Decision Tree

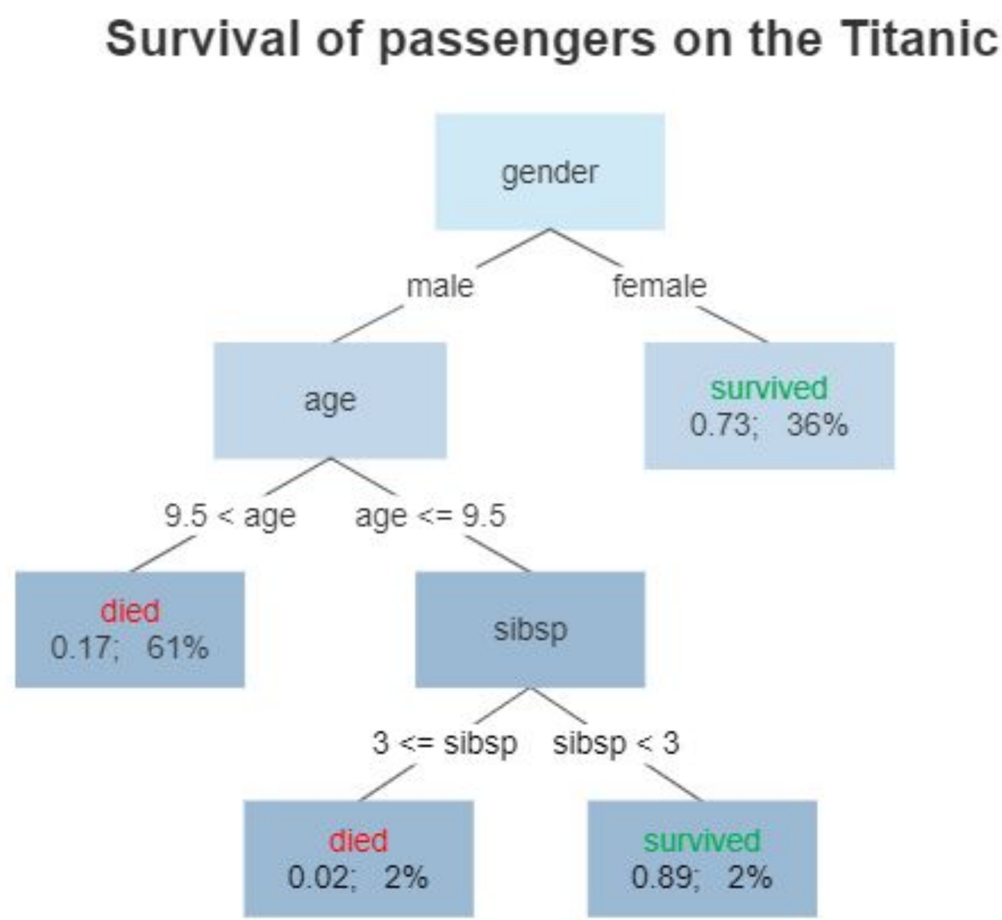
the excuse

marco milanesio
MScDSAI

UNIVERSITÉ CÔTE D'AZUR

Decision Tree

- Supervised learning
  - Classification
  - Regression
- Leaves = class labels
- Many algorithms
  - ID3
  - CART
  - FDT
  - ...



source: wikipedia

Key idea

- Split the dataset
- Find the "best" split

- Given training vectors: $x_i \in R^n, i = 1,\ldots,l$
- and a label vector: $y \in R^l$
- Recursively partitions the feature space s.t. samples with the same labels are grouped together

Maths

- data at node m : Q_m with n_m samples
- For each candidate split $s = (i, t_m)$ consisting of (feature, threshold)
- Do the partition

$$Q_m^L(s) = \{(x, y) \,|\, x_i \leq t_m\}$$

$$Q_m^R(s) = Q_m \setminus Q_m^L(s)$$

Impurity

$$Q_m^L(s) = \{(x, y) \,|\, x_i \le t_m\}$$

$$Q_m^R(s) = Q_m \setminus Q_m^L(s)$$

$$G(Q_m, s) = \frac{n_m^L}{n_m} G(Q_m^L(s)) + \frac{n_m^R}{n_m} G(Q_m^R(s))$$

Impurity function

The goal

- Select the parameters that minimize the impurity

$$s* = argmin_s G(Q_m, s)$$

- Recurse until max_depth or n_m = 1

Gini impurity

- measures how often a random element would be incorrectly labeled if it were labeled randomly
- J classes
- relative frequencies p_i, $i \in 1,..,J$
- p_i is the probability of choosing an item of label i
- Prob of miscategorizing $\sum_{k \neq i} p_k = 1 - p_i$

Gini impurity

$$I_G(p) = \sum_{i=1}^{J} \left(p_i \sum_{k \neq i} p_k \right)$$

$$I_G(p) = \sum_{i=1}^{J} p_i (1 - p_i) = 1 - \sum_{i=1}^{J} p_i^2$$

An example

x : [1,2,3,6,7,8]

y : [0,0,0,1,1,1]

Total n = 6.
Threshold = 3

Left (<= 3): values [1,2,3], y_left = [0,0,0]

Counts: class0: 3, class1 = 0

p0 = 3/3 = 1;   p1 = 0/3 = 0

$G(L) = 1 - (1^2 + 0^2) = 1 - 1 = 0$

Right (> 3): values [6,7,8], y_right = [1,1,1]

Counts: class0: 0, class1 = 3

p0 = 0/3 = 1;   p1 = 3/3 = 1

$G(R) = 1 - (0^2 + 1^2) = 1 - 1 = 0$

GINI = (3/6) * 0 + (3/6) * 0 = 0

An example

x : [1,2,3,6,7,8]

y : [0,0,0,1,1,1]

Total n = 6.
Threshold = 2

Left (<= 2): values [1,2], y_left = [0,0]

Counts: class0: 2, class1 = 0

p0 = 2/2 = 1;   p1 = 0/2 = 0

$G(L) = 1 - (1^2 + 0^2) = 1 - 1 = 0$

Right (> 2): values [3,6,7,8], y_right = [0,1,1,1]

Counts: class0: 1, class1 = 3

p0 = 1/4;   p1 = 3/4

$G(R) = 1 - (1/4^2 + 3/4^2) = 1 - 1/16 - 9/16 = 6/16 = 3/8$

GINI = (2/6) * 0 + (4/6) * 3/8 = 1/4 = 0.25

Spelled out

- for each feature index:
    - compute thresholds (use unique values)
    - for each threshold:
        - split the dataset into L and R
        - compute gini factor
        - if best:
            - update
- return solution

EXAM

- Implement your own best_split algorithm
- 1 script "gini.py" with 3 functions (1 given)

```
def gini_inpurity(y)     # given

def split_dataset(X, y, feature_index, threshold)

def best_split(X, y)
```

EXAM

- 1 script "main.py" is given. Use it and complete it.
- [OPTIONAL1] plot the history of the gini factor computation
- [OPTIONAL2]: implement optimization strategy (next slide)
- DON'T DO OPTIONALS BEFORE COMPLETING
- DON'T DO OPTIONAL2 BEFORE OPTIONAL1

EXAM: OPTIONAL 2

- Sort feature values and consider splits only between distinct consecutive values (and only when the class label actually changes across that boundary).
- Set threshold to the midpoint.

EXAM

```
def split_dataset(X, y, feature_index, threshold):
    """
    Splits dataset into left/right based on threshold
    param X: np.array / list
    param y: np.array / list
    feature_index: np.array / list
    threshold: numerical

    return tuple
    """
```

EXAM

```
def best_split(X, y):
    """
    Find the best split for dataset
    param X: np.array / list
    param y: np.array / list

    return tuple
    """
```

EXAM

- Return:
  - 1 python script "yourname_gini.py" containing the main algorithm
  - 1 python script "yourname_main.py"
- Zip them (yourname_exam.zip) and send email
- Deadline: <u>**TODAY AT 12:00 PM**</u>.
- EMAIL TIMESTAMP will rule out late comings
  - <insert trivia about pasts courses here>