6.7 Tree-Based Methods

Navona Calarco

The University of Toronto

Introduction

This section will cover tree-based methods for regression and classification. These methods are easy to interpret yet their prediction accuracy is not as good as the other methods we have seen.

We will also cover several methods that combine multiple trees to overcome this problem:

- Bagging
- Random forests
- Boosting
- Bayesian additive regression trees

Regression Trees

Regression trees are able to make predictions for quantitative responses based on predictors. The method is summarized in two steps:

1. Divide the predictor space $X_1, X_2, \dots, X_p$ into J distinct non-overlapping regions $R_1, R_2, \dots R_J$.
2. The predicted response of an observation that falls into the region R_j is the mean of the response values of the training observations in R_j.

How do we choose the regions $R_1, R_2, \dots R_J$?

Constructing $R_1, R_2, \dots R_J$

The goal is to find regions $R_1, R_2, \dots R_J$ (boxes for simplicity) that minimize

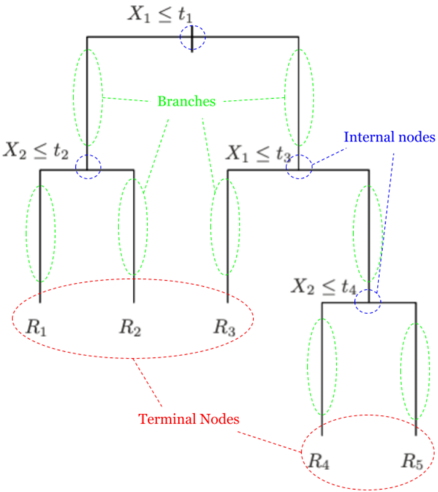
$$\mathrm{RSS} = \sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j}\right)^2$$

- $\hat{y}_{R_j}$ is the mean response for the training observations in the j-th box.
- We cannot consider every possible splitting so we use **recursive binary splitting** to construct the regions.

Recursive Binary Splitting

1. Consider all predictors $X_1, \ldots, X_p$ and all possible values of the cutpoint s for each predictor.
2. Compute the RSS of the tree for each predictor and cut point combination.
3. Select the predictor X_j and cut-point s such that splitting the predictor space into the regions $\{X|X_j < s\}$ and $\{X|X_j \geq s\}$ results in the greatest reduction in RSS.
4. Repeat steps 1-3 to minimize the RSS within each of the regions until we decide to stop (stop when we reach no more than 5 observations per region or some other criteria).

Regression Trees



- The **terminal nodes** of the tree are the resulting regions.
- The **internal nodes** are the points on the tree where the predictor space is split.
- The **branches** of the tree are the segments that connect the nodes.

Tree Pruning

Using recursive binary splitting by itself yields a large tree T_0 that is prone to overfitting, high variance, and poor test error rates. So we use **cost complexity pruning** (aka weakest link pruning) after the fact to shrink the tree. For each α there is a subtree $T \subset T_0$ that minimizes

$$\sum_{m=1}^{|T|} \sum_{i:} x_i \in R_m \left(y_i - \hat{y}_{R_m}\right)^2 + \alpha|T|$$

- α is the tuning parameter.
- $|T|$ is the number of terminal nodes of the tree T.
- R_m is the region (rectangle) that corresponds to the mth terminal node.
- $\hat{y}_{R_m}$ is the mean of the training observations in R_m .

Tuning Parameter α

The tuning parameter α from the cost complexity method has several features.

- It is non-negative.
- It controls the trade-off between the complexity of the subtree and its fit to the training data.
  - $\alpha = 0$ implies maximum complexity so $T = T_0$
  - As α increases, there is a penalty for having meany terminal nodes so the subtree will shrink.
- We use a validation set of cross-validation to choose a value for α.

Regression Trees

The complete process for building a regression tree is summarised by:

- Build a large tree using **recursive binary splitting** on the training data. Stop when each terminal node has no more than some fixed number of observations.
- Perform **cost complexity pruning** to the large tree with many values of α to obtain a sequence of best subtrees.
- Use K-fold **cross-validation** to choose the α.
- Return the subtree from step 2 that corresponds to the chosen value of α.

Classification Trees

A classification tree can be used to make predictions for a qualitative response. It assigns observations to the most commonly occurring class of training observations in the region to which the observation belongs.

Before we describe the method, we define three indices.

Classification Error Rate

The **classification error rate** is the fraction of the training observations in a region that do not belong to the most common class of the region.

$$E = 1 - \max_{k} \left(\hat{p}_{mk} \right)$$

- $\hat{p}_{m_k}$ is the proportion of training observations in the mth region that are from the kth class.

- We want a small E.

Entropy

Entropy is another measure of the total variance.

$$D = -\sum_{k=1}^{K} \widehat{p}_{mk} \log \widehat{p}_{mk}.$$

- E is small if a node contains predominantly observations from a single class.

Classification Trees

The complete process for building a regression tree is summarised by:

1. Build a large tree using **recursive binary splitting** on the training data. You can use either the Gini index or entropy with the goal of minimizing it with every split. Stop when each terminal node has no more than some fixed number of observations.
2. Perform **cost complexity pruning** to the large tree with many values of α to obtain a sequence of best subtrees using classification error rate.
3. Use K-fold **cross-validation** to choose the α.
4. Return the subtree from step 2 that corresponds to the chosen value of α.

Exercises: Trees for Regression and Classification}

Open the Tree Based Methods Exercises R Markdown or Jupyter Notebook file.

- Go over the "Fitting Classification Trees" section together as a class.
- 7 minutes for students to complete the questions at the end of the section.
- Questions should be completed at home if time does not allow.
- Go over the "Fitting Regression Trees" section together as a class.

Pros and Cons of Trees

Advantages of using decision trees for regression and classification:

- They are very easy to interpret
- They can be displayed graphically
- They can handle qualitative predictors without the need for dummy variables

Disadvantages:

- The predictive accuracy of trees is lower than other methods
- They are not very robust so a small change in the data can cause a large change in the tree

Bagging

Bagging, also known as the bootstrap, is a method that can applied to decision trees in order to reduce their variance. It results in improved prediction accuracy but worse interpretability.

Bagging Regression Trees

1. Sample B bootstrapped training sets from the data set.
2. Construct B regression trees from the bootstrapped training sets and leave them unpruned.
3. For a given test observation, average the resulting predictions from the B trees.

Bagging

Bagging Classification Trees

1. Sample B bootstrapped training sets from the data set.
2. Construct B classification trees from the bootstrapped training sets and leave them unpruned.
3. For a given test observation, record the prediction that results from each tree and then take the most commonly occurring class among the predictions.

The number of trees B is not of great importance as long as it is sufficiently large so the error is relatively constant.

Out-of-Bag Error Estimation

Out-of-bag error estimation can be used to estimate the test error of a bagged model.

- Each bagged tree uses on average two thirds of the observations to fit the tree.
- The remaining observations are call out-of-bag (OOB) observations.

The method is outlined by:

1. For each observation we predict the response using the trees for which the observations was OOB.
2. Average the predicted quantitative responses or choose the most common predicted qualitative response.
3. Compute the MSE or classification error and use this as the estimated test error for the bagged model.

Random Forests

Random forests operate very similarly to bagging but often provide better results.

1. Sample B bootstrapped training sets from the data set.
2. Construct B trees from the bootstrapped training sets but when building each split in the tree using recursive binary splitting, only a random subset of $m \approx \sqrt{p}$ predictors are considered as candidates.
3. For a given test observation, record the prediction that results from each tree and then take the average OR the most commonly occurring class among the predictions.

- Choosing between $m < p$ predictors at each split ensures that the trees will not always choose the most powerful predictors.
- This results in uncorrelated trees with a lower resulting variance.

Exercises: Bagging and Random Forests

Open the Tree Based Methods Exercises R Markdown or Jupyter Notebook file.

- Go over the “Bagging and Random Forests” section together as a class.
- 2 minutes for students to complete the questions at the end of the section.
- Questions should be completed at home if time does not allow.

Boosting

Boosting is very different from bagging since it grows trees sequential, using information from previously grown trees. Although this method can also be applied to classification trees we will only outline the procedure for regression trees.

1. Fit a regression tree to the data set in the usual way.
2. Compute the residuals for this tree.
3. Fit a new decision tree with d nodes using the residuals as the response values.
4. Take this to be the base tree.
5. Update the residuals of the tree and fit a new tree, adding a shrunken version of this tree to the base tree.
6. Repeat step 5 B times.

The formal algorithm is on the following slide.

Boosting for Regression Trees

1. Fit a regression tree to the training set and compute the resulting residuals r_i.
2. Set $\hat{f}(x)$ to be a blank tree and $r_i = y_i$ for all i.
3. For $b = 1, 2, \ldots, B$:
   - Fit a tree $\hat{f}^b$ with d internal nodes to the training data (X, r).
   - Update $\hat{f}$ by adding a shrunken version of $\hat{f}^b$

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

   - Update the residuals

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

4. Output the boosted tree

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

Boosting

The key parameters of this procedure are:

- B: the number of trees.
  - If B is too large, we risk overfitting.
  - Cross-validation is used to select B.
- λ: the shrinkage parameter.
  - A small positive number that control the rate at which boosting learns.
- d: the number of splits in each tree.
  - Controls the complexity of the boosted tree.

Exercises: Boosting

Open the Tree Based Methods Exercises R Markdown or Jupyter Notebook file.

- Go over the "Boosting" section together as a class.
- 7 minutes for students to complete the questions at the end of the section.
- Questions should be completed at home if time does not allow.

Bayesian Additive Regression Trees

Bayesian additive regression trees (BART) work iteratively. At each iteration, K regression trees are created and then summed.

- The K trees for the 1st iteration have a single root node: the mean response values divided by the total number of trees.
- The K trees are summed so the 1st iteration tree is the mean response value.
- In the bth iteration, the response values of the kth tree from the $b-1$ iteration are subtracted by the predictions from the other $K-1$ trees (partial residual).
- Then the kth tree is updated by choosing a random perturbation from a set of possible perturbations to improve the fit of the partial residual.
- The K trees are summed to acquire the bth iteration tree.

Bayesian Additive Regression Trees

Before we describe the algorithm more formally we need some notation:

- K: the number of regression trees.
- B: the number of iterations of the BART algorithm.
- $\widehat{f}_k^b(x)$: the prediction for x from the kth regression tree used in the bth iteration.

The perturbations that we discussed in step 4 can only

- Add or prune branches from the tree.
- Change the prediction for each terminal node of the tree.

The first few iterations of BART do not provide good results, known as the **burn-in** period, so we usually exclude these L samples from the final average.

Bayesian Additive Regression Trees

1. Let $\hat{f}_1^1(x) = \hat{f}_2^1(x) = \cdots = \hat{f}_K^1(x) = \frac{1}{nK}\sum_{i=1}^n y_i$.
2. Then $\hat{f}^1(x) = \sum_{k=1}^K \hat{f}_k^1(x) = \frac{1}{n}\sum_{i=1}^n y_i$.
   - For $b = 2, \dots, B$:
     - For $k = 1, 2, \dots K$:
       - For $i = 1, \dots, n$, compute the partial residual

$$r_i = y_i - \sum_{k'<k} \hat{f}_{k'}^b(x_i) - \sum_{k'>k} \hat{f}_{k'}^{b-1}(x_i)$$

       - Fit a new tree $\hat{f}_k^b(x)$ to r_i by randomly perturbing $\hat{f}_k^{b-1}(x)$. Perturbations that improve the fit are favored.
   - Compute $\hat{f}^b(x) = \sum_{k=1}^K \hat{f}_k^b(x)$.
3. Compute the mean, excluding the L burn-in samples: $\hat{f}(x) = \frac{1}{B-L}\sum_{b=L+1}^B \hat{f}^b(x)$

Exercises: Bayesian Additive Regression Trees

Open the Tree Based Methods Exercises R Markdown or Jupyter Notebook file.

- Go over the “Bayesian Additive Regression Trees” section together as a class.

References

Chapter 8 of the ISLR2 and ISLP books:

James, Gareth, et al. "Tree-Based Methods." An Introduction to Statistical Learning: with Applications in R, 2nd ed., Springer, 2021.

James, Gareth, et al. "Tree-Based Methods." An Introduction to Statistical Learning: with Applications in Python, Springer, 2023.