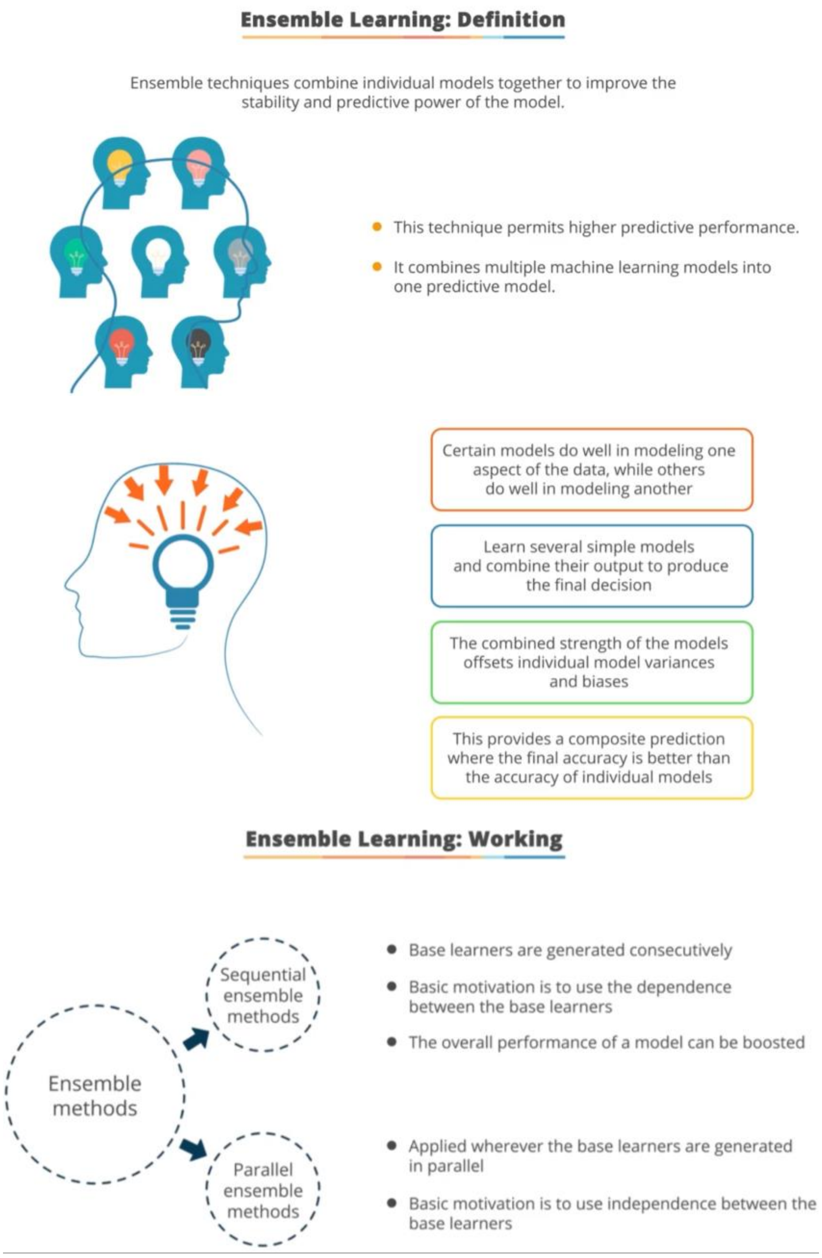
ENSEMBLES in Machine Learning

Ensemble-Learning Features:

1) Decrease Variance using Bagging
2) Decrease Bias using Boosting
3) Improves Predictions using Stacking



Sequential Ensemble Methods:

Example: AdaBoost

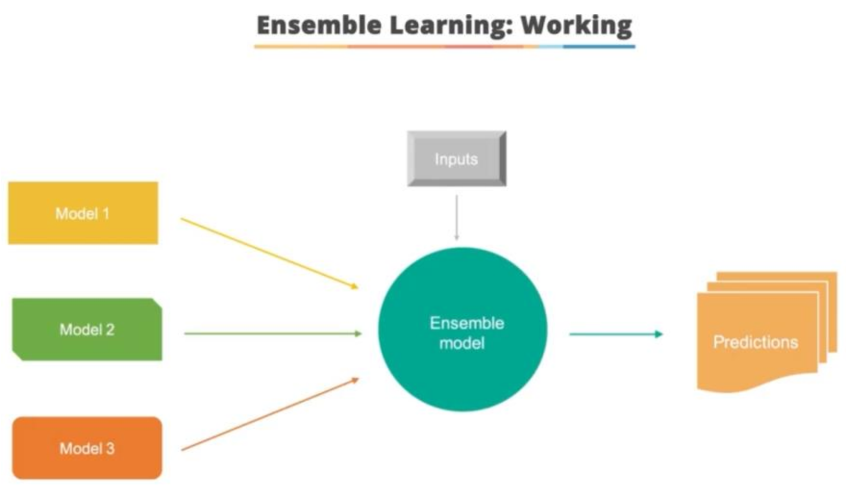
Use the dependence between the base learners by assigning larger weight values to previously misclassified instances.

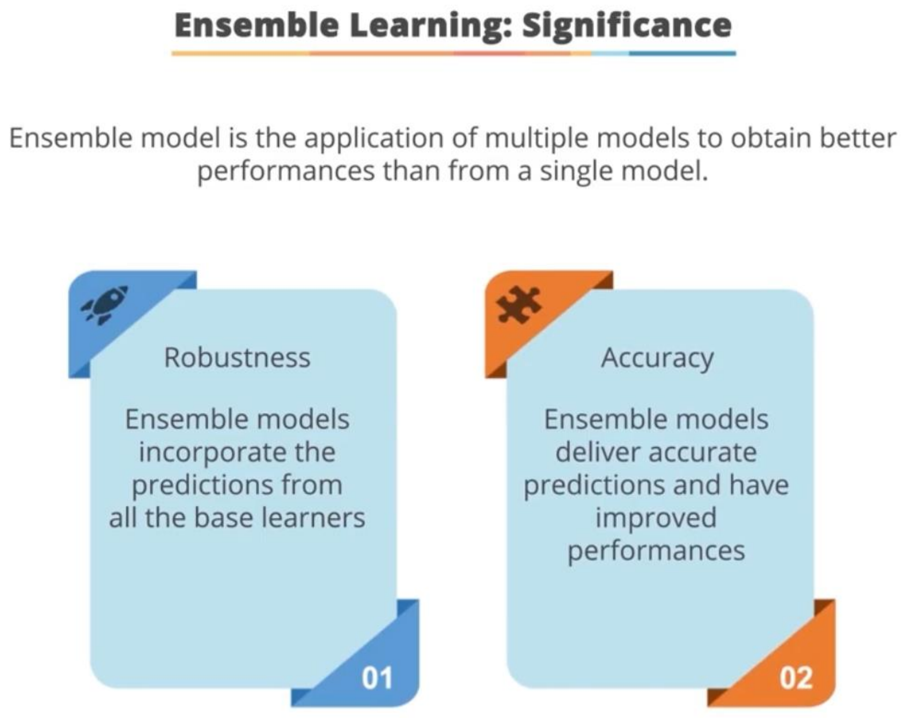
Parallel Ensemble Methods:

Example: RandomForests

The errors are often reduced dramatically by averaging.

The basic motivation is to use the independence between the base learners





Ensemble Learning Methods

Combine all "weak" learners to form an ensemble

OR

Create an ensemble of well-chosen strong and diverse models

Ensemble models gain more accuracy and robustness by combining data from numerous modeling approaches.

Averaging

Ensemble prediction

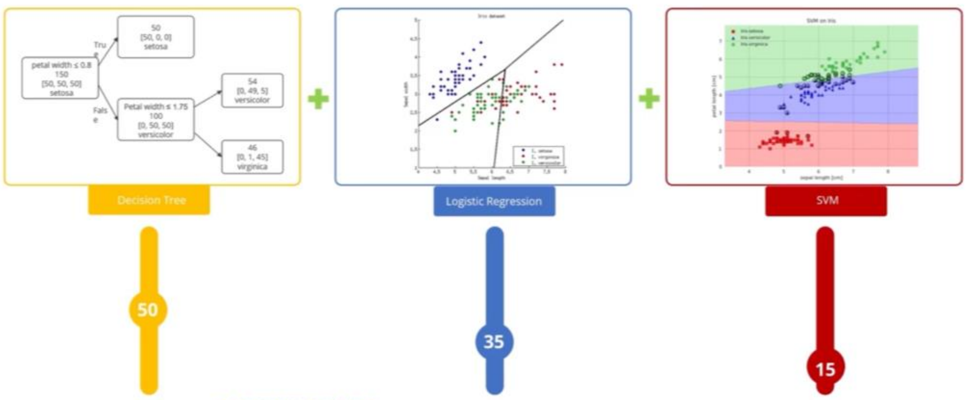
$$P = \frac{p1 + p2 + p3}{3}$$

A class label is predicted using mode of member predictions.

The class probability is calculated as the argmax of the summed probabilities for each class label.

Equal weights are assigned to different models despite some models performing better than others.

Weighted Averaging

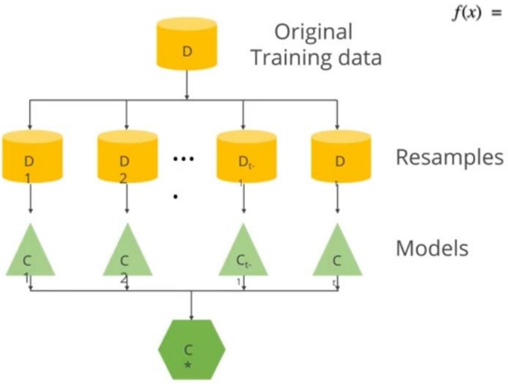


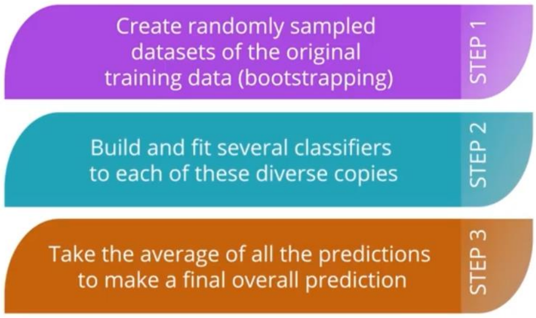
w1 + w2 + w3 = 1

Bagging

Bagging or bootstrap aggregation **reduces variance** of an estimate by taking mean of multiple estimates.

$$f(x) = 1/M \sum_{m=1}^{M} f_m(x)$$





Random Forests

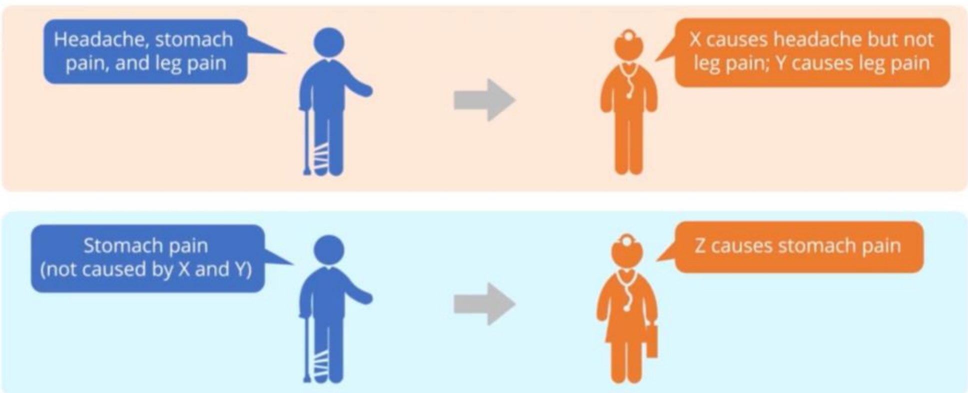
Random forest is a good example of ensemble machine learning method.

- Random forest technique combines various decision trees to produce a more generalized model.
- Random forests are utilized to produce de-correlated decision trees.
- Random forest creates random subsets of the features.
- Smaller trees are built using these subsets, creating tree diversity.

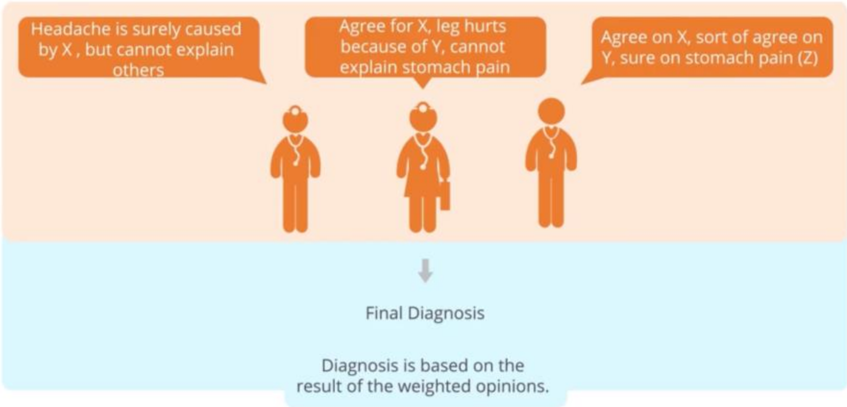
To overcome overfitting, diverse sets of decision trees are required.

Boosting

Boosting reduces bias by training weak learners sequentially, each trying to correct its predecessor.



Boosting



Boosting Algorithm

STEP 1	STEP 2	STEP 3
Train a classifier H1 that best classifies the data with respect to accuracy	Identify the regions where H1 produces errors, add weights to them, and produce a H2 classifier	Aggregate those samples for which H1 gives a different result from H2 and produces H3 classifier

AdaBoost

Boosting is a technique of changing weak learners into strong learners.

Each new tree is a fit on a modified version of the original dataset.

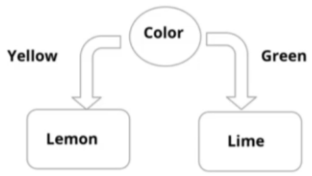
- AdaBoost is the first boosting algorithm to be adapted in solving practices
- It helps mixing multiple weak classifiers into one strong classifier

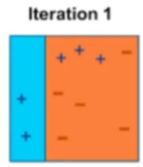
Consider a scenario, where there are '+' and '–'

Objective : Classify '+' and '–'

AdaBoost Working: Step 1

Assign equal weights to each data point and apply a decision stump
to classify them as + (plus) and – (minus).



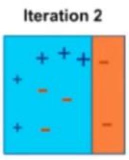


For distinct attributes, the tree consists only of a single interior node.

Now, apply higher weights to incorrectly predicted three + (plus) and add another decision stump.

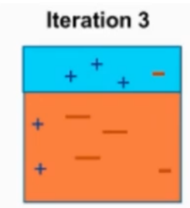
AdaBoost Working: Step 2

- The size of three incorrectly predicted + (plus) is mad bigger than the rest of the data points.
- The second decision stump (D2) will try to predict them correctly.
- Now, vertical plane (D2) has classified three mis-classified + (plus) correctly.
- D2 has also caused mis-classification errors to three – (minus).



AdaBoost Working: Step 3

- D3 adds higher weights to three – (minus).
- A horizontal line is generated to classify + (plus) and – (minus) based on higher weight of mis-classified observation.

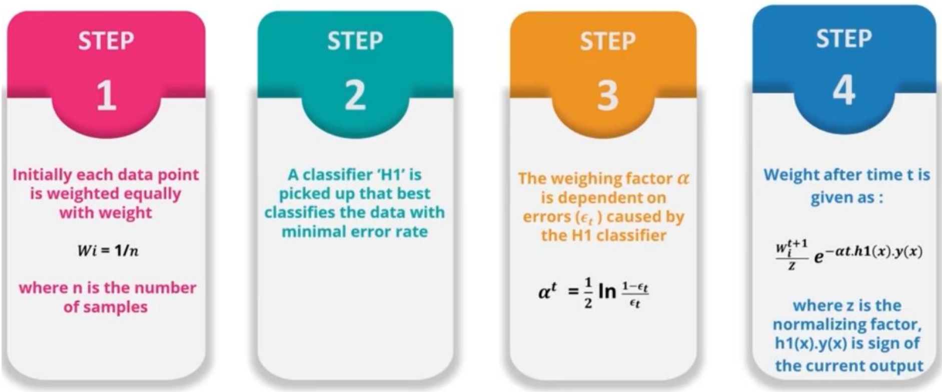


AdaBoost Algorithm

- A weak classifier is prepared on the training data using the weighted samples.
- Only binary classification problems are supported.
- Every decision stump makes one decision on one input variable and outputs a +1.0 or -1.0 value for the first- or second-class value.

Misclassification rate

error = (correct – N)/N



AdaBoost Flowchart

AdaBoost selects a training subset randomly.

↓

It iteratively trains the AdaBoost machine learning model.

↓

It assigns a higher weight to wrongly classified observations.

↓

It assigns weight to the trained classifier in each iteration according to the accuracy of the classifier.

↓

This process iterates until the complete training data fits without any error.