DECISSION TREE

Decision Tree

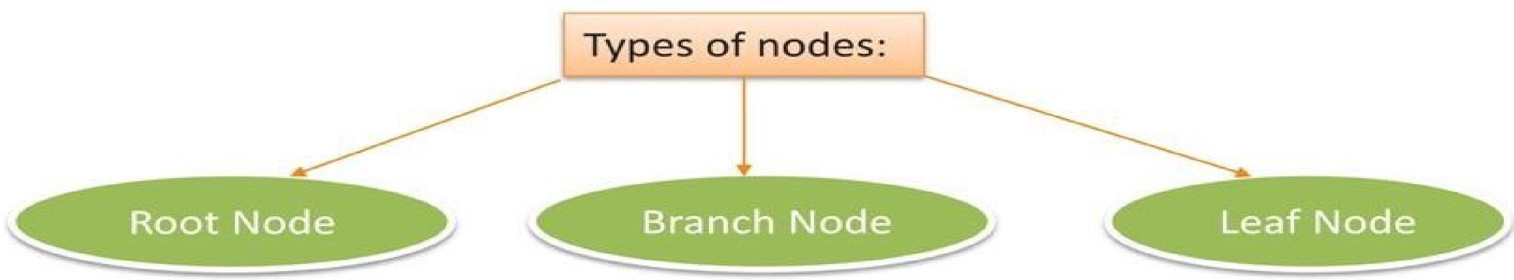
Decision Tree is a predictive data mining method with an objective to create a model that predicts the value of the dependent variable based on several independent variables. The two main types of Decision Trees are:

Classification Trees	Regression Trees
The predicted outcome is the class to which the data belongs.	The predicted outcome is a numeric estimate of the outcome.
Example 1: Customer is "Profitable" or "Non-profitable" Example 2: House price is "<100,000" or ">100,000"	Example 1: Customer will give a profit of $178.35 Example 2: House price is "125,350"

What are Decision trees?

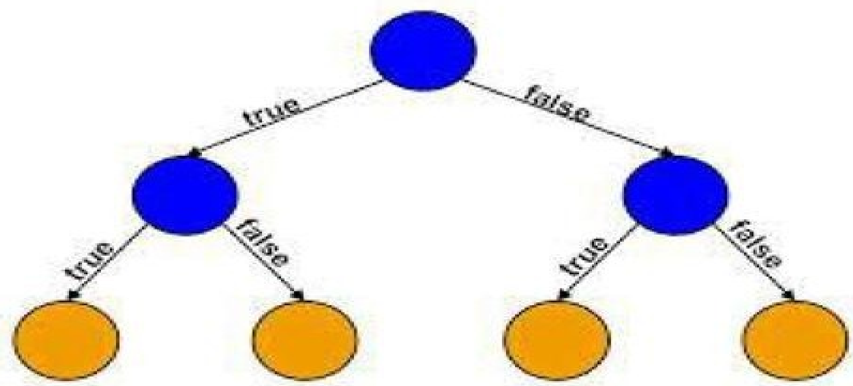
A decision tree is a tree-like structure in which internal node represents test on an attribute, each branch represents outcome of test and each leaf node represents class label (decision taken after computing all attributes). A path from root to leaf represents classification rules.

Thus, a decision tree consists of 3 types of nodes:

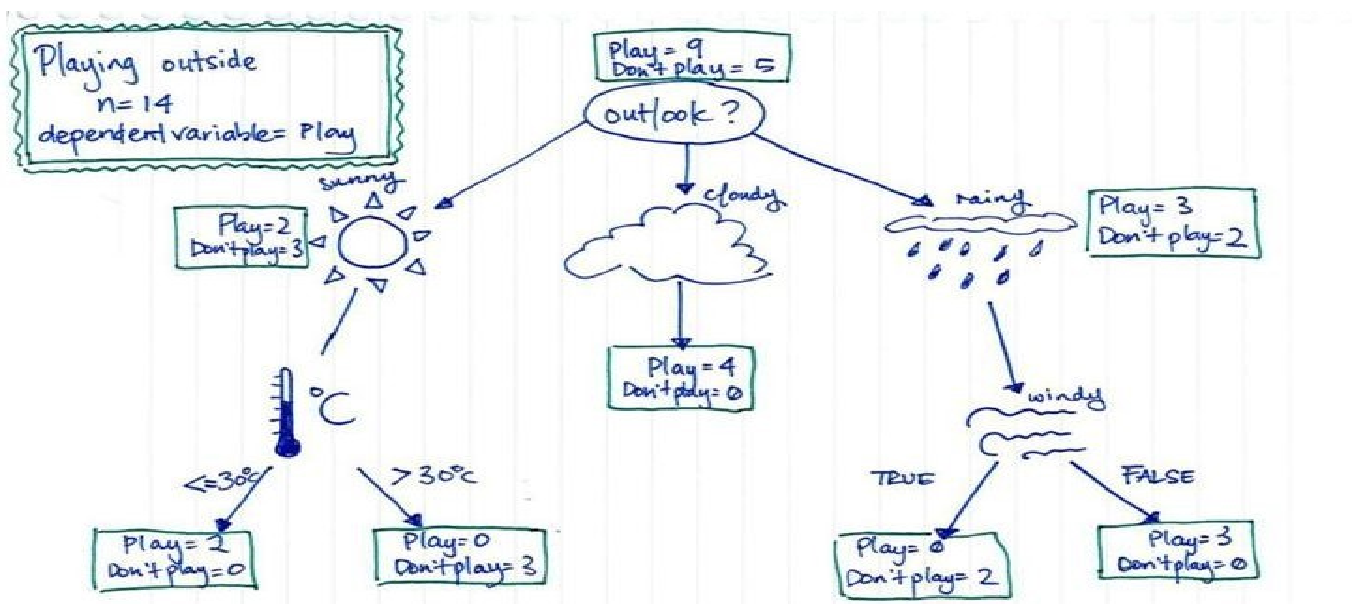


Decision Trees

Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value.



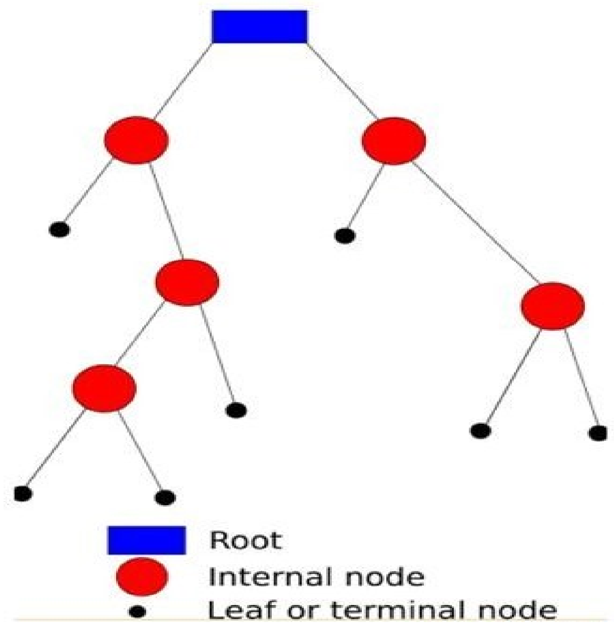
Decision Trees : Example



How to build decision trees?

Use training data to build model Tree generator determines:

- ✓ Which variable to split at a node and the value of the split
- ✓ Decision to stop (make a terminal note) or split again
- ✓ Assign terminal nodes to a class



Decision Tree Examples

Training Data

rec	Age	Income	Student	Credit_rating	Buys_computer
r1	<=30	High	No	Fair	No
r2	<=30	High	No	Excellent	No
r3	31…40	High	No	Fair	Yes
r4	>40	Medium	No	Fair	Yes
r5	>40	Low	Yes	Fair	Yes
r6	>40	Low	Yes	Excellent	No
r7	31…40	Low	Yes	Excellent	Yes
r8	<=30	Medium	No	Fair	No
r9	<=30	Low	Yes	Fair	Yes
r10	>40	Medium	Yes	Fair	Yes
r11	<-=30	Medium	Yes	Excellent	Yes
r12	31…40	Medium	No	Excellent	Yes
r13	31…40	High	Yes	Fair	Yes
r14	>40	Medium	No	Excellent	No

Decision Tree 1, Root : Student

Step-1:

student

No

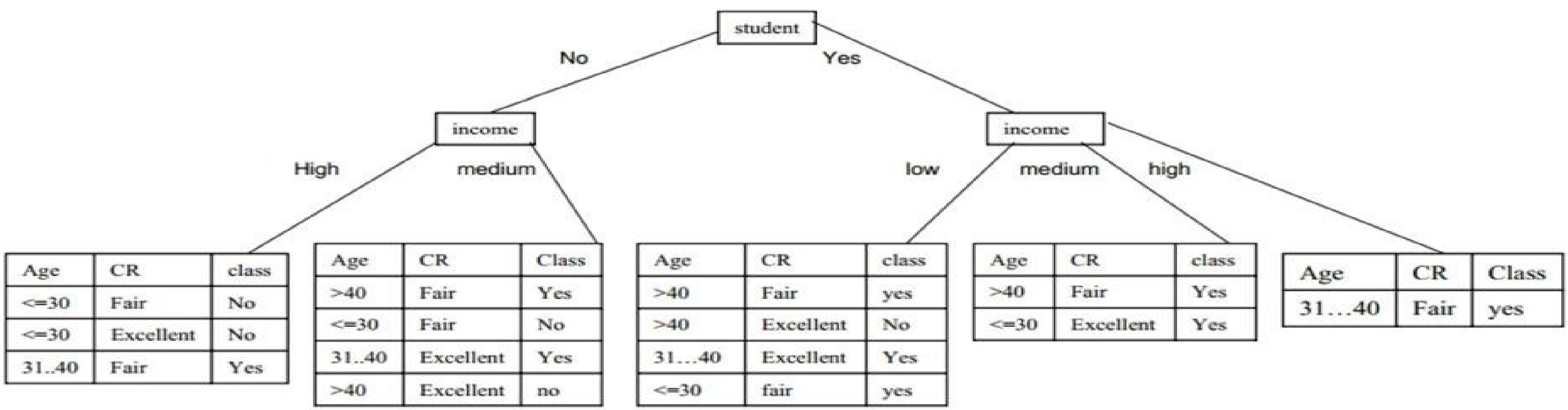
Age	income	CR	Class
<=30	High	Fair	No
<=30	High	Excellent	No
30…40	High	Fair	Yes
>40	Medium	Fair	Yes
<=30	Medium	Fair	No
31…40	Medium	Excellent	Yes
>40	Medium	Excellent	no

Yes

Age	income	CR	Class
>40	Low	Fair	Yes
>40	Low	excellent	No
31…40	Low	Excellent	Yes
<=30	Low	Fair	Yes
>40	Medium	Fair	Yes
<=30	Medium	Excellent	Yes
31…40	high	fair	yes

Decision Tree 1, Root : Student

Step-2:



Age	CR	class
<=30	Fair	No
<=30	Excellent	No
31..40	Fair	Yes

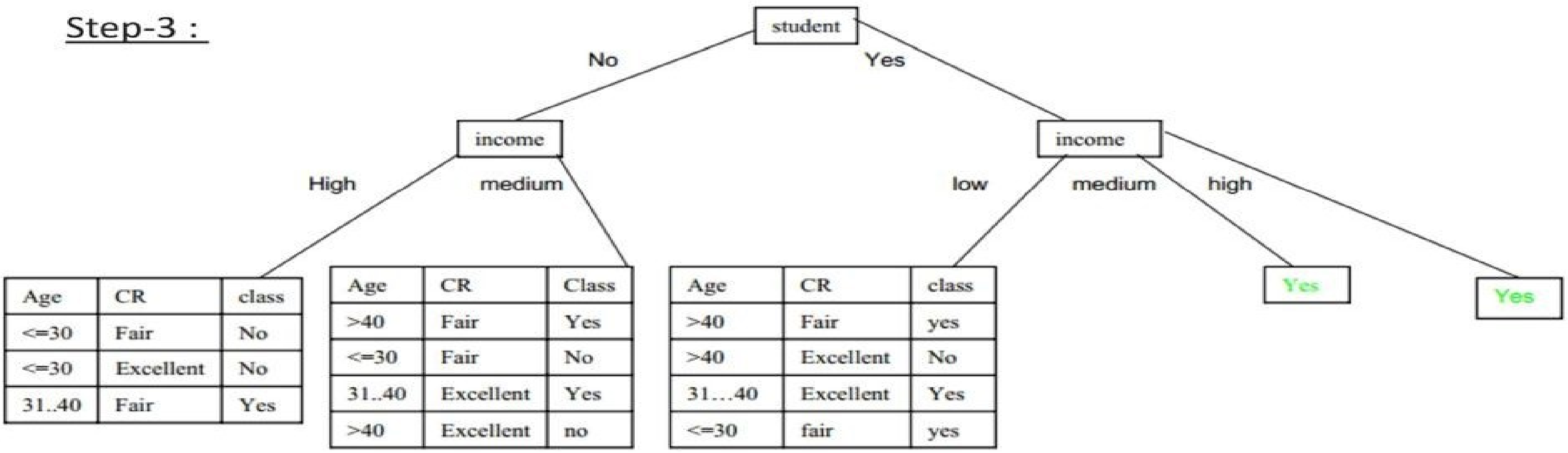
Age	CR	Class
>40	Fair	Yes
<=30	Fair	No
31..40	Excellent	Yes
>40	Excellent	no

Age	CR	class
>40	Fair	yes
>40	Excellent	No
31…40	Excellent	Yes
<=30	fair	yes

Age	CR	class
>40	Fair	Yes
<=30	Excellent	Yes

Age	CR	Class
31…40	Fair	yes

Step-3 :



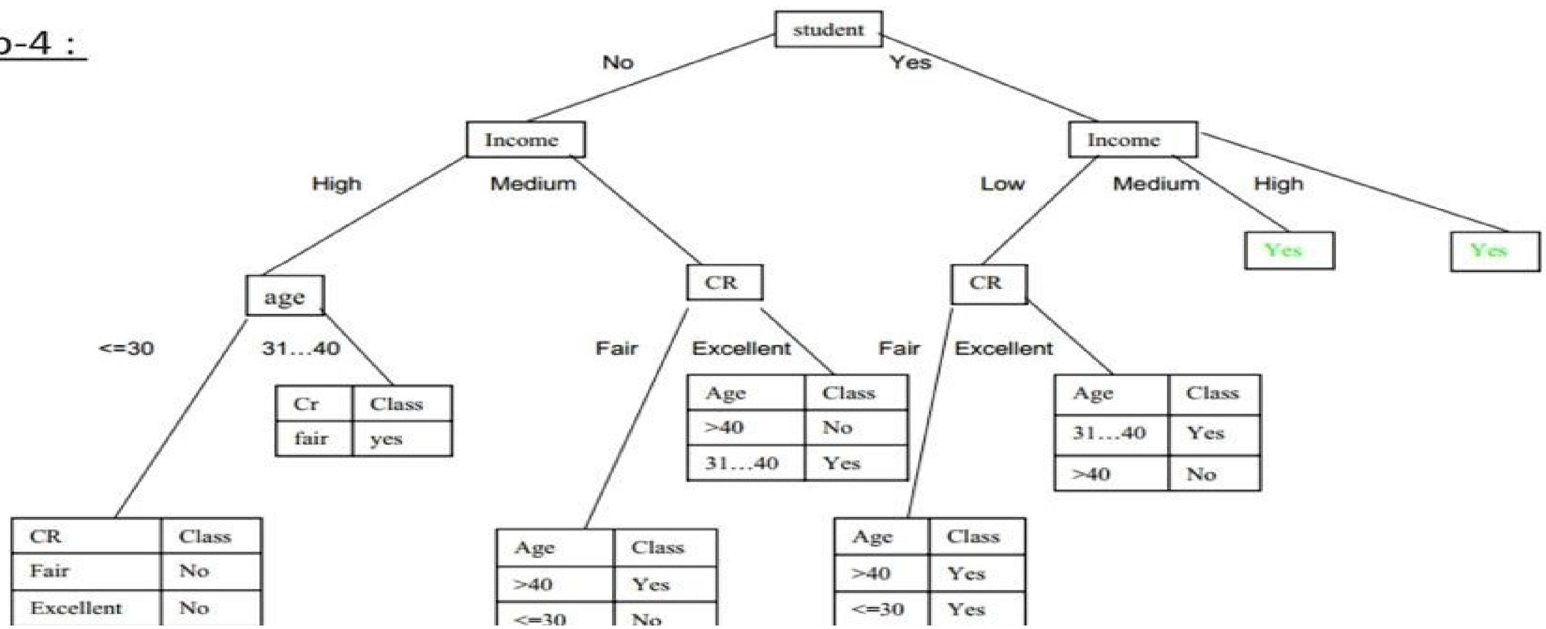
Age	CR	class
<=30	Fair	No
<=30	Excellent	No
31..40	Fair	Yes

Age	CR	Class
>40	Fair	Yes
<=30	Fair	No
31..40	Excellent	Yes
>40	Excellent	no

Age	CR	class
>40	Fair	yes
>40	Excellent	No
31…40	Excellent	Yes
<=30	fair	yes

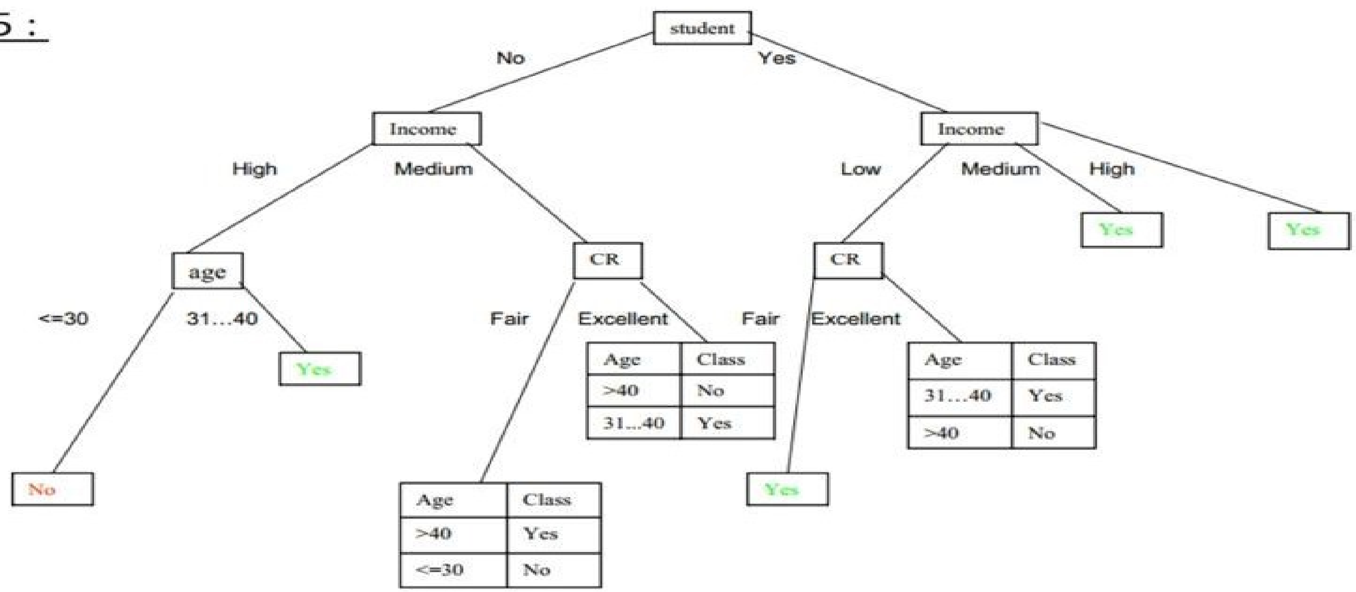
Decision Tree 1, Root : Student

<u>Step-4 :</u>

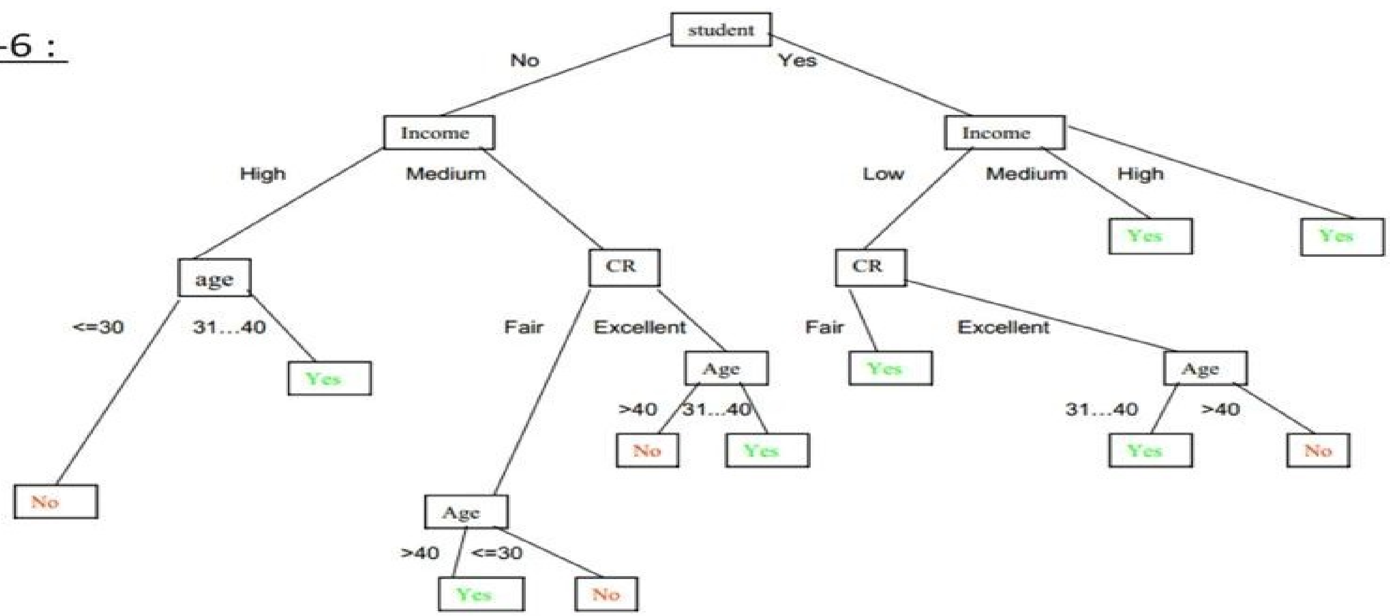


Decision Tree 1, Root : Student

Step-5 :



Step-6 :


student
No
Yes
Income
High
Medium
age
<=30
31…40
Yes
No
CR
Fair
Excellent
Age
>40
31…40
No
Yes
Age
>40
<=30
Yes
No
Income
Low
Medium
High
Yes
Yes
CR
Fair
Excellent
Yes
Age
31…40
>40
Yes
No

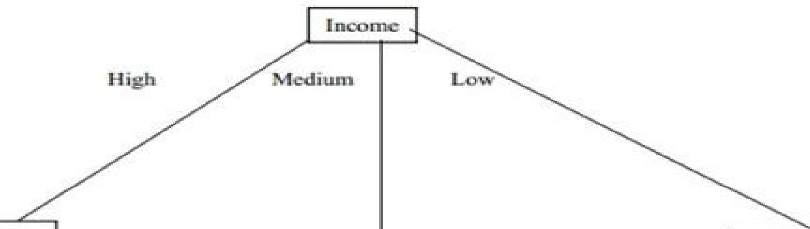
Decision Tree 1, Root : Student

Classifcation rules :

- 1. student(no)^income(high)^age(<=30) => buys_computer(no)
- 2. student(no)^income(high)^age(31...40) => buys_computer(yes)
- 3. student(no)^income(medium)^CR(fair)^age(>40) => buys_computer(yes)
- 4. student(no)^income(medium)^CR(fair)^age(<=30) => buys_computer(no)
- 5. student(no)^income(medium)^CR(excellent)^age(>40) => buys_computer(no)
- 6. student(no)^income(medium)^CR(excellent)^age(31..40) =>buys_computer(yes)
- 7. student(yes)^income(low)^CR(fair) => buys_computer(yes)
- 8. student(yes)^income(low)^CR(excellent)^age(31..40) => buys_computer(yes)
- 9. student(yes)^income(low)^CR(excellent)^age(>40) => buys_computer(no)
- 10. student(yes)^income(medium)=> buys_computer(yes)
- 11. student(yes)^income(high)=> buys_computer(yes)

Decision Tree 2, Root : Income

Step-1 :



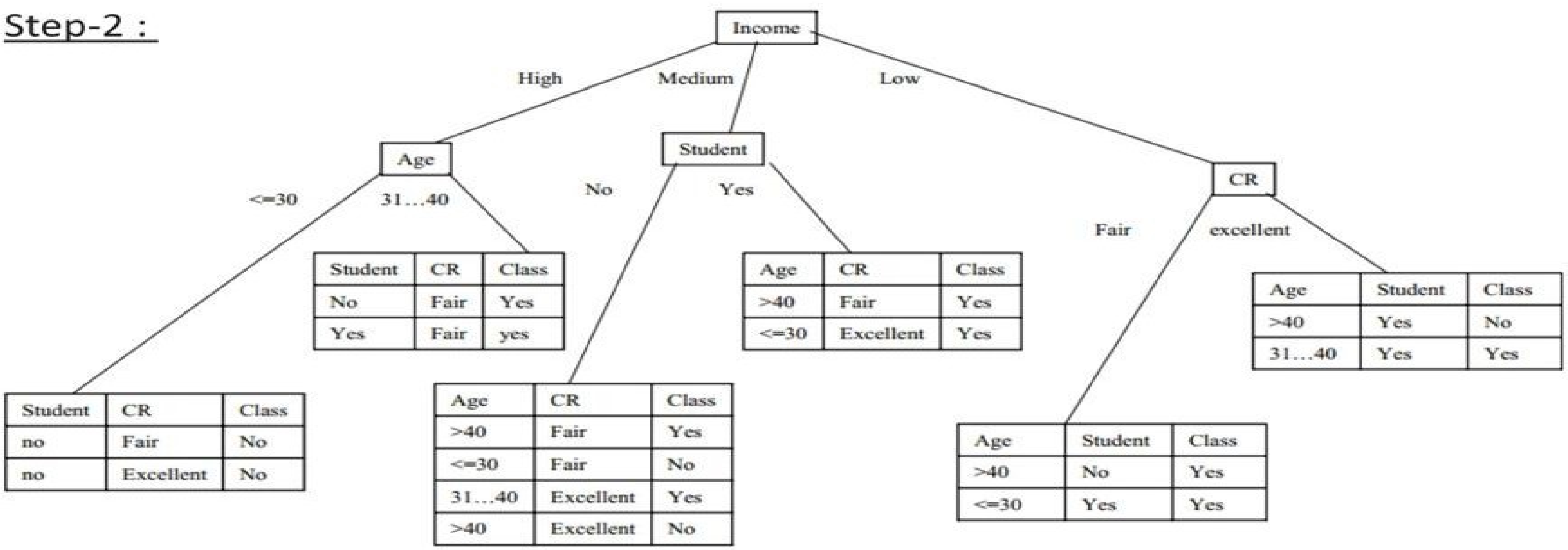
Age	Student	CR	Class
<=30	No	Fair	No
<=30	No	Excellent	No
31…40	No	Fair	Yes
31…40	Yes	Fair	yes

Age	Student	CR	Class
>40	No	Fair	Yes
<=30	No	Fair	No
>40	Yes	Fair	Yes
<=30	Yes	Excellent	yes
31…40	No	Excellent	yes
>40	No	Excellent	no

Age	Student	CR	Class
>40	No	Fair	Yes
>40	Yes	Excellent	No
31…40	Yes	Excellent	Yes
<=30	Yes	Fair	yes

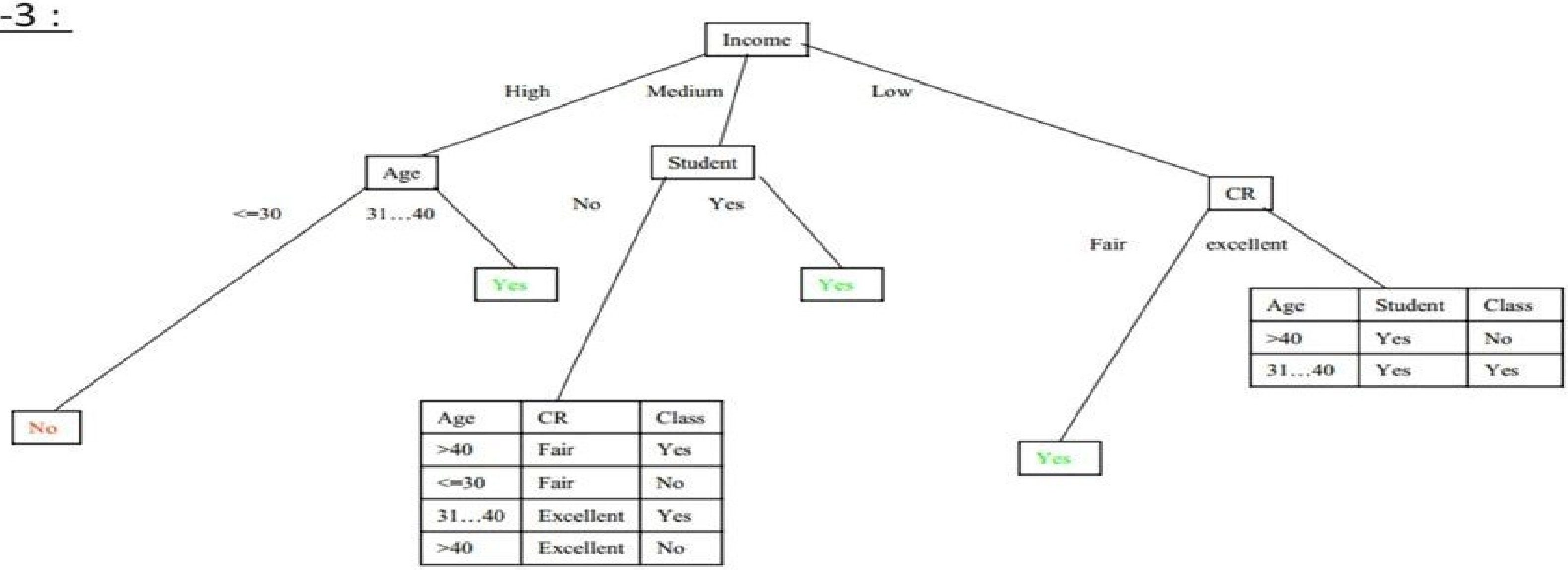
Decision Tree 2, Root : Income

Step-2 :



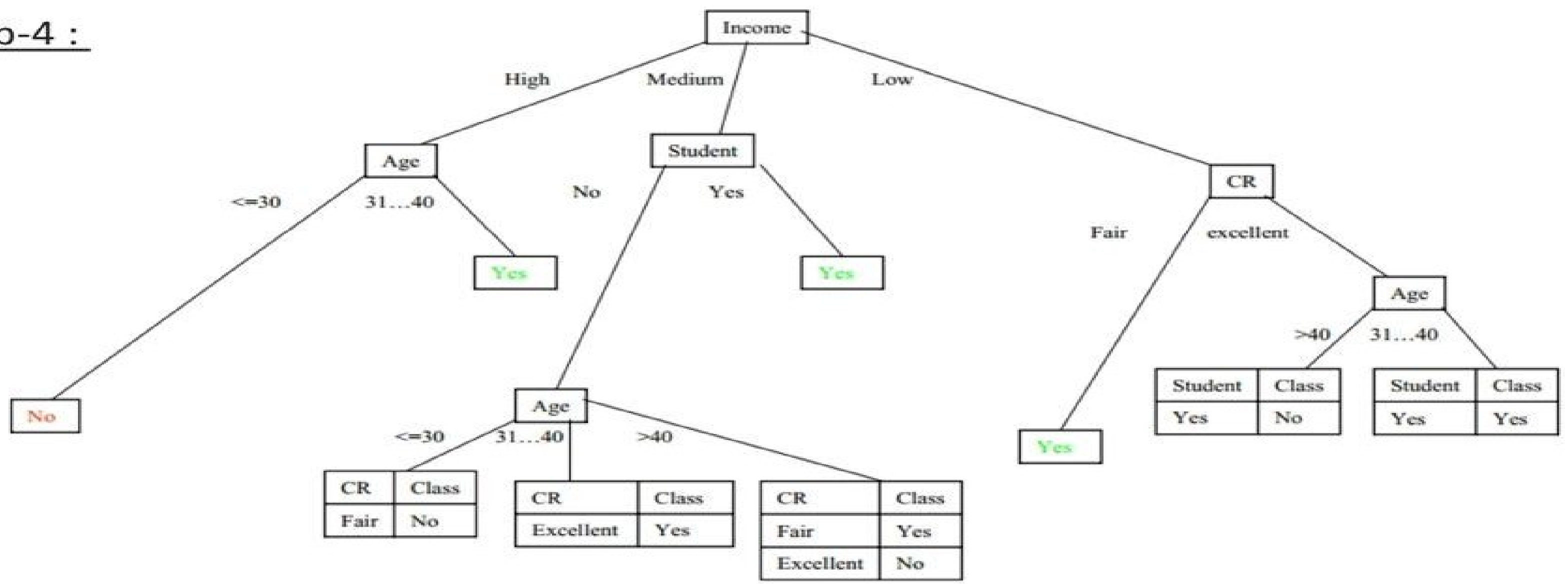
Decision Tree 2, Root : Income

Step-3 :



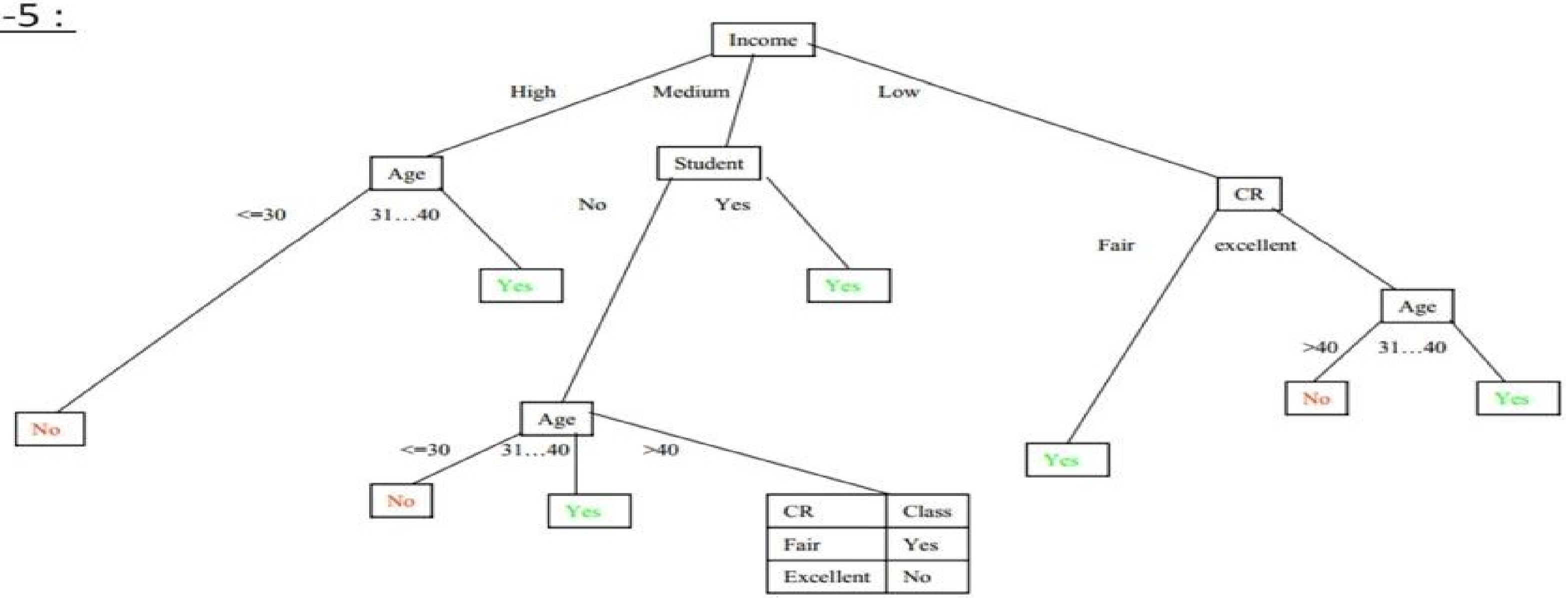
Decision Tree 2, Root : Income

Step-4 :



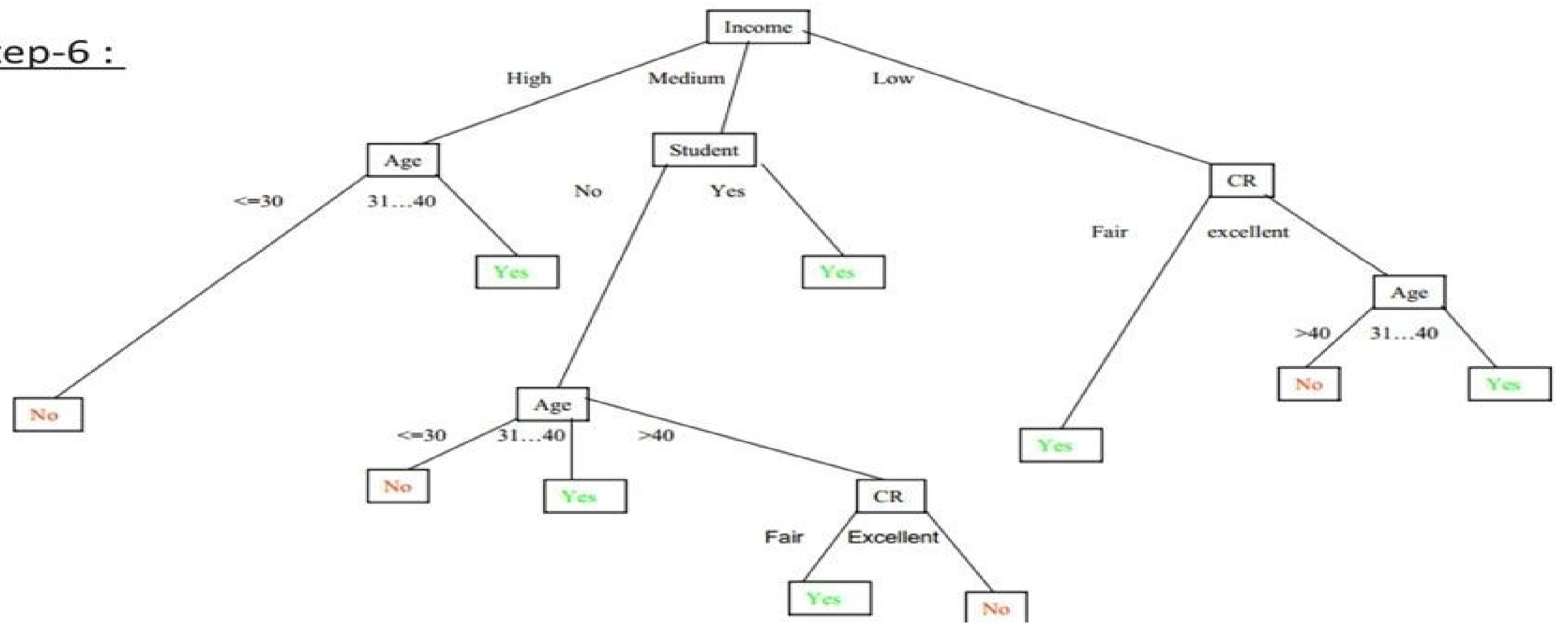
Decision Tree 2, Root : Income

Step-5 :



Decision Tree 2, Root : Income

Step-6 :



Decision Tree 2, Root : Income

Classifcation rules :

- 1. income(high)^age(<=30) => buys_computer(no)
- 2. income(high)^age(31...40) => buys_computer(yes)
- 3. income(medium)^student(no)^age(<=30) => buys_computer(no)
- 4. income(medium)^student(no)^age(31...40) => buys_computer(yes)
- 5. income(medium)^student(no)^age(>40)^CR(fair) => buys_computer(yes)
- 6. income(medium)^student(no)^age(>40)^CR(excellent) => buys_computer(no)
- 7. income(medium)^student(yes)=> buys_computer(yes)
- 8. income(medium)^CR(fair)=> buys_computer(yes)
- 9. income(medium)^ CR(excellent)^age(>40)=> buys_computer(no)
- 10. income(medium)^ CR(excellent)^age(31...40)=> buys_computer(yes)

Which Tree to choose?

The core algorithm for building decision trees called ID3 by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses Entropy and Information Gain to construct a decision tree.

The topmost decision node in a tree which corresponds to the best predictor is called root node.

Information Gain:

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.

Formulas for information gain

$$I(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$$E(A) = \sum_{i=1}^{v}\frac{p_i + n_i}{p+n} I(p_i, n_i)$$

$$Gain(A) = I(p,n) - E(A)$$

Which Tree to choose?

Calculations of information gain for Tree 1,

Root: Student

- I(P,N) = - (9/(9+5))Logsub2*(9/(9+5))-(5/(9+5))logsub2*(5/(9+5))
  = -.643(-0.64)+(-.357)(-1.49) = .944
- I(Psub1,Nsub1) = -(6/(6+1)Logsub2*(6/(6+1)-(1/(6+1))logsub2*(1/(6+1))
  = -.857(-.22)+(-.143)(-2.81) = .591
- I(Psub2,Nsub2) = -(3/(3+4)Logsub2*(3/(3+4)-(4/(3+4))logsub2*(4/(3+4))
  = -.423(-1.24)+(-.571)(-0.81) = .987

Student	P	N	I(Psubi,Nsubi)
Yes	6	1	.591
No	3	4	.987

- E(Student) = (((6+1)/14) * .591) = .296 + ((3+4)/14) * .987 = .493
  = .789

Gain(Student) = .944 - .789 = .155

Which Tree to choose?

Calculations of information gain for Tree 1,

Income(Left) node

- I(P,N) = -(3/(3+4)Logsub2*(3/(3+4)-(4/(3+4))logsub2*(4/(3+4))
  = -.423(-1.24)+(-.571)(-0.81) = .987
- I(Psub1,Nsub1) = -(1/(1+2)Logsub2*(1/(1+2)-(2/(1+2))logsub2*(2/(1+2))
  = -.333(-1.59)+(-.667)(-0.58) = .916
- I(Psub2,Nsub2) = -(2/(2+2)Logsub2*(2/(2+2)-(2/(2+2))logsub2*(2/(2+4))
  = -.5(-1)+(-.5)(-1) = 1

Income	P	N	I(Pi,Ni)
High	1	2	.916
Medium	2	2	1

- E(Income(L)) = (((1+2)/7) * .916) = .393 + ((2+2)/7) * 1 = .57
  = .963

Gain(Income(L)) = .987 - .963 = .024

Which Tree to choose?

Calculations of information gain for Tree 1,

Income(Right) node

- I(P,N) = -(6/(6+1)Logsub2*(6/(6+1)-(1/(6+1))logsub2*(1/(6+1))
  = -.857(-.22)+(-2.81)(-.143) = .591
- I(Psub1,Nsub1) = -(3/(3+1)Logsub2*(3/(3+1)-(1/(3+1))logsub2*(1/(3+1))
  = -.75(-0.42)+(-.25)(-2) = .815
- I(Psub2,Nsub2) = -(2/(2+0)Logsub2*(2/(2+0)-(0/(2+0))logsub2*(0/(2+0))
  = -1(0)-(0)(infinity) = 0
- I(Psub3,Nsub3) = -(1/(1+0)Logsub2*(1/(1+0)-(0/(1+0))logsub2*(0/(1+0))
  = -1(0)-(0)(infinity) = 0

Income	P	N	I(Pi,Ni)
Low	3	1	.815
Medium	2	0	0
High	1	0	0

- E(Income(R)) = (((3+1)/7) * .815) = .465 + ((2+0)/7) * 0 = 0 + ((1+0)/7) * 0 = 0 = .465

Gain(Income(R) = .987 - .465 = .522

Which Tree to choose?

Information gain measure :

Gain(student) =.155

Gain(income(L)) = .024

Gain(income(R)) = .522

Gain(age(1)) = .916

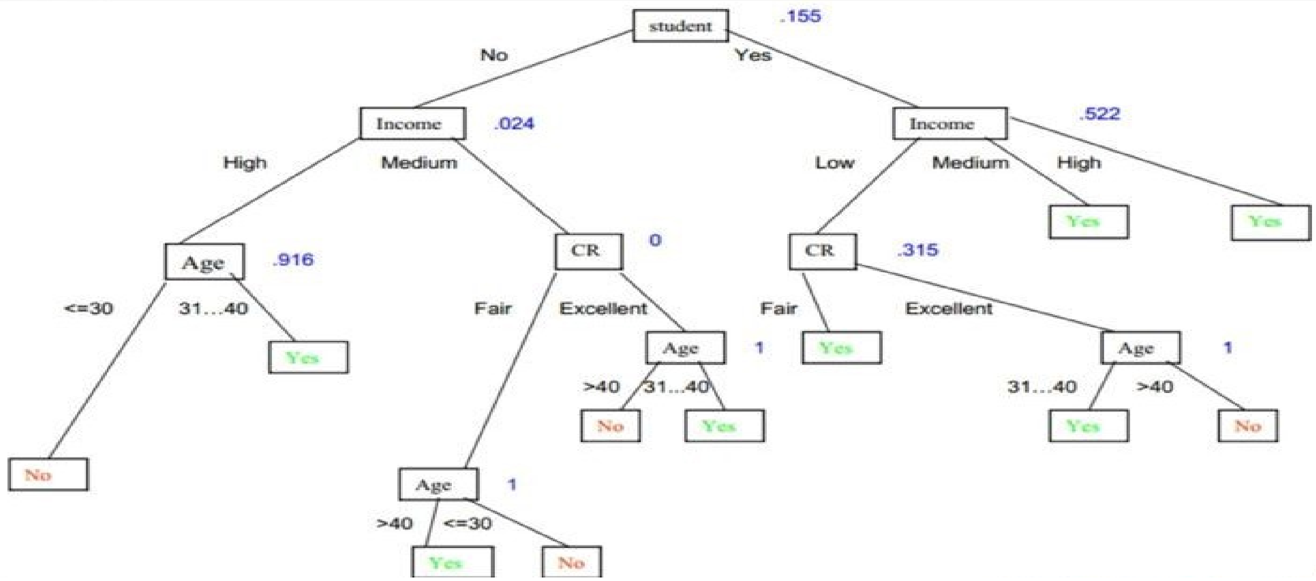
Gain(CR(L)) = 0

Gain(CR(R)) = .315

Gain(age(2)) = 1

Gain(age(3)) = 1

Gain(age(4)) = 1



Which Variable to use?

Criteria to split a node:

- Root node is the entire dataset (value of target variable is known for all cases)
- The root node is split such that in the nodes developed members of a single class (1 or 0) predominate
- Further splits performed similarly

Best Split

Purity measures for evaluating splits are:

- Gini co-efficient
- Entropy
- Chi-Square test

Purity is an indicator of how homogenous the child nodes (or splits) are:

- Degree to which the child nodes are made up of cases with the same target value
- Higher purity in the child nodes indicate a better split

Entropy

- Assume, there is a sample of size S
  - p+ = proportion of positive examples (Response = Yes)
  - p- = proportion of negative examples (Response = No)
- Entropy
  - Measures the impurity of S
  - Is calculated as the proportion of records with a particular value multiplied by the base 2 logarithm of that proportion

Algorithms for Decision Tree

Two popular algorithms for Decision Tree are:

CART (Classification and Regression Trees)	It is a non-parametric Decision Tree technique. It produces either classification or regression trees, depending on the variable type (categorical or numeric)
CHAID (Chi-Square Automatic Interaction Detection)	It is a Decision Tree technique, which is based on adjusted significance testing. It is also a non-parametric technique

In both the techniques, the output is highly visual and easy to interpret.

Differences between CART and CHAID

CART

- CART uses Gini Index as the splitting criteria at each node
- CART always splits the data into only two nodes
- CART splits the data as much as possible and then the tree is pruned using validation data to minimize classification error

CHAID

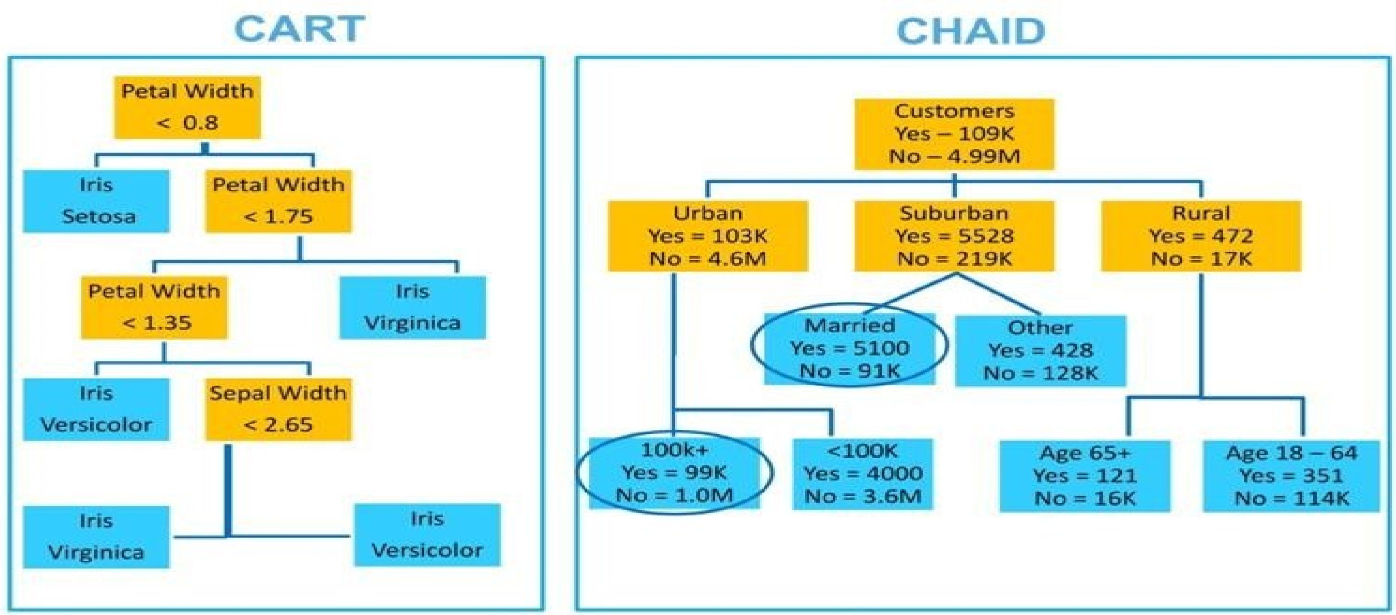
- CHAID uses the Chi-Square statistics
- CHAID can split the data into more than two nodes
- CHAID stops growing the tree when no further gain can be made in differentiating the segments

Example: CART vs. CHAID (Contd.)

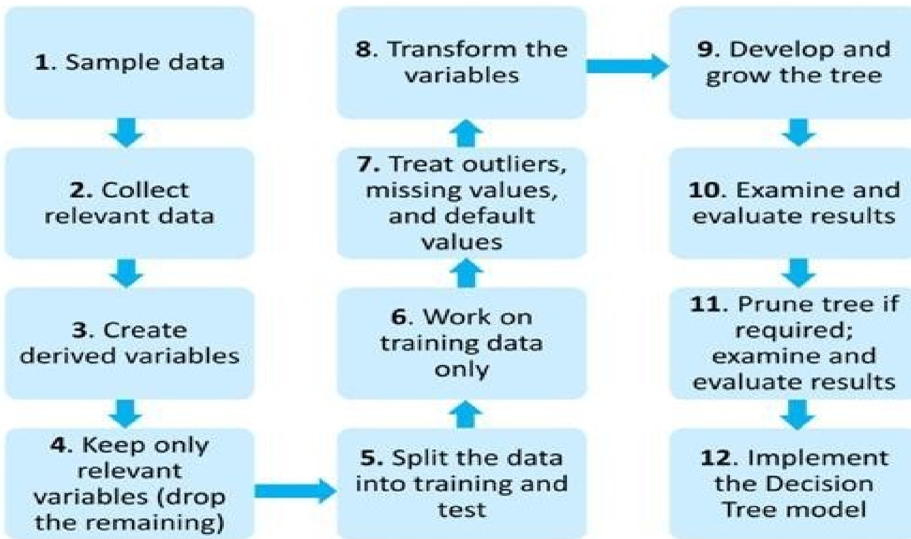
- Classification problem for CART example: Classify the Iris flower species into three categories Setosa, Virginica, and Versicolor
  - Attributes captured for the flowers: Sepal length and width, Petal length and width
  - CART method does recursive binary split of the data and creates the decision tree

- Classification problem for CHAID example: Classify the customers who will respond to the marketing campaign
  - Attributes used for splitting the data: Location, income, marital status, and age
  - The nodes circled are the ones with the highest response. These can be selected as the target set for the next campaign

Example: CART vs. CHAID



Steps for Creating a Decision Tree



DECISSION TREE

CASE STUDY

A Decision Tree is a supervised algorithm used in machine learning.

It is using a binary tree graph (each node has two children) to assign for each data sample a target value. The target values are presented in the tree leaves.

To reach to the leaf, the sample is propagated through nodes, starting at the root node. In each node a decision is made, to which descendant node it should go.
A decision is made based on the selected sample's feature.

Decision Tree learning is a process of finding the optimal rules in each internal tree node according to the selected metric.

The decision trees can be divided, with respect to the target values, into:

Classification trees used to classify samples, assign to a limited set of values - classes. In scikit-learn it is DecisionTreeClassifier.
Regression trees used to assign samples into numerical values within the range. In scikit-learn it is DecisionTreeRegressor.

Decision trees are a popular tool in decision analysis. They can support decisions thanks to the visual representation of each decision.

Below I show 4 ways to visualize Decision Tree in Python:

print text representation of the tree with sklearn.tree.export_text method
plot with sklearn.tree.plot_tree method (matplotlib needed)
plot with sklearn.tree.export_graphviz method (graphviz needed)
plot with dtreeviz package (dtreeviz and graphviz needed)