**Description of the dataset**

The dataset below contains data about the thyroid gland originally owned by Danny Coomans of the Dept. of Maths. and Stats. of James Cook University, Townsville 4811, Australia. It was donated by Stefan Aeberhard of the Dept. of Comp. Science of the same university in July of 1992.

The data was collected by running 5 different tests on to predict a patient's thyroid to the class euthyroidism, hypothyroidism or hyperthyroidism. The 5 attributes measured were T3-resin uptake, total serum thyroxin, total serum triiodothyronine, basal thyroid-stimulating hormone, and the maximal absolute difference of TSH value after injection of 200 micro grams of thyrotropin-releasing hormone as compared to the basal value. All 5 attribute values are continuous and there were no missing attribute values.

The tests were run on 215 patients giving a total of 215 instances of data divided into 3 classes. Class 1 which represents normal thyroid glands has 150 instances. Class 2 which represents hyperthyroidism has 35 instances while Class 3 which represents hypothyroidism has the final 30 instances of data.

**Brief description of the chosen machine learning methods**

**K-Means Clustering**

K-Means Clustering is an iterative clustering algorithm that partitions the dataset into K distinct subgroup or clusters where each data point only belongs in one group. It assigns data points to clusters where the Euclidean distance between the data point and the centroid of the cluster are at a minimum. The algorithm tries to make clusters as different from each other as possible while ensuring that data points within the same cluster are as similar as possible.

In K-Means Clustering, we first specify K number of clusters. The initial centroids of the cluster are chosen randomly after shuffling the dataset. Then, we run the following steps iteratively. The Euclidean distance between the data point and each centroid is calculated and the data point is assigned to the nearest cluster. The new centroid is then calculated by taking the average of each data point in the cluster. The algorithm is run until there is no change in the assignment of data points to clusters. We now have our K clusters of data points.

**Expectation Maximization Clustering**

Expectation Maximization Clustering is also an iterative clustering algorithm. It is used to infer the values of latent variables which are not directly observable from other known values. We start with a set of initial values which are incomplete observed data from a specific model. The algorithm cycles between two steps, the Estimation step or E-step and the Maximization step or M-step. In the estimation step, the values of the missing variables are estimated. In the maximization step, the parameters of the model are maximized to best explain the data. The algorithm continues until convergence occurs.

**Hierarchical Clustering**

Hierarchical Clustering is a clustering algorithm which aims to build a hierarchy of clusters. There are two main approaches to hierarchical clustering which are agglomerative or bottom-up and divisive or top-down. In the agglomerative approach, each data point is initially considered as a cluster. The two closest clusters are merged and then the proximity matrix is recalculated. The algorithm repeats this process until only K clusters remain. For the divisive approach, all the data points are initially considered as one cluster. We then slowly separate points which are not similar into their own cluster. The algorithm repeats until K clusters are formed.

**Experimental design**

The experiment was run using Weka Experimenter. The two clustering algorithms compared are K-Means Clustering and Hierarchical Clustering. Before we can start the experiments, we must first pre-process the dataset.

First, we must clean the data. Most algorithms do not work well in the presence of outliers so we will be removing them first. In Weka, we choose the InterquartileRange filter from the list of unsupervised attribute filters. To remove the outlier, we use the RemoveWithValues filter from the list of unsupervised instance filters. We specify the row and the column of the outliers and apply the filter. Now, the outliers have been removed. We then remove the outliers column from the dataset using the Remove filter.

Second, we must deal with any missing data. Normally, we either remove the data points with missing data or replace the missing data with some predefined values. Luckily, there are no data in the new-thyroid dataset, so we can skip this step.

Finally, we normalise the data to the range of 0 – 1. This is to ensure that the unit of dimension of the attributes of the data points do not distort the results of the data. As seen here, the first attribute has a range of 100 while the others only have a range of 1. We normalise the data by choosing the Normalise filter from the list of unsupervised attribute filters.