A Report on
ID3 Algorithm Implementation and Observed Accuracy

by

Kshitij Sharma - 2012A7PS009H
Rohit Sharma - 2012A7PS050H
Abhishek Kaushik - 2012A7PS056H
Prakhar Gupta - 2012A7PS059H

BITS F464 - Machine Learning

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)
HYDERABAD CAMPUS

(12th November 2014)

Contents

1. Abstract
2. Code Description
3. Control Experiment
4. Summary

Abstract

This report covers an assignment on building a Decision Tree for predicting whether the income of a person is less than or equal to fifty thousand or greater than fifty thousand. The data used for building the decision tree is the census-income dataset from the US Census Bureau. The algorithm used for building the Decision tree is ID3 by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking.The language choosen for implementation is Java. The Decision Tree constructed from the training dataset is then tested on 500 examples and it's accuracy on predicting the income over these examples is recorded.

Code Description

The ID3 algorithm has been implemented in Java. The code has been written entirely by ourselves.

Algorithm:

The ID3 algorithm begins with the original set ***S*** as the root node. On each iteration of the algorithm, it iterates through every unused attribute of the set ***S*** and calculates the entropy and the information gain for that attribute. It then selects the attribute which has the smallest entropy (or largest information gain) value. The set ***S*** is then split by the selected attribute (example- age < 40, age >= 40) to produce subsets of the data. The algorithm continues to recurse on each subset, considering only attributes never selected before.

Recursion on a subset may stop in one of these cases:

•every element in the subset belongs to the same class (+ or -), then the node is turned into a leaf and labelled with the class of the examples
•there are no more attributes to be selected, but the examples still do not belong to the same class (some are + and some are -), then the node is turned into a leaf and labelled with the most common class of the examples in the subset
•there are no examples in the subset, this happens when no example in the parent set was found to be matching a specific value of the selected attribute, for example if there was no example with age >= 40. Then a leaf is created, and labelled with the most common class of the examples in the parent set.
Throughout the algorithm, the decision tree is constructed with each non-terminal node representing the selected attribute on which the data was split, and terminal nodes representing the class label of the final subset of this branch.

Experiment

1. The attributes which have continuous values have been discretized by taking the mean of all the attribute values in the instances present in the dataset.

2. The decision tree is constructed using the entire census-income.data.docx which consists of 32,561 examples used as training data.

3.The resultant decision tree is tested using five-hundred examples from file census-income.test.docx which did not contain missing attribute values.

4. The accuracy of the tree over the testing data is recorded.

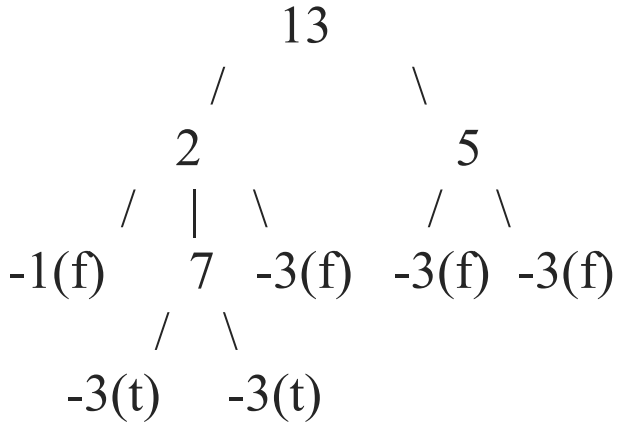
Summary

Accuracy of the resulting tree over the testing data : 75.2 %

The resultant decision tree is included in the file “decisionTree.txt”

in the form of pre-order traversal.

Example: For the tree,



the pre-order traversal as shown in the file would be:

13 → 2 → -1
7 → -3
-3

-3

5 → -3
-3

where leaves are depicted as two negative numbers -1 and -3 and hence they also have their classification value indicated as True(t) or False(f).

Strengths:

1. The decision tree constructed can handle dataset with continuous valued attributes.

2. The decision tree is constructed using all the examples from the training dataset. So, the code is capable of handling even enormous datasets.

Weaknesses:

1. No kind of pruning has been performed while constructing the decision tree. Hence there is a possibility of over-fitting the data in case of large data sets.

2. The testing data must not contain unknown attributes.

3. For discretizing the continuous valued attributes, we have taken the mean of all the attribute values to classify them into two categories “high” and “low”.