Machine learning

Decision trees

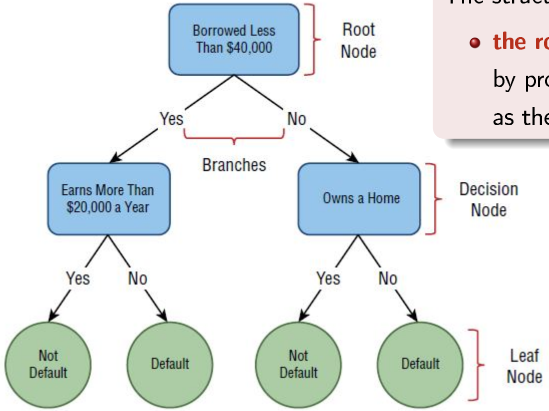
Decision trees use a tree-like structure to represent the relationship between input variables and potential outcomes.

The potential outcomes of a decision tree can be

- discrete - **classification tree**;
- continuous - **regression tree**.

Decision trees

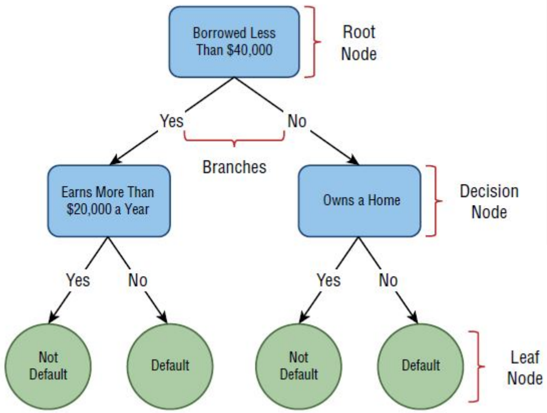
default



The structure of a decision tree

- **the root node** - which is followed by progressively smaller partitions as the tree splits and grows;

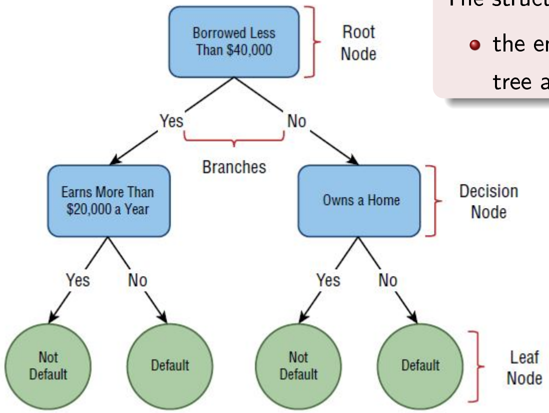
Decision trees



The structure of a decision tree

- at each point where the **tree splits**, a decision is taken, based on the values of a particular predictor;

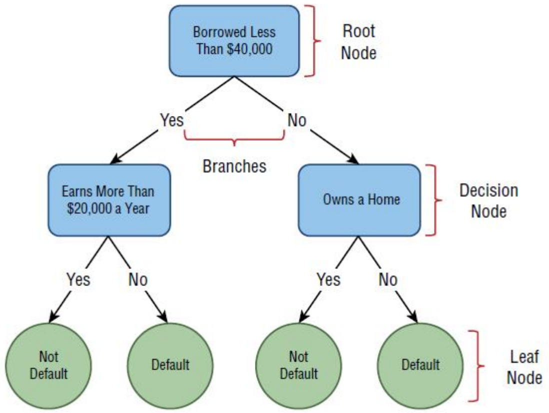
Decision trees



The structure of a decision tree

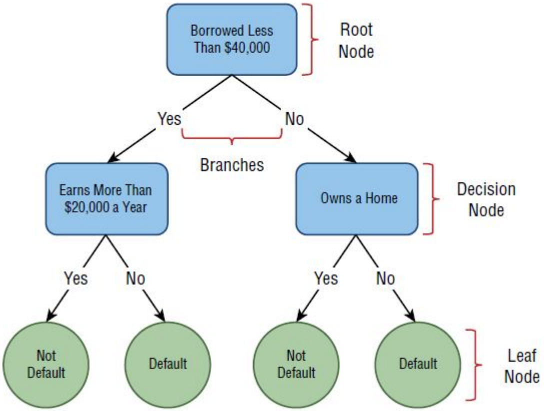
- the end or terminal nodes of the tree are known as the **leaf nodes**.

Decision trees



The leafs nodes **represent the predicted outcome based on the set of decisions made from the root node, through the decision nodes to the leaf node.**

Decision trees

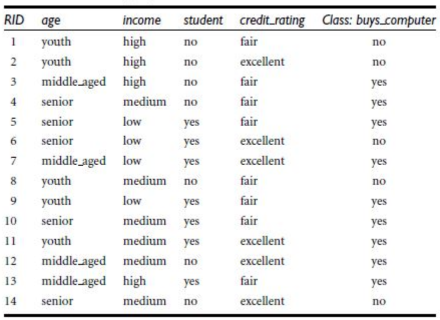


The logic of the tree can be easily interpreted as a rule for predicting:
whether the bank's future customers will or will not repay the loan.

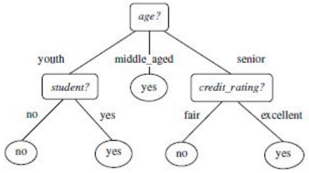
IF (customer borrows more than 40,000) AND (customer owns a home)
THEN (customer will not default)

Divide and conquer

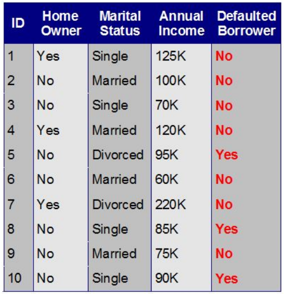
Decision trees are built using a technique called recursive partitioning - **divide and conquer**. This approach **repeatedly splits data into smaller and smaller subsets** until the algorithm identifies data within which the subsets are sufficiently **homogeneous** or another stopping criterion are met.



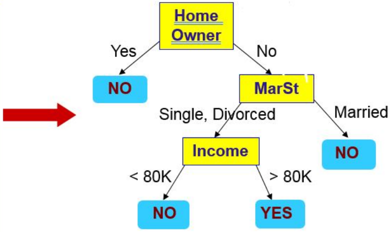
RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	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 of a Decision Tree

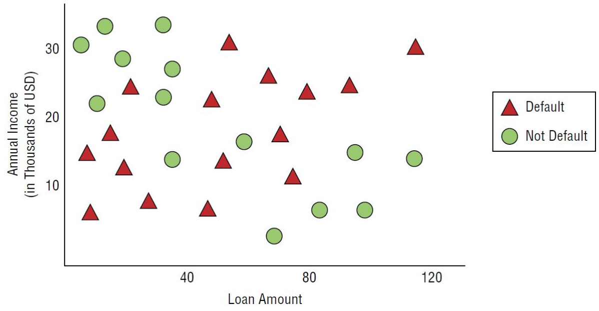


ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



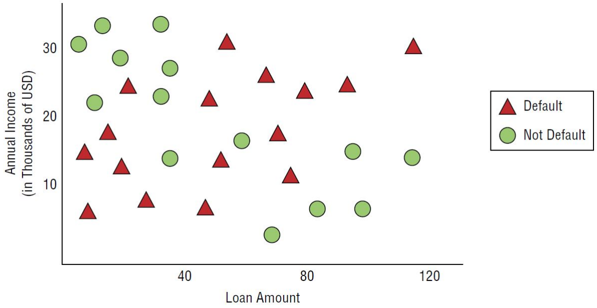
A partition is called **pure** if the decision variable belongs to a one class.

Example of a Decision Tree 2



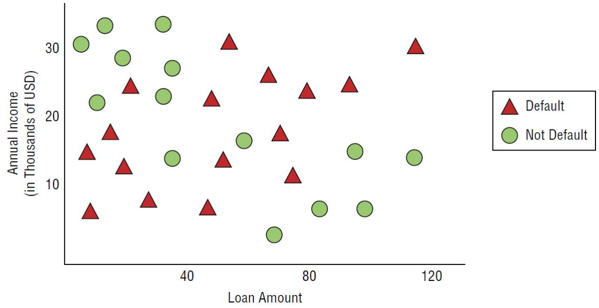
- let us assume that we have data about 30 personal loans issued by a bank.
- dataset includes information about the amount that was borrowed, the annual income of the customer, and whether the customer defaulted on the loan.
- of the 30 customers represented in the dataset, 16 defaulted and 14 did not.

Example of a Decision Tree 2



Which of the two features **Annual Income** or **Loan Amount** is most predictive of the target outcome **Default or Not Default**?.

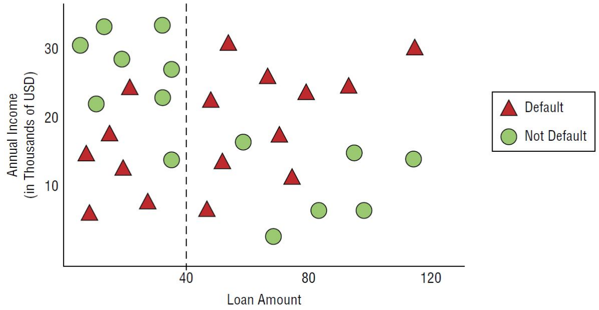
Example of a Decision Tree 2



Which of the two features **Annual Income** or **Loan Amount** is most predictive of the target outcome **Default or Not Default**?.

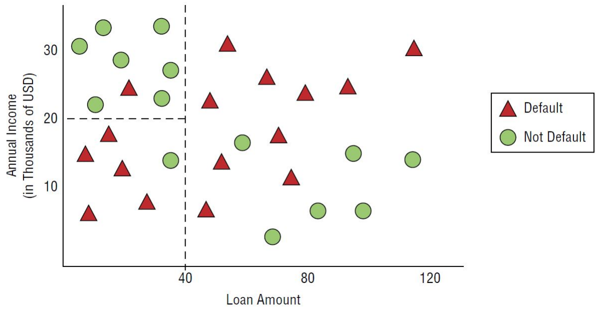
A loan amount of 40,000 as our best split.

Example of a Decision Tree 2



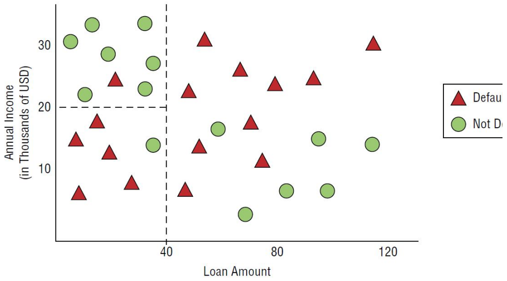
- with the split that we chose, we get 14 customers with a loan amount of 40,000 or less and 16 with a loan amount of 40,000 or more.
- among the customers who borrowed less than 40,000, eight of them did not default on their loans while six of them did.
- while for the customers who borrowed more than 40,000, 10 of them defaulted on their loans and 6 of them did not.

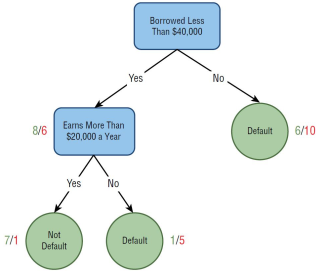
Example of a Decision Tree 2



- the next partition we make is within the group of customers who borrowed less than 40,000.
- among these customers, we can further partition the data into those who earn more than 20,000 a year and those who don't.

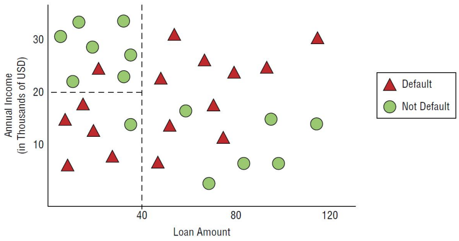
Example of a Decision Tree 2

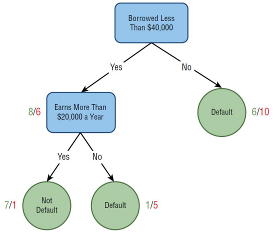




Decision tree of bank customers based on the loan amount and annual income. Each decision and leaf node shows the number of customers who defaulted (red number) and those who did not (green number).

Example of a Decision Tree 2





Decision tree algorithms use a **quantitative measure** which is commonly referred to as **purity** or **impurity** to determine the best split.

Purity in this sense means the degree to which data points within a partition are of the same class. **A partition where all of the data points are of the same class is considered pure**, while a partition with half of its data points are of one class and the other half are of a different class is considered impure.

Issues of Decision Tree

How should observations be split?

- method for specifying test condition - depending on attribute types;
- measure for evaluating the goodness of a test condition

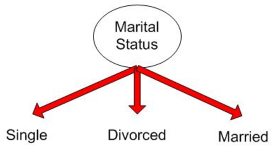
How should the splitting procedure stop?

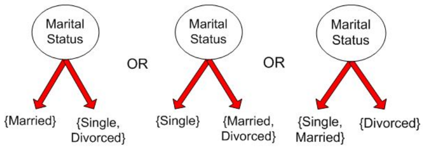
- stop splitting if all the observations belong to the same class
- early termination

Attributes split

Decision tree induction algorithms must provide a method for partitioning observations according to attribute types.

- **binary**
- **nominal**



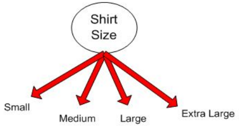


multi-way split: use as many partitions as distinct values.

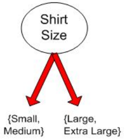
binary split: divides values into two subsets

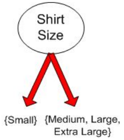
Attributes split

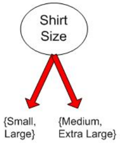
- **ordinal**



multi-way split: use as many partitions as distinct values.



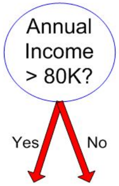




binary split: divides values into two subsets, preserve order property among attribute values

Attributes split

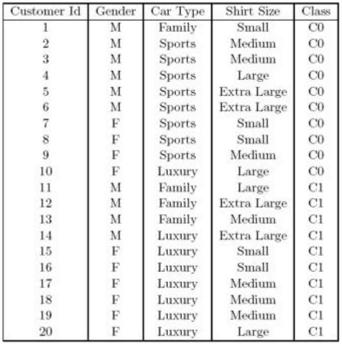
- **Continuous**



binary split: $(A < v)$ or $(A \leq v)$ - first sort the values of A in increasing order. The midpoint between each pair of adjacent values is considered as a possible split-point. Therefore, given v values of A, then $v - 1$ possible splits are evaluated.

How to determine the best split?

Let us consider the dataset

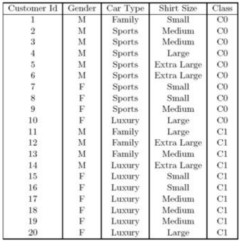


Customer Id	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1

Should we choose?

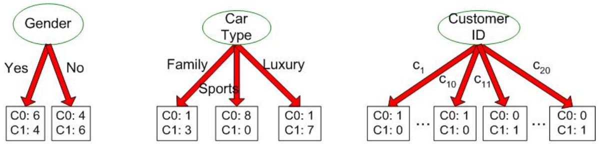
Before splitting we have 10 observations of class **0** and 10 observations of class **1**. We have high degree of impurity in dataset D.

How to determine the best split?



Customer Id	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1

Should we choose?



Which test condition is the best?

How to determine the best split?

Greedy approach: nodes with purer class distribution are preferred.

C0: 5
C1: 5

High degree of impurity

C0: 9
C1: 1

Low degree of impurity

Measures of impurity

The two most common measures of impurity are

1. **information gain**
2. **gini index**

Entropy

Entropy is an information theory metric that measures the impurity in a group of observations.

Entropy denotes the degree of mixing of observations in relation to the value of the target variable.

Let's consider an example. We have three boxes. There are 100 M&Ms candies in each.

1. in the first box we have: **1 yellow** candy and **99 reds**;
2. in the second box we have: **50 yellows** candies and **50 reds**;
3. in the third box we have: **25 yellows** candies and **75 reds**;

What colour candy do we take out of each box?

Entropy

- If the dataset contains only observations with the same value of the target variable, then we have no impurity, so we have: no entropy or zero entropy;
- If the two values of the target variable are evenly distributed across the dataset, then the dataset contains the maximum impurity;
- **The maximum entropy value is 1**.
- **A dataset containing the mixed values of the target variable will have an entropy between 0 and 1.**

Entropy

Let D, the dataset, be a training set of class-labeled observations. Suppose the class label attribute has m distinct values defining m distinct classes, C_i (for $i = 1, ..., m$). Let $C_{i,D}$ be the subset of observations of class C_i in D. Let $|D|$ and $|C_{i,D}|$ denote the number of observations in D and $C_{i,D}$ respectively.

The expected information needed to classify a observation in D is given by

$$H(D) = Info(D) = \sum_{i=1}^{n} p(x_i) log_2 \frac{1}{p(x_i)} = -\sum_{i=1}^{n} p(x_i) log_2 p(x_i)$$

$p(x_i)$ is the probability that an arbitrary observation in D belongs to class C_i and is $p(x_i) = \frac{|C_{i,D}|}{|D|}$.

A log function to the base 2 is used, because the information is encoded in bits.

Info(D) is also known as the **entropy of D.**

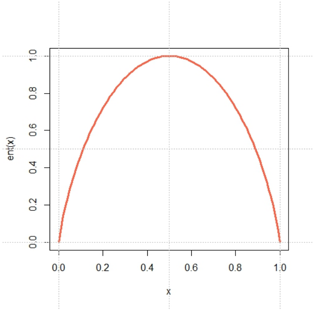
Entropy

Entropy of M&M

1. in the first box we have: **1 yellow** candy and **99 reds**;
2. in the second box we have: **50 yellows** candies and **50 reds**;
3. in the third box we have: **25 yellows** candies and **75 reds**;

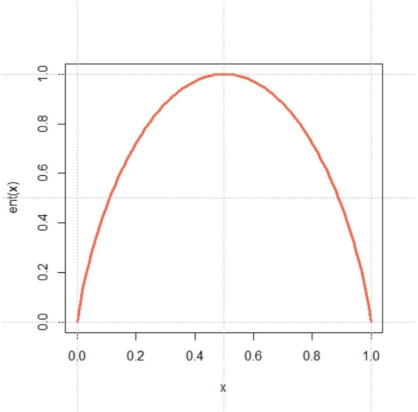
$$H(S) = -p\_red * log_2(p\_red) - p\_yellow * log_2(p\_yellow)$$

1. $-(99/100) * log_2(99/100) - (1/100) * log_2(1/100) = 0.08079314$
2. $-(50/100) * log_2(50/100) - (50/100) * log_2(50/100) = 1$
3. $-(25/100) * log_2(25/100) - (75/100) * log_2(75/100) = 0.8112781$



Entropy

```
ent<-function(p){
  ifelse(p==0 | p==1, 0, -p*log2(p)-(1-p)*log2(1-p))
}
curve(ent,0,1,col="tomato",lwd=3.5)
abline(h = 0, v = 0:2/2, lty = 3, col = "gray")
abline(v = 0, h = 0:2/2, lty = 3, col = "gray")
```



Information gain

Let **A** be variable from D.

- Let **A** has v distinct values: $a_1, a_2, ..., a_v$;
- Attribute A can be used to split D into v partitions or subsets $D_1, D_2, ..., D_v$, where D_i contains observations from a set of **D**, whose value of the variable A is a_i;
- for each value a_i a labelled edge is created a_i;

$Info_A(D)$ is the **expected information required to classify an observation from D based on the partitioning by A**.

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

The smaller the expected information required, the greater the purity of the partitions.

Information gain

Information gain **Gain(A)** is defined as the difference between the original information requirement (i.e., based on just the proportion of classes) and obtained after partitioning on A.

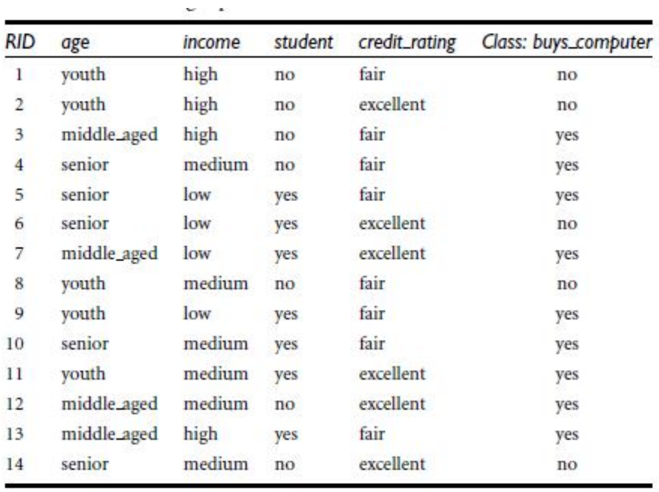
$$Gain(A) = Info(D) - Info_A(D)$$

The attribute A with the **highest** information gain, **Gain(A)**, is chosen as the splitting attribute at tree node.

Gain(A) tells us how much would be gained by branching on A.

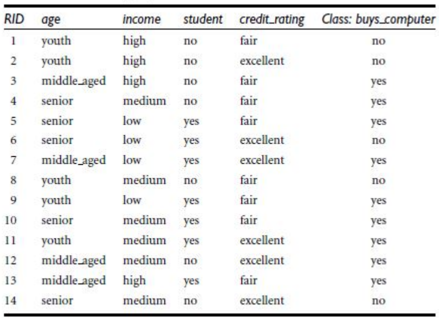
Induction of a decision tree using information gain

Let us consider



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

Induction of a decision tree using information gain



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

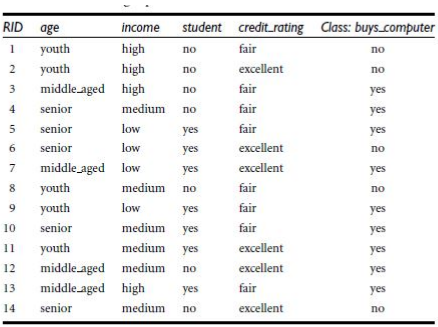
- $|D| = 14$
- Let class C_1 correspond to **yes** and class C_2 correspond to **no**
- $|C_1| = 9$, $|C_2| = 5$

Induction of a decision tree using information gain

Entropy of D

$$\begin{aligned} Info(D) &= -\sum_{i=1}^{n} p(x_i) log_2 p(x_i) \\ &= -p_1 log_2 p_1 - p_2 log_2 p_2 \\ &= -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} \\ &= 0.94 \end{aligned}$$

Induction of a decision tree using information gain



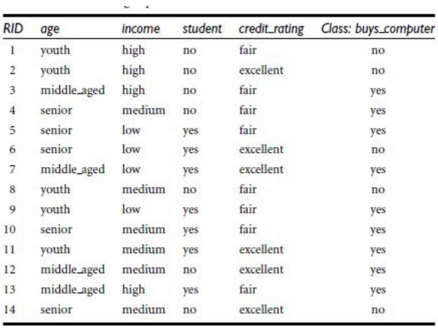
RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes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

we need to compute the expected information requirement for each attribute.

- age
- income
- student
- credit rating

Induction of a decision tree using information gain

Let's start with the attribute age. The variable age divides the set D into 3 partitions: D_1, D_2 and D_3.

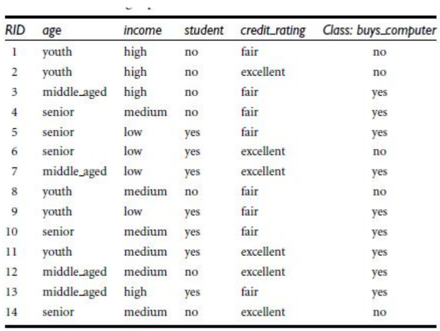


RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes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

$|D_1| = 5$, $|D_2| = 4$, $|D_3| = 5$

- for the category youth, there are two **yes** and three **no**;
- for the category middle aged, there are four **yes** and zero **no**;
- for the category senior, there are three **yes** and two **no**.

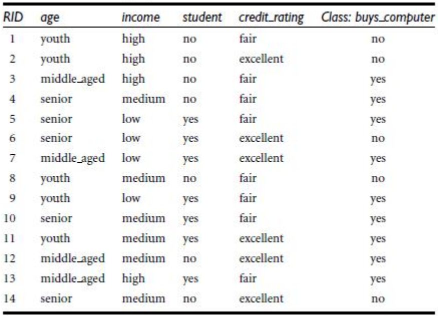
Induction of a decision tree using information gain



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

$$\begin{aligned} Info_{age}(D_1) &= -\frac{2}{5} log_2 \frac{2}{5} - \frac{3}{5} log_2 \frac{3}{5} = 0.971 \\ Info_{age}(D_2) &= 0 \\ Info_{age}(D_3) &= -\frac{3}{5} log_2 \frac{3}{5} - \frac{2}{5} log_2 \frac{2}{5} = 0.971 \end{aligned}$$

Induction of a decision tree using information gain



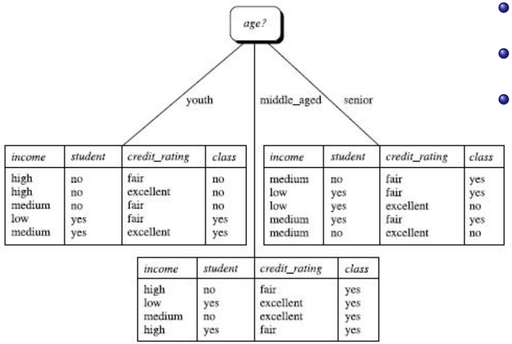
RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes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

$$\begin{aligned} Info_{age}(D) &= \frac{5}{14} \cdot 0.971 + \\ &+ \frac{4}{14} \cdot 0 \\ &+ \frac{5}{14} \cdot 0.971 \\ &= 0.693 \end{aligned}$$

The gain in information according to age

$$Gain(age) = Info(D) - Info_{age}(D) = 0.94 - 0.693 = 0.247$$

Induction of a decision tree using information gain



income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

Similarly, we can compute

- Gain(income) = 0.029
- Gain(student) = 0.151
- Gain(credit rating) = 0.048

Because age has the highest information gain among the attributes, it is selected as the splitting attribute.

Gini index

The Gini index measures the impurity of D

$$gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

gdzie $p_i = \frac{|s_i|}{n}$ is the probability that a given observation from **D** belongs to class C_i.

The Gini index considers a binary split for each attribute.

If a binary split on A partitions D into D_1 and D_2, the **gini index of D** given that partitioning is

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

Gini index

The reduction in impurity that would be incurred by a binary split on a discrete- or continuous-valued attribute A is

$$Gain_{gini}(D, A) = gini(D) - gini_A(D)$$

The attribute that maximizes the reduction in impurity $Gain_{gini}(D, A)$ is selected as the splitting attribute.

Errors made by the classifier

Errors made by the classifier can be divided into three groups

- **training error** - the ratio of incorrectly classified observations in the training set to the number of observations in the training set.(**overall**)
- **test error** - the ratio of incorrectly classified observations in the test set to the number of observations in the test set.
- **generalisation error** - classification error on a set of new observations for which the value of the decision attribute is unknown.

The default assumption is that minimising test error leads to minimising generalisation error.

Pruning

The recursive partitioning process of building the tree, continues indefinitely until it encounters a stopping criterion. Conditions, which signals the partitioning process to stop is

- when all of the instances within a partition are of the same class;
- when all the features in the dataset have been exhausted.

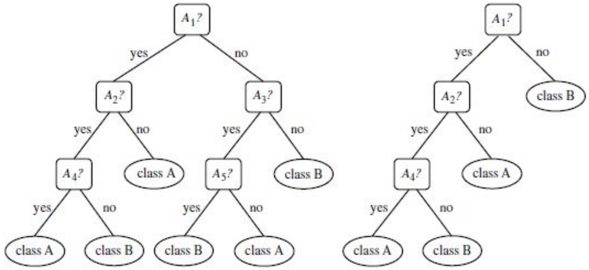
If the tree is allowed to grow uninhibited until it meets one or both of these criteria, it may already be too large and **overfit against the training data.**

To avoid this, the size of a decision tree is often **reduced** during or after the growth process for it to generalize better against unseen data. This process is known as **pruning.**

Tree Pruning

There are two basic methods of pruning a tree

- **prepruning**;
- **postpruning**;



Tree Pruning - prepruning

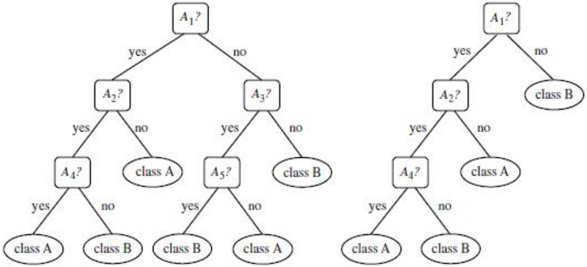
Prepruning is done during the building of the tree

- when it is decided that there are no further splits;
- a sufficient number of observations are left in a given vertex;

When interrupted, the vertex becomes a leaf, which is assigned the most frequent decision class.

Tree Pruning - postpruning

Postpruningis done after the entire tree has been built. The removed subtree is converted into a leaf, which is assigned the most frequent decision class.



Thank you for your attention!!!