Biologically Inspired Computation

Dr Marta Vallejo

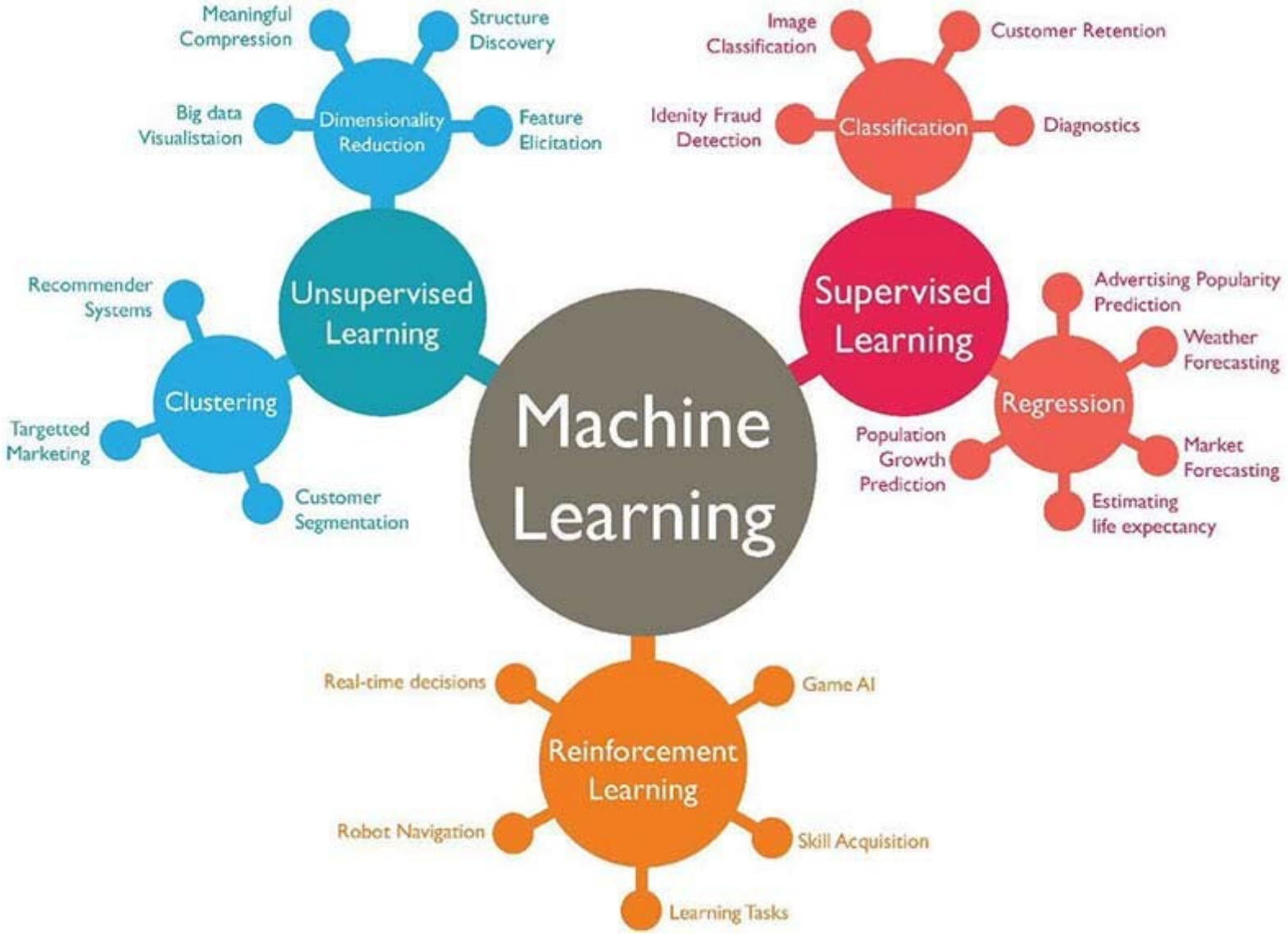
m.vallejo@hw.ac.uk

Some machine learning concepts

These are some concepts that we need to be familiar with when we speak about Neural Networks and Evolutionary Algorithms:

1. Learning Paradigms
2. Offline/Batch vs. Online Learning
3. Structured vs. Unstructured Data
4. Search Space: complexity
5. Local/Global Maxima and Minima
6. Size of the Search Space
7. Too-big problems
8. Polynomial and Exponential Complexity
9. Exact & Approximate Algorithms
10. Metaheuristics
11. Evaluation of our Model
12. Generalisation: Overfitting / Underfitting
13. Bias – Variance trade-off
14. Linear Separability
15. Outliers

Learning Paradigms

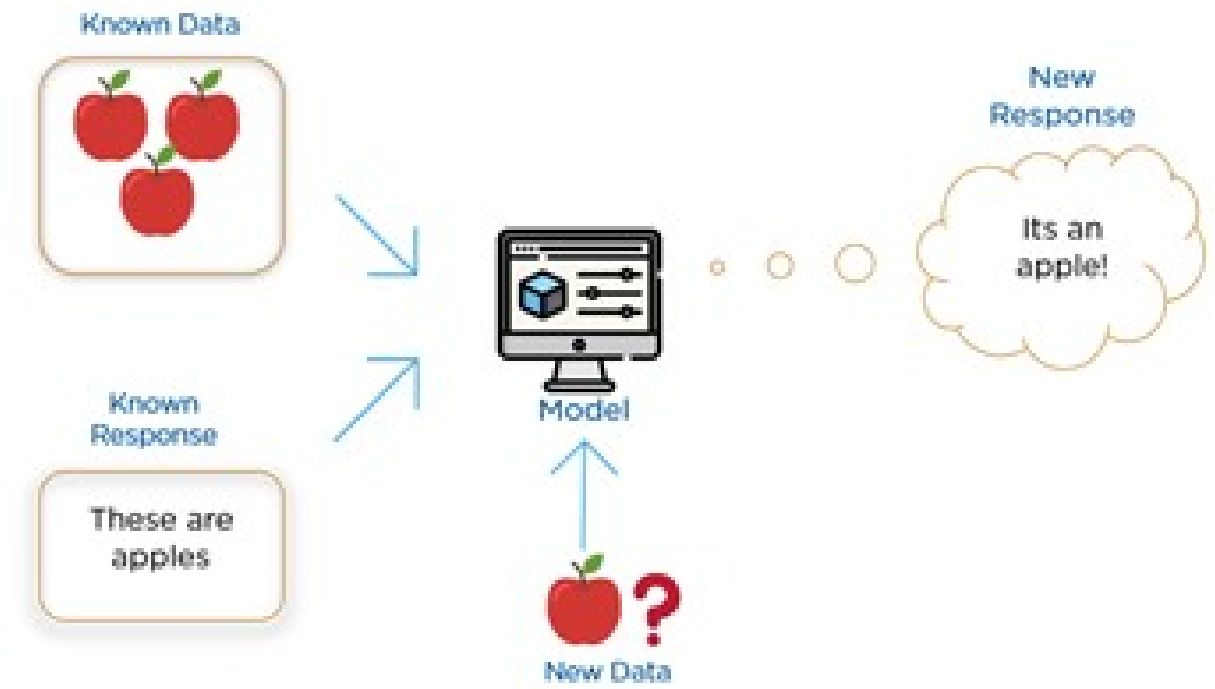


Learning Paradigms

Supervised learning

Supervised learning consists of inferring a function from **labelled** training data. Each sample in this data is a pair consisting of an input vector x and a desired output value y (label).

The goal is to learn the **mapping** from x to y, producing a **generalisation** function, which can be used for mapping new examples that do not belong to our data.



Learning with a teacher

Learning Paradigms

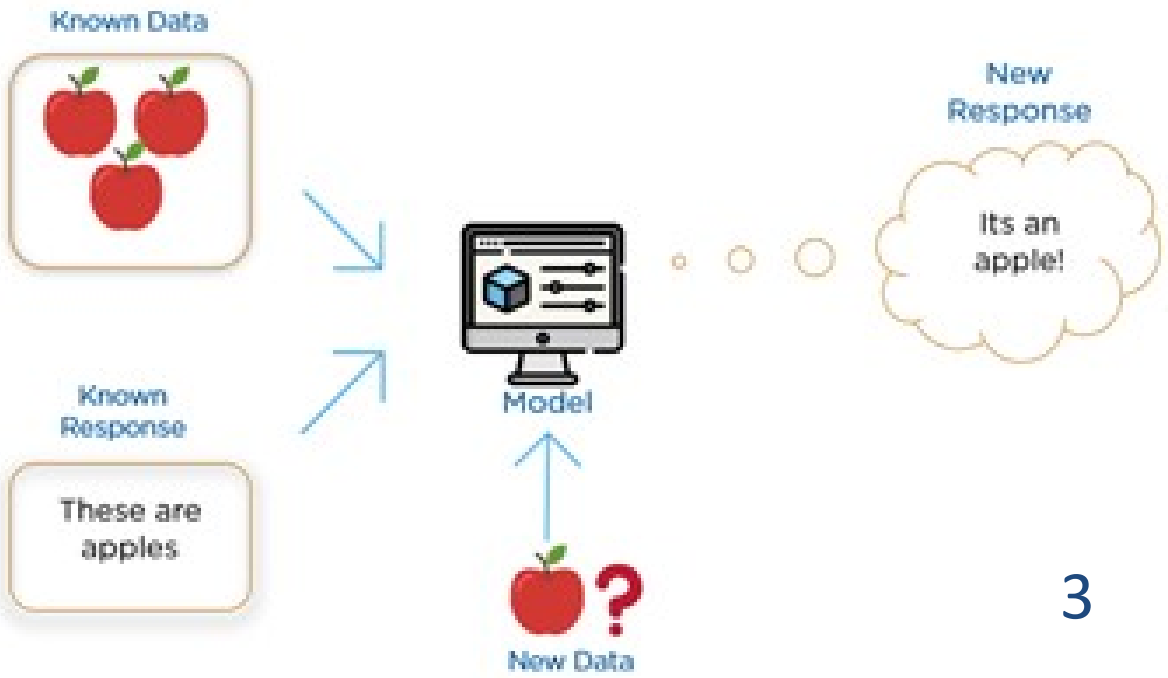
Unsupervised learning

A learning task that consists of inferring a function from **unlabelled** training data formed by a set of training samples. Each sample has only an input object without output values.

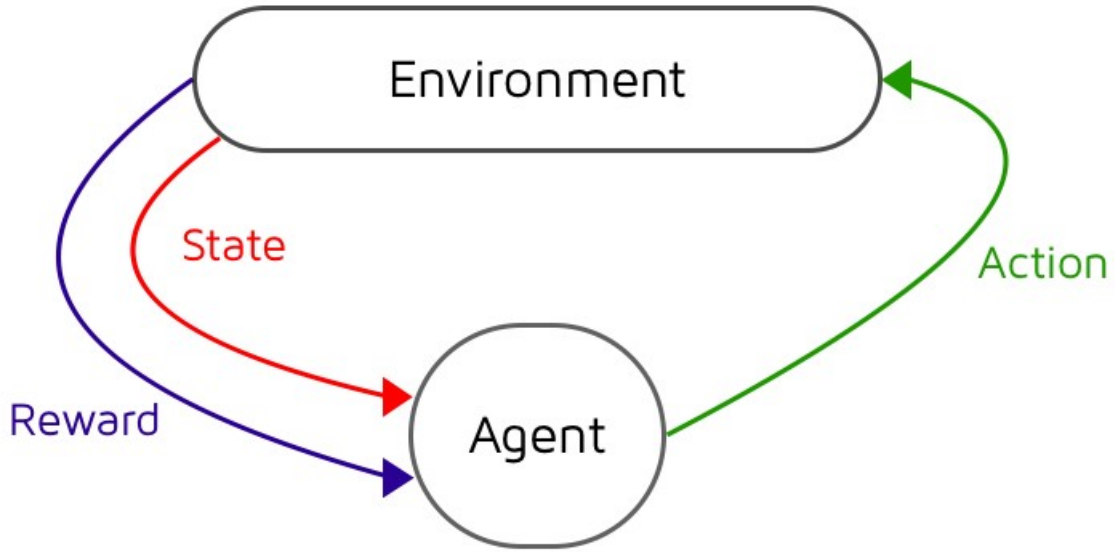
The algorithm analyses the training data grouping them with a **similarity metric,** without receiving feedback to indicate the desired output.

The goal is to discover some **hidden structure** in the data. This is an autonomous learning and there is no external control on the error.

More complicated than supervised learning



Reinforcement Learning



Reinforcement learning (RL) uses a trade-off between exploring unknown areas and the application of existing knowledge through a trial and error process.

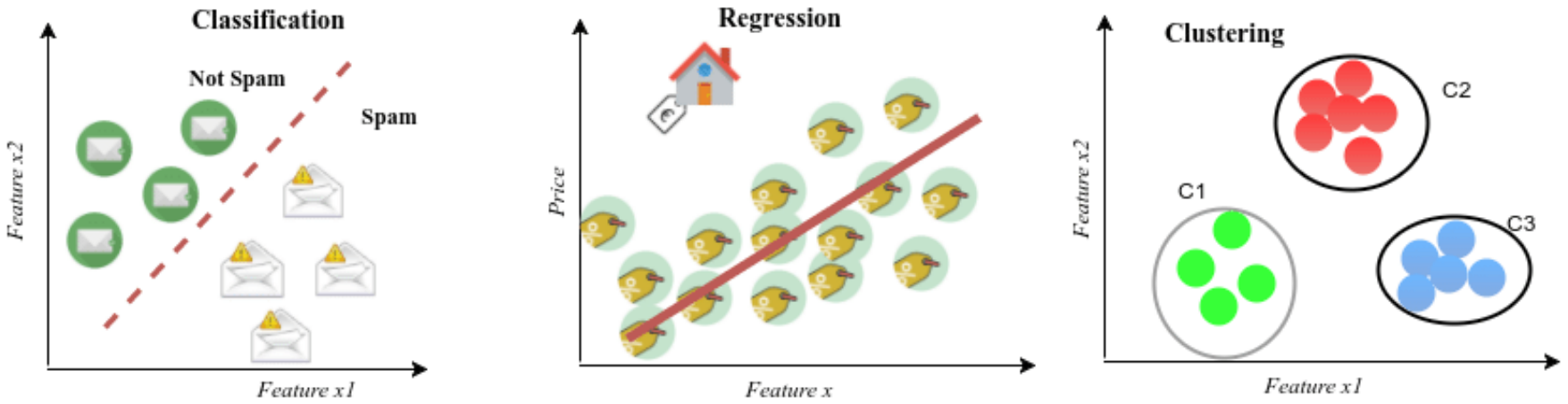
RL explicitly considers the problem as a goal-directed group of agents interacting with an uncertain environment receiving **reward** and **punishment** for their actions.

Learning Paradigms

Classification: the labels can only take a finite set of values (categories).

Regression: labels can take any real continuous value.

Clustering: Grouping similar points together within clusters.



Offline/Batch vs. Online Learning

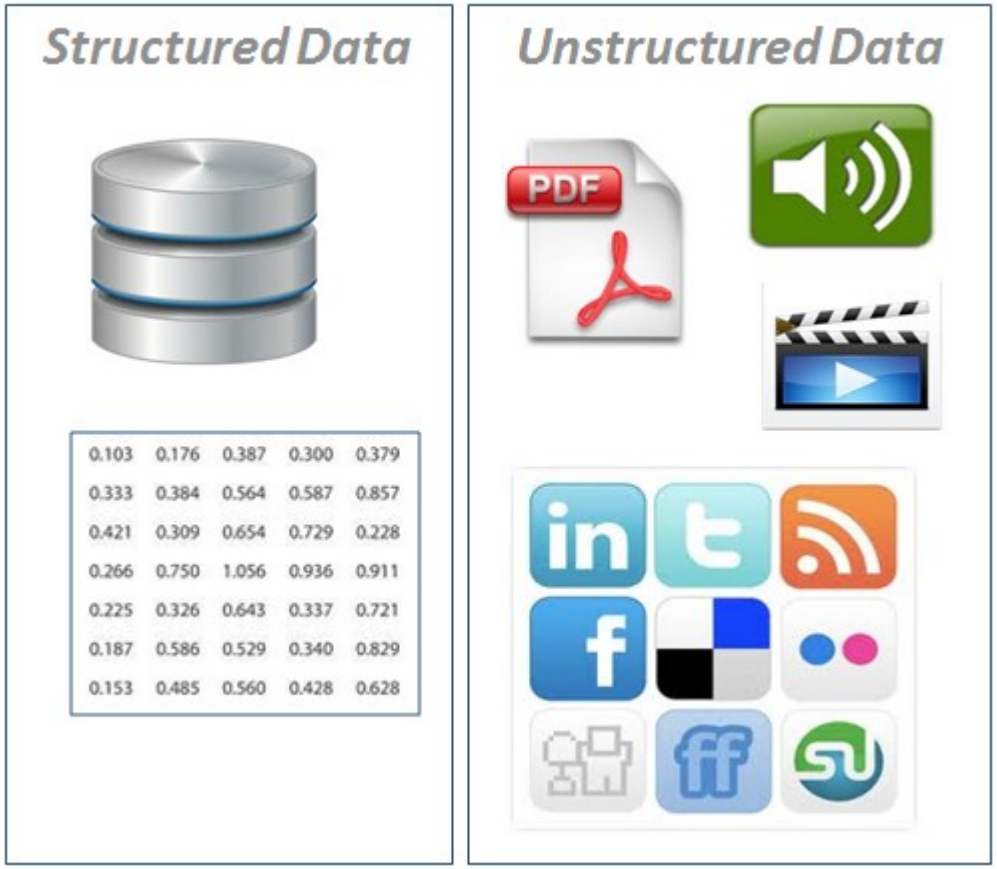
Batch Learning:

All data given at once

Online Learning:

- Data processed sample by sample
- Parameters are updated on each new instance
- Faster learning
- Suitable when data changes with the time or we do not have access to all the training data in advance
- Can cause residual errors if we are dealing with an outlier sample

Structured vs Unstructured Data



Structured Data

- Data that resides in a fixed field within a record or file
- Ex: data in a database table
- Easy to enter, store, and analyse

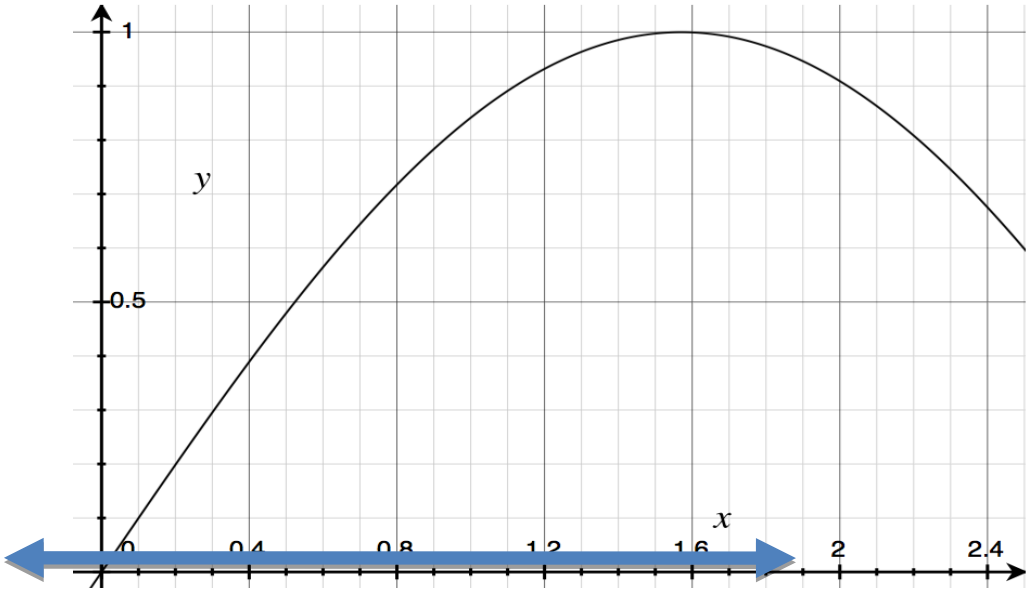
Unstructured Data

- Does not reside in a traditional Database
- Ex: e-mail, videos, audio files, web pages, presentations
- Difficult and costly to analyse

Search Space

A search space **S** is the space of all possible solutions (including bad ones) to a problem

Example: For the problem of finding the maximum value of the function y=f(x), the solution space is the range of all possible values of x:

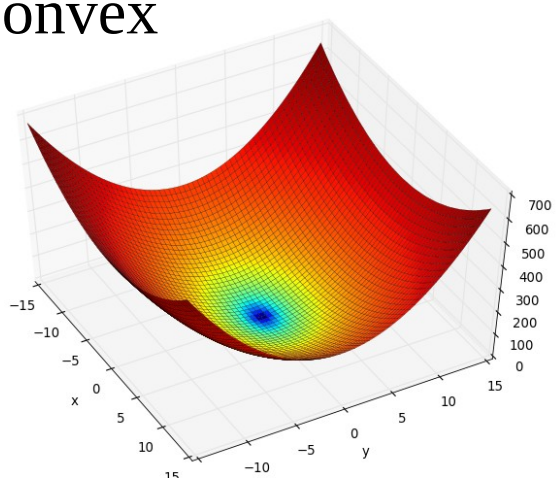


The two most important characteristics of a search space for us are their **size** and **complexity**

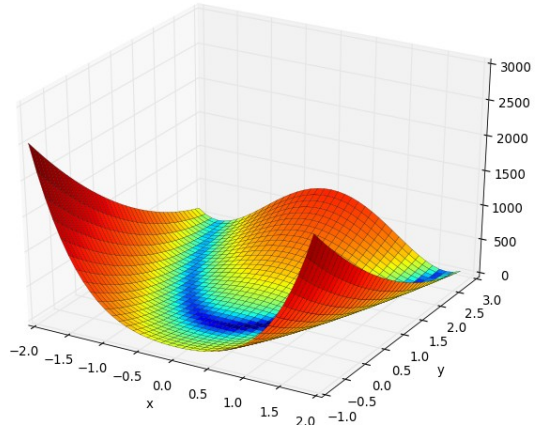
Complex Search Spaces

The solutions of a problem can be arranged with different levels of complexity

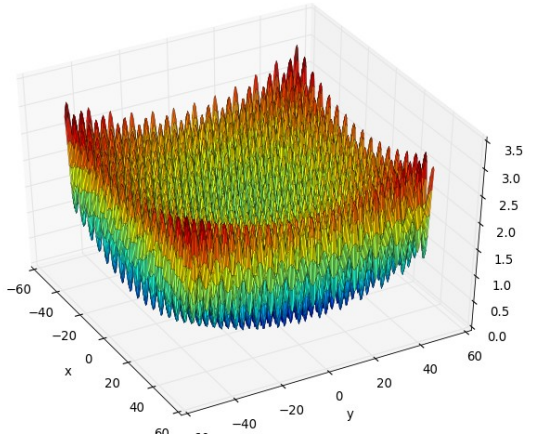
Convex



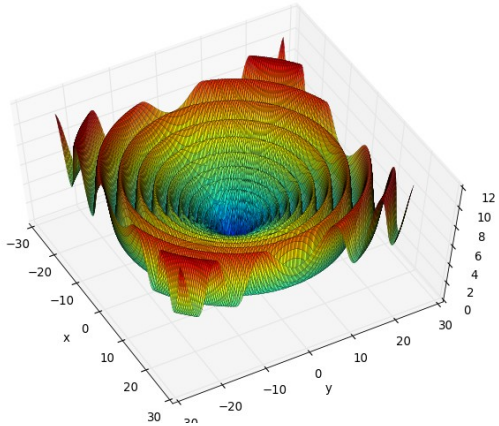
Non Convex



Ruggedness vs Smoothness

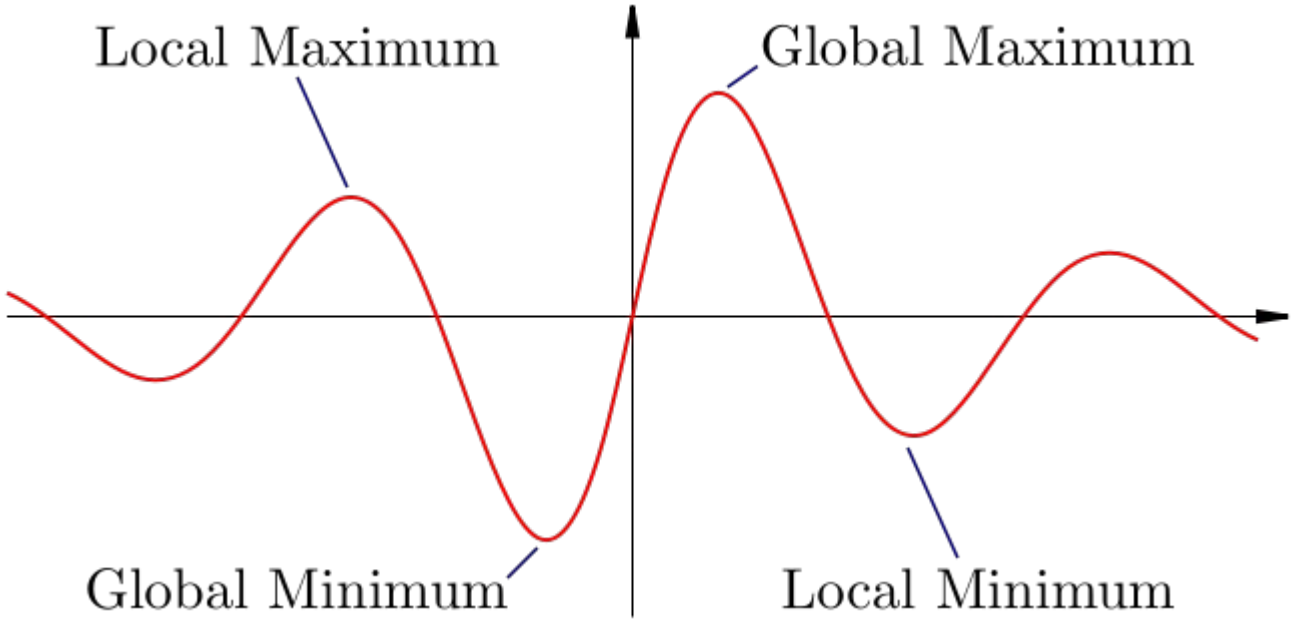


Neutrality



Local/Global Maxima and Minima

If we plot all the values of a function we want to minimise/maximise, we can find some areas of interest. In a one dimensional function:



Global Maximum: largest value of a function f(x) across all valid x. If a value is larger than all their neighbours, then this is a **local maximum.**

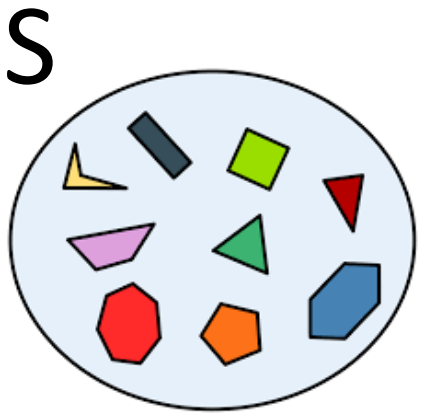
$$f(x) > \left(f(x-h) \wedge f(x+h)\right)$$

Global Minimum/ Local minimum: the same concept but for the smallest values of the function

Size of the Search Space

For easy problems (S is small) → it is easy to find an optimal solution.

We can simply perform an **exhaustive search**.



Exhaustive search: Generate every possible solution, evaluate them, and hence discover which is best.

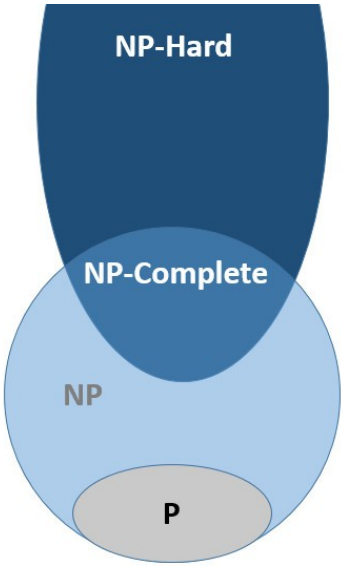
However, in real-world problems the best solution is less obvious. S is too large to search one by one. So we need to find other ways to search through S.

Problem complexity, in the context of computer science, is how hard it is to solve a given problem.

‘too-big’ problems

But, how can we measure what is easy or not?

If **A** is the fastest algorithm known for solving a problem **Q** of size **n** exactly. The complexity of problem Q is the time it takes A to solve it, **as a function of n**.



There are two types of ‘too-big’ problem:

- **easy** (or `tractable’, or `in P’) that can be solved in polynomial time (the dominant term is polynomial). E.g. 34n, nlog.n, sin(2.2n)…

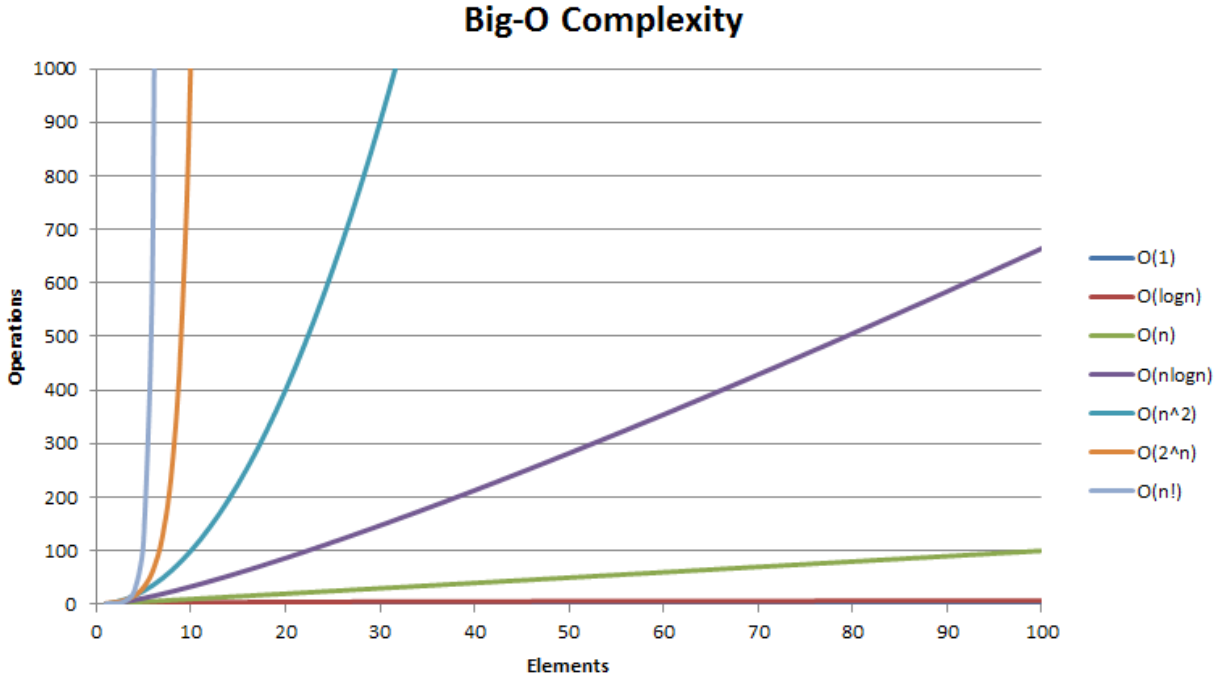
- **hard** (or `intractable’, or `not known to be in P’) The fastest known algorithm is usually not significantly faster than exhaustive search E.g. 1.1n, n(n+2), 2n...

The rigorous mathematical definitions of the these types is outside the scope of this course, but you need to know that it is used the **big O notation**.

Polynomial and Exponential Complexity

An exponential curve always takes over a polynomial one.

E.g. time needed on fastest computers to search all protein structures with 500 amino acids: trillions of times longer than the current age of the universe.



$$\text{As } x \to \infty$$

Factorial	Exponential	Polynomial	Logarithmic
$x!$	e^x	$x\sqrt{1+x^2}$	$\ln(\ln x)$
	$\cosh x$	x^2+1	$(\ln x)^3$
	$3^{\sqrt{x}}$	$\sqrt{x}$	$\ln x$
	2^x		

Big O Notation Summary

Notation	Type	Examples	Description
O(1)	Constant	Hash table access	Remains constant regardless of the size of the data set
O(log n)	Logarithmic	Binary search of a sorted table	Increases by a constant. If n doubles, the time to perform increases by a constant, smaller than n amount
O(< n)	Sublinear	Search using parallel processing	Performs at less than linear and more than logarithmic levels
O(n)	Linear	Finding an item in an unsorted list	Increases in proportion to n. If n doubles, the time to perform doubles
O(n log(n))	n log(n)	Quicksort, Merge Sort	Increases at a multiple of a constant
$O(n^2)$	Quadratic	Bubble sort	Increases in proportion to the product of n*n
$O(c^n)$	Exponential	Travelling salesman problem solved using dynamic programming	Increases based on the exponent n of a constant c
O(n!)	Factorial	Travelling salesman problem solved using brute force	Increases in proportion to the product of all numbers included (e.g., 1*2*3*4…)

Exact & Approximate Algorithms

We have seen how an algoritm can find a solution which is guaranteed to be the best in S. These algorithms are called **exact algorithms**.

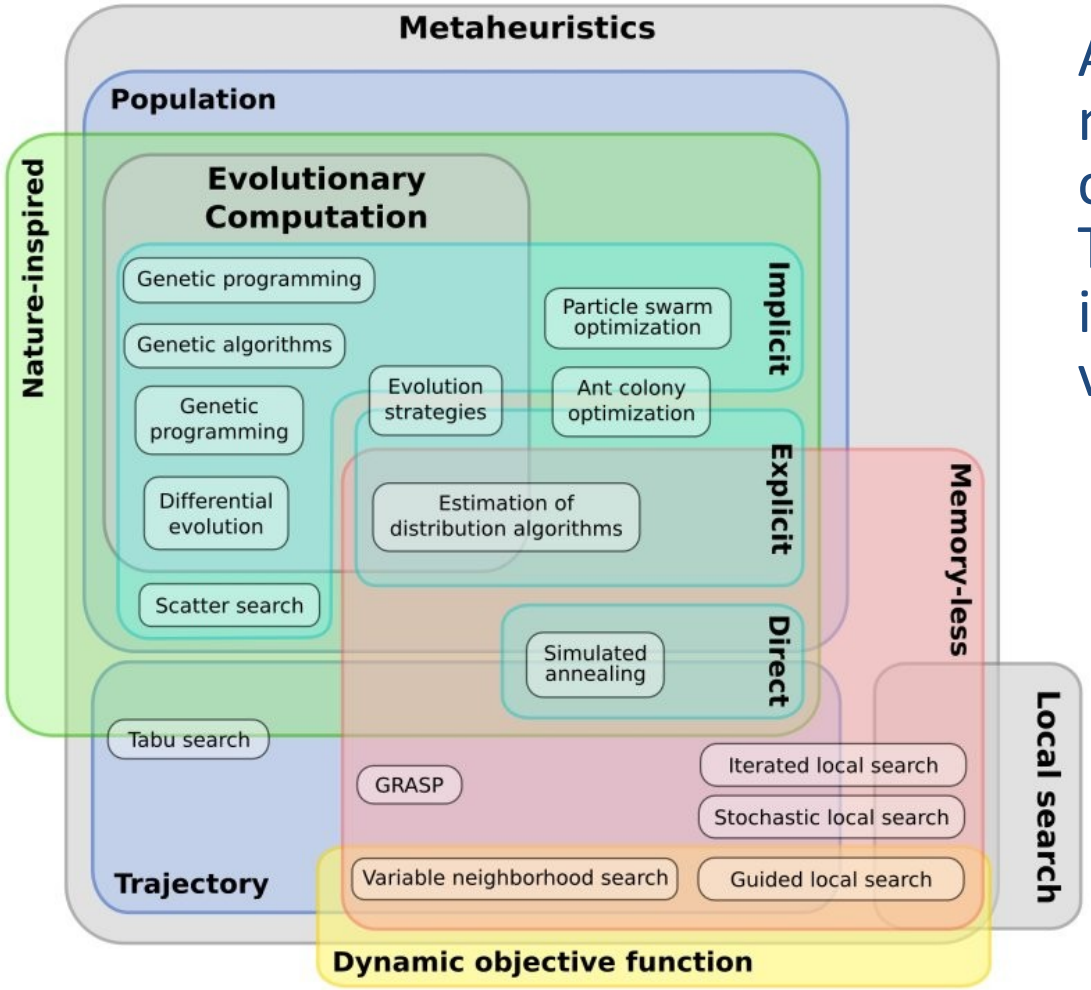
For hard optimisation problems (again, which turns out to be nearly all the important ones), we need **approximate algorithms**.

These algorithms:

- deliver solutions in reasonable time.
- try to find pretty good ('near optimal') solutions, and often get optimal ones.
- cannot guarantee that they have delivered the optimal solution.

Metaheuristics

The learning algorithms seen in this course are **approximative** algorithms that are part of a group called **metaheuristics**.

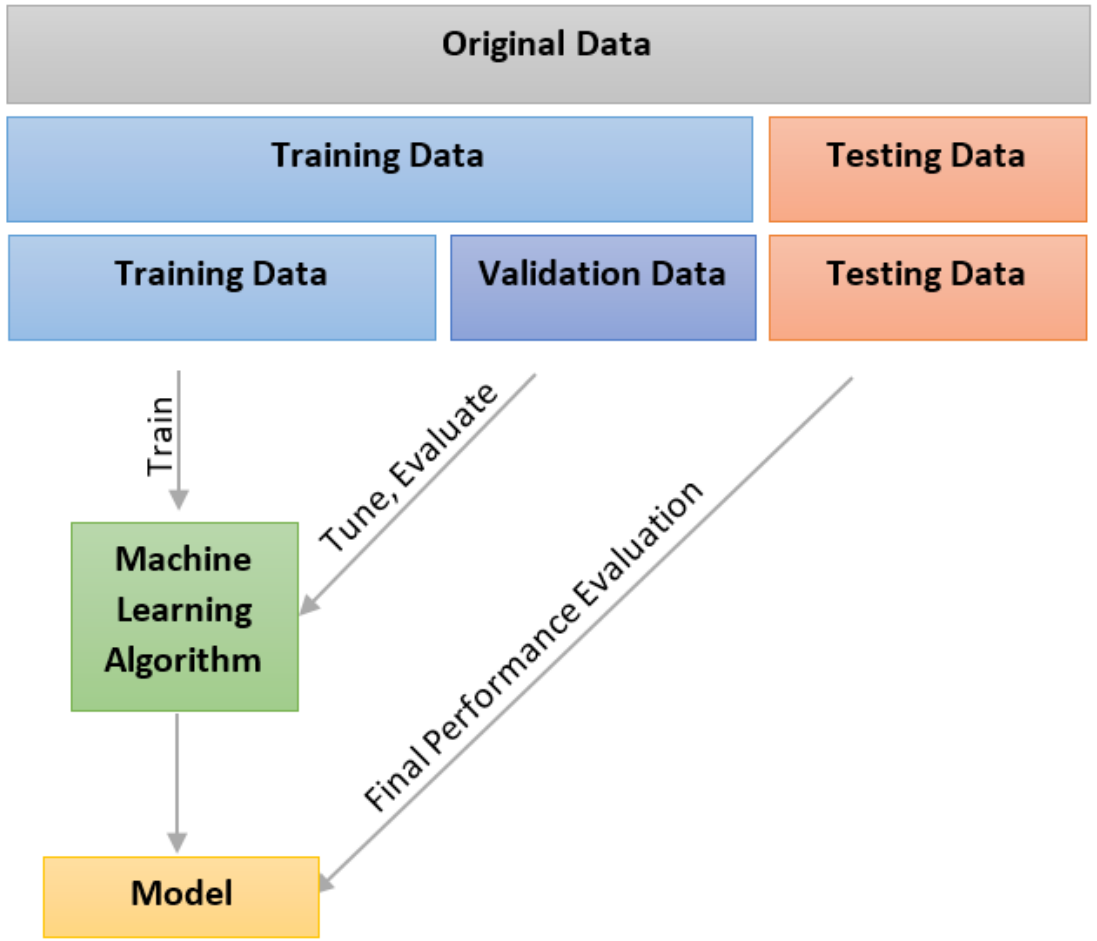


A metaheuristic is a heuristic method for solving a very general class of computational problems. They tend to be problem independent. Metaheuristics are very diverse.

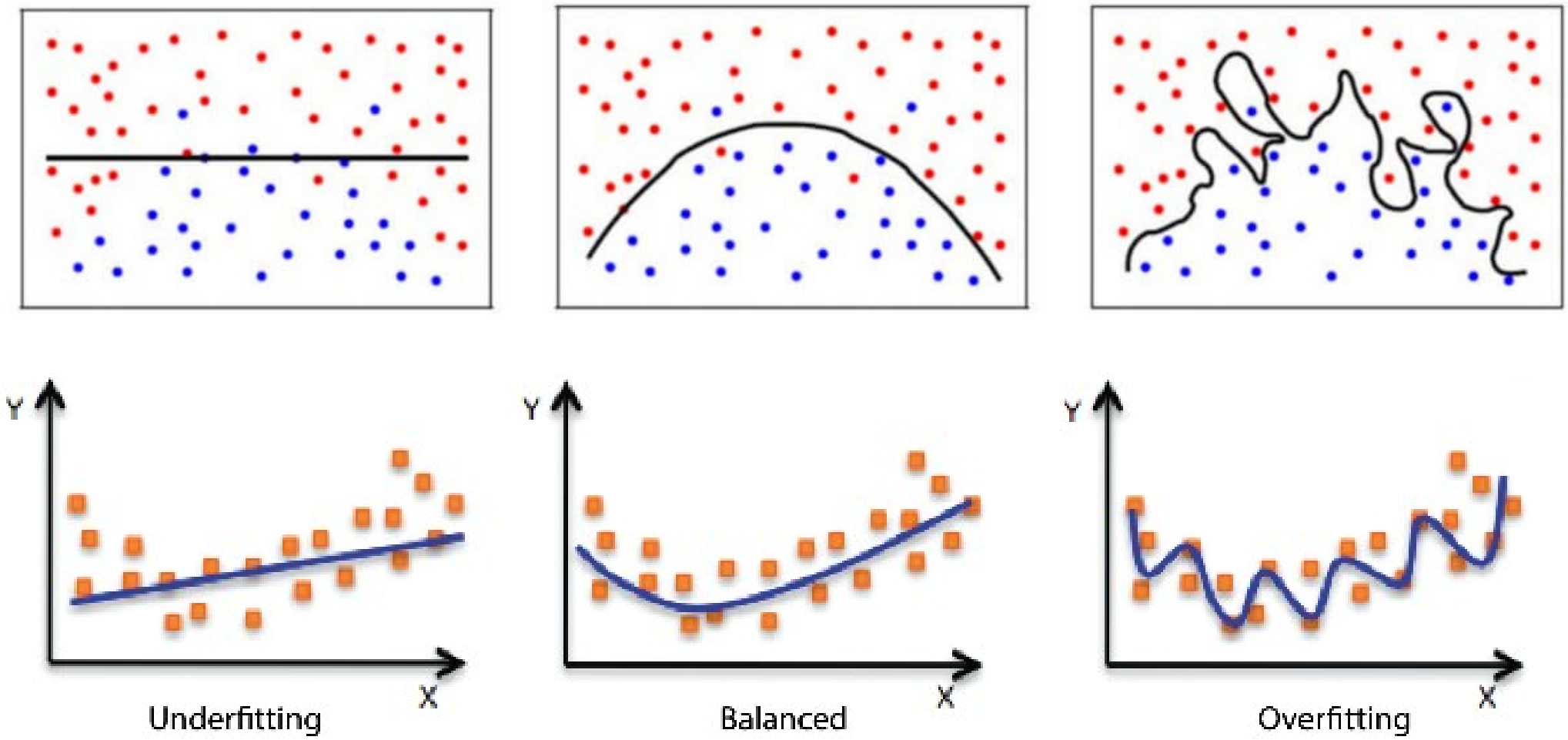
BIC methods are very successful approximate algorithms

Evaluation of our Model

We typically split the input data into learning and testing datasets. Then, we run the machine learning algorithm on the learning dataset to generate the prediction model. Later, we use the test dataset to evaluate our model.

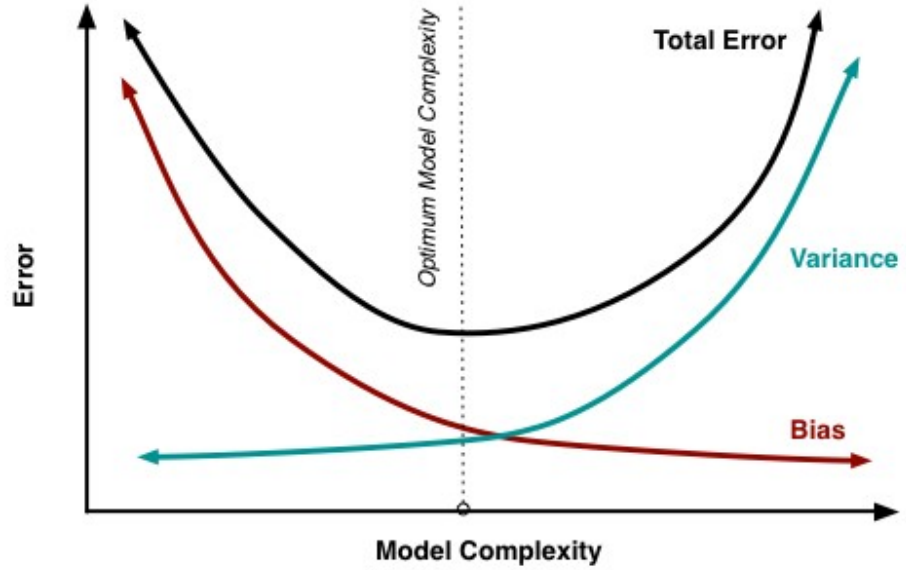


Generalisation: Overfitting / Underfitting

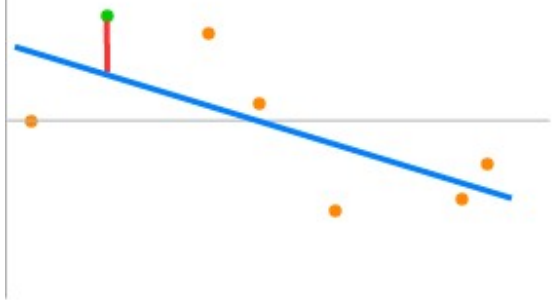


Overfitting may occur when the learned function performs well on the data used for training and poorly with new data. It does not generalised. Most probable when the size of our training data is small

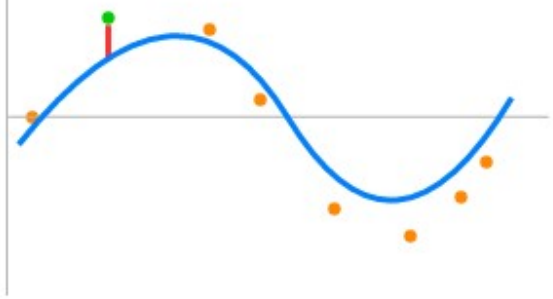
Bias – Variance Trade-off



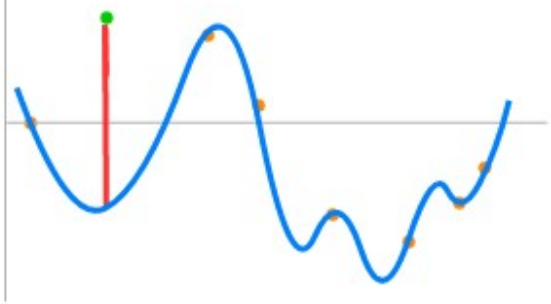
Underfit
High Bias - Low variance



Trade-off fit
Medium Bias - Medium variance



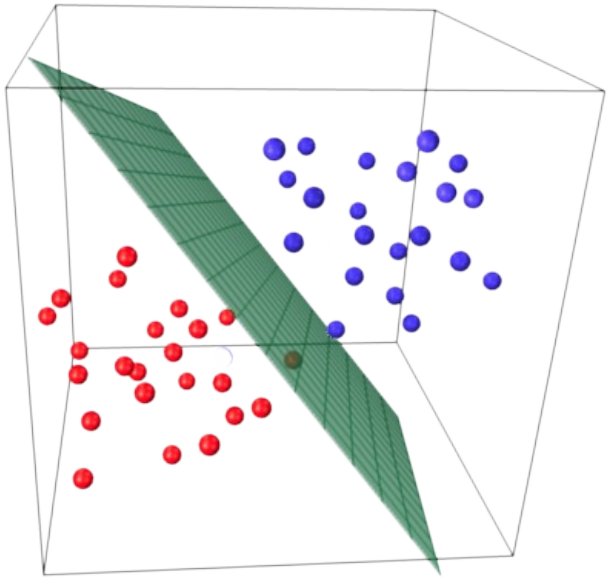
Overfit
Low Bias - High variance



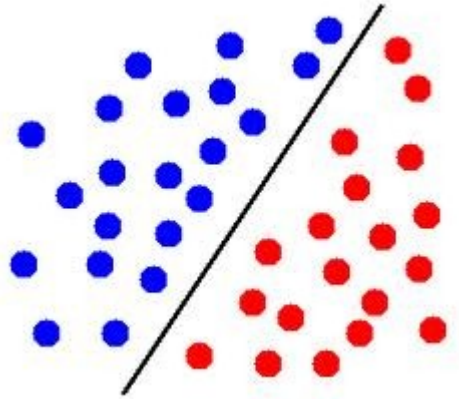
Linear Separability

Two sets of points in p-dimensional space are said to be linearly separable if they can be separated using a p-1 dimensional hyperplane.

3 Dimensions



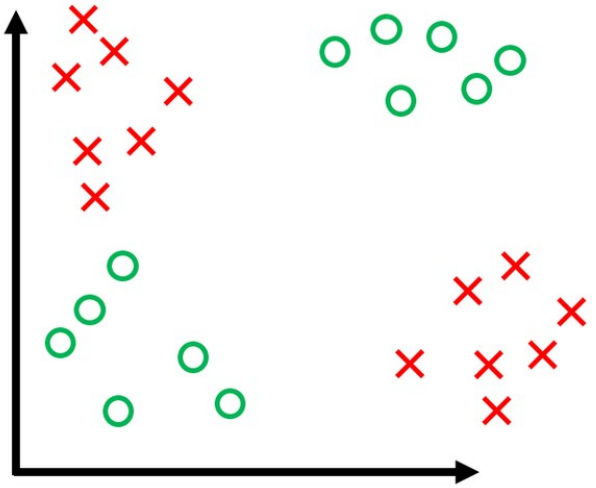
2 Dimensions

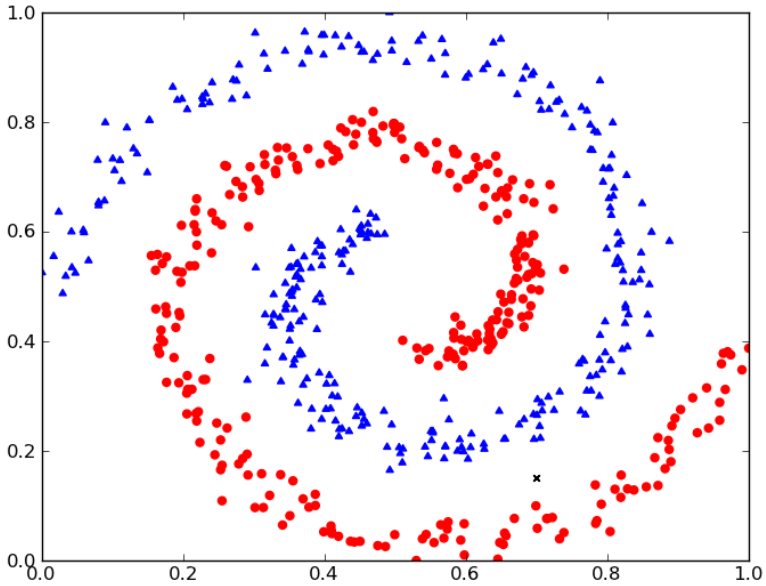


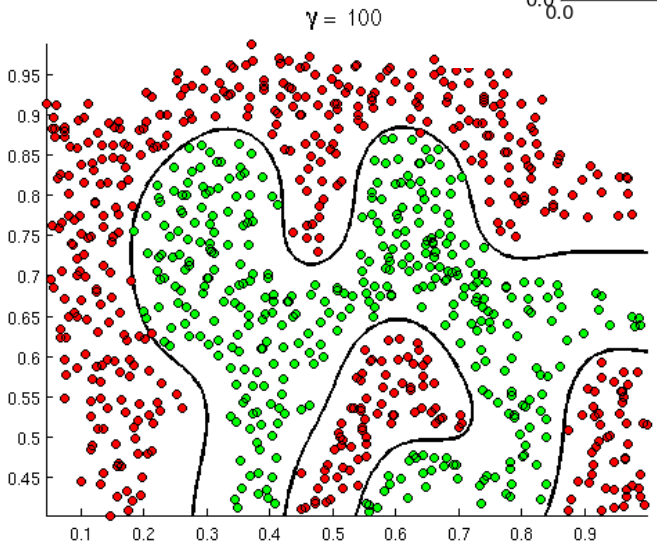
The hyperplane that separates the two sets of data is called the decision boundary (2D) or linear discriminant (+2D).

Linear Separability

Problems that are not linear separable:





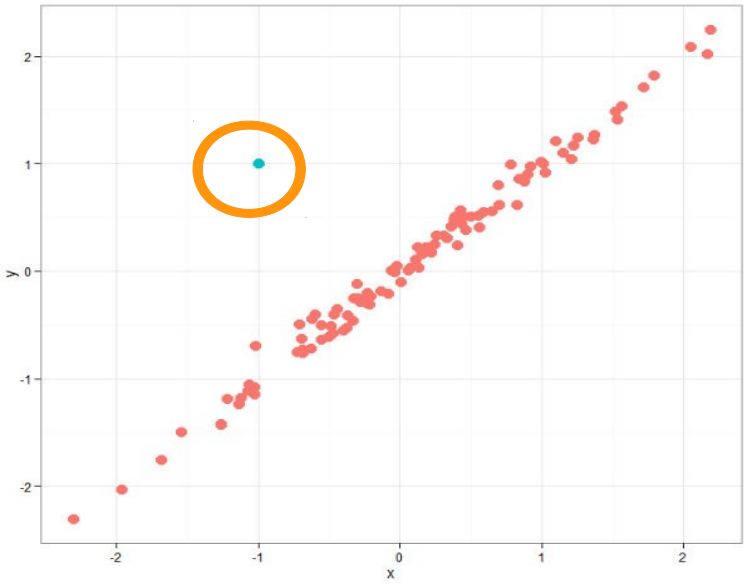


Outliers

An outlier is a data instance that does not comply with the general behaviour of the data

Most data mining methods discard them as noise or exceptions

They can be detected using statistical techniques or using distance measurements, where objects at a certain distance of the cluster are consider outliers



Applications where the study of outliers is essential:

- Credit card fraud detection
- Telecom fraud detection
- Customer segmentation
- Medical analysis