Introduction to Machine Learning

Chapter 15: Bagging and Random Forests

Bernd Bischl, Christoph Molnar

Department of Statistics – LMU Munich

Winter term 2017/18

ENSEMBLE METHODS

- Ensemble methods combine the predictions of several base learners and combine them into an aggregated estimator.
- Homogeneous ensembles (multiple models of same base learner)
  - Bagging: Fit models on bootstrapped versions of training data
  - Boosting: runs sequentially: each model on reweighted data version / the previous residuals so it improves the errors of the previous round
- Heterogeneous ensembles (different base learners)
  - Fit different base learners on the same data or different “views” of the same data. Then learn how to aggregate their predictions, often with a 2nd-layer model.

ENSEMBLE METHODS

General homogenous approach (often it works like this but not always)

- A "base learner" is selected and fitted multiple times to either resampled or reweighted versions of the original data.
- The base learner is applied to either resampled or reweighted versions of the original dataset. This results in M prediction functions $b^{[1]}(x), \dots, b^{[M]}(x)$.
- These M function are aggregated, usually in a linear fashion. This results in the following final prediction function:

$$f(x) = \sum_{m=1}^{M} \beta^{[m]} b^{[m]}(x)$$

with coefficients $\beta^{[1]}, \dots, \beta^{[M]}$.

BAGGING

- Bagging is based on **B**ootstrap **Agg**regation.
- Proposed by Breiman (1996).
- Train on multiple bootstrap samples of data $\mathcal{D}$, then combine:
  1. Create *M* bootstrap samples of size *n*.
  2. Fit the base learner on each of the *M* bootstrap samples.
  3. Aggregate the predictions of the *M* estimators via averaging or majority voting.
- *M* affects Monte-Carlo approximation error; main hyperparameter.
- Interpretability of the model becomes harder.

BAGGING

Bagging algorithm

Input: Dataset $\mathcal{D}$, base learner, number of bootstraps M

for $m = 1 \to M$ **do**

  Draw a bootstrap sample $\mathcal{D}^{[m]}$ from $\mathcal{D}$.

  Train base learner on $\mathcal{D}^{[m]}$, obtain model $b^{[m]}(x)$

end for

Aggregate the predictions of the M estimators (via averaging or majority voting), to determine the bagging estimator:

$$f(x) = \frac{1}{M} \sum_{m=1}^{M} b^{[m]}(x)$$

BAGGING

- Bagging reduces the variance of the estimator, but increases the bias in return.
- Bagging works best for unstable/high variance learners (learners where small perturbations in training set lead to larger changes in the prediction)
  - Classification and regression trees
  - Neural networks
  - Piece-wise variable selection in the regression case, etc.
- For stable estimation methods bagging might degrade performance
  - k-nearest neighbor
  - discriminant analysis
  - naive bayes
  - linear regression

WHY DOES BAGGING WORK?

- Suppose we have a numerical dependent variable.
- The training datasets are given by $\mathcal{D}$ and base learner estimator by $f(x)$.
- The datasets are sampled independently from distribution $\mathbb{P}_{xy}$ (data generating process).
- The *theoretical* aggregated estimator is given by

$$f_A(x) = \mathbb{E}_{\mathcal{D}}[f(x)].$$

WHY DOES BAGGING WORK?

- Let x,y be a random sample from $\mathbb{P}_{xy}$ but independent of $\mathcal{D}$. The average error of the normal $f(x)$ is then $e = E_{\mathcal{D}} E_{xy}[(y - f(x))^2]$ and of the aggregated estimator $e_A = E_{xy}[(y - f_A(x))^2]$.
- It follows:

$$e = E_{\mathcal{D}} E_{xy}[(y - f(x))^2] = E_{xy}[y^2] - 2E_{xy}[yf_A] + E_{\mathcal{D}} E_{xy}[f^2(x)]$$

- And we apply Jensen's inequality to e:

$$e = E_{xy} E_{\mathcal{D}}[(y - f(x))^2] \geq E_{xy}(E_{\mathcal{D}}[y - f(x)])^2 = E_{xy}[(y - f_A(x)])^2] = e_A$$

$$= E_{xy}[y^2] - 2E_{xy}[yf_A] + E_{xy}[f_A^2(x)]$$

WHY DOES BAGGING WORK?

- The difference between e and e_A is $E_{\mathcal{D}}E_{xy}[f^2(x)] \geq \mathbb{E}_{xy}[f_A^2(x)]$
- The more unstable $f(x)$, the more error reduction we obtain.
- But the bagging estimator only approximates the theoretical f_A (bootstrap), we therefore suffer from approximation error (bias) by using the empirical distribution function instead of the true data generating process and only perform M bootstrap iterations instead of an infinite number.
- Bagging does not necessarily lead to an improved classifier.
  - Example: binary outcome, $y = 1$ for all values of x
  - Consider random classifier f with $\mathrm{P}(f(x) = 1) = 0.4$ (independent of x)
  - Prediction error for f is 0.6
  - Prediction error for the bagging estimator is 1

RANDOM FORESTS

- Modification of bagging for trees.
- Proposed by Breiman (2001).
- Construction of bootstrapped **decorrelated** trees
- The variance of the bagging prediction depends on the correlation between the trees ρ

$$\rho\sigma^2 + \frac{1-\rho}{M}\sigma^2,$$

  where σ^2 describes the variance of a tree.

$\Rightarrow$ Reduce correlation by randomization in each split. Instead of all p features, draw $mtry \leq p$ random split candidates.

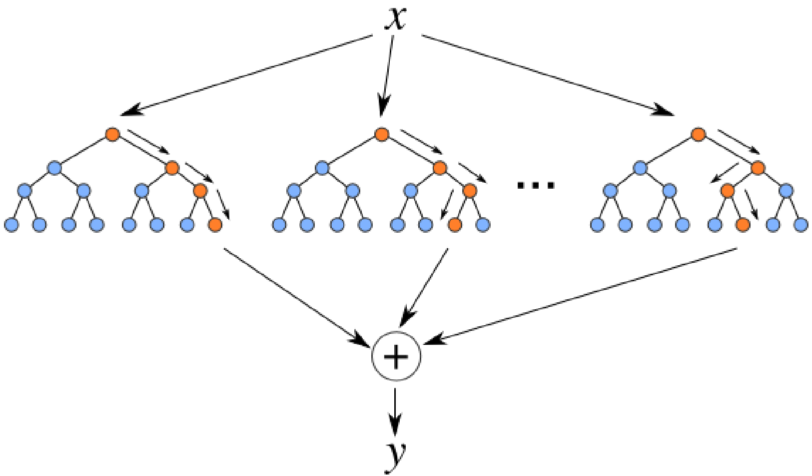
$\Rightarrow$ Trees are expanded as much as possible, without aggressive early stopping or pruning, to increase variance.

RANDOM FORESTS

Random Forest algorithm

```
Input: A dataset $\mathcal{D}$ of $n$ observations, number $M$ of trees in the forest, number mtry of variables to draw for each split
for $m = 1 \to M$ do
    Draw a bootstrap sample $\mathcal{D}^{[m]}$ from $\mathcal{D}$
    Grow tree $b^{[m]}(x)$ using $\mathcal{D}^{[m]}$
    For each split only consider mtry randomly selected features
    Grow tree without early stopping or pruning
end for
Aggregate the predictions of the $M$ estimators (via averaging or majority voting), to predict on new data.
```

RANDOM FORESTS



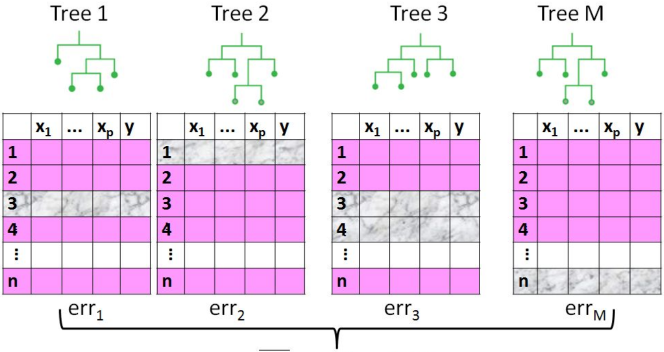
RANDOM FORESTS

- The following values are recommended for *mtry*:
  - Classification: $\lfloor \sqrt{p} \rfloor$
  - Regression: $\lfloor p/3 \rfloor$
- Out-Of-Bag error: On average ca. 1/3 of points are not drawn.

$$\mathbb{P}(\text{Obs. not drawn}) = \left(1 - \frac{1}{n}\right)^n \overset{n \to \infty}{\longrightarrow} \frac{1}{e} \approx 0.37.$$

  To compute the OOB error, each observation x is predicted only with those trees that did not use x in their fit.
- The OOB error is similar to cross-validation estimation. It can also be used for a quicker model selection.

RANDOM FORESTS



$$\overline{err} = oob - error$$

$$err_m = \sum_{x^{(i)} \in OOB_m} L(y^{(i)}, \hat{y}_m^{(i)})$$

in-bag observations, used to build the trees (Remember: The same observation can enter the in-bag sample more than once.)

out-of-bag observations (OOB_m), used to evaluate prediction performance (err_m)