Ensemble Methods

Thomas Bonald

March 2021

Supervised learning

Objective: Predict the **label** (classification) or the **value** (regression) of an object by training.

Formally, learn some mapping $f : x \mapsto y$ minimizing:

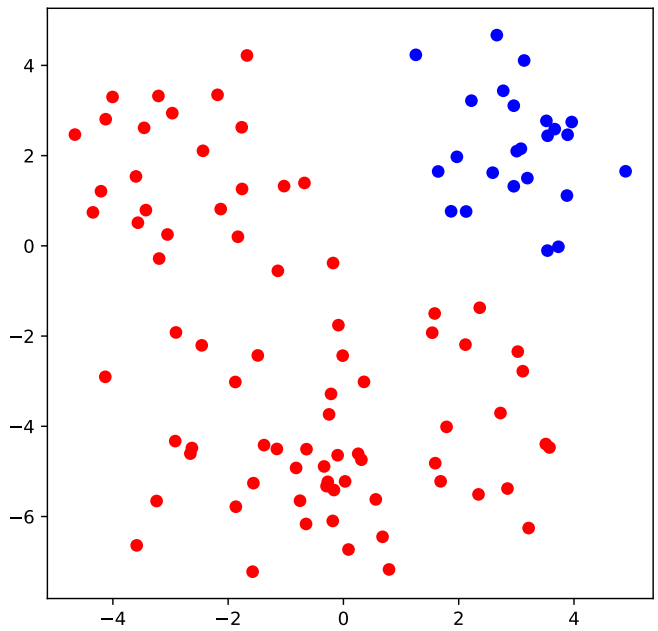
$$\frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) + \Omega(f)$$

where

- $x \in \mathbb{R}^d$
- $y \in \{0, 1\}, \{1, \ldots, K\}$ or $\mathbb{R}$
- $(x_1, y_1), \ldots, (x_n, y_n)$ are the training examples
- ℓ is the loss function
- Ω is the (optional) regularization function

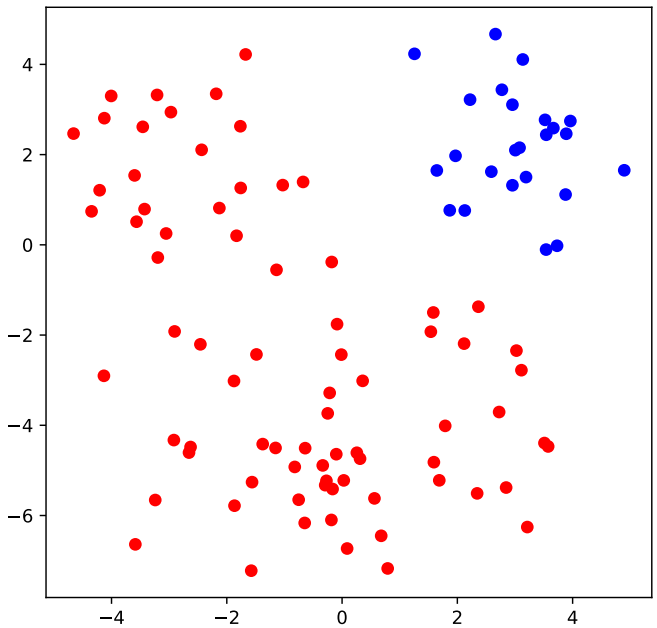
Example

$$x \in \mathbb{R}^2,\ y \in \{0, 1\},\ n = 100$$



Decision tree

Recursive split of the training set maximizing **loss reduction**.



Regression

$$\ell(\hat{y}, y) = (\hat{y} - y)^2 \quad y, \hat{y} \in \mathbb{R}$$

- At the root, the best value for $\hat{y}$ is the **mean** and the corresponding loss is the **variance**:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} y_i \quad V = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)^2$$

- At any leaf with samples $A \subset \{1, \ldots, n\}$ and variance V, find the split $A = A_{\leftarrow} \cup A_{\rightarrow}$ maximizing **variance reduction**:

$$\Delta = V - (\alpha_{\leftarrow} V_{\leftarrow} + \alpha_{\rightarrow} V_{\rightarrow}) \geq 0$$

with

$$\alpha_{\leftarrow} = \frac{|A_{\leftarrow}|}{|A|} \quad \alpha_{\rightarrow} = \frac{|A_{\rightarrow}|}{|A|}$$

Classification

$$\ell(\hat{y}, y) = 1_{\{\hat{y} \neq y\}} \quad y, \hat{y} \in \{1, \ldots, K\}$$

- At the root, the best value for $\hat{y}$ is the **mode** and the corresponding loss is the **proportion** of other labels:

$$k^\star = \arg\max_k \sum_{i=1}^{n} 1_{\{y_i = k\}} \quad P = 1 - \frac{1}{n} \sum_{i=1}^{n} 1_{\{y_i = k^\star\}}$$

- Many different ways of splitting each leaf node...

Diversity

- Gini index:

$$G = 1 - \sum_{k=1}^{K} p_k^2$$

  with p_k the proportion of label k

- At any leaf with samples $A \subset \{1, \ldots, n\}$ and Gini index G, find the split $A = A_{\leftarrow} \cup A_{\rightarrow}$ maximizing **diversity reduction**:

$$\Delta = G - (\alpha_{\leftarrow} G_{\leftarrow} + \alpha_{\rightarrow} G_{\rightarrow}) \geq 0$$

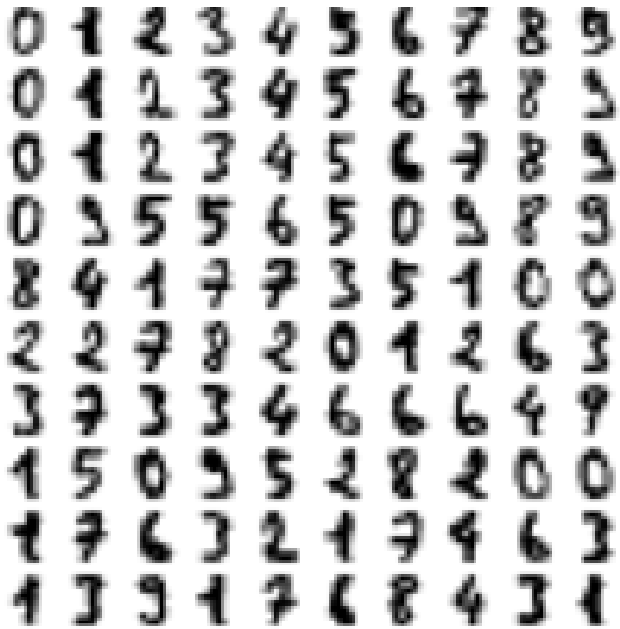
  with

$$\alpha_{\leftarrow} = \frac{|A_{\leftarrow}|}{|A|} \quad \alpha_{\rightarrow} = \frac{|A_{\rightarrow}|}{|A|}$$

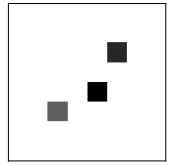
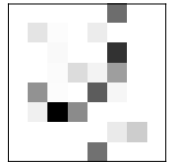
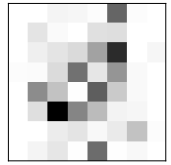
- For **binary classification**, $V = G$
  $\rightarrow$ equivalent to regression with $\hat{y}$ = probability of label 1

Example: Digits

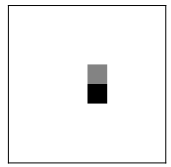
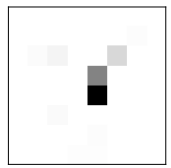
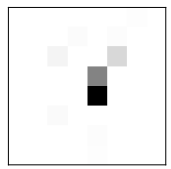
$$x \in \mathbb{R}^{8\times 8},\ y \in \{0, 1, \ldots, 9\}$$
$$n = 1438 \text{ (train)},\ n' = 359 \text{ (test)}$$



Multi-class decision tree

Depth	Feature importance	Accuracy (test)
2		0.32
5		0.68
10		0.84

Binary decision trees (one-vs-all)

Depth	Feature importance ($y = 0$)	Accuracy (test)
2		0.95
5		0.98
10		0.98

1. Bagging

Idea: Train multiple estimators using independent **bootstrap** samples and **aggregate** them.

Algorithm

Parameters: nb of estimators T, boostrap sample size m

Bootstrap

For $t = 1, \dots, T$,

- $B \leftarrow m$ samples drawn independently with replacement
- Train estimator f_t using B

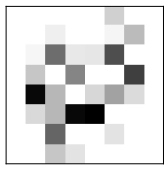
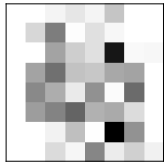
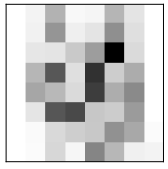
Aggregate

$f = \frac{1}{n} \sum_{t=1}^{T} f_t$ (regression) or $f = \arg\max_k \sum_{t=1}^{T} 1_{\{f_t = k\}}$ (classification)

Variants: Random Forest, ExtraTree

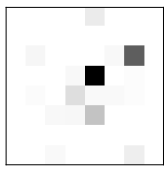
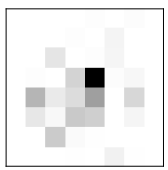
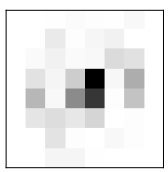
Multi-class random forest

Tree depth $= 5$

Nb. estimators	Feature importance	Accuracy (test)
1		0.62
5		0.84
20		0.94

Binary random forests (one-vs-all)

Tree depth $= 5$

Nb. estimators	Feature importance ($y = 0$)	Accuracy (test)
1		0.95
5		0.97
20		0.98

2. Gradient Boosting

Idea: Improve the estimator sequentially by predicting the **errors** (regression or binary classification).

Algorithm

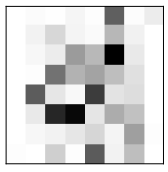
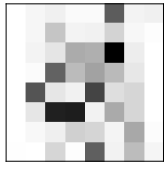
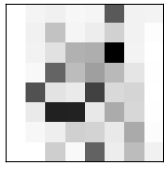
Parameters: nb of estimators T
For $t = 1, \dots, T$,

- Train h on samples $(x_1, y_1 - f(x_1)), \dots, (x_n, y_n - f(x_n))$
- $f \leftarrow f + h$

Variants: Add learning rate, weights (AdaBoost)

Gradient Boosting

Tree depth = 5

Nb. estimators	Feature importance	Accuracy (test)
1		0.86
5		0.92
20		0.96

XGBoost

Chen & Guestrin 2016

Learn sequentially a **regression tree** h minimizing:

$$\sum_{i=1}^{n} \ell(\hat{y}_i + h(x_i), y_i) + \Omega(h)$$

using **second-order** Taylor expansion:

$$\ell(\hat{y} + h, y) \approx \ell(\hat{y}, y) + h\frac{\partial \ell}{\partial \hat{y}}(\hat{y}, y) + \frac{h^2}{2}\frac{\partial^2 \ell}{\partial \hat{y}^2}(\hat{y}, y)$$

and **regularization** function:

$$\Omega(h) = \sum_{l=1}^{L}(\gamma + \frac{\lambda}{2}\alpha_l^2) \quad \text{for } h = \sum_{l=1}^{L} \alpha_l 1_{A_l}$$

XGBoost

This is equivalent to minimizing:

$$\sum_{l=1}^{L}(G_l\alpha_l + (\lambda + H_l)\frac{\alpha_l^2}{2} + \gamma)$$

with

$$G_l = \sum_{i \in A_l} \frac{\partial \ell}{\partial \hat{y}}(\hat{y}_i, y_i) \quad H_l = \sum_{i \in A_l} \frac{\partial^2 \ell}{\partial \hat{y}^2}(\hat{y}_i, y_i)$$

We get:

$$\alpha_l^\star = -\frac{G_l}{\lambda + H_l}$$

so that we seek the partition $A_1, \ldots, A_L$ minimizing:

$$\sum_{l=1}^{L}\left(\gamma - \frac{1}{2}\frac{G_l^2}{\lambda + H_l}\right)$$

Loss functions

- Regression

$$\ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$$

$$\frac{\partial \ell}{\partial \hat{y}}(\hat{y}, y) = \hat{y} - y, \quad \frac{\partial^2 \ell}{\partial \hat{y}^2} = 1$$

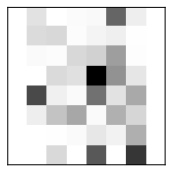
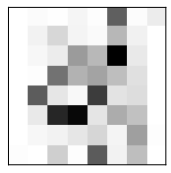
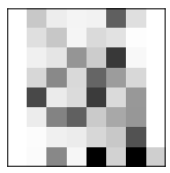
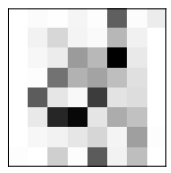
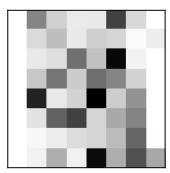
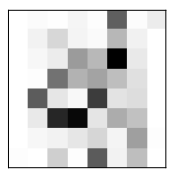
- Binary classification

$$\ell(\hat{y}, y) = -y \log \hat{y} - (1 - y) \log(1 - \hat{y}) \quad \text{with} \quad \hat{y} = \frac{e^{\hat{z}}}{1 + e^{\hat{z}}}$$

$$\frac{\partial \ell}{\partial \hat{z}}(\hat{y}, y) = -|\hat{y} - y|, \quad \frac{\partial^2 \ell}{\partial \hat{z}^2} = \hat{y}(1 - \hat{y})$$

XGBoost

Tree depth = 5

Nb. estimators	Feature imp. (XGB / GB)		Accuracy (test)
1			0.91
5			0.94
20			0.96

Summary

Two categories of ensemble methods:

1. **Bagging**
   parallel training / aggregation
   e.g., Random Forest
2. **Gradient boosting**
   sequential training / correction
   e.g., AdaBoost, XGBoost, LightGBM, CatBoost