**Student ID: 201680340**

# 1 K-means

The K-means is a representative based, clustering algorithm that uses centroid's to creates K number of clusters, within a unsupervised clustering problems. As such, the target function is to minimise the squared distance between data objects and the centroid in its clusters.

This functions through the centroid's being initialised as a random point within the feature-space, but not necessarily an actual datapoint. Next, each data object is assigned to its nearest centroid, all data objects with the same centroid form a cluster. The centroid is then updated according to the mean of the cluster. This occurs iterativley until convergence where the updating of a centroid does not cause any changes to the objects within each cluster, or the maximum number of iterations is reached .

## 1.1 K-Means: Pesudo-code

```
Algorithm 1: K-means
Input: Dataset: $X$, Number of clusters: $K$, Maximum number of iterations
Output: $K$ clusters

Randomly initialize $K$ number of centroids $\in$; repeat
    for every data point $x_i \in X$ do
        Calculate the distance $d_{ij}$ to every centroid $c_j$; Assign $x_i$ to the cluster $c_j$ of the
         centroid with min($d_{ij}$);
    end
    for each cluster $C_j$ do
        Calculate the mean of all data points within it and update centroid $c_j$;
    end
until max iteration is met;
return set of $K$ clusters;
```

# 2 K-means++

The K-means++ algorithm improves upon K-means by selecting the initial centroid representatives in a systematic way that increases the chances of optimal clustering. The first centroid is an actual datapoint from the dataset chosen at random, then the squared distance from this centroid to every other data point is calculated. The squared distances are summed up and is used to create a probability distribution such that the next centroid is chosen with probability proportional to its squared distance from the centroid. This repeats for the next centroid, with the adjustment that the distance to the nearest centroid is used in constructing the probability distribution.

The use of the probability distribution to choose the centroids ensures that the clusters will be more spaced out than that of random initialisation. After K number of centroids have been initialized, the process continues as per the standard K-means algorithm.

### 2.1 K-Means++: Pesudo-code

```
Algorithm 2: k-means++ algorithm
  Input: Data set: $X$, number of clusters: $K$
  Output: $K$ clusters
  Randomly select a datapoint $\in X$ as first centroid $c_j = 1$. for $j = 2$ to $K$ do
      Calculate the squared distance $d(x_i)^2$ for $x_i \in X$ to the nearest centroid in the set
      of centroids $C_{j-1}$.
      calculate sum of distances squared, $\sum d(x_i)^2$
      Select the next centroid $c_j$ from $X$ with probability proportional to $d(x)^2$
  end
  repeat
      for every data point $x_i$ $\in X$ do
          Calculate the distance $d_{ij}$ to every centroid $c_j$; Assign $x_i$ to the cluster $c_j$ of the
          centroid with min($d_{ij}$);
      end
      for each cluster $C_j$ do
          Calculate the mean of all data points within it and update centroid $c_j$;
      end
  until max iteration is met
  return Cluster centroids $C$
```

## 3 Bisecting k-means

The Bisecting k-means algorithm is a heirarchical divisive clustering algorithm that forms clusters in a 'top-down' approach starting with a single cluster containing the entirety of the dataset. The cluster with the largest sum of squared distances between the objects of the cluster and the cluster centroid is subject to division into two new clusters. This target cluster is split using the k-means algorithm with the k being equalt to 2. This iterative procedure continues until the specified number of clusters is reached.

### 3.1 Bisecting K-Means: Pesudo-code

```
Algorithm 3: Bisecting K-means algorithm
  Input: Data set $X$, number of clusters $S$
  Output: Cluster assignments for each data point
  Initialize one cluster with all $x_i \in D$;
  repeat
      Select the cluster with highest error $\sum_{i=1}^{n}(x_i - c_j)^2$;
      Randomly initialize $K$=2 number of centroids for K-means; repeat
          for every data point $x_i$ $\in X$ do
              Calculate the distance $d_{ij}$ to every centroid $c_j$; Assign $x_i$ to the cluster $c_j$ of
              the centroid with min($d_{ij}$);
          end
          for each cluster $C_j$ do
              Calculate the mean of all data points within it and update centroid $c_j$;
          end
      until max iteration is met;
  until number of clusters: S, is reached;
```

# 4 Clustering Comparison

**Table 1:** Silhouette coefficients for each algorithm

| K or S value | Kmeans | Kmeans++ | Bi-secting Kmeans |
|---|---|---|---|
| 1 | -1 | -1 | -1 |
| 2 | 0.1470 | 0.1470 | 0.1470 |
| 3 | 0.0787 | **0.1424** | **0.1431** |
| 4 | **0.9534** | 0.1400 | 0.1300 |
| 5 | 0.921 | 0.1200 | 0.1263 |
| 6 | 0.090 | 0.0893 | 0.1272 |
| 7 | 0.0930 | 0.0829 | 0.1182 |
| 8 | 0.090 | 0.0942 | 0.1282 |
| 9 | 0.0924 | 0.0908 | 0.1233 |

The first observation is that for one cluster, all algorithms have a silhouette coefficient of -1. This is because for one cluster, there is no other cluster to calculate a b value from, and according to the silhouette coefficient equation, produces a value of -1.

The next observation is that all algorithms produced the same highest silhouette coefficient value of 0.1470 for 2 clusters. If the labels of the objects within the two clusters is printed out, it can be seen that all algorithms produce one cluster containing all objects within the "Country" category and all the rest of the objects in the other cluster. Whilst potentially an erroneous result, it could alternatively be explained by the fact the cluster size of the "Country" cluster being significantly larger than the other clusters with n=161. This would consequently inflate the mean silhouette coefficient value.

For the Kmeans algorithm, the number of clusters that produced the second highest silhouette coefficient was k=4, shown by the bold value in Table 1 of 0.9534 and the highest value on the graph. In comparison the second highest Silhouette coefficient for Kmeans++ and Bisecting Kmeans both occured under k or s=4, with 0.1424 and 0.1431 respectively.

Assuming that the ground truth clustering is four clusters of objects with the category labels Country, Animal, Fruit or Vegetable: Table 2 shows the composition of the four clusters. (Note: this can be seen by unhasing the "print_cluster_contents()" function for each algorithm at the bottom of the code.)

**Table 2:** Cluster compositions for four clusters

| | C1 | C2 | C3 | C4 | Silhouette |
|---|---|---|---|---|---|
| Kmeans (K=4) | Country(58) | Country(103) | Animal(50)<br>Fruit(4) | Fruit(55)<br>Vegetables(56) | 0.9534 |
| Kmeans++ (K=4) | Fruit (3)<br>Animal (49) | Country (n=159) | Country(2)<br>Animal(1)<br>Fruit (52)<br>Vegetable (8) | Vegetables (52) | 0.1400 |
| Bi-secting Kmeans (S=4) | Country(161) | Animal(50)<br>Fruit(20)<br>Vegetable(4) | Vegetable (n=50) | Fruit(34)<br>Vegetable(7) | 0.13 |

This displays that for K-means, it split up objects with the common ”Country” category label in to separate clusters. This is likely due to two of the randomly initialised centroids being placed within the same cluster.

All Animal objects were able to be correctly placed into the same cluster (C3) along side a small amount of misplaced ”Fruit” objects. It failed to split the large amount of Fruit and Vegetable objects incorrectly placed together into sepeate categories. This is also due to the required centroid being randomly initialised within the true ”country” cluster.

K-means++ was able to achieve a higher coefficient score of 0.14 due to the centroid selection through the probability distribution creating more separated clusters. This can be seen with C2 and C3 containing all objects of one category.

Bi-secting Kmeans also was able to create the essentially the same clusters, however the silhouette coefficient was reduced by C2, containing all the animal objects, being polluted with a large portion of the fruit objects.

Overall it can be concluded that the best clustering is achieved for equally for all algorithms with two clusters due to the country objects being easily separable. Yet, making the assumption that the ground truth clustering consists of four clusters; Kmeans ++ achieves the highest Silhouette Coefficent.