EECS 349 (Machine Learning) Homework 1

How to submit your homework

1. Create a **PDF document** containing answers to the homework questions. Repeat: a **PDF**.
2. Submit source code for the program you write.
3. Compress all of the files specified into a .zip file.
4. Name the file in the following manner, *firstname_lastname_hw1.zip*. For example, *Bryan_Pardo_hw1.zip*. **Your full name and the assignment name are important.** We have 100 people in the class and it avoids confusion.

3. Submit this .zip file via Canvas by the date specified on Canvas.

Problem 1 (2 points):

A typical machine learning task is supervised learning of a classification function (aka a concept): give example ***x*** the right class label ***l***. Assume there is an unknown target concept ***c*(*x*)**, whose domain is a set of unique examples ***X***, called the *instance space*, and whose range is a set of unique labels ***L***. The task is to learn that function. A learner does this by testing hypotheses. A hypothesis ***h*(*x*)** is also function whose domain is ***X*** and whose range is ***L***. If the set of hypotheses a learner is able to consider is the same as the set of possible concepts, then the set of unique hypotheses (the *hypothesis space* ***H***) is identical to the set of unique concepts (the *concept space* ***C***). For this question, assume ***H* = *C***.

In supervised learning, exploration of the hypothesis space is guided by the use of a finite data set, ***D*** that contains examples from ***X*** with known labels from the target concept ***c*(*x*)**. Since we can only measure success with respect to the data, we reframe the task of learning ***c*(*x*)** to that of finding a hypothesis consistent with the data: $\forall x \in D,\ c(x) = h(x)$

Definition: Two functions f_1 and f_2 are *distinguishable*, given ***D***, if they differ in their labeling of at least one of the examples in ***D***.

Definition: A set of hypotheses is distinguishable, given ***D***, iff ALL pairs of hypotheses in the set are distinguishable given ***D***. Call H_D a largest set of distinguishable hypotheses, given ***D***.

A) (1/2 point) Assume that ***X***, ***L*** and ***D*** are all finite and given (i.e. they are all fixed). Is there one unique H_D? Explain your reasoning.

B) (1/2 point) Let the size of the data and the label set be drawn from the counting numbers (i.e. $|D|,|L| \in \{1,2,3...\}$). Formulate the size of H_D, as a function of |***L***| and |***D***|. Explain your reasoning.

For parts C and D assume the following.
The label set ***L*** is {0,1}. The size of the data set |***D***| = 100. The size of the instance space |X| = 200.
Assume a learner able to consider a maximal set of distinguishable hypotheses H_D.

C) (1/2 point) Assume the learner is able to consider 10^9 (one billion) hypotheses per second. In the worst case, how long would it have to work to find a hypothesis ***h*** that is indistinguishable from the target function ***c***, given ***D***? Would this be a reasonable time to wait? Explain your reasoning. State your assumptions.

D) (1/2 point) Assume the learner HAS found a hypothesis ***h*** that is indistinguishable from the target concept ***c*** , given ***D***. What is the probability that hypothesis ***h*** is also indistinguishable from the target concept ***c*** , given X? Explain your reasoning. State your assumptions.

EECS 349 (Machine Learning) Homework 1

Problem 2 (1 point):

Given the following:

Instances ***X***: cars described by following attributes

x is a tuple: $< country, manufacturer, color, decade, type >$

$country \in \{Japan, USA, Korea, Germany\}$

$manufacturer \in \{Honda, Chrysler, Toyota\}$

$color \in \{blue, green, red, white\}$

$decade \in \{1970, 1980, 1990, 2000\}$

$type \in \{Economy, Sport, SUV\}$

The concept to learn, expressed in English is "Japanese Economy Car". The target concept function ***c(x)*** **that maps from *X* to** $\{0,1\}$ is illustrated by the data ***D*** in the following table.

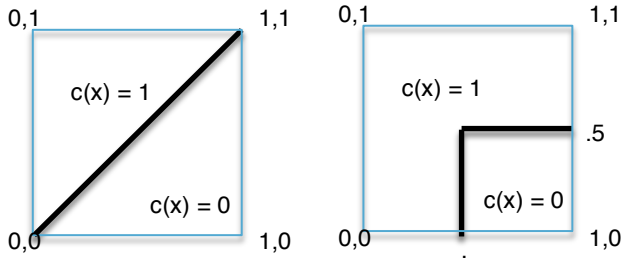
Origin	Manufacturer	Color	Decade	Type	Classification
Japan	Honda	Blue	1980	Economy	**1**
Japan	Toyota	Green	1970	Sports	**0**
Japan	Toyota	Blue	1990	Economy	**1**
USA	Chrysler	Red	1980	Economy	**0**
Japan	Honda	White	1980	Economy	**1**

A) (1/2 point) Calculate the decision tree that would be learned from the training examples in the "Japanese Economy Car" concept by running ID3 (see the Machine Learning textbook chapter 3). Show information gain for each attribute at each level of the tree.

B) (1/2 point) Write out the logical function encoded in the decision tree for "Japanese Economy Car." How well does this function categorize the data used to build the tree? How well do you feel this captures the concept of "Japanese Economy Car"? What about the approach or the data caused this result?

Problem 3 (1 point):

A) (1/2 point) Assume a binary labeling problem on some set of examples ***X***. ID3 assumes every attribute of an example takes a finite set of values and assigns a different branch to each value an attribute can take. This obviously can't work with attributes drawn from the set of real numbers. One easy modification is to pick a split point on the real values (i.e. if attribute ***a*** < 3, go left). Describe how you would pick a split point that makes sense. What are the strengths and weaknesses of your approach to picking a split point?



B) (1/2 point) Assume your examples are all 2-dimensional real-valued points on the unit square. Assume you can build a decision tree on real valued data by picking a split point on each dimension. Assume you can always pick the optimal split point. Which of the above concepts can be represented? One? Both? Neither? Explain your reasoning.

EECS 349 (Machine Learning) Homework 1

Problem 4 (2 point): Some basic definitions for measuring error

A) (1/2 point) Look up and define the following: True Positive, False Positive, True Negative, False Negative. Cite your source(s).

B) (1/2 point) Look up and define the following: Precision, Recall. Cite your source(s).

C) (1/2 point) Look up and define the F1 score (aka F-measure). Cite your source(s).

D) (1/2 point) Look up what a confusion matrix is. Explain why it is useful in analysis of machine learning. Cite your sources (give links & references). Give an example of why it is useful when a classifier has to apply more than two classes.

Problem 5 (4 points): Create a Decision Tree classifier that builds a tree using the ID3 algorithm described in the textbook and in class.

You are going to code up a decision tree, yourself. Recall that decision tree induction is a machine learning approach to approximating a target concept function *f*, given a set of examples. An *example* in our training data is a tuple $<x_1, x_2, \ldots, x_n, c(x_1, x_2, \ldots, x_n)>$ consisting of values for the *n* inputs to the function *c* and the output of *c*, given those values.

For this problem, you will construct a binary decision tree learner for examples with categorical attributes.

PROVIDED FILES

You have been provided with two example input files for the learner.

***IvyLeague.txt* –** a file containing examples of target concept "people who will be accepted to an Ivy League university"

***MajorityRule.txt* –** a file containing examples of target concept "over half of them are 'true'"

An input file is expected to be an ASCII file, where the first line is a tab-delimited list of *j* attribute names, followed by the word "CLASS". Each subsequent line consists of a fixed number of tab-delimited values. Each value must be the string **'true'** or the string **'false'** and there must be one value for each attribute in the first line of the file, plus an additional Boolean value (at the end of each line) that specifies how the target concept function would classify the example defined by these attribute values. The task for a machine learner is to learn how to categorize examples, using only the values specified for the attributes, so that the machine's categorization matches the categorization specified in the file. What follows is an example 3-line input file for a target concept 'People accepted to an Ivy League School'

```
GoodGrades GoodLetters GoodSAT IsRich Scholarship ParentAlum SchoolActivities CLASS
true  true  true  true  true  true  true  true

true  true  true  false  true  true  false false
```

YOUR PROGRAM

Your program must be written in Python 2.7 and run in the test environment without alteration. We are not responsible for debugging code written in other languages or other versions of Python. We are not responsible for linking to any 3rd party input/output libraries, except for SciPy.

The core decision tree algorithm code must be written by you. You cannot use an existing packages/modules/library that has implemented a decision tree. The point is to write your own decision tree. Your source code must be well commented so that can be easily understood.

Your folder must contain a file named "decisiontree.py" that can be called according to the following usage spec:

```
Usage: python decisiontree.py <inputFileName> <trainingSetSize> numberOfTrials> <verbose>
```

Note: You can read our input files with ease by using the module csv, which comes with Python 2.7. The items in the files are tab-delimited, with each new line beginning a new row. You could read each row as a dictionary, so that the element looks like, for example the first row could be read as follows:

EECS 349 (Machine Learning) Homework 1

```
{'GoodGrades': True, 'GoodLetters': True, 'GoodSAT': True, 'IsRich':
True, 'HasScholarship': True, 'ParentAlum': True, 'SchoolActivities':
True}
```

Note: Use string comparisons to change the string 'true' to the python boolean True. The format above will probably make it easier to write up the code.

What the Parameters Do

```
inputFileName   - the fully specified path to the input file. Note that windows
                  pathnames may require double backslash '\\'
trainingSetSize - an integer specifying the number of examples from the input
                  file that will be used to train the system
numberOfTrials  - an integer specifying how many times a decision tree will be built
                  from a randomly selected subset of the training examples.
verbose         - a string that must be either '1' or '0'
                  If verbose is '1' the output will include the training and test
                  sets. Else the output will only contain a description of the tree
                  structure and the results for the trials.
```

What the Program Must Do

1) Read in the specified text file containing the examples.
2) Divide the examples into a training set and a testing set. The training set should contain <trainingSetSize> many examples. Training set examples should chosen randomly, without replacement. Use the remaining examples for the testing set.
3) Estimate the expected prior probability of TRUE and FALSE classifications, based on the distribution of examples in the training set.
4) Construct a decision tree, based on the training set, using the approach described in Chapter 3 of Mitchell's Machine Learning.
5) Classify the examples in the testing set using the decision tree built in step 4.
6) Classify the examples in the testing set by selecting the most likely class, as determined by the prior probabilities calculated in step 3.
7) Determine the proportion of correct classifications made in steps 5 and 6 by comparing the classifications to the correct answers.
8) Steps 2 through 7 constitute a *trial*. Repeat steps 2 through 7 until the number of trials is equal to the value specified in the command-line input <number of trials>.
9) Print the results for each trial to the command line (standard output). The format of the output is specified in the following section.

OUTPUT FORMAT

Each run of your decision tree program should create an output on the command line that contains the following information:

- The input file name
- The training set size
- The testing set size
- The number of trials
- The mean classification performance of the decision tree on the testing sets
- The mean classification performance on the testing sets achieved by using the probability of "true," as derived from the training sets.

EECS 349 (Machine Learning) Homework 1

For each trial your program should output

- The number of the trial
- The proportion of correct classifications of the test set returned by the decision tree
- The proportion of correct classifications returned by applying prior probability (based on probabilities derived from the training set) to classify the test set
- The structure of the decision tree built from the training set

When VERBOSE = 1 you must provide the following information for each trial:

- The set of examples in the training set
- The set of examples in the testing set
- The classification returned by the decision tree for each member of the testing set
- The classification returned by applying prior probability to each member of the testing set.

We will not require that a particular layout be used. That said, if we have ANY problem finding the information specified above in your output file, you WILL lose points. It is up to you to make your output format CLEAR.

EECS 349 (Machine Learning) Homework 1

What follows is an example output for a non-verbose run with two trials in a format that would be completely acceptable.

```
TRIAL NUMBER: 0
--------------------

DECISION TREE STRUCTURE:
parent: root  attribute: IsRich  trueChild:GoodLetters  falseChild:HasScholarship
parent: IsRich  attribute: GoodLetters  trueChild:leaf  falseChild:GoodGrades
parent: GoodLetters -
parent: GoodLetters  attribute: GoodGrades  trueChild:leaf  falseChild:leaf
parent: GoodGrades -
parent: GoodGrades -
parent: IsRich  attribute: HasScholarship  trueChild:GoodLetters  falseChild:leaf
parent: HasScholarship  attribute: GoodLetters  trueChild:GoodSAT  falseChild:leaf
parent: GoodLetters  attribute: GoodSAT  trueChild:leaf  falseChild:leaf
parent: GoodSAT -
parent: GoodSAT -
parent: GoodLetters -
parent: HasScholarship -

       Percent of test cases correctly classified by a decision tree built with ID3 = 88%
       Percent of test cases correctly classified by using prior probabilities from the
training set = 45%

TRIAL NUMBER: 1
--------------------

DECISION TREE STRUCTURE:
parent: root  attribute: IsRich  trueChild:GoodSAT  falseChild:HasScholarship
parent: IsRich  attribute: GoodSAT  trueChild:leaf  falseChild:GoodGrades
parent: GoodSAT -
parent: GoodSAT  attribute: GoodGrades  trueChild:GoodLetters  falseChild:leaf
parent: GoodGrades  attribute: GoodLetters  trueChild:leaf  falseChild:SchoolActivities
parent: GoodLetters -
parent: GoodLetters  attribute: SchoolActivities  trueChild:leaf  falseChild:leaf
parent: SchoolActivities -
parent: SchoolActivities -
parent: GoodGrades -
parent: IsRich  attribute: HasScholarship  trueChild:leaf  falseChild:leaf
parent: HasScholarship -
parent: HasScholarship -

       Percent of test cases correctly classified by a decision tree built with ID3 = 76%
       Percent of test cases correctly classified by using prior probabilities from the
training set = 43%

example file used = IvyLeague.txt
number of trials = 2
training set size for each trial = 20
testing set size for each trial = 42
mean performance of decision tree over all trials = 82% correct classification
mean performance of using prior probability derived from the training set = 44% correct
classification
```