Ioana-Raluca Tiriac
ASU ID : 1217060520
Email : itiriac@asu.edu

Project 2 : K-means Clustering

CSE 575: Statistical Machine Learning

Arizona State University

Fall 2021

1. K-means using Strategy 1

In Strategy 1 we needed to pick the initial centroids randomly from the samples by providing the last four digits of our ID – "0520". *Precode.py* already implemented picking the centers randomly for cluster size k =3 and k =5. My implementation extends *Precode.py* by another method *"rest_of_k_and_Centroids()"* that returns a set of randomly chosen initial centroids for the cases of the cluster size being k=2,4,6,7,8,9,10.

The main k-means algorithm iterates over all 2D data points in a loop until the stopping condition is reached and the centroids don't change anymore, calculating for each 2D data sample its distances to each of the current centroids. The index of the minimum distance, which designates the cluster the data belongs to, is stored in a data membership array, to mark which cluster the data point is closest to. In a further step based on the data membership array, the cluster arrays are created holding each centroid and the data samples closest to it. All the cluster arrays are stored in the main "clusters" array. Next the *"update_centroids()"* function generates the new centroids of the previously formed clusters by calculating the mean of the data in each cluster. Then the new centroids are compared with the old centroids by using the *compare_centroids(oldCentroids, newCentroids)* function and if they coincide then the stopping condition of the algorithm is reached and the total loss (or cost) is computed and stored. If the new centroids don't coincide with the old centroids, then the iteration over all data samples starts again and the distances to the updated centroids are recalculated.

2. Results of K-means with Strategy1

Using number of clusters k = 3 and initial centroids =

```
[[ 2.38952606, 7.22195564]
 [ 4.59083727  7.53490523]
 [ 1.3483716   3.96379638]]
```

the k-means algorithm converges after 12 iterations and stops at the following final centroids =

```
[[ 2.56146449  6.08861338]
 [ 6.49724962  7.52297293]
 [ 5.47740039  2.25498103]]
```

The computed loss fct. for k = 3 using the formula *(x0 – c0)^2 + (x1 – c1)^2* is `1293.77745239`.

Using number of clusters k = 5 and initial centroids =

```
[[ 5.77144223  9.04075394]
 [ 1.96633923  7.30845038]
 [ 2.97097541  2.39669382]
 [ 5.36626615  6.51434231]
 [ 7.93432052  8.17735191]]
```

the k-means algorithm converged after 8 iterations and stops at the following final centroids =

```
[[ 5.40252508  6.73636175]
 [ 2.60123296  6.91610506]
 [ 3.21257461  2.49658087]
 [ 7.25262683  2.40015826]
 [ 7.75648325  8.55668928]]
```

The computed loss fct. for k = 5 is `613.282439206.`

For k=2,4,6,7,8,9,10 the results are :

```
Number of clusters: 2
Initial centroids are:  [[ 4.99874427  2.87525327]
 [ 4.32239695  0.33088885]]
Loss function:  1921.03348586
Final centroids are:  [[ 4.85261193  7.27164171]
 [ 5.00056234  2.48542748]]
Iterations:  6


Number of clusters: 4
Initial centroids are:  [[ 4.75184863  4.20214023]
 [ 7.45225989  2.26860809]
 [ 2.0614632   8.22584366]
 [ 6.5807212  -0.0766824 ]]
Loss function:  789.237972218
Final centroids are:  [[ 2.90547741  6.90512276]
 [ 7.25262683  2.40015826]
 [ 6.62592538  7.57614917]
 [ 3.22853009  2.52404863]]
Iterations:  16


Number of clusters: 5
Initial centroids are:  [[ 5.77144223  9.04075394]
 [ 1.96633923  7.30845038]
 [ 2.97097541  2.39669382]
 [ 5.36626615  6.51434231]
 [ 7.93432052  8.17735191]]
Loss function:  613.282439206
Final centroids are:  [[ 5.40252508  6.73636175]
 [ 2.60123296  6.91610506]
 [ 3.21257461  2.49658087]
 [ 7.25262683  2.40015826]
 [ 7.75648325  8.55668928]]
Iterations:  8
```

Number of clusters: 6

```
Initial Centroids are:  [[ 2.69511302  5.93967352]
 [ 5.52279832  5.52162016]
 [ 7.59731342  1.16504743]
 [ 8.03150205  8.88381354]
 [ 1.76496239  6.98004057]
 [ 8.22627485  2.26048701]]
Loss function:  476.296570527
```

```
Final centroids are:  [[ 3.502455    3.62870476]
 [ 5.46427736  6.83771354]
 [ 3.14506148  0.90770655]
 [ 7.75648325  8.55668928]
 [ 2.52382885  7.02897469]
 [ 7.41419243  2.32169114]]
Iterations:  11

Number of clusters: 7
Initial Centroids are:  [[ 3.75004647  4.90070114]
 [ 2.10606162  8.23183769]
 [ 2.81629029  3.1999725 ]
 [ 4.30228618  7.08489147]
 [ 2.69511302  5.93967352]
 [ 3.81485895  6.91844078]
 [ 3.04101702 -0.36138487]]
Loss function:  367.665846495
Final centroids are:  [[ 7.55616782  2.23516796]
 [ 2.56333815  6.9782248 ]
 [ 2.24204752  3.25100749]
 [ 7.75648325  8.55668928]
 [ 4.86813713  3.71934185]
 [ 5.46427736  6.83771354]
 [ 3.16906145  0.81432515]]
Iterations:  16
```

Number of clusters: 8

```
Initial Centroids are:  [[ 7.15364076  2.61344894]
 [ 5.2979492   3.65258141]
 [ 5.07631894  3.30296197]
 [ 4.78363211  7.10644288]
 [ 5.30543981  3.39751664]
 [ 4.21807424  4.26660054]
 [ 1.89256383  3.05142539]
 [ 8.07641652  9.27162002]]
Loss function:  289.932726045
Final centroids are:  [[ 7.55616782  2.23516796]
 [ 6.15468228  5.70140721]
 [ 3.16906145  0.81432515]
 [ 4.85939875  7.94163821]
 [ 4.81833058  3.6950232 ]
 [ 2.53650108  6.85941978]
 [ 2.24204752  3.25100749]
 [ 7.91430998  8.51990981]]
Iterations:  12
```

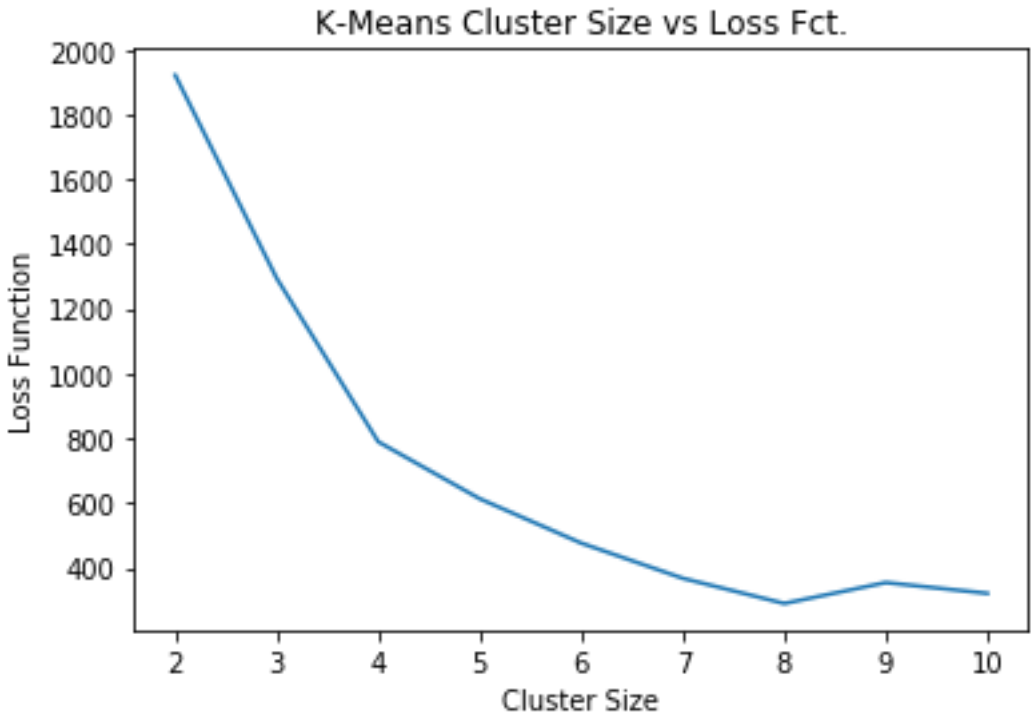
Number of clusters: 9

```
Initial Centroids are:  [[ 7.33424973  2.97894225]
 [ 4.21807424  4.26660054]
 [ 1.76496239  6.98004057]
 [ 1.20162248  7.68639714]
 [ 2.95147442  7.76615605]
 [ 2.64683045  6.32344268]
 [ 7.95300821  3.1028738 ]
 [ 4.66005931  7.06059555]
 [ 3.2492998   5.59125171]]
Loss function:  354.443495643
Final centroids are:  [[ 6.86875852  1.67600681]
```

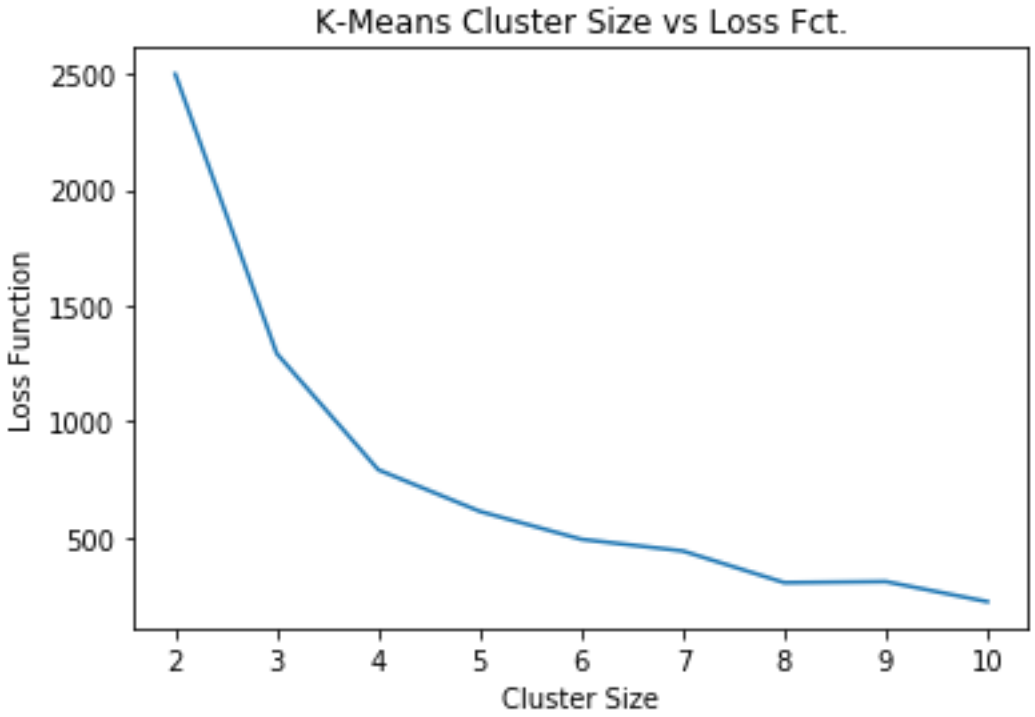
```
 [ 3.14506148  0.90770655]
 [ 3.13834768  5.93372322]
 [ 1.94193284  7.27196866]
 [ 2.5729775   8.40072877]
 [ 5.43207068  6.86930884]
 [ 7.97671279  3.20722346]
 [ 7.75648325  8.55668928]
 [ 3.44436605  3.49453386]]
Iterations:  9
```

Number of clusters: 10

```
Initial Centroids are:  [[ 4.34489155  3.99726667]
 [ 6.46350009  0.77471754]
 [ 7.39015357  1.13206806]
 [ 2.37650624  8.15241778]
 [ 5.36626615  6.51434231]
 [ 8.67805277  9.08757916]
 [ 6.40483149  5.60578084]
 [ 7.94375954  8.21165063]
 [ 7.56399709  7.83135288]
 [ 4.10720306  0.25056515]]
Loss function:  321.176643622
Final centroids are:  [[ 3.49556658  3.56611232]
 [ 7.0238732   1.16206989]
 [ 7.65742297  2.91907608]
 [ 2.46502891  6.89910678]
 [ 4.68506202  6.99101757]
 [ 8.41127011  8.97490383]
 [ 6.21824223  5.62909218]
 [ 7.52197303  8.160704  ]
 [ 4.93112384  8.45618312]
 [ 3.14506148  0.90770655]]
Iterations:  15
```



Iteration 2 of K-means strategy 1 for k = 2, …10 resulted in :



3. K-means using Strategy 2

The centroids are not initialized randomly, only the first centroid is initialized randomly; the other centroids are calculated by choosing the *i*-th center (i*i*>1) among all possible samples such that the average distance of this chosen one to all previous (i*i*-1) centers is maximal.

This is done till all the k centroids are initialized. After initializing the centroids the same algorithm used in strategy 1 is applied and centroids are calculated subsequently till stopping criterion is met. This initialization of centroids in strategy2 is done by function *init_centroids_strategy2(k, firstCentroid, data)*.

4. Results of K-means with Strategy 2

Using number of clusters k = 4, k-means algorithm converged after 9 iterations :

```
Initial centroids:  [[ 3.35409838  5.79603723]
 [ 9.26998864  9.62492869]
 [ 3.85212146 -1.08715226]
 [ 2.95297924  9.65073899]]
Number of clusters: 4
Loss function:  792.537810441
Final centroids are:  [[ 3.30296804  2.55443267]
 [ 6.85658333  7.6614342 ]
```

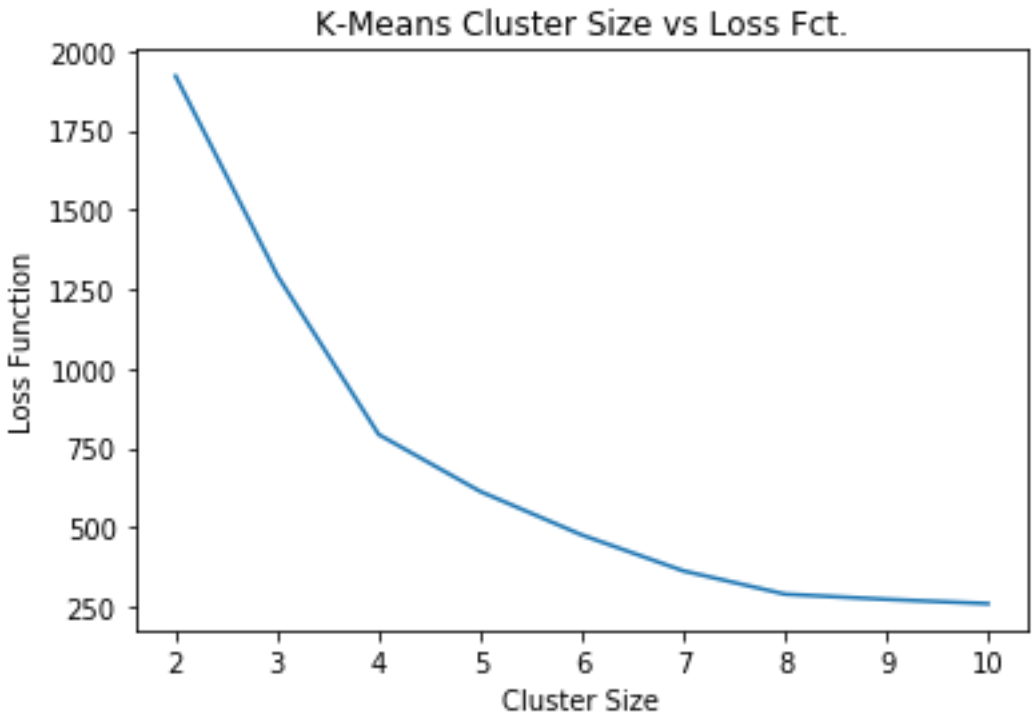
```
 [ 7.34802851  2.35222497]
 [ 3.153427    6.9129207 ]]
Iterations:  9
```

Using number of clusters k = 6, k-means algorithm converged after 9 iterations :

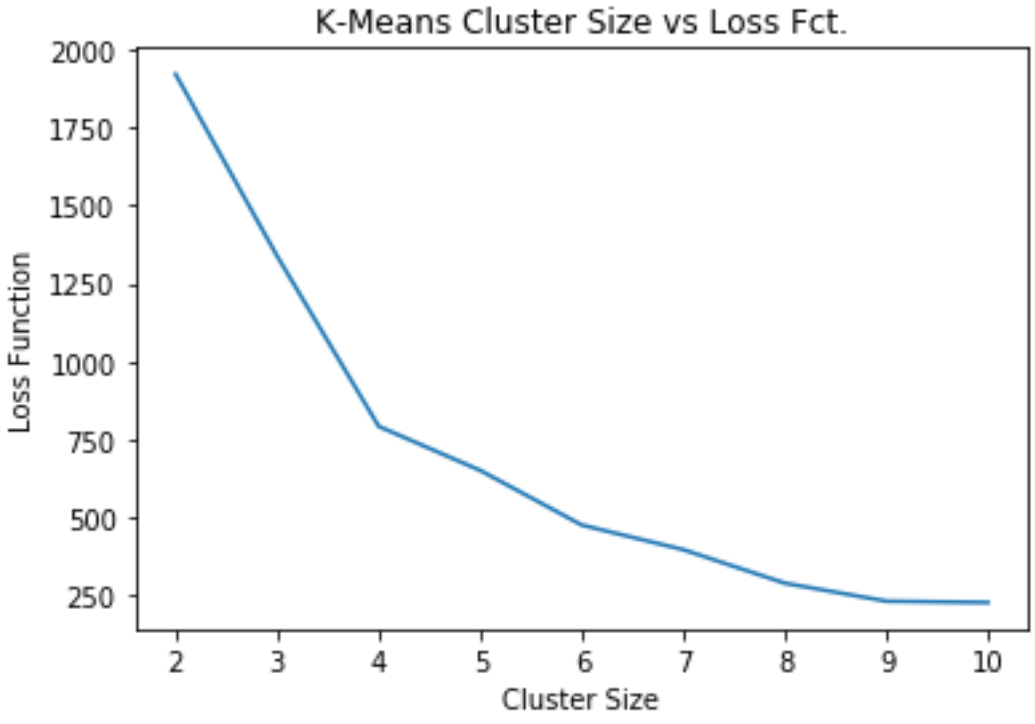
```
Initial centroids:  [[ 1.52668895  4.24557918]
 [ 9.26998864  9.62492869]
 [ 3.85212146 -1.08715226]
 [ 2.95297924  9.65073899]
 [ 7.68097556  0.83542043]
 [ 8.87578072  8.96092361]]
Number of clusters: 6
Loss function:  476.296570527
Final centroids are:  [[ 3.502455    3.62870476]
 [ 7.75648325  8.55668928]
 [ 3.14506148  0.90770655]
 [ 2.52382885  7.02897469]
 [ 7.41419243  2.32169114]
 [ 5.46427736  6.83771354]]
Iterations:  9
```

For k =2,3,5,7,8,9,10 results are :

```
Number of clusters: 2
Loss function:  1921.03348586
Iterations:  5

Number of clusters: 3
Loss function:  1294.29841749
Iterations:  7

Number of clusters: 5
Loss function:  613.282439206
Iterations:  12

Number of clusters: 7
Loss function:  363.220444386
Iterations:  8

Number of clusters: 8
Loss function:  289.932726045
Iterations:  11

Number of clusters: 9
Loss function:  273.573098789
Iterations:  16

Number of clusters: 10
Loss function:  260.040198291
Iterations:  20
```



Iteration 2 of K-means Strategy 2 resulted in :



Conclusion

In this project the K-means algorithm was implemented using 2 initialization strategies. The clustering of the data with k-means was tested with cluster sizes ranging from 2 to 10.
A graph representing the loss function vs. cluster size was plotted for 2 iterations for each

initialization strategy. By observing the graphs, the optimal value of k is 4.