

# Decision trees

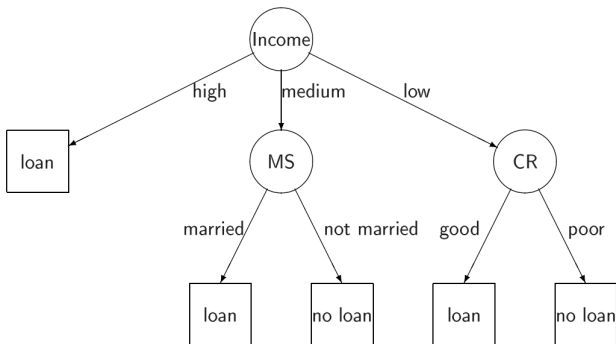
Victor Kitov

[v.v.kitov@yandex.ru](mailto:v.v.kitov@yandex.ru)

# 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  $r_t(1), \dots, r_t(K_t)$  a set of values of check-function  $Q_t(x)$  is associated:  $S_t(1), \dots, S_t(K_t)$  such that:
  - $\bigcup_k S_t(k) = \text{range}[Q_t]$
  - $S_t(i) \cap S_t(j) = \emptyset \ \forall i \neq j$

## Prediction process

- a set of nodes is divided into:
  - internal nodes  $int(T)$ , each having  $\geq 2$  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  $\geq 2$  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  $S_t(1), \dots, S_t(K_t)$
  - 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  $v_1, \dots, v_K$  are unique values of feature  $x^{i(t)}$ .
- $S_t(1) = \{x^{i(t)} \leq h_t\}$ ,  $S_t(2) = \{x^{i(t)} > h_t\}$
- $S_t(j) = \{h_j < x^{i(t)} \leq h_{j+1}\}$  for set of partitioning thresholds  $h_1, h_2, \dots, h_{K_t+1}$ .
- $S_t(1) = \{x : \langle x, v \rangle \leq 0\}$ ,  $S_t(2) = \{x : \langle x, v \rangle > 0\}$
- $S_t(1) = \{x : \|x\| \leq h\}$ ,  $S_t(2) = \{x : \|x\| > h\}$
- 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  $h_t$ :

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

- $h(t) \in \{x_1^{i(t)}, x_2^{i(t)}, \dots, x_N^{i(t)}\}$ 
  - 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:

$$K_t = 2$$

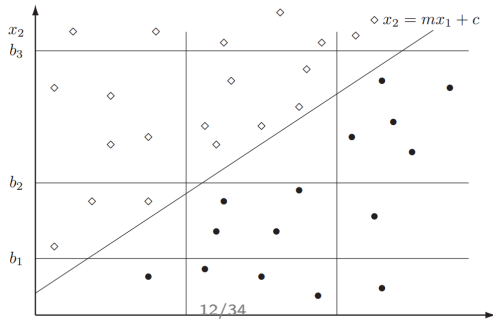
- split based on threshold  $h_t$ :

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

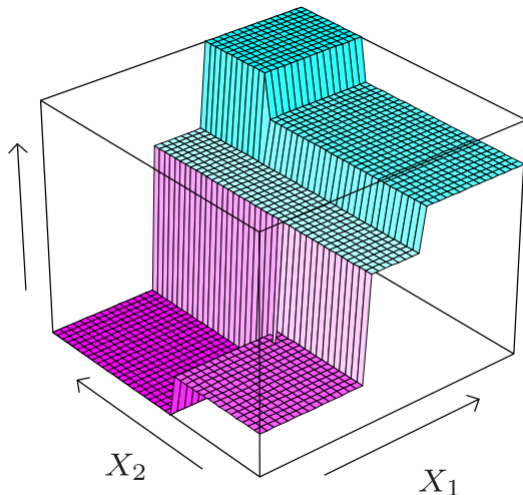
- $h(t) \in \{x_1^{i(t)}, x_2^{i(t)}, \dots, x_N^{i(t)}\}$ 
  - applicable only for real, ordinal and binary features
  - discrete unordered features: may use one-hot encoding.

# Analysis of CART splitting rule

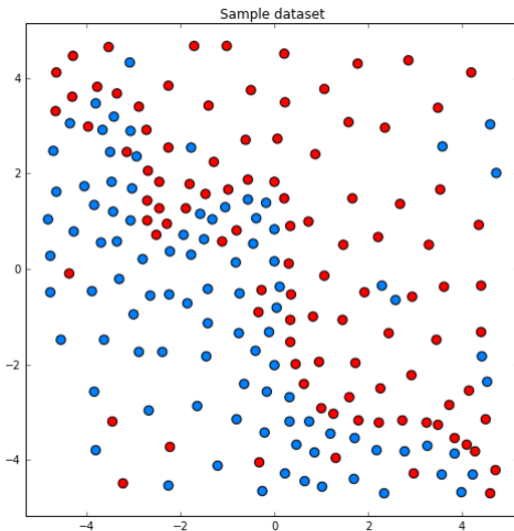
- Advantages:
  - simplicity
  - estimation efficiency
  - interpretability
- Drawbacks:
  - many nodes may be needed to describe boundaries not parallel to axes:



# Piecewise constant predictions of decision trees

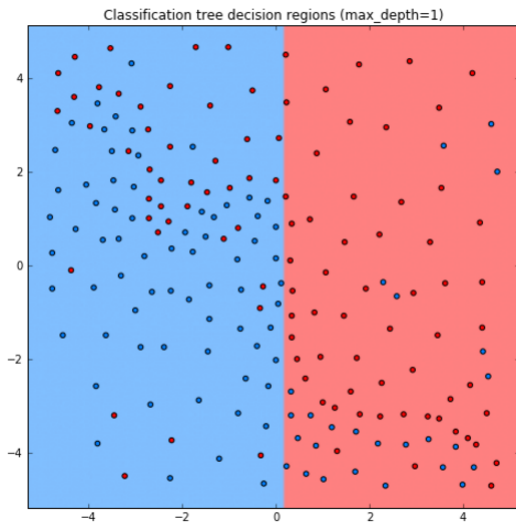


# Sample dataset

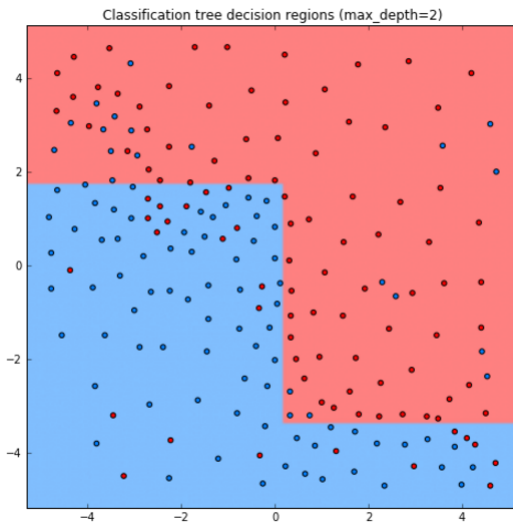




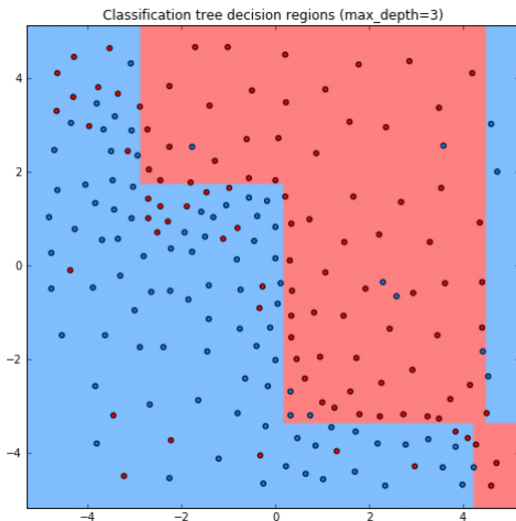
# Example: Decision tree classification



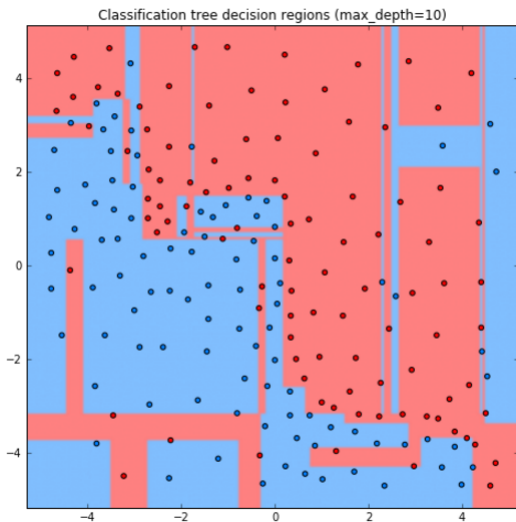
# Example: Decision tree classification



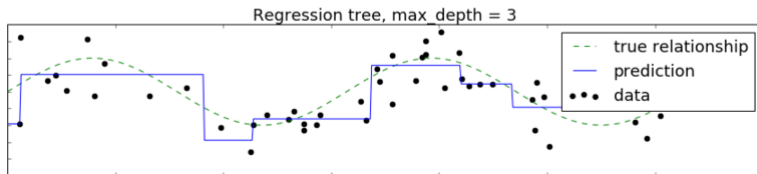
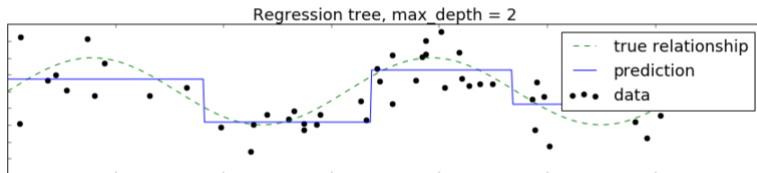
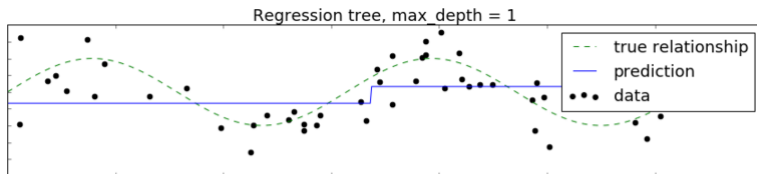
# Example: Decision tree classification



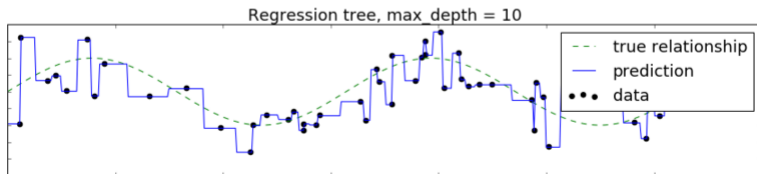
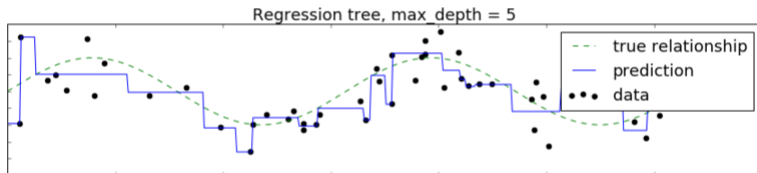
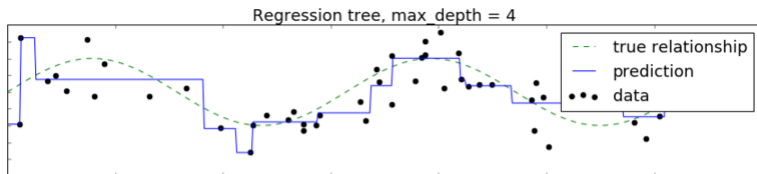
# Example: Decision tree classification



# 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 functions

- Impurity function measures uncertainty in  $y$  for objects falling inside node  $t$ .
- Regression:
  - let objects falling inside node  $t$  be  $I = \{i_1, \dots, i_K\}$ . We may define

$$\phi(t) = \frac{1}{K} \sum_{i \in I} (y_i - \mu)^2$$

$$\phi(t) = \frac{1}{K} \sum_{i \in I} |y_i - \mu|$$

where  $\mu = \frac{1}{K} \sum_{i \in I} y_i$ .



# Classification impurity functions

- For classification: let  $p_1, \dots, p_C$  be class probabilities for objects in node  $t$ .
- Then impurity function  $\phi(t) = \phi(p_1, p_2, \dots, p_C)$  should satisfy:
  - $\phi$  is defined for  $p_j \geq 0$  and  $\sum_j p_j = 1$ .
  - $\phi$  attains maximum for  $p_j = 1/C$ ,  $k = 1, 2, \dots, C$ .
  - $\phi$  attains minimum when  $\exists j : p_j = 1, p_i = 0 \forall i \neq j$ .
  - $\phi$  is symmetric function of  $p_1, p_2, \dots, p_C$ .

# Typical classification impurity functions

- **Gini criterion**

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

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

- **Entropy**

- interpretation: measure of uncertainty of random variable

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

- **Classification error**

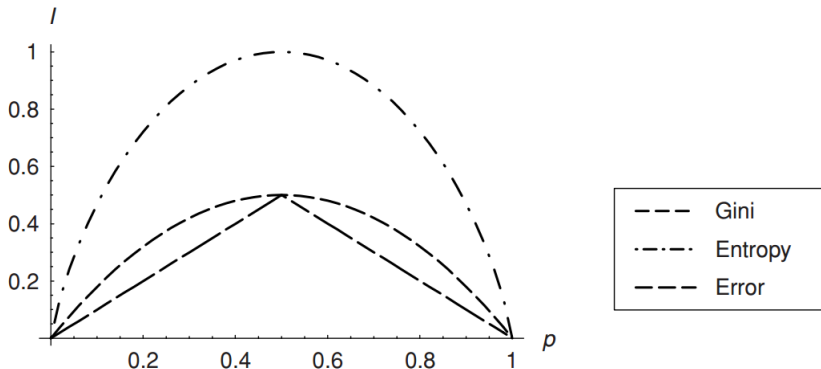
- interpretation: frequency of errors when classifying with the most common class

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

## Typical classification 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

- Define  $\Delta I(t)$  - is the quality of the split<sup>1</sup> of node  $t$  into child nodes  $t_1, \dots, t_R$ .

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

- CART optimization (regression, classification): 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:  
$$\begin{cases} \text{left child } t_1, & \text{if } x^{i_t} \leq h_t \\ \text{right child } t_2, & \text{if } x^{i_t} > h_t \end{cases}$$

---

<sup>1</sup>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

# Prediction assignment to leaves

- Regression:
  - mean (optimal for MSE loss)
  - median (optimal for MAE loss)
- Classification
  - most common class (optimal for constant misclassification cost)

# Table of Contents

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

# Termination criterion

- Bias-variance tradeoff:
  - very large complex trees -> overfitting
  - very short simple trees -> underfitting
- Approaches to stopping:
  - rule-based
  - based on pruning (not considered here)



## 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

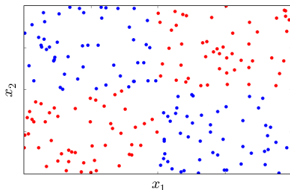
# Analysis of rule-based termination

Advantages:

- simplicity
- interpretability

Disadvantages:

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



# Tree feature importances

- Tree feature importances (`clf.feature_importances_` in sklearn).
  - Consider feature  $f$
  - Let  $T(f)$  be the set of all nodes, relying on feature  $f$  when making split.
  - efficiency of split at node  $t$ :  $\Delta I(t) = I(t) - \sum_{c \in \text{children}(t)} \frac{n_c}{n_t} I(c)$
  - feature importance of  $f$ :  $\sum_{t \in T(f)} n_t \Delta I(t)$
- Alternative: difference in decision tree prediction quality for
  - 1 original validation set
  - 2 validation set with  $j$ -th feature randomly shuffled

# Analysis of decision trees

- **Advantages:**

- simplicity of algorithm
- interpretability of model
- implicit feature selection
- good for features of different nature:
  - naturally handles both discrete and real features
  - prediction is invariant to monotone transformations of features for  $Q_t(x) = x^{i(t)}$

- **Disadvantages:**

- not very high accuracy:
  - high overfitting of tree structure up to top
  - 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.