

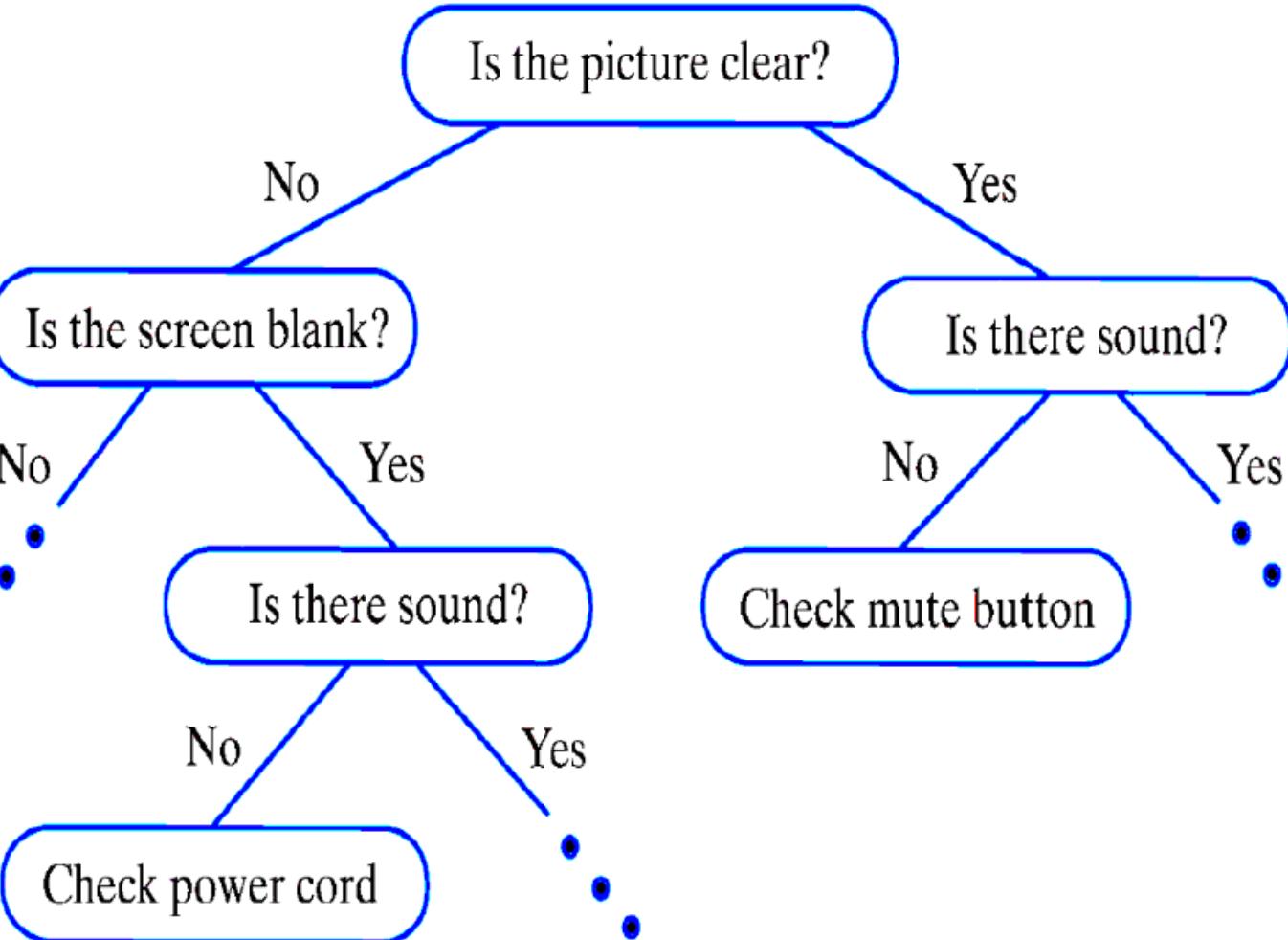
Лекция 8

Решающие деревья

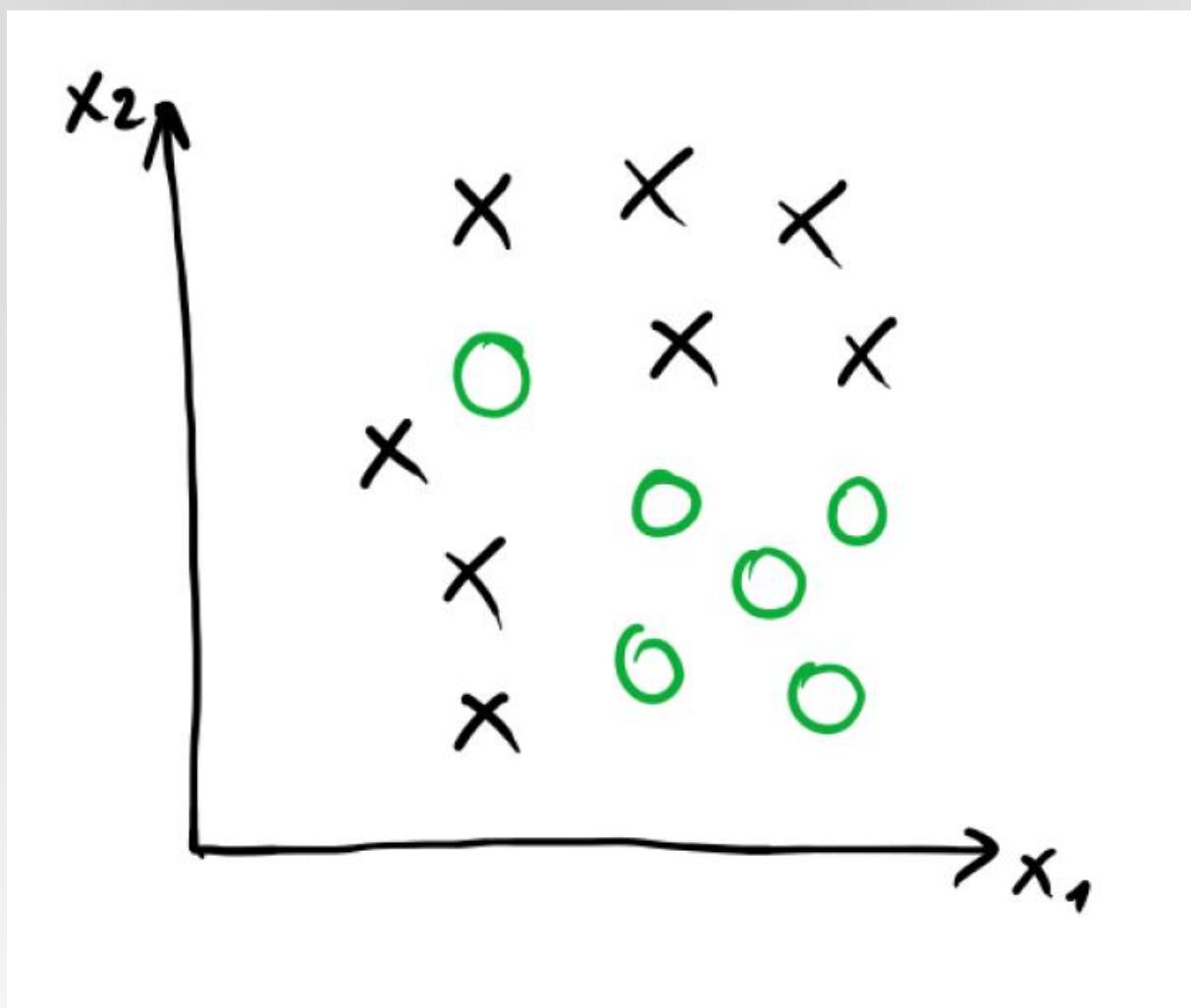
Дубенюк Анна Антоновна
anya.dubenyuk@yandex.ru
@andu192

ВШЭ, 2023

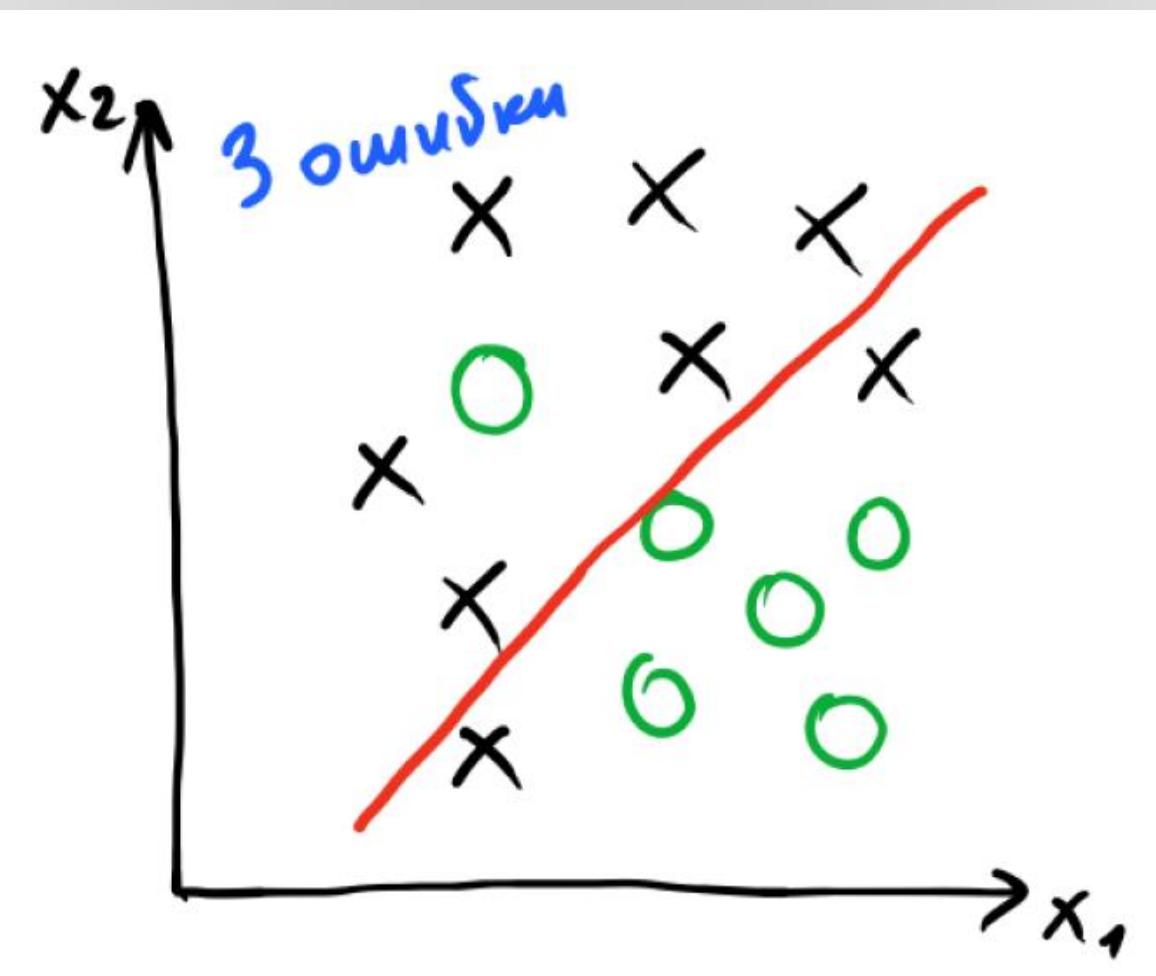
ПРИМЕР РЕШАЮЩЕГО ДЕРЕВА



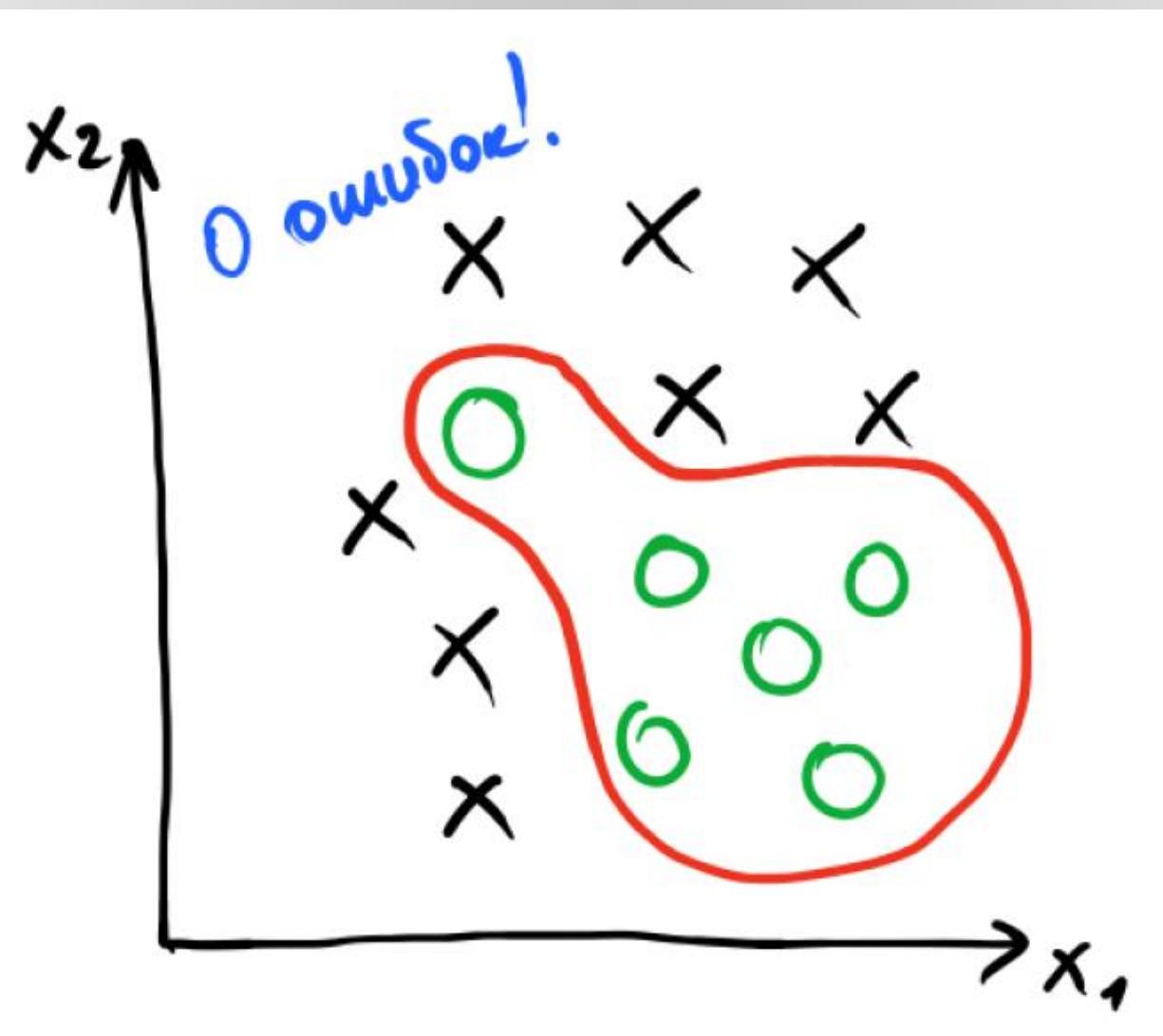
ПРИМЕР



ЛИНЕЙНАЯ МОДЕЛЬ



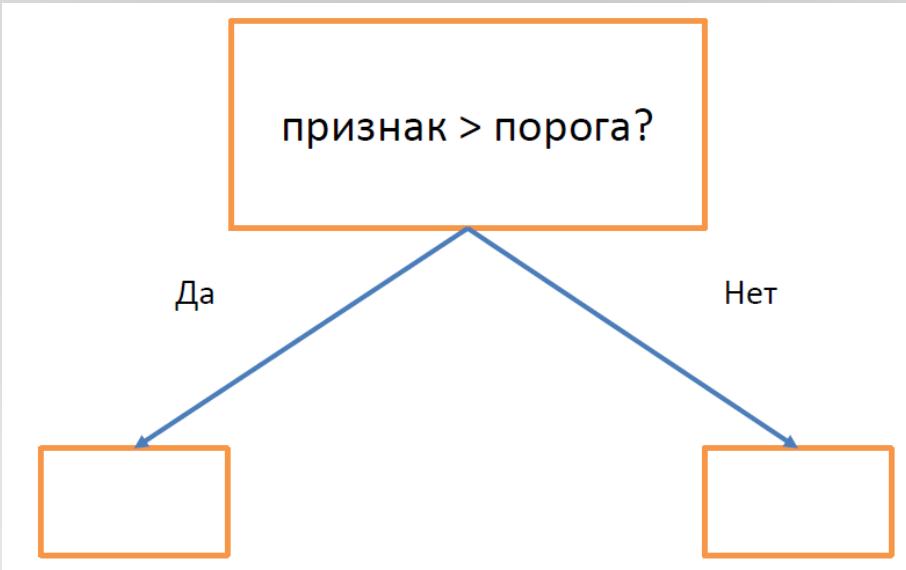
НЕЛИНЕЙНЫЙ АЛГОРИТМ



РЕШАЮЩЕЕ ДЕРЕВО

Решающее дерево – это бинарное дерево, в котором:

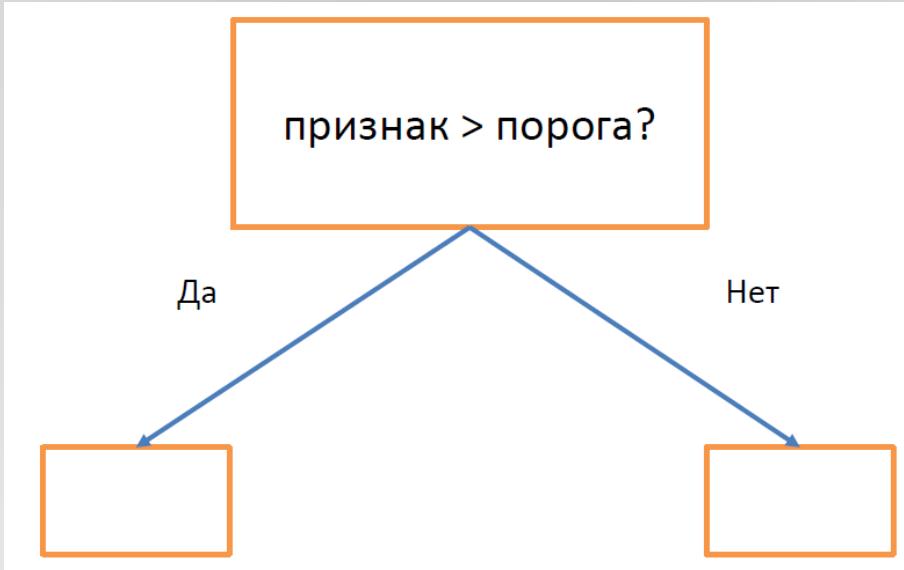
- 1) каждой вершине v приписана функция (предикат) $\beta_v: X \rightarrow \{0,1\}$



РЕШАЮЩЕЕ ДЕРЕВО

Решающее дерево – это бинарное дерево, в котором:

- 1) каждой вершине v приписана функция (предикат) $\beta_v: X \rightarrow \{0,1\}$



- 2) каждой листовой вершине v приписан прогноз $c_v \in Y$
(для классификации – класс или вероятность класса, для
регрессии – действительное значение целевой переменной)

ЖАДНЫЙ АЛГОРИТМ ПОСТРОЕНИЯ РЕШАЮЩЕГО ДЕРЕВА

1 шаг: найдем наилучшее разбиение всей выборки X на две части: $R_1(j, t) = \{x \mid x_j < t\}$ и $R_2(j, t) = \{x \mid x_j \geq t\}$ с точки зрения некоторого функционала $Q(X, j, t)$:

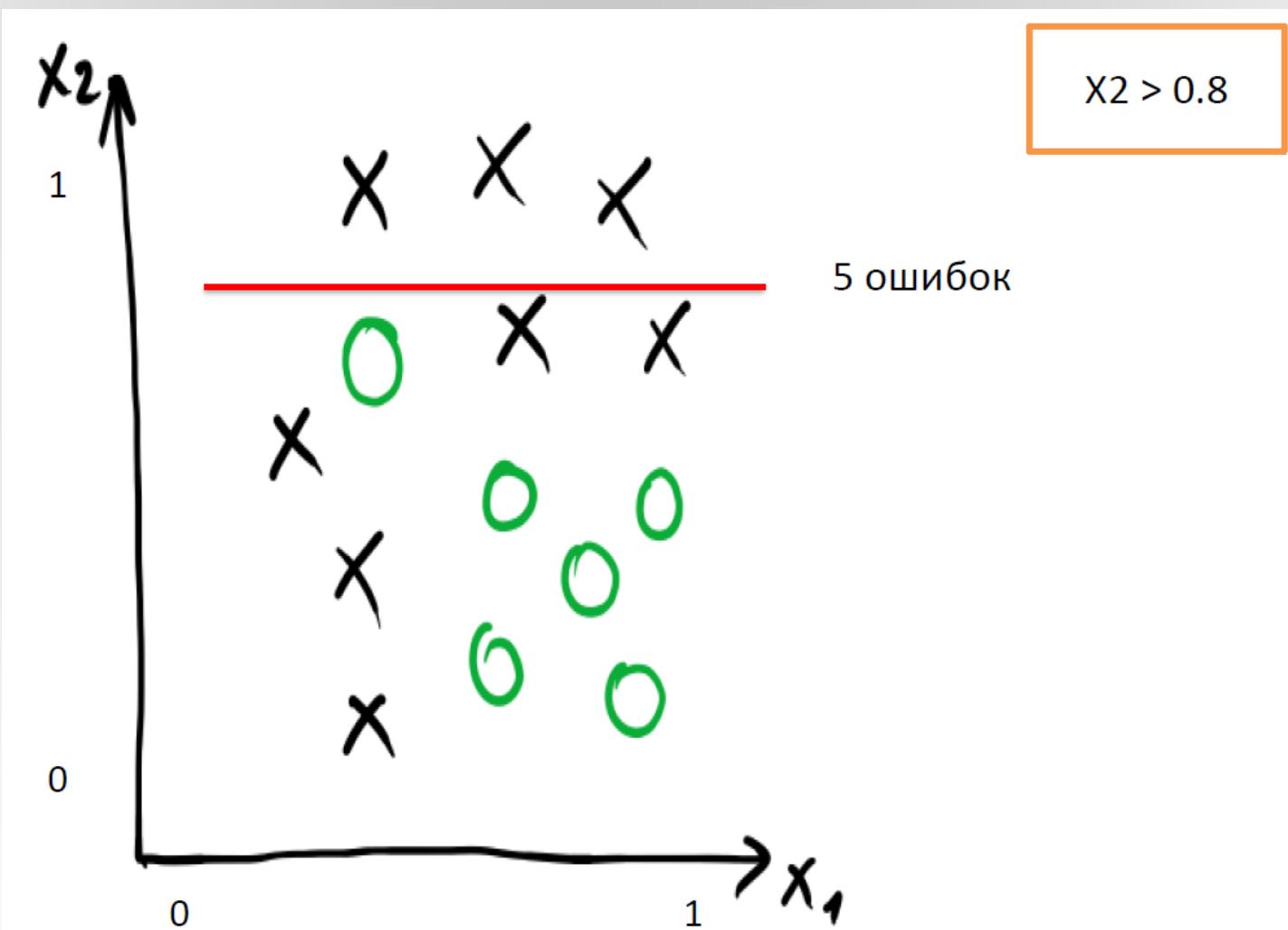
- найдем наилучшие j и t
- создадим корень дерева, поставив в него предикат $[x_j < t]$.

2 шаг: Для каждой из полученных подвыборок R_1 и R_2 рекурсивно применим шаг 1. И т.д.

В каждой вершине на каждом шаге проверяем, не выполнилось ли условие останова. Если выполнилось, то объявляем вершину листом и записываем в него предсказание.

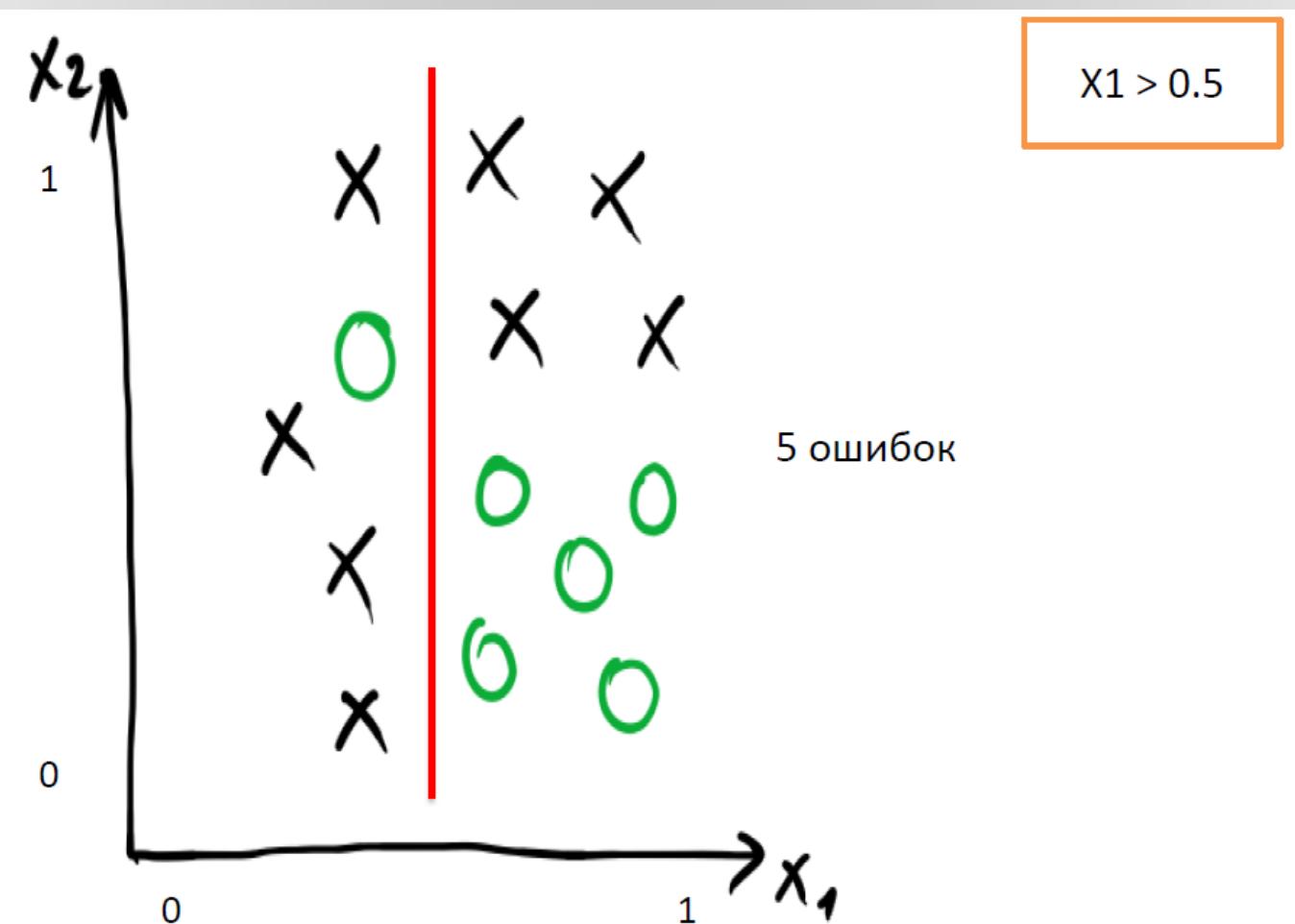
ПРИМЕР

- Жадно найдем наилучший предикат



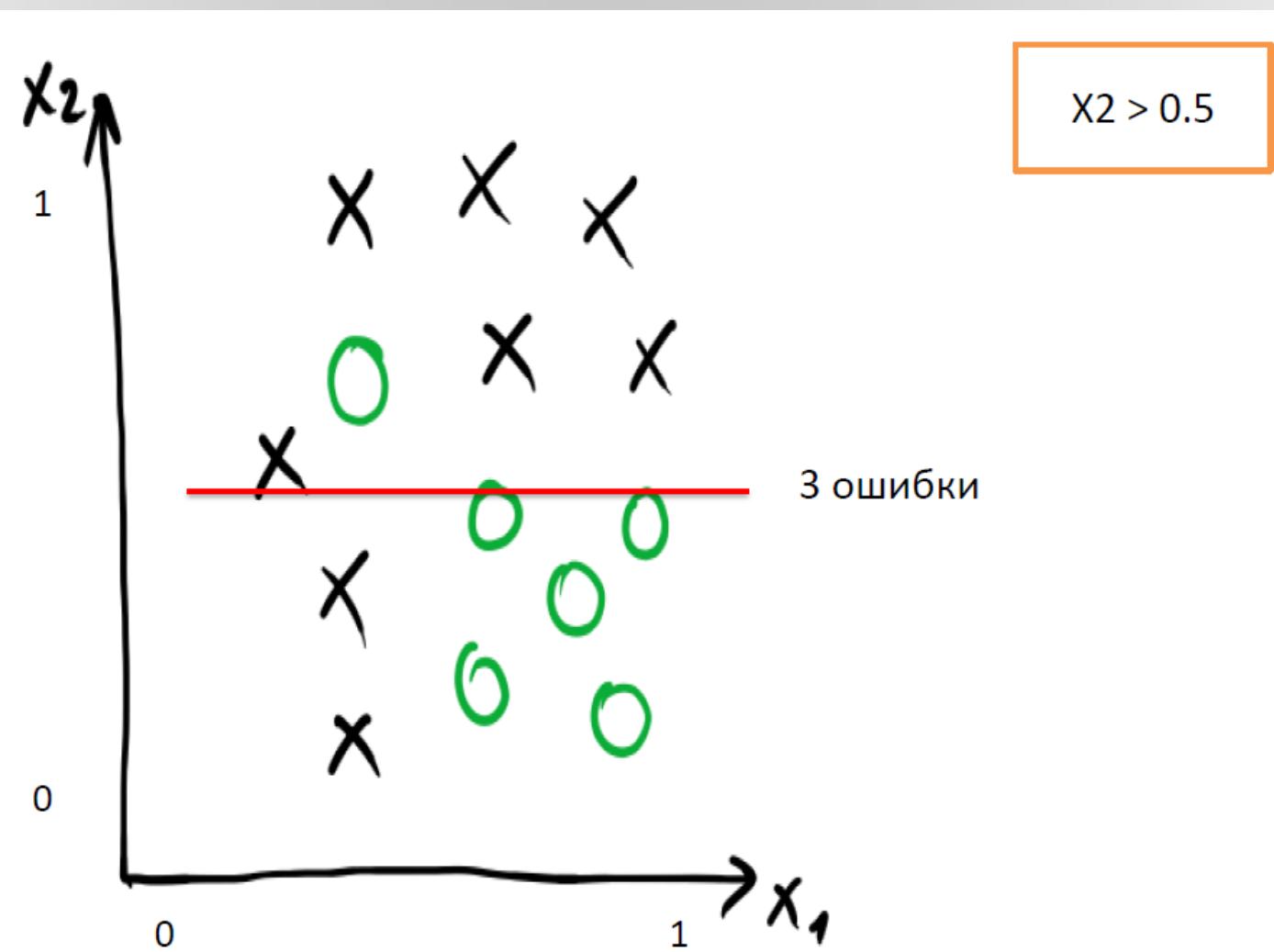
ПРИМЕР

- Жадно найдем наилучший предикат



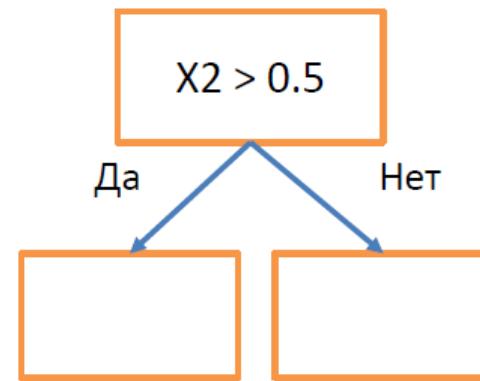
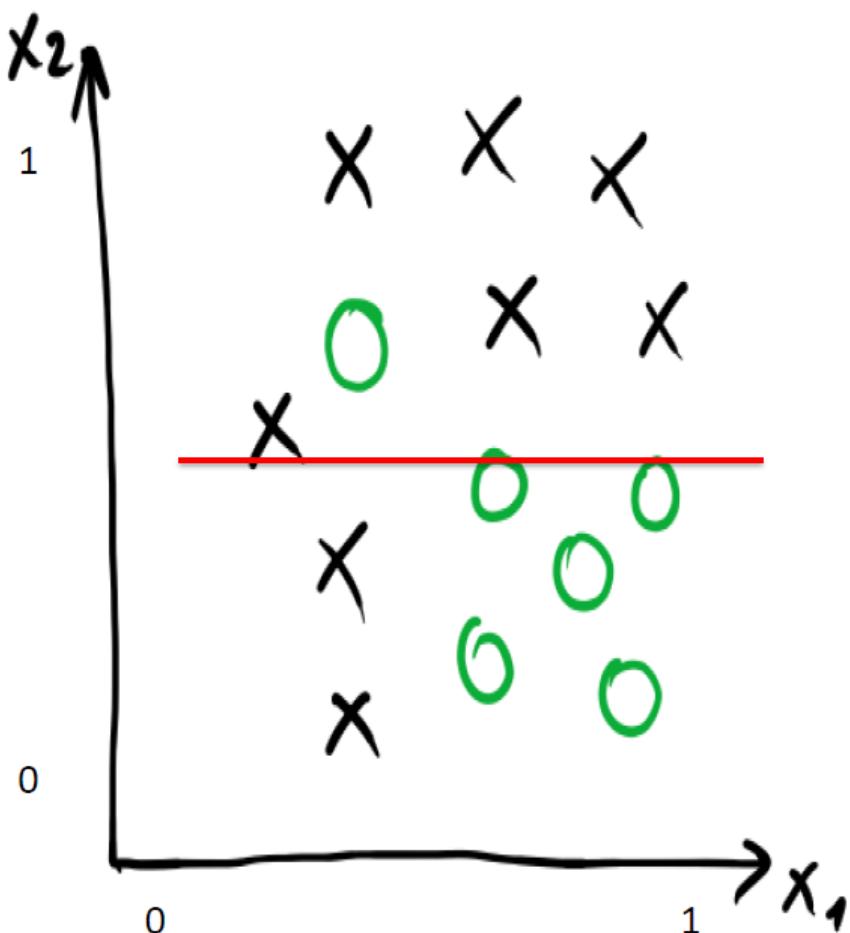
ПРИМЕР

- Жадно найдем наилучший предикат



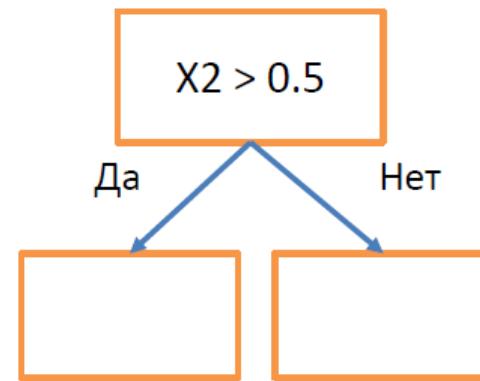
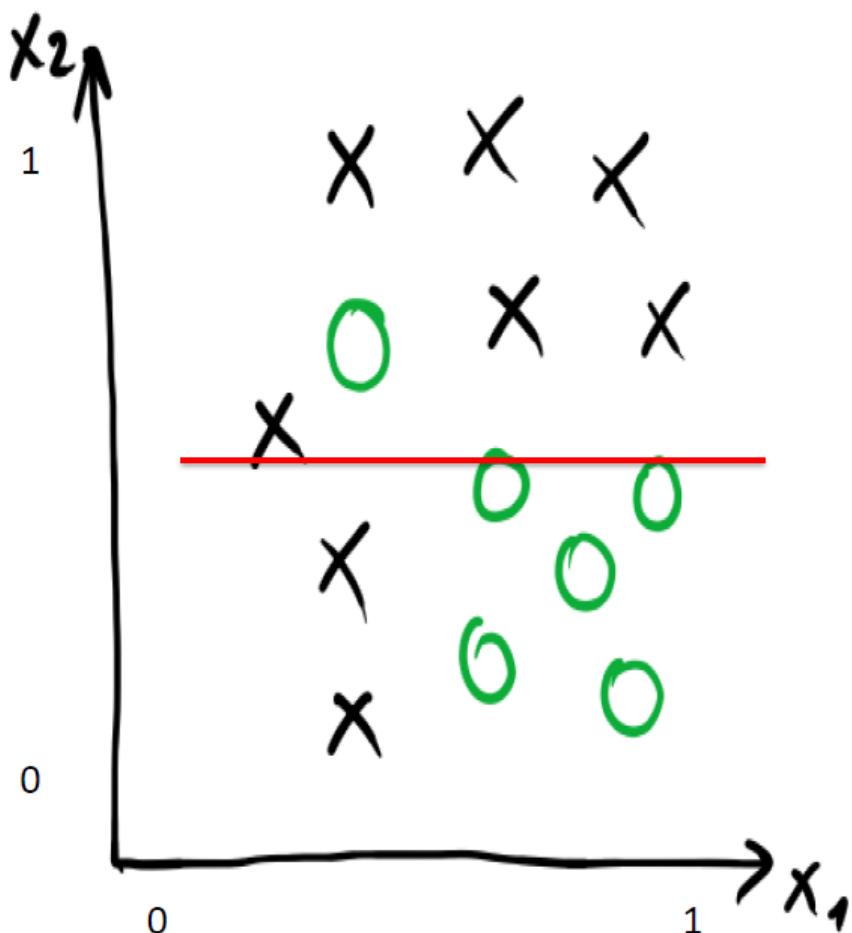
ПРИМЕР

- Нашли лучшее первое ветвление



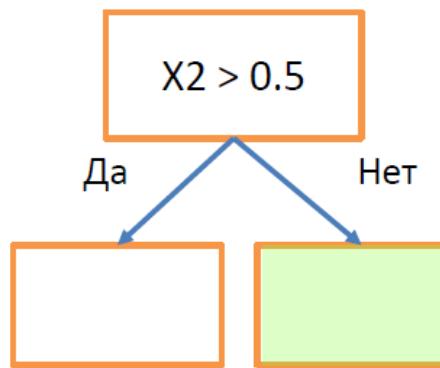
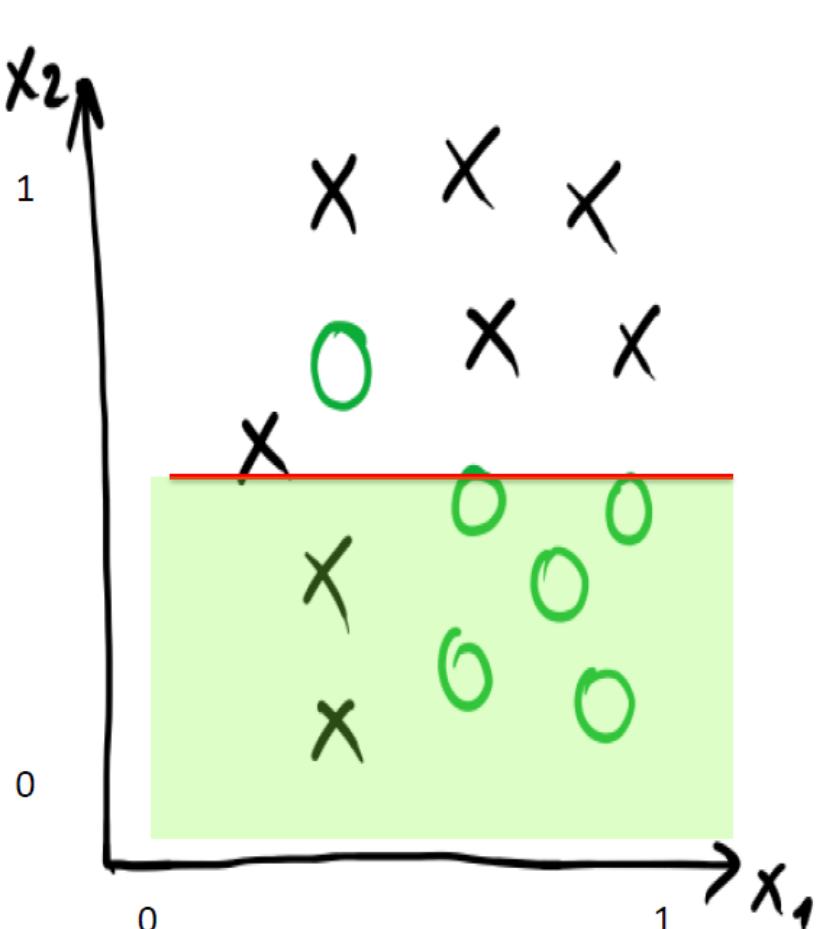
ПРИМЕР

- Нашли лучшее первое ветвление



ПРИМЕР

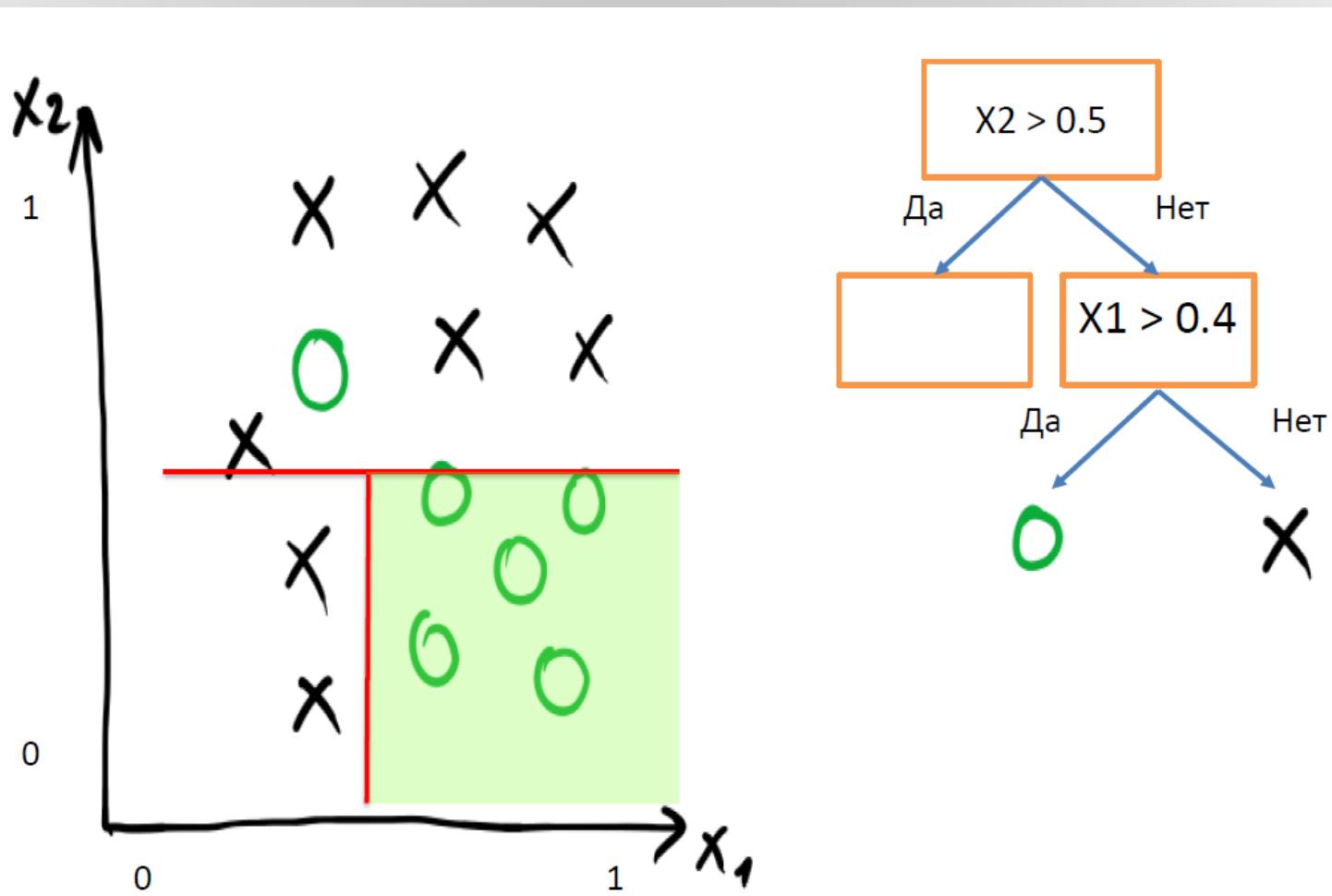
- Нашли лучшее первое ветвление



Продолжим
этую ветку

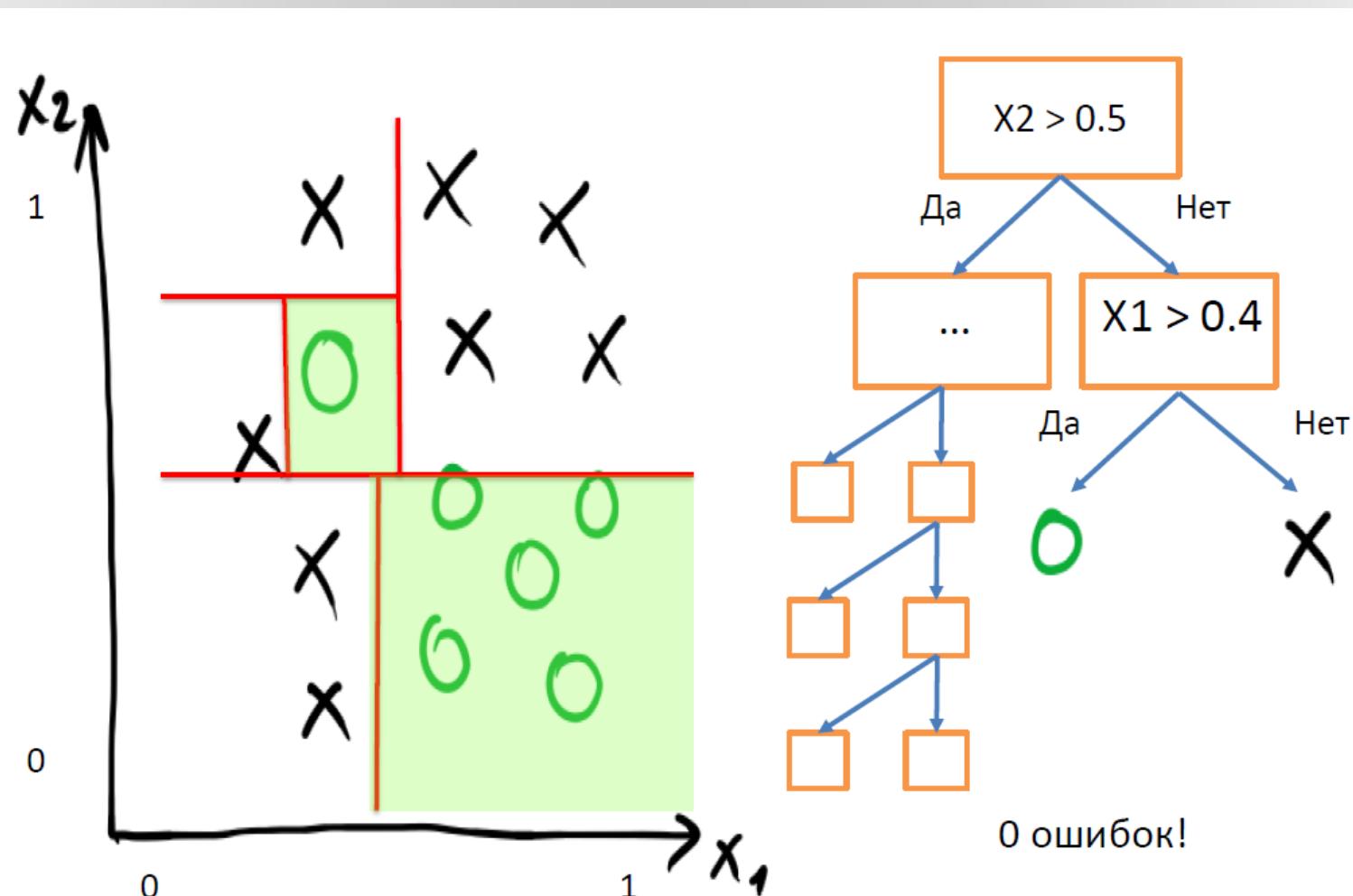
ПРИМЕР

- Нашли лучшее второе ветвление



ПРИМЕР

- Построили всё дерево



ПЕРЕОБУЧЕНИЕ

Для любой выборки можно построить решающее дерево, не допускающее на ней ни одной ошибки. Такое дерево скорее всего будет переобученным.

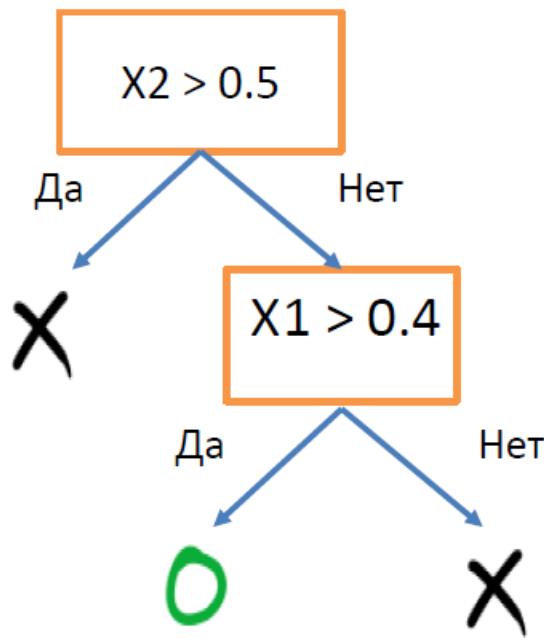
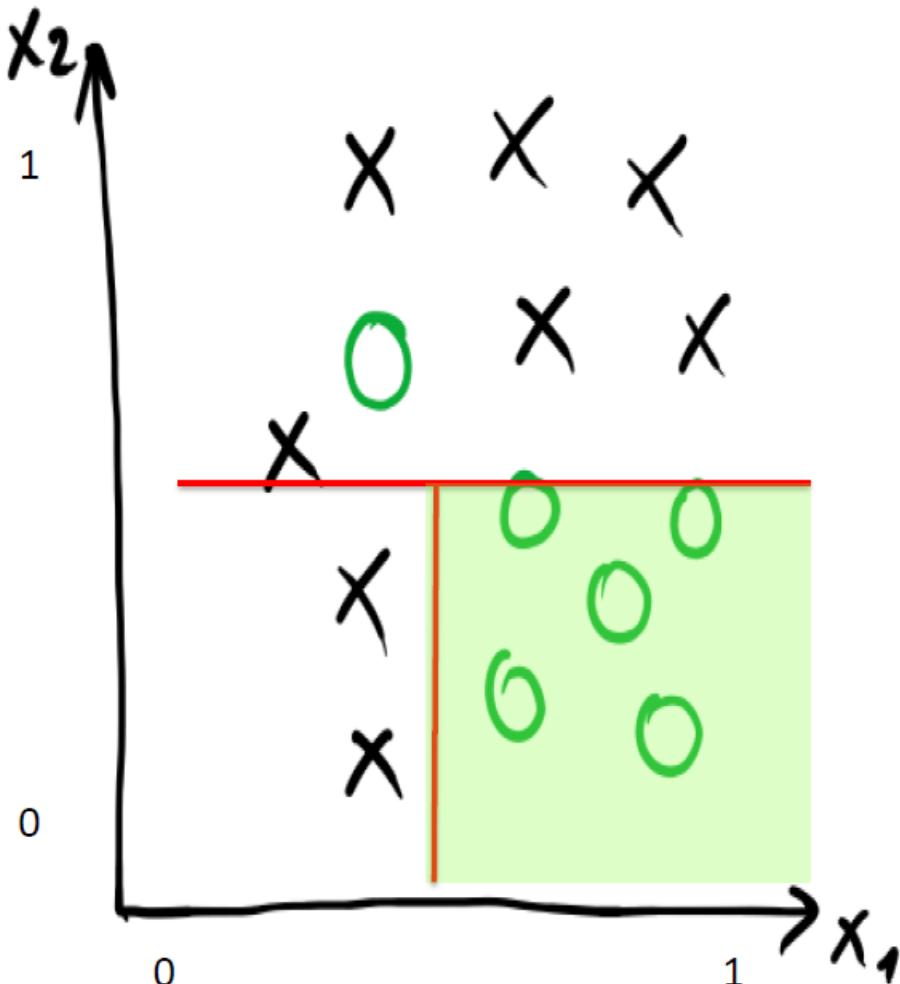
ОБРАБОТКА ПРОПУЩЕННЫХ ЗНАЧЕНИЙ И СТРИЖКА ДЕРЕВА

- Можно модифицировать процедуру разбиения выборки в вершине дерева так, чтобы была возможность обрабатывать пропущенные значения
- После того, как дерево построено, можно удалить несколько его вершин, чтобы понизить сложность алгоритма и снизить переобучение (стрижка дерева или *pruning*).

ЧТО ВЛИЯЕТ НА ПОСТРОЕНИЕ РЕШАЮЩЕГО ДЕРЕВА

- вид предикатов в вершинах
- функционал качества $Q(X, j, t)$
- критерий останова
- метод обработки пропущенных значений
- метод стрижки

ПРИМЕР



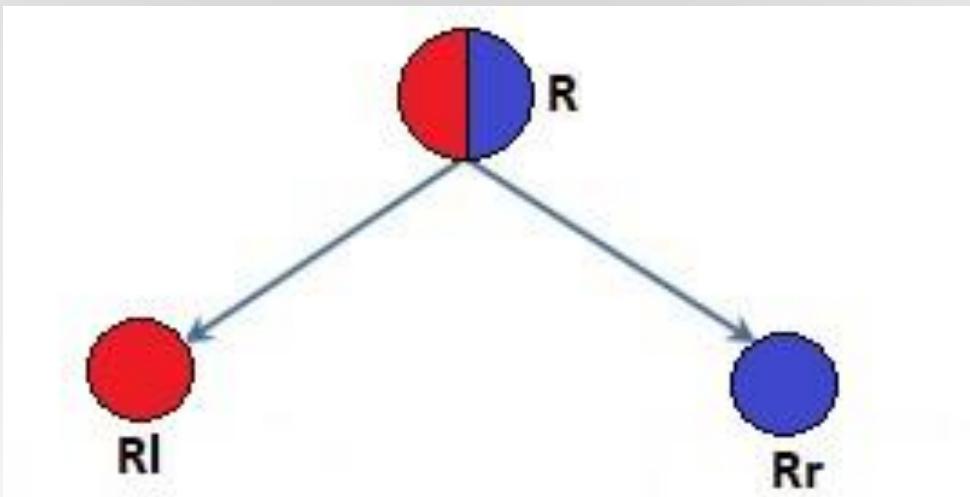
1 ошибка, но
скорее всего будет
лучше на тесте!

КРИТЕРИИ ИНФОРМАТИВНОСТИ

В каждой вершине оптимизируем функционал $Q(X, j, t)$.

- Пусть R – множество объектов, попадающих в вершину на данном шаге, а R_l и R_r - объекты, попадающие в левую и правую ветки после разбиения.

Цель: хотим, чтобы после разбиения объектов на две группы внутри каждой группы как можно больше объектов было одного класса.



КРИТЕРИИ ИНФОРМАТИВНОСТИ

- Пусть R – множество объектов, попадающих в вершину на данном шаге, а R_l и R_r - объекты, попадающие в левую и правую ветки после разбиения.

Цель: хотим, чтобы после разбиения объектов на две группы внутри каждой группы как можно больше объектов было одного класса.

- $H(R)$ - критерий информативности - мера неоднородности целевых (разнообразие) переменных внутри группы R .
- Чем меньше разнообразие целевой переменной внутри группы, тем меньше значение $H(R)$. То есть хотим

$$H(R_l) \rightarrow \min, H(R_r) \rightarrow \min$$

КРИТЕРИИ ИНФОРМАТИВНОСТИ

- Пусть R – множество объектов, попадающих в вершину на данном шаге, а R_l и R_r - объекты, попадающие в левую и правую ветви после разбиения.

Цель: хотим, чтобы после разбиения объектов на две группы внутри каждой группы как можно больше объектов было одного класса.

- Чем меньше разнообразие целевой переменной внутри группы, тем меньше значение $H(R)$. То есть

$$H(R_l) \rightarrow \min, H(R_r) \rightarrow \min$$

- Определим функционал Q по формуле:

$$Q(R, j, t) = H(R) - \frac{|R_l|}{|R|} H(R_l) - \frac{|R_r|}{|R|} H(R_r)$$

КРИТЕРИИ ИНФОРМАТИВНОСТИ

- Пусть R – множество объектов, попадающих в вершину на данном шаге, а R_l и R_r - объекты, попадающие в левую и правую ветви после разбиения.

Цель: хотим, чтобы после разбиения объектов на две группы внутри каждой группы как можно больше объектов было одного класса.

- Чем меньше разнообразие целевой переменной внутри группы, тем меньше значение $H(R)$. То есть

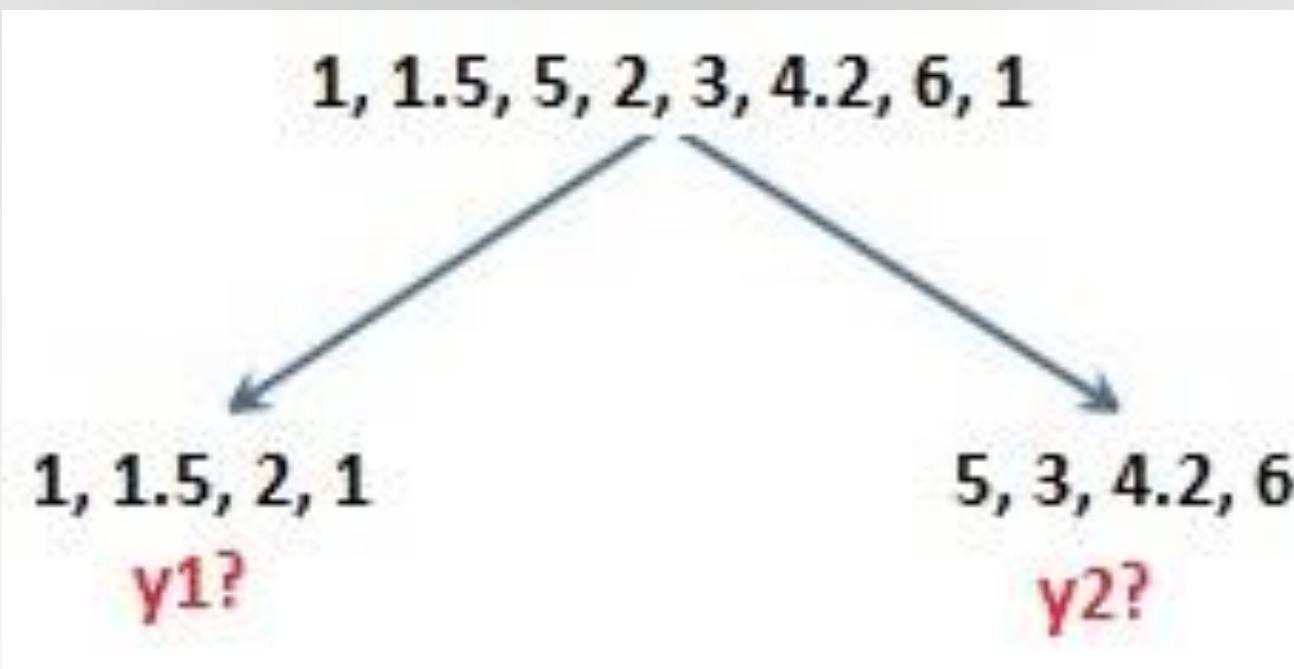
$$H(R_l) \rightarrow \min, H(R_r) \rightarrow \min$$

- Определим функционал Q по формуле:

$$Q(R, j, t) = H(R) - \frac{|R_l|}{|R|} H(R_l) - \frac{|R_r|}{|R|} H(R_r) \rightarrow \max_{j,t}$$

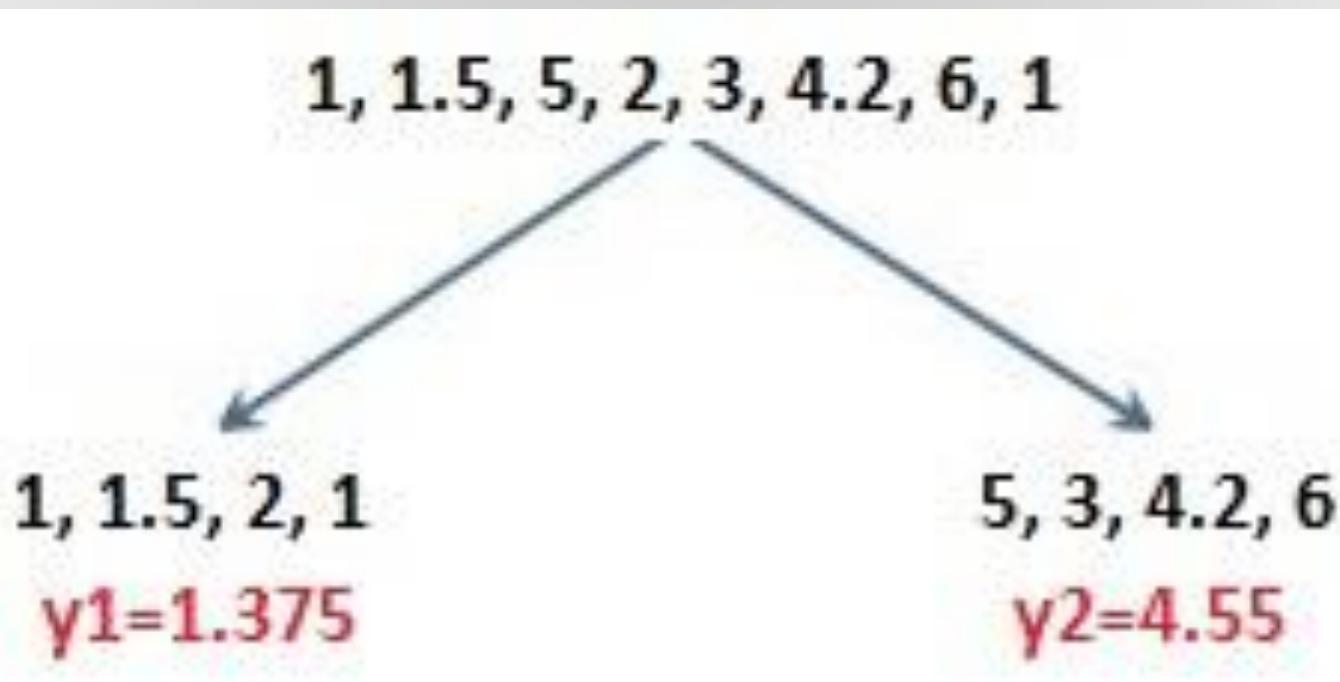
ПРИМЕР: РЕШАЮЩЕЕ ДЕРЕВО В ЗАДАЧЕ РЕГРЕССИИ

Предположим, что в лист дерева попало несколько объектов. В каждом листе дерево предсказывает константу. Какую константу выгоднее всего выдать в качестве ответа?



ПРИМЕР: РЕШАЮЩЕЕ ДЕРЕВО В ЗАДАЧЕ РЕГРЕССИИ

Если в качестве функционала ошибки в листе использовать среднеквадратичную ошибку, то в качестве ответа надо выдавать среднее значение целевых переменных всех объектов, попавших в лист.



ВИД КРИТЕРИЯ ИНФОРМАТИВНОСТИ

- В каждом листе дерево выдает константу c (вещественное число – в регрессии, класс или вероятность класса – в классификации).
- Чем лучше объекты в листе предсказываются этой константой, тем меньше средняя ошибка на объектах:

$$H(R) = \min_{c \in \mathbb{R}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} L(y_i, c),$$

где $L(y, c)$ – некоторая функция потерь.

$H(R)$ В ЗАДАЧЕ РЕГРЕССИИ

Если в качестве функции потерь мы берём квадратичную ошибку, то

$$H(R) = \min_{c \in \mathbb{R}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2.$$

$H(R)$ В ЗАДАЧЕ РЕГРЕССИИ

Если в качестве функции потерь мы берём квадратичную ошибку, то

$$H(R) = \min_{c \in \mathbb{R}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2.$$

Её минимум достигается при

$$c = \frac{1}{|R|} \sum_{(x_i, y_i) \in R} y_i,$$

то есть в листе предсказывается среднее значение целевой переменной на объектах, попавших в лист.

$H(R)$ В ЗАДАЧЕ РЕГРЕССИИ

Если в качестве функции потерь мы берём квадратичную ошибку,
то

$$H(R) = \min_{c \in \mathbb{R}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2.$$

Её минимум достигается при

$$c = \frac{1}{|R|} \sum_{(x_i, y_i) \in R} y_i,$$

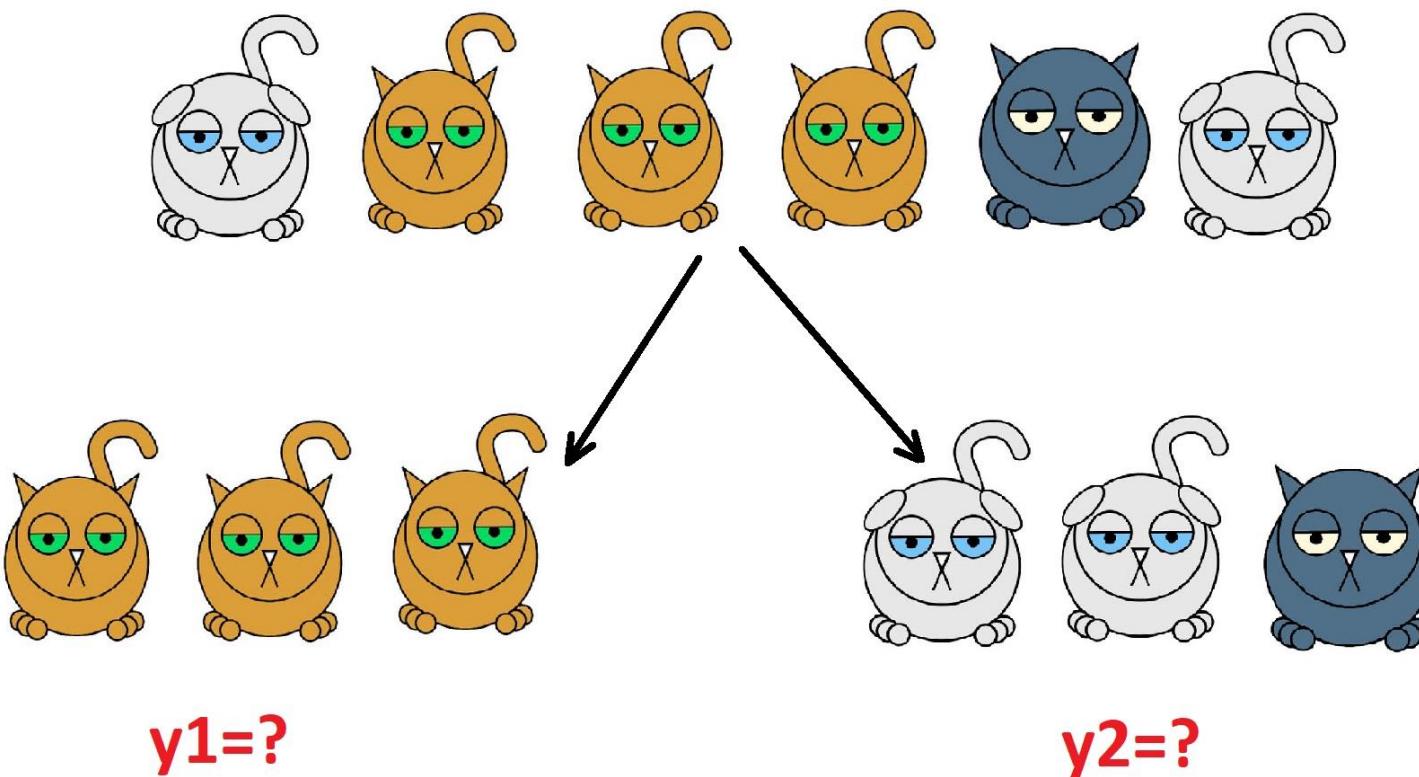
то есть в листе предсказывается среднее значение целевой переменной на объектах, попавших в лист.

Значит, *информативность $H(R)$ в листе – это дисперсия целевой переменной (для объектов, попавших в этот лист)*.

Чем меньше дисперсия, тем меньше разброс целевой переменной объектов, попавших в лист.

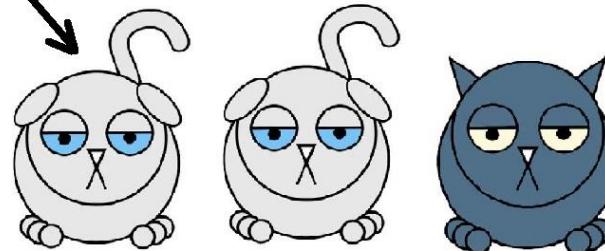
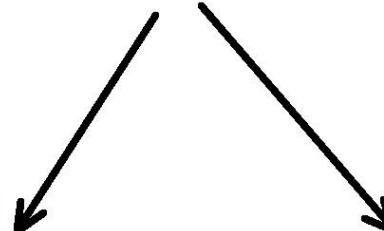
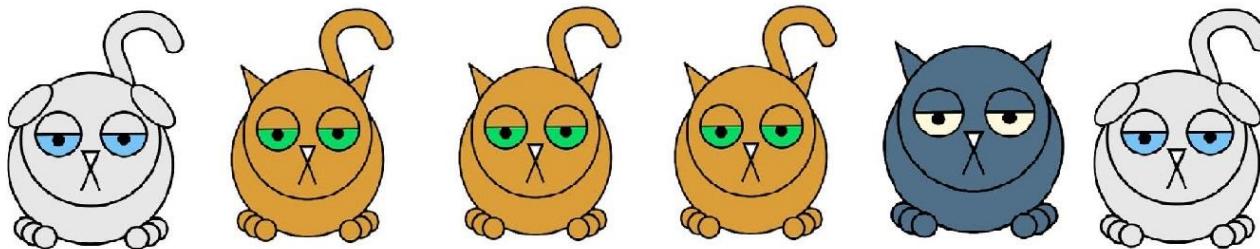
ПРИМЕР: РЕШАЮЩЕЕ ДЕРЕВО В ЗАДАЧЕ КЛАССИФИКАЦИИ

Предположим, что в лист дерева попало несколько объектов. В каждом листе дерево предсказывает класс объекта. Какой класс выгоднее всего выдать в качестве ответа?



ПРИМЕР: РЕШАЮЩЕЕ ДЕРЕВО В ЗАДАЧЕ КЛАССИФИКАЦИИ

Разумнее всего в качестве ответа в листе выдавать самый представительный класс.



$y_1 =$ 

$y_2 =$ 

$H(R)$ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Решаем задачу классификации с K классами: $1, 2, \dots, K$.

- Пусть p_k доля объектов класса k , попавших в вершину:

$$p_k = \frac{1}{|R|} \sum_{(x_i, y_i) \in R} [y_i = k]$$

- Пусть k_* - самый представительный класс в данной вершине:

$$k_* = \operatorname{argmax}_k p_k$$

Ошибка классификации:

$$H(R) = \min_{c \in Y} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} [y_i \neq c]$$

$H(R)$ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Критерий Джини

- Будем в каждой вершине в качестве ответа выдавать не класс, а распределение вероятностей классов:
 $c = (c_1, \dots, c_K), \sum_i c_i = 1.$
- Качество распределения можно измерить с помощью критерия Бриера:

$$H(R) = \min_c \frac{1}{|R|} \sum_{(x_i, y_i) \in R} \sum_{k=1}^K (c_k - [y_i = k])^2$$

Утверждение.

- 1) Минимальное значение функционала $H(R)$ достигается на векторе, состоящем из долей классов: $c_* = (p_1, \dots, p_K)$
- 2) На векторе c_* функционал (*) переписывается в виде

$$H(R) = \sum_{k=1}^K p_k(1 - p_k) \text{ (критерий Джини).}$$

$H(R)$ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Энтропийный критерий

Запишем логарифм правдоподобия:

$$H(R) = \min_c \left(-\frac{1}{|R|} \sum_{(x_i, y_i) \in R} \sum_{k=1}^K [y_i = k] \log c_k \right) (*)$$

На векторе $c_* = (p_1, \dots, p_K)$ функционал (*) записывается в виде

$$H(R) = -\sum_{k=1}^K p_k \log p_k \text{ (энтропия)}$$

$H(R)$ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Энтропийный критерий

Запишем логарифм правдоподобия:

$$H(R) = \min_c \left(-\frac{1}{|R|} \sum_{(x_i, y_i) \in R} \sum_{k=1}^K [y_i = k] \log c_k \right) (*)$$

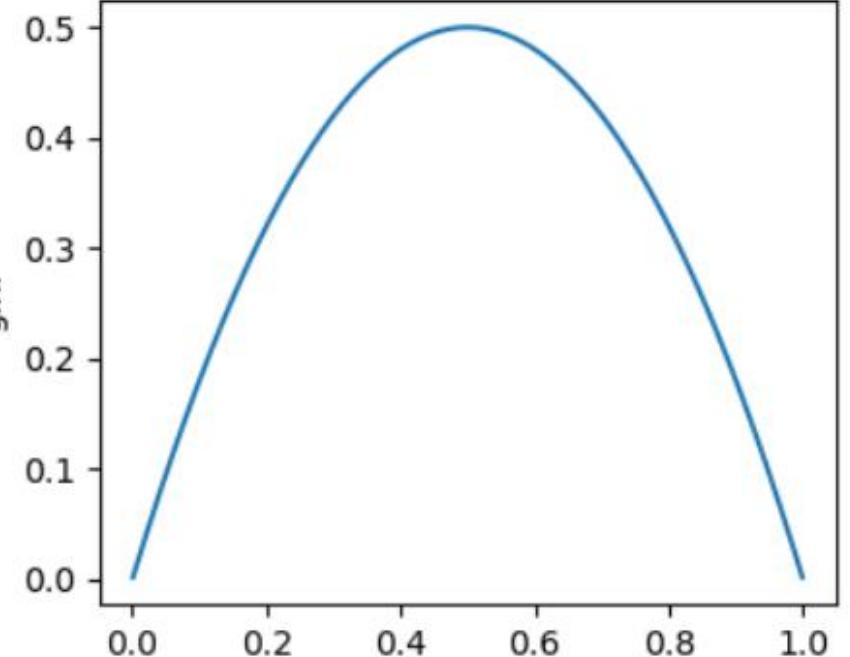
На векторе $c_* = (p_1, \dots, p_K)$ функционал (*) записывается в виде

$$H(R) = -\sum_{k=1}^K p_k \log p_k \text{ (энтропия)}$$

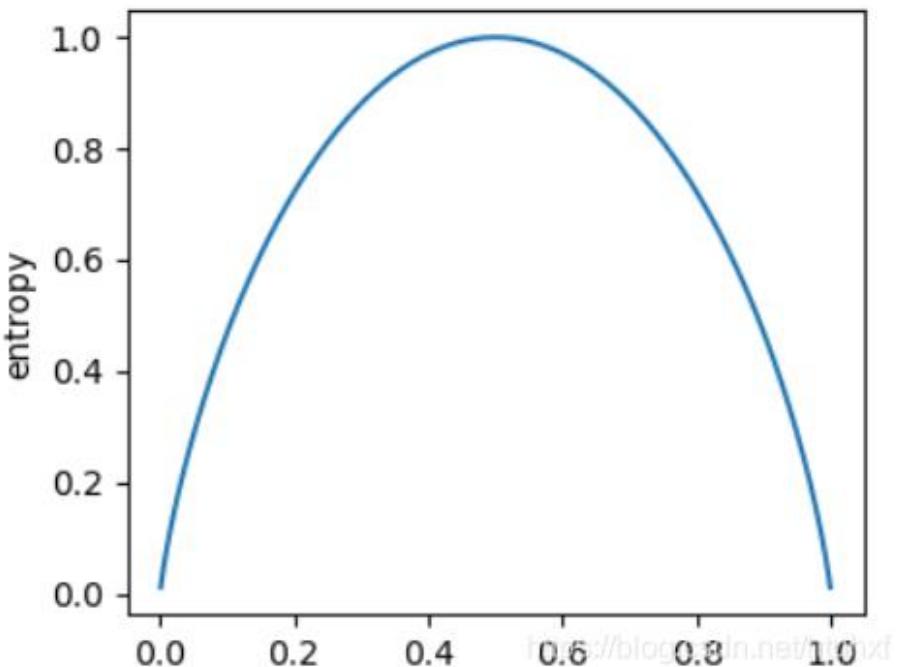
- Энтропия $H(R) \geq 0$ (минимум на распределении $p_i = 1, p_j = 0, j \neq i$)
- $\max H(R)$ достигается на равномерном распределении $p_1 = \dots = p_K = \frac{1}{K}$.

КРИТЕРИИ GINI И ENTROPY

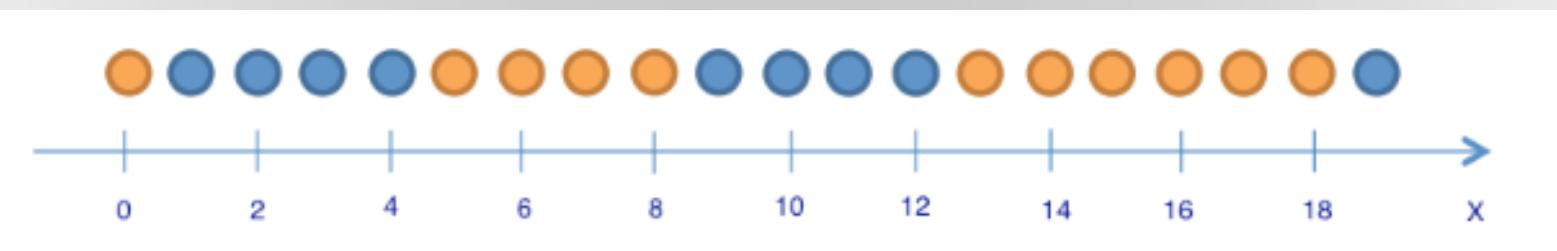
GINI



ENTROPY

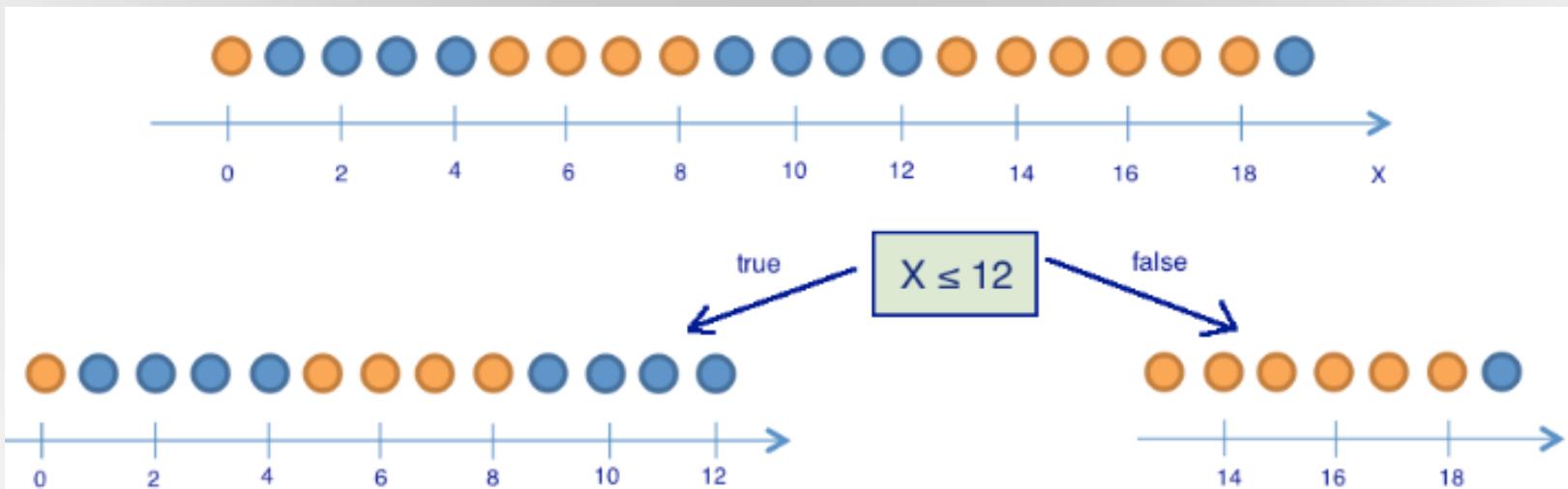


ПРИМЕР ИСПОЛЬЗОВАНИЯ ЭНТРОПИЙНОГО КРИТЕРИЯ



- $p_1 = \frac{9}{20}, p_2 = \frac{11}{20} \Rightarrow$ энтропия $H_0 = -\frac{9}{20} \log \frac{9}{20} - \frac{11}{20} \log \frac{11}{20} \approx 1$

ПРИМЕР ИСПОЛЬЗОВАНИЯ ЭНТРОПИЙНОГО КРИТЕРИЯ



- В левой части $H_l = -\frac{5}{13} \log \frac{5}{13} - \frac{8}{13} \log \frac{8}{13} \approx 0.96$
 - В правой части $H_r = -\frac{1}{7} \log \frac{1}{7} - \frac{6}{7} \log \frac{6}{7} \approx 0.6$
- То есть $Q = H_0 - \frac{|R_l|}{R} H_l - \frac{|R_r|}{|R|} H_r = 1 - \frac{13}{20} \cdot 0.96 - \frac{7}{20} \cdot 0.6 \approx 0.16$

КРИТЕРИИ ОСТАНОВА

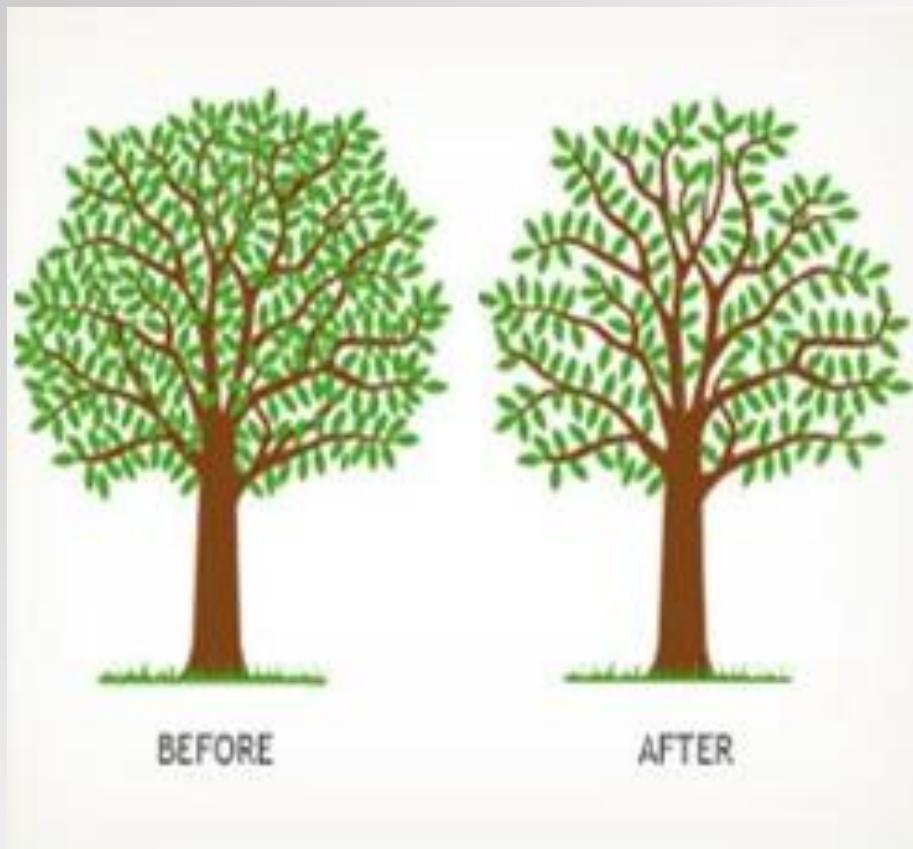
- Ограничение максимальной глубины дерева (`max_depth`)
- Ограничение минимального числа объектов в листьях (`min_samples_leaf`)
- Ограничение максимального числа листьев в дереве
- Останов в случае, если все объекты в листе из одного класса
- Требование, что функционал качества при дроблении увеличивался как минимум на $s\%$.

СТРИЖКА ДЕРЕВА (PRUNING)

Алгоритм:

- 1) Строится переобученное дерево (в каждом листе один объект)
- 2) Производится оптимизация его структуры с целью уменьшения переобучения

- Стрижка – альтернатива критериям останова



COST-COMPLEXITY PRUNING

Пусть $R(T)$ – ошибка на дереве T .

Введём регуляризованный функционал

$$R_\alpha(T) = R(T) + \alpha|T|,$$

где $|T|$ - количество вершин в дереве.

- Тогда при построении дерева и оптимизации функционала $R_\alpha(T)$ дерево не будет иметь большое количество вершин, а потому, будет менее переобученным.

ОБРАБОТКА ПРОПУЩЕННЫХ ЗНАЧЕНИЙ

- Пусть есть некоторый предикат $\beta(x) = [x_j < t]$, но в выборке R есть объекты, для которых неизвестно значение признака x_j : обозначим их V_j .

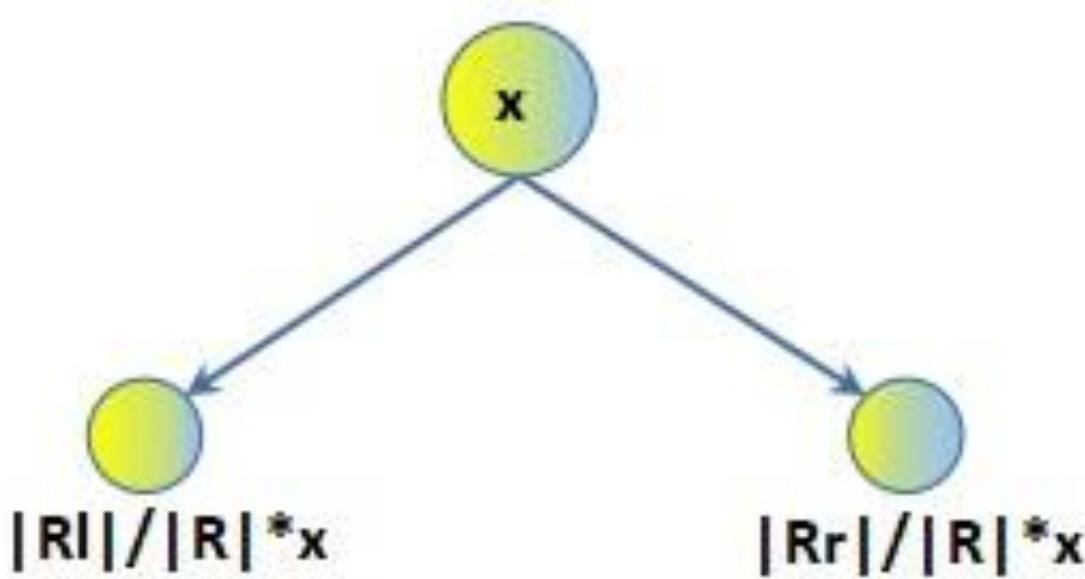
Тогда функционал качества можно вычислять, просто игнорируя объекты из V_j :

$$Q(R, j, t) \approx \frac{|R \setminus V_j|}{|R|} Q(R \setminus V_j, j, t)$$

- Далее выбираем наилучший предикат по данному функционалу качества

ОБРАБОТКА ПРОПУЩЕННЫХ ЗНАЧЕНИЙ

- Поместим объекты из V_j и в левое, и в правое поддерево после разбиения предикатом β . Присвоим этим объектам веса $\frac{|R_l|}{|R|}$ и $\frac{|R_r|}{|R|}$ соответственно.



ПОЛУЧЕНИЕ ПРЕДСКАЗАНИЙ ДЛЯ ПРОПУЩЕННЫХ ЗНАЧЕНИЙ

- Если объект попал в вершину, предикат в которой для него не может быть вычислен из-за пропуска, то он попадает в оба поддерева с весами, и ответы в обоих поддеревьях затем усредняются.

Обозначим прогноз вероятности класса k для объекта x в поддереве R_m как $a_{mk}(x)$. Тогда:

$$\begin{cases} a_{lk}(x), & \beta(x) = 0 \\ a_{rk}(x), & \beta(x) = 1 \\ \frac{|R_l|}{|R|} a_{lk}(x) + \frac{|R_r|}{|R|} a_{rk}(x), & \beta(x) \text{ нельзя вычислить} \end{cases}$$

ДРУГИЕ СПОСОБЫ ОБРАБОТКИ ПРОПУЩЕННЫХ ЗНАЧЕНИЙ

- Метод суррогатных предикатов:

Если есть объекты, для которых невозможно вычислить значение предиката $[x_j < t]$, то находим другой предикат, который дает максимально похожее разбиение на R_l и R_m .

- Можно заменить все пропущенные значения на 0 или на какое-то уникальное число (например, -999). Тогда можно будет использовать обычную процедуру построения дерева.

ОБРАБОТКА КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ

- *Multi-way splits*: можно разбивать вершину на столько поддеревьев, сколько различных значений имеет категориальный признак.

ОБРАБОТКА КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ

- Multi-way splits: можно разбивать вершину на столько поддеревьев, сколько различных значений имеет категориальный признак. **Получаем очень большие деревья.**
- Поделим множество возможных значений признака $Q = \{u_1, \dots, u_m\}$ на два подмножества $Q = Q_1 \sqcup Q_2$, определим предикат $\beta(x) = [x_j \in Q_1]$.

ОБРАБОТКА КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ

- Multi-way splits: можно разбивать вершину на столько поддеревьев, сколько различных значений имеет категориальный признак. **Получаем очень большие деревья.**
- Поделим множество возможных значений признака $Q = \{u_1, \dots, u_m\}$ на два подмножества $Q = Q_1 \sqcup Q_2$, определим предикат $\beta(x) = [x_j \in Q_1]$. **Для построения оптимального предиката нужно перебрать $2^{m-1} - 1$ вариантов разбиения $Q = Q_1 \sqcup Q_2$.**

ОБРАБОТКА КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ В ЗАДАЧАХ БИНАРНОЙ КЛАССИФИКАЦИИ

- Рассмотрим вершину дерева. Пусть $N(u_1)$ – количество объектов в этой вершине со значением категориального признака $x_j = u_1$.
- Тогда $\frac{\sum_{x_{ij}=u_1} [y_i=+1]}{N(u_1)}$ – доля положительных объектов среди объектов с $x_j = u_1$.
- Упорядочим значения категориального признака x_j по возрастанию этих долей:

$$\frac{\sum_{x_{ij}=u_{(1)}} [y_i=+1]}{N(u_{(1)})} \leq \frac{\sum_{x_{ij}=u_{(2)}} [y_i=+1]}{N(u_{(2)})} \leq \dots \leq \frac{\sum_{x_{ij}=u_{(m)}} [y_i=+1]}{N(u_{(m)})}$$

- Заменим категорию $u_{(i)}$ на i и будем работать с полученным вещественным признаком.

ОБРАБОТКА КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ В ЗАДАЧАХ БИНАРНОЙ КЛАССИФИКАЦИИ

Утверждение.

Данный подход с использованием критерия Джини или энтропийного критерия дает тот же результат, как если бы мы искали оптимальное разбиение на $Q = Q_1 \sqcup Q_2$.

ОБРАБОТКА КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ В ЗАДАЧАХ РЕГРЕССИИ

- Упорядочим значения категориального признака x_j по среднему ответу на объектах с таким значением:

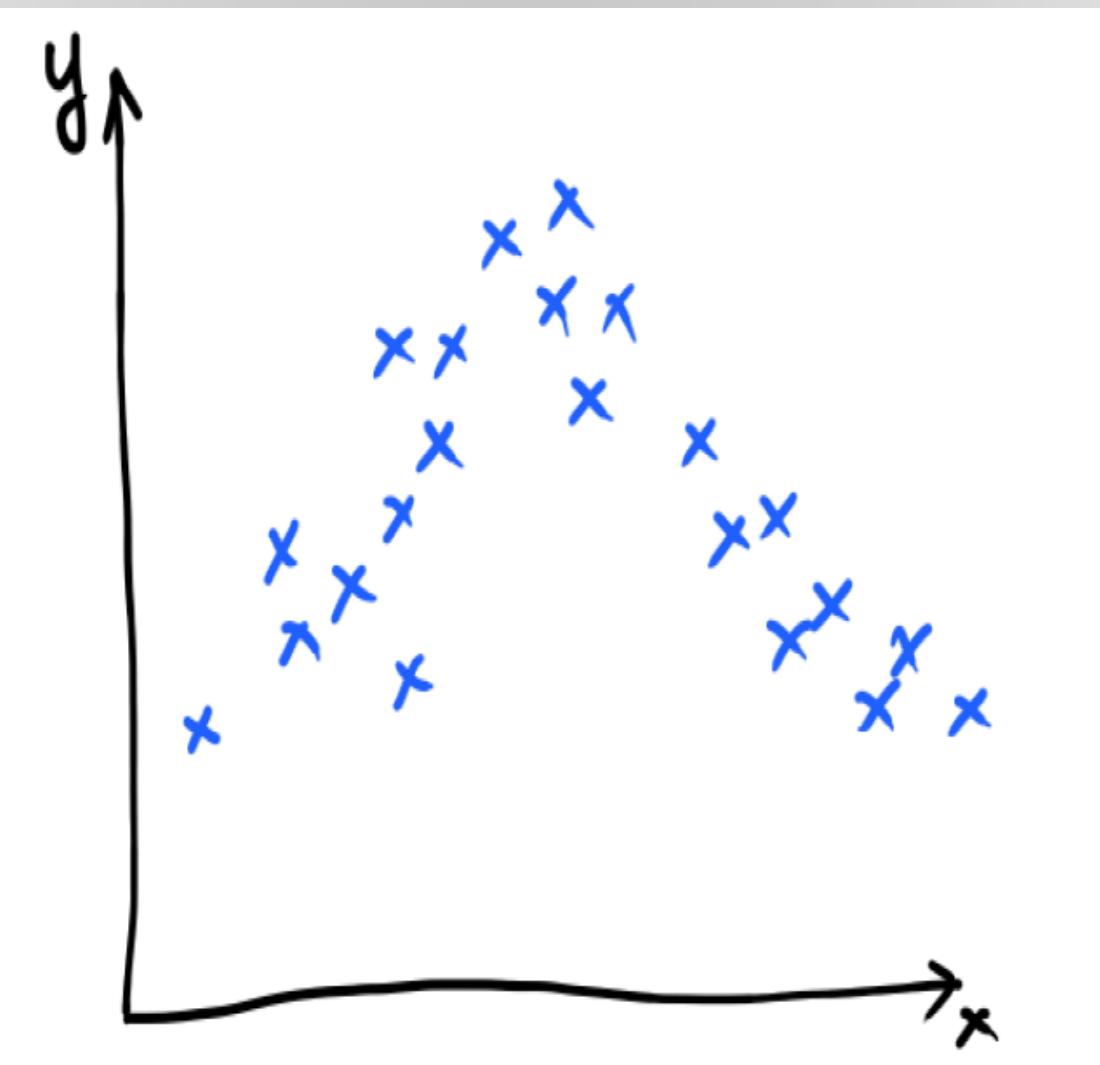
$$\frac{\sum_{x_{ij}=u_{(1)}} y_i}{N(u_{(1)})} \leq \frac{\sum_{x_{ij}=u_{(2)}} y_i}{N(u_{(2)})} \leq \dots \leq \frac{\sum_{x_{ij}=u_{(m)}} y_i}{N(u_{(m)})}$$

- Заменим категорию $u_{(i)}$ на i и будем работать с полученным вещественным признаком.

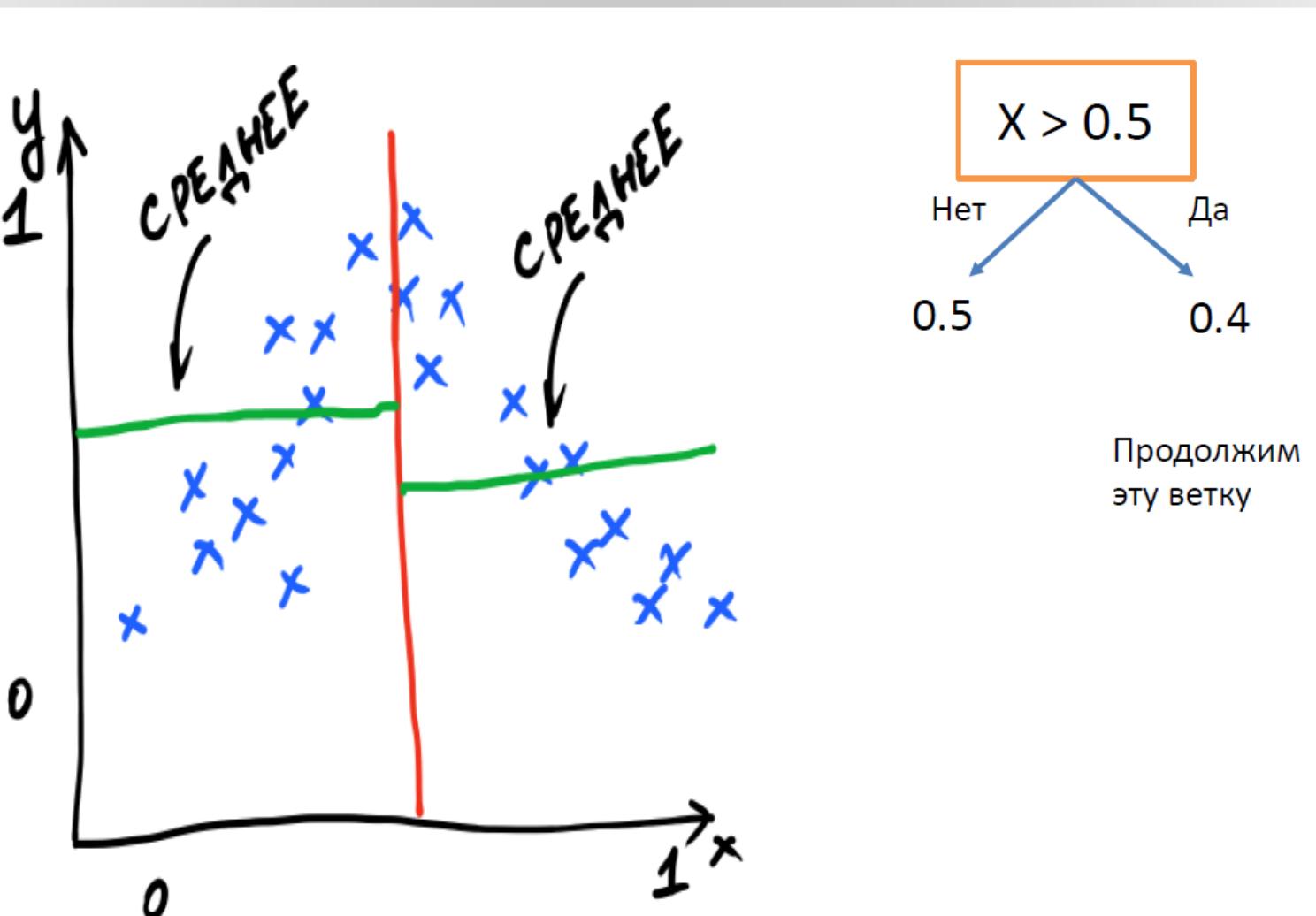
Утверждение.

Данный подход с использованием MSE-функционала дает тот же результат, как если бы мы искали оптимальное разбиение на $Q = Q_1 \sqcup Q_2$.

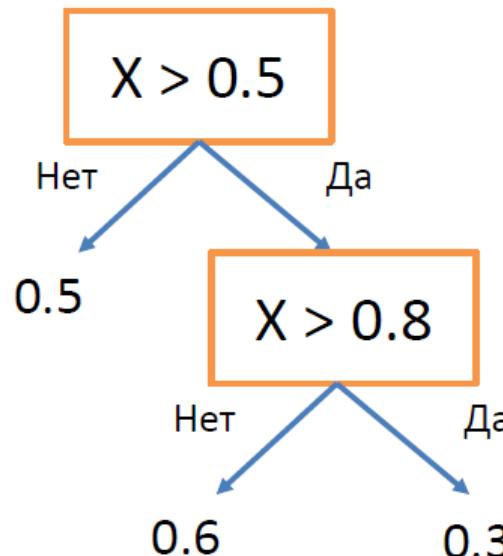
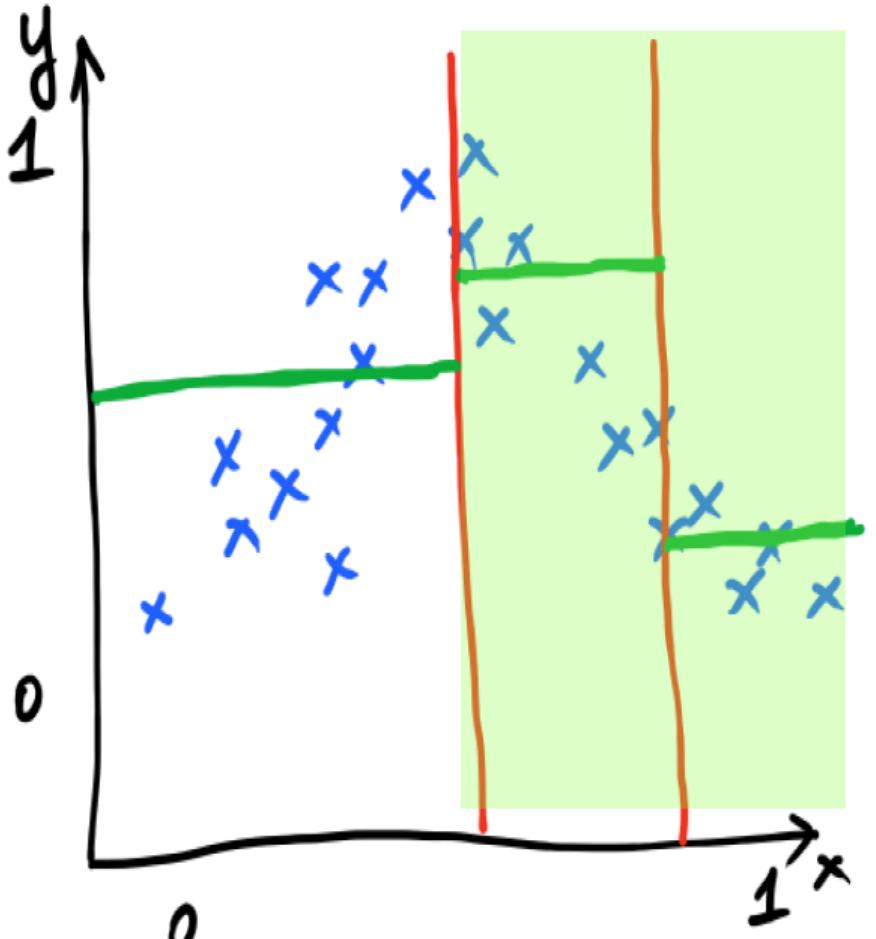
ПРИМЕР ПОСТРОЕНИЯ РЕШАЮЩЕГО ДЕРЕВА В ЗАДАЧЕ РЕГРЕССИИ



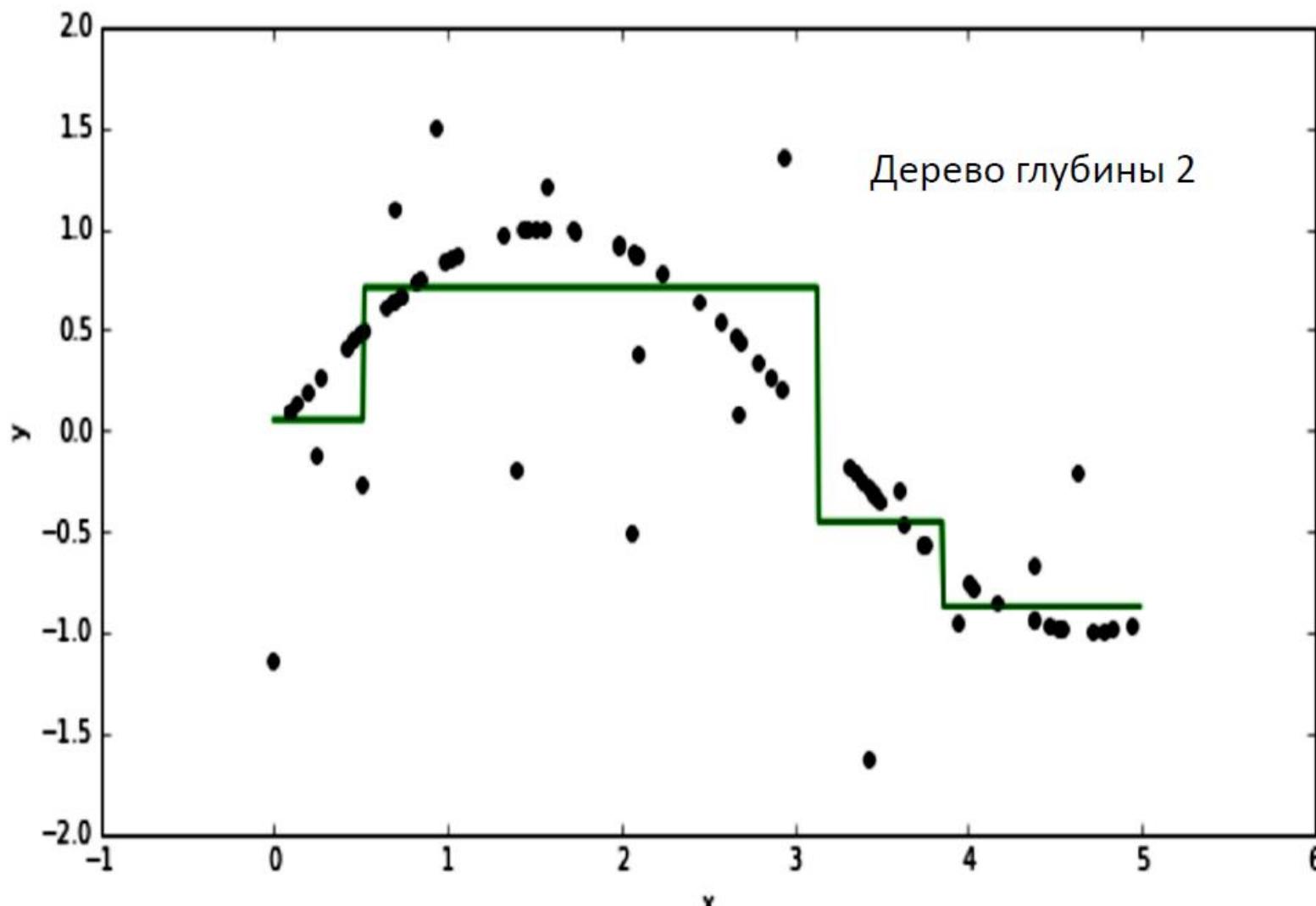
ПРИМЕР ПОСТРОЕНИЯ РЕШАЮЩЕГО ДЕРЕВА В ЗАДАЧЕ РЕГРЕССИИ



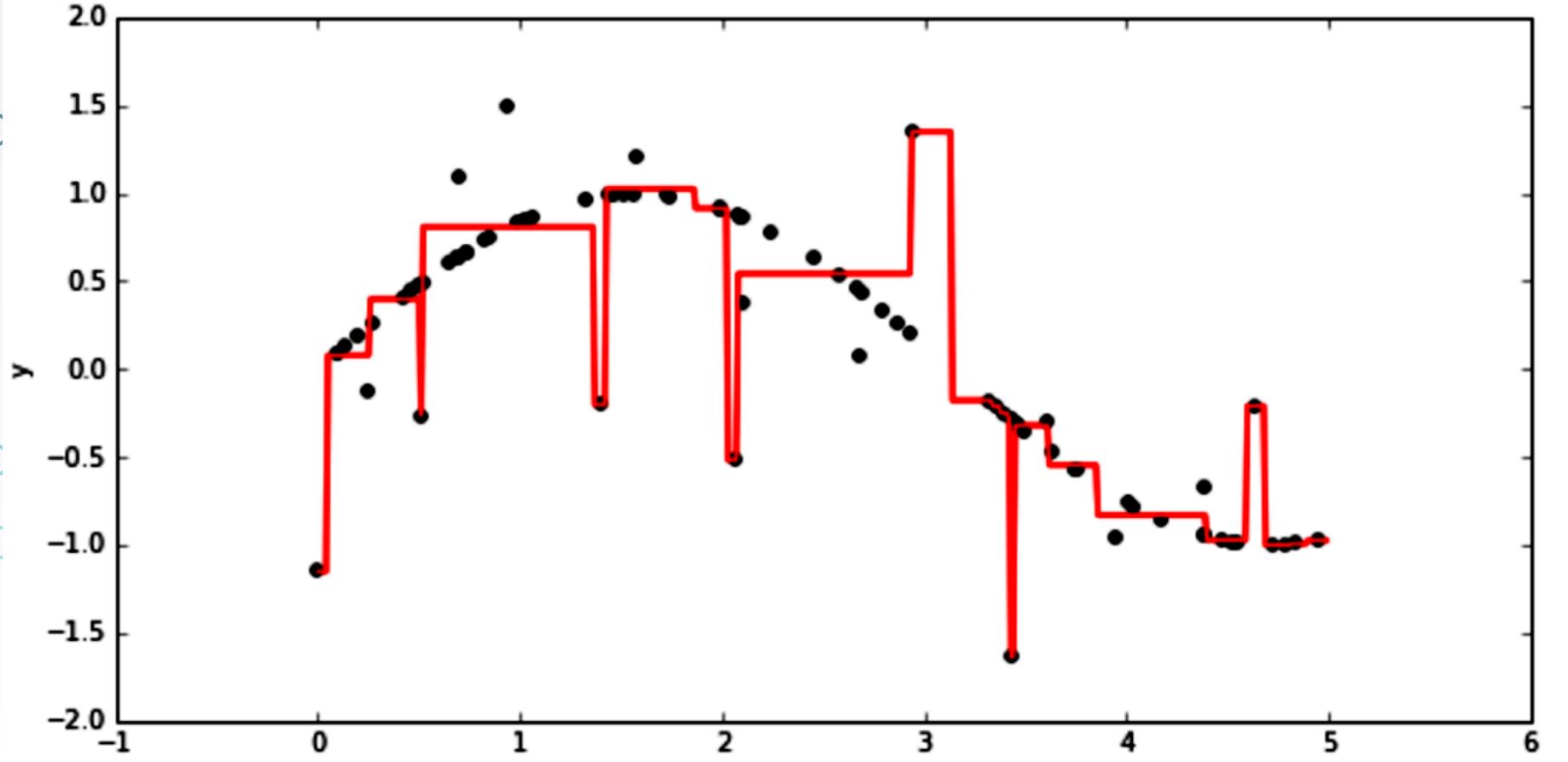
ПРИМЕР ПОСТРОЕНИЯ РЕШАЮЩЕГО ДЕРЕВА В ЗАДАЧЕ РЕГРЕССИИ



ПРИМЕР: ДЕРЕВО ГЛУБИНЫ 2



ПРИМЕР: ДЕРЕВО ГЛУБИНЫ 5



ПЛЮСЫ РЕШАЮЩИХ ДЕРЕВЬЕВ

- Четкие правила классификации (интерпретируемые предикаты, например, “возраст > 25”)
- Деревья решений легко визуализируются, то есть хорошо интерпретируются
- Быстро обучаются и выдают прогноз
- Малое число параметров

МИНУСЫ РЕШАЮЩИХ ДЕРЕВЬЕВ

- Очень чувствительны к шумам в данных, модель сильно меняется при небольшом изменении обучающей выборки
- Разделяющая граница имеет свои ограничения (состоит из гиперплоскостей)
- Необходимость борьбы с переобучением (стрижка или какой-либо из критериев останова)
- Проблема поиска оптимального дерева (NP-полная задача, поэтому на практике используется жадное построение дерева)