Jason Ballantyne
13432788

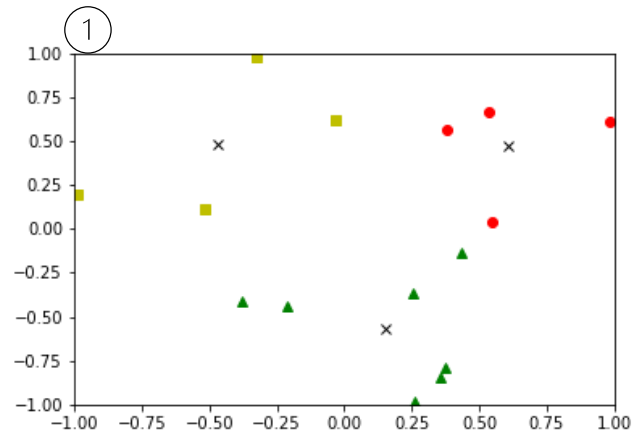
Text Analytics COMP47600
Practical Number 7

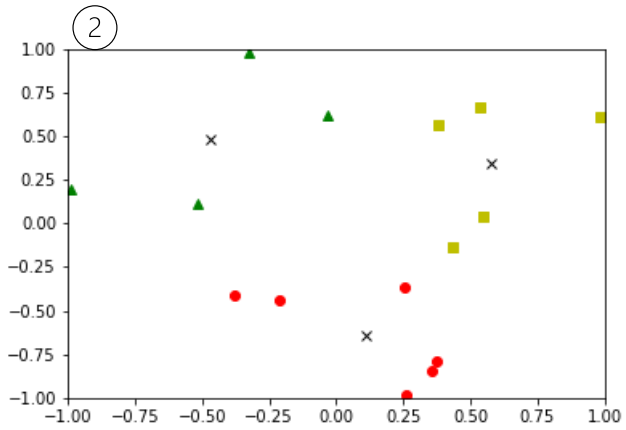
01/11/2021

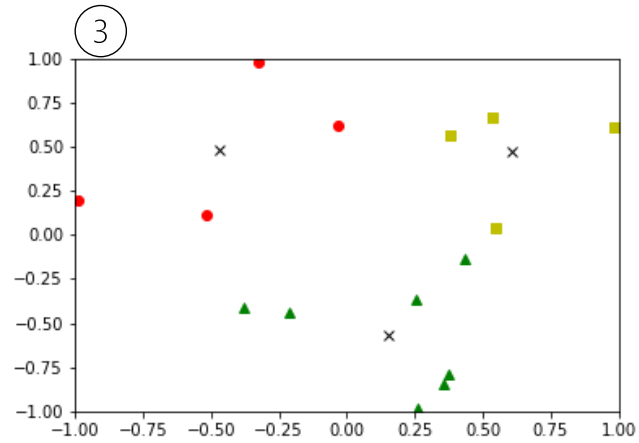
Q1: K-means clustering is one of the simplest and popular unsupervised machine learning algorithms. The objective of K-means is simple, it groups similar data points together and discover underlying patterns. It aims to partition the data into k clusters in a way that data points in the same cluster are similar and data points in the different clusters are farther apart. The similarity of two points is determined by the distance between them. It finds the similarity between the items and groups them into the clusters.

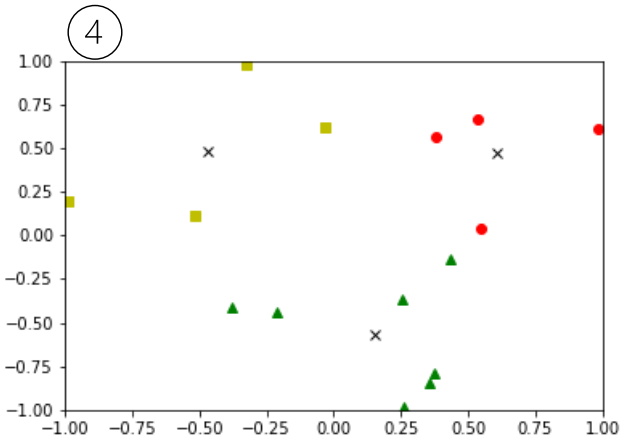
To do this, we begin with choosing the number of K clusters. The K signifies the number of clusters that the algorithm would find in the dataset. The second step is to allocate K random points as centroids (arithmetic mean of all the data points that belong to that cluster). Thirdly, we must assign each data point to the closest centroid, this forms K clusters. The fourth step is to compute and place the new centroid of each cluster. Lastly, we must reassign each data point to the new closest centroid.

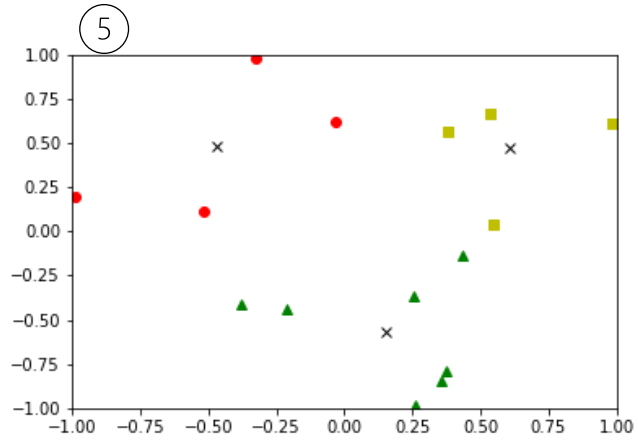
Using the “init_board” method, I have run the K-means algorithm 10 times and the following the results of all the clusters obtained.

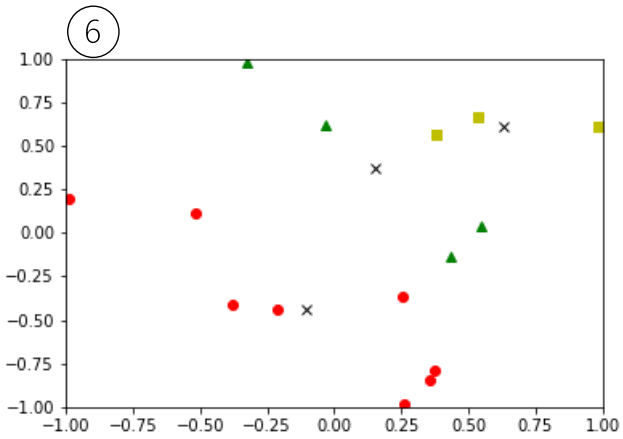


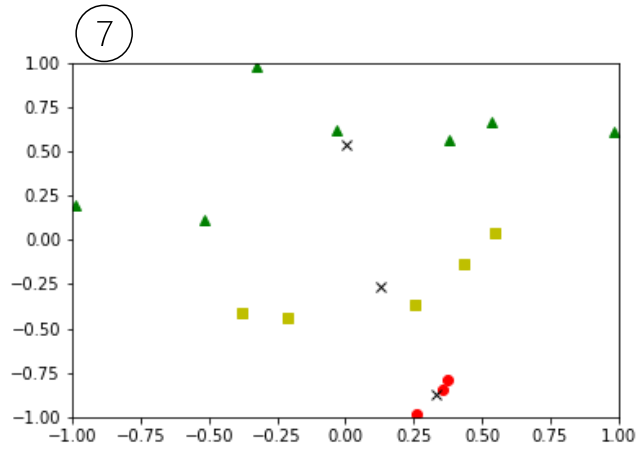


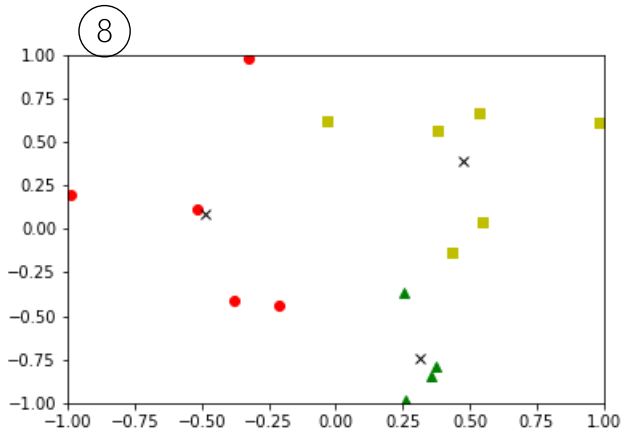


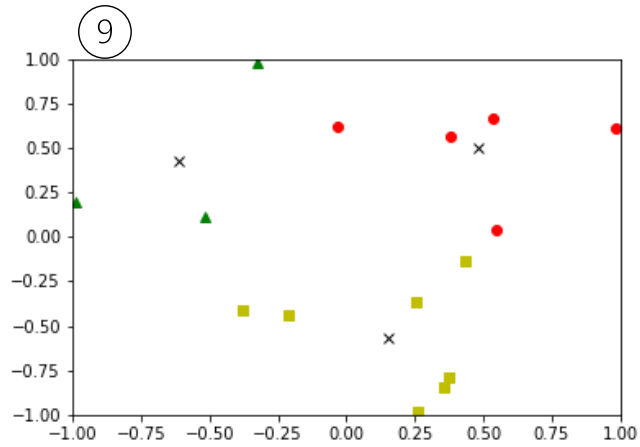


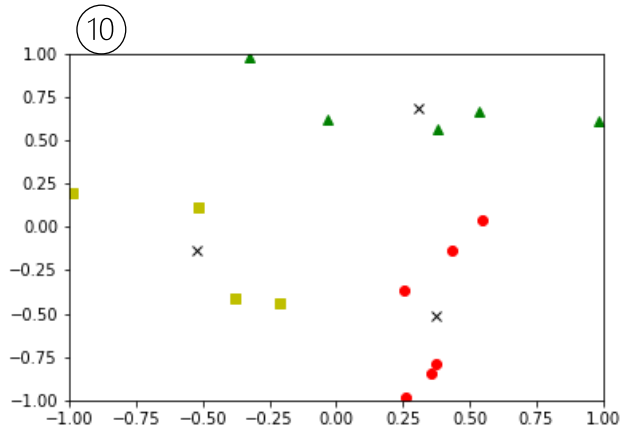












Based on the above 10 clusters, we can see that there are **four unique** results of clusters obtained. The largest group of clusters appear in **cluster 1,2,3,4,5,9** where these clusters are very similar to one another. The second largest group of clusters exists in **cluster 8 and 10** where they are roughly similar. The two remaining groups of clusters are completely unique do not resemble each other or any of the other clusters, these exist in **cluster 6 and 7**.

Within the most common cluster, the centroids exist at roughly the following destinations: 1: (-0.50, 0.50) 2: (0.125, -0.50) 3: (0.50, 0.50). The second largest cluster had centroids at roughly the following destinations: 1: (-0.50, 0.125) 2: (0.375, -0.50) 3: (0.50, 0.50). The remaining two unique clusters did not resemble either of these centroid locations and varied quite a lot.

Jason Ballantyne
13432788

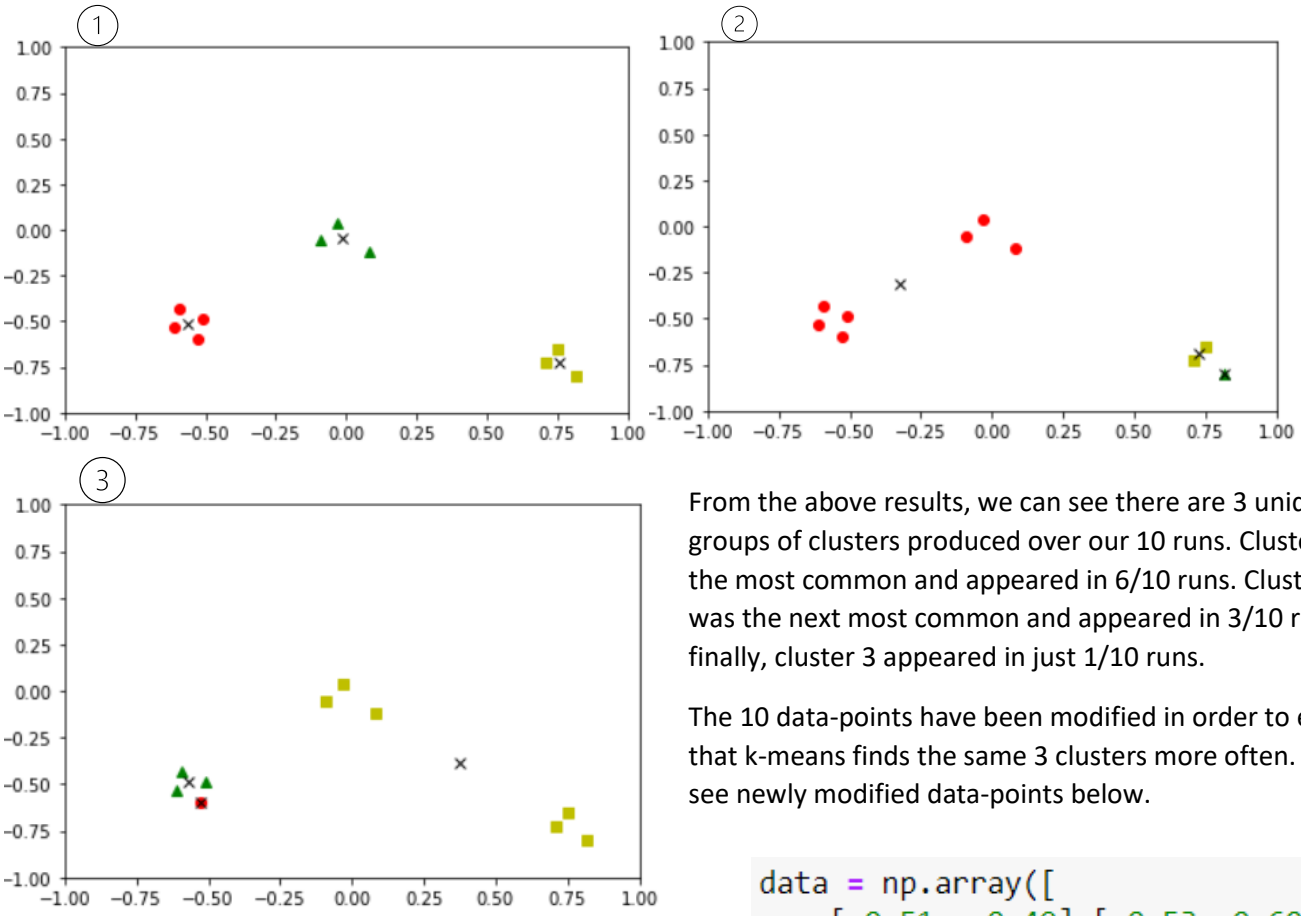
Text Analytics COMP47600
Practical Number 7

01/11/2021

Q2: Please see my own 10 constructed data-points in the snippet below.

```
data = np.array([
    [-0.51, -0.49],[-0.53,-0.60],
    [-0.59,-0.43], [-0.61,-0.53],
    [-0.03, 0.04],[-0.09, -0.05],
    [0.08, -0.12],[0.82,-0.80],
    [0.75,-0.65], [0.71, -0.73],
])
```

I have run the above dataset 10 times through the k-means algorithm and the below is the results of the unique clusters identified.

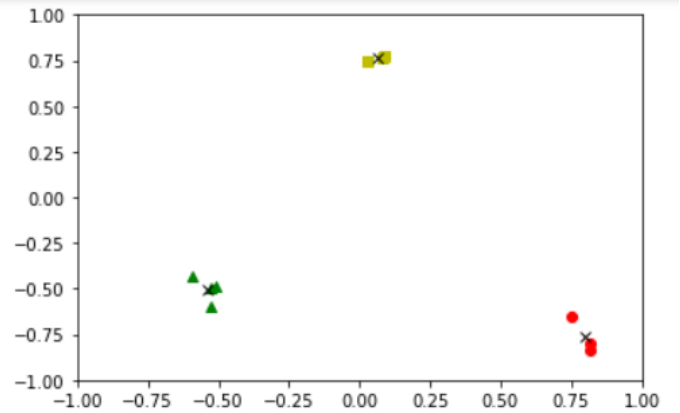


From the above results, we can see there are 3 unique groups of clusters produced over our 10 runs. Cluster 1 was the most common and appeared in 6/10 runs. Cluster 2 was the next most common and appeared in 3/10 runs and finally, cluster 3 appeared in just 1/10 runs.

The 10 data-points have been modified in order to ensure that k-means finds the same 3 clusters more often. Please see newly modified data-points below.

```
data = np.array([
    [-0.51, -0.49],[-0.53,-0.60],
    [-0.59,-0.43], [-0.53,-0.50],
    [0.03, 0.75],[0.08, 0.76],
    [0.09, 0.77],[0.82,-0.80],
    [0.75,-0.65], [0.82,-0.84],
])
```

The changes I made to the data points to ensure that most of the time k-means would find the same 3 clusters, include further spacing out of the data set from one another. When the data has well separated clusters, the performance of k-means depends completely on the goodness of the initialization. As well as this, I adjusted the individual data points within their clusters to be closer to one another, in order for the algorithm to correctly identify this as a cluster. As a result of these modifications, I achieved the below cluster for a more consistent 9/10 runs.



Q3: As we have seen from the above results, k-means does not guarantee unique clustering in each of its runs and each run may lead to different clustering results. Some problems certainly do exist with k-means, firstly k-means assumes the variance of the distribution of each attribute is spherical. Secondly it assumes all variables have the same variance. This means k-means cannot handle noisy data and outliers. Lastly, the prior probability for all k clusters is the same. This means that each cluster has roughly an equal number of observations.

Despite these problems, there are methods that have been developed to improve the clusters found by k-means. Some of these methods include Specific random seed value, Single pass seed selection (SPSS) algorithm, Subsampling and choosing random, Random partitions, Splitting algorithm & Sorting Heuristics.

Sorting Heuristics:

For the purposes of this answer, we will be further investigating "Sorting Heuristics" as a method to improve clusters found by k-means. It is based on a popular technique to sort the data points based on a given criteria. Abdul Nazeer and Sebastian outline that the basic idea of this heuristic sorting algorithm is "to determine the initial centroids of the clusters in a heuristic manner, so as to ensure that the centroids are chosen in accordance with the distribution of data." (Nazeer, Kumar & Sebastian, 2011). In regard to the sorting, it can be sorted based upon; The distance to the center point, density, centrality and attribute with the greatest variance.

Distance to the center point - This involves sorting the data points based upon their distance to the center of the data. The centroids are then selected as every N/kth point in this order. This variant is included in our tests. To add randomness, we choose a random data point as a reference point instead of the center. This heuristic fulfills our requirements. It is fast, simple, and requires no additional parameters.

Density – The density is based upon the number of other points within a distance d1. First centroid is the point with the highest density, and the remaining k -1 centroids are chosen at decreasing order with the condition that they are not closer than distance d2 from an already chosen centroid

Centrality – This method involves using a primary criterion to estimate how central a point is (how far from boundary). Secondary threshold criterion is used to prevent centroids from being neighbours.

Attribute with the greatest variance – This method sorts the data points according to the dimension with the largest variance. The data points are then partitioned into 'k' equal size clusters. The median of each cluster is selected instead of the mean. This approach belongs to a more general class of projection-based techniques where the objects are mapped to some linear axis such as diagonal or principal axis.

After sorting has taken place, k points are selected from the sorted list using one of the following heuristics:

- First k points
- First k points while disallowing points closer than ε to already chosen centroids.
- Every (N/k)th point (uniform partition)

The sorting heuristic would work great if the clusters were well separated and each have different criterion value. However, with most datasets the clusters tend to be randomly located in respect to the center point, and it is unlikely that all the clusters would have different criterion values.

We have discussed what the “Sorting Heuristics” method is and how to implement it. However, does it improve the clusters found by the original k-means algorithm? Based on a paper by Madhu Kumar, he has highlighted that their “Experimental results have shown that the proposed algorithm produces better clusters in less computation time compared to the original k-means algorithm” (Nazeer, Kumar & Sebastian, 2011). To conclude, we have illustrated what the “Sorting Heuristics” method is and how it works. Then, based on evidence from research papers, how the implementation of an improved k-means algorithm using this heuristic method has resulted in clusters with better accuracy and in less computation time.

References:

Abdul Nazeer, K., Sebastian, M., & Madhu Kumar, S. (2011). A Heuristic k-Means Algorithm with Better Accuracy and Efficiency for Clustering Health Informatics Data. Journal Of Medical Imaging And Health Informatics, 1(1), 66-71. doi: 10.1166/jmihi.2011.1010

Fränti, P., & Sieranoja, S. (2019). How much can k-means be improved by using better initialization and repeats?. Pattern Recognition, 93, 95-112. doi: 10.1016/j.patcog.2019.04.014

Mahmud, M., Rahman, M., & Akhtar, M. (2012). Improvement of K-means clustering algorithm with better initial centroids based on weighted average. 2012 7Th International Conference On Electrical And Computer Engineering. doi: 10.1109/icece.2012.6471633

Nazeer, K., Kumar, S., & Sebastian, M. (2011). Enhancing the K-means Clustering Algorithm by Using a O(n logn) Heuristic Method for Finding Better Initial Centroids. 2011 Second International Conference On Emerging Applications Of Information Technology. doi: 10.1109/eait.2011.57

Sieranoja, S., & Fränti, P. (2018). Random Projection for k-means Clustering. Artificial Intelligence And Soft Computing, 680-689. doi: 10.1007/978-3-319-91253-0_63