Random Forests

Ensemble Learning

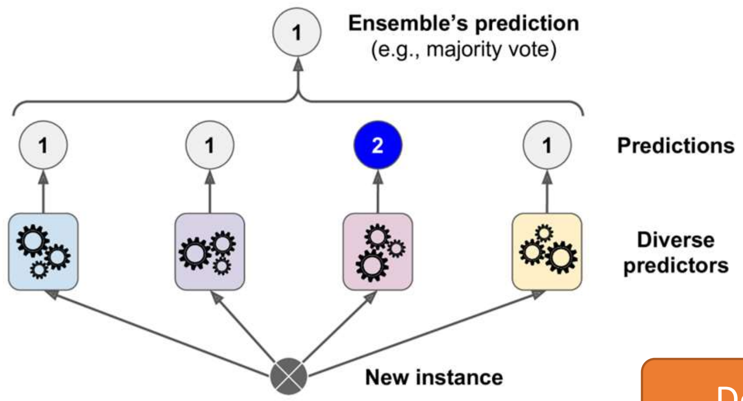
- Suppose you ask a complex question to thousands of random people, then aggregate their answers. In many cases you will find that this aggregated answer is better than an expert’s answer. This is called the wisdom of the crowd.
- Similarly, if you aggregate the predictions of a group of predictors (such as classifiers or regressors), you will often get better predictions than with the best individual predictor.
- A group of predictors is called an ensemble; thus, this technique is called **Ensemble Learning**, and an Ensemble Learning algorithm is called an **Ensemble method**.

Ensemble Learning

- Ensemble Methods can be used for various reasons, mainly to:
  - Decrease Variance (Bagging)
  - Decrease Bias (Boosting)
  - Improve Predictions (Stacking)

Voting Classifier

- A classifier that aggregates the predictions of each classifier and predict the class that gets most votes is called as Voting Classifier
- The majority vote classifier is called a hard voting classifier
- Predicting the class with the highest class probability, averaged over all the individual classifiers is called soft voting



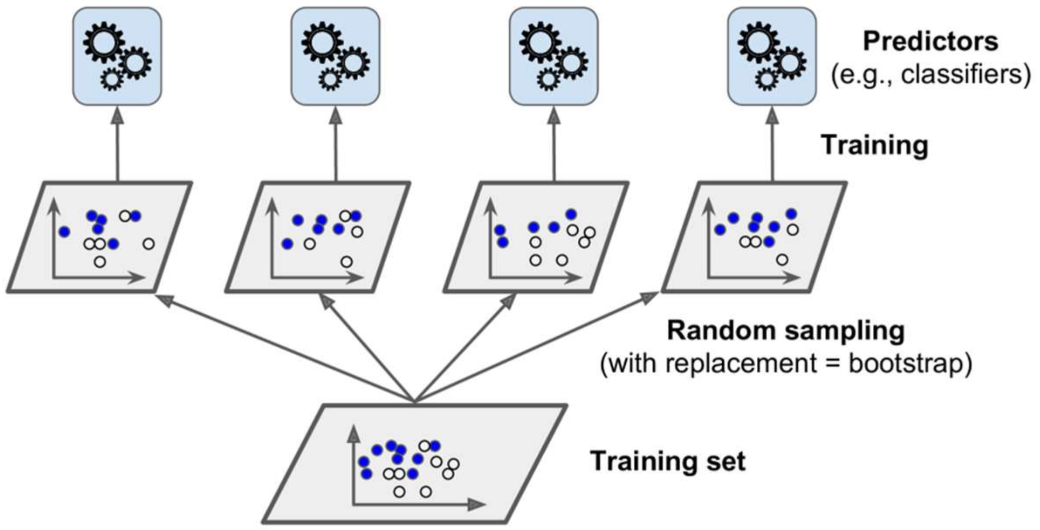
Demo

Bagging and Pasting

- One way to get a diverse set of classifiers is to use very different training algorithms
- Another approach is to use the same training algorithm for every predictor, but to train them on different random subsets of the training set.
- When sampling is performed with replacement, this method is called **bagging (Bootstrap Aggregating)**
- When sampling is performed without replacement, it is called **pasting**

Bagging and Pasting

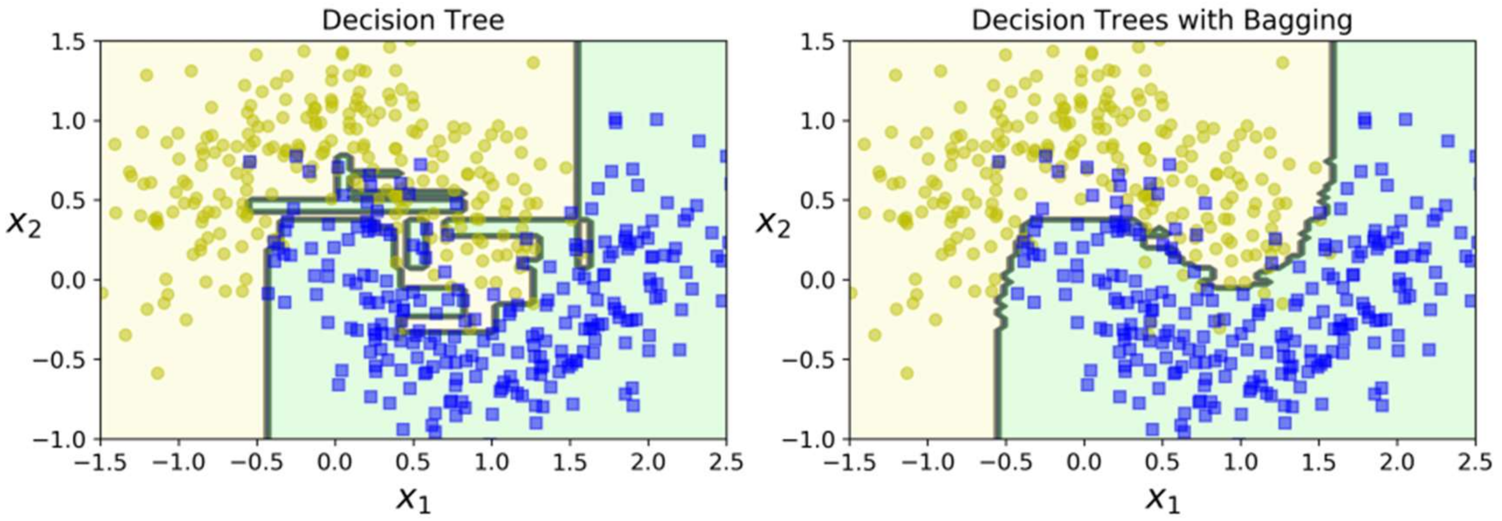
- Both bagging and pasting allow training instances to be sampled several times across multiple predictors, but only bagging allows training instances to be sampled several times for the same predictor



Predictors can be trained in parallel using several CPU cores or servers. This is one of the reasons why bagging and pasting are popular. They scale very well

Demo

Bagging and Pasting



Out-of-Bag Evaluation

- With bagging, some instances may be sampled several times for any given predictor, while others may not be sampled at all.
- By default a **BaggingClassifier** samples m training instances with replacement (bootstrap=True), where m is the size of the training set. This means that only about 63% of the training instances are sampled on average for each predictor.
- The remaining 37% of the training instances that are not sampled are called out-of-bag (oob) instances.
- In Scikit-Learn, you can set **oob_score=True** when creating a BaggingClassifier to request an automatic oob evaluation after training.

Random Forest

- A Random Forest is an ensemble of Decision Trees, generally trained via the bagging method (or sometimes pasting), typically with **max_samples** set to the size of the training set.
- Instead of building a **BaggingClassifier** and passing it a **DecisionTreeClassifier**, you can instead use the **RandomForestClassifier** class, which is more convenient and optimized for Decision Trees

Demo

Advantages of Random Forests

- Reduces overfitting, reduces variance and therefore improves the accuracy
- Can solve both classification and regression problems Demo
- Can automatically handle missing values
- No feature scaling required
- Handles non-linear parameters efficiently
- Robust to outliers and can handle them automatically
- Very stable and less impacted by noise

Disadvantages of Random Forests

- Complexity: It creates a lot of trees and combines their outputs
- Longer training period
- Consumes more computation power

Feature Importance

- Yet another great quality of Random Forests is that they make it easy to measure the relative importance of each feature.
- Scikit-Learn measures a feature's importance by looking at how much the tree nodes that use that feature reduce impurity on average (across all trees in the forest)
- You can access the result using the **feature_importances_** variable
- After running **RandomForestClassifier** on the iris dataset we see that the most important features are the petal length (44%) and width (42%), while sepal length and width are rather unimportant in comparison (11% and 2%, respectively)

Demo

Boosting

- Boosting (originally called hypothesis boosting) refers to any Ensemble method that can combine several weak learners into a strong learner.
- The general idea of most boosting methods is to train predictors sequentially, each trying to correct its predecessor
- There are many boosting methods: AdaBoost, Gradient Boosting and XGBoost

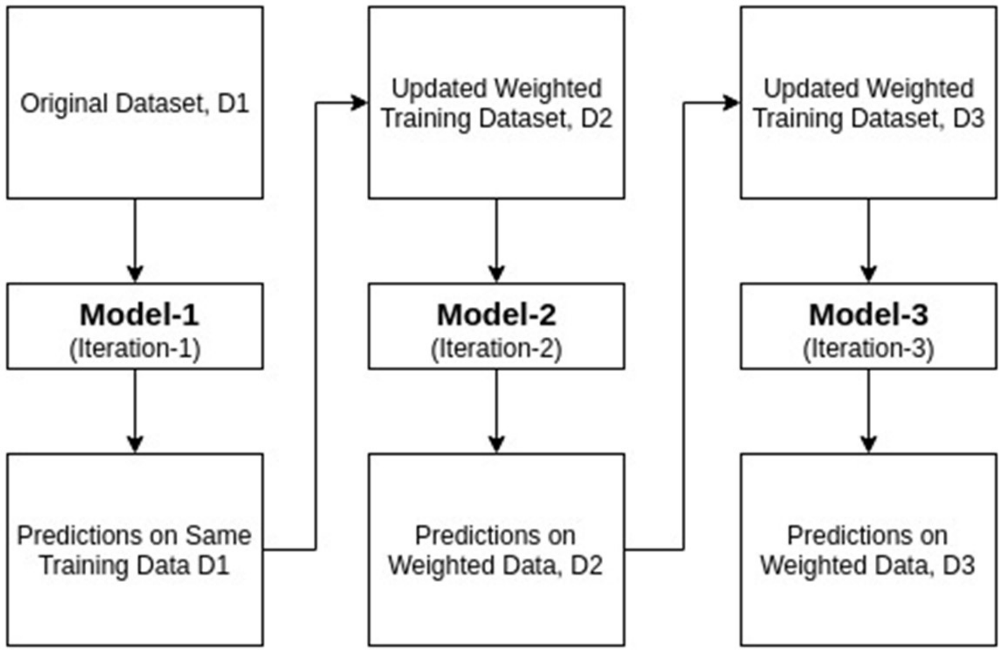
AdaBoost

- One way for a new predictor to correct its predecessor is to pay a bit more attention to the training instances that the predecessor underfitted.
- This results in new predictors focusing more and more on the hard cases. This is the technique used by AdaBoost.
- AdaBoost is an iterative ensemble method
- AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier.
- The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations.

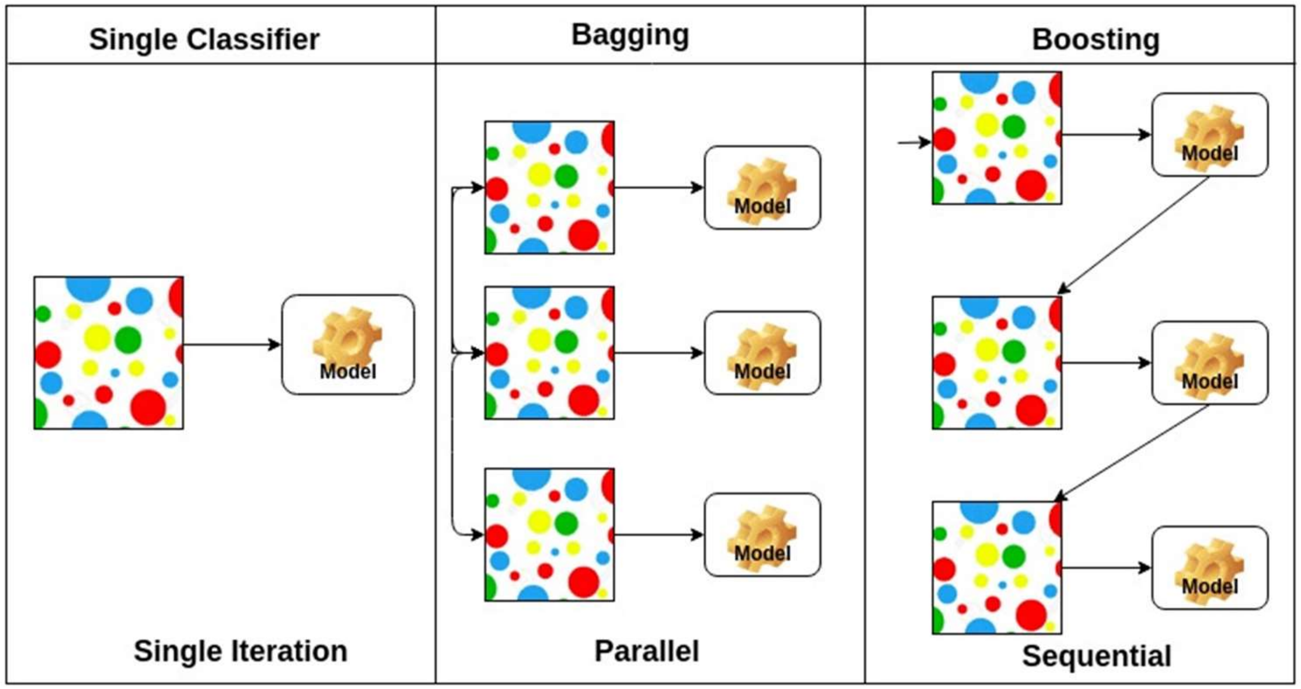
AdaBoost

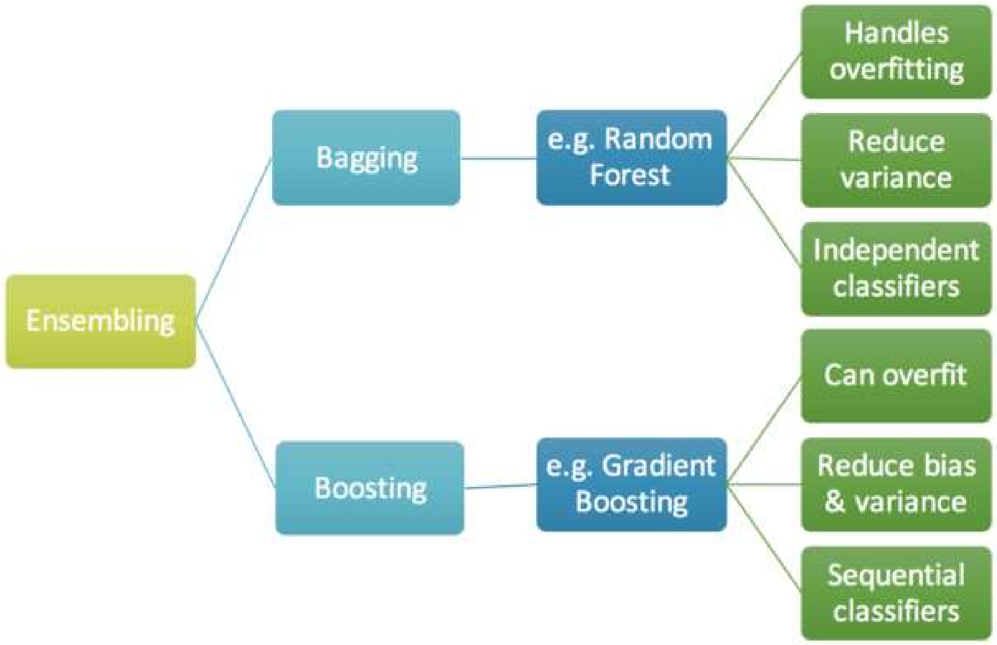
- Initially, Adaboost selects a training subset randomly.
- It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.
- It assigns the higher weight to wrong classified observations so that in the next iteration these observations will get the high probability for classification.
- Also, It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.
- This process iterate until the complete training data fits without any error or until reached to the specified maximum number of estimators.
- To classify, perform a "vote" across all of the learning algorithms you built.

AdaBoost


Original Dataset, D1
Model-1
(Iteration-1)
Predictions on Same Training Data D1
Updated Weighted Training Dataset, D2
Model-2
(Iteration-2)
Predictions on Weighted Data, D2
Updated Weighted Training Dataset, D3
Model-3
(Iteration-3)
Predictions on Weighted Data, D3

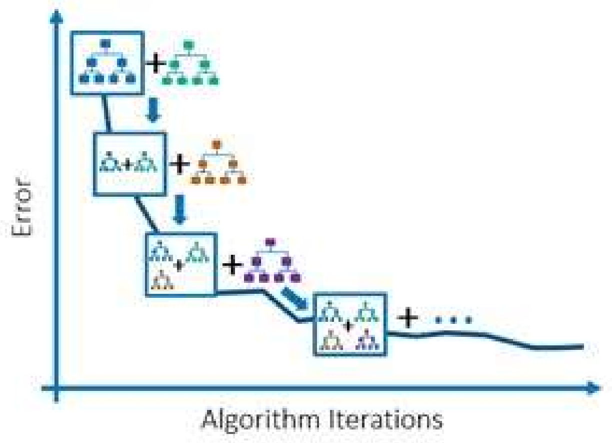
Bagging Vs Boosting




Ensembling
Bagging
e.g. Random Forest
Handles overfitting
Reduce variance
Independent classifiers
Boosting
e.g. Gradient Boosting
Can overfit
Reduce bias & variance
Sequential classifiers

Early Stopping

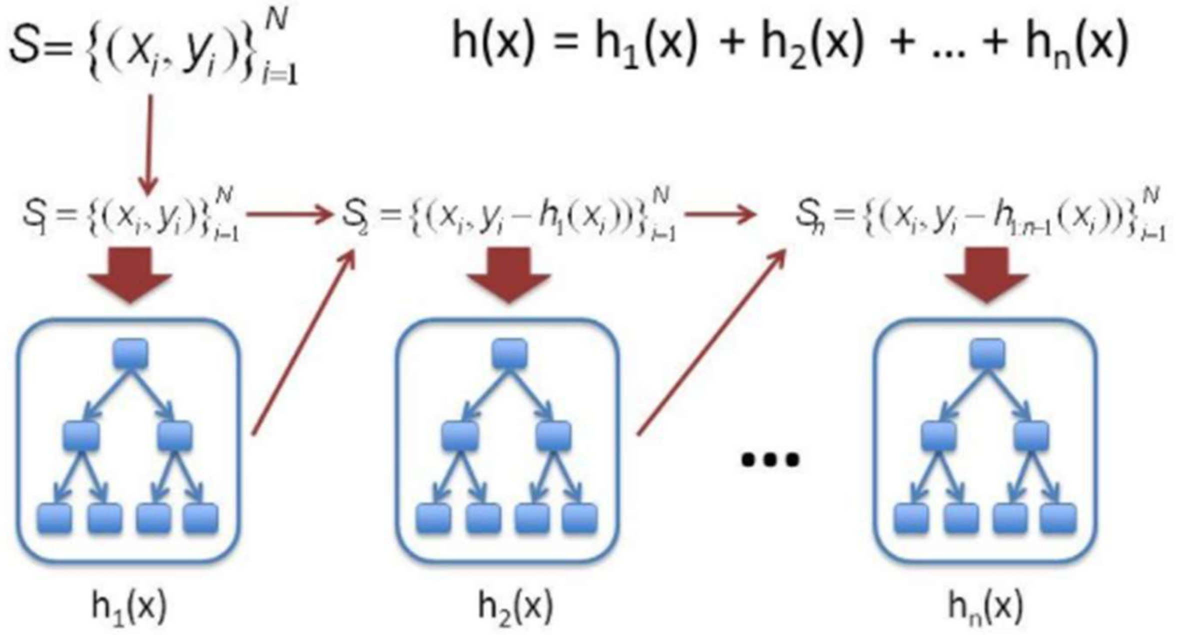
- Finishes training of the model early if the hold-out metric ("rmse" for example) does not improve for a given number of rounds.



Gradient Boosting

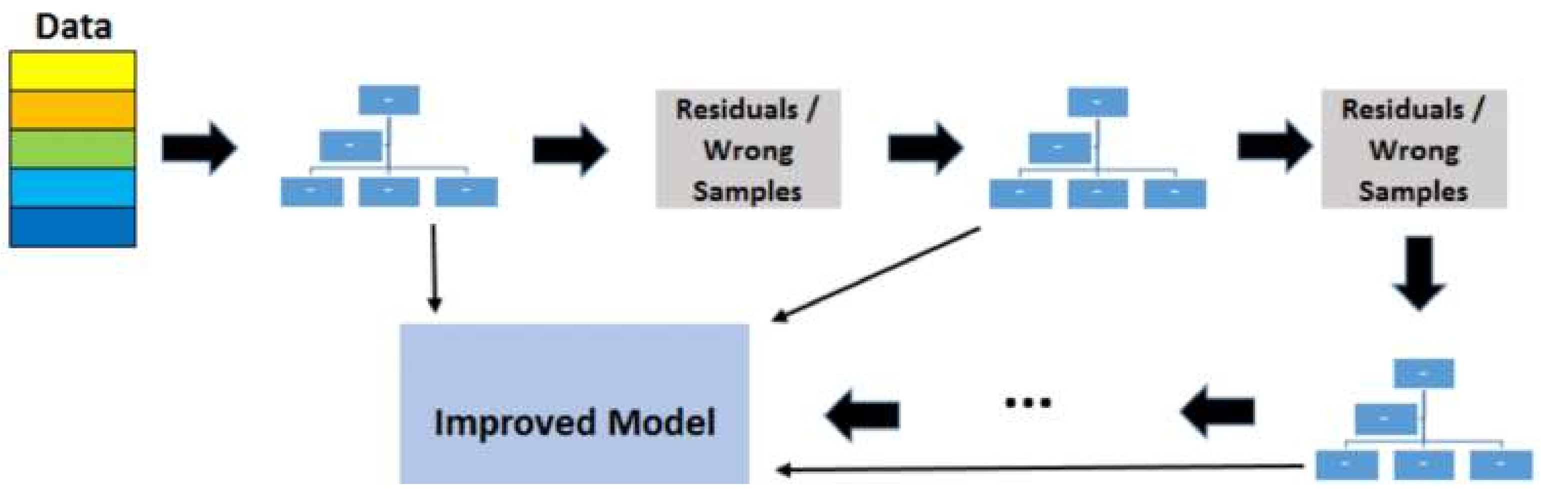
- In gradient boosting decision trees, we combine many weak learners to come up with one strong learner. The weak learners here are the individual decision trees.
- All the trees are connected in series and each tree tries to minimize the error of the previous tree.
- Due to this sequential connection, boosting algorithms are usually slow to learn (controllable by the developer using the learning rate parameter), but also highly accurate.

Gradient Boosting



$S = \{(x_i, y_i)\}_{i=1}^{N}$
$h(x) = h_1(x) + h_2(x) + \ldots + h_n(x)$
$S_1 = \{(x_i, y_i)\}_{i=1}^{N}$
$S_2 = \{(x_i, y_i - h_1(x_i))\}_{i=1}^{N}$
$S_n = \{(x_i, y_i - h_{1:n-1}(x_i))\}_{i=1}^{N}$
$h_1(x)$
$h_2(x)$
$h_n(x)$

Gradient Boosting



Gradient Boosting

- Gradient Boost has three main components
  - Loss Function: The role of the loss function is to estimate how best is the model in making predictions with the given data.
  - Weak Learner: Weak learner is one that classifies the data so poorly when compared to random guessing
  - Additive Model: It is an iterative and sequential process in adding the decision trees one step at a time
- Improvements to Gradient Boosting
  - Subsampling: At each learning iteration, only a random part of the training data is used to fit a consecutive base-learner. The training data is sampled without replacement.
  - Shrinkage: shrinks regression coefficients to zero and, thus, reduces the impact of potentially unstable regression coefficients
  - Early Stopping

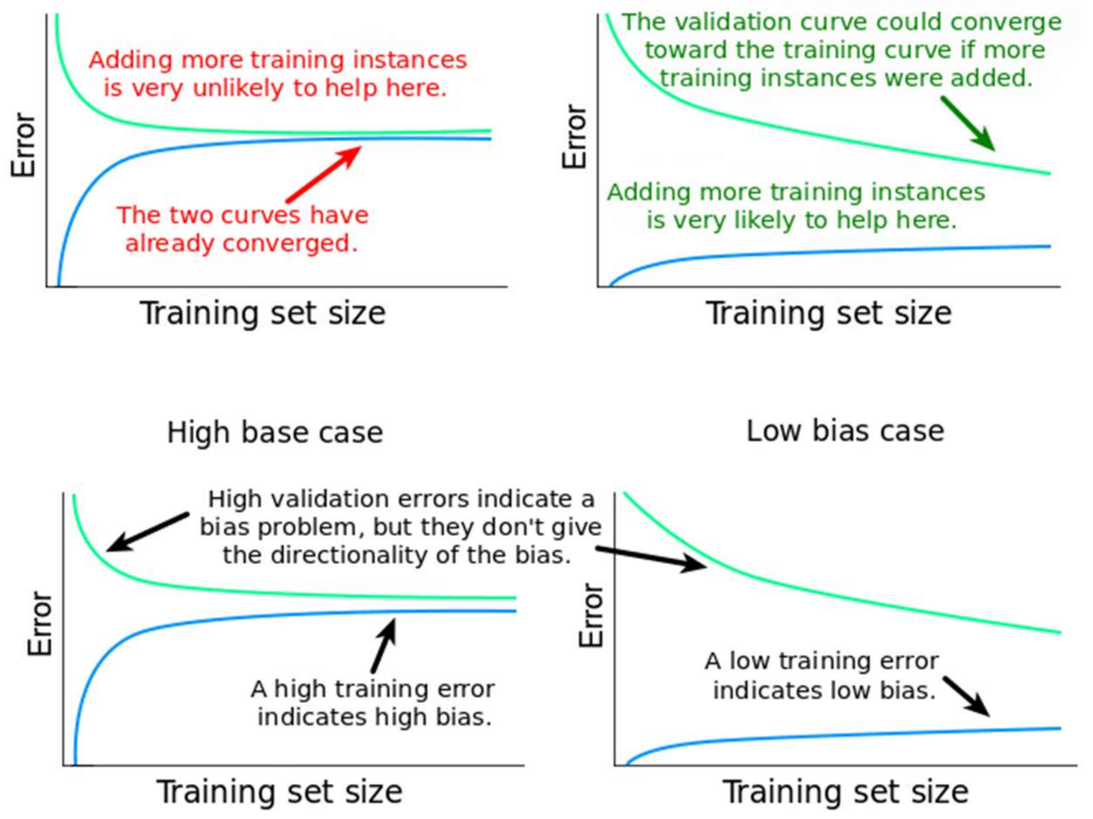
XGBoost

- XGBoost stands for “Extreme Gradient Boosting”
- Features:
  - Regularized Learning: avoids over fitting
  - Gradient Tree Boosting: trained in additive manner
  - Shrinkage and column sub-sampling: Column sub-sampling prevents over-fitting even more so than the traditional row sub-sampling, also speeds up computation
- Goals of XGBoost:
  - Execution speed
  - Model performance: XGBoost dominates structured or tabular datasets on classification and regression predictive modelling problems

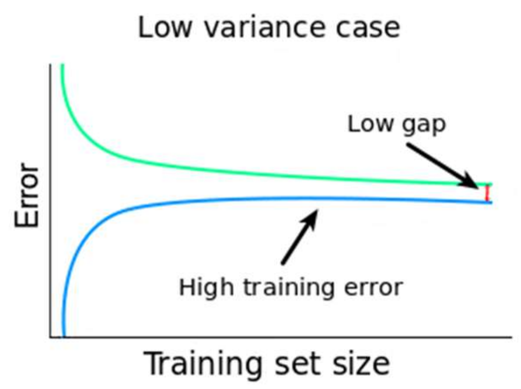
Learning Curve

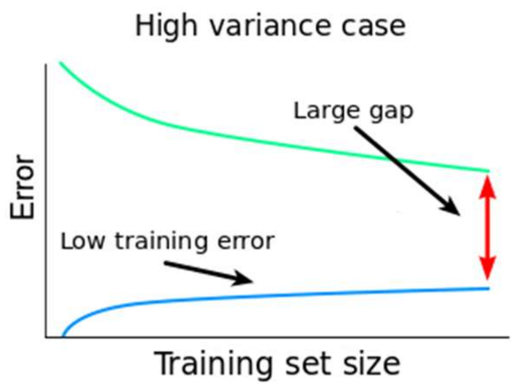
- A learning curve is a plot of model learning performance over experience or time
- Learning curves are a widely used diagnostic tool in machine learning for algorithms that learn from a training dataset incrementally.
- The model can be evaluated on the training dataset and on a hold out validation dataset after each update during training
- Plots of the measured performance can created to show learning curves.

Learning Curve

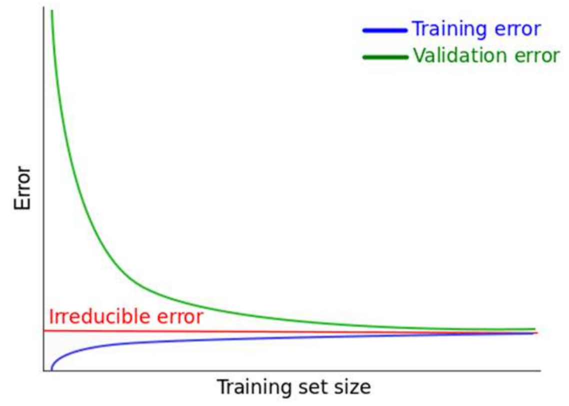


Learning Curve





Ideal Learning Curve


Training error
Validation error
Irreducible error
Error
Training set size