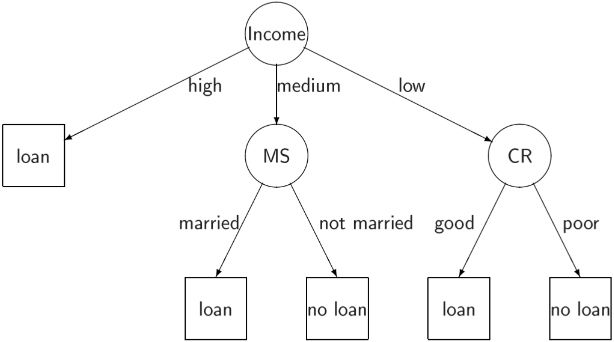
# Decision trees

Victor Kitov

Yandex School of Data Analysis

# Table of Contents

# 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 $r_1(t), ...r_{K(t)}(t)$ a set of values of check-function $Q_t(x)$ is associated: $S_1(t), ...S_{K(t)}(t)$ 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 $\geq 2$ 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 $S_j$ out of $S_1(t), ...S_{K(t)}(t)$, where $Q_t(x)$ belongs: $Q_t(x) \in S_j(t)$
    - follow edge $r_j(t)$ to child node $\tilde{t}_j : t = \tilde{t}_j$
  - 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

# CART version of splitting rule

- single feature value is considered:

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

- binary splits:

$$K(t) = 2$$

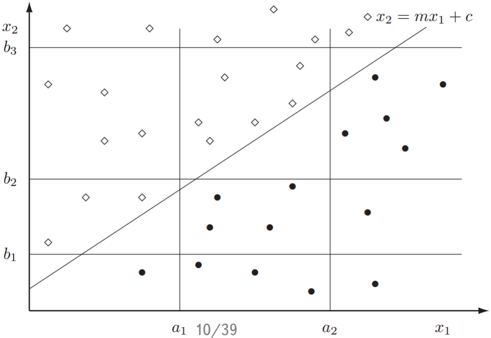
- split based on threshold:

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

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

# Analysis of CART splitting rule

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



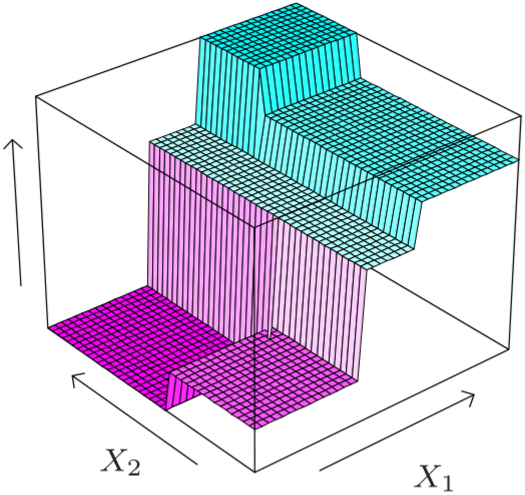
# Alternative definitions of splitting rules

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

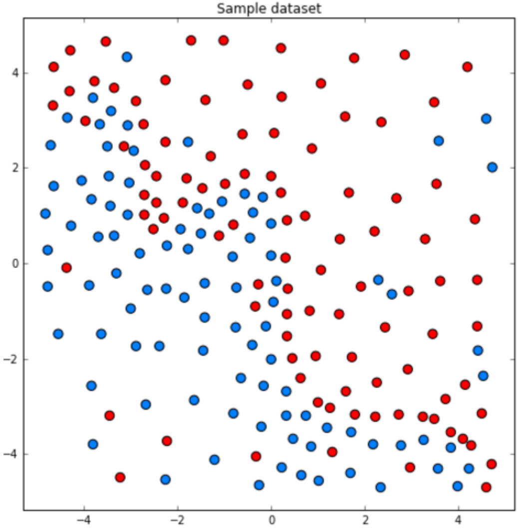
# Alternative definitions of splitting rules

- $S_t(i) = \{h_i < x^{k(t)} \le h_{i+1}\}$ for set of partitioning thresholds $h_1, h_2, ...h_{K+1}$.
- $S_t(1) = \{x : \langle x, v\rangle \le 0\}, \quad S_t(2) = \{x : \langle x, v\rangle > 0\}$
- $S_t(1) = \{x : \|x\| \le h\}, \quad S_t(2) = \{x : \|x\| > h\}$
- $Q_t(x) = x^{i(t)}$, where $S_t(j) = v_j$, where $v_1, ...v_K$ are unique values of feature $x^{i(t)}$.
- Properties:
    - may need much fewer nodes than binary splits by threshold
    - less interpretable

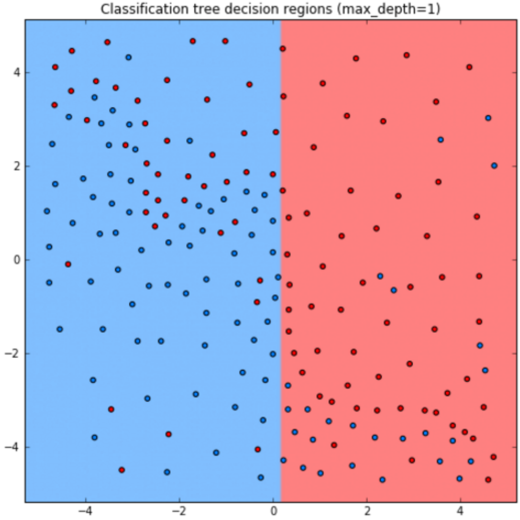
# Piecewise constant predictions of decision trees



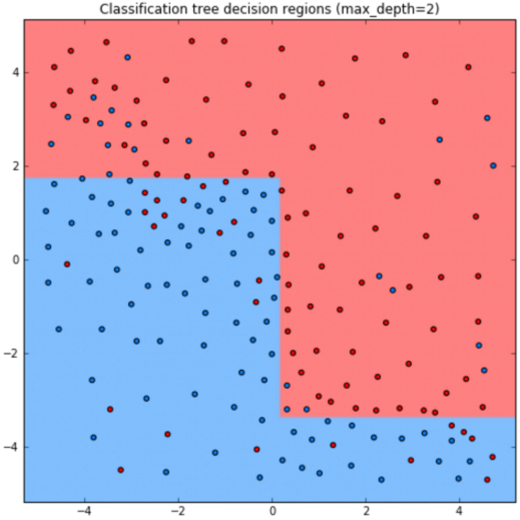
# Sample dataset



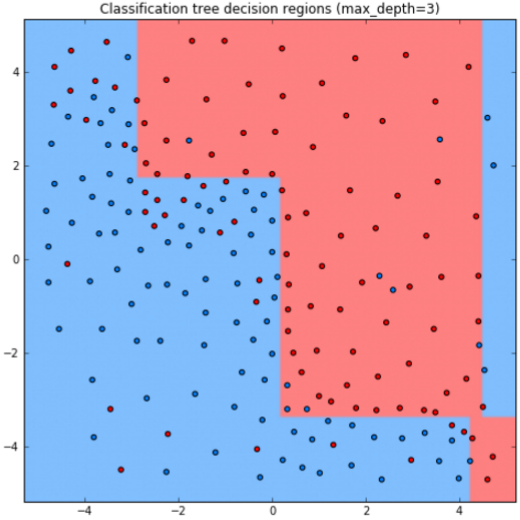
# Example: Decision tree classification



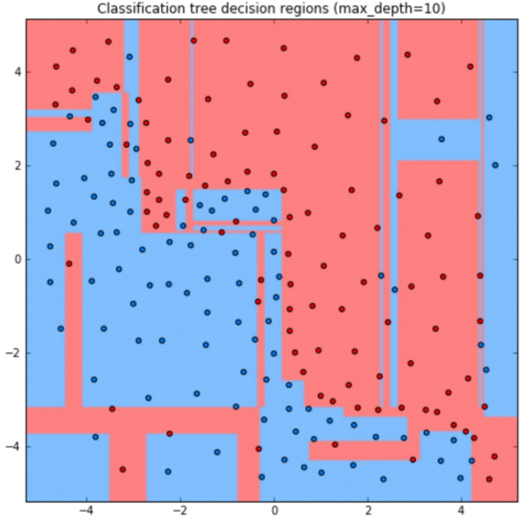
# Example: Decision tree classification



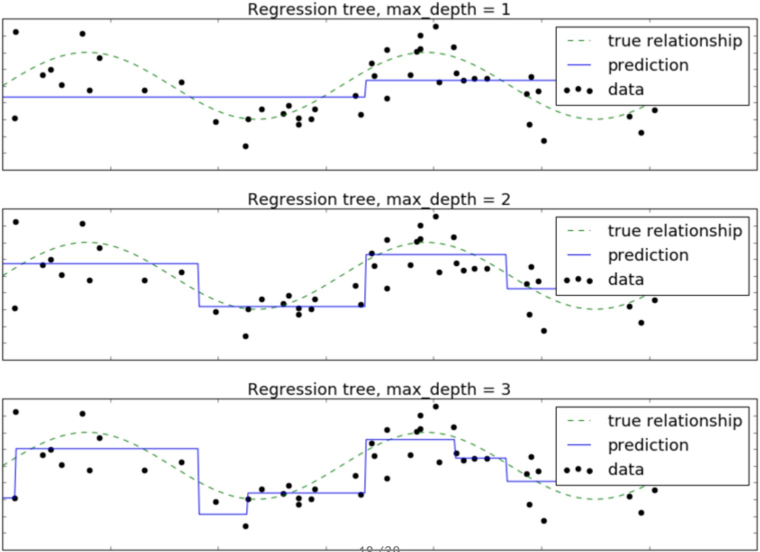
# Example: Decision tree classification



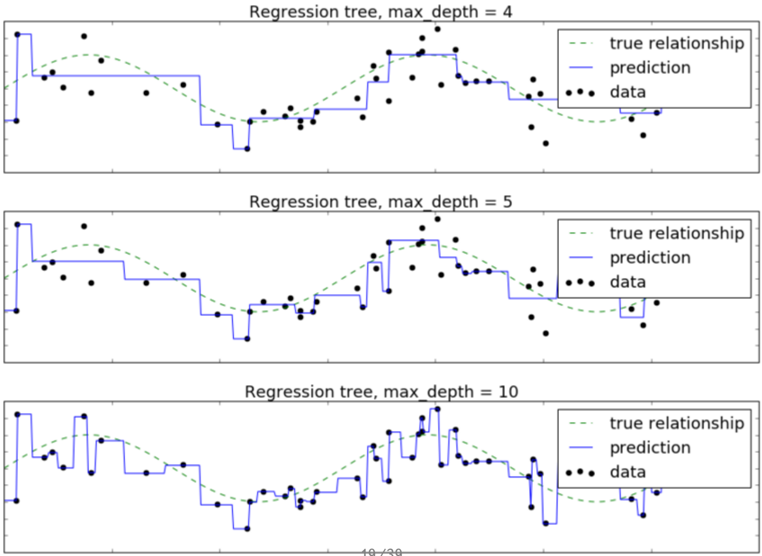
# Example: Decision tree classification



# Example: Regression tree



# Example: Regression tree



# Table of Contents

# Impurity function

- Let $t$ be any node and $u(t)$ - associated objects with node $t$,
- $N(t)$ - total number of objects and $N_j(t)$ - number of objects of class $j$ in $t$
- Probabilities of classes within node $t$:

$$p(\omega_j|x \in u(t)) = p(\omega_j|t) \approx \frac{N_j(t)}{N(t)}$$

- Impurity function $I(t) = \phi(p(\omega_1|t), ...p(\omega_C|t))$ has the following properties:
  - $\phi(q_1, q_2, ...q_C)$ is defined for $q_j \geq 0$ and $\sum_j q_j = 1$.
  - $\phi$ attains maximum for $q_j = 1/C$, $k = 1, 2, ...C$ .
  - $\phi$ attains minimum when $\exists j : q_j = 1, q_i = 0 \; \forall i \neq j$.
  - $\phi$ is symmetric function of $q_1, q_2, ...q_C$.

# Typical impurity functions

- **Gini criterion**
  - interpretation: probability to make mistake when classifying object 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

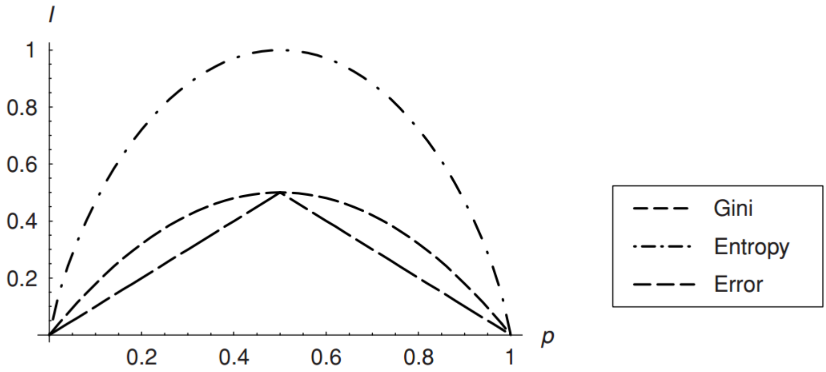
$$I(t) = -\sum_i p(\omega_i|t) \ln p(\omega_i|t)$$

- **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

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

- $\Delta I(t)$ is the quality of the split of node $t$ into child nodes $t_1, ... t_S$.
- If $I(t)$ is entropy, then $\Delta I(t)$ is called *information gain*.

# Splitting criterion selection

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

- $\Delta I(t)$ is the quality of the split of node $t$ into child nodes $t_1, ...t_S$.
- If $I(t)$ is entropy, then $\Delta I(t)$ is called *information gain*.
- CART selection: select feature $k(t)$ and threshold $h(t)$, which maximize $\Delta I(t)$:

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

- CART decision making: from node $t$ follow:
$$\begin{cases} \text{child } t_1, & \text{if } x^{k(t)} \geq h(t) \\ \text{child } t_2, & \text{if } x^{k(t)} < h(t) \end{cases}$$

# Table of Contents

# Prediction assignment for leaf nodes

- Define $I_t = \{i : x_i \in u(t)\}$, $N_t$ - number of elements in $I_t$.
- **Regression: quadratic loss** $(\widehat{y} - y)^2$:

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

- **Regression: abs. deviation loss** $|\widehat{y} - y|$ :

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

- **Classification with symmetrical costs**: most common class
  - solution for skewed costs $\lambda_{yf}$?

# Table of Contents

# 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

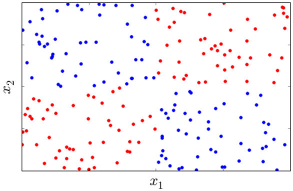
# Analysis of rule-based termination

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

- General idea: build tree up to pure nodes and then prune.
- Let $T$ be some subtree of out tree, $\tilde{T}$ be a set of leaf nodes of tree $T$.
- For each leaf $t \in \tilde{T}$ define $R(t)$ - the number of errors on the training set by this node.
- Also define

$$\begin{array}{ll} \text{error-rate loss :} & R(T) = \sum_{t \in \tilde{T}} R(t) \\ \text{complexity+error-rate loss:} & R_\alpha(T) = \sum_{t \in \tilde{T}} R_\alpha(t) = R(T) + \alpha|\tilde{T}| \end{array}$$

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

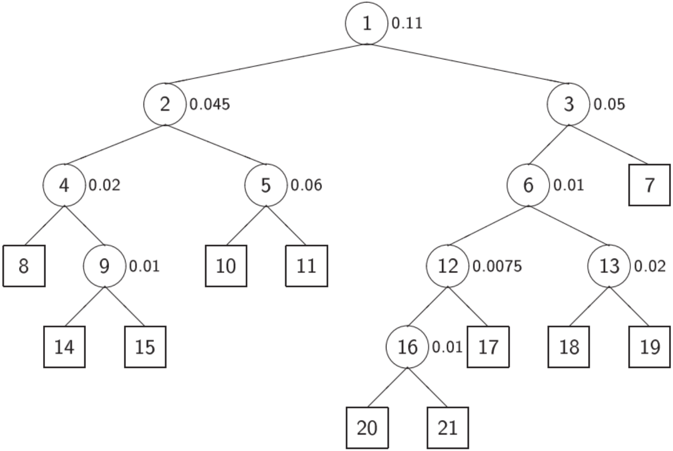
$$\alpha_t = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1}$$

# 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 $\alpha_t$ with its root t
    - recalculate $\alpha_t$ 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:

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

# 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. | $\alpha_k$ | $\lvert\tilde{T}^k\rvert$ | $R(T^k)$ |
|---|---|---|---|
| 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 |

# Table of Contents

# 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
- C4.5 uses averaging of predictions made by each child node with weights $N(t_1)/N(t)$, $N(t_2)/N(t)$, ... $N(t_S)/N(t)$.

# 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 $Q_t(x) = x^{i(t)}$
    - in particular, to normalization of features
- 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.