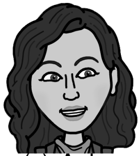
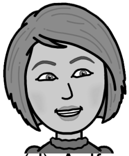
Information-based Learning

Outline

(a) Brian

(b) John


(c) Aphra

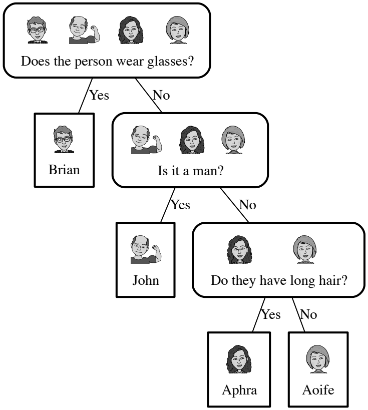

(d) Aoife

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

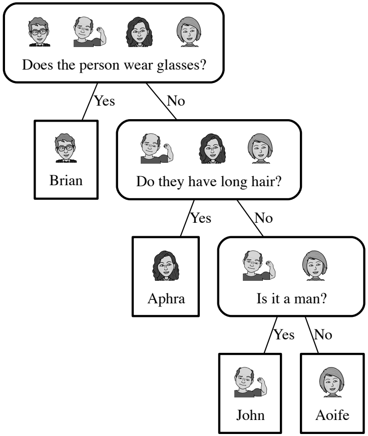
Man	**Long Hair**	**Glasses**	**Name**
Yes	No	Yes	Brian
Yes	No	No	John
No	Yes	No	Aphra
No	No	No	Aoife

Which question would you ask first:

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



(e)

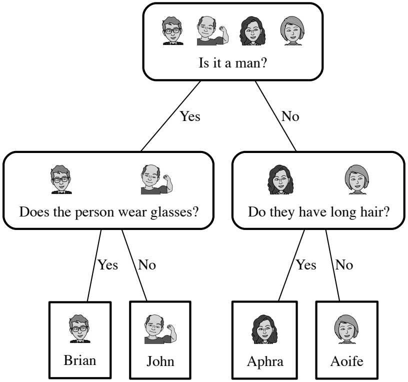


(f)

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:

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



Is it a man?

- All the paths in this diagram are two questions long.

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

Big Idea

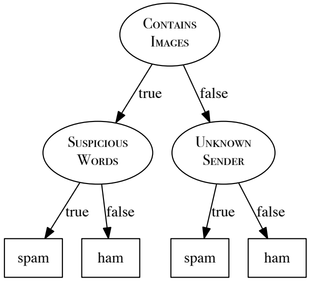
- So the big idea here is to figure out which features are the most informative ones to ask questions about by considering the effects of the different answers to the questions, in terms of:
    1. how the domain is split up after the answer is received,
    2. and the likelihood of each of the answers.

Decision Trees

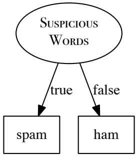
- A decision tree consists of:
  1. a **root node** (or starting node),
  2. **interior nodes**
  3. and **leaf nodes** (or terminating nodes).
- Each of the non-leaf nodes (root and interior) in the tree specifies a test to be carried out on one of the query's descriptive features.
- Each of the leaf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

An email spam prediction dataset.

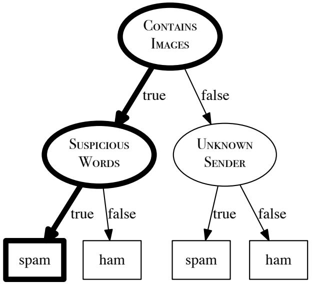
ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham



(g)



(h)



(i)

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

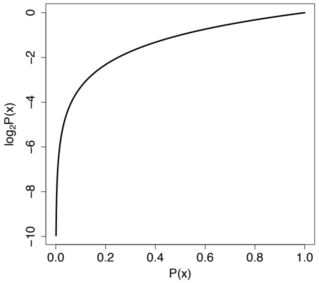
- Which tree should we use?

How do we create shallow trees?

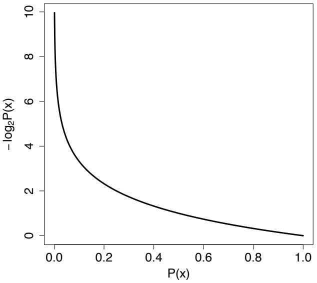
- The tree that tests SUSPICIOUS WORDS at the root is very shallow because the SUSPICIOUS WORDS feature perfectly splits the data into pure groups of *'spam'* and *'ham'*.
- Descriptive features that split the dataset into pure sets with respect to the target feature provide information about the target 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

Shannon's Entropy Model

- Claude Shannon's entropy model defines a computational measure of the impurity of the elements of a set.
- An easy way 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 $\rightarrow$ Low entropy
  - Low probability $\rightarrow$ High entropy
- If we take the log of a probability and multiply it by -1 we get this mapping!



(j)



(k)

(j) 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. (k) 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.

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

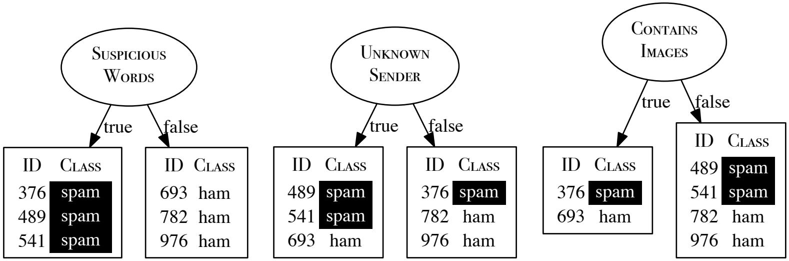
- What is the entropy of a set of 52 different playing cards?

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

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

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

Information Gain



(l) (m) (n)

How the instances in the spam dataset split when we partition using each of the different descriptive features from the spam dataset

- 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:

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

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

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

- Example: 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.

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

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

$$\begin{aligned}
H(t,\mathcal{D}) &= -\sum_{l \in \{'spam','ham'\}} \left(P(t=l) \times log_2(P(t=l))\right) \\
&= -\,((P(t='spam') \times log_2(P(t='spam')) \\
&\qquad + (P(t='ham') \times log_2(P(t='ham'))) \\
&= -\left(\left({}^3/_6 \times log_2({}^3/_6)\right) + \left({}^3/_6 \times log_2({}^3/_6)\right)\right) \\
&= 1\ bit
\end{aligned}$$

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

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

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

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

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

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

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

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

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

- 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.

A Worked Example: Predicting Vegetation Distributions

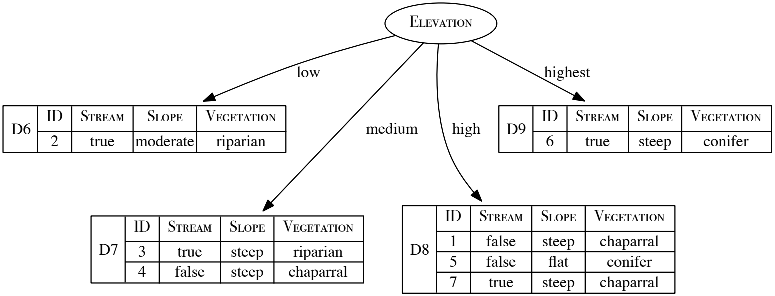
The vegetation classification dataset.

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

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

Partition sets (Part.), entropy, remainder (Rem.) and information gain (Info. Gain) by feature for the dataset

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		



D6

ID	STREAM	SLOPE	VEGETATION
2	true	moderate	riparian

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

D9

ID	STREAM	SLOPE	VEGETATION
6	true	steep	conifer

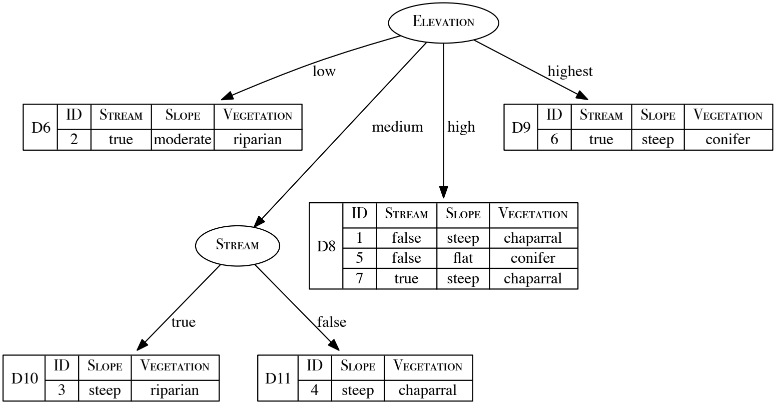
The decision tree after the data has been split using ELEVATION.

$$
\begin{aligned}
& H\left(\textsc{Vegetation}, \mathcal{D}_7\right) \\
& = - \sum_{l \in \left\{ \begin{array}{l} \textit{'chaparral'}, \\ \textit{'riparian'}, \\ \textit{'conifer'} \end{array} \right\}} P(\textsc{Vegetation} = l) \times log_2\left(P(\textsc{Vegetation} = l)\right) \\
& = - \left(\left({}^{1}/_{2} \times log_2({}^{1}/_{2}) \right) + \left({}^{1}/_{2} \times log_2({}^{1}/_{2}) \right) + \left({}^{0}/_{2} \times log_2({}^{0}/_{2}) \right) \right) \\
& = 1.0 \ \textit{bits}
\end{aligned}
$$

Partition sets (Part.), entropy, remainder (Rem.) and information gain (Info. Gain) by feature for $\mathcal{D}_7$

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		

*



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

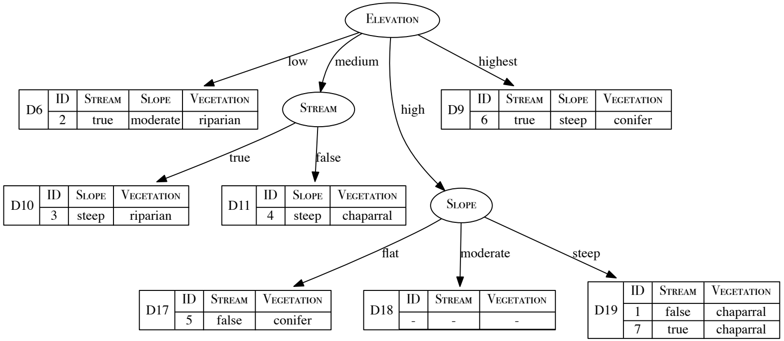
D10

ID	SLOPE	VEGETATION
3	steep	riparian

D11

ID	SLOPE	VEGETATION
4	steep	chaparral

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



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

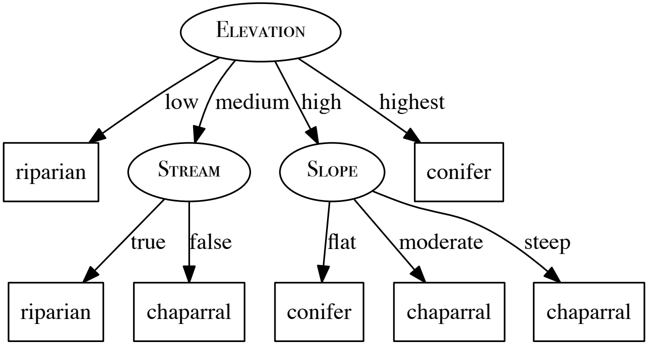
D18

ID	STREAM	VEGETATION
-	-	-

D19

ID	STREAM	VEGETATION
1	false	chaparral
7	true	chaparral

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



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

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

VEGETATION = *'Chaparral'*

Alternative Feature Selection Metrics

- Entropy based IG prefers features with many values.
- Use **information gain ratio** which is computed by dividing the information gain of a feature by the amount of information used to determine the value of the feature:

$$GR(d, \mathcal{D}) = \frac{IG(d, \mathcal{D})}{-\sum\limits_{l \in levels(d)} (P(d = l) \times log_2(P(d = l)))} \tag{5}$$

- Another commonly used measure of impurity is the **Gini index**:

$$Gini\,(t, \mathcal{D}) = 1 - \sum_{l \in levels(t)} P(t = l)^2 \tag{6}$$

- The Gini index can be thought of as calculating how often you would misclassify an instance in the dataset if you classified it based on the distribution of classifications in the dataset.
- Information gain can be calculated using the Gini index by replacing the entropy measure with the Gini index.

Handling Continuous Descriptive Features

- Turn continuous valued descriptive features into boolean features by defining a threshold
  1. The instances in the dataset are sorted according to the continuous feature values.
  2. The adjacent instances in the ordering that have different classifications are selected as possible threshold points.
  3. The optimal threshold is found by computing the information gain for each of these classification transition boundaries and selecting the boundary with the highest information gain as the threshold.
- Once a threshold has been set, the new feature can compete with the other categorical features.
- This process is repeated at each node as the tree grows.

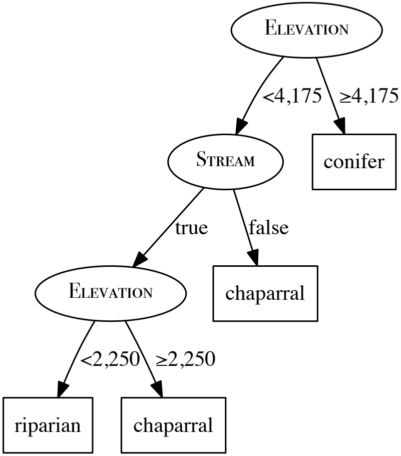
Dataset for predicting the vegetation in an area with a continuous ELEVATION feature (measured in feet).

ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	3 900	chapparal
2	true	moderate	300	riparian
3	true	steep	1 500	riparian
4	false	steep	1 200	chapparal
5	false	flat	4 450	conifer
6	true	steep	5 000	conifer
7	true	steep	3 000	chapparal

ID	STREAM	SLOPE	ELEVATION	VEGETATION
2	true	moderate	300	riparian
4	false	steep	1 200	chapparal
3	true	steep	1 500	riparian
7	true	steep	3 000	chapparal
1	false	steep	3 900	chapparal
5	false	flat	4 450	conifer
6	true	steep	5 000	conifer

Partition sets (Part.), entropy, remainder (Rem.), and information gain (Info. Gain) for the candidate ELEVATION thresholds: $\geq$750, $\geq$1 350, $\geq$2 250 and $\geq$4 175.

Split by Threshold	Part.	Instances	Partition Entropy	Rem.	Info. Gain
$\geq$750	$\mathcal{D}_1$	$\mathbf{d}_2$	0.0	1.2507	0.3060
	$\mathcal{D}_2$	$\mathbf{d}_4, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.4591		
$\geq$1 350	$\mathcal{D}_3$	$\mathbf{d}_2, \mathbf{d}_4$	1.0	1.3728	0.1839
	$\mathcal{D}_4$	$\mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.5219		
$\geq$2 250	$\mathcal{D}_5$	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_3$	0.9183	0.9650	0.5917
	$\mathcal{D}_6$	$\mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.0		
$\geq$4 175	$\mathcal{D}_7$	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1$	0.9710	0.6935	0.8631
	$\mathcal{D}_8$	$\mathbf{d}_5, \mathbf{d}_6$	0.0		



The decision tree that would be generated for the vegetation classification dataset using information gain.

Noisy Data, Overfitting and Tree Pruning

- In the case of a decision tree, over-fitting involves splitting the data on an irrelevant feature.

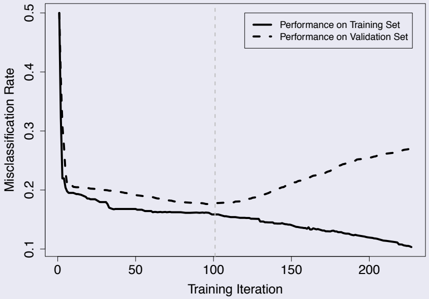
The likelihood of over-fitting occurring increases as a tree gets deeper because the resulting classifications are based on smaller and smaller subsets as the dataset is partitioned after each feature test in the path.

Advantages of pruning:

- Smaller trees are easier to interpret
- Increased generalization accuracy when there is noise in the training data (**noise dampening**).
- **Pre-pruning**: stop the recursive partitioning early. Pre-pruning is also known as **forward pruning**.
- **Post-pruning**: allow the algorithm to grow the tree as much as it likes and then prune the tree of the branches that cause over-fitting.

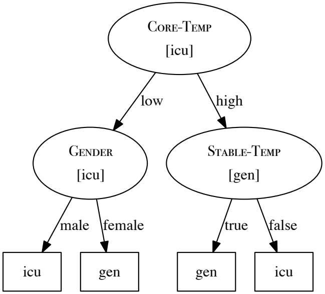
Common Post-pruning Approach

- Using the validation set evaluate the prediction accuracy achieved by both the fully grown tree and the pruned copy of the tree. If the pruned copy of the tree performs no worse than the fully grown tree the node is a candidate for pruning.

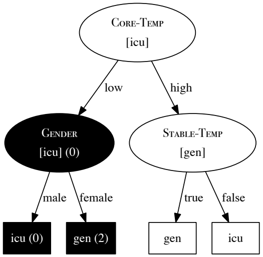


An example validation set for the post-operative patient routing task.

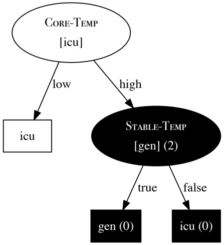
ID	CORE-TEMP	STABLE-TEMP	GENDER	DECISION
1	high	true	male	gen
2	low	true	female	icu
3	high	false	female	icu
4	high	false	male	icu
5	low	false	female	icu
6	low	true	male	icu



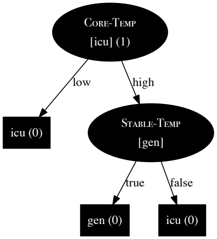
The decision tree for the post-operative patient routing task.



(o)



(p)



(q)

The iterations of reduced error pruning for the decision tree using the validation set. The subtree that is being considered for pruning in each iteration is highlighted in black. The prediction returned by each non-leaf node is listed in square brackets. The error rate for each node is given in round brackets.