MONASH University

MONASH
INFORMATION
TECHNOLOGY

Machine Learning: Classification Techniques

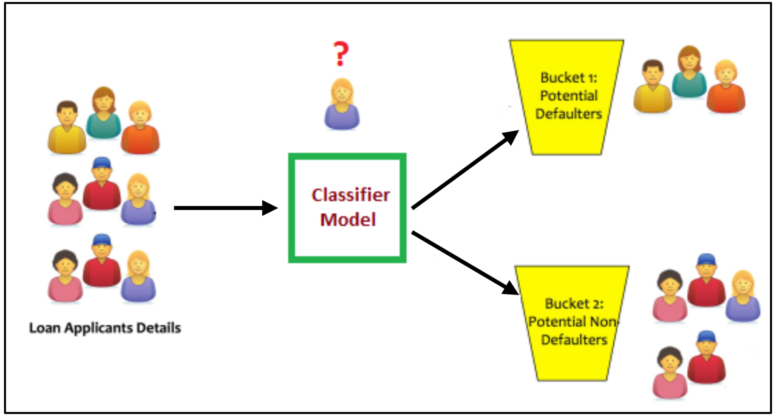
GROUP OF EIGHT AUSTRALIA

Last week

- Data Transformer, Estimators, Pipelines
- Feature Selection and Extraction

Classification

- Predictive Data Modelling
- Training: A classifier model needs to be created using the training dataset
- Testing: After the classifier is created, classification is the process of assigning new instances from the testing dataset to predefined classes
- The label for each class is predefined

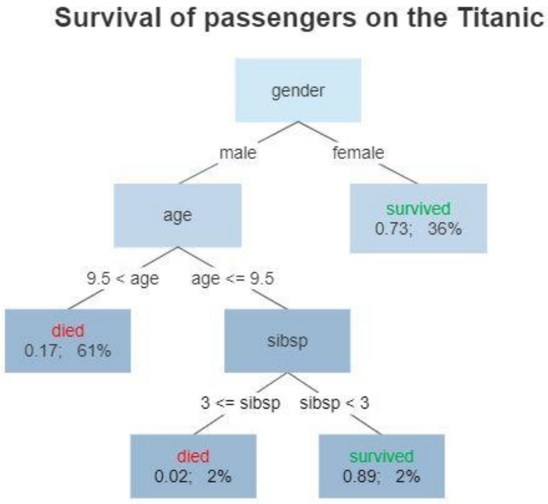


Decision Trees

- A tree-like predictive model for decision making
- In DTs, a record/sample which falls into a certain class or category is identifiable through its features/attributes.
- It splits samples (use if-then rule) into two or more homogeneous sets (leaves) based on the most significant attributes (predictors)

Homogeneous = all samples belong to same class

Samples	Features/Attributes			Class
	gender	age	sibsp	
Person 1	male	30	1	**died**
Person 2	female	20	2	**survived**

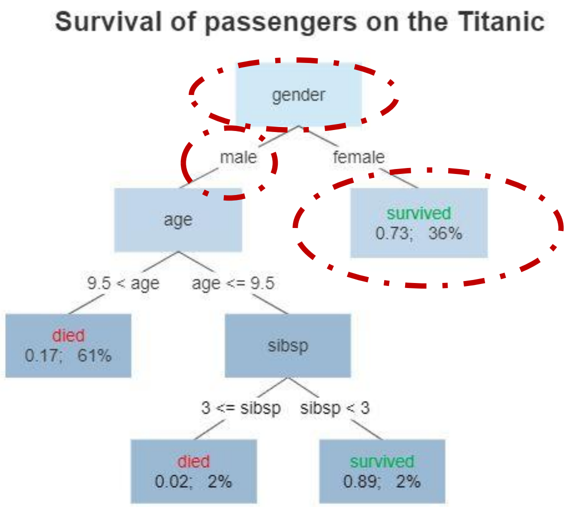


Example: Titanic dataset

DT: a hierarchy of conditional control statements

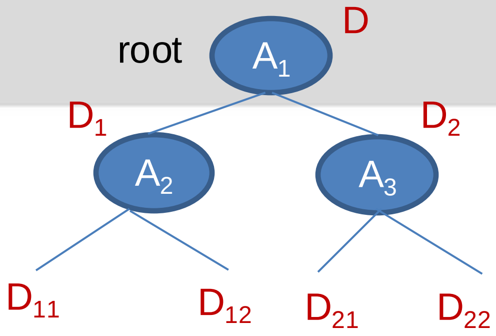
Decision Trees

- Each **internal node** represents a "test" on an attribute (e.g. gender)
- The first node is called a **Root Node**
- Each **branch** corresponds to attribute values (outcome of the test) – e.g. male or female
- Each **leaf/terminal node** assigns class label (e.g., died or survived)



Example: Titanic dataset

Decision Tree Algorithm (Training)



Supervised Learning – need output labels to build a DT

Constructing a DT is generally a recursive process

- Initialization: All training data at the root node
- Partition training data recursively by choosing one attribute at a time
- Repeat process for partitioned dataset
- Stopping criteria: When all training data in each partition have same target class

The most common approach in building a decision tree:

ID3 (Iterative Dichotomiser 3) → uses ***Entropy function*** and ***Information gain*** as metrics to construct a DT.

Entropy & IG = criteria used to determine features used in splitting the data

Entropy

- Measure of uncertainty or randomness in data
- Informs the predictability of an event
  - Low value -> Less uncertainty, high value -> high uncertainty

Less homogeneous (high uncertainty)

Play Basketball	
Yes	No
9	5

$$H(S) = \sum_{i=1}^{n} p_i \log \frac{1}{p_i}$$

p_i - Probability of event *i*

n - Number of events

$$H(Play\_basketball) = p(yes)\log\frac{1}{p(yes)} + p(no)\log\frac{1}{p(no)}$$

$$= -(\frac{9}{14}\log\frac{9}{14}) - (\frac{5}{14}\log\frac{5}{14})$$

= 0.2831

If samples are completely homogeneous, the entropy is zero

Log base 10, 2 or e?

More homogeneous (less uncertainty)

Play Basketball	
Yes	No
13	1

$$H(Play\_basketball) = -(\frac{13}{14}\log\frac{13}{14}) - (\frac{1}{14}\log\frac{1}{14})$$

= 0.1115

Information Gain

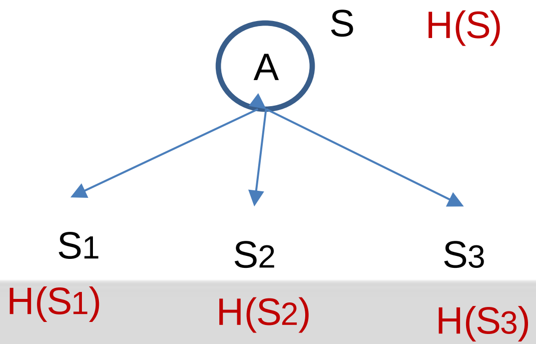
- IG for a set S is change in entropy after deciding on a attribute A.
- It computes difference between entropy before split and average entropy after split of the dataset based on an attribute A
- Used to decide which attributes are more relevant in ID3 algorithm

$$IG(S,A) = H(S) - H(S,A)$$

Weighted sum entropy given A

$$= \underbrace{H(S)}_{\text{Entropy before (on entire set A)}} - \underbrace{\sum_{i \in Values(A)} p_i H(S_i)}_{\text{Entropy after a decision based on A}}$$

S_i Subset/partition of data after splitting *S*

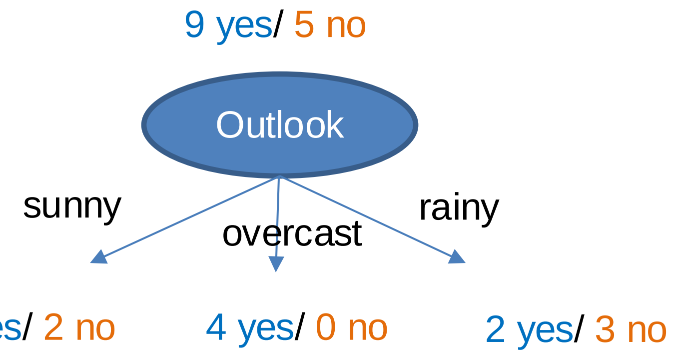


Example

$$H(S) = p(yes)\log\frac{1}{p(yes)} + p(no)\log\frac{1}{p(no)}$$
$$= 0.2831$$

		Play Basketball		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

Play Basketball	
Yes	No
9	5



$$H(S_{sunny}) = -(\tfrac{3}{5}\log\tfrac{3}{5}) - (\tfrac{2}{5}\log\tfrac{2}{5}) = 0.2922$$
$$H(S_{overcast}) = 0$$
$$H(S_{rain}) = -(\tfrac{2}{5}\log\tfrac{2}{5}) - (\tfrac{3}{5}\log\tfrac{3}{5}) = 0.2922$$

$$H(S,A) = p(sunny)H(S_{sunny}) + p(overcast)H(S_{overcast}) + p(rain)H(S_{rain})$$
$$= \frac{5}{14}(0.2922) + \frac{4}{14}(0) + \frac{5}{14}(0.2922)$$
$$= 0.2087$$

In ID3 algorithm, we select attribute with the highest gain to be the node in the tree

$$IG(Play\_basketball, outlook)$$
$$= H(Play\_basketball) - H(Play\_basketball, outlook)$$
$$= 0.2831 - 0.2087 = 0.0744$$

$$IG(S,A) = H(S) - H(S,A)$$
$$= H(S) - \sum_{i \in Values(A)} p_i H(S_i)$$

MONASH University

ID3 (Iterative Dichotomiser 3)

- It constructs DT, by finding for each node attribute that returns the highest information gain to split the data

- **Steps**

1. Compute the entropy for dataset *S* → $H(S)$

2. For every attribute/feature *A*:

   2.1. Calculate entropy for each categorical value of *A* → $H(S_i)$

   2.2 Take weighted average entropy for the current attribute → $H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$

   2.3 Calculate IG for the current attribute → $IG(S,A) = H(S) - H(S,A)$

3. Pick the attribute with highest IG to be a node, and split dataset by its branch to child nodes/subsets

4. Repeat same process at every child node until the tree is complete

Stopping condition: when data in each partition have same target class

Decision Trees: To Jog or Not To Jog

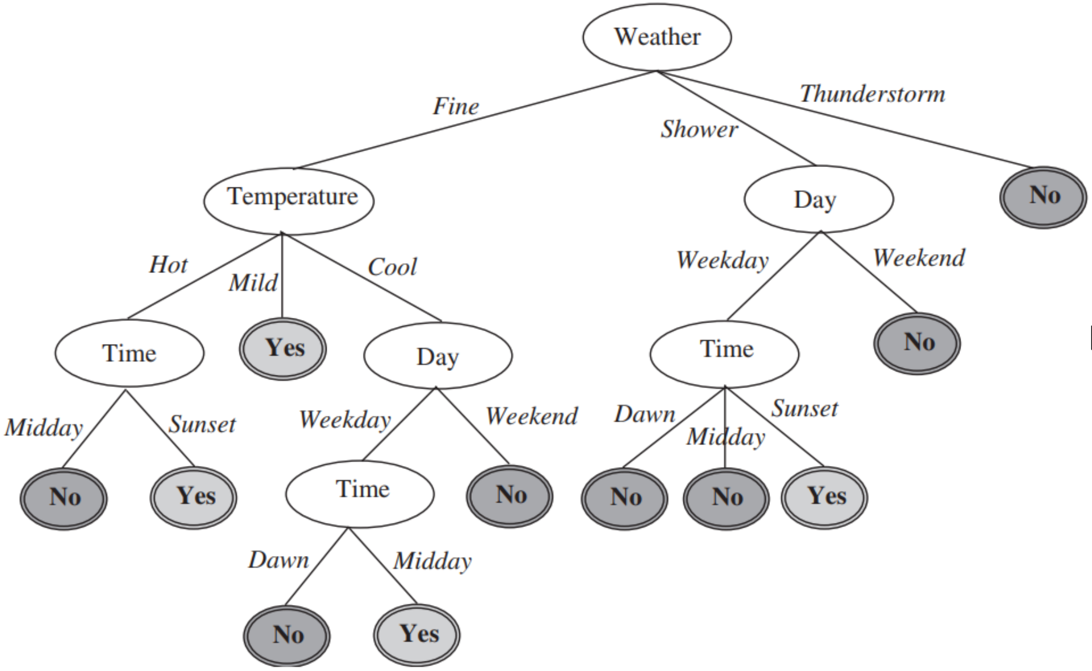


Figure 17.10 A decision tree

- A decision tree is constructed based on given training dataset.

ID3

Example:

Consider data collected over the course of 15 days

Features/Attributes: Weather, Temperature, Time, Day

Outcome variable: whether Jogging was done on the day.

Problem: to build a predictive model that takes in the above 4 parameters and predicts whether Jogging will be done on the day.

We'll build a decision tree to do that using the **ID3 algorithm.**

Samples Features/Attributes Output/Target

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

ID3 (Iterative Dichotomiser 3)

- It constructs DT, by finding for each node attribute that returns the highest information gain to split the data

- **Steps**

1. Compute the entropy for dataset S ⇒ $H(S)$
2. For every attribute/feature A:
   - 2.1. Calculate entropy for each categorical value of A ⇒ $H(S_i)$
   - 2.2 Take weighted average entropy for the current attribute ⇒ $H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$
   - 2.3 Calculate IG for the current attribute ⇒ $IG(S,A) = H(S) - H(S,A)$
3. Pick the attribute with highest IG to be a node, and split dataset by its branch to child nodes/subsets
4. Repeat same process at every child node until the tree is complete

Stopping condition: when data in each partition have same target class

ID3

Entropy for the given probability of the target classes, $p_1, p_2, \ldots, p_n$ where $\sum_{i=1}^{n} p_i = 1$, can be calculated as follows:

$$entropy(p_1, p_2, \ldots, p_n) = \sum_{i=1}^{n} \left(p_i \log(1/p_i)\right) \tag{17.2}$$

$$\begin{aligned} entropy(Yes, No) &= 5/15 \times \log(15/5) + 10/15 \times \log(15/10) \\ &= 0.2764 \end{aligned} \tag{17.3}$$

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

- Step 1: Calculate entropy for the training dataset in Figure 17.11. The result is previously calculated as 0.2764 (see equation 17.3).

Jog	
Yes	No
5	10

$$H(Yes, No) = p(yes) \log \frac{1}{p(yes)} + p(no) \log \frac{1}{p(no)}$$

ID3 (Iterative Dichotomiser 3)

- It constructs DT, by finding for each node attribute that returns the highest information gain to split the data

- **Steps**

1. Compute the entropy for dataset S → $H(S)$

2. For every attribute/feature A:

   2.1. Calculate entropy for each categorical value of A → $H(S_i)$

   2.2 Take weighted average entropy for the current attribute → $H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$

   2.3 Calculate IG for the current attribute → $IG(S,A) = H(S) - H(S,A)$

3. Pick the attribute with highest IG to be a node, and split dataset by its branch to child nodes/subsets

4. Repeat same process at every child node until the tree is complete

Stopping condition: when data in each partition have same target class

ID3

$$H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$$

$$IG(S,A) = H(S) - H(S,A)$$

entropy(Weather=*Fine*) = 4/7×log(7/4) + 3/7×log(7/3)
= 0.2966 (17.4)

entropy(Weather=*Shower*) = 1/4×log(4/1) + 3/4×log(4/3)
= 0.2442 (17.5)

- Step 2: Process attribute *Weather*
  - Calculate weighted sum entropy of attribute *Weather*:
    entropy(*Fine*) = 0.2966
    entropy(*Shower*) = 0.2442
    entropy(*Thunderstorm*)=0 + 4/4×log(4/4) = 0
    weighted sum entropy(*Weather*) = 0.2035
  - Calculate information gain for attribute *Weather*:
    gain (*Weather*) = 0.0729

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

		Jog		
		Yes	No	
Weather	Fine	4	3	7
	Shower	1	3	4
	Thunderstorm	0	4	4
				15

ID3

Weighted sum entropy (Weather) =
p(fine) H(fine) + p(shower) H(shower) + p(storm) H(storm)

$$H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$$

$$IG(S,A) = H(S) - H(S,A)$$

Weighted sum entropy (*Weather*) = Weighted entropy (*Fine*)
+ Weighted entropy (*Shower*)
+ Weighted entropy (*Thunderstorm*)

= 7/15 × 0.2966 + 4/15 × 0.2442 + 4/15 × 0
= 0.2035

(17.6)

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

- Step 2: Process attribute *Weather*
  - Calculate weighted sum entropy of attribute *Weather*:
    entropy(*Fine*) = 0.2966
    entropy(*Shower*) = 0.2442
    entropy(*Thunderstorm*)=0 + 4/4×log(4/4) = 0
    weighted sum entropy(*Weather*) = 0.2035
  - Calculate information gain for attribute *Weather*:
    gain (*Weather*) = 0.0729

		Jog		
		Yes	**No**	
Weather	**Fine**	4	3	7
	Shower	1	3	4
	Thunderstorm	0	4	4
				15

ID3

$$H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$$

$$IG(S,A) = H(S) - H(S,A)$$

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

gain(*Weather*) = *entropy*(training dataset *D*) – *entropy*(attribute *Weather*)
= 0.2764 – 0.2035
= 0.0729
(17.7)

- Step 2: Process attribute *Weather*
  - Calculate weighted sum entropy of attribute *Weather*:
    entropy(*Fine*) = 0.2966 (equation 17.4)
    entropy(*Shower*) = 0.2442 (equation 17.5)
    entropy(*Thunderstorm*)=0 + 4/4×log(4/4) = 0
    weighted sum entropy(*Weather*) = 0.2035 (equation 17.6)
  - Calculate information gain for attribute *Weather*:
    gain (*Weather*) = 0.0729 (equation 17.7)

ID3

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

- Step 3: Process attribute *Temperature*
  - Calculate weighted sum entropy of attribute *Temperature*:

    $entropy(Hot) = 2/5\times\log(5/2) + 3/5\times\log(5/3) = 0.2923$

    $entropy(Mild) = entropy(Hot)$

    $entropy(Cool) = 1/5\times\log(5/1) + 4/5\times\log(5/4) = 0.2173$

    $weighted\ sum\ entropy(Temperature) = 5/15\times0.2923 + 5/15\times0.2173$

    $= 0.2674$

  - Calculate information gain for attribute *Temperature*:

    $gain\ (Temperature) = 0.2764 - 0.2674 = 0.009$

		Jog		
		Yes	**No**	
Temperature	**Hot**	2	3	5
	Mild	3	2	5
	Cool	1	4	5
				15

ID3

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

- Step 4: Process attribute *Time*
  - Calculate weighted sum entropy of attribute *Time*:
    entropy(*Dawn*) = 0 + 5/5×log(5/5) = 0
    entropy(*Midday*) = 2/6×log(6/2) + 4/6×log(6/4) = 0.2764
    entropy(*Sunset*) = 3/4×log(4/3) + 1/4×log(4/1) = 0.2443
    weighted sum entropy (*Time*) = 0 + 6/15×0.2764 + 4/15×0.2443 = 0.1757
  - Calculate information gain for attribute *Time*:
    gain (*Time*) = 0.2764 – 0. 1757= 0.1007

		Jog		
		Yes	No	
Time	Dawn	0	5	5
	Midday	2	4	6
	Sunset	3	1	4
				15

ID3

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

Step 5: Process attribute *Day*

- Calculate weighted sum entropy of attribute *Day*:

  $entropy(Weekday) = 4/10 \times \log(10/4) + 6/10 \times \log(10/6) = 0.2923$

  $entropy(Weekend) = 1/5 \times \log(5/1) + 4/5 \times \log(5/4) = 0.2173$

  $weighted\ sum\ entropy\ (Day) = 10/15 \times 0.2923 + 5/15 \times 0.2173 = 0.2674$

- Calculate information gain for attribute *Day*:

  $gain(Day) = 0.2764 — 0.2674 = 0.009$

Corrections: IG(day) = 0.0341

		Jog		
		Yes	No	
Day	Weekday	4	6	10
	Weekend	1	4	5
				15

ID3 (Iterative Dichotomiser 3)

- It constructs DT, by finding for each node attribute that returns the highest information gain to split the data

- **Steps**

1. Compute the entropy for dataset *S* ⇒ $H(S)$

2. For every attribute/feature *A*:

   2.1. Calculate entropy for each categorical value of *A* ⇒ $H(S_i)$

   2.2 Take weighted average entropy for the current attribute ⇒ $H(S,A) = \sum_{i \in Values(A)} p_i H(S_i)$

   2.3 Calculate IG for the current attribute ⇒ $IG(S,A) = H(S) - H(S,A)$

3. Pick the attribute with highest IG to be a node, and split dataset by its branch to child nodes/subsets

4. Repeat same process at every child node until the tree is complete

Stopping condition: when data in each partition have same target class

ID3

gain(whether) = 0.0729
gain(Temperature) = 0.009
gain(Time) = 0.1007
gain(Day) = 0.0341

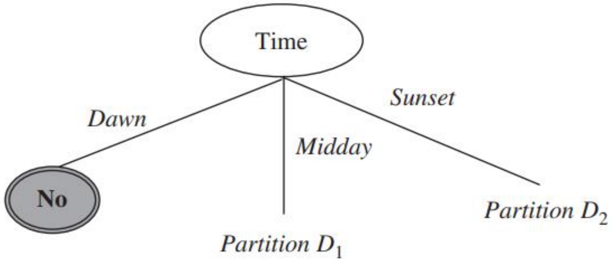


Figure 17.13 Attribute *Time* as the root node

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

Comparing equations 17.7, 17.8, 17.9, and 17.10 ,and 17.10 for the gain of each other attributes (Weather, Temperature, Time, and Day), the biggest gain is *Time*, with gain value = 0.1007 (see equation 17.9), and as a result, attribute *Time* is chosen as the first splitting attribute. A partial decision tree with the root node *Time* is shown in Figure 17.13.

ID3

Jog	
Yes	No
2	4

- The next stage is to process partition D_1 consisting of records with Time=*Midday*.
- Training dataset partition D_1 consists of 6 records with record#: 3, 6, 8, 9, 10, and 15.
- The next task is to determine the splitting attribute for partition D_1, whether it is *Weather*, *Temperature*, or *Day*.

		Jog		
		Yes	No	
Day	Weekend	0	0	0
	Weekday	2	4	6
				6
		Yes	No	
Weather	Fine	2	1	3
	Shower	0	1	1
	Thunderstorm	0	2	2
				6
		Yes	No	
Temperature	Hot	0	2	2
	Mild	1	1	2
	Cool	1	1	2
				6

Step 1: Calculate entropy for the training dataset partition D_1.

$$entropy(D_1) = 2/6\log(6/2) + 4/6\log(6/4) = 0.2764 \qquad (17.11)$$

Step 2: Process attribute *Weather*

- Calculate weighted sum entropy of attribute *Weather*

  $entropy(Fine) = 2/3 \times \log(6/2) + 1/3 \times \log(3/1) = 0.2764$

  $entropy(Shower) = entropy(Thunderstorm) = 0$

  $weighted\ sum\ entropy\ (Weather) =$ 3/6 x 0.2764 = 0.1382

- Calculate information gain for attribute *Weather*:

$$gain(Weather) = 0.2764 - 0.1382 = 0.1382 \qquad (17.12)$$

Step 3: Process attribute *Temperature*

- Calculate weighted sum entropy of attribute *Temperature*

  $entropy(Hot) = 0$

  $entropy(Mild) = entropy(Cool) = 1/2 \times \log(2/1) + 1/2 \times \log(2/1) = 0.3010$

  $weighted\ sum\ entropy\ (Temperature) = 2/6 \times 0.3010 + 2/6 \times 0.3010 = 0.2006$

- Calculate information gain for attribute *Temperature*:

$$gain(Temperature) = 0.2764—0.2006 = 0.0758 \qquad (17.13)$$

Step 4: Process attribute *Day*

- Calculate weighted sum entropy of attribute *Day*:

  $entropy(Weekday) = 2/6 \times \log(6/2) + 4/6 \times \log(6/4) = 0.2764$

  $entropy(Weekend) = 0$

  $weighted\ sum\ entropy\ (Day) = 0.2764$

- Calculate information gain for attribute *Day*:

$$gain(Temperature) = 0.2764—0.2764 = 0 \qquad (17.14)$$

The best splitting node for partition *D*1 is attribute Weather with information gain value of 0.1382 (see equation 17.12).

ID3

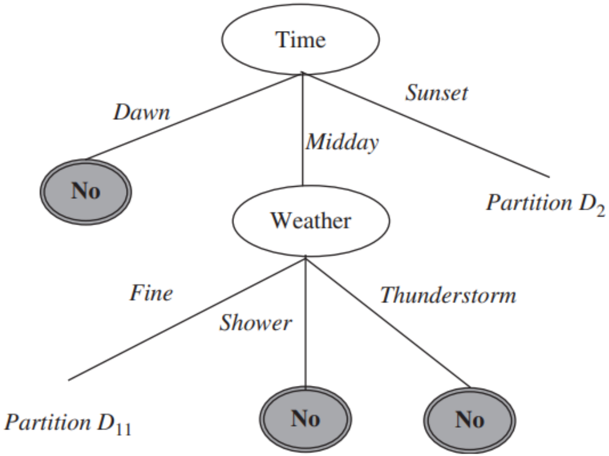


Figure 17.14 Attribute *Weather* as next splitting attribute

Rec#	Weather	Temperature	Time	Day	Jog (*Target Class*)
1	Fine	Mild	Sunset	Weekend	Yes
2	Fine	Hot	Sunset	Weekday	Yes
3	Shower	Mild	Midday	Weekday	No
4	Thunderstorm	Cool	Dawn	Weekend	No
5	Shower	Hot	Sunset	Weekday	Yes
6	Fine	Hot	Midday	Weekday	No
7	Fine	Cool	Dawn	Weekend	No
8	Thunderstorm	Cool	Midday	Weekday	No
9	Fine	Cool	Midday	Weekday	Yes
10	Fine	Mild	Midday	Weekday	Yes
11	Shower	Hot	Dawn	Weekend	No
12	Shower	Mild	Dawn	Weekday	No
13	Fine	Cool	Dawn	Weekday	No
14	Thunderstorm	Mild	Sunset	Weekend	No
15	Thunderstorm	Hot	Midday	Weekday	No

Figure 17.11. Training dataset

ID3

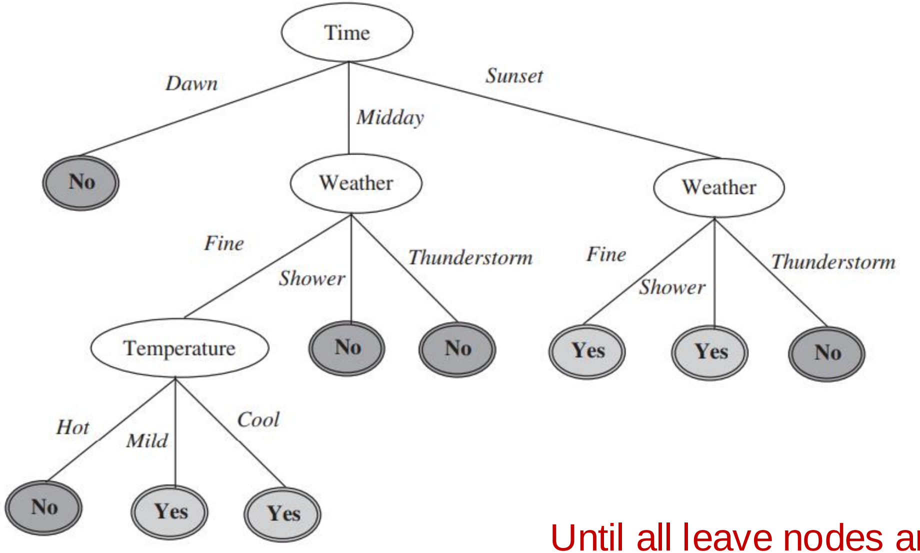


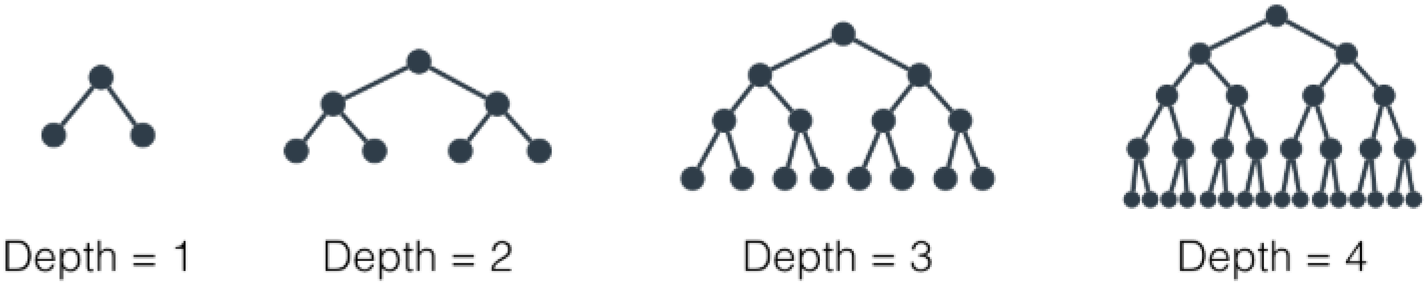
Figure 17.15 Final decision tree

Until all leave nodes are homogenous (i.e., all samples in leave node are classified to yes/no)

Maximum Depth of DT

Question: What is the potential problem if a DT is built to maximum depth on training data?

maxDepth: the largest possible length between the root to a leaf (or maximum level of the tree).



Maximum depth of a decision tree

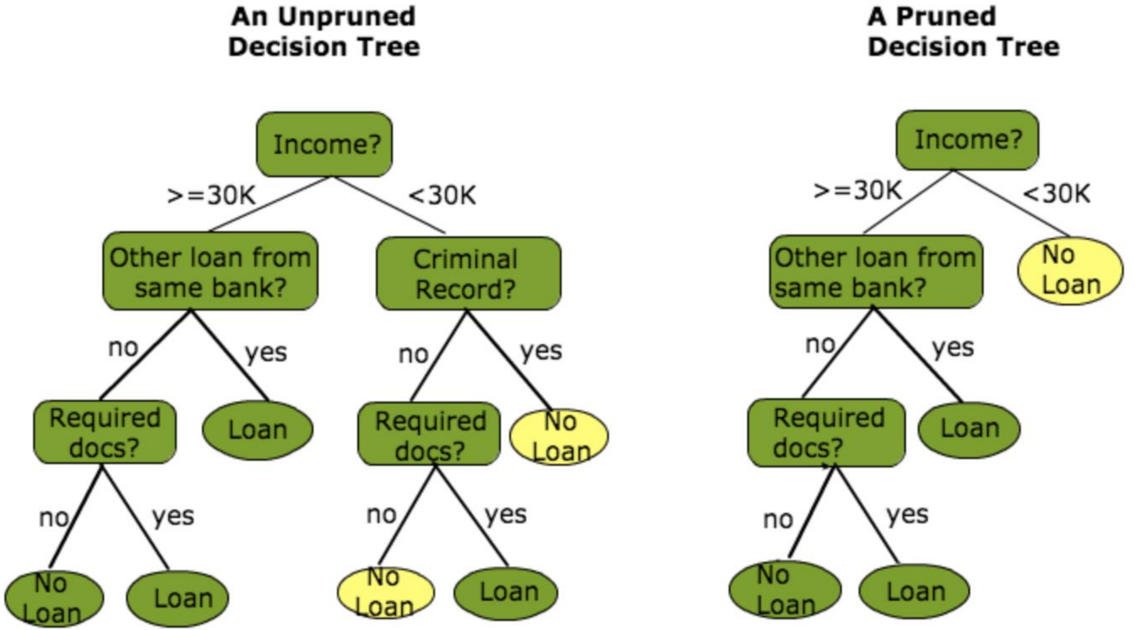
Hyperparameter: A parameter whose value is used to control the learning process and whose value cannot be estimated from data.

Decision Tree Algorithm

- **Advantages:**
  - Easy to understand.
  - Easy to generate rules.
  - There are almost null hyper-parameters to be tuned.
  - Complex Decision Tree models can be significantly simplified by its visualizations.

- **Disadvantages:**
  - Might suffer from overfitting.
  - Does not easily work with non-numerical data.
  - Low prediction accuracy for a dataset in comparison with other machine learning classification algorithms.
  - When there are many class labels, calculations can be complex.

Pruning



https://kaumadiechamalka100.medium.com/decision-tree-in-machine-learning-c610ef087260

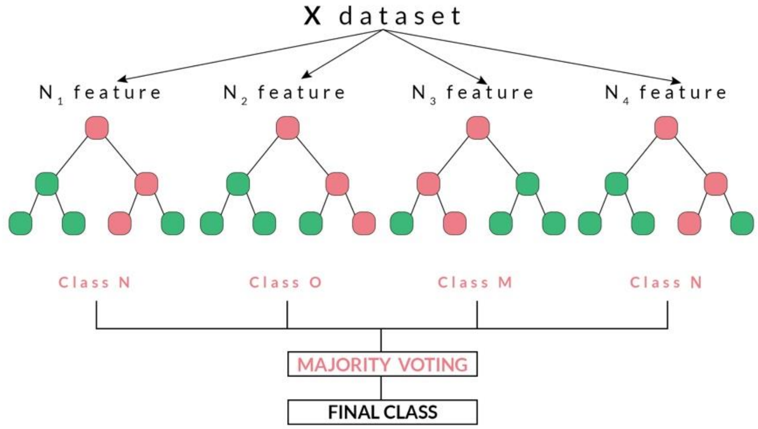
Ensemble methods

- A single decision tree have the tendency to overfit
- But it is super fast
- How about multiple trees at once?

Make sure they do not all just learn the same!

Random Forest Algorithm

- **Random forest** (or **random forests**) is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees.

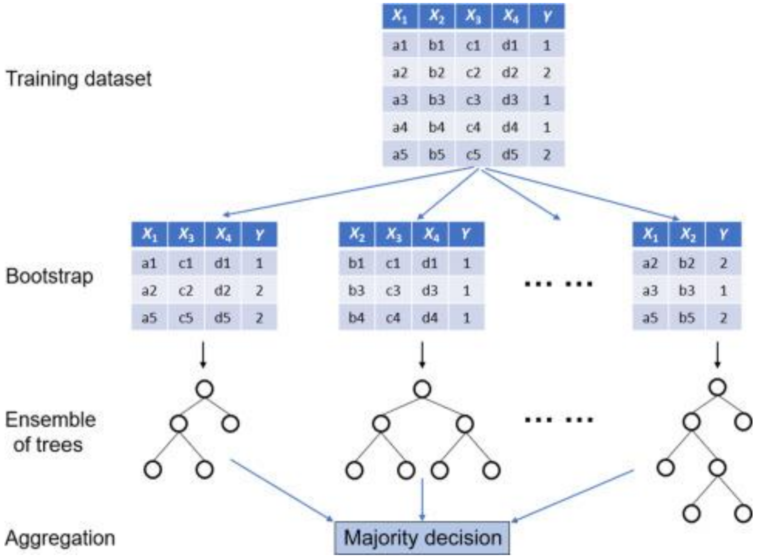


Randomness is introduced into dataset to produce different trees

Optimisations

1. Bagging: Bootstrap **agg**regat**ing** is a method that result in low variance – used to reduce variance of DTs

Rather than training each tree on all the inputs in the training set (producing multiple identical trees), each tree is trained on different set of sample data

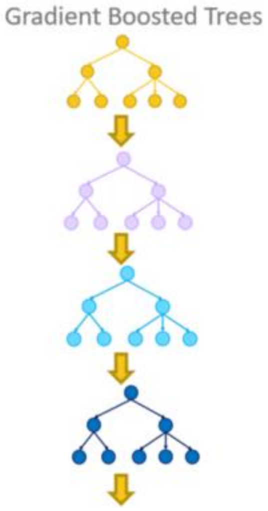


Bootstrap: use **random sampling** in terms of features or examples/records to create different subsets of data

Image source: https://www.sciencedirect.com/topics/engineering/random-forest

Optimisations

2. Gradient boosting: selecting best classifiers to improve prediction accuracy with each new tree.



- It works by combining several weak learners (typically high bias, low variance models) to produce an overall strong model.
- It builds one tree at a time, works in a forward stage-wise manner, - adding a classifier at a time, so that the next classifier is trained to improve the already trained ensemble.

Image from https://medium.com/analytics-vidhya/ensemble-models-bagging-boosting-c33706db0b0b