Table 8.1 Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Example 8.1 **Induction of a decision tree using information gain.** Table 8.1 presents a training set, D, of class-labeled tuples randomly selected from the *AllElectronics* customer database. (The data are adapted from Quinlan [Qui86]. In this example, each attribute is discrete-valued. Continuous-valued attributes have been generalized.) The class label attribute, *buys_computer*, has two distinct values (namely, {*yes, no*}); therefore, there are two distinct classes (i.e., $m = 2$). Let class C_1 correspond to *yes* and class C_2 correspond to *no*. There are nine tuples of class *yes* and five tuples of class *no*. A (root) node N is created for the tuples in D. To find the splitting criterion for these tuples, we must compute the information gain of each attribute. We first use Eq. (8.1) to compute the expected information needed to classify a tuple in D:

$$Info(D) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940 \text{ bits}.$$

Next, we need to compute the expected information requirement for each attribute. Let's start with the attribute *age*. We need to look at the distribution of *yes* and *no* tuples for each category of *age*. For the *age* category "youth," there are two *yes* tuples and three *no* tuples. For the category "middle_aged," there are four *yes* tuples and zero *no* tuples. For the category "senior," there are three *yes* tuples and two *no* tuples. Using Eq. (8.2), the expected information needed to classify a tuple in D if the tuples are partitioned according to *age* is

$$Info_{age}(D) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right)$$

$$+\frac{4}{14}\times\left(-\frac{4}{4}\log_2\frac{4}{4}\right)$$
$$+\frac{5}{14}\times\left(-\frac{3}{5}\log_2\frac{3}{5}-\frac{2}{5}\log_2\frac{2}{5}\right)$$
$$=0.694 \text{ bits}.$$

Hence, the gain in information from such a partitioning would be

$$Gain(age) = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246 \text{ bits}.$$

Similarly, we can compute $Gain(income) = 0.029$ bits, $Gain(student) = 0.151$ bits, and $Gain(credit\_rating) = 0.048$ bits. Because *age* has the highest information gain among the attributes, it is selected as the splitting attribute. Node N is labeled with *age*, and branches are grown for each of the attribute's values. The tuples are then partitioned accordingly, as shown in Figure 8.5. Notice that the tuples falling into the partition for *age* = *middle_aged* all belong to the same class. Because they all belong to class "*yes*," a leaf should therefore be created at the end of this branch and labeled "*yes*." The final decision tree returned by the algorithm was shown earlier in Figure 8.2. ■

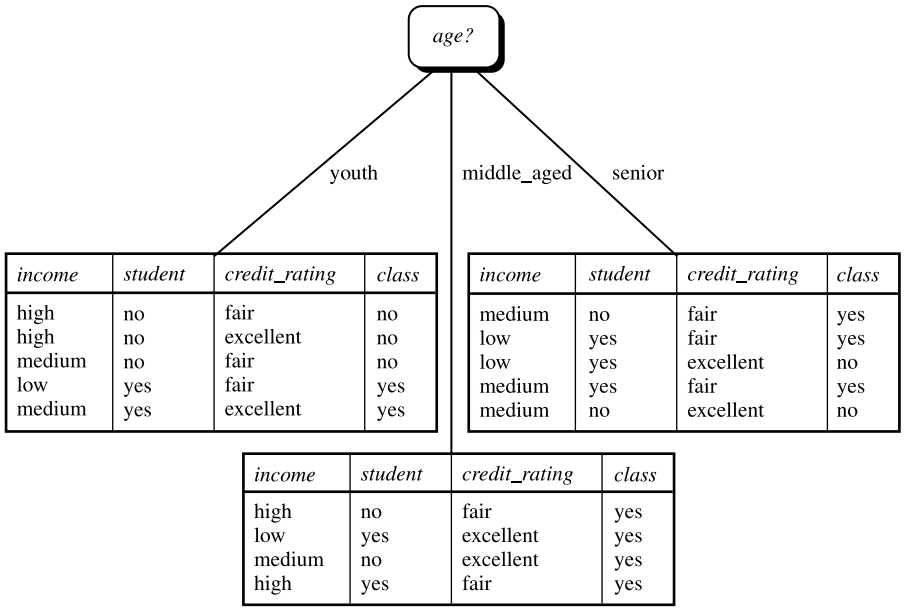


Figure 8.5 The attribute *age* has the highest information gain and therefore becomes the splitting attribute at the root node of the decision tree. Branches are grown for each outcome of *age*. The tuples are shown partitioned accordingly.

Table 8.1 Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Y - class Yes buys computer,

N - class No

if I use student as split attribute:

No	Yes
1 — N	5 — Y
2 — N	6 — N
3 — Y	7 — Y
4 — Y	9 — Y
8 — N	10 — Y
12 — Y	11 — Y
14 — N	13 Y

7 — 3Y, 4N

7 — 6Y, 1N

5 — 2Y, 3N

VS age ⇒ pure 4 — 4Y

5 — 3Y, 2N

Table 8.1 Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

If I use income as split attr:

low	medium	high
5 - y	4 - y	1 - n
6 - n	8 - n	2 - n
7 - y	10 - y	3 - y
9 - y	11 - y	13 - y
	12 - y	
	14 - n	

⇒

4 < 3 y / 1 n

6 < 4 y / 2 n

4 < 2 y / 2 n

---

vs age ⇒

5 < 2 y / 3 n

4 < 4 y

5 < 3 y / 2 n

vs student ⇒

7 < 3 y / 4 n

7 < 6 y / 1 n

Table 8.1 Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

If I use credit as split attr:

fair
1 - n
3 - y
4 - y
5 - y
8 - n
9 - y
10 - y
13 - y

excellent
2 - n
6 - n
7 - y
11 - y
12 - y
14 - n

⇒

8 — 6y
— 2n

6 — 3y
— 3n

vs age ⇒

5 — 2y
— 3n

4 — 4y

5 — 3y
— 2n

vs income ⇒

4 — 3y
— 1n

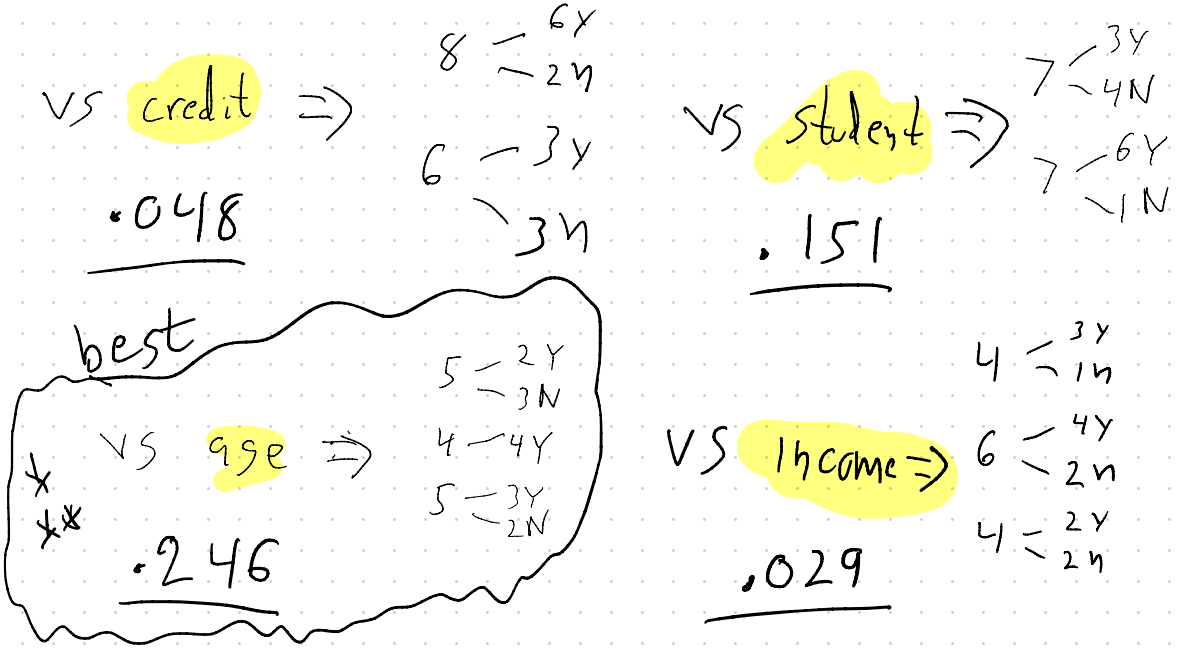
6 — 4y
— 2n

4 — 2y
— 2n

vs student ⇒

7 — 3y
— 4n

7 — 6y
— 1n



* Using age as splitter means only need to continue with 10/14, all others require going on with all

$$Gain(age) = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246 \text{ bits}.$$

Similarly, we can compute $Gain(income) = 0.029$ bits, $Gain(student) = 0.151$ bits, and $Gain(credit\_rating) = 0.048$ bits. Because *age* has the highest information gain among the attributes, it is selected as the splitting attribute. Node N is labeled with *age*,

How calculate:

vs credit ⇒ .048

8 — 6Y, 2N

6 — 3Y, 3N

vs student ⇒ .151

7 — 3Y, 4N

7 — 6Y, 1N

vs age ⇒ .246

5 — 2Y, 3N

4 — 4Y

5 — 3Y, 2N

vs income ⇒ .029

4 — 3Y, 1N

6 — 4Y, 2N

4 — 2Y, 2N

credit: $Info_{credit}(D) = -\frac{8}{14}\cdot\left(\frac{2}{8}\log\frac{2}{8} + \frac{6}{8}\log\frac{6}{8}\right)$

$+ -\frac{6}{14}\left(\frac{1}{2}\cdot\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2}\right)$

$= -\frac{8}{14}\left(\frac{1}{4}(-2) + \frac{3}{4}(-.415)\right) + \frac{-6}{14}\left(\frac{1}{2}(-1) + \frac{1}{2}(-1)\right)$

$= .892$

$Info(D) - Info_{credit}(D) = .940 - .892 = .048$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j). \qquad Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

$$Gain(age) = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246 \text{ bits}.$$