```python
import numpy as np
import nltk
from nltk.corpus import stopwords
from mpi4py import MPI
import os
import codecs
import re
from os import listdir
from os.path import isfile, join
from sklearn.feature_extraction.text import CountVectorizer
```

```python
comm = MPI.COMM_WORLD
worker_id = comm.Get_rank()
p = comm.Get_size()
```

```python
def cleaner_tokenizer(docs):
    l=[]
    line_all=[]
    for j in range(0,len(docs)):
        text=codecs.open(str(docs[j]),encoding="Latin-1")       #read docs
        line_all=""
        for line in text:
            line_all = line_all + " " + line
        line_all=re.sub("[^A-Za-z]+", " ", line_all)            #remove only character
        line_all=re.sub(r'\s+[a-zA-Z]\s+', ' ', line_all)       #multiple space to single
        line_all=line_all.rstrip()                              #remove space on right side
        l+=[line_all]
    return l
    text.close()
```

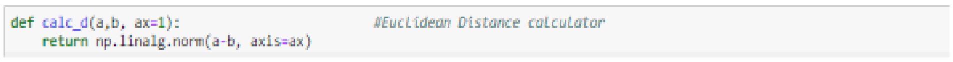
```python
def reader(directory):
    os.chdir(directory)
    fnames=[]
    l=[]
    line=[]
    for r,d,f in os.walk("."):
        for fname in d:                                         #folders in 20newsgroups
            fnames+=[fname]
    for i in range(len(fnames)):
        docs=[]
        path=(directory+str(fnames[i]))
        os.chdir(path)
        docs+=[f for f in listdir(path) if isfile(join(path, f))]    #list from documents in folder
        N=len(docs)
        group=round(N/p)
        if worker_id!=0:
            docs=comm.recv(source=0, tag=1)
            docs=docs[(group*worker_id):(group*(1+worker_id))]       #each worker assigned to task other than worker 0
            line2=cleaner_tokenizer(docs)
            comm.send(line2,dest=0)                                  #send to worker 0
        else:
            for i in range(1,p):
                comm.send(docs,dest=i,tag=1)                     #send corresponding documents to each worker

            docs = docs[(group*worker_id):(group*(1+worker_id))]
            line=cleaner_tokenizer(docs)
            line2 = np.zeros(group)
            for i in range(1,p):
                line2 = comm.recv(source=i)
                line=np.append(line,line2)                      #append results
        l=np.append(l,line)
    return l
```

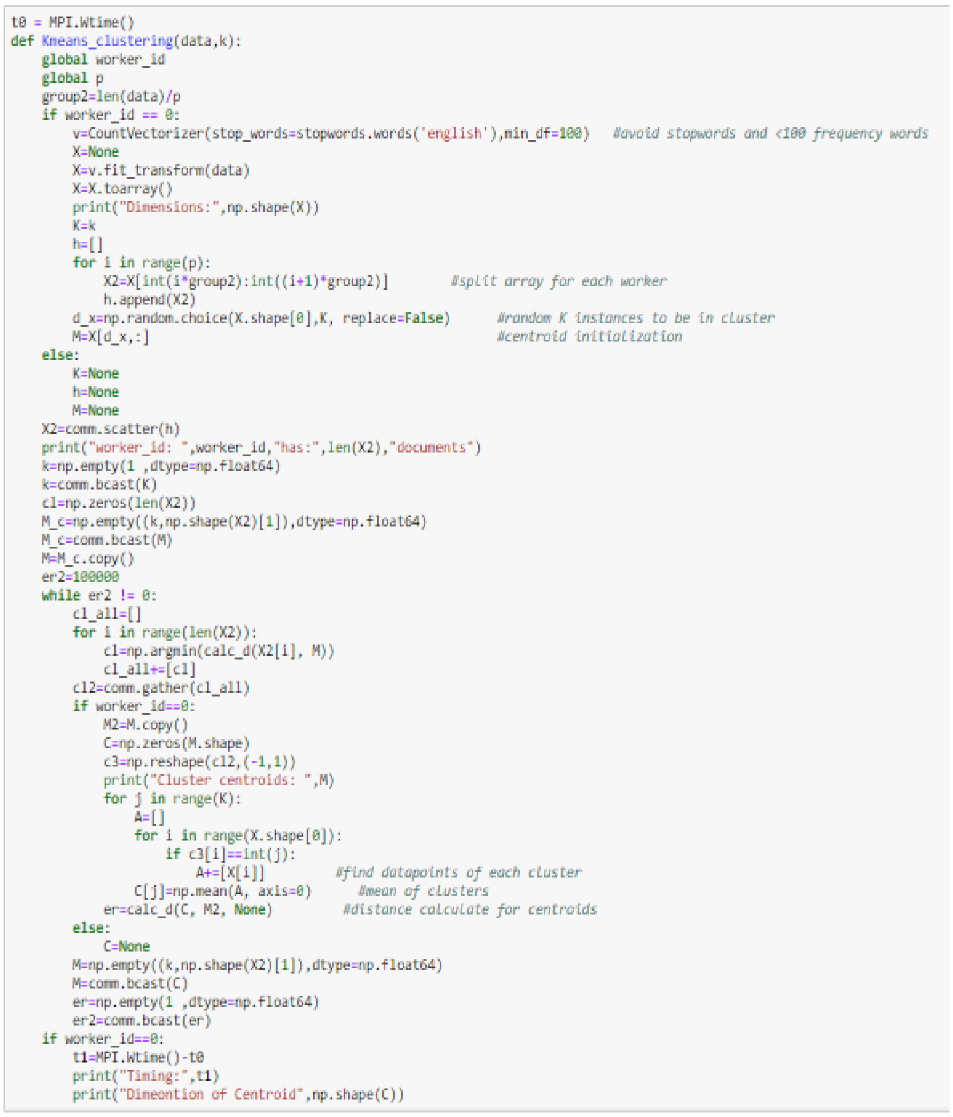
```python
os.chdir("C:/Users/user")
data_path=os.path.join(os.getcwd()+"/20_newsgroups/")
data=reader(data_path)
```

Path is assigned. 19997 documents in 20 folders are read in lines.

K-means Clustering

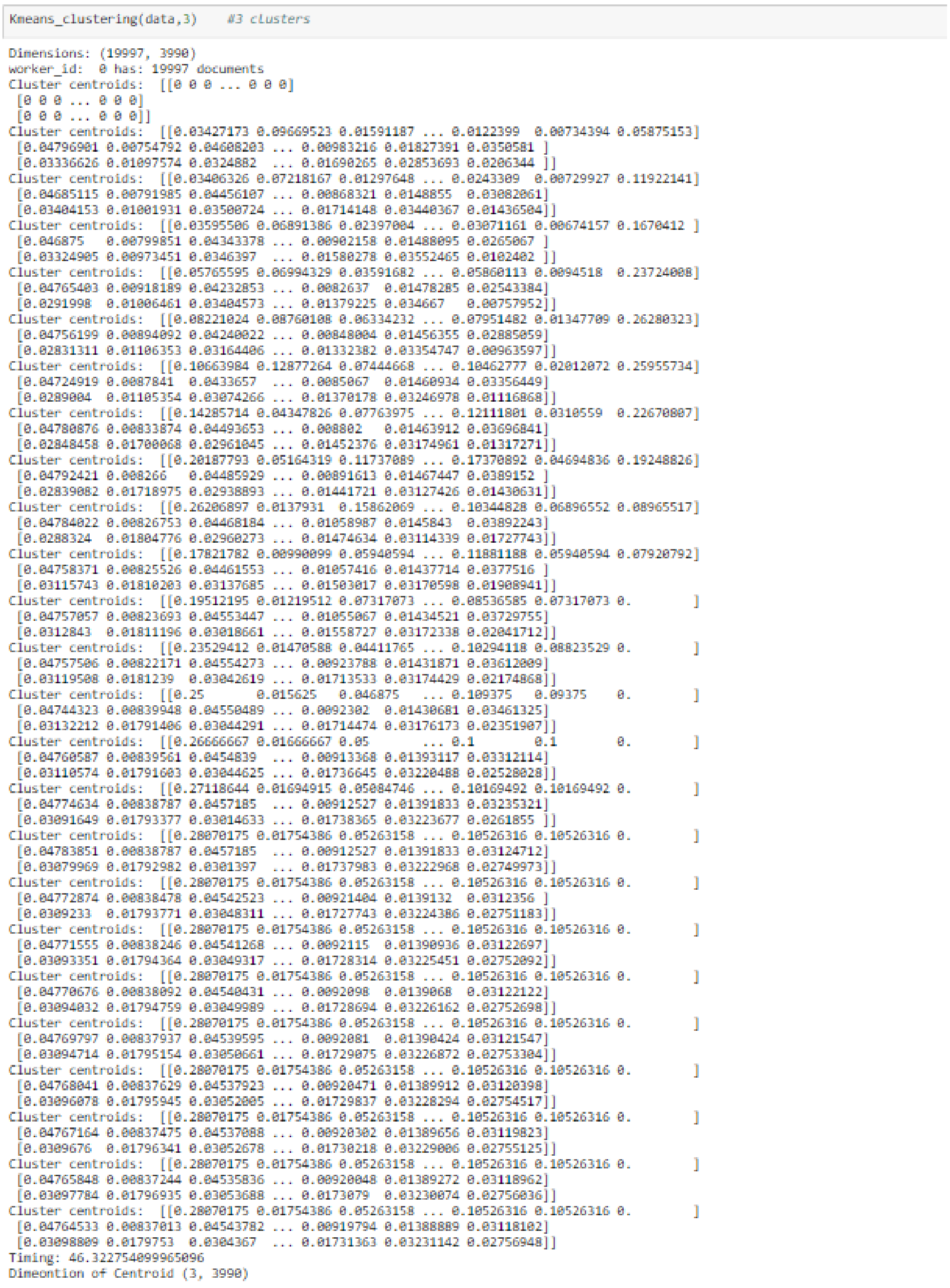


```python
def calc_d(a,b, ax=1):                          #Euclidean Distance calculator
    return np.linalg.norm(a-b, axis=ax)
```



```python
t0 = MPI.Wtime()
def Kmeans_clustering(data,k):
    global worker_id
    global p
    group2=len(data)/p
    if worker_id == 0:
        v=CountVectorizer(stop_words=stopwords.words('english'),min_df=100)   #avoid stopwords and <100 frequency words
        X=None
        X=v.fit_transform(data)
        X=X.toarray()
        print("Dimensions:",np.shape(X))
        K=k
        h=[]
        for i in range(p):
            X2=X[int(i*group2):int((i+1)*group2)]        #split array for each worker
            h.append(X2)
        d_x=np.random.choice(X.shape[0],K, replace=False)      #random K instances to be in cluster
        M=X[d_x,:]                                             #centroid initialization
    else:
        K=None
        h=None
        M=None
    X2=comm.scatter(h)
    print("worker_id: ",worker_id,"has:",len(X2),"documents")
    k=np.empty(1 ,dtype=np.float64)
    k=comm.bcast(K)
    cl=np.zeros(len(X2))
    M_c=np.empty((k,np.shape(X2)[1]),dtype=np.float64)
    M_c=comm.bcast(M)
    M=M_c.copy()
    er2=100000
    while er2 != 0:
        cl_all=[]
        for i in range(len(X2)):
            cl=np.argmin(calc_d(X2[i], M))
            cl_all+=[cl]
        cl2=comm.gather(cl_all)
        if worker_id==0:
            M2=M.copy()
            C=np.zeros(M.shape)
            c3=np.reshape(cl2,(-1,1))
            print("Cluster centroids: ",M)
            for j in range(K):
                A=[]
                for i in range(X.shape[0]):
                    if c3[i]==int(j):
                        A+=[X[i]]         #find datapoints of each cluster
                C[j]=np.mean(A, axis=0)      #mean of clusters
            er=calc_d(C, M2, None)         #distance calculate for centroids
        else:
            C=None
        M=np.empty((k,np.shape(X2)[1]),dtype=np.float64)
        M=comm.bcast(C)
        er=np.empty(1 ,dtype=np.float64)
        er2=comm.bcast(er)
    if worker_id==0:
        t1=MPI.Wtime()-t0
        print("Timing:",t1)
        print("Dimeontion of Centroid",np.shape(C))
```

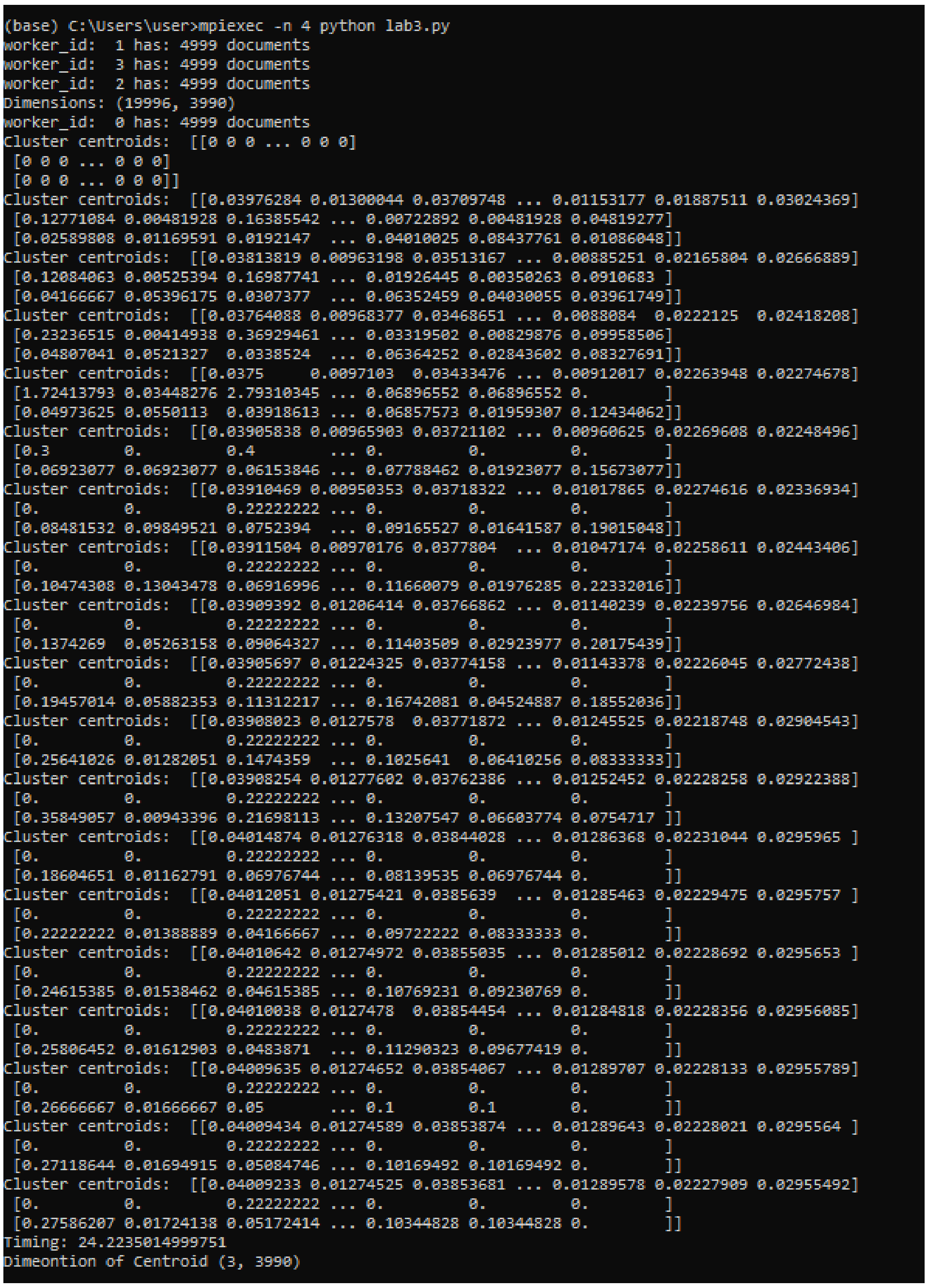
Result of example without additional workers and 3 clusters



```
Kmeans_clustering(data,3)    #3 clusters
```

```
Dimensions: (19997, 3990)
worker_id:  0 has: 19997 documents
Cluster centroids:  [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
Cluster centroids:  [[0.03427173 0.09669523 0.01591187 ... 0.0122399  0.00734394 0.05875153]
 [0.04796901 0.00754792 0.04608203 ... 0.00983216 0.01827391 0.0350581 ]
 [0.03336626 0.01097574 0.0324882  ... 0.01690265 0.02853693 0.0206344 ]]
Cluster centroids:  [[0.03406326 0.07218167 0.01297648 ... 0.0243309  0.00729927 0.11922141]
 [0.04685115 0.00791985 0.04456107 ... 0.00868321 0.0148855  0.03082061]
 [0.03404153 0.01001931 0.03500724 ... 0.01714148 0.03440367 0.01436504]]
Cluster centroids:  [[0.03595506 0.06891386 0.02397004 ... 0.03071161 0.00674157 0.1670412 ]
 [0.046875   0.00799851 0.04343378 ... 0.00902158 0.01488095 0.0265067 ]
 [0.03324905 0.00973451 0.0346397  ... 0.01580278 0.03552465 0.0102402 ]]
Cluster centroids:  [[0.05765595 0.06994329 0.03591682 ... 0.05860113 0.0094518  0.23724008]
 [0.04765403 0.00918189 0.04232853 ... 0.0082637  0.01478285 0.02543384]
 [0.0291998  0.01006461 0.03404573 ... 0.01379225 0.034667   0.00757952]]
Cluster centroids:  [[0.08221024 0.08760108 0.06334232 ... 0.07951482 0.01347709 0.26280323]
 [0.04756199 0.00894092 0.04240022 ... 0.00848004 0.01456355 0.02885059]
 [0.02831311 0.01106353 0.03164406 ... 0.01332382 0.03354747 0.00963597]]
Cluster centroids:  [[0.10663984 0.12877264 0.07444668 ... 0.10462777 0.02012072 0.25955734]
 [0.04724919 0.0087841  0.0433657  ... 0.0085067  0.01460934 0.03356449]
 [0.0289004  0.01105354 0.03074266 ... 0.01370178 0.03246978 0.01116868]]
Cluster centroids:  [[0.14285714 0.04347826 0.07763975 ... 0.12111801 0.0310559  0.22670807]
 [0.04780876 0.00833874 0.04493653 ... 0.008802   0.01463912 0.03696841]
 [0.02848458 0.01700068 0.02961045 ... 0.01452376 0.03174961 0.01317271]]
Cluster centroids:  [[0.20187793 0.05164319 0.11737089 ... 0.17370892 0.04694836 0.19248826]
 [0.04792421 0.008266   0.04485929 ... 0.00891613 0.01467447 0.0389152 ]
 [0.02839082 0.01718975 0.02938893 ... 0.01441721 0.03127426 0.01430631]]
Cluster centroids:  [[0.26206897 0.0137931  0.15862069 ... 0.10344828 0.06896552 0.08965517]
 [0.04784022 0.00826753 0.04468184 ... 0.01058987 0.0145843  0.03892243]
 [0.0288324  0.01804776 0.02960273 ... 0.01474634 0.03114339 0.01727743]]
Cluster centroids:  [[0.17821782 0.00990099 0.05940594 ... 0.11881188 0.05940594 0.07920792]
 [0.04758371 0.00825526 0.04461553 ... 0.01057416 0.01437714 0.0377516 ]
 [0.03115743 0.01810203 0.03137685 ... 0.01503017 0.03170598 0.01908941]]
Cluster centroids:  [[0.19512195 0.01219512 0.07317073 ... 0.08536585 0.07317073 0.        ]
 [0.04757057 0.00823693 0.04553447 ... 0.01055067 0.01434521 0.03729755]
 [0.0312843  0.01811196 0.03018661 ... 0.01558727 0.03172338 0.02041712]]
Cluster centroids:  [[0.23529412 0.01470588 0.04411765 ... 0.10294118 0.08823529 0.        ]
 [0.04757506 0.00822171 0.04554273 ... 0.00923788 0.01431871 0.03612009]
 [0.03119508 0.0181239  0.03042619 ... 0.01713533 0.03174429 0.02174868]]
Cluster centroids:  [[0.25       0.015625   0.046875   ... 0.109375   0.09375    0.        ]
 [0.04744323 0.00839948 0.04550489 ... 0.0092302  0.01430681 0.03461325]
 [0.03132212 0.01791406 0.03044291 ... 0.01714474 0.03176173 0.02351907]]
Cluster centroids:  [[0.26666667 0.01666667 0.05       ... 0.1        0.1        0.        ]
 [0.04760587 0.00839561 0.0454839  ... 0.00913368 0.01393117 0.03312114]
 [0.03110574 0.01791603 0.03044625 ... 0.01736645 0.03220488 0.02528028]]
Cluster centroids:  [[0.27118644 0.01694915 0.05084746 ... 0.10169492 0.10169492 0.        ]
 [0.04774634 0.00838787 0.0457185  ... 0.00912527 0.01391833 0.03235321]
 [0.03091649 0.01793377 0.03014633 ... 0.01738365 0.03223677 0.0261855 ]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04783851 0.00838787 0.0457185  ... 0.00912527 0.01391833 0.03124712]
 [0.03079969 0.01792982 0.0301397  ... 0.01737983 0.03222968 0.02749973]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04772874 0.00838478 0.04542523 ... 0.00921404 0.0139132  0.0312356 ]
 [0.0309233  0.01793771 0.03048311 ... 0.01727743 0.03224386 0.02751183]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04771555 0.00838246 0.04541268 ... 0.0092115  0.01390936 0.03122697]
 [0.03093351 0.01794364 0.03049317 ... 0.01728314 0.03225451 0.02752092]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04770676 0.00838092 0.04540431 ... 0.0092098  0.0139068  0.03122122]
 [0.03094032 0.01794759 0.03049989 ... 0.01728694 0.03226162 0.02752698]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04769797 0.00837937 0.04539595 ... 0.0092081  0.01390424 0.03121547]
 [0.03094714 0.01795154 0.03050661 ... 0.01729075 0.03226872 0.02753304]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04768041 0.00837629 0.04537923 ... 0.00920471 0.01389912 0.03120398]
 [0.03096078 0.01795945 0.03052005 ... 0.01729837 0.03228294 0.02754517]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04767164 0.00837475 0.04537088 ... 0.00920302 0.01389656 0.03119823]
 [0.0309676  0.01796341 0.03052678 ... 0.01730218 0.03229006 0.02755125]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04765848 0.00837244 0.04535836 ... 0.00920048 0.01389272 0.03118962]
 [0.03097784 0.01796935 0.03053688 ... 0.0173079  0.03230074 0.02756036]]
Cluster centroids:  [[0.28070175 0.01754386 0.05263158 ... 0.10526316 0.10526316 0.        ]
 [0.04764533 0.00837013 0.04543782 ... 0.00919794 0.01388889 0.03118102]
 [0.03098809 0.0179753  0.0304367  ... 0.01731363 0.03231142 0.02756948]]
Timing: 46.322754099965096
Dimeontion of Centroid (3, 3990)
```

Result of example with 4 workers and 3 clusters



```
(base) C:\Users\user>mpiexec -n 4 python lab3.py
worker_id:  1 has: 4999 documents
worker_id:  3 has: 4999 documents
worker_id:  2 has: 4999 documents
Dimensions: (19996, 3990)
worker_id:  0 has: 4999 documents
Cluster centroids:  [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
Cluster centroids:  [[0.03976284 0.01300044 0.03709748 ... 0.01153177 0.01887511 0.03024369]
 [0.12771084 0.00481928 0.16385542 ... 0.00722892 0.00481928 0.04819277]
 [0.02589808 0.01169591 0.0192147  ... 0.04010025 0.08437761 0.01086048]]
Cluster centroids:  [[0.03813819 0.00963198 0.03513167 ... 0.00885251 0.02165804 0.02666889]
 [0.12084063 0.00525394 0.16987741 ... 0.01926445 0.00350263 0.0910683 ]
 [0.04166667 0.05396175 0.0307377  ... 0.06352459 0.04030055 0.03961749]]
Cluster centroids:  [[0.03764088 0.00968377 0.03468651 ... 0.0088084  0.0222125  0.02418208]
 [0.23236515 0.00414938 0.36929461 ... 0.03319502 0.00829876 0.09958506]
 [0.04807041 0.0521327  0.0338524  ... 0.06364252 0.02843602 0.08327691]]
Cluster centroids:  [[0.0375     0.0097103  0.03433476 ... 0.00912017 0.02263948 0.02274678]
 [1.72413793 0.03448276 2.79310345 ... 0.06896552 0.06896552 0.        ]
 [0.04973625 0.0550113  0.03918613 ... 0.06857573 0.01959307 0.12434062]]
Cluster centroids:  [[0.03905838 0.00965903 0.03721102 ... 0.00960625 0.02269608 0.02248496]
 [0.3        0.         0.4        ... 0.         0.         0.        ]
 [0.06923077 0.06923077 0.06153846 ... 0.07788462 0.01923077 0.15673077]]
Cluster centroids:  [[0.03910469 0.00950353 0.03718322 ... 0.01017865 0.02274616 0.02336934]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.08481532 0.09849521 0.0752394  ... 0.09165527 0.01641587 0.19015048]]
Cluster centroids:  [[0.03911504 0.00970176 0.0377804  ... 0.01047174 0.02258611 0.02443406]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.10474308 0.13043478 0.06916996 ... 0.11660079 0.01976285 0.22332016]]
Cluster centroids:  [[0.03909392 0.01206414 0.03766862 ... 0.01140239 0.02239756 0.02646984]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.1374269  0.05263158 0.09064327 ... 0.11403509 0.02923977 0.20175439]]
Cluster centroids:  [[0.03905697 0.01224325 0.03774158 ... 0.01143378 0.02226045 0.02772438]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.19457014 0.05882353 0.11312217 ... 0.16742081 0.04524887 0.18552036]]
Cluster centroids:  [[0.03908023 0.0127578  0.03771872 ... 0.01245525 0.02218748 0.02904543]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.25641026 0.01282051 0.1474359  ... 0.1025641  0.06410256 0.08333333]]
Cluster centroids:  [[0.03908254 0.01277602 0.03762386 ... 0.01252452 0.02228258 0.02922388]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.35849057 0.00943396 0.21698113 ... 0.13207547 0.06603774 0.0754717 ]]
Cluster centroids:  [[0.04014874 0.01276318 0.03844028 ... 0.01286368 0.02231044 0.0295965 ]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.18604651 0.01162791 0.06976744 ... 0.08139535 0.06976744 0.        ]]
Cluster centroids:  [[0.04012051 0.01275421 0.0385639  ... 0.01285463 0.02229475 0.0295757 ]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.22222222 0.01388889 0.04166667 ... 0.09722222 0.08333333 0.        ]]
Cluster centroids:  [[0.04010642 0.01274972 0.03855035 ... 0.01285012 0.02228692 0.0295653 ]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.24615385 0.01538462 0.04615385 ... 0.10769231 0.09230769 0.        ]]
Cluster centroids:  [[0.04010038 0.0127478  0.03854454 ... 0.01284818 0.02228356 0.02956085]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.25806452 0.01612903 0.0483871  ... 0.11290323 0.09677419 0.        ]]
Cluster centroids:  [[0.04009635 0.01274652 0.03854067 ... 0.01289707 0.02228133 0.02955789]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.26666667 0.01666667 0.05       ... 0.1        0.1        0.        ]]
Cluster centroids:  [[0.04009434 0.01274589 0.03853874 ... 0.01289643 0.02228021 0.0295564 ]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.27118644 0.01694915 0.05084746 ... 0.10169492 0.10169492 0.        ]]
Cluster centroids:  [[0.04009233 0.01274525 0.03853681 ... 0.01289578 0.02227909 0.02955492]
 [0.         0.         0.22222222 ... 0.         0.         0.        ]
 [0.27586207 0.01724138 0.05172414 ... 0.10344828 0.10344828 0.        ]]
Timing: 24.2235014999751
Dimeontion of Centroid (3, 3990)
```

Explanation:

First, Euclidean distance calculator function is defined for clustering. Timing set from beginning of the clustering to the end. Less than 100 occurrence words and stopwords are avoided by CountVectorizer and in 19997 document and we have 3990 words. Master node Worker 0 assigned each worker part of task by 19997/number of workers using comm.scatter(). After that, initial centroids are assigned through comm.bcast() from X matrix for a given number of k clusters. Then, comm.gather() is used to send completed tasks to Worker 0, so that results are appended by Worker 0 and get new cluster centroids with the help of comm.bcast() in order to find centroid values for a given number of clusters. In the example, increasing number of workers from 1 to 4 decreased operation time. I would expect timing to increase after few additional workers because communication between workers also takes time.