EXPERIMENT NO: 09

Aim: Apply Cobweb , EM, Farthest First algorithm on Banking data set.

Theory:

Cobweb:COBWEB is an incremental system for hierarchical conceptual clustering. COBWEB was invented by Professor Douglas H. Fisher.

COBWEB incrementally organizes observations into a classification tree. Each node in a classification tree represents a class (concept) and is labeled by a probabilistic concept that summarizes the attribute-value distributions of objects classified under the node. This classification tree can be used to predict missing attributes or the class of a new object.[3]

There are four basic operations COBWEB employs in building the classification tree. Which operation is selected depends on the category utility of the classification achieved by applying it. The operations are:

Merging Two Nodes

Merging two nodes means replacing them by a node whose children is the union of the original nodes' sets of children and which summarizes the attribute-value distributions of all objects classified under them.

Splitting a node

A node is split by replacing it with its children.

Inserting a new node

A node is created corresponding to the object being inserted into the tree.

Passing an object down the hierarchy

Effectively calling the COBWEB algorithm on the object and the subtree rooted in the node.

The COBWEB Algorithm

```
COBWEB(root, record):
Input: A COBWEB node root, an instance to insert record
if root has no children then
   children := {copy(root)}
   newcategory(record) \\ adds child with record's feature values.
   insert(record, root) \\ update root's statistics
else
```

```
    insert(record, root)
    for child in root's children do
      calculate Category Utility for insert(record, child),
      set best1, best2 children w. best CU.
    end for
    if newcategory(record) yields best CU then
      newcategory(record)
    else if merge(best1, best2) yields best CU then
      merge(best1, best2)
      COBWEB(root, record)
    else if split(best1) yields best CU then
      split(best1)
      COBWEB(root, record)
    else
      COBWEB(best1, record)
    end if
  end
```

EM Algorithm:

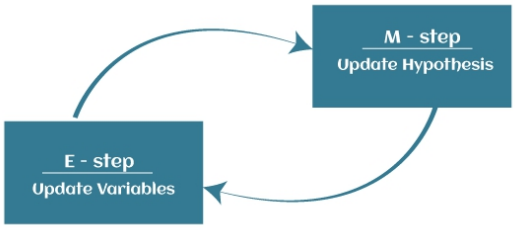
The Expectation-Maximization (EM) algorithm is defined as the combination of various unsupervised machine learning algorithms, which is used to determine the **local maximum likelihood estimates (MLE)** or **maximum a posteriori estimates (MAP)** for unobservable variables in statistical models. Further, it is a technique to find maximum likelihood estimation when the latent variables are present. It is also referred to as the **latent variable model.**

A latent variable model consists of both observable and unobservable variables where observable can be predicted while unobserved are inferred from the observed variable. These unobservable variables are known as latent variables.

EM Algorithm

The EM algorithm is the combination of various unsupervised ML algorithms, such as the **k-means clustering algorithm.** Being an iterative approach, it consists of two modes. In the first mode, we estimate the missing or latent variables. Hence it is referred to as the **Expectation/estimation step (E-step).**

Further, the other mode is used to optimize the parameters of the models so that it can explain the data more clearly. The second mode is known as the **maximization-step or M-step.**



- **Expectation step (E - step):** It involves the estimation (guess) of all missing values in the dataset so that after completing this step, there should not be any missing value.
- **Maximization step (M - step):** This step involves the use of estimated data in the E-step and updating the parameters.
- **Repeat E-**step and M-step until the convergence of the values occurs.

The primary goal of the EM algorithm is to use the available observed data of the dataset to estimate the missing data of the latent variables and then use that data to update the values of the parameters in the M-step.

Farthest First Algorithm:

A farthest-first traversal is a sequence of points in a compact metric space, with each point appearing at most once. If the space is finite, each point appears exactly once, and the traversal is a permutation of all of the points in the space. The first point of the sequence may be any point in the space. Each point p after the first must have the maximum possible distance to the set of points earlier than p in the sequence, where the distance from a point to a set is defined as the minimum of the pairwise distances to points in the set. A given space may have many different farthest-first traversals, depending both on the choice of the first point in the sequence (which may be any point in the space) and on ties for the maximum distance among later choices

Farthest-point traversals may be characterized by the following properties. Fix a number k, and consider the prefix formed by the first k points of the farthest-first traversal of any metric space. Let r be the distance between the final point of the prefix and the other points in the prefix. Then this subset has the following two properties:

- All pairs of the selected points are at distance at least r from each other, and

- All points of the metric space are at distance at most r from the subset.

Farthest first algorithm proposed by Hochbaum and Shmoys 1985 has same procedure as kmeans, this also chooses centroids and assign the objects in cluster but with max distance and initial seeds are value which is at largest distance to the mean of values, here cluster assignment is different, at initial cluster we get link with high Session Count, like at cluster-0 more than in cluster-1, and so on. Farthest first algorithm need less adjustments and basic for this explained in [15]. Working as described here, it also defines initial seeds and then on basis of „k" number of cluster which we need to know prior. In farthest first it takes point P i then chooses next point P i which is at maximum distance. P i is centroid and p1, p2, ……., pn are points or objects of dataset belongs to cluster from equation

$$\min\{\max dist(p_i, p_1), \max dist(p_i, p_2).....\}$$

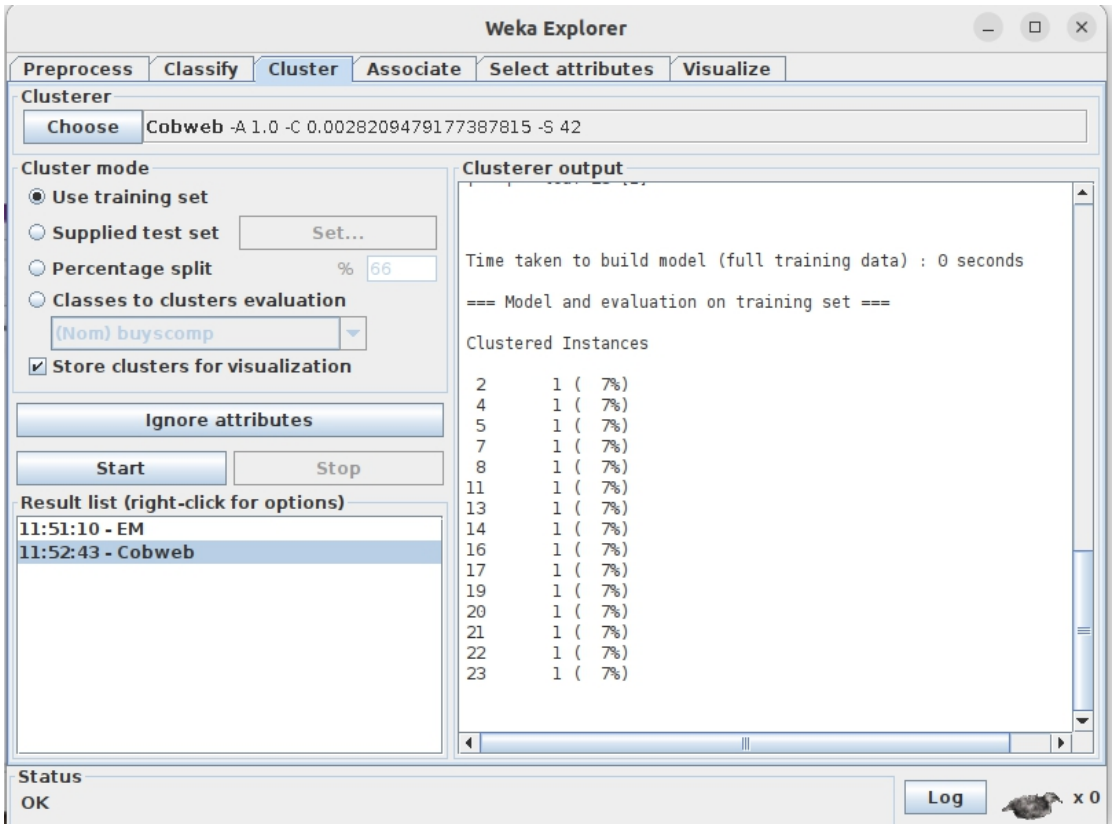
Farthest first actually solves problem of k-centre and it is very efficient for large set of data.In farthest first algorithm we are not finding mean for calculating centroid ,it takes centrod arbitrary and distance of one centroid from other is maximum figure-2 shows cluster assignment using farthest –first. When we performed outlier detection for our dataset we get which objects is outlier.

Procedure:

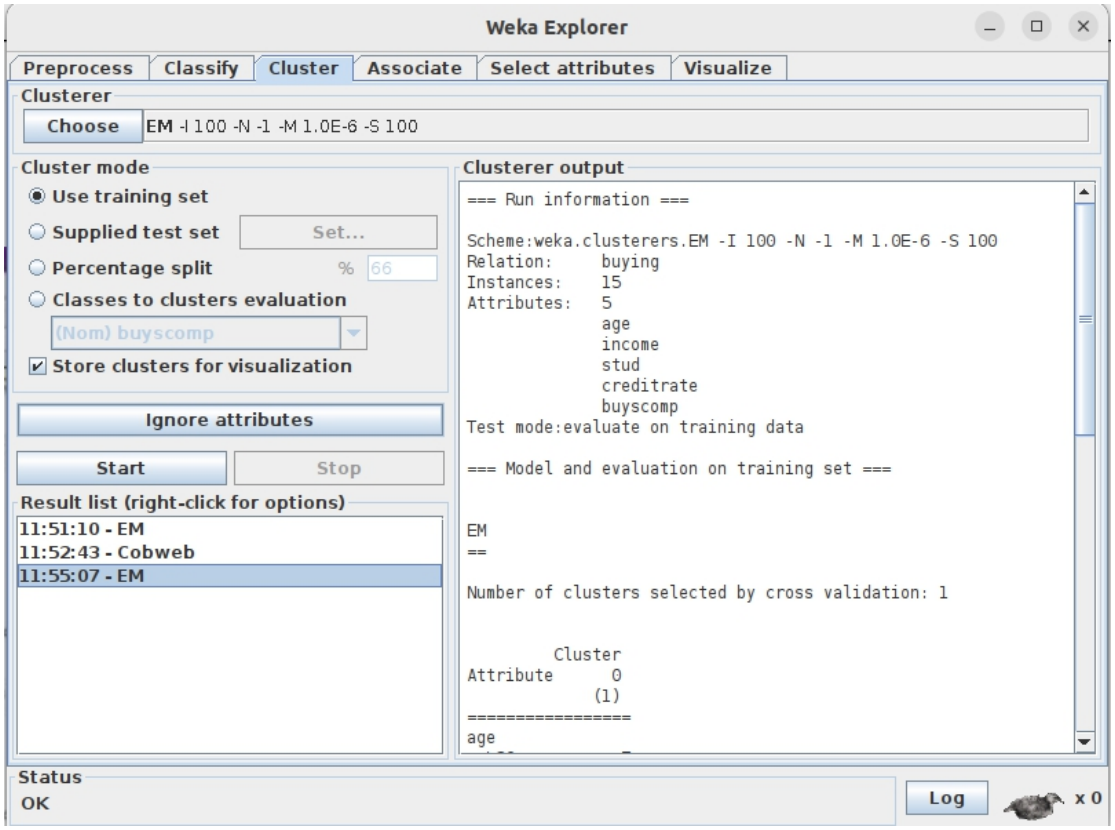
1. Open the weka tool.
2. Download a dataset by using UCI.
3. Apply replace missing values.
4. Apply normalize filter.
5. Click the cluster tab.
6. Apply all algorithms one by one.
7. Find the no of clusters that are formed
8. Note the output.

Output:

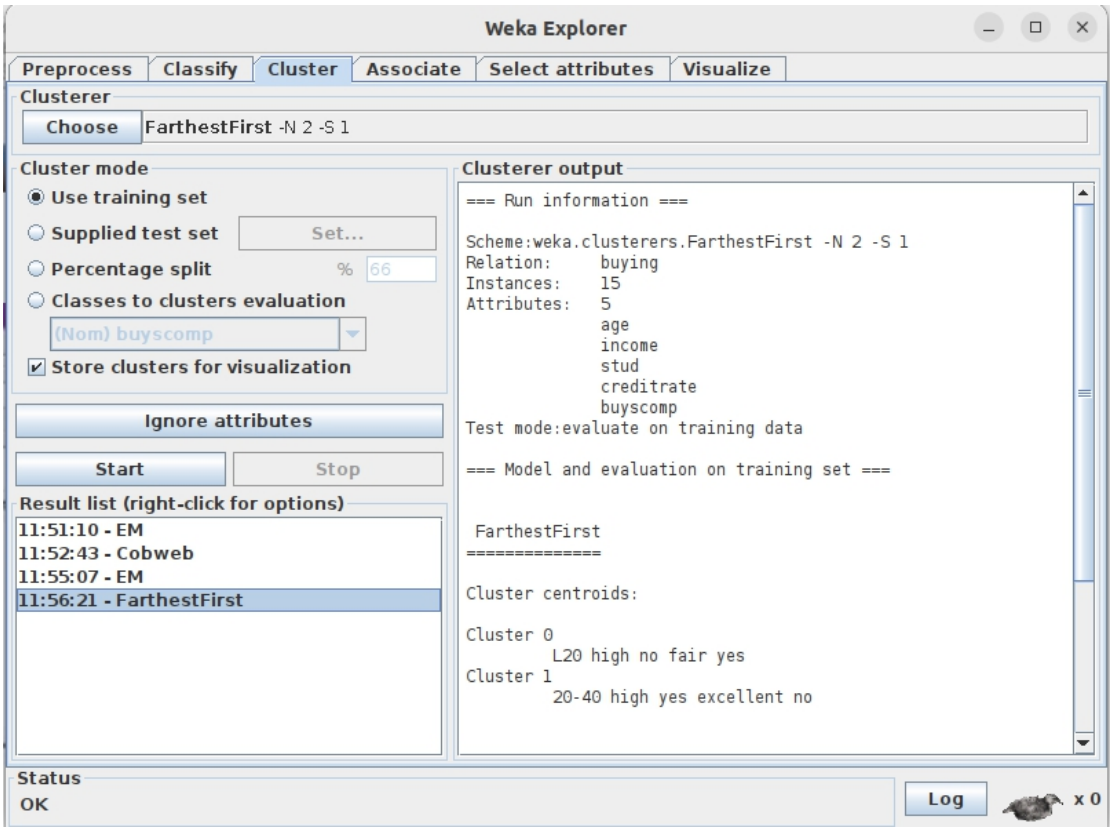
Cobweb:


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
Cobweb -A 1.0 -C 0.0028209479177387815 -S 42
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) buyscomp
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
11:51:10 - EM
11:52:43 - Cobweb
Clusterer output
Time taken to build model (full training data) : 0 seconds
=== Model and evaluation on training set ===
Clustered Instances
2 1 (7%)
4 1 (7%)
5 1 (7%)
7 1 (7%)
8 1 (7%)
11 1 (7%)
13 1 (7%)
14 1 (7%)
16 1 (7%)
17 1 (7%)
19 1 (7%)
20 1 (7%)
21 1 (7%)
22 1 (7%)
23 1 (7%)
Status
OK
Log
x 0

EM:


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
EM -I 100 -N -1 -M 1.0E-6 -S 100
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) buyscomp
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
11:51:10 - EM
11:52:43 - Cobweb
11:55:07 - EM
Clusterer output
=== Run information ===
Scheme:weka.clusterers.EM -I 100 -N -1 -M 1.0E-6 -S 100
Relation: buying
Instances: 15
Attributes: 5
age
income
stud
creditrate
buyscomp
Test mode:evaluate on training data
=== Model and evaluation on training set ===
EM
==
Number of clusters selected by cross validation: 1
Cluster
Attribute 0
(1)
age
Status
OK
Log
x 0

Farthest First:



Result:

This program has been successfully executed.

Viva Questions:

1. What is the use of COBWEB clustering?
2. What is EM algorithm used for?
3. What are the steps of EM algorithm?
4. What is the use of farthest first algorithm?