CSE 231 Spring 2008

Programming Project 08

This assignment is worth 50 points (5.0% of the course grade) and must be **completed and turned in before 11:59 on Monday, Oct 27, 2008.**

Assignment Overview
This assignment will give you more experience on the use of:
1. Functions
2. Dictionaries and Lists

The goal of this project is to gain more practice with using functions and dictionaries. This project offers a lot of opportunity to modularize your code into functions. In general, any time you find yourself copying and pasting your code, you should probably place the copied code into a separate function and then call that function.

Problem Statement
Given a data file containing hundreds of records with values describing cancer tumors and whether or not each tumor is malignant or benign, develop a simple rule-based 'classifier' that can be used to predict the class (malignant or benign) of a set of unknown records.

Here is the general idea: Malignant tumors are different than benign tumors. Malignant tumors tend to have larger radii, to be more smooth, to be more symmetric, etc. Measurements have been taken on many tumors whose status (malignant or benign) is known. The code you are going to write will get the average score across all the malignant tumors for an attribute (e.g. 'area') as well as the average score for that attribute for benign tumors. Let's say that the average area for malignant tumors is 100, and it is 50 for benign tumors. We can then use that information to try to predict whether a given tumor is malignant or benign. Imagine I presented you with a new tumor and told you the area was 99. All else being equal, we would have reason to think this tumor is more likely to be malignant than had its area been 51. While real classifiers use more complicated rules, we are going to create a very simple classification scheme. We will calculate the mid-point between the malignant average and the benign average (75 in our hypothetical example), and simply say that for each new tumor, if its value for that attribute is greater than or equal to the midpoint value for that attribute, that is one vote for the tumor being malignant. Each attribute that we are using produces a vote, and at the end of counting votes for each attribute, if the malignant votes are greater than or equal to the benign votes, we predict that the tumor is malignant.

To 'learn' what the midpoint values are we need access to attribute values of many tumors whose status is known. You can find these in the training set file. Once we have the classifier (which is the midpoint values), we will then predict the status of the tumors in the test data file. It turns out that we also know the status of the tumors in the test data file. As such, a final step will be to compare the prediction your classifier makes to the real value, to see how accurate your classifier is.

Background
Making predictions is hard and for as long as we've had computers, we've used them to try and make better predictions. In this project, we'll be writing a small program to predict whether or not a cancer tumor is malignant or benign when given the mean for each of 10 attributes describing the tumor.

Each tumor is described by 3 statistics (mean, standard error, min/worst) for 10 values, we will only worry about the mean values for this project. Each tumor also has data for the identification number and the class (malignant or benign). That makes 32 total values in the file per tumor. Look at the top of the data files (e.g. cancerTraining.txt) to see all 32 values.

Here are the 10 tumor attributes:

1. radius
2. texture
3. perimeter
4. area
5. smoothness
6. compactness
7. concavity
8. concave
9. symmetry
10. fractal

The mean, standard error, and minimum values are described with the suffixes '_mean', '_se', and '_worst', respectively, in the data files. For the class label, the M stands for malignant and the B for Benign.

We'll use these attributes to predict the value of the class (malignant or bening) for a new tumor. Note the ID number has no bearing on the tumor's class and is used only to differentiate tumors. A single set of attributes for a single tumor is known as a **record**.

We don't need to know what any of the attributes' names mean (what does 'fractal' mean in this context?). All we need to know is that they are measurements of the tumors, and that benign and malignant tumors tend to have different attribute values. For these 10 attributes when comparing means, **higher numbers indicate malignancy**.

There are six tasks associated with this project.

1. Create a training set.
A training set is a set of records where we know the value of the class. We'll use the training set in the next step to build up a model that can be used to make predictions. This step is done for you.

2. Train a simple classifier.
A classifier is a model of the problem such that when we're given a new record we can compare the new record to the model in order to predict the class of the new record. We use the training set to build up this model. Our model is very simple.

For all malignant records, for each attribute, we calculate the average value of each attribute. For all benign records, for each attribute, we calculate the average value of each attribute.

To create the model, we then calculate the midpoint of these averages for each attribute. Then to classify new records, if the majority (5 or more) of the new record's attributes are above their respective midpoints, then the new record is predicted to be malignant, otherwise (4 or less), benign.

There are many different methods in the areas of Artificial Intelligence and Machine Learning that have been used by computer programmers to make predictions. Most of these methods rely heavily on statistics-based methods that use computers to crunch a lot of numbers. We're more interested in developing our programming skills than delving deep into statistics so we are going to use a very simple method to make predictions. That is to say, our classifier is probably not statistically sound but it serves as a good programming exercise as well as a good introduction to the problem of predicting classes.

Furthermore, in the real world, we commonly face lots of issues that crop up with missing data, noisy data, or other problems. We don't face any of these issues in this assignment. It is safe to assume that all of the data is there and correct.

3. Create a test set.
A test set is a set of records where (1) the records were not used to train the classifier and (2) we know the value of the class.

Since the classifier hasn't seen these records yet but we know the true class, we can test out the classifier's accuracy on these new records. The classifier will predict the class of the record which we can then compare to the actual class.

4. Apply the classifier to test set.
For each record in the test set, if the majority (5 or more) of the new record's attributes are above their respective midpoints, then the new record is predicted to be malignant, otherwise (4 or less), benign.

5. Report accuracy of classifier.
For each record in the test set, compare the predicted class to the actual class and then print out the accuracy of the classifier both as number correct / total number and as a percentage.

6. Output Results
You should provide two other kinds of output besides the accuracy.

First, report the statistics you gathered (the malignant and benign averages, as well as the classifier midpoint value). Second, you should provide some feedback on an individual patient. The system should prompt for a patient ID from the test set, and then provide the patient values from the test set, the classifier cutoff for that value and the diagnosis for that particular value, as well as the patient's overall predicted diagnosis (see example output).

Project Description / Specification

Most importantly, ***demo output is provided below***. You can look at the example output to see how the program should run on the provided data files. **Make your output look exactly like the output in the demo!**

A starting program is provided. Code your solution inside this file. Do not start from scratch. Finish implementing project08Start.py to complete the six tasks listed above.

Functions have been provided for you to create both the training and test sets. You'll need you to finish the functions to train the classifier, apply the classifier to the test set, and report the accuracy of the classifier and output the results. Stub functions have been started for each of these tasks.

In addition to completing the three tasks, **you must implement at least two non-trivial 'helper' functions** that will be used to help complete the above tasks. (Hint: this should be especially useful when training the classifier to do something that's repeated several times such as initializing a dictionary for attributes 2-9 or for calculating an average.) By non-trivial function, we mean that the helper functions should complete some meaningful task that requires passing of parameters and the use of return values. (Your helper functions should not just print one line out.)

Implementation Checklist:
1. Finish implementing the function "trainClassifier"
2. Implement your first helper function
3. Implement your second helper function
4. Finish implementing the function "classifyTestRecords"
5. Finish implementing the function "reportAccuracy"
6. Finish implementing "dumpStats"
7. Finish implementing "checkSomePatients"

Deliverables
Turn in project08.py files, using the handin program. Save a copy to your H drive.

List of Files to Download
project08Start.py
-cancerTestingData.txt
-cancerTrainingData.txt

Notes and Hints:
The provided code is marked with 'TODO' where you need to modify the provided program.

The provided code has a lot of comments describing pseudo-code that should help you implement each function.

Start by familiarizing yourself with code and comments that are provided. It should run without any errors. (Don't be confused by the print statements, the provided program only creates the training and test sets – the provided program doesn't actually do anything with them other than read in the two files and store their contents in some data structures.)

The tasks have to be completed in order. You obviously can't use a classifier before you've trained it.

Don't try to tackle this project all at once. Complete one function (or part of a function) and test it out.
Continuously save, run, and test your code as you're working. The provided program doesn't have any code for testing whether or not a function is correct. If you're working on a function it would be helpful to add print statements so you can watch what your program is doing. You can comment these out later once you're sure the code works.

Extra credit:
You are eligible to receive extra credit if you turn your project in by (and do not turn anything else in after) Friday at midnight. If you are eligible, email your TA with the answers to the following questions, plus the code in which you use all 3 types of data for each attribute (mean, SE, and worst) to create your classifier. DO NOT turn this code or anything else in after Friday at midnight or you will not be eligible for the extra credit.

What is the accuracy of the classifier if you

- Use the mean values, plus the standard error and worst attributes (30 total)?
    - This means you will need to calculate midpoints for the SE's and worsts for the training set and include these in your voting on the test set, so now 15 or more values greater than the midpoints will predict malignancy, 14 or less will predict a bening tumor.
- Use only the mean values and the worst values?
- Use only the mean values and the standard error values?
- Use only the worst values?
- Use only the standard error values?

Including the approach from the main project (where you used only the mean values), which classifier is the most accurate?

Example Output

Note: user input is in bold (just the patients' ids to check and 'quit' to quit), everything else is output by the program.

```
Classifier, benign and malignant stats
======================================================================
                       Key    Malignant   Classifier       Benign
                                Average     Midpoint      Average
                    radius       17.075       14.545       12.016
                   texture       21.385       19.279       17.174
                 perimeter      112.687       94.919       77.152
                      area      934.017      693.338      452.659
                smoothness        0.103        0.098        0.093
               compactness        0.144        0.110        0.077
                 concavity        0.153        0.100        0.046
                   concave        0.084        0.055        0.025
                  symmetry        0.194        0.185        0.175
                   fractal        0.063        0.063        0.063

Reading in test data...
Done reading test data.

Classifying records...
Done classifying.

The classifier correctly predicted the class (malignant/benign) of 213
records out of 231 records.
The classifier achieved an accuracy of 92.21 percent.

Type an ID to check a patient ('quit' to stop):897880
Checking ID:897880's classification
                       Key      Patient   Classifier        Class
                                  Value       Cutoff
                 perimeter       64.410       94.919       Benign
                  symmetry        0.189        0.185    Malignant
                      area      310.800      693.338       Benign
                   concave        0.018        0.055       Benign
                   texture       17.530       19.279       Benign
                 concavity        0.025        0.100       Benign
                    radius       10.050       14.545       Benign
               compactness        0.073        0.110       Benign
                   fractal        0.063        0.063    Malignant
                smoothness        0.101        0.098    Malignant

Overall Diagnosis for patient 897880: Benign

Type an ID to check a patient ('quit' to stop):89812
Checking ID:89812's classification
                       Key      Patient   Classifier        Class
                                  Value       Cutoff
                 perimeter      155.100       94.919    Malignant
```

```
                    symmetry       0.180        0.185       Benign
                        area    1747.000      693.338    Malignant
                     concave       0.141        0.055    Malignant
                     texture      24.270       19.279    Malignant
                   concavity       0.231        0.100    Malignant
                      radius      23.510       14.545    Malignant
                 compactness       0.128        0.110    Malignant
                     fractal       0.055        0.063       Benign
                  smoothness       0.107        0.098    Malignant

Overall Diagnosis for patient 89812: Malignant

Type an ID to check a patient ('quit' to stop):quit
Program finished.
```