Decision trees

Nathanaël Carraz Rakotonirina

Mathématiques Informatique et Statistique Appliquées (MISA)
Université d'Antananarivo

Model

Decision trees or **Classification and regression trees (CART)** are non-parametric models that recursively partition the input space, and define a local model in each region.

The tree consists of a set of nested decision rules.

1. At each node i, the feature dimension d_i of the input vector $\boldsymbol{x}$ is compared to a threshold value t_i (for categorical features to each of the possible values for that feature).
2. The input is passed down to the left if above the threshold or to the right otherwise.
3. At the leaves of the tree, the model specifies the predicted output for any input that falls into that region.
4. We associate a mean response to each of the leaf or region for regression and a distribution over the class labels for classification.

The output for a specific region can be defined as:

$$w_j = \frac{\sum_{i=1}^{N} y_i \mathbb{I}(\boldsymbol{x}_i \in R_j)}{\sum_{i=1}^{N} \mathbb{I}(\boldsymbol{x}_i \in R_j)}$$

A regression tree can be defined by

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{j=1}^{J} w_j \mathbb{I}(\boldsymbol{x} \in R_j)$$

- R_j is the region specified by the jth leaf node
- w_j is the predicted output for that node
- J is the number of nodes
- $\boldsymbol{\theta} = \{(R_j, w_j) : j = 1...J\}$

The regions are defined by the feature dimensions and thresholds used in each split on the path from the root to the leaf.

Model fitting

Finding the optimal partitioning of the data is NP-complete. The alternative is to use a greedy procedure, in which we iteratively grow the tree one node at a time.

At node n, we define the following:

- the set of examples that reach this node $\mathcal{D}_n = \{(\boldsymbol{x}_i, y_i) \in N_n\}$
- the partition of the examples into the left subtree for real-valued or categorical inputs $\mathcal{D}_n^L(j,t) = \{(\boldsymbol{x}_i, y_i) \in N_n : y_{i,j} \leq t\}$ or $\mathcal{D}_n^L(j,t) = \{(\boldsymbol{x}_i, y_i) \in N_n : y_{i,j} = t\}$
- the partition of the examples into the right subtree for real-valued or categorical inputs $\mathcal{D}_n^R(j,t) = \{(\boldsymbol{x}_i, y_i) \in N_n : y_{i,j} > t\}$ or $\mathcal{D}_n^L(j,t) = \{(\boldsymbol{x}_i, y_i) \in N_n : y_{i,j} \neq t\}$

We choose the best feature j_n to split on, and the best value for that feature, t_n, as follows:

$$(j_n, t_n) = \arg \min_{j \in \{1..D\}} \min_{t \in \mathcal{T}_j} \frac{|\mathcal{D}_n^L(j,t)|}{|\mathcal{D}|} c(\mathcal{D}_n^L(j,t)) + \frac{|\mathcal{D}_n^R(j,t)|}{|\mathcal{D}|} c(\mathcal{D}_n^R(j,t))$$

- c is a cost function
- $\mathcal{T}_j$ is the set of possible thresholds for feature j which can be obtained by sorting the unique values of $\{x_{i,j}\}$

Cost function for regression

The cost function $c(\mathcal{D}_n)$ is used to evaluate the cost of node n.

Mean squared error

For regression, we use the mean squared error:

$$c(\mathcal{D}_n) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}_n} (y_i - \bar{y})^2$$

where $\bar{y} = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}_n} \mathbb{I}(y_i = c)$ is the mean of the response variable for examples reaching node n

Cost function for classification

We first compute the empirical distribution over class labels for this node:
$\hat{\pi}_{nc} = \frac{1}{|\mathcal{D}_n|} \sum_{i \in \mathcal{D}_n} \mathbb{I}(y_i = c)$

Gini index

For classification, we use the **Gini index** (which is the expected error rate):

$$G_i = \sum_{c=1}^{C} \hat{\pi}_{nc}(1 - \hat{\pi}_{nc})$$

Deviance

We can also define cost as the entropy or **deviance** of the node:

$$H_n = \mathbb{H}(\hat{\boldsymbol{\pi}}_i) = -\sum_{c=1}^{C} \hat{\pi}_{nc} \log \hat{\pi}_{nc}$$

Regularization

- We to stop the tree growing process according to some heuristic like :
  - minimum number of examples at a node
  - maximum depth
- Or we can grow the tree to its maximum depth, where no more splits are possible, and then to **prune** it back, by merging split subtrees back into their parent.

Advantages and disadvantages

- They are easy to interpret.
- They can easily handle mixed discrete and continuous inputs.
- They are insensitive to monotone transformations of the inputs so no need to standardize the data.
- They perform automatic variable selection.
- They are relatively robust to outliers.
- They are fast to fit, and scale well to large data sets.
- They can handle missing input features.

However,

- They do not predict very accurately compared to other kinds of model.
- They are unstable (small changes to the input data can have large effects on the structure of the tree).

More

Explore

- Handling missing input features
- Random forests
- Boosting