb-tax-evasion-1446035.ht

        for i in range(len(doclist)):
            page=requests.get(links[i])
            soup=BeautifulSoup(page.text,'html.parser')
            p_tags=soup.find_all('p')
            text = (''.join(s.findAll(text=True))for s in soup.findAll('p'))
            f=open(doclist[i],'w')
            #f.write("abc")
            gen=[str(y.lower()) for y in text ]
            count=0
            for t in gen:
                f.write(t)
                count=count+1
                if count>=500:
                    break
            f.close()
```

In [5]:
```
# Code for the clustering of mined text in part 2

        files = ['doc1.txt', 'doc2.txt', 'doc3.txt', 'doc4.txt','doc5.txt', 'doc6.txt', 'doc7.t
        terms=[['tesla',"tesla's"], ['electric'], ['car','cars','vehicle','vehicles','automobi
        docvectors=[]
        for fname in files:
            file=open(fname,'r')
            doclines=file.read().split('.')
            doc=''
            for line in doclines:
                doc+=" "+str(line)
            docvectors.append(vectorize(doc,terms))
        nearClustering(docvectors)
```

```
level:  1  clustering

vectors to cluster:
[[5, 4, 3, 0, 0, 0, 0],
 [0, 22, 13, 0, 0, 0, 0],
 [0, 1, 7, 0, 0, 0, 0],
 [0, 3, 3, 0, 0, 0, 0],
 [0, 5, 2, 0, 1, 0, 0],
 [0, 0, 1, 0, 6, 0, 0],
 [0, 0, 4, 0, 0, 14, 0],
```

```
 [0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 4, 0]]

avg  dist= 15.017850000000001

clusters formed:
[[0, 3, 4], [1], [2], [5, 7, 11, 8, 9, 10], [6, 11, 7, 8, 9, 10]]

centroid of the new clusters:
[[1.667, 4.0, 2.667, 0.0, 0.333, 0.0, 0.0],
 [0.0, 22.0, 13.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 1.0, 7.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 0.0, 0.167, 0.0, 1.0, 0.667, 0.0],
 [0.0, 0.0, 0.667, 0.0, 0.0, 3.0, 0.0]]
************************************************
level:  2  clustering

vectors to cluster:
[[1.667, 4.0, 2.667, 0.0, 0.333, 0.0, 0.0],
 [0.0, 22.0, 13.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 1.0, 7.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 0.0, 0.167, 0.0, 1.0, 0.667, 0.0],
 [0.0, 0.0, 0.667, 0.0, 0.0, 3.0, 0.0]]

avg  dist= 14.042399999999999

clusters formed:
[[0, 3, 4], [1], [2]]

centroid of the new clusters:
[[0.556, 1.333, 1.167, 0.0, 0.444, 1.222, 0.0],
 [0.0, 22.0, 13.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 1.0, 7.0, 0.0, 0.0, 0.0, 0.0]]
************************************************
level:  3  clustering

vectors to cluster:
[[0.556, 1.333, 1.167, 0.0, 0.444, 1.222, 0.0],
 [0.0, 22.0, 13.0, 0.0, 0.0, 0.0, 0.0],
 [0.0, 1.0, 7.0, 0.0, 0.0, 0.0, 0.0]]

avg  dist= 14.93395

clusters formed:
[[0, 2], [1]]
```

```
centroid of the new clusters:
[[0.278, 1.167, 4.083, 0.0, 0.222, 0.611, 0.0],
 [0.0, 22.0, 13.0, 0.0, 0.0, 0.0, 0.0]]
**************************************************
level:  4  clustering

vectors to cluster:
[[0.278, 1.167, 4.083, 0.0, 0.222, 0.611, 0.0],
 [0.0, 22.0, 13.0, 0.0, 0.0, 0.0, 0.0]]

avg  dist= 22.6722

clusters formed:
[[0, 1]]

centroid of the new clusters:
[[0.139, 11.584, 8.541, 0.0, 0.111, 0.305, 0.0]]
**************************************************
```