|  |  |  |
| --- | --- | --- |
| Picture1 | ***DEPARTMENT OF COMPUTER ENGINEERING EXPERIMENT NO. 3*** | |
|  | |
| Semester | | BE Semester VIII – Computer Engineering |
| Subject | | Machine Learning |
| Professor In-charge | | Prof. Avinash Shrivas |
| Laboratory In-charge | | Prof. Suja Jayachandran |
| Laboratory | | M 310 |

|  |  |  |
| --- | --- | --- |
| Student Name | Chinmay Terse | |
| Roll Number | 13106B0004 | |
| Grade and Professor’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment No. | 3 | |
| Experiment Title | Construction of a decision tree using the ID3 algorithm. | |
| Resources/Apparatus Required | Hardware:  IBM PC compatible computer system | Software:  Eclipse, Java, Weka Tool |
| Objectives | * Learn about the concept of Decision Trees * Learn about the ID3 algorithm * Implement a program to construct a decision tree using ID3 algorithm | |
| Theory of Operation | A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represents classification rules. A decision tree consists of 3 types of nodes:   1. Decision nodes - commonly represented by squares 2. Chance nodes - represented by circles 3. End nodes - represented by triangles   In decision tree learning, **ID3** (**Iterative Dichotomiser 3**) is an algorithm invented by Ross Quinlan[1] used to generate a decision tree from a dataset. ID3 is the precursor to the C4.5 algorithm, and is typically used in the machine learning and natural language processing domains.  Psuedo code:  ID3 (Examples, Target\_Attribute, Attributes)  Create a root node for the tree  If all examples are positive, Return the single-node tree Root, with label = +.  If all examples are negative, Return the single-node tree Root, with label = -.  If number of predicting attributes is empty, then Return the single node tree Root,  with label = most common value of the target attribute in the examples.  Otherwise Begin  A ← The Attribute that best classifies examples.  Decision Tree attribute for Root = A.  For each possible value, *vi*, of A,  Add a new tree branch below Root, corresponding to the test A = *vi*.  Let Examples(*vi*) be the subset of examples that have the value *vi* for A  If Examples(*vi*) is empty  Then below this new branch add a leaf node with label = most common target value in the examples  Else below this new branch add the subtree ID3 (Examples(*vi*), Target\_Attribute, Attributes – {A})  End  Return Root  We use the Weka tool to create the Decision tree. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. | |
| Implementation | import java.util.Scanner;  import weka.classifiers.trees.Id3;  import weka.classifiers.trees.J48;  import weka.core.Instances;  import weka.core.converters.ConverterUtils.DataSource;  import weka.filters.Filter;  import weka.filters.unsupervised.attribute.Remove;  public class ID3Algorithm {  public static void main(String[] args) {  DataSource source = null;  Instances data = null;  try {  source = new DataSource("Datasets/ds1.csv");  data = source.getDataSet();  } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  Instances newData = null;  try {  if( isAttributeNumeric(data, 0) ) {  String[] options = weka.core.Utils.splitOptions("-R 1");  Remove remove = new Remove();  remove.setOptions(options);  remove.setInputFormat(data);  newData = Filter.useFilter(data, remove);  } else {  newData = data;  }  newData.setClassIndex(newData.numAttributes() - 1);  } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  System.out.println(newData+"\n");  Id3 id3 = new Id3();  try {  String[] classifierOptions = weka.core.Utils.splitOptions("-U");  id3.setOptions(classifierOptions);  id3.buildClassifier(newData);  } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  System.out.println("-----------------\n"+id3);  }  private static boolean isAttributeNumeric(Instances data, int i) {  // TODO Auto-generated method stub  String temp = data.attributeStats(i).toString();  Scanner t = new Scanner(temp);  while(t.hasNext()) {  if( t.next().equalsIgnoreCase("NUM") )  return true;  }  return false;  }  } | |
| Output | Input:    Output: | |
| Conclusion | Thus, we learnt that decision trees help us to create a sequence of questions that we can ask to the dataset to get to a specific output. We also learnt that the ID3 algorithm used to select the nodes with the highest purity for selection. We learn that the nodes with the highest purity are the ones with the lowest entropy. We learnt about the Weka tool and used to successfully to implement a program to create a decision tree give a dataset in the .arff or the .csv format. | |