Improve Machine Learning Results with Ensemble Learning

CDC Data Science Certificate Program

Agenda

Course objectives

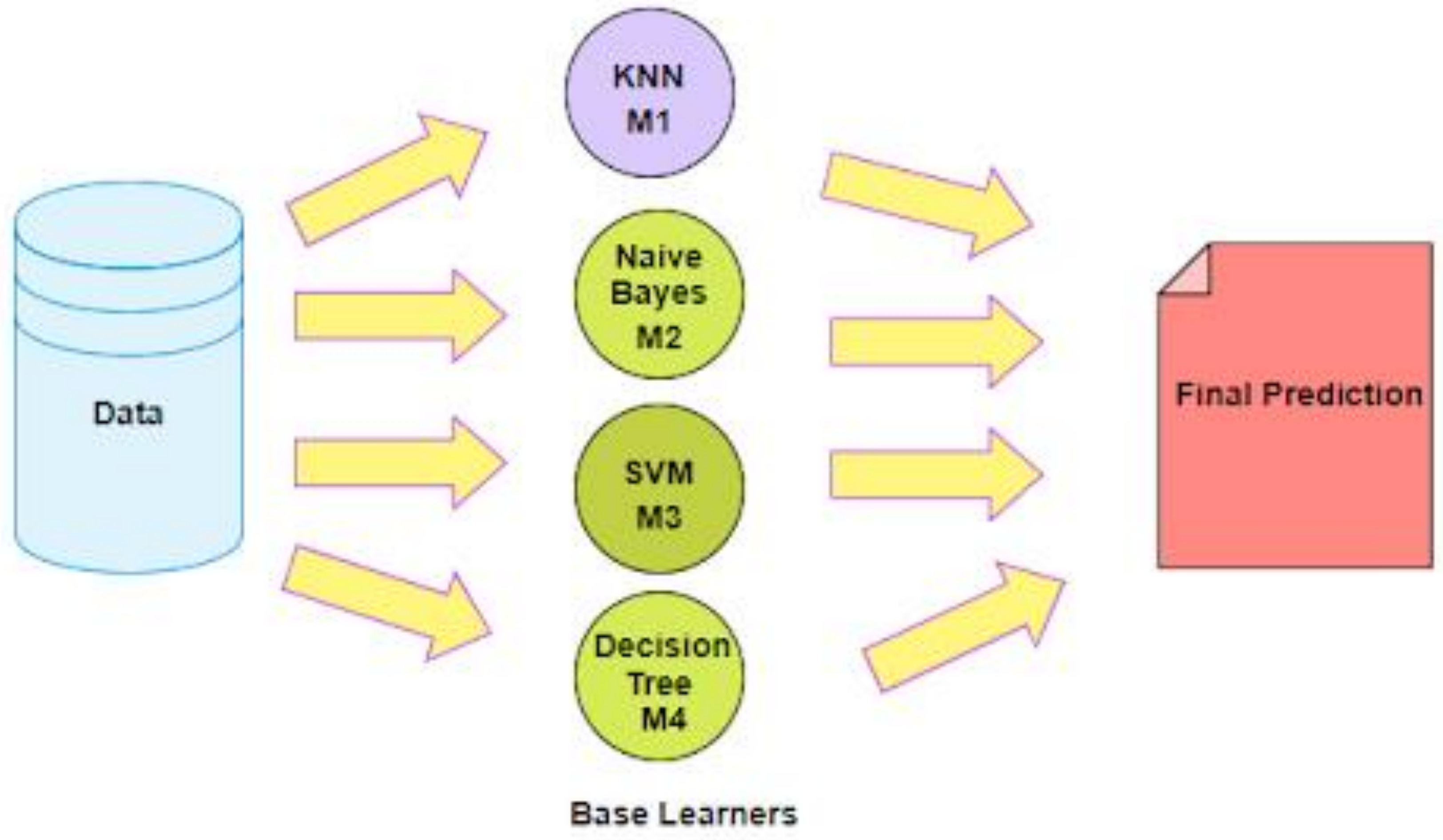
- Carry out an essential review of re-sampling methods, bootstrap
- Explore the key ensemble methods: bagging, random forests, and boosting
- Understanding the idea and usage of Boosting and AdaBoost
- Use multiple algorithms to make strong predictive models
- Supplement methods with statistical tests, such as ROC
- Use the supplied R code to implement ensemble methods

BeVera

Background Literature Review

Methods discussed in this course are mathematically intensive. Literature is provided for the reader to better understand how the algorithms compute. Please take time to review the papers.

Ensemble learning is a machine learning technique in which the same algorithm is used multiple times, or a group of different algorithms are combined to build a more powerful model. The ensemble is primarily used to improve the performance of the model.



Ensemble Learning:

- Combine predictions of multiple learning algorithms → ensemble
- Often leads to a better predictive performance than a single learner
- Well-suited when small differences in the training data produce very different classifiers (e. g. decision trees)
- Drawbacks: increases computation time, reduces interpretability

Reasoning:

- Classifiers $C_1,\ldots,C_K$ which are independent. i.e., $\mathrm{cor}(C_i, C_j) = 0$
- Each has an error probability of $P_i < 0.5$ on the training data
- Then an ensemble of classifiers should have an error probability lower than each individual P_i

These are the main instanced when you should use ensemble learning:

WISDOM OF ALL

THE DATASET IS TOO LARGE OR TOO SMALL

COMPLEX (NONLINEAR) DATA

HIGH CONFIDENCE

DECREASE VARIANCE AND BIAS

Bootstrapping: Bootstrap randomly performs row sampling and feature sampling from the dataset to form sample datasets for every model. By bootstrapping, a better overall estimate of the desired quantity can be achieved than simply estimating from the dataset directly.

Types

Stacking

Used to improve accuracy.

Supervised ensemble machine learning algorithm that finds the optimal combination of a collection of prediction algorithms using a process called stacking.

Sequential

Used to reduce bias.

Boosting is an iterative technique which adjust the weight of an observation based on the last classification.

Parallel

Used to reduce varience.

Bagging tries to implement similar learners on small sample populations and then takes a mean of all the predictions. In generalized bagging, you can use different learners on different population. As you can expect this helps us to reduce the variance error.

Stacking / Super Learning: Stacking is a broad class of algorithms that unlike bagging and boosting, the goal in stacking is to ensemble strong, diverse sets of learners together.

Boosting: designed to reduce bias and variance. A boosting algorithm iteratively learns weak classifiers and adds them to a final strong classifier.

- After a weak learner is added, the data is reweighted: examples that are misclassified gain weight and examples that are classified correctly lose weight. Thus, future weak learners focus more on the examples that previous weak learners misclassified. This causes boosting methods to be not very robust to noisy data and outliers.

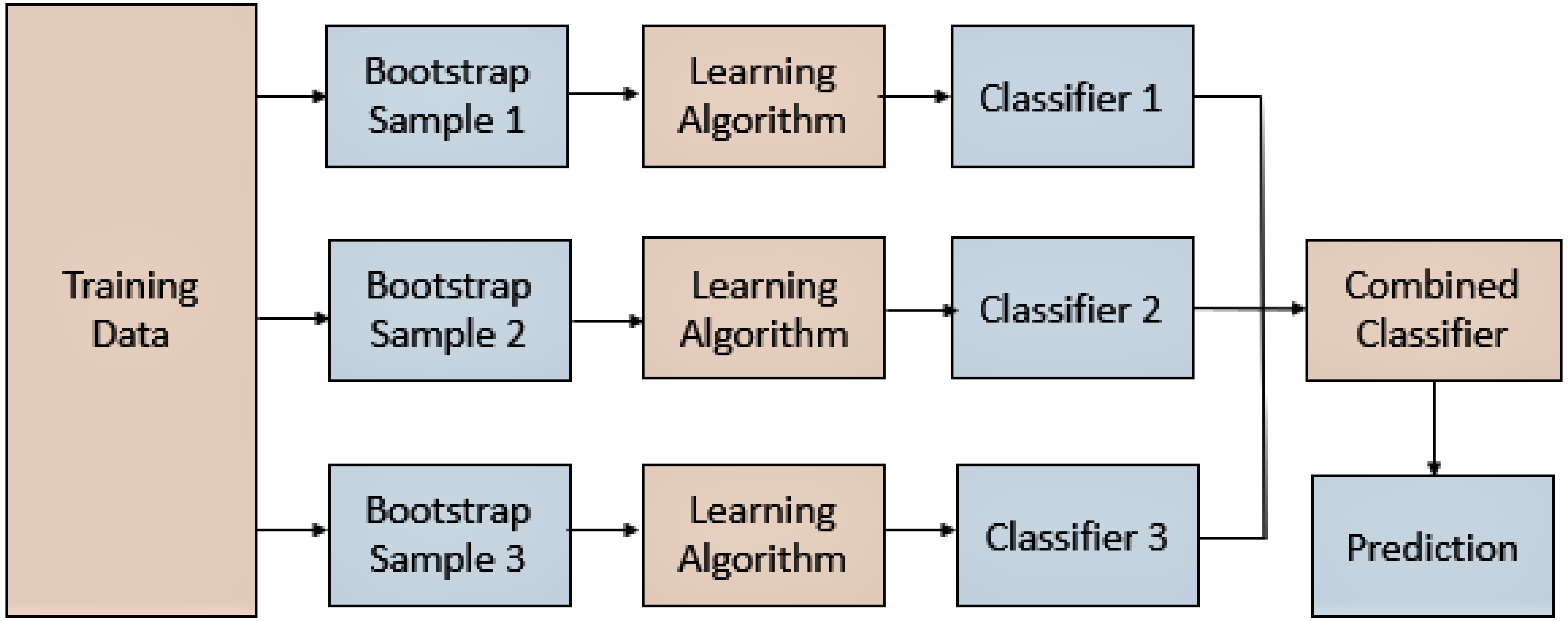
Bagging: Bootstrap aggregating, or bagging, designed to improve the stability and accuracy of machine learning algorithms.

- It reduces variance and helps to avoid overfitting. Bagging is a special case of the model averaging approach and is relatively robust against noisy data and outliers.
- One of the most well-known bagging ensembles is the Random Forest algorithm, which applies bagging to decision trees.

Both bagging and boosting are ensembles that take a collection of weak learners and forms a single, strong learner.

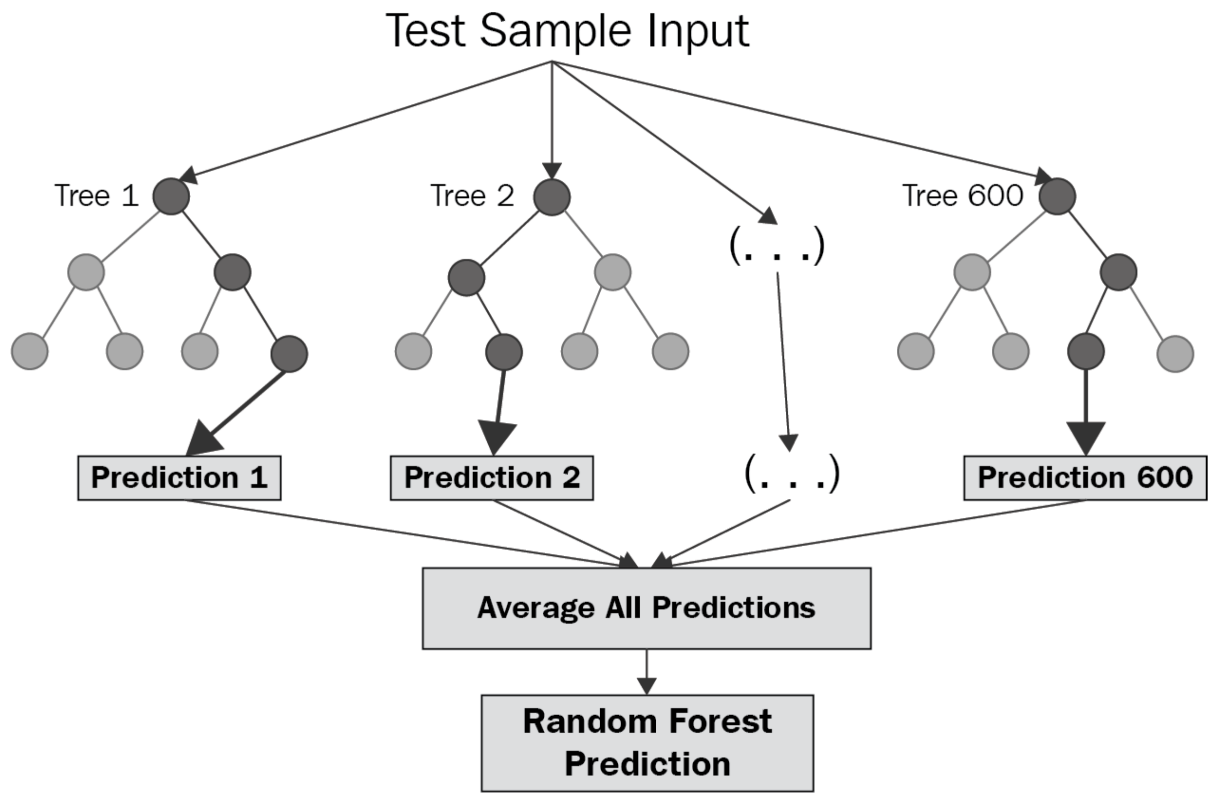
Bagging: Bootstrap Aggregation

Instead of running various models on a single dataset, you can use a single machine learning algorithm, almost always an unpruned decision tree, and train each model on a different sample of the same training dataset. The predictions made by the ensemble members are then combined using simple statistics, such as voting or averaging.



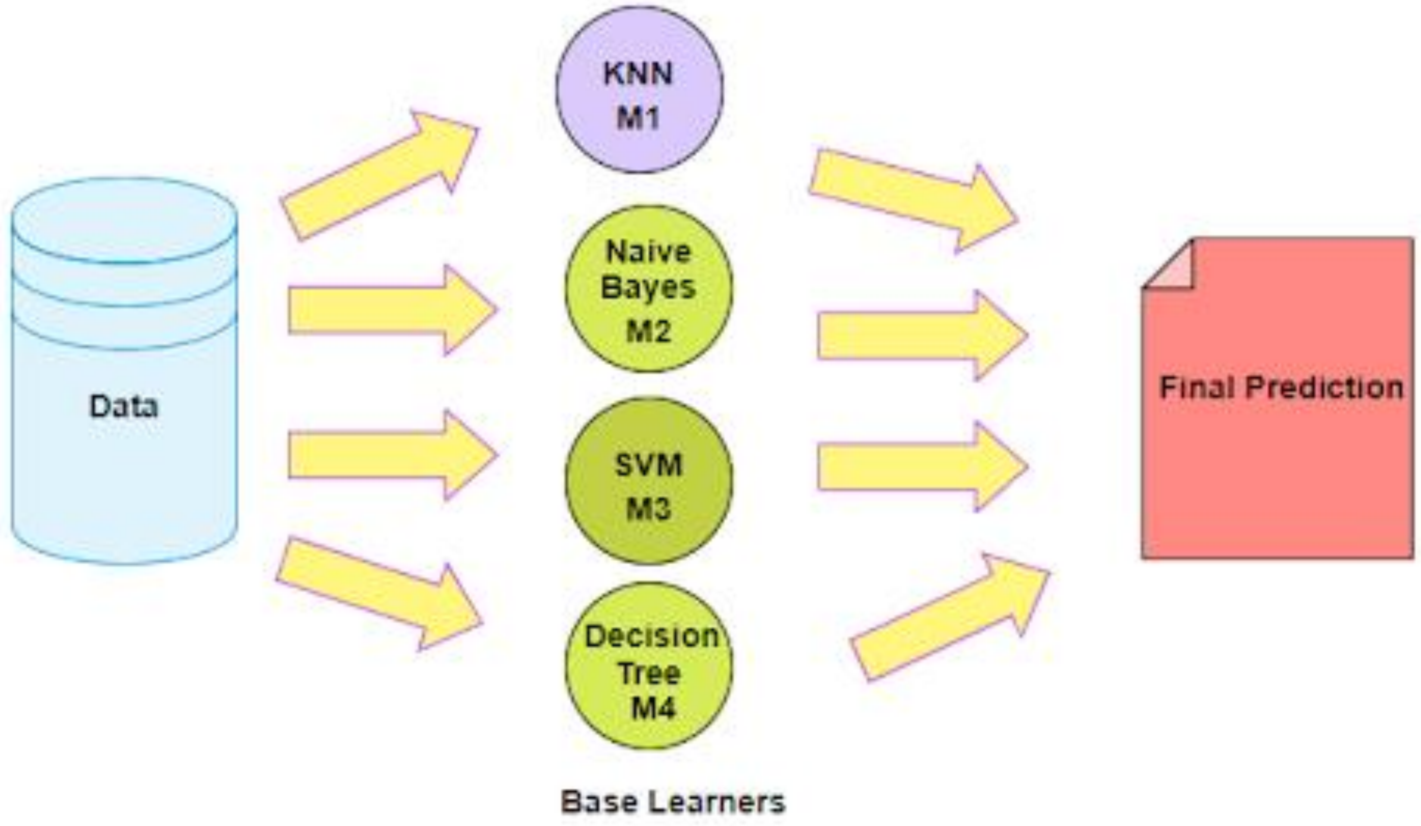
Random Forest = Row Sampling + Column Sampling

Random forest is an implementation of the bagging technique. You can see from its name; it creates a forest of the decision trees from randomly sampled data and combines them. So base learners in the random forest are decision trees.



Stacked Generalization, or **stacking** for short, is an ensemble method that involves fitting many different models' types on the same data and using another model to learn how to best combine the predictions.

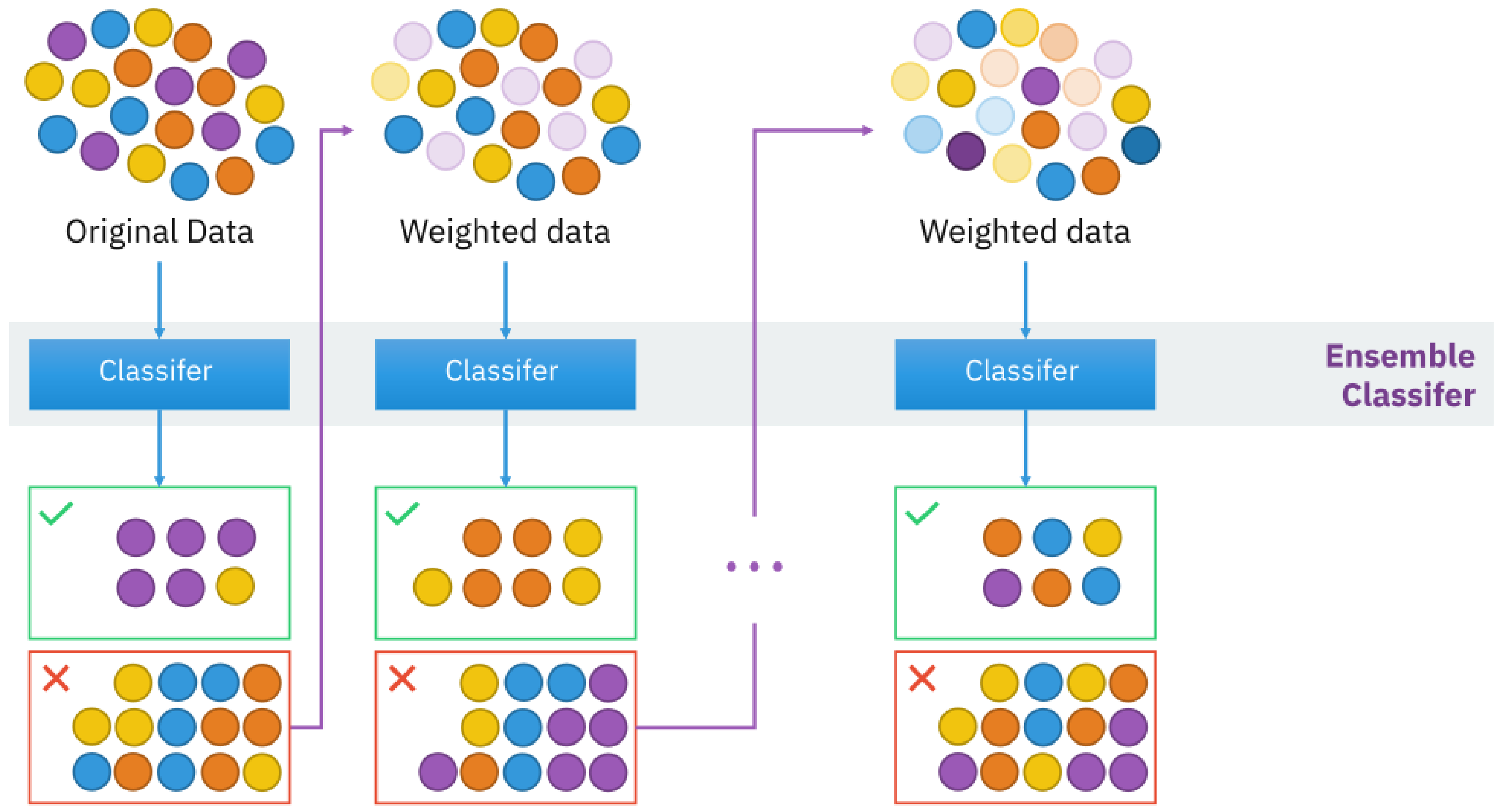
Stacking has its own nomenclature where ensemble members are referred to as level-0 models and the model that is used to combine the predictions is referred to as a level-1 model.



Boosting

Boosting in machine learning is an ensemble-based method used to boost the performance of weak learners.

Boosting trains models in succession, with each new model being trained to correct the errors made by the previous ones.



Different types of boosting algorithms

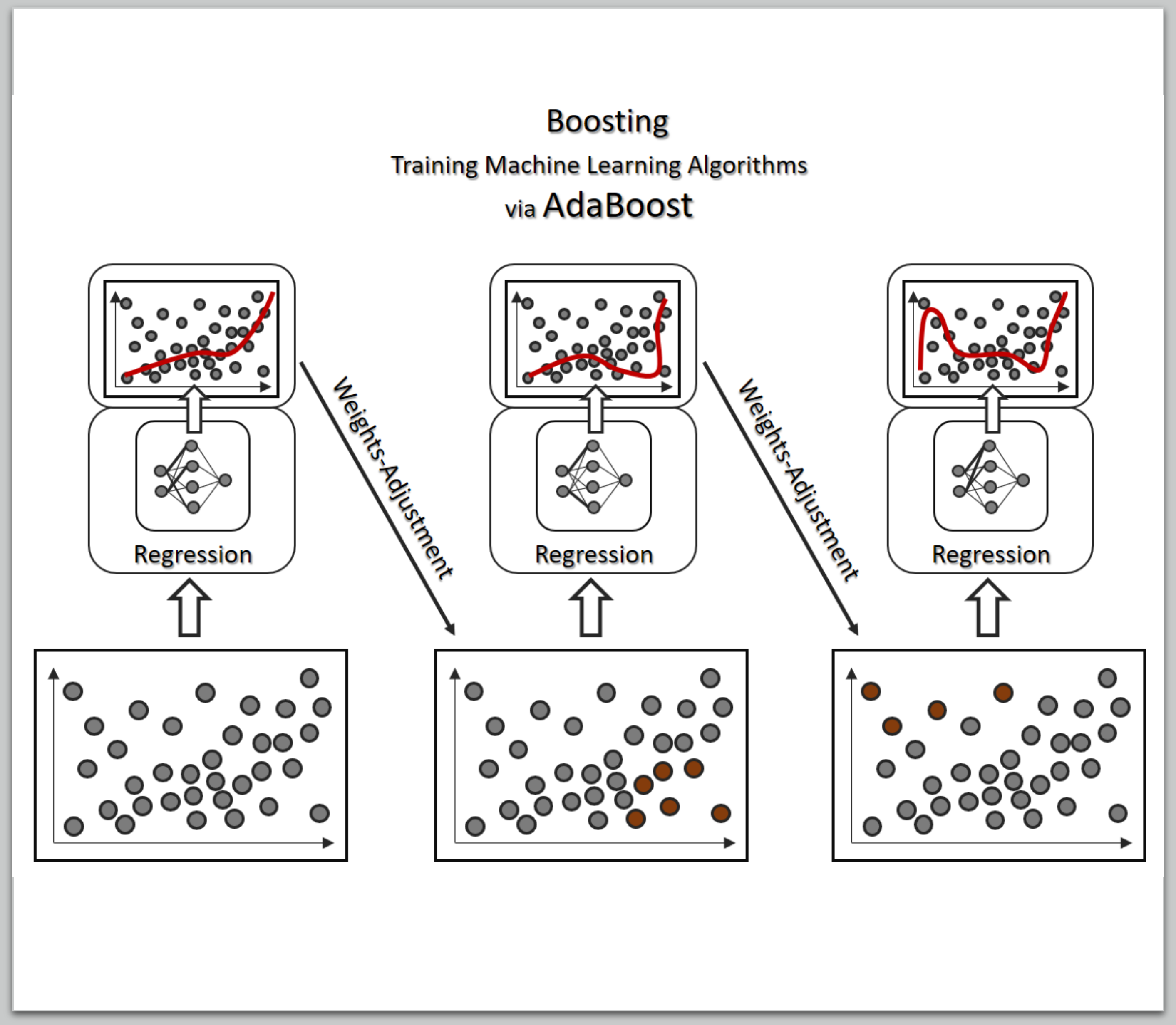
1. AdaBoost (Adaptive Boosting)
2. Gradient Boosting Machines
3. Stochastic Gradient Boosting (XGBoost and similar)

AdaBoost algorithm in Machine Learning

Instead of a Decision Tree, AdaBoost uses a decision stump which is a decision tree with tree maximum depth = 1.

Stumps that are created are not equally weighted in the final prediction. Stumps that create more error will have less say in the final decision.

The order in which the stumps are made is important, because each stump aims to reduce the errors that the previous stump(s) made.



Steps of Gradient Boost algorithm

Steps of Adaboost algorithm

Steps of Gradient Boost algorithm

Step 1: Assign a sample weight for each sample

Step 2: Calculate the Gini Impurity for each variable

Step 3: Calculate the Amount of Say for the stump that was created

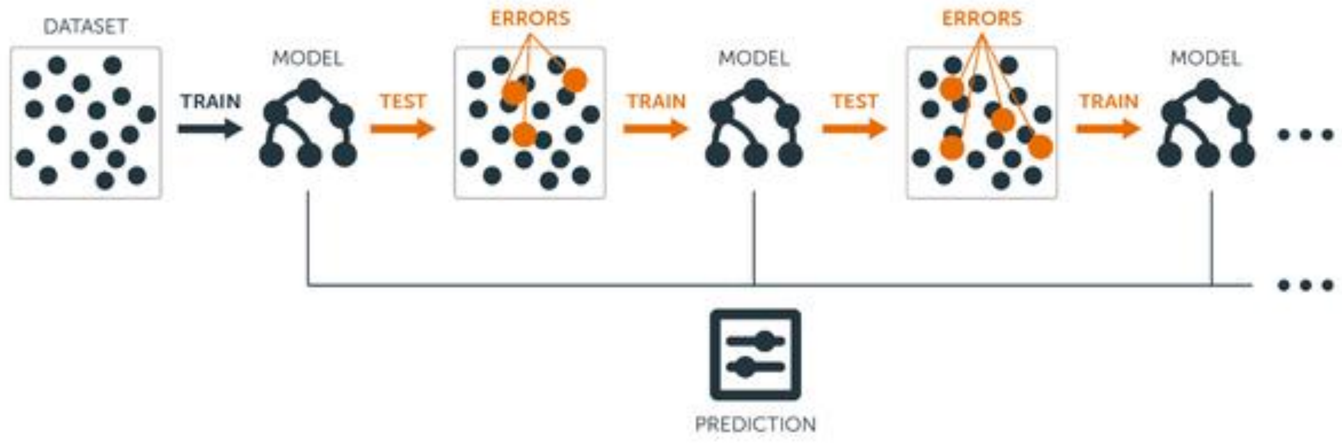
Step 4: Calculate the new sample weights for the next stump

Step 5: Create a bootstrapped dataset with the odds of each sample being chosen based on their new sample weights.

Step 6: Repeat the process n number of times

https://towardsdatascience.com/a-mathematical-explanation-of-adaboost-4b0c20ce4382

Gradient Boosting Machine



- A machine learning technique for regression, classification and other tasks, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It follows a loss function to minimize the error. BG aims to generate least residual as possible.
- The guiding heuristic is that good predictive results can be obtained through increasingly refined approximations.

Steps of Gradient Boost algorithm

Step 1 : Assume mean is the prediction of all variables.

Step 2 : Calculate errors of each observation from the mean (latest prediction).

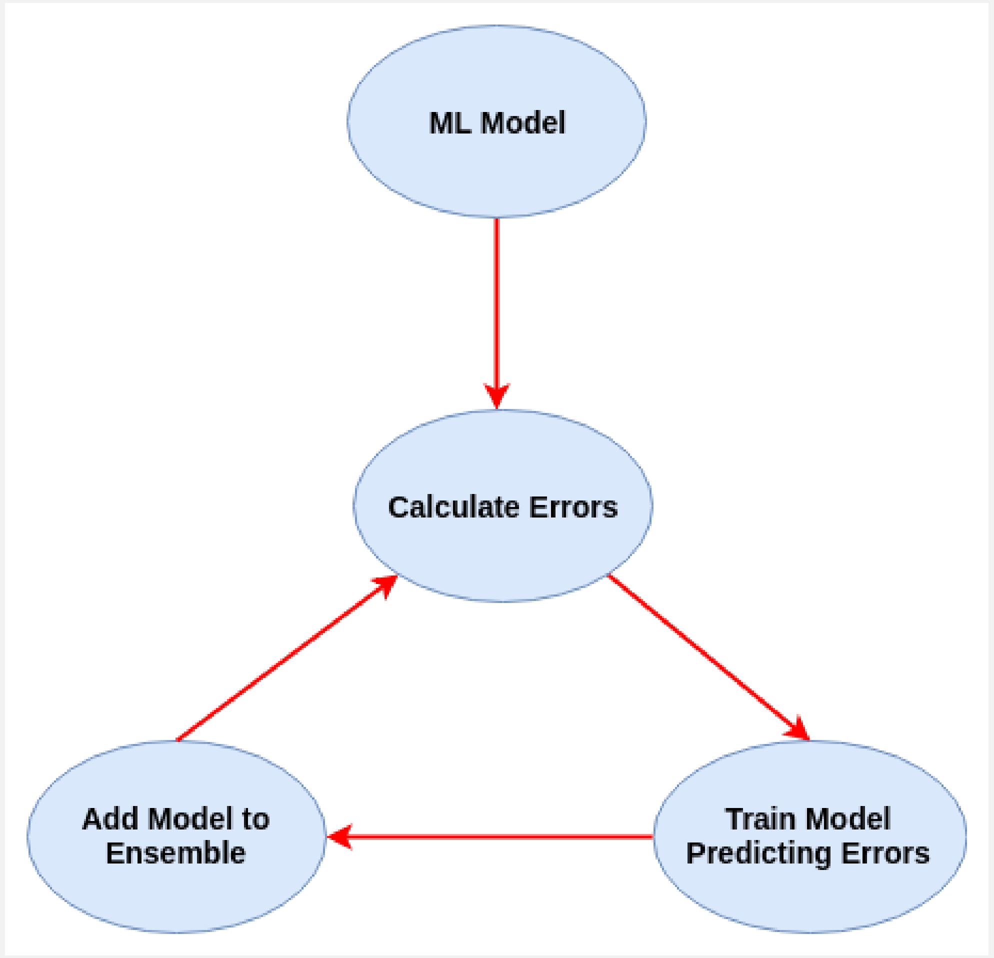
Step 3 : Find the variable that can split the errors perfectly and find the value for the split. This is assumed to be the latest prediction.

Step 4 : Calculate errors of each observation from the mean of both the sides of split (latest prediction).

Step 5 : Repeat the step 3 and 4 till the objective function maximizes/minimizes.

Step 6 : Take a weighted mean of all the classifiers to come up with the final model.

https://www.analyticsvidhya.com/blog/2015/05/boosting-algorithms-simplified/



Diagram

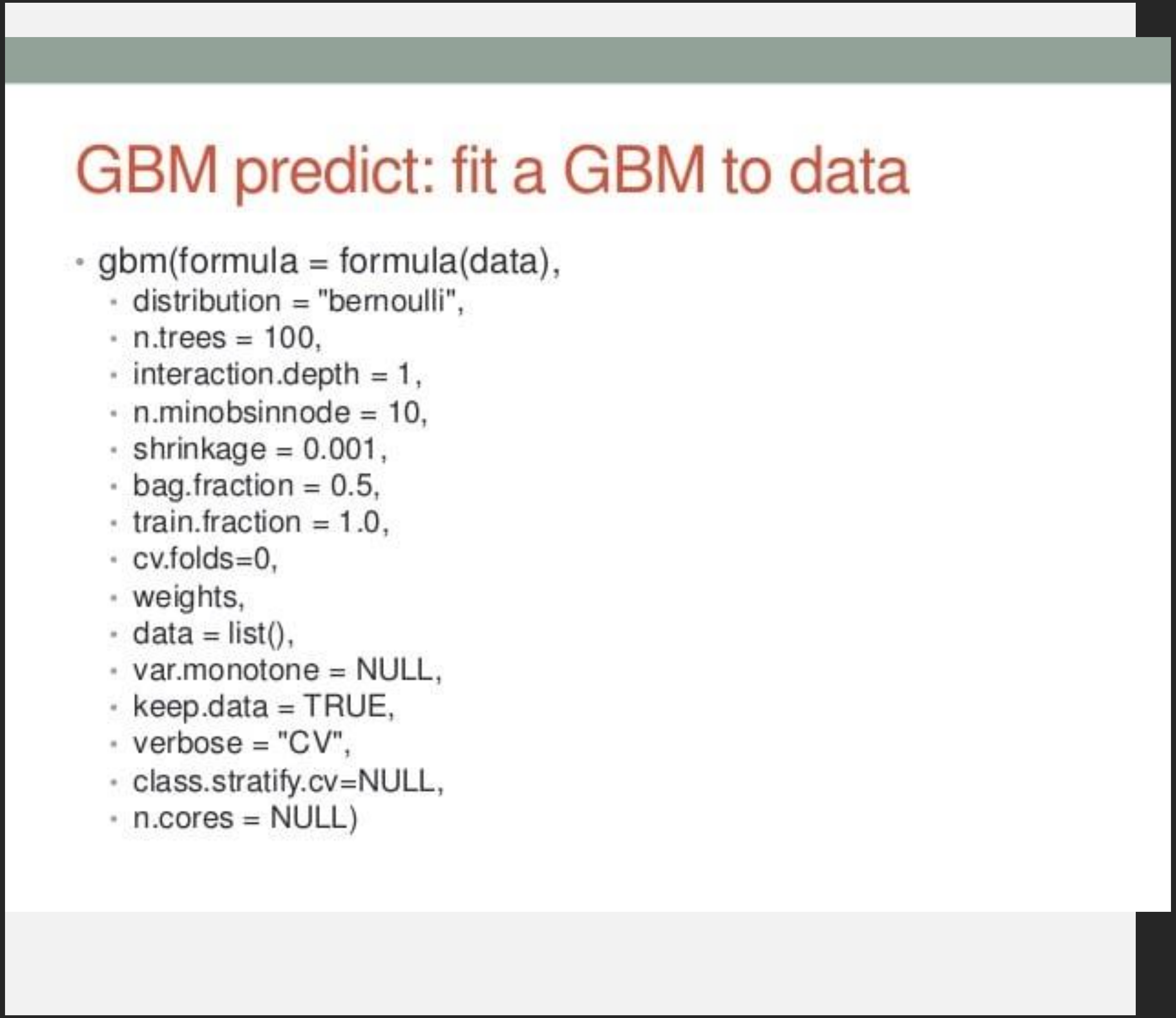
Gradient Boosting Machine

When a decision tree is the weak learner, the resulting algorithm is called gradient boosted trees, which usually outperforms random forest. Best used:

It can be used over regression when there is:

1. non-linearity in data
2. data is sparsely populated,
3. has low fill rate
4. basically, when regression is just unable to give expected results.

- learning rate (the “step size” with which we descend the gradient)
- shrinkage (reduction of the learning rate)
- loss function
- the number of iterations (i.e., the number of trees to ensemble)
- the number of observations in each leaf
- tree complexity and depth
- the proportion of samples
- the proportion of features on which to train on.



GBM predict: fit a GBM to data

- gbm(formula = formula(data),
  - distribution = "bernoulli",
  - n.trees = 100,
  - interaction.depth = 1,
  - n.minobsinnode = 10,
  - shrinkage = 0.001,
  - bag.fraction = 0.5,
  - train.fraction = 1.0,
  - cv.folds=0,
  - weights,
  - data = list(),
  - var.monotone = NULL,
  - keep.data = TRUE,
  - verbose = "CV",
  - class.stratify.cv=NULL,
  - n.cores = NULL)

Hyperparameters of Gradient Boosting

- learning rate (the “step size” with which we descend the gradient)
- shrinkage (reduction of the learning rate)
- loss function
- the number of iterations (i.e., the number of trees to ensemble)
- the number of observations in each leaf
- tree complexity and depth
- the proportion of samples
- the proportion of features on which to train on.

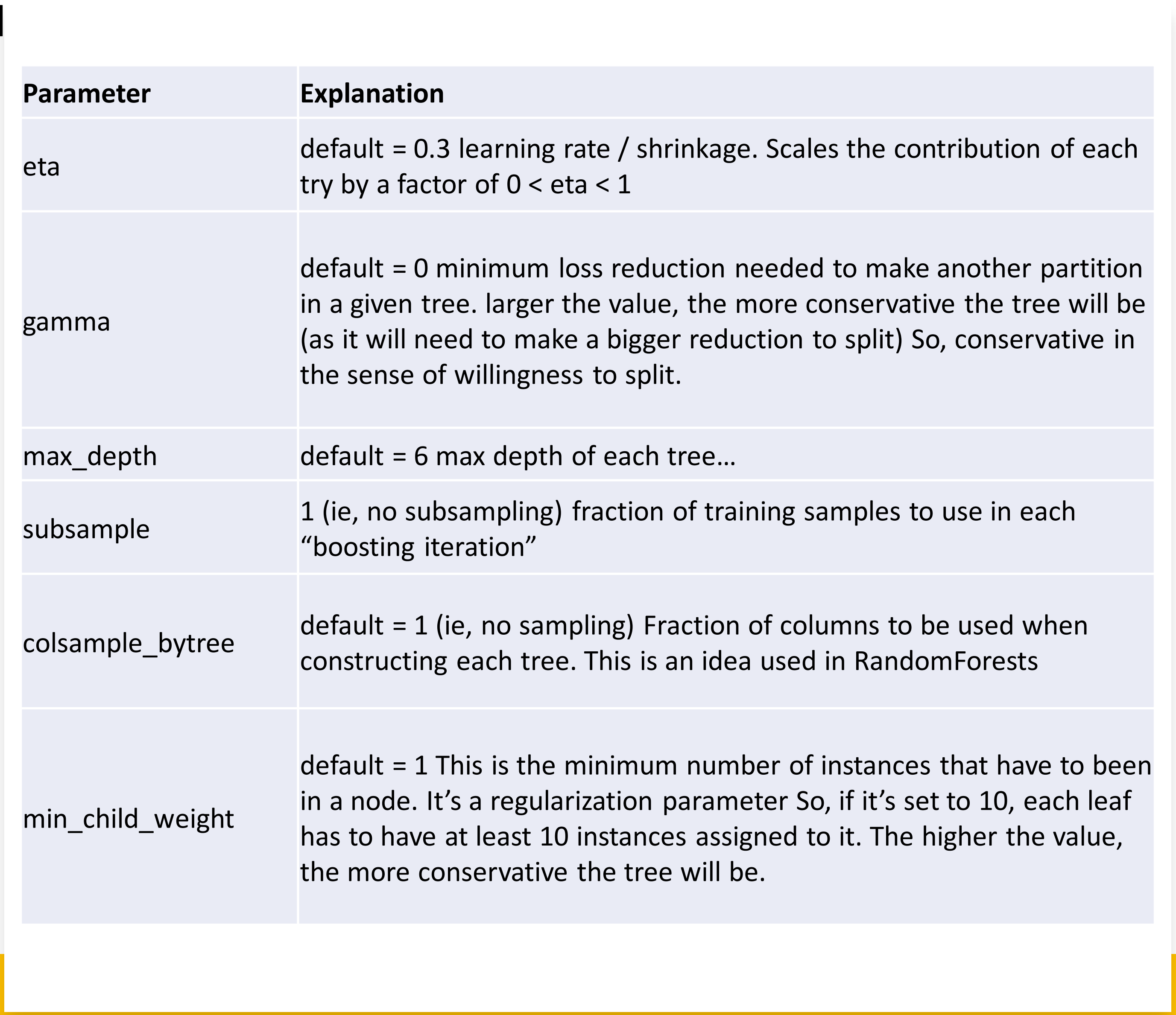
Different Gradient Boosting implementations, you can use with caret R package.

Model	method Value	Type	Libraries	Tuning Parameters
Stochastic Gradient Boosting	gbm	Classification, Regression	gbm, plyr	n.trees, interaction.depth, shrinkage, n.minobsinnode
Gradient Boosting Machines	gbm_h2o	Classification, Regression	h2o	ntrees, max_depth, min_rows, learn_rate, col_sample_rate
eXtreme Gradient Boosting	xgbDART	Classification, Regression	xgboost, plyr	nrounds, max_depth, eta, gamma, subsample, colsample_bytree, rate_drop, skip_drop, min_child_weight
eXtreme Gradient Boosting	xgbLinear	Classification, Regression	xgboost	nrounds, lambda, alpha, eta
eXtreme Gradient Boosting	xgbTree	Classification, Regression	xgboost, plyr	nrounds, max_depth, eta, gamma, colsample_bytree, min_child_weight, subsample

eXtreme Boosting = Better speed + performance

- It is decision tree-based ensemble machine learning algorithm.
- In XGBoost we generally have decision tree sequentially. Hence this is called sequential ensemble technique.
- Since XGBoost uses decision tree up to some depth than there will be high bias and low variance.
- Very high in predictive power but relatively slow with implementation
- XGBoost only works with numeric vectors
- Need to transform data into a specific format that XGBoost can handle. That format is called DMatrix.
- Define the parameters of the gradient boosting ensemble.

Outline some of the more commonly used hyperparameters.



Parameter	Explanation
eta	default = 0.3 learning rate / shrinkage. Scales the contribution of each try by a factor of 0 < eta < 1
gamma	default = 0 minimum loss reduction needed to make another partition in a given tree. larger the value, the more conservative the tree will be (as it will need to make a bigger reduction to split) So, conservative in the sense of willingness to split.
max_depth	default = 6 max depth of each tree…
subsample	1 (ie, no subsampling) fraction of training samples to use in each “boosting iteration”
colsample_bytree	default = 1 (ie, no sampling) Fraction of columns to be used when constructing each tree. This is an idea used in RandomForests
min_child_weight	default = 1 This is the minimum number of instances that have to been in a node. It’s a regularization parameter So, if it’s set to 10, each leaf has to have at least 10 instances assigned to it. The higher the value, the more conservative the tree will be.

Full reference: https://xgboost.readthedocs.io/en/latest/parameter.html

With this guide, you discovered the three standard ensemble learning techniques for machine learning and how to code this algorithms in the R programming language. Specifically, you learned:

- Bagging involves fitting many decision trees on different samples of the same dataset and averaging the predictions.
- Stacking involves fitting many different models, types on the same data and using another model to learn how to best combine the predictions.
- Boosting involves adding ensemble members sequentially that correct the predictions made by prior models and outputs a weighted average of the predictions.

BeVera