



Случайные леса

Александр Дьяконов

07 декабря 2020 года

План

Об универсальности метода
Случайный лес (RF)
Параметры метода
Важность признаков
ExtraTrees
Приложения

Универсальные методы

метод	классификация	регрессия	кластеризация	выбросы
Основанные на близости (~kNN) – проблема перебора – настройки параметров – выбора метрики	+	+	+	+
SVM – узкая группа задач – нужны однородные признаки в одной шкале	+	+	– Формально [Winters-Hilt, 2007]	+
Случайные леса – плохо работает с линейными закономерностями	+	+	+ / – можно построить метрику	+
Нейронные сети – нужна большая выборка	+	+	+ / –	+

Нет полностью универсальных методов! Всё идёт от задачи...

Случайные леса

+ наиболее универсальный
(~75% задач машинного обучения)

+ все типы задач
(классификация, регрессия, кластеризация, аномалии)

+ настраивается «сразу под все функционалы»
(или можно преобразовать – не всегда надо)

+ нечувствителен к монотонным преобразованиям признаков
не совсем так! только к линейному масштабированию

+ легко реализуется

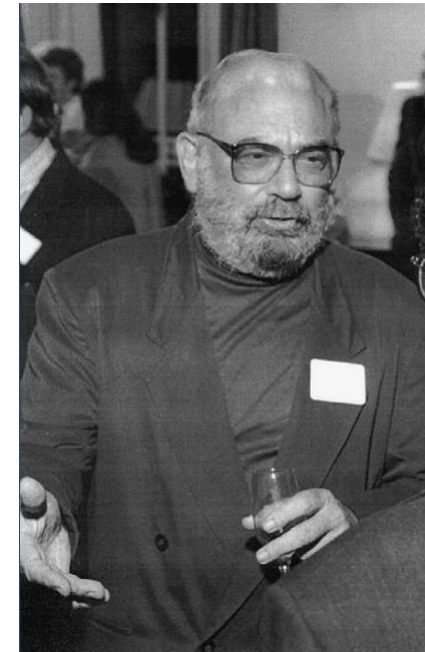
Случайный лес (Random Forest)

– специальный метод ансамблирования

**= бэггинг + специальное построение деревьев
(подмножество признаков при расщеплении)**

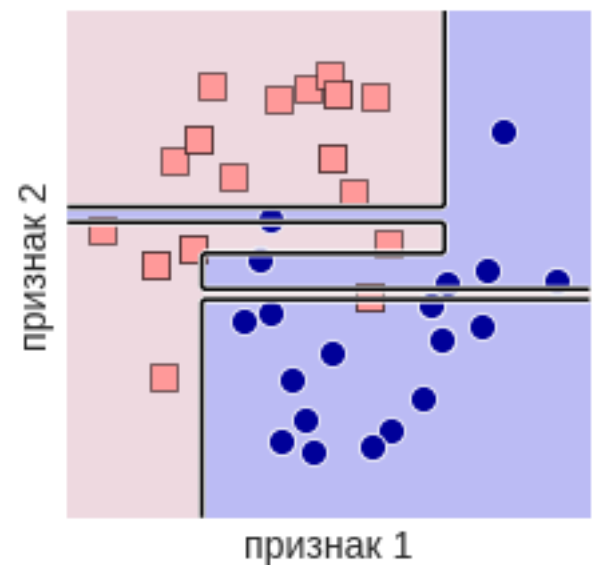
Качество одного дерева очень низкое!

$$\frac{1}{N_{\text{tree}}} \left(\begin{array}{c} \square \\ \swarrow \quad \searrow \\ \square \end{array} + \begin{array}{c} \square \\ \swarrow \quad \searrow \\ \square \quad \square \end{array} + \dots + \begin{array}{c} \square \\ \swarrow \quad \searrow \\ \square \quad \square \\ \quad \quad \searrow \\ \quad \quad \square \end{array} \right)$$

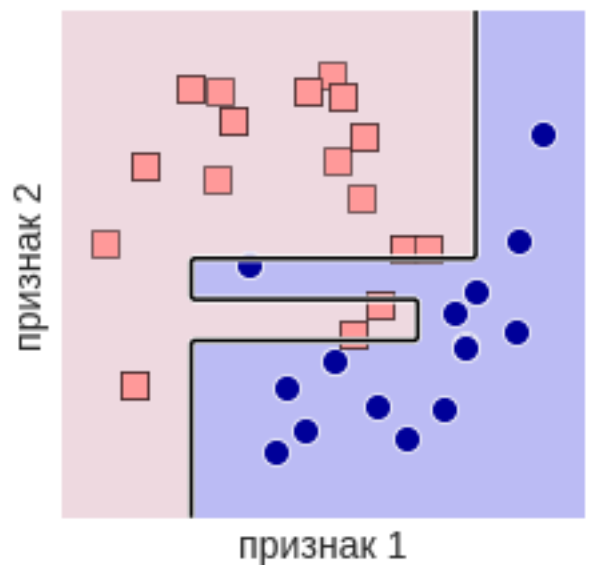


Лео Брейман, 1928 – 2005

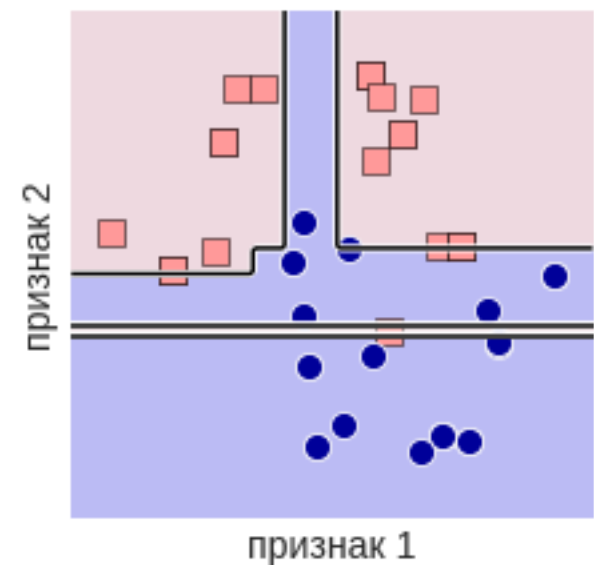
Случайный лес (Random Forest)



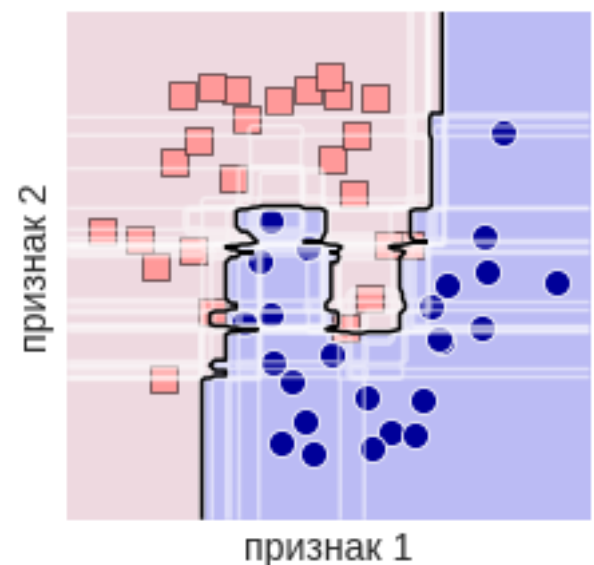
дерево № 1



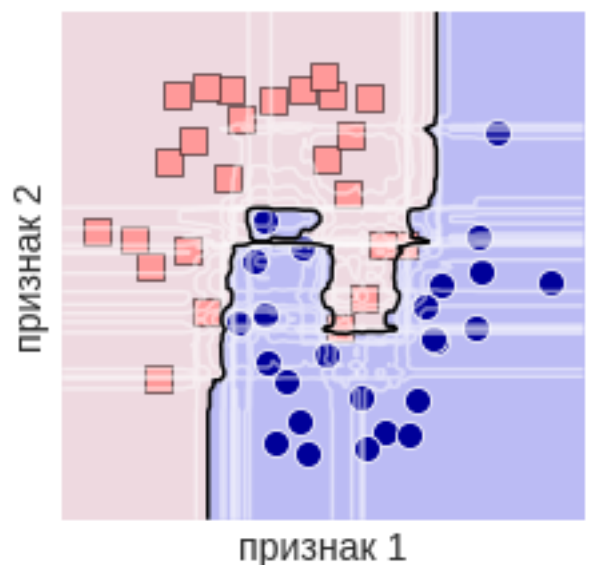
дерево № 2



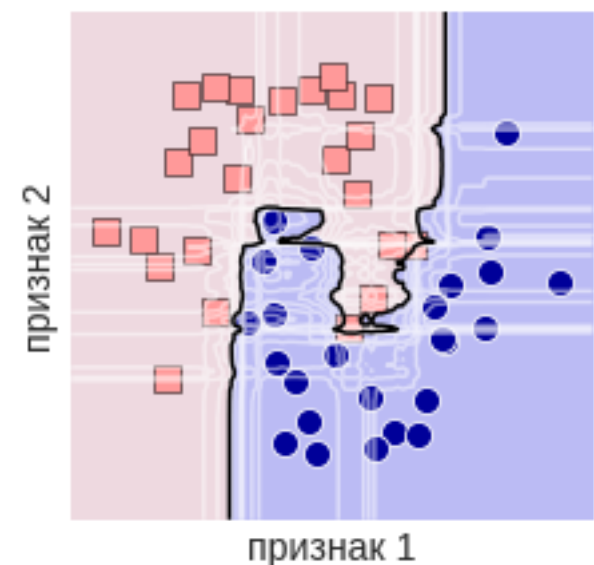
дерево № 3



RF, число деревьев=10



RF, число деревьев=100



RF, число деревьев=1000

Построение случайного леса

1. Выбирается подвыборка `samplesize` (м.б. с повторением) – на ней строится дерево
2. Строим дерево
 - 2.1. Для построения каждого расщепления просматриваем `mtry` / `max_features` случайных признаков
 - 2.2. Как правило, дерево строится до исчерпания выборки (без прунинга)

Ответ леса:

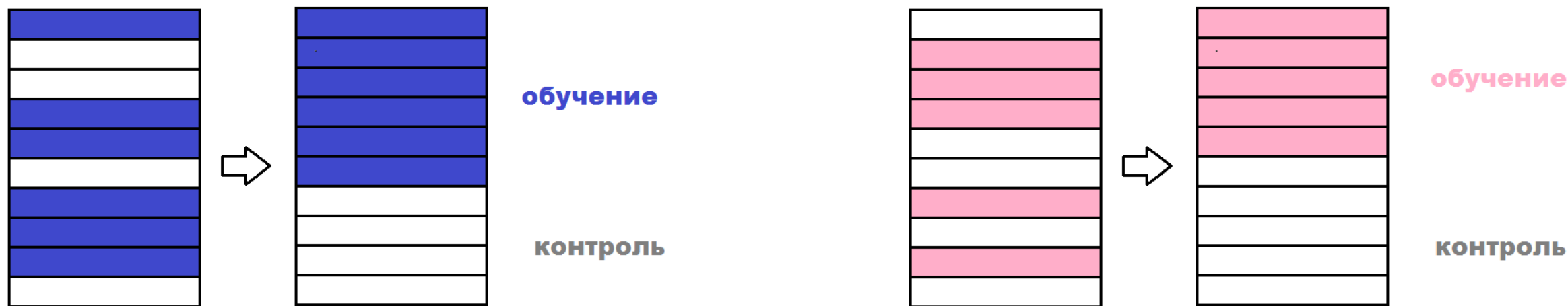
по большинству (в задачах классификации),
среднее арифметическое (в задачах регрессии)

Автоматически: рейтинг признаков –

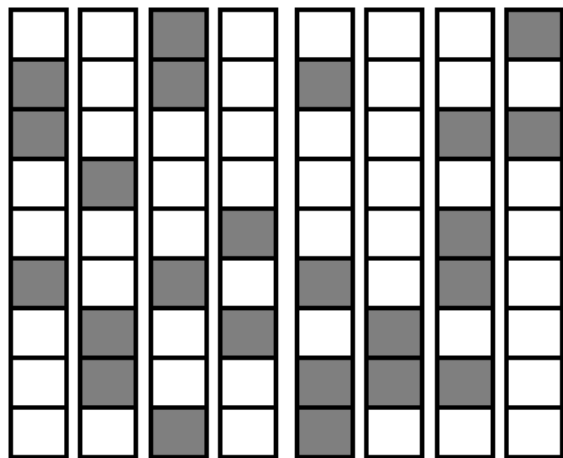
`importance(model) / .feature_importances_`

Автоматически: уверенность в ответе – дисперсия ответов деревьев

Бэггинг и OOB (out of bag)



**Выбор объектов для обучения (с помощью бутстрепа),
остальные – локальный контроль...**



Ответы разных деревьев – можно усреднить и вычислить качество

Вспомним реализацию решающего дерева

```
sklearn.tree import DecisionTreeClassifier
```

criterion	критерий расщепления «gini» / «entropy»
splitter	разбиение «best» / «random»
max_depth	допустимая глубина
min_samples_split	минимальная выборка для разбиения
min_samples_leaf	минимальная мощность листа
min_weight_fraction_leaf	аналогично с весом
max_features	число признаков, которые смотрим для нахождения разбиения
random_state	инициализация генератора случайных чисел
max_leaf_nodes	допустимое число листьев
min_impurity_decrease	порог «зашумлённости» для разбиения
min_impurity_split	порог «зашумлённости» для останова
class_weight	веса классов («balanced» или словарь, список словарей)

Теперь – случайный лес

```
sklearn.ensemble import RandomForestClassifier
```

n_estimators=100	число деревьев
criterion='gini'	критерий расщепления «gini» / «entropy»
splitter	разбиение «best» / «random»
max_depth=None	допустимая глубина
min_samples_split=2	минимальная выборка для разбиения
min_samples_leaf =1	минимальная мощность листа
min_weight_fraction_leaf=0.0	аналогично с весом
max_features='auto'	число признаков, которые смотрим для нахождения разбиения
max_leaf_nodes=None	допустимое число листьев
min_impurity_decrease=0.0	порог изменения «зашумлённости» для разбиения
min_impurity_split=None	порог «зашумлённости» для останова
class_weight	веса классов («balanced» или словарь, список словарей)
bootstrap=True	делать ли бутстреп
oob_score	вычислять ли OOB-ошибку
warm_start	использовать ли существующий лес, чтобы его дополнить или учить заново

```
n_jobs, random_state, verbose
```

Особенности

Изменение impurity – порог на

$$\frac{|R|}{m} \left(H(R) - \frac{|R_{\text{left}}|}{|R|} H(R_{\text{left}}) - \frac{|R_{\text{right}}|}{|R|} H(R_{\text{right}}) \right)$$

Параметры случайного леса

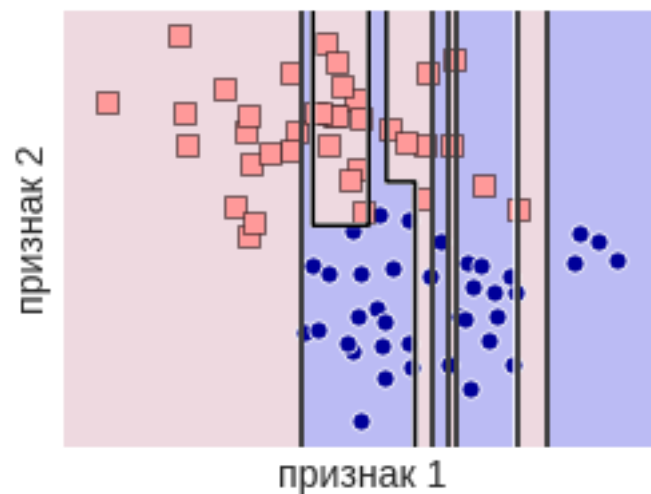
```
class sklearn.ensemble.RandomForestClassifier
    (n_estimators=10,
     criterion='gini',
     max_depth=None,
     min_samples_split=2,
     min_samples_leaf=1,
     min_weight_fraction_leaf=0.0,
     max_features='auto',
     max_leaf_nodes=None,
     bootstrap=True,
     oob_score=False,
     n_jobs=1,
     random_state=None,
     verbose=0,
     warm_start=False,
     class_weight=None)
```

```
{randomForest} randomForest(
  x, y, xtest, ytest,
  ntree=500,
  mtry=if (!is.null(y) &&
    !is.factor(y))
    max(floor(ncol(x)/3), 1) else
    floor(sqrt(ncol(x))),
  replace=TRUE,
  classwt=NULL,
  cutoff,
  strata,
  sampsize = if (replace) nrow(x) else
    ceiling(.632*nrow(x)),
  nodesize = if (!is.null(y) &&
    !is.factor(y)) 5 else 1,
  maxnodes = NULL,
  importance=FALSE, localImp=FALSE,
  nPerm=1,
  proximity, oob.prox=proximity)
```

«Случайный лес»

```
from sklearn.ensemble import RandomForestClassifier  
rf = RandomForestClassifier(n_estimators=1)  
rf.fit(X_train, y_train)
```

n_estimators=1



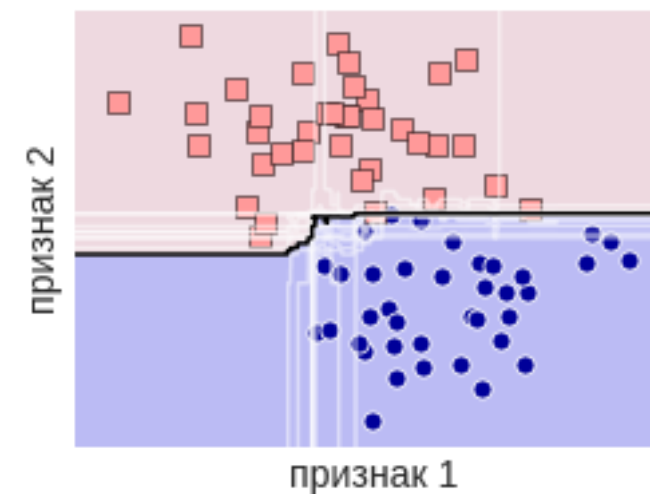
86%

n_estimators=5



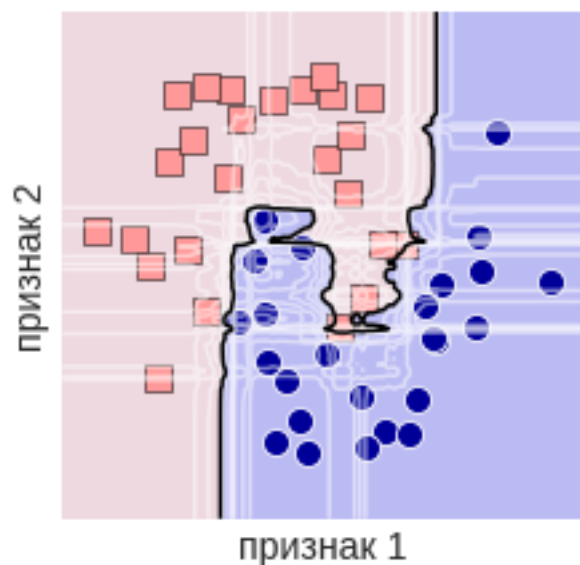
84%

n_estimators=100

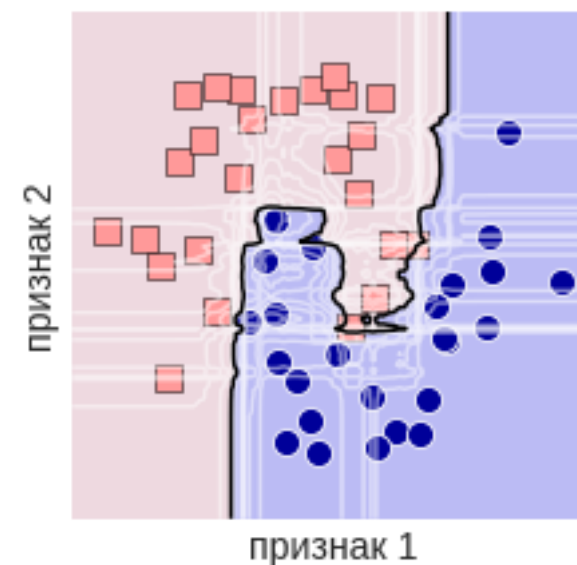


88%

Различные критерии расщепления



`criterion='gini'`



`criterion='entropy'`

в модельных задачах «на глаз» разницы не видно

в авторском коде был реализован Джини...

Настройка параметров: размер подвыборки `samplesize`

1. Определиться с типом выбора
с возвратом / без возврата

2. Настройка по объёму
– не в первую очередь

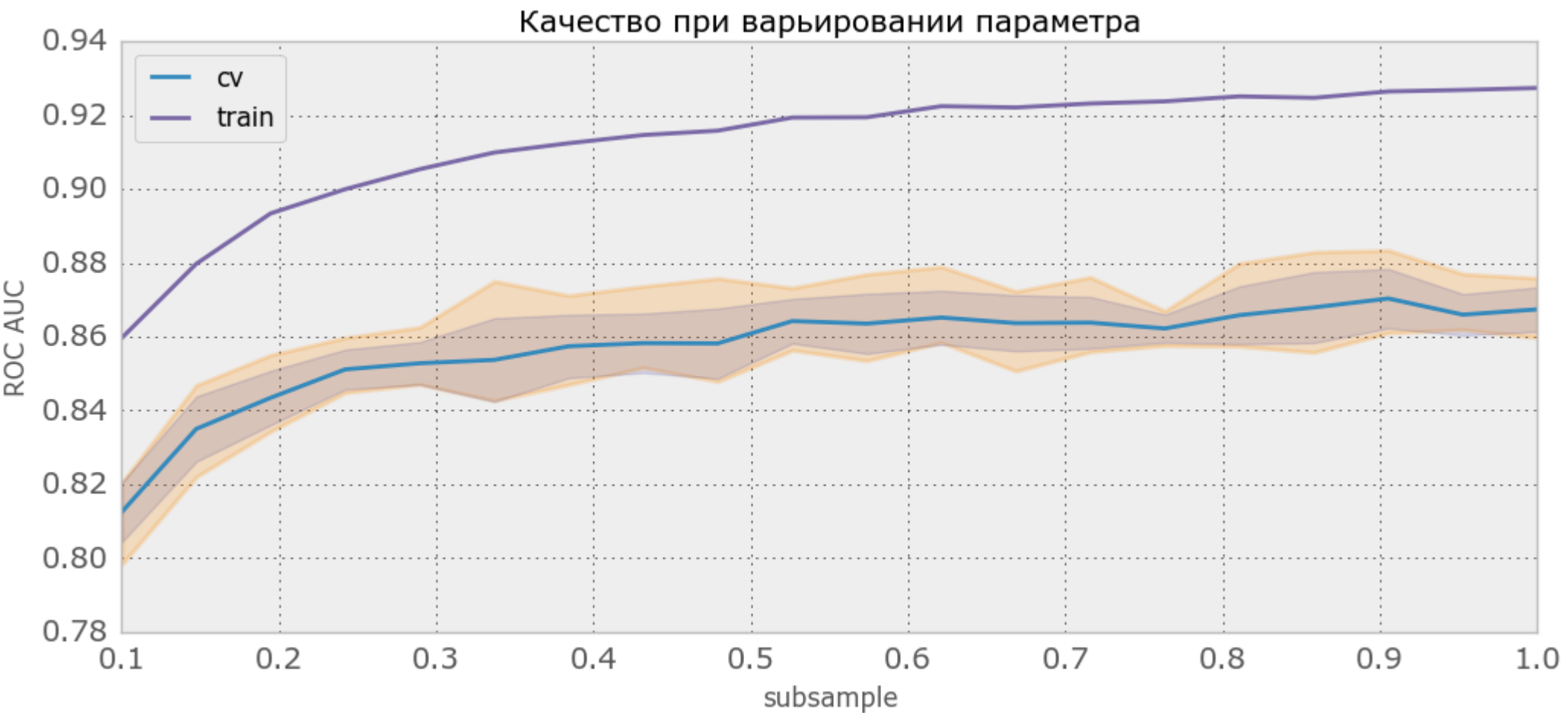
Часто «нужны все объекты»

Чем больше – тем однотипнее деревья

Что из этого следует?

Этого параметра нет в `sklearn`-реализации

Настройка параметров: размер подвыборки `samplesize` (СберБанк)



Всю выборку надо использовать по максимуму!

Настройка параметров: число признаков `mtry` / `max_features`

Самый серьёзный параметр

По умолчанию часто: \sqrt{n} – классификация
 $n / 3$ – регрессия

Зависимость унимодальная
Настраивается в первую очередь

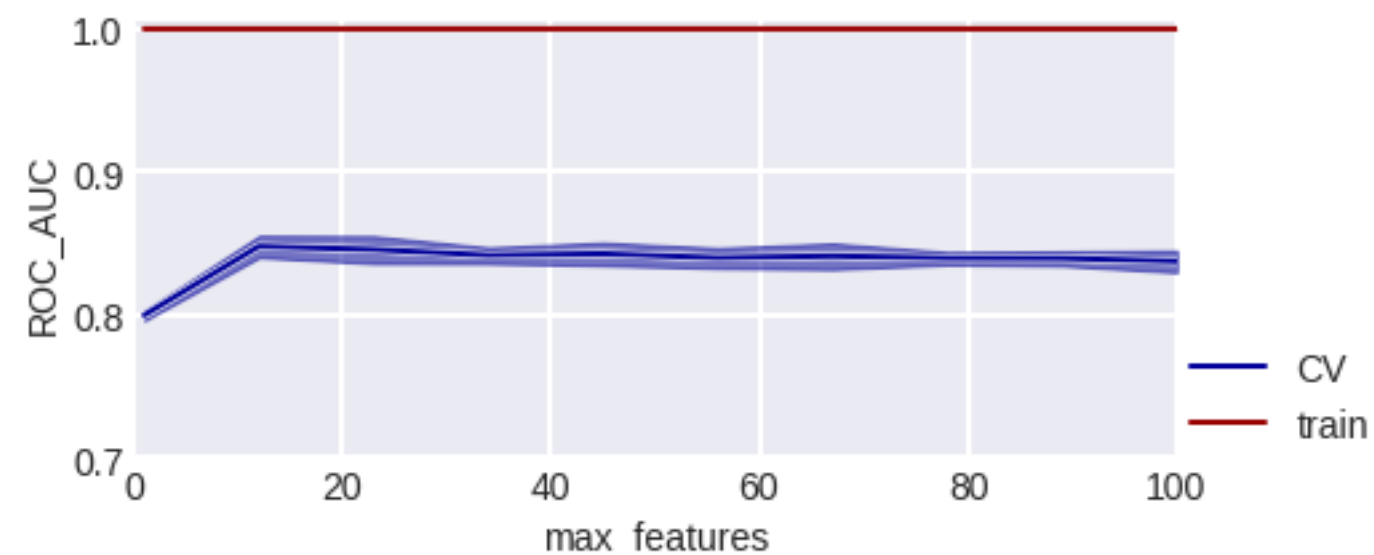
Зависит от числа шумовых признаков
Надо перенастраивать при добавлении новых признаков

Чем больше – тем однотипнее деревья.
Чем больше – тем медленнее настройка!

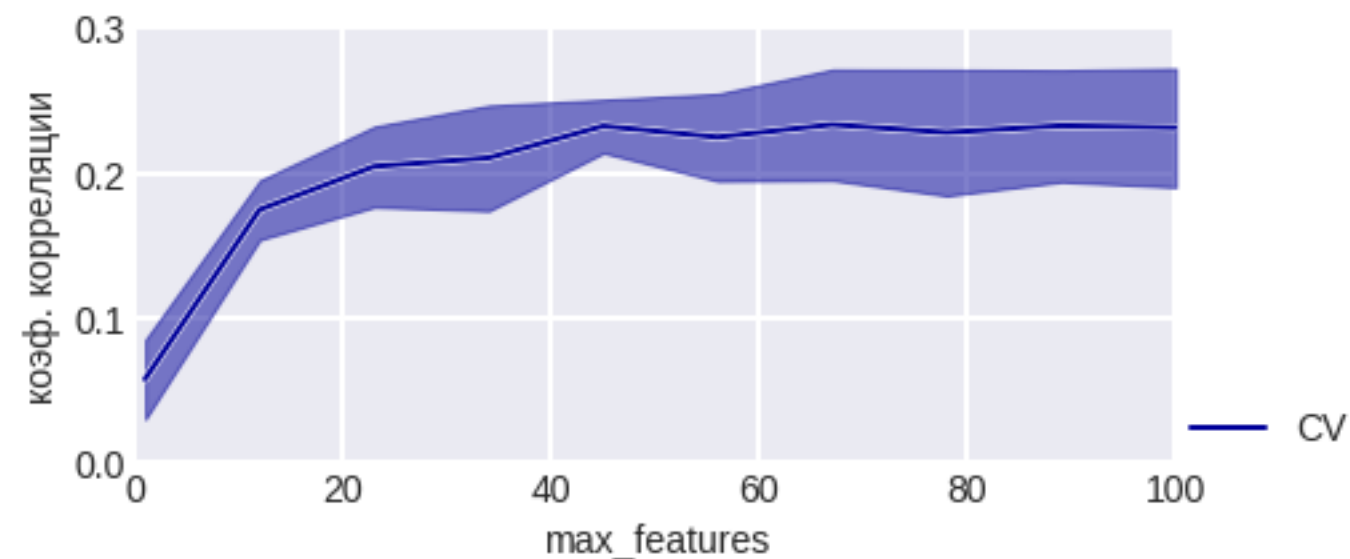
Kaggle: часто суммируют алгоритмы с разными `mtry`.

Настройка `mtry` / `max_features` (СберБанк)

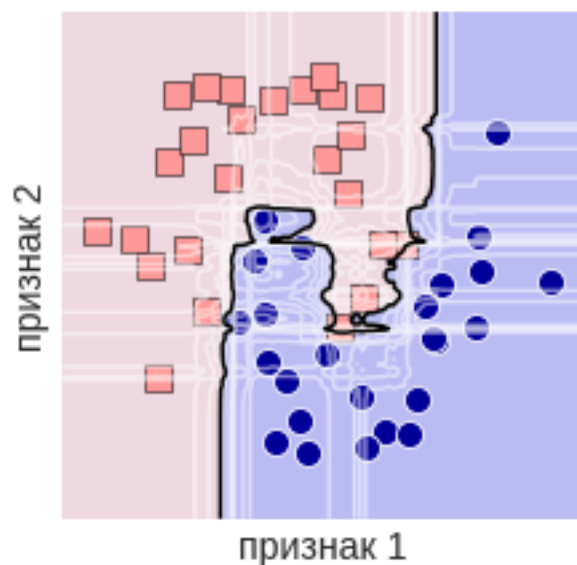
**Качество от числа признаков, как
правило, унимодально**



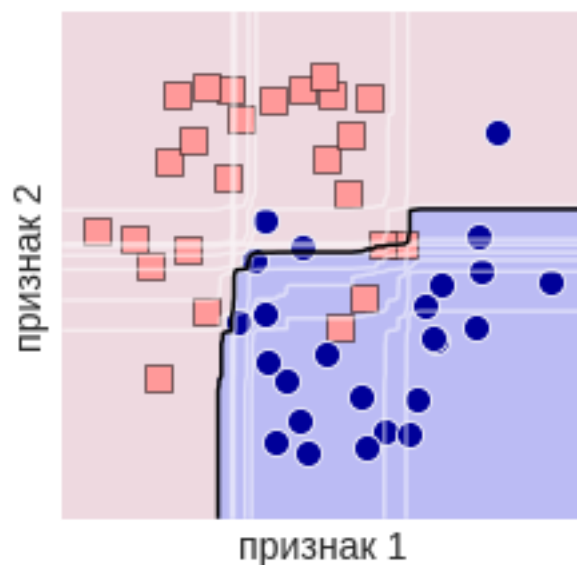
**Коэффициент корреляции между
ответами деревьев леса**



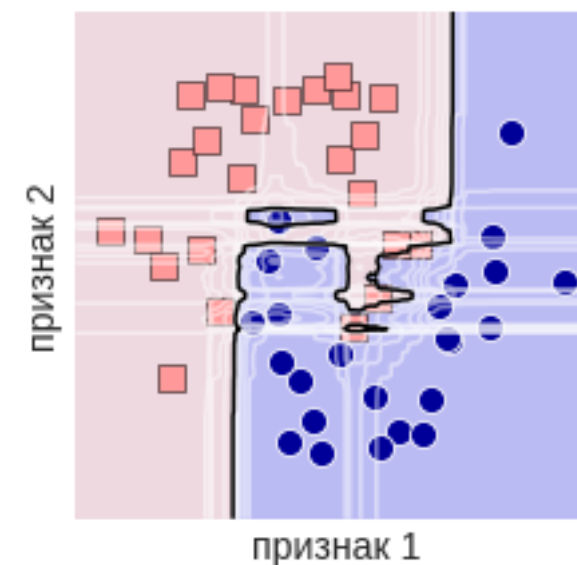
Геометрия `mtry` / `max_features` что можно сказать?



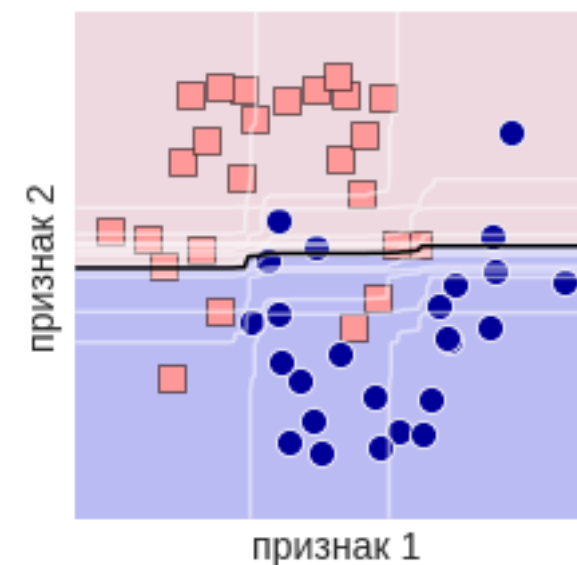
`max_features=1`



`max_depth=1, max_features=1`

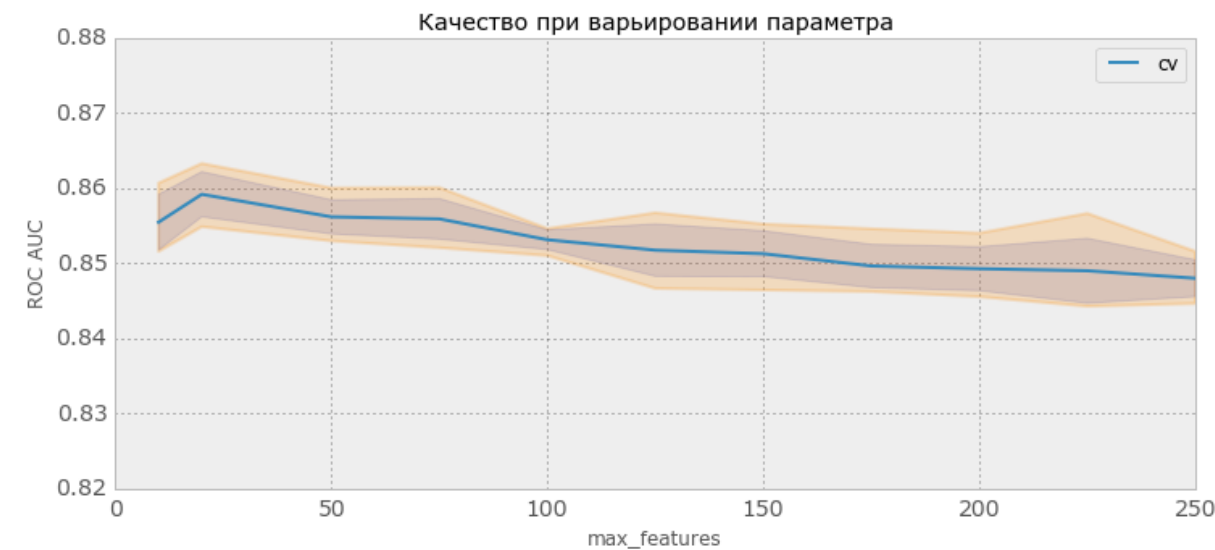


`max_features=2`

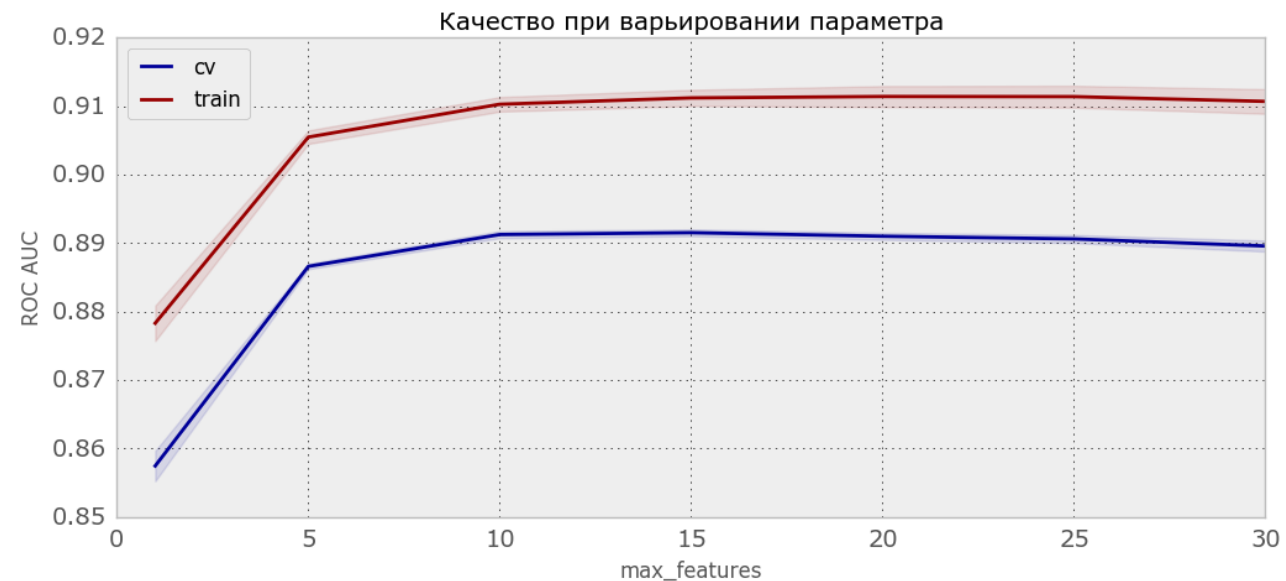


`max_depth=1, max_features=2`

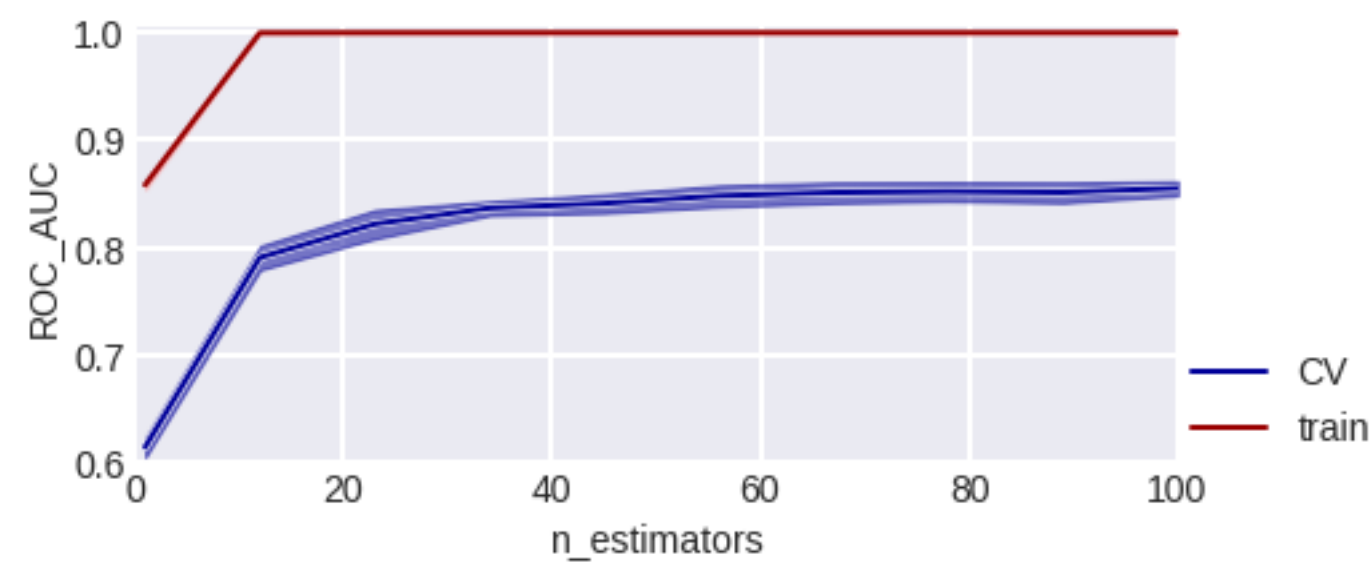
Настройка mtry / max_features



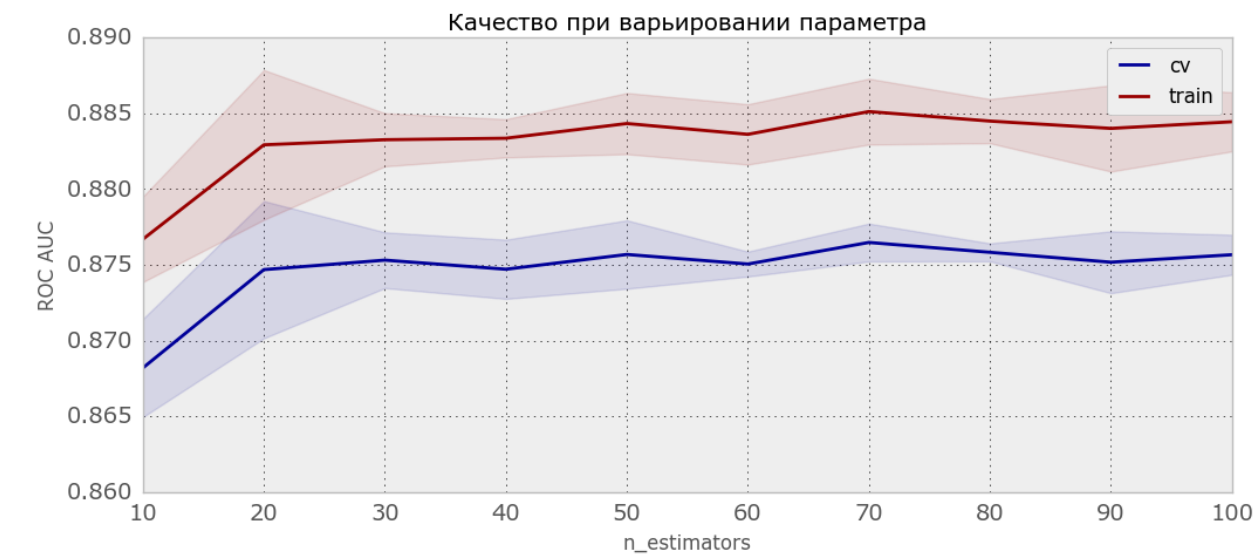
(ed Бозон) в задаче ~ 33 признака



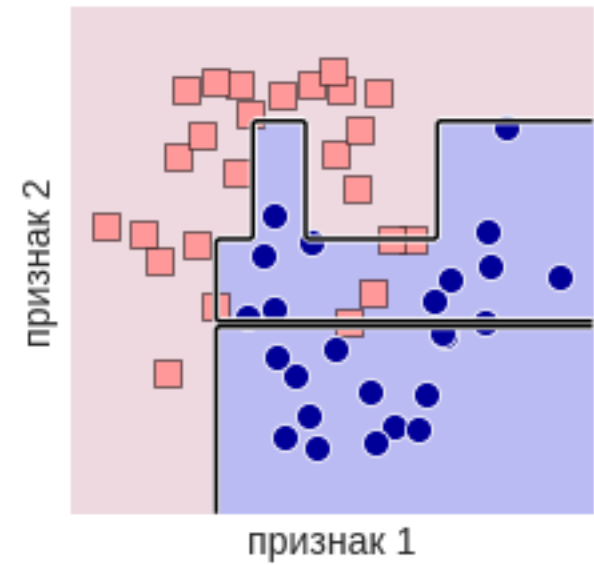
Настройка параметров `ntree` / `n_estimators` (СберБанк)
Чем больше деревьев – тем лучше!



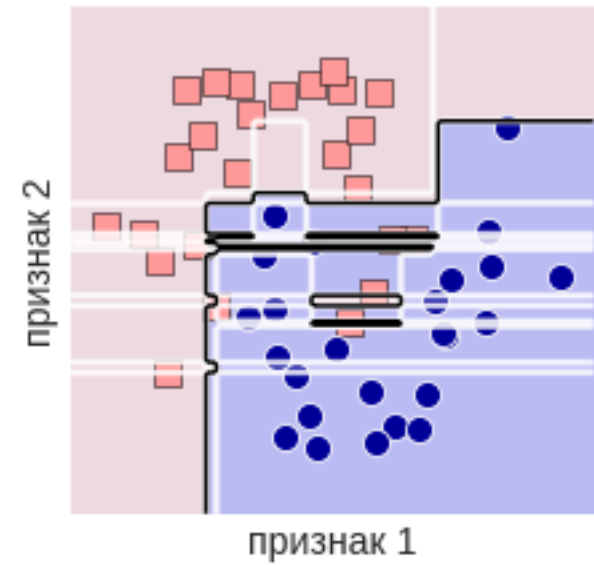
(ed Бозон)



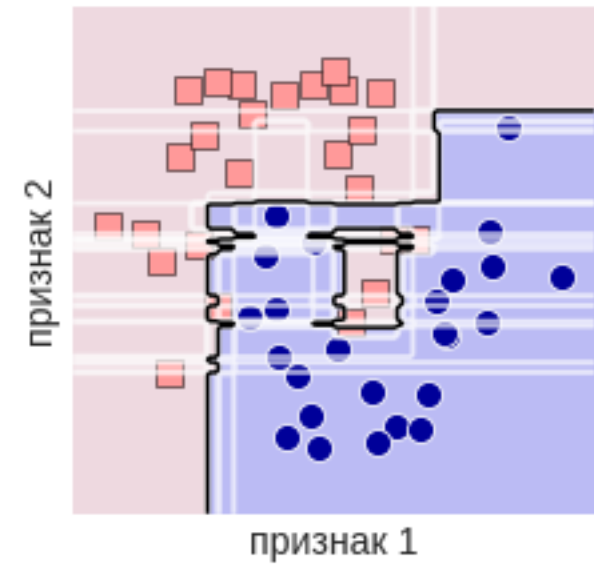
Настройка параметров `ntree` / `n_estimators`



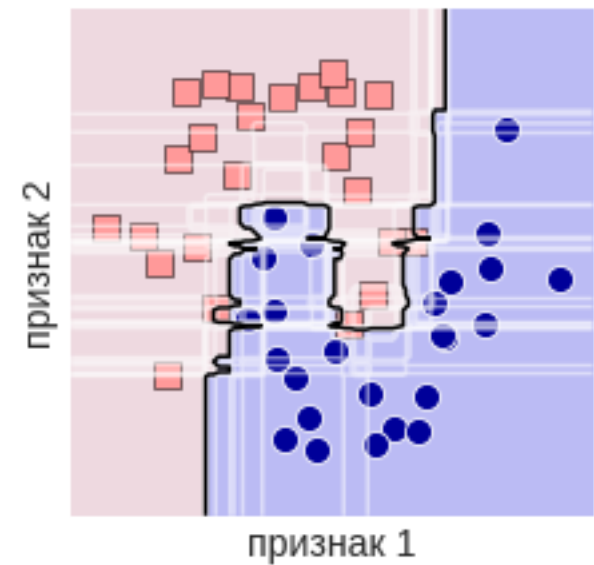
RF, число деревьев=1



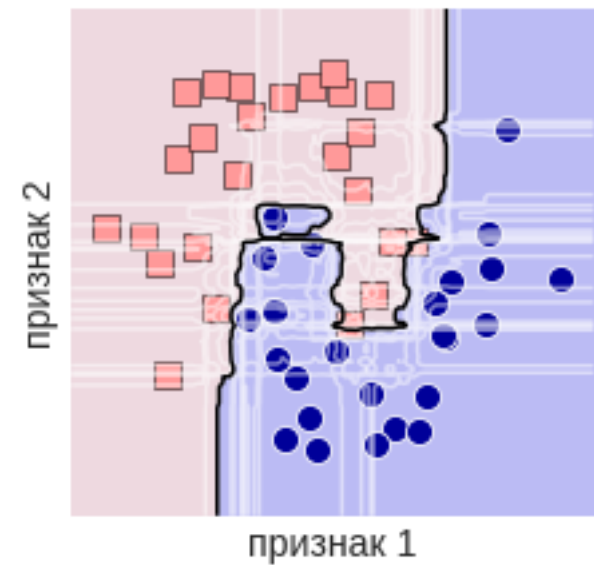
RF, число деревьев=3



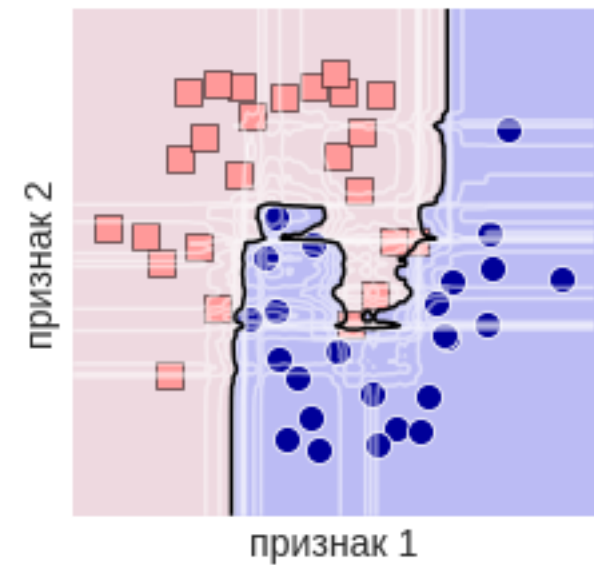
RF, число деревьев=5



RF, число деревьев=10



RF, число деревьев=100



RF, число деревьев=1000

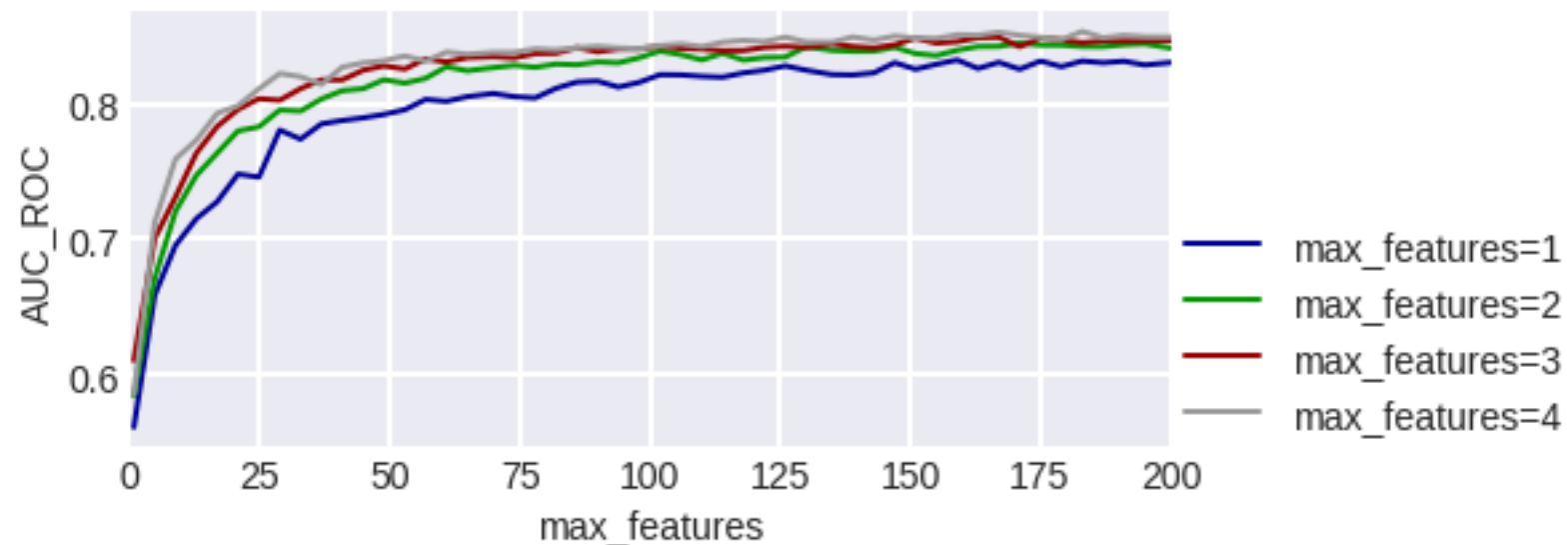
Настройка параметров `ntree` / `n_estimators` (СберБанк)

Чем больше – тем лучше!

Проблемы:

- **как использовать при настройке параметров очень большое число деревьев**
- **что делать, если не помещаются в память... (например, в R)**
можно строить по одному, получать ответ на тесте, не хранить в памяти

Совет по числу деревьев: область устойчивости функционала

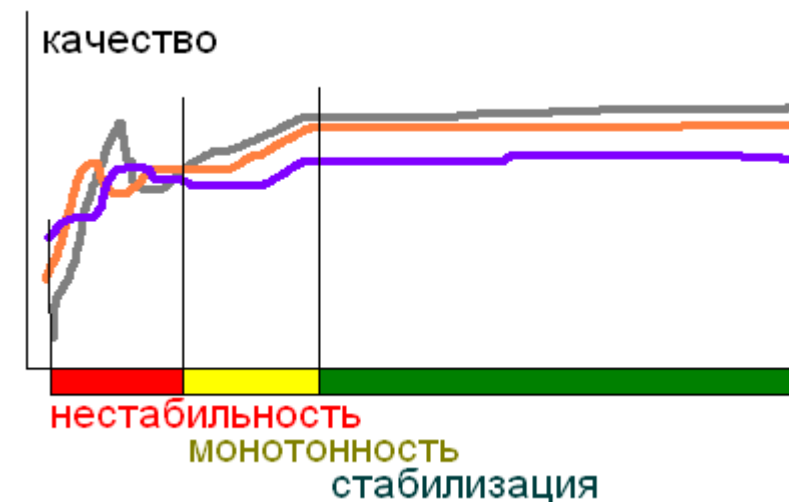


Неверная подпись – n_estimators

Аналогично в подобных ситуациях:



Обычно три зоны...



**AUC в зависимости от объёма
контрольной выборки**

Иногда причины некорректности:
алгоритм выдаёт похожие
(совпадающие) оценки
(AUC вычисляется в
среднем/худшем случае)

Настройка параметров: ограничения в листьях

число объектов в листе,
число объектов для расщепления,
максимальная глубина дерева

От параметров существенно зависит скорость построения леса

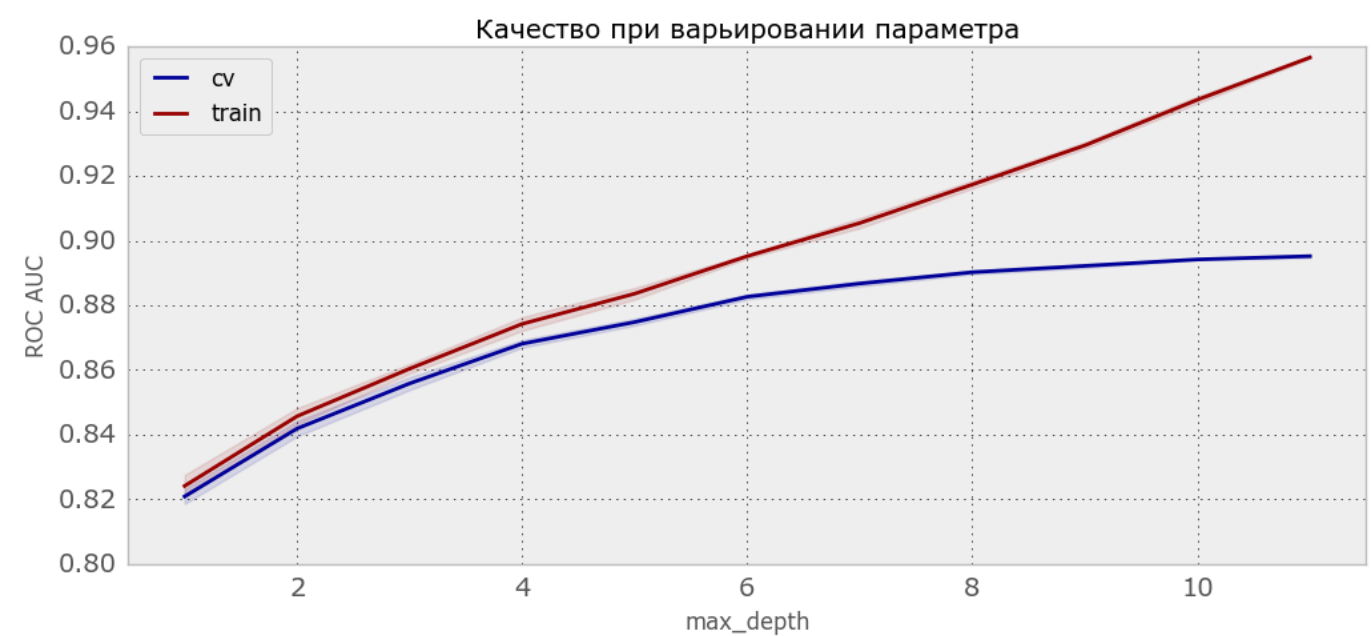
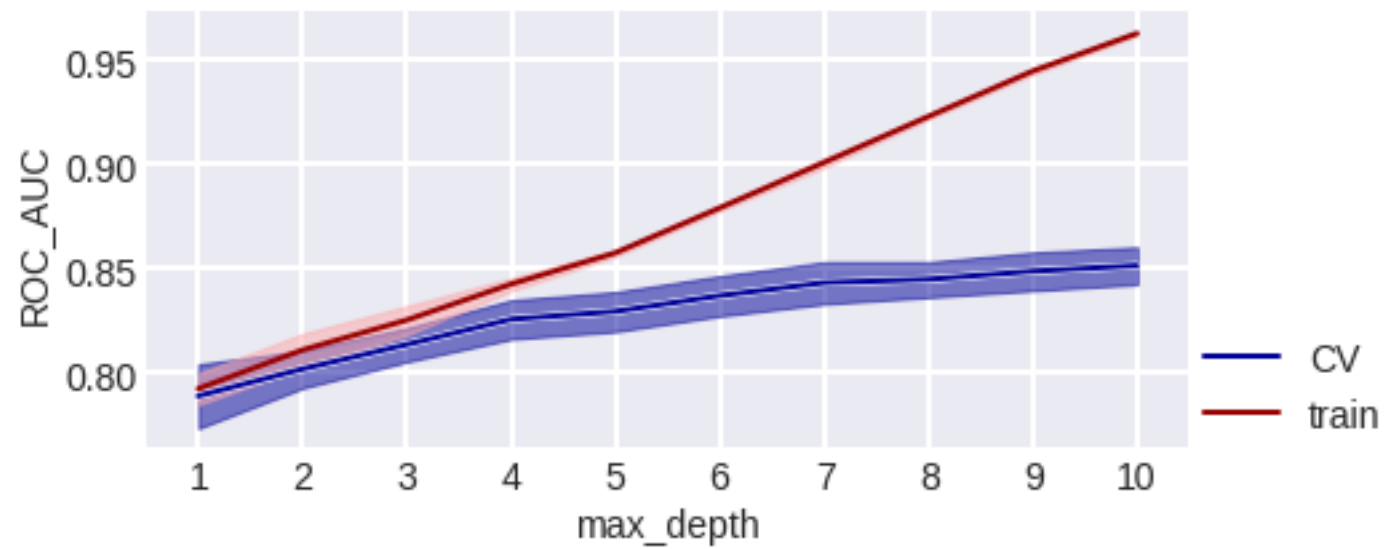
Оптимальные значения, как правило, – несколько объектов в листе.

Настраиваются не в первую очередь

В классическом случайном лесе деревья строятся до исчерпания выборки...

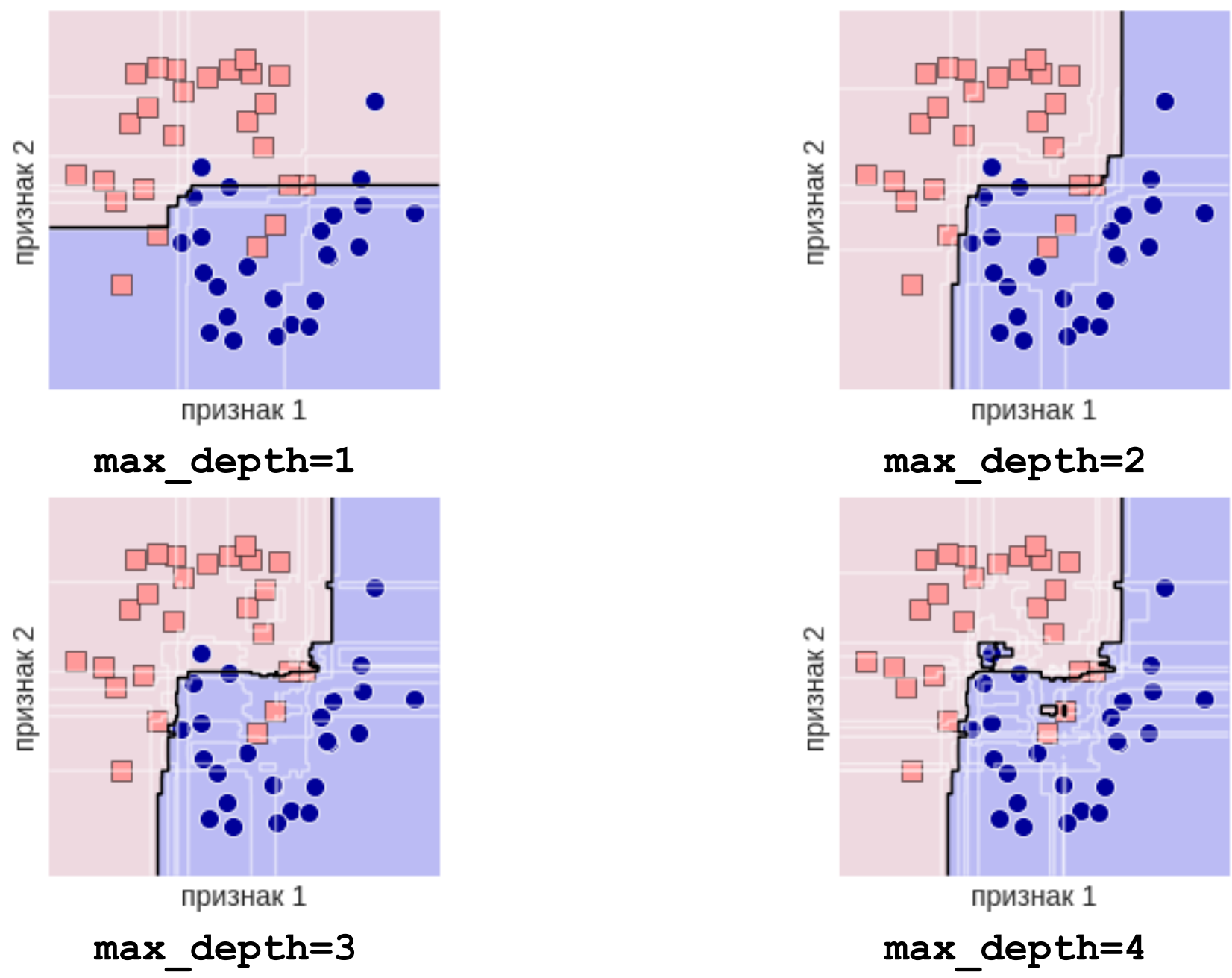
«Good results are often achieved when setting `max_depth=None` in combination with `min_samples_split=1`»

Глубина дерева: max_depth (СберБанк)



Как правило, чем больше, тем лучше!

Глубина дерева: max_depth

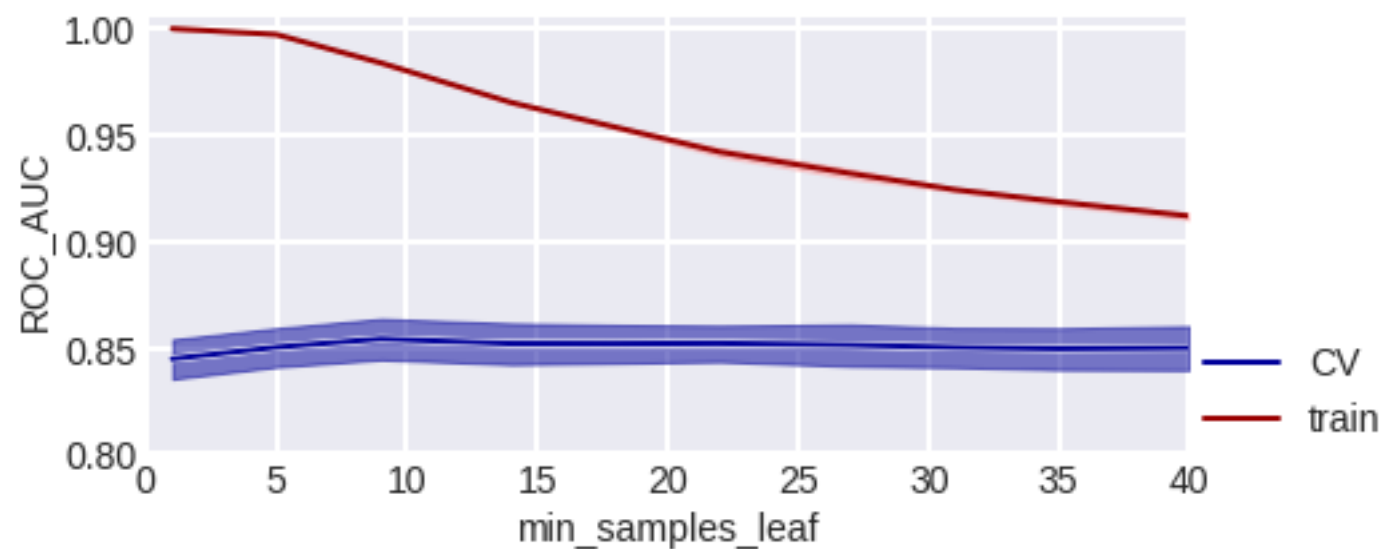


Глубина дерева: `max_depth`

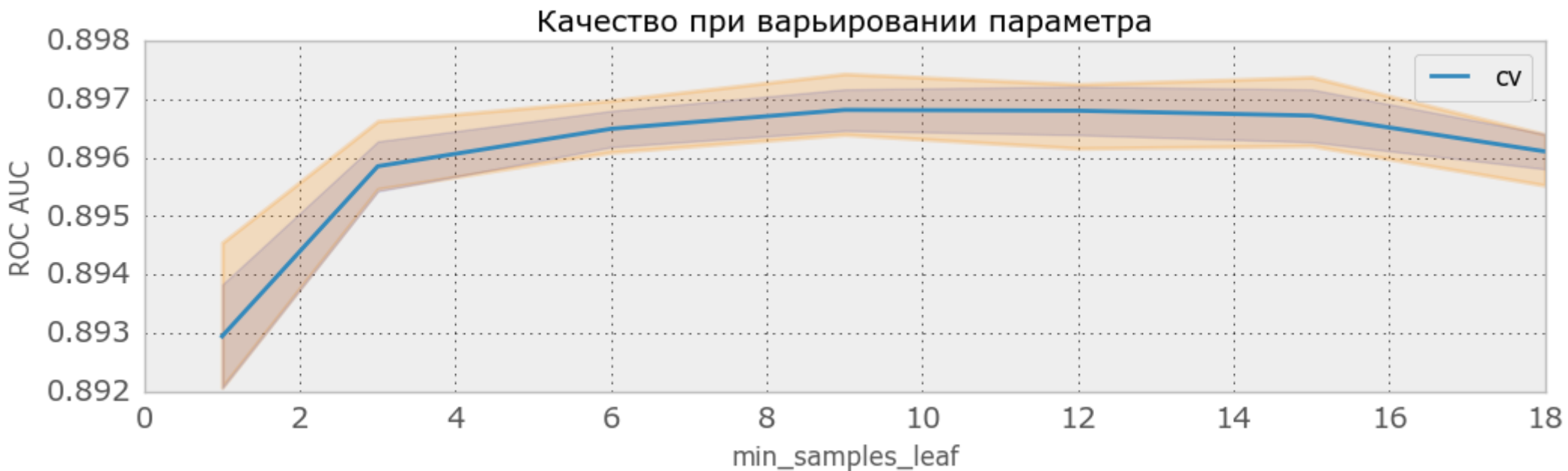
Неглубокие деревья:

- в задачах с выбросами
- когда много объектов
(деревья большие и долго строятся)
- настройка некоторых других (**каких?**) параметров
не имеет смысла

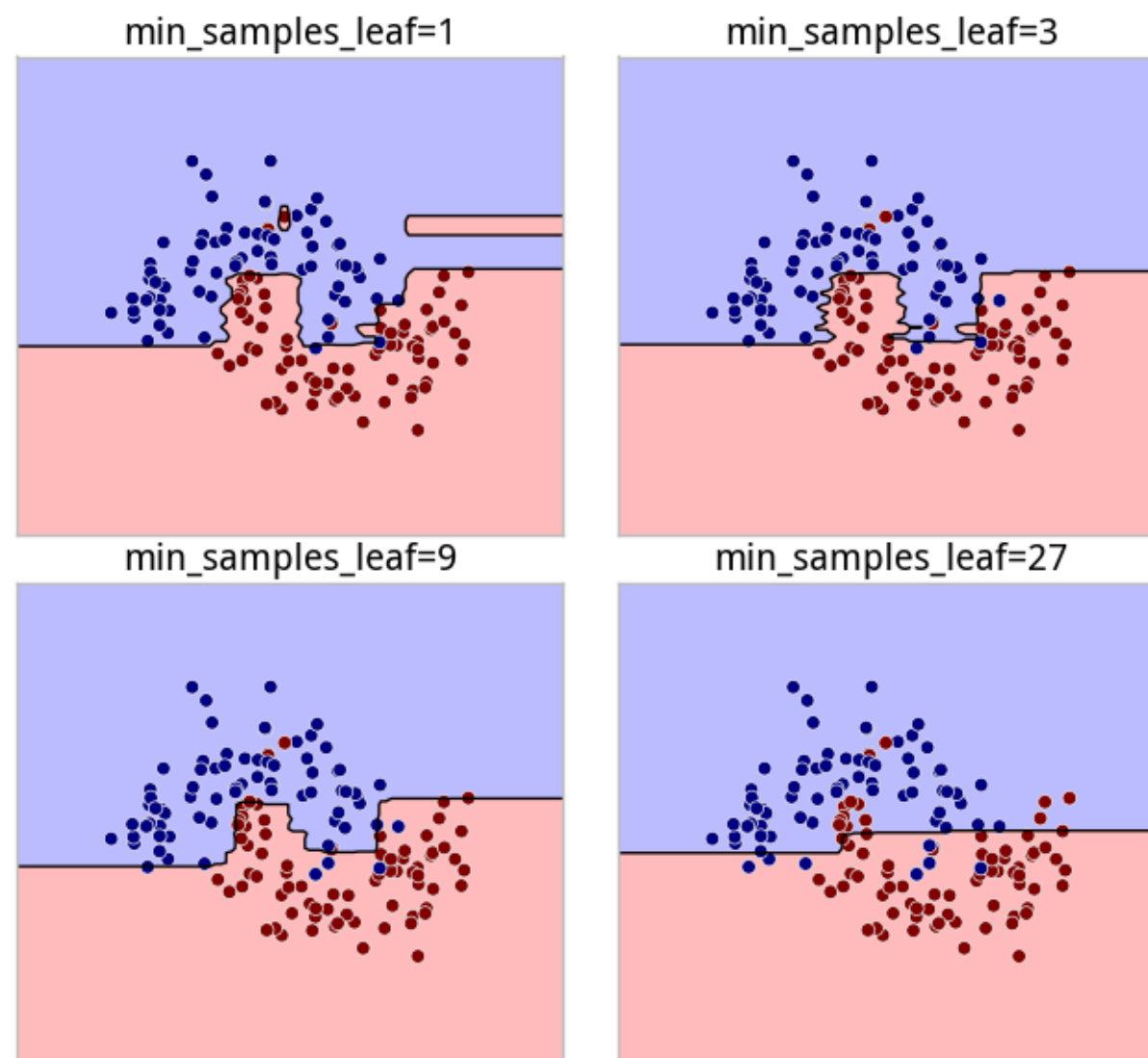
RandomForestClassifier: min_samples_leaf



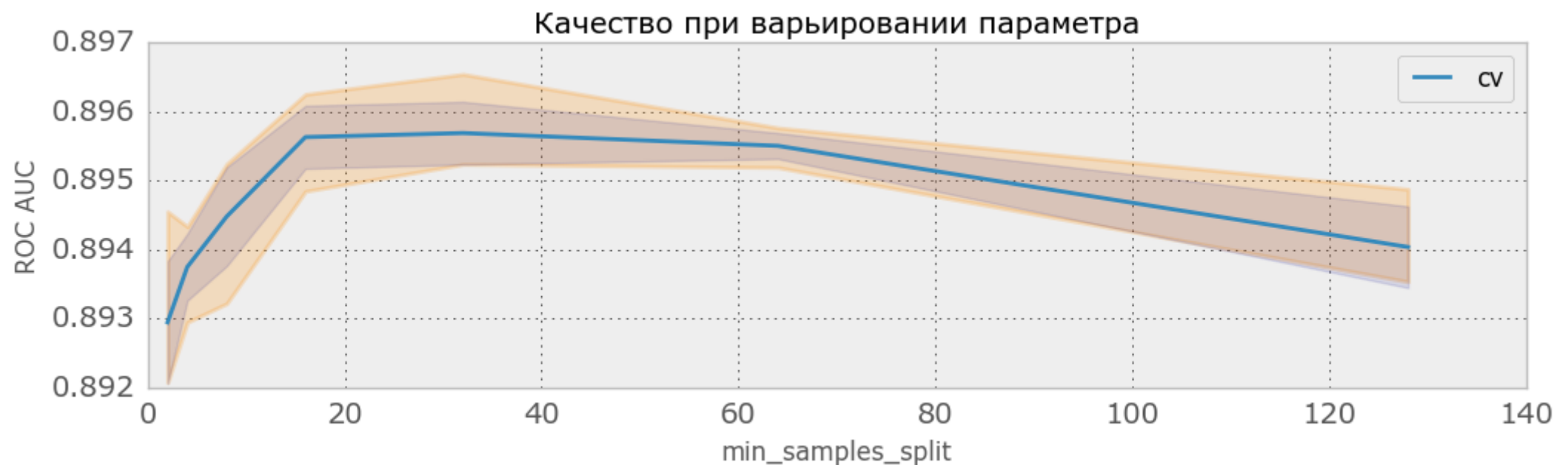
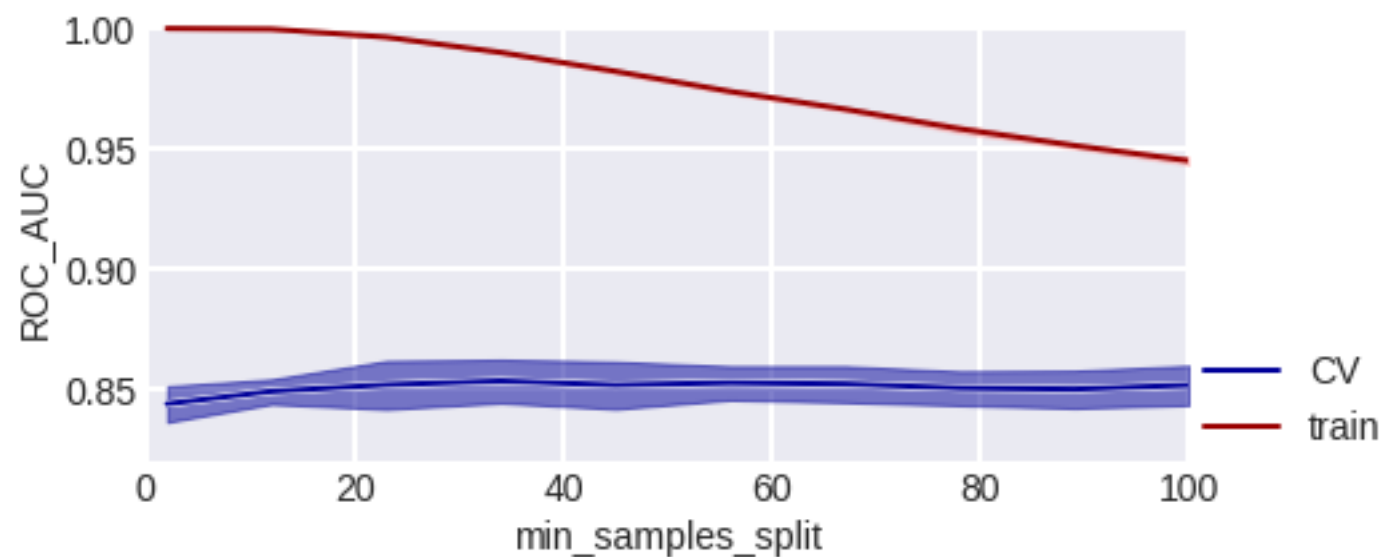
умолчание: 1 – классификация, 5 – регрессия



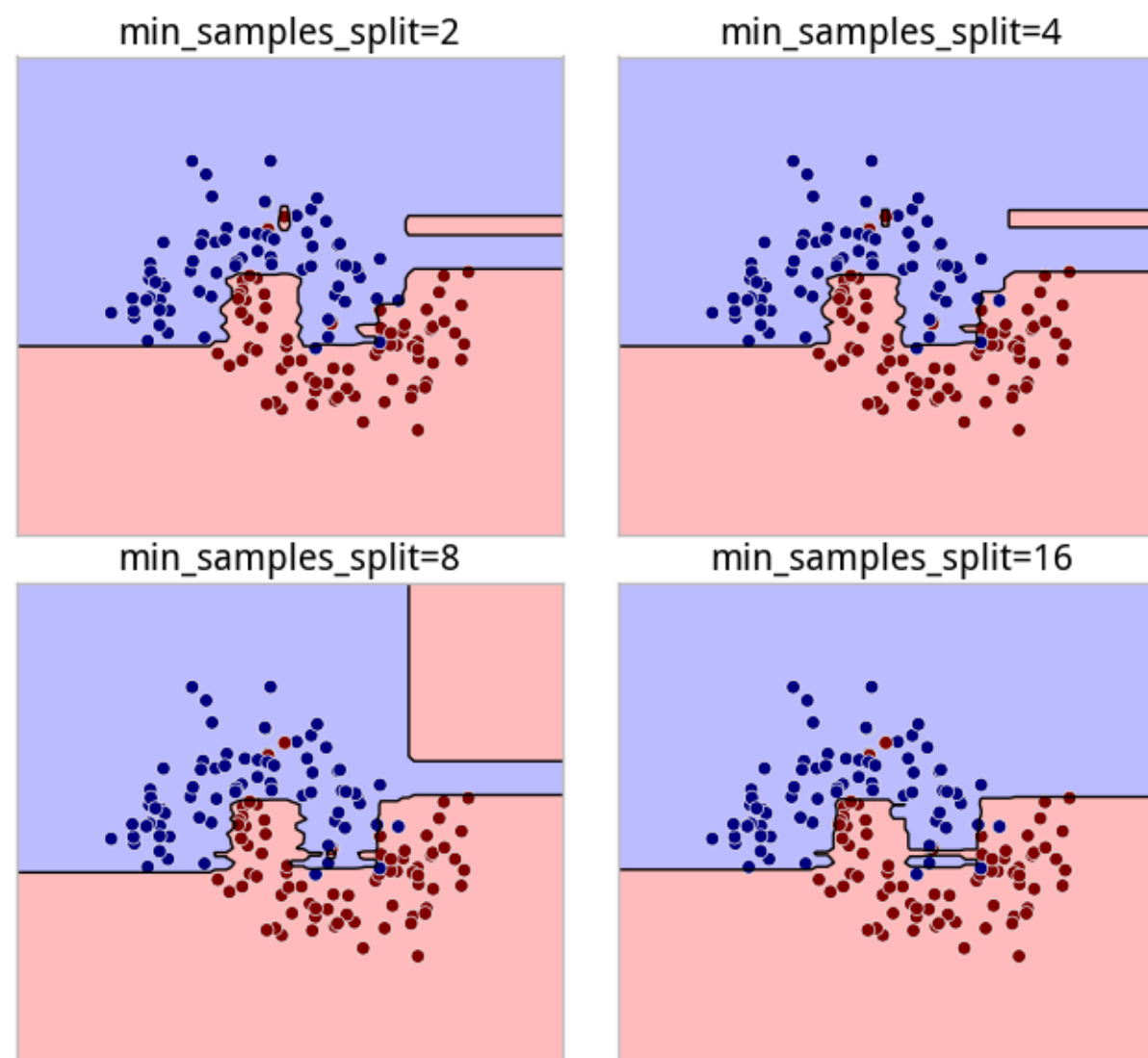
RandomForestClassifier: `min_samples_leaf`



RandomForestClassifier: min_samples_split



RandomForestClassifier: min_samples_split



Важности признаков в RF

отдельная тема

Проблемы RF

Может долго считаться...

Вместо CV – разбиение на обучение и контроль (hold out)

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.33, random_state=42)
```

sklearn: не забывать n_jobs

Какие тонкости?

ответ ~ ответ на LB

не потерять важные части обучения (сохранение пропорции классов, представительности признаков и т.п.)

размер контроля = область устойчивости функционала

Proximity

при построении деревьев можно много чего считать...

**Чем чаще 2 объекта попадают в один лист,
тем они ближе...**

Какую метрику можно придумать?

Extreme Random Trees (ExtraTrees)

- нет бутстрепа (используем всю выборку)
 - генерируем несколько пар (признак, порог)
 - выбираем оптимальную для разбиения пару
 - также есть параметр «число признаков для просмотра»
-
- ET быстрее RF
 - ET чуть хуже RF, когда много шумных признаков

```
from sklearn.ensemble import ExtraTreesClassifier
clf = ExtraTreesClassifier(n_estimators=10, max_depth=None,
                           min_samples_split=2, random_state=0)
```

Ансамбли на базе RF

Синтетический случайный лес (Synthetic RF) – стекинг лесов с разным nodesize

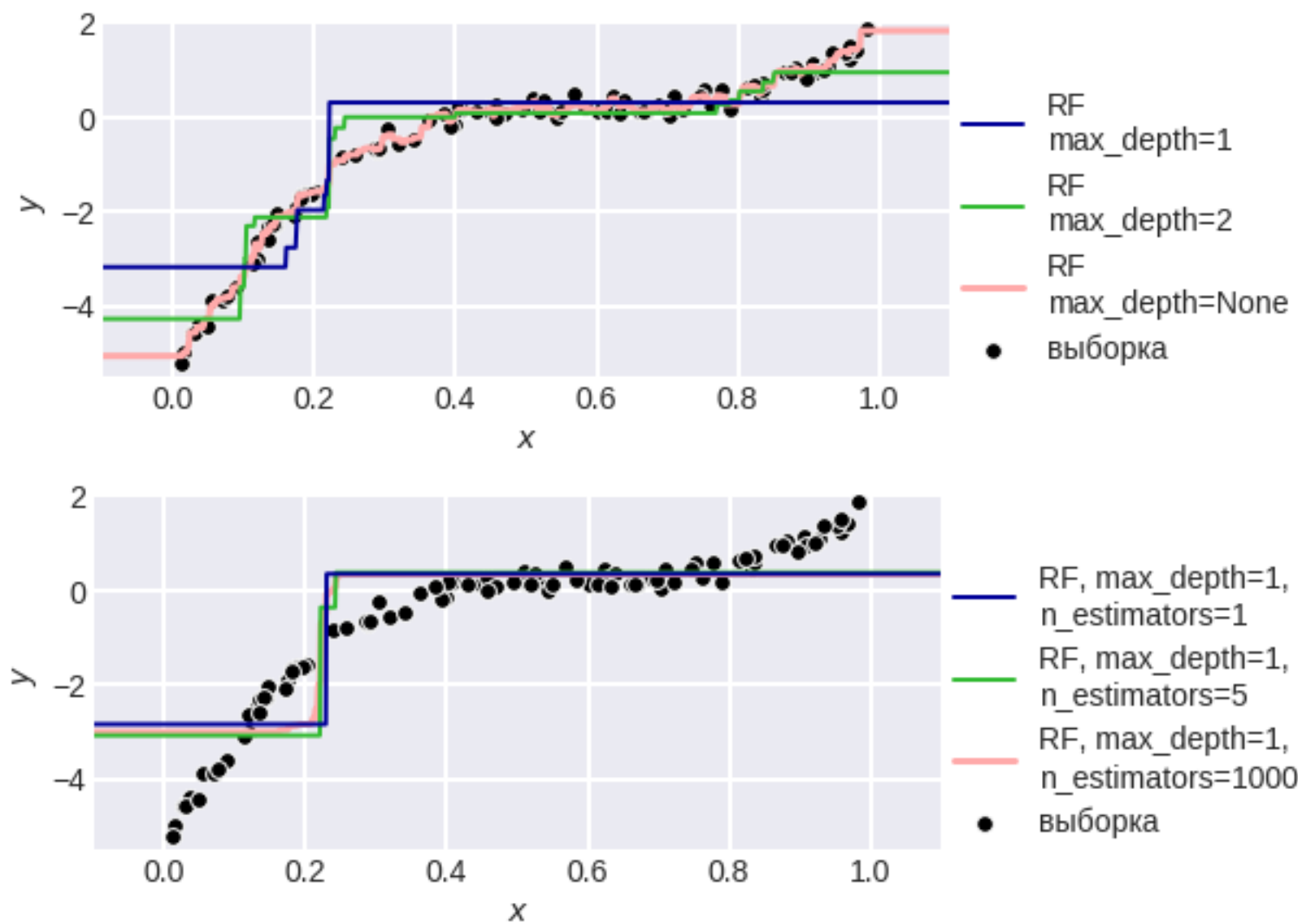
Algorithm 1 *Synthetic Random Forests (SRF)*

- 1: Choose a set of candidate nodesize values $\mathcal{N} = \{n_1, n_2, \dots, n_D\}$.
 - 2: Fit a RF with $nodesize = n_j$ for $j = 1, \dots, D$. Use the same $ntree$ and $mtry$ value for each forest. Denote the resulting forests by RF_1, \dots, RF_D .
 - 3: Calculate the predicted value for each random forest RF_j , $j = 1, \dots, D$. We call the predicted value the synthetic feature.
 - 4: Fit a RF using for features both the newly created synthetic features and the original p features (using the same $ntree$ and $mtry$ value as before). We call this the synthetic RF.
-

чтобы не переобучаться используется OOB-прогноз

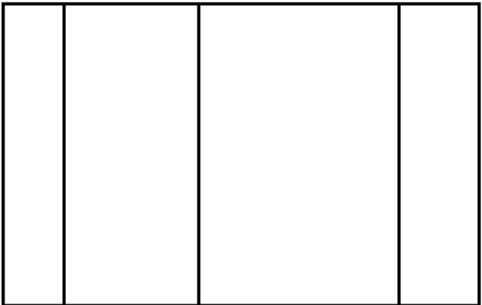
Ishwaran H, Malley JD. «Synthetic learning machines». BioData Min. 2014;7(1):28 // https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4279689/pdf/13040_2014_Article_28.pdf

Когда плохи методы, основанные на деревьях...

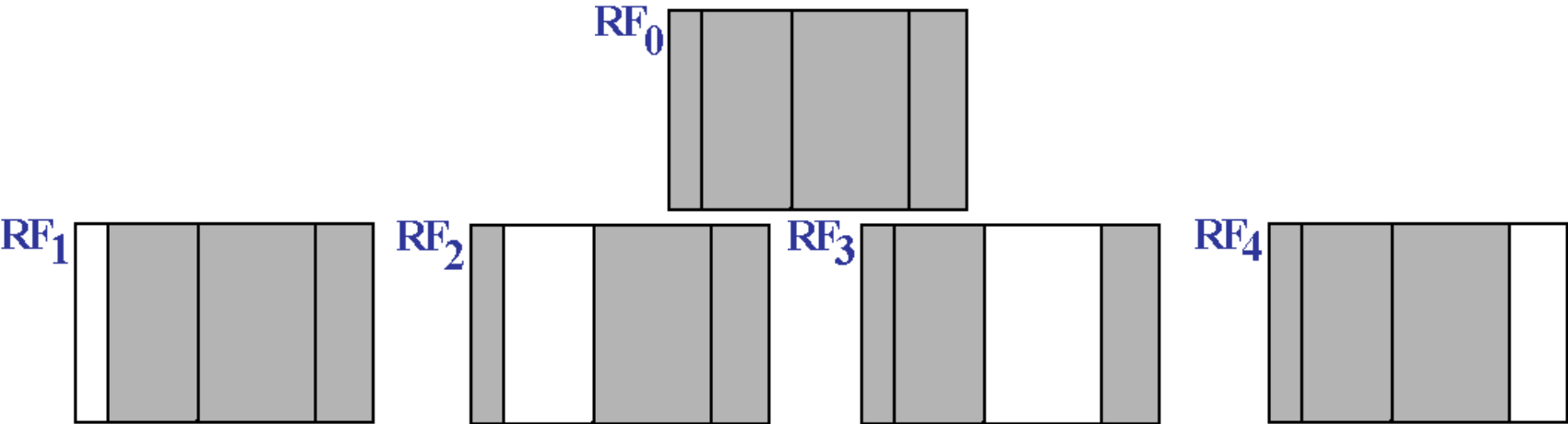


Приложения RF: Biological Response

1. Генерация групп признаков



2. Обучение RF на всех группах и на данных без некоторых групп



3. Комбинация полученных алгоритмов

Одна из технологий решения задач!

Приложения RF: Реальная задача (Photo)

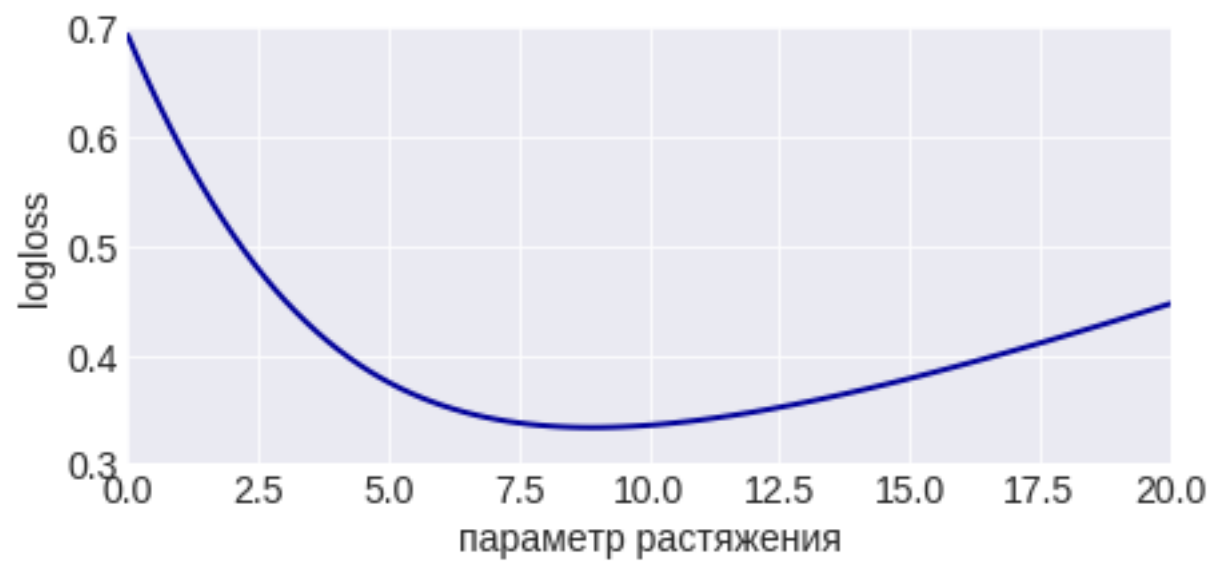
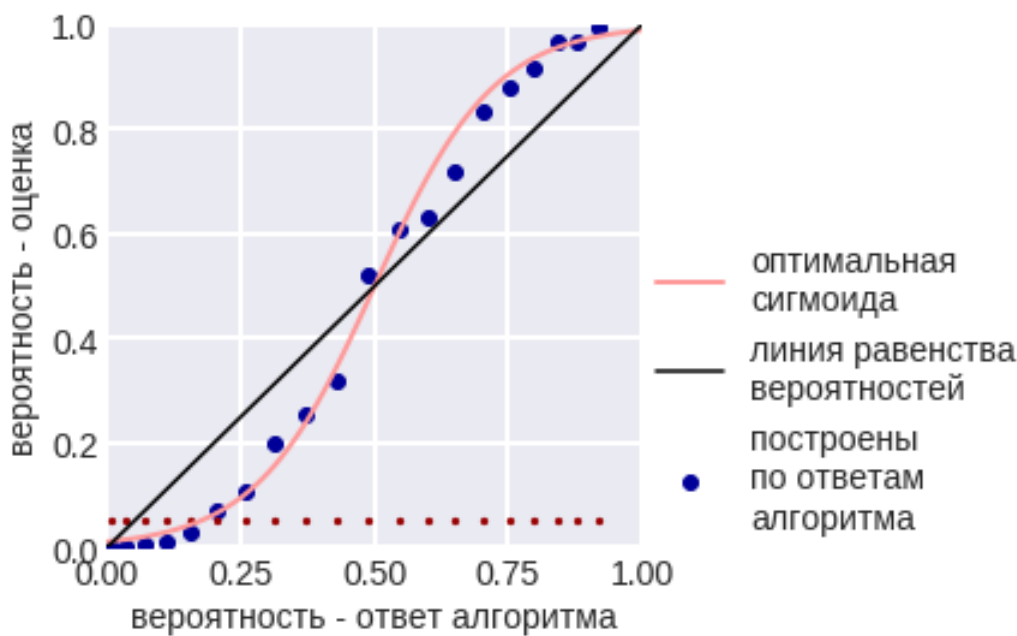
поиск решения в таком виде

$$\sqrt{c_1 \cdot (\text{rf}_1)^2 + \dots + c_{24} \cdot (\text{rf}_{24})^2 + c_{25} (\text{knn})^2},$$

где разные RF настроены на разных признаковых пространствах

Очень хорошо смешивать разнотипные алгоритмы!

Приложения RF: Калибровка RF






Задача Search Results Relevance

row	id	query	product_title	product_description	median relevance	relevance variance
78	230	harleydavidson	Harley-Davidson and Philosophy (Paperback)		4	0.943
120	367	ninja turtle socks	Highpoint Kids' 5PK No Show (Assorted)	He will feel like a superhero in the Teenage Mutant Ninja Turtles 5PK No Show Socks by Highpoint.	4	0.800
607 6	1955 7	long prom dress	Daniella Collection	This one piece dress has mesh and rhinestone detailing. This dress features long sleeves, a round neckline and a flared hem.	3	0.866
153	472	16 gb memory card	Sandisk 16GB non-HS MSD Flash Drive 3in1 - Black (SDSDQR-016G- T46A)	SanDisk Elevate 16GB non-HS MSD 3in, microSD memory card with USB adapter and SD adapter. Includes RescuePro recovery software. Memory Storage Capacity: 16GB Wired Connectivity: Micro SD Slot Features: Plug and Play Includes: MicroSD Adapter, Adapter, USB Adapter Battery no battery used	4	0.000








Похожа на классическую задачу поиска

Задача Search Results Relevance

Completed • \$20,000 • 1,326 teams

 **Search Results Relevance**

Mon 11 May 2015 – Mon 6 Jul 2015 (2 months ago)

#	Team Name <small>‡ model uploaded * in the money</small>	Score 	Entries	Last Submission UTC (Best – Last Submission)
1	Chenglong Chen ‡ *	0.72189	160	Mon, 06 Jul 2015 09:53:22
2	Mikhail & Stanislav & Dmitry  ‡ *	0.71871	83	Mon, 06 Jul 2015 22:55:46 (-1.2h)
3	Quartet  ‡ *	0.71861	279	Mon, 06 Jul 2015 17:24:26 (-3.3d)
4	Shize & Shail & Phil 	0.71802	252	Mon, 06 Jul 2015 23:44:14
5	I love Phở Bò	0.71700	48	Mon, 06 Jul 2015 08:48:29 (-10.5h)
6	Gzs_iceberg	0.71681	122	Mon, 06 Jul 2015 14:27:09 (-9.9d)
7	YDM 	0.71374	283	Mon, 06 Jul 2015 16:31:54 (-1.7h)
8	A & A & G 	0.71297	229	Mon, 06 Jul 2015 19:13:28 (-25.6h)
9	ë 	0.71265	96	Sun, 05 Jul 2015 21:50:24 (-5.9d)
10	Alexander D'yakonov (PZAD, Russia)	0.71262	93	Mon, 06 Jul 2015 19:40:28 (-33d)
11	SearchSearchSearch	0.71022	58	Mon, 06 Jul 2015 01:06:18 (-3.9h)
12	woshialex	0.70889	52	Thu, 02 Jul 2015 00:21:23 (-10.1h)
13	Alexander Ryzhkov (PZAD, Russia)	0.70777	64	Mon, 06 Jul 2015 22:57:31

Задача

Дан **запрос**: «16 gb memory card»
Ему соответствует **выдача**,
элемент выдачи – (название товара, описание)

Sandisk 16GB non-HS MSD Flash Drive 3in1 - Black (SDSDQR-016G-T46A)	SanDisk Elevate 16GB non-HS MSD 3in, microSD memory card with USB adapter and SD adapter. Includes RescuePro recovery software. Memory Storage Capacity: 16GB Wired Connectivity: Micro SD Slot Features: Plug and Play Includes: MicroSD Adapter, Adapter, USB Adapter Battery no battery used
---------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Паре (запрос, выдача) соответствует **релевантность** – 1, 2, 3 или 4.

Дана ещё дисперсия релевантностей (т.к. несколько **ассесоров** оценивало выдачу),
но не была использована

Запомним:

релевантность в обучении – **медиана** релевантностей, которые проставили ассесоры

Задача

Почему не классическая задача поиска?

обучение		тест	
запрос 1	выдача 1	запрос 1	выдача 4
запрос 1	выдача 2	запрос 1	выдача 5
запрос 1	выдача 3	запрос 2	выдача 4
запрос 2	выдача 1	запрос 2	выдача 5
запрос 2	выдача 2	запрос 2	выдача 6
запрос 2	выдача 3		

Запросы в тесте такие же как и в обучении!

Как это использовать?

Задача

обучение

запрос 1	выдача 1
запрос 1	выдача 2
запрос 1	выдача 3
запрос 2	выдача 1
запрос 2	выдача 2
запрос 2	выдача 3

тест

запрос 1	выдача 4
----------	----------

у нас:
схожесть выдачи и
выдачи с определённой
оценкой

классика: схожесть выдачи и запроса

**Хорошая выдача (для определённого запроса)
должна быть похожа на хорошие выдачи (этого запроса)!**

Функционал качества: Quadratic Weighted Kappa

показывает согласованность порядков,
когда ответы "мера релевантности"

y = 1 1 1 2 2 3 3 3 # правильный ответ a = 1 1 2 1 3 2 3 3 # наш ответ	0.6666667
a = 1 1 1 2 2 3 3 3 # наш ответ	1
a = 3 3 3 2 2 1 1 1 # наш ответ	-1

будет потом

Метод решения

1. Предобработка данных

2. Генерация признаков

3. Выбор модели / настройка

4. Ансамбли

5. Деформация ответов / решающее правило

Построение очень простой модели!

<p>Sandisk 16GB non-HS MSD Flash Drive 3in1 - Black (SDSDQR-016G-T46A)</p>	<p>Удаляем html-теги, схлопываем текст, удаляем спецсимволы</p> <p>sandisk16gbnonhsm sdflashdrive3in1 blacksdsql016gt46a</p>
<p>Делаем 3-граммы</p> <p>san, and, ndi, dis, isk, sk1, k16, 16g, 6gb, gbn ...</p> <p>Кстати, для правильного слова "scandisk": sca, can, and, ndi, dis, isk</p>	<p>Переходим к модели "мешок слов" + tf-idf</p> <div><div>san</div><div>16g</div><div>1gb</div><div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div></div>

3-граммы могут победить опечатки.

Другие подходы (2е место)

```
hardisk hard drive
extenal external
soda stream sodastream
fragrance fragrance
16 gb 16gb
32 gb 32gb
500 gb 500gb
2 tb 2tb
shoppe shop
refrigirator refrigerator
assassinss assassins
harleydavidson harley davidson
harley-davidson harley davidson
```

На практике так тоже делают – много ручной разметки.

Здесь для простоты модели не стали

Заметим, что модель становится переобученной.

Для новых запросов такого ручного устранения неоднозначностей нет!

Другие подходы (1е место)

Здесь создан словарь синонимов

child, kid kid
bicycle, bike bike
refrigerator, fridge, freezer fridge
fragrance, perfume, cologne, eau de toilette perfume

Все делали стемминг (мы нет)

Признаки

Насколько похожи тексты?

Точнее: множества

san, and, ndi, dis, isk, sk1, k16, 16g, 6gb, gbn ...
sca, can, and, ndi, dis, isk

**cos-мера,
мощность пересечения (нормированная)**

Признаки

$\cos(\text{описание}, \text{запрос})$

**$\cos(\text{описание}, \text{запрос})$ нормируем на
 $\max(\text{описание}, \text{запрос})$ по всем описаниям этого запроса**

$\text{mean}(\cos(\text{описание}, \text{описание всех товаров этого запроса в обучении: оценка} = 4))$

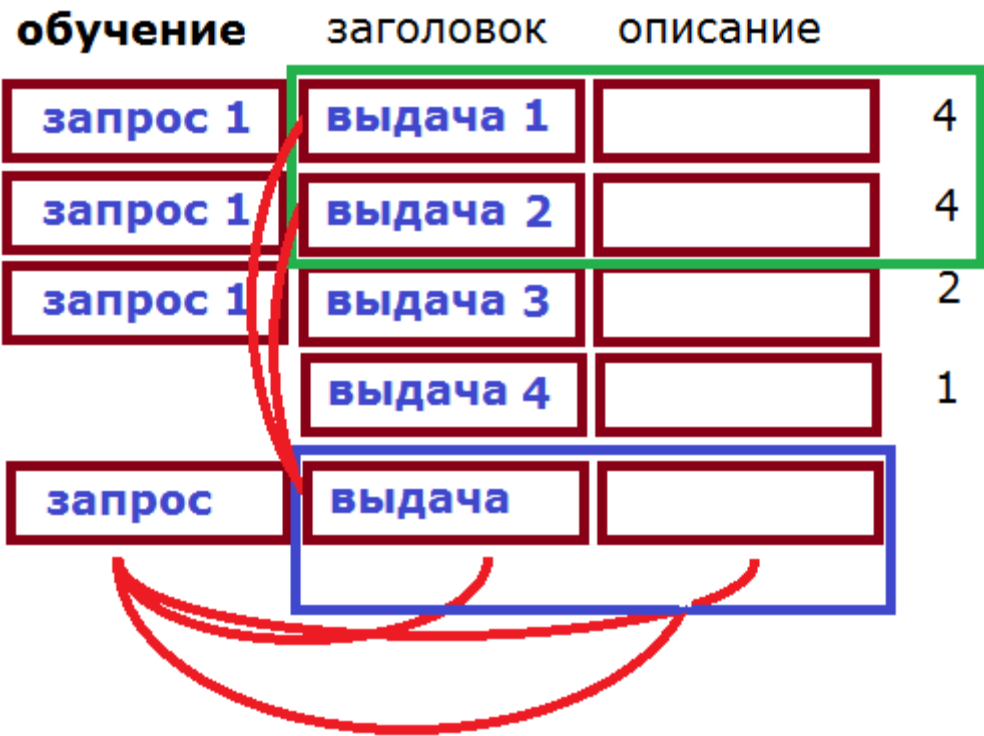
оценка товара: на нём $\max \cos(\text{описание}, \text{его описание})$

описание =

- **само описание**
 - **заголовок**
- **заголовок + описание**

Признаки

– это взаимодействия блоков данных!



Выбор модели

Изначально проблема как настраиваться:

1. Задача классификации с классами [1, 2, 3, 4]

2. Задача регрессии с метками [1, 2, 3, 4]

3. Задача регрессии с метками [1, 4, 9, 16]

4. Задачи классификации/регрессии с метками

[0, 1, 1, 1]

[0, 0, 1, 1]

[0, 0, 0, 1]

Выбор модели

Изначально проблема как настраиваться:

1. Задача классификации с классами [1, 2, 3, 4]

плохо – нет учёта порядка

2. Задача регрессии с метками [1, 2, 3, 4]

хорошо

3. Задача регрессии с метками [1, 4, 9, 16]

нужны эксперименты

4. Задачи классификации/регрессии с метками

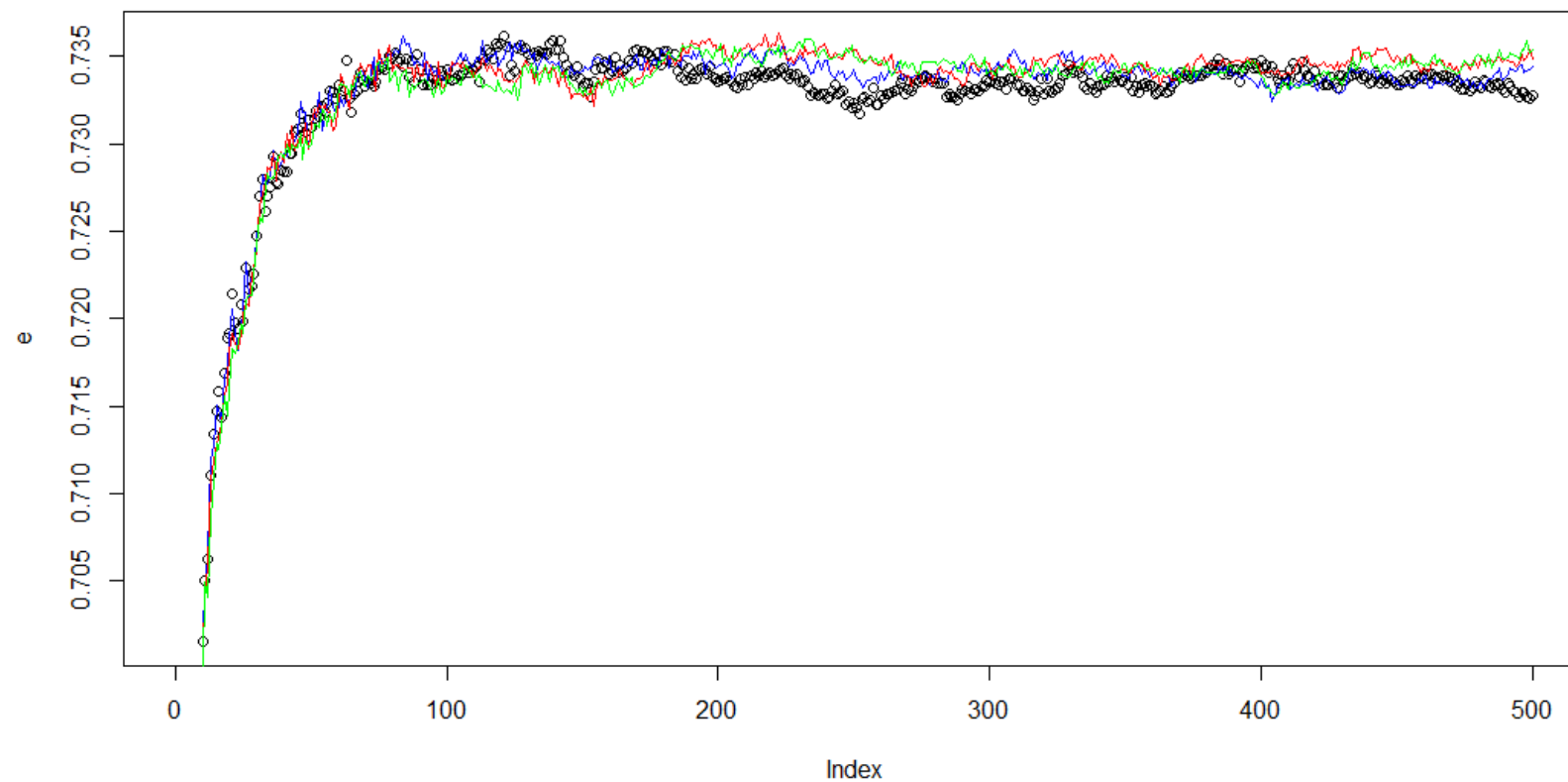
[0, 1, 1, 1]

[0, 0, 1, 1]

[0, 0, 0, 1]

Как потом использовать ответы?

Настройка модели

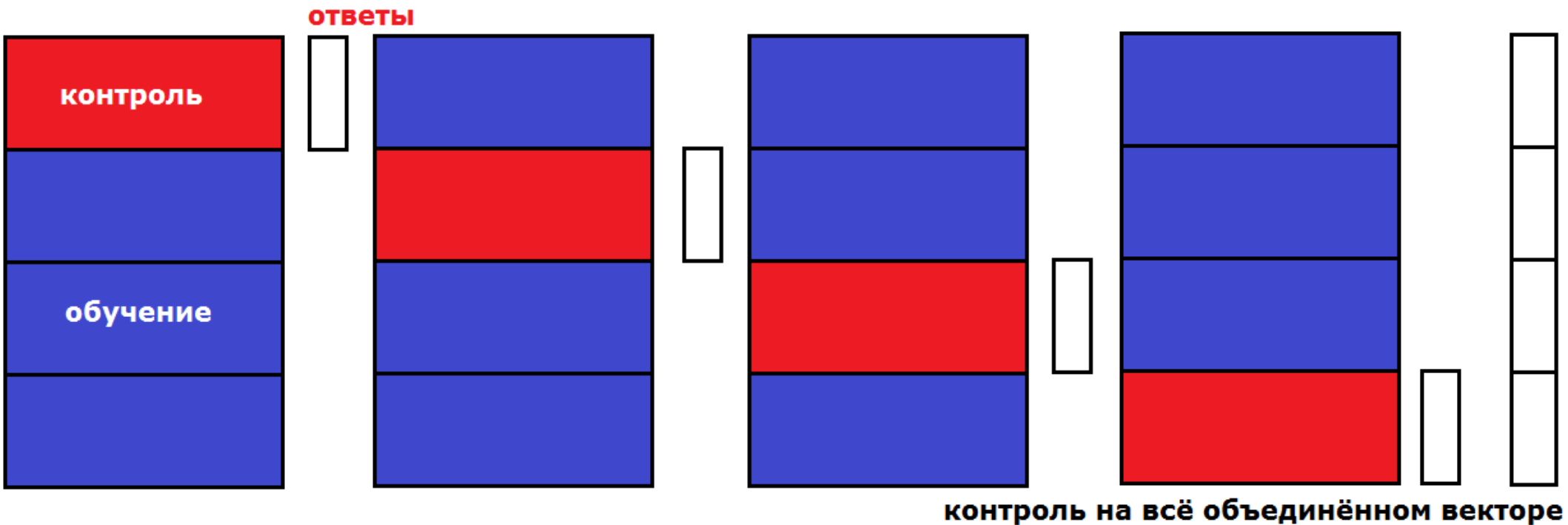


```
model <- randomForest(M1, mr, mtry=40, ntree=200, nodesize=10)
a <- predict(model, M2)
b = pmin(pmax(round(1.45*(a-3.309805)+3.309805), 1), 4)
```

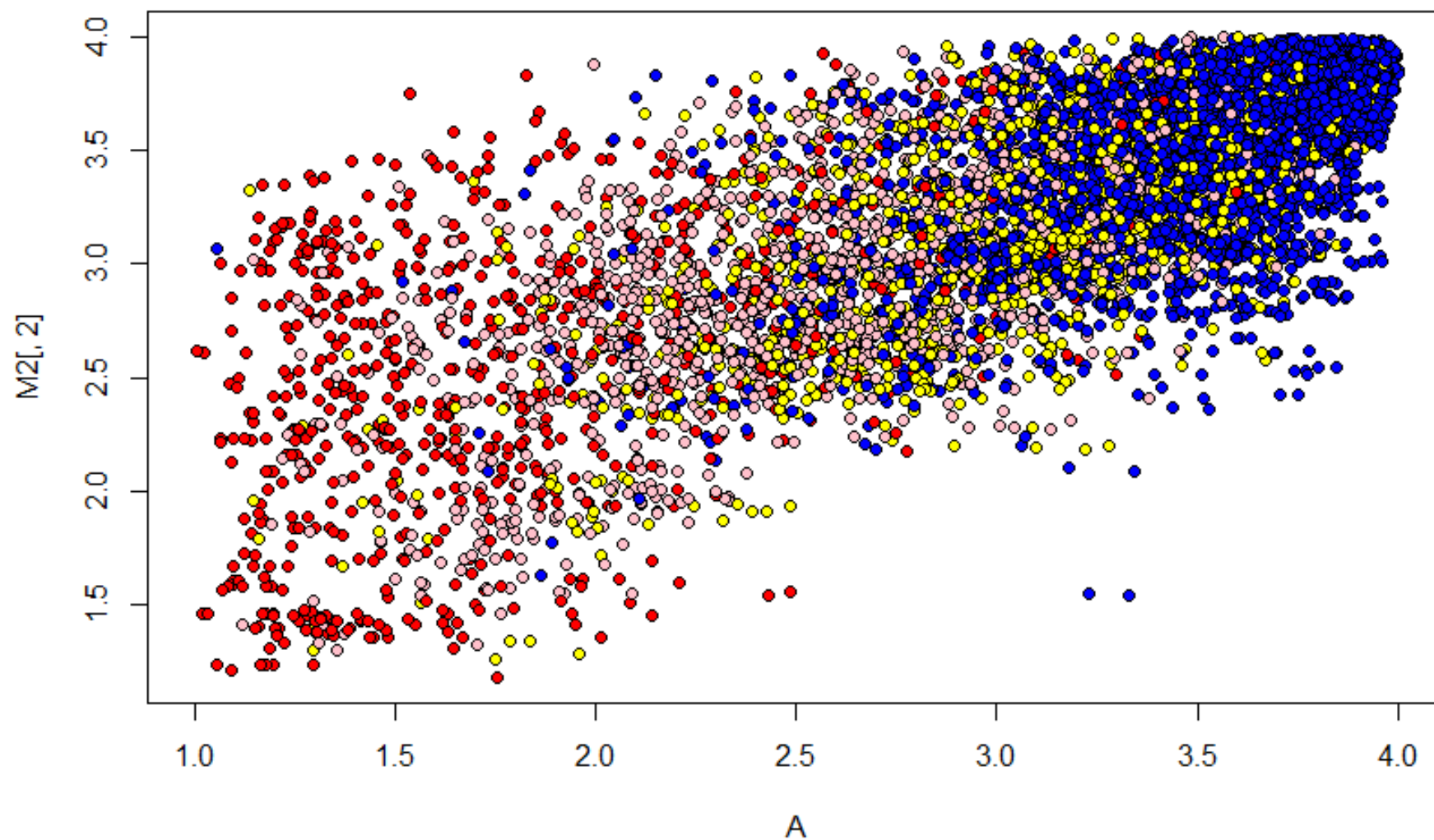
Использовались случайные леса (даже не было зависимости от параметров!)

Отладка

k-fold

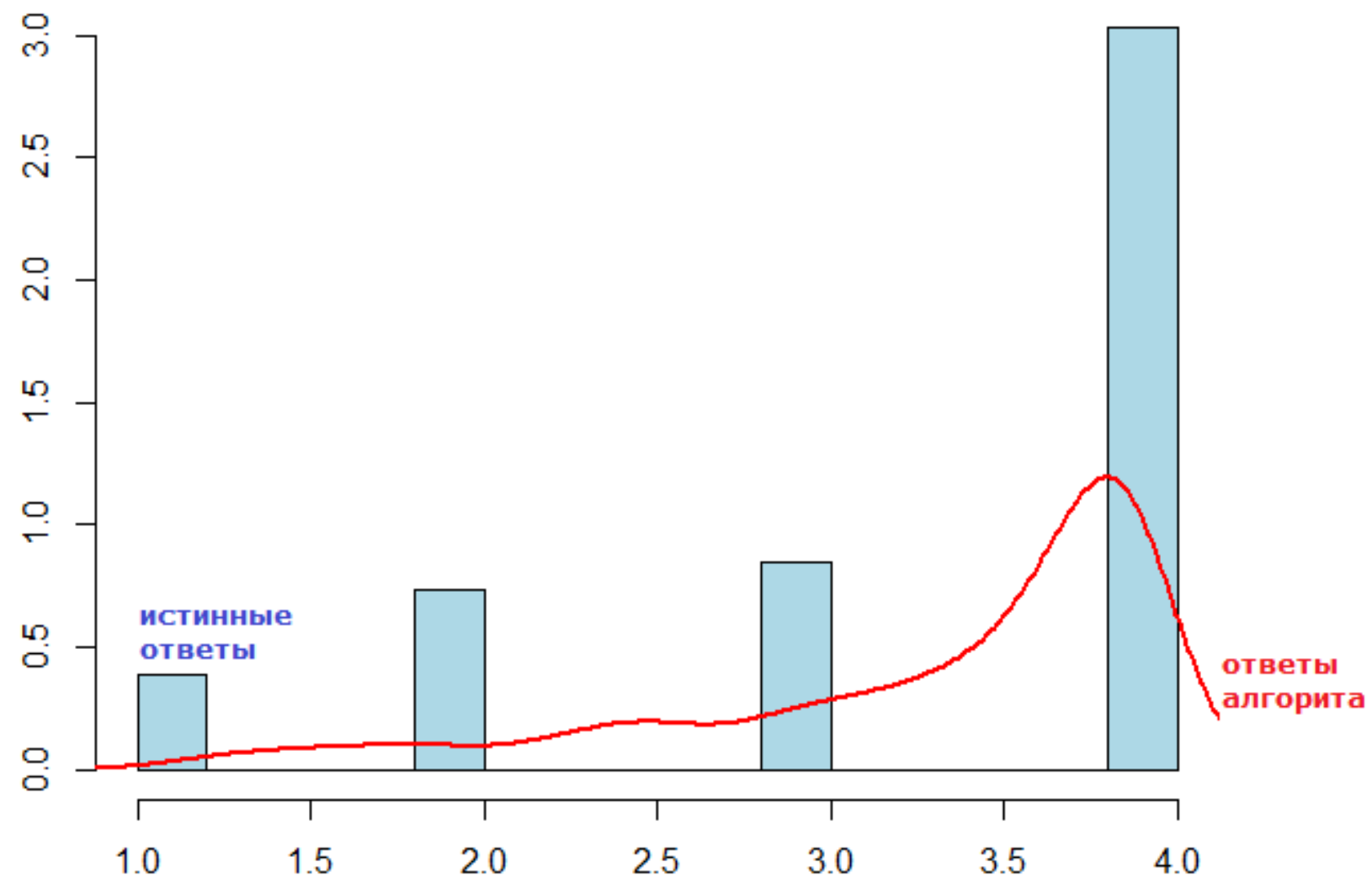


Ответы алгоритма



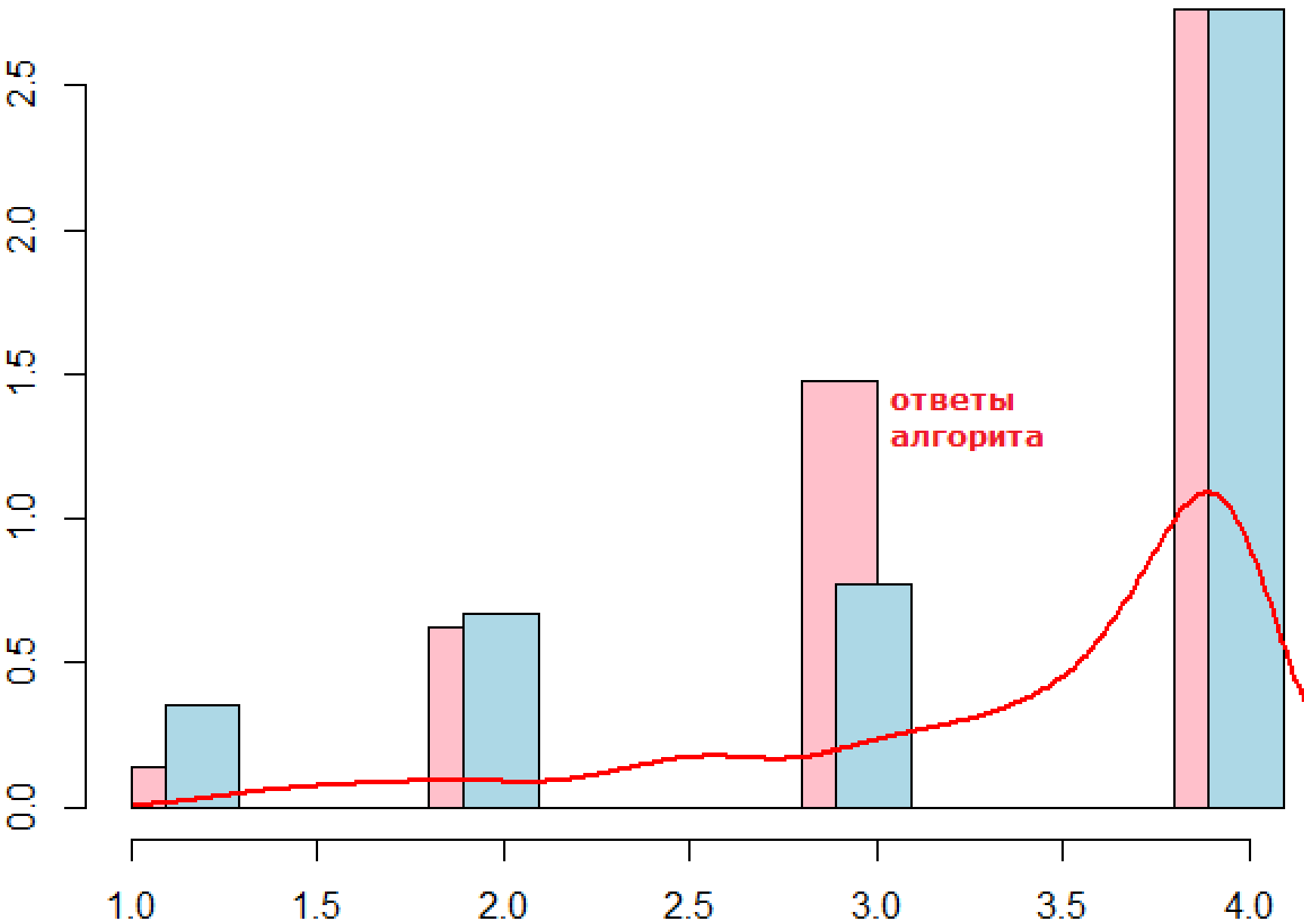
Почему медиана разных алгоритмов лучше усреднения?!

Почему нужны деформации – см. распределения ответов



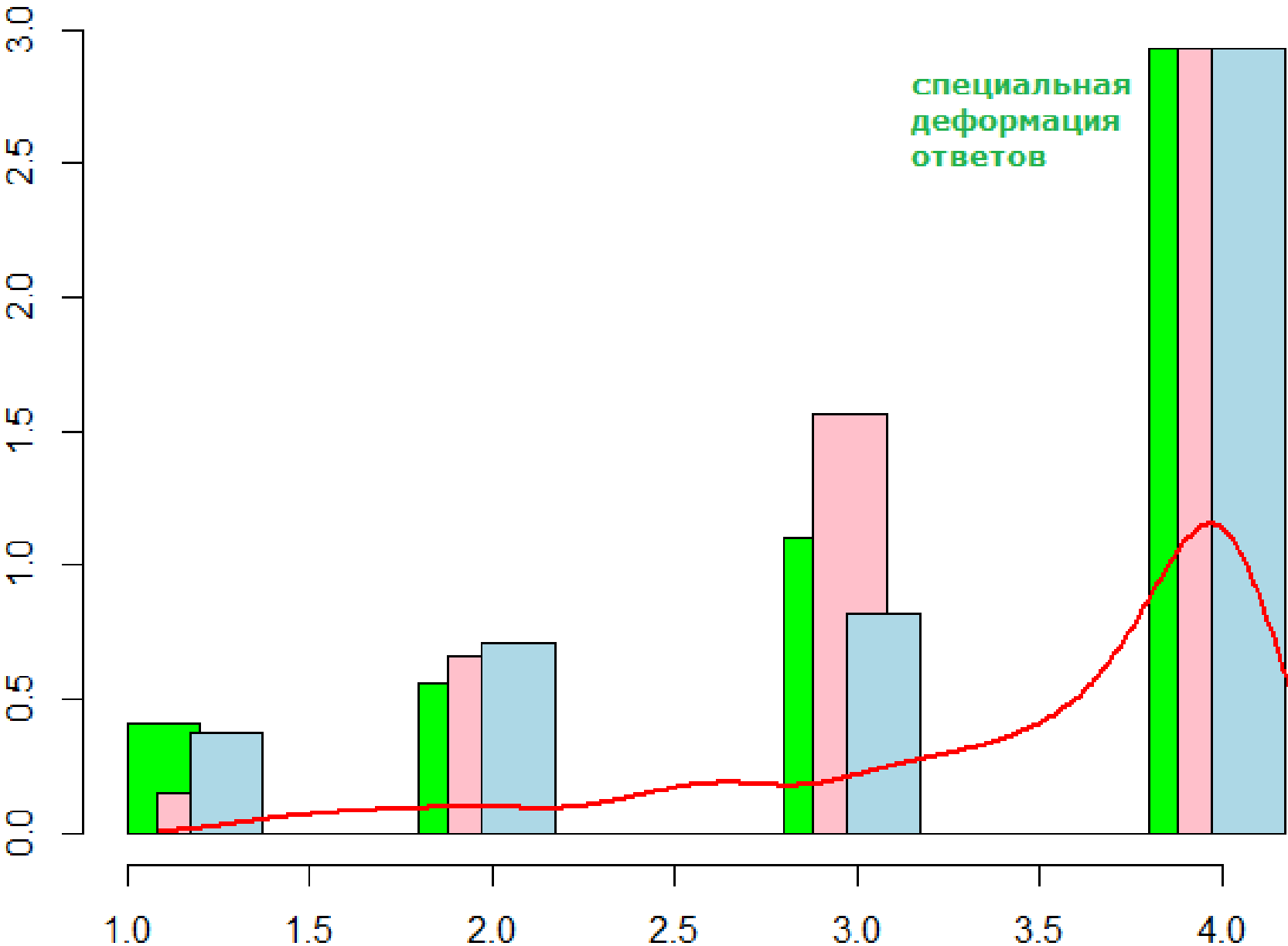
Синий – распределение настоящих ответов
Красный – распределение ответов алгоритма
(они вещественные)

Почему нужны деформации – см. распределения ответов



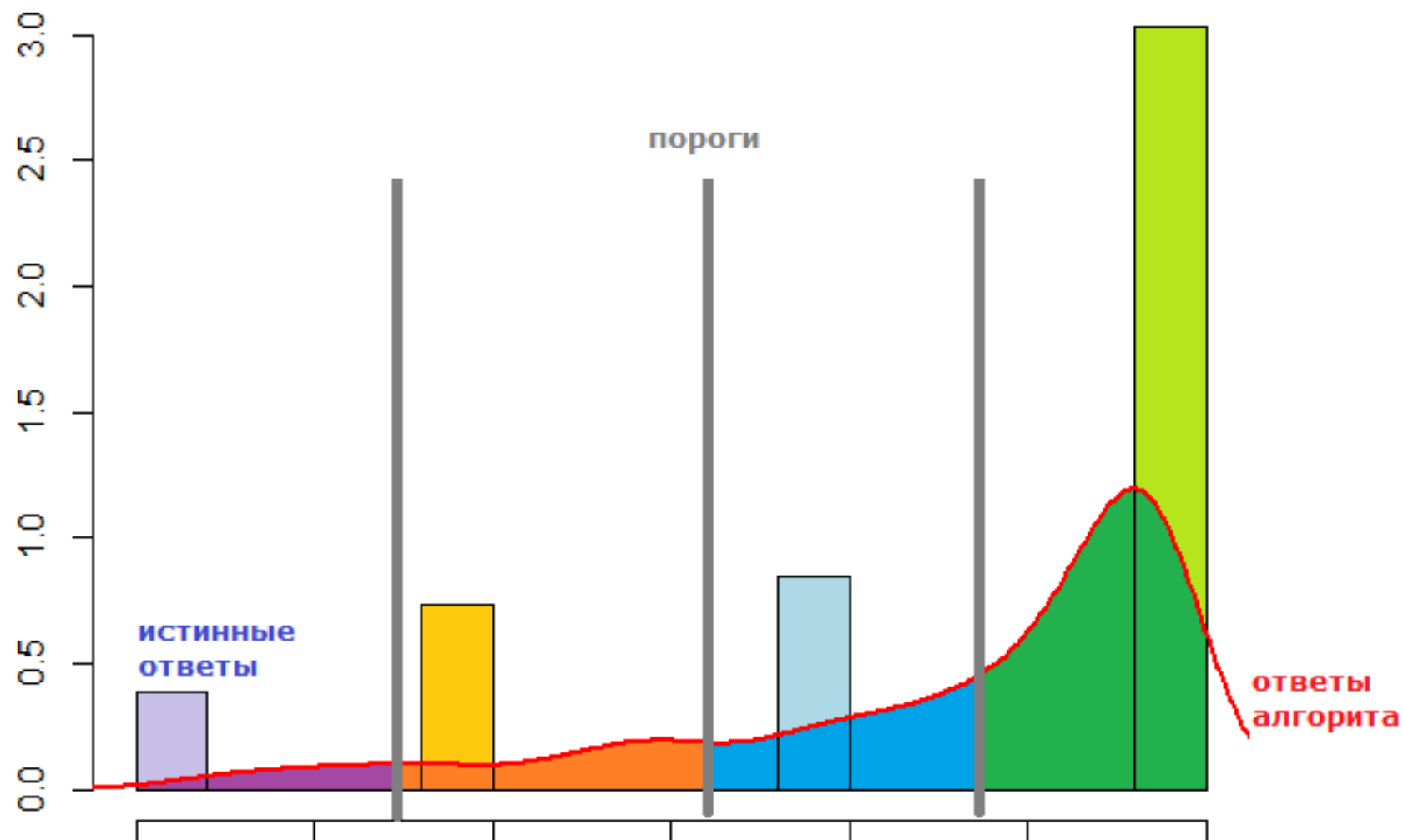
Если просто округлить – перекос в распределениях.

Почему нужны деформации – см. распределения ответов



Если использовать специальную деформацию, то распределения выравниваются

Почему нужны деформации – см. распределения ответов

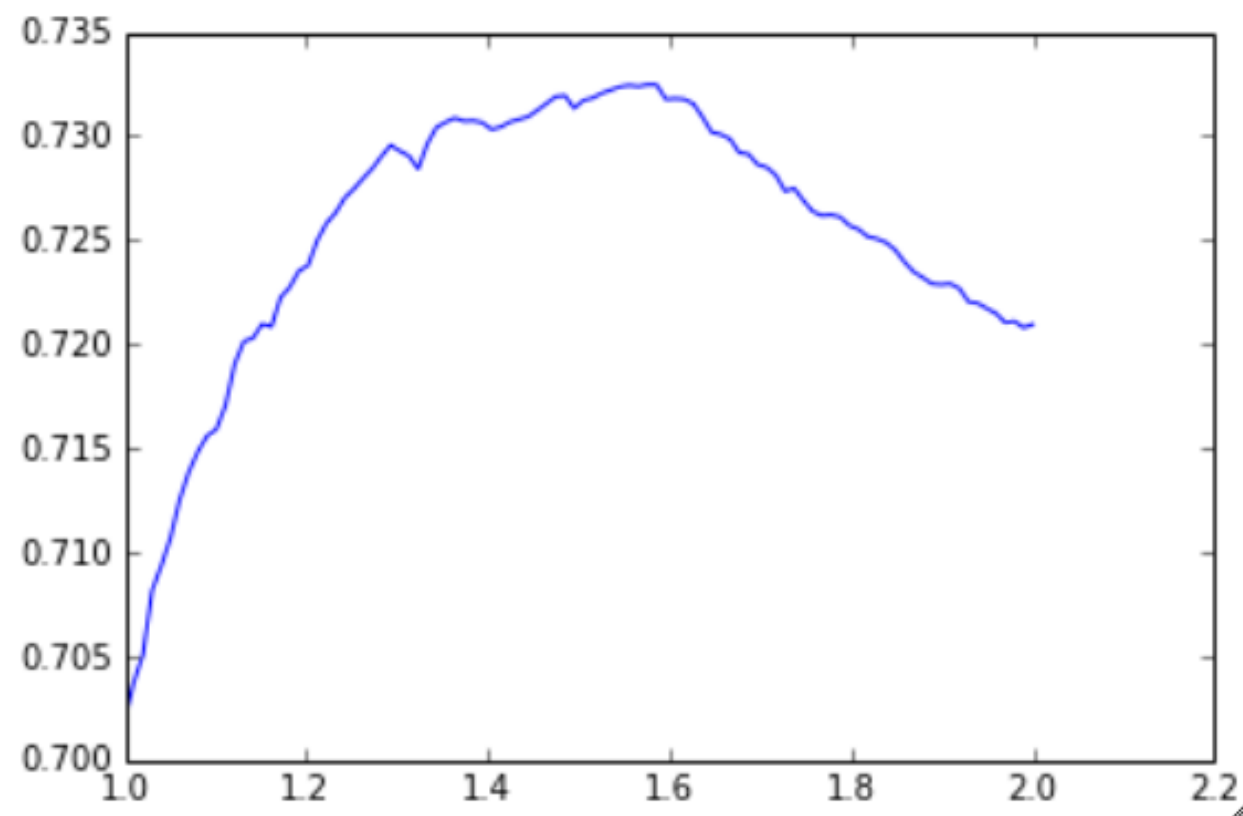


В принципе, можно просто выбирать пороги, чтобы

- **выровнять распределения**
- **повысить качество на контроле**

Какую из двух стратегий выбрать?

Почему нужны деформации – Распределения ответов



Как зависит качество от множителя

$\text{pmin}(\text{pmax}(\text{round}(1.45 * (a - 3.309805) + 3.309805), 1), 4)$

Кстати, 3.309805 – средняя оценка.

Для справки



Хорошая выдача (рел.3-4) похожа на хорошую выдачу в обучении (3-4).

Плохая выдача (1-2) **может быть не похожа на плохую выдачу!**

Для справки

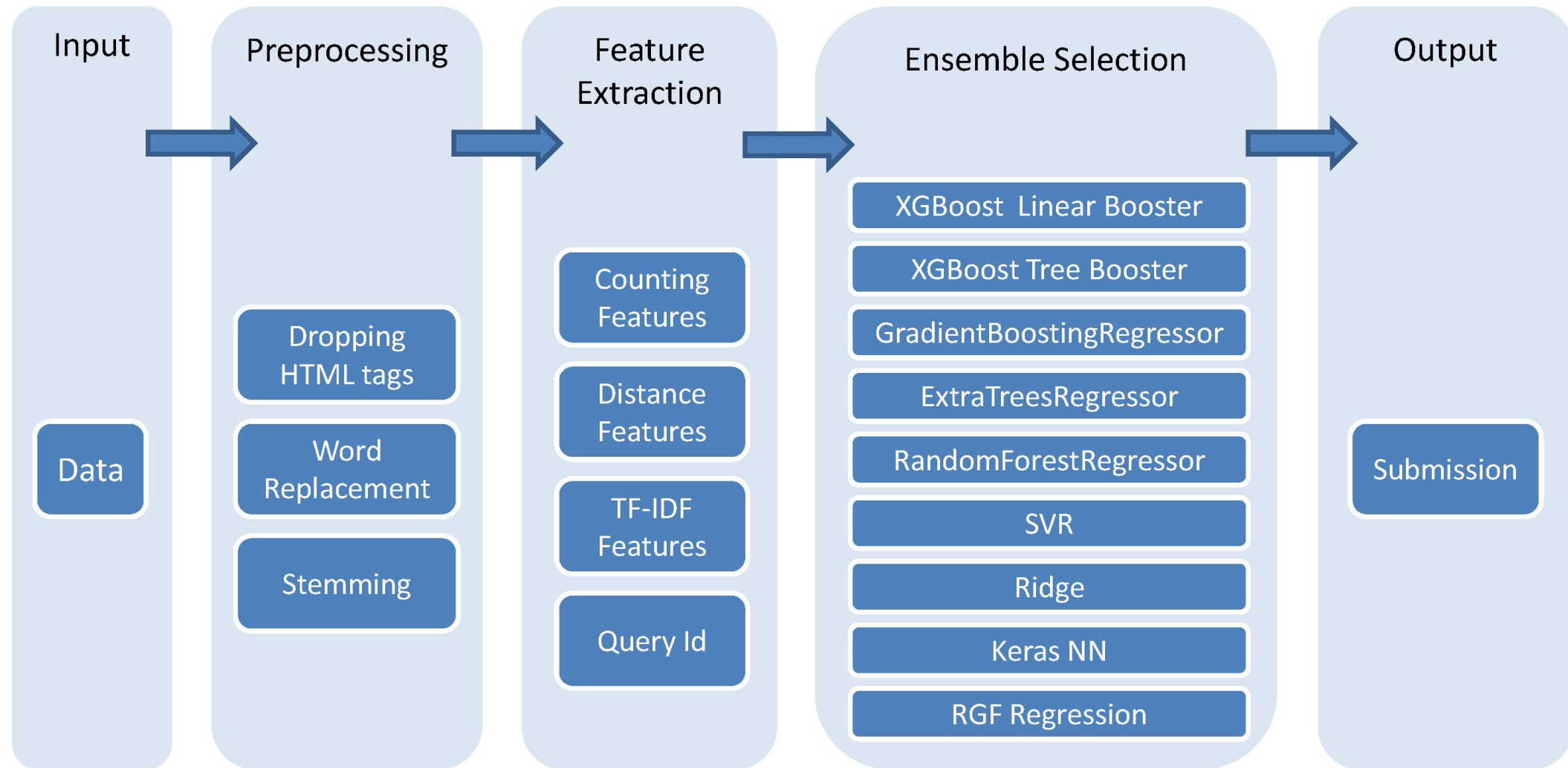
Решающее правило – переводит ответы регрессионных алгоритмов в окончательные.

Обычно

- простое
- тоже требует настройки
- использует постановку задачи
(непересечение классов,
малое число меток у объектов и т.п.)

Это отдельная тема!

Решение победителя (Chenglong Chen)



https://github.com/ChenglongChen/Kaggle_CrowdFlower

Для справки

Полезно посмотреть, на каких объектах модель ошибается
– что можно ещё учесть в признаках.

Итог

Число признаков (max_features) – самый важный параметр (унимодальность)

Число базовых алгоритмов (n_estimators) – чем больше, тем лучше

Глубина (max_depth) – скорее всего, максимальная

Параметры сложности (min_samples_leaf, min_samples_split) – чуть подкорректировать (не очень важно)

Подвыборка (samsize) – брать всё (часто и выбора нет)

Важность

**много способов ввести важность,
особенно с RF, где есть OOB**

Литература

A. Liaw, M. Wiener Classification and Regression by randomForest // R News (2002) Vol. 2/3 p. 18.

<http://www.bios.unc.edu/~dzeng/BIOS740/randomforest.pdf>

И. Генрихов О критериях ветвления, используемых при синтезе решающих деревьев // Машинное обучение и анализ данных, 2014, Т.1, №8, С.988-1017

<http://jmla.org/papers/doc/2014/no8/Genrikhov2014Criteria.pdf>