

Машинное обучение

Лекция 4

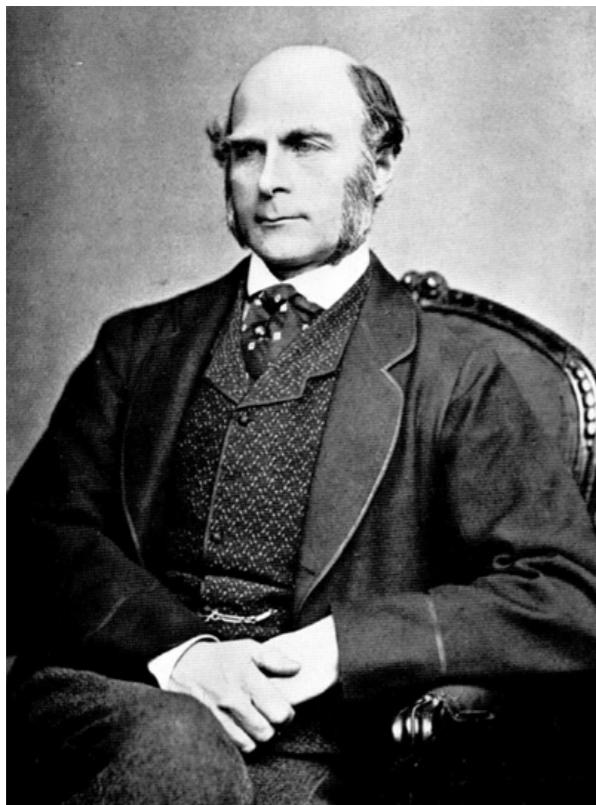
Ансамблирование моделей. Случайный лес.

Власов Кирилл Вячеславович



2020

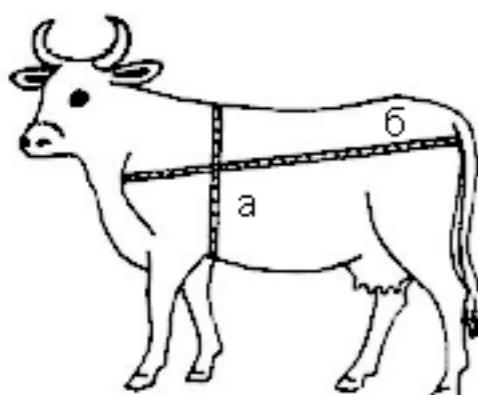
Феномен: «Мудрость толпы»



В 1906 году британский ученый сэр **Фрэнсис Гальтон** посещал выставку достижений животноводства, где случайно провел крайне важное наблюдение.

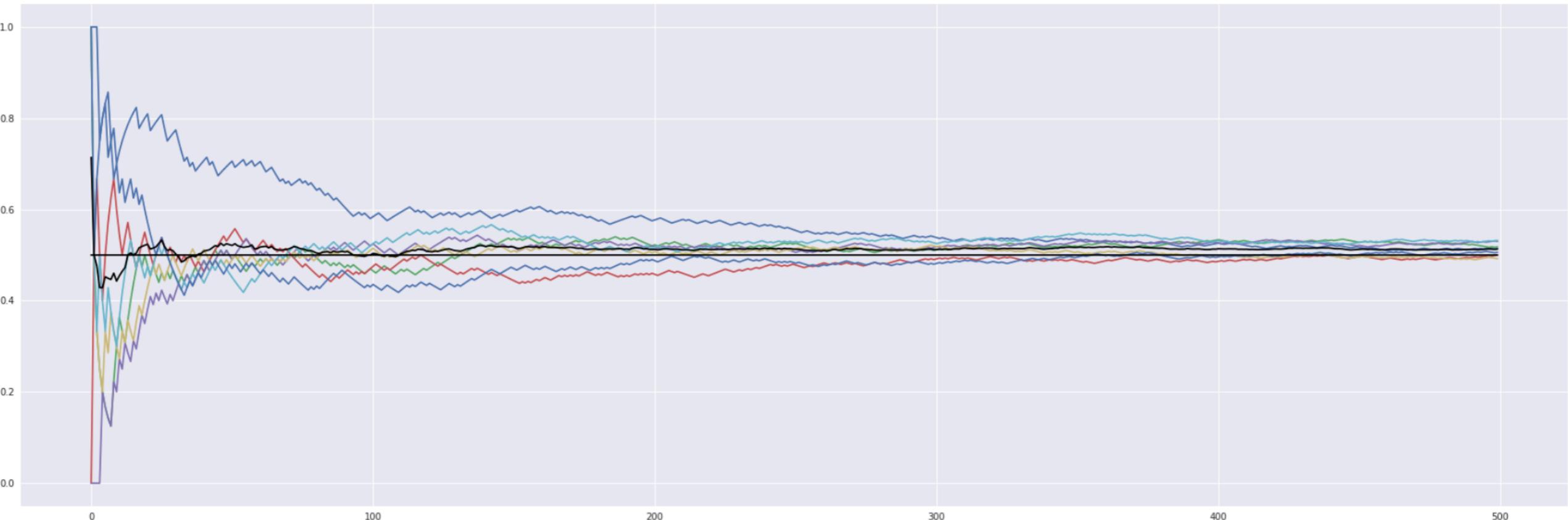
На выставке проводился конкурс, в рамках которого всем желающим предлагалось на глаз угадать точный вес забитого быка. Побеждал тот, кто называл самое близкое к истинному значение.

Полагая, что справиться с подобной задачей под силу только профессионалу, и чтобы доказать некомпетентность толпы, Гальтон посчитал среднее значение из почти восьми сотен догадок посетителей ярмарки. К удивлению ученого, толпа ошиблась меньше, чем на килограмм.



Определение живой массы коровы с помощью обмеров:
а - обхват туловища
б - косая длина

Пример: «Бросание монетки»



Если будем кидать монетку 500 раз, то
соотношение «Орлов» и «Решек» с каждым
броском будем стремиться к 1/2

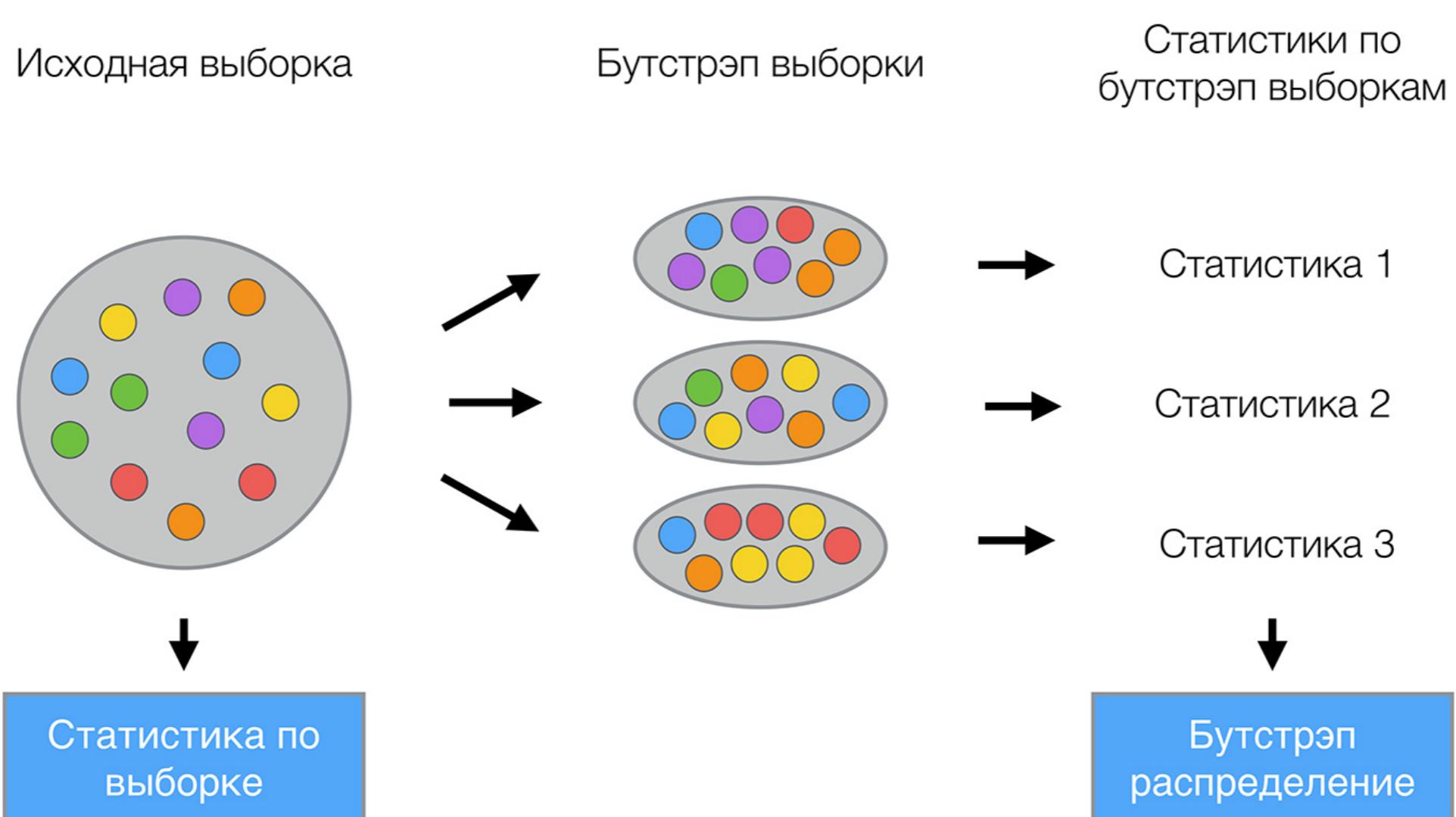
Если повторить эксперимент несколько раз, то
результаты будут отличаться (но все равно
стремиться к 1/2)



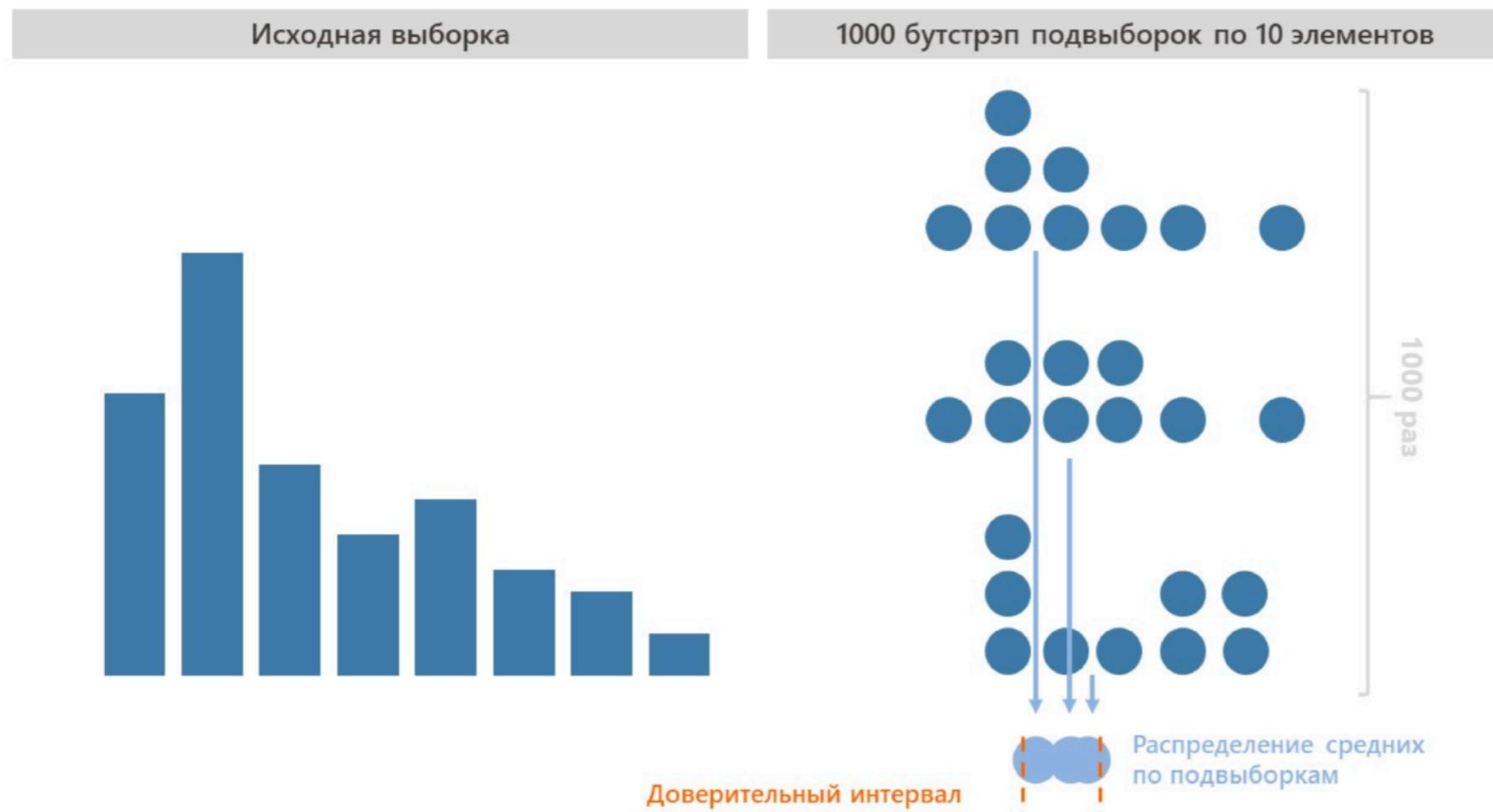
Bootstrap

© Брэдли Эфрон – 1979 год

Bootstrap

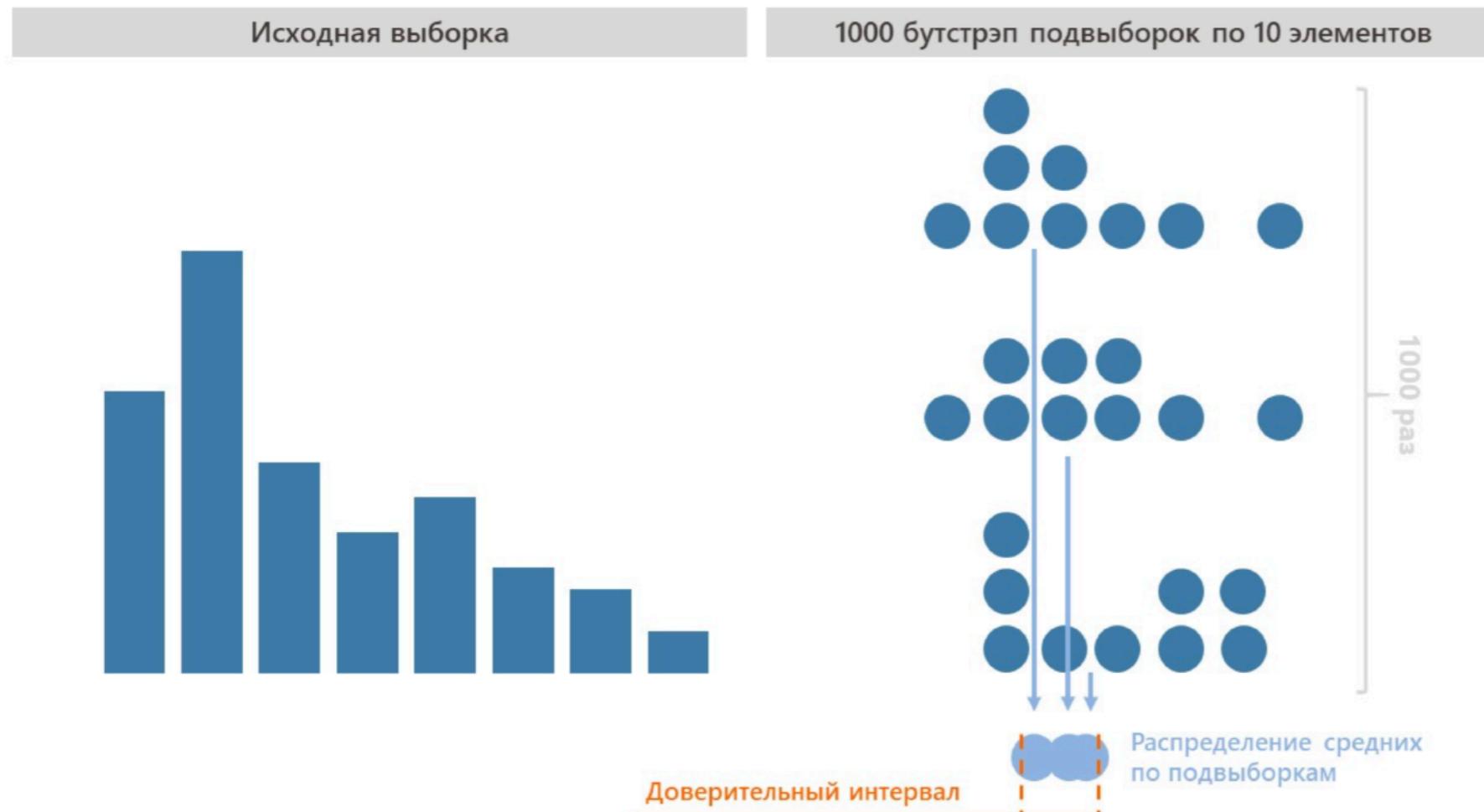


Bootstrap



1. Выберите 2/3 наблюдений из выборки. Отбор осуществляется с возвратом, то есть некоторые наблюдения могут быть выбраны несколько раз, а некоторые могут остаться невыбранными.
2. Вычислите среднее для полученного набора
3. Повторите шаги 500 - 10 000 раз.
4. Отсортируйте тысячу выборочных средних по возрастанию.
5. Найдите средние, которые представляют собой 2.5 и 97.5 процентили. Два выбранных значения и будут границами 95%-го доверительного интервала.

Bootstrap



1. Выберите 2/3 наблюдений из выборки. Отбор осуществляется с возвратом, то есть некоторые наблюдения могут быть выбраны несколько раз, а некоторые могут остаться невыбранными.
2. Вычислите среднее для полученного набора
3. Повторите шаги 500 - 10 000 раз.
4. Отсортируйте тысячу выборочных средних по возрастанию.
5. Найдите средние, которые представляют собой 2.5 и 97.5 процентили. Два выбранных значения и будут границами 95%-го доверительного интервала.

Sub-sampling (*pasting*) то есть выборка без повторений - достойная альтернатива

Bootstrap (Пример)

Конверсия

Было:

923 посетителя по рекламе из них у нас что-то купили 28



3.03%

Стало:

893 посетителя по рекламе из них у нас что-то купили 34



3.81%

Аналитическое решение:

Нас интересует разность => вычитаем матожидания и складываем дисперсии

Истинное значение с 95% вероятностью лежит в пределах плюс-минус двух стандартных отклонений от матожидания, то есть между -0.93% и 2.48%.

Bootstrap (Пример)

```
In [19]: def get_bootstrap_samples(data, n_samples):
    # функция для генерации подвыборок с помощью бутстрэпа
    indices = np.random.randint(0, len(data), (n_samples, len(data)))
    samples = data[indices]
    return samples

def stat_intervals(stat, alpha):
    # функция для интервальной оценки
    boundaries = np.percentile(stat, [100 * alpha / 2., 100 * (1 - alpha / 2.)])
    return boundaries

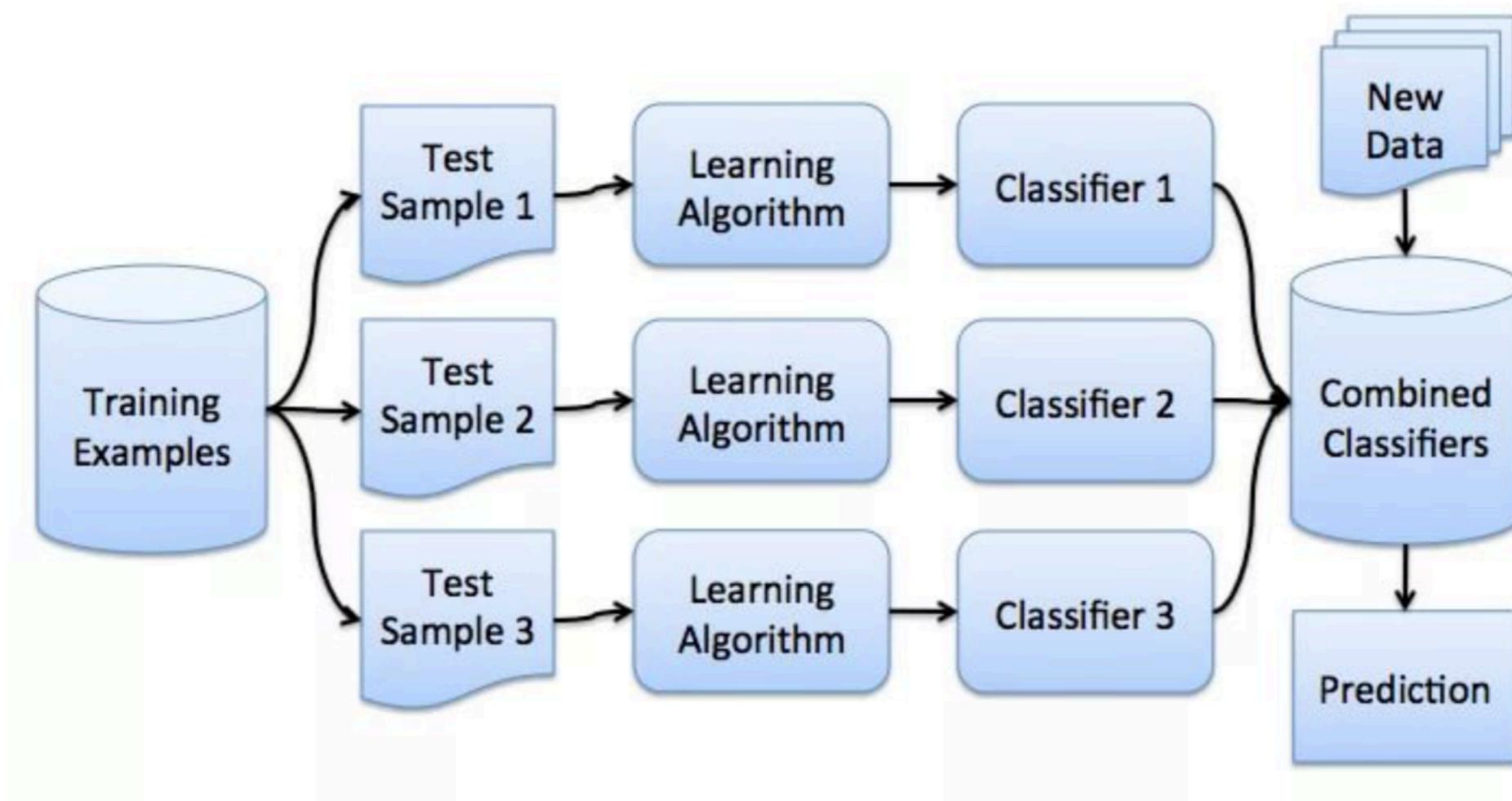
def gen_data(i,q):
    data = np.zeros(i)
    for i in range(q):
        data[i] = 1
    return data

test = gen_data(923, 28)
control = gen_data(893, 34)

mean_scores = get_bootstrap_samples(control, 100000).mean(axis=1) - get_bootstrap_samples(test, 100000).mean(axis=1)
stat_intervals(mean_scores, 0.05)
```

Out[19]: array([-0.00880449, 0.02439068])

Bagging (Bootstrap aggregating)



Bagging (Bootstrap aggregating)



`sklearn.ensemble.BaggingClassifier`

```
(base_estimator=None, n_estimators=10, *,  
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)
```

`sklearn.ensemble.BaggingRegressor`

```
(base_estimator=None, n_estimators=10, *, 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)
```

Out-of-bag error

На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, значит:
Вероятность, что объект попадет в выборку: ?

Out-of-bag error

На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, значит:

Вероятность, что объект попадет в выборку: $\frac{1}{l}$

Вероятность, что объект не попадет в выборку: ?

Out-of-bag error

На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, значит:

Вероятность, что объект попадет в выборку: $\frac{1}{l}$

Вероятность, что объект не попадет в выборку: $1 - \frac{1}{l}$

Так как мы тянем l раз, то вероятность, что объект не попадет во всю выборку: ?

Out-of-bag error

На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, значит:

Вероятность, что объект попадет в выборку: $\frac{1}{l}$

Вероятность, что объект не попадет в выборку: $1 - \frac{1}{l}$

Так как мы тянем l раз, то вероятность, что объект не попадет во всю выборку: $\left(1 - \frac{1}{l}\right)^l$

Значит, при $l \rightarrow \infty$ вероятность, что объект не попадает в выборку: ?

Out-of-bag error

На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, значит:

Вероятность, что объект попадет в выборку: $\frac{1}{l}$

Вероятность, что объект не попадет в выборку: $1 - \frac{1}{l}$

Так как мы тянем l раз, то вероятность, что объект не попадет во всю выборку: $\left(1 - \frac{1}{l}\right)^l$

Значит, при $l \rightarrow \infty$ вероятность, что объект не попадает в выборку: $\frac{1}{e} \approx 0.37$

Out-of-bag error

На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, значит:

Вероятность, что объект попадет в выборку: $\frac{1}{l}$

Вероятность, что объект не попадет в выборку: $1 - \frac{1}{l}$

Так как мы тянем l раз, то вероятность, что объект не попадет во всю выборку: $\left(1 - \frac{1}{l}\right)^l$

Значит, при $l \rightarrow \infty$ вероятность, что объект не попадает в выборку: $\frac{1}{e} \approx 0.37$

Вывод: При формировании Bootstrap выборки в нее попадает только **63%** объектов

Свойство: Можно вычислять Out-of-bag error и не проводить кроссвалидацию

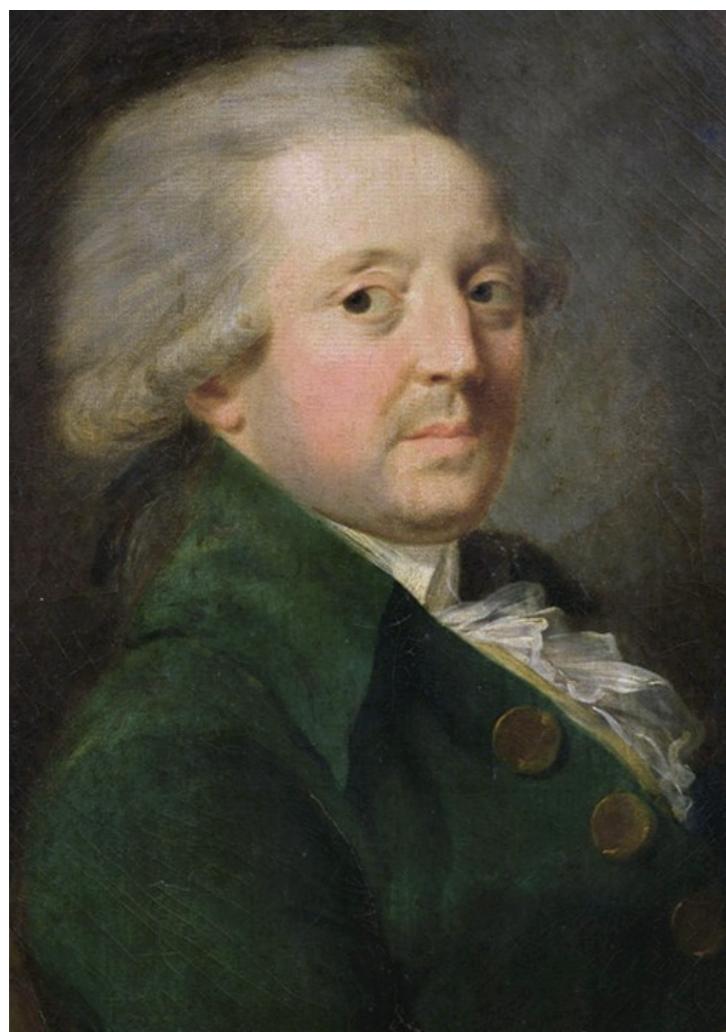
Метод случайных подпространств



Теорема Кондорсе о присяжных

Если каждый член жюри присяжных имеет **независимое мнение**, и если вероятность правильного решения члена жюри больше 0.5, то тогда вероятность правильного решения присяжных в целом возрастает с увеличением количества членов жюри, и стремиться к единице. Если же вероятность быть правым у каждого из членов жюри меньше 0.5, то вероятность принятия правильного решения присяжными в целом монотонно уменьшается и стремится к нулю с увеличением количества присяжных.

Метод случайных подпространств



Теорема Кондорсе о присяжных

Если каждый член жюри присяжных имеет **независимое мнение**, и если вероятность правильного решения члена жюри больше 0.5, то тогда вероятность правильного решения присяжных в целом возрастает с увеличением количества членов жюри, и стремиться к единице. Если же вероятность быть правым у каждого из членов жюри меньше 0.5, то вероятность принятия правильного решения присяжными в целом монотонно уменьшается и стремится к нулю с увеличением количества присяжных.

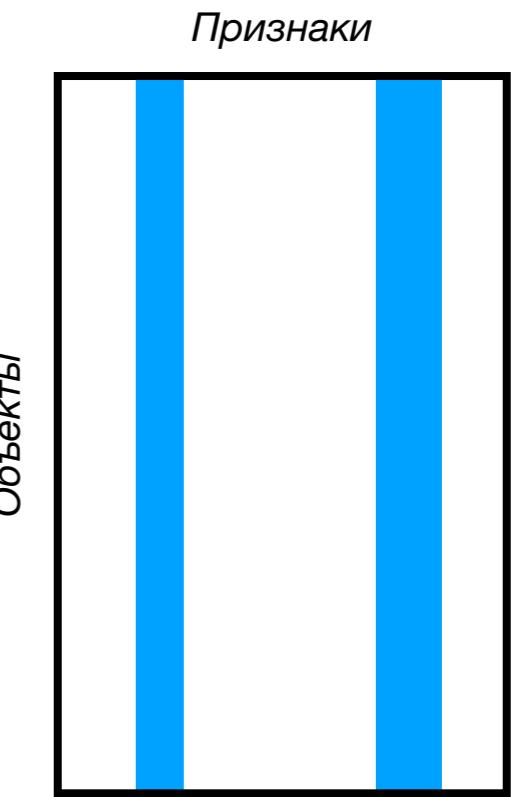
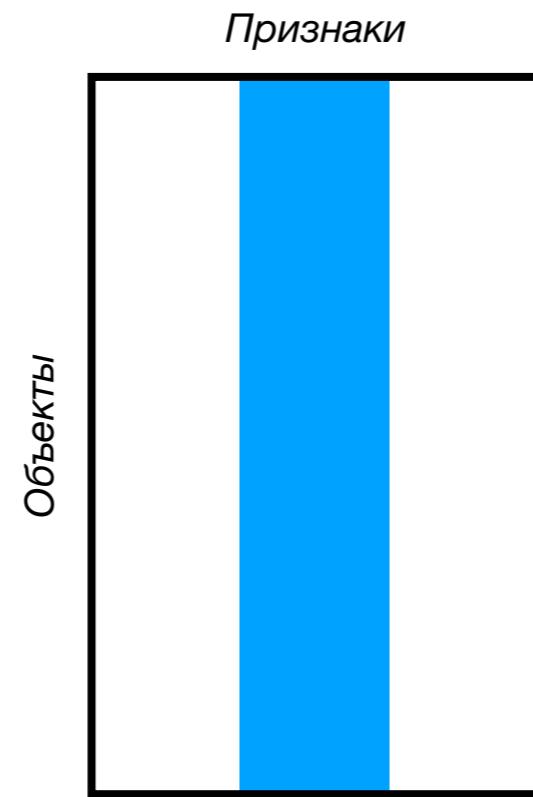
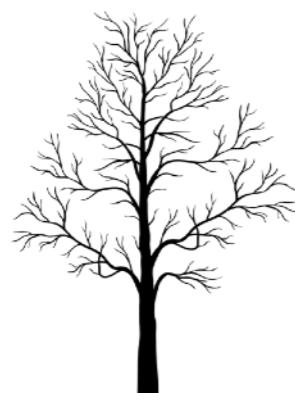
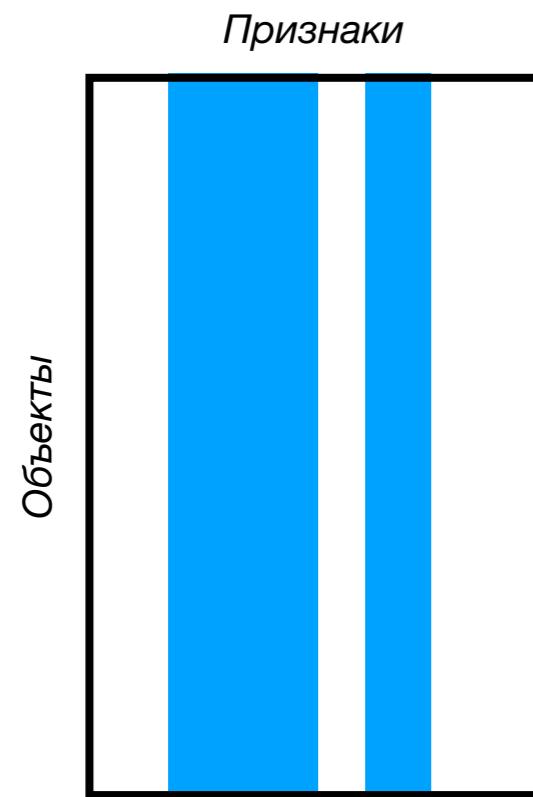
$$\mu = \sum_{i=m}^N C_N^i p^i (1-p)^{N-i}$$

$$p > 0.5 \quad \mu > p$$

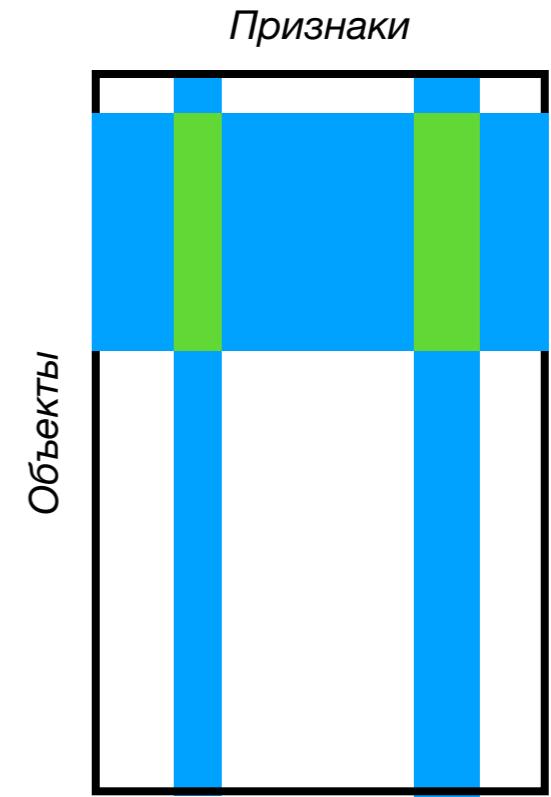
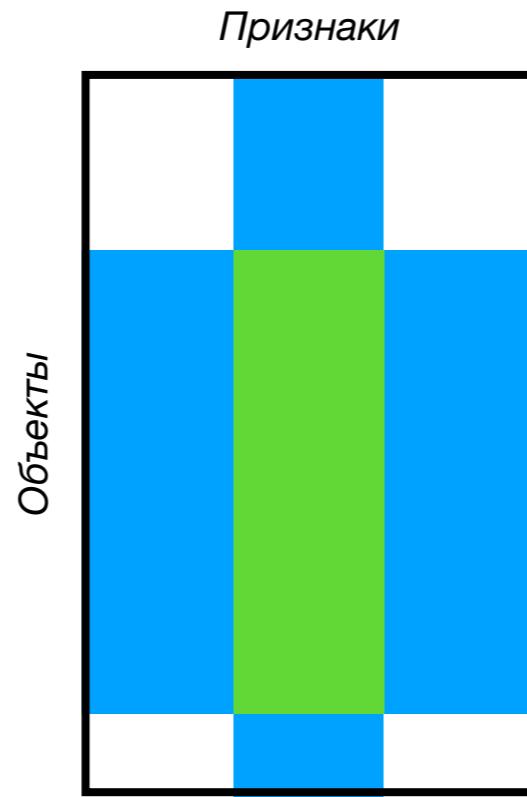
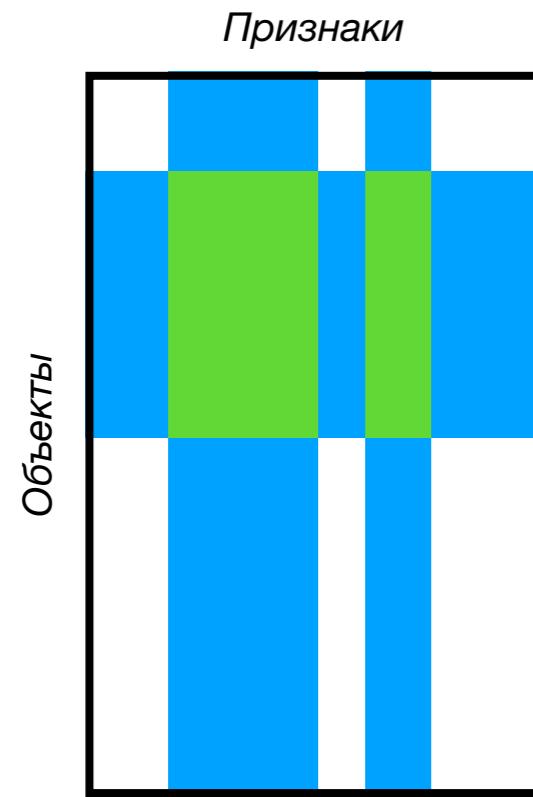
$$N \rightarrow \infty \quad \mu \rightarrow 1$$

Еще больше случайности независимости

Метод случайных подпространств



Случайный лес



Алгоритм:

1. Генерируем bootstrap выборку
2. Строим дерево, такое что
 - **В каждом узле** выбираем m случайных признаков из n возможных и по ним делаем разбиение
 - Дерево строим, до тех пор пока не достигнем определенной глубины дерева или пока в каждом листе больше объектов, чем пороговое
3. Повторяем п. 1 и 2 пока не достигнем заданного количества деревьев
4. Усредняем ответы деревьев

Для классификации:

$$m = \sqrt{n}$$

Пока один объект в листе

Для регрессии:

$$m = \frac{n}{3}$$

Пока пять объектов в листе

Random Forest



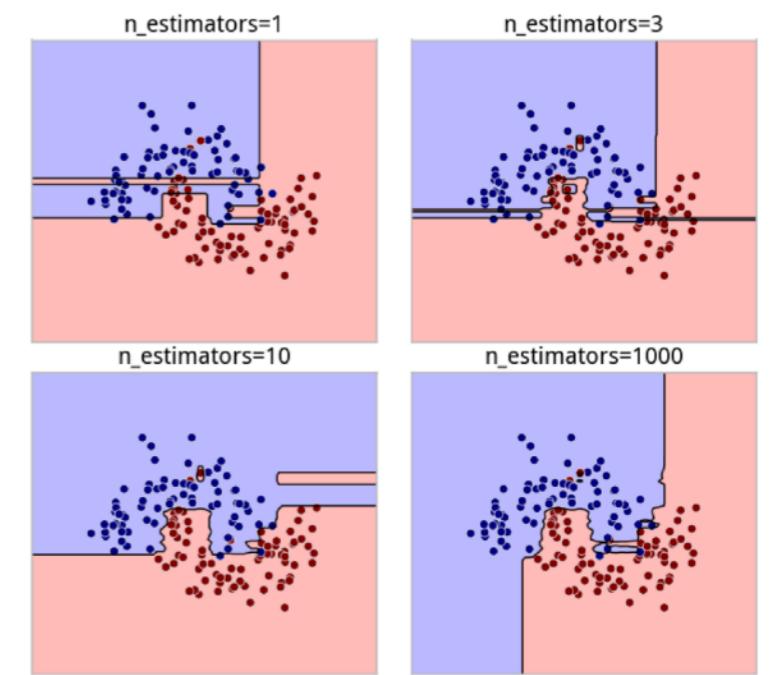
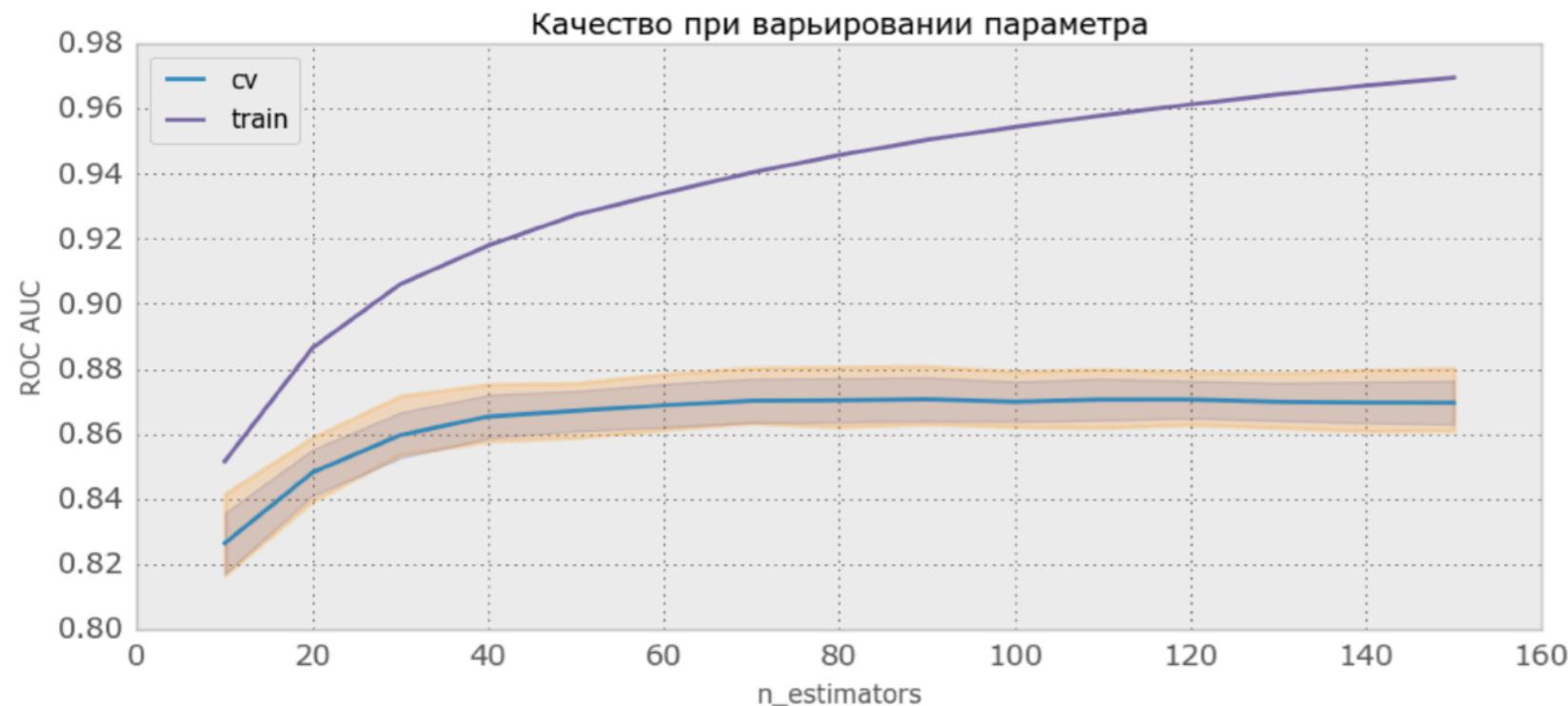
`sklearn.ensemble.RandomForestClassifier`¶

```
(n_estimators=100, *, 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, min_impurity_decrease=0.0,  
min_impurity_split=None, bootstrap=True,  
oob_score=False, n_jobs=None, random_state=None,  
verbose=0, warm_start=False, class_weight=None,  
ccp_alpha=0.0, max_samples=None)
```

`sklearn.ensemble.RandomForestRegressor`

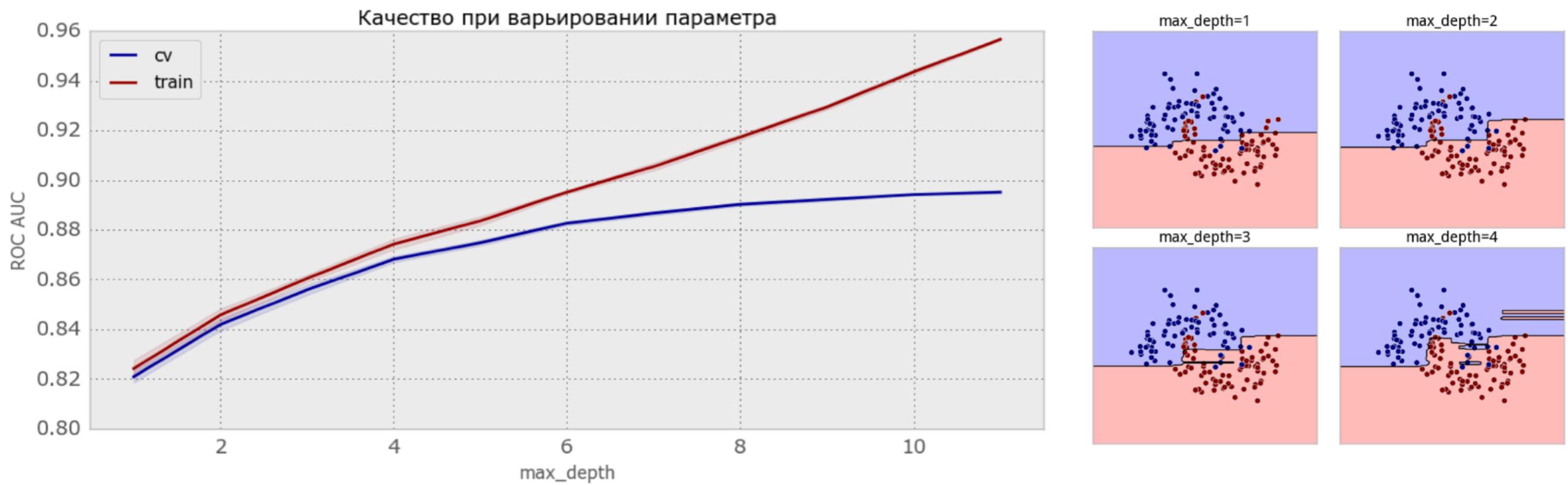
```
(n_estimators=100, *, criterion='gini', max_depth=None,  
min_samples_split=2, min_samples_leaf=1,  
min_weight_fraction_leaf=0.0, max_features='auto',  
max_leaf_nodes=None, min_impurity_decrease=0.0,  
min_impurity_split=None, bootstrap=True,  
oob_score=False, n_jobs=None, random_state=None,  
verbose=0, warm_start=False, class_weight=None,  
ccp_alpha=0.0, max_samples=None)
```

Random Forest (n_estimators)



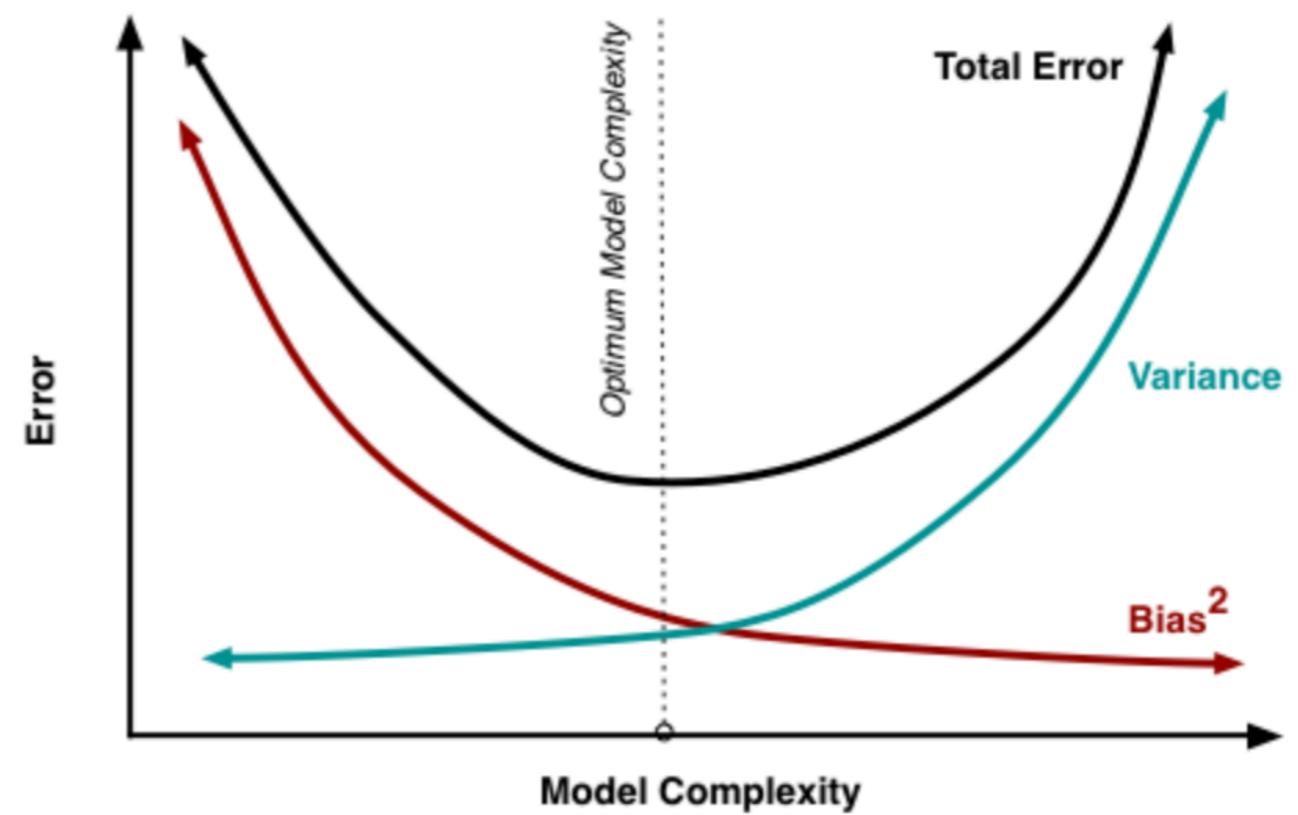
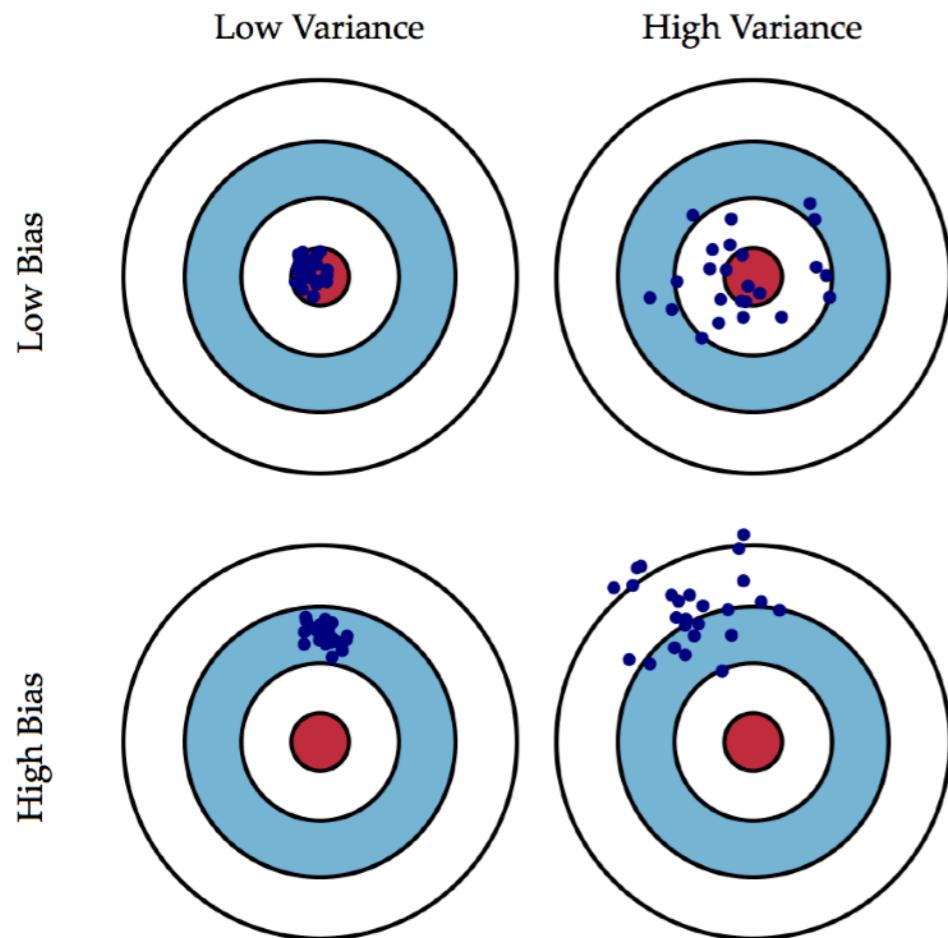
По мотивам: <https://dyakonov.org/2016/11/14/случайный-лес-random-forest/>

Random Forest (max_depth)



По мотивам: <https://dyakonov.org/2016/11/14/случайный-лес-random-forest/>

Bias and Variance tradeoff

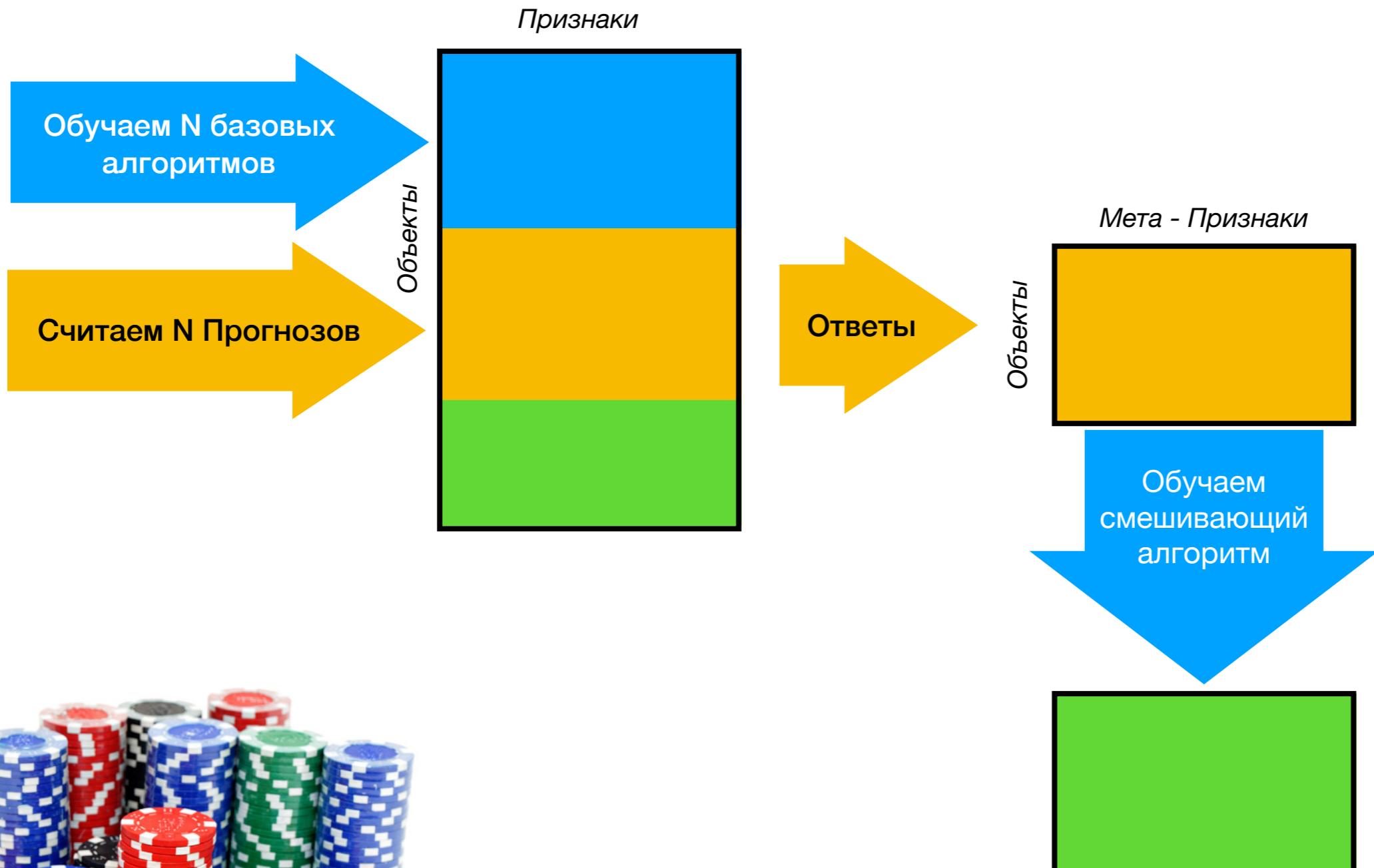


$$Err(x) = E[(Y - \hat{f}(x))^2]$$

$$Err(x) = Bias^2 + Variance + IrreducibleError$$

Еще больше ансамблей...

Stacking



Blending

Blending – частный случай Stacking'a

В качестве решающего алгоритма используется функция:

$$\sum_{i=1}^N a_i f_i(x), \quad \sum_{i=1}^N a_i = 1$$

$f_i(x)$ Базовый алгоритм

N Количество базовых алгоритмов



Ссылки

Открытый курс машинного обучения: [Тема 5](#) и [Тема 10](#)

[Репозитории Евгения Соколова](#)

Статья про [Стекинг \(Stacking\)](#) и [блэндинг \(Blending\)](#) в блоге А.Г. Дьяконова