MONASH University

MONASH
INFORMATION
TECHNOLOGY

Machine Learning: Classification Techniques

Prajwol Sangat

GROUP OF EIGHT AUSTRALIA

Last week

Data Transformer, Estimators, Pipelines
Feature Selection and Extraction

This week

Classification Algorithms

- Decision Tree
- Random Forest
- DEMO

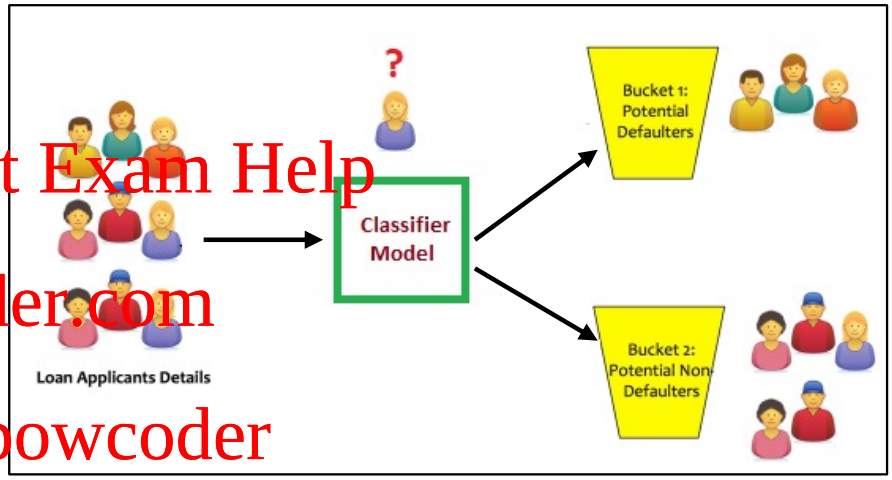
Classification

Predictive Data Modeling

A classifier model needs to be created using training dataset

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



Classification

Classifiers can be:

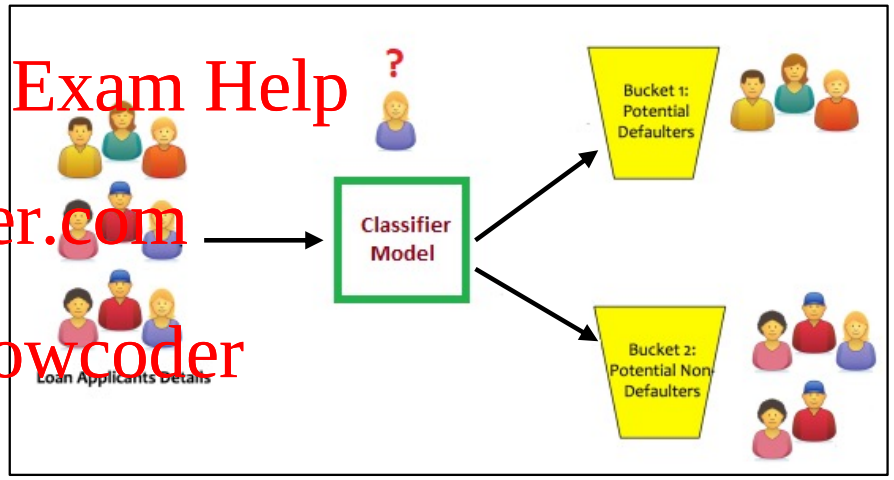
- *Binary classifier*
- *Multi-Class classifiers*

Binary classifiers: Classification with only 2 distinct classes or with 2 possible outcomes

Example: classification of spam email and non spam email, potential defaulter and non defaulter

Multi-Class classifiers: Classification with more than two distinct classes

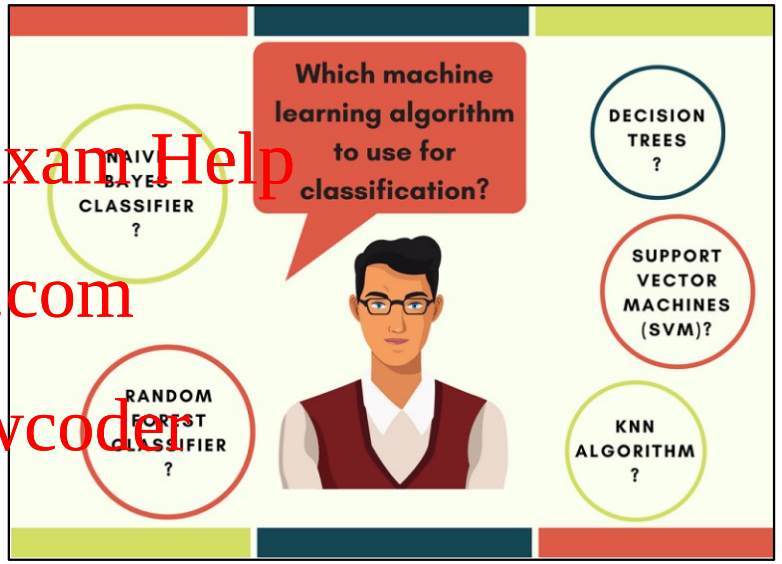
Example: classification of types of animals, classification of books into categories.



Classification Algorithms

There are several types of classification algorithms in Machine Learning:

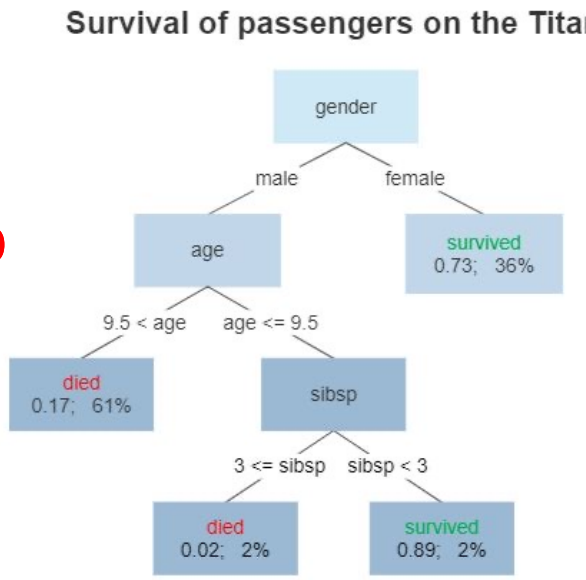
- Decision Trees
- Random Forest
- Logistic Regression
- And Many More.



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 into two or more homogeneous sets (leaves) based on the most significant attributes (predictors)

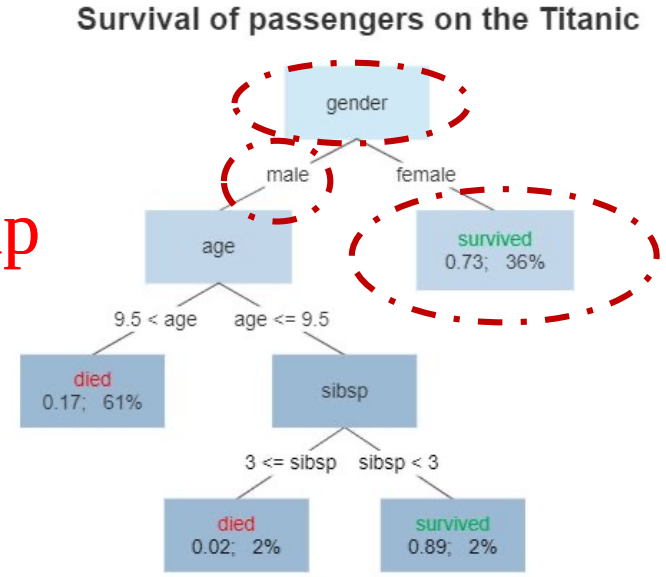
Samples	Features/Attributes			Class
	gender	age	sibsp	
Person 1	male	30	1	**died**
Person 2	female	20	2	**survived**



Example: Titanic dataset

Decision Trees

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



Example: Titanic dataset

Decision Tree Algorithm

Common terms used with Decision trees:

Root Node: It represents entire population or sample and this further gets divided into two or more homogeneous sets.

Splitting: It is a process of dividing a node into two or more sub-nodes.

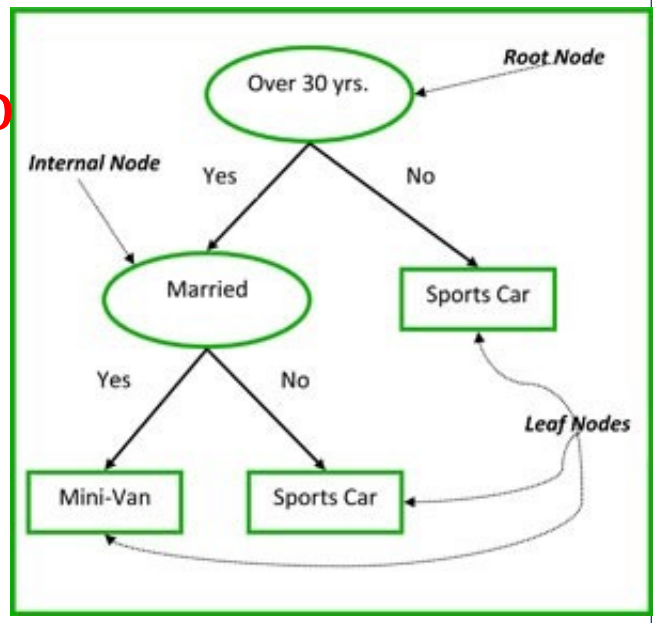
Decision Node: When a sub-node splits into further sub-nodes, then it is called decision node.

Leaf/ Terminal Node: Nodes do not split is called Leaf or Terminal node.

Pruning: When sub-nodes of a decision node is removed, this process is called pruning (an opposite process of splitting).

Branch / Sub-Tree: A sub section of entire tree is called branch or sub-tree.

Parent and Child Node: A node, which is divided into sub-nodes is called parent node of sub-nodes whereas sub-nodes are the child of parent node.



Decision Tree Algorithm

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.

ID3 (Iterative Dichotomiser 3)

ID3 (Iterative Dichotomiser 3) was developed in 1986 by Ross Quinlan.

The algorithm creates **a multiway tree**, finding for each node (i.e. in a greedy manner) the **categorical feature that will yield the largest information gain for categorical targets.**

Trees are grown to their maximum size and then a pruning step is usually applied to improve the ability of the tree to generalise to unseen data.

Entropy

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

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

p_i - Probability of event *i*

n - Number of events

Less homogeneous

Play Basketball	
Yes	No
9	5

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

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

$$= 0.2831$$

More homogeneous

Play Basketball	
Yes	No
13	1

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

$$= 0.1115$$

If samples are completely homogeneous, the entropy is zero

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

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

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

Weighted sum entropy given A

Entropy before (on entire set A)

Entropy after a decision based on A

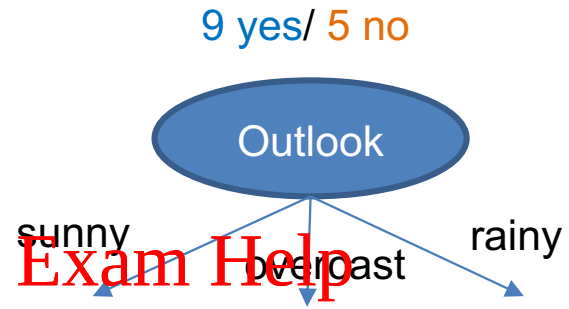
S_i Subset/partition of data after splitting *S*

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

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

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

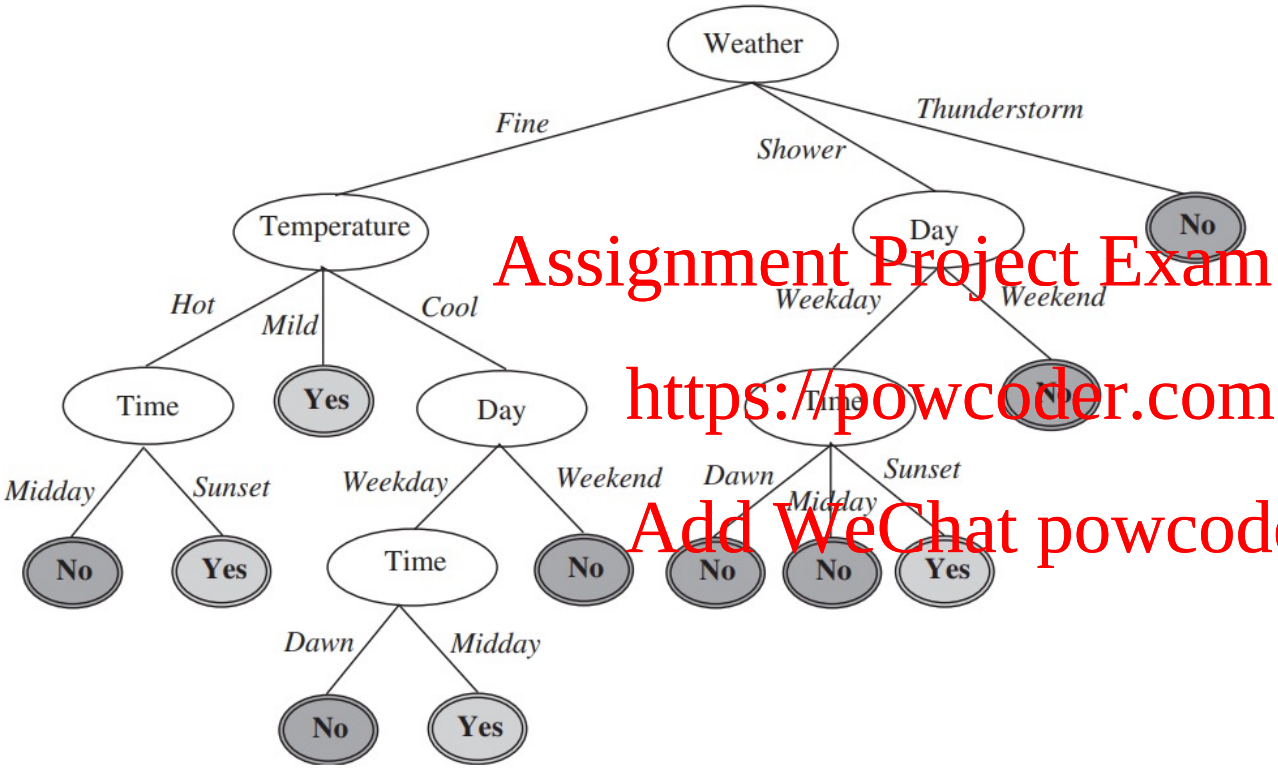


Figure 17.10 A decision tree

A decision tree is constructed based only on the given training dataset. It is not based on a universal belief.

ID3

Example: Consider a piece of data collected over the course of 15 days where the features are Weather, Temperature, Time, Day and the outcome variable is whether Jogging was done on the day. Now, our job is to build a predictive model which takes in above 4 parameters and predicts whether Jogging will be done on the day. We'll build a decision tree to do that using **ID3 algorithm.**

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, …, p_n where $\sum_{i=1}^{n} p_i = 1$, can be calculated as follows:

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

$$entropy(Yes, No) = 5/15 \times \log(15/5) + 10/15 \times \log(15/10) = 0.2764 \quad (17.3)$$

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

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

$$entropy(\text{Weather}=Fine) = 4/7\times\log(7/4) + 3/7\times\log(7/3) = 0.2966 \quad (17.4)$$

$$entropy(\text{Weather}=Shower) = 1/4\times\log(4/1) + 3/4\times\log(4/3) = 0.2442 \quad (17.5)$$

- 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*) = Weighted entropy (*Fine*)
+ Weighted entropy (*Shower*)
+ Weighted entropy (*Thunderstorm*)

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

(17.6)

- 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

$$gain(Weather) = entropy(\text{training dataset } D) - entropy(\text{attribute } Weather)$$
$$= 0.2764 - 0.2035$$
$$= 0.0729 \qquad (17.7)$$

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*:
    $entropy(Fine) = 0.2966$ (equation 17.4)
    $entropy(Shower) = 0.2442$ (equation 17.5)
    $entropy(Thunderstorm) = 0 + 4/4 \times \log(4/4) = 0$
    $weighted\ sum\ entropy(Weather) = 0.2035$ (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×log(5/2) + 3/5×log(5/3) = 0.2923
    entropy(*Mild*) = *entropy*(*Hot*)
    entropy(*Cool*) = 1/5×log(5/1) + 4/5×log(5/4) = 0.2173
    weighted sum entropy(*Temperature*) = 5/15×0.2923 + 5/15×0.2173
    = 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 (*Temperature*) = 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*:

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

		Jog		
		Yes	No	
Day	Weekend	4	6	10
	Weekday	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

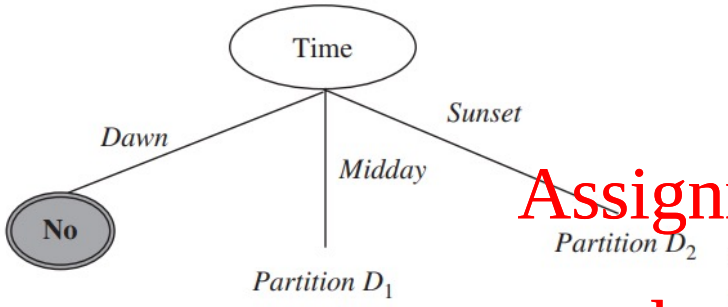


Figure 17.13 Attribute *Time* as the root node

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

		Jog		
		Yes	No	
Weather	Fine	2	1	3
	Shower	0	1	1
	Thunderstorm	0	2	2
				6

		Jog		
		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) $= 3/5 \times 0.2764 = 0.1382$
- 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

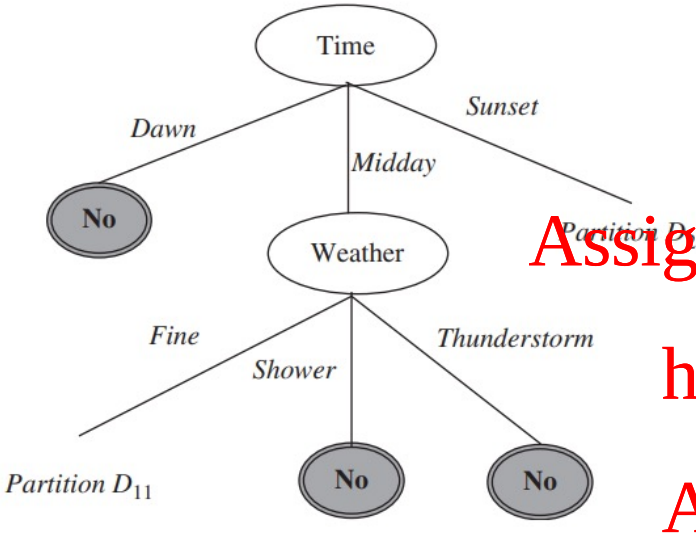


Figure 17.14 Attribute *Weather* as next splitting attribute

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

- 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*.

ID3

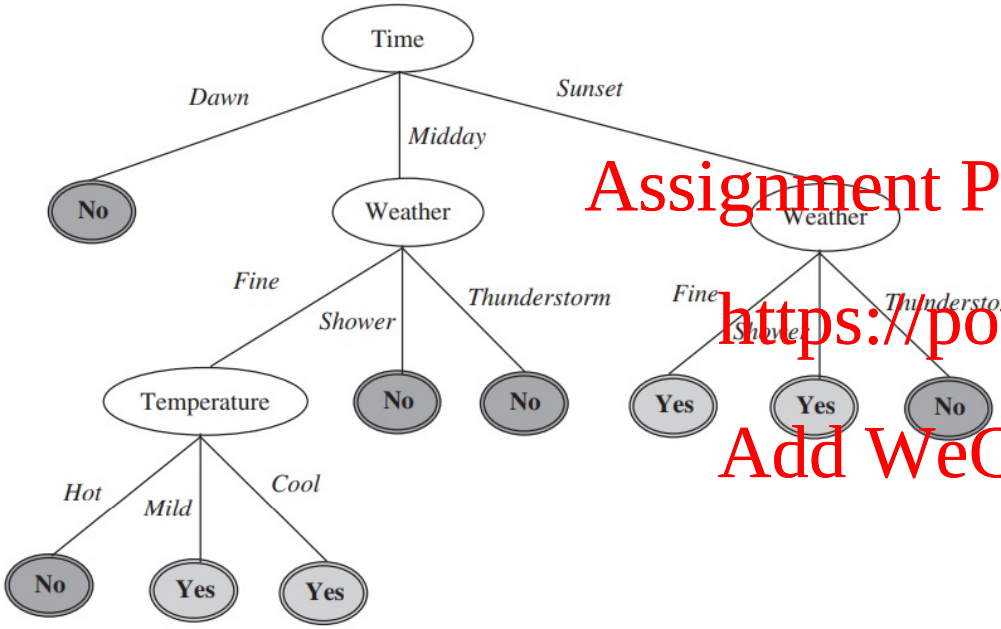


Figure 17.15 Final decision tree

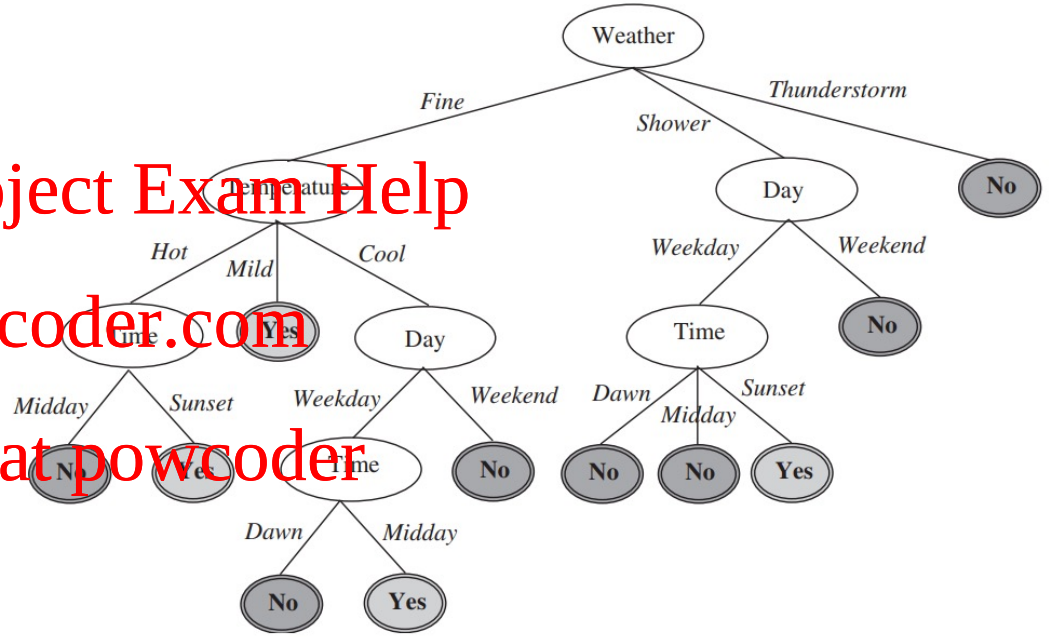


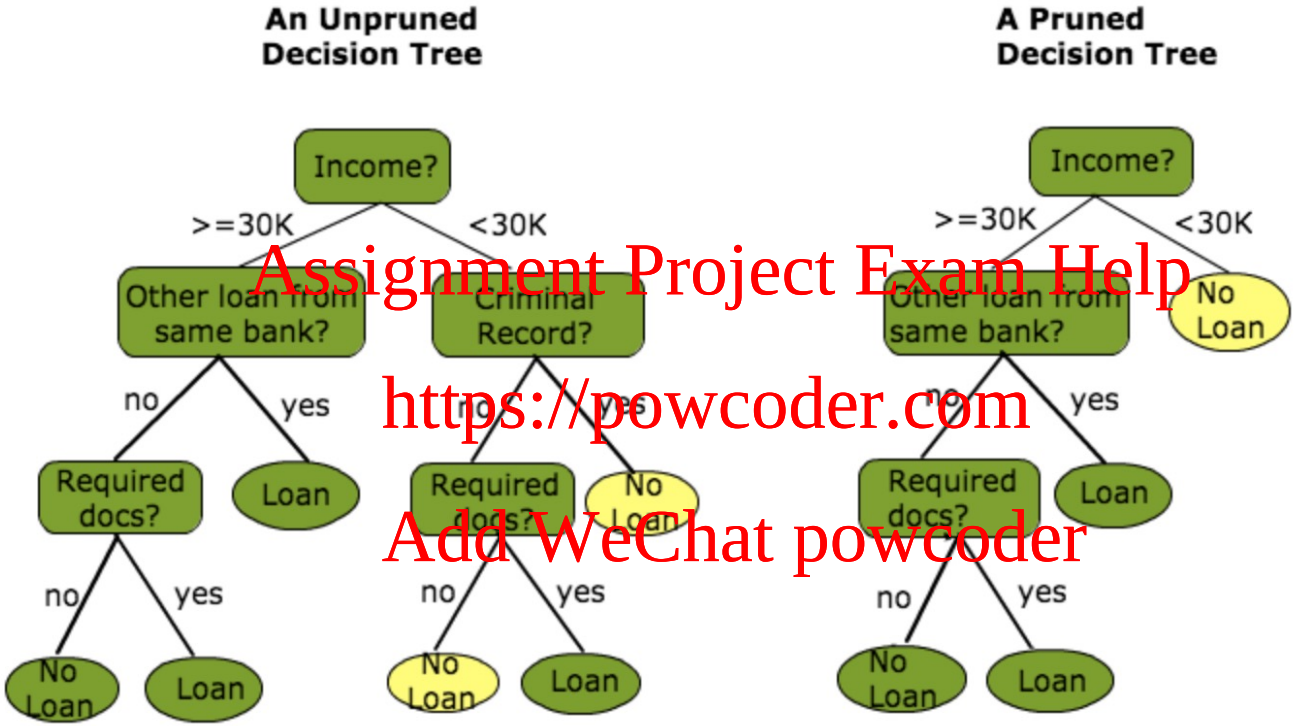
Figure 17.10 A decision tree

Maximum Depth of DT

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

Maximum depth of a decision tree

Pruning



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

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.

Ensemble methods

A single decision tree have the tendency to overfit

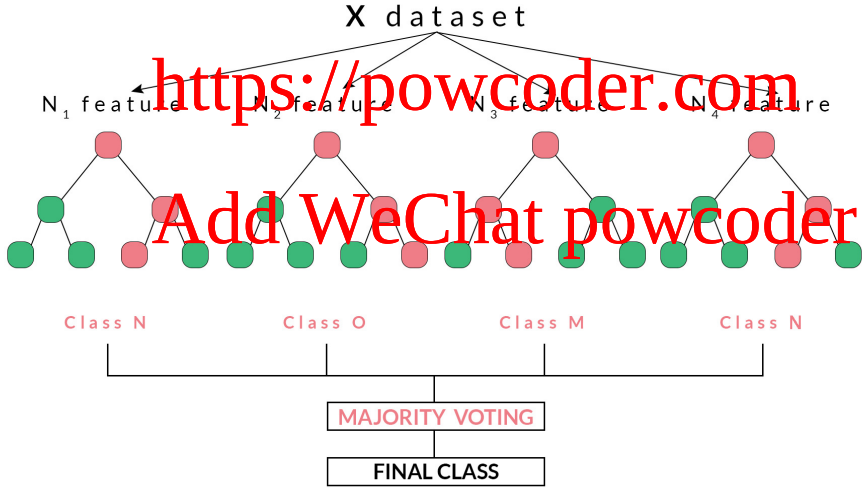
But, it is super fast

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.



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

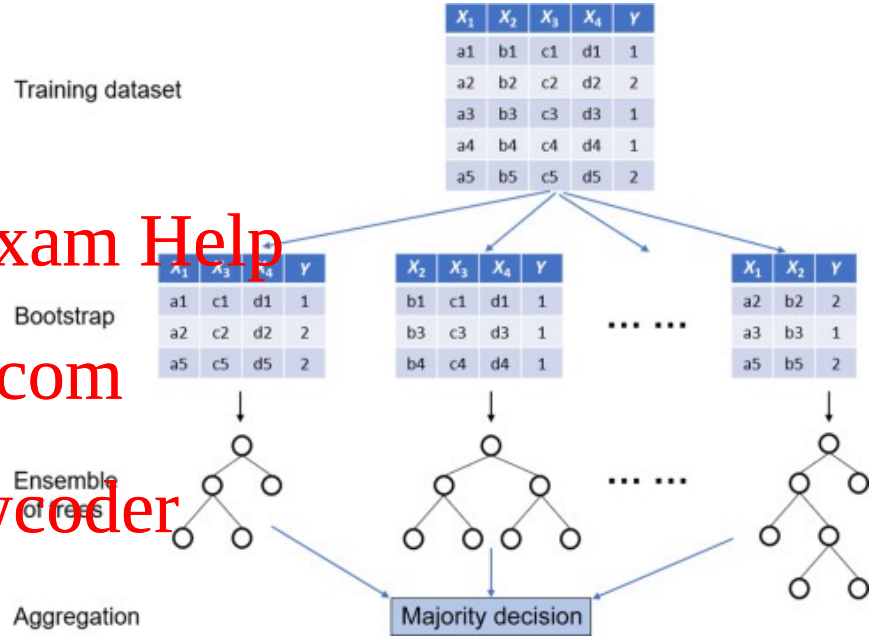
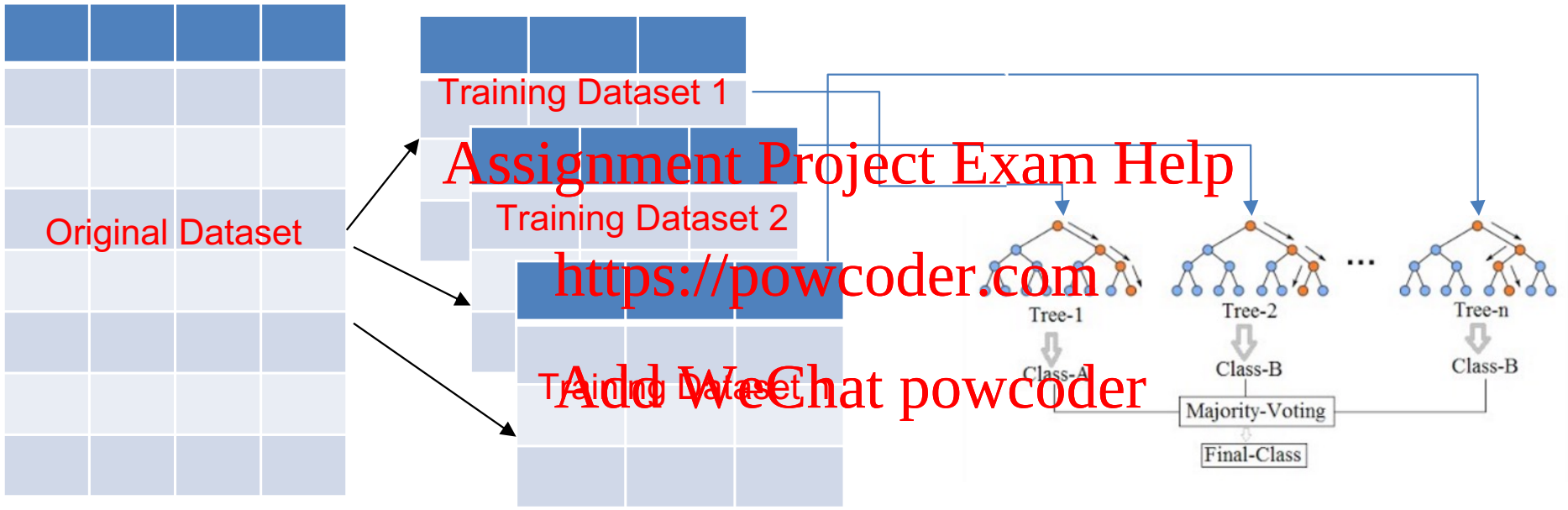


Image source:
https://www.sciencedirect.com/topics/engineering/random-forest

Example



Randomly selected entry

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 a powerful 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.

Advantages and Disadvantages of Random Forest

Advantages

- It is robust to correlated predictors.
- It is used to solve both regression and classification problems.
- It can be also used to solve unsupervised ML problems.
- It can handle thousands of input variables without variable selection.
- It can be used as a feature selection tool using its variable importance plot.
- It takes care of missing data internally in an effective manner.

Advantages and Disadvantages of Random Forest

Disadvantages

- The Random Forest model is difficult to interpret.
- It tends to return erratic predictions for observations out of range of training data. For example, the training data contains two variable x and y. The range of x variable is 30 to 70. If the test data has x = 200, random forest would give an unreliable prediction.
- It can take longer than expected time to computer a large number of trees.

What have we learnt today?

Classification techniques
Decision Trees and Random Forest and KNN
When and how to use each technique

MONASH University

MONASH
INFORMATION
TECHNOLOGY

Parallel Classification

Prajwol Sangat

GROUP OF EIGHT AUSTRALIA

Parallel Classification: Decision Tree

Data parallelism

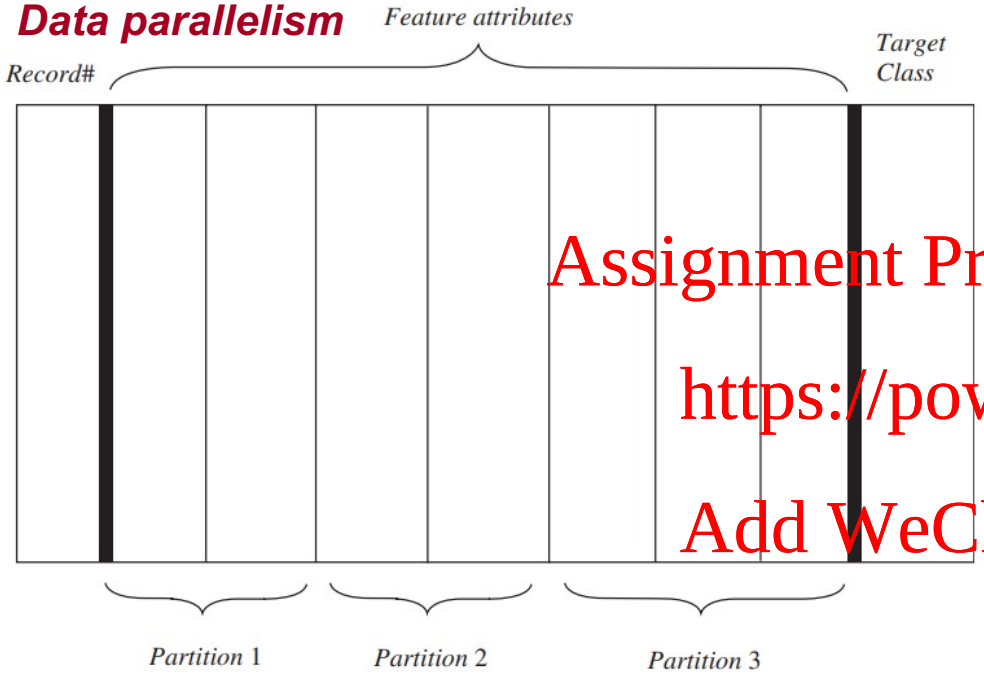


Figure 17.16 Vertical data partitioning of training data set

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

Parallel Classification: Decision Tree

Data parallelism: ***Vertical Partitioning of Training dataset***

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

Partition 1

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

Partition 2

Parallel Classification: Decision Tree

Data parallelism

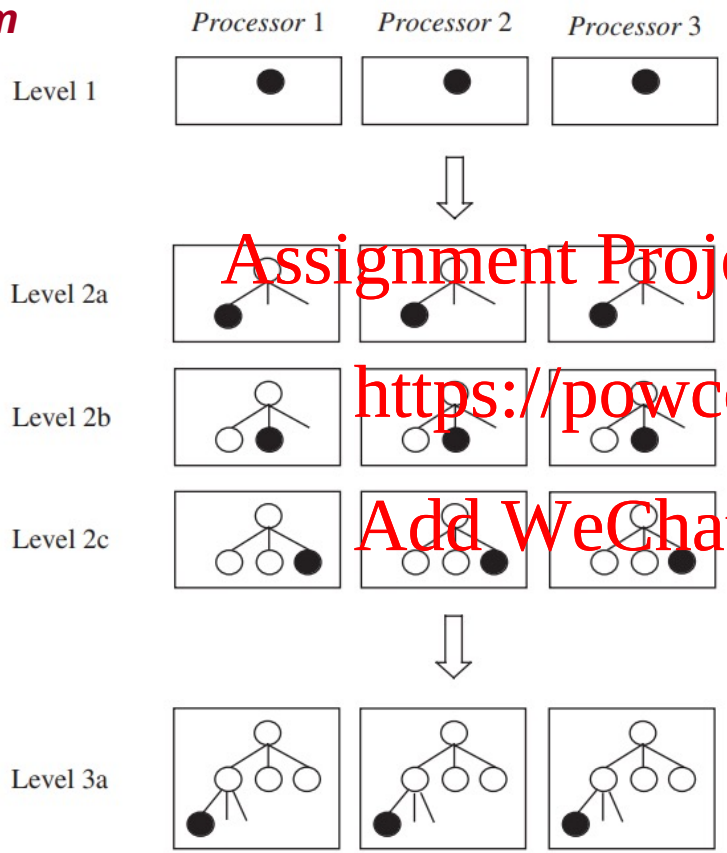


Figure 17.17 Data parallelism of parallel decision tree construction

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

Parallel Classification: Decision Tree

Level 1 (Root Node):

Data parallelism

Processor 1

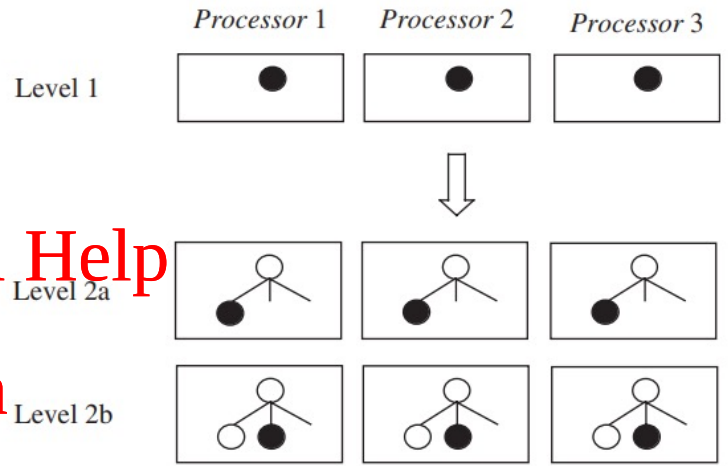
Rec#	Weather	Temperature	Target Class
1			
2			
…			
15			

Processor 2

Rec#	Time	Day	Target Class
1			
2			
…			
15			

Locally calculate the information gain values for: *Weather* and *Temperature*

Locally calculate the information gain values for: *Time* and *Day*



Global information sharing stage:

a. Share target class counts to calculate dataset entropy value
b. Exchange dataset entropy value to determine splitting attribute (e.g. Time attribute is decided to be the splitting attribute)
c. Distribute selected records# to all processor for the next phase (e.g. records 3, 6, 8, 9, 10, 15 for Time *Midday*, and records 1, 2, 5, 14 for Time *Sunset*)

Decision tree for Level 1:

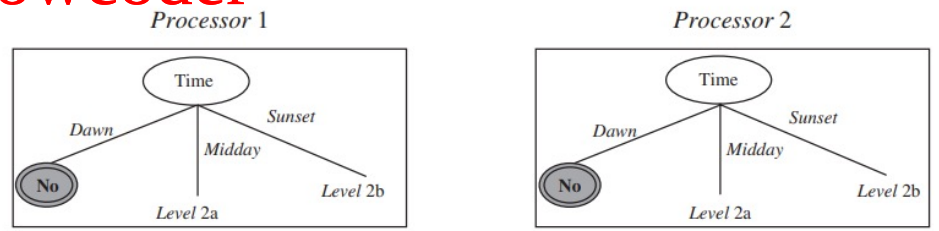


Figure 17.18 Data parallelism in decision tree

Parallel Classification: Decision Tree

Data parallelism

***Level* 2a:**

Processor 1

Rec#	Weather	Temperature	TargetClass
3			
6			
8			
9			
10			
15			

Processor 2

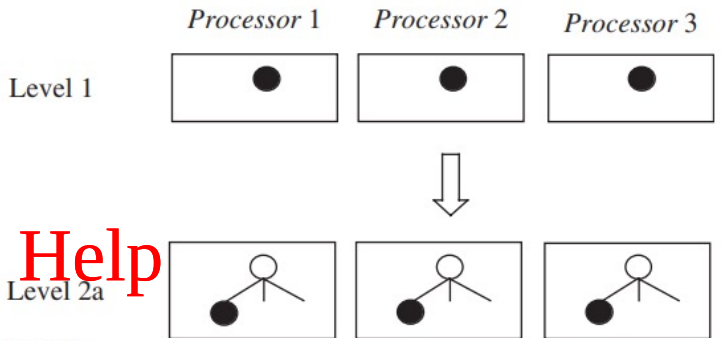
Rec#	Time	Day	TargetClass
3			
6			
9			
10			
15			

Locally calculate the information gain values for: *Weather* and *Temperature*

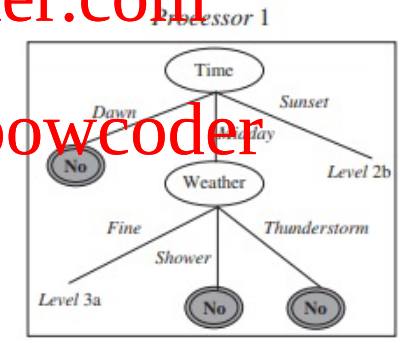
Locally calculate the information gain values for: *Day*

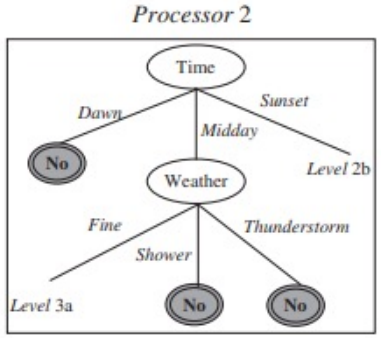
Global information sharing stage:

a. Share target class counts of each partition to calculate dataset entropy value
b. Exchange dataset entropy value to determine splitting attribute (e.g. Weather attribute is decided to be the splitting attribute)
c. Distribute selected records# to all processor for the next phase



***Result decision tree for Level* 2:**





***Level* 2b: to continue…**

Parallel Classification: Decision Tree

Result parallelism

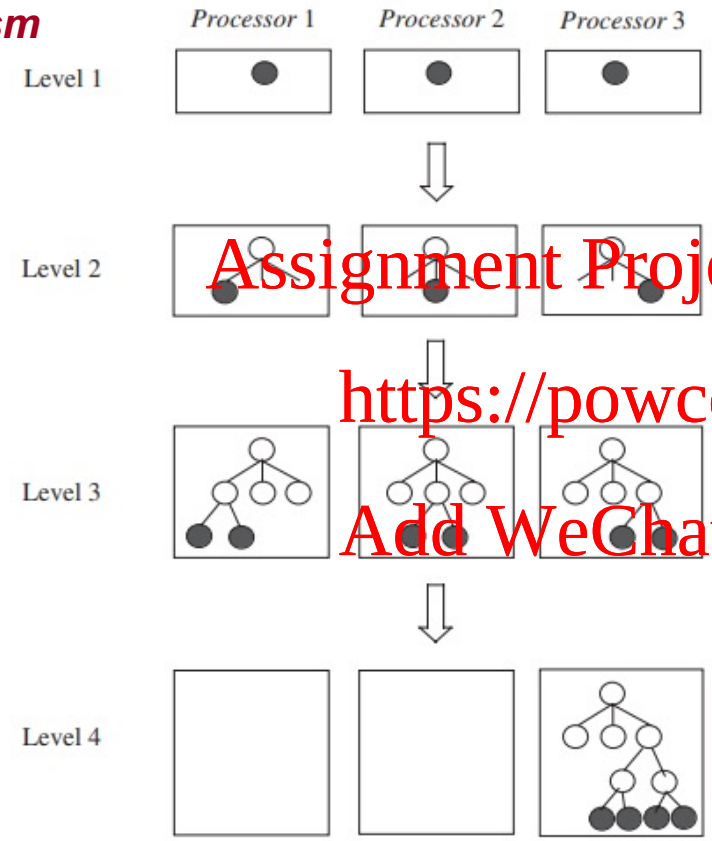


Figure 17.20 Result parallelism of parallel decision tree construction

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

Parallel Classification: Decision Tree

Result parallelism

***Horizontal Data Partitioning*:**

Processor 1

Rec#	Weather	Temp	Time	Day	Target Class
1					
2					
…					
8					

Processor 2

Rec#	Weather	Temp	Time	Day	Target Class
9					
10					
…					
15					

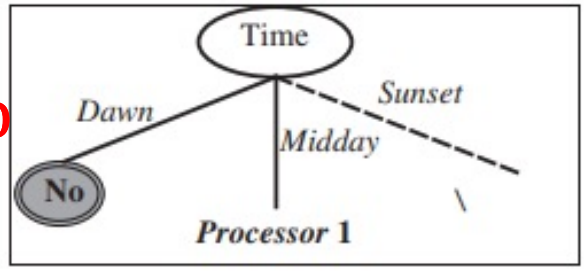
***Level* 1 (Root Node):**

a. Count target class on each partition

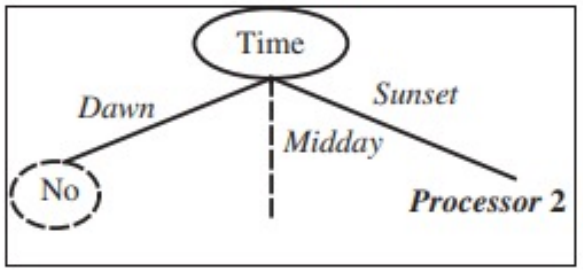
b. Perform intra-nod eparallelism the same as for data parallelism to share target class counts to calculate dataset entropy value, exchange dataset entropy value todetermine splitting attribute, and distribute selected records# to all other processors for the next phase)

***Decision tree for Level* 1:**

Processor 1



Processor 2



Parallel Classification: Decision Tree

Result parallelism

Jog	
Yes	No
5	10

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

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

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

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

Parallel Classification: Decision Tree

Result parallelism

Level 2:

Processor 1 *Processor 2*

Rec#	Weather	Temp	Time	Day	Target Class	Rec#	Weather	Temp	Time	Day	Target Class
3						1					
6						2					
8						5					
9						14					
10											
15											

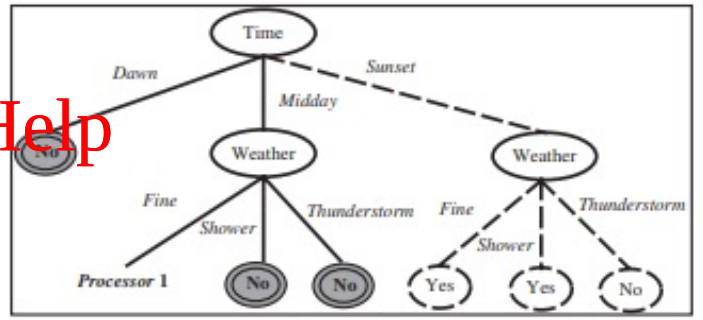
Global information sharing stage:

a. Count target class on each partition

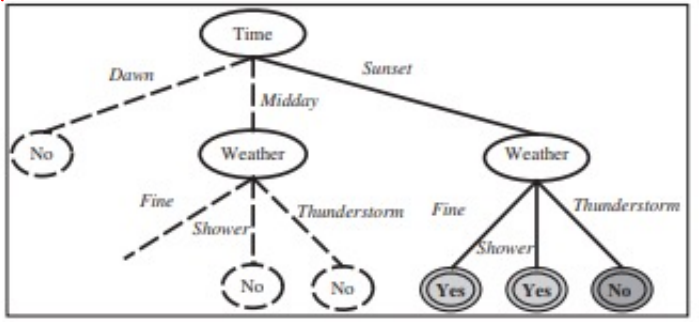
b. Perform intra-node parallelism the same as for data parallelism to share target class counts to calculate dataset entropy value, exchange dataset entropy value to determin esplitting attribute,and distribute selected records# to allother processors for the next phase)

Result decision tree for Level 2:

Processor 1



Processor 2



Parallel Classification: Decision Tree

Level 3:

Processor 1

Rec#	Weather	Temp	Time	Day	Target Class
6					
9					
10					

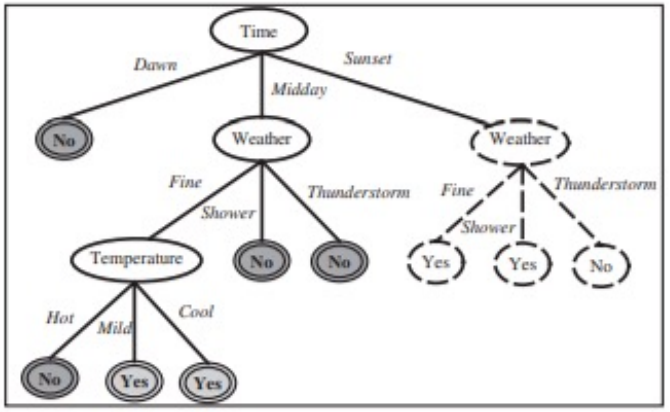
Processor 2

Rec#	Weather	Temp	Time	Day	Target Class

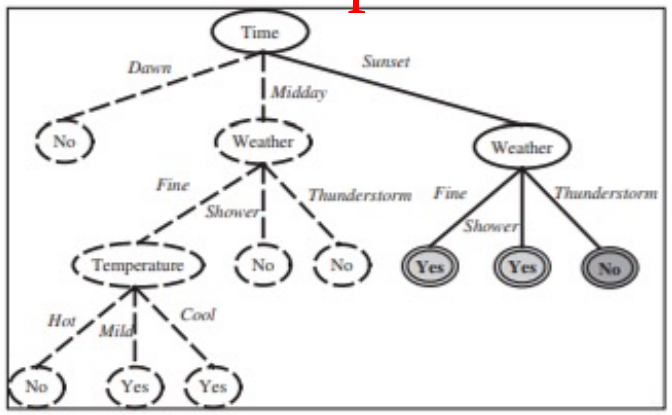
Global information sharing stage:… as like in Level 2

Result decision tree for Level 3:

Processor 1



Processor 2



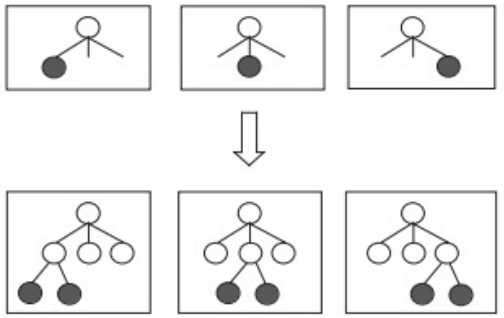


Figure 17.20 Result parallelism of parallel decision tree construction

See you next week..

Questions???