Machine Learning for SUSY Model Building

pre-SUSY 2023: School on Supersymmetry
and Unification of Fundamental forces
July 2023

Miguel Crispim Romão
LIP-Minho, Pheno Group
Southampton HEP
mcromao@lip.pt
miguel@miguelromao.me

LABORATÓRIO DE INSTRUMENTAÇÃO
E FÍSICA EXPERIMENTAL DE PARTÍCULAS

University of Southampton

Disclaimer

I cannot really teach you Machine Learning and go through thorough examples in model building in 2 x 45 min

My hope is to teach you some important take-home concepts and leave you with some pointers on where to go next

Today:

- Introduction to Machine Learning
- Types of Learning
- Some Models
- Machine Learning Workflow

Tomorrow:

- Classifier-guided Parameter Space Search
- Observable Prediction with a Regressor
- Evolutionary Strategy for Exploration

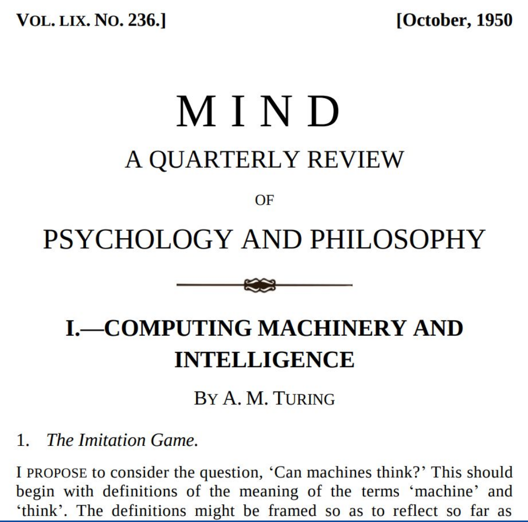
Code & Data

Code: https://gitlab.com/miguel.romao/ml-for-model-building-susy-2023

Data: https://zenodo.org/record/8146636

Lecture 1: Introduction to Machine Learning

“Artificial Intelligence is the quest of creating machines that think and act intelligently


VOL. LIX. NO. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

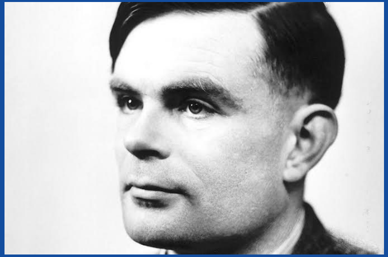
PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

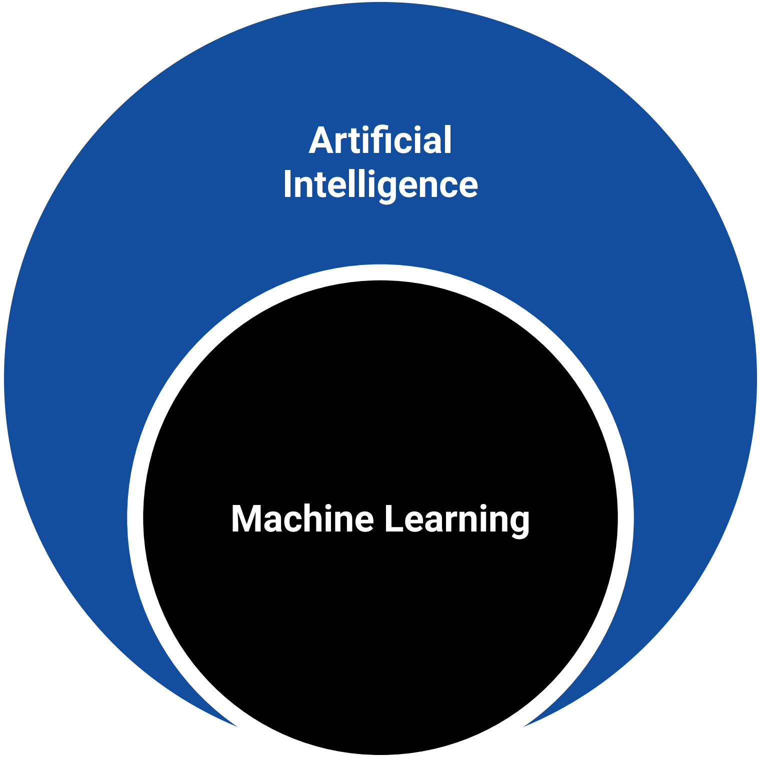
BY A. M. TURING

1. *The Imitation Game.*

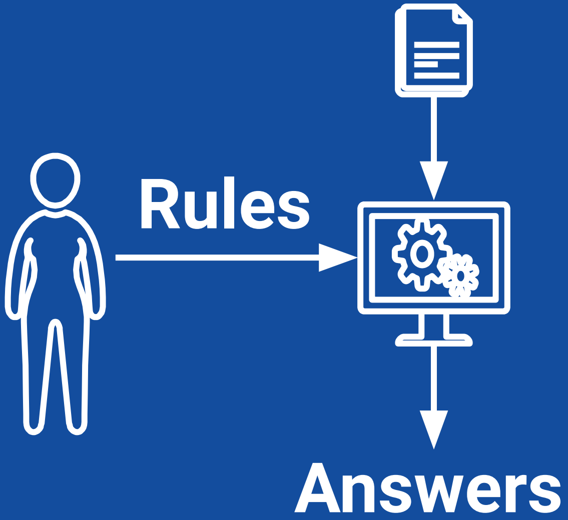
I PROPOSE to consider the question, ‘Can machines think?’ This should begin with definitions of the meaning of the terms ‘machine’ and ‘think’. The definitions might be framed so as to reflect so far as



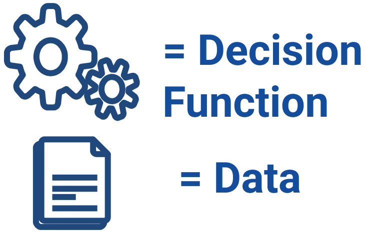
Machine Learning is the subfield of Artificial Intelligence that concerns how a machine learns from experience

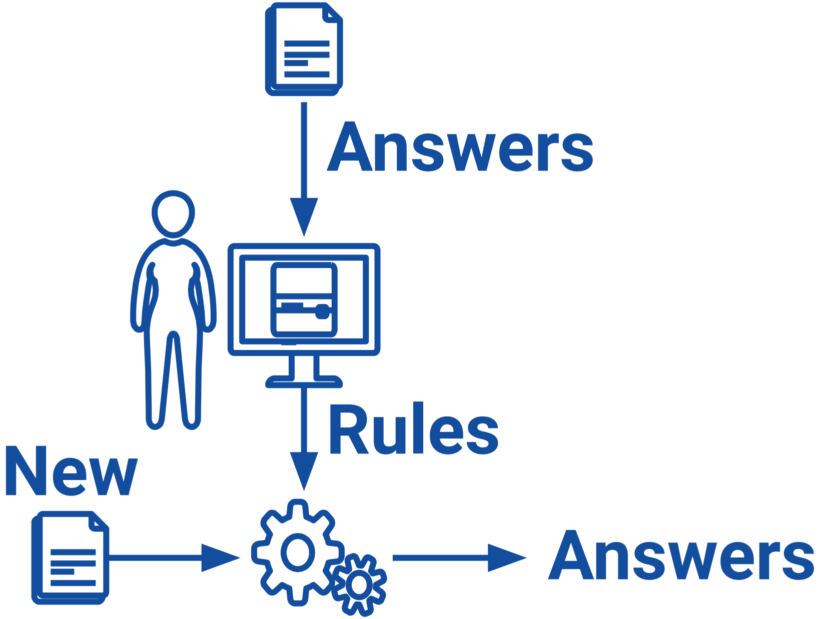


Classical Programming



Machine Learning

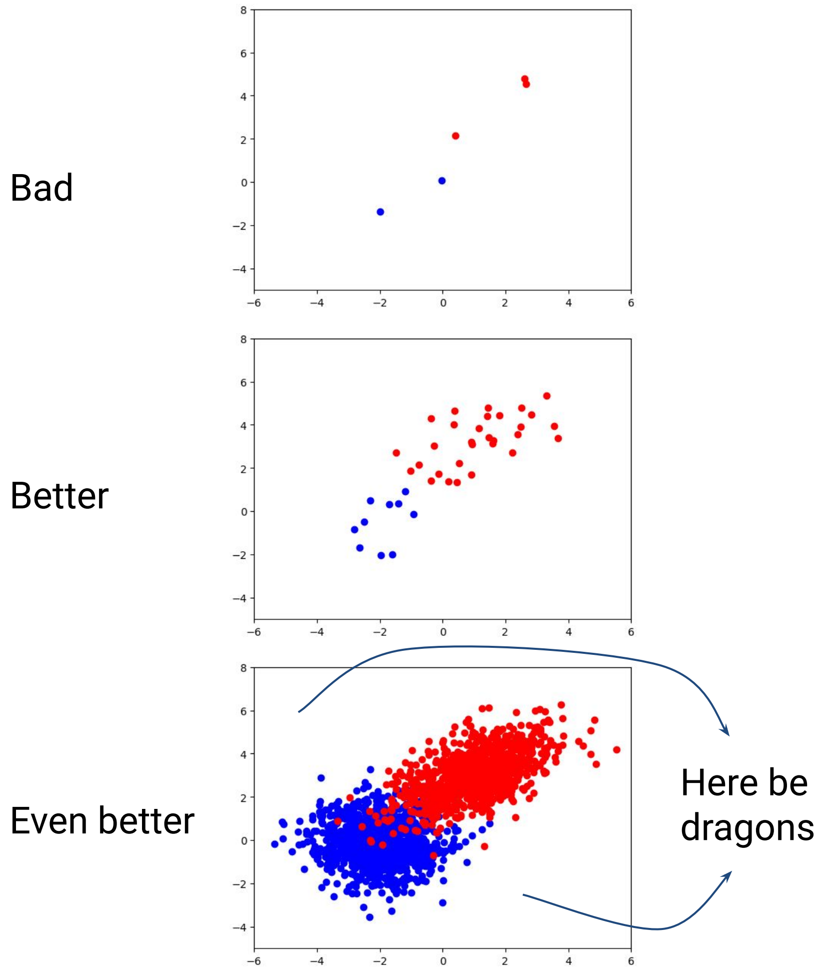




“The current paradigm of learning is that of Statistical Learning: The machine learns functions over the distributions of the data

Learning from Data has Consequences

- Bound to the quality and quantity of the data
- Bound to the (by definition) compact support of the data
- Bigger and more complex models require more and better data
- Machine Learning excels at interpolation, not so much at extrapolation



The Unreasonable Effectiveness of Data - Alon Halevy, Peter Norvig, Fernando Pereira (2009)

Taxonomy

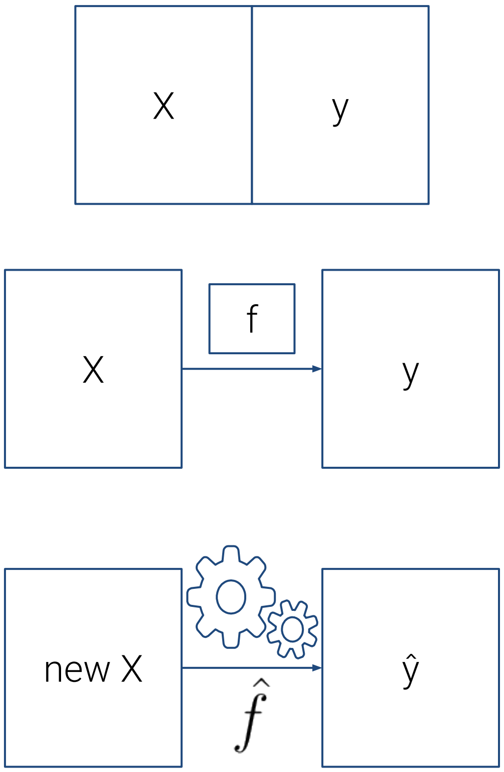
Machine Learning
Taxonomy: Types of Learning

The main differentiator is the type of learning, i.e. by task

- Supervised
  - Data includes the answers
- Unsupervised
  - Algorithm embodies the answers
- Other types
  - Semi-supervised
  - Self-supervised
  - Reinforcement

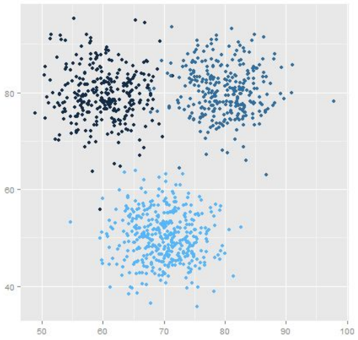
Machine Learning
Taxonomy: Supervised Learning

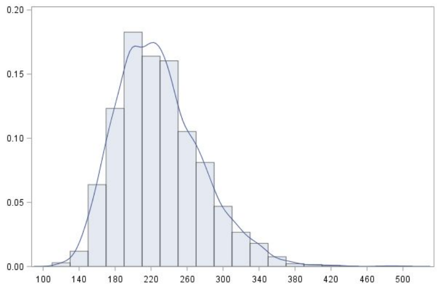
- The training data include the answer we want to reproduce
  $$\mathcal{D} = \{(X_i, y_i)\}$$
  - X: Independent Variables/Features
  - y: Target Variables/Labels
- Assume (hope?) a map exists such that
  $$f : X_i \mapsto y_i$$
- The model will approximate f, $\hat{f}$
- The type of y defines two sub-classes
  - y is a real variable: **Regression**
  - y is categorical: **Classification**

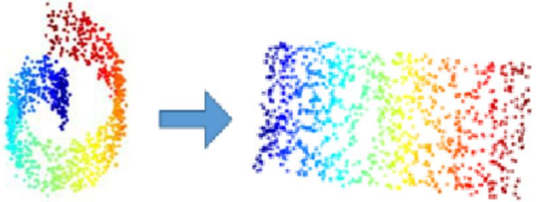


Machine Learning
Taxonomy: Unsupervised Learning

- The training data do not include the answer we want to reproduce
- The answer is embodied in the Learning Algorithm
- The model will learn how to map X to the desired answers
- Answers define the type of model
    - Clustering
    - Density Estimation
    - Dimensional Reduction

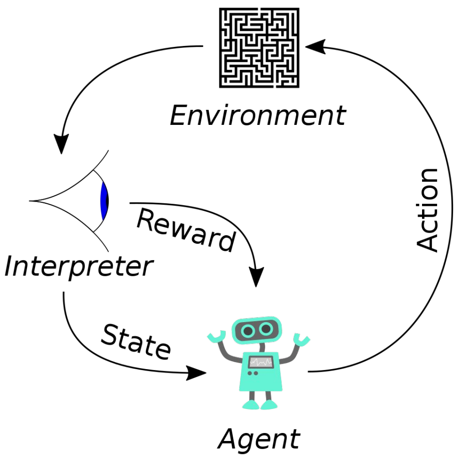






Machine Learning
Taxonomy: Other Types of Learning

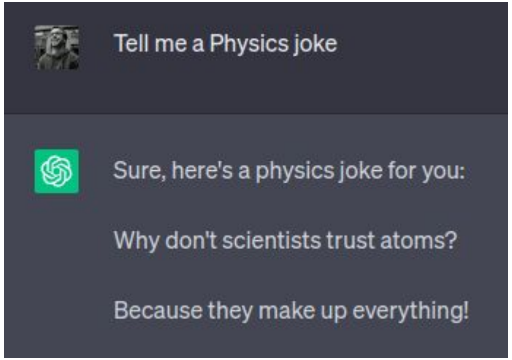
Wikipedia



- Reinforcement learning:
  - An agent interacting with environment
- Self-supervised:
  - Representation learning
  - Generative models

Prompt: An astronaut riding a horse in a photorealistic style

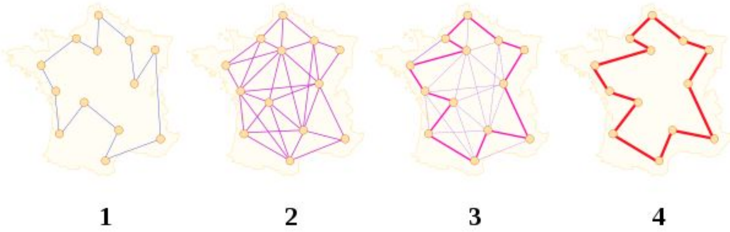
https://stablediffusionweb.com/#demo



Machine Learning
Taxonomy: Other AI Approaches

- Search
  - Travel salesman problem
  - Combinatorics
- Optimisation
  - Bayesian optimisation
  - Genetic and evolutionary algorithms

Wikipedia



https://interestingengineering.com/innovation/evolutionary-algorithms-how-natural-selection-beats-human-design

Simple Parametric Supervised Models

Regression Example
Linear Regression

- Pairs (x_i, y_i) that appear linearly related
- The hypothesis function space is composed of all linear functions

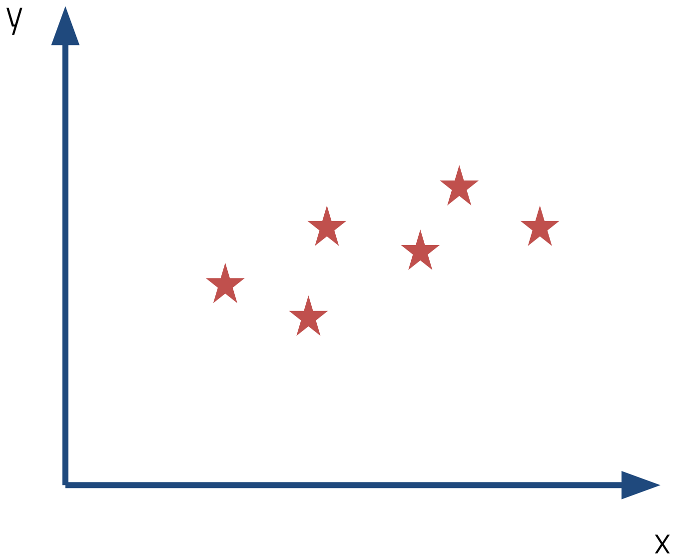
$$\hat{y} = w\,x + b$$

w: weight, b: bias: **learnable parameters**

- Convenient to rewrite as

$$\hat{y} = \mathbf{W} \cdot \mathbf{x}$$

$$\mathbf{W} = [b \quad w],\ \mathbf{x} = \begin{bmatrix} 1 \\ x \end{bmatrix}$$



Exercise: Convince yourself that the generalisation to multivariate linear regression is trivial

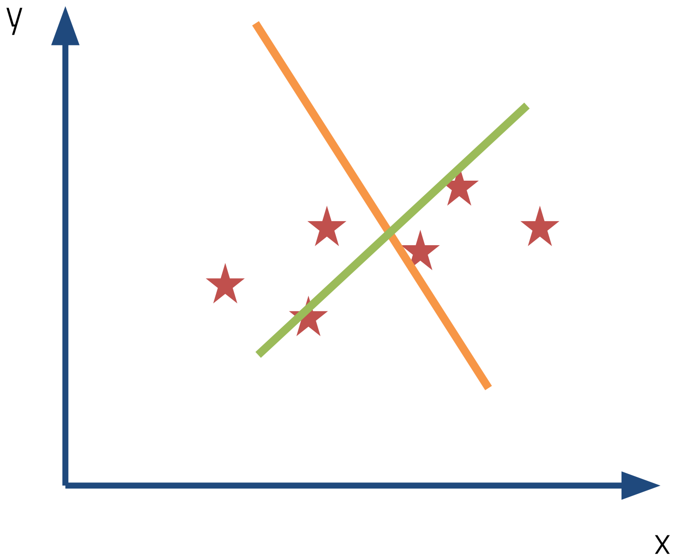
Regression Example

Solving the Linear Regression: Normal Equation

- Intuitively: **Green** line is **better** than **Orange** line
- **Quantify** this using a **loss function**
- For regression problems, we use **Mean Square Error**

$$MSE_i = (y_i - \hat{y}_i)^2 = (y_i - \mathbf{W} \cdot \mathbf{x}_i)^2$$

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \mathbf{W} \cdot \mathbf{x}_i)^2$$



Exercise: Show that MSE_i can be obtained from the minus log-likelihood of $N(\hat{y}_i, \sigma)$ with constant σ

Regression Example

Solving the Linear Regression: Normal Equation

- We have to **minimise** the **loss** with respect to **the model parameters, w**
- It can be shown (exercise) that there is a closed form solution: **Normal Equation**

$$\mathbf{w} = \left(X^T \cdot X\right)^{-1} X^T Y$$

where

$$X = [\mathbf{x}_1 \dots \mathbf{x}_N]^T$$
$$Y = [y_1 \dots y_N]^T$$

- But this has many issues:
  - The $X^T.X$ matrix needs to be invertible
  - Computationally prohibitive for large datasets
  - It does not exist for more complicated (non-linear) models

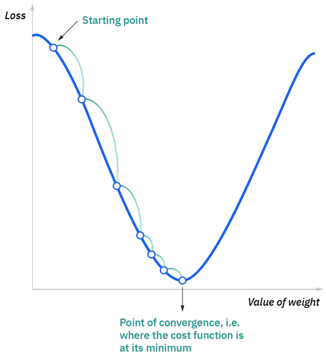
Regression Example

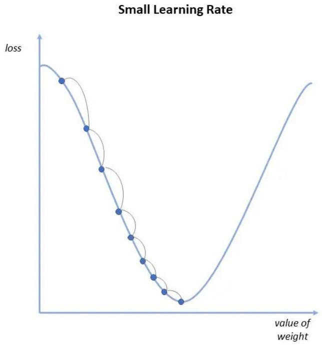
Solving the Linear Regression: Gradient Descent

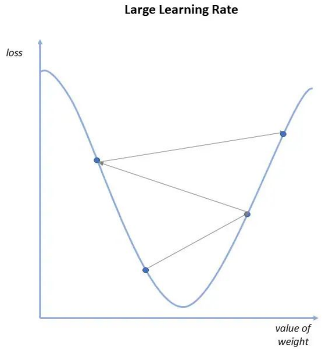
- **Iterative first order** method that uses the **gradient** of the loss to **minimise** it

$$\mathbf{W}^{\mathbf{t+1}} = \mathbf{W}^{\mathbf{t}} - \eta \nabla_{\mathbf{W}^{\mathbf{t}}} \mathbf{Loss}$$

where η is called the **learning rate**







https://www.kaggle.com/code/bhatnagardaksh/gradient-descent-from-scratch

Let's go to the first notebook of examples!

Classification Example
Logistic Regression

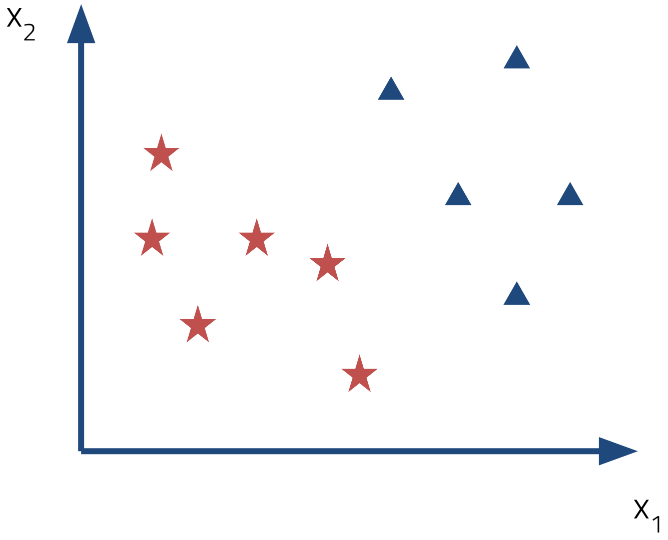
- Pairs ($\mathbf{x}_i$, y_i) where y_i = **star** or **triangle**
- We will define one class as 1 and the other as 0, e.g.

$$y_i = \begin{cases} 0 & \text{, if Star} \\ 1 & \text{, if Triangle} \end{cases}$$

with probability

$$\Pr(y_i = 1|p) = p$$
$$\Pr(y_i = 0|p) = (1-p)$$

- Naively: Bernoulli trial. 5 triangles, 6 stars. Chance, p, of being triangle is 5/11
- But it is clear that $p=p(\mathbf{x}_i)$!



Classification Example
Logistic Regression

- We want to find a map, **$\mathbf{p(x_i)}$**, from **$\mathbf{x_i}$** to [0,1] that **maximises** the (Bernoulli) **likelihood**

$$\Pr(y_i \,|\, p(\mathbf{x}_i)) = p(\mathbf{x}_i)^{y_i}(1 - p(\mathbf{x}_i))^{1-y_i}$$

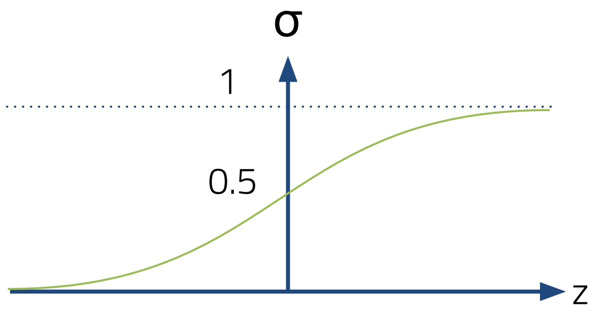
or, conversely, that minimises the negative log-likelihood

$$BCE_i = -y_i \log\left(p(\mathbf{x}_i)\right) - (1 - y_i) \log\left(1 - p(\mathbf{x}_i)\right)$$

- This is known as **binary cross-entropy**, and it is the **loss for binary classification**

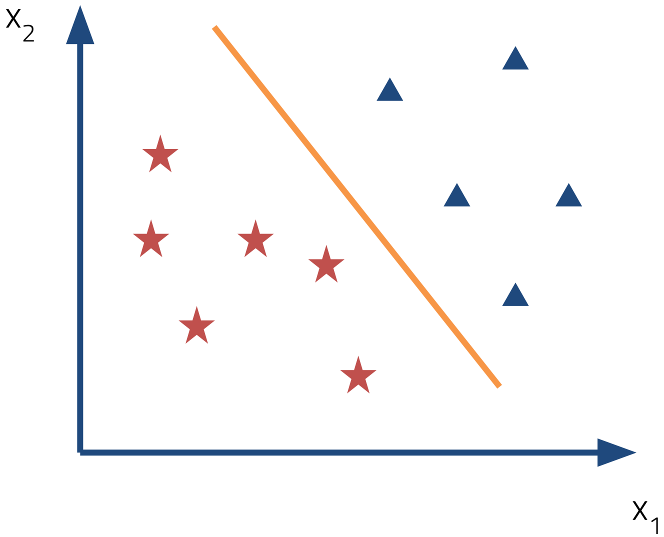
$$BCE_i = -y_i \log\left(p(\mathbf{x}_i)\right) - (1 - y_i) \log\left(1 - p(\mathbf{x}_i)\right)$$

Classification Example
Solving Logistic Regression



- Following the Linear Regression example, we want to **learn a function**
- We assume a **linear learner**

$$\sigma(z) = \frac{1}{1+e^{-z}}$$
$$z = \mathbf{W} \cdot \mathbf{x}$$
$$p(\mathbf{x}) = p(\mathbf{x}, \mathbf{W}) = \sigma(z)$$
$$\mathbf{W} = [b \quad w_1 \quad w_2], \mathbf{x} = [1 \quad x_1 \quad x_2]^T$$



- Where the **surface z=0** is known as **decision boundary**

Classification Example
Solving Logistic Regression

- Logistic Regression has no closed form solution => Gradient Descent

$$\mathbf{W^{t+1}} = \mathbf{W^t} - \eta \nabla_{\mathbf{W^t}} \mathbf{Loss}$$

with

$$\text{Loss} = \text{BCE} = -\frac{1}{N} \sum_{i=1}^{N} y_i \log(p(\mathbf{x}_i, \mathbf{W})) + (1 - y_i) \log(1 - p(\mathbf{x}_i, \mathbf{W}))$$

$$p(\mathbf{x}_i, \mathbf{W}) = \sigma(\mathbf{x}_i \cdot \mathbf{W}) = \frac{1}{1 + \exp(-\mathbf{W} \cdot \mathbf{x}_i)}$$

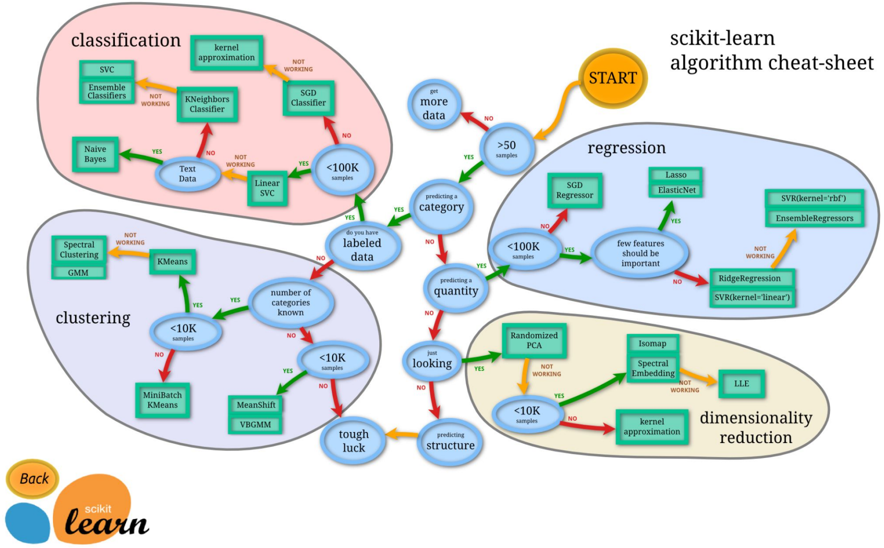
Exercise: Compute $\nabla_{\mathbf{W}} \text{Loss}$

Let's go back to the first notebook of examples!

The Scikit-Learn Package

Machine Learning
Scikit-Learn

- **Scikit-Learn** (scikit-learn.org) is the go-to ML package for python
- Has defined the best practices for ML API development
- Has **great documentation** and **tutorials**
- You can learn ML from Scikit-Learn documentation!



Machine Learning
Scikit-Learn

- We will start by implementing Linear and Logistic regressions
  - sklearn.linear.LinearRegression
  - sklearn.linear.LogisticRegression
- Not estimator modules worth remembering:
  - sklearn.preprocessing
  - sklearn.model_selection
  - sklearn.metrics

Let’s go back to the first notebook of examples!

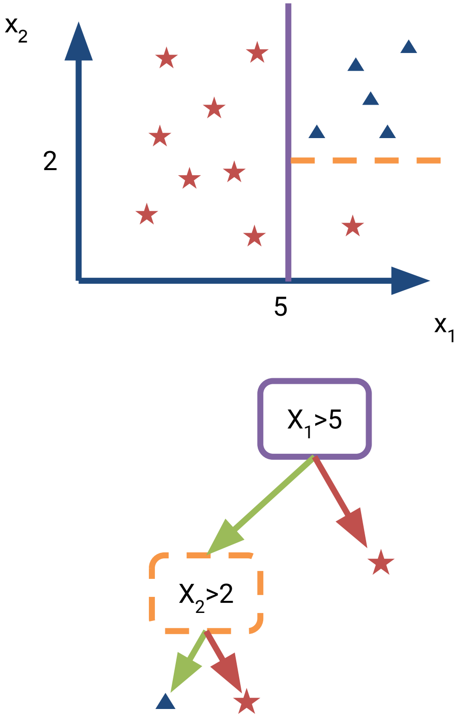
Trees and Ensembles

Decision Tree
Classification Example

- **Split** the data with **recursive** partitions with **half-spaces**
- **Isolate** each class in an **end node**, a **leaf**
- Quantify class isolation using **Gini impurity** or **Entropy**

$$Gini(D) = p_D(1 - p_D)$$
$$H(D) = -p_D \log p_D - (1 - p_D) \log (1 - p_D)$$
$$p_D = \frac{\#class_0 \in D}{\#class_0 \in D + \#class_1 \in D}$$

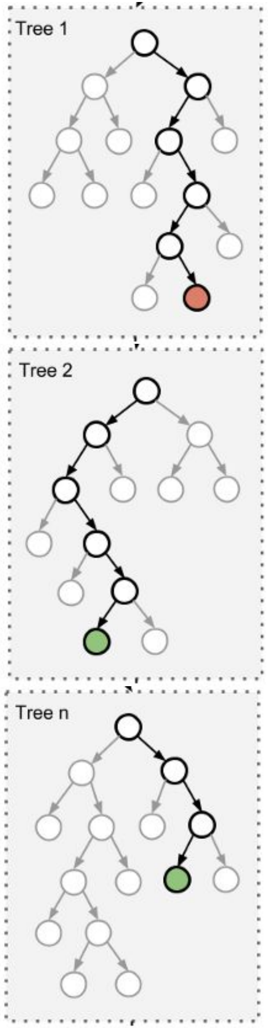
- Repeat until no more splits can be made



Ensembles
Strength in Numbers

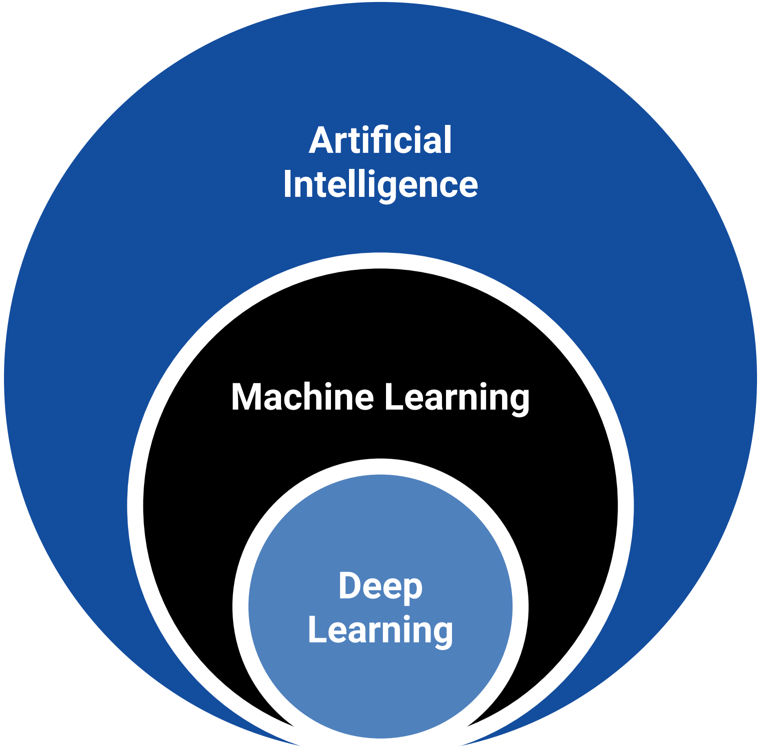
- Trees can **memorise** the training set
  - Bad for generalisation to new data
  - Computationally forbidding for large datasets
- Ensembles are a way of **combining many small trees**
- The idea: **many weaker learners** perform better together, producing **a stronger learner**
- Example: Random Forest is a collection of **smaller trees** (with a maximum depth) **trained on subsamples** of the data
  - The final prediction: average of the predictions

Let's go back to the first notebook of examples!



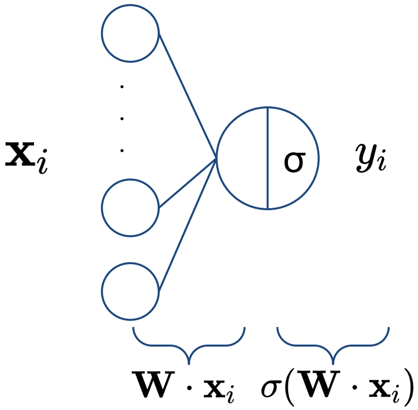
A Taste of Deep Learning

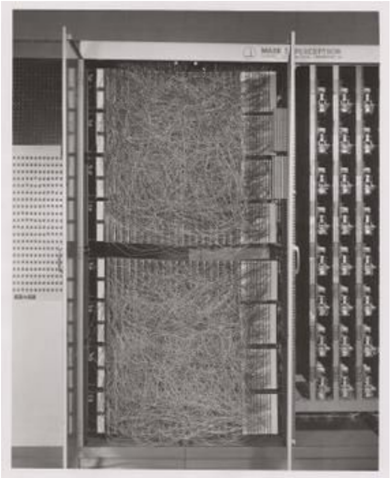
Deep Learning is a subclass of Machine Learning algorithms that train Neural Networks to perform tasks



Deep Learning and Neural Networks
Terrible name, great idea

- Notice that we can represent a Logistic (or linear) Regression diagrammatically





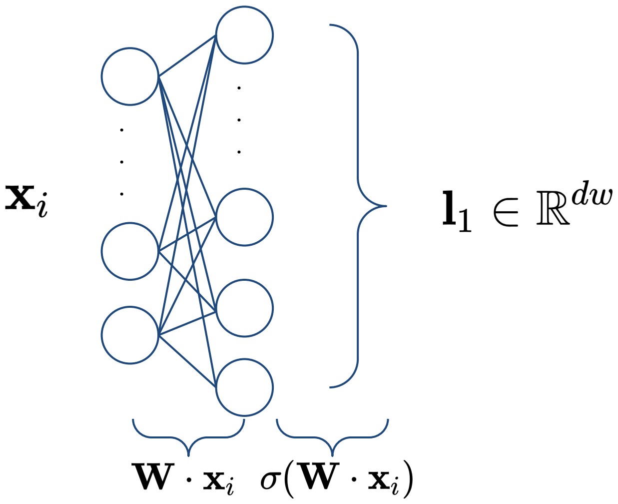
wikipedia

This has a historical name: **perceptron** and it is the first “neural network”

Deep Learning and Neural Networks

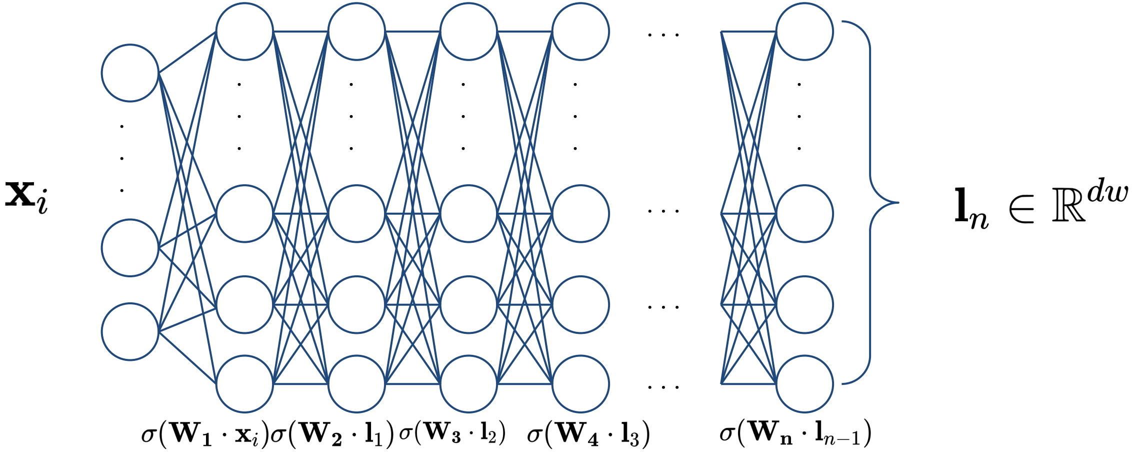
Terrible name, great idea

- By allowing $\mathbf{W} \in \mathbb{R}^{dw \times dx},\ dw > 1$
  we can extend this to multiple outputs, e.g. multi-label classification



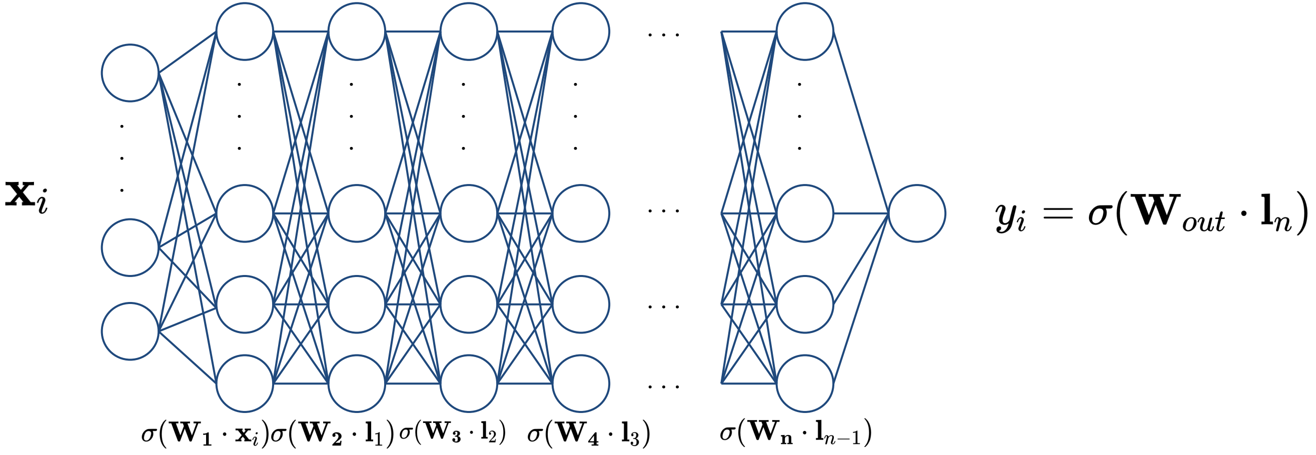
Deep Learning and Neural Networks
Terrible name, great idea

- What if I continue to chain **n layers** with $\mathbf{W}_i \in \mathbb{R}^{dl_{i-1} \times dl_i}$ where dl_i are the **number of neurons in layer i**



Deep Learning and Neural Networks
Terrible name, great idea

- And finally collapse on a single output
- The output is a non-trivial **highly non-linear function** of the inputs



This neural network is called **multi-layer perceptron** or **dense neural network**

Deep Learning and Neural Networks
Terrible name, great idea

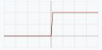
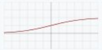
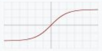
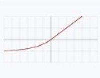
- **Differentiable models**
  - Can be trained with (Stochastic) **Gradient Descent**
- **Highly compositional functions**
  - $\hat{y}_i = NN(\mathbf{x}_i) = Out \circ l_n \circ l_{n-1} \circ \cdots \circ l_1(\mathbf{x}_i)$
- **Universal Function Approximators**
- Have **unmatched representational power** and are capable of **feature abstraction**
  - Each layer can be seen as a **data transformation** step
- Extremely versatile and can take in **data of many different shapes and formats**

Exercise: Show that if we do not use a non-linear function between layers, that NN is only performing a single affine transformation

Deep Learning and Neural Networks
Defining and Training

- Define **how many layers** and **their size** (number of neurons)
- Choose a **non-linear activation** for the **hidden layers**
- Output and loss defined by task
    - Classification: sigmoid and binary cross-entropy
    - Regression: identity function and mean square error
- Iteratively train on mini-batches of data using (Stochastic) **Gradient Descent**

Let's go back to the first notebook of examples!

Name	Plot	Function, $g(x)$	Derivative of g, $g'(x)$
Identity		x	1
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	0
Logistic, sigmoid, or soft step		$\sigma(x) \doteq \frac{1}{1+e^{-x}}$	$g(x)(1-g(x))$
Hyperbolic tangent (tanh)		$\tanh(x) \doteq \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - g(x)^2$
Rectified linear unit (ReLU)[8]		$(x)^+ \doteq \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} = \max(0, x) = x\mathbf{1}_{x>0}$	$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$
Gaussian Error Linear Unit (GELU)[5]		$\frac{1}{2}x\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right) = x\Phi(x)$	$\Phi(x) + x\phi(x)$
Softplus[9]		$\ln(1+e^x)$	$\frac{1}{1+e^{-x}}$
Exponential linear unit (ELU)[10]		$\begin{cases} \alpha\left(e^x - 1\right) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α	$\begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ 1 & \text{if } x = 0 \text{ and } \alpha = 1 \end{cases}$
Scaled exponential linear unit (SELU)[11]		$\lambda\begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameters $\lambda = 1.0507$ and $\alpha = 1.67326$	$\lambda\begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$

wikipedia

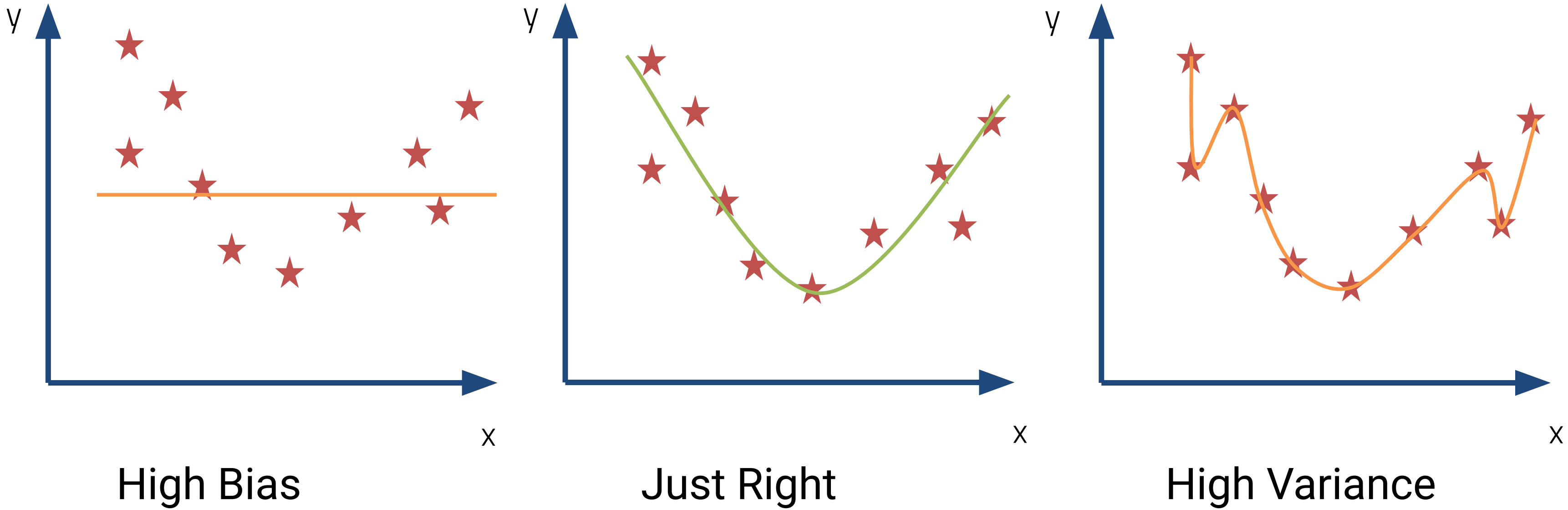
Machine Learning Workflow

Machine Learning Workflow
Choosing a Model

- We have seen many models
  - Which one is better?
  - How do we define better?
  - **“No free lunch” theorem**
- Some models are more complex than others
  - Many **hyperparameters** to choose
    - E.g. Number of estimators in a forest, number of layers in a neural network, etc
  - Complex models have **high capacity** to **memorise the training data** and **perform badly on new data**
- What are the **principled steps** to choose a machine learning model?

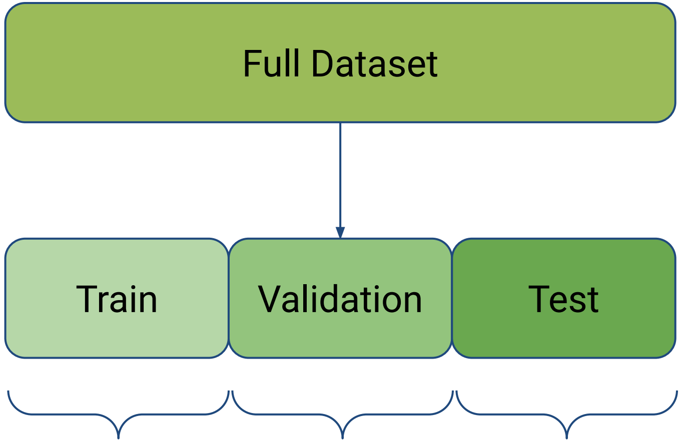
Machine Learning Workflow
Choosing a Model: the Bias-Variance Trade-Off

A model with insufficient capacity will fail to fit: **underfitting**
A model with too much capacity will fit the noise: **overfitting.**



Machine Learning Workflow
Choosing a Model: Data Split

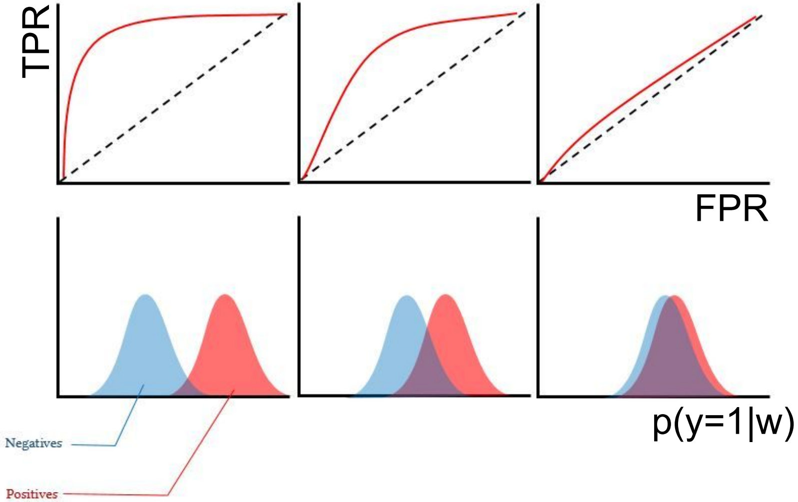
- Split the dataset into three sets
  - Train: for **fitting** the model
  - Validation: for **model selection**
  - Test: to assess the **final performance**
- **Never use the Test set at any stage of your training or validation => Information Leakage** (a.k.a. cheating)



The relative proportions vary across the literature and application. We'll work with 0.6-0.2-0.2

Machine Learning Workflow
Choosing a Model: Classification Metric

- There are many metrics in the Machine Learning literature that help you **assess the performance of a classifier**
- We will focus on the **Area under ROC** (Receiver operator characteristic) curve
  - Values between 0 and 1
  - Easy to implement and intuitive
  - Sample-wide statistics



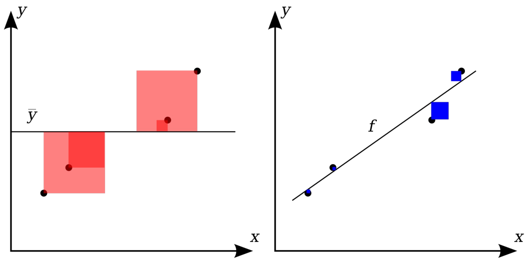
$$TPR = \frac{TP}{TP + FN}$$ True Positive Rate

$$FPR = \frac{FP}{FP + TN}$$ False Positive Rate

Machine Learning Workflow
Choosing a Model: Regression Metric

- Likewise, there are many metrics that let you **assess the performance of a regressor**
- A common one is the **Coefficient of Determination**
    - Normalised Error, i.e. usually between 0 and 1
    - Easy to implement and intuitive
    - Sample-wide statistics

wikipedia



$$R^2 = 1 - \frac{SS_{res}}{SS_{total}}$$

$$SS_{res} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

$$SS_{total} = \sum_{i=1}^{N} (y_i - \bar{y})^2$$

Machine Learning Workflow
Steps for Success

- Define the **task**
- Get **plenty** of **good data**
- **Split** into three datasets
  - **Train, Validation, Test**
- **Choose a model**
  - Start with a **simple baseline**
  - **Upgrade** to a more **complex** model
  - **Tune hyperparameters**
- Assess the **final performance**
- Deploy to production to perform the task on **new data**

Let's go back to the first notebook of examples!

Lecture 2: Machine Learning for (not only) SUSY Model Building

Machine Learning in HEP

Machine Learning in HEP
A flourishing area of research

https://iml-wg.github.io/HEPML-LivingReview/

HEPML-LivingReview

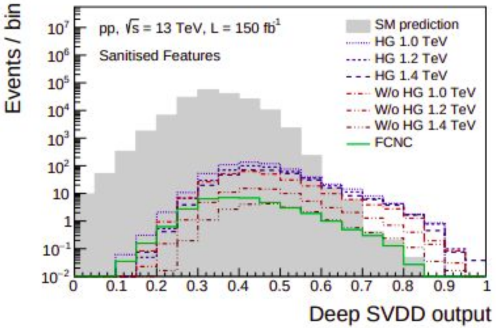
A Living Review of Machine Learning for Particle Physics

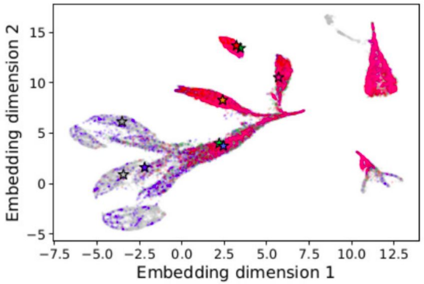
Modern machine learning techniques, including deep learning, is rapidly being applied, adapted, and developed for high energy physics. The goal of this document is to provide a nearly comprehensive list of citations for those developing and applying these approaches to experimental, phenomenological, or theoretical analyses. As a living document, it will be updated as often as possible to incorporate the latest developments. A list of proper (unchanging) reviews can be found within. Papers are grouped into a small set of topics to be as useful as possible. Suggestions are most welcome.

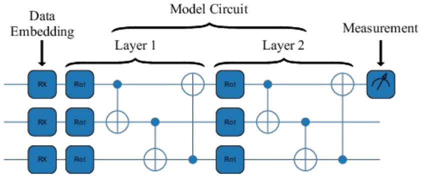
Impossible to cover everything here…

Machine Learning in HEP
Shameless self-promotion

- Generic Searches for New Physics
  - Transferability of Deep Learning Models in Searches for New Physics at Colliders, Phys. Rev. D 101, 035042 (2020), 1912.04220
  - Finding New Physics without learning about it: Anomaly Detection as a tool for Searches at Colliders, Eur.Phys.J.C 81 (2021), 2006.05432
- Grouping Events Together
  - Use of a Generalized Energy Mover's Distance in the Search for Rare Phenomena at Colliders, Eur. Phys. J. C 81, 192 (2021), 2004.09360
- Jet Quenching by the Quark Gluon Plasma
  - Deep Learning for the classification of quenched jets, JHEP 11 (2021) 219, 2106.08869
  - Jet substructure observables for jet quenching in Quark Gluon Plasma: a Machine Learning driven analysis, 2304.07196
- Quantum Machine Learning in HEP
  - Fitting a Collider in a Quantum Computer: Tackling the Challenges of Quantum Machine Learning for Big Datasets, 2211.03233



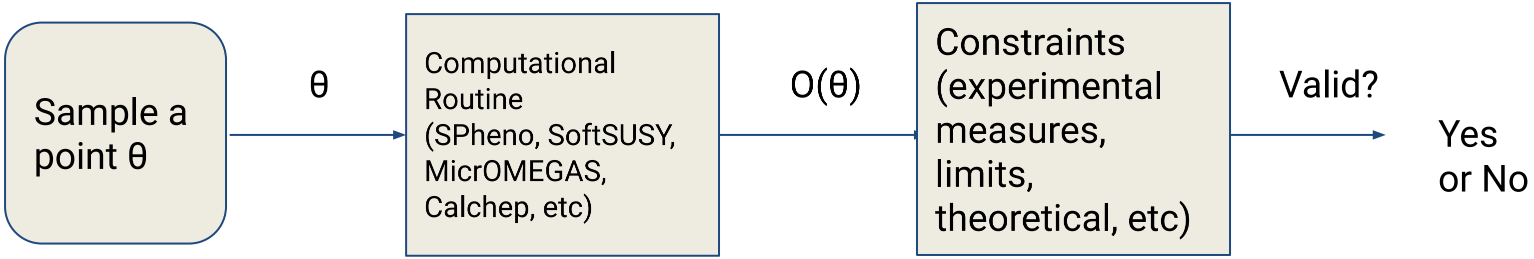




Supervised Learning for Parameter Space Scans

Machine Learning in SUSY Model Building
Applications to Parameter Space Scans

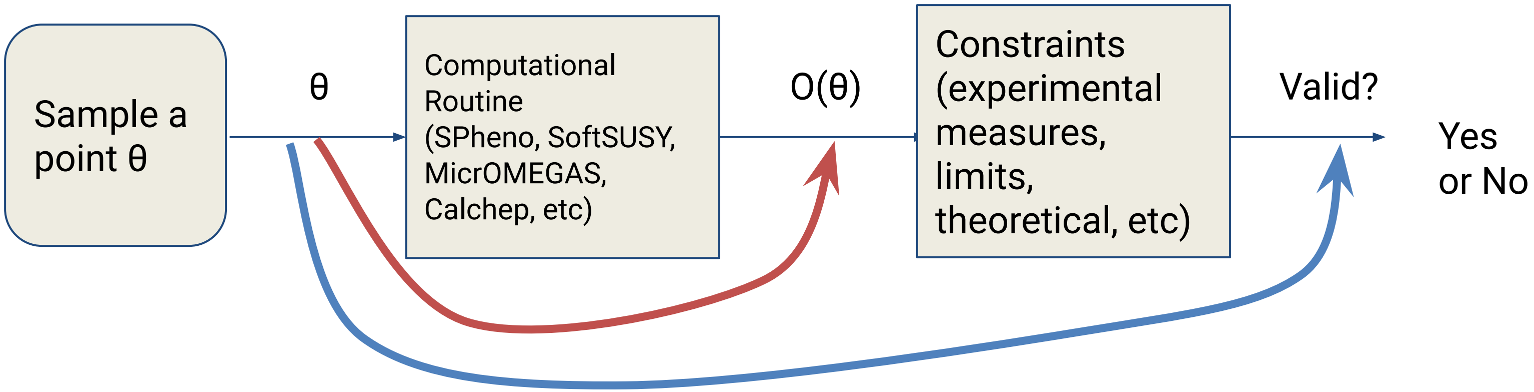
- Parameter space scanning is usually **computationally-** and **time-consuming**
- Difficulty increases for highly constrained cases: **low parameter sampling efficiency**



For more difficult scans one usually adapts for simplicity

Machine Learning in SUSY Model Building
Supervised Learning for Parameter Space Scans

- Considering that the **observable computation** is the heavy step, try to **replace it**, either by **predicting the observables (regression)** or **predicting if a point is valid (classification)**



Kronheim, et al [2007.04506]

Caron, et al [1605.02797]; Ren, et al [1708.06615]; Staub [1906.03277]

Machine Learning in SUSY Model Building
Supervised Learning for Parameter Space Scans

- Application 1: Supervised Classifier to predict whether a point is valid
  - Physics case: cMSSM
  - Software: SPheno, MicrOMEGAS
  - Constraints: Higgs Mass and Dark Matter relic density
- Application 2: Supervised Regressor to predict Dark Matter relic density
  - Physics case: cMSSM
  - Software: SPheno, MicrOMEGAS
  - Constraints: Higgs Mass and Dark Matter relic density

$m_0(M_{GUT})$	[0,10] TeV
$m_{1/2}(M_{GUT})$	[0,10] TeV
$A_0(M_{GUT})$	[-10,10] TeV
$\tan(\square)(M_{SUSY})$	[1.5,50]

m_h	[122,128] GeV
$h\Omega_{DM}$	[0.08,0.14]

Let's go to the second notebook of examples!

Exploring Parameter Spaces with Search Algorithms

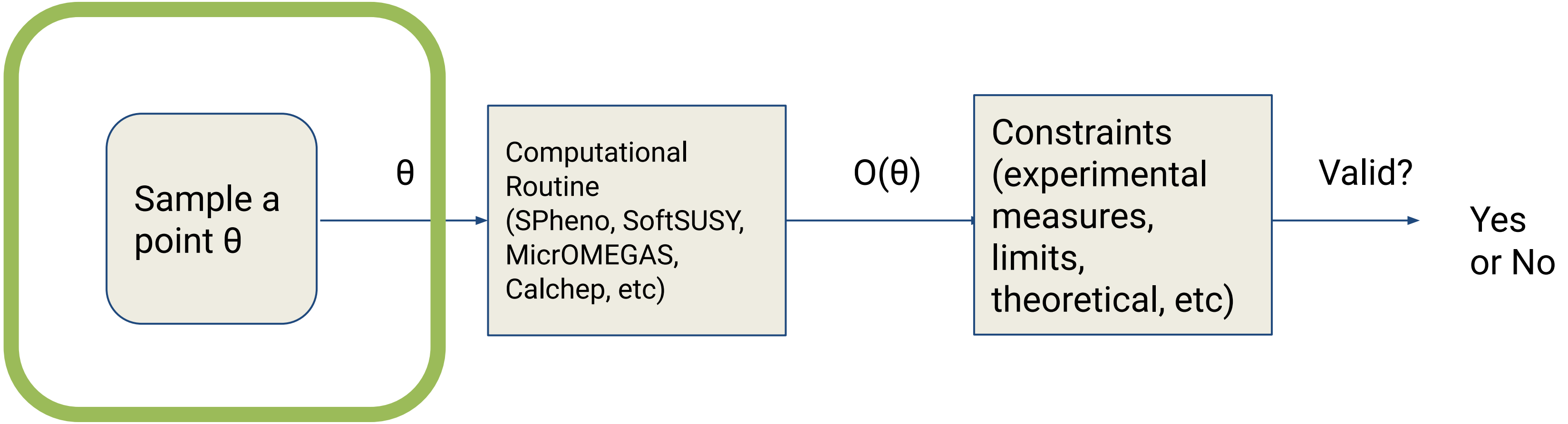
Exploring Parameter Spaces with AI/ML
Shortcomings of Previous Attempts

- These methodologies require **large amounts of training data** to **cover the whole parameter space**
- Predicting whether a point is valid using a **classifier**:
  - If training data do not cover the whole parameter space: **wrong guess**
- Predicting the observables using a **regressor**:
  - If training data do not cover the whole parameter space: **might map the parameter to observables incorrectly**
- **For highly constrained and realistic scans, it is computationally prohibitive to get enough valid points to use some of these methods**

Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

Exploring Parameter Spaces with AI/ML
Problem (re)framing: face the sampling

- The “aha” moment was to consider: **what if we change the sampling itself**



Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

Exploring Parameter Spaces with AI/ML
Problem (re)framing: BSM as a Black-Box

- We do not attempt to predict the observables or whether a point is valid
- Instead, we look at **how far a point is from being valid**
- Let **C(O)** be a function of a observable **O**

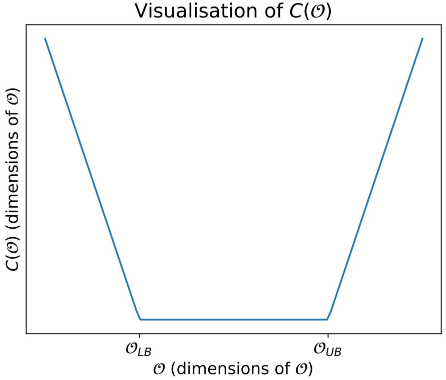
$$C(\mathcal{O}) = max(0, -\mathcal{O} + \mathcal{O}_{LB}, \mathcal{O} - \mathcal{O}_{UB})$$

- The set of valid points

$$\mathcal{V} = \{\theta^* : \theta \in \mathcal{P} \text{ s.t. } C(\theta) = 0\}$$

- Equivalently

$$\mathcal{V} = \{\theta^* : \theta \in \mathcal{P} \text{ s.t. } \theta^* = \text{argmin } C(\theta)\}$$

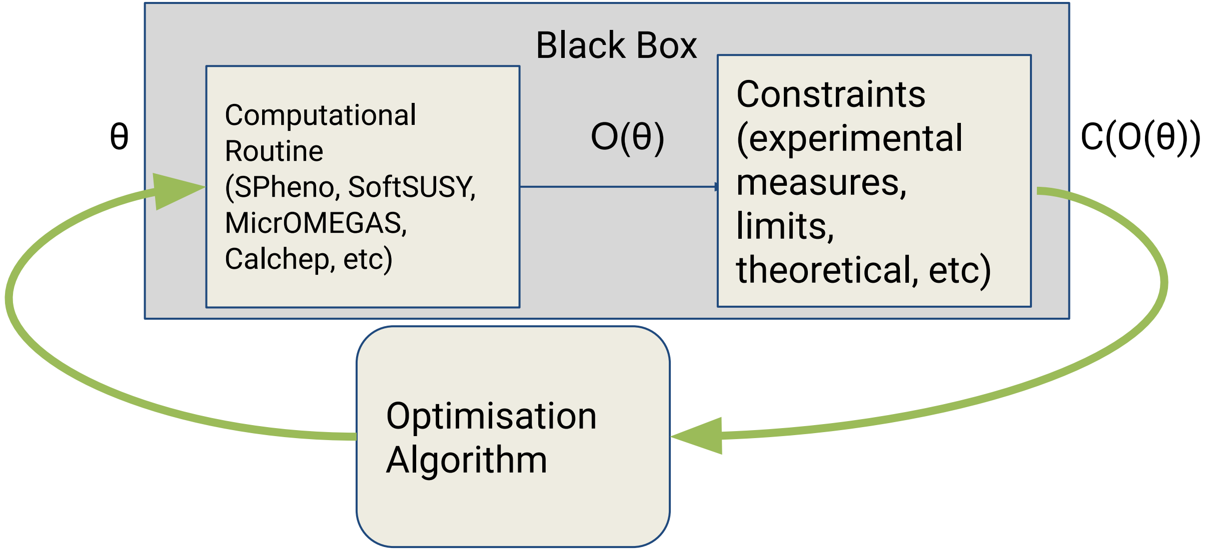


=> Finding the valid points is the same as minimising C(O)

Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

Exploring Parameter Spaces with AI/ML
Problem (re)framing: BSM as a Black-Box

- Since **O=O(θ)** we can **close the loop** and **optimise with regards to the parameters C(O)=C(O(θ)).** From the outside, **C(O(θ))** is a **Black-Box => Black-Box Optimisation Problem**



Exploring Parameter Spaces with AI/ML

Meet the Algorithms

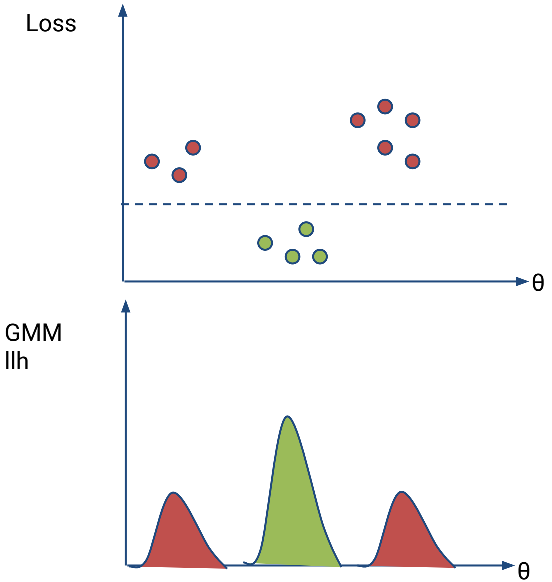
Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

- The fields of Artificial Intelligence and Machine Learning have a multitude of **search algorithms** for **black-box optimisation**
- We explored **three different classes of algorithms** to see their differences
  - A **Bayesian** Optimisation Algorithm: Tree-Parzen Estimator (TPE)
  - A **Genetic** Algorithm: Non-dominated Sorting Genetic Algorithm II (NSGA-II)
  - An (non-genetic) **Evolutionary** Algorithm: Covariant Matrix Approximation Evolution Strategy (CMA-ES)
- The algorithms are sequential, i.e. a new suggested point depends on the points seen so far
- All algorithms **do not require prior data**
  - => They adapt the search dynamically

Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

Exploring Parameter Spaces with AI/ML
Meet the algorithms: TPE

- Sample randomly an initial set of points
  - **Sort the parameter points** by their **loss**
  - **Split** points between **good** and **bad** through a moving quantile heuristic
  - **Fit** a Gaussian Mixture Model (learnable component of the algorithm) on each **good** and **bad** set
  - **Sample** a point from the **good**, and **keep** it if its **likelihood is greater** than being **bad**
  - Repeat



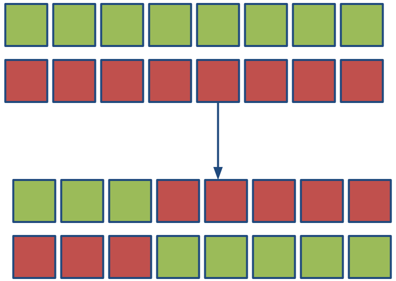
Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

Exploring Parameter Spaces with AI/ML

Meet the algorithms: NSGA-II

- Encode parameter space point (a vector) as genes

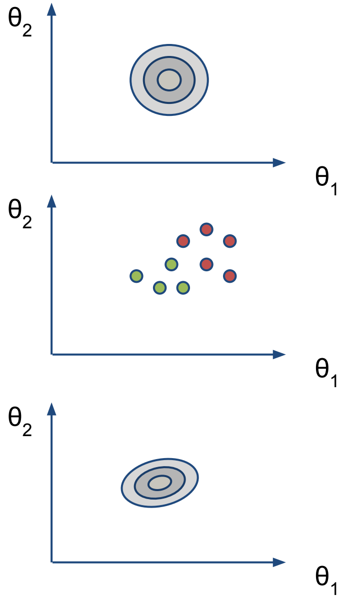
- Prepare initial population
  - Evaluate their **fitness** (i.e. the **loss**)
  - **Sort** them by their fitness
  - **Keep** the best, discard the rest
  - Create **offspring** from the best (**crossover**)
  - Apply random **mutations**
  - Repeat



Exploring Parameter Spaces with Artificial Intelligence and Machine Learning Black-Box Optimisation Algorithms
Fernando Abreu de Souza, MCR, Nuno Filipe Castro, Mehraveh Nikjoo, Werner Porod
Phys.Rev.D 107 (2023) 3, 035004
https://arxiv.org/abs/2206.09223

Exploring Parameter Spaces with AI/ML
Meet the algorithms: CMA-ES

- Initialise a **multivariate normal** with random mean and identity covariance matrix
    - Sample a population
    - **Evaluate** members of population and **sort** by **loss**
    - Use the best and compute their statistics
        - Mean
        - Covariance
    - **Update** mean and covariance matrix with weighted rolling updates
    - Repeat



Exploring Parameter Spaces with AI/ML

Evolutionary Strategy for Parameter Space Scans

- Application 3: Covariant Matrix Approximation Evolutionary Strategy for Parameter Space Exploration
  - Physics case: cMSSM
  - Software: SPheno
  - Constraints: Higgs Mass and muon (g-2)

$m_0(M_{GUT})$	[0,10] TeV
$m_{1/2}(M_{GUT})$	[0,10] TeV
$A_0(M_{GUT})$	[-10,10] TeV
$\tan(\square)(M_{SUSY})$	[1.5,50]

$$\Delta a_\mu = a_\mu^{\exp} - a_\mu^{SM} = (25.1 \pm 5.9) \times 10^{-10}$$

m_h	[122,128] GeV
Δa_μ	[7.4,42.8] 10^{-10}

Let's go to the third notebook of examples!

Ongoing work

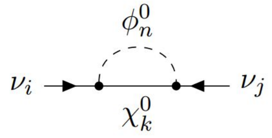
[No working title yet]
Fernando Abreu de Souza, MCR, Andreas Karle, Nuno Filipe Castro, Werner Porod
230X.ABCDE

Based on: Leptogenesis and muon (g − 2) in a scotogenic model, A. Alvarez, A. Banik, R. Cepedello, B. Herrmann, W. Porod, M. Sarazin, M. Schnelke [2301.08485]

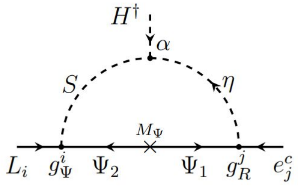
Exploring Parameter Spaces with AI/ML
Scotogenic explanation to muon (g-2)

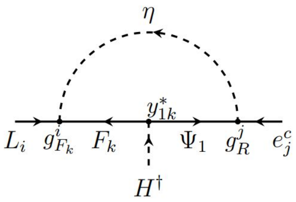
- Additional states can generate radiative neutrino masses as well as a BSM contribution to muon (g-2)
- 46 free parameters 🤯 (we have complex numbers)
- 30 constraints
  - Higgs mass, neutrino data, muon (g-2), flavour violation bounds

	Ψ_1	Ψ_2	F_1	F_2	η	S
$SU(2)_L$	**2**	**2**	**1**	**1**	**2**	**1**
$U(1)_Y$	-1	1	0	0	1	0



$\equiv \overline{\nu_j^c} \left(\mathcal{M}_\nu\right)_{ji} \nu_i$



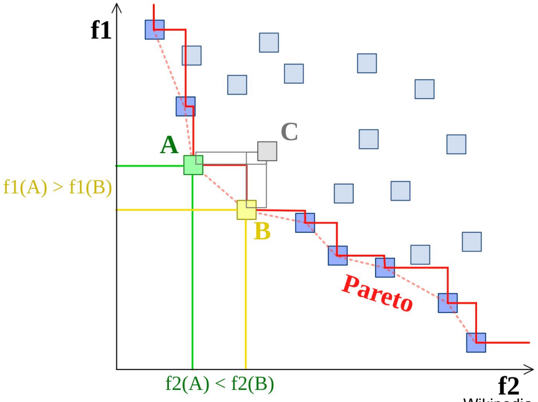


[No working title yet]
Fernando Abreu de Souza, MCR, Andreas Karle, Nuno Filipe Castro, Werner Porod
230X.ABCDE

Based on: Leptogenesis and muon (g − 2) in a scotogenic model, A. Alvarez, A. Banik, R. Cepedello, B. Herrmann, W. Porod, M. Sarazin, M. Schnelke [2301.08485]

Exploring Parameter Spaces with AI/ML
Scotogenic explanation to muon (g-2)

- Exploring a new approach: **Multi-Objective Optimisation**
  - Instead of optimising to the sum of the objectives, **optimise all the objectives jointly**
  - NSGA-3: a bettered NSGA-II, for **many** objectives
  - **Beyond** Casas-Ibarra parametrisation
- So far:
  - Random sampling: no valid points after 1 Million points
  - NSGA-3: Valid points after O(10^4) steps



Wikipedia

[No working title yet]
Jorge C. Romao, MCR
23WX.ABCDE

Based on: BFB conditions on a class of symmetry constrained 3HDM
Rafael Boto, Jorge C. Romao, Joao P. Silva
[2106.11977]

Exploring Parameter Spaces with AI/ML
Highly Constrained 3HDM

- 16 free parameters
- 60 constraints 🤯
  - Oblique STU parameters
  - Boundedness from Below
  - Perturbative Unitarity
  - LHC Higgs couplings constraints
  - B-> S gamma
- Random search efficiency: **Around 1:10 billion** (1 week on 16 cores produces O(10) points)
  - Model builders often focus around **alignment limits**

$$\alpha_1,\ \alpha_2,\ \alpha_3,\ \gamma_1,\ \gamma_2\ \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]; \qquad \tan\beta_1,\ \tan\beta_2\ \in [0, 10]\,;$$

$$m_{H_1} \equiv m_{h_2},\ m_{H_2} \equiv m_{h_3}\ \in [125, 1000]\ \text{GeV};$$

$$m_{A_1},\ m_{A_2}\ m_{H_1^\pm},\ m_{H_2^\pm}\ \in [100, 1000]\ \text{GeV};$$

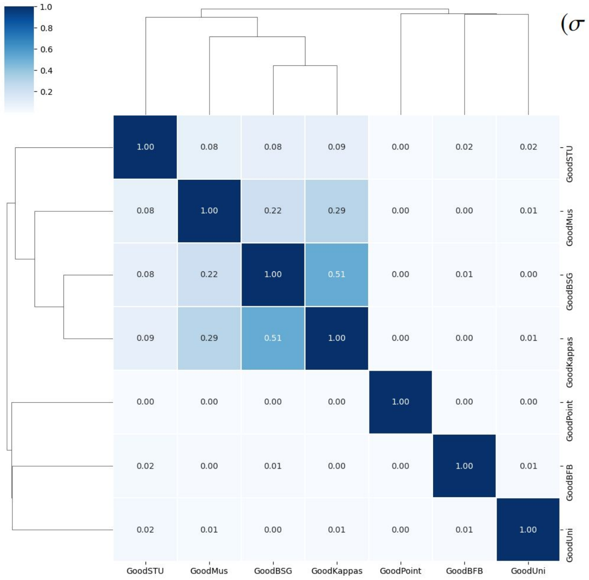
$$m_{12}^2, m_{13}^2, m_{23}^2 \in \left[\pm 10^{-1}, \pm 10^{7}\right]\ \text{GeV}^2\,,$$

$$2.87 \times 10^{-4} < \text{BR}(B \to X_s\gamma) < 3.77 \times 10^{-4}$$

$$\mu_{if}^h = \left(\frac{\sigma_i^{\text{3HDM}}(pp \to h)}{\sigma_i^{\text{SM}}(pp \to h)}\right)\left(\frac{\text{BR}^{\text{3HDM}}(h \to f)}{\text{BR}^{\text{SM}}(h \to f)}\right)$$

ATLAS-CONF-2018-031

$$(\sigma \times \text{B})_{if} = \kappa_i^2 \sigma_i^{\text{SM}} \frac{\kappa_f^2 \Gamma_f^{\text{SM}}}{\kappa_H^2 \Gamma_H^{\text{SM}}},$$

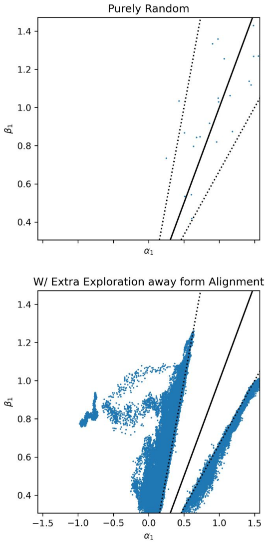


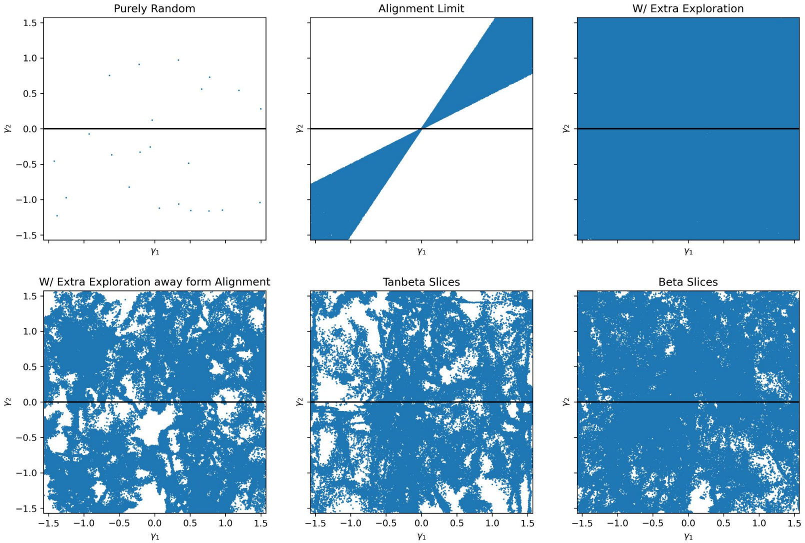
[No working title yet]
Jorge C. Romao, MCR
23WX.ABCDE

Based on: BFB conditions on a class of symmetry constrained 3HDM
Rafael Boto, Jorge C. Romao, Joao P. Silva
[2106.11977]

Exploring Parameter Spaces with AI/ML
Highly Constrained 3HDM

- Go **beyond alignment limits**
- **We find valid points very quickly**: Around 1 after 1000 attempts (c.f. 1 to 10 billion) in **<O(10) minutes**
- Algorithm tends to find the same region regularly => **Adapt for exploration** [current WIP]



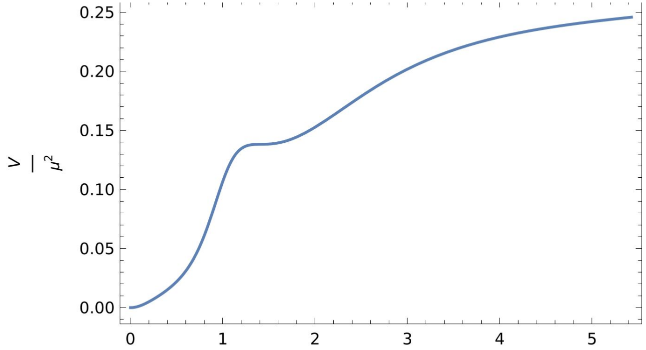


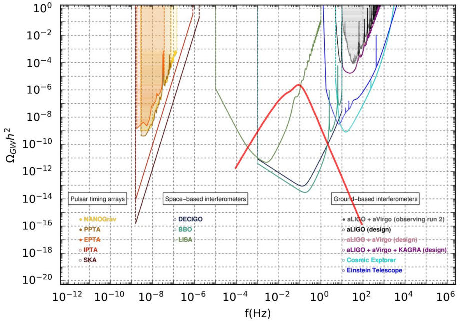
Gravitational Waves and Gravitino Mass in No-Scale SUGRA Wess-Zumino model with Polonyi Term [Working title]
MCR, Stephen F. King
23WX.ABCDE

To be presented next week, Wednesday 19th at 5:40pm

Exploring Parameter Spaces with AI/ML
Gravitational Waves from No-Scale SUGRA

- Gravitational Waves production from a feature of the inflation potential
- The production of observable GW is tied to the nuanced shape of the kink, i.e. **fine-tuned**
- With CMA-ES, we were able to find potentials with the appropriate shape

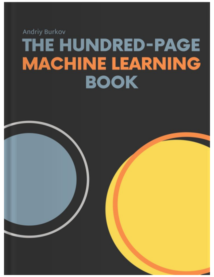


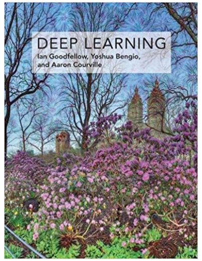


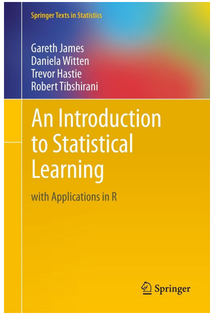
Further Reading

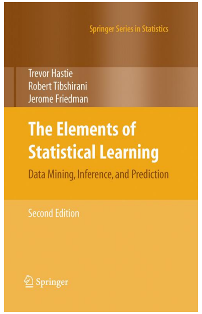
These are free

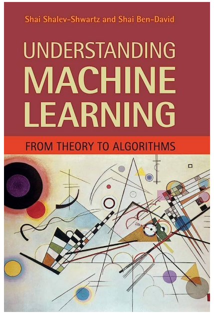
- The Hundred-Page Machine Learning Book - Burkob
- Deep Learning - Goodfellow, Bengio, Courville
- An Introduction to Statistical Learning - James, Witten, Hastie, Tibshirani
- The Elements of Statistical Learning - Hastie, Tibshirani, Friedman
- Understanding Machine Learning - Shalev-Shwartz, Ben-David





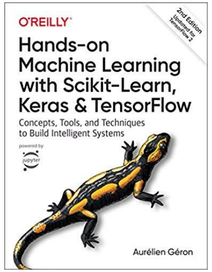


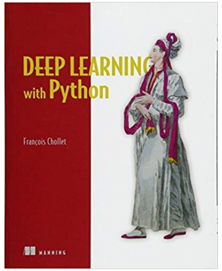


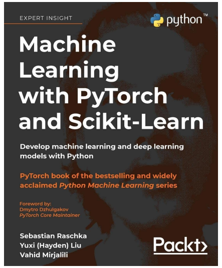


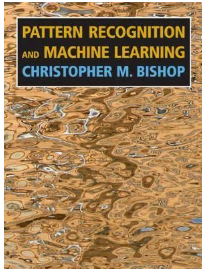
Not free, but very good

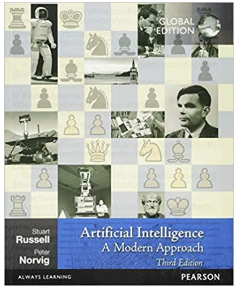
- Hands-On Machine Learning w/ Scikit-Learn & TensorFlow - Geron
- Deep Learning w/ Python - Chollet
- Machine Learning with PyTorch and Scikit-Learn - Raschka
- Pattern Recognition and Machine Learning - Bishop
- Artificial Intelligence, a Modern Approach - Russell, Norvig
- Machine Learning, a Probabilistic Perspective - Murphy

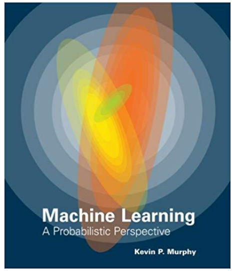












Thanks!

mcromao@lip.pt
miguel@miguelromao.me