Name:Samer Attrah
Email: s.attrah@student.han.nl
student number: 2119231

K-means assignment:

Starting with implementing the K-means function in a separate file titled “k-means.m”,
the approach that will be used to initialize the centroids is to divide the number of events by the number of clusters, and choose points from the data as many as the number of clusters, there indices separated by the number resulted from the division. So it is not random initialization.
the function need to include the following elements:
- first initializing some variables and arrays, such as:
    1. number of the events to be clustered.
    2. the number of points separating each one of the centroids.
    3. An array to save the centroids in later
    4. the “clustered” array to be returned in the end of the function, and it includes the events with a column has there clusters

- after initializing variables, now initialize the centroids, create a loop over the number of clusters, to assign each cluster an event, to be it’s centroid and as the criteria described above, the loop will be similar to the following:

```
for c = 1 : number of clusters
        index = (length(events) / number of clusters) * c
        mean(c,:) = events(index,:)
```

- now the centroids are initialized, loop through the events and assign each event to the centroid closest in euclidean distance to it, and that is by finding the euclidean distance using the MATLAB function (pdist2) from each point in of the events to each one of the centroids and assigning the event to the centroid closest to. And the code will be like what follows:

```
for I = 1:length of the events set
        [centroid, index of the centroid] = pdist2(mean, first two columns of clustered events
        matrix, ‘euclidean’, ‘smallest’, 1)
        clustered events(i) = index of the centroid
```

Note: using the (pdist2) to save time when running the function, since other wise there will be another loop added to the function.
- after assigning the events to the centroids and creating the clusters, now move each one of the centroids to the center of there clusters. And that happens by finding the average position for each of the points assigned to the cluster and updating the centroid of the cluster with it.

after running the algorithm described above the result will probably not be the best possible because the initialization of the centroids is happening once and the updating of them is also happening once. But to get better clustering happens by trying different initializations and finding the cost for each one of the clusters and updating the centroids to give the lowest cost possible. And maybe should try random initialization for the centroids.

Also this implementation takes relatively too long to finish, so it might be possible to save some time by vectorizing.

The result from running the algorithm was as shown in the following figure: