HOMEWORK #3

Submitted by:

Sreekhar Ale, M.Eng CS,
Course: Intelligent Data Analysis – fall 2015
Language used: MATLAB

ANSWER 1:

List of steps

1. Download the data from the link provided.
2. Open the file and save it with .xlsx extension
3. Read the file into a variable using xlsread function and name the variable as rawdata
4. Using knnimpute function remove the missing values and replace it with the weighted mean from one of the nearest neighbors surrounding it.

MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);
TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);
```

RESULTS:

rawdata = 699 * 11 double values
mydata = 699 * 11 double values

SCREEN SHOT:

Not Applicable

ANSWER 2:

List of steps

1. Perform the same steps from before answer
2. Then select random 500 rows and store it in a variable called TrainingData
3. Select remaining 199 rows and store it in a variable called TestingData

MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);

TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);
```

RESULTS:

rawdata = 699 * 11 double values
mydata = 699 * 11 double values
TrainingData = 500 * 11 double values
ValidationData = 199 * 11 double values

SCREEN SHOT:

Not Applicable

ANSWER 3:

List of steps

1. Continue from the previous steps in the above mentioned answer
2. Keep the first ten rows in a variable called Features
3. And in class label variable keep the last column
4. Construct a decision tree with a minimum leaf size as 25 and display it
5. Check the rule and probability for each leaf node present

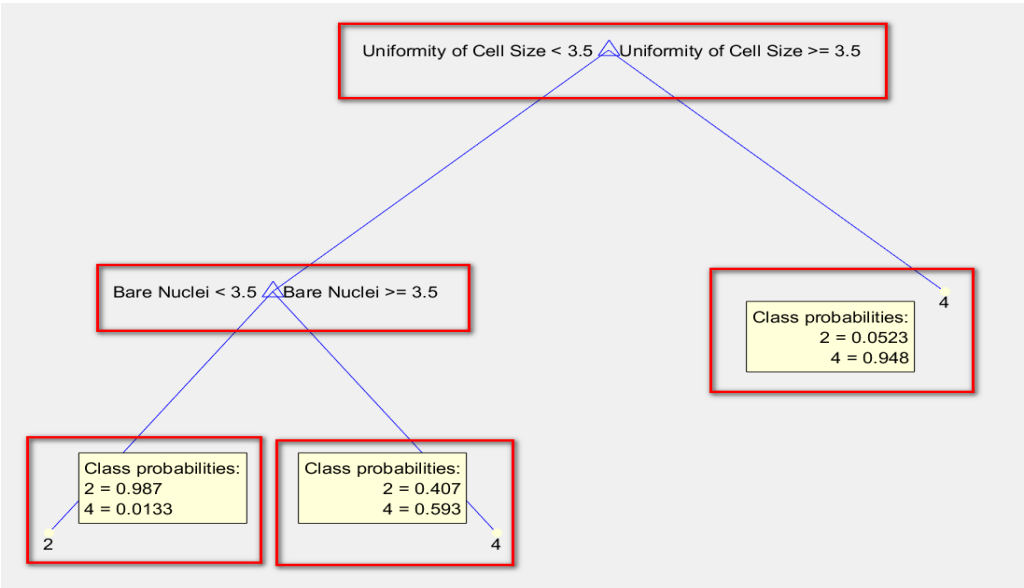
MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);

TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);

Features = TrainingData(:,2:10);
ClassLabels = TrainingData(:,11);
attribute_names={'Clump Thickness','Uniformity of Cell Size','Uniformity of Cell Shape','Marginal Adhesion','Single Epithelial Cell Size','Bare Nuclei','Bland Chromatin','Normal Nucleoli','Mitoses'};
training_decision_tree =
fitctree(Features,ClassLabels,'PredictorNames',attribute_names,'MinLeafSize',25);
view(training_decision_tree,'Mode','graph');
```

RESULTS:

rawdata = 699 * 11 double values
mydata = 699 * 11 double values
TrainingData = 500 * 11 double values
ValidationData = 199 * 11 double values
Features = 500 * 10 double values
ClassLabels = 500 * 1 double values

SCREEN SHOT:



Rules and rule numbers obtained:

1. Uniformity of Cell Size is greater than 3.5 is classified as 4 and has purity of 94.8% and purity number is 0.948
2. Bare nuclei is less than 3.5 is classified as 2 and has purity of 98.7% and purity number is 0.987

ANSWER 4:

List of steps

1. Continue the previous steps mentioned above
2. Create Testing_Features variable and keep the features of the testing data in it
3. Create Testing_OriginalLabels variable and keep the class labels of the testing data in it
4. Create predict_labels and predict the class labels from the decision tree obtained in the previous problem
5. Using confusion matrix calculate accuracy, precision, recall and F1 score

MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);

TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);

Features = TrainingData(:,2:10);
ClassLabels = TrainingData(:,11);
attribute_names={'Clump Thickness','Uniformity of Cell Size','Uniformity of Cell Shape','Marginal Adhesion','Single Epithelial Cell Size','Bare Nuclei','Bland Chromatin','Normal Nucleoli','Mitoses'};
training_decision_tree =
fitctree(Features,ClassLabels,'PredictorNames',attribute_names,'MinLeafSize',25);
view(training_decision_tree,'Mode','graph');

Testing_Features = TestingData(:,2:10);
Testing_OriginalLabels = TestingData(:,11);
predict_labels = predict(training_decision_tree,Testing_Features);
order = [2,4];
```

```
[count ,order]= confusionmat(predict_labels,Testing_OriginalLabels,'order',order);
TPCount_25= count(1,1);
FPCount_25= count(1,2);
FNCount_25= count(2,1);
TNCount_25= count(2,2);
accuracy_25 = (TPCount_25+TNCount_25)/(TPCount_25+FPCount_25+FNCount_25+TNCount_25);
precision_25 = TPCount_25/(TPCount_25+FPCount_25);
recall_25 = TPCount_25/(TPCount_25+FNCount_25);
f1_metric = 2*((precision_25*recall_25)/(precision_25+recall_25));
```

RESULTS:

TPCount_25= 131
FPCount_25= 3
FNCount_25= 10
TNCount_25= 55
accuracy_25 = 0.934673366834171
precision_25 = 0.977611940298508
recall_25 = 0.929078014184397
f1_metric = 0.952727272727273

SCREEN SHOT:

Not applicable

ANSWER 5:

List of steps

1. Perform the same steps as mentioned in the previous problem.
2. Construct the SVM model using the below mentioned function:
   fitcsvm(Features,ClassLabels,'Standardize',true,'KernelFunction','RBF','KernelScale','auto')
3. According to the question, we have to use Kernel Function as parameter with value RBF to tell the trainer to use the Radial Basis Function as the non-linear transformation of the data space.
4. Create predicted_labels_SVM to predict the class labels using the testing features of test data (199 records)
5. Using confusion matrix calculate accuracy, precision, recall and F1 score

MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);

TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);

Features = TrainingData(:,2:10);
ClassLabels = TrainingData(:,11);
attribute_names={'Clump Thickness','Uniformity of Cell Size','Uniformity of Cell Shape','Marginal Adhesion','Single Epithelial Cell Size','Bare Nuclei','Bland Chromatin','Normal Nucleoli','Mitoses'};
training_decision_tree =
fitctree(Features,ClassLabels,'PredictorNames',attribute_names,'MinLeafSize',25);
%view(training_decision_tree,'Mode','graph');

Testing_Features = TestingData(:,2:10);
Testing_OriginalLabels = TestingData(:,11);
predict_labels = predict(training_decision_tree,Testing_Features);
order = [2,4];
[count ,order]= confusionmat(predict_labels,Testing_OriginalLabels,'order',order);
TPCount_25=count(1,1);
FPCount_25=count(1,2);
FNCount_25=count(2,1);
TNCount_25=count(2,2);
accuracy_25 = (TPCount_25+TNCount_25)/(TPCount_25+FPCount_25+FNCount_25+TNCount_25);
precision_25 = TPCount_25/(TPCount_25+FPCount_25);
recall_25 = TPCount_25/(TPCount_25+FNCount_25);
f1_metric = 2*((precision_25*recall_25)/(precision_25+recall_25));

SVM_model = fitcsvm(Features,
ClassLabels,'Standardize',true,'KernelFunction','RBF','KernelScale','auto');
predicted_labels_SVM = predict(SVM_model,Testing_Features);
order = [2,4];
[SVM_test,order] = confusionmat(predicted_labels_SVM,Testing_OriginalLabels,'order',order);
TPCount_SVM=SVM_test(1,1);
FPCount_SVM=SVM_test(1,2);
FNCount_SVM=SVM_test(2,1);
TNCount_SVM=SVM_test(2,2);
accuracy_SVM =
(TPCount_SVM+TNCount_SVM)/(TPCount_SVM+FPCount_SVM+FNCount_SVM+TNCount_SVM);
precision_SVM = TPCount_SVM/(TPCount_SVM+FPCount_SVM);
```

```
recall_SVM = TPCount_SVM/(TPCount_SVM+FNCount_SVM);
f1_metric_SVM = 2*((precision_SVM*recall_SVM)/(precision_SVM+recall_SVM));
```

RESULTS:

TPCount_SVM= 138
FPCount_SVM= 2
FNCount_SVM= 3
TNCount_SVM= 56
accuracy_SVM = 0.974874371859297
precision_SVM = 0.985714285714286
recall_SVM = 0.978723404255319
f1_metric_SVM = 0.982206405693950

SCREEN SHOT:

Not applicable

ANSWER 6:

Performance metrics obtained in above answers are as follows:

TPCount_25= 131
FPCount_25= 3
FNCount_25= 10
TNCount_25= 55
accuracy_25 = 0.934673366834171
precision_25 = 0.977611940298508
recall_25 = 0.929078014184397
f1_metric = 0.952727272727273

TPCount_SVM= 138
FPCount_SVM= 2
FNCount_SVM= 3
TNCount_SVM= 56
accuracy_SVM = 0.974874371859297
precision_SVM = 0.985714285714286
recall_SVM = 0.978723404255319
f1_metric_SVM = 0.982206405693950

From the above analysis we get to know that Accuracy, Precision, Recall and F1 metric scores are more for SVM model when compared to normal decision tree. There is misclassification in decision tree and that's why we have to use SVM model to classify the data. SVM model classifies uses hyperplanes to classify the data and to linearly separate the data. This model classifies with maximum marginal hyperplane and this margin gives largest separational margin. SVM model uses kernel RBF and often takes a lot of time to train. The best classifier for a particular task is itself task-dependent and hence we can tell that SVM model is better that decision tree.

ANSWER 7:

List of steps

1. Implement the previous steps and then as per the question, create a variable cost_FP=10 (cost of predicting an actual benign case as malignant) and cost_FN=30 (cost of predicting an actual malignant case as benign)

2. Calculate the misclassification of the decision tree and SVM model using the below formula:

   Misclassification = (FalsePositive * Cost of FalsePositive) + (FalseNegative * Cost of FalseNegative)

MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-
wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);

TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);

Features = TrainingData(:,2:10);
ClassLabels = TrainingData(:,11);
attribute_names={'Clump Thickness','Uniformity of Cell Size','Uniformity of Cell Shape','Marginal
Adhesion','Single Epithelial Cell Size','Bare Nuclei','Bland Chromatin','Normal Nucleoli','Mitoses'};
training_decision_tree = fitctree(Features,
ClassLabels,'PredictorNames',attribute_names,'MinLeafSize',25);
view(training_decision_tree,'Mode','graph');

Testing_Features = TestingData(:,2:10);
Testing_OriginalLabels = TestingData(:,11);
predict_labels = predict(training_decision_tree,Testing_Features);
order = [2,4];
[count ,order]= confusionmat(predict_labels,Testing_OriginalLabels,'order',order);
TPCount_25=count(1,1);
FPCount_25=count(1,2);
FNCount_25=count(2,1);
TNCount_25=count(2,2);
accuracy_25 = (TPCount_25+TNCount_25)/(TPCount_25+FPCount_25+FNCount_25+TNCount_25);
precision_25 = TPCount_25/(TPCount_25+FPCount_25);
recall_25 = TPCount_25/(TPCount_25+FNCount_25);
f1_metric = 2*((precision_25*recall_25)/(precision_25+recall_25));

SVM_model = fitcsvm(Features,
ClassLabels,'Standardize',true,'KernelFunction','RBF','KernelScale','auto');
predicted_labels_SVM = predict(SVM_model,Testing_Features);
order = [2,4];
[SVM_test,order] = confusionmat(predicted_labels_SVM,Testing_OriginalLabels,'order',order);
TPCount_SVM=SVM_test(1,1);
FPCount_SVM=SVM_test(1,2);
FNCount_SVM=SVM_test(2,1);
```

```
TNCount_SVM=SVM_test(2,2);
accuracy_SVM =
(TPCount_SVM+TNCount_SVM)/(TPCount_SVM+FPCount_SVM+FNCount_SVM+TNCount_SVM);
precision_SVM = TPCount_SVM/(TPCount_SVM+FPCount_SVM);
recall_SVM = TPCount_SVM/(TPCount_SVM+FNCount_SVM);
f1_metric_SVM = 2*((precision_SVM*recall_SVM)/(precision_SVM+recall_SVM));

cost_FP = 10;
cost_FN = 30;
misclassification_decision_tree = (FPCount_25*cost_FP)+(FNCount_25*cost_FN);
misclassification_SVM = (FPCount_SVM*cost_FP)+(FNCount_SVM*cost_FN);
```

RESULTS:

misclassification_decision_tree = 330
misclassification_SVM = 110

SCREEN SHOT:

Not applicable

ANSWER 8:

List of steps

1. From the results obtained in Answer 7, we have to find out a misclassified class label.
2. Then consider Training_features and keep all the training data features in it.
3. Then consider Testing_features and keep all the testing data features in it.
4. Use knnsearch function with 3 nearest neighbors and Euclidean distance and Training features and testing features data and put the data into a matrix
5. Find the three nearest neighbors with their ID numbers
6. Find the mode of the class labels
7. Find which kind of class label it is
8. And repeat from the step 4 and do it for 1,5 and 7 nearest neighbors and find the same above three neighbors, mode and kind of class

MATLAB SOURCE CODE:

```
rawdata = xlsread('D:\My work\M.Eng 1st sem\IDA\Assignment\Homework 3\breast-cancer-wisconsin.xlsx');
mydata = knnimpute(rawdata,1);
NewData = mydata(randperm(699),:);

TrainingData = NewData(1:500,:);
TestingData = NewData(501:699,:);

Features = TrainingData(:,2:10);
ClassLabels = TrainingData(:,11);
attribute_names={'Clump Thickness','Uniformity of Cell Size','Uniformity of Cell Shape','Marginal Adhesion','Single Epithelial Cell Size','Bare Nuclei','Bland Chromatin','Normal Nucleoli','Mitoses'};
training_decision_tree = fitctree(Features,
ClassLabels,'PredictorNames',attribute_names,'MinLeafSize',25);
view(training_decision_tree,'Mode','graph');

Testing_Features = TestingData(:,2:10);
Testing_OriginalLabels = TestingData(:,11);
predict_labels = predict(training_decision_tree,Testing_Features);
order = [2,4];
[count ,order]= confusionmat(predict_labels,Testing_OriginalLabels,'order',order);
TPCount_25=count(1,1);
FPCount_25=count(1,2);
FNCount_25=count(2,1);
TNCount_25=count(2,2);
accuracy_25 = (TPCount_25+TNCount_25)/(TPCount_25+FPCount_25+FNCount_25+TNCount_25);
precision_25 = TPCount_25/(TPCount_25+FPCount_25);
recall_25 = TPCount_25/(TPCount_25+FNCount_25);
f1_metric = 2*((precision_25*recall_25)/(precision_25+recall_25));

SVM_model = fitcsvm(Features,
ClassLabels,'Standardize',true,'KernelFunction','RBF','KernelScale','auto');
predicted_labels_SVM = predict(SVM_model,Testing_Features);
order = [2,4];
[SVM_test,order] = confusionmat(predicted_labels_SVM,Testing_OriginalLabels,'order',order);
TPCount_SVM=SVM_test(1,1);
FPCount_SVM=SVM_test(1,2);
FNCount_SVM=SVM_test(2,1);
TNCount_SVM=SVM_test(2,2);
accuracy_SVM =
(TPCount_SVM+TNCount_SVM)/(TPCount_SVM+FPCount_SVM+FNCount_SVM+TNCount_SVM);
precision_SVM = TPCount_SVM/(TPCount_SVM+FPCount_SVM);
```

```
recall_SVM = TPCount_SVM/(TPCount_SVM+FNCount_SVM);
f1_metric_SVM = 2*((precision_SVM*recall_SVM)/(precision_SVM+recall_SVM));

cost_FP = 10;
cost_FN = 30;
misclassification_decision_tree = (FPCount_25*cost_FP)+(FNCount_25*cost_FN);
misclassification_SVM = (FPCount_SVM*cost_FP)+(FNCount_SVM*cost_FN);

for iRowCount= 1:199
   if(~(Testing_OriginalLabels(iRowCount) == predict_labels(iRowCount)))
      RowIndex=iRowCount;
      disp('Original Label is')
      disp(Testing_OriginalLabels(iRowCount))
      disp('Misclassified as')
      disp(predict_labels(iRowCount))
      break;
   end
end
disp('Misclassified row index is')
disp(RowIndex)
Training_features = TrainingData(:,2:10);

TestingData_features = (TestingData(RowIndex,2:10));
[var3,nvar3]=knnsearch(Training_features,TestingData_features,'k',3,'distance','euclidean');
disp('Three nearest neighbors with their ID numbers are')
disp(TrainingData(var3,[1,11]));
disp('Mode of the class labels is')
disp(mode(TrainingData(var3,11)))
disp('Class label assigned with three nearest neighbors is')
if(mode(TrainingData(var3,11)== 2))
   disp('Benign')
elseif(mode(TrainingData(var3,11)== 4))
   disp('Malignant')
end

[var1,nvar1]=knnsearch(Training_features,TestingData_features,'k',1,'distance','euclidean');
disp('One nearest neighbors with their ID number is')
disp(TrainingData(var1,[1,11]));
disp('Class label assigned with one nearest neighbors is')
if(mode(TrainingData(var1,11)== 2))
   disp('Benign')
elseif(mode(TrainingData(var1,11)== 4))
   disp('Malignant')
end
```

```
[var5,nvar5]=knnsearch(Training_features,TestingData_features,'k',5,'distance','euclidean');
disp('Five nearest neighbors with their ID numbers are')
disp(TrainingData(var5,[1,11]));
disp('Mode of the class labels is')
disp(mode(TrainingData(var5,11)))
disp('Class label assigned with five nearest neighbors is')
if(mode(TrainingData(var5,11)== 2))
   disp('Benign')
elseif(mode(TrainingData(var5,11)== 4))
   disp('Malignant')
end

[var7,nvar7]=knnsearch(Training_features,TestingData_features,'k',7,'distance','euclidean');
disp('Seven nearest neighbors with their ID numbers are')
disp(TrainingData(var7,[1,11]));
disp('Mode of the class labels is')
disp(mode(TrainingData(var7,11)))
disp('Class label assigned with Seven nearest neighbors is')
if(mode(TrainingData(var7,11)== 2))
   disp('Benign')
elseif(mode(TrainingData(var7,11)== 4))
   disp('Malignant')
end
```

RESULTS:

```
Original Label is
   4

Misclassified as
   2

Misclassified row index is
   25

Three nearest neighbors with their ID numbers are
    1168736         4
     785615         4
    1106829         4

Mode of the class labels is
    4
```

Class label assigned with three nearest neighbors is
Malignant
One nearest neighbors with their ID number is
1168736 4

Class label assigned with one nearest neighbors is
Malignant
Five nearest neighbors with their ID numbers are
1168736 4
785615 4
1106829 4
1293439 2
1185609 4

Mode of the class labels is
4

Class label assigned with five nearest neighbors is
Malignant
Seven nearest neighbors with their ID numbers are
1168736 4
785615 4
1106829 4
1293439 2
1185609 4
1105257 4
1016277 2

Mode of the class labels is
4

Class label assigned with Seven nearest neighbors is
Malignant

Comments

From above we get to know that, the above original data is 4 and is misclassified as 2. But from SVM model we know that original data is classified as 4 itself and hence is a best model for classification.

SCREEN SHOT:

Not applicable