Lecture 7: Classification trees

Introduction to Machine Learning

Sophie Robert

L3 MIASHS | Semestre 2

2023-2024

Principle

Principle

Decision trees

Decision tree learning is a supervised learning approach that divides the space into subsets according **to linear rules on features** to split the *target* measured on the population into **homogeneous classes**.

Usually, trees are regrouped under the umbrella term **CART** (*classification and regression trees*), introduced by Breiman et al. in 1984 (see full book on Moodle).
Different algorithms exist (CART, ID3, C4.5, C5.0 ...).

We will see CART and ID3.

Displaying the trees

They are called trees because **they can be displayed as a tree**, each end-node corresponding to a classification choice: the decision process consists in **moving down in the tree**.

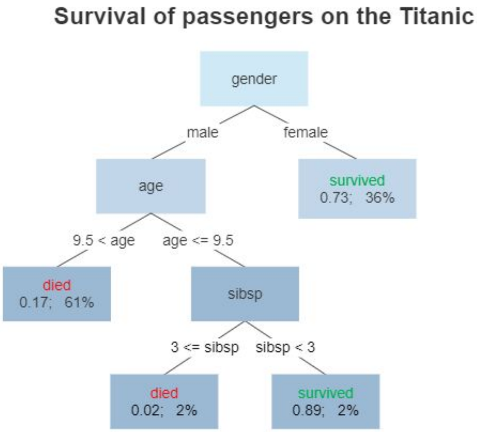
Displaying the trees

They are called trees because **they can be displayed as a tree**, each end-node corresponding to a classification choice: the decision process consists in **moving down in the tree**.

Reading the tree

- Each **final node** is a decision.
- Each **internal node** is a test.
- Each **branch** is the result from this test.

Example tree on Titanic dataset

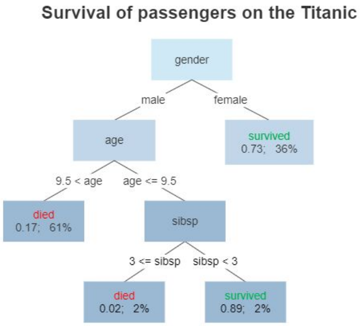


Example tree on Titanic dataset

Question

Given this tree, can you give the predicted class of:

- Rose, a female aged 20 years old with 1 spouse
- Jack, a male aged 23 years old without any sibling onboard



Principle

The space is recursively divided into smaller and smaller sections, by conducting a greedy search to identify the **split** providing **maximal information** within a tree, until a **stop criterion** is reached.

Question

What could be a good split criterion ? What could be good "information metric" regarding the quality of information split ?

Algorithm

Algorithm

Until the stop criterion is reached, recursively:

- Compute information gain for every feature and every possible split
- Select split minimizing wanted criterion (node impurity - information gain)

Mathematical framework

Given training vectors $x_i \in \mathbb{R}^n$ and corresponding label $y_i \in \mathbb{R}^l$, data at node Q_m with n_m samples. Let $\theta = (j, t_m)$ a candidate split of feature j and threshold t_m.
The impurity of the split using H as the loss function is:

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} H(Q_m^{left}(\theta)) + \frac{n_m^{right}}{n_m} H(Q_m^{right}(\theta))$$

Then select θ^* which minimizes impurity.

Stop criteria

Usual stop criteria:

- Number of end nodes
- Number of individuals per end node
- Optimal measure has reached a minimum value
- All features have been used or every individual is in the same class

Trees are very sensitive to overfitting !

Split criteria

Split criteria

Usual split criteria:

- **Gini Index**: CART trees (Breiman)
- **Entropy / Information loss**: ID3, C4.5, C5.0 (Quinlan)

Gini index

Gini index

The Gini index* measures of how often a **randomly chosen** element from the set would be incorrectly labeled if **it was randomly labeled** according to the distribution of labels in the subset.

We want this index to be as close to 0 as possible.

Gini index

Gini index

The Gini index* measures of how often a **randomly chosen** element from the set would be incorrectly labeled if **it was randomly labeled** according to the distribution of labels in the subset.

We want this index to be as close to 0 as possible.

Question

What is the Gini index equals to when every individual within a leaf has the same class ?

Gini index

Gini index is for every class the sum of the product of:

- The probability of being class p_i in node (frequency in the case of classification)
- The probability of not being class p_i: $\sum_{i \neq k}(1 - p_k) = (1 - p_i)$

Gini index

Gini index is for every class the sum of the product of:

- The probability of being class p_i in node (frequency in the case of classification)
- The probability of not being class p_i: $\sum_{i \neq k}(1 - p_k) = (1 - p_i)$

For a classification problem with J classes and p_i, the Gini index for node m:

$$I_g(m) = \sum_{i=1}^{J} p_i(1 - p_i) = 1 - \sum_{i=1}^{J} p_i^2$$

Gini index

Questions

- Give formula for the Gini index for binary classification
- Compute Gini index for the following sets:

$$S_1 = (\text{Legendary, Nonlegendary, Nonlegendary}),$$

$S_2 =$
(Nonlegendary, Nonlegendary, Nonlegendary, Nonlegendary)

Gini index

Questions

- Give formula for the Gini index for binary classification
- Compute Gini index for the following sets:

$$S_1 = (\text{Legendary, Nonlegendary, Nonlegendary}),$$

$S_2 =$
(Nonlegendary, Nonlegendary, Nonlegendary, Nonlegendary)

Gini index for binary classification (class 0 or 1):

$$I_g(m) = 1 - p_0^2 - p_1^2$$

Entropy and information gain

Entropy

Entropy* measures the the **uncertainty** of a set distribution: it is minimal (0) when every individual is correctly classified and maximal (1) when individuals are distributed evenly across classes.

$$Entropy(S) = -\sum_{i=1}^{J} p_i \times log_2(p_i)$$

(with the approximation that for $p_i = 0$, $p_i \times log_2(p_i) = 0$)

Entropy and information gain

Questions

- Give formula for entropy for binary classification
- Compute entropy for the following sets:

$$S_1 = (\text{Legendary, Nonlegendary, Nonlegendary}),$$

$\mathrm{S}_2 =$
(Nonlegendary, Nonlegendary, Nonlegendary, Nonlegendary)

Entropy and information gain

Questions

- Give formula for entropy for binary classification
- Compute entropy for the following sets:

$$S_1 = (\text{Legendary, Nonlegendary, Nonlegendary}),$$

$S_2 =$
$(\text{Nonlegendary, Nonlegendary, Nonlegendary, Nonlegendary})$

For binary classification:

$$Entropy(S) = -p_0 \times log_2(p_0) - p_1 \times log_2(p_1)$$

Information gain

Information gain

The information gain* computes the entropy gain of a new split, compared to the current entropy, normalized by the number of items in the set.

$$Gain(S, split) = \text{entropy}(S) - \sum_{i=0}^{j} \frac{|S_i|}{|S|} \times entropy(S_i)$$

Dealing with numerical variables

What about numerical variables ?

Dealing with numerical variables

What about numerical variables ?

Numerical variables are much more expensive to deal with: for each possible split, the chosen index needs to be computed to find the optimum split (either by using an optimization algorithm for finding the split or if the rand of variable is not too large, by ordering the values and computing the information gain at each round).

Example

Example: building a tree using Gini Index (CART)

Training dataset:

Height	Color	Label
Short	Brown	Legendary
Long	Black	Legendary
Short	Brown	Legendary
Short	Black	Not legendary
Short	Brown	Not legendary
Long	Black	Not legendary

Individual to classify on the tree (using Gini Index)

Height	Color	Label
Short	Black	?

Example: building a tree using Gini Index (CART)

Reminder on algorithm

- Compute for each feature the Gini index and select the split minimizing the index
- Stop when every individual have been classified OR no more features OR node homogeneity has been reached

Example: building a tree using Gini Index (CART)

Possible option: split on *Height*
If I select the split S_{height},

Example: building a tree using Gini Index (CART)

Possible option: split on *Height*
If I select the split S_{height},

$S_{(height=short)}$ = (Legendary, Legendary, Nonlegendary, Nonlegendary)

Example: building a tree using Gini Index (CART)

Possible option: split on *Height*
If I select the split S_{height},

$S_{(height=short)}$ = (Legendary, Legendary, Nonlegendary, Nonlegendary)
$I_g(S_{height=short}) = 1 - (\frac{2}{4})^2 - (\frac{2}{4})^2 = \frac{1}{2}$

Example: building a tree using Gini Index (CART)

Possible option: split on *Height*
If I select the split S_{height},

$S_{(height=short)}$ = (Legendary, Legendary, Nonlegendary, Nonlegendary)
$I_g(S_{height=short}) = 1 - (\frac{2}{4})^2 - (\frac{2}{4})^2 = \frac{1}{2}$

$S_{(height=long)}$ = (Legendary, Nonlegendary)

Example: building a tree using Gini Index (CART)

Possible option: split on *Height*
If I select the split S_{height},

$S_{(height=short)}$ = (Legendary, Legendary, Nonlegendary, Nonlegendary)
$I_g(S_{height=short}) = 1 - (\frac{2}{4})^2 - (\frac{2}{4})^2 = \frac{1}{2}$

$S_{(height=long)}$ = (Legendary, Nonlegendary)
$I_g(S_{height=long}) = 1 - (\frac{1}{2})^2 - (\frac{1}{2})^2 = \frac{1}{2}$

Example: building a tree using Gini Index (CART)

Possible option: split on *Height*
If I select the split S_{height},

$S_{(height=short)}$ = (Legendary, Legendary, Nonlegendary, Nonlegendary)
$I_g(S_{height=short}) = 1 - (\frac{2}{4})^2 - (\frac{2}{4})^2 = \frac{1}{2}$

$S_{(height=long)}$ = (Legendary, Nonlegendary)
$I_g(S_{height=long}) = 1 - (\frac{1}{2})^2 - (\frac{1}{2})^2 = \frac{1}{2}$

Total Gini index for split on height: $\frac{4}{6} \times \frac{1}{2} + \frac{2}{6} \times \frac{1}{2} = \frac{1}{2}$

Example: building a tree using Gini Index (CART)

Possible option: split on *Color*
If I select the split S_{color},

$$S_{(color=brown)} = (\text{Legendary, Legendary, Nonlegendary})$$

Example: building a tree using Gini Index (CART)

Possible option: split on *Color*
If I select the split S_{color},

$S_{(color=brown)} =$ (Legendary, Legendary, Nonlegendary)
$I_g(S_{color=brown}) = 1 - (\frac{2}{3})^2 - (\frac{1}{3})^2 = \frac{4}{9}$

Example: building a tree using Gini Index (CART)

Possible option: split on *Color*
If I select the split S_{color},

$S_{(color=brown)}$ = (Legendary, Legendary, Nonlegendary)
$I_g(S_{color=brown}) = 1 - (\frac{2}{3})^2 - (\frac{1}{3})^2 = \frac{4}{9}$

$S_{(color=black)}$ = (Legendary, Nonlegendary, Nonlegendary)

Example: building a tree using Gini Index (CART)

Possible option: split on *Color*
If I select the split S_{color},

$S_{(color=brown)} =$ (Legendary, Legendary, Nonlegendary)
$I_g(S_{color=brown}) = 1 - (\frac{2}{3})^2 - (\frac{1}{3})^2 = \frac{4}{9}$

$S_{(color=black)} =$ (Legendary, Nonlegendary, Nonlegendary)
$I_g(S_{color=black}) = 1 - (\frac{2}{3})^2 - (\frac{1}{3})^2 = \frac{4}{9}$

Example: building a tree using Gini Index (CART)

Possible option: split on *Color*
If I select the split S_{color},

$S_{(color=brown)} =$ (Legendary, Legendary, Nonlegendary)
$I_g(S_{color=brown}) = 1 - (\frac{2}{3})^2 - (\frac{1}{3})^2 = \frac{4}{9}$

$S_{(color=black)} =$ (Legendary, Nonlegendary, Nonlegendary)
$I_g(S_{color=black}) = 1 - (\frac{2}{3})^2 - (\frac{1}{3})^2 = \frac{4}{9}$

Total Gini index for split on color: $\frac{3}{6} \times \frac{4}{9} + \frac{3}{6} \times \frac{4}{9} = \frac{4}{9}$

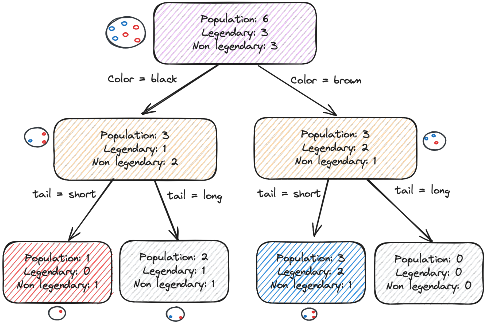
Example: building a tree using Gini Index (CART)

$\frac{4}{9} < \frac{1}{2}$ so I select the split on **color**.

Do I stop the algorithm ? No, there is a remaining feature and no node is pure.

Run algorithm using the remaining variable: **height**.

Example: building a tree using Gini Index (CART)



Pruning

Pruning to avoid overfitting

Decision trees have the tendency to **overfit**.

Pruning

Pruning* consists in **removing sections of the tree that are non-critical and redundant** from a decision tree.

Pruning can either be:

- **Pre-pruning**: set from the beginning a stop criterion that limits the tree depth or information metric.

Pruning to avoid overfitting

Decision trees have the tendency to **overfit**.

Pruning

Pruning* consists in **removing sections of the tree that are non-critical and redundant** from a decision tree.

Pruning can either be:

- **Pre-pruning**: set from the beginning a stop criterion that limits the tree depth or information metric.
- **Post-pruning**: after building the tree, remove some of the nodes and splits that do not carry too much information.

Pruning to avoid overfitting

Many existing possible methods.
Simplest one: **reduced error pruning**

Reduced error pruning

Reduced error pruning* is a pruning algorithm that consists in replacing in a bottom-up fashion in removing each test. If the accuracy is the same, then remove test.

Hyperparameters

Hyperparameters

Question

What are the hyperparameters of classification trees ?

Hyperparameters

Question

What are the hyperparameters of classification trees ?

- Stop criterion
- Information criterion: entropy, Gini ...

Advantages and drawbacks

Advantages and drawbacks

Advantages:

Advantages and drawbacks

Advantages:

- Interpretation is very simple
- Using categorical data is straight-forward
- Easy feature reduction
- Do not need expensive computation after initial computation

Advantages and drawbacks

Advantages:

- Interpretation is very simple
- Using categorical data is straight-forward
- Easy feature reduction
- Do not need expensive computation after initial computation

Drawbacks:

Advantages and drawbacks

Advantages:

- Interpretation is very simple
- Using categorical data is straight-forward
- Easy feature reduction
- Do not need expensive computation after initial computation

Drawbacks:

- Tendency to overfit !
- Training can take a long time (especially with quantitative variables)

Random forests

Random forests

Because of this tendency to overfit, trees are usually used together as **forests** as a **bagging*** algorithm called **random forest**.

Bagging (bootstrap aggregated)

Bagging consists in resampling the training data with replacement, building several models on this resampled data, and voting the algorithms when performing a new prediction.

Random forests

In practice, bagging consists in performing n times:

- Resampling the dataset (using replacement or not)
- Building a tree on this dataset

Random forests

In practice, bagging consists in performing n times:

- Resampling the dataset (using replacement or not)
- Building a tree on this dataset

When a new sample comes in, ask the n different trees what class to assign the trees to (possibly by weighting the answer depending on trees performance).

Questions

Questions ?