Ensemble Learning

An ensemble method tries to make a strong learner by combining the predictions of many weak learners (**whose accuracy is a little bit higher than random learner**). Actually, for binary classification, we could prove that the majority vote makes correct decision if the weak classifies are **independent** to each other. It's called condorcet's rule. Two core elements have to be considered in ensemble learning: accuracy and diversity.

Ensemble learning is a method which could be used in both regression and classification and now a lot of ensemble algorithms are tree-based. There are two strategies to ensemble weak learners: boosting and bagging.

Boosting

Boosting is composed of:

- additive model $F(x) = \Sigma_{m=1}^{M} \alpha_m g_m(x)$, where $g_m(x)$ is a weak learner.
- reweighted samples after training each weak learner.

The most famous algorithm is Adaboost.

1. Algorithm - Adaboost

Input

- Training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$
- Algorithm parameter: Number M of weak learners

Training algorithm

1. Initialize the observation weights $w_i = \frac{1}{n}$ for $i = 1, 2, ..., n$.
2. For $m = 1$ to M:

   2.1 Fit a classifier $g_m(x)$ to the training data using weights w_i.

   2.2 Compute
   $$\text{err}_m := \frac{\sum_{i=1}^{n} w_i \mathbb{I}\{y_i \neq g_m(x_i)\}}{\sum_i w_i}$$

   2.3 Compute $\alpha_m = \frac{1}{2}\log(\frac{1-\text{err}_m}{\text{err}_m})$

   2.4 Set $w_i \leftarrow w_i \cdot \exp(\alpha_m \cdot \mathbb{I}(y_i \neq g_m(x_i)))$ for $i = 1, 2, ..., n$.
3. Output
   $$f(x) := \text{sign}\left(\sum_{m=1}^{M} \alpha_m g_m(x)\right)$$

5 / 29

Note: 1. standard adaboost could only be used in binary classification problem; 2. the weak learn whose error rate is larger than 0.5 would be discarded in iteration.

The idea of adaboost is to make the sample which is classified into a wrong class currently have a larger weight. In this case, if we want to minimize our loss, it's better to classify these wrong-classified samples correctly in later iterations.

2. Loss Function - Adaboost

The loss function used in Adaboost is exponential loss function:

$$L(y, f(x)) = e^{-yf(x)}$$

Actually we could derivate the updating policy for parameters in Adaboost from the perspective of minimizing loss function. The final policy is the same as the first part stated. The specific derivation could be found a chinese version in here.

3. Additionals

- How to use it in multi-classification problem?

  As we stated before, the standard adaboost could only be used in binary classification problem. What if we want to apply this algorithm for multi-classes problem? We could transform the standard algorithm in three ways:

  - choose weak learners that can handle multiple classification.
  - still use binary weak learners and combine the output and features into new sample set. Then train a multi-classifier based on the new sample set.
  - encode label by using m bits binary code and then train m weak learners. The final output is the label we predict.