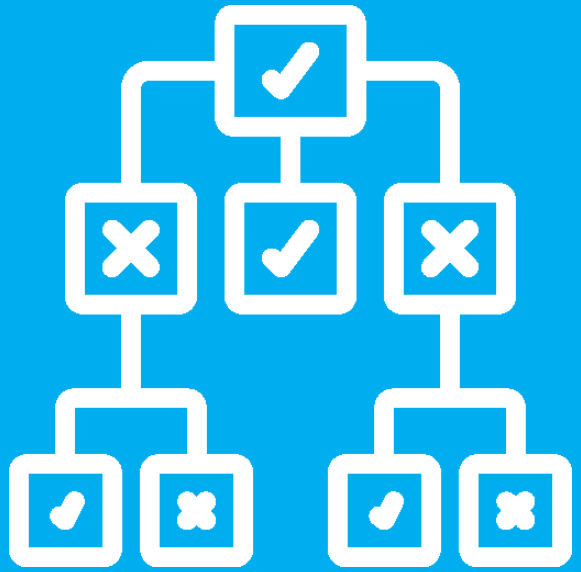
Warwick Business School

Data Science & Generative AI

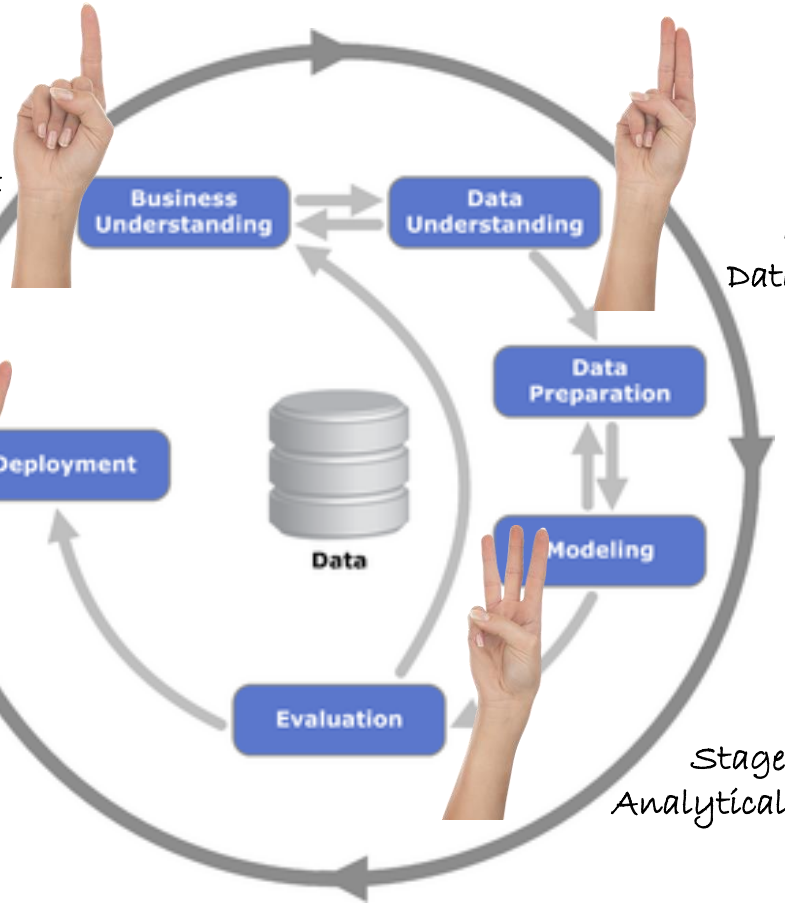
Dr Michael Mortenson
Associate Professor (Reader)
michael.mortenson@wbs.ac.uk



Session 4:
ML Methodology II and Decision Trees

1.1 Stage Three

Stage one:
Business context



Stage two:
Data Engineering

Stage four:
Implementation

Stage three:
Analytical Modelling

1.2 Machine Learning Mindset

Statistics is the science concerned with developing and studying methods for collecting, analyzing, interpreting and presenting empirical data.

UCI Department of Statistics (2024). *What is Statistics?* https://www.stat.uci.edu/what-is-statistics/

1.2 Machine Learning Mindset

"Machine learning (ML) is the process of using mathematical models of data to help a computer learn without direct instruction"

Microsoft (2023)

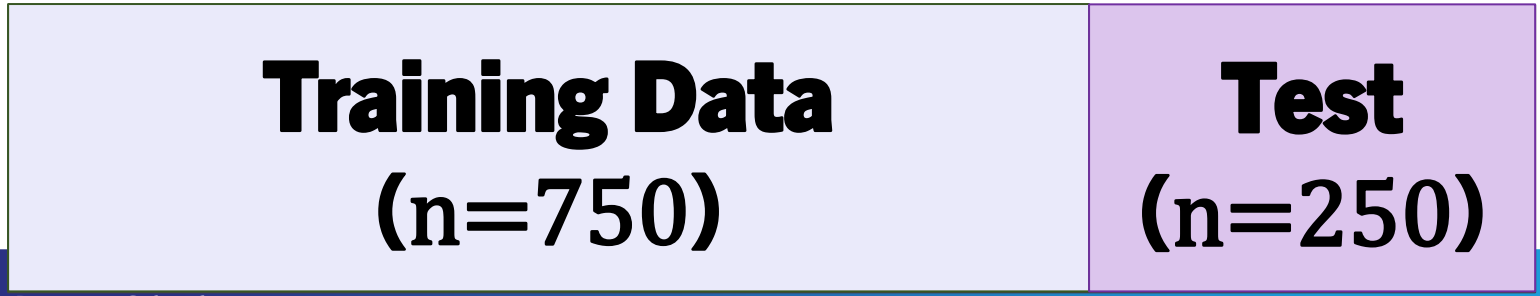
https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-machine-learning-platform/

1.2 Machine Learning Mindset

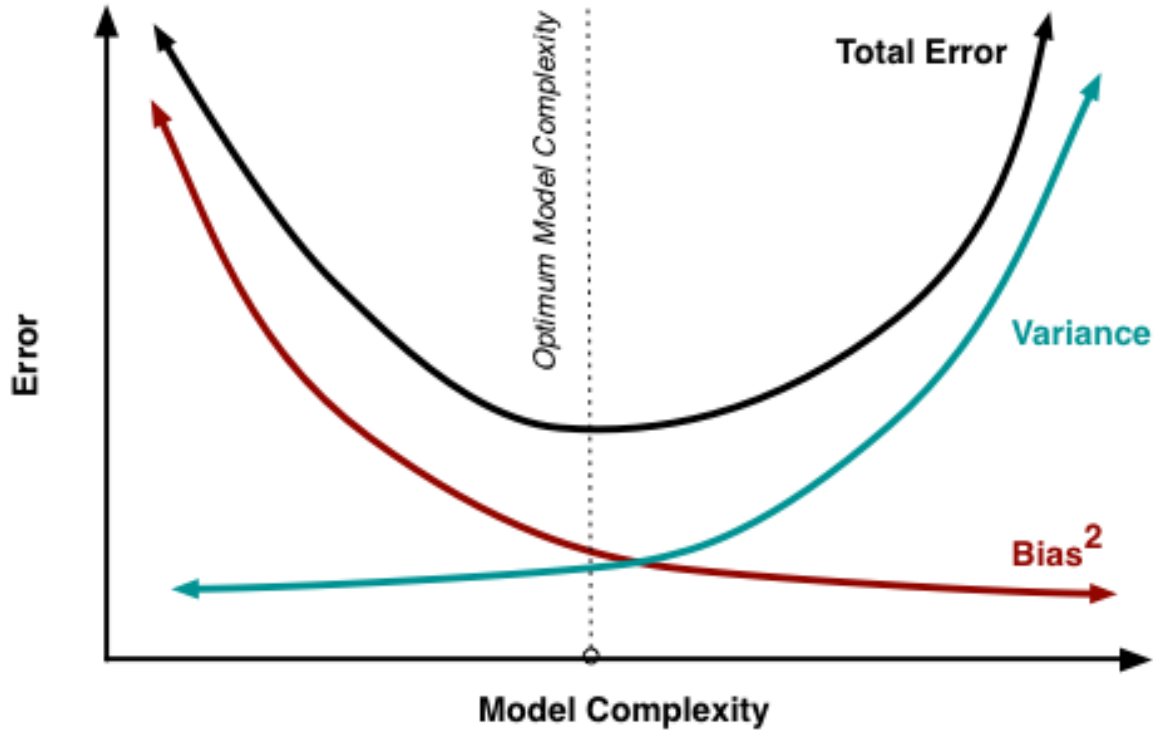
Image Credit:
Gair Rhydd

1.3 ML Methodology Recap

- To assess the (generalisable) learning of a model we need it to predict data it has not seen.
- I.e. what we really care about (usually) is it's ability to predict future data.
- As a proxy for this we can reserve some data to use exclusively for testing.



1.3 ML Methodology Recap



https://www.cs.cornell.edu/courses/cs4780/2015fa/web/lecturenotes/lecturenote12.html

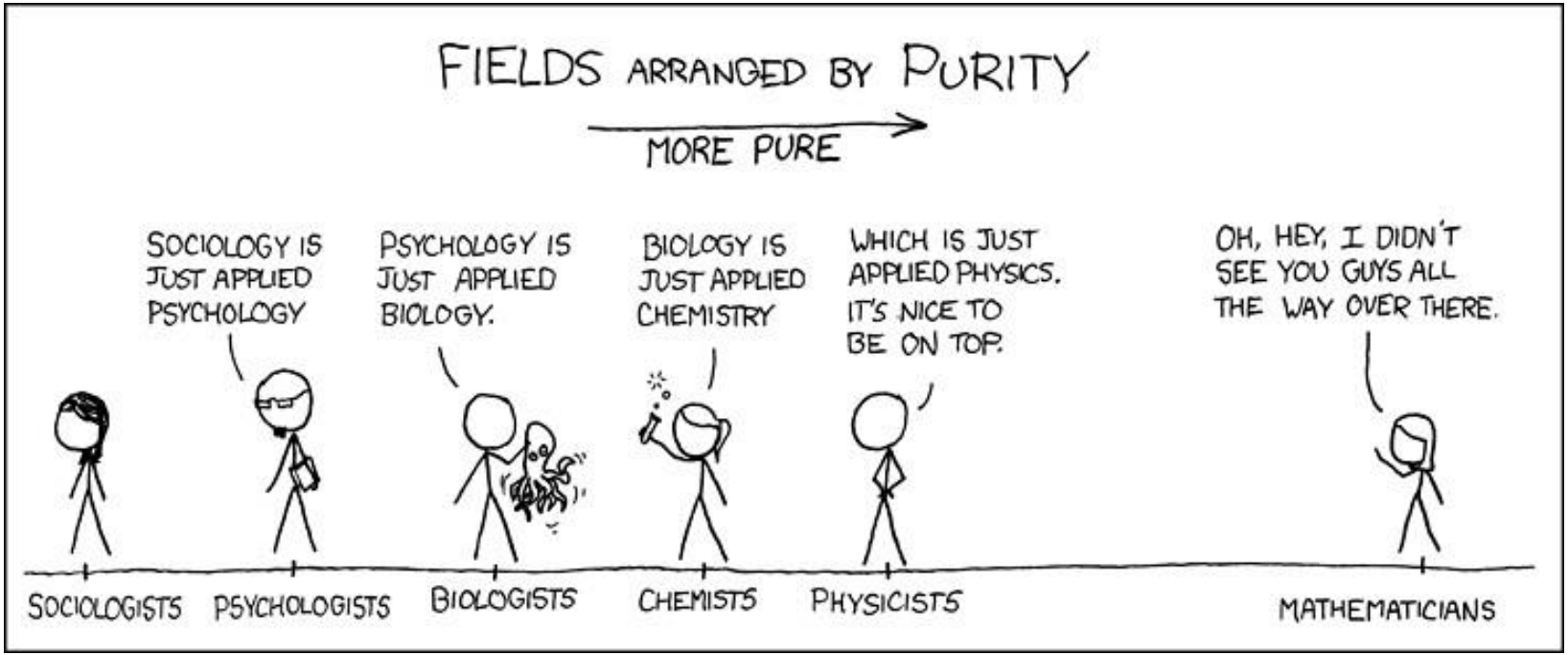
1.3 ML Methodology Recap

Facilitator

Policeman

Image Credit: https://medium.com/summit-plus/decision-making-is-killing-organizations-b2bde26d5aa7

1.4 Practical Value > Theoretical Value



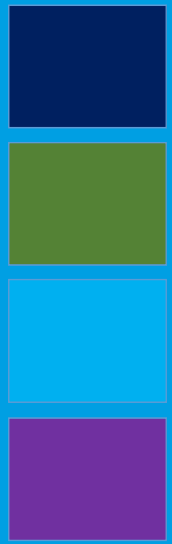
Session Aims

Introduction

Modelling with Machine Learning

Decision Trees

Asynchronous Tasks



2.1 Data Engineering for Machine Learning

- As we have discussed previously, any ML work we do we first must start with business understanding:
  - What problem are we trying to solve?
  - How practically does our solution need to work? A real-time software application? A powerpoint presentation?
  - Is machine learning a good fit?
- After this (and only after this) we need to work on the data. Again, this is critically important and has a huge influence on the success of the project of otherwise.
- This also introduces some ML specific processes…

2.1 Data Engineering for Machine Learning

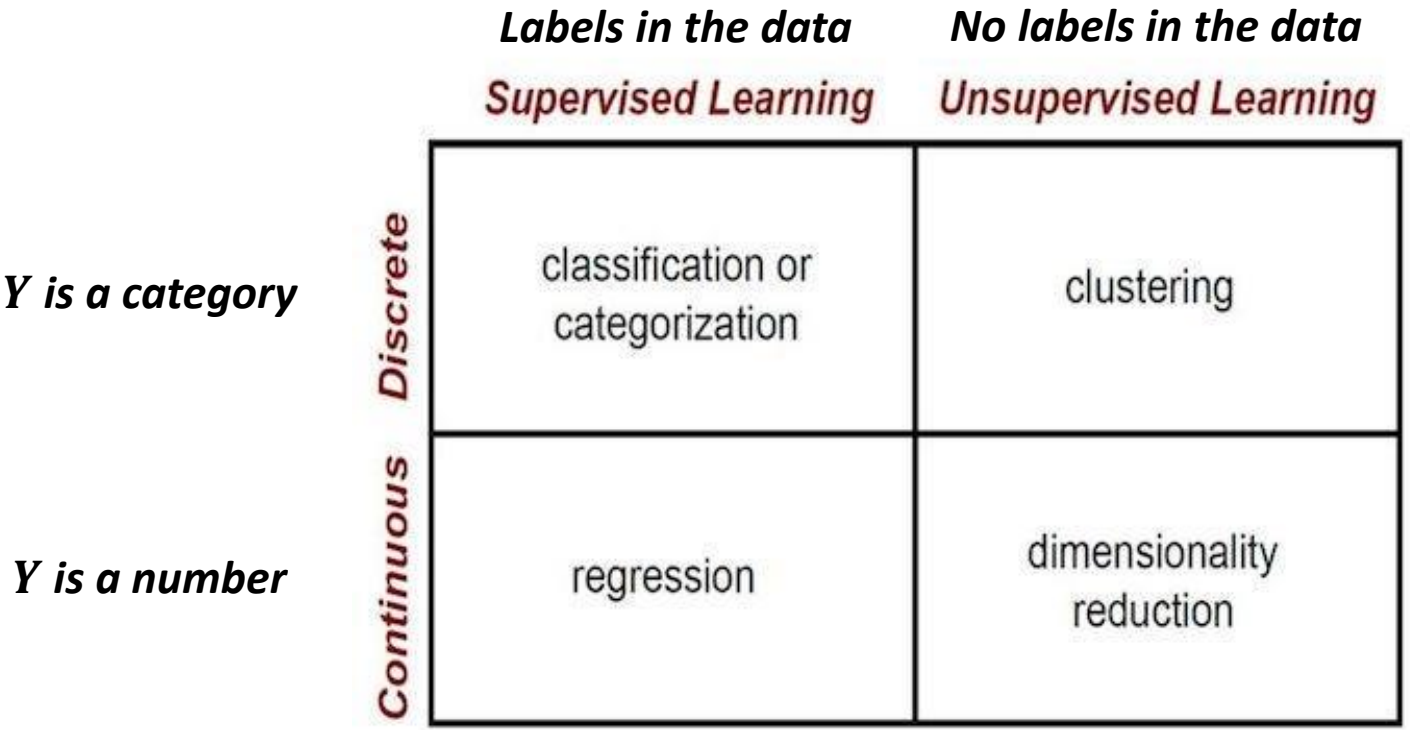


Image Credit: https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d

2.1 Data Engineering for Machine Learning

- **Continuous data**: real numbers (i.e. decimal numbers). This is the data we ideally want for ML models;
- **Ordinal data:** ranked data (e.g. star ratings). Can be treated as/converted to integer format and treated as continuous. E.g. `[School,UG,PG,PhD]` can be converted to `[0,1,2,3]`;
- **Nominal/categorical data:** "name" data which has no logical numerical order. E.g. you ***should not*** change `[Red,Blue,Green]` to `[0,1,2]`;
- **Binary categorical data:** however, if we have a binary category, e.g. `[Live_on_campus, Live_off_campus]` we can change to `[0,1]`.

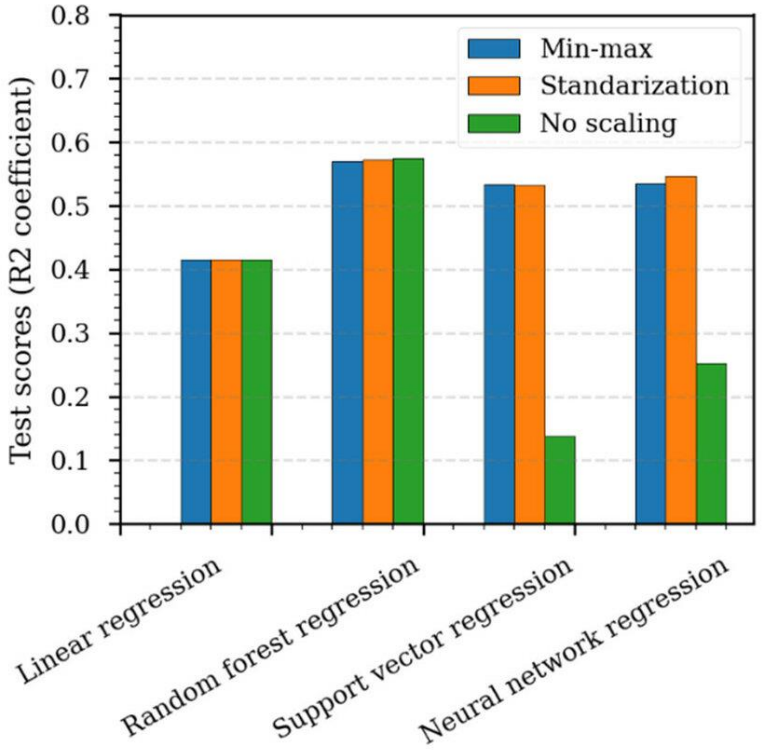
2.1 Data Engineering for Machine Learning

id	color
1	red
2	blue
3	green
4	blue

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

2.1 Data Engineering for Machine Learning

- For many (not all) ML models the *scale* of the features can have an influence on how much the model relies upon them to make decisions:
  - `a = [0.1, 0.5, 0.7]`
  - `b = [123, 456, 789]`
- In this example, for certain algorithms variation in `b` will be much more important than in `a`, sometimes making `a` irrelevant.



Choi JC, Liu Z, Lacasse S and Skurtveit E (2021). Leak-Off Pressure Using Weakly Correlated Geospatial Information and Machine Learning Algorithms. *Geosciences* **11**.

3.3 Scaling Data

- Typically we just scale all features regardless of model used:
  - Min-max scaling:

    $$x_{norm} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

    where x_i is an individual value (cell) in the feature (column) x.

  - Standardisation:

    $$z = \frac{x - \mu}{\sigma}$$

  Where x is a feature; $\max(x)$ is the maximum of x; $\min(x)$ its minimum; μ its mean; σ its standard deviation.

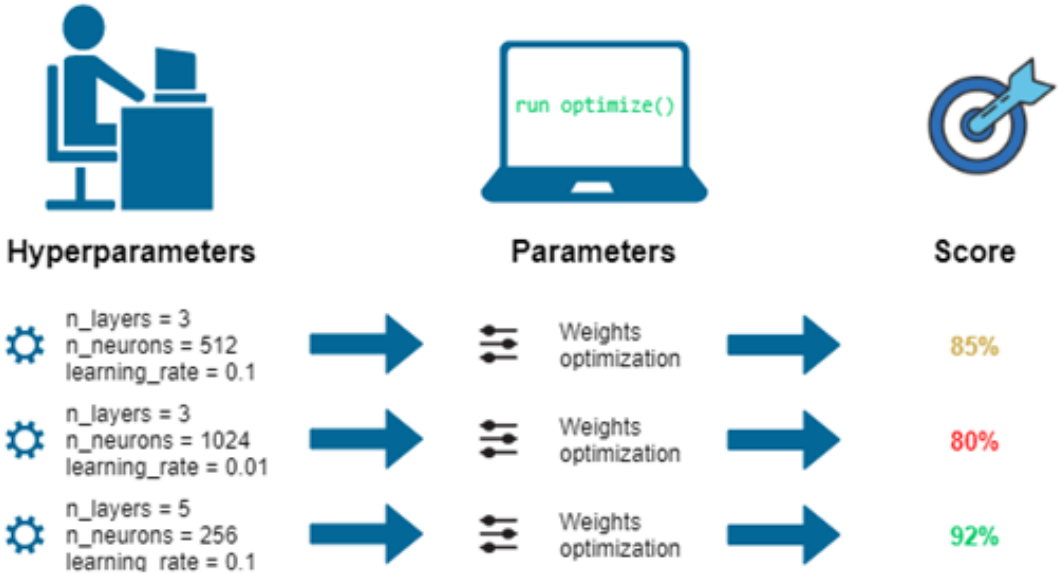
- These approaches will transform all the data to be on the same scale as each other, giving them all equal opportunity to influence the model.

3.4 Algorithms and Models

- In this setting, an algorithm is a set of steps to learn a model. A model is an algorithm that has been trained on our data.
- Based on our analysis of the labels (we will ignore unsupervised problems from hereon), we should know if we are doing classification (Y is a category) or regression (Y is a number). We should be looking at an algorithm(s) appropriate to this problem.
- Some algorithms are regression only (linear regression) or classification only (logistic regression), but most do both.
- Determining the “best” choice of algorithm is very much *it depends*. However, we will look at the most commonly used and some of the factors involved in choosing.

3.5 Scores, Parameters and Hyperparameters

- **Score**: measures of the model's performance.
- **Parameters**: variables specific to the model that determine how it predicts. Parameters **are** the model.
- **Hyperparameters**: user defined parameters that govern the training process of the model. Mostly these balance under and over-fitting.



3.6 Scores

- Choice of “score” for the model largely depends on task.
- For regression we would typically use:
  - Mean Squared Error: $MSE = \sum(y_i - \hat{y}_i)^2$
  - Mean Absolute Error: $MAE = \sum |y_i - \hat{y}_i|$
- These are very similar calculations to those we have already seen.
- For classification these will not work. If our labels are either 0 or 1 (e.g. “pass” or “fail”), our predictions are 0 or 1. It is not meaningful to measure the gap between label and prediction.
- Instead we may use something like accuracy:
  - $Accuracy = \frac{Correct\ Predicitons}{Total\ Predictions}$

3.7 Parameters

- Parameters are the things we learn when training an algorithm to become a model.
- In linear regression … e.g. $Y = \alpha + \beta x$:
  - Y and x are fixed … they are the data.
  - α and β are the parameters. When we learn a line we learn the slope of the line (β) and where it crosses the intercept (α) … the value of Y when x is zero.
- The parameters we learn are specific to the data (they are learned in relation to Y and x) and the choices we make such as regularisation methods.

3.8 Hyperparameters

- Hyperparameters are the nobs and dials we can tweak to guide how the model is learned (normally managing bias vs variance).
- In L1 regularisation we modified the learning objective to add a penalty to the OLS objective … e.g. $OLS + \alpha \cdot \sum |\beta|$. The value of α determines how much we pay attention to the facilitator (line of best fit / least error) or the policeman (minimising the size of the β values). But how do we decide what α should be?
- Potentially we may have some theory on what this should be – if our data is very messy maybe a higher α. However, in ML we care more about results than theory, so normally we just solve this experimentally…

3.1 Hyperparameter Search

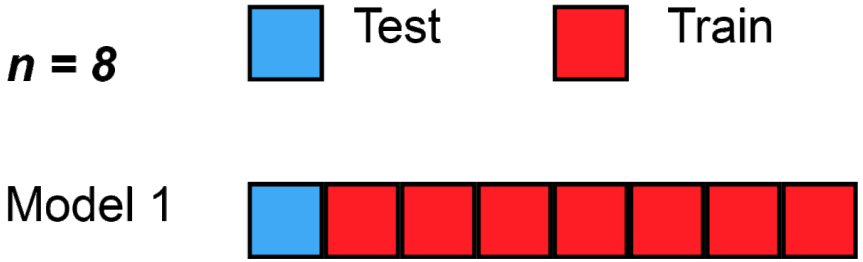
- **Grid search:** We don't know which hyperparameters are best, so we can just search for them. If we have determined the best metric(s) to use (e.g. accuracy), we can compare different combinations of hyperparameters and simply choose the combination that returns the best result. E.g.
  - `criterion = ['gini', 'information gain']`
  - `max_depth = [None, 3, 5]`
  - `min_samples_split = [None, 5, 10]`
  - `max_features = [0.8, 'sqrt', 'log2']`
- This gives 2*3*3*3 = 54 permutations that we search to find the best combination.

3.2 Cross Validation

- ✓ We want to test different hyperparameter choices to find the “best” ones;
- ✓ We do not want to use the test data to do this … test data is just for our final tests;
- ✓ We can assign some of our training data to be (temporary) test data;
- ✓ However, we do not know which data should be used as the test data …
- ✓ **SOLUTION**: use all of it ….

3.2 K–fold Cross Validation

- Split your training data (randomly) in to k separate chunks (e.g. 8);
- For each hyperparameter combination (chosen by grid or random):
  1. Allocate the 1st chunk as test, train on the remaining 7 and then test the model on the test set (1st chunk);
  2. Repeat step 1 seven times each time changing the test set (2nd chunk, 3rd chunk … kth chunk);
  3. Average the results of the eight experiments to get the score for that hyperparameter combination.



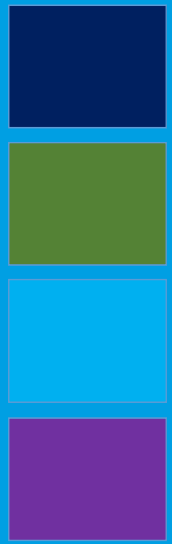
Session Aims

Introduction

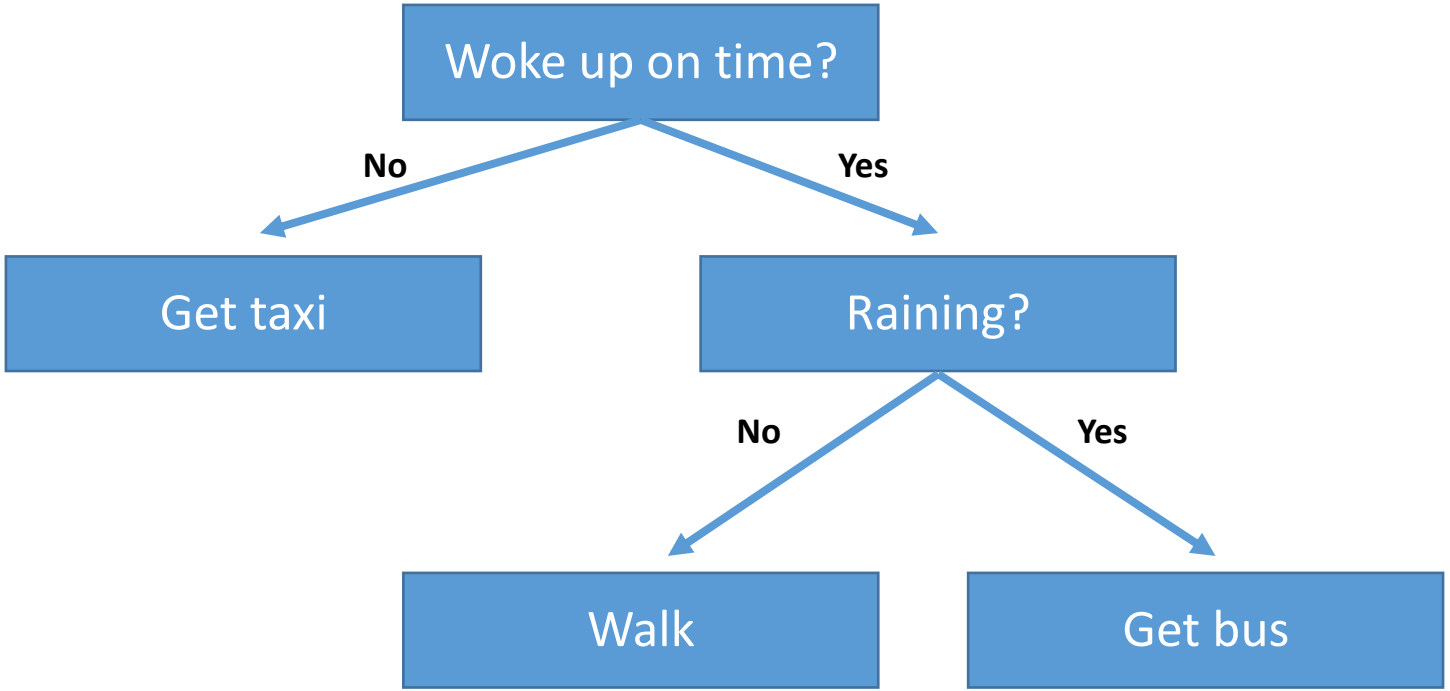
Modelling with Machine Learning

Decision Trees

Asynchronous Tasks

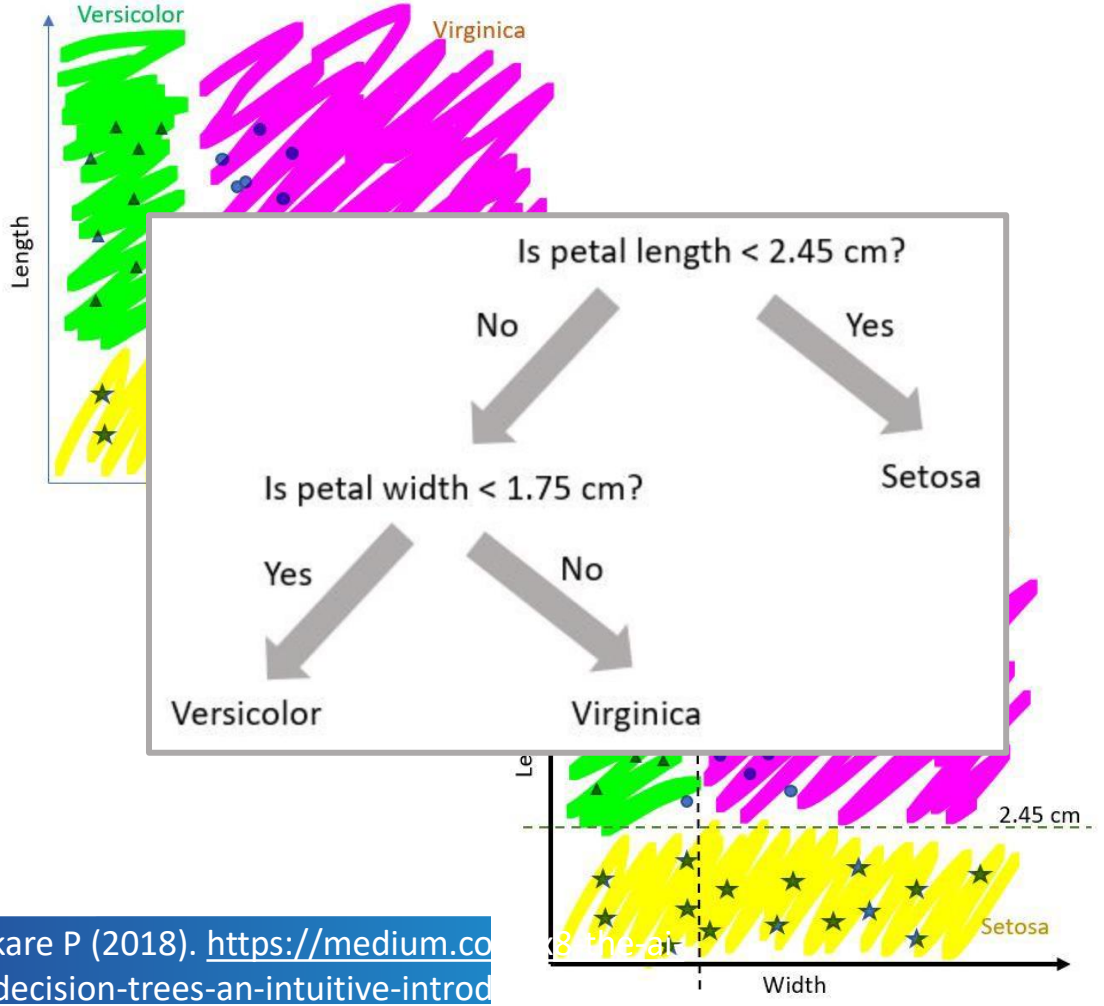


3.1 Decision Trees (human version)



3.2 Decision Trees by Eye

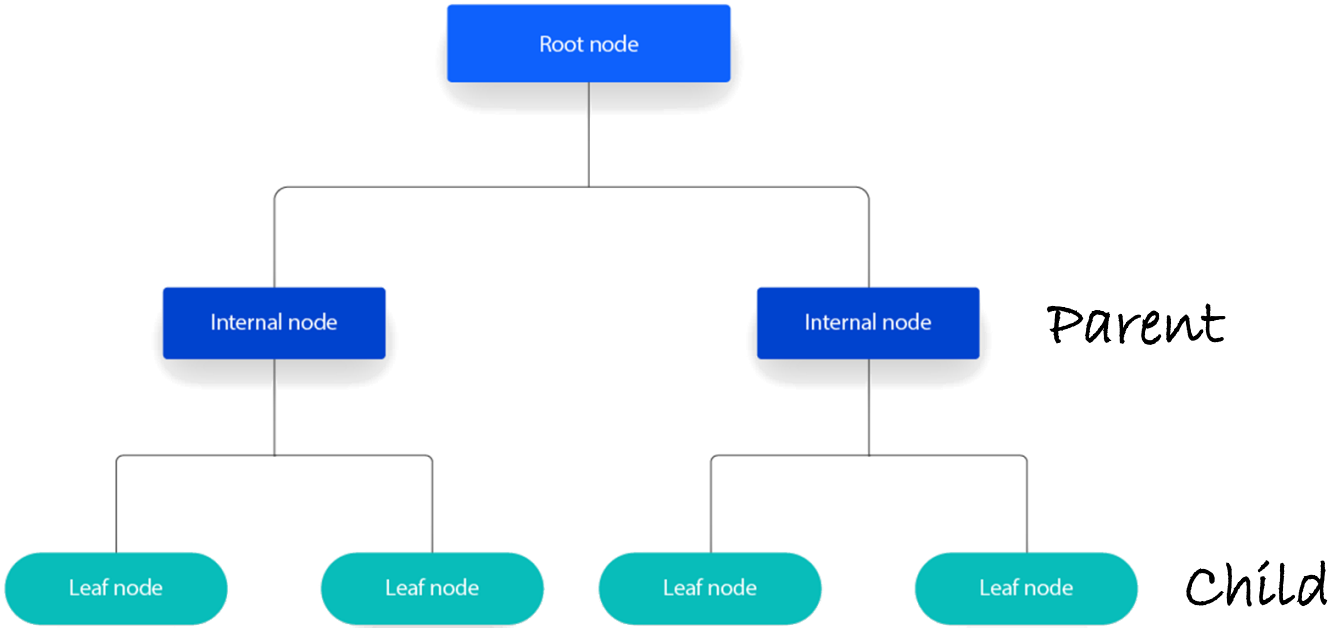
1. Plot the data onto a graph based on the most influential features
2. Identify a line which separates one or more class based on a specific value rule
3. Repeat (2) until all the classes are separated
4. Generalise the above to create an overall decision rule for the problem



Adapted from: Karkare P (2018). https://medium.co
community/decision-trees-an-intuitive-introd

3.3 Decision Trees by Algorithm

- Decision trees find a set of *if > else* conditions in which to split the data into classes.



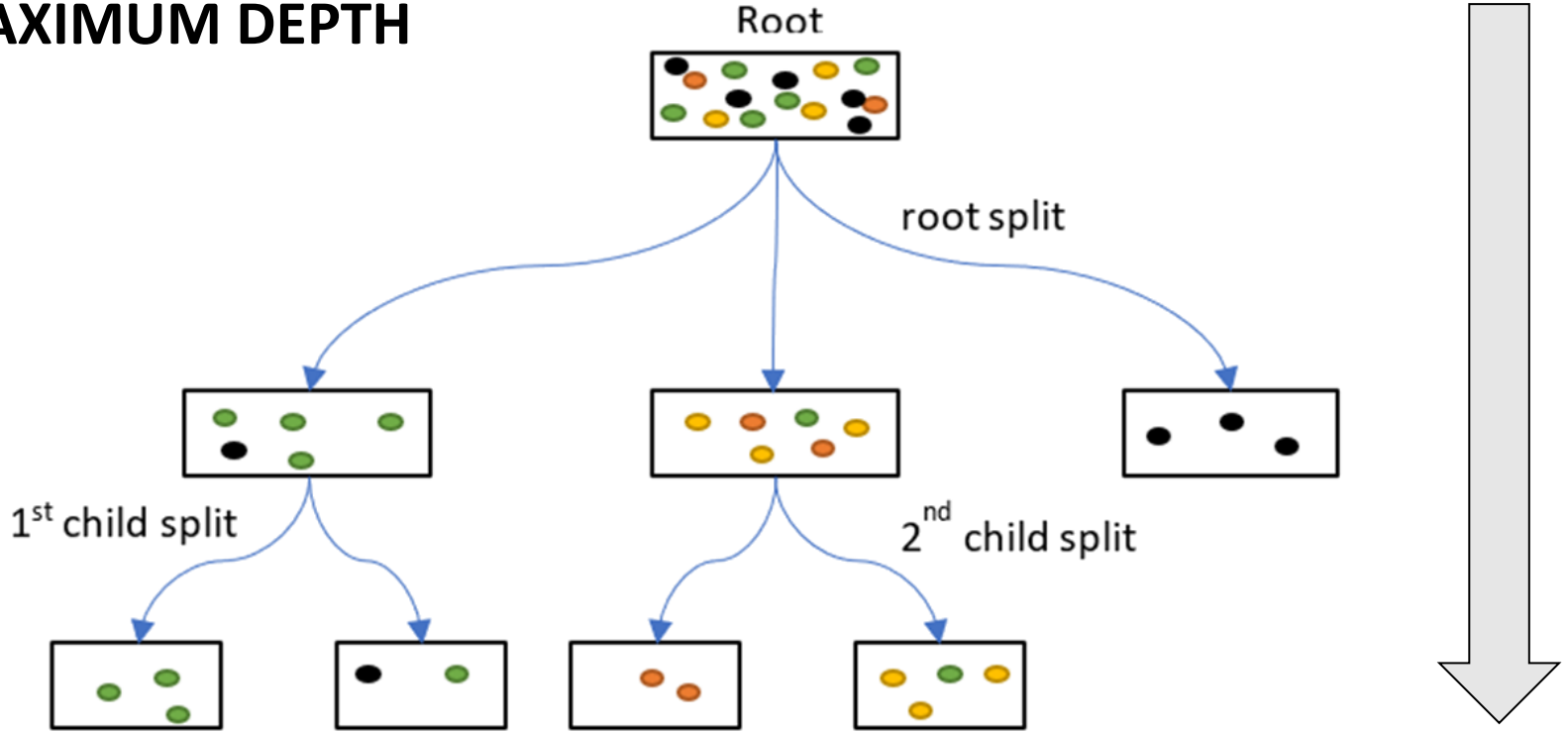
Source: https://www.ibm.com/topics/decision-trees

3.4 Decision Trees Hyperparameters

- How can the decision tree determine what is the best condition to split the data on?
  - The goal is to find a split that results in the “cleanest” divide between classes … e.g. in a 2-class (binary) problem ideally we want a split that puts mostly class 0 on one side and/or mostly class 1 on the other.
  - With this in mind, each split is evaluated according to one of two different criteria *Information Gain* or *Cross Entropy*.
  - We could explain each of these mathematically, but actually we don’t really need to. They are both just measures of how good a split each possible split would be, and effectively which is best is just *“it depends”*. Instead we can just treat it as a hyperparameter.

3.4 Decision Trees Hyperparameters

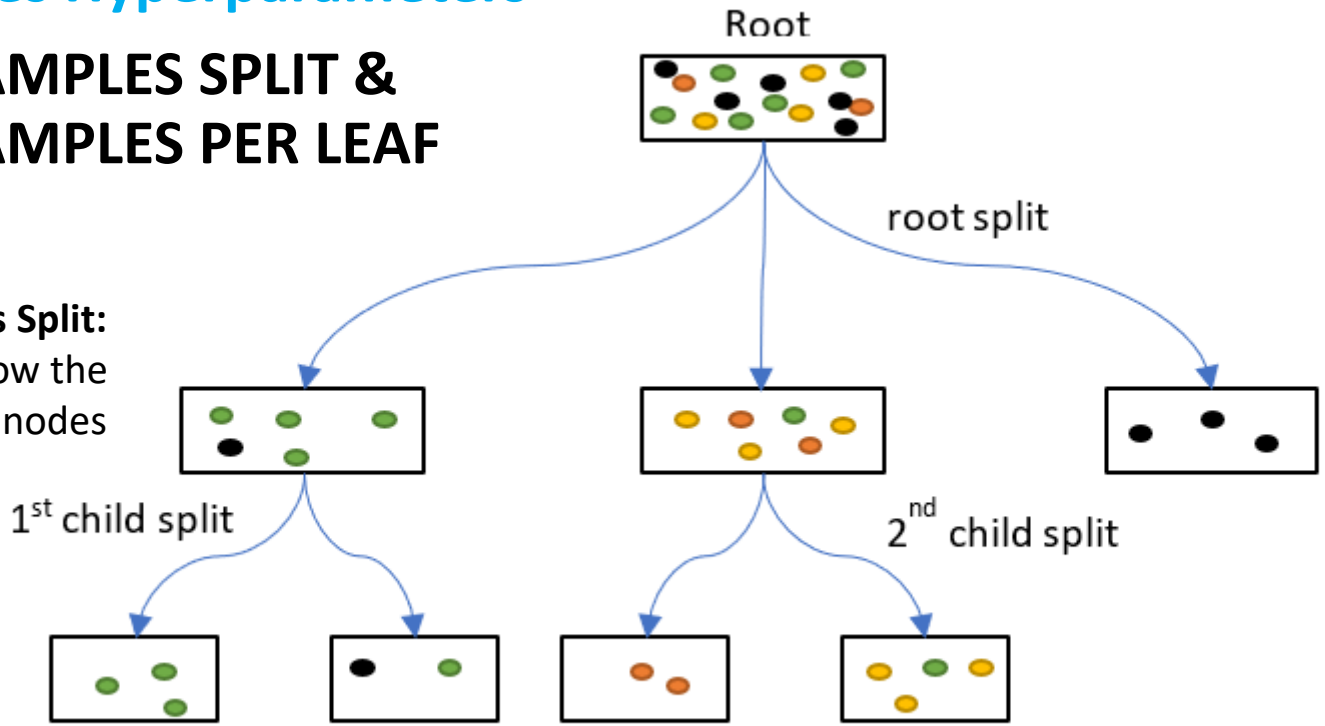
- **MAXIMUM DEPTH**



3.4 Decision Trees Hyperparameters

- **MINIMUM SAMPLES SPLIT & MINIMUM SAMPLES PER LEAF**

Min Samples Split:
If samples >= 5 then allow the node to split to child nodes



Min Samples Leaf:
Only allow a leaf to be created if it has >= 2 samples

3.4 Decision Trees Hyperparameters

- **MAXIMUM FEATURES**

  Max features controls how many features are considered when picking a split condition. I.e. we may want to only consider a random selection of 80% of the features each time.

max_features : *int, float, string or None, optional (default=None)*

The number of features to consider when looking for the best split:

- If int, then consider max_features features at each split.
- If float, then max_features is a fraction and int(max_features * n_features) features are considered at each split.
- If “auto”, then max_features=sqrt(n_features).
- If “sqrt”, then max_features=sqrt(n_features).
- If “log2”, then max_features=log2(n_features).
- If None, then max_features=n_features.