Machine Learning

Lecture 4 - Ensemble methods

Mattias Villani

Department of Statistics
Stockholm University

Department of Computer and Information Science
Linköping University

mattiasvillani.com @matvil mattiasvillani

Lecture overview

- Ensembles
- Bagging
- Random forest
- Boosting
- XGboost

Tree ensemble

- Regression trees suffer from large variance.
- **Tree ensembles** combine many trees additively

$$\hat{f}(\mathrm{x}) = \sum_{k=1}^{K} \hat{f}_k(\mathrm{x}),\ \hat{f}_k \in \mathcal{F}$$

  where $\mathcal{F}$ is the collection of all trees.
- **Bagging**: learn trees $\hat{f}_k(\mathrm{x})$ on separate bootstrap samples.
- **Boosting**: learn trees $\hat{f}_k(\mathrm{x})$ sequentially by fitting to amplified residuals.
- Ensemble members need not be trees, any model works.

Bagging

- Fit a **low bias/high variance base model** to B boostrap replicate datasets.
- Average the predictions over all bootstrap samples. **Bootstrap aggregation**.
- **Regression**

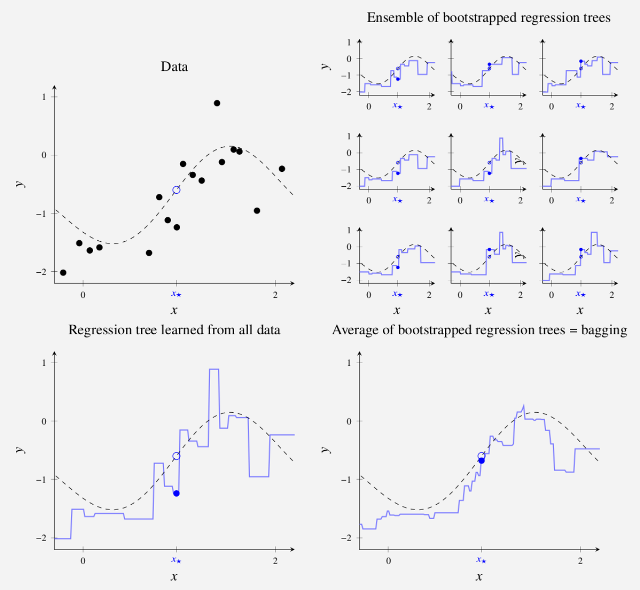
$$\hat{y}_{\text{bag}}(\mathrm{x}_\star) = \frac{1}{B}\sum_{b=1}^{B}\tilde{y}^{(b)}(\mathrm{x}_\star)$$

- **Classification**

$$\mathrm{g}_{\text{bag}}(\mathrm{x}_\star) = \frac{1}{B}\sum_{b=1}^{B}\tilde{\mathrm{g}}^{(b)}(\mathrm{x}_\star),$$

  $\tilde{\mathrm{g}}^{(b)}(\mathrm{x}_\star)$ is a vector of class probabilities in bootstrap sample b.
- When classifier only returns predictions: majority vote.

Bagging trees



Bagging reduces variance

- Assume $\mathbb{E}_{\text{boot}}\left(\tilde{y}^{(b)}(\mathbf{x}_\star)\right) = \bar{f}(\mathbf{x}_\star)$ and $\mathbb{V}_{\text{boot}}\left(\tilde{y}^{(b)}(\mathbf{x}_\star)\right) = \sigma^2(\mathbf{x}_\star)$ for all $b = 1, \ldots, B$ (approx true).
- Then

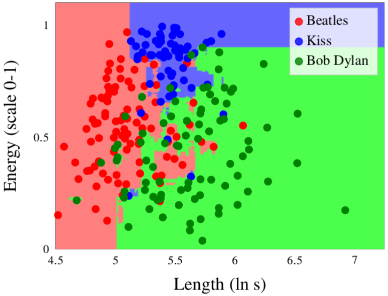
$$\mathbb{E}_{\text{boot}}\left(\hat{y}_{\text{bag}}(\mathbf{x}_\star)\right) = \bar{f}(\mathbf{x}_\star)$$
$$\mathbb{V}_{\text{boot}}\left(\hat{y}_{\text{bag}}(\mathbf{x}_\star)\right) = \frac{1-\rho}{B}\sigma^2(\mathbf{x}_\star) + \rho\sigma^2(\mathbf{x}_\star),$$

  where ρ is the average correlation of $\hat{y}_{\text{bag}}(\mathbf{x}_\star)$ over the bootstrap replicates.
- **Bias** remains approx unchanged by bootstrap aggregation.
- **Variance** of the prediction reduced by bootstrap aggregation.
- The base model is fitted in isolatation on each bootstrap sample, so no risk of overfitting solely from using a large B.
- **Out-of-bag estimation** of E_{new} [Section 7.1 in MLES book].

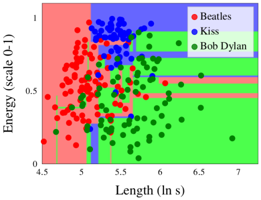
Random forest

- **Random forest** is a tree ensemble with trees grown by **bagging**.
- **Bagging observations**: tree grown on subsets of observations sampled with replacement.
- **Bagging features**: random choice of allowed splitting variables at each tree node.
- Bagging features **de-correlates the prediction** for different bootstrap samples. ☺
- Bagging features inflates the variance of the prediction for individual trees. ☹

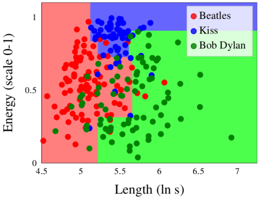
Random forest for song classification



Fully grown tree



Tree with max depth 4



Boosted tree ensembles

- **Boosting**: iterative fitting. **Poorly predicted observations** at previous iteration are **upweighted** (**boosted**).
- Boosting aims at **reducing bias from weak learners** (shallow trees).
- **Boosted tree ensembles**: add tree that fits boosted errors.
- Boosting $\approx$ Greedy forward selection (with special loss).
- Bagging learns independently. Boosting learns sequentially.

Algorithm 2: Greedy forward algorithm for tree ensembles.

Input: Data $\{y_i, \mathbf{x}_i\}_{i=1}^n$, tree generator $f(\mathbf{x}; \gamma)$ parametrized by split variables, split points and leave values.

set $\phi_0(\mathbf{x}) = 0$
for $m = 1$ **to** M **do**
  Compute $\gamma_m = \arg\min_\gamma \sum_{i=1}^n \ell(y_i, \phi_{m-1}(\mathbf{x}_i; \gamma) + f(\mathbf{x}_i; \gamma))$
  Set $\phi_m(\mathbf{x}; \gamma) = \phi_{m-1}(\mathbf{x}; \gamma) + f(\mathbf{x}; \gamma_m)$
end

Output: Ensemble $\phi_M(\mathbf{x}; \gamma)$ and tree parameters $\gamma_1, \ldots, \gamma_M$.

XGBoost - Extreme Gradient Boosting

- Boosted tree ensemble with smooth penalty $\eta |T| + \lambda \|\mathrm{w}\|_2^2$.
- **Gradient boosting**: approximate objective at iteration t

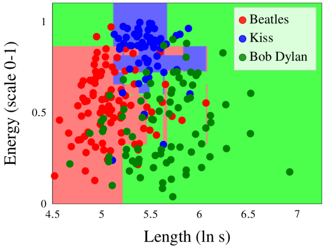
$$\sum_{i=1}^{n} L\left(y_i, \hat{y}_i^{(t-1)} + f_t(\mathrm{x}_i)\right) \approx \sum_{i=1}^{n} L\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(\mathrm{x}_i) + \frac{1}{2} h_i f_t^2(\mathrm{x}_i)$$

  - $\hat{y}_i^{(t-1)}$ the fit from ensemble at previous iteration
  - $g_i = \frac{\partial L(y_i,\hat{y})}{\partial \hat{y}}\Big|_{\hat{y}=\hat{y}_i^{(t-1)}}$ and $h_i = \frac{\partial^2 L(y_i,\hat{y})}{\partial^2 \hat{y}}\Big|_{\hat{y}=\hat{y}_i^{(t-1)}}$.
- Tree structure: $q(\mathrm{x}) : \mathbb{R}^p \to T$, splits and splitting points.
- Note that $f_t(\mathrm{x}_i) = w_\ell$ for all $\mathrm{x}_i \in R_\ell$.
- Given a tree structure $q(\mathrm{x})$ solve for w_ℓ to get the objective

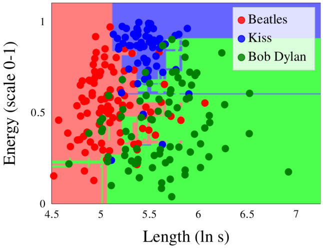
$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2} \sum_{\ell=1}^{|T|} \frac{\left(\sum_{i \in I_\ell} g_i\right)^2}{\sum_{i \in I_\ell} h_i + \lambda} + \eta |T|, \text{ where } I_\ell = \{i | q(\mathrm{x}_i) = \ell\}$$

- $\tilde{\mathcal{L}}^{(t)}(q)$ can be optimized w.r.t. tree structure $q_t(\mathrm{x})$ in a greedy fashion, starting with a single leave and adding splits.

Boosting for song classification



(a) AdaBoost



(b) A gradient boosting classifier