CSE 446: Machine Learning (Winter 2012)
Assignment 1: Decision tree learning

Due: **Sunday Jan. 22, 2012 at 11:59pm**

submit report and code online[1]

In this mini-project, you will implement a decision-tree algorithm. We have some data that describe the positions on 16 key votes for the U.S. House of Representatives Congressmen. Each vote can be valued as yea (y), nay (n) or unknown (?). **The task for you is to develop a decision tree algorithm and predict if the congressman is a democrat or a republican given his/her votes by learning from the data.**

Our data set, provided by UCI Machine Learning Repository, is called Congressional Voting Records Data Set[2]. It has 435 instances and 16 categorical features corresponding to the 16 key votes[3]. We randomly split the data set into the a training set (335 instances), a validation set (50 instances) and a test set (50 instances). The training and validation sets will be available with the release of the homework. The test set will be used for evaluating your implementation[4].

1. Download the data set[5] which contains the training and validation sets. Each congressman is represented by one line, with the 17 values separated by commas, the first of which indicates his/her party and the rest are the values of the 16 features.
2. Implement the ID3 decision tree learner as described in class and the reading material [2]. Another very good description of the algorithm can be found in the original ID3 paper [1]. You may program in Python or Java[6]. Your program should assume input in the above format. For initial debugging, it is recommended that you construct a very simple data set (e.g., based on a boolean formula) and test your program on it.
3. Implement both accuracy (misclassification impurity) and information gain (entropy impurity) for evaluation criterion.
4. To overcome overfitting, implement reduced error pruning on the decision tree. Use the validation set for pruning. For more details, please read Chapter 3.7.1 of the given reading material [2].
5. Use your algorithm to train a decision tree classifier and report accuracy on the test set. Compare accuracies by varying the evaluation criteria and whether reduced error pruning is applied.
6. Turn in the following:
   - Your source code. Your code should contain appropriate comments to facilitate understanding. Include a script[7] to compile, if necessary, and run your program. The script takes **five** arguments that are (in this order) the choice of evaluation criterion ("ACCURACY" or "IG" for information gain), the name of the training file, the name of the validation file, the name of the test file in the same format described above and the name of output file containing

[1]`https://catalyst.uw.edu/collectit/dropbox/xling/19368`
[2]`http://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`
[3]Treat the missing value "?" as a normal value for now.
[4]It will be available for downloading at 10:30am on Friday, January 20th.
[5]`www.cs.washington.edu/education/courses/cse446/12wi/ps1_data.tar.gz`
[6]If neither is your preferred choice, C/C++ will also be acceptable.
[7]named as "run.sh" for Linux/Mac users and "run.bat" for Windows users.

the predictions of your program. For example, we should be able to execute "`./run.sh IG train.data validation.data test.data test.output`" to train a decision tree using information gain criterion using "train.data", prune it using "validation.data", test on the file "test.data" and get results from "test.output". Each line of the output file has the predicted value ("democrat" or "republican") for its corresponding line in the test file.

- A report of at most 4 pages (letter size, 1 inch margins, 12pt font) that describes:
  - A high-level description on how your code works.
  - The accuracies you obtain under various settings.
  - Explain which options work well and why. If you implement something for extra credit, please describe how to test your additional features.
  - If all your accuracies are low, tell us what you have tried to improve the accuracies and what you suspect is failing.

7. **Extra credit:**

- *missing values*: Read Chapter 3.7.4 of the reading material [2] for a more proper way to handle missing values. Describe and discuss your implementation in the report.
- *chi-square pruning*: Another way of pruning the decision tree is chi-square pruning. Statistical significance test is used to decide if a node in the tree is irrelevant. For more details, please read page 662-663 of Russell & Norvig (2e) [3] or the original ID3 paper [1]. Compare with reduced error pruning in the report.

References

[1] J. R. Quinlan, *Induction of Decision Trees*, Machine Learning 1 (1) (1986): 81-106. `http://www.springerlink.com/content/ku63wm5513224245/?p=396a2aaabe0d4b008e3d4a7fe7ec8bad&pi=3`. UW IP address may be required.

[2] Tom Mitchell, *Machine Learning*, Chapter 3. `https://catalyst.uw.edu/sharespaces/download/15417/272436`

[3] Stuart J. Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach (2nd Edition)*, page 662-663