LECTURE 7

BOGDAN ICHIM

Decision Trees

Advantages:

- simple and useful for interpretation;
- good graphical representations;
- many situations where they are useful and fit.

Disadvantages:

- in general, decision trees are not competitive with other supervised learning approaches in terms of prediction accuracy;
- small change in data leads to large change in the final tree (non-robust).

In order to combat the disadvantages one may use methods which are producing multiple trees which are combined to yield a single prediction (for example by voting).

Examples of methods aggregating a large number of trees:

- Bagging
- Random Forests
- Boosting

Types of decision trees:

- Regression Trees
- Classification Trees

Elements of a tree:

- Terminal nodes = leaves
- Internal nodes
- Branches

Main idea: Segmenting the predictor space into a number of simple regions.

Prediction = "mean" of the training observations corresponding to one region.

Example (2 dim)

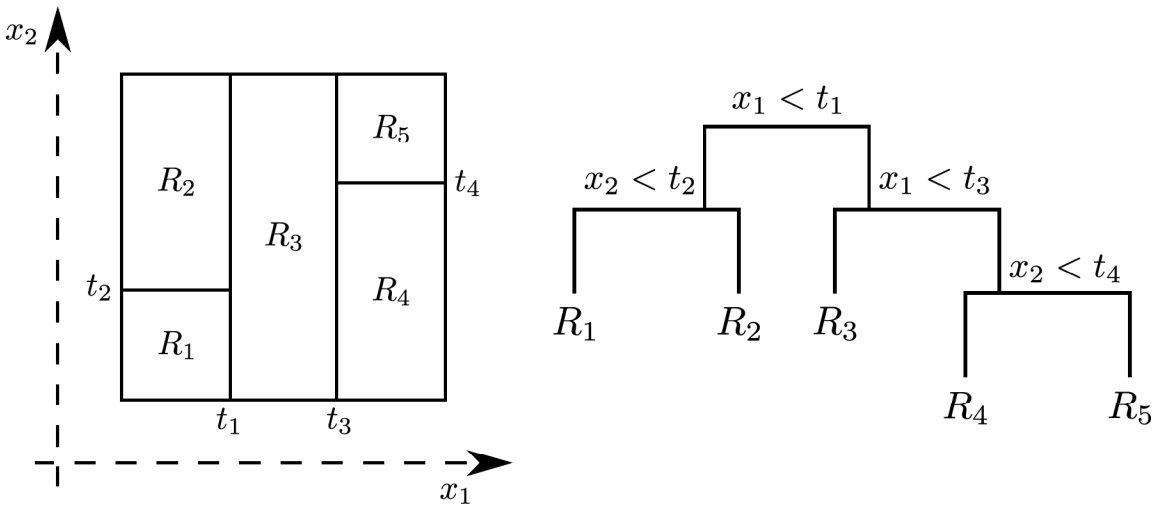

FIGURE 1

Regression Trees

Building a regression tree:

(1) We divide the predictor space (i.e. the set of possible values for $X_1, X_2 \dots X_p$) into J distinct and non-overlapping regions $R_1, R_2 \dots R_J$.

(2) For every observation in the region R_j we make the same prediction, which is the mean of the response values for all the the training observations in R_j.

In order to get the regions we need to set up some clear goals:

- divide the predictor space into high-dimensional rectangles (boxes) for simplicity and interpretation ability.
  (**Note:** in theory, the regions may have any shape!)
- minimize a certain criterion, for example find boxes $R_1, R_2 \dots R_J$ that minimize the RSS, given by:

$$\boxed{RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i \quad \hat{y}_{R_j})^2}$$

(**Note:** there are other performance measures we may want to minimize.)

In general, it is computationally **impossible** to consider **all** partitions of the feature space into boxes.

One possible solution is to use **recursive binary splitting**. This approach has the following properties:

- **Top-Down:** begins at the top of the tree, i.e. at a point where all observations belong to a single region;
- **Greedy:** at each step of the tree-building process the best split is made at that particular step (local), rather than looking ahead and picking a split that will lead to a better tree in some future step (global).

Recursive Binary Splitting

I. We consider all predictors $X_1, X_2 \dots X_p$ and all possible values S for each of the predictors and then chose the predictor and the corresponding cutpoint such that the resulting tree (1 node) has the lowest RSS possible.

In other words, for any k and S we define the pair of half-spaces:

$$R_1(k, S) = \{X | X_k < S\} \text{ and } R_2(k, S) = \{X | X_k \geq S\},$$

and choose the k and S that minimize the equation:

$$\sum_{\{i|x_i \in R_1(k,S)\}} (y_i - \hat{y}_{R_1})^2 + \sum_{\{i|x_i \in R_2(k,S)\}} (y_i - \hat{y}_{R_2})^2,$$

where $\hat{y}_{R_1} = \overline{y}_{R_1}$ estimates the mean response for all training observations in R_1 (and similarly $\hat{y}_{R_2}$).

II. We repeat the process in order to split the data further in each of the resulting regions, with the goal of minimizing RSS. However, we do not split the entire predictor space, but only one of the two previously identified regions $\Rightarrow$ three regions.
We repeat with the three regions $\Rightarrow$ four regions etc...

III. We stop when a certain stopping criterion is reached, for example no region contains more than $\frac{N}{100}$ observations.

IV. Once the regions $R_1, R_2 \dots R_J$ have been created (for example by using I, II, III), we predict the response for a given test observation using the mean of all training observations in the region to which the observation belongs.

Classification Trees

Main difference in contrast to regression trees: we predict that each observation belongs to the **most commonly occurring class** of training observations in the region to which it belongs. (However, sometimes we are also interested in the **class proportions** among the training observations that fall into that region).

We may use recursive binary splitting to build a classification tree, but we cannot use RSS as a performance criterion.

Denote by $\hat{p}_{mk}$ the proportion of training observations in the m-th region that are from the k-th class. We may use the following classification measures:

(1) **Classification Error Rate:**

$$E = 1 - \max_k(\hat{p}_{mk})$$

where a small value of E means that one region contains many observations from the same class. However, it seems not be to really good in practice.

(2) **Gini Index:** Let K be the number of classes:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

(3) **Cross-Entropy:**

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

Note: $0 \leq \hat{p}_{mk} \leq 1 \Rightarrow 0 \leq -\hat{p}_{mk} \log \hat{p}_{mk}$

Tree versus Linear Model

Linear (Regression) Model:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

Regression Tree:

$$Y = c_1 \mathbb{1}_{(x \in R_1)} + c_2 \mathbb{1}_{(x \in R_2)} + \cdots + c_J \mathbb{1}_{(x \in R_J)} + \epsilon$$

Linear (Classification) Model *Classification Tree*

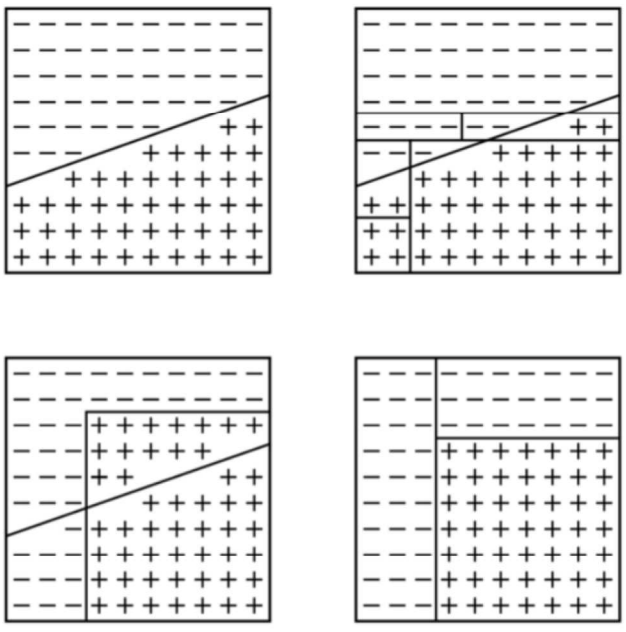

FIGURE 2