Clickstream Mining with Decision Trees

Analyzing data obtained from web server logs, so-called "Clickstreams", which are saved on the web and proxy servers, when users are visiting them is rapidly becoming one of the most important activities for companies in any sector as most businesses become e-businesses. Clickstream Analysis can reveal usage patterns on the company's web site and give a highly improved understanding of customer behavior. This understanding can then be utilized for improving customer satisfaction with the web site and the company in general, yielding a huge business advantage (Compare cp. Andersen J., Giversen A., Jensen A. H., Larsen R.S., Pedersen, T.B., Skyt J, 2000).

In this task, we implemented a decision tree based classifier (ID3) which predicts, given a set of page views, will the visitor view another page on the site or will he leave?

Learning in decision tree:

Following steps are performed while learning a decision tree:

```
1. X = the best decision attribute for next node
2. Assign X as decision attribute for node
3. For each value of X, create new descendant of node
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then
           STOP
   Else
           Iterate over new leaf nodes
```

Determination of best decision tree attribute:

We choose a decision tree attribute which results into a good split. A good split is one where we are more certain about the classification after the split- uniform distribution of data after the split is considered bad.

This is accomplished using the attribute which results into maximum information gain,

$$\arg\max_i I(Y, X_i) = \arg\max_i [H(Y) - H(Y|X_i)]$$

$H(Y)$ – entropy of Y $H(Y|X_i)$ – conditional entropy of Y

Feature which yields maximum reduction in entropy provides maximum information about Y

Issues with decision trees:

Standard decision trees have no learning bias and in such situation the training error may become zero. Nevertheless, this results into over-fitting. This over-fitting can be prevented if we build simpler tree by performing pruning.

Resolution- Chi-square Test

The approach which we followed in current implementation is Post-pruning based on Chi-square test, which is done as follows.

Build the full decision tree as before

1. When you can grow it no more, start to prune:
2. Beginning at the bottom of the tree, delete splits in which pchance > MaxPchance
3. Continue working your way up until there are no more prunable nodes

In order to perform pruning we evaluate the chi-square statistics (Dev(X)) of the data. Then we compute the probability (pchance) of chi-squared statistic with $v - 1$ degrees of freedom ≥ Dev(X). If the computed probability is lesser than the MaxPChance then we prune the tree as it represent irrelevance between that decision tree attribute and outcome of the decision tree.

Implementation:

Main():
This is the main function, where all the methods to extract data from the excel ,creating the decision tree and the test data classification methods are called.

classify_test_instance(node,data):
This method is used to recursively traverse the tree and classify the test data. This is a recursive method.
The tree is traversed till a leaf node is reached and the test data is classified on the leaf node.

create_decision_tree():
This is the main method which creates the decision tree in a recursive manner.
Based on the attribute, the sample space is split into at most two parts and assigned to the children. Pruning methods are done by calculating the chi square statistic. In addition, other pruning is also done, based on the number of true and false values in each attribute. For ex: If

there is a 100% majority of either True or False values, the node is terminated as a leaf node and further tree building does not happen in that node.

find_next_attribute(attribute_flag,attribute_sample_space):
This method is used to calculate the entropy for each attribute, depending on the sample space of the current node and hence, is used to pick the attribute with the maximum information.

attribute_domain():
This method is used to find what different ranges of values are, the attribute takes and this can be used to take care of the splitting.

csvtest():
This procedure is used to extract the data from the csv files and put them into usable lists.

Execution:

Instructions to follow:

1. Store the traindataset, trainlabs, test data set and test labs and give the file paths in the function, csvtest().
2. The threshold values can be changed in the function, create_decision_tree(node). Kindly search for p_value and the change the threshold in the condition.
3. Just run the csvtst.py.