курс «Машинное обучение»

Ансамбли алгоритмов машинного обучения

Александр Дьяконов

9 ноября 2021 года

План

Ансамбли алгоритмов: примеры и обоснование (статистическое, вычислительное, функциональное)

Повышение разнообразия в ансамблях

Комитеты (голосование, Voting Ensembles), усреднение
Бэгинг (Bagging)
Пэстинг (Pasting)
Случайные подпространства (Random Subspaces)
Случайные патчи (Random Patches)
Cross-Validated Committees
Стекинг (Stacking)
Блендинг (Blending)

…

Ансамбль алгоритмов (Ensemble / Multiple Classifier System)

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

простой ансамбль в регрессии:

$$a(x) = \frac{1}{n}\left(b_1(x) + \ldots + b_n(x)\right)$$

простой ансамбль в классификации:

$$a(x) = \text{mode}\left(b_1(x), \ldots, b_n(x)\right)$$

комитет большинства

В чём может быть усложнение?

Ансамбль алгоритмов

$$a(x) = b\left(b_1(x),\ldots,b_n(x)\right)$$

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

b_i – **базовые алгоритмы (base learners)**

в бустинге – слабые (weak)

Реализация в scikit-learn

`sklearn.ensemble.VotingClassifier`

`estimators`	**Список базовых алгоритмов**
`voting="hard"`	**Голосование по меткам или усреднение вероятностей**
`weights=None`	**Веса**
`n_jobs=None`	**«number of jobs»**
`flatten_transform=True`	**Формат ответа (для soft-ансамбля)**

есть ещё

`ensemble.VotingRegressor`

Ошибка суммы регрессоров: теоретическое обоснование

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

$$\xi = \frac{1}{n}\left(\xi_1 + \ldots + \xi_n\right)$$

$$\mathrm{E}\xi = \frac{1}{n}\left(\mathrm{E}\xi_1 + \ldots + \mathrm{E}\xi_n\right) = \mathrm{E}\xi_i$$

$$\mathbf{D}\xi = \frac{1}{n^2}\left(\mathbf{D}\xi_1 + \ldots + \mathbf{D}\xi_n\right) = \frac{\mathbf{D}\xi_i}{n}$$

ДЗ А если есть корреляция между базовыми алгоритмами?

решите в постановке, что корреляция между любыми двумя алгоритмами равна ρ

Ошибка комитета большинства: теоретическое обоснование

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

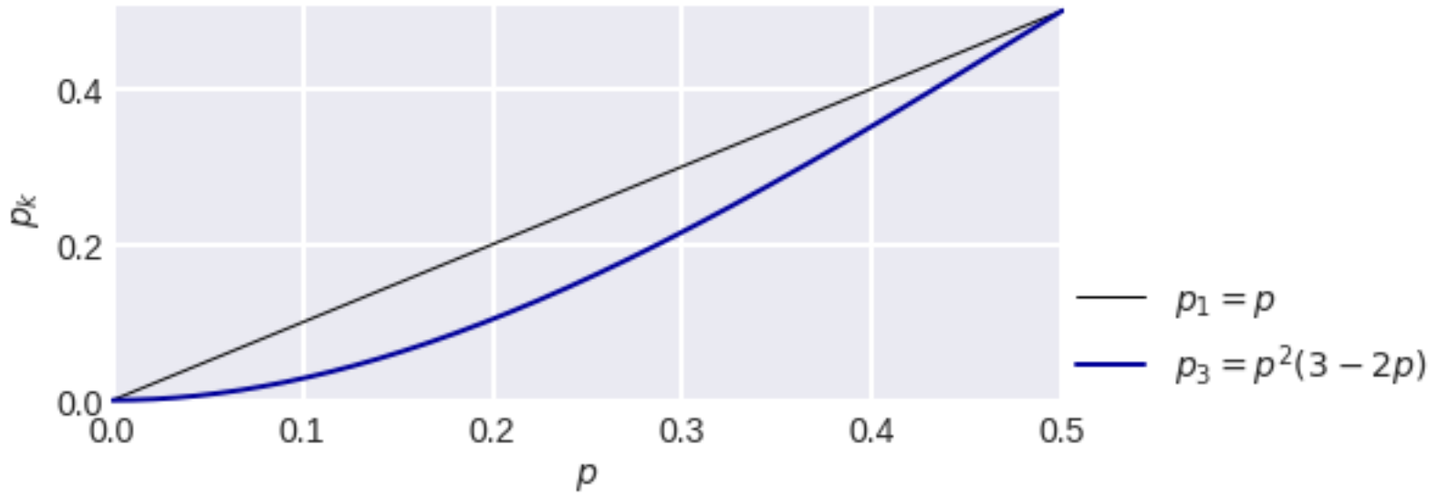
$$\left.\begin{array}{ll}(1,1,1) & ppp\\ (1,1,0) & pp(1-p)\\ (0,1,1) & (1-p)pp\\ (1,0,1) & p(1-p)p\end{array}\right\}$$

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:

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

неравенство Хёфдинга (Hoeffding)

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	похожие	разные
1111110000 q=0.6	1111110000 q=0.6	1010010111
1111110000 q=0.6	1111101000 q=0.6	1100110011
1111110000 q=0.6	1111100100 q=0.6	1111110000
		1110110011
q_ens = 0.6	q_ens = 0.5	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

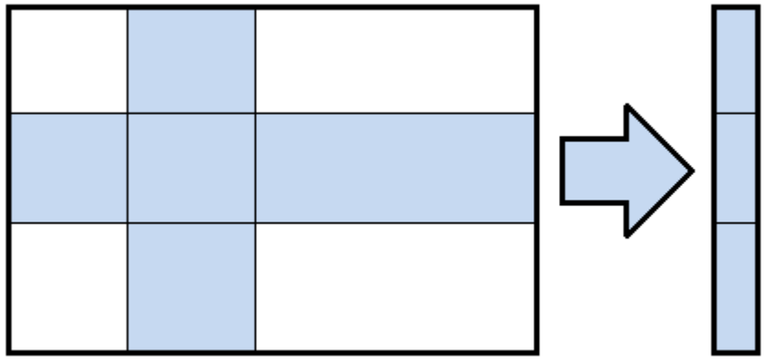
Пример: подвыборки и подмножества признаков в RF

Ещё почему алгоритмы в ансамбле должны быть разными

одинаковые	похожие	разные
1111110000 q=0.6	1110111000 q=0.6	1010010111
1111110000 q=0.6	1101110100 q=0.6	1100110011
1111110000 q=0.6	1011101100 q=0.6	1111110000
		1110110011
q_ens = 0.6	**q_ens = 0.8**	q_ens = 0.7

ДЗ Честный эксперимент, оценивающий зависимость качество (разнообразие)

Повышения разнообразия – что «варьируют»



- **обучающую выборку**
  (бэгинг)
- **признаки**
  (Random Subspaces)
- **целевой вектор**
  (ECOC, f(y))
- **модели**
  (стекинг)
- **алгоритмы в модели**

(разные гиперпараметры, инициализации, snapshot, разные random seed в RF, …)

Варьирование алгоритмов в модели

л/к полиномов разной степени

л/к случайных лесов с разной глубиной

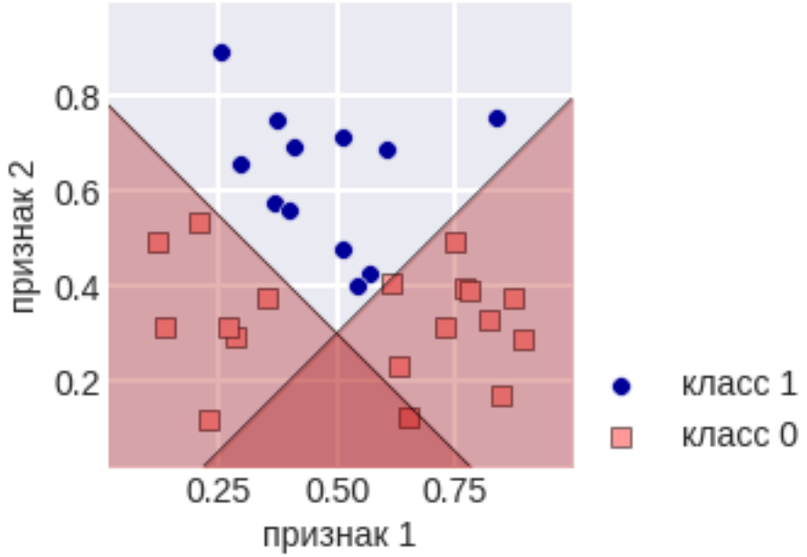
л/к НС с разной инициализацией / после разных эпох

Обоснования применения ансамблей

Статистическое (Statistical) – ошибка может быть меньше

Вычислительное (Computational) – обучение = оптимизация функции, а ансамбль «распараллеливает» процесс

Функциональное (Representational) – можно представить функции, которые нельзя было с помощью базовых алгоритмов



Ансамбли

- **комитеты (голосование) / усреднение**

в том числе, различные усреднения, с предварительной деформацией, калибровкой, **бэгинг (bagging) + обобщения (RF)**

- **перекодировки ответа**

кодирование целевого вектора, **ECOC (error-correcting output coding)**

- **стекинг (stacking)**

построение **метапризнаков** — ответов алгоритмов на объектах выборки, обучение на них мета-алгоритма

- **бустинг (boosting)**

построение суммы алгоритмов: каждое следующее слагаемое строится с учётом ошибок предыдущих

- **«ручные методы»**

эвристические способы комбинирования ответов базовых алгоритмов

- **однородные ансамбли**

рекурсия в формуле мета-алгоритм(базовые) + общая схема оптимизации (пример: нейросети)

Комитеты (голосование, Voting Ensembles)

голосование по большинству (Majority vote)

$$a(x) = \text{mode}\left(b_1(x),\ldots,b_n(x)\right)$$

комитеты единогласия

в бинарной задаче классификации – $a(x) = \min\left(b_1(x),\ldots,b_n(x)\right)$

обнаружение аномалий:

мата-алгоритм – максимум

«тревога при малейшем подозрении»

$$a(x) = \max\left(b_1(x),\ldots,b_n(x)\right)$$

Усреднение

«среднее арифметическое»

$$a(x) = \frac{1}{n}\left(b_1(x) + \ldots + b_n(x)\right)$$

+ любые другие средние (ex: по Колмогорову)

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

Ранговое усреднение (Rank Averaging)

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

ориентировано на конкретный AUC ROC

Усреднение с весами (weighted averaging)

Усреднение (регрессия)

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

Голосование (классификация)

$$a(x) = \arg\max_j \left[\sum_{t:b_t(x)=j} w_t \right]$$

Feature-Weighted Linear Stacking

Области компетентности алгоритмов – линейные регресии

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

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

Бэгинг (Bagging)

– bootstrap aggregating

каждый базовый алгоритм настраивается на случайной подвыборке обучения

Бэгинг (Bagging)	**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**

Бэгинг (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

$$a(x) = \frac{1}{n}\left(b_1(x) + \ldots + b_n(x)\right)$$

(для задач регрессии).

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

$$1 - \frac{1}{e} \approx 0.632$$

~ 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 бэгинга (OOB-prediction)

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

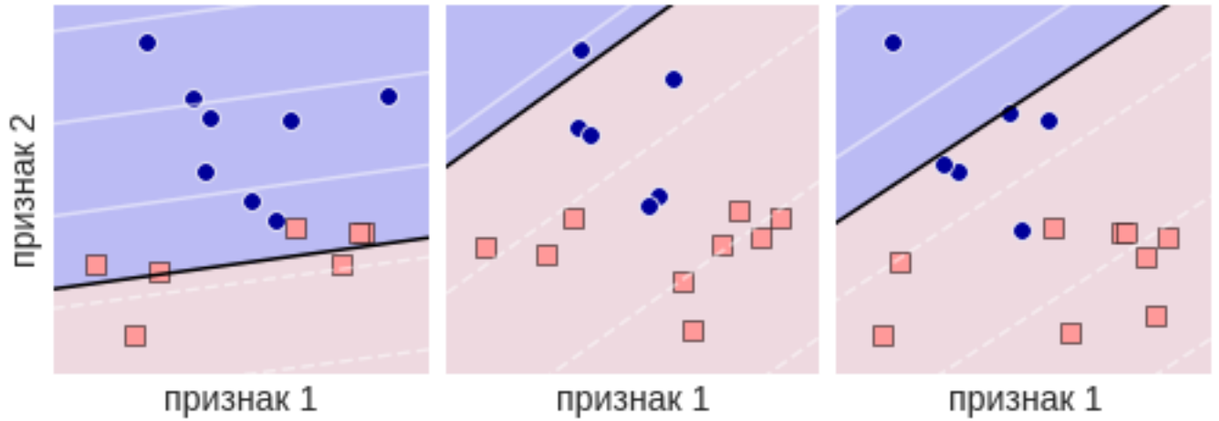
Можно вычислить
OOB-ошибку бэгинга

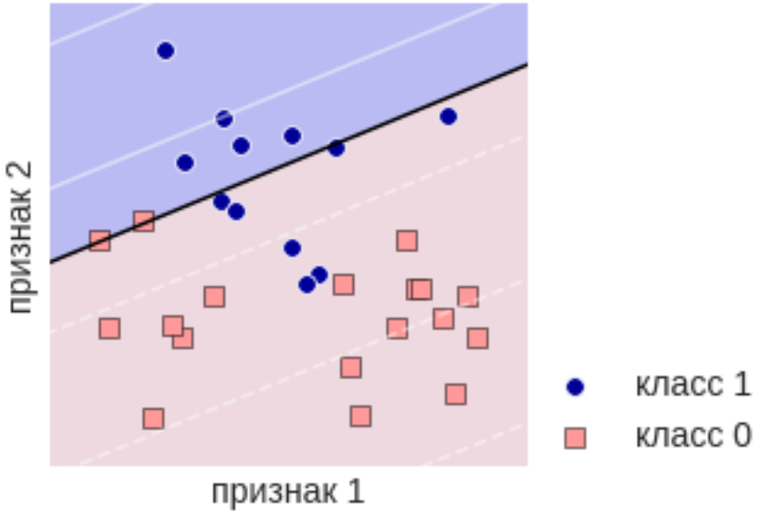
хорошая оценка ошибки на тесте
похожа на CV-ошибку…

Д/З Сравнить OOB-ошибку и ошибку на OOB-ответе (экспериментально)

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)
```

Устойчивость модели (stable learners)

Разброс (variance) показывает изменение алгоритма из модели при незначительном изменении обучающей выборки

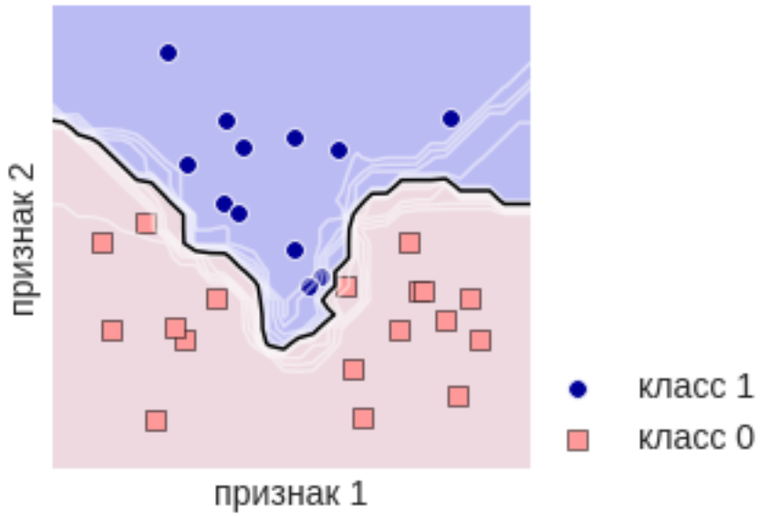
high variance	low variance
CART 1NN	SVM kNN, k>>1

В бэгинге используются неустойчивые модели, но несмещённые (small bias)!

Но тут нет хороших теоретических результатов…

Устойчивость модели (stable learners)

Пример – если выбрать правильную базовую модель для бэгинга



здесь – kNN(1)

Реализация в scikit-learn

`sklearn.ensemble.BaggingClassifier`

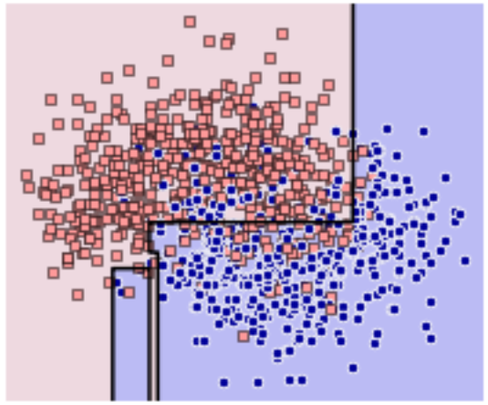
`base_estimator`	**Базовая модель**
`n_estimators=10`	**Число алгоритмов в ансамбле**
`max_samples=1.0`	**Размер подобучения (доля или число)**
`max_features=1.0`	**Число / доля признаков для обучения базового алгоритма**
`bootstrap=True`	**Выбирать ли подобучение с возвращением**
`bootstrap_features=False`	**Аналогичная опция для признаков**
`oob_score=False`	**Вычислять ли OOB-ошибку**
`warm_start=False`	**Использовать ли в качестве начальных приближений старые веса**

`n_jobs=None, random_state=None, verbose=0`

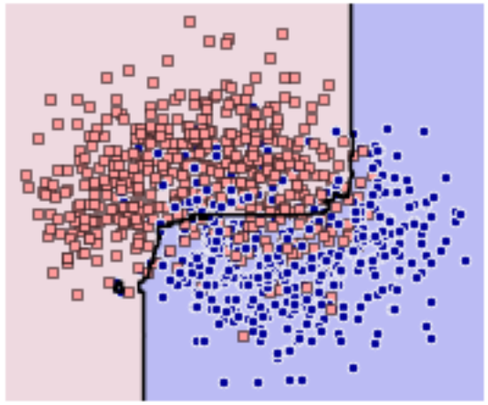
есть ещё

`ensemble.BaggingRegressor`

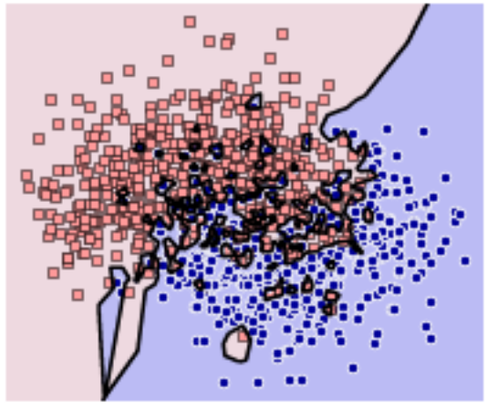
Примеры бэгинга



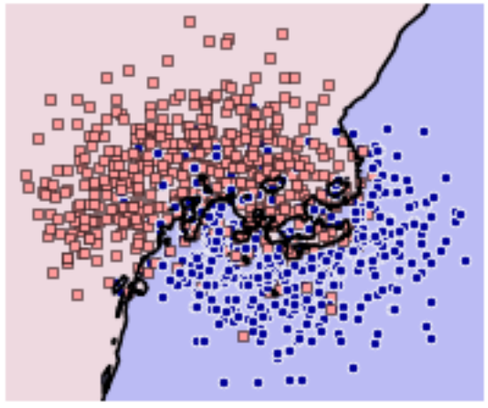
одно дерево



бэгинг 100 деревьев



ближайший сосед



бэгинг 100 ближайших соседей

Идеи из бэгинга, RS и т.п. на практике

Часто признаки делятся / можно разделить на группы:

- **по источнику данных (БКИ1, БКИ2, …)**
- **по типу признака (вещественный, категориальный, …)**
- **по кодированию (OHE, 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…

как?

Случайный лес (Random Forest)

дальнейшие улучшения независимости базовых классификаторов

бэгинг + случайности при построении деревьев

отдельная лекция

Стекинг (stacking)

Идея: хорошо усреднять алгоритмы, но почему именно усреднять?

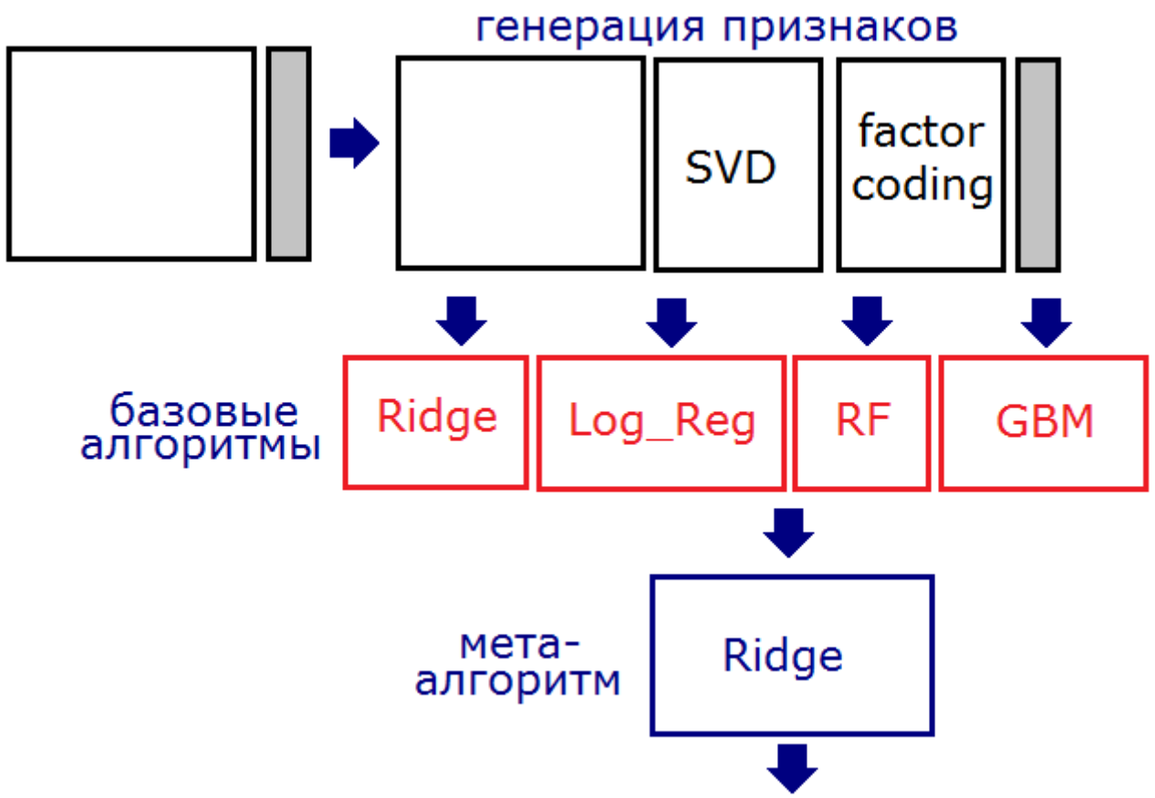
приходит в голову всем…

$$a(x) = b\left(b_1(x), \ldots, b_n(x)\right)$$

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

Д. Волпертом, автором серии теорем «No free lunch…» в 1992 году

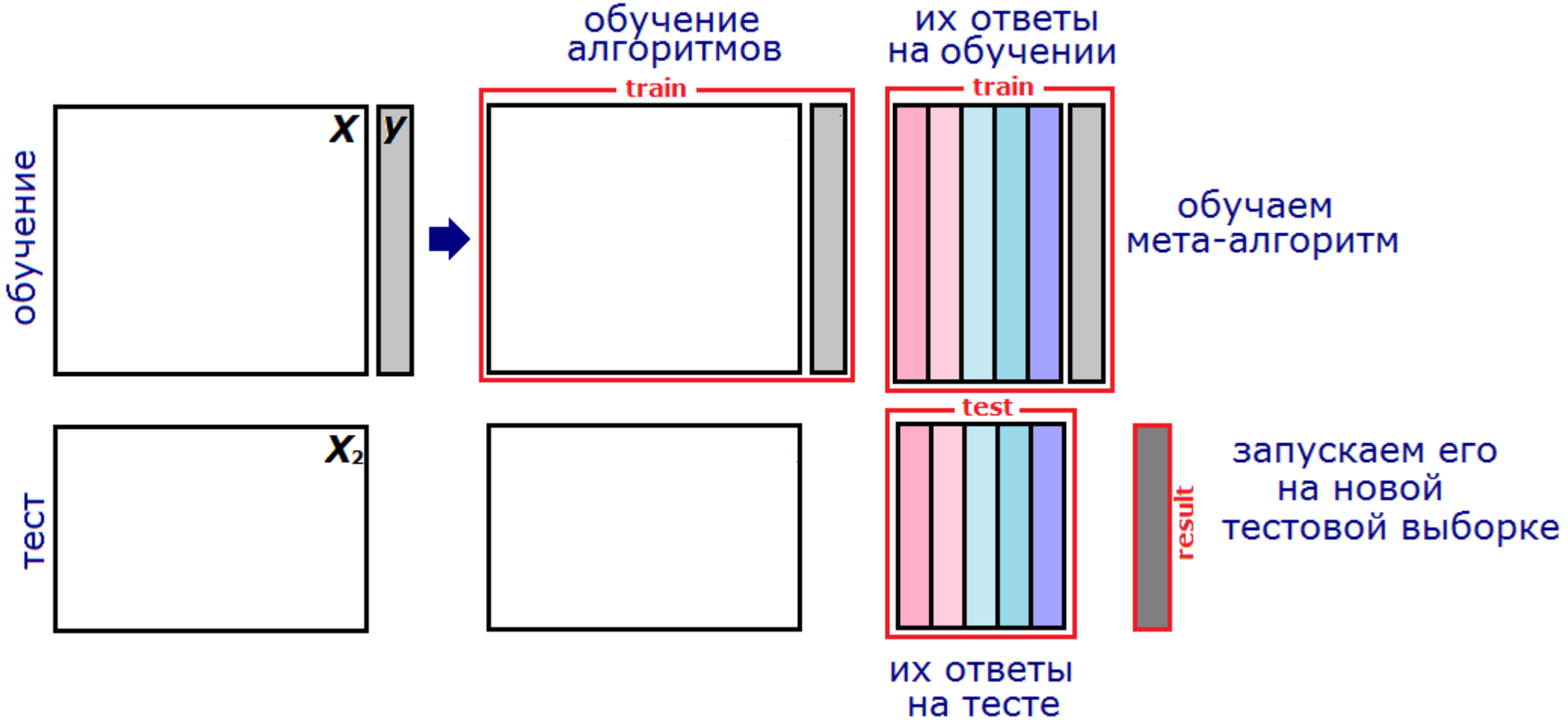
Стекинг (stacking)



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

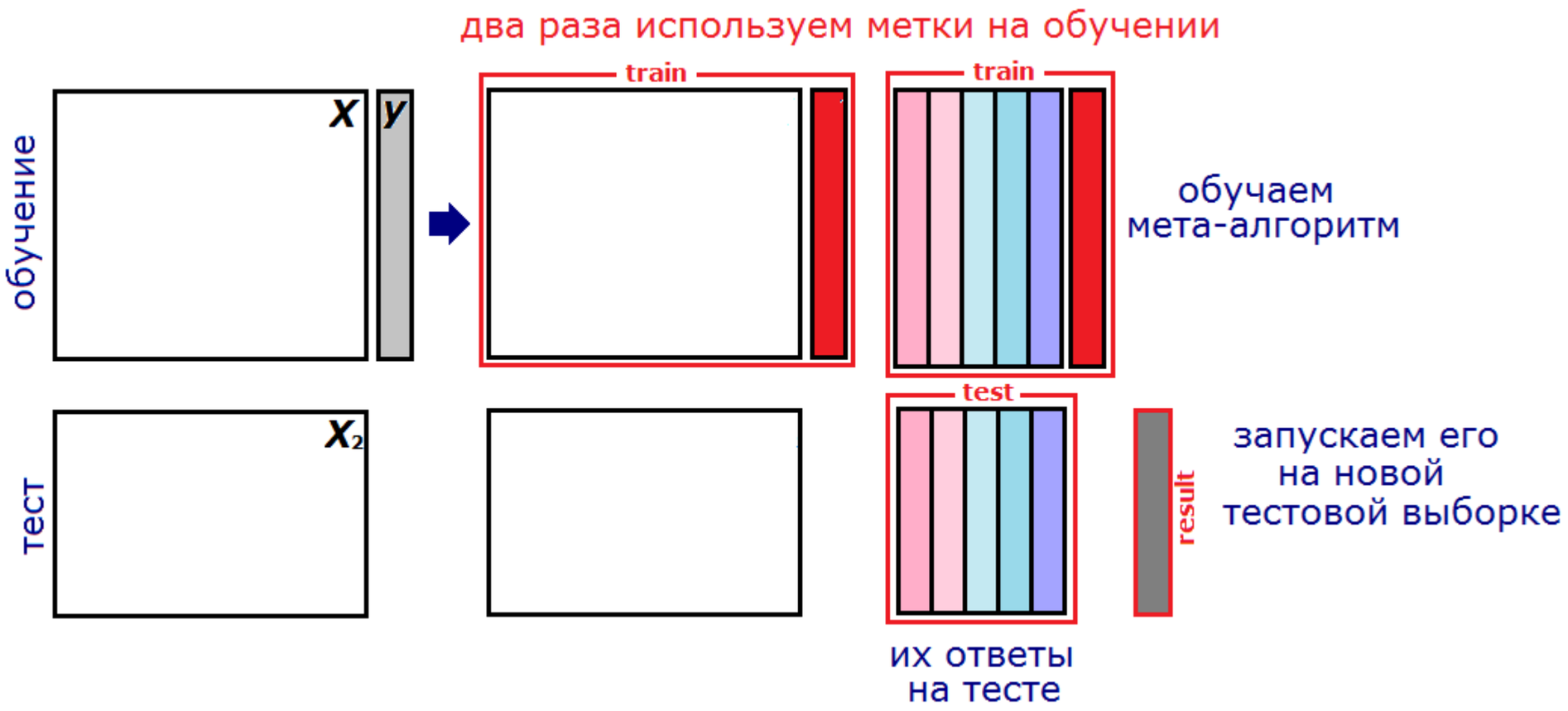
уже есть реализация в scikit-learn!

Наивная форма стекинга

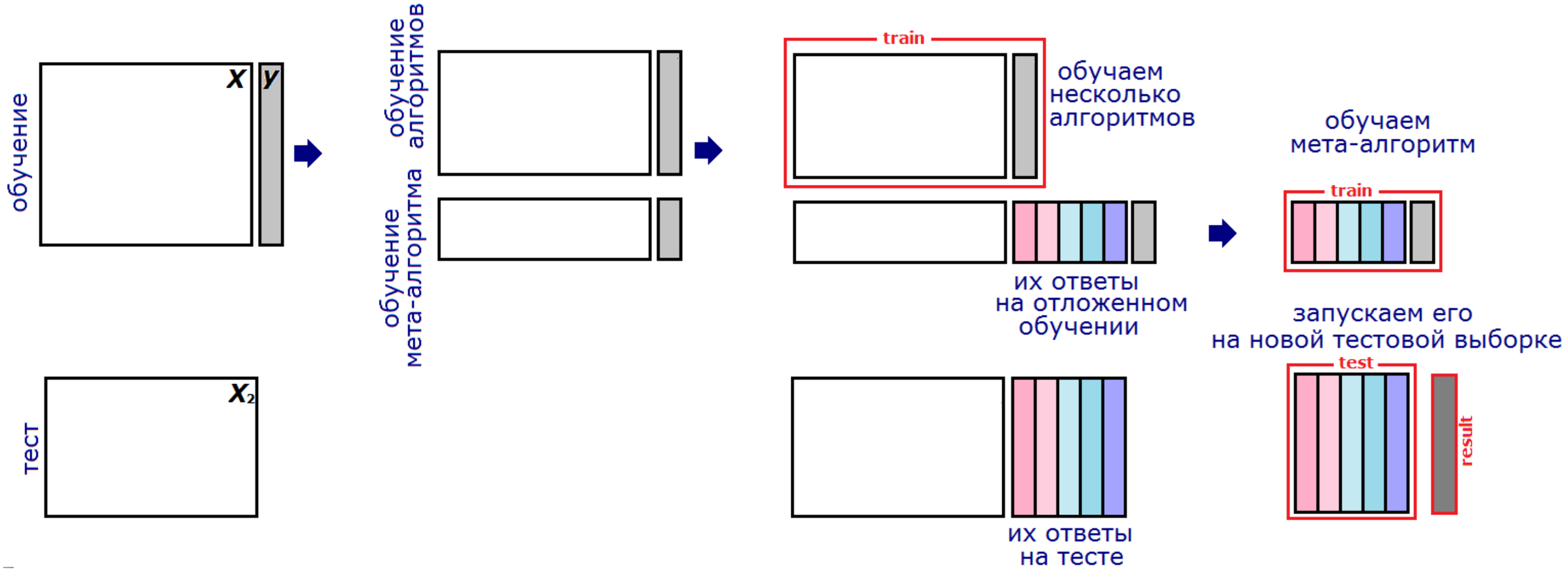


что здесь неправильно?

Наивная форма стекинга



Блендинг (Blending) – простейшая форма стекинга



Блендинг

– термин введён победителями конкурса Netflix

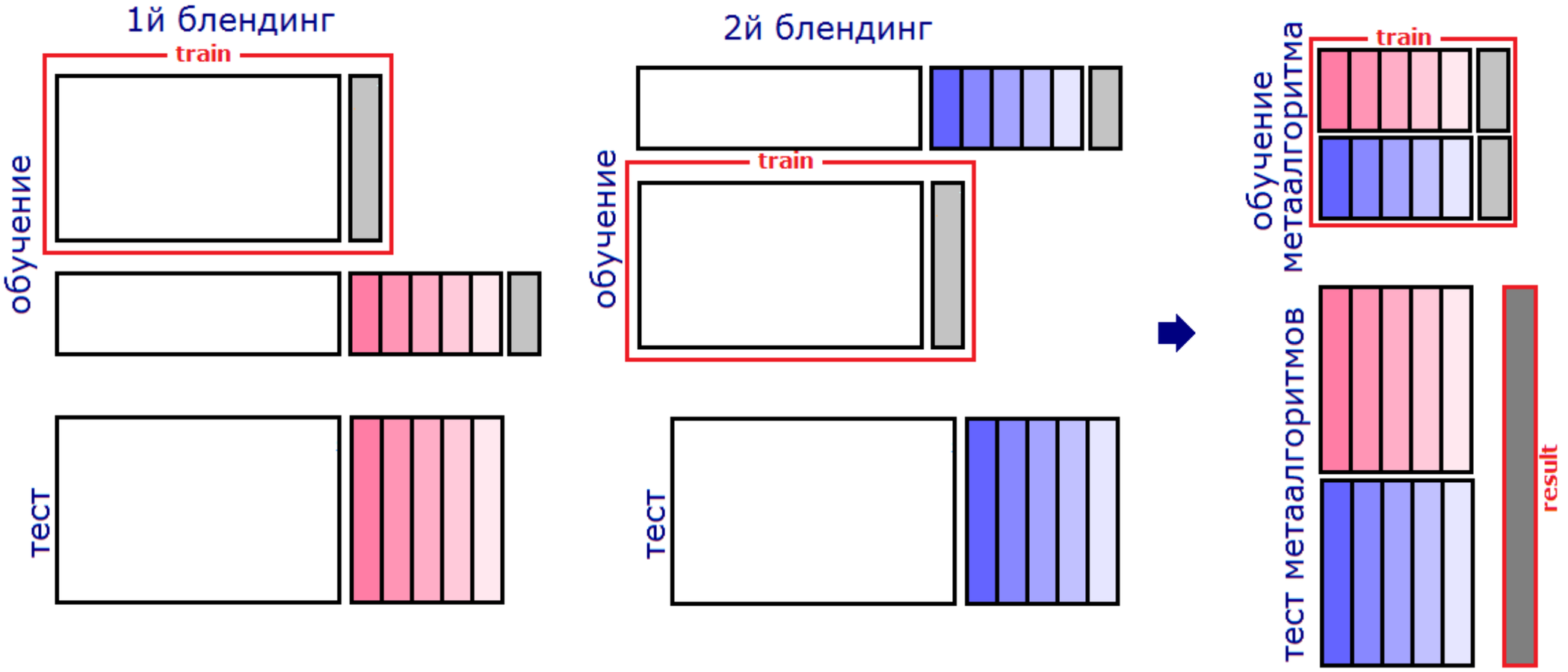
Сейчас блендингом называются простейшие формы стекинга, например, выпуклую комбинацию алгоримтов

Недостатки

Используется не вся обучающая выборка

- **можно усреднить несколько блендингов**
- **можно «состыковать»**
  - **долго и не всегда лучше по качеству**
  - **ответы всё равно надо будет усреднить**

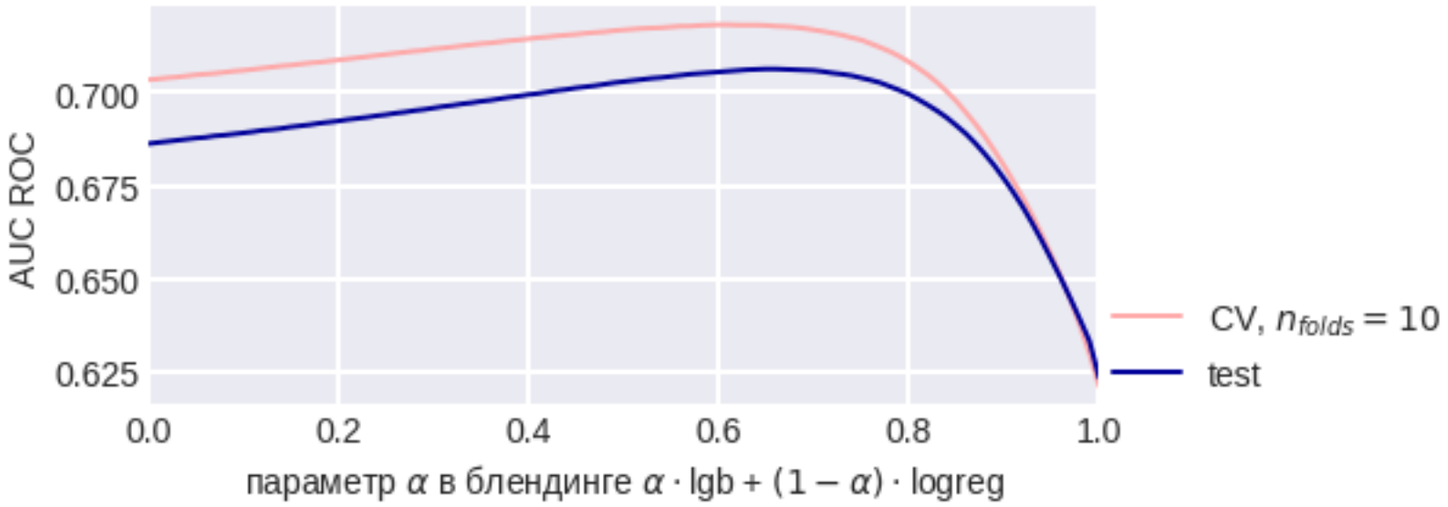
Блендинг: состыковка таблиц



ещё можно усреднить значения мета-признаков на тесте

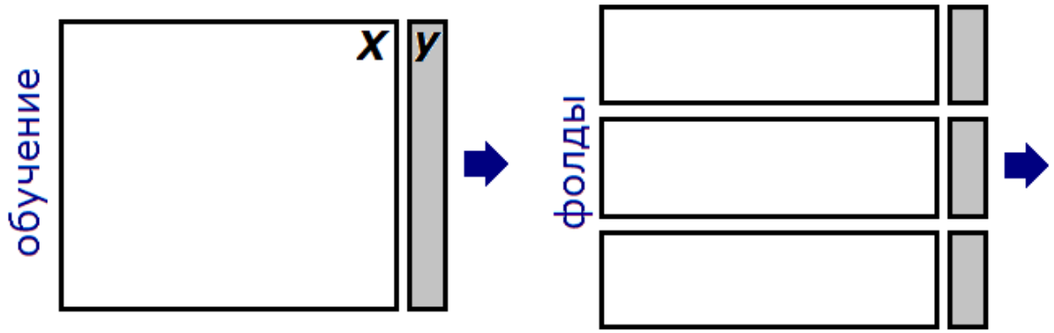
но это меняет распределения

Настройка параметров блендинга

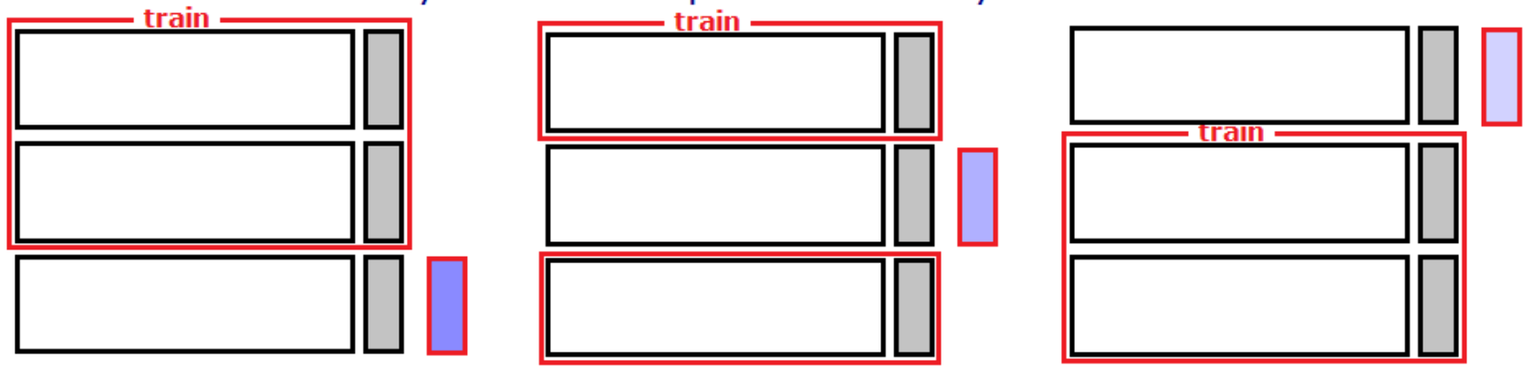


задача Boosters

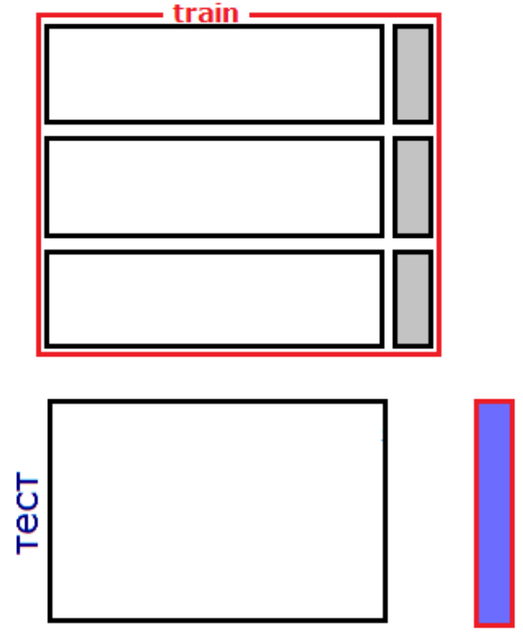
Стекинг – хотим использовать всю обучающую выборку



получение метапризнака на обучении

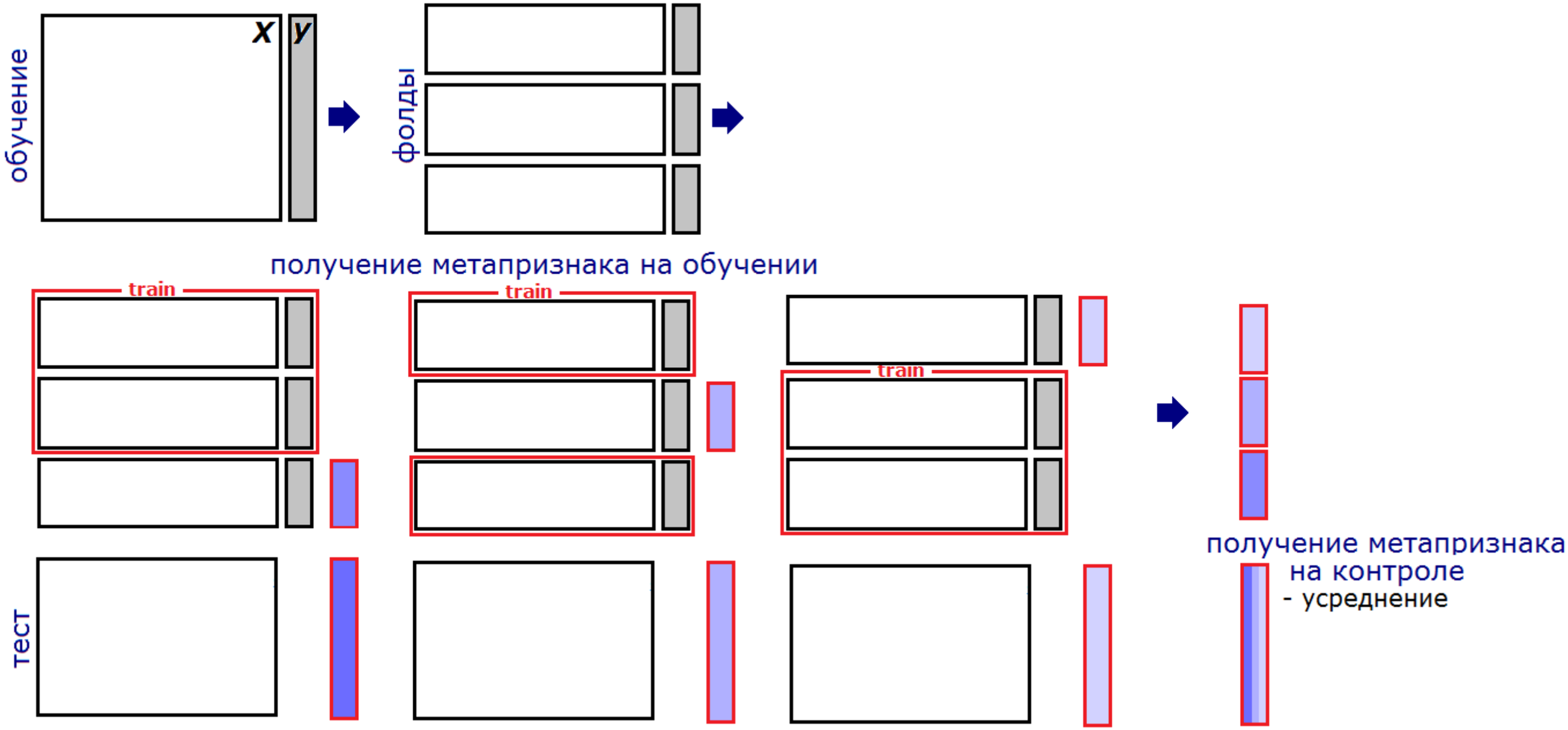


получение метапризнака на контроле



м.б. разные разбиения на фолды и усреднить ответы базовых алгоритмов или стекингов

Стекинг – другой способ получения метапризнаков на контроле



Стекинг

также можно брать разные разбиения и усреднять

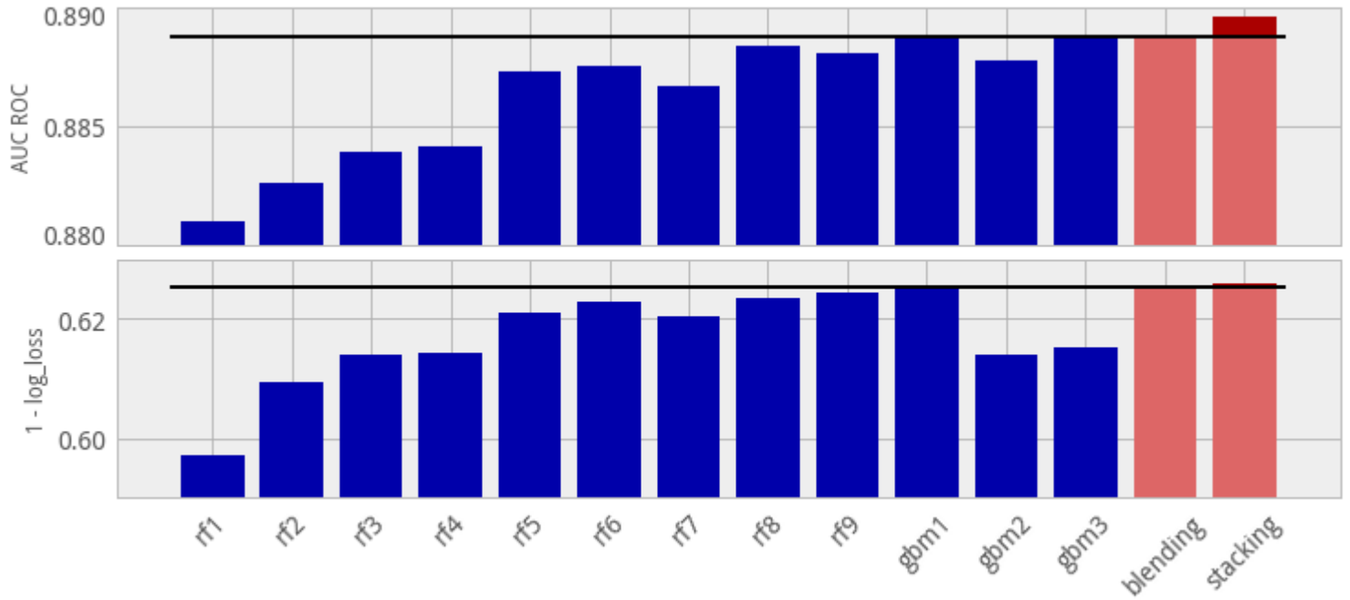
Недостаток

Метапризнаки на обучении и тесте разные!

- регуляризация
- нормальный шум к метапризнакам

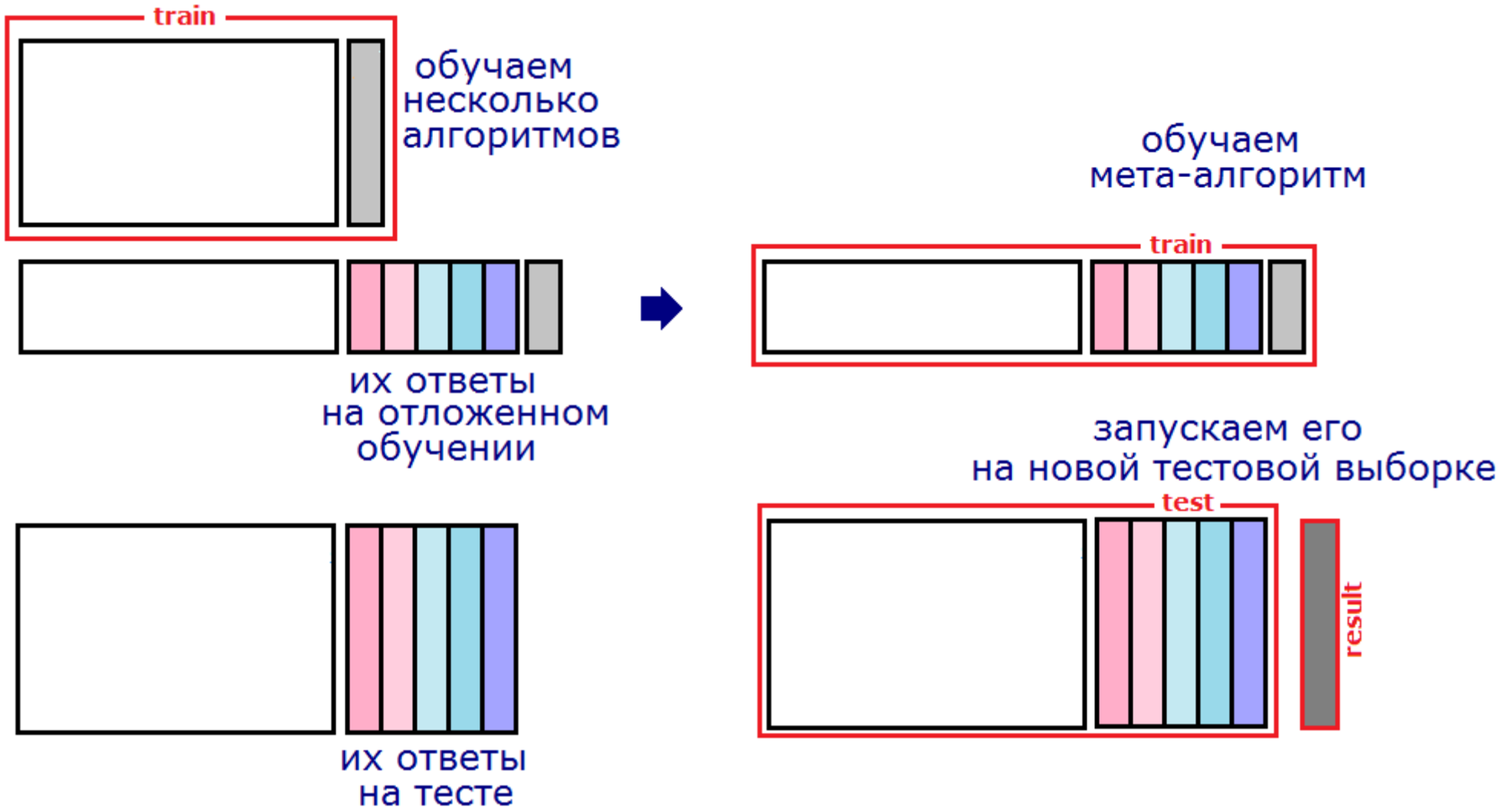
ДЗ Реализовать и сравнить разные виды стекинга

Стекинг



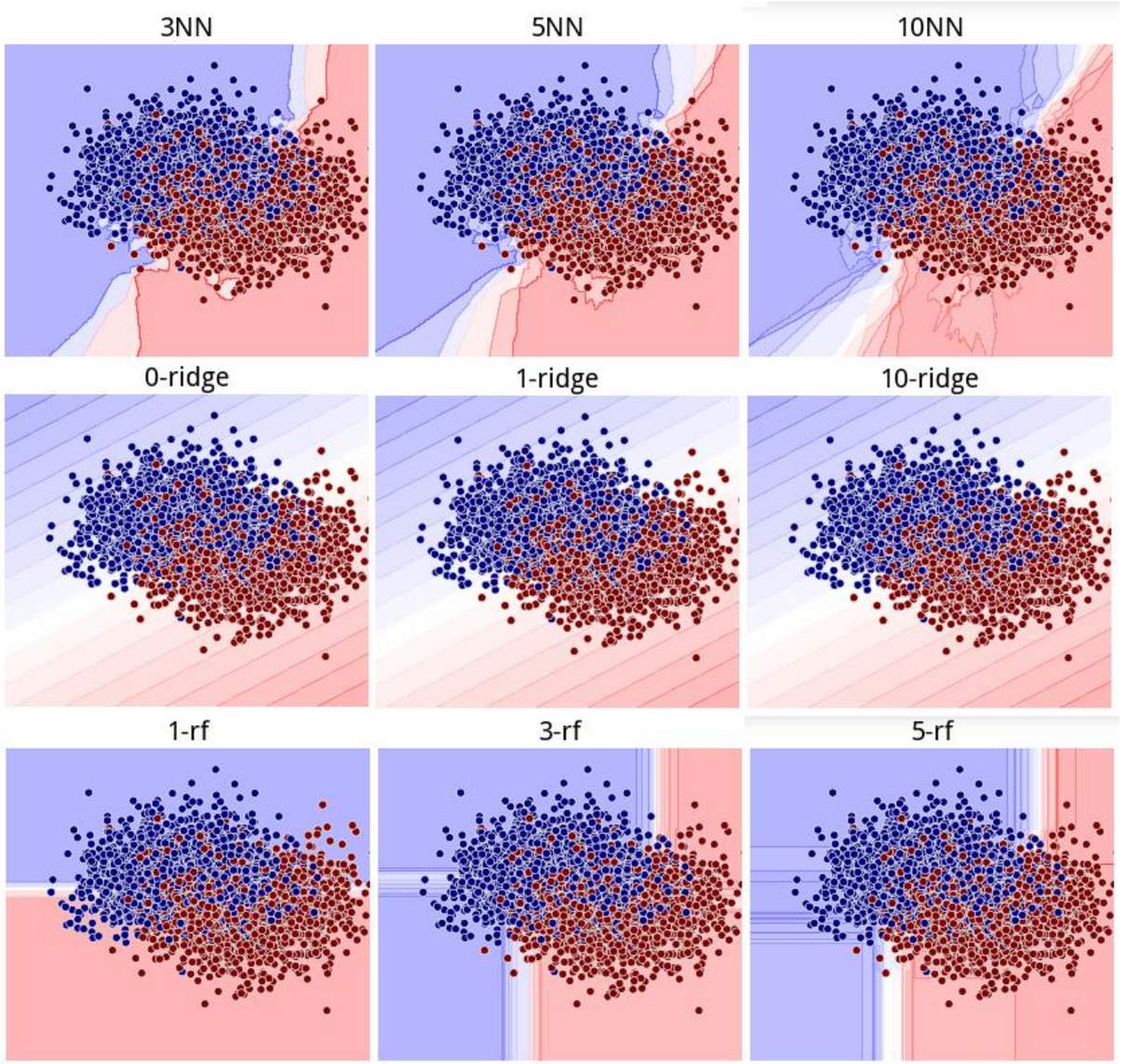
На данных реальной задачи mlbootcamp

Использование признаков с мета-признаками

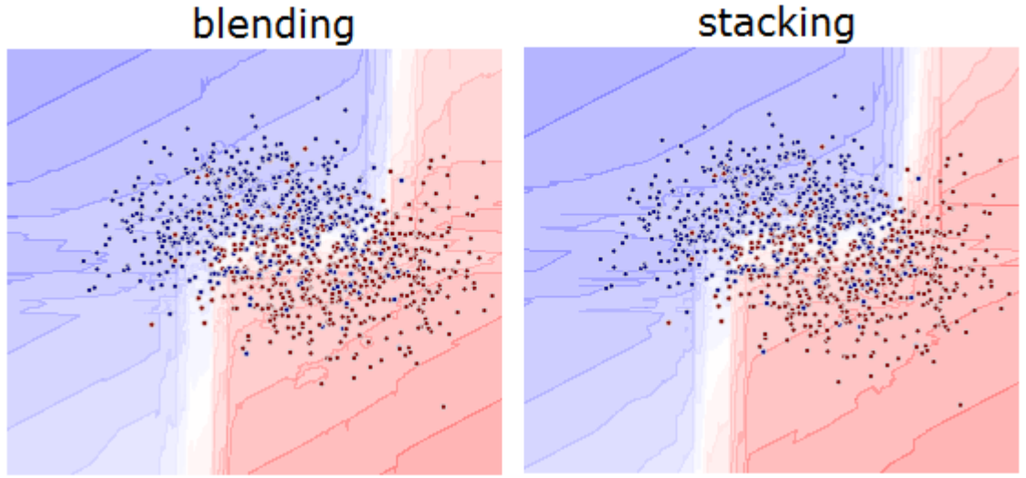


можно добавлять результаты обучения без учителя…

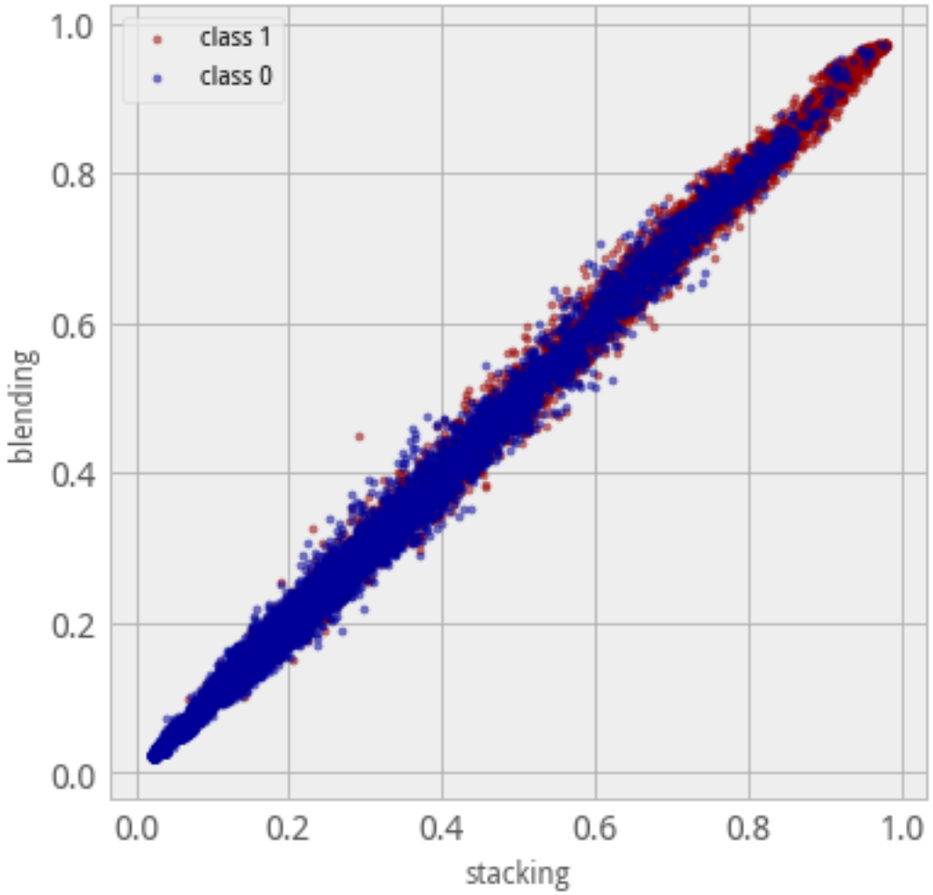
Геометрия стекинга

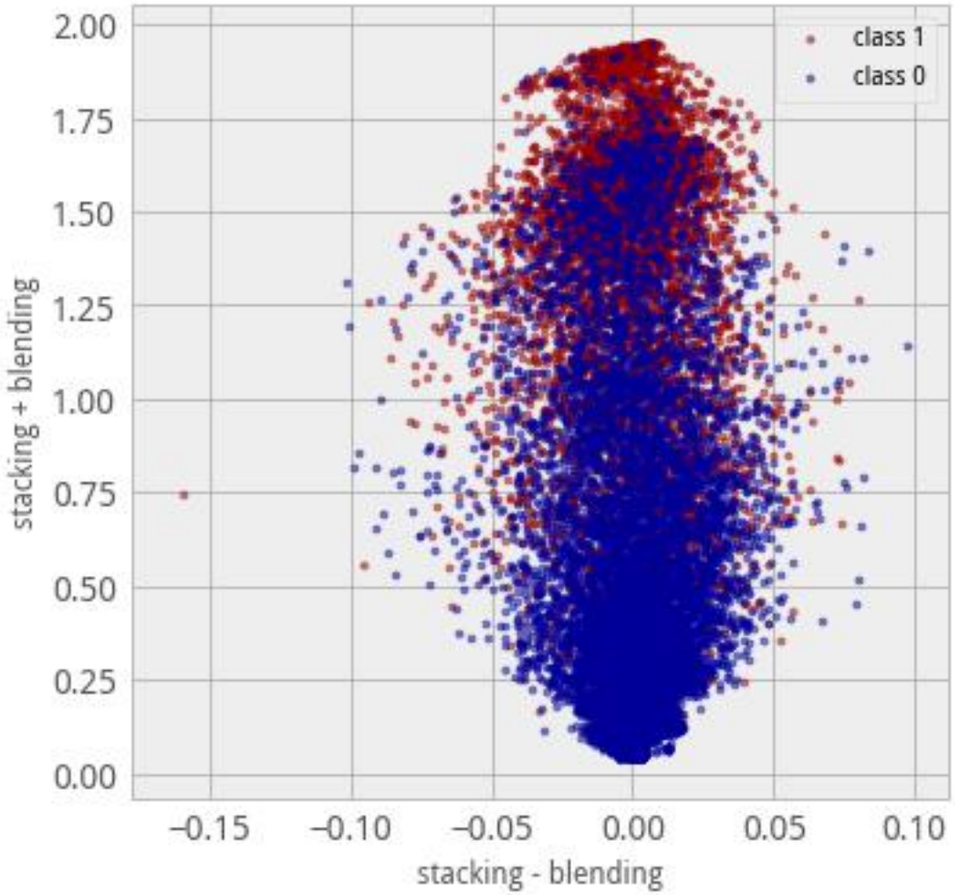


Геометрия стекинга



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_regs + log_reg может не справится с Log_loss

- **Многоуровневый стекинг**

Оправдан только в спортивном анализе данных

Стекинг

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

Но нет хорошей теории на эту тему

- используют, как правило, регрессоры

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)

- **Появляются дополнительные параметры**

количество фолдов, уровень шума

Простейший стекинг – кодирование категорий

mean-target-encoding

~

использование байесовского алгоритма

для формирования мета-признака

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
```

можно по max вероятности среди 4х вероятностей принадлежности к 4м классам

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

```
0 – 111---
1 – 0--11-
2 – -0-0-1
3 – --0-00
```

прочерк – объекты соответствующего класса не участвуют в задаче

можно сложить вероятности принадлежности к классам для каждого класса и по max

ECOC

3. Допустима произвольная кодировка классов:

```
0 – 00
1 – 10
2 – 01
3 – 11
```

это минимальная кодировка, но тут высока цена ошибки бинарного классификатора

4. В том числе, с помощью ECOC

```
0 – 000111
1 – 011100
2 – 101010
3 – 110001
```

Литература

Статья про ансамбли

Dietterich, T. G. (2000). «Ensemble Methods in Machine Learning» // First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science (pp. 1-15). New York: Springer Verlag.

Предложен Feature-Weighted Linear Stacking

Sill, J.; Takacs, G.; Mackey, L.; Lin, D. (2009). «Feature-Weighted Linear Stacking». arXiv:0911.0460.

Бэгинг и аналогичные идеи:

L. Breiman, Pasting small votes for classification in large databases and on-line, Machine Learning, 36(1), 85-103, 1999.

L. Breiman, Bagging predictors, Machine Learning, 24(2), 123-140, 1996.

T. Ho, The random subspace method for constructing decision forests, Pattern Analysis and Machine Intelligence, 20(8), 832-844, 1998.

G. Louppe and P. Geurts, Ensembles on Random Patches, Machine Learning and Knowledge Discovery in Databases, 346-361, 2012.

Ансамбли в машинном обучении

https://dyakonov.org/2019/04/19/ансамбли-в-машинном-обучении/

AdaBoost – немного другой вывод:

http://www.cs.toronto.edu/~rgrosse/courses/csc411_f18/slides/lec09-slides.pdf