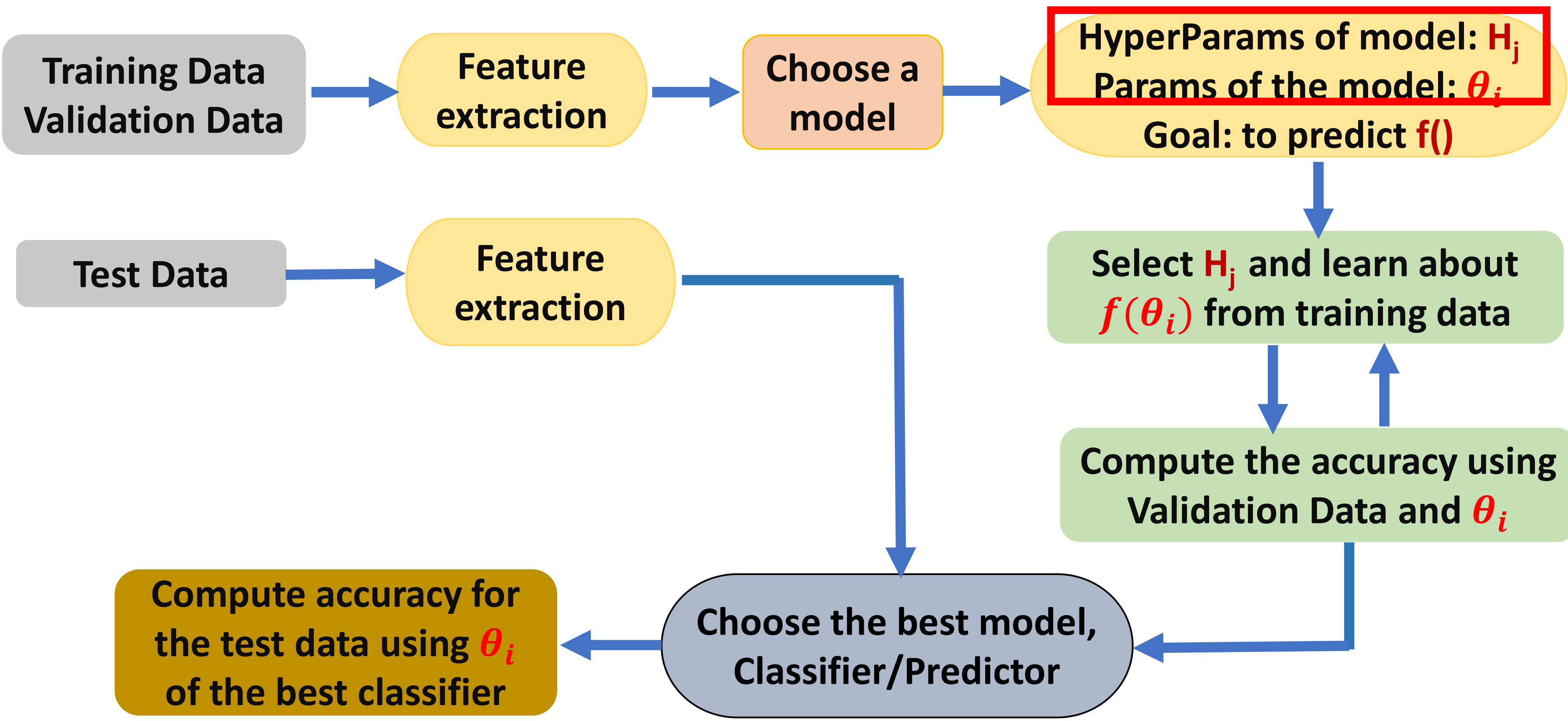
Machine Learning Paradigms

Parameters, Hyperparameters

Recap



Training Data
Validation Data
Feature extraction
Choose a model
HyperParams of model: H_j
Params of the model: θ_i
Goal: to predict f()
Test Data
Feature extraction
Select H_j and learn about $f(\theta_i)$ from training data
Compute the accuracy using Validation Data and θ_i
Compute accuracy for the test data using θ_i of the best classifier
Choose the best model, Classifier/Predictor

To Note:

Types of Parameters:

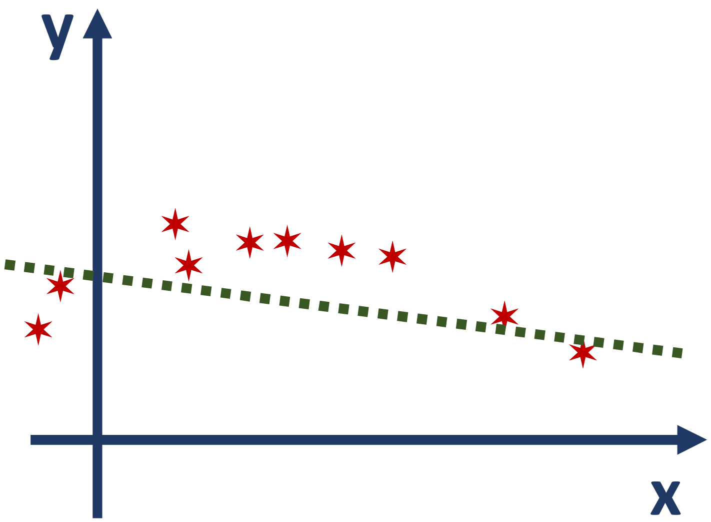
- **Learnable parameters**: Estimated from the training data
- **Hyperparameter**: Assigned by the programmer during training
- Each model has its own parameters and hyper-parameters
- Learning is modelled in the parameter space

Regression Model and its Parameters

f(x) = Linear Model

$y = \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b}$

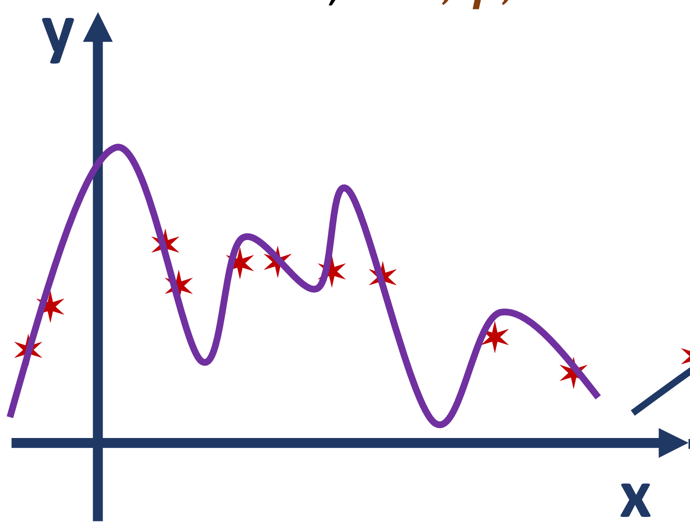
Parameter, $\boldsymbol{\theta}$: $\boldsymbol{w}, \boldsymbol{b}$



f(x) = Trigonometric Function

$y = \boldsymbol{a}\ \sin(\phi\ x) + \boldsymbol{b}$

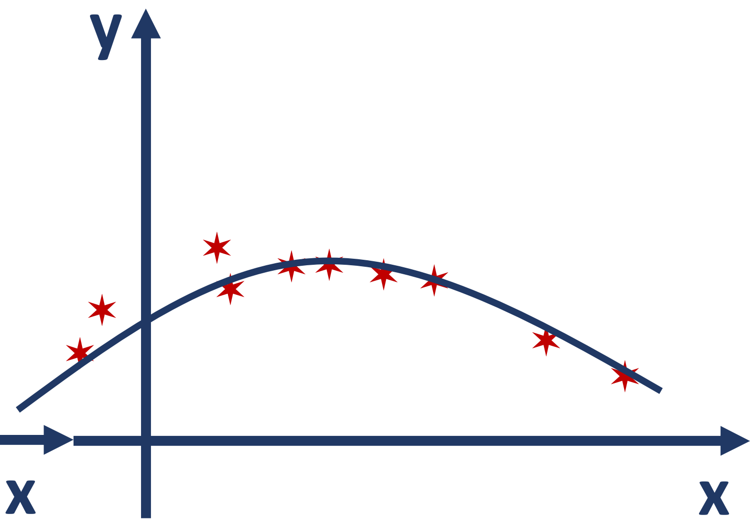
Parameter, $\boldsymbol{\theta}$: $\boldsymbol{a}, \boldsymbol{\phi}, \boldsymbol{b}$



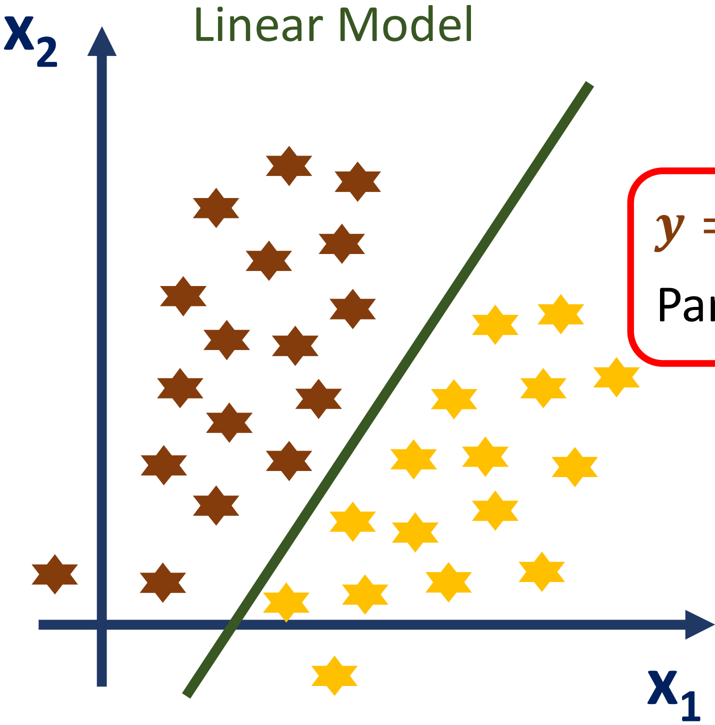
f(x) = Quadratic Model

$y = \boldsymbol{x}^T \boldsymbol{M}\ \boldsymbol{x} + \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b}$

Parameter, $\boldsymbol{\theta}$: $\boldsymbol{M}, \boldsymbol{w}, \boldsymbol{b}$



Classification Model and its Parameters



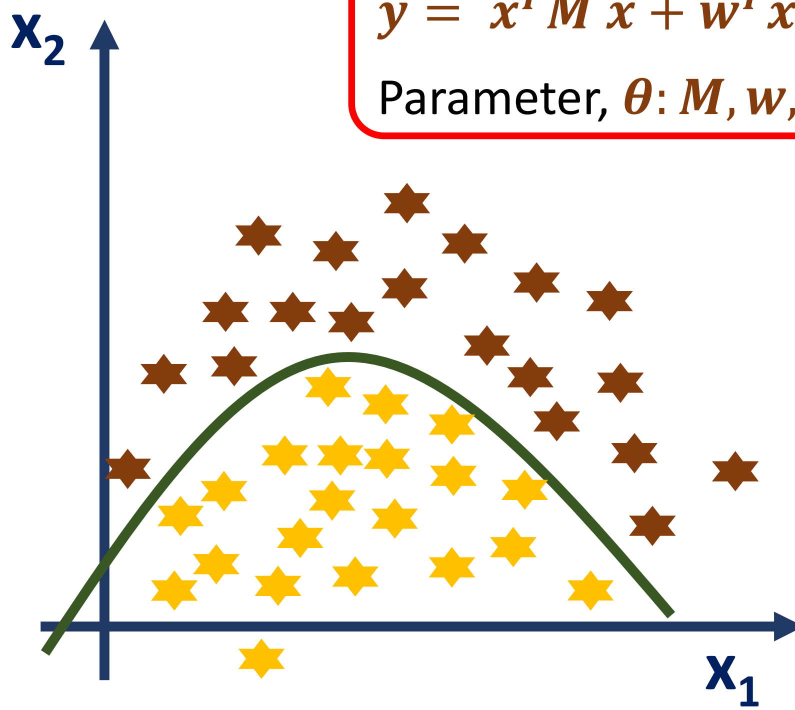
$y = \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b}$

Parameter, $\boldsymbol{\theta}$: $\boldsymbol{w}, \boldsymbol{b}$

Quadratic Model

$y = \boldsymbol{x}^T \boldsymbol{M}\, \boldsymbol{x} + \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{b}$

Parameter, $\boldsymbol{\theta}$: $\boldsymbol{M}, \boldsymbol{w}, \boldsymbol{b}$

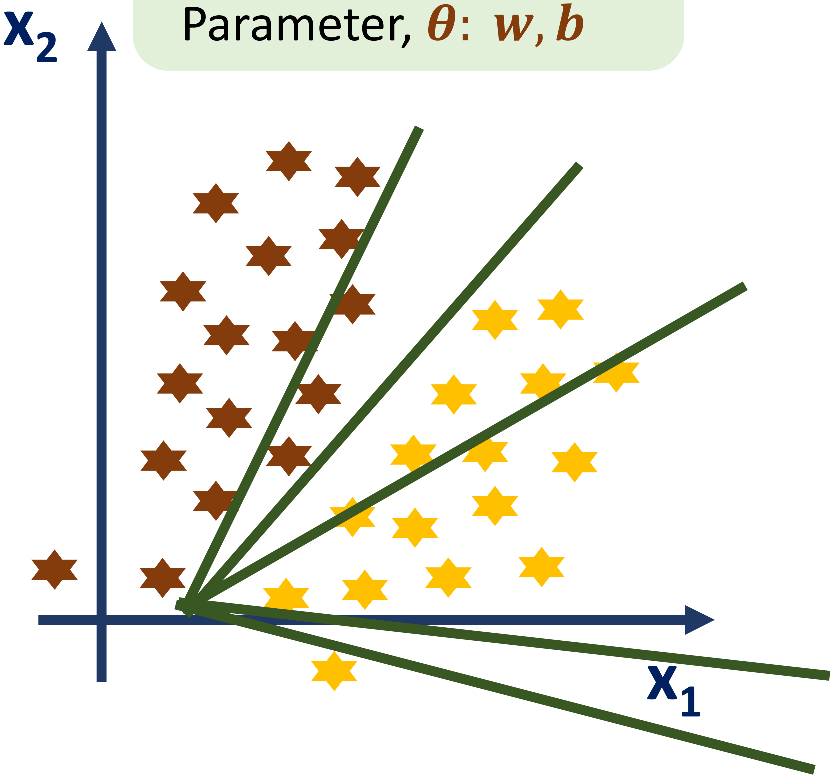


Parameter Space

Linear Model

$y = w^T x + b$

Parameter, θ: w, b



- During training, the parameter vector, θ, moves in the parameter space.

❖ $Step\ 1: w = \begin{bmatrix} 0.4 \\ 1 \end{bmatrix}, b = -1$

❖ $Step\ 2: w = \begin{bmatrix} 0.05 \\ 1 \end{bmatrix}, b = -0.1$

❖ $Step\ 3: w = \begin{bmatrix} -0.4 \\ 1 \end{bmatrix}, b = 0.5$

❖ $Step\ 4: w = \begin{bmatrix} -0.7 \\ 1 \end{bmatrix}, b = 0.7$

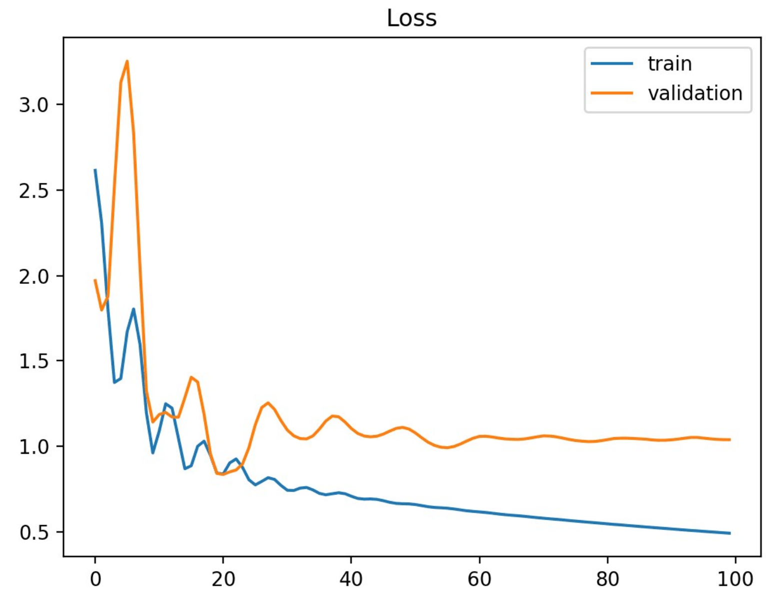
❖ $Step\ 5: w = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, b = 0.9$

Model Hyperparameters

- A model hyperparameter is a parameter whose value is set before the model starts training. They cannot be learned by fitting the model to the data

Model hyperparameters in different models:

- Number of neighbours in KNN
- Number of hidden layers in a Neural Network
- Number of neurons per hidden layer in a Neural Network
- Number of clusters (k) in k means clustering



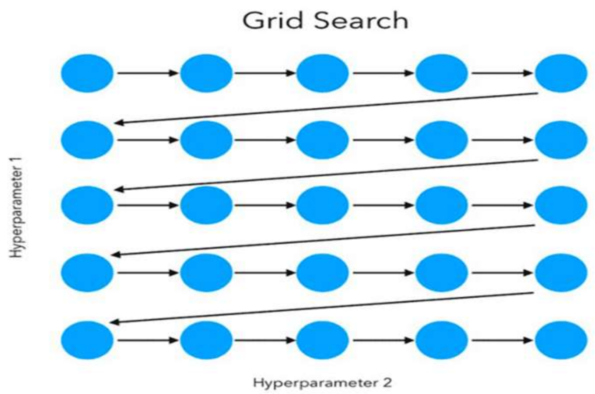
Hyperparameter Optimization

- Hyperparameters are important because they directly control the behaviour of the training algorithm and have a significant impact on the performance of the model being trained.
- "A good choice of hyperparameters can really make an algorithm shine".
- The process of finding most optimal hyperparameters in machine learning is called hyperparameter optimization.
- Common algorithms include:
  1. Grid Search
  2. Random Search
  3. Bayesian Optimization

Grid Search

- It performs an exhaustive search by evaluating any candidates' combinations.
- It results in an unfeasible computing cost, so grid search is an option only when the number of candidates is limited enough.

search_space = {'param_1': [val_1, val_2, val_3],
'param_2': [val_1, val_2, val_3],
'param_3':['str_val_1', 'str_val_2']}

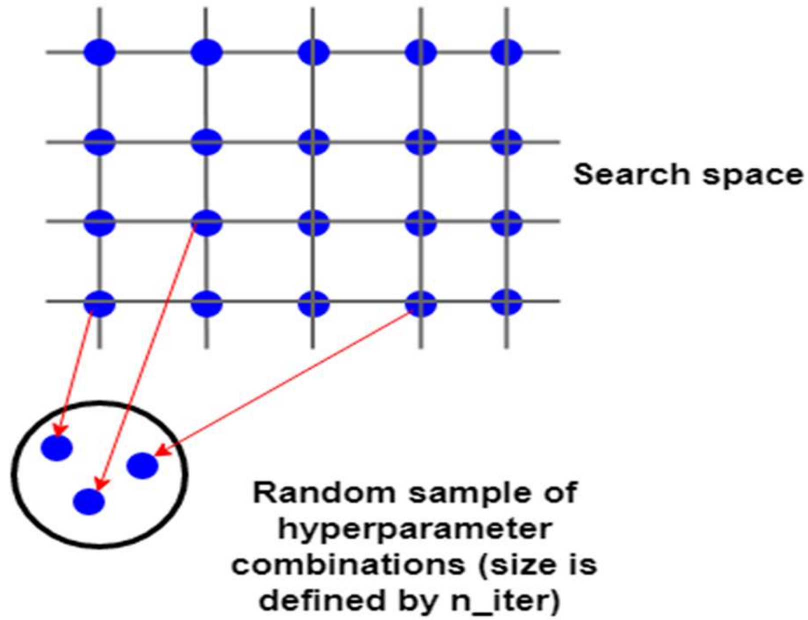


https://www.pyimagesearch.com

Random Search

- In random search the selection of the values to evaluate is completely random and the required time decreases significantly
- The chances of finding the optimal parameter are comparatively higher in random search

```
param_dist = {“param_1": sp_randint (1, 11),
              “param_2": [3, None],
              “param_3": sp_randint(2, 5),
              “param_4": sp_randint(1, 20),
              “param_5": sp_randint(11, 17),
             }
```



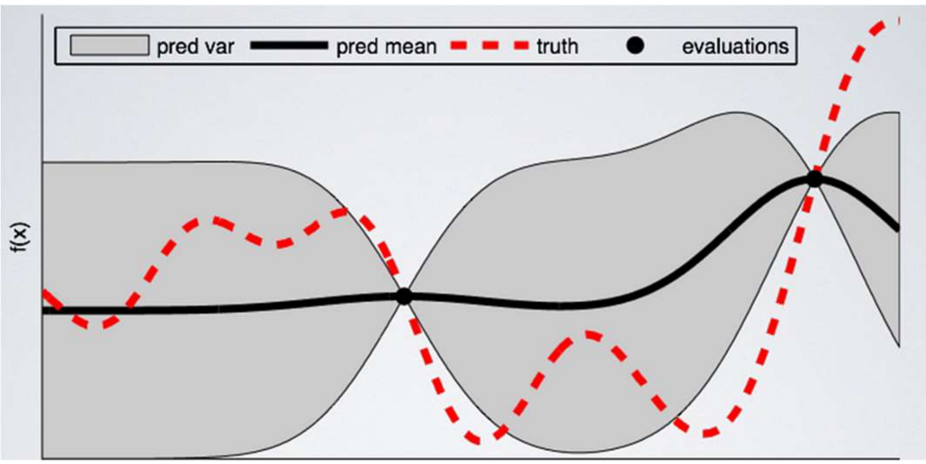
[Random Search](#)

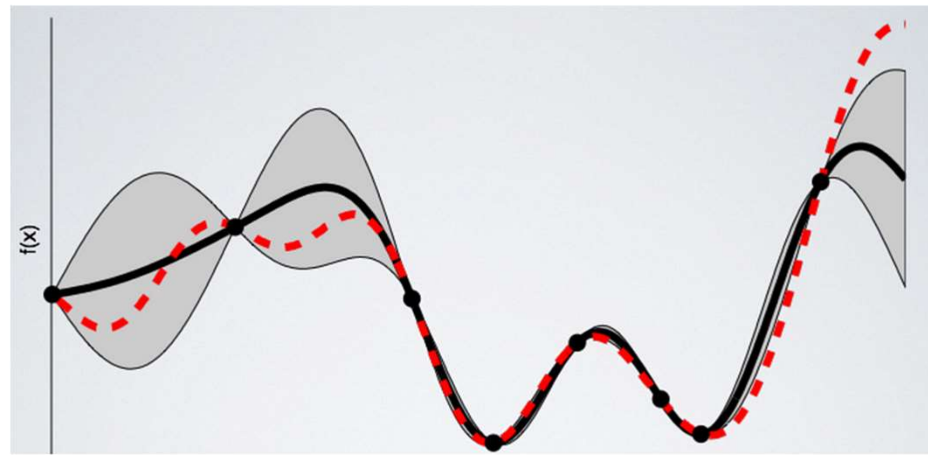
Note: Grid and random search are completely uninformed by past evaluations

Bayesian Optimization

- Hyperparameter optimization functions f(x) are not known analytically, and expensive to evaluate
- Construct a posterior distribution of functions (**surrogate function**) that best approximates the mapping of input examples to an output score
- The algorithm using **acquisition function** detects which regions in the hyperparameter space are more interesting to explore and which are not
- Additional points are added, and the surrogate function is re-evaluated.
- After a defined number of iterations, the algorithm stops and returns the optimum tuple
    - surrogate function max does not change
    - Or f is exhausted

Bayesian Search



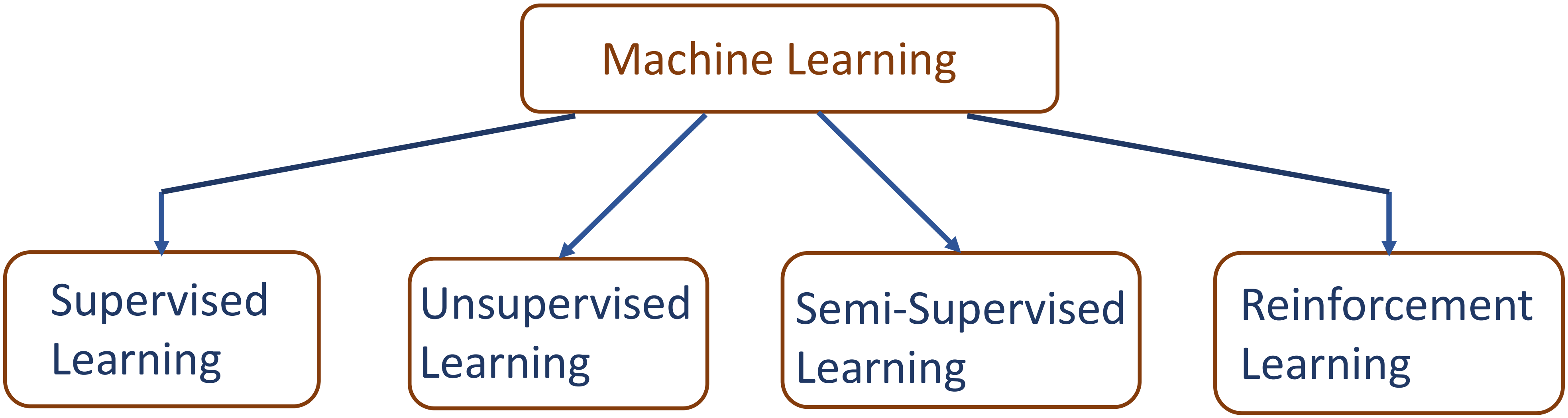


https://towardsdatascience.com/

Branches of Machine Learning

Unsupervised Learning, Semi Supervised Learning, Reinforcement Learning

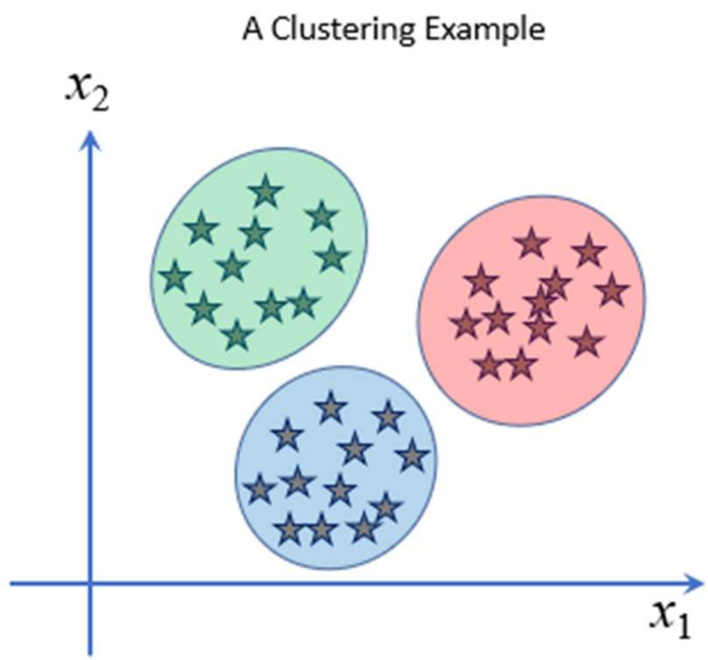
Categorization of ML Based on Learning



- **Labeled data**
- **Direct feedback**
- **Prediction**

Unsupervised Learning

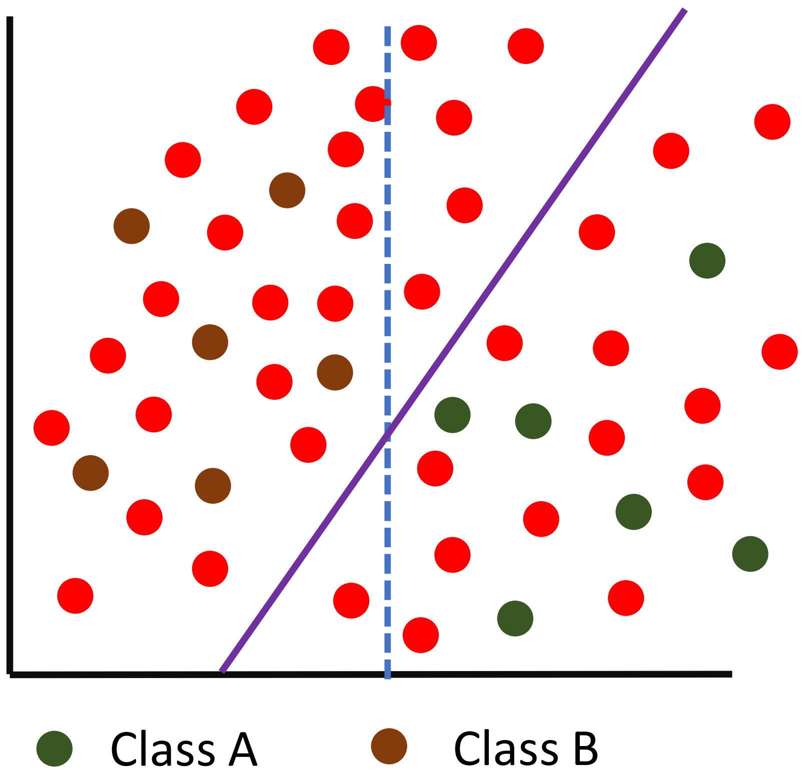
- It is trained with unlabeled data, only with feature vectors, $x_i,\ i = 1..n$
- It lets the model discover and learn on their own, i.e., it works on its own to discover pattern and information
- For a grouping of feature vectors, x_i, it learns a representation e_i, that is appropriate for the problem



Semi-Supervised Learning

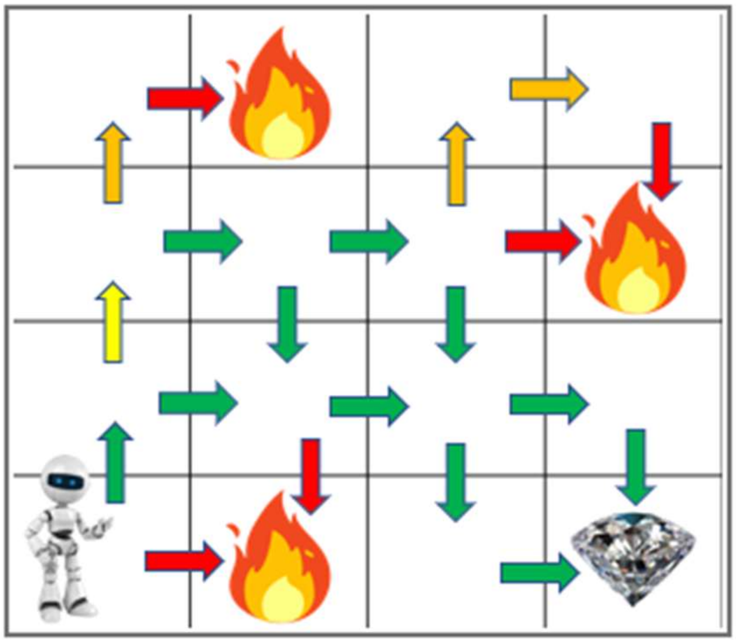
- Can we use unlabeled data to augment a small labeled sample to improve learning?

1. Use small labeled sample to learn initial rules training set ***T = {x_i},*** and unlabeled set ***U = {u_j}***
2. We first train on ***T*** and find ***f()***
3. Get the predictions ***P = f(U)***
4. If ***$P_i > \alpha$***, we then add ***(u_i, f(u_i))*** to ***T***
5. The modified training set is then retrained
6. Repeat the process until the model converges



Reinforcement Learning

- It is a type of machine learning where an agent learns to behave in an environment by trial and error.
- The agent is rewarded for taking actions that lead to desired outcomes, and penalized for taking actions that lead to undesired outcomes.
- The agent learns to take actions that maximize its expected reward over time.
- Often used for tasks such as:
  - Game playing, Robotics, Resource management, Finance



Categorization of ML Based on Learning

Machine Learning

Supervised Learning

- **Labeled data**
- **Direct feedback**
- **Prediction**

Unsupervised Learning

- **No Labels**
- **No feedback**
- **Find hidden structure**

Semi-Supervised Learning

- **Partial Label**
- **Combine SL & UL**
- **Self-training**

Reinforcement Learning

- **Decision process**
- **Reward system**
- **Learn series of actions**

Understanding Data

Data Quality

Data Quality

- The quality and quantity of training data is the most important aspect that decides the quality of the ML solution

- The data may be limited by several issues:

  - Outliers
  - Missing feature values
  - Limited quantity