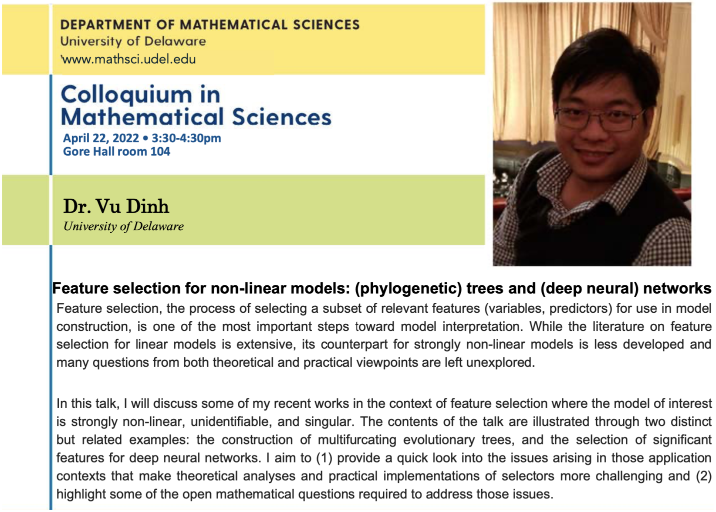
Mathematical techniques in data science

Lecture 30: Decision trees



DEPARTMENT OF MATHEMATICAL SCIENCES
University of Delaware
www.mathsci.udel.edu

Colloquium in Mathematical Sciences

April 22, 2022 • 3:30-4:30pm
Gore Hall room 104

Dr. Vu Dinh
University of Delaware

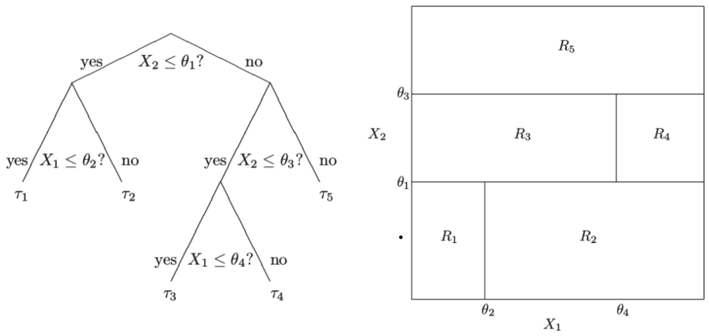
Feature selection for non-linear models: (phylogenetic) trees and (deep neural) networks

Feature selection, the process of selecting a subset of relevant features (variables, predictors) for use in model construction, is one of the most important steps toward model interpretation. While the literature on feature selection for linear models is extensive, its counterpart for strongly non-linear models is less developed and many questions from both theoretical and practical viewpoints are left unexplored.

In this talk, I will discuss some of my recent works in the context of feature selection where the model of interest is strongly non-linear, unidentifiable, and singular. The contents of the talk are illustrated through two distinct but related examples: the construction of multifurcating evolutionary trees, and the selection of significant features for deep neural networks. I aim to (1) provide a quick look into the issues arising in those application contexts that make theoretical analyses and practical implementations of selectors more challenging and (2) highlight some of the open mathematical questions required to address those issues.

Tree-based methods

- Partition the feature space into a set of rectangles
- Fit a simple model (e.g. a constant) in each rectangle
- Conceptually simple yet powerful



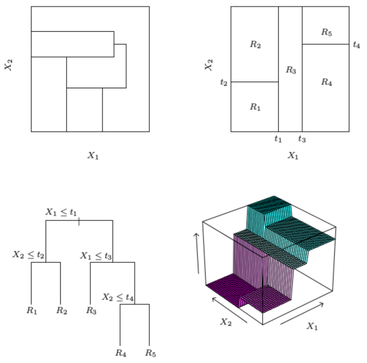
Izenman, 2013, Figure 9.1.

Tree-based methods

- Advantages:
  - Often mimics human decision-making process (e.g. doctor examining patient).
  - Very easy to explain and interpret.
  - Can handle both regression and classification problems.
- Disadvantage: Basic implementation is generally not competitive compared to other methods.
- However, by aggregating many decision trees and using other variants, one can improve the performance significantly.
- Such techniques may lead to state-of-the-art models. However, in doing so, one loses the easy interpretability of decision trees.

Decision trees

To simplify, we will only consider **binary** decision trees.



ESL, Figure 9.2.

Top Left: Not binary. Top Right: binary.

Bottom Left: Tree corresponding to Top Right partition. Bottom Right: Prediction surface.

How to grow a decision tree?

Regression tree:

- Data: $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times p}$.
- Each observation: $(y_i, x_i) \in \mathbb{R}^{p+1}$, $i = 1, \dots, n$.

Suppose we have a partition of $\mathbb{R}^p$ into M regions $R_1, \dots, R_m$.

We predict the response using a constant on each R_i:

$$f(x) = \sum_{i=1}^{m} c_i \cdot \mathbf{1}_{x \in R_i}.$$

In order to minimize $\sum_{i=1}^n (y_i - f(x_i))^2$, one needs to choose:

$$\hat{c}_i = \mathrm{ave}(y_j : x_j \in R_i).$$

How do we determine the regions R_i, i.e., how do we “grow” the tree?

We need to decide:

1. Which variable to split.
2. Where to split that variable.

How to grow a decision tree?

- Finding a (globally) optimal tree is generally computationally infeasible.
- We use a greedy algorithm.

Consider a splitting variable $j \in \{1, \dots, p\}$ and splitting point $s \in \mathbb{R}$.
Define the two half-planes:

$$R_1(j,s) := \{x \in \mathbb{R}^p : x_j \le s\}, \qquad R_2(j,s) := \{x \in \mathbb{R}^p : x_j > s\}.$$

We choose j, s to minimize

$$\min_{j,s} \left[\min_{c_1 \in \mathbb{R}} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2 \in \mathbb{R}} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right].$$

- The determination of the splitting point s can be done very quickly.
- Hence, determining the best pair (j, s) is **feasible**.

Repeat the same process to each block.

Stoping and pruning

- Generally, the process is stopped for a given region when there are **less than** 5 observations in that region.

Problem with previous methodology:

- Likely to **overfit** the data.
- Can lead to poor prediction error.

Pruning the tree. Strategy: Grow a large tree (overfits), and the prune it (better).

- **Weakest link pruning:**

(a.k.a cost complexity pruning)
Let $T \subset T_0$ be a **subtree** of T_0 with $|T|$ **terminal nodes**. For $\alpha > 0$, define:

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

Pick a subtree minimizing $C_\alpha(T)$.

Stoping and pruning

Pick a subtree $T \subset T_0$ minimizing:

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

(Here, $\hat{y}_{R_m}$ =average response for observations in R_m.)

- α is a **tuning parameter**.
- Trade-off between fit of the model, and tree complexity.
- Choose α using cross-validation.

Once α has been chosen by CV, use whole dataset to find the tree corresponding to that value.

Classification trees

- So far, we discussed **regression** trees (continuous output).
- We can easily modify the methodology to predict a *categorical* output.
- We only need to modify our *splitting and pruning criteria*.

For continuous variables, we picked a constant in each box R_i to minimize the sum of squares in that region:

$$\min_{c \in \mathbb{R}} \sum_{x_i \in R_i} (y_i - c)^2.$$

As a result, we choose:

$$\hat{c}_i = \frac{1}{N_i} \sum_{x_k \in R_i} y_k,$$

where N_i denotes the number of observations in R_i.

Classification trees

Similarly, when the output is categorical, we can count the proportion of class k observations in node i:

$$\hat{p}_{ik} = \frac{1}{N_i} \sum_{x_l \in R_i} \mathbf{1}_{y_l \in R_i}.$$

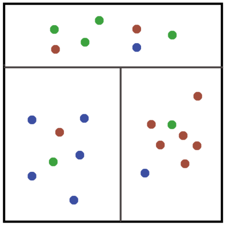
We then classify the observations in node i using a **majority vote**:

$$k(i) := \operatorname*{argmax}_k \hat{p}_{ik}.$$

Different measures are commonly used to determine how good a given partition is (and how to split a given partition):

1. **Misclassification error:** $\frac{1}{N_i} \sum_{x_l \in R_i} \mathbf{1}_{y_l \neq k(i)} = 1 - \hat{p}_{i,k(i)}$.
2. **Gini index:** $\sum_{k=1}^K \hat{p}_{ik}(1 - \hat{p}_{ik}) = 1 - \sum_{k=1}^K \hat{p}_{ik}^2$.
   (Probability that a randomly chosen point is incorrectly classified.)
3. **Entropy:** $-\sum_{k=1}^K \hat{p}_{ik} \log \hat{p}_{ik}$.
   (Measure of “disorder” in a given category.)

Examples



Let us focus on the **top** box.

- (Gini index) Error from classifying according to proportions:

$$\begin{aligned} P(\text{error}) &= P(\text{error}|\text{green})P(\text{green}) + P(\text{error}|\text{blue})P(\text{blue}) + P(\text{error}|\text{red})P(\text{red}) \\ &= 3/7 \cdot 4/7 + 6/7 \cdot 1/7 + 5/7 \cdot 2/7 = 4/7. \end{aligned}$$

- (Entropy) The probability distribution associated to the top box: $(4/7, 2/7, 1/7)$.

$$\text{Entropy} = -(4/7)\log_2(4/7) - (2/7)\log_2(2/7) - (1/7)\log_2(1/7) \approx 1.38.$$

Best case possible: $(1,0,0), (0,1,0), (0,0,1)$. Entropy $= 0$.

Worst case possible $(1/3, 1/3, 1/3)$. Entropy $= 1.58$.