

CIT

COMP9033 DATA ANALYTICS

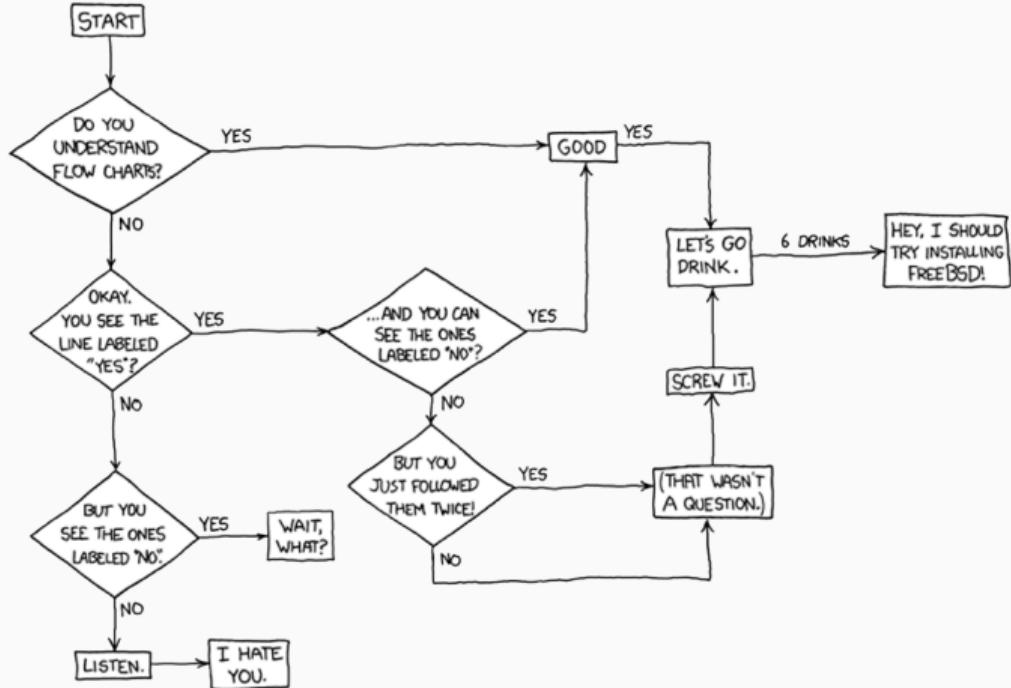
7/12

DECISION TREES

DR. DONAGH HORGAN

DEPARTMENT OF COMPUTER SCIENCE
CORK INSTITUTE OF TECHNOLOGY

2018.03.13



Overview

1. Linear regression:

- What it is.
- How it works.
- Measuring model error.

2. The least squares technique:

- The residual sum of squares.
- The least squares solution.
- Performance considerations.

3. Shrinkage methods:

- Problems with least squares.
- Ridge regression.
- Hyperparameters.

4. Subset selection:

- Forward stepwise selection.
- Backward stepwise selection.
- Hybrid methods.

1. Measuring classifier error:

- Confusion matrices.
- Accuracy.
- Precision.
- Recall.

2. Decision trees:

- Tree structures.
- Classification trees.
- Regression trees.

3. The CART algorithm:

- How to build a decision tree.
- Impurity measures.
- Stopping criteria.

4. Advanced techniques:

- Pruning.
- Random forests.

Measuring classifier error

1.1 / CONFUSION MATRICES

- Classifiers produce categoric predictions, not numeric predictions.
- This means that we can't use quantitative measures (e.g. mean absolute error, root mean square error) to measure how accurate a classifier is.
- Instead, we must use alternative measurements of accuracy.

predicted →

		Cat	Dog	Pig
true	Cat	3	0	1
	Dog	0	6	1
	Pig	2	2	5

1.2 / CONFUSION MATRICES

- One commonly used technique is the confusion matrix, shown opposite.
- In a confusion matrix, the rows represent the true classes of the data, while the columns represent the predicted classes.
- By examining a particular entry, we can check how many correct classifications were made, e.g.
 - Three cats were correctly identified.
 - One dog was mistaken for a pig.
 - Two pigs were mistaken for cats.

predicted →

		Cat	Dog	Pig
true	Cat	3	0	1
	Dog	0	6	1
	Pig	2	2	5

1.3 / CONFUSION MATRICES

- We can condense the data in the confusion matrix to get a better picture about the accuracy of our classifier on a single category.
- For instance, if we focus on cats, then we can create the new confusion matrix on the bottom right.

	Cat	Dog	Pig
Cat	3	0	1
Dog	0	6	1
Pig	2	2	5



	Cat	Not a cat
Cat	3	1
Not a cat	2	14

1.4 / CONFUSION MATRICES

- The top left entry represents the number of *true positives* (TP), e.g. the number of correctly identified cats.
- The top right entry represents the number of *false negatives* (FN), e.g. the number of cats identified as other animals.
- The bottom left entry represents the number of *false positives* (FP), e.g. the number of other animals identified as cats.
- The bottom right entry represents the number of *true negatives* (TN), e.g. the number of other animals identified as other animals.

	Cat	Not a cat
Cat	3	1
Not a cat	2	14

	Cat	Not a cat
Cat	TP	FN
Not a cat	FP	TN

1.5 / CONFUSION MATRICES

- Ideally, we'd like the number of *false negatives* and the number of *false positives* to be zero, i.e. only the diagonal elements of the matrix would be non-zero.
- This is also true for uncondensed confusion matrices: for zero error, the only non-zero elements should lie on the diagonal running from the top left to the bottom right.

	Cat	Not a cat
Cat	3	1
Not a cat	2	14

	Cat	Not a cat
Cat	TP	FN
Not a cat	FP	TN

1.6 / ACCURACY

- The accuracy of a classifier is defined as the ratio of correct predictions to total predictions, *i.e.*

$$\begin{aligned}\text{Accuracy} &= \frac{\text{Correct predictions}}{\text{Total predictions}} \\ &= \frac{TP + TN}{TP + FN + FP + TN}. \quad (7.1)\end{aligned}$$

		Cat	Not a cat
	Cat	TP	FN
	Not a cat	FP	TN

- By definition, accuracy lies in the range $[0, 1]$.

1.7 / ACCURACY

- Accuracy is a useful measure of the overall performance of a classifier:
 - As the proportion of our correct predictions grows larger, accuracy grows larger.
 - As the proportion of our correct predictions grows smaller, accuracy grows smaller.
- Need to be careful when dealing with imbalanced classes, e.g. if 90% of our data samples are “cats”, then simply predicting “cat” every time will give 90% accuracy!
- Typically, precision and recall are used for extra insight.

	Cat	Not a cat
Cat	TP	FN
Not a cat	FP	TN

1.8 / MISCLASSIFICATION RATE

- The *misclassification rate* (MR) of a classifier is defined as the ratio of incorrect predictions to total predictions, i.e.

$$\begin{aligned} \text{MR} &= \frac{\text{Incorrect predictions}}{\text{Total predictions}} \\ &= \frac{FN + FP}{TP + FN + FP + TN}. \end{aligned} \quad (7.2)$$

- By definition, the misclassification rate lies in the range $[0, 1]$.
- Using Equation 7.1, it can be shown that $\text{MR} = 1 - \text{Accuracy}$.

	Cat	Not a cat
Cat	TP	FN
Not a cat	FP	TN

1.9 / PRECISION

- The *precision* of a classifier is the ratio of true positives to the total positives, *i.e.*

$$\begin{aligned}\text{Precision} &= \frac{\text{True positive predictions}}{\text{Total positive predictions}} \\ &= \frac{TP}{TP + FP}. \quad (7.3)\end{aligned}$$

- By definition, precision lies in the range $[0, 1]$.
- In our earlier example, precision would measure the proportion of animals labelled as cats that were actually cats.

		Cat	Not a cat
	Cat	TP	FN
	Not a cat	FP	TN

1.10 / PRECISION

- Precision is useful because it gives us finer detail than accuracy about how our classifier behaves on the *positive* class:
 - As the number of **true positive** predictions increases, precision grows larger.
 - As the number of **true positive** predictions decreases, precision grows smaller.
 - As the number of **false positive** predictions increases, precision grows smaller.
 - As the number of **false positive** predictions decreases, precision grows larger.
- We should always compute precision when dealing with imbalanced classes!

	Cat	Not a cat
Cat	TP	FN
Not a cat	FP	TN

1.11 / RECALL

- The *recall* of a classifier is the ratio of the number of true positive classifications to the total number of elements in the target class:

$$\begin{aligned}\text{Recall} &= \frac{\text{True positive predictions}}{\text{Target class}} \\ &= \frac{TP}{TP + FN}. \quad (7.4)\end{aligned}$$

		Cat	Not a cat
	Cat	TP	FN
	Not a cat	FP	TN

- By definition, recall lies in the range $[0, 1]$.
- In our earlier example, recall would measure the proportion of true cats that were labelled as cats.

1.12 / RECALL

- Recall is useful because it gives us finer detail than accuracy about how our classifier behaves on the *target class*:
 - As the number of **true positive** predictions increases, recall grows larger.
 - As the number of **true positive** predictions decreases, recall grows smaller.
 - As the number of **false negative** predictions increases, recall grows smaller.
 - As the number of **false negative** predictions decreases, recall grows larger.
- We should always compute recall when dealing with imbalanced classes!

	Cat	Not a cat
Cat	TP	FN
Not a cat	FP	TN

1.13 / EXAMPLE: MEASURING CLASSIFICATION ERROR

- Q. Compute accuracy, precision and recall using confusion matrix opposite.
- A. We can compute the accuracy using Equation 7.1 as

$$\begin{aligned}\text{Accuracy} &= \frac{TP + TN}{TP + FN + FP + TN} \\ &= \frac{3 + 14}{3 + 1 + 2 + 14} \\ &= 0.85.\end{aligned}$$

		Cat	Not a cat
	Cat	3	1
	Not a cat	2	14

1.14 / EXAMPLE: MEASURING CLASSIFICATION ERROR

- We can compute the precision using Equation 7.3 as

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{3}{3 + 2} = 0.6.$$

- Finally, we can compute the recall using Equation 7.4 as

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{3}{3 + 1} = 0.75.$$

	Cat	Not a cat
Cat	3	1
Not a cat	2	14

1.15 / EXAMPLE: MEASURING CLASSIFICATION ERROR

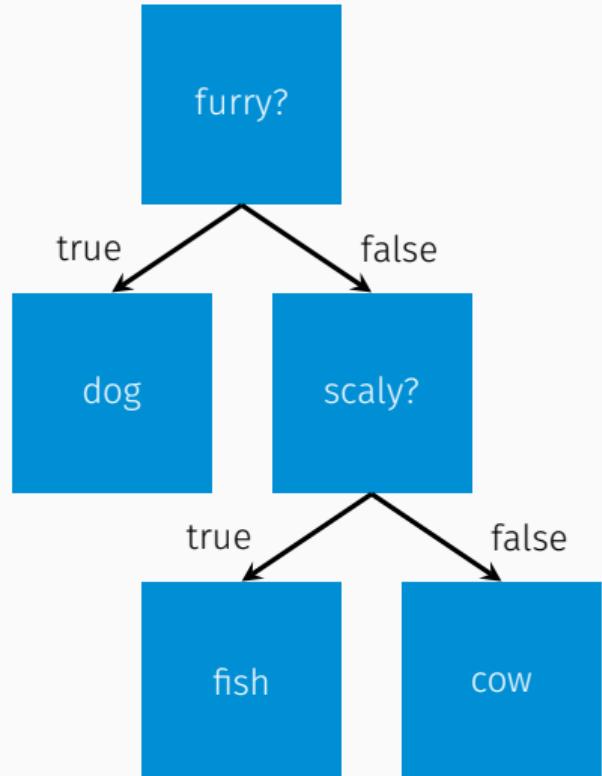
- The precision and recall are lower than the accuracy!
 - Our accuracy is high because we're relatively good at correctly labelling cats and non-cats.
 - Our precision and recall are lower because, relative to the total number of cats, the number of mistakes we make is higher.
- When dealing with imbalanced classes, precision and recall are sanity checks on our overall accuracy figure!

	Cat	Not a cat
Cat	3	1
Not a cat	2	14

Decision trees

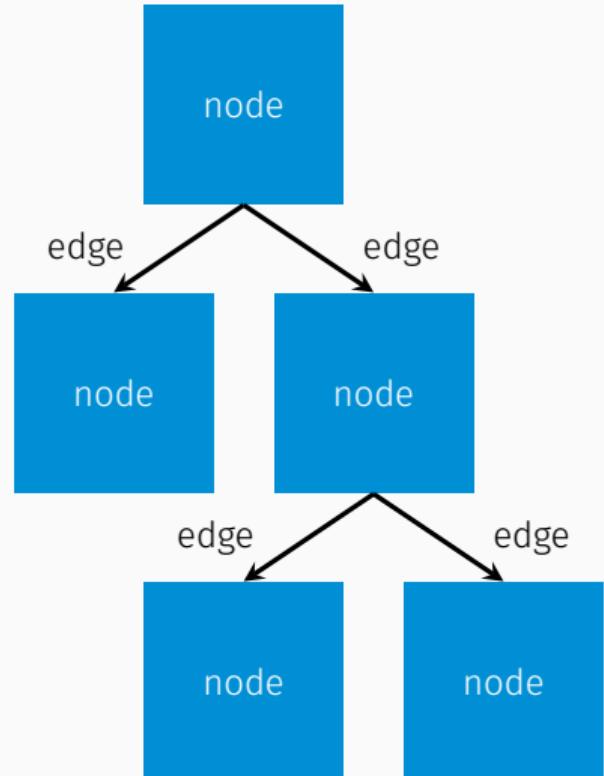
2.1 / DECISION TREES

- *Decision trees* are a type of hierarchical model consisting of nested if-else clauses.
- They are often used to describe processes or chains of decisions.
- In the data analytics domain, they are typically created using machine learning algorithms.
- The tree to the right illustrates a (very) simple animal classification model.



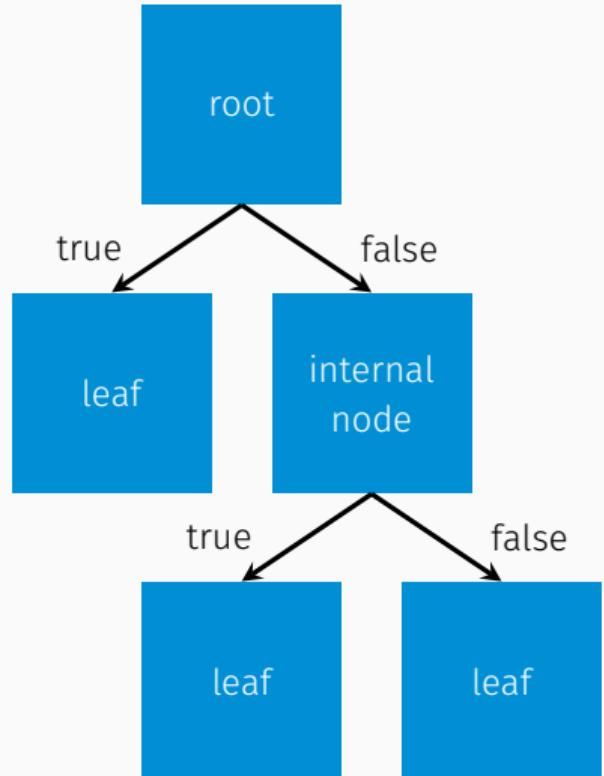
2.2 / DECISION TREES

- A decision tree is essentially a graph structure, consisting of *nodes* and *edges*:
 - Nodes represent states.
 - Edges represent connections between states.
- In binary decision trees, nodes either have two edges (representing a true/false condition) or *none*.



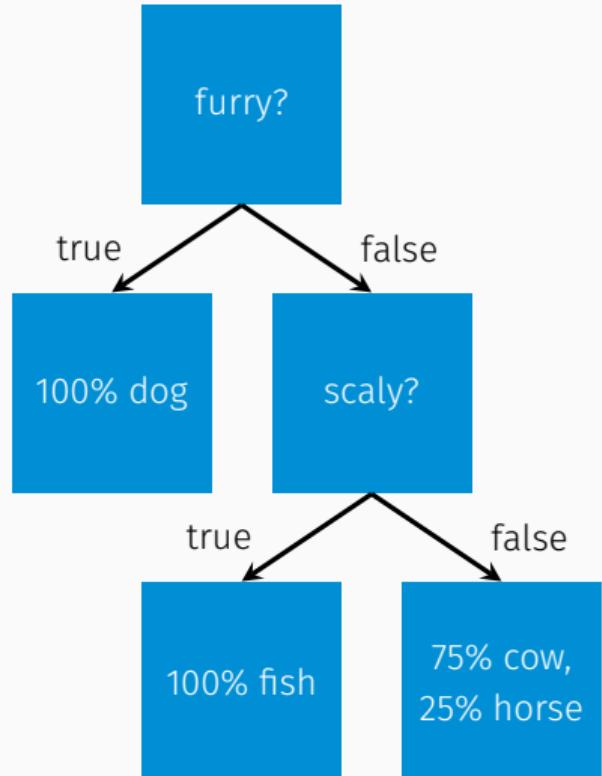
2.3 / DECISION TREES

- Depending on their position within the tree, nodes have different meanings:
 - The uppermost node in the tree is known as the *root node* and is the starting state of any prediction attempt.
 - The end nodes (*i.e.* those with no edges) are known as *leaf nodes* and represent a final outcome (*e.g.* spam or ham) or set of outcomes.
 - All other nodes are called *internal nodes* and represent some intermediate state between the root and one of the leaves.



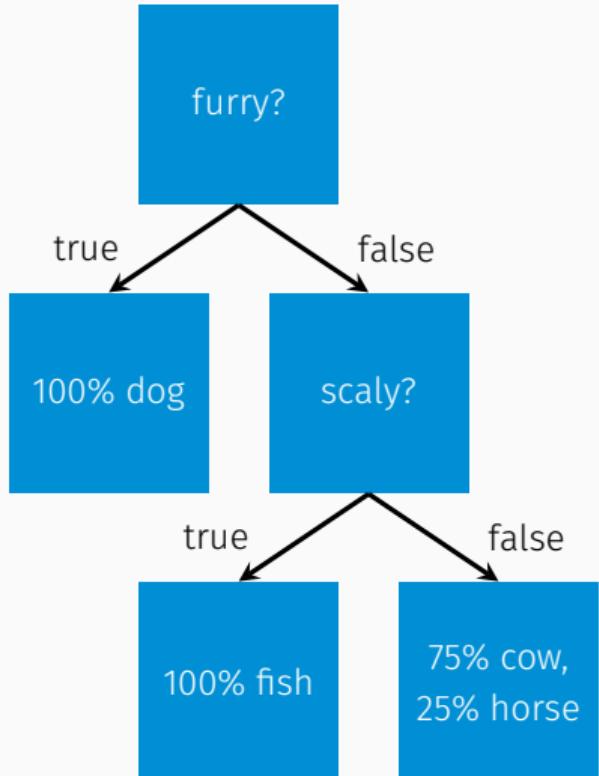
2.4 / DECISION TREE CLASSIFICATION

- Decision trees can be used to classify data samples based on the values of their features.
- The general process is as follows:
 - Start at the root node.
 - Check which edge matches our criteria (one always will) and then follow it to the next node.
 - Repeat Step 2 until we reach a leaf node.



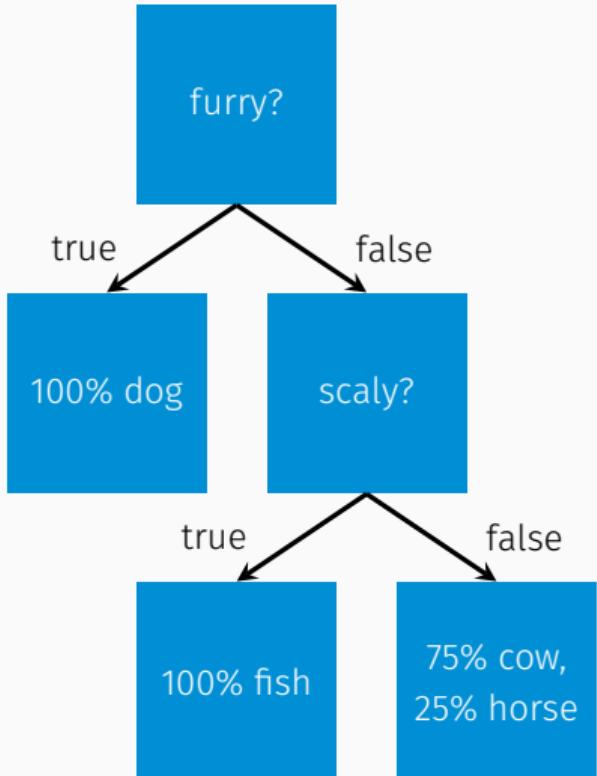
2.5 / DECISION TREE CLASSIFICATION

- In decision tree classification, leaf nodes represent a category or set of categories, depending on how the decisions in the tree have split the training data:
 - If a leaf node represents a single outcome, then that outcome is predicted.
 - If a leaf node represents more than one outcome, then we select the most likely outcome (*i.e.* the mode) as the prediction.



2.6 / DECISION TREE CLASSIFICATION

- For instance, in the tree to the right:
 - If an animal is furry, then we would predict that it is a dog.
 - If an animal is not furry, but is scaly, then we would predict that it is a fish.
 - If an animal is not furry and is not scaly, then we would predict that it is a cow.



2.7 / EXAMPLE: IRIS DATA SET



Iris setosa



Iris virginica

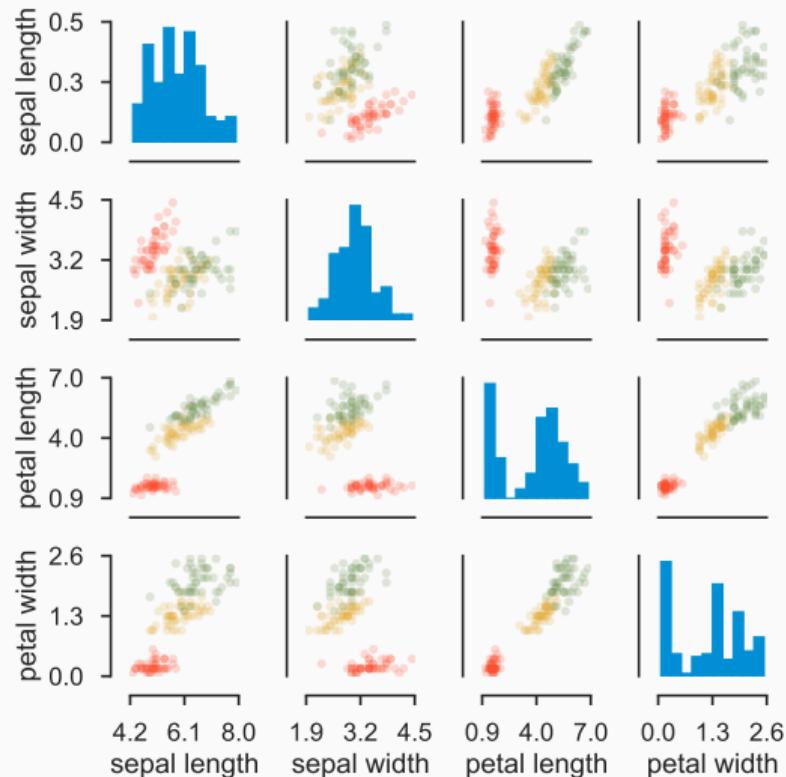


Iris versicolor

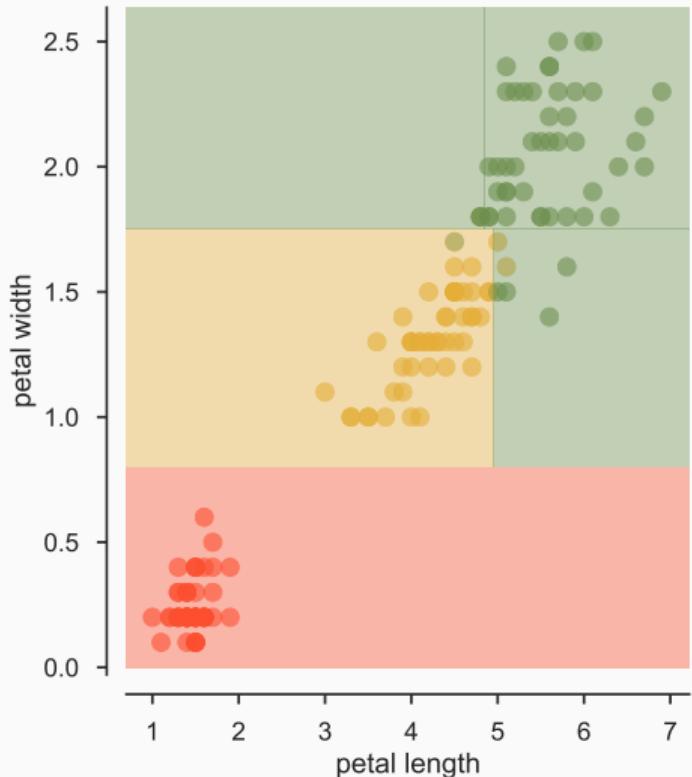
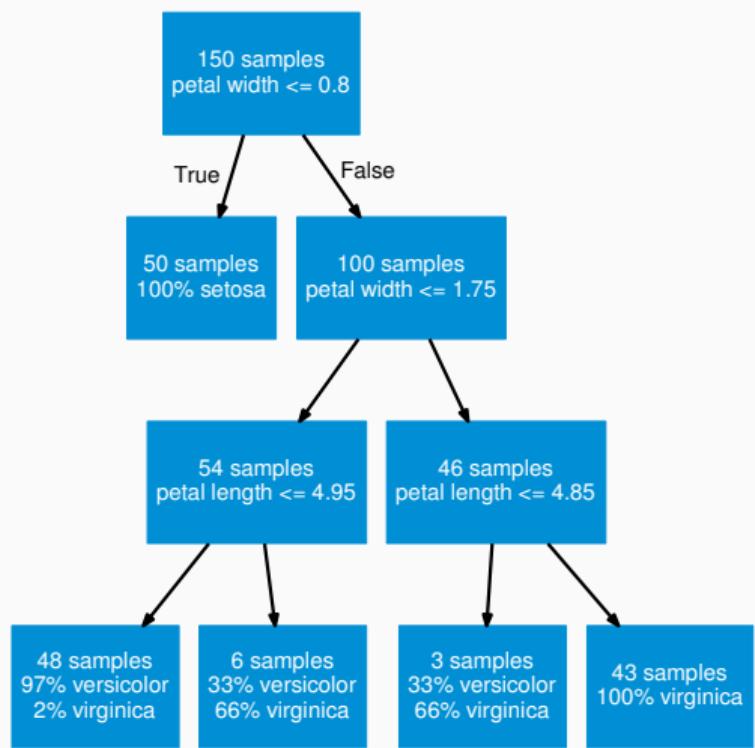
- Let's use decision trees to analyse the Iris data set:
 - The data set consists of 150 evenly distributed categoric samples, i.e. each category has the same number of samples.
 - There are three categories: *Iris setosa*, *Iris virginica*, *Iris versicolor*.
 - There are four explanatory variables: sepal length, sepal width, petal length and petal width.

2.8 / EXAMPLE: IRIS DATA SET

- *Iris setosa* (red) is easily distinguished as its characteristics are very different to the other two species.
- *Iris versicolor* (yellow) and *Iris virginica* (green) are more difficult to separate, as their properties are more similar.

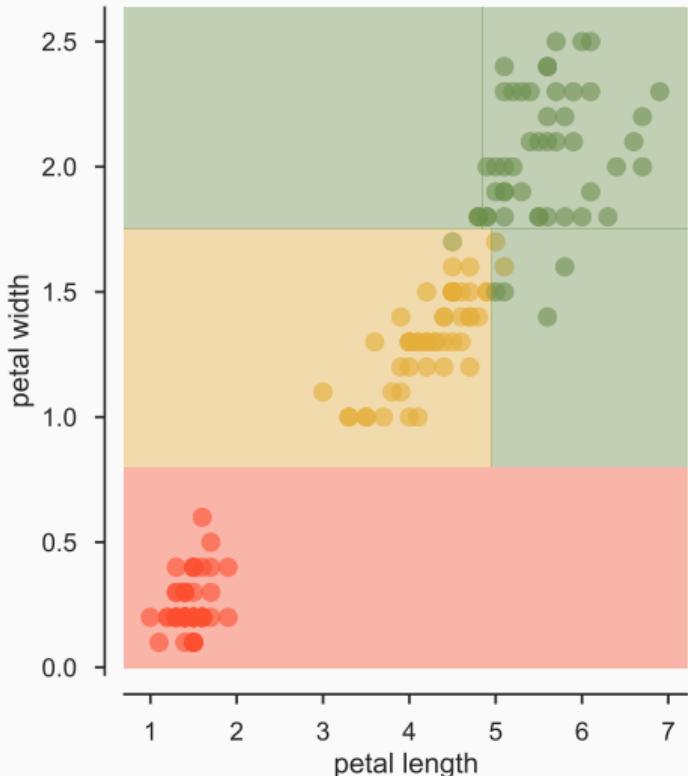


2.9 / EXAMPLE: IRIS DATA SET



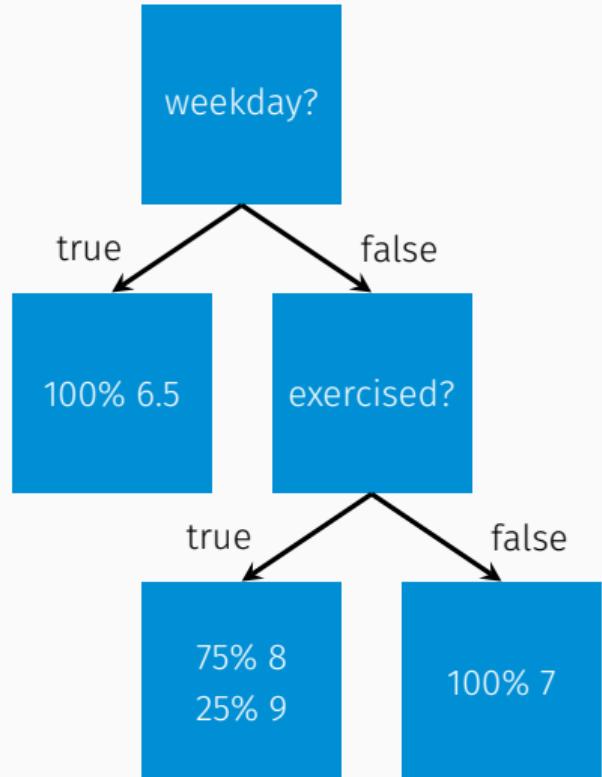
2.10 / EXAMPLE: IRIS DATA SET

- The decision tree logically separates the data into its true classes quite well, although does make some mistakes between *Iris versicolor* and *Iris virginica*.
- However, the decision boundary is quite “jumpy” — a slight change in the input value can lead to an entirely different classification.
- This instability is inherent to decision trees!



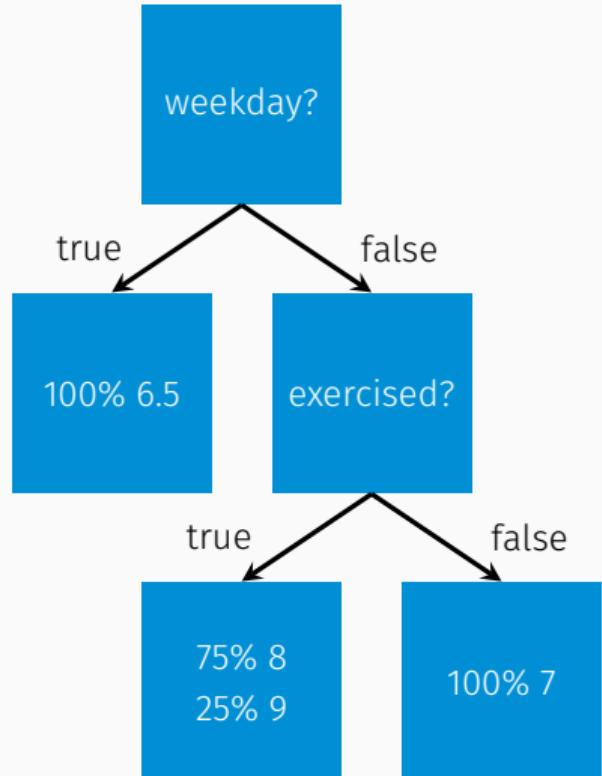
2.11 / DECISION TREE REGRESSION

- Decision trees can also be used for regression:
 - If a leaf node represents a single quantity value, then that value is predicted.
 - More generally, leaf nodes represent many different quantity values, in which case a weighted average value is predicted.
- In the case where a weighted average is used, the outcomes are weighted according to the proportions of samples at the leaf node.



2.12 / DECISION TREE REGRESSION

- For instance, the tree to the right illustrates a model that predicts the number of hours of sleep a person gets every night:
 - If it's a weekday, then we would predict 6.5 hours of sleep.
 - If it's not a weekday, and the person didn't exercise, then we would predict 7 hours of sleep.
 - If it's not a weekday, and the person did exercise, then we would predict $0.75 \times 8 + 0.25 \times 9 = 8.25$ hours of sleep.

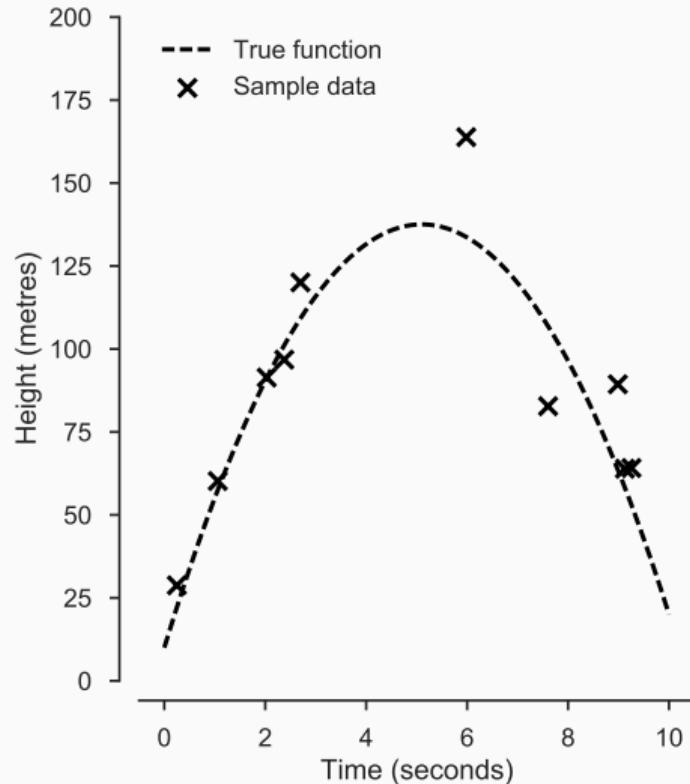


2.13 / EXAMPLE: ROCKET TRAJECTORY

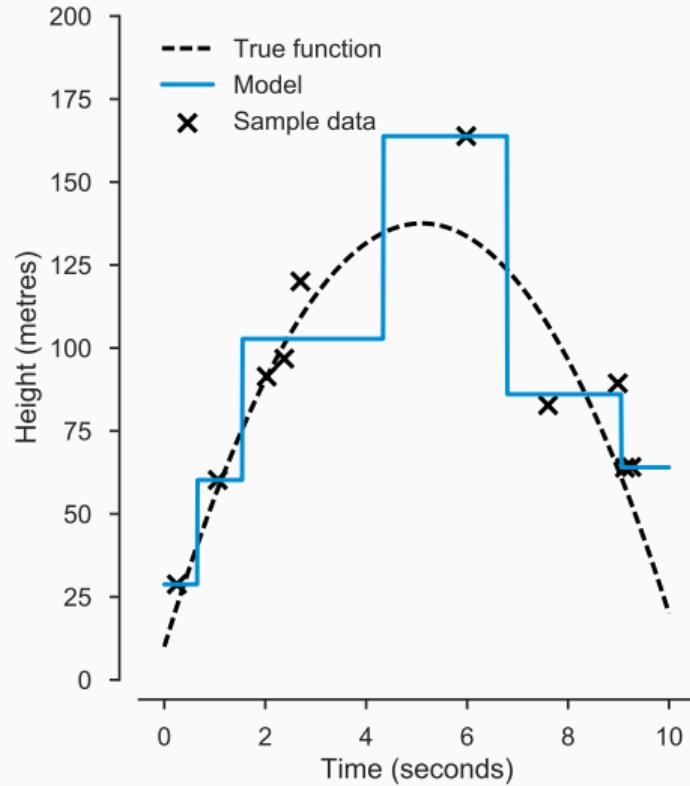
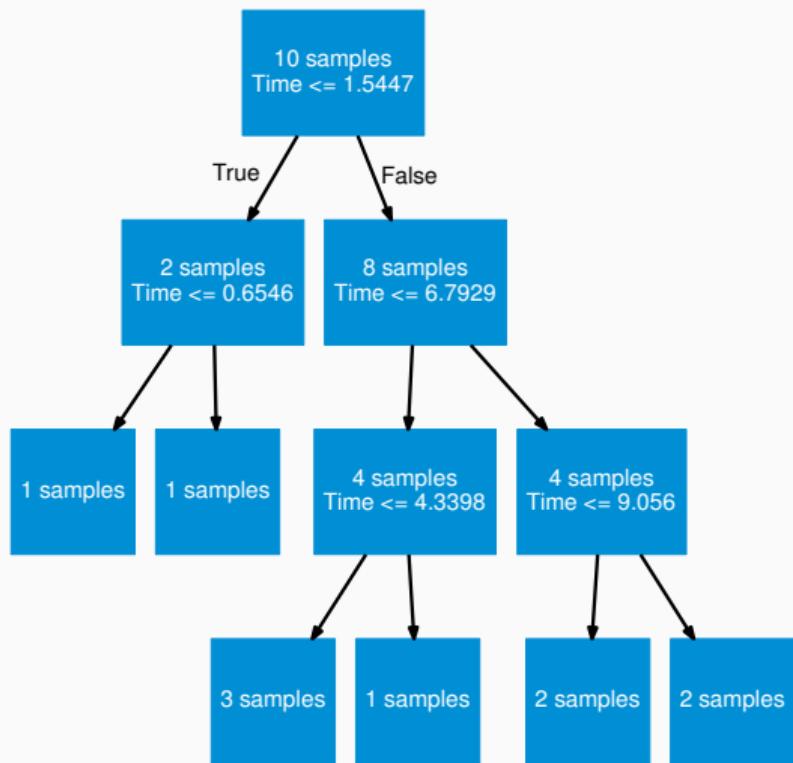
- In an experiment, a rocket is projected into the air at a speed of 50 m s^{-1} from a height of 10 m.
- The true relationship between the height (h) and time (t) is well-known from the laws of physics:

$$h = -4.9t^2 + 50t + 10.$$

- However, we can approximate it using decision tree regression.

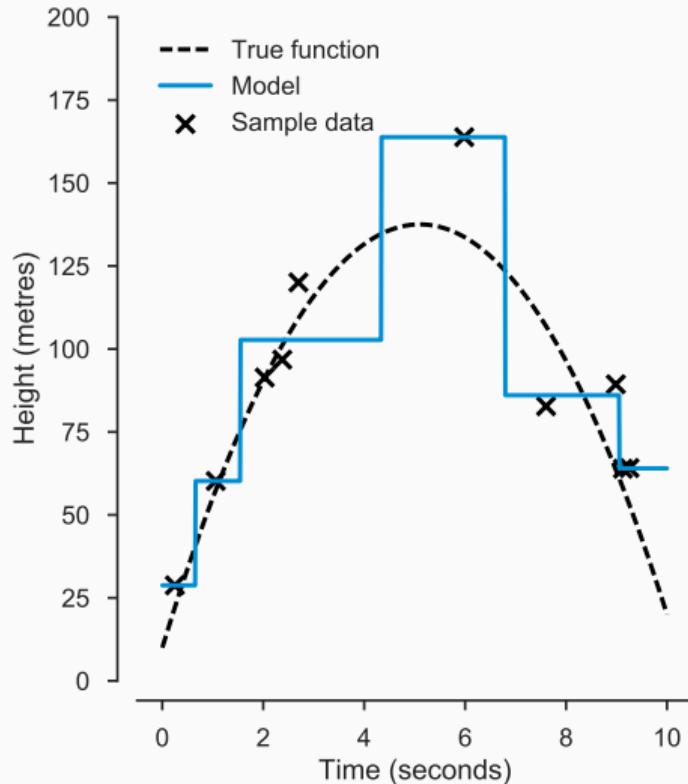


2.14 / EXAMPLE: ROCKET TRAJECTORY



2.15 / EXAMPLE: ROCKET TRAJECTORY

- Unlike linear regression, decision tree regression does not produce a “smooth” line.
- Instead, a discontinuous function is produced, where the fit is optimised within some local neighbourhood of one or more samples.
- These “jumps” mean that small variations in the input can cause large variations in the output, just as with decision tree classification.



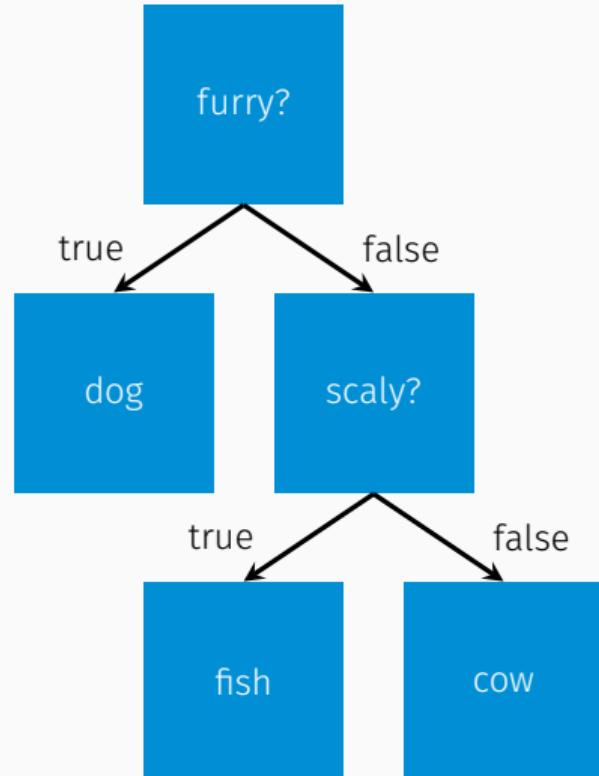
2.16 / ADVANTAGES AND DISADVANTAGES

- Decision trees offer a number of advantages over other kinds of models:
 - They are easy to explain to non-technical audiences due to their hierarchical nature, e.g. flow chart representation.
 - Unlike linear regression, feature generation is not required when dealing with non-linear predictors, nor is multicollinearity a concern.
 - In general, they are robust to the effects of outliers.
 - After fitting a decision tree, it's possible to rank the input features in order of their importance by examining their contribution to the tree structure.
- However, there are also some drawbacks to using decision trees:
 - As trees grow larger, they become much more difficult to explain to non-technical audiences.
 - Decision tree boundaries are fixed and so small variations in the input data can cause “jumps” in the output of the tree (i.e. they tend to overfit).

The CART algorithm

3.1 / DECISION TREE ALGORITHMS

- A number of algorithms exist for creating decision trees, e.g.
 - ID3: classification only.
 - C4.5: classification only.
 - CART: classification and regression.
- For a more complete list, see bit.ly/2n417Sj.
- For the remainder of this lecture, we'll focus on CART, which stands for *Classification And Regression Trees*.
- The decision tree subpackage in scikit-learn uses a variant of CART to build its trees.



3.2 / THE CART ALGORITHM

- The CART algorithm works as follows:
 1. Select an explanatory variable to split on.
 2. Choose one of the explanatory variable values to be the threshold value.
 3. Using an *impurity measure*, calculate the reduction in impurity that results from the splitting the data according to the chosen threshold.
 4. Repeat Steps 2-3 for various threshold values and select the threshold that results in the best impurity reduction.
 5. Repeat Steps 1-4 for each explanatory variable and select the variable split that gives the best reduction in impurity.
 6. Split the data in two, according to the selected variable threshold.
 7. Repeat Steps 5 and 6 until some *stopping criterion* is met.

3.3 / IMPURITY MEASURES

- An *impurity measure* is a mathematical function that quantifies how good a candidate split is.
- If we split a parent node (P) into a left child (L) and a right child (R), then the reduction in impurity is given by

$$\text{Impurity reduction} = I_P - \left(\frac{N_L}{N_P} I_L + \frac{N_R}{N_P} I_R \right), \quad (7.5)$$

where I_X represents the impurity of node X and N_X represents the number of samples at node X .

- There are many ways to measure the impurity at a node, but two common measures are Gini impurity and entropy.

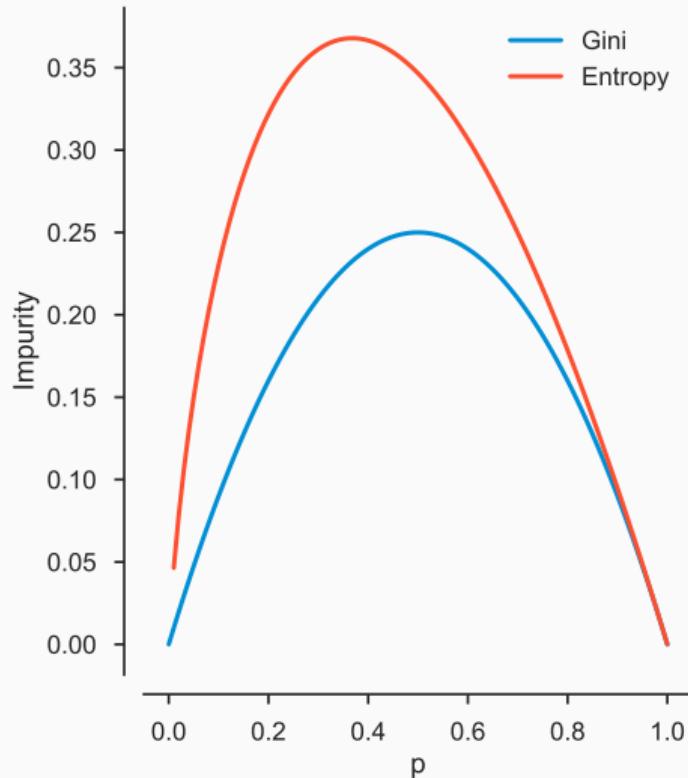
3.4 / IMPURITY MEASURES: CLASSIFICATION

- For a classification problem with k classes, the Gini impurity at node X is defined as

$$I_X = \sum_{i=1}^k p_i(1 - p_i), \quad (7.6)$$

where p_i is the proportion of samples at node X belonging to class i .

- The chart opposite shows the Gini impurity for a problem with $k = 2$ (i.e. $p_1 = 1 - p_2$).



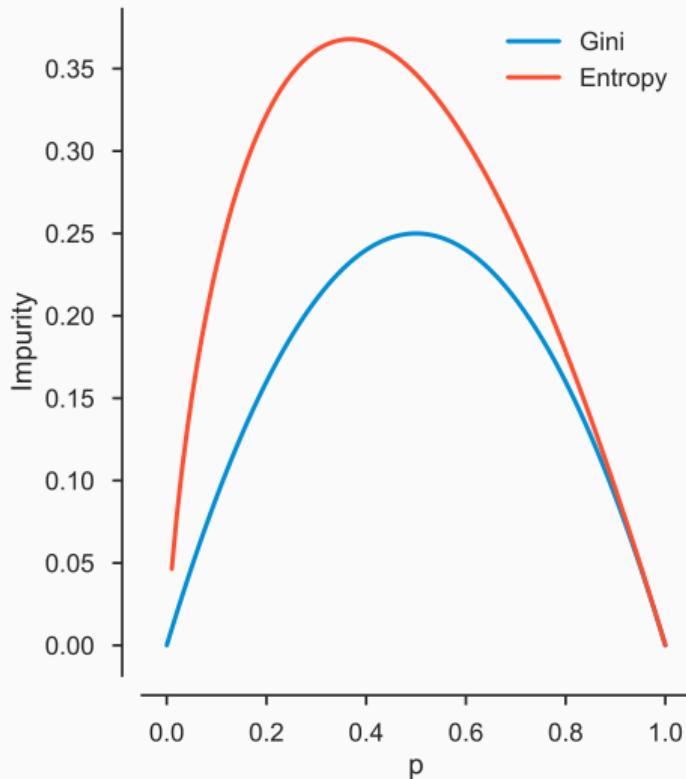
3.5 / IMPURITY MEASURES: CLASSIFICATION

- For a classification problem with k classes, the entropy impurity at node X is defined as

$$I_X = - \sum_{i=1}^k p_i \log(p_i), \quad (7.7)$$

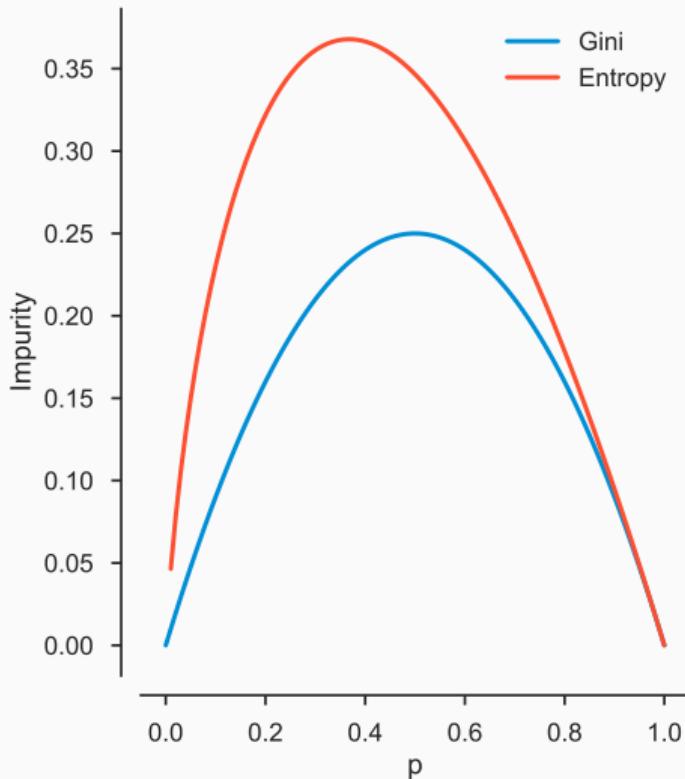
where p_i is the proportion of samples at node X belonging to class i .

- The chart opposite shows the entropy impurity for a problem with $k = 2$ (i.e. $p_1 = 1 - p_2$).



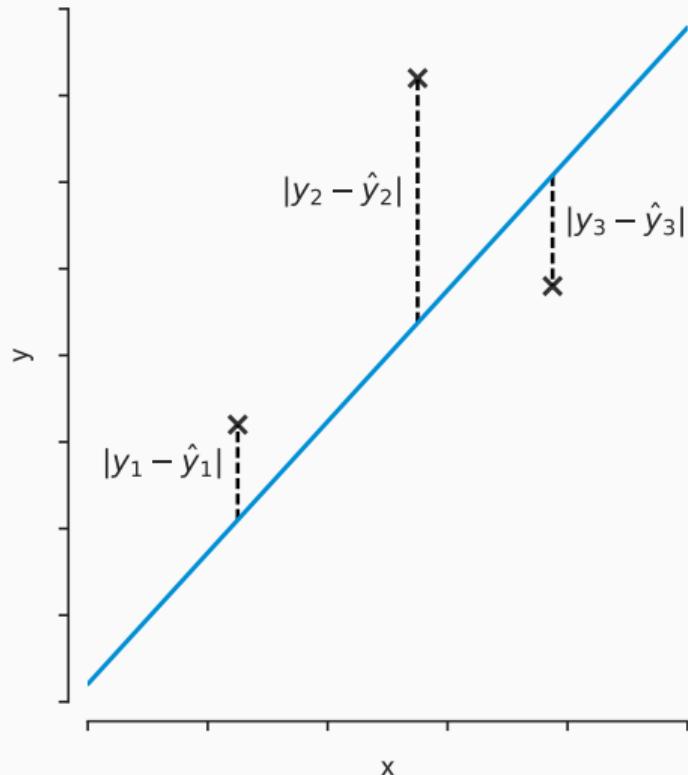
3.6 / IMPURITY MEASURES: CLASSIFICATION

- Entropy tends to select splits on attributes with large numbers of values.
- This isn't ideal in some cases, can lead to overfitting.
- Gini impurity tends to select splits based on the largest class.
- Again, this isn't ideal in some cases, but may be preferable to entropy.
- Can use model selection to choose which to use in practice.



3.7 / IMPURITY MEASURES: REGRESSION

- For regression models, we must a quantitative impurity measure, rather than a categorical one.
- Typically, either mean absolute error (MAE) or root mean square error (RMSE) are used.
- As RMSE tends to be larger than MAE, it tends to select splits that result in more extreme values being split out.
- Again, model selection can be used to decide which to use, if necessary.



3.8 / STOPPING CRITERIA

- A *stopping criterion* is a condition that, when met, halts tree building.
- If we don't specify a stopping criterion, then the input data will be split until we reach a state where each leaf node represents just one outcome.
- This can lead to large and complex trees, *i.e.* overfitted models.
- On the other hand, if specify a harsh stopping criterion, we may stop splitting too soon and produce an underfitted model.
- As with impurity measures, we can use model selection (*e.g.* grid search) to find the best values to use.
- Stopping criteria can be used individually or in combination and are the same for both classification and regression trees.

3.9 / EXAMPLES OF STOPPING CRITERIA

- Minimum samples for a split:
 - If the number of samples available at a node does not reach some threshold value, then do not split it any further.
- Minimum samples to be a leaf:
 - If the number of samples available at a node is larger than some threshold value, then split it; otherwise, make the node a leaf.
 - Guarantees a minimum number of samples in each leaf.
- Impurity reduction threshold:
 - Only split when the impurity reduction is greater than some threshold value.
 - Can lead us to halt at a weak split, when the following split would have added significant value.
- Maximum depth:
 - Only split while the number of levels in the tree is less than some threshold value.
 - Can lead us to miss valuable splits beyond the maximum depth.

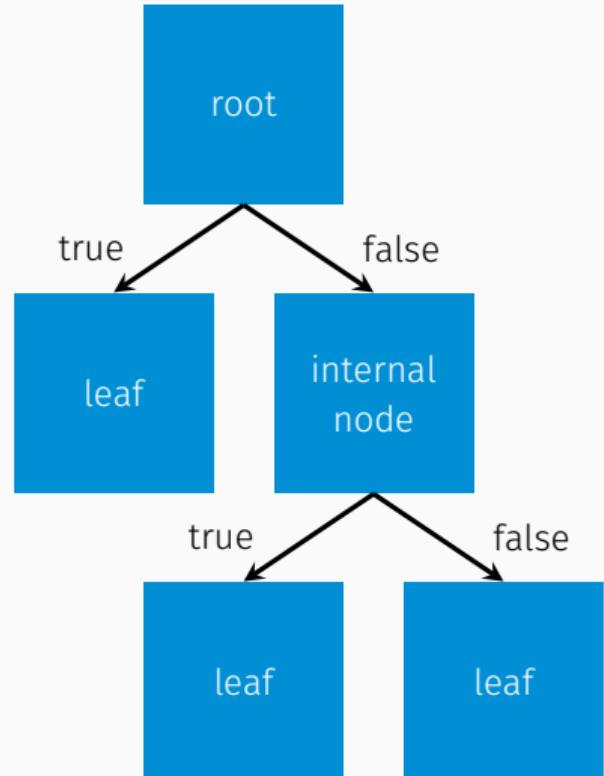
Advanced techniques

4.1 / DECISION TREE PRUNING

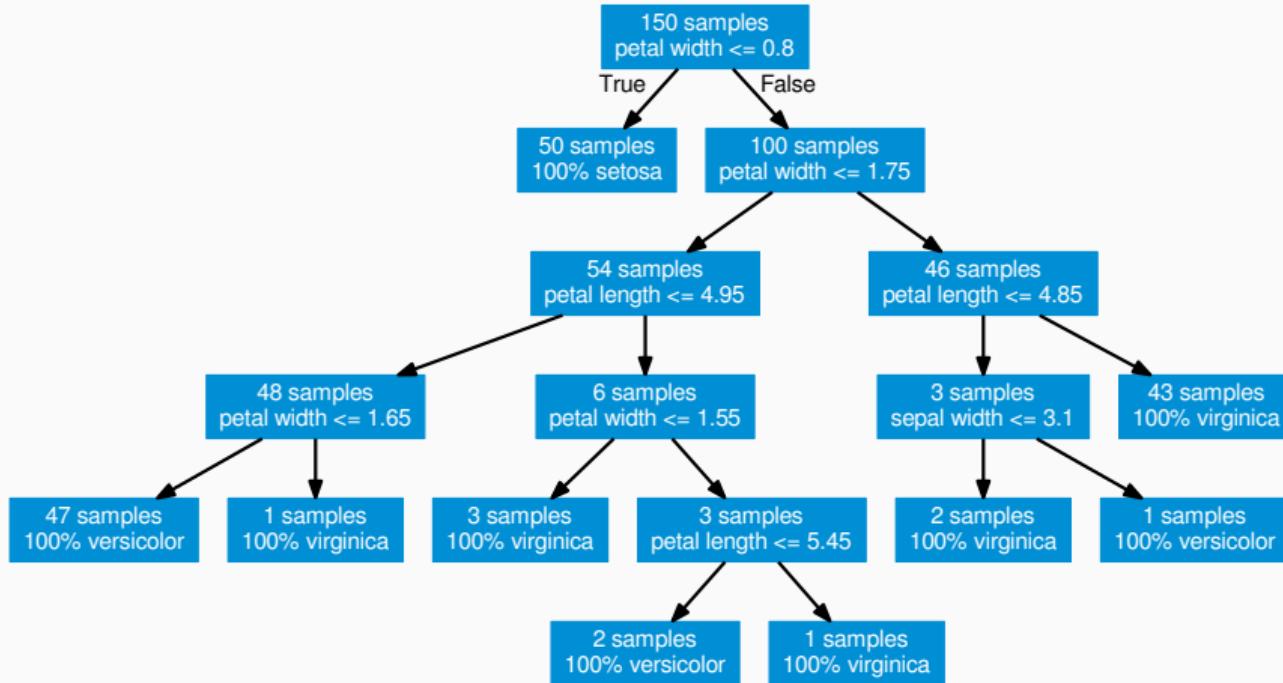
- *Decision tree pruning* is an alternative to using stopping criteria.
- Weak splits are eliminated after the *full* tree is built:
 1. Starting at the tree leaves, each pair of splits in a branch is analysed.
 2. If the splits do not reduce the impurity of their parent by more than some pre-defined threshold, then they are eliminated and their parent is made a leaf node.
 3. The process continues until all leaf nodes are those which reduce the impurity of their parent by more than the threshold amount.
- The idea is similar to using an impurity reduction threshold as a stopping criterion, but ensures that we never miss a valuable split because its parent was weak (*i.e.* it is a bottom up rather than a top down approach).
- Pruning is currently *unsupported* in scikit-learn!

4.2 / EXAMPLE: PRUNING THE IRIS DATA SET

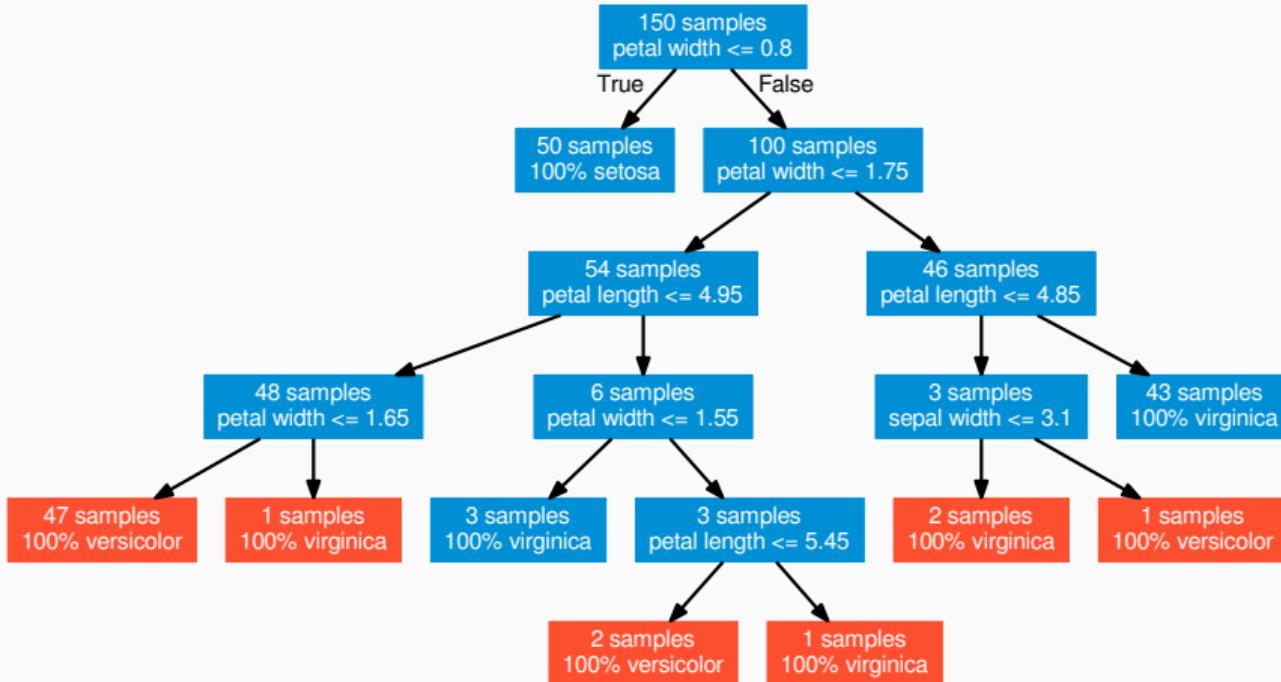
- Earlier, we built a decision tree for the Iris flower data set that was limited to a maximum depth of three levels.
- Instead of stopping early like this, let's build the full tree and prune it.
- Normally, we would prune nodes that don't meet a certain impurity reduction threshold but for simplicity let's prune nodes that have fewer than three samples.



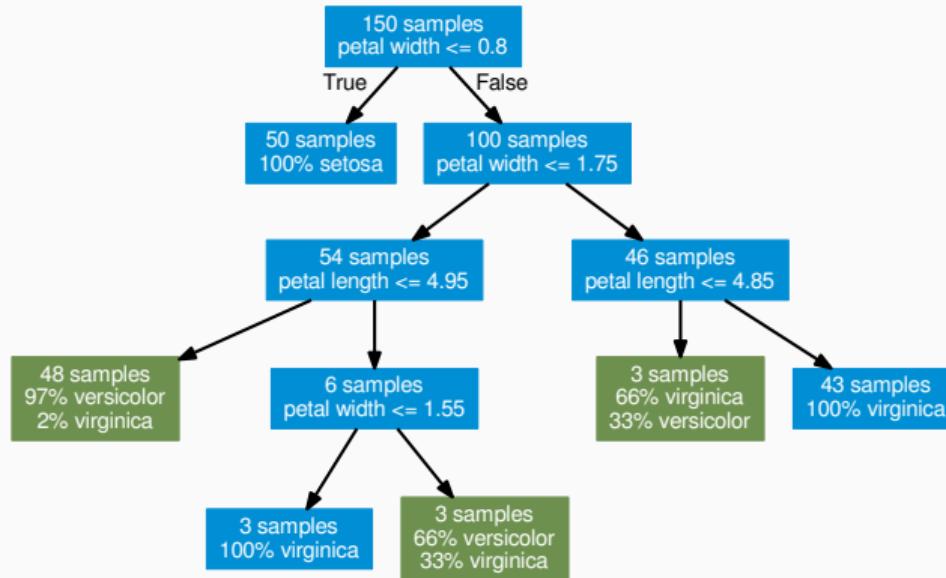
4.3 / EXAMPLE: PRUNING THE IRIS DATA SET



4.4 / EXAMPLE: PRUNING THE IRIS DATA SET

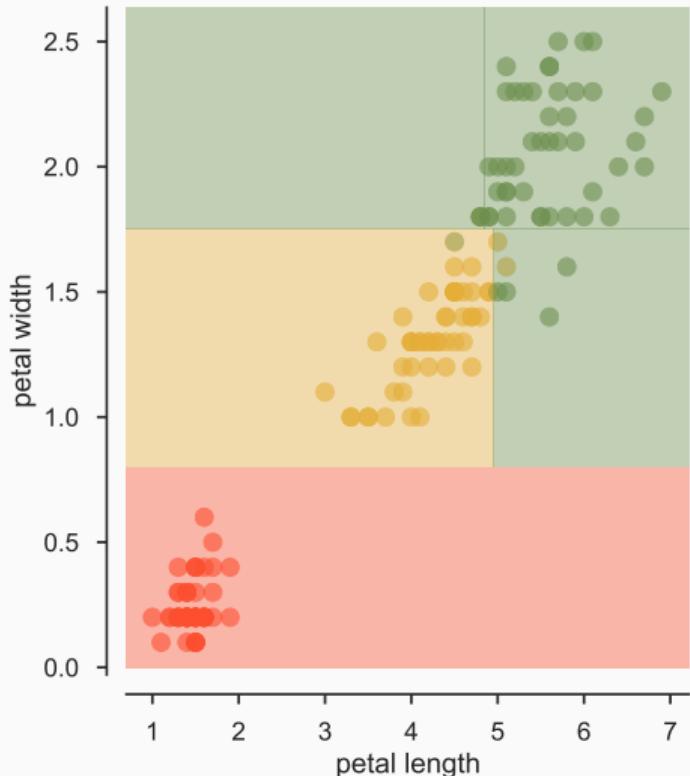


4.5 / EXAMPLE: PRUNING THE IRIS DATA SET



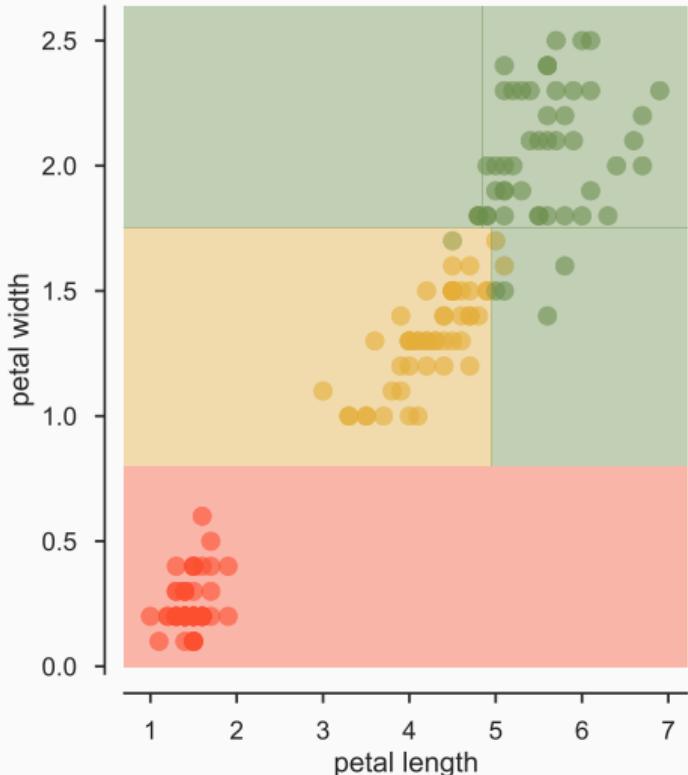
4.6 / RANDOM FORESTS

- Decision trees rely on threshold-based comparisons:
 - This makes them easy to understand and visualise.
 - However, it can make their predictions unstable.



4.7 / RANDOM FORESTS

- For instance, in the chart to the right, if we encounter a set of flowers whose petal widths are all close to 0.8, we will classify some as setosa (red) but others differently (*i.e.* yellow and green), even if they are all the same species.
- One solution to this problem is the use of *random forests*.

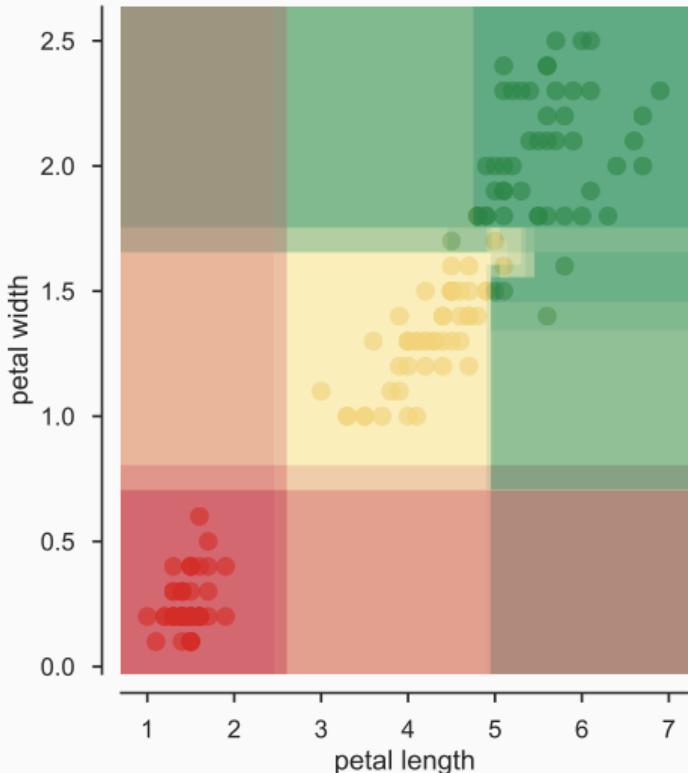


4.8 / RANDOM FORESTS

- A random forest is a predictive model that consists of a collection of decision trees.
- Each tree is trained on a random subset of the data:
 - Individual trees learn different aspects of the data.
 - Trees are assigned weights based on their accuracy.
 - The better an individual tree's predictions, the higher the weight.
- An overall prediction is made by a weighted average/mode of the individual predictions:
 - Good trees have more of a say, poor trees have less of a say.
 - This avoids overfitting and generally produces a more accurate model.
- Random forests are an example of an *ensemble method*, i.e. the combination of two or more weak models to create a stronger overall model.

4.9 / RANDOM FORESTS

- The chart to the right shows the layered decision surfaces of a random forest with ten trees.
- If many trees agree about a particular point or region, then we can have more confidence that the prediction will be correct.
- When the trees disagree, this creates a less rigid decision boundary and the model output is less “jumpy”.



Summary

X.1 / SUMMARY

- Decision trees:
 - Used for classification and regression.
 - Flow chart representation, easy to understand.
 - Hyperparameters: impurity measure and stopping criteria.
 - Pruning: alternative to using stopping criteria, ensures that no valuable splits are missed.
 - Random forests: combine trees to reduce sensitivity around boundaries.
- Lab work:
 - Build a decision tree and random forest classification model.
 - Build a decision tree and random forest regression model.
- Next week: k nearest neighbours!

X.2 / REFERENCES

1. Hastie et al. *The Elements of Statistical Learning: Data mining, Inference and Prediction*. 2nd edition, February 2009. (stanford.io/2i1T6fN)
2. Rokach and Maimon. *The Data Mining and Knowledge Discovery Handbook*. Springer, 2010. (bit.ly/2n417Sj)