

ASSIGNMENT 5

AIM: Implement Decision trees on Digital Library Data to mirror more titles (PDF) in the library application, compare it with Naive Bayes algorithm.

OBJECTIVE:

- To understand the basic concept to K-means Clustering.
- To implement K-means clustering algorithm using Java.

SOFTWARE REQUIREMENTS:

- Linux Operating System
- Java compiler
- Weka Tool

MATHEMATICAL MODEL:

Consider a set S consisting of all the elements related to a program.

The Mathematical model is given as below,

$S = \{s, e, X, Y, Fme, DD, NDD, Mem\}$ shared

Where,

s = Initial State

e = End State

X = Input. Here it is number of elements, actual element, number of clusters

Y = Output. Here output is final cluster by K-means.

Fme = Algorithm/Function used in program. f = $\text{fcalmean}()$, c = $\text{caldif}()$ DD = Deterministic Data

NDD = Nondeterministic Data

Mem_{shared} = Memory shared by processor.

THEORY:

Decision Tree:

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal.

K-means Clustering:

k-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k center, one for each cluster. These centers should be placed in a cunning way because of different location

causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belong in given data set and assign it to the nearest center. When no point is pending, the first step is completed and an early group age is done. At this point we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new center. A loop has been generated. As a result of this loop we may notice that the k centers change their location step by step until no more changes are done or in other words centers do not move anymore. Finally, this algorithm aims at minimizing an objective function known as squared error function given by: $J(V) = \sum_{i=1}^c \sum_{x \in C_i} \|x - v_i\|^2$ (1) Where, $\|x - v_i\|$ is the Euclidean distance between x and v_i . ' c ' is the number of data points in i th cluster. ' c ' is the number of cluster centers.

Algorithmic steps for k-means clustering:

Let $X = \{x_1, x_2, x_3, \dots, x_n\}$ be the set of data points and $V = \{v_1, v_2, \dots, v_c\}$ be the set of centers.

1. Randomly select ' c ' cluster centers.
2. Calculate the distance between each data point and cluster centers.
3. Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
4. Recalculate the new cluster center using: $V_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ (2) where, ' c ' represents the number of data points in i th cluster.
5. Recalculate the distance between each data point and new obtained cluster centers.
6. If no data point was reassigned then stop, otherwise repeat from step 3.

Advantages:

1. Fast, robust and easier to understand.
2. Relatively efficient: $O(tkn d)$, where n is objects, k is clusters, d is dimension of each object, and t is iterations. Normally, $k, t, d \ll n$.
3. Gives best result when dataset are distinct or well separated from each other.

Disadvantages:

1. The learning algorithm requires a priori specification of the number of cluster centers.
2. The use of Exclusive Assignment-If there are two highly overlapping data then k-means will not be able to resolve that there are two clusters.
3. The learning algorithm is not invariant to non-linear transformations i.e. with different representation of data we get different results (data represented in form of cartesian co-ordinates and polar coordinates will give different results).

4. Euclidean distance measures can unequally weight underlying factors.
5. The learning algorithm provides the local optima of the squared error function.
6. Randomly choosing of the cluster center cannot lead us to the fruitful result.
7. Applicable only when mean is defined i.e.fails for categorical data.
8. Unable to handle noisy data and outliers.
9. Algorithm fails for non-linear dataset.

K-means : Step-By-StepExample

As a simple illustration of a k -means algorithm, consider the following dataset consisting of the scores of two variables on each of seven individuals: This dataset is to be grouped into two clusters. As a first step in finding a sensible initial partition, let the A and B values of the two individuals furthest apart (using the Euclidean distance measure), define the initial cluster means, giving: The remaining individuals are now examined in sequence and allocated to the cluster to which they are closest, in terms of Euclidean distance to the cluster mean. The mean vector is recalculated each time a new member is added. This leads to the following series of steps: Now the initial partition has changed, and the two clusters at this stage having the following characteristics:[But we cannot yet be sure that each individual has been assigned to the right cluster. So, we compare each individual's distance to its own cluster mean and to that of the opposite cluster. And we find: Only individual 3 is nearer to the mean of the opposite cluster(Cluster2)than its own (Cluster1).In other words, each individual's distance to its own cluster mean should be smaller than the distance to the other cluster's mean(which is not the case with individual3). Thus, individual 3 is relocated to Cluster 2resulting in the new partition: The iterative relocation would now continue from this new partition until no more relocations occur. However, in this example each individual is now nearer its own cluster mean than that of the other cluster and the iteration stops, choosing the latest partitioning as the final cluster solution. Also, it is possible that the k-means algorithm won't find a final solution. In this case it would be a good idea to consider stopping the algorithm after a pre-chosen maximum of iterations.

Testing for all functions (For verify input, analyze expected output):

Test Case No.	Function	Input	Output	Expected
1	Accept() Node Elements No of Cluster	Integer(nodes) Integer(elements) Integer (clusters)	Find New Centroid till previous value and latest value of centroid is same	Nodes are initialized and assigned to clusters
2	Cal _mean()	Elements, cluster values	New centroid	Calculate new centroid and assign to clusters
3	Display()	Cal _mean() result	Final centroid	Display final centroid values

PSEUDOCODE:

Pseudo code for calmean() :static
void calmean()

```
int  
temp1=0;for(int i=0;i<p;++i)  
if(a[i]<m[i])dif[i]=am[i];  
else if [i]=m[i]-a;  
  
int val=0;  
double  
temp=dif[0];for(int i=0;i<p;++i)  
  
if(dif[i]<temp)  
  
temp=dif[i];  
val=i;  
  
return val;
```

CONCLUSION: Thus, we have implemented K-means clustering algorithm using Java.

Roll No.	Name of Student	Date of Performance	Date of Submission	Sign.
BECOC357	Sunny Shah	31 / 08 / 2017	21 / 09 / 2017	