Data Structures and Algorithms: Homework #4

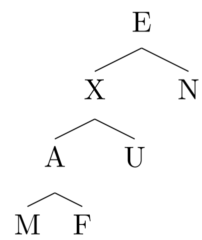
Due on May 15, 2015 at 16:20

Instructors Hsuan-Tien Lin, Roger Jang

Tim Liou (b03902028)

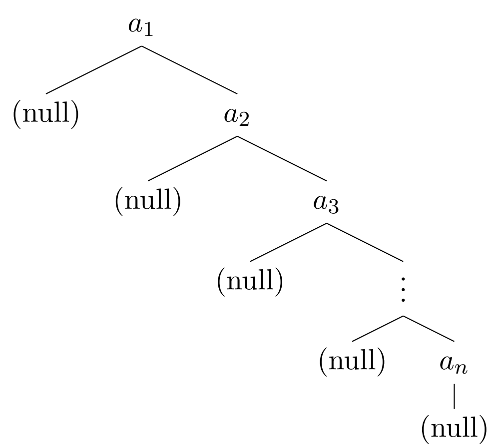
4.1 Tree

(1) Do Exercise R-7.15 of the textbook.



(2) Do Exercise R-7.24 of the textbook.

(a) We can easily find the level of right child is larger than its parent and its sibling. It implies that it will obtain the largest level if all the nodes (except the root) is its parent's right child.



Note that $f(a_n) = 2f(a_{n-1}) + 1$, and we can find

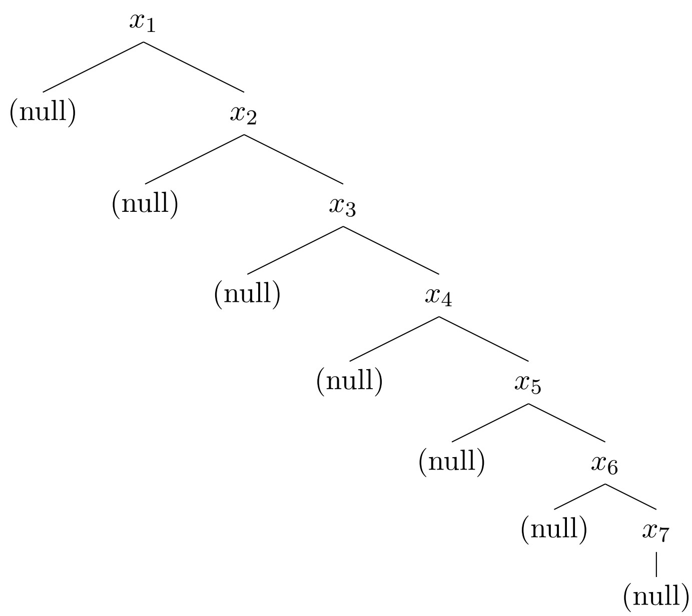
$$\begin{aligned}
&\Rightarrow f(a_n) + 1 = 2(f(a_{n-1}) + 1) \\
&\Rightarrow f(a_n) + 1 = 2^2(f(a_{n-2}) + 1) = \cdots = 2^{n-1}(f(a_1) + 1) \\
&\Rightarrow f(a_n) + 1 = 2^n \\
&\Rightarrow f(a_n) = 2^n - 1
\end{aligned}$$

We find the upper bound of level for a binary tree with n nodes. That is, for every node v of T,

$$f(v) \leq 2^n - 1$$

Problem 4.1 continued on next page...

(b)



There are 7 nodes in this tree. The upper bound of the level is $2^7 - 1 = 127$. We can easily find $f(x_7) = 127$ by the formula in (a). The node x_7 attains the above upper bound on f in this tree.

(3) Do Exercise C-7.3 of the textbook.

$post(v) = pre(v) - depth(v) + desc(v)$

For the root, with post-order, we will visit its descendents first, then visit it, but with pre-order, we will visit it first, then visit its descendents. And for other non-root node, it also satifies the statement I just mentioned. However the formula need to be fix by $depth(v)$. Note that $depth(root) = 0$. Eventually, we have the formula written above.

(4) Do Exercise C-7.7 of the textbook with pseudo code.

function PRINT(*root, NumOfIndent*)
  if root has children **then**
    print out the content in root and ’(’ with NumOfIndent indent.
    for all $i \leftarrow$ children **do**
      PRINT(*i, NumOfIndent + 1*)
    end for
    print out ’)’ with NumOfIndent indent
  else
    print out the content in root with NumOfIndent indent.
  end if
  return
end function

Algorithm 1: Print a tree with indented parenthetic representation

Call this function by **Print**$(root, 0)$.

4.2 Decision Tree

(1) Using the property that the v_m values are sorted, describe an $O(M)$ algorithm to calculate the best threshold.

for all $i \leftarrow$ data **do**
  if The value of this feature is bigger than $threshold_j$ **then**
    initial a_{j+1}Yb_{j+1}N with the value in a_jYb_jN
    $j \leftarrow j + 1$
  else
    Update a_jYb_jN.
  end if
end for
for all $i \leftarrow$ thresholds **do**
  calculate the confusion by using a_iYb_iN and $(TotalY - a_i)$Y$(TotalN - b_i)$N.
  check if this confusion is smaller than previous.
end for

Algorithm 2: Calculate the best threshold with an $O(M)$ algorithm

The purpose of first **for loop** is to calcuate and store how many Yes and No below each threshold. We will know the values of TotalY and TotalN because we can get them when we encounter the biggest thresold $v_M + 1$.

(2) Implement the decision tree algorithm.

Code part.

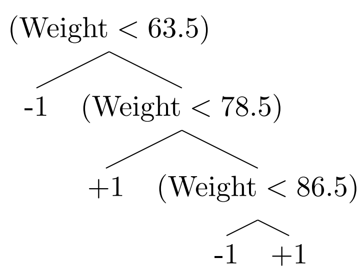
(3) Illustrate (ideally with drawing) the internal data structure you use to represent the decision tree in your memory. Please be as precise as possible.

I use this data structure to represent the decision tree.

```
struct BranchChoice {
    int BestFeature;
    double BestTotalConfusion;
    double BestThresholdID;
    double BestThreshold;
};
struct DTree {
    BranchChoice Choice;
    DTree *left;
    DTree *right;
};
```

(4) Teach your decision tree with the following examples to learn a function f with $\epsilon = 0$. Draw the tree you get.

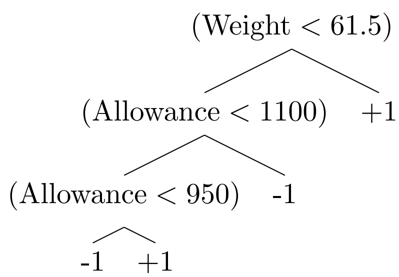
Left side means Yes, and right side means No.



(5) Construct your own data set with at least 2 numerical factors and at least 6 examples. Teach your program to make a decision tree of at least 2 levels with this data set. List the examples as well as draw the tree found. Briefly explain the tree.

Allowances per week	Weight	Height	Eat Dinner Every Day
1000	53	172	+1
2000	90	178	+1
1500	83	162	+1
1200	45	164	-1
900	53	169	-1
10000	70	172	+1
1300	77	185	+1
800	74	170	-1
11000	40	165	-1

Left side means Yes, and right side means No.



We can find whether a person eat dinner every day is related to his or her weight and allowance by this tree. It seem that the result isn't really related to his or her height. However, this is a small data set, this decision tree may be meaningless.

(6) Implement the random forest algorithm.

Code part.