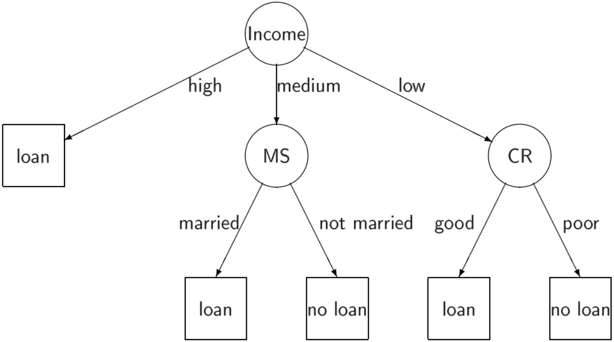
Decision trees

Victor Kitov

Table of Contents

1. Definition of decision tree
2. Splitting rules
3. Splitting rule selection
4. Prediction assignment to leaves
5. Termination criterion

Example of decision tree



Definition of decision tree

- Prediction is performed by tree T:
  - directed graph
  - without loops
  - with single root node

Definition of decision tree

- for each internal node t a check-function $Q_t(x)$ is associated
- for each edge $r_t(1),...r_t(K_t)$ a set of values of check-function $Q_t(x)$ is associated: $S_t(1),...S_t(K_t)$ such that:
  - $\bigcup_k S_t(k) = range[Q_t]$
  - $S_t(i) \cap S_t(j) = \emptyset \ \forall i \neq j$

Prediction process

- a set of nodes is divided into:
    - internal nodes $int(T)$, each having ≥ 2 child nodes
    - terminal nodes $terminal(T)$, which do not have child nodes but have associated prediction values.

Prediction process

- a set of nodes is divided into:
  - internal nodes $int(T)$, each having ≥ 2 child nodes
  - terminal nodes $terminal(T)$, which do not have child nodes but have associated prediction values.
- Prediction process for tree T:
  - $t = root(T)$
  - while t is not a leaf node:
    - calculate $Q_t(x)$
    - determine j such that $Q_t(x) \in S_t(j)$
    - follow edge $r_t(j)$ to j-th child node: $t = \tilde{t}_j$
  - return prediction, associated with leaf t.

Specification of decision tree

- To define a decision tree one needs to specify:
    - the check-function: $Q_t(x)$
    - the splitting criterion: K_t and $S_t(1), ...S_t(K_t)$
    - the termination criteria (when node is defined as a terminal node)
    - the predicted value for each leaf node.

Table of Contents

1. Definition of decision tree
2. Splitting rules
3. Splitting rule selection
4. Prediction assignment to leaves
5. Termination criterion

Possible definitions of splitting rules

- $Q_t(x) = x^{i(t)}$, where $S_t(j) = v_j$, where $v_1, ...v_K$ are unique values of feature $x^{i(t)}$.
- $S_t(1) = \{x^{i(t)} \le h_t\}$, $S_t(2) = \{x^{i(t)} > h_t\}$
- $S_t(j) = \{h_j < x^{i(t)} \le h_{j+1}\}$ for set of partitioning thresholds $h_1, h_2, ...h_{K_t+1}$.
- $S_t(1) = \{x : \langle x, v\rangle \le 0\}, \quad S_t(2) = \{x : \langle x, v\rangle > 0\}$
- $S_t(1) = \{x : \|x\| \le h\}, \quad S_t(2) = \{x : \|x\| > h\}$
- etc.

Most famous decision tree algorithms

- CART (classification and regression trees)
    - implemented in scikit-learn
- C4.5

CART version of splitting rule

- single feature value is considered:

$$Q_t(x) = x^{i(t)}$$

- binary splits:

$$K_t = 2$$

- split based on threshold h_t:

$$S_1 = \{x^{i(t)} \le h_t\},\ S_2 = \{x^{i(t)} > h_t\}$$

- $h(t) \in \{x_1^{i(t)}, x_2^{i(t)}, ...x_N^{i(t)}\}$
    - applicable only for real, ordinal and binary features
    - discrete unordered features:

CART version of splitting rule

- single feature value is considered:

$$Q_t(x) = x^{i(t)}$$

- binary splits:

$$K_t = 2$$

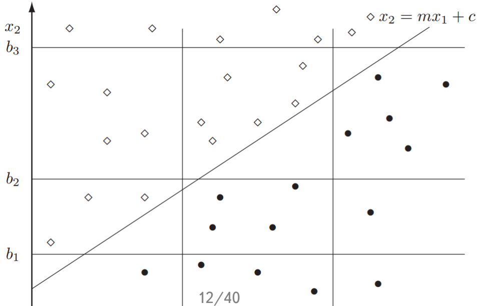
- split based on threshold h_t:

$$S_1 = \{x^{i(t)} \le h_t\},\ S_2 = \{x^{i(t)} > h_t\}$$

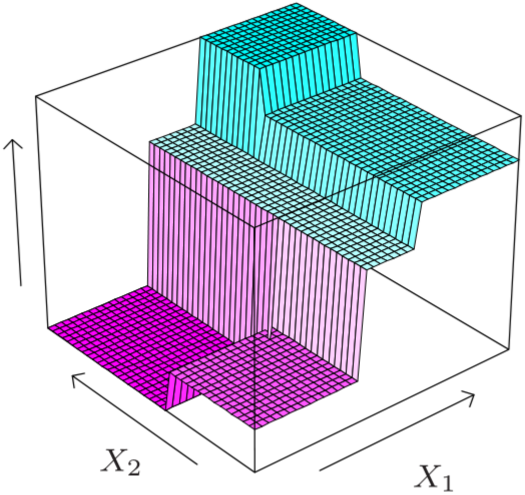
- $h(t) \in \{x_1^{i(t)}, x_2^{i(t)}, ...x_N^{i(t)}\}$
  - applicable only for real, ordinal and binary features
  - discrete unordered features:may use one-hot encoding.

Analysis of CART splitting rule

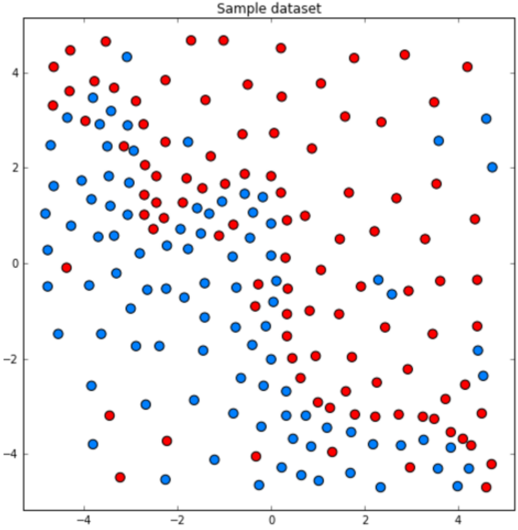
- Advantages:
  - simplicity
  - estimation efficiency
  - interpretability
- Drawbacks:
  - many nodes may be needed to describe boundaries not parallel to axes:



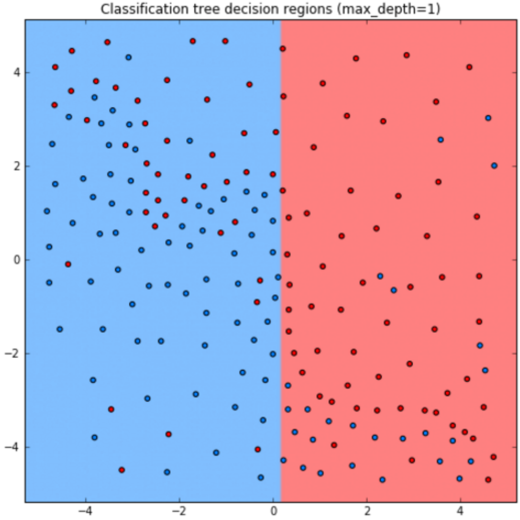
Piecewise constant predictions of decision trees



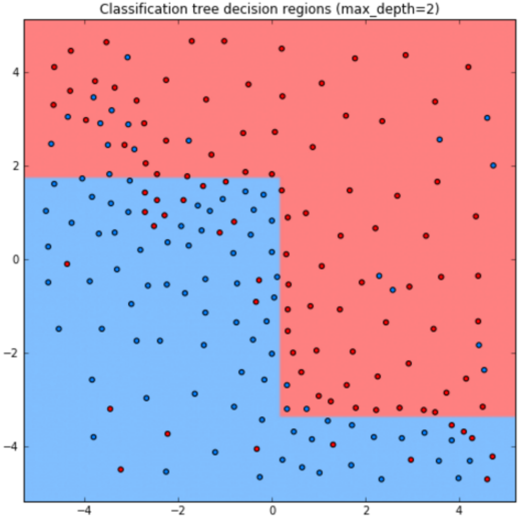
Sample dataset



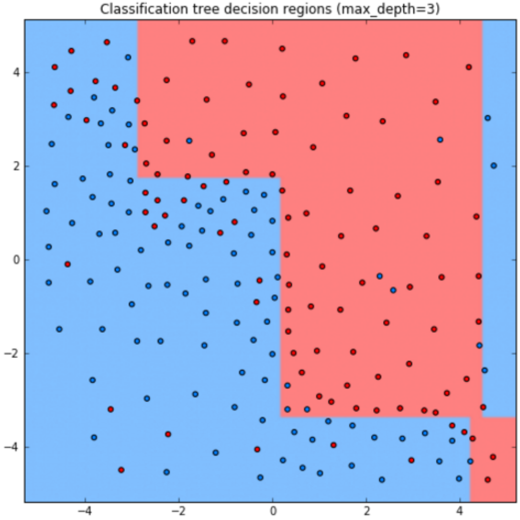
Example: Decision tree classification



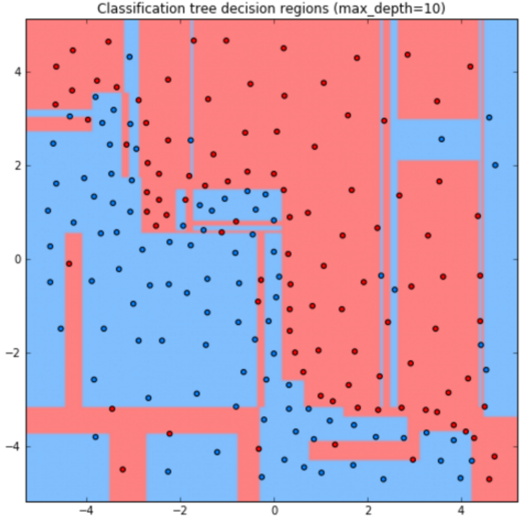
Example: Decision tree classification



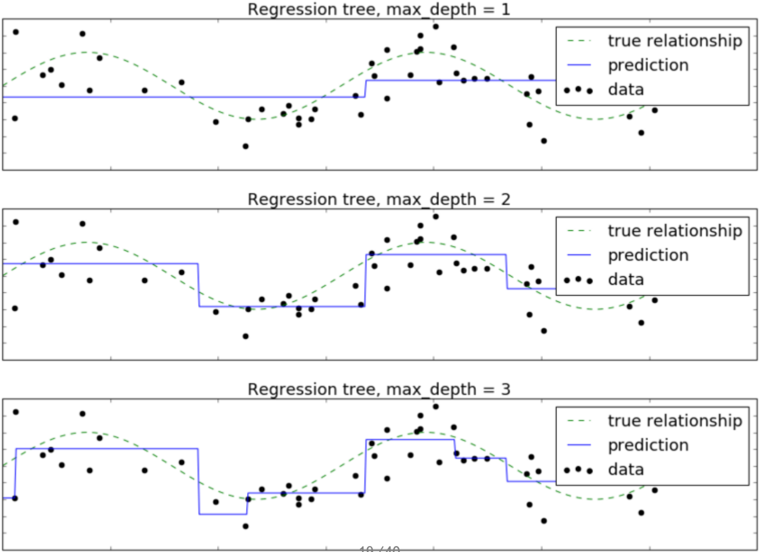
Example: Decision tree classification



Example: Decision tree classification



Example: Regression tree



Example: Regression tree

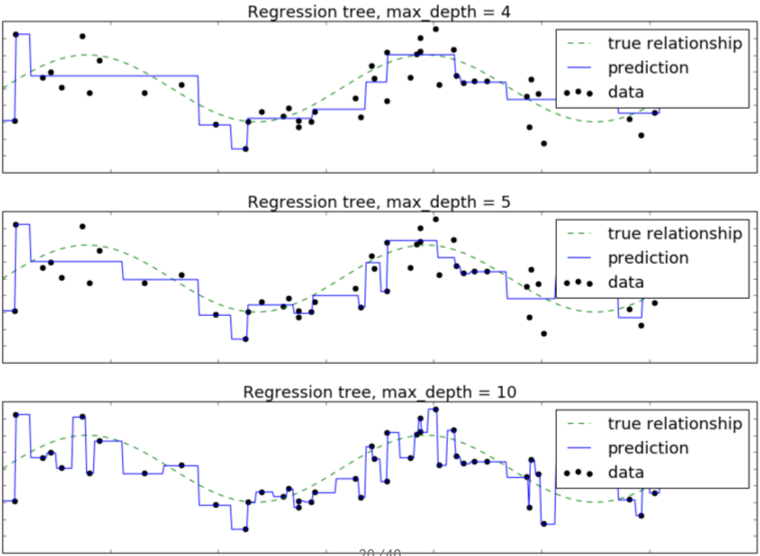


Table of Contents

1. Definition of decision tree
2. Splitting rules
3. Splitting rule selection
4. Prediction assignment to leaves
5. Termination criterion

Impurity function

- Impurity function $\phi(t) = \phi(p(\omega_1|t), ...p(\omega_C|t))$ measures the mixture of classes using class probabilities inside node t.
- It can be any function $\phi(q_1, q_2, ...q_C)$ with the following properties:
  - ϕ is defined for $q_j \geq 0$ and $\sum_j q_j = 1$.
  - ϕ attains maximum for $q_j = 1/C$, $k = 1, 2, ...C$.
  - ϕ attains minimum when $\exists j : q_j = 1$, $q_i = 0 \ \forall i \neq j$.
  - ϕ is symmetric function of $q_1, q_2, ...q_C$.
- Note: in regression $\phi(t)$ measures the spread of y inside node t.
  - may be MSE, MAE.

Typical impurity functions

- **Gini criterion**
  - interpretation: probability to make mistake when predicting class randomly with class probabilities $[p(\omega_1|t), ...p(\omega_C|t)]$:

$$I(t) = \sum_i p(\omega_i|t)(1 - p(\omega_i|t)) = 1 - \sum_i [p(\omega_i|t)]^2$$

- **Entropy**
  - interpretation: measure of uncertainty of random variable

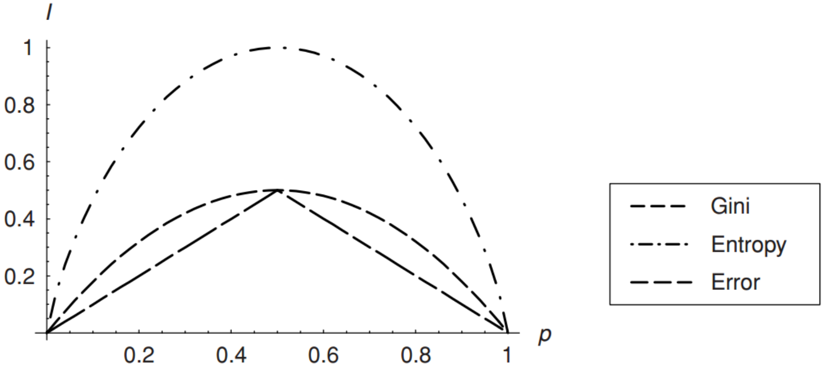
$$I(t) = -\sum_i p(\omega_i|t) \ln p(\omega_i|t)$$

- **Classification error**
  - interpretation: frequency of errors when classifying with the most common class

$$I(t) = 1 - \max_i p(\omega_i|t)$$

Typical impurity functions

Impurity functions for binary classification with class probabilities $p = p(\omega_1|t)$ and $1 - p = p(\omega_2|t)$.



Splitting criterion selection

$$\Delta I(t) = I(t) - \sum_{i=1}^{R} I(t_i) \frac{N(t_i)}{N(t)}$$

- $\Delta I(t)$ is the quality of the split[1] of node t into child nodes $t_1, ...t_R$.

[1] If $I(t)$ is entropy, then $\Delta I(t)$ is called *information gain*.

Splitting criterion selection

$$\Delta I(t) = I(t) - \sum_{i=1}^{R} I(t_i)\frac{N(t_i)}{N(t)}$$

- $\Delta I(t)$ is the quality of the split[1] of node t into child nodes $t_1, ...t_R$.
- CART selection: select feature i_t and threshold h_t, which maximize $\Delta I(t)$:

$$i_t,\, h_t = \arg\max_{k,h} \Delta I(t)$$

- CART decision making: from node t follow:

$$\begin{cases} \text{left child } t_1, & \text{if } x^{i_t} \le h_t \\ \text{right child } t_2, & \text{if } x^{i_t} > h_t \end{cases}$$

[1] If $I(t)$ is entropy, then $\Delta I(t)$ is called *information gain*.

Table of Contents

1. Definition of decision tree
2. Splitting rules
3. Splitting rule selection
4. Prediction assignment to leaves
5. Termination criterion

Regression: prediction assignment for leaf nodes[2]

- Define $I_t = \{i : x_i \in \text{node}\, t\}$
- For mean squared error loss (MSE):

$$\widehat{y} = \arg\min_{\mu} \sum_{i \in I_t} (y_i - \mu)^2 = \frac{1}{|I_t|} \sum_{i \in I_t} y_i,$$

- For mean absolute error loss (MAE):

$$\widehat{y} = \arg\min_{\mu} \sum_{i \in I_t} |y - \mu| = median\{y_i : i \in I_t\}.$$

[2] Prove optimality of estimators for MSE and MAE loss.

Classification: prediction assignment for leaf nodes

- Define $\lambda(\omega_i, \omega_j)$ - the cost of predicting object of class ω_i as belonging to class ω_j.
    - Minimum loss class assignment:

Classification: prediction assignment for leaf nodes

- Define $\lambda(\omega_i, \omega_j)$ - the cost of predicting object of class ω_i as belonging to class ω_j.
    - Minimum loss class assignment:

$$c = \arg\min_{\omega} \sum_{i \in I_t} \lambda(c_i, \omega)$$

    - For $\lambda(\omega_i, \omega_j) = \mathbb{I}[\omega_i \neq \omega_j]$:

Classification: prediction assignment for leaf nodes

- Define $\lambda(\omega_i, \omega_j)$ - the cost of predicting object of class ω_i as belonging to class ω_j.
    - Minimum loss class assignment:

$$c = \arg\min_{\omega} \sum_{i \in I_t} \lambda(c_i, \omega)$$

    - For $\lambda(\omega_i, \omega_j) = \mathbb{I}[\omega_i \neq \omega_j]$:most common class will be associated with the leaf node:

$$c = \arg\max_{\omega} \left|\{i : \, i \in I_t, \, y_i = \omega\}\right|$$

Table of Contents

1. Definition of decision tree
2. Splitting rules
3. Splitting rule selection
4. Prediction assignment to leaves
5. Termination criterion
    - Rule based termination
    - CART pruning algorithm

Termination criterion

- Bias-variance tradeoff:
    - very large complex trees -> overfitting
    - very short simple trees -> underfitting
- Approaches to stopping:
    - rule-based
    - based on pruning

5. Termination criterion
   - Rule based termination
   - CART pruning algorithm

Rule-base termination criteria

- Rule-based: a criterion is compared with a threshold.
- Variants of criterion:
    - depth of tree
    - number of objects in a node
    - minimal number of objects in one of the child nodes
    - impurity of classes
    - change of impurity of classes after the split

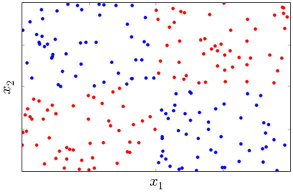
Analysis of rule-based termination

Advantages:

- simplicity
- interpretability

Disadvantages:

- specification of threshold is needed
- impurity change is suboptimal: further splits may become better than current one
    - example:



5. Termination criterion
    - Rule based termination
    - CART pruning algorithm

CART[3]

- General idea: build tree up to pure nodes and then prune.
- Define:
    - T be some subtree of out tree
    - T_t full subtree with root at node t
    - $\tilde{T}$ be a set of leaf nodes of tree T
    - $M(t)$ - the number of mistakes inside node t of the tree on the training set.
- Also define

error-rate loss : $R(T) = \sum_{t \in \tilde{T}} R(t)$

complexity+error-rate loss: $R_\alpha(T) = \sum_{t \in \tilde{T}} R_\alpha(t) = R(T) + \alpha|\tilde{T}|$

- Condition when $R_{\alpha_t}(T_t) = R_{\alpha_t}(t)$:

$$\alpha_t = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1}$$

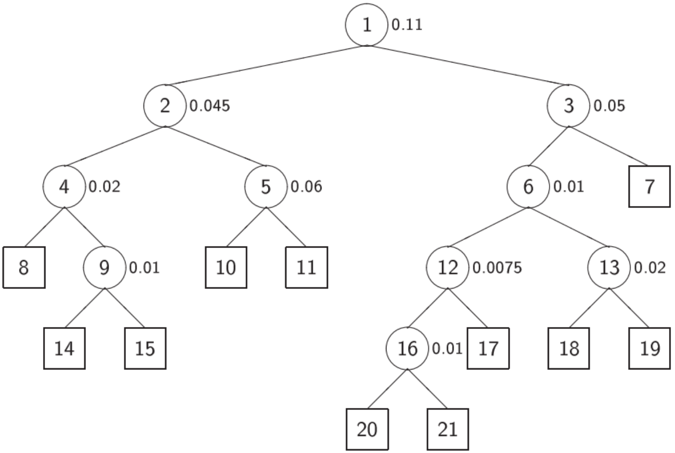
[3] Simple pruning based on validation set.

Pruning algorithm

1. Build tree until each node contains representatives of only single class - obtain tree T.
2. Build a sequence of nested trees $T = T_0 \supset T_1 \supset ... \supset T_{|T|}$ containing $|T|$, $|T| - 1$,...1 nodes, repeating the procedure:
   - replace the tree T_t with smallest α_t with its root t
   - recalculate α_t for all ancestors of t.
3. For trees $T_0, T_1, ...T_{|T|}$ calculate their validation set error-rates $R(T_0), R(T_1), ...R(T_{|T|})$.
4. Select T_i, giving minimum error-rate on the validation set:

$$i = \arg\min_i R(T_i)$$

Example


1 0.11
2 0.045
3 0.05
4 0.02
5 0.06
6 0.01
7
8
9 0.01
10
11
12 0.0075
13 0.02
14
15
16 0.01
17
18
19
20
21

Example

Logs of the performance metrics of the pruning process:

step num.	α_k	$\lvert\tilde{T}^k\rvert$	$R(T^k)$
1	0	11	0.185
2	0.0075	9	0.2
3	0.01	6	0.22
4	0.02	5	0.25
5	0.045	3	0.34
6	005	2	0.39
7	0.11	1	0.5

Handling missing values

If checked feature is missing:

- fill missing values:
  - with feature mean
  - with new categorical value "missing" (for categorical values)
  - predict them using other known features
- CART uses prediction of unknown feature using another feature that best predicts the missing one: "surrogate split" - technique
- ID3 and C4.5 decision trees use averaging of predictions made by each child node with weights $N(t_1)/N(t), N(t_2)/N(t), \ldots N(t_S)/N(t)$.

Analysis of decision trees

- Advantages:
  - simplicity
  - interpretability
  - implicit feature selection
  - naturally handles both discrete and real features
  - prediction is invariant to monotone transformations of features for $Q_t(x) = x^{i(t)}$
    - work well for features of different nature
- Disadvantages:
  - non-parallel to axes class separating boundary may lead to many nodes in the tree for $Q_t(x) = x^{i(t)}$
  - one step ahead lookup strategy for split selection may be insufficient (XOR example)
  - not online - slight modification of the training set will require full tree reconstruction.