

Machine Learning



Machine Learning

Lecture: Decision Trees

Ted Scully

Decision Trees

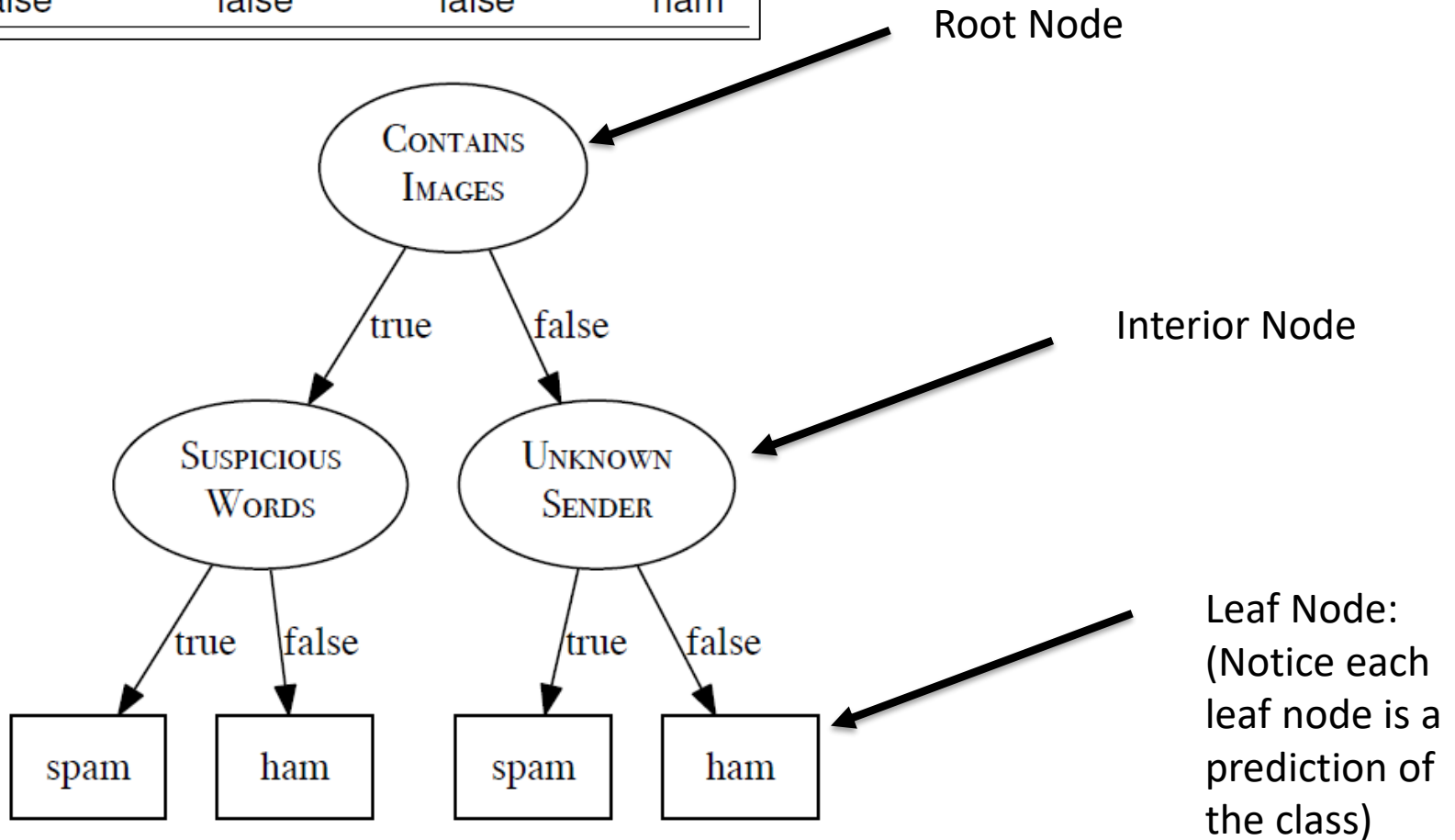
- A decision tree is a **graphical representation of a decision process**.
 - A decision tree is a machine learning model that takes as input a vector of feature values and returns a prediction – a single output value
 - The **nodes** in the decision tree represent the **features** of your dataset and the **leaves** represent the predicted **class**.
- A useful aspect of a decision tree model is that they are **easy to read** when presented in graphical form and can provide valuable insights into the problem domain.

Structure of a Decision Tree

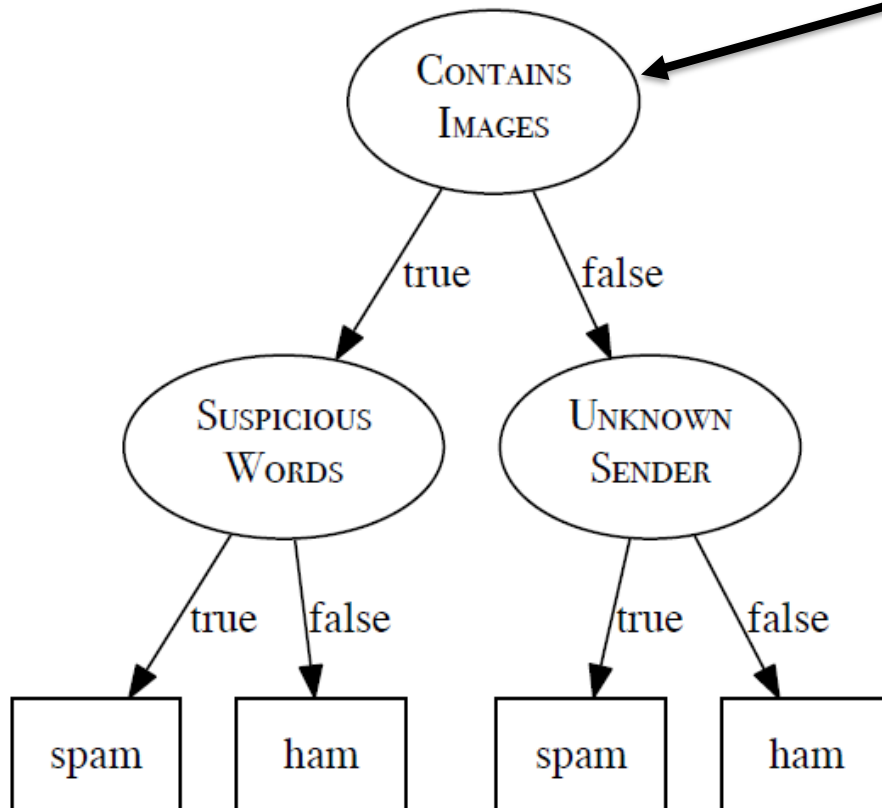
- A decision tree consists of:
 - a root node (or starting node),
 - interior nodes
 - and leaf nodes (or terminating nodes).
- Each of the non-leaf nodes (root and interior) in the tree specifies a **test** to be carried out on one of the query's **descriptive features**.
- Each of the **leaf nodes**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

SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham

SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham



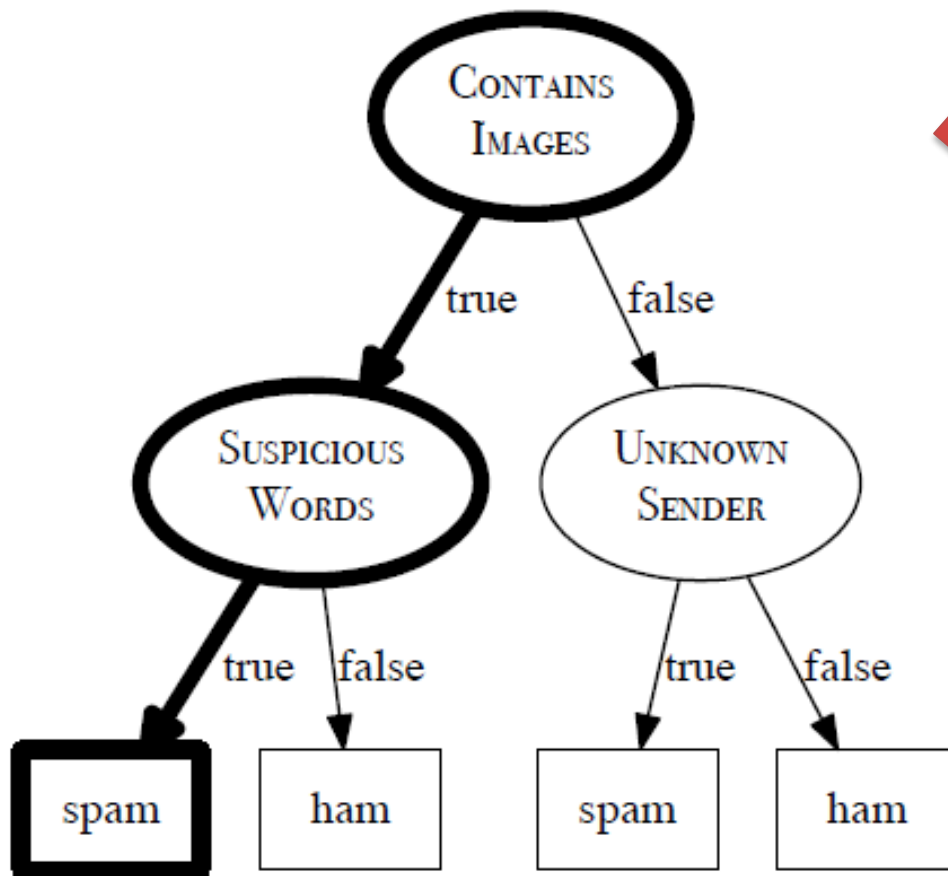
SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham



Remember each non-leaf node in a tree is a test on the data. The first test on the data here is on the feature “Contains Images”.

It is important to understand that this test serves to **partition the data**.

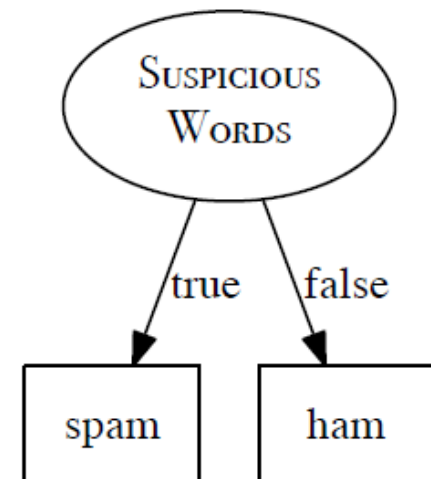
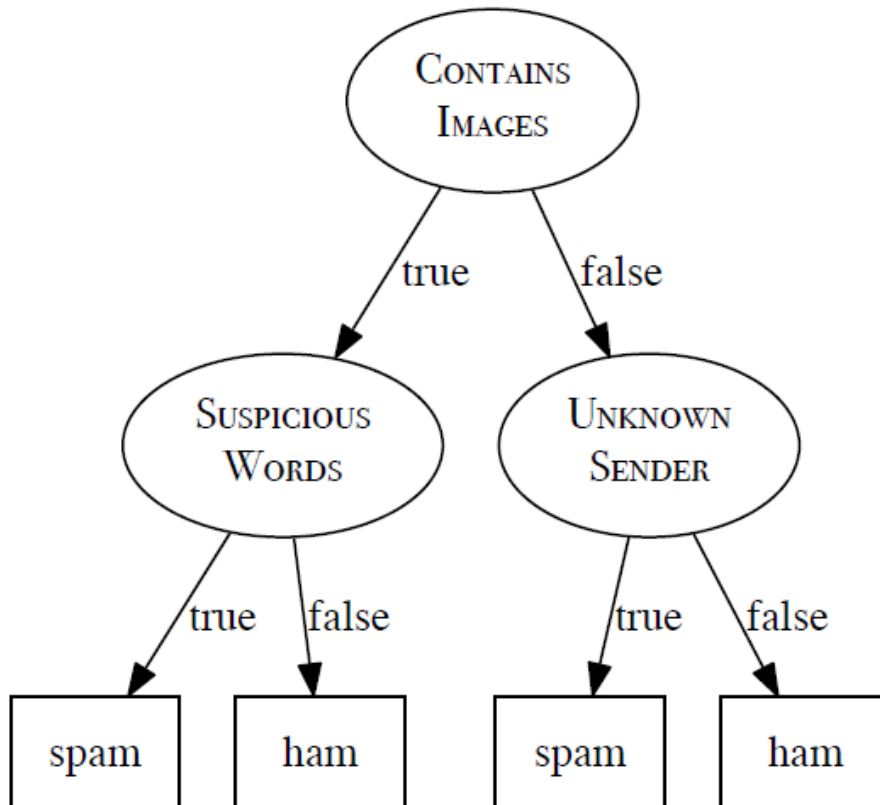
SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham



The process of using a decision tree to make a prediction for a query instance starts by testing the value of the descriptive feature at the root node of the tree. The result then determines which of the two routes we should descend. This process is repeated until we reach a leaf (a prediction).

Structure of a Decision Tree

- Now consider the second shorter decision tree depicted on the right.
- Both of the decision trees below perfectly classify the training data.
- Which is preferable?

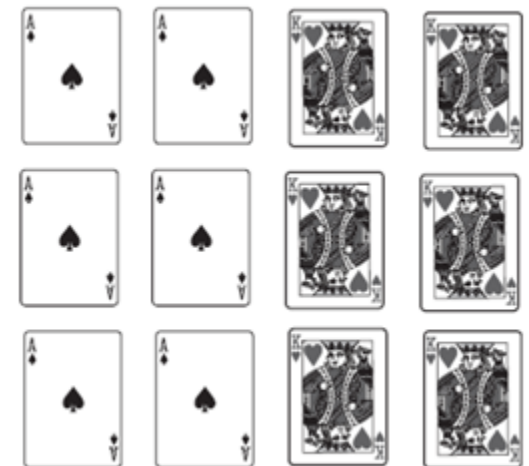
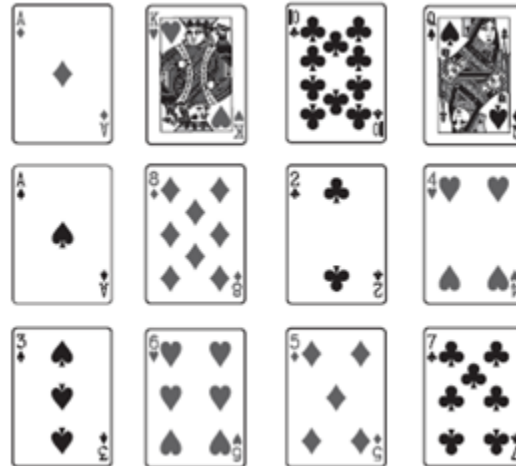
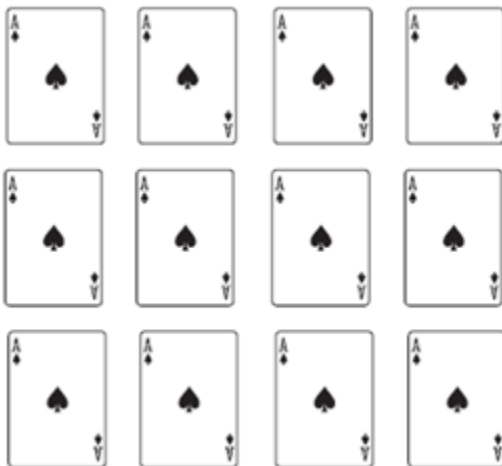


Structure of a Decision Tree

- The second is preferable as we only use a **single feature** to split the data and we only have one test as opposed to two. In fact what we see from the previous slide is that the “Suspicious Words” feature **perfectly splits** the data in ham and spam emails.
- Therefore, “**Suspicious Words**” as a feature provides **more information** about the value of a target feature than the “**Contains Images**” feature. Therefore, it is more appropriate to use it as the **root node**.
- So we can make shallow trees by testing the features and placing the more informative features earlier in the decision tree.
- Therefore, we need some method for ranking or quantifying which features carry more information about the target.
- One of the most common way of achieving this is called **Shannon’s Entropy Model**.

Shannon Entropy Metric

- 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.
- An entropy value of zero means zero uncertainty, the higher the entropy value the higher the level of uncertainty.



Entropy

- Now let's consider entropy in the context of a simple dataset.
- What is the entropy of the Surf feature in the dataset below when Swell = Big.
 - $\text{Ent}(\text{Surf}, \text{Swell} = \text{Big}) = 0$
- Every time the swell is big we go surf.
- The swell feature has two values. What is the entropy for **$\text{Ent}(\text{Surf}, \text{Swell} = \text{Small})$**

Rain	Swell	Surf
Yes	Big	Yes
Yes	Small	No
No	Small	No
No	Big	Yes

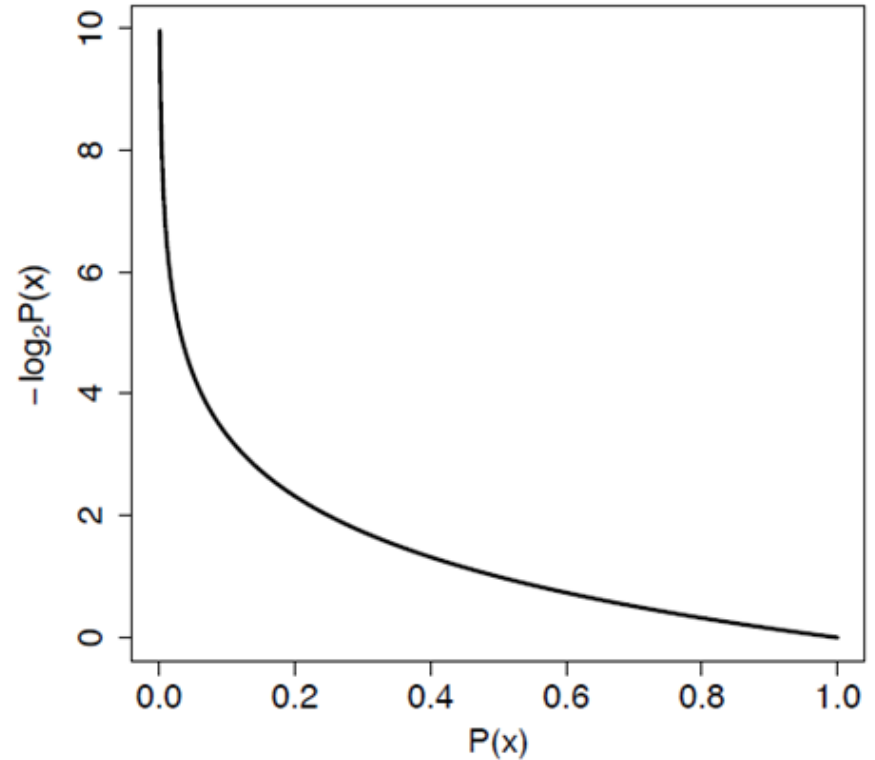
Entropy

- The entropy for the Surf feature when Rain = Yes is high because there is a high level of uncertainty.
 - $\text{Ent}(\text{Surf}, \text{Rain} = \text{Yes}) = 1$.
- In other words when it was raining, we went surfing on one occasion and once we didn't go surfing. Therefore the influence of this feature on whether or not we go surfing is highly uncertain.
- What is the entropy for the rain feature when it's value is No $\text{Ent}(\text{Surf}, \text{Rain} = \text{'No'})$.

Rain	Swell	Surf
Yes	Big	Yes
Yes	Small	No
No	Small	No
No	Big	Yes

Entropy

- Shannon's entropy is a **weighted sum** of the **logs** of the **probabilities** of each possible outcome when we make a random selection from a set.
- The weights used in the sum are the probabilities of the outcomes themselves so that the outcomes with the higher probabilities contribute more to the overall entropy.



Entropy

We can calculate the entropy of the sample S as:

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

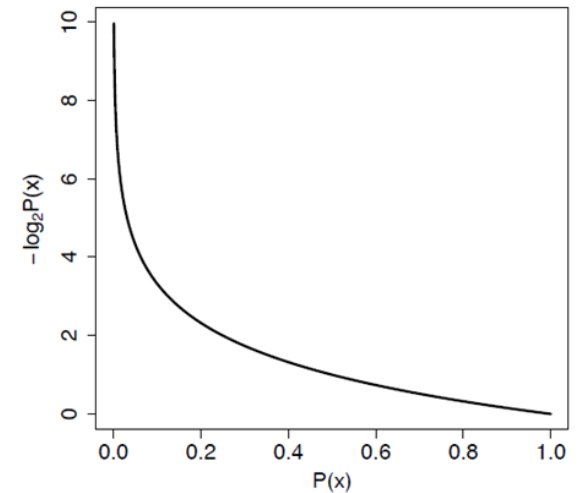
For a sample S with n different values, the probability of value i is p_i (number of instances of value i divided by total number of instances in the sample S)

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

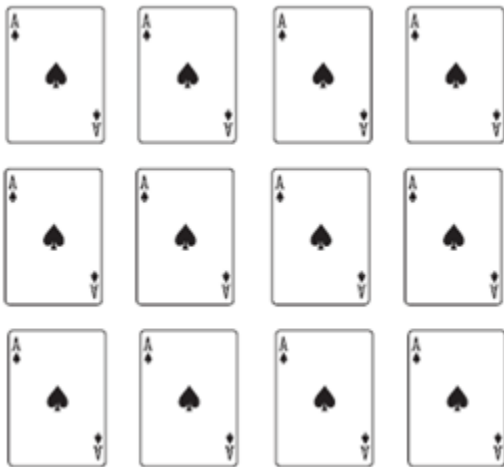
$$\text{Ent}(s_1) = -(12/12)\log_2 (12/12) = 0$$

$$\text{Ent}(s_2) = - (1/12)(\log_2 (1/12)) * 12 = 3.58$$

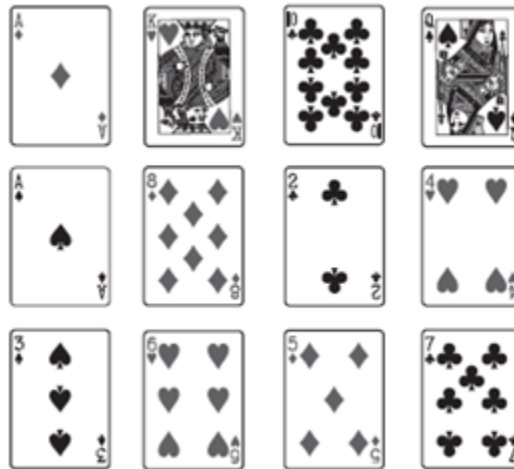
$$\text{Ent}(s_3) = - (6/12)(\log_2 (6/12)) - (6/12) (\log_2 (6/12)) = 1$$



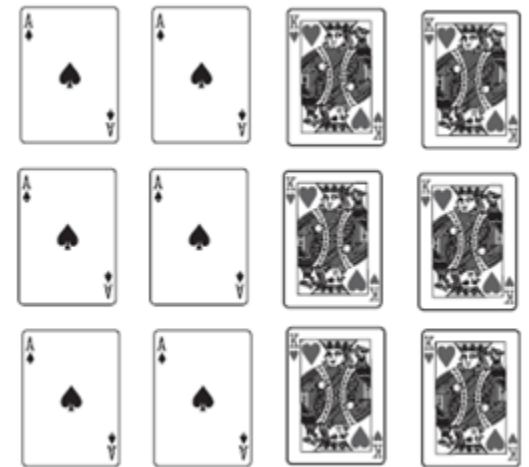
s1



s2



s3



Entropy Example

- In the example below I'm going to determine the entropy for the entire dataset S

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

- There are two features in our dataset, therefore $n=2$.
- **Ent(Surf)** = $((-2/4) \log_2(2/4)) + ((-2/4) \log_2(2/4)) = 1$

Rain	Swell	Surf
Yes	Big	Yes
Yes	Small	No
No	Small	No
No	Big	Yes

Entropy Example

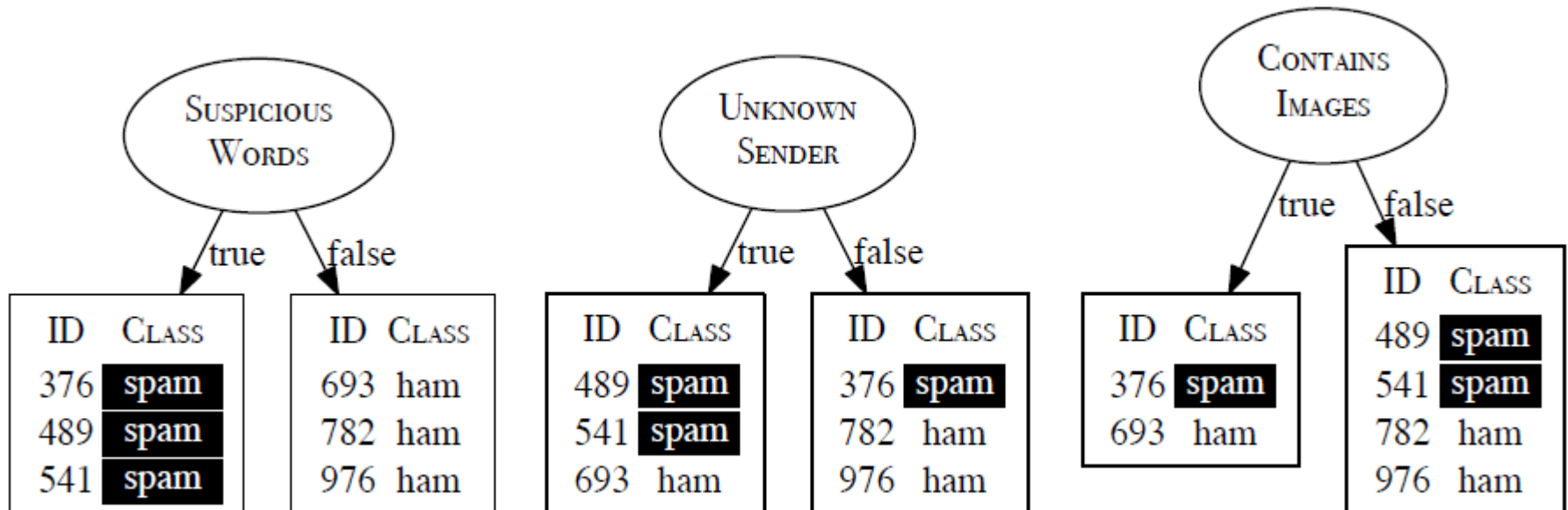
$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

- Now let's examine the entropy of the swell feature when Swell = Big.
- **Ent(Surf, Swell=Big)** = $(-2/2) \log_2(2/2) = 0$ (No uncertainty in this data)

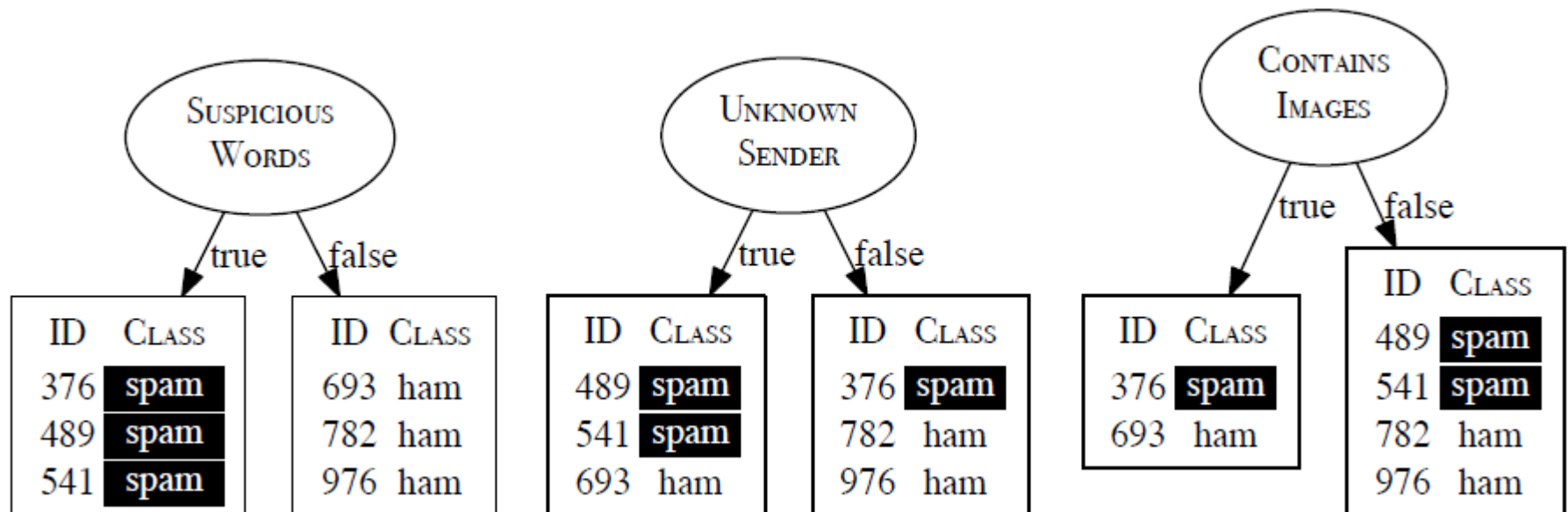
Rain	Swell	Surf
Yes	Big	Yes
Yes	Small	No
No	Small	No
No	Big	Yes

Obtaining Information Gain

- So what is the relationship between using entropy and building a decision tree?
- As we can see below using the feature “Suspicious Words” allows us to split the target class perfectly (so we say this feature provides a high level of **information gain** about the target class). Which feature is the second best predictor in example below and why?
- Therefore, if we have some way of quantifying the information gain provided by a feature then we can rank and select the best feature to split the data.
- But how do we quantify the information of one feature compared to that of another?



SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham



Obtaining Information Gain

- A common method used to quantify the level of information carried by a feature for a specific class is called Information Gain.
- **Information Gain is a measure of the reduction in the overall entropy of a set of instances that is achieved through testing a specific feature.**
- Computing information gain is broken into three steps:
 - Compute the entropy of the original sample with respect to the feature.
 - For each descriptive feature, create the sets that result by partitioning the instances in the sample using their feature values. Then sum the entropy scores of each of these sets.
 - Subtract the entropy value after we have split the sample from the original entropy to give the information gain.

Information Gain

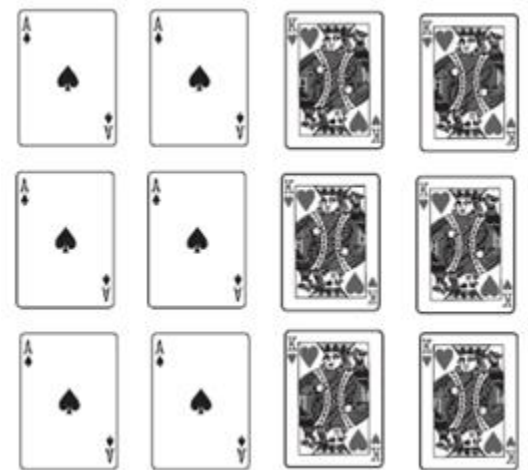
- Information Gain of an feature is the reduction in Entropy from partitioning the data according to values of that feature.

$$\text{Gain}(S, A) = \text{Ent}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Ent}(S_v)$$

Subset of S
where A has
value v

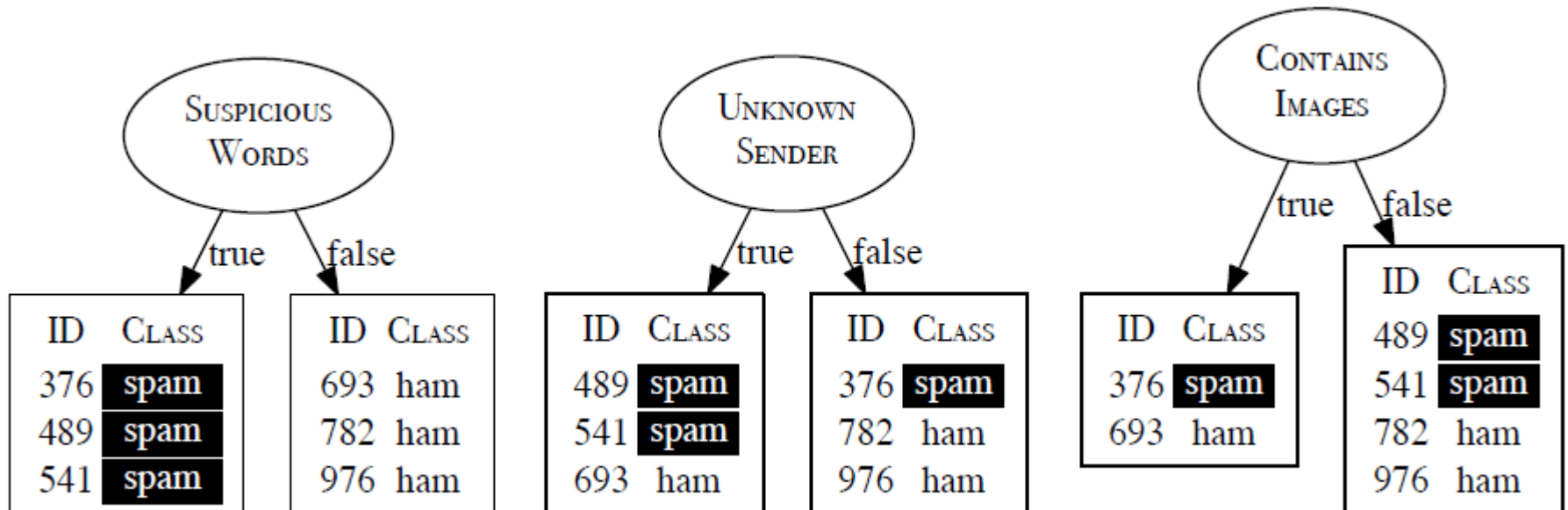
Size of S

Use whichever feature A gives the greatest Gain



Recap

SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham



High Level Steps of Creating a Decision Tree

- High Level Steps for Creating a Decision Tree
 1. Begin **with a dataset** containing all features and resulting classes
 2. Find the **feature that best splits** the dataset class using information gain
 3. Divide the data in groups based on the feature that you have used to split (**node**)
 4. With **each group** find the feature to best **split the data** set class. Continue this process.

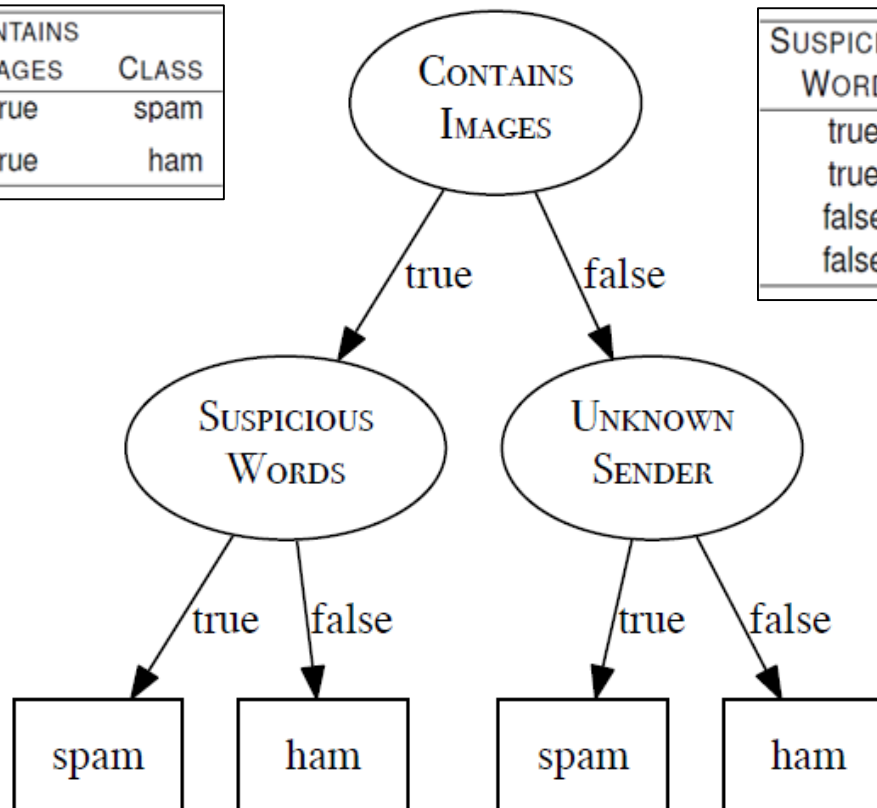
SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
true	true	false	spam
true	true	false	spam
false	true	true	ham
false	false	false	ham
false	false	false	ham

true

false

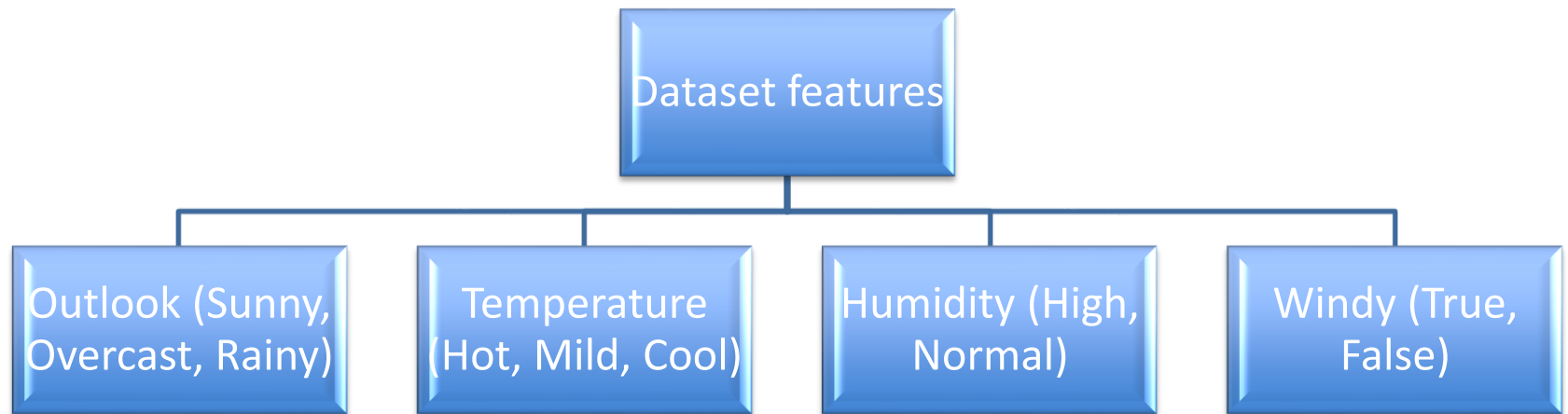
SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	false	true	spam
false	true	true	ham

SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
true	true	false	spam
true	true	false	spam
false	false	false	ham
false	false	false	ham



Weather Dataset features

- We will use the tennis dataset for illustrating the operation of decision trees
- Weather dataset contains four features that are used to decide whether or not to play tennis



- Objective: Find an hypothesis that describes the cases given and can be used to make decisions in other cases
- Notice that all features above are discrete

Weather Dataset

Anyone for Tennis?					
ID	Outlook	Temp	Humidity	Windy	Play?
A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

There are 36 ($3 \times 3 \times 2 \times 2$) possible instances with the dataset depicting 14 of those.

Entropy Examples

- In the following slides we will look more closely at the entropy calculations for our weather dataset:
 - **Example A** – We calculate the **entropy value of the dataset as a whole**, in other words the entropy of the weather dataset S .
 - **Example B** – We calculate the **entropy value of the Windy feature**. We calculate the entropy for $\text{Windy} = \text{false}$ and for $\text{Windy} = \text{true}$

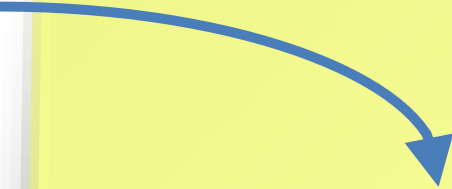
Entropy Example A – Entire Dataset

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

- The dataset S refers to our Weather dataset, which has two classes (yes and no).
- We must first determine **the probability of each class i (p_i) in S .**
 - There is a total of **14 examples** in the dataset (the class **yes occurs 9 times** and the class **no occurs 5 times**.)
 - Therefore, the probability of class yes is simply **9/14** and the probability of class no is **5/14**. We can now plug these values into the formula and calculate the entropy of the weather dataset.

Entropy Example A – Entire Dataset

S refers to
the
Weather
dataset.



$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

$$\begin{aligned}\text{Ent}(S) &= -9/14 \log_2(9/14) - \\ &\quad 5/14 \log_2(5/14) \\ &= 0.940\end{aligned}$$

Entropy Example B – Windy feature

- If we were to divide the data based on the feature Windy. We could need to assess how well this feature splits the data.
- We need to calculate the entropy for the **Windy = true** subset and the **Windy = false** subset.
- In order to calculate the entropy for the **Windy = true** subset we must extract the result (play tennis 'yes' or 'no') when **Windy = true**.
 - This subset is highlighted in green on the next slide.
 - In order to calculate the entropy for the Windy = false subset we must extract the result when Windy = false. This subset is highlighted in yellow in the slide.

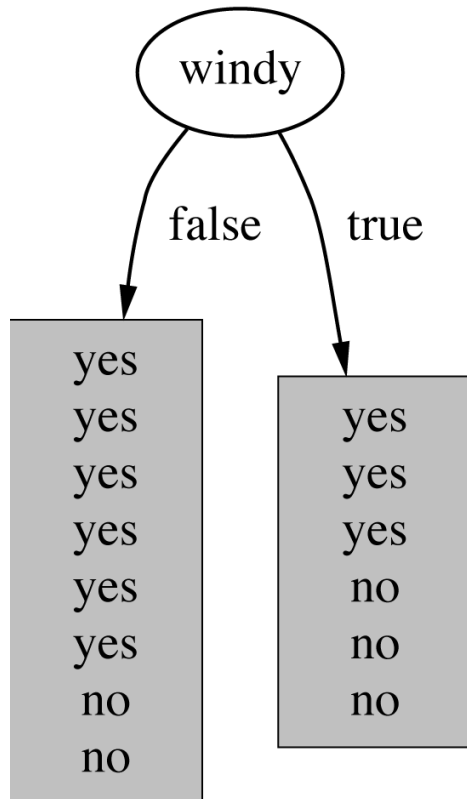
Entropy Example B

Anyone for Tennis?

ID	Outlook	Temp	Humidity	Windy	Play?
A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

Entropy Example B

The diagram shows the division of the Weather dataset based on the value of the feature windy.



The grouping on the column on the left shows the output (play tennis 'yes' or 'no') for when **Windy = false** and corresponds to the subset highlighted in green in the previous slide.

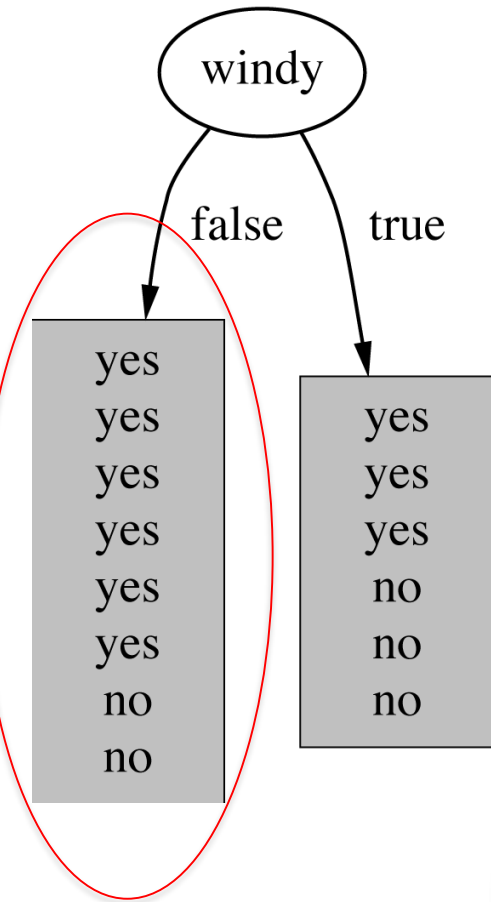
The grouping on the right shows the output for when **Windy = true** and corresponds to the subset highlighted in yellow in the previous slide.

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

Entropy Example B

We now have the information we need to calculate the entropy for **Windy = true** and **Windy = false**.

The **Windy = false** subset we have a total of eight entries, 6 are positive ('yes') and 2 are negative ('no').



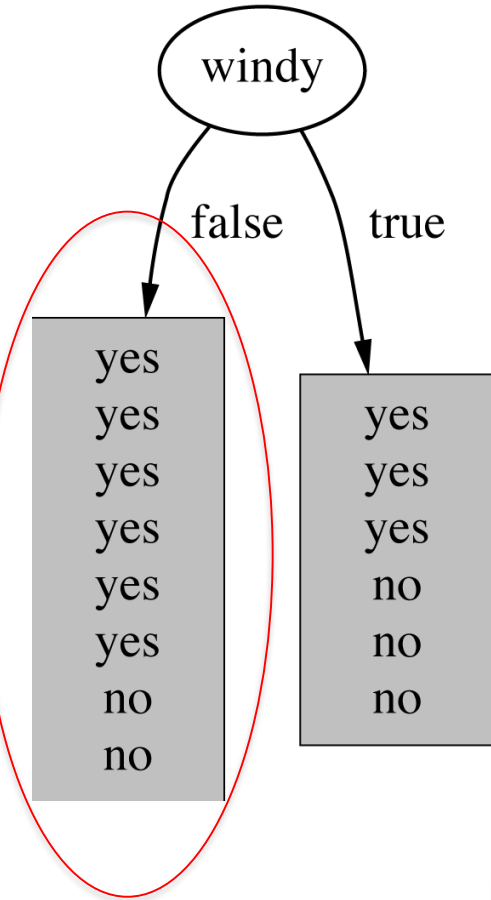
`Ent(S, windy=false)`

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

Entropy Example B

We now have the information we need to calculate the entropy for **Windy = true** and **Windy = false**.

The **Windy = false** subset we have a total of eight entries, 6 are positive ('yes') and 2 are negative ('no').

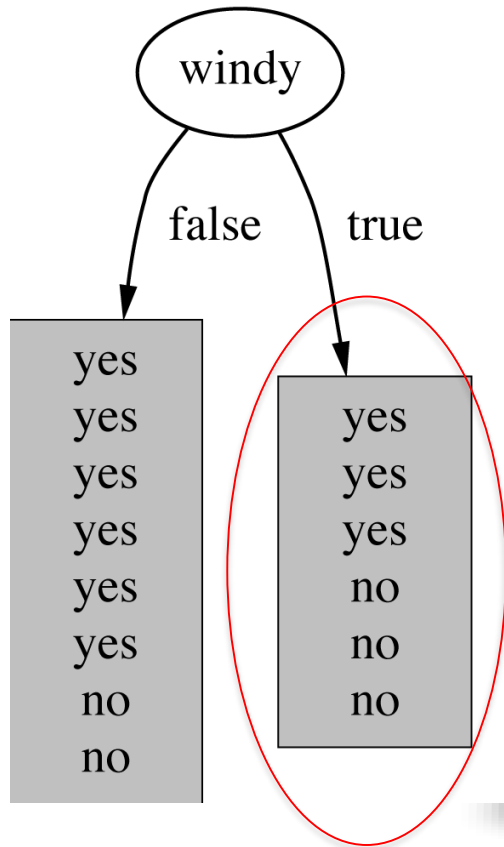


$$\begin{aligned}
 &\text{Ent}(S, \text{ windy=false}) \\
 &= -6/8 \log_2(6/8) - 2/8 \log_2(2/8) \\
 &= 0.3112 + 0.5 = 0.811
 \end{aligned}$$

$$\text{Ent}(S) = \sum_{i=1}^n -p_i \log_2 p_i$$

Entropy Example B

The Windy = true subset we have a total of six entries, 3 are positive ('yes') and 3 are negative ('no'). What would you expect the entropy value to be?



$$\begin{aligned} & \text{Ent}(S, \text{windy}=\text{true}) \\ &= -3/6 \log_2 (3/6) - 3/6 \log_2 (3/6) \\ & \quad 0.5 + 0.5 = 1.0 \end{aligned}$$

Information Gain

- As we mentioned information gain of a feature is the reduction in entropy from dividing the data into two or more subsets based on that feature.
- Therefore, in our Weather example the information gain of the feature **Temperature** would be the overall reduction in entropy caused by **partitioning** the data into three subsets: (i) Those where the value is hot; (ii) Those where the value is mild and (iii) Those where the value is cool.
- Ultimately, we are looking for the feature that would lead to the highest information gain (or put another way that would give the largest reduction in entropy).
 - The higher the information gain of a particular feature then the better that feature partitions the data.

Information Gain

- Information Gain of an feature is the reduction in Entropy from partitioning the data according to that feature.

$$\text{Gain}(S, A) = \text{Ent}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Ent}(S_v)$$

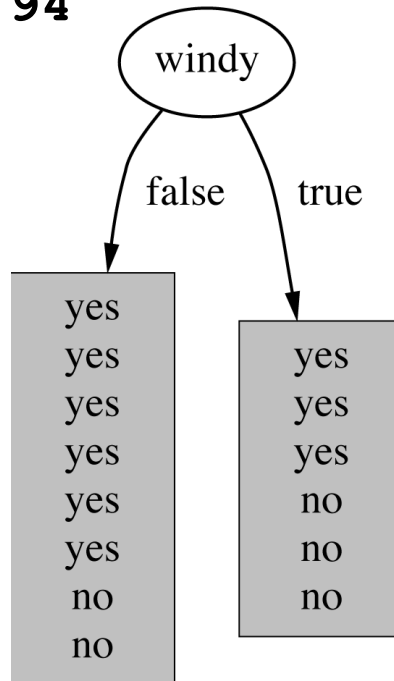
Subset of S
where A has
value v

Size of S

*Use whichever feature A gives
greatest Gain*

$$\text{Gain}(S, A) = \text{Ent}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Ent}(S_v)$$

Ent(S) = 0.94



Ent(S, w=false)=0.811

Ent(S, w=true)= 1.0

Information Gain Example 1

$$\begin{aligned} & \text{InformationGain}(S, \text{Windy}) \\ &= \text{Entropy}(S) - \frac{8}{14} \text{Ent}(S, \text{windy}=\text{false}) \\ & \quad - \frac{6}{14} \text{Ent}(S, \text{windy}=\text{true}) \end{aligned}$$

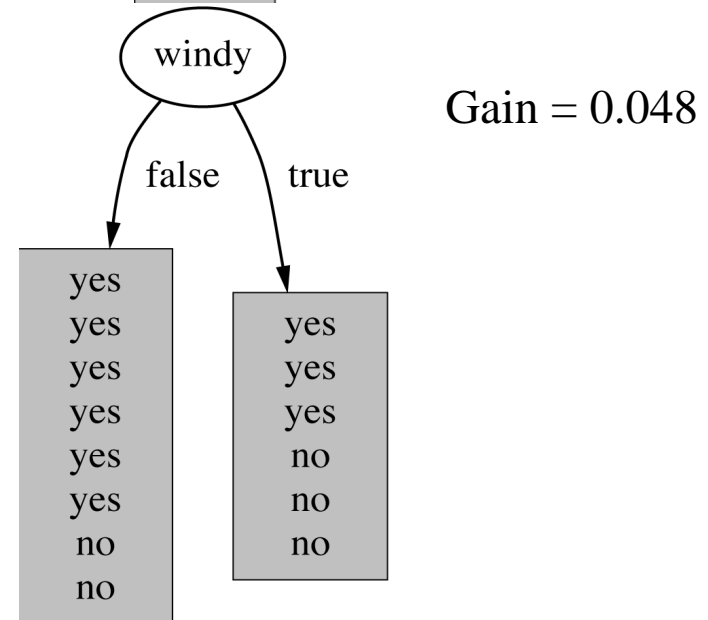
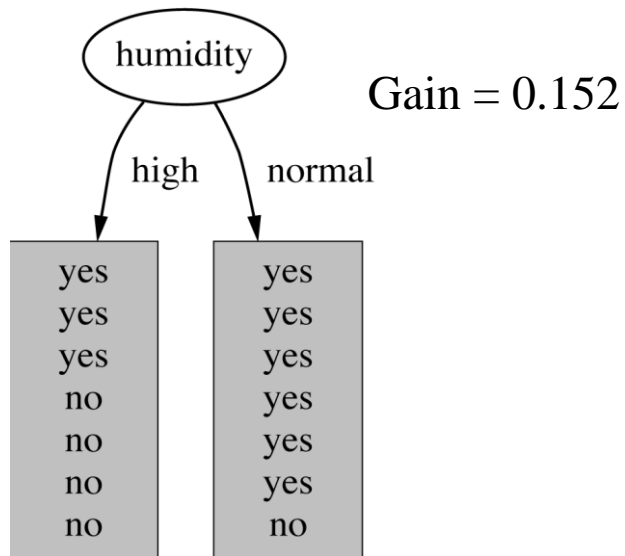
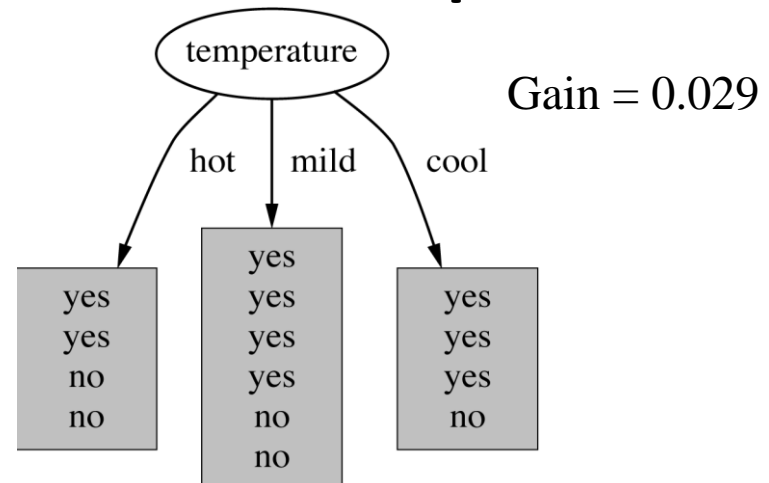
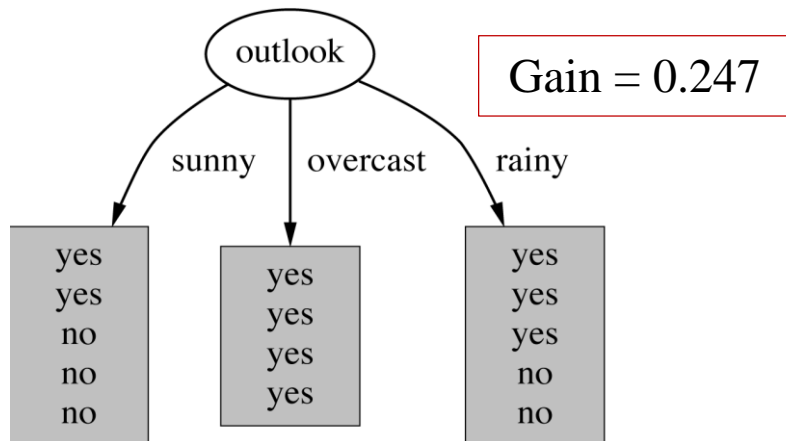
In previous slides we have calculated the entropy value of the weather dataset S as 0.940, the entropy of **Windy = true** as 1.0 and the entropy of **Windy = false** as 0.811. We can now use these values to calculate the information gain for the feature Windy.

Information Gain Example 1

$$\begin{aligned} & \text{InformationGain}(S, \text{Windy}) \\ &= \text{Entropy}(S) - \frac{8}{14} \text{Ent}(S, \text{windy}=\text{false}) \\ & \quad - \frac{6}{14} \text{Ent}(S, \text{windy}=\text{true}) \\ &= 0.940 - \left(\frac{8}{14}\right) 0.811 - \left(\frac{6}{14}\right) 1.00 \\ &= 0.048 \end{aligned}$$

The information gain for Windy is 0.048, which is quite low and means that the Windy feature does not partition the data very well. (The higher the information gain value the better)

Information Gain Example 2



Overview of ID3

Step 1

- For all features that have not yet been used in the tree calculate their **entropy** and **information gain** values for the test samples.

Step 2

- Select the feature that has the highest information gain.

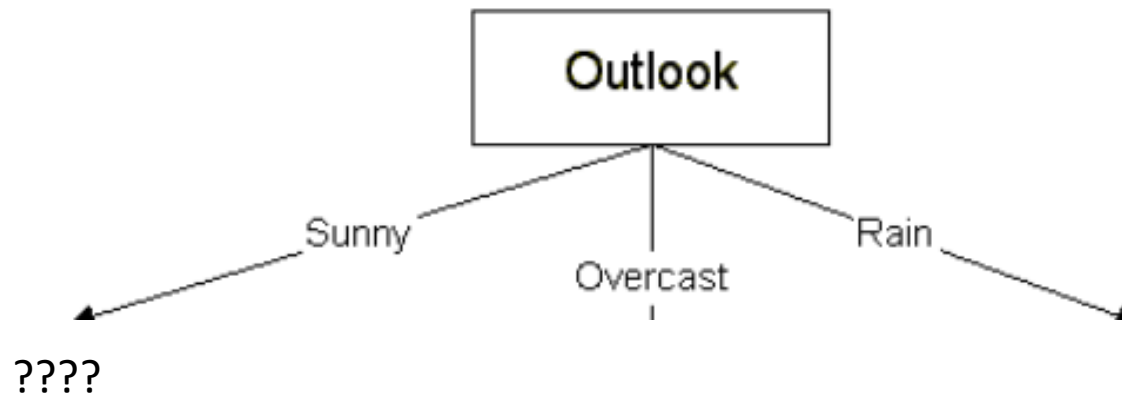
Step 3

- Make a tree node containing that feature.

Before delving into the ID3 algorithm it is important to fully understand the concepts of entropy and information gain.

Adding Nodes to the Decision Tree

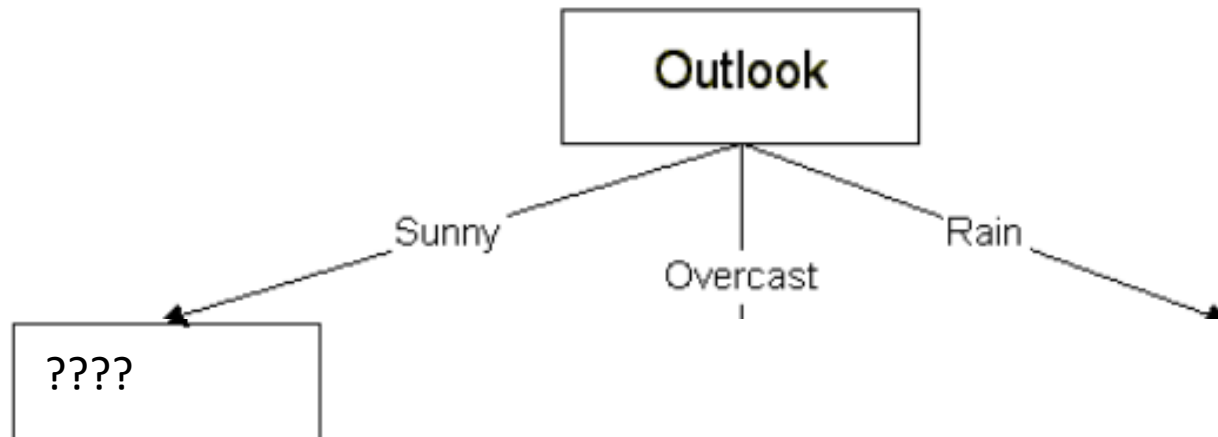
- The previous slides shows the information gain for each feature in the weather dataset.
- The next step is to “select the feature that has the highest information gain” and make it the root node of our decision tree.
- The **outlook feature has the highest information gain** value and therefore becomes a root node.
- We split our tree based on the three possible values of the outlook feature



Adding Nodes to the Decision Tree

- Lets find the best feature to place on the Sunny branch of our decision tree below.
- To do this we will need to calculate the information gain for all remaining features (**Humidity**, **Temperature**, **Windy**) for the data where Outlook = Sunny

ID	Outlook	Temp	Humidity	Windy	Play?
A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no



Information Gain for Humidity feature

- Consider the Gain(Sunny, Humidity). Any observations?

ID	Outlook	Humidity	Play?
A	sunny	high	no
B	sunny	high	no
C	overcast	high	yes
D	rainy	high	yes
E	rainy	normal	yes
F	rainy	normal	no
G	overcast	normal	yes
H	sunny	high	no
I	sunny	normal	yes
J	rainy	normal	yes
K	sunny	normal	yes
L	overcast	high	yes
M	overcast	normal	yes
N	rainy	high	no

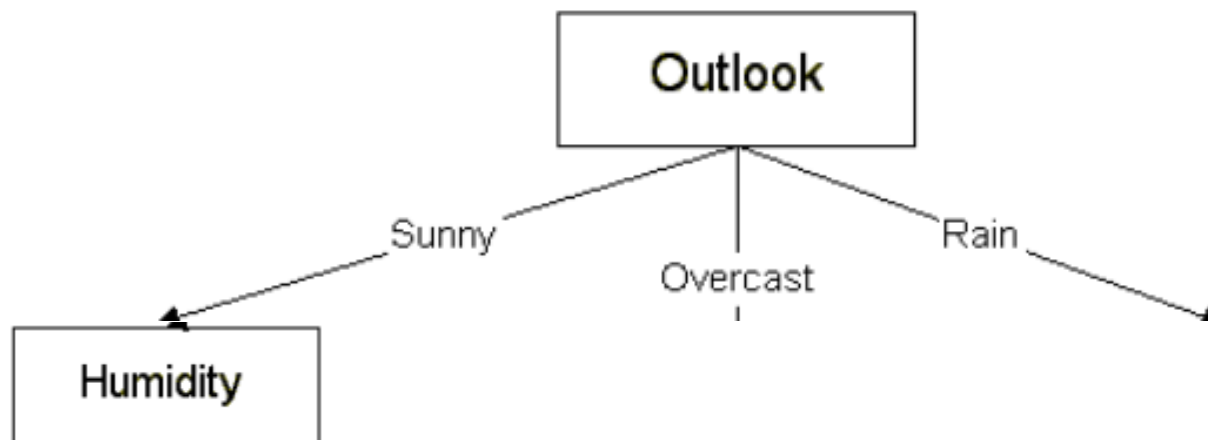
Information Gain for Humidity feature

- $\text{Ent}(S_{\text{sunny}}) = -((2/5) * \log_2(2/5)) - ((3/5) * \log_2(3/5)) = 0.97095059445$
 - (5 Data items where $S_{\text{sunny}} = \text{yes}$ and no (3 where it is no and 2 where it is yes))
- $\text{Ent}(S_{\text{sunny}}, \text{Humidity} = \text{High}) = 0$
 - (3 data items where sunny and humidity = high, of those 3 they all had an outcome of play = no)
- $\text{Ent}(S_{\text{sunny}}, \text{Humidity} = \text{Normal}) = 0$
 - (2 data items where sunny and humidity = normal, of those 2 they all have an outcome of play = yes)
- $\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = 0.97095059445 - ((3/5)0 + (2/5)0) = 0.97095059445$

ID	Outlook	Humidity	Play?
A	sunny	high	no
B	sunny	high	no
C	overcast	high	yes
D	rainy	high	yes
E	rainy	normal	yes
F	rainy	normal	no
G	overcast	normal	yes
H	sunny	high	no
I	sunny	normal	yes
J	rainy	normal	yes
K	sunny	normal	yes
L	overcast	high	yes
M	overcast	normal	yes
N	rainy	high	no

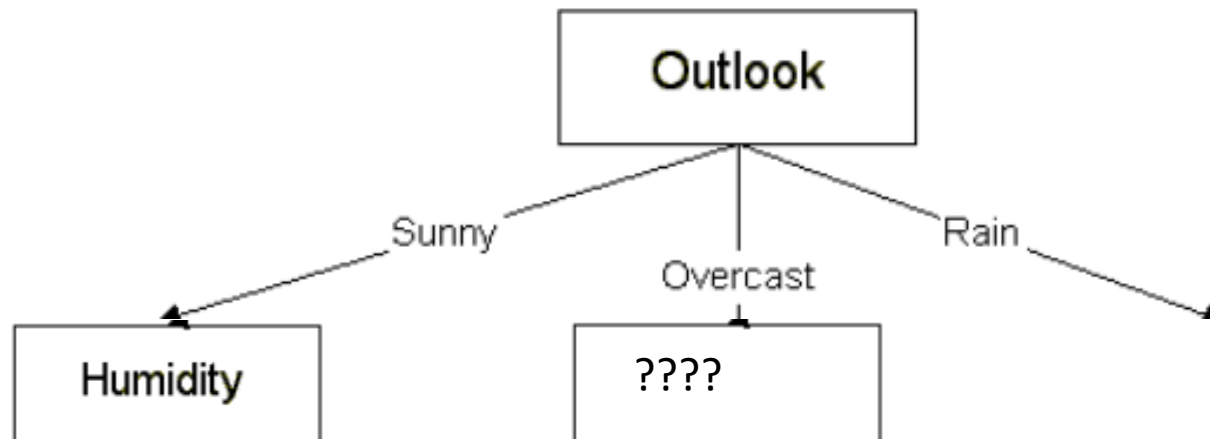
Adding Nodes to the Decision Tree

- Lets now look at applying ID3 to the subset of the dataset where Outlook is Sunny
 - There are 5 examples from table 1 with outlook = sunny
 - $\text{Gain}(\text{Sunny}, \text{Humidity}) = 0.970$
 - $\text{Gain}(\text{Sunny}, \text{Temperature}) = 0.570$
 - $\text{Gain}(\text{Sunny}, \text{Wind}) = 0.019$
- What should be the new node added to the decision tree



Adding Nodes to the Decision Tree

- Lets find the best feature to place on the **Overcast** branch of our decision tree below.



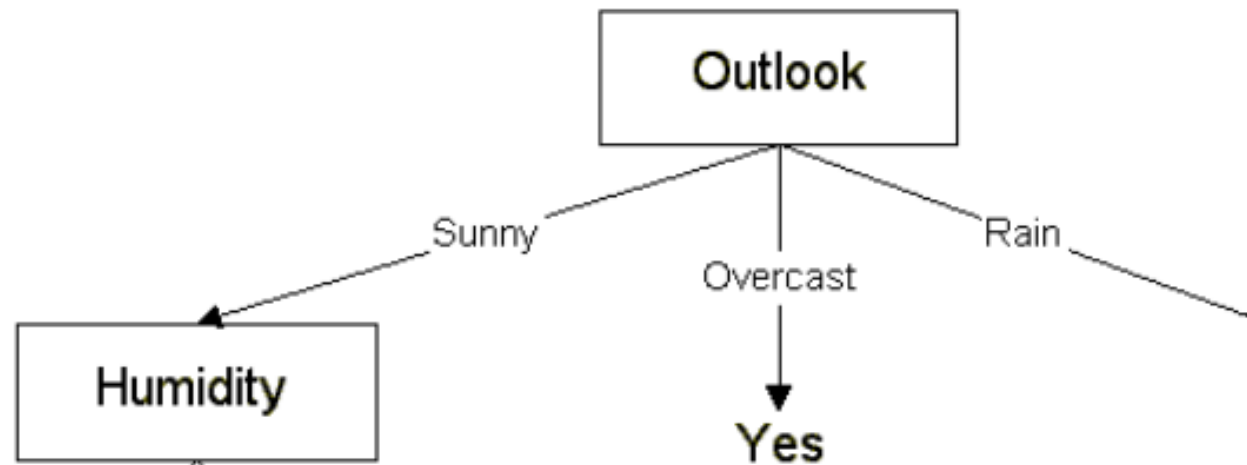
Adding Nodes to the Decision Tree

- Any observations on the subset of the dataset where outlook is overcast?

B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

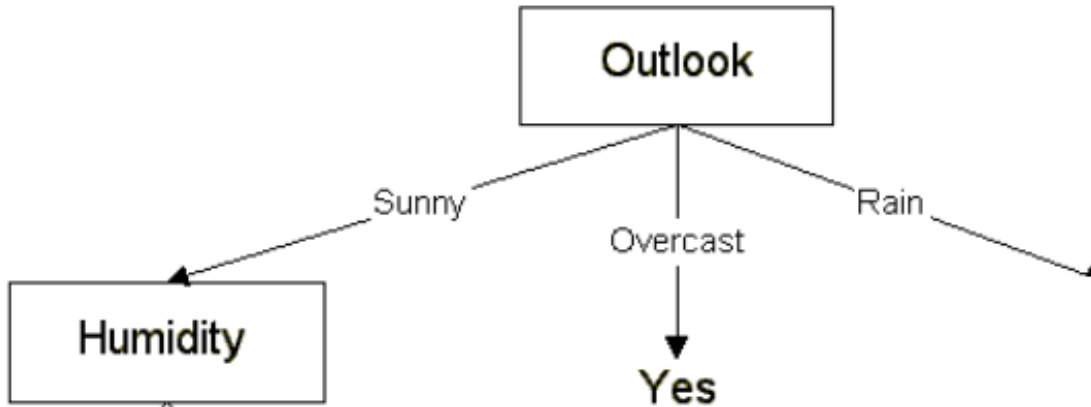
When we come to apply ID3 to the subset where Outlook=overcast we will find that further partitioning of this subset is not needed, as all the cases have the same class

Adding Nodes to the Decision Tree



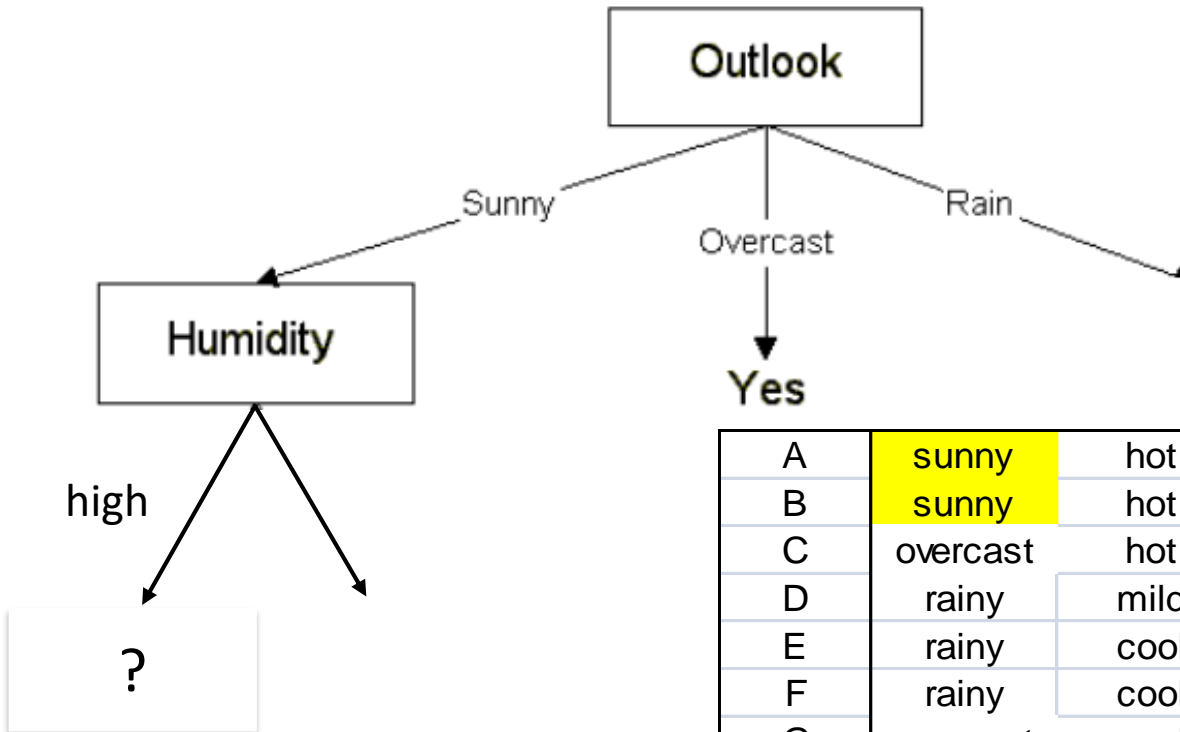
Exercise

- Determine the structure of our decision tree **after the humidity node**.



A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

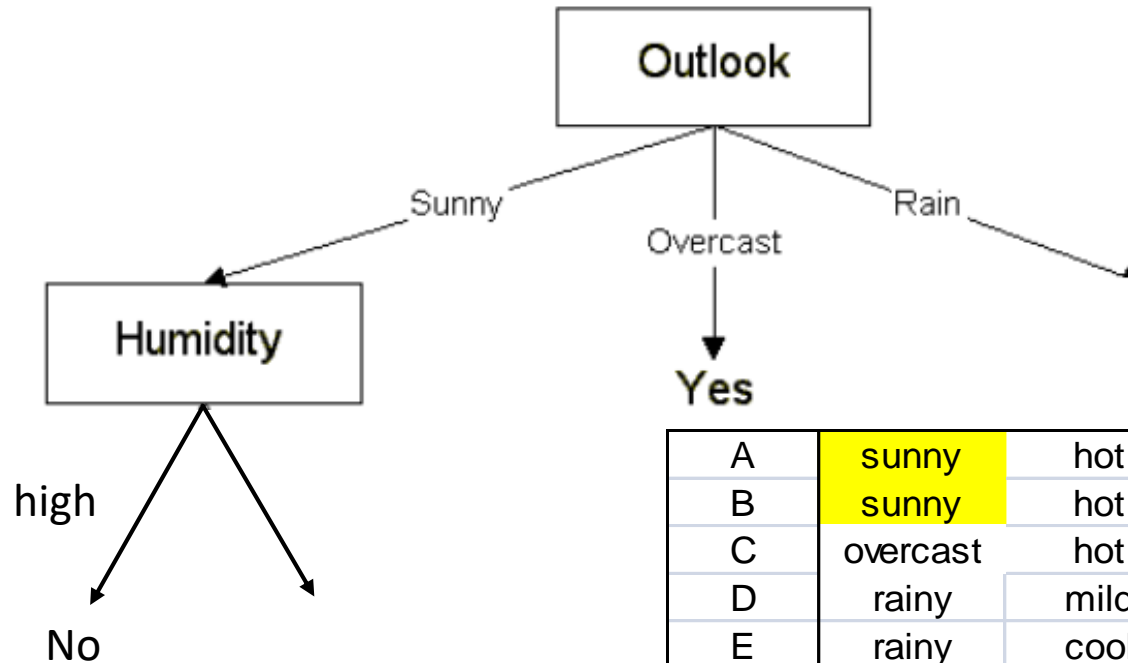
Exercise



- Determine the structure of our decision tree after the humidity node.

A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

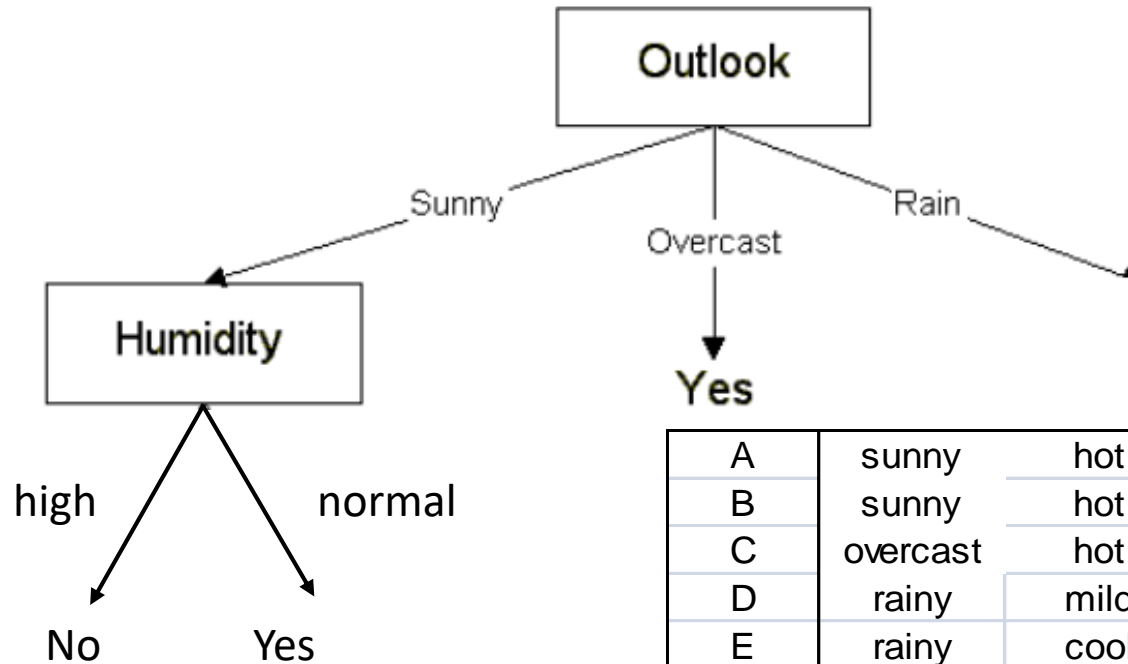
Exercise



- Determine the structure of our decision tree after the humidity node.

A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

Exercise



- Determine the structure of our decision tree after the humidity node.

A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes
D	rainy	mild	high	false	yes
E	rainy	cool	normal	false	yes
F	rainy	cool	normal	true	no
G	overcast	cool	normal	true	yes
H	sunny	mild	high	false	no
I	sunny	cool	normal	false	yes
J	rainy	mild	normal	false	yes
K	sunny	mild	normal	true	yes
L	overcast	mild	high	true	yes
M	overcast	hot	normal	false	yes
N	rainy	mild	high	true	no

ID3 Algorithm

ID3(Examples, features, Target):

Input:	Examples:	set of classified examples
	features:	set of features in the examples
	Target:	classification to be predicted

if all Examples have same class **then return** this class

else if all features are tested **then return** majority class

else:

let Best = choosefeature()

 #selects feature that **best separates** Examples relative to Target

let Tree = new decision tree with Best as root node

foreach value v_i **of** Best

let Examples _{i} = subset of Examples that have Best= v_i

let Subtree = **ID3(Examples _{i} , features-Best, Target)**

add branch from Tree to Subtree with label v_i

return Tree

features: Outlook, Temp, Humidity, Windy

Examples: Entire Dataset

Target: Play = Yes, Pay = No

Best = Outlook

For Loop * 3

(i) v1 = Sunny: Examples1 = Subset s.t. outlook = Sunny

(ii) v2 = Overcast: Examples2 = Subset s.t. outlook = Overcast

(iii) v3 = Rainy: Examples3 = Subset s.t. outlook = Rainy

features: Temp, Humidity, Windy

Examples1: outlook = Sunny

Target: Play = Yes, Pay = No

Best = Humidity

For Loop * 2

(i) v1 = High: Examples1 = Subset s.t. Humidity = High

(ii) v2 = Normal: Examples2 = Subset s.t. outlook = Normal

Different Metrics for Splitting Data

- We have examined the ID3 algorithm, which uses the information gain and entropy formulas as a means of splitting the data.
 - For a dataset S with n different classes(outputs), proportion of class i is denoted p_i
- Misclassification Error: $CE(t) = 1 - \max_i [p_i]$

Misclassification Error focuses on the class that is best represented in the dataset. For example, if we had a dataset where we had 2 * Play = yes and 8 * Play = No then it would make the assumption that the best represented class, Play = No, was the correct one and assumes that $1 - p(\text{play} = \text{No})$ represents the error rate. CE in this case would be $(1 - 8/10)$
As with the entropy we attempt to minimise this value.

Different Metrics for Splitting Data

- We have examined the ID3 algorithm, which uses information gain formula as a means of splitting the data.
 - For a dataset S with n different classes(outputs), proportion of class i is denoted p_i
- Gini Index:
$$GI(t) = 1 - \sum_{i=1}^n [p_i]^2$$

The Gini index can be understood as calculating **how often the target in a dataset would be misclassified if predictions were made based on the distribution of the target in the dataset**. For example, if there were two classes that occurred in the dataset with equal frequency then the expected rate of misclassification would be 0.5. Again the objective here is to minimise the Gini Index. If all instance in a dataset belong to one class the Gini Index would be 0.

Different Metrics for Splitting Data

- The following is an example of using each metric below

Node N_1	Count
Class=0	0
Class=1	6

$$\text{Gini} = 1 - (0/6)^2 - (6/6)^2 = 0$$

$$\text{Entropy} = -(0/6) \log_2(0/6) - (6/6) \log_2(6/6) = 0$$

$$\text{Error} = 1 - \max[0/6, 6/6] = 0$$

Node N_2	Count
Class=0	1
Class=1	5

$$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$$

$$\text{Entropy} = -(1/6) \log_2(1/6) - (5/6) \log_2(5/6) = 0.650$$

$$\text{Error} = 1 - \max[1/6, 5/6] = 0.167$$

Node N_3	Count
Class=0	3
Class=1	3

$$\text{Gini} = 1 - (3/6)^2 - (3/6)^2 = 0.5$$

$$\text{Entropy} = -(3/6) \log_2(3/6) - (3/6) \log_2(3/6) = 1$$

$$\text{Error} = 1 - \max[3/6, 3/6] = 0.5$$

Information Gain

- Note: The Gini Index and the Misclassification Error metrics can be plugged into the Information Gain function in the same way as the entropy metric.

$$\text{Gain}(S, A) = M(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} M(S_v)$$

Subset of S
where A has
value v

Where the function M can be either the **Entropy** function, the **Gini** Index or the **Misclassification Error**