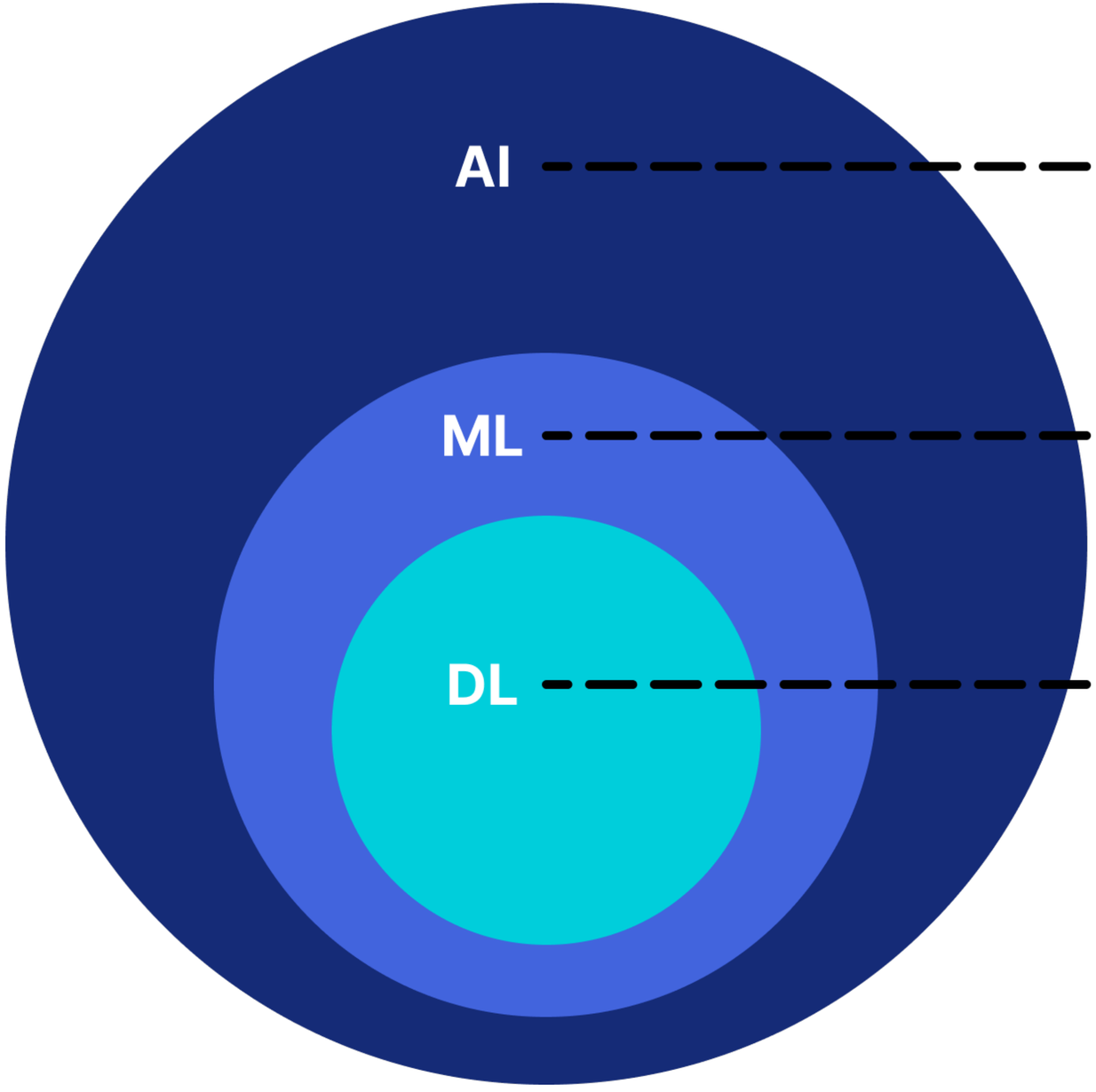
Practical Machine Learning

With The Three Hundred simulations

Daniel de Andrés 9 July, 2024

This workshop is for you if you have zero knowledge of ML. Basic python might be useful.



Artificial Intelligence
Engineering of making intelligent machines and programs.

Machine Learning
Ability to learn without being explicitly programmed.

Deep Learning
Learning based on deep neural networks.

neurosnap.ai

This workshop is about:

- Have an insight about what problems ML can address.
- The most used ML algorithms.
- The software for ML.
- Exercise.

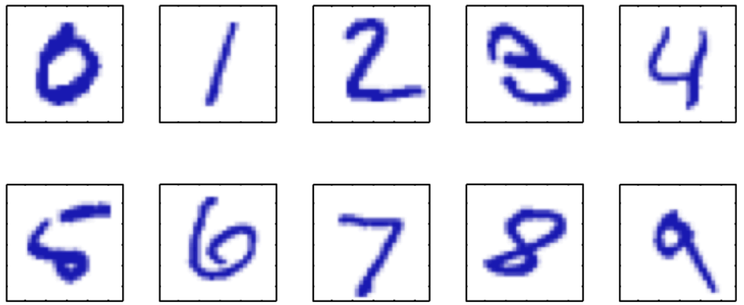
This workshop is NOT about:

- Mathematics for ML
- Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: springer.

Why ML

Bishop (2006)

Figure 1.1 Examples of hand-written digits taken from US zip codes.



- The field of **pattern recognition** is concerned with the **automatic discovery of regularities in data** through the use of **computer algorithms** and with the use of these regularities to take actions such as **classifying** the data into different categories.

- Example can be **identifying handwriting numbers**: The goal is to build a machine that will take an **image tensor x** as **input** and that will produce the identity of the **digit 0, . . . , 9** as the output. This is no a trivial task due to the high variability of handwriting.

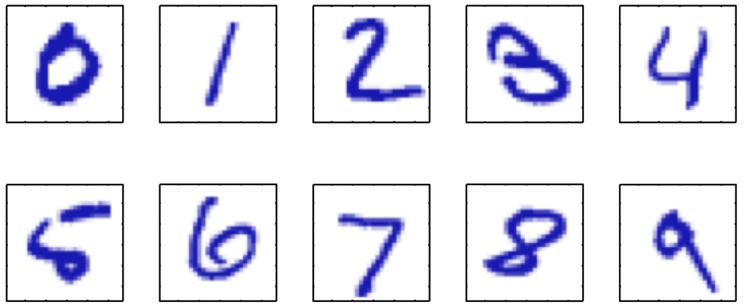
- **Modelling approach** (as traditional physics): **handcrafted rules** or heuristics for distinguishing the digits **based on the shapes of the strokes**, but in practice such an approach leads to a proliferation of rules and of exceptions to the rules and so on, and invariably gives **poor results.**

- Far better results can be obtained by adopting **a machine learning approach** in which a large set of **N digits** {x_1,…,x_n} called a ***training set*** is used to tune the **parameters** of an **adaptive model.**

Why ML

Bishop (2006)

Figure 1.1 Examples of hand-written digits taken from US zip codes.



- The **categories** of the digits are typically **known in advanced** *(supervised learning*), typically by volunteers that labelled them. This is express with the ***target vector***.

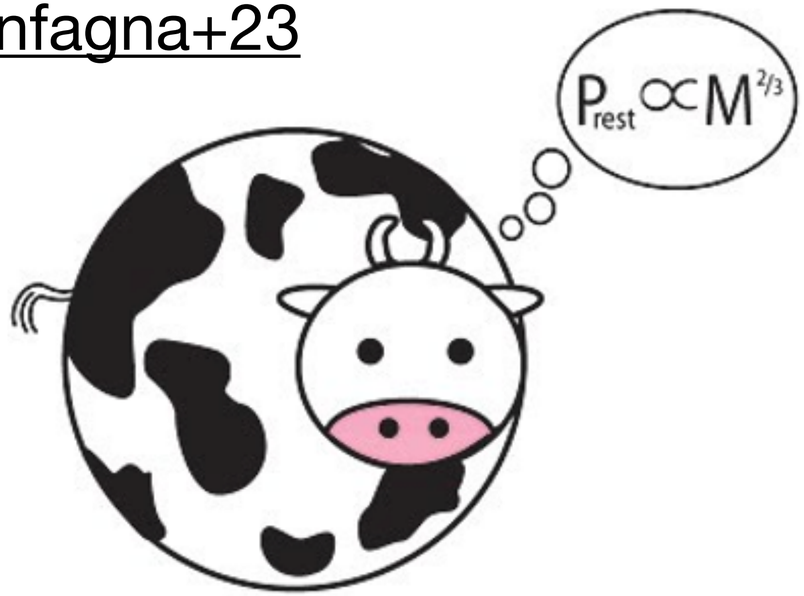
- The ML model can be expressed as $y(x)$ and its precise form is determined during the ***training phase***. Once the model is trained it can then determine the identity of **new image digits**, which comprises the ***test set***.

- The ability to categorise correctly new examples that differ from those used for training is known as ***generalisation.*** In **practical applications,** the variability of the input vectors will be such that the **training data can comprise only a tiny fraction of all possible input vectors**, and so generalisation is a central goal in pattern recognition.

.

Why ML

"**Traditional methods**" to infer masses use The300 data and assume **symmetries of the ICM** that lead to a **bias** result. Example: Hydrostatic Equilibrium masses. Gianfagna+23



- **ML** methods use The300 data to learn **directly the underlying relation between mass and observables**.
- The main **limitation** is the **physics** implemented in the simulations.
- In general, ML allows to **address problems** in a **different way**, o problems that were **intractable**.

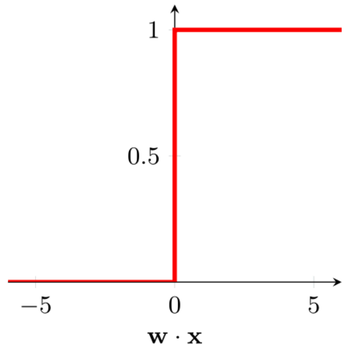
de Andres+22

Artificial intelligence applications in astronomy

Not a casual success Smith and Geach, 2023

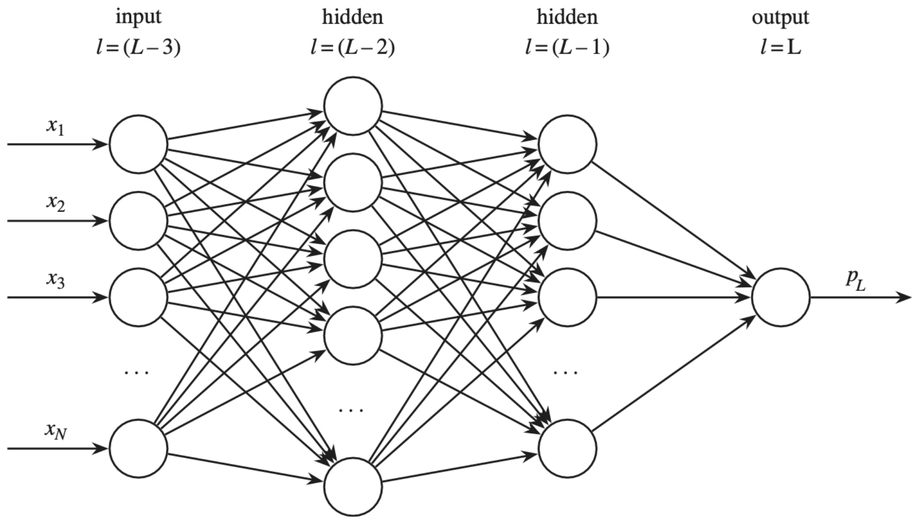
- Data available (e.g, The300) for training models and also GPU acceleration are exponentially growing.
- One example is the neural network, where data is analysed by a set of artificial neurons.
- The parameters/weights of the neurons are randomly initialised and are updated by experience, minimising a loss function. The neurons learn specific tasks.
- Deep Learning is a subset of ML learning methods that refers to “deep” neural networks.

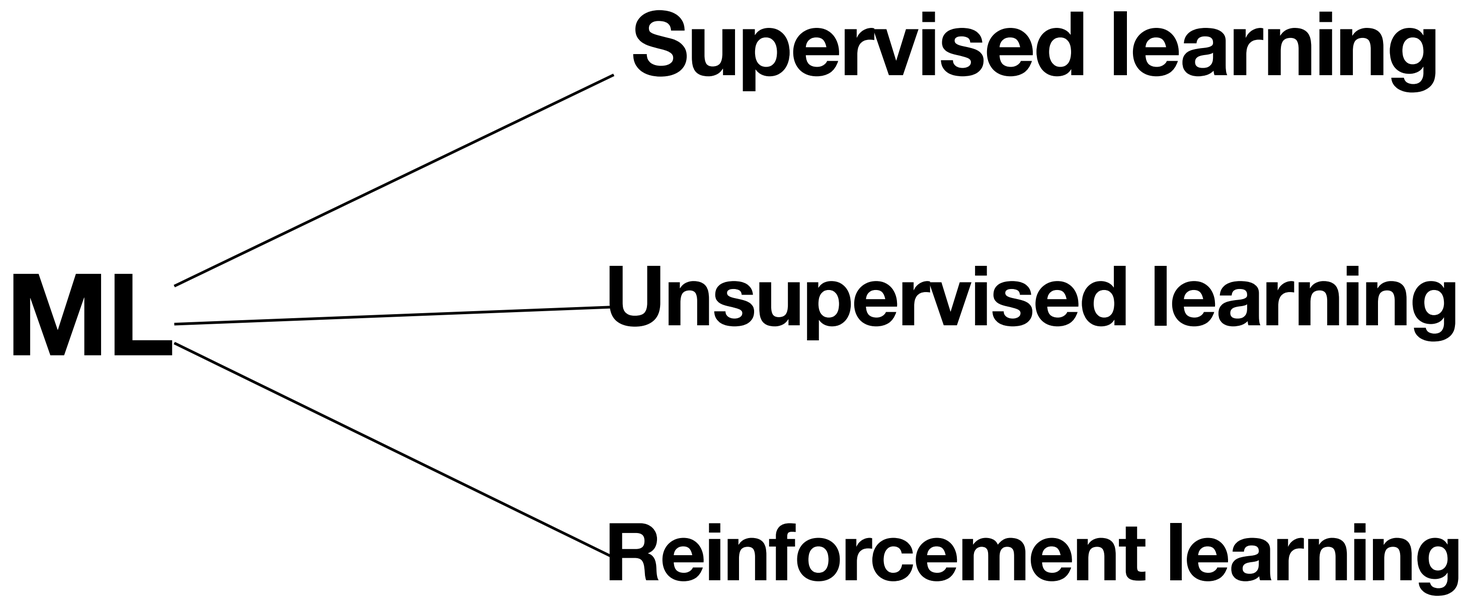
Perceptron



$$\text{prediction} = H(\mathbf{w}\cdot\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{w}\cdot\mathbf{x} < 0, \\ 1 & \text{if } \mathbf{w}\cdot\mathbf{x} \geq 1, \end{cases}$$

Multilayer perceptron (MLP), or feed-forward NN

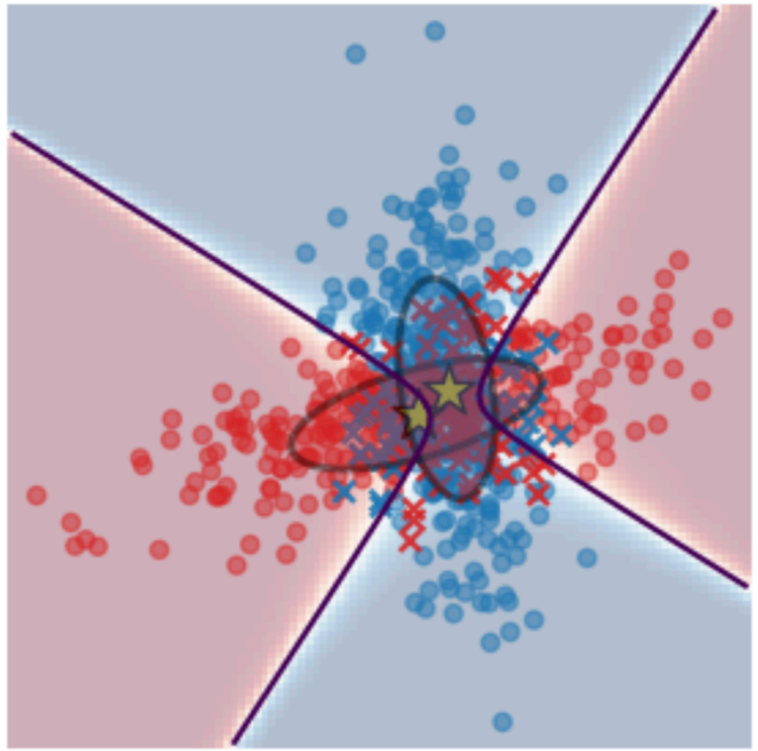



ML
Supervised learning
Unsupervised learning
Reinforcement learning

Supervised learning (labeled data)

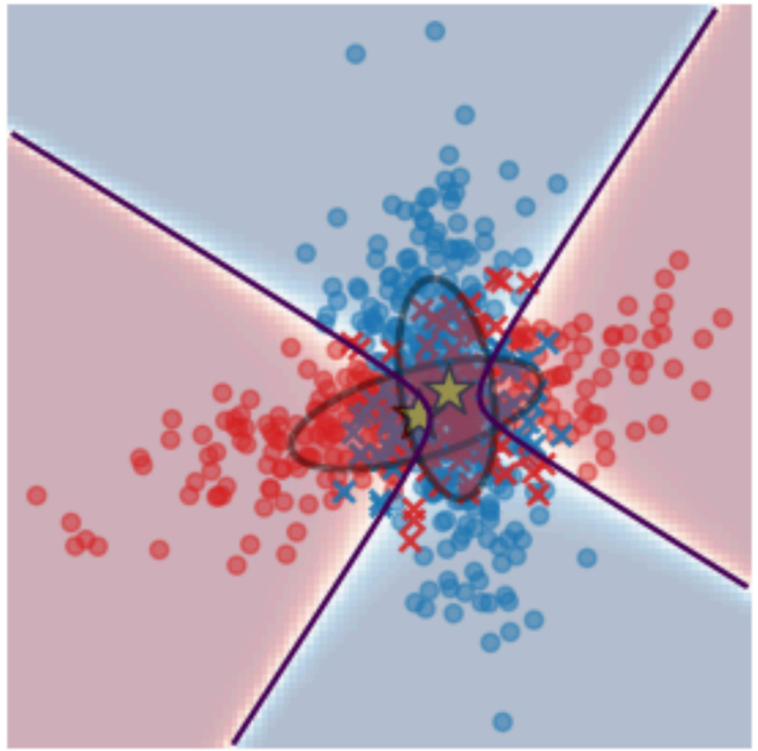
Supervised learning (labeled data)

Classification

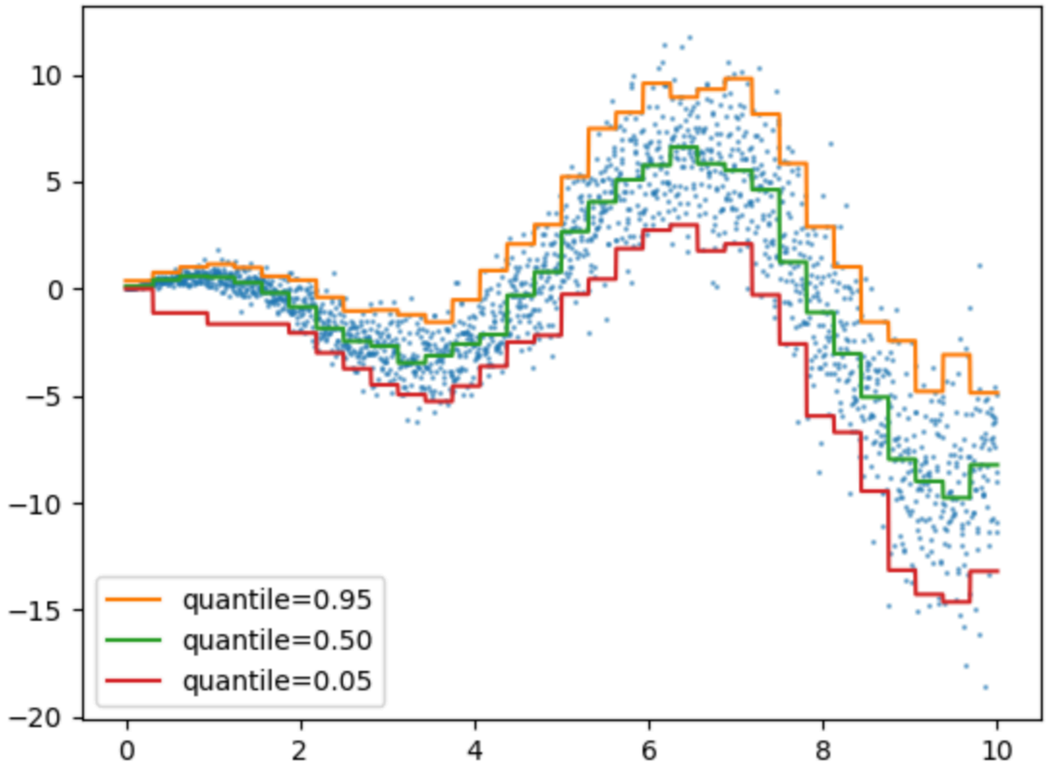


Supervised learning (labeled data)

Classification



Regression



Supervised learning (labeled data)

Classification

2. Classification Loss:

Binary Classification:

- Hinge Loss
- Sigmoid Cross Entropy Loss
- Weighted Cross Entropy Loss

Regression

1. Regression Loss:

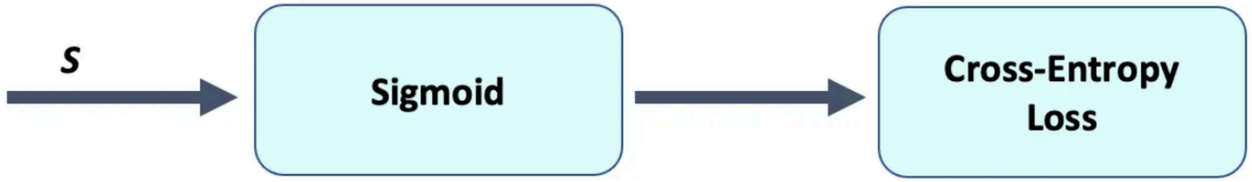
- Mean Square Error or L2 Loss
- Mean Absolute Error or L1 Loss
- Huber Loss

Supervised learning (labeled data)

Classification

Sigmoid Cross Entropy Loss

$$J(\hat{y}, y) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)]$$

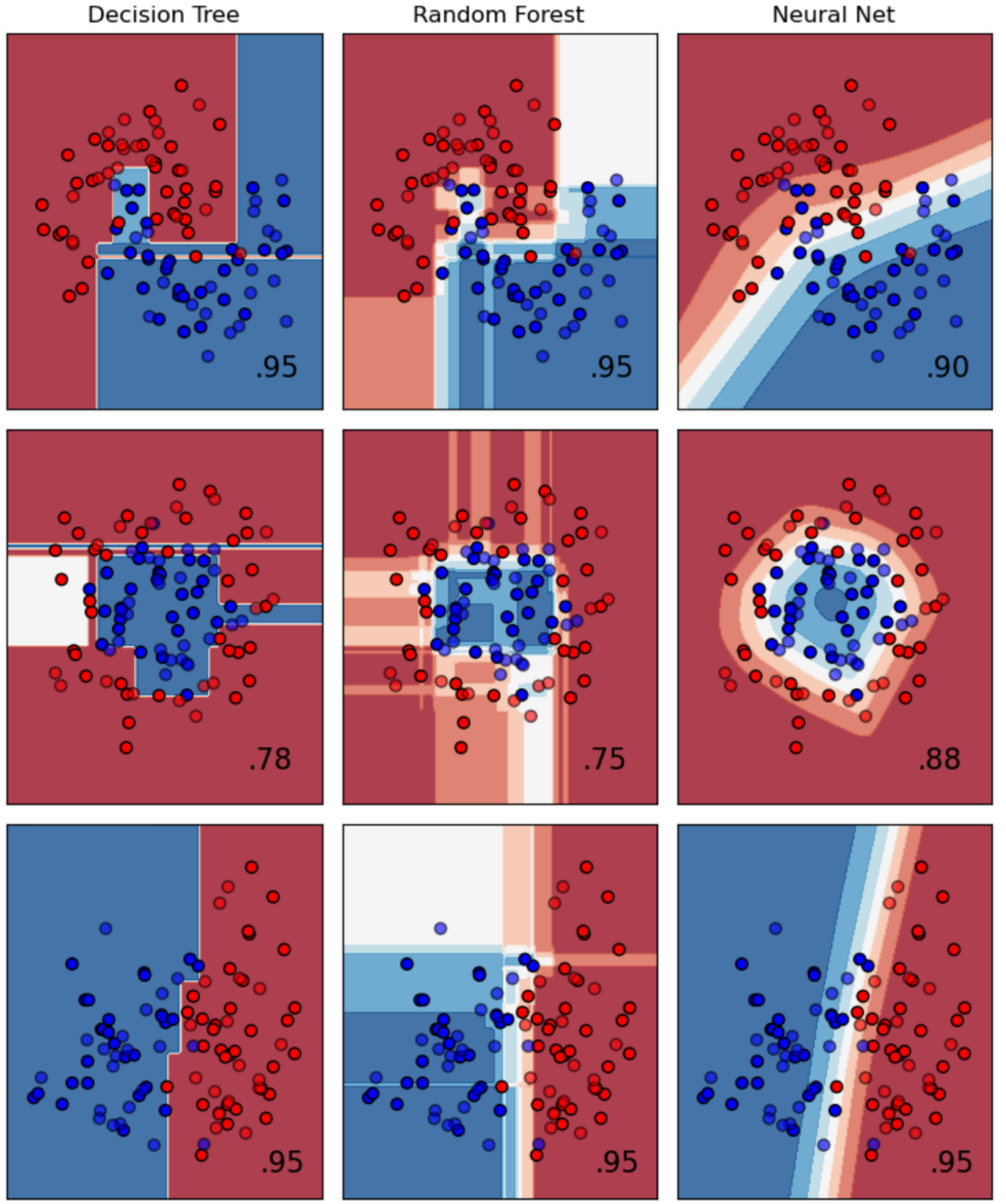


Regression

1. Regression Loss:

- Mean Square Error or L2 Loss
- Mean Absolute Error or L1 Loss
- Huber Loss

- Different models yield to different solutions, even if the task is the same. There is a plethora of models!



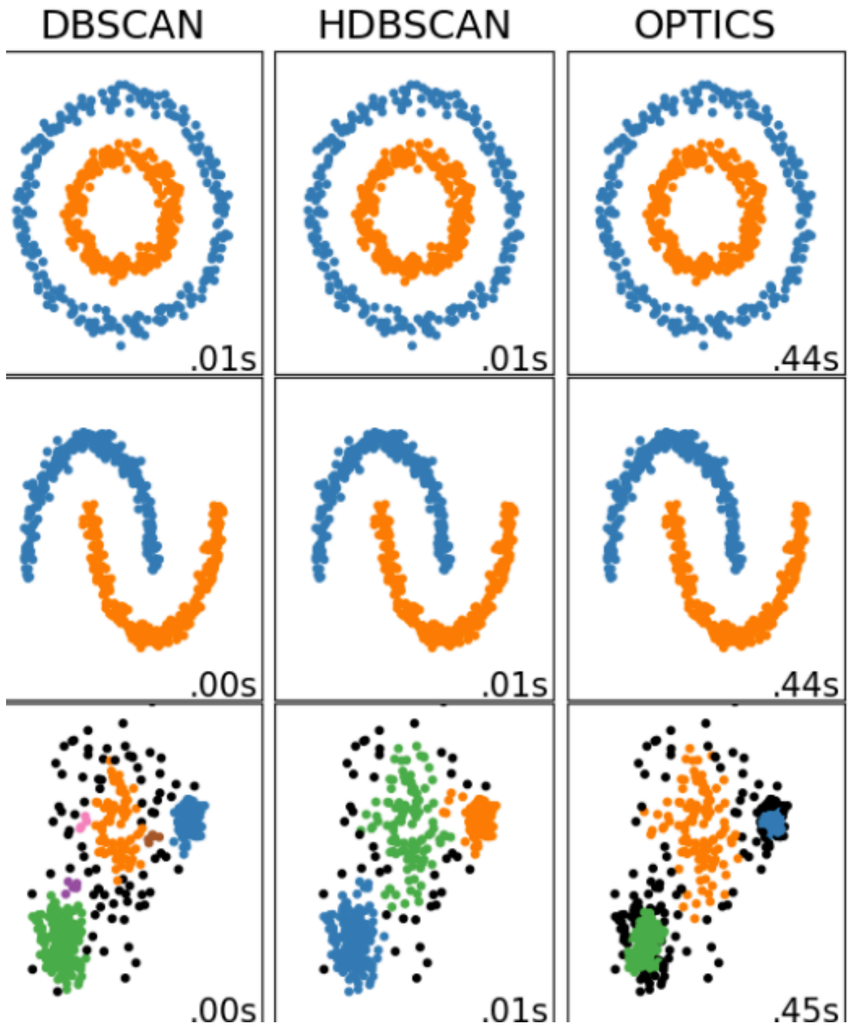
https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier-comparison-py

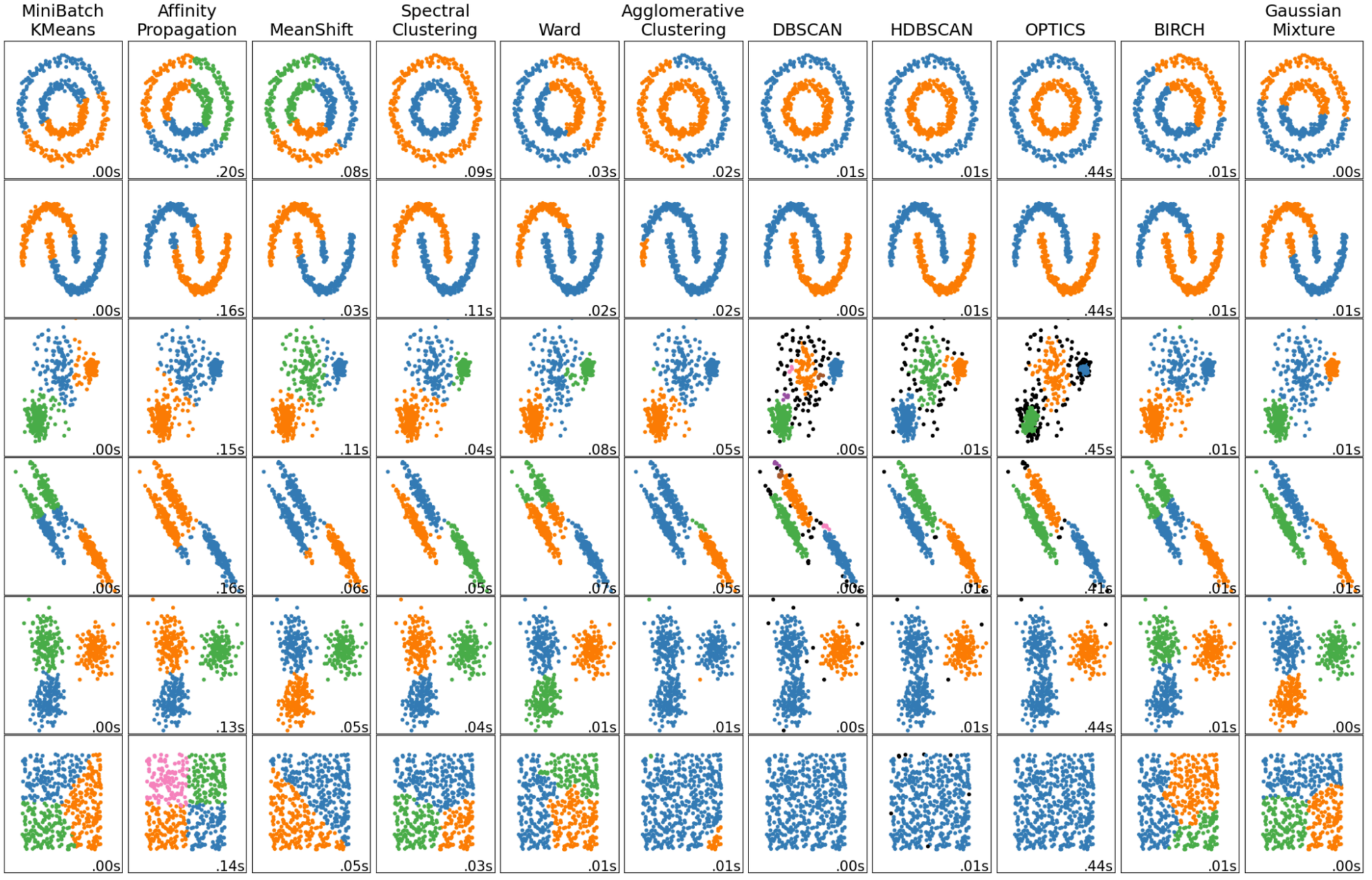
Unsupervised learning (no labels)

Comparing different **clustering** algorithms on toy datasets.

This example shows characteristics of different clustering algorithms on datasets that are “interesting” but still in 2D.

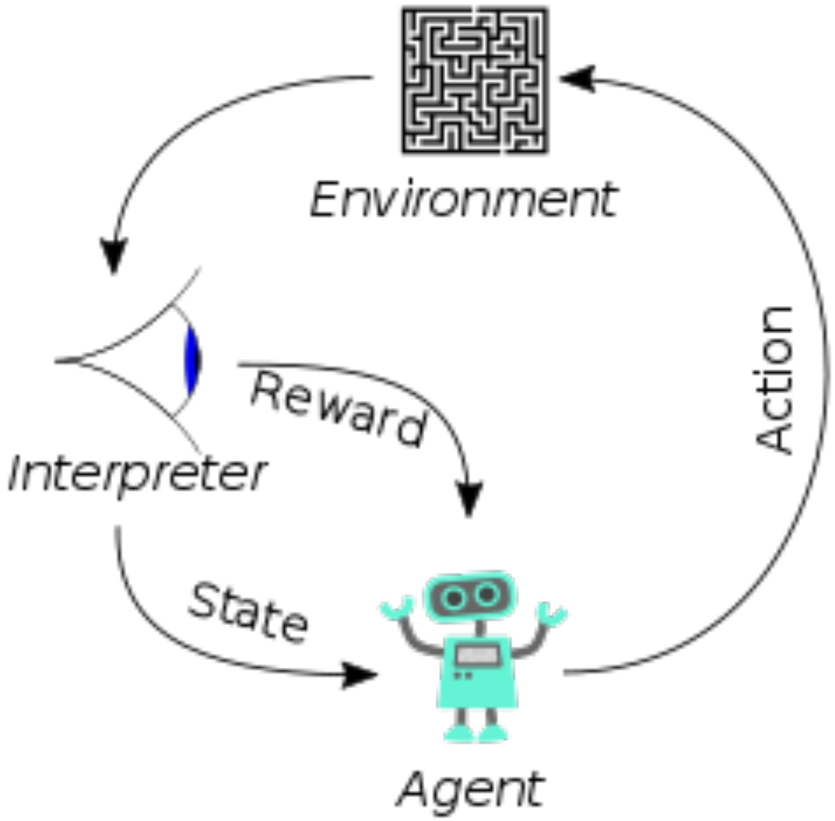
https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py




MiniBatch KMeans
Affinity Propagation
MeanShift
Spectral Clustering
Ward
Agglomerative Clustering
DBSCAN
HDBSCAN
OPTICS
BIRCH
Gaussian Mixture
.00s .20s .08s .09s .03s .02s .01s .01s .44s .01s .00s
.00s .16s .03s .11s .02s .02s .00s .01s .44s .01s .01s
.00s .15s .11s .04s .08s .05s .00s .01s .45s .01s .01s
.00s .16s .06s .05s .07s .05s .00s .01s .41s .01s .01s
.00s .13s .05s .04s .01s .01s .00s .01s .44s .01s .00s
.00s .14s .05s .03s .01s .01s .00s .01s .44s .01s .00s

Reinforcement learning

It is a machine learning technique that trains software to make decisions to achieve the most optimal results. It mimics the trial-and-error learning process that humans use to achieve their goals



Basically it play chess again itself innumerable times to achieve superhuman performance

http://arxiv.org/pdf/1811.12560

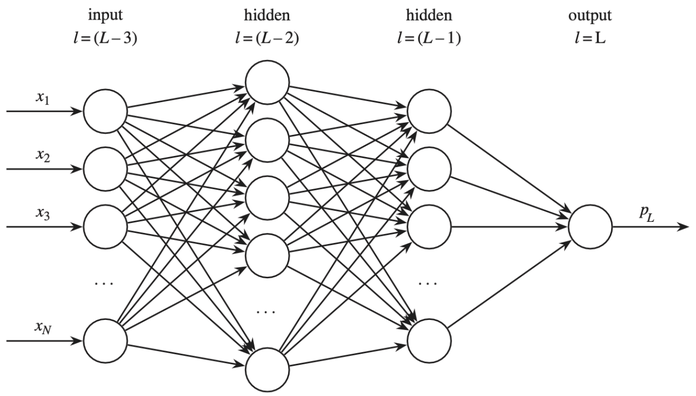
Artificial intelligence applications in astronomy

Many applications (Huertas-Company & Lanusse, 2023)

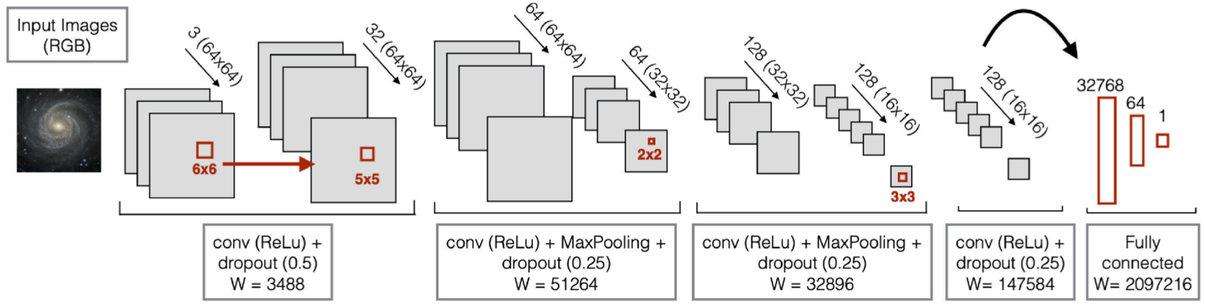
- Applications closer to the general computer vision tasks, such as morphological classifications of galaxies. Vega-Ferrero+2020
- To derive physical properties such as the mass of galaxy clusters (this work).
- For assisted discovery, e.g., using symbolic regression to discover a new analytic formula for the “overdensity” in dark matter halos (Cranmer+2020).
- For cosmology, e.g, reconstructing the initial conditions from halo mass fields in N-body simulations (Modi+2021).

ML in astronomy: A historical perspective

Multilayer perceptron (MLP), or feed-forward NN



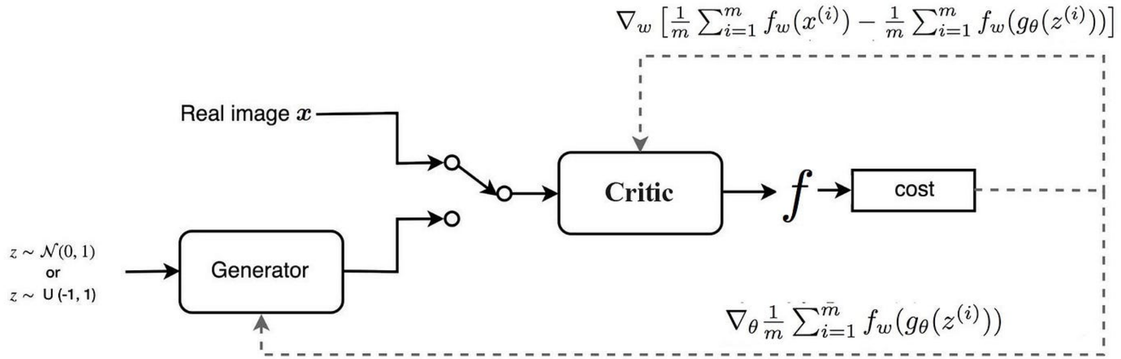
First wave: ~1990’, astronomers needed to create a data set of well designed features. Model MLP.



Second wave: early 2010’, Deep learning methods are fed with raw data. The model selects automatically the relevant features. Model: Convolutional neural networks.

Vega-Ferrero J., et al., 2021

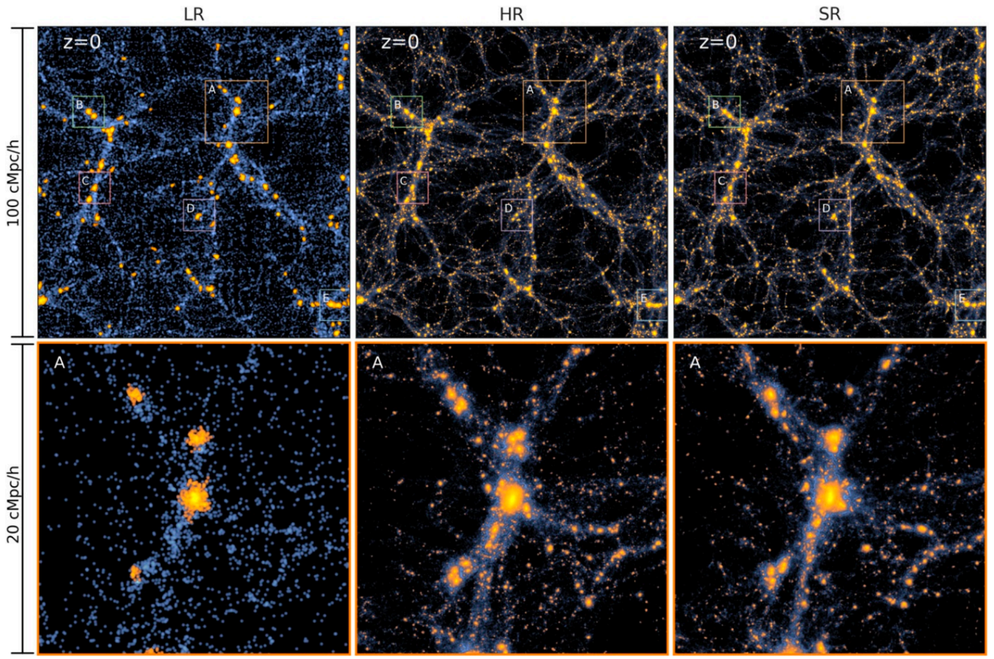
ML in astronomy: A historical perspective

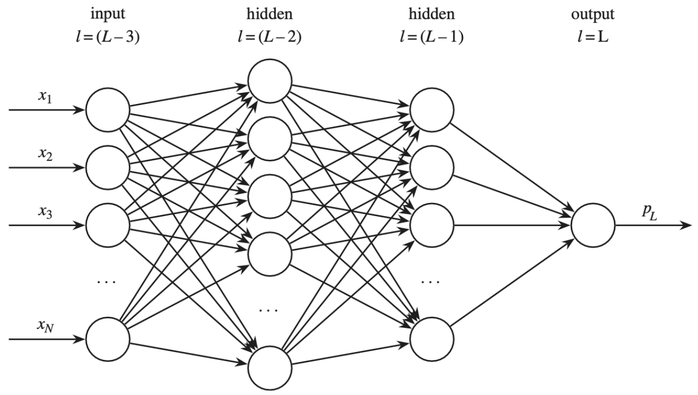


https://jonathan-hui.medium.com/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

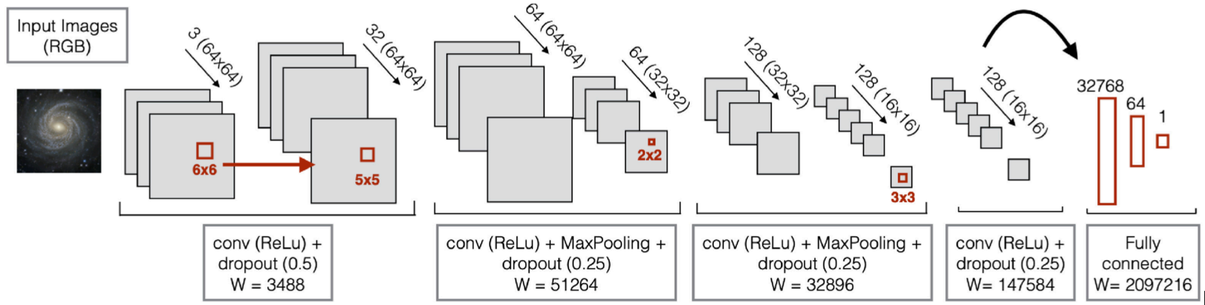
Third wave: mid 2010', Deep generative models. Finding representations with unlabelled data. Model: Generative adversarial network (GAN).

Super-Resolution (SR) corresponds to the simulation generated by the GAN model (output) from the LR as input. Credits: Li et al. (2021).

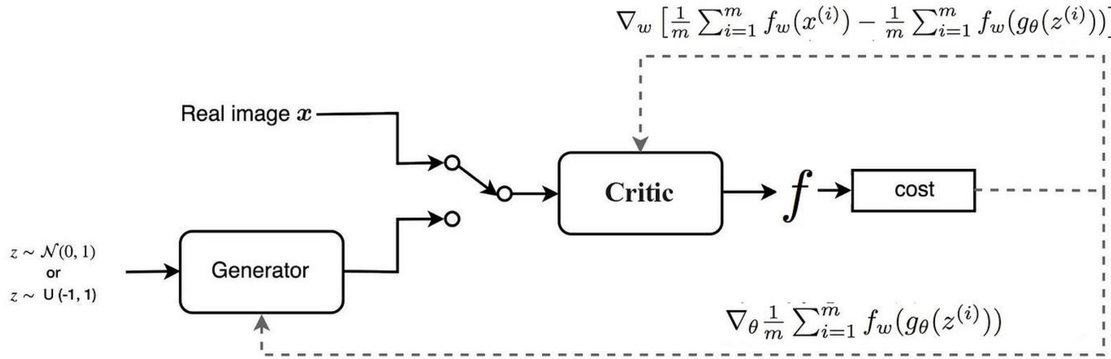


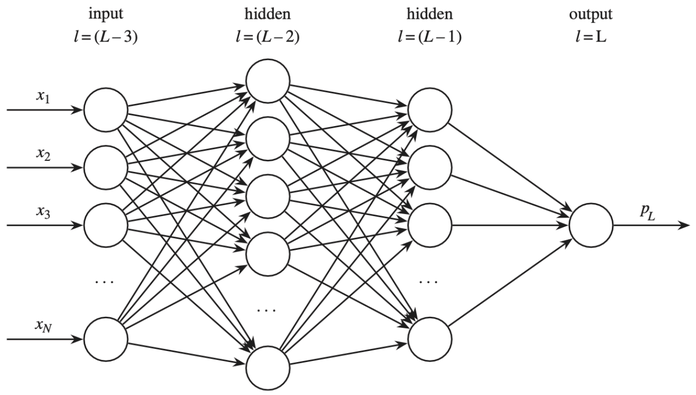


- General ML algorithms

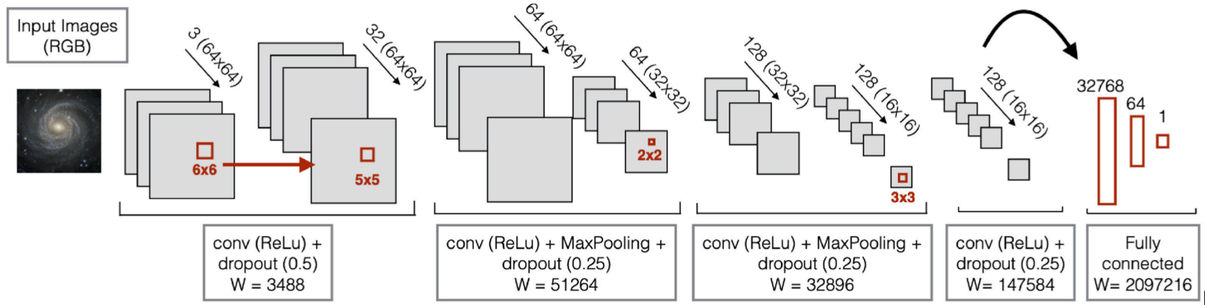


- Deep learning

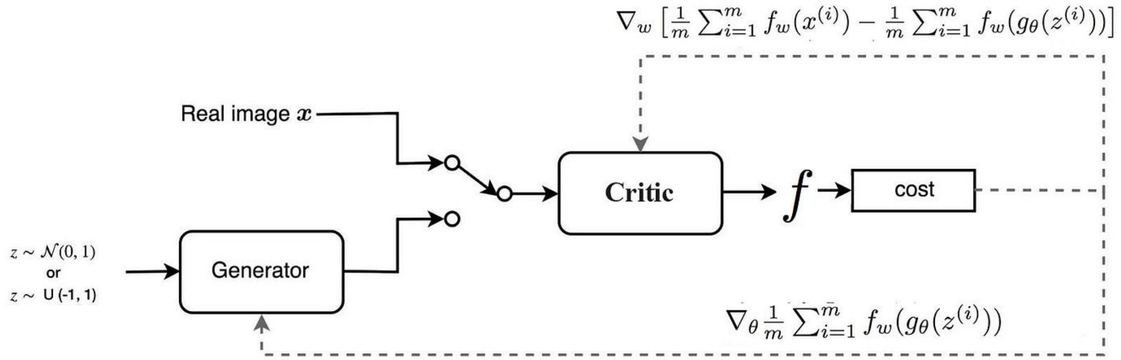


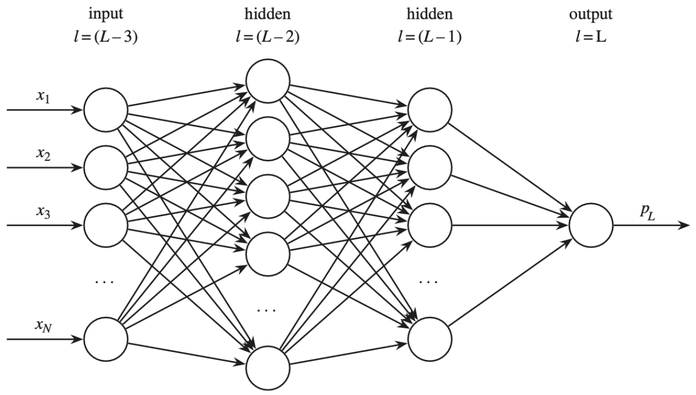


- ML: This workshop

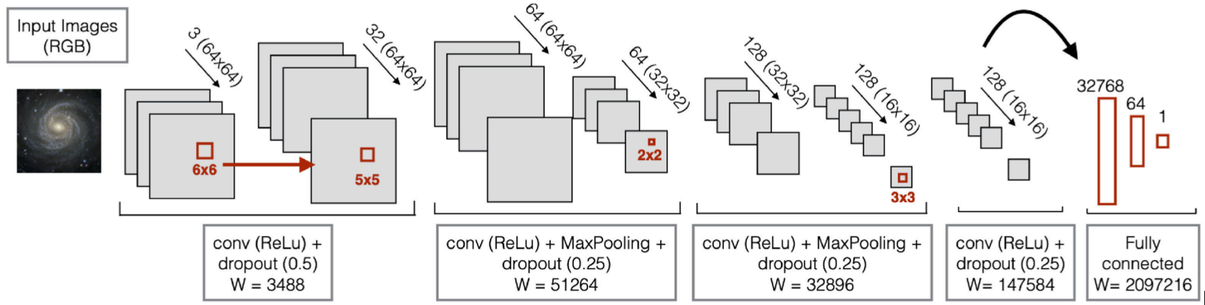


- Deep learning: Next workshop ;)

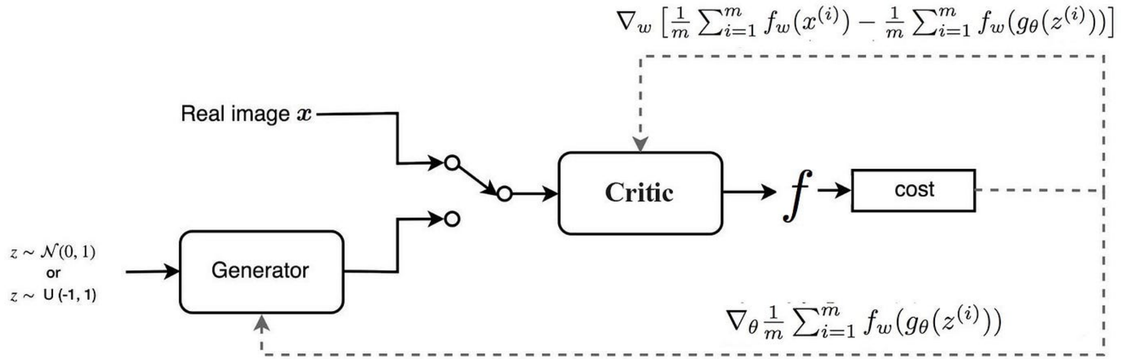




- **ML**: This workshop
- Topology: Best for tabular data, e.g., AHF.

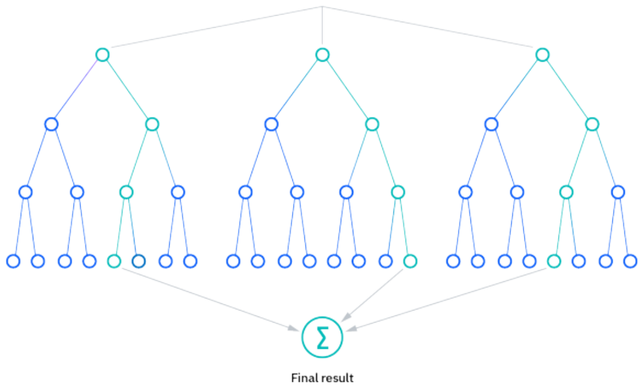


- **Deep learning:** Next workshop ;)
- Topology: Best for images, such as, density fields, etc.

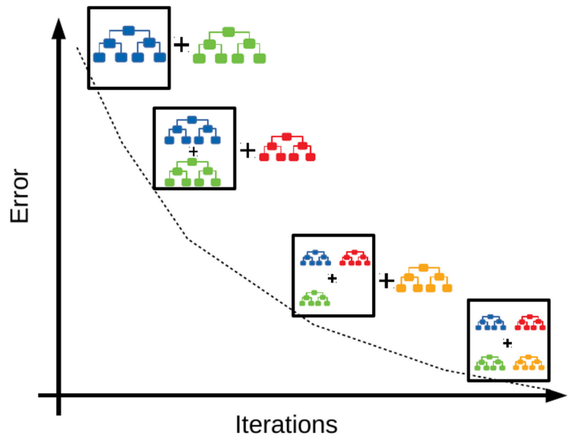


- ML: This workshop

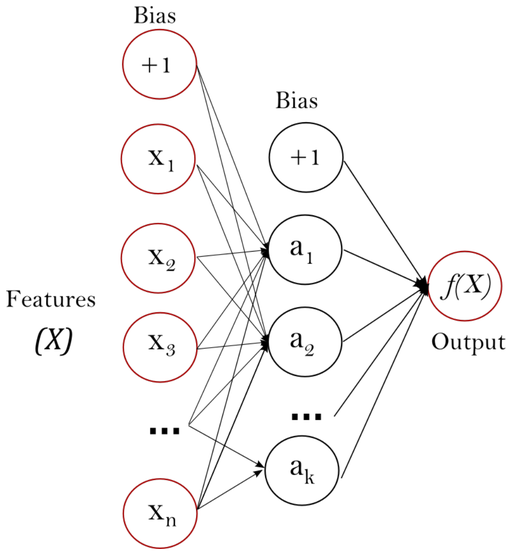
Random Forest



XGBoost NGBoost



Neural Network



- Random forest, XGBoost, NGBoost are **tree**-based approaches. They are best if the topology of the data is a table, e.g. AHF. No mathematical theorem, but XGB is typically the most accurate. Neural networks could also work well. Random forest the most famous.

Hands on!

https://scikit-learn.org/stable/install.html

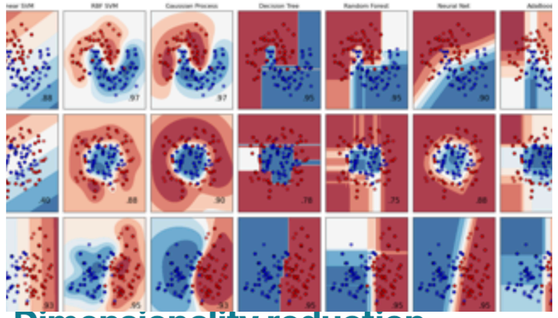
```
python3 -m venv sklearn-env
source sklearn-env/bin/activate  # activate
pip3 install -U scikit-learn

pip3 install pandas
pip3 install jupyter lab
```

Scikit learn

Classification

Identifying which category an object belongs to.

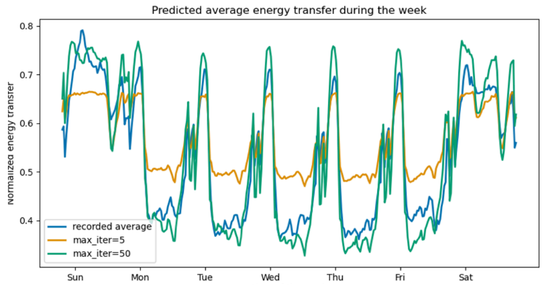
Applications: Spam detection, image recognition.
Algorithms: Gradient boosting, nearest neighbors, random forest, logistic regression, and more...



Regression

Predicting a continuous-valued attribute associated with an object.

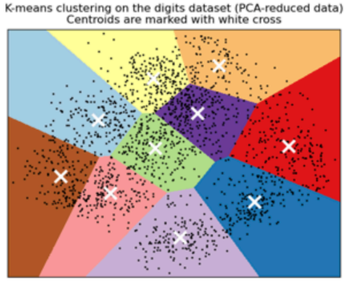
Applications: Drug response, stock prices.
Algorithms: Gradient boosting, nearest neighbors, random forest, ridge, and more...



Clustering

Automatic grouping of similar objects into sets.

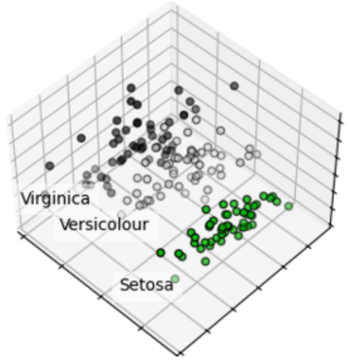
Applications: Customer segmentation, grouping experiment outcomes.
Algorithms: k-Means, HDBSCAN, hierarchical clustering, and more...



Dimensionality reduction

Reducing the number of random variables to consider.

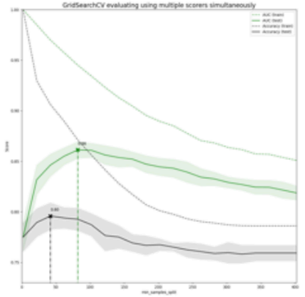
Applications: Visualization, increased efficiency.
Algorithms: PCA, feature selection, non-negative matrix factorization, and more...



Model selection

Comparing, validating and choosing parameters and models.

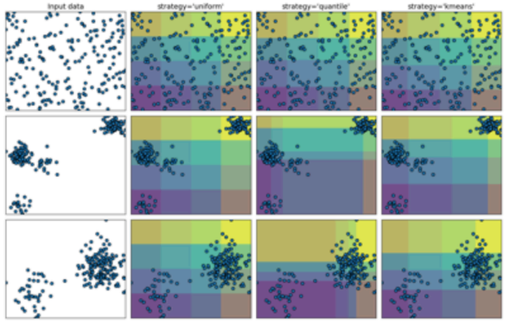
Applications: Improved accuracy via parameter tuning.
Algorithms: Grid search, cross validation, metrics, and more...



Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.
Algorithms: Preprocessing, feature extraction, and more...



https://scikit-learn.org/stable/

We will learn

- Decision trees and random forest models.
- Hyper-parameter tuning
- Feature importance

If time…

- Cross validation
- Neural networks, xgb, etc.

Problem 1: Validate the tree model, and find the best hyperparameters given the dataset.

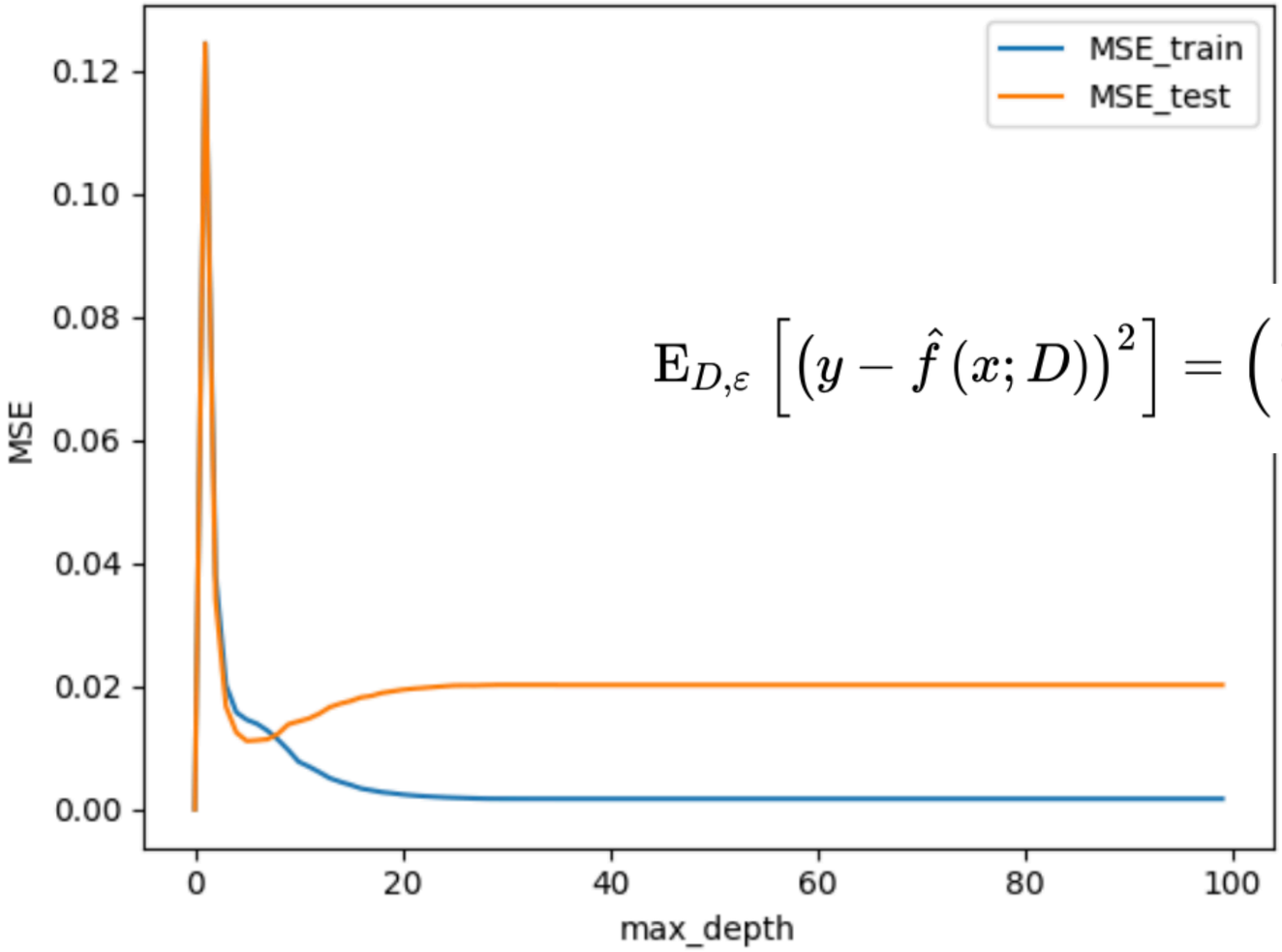
Basically, the tree model overfits if max_depth=100. For that we need a training set and a validation/test set. Find the best max_depth of the tree.

* Create a training and test set

* train the model with a grid of max_depth

* find the best max_depth

Problem 1: Create a train/test data set and validate the tree model



$$\mathrm{E}_{D,\varepsilon}\left[\left(y-\hat{f}\left(x;D\right)\right)^2\right]=\left(\mathrm{Bias}_D\left[\hat{f}\left(x;D\right)\right]\right)^2+\mathrm{Var}_D\left[\hat{f}\left(x;D\right)\right]+\sigma^2$$

Problem 2: Feature importances

The main advantage of ML models is that they can find non-linear mappings between high dimensional manifolds. One example can be the face of a person and a feeling (happy, sad, etc). So we should be complicating our problem and stop killing flies with cannons.

* Create a list of the input features that you want from the database.

* train the model

* compute the mean impurity decrease feature importance

* compute the permutation feature importance

Problem 3: What about deleting the most important feature?

* Delete the mass, e.g. `M_500` from the list of inputs.

* retrain the model

* compute the feature importance

* discuss the results

It is likely the mass is the most important feature. However, we can exclude the mass from the input dataset and then compute the feature importance again. Discuss the results.