40.016: The Analytics Edge
Week 8 Lecture 1

Forecasting the Supreme Court's decisions with CARTs (Part 1)

Term 5, 2022

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Course overview

Domains:

Wine analytics, Challenger, Framingham Heart Study, Oscars, Sports, Economics, Lex Analytics, Ethics in Analytics, Text Analytics, Netflix, Aviation.

Tools:

Linear Regression, Principal Component Analysis, Logistic Regression, Multinomial Logit, Model Selection, Classification and Regression Trees, Random Forests, Naïve Bayes Classifier, Clustering, Optimization.

Outline

Outline

Brief Introduction to the US Supreme Court

What is the Supreme Court? See:

`https://www.youtube.com/watch?v=QVIVEKY5YWI`

Key points:

- Nine justices, or judges, appointed by the US President
- Lifetime tenure
- The court handles $\sim$ 80 cases per year
- A decision happens when the majority agrees on an outcome (discrete responses $\rightarrow$ classification problem)

Brief Introduction to the US Supreme Court (cont'd)

How does a case get to the Supreme Court? See:
`https://www.youtube.com/watch?v=KEjgAXxrkXY`

Categories for case selection:

- Cases of national importance
- Lower court invalidates federal law
- Resolve split decision

A **key point**: The decision is to affirm or reverse, so we can model it as a **binary variable**.

Outline

The Supreme Court Forecasting Project

This is a study published by Martin et al. (2004), who:

- Used data spanning the period 1994-2001 (longest period with the same justices) $\rightarrow$ training dataset
- Compared predictions (for the year 2002) made by legal experts and statistical models $\rightarrow$ testing dataset or validation dataset
- Found very interesting results:
  - Accuracy on the entire court decision: models, 75%; experts, 59.1%
  - Accuracy at the individual justice level: models, 66.7%; experts, 67.9%

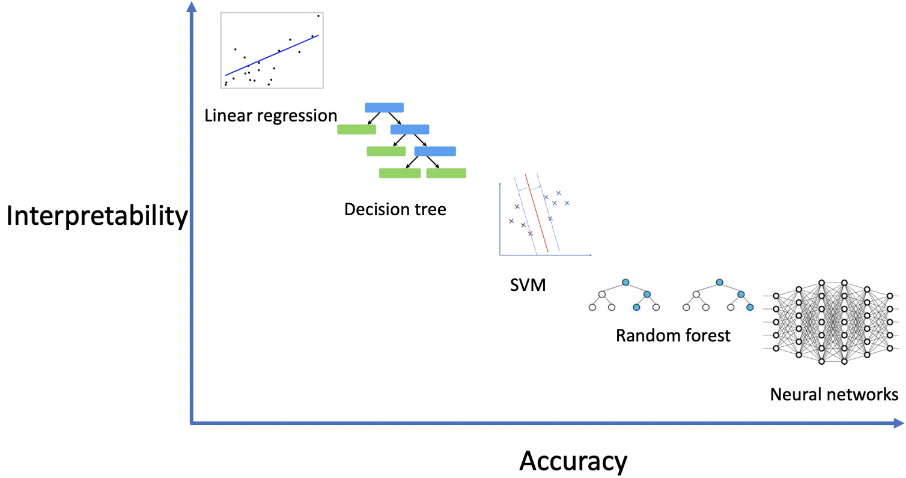
The Supreme Court Forecasting Project

Our (training) data:

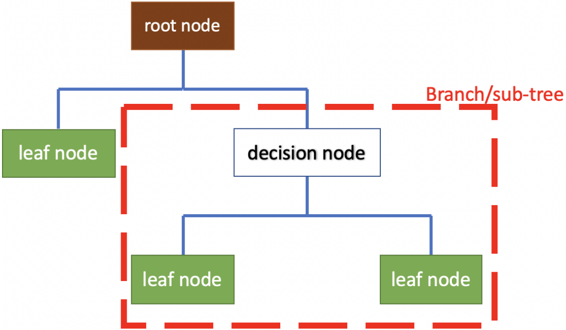
- 623 observations (about 80 cases per year), 20 variables
- Output variable, or predictand: `result`, which takes value 0 (liberal) or 1 (conservative). Liberal: reverse; conservative: affirm
- Input variables, or predictors:
  - `petit`: petitioner type (e.g., US, employer, injured person)
  - `respon`: type of respondent
  - `circuit`: circuit of origin of the case
  - `unconst`: whether the petitioner argued the constitutionality of a law of practice
  - `lctdir`: ideaological direction of the lower court (liberal or conservative)
  - `issue`: issue area of the case

Outline

Supervised learning



Decision Trees

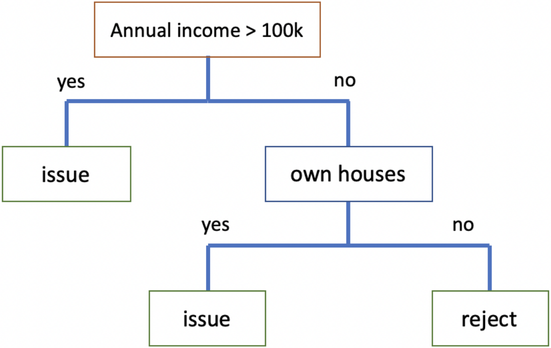


- The root node: the node that starts the graph, including all data in the training set
- Leaf nodes: final nodes of the tree, where the predictions are made.

Example

Question: How a bank determines whether to issue loans to customers?

Answer:



Decision Trees

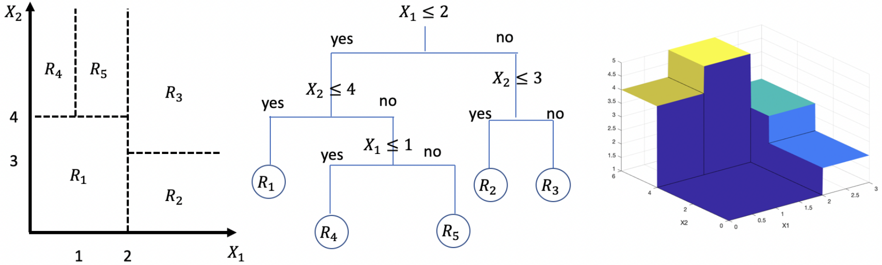
- Decision Trees can be applied to both regression and classification problems
- The term Classification And Regression Tree (CART) is used to refer to procedures that learn a Classification or Regression Tree
- **Note:** We begin by considering Regression Trees

Outline

Regression Trees

Intuition: Suppose we are working on a regression problem with response variable Y and predictors X_1 and X_2. The underlying idea of Regression Trees is to divide, or partition, the predictor space into a number of regions, where we then apply a simple model.

Example:



How do we learn a Regression Tree?

Given a dataset $\{(X_1, y_1), \cdots, (X_n, y_n)\}$, with variable $X_i \in \mathbb{R}^p$, response y_i. The goal of a regression tree is to construct a function $f(\cdot)$ to minimize RSS:

$$\min \sum_{i=1}^{n} (f(X_i) - y_i)^2.$$

There are two main steps:

Step 1. Partition the predictor space into J distinct and non-overlapping regions $(R_1, R_2, \ldots, R_J)$.

Step 2. For every observation that falls into the j-th region R_j, we make the same prediction c_j.

$$\text{Estimated function:} \quad \hat{f}(X) = \sum_{j=1}^{J} c_j 1_{R_j}(X).$$

How do we learn a Regression Tree? (cont'd)

In **Step 2**, we need to solve

$$\min_{c_1,\ldots,c_J} \sum_{j=1}^{J} \sum_{i:X_i \in R_j} (y_i - c_j)^2.$$

Based on the criterion minimization of the sum of squares, c_j takes the mean of the response values for the observations in R_j, that is

$$c_j = \text{average}(y_i | X_i \in R_j).$$

How do we learn a Regression Tree? (cont'd)

In **Step 1**, the problem of partitioning the predictor space into J regions can be formulated as follows:

$$\min_{R_1,\ldots,R_J} \sum_{j=1}^{J} \sum_{i:X_i \in R_j} (y_i - c_j)^2.$$

In general, the problem is computationally unfeasible. To solve it, we use a top-down, greedy approach known as **recursive binary splitting**

– A heuristic algorithm to find $R_1, \cdots, R_J$

Recursive binary splitting

- Start with all variables in one region
- Consider all predictors $X^{(1)}, \dots, X^{(p)}$ and all possible values of the cut (or split) point s, and choose the predictor and cut point s.t. the resulting partition has the lowest RSS
- We repeat the process, splitting one of the two previously identified regions. The process continues until an exit condition is met (e.g., minimum number of points in each region)

Recursive binary splitting

Given the k-th predictor and the cut point s, we define the following half-planes

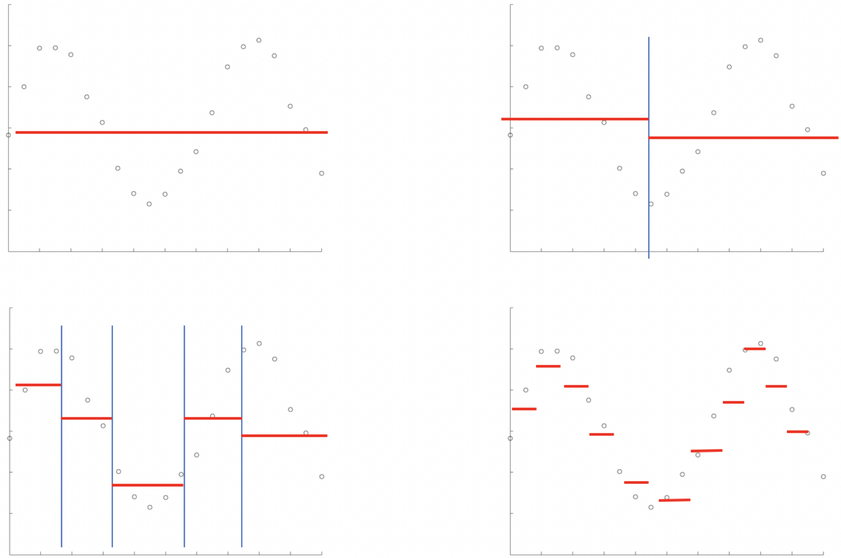
$$R_1(k,s) = \{X | X^{(k)} < s\} \text{ and } R_2(k,s) = \{X | X^{(k)} \geq s\},$$

And we seek the value of k and s that minimizes

$$\underbrace{\sum_{i: X_i \in R_1(k,s)} (y_i - c_1)^2}_{\text{error in } R_1} + \underbrace{\sum_{i: X_i \in R_2(k,s)} (y_i - c_2)^2}_{\text{error in } R_2}.$$

Specifically, we first fix k, find the best s; then we get p different policies, choose the one with the lowest RSS.

Illustration of regression tree



Outline

Classification Trees

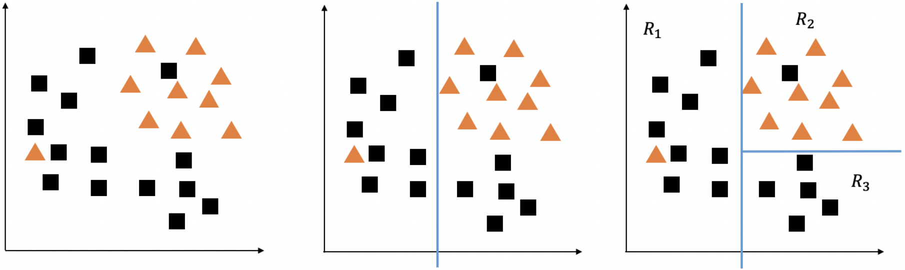
Similarities with Regression Trees:

- The representation for the CART model is a binary tree, where each node (except leaf nodes) has two child nodes.
- The predictions in one leaf node are the same.

Two differences w.r.t. Regression Trees:

- For each region, the prediction c_j is the most commonly occurring class
- When learning a tree, we cannot use the RSS. Instead, we use a measure of impurity (a split is pure if, for all branches, all the instances choosing a branch fall within the same class)

Example



Measures of impurity

1. Classification error rate:

$$E = 1 - \max_k (p_{mk})$$

where p_{mk} is the proportion of training observations in the m-th region that are from the k-th class.

- $N_m = \#$ instances in the region R_m
- $N_{mk} = \#$ instances in the region R_m belonging to class k
- $p_{mk} = \frac{N_{mk}}{N_m}$

Measures of impurity (cont'd)

2. Gini index:

$$G = \sum_{k=1}^{K} p_{mk}(1 - p_{mk})$$

where K is the total number of classes, and G varies between 0 and 0.5.

Example: $K = 2$ classes, 10 instances in the region R_m:

Case 1: (Best case) all instances are from class 1,

$$p_{m1} = 1,\ p_{m2} = 0,\ G = 1(1-1) + 0(1-0) = 0$$

Case 2: (Worst case) 5 instances are from class 1, 5 are from class 2,

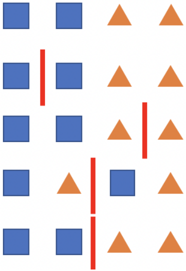
$$p_{m1} = \frac{1}{2},\ p_{m2} = \frac{1}{2},\ G = \frac{1}{2}(1 - \frac{1}{2}) + \frac{1}{2}(1 - \frac{1}{2}) = \frac{1}{2}$$

Measures of impurity (cont'd)

If we partition the points into 2 regions, the total Gini index is

$$G = \frac{\#instances\ in\ Region\ 1}{\#total\ instances} \times G(\text{Region } 1) + \frac{\#instances\ in\ Region\ 2}{\#total\ instances} \times G(\text{Region } 2)$$

Data:



Case 1: $G(\text{Region 1}) = 0,\ G(\text{Region 2}) = \frac{4}{9},\ \ G = \frac{1}{4} \times 0 + \frac{3}{4} \times \frac{4}{9} = \frac{1}{3}$

Case 2: $G(\text{Region 1}) = \frac{4}{9},\ G(\text{Region 2}) = 0,\ \ G = \frac{3}{4} \times \frac{4}{9} + \frac{1}{4} \times 0 = \frac{1}{3}$

Case 3: $G(\text{Region 1}) = \frac{1}{2},\ G(\text{Region 2}) = \frac{1}{2},\ \ G = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = \frac{1}{2}$ Worst case

Case 4: $G(\text{Region 1}) = 0,\ G(\text{Region 2}) = 0,\ \ G = \frac{1}{2} \times 0 + \frac{1}{2} \times 0 = 0$ Best case

Measures of impurity (cont'd)

3. Entropy:

$$D = -\sum_{k=1}^{K}(p_{mk}\log_2(p_{mk})).$$

Since $0 \le p_{mk} \le 1$, D varies between 0 and 1. (Note: let $0\log_2 0 = 0$)

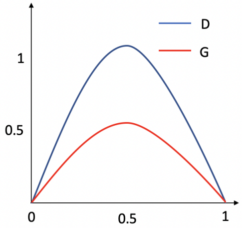
Example: $K = 2$ classes, 10 instances in the region R_m:

Case 1: (Best case) all instances are from class 1,

$$p_{m1} = 1,\ p_{m2} = 0,\ D = -1\log_2 1 - 0\log_2 0 = 0$$

Case 2: (Worst case) 5 instances are from class 1, 5 are from class 2,

$$p_{m1} = \frac{1}{2},\ p_{m2} = \frac{1}{2},\ D = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$$



Outline

Back to R!

To learn CARTs, we will use the function `rpart`, implemented in the package ... `rpart`:

```
rpart(formula, data, method, control, ...)
```

Advantages and Disadvantages of CARTs

Pros:

- Interpretability
- Can be displayed graphically
- Can handle qualitative predictors (that take no continuous values)
- No assumptions on the relationship between input and output variables

Cons:

- They are not very accurate
- Not robust

References

- Martin et al. (2004) Competing approaches to predicting supreme court decision making. *Perspectives on Politics*, 2 (4), 761767.
- James et al. (2014) *An Introduction to Statistical Learning with Applications in R*, Springer, 2014. Chapter 8.1.