CSE6242 / CX4242: Data and Visual Analytics | Georgia Tech | Fall 2014

Homework 4: Decision Tree and Weka

Due: Monday, November 24, 2014, 11:59PM EST

Prepared by Brian Minsuk Kahng, Alan Zhang, Seungyeon Kim, Drew Wei, and Polo Chau

In this homework, you will implement decision trees and learn to use Weka. We expect you will spend about 10 hours on Task 1 and 4 hours on Task 2. You will submit a single archive file; see detailed submission instructions in the last section. All GT students must observe the honor code.

Task 1: Decision Trees (70 points)

In this task, you will implement a well-known decision tree classifier. The performance of the classifier will be evaluated by 10-fold cross validation on a provided dataset. Decision trees and cross validation were covered in class (slides).

You will implement a decision tree classifier from scratch using Python with the skeleton code we provide or other programming languages of your choice (e.g., Python, Java, C++). You may not use any existing machine learning or decision tree library.

Download the dataset here. It is a common dataset for evaluating classification algorithms[1], where the classification task is to determine whether a person makes over $50K. The dataset was extracted from the 1994 US census data. The data are stored in a tab-separated-value (TSV) file, where each line represents a person. For each line, there are 15 columns: the first 14 columns represent a person's characteristics (details), and the last column is a ground-truth label for his/her income (either greater or less than 50K). You must not use them as input features when you classifies the data.

A. Implementing Decision Tree (25 pt)

While implementing your decision tree, you will address the following challenges:

- Choosing the attribute for splitting (page 24-26 in the slides)
  - You can use entropy and information gain covered in the class.
- When to stop splitting (page 27)
  - Implementing advanced techniques, such as pruning or regularization (page 28), is completely optional.
  - You do not need to strictly follow the three conditions in page 27; you may use similar (or better) approaches.

[1] This data set is provided by UCI machine learning repository. We preprocessed the data set for the homework.

In your decision tree implementation, you may choose any variant of decision tree (e.g. entropy, Gini index, or other measures; binary split or multi-way split). However, you must explain your approaches and their effects on the classification performance in a text file ***description.txt***.

We provide a skeleton code here written in Python. It helps you setup the environment (loading the data and evaluating your model). You may choose to use this skeleton, write your own code from scratch in Python or other languages (e.g., Java, R, C++).

B. Evaluation using Cross Validation (15 pt)

You will evaluate your decision tree using 10-fold cross validation. Please see the lecture slides for details, but basically you will first make a split of the provided data into 10 parts. Then hold out 1 part as the test set and use the remaining 9 parts for training. Train your decision tree using the training set and use the trained decision tree to classify entries in the test set. Repeat this process for all 10 parts, so that each entry will be used as the test set exactly once. To get the final accuracy value, take the average of the 10 folds' accuracies.

With correct implementation of both parts (decision tree and cross validation), your classification accuracy should be around 0.8 or higher.

C. Improvements (25 pt)

Try to improve your decision tree algorithm. Some examples of strategies are:

- Different splitting strategies
- Different stopping criteria
- Pruning tree after splitting
- Use randomness and/or bagging

Your improvement can be simple; using as few as one or two simple heuristics or ideas is acceptable. The goal here is for you to try different ways to improve the classifier. You do not need to implement separate code for this improvement. It is okay to build on top of your initial implementation and only submit the best version of your decision trees. But you should discuss what you have tried and why you think it performs better in the *description.txt* file.

In the text file, you will also report the performance comparison between the implementation you have tried. For example,

- Initial (in section A): 0.80
- After applying strategy 1: 0.82
- After applying strategy 2: 0.83

Deliverables

1. **source code**: A source file(s) of your program. The source files should contain brief comments (what is this section for, such as calculating information gain). If you don't use the skeleton, please include a readme.txt explaining how to compile and run the code on the provided data. It should be runnable using a single command via terminal. Please do not compress the source files.

2. **description.txt**: This file should include:
   a. How you implemented the initial tree (Section A) and why you chose your approaches (< 50 words)
   b. Accuracy result of your initial implementation (with cross validation)
   c. Explain improvements that you made (Section C) and why you think it works better (or worse) (< 100 words)
   d. Accuracy results for your improvements (It will be graded on a curve. The full credit (10 pt) will be given to students with relatively high accuracy.)

You should implement your own code. You can reference some implementations on the Web, but you must not copy and paste those codes, which constitutes plagiarism.

Task 2: Using Weka (30 points)

You will apply Weka to classification of multi-class data. Download and install Weka. Note that Weka requires Java Runtime Environment (JRE) to run. We suggest you install the latest JRE, to avoid Java or runtime-related issues.

You will use the same data file from Task 1, and compare the performance of your implementation with Weka's.

How to use Weka:

- Load data into *Weka Explorer*: Weka supports file formats such as arff, csv, xls.
- Preprocessing: you can view your data, select attributes, and apply filters.
- Classify: under *Classifier* you can select the different classifiers that Weka offers. You can adjust the input parameters to many of the models by clicking on the text to the right of the *Choose* button in the Classifier section.

These are just some fundamentals of Weka usage. You can find many tutorials on how to use weka on the Internet.

A. Experiment (15 pt)

Run the following experiments. After each experiment, report your **parameters, running**

time, confusion matrix, prediction accuracy. An example is provided in the Deliverable part.

1. Decision Tree - C4.5 (J48) is commonly used and similar to what you implemented in Task 1. Under *classifiers -> trees*, select J48. For the Test options, choose **10-fold cross validation**, which should be same for A2 and A3. (5 pt)
2. Support Vector Machine - Under the *classifiers -> functions* select SMO. (5 pt)
3. Your choice -- choose any classifier you like from the numerous classifiers Weka provides. You can use package manager to install the ones you need. (5 pt)

B. Discussion (15 pt)

Discuss in your report and answer following questions:

1. Compare the Decision Tree result from A1 to your implementation in Task 1 and discuss possible reasons for the difference in performance. (< 50 words, 5 pt)
2. Describe the classifier you choose in A3: what it is, how it works, what are its strengths & weaknesses. (< 50 words, 5 pt)
3. Compare the 3 classification results in Section A: running time, accuracy, confusion matrices and explain why. If you change any of the parameters, briefly explain what you have changed and why they improve the prediction accuracy. (< 100 words, 5 pt)

Deliverables

report.txt - a text file containing the Weka result and your discussion for all questions above. For example:

Section A
1.

```
        J48 -C 0.25 -M 2
        Time taken to build model: 3.73 seconds
        Overall accuracy: 86.0675 %
        Confusion Matrix:
            a     b   <-- classified as
         33273  2079 |     a = <=50K
          4401  6757 |     b = >50K
```

2.
…

Section B
1. The result of Weka is 86.1% compared to my result <accuracy> because…
2. I choose <classifier> which is <algorithm>…
...

Submission Guidelines

Submit the deliverables as a single **zip** file named **hw4-*Lastname-Firstname.zip*** (should start with lowercase hw4). Please specify the name(s) of any students you have collaborated with on this assignment, using the text box on the T-Square submission page.

The directory structure of the zip file should be exactly as below (the unzipped file should look like this):

```
Task1/
     description.txt
     tree.py
     OR
     tree.xxx
     additional_source_files (if you have)
     readme.txt (unless you used tree.py)

Task2/
     report.txt
```

You must follow the naming convention specified above.