Seed Clustering

Dataset – Seeds Data Set
Method – Hierarchical Clustering – Complete Linkage using Euclidean distance
K-Means – Clustering
Gap Statistics – Choosing K
Rand Index and Adjusted Rand Index – Cluster performance

Seeds data set contains attributes as below:

Attribute Information:

To construct the data, seven geometric parameters of wheat kernels were measured:
1. area A,
2. perimeter P,
3. compactness C = 4*pi*A/P^2,
4. length of kernel,
5. width of kernel,
6. asymmetry coefficient
7. length of kernel groove.

All of these parameters were real-valued continuous.

These attributes help to group different kernels of varieties of wheat. Let's use clustering to see the grouping done is appropriate based on field seed group.

Dataset contains different attributes of different metrics so we have to scale the data to normalize data before we proceed for clustering.
Step 1-> Load data from text file

```
> data<-read.delim("seeds_dataset.txt",header=TRUE,sep="\t")
> dim(data)
[1] 199   8
> head(data)
   Area Perimeter Compactness Length.Kernel Width.Kernel Asymmetry Length.Ker
nel.Grove
1 15.26     14.84      0.8710         5.763        3.312     2.221
5.220
2 14.88     14.57      0.8811         5.554        3.333     1.018
4.956
3 14.29     14.09      0.9050         5.291        3.337     2.699
4.825
4 13.84     13.94      0.8955         5.324        3.379     2.259
4.805
5 16.14     14.99      0.9034         5.658        3.562     1.355
5.175
6 14.38     14.21      0.8951         5.386        3.312     2.462
4.956
  Seed.Group
1          A
2          A
3          A
4          A
```

```
5           A
6           A
```

It has 199 observations with 8 attributes

Step 2-> We have to exclude one attribute which is seed group which will be used to validate the clustering.

```
> data_final<-data[c(1,2,3,4,5,6,7)]
> dim(data_final)
[1] 199   7
> head(data_final)
   Area Perimeter Compactness Length.Kernel Width.Kernel Asymmetry Length.Kernel.Grove
1 15.26     14.84      0.8710         5.763        3.312     2.221               5.220
2 14.88     14.57      0.8811         5.554        3.333     1.018               4.956
3 14.29     14.09      0.9050         5.291        3.337     2.699               4.825
4 13.84     13.94      0.8955         5.324        3.379     2.259               4.805
5 16.14     14.99      0.9034         5.658        3.562     1.355               5.175
6 14.38     14.21      0.8951         5.386        3.312     2.462               4.956
```

Step 3-> Normalize data using scale to make data uniform before clustering

```
> datascaled<-scale(data_final)
> summary(datascaled)
      Area              Perimeter          Compactness        Length.Kernel       Wid
th.Kernel
 Min.   :-1.4825   Min.   :-1.6680   Min.   :-2.6891   Min.   :-1.6776   Min.   
:-1.67987
 1st Qu.:-0.8866   1st Qu.:-0.8591   1st Qu.:-0.5879   1st Qu.:-0.8480   1st 
Qu.:-0.82214
 Median :-0.1674   Median :-0.1723   Median : 0.1110   Median :-0.2303   Medi
an :-0.05427
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   
: 0.00000
 3rd Qu.: 0.8686   3rd Qu.: 0.9227   3rd Qu.: 0.6857   3rd Qu.: 0.8090   3rd 
Qu.: 0.79025
 Max.   : 2.1443   Max.   : 2.0254   Max.   : 2.0364   Max.   : 2.3261   Max.   
: 2.02861
   Asymmetry         Length.Kernel.Grove
 Min.   :-1.99450   Min.   :-1.8300
 1st Qu.:-0.76760   1st Qu.:-0.7604
 Median :-0.04637   Median :-0.3910
 Mean   : 0.00000   Mean   : 0.0000
 3rd Qu.: 0.74759   3rd Qu.: 0.9302
 Max.   : 3.13764   Max.   : 2.2921
> round(sd(datascaled),0)
[1] 1
> head(datascaled)
            Area   Perimeter Compactness Length.Kernel Width.Kernel  Asymmetr
y
[1,]  0.11686956  0.18632674 0.008123812     0.2701781    0.1228250 -1.004836
3
[2,] -0.01326851 -0.01971022 0.441228579    -0.2009740    0.1783333 -1.822590
5
[3,] -0.21532498 -0.38599814 1.466100254    -0.7938592    0.1889063 -0.679909
9
[4,] -0.36943585 -0.50046311 1.058724484    -0.7194668    0.2999230 -0.979005
4
```

```
[5,]  0.41824192  0.30079171 1.397489598     0.0334749    0.7836384 -1.593510
6
[6,] -0.18450280 -0.29442616 1.041571820    -0.5796992    0.1228250 -0.841013
6
     Length.Kernel.Grove
[1,]          -0.4072377
[2,]          -0.9430413
[3,]          -1.2089135
[4,]          -1.2495047
[5,]          -0.4985678
[6,]          -0.9430413
```

Step 4-> Calculate distance matrix using Euclidean distance

```
> distdata<-dist(datascaled,method = "euclidean")
> distmat<-as.matrix(distdata)
> dim(distmat)
[1] 199 199
```

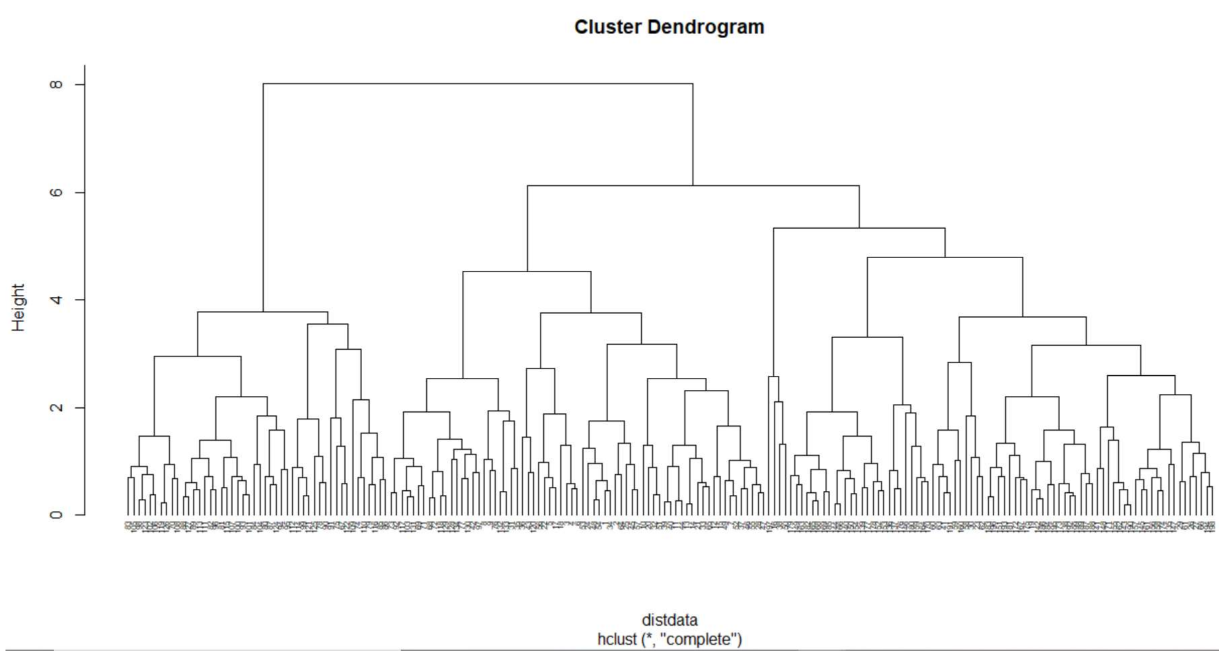
We have 199 observations so distance matrix will be formed which shows distance between each observation which is input to clustering algorithm

Step 5-> Hierarchical clustering - Complete, average and single

```
> #hierarchical clustering - Complete,average and single
> wheatclust_comp<-hclust(distdata,method="complete")
> wheatclust_avg<-hclust(distdata,method="average")
> wheatclust_single<-hclust(distdata,method="single")
```

Step 6-> Validate the clusters formed by each linkage type

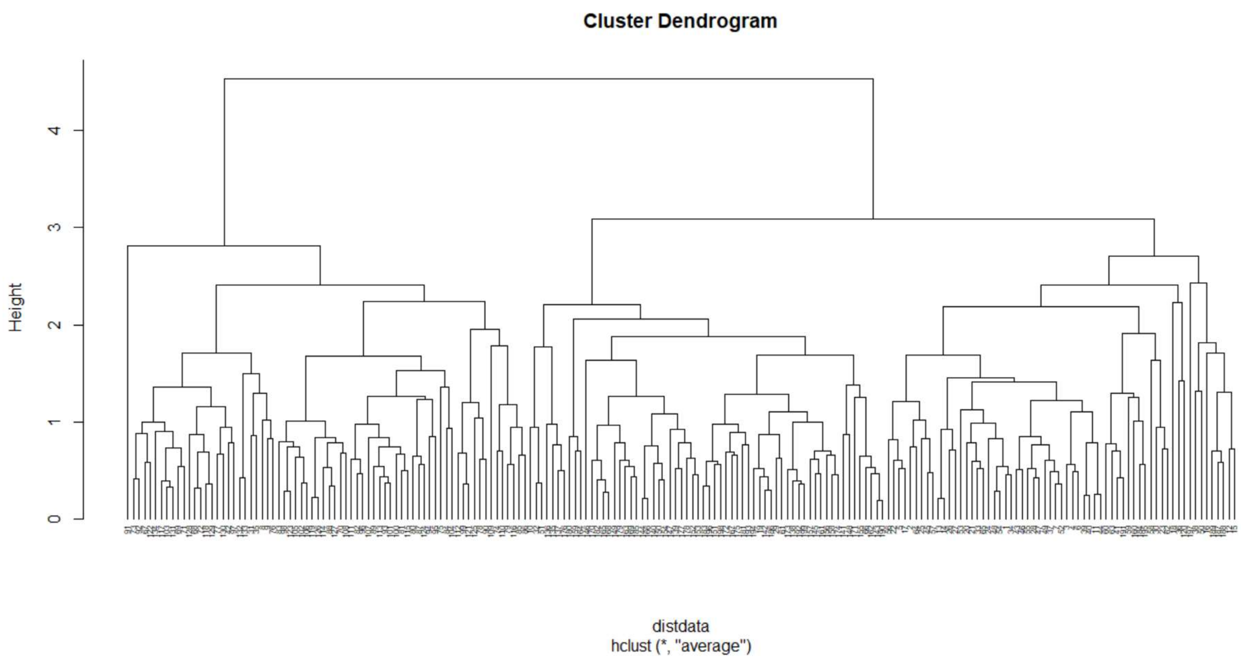
Complete:



```
> wheatclust_comp<-cutree(wheatclust_comp,k=3)
> table(wheatclust_comp, data$Seed.Group)

wheatclust_comp  A  B  C
              1 49 20  0
              2 17  0 65
              3  0 48  0
> rand.index(wheatclust_comp, as.numeric(data$Seed.Group))
[1] 0.8031572
> adj.rand.index(wheatclust_comp, as.numeric(data$Seed.Group))
[1] 0.5599257
```

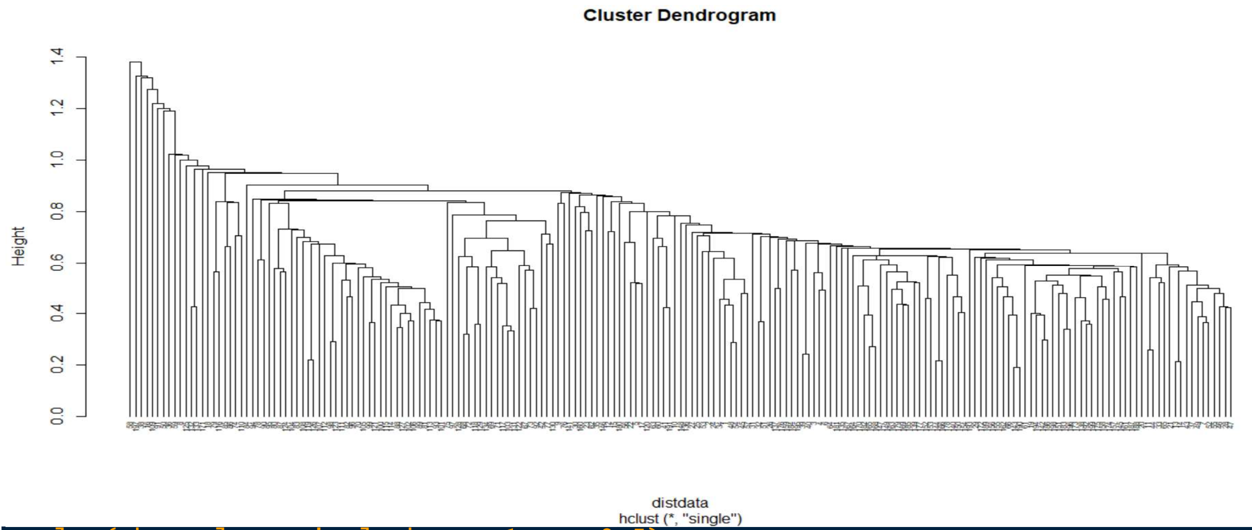
Average:




```
> plot(wheatclust_avg,hang=-1,cex=0.5)
> wheatclust_avg<-cutree(wheatclust_avg,k=3)
> table(wheatclust_avg, data$Seed.Group)

wheatclust_avg  A  B  C
             1 54  1  8
             2  5 67  0
             3  7  0 57
> rand.index(wheatclust_avg, as.numeric(data$Seed.Group))
[1] 0.876453
> adj.rand.index(wheatclust_avg, as.numeric(data$Seed.Group))
[1] 0.7208842
```

Single:




```
> plot(wheatclust_single,hang=-1,cex=0.5)
> wheatclust_single<-cutree(wheatclust_single,k=3)
> table(wheatclust_single, data$Seed.Group)

wheatclust_single  A  B  C
                1 65 68 64
                2  1  0  0
                3  0  0  1
> rand.index(wheatclust_single, as.numeric(data$Seed.Group))
[1] 0.3370387
> adj.rand.index(wheatclust_single, as.numeric(data$Seed.Group))
[1] 0.0002118046
```

Summary:

		Seed Group A	Seed Group B	Seed Group C
Cluster1	Single Linkage	65	68	64
	Average Linkage	54	1	8
	Complete Linkage	49	20	0
Cluster2	Single Linkage	1	0	0
	Average Linkage	5	67	0
	Complete Linkage	17	0	65
Cluster3	Single Linkage	0	0	1
	Average Linkage	7	0	57
	Complete Linkage	0	48	0

	Rand Index	Adjusted Rand Index
Complete Linkage	0.803	0.5599
Average Linkage	0.876	0.72
Single Linkage	0.337	0.0002

Rand index a cluster measure which verifies cluster stability based on
True Positive – Observations which are similar belong to same cluster not distributed across clusters
True Negative – Observations which are dissimilar belong to different clusters not in the same cluster

Rand Index = 1 represents perfect cluster.

Single linkage performed worst which has lower rand index showing that elements are distributed just in one cluster.

Average linkage performed best which has higher rand index showing that elements are distributed based on their similarities but its not perfect cluster but its best of what we have.

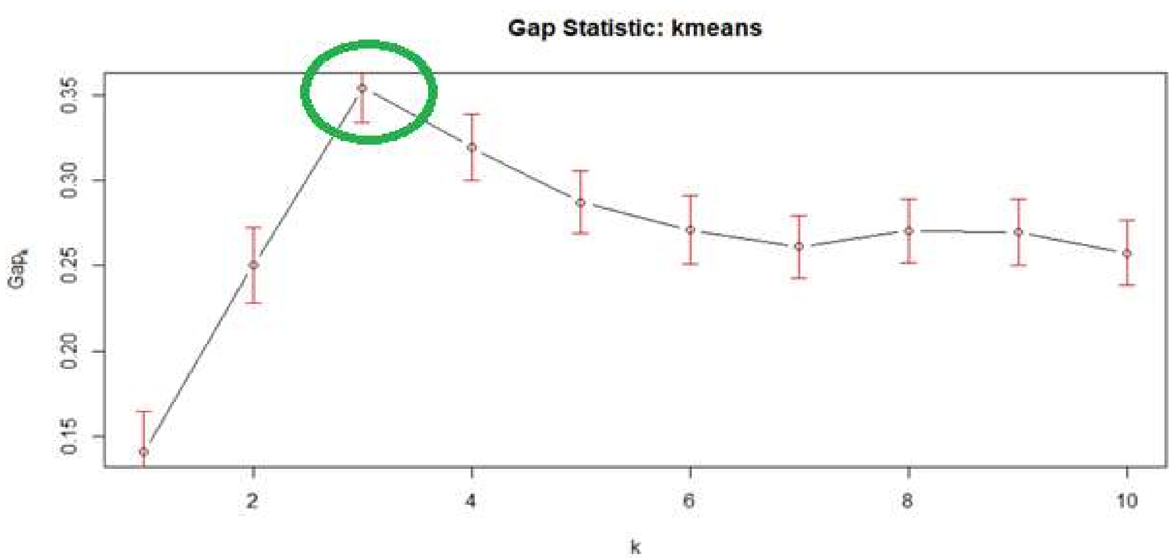
Expectation:

As per the seed.group which classified kernel into three groups A, B and C, I was expecting a perfect cluster. But clustering did not result in perfect clustering instead the observations formed 3 clusters with combination of seed.group A, B and C. Hence seed.group has not classified the kernel types properly.

Cluster using K-means and compare performance with Hierarchical

Let's use gap statistics which will tell us what is the optimal value for k (number of clusters).

```
> gap_kmeans <- clusGap(data_final, kmeans, nstart = 20, K.max = 10, B = 100)
Clustering k = 1,2,..., K.max (= 10): .. done
Bootstrapping, b = 1,2,..., B (= 100)  [one "." per sample]:
.................................................. 50
.................................................. 100
>
> plot(gap_kmeans, main = "Gap Statistic: kmeans")
```



By looking at elbow in the plot, the optimal value of k is 3 which is the appropriate number of clusters for this dataset.

Let's cluster data using k-means with k as 3.

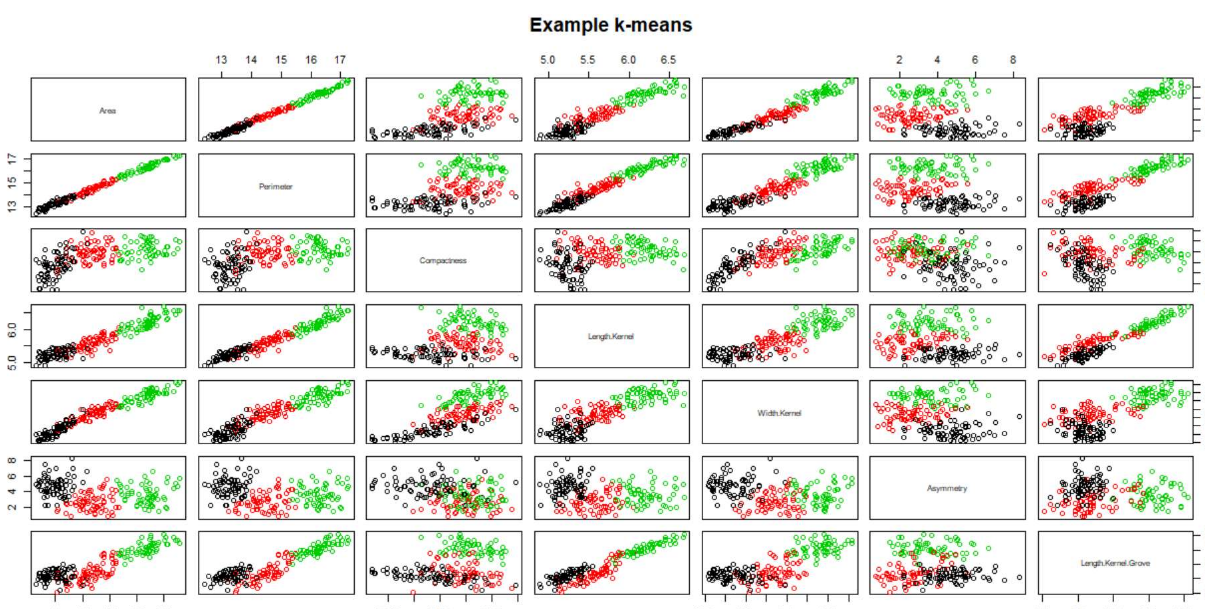
```
> #cluster using k means for k obtained from gap-stat
> km3 <- kmeans(data_final, centers = 3, nstart = 10)
```

Let's validate the result of k means clustering:

```
> table(km3$cluster, data$Seed.Group)

     A  B  C
  1  8  0 64
  2 57  9  1
  3  1 59  0
> rand.index(km3$cluster, as.numeric(data$Seed.Group))
[1] 0.8849805
> adj.rand.index(km3$cluster, as.numeric(data$Seed.Group))
[1] 0.7402708
```

	Rand Index	Adjusted Rand Index
Complete Linkage	0.803	0.5599
Average Linkage	0.876	0.72
Single Linkage	0.337	0.0002
K Means	0.8849	0.74



Conclusion:

K means clustering had better performance when compared to hierarchical -average linkage clustering. Clustering does not overlap much with observations.

K means clustering is better as data is less and finding number of centroids using GAP stats is efficient and applying k means on top of this data resulted in nearly a perfect cluster with Rand index of 0.88 which is better than other hierarchical methods.

I would suggest K-means algorithm for this data set which is efficient in terms of performance and cluster stability.