CARNEGIE MELLON UNIVERSITY

10601a: Homework #5 - “Decision Trees”

TAs-in-charge:
Krishna Aditya Gabbita (kgabbita@andrew)
Joe Runde (jrunde@andrew.cmu.ed)

Assigned: Wednesday, 11 February 2015.
Due: 11:59pm on Tuesday, 17 February 2015.
Late Penalty: 25% per day.

Office Hours:

	Krishna Aditya Gabbita GHC Citadel Commons	Joe Runde outside GHC 7110
Wed 2/11	-	9 - 10pm
Thurs 2/12	-	8 - 9pm
Fri 2/13	5-7pm	9 - 10pm
Sat 2/13	5-7pm	-
Sun 2/13	-	5 - 6pm
Mon 2/13	6-8pm	9 - 10pm
Tues 2/13	5-7pm	9 - 10pm

Policy on Collaboration among Students

Previously Used Assignments

Some of the homework assignments used in this class may have been used in prior versions of this class, or in classes at other institutions. Avoiding the use of heavily tested assignments will detract from the main purpose of these assignments, which is to reinforce the material and stimulate thinking. Because some of these assignments may have been used before, solutions to them may be (or may have been) available online, or from other people. **It is explicitly forbidden to use any such sources, or to consult people who have solved these problems before. It is explicitly forbidden to search for these problems or their solutions on the internet.** You must solve the homework assignments completely on your own. I will be actively monitoring your compliance, and any violation will be dealt with harshly. Collaboration with other students who are currently taking the class is allowed, but only under the conditions stated below.

The purpose of student collaboration is to facilitate learning, not to circumvent it. Studying the material in groups is strongly encouraged. It is also allowed to seek help from other students in understanding the material needed to solve a particular homework problem, provided no written notes are shared, or are taken at that time, and provided learning is facilitated, not circumvented. **The actual solution must be done by each student alone**, and the student should be ready to reproduce their solution upon request. In the case of programming assignments, **all code must be written by each student alone.** We will strictly enforce this policy. **The presence or absence of any form of help or collaboration, whether given or received, must be explicitly stated and disclosed in full by all involved.** Specifically, **each assignment must contain a file named Collaboration.txt where you will answer the following questions:**

- Did you receive any help whatsoever from anyone in solving this assignment? Yes / No. If you answered 'yes', give full details? (e.g."Jane explained to me what is asked in Question 3.4").
- Did you give any help whatsoever to anyone in solving this assignment? Yes / No. If you answered 'yes', give full details? (e.g. "I pointed Joe to section 2.3 to help him with Question 2").

If you gave help after turning in your own assignment and/or after answering the questions above, you must update your answers before the assignment?s deadline, if necessary by emailing the TA in charge of the assignment.

Collaboration without full disclosure will be handled severely, in compliance with CMU?s Policy on Cheating and Plagiarism.

Duty to Protect One?s Work

Students are responsible for pro-actively protecting their work from copying and misuse by other students. If a student?s work is copied by another student, the original author is also considered to be at fault and in gross violation of the course policies. It does not matter whether the author allowed the work to be copied or was merely negligent in preventing it from being copied. When overlapping work is submitted by different students, **both students will be punished.**

Severe Punishment of Violations of Course Policies

All violations (even the first one) of course policies will always be reported to the university authorities, will carry severe penalties, usually failure in the course, and can even lead to dismissal from the university. This is not an idle threat - it is my standard practice. You have been warned!

0 General Instructions

The goal of this assignment is for you to implement a decision tree learner, entirely from scratch. We're going to try decision trees in two domains. For simplicity, all variables are discretized into just two categories. The datasets for this assignment are available on autolab, go to Homework 5 and click on "Download Handout."

The first task is to predict whether a song was a "hit" meaning it made it onto the Billboard Top 50 –each instance has a label **hit** equal to "yes" or "no". Attribute names are listed in bold; check the csv file to see their possible values: **year** of release, **solo** recording or band, **vocal** or instrumental, **length** of recording (< 3 minutes or > 3 minutes), **original** composition or a "cover", **tempo**, **folk** song, **classical** piece, **rhythm** and blues, **jazz**, **rock** and roll.

The second task is to predict the final **grade** (A, not A) for high school students. The attributes (covariates, predictors) are student grades on 5 multiple choice assignments **M1** through **M5**, 4 programming assignments **P1** through **P4**, and the final exam **F**. Again, check the csv files to see the attribute values.

Before you begin, think about whether decision trees are appropriate for these two tasks. Write down your thoughts in a text file **Q0.txt**: do you think a decision tree will work for the music dataset? What about for the education dataset?

Next, inspect the training data manually; look for any unusual findings and think about which variables seem useful. In **Q1_music.txt** make your best guess about which variables are useful for the music task. Do the same in **Q1_education.txt**.

We've provided you with attributes and labels split into training and testing data in files "music*.csv" and "education*.csv". Throughout, we show results for "example1.csv" and "example2.csv," a small, purely for demonstration version of the music dataset. The format is comma separated, one row per observation, one column per attribute. Your program will take a training and a testing dataset as input.

1 Warmup

First, let's think a little bit about decision trees. Please answer the following questions by writing a single number in the .txt file:

Q2.txt (2 pts.) What is the maximum depth of a tree if the data has 12 binary attributes? (Please include only one number in your file, for this and the following problems)

Q3.txt (2 pts.) What is the maximum number of splits in a decision tree given 6 binary attributes? Assume only sensible trees where you cannot split on the same attribute twice in one branch.

Now on to programming, you may use Java or python to complete this assignment. Make sure to use the versions of python and Java supported by Autolab (currently python 2.6, Java 7), or your submission will not be graded correctly. Write a program `inspect.py` or `inspect.java` to calculate the label entropy at the root (i.e. the entropy of the labels before any splits) and the error rate (the percent of incorrectly classified instances) of classifying using a majority vote (picking the label with the most examples). You do not need to look at the values of any of the attributes to do these calculations, knowing the labels of each example is sufficient.

```
$ python inspect.py example1.csv
entropy: 0.981
error: 0.42

$ javac inspect.java
$ java inspect example1.csv
entropy: 0.981
```

```
error: 0.42
```

Test your program on both datasets –this error rate is a baseline over which we would (ideally) like to improve.

2 Training the Tree

Implement a decision tree learner with the following guidelines. (As a reference, consult Mitchell, Chapter 3, page 56.)

- Use mutual information to determine which attribute to split on
- Be sure you're correctly weighting your calculation of mutual information. For a split on attribute X, $I(Y;X) = H(Y) - H(Y|X) = H(Y) - P(X=0)H(Y|X=0) - P(X=1)H(Y|X=1)$. Equivalently, you can calculate $I(Y;X) = H(Y) + H(X) - H(Y,X)$.
- As a stopping rule, only split on an attribute if the mutual information is $\geq .1$.
- Do not grow the tree beyond depth 2. Namely, split a node only if the mutual information is $\geq .1$ and the node is the root or a direct child of the root.
- Use a majority vote of the labels at each leaf to make classification decisions

Hints on getting started: write helper functions to calculate entropy and mutual information. Write a function to train a stump (tree with only one level). The correct tree and output format for the example data are shown below, where we are training on example1.csv and testing on example2.csv. For the music data, use "+" for **hit** = "yes" and "-" for **hit** = "no". With the education data, use "+" for final **grade** = "A" and "-" for final **grade** = "not A". Don't worry about the order in which you list the left and right children, the autograder will take care of it. Your program should be named `decisionTree` and take two arguments, a training file, and a test file.

```
$ python decisionTree.py example1.csv example2.csv
or
$ javac decisionTree.java
$ java decisionTree example1.csv example2.csv

[58+/42-]
love = yes: [46+/11-]
| debut = yes: [29+/0-]
| debut = no: [17+/11-]
love = no: [12+/31-]
| debut = yes: [12+/11-]
| debut = no: [0+/20-]
error(train): 0.22
error(test): 0.48
```

Remember, the tree might not be full. Here's what happens when we train on example2.csv and test on example1.csv:

```
$ python decisionTree.py example2.csv example1.csv
or
$ javac decisionTree.java
$ java decisionTree example2.csv example1.csv

[28+/72-]
love = yes: [27+/25-]
| debut = yes: [26+/0-]
| debut = no: [1+/25-]
love = no: [1+/47-]
error(train): 0.02
error(test): 0.29
```

The numbers in brackets give the number of positive and negative labels from the training data in that part of the tree. The last two numbers are the error rate on the training data and the error rate on the testing data. Make sure your answers are accurate to within 0.01.

3 Evaluation

Train and test a decision tree for the music dataset and the education dataset. Which is more accurate on the training data? Which is more accurate on the testing data? Write down your observations in **Q4.txt**.

In addition to the music and education datasets, autolab will test your code on a third dataset, which will not be shown to you. This data contains information about various cars, and whether or not consumers decided to buy them. The data will be in .csv files similar to the ones provided, and is formatted as follows:

```
Class
Name : class
Values : yes/no

Attributes
Name : buying
Values : expensive/cheap

Name : maint
Values : high/low

Name : doors
Values : Two/MoreThanTwo

Name : person
Values : Two/MoreThanTwo
```

```
Name : boot
Values : small/large

Name : safety
Values : low/high
```

Please ensure your solution can handle data with these values.

4 Autolab Submission

Submit a .tgz file containing your source code, written assignments, and collaboration.txt. You can create this archive by running "*tar -cvf hw5.tgz *.py *.java *.txt*". **DO NOT** put the above files in a folder and then tar gzip the folder. You must submit this file to the "homework5" link on Autolab.