Bagging and Model Averaging

Wei Li

Syracuse University

Spring 2024

OVERVIEW

Introduction

Bagging in Regression problems

Bagging in Classification

Model Averaging

Bumping

Introduction

Introduction

Bootstrap aggregating (bagging) and boosting are useful techniques to improve the predictive performance of models.

- Boosting may also be useful in connection with many other models, e.g. for additive models with high-dimensional predictors
- bagging is most prominent for improving tree algorithms.

Bagging in Regression problems

Bagging in Regression problems

Let training data be $z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, from z, obtain the prediction $\hat{f}(x)$ at input x.

Bagging works as follows:

1. Generate a bootstrap sample

$$(X_1^*, Y_1^*), \ldots, (X_n^*, Y_n^*)$$

   and compute the bootstrapped estimator $\hat{f}^*(\cdot)$.
2. Repeat step B times, yielding

$$\hat{f}^{*(1)}(\cdot), \ldots, \hat{f}^{*(B)}(\cdot)$$

3. Aggregate the bootstrap estimates

$$\hat{f}_{bag}(\cdot) = B^{-1} \sum_{b=1}^{B} \hat{f}^{*(b)}(\cdot)$$

The bagging estimator is essentially (with $B = \infty$)

$$\hat{f}_{bag}(\cdot) \cong \mathrm{E}^*\big(\hat{f}^*(\cdot)\big)$$

here, E^* is with respect to the bootstrap distribution

The bagged estimate will differ from the original estimate $\hat{f}(x)$ **only when** the latter is a nonlinear or adaptive function of the data.

An identity indicates properties of bagging: with $B = \infty$

$$\hat{f}_{bag}(\cdot) \cong \hat{f}(\cdot) + \Big(\mathrm{E}^*\big(\hat{f}^*(\cdot)\big) - \hat{f}(\cdot)\Big)$$
$$\cong \hat{f}(\cdot) + \text{ bootstrap bias estimate.}$$

- Adding the bootstrap bias estimate

What we can hope for is a variance reduction at the price of a higher bias. How?

- An **ideal aggregate estimator** $f_{\mathrm{ag}}(\cdot) := \mathrm{E}_{\mathbb{P}}\hat{f}(\cdot)$ never increases MSE.
- By the bootstrap principle

$$f_{\mathrm{ag}}(x) := \mathrm{E}_{\mathbb{P}}(\hat{f}(x)) \approx \mathrm{E}^*(\hat{f}^*(x)) =: \hat{f}_{bag}(x)$$

  we hope that using $\hat{f}_{bag}(x)$ can decrease MSE
  - the decrease in MSE is possible when $\hat{f}(\cdot)$ has high variance (e.g., trees)

Bagging in Classification

Hard classification

Hard classification: If $\hat{f}^{*(b)}(x)$ is indicator-vector, with one 1 and $K-1$ 0's (**hard classification**), equivalently, for some $\hat{h}^{*(b)}(x) \in \{1, \ldots, K\}$.

- "Consensus vote" (majority vote): selects the most "votes" from the B classifiers.

$$\hat{f}_{bag,k}(x) = \frac{1}{B} \sum_{b=1}^{B} 1\{\hat{h}^{*(b)}(x) = k\}$$

so the classifier is

$$\hat{f}_{bag}(x) = \arg\max_k \hat{f}_{bag,k}(x)$$

Soft classification

If $\hat{f}^{*(b)}(x) = \left(\hat{p}_1^{*(b)}, \ldots, \hat{p}_K^{*(b)}\right)$, the estimates of class probabilities $\hat{p}_k^{*(b)} := \hat{P}^{(b)}(Y = k \mid X = x), k = 1, \cdots, K$.

"averaging probabilities": the bagged estimates are the average prediction at x from B classifiers

$$\hat{p}_{\text{bag},k}(x) = B^{-1} \sum_{b=1}^{B} \hat{p}_k^{*(b)}(x), \quad k = 1, \cdots, K$$

If a predicted label is desired,

$$\hat{p}_{bag}(x) = \arg\max_k \hat{f}_{\text{bag},k}(x)$$

Effects of bagging

In the classification under $0-1$ loss, bagging a good classifier can make it better, but bagging a bad classifier can make it worse.

The **Wisdom of Crowds** asserts that the collective knowledge of a diverse and independent body of people typically exceeds the knowledge of any single individual and can be harnessed by voting.

The Wisdom of Crowds

Consider an example: optimal decision at x be $h^*(x) = 1$ in a two-class example.

- Suppose each of the weak learners $h^{*(b)}$ have an error-rate $e_b = e$, say slightly less than 0.5
- let $S_1(x) = \sum_{b=1}^{B} 1\left(h^{*(b)}(x) = 1\right)$ be the consensus vote for class-1.
- the weak learners are assumed to be **independent**

$$S_1(x) \sim \text{Bin}(B, 1-e),$$

$$\Pr(S_1/B > 1/2) \to 1, \text{ as } B \text{ gets large.}$$

Model Averaging

Bayesian model averaging

- J models, denoted by $M_j, j = 1, 2, \ldots, J$.
- Consider the prediction $f(x)$ at some x as a parameter

The posterior distribution of $f(x)$ is

$$\Pr(f(x) \mid Z) = \sum_{m=1}^{M} \Pr\left(f(x) \mid M_m, Z\right) \Pr\left(M_m \mid Z\right)$$

with posterior mean

$$\mathrm{E}(f(x) \mid Z) = \sum_{m=1}^{M} \mathrm{E}\left(f(x) \mid M_m, Z\right) \Pr\left(M_m \mid Z\right)$$

- $\Pr\left(M_m \mid Z\right)$ are the weights proportional to posterior probability of each model.
- The **committee method** uses a simple unweighted average (equal probability).
- Bagging estimate can be viewed as some approximate to the posterior mean.

$\Pr(M_m|Z)$

To obtain $\mathrm{P}\,r(M_m|Z)$, there are two methods

- Use BIC

$$\Pr\left(M_m \mid Z\right) \approx \frac{\exp\left(-\frac{1}{2}BIC_m\right)}{\sum_{l=1}^{J}\exp\left(-\frac{1}{2}BIC_l\right)}$$

- a full Bayesian procedure: suppose each model M_m has parameters θ_m

$$\begin{aligned}\Pr(M_m|Z) &\propto \Pr(M_m)\Pr(Z|M_m)\\ &\propto \Pr(M_m)\int p(Z|M_k,\theta_k)p(\theta_k|M_k)d\theta_k\end{aligned}$$

Frequentist model averaging

Given predictions $\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_M(x)$, under squared-error loss, we can seek the weights $w = (w_1, w_2, \ldots, w_M)$ such that

$$\hat{w} = \underset{w}{\operatorname{argmin}} \quad \mathrm{E}\left(Y - \sum_{m=1}^{M} w_m \hat{f}_m(x)\right)^2$$

The solution is the population LS of Y on $\hat{F}(x)^\top \equiv \left(\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_M(x)\right)$:

$$\hat{w} = \mathrm{E}\left(\hat{F}(x)\hat{F}(x)^\top\right)^{-1} \mathrm{E}(\hat{F}(x)Y)$$

It holds that

$$\mathrm{E}\left(Y - \sum_{m=1}^{M} \hat{w}_m \hat{f}_m(x)\right)^2 \leq \mathrm{E}\left(Y - \hat{f}_m(x)\right)^2, \quad \forall m$$

so combining models never makes things worse at the population level.

Stacking (emsemble)

However, the in-sample analog of the loss won't work:

$$\hat{w}^{\text{st}} = \arg\min_{w} \sum_{i=1}^{n} \left(y_i - \sum_{m=1}^{M} w_m \hat{f}_m(x_i) \right)^2$$

Instead, the stacking weights are solved by

$$\hat{w}^{\text{st}} = \arg\min_{w} \sum_{i=1}^{n} \left(y_i - \sum_{m=1}^{M} w_m \hat{f}_m^{-(i)}(x_i) \right)^2$$

- $\hat{f}_m^{-(i)}(x)$ is the **LOO prediction** at x, i.e., estimated leaving out i-th training example.
- The final prediction at point x is

$$\sum_{m} \hat{w}_m^{\text{st}} \hat{f}_m(x)$$

The weights may be restriced to be sum up to 1.

$$\hat{w}^{\text{st}} = \arg\min_{w} \sum_{i=1}^{n} \left(y_i - \sum_{m=1}^{M} w_m \hat{f}_m^{-(i)}(x_i) \right)^2$$

The objective function in the stacking weights problem is related to leave-one-out cross-validation error estimate

$$\text{LOOCV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{f}^{-(i)}(x_i) \right)^2$$

- If the weight vectors w are restricted so that have one unit weight and the rest zero, this leads to a model choice with smallest leave-one-out cross-validation error.
- Stacking combines them with estimated optimal weights.
  - better prediction, but less interpretability

Estimate weights (K-fold CV)

Suppose there are M methods to predict f. We choose $w = (w_1, \ldots, w_M) \in \mathcal{W}$ (a grid of weights) by minimizing the K-fold cross validation loss. Suppose the data is divided into K folds, say $C_1, \ldots, C_k$.

1. For $k = 1, \ldots, K$:
   (1) For $m = 1, \ldots, M$:
      (i) Estimate $\hat{f}_m^{(-k)}$ based on the $\{i \in C_k^c\}$
      (ii) Obtain the out-of-sample estimate $\hat{f}_m^{(-k)}(x_i)$ for all $i \in C_k$
   (2) For each $\tilde{w} \in \mathcal{W}$: Obtain the out-of-sample loss $MSE_k(\tilde{w}) = \sum_{i \in C_k} (Y_i - \sum_{m=1}^{M} \tilde{w} \hat{f}_m^{(-k)}(x_i))^2$
2. Obtain the overall out-of-sample loss $MSE(\tilde{w}) = \sum_{k=1}^{K} MSE_k(\tilde{w})$ for each $\tilde{w} \in \mathcal{W}$.
3. Choose $w^* = \arg\min_{\tilde{w} \in \mathcal{W}} MSE(\tilde{w})$

Bumping

Bumping

To maintain original interpretation of the model, "Bumping" uses bootstrap sampling to estimate potential models, aiming to stochastically identify the optimal *single* (fitted) model.

- Draw bootstrap samples $\mathbf{Z}^{*(1)}, \ldots, \mathbf{Z}^{*(B)}$, for $b = 1, \ldots, B$
  - By convention, the original training sample is included in the set of bootstrap samples.
- Fit the model to $\mathbf{Z}^{*(b)}$ giving $\hat{f}^{*(b)}(x)$.
- Choose the model

$$\hat{b} = \arg\min_{b} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{f}^{*(b)}\left(x_i\right) \right)^2$$

- The prediction is

$$\hat{f}^{*(\hat{b})}(x)$$

- Bumping introduces variations in the data to navigate the model fitting process.
  - if certain data points lead to suboptimal solutions, any bootstrap sample that exclude these point should result in improved outcomes.
- When employing bumping to compare models based on the training data, it's crucial to match the complexity across the models.
  - For each bootstrap sample, the resulting model may vary but should maintain comparable complexity levels.
  - Models from the same family may be used as well.
    - E.g., in the context of decision trees, this could involve adjusting trees to have the same number of terminal nodes for each bootstrap sample.
- It's a standard practice to include the original training sample within the pool of bootstrap samples during the bumping process.