WQD7005 Individual Report Writing: Comparison of clustering algorithms using wine dataset

Baharul Hisyam bin Baharudin (S2039609)

Contents

Introduction

In this study, two clustering algorithms are assessed and evaluated, which are k-means clustering algorithm and k-medoids clustering algorithm. The description of the algorithms are briefly described below:

- K-means Clustering Algorithm This algorithm mainly aims to find similar clusters represented by variable k. For this purpose, this algorithm uses the mean or centroid as a metric to characterize the cluster. So, it divides n data points into k number of clusters and then each data point n belongs to appropriate cluster with the nearest possible centroid. Next, the Euclidean distance is accurately calculated from each data point n to the centroid in a given cluster. Always, the data points in a cluster are assigned to the centroid depending on the minimum euclidean distance from that centroid point. When there no data point is available to assign, an early grouping is considered. In such case, 'c' new centroids are re-calculated, thus new iteration continues until the 'c' centroids stop changing their position.

- K-medoids Clustering Algorithm
  K-medoid is a variant type of algorithm which is also termed as Partition Around Medoids (PAM). In this algorithm, data point act as a medoid within a cluster that are centrally located whose disparity over all data points in the cluster is minimal. Therefore, this medoid can be used as a representative of other data points within a cluster. The main core idea of PAM is to first calculate major data point as a medoid in a specific cluster, group the set of medoids, and then each data point is assigned to the nearest medoid in a given cluster. A new medoid is created by exchanging data points in the old medoid with the data points in a new non-medoid.

Data Selection

Load wine dataset from https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data

```
library(foreign)
wine_data <- read.table("https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data", sep
wine = wine_data[,-1] # first column is a class label
```

In the first steps of the analysis, the detailed information of the data, basic statistics and relationship between variables are explored.

```
cat("Number of different wines in the dataset:", nrow(wine))
```

```
## Number of different wines in the dataset: 178
```

```
cat("Number of features describing wines:", ncol(wine))
```

```
## Number of features describing wines: 13
```

```
summary(wine)
```

```
##        V2              V3              V4              V5
##  Min.   :11.03   Min.   :0.740   Min.   :1.360   Min.   :10.60
##  1st Qu.:12.36   1st Qu.:1.603   1st Qu.:2.210   1st Qu.:17.20
##  Median :13.05   Median :1.865   Median :2.360   Median :19.50
##  Mean   :13.00   Mean   :2.336   Mean   :2.367   Mean   :19.49
##  3rd Qu.:13.68   3rd Qu.:3.083   3rd Qu.:2.558   3rd Qu.:21.50
##  Max.   :14.83   Max.   :5.800   Max.   :3.230   Max.   :30.00
##        V6               V7              V8              V9
##  Min.   : 70.00   Min.   :0.980   Min.   :0.340   Min.   :0.1300
##  1st Qu.: 88.00   1st Qu.:1.742   1st Qu.:1.205   1st Qu.:0.2700
##  Median : 98.00   Median :2.355   Median :2.135   Median :0.3400
##  Mean   : 99.74   Mean   :2.295   Mean   :2.029   Mean   :0.3619
##  3rd Qu.:107.00   3rd Qu.:2.800   3rd Qu.:2.875   3rd Qu.:0.4375
##  Max.   :162.00   Max.   :3.880   Max.   :5.080   Max.   :0.6600
##       V10              V11              V12              V13
##  Min.   :0.410   Min.   : 1.280   Min.   :0.4800   Min.   :1.270
##  1st Qu.:1.250   1st Qu.: 3.220   1st Qu.:0.7825   1st Qu.:1.938
##  Median :1.555   Median : 4.690   Median :0.9650   Median :2.780
##  Mean   :1.591   Mean   : 5.058   Mean   :0.9574   Mean   :2.612
##  3rd Qu.:1.950   3rd Qu.: 6.200   3rd Qu.:1.1200   3rd Qu.:3.170
##  Max.   :3.580   Max.   :13.000   Max.   :1.7100   Max.   :4.000
##       V14
##  Min.   : 278.0
##  1st Qu.: 500.5
##  Median : 673.5
##  Mean   : 746.9
##  3rd Qu.: 985.0
##  Max.   :1680.0
```

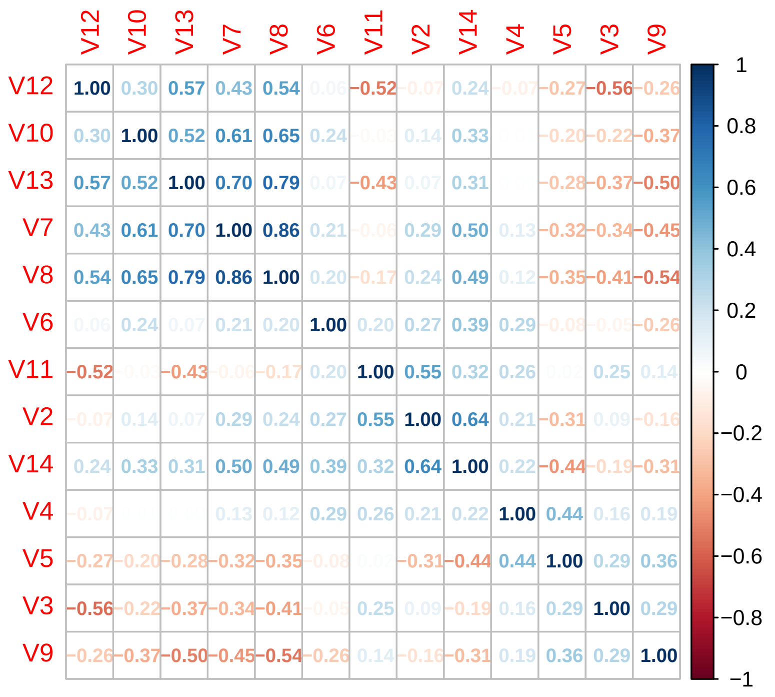
Data Pre-Processing

Since the values of variables are on different scales, normalize the values using scale() function in order to get proper and interpretable results.

```r
wine <- scale(wine)
```

Next step will be examining the relationship between variables.

```r
library(corrplot)
wine_matrix <- data.matrix(wine, rownames.force = NA)
M <- cor(wine_matrix)
corrplot(M, method = "number", number.cex = 0.75, order="hclust")
```



From the correlation matrix one can see that dataset contains some highly correlated features, eg. flavanoids and total phenols (0.86), OD280/OD315 of diluted wines and flavanoids (0.79), OD280/OD315 of diluted wines and total phenols (0.7), flavanoids and proanthocyanins (0.65) or proline and alcohol (0.64). The highest dependencies (over 0.6) are mainly positive.

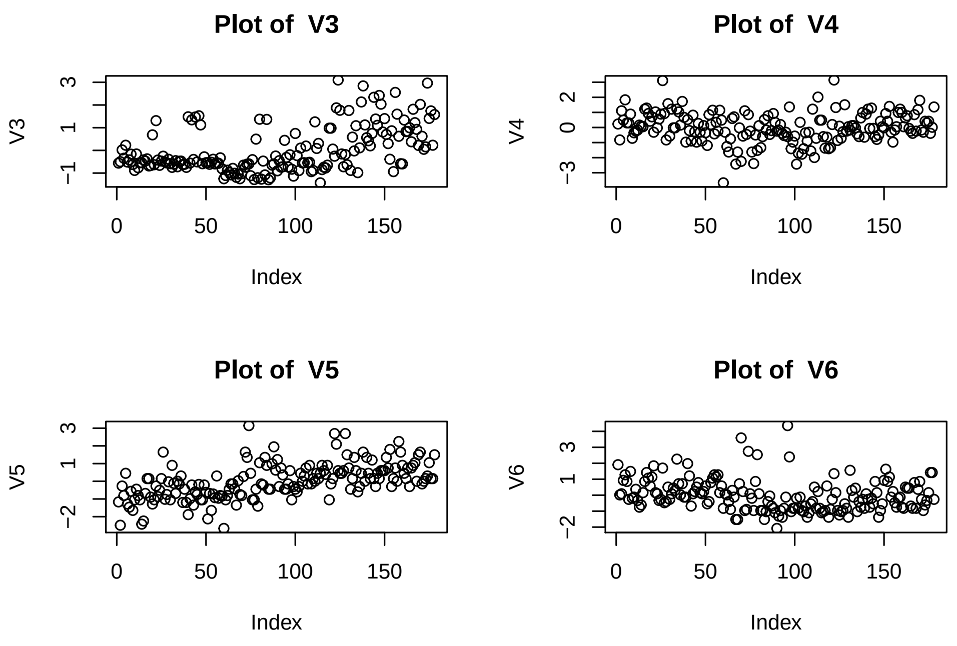
In other to check if the dataset contains outliers, one can calculate interquartile range statistic.

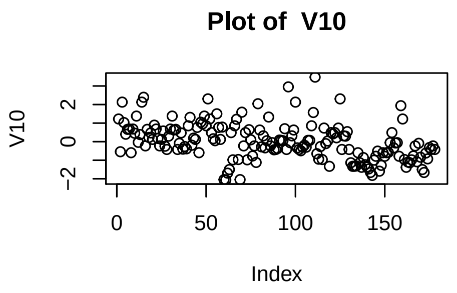
```r
vars <- c(colnames(wine))
Outliers <- c()
for(i in vars){
  max <- quantile(wine[,i], 0.75) + (IQR(wine[,i]) * 1.5 )
  min <- quantile(wine[,i], 0.25) - (IQR(wine[,i]) * 1.5 )
  idx <- which(wine[,i] < min | wine[,i] > max)
  print(paste(i, length(idx), sep=' ')) # printing variable and number of potential outliers
  Outliers <- c(Outliers, idx)
}
```

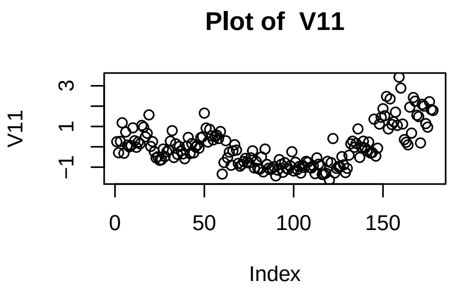
```
## [1] "V2 0"
## [1] "V3 3"
## [1] "V4 3"
## [1] "V5 4"
## [1] "V6 4"
## [1] "V7 0"
## [1] "V8 0"
## [1] "V9 0"
## [1] "V10 2"
## [1] "V11 4"
## [1] "V12 1"
## [1] "V13 0"
## [1] "V14 0"
```

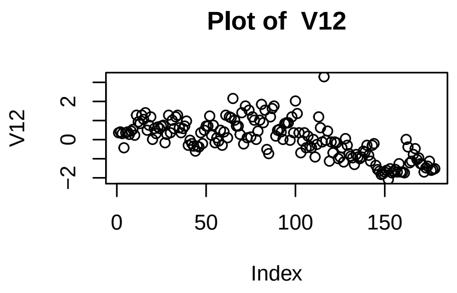
It seems that the statistic indicated up to four outliers in particular features. Let's now plot these variables which were indicated to contain outliers.

```r
par(mfrow=c(2,2))
colnames <- colnames(wine[,c(2:5,9:11)])
for (i in colnames) {
  plot(wine[,i], main = paste("Plot of ", i), ylab = i)
}
```









After examining the plots, the decision is not to exclude any of the observations from the analysis.

In order to choose the clustering algorithm best fitted to the data, one must consider characteristics of the dataset. From initial analysis, we know that we deal with continuous variables and most of them are correlated (mainly positively). Dataset doesn't contain worrisome outliers.

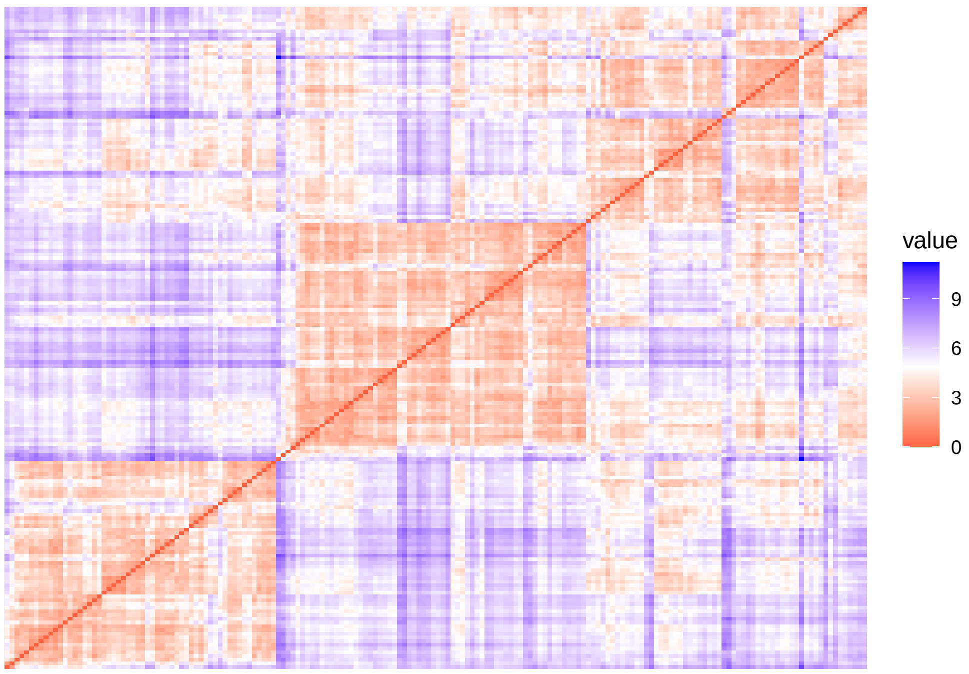
Data Transformation

To check if we need transformation, we run prediagnostics in order to check wheather data can be clustered and also to choose the optimal number of clusters.

In order to assess clusterability of the data, we will run Hopkins statistic. The null hypothesis tells that the dataset is uniformly distributed and does not contain meaningful clusters.

```
library(clustertend)
library(factoextra)
library(ggplot2)
get_clust_tendency(wine, 2, graph=TRUE, gradient=list(low="red", mid="white", high="blue"))
```

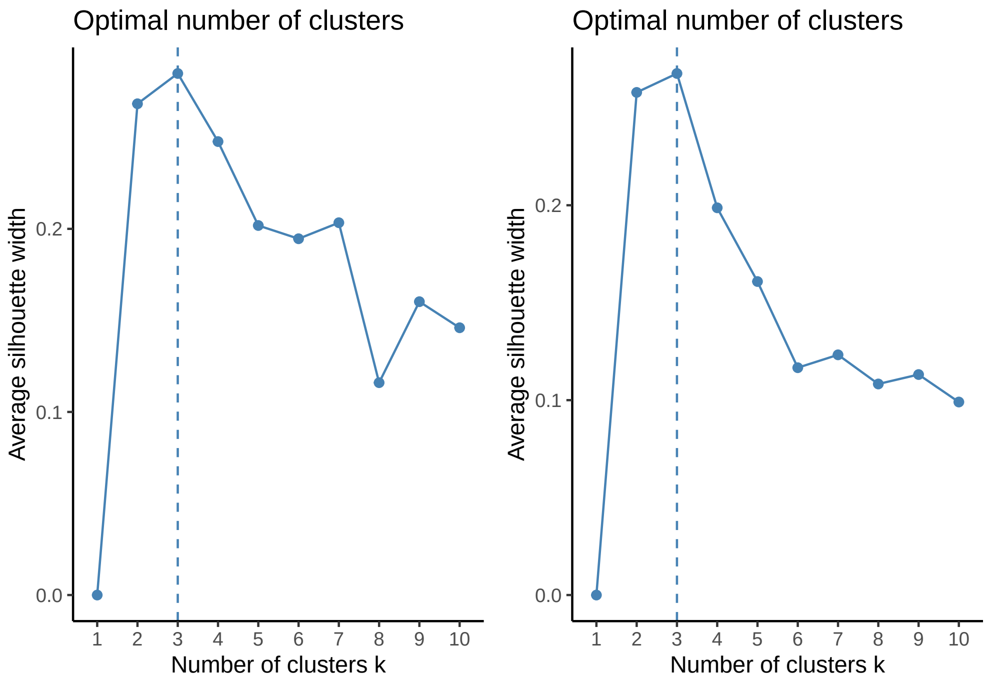
```
## $hopkins_stat
## [1] 0.6530508
##
## $plot
```



Hopkins statistc is equal to 0.65. As this value is close to 0.5, we can conclude that the dataset is random and can be further clustered. The same conclusion can be made based on based on the ordered dissimilarity plot. One can see fields of different colors which indicate that there is possibility of finding clusters in the data.

Next step will be finding the optimal number of clusters. To do this, we use silhouette statistic and apply it to k-means and PAM clustering algorithm.

```
library(gridExtra)
a <- fviz_nbclust(wine, FUNcluster = kmeans, method = "silhouette") + theme_classic()
b <- fviz_nbclust(wine, FUNcluster = cluster::pam, method = "silhouette") + theme_classic()
grid.arrange(a, b, ncol=2)
```



According to silhouette statistic, the optimal number of clusters is three for k-means and three for PAM.

K-Means clustering (Data Mining & Interpretation)

One thing is that need to be considered is the linear dependency of the variables, important step in flat clustering is to find the distance which will be proper for this kind of data. One can consider at least two of them: Pearson's correlation and Eucledian distance.

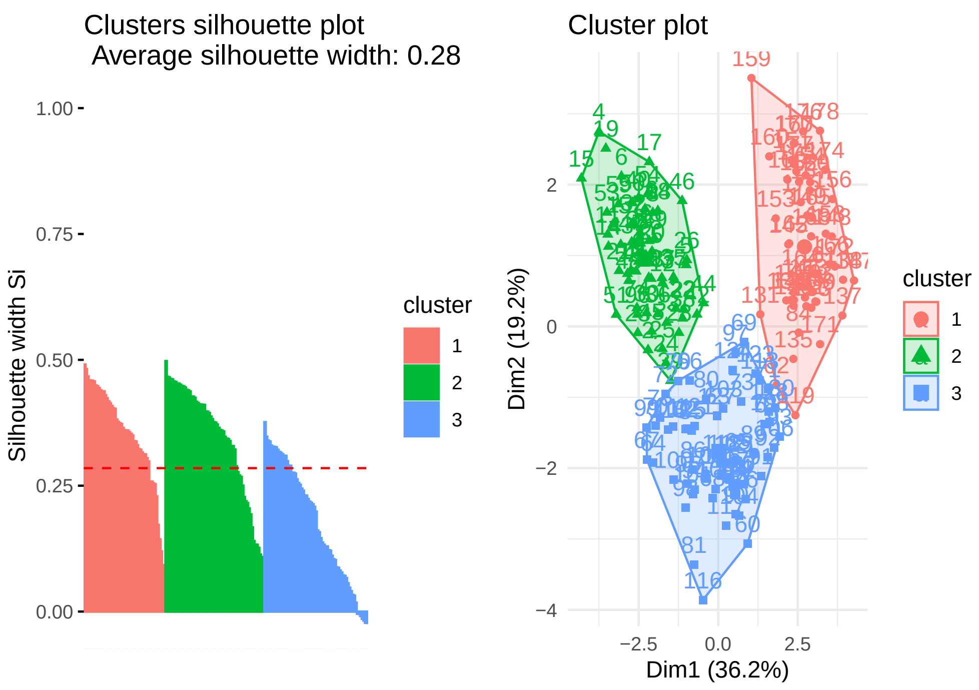
The basic concept of k-means is that it takes cluster centres (means) to represent cluster. Its goal is to minimize square error of the intra-class dissimilarity. It means that the algorithm aims to the situation in which clusters are consistent and different from each other.

K-Means Clustering Using Pearson Correlation:

```r
library(factoextra)
cl_kmeans <- eclust(wine, k=3, FUNcluster="kmeans", hc_metric="pearson", graph=FALSE)
a <- fviz_silhouette(cl_kmeans)
```

```
##   cluster size ave.sil.width
## 1       1   51          0.35
## 2       2   62          0.34
## 3       3   65          0.18
```

```r
b <- fviz_cluster(cl_kmeans, data = wine, elipse.type = "convex") + theme_minimal()
grid.arrange(a, b, ncol=2)
```



Efficiency testing of K-Mean Clustering using Pearson Correlation:

```
table(cl_kmeans$cluster, wine_data$V1)
```

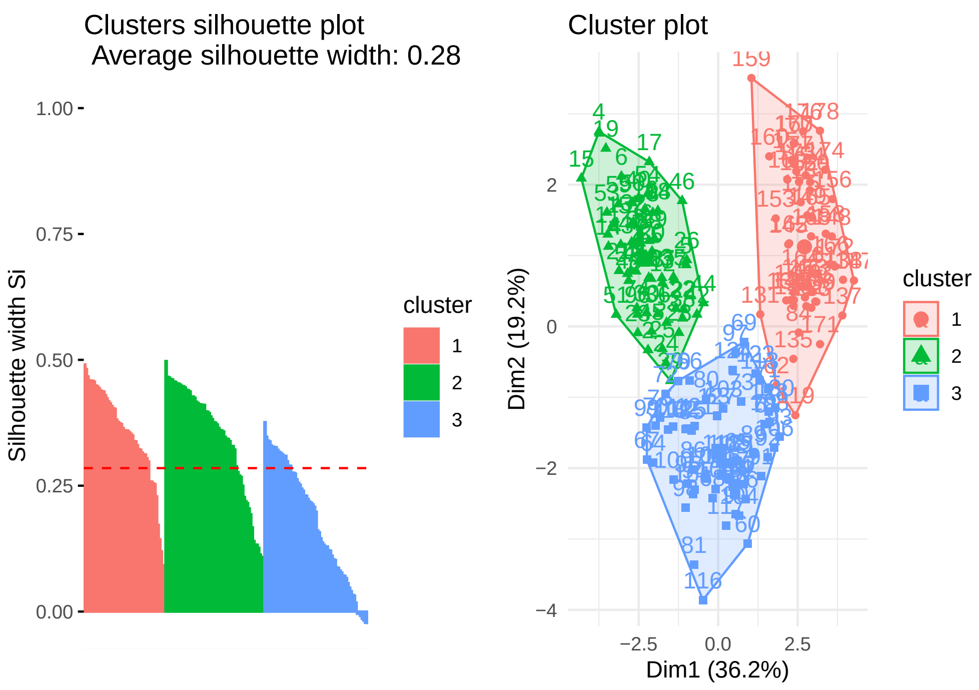
```
##
##       1  2  3
##    1  0  3 48
##    2 59  3  0
##    3  0 65  0
```

K-Means Clustering Using Euclidean Distance:

```
cl_kmeans1 <- eclust(wine, k=3, FUNcluster="kmeans", hc_metric="euclidean", graph=FALSE)
g <- fviz_silhouette(cl_kmeans1)
```

```
##   cluster size ave.sil.width
## 1       1   51          0.35
## 2       2   62          0.34
## 3       3   65          0.18
```

```
h <- fviz_cluster(cl_kmeans1, data = wine, elipse.type = "convex") + theme_minimal()
grid.arrange(g, h, ncol=2)
```



Efficiency testing of K-Mean Clustering using Euclidean Distance:

```
table(cl_kmeans1$cluster, wine_data$V1)
```

```
##
##      1  2  3
##   1  0  3 48
##   2 59  3  0
##   3  0 65  0
```

Summarizing the above results, it shows that there is basically no difference between Pearson's correlation and Euclidean distance in k-mean clustering algorithm using wine dataset. Silhouette statistic for both is equal to 0.28. As for accuracy, both methods seems to wrongly assign labels of class 2 and 3. Thus number of wrongly assigned wines is 6.

PAM clustering (Data Mining & Interpretation)

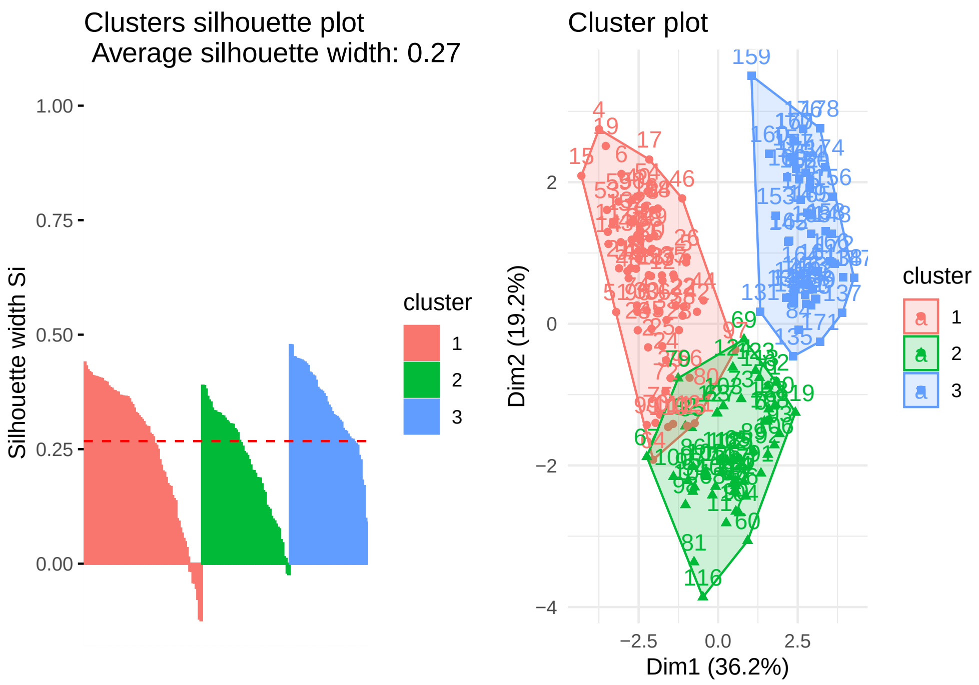
PAM is also called medoid-based method, which means that is chooses datapoints as centers (medoids) of the clusters. It is generally more robust to outliers than k-means since it uses median instead of mean.

PAM Clustering Using Pearson Correlation:

```
cl_pam <- eclust(wine, k=3, FUNcluster="pam", hc_metric="pearson", graph=FALSE)
c <- fviz_silhouette(cl_pam)
```

```
##   cluster size ave.sil.width
## 1       1   74          0.25
## 2       2   55          0.22
## 3       3   49          0.34
```

```
d <- fviz_cluster(cl_pam, data = wine, elipse.type = "convex") + theme_minimal()
grid.arrange(c, d, ncol=2)
```



Efficiency testing of PAM Clustering using Pearson Correlation:

```
table(cl_pam$cluster, wine_data$V1)
```

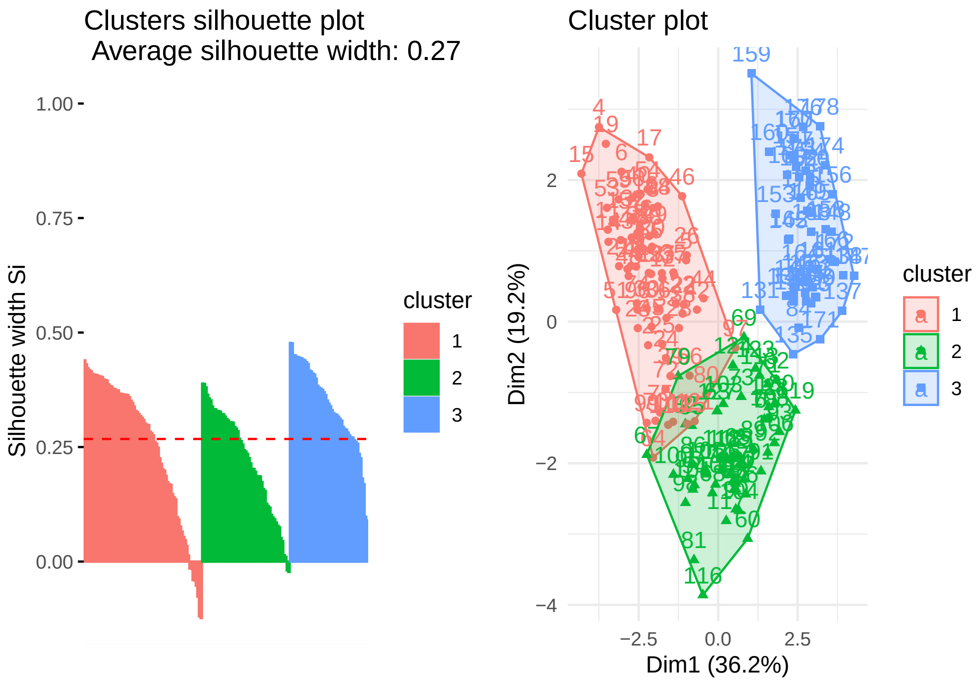
```
##
##       1  2  3
##    1 59 15  0
##    2  0 55  0
##    3  0  1 48
```

PAM Clustering Using Euclidean distance:

```
cl_pam1 <- eclust(wine, k=3, FUNcluster="pam", hc_metric="euclidean", graph=FALSE)
i <- fviz_silhouette(cl_pam1)
```

```
##   cluster size ave.sil.width
## 1       1   74          0.25
## 2       2   55          0.22
## 3       3   49          0.34
```

```
j <- fviz_cluster(cl_pam1, data = wine, elipse.type = "convex") + theme_minimal()
grid.arrange(i, j, ncol=2)
```



Efficiency testing of PAM Clustering using Euclidean Distance:

```
table(cl_pam1$cluster, wine_data$V1)
```

```
##
##      1  2  3
##   1 59 15  0
##   2  0 55  0
##   3  0  1 48
```

Comparing the above PAM implementaitons it occurs that there is also no difference between Pearson's correlation and Euclidean distance, just like in case of k-means. Silhouette statistic for both is equal to 0.27 and the number of wrongly assignes labels for both is 16.

Conclusion

To sum up and compare the results obtained with k-means and PAM, it occured that K-means algorithm with euclidian distance or pearson correlation performed the best in the case of efficiency, only 6 misclassification when compared to PAM, where the misclassification is 16. For average silhoutte width, both algorithms performs almost the same where k-means scores 0.28 while PAM scores 0.27.