SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

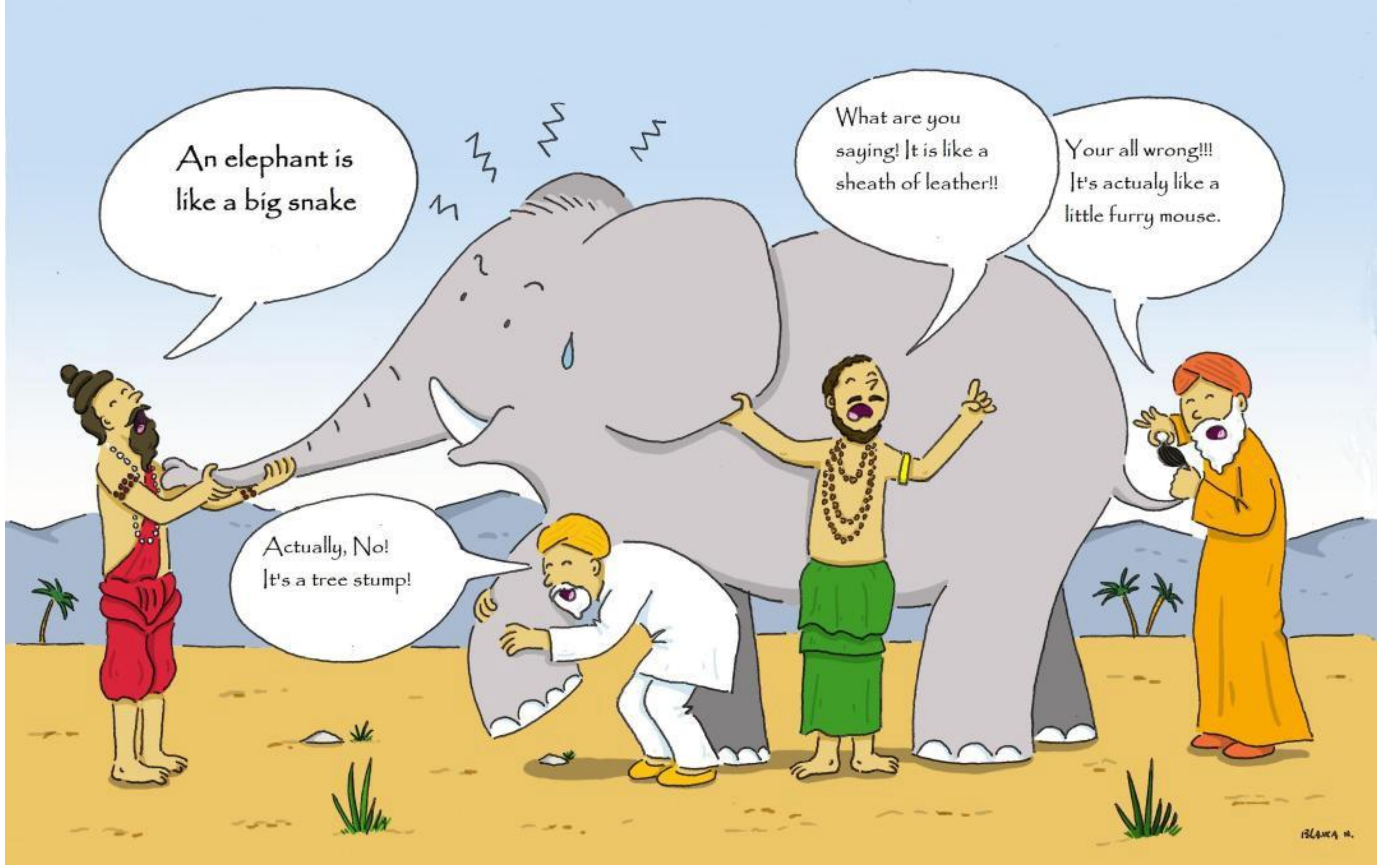
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME : 19CS501 Introduction to Machine Learning

III YEAR /V SEMESTER

Unit 2- SUPERVISED LEARNING

Topic : Ensemble Methods, Bagging, Boosting



An elephant is like a big snake
Actually, No! It's a tree stump!
What are you saying! It is like a sheath of leather!!
Your all wrong!!! It's actualy like a little furry mouse.
BLANKA R.

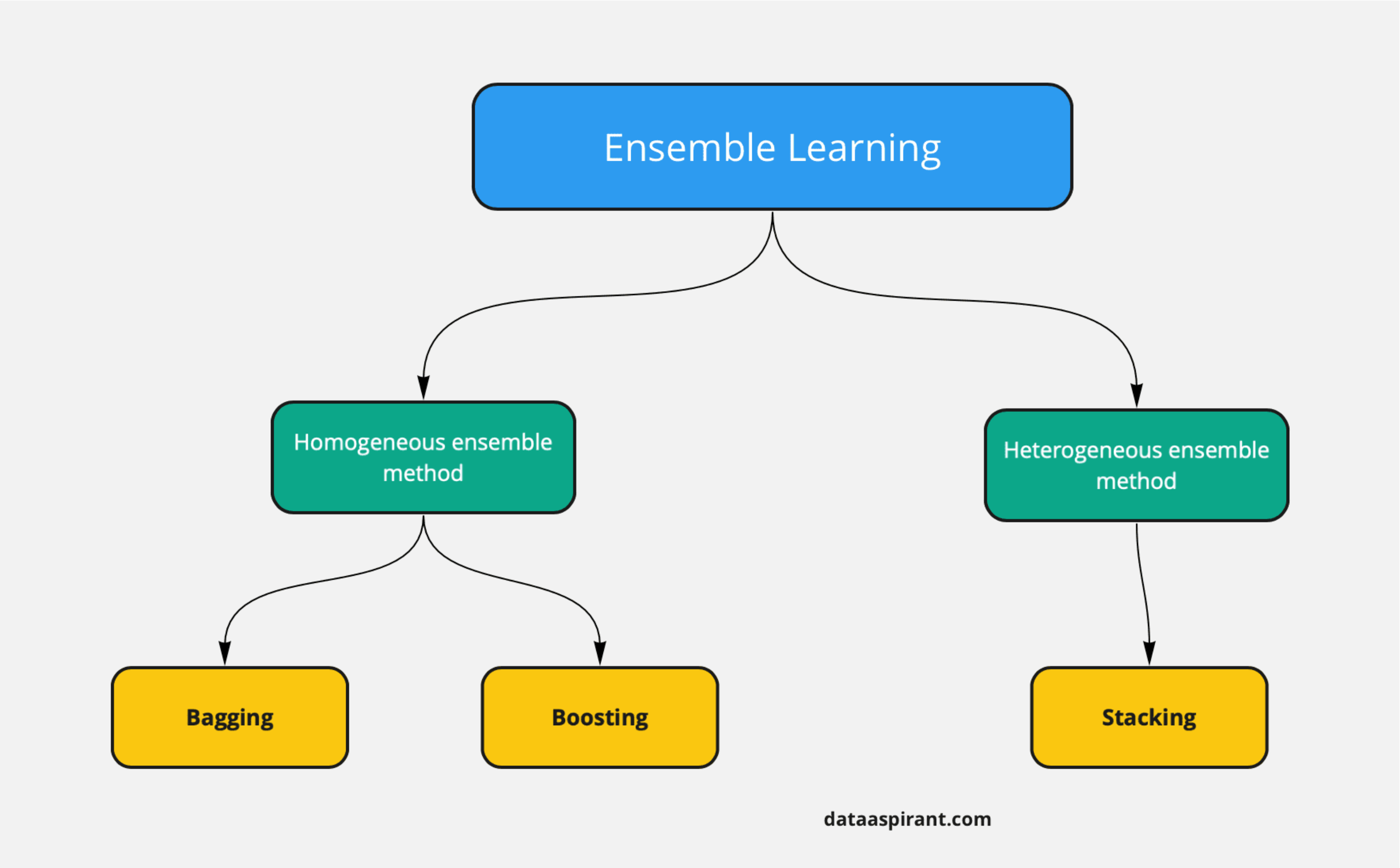
Introduction

- Roughly, ensemble learning methods, that often trust the top rankings of many machine learning competitions (including Kaggle's competitions), are based on the hypothesis that combining multiple models together can often produce a much more powerful model.
- We will discuss some well known notions such as boostrapping, bagging, random forest, boosting, stacking and many others that are the basis of ensemble learning. In order to make the link between all these methods as clear as possible

What are ensemble methods?

- ❑ Ensemble learning is a machine learning paradigm where multiple models (often called “weak learners”) are trained to solve the same problem and combined to get better results.

- ❑ The main hypothesis is that when weak models are correctly combined we can obtain more accurate and/or robust models.



Single weak learner

❑ In machine learning, no matter if we are facing a classification or a regression problem, the choice of the model is extremely important to have any chance to obtain good results. This choice can depend on many variables of the problem: quantity of data, dimensionality of the space, distribution hypothesis

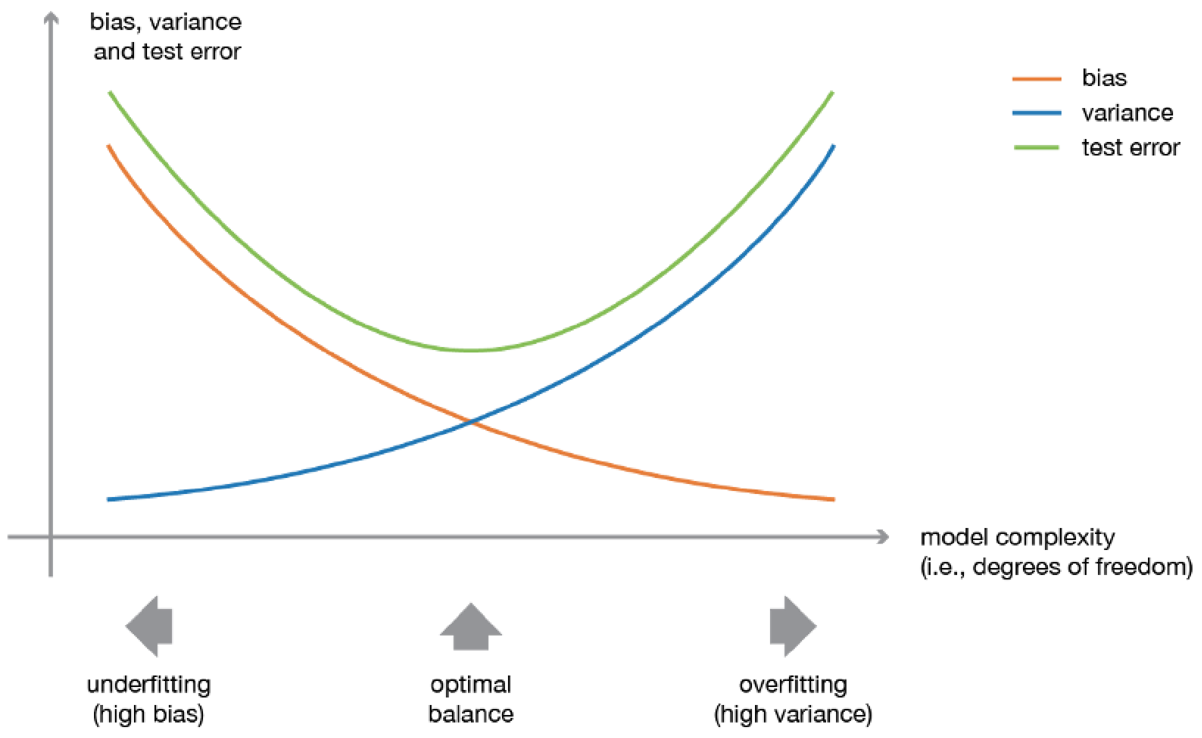
❑ A low bias and a low variance, although they most often vary in opposite directions, are the two most fundamental features expected for a model.

❑ Indeed, to be able to “solve” a problem, we want our model to have enough degrees of freedom to resolve the underlying complexity of the data we are working with, but we also want it to have not too much degrees of freedom to avoid high variance and be more robust. This is the well known bias-variance tradeoff.

Single weak learner

- ❑ In ensemble learning theory, we call weak learners (or base models) models that can be used as building blocks for designing more complex models by combining several of them.

- ❑ Most of the time, these basics models perform not so well by themselves either because they have a high bias (low degree of freedom models, for example) or because they have too much variance to be robust (high degree of freedom models, for example).

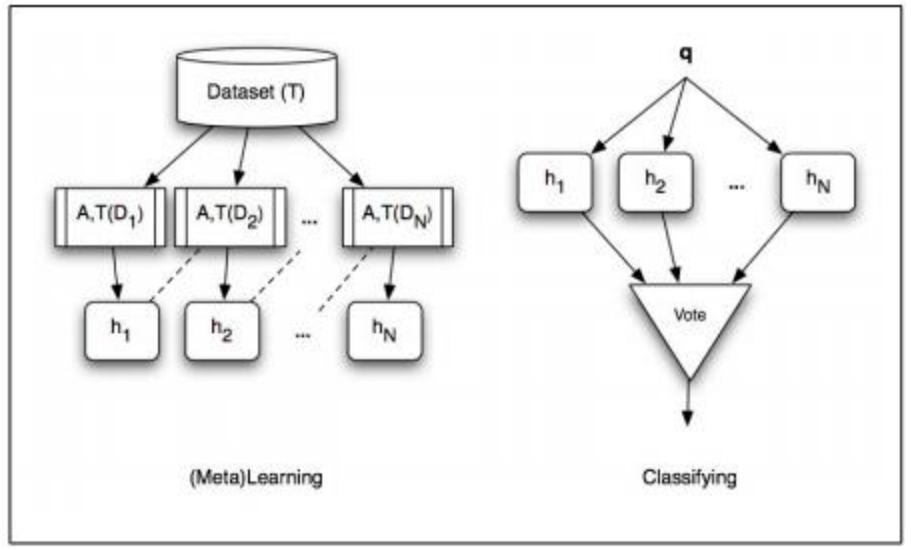


Combine weak learners

- ❑ The ensemble model we obtain is then said to be “homogeneous”. However, there also exist some methods that use different type of base learning algorithms: some heterogeneous weak learners are then combined into an “heterogeneous ensembles model”.

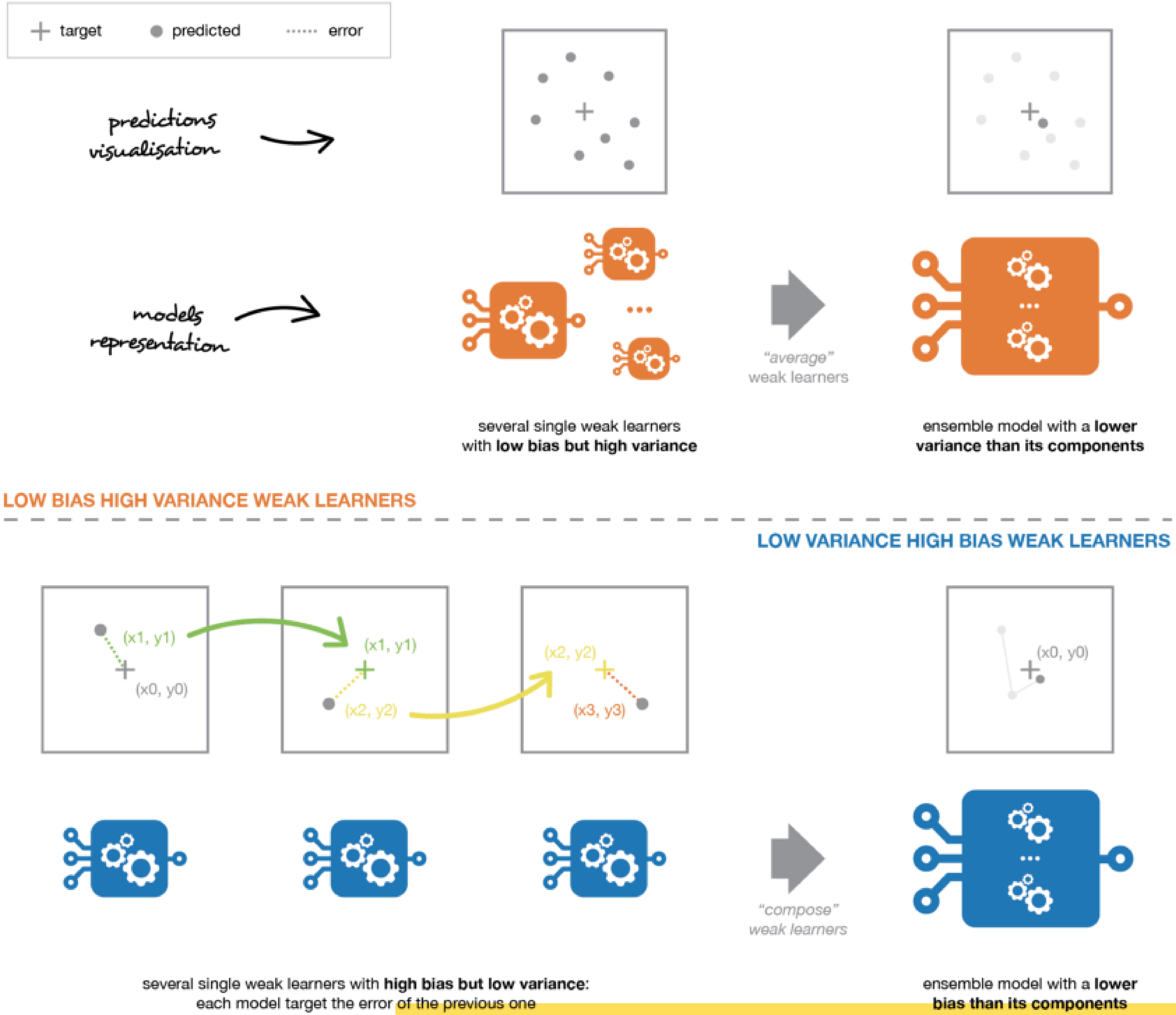
- ❑ One important point is that our choice of weak learners should be coherent with the way we aggregate these models.

- ❑ If we choose base models with low bias but high variance, it should be with an aggregating method that tends to reduce variance whereas if we choose base models with low variance but high bias, it should be with an aggregating method that tends to reduce bias.



meta-algorithms

- ✓ bagging, that often considers homogeneous weak learners, learns them independently from each other in parallel and combines them following some kind of deterministic averaging process
- ✓ boosting, that often considers homogeneous weak learners, learns them sequentially in a very adaptative way (a base model depends on the previous ones) and combines them following a deterministic strategy
- ✓ stacking, that often considers heterogeneous weak learners, learns them in parallel and combines them by training a meta-model to output a prediction based on the different weak models predictions



several single weak learners with **low bias but high variance**

ensemble model with a **lower variance than its components**

LOW BIAS HIGH VARIANCE WEAK LEARNERS

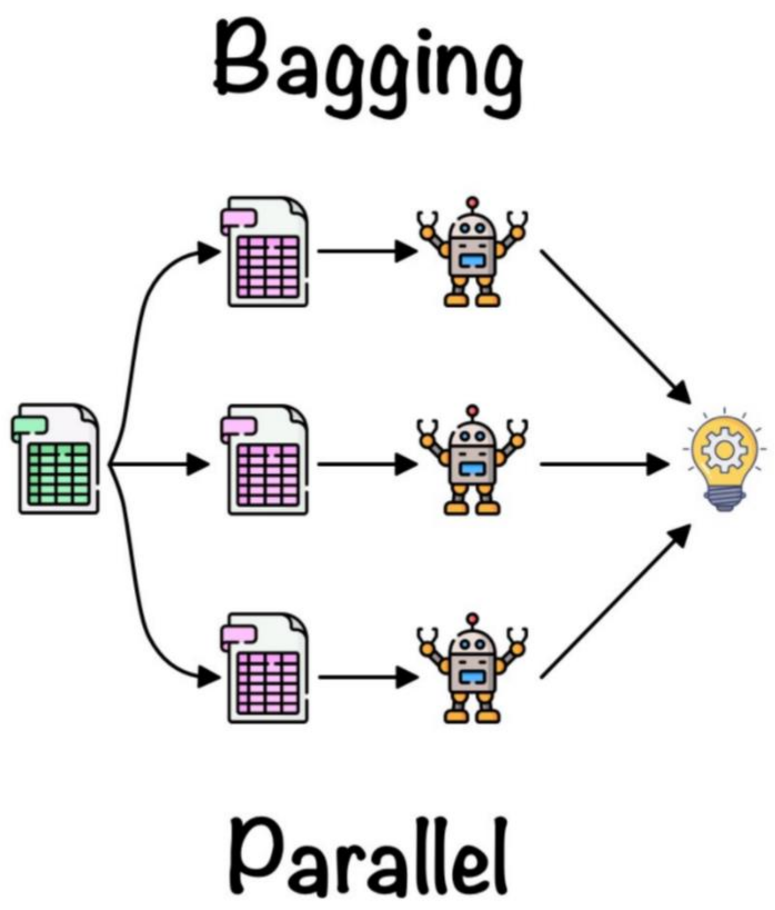
LOW VARIANCE HIGH BIAS WEAK LEARNERS

several single weak learners with **high bias but low variance**: each model target the error of the previous one

ensemble model with a **lower bias than its components**

Focus on bagging

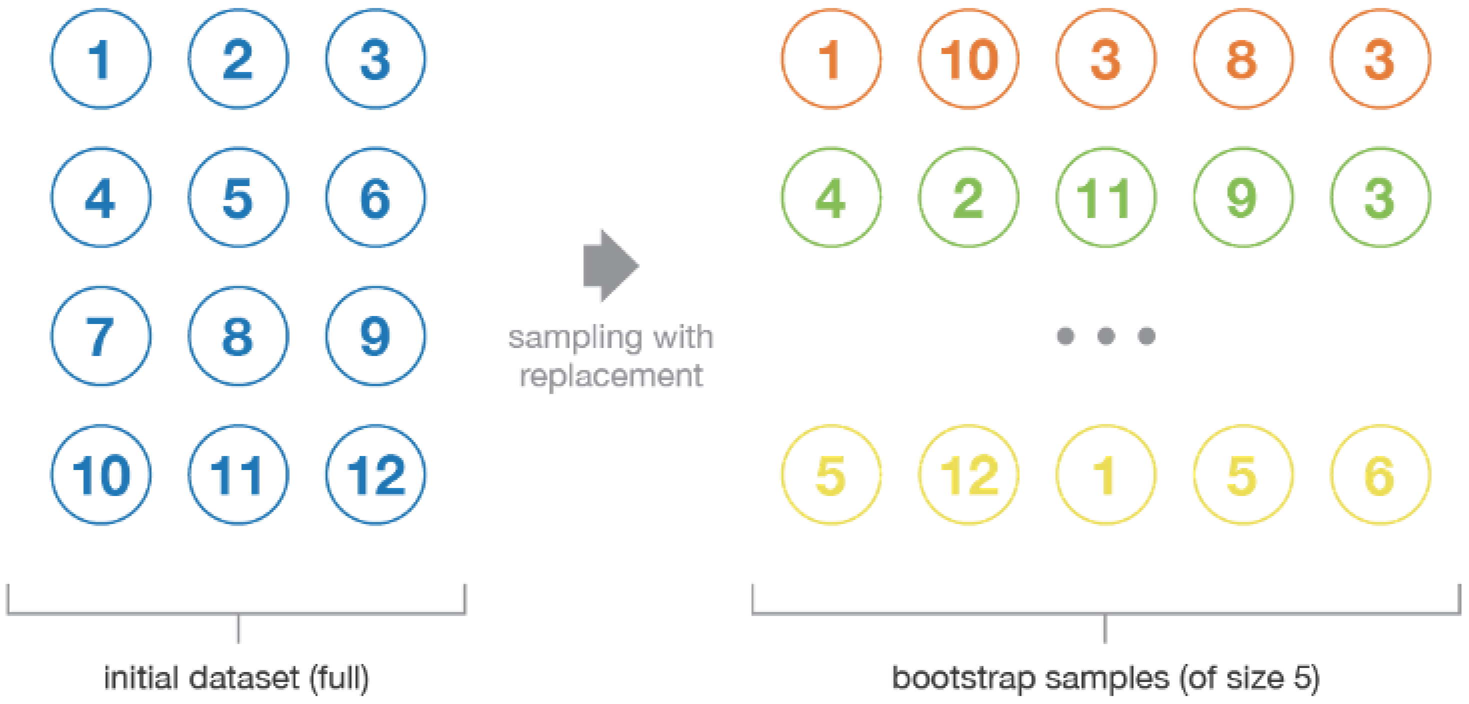
- ✓ In parallel methods we fit the different considered learners independently from each others and, so, it is possible to train them concurrently.
- ✓ The most famous such approach is “bagging” (standing for “bootstrap aggregating”) that aims at producing an ensemble model that is more robust than the individual models composing it



Bootstrapping

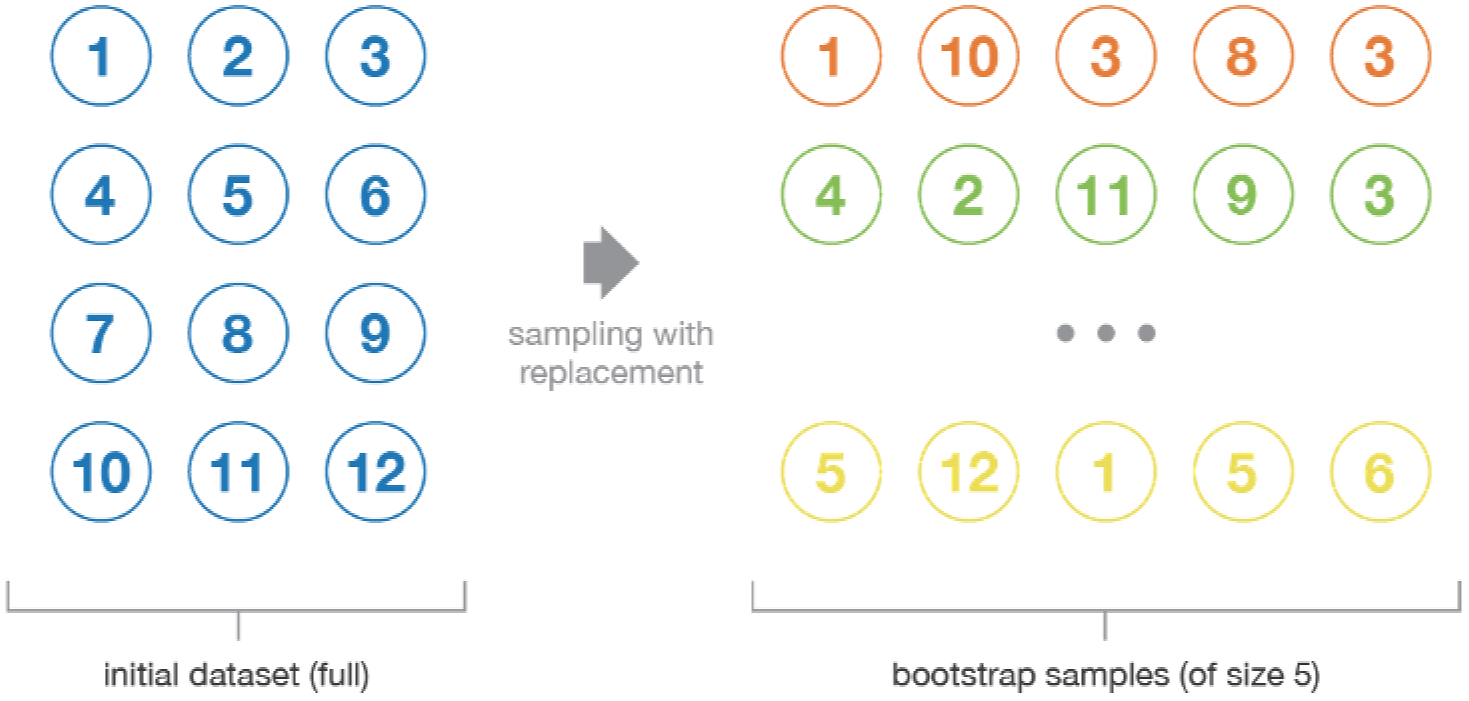
- ✓ This statistical technique consists in generating samples of size B (called bootstrap samples) from an initial dataset of size N by randomly drawing with replacement B observations.

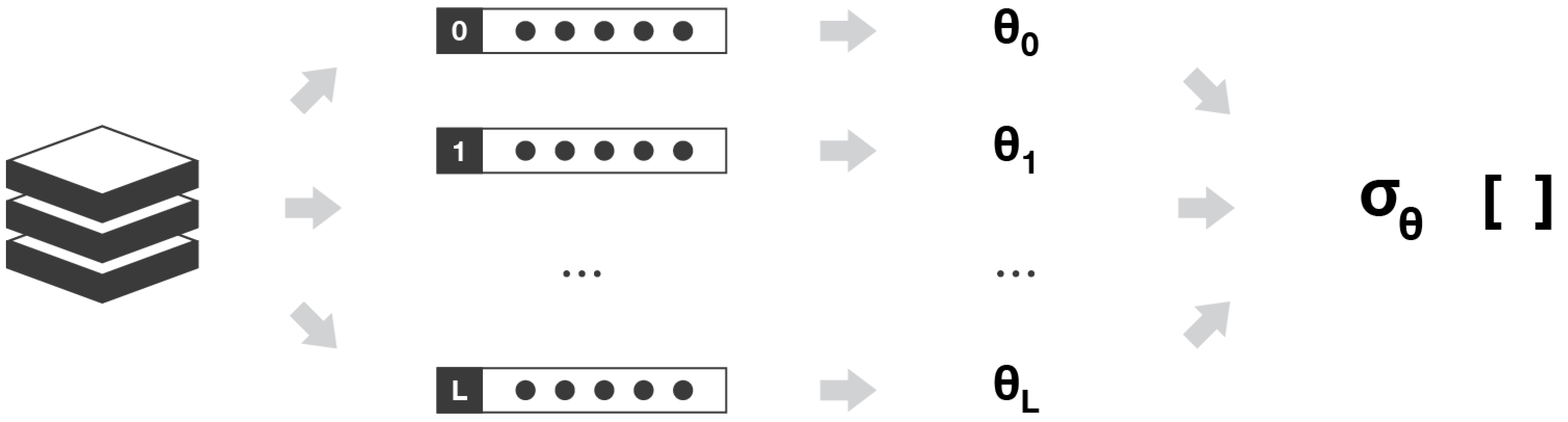
- ✓ Bootstrap samples are often used, for example, to evaluate variance or confidence intervals of a statistical estimators. By definition, a statistical estimator is a function of some observations and, so, a random variable with variance coming from these observations. In order to estimate the variance of such an estimator, we need to evaluate it on several independent samples drawn from the distribution of interest



Bootstrapping-good statistical properties

- ✓ first approximation, they can be seen as being drawn both directly from the true underlying (and often unknown) data distribution and independently from each others.
- ✓ The hypothesis that have to be verified to make this approximation valid are twofold.
  - ✓First, the size N of the initial dataset should be large enough to capture most of the complexity of the underlying distribution so that sampling from the dataset is a good approximation of sampling from the real distribution (representativity).
  - ✓ Second, the size N of the dataset should be large enough compared to the size B of the bootstrap samples so that samples are not too much correlated (independence)



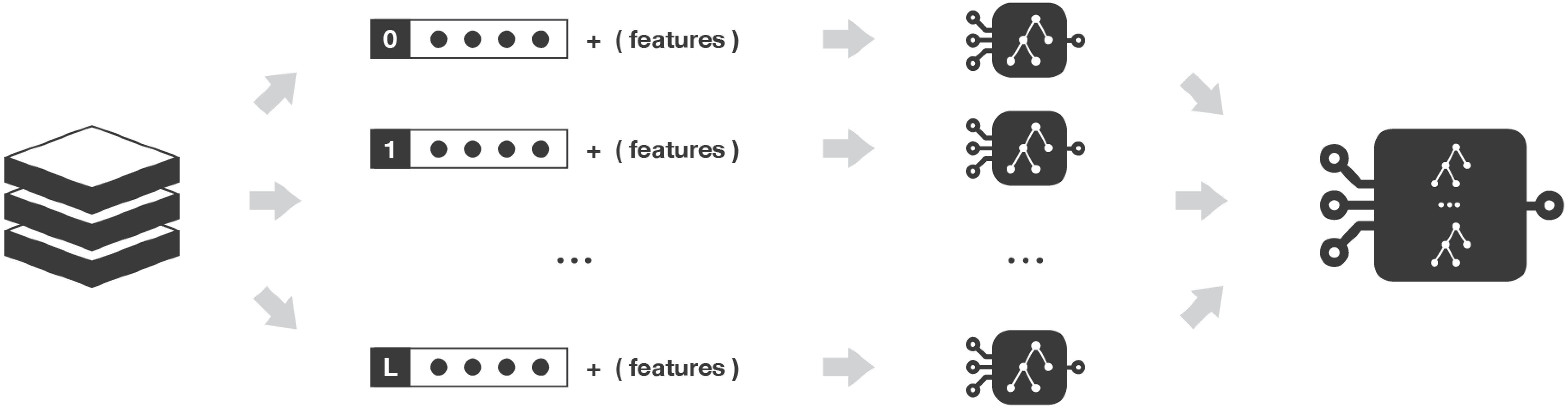


initial dataset

L bootstrap samples

estimator of interest evaluated for each bootstrap sample

variance and confidence intervals computed based on the L realisations of the estimator



initial dataset

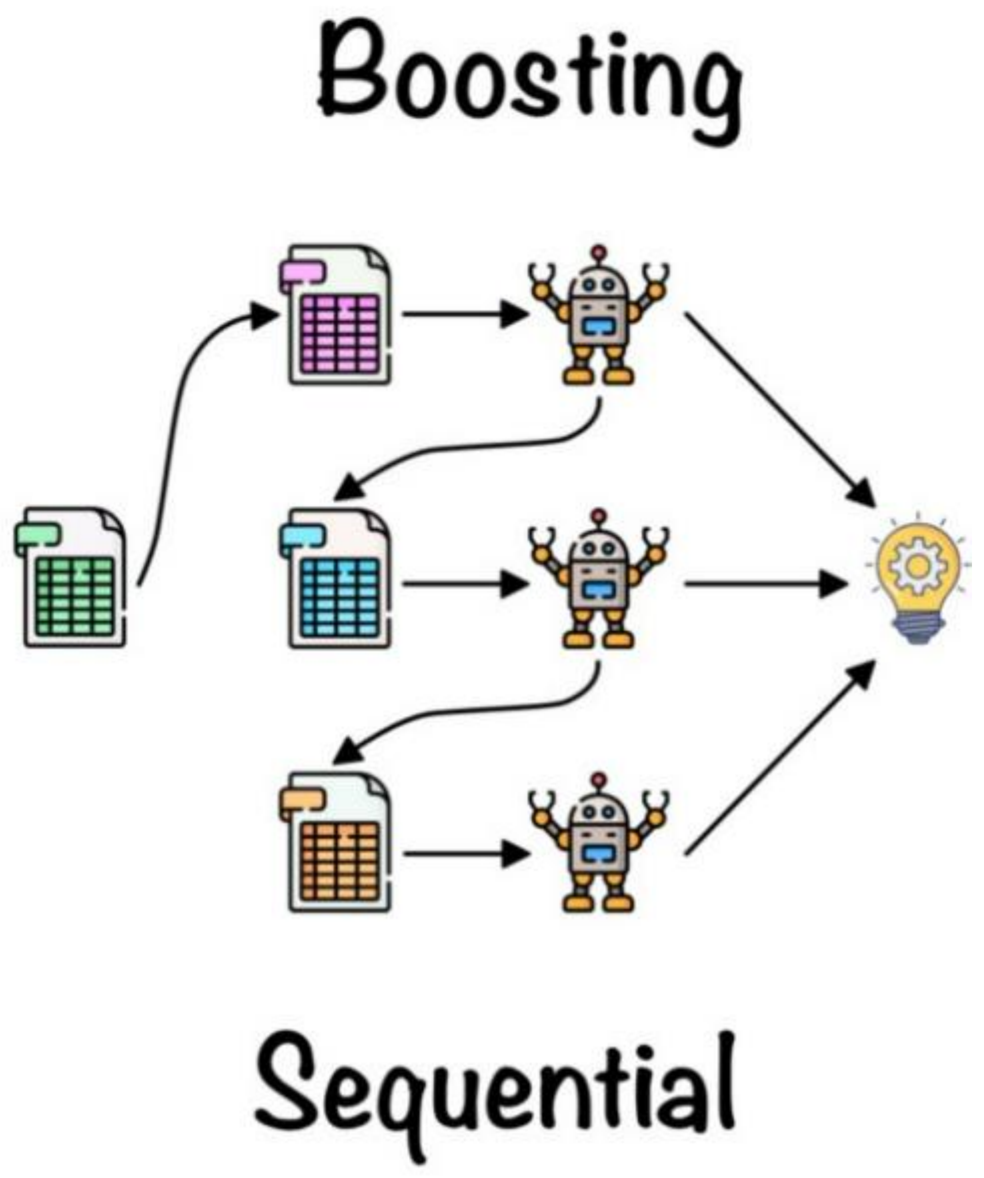
bootstrap samples + selected features

deep trees fitted on each bootstrap sample and considering only selected features

random forest
(kind of average of the trees)

Boosting

- ✓ Boosting is an Ensemble Learning technique that, like bagging, makes use of a set of base learners to improve the stability and effectiveness of a ML model.
- ✓ The idea behind a boosting architecture is the generation of sequential hypotheses, where each hypothesis tries to improve or correct the mistakes made in the previous one .
- ✓ The central idea of boosting is the implementation of homogeneous ML algorithms in a sequential way, where each of these ML algorithms tries to improve the stability of the model by focusing on the errors made by the previous ML algorithm.
- ✓ The way in which the errors of each base learner is considered to be improved with the next base learner in the sequence, is the key differentiator between all variations of the boosting technique



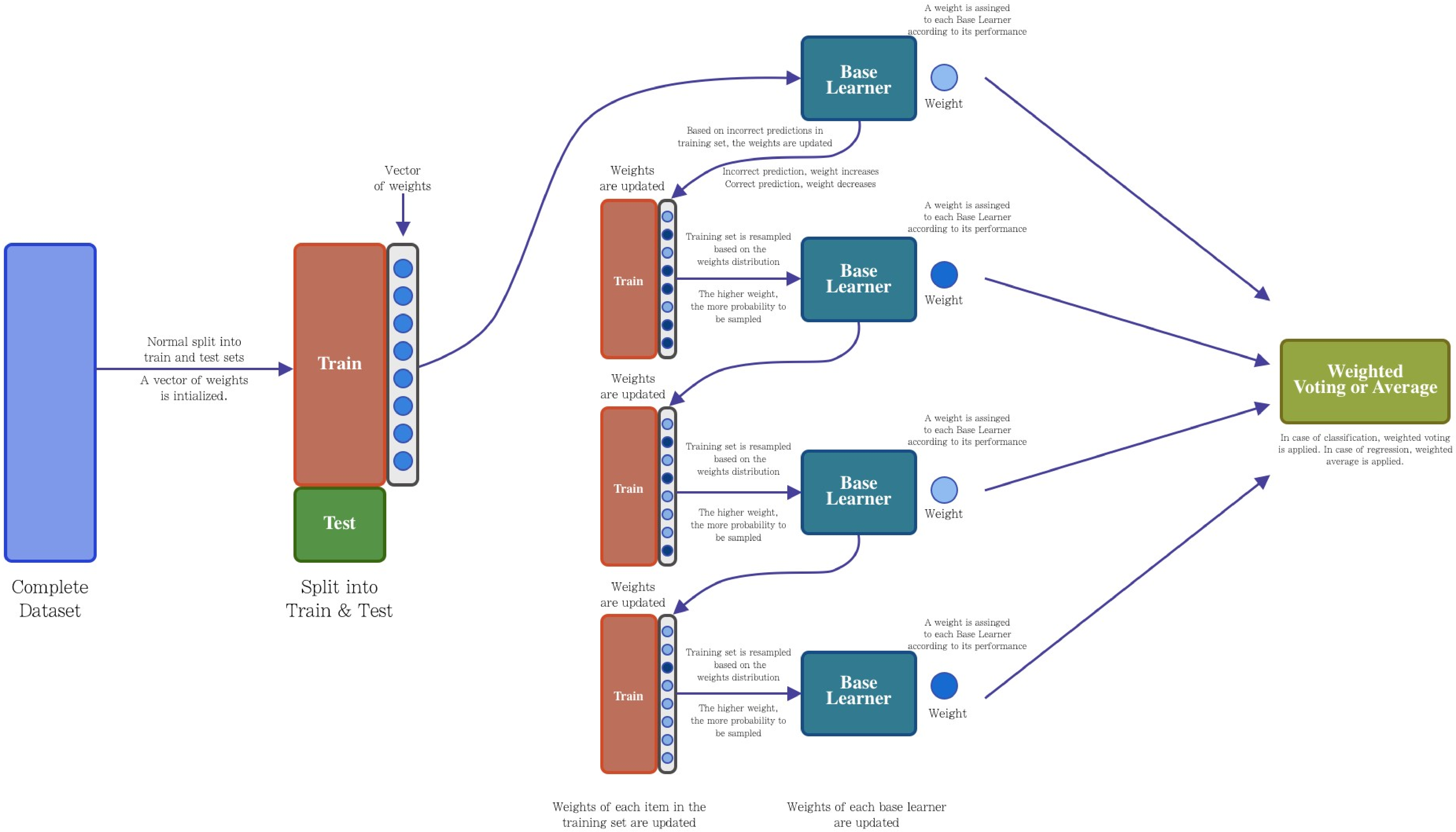
AdaBoost

❑ AdaBoost is an algorithm based on the boosting technique, it was introduced in 1995 by Freund and Schapire .

❑ AdaBoost implements a vector of weights to penalize those samples that were incorrectly inferred (by increasing the weight) and reward those that were correctly inferred (by decreasing the weight).

❑ Updating this weight vector will generate a distribution where it will be more likely to extract those samples with higher weight (that is, those that were incorrectly inferred), this sample will be introduced to the next base learner in the sequence.

❑ This will be repeated until a stop criterion is met

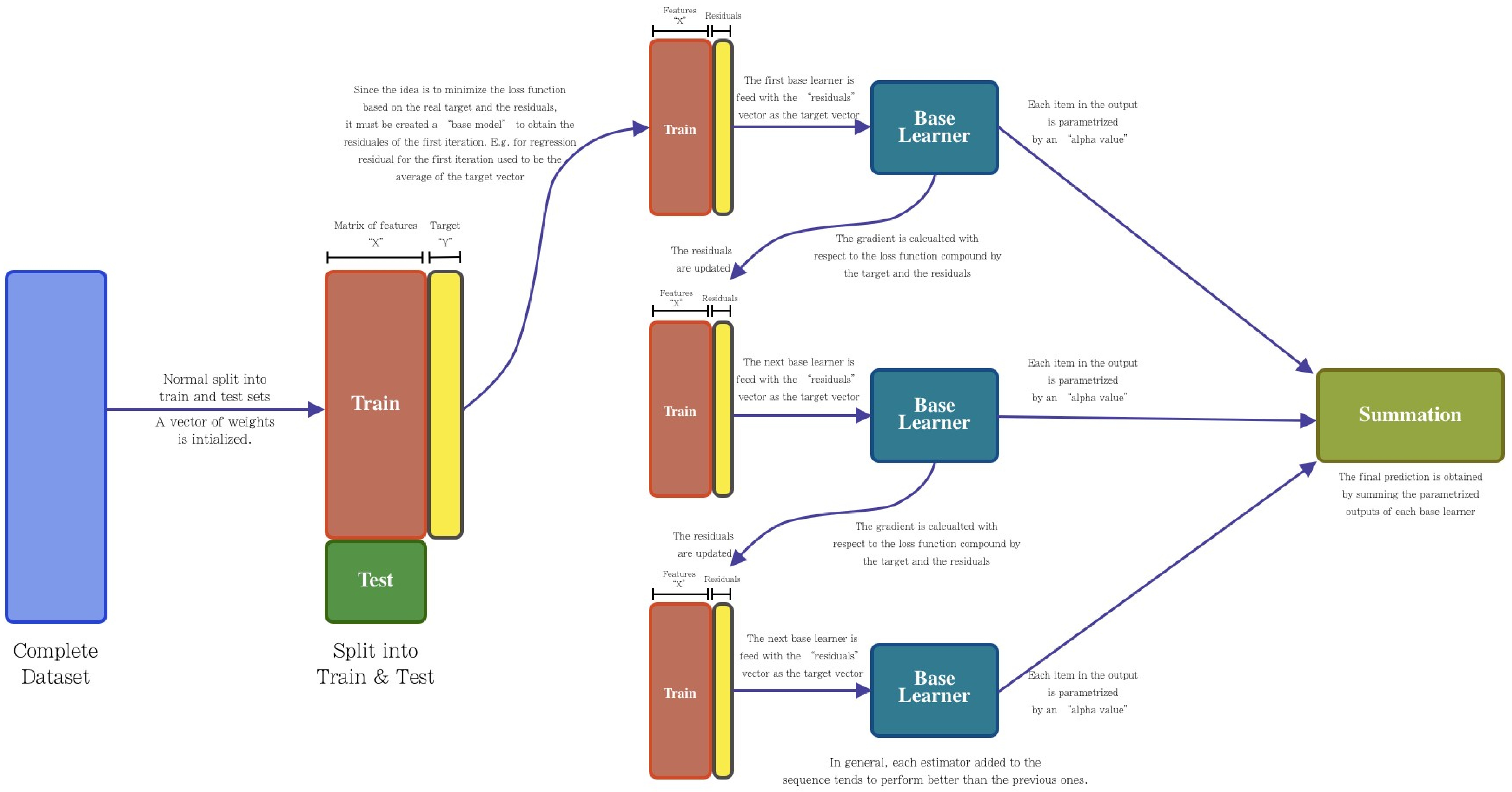

Complete Dataset
Normal split into train and test sets
A vector of weights is intialized.
Vector of weights
Train
Test
Split into Train & Test
Base Learner
Weight
A weight is assinged to each Base Learner according to its performance
Based on incorrect predictions in training set, the weights are updated
Weights are updated
Incorrect prediction, weight increases Correct prediction, weight decreases
Train
Training set is resampled based on the weights distribution
The higher weight, the more probability to be sampled
Base Learner
Weight
A weight is assinged to each Base Learner according to its performance
Weights are updated
Train
Training set is resampled based on the weights distribution
The higher weight, the more probability to be sampled
Base Learner
Weight
A weight is assinged to each Base Learner according to its performance
Weights are updated
Train
Training set is resampled based on the weights distribution
The higher weight, the more probability to be sampled
Base Learner
Weight
Weighted Voting or Average
In case of classification, weighted voting is applied. In case of regression, weighted average is applied.
Weights of each item in the training set are updated
Weights of each base learner are updated

Gradient Boosting

- ❑ The Gradient Boosting method does not implement a vector of weights like AdaBoost does. As its name implies, it implements the calculation of the gradient for the optimization of a given loss function.

- ❑ The core idea of Gradient Boosting is based on minimizing the residuals of each learner base in a sequential way, this minimization is carried out through the calculation of the gradient applied to a specific loss function (either for classification or regression).

- ❑ Then each base learner added to the sequence will minimize the residuals determined by the previous base learner



Normal split into train and test sets

A vector of weights is intialized.

Since the idea is to minimize the loss function based on the real target and the residuals, it must be created a “base model” to obtain the residuales of the first iteration. E.g. for regression residual for the first iteration used to be the average of the target vector

The first base learner is feed with the “residuals” vector as the target vector

Each item in the output is parametrized by an “alpha value”

The gradient is calcualted with respect to the loss function compound by the target and the residuals

The residuals are updated

The next base learner is feed with the “residuals” vector as the target vector

Each item in the output is parametrized by an “alpha value”

The gradient is calcualted with respect to the loss function compound by the target and the residuals

The residuals are updated

The next base learner is feed with the “residuals” vector as the target vector

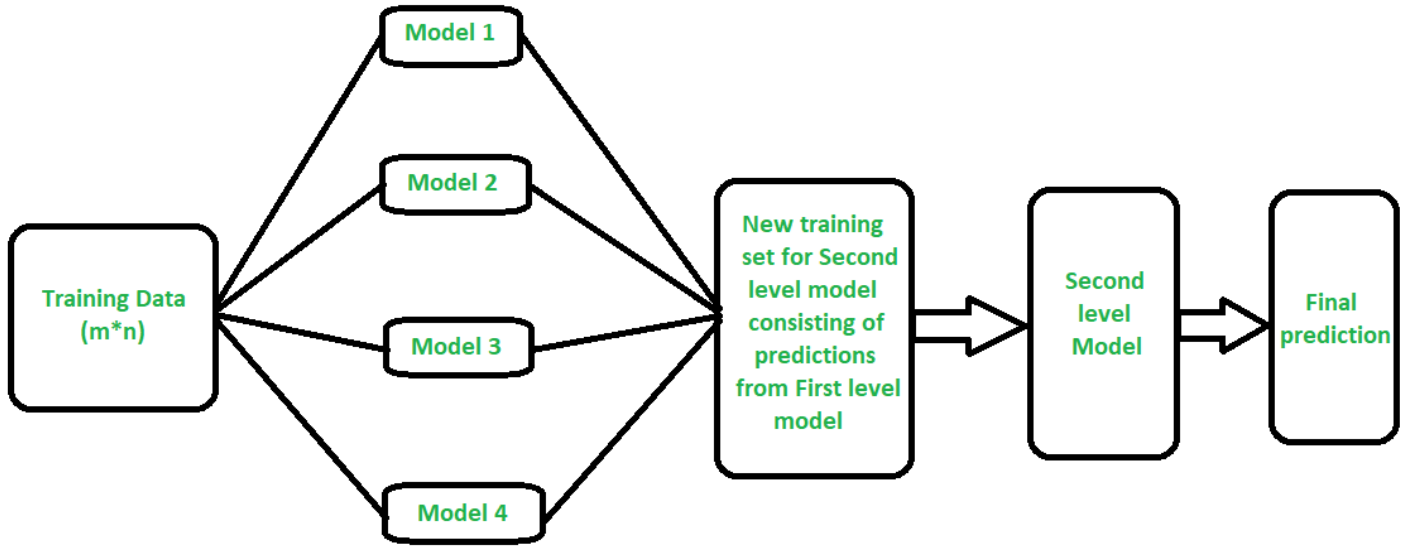
Each item in the output is parametrized by an “alpha value”

The final prediction is obtained by summing the parametrized outputs of each base learner

In general, each estimator added to the sequence tends to perform better than the previous ones.

Overview of stacking

- Stacking mainly differ from bagging and boosting on two points. First stacking often considers heterogeneous weak learners (different learning algorithms are combined) whereas bagging and boosting consider mainly homogeneous weak learners. Second, stacking learns to combine the base models using a meta-model whereas bagging and boosting combine weak learners following deterministic algorithms.


Training Data (m*n)
Model 1
Model 2
Model 3
Model 4
New training set for Second level model consisting of predictions from First level model
Second level Model
Final prediction

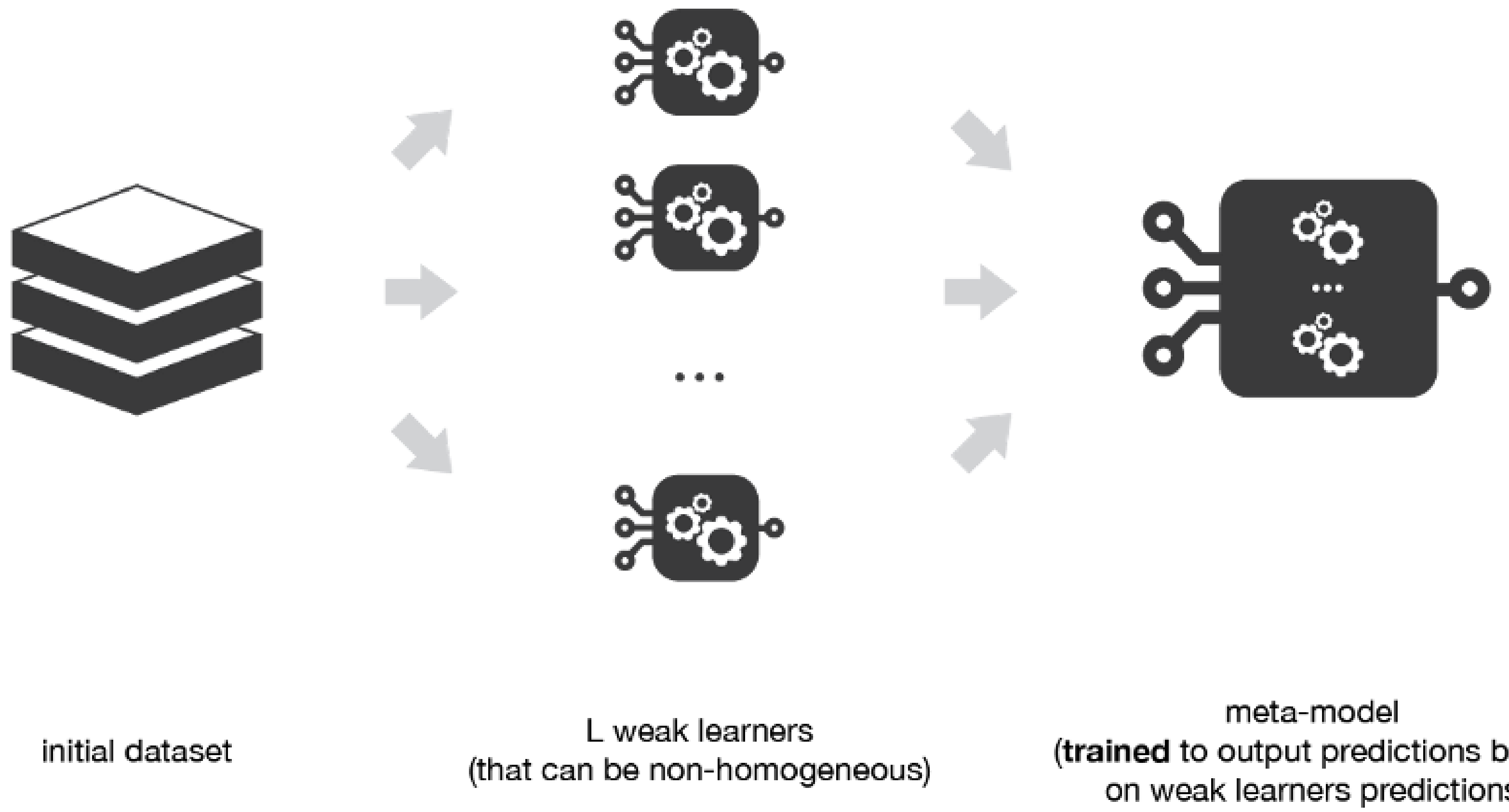
Stacking

- the idea of stacking is to learn several different weak learners and combine them by training a meta-model to output predictions based on the multiple predictions returned by these weak models. So, we need to define two things in order to build our stacking model: the L learners we want to fit and the meta-model that combines them.

- For example, for a classification problem, we can choose as weak learners a KNN classifier, a logistic regression and a SVM, and decide to learn a neural network as meta-model. Then, the neural network will take as inputs the outputs of our three weak learners and will learn to return final predictions based on it.

Beta Prior Distribution

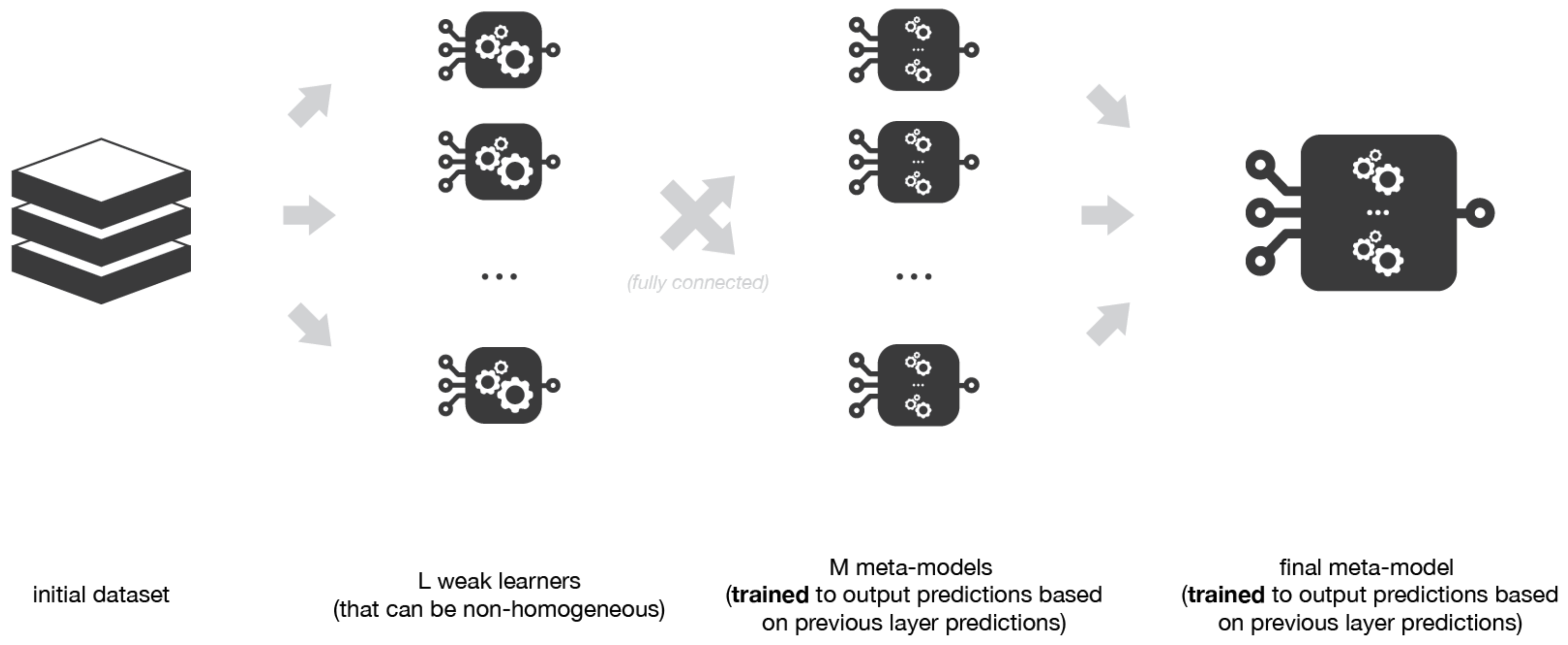
- split the training data in two folds
- choose L weak learners and fit them to data of the first fold
- for each of the L weak learners, make predictions for observations in the second fold
- fit the meta-model on the second fold, using predictions made by the weak learners as inputs


initial dataset
L weak learners
(that can be non-homogeneous)
meta-model
(**trained** to output predictions based on weak learners predictions)

Multi-levels Stacking

- A possible extension of stacking is multi-level stacking. It consists in doing stacking with multiple layers. As an example, let’s consider a 3-levels stacking. In the first level (layer), we fit the L weak learners that have been chosen.

- Then, in the second level, instead of fitting a single meta-model on the weak models predictions (as it was described in the previous subsection) we fit M such meta-models. Finally, in the third level we fit a last meta-model that takes as inputs the predictions returned by the M meta-models of the previous level


...
(fully connected)
...
initial dataset
L weak learners
(that can be non-homogeneous)
M meta-models
(trained to output predictions based
on previous layer predictions)
final meta-model
(trained to output predictions based
on previous layer predictions)

Assessment

- Python Implementation of the Ensemble methods

REFERENCES

1. Tom M. Mitchell, “Machine Learning”, McGraw-Hill Education (India) Private Limited, 2513.
2. Trevor Hastie, Robert Tibshirani, Jerome Friedman, “The Elements of Statistical Learning: Data Mining, Inference, and Prediction”, Springer; Second Edition, 2509.

THANK YOU