Information-based Learning
Sections 4.1, 4.2, 4.3

John D. Kelleher and Brian Mac Namee and Aoife D’Arcy

1. **Big Idea**

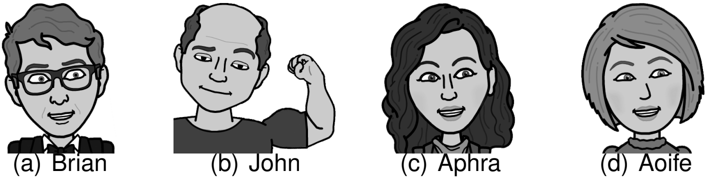
2. **Fundamentals**
   - Decision Trees
   - Shannon’s Entropy Model
   - Information Gain

3. **Standard Approach: The ID3 Algorithm**
   - A Worked Example: Predicting Vegetation Distributions

4. **Summary**

- In this chapter we are going to introduce a machine learning algorithm that tries to build predictive models **using only the most informative features**.
- In this context an informative feature is a **descriptive feature** whose values split the instances in the dataset into **homogeneous sets** with respect to the target feature value.

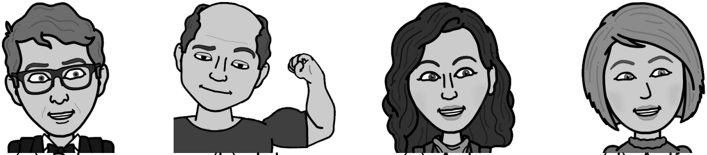
Big Idea



(a) Brian (b) John (c) Aphra (d) Aoife

Figure: Cards showing character faces and names for the *Guess-Who* game

Man	Long Hair	Glasses	Name
Yes	No	Yes	Brian
Yes	No	No	John
No	Yes	No	Aphra
No	No	No	Aoife



(a) Brian (b) John (c) Aphra (d) Aoife

Figure: Cards showing character faces and names for the *Guess-Who* game

.

Which question would you ask first:

1. Is it a man?
2. Does the person wear glasses?

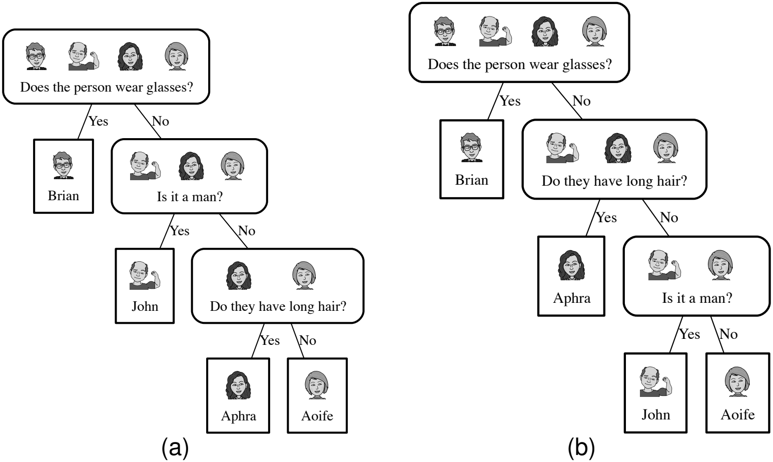


Figure: The different question sequences that can follow in a game of *Guess-Who* beginning with the question Does the person wear glasses?

- In both of the diagrams:
  - one path is 1 question long,
  - one path is 2 questions long,
  - and two paths are 3 questions long.
- Consequently, if you ask Question (2) first the average number of questions you have to ask per game is:

$$\frac{1+2+3+3}{4} = 2.25$$

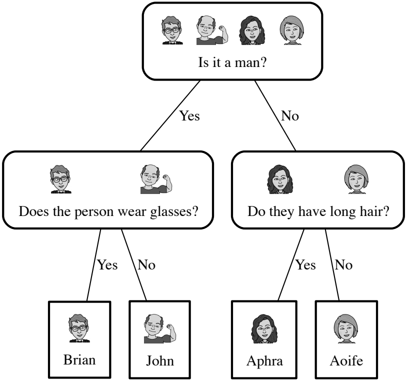


Figure: The different question sequences that can follow in a game of *Guess-Who* beginning with the question Is it a man?

- All the paths in this diagram are two questions long.
- So, on average if you ask Question (1) first the average number of questions you have to ask per game is:

$$\frac{2+2+2+2}{4}=2$$

- On average getting an answer to Question (1) seems to give you more information than an answer to Question (2): less follow up questions.
- This is not because of the literal message of the answers: YES or NO.
- It is to do with how the answer to each questions splits the domain into different sized sets based on the value of the descriptive feature the question is asked about and the likelihood of each possible answer to the question.

Big Idea

- So the big idea here is to figure out which features are the most informative ones to ask questions about by considering the effects of the different answers to the questions, in terms of:
  1. how the domain is split up after the answer is received,
  2. and the likelihood of each of the answers.

Fundamentals

Decision Trees

- A decision tree consists of:
  1. a **root node** (or starting node),
  2. **interior nodes**
  3. and **leaf nodes** (or terminating nodes).
- Each of the non-leaf nodes (root and interior) in the tree specifies a test to be carried out on one of the query's descriptive features.
- Each of the leaf nodes specifies a predicted classification for the query.

Table: An email spam prediction dataset.

	Suspicious	Unknown	Contains	
ID	Words	Sender	Images	Class
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

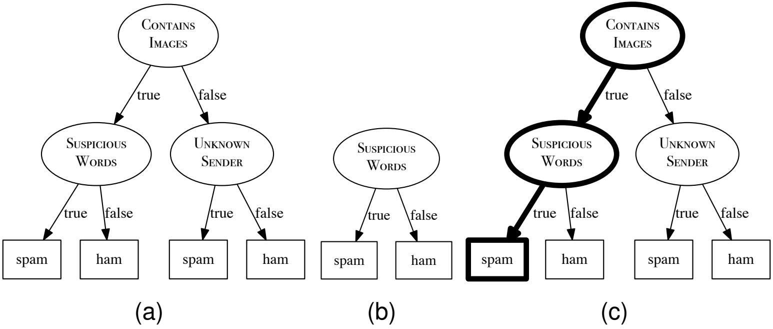


Figure: (a) and (b) show two decision trees that are consistent with the instances in the spam dataset. (c) shows the path taken through the tree shown in (a) to make a prediction for the query instance: SUSPICIOUS WORDS = *'true'*, UNKNOWN SENDER = *'true'*, CONTAINS IMAGES = *'true'*.

- Both of these trees will return identical predictions for all the examples in the dataset.
- So, which tree should we use?

- Apply the same approach as we used in the *Guess-Who* game: prefer decision trees that use less tests (shallower trees).
- This is an example of Occam's Razor.

How do we create shallow trees?

- The tree that tests SUSPICIOUS WORDS at the root is very shallow because the SUSPICIOUS WORDS feature perfectly splits the data into pure groups of *'spam'* and *'ham'*.
- Descriptive features that split the dataset into pure sets with respect to the target feature provide information about the target feature.
- So we can make shallow trees by testing the informative features early on in the tree.
- All we need to do that is a computational metric of the purity of a set: entropy

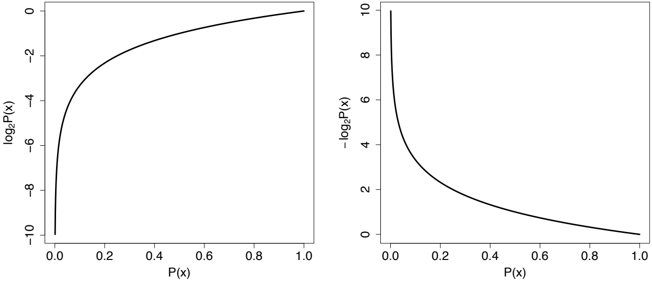
- Claude Shannon’s entropy model defines a computational measure of the impurity of the elements of a set.
- An easy way to understand the entropy of a set is to think in terms of the uncertainty associated with guessing the result if you were to make a random selection from the set.

- Entropy is related to the probability of a outcome.
  - High probability → Low entropy
  - Low probability → High entropy
- If we take the log of a probability and multiply it by -1 we get this mapping!

What is a log?

Remember the *log* of a to the base b is the number to which we must raise b to get a.

- $log_2(0.5) = -1$ because $2^{-1} = 0.5$
- $log_2(1) = 0$ because $2^0 = 1$
- $log_2(8) = 3$ because $2^3 = 8$
- $log_5(25) = 2$ because $5^2 = 25$
- $log_5(32) = 2.153$ because $5^{2.153} = 32$



(a) (b)

Figure: (a) A graph illustrating how the value of a binary log (the log to the base 2) of a probability changes across the range of probability values. (b) the impact of multiplying these values by -1.

- Shannon's model of entropy is a weighted sum of the logs of the probabilities of each of the possible outcomes when we make a random selection from a set.

$$H(t) = -\sum_{i=1}^{l} (P(t = i) \times log_s(P(t = i))) \qquad (1)$$

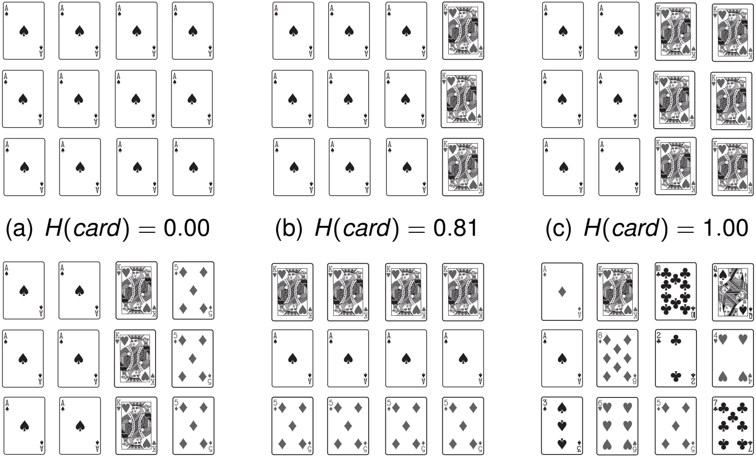
- What is the entropy of a set of 52 different playing cards?

$$
\begin{aligned}
H(card) &= -\sum_{i=1}^{52} P(card = i) \times log_2(P(card = i)) \\
&= -\sum_{i=1}^{52} 0.019 \times log_2(0.019) = -\sum_{i=1}^{52} -0.1096 \\
&= 5.700\ bits
\end{aligned}
$$

- What is the entropy of a set of 52 playing cards if we only distinguish between the cards based on their suit $\{\heartsuit, \clubsuit, \diamondsuit, \spadesuit\}$?

$$
\begin{aligned}
H(suit) &= -\sum_{l \in \{\heartsuit,\clubsuit,\diamondsuit,\spadesuit\}} P(suit = l) \times log_2(P(suit = l)) \\
&= -\big((P(\heartsuit) \times log_2(P(\heartsuit))) + (P(\clubsuit) \times log_2(P(\clubsuit))) \\
&\qquad + (P(\diamondsuit) \times log_2(P(\diamondsuit))) + (P(\spadesuit) \times log_2(P(\spadesuit))) \big) \\
&= -\Big(\left({}^{13}/_{52} \times log_2({}^{13}/_{52})\right) + \left({}^{13}/_{52} \times log_2({}^{13}/_{52})\right) \\
&\qquad + \left({}^{13}/_{52} \times log_2({}^{13}/_{52})\right) + \left({}^{13}/_{52} \times log_2({}^{13}/_{52})\right) \Big) \\
&= -\big((0.25 \times -2) + (0.25 \times -2) \\
&\qquad + (0.25 \times -2) + (0.25 \times -2) \big) \\
&= 2\ bits
\end{aligned}
$$

Shannon's Entropy Model



(a) $H(card) = 0.00$

(b) $H(card) = 0.81$

(c) $H(card) = 1.00$

(d) $H(card) = 1.50$

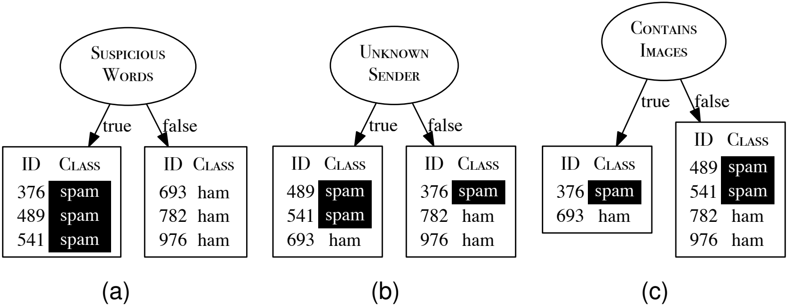
(e) $H(card) = 1.58$

(f) $H(card) = 3.58$

Figure: The entropy of different sets of playing cards measured in bits.

Table: The relationship between the entropy of a message and the set it was selected from.

Entropy of a Message	**Properties of the Message Set**
High	A large set of equally likely messages.
Medium	A large set of messages, some more likely than others.
Medium	A small set of equally likely messages.
Low	A small set of messages with one very likely message.



(a)

Suspicious Words = true

ID	Class
376	spam
489	spam
541	spam

Suspicious Words = false

ID	Class
693	ham
782	ham
976	ham

(b)

Unknown Sender = true

ID	Class
489	spam
541	spam
693	ham

Unknown Sender = false

ID	Class
376	spam
782	ham
976	ham

(c)

Contains Images = true

ID	Class
376	spam
693	ham

Contains Images = false

ID	Class
489	spam
541	spam
782	ham
976	ham

Figure: How the instances in the spam dataset split when we partition using each of the different descriptive features from the spam dataset in Table 1 [15]

Information Gain

- Our intuition is that the ideal discriminatory feature will partition the data into **pure** subsets where all the instances in each subset have the same classification.
  - SUSPICIOUS WORDS perfect split.
  - UNKNOWN SENDER mixture but some information (when *'true'* most instances are *'spam'*).
  - CONTAINS IMAGES no information.
- One way to implement this idea is to use a metric called **information gain**.

Information Gain

- The information gain of a descriptive feature can be understood as a measure of the reduction in the overall entropy of a prediction task by testing on that feature.

Computing information gain involves the following 3 equations:

$$H(t, \mathcal{D}) = -\sum_{l \in levels(t)} (P(t = l) \times log_2(P(t = l))) \tag{2}$$

$$rem(d, \mathcal{D}) = \sum_{l \in levels(d)} \underbrace{\frac{|\mathcal{D}_{d=l}|}{|\mathcal{D}|}}_{\text{weighting}} \times \underbrace{H(t, \mathcal{D}_{d=l})}_{\substack{\text{entropy of} \\ \text{partition } \mathcal{D}_{d=l}}} \tag{3}$$

$$IG(d, \mathcal{D}) = H(t, \mathcal{D}) - rem(d, \mathcal{D}) \tag{4}$$

- As an example we will calculate the information gain for each of the descriptive features in the spam email dataset.

- Calculate the entropy for the target feature in the dataset.

$$H(t, \mathcal{D}) = - \sum_{l \in levels(t)} (P(t = l) \times log_2(P(t = l)))$$

ID	Suspicious Words	Unknown Sender	Contains Images	Class
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

Information Gain

$$
\begin{aligned}
H(t,\mathcal{D}) &= -\sum_{l \in \{\text{'spam'},\text{'ham'}\}} \left(P(t=l) \times log_2(P(t=l))\right) \\
&= -\quad ((P(t=\text{'spam'}) \times log_2(P(t=\text{'spam'})) \\
&\qquad + (P(t=\text{'ham'}) \times log_2(P(t=\text{'ham'}))) \\
&= -\left(\left({}^{3}/_{6} \times log_2({}^{3}/_{6})\right) + \left({}^{3}/_{6} \times log_2({}^{3}/_{6})\right)\right) \\
&= 1\ bit
\end{aligned}
$$

- Calculate the remainder for the SUSPICIOUS WORDS feature in the dataset.

$$rem\,(d,\mathcal{D}) = \sum_{l \in levels(d)} \underbrace{\frac{|\mathcal{D}_{d=l}|}{|\mathcal{D}|}}_{\text{weighting}} \times \underbrace{H\,(t,\mathcal{D}_{d=l})}_{\substack{\text{entropy of}\\ \text{partition } \mathcal{D}_{d=l}}}$$

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

$$
\begin{aligned}
&rem\,(\textsc{words}, \mathcal{D}) \\
&= \left(\frac{|\mathcal{D}_{\textsc{words}=T}|}{|\mathcal{D}|} \times H\left(t, \mathcal{D}_{\textsc{words}=T}\right)\right) + \left(\frac{|\mathcal{D}_{\textsc{words}=F}|}{|\mathcal{D}|} \times H\left(t, \mathcal{D}_{\textsc{words}=F}\right)\right) \\
&= \left({}^{3}/_{6} \times \left(-\sum_{l \in \{\textit{'spam'}, \textit{'ham'}\}} P(t=l) \times log_2(P(t=l))\right)\right) \\
&\quad + \left({}^{3}/_{6} \times \left(-\sum_{l \in \{\textit{'spam'}, \textit{'ham'}\}} P(t=l) \times log_2(P(t=l))\right)\right) \\
&= \left({}^{3}/_{6} \times \left(-\left(\left({}^{3}/_{3} \times log_2({}^{3}/_{3})\right) + \left({}^{0}/_{3} \times log_2({}^{0}/_{3})\right)\right)\right)\right) \\
&\quad + \left({}^{3}/_{6} \times \left(-\left(\left({}^{0}/_{3} \times log_2({}^{0}/_{3})\right) + \left({}^{3}/_{3} \times log_2({}^{3}/_{3})\right)\right)\right)\right) = 0\ \textit{bits}
\end{aligned}
$$

- Calculate the remainder for the UNKNOWN SENDER feature in the dataset.

$$rem\left(d,\mathcal{D}\right) = \sum_{l \in levels(d)} \underbrace{\frac{|\mathcal{D}_{d=l}|}{|\mathcal{D}|}}_{\text{weighting}} \times \underbrace{H\left(t,\mathcal{D}_{d=l}\right)}_{\substack{\text{entropy of}\\ \text{partition } \mathcal{D}_{d=l}}}$$

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

$$
\begin{aligned}
&\mathit{rem}\left(\text{SENDER}, \mathcal{D}\right)\\
&\quad= \left(\frac{\left|\mathcal{D}_{\text{SENDER}=T}\right|}{\left|\mathcal{D}\right|} \times H\left(t, \mathcal{D}_{\text{SENDER}=T}\right)\right) + \left(\frac{\left|\mathcal{D}_{\text{SENDER}=F}\right|}{\left|\mathcal{D}\right|} \times H\left(t, \mathcal{D}_{\text{SENDER}=F}\right)\right)\\
&\quad= \left({}^{3}/_{6} \times \left(- \sum_{l \in \{\text{'spam'}, \text{'ham'}\}} P(t = l) \times \log_2(P(t = l))\right)\right)\\
&\quad\;\; + \left({}^{3}/_{6} \times \left(- \sum_{l \in \{\text{'spam'}, \text{'ham'}\}} P(t = l) \times \log_2(P(t = l))\right)\right)\\
&\quad= \left({}^{3}/_{6} \times \left(- \left(\left({}^{2}/_{3} \times \log_2({}^{2}/_{3})\right) + \left({}^{1}/_{3} \times \log_2({}^{1}/_{3})\right)\right)\right)\right)\\
&\quad\;\; + \left({}^{3}/_{6} \times \left(- \left(\left({}^{1}/_{3} \times \log_2({}^{1}/_{3})\right) + \left({}^{2}/_{3} \times \log_2({}^{2}/_{3})\right)\right)\right)\right) = 0.9183 \; \mathit{bits}
\end{aligned}
$$

- Calculate the remainder for the Contains Images feature in the dataset.

$$rem\left(d, \mathcal{D}\right) = \sum_{l \in levels(d)} \underbrace{\frac{\left|\mathcal{D}_{d=l}\right|}{\left|\mathcal{D}\right|}}_{\text{weighting}} \times \underbrace{H\left(t, \mathcal{D}_{d=l}\right)}_{\substack{\text{entropy of} \\ \text{partition } \mathcal{D}_{d=l}}}$$

	Suspicious	Unknown	Contains	
ID	Words	Sender	Images	Class
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

$$
\begin{aligned}
&\mathit{rem}\,(\text{IMAGES}, \mathcal{D}) \\
&= \left(\frac{\left|\mathcal{D}_{\text{IMAGES}=T}\right|}{\left|\mathcal{D}\right|} \times H\left(t, \mathcal{D}_{\text{IMAGES}=T}\right) \right) + \left(\frac{\left|\mathcal{D}_{\text{IMAGES}=F}\right|}{\left|\mathcal{D}\right|} \times H\left(t, \mathcal{D}_{\text{IMAGES}=F}\right) \right) \\
&= \left({}^{2}/_{6} \times \left(- \sum_{l \in \{\text{'spam'}, \text{'ham'}\}} P(t = l) \times log_2(P(t = l)) \right) \right) \\
&\quad + \left({}^{4}/_{6} \times \left(- \sum_{l \in \{\text{'spam'}, \text{'ham'}\}} P(t = l) \times log_2(P(t = l)) \right) \right) \\
&= \left({}^{2}/_{6} \times \left(- \left(\left({}^{1}/_{2} \times log_2({}^{1}/_{2}) \right) + \left({}^{1}/_{2} \times log_2({}^{1}/_{2}) \right) \right) \right) \right) \\
&\quad + \left({}^{4}/_{6} \times \left(- \left(\left({}^{2}/_{4} \times log_2({}^{2}/_{4}) \right) + \left({}^{2}/_{4} \times log_2({}^{2}/_{4}) \right) \right) \right) \right) = 1 \; \mathit{bit}
\end{aligned}
$$

- Calculate the information gain for the three descriptive feature in the dataset.

$$IG\left(d,\mathcal{D}\right)=H\left(t,\mathcal{D}\right)-rem\left(d,\mathcal{D}\right)$$

$$\begin{aligned}
IG(\textsc{Suspicious Words}, \mathcal{D}) &= H(\textsc{Class}, \mathcal{D}) - rem(\textsc{Suspicious Words}, \mathcal{D}) \\
&= 1 - 0 = 1 \; bit
\end{aligned}$$

$$\begin{aligned}
IG(\textsc{Unknown Sender}, \mathcal{D}) &= H(\textsc{Class}, \mathcal{D}) - rem(\textsc{Unknown Sender}, \mathcal{D}) \\
&= 1 - 0.9183 = 0.0817 \; bits \\
IG(\textsc{Contains Images}, \mathcal{D}) &= H(\textsc{Class}, \mathcal{D}) - rem(\textsc{Contains Images}, \mathcal{D}) \\
&= 1 - 1 = 0 \; bits
\end{aligned}$$

- The results of these calculations match our intuitions.

Standard Approach: The ID3 Algorithm

- ID3 Algorithm (Iterative Dichotomizer 3)
- Attempts to create the shallowest tree that is consistent with the data that it is given.
- The ID3 algorithm builds the tree in a recursive, depth-first manner, beginning at the root node and working down to the leaf nodes.

1. The algorithm begins by choosing the best descriptive feature to test (i.e., the best question to ask first) using **information gain**.
2. A root node is then added to the tree and labelled with the selected test feature.
3. The training dataset is then partitioned using the test.
4. For each partition a branch is grown from the node.
5. The process is then repeated for each of these branches using the relevant partition of the training set in place of the full training set and with the selected test feature excluded from further testing.

The algorithm defines three situations where the recursion stops and a leaf node is constructed:

1. All of the instances in the dataset have the same classification (target feature value) then return a leaf node tree with that classification as its label.
2. The set of features left to test is empty then return a leaf node tree with the majority class of the dataset as its classification.
3. The dataset is empty return a leaf node tree with the majority class of the dataset at the parent node that made the recursive call.

Table: The vegetation classification dataset.

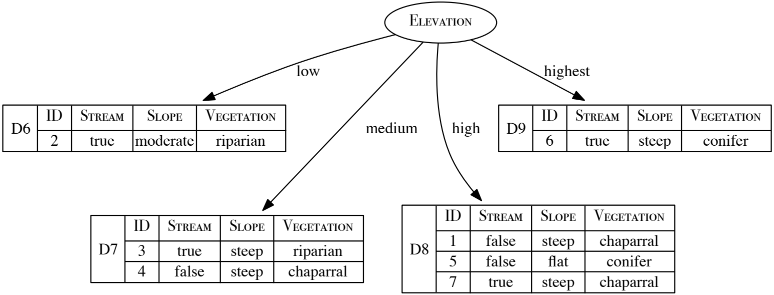
ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	high	chaparral
2	true	moderate	low	riparian
3	true	steep	medium	riparian
4	false	steep	medium	chaparral
5	false	flat	high	conifer
6	true	steep	highest	conifer
7	true	steep	high	chaparral

$$
\begin{aligned}
& H(\textsc{Vegetation}, \mathcal{D}) \\
&= - \sum_{l \in \left\{ \begin{array}{l} \textit{'chaparral'}, \\ \textit{'riparian'}, \\ \textit{'conifer'} \end{array} \right\}} P(\textsc{Vegetation} = l) \times log_2 \left(P(\textsc{Vegetation} = l) \right) \\
&= - \left(\left({}^{3}/_{7} \times log_2({}^{3}/_{7}) \right) + \left({}^{2}/_{7} \times log_2({}^{2}/_{7}) \right) + \left({}^{2}/_{7} \times log_2({}^{2}/_{7}) \right) \right) \\
&= 1.5567 \; \textit{bits}
\end{aligned}
$$

Table: Partition sets (Part.), entropy, remainder (Rem.) and information gain (Info. Gain) by feature for the dataset in Table 3 [49].

Split By Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
STREAM	*'true'*	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	1.5	1.2507	0.3060
	'false'	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.9183		
SLOPE	*'flat'*	$\mathcal{D}_3$	$\mathbf{d}_5$	0	0.9793	0.5774
	'moderate'	$\mathcal{D}_4$	$\mathbf{d}_2$	0		
	'steep'	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$	1.3710		
ELEVATION	*'low'*	$\mathcal{D}_6$	$\mathbf{d}_2$	0	0.6793	0.8774
	'medium'	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$	1.0		
	'high'	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$	0.9183		
	'highest'	$\mathcal{D}_9$	$\mathbf{d}_6$	0		

A Worked Example: Predicting Vegetation Distributions



D6

ID	STREAM	SLOPE	VEGETATION
2	true	moderate	riparian

D9

ID	STREAM	SLOPE	VEGETATION
6	true	steep	conifer

D7

ID	STREAM	SLOPE	VEGETATION
3	true	steep	riparian
4	false	steep	chaparral

D8

ID	STREAM	SLOPE	VEGETATION
1	false	steep	chaparral
5	false	flat	conifer
7	true	steep	chaparral

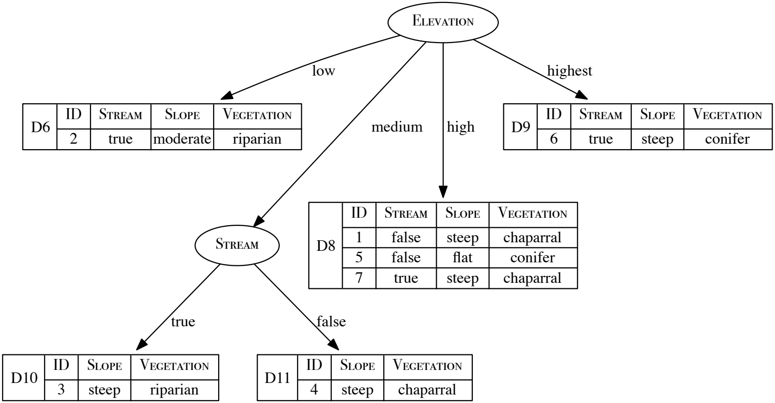
Figure: The decision tree after the data has been split using ELEVATION.

$$
\begin{aligned}
&H\left(\textsc{Vegetation}, \mathcal{D}_7\right)\\
&= - \sum_{l \in \left\{\substack{\textit{'chaparral'},\\ \textit{'riparian'},\\ \textit{'conifer'}}\right\}} P(\textsc{Vegetation} = l) \times log_2\left(P(\textsc{Vegetation} = l)\right)\\
&= -\left(\left({}^1/_2 \times log_2({}^1/_2)\right) + \left({}^1/_2 \times log_2({}^1/_2)\right) + \left({}^0/_2 \times log_2({}^0/_2)\right)\right)\\
&= 1.0\ \textit{bits}
\end{aligned}
$$

Table: Partition sets (Part.), entropy, remainder (Rem.) and information gain (Info. Gain) by feature for the dataset $\mathcal{D}_7$ in Figure 9 [52].

Split By Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
STREAM	*'true'*	$\mathcal{D}_{10}$	$\mathbf{d}_3$	0	0	1.0
	'false'	$\mathcal{D}_{11}$	$\mathbf{d}_4$	0		
SLOPE	*'flat'*	$\mathcal{D}_{12}$		0	1.0	0
	'moderate'	$\mathcal{D}_{13}$		0		
	'steep'	$\mathcal{D}_{14}$	$\mathbf{d}_3, \mathbf{d}_4$	1.0		

A Worked Example: Predicting Vegetation Distributions



D6

ID	STREAM	SLOPE	VEGETATION
2	true	moderate	riparian

D9

ID	STREAM	SLOPE	VEGETATION
6	true	steep	conifer

D8

ID	STREAM	SLOPE	VEGETATION
1	false	steep	chaparral
5	false	flat	conifer
7	true	steep	chaparral

D10

ID	SLOPE	VEGETATION
3	steep	riparian

D11

ID	SLOPE	VEGETATION
4	steep	chaparral

Figure: The state of the decision tree after the $\mathcal{D}_7$ partition has been split using STREAM.

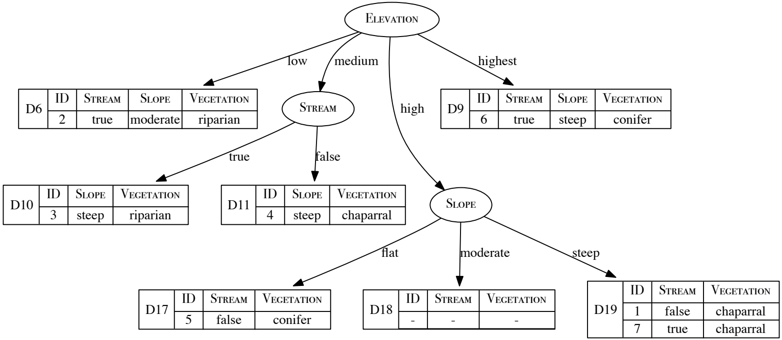
A Worked Example: Predicting Vegetation Distributions

$$
\begin{aligned}
& H\left(\textsc{Vegetation}, \mathcal{D}_8\right) \\
& = - \sum_{l \in \left\{\substack{\textit{'chaparral'}, \\ \textit{'riparian'}, \\ \textit{'conifer'}}\right\}} P(\textsc{Vegetation} = l) \times log_2\left(P(\textsc{Vegetation} = l)\right) \\
& = - \left(\left({}^{2}/_{3} \times log_2({}^{2}/_{3})\right) + \left({}^{0}/_{3} \times log_2({}^{0}/_{3})\right) + \left({}^{1}/_{3} \times log_2({}^{1}/_{3})\right)\right) \\
& = 0.9183 \; \textit{bits}
\end{aligned}
$$

Table: Partition sets (Part.), entropy, remainder (Rem.) and information gain (Info. Gain) by by feature for the dataset $\mathcal{D}_8$ in Figure 10 [55].

Split By				Partition		Info.
Feature	Level	Part.	Instances	Entropy	Rem.	Gain
STREAM	*’true’*	$\mathcal{D}_{15}$	$\mathbf{d}_7$	0	0.6666	0.2517
	’false’	$\mathcal{D}_{16}$	$\mathbf{d}_1, \mathbf{d}_5$	1.0		
SLOPE	*’flat’*	$\mathcal{D}_{17}$	$\mathbf{d}_5$	0	0	0.9183
	’moderate’	$\mathcal{D}_{18}$		0		
	’steep’	$\mathcal{D}_{19}$	$\mathbf{d}_1, \mathbf{d}_7$	0		

A Worked Example: Predicting Vegetation Distributions



D6

ID	Stream	Slope	Vegetation
2	true	moderate	riparian

D9

ID	Stream	Slope	Vegetation
6	true	steep	conifer

D10

ID	Slope	Vegetation
3	steep	riparian

D11

ID	Slope	Vegetation
4	steep	chaparral

D17

ID	Stream	Vegetation
5	false	conifer

D18

ID	Stream	Vegetation
-	-	-

D19

ID	Stream	Vegetation
1	false	chaparral
7	true	chaparral

Figure: The state of the decision tree after the $\mathcal{D}_8$ partition has been split using Slope.

A Worked Example: Predicting Vegetation Distributions

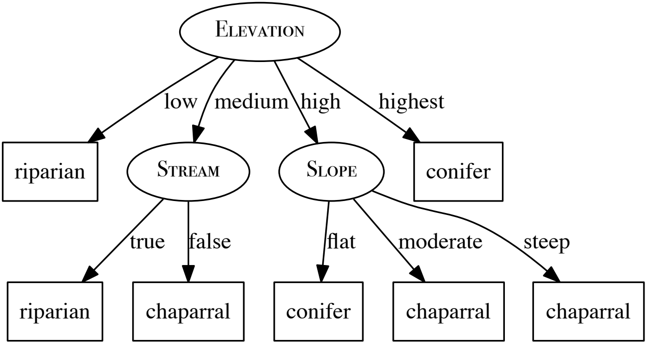
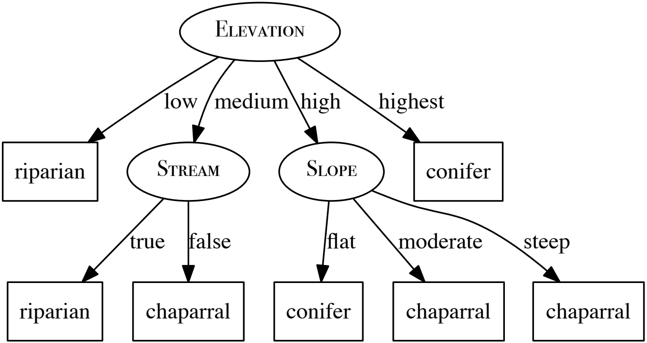
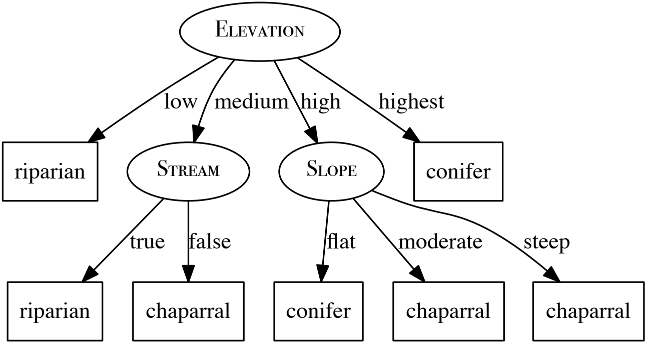


Figure: The final vegetation classification decision tree.



- What prediction will this decision tree model return for the following query?

STREAM = *’true’*, SLOPE=*’Moderate’*, ELEVATION=*’High’*



- What prediction will this decision tree model return for the following query?

STREAM = *'true'*, SLOPE=*'Moderate'*, ELEVATION=*'High'*

VEGETATION = *'Chaparral'*

ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	high	chaparral
2	true	moderate	low	riparian
3	true	steep	medium	riparian
4	false	steep	medium	chaparral
5	false	flat	high	conifer
6	true	steep	highest	conifer
7	true	steep	high	chaparral

STREAM = *'true'*, SLOPE=*'Moderate'*, ELEVATION=*'High'*

VEGETATION = *'Chaparral'*

- This is an example where the model is attempting to generalize beyond the dataset.
- Whether or not the generalization is correct depends on whether the assumptions used in generating the model (i.e. the inductive bias) were appropriate.

Summary

- The ID3 algorithm works the same way for larger more complicated datasets.
- There have been many extensions and variations proposed for the ID3 algorithm:
  - using different impurity measures (Gini, Gain Ratio)
  - handling continuous descriptive features
  - to handle continuous targets
  - pruning to guard against over-fitting
  - using decision trees as part of an ensemble (Random Forests)
- We cover these extensions in Section 4.4.

1. **Big Idea**
2. **Fundamentals**
   - Decision Trees
   - Shannon's Entropy Model
   - Information Gain
3. **Standard Approach: The ID3 Algorithm**
   - A Worked Example: Predicting Vegetation Distributions
4. **Summary**