

The background of the slide is a photograph of the main building of Moscow State University, featuring its iconic Spasskaya Tower with a tall spire. The building is set against a blue sky with light, wispy clouds. In the foreground, there are dark, leafless trees and some lower-level urban buildings.

Прикладные задачи анализа данных

**Ансамбли алгоритмов машинного
обучения**

Дьяконов А.Г.

**Московский государственный университет
имени М.В. Ломоносова (Москва, Россия)**

Ансамбль алгоритмов (Ensemble)

- алгоритм, который состоит из нескольких алгоритмов машинного обучения (**базовых алгоритмов** – **base learners**)

Простой ансамбль в регрессии:

$$a(x) = \frac{1}{n} (b_1(x) + \dots + b_n(x))$$

Простой ансамбль в классификации:

$$a(x) = \text{mode}(b_1(x), \dots, b_n(x))$$

В чём может быть усложнение?

Ансамбль алгоритмов

$$a(x) = b(b_1(x), \dots, b_n(x))$$

b – мета-алгоритм (**meta-estimator**),

b_i – базовые алгоритмы (**base learners**)
в бустинге – слабые (**weak**)

Теоретическое обоснование

Почему несколько алгоритмов лучше одного...

Ошибка суммы регрессоров

Если ответы регрессоров на объекте – независимые случайные величины с одинаковым матожиданием и дисперсией

$$\xi = \frac{1}{n} (\xi_1 + \dots + \xi_n)$$

$$E\xi = \frac{1}{n} (E\xi_1 + \dots + E\xi_n) = E\xi_i$$

$$D\xi = \frac{1}{n^2} (D\xi_1 + \dots + D\xi_n) = \frac{D\xi_i}{n}$$

ДЗ А если есть корреляция между базовыми алгоритмами?
решите в постановке, что корреляция между любыми двумя
алгоритмами равна ρ

Ошибка комитета большинства

Пусть три (независимых) классификатора на два класса с вероятностью ошибки p

Пусть верный ответ – 0

$$\left. \begin{array}{ll} (0,0,0) & (1-p)(1-p)(1-p) \\ (1,0,0) & p(1-p)(1-p) \\ (0,1,0) & (1-p)p(1-p) \\ (0,0,1) & (1-p)(1-p)p \end{array} \right\}$$

верный ответ

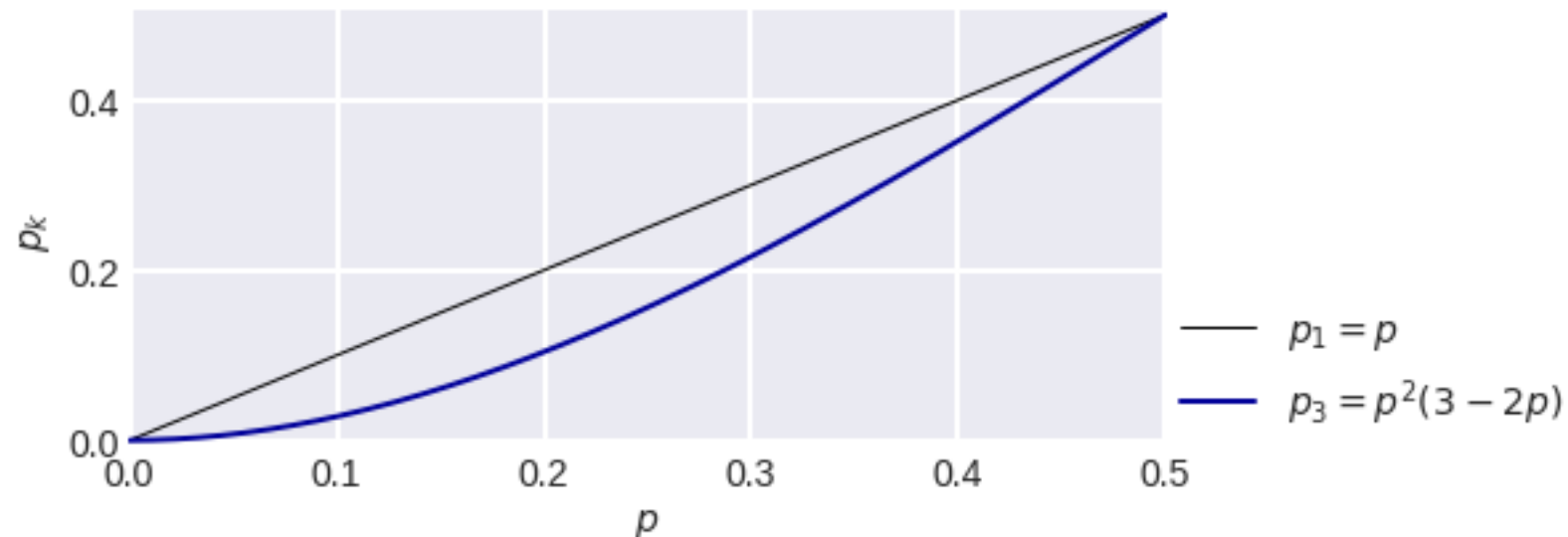
$$\left. \begin{array}{ll} (1,1,1) & ppp \\ (1,1,0) & pp(1-p) \\ (0,1,1) & (1-p)pp \\ (1,0,1) & p(1-p)p \end{array} \right\}$$

ошибка

вероятность ошибки

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

Ошибка комитета большинства



При малых p ошибка комитета очень мала!

При $p = 0.2$ – почти в два раза меньше

Ошибка комитета большинства

Общий случай:

$$\sum_{t=0}^{\lfloor n/2 \rfloor} C_n^t (1-p)^t p^{n-t} \leq e^{-\frac{1}{2}n(2p-1)^2}$$

неравенство Хэфдинга

**Ошибка экспоненциально снижается
с увеличением числа базовых алгоритмов...
но это в теории**

На практике

Классификаторы / регрессоры точно не являются независимыми

Почему?

На практике

Классификаторы / регрессоры точно не являются независимыми

Почему?

- Решают одну задачу
- Настраиваются на один целевой вектор
- Могут быть из одной модели (ну, 2-3 разных)!

Выход

Пытаться делать алгоритмы разнообразными

**Пусть алгоритмы ошибаются, но по-разному:
ошибки одних компенсируются правильными ответами других**

Пример: подвыборки и подмножества признаков в RF

Ещё почему алгоритмы в ансамбле должны быть разными

одинаковые

1111110000 $q=0.6$

1111110000 $q=0.6$

1111110000 $q=0.6$

$q_{ens} = 0.6$

похожие

1111110000 $q=0.6$

1111101000 $q=0.6$

1111100100 $q=0.6$

$q_{ens} = 0.5$

разные

1010010111

1100110011

1111110000

1110110011

$q_{ens} = 0.7$

Нет ли здесь обмана?

Выход

Пытаться делать алгоритмы разнообразными

**Пусть алгоритмы ошибаются, но по-разному:
ошибки одних компенсируются правильными ответами других**

Пример: подвыборки и подмножества признаков в RF

Ещё почему алгоритмы в ансамбле должны быть разными

одинаковые

1111110000 $q=0.6$

1111110000 $q=0.6$

1111110000 $q=0.6$

$q_{ens} = 0.6$

похожие

1110111000 $q=0.6$

1101110100 $q=0.6$

1011101100 $q=0.6$

$q_{ens} = 0.8$

разные

1010010111

1100110011

1111110000

1110110011

$q_{ens} = 0.7$

Нет ли здесь обмана?

Повышения разнообразия

- **«варьирование» обучающей выборки**
(бэггинг)
- **«варьирование» признаков**
(Random Subspaces)
- **«варьирование» целевого вектора**
(ЕСОС, $f(y)$)
- **«варьирование» моделей**
(стекинг)
- **«варьирование» в модели**
(разные алгоритмы в рамках одной – специальные НС,
рандомизация в алгоритме – случайный лес, инициализация в
НС)

Обоснования применения ансамблей

**Статистическое
(Statistical)**

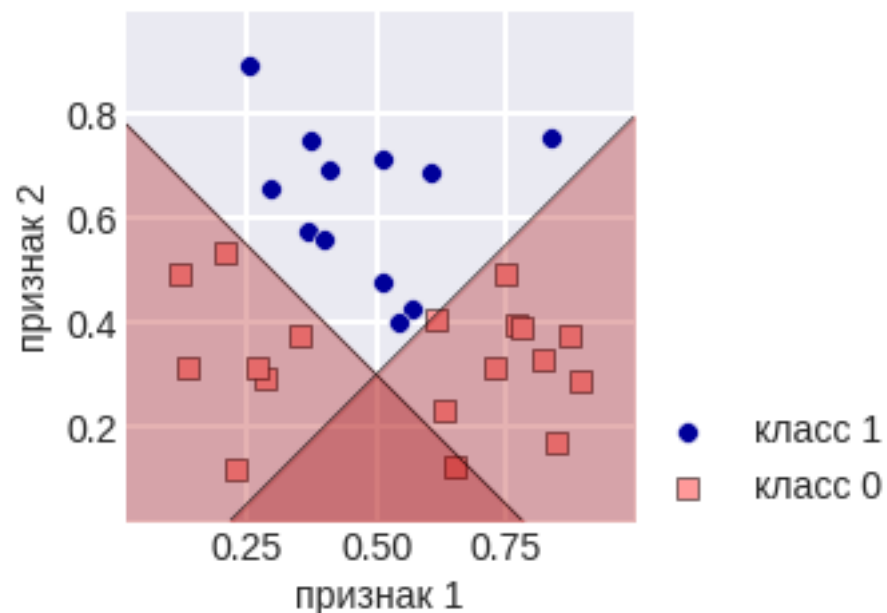
**Вычислительное
(Computational)**

**Функциональное
(Representational)**

– ошибка может быть меньше

– обучение = оптимизация функции,
а ансамбль «распараллеливает» процесс

– можно представить функции, которые
нельзя было с помощью базовых
алгоритмов



Ансамбли

- **комитеты (голосование) / усреднение**
в том числе, усреднение по Коши,
калибровка + усреднение
- **ECOC (error-correcting output coding)**
кодирование целевого вектора

- **стекинг (stacking)**

построение метапризнаков — ответы алгоритмов на объектах обучающей (под)выборки, обучение на них нового алгоритма

- **бэгинг (bagging)**

усреднение моделей обученных на бутстреп-подвыборках
+ обобщения (RF)

- **бустинг (boosting)**

веса объектов обучения, обучение на взвешенной выборке,
увеличение весов неверно классифицированных объектов

- **«ручные методы»**

Комитеты (голосование, Voting Ensembles)

голосование по большинству (Majority vote)

$$a(x) = \text{mode}(b_1(x), \dots, b_n(x))$$

комитеты единогласия

в бинарной задаче классификации –

$$a(x) = \min(b_1(x), \dots, b_n(x))$$

обнаружение аномалий:

мата-алгоритм – максимум

«тревога при малейшем подозрении»

$$a(x) = \max(b_1(x), \dots, b_n(x))$$

Усреднение

«среднее арифметическое»

$$a(x) = \frac{1}{n} (b_1(x) + \dots + b_n(x))$$

+ любые другие средние (ех: по Колмогорову)

$$a(x) = \frac{1}{n} f^{-1} (f(b_1(x)) + \dots + f(b_n(x)))$$

Ранговое усреднение (Rank Averaging)

$$a(x) = \frac{1}{n} (\text{rank}(b_1(x)) + \dots + \text{rank}(b_n(x)))$$

ориентировано на конкретный AUC ROC

Усреднение / голосование с весами (weighted averaging)

$$a(x) = \frac{1}{w_1 + \dots + w_n} (w_1 \cdot b_1(x) + \dots + w_n \cdot b_n(x))$$

Усреднение

Feature-Weighted Linear Stacking

Области компетентности алгоритмов – линейные регрессии

$$a(x) = w_1(x) \cdot b_1(x) + \dots + w_n(x) \cdot b_n(x) =$$

$$= \sum_t \left(\sum_i w_{ti} x_i \right) b_t(x) = \sum_{t,i} w_{ti} x_i b_t(x)$$

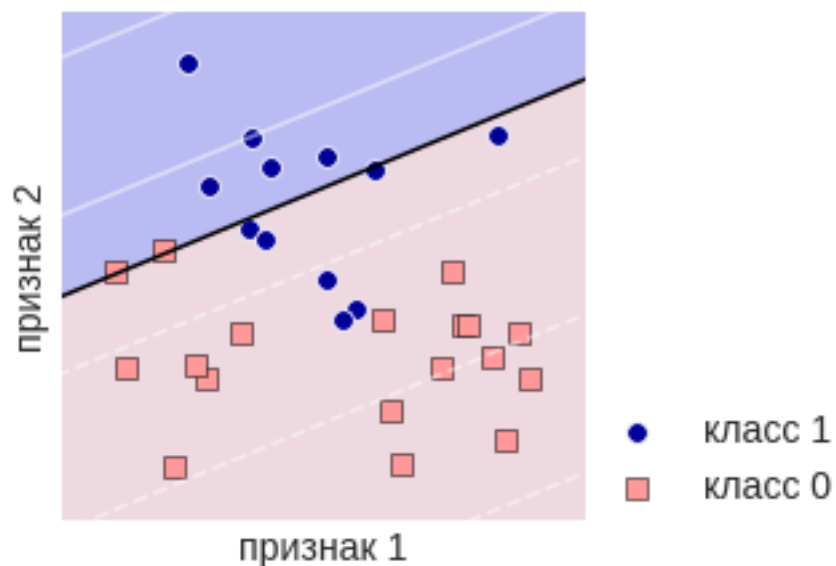
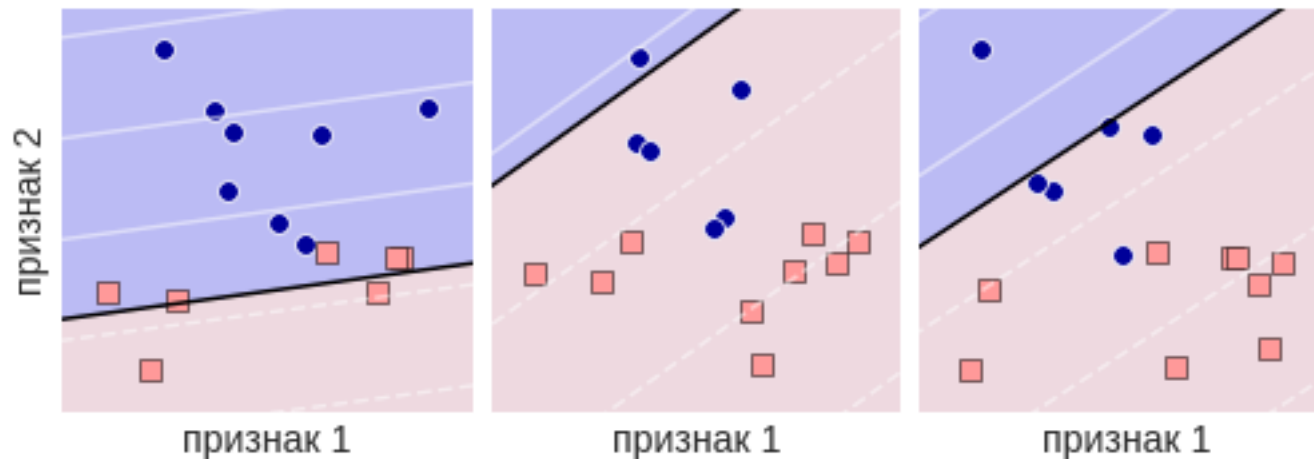
Бэгинг (Bagging)
– **bootstrap aggregating**

Каждый базовый алгоритм настраивается на случайной подвыборке обучения

Бэгинг	Подвыборка обучающей выборки берётся с помощью бутстрепа
Пэстинг (Pasting)	Случайная обучающая подвыборка.
Случайные подпространства (Random Subspaces)	Случайное подмножество признаков
Случайные патчи (Random Patches)	Одновременно берём случайное подмножество объектов и признаков
cross-validated committees	k обучений на (k-1)-м фолде

Особенности ансамблирования

Не всегда получается «как было задумано»...



```
model =  
BaggingClassifier(base_estimator=LogisticRegression(),  
                  n_estimators=100,  
                  max_samples=1.0,  
                  max_features=1.0,  
                  bootstrap=True,  
                  bootstrap_features=False,  
                  oob_score=False,  
                  warm_start=False,  
                  n_jobs=None,  
                  random_state=None,  
                  verbose=0)
```

```
model.fit(X, y)
```

Бэггинг (Bagging)

1. Цикл по t (номер базового алгоритма)

1.1. Взять подвыборку $[X', y']$ обучающей выборки $[X, y]$

1.2. Обучить t -й базовый алгоритм на этой подвыборке:

$$b_t = \text{fit}(X', y')$$

2. Ансамбль

$$a(x) = \frac{1}{n} (b_1(x) + \dots + b_n(x))$$

(для задач регрессии).

Вероятность отбора при бутстрепе (при $m \rightarrow \infty$): $1 - \frac{1}{e} \approx 0.632$

**Каждый базовый алгоритм обучается ~ на 63% данных
Остальные называются – out-of-bag-наблюдениями (OOB)**

~процедура снижения variance в статистическом обучении

OOB-prediction

На OOB-части выборки можно получить ответы алгоритма

Пусть на i -й итерации это часть: OOB_i

и мы построили алгоритм b_i

OOB-ответы бэггинга (OOB-prediction)

$$a_{\text{OOB}}(x_j) = \frac{1}{|\{i : x_j \in \text{OOB}_i\}|} \sum_{i: x_j \in \text{OOB}_i} b_i(x_j)$$

**Можно вычислить
OOB-ошибку бэггинга**

**хорошая оценка ошибки на тесте
похожа на CV-ошибку...**

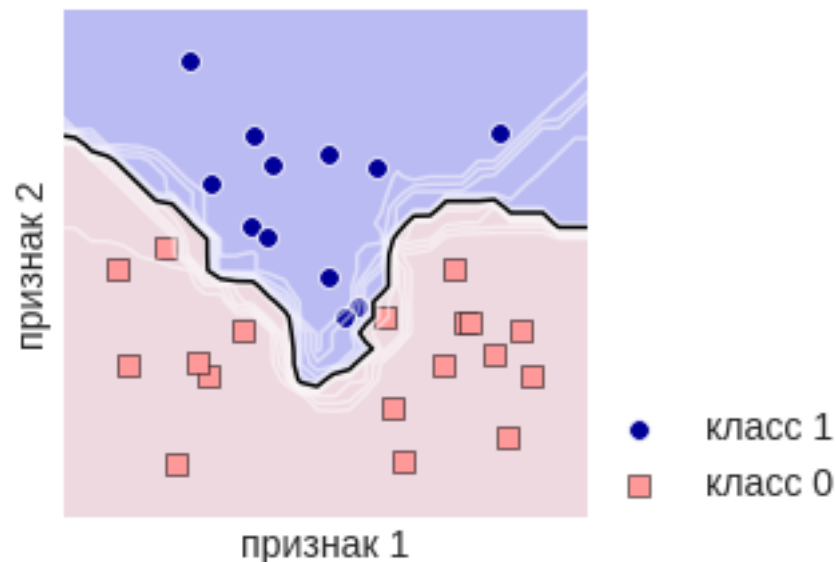
Устойчивость (stable learners) модели –

**незначительное изменение оптимальных параметров при
взятии подвыборки (SVM, kNN $k > 3$)**

**В бэггинге используются неустойчивые модели
(high variance)!**

Но несмещённые (small bias)!

Но тут нет хороших теоретических результатов...



**Пример – если выбрать
правильную базовую модель
для бэггинга**

Здесь – kNN(1)

Идеи для бэггинга...

Часто признаки делятся / можно разделить на группы:

- по источнику данных (БКИ1, БКИ2, ...)
- по типу признака (вещественный, категориальный, ...)
 - по кодированию (ONE, hash, label, ...)
- по способу агрегирования (PCA, t-SNE, кластеризация,...)

Иногда объекты:

- по источнику данных
 - по времени
- по значениям каких-то признаков (в том числе по кластерам)

– эти деления можно использовать при формировании подвыборок... вопрос – **как?**

Случайный лес (Random Forest)

**дальнейшие улучшения независимости базовых
классификаторов**

бэггинг + случайности при построении деревьев

отдельная лекция

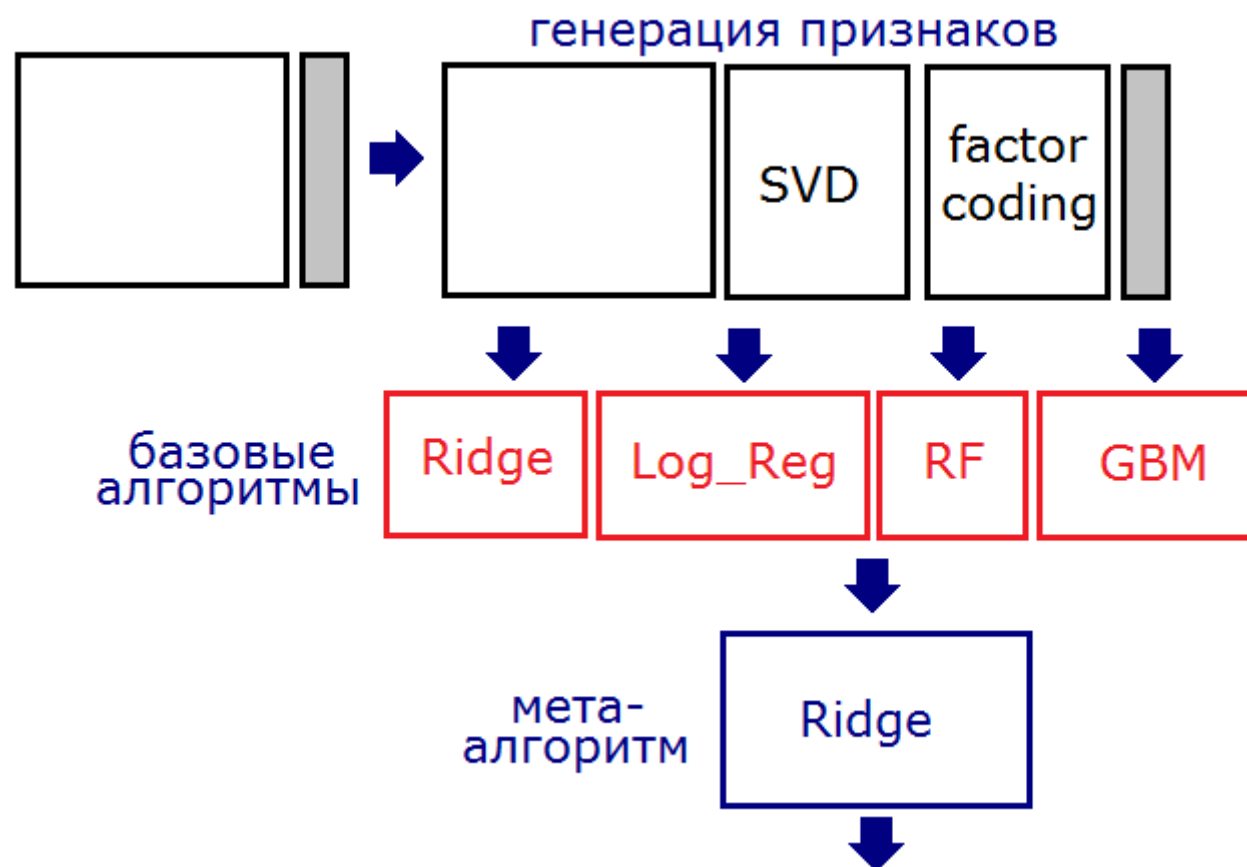
Стекинг (stacking)

Идея: хорошо усреднять алгоритмы, но почему именно усреднять?

$$a(x) = b(b_1(x), \dots, b_n(x))$$

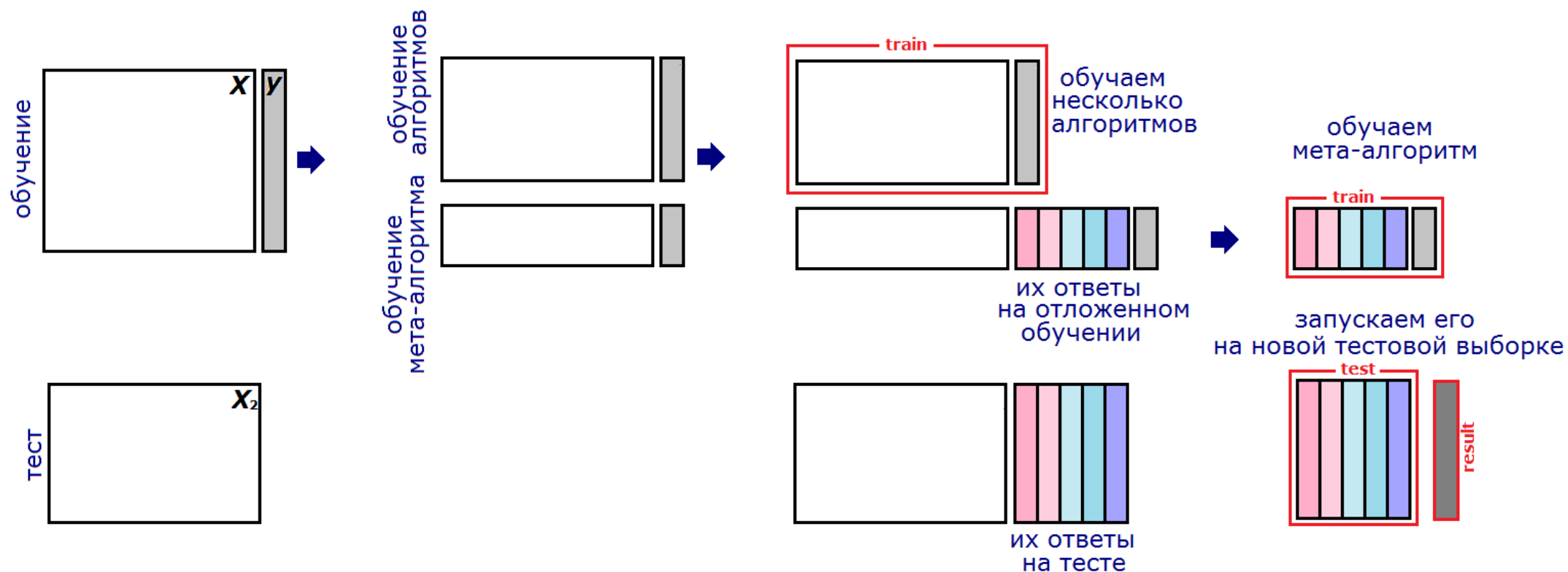
**b – мета-алгоритм,
который нужно отдельно настроить!**

Стекинг



**Используем ответы алгоритмов как признаки
для нового мета-алгоритма машинного обучения**

Блендинг (простейшая форма стекинга)



Блендинг

– термин введён победителями конкурса Netflix

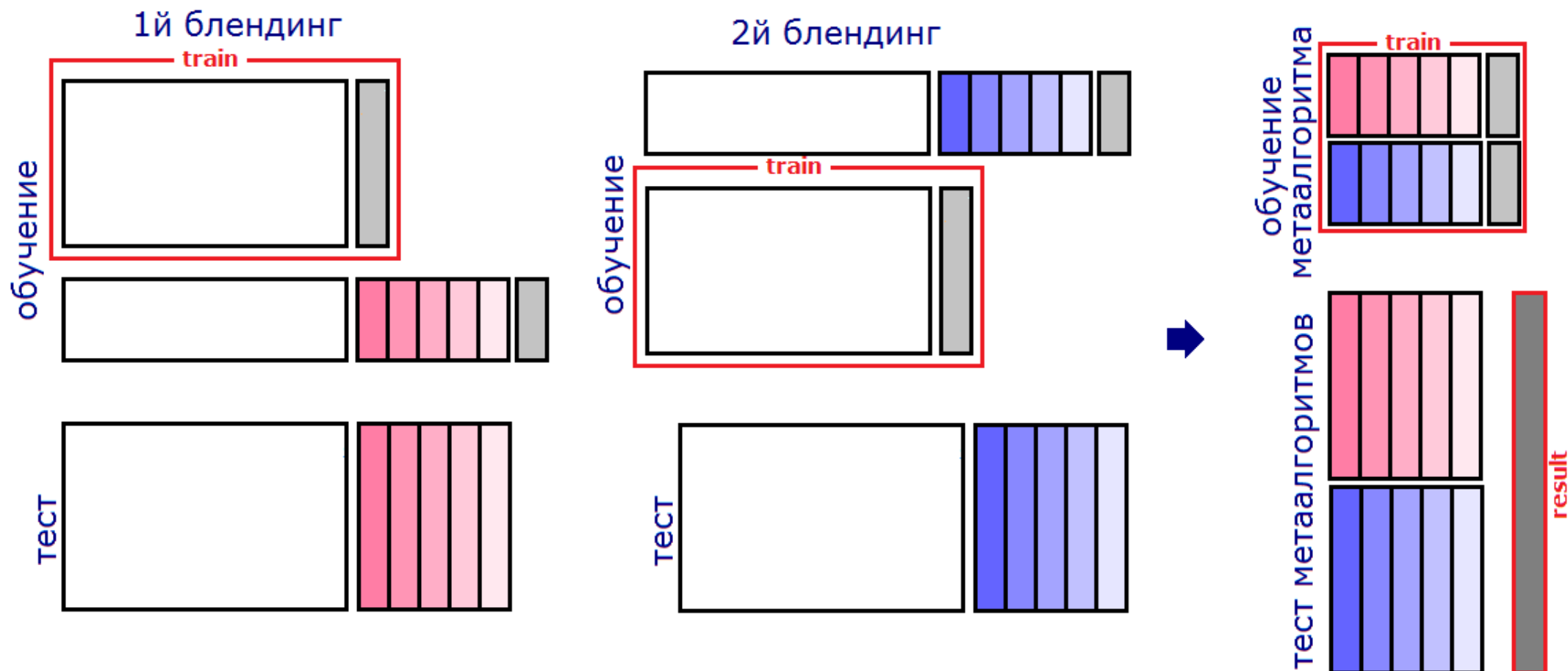
Сейчас блендингом называются простейшие формы стекинга

Недостатки

Используется не вся обучающая выборка

- можно усреднить несколько стекингов
- можно «состыковать»

Блендинг

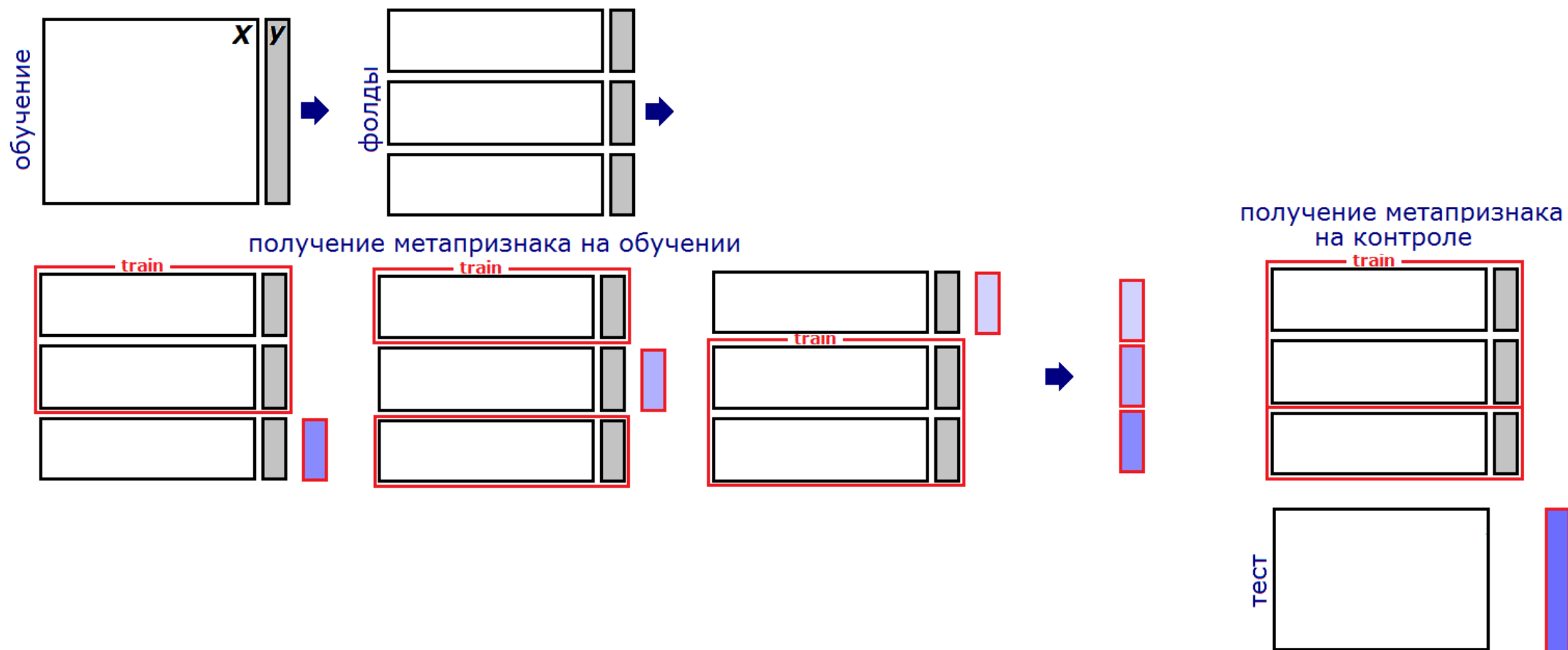


Состыковка таблиц

- Долго и не всегда лучше по качеству
- Ответы всё равно надо будет усреднить

Стекинг

Хотим использовать всю обучающую выборку



Стекинг

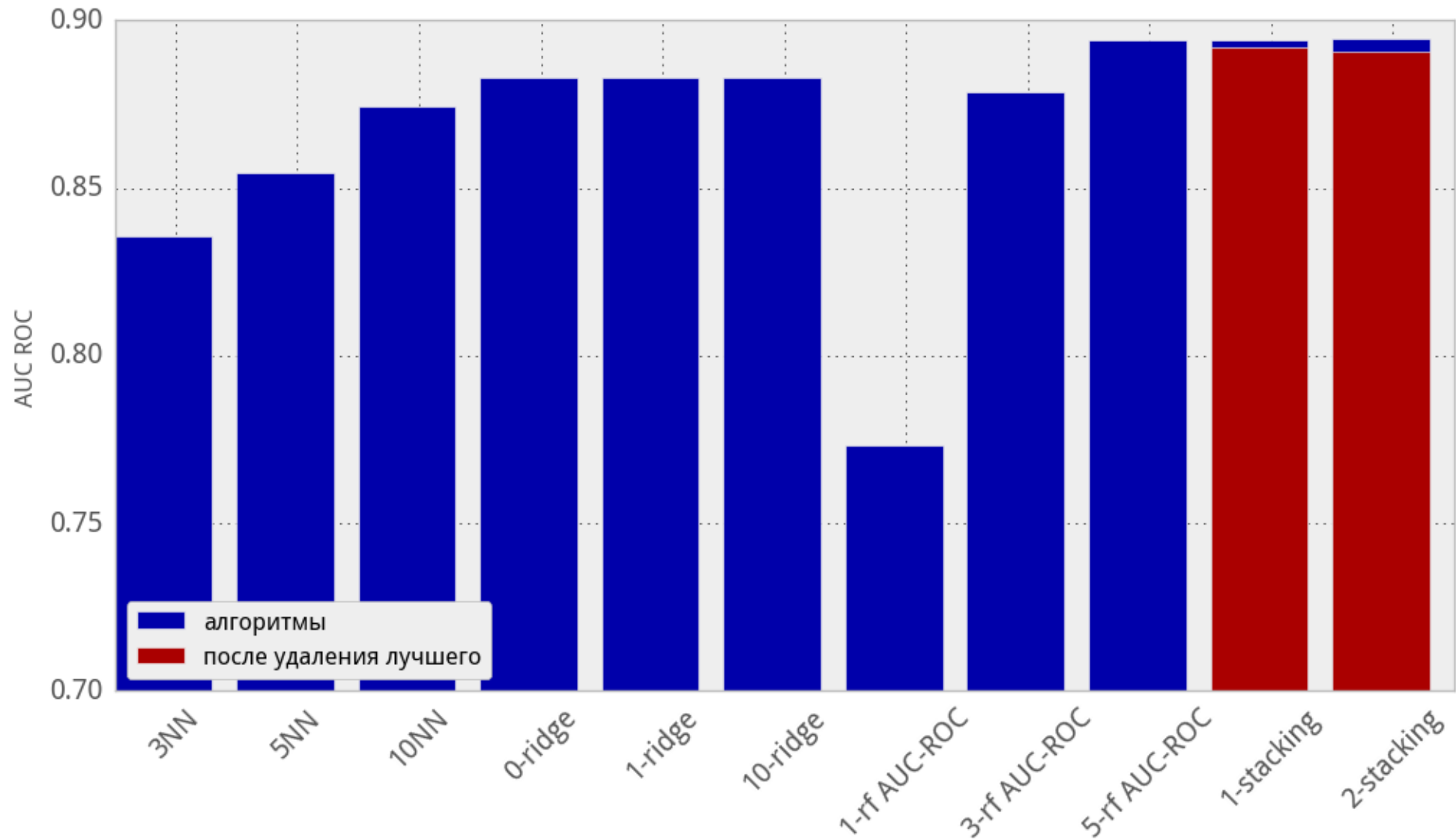
также можно брать разные разбиения и усреднять

Недостаток

Метапризнаки на обучении и тесте разные!

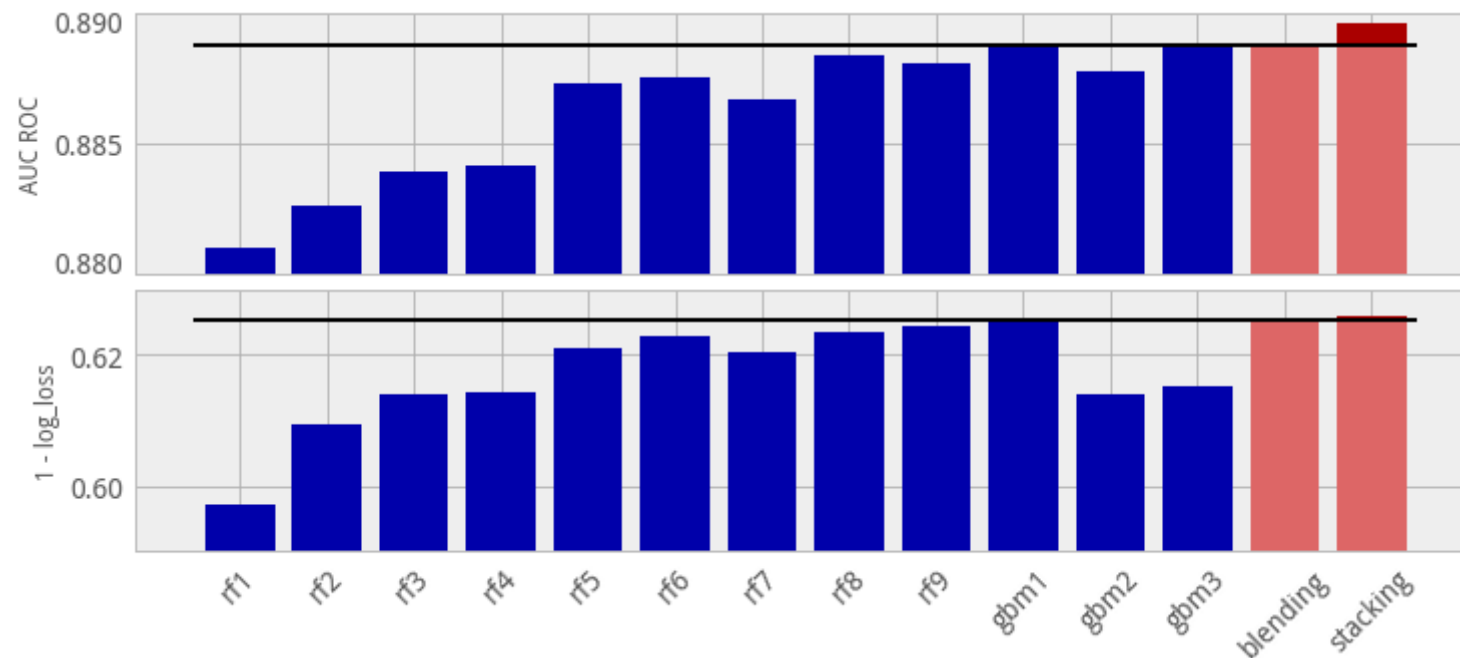
- регуляризация
- нормальный шум к метапризнакам

Стекинг



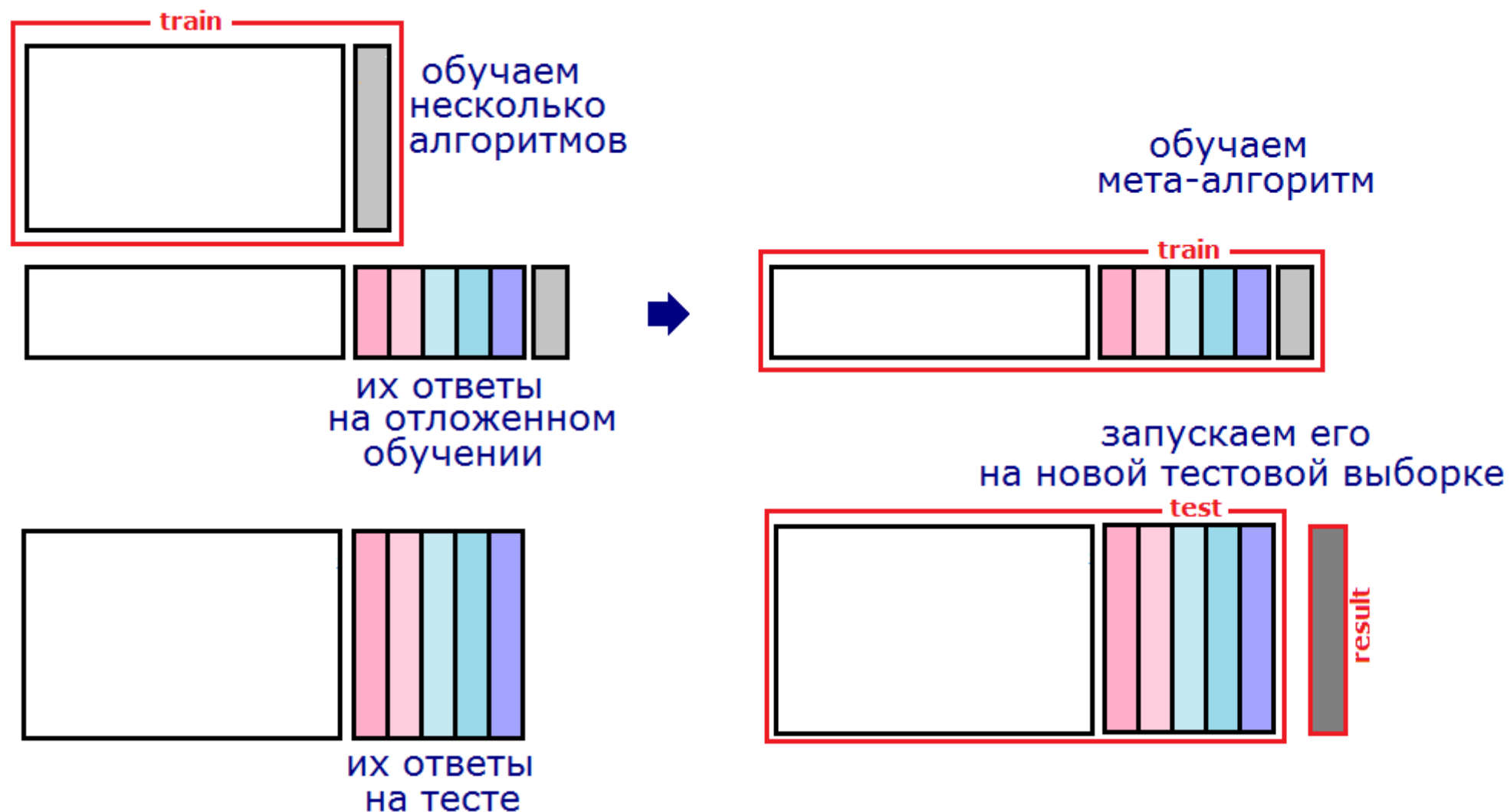
Не сильно повышает качество...

Стекинг



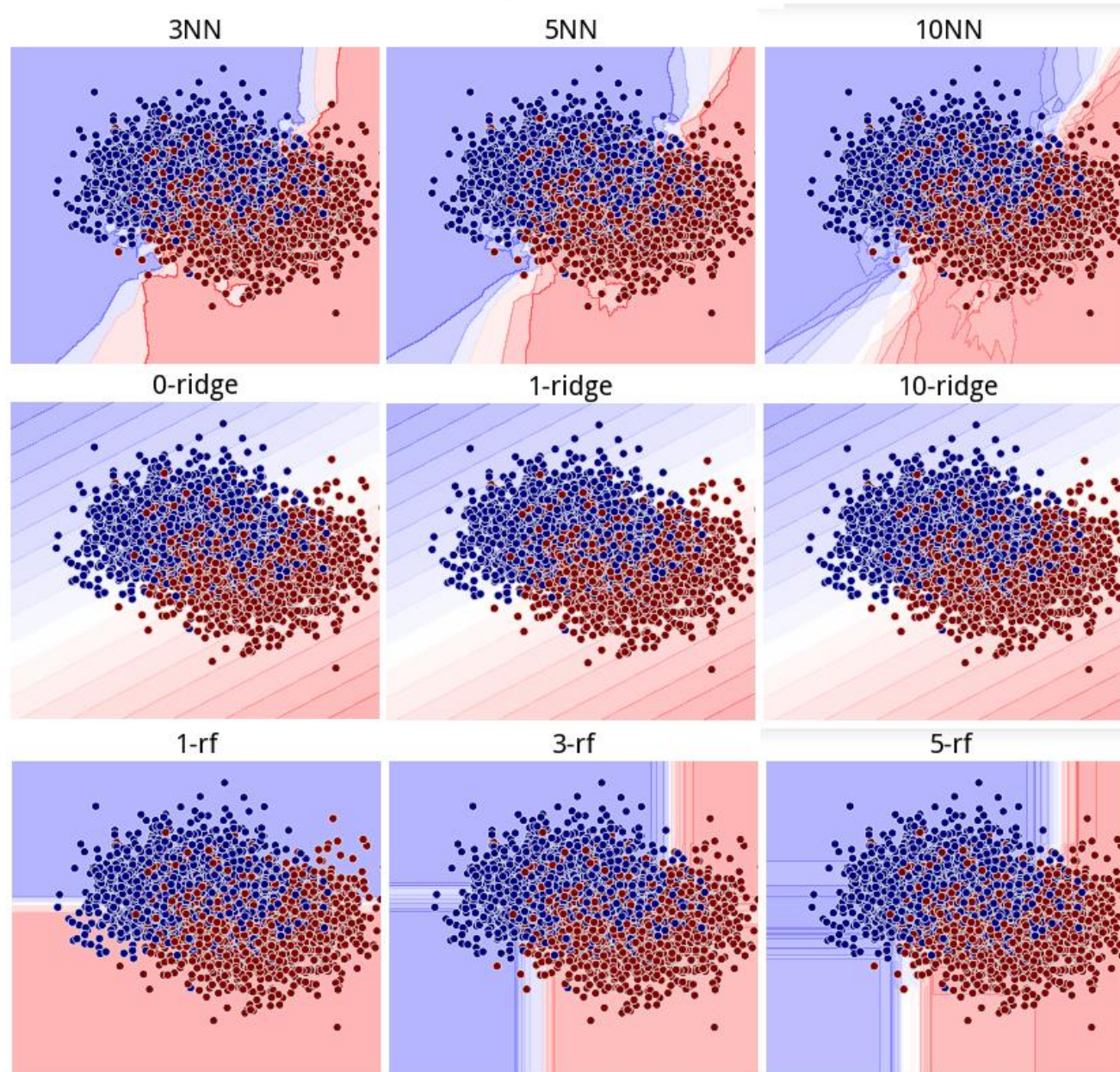
На данных реальной задачи mlbootcamp

Использование признаков с мета-признаками



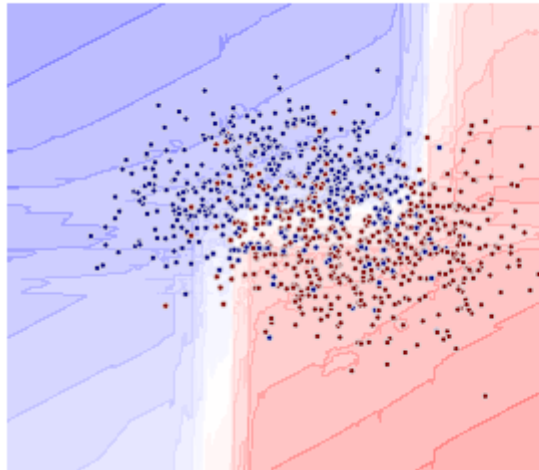
можно добавлять результаты обучения без учителя...

Геометрия стекинга

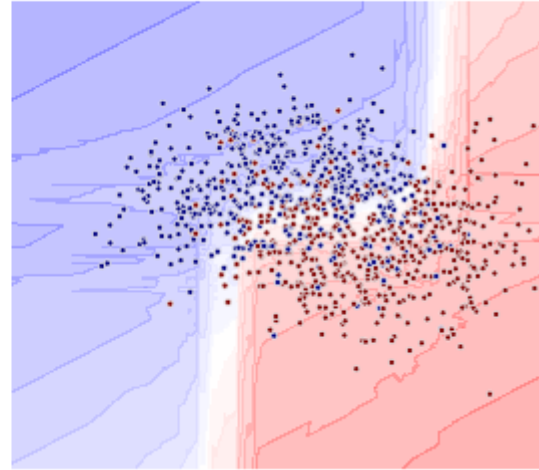


Геометрия стекинга

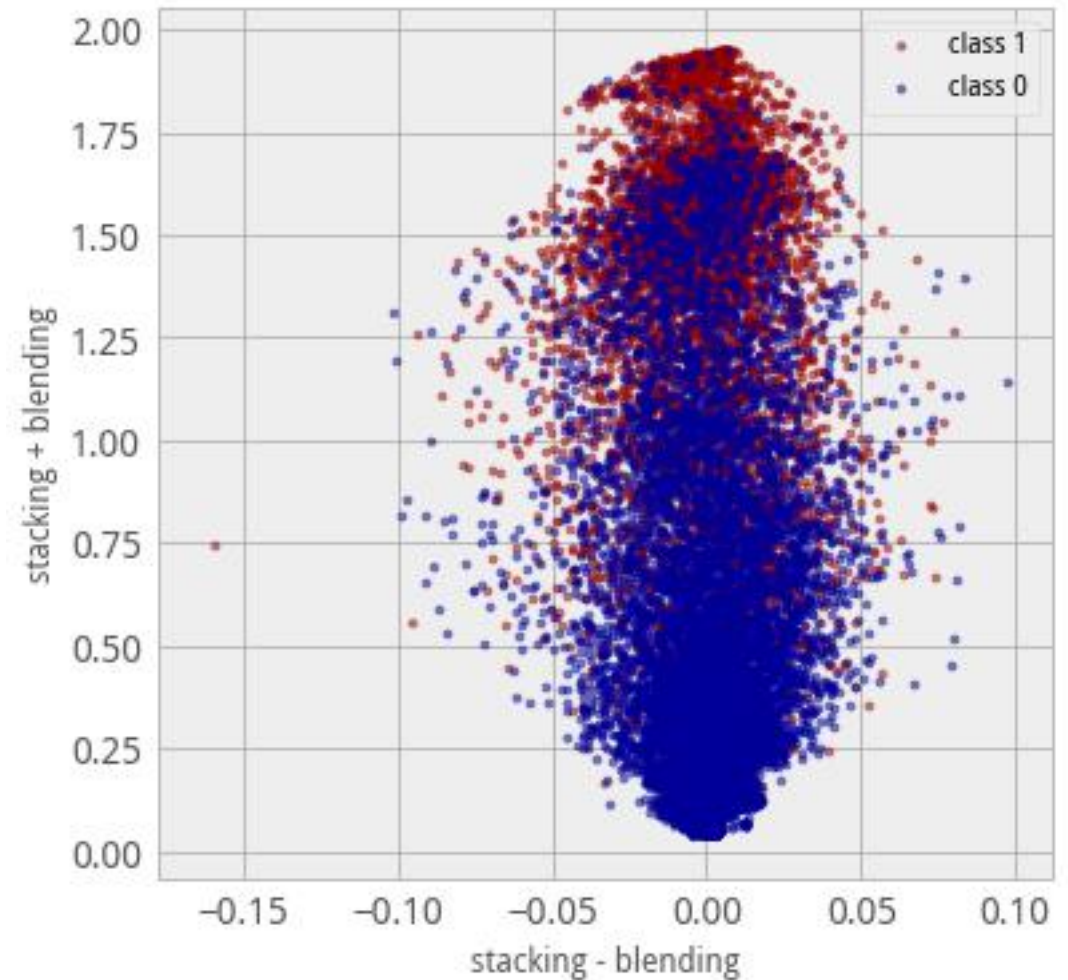
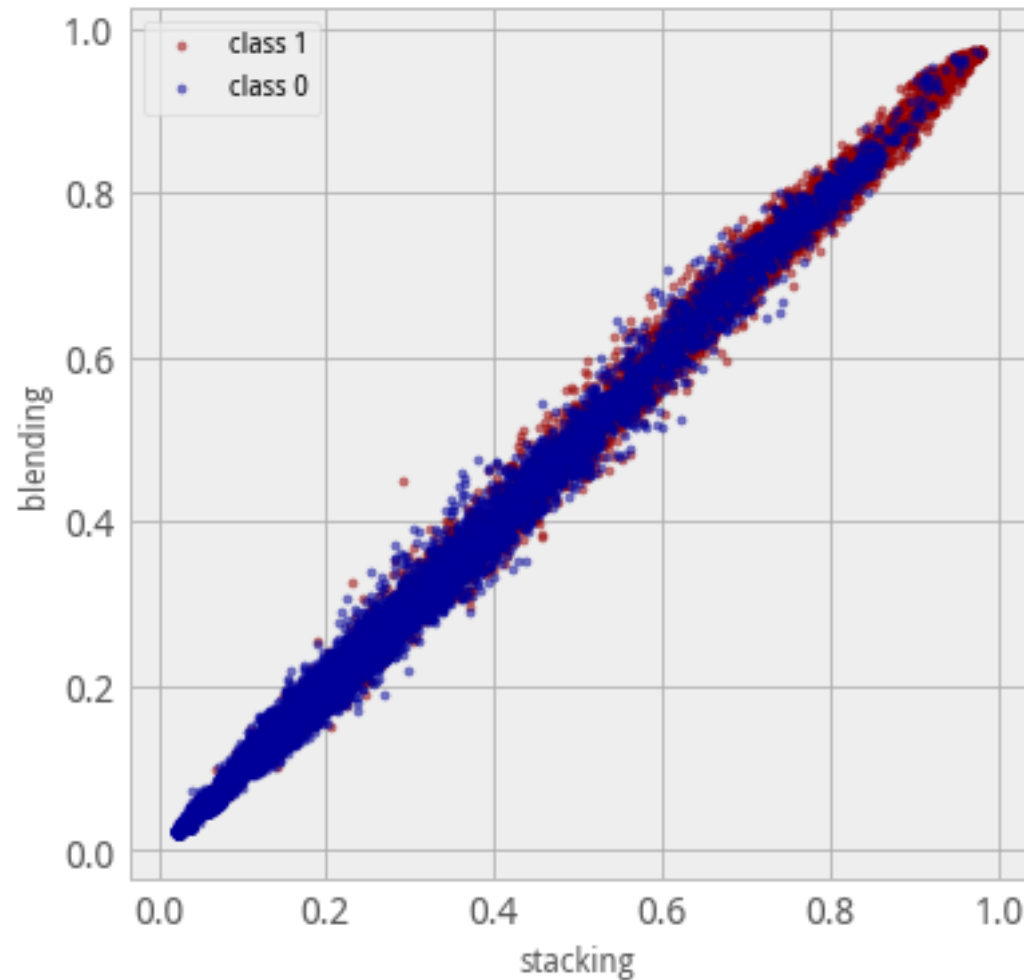
blending



stacking



Стекинг vs Блендинг



Результаты очень похожи...

Стекинг

- **Нужны достаточно большие выборки**

- **Заточен на работу алгоритмов разной природы**

Но для каждого м.б. своё признаковое пространство

- **Хорош на практике (бизнес-задачи)**

Пример: регрессоры + RF =

устойчивость к аномальным значениям признаков

- **Метаалгоритм должен минимизировать целевую функцию**

Не всё так просто... $\log_{\text{regs}} + \log_{\text{reg}}$ может не справиться с Log_loss

- **Многоуровневый стекинг**

Оправдан только в спортивном анализе данных

Стекинг

- **Пространство метапризнаков удобнее признакового, но признаки сильно коррелированы**

Но нет хорошей теории на эту тему

- базовые алгоритмы не сильно оптимизируют,
- настраиваются не на целевой признак (на его квадрат, на разницу между каким-то признаком и целевым),
- используют модели ориентированные на разные функционалы качества,
- пополняют множество базовых алгоритмов алгоритмами, которые решают другую задачу (например кластеризаторами).

ECOC

= Error-Correcting Output Code

**Пусть есть задача с L классами,
а у нас классификаторы на 2 класса**

1. One-vs-All – каждый класс отделяем от остальных

0 – 1000

1 – 0100

2 – 0010

3 – 0001

2. One-vs-One – попарно классы друг от друга

0 – 111--

1 – 0--11-

2 – -0-0-1

3 – --0-00

ЕСОС

3. Допустима произвольная кодировка классов:

0 – 00

1 – 10

2 – 01

3 – 11

4. В том числе, с помощью ЕСОС

0 – 000111

1 – 011100

2 – 101010

3 – 110001

Бустинг: Forward stagewise additive modeling (FSAM)

Центральная идея бустинга

Задача регрессии

$$(x_i, y_i)_{i=1}^m$$

функция ошибки

$$L(y, a)$$

уже есть алгоритм $a(x)$ строим $b(x)$:

$$a(x_i) + b(x_i) = y_i, i \in \{1, 2, \dots, m\}.$$

Надо:

$$\sum_{i=1}^m L(y_i, a(x_i) + b(x_i)) \rightarrow \min,$$

Бустинг: Forward stagewise additive modeling (FSAM)

0. Начать с $a_0(x) \equiv 0$

1. Цикл

$$(b, \eta) = \arg \min_{b, \eta} \sum_{i=1}^m L(y_i, a_{k-1}(x_i) + \eta b(x_i))$$
$$a_k = a_{k-1} + \eta b$$

Пример: L_2 -бустинг

$$\eta = 1, L(y, a) = (y - a)^2$$

$$\sum_{i=1}^m (y_i - a_{k-1}(x_i) - b(x_i))^2 \rightarrow \min$$

тут м.б. обычная регрессия

градиентный бустинг – отдельная лекция...

AdaBoost: постановка задачи

- **FSAM** для бинарной задачи классификации $Y = \{+1, -1\}$
базовые классификаторы генерируют классы
 $b(x) \in \{+1, -1\}$

Ансамбль

$$a(x) = \text{sgn} \left(\sum_{j=1}^s \alpha_j b_j(x) \right)$$

exponential loss

$$L(y, a) = \exp \left(-y \sum_{j=1}^s \alpha_j b_j(x) \right)$$

AdaBoost: весовая схема

У каждого объекта – вес (распределение!)

$$\sum_{t=1}^m w^t = 1$$

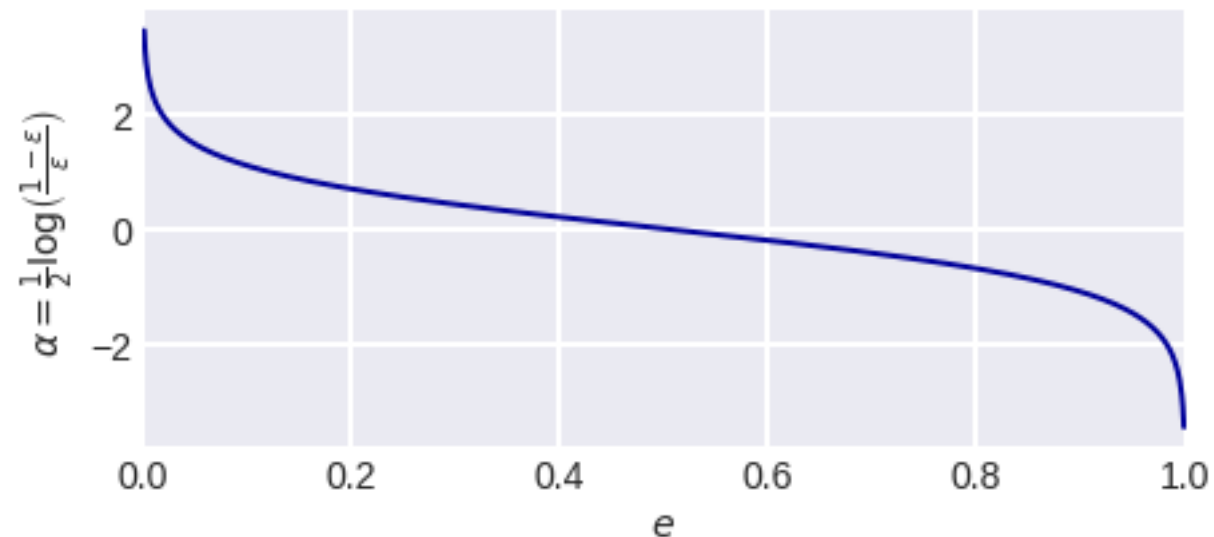
«ошибка, порождённая распределением» (не имеет общего с экспоненциальной ошибкой – просто обозначение) –

$$e(a) = \sum_{\substack{t=1, \\ a(x^t) \neq y(x^t)}}^m w^t$$

AdaBoost: алгоритм

0. Зададим начальное вероятностное распределение (веса)	$W = \left(\frac{1}{m}, \dots, \frac{1}{m} \right)$
1. Цикл по j от 1 до S	
1.1. Построить классификатор b_j, который допускает ошибку $e(b_j)$	(вычисляется по распределению W) предполагаем, что $0 < e(b_j) < 0.5$
1.2. Пусть $\alpha_j = \frac{1}{2} \ln \left(\frac{1 - e(b_j)}{e(b_j)} \right)$ «Перестроить» распределение $W = (w^1, \dots, w^m)$:	$w^t = \frac{w^t \exp(-\alpha_j y(x^t) b_j(x^t))}{\sum_{t=1}^m w^t \exp(-\alpha_j y(x^t) b_j(x^t))}$

AdaBoost: коэффициент



Зависимость коэффициента от ошибки

вариант: перенастраивать веса только объектов, на которых
ошибки...

AdaBoost: вывод формул

экспоненциальная ошибка:

$$L(y, a(x)) = \exp\left(-y \sum_{j=1}^s \alpha_j b_j(x)\right)$$

распишем (выделим последний «только что построенный» базовый)

$$\sum_{t=1}^m w_t \exp\left(\underbrace{-y_t \sum_{j=1}^{s-1} \alpha_j b_j(x_t)}_H - y_t \alpha_s b_s(x_t)\right) = \exp(H) \sum_{t=1}^m w_t \exp(-y_t \alpha_s b_s(x_t))$$

можно множитель не учитывать

$$\begin{aligned} \sum_{t=1}^m w_t \exp(-y_t \alpha_s b_s(x_t)) &= \sum_{t: y_t = a_s(x_t)} w_t \exp(-\alpha_s) + \sum_{t: y_t \neq a_s(x_t)} w_t \exp(\alpha_s) = \\ &= (1 - e) \exp(-\alpha_s) + e \exp(\alpha_s) \end{aligned}$$

AdaBoost: вывод формул

в итоге ошибка $\propto (1 - e) \exp(-\alpha_s) + e \exp(\alpha_s)$

**если хотим найти оптимальный множитель,
продифференцируем и приравняем к нулю**

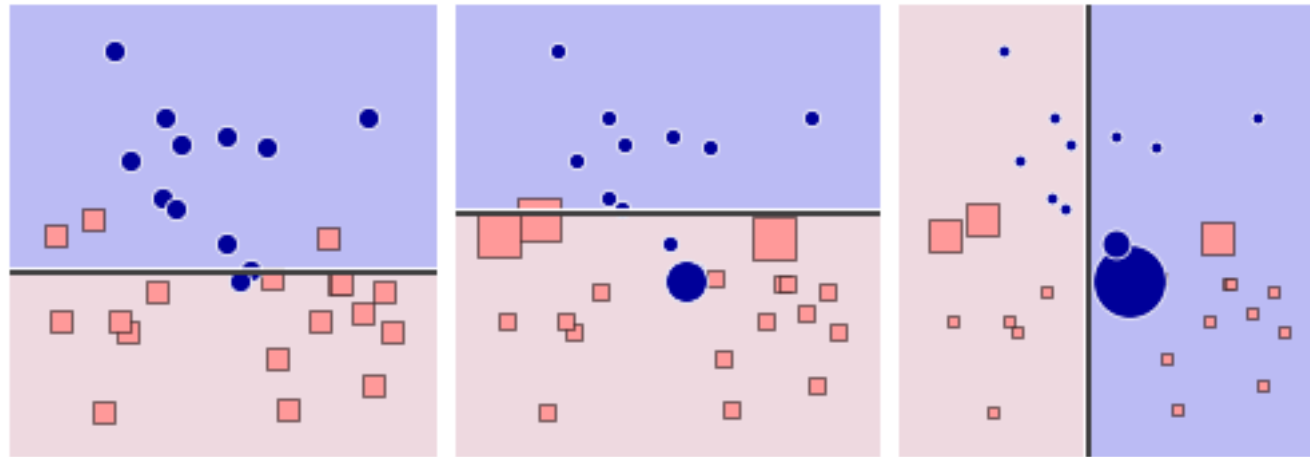
$$\alpha_s = \frac{1}{2} \log \frac{1 - e}{e}$$

теперь смотрим на формулу ошибки

$$\exp(H) \sum_{t=1}^m w_t \exp(-y_t \alpha_s b_s(x_t))$$

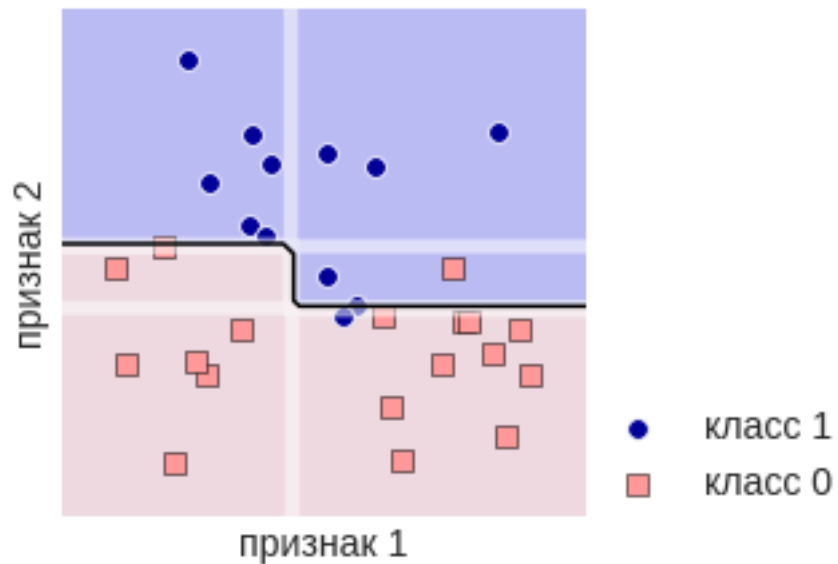
**и пересчёта весов... это просто учёт в весах новых ответов
всегда происходит умножение...**

AdaBoost: пример



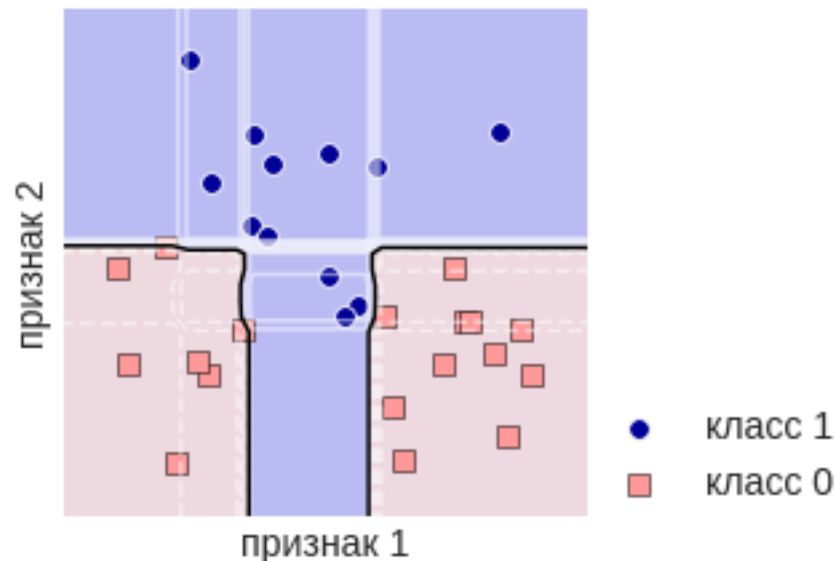
объектов пропорционален весу...

**– полученная комбинация
классификаторов**

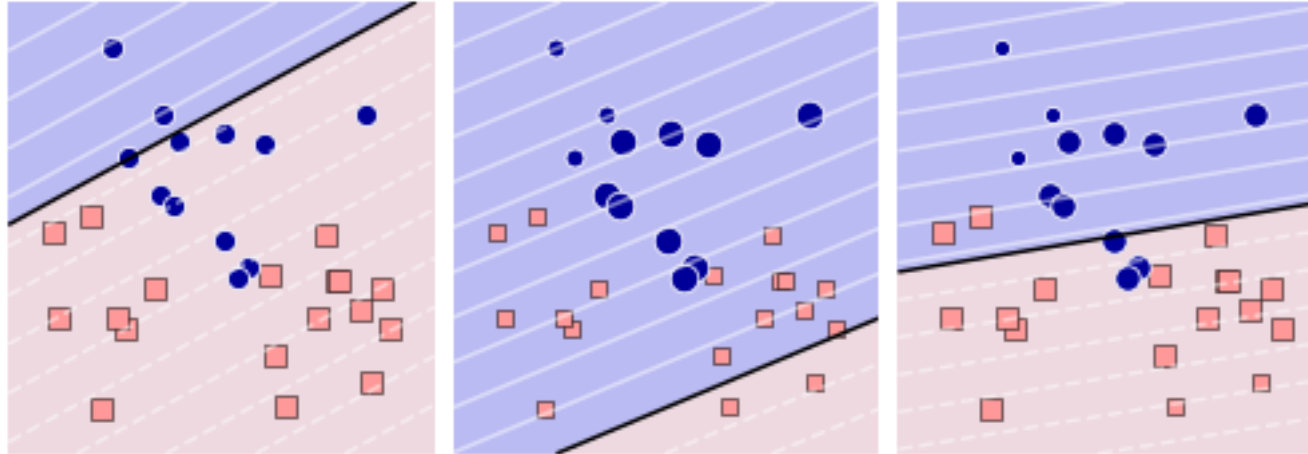


AdaBoost: минутка кода

```
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
model =
AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=1),
                    n_estimators=20, learning_rate=1.0,
                    algorithm='SAMME.R', random_state=1)
model.fit(X, y)
```



AdaBoost: недостатки



Бустинг плох, когда есть выбросы.

**В приведённом примере бустинг плох над логистической регрессией
(над стабильными алгоритмами)!**

Ручные методы ансамблирования

Метод Ефимова

- y_{FE} - Functional Ensembling response
- y_{GBM} - Generalized Boosting response

	$y_{FE} \leq 0.1$	$0.1 < y_{FE} < 0.9$	$y_{FE} \geq 0.9$
$y_{GBM} \leq 0.1$	min	min	0.55
$y_{GBM} \leq 0.1$	0.1	mean	0.9
$y_{GBM} \geq 0.9$	0.75	max	max

Amazon Employee Access Challenge

Итог: ключевые идеи ансамблирования

1. Объединение ответов разных алгоритмов
усреднение / голосование / стекинг ...

2. Повышения разнообразия / независимости базовых алгоритмов
«варьирование» признаков, объектов, моделей, в модели и т.п.
Использование подвыборок / весов

3. Ансамблирование: параллельное и последовательное

Parallel ensembles – все алгоритмы строятся независимо
Идея: усреднить (high complexity, low bias)-модели, для снижения
variance

Sequential ensembles – алгоритмы строятся последовательно

Общая классификация главных мета-алгоритмов

	разброс (model's variance)	смещение (model's bias)	функциональна я выразимость	основа техники
Bagging «среднее»	уменьшает			bootstrap
Boosting «взвешенное среднее»		уменьшает	(увеличивает)	градиентный спуск (сейчас)
Stacking Мета-алгоритм	(уменьшает)	(уменьшает)	увеличивает	суперпозиция алгоритмов