

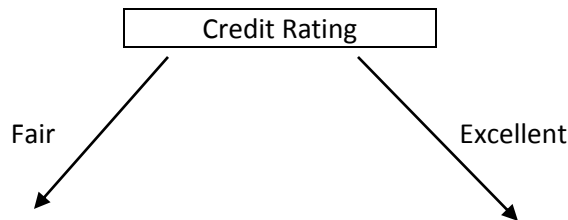
David Hayman HW2: Part 1 –decision tree and explanation on last two pages

Training data:

Age	Income	Student	Credit Rating	Buys Computer
<=30	High	No	Fair	No
<=30	High	No	Excellent	No
31...40	High	No	Fair	Yes
>40	Medium	No	Fair	Yes
>40	Low	Yes	Fair	Yes
>40	Low	Yes	Excellent	No
31...40	Low	Yes	Excellent	Yes
<=30	Medium	No	Fair	No
<=30	Low	Yes	Fair	Yes
>40	Medium	Yes	Fair	Yes
<=30	Medium	Yes	Excellent	Yes
31...40	Medium	No	Excellent	Yes
31...40	High	Yes	Fair	Yes
>40	Medium	No	Excellent	No

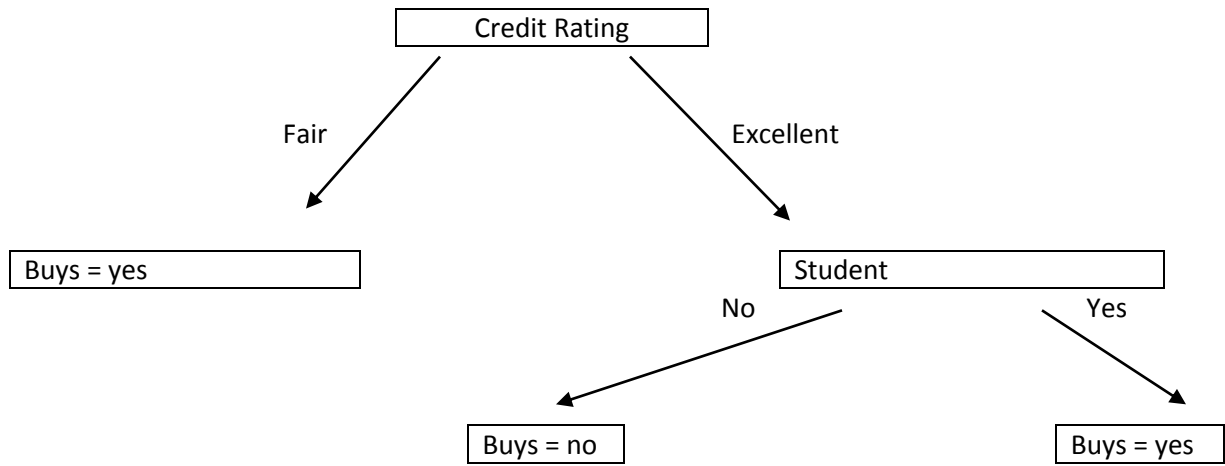
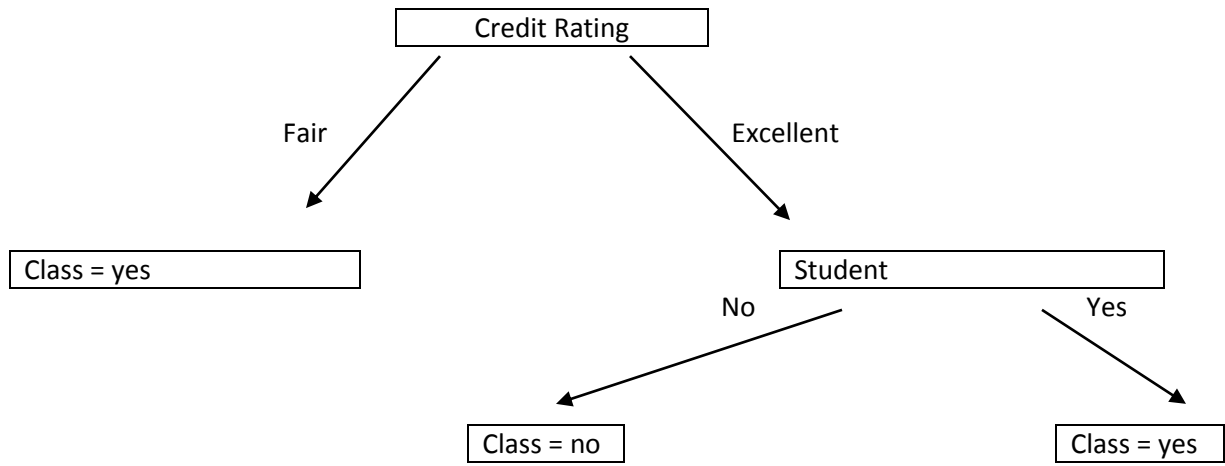
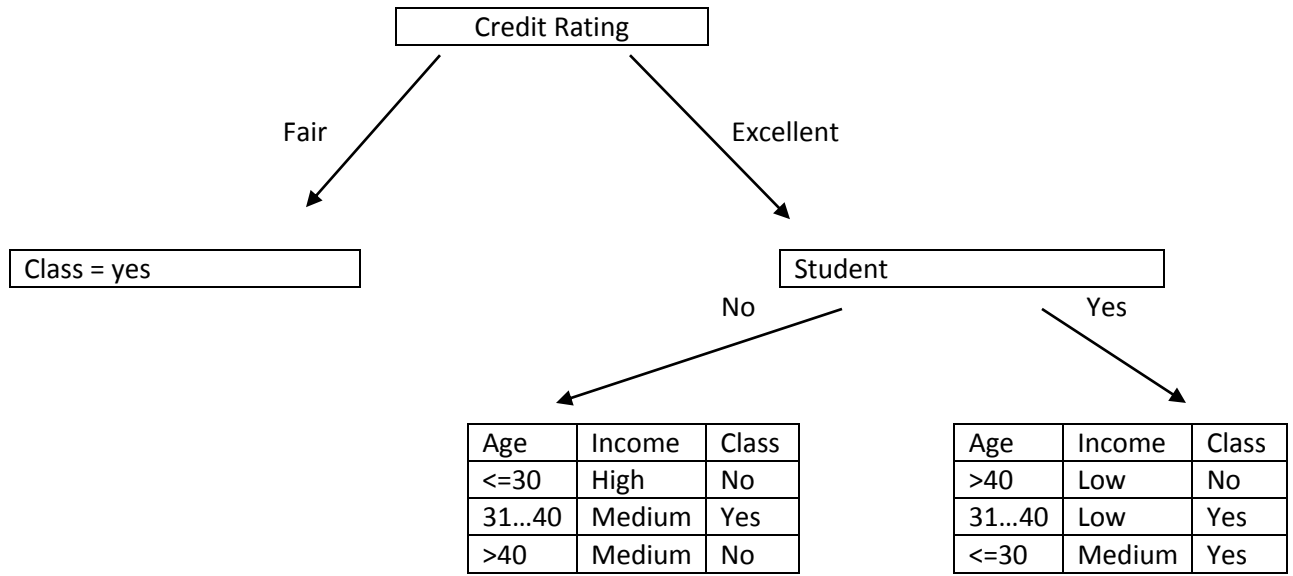
Problem 1: 1: “one with **general majority** voting , as defined in lecture notes, i.e. majority voting at any node of your choice.

Use CREDIT RATING as the root attribute, and nodes attributes of your choice”

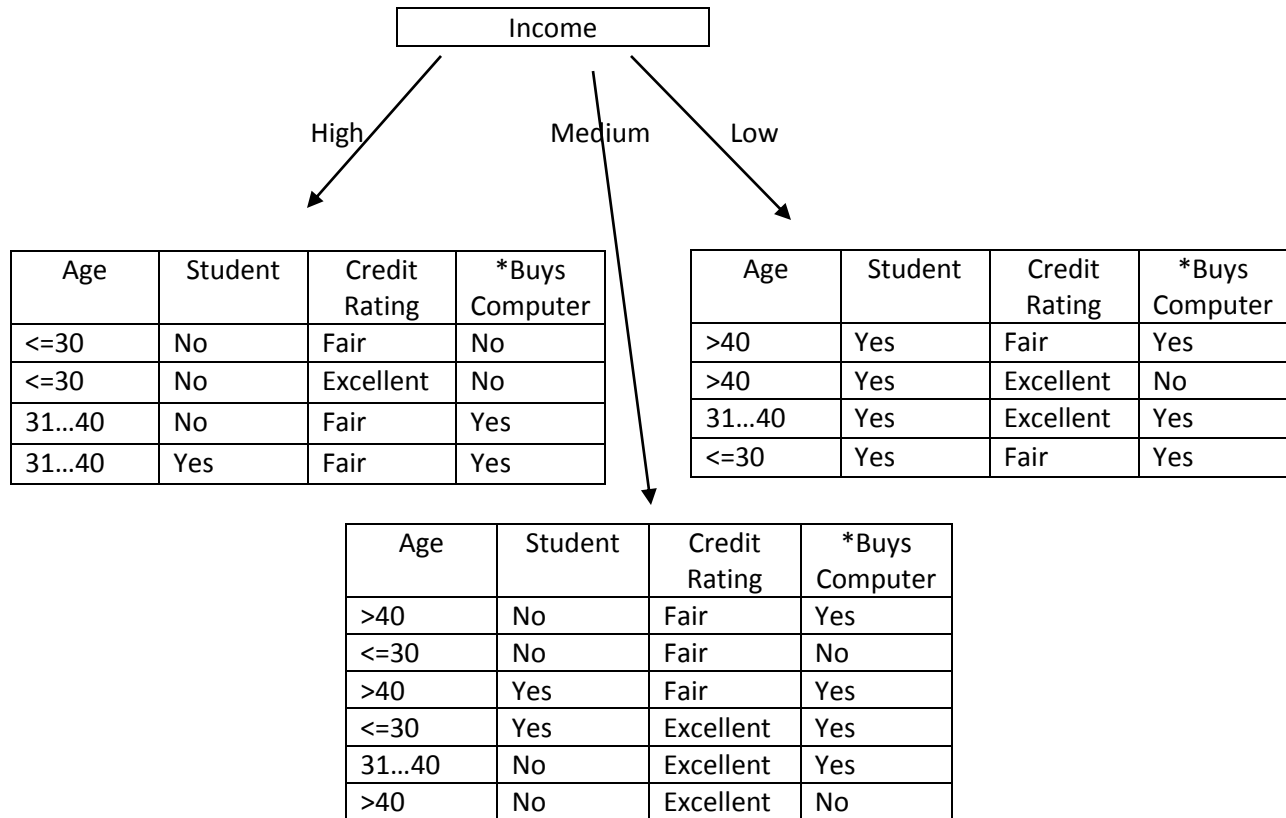


Age	Income	Student	Class
<=30	High	No	No
31...40	High	No	Yes
>40	Medium	No	Yes
>40	Low	Yes	Yes
<=30	Medium	No	No
<=30	Low	Yes	Yes
>40	Medium	Yes	Yes
31...40	High	Yes	Yes

Age	Income	Student	Class
<=30	High	No	No
>40	Low	Yes	No
31...40	Low	Yes	Yes
<=30	Medium	Yes	Yes
31...40	Medium	No	Yes
>40	Medium	No	No



2. one without general majority voting; i.e use ID3 algorithm (without Information Gain). Use INCOME as root attribute, and nodes attributes of your choice;

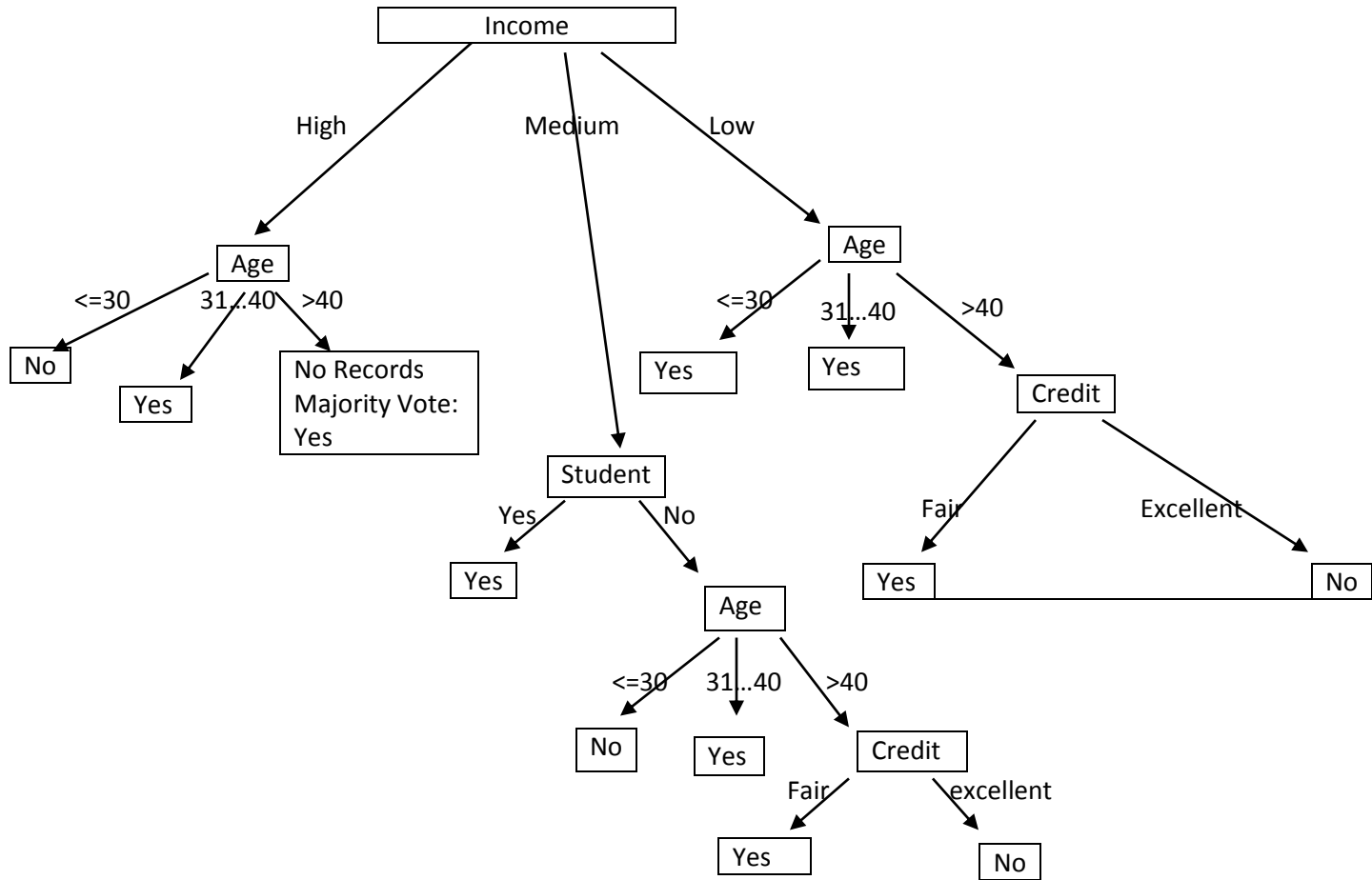


The first tree will have the list passed recursively. That is, all unused attributes get passed to each child node when the algorithm is called to generate each sub tree, and attributes used by the children of one child won't affect children of another child.

Age is an obvious split for High income as it requires no additional steps, and a shallow tree is preferable. Also, it generates only three rules.

Medium will be tougher. Age worked well enough in the lecture 8 example, and I've looked at all four combinations, and Medium will generate no less than 5 rules no matter what. Starting with student would take an extra step though, because something will have to be split by age, and something will have to be split by credit no matter what. Also, two additional rules would be generated for S-C-A for fair, 31...40 and excellent, <=30 due to a lack of records leading to a majority vote. S-A-C will have 5 rules though just like A-C and C-A. I'll do S-A-C because there's a building with that name. I don't need a better reason than that. Though A-C and C-A would be a quicker method to reach the same number of rules.

If I divide Low by Age, then it's four rules and two steps – the same as credit then age. What would happen if credit was the same for those two, anyway?



EVALUATE Predictive accuracy for each of your trees (sets of rules) – use the TEST Dataset below.

Tree 1:

a) $\text{Credit}(x, \text{Fair}) \rightarrow \text{Buys}(x, \text{Yes})$

b) $\text{Credit}(x, \text{Excellent}) \wedge \text{Student}(x, \text{No}) \rightarrow \text{Buys}(x, \text{No})$

c) $\text{Credit}(x, \text{Excellent}) \wedge \text{Student}(x, \text{Yes}) \rightarrow \text{Buys}(x, \text{Yes})$

Tree 2:

1) $\text{Income}(x, \text{High}) \wedge \text{Age}(x, \leq 30) \rightarrow \text{Buys}(x, \text{No})$

2) $\text{Income}(x, \text{High}) \wedge \text{Age}(x, 31 \dots 40) \rightarrow \text{Buys}(x, \text{Yes})$

3) $\text{Income}(x, \text{High}) \wedge \text{Age}(x, > 40) \rightarrow \text{Buys}(x, \text{Yes})$

4) $\text{Income}(x, \text{Medium}) \wedge \text{Student}(x, \text{Yes}) \rightarrow \text{Buys}(x, \text{Yes})$

5) $\text{Income}(x, \text{Medium}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(x, \leq 30) \rightarrow \text{Buys}(x, \text{No})$

6) $\text{Income}(x, \text{Medium}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(x, 31 \dots 40) \rightarrow \text{Buys}(x, \text{Yes})$

7) $\text{Income}(x, \text{Medium}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(x, > 40) \wedge \text{Credit}(x, \text{Fair}) \rightarrow \text{Buys}(x, \text{Yes})$

8) $\text{Income}(x, \text{Medium}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(x, > 40) \wedge \text{Credit}(x, \text{Excellent}) \rightarrow \text{Buys}(x, \text{No})$

9) $\text{Income}(x, \text{Low}) \wedge \text{Age}(x, \leq 30) \rightarrow \text{Buys}(x, \text{Yes})$

10) $\text{Income}(x, \text{Low}) \wedge \text{Age}(x, 31 \dots 40) \rightarrow \text{Buys}(x, \text{Yes})$

11) $\text{Income}(x, \text{Low}) \wedge \text{Age}(x, > 40) \wedge \text{Credit}(x, \text{Fair}) \rightarrow \text{Buys}(x, \text{Yes})$

12) $\text{Income}(x, \text{Low}) \wedge \text{Age}(x, > 40) \wedge \text{Credit}(x, \text{Excellent}) \rightarrow \text{Buys}(x, \text{No})$

Age	Income	Student	Credit	*Buys	Hit or Miss 1	Hit or Miss 2
≤ 30	High	Yes	Fair	Yes	Hit(a)	Miss
31...40	Low	No	Fair	Yes	Hit(a)	Hit(10)
31...40	High	Yes	Excellent	No	Miss	Miss
> 40	Low	Yes	Fair	Yes	Hit(a)	Hit(11)
> 40	Low	Yes	Excellent	No	Miss	Hit(12)
≤ 30	Low	No	Fair	No	Miss	Miss

Both are 50% (3/6). Therefore, given the difficulty of fighting with Microsoft Word to align the little labels without scrambling everything that's been done so far, the second tree was not worth the trouble.

Problem 2

Create test data sets for your sets rules corresponding to trees 1 and 2 that guarantees 100% predictive accuracy.

(It'll be easier to use the old test data and adjust it where it missed)

Age	Income	Student	Credit	*Buys
31...40	High	Yes	Fair	Yes
31...40	Low	No	Fair	Yes
<=30	High	No	Excellent	No
>40	Low	Yes	Fair	Yes
>40	Low	No	Excellent	No
<=30	Medium	No	Excellent	No

Row 1: Already satisfies tree 1 (a) - change age to 31...40 and it meets tree 2 (2).

Row 2: Already good (a,10)

Row 3: Double Miss - change student to no (b) and age to <=30 (1)

Row 4: Already good (a,11)

Row 5: tree 2 is fine (12) - can only change student without effecting tree 2. Satisfies (b), luckily

Row 6: Double Miss – Excellent credit (b) and Medium Income (5)

Checked for duplicates, of which there are none.

Problem 3

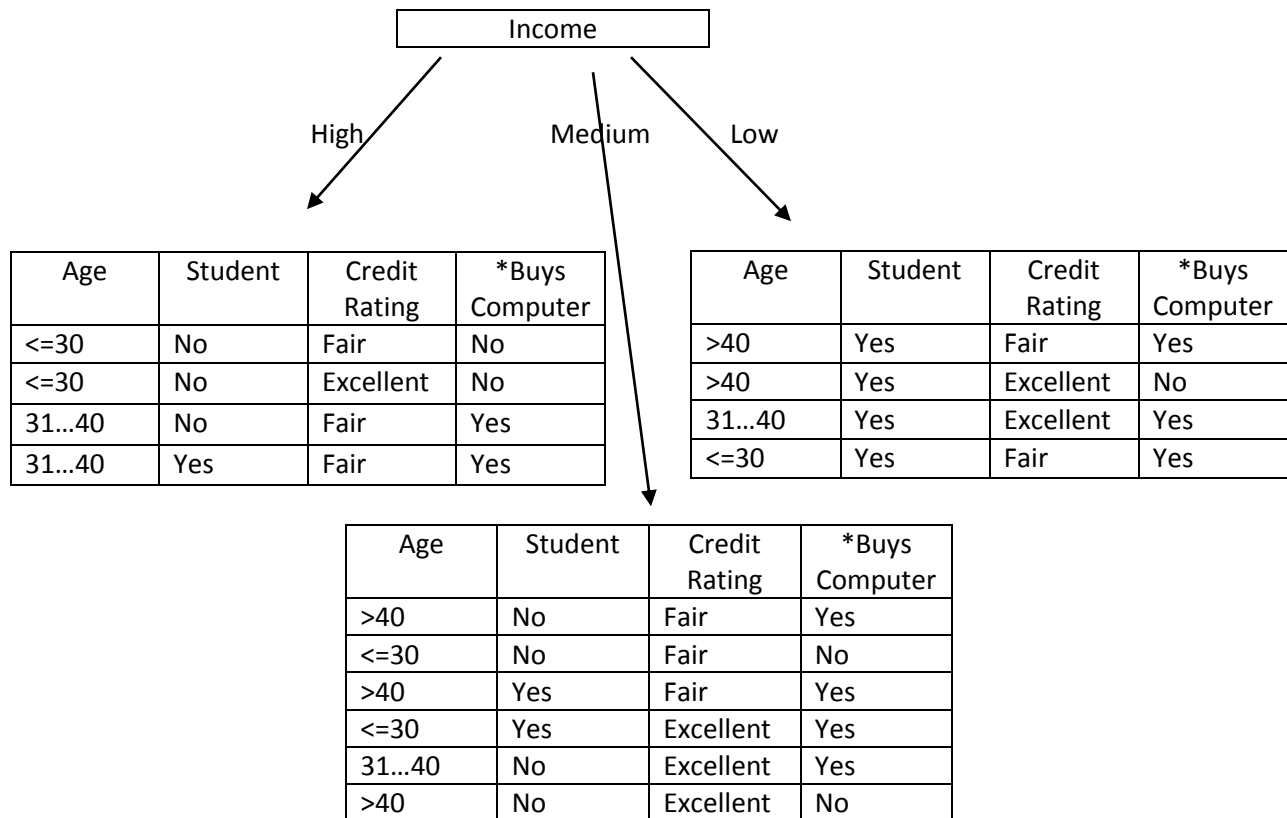
Compute the predictive accuracy of the set of discriminant rules in the lecture notes L8 with respect of the TEST Dataset from Problem 1.

The rules are:

- 1) IF *age* = " ≤ 30 " AND *student* = "no" THEN *buys_computer* = "no"
- 2) IF *age* = " ≤ 30 " AND *student* = "yes" THEN *buys_computer* = "yes"
- 3) IF *age* = "31...40" THEN *buys_computer* = "yes"
- 4) IF *age* = " > 40 " AND *credit_rating* = "excellent" THEN *buys_computer* = "no"
- 5) IF *age* = " > 40 " AND *credit_rating* = "fair" THEN *buys_computer* = "yes"

Age	Income	Student	Credit	*Buys	Hit or Miss
≤ 30	High	Yes	Fair	Yes	Hit(2)
31...40	Low	No	Fair	Yes	Hit(3)
31...40	High	Yes	Excellent	No	Miss
> 40	Low	Yes	Fair	Yes	Hit(5)
> 40	Low	Yes	Excellent	No	Hit(4)
≤ 30	Low	No	Fair	No	Hit(1)

$$5/6 * 100\% = 83.3\%$$



The second tree will be generated by passing a pointer to one absolute list of attributes so that when an attribute is removed to be used by a node, cousin nodes will also be unable to use that attribute and not just children.

The first step was just fine. In fact, so was the second. But after High gets split at Age, nothing else can be. Because the algorithm is recursive, it has to hit the bottom before it can move to the right, so it'll exhaust all of the attributes in the attribute list before it reaches low income. Line 13 (or 14 of the other algorithm) runs itself for each node, requiring it to finish on the leftmost before moving to the next and so on. And for it to finish on the left node, it must finish on the subnodes of that node. This is a waste of the credit rating attribute in this case.

