Ensemble Methods

* Combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator

① Averaging Methods

- build several estimators independently & then average predictions
- low variance
- eg: bagging, forests of randomized trees

② Boosting Methods

- base estimators are built sequentially
- reduce the bias of the combined estimator
- Combine several weak models to create powerful ensembles.

Bagging Meta-Estimator

* Builds several instances of a black-box estimator on random subsets of the original training set. & aggregate their individual predictions to form a final prediction.

* Reduces variance of the base estimator by introducing randomization in to its construction

* reduces overfitting

* work best with strong & complex models.

* Different bagging methods:
  - Pasting
  - Bagging
  - Random Subspaces
  - Random Patches

Forests of Randomized Trees

- diverse set of trees are created by introducing randomness in the classifier construction.
- The prediction of the ensemble is given by the averaged prediction of individual classifiers.

Random Forests

- each tree in the ensemble is built from a sample drawn with replacement. (bootstrap sample) from the training set.
- Best split (at each node) is found either from all-input features or a random subset of features.

Soures of randomness

* The injection of randomness → trees with decoupled prediction errors.
* By averaging those predictions some errors cancel out → ↓ variance of error → ↓ model error

Feature Importance Evaluation

* features used at the top of the tree contribute to the final prediction decision of a larger fraction of input samples.

Expect Fraction of samples $\xrightarrow{\text{estimate}}$ relative importance of features.

* Mean Decrease in Impurity (MDI)

Ada Boost

* fit a sequence of weak learners (only better than random guessing) on repeatedly modified data.
* The predictions of weak learners are combined through a majority vote or a sum to get the final prediction.
* Boosting → add weights $w_1, w_2, \dots, w_N$ to all the training samples (initially $w_i = 1/N$)
* In later steps, learning occurs on reweighted data.

* wrongly predicted samples from previous iteration, get higher weights in the next iteration & vice versa.
* ∴ each weak leaner is forced concentrate on samples that were previously misclassified.

Gradient Tree Boosting

* generalizes boosting to arbitrarily differentiable loss function.

* Regression with GBRT

$$\hat{y}_i = F_M(x_i) = \sum_{m=1}^{M} h_m(x_i)$$

where h_m is a weak learner.

$$F_m(x) = F_{m-1}(x) + h_m(x)$$

newly added weak learner (shallow tree)

h_m is fitted s.t.

$$h_m = \arg\min_h h_m = \arg\min_h \sum_{i=1}^{n} \underbrace{l\left(y_i, F_{m-1}(x_i) + h(x_i)\right)}_{\text{loss}}$$

F_0 is initialized as a constant value (learns the empirical mean)

using first order Taylor Approximation ⇒

$$l(z) = l(a) + (z-a)\frac{\partial l(a)}{\partial a}$$

$$\therefore \quad l(y_i, F_{m-1}(x_i) + h(x_i)) = l(y_i, F_{m-1}(x_i)) + h_m(x_i)\underbrace{\left[\frac{\partial l(y_i, F(x_i))}{\partial F(x_i)}\right]_{F=F_{m-1}}}_{g_i}$$

Then,

$$h_m \approx \underset{h}{\operatorname{argmin}} \sum_{i=1}^{n} h(x_i)\, g_i$$

This is minimized if $h(x_i)$ is fitted to predict a negative gradient $-g_i$ vale.

* Classification with GBRT

$$P(y_i = 1 \mid x_i) = \sigma(F_M(x_i))$$ ← binary classification

sigmoid function

for multi-class classification (K) ⇒ K trees are built at each M iteration.

$$p(y_i = k \mid x_i) = \mathrm{Softmax}(F_{M,k}(x_i))$$

* even for classification h is still a regressor.