**Purpose**

Decision trees are supervised pattern classifiers that are designed to work with categorical (non-numeric) data. They can function well when there are missing data values or irrelevant attributes in the data set. Decision tree induction is the process of learning an appropriate decision tree for a particular problem based on labeled samples.

**Goals**

        Learn how decision trees work

        Understand how to apply decision trees for supervised classification

        Understand which data sets decision trees are appropriate for

**Description**

Write a program to perform supervised pattern classification using decision trees as described in class and in your textbook (p. 164). Your program should have two components: training and application. In training mode, a data set with labeled samples will be presented to the program, and it will use those samples to generate a decision tree. In application mode, the program will use the decision tree generated by the training mode and use it to classify samples with unknown class assignment. You have to implement these two modes in one program. Your program may do pruning, but it is not required. A sample data set will be provided.

**Input**

The sample data set to be used for this project is given below. The bcwdisc.arff file is a discretized version of a breast cancer database published by the University of Wisconsin Hospitals. The original data set was obtained from the University of California Irvine (UCI) machine learning repository (<http://archive.ics.uci.edu/ml/> ). The data set is in ARFF format, with one sample / pattern vector per line of input. Each line will contain a series of attribute values, separated by commas or spaces. The file has an ARFF header consisting of a set of tags beginning with the ‘@’ character. These tags provide descriptions of the attributes (name and type). The data begins after the @data tag, which denotes the end of the header.

* bcwdisc.arff                (Many different types of ML problems, copied from)

**Outputs**

For each input data set, your program should produce three outputs:

1. Training Mode: Produce a text file (“YourLoginDTTrainMInputFile.dt”) with the decision tree in it. M indicates the percentage of data used for training.Write your nodes in depth-first order with one node each line. Each node should occupy one line in the file, with the node test clearly labeled (check Figure 4.18 on page 168 as an example). As the depth of the tree increases, increase the indentation. For example (to show the structure of the tree),

Uniformity =0

| Thickness =2

| | Mitoses =0

| | | Nuclei =1 2

| | | Nuclei =2 4

| | Mitoses =2

The values at the end of Nuclei lines indicate the class.

Explanation on the implementation of the algorithm and differences from the example in the textbook:

    The attribute values in the book example have continuous values. The book applies binary split algorithm for that purpose. You do not need to consider binary split for the assignment.

    The bcwdisc data set has only discrete values. Each attribute has only three values (like the age attribute in the example I provided in the class). So, this is a multiway split. You may assume that I will test your data with categorical data with at most 3 classes as in the sample data set.

    You do not need to check the inequality for the values but rather check the equality. Whenever you select an attribute, it is likely to have 3 branches.

    The book example works on continuous attributes, therefore it may use the same attribute down the tree. In the assignment, you will not need to use an attribute down the tree again.

1. Application Mode: Produce an output file (“YourLoginDTApplyMInputFile”) with the class labels for each input pattern vector, in the same order as the vectors in the input file. Add one more attribute to your ARFF file as *@attribute dt\_class real*as the last attribute for the predicted class. In addition, produce an output file (“YourLoginDTAccuracyMInputFile”) that has the confusion matrix and the accuracy of the classification. Run your decision tree for the complete dataset (including the training set).

**Deliverables**

You should submit the following to Canvas:

1. All C/C++/java source code required to build your program
2. The output files generated by running your program on the sample input files

**Notes**

The following additional comments apply:

1. Good programming style must be observed. This includes using meaningful variable names, descriptive comments, and readable formatting.
2. All your files (submitted and output files) must start with your login name + DT (i.e. YourLoginDT) including header files.
3. If you submit multiple files, you should either prepare a make file or a script file.
   * If you prepare a make file, you should name it as YourLoginDTMakefile.mk
   * If you prepare a script file, you should name it as YourLoginDTRun.sh Make sure that your file only processes your files. For example, “gcc \*.cpp” is not acceptable, since it is compiling all files.
   * Make sure that output executable file name is generated correctly (i.e., YourLoginDT)
4. The program source code file must contain a header comment at the beginning of the file. This comment must include the following items:
   * Source code file name, e.g., "YourLoginDT.cpp."
   * Student Name
   * Date
5. I will run your programs with gcc/c++/javac compiler under linux. So make sure that you do not use any additional libraries and your program is runnable under linux.
6. Your program will take inputs from command-line.

linux$ YourLoginDT –i inputFile –c classattribute -T M

(M (0<M<=100) is the percentage of samples to be used for training; class attribute is the target class for building the classifier).

* + Note: Java programs will be run with ‘java’ such as
  + java YourLoginDT –i inputFile –c classattribute -T M

1. Students should work independently. Each student is responsible for handing in an original program.