M8L2: Trees - and forests — TMA4268, 07.03.2019

→ both for regression and classification

1) Tree constructed by minimizing a criterion $j = 1, \dots, J$ (leaf nodes)

$$\sum_{j=1}^{J} \sum_{i: x_i \in R_j} (y_i - \hat{y}_{R_j})^2$$

leaf node — (RSS) — mean of obs in R_j

$k = 1, \dots, K$ (# classes)

$$\sum_{j=1}^{J} \sum_{k=1}^{K} \hat{p}_{jk}(1 - \hat{p}_{jk}) \quad \text{(Gini)}$$

$$\hat{p}_{jk} = \frac{n_{jk}}{N_j}$$ — obs in j of class k / obs in leaf node j

$$\sum_j \sum_k n_{jk} \log(\hat{p}_{jk})$$ (similar?)

Deviance

$$\sum_j \sum_k \hat{p}_{jk} \log(\hat{p}_{jk})$$

give same splits in tree — cross entropy

2) by recursive binary splitting
- greedy approach

$$R_1(j, s) = \{x \mid x_j < s\} \text{ and } R_2(j, s) = \{x \mid x_j \geq s\}$$

until some stopping criterion is reached.

3) (Full) tree used for prediction by dropping (new) observations down the tree and predict as

$[x_i, y_i \; i = 1, \dots, n$ training data$]$

$$\hat{y} = \frac{1}{N_j} \sum_{i: x_i \in R_j} y_i$$

regression

$\hat{y}$ = majority vote among y's in R_j

or

$$\hat{p}_{jk} = \frac{1}{N_j} \underbrace{\sum_{i: x_i \in R_j} I(y_i = k)}_{n_{jk}}$$

Observe: - linear boundary: hard
- interaction: easy

PRUNING

- overfitting
- unnessesary splits } reduce the number of leaf nodes
- interpretability

Many possible pruned trees → we use cost complexity pruning

$T \subset T_0$

T: subtree, T_0: full tree

$$C_\alpha = Q(T) + \alpha |T|$$

$Q(T)$: cost function (RSS, Gini, Misclass rate)

α: penalty

$|T|$: size of tree = # leaf nodes

Given $\alpha \Rightarrow$ get a pruned tree $\Rightarrow$ See ALG on Module page

→ Bias-variance tradeoff:

BAGGING

$\hat{f}^{*b}(x)$ = decision tree from bootstrap sample b

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

Make bushy trees. B need to be large enough (500-2000)
Out of bag OOB estimation of error.

It is hard to examine B tree for interpretation, instead we look at "variable importance plots".

RANDOM FOREST

Bagging may not help so much when we have a strong predictor and all our B trees are very similar
→ need to decorrelate trees.

only consider $m < p$ predictors at each split in the tree

- $\sqrt{p}$ classification
- $p/3$ regression

} tuning parameter

B needs to be large enough (not a tuning parameter)

BOOSTING (only regression trees here)

$\hat{f}(x) = 0$, $r_i = y_i$

- d = # splits
- B = # trees
- λ = learning rate

tuning parameters

$b = 1, \dots, B$

$\hat{f}^b(x)$ fitted with d splits to r_i-data

$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \cdot \hat{f}^b(x)$

$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \cdot \hat{f}^b(x)$$