CSEP 546: Data Mining (Spring 2017)
Assignment 1: Clickstream Mining and Rule Induction

- Nithin Mahesh

Code:

The code for this assignment is submitted as a zip file. It is written in Python 3 (Tested with Python 3.5). All the core logic is present in the file main.py. Please run the following command to install all dependencies.

pip install -r requirements.txt

The code submission also has various outputs available that were captured by me while building up the core logic. The code is also available at https://github.com/nithinmahesh/DataMiningAssignments/tree/master/Assignment1 while the training data from the zip file needs to be placed in the same folder this is cloned to.

Here are the key functions in the code:

1. **ID3 decision tree builder**
    - *def ID3(Examples, Targetattribute, Attributes, confidence, usegainratio)*
    - This is the function which implements the ID3 decision tree trainer. This takes the training data and various configurations as input and builds an appropriate decision tree for the training set based on the configurations provided. The configurations decide the confidence level used for Chi Square Split Stopping Test and also decide whether to use gain ratio or gain for choosing the best attribute.
2. **Choosing Best Attributes for Current Node**
    - This set of functions are responsible for choosing the best attribute for splitting at the current level.
    - *def ChooseBestAttribute(Examples, Targetattribute, Attributes, confidence, usegainratio):*
        1. *i.* This is the main function that chooses the best attribute at the current node to split the tree further. This may end up calling one of the below two functions based on the provided configuration
    - *def ChooseBestAttributeByGain(Examples, Targetattribute, Attributes):*
        1. *i.* This chooses the best attribute among the given attributes based on information gain as the deciding factor. The logic to calculate

information gain for every attribute and choosing the best attribute lies within this function.

- *def ChooseBestAttributeByGainRatio(Examples, Targetattribute, Attributes):*
    i. This chooses the best attribute among the given attributes based on information gain ratio as the deciding factor. The logic to calculate information gain ratio for every attribute and choosing the best attribute lies within this function.

3. **Split Stopping through Chi Square Test**
    - *def ShouldStopByChiSquare(Examples, Targetattribute, Attributes, bestattr, confidence):*
    - This function implements the calculation of Chi Square Test Statistic and compares the calculated value from the critical value obtained through the library functions. It further returns a Boolean value that informs the higher layer whether to stop or proceed further tree splitting.
4. **Evaluation Functions**
    - These functions are responsible for evaluating the built tree with test data.
    - *def Evaluate(tree, testSet, TargetAttribute, Attributes):*
        i. This function takes a set of test data and predicts the expected value for every test case and calculates the accuracy obtained by the given tree.
    - *def GetPrediction(tree, test, TargetAttribute, Attributes):*
        i. This function takes in a single test case and predicts the target value by making a pass from the root of the tree to a leaf node. This is called by Evaluate function for every test case
5. **Main Function**
    - The main code loads the training data, runs the decision tree builder and evaluates the tree with validation and test data set and reports on it.
    - It repeats the above for various configurations that have been configured.
6. **Helper functions**
    - To make various repeatable calculations and display of tree easier, there are a bunch of additional helper functions for these tasks.

Analysis of results:

The training data was split into two to create a training data (75%) and validation data(25%). Following are the results for accuracies obtained with this training data set.

Confidence Level (%)	**Prediction Accuracy using Gain (%)**	**Prediction Accuracy using Gain Ratio (%)**
0	70.932	71.016
1	72.516	73.840
10	72.256	74.184
90	74.280	**74.832**
95	74.536	**74.832**
99	74.632	**74.832**

The best configuration that works is by using the Gain Ratio at confidence level higher than 90%. The best prediction accuracy obtained is 74.832% but the interesting thing is the tree built for this configuration contains only one node which returns False. At lower confidence levels, the tree is built up to ~24000 nodes but gives much poorer accuracy.

When we use the complete training data for training the decision tree classifier, the result is much better. We can achieve an accuracy of 75.084% with this.

Confidence Level (%)	**Prediction Accuracy using Gain (%)**	**Prediction Accuracy using Gain Ratio (%)**
0	70.088	70.316
95	74.756	74.852
99	**75.084**	74.852

This shows that with more training data, we can model the decision tree classifier better to achieve better accurate predictions. Following are the precision and recall rates for the same.

Confidence Level (%)	**Precision using Gain (%)**	**Precision using Gain Ratio (%)**
0	34.700	35.877
95	48.536	51.879
99	**55.977**	51.879

We can see that the precision is also highest for our best performing tree based on accuracy.

Confidence Level (%)	**Recall using Gain (%)**	**Recall using Gain Ratio (%)**
0	21.376	22.790
95	5.006	1.096
99	**4.688**	1.096

Recall is not the lowest for our best performing tree. By recall, our decision tree classifier built through gain ratio as the choice of choosing best attribute fares the best.

Best Performing Tree:

We will consider the tree built using gain and 99% confidence with all data as training data that gives a prediction accuracy of 75.084% as the best performing tree.

The path that contains the largest fraction of true labeled data is the following:

If Session Request Count is between 1.5 and 2.5 and Assortment Level 2 Path Last is /Assortments/Main/UniqueBoutiques and Send Email is NULL and Session First Referrer Top 5 is Other and Num main Template Views is between 0.5 to 1.5, then the user would view another page on the site.

The path that contains the largest fraction of false labeled data is the following:

If Session Request Count is less than 1.5 and Session Browser Family is Netscape and Session First Referrer Top 5 is Other and Session Visit Count is less than 1.5 and Send Email is NULL, then the user would ***not*** view another page on the site.