Unit 6

Ensemble learning: Bagging and random forests

Ensemble learning

- Builds a prediction model by combining the strengths of a collection of simpler base models.
- Ensemble learning can be broken down into 2 tasks:
  1. Developing a population of base learners from the training data.
  2. Combining them to form the composite predictor.
- The ensemble estimate is given by

$$f(x) = \sum_{m=1}^{M} \alpha^{[m]} g^{[m]}(x),$$

where $g^{[m]}(x)$, $m = 1, 2, \ldots, M$ correspond to the fitted models of the base learners.

Ensemble learning / 2

- A number of different statistical methods are ensemble methods, e.g.:
  - Bagging
  - Random forests
  - Boosting
- In general as base learners *weak learners* are used:
  - In a classification setting a weak learner is defined to be a classifier that is better than random guessing, but only slightly correlated with the response.
- By combining weak learners a *strong learner* is created:
  - By combining the base learners the model space is enlarged.
  - In a classification setting a strong learner is defined as being arbitrarily well-correlated with the true classification.

Bagging

- Bagging is based on *bootstrap aggregation*.
- The base learners fitted to different bootstrap samples of the training data are combined.
- Easy to implement:
  - Repeated application of the base method.
  - Aggregation step.
- Number of bootstrap samples is selected to control the Monte-Carlo approximation error.
- Bagging performs well for non-stable base methods:
  - Classification and regression trees.
  - Neural networks.
  - Stepwise variable selection in regression.
  - . . .

Bagging: Algorithm

1. The training data is given by $\boldsymbol{Z} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$.
2. Generate B bootstrap samples $\boldsymbol{Z}^{*b}$, $b = 1, \ldots, B$, of size N by drawing with replacement from the original dataset of size N.
3. Fit the base method to each of the B bootstrap samples leading to estimates $\hat{f}^{*b}(x)$, $b = 1, \ldots, B$.
4. Aggregate the prediction of the B fitted models by:
   - Regression:

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x).$$

   - Classification:

$$\hat{G}_{\text{bag}}(x) = \arg\max_k \hat{f}_{\text{bag}}(x),$$

   where $\hat{f}_{\text{bag}}(x)$ is a K-dimensional vector indicating the proportion of bootstrapped classifiers predicting each class.

Bagging

- Denote by $\hat{\mathcal{P}}$ the empirical distribution putting equal probability $1/N$ on each of the data points (x_i, y_i).
- The "true" bagging estimate is defined by

$$\mathrm{E}_{\hat{\mathcal{P}}} \hat{f}^*(x),$$

where $\boldsymbol{Z}^* = \{(x_1^*, y_1^*), \ldots, (x_N^*, y_N^*)\}$ and each

$$(x_i^*, y_i^*) \sim \hat{\mathcal{P}}.$$

- The bagging estimator is a Monte-Carlo estimator of the true bagging estimator, approaching the true one for $B \to \infty$.
- The bagged estimator $\hat{f}_{\text{bag}}(x)$ will only differ from $\hat{f}(x)$ on the original dataset when the latter is a nonlinear or adaptive function of the data.

Why does bagging work? – Regression

- Assume the training data (x_i, y_i), $i = 1, \ldots, N$ are independently drawn from a distribution $\mathcal{P}$.
- Consider the ideal aggregate estimator

$$f_{\text{ag}}(x) = \mathrm{E}_{\mathcal{P}}(\hat{f}^*(x)).$$

- $\hat{f}^*(x)$ is the estimator for the dataset $\boldsymbol{Z}^*$ sampled from $\mathcal{P}$.
- One obtains

$$\begin{aligned}
\mathrm{E}_{\mathcal{P}}[Y - \hat{f}^*(x)]^2 &= \mathrm{E}_{\mathcal{P}}[Y - f_{\text{ag}}(x) + f_{\text{ag}}(x) - \hat{f}^*(x)]^2 \\
&= \mathrm{E}_{\mathcal{P}}[Y - f_{\text{ag}}(x)]^2 + \mathrm{E}_{\mathcal{P}}[\hat{f}^*(x) - f_{\text{ag}}(x)]^2 \\
&\geq \mathrm{E}_{\mathcal{P}}[Y - f_{\text{ag}}(x)]^2.
\end{aligned}$$

- Hence true population aggregation never increases mean squared error.

Why does bagging work? – Regression / 2

- The bagged estimate is not $f_{\text{ag}}(x)$ but rather an approximation based on $\hat{\mathcal{P}}$.
- There are two opposite effects of bootstrapping:
  - Variability is reduced by aggregating over the bootstrap estimates.
  - The bagged estimate based on bootstrapping from the original dataset is only an approximation to $f_{\text{ag}}(x)$.

  $\Rightarrow$

  - Instable predictors are improved.
  - Stable predictors might become worse.
- In the bagged estimates often any simple structure of the base learners implying easy interpretability is lost.

Why does bagging work? – Classification

- For classification under the 0–1 loss bagging a good classifier can make it better, while bagging a bad classifier can make it worse.
- Example:
  - Suppose $Y = 1$ for all x and the classifier $\hat{G}(x)$ predicts $Y = 1$ with probability 0.4 and $Y = 0$ with probability 0.6.
  - Then the misclassification rate of $\hat{G}(x)$ equals 0.6 and that of the bagged classifier is 1.0.

Why does bagging work? – Classification / 2

- Assume that bagging gives the consensus estimate of *weak learners*.
  - Let the Bayes optimal decision at x be $G(x) = 1$ in a two-class example.
  - Suppose each of the weak learners G_b^* have an error rate $e_b = e < 0.5$ and let

$$S_1(x) = \sum_{b=1}^{B} I(G_b^*(x) = 1)$$

    be the consensus vote for class 1 at x.
  - Since the weak learners are assumed to be independent,

$$S_1(x) \sim \text{Bin}(B, 1 - e)$$
$$\Pr(S_1 > B/2) \to 1 \quad \text{as } B \text{ gets large.}$$

Random forests

- Random forests are a substantial modification of bagging trees.
- A large collection of *de-correlated trees* are constructed and averaged.
- Trees are ideal candidates for bagging because:
  - They can capture complex interaction structures in the data.
  - If grown sufficiently deep, they have relatively low bias.
  - They are notoriously noisy / instable.
  - Trees generated in bagging are identically distributed, thus having the same mean.
- An average of B i.i.d. random variables with variance σ^2 has variance σ^2/B. However, if they are pairwise positively correlated with ρ, the variance equals

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2.$$

Random forests / 2

- Random forests improve bagging by reducing the correlation between the trees without increasing the variance too much.
- De-correlation is achieved by randomly subsetting the input variables.

Random forests: correlated trees

Assume

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i,$$

with X_i identically distributed with mean μ and variance σ^2 and pairwise correlations $\text{Cor}(X_i, X_j) = \rho$. This implies:

$$\text{E}(\bar{X}) = \mu$$

$$\text{Var}(\bar{X}) = \frac{\sigma^2}{N} + \frac{2}{N^2}\sum_{i=2}^{N}\sum_{j=1}^{i-1}\text{COV}(X_i, X_j) = \frac{\sigma^2}{N} + \frac{2\rho\sigma^2}{N^2}\sum_{i=1}^{N-1} i$$

$$= \frac{\sigma^2}{N} + \frac{2\rho\sigma^2}{N^2}\frac{N(N-1)}{2} = \rho\sigma^2 + \frac{\sigma^2}{N}(1-\rho)$$

Random forests: algorithm

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\boldsymbol{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

Random forests: algorithm / 2

To make a prediction at a new point x:

- Regression:

$$\hat{f}_{\text{rf}}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x; \Theta_b),$$

  where Θ_b characterizes the bth random forest tree in terms of split variables, cutpoints at each node and terminal-node values.

- Classification:
  Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree.
  Then

$$\hat{C}_{\text{rf}}^{B} = \text{majority vote}\{\hat{C}_b(x)\}_1^B.$$

Details of random forests

- Regression:
  - The random forest prediction is obtained by averaging over the preditions from each tree.
  - The default value for the subset size m is $\lfloor p/3 \rfloor$ and the minimum node size is five.
- Classification:
  - The random forest uses majority vote based on the class votes from each tree.
  - The default value for the subset size m is $\lfloor \sqrt{p} \rfloor$ and the minimum node size is one.

Out-of-bag (OOB) error

- For each observation $z_i = (x_i, y_i)$ construct its random forest predictor by averaging over trees corresponding to bootstrap samples where z_i is not contained.
- The bootstrap samples are drawn with replacement from the data and have the same size. Thus on average a data point is not contained in $1/e \approx 37\%$ of the B bootstrap samples.
- The OOB error is almost identical to that obtained by N-fold cross-validation.
  $\Rightarrow$ Computationally much less demanding than N-fold cross-validation.

Variable importance

- *Mean decrease impurity importance*: results in Gini importance for Gini gain as impurity measure.
  1. For each split in the tree T_b the improvement in split criterion is attributed to the corresponding split variable.
  2. The improvements are summed for each variable for a tree and averaged over trees.
- *Mean decrease importance / permutation accuracy importance*:
  1. Determination of the prediction accuracy of the OOB observations for a tree T_b.
  2. Permutation of the OOB observations of the jth variable.
  3. Determination of the prediction accuracy of the tree T_b for the permuted observations.
  4. The performance decrease due to permutation is averaged over all trees.

Variable importance / 2

- Mean decrease impurity importance is computationally less expensive than mean decrease importance which requires the permutation step.
- Mean decrease impurity importance may give biased results if the predictor variables vary in their measurement scales, i.e., continuous versus categorical and categorical with different number of categories.
  - In a simulation study the mean Gini importance of five unrelated predictor variables was assessed.
  - The binary predictor variable had a much lower mean Gini importance than a standard normal predictor variable.
  - The 4-category predictor variable had a similar mean Gini importance than a standard normal predictor variable.

Example: spam

- Binary classification problem.
- 57 potential predictor variables.

```
> data("spam", package = "ElemStatLearn")
> library("randomForest")
> set.seed(1234)
> rf <- randomForest(spam ~ ., data = spam,
+   importance = TRUE, ntree = 300)
> rf
```

Example: spam / 2

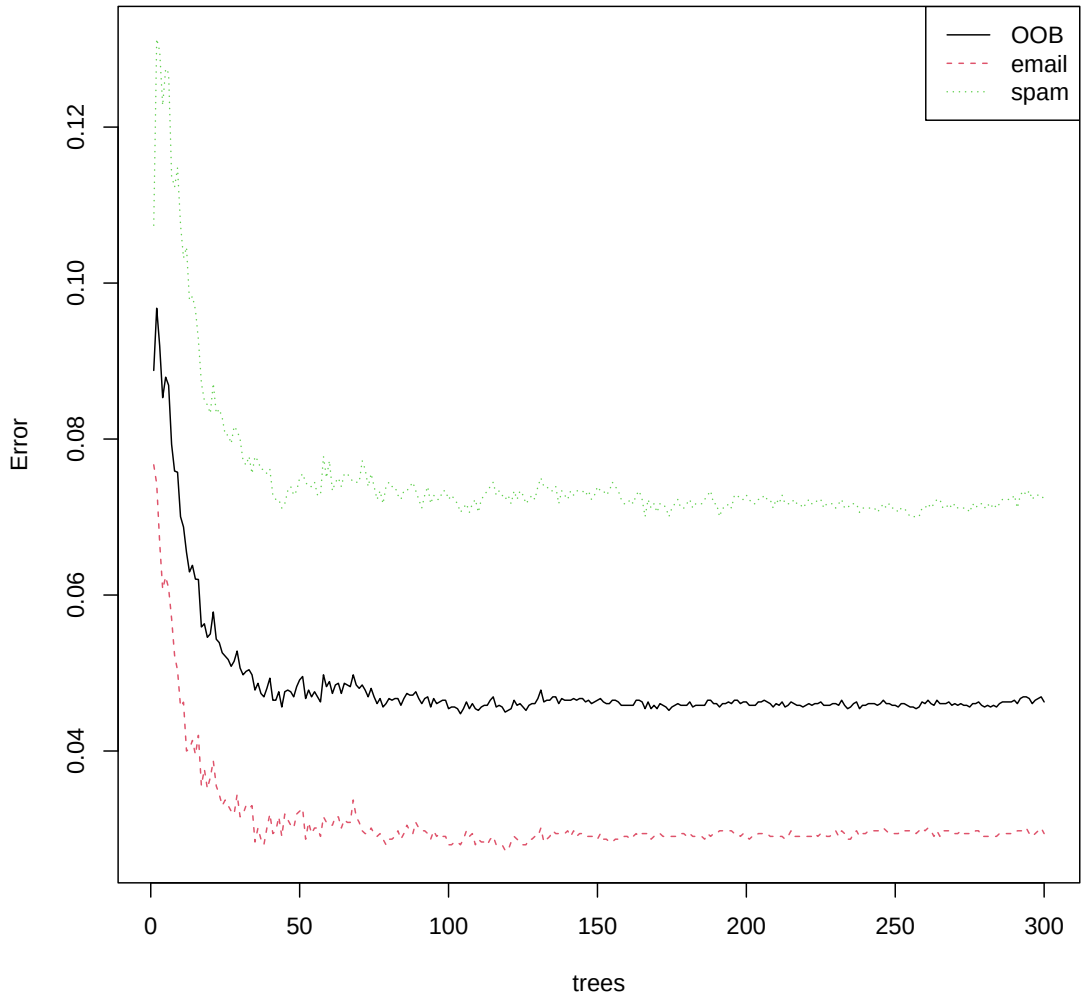
```
Call:
 randomForest(formula = spam ~ ., data = spam, importance =
               Type of random forest: classification
                     Number of trees: 300
No. of variables tried at each split: 7

        OOB estimate of  error rate: 4.63%
Confusion matrix:
      email spam class.error
email  2706   82  0.02941176
spam    131 1682  0.07225593
```

Example: spam / 3

```
> head(rf$err.rate)
            OOB      email       spam
[1,] 0.08879619 0.07669617 0.1074130
[2,] 0.09677419 0.07425150 0.1313131
[3,] 0.09194396 0.06669912 0.1297376
[4,] 0.08530559 0.06072351 0.1227840
[5,] 0.08793356 0.06234916 0.1274876
[6,] 0.08689559 0.06088632 0.1269276
```

Example: spam / 4

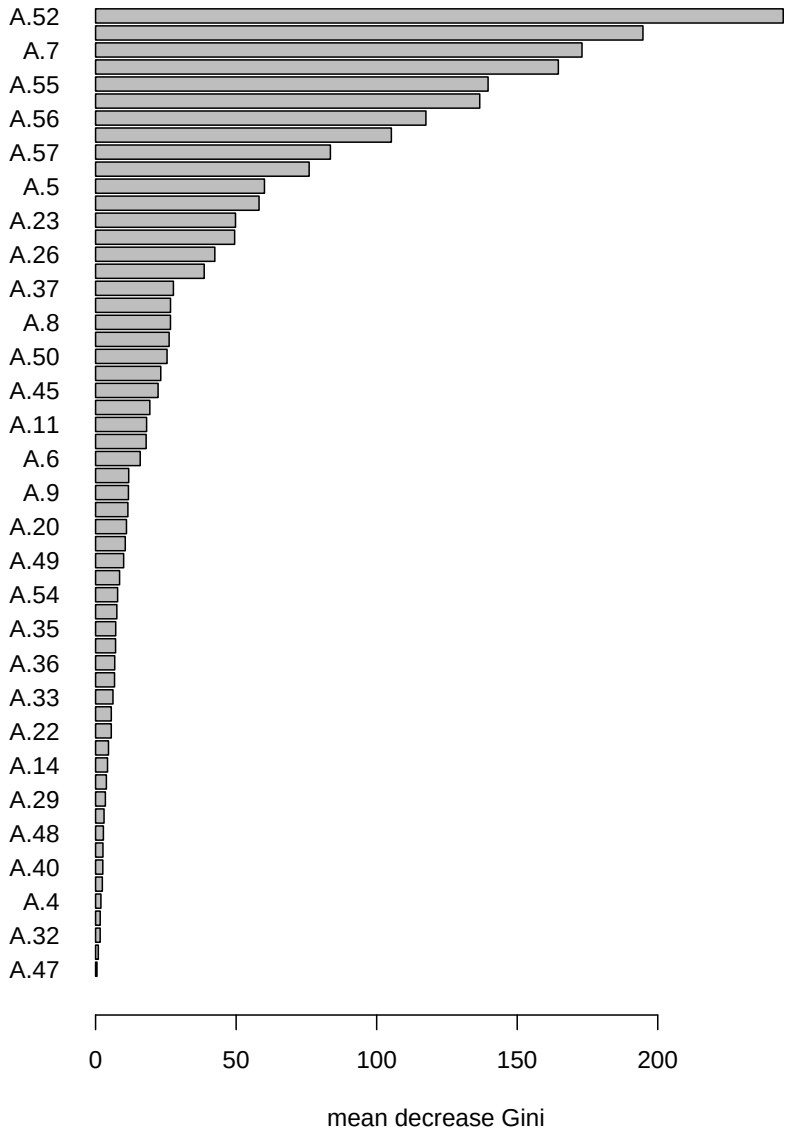


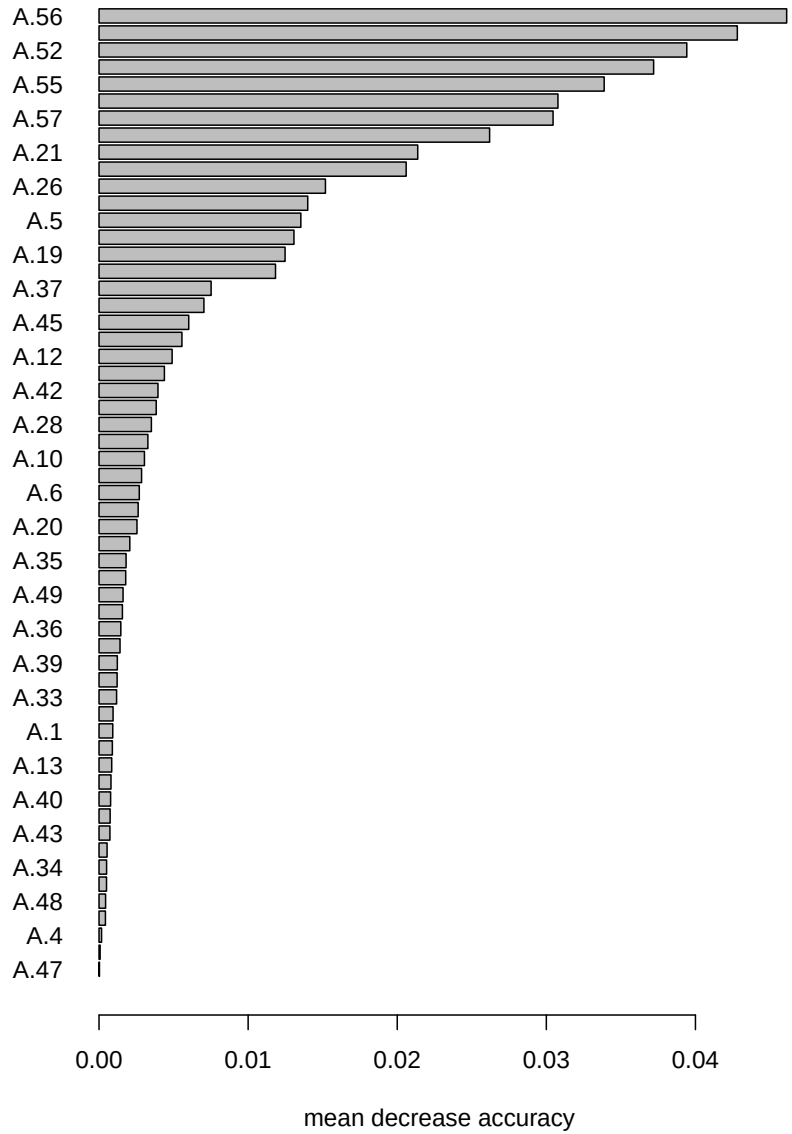
Example: spam / 5

The default is to standardize the mean decrease accuracy as suggested by Breiman and Cutler (2008). However, standardization has been shown to give unsuitable results as it depends on the number of trees grown.

```
> options(digits = 2)
> head(importance(rf, scale = FALSE))
      email    spam MeanDecreaseAccuracy MeanDecreaseGini
A.1 0.00042 0.00169                0.00092             7.6
A.2 0.00122 0.00211                0.00157            11.5
A.3 0.00074 0.00720                0.00328            26.2
A.4 0.00026 0.00004                0.00017             1.9
A.5 0.01017 0.01871                0.01354            60.1
A.6 0.00220 0.00348                0.00270            15.9
```

Example: spam / 6





Random forests and overfitting

- When the number of variables is large, but the fraction of relevant variables small, random forests are likely to perform poorly with small m.
- The number of bootstrap samples B is a hyperparameter of random forests, but does not impact on the overfitting behavior.
- The selection of the tree depth influences the overfitting behavior.

Analysis of random forests – regression

- The limiting form ($B \to \infty$) of the random forest regression estimator is

$$\hat{f}_{\text{rf}}(x) = \mathrm{E}_{\Theta|\boldsymbol{Z}} T(x; \Theta(\boldsymbol{Z})),$$

  where the dependence on the training data $\boldsymbol{Z}$ has been made explicit.
- Estimation at a single target point x is considered.

Analysis of random forests – regression / 2

- The variability is given by:

$$\text{Var}(\hat{f}_{\text{rf}}(x)) = \rho(x)\sigma^2(x),$$

where

- $\rho(x)$ is the *sampling* correlation between any pairs of trees used in averaging:

$$\rho(x) = \text{Cor}(T(x; \Theta_1(\boldsymbol{Z})), T(x; \Theta_2(\boldsymbol{Z}))),$$

where $\Theta_1(\boldsymbol{Z})$ and $\Theta_2(\boldsymbol{Z})$ are a randomly drawn pair of random forest trees grown to the randomly sampled $\boldsymbol{Z}$;

- $\sigma^2(x)$ is the sampling variance of any single randomly drawn tree,

$$\sigma^2(x) = \text{Var}(T(x; \Theta(\boldsymbol{Z}))).$$

Analysis of random forests – regression / 3

- $\rho(x)$ is the correlation induced by the sampling distribution of $\boldsymbol{Z}$ and $\boldsymbol{\Theta}$. I.e., $\rho(x)$ is the theoretical correlation between a pair of random-forest trees evaluated at x, induced by repeatedly making training sample draws $\boldsymbol{Z}$ from the population, and then drawing a pair of random forest trees.
- The total variance of a single tree can be decomposed into two parts:

  $$\mathrm{Var}_{\Theta,\boldsymbol{Z}} T(x;\Theta(\boldsymbol{Z})) = \mathrm{Var}_{\boldsymbol{Z}} \mathrm{E}_{\Theta|\boldsymbol{Z}} T(x;\Theta(\boldsymbol{Z})) + \mathrm{E}_{\boldsymbol{Z}} \mathrm{Var}_{\Theta|\boldsymbol{Z}} T(x;\Theta(\boldsymbol{Z})),$$

  where the first summand is equal to $\mathrm{Var}_{\boldsymbol{Z}}(\hat{f}_{\mathrm{rf}}(x))$.

Analysis of random forests – regression / 4

- The bias is given by:

$$\text{Bias}(x) = \mu(x) - \text{E}_{\boldsymbol{Z}}\hat{f}_{\text{rf}}(x) = \mu(x) - \text{E}_{\boldsymbol{Z}}\text{E}_{\Theta|\boldsymbol{Z}}T(x;\Theta(\boldsymbol{Z})).$$

  So the bias is typically larger than the bias of an unpruned tree, since the randomization and reduced sample space impose restrictions.

- Performance improvement of random forests compared to trees is solely due to *variance reduction*.