Data Mining and Machine Learning

Comp 3027J

Dr Catherine Mooney
Assistant Professor

catherine.mooney@ucd.ie

Lectures and Text

- **Core Text:**
  Fundamentals of Machine Learning for Predictive Data Analytics
  By John D. Kelleher, Brian Mac Namee and Aoife D'Arcy
- Last week we covered Chapter 5, sections 5.4.6 (Feature Selection) and Chapter 8 (Evaluation), sections 8.2, 8.3 and 8.4.1–8.4.2.3.
- This week we will cover Chapter 4, sections 4.2 and 4.3 (Information-based Learning – Decision Trees and The ID3 Algorithm).
- Please read these sections of the book.

1. Information-based Learning
2. Decision Trees
3. Shannon's Entropy Model
4. Information Gain
5. Standard Approach: The ID3 Algorithm
6. A Worked Example: Predicting Vegetation Distributions

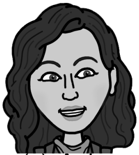
Information-based Learning

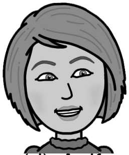
- In this lecture we are going to introduce a machine learning algorithm that tries to build predictive models **using only the most informative features**.
- In this context an informative feature is a **descriptive feature** whose values split the instances in the dataset into **homogeneous sets**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

Character faces and names for the *Guess-Who* game

(a) Brian

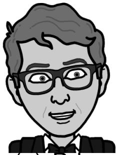
(b) John

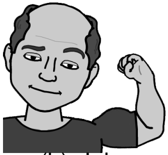

(c) Aphra

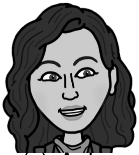

(d) Aoife

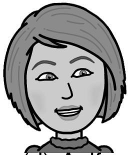
Man	**Long Hair**	**Glasses**	**Name**
Yes	No	Yes	Brian
Yes	No	No	John
No	Yes	No	Aphra
No	No	No	Aoife

Character faces and names for the *Guess-Who* game


(a) Brian

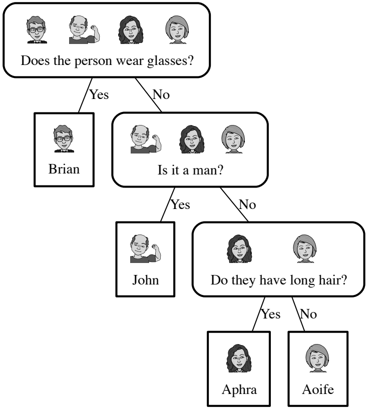

(b) John


(c) Aphra

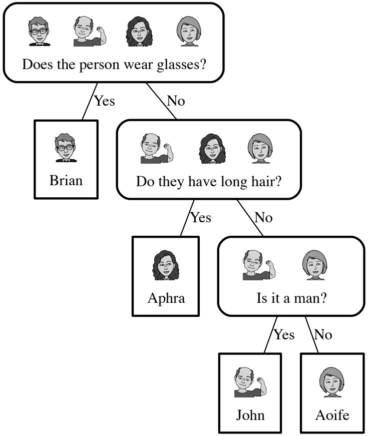

(d) Aoife

Which question would you ask first:

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



(a)

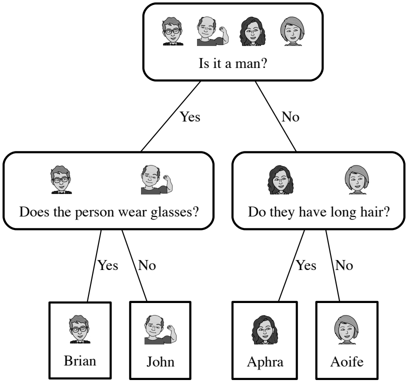


(b)

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 "Does the person wear glasses?" first, the average number of questions you have to ask per game is:

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



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 "Is it a man?" 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 “Is it a man?” seems to give you more information than an answer to “Does the person wear glasses?”: 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

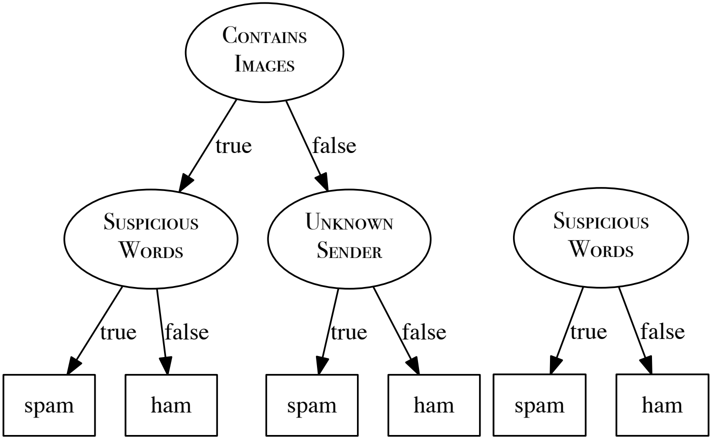
- So the big idea here is to figure out which features are the most informative ones to ask questions about by considering the effects of the different answers to the questions, in terms of:
  1. how the domain is split up after the answer is received,
  2. and the likelihood of each of the answers.

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



Two decision trees that are consistent with the instances in the spam dataset.

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

- It is often possible to build many different decision trees that are consistent with the data.
- We should chose decision trees that use less tests (shallower trees).

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

- 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 $\rightarrow$ Low entropy
  - Low probability $\rightarrow$ High entropy
- If we take the log of a probability and multiply it by -1 we get this mapping!

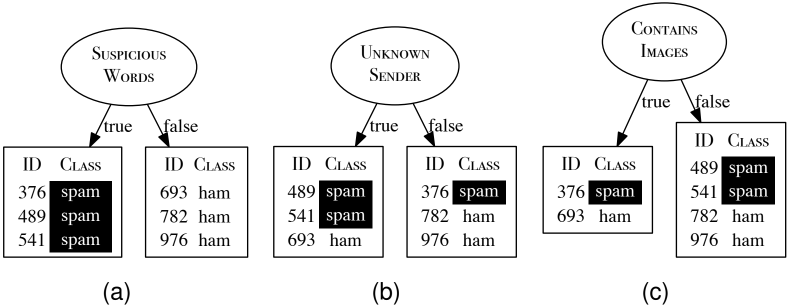
- Shannon's model of entropy is a weighted sum of the logs of the probabilities of each of the possible outcomes when we make a random selection from a set.

$$H(t) = -\sum_{i=1}^{l} (P(t = i) \times log_2(P(t = i)))$$

Information Gain

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



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 is a <u>three-step</u> process:

1. Compute the entropy of the original dataset with respect to the target feature.
    - This gives us a measure of how much information is required in order to organize the dataset into pure sets.
2. For each descriptive feature, create the sets that result by partitioning the instances in the dataset using their feature values, and then sum the entropy scores of each of these sets.
    - This gives a measure of the information that remains required to organize the instances into pure sets after we have split them using the descriptive feature.
3. Subtract the remaining entropy value (computed in step 2) from the original entropy value (computed in step 1) to give the information gain.

- We need to define three equations to formally specify information gain (one for each step).

The first equation calculates the entropy for a dataset with respect to a target feature:

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

where *levels(t)* is the set of levels in the domain of the target feature *t*, and *P(t = l)* is the probability of a randomly selected instance having the target feature level *l*.

The second equation defines how we compute the entropy remaining after we partition the dataset using a particular descriptive feature *d*.

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

The weighting is determined by the size of each partition – so a large partition should contribute more to the overall remaining entropy than a smaller partition.

The third equation – Using Equation 1 and Equation 2, we can now calculate the information gain made from splitting the dataset $\mathcal{D}$ using the feature *d*

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

- 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

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

- 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

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

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

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

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

$$
\begin{aligned}
&\mathit{rem}\left(\text{SENDER}, \mathcal{D}\right)\\
&\quad = \left(\frac{\left|\mathcal{D}_{\text{SENDER}=T}\right|}{\left|\mathcal{D}\right|} \times H\left(t, \mathcal{D}_{\text{SENDER}=T}\right)\right) + \left(\frac{\left|\mathcal{D}_{\text{SENDER}=F}\right|}{\left|\mathcal{D}\right|} \times H\left(t, \mathcal{D}_{\text{SENDER}=F}\right)\right)\\
&\quad = \left({}^{3}/_{6} \times \left(- \sum_{l \in \{\textit{'spam'}, \textit{'ham'}\}} P(t=l) \times \mathit{log}_2(P(t=l))\right)\right)\\
&\quad\quad + \left({}^{3}/_{6} \times \left(- \sum_{l \in \{\textit{'spam'}, \textit{'ham'}\}} P(t=l) \times \mathit{log}_2(P(t=l))\right)\right)\\
&\quad = \left({}^{3}/_{6} \times \left(-\left(\left({}^{2}/_{3} \times \mathit{log}_2({}^{2}/_{3})\right) + \left({}^{1}/_{3} \times \mathit{log}_2({}^{1}/_{3})\right)\right)\right)\right)\\
&\quad\quad + \left({}^{3}/_{6} \times \left(-\left(\left({}^{1}/_{3} \times \mathit{log}_2({}^{1}/_{3})\right) + \left({}^{2}/_{3} \times \mathit{log}_2({}^{2}/_{3})\right)\right)\right)\right) = 0.9183 \ \mathit{bits}
\end{aligned}
$$

- Calculate the remainder for the CONTAINS IMAGES feature in the dataset.

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

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

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

- 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.

Any questions? 5 min break...

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

(a) Chaparral

(b) Riparian

(c) Conifer

- Chapparal is a type of evergreen shrubland that can be fire-prone. The animal species typically found in this vegetation include gray foxes, bobcats, skunks, and rabbits.
- Riparian vegetation occurs near streams and is characterized by trees and shrubs. It is usually home to small animals, including raccoons, frogs, and toads.
- Conifer refers to forested areas that contain a variety of tree species (including pine, cedar, and fir trees), with a mixture of shrubs on the forest floor. The animals that may be found in these forests include bears, deer, and cougars.

- The first step in building the decision tree is to determine which of the three descriptive features is the best one to split the dataset on at the root node.
- The algorithm does this by computing the information gain for each feature.
- This is a three-step process.

1. Compute the entropy of the original dataset with respect to the target feature.
    - This gives us an measure of how much information is required in order to organize the dataset into pure sets.

The total entropy of the original dataset is computed as:

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

$$H(\text{VEGETATION}, \mathcal{D}) = -\sum_{l \in \left\{\begin{array}{l}\text{'chaparral'},\\ \text{'riparian'},\\ \text{'conifer'}\end{array}\right\}} P(\text{VEGETATION} = l) \times log_2\left(P(\text{VEGETATION} = l)\right)$$

$$
\begin{aligned}
&H(\text{VEGETATION}, \mathcal{D}) \\
&= - \sum_{l \in \left\{ \substack{\textit{'chaparral'}, \\ \textit{'riparian'}, \\ \textit{'conifer'}} \right\}} P(\text{VEGETATION} = l) \times log_2 \left(P(\text{VEGETATION} = l) \right) \\
&= ?
\end{aligned}
$$

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

$$
\begin{aligned}
&H(\text{VEGETATION}, \mathcal{D}) \\
&= - \sum_{l \in \left\{ \begin{array}{l} \textit{'chaparral'}, \\ \textit{'riparian'}, \\ \textit{'conifer'} \end{array} \right\}} P(\text{VEGETATION} = l) \times log_2 \left(P(\text{VEGETATION} = l) \right) \\
&= - \left(\left({}^3/_7 \times log_2({}^3/_7) \right) + \left({}^2/_7 \times log_2({}^2/_7) \right) + \left({}^2/_7 \times log_2({}^2/_7) \right) \right) \\
&= 1.5567 \; bits
\end{aligned}
$$

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

2. For each of the three descriptive features (STREAM, SLOPE and ELEVATION), create the sets that result by partitioning the instances in the dataset using their feature values, and then sum the entropy scores of each of these sets.
   - This gives a measure of the information that remains required to organize the instances into pure sets after we have split them using the descriptive feature (*rem*).

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

(Note – The weighting is determined by the size of each partition)

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

Split By Feature	Level	Partition Sets	Instances	Weighting	Partition Entropy	Remainder *rem*
STREAM	*'true'*	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$			
	'false'	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$			
SLOPE	*'flat'*	$\mathcal{D}_3$	$\mathbf{d}_5$			
	'moderate'	$\mathcal{D}_4$	$\mathbf{d}_2$			
	'steep'	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$			
ELEVATION	*'low'*	$\mathcal{D}_6$	$\mathbf{d}_2$			
	'medium'	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$			
	'high'	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$			
	'highest'	$\mathcal{D}_9$	$\mathbf{d}_6$			

$$rem(\text{STREAM}, \mathcal{D}) = \left(\frac{|\mathcal{D}_{\text{STREAM}=T}|}{|\mathcal{D}|} \times H(t, \mathcal{D}_{\text{STREAM}=T}) \right) + \left(\frac{|\mathcal{D}_{\text{STREAM}=F}|}{|\mathcal{D}|} \times H(t, \mathcal{D}_{\text{STREAM}=F}) \right)$$

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

Split By Feature	Level	Partition Sets	Instances	Weighting	Partition Entropy	Remainder *rem*
STREAM	*'true'*	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	0.5714		
	'false'	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.4286		
SLOPE	*'flat'*	$\mathcal{D}_3$	$\mathbf{d}_5$			
	'moderate'	$\mathcal{D}_4$	$\mathbf{d}_2$			
	'steep'	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$			
ELEVATION	*'low'*	$\mathcal{D}_6$	$\mathbf{d}_2$			
	'medium'	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$			
	'high'	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$			
	'highest'	$\mathcal{D}_9$	$\mathbf{d}_6$			

(Note – The weighting is determined by the size of each partition i.e. 4/7 = 0.5714 and 3/7 = 0.4286)

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

Split By Feature	Level	Partition Sets	Instances	Weighting	Partition Entropy	Remainder *rem*
STREAM	*'true'*	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	0.5714	1.5	
	'false'	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.4286	0.9183	
SLOPE	*'flat'*	$\mathcal{D}_3$	$\mathbf{d}_5$			
	'moderate'	$\mathcal{D}_4$	$\mathbf{d}_2$			
	'steep'	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$			
ELEVATION	*'low'*	$\mathcal{D}_6$	$\mathbf{d}_2$			
	'medium'	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$			
	'high'	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$			
	'highest'	$\mathcal{D}_9$	$\mathbf{d}_6$			

Next calculate the entropy of each partition in STREAM i.e.

$$H(t,\mathcal{D}_{\text{STREAM}=T})\; H(t,\mathcal{D}_{\text{STREAM}=F})$$

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

$$H\left(t, \mathcal{D}_{\text{STREAM}=T}\right)$$

$$= - \sum_{l \in \left\{ \begin{array}{l} \textit{'chaparral'}, \\ \textit{'riparian'}, \\ \textit{'conifer'} \end{array} \right\}} P(t = l) \times log_2(P(t = l))$$

$$= -(({}^{1}/_{4} \times log_2({}^{1}/_{4}) + ({}^{2}/_{4} \times log_2({}^{2}/_{4}) + ({}^{1}/_{4} \times log_2({}^{1}/_{4}))$$

$$= -((0.25 \times -2) + (0.5 \times -1) + (0.25 \times -2))$$

$$= 1.5$$

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

$$H\left(t, \mathcal{D}_{\text{STREAM}=F}\right)$$

$$= - \sum_{l \in \left\{ \substack{\text{'chaparral'}, \\ \text{'riparian'}, \\ \text{'conifer'}} \right\}} P(t=l) \times log_2(P(t=l))$$

$$= -((^2/_3 \times log_2(^2/_3) + (^0/_3 \times log_2(^0/_3) + (^1/_3 \times log_2(^1/_3))$$

$$= -((0.66 \times -.58) + (0) + (0.33 \times -1.53))$$

$$= 0.9183$$

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

Split By Feature	Level	Partition Sets	Instances	Weighting	Partition Entropy	Remainder *rem*
STREAM	*'true'*	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	0.5714	1.5	1.2507
	'false'	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.4286	0.9183	
SLOPE	*'flat'*	$\mathcal{D}_3$	$\mathbf{d}_5$			
	'moderate'	$\mathcal{D}_4$	$\mathbf{d}_2$			
	'steep'	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$			
ELEVATION	*'low'*	$\mathcal{D}_6$	$\mathbf{d}_2$			
	'medium'	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$			
	'high'	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$			
	'highest'	$\mathcal{D}_9$	$\mathbf{d}_6$			

Remainder (*rem*) = (0.5714 x 1.5) + (0.4286 x 0.9183) = 1.2507

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

Split By Feature	Level	Partition Sets	Instances	Weighting	Partition Entropy	Remainder *rem*
STREAM	*'true'*	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	0.5714	1.5	1.2507
	'false'	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.4286	0.9183	
SLOPE	*'flat'*	$\mathcal{D}_3$	$\mathbf{d}_5$	0.1429	0	0.9793
	'moderate'	$\mathcal{D}_4$	$\mathbf{d}_2$	0.1429	0	
	'steep'	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$	0.7143	1.3710	
ELEVATION	*'low'*	$\mathcal{D}_6$	$\mathbf{d}_2$	0.1429	0	0.6793
	'medium'	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$	0.2857	1.0	
	'high'	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$	0.4286	0.9183	
	'highest'	$\mathcal{D}_9$	$\mathbf{d}_6$	0.1429	0	

3. Subtract the remaining entropy value (computed in step 2) from the original entropy value (computed in step 1) to give the information gain.

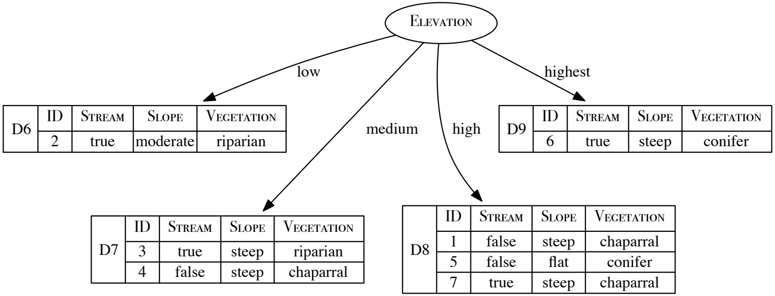
$$IG(d, \mathcal{D}) = H(t, \mathcal{D}) - rem(d, \mathcal{D})$$

Split By Feature	Total Entropy	Remainder *rem*	Info. Gain
STREAM	1.5567	1.2507	
SLOPE	1.5567	0.9793	
ELEVATION	1.5567	0.6793	

Which feature should we select as the root node of the tree i.e. which feature has the largest information gain?

Split By Feature	Total Entropy	Remainder *rem*	Info. Gain
STREAM	1.5567	1.2507	0.3060
SLOPE	1.5567	0.9793	0.5774
ELEVATION	1.5567	0.6793	0.8774

ELEVATION has the largest information gain of the three features and is selected as the root node of the tree



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 $\mathcal{D}_6$ and $\mathcal{D}_9$ partitions each contain just one instance.
- Consequently, they are pure sets, and these partitions can be converted into leaf nodes.
- The $\mathcal{D}_7$ and $\mathcal{D}_8$ partitions, however, contain instances with a mixture of target feature levels, so the algorithm needs to continue splitting these partitions.
- To do this, the algorithm needs to decide which of the remaining descriptive features has the highest information gain for each partition.

First the algorithm computes the entropy of $\mathcal{D}_7$:

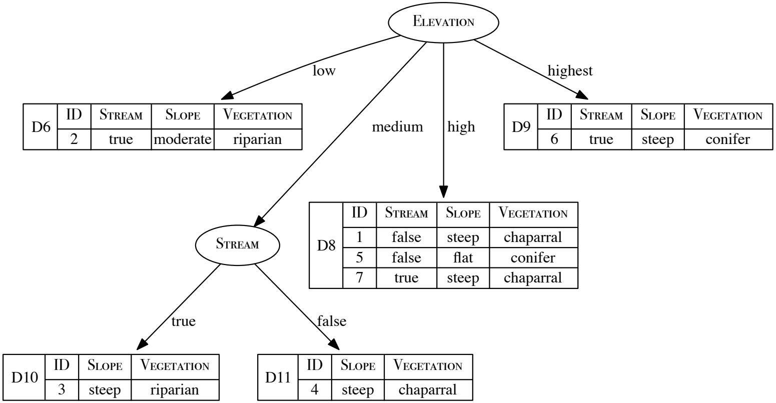
$$\begin{aligned}
&H\left(\text{VEGETATION}, \mathcal{D}_7\right)\\
&= - \sum_{l \in \left\{ \begin{array}{l} \text{'chaparral'}, \\ \text{'riparian'}, \\ \text{'conifer'} \end{array} \right\}} P(\text{VEGETATION} = l) \times log_2\left(P(\text{VEGETATION} = l)\right)\\
&= -\left(\left({}^{1}/_{2} \times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

Calculation of the information gain (Info. Gain) for features STREAM and SLOPE for the dataset $\mathcal{D}_7$

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		

Partition sets (Part.), remainder (Rem.)

STREAM has a higher information gain than SLOPE and so is the best feature with which to split $\mathcal{D}_7$.



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

- Splitting $\mathcal{D}_7$ creates two new partitions ($\mathcal{D}_{10}$ and $\mathcal{D}_{11}$).
- Both of these new partitions are pure sets with respect to the target feature and can be converted into leaf nodes.
- $\mathcal{D}_8$ is the only partition that is not a pure set.
- There are two descriptive features that can be used to split $\mathcal{D}_8$: STREAM and SLOPE.
- The decision regarding which of these features to split on is made by calculating which feature has the highest information gain for $\mathcal{D}_8$.

The overall entropy for $\mathcal{D}_8$ is calculated as

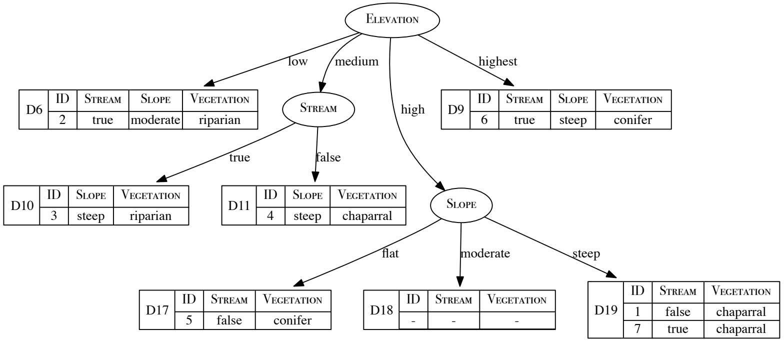
$$\begin{aligned}
&H\left(\textsc{Vegetation}, \mathcal{D}_8\right)\\
&= - \sum_{l \in \left\{\substack{\textit{'chaparral'},\\ \textit{'riparian'},\\ \textit{'conifer'}}\right\}} P(\textsc{Vegetation} = l) \times log_2\left(P(\textsc{Vegetation} = l)\right)\\
&= -\left(\left({}^{2}/_{3} \times log_2({}^{2}/_{3})\right)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

Calculation of the information gain (Info. Gain) for features STREAM and SLOPE for the dataset $\mathcal{D}_8$

Split By Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. 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		

Partition sets (Part.), remainder (Rem.)

SLOPE has a higher information gain than STREAM and so is the best feature with which to split $\mathcal{D}_8$.



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

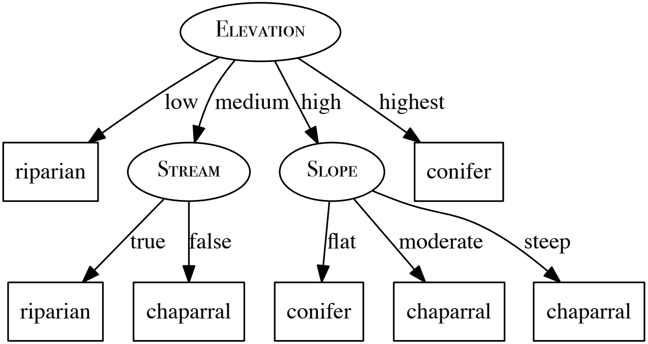
D18

ID	STREAM	VEGETATION
-	-	-

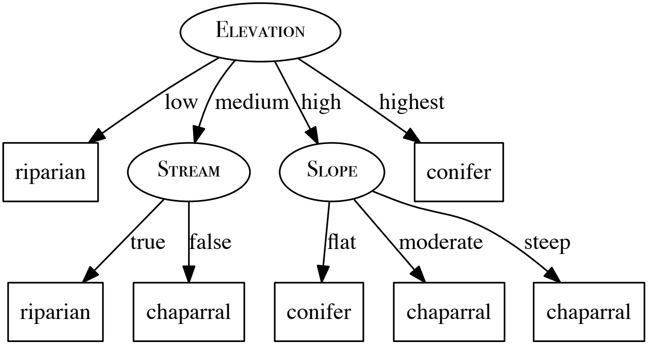
D19

ID	STREAM	VEGETATION
1	false	chaparral
7	true	chaparral

- Notice that one of the partitions created by splitting $\mathcal{D}_8$ based on SLOPE is empty: $\mathcal{D}_{18}$.
- This is because there were no instances in $\mathcal{D}_8$ that had a value of *moderate* for the SLOPE feature.
- This empty partition will result in a leaf node that returns a prediction of the majority target level in $\mathcal{D}_8$, *chapparal*.
- The other two partitions created by splitting $\mathcal{D}_8$ are pure with respect to the target feature and can be converted into leaf nodes.

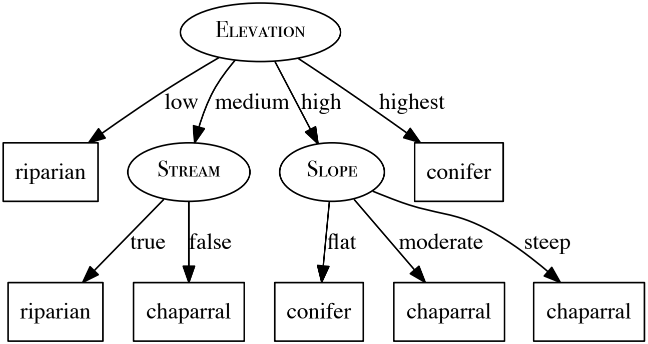


The final vegetation classification decision tree.



- 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

1. Information-based Learning
2. Decision Trees
3. Shannon's Entropy Model
4. Information Gain
5. Standard Approach: The ID3 Algorithm
6. A Worked Example: Predicting Vegetation Distributions

- 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
  - using decision trees as part of an ensemble (Random Forests)
- We cover some of these extensions next week

Recommended Reading

- **Core Text:**
  Fundamentals of Machine Learning for Predictive Data Analytics
  By John D. Kelleher, Brian Mac Namee and Aoife D'Arcy
- This week we covered Chapter 4, sections 4.2 and 4.3 (Information-based Learning – Decision Trees and The ID3 Algorithm).
- I would suggest that you would read over these sections again.
- Email me if you have any questions and I will cover them at the beginning of class next week.

Review of Lab 4

Review of Lab 4 – KNN

- In Lab 4 we used the k-Nearest Neighbors algorithm (kNN) to predict to predict upcoming generator failures
- The solution for Lab 4 is on Moodle
- Make sure that you submit something at the end of the labs

Review of Lab 4 – Feedback

- Most people found the lab too long
- But, not too difficult
- Internet issues, again!
- Remember to download packages before the lab
- Will I post the lab om Moodle in advance? No...
- Why not...?!

The structure of a confusion matrix.

		Prediction	
		POSITIVE	**NEGATIVE**
Target	**POSITIVE**	*TP*	*FN*
	NEGATIVE	*FP*	*TN*

The structure of a confusion matrix.

		Prediction	
		FAULTY	**GOOD**
Target	**FAULTY**	*TP*	*FN*
	GOOD	*FP*	*TN*

Assignment – Due 4.30pm on Friday

Assignment

- Any questions/problems?
- What am I looking for...?
- I want to see that you can complete the tasks
- But I also what to see that you understand what you are doing