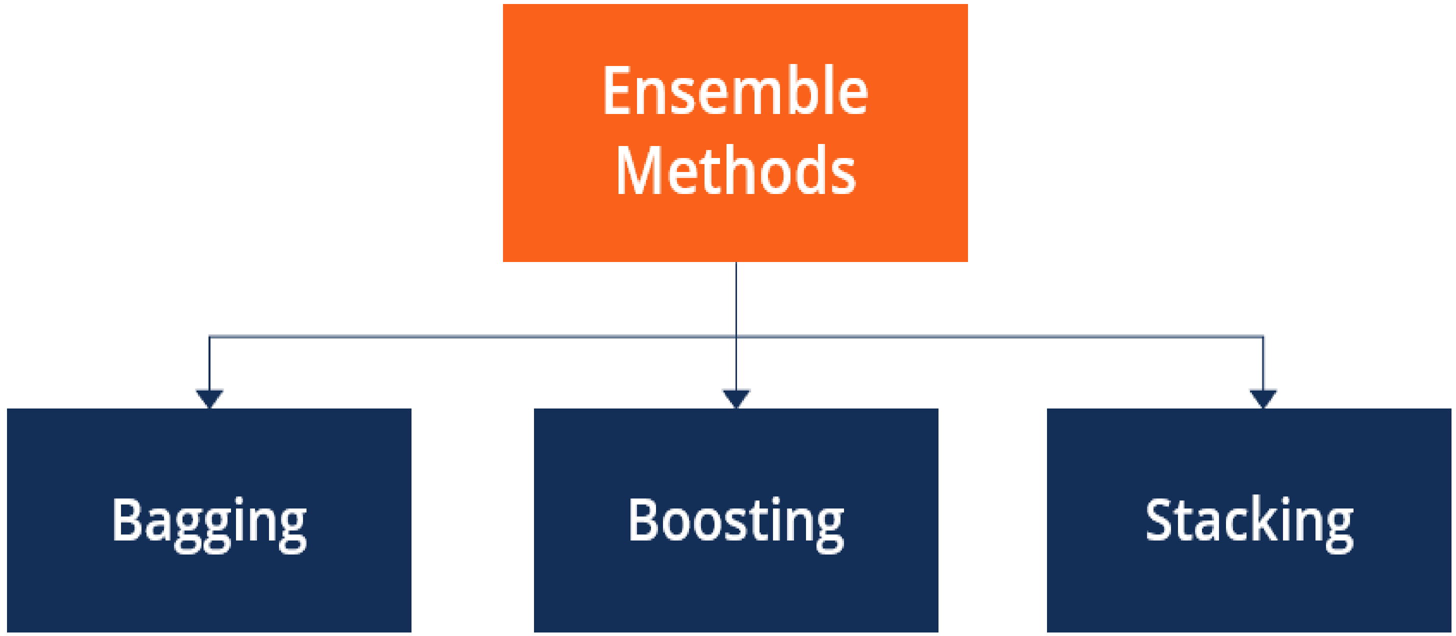
INT247
Machine Learning Foundations

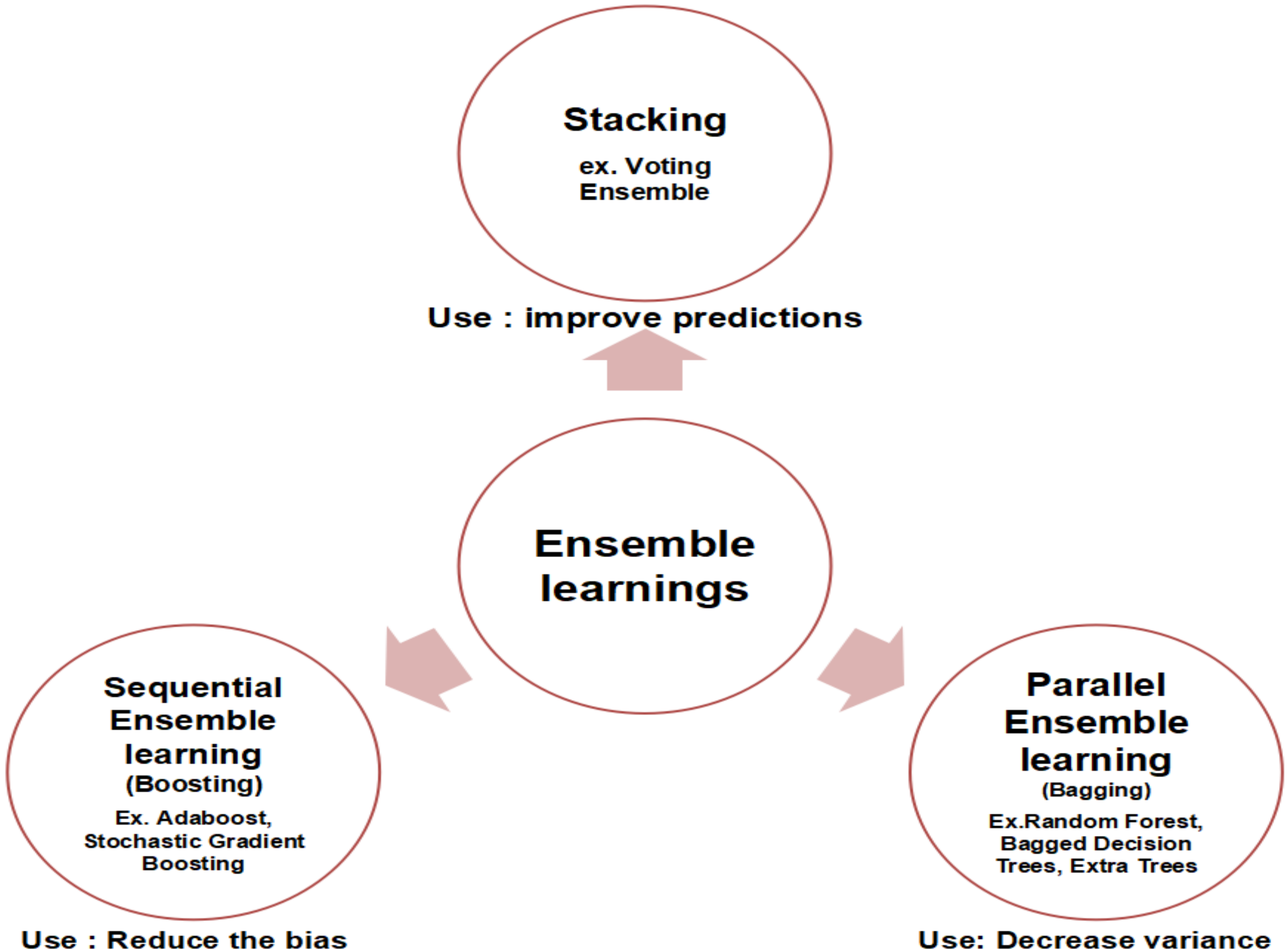
Lecture #4.1

Ensemble learning, Bagging and AdaBoost classifier

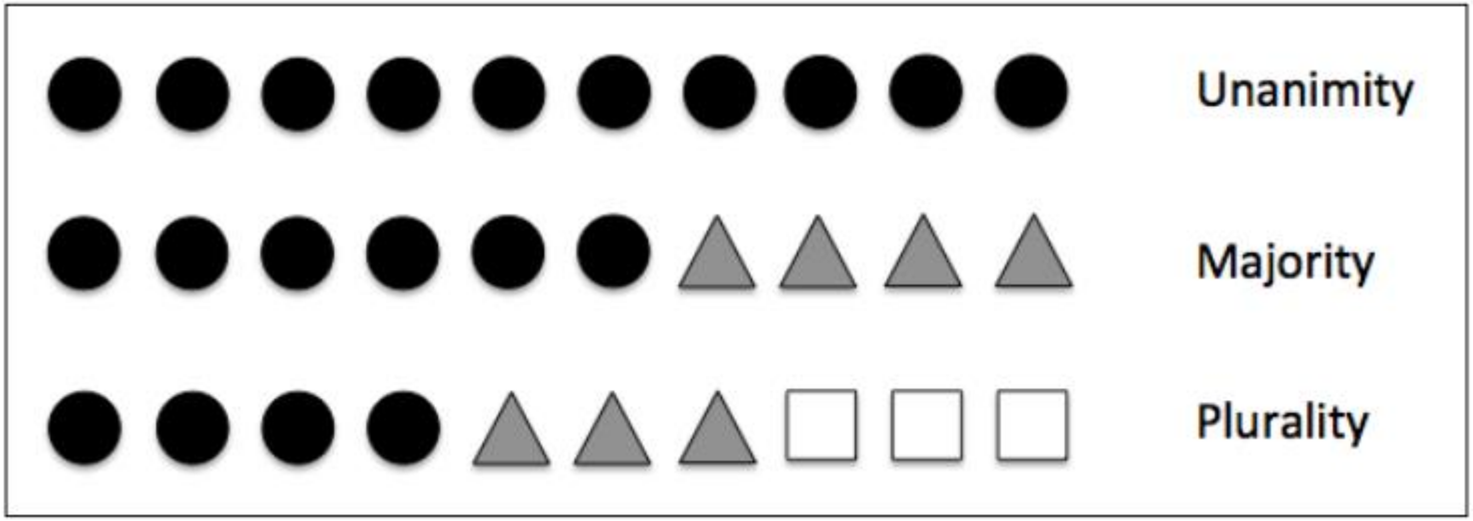
Learning with ensembles

- The goal behind **ensemble methods** is to combine different classifiers into a meta-classifier that has a better generalization performance than each individual classifier alone.
-

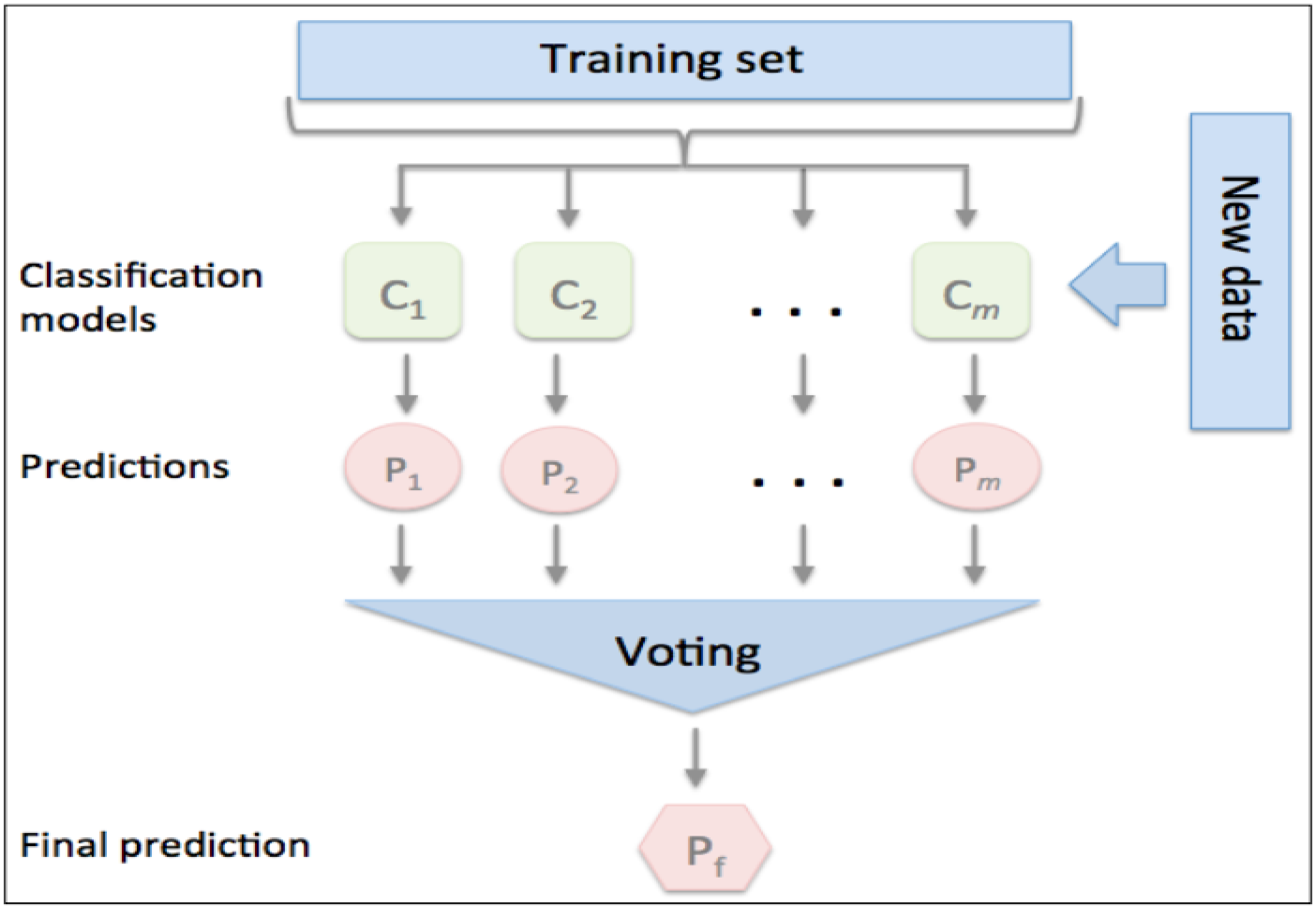

Ensemble Methods
Bagging
Boosting
Stacking


Stacking
ex. Voting Ensemble
Use : improve predictions
Ensemble learnings
Sequential Ensemble learning
(Boosting)
Ex. Adaboost, Stochastic Gradient Boosting
Use : Reduce the bias
Parallel Ensemble learning
(Bagging)
Ex.Random Forest, Bagged Decision Trees, Extra Trees
Use: Decrease variance

Voting



Majority Voting Classifier



Majority Vote prediction

$$\hat{y} = mode\{C_1(\boldsymbol{x}), C_2(\boldsymbol{x}), \ldots, C_m(\boldsymbol{x})\}$$

For example, in a binary classification task where $class1 = -1$ and $class2 = +1$, we can write the majority vote prediction as follows:

$$C(\boldsymbol{x}) = sign\left[\sum_j^m C_j(\boldsymbol{x})\right] = \begin{cases} 1 \ \ if \sum_i C_j(\boldsymbol{x}) \geq 0 \\ -1 \ \ otherwise \end{cases}$$

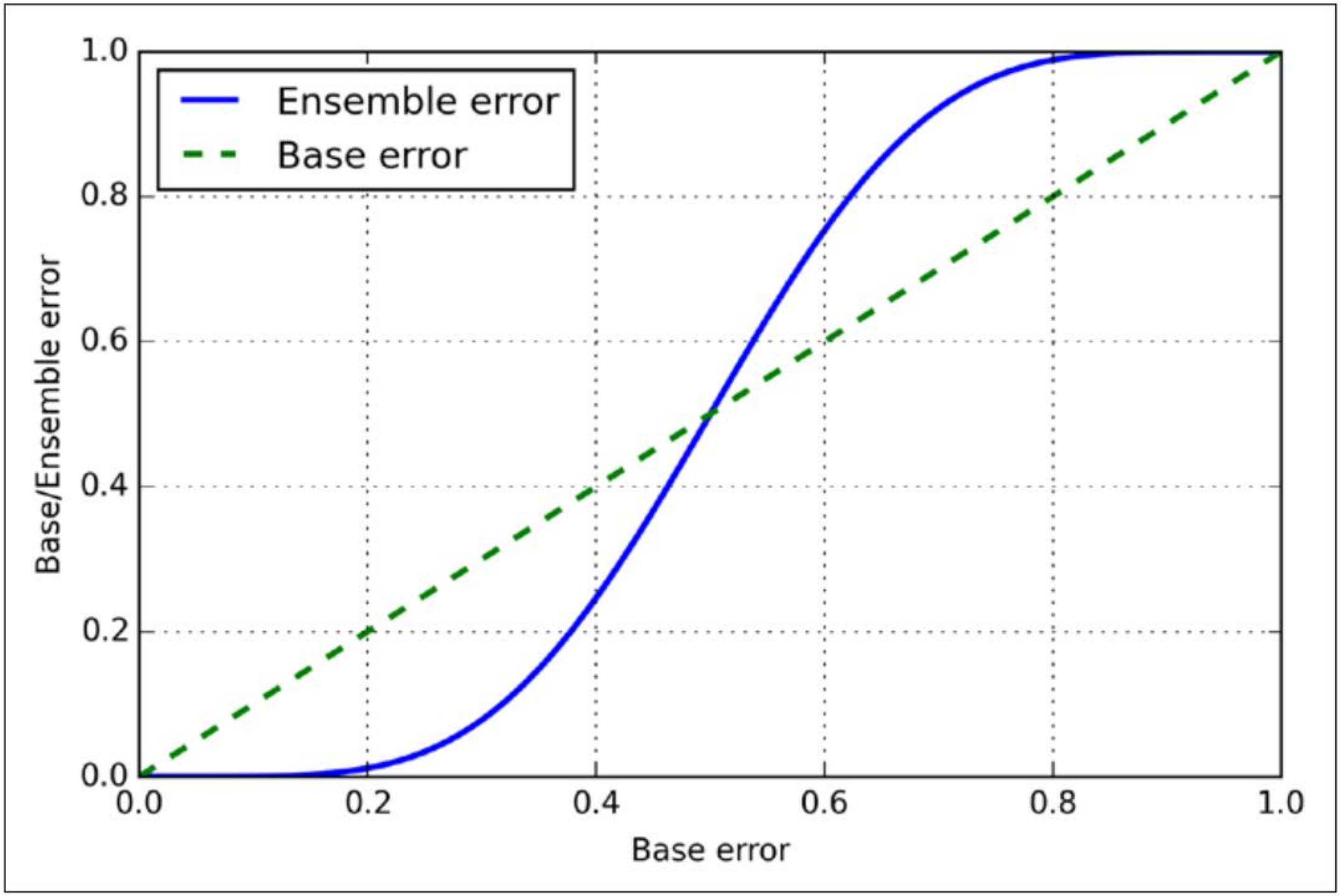
Error rate for ENSEMBLE Learner

$$P(y \geq k) = \sum_{k}^{n} \left\langle \begin{matrix} n \\ k \end{matrix} \right\rangle \varepsilon^{k} (1-\varepsilon)^{n-k} = \varepsilon_{ensemble}$$

Here, $\left\langle \begin{matrix} n \\ k \end{matrix} \right\rangle$ is the binomial coefficient *n choose k*. In other words, we compute the probability that the prediction of the ensemble is wrong. Now let's take a look at a more concrete example of 11 base classifiers ($n = 11$) with an error rate of 0.25 ($\varepsilon = 0.25$):

$$P(y \geq k) = \sum_{k=6}^{n} \left\langle \begin{matrix} 11 \\ k \end{matrix} \right\rangle 0.25^{k} (1-\varepsilon)^{11-k} = 0.034$$

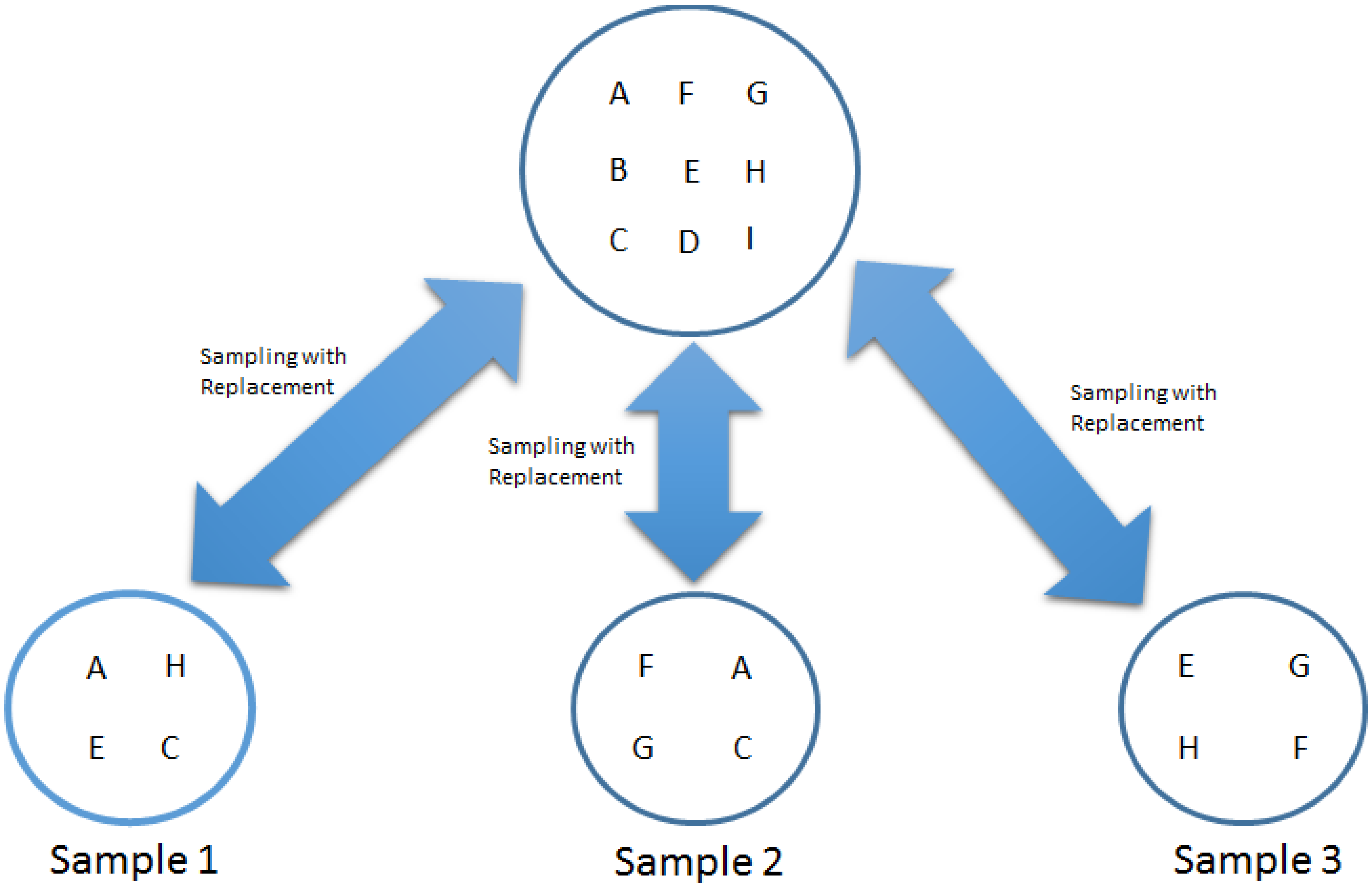
Ensemble Error V/s. Base Error

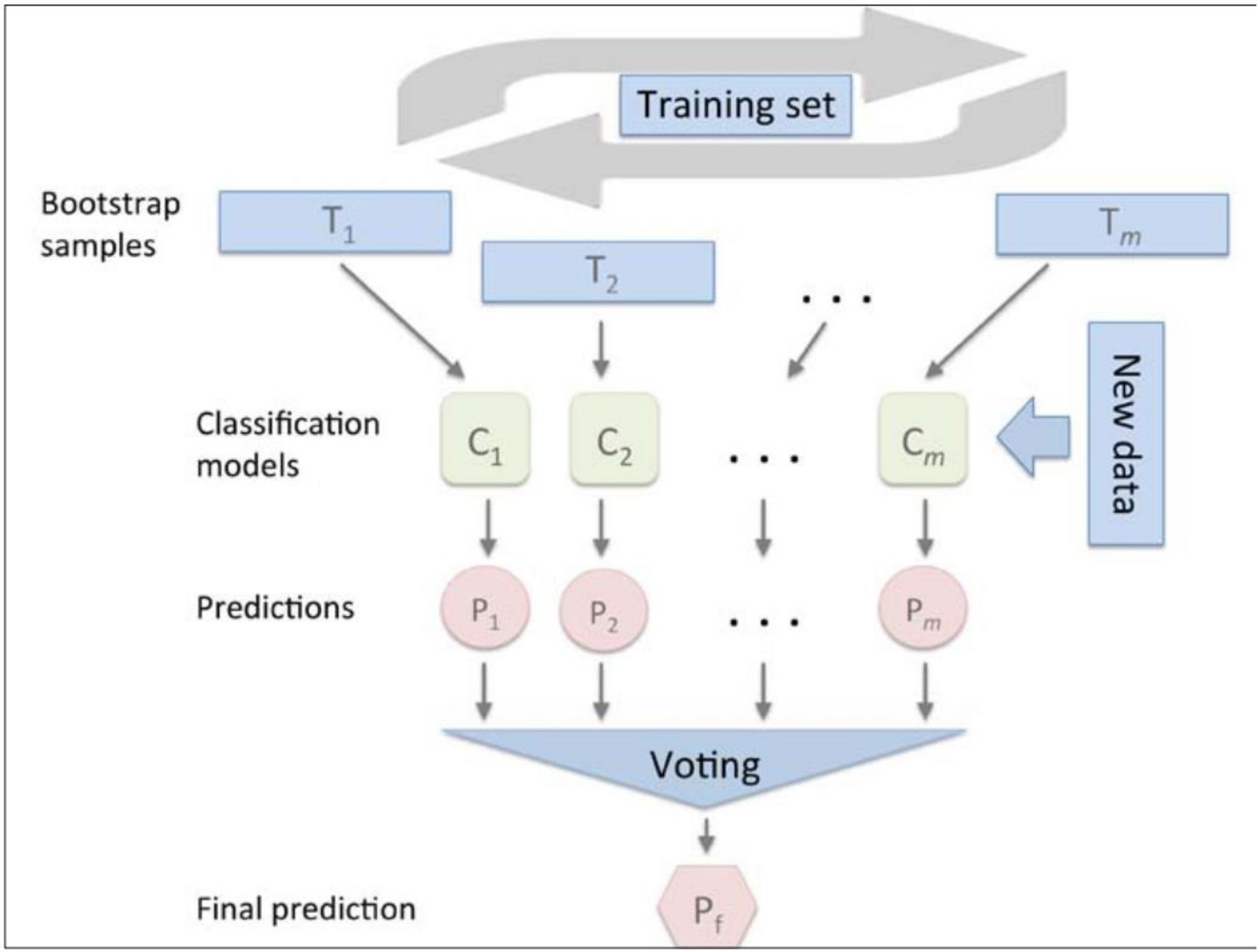


Bagging

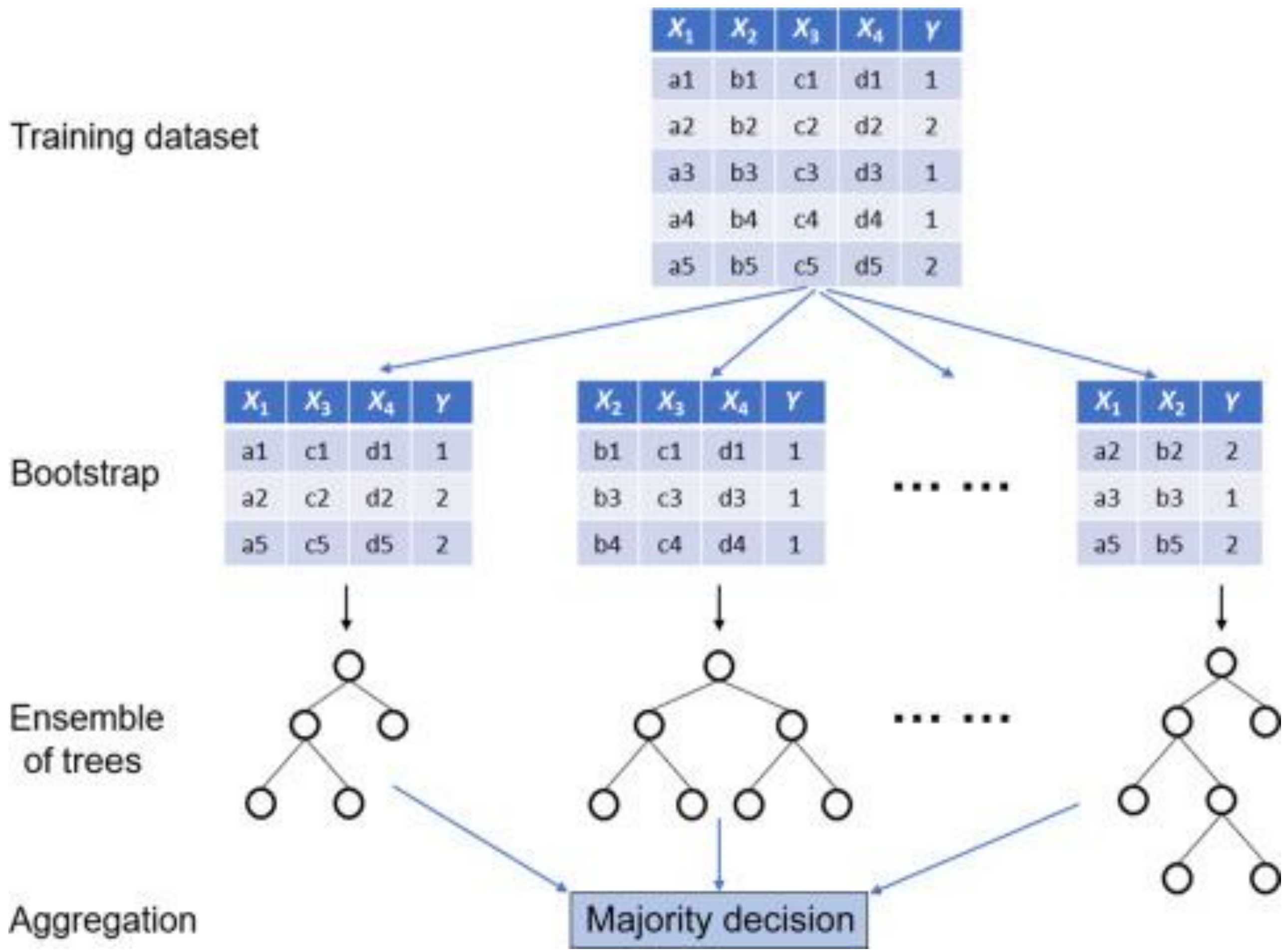
- Building an ensemble of classifiers from bootstrap samples [**Random Samples with Replacement**].

Bootstrap samples




Training set
Bootstrap samples
T_1
T_2
. . .
T_m
New data
Classification models
C_1
C_2
. . .
C_m
Predictions
P_1
P_2
. . .
P_m
Voting
Final prediction
P_f

Random Forest classifier



Boosting

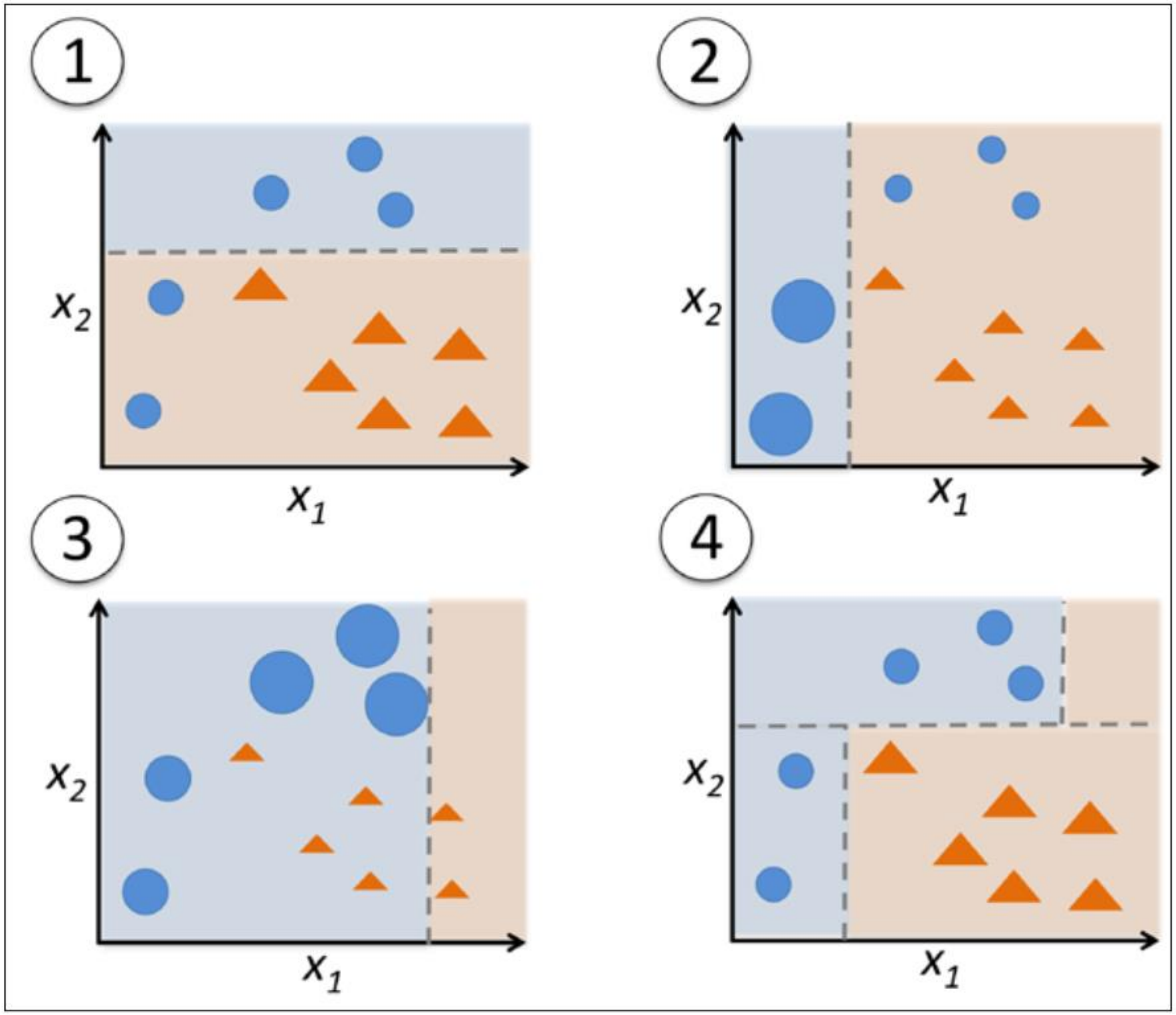
- **Leveraging weak learners via adaptive boosting** [**AdaBoost**]
- boosting is to ***focus on training samples that are hard to classify***, that is, to let the weak learners subsequently learn from misclassifed training samples to improve the performance of the ensemble.

Boosting

- In contrast to bagging, the initial formulation of boosting, the algorithm uses **random subsets of training samples drawn from the training dataset without replacement**.
- The original boosting procedure is summarized in four key steps as follows:

Boosting

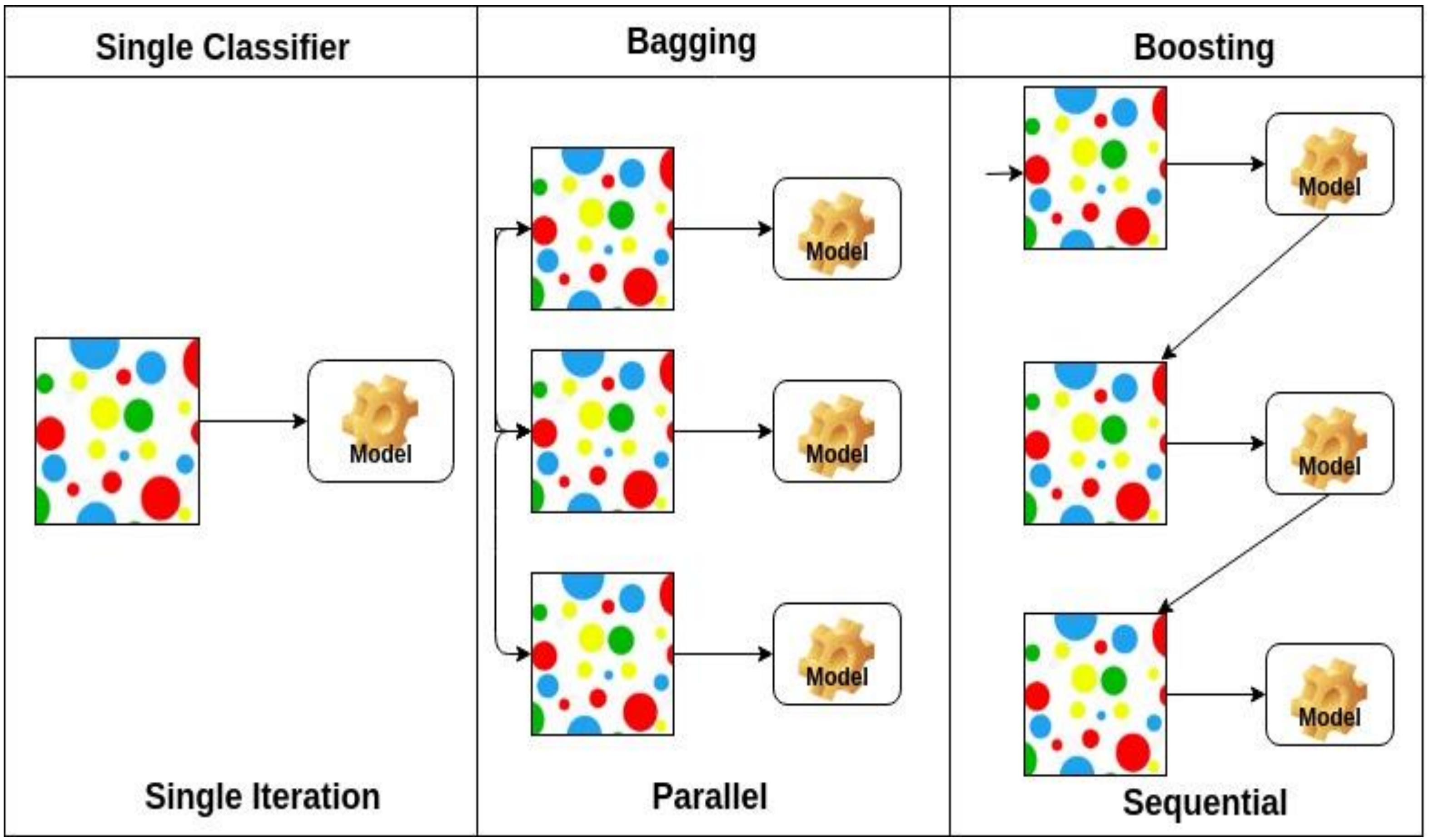
1. Draw a random subset of training samples $d1$ without replacement from the
   training set D to train a weak learner $C1$.
2. Draw second random training subset $d2$ without replacement from the
   training set and add 50 percent of the samples that were previously
   misclassified to train a weak learner $C2$.
3. Find the training samples $d3$ in the training set D on which $C1$ and $C2$ disagree to train a third weak learner $C3$.
4. Combine the weak learners $C1$, $C2$, and $C3$ via majority voting.


1
2
3
4
x_1
x_2

AdaBoost Classifier Algorithm

1. Set weight vector $\boldsymbol{w}$ to uniform weights where $\sum_i w_i = 1$
2. For j in m boosting rounds, do the following:
3. Train a weighted weak learner: $C_j = \text{train}(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w})$.
4. Predict class labels: $\hat{y} = \text{predict}(C_j, \boldsymbol{X})$.
5. Compute weighted error rate: $\varepsilon = \boldsymbol{w} \cdot (\hat{\boldsymbol{y}} == \boldsymbol{y})$.
6. Compute coefficient: $\alpha_j = 0.5 \log \dfrac{1-\varepsilon}{\varepsilon}$.
7. Update weights: $\boldsymbol{w} := \boldsymbol{w} \times \exp\left(-\alpha_j \times \hat{\boldsymbol{y}} \times \boldsymbol{y}\right)$.
8. Normalize weights to sum to 1: $\boldsymbol{w} := \boldsymbol{w} / \sum_i w_i$.
9. Compute final prediction: $\hat{\boldsymbol{y}} = \left(\sum_{j=1}^{m} \left(\boldsymbol{\alpha}_j \times \text{predict}\left(C_j, \boldsymbol{X}\right)\right) > 0\right)$.

Bagging v/s. Boosting

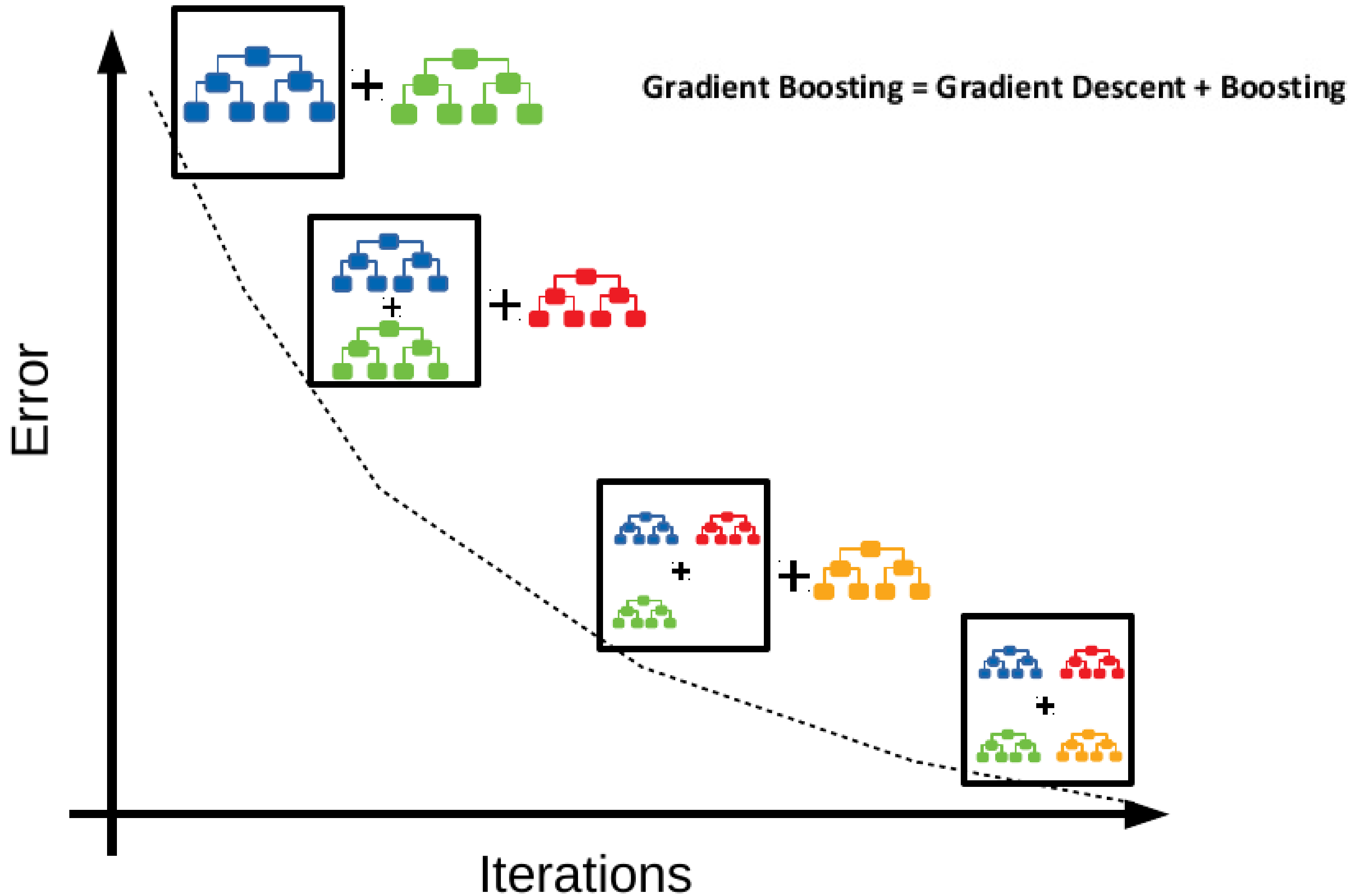


Gradient Boosting

- **Generalization of AdaBoost as Gradient Boosting**

- Gradient boosting involves three elements:

1. A loss function to be optimized.
2. A weak learner to make predictions.
3. An additive model to add weak learners to minimize the loss function.


Gradient Boosting = Gradient Descent + Boosting
Error
Iterations

Other Boosting Algorithms

- XGBoost
- CatBoost
- LIghtGBM


COMING UP