Assignment – 2

Vivek Roshan Suggala | M10725902

1. (10) Load the dataset in MATLAB as an array named d1 and compute the covariance matrix of the data by using the command: “cm = cov(d1);”. Write all the information that you can infer by interpreting the values in the covariance matrix cm.

   Ans:

   The code for the covariance-matrix is:

```
d1 = xlsread('C:\Users\suggalvn\Downloads\data_banknote_authentication');
cm = cov(d1);
```

```
cm =

    8.0813    4.4051   -4.6663    1.6533   -1.0243
    4.4051   34.4457  -19.9051   -6.4900   -1.2974
   -4.6663  -19.9051   18.5764    2.8872    0.3340
    1.6533   -6.4900    2.8872    4.4143   -0.0245
   -1.0243   -1.2974    0.3340   -0.0245    0.2471
```

- The covariance matrix generalizes the notion of variance to multiple dimensions
- Covariance is unscaled correlation. It values are Covariance matrix of dataset gives the relation between 2 atrributes. When the value is far to 0, it indicates how one attribute affects other one.
- The magnitude of the value gives us the relationship between the attributes.
- Furthermore, If the value is negative it is inversely proportional and if the value is positive, they are directly proportional.
- For eg: covariance of 2 and 3 is -19.9051. so when attribute 2 increases, attribute 3 decreases. They are inversely related.
- Similarly, covariance of attribute 3 and attribute 5 is 0.3340. which is near to 0. This indicates that attribute 3 and attribute 5 are not strongly related. Changes in attribute 3 does not matter much with attribute 5.
- Values in diagonal are just giving relation to same attribute.
- They are just variances of values of same attribute. This matrix is also called variance-covariance matrix.

2. Do a scatter plot of attribute-1 and attribute-2. Use different colors to mark the points from the two different classes. Write your interpretation of the separability of the two classes using attribute1 and attribute2, and also any other insights that you can obtain from this scatter plot.

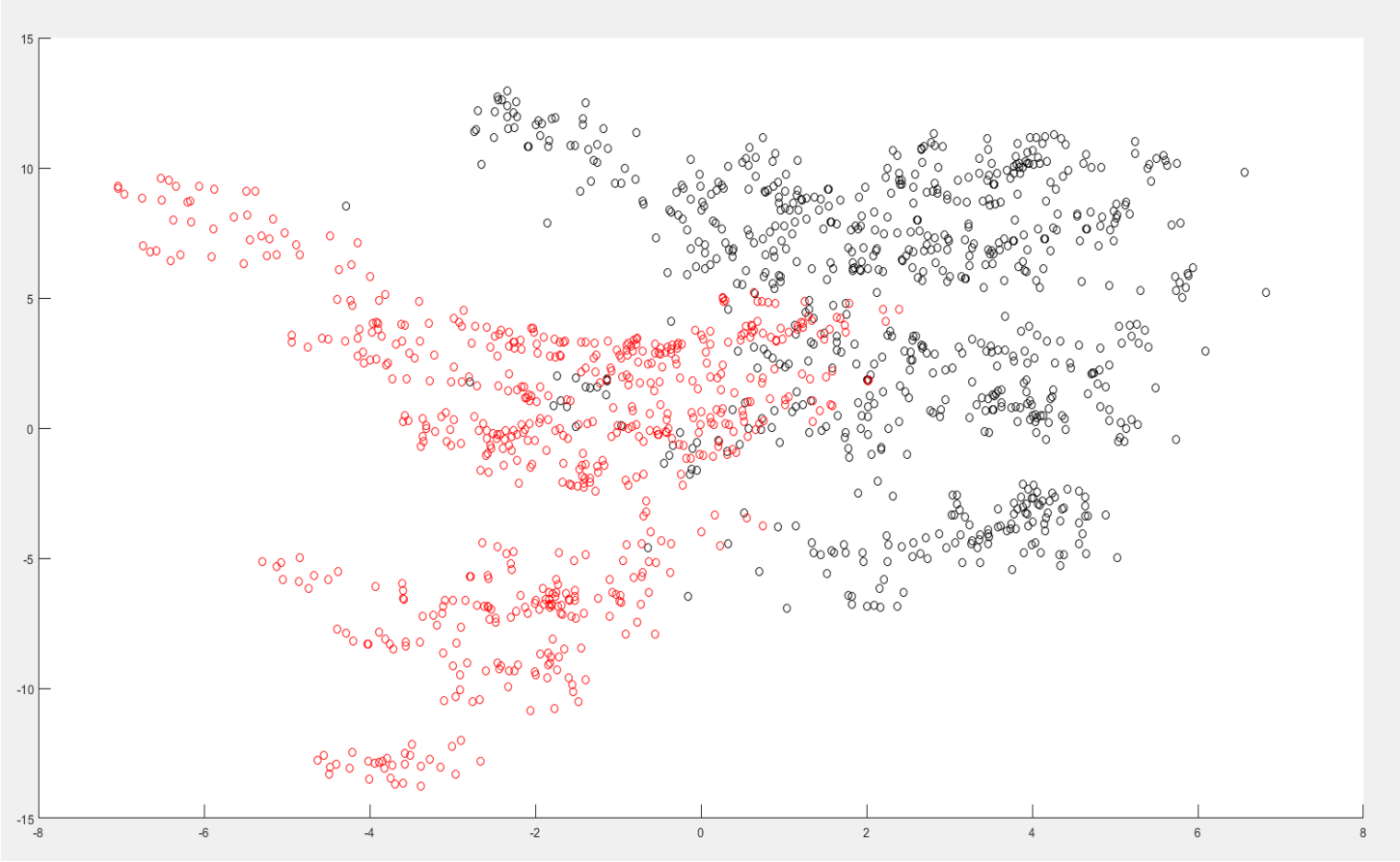
   Ans:

   Steps:

   - Obtain a scatter plot for the attributes1 and attributes2.
   - Use different colors for the class0 and class1.
   - In our scatter plot, class red is represented as color 1 and color black is represented to class 0
   - The axes in the graph are attribute1 and attribute2 of the data set.

Code for the scatter plot is :

```
D10 = d1(d1(:,5)==0,:);
D11 = d1(d1(:,5)==1,:);
scatter(D10(:,1),D10(:,2),'k');
hold
scatter(D11(:,1),D11(:,2),'r');
```

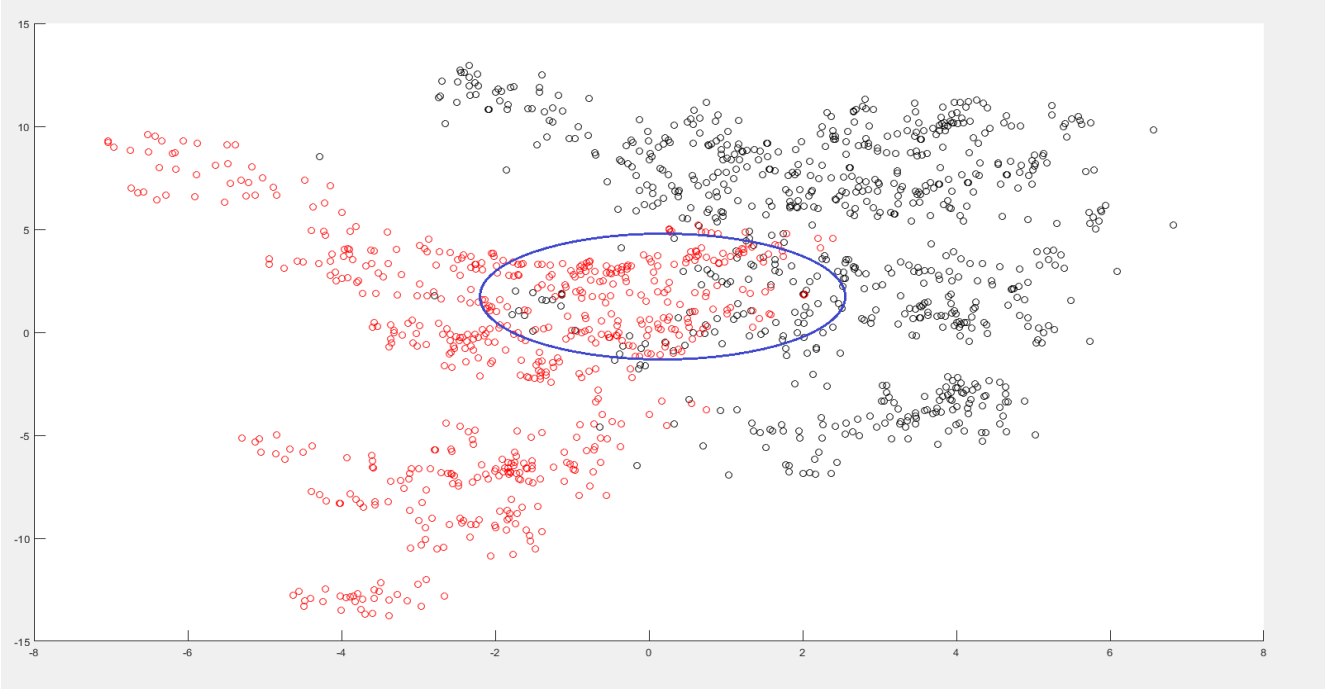


- The above scatter plot is plotted against attribute 1 and attribute2.
- Records with class label 0 is marked as red and those with class label 1 is marked as black.
- Graphs gives us a general idea that classes 0 and 1 are not linearly seperable when classified using attribute 1 and attribute 2.
- When a classifier is used to separate them it leaves noise in other side of classifier around -2 to 2 values of attribute 1. That region is confused area of the plot.

3. On the scatter plot obtained in Q#2 above, draw the axis-parallel boundaries, using your intuition, to suggest the boundaries that an efficient decision tree may draw. Show the resulting partitions in the form of a decision tree.

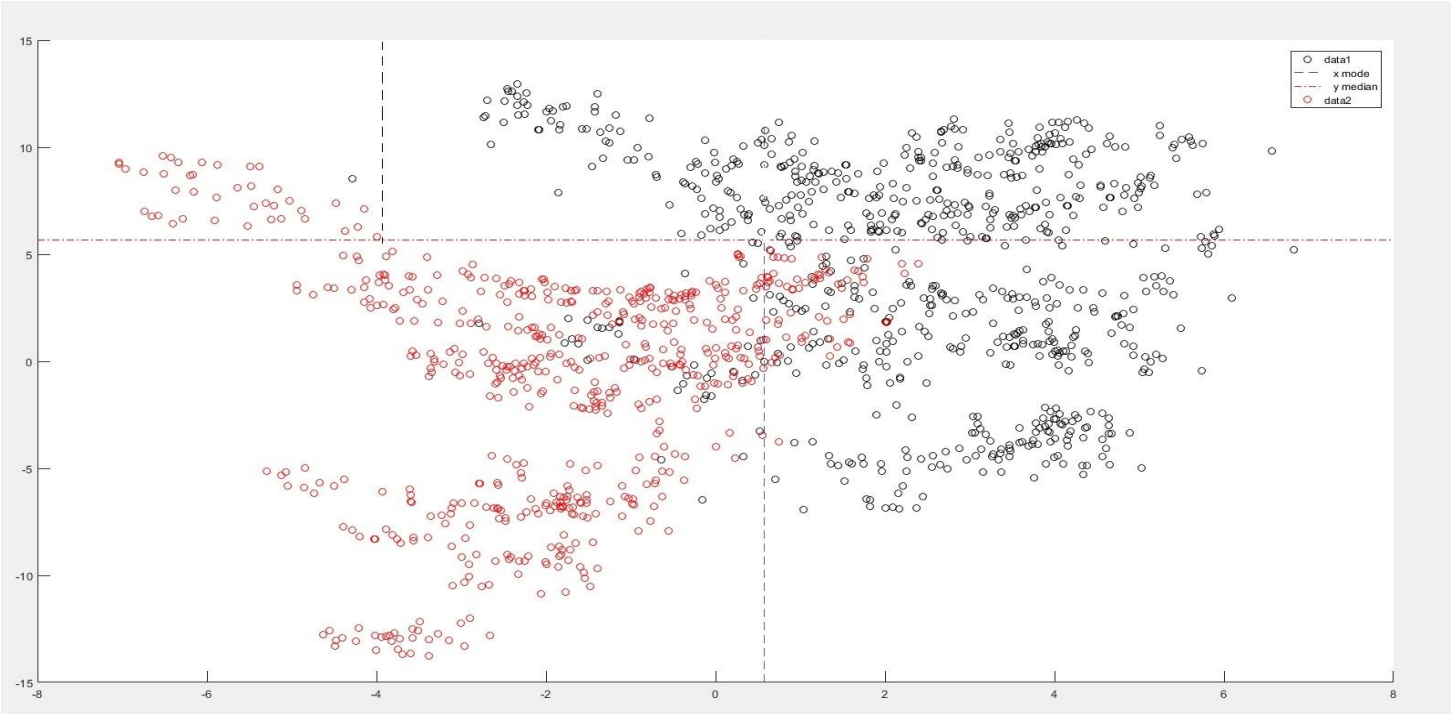
Ans:

Scatter plot with the erroneous(Noise) data points are:

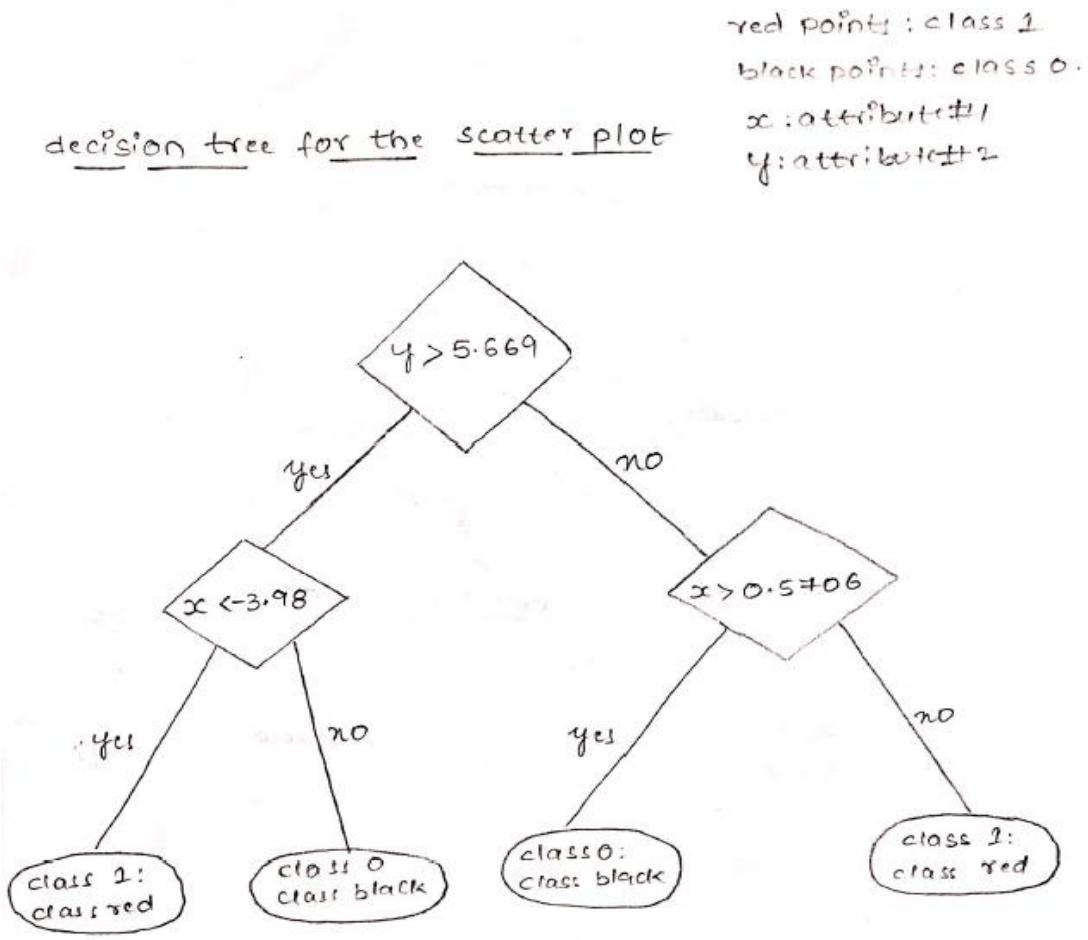


- The decision boundaries are drawn in such a way that the boundaries classify the classes properly.
- The data points which cannot be classified are termed to be as noise or erroneous data.
- The erroneous points are represented in the figure above.
- The decision boundaries in the figure below, are drawn parallel to the axis, classifying the two classes.

The scatter plot for the decision boundaries parallel to axis is:



Decision tree that can be generated for the above scatted plot with decision boundaries is:



4. From the dataset randomly select 800 records records for testing, 200 for validation, and the rest for testing. These selections MUST be random. Use a random number generator function to generate random numbers between 1 and 1372, and then choose records for training, validation, and testing. Show the code used for selecting the three different sets of records. Write the record numbers of rows that are chosen for your test set.

Ans:

Steps Followed:

- The data set is to be divided randomly in such a way that, there are 800 records for training, 200 for testing and 372 for test data.
- 'dividerand' is a function which can be used to split the data according to the requirement.

Matlab Code for selecting the appropriate records for the training, testing and validation records is:

```
TotalSize = size(d1,1);
Training = 800/TotalSize;
Validation = 200/TotalSize;
Testing = 1-(Training+Validation)
[TrainingIndexes, ValidationIndexes, TestingIndexes] = dividerand(TotalSize, Training, Validation,
Testing);
TrainingSet = d1(TrainingIndexes, :);
ValidationSet = d1(ValidationIndexes, :);
TestingSet = d1(TestingIndexes, :);
TestingIndexList = TestingIndexes';
allInOneString = sprintf('%.0f,' , TestingIndexList);
allInOneString = allInOneString(1:end-1);
```

Output:

```
allInOneString =

1,2,4,9,10,12,13,15,18,19,20,23,24,26,27,36,39,43,44,50,57,61,65,66,71,73,75,82,85,87,88,94,95,102,1
10,114,116,118,122,124,125,126,130,132,137,141,153,154,158,161,167,168,173,178,183,189,191,192,199,2
00,203,206,207,215,222,225,226,243,245,246,251,260,262,263,264,270,274,276,277,282,285,288,289,293,2
97,307,308,309,313,315,318,323,324,334,341,342,344,348,350,359,363,368,370,385,386,387,391,392,393,4
14,416,421,430,432,434,444,446,451,456,458,462,468,470,472,484,493,497,498,503,507,516,524,526,527,5
28,535,539,546,551,561,563,565,567,568,571,572,576,577,579,580,582,587,588,589,601,603,606,607,608,6
10,612,615,618,620,621,628,629,630,631,633,637,639,663,664,671,679,680,681,683,684,685,686,687,693,7
00,702,709,711,714,718,719,728,731,733,734,741,747,753,760,764,765,768,771,777,779,785,788,793,794,7
95,799,800,803,809,811,813,814,815,821,825,833,842,845,848,852,855,857,867,875,889,895,901,908,914,9
15,920,928,936,937,940,941,942,943,949,953,957,961,962,972,974,976,977,981,984,990,992,993,994,997,1
000,1003,1012,1014,1015,1018,1020,1027,1033,1034,1035,1036,1045,1046,1050,1051,1053,1056,1058,1059,1
067,1070,1071,1073,1074,1083,1090,1092,1094,1100,1101,1104,1105,1106,1110,1118,1119,1120,1126,1130,1
136,1143,1144,1146,1149,1150,1154,1156,1158,1160,1161,1164,1169,1173,1175,1176,1177,1192,1194,1196,1
200,1203,1206,1208,1215,1218,1229,1231,1235,1236,1239,1245,1246,1249,1250,1253,1255,1260,1267,1268,1
274,1275,1277,1278,1280,1281,1283,1290,1291,1292,1295,1296,1303,1304,1306,1308,1309,1310,1314,1315,1
316,1320,1323,1324,1325,1334,1335,1339,1345,1352,1363,1364,1366
```

5. (20) Use the training set to learn decision trees (use fitctree command of Matlab) by varying the parameter that controls the minimum number of records in a leaf node. For this parameter use the values of 1, 2, 5, 10, 15, 20, 25, 30, . . , 50. For each generated tree find its performance (accuracy) on the test and validation sets. Plot this accuracy value against the parameter values for both the data sets. Choose that value of parameter which in your view does not overfit the data. Justify your choice of the parameter value and the corresponding decision tree. Show the selected tree graphically using Matlab's view command. The role of the validation data set is to help you pick that decision tree which gives the best performance without overfitting the training data.

Ans:

Steps:

- The aim of the question is to find out the optimal parameter value for the decision tree.
- Generate the trees for different given parameters using the training data and training class labels.
- Predict the class labels of the training and validation sets data individually.
- Compare the predicted values with the original values, and find out the accuracy of the generated tree.
- For calculating the predicted tree accuracy, I have written a function named 'input parameters'.
- I pass the predicted class labels and the actual class labels into this function. It gives out the accuracy values and store the accuracy values in an array.

- Continue this process, generate different trees with the given predicted values, calculate the accuracy values and store them on a vector.
- Do this process and compute the accuracy values for validation and training sets differently.
- Plot graphs for the accuracy values and the leaf nodes for validation data set and training data set.

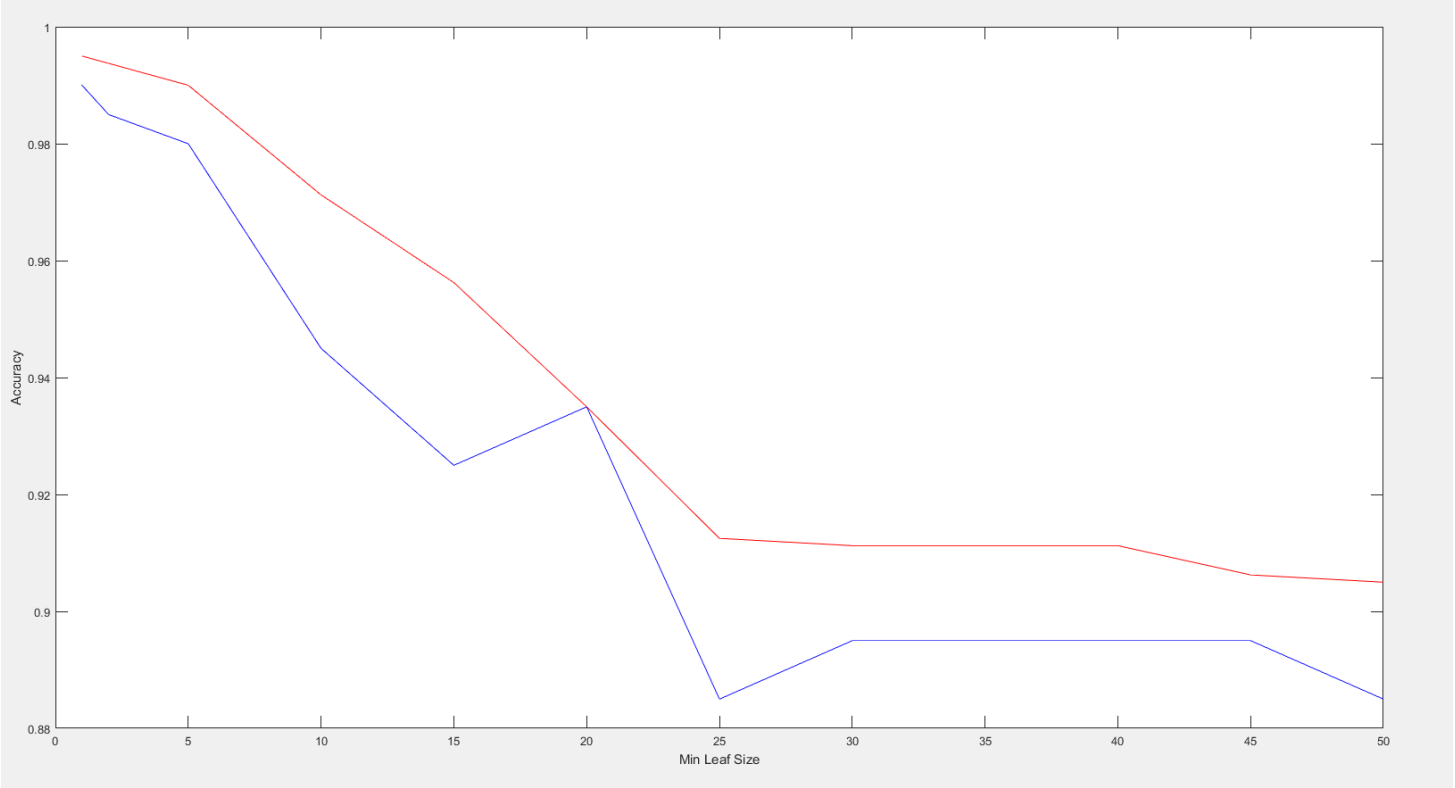
Matlab Code:

```
%problem 5
N = numel(leafs);
for n=1:N
    t = fitctree(TrainingSet(:,1:4), TrainingSet(:,5),'MinLeafSize',leafs(n));
    testlabels1 = predict(t,TrainingSet(:,1:4));
    testlabels2 = predict(t,ValidationSet(:,1:4));
    accuracy1(n) = inputParameters(testlabels1,TrainingSet(:,5));
    accuracy2(n) = inputParameters(testlabels2,ValidationSet(:,5));
end
plot(leafs,accuracy1,'color','r');
hold on;
plot(leafs,accuracy2,'color','b');
xlabel('Min Leaf Size');
ylabel('Accuracy');
```

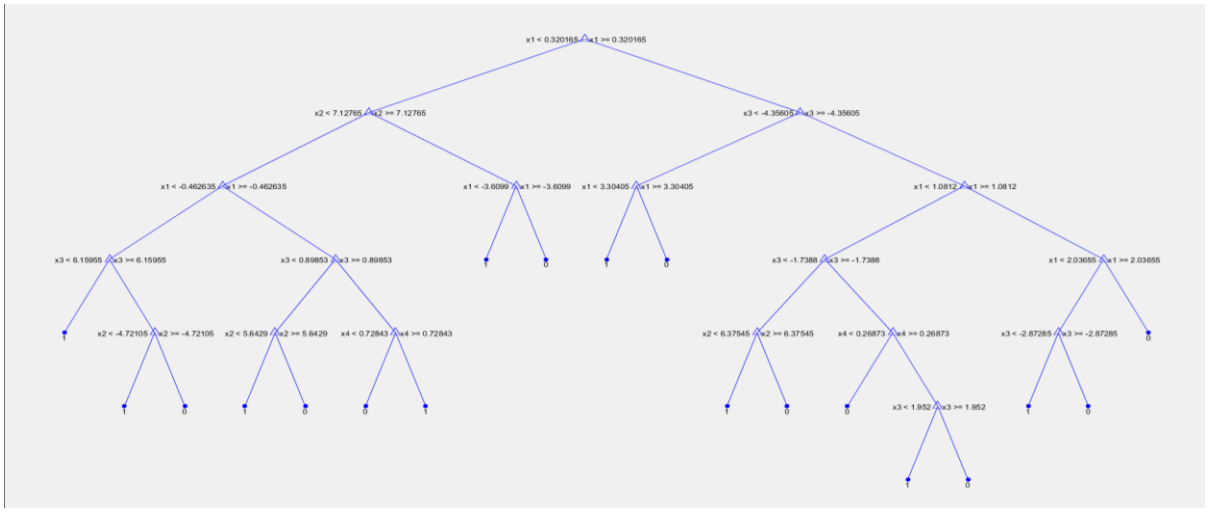
Function for calculating the accuracy is:

```
function accuracy = inputParameters(a,b)
    differences = (a==b);
    totalOnes = 0;
    for i=1:length(differences)
       if(differences(i)==1)
           totalOnes = totalOnes+1;
       end
    end
    accuracy = (totalOnes/length(differences));
end
```

- This function calculates the accuracy of the tree generated and returns the accuracy values for the trees.
- The predicted values are to be compared with the actual values and the accuracy for the tree is recorded.
- Now all the predicted values are to be calculated and recorded for all the different data sets for the training and validation



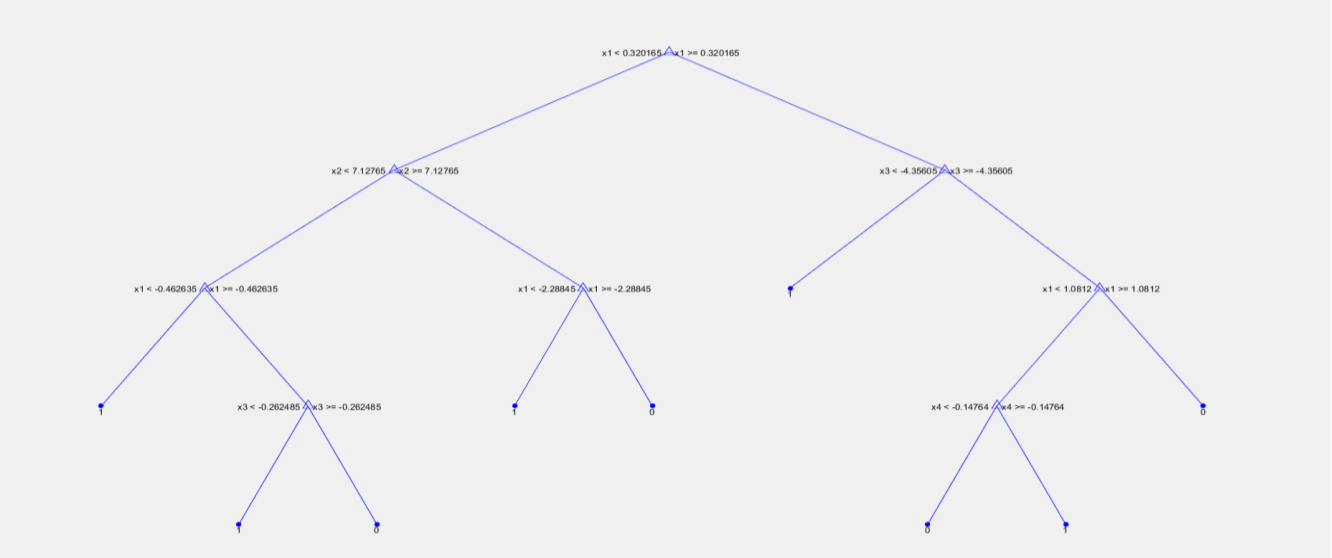
- The red plot indicates the accuracy values for the training data and the blue plot indicates the accuracy values for the validation test set.
- From the graph obtained, lets calculate the accuracy for the leafNode 15 and 20.
- For leaf node 15: training: 95.50 Validation: 92.25
- For leaf node 20: training:94 Validation: 94
- As mentioned in the question paper, the validation set is useful for selecting the decision tree.
- The validation accuracy for the leafMinNode 15 is 92.25 and for the leafMinNode 20 is **94**.
- And hence, the average of the accuracy for the leaf node 20 is higher for my generated data---------- Point **1**
- The accuracy (vs) Min leaf size graph is drawn above. From the graph we can say that, when the parameter value for the leaf Node increases, the accuracy gradually decreases (approximately).
- But, when we take the graph for the MinLeafNode as 1, the graph looks as below:



- From the above graph, it is clear that, when the MinLeafNode is less, the depth of the tree is more ---- Point 2.
- When the depth of the tree is greater, the number of questions asked will be increased, which is not efficient.
- Hence, we have to consider two parameters for selecting a decision tree.
  1. The accuracy should be good------- from the Point 1.
  2. The depth should be considerable -------- from the Point 2.
- From the two Points, we can say that, for the **Min leaf node 20**, we have a good accuracy with a considerable depth for the decision tree.

```
treeChoosen = fitctree(TrainingSet(:,1:4), TrainingSet(:,5), 'MinLeafSize', 20);
view(treeChoosen,'Mode','graph');
```

- From the above matlab code, we can create a tree of MinLeafNode 20 and view the tree using the matlab view command.
- The decision tree obtained for the MinLeafNode 20 for the trianing data is as follows:
- The tree can be viewed by the view command of the matlab.



6. (10) Test the selected decision tree using the test portion of the dataset. Report accuracy, precision, and recall for the test dataset. Report the precision and recall values for both classes separately.

Steps Followed:

- Predict the class labels of the test set by passing the test dataset into the tree chosen.
- Compute the Confusion matrix for the actual and predicted class labels.
- From the Confusion matrix obtained, calculate the accuracy, precision and recall values.
- Precision and recall values can be calculated for different classes individually.

Matlab Code:

```
testlabelsChoosen = predict(treeChoosen,TestingSet(:,1:4));
C1 = confusionmat(TestingSet(:,5),testlabelsChoosen);
aTP = C1(1,1); bFN = C1(1,2); cFP = C1(2,1); dTN = C1(2,2);
```

```
AccuracyDataSet = (aTP+dTN)/(aTP+bFN+cFP+dTN);

Precision0 = aTP/(aTP+cFP);
Recall0 = aTP/(aTP+bFN);
Precision1 = dTN/(dTN+bFN);
Recall1 = dTN/(dTN+cFP);
```

The Cost Matrix is:

		PREDICTED CLASS	
	C(i|j)	**Class=Yes**	**Class=No**
ACTUAL CLASS	**Class=Yes**	C(Yes|Yes)	C(No|Yes)
	Class=No	C(Yes|No)	C(No|No)

The Confusion matrix obtained is:
Where TP is true positive, TN is true negative, FP is false positive, FN is false negative.

```
C1 =

   177    22
    14   159
```

Accuracy =(TP+TN)/(TP+TN+FP+FN)=336/372= 0.9032.

```
>> AccuracyDataSet

AccuracyDataSet =

    0.9032
```

Accuracy percentage is used to calculate how well classifier predicts class label of test data. Here 'treeChoosen' is able to predict 177+159=336 records accurately out of 372 records of test data. Thus accuracy is 336/372.

```
>> Precision0

Precision0 =

    0.9267
```

```
>> Recall0

Recall0 =

    0.8894
```

```
>> Precision1

Precision1 =

    0.8785
```

```
>> Recall1

Recall1 =

    0.9191
```

Precision of class label 0=TP/(TP+FP) = 0.9267

Precision of class label 1=TN/(TN+FN) = 0.8785

Recall of class label 0= TP/(TP+FN) = 0.8894

Recall of class label 1= TN/(TN+FP) = 0.9191

7. Consider only the attribute-1 and attribute-2 of the dataset for predicting the class label. Train a perceptron using Matlab's Train function that finds the best discriminating surface for this 2-D data space. Report the weight vector output by this function and also report the number of data points that are misclassified. Plot all the data points as done in the scatter plot in Q#2 above. Now plot the line representing the learned perceptron on this scatter plot. Mark the points that are misclassified.

Ans:

Steps Followed:

- Take the values of attribute1 and attribute2 in a vector, 'x'.
- Take the class labels in another vector.
- Create a perceptron which is used to train the data set and outputs the resultant weighted vector.
- Set the epochs accordingly, the higher the epoch count the lesser the error value. Hence consider the epoch value to be 1000.
- Configure the perceptron accordingly. Code is written below accordingly.
- Plot a scatter plot for the attribute1 and attribute2 as done in the problem2.
- Hold the figure and plot the perceptron trained, with the obtained result and bias weights on this scatter plot.
- Classes can be represented using different colors in the scatter plot.

Matlab Code:

```
x = [d1(:,1) d1(:,2)];
t = d1(:,5);
x = x';
t = t';
net = perceptron;
net.trainParam.epochs=1000;
net = configure(net,x,t);
net = train(net,x,t);
figure;
D10 = d1(d1(:,5)==0,:);
D11 = d1(d1(:,5)==1,:);
scatter(D10(:,1),D10(:,2),'k');
hold
scatter(D11(:,1),D11(:,2),'r');
w = net.iw{1,1};
b = net.b{1};
%plotpv(z,t)
```

```
plotpc(w,b)
%for calculating the misclassified points
%y = ax+by+c
y=w(1).*d1(:,1)+w(2).*d1(:,2)+b;
y(find(y<0))=0;
y(find(y>0))=1;
misclassified=find(y~=d1(:,5));
fprintf('NUMBER OF misclassified datapoints in the entire dataset are: %d',length(misclassified));
```

Results Obtained:

The weighted vector obtained after 1000 epoch's is :

```
>> w

w =

   -8.3614    1.4171

>> b

b =

    15
```

- After getting the output vectors for the weights, we can plot the perceptron using the plotpc command in Matlab.
- Draw a scatter plot as done in the problem#2 and plot a perceptron on to the scatter plot, as done in the code

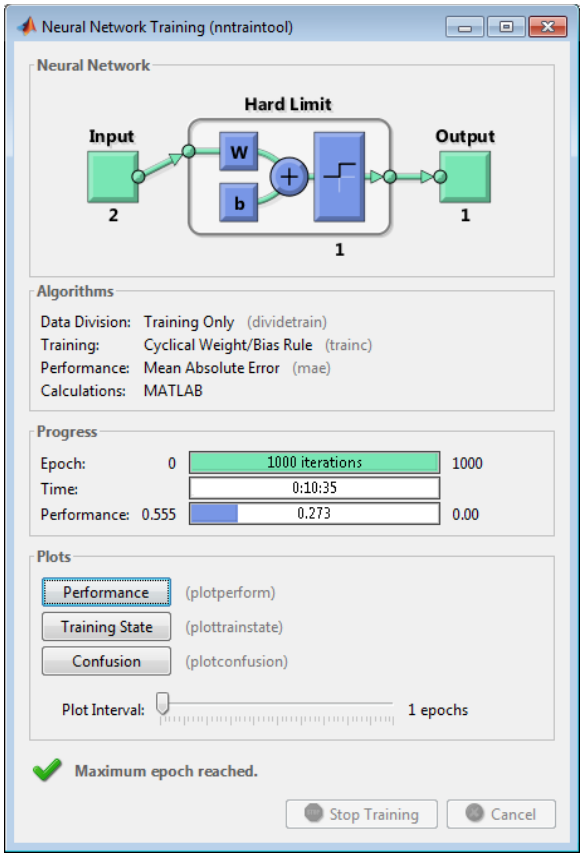
Code and the output for the misclassified data points are:

```
y(find(y<0))=0;
y(find(y>0))=1;
misclassified=find(y~=d1(:,5));
fprintf('NUMBER OF misclassified datapoints in the entire dataset are: %d',length(misclassified));
NUMBER OF misclassified datapoints in the entire dataset are: 374>>
```

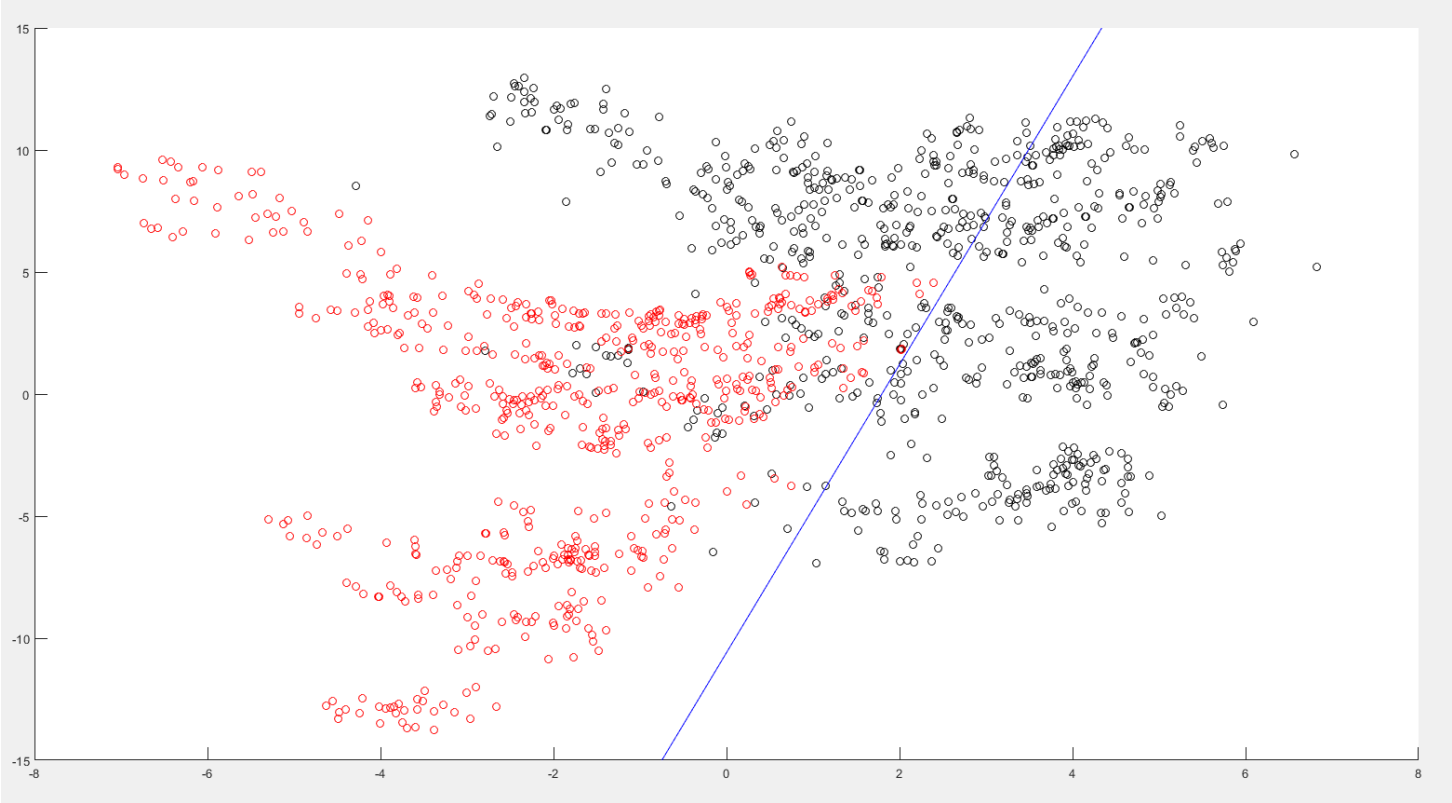
Explanation:

- The perceptron is a line with the equation, y=ax+by+c.
- a is the value obtained in the w(1), the weight vector. b is w(2) and c is b.
- x is the first attribute d1(:,1) and y(:,1) is the second attribute. When substituted in the line equation of the perceptron, if y>0, it belongs to one class and y<0 it belongs to class 0.
- Now we can find the misclassified points by the code written above.
- The number of misclassified points obtained are 374, for our dataset.

Training the perceptron:



Perceptron on the scatter plot:



Misclassified Points marked(manually) are: