Ensemble Learning

–

Bagging, Boosting and Random Forests

Agenda

- Ensemble learning approach - Committee-based methods
- Bagging - increasing stability
- Boosting - "Best off-the-shelf classification technique"
- "Fate is a great provider!" - Random Forests

Committee-based methods - Consensus approach

- Instead of a single classifier, combine the predictions of an **ensemble** of classifiers
$$C_1(X),\ \ldots,\ C_M(X).$$
Amit and Geman (1997)

- Majority vote:
$$sign\left(\sum_{m=1}^{M} C_m(X)\right)$$

- Variant - weighted majority vote: $\alpha_i \geq 0$, $\sum_i \alpha_i = 1$
$$sign\left(\sum_{m=1}^{M} \alpha_m C_m(X)\right)$$

- Can be easily extended to multi-class setup, regression

- A challenge of today: extend the consensus approach to "ranking"

Bagging

- **B**ootstrap **agg**regat**ing** technique - Breiman (1996)
- Can be applied to any learning algorithm $\mathcal{L}$
- Based on training data $\mathcal{D}_n$:
  1. Generate independently $B \geq 1$ bootstrap datasets $\mathcal{D}_n^{*(b)}$ (by drawing with replacement in $\mathcal{D}_n$)
  2. For $b : 1$ to B, run algorithm $\mathcal{L}$ based on $\mathcal{D}_n^{*(b)}$, yielding rule $C^{*(b)}$
  3. Aggregate the bootstrap predictors by taking the majority vote:

$$C_{bag}(X) = sign\left(\sum_{b=1}^{B} C^{*(b)}(X)\right)$$

- Variant: if $C^{*(b)}(X) = sign(f^{*(b)}(X))$,

$$\widetilde{C}_{bag} = sign\left(\sum_{b=1}^{B} f^{*(b)}(X)\right)$$

Bagging - Some stylized facts

- Bagging can **dramatically reduce the variance** of unstable procedures (ex: decision trees)
- Variance reduction may lead to a smaller test error
- Regression setup: $f_{bag}(x) = \mathbb{E}[f^*(x)]$ (expectation taken over the training data)

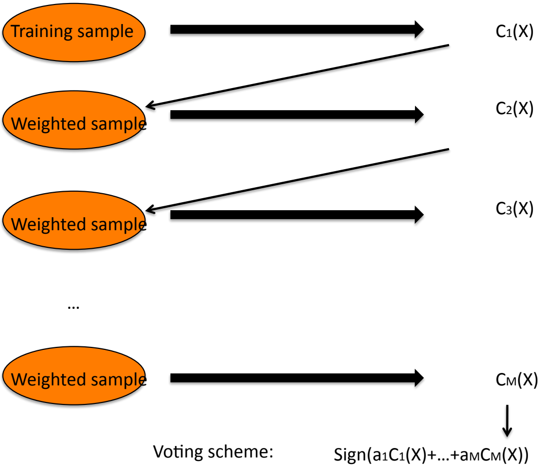
$$\mathbb{E}\left[(Y - f^*(x))^2\right] = \mathbb{E}\left[(Y - f_{bag}(x))^2\right] + \mathbb{E}\left[(f_{bag}(x) - f^*(x))^2\right] \geq \mathbb{E}\left[(Y - f_{bag}(x))^2\right]$$

- In classification:
  bagging a good classifier makes it better, and ...
  bagging a bad classifier can make it worse!

Boosting classifiers

- AdaBoost - Freund Schapire (1995)
- Ingredients for "slow learning": a weak classification method $\mathcal{L}$
- Heuristics:
  - apply $\mathcal{L}$ to weighted versions of the original sample
  - increase the weights of the data which are currently misclassified
  - aggregate the classifiers in a nonuniform fashion
    (a good predictor should not work for a few outliers)
- Outperforms its competitors for most benchmark datasets
- Statistical explanation: five years later...

Boosting - General scheme

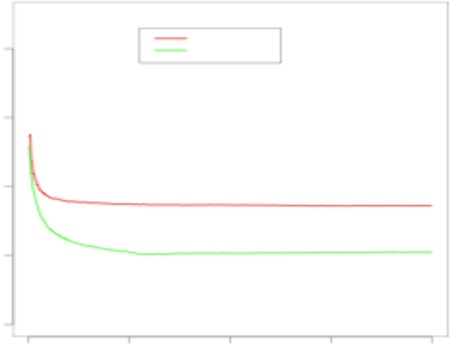


The "Adaptive Boosting" algorithm

- Initialization: uniform weights, $\omega_i = 1/n$ assigned to labeled example (X_i, Y_i), $1 \leq i \leq n$
- For m : 1 to M,
  1. Using algorithm $\mathcal{L}$, fit the weak classifier C_m based on the weighted data sample $\{(X_i, Y_i, \omega_i) : 1 \leq i \leq n\}$
  2. Compute the weighted classification error
  $$err_m = \sum_{i=1}^{n} \omega_i \mathbb{I}\{Y_i \neq C_m(X_i)\}$$
  and $a_m = \log((1 - err_m)/err_m)$
  3. Update the weights:
     - ★ $\omega_i \leftarrow \omega_i \exp\left(a_m \mathbb{I}\{Y_i \neq C(X_i)\}\right)$
     - ★ $\omega_i \leftarrow \omega_i / \sum_{j=1}^{n} \omega_j$
- Output: $C_{Boost}(X) = sign\left(\sum_{m=1}^{M} a_m C_m(X)\right)$

AdaBoost resists overfitting

- Typical weak learner: stumps (trees of depth 1)
- As M increases, the test error decreases and stabilizes



Practical issues

- How to run $\mathcal{L}$ based on a **weighted** sample?
  - modify the criterion analytically (ex: CART, SVM, k-NN, *etc.*)
  - draw a sample using the distribution $\sum_i \omega_i \delta_{(X_i, Y_i)}$
- When to stop?
  - plot the test error vs M
  - stop when the test error stabilizes

A statistical view of Boosting

- Friedman, Hastie & Tibshirani (2000)
- Stagewise forward additive modelling
- Exponential loss: $C(X) = sign(f(X))$

$$L_e(f) = \mathbb{E}[\exp(-Yf(X))]$$

- Optimal solution:

$$f^*(X) = \frac{1}{2}\log\left(\frac{\eta(X)}{1-\eta(X)}\right)$$

Forward stagewise additive modelling

- Heuristics: refine a current predictive rule $f_{m-1}(x)$ by adding $\alpha_m C_m(x)$, with $\alpha_m \in \mathbb{R}$ and $C_m(x) \in \{-1, \ +1\}$
- How to choose α_m and $C_m(x)$ so as to minimize the empirical exponential risk?

$$\arg\min_{\alpha,\, C} \sum_{i=1}^{n} \exp\left(-Y_i(f_{m-1}(X_i) + \alpha C(X_i))\right) = ?$$

- Set $\omega_i = \exp(-Y_i f_{m-1}(X_i))$, the empirical risk can be written as:

$$\sum_{i=1}^{n} \omega_i \exp\left(-Y_i \alpha C(X_i)\right)$$

- Whatever $\alpha > 0$, the classifier that achieves the minimum risk is that which minimizes the weighted risk:

$$\sum_{i=1}^{n} \omega_i \mathbb{I}\{Y_i \neq C(X_i)\}$$

Forward stagewise additive modelling

- Let $C_m(X)$ be the rule, solution of the weighted classification problem, set:

$$err_m = \sum_{i=1}^{n} \omega_i \mathbb{I}\{Y_i \neq C_m(X_i)\}$$

- Now, minimize in α:

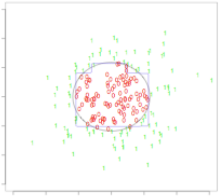
$$e^{\alpha} err_m + e^{-\alpha}(1 - err_m),$$

yielding $\alpha_m = (1/2) \cdot \log((1 - err_m)/err_m)$

- Many variants: other losses, weight trimming, etc.

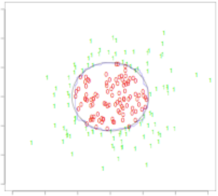
Model averaging produces smoother boundaries

Decision Boundary: Tree



When the nested spheres are in R^{10}, CART™ produces a rather noisy and inaccurate rule $\hat{C}(X)$, with error rates around 40%.

Decision Boundary: Boosting



Bagging and Boosting average many trees, and produce smoother decision boundaries.

Random Forests

- Ingredients: bagging + randomization
- Randomize over the set of predictive variables (X's components):
  - before growing a bootstrap decision tree
  - when splitting an interior node of the classification tree
- No pruning, small trees
- Aggregation preserves consistency...
  but no theoretical explanation for the performance observed!
- Heuristics: randomization "enriches" the rule
- Randomize over the training data (when massive)