STAT406- Methods of Statistical Learning Lecture 16

Matias Salibian-Barrera

UBC - Sep / Dec 2018

Failing hard and failing often

Using Packages You Don't Understand

Hey, it runs!

O RLY?

Joe Hilgard

Random forests

(1) `for(b in 1:B)`

  (a) Draw a bootstrap sample from the training data

  (b) Grow a “random forest tree” as follows: for each terminal node:

    (i) Randomly select `m` features

    (ii) Pick the best split among these

    (iii) Split the node into two children

  (c) Repeat (b) to grow a (very very) large tree

(2) Return the ensemble of trees $(T_b)_{1 \leq b \leq B}$

Random forests

- Given a new point $\mathbf{x}$, for regression we use

$$\hat{f}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} T_b(\mathbf{x})$$

- For classification:

$$\hat{f}(\mathbf{x}) = \text{majority vote among} \Big\{ T_b(\mathbf{x}) , \; 1 \le b \le B \Big\}$$

Q: why not average conditional prob's?

Out-of-bag error estimates

- Each bagged tree is trained on a bootstrap sample
- Predict the observations not in the bootstrap sample with that tree
- One will have “about” $B/3$ predictions for each point in the training set
- These can be used to estimate the prediction error (classification error rate) without having to use CV

Out-of-bag error estimates

- For each training observation $(y_i, \mathbf{x}_i)$, obtain a prediction using only those trees in which $(y_i, \mathbf{x}_i)$ was **NOT** used
- In other words, let $\mathcal{I}_i$ the set of trees (bootstrap samples) where $(y_i, \mathbf{x}_i)$ does not appear, then

$$\hat{y}_i \; = \; \frac{1}{|\mathcal{I}_i|} \sum_{j \in \mathcal{I}} T_j(\mathbf{x}_i)$$

Random forests

- This error estimate can be computed at the same time as the trees are being built
- When this error estimate is stabilized we can stop adding trees to the ensemble

Example

OOB example

Random forests

- Feature ranking - relative importance of each variable
- Given a single tree T, at each node t split we can compute the sum of reductions in sum of squares (or gini or deviance measures) m_t^2
- We assign this squared measure m_t^2 to the variable (feature) used in the split

Random forests

- To each feature, we assign the sum of "squared gains" attributed to it
- For the i-th variable X_i we have

$$\mathcal{J}_i^2(T) = \begin{cases} m_t^2 & \text{if split involved } X_i \\ 0 & \text{otherwise} \end{cases}$$

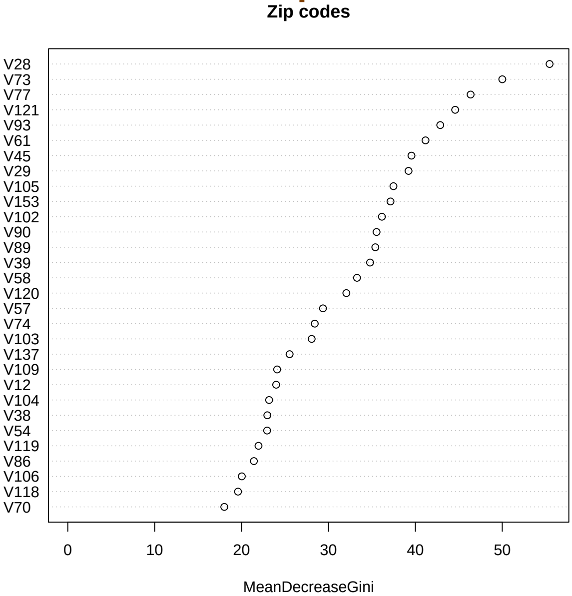
Random forests

- For a random forest we use

$$\mathcal{J}_i^2 = \sum_T \mathcal{J}_i^2(T)$$

- In other words, we sum (or average) the importance of the variable across the trees in the forest

Random Forest - plot



Ensembles

- Ensembles of classifiers
- Combine classifiers trained on the same (or similar [e.g. bootstrapped]) data
- Consensus is reached by (equally weighted) voting or averaged estimated probabilities.
- Bagging and Random Forests are examples of ensembles.

Boosting

- Originally proposed for classification
- Main idea: sequentially re-train a simple classifier assigning more importance to points that were previously misclassified

Boosting

- The end result is a weighted average of all the classifiers
- Interesting ideas:
  - Not all components of the ensemble are treated equally
  - Members of the ensemble use information about other members
  - The underlying loss function has a "margin" (unlike 0-1 losses)

Boosting - AdaBoost.M1

Algorithm. Data $(y_{\mathbf{i}}, \mathbf{x}_{\mathbf{i}})$, with $y_{\mathbf{i}} \in \{-1, 1\}$

- Set initial weights $w_{\mathbf{i}} = 1/n$, $1 \leq \mathbf{i} \leq n$
- For $\mathbf{j} = 1, \ldots, K$
- Build a classifier $T_{\mathbf{j}}(\mathbf{x})$ to the data using weights $w_{\mathbf{i}}$, $1 \leq \mathbf{i} \leq n$

Boosting - AdaBoost.M1

- Let

$$e_j = \sum_{i=1}^{n} w_i \, I\left(y_i \neq T_j(\mathbf{x}_i)\right) \Bigg/ \sum_{\ell=1}^{n} w_\ell$$

- Let $\alpha_j = \log\left((1 - e_j)/e_j\right)$ and

$$w_i = w_i \exp\left(\alpha_j \, I\left(y_i \neq T_j(\mathbf{x}_i)\right)\right), \; i = 1, \ldots, n$$

- Final classifier:

$$T(\mathbf{x}) = \operatorname{sign}\left(\sum_{j=1}^{K} \alpha_j \, T_j(\mathbf{x})\right)$$