Report on Clustering

Srujitha Reddy Mekala (G01326037)

Description of approach / How the algorithm works?

Initially, centroids have been assigned to random data points using the "initialize_centroid" function. Once the initialization is complete, the distance of each data point is calculated from all the centroids and the data points are assigned to respective centroids with the minimum distance between them.

Now that all the data points have been assigned to one of the clusters, the centroids are recalculated and updated using the "calcNewCentroid" function. Further, the difference between the position of the particular centroid before and after updating is calculated using the "centroid_diff" function. This difference in position is used to decide on when to stop updating the centroid position i.e., when there is no significant change in the position of centroid.

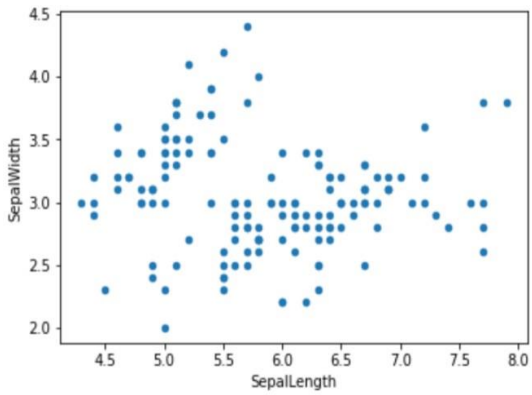
The "display" function is used to plot the clusters and indicate the initial and final centroids. Also, the initial difference between the position of centroid is considered to be 100. The "cluster" array stores the label of the cluster that each data point belongs to. The initial and final position of the centroids is printed along with the plots.

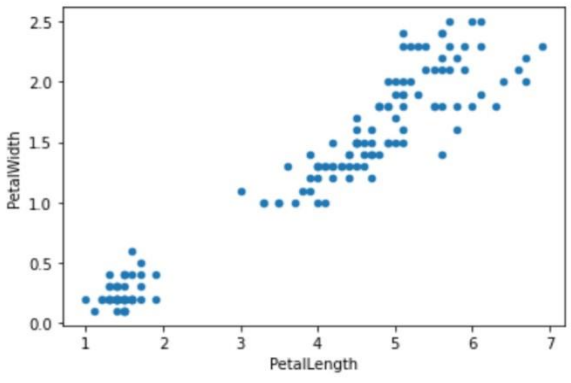
Finally, the same is validated by using the inbuilt function from the sklearn library.

Part a. Iris dataset

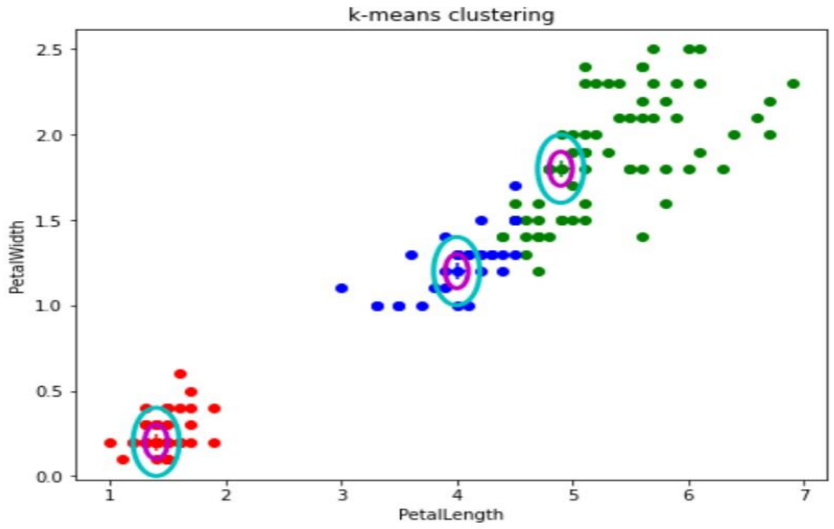
The data set (iris_new_data.txt) is read which has 150 instances and 4 features. Two features are selected from the four that give the highest score without employing any data reduction technique by trial-and-error method (i.e., checking score each time by considering different pair of features). The pair "Petal length" and "Petal width" have proven to show better score/V-measure than the rest of the pairs.

The features are distributed as follows:





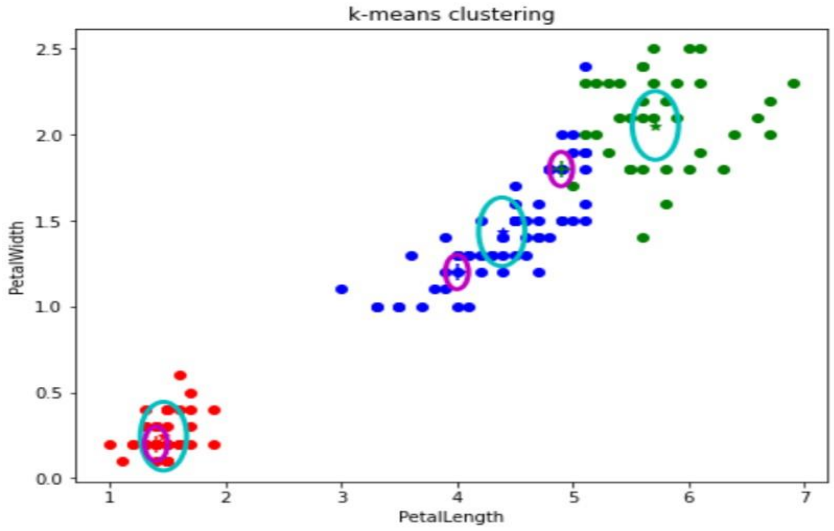
The plot with initial centroids for features petal length and petal width is as follows:



Centroids: `[[5.1, 3.5, 1.4, 0.2]),[5.8, 2.6, 4. , 1.2]), [6.1, 3. , 4.9, 1.8])]`

The final plot with updated centroids is as follows:

Final plot(cyan indicates final and magnetta indicates initial centroids):

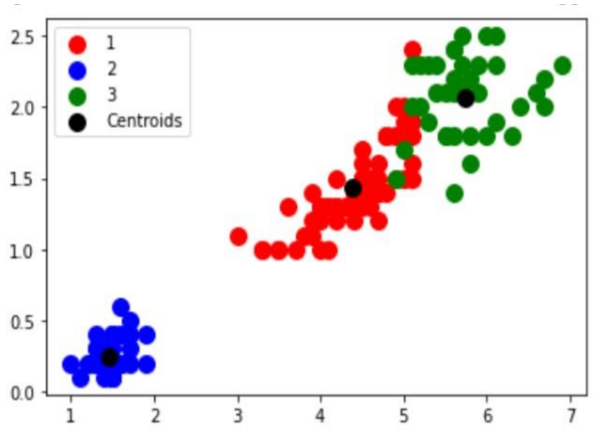


Centroids:
```
[[5.006      3.418      1.464      0.244     ]
 [5.88360656 2.74098361 4.38852459 1.43442623]
 [6.85384615 3.07692308 5.71538462 2.05384615]]
```

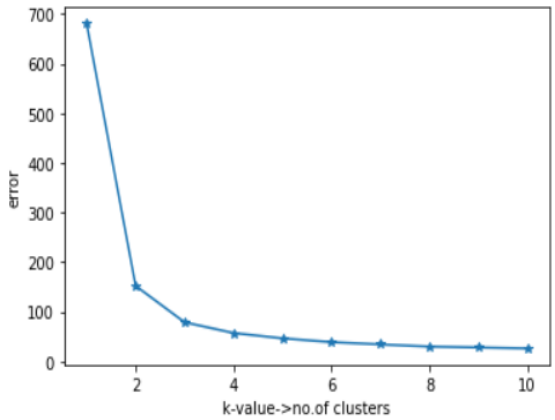
Validation with inbuilt function:

Centroids:
```
[[5.9016129  2.7483871  4.39354839 1.43387097]
 [5.006      3.418      1.464      0.244     ]
 [6.85       3.07368421 5.74210526 2.07105263]]
```

Output plot using inbuilt function:



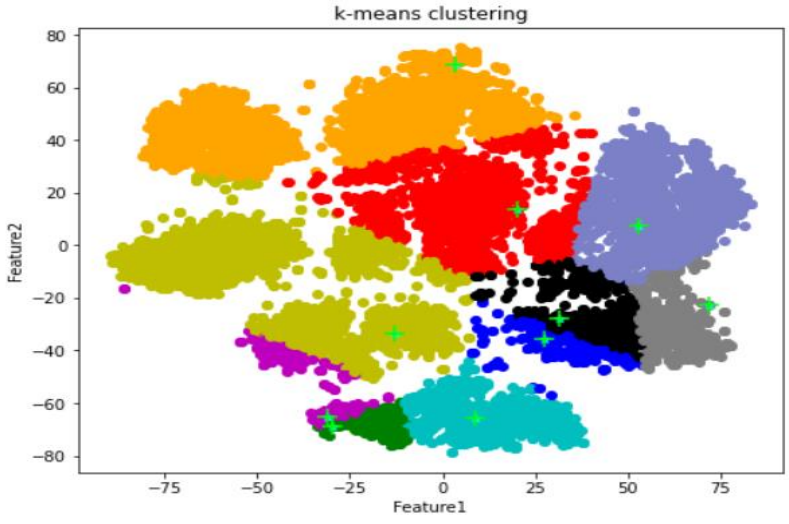
Elbow plot: SSE vs k-value



By comparing the centroid values of the code developed from scratch with those generated by the inbuilt function we can observe that they are almost similar and hence the code is equally efficient. Also, from the elbow plot we can say that the optimal value for k is 3 since after k=3 the line is almost parallel to x-axis indicating almost same SSE value.

Part b. Image dataset

The image dataset is read which has 10,000 instances with 784 features each. Since the data is voluminous, the **t-SNE** (t-Distributed Stochastic Neighbor Embedding) technique is used for feature selection/ data reduction. It is used for non-linear data. Dimensionality reduction maps data from a higher dimensional space to a lower dimensional space with minimal information loss. This makes use of joint probability distribution. This is a symmetric algorithm, to be robust to outliners. The initial plot with centroids is as follows:



Initial centroids:

```
[[20.081482, 13.438108],[ 27.319973, -35.48972 ],[-29.581121, -69.015076],
[  8.762394, -65.76207 ],[-31.28161, -65.19953],[-13.032734, -33.562878], [
```

```
31.483753, -27.720129],[ 3.4006977, 68.63864  ],[ 71.95225 , -22.825268],
[52.99084,  7.46305]]
```

Final centroids:

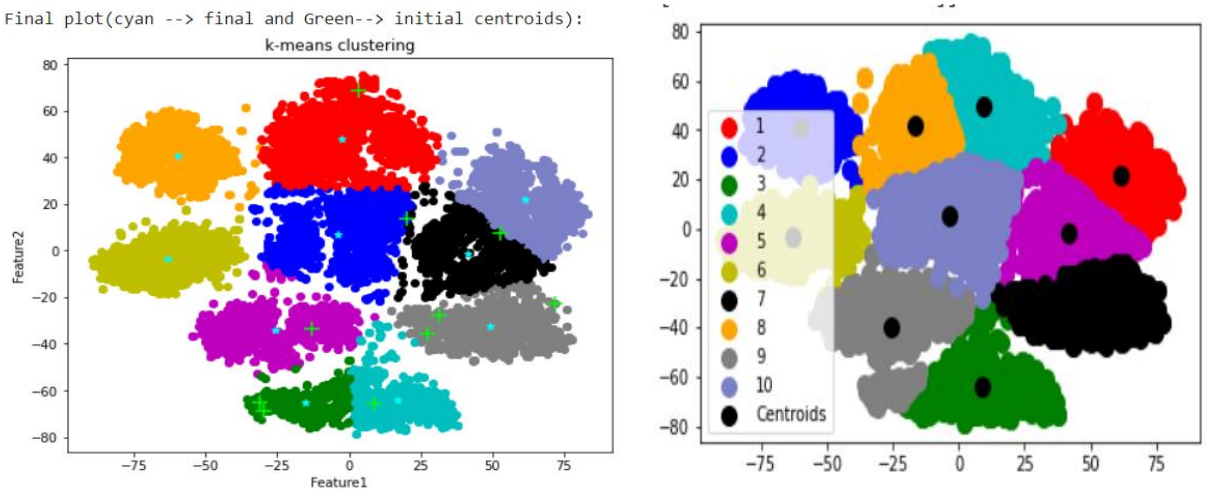
```
[[ -2.7108407  47.630863 ]
 [ -3.668123    6.957359 ]
 [-15.250223  -65.07816  ]
 [ 16.953186  -63.925293 ]
 [-25.554602  -33.915607 ]
 [-63.038715   -3.3487077]
 [ 41.33832    -1.2773286]
 [-59.548412   40.62421  ]
 [ 49.271397  -32.6301   ]
 [ 61.155235   21.79532  ]]
```

Comparing with inbuilt function:

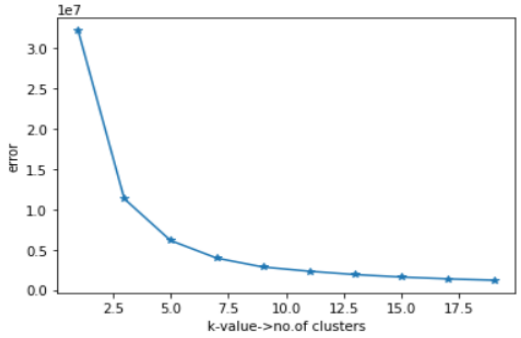
```
[[ 61.449028   21.594275 ]
 [-60.093266   40.71836  ]
 [  8.758468  -64.31005  ]
 [  9.442641   49.916042 ]
 [ 41.50916    -1.5284667]
 [-62.965164   -3.418693 ]
 [ 49.174545  -32.702766 ]
 [-16.820034   42.12124  ]
 [-25.802057  -39.827972 ]
 [ -3.4089608   5.3386154]]
```

The final plot with updated centroids is as follows:

[left image→ developed code output, right image→inbuilt function output]



Below is a plot of SSE vs no. of clusters (k) to find the optimal value for k:



From the above elbow plot, we can observe that after k=7 the line is almost parallel to x-axis which indicates 7 to be the optimal k-value.

Miner details:

username: ds18 **Rank, score:** **part a.** 28, 0.72 **part b.** 20, 0.78