МАШИННОЕ ОБУЧЕНИЕ И АНАЛИЗ ДАННЫХ

(Machine Learning and Data Mining)

Н. Ю. Золотых

http://www.uic.unn.ru/~zny/ml

Лекция 14

Ансамбли решающих правил. Баггинг

Рассмотрим задачу классификации на K классов.

$$\mathscr{Y} = \{1, 2, \ldots, K\}.$$

Пусть имеется M классификаторов («экспертов») $f_1, f_2, \ldots, f_M$

$$f_m : \ \mathscr{X} \to \mathscr{Y}, \qquad f_m \in \mathscr{F}, \qquad (m = 1, 2, \ldots, M)$$

Построим новый классификатор:

простое голосование:

$$f(x) = \underset{k=1,\ldots,K}{\operatorname{argmax}} \sum_{m=1}^{M} I\big(f_m(x) = k\big),$$

взвешенное (выпуклая комбинация классификаторов, или «смесь экспертов»):

$$f(x) = \underset{k=1,\ldots,K}{\operatorname{argmax}} \sum_{m=1}^{M} \alpha_m \cdot I\big(f_m(x) = k\big), \qquad \alpha_m \geq 0, \qquad \sum_{m=1}^{M} \alpha_m = 1,$$

или

$$f(x) = \underset{k=1,\ldots,K}{\operatorname{argmax}} \sum_{m=1}^{M} \alpha_m(x) \cdot I\big(f_m(x) = k\big), \qquad \alpha_m(x) \geq 0, \qquad \sum_{m=1}^{M} \alpha_m(x) = 1.$$

В задаче восстановления регрессии

простое голосование:

$$f(x) = \frac{1}{M} \sum_{m=1}^{M} f_m(x),$$

взвешенное голосование («смесь экспертов»):

$$f(x) = \sum_{m=1}^{M} \alpha_m(x) \cdot f_m(x), \qquad \alpha_m(x) \geq 0, \qquad \sum_{m=1}^{M} \alpha_m(x) = 1.$$

Пример: $K = 2$, $M = 3$.

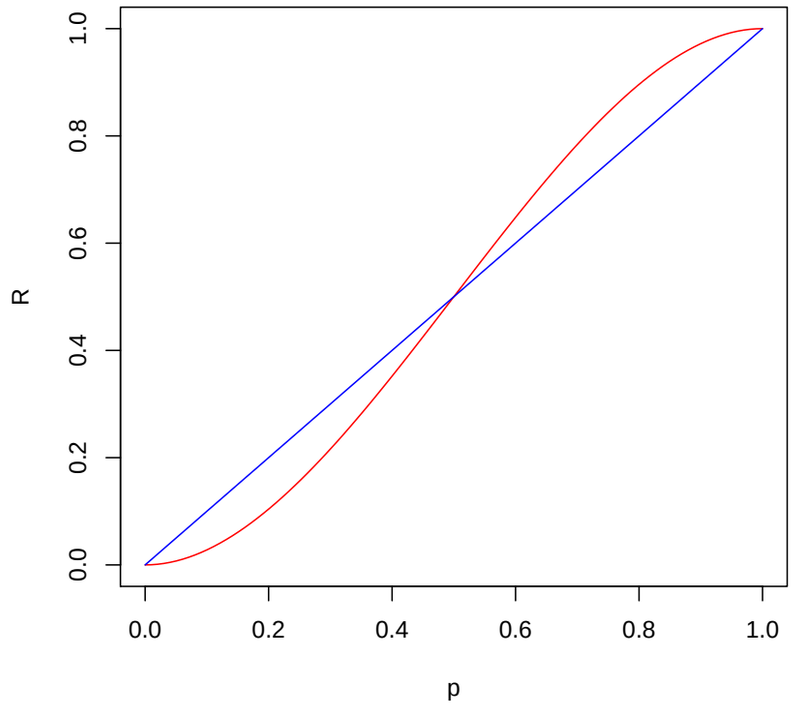
Решение принимается с использованием простого голосования.

Пусть классификаторы независимы (на практике недостижимое требование!).

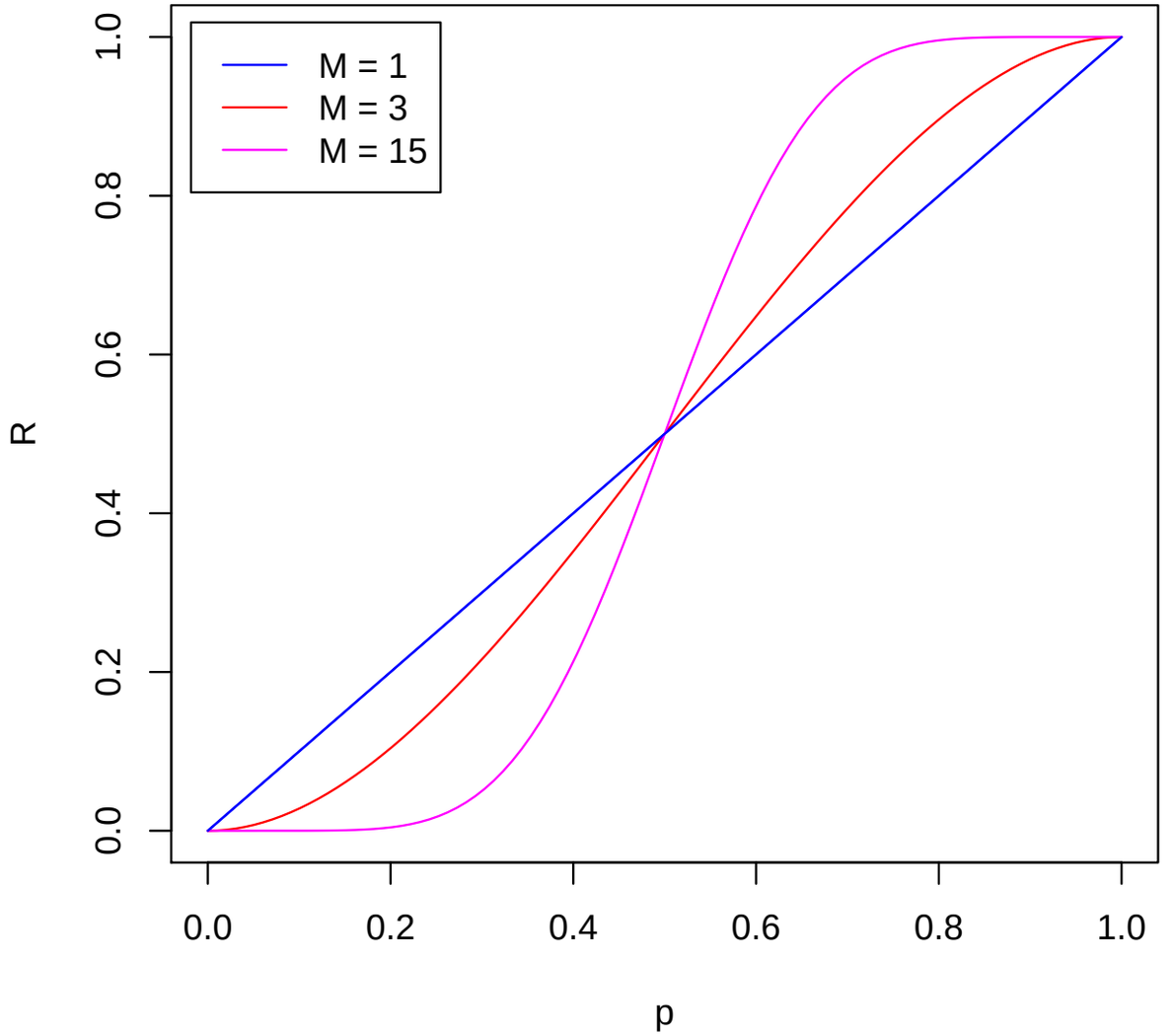
p — вероятность ошибки каждого отдельного классификатора.

Тогда вероятность ошибки общего решения (ожидаемый риск) равен

$$R = p^3 + 3p^2(1-p) = 3p^2 - 2p^3.$$



$$M = 1, 3, 15$$



$K = 2$

- $p > \frac{1}{2}$ — плохой классификатор,
- $p = \frac{1}{2}$ — 50/50,
- $p = \frac{1}{2} - \varepsilon$ — хороший, но слабый классификатор (ε мало),
- $p = \varepsilon$ — сильный классификатор.

Можно ли научиться комбинировать слабые классификаторы, чтобы получить сильный [Kearns, Valiant, 1988]?

Два известных подхода:

- *Баггинг* и т. п.: пытаемся снизить зависимость экспертов друг от друга.
- *Бустинг* и т. п.: эксперты учатся на ошибках других.

14.1. Баггинг

Bagging — от bootstrap aggregation [Breiman, 1994]

Классификатор f_m обучается на bootstrap–выборке (повторной выборке) ($m = 1, 2, \ldots, M$).

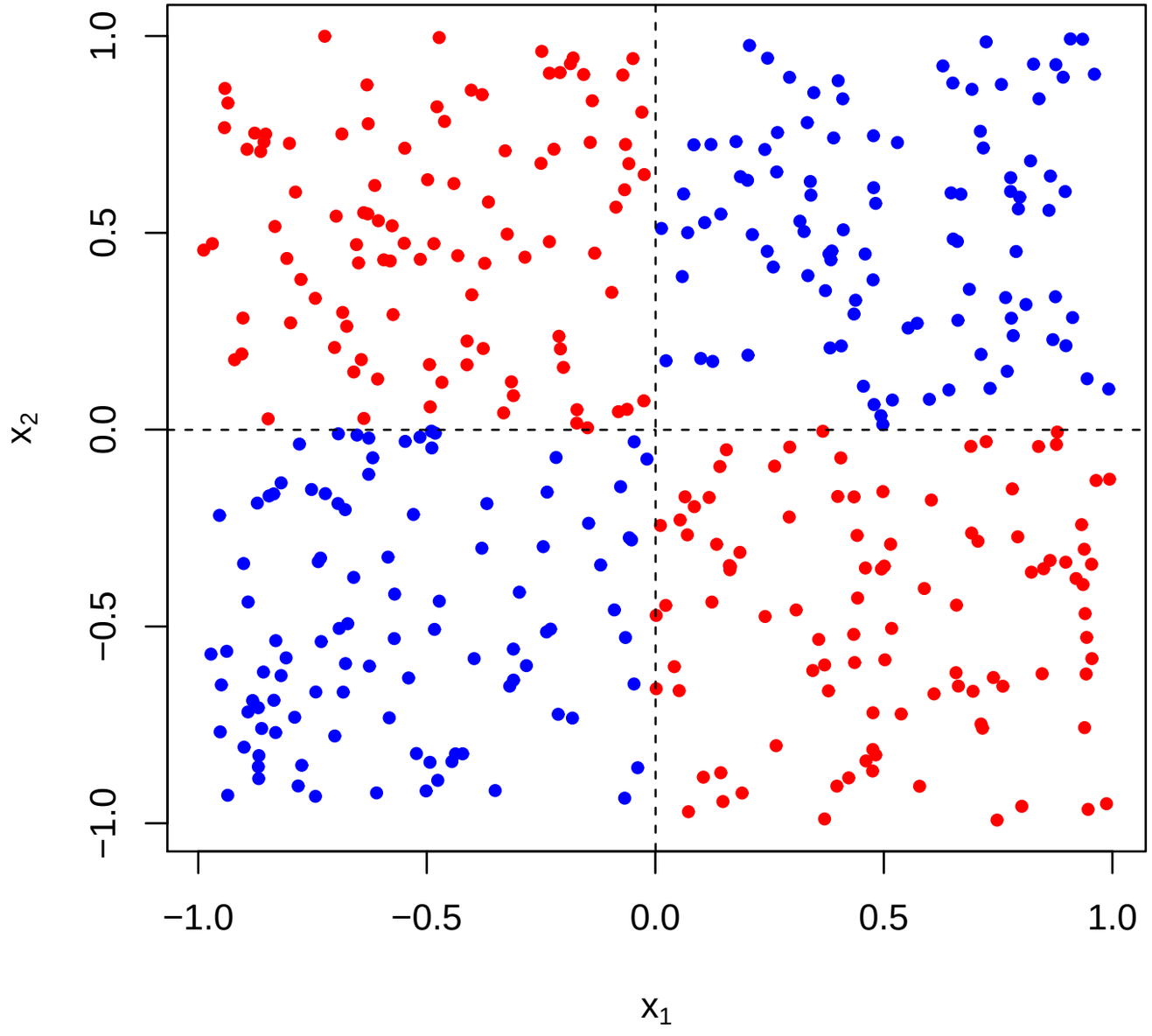
Финальный классификатор f — функция голосования:

для задачи восстановления регрессии:

$$f(x) = \frac{1}{M} \sum_{m=1}^{M} f_m(x),$$

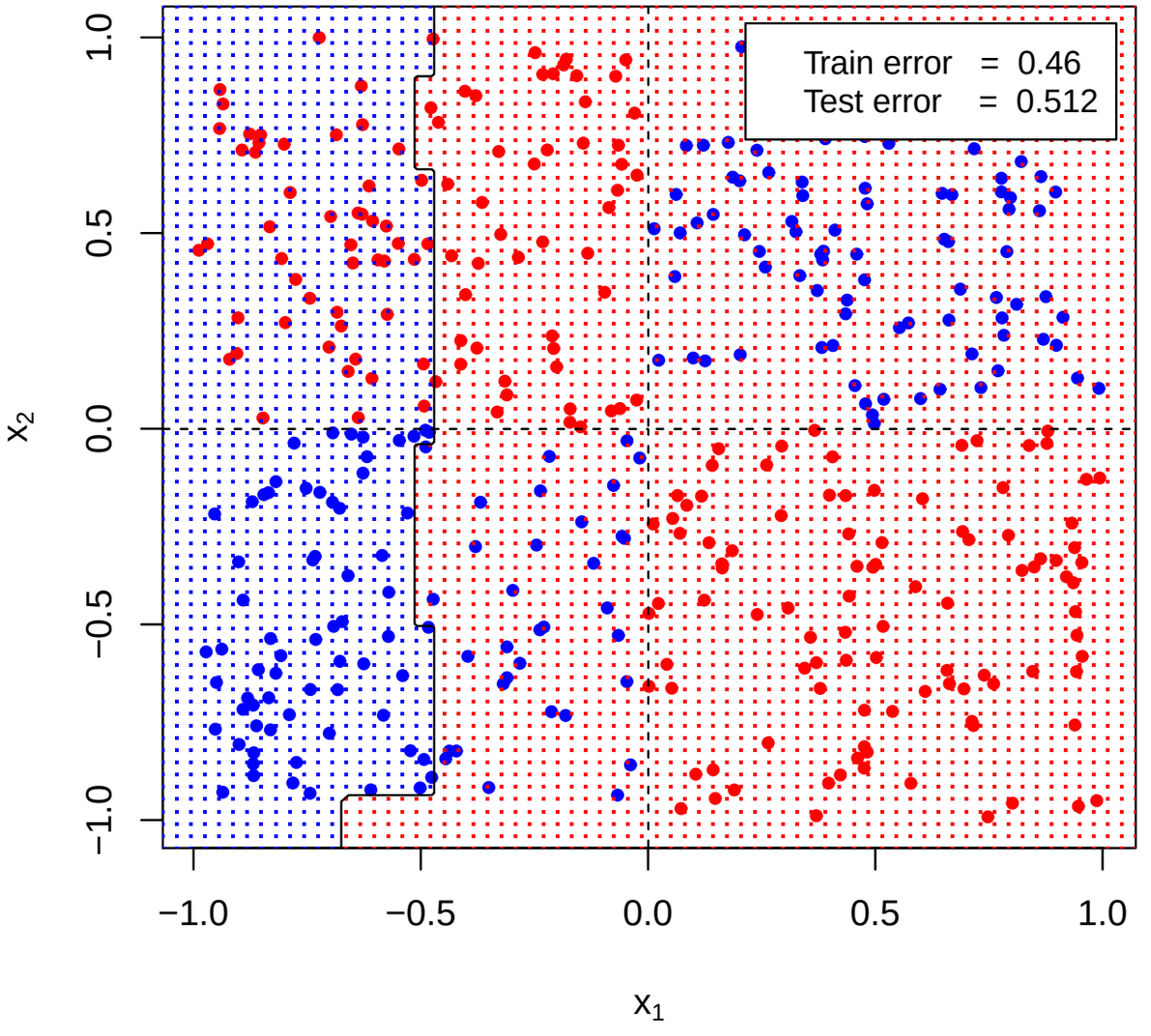
для задачи классификации:

$$f(x) = \underset{k=1,\ldots,K}{\operatorname{argmax}} \sum_{m=1}^{M} I\big(f_m(x) = k\big).$$

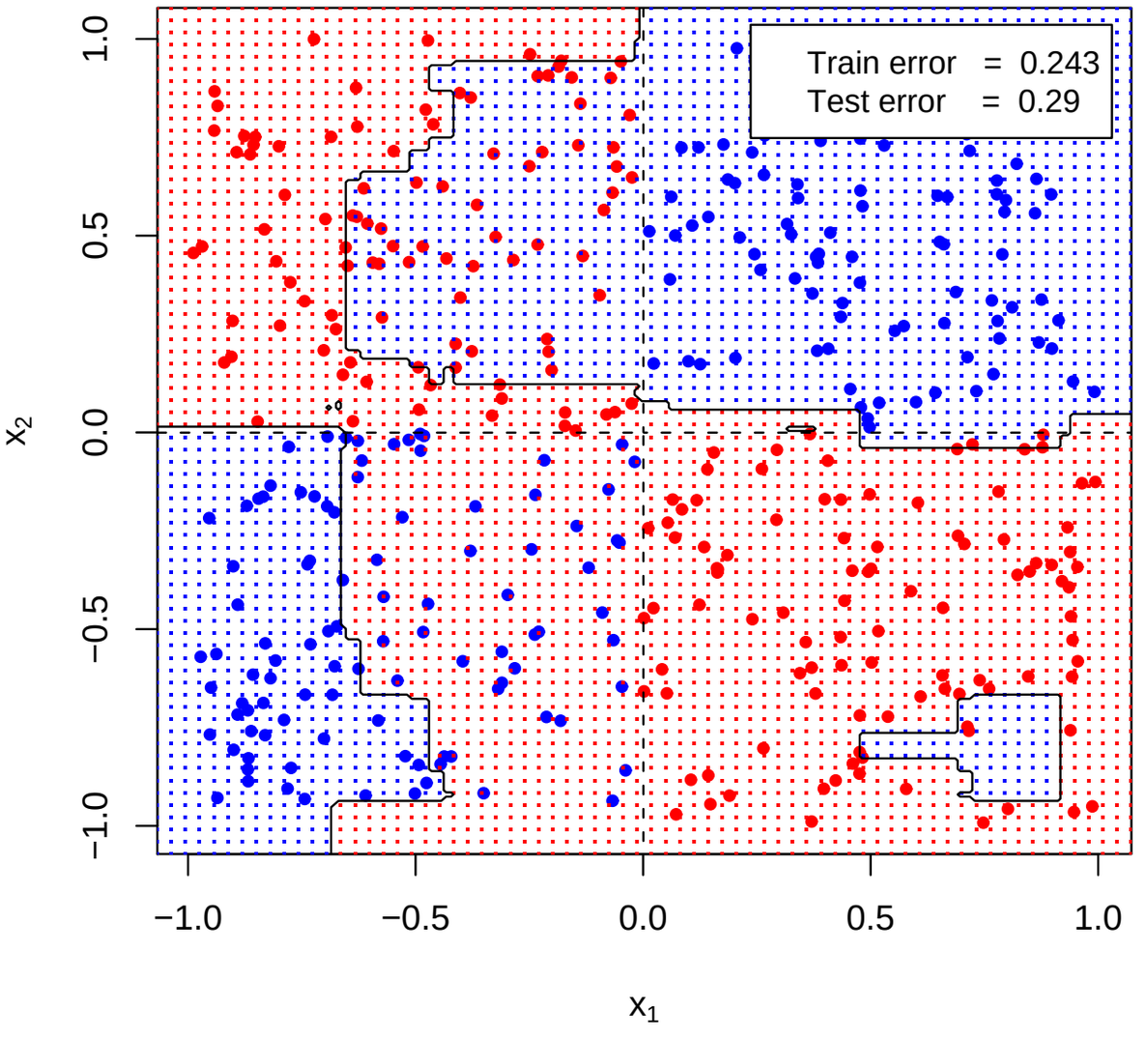

1.0
0.5
0.0
−0.5
−1.0
x_2
−1.0
−0.5
0.0
0.5
1.0
x_1

xor: низких деревьев (высоты 1 или 2) должно быть много.

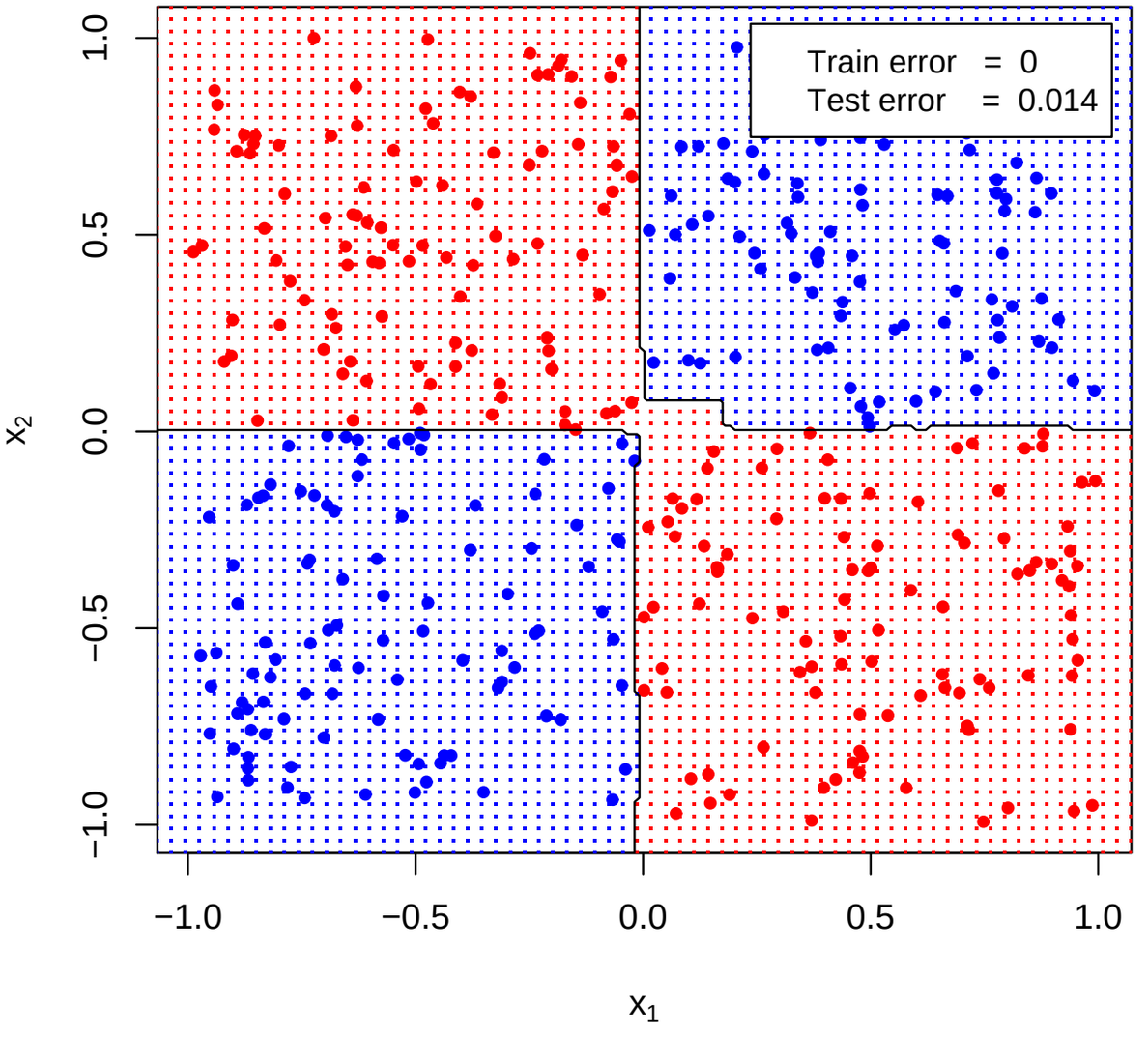
На рис.: деревья решений высоты 1 (100 деревьев).

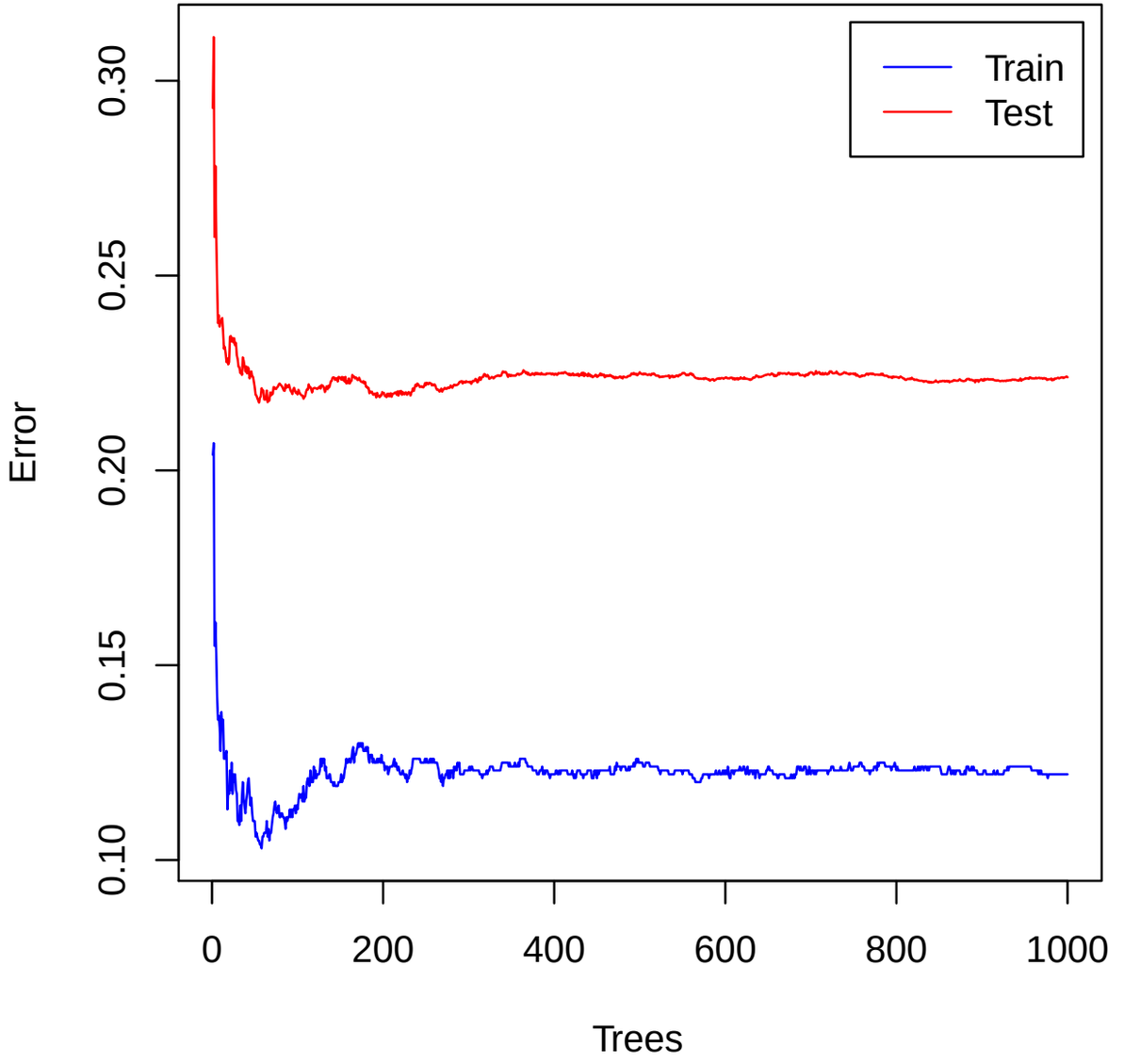


Bagging — деревья решений высоты 2 (100 деревьев)



Bagging — деревья решений высоты 3 (100 деревьев)





Переобучается ли баггинг?

14.2. Случайный лес

Развитие идеи баггинга: Random forests [Breiman, 2001]

Ансамбль параллельно обучаемых «независимых» деревьев решений.

Независимое построение определенного количества M (например, 500) деревьев:

- генерация случайной bootstrap-выборки из обучающей выборки (50–70% от размера всей обучающей выборки);
- построение дерева решений по данной подвыборке: в каждом новом узле дерева переменная для разбиения выбирается не из всех признаков, а из случайно выбранного их подмножества небольшой мощности.

procedure RandomForests

for $m = 1, 2, \ldots, M$

begin

По обучающей выборке построить бутстрэп-выборку

Построить дерево f_m, рекурсивно применяя следующую процедуру,

пока не будет достигнут минимальный размер sz:

begin

Построить случайный набор из δ признаков

Выбрать из него лучшую переменную и построить 2 дочерних узла

end

end

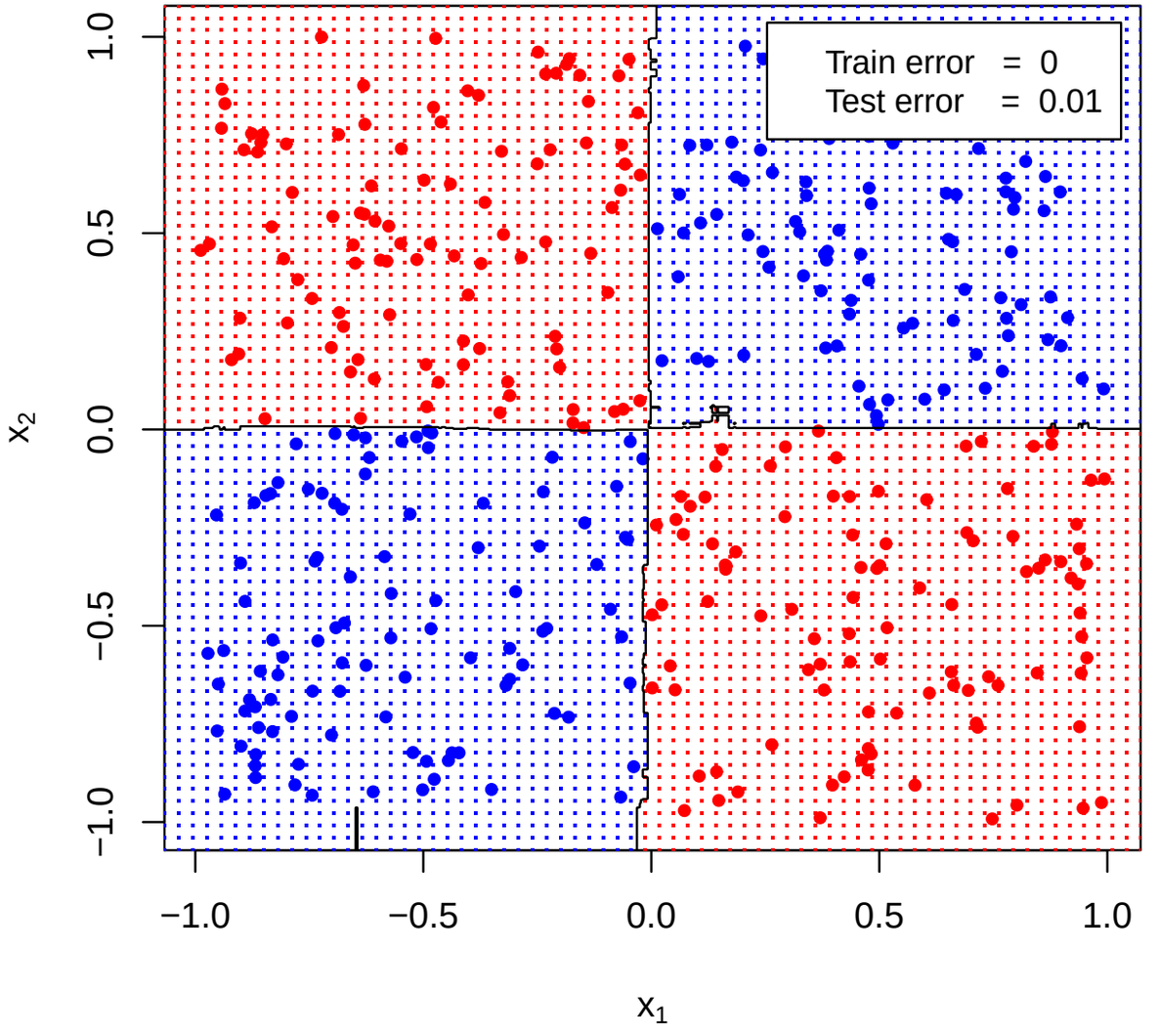
Для задачи восстановления регрессии **return** $f = \frac{1}{M} \sum\limits_{m=1}^{M} f_m$

Для задачи классификации **return** $f = \underset{k}{\mathrm{argmax}} \sum\limits_{m=1}^{M} I(f_m = k)$

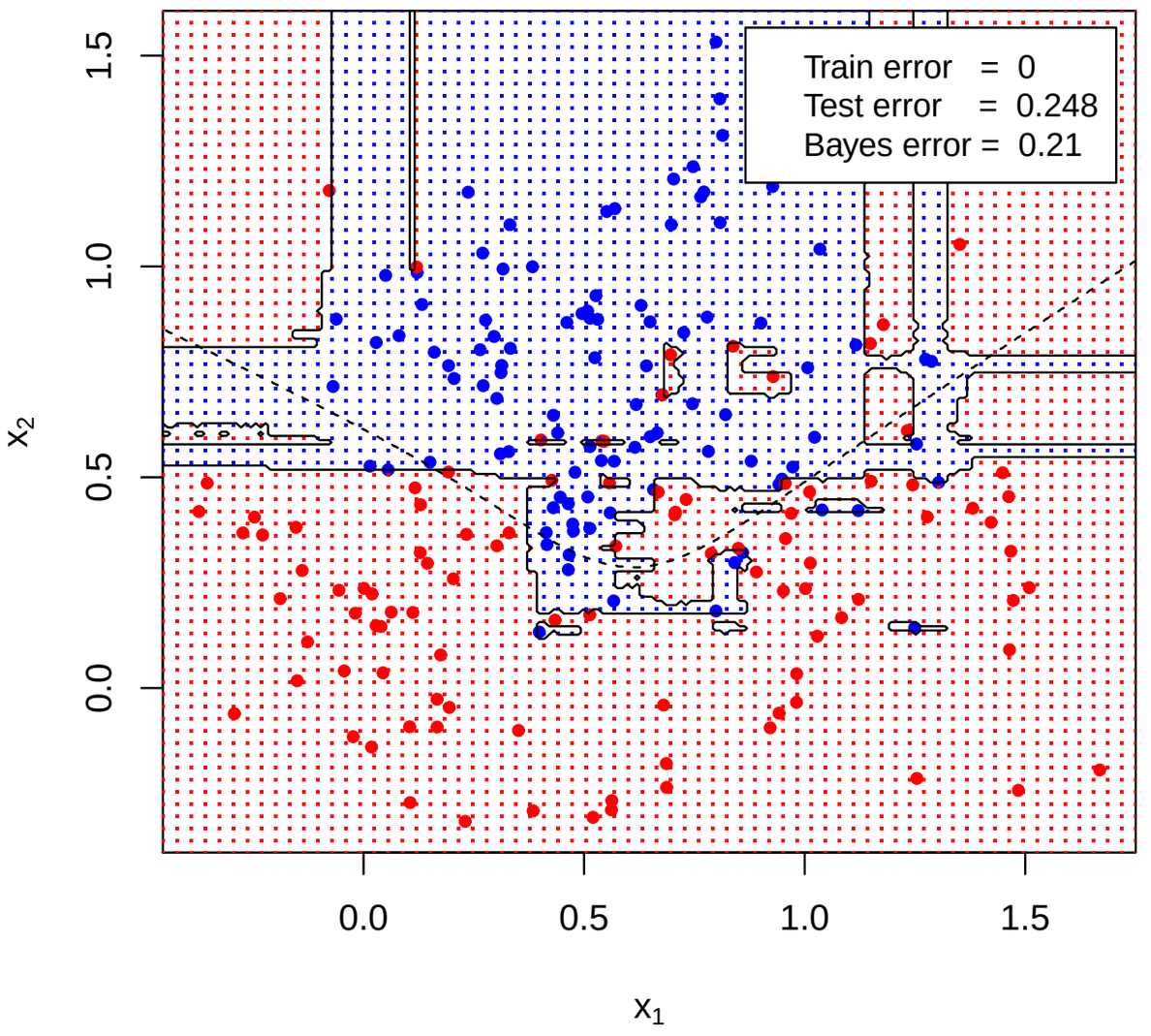
end

Для регрессии, например, $\delta = \sqrt{d}$, $\mathrm{sz} = 3$. Для классификации, например, $\delta = d/3$, $\mathrm{sz} = 1$

Random Forest (50 деревьев)

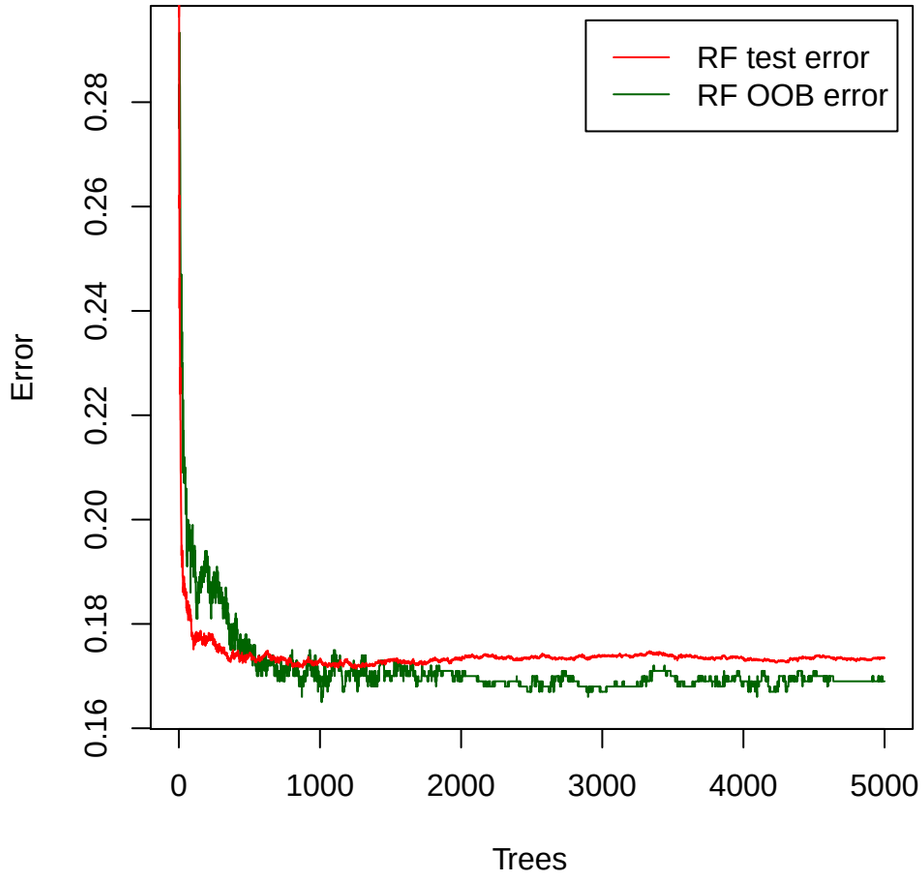


Random forest: 500 деревьев



Out of bag error

OOB ошибка — средняя ошибка предсказаний классификатора (RF) на обучающей выборке, усредненная по всем слабым классификаторам (деревьям), при обучении которых данных объект не вошел в бутстрэп-выборку.



Extremely randomized trees

[Geurts, Ernst, Wehenkel, 2006]

Основные отличия от Random Forests:

- Бутстрэп-выборки не используются. Для обучения каждого дерева используется целиком все обучающее множество.
- При ветвлении, как и в RF, выбираем случайное подмножество (неконстантных в данном ящике) признаков. Затем для каждой такое переменной строим случайное разбиение и выбираем из них лучшее.