COMP47750 Tutorial

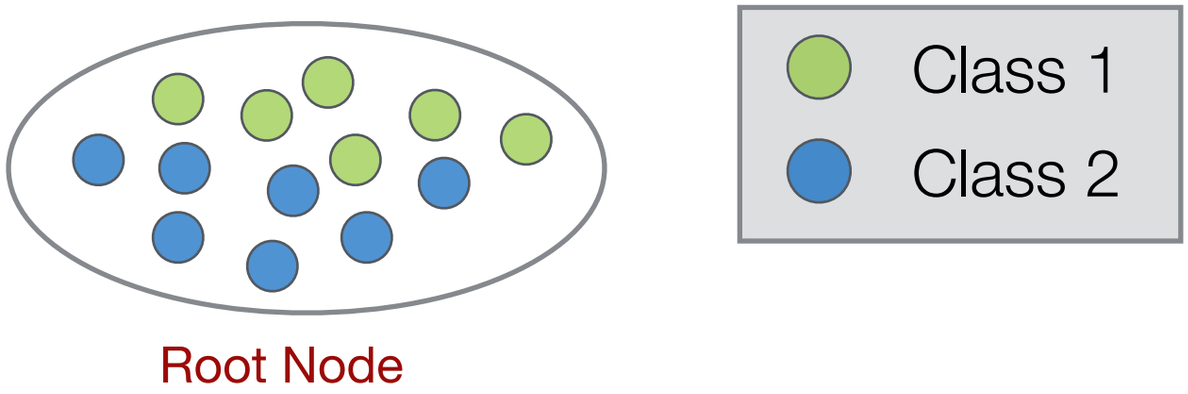
Decision Trees

Pádraig Cunningham

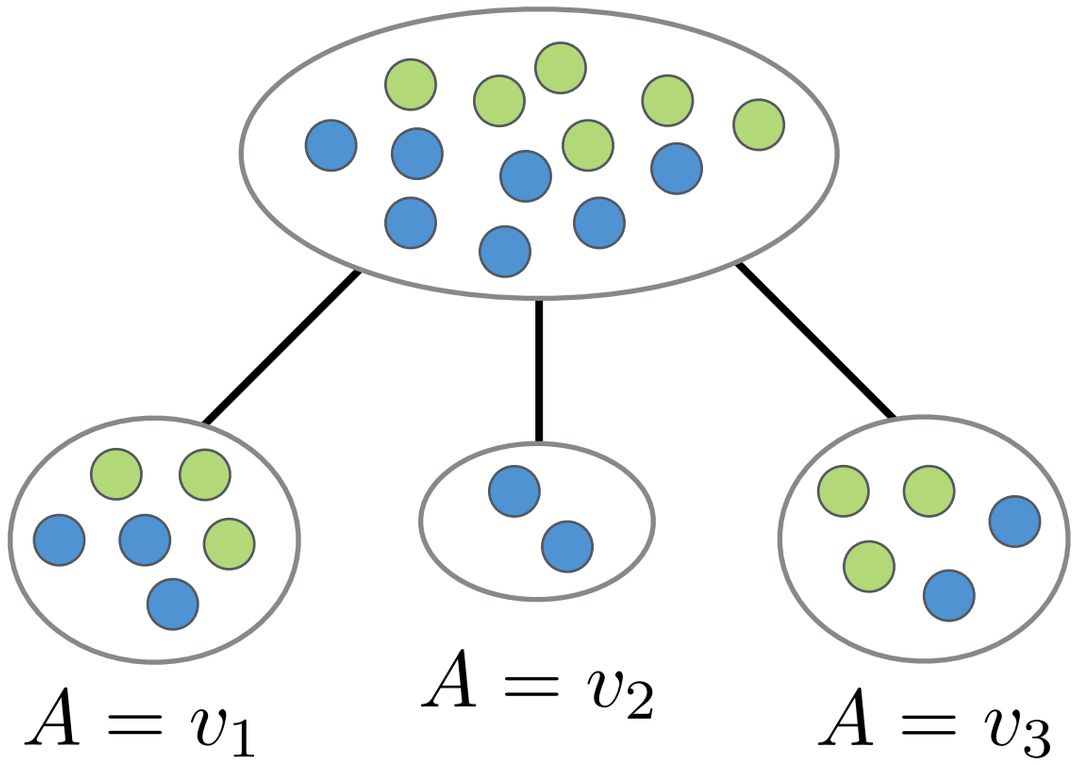
School of Computer Science

Reminder: Decision Tree Learning

1. Initially all examples in the training set are placed at the root node of the tree.

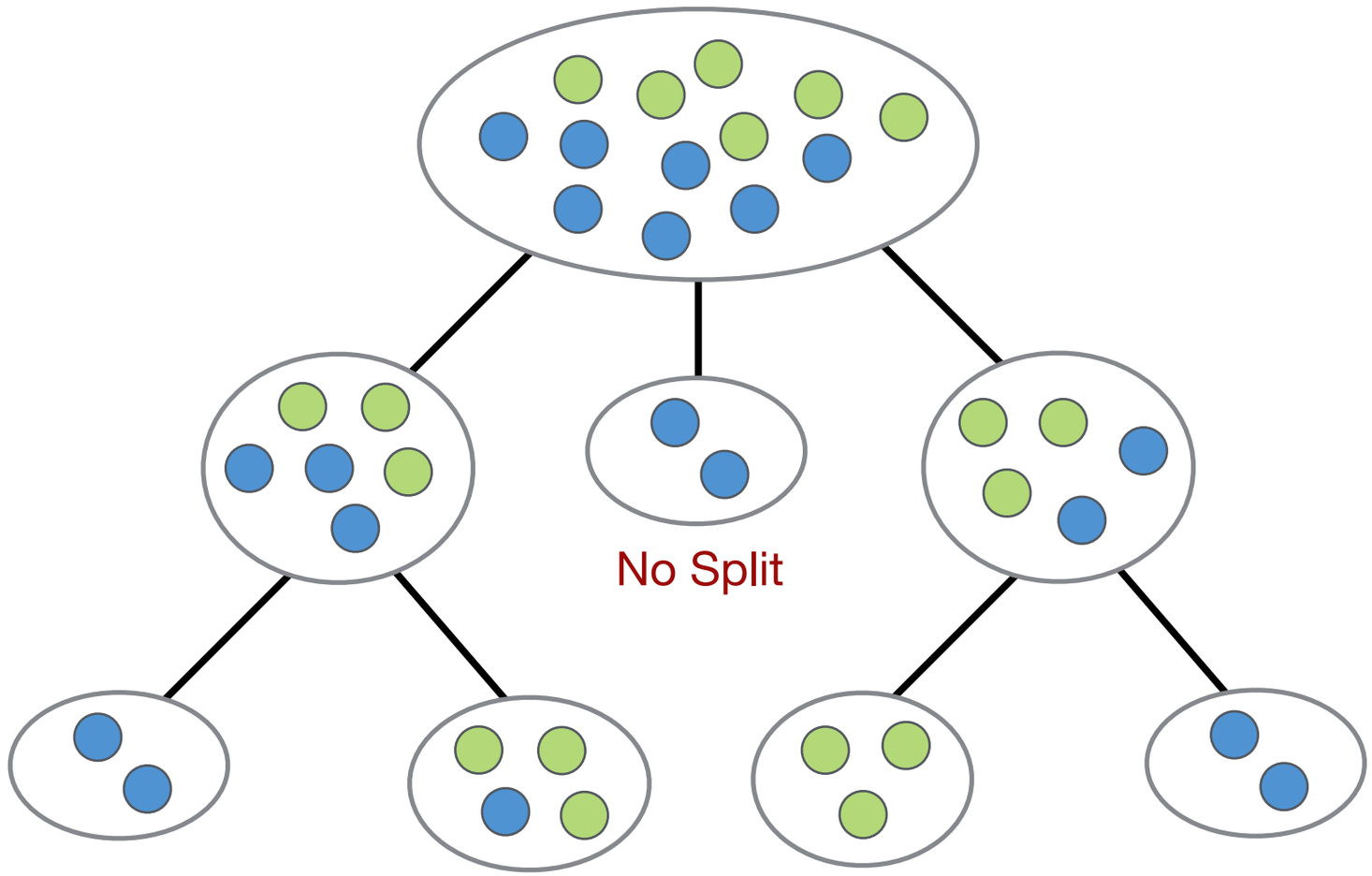


2. One of the available features (*A*) is now used to split the examples at the root node into two or more child nodes containing subsets of examples.



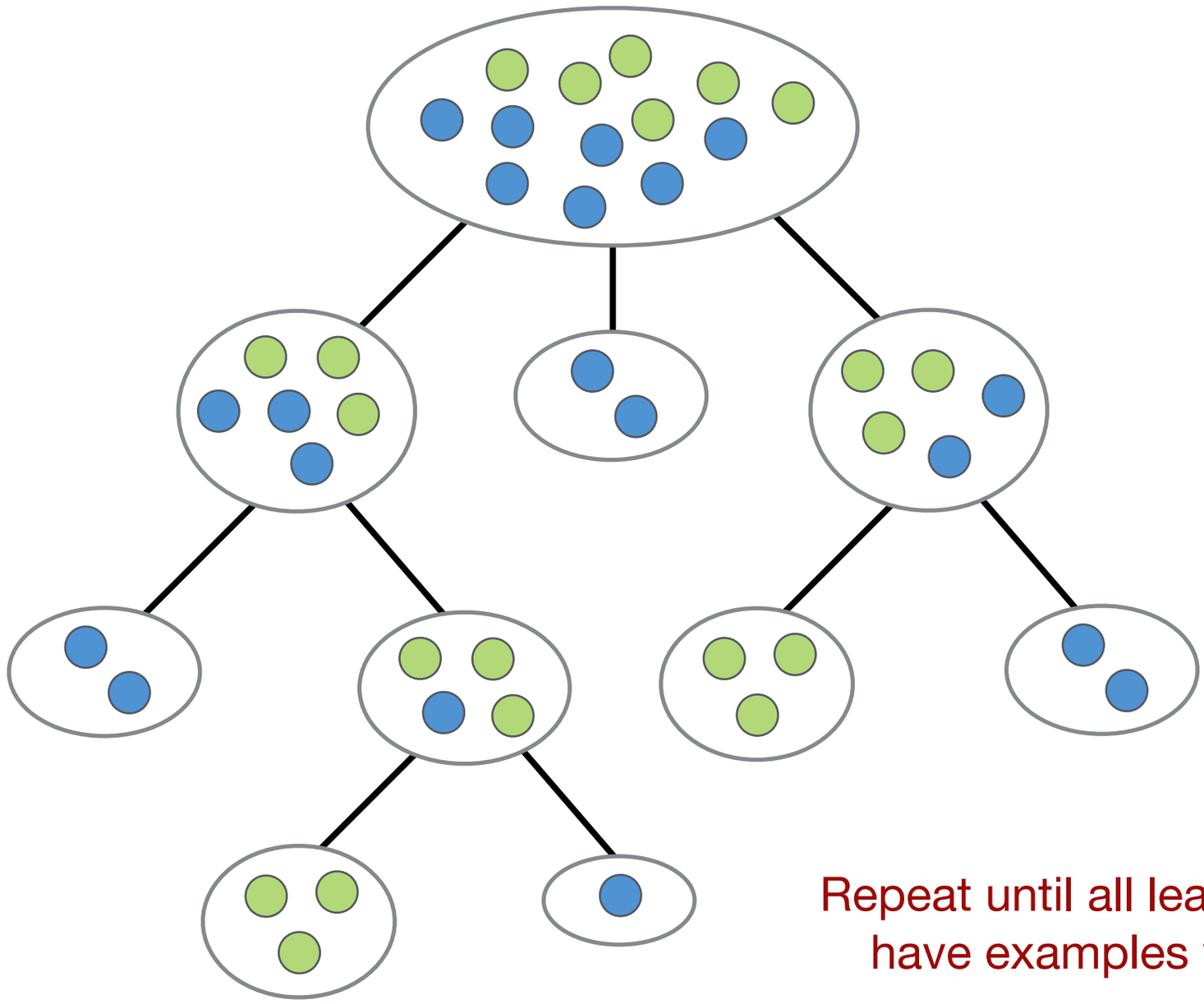
Reminder: Decision Tree Learning

3. The same process is now applied to each child node, except for any child node at which all examples have the same class.



Reminder: Decision Tree Learning

4. This continues until the training set has been divided into subsets in which all the examples have the same class.



Repeat until all leaf nodes in the tree have examples with same class

Reminder: Entropy & Information Gain

- Entropy provides a measure of impurity - how uncertain we are about the decision for a given set of examples.

Entropy of a set of examples S with class labels $\{C_1, \ldots, C_n\}$:

$$H(S) = \sum_{j=1}^{n} - p_j log_2(p_j)$$

where p_i is the relative frequency (probability) of class C_i.

- Information Gain measures the reduction in entropy when a feature is used to split a set into two or more subsets.

IG for feature A that splits a set of examples S into $\{S_1, \ldots, S_m\}$:

$$IG(S, A) = \text{(original entropy) - (entropy after split)}$$

$$IG(S, A) = H(S) - \sum_{i=1}^{m} \frac{|S_i|}{|S|} H(S_i)$$

Each subset is weighted in proportion to its size

Tutorial Q1(a)

a) What is the entropy of this dataset with respect to the target class label *Result*?

	Name	Hair	Height	Build	Lotion	Result
1	Sarah	blonde	average	light	no	sunburned
2	Dana	blonde	tall	average	yes	none
3	Alex	brown	short	average	yes	none
4	Annie	blonde	short	average	no	sunburned
5	Emily	red	average	heavy	no	sunburned
6	Pete	brown	tall	heavy	no	none
7	John	brown	average	heavy	no	none
8	Katie	brown	short	light	yes	none

2 classes: p1=(3/8) p2=(5/8)

Entropy(Dataset)

= -(3/8) x $\log_2$(3/8) -(5/8) x $\log_2$(5/8)

= 0.5306 + 0.4238 = 0.9544

$$H(S) = \sum_{j=1}^{n} - p_j log_2(p_j)$$

Assume $\log_2(0)=0$

Also $\log_2(x)=\log_{10}(x)/\log_{10}(2)$

Tutorial Q1(b)

b) Construct the decision tree that would be built with Information Gain for this data set. Show your work for selection of the root feature in your tree.

	Name	Hair	Height	Build	Lotion	Result
1	Sarah	blonde	average	light	no	sunburned
2	Dana	blonde	tall	average	yes	none
3	Alex	brown	short	average	yes	none
4	Annie	blonde	short	average	no	sunburned
5	Emily	red	average	heavy	no	sunburned
6	Pete	brown	tall	heavy	no	none
7	John	brown	average	heavy	no	none
8	Katie	brown	short	light	yes	none

Steps:

1. Calculate overall dataset entropy (Done).
2. Calculate entropy for each feature.
3. Calculate Information Gain for each feature.

Tutorial Q1(b)

Calculate entropy values for all features by looking at number of times each possible value occurs at root node:

Entropy(Hair=blonde) =
Entropy(Hair=brown) =
Entropy(Hair=red) =

Hair	Result
blonde	sunburned
blonde	none
blonde	sunburned
brown	none
brown	none
brown	none
brown	none
red	sunburned

Tutorial Q1(b)

Calculate entropy values for all features by looking at number of times each possible value occurs at root node:

```
Entropy(Hair=blonde) = Entropy(1/3,2/3) = 0.9183
Entropy(Hair=brown) = Entropy(0/4,4/4) = 0
Entropy(Hair=red) = Entropy(1/1,0/1) = 0

Entropy(Height=average) = Entropy(1/3,2/3) = 0.9183
Entropy(Height=tall) = Entropy(2/2,0/2) = 0
Entropy(Height=short) = Entropy(1/3,2/3) = 0.9183
```

Hair	Result
blonde	sunburned
blonde	none
blonde	sunburned
brown	none
brown	none
brown	none
brown	none
red	sunburned

Similarly...

```
Entropy(Build=light) = Entropy(1/2,1/2) = 1
Entropy(Build=average) = Entropy(1/3,2/3) = 0.9183
Entropy(Build=heavy) = Entropy(1/3,2/3) = 0.9183

Entropy(Lotion=no) = Entropy(2/5,3/5) = 0.9710
Entropy(Lotion=yes) = Entropy(3/3,0/3) = 0
```

Height	Result
average	sunburned
average	sunburned
average	none
short	none
short	sunburned
short	none
tall	none
tall	none

Tutorial Q1(b)

Use Information Gain to choose best feature to split for root node. Try each feature in turn…

```
IG(Hair) = Entropy(Dataset)
 - p(Hair=blonde) x Entropy(Hair=blonde)
 - p(Hair=brown) x Entropy(Hair=brown)
 - p(Hair=red) x Entropy(Hair=red)

= 0.9544 - (3/8)x0.9183 - (4/8)x0 - (1/8)x0
= 0.610
```

```
Entropy(Dataset) = 0.9544

Entropy(Hair=blonde) = 0.9183
Entropy(Hair=brown) = 0
Entropy(Hair=red) = 0

Entropy(Height=average) = 0.9183
Entropy(Height=tall) = 0
Entropy(Height=short) = 0.9183

Entropy(Build=light) = 1
Entropy(Build=average) = 0.9183
Entropy(Build=heavy) = 0.9183

Entropy(Lotion=no) = 0.9710
Entropy(Lotion=yes) = 0
```

```
IG(Height) = 0.9544 - (3/8)x0.9183-(2/8)x0-(3/8)x0.9183 = 0.2657

IG(Build) = 0.9544 - (2/8)x1-(3/8)x0.9183-(3/8)x0.9183 = 0.0157

IG(Lotion) = 0.9544 - (5/8)x0.9710-(3/8)x0 = 0.3475
```

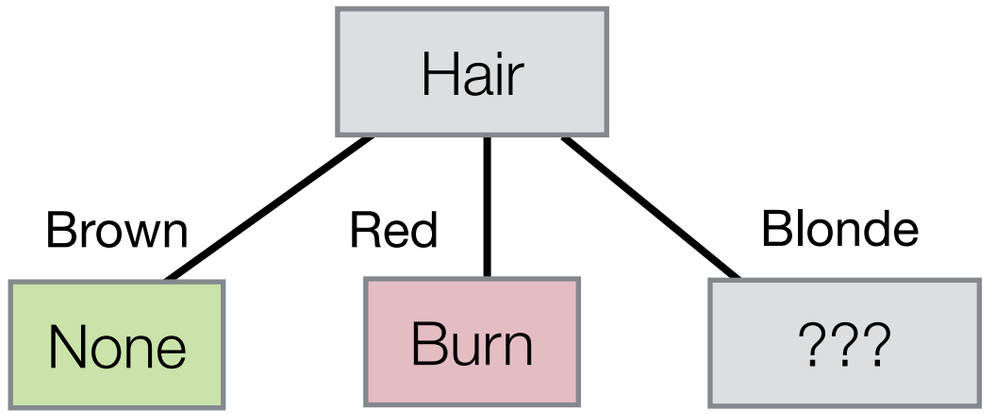
➡ “Hair” will be selected as the feature with the highest IG value. It perfectly classifies the data for *Hair=brown* & *Hair=red*

Tutorial Q1(b)

- "Hair" selected as the feature with the highest IG value ⇒ used to split the root node of the tree.

Child node *Hair=blonde:*

	Name	Hair	Height	Build	Lotion	Result
1	Sarah	blonde	average	light	no	sunburned
2	Dana	blonde	tall	average	yes	none
4	Annie	blonde	short	average	no	sunburned

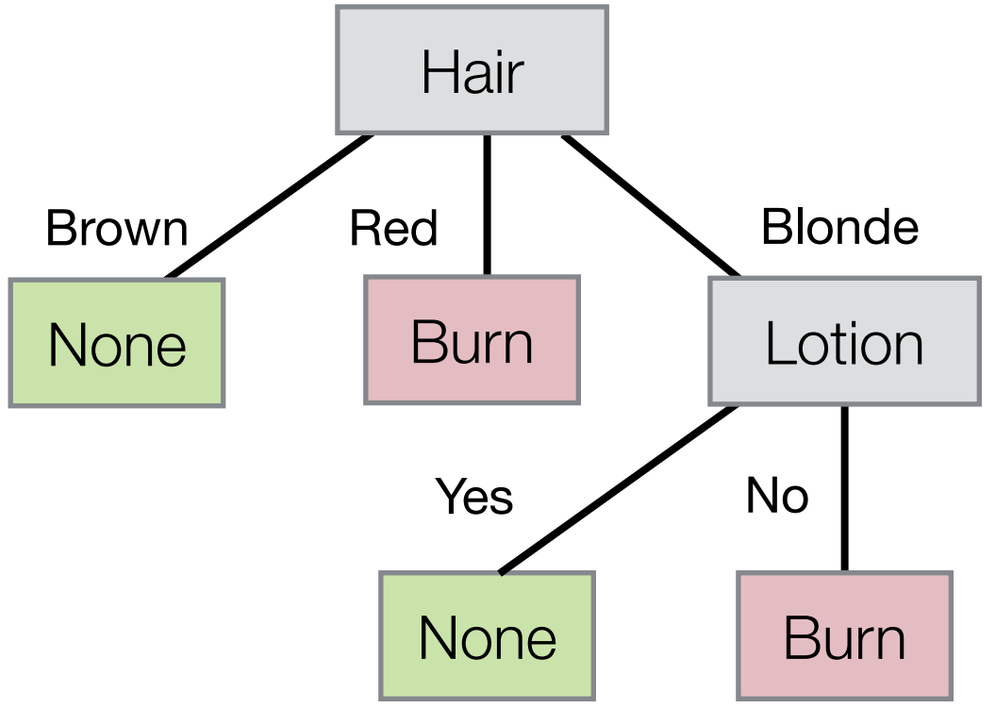


Tutorial Q1(b)

- "Hair" selected as the feature with the highest IG value ⇒ used to split the root node of the tree.

Child node *Hair=blonde:*

	Name	Hair	Height	Build	Lotion	Result
1	Sarah	blonde	average	light	no	sunburned
2	Dana	blonde	tall	average	yes	none
4	Annie	blonde	short	average	no	sunburned



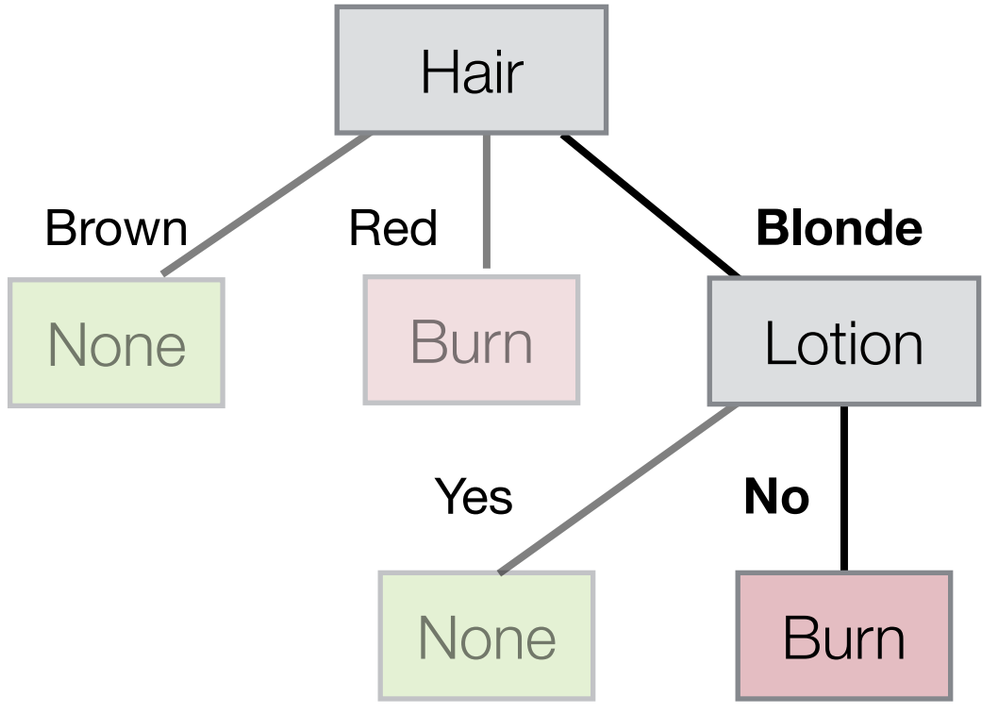
➡ The case for *Hair=blonde* contains (2 sunburned, 1 none).
Can split these into pure child nodes using feature "Lotion".

Tutorial Q1(c)

c) Using your decision tree from (b), how would you classify the following example?

	Hair	Height	Build	Lotion	Result
X	blonde	average	heavy	no	???

- First, check *Hair=Blonde*
- Next, check *Lotion=No*
- **Output: Sunburned**



Tutorial Q2(a)

a) What is the entropy of this dataset with respect to the target class label Risk based on the 14 examples above?

Example	Credit_History	Debt	Income	Risk
1	bad	low	0to30	high
2	bad	high	30to60	high
3	bad	low	0to30	high
4	unknown	high	30to60	high
5	unknown	high	0to30	high
6	good	high	0to30	high
7	bad	low	over60	medium
8	unknown	low	30to60	medium
9	good	high	30to60	medium
10	unknown	low	over60	low
11	unknown	low	over60	low
12	good	low	over60	low
13	good	high	over60	low
14	good	high	over60	low

$$H(S) = \sum_{j=1}^{n} - p_j log_2(p_j)$$

NB: Define $\log_2(0)=0$

p1=(6/14)

p2=(3/14)

p3=(5/14)

-(6/14) x $\log_2$(6/14) - (3/14) x $\log_2$(3/14) -(5/14) x $\log_2$(5/14)
= 0.5239 + 0.4762 + 0.5305
= 1.5306

Tutorial Q2(b)

b) Compute the entropy of each of the 3 descriptive features.

	CH	Debt	Income	Risk
1	bad	low	0to30	high
2	bad	high	30to60	high
3	bad	low	0to30	high
4	unknown	high	30to60	high
5	unknown	high	0to30	high
6	good	high	0to30	high
7	bad	low	over60	medium
8	unknown	low	30to60	medium
9	good	high	30to60	medium
10	unknown	low	over60	low
11	unknown	low	over60	low
12	good	low	over60	low
13	good	high	over60	low
14	good	high	over60	low

$$\text{Entropy}(CH=bad) = -(1/4) \times \log_2(1/4) - (3/4) \times \log_2(3/4) = 0.8113$$

$$\text{Entropy}(CH=unknown) = -(2/5)\log_2(2/5) - (1/5)\log_2(1/5) - (2/5)\log_2(2/5) = 1.5219$$

$$\text{Entropy}(CH=good) = -(1/5)\log_2(1/5) - (1/5)\log_2(1/5) - (3/5)\log_2(3/5) = 1.3710$$

$$\text{Entropy}(Debt=low) = -(2/7)\log_2(2/7) - (2/7)\log_2(2/7) - (3/7)\log_2(3/7) = 1.5567$$

$$\text{Entropy}(Debt=high) = -(4/7)\log_2(4/7) - (1/7)\log_2(1/7) - (2/7)\log_2(2/7) = 1.3788$$

$$\text{Entropy}(Income=0to30) = -(4/4)\log_2(4/4) = 0$$

$$\text{Entropy}(Income=30to60) = -(2/4)\log_2(2/4) - (2/4)\log_2(2/4) = 1$$

$$\text{Entropy}(Income=over60) = -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.65$$

Tutorial Q2(c)

c) Which one of the predicting features would be selected by ID3 at the root of a decision tree? Explain your answer.

Use Information Gain to choose best feature to split for root node…

```
IG(CH) = Entropy(Dataset)
  - p(CH=bad) x Entropy(CH=bad)
  - p(CH=unknown) x Entropy(CH=unknown)
  - p(CH=good) x Entropy(CH=good)

= 1.5306 - (4/14) x 0.8113 - (5/14) x 1.5219
   - (5/14) x 1.3710
= 0.2656
```

```
Entropy(CH=bad) = 0.8113
Entropy(CH=unknown) = 1.5219
Entropy(CH=good) = 1.3710

Entropy(Debt=low) = 1.5567
Entropy(Debt=high) = 1.3788

Entropy(Income=0to30) = 0
Entropy(Income=30to60) = 1
Entropy(Income=over60) = 0.65
```

```
IG(Debt) = 1.5306 - (7/14) x 1.5567 - (7/14) x 1.3788 = 0.0628

IG(Income) = 1.5306 - (4/14) x 0 - (4/14) x 1 - (6/14) x 0.65 = 0.966
```

➡ “Income” will be selected as the feature to split as it has the highest IG value.

Tutorial Q2(d)

d) What is the main problem with the Information Gain criterion for feature selection in decision trees?

"Although information gain is usually a good measure for deciding the relevance of an attribute, it is not perfect. A notable problem occurs when information gain is applied to attributes that can take on a large number of distinct values. For example, suppose that one is building a decision tree for some data describing the customers of a business. Information gain is often used to decide which of the attributes are the most relevant, so they can be tested near the root of the tree. One of the input attributes might be the customer's credit card number. This attribute has a high mutual information, because it uniquely identifies each customer, but we do not want to include it in the decision tree: deciding how to treat a customer based on their credit card number is unlikely to generalise to customers we haven't seen before (overfitting)."

http://en.wikipedia.org/wiki/Information_gain_in_decision_trees

Tutorial Q3

- For the datasets analysed in the **`04 DTrees`** notebook, will the resulting trees be different if the feature selection criterion is '**`gini`**' instead of '**`entropy`**'.

```
tree = DecisionTreeClassifier(criterion='gini')
ap_tree = tree.fit(X, y)
```

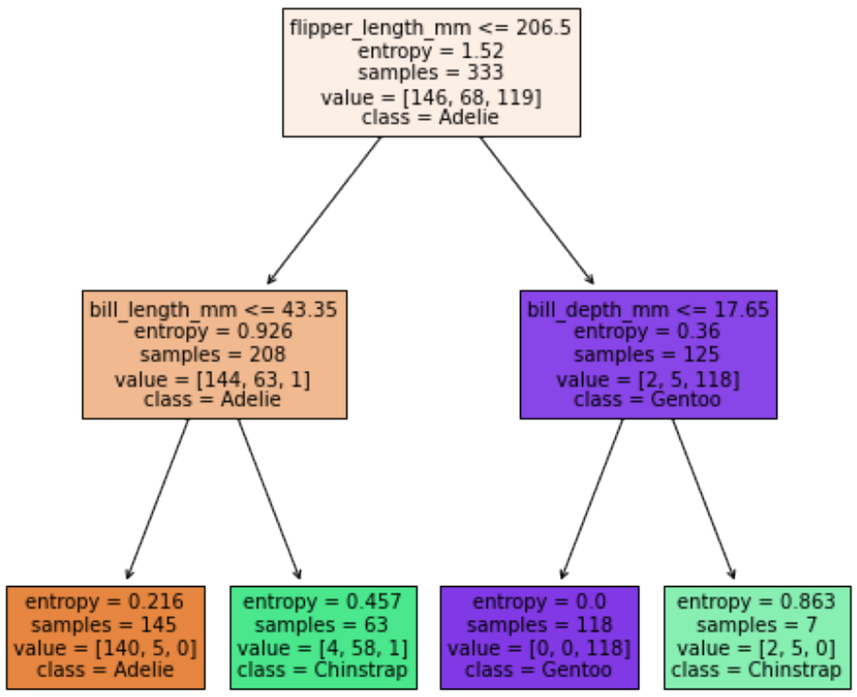
Tutorial Q4

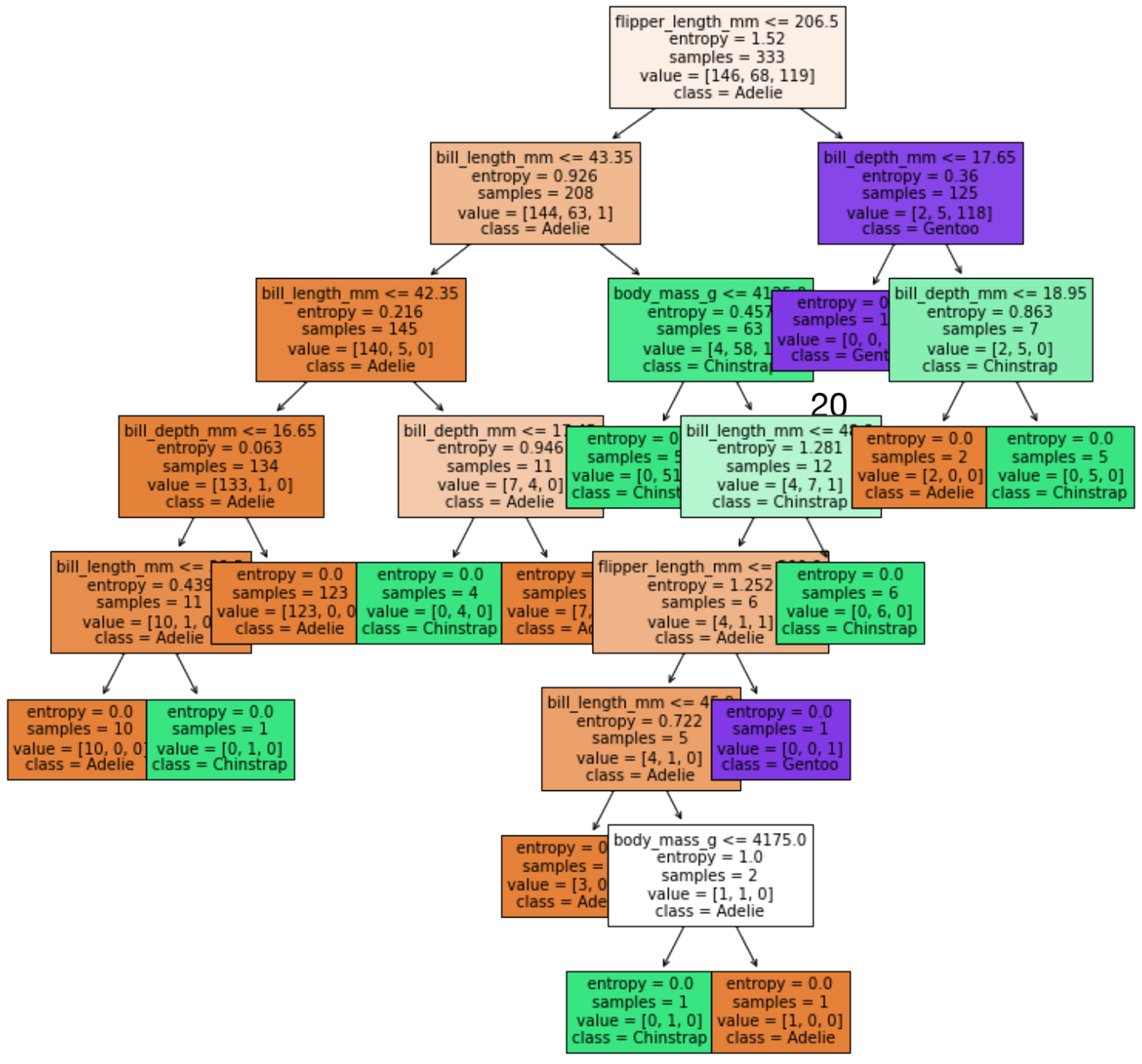
If a decision tree is allowed to be too *bushy* it is likely to overfit the training data. Consequently decision trees are often pruned to prevent overfitting.
In the example in the '**`05 DTrees Tutorial`**' notebook we use the **`min_impurity_decrease`** attribute to control the size of the tree.

1. What does the Penguins Data tree look like when no pruning is enforced?
2. What other options does sklearn provide to manage the bushiness of the tree? https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
3. Use two other pruning strategies to produce similar trees.

Tutorial Q4

1. What does the Penguins Data tree look like when no pruning is enforced?





20

```
ptree.get_n_leaves()
Out[17]:
4
```

```
ptree = DecisionTreeClassifier(criterion='entropy')
```

Tutorial Q4

2. What other options does sklearn provide to manage the bushiness of the tree? https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

max_depth *int, default=None*
The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split *int or float, default=2*
The minimum number of samples required to split an internal node:

min_samples_leaf *int or float, default=1*
The minimum number of samples required to be at a leaf node.

max_leaf_nodes *int, default=None*
Grow a tree with `max_leaf_nodes` in best-first fashion. Best nodes are defined as relative reduction in impurity.

min_impurity_decrease *float, default=0.0*
A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

Deprecated

min_impurity_split *float, default=0*
Threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise it is a leaf.