Assignment 1

CS6052: Intelligent Data Analysis: Fall 2017

Name: Aditya Gurram

ID: M 12585180

Email: gurramva@mail.uc.edu

Please find the attachment below for the complete python code (Questions 1,2,3):

DecisionTrees.py

1. Normalize the columns for their values to be in uniform ranges. Describe the process you followed to do the normalization.

   Answer:-

   - First, loop through all the columns and find the max and min value for the entire column and for each row calculate the (dataVal-minVal)/(max-min) and do the process for each and every element of the data matrix

```
#This function normalise the given input data
def normalizeValues(datamatrix):
    newMatrix = np.zeros((len(datamatrix),len(datamatrix[0])), dtype=np.float32)
    for column in range(len(datamatrix[0])):
        minVal = np.amin(datamatrix[:,column])
        maxVal = np.amax(datamatrix[:,column])
        denominator = maxVal - minVal
        for row in range(len(datamatrix)):
            newMatrix[row,column] = (datamatrix[row,column] - minVal)/denominator
    return newMatrix
```

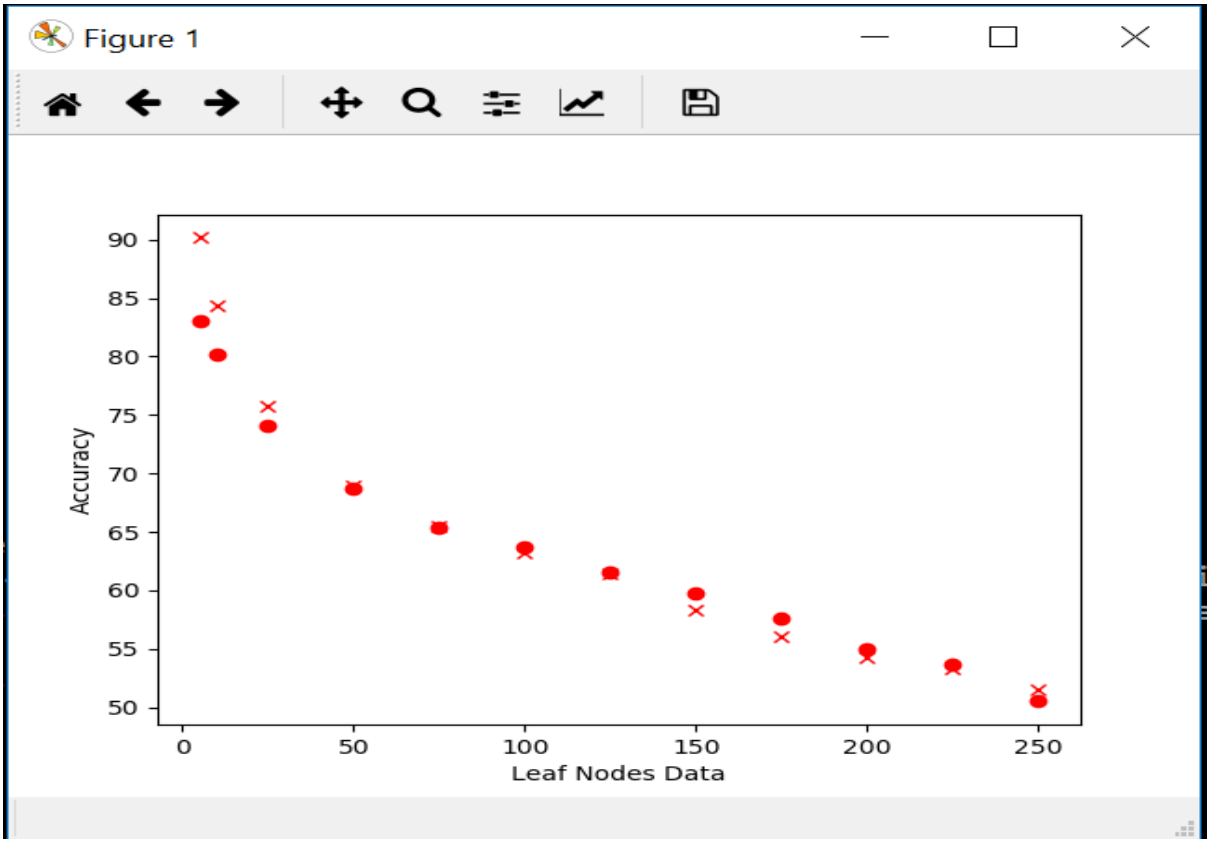
2. Split the dataset into three randomly selected parts: 12000 instances for training, 4000 for validation, and 4000 for testing. Describe how you made these partitions

Answer:-

- Read the entire data set using the function “pandas.read_csv” and load them into data object.
- Use the train_test_split function to split the “20000” data set into 12000 for training , 4000 for validation and 4000 for testing.
- Do this twice once the validation split is done and then use the 16000 of training set to get the 4000 test data.

```
#This function reads the input data file(.csv) and partition the data as Test, Train and Validation using  train_test_split function
def BuildDecisionTree():
    Alphabet_data = pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-databases/letter-recognition/letter-recognition.data',sep= ',', header= None)
    print ("Dataset Dimension  ", Alphabet_data.shape)
    print ("Dataset::")
    Alphabet_data.head()
    X = Alphabet_data.values[:, 1:17]
    Y = Alphabet_data.values[:,0]
    Xnorm=normalizeValues(X)
    print("the normlaised data is ", Xnorm)
    X_train, X_test, y_train, y_test = train_test_split(Xnorm, Y, test_size = 0.2, random_state = 1)#train: 12000 #test: 4000 #val:4000
    X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, random_state=1)#validation split
    return X_train,X_val,X_test, y_train, y_val,y_test
```

3. Plot the accuracy values of the trees for all the above cases, and for the training and the validation datasets. How do you interpret the plots?
   Answer:-
   - The below graphs represents plot between Accuracy and Size of Lead node which ranges from 0 to 250.
   - Here the circular dots represents Validation data set accuracy and Cross sign (X) represents training data accuracy for the model.
   - From the graph it can inferred that the accuracy is decreasing when the sample size is increasing .



A. Which decision tree is the best from among all the above decision trees and why?
B. Show the selected (best) decision tree in tabular (list of rules) or tree structure
The decision tree with the least sample size at leaf nodes , accuracy is higher. 5 sample size has greater accuracy which is the best one to be selected. Please open the below DSTree.pdf in google chrome for better visualisation.

C. Use the selected decision tree to compute the Confusion Matrix for the test data set.

Command Prompt

```
The confusion matrix is  [[149   0   0   1   0   0   0   1   1   0   2   4   0   0   0   0   1   0
    0   0   1   0   0   1   0   1]
 [  0 114   1   4   2   1   2   1   6   1   1   0   0   1   2   0   2   5
    3   1   0   3   0   1   0   1]
 [  0   0 129   0   5   0   1   0   0   0   2   2   0   0   3   0   0   0
    0   1   1   0   0   0   0   0]
 [  2   5   0 133   0   0   0   1   0   0   0   3   0   2   4   0   0   4
    0   0   0   1   0   1   0   0]
 [  0   0   5   1 110   0   5   0   0   0   2   2   0   0   0   0   0   1
   11   0   0   0   0   1   0   1]
 [  0   0   0   1   2 113   0   2   2   1   1   0   0   0   0   5   3   1
    0   3   0   1   1   2   6   0]
 [  0   1   5   7   4   2 121   0   1   0   2   1   0   0   4   0   2   0
    2   0   1   0   0   0   0   1]
 [  2   9   4   7   0   2   3  96   0   0   7   0   2   1   1   0   2   6
    1   0   1   1   0   0   0   1]
 [  1   0   1   0   0   2   0   0 126   2   1   1   0   0   0   2   0   1
    0   1   0   0   0   0   0   0]
 [  0   1   1   1   1   2   0   0   3 123   0   0   0   0   1   1   1   0
    2   0   0   0   0   3   0   1]
 [  0   3   4   2   4   0   0   7   0   0 105   1   0   1   0   0   0   4
    0   2   0   0   0   4   0   0]
 [  3   0   1   0   3   0   0   3   0   0   1 122   0   1   0   0   1   0
    2   1   0   1   0   0   0   0]
 [  0   1   0   2   0   0   5   1   0   0   1   0 130   2   0   0   0   0
    0   0   0   1   2   0   0   0]
 [  0   0   1   4   0   2   1   0   0   1   2   0   7 129   1   2   0   0
    0   0   4   2   2   0   1   2]
 [  1   4   7   4   1   1   2   4   1   0   0   0   0   1 130   0   5   3
    1   0   4   1   2   1   0   0]
 [  0   3   0   3   1  15   2   0   1   2   0   0   1   1   2 141   0   0
    0   0   0   1   0   0   1   0]
 [  0   1   4   2   1   0   1   3   0   0   0   1   0   0   1   2 135   1
    1   2   1   0   0   2   0   1]
 [  2  13   1   4   3   0   1   1   0   0   4   2   4   0   2   0   0 115
    3   1   0   1   0   0   0   0]
 [  2   7   2   1   3   1   4   1   3   0   0   2   0   0   1   0   3   2
  130   2   0   0   0   1   0   5]
 [  1   2   8   1   0   1   0   2   0   0   1   0   0   1   0   2   1   0
    3 153   0   1   0   0   2   2]
 [  0   0   2   2   0   0   4   2   0   0   1   0   0   5   3   0   1   0
```

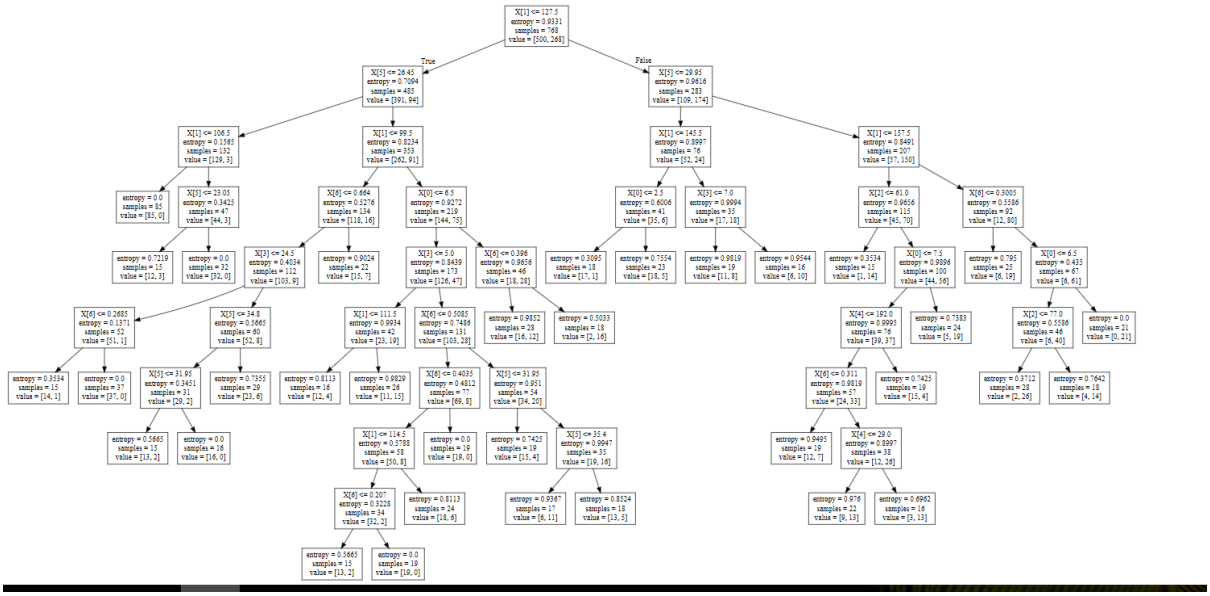
D. Compute Precision, Recall, and F metrics for any randomly selected three of the 26 classes.

```
Calculating Precision, Recall, F Metric for class
Precision for class N is 0.807453416149
Recall for class  N is 0.751445086705
F metric for class N is  0.778443113772

Calculating Precision, Recall, F Metric for class
Precision for class Y is 0.886075949367
Recall for class  Y is 0.886075949367
F metric for class Y is  0.886075949367

Calculating Precision, Recall, F Metric for class
Precision for class T is 0.896103896104
Recall for class  T is 0.857142857143
F metric for class T is  0.87619047619
```

4. Please find the attachment below for the complete python code for Prima Indians problem.

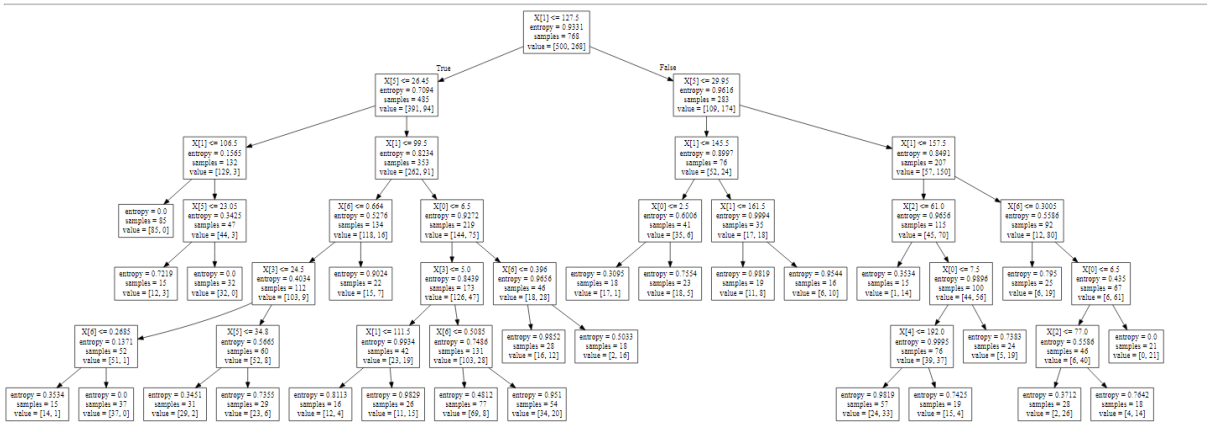
KFD.py

A. Show the decision tree in the table/list/tree form



B. Find the best possible decision tree by adopting the appropriate parameter values. Show this decision tree

```
The scores for 10 folds are  [ 0.71428571  0.74025974  0.68831169  0.72727273  0.76623377  0.81818182
  0.77922078  0.83116883  0.72368421  0.80263158]
```



C. Show the parameters that yield this tree and also list its precision, recall, and accuracy values.

```
The scores for 10 folds are  [ 0.71428571  0.74025974  0.68831169  0.72727273  0.76623377  0.81818182
  0.77922078  0.83116883  0.72368421  0.80263158]
Grid Parameter Search using 10-fold Cross Validation

Grid Search Cross Validation Operation parameter settings.

-- Best Parameters:
parameter: criterion           setting: entropy
parameter: max_depth           setting: 5
parameter: max_leaf_nodes      setting: 50
parameter: min_samples_leaf    setting: 15
parameter: min_samples_split   setting: 5


-- Testing best parameters [Grid]...
mean: 0.763 (std: 0.055)
```

```
The confusion Matrix [[74 27]
 [25 28]]
The Accuracy is  0.662337662338
The Precision is  0.747474747475
The Recall is  0.732673267327
```