

# knn cosine

October 15, 2018

## 1 Nearest Neighbour Clustering using cosine similarity

16BCE1259

Shushil Kumar Ravishankar

```
In [1]: from pprint import pprint
import numpy as np
```

```
In [4]: 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
    a=np.array(a)
    b=np.array(b)
    return round(1-np.dot(a,b)/(np.linalg.norm(a)*np.linalg.norm(b)),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,doclust,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 [5]: 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
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 poss
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= 0.59175

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= 0.8947

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

centroid of the new clusters:
[[0.333, 0.555, 0.0, 0.167, 0.333, 0.111, 0.333],
 [0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0]]
*****
level: 3  clustering

vectors to cluster:
[[0.333, 0.555, 0.0, 0.167, 0.333, 0.111, 0.333],
 [0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0]]

avg  dist= 1.0

clusters formed:
[[0, 1]]

centroid of the new clusters:
[[0.167, 0.278, 0.5, 0.084, 0.167, 0.056, 0.167]]
*****

```