Exercise Sheet #10

Problem 1 (*Forking*) 5 Pts

In the lecture, forking was introduced as one approach to parallel processing in C++. A fork generates a full copy of the program and independently continues the execution of the program in both processes after the `fork()` command. This means that simply calling a fork command N times will lead to 2^N independent processes. However, instead of this nested forking, it would be more useful to be able to split an arbitrary number of child processes from a single parent process. Your task is to find a way to use `fork()` such that the program behaves in this way. As a *hint*, you should make use of the fact that `fork()` has a return value: it returns a negative value if the forking failed, zero if the value was returned in the child process and a positive integer if the value was returned in the parent process. To check if your approach worked, check the process IDs using `getpid()` (own process id) and `getppid()` (parent process id).

Problem 2 (*Perceptron training*) 15 Pts

The perceptron is a basic algorithm capable of performing linear classification. It takes the product of an input vector with a fixed vector of weights and adds a bias to create a single number, the activation, which is used to predict the label of the input. Implement a training function and teach a perceptron model to tell the difference between sonar readings of rocks and mines.

1. Create a prediction function that receives an input vector $\boldsymbol{x}$ and a weights vector $\boldsymbol{w}$ and calculates the activation y':

$$y' = \boldsymbol{w} \cdot \boldsymbol{x} + b \tag{1}$$

   The bias b can be stored in the weights vector. Return the activation after applying a step function to it: `1 if y>=0, else 0`.

2. Create a training function that receives a dataset and trains the perceptron to predict its labels. Start with all weights set to 0 and calculate the prediction error of each datapoint, $l = (y - y')^2$. The true label y should be the last element of each data vector. Use this error to update the weights after every datapoint with the update rule:

$$b \to b + \eta \cdot l \tag{2}$$

$$\boldsymbol{w} \to \boldsymbol{w} + \eta \cdot l \cdot \boldsymbol{x} \tag{3}$$

with the learning rate $\eta = 0.01$. Run through the whole dataset and print out the average prediction error.

3. Test the power of your algorithm by learning on real data. Download the dataset `sonar.all-data` from this link (click on 'Data Folder'). Load the dataset and change all R labels to 0 and M labels to 1. Notice that the data is ordered, shuffle the order randomly to prevent overfitting. Call the training function and train your model for 30 steps (epochs) over the entire dataset. Can you see an improvement in classification error?