**CSI3120: Programming Language Concepts**

**Fall 2014**

**Assignment 4**

**Due Date: December 2, 2014 Due Time: 11:59pm**

## Question 1: Visual Analysis in R [25 points]

Consider the Iris Data Set which is included in the basic R library. [Just type "iris" once you've loaded R]

Basically, there are 3 species of Irises (Setosa, Versicolor, Virginica) and each iris is described by 4 characteristics (Sepal.Length, Sepal.Width, Petal.Length, Petal.Width).

The purpose of this question is to analyze this data set visually. In particular, I will ask you to:

- Draw a pie chart indicating the proportion of irises of each species. Print out the actual proportion of each species on the graph.
- For each characteristic, draw a scatter plot of all the data points, using a different colour for each species [blue for Setosa, red for Versicolor and green for Virginica]. [There should be 4 scatter plots]
- For each species of Iris, plot the average value of each of the 4 characteristics in a barplot graph. [There should be 3 barplot graphs]. Draw a density like across each barplot graph to visualize the distribution.
- Treat the problem as a multivariate problem, creating a single barplot graph (with 4 clusters of 3 bars) showing the same information as the previous 3 barplot graphs.
- Draw four boxplot graphs, each representing a different characteristic, and each containing three boxes per graph (corresponding to the three iris species).

[Note: all your graphs have to be properly labeled and have a proper title and legend]

Discuss what you can learn about the three different species of Irises (a fascinating topic, indeed!) from your graphs. Which graphs were the most useful for your analysis. Can you come up with a sure way to distinguish one species from another based on their characteristics?

## Question 2: Perceptron Learning [45 points]

In Assignment 1, you were asked to program a decision tree in Scheme. You used that decision tree to assign instances into their most likely class. This is called classification and represents an important part of the field of data mining.

In this question, the problem remains one of classification, but I am asking you to use a different technology and a different implementation language. The technology is that of a single-layered perceptron (a simple type of neural networks) and the language is R.

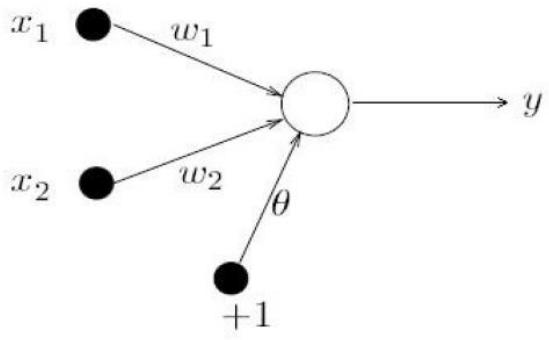
In this problem, you will design and train a perceptron on three domains (three versions of the Iris domain reduced to two classes: setosa vs versicolor; setosa versus virginica; and versicolor versus virginica), and for each domain, you will output the percent accuracy (i.e., 100* number of correct classifications / total number of instances) of your trained perceptron. Your perceptron will have exactly 4 input units. There is no need to generalize it to an unspecified number of units.

Here is a description of the single-layered perceptron with 2 input units [reference:

http://aass.oru.se/~lilien/ml/seminars/2007_02_01b-Janecek-Perceptron.pdf

(see pp. 9-25 for this and more information)]

A 2-input single-layered perceptron can be represented as follows:




where x1 and x2 are inputs to the perceptron (in the Iris domain, there will be 4 rather than 2 such inputs), w1, w2 are the perceptron's weights (there will also be 4 weights in the Iris domain), Θ is the bias, and y is the output. x1, x2, w1, w2, and Θ are real values, y can take the values 1 or -1. w1, w2 and Θ are initially assigned a random number. As the perceptron is "trained", the values of w1, w2, and Θ become meaningful (to the perceptron… To you they may not be).

Throughout the training of the perceptron and once it is trained, y can be calculated as follows (eq.1):

$$y = \operatorname{sgn}\left(\sum_{i=1}^{2} w_i x_i + \theta\right)$$

$$\operatorname{sgn}(s) = \begin{cases} 1 & \text{if } s > 0 \\ -1 & \text{otherwise.} \end{cases}$$

The idea of the training algorithm is the following:

Initialisation and generalities:

1. There is a training set. For example, in the setosa vs versicolor domain, you can set the setosa class as 1 and the versicolor class as -1 (i.e., y will be 1 when the values of x1, x2, x3, and x4 correspond to those of a setosa; and -1 when the values of x1, x2, x3, and x4 correspond to those of a versicolor). All the instances of setosa and versicolor irises will be included in the training set. The instances of virginica will not be.
2. During training, all of the w1, w2, w3, w4 and Θ will be modified.
3. In order to make your program easier, define w0 to be Θ and set x0 to 1.
4. Let η be the learning rate. This is a constant that you assign yourself. η must be a small positive number. You can try 0.1 or 0.2.
5. w0, w1, w2, w3 and w4 will be assigned to a random value at the beginning of the program.

Let (eq. 2)

$$\textit{Desired output} \qquad d(n) = \begin{cases} +1 & \textit{if } x(n) \in \textit{set } A \\ -1 & \textit{if } x(n) \in \textit{set } B \end{cases}$$

Perceptron Learning:

For epochs 1 to N {

- Randomize the order in which the instances are stored in your table
- For each instance in the table {
  - Select the next instance. This instance will be used as input.

- Compute y for this instance according to eq.1.
- If the classification is correct, do nothing
- If the classification is incorrect, modify the weight vector w using eq.3 below (which makes use of eq.2, above)

eq.3:

$$w_i = w_i + \eta d(n) x_i(n)$$

}

}

- Please note that N is a constant that you set representing the maximum number of epochs for which the program will run.
- How to set N? It is important to know that in the Iris data, one class is linearly separable from the other two; but the other two are not linearly separable from each other (linearly separable means that you can separate all the instances of one class from all the instances of the other by a single line (if you have only 2 inputs) or by a plane or a "hyperplane" (the equivalent of a line in higher dimensional spaces) if you have more than 2 dimensions.). When a data set is linearly separable, the perceptron is guaranteed to converge (i.e., to obtain a 100% accuracy rate) as long as it is run for a sufficiently long number of epochs. You should, therefore, make sure that you set N to a high enough number to reach 100% accuracy on the linearly separable Iris domains. In the non linearly-separable domains, you will never reach a 100% accuracy with a perceptron since it can only learn a line, plane or hyperplane.

**Question 3: Subprograms [10 points]**

Consider the following program written in C syntax:

```
void fun (int first, int second) {
     first += first;
     second += second;
}
void main( ) {
    int list[2] = {1, 3};
     fun(list[0],list[1]);}
```

For each of the following parameter-passing methods, what are the values of the 'list' array after execution?

1. Passed by value
2. Passed by reference
3. Passed by value-result

**Question 4: Scope [10 points]**

Consider the following C program:

```
void fun(void) {
    int a, b, c; /* definition 1 */
     …
    while (…) {
       int b, c, d;  /* definition 2 */
          …          <----------------------- 1
          while (…) {
             int c, d, e;  /* definition 3 */
             …       <------------------------ 2
             }
          …          <------------------------- 3
        }
      …             <------------------------- 4
   }
```

For each of the four marked points in this function, list each visible variable, along with the number of the definition statement that defines it.

**Question 5: Data Types [10 points]**

Multi-dimensional arrays can be stored in row major order, as in C++, or in column-major order, as in Fortran. Develop the access functions for both of these arrangements for three-dimensional arrays.