11.1 What's Integrated Algorithm?

Integrated Algorithm can also be called ensemble algorithm, which is used for classification and regression. **The core idea of integrated algorithm is to use various algorithm together so that we can solve a problem well.** There are three main parts for integrated algorithm: bagging, boosting and stacking.

11.1.1 Bagging

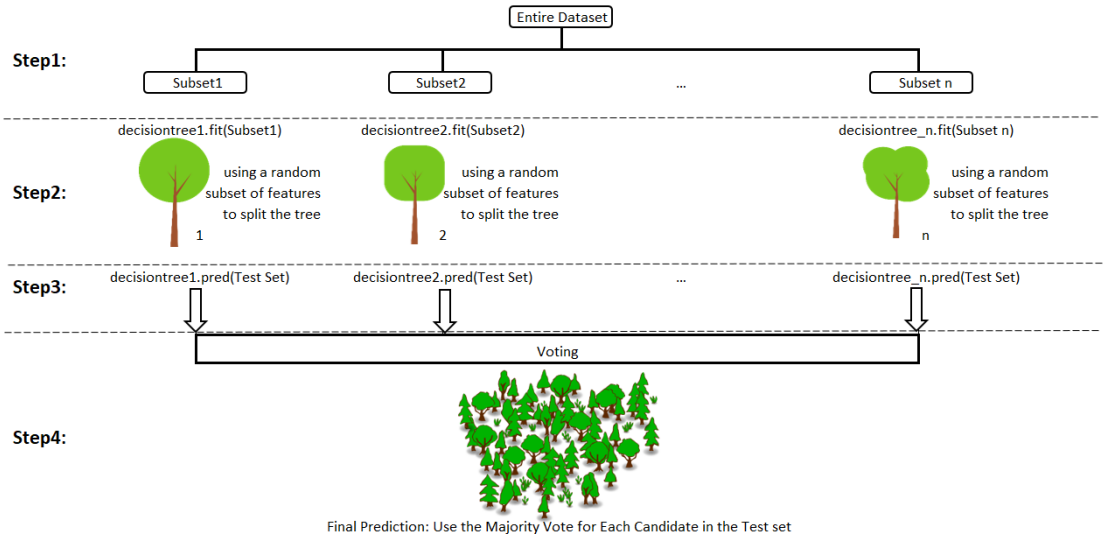
Bagging, which is short for Boostrap aggregation, is a kind of **parallel** type of integrated learning method. **The base learner can be done at the same time**. Bagging using "back" sampling method to select training set, for training set containing m sample, we randomly pick samples m times with "back". There is a close probability of 36.8% of the sample not be picked:

$$\lim_{m\to\infty}\left(1-\frac{1}{m}\right)^{m}\to\frac{1}{e}\approx 0.368 \tag{1}$$

Repeated in the same way, we collect T data sets containing m samples to train T base learners, and finally combine the outputs of these T base learners and average them.

$$f(x)=\frac{1}{T}\sum_{m=1}^{T}f_m(x) \tag{2}$$

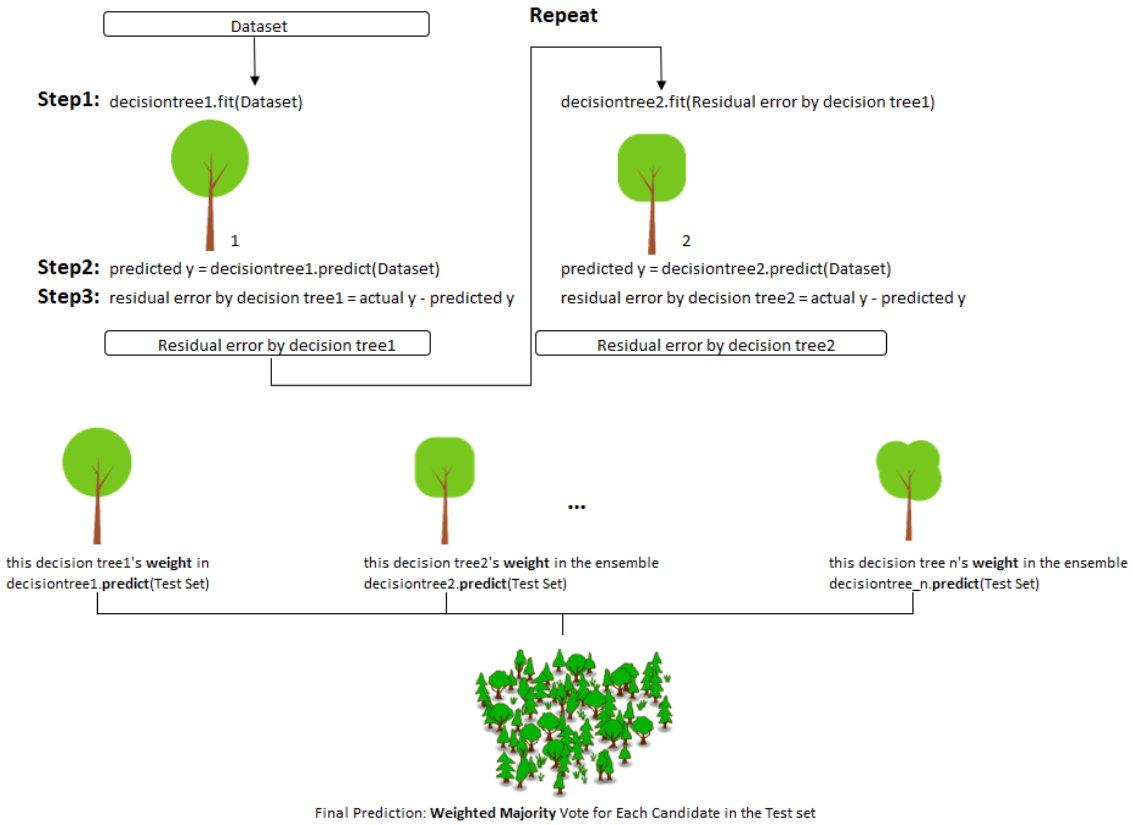
Random Forest is a classic bagging algorithm whose base learner is consisted of with many decision trees. And the "Random" means:
(1) random sample selection, freely choose samples from the whole data set (for example, every time 80% as the input data for different learners) and put them back into dataset and repick
(2) random feature selection, randomly select features as the input data for different learners. Based on the above random selection, we can make different learner with different models, which makes more sense for bagging algorithm. Random forest structure is shown in the following figure:



Final Prediction: Use the Majority Vote for Each Candidate in the Test set

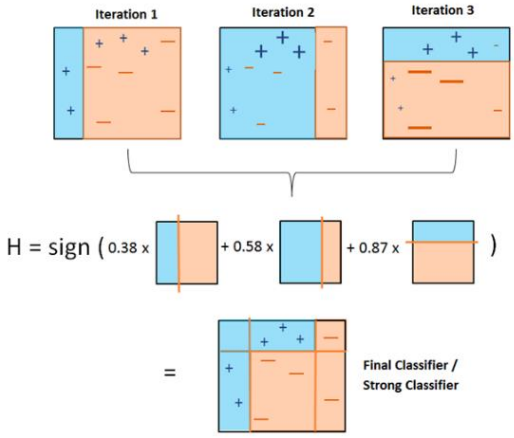
11.1.2 Boosting

Boosting is a kind of **serial working mechanism**. The basic idea is: to increase the weight for samples which got predicted wrong, make the follow-up learner to pay more attention to the marking error of the training sample, as far as possible to correct these errors. Then we add all learners as a serial model. Boosting structure is shown in the following figure:



Final Prediction: **Weighted Majority** Vote for Each Candidate in the Test set

- **Adaboost:**
  Specifically, the whole Adaboost iterative algorithm is divided into three steps:
  1. Initializes the weight distribution of the training data. If there are N samples, each training sample is initially given the same weight Value: 1 / N.
  2. Train the weak classifier. In the specific training process, if a sample point has been accurately classified, the next training set needs to reduce its weight; Conversely, if a sample point is not accurately classified, its weight is increased. And then, we use the updated sample set with update weights to train the next classifier, and the whole training process continues iteratively.
  3. The weak classifier from each training is combined into strong classifier. Then we need to increase the weight of the weak classifier with small rate so that it can play a big role in the final classification function. Meanwhile, we need to decrease the weight of the weak classifier with big rate so that it can play a less role in the final classification function.



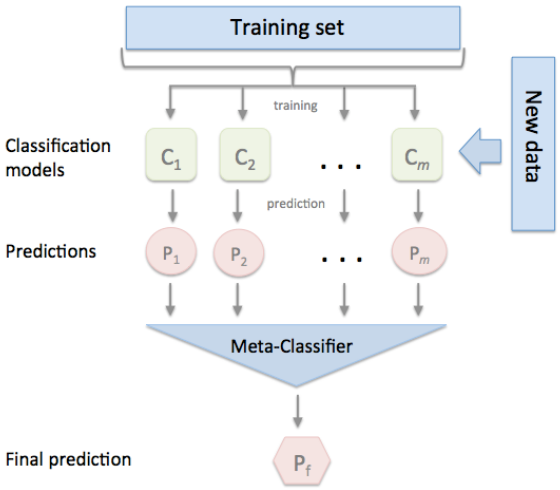
- **Xgboost: Extreme Gradient Boosting**
  Firstly, xgboost is an efficient system implementation of Gradient Boosting. In addition to tree(gbtree), linear classifier (gblinear) can also be used for base learner in xgboost. GBDT in particular refers to the gradient boosting decision tree algorithm.
  https://www.zhihu.com/question/41354392

$$
\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\cdots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned}
$$

第t轮的模型预测　　保留前面t-1轮的模型预测　　加入一个新的函数

11.1.3 Stacking

For bagging and boosting algorithms, their base learner must be same, especially the base learner of random forest algorithm in bagging must be decision tree. But in stacking algorithm, we can use all kinds of machine learning algorithm to build model in the first phase, and sent the outputs from first phase to the overall classifier. Stacking structure is shown in the following figure:



11.2 Combine strategies

Combining strategies refer to how to combine the outputs of the base learners to produce the final output of the integration model after training the base learners. The following are some common combining strategies:

11.2.1 Average method (regression problem)

- **Simple averaging:**

$$f(x) = \frac{1}{T}\sum_{m=1}^{T} f_m(x) \tag{3}$$

- **Weighted averaging:**

$$f(x) = \sum_{m=1}^{T} w_m \cdot f_m(x) \tag{4}$$

As the weights of each base learner are obtained in the training, generally speaking, the weighted average method is used when the performance difference of the individual learner is large, and the simple average method is used when the performance difference of the individual learner is small.

11.2.2 Voting method (classification)

- **Majority voting:**

绝对多数投票法(majority voting) 必须要占一半以上

$$H(\boldsymbol{x}) = \begin{cases} c_j, & \text{if } \sum_{i=1}^{T} h_i^j(\boldsymbol{x}) > 0.5 \sum_{k=1}^{N}\sum_{i=1}^{T} h_i^k(\boldsymbol{x}); \\ \text{reject}, & \text{otherwise.} \end{cases}$$

- **Plurality voting:**

相对多数投票法(plurality voting) 最多票数即可

$$H(\boldsymbol{x}) = c_{\arg\max_j \sum_{i=1}^{T} h_i^j(\boldsymbol{x})}.$$

11.3. Diversity

In integrated learning, the diversity among the base learners is an important factor affecting the generalization performance of the integrator. Therefore, increasing diversity is very important for integrated learning research. The general idea is to introduce randomness into the learning process, and the common practice is to disturb data samples, input attributes, output representation and algorithm parameters.

- **data sample disturbance**, that is, using different data sets to train different base learners. For example, there is a fallback self - sampling method, but this method is only effective for unstable learning algorithms, such as decision trees and neural networks.
- **input attribute disturbance**, that is, a subspace of the original space is randomly selected to train the base learner. For example, random forest, extract a subset from the initial attribute set, and train the base learner based on each subset. However, if the training set contains only a few attributes, it is not appropriate to use attribute perturbation.
- **the output represents the disturbance**, which can slightly change the class standard of the training sample or transform the output of the base learner.
- **algorithm parameter disturbance**, through the random setting of different parameters, such as: neural network, random initialization weight and random setting of hidden layer node number. You will also see here that integrated learning is essentially a generic framework that can use any base learner to improve the generalization performance of a single learner.