

Assignment -3 IDA solutions

Sampath Dontharaju

M08879603

1 sol)

downloaded the file as csv file. Data contains some missing values like '?' and 'NAN'. to remove these missing values i have used the knnimpute() function in matlab. This replaces the missing data with corresponding value from the nearest neighbor column.

following is the logic used for 1st solution

```
csvdata = xlsread('D:\Assignments UC sem1\IDA\Assignment 3\breast-cancer-wisconsin.csv');  
week3_data = knnimpute(csvdata);
```

2 sol)

Used the **randperm()** function of matlab to generate the random numbers from 1 to 699.

```
i=randperm(699);
```

```
Trainingdata = week3_data(i(1:500),:);
```

Using the above line of code to take 500 records into Training data dataset.

```
Testdata = week3_data(i(501:end),:);
```

The above line of code takes 199 records into Testdata dataset.

3 sol)

Trainingdata dataset created from the previous solution is used to create the decision tree

```
Features = Trainingdata(:,2:10);
```

```
Class_Labels = Trainingdata(:,11);
```

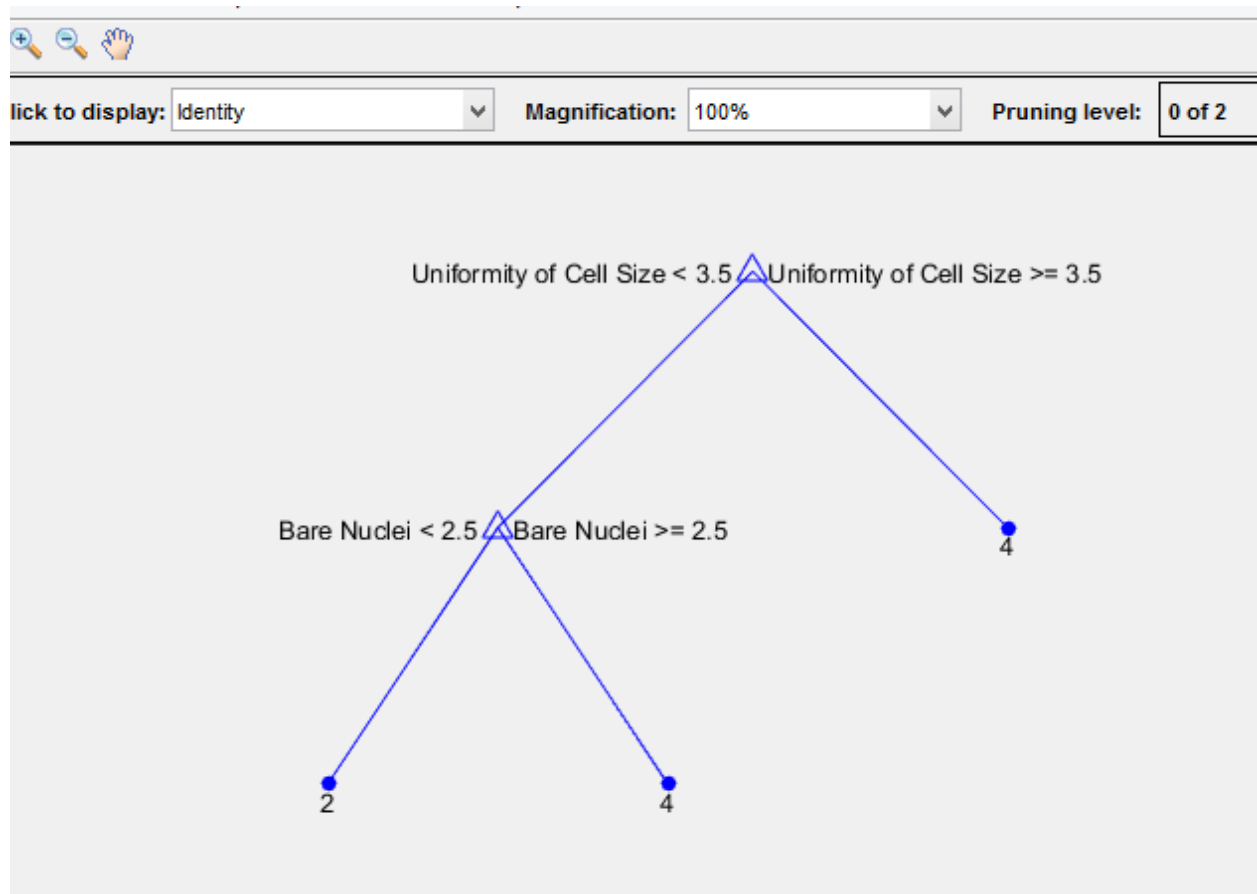
created a decision tree on the training data using the **fitctree()** function of matlab.

```
Training_decision_tree = fitctree(Features, Class_Labels, 'MinLeafSize',25);
```

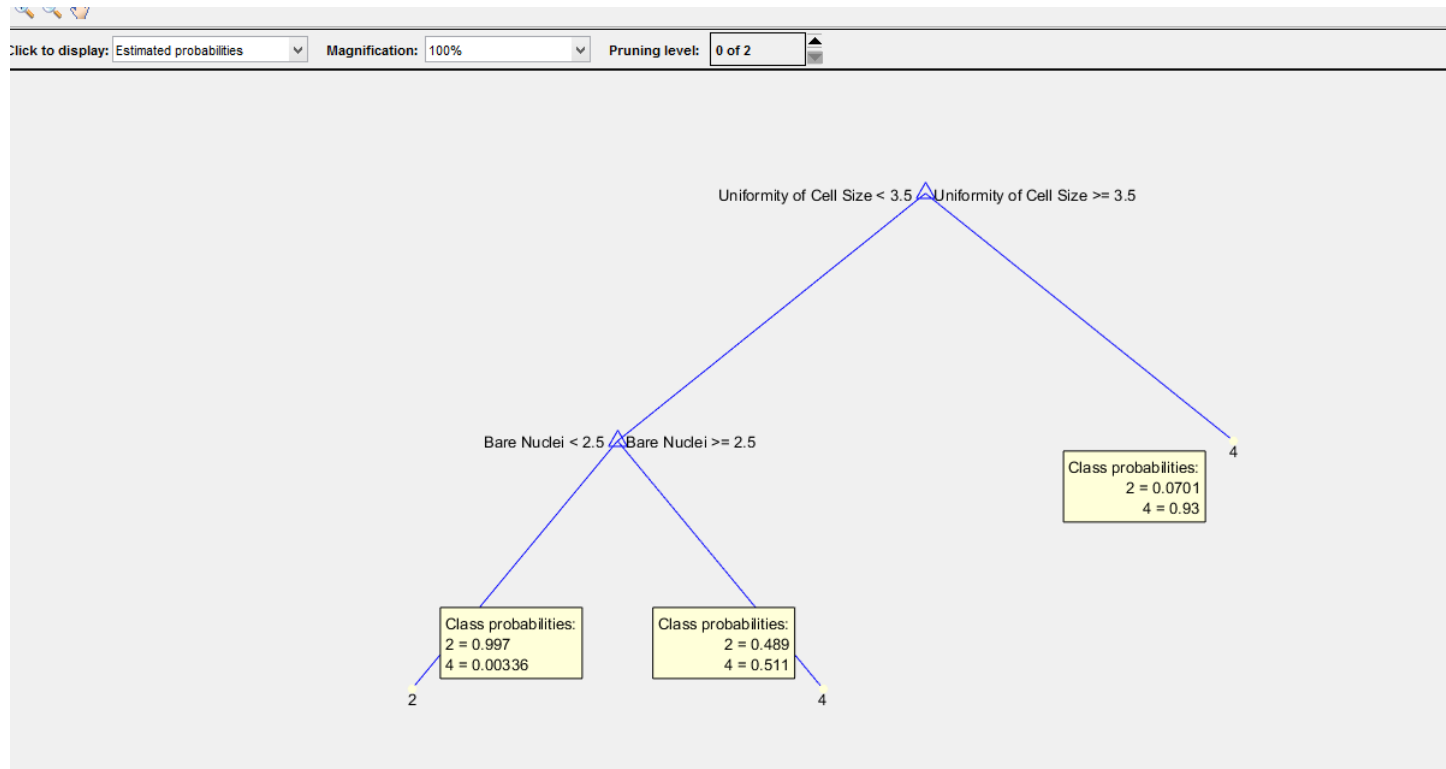
```
view(Training_decision_tree,'Mode','graph');
```

```
num_nodes = Training_decision_tree.NumNodes;
```

following is the decision tree output



By selecting the estimated probabilities option in the decision tree result we will get the probabilities of each class at leaf nodes . which shows the purity level.



when uniformity of cell size ≥ 3.5 **leaf node** has 93% purity

when uniformity of cell size < 3.5 and Bare Nuclei < 2.5 **leaf node** has purity of 99.7%

4th soln)

classified the Test features and Class labels of the testdata(199 records) using the follwoing code

```
TestFeatures = Testdata(:,2:10);  
TestdataGivenLabels= Testdata(:,11);
```

Predicted the labels of testdata with the training data decision tree created in previous solution using the following code

```
TestdataPredictLabes = predict(Training_decision_tree,TestFeatures);
```

Taken the results of original labels and predicted labels of Testdata into OLabels and PLabels and passed these to confusionmat() function of matlab which compares the original labels(OLabels) and predicted labels(PLabels) and created a precision matrix 'e' with TP,FP,FN,TN.

```
OLabels = TestdataGivenLabels;
PLabels = TestdataPredictLabes;
order = [2,4];
[e,order] = confusionmat(PLabels,OLabels,'order',order);

a=e(1,1);
b=e(1,2);
c=e(2,1);
d=e(2,2);
```

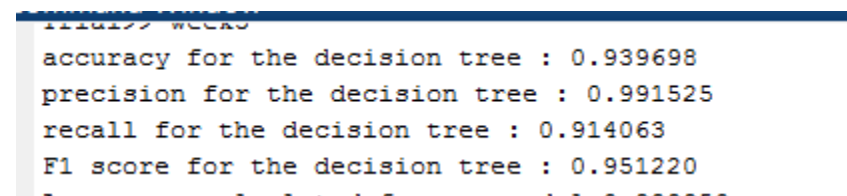
calculated the accuracy , precision, recall, f1 score using the following code

```
accuracy=(a+d)/(a+b+c+d);
precision=a/(a+b);
recall = a/(a+c);
f1_score = 2*((precision*recall)/(precision+recall));

fprintf('accuracy for the decision tree : %f \n',accuracy );
fprintf('precision for the decision tree : %f \n',precision );
fprintf('recall for the decision tree : %f \n',recall );
fprintf('F1 score for the decision tree : %f \n',f1_score );
```

Output:

Accuracy :0.93, precision :0.99, recall :0.91, f1 score: 0.95



```
accuracy for the decision tree : 0.939698
precision for the decision tree : 0.991525
recall for the decision tree : 0.914063
F1 score for the decision tree : 0.951220
```

5th solution)

created the svmmodel using the **fitcsvm()** function of matlab.

created the **svmmmodel** using **fitcsvm** matlab function and created the svm model using the training data of 500 records .

```
SVM_model =  
fitcsvm(Features,Class_Labels,'Standardize',true,'KernelFunction','RBF',...  
'KernelScale','auto');  
Predicted the labels for Testdata using the predict function  
PredictedLabels_SVM = predict(SVM_model,TestFeatures);  
order = [2,4];  
[k,order] =  
confusionmat(PredictedLabels_SVM,TestdataGivenLabels,'order',order);  
%view(SVM_model,'Mode','graph');  
  
p =k(1,1);  
q =k(1,2);  
r =k(2,1);  
s = k(2,2);  
  
accuracy_svm = (p+s)/(p+q+r+s);  
precision_svm = p/(p+q);  
recall_svm = p/(p+r);  
f1_score_svm = 2*((precision_svm*recall_svm)/(precision_svm+recall_svm));  
  
fprintf('Precision calculated for svm model %f \n',precision_svm);  
fprintf('Recall calculated for svm model %f \n',recall_svm);  
fprintf(' F1 score calculated for svm model %f \n',f1_score_svm);
```

Precision : 0.97 , Recall : 0.96 , F1 score :0.96

```
Precision calculated for svm model 0.977099  
Recall calculated for svm model 0.962406  
F1 score calculated for svm model 0.969697  
Trial>>
```

6th soln)

SVM model has better precision, recall and f1 score than decision tree .

svm model

precision - 0.97 , recall - 0.96, f1 score - 0.96

decision tree

precision - 0.99 recall - 0.91 f1 score 0.95

when data is nonlinear **decision tree** cannot classify the data efficiently whereas **svm model** uses **RBF function** and takes long time to train and will classify the non linear data more efficiently .

7th soln)

Misclassification is calculated as follows

```
MisclassificationRate_Decision_Tree = (b*10)+(c*30);
MisclassificationRate_SVM = (q*10)+(r*30);

fprintf('misclassification_rate for decision tree is %f
\n',MisclassificationRate_Decision_Tree);
fprintf('misclassification_rate for SVM model is %f
\n',MisclassificationRate_SVM);
```

output:

```
misclassification_rate for decision tree is 280.000000
misclassification_rate for SVM model is 160.000000
```

8th soln)

```
%%% 8th solution %%%

for i= 1:199

    if ~(TestdataGivenLabels(i)==TestdataPredictLabels(i))
        %%misclassified_record(i) = Test_set(i);
        j=i; %% saving the first instance of missclassified record in j
        break;
    end

end;

fprintf('first instance of missclassified row index is %f \n',j);

m=Trainingdata(:,2:10);
n= (Trainingdata(j,2:10));

[w1,z1]=knnsearch(m,n,'k',1,'distance','euclidean');
disp('One nearest neighbors with their Sample Code number is')
disp(Trainingdata(w1,[1,11]));
disp('Class label assigned with one nearest neighbors is')
```

```

if(mode(Trainingdata(w1,11)== 2))
    disp('Benign')
elseif(mode(Trainingdata(w1,11)== 4))
    disp('Malignant')
end

[w3,z3]=knnsearch(m,n,'k',3,'distance','euclidean');
disp('Three nearest neighbors with their Sample Code numbers are')
disp(Trainingdata(w3,[1,11]));
disp('Mode of the class labels is')
disp(mode(Trainingdata(w3,11)))
disp('Class label assigned with three nearest neighbors is')
if(mode(Trainingdata(w3,11)== 2))
    disp('Benign')
elseif(mode(Trainingdata(w3,11)== 4))
    disp('Malignant')
end

[w5,z5]=knnsearch(m,n,'k',5,'distance','euclidean');
disp('Five nearest neighbors with their Sample Code numbers are')
disp(Trainingdata(w5,[1,11]));
disp('Mode of the class labels is')
disp(mode(Trainingdata(w5,11)))
disp('Class label assigned with five nearest neighbors is')
if(mode(Trainingdata(w5,11)== 2))
    disp('Benign')
elseif(mode(Trainingdata(w5,11)== 4))
    disp('Malignant')
end

[w7,z7]=knnsearch(m,n,'k',7,'distance','euclidean');

disp('Seven nearest neighbors with their Sample Code numbers are')
disp(Trainingdata(w7,[1,11]));
disp('Mode of the class labels is')
disp(mode(Trainingdata(w7,11)))
disp('Class label assigned with Seven nearest neighbors is')
if(mode(Trainingdata(w7,11)== 2))
    disp('Benign')
elseif(mode(Trainingdata(w7,11)== 4))
    disp('Malignant')
end

```

output :

Test_OriginalLabel is
4

Misclassified as
2

first instance of missclassified row index is 5.000000
One nearest neighbors with their Sample Code number is
1185609 4

Class label assigned with one nearest neighbors is
Malignant

Three nearest neighbors with their Sample Code numbers are
1185609 4
706426 4
1148278 4

Mode of the class labels is
4

Class label assigned with three nearest neighbors is
Malignant

Five nearest neighbors with their Sample Code numbers are
1185609 4
706426 4
1148278 4
1174428 4
1002945 2

Mode of the class labels is
4

Class label assigned with five nearest neighbors is
Malignant

Seven nearest neighbors with their Sample Code numbers are

1185609	4
706426	4
1148278	4
1174428	4
1002945	2
320675	4
320675	4

Mode of the class labels is

4

Class label assigned with Seven nearest neighbors is

Malignant

----- |

From the above output, it's evident that knnsearch() method classified the malignant record to malignant class where as decision tree classified the same record as benign which is very expensive

code for all the questions is attached here



week3.m