Chapter 4: Information-based Learning

MATH2319

- This chapter discusses decision trees, which are algorithms that 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 **pure sets** with respect to the target feature value.
- The resulting structure is then called a **decision tree**.
- Decision trees are among the most important and most commonly used ML methods in the real world.

Big Idea

- 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.
- The goal is to keep splitting the training data in such a way that we will end up with (almost!) pure sets (in terms of the values of the target feature) as quickly as possible.

Fundamentals

- 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.

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

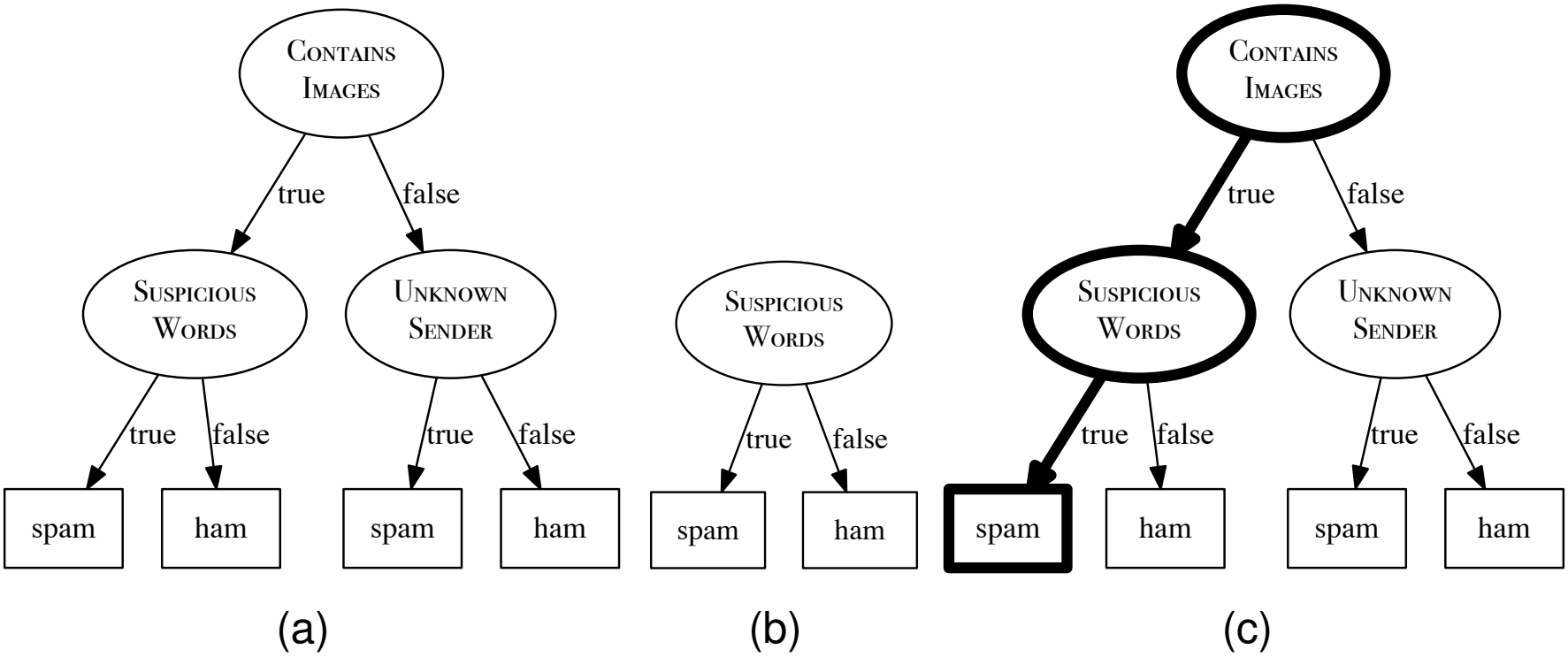


Figure: (a) and (b) show two decision trees that are consistent with the instances in the spam dataset. (c) shows the path taken through the tree shown in (a) to make a prediction for the query instance: SUSPICIOUS WORDS = *'true'*, UNKNOWN SENDER = *'true'*, CONTAINS IMAGES = *'true'*.

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

- We should prefer decision trees that use less tests (shallower trees).
- This is an example of **Occam's Razor**: use the simplest model that is consistent with the data.

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

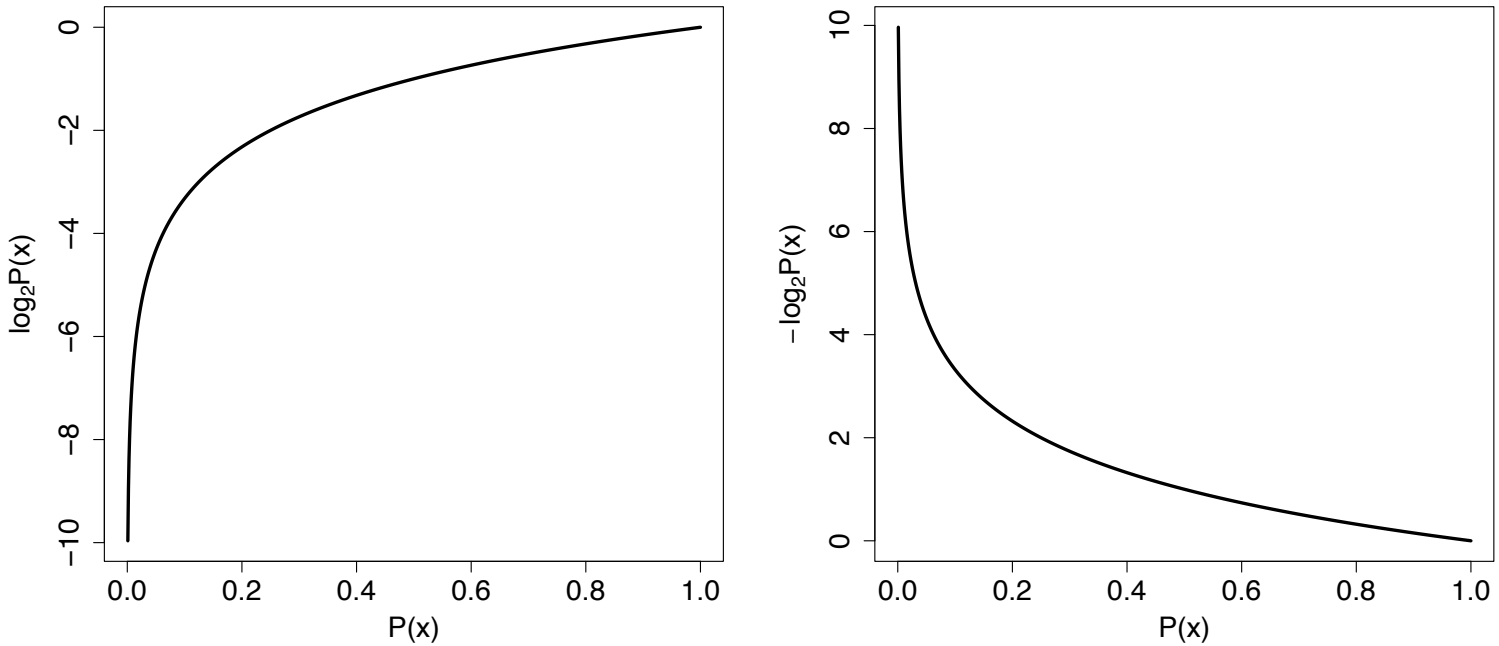
- Claude Shannon's entropy model defines a computational measure of the impurity of the elements of a set.
- An easy way to understand the entropy of a set is to think in terms of the uncertainty associated with guessing the result if you were to make a random selection from the set.

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

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 $2^3 = 8$
- $log_{10}(100) = 2$ because $10^2 = 100$
- $log_{10}(1000) = 3$ because $10^3 = 1000$
- How to change the base: $log_a(x) = \frac{log_b(x)}{log_b(a)}$
- Example: $log_2(x) = \frac{log_{10}(x)}{log_{10}(2)}$. This will be handy for entropy calculations with a calculator.



(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.

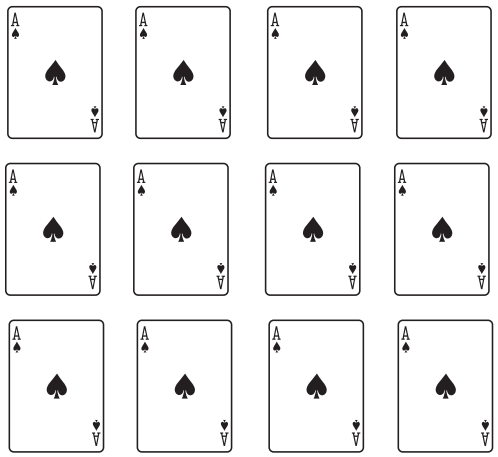
$$H(t) = -\sum_{i=1}^{I} (P(t = i) \times log_2(P(t = i))) \quad (1)$$

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

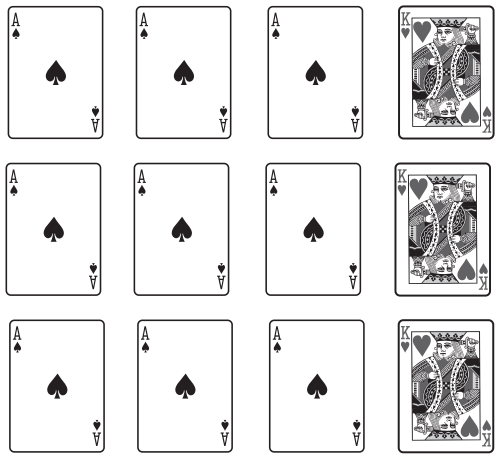
$$
\begin{aligned}
H(card) &= -\sum_{i=1}^{52} P(card = i) \times log_2(P(card = i)) \\
&= -\sum_{i=1}^{52} 0.019 \times log_2(0.019) = -\sum_{i=1}^{52} -0.1096 \\
&= 5.700 \; bits
\end{aligned}
$$

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

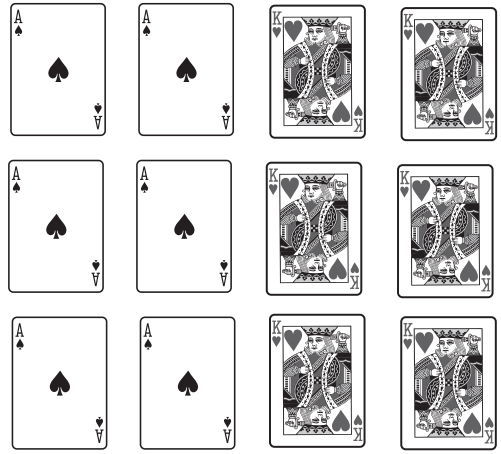
$$
\begin{aligned}
H(suit) = & -\sum_{l \in \{\heartsuit, \clubsuit, \diamondsuit, \spadesuit\}} P(suit = l) \times log_2(P(suit = l)) \\
= & -\big((P(\heartsuit) \times log_2(P(\heartsuit))) + (P(\clubsuit) \times log_2(P(\clubsuit))) \\
& \quad + (P(\diamondsuit) \times log_2(P(\diamondsuit))) + (P(\spadesuit)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



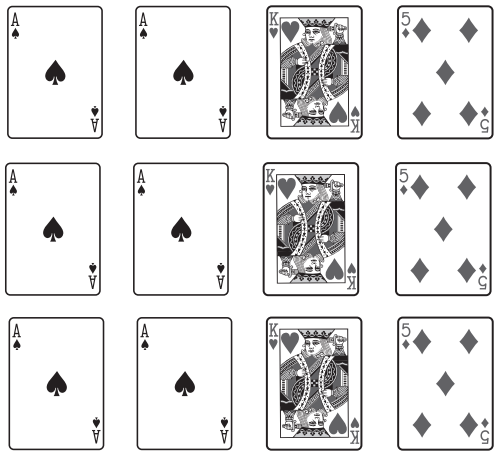
(a) $H(card) = 0.00$



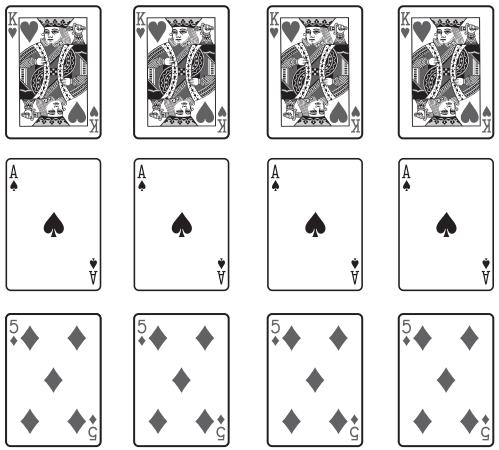
(b) $H(card) = 0.81$



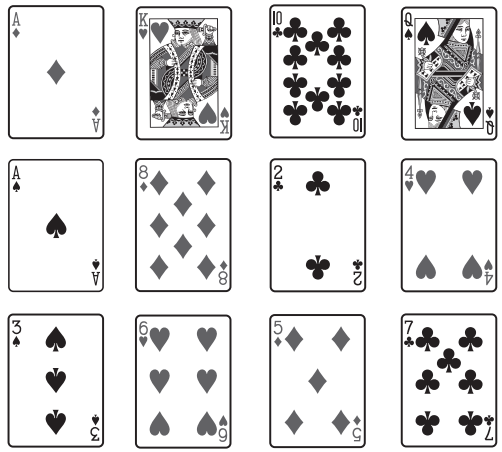
(c) $H(card) = 1.00$



(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.

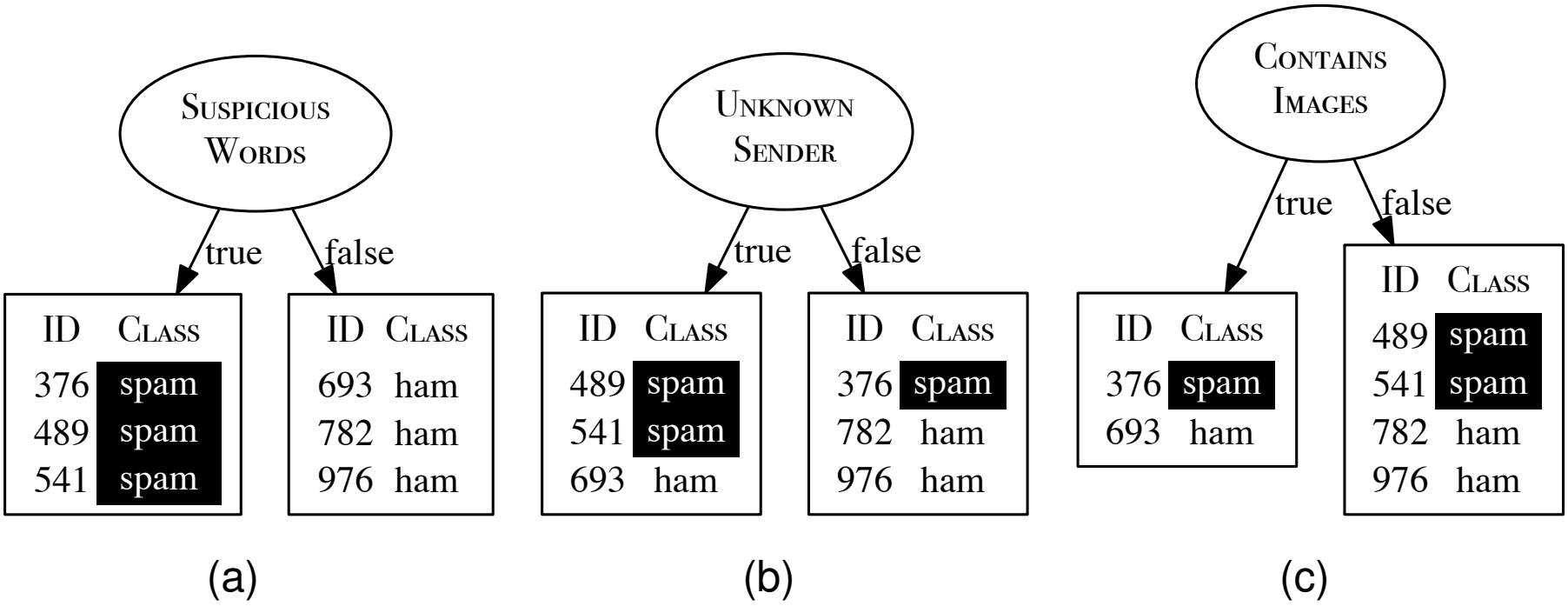


Figure: How the instances in the spam dataset split when we partition using each of the different descriptive features from the spam dataset in Table 1 [7]

- 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

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

- 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

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

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

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

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

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

- Calculate the 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

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

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

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

$$
\begin{aligned}
IG(\text{SUSPICIOUS WORDS}, \mathcal{D}) &= H(\text{CLASS}, \mathcal{D}) - rem(\text{SUSPICIOUS WORDS}, \mathcal{D}) \\
&= 1 - 0 = 1 \textit{ bit} \\
IG(\text{UNKNOWN SENDER}, \mathcal{D}) &= H(\text{CLASS}, \mathcal{D}) - rem(\text{UNKNOWN SENDER}, \mathcal{D}) \\
&= 1 - 0.9183 = 0.0817 \textit{ bits} \\
IG(\text{CONTAINS IMAGES}, \mathcal{D}) &= H(\text{CLASS}, \mathcal{D}) - rem(\text{CONTAINS IMAGES}, \mathcal{D}) \\
&= 1 - 1 = 0 \textit{ bits}
\end{aligned}
$$

- The results of these calculations match our intuitions.

Handling Continuous Descriptive Features

- The easiest way to handle continuous valued descriptive features is to turn them into boolean features by defining a threshold and using this threshold to partition the instances based their value of the continuous descriptive feature.
- How do we set the 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 then 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 dynamically created new boolean feature can compete with the other categorical features for selection as the splitting feature at that node.
- This process can be repeated at each node as the tree grows.

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

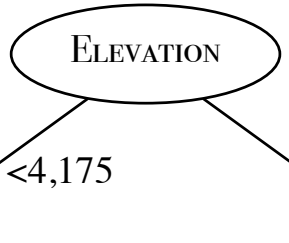
ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	3 900	chapparal
2	true	moderate	300	riparian
3	true	steep	1 500	riparian
4	false	steep	1 200	chapparal
5	false	flat	4 450	conifer
6	true	steep	5 000	conifer
7	true	steep	3 000	chapparal

Table: Dataset for predicting the vegetation in an area sorted by the continuous ELEVATION feature.

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

Table: 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		



	ID	Stream	Slope	Elevation	Vegetation
D7	2	true	moderate	300	riparian
	4	false	steep	1,200	chaparral
	3	true	steep	1,500	riparian
	7	true	steep	3,000	chaparral
	1	false	steep	3,900	chaparral

	ID	Stream	Slope	Elevation	Vegetation
D8	5	false	flat	4,450	conifer
	6	true	steep	5,000	conifer

Figure: The vegetation classification decision tree after the dataset has been split using ELEVATION $\geq$ 4 175.

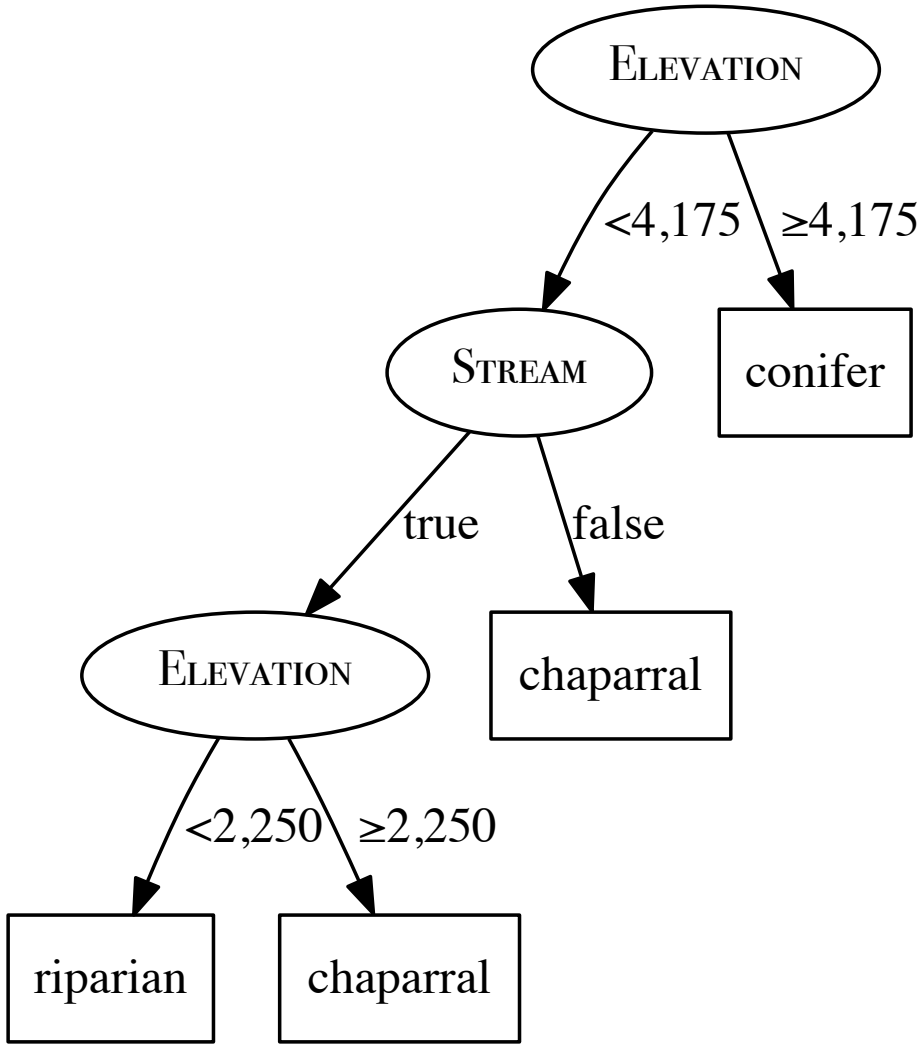


Figure: The decision tree that would be generated for the vegetation classification dataset listed in Table 4 [42] using information gain.

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.
6. The prediction at a leaf node is made by a majority voting of the instance labels at that leaf (with ties broken arbitrarily). Labels of the instances at a leaf can also be used to compute probabilities for each target feature label.
   - Example: 3 instances in a leaf node with 2 spam labels and 1 ham label: Pr(spam) = 0.67, Pr(ham)=0.33, with prediction=spam.

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.

Model Ensembles

- Rather than creating a single model, these methods generate a set of models and then make predictions by aggregating the outputs of these models.
- A prediction model that is composed of a set of models is called a **model ensemble**.
- In order for this approach to work the models that are in the ensemble must be different from each other.

Bagging

- When we use **bagging** (or **bootstrap aggregating**) each model in the ensemble is trained on a random sample of the dataset known as **bootstrap samples**.
- Each random sample is the same size as the dataset and **sampling with replacement** is used.
- Consequently, every bootstrap sample will be missing some of the instances from the dataset so each bootstrap sample will be different and this means that models trained on different bootstrap samples will also be different

- When bagging is used with decision trees each bootstrap sample only uses a randomly selected subset of the descriptive features in the dataset. This is known as **subspace sampling**.
- The combination of bagging, subspace sampling, and decision trees is known as a **random forest** model.

How to calculate the probability of a model ensemble that uses simple majority vote making an incorrect prediction in the following two scenarios using the binomial distribution:

The ensemble contains 3 independent models, all of which have an error rate of 40%.

$$\begin{aligned}\Pr(\text{incorrect prediction}) &= \binom{3}{2}(0.4)^2(0.6)^1 + \binom{3}{3}(0.4)^3(0.6)^0 \\ &= 3*0.16*0.6 + 1*0.064 \\ &= 0.0288 + 0.064 \\ &= \mathbf{0.35}\end{aligned}$$

Alternatively, compute directly: (A: Accurate prediction, N: Inaccurate prediction)

$$\begin{aligned}\Pr(\text{incorrect prediction}) &= \Pr(\text{NNA}) + \Pr(\text{ANN}) + \Pr(\text{NAN}) + \Pr(\text{NNN}) \\ &= 3*0.16*0.6 + 0.064 \\ &= 0.35\end{aligned}$$

The ensemble contains 5 independent models, all of which have an error rate of 40%.

$$\begin{aligned}\Pr(\text{incorrect prediction}) &= \binom{5}{3}(0.4)^3(0.6)^2 + \binom{5}{4}(0.4)^4(0.6)^1 + \binom{5}{5}(0.4)^5(0.6)^0 \\ &= 10*0.064*0.36 + 5*0.0256*0.6 + 1*0.01 \\ &= 0.02304 + 0.0768 + 0.01024 \\ &= \mathbf{0.32}\end{aligned}$$

Based on the above calculations, the ensemble with 5 independent models would be selected because it gives a lower probability for an incorrect prediction.

Figure: Why ensembles outperform single classifiers in general

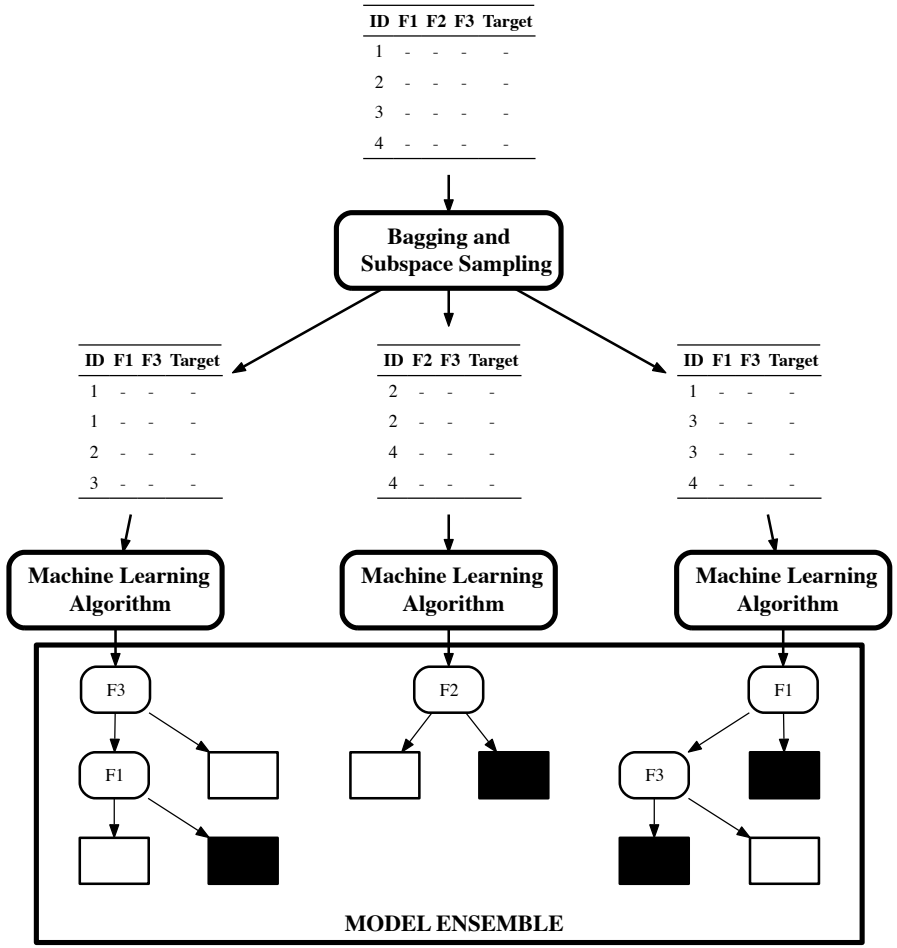


Figure: The process of creating a model ensemble using bagging and subspace sampling.

Summary

- The **decision tree** model makes predictions based on sequences of tests on the descriptive feature values of a query
- The **ID3** algorithm as a standard algorithm for inducing decision trees from a dataset.
- **Random forests**, which are ensembles of decision trees, are some of the most powerful ML methods out there. They are relatively easy to train, have very few parameters to fine-tune, and outperform most other methods in practice.

Decision Trees: Advantages

- interpretable.
- handle both categorical and continuous descriptive features.
- has the ability to model the interactions between descriptive features
- relatively, robust to the **curse of dimensionality**.
- relatively, robust to noise in the dataset if **ensembles** are used.

Decision Tress: Potential Disadvantages

- trees become large when dealing with continuous features.
- decision trees are very expressive and sensitive to the dataset, as a result they can overfit the data if there are a lot of features (curse of dimensionality)
- eager learner (concept drift).

1. **Fundamentals**
   - Decision Trees
   - Shannon's Entropy Model
   - Information Gain
2. **Handling Continuous Descriptive Features**
3. **Standard Approach: The ID3 Algorithm**
4. **Model Ensembles**
5. **Summary**