**IDA Assignment 2**

**Sampath Dontharaju**

**M08879603**

**Attached the code file which consists of solutions for all the questions. However explained the code in parts for each question is explained briefly**

****

**1)** split the dataset into three parts by random selection: 13020 records for training, 3000 records

for validation and tuning, and 3000 records for testing. You can read the attached .xlsx file by

using the Matlab command: data=xlsread(‘Magic04.xlsx’);

**solution )** Read the document using **xlsread()** function

then used the random function **randperm** to get the random numbers of 19020 and then splitted the data into 3 partitions  **Train\_data(1-13020) , Val\_data(13021 to 16020), test\_data (16021,end).**

**used the following code**

week2\_data = xlsread('D:\Assignments UC sem1\IDA\Assignment 2\magic04.xlsx');

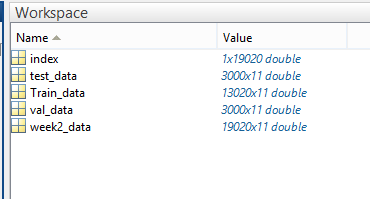
index=randperm(19020);

Train\_data=week2\_data(index(1:13020),:);

val\_data = week2\_data(index(13021:16020),:);

test\_data = week2\_data(index(16021:end),:);

**output :after executing first code you can see the result in workspace as follows**



**2) split the training data into two tables: the first ten columns contained in a “Features” table and**

**the last one column contained in the “ClassLabels” vector.**

**sol)**  i have split the training data to features and classlabels using the code below

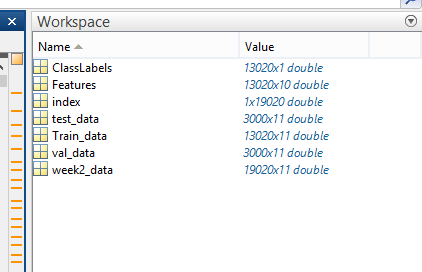
**code :**

**Features = Train\_data(:,1:10);**

**ClassLabels = Train\_data(:,11);**

**output screen shot:**

you can find the classified **Features and ClassLabels in the output workspace as follows**

****

3) In Matlab environment, use fitctree command to generate decision tree (dtr) as follows: dtr=fitctree(Features, ClassLabels, ‘MinLeafSize’, <N>); In this command MinLeafSize parameter specifies the minimum number of records, ‘N’, in each leaf node of the generated decision tree.

If N is set to a high number, say 1200, then a node will not be split to grow the tree if its splitting results in a child node containing fewer than 1200 records. Therefore, high values of N will result in shallower trees and small values of N will result in deeper trees. You can view the generated decision trees by using the commands view(dtr) or view(dtr, ‘Mode’, ‘graph’).

**sol)** decTreeOne = fitctree(Features,ClassLabels,'MinLeafSize',1200);

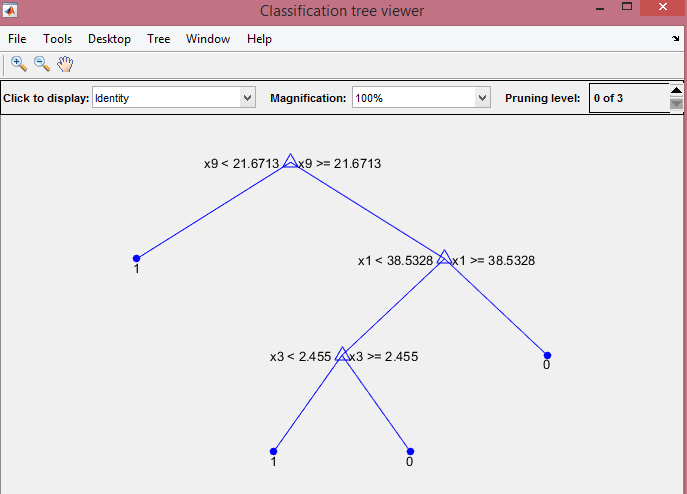
decision tree is created using the **fitctree function as shown above**

decision tree can be seen using the following code

view(decTreeOne,'Mode','graph');

numOfNodes = decTreeOne.NumNodes;

fprintf('number of nodes %d', numOfNodes);



4) generated decision tree can be used to find the predicted class labels as follows. Create the

3000 by 10 matrix of features (say, it is called TestFeatures) from the validation or test data

partition. Exclude the class labels from this table. Then use the command: PredictLabels =

predict(dtr, TestFeatures); The predicted class labels can be compared to the original class

labels to determine accuracy, precision, and recall values.

**solution)**

I used the test\_data and classified the data as TestFeatures and Original labels by spliiting them as 1-10 columns and 11th column as original\_labels.

used the **confusionmat()** to compare the original labels and predicted labels and created a precision matrix C with **TP,FP,FN,TN**

TestFeatures = test\_data(:,1:10);

PredictLabels = predict(decTreeOne,TestFeatures);

original\_labels = test\_data(:,11);

OLabels = original\_labels;

PLabels = PredictLabels;

order = [1,0];

[c,order] = confusionmat(PLabels,OLabels,'order',order);

e=c(1,1);

f=c(1,2);

g=c(2,1);

h=c(2,2);

accuracy\_1200 = (e+h)/(e+f+g+h);

precision = e/(e+f);

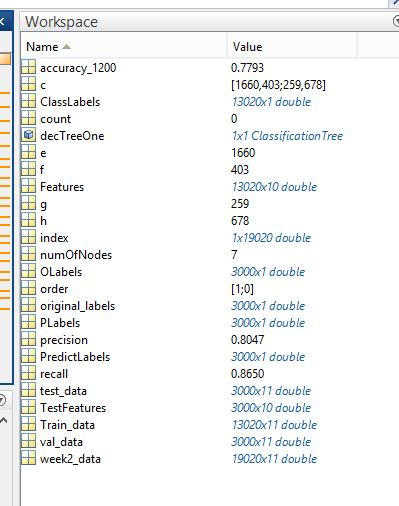
recall = e/(e+g);

**output screen**

**accuracy : 0.7793**

**precision :0.8047**

**recall: 0.8650**

****

5) Generate decision tree from training data such that no leaf node has fewer than 1000 records.

a. For this tree submit the graphical view of the decision tree.

b. Test the tree against the training data itself and find the predicted labels. Compare the

actual and the predicted labels and determine the accuracy, precision, and recall values.

Submit these results along with the table containing the numbers of TP, FN, FP, and TN

counts.

c. Repeat the past (b) above for the 3000 records of the validation data partition.

**solution:**

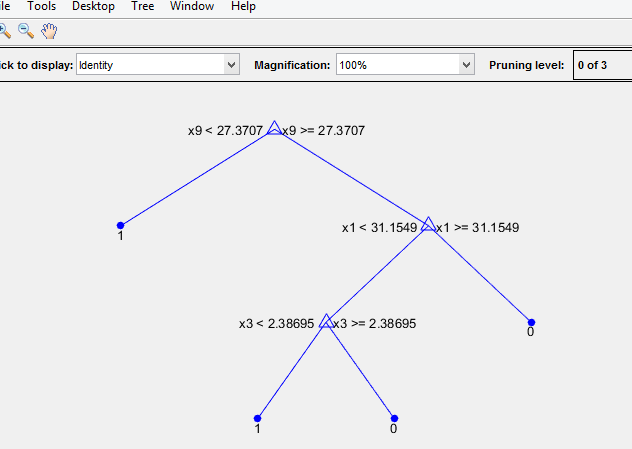
**a)**

displayed the decision tree using the following code

dtr\_Training\_1000= fitctree(Features,ClassLabels,'MinLeafSize',1000);

Numnodes\_Training\_1000=dtr\_Training\_1000.NumNodes;

view(dtr\_Training\_1000,'Mode','graph');

****

**B)** Tested the decision tree against the training data and calculated the

accuracy precision and recall as follows.

PredictLabels\_Training\_1000=predict(dtr\_Training\_1000,Features);

order = [1,0];

[td1000,order] = confusionmat(PredictLabels\_Training\_1000,ClassLabels,'order',order);

p1=td1000(1,1);

q1=td1000(1,2);

r1=td1000(2,1);

s1=td1000(2,2);

accuracy\_Training\_1000 = (p1+s1)/(p1+q1+r1+s1);

precision\_Training\_1000 = p1/(p1+q1);

recall\_Training\_1000 = p1/(p1+r1);

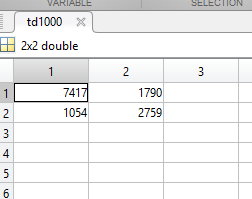
**output screen**

TP =7417

FP =1790

FN=519

TN=2590



**we can see the accuracy precision and recall outputs from workspace.**

**accuracy\_Training\_1000 =0.7816 **

**precision\_Training\_1000 =0.8056 **

**recall\_Training\_1000 =0.8576 **

**solution C)** used the validation data set and tested on the decision tree and calculated the accuracy , precision and recall as follows

**code :**

validation\_features = val\_data(:,1:10);

validation\_ClassLabels = val\_data(:,11);

Predicted\_validation1000 = predict(dtr\_Training\_1000,validation\_features);

order = [1,0];

[v,order] = confusionmat(Predicted\_validation1000,validation\_ClassLabels,'order',order);

pv= v(1,1);

qv= v(1,2);

rv= v(2,1);

sv= v(2,2);

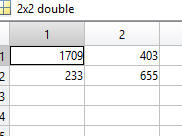
accuracy\_val\_data\_1000= (pv+sv)/(pv+qv+rv+sv);

precision\_val\_data\_1000= pv/(pv+qv);

recall\_val\_data\_1000= pv/(pv+rv);

**output screen:**

precision matrix is shown as below in the



Here TP = 1709, FP =403, FN=233, TN=655

accuracy\_val\_data\_1000 =0.7880 

precision\_val\_data\_1000=0.8092 

recall\_val\_data\_1000 = 0.8800 

**6) Repeat #5 above for the case in which no leaf node has fewer than 20 records. Submit the**

**results similar to those described in #5.**

sol)

constructed a new decision tree with no leaf node less than 20 records as follows

**code :**

train\_data\_decisiontree\_20 = fitctree(Features,ClassLabels,'MinLeafSize',20);

view(train\_data\_decisiontree\_20,'Mode','graph');

Train\_data\_Predicted\_labels\_20 = predict(train\_data\_decisiontree\_20,Features);

order = [1,0];

[td\_20,order] = confusionmat(Train\_data\_Predicted\_labels\_20,ClassLabels,'order',order);

p\_t20 = td\_20(1,1);

q\_t20 = td\_20(1,2);

r\_t20 = td\_20(2,1);

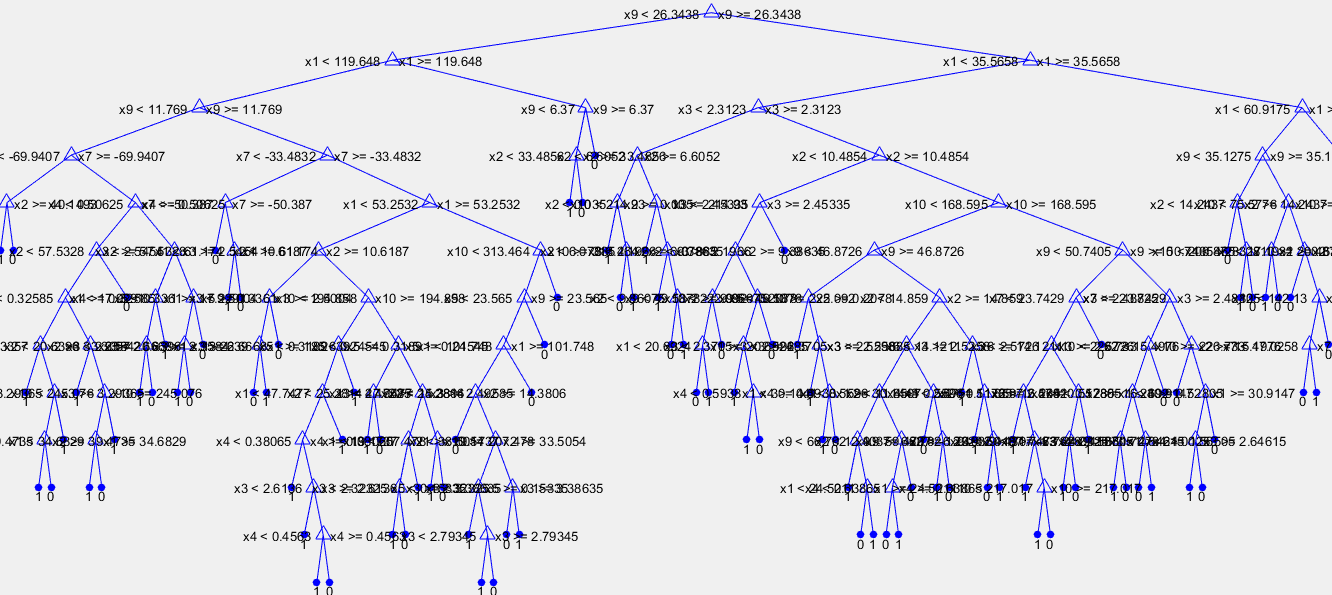
s\_t20 = td\_20(2,2);

accuracy\_train\_20= (p\_t20+s\_t20)/(p\_t20+q\_t20+r\_t20+s\_t20);

precision\_train\_20 =p\_t20/(p\_t20+q\_t20);

recall\_train\_20 = p\_t20/(p\_t20+r\_t20);

**output screen :**

****

**output :**

**accuracy using training : 0.8822 **

**precision\_train\_20: 0.8910** 

**recall\_train\_20 :0.9332 **

**tested the same decision tree with validation data set**

% %%%% 6 th solution part b with validationdata

% % view(train\_data\_decisiontree\_20,'Mode','graph');

val\_dat\_features = val\_data(:,1:10);

val\_dat\_ClassLabels = val\_data(:,11);

Predicted\_validation\_labels\_20 = predict(train\_data\_decisiontree\_20,val\_dat\_features);

order = [1,0];

[vd\_20,order] = confusionmat(Predicted\_validation\_labels\_20,val\_dat\_ClassLabels,'order',order);

p\_val20 = vd\_20(1,1);

q\_val20 = vd\_20(1,2);

r\_val20 = vd\_20(2,1);

s\_val20 = vd\_20(2,2);

accuracy\_val\_20= (p\_val20+s\_val20)/(p\_val20+q\_val20+r\_val20+s\_val20);

precision\_val\_20 =p\_val20/(p\_val20+q\_val20);

recall\_val\_20 = p\_val20/(p\_val20+r\_val20);

**output :**

accuracy\_val\_20 : 0.8517

precision\_val\_20 : 0.8659

recall\_val\_20 : 0.9134

these output can be seen in the workspace variables after executing the code.

**7. Find the accuracy of the training data and the validation data for the following values of N**

**(minimum number of records at leaf nodes): 1000, 750, 500, 250, 125, 100, 50, 20, 10, 5. Plot**

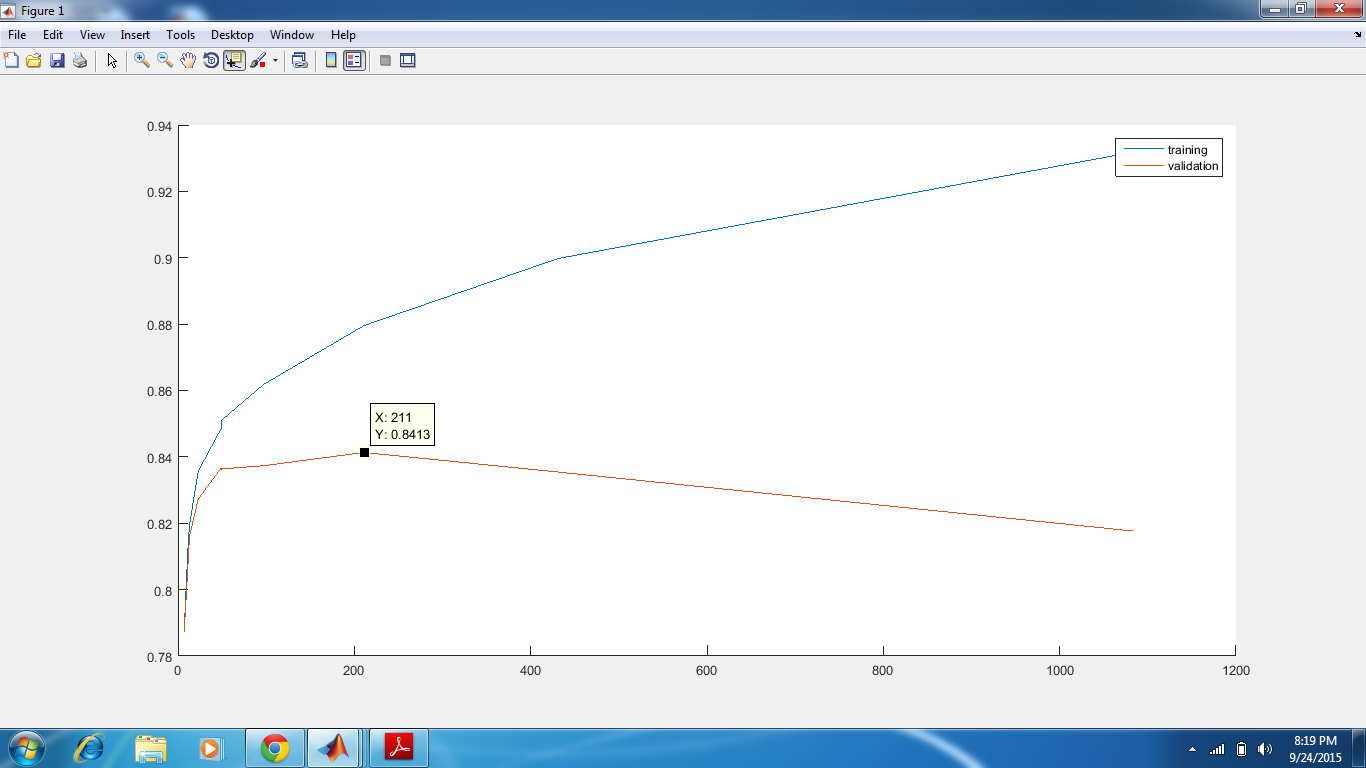
**these accuracy values on a single plot. Also plot the number of nodes in the decision tree for**

**each of the above values of N. Submit both these plots along with your interpretation of these results.**

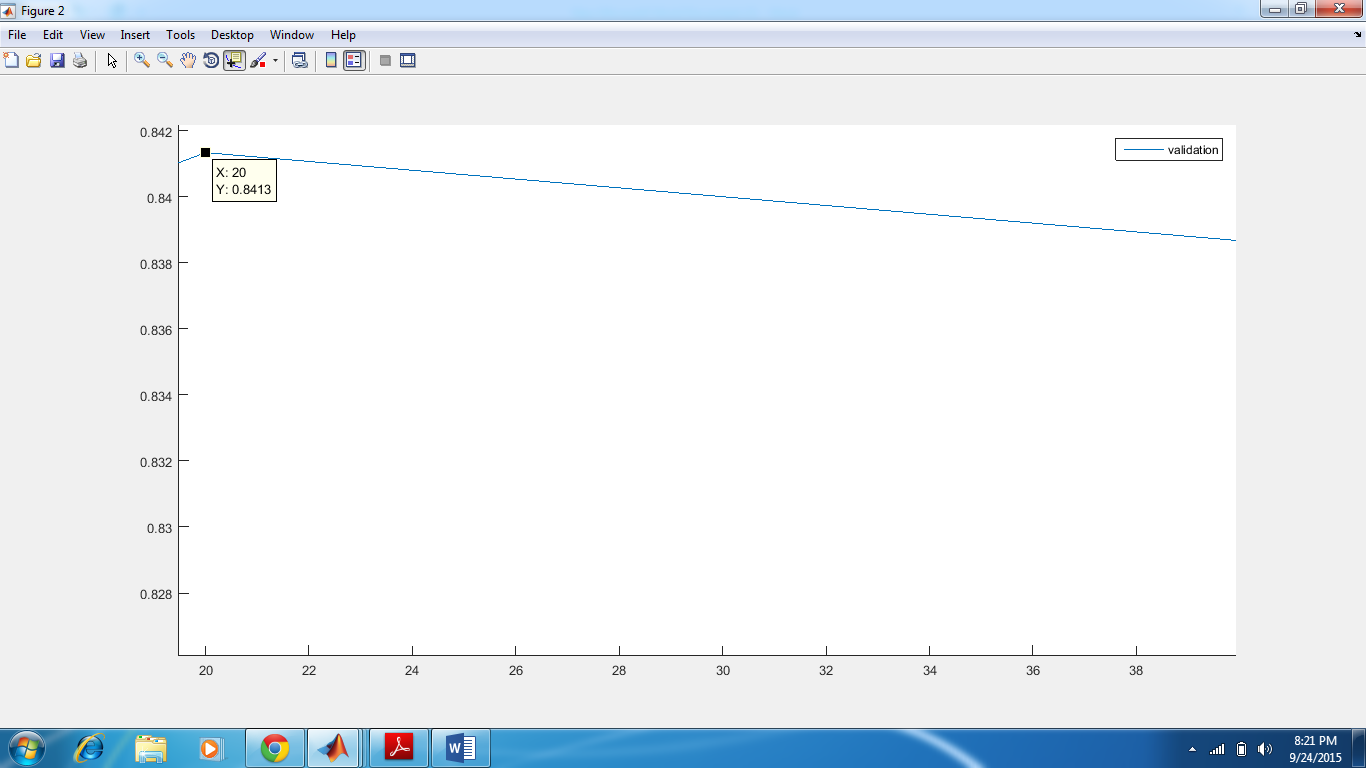
**solution)** calculated the decision trees for different nodes **1000, 750, 500, 250, 125, 100, 50, 20, 10, 5**. and calculated the accuracy values for **Training data** and  **validation data**  for each decision tree .

saved the accuracy values for training data , validation data and number of nodes in different arrays and plotted the graphs as follows.

**output screen**:



X-axis – Number of nodes, Y-axis – Accuracy of training and validation, X=211 is number of nodes, Y=0.8513 is the accuracy



X-axis – Number of records per leaf node, Y-axis – Accuracy of validation, X=20 is number of records per leaf node, Y=0.8513 is the accuracy.

**code :**

Training\_750= fitctree(Features,ClassLabels,'MinLeafSize',750);

Numnodes\_Training\_750=Training\_750.NumNodes;

%%%view(dtr\_Training\_750,'Mode','graph');

PredictLabels\_Training750=predict(Training\_750,Features);

order = [1,0];

[t750,order] = confusionmat(PredictLabels\_Training750,ClassLabels,'order',order);

p5=t750(1,1);

q5=t750(1,2);

r5=t750(2,1);

s5=t750(2,2);

accuracy\_Training750 = (p5+s5)/(p5+q5+r5+s5);

%%% validation 750 records %%%%

PredictLabels\_validation750 =predict(Training\_750,val\_dat\_features);

order =[1,0];

[v750,order] = confusionmat(PredictLabels\_validation750,val\_dat\_ClassLabels,'order',order);

pv5=t750(1,1);

qv5=t750(1,2);

rv5=t750(2,1);

sv5=t750(2,2);

accuracy\_validation\_750 = (pv5+sv5)/(pv5+qv5+rv5+sv5);

%%% training 500 reocordss %%

Training\_500= fitctree(Features,ClassLabels,'MinLeafSize',500);

Numnodes\_Training\_500=Training\_500.NumNodes;

%%%view(dtr\_Training\_750,'Mode','graph');

PredictLabels\_Training500=predict(Training\_500,Features);

order = [1,0];

[t500,order] = confusionmat(PredictLabels\_Training500,ClassLabels,'order',order);

p6=t500(1,1);

q6=t500(1,2);

r6=t500(2,1);

s6=t500(2,2);

accuracy\_Training500 = (p6+s6)/(p6+q6+r6+s6);

%%%% validation 500 records %%%%

PredictLabels\_validation500 =predict(Training\_500,val\_dat\_features);

order =[1,0];

[v500,order] = confusionmat(PredictLabels\_validation500,val\_dat\_ClassLabels,'order',order);

pv6=v500(1,1);

qv6=v500(1,2);

rv6=v500(2,1);

sv6=v500(2,2);

accuracy\_validation\_500 = (pv6+sv6)/(pv6+qv6+rv6+sv6);

%%% training 250 reocordss %%

Training\_250= fitctree(Features,ClassLabels,'MinLeafSize',250);

Numnodes\_Training\_250=Training\_250.NumNodes;

%%%view(Training\_250,'Mode','graph');

PredictLabels\_Training250=predict(Training\_250,Features);

order = [1,0];

[t250,order] = confusionmat(PredictLabels\_Training250,ClassLabels,'order',order);

p7=t250(1,1);

q7=t250(1,2);

r7=t250(2,1);

s7=t250(2,2);

accuracy\_Training250 = (p7+s7)/(p7+q7+r7+s7);

%%%% validation 250 records %%%%

PredictLabels\_validation250 =predict(Training\_250,val\_dat\_features);

order =[1,0];

[v250,order] = confusionmat(PredictLabels\_validation250,val\_dat\_ClassLabels,'order',order);

pv7=v250(1,1);

qv7=v250(1,2);

rv7=v250(2,1);

sv7=v250(2,2);

accuracy\_validation\_250 = (pv7+sv7)/(pv7+qv7+rv7+sv7);

%%%%training 125 records %%%%

Training\_125= fitctree(Features,ClassLabels,'MinLeafSize',125);

Numnodes\_Training\_125=Training\_125.NumNodes;

%%%view(Training\_250,'Mode','graph');

PredictLabels\_Training125=predict(Training\_125,Features);

order = [1,0];

[t125,order] = confusionmat(PredictLabels\_Training125,ClassLabels,'order',order);

p8=t125(1,1);

q8=t125(1,2);

r8=t125(2,1);

s8=t125(2,2);

accuracy\_Training125 = (p8+s8)/(p8+q8+r8+s8);

%%%% validation 125 records %%%%

PredictLabels\_validation125 =predict(Training\_125,val\_dat\_features);

order =[1,0];

[v125,order] = confusionmat(PredictLabels\_validation125,val\_dat\_ClassLabels,'order',order);

pv8=v125(1,1);

qv8=v125(1,2);

rv8=v125(2,1);

sv8=v125(2,2);

accuracy\_validation\_125 = (pv8+sv8)/(pv8+qv8+rv8+sv8);

%%%%training 100 records %%%%

Training\_100= fitctree(Features,ClassLabels,'MinLeafSize',100);

Numnodes\_Training\_100=Training\_100.NumNodes;

%%%view(Training\_100,'Mode','graph');

PredictLabels\_Training100=predict(Training\_100,Features);

order = [1,0];

[t100,order] = confusionmat(PredictLabels\_Training100,ClassLabels,'order',order);

p9=t100(1,1);

q9=t100(1,2);

r9=t100(2,1);

s9=t100(2,2);

accuracy\_Training100 = (p9+s9)/(p9+q9+r9+s9);

%%%% validation 100 records %%%%

PredictLabels\_validation100 =predict(Training\_100,val\_dat\_features);

order =[1,0];

[v100,order] = confusionmat(PredictLabels\_validation500,val\_dat\_ClassLabels,'order',order);

pv9=v100(1,1);

qv9=v100(1,2);

rv9=v100(2,1);

sv9=v100(2,2);

accuracy\_validation\_100= (pv9+sv9)/(pv9+qv9+rv9+sv9);

%%%%training 50 records %%%%

Training\_50= fitctree(Features,ClassLabels,'MinLeafSize',50);

Numnodes\_Training\_50=Training\_50.NumNodes;

%%%view(Training\_50,'Mode','graph');

PredictLabels\_Training50=predict(Training\_50,Features);

order = [1,0];

[t50,order] = confusionmat(PredictLabels\_Training50,ClassLabels,'order',order);

p10=t50(1,1);

q10=t50(1,2);

r10=t50(2,1);

s10=t50(2,2);

accuracy\_Training50 = (p10+s10)/(p10+q10+r10+s10);

%%%% validation 50 records %%%%

PredictLabels\_validation50 =predict(Training\_50,val\_dat\_features);

order =[1,0];

[v50,order] = confusionmat(PredictLabels\_validation50,val\_dat\_ClassLabels,'order',order);

pv10=v50(1,1);

qv10=v50(1,2);

rv10=v50(2,1);

sv10=v50(2,2);

accuracy\_validation\_50= (pv10+sv10)/(pv10+qv10+rv10+sv10);

%%%%training 20 records %%%%

Training\_20= fitctree(Features,ClassLabels,'MinLeafSize',20);

Numnodes\_Training\_20=Training\_20.NumNodes;

%%%view(Training\_50,'Mode','graph');

PredictLabels\_Training20=predict(Training\_20,Features);

order = [1,0];

[t20,order] = confusionmat(PredictLabels\_Training20,ClassLabels,'order',order);

p11=t20(1,1);

q11=t20(1,2);

r11=t20(2,1);

s11=t20(2,2);

accuracy\_Training20 = (p11+s11)/(p11+q11+r11+s11);

%%%% validation 20 records %%%%

PredictLabels\_validation20 =predict(Training\_20,val\_dat\_features);

order =[1,0];

[v20,order] = confusionmat(PredictLabels\_validation20,val\_dat\_ClassLabels,'order',order);

pv11=v20(1,1);

qv11=v20(1,2);

rv11=v20(2,1);

sv11=v20(2,2);

accuracy\_validation\_20= (pv11+sv11)/(pv11+qv11+rv11+sv11);

%%%%training 10 records %%%%

Training\_10= fitctree(Features,ClassLabels,'MinLeafSize',10);

Numnodes\_Training\_10=Training\_10.NumNodes;

%%%view(Training\_50,'Mode','graph');

PredictLabels\_Training20=predict(Training\_10,Features);

order = [1,0];

[t10,order] = confusionmat(PredictLabels\_Training50,ClassLabels,'order',order);

p12=t10(1,1);

q12=t10(1,2);

r12=t10(2,1);

s12=t10(2,2);

accuracy\_Training10 = (p12+s12)/(p12+q12+r12+s12);

%%%% validation 10 records %%%%

PredictLabels\_validation10 =predict(Training\_10,val\_dat\_features);

order =[1,0];

[v10,order] = confusionmat(PredictLabels\_validation20,val\_dat\_ClassLabels,'order',order);

pv12=v10(1,1);

qv12=v10(1,2);

rv12=v10(2,1);

sv12=v10(2,2);

accuracy\_validation\_10= (pv12+sv12)/(pv12+qv12+rv12+sv12);

%%%%training 5 records %%%%

Training\_5= fitctree(Features,ClassLabels,'MinLeafSize',5);

Numnodes\_Training\_5=Training\_5.NumNodes;

%%%view(Training\_50,'Mode','graph');

PredictLabels\_Training5=predict(Training\_5,Features);

order = [1,0];

[t5,order] = confusionmat(PredictLabels\_Training5,ClassLabels,'order',order);

p13=t5(1,1);

q13=t5(1,2);

r13=t5(2,1);

s13=t5(2,2);

accuracy\_Training5 = (p13+s13)/(p13+q13+r13+s13);

%%%% validation 5 records %%%%

PredictLabels\_validation5 =predict(Training\_5,val\_dat\_features);

order =[1,0];

[v5,order] = confusionmat(PredictLabels\_validation5,val\_dat\_ClassLabels,'order',order);

pv13=v5(1,1);

qv13=v5(1,2);

rv13=v5(2,1);

sv13=v5(2,2);

accuracy\_validation\_5= (pv13+sv13)/(pv13+qv13+rv13+sv13);

accuracy\_Training\_array =[accuracy\_Training5,accuracy\_Training10,accuracy\_Training20,accuracy\_Training50,accuracy\_Training100,accuracy\_Training125,accuracy\_Training250,accuracy\_Training500,accuracy\_Training750,accuracy\_Training\_1000];

accuracy\_Validation\_array =[accuracy\_validation\_5,accuracy\_validation\_10,accuracy\_validation\_20,accuracy\_validation\_50,accuracy\_validation\_100,accuracy\_validation\_125,accuracy\_validation\_250,accuracy\_validation\_500,accuracy\_validation\_750,accuracy\_Training\_1000];

NumberOfNodes\_array=[Numnodes\_Training\_5,Numnodes\_Training\_10,Numnodes\_Training\_20,Numnodes\_Training\_50,Numnodes\_Training\_100,Numnodes\_Training\_125,Numnodes\_Training\_250,Numnodes\_Training\_500,Numnodes\_Training\_750,Numnodes\_Training\_1000];

plot(NumberOfNodes\_array,accuracy\_Training\_array,NumberOfNodes\_array,accuracy\_Validation\_array);

plot(NumberOfNodes\_array,accuracy\_Training\_array)

plot(NumberOfNodes\_array,accuracy\_Validation\_array)

**8) As per the results obtained in #7 above which decision tree is the best model for the training**

**data? How many nodes does it have? Find the accuracy, precision, and recall when this decision**

**tree model is tested against the 3000 records in the test partition of the dataset. Submit all**

**these answers/results.**

**sol)** The decision tree with **211** **nodes** would be the best model for the training data with minimum number of 20 records at leaf nodes.

Accuracy, Precision and Recall values when this decision tree model is tested against the 3000 records present in test partition data set.

**code :**

%%% 8th solution %%%

Records\_Per\_Leaf\_nodes = [5,10,20,50,100,125,250,500,750,1000];

figure()

hold all

%Finding the number of records

plot(Records\_Per\_Leaf\_nodes,Total\_accuracy\_validation)

legend('Leaf Nodes','Accuracy');

%Creatinng decision tree for leaf node no less than 20 records

training\_data\_dtr\_20\_final\_ques = fitctree(Features,ClassLabels,'MinLeafSize',20);

view(training\_data\_dtr\_20\_final\_ques,'Mode','graph');

noofnodes = training\_data\_dtr\_20\_final\_ques.NumNodes;

Predicted\_labels\_20\_final\_ques = predict(training\_data\_dtr\_20\_final\_ques,TestFeatures);

order = [1,0];

[mat\_ques,order] = confusionmat(Predicted\_labels\_20\_final\_ques,original\_labels,'order',order);

pf=mat\_ques(1,1);

qf=mat\_ques(1,2);

rf=mat\_ques(2,1);

sf=mat\_ques(2,2);

accuracy\_final\_ques = (pf+sf)/(pf+qf+rf+sf);

precision\_final\_ques = pf/(pf+qf);

recall\_final\_ques = pf/(pf+rf);

fprintf('Accuracy for best decision tree : %f\n Precision for best decision tree : %f \n Recall for best decision tree is: %f ',accuracy\_final\_ques,precision\_final\_ques,recall\_final\_ques);

**output :**

**Accuray for best decision tree : 0.83922**

**Precision for best decision tree : 0.8565**

**Recall for best decision tree : 0.8757**