Decision Trees and CART (CS3780/5780, Sp26)

1 Motivation: decision trees are high-variance learners

Decision trees illustrate the bias–variance tradeoff very clearly:

- If the tree is too shallow, it underfits.
- If the tree is grown to maximum depth, it can drive the training error to zero whenever the examples have distinct feature vectors, but it often overfits badly.
- Tree depth is not always a fine-grained enough knob to get the best tradeoff.

This motivates *ensemble methods*, which combine multiple predictors trained on different views of the training data (say carefuly chosen sub-samples).

1. **Bagging** mainly targets variance.
2. **Boosting** mainly targets bias.

A full-depth tree is a canonical example of a learner with low bias and high variance, so it is a natural base learner for bagging.

2 Bias–variance decomposition

For squared loss at a fixed input x, we can decompose the prediction error as

$$\mathbb{E}\big[(h_D(x) - y)^2\big] = \underbrace{\mathbb{E}\big[(h_D(x) - \bar{h}(x))^2\big]}_{\text{variance}} + \underbrace{\mathbb{E}\big[(\bar{h}(x) - \bar{y}(x))^2\big]}_{\text{bias}^2} + \underbrace{\mathbb{E}\big[(\bar{y}(x) - y)^2\big]}_{\text{inherent noise}}. \tag{1}$$

Here:

- h_D is the predictor trained on dataset D,
- $\bar{h}(x) = \mathbb{E}_D[h_D(x)]$ is the mean predictor,
- $\bar{y}(x) = \mathbb{E}[y \mid x]$ is the regression function.

The goal of bagging is to reduce the *variance term*.

3 A thought experiment: averaging predictors from many datasets

Suppose we could draw m independent datasets of size n,

$$D_1, D_2, \ldots, D_m \overset{\text{iid}}{\sim} P^n,$$

train the algorithm on each dataset, and average the resulting predictors:

$$\hat{h}(x) = \frac{1}{m}\sum_{i=1}^{m} h_{D_i}(x). \tag{2}$$

This suggests a variance reduction mechanism. As a warm-up, recall the scalar setting: if $X_1, \ldots, X_m$ are iid with variance σ^2, then

$$\mathrm{Var}\left(\frac{1}{m}\sum_{i=1}^{m} X_i\right) = \frac{\sigma^2}{m}. \tag{3}$$

So averaging independent noisy quantities reduces variance by a factor of m.

If the predictors $h_{D_i}(x)$ behaved like independent random quantities centered around the mean predictor, then averaging would push us toward a lower-variance predictor. This is the core intuition behind bagging.

Remark: The clean $1/m$ calculation depends on independence. If the terms are correlated, the variance reduction is weaker. This is one reason predictor *diversity* matters in ensembles.

4 The practical obstacle: we only have one dataset

The thought experiment above assumes access to many independent datasets from P^n. In practice we only observe one training set D.

Bagging overcomes this using the **bootstrap**:

1. Start from the original training set $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.
2. Form a new dataset D_j of size n by repeatedly sampling elements of D *with replacement*, uniformly at random.
3. Train a base learner h_{D_j} on each bootstrap dataset.
4. Average the resulting predictors.

This gives the bagged predictor

$$\widehat{h}_{\mathrm{Bag}}(x) = \frac{1}{m}\sum_{j=1}^{m} h_{D_j}(x). \tag{4}$$

For classification, this averaging becomes voting (or averaging scores followed by thresholding). For regression, it is literal averaging.

5 Bootstrap view via the empirical distribution

Given the dataset D, define the empirical distribution Q_D by

$$Q_D\big((x_i, y_i) \mid D\big) = \frac{1}{n}, \qquad i = 1, \ldots, n. \tag{5}$$

A bootstrap bag D_j is simply an iid sample of size n from Q_D.

So bagging can be written as

$$\widehat{h}_{\mathrm{Bag}}(x) = \frac{1}{m}\sum_{j=1}^{m} h_{D_j}(x), \qquad D_j \sim Q_D^n. \tag{6}$$

The bags are not independent draws from the original population distribution P, but they are easy to generate and often create enough instability in the base learner to reduce variance after averaging.

6 Why does bagging still make sense?

A useful fact from the handwritten notes is that bootstrap resampling preserves the *marginal* distribution of a sample, even though the resulting bags are not independent in the same way as fresh draws from P.

Assume for simplicity that the feature space is finite:

$$\Omega = \{x_1, \ldots, x_N\}, \qquad P(X = x_i) = p_i.$$

Let Q denote the distribution of a single bootstrap sample generated by the two-stage procedure:

1. draw a dataset D of size n iid from P;
2. draw one example uniformly from D.

Claim: For every datapoint (x_i, y_i), the marginal distribution of Q matches P:

$$Q((X, Y) = (x_i, y_i)) = P((X, Y) = (x_i, y_i)) \tag{7}$$

Proof sketch. Say $p_i = P((X, Y) = (x_i, y_i))$. Now note that

$$\begin{aligned}
Q(X = x_i) &= \sum_{k=1}^{n} \text{Prob}\left(\text{there are k copies of } (x_i, y_i) \text{ in } D\right) \times \text{Prob we pick one of the k copies} && (8)\\
&= \sum_{k=1}^{n} \binom{n}{k} p_i^k (1 - p_i)^{n-k} \frac{k}{n} && (9)\\
&= \frac{1}{n} \left(\sum_{k=1}^{n} \binom{n}{k} p_i^k (1 - p_i)^{n-k} k \right) && (10)\\
&= \frac{1}{n} \mathbb{E}[\text{Binomial}(p_i, n)] && (11)\\
&= \frac{1}{n} n p_i = p_i && (12)
\end{aligned}$$

So each bootstrap draw has the correct marginal distribution, even though the bags are not fresh independent datasets from P^n.

7 Advantages of bagging

7.1 Variance reduction

Bagging is particularly effective when the base learner has high variance. Full-depth decision trees are a textbook example: two nearby bootstrap samples can produce very different trees, but averaging many such trees stabilizes the final prediction.

7.2 Simple implementation

Bagging is conceptually and computationally simple:

- the bags can be generated with a few lines of code,
- the component models can be trained independently and in parallel,
- the ensemble prediction is just an average or a vote.

7.3 Uncertainty from variability across bags

Because the final prediction is an average over many predictors, the spread of the individual predictions can be interpreted as a measure of predictive uncertainty. This is especially useful in regression settings.

8 Out-of-bag (OOB) error

A striking feature of bagging is that it gives a nearly free test-error estimate.

For each training example $(x_i, y_i) \in D$, define the set of bags that *did not* contain that example:

$$S_i = \{k \in \{1, \dots, m\} : (x_i, y_i) \notin D_k\}. \tag{13}$$

Now average only the models that never saw (x_i, y_i):

$$\tilde{h}_i(x) = \frac{1}{|S_i|} \sum_{k \in S_i} h_k(x). \tag{14}$$

The **out-of-bag error** is then

$$\epsilon_{\text{OOB}} = \frac{1}{n} \sum_{(x_i, y_i) \in D} \ell\big(\tilde{h}_i(x_i), y_i\big). \tag{15}$$

This acts like a built-in validation estimate because, for each point, we evaluate it only using models that were trained without that point.

9 Random forests

A **random forest** is bagging applied to full-depth decision trees, with one extra randomization step.

9.1 Algorithm

Let d denote the number of features. To train a random forest:

1. Sample bootstrap datasets $D_1, \dots, D_m$ from D.
2. For each bag D_j, train a full decision tree h_j.
3. Before each split in each tree, randomly choose only $k \le d$ features (without replacement), and search for the best split using only those k features.
4. Return the ensemble
$$h(x) = \frac{1}{m} \sum_{j=1}^{m} h_j(x) \tag{16}$$
in regression, or the corresponding vote / averaged score in classification.

9.2 Why feature subsampling helps

Bagging works best when the component predictors are not too correlated. If there is one extremely strong feature, then plain bagging of trees may repeatedly split on that same feature near the root, causing many trees to look too similar.

Randomly limiting the candidate features at each split forces different trees to explore different structures, thereby reducing correlation among the trees and improving the gain from averaging.

9.3 Practical defaults

Two standard hyperparameters are:

- m: the number of trees,
- k: the number of features considered at each split.

A very common default for classification is

$$k = \sqrt{d}. \tag{17}$$

In practice, performance is often fairly insensitive to the exact choice of m and k, one reason random forests are such strong out-of-the-box methods.

Random forests inherit several advantages from decision trees:

- They need relatively little preprocessing.
- Features can live on different scales and in different units.
- They handle heterogeneous data naturally.

For example, a medical dataset may include blood pressure, age, glucose level, and categorical variables such as gender or smoking status, all on different scales. Trees and forests can often use these variables directly.