KIT306/606 Tutorial 7

Student ID	Name

The following tutorial work should be completed by tutorial 8 (week9).

In this tutorial, we will learn how to apply clustering techniques in the data mining task. Clustering techniques are also called as 'Descriptive Modelling' or 'Unsupervised (machine) learning'.

The goal of clustering is finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups.

- No labels/classes in data
- Group data points into clusters based on how "near" they are to one another
- Identify structure in data

- **Datasets: Iris Plants Database**

The dataset we are using today is `iris` plant dataset, which is the best known database to be found in the pattern recognition (classification or clustering).

Iris Data Set

Download: Data Folder, Data Set Description

Abstract: Famous database; from Fisher, 1936

The Iris data set contains following 3 classes of 150 instances each, where each class refers to a type of iris plant.

Iris setosa

Iris versicolor

Iris virginica

In R, iris is pre-defined dataset so you can find the data by using the following command.

```
> iris
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
6            5.4         3.9          1.7         0.4     setosa
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
```

In last few weeks, we learned how to apply classification techniques in learning unique pattern from the data if it contains species (class/label)

But, what if the **classes or labels IS NOT AVAILBLE?**

```
> iris
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
6            5.4         3.9          1.7         0.4     setosa
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
```

Let's make a temp variable, called datairis, and remove the classes/labels (Species).

```
> datairis<-iris
> datairis$Species<-NULL
> datairis
    Sepal.Length Sepal.Width Petal.Length Petal.Width
1            5.1         3.5          1.4         0.2
2            4.9         3.0          1.4         0.2
3            4.7         3.2          1.3         0.2
4            4.6         3.1          1.5         0.2
5            5.0         3.6          1.4         0.2
6            5.4         3.9          1.7         0.4
7            4.6         3.4          1.4         0.3
8            5.0         3.4          1.5         0.2
9            4.4         2.9          1.4         0.2
10           4.9         3.1          1.5         0.1
```

In classification techniques, we learned patterns from training dataset (the historical dataset which contains class/label) and classify the testing dataset.

Now, we do not have any class for training dataset so no unique learned pattern. We can only group/cluster data points based on the only similarity of features.

- **Partitional Clustering: K means Clustering**

The **k-means algorithm** is an algorithm to cluster n objects based on attributes into k partitions, where k < n.
The following list contains the process of k-means clustering.

1. Decide the number of clusters k
2. Initialise the centre of the clusters
3. Attribute the closest cluster to each data point
4. Set the position of each cluster to the mean of all data points belonging to that cluster
5. Repeat steps 2-3 until convergence

Let's apply kmeans function and define K as **3**.

```
> kcluster<-kmeans(datairis,3)
```

```
> kcluster
K-means clustering with 3 clusters of sizes 62, 50, 38

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.901613    2.748387     4.393548    1.433871
2     5.006000    3.428000     1.462000    0.246000
3     6.850000    3.073684     5.742105    2.071053

Clustering vector:
  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [36] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [71] 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 3 3 3
[106] 3 1 3 3 3 3 3 3 1 1 3 3 3 3 1 3 1 3 1 3 3 1 1 3 3 3 3 3 1 3 3 3 3 1 3
[141] 3 3 1 3 3 3 1 3 3 1

Within cluster sum of squares by cluster:
[1] 39.82097 15.15100 23.87947
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
```

The following components are available to have a look at the detailed information of the created clusters.

- cluster: A vector of integers (from 1:k) indicating the cluster to which each point is allocated
- centers: A matrix of cluster centres
- totss: The total sum of squares
- withinss: Vector of within-cluster sum of squares, one component per cluster
- tot.withinss: Total within-cluster sum of squares
- betweenss: The between-cluster sum of squares
- size: The number of points in each cluster
- iter: The number of (outer) iterations
- ifault: integer: indicator of a possible algorithm problem – for experts

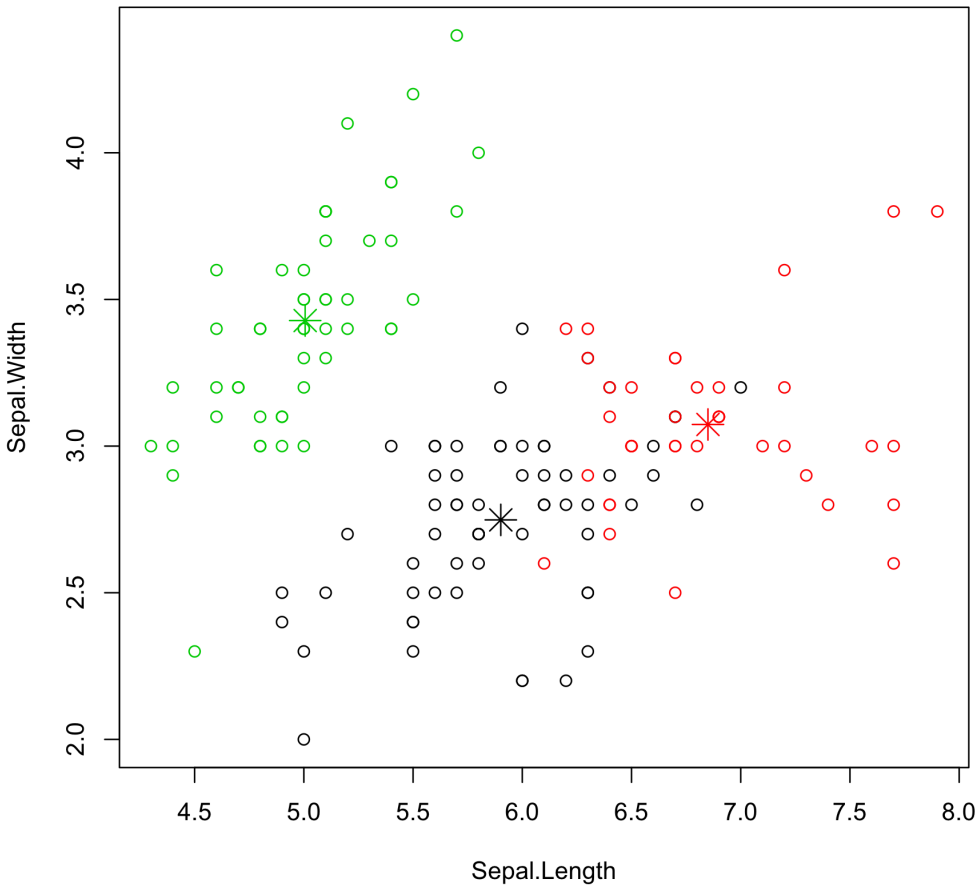
You can see the way clusters are grouped and the centre location using the following commands:

```
> kcluster$cluster
> kcluster$centers
```

Check the results.

By using those information, you can plot the data points and specify the clusters. Note that there are four dimensions in the data and that only the first two dimensions are used to draw the plot below. Some black points close to the red centre (asterisk) are actually closer to the black centre in the four dimensional space.

```
> plot(datairis[c("Sepal.Length", "Sepal.Width")], col=kcluster$cluster)
> points(kcluster$centers[,c("Sepal.Length", "Sepal.Width")], col=1:3, pch=8, cex=2)
```



Try different two features to plot the data points and you will see different view. However, it basically shows all the same view in the four dimensional space.

```
> plot(datairis[c("Petal.Length", "Petal.Width")], col=kcluster$cluster)
> points(kcluster$centers[,c("Petal.Length", "Petal.Width")], col=1:3, pch=8, cex=2)

> plot(datairis[c("Sepal.Length", "Petal.Length")], col=kcluster$cluster)
> points(kcluster$centers[,c("Sepal.Length", "Petal.Length")], col=1:3, pch=8, cex=2)

> plot(datairis[c("Sepal.Width", "Petal.Width")], col=kcluster$cluster)
> points(kcluster$centers[,c("Sepal.Width", "Petal.Width")], col=1:3, pch=8, cex=2)
```

Try different features, and see what happens ☺

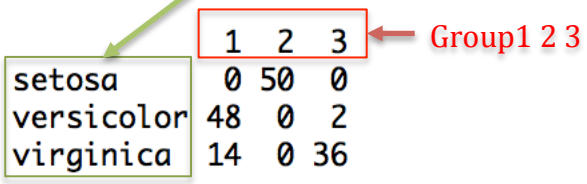
Normally, clustering result is very hard to evaluate since there is no class/label available in the dataset. Hence, it is difficult to say the data is well grouped or not.

HOWEVER, in this dataset, we hided the actual answer in iris dataset

```
> iris$Species
```

So you can check whether your clustering approach is actually working well as follows.

```
> table(iris$Species,kcluster$cluster)
```



- **Hierarchical Clustering**

Hierarchical clustering is builds a binary tree of the data that successively merges similar groups of points.

In order to see the clear and neat binary tree, it would be better to have small amount of dataset. So, let's select only 50 instance from the dataset.

dim() function and sample function would be helpful to randomly select the 50 dataset.

- **sample**: It takes a sample of the specified size from the elements of x using either with or without replacement.
- **dim**: It retrieve or set the dimension of an object.

```
> dim(iris)[1]
[1] 150
> random=sample(1:dim(iris)[1], 50)
```

And, you will see the following selected instances.

```
> random
 [1]  92  66  13  56   6 118 121 142  68  98  37  42  87  69 112  31 122 143  38 116 133 131 150 120
[25] 107 127  96 146  60 103 128  43 139 119  15 140 136  27  59  52  21 108 111  58 113  70 141 100
[49] 134  20
```

Let's make a temp variable, called datairis, and put randomly selected 50 instances ☺

```
> datairis=iris[random,]
```

Then, remove the classes/labels (Species) and prepare the dataset for clustering

```
> datairis$Species=NULL
```

```
> datairis
    Sepal.Length Sepal.Width Petal.Length Petal.Width
92           6.1         3.0          4.6         1.4
66           6.7         3.1          4.4         1.4
13           4.8         3.0          1.4         0.1
56           5.7         2.8          4.5         1.3
```

Now, we are ready to apply hierarchical clustering technique into this dataset.

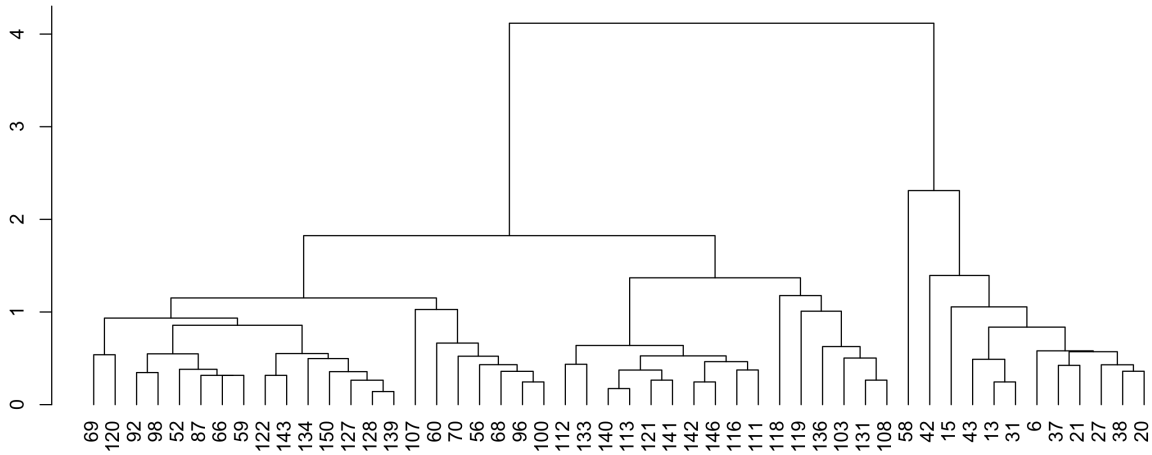
In order to use hierarchical clustering, it is important to calculate distance each data points. dist() function computes and returns the distance matrix computed by using the specified distance measure to compute the distances between the rows of a data matrix.

```
> hcluster<-hclust(dist(datairis), method="ave")

> hcluster

Call:
hclust(d = dist(datairis), method = "ave")

Cluster method   : average
Distance         : euclidean
Number of objects: 50
```

Let's plot the hcluster in order to check how those data points are grouped.

```
> plot(hcluster,hang=-1)
```

- hang: The fraction of the plot height by which labels should hang below the rest of the plot. A negative value will cause the labels to hang down from 0.



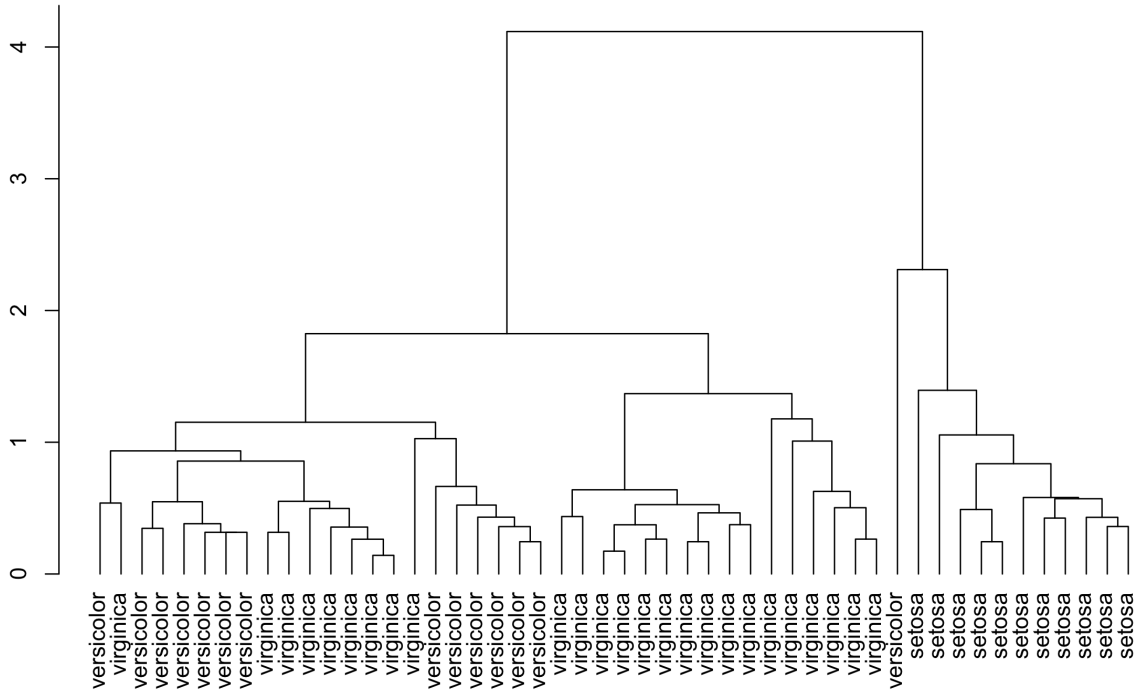
As mentioned before, clustering result is very hard to evaluate since there is no class/label available in the dataset. Hence, it is difficult to say the data is well grouped or not.

HOWEVER, in this dataset, we hided the actual answer in iris dataset

```
> iris$Species
```

So you can check whether your clustering approach is actually working well as follows.

```
> plot(hcluster,hang=-1,labels=iris$Species[random])
```



The result of labels and clusters comparison can show the performance of clustering technique.

Hope you understood the concept of using k-means and hierarchical clustering techniques.

In the following weeks, we will learn how to apply those clustering techniques in image recognition and text clustering (Clustering with Application).

Tutorial 7 Question

The data we will use for tutorial 7 is Blood Transfusion Service Centre Data Set.
https://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center

Blood Transfusion Service Center Data Set

Download: Data Folder, Data Set Description

Abstract: Data taken from the Blood Transfusion Service Center in Hsin-Chu City in Taiwan -- this is a classification problem.

Data Set Characteristics:	Multivariate	**Number of Instances:**	748	**Area:**	Business
Attribute Characteristics:	Real	**Number of Attributes:**	5	**Date Donated**	2008-10-03
Associated Tasks:	Classification	**Missing Values?**	N/A	**Number of Web Hits:**	93258

Click Data Folder link and you will see the following link.

Name	Last modified	Size	Description
Parent Directory		-	
transfusion.data	30-Nov-2008 00:14	13K	
transfusion.names	30-Nov-2008 00:17	3.1K	

Apache/2.2.15 (CentOS) Server at archive.ics.uci.edu Port 80

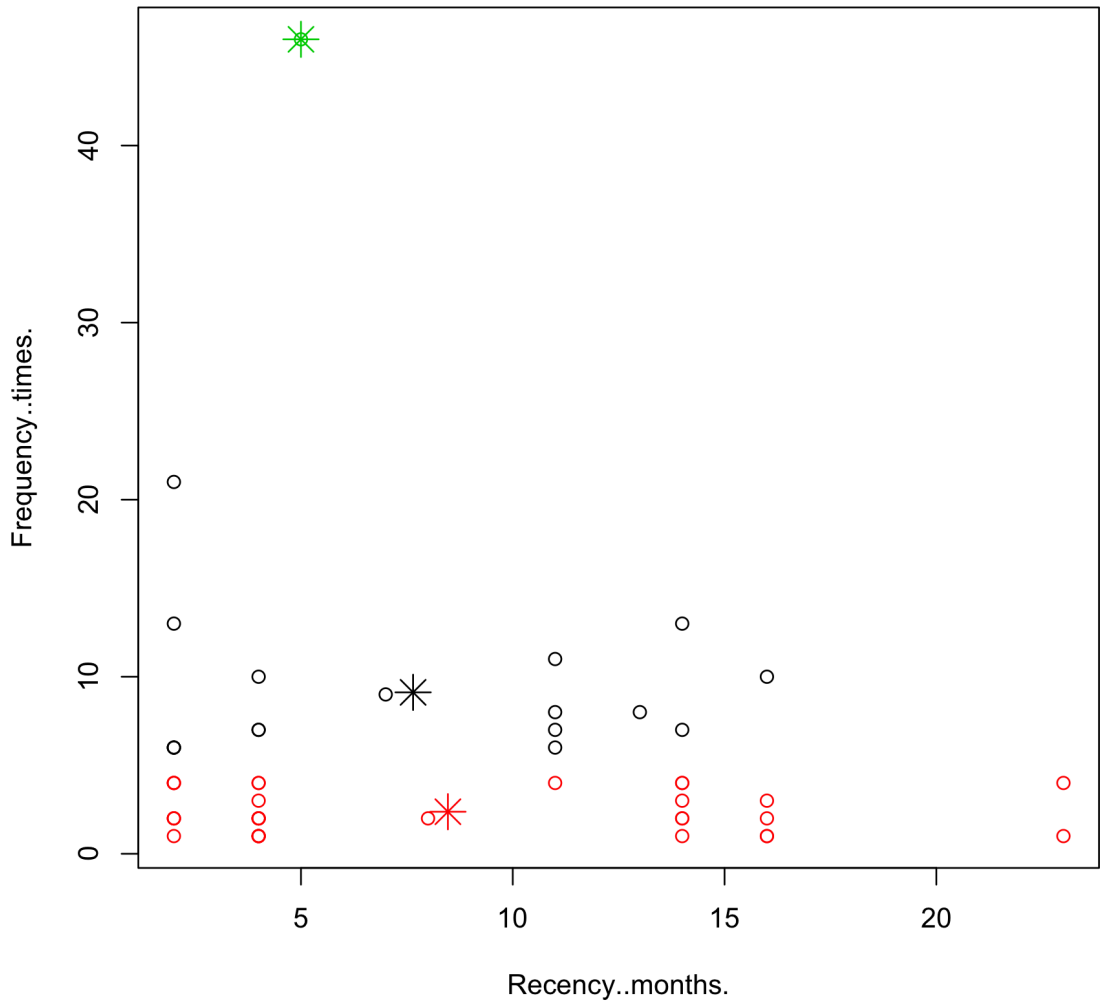
1. **Download the data set from transfusion.data by using read.table command. The downloaded dataset should be saved in the variable 'blood'**

(TIP: you can use **sep=","** and **header=TRUE** in **read.table** function)

2. **Randomly select only 50 instances from the dataset using dim and sample function. The randomly selected 50 instances should be saved in the variable 'datablood'**

3. **Remove class (whether.he.she.donated.blood.in.March.2007) from the data**

4. **Run k-means clustering using the dataset (K=3) and plot it as follows:**


Frequency..times.
0
10
20
30
40
5
10
15
20
Recency..months.