

# Машинное обучение

## Часть I

*Власов Кирилл Вячеславович*



2019

# Содержание курса

**6 января**

AI, ML, DL - Введение  
Формальная постановка задачи  
Простейшие методы  
Метрики качества  
  
+ Домашнее задание

**11 января**

Deep Learning  
Нейронные сети  
Обучение нейронных сетей  
Свёрточные нейронные сети  
  
+ Домашнее задание

**Соревнование:**

<https://www.kaggle.com/c/bird-airplane-mipt>

# Data Scientist: The Sexiest Job of the 21st Century

© 2012 – Harvard Business Review

# Причина шумихи и ажиотажа

1.

**Быстрее, выше, сильнее!**

Вычислительные  
мощности компьютеров  
стали в разы больше и они  
только растут

2.

**Инфраструктура**

Появилась много решений  
и фреймворков для  
анализа данных.  
Большинство open-source

3.

**Больше данных!**

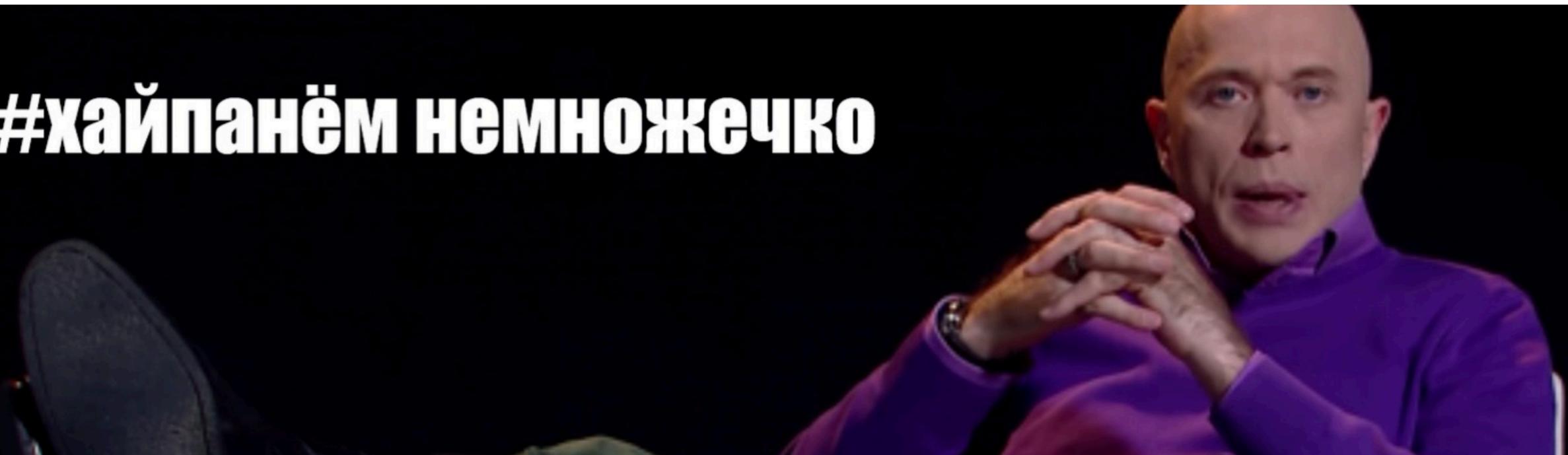
Компании собирают  
БигДату и пытаются  
извлекать из нее пользу

4.

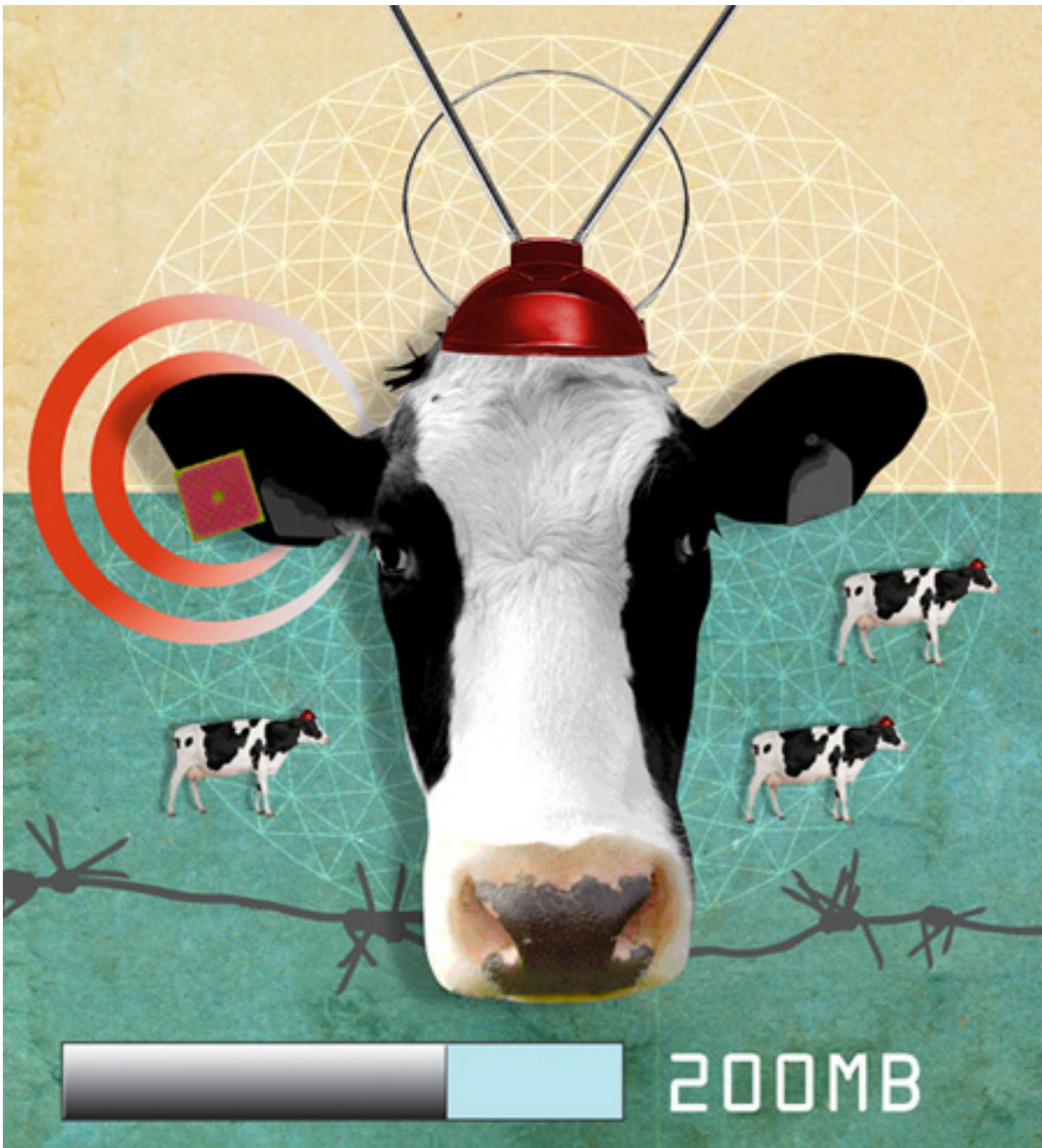
**Превосходство ИИ!**

В некоторых задачах ИИ  
справляется с решением  
лучше человека

#хайпанём немножечко



# Откуда столько данных?



**On average, each cow generates about 200 megabytes of information a year.**

© 2010 – *The Economist*,  
*Augmented business*

# Искусственный интеллект лучше человека?



1997

*Deep Blue выиграл чемпиона мира по шахматам Гарри Каспарова.*



2016

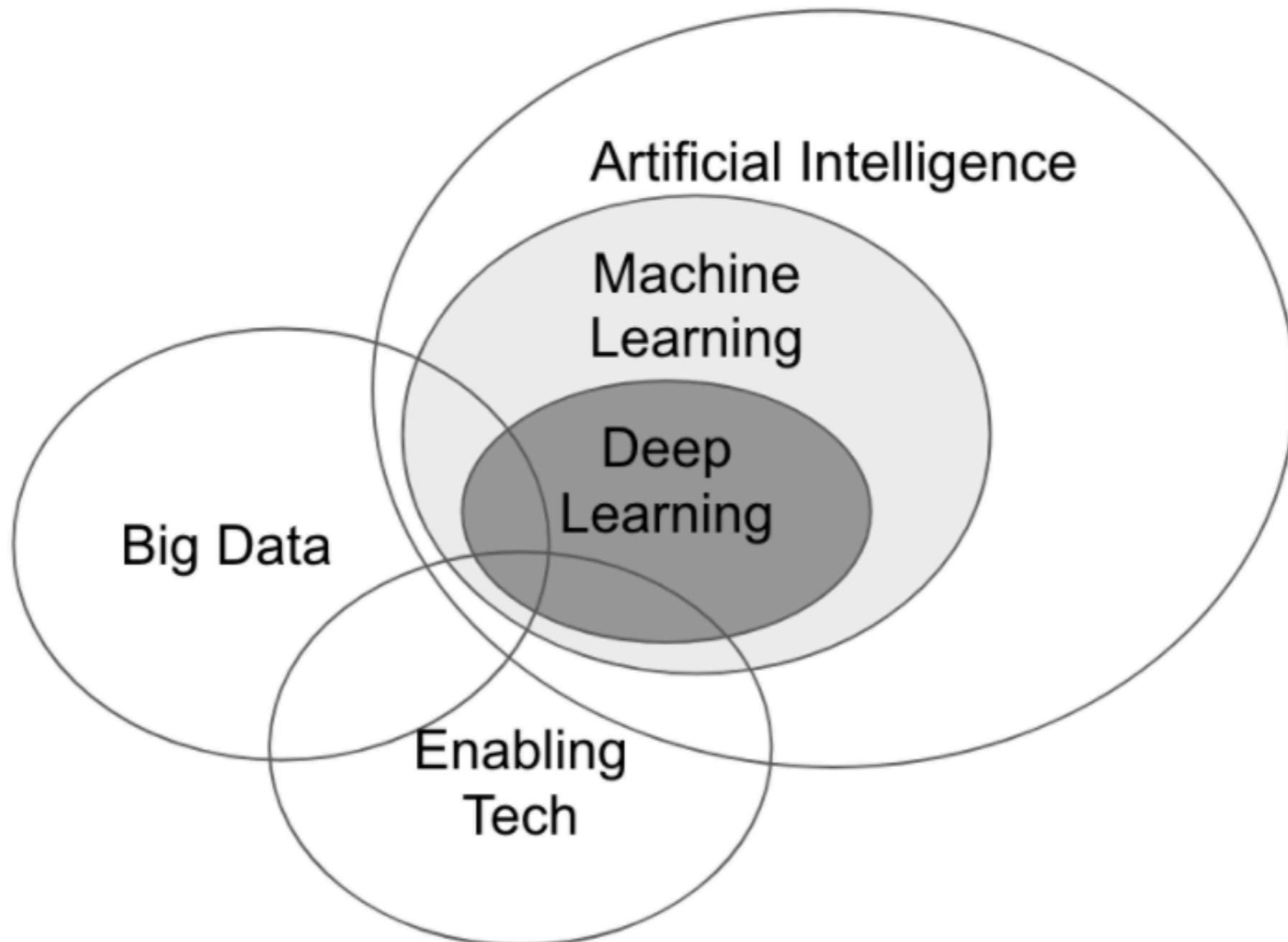
*AlphaGo выиграла матч у профессионала Ли Седоль*

# Искусственный интеллект лучше человека?

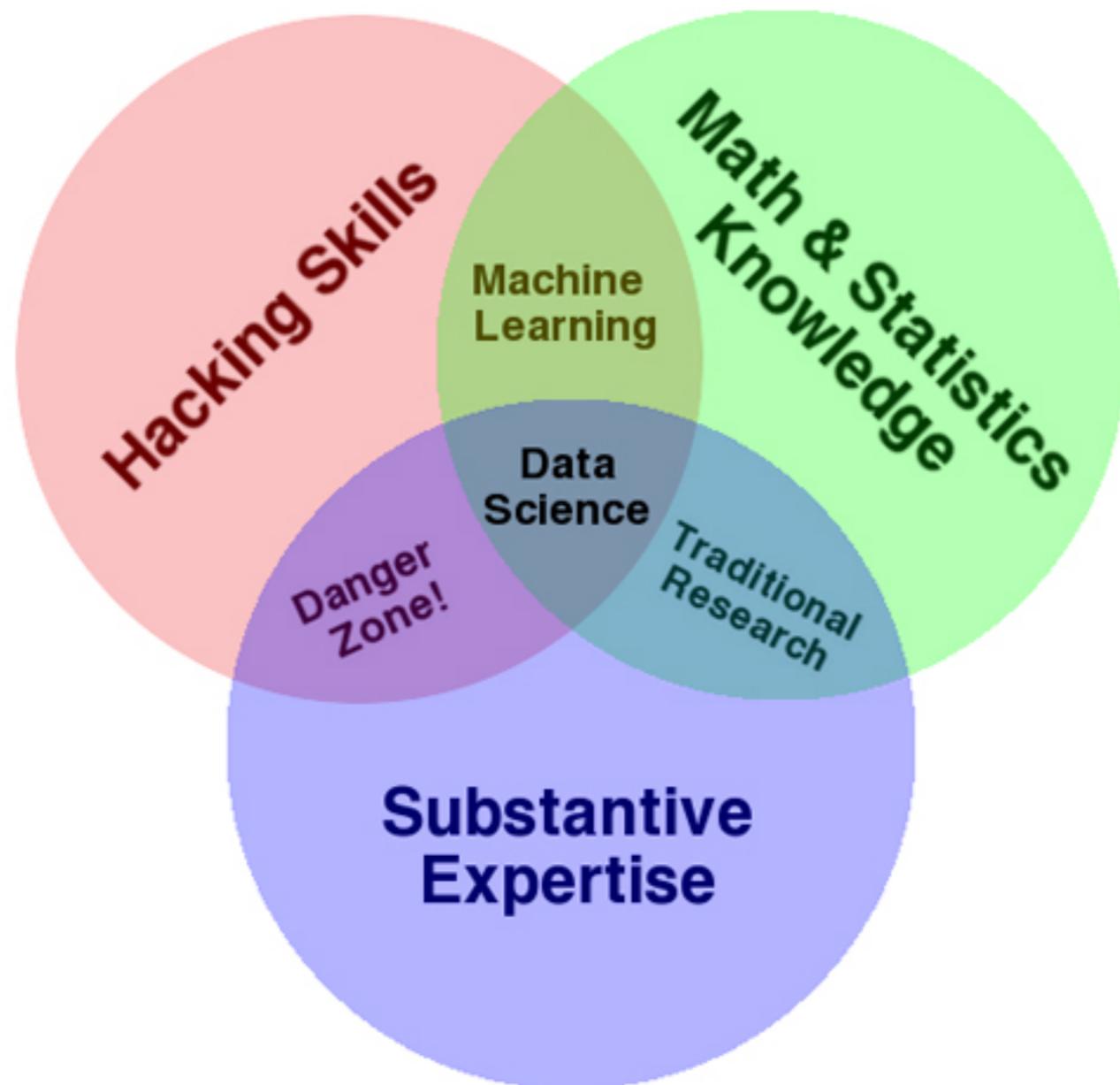
И.И.



# Место машинного обучения в области ИИ



# Место машинного обучения в области ИИ

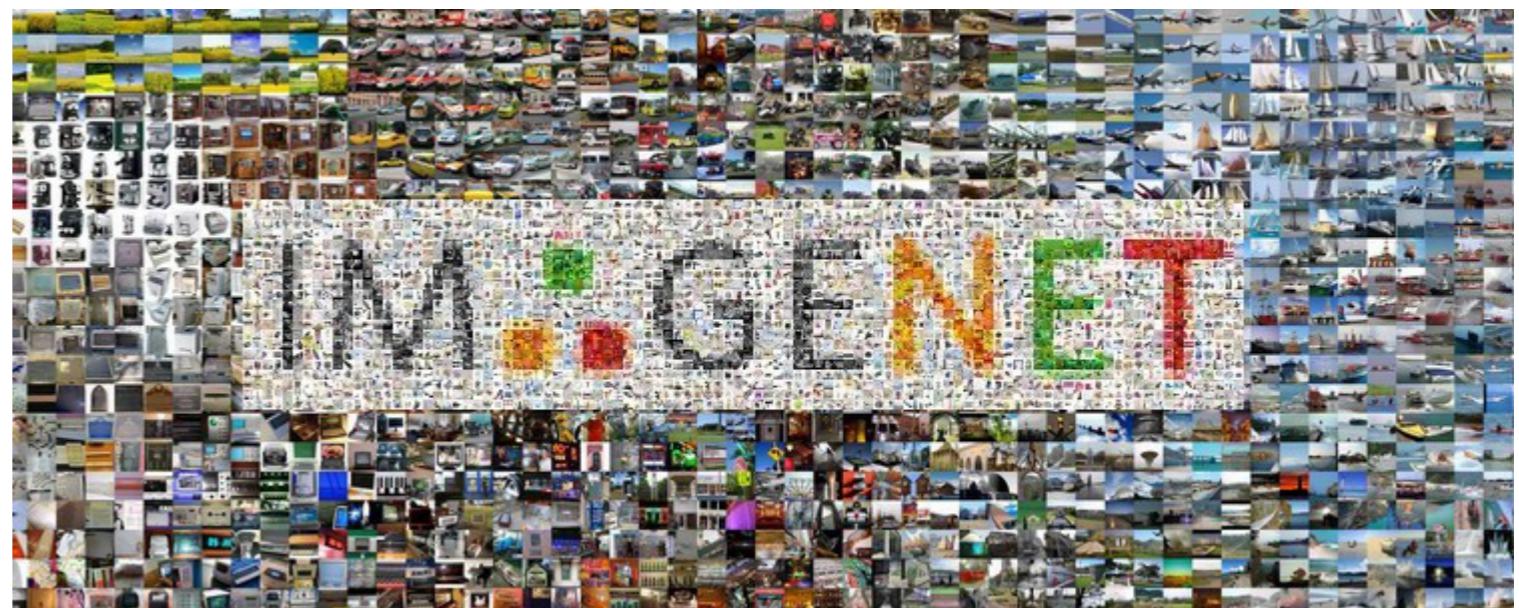


# Известные датасеты

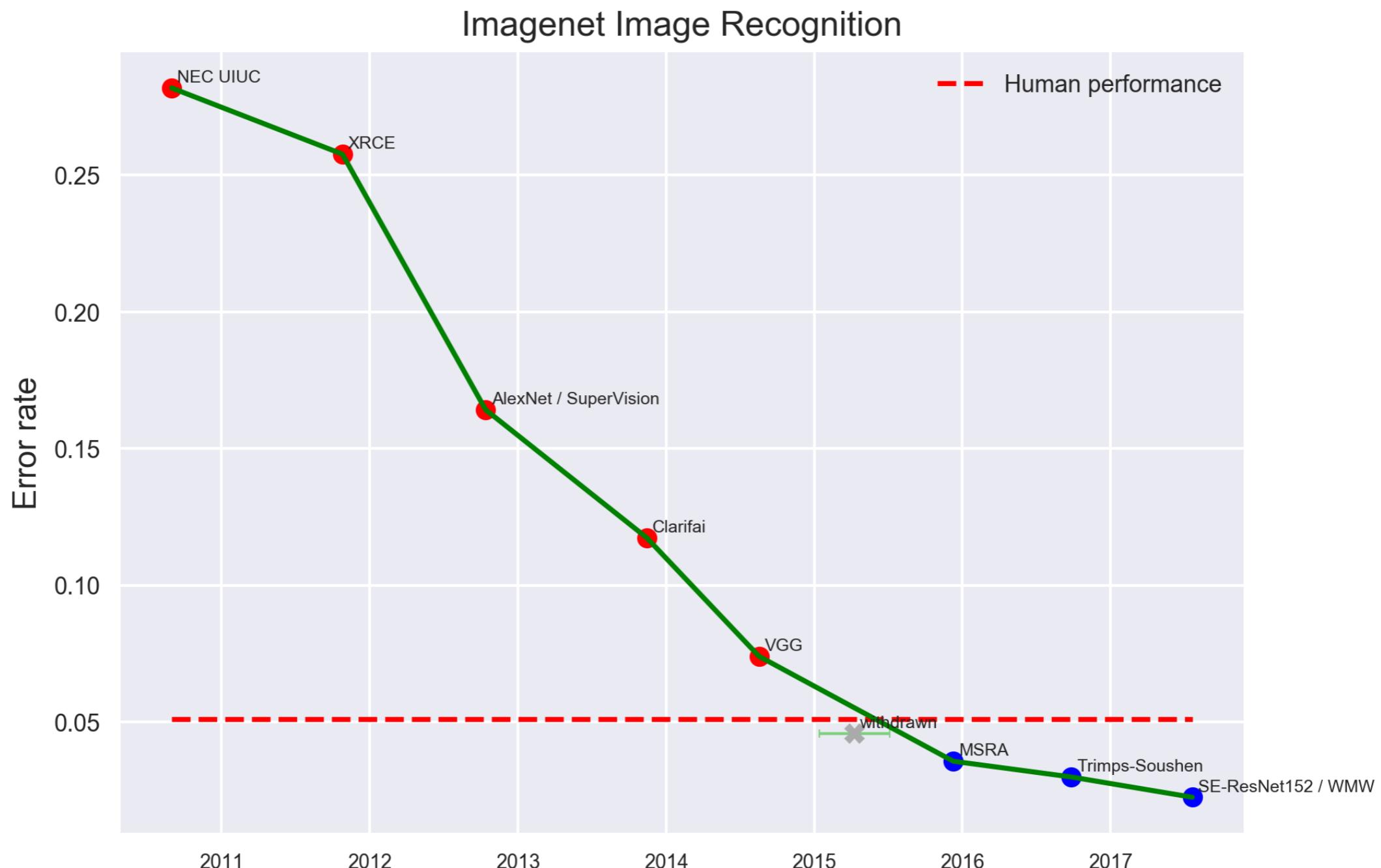
0 1 2 3 4 5 6 7 8 9  
0 1 2 3 4 5 6 7 9 9  
0 1 2 3 4 5 6 7 8 9  
0 1 2 3 4 5 6 7 8 9  
0 1 2 3 4 5 6 7 8 9

MNIST

ImageNet



# Превосходство NN над человеком



<https://www.eff.org/ai/metrics>

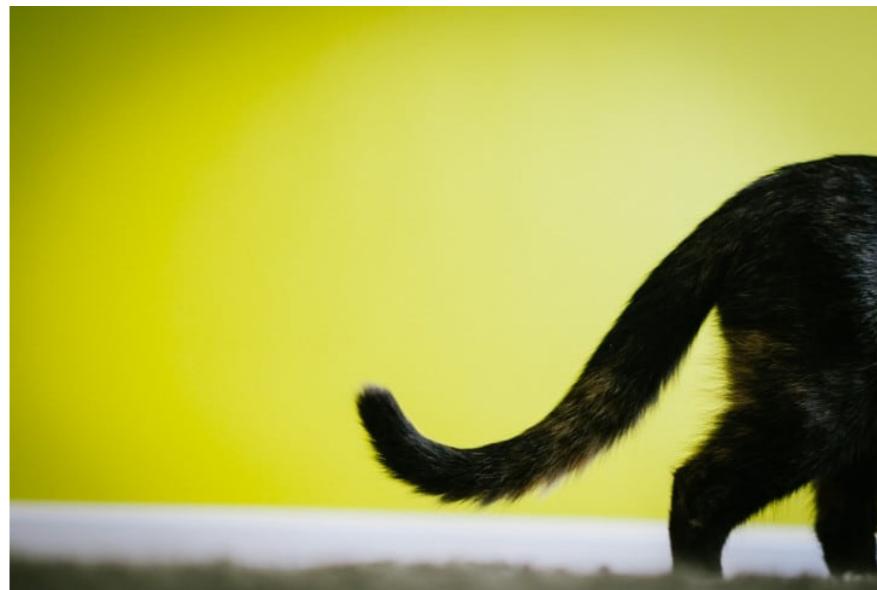
# Распознавание образов



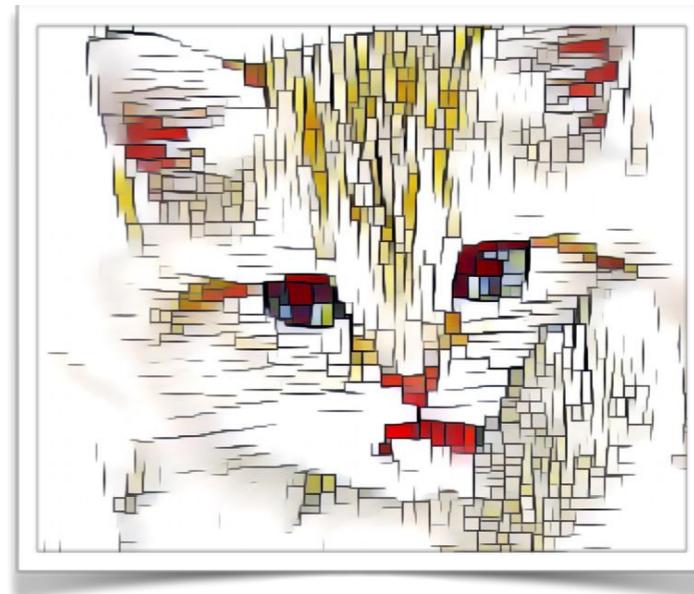
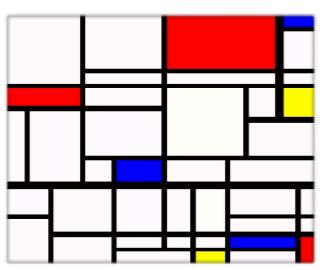
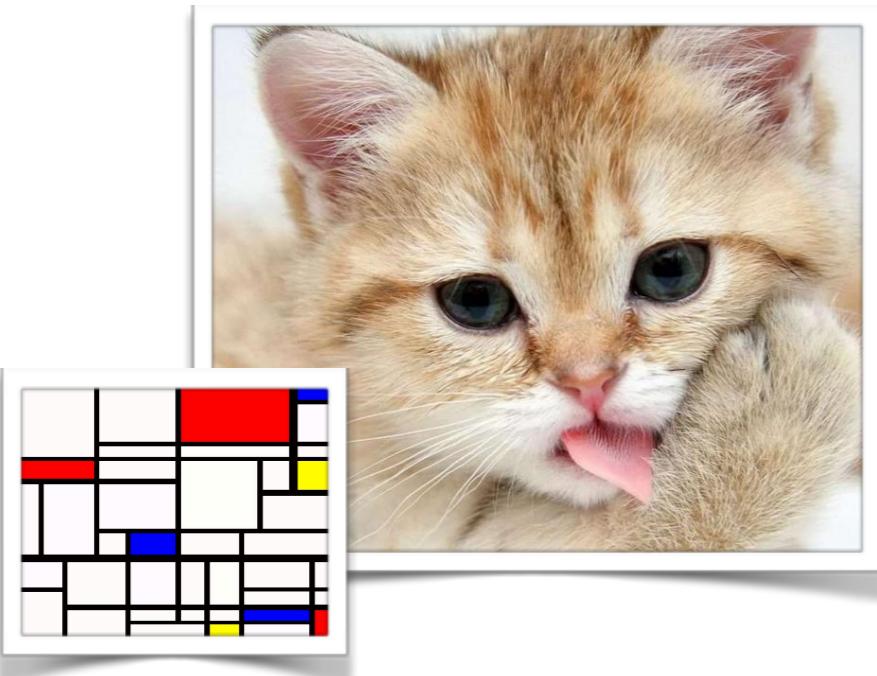
# Распознавание образов



# Распознавание образов

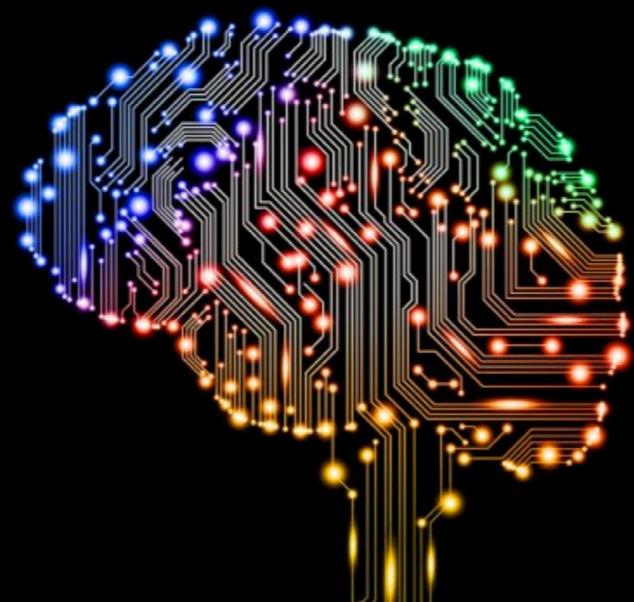


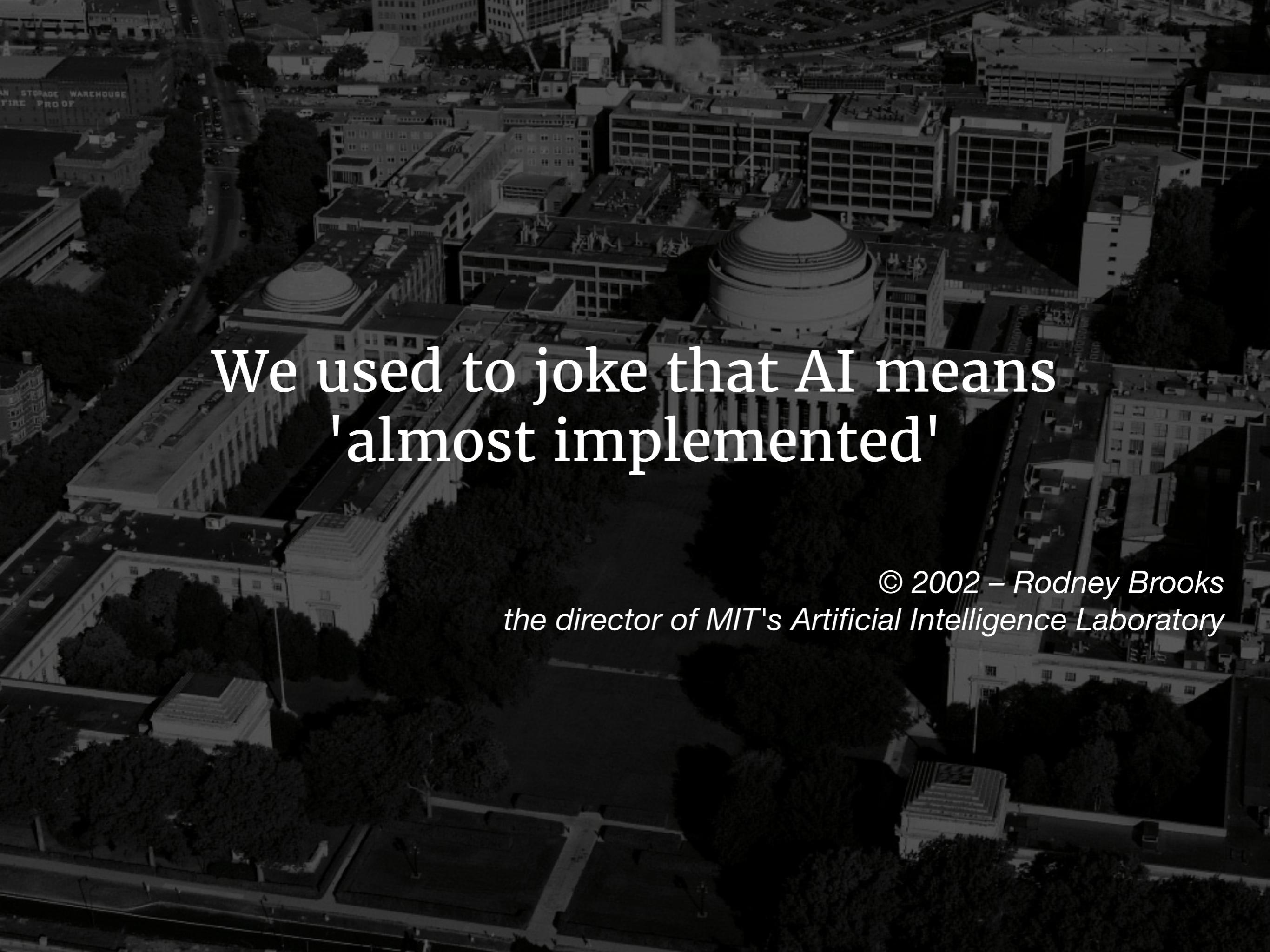
# Neural Style Transfer



Искусственный интеллект — способность интеллектуальных машин выполнять творческие функции, которые традиционно считаются прерогативой человека. Также этим термином обозначают науку и технологию создания интеллектуальных машин.

© 1956 – Джон Маккарти

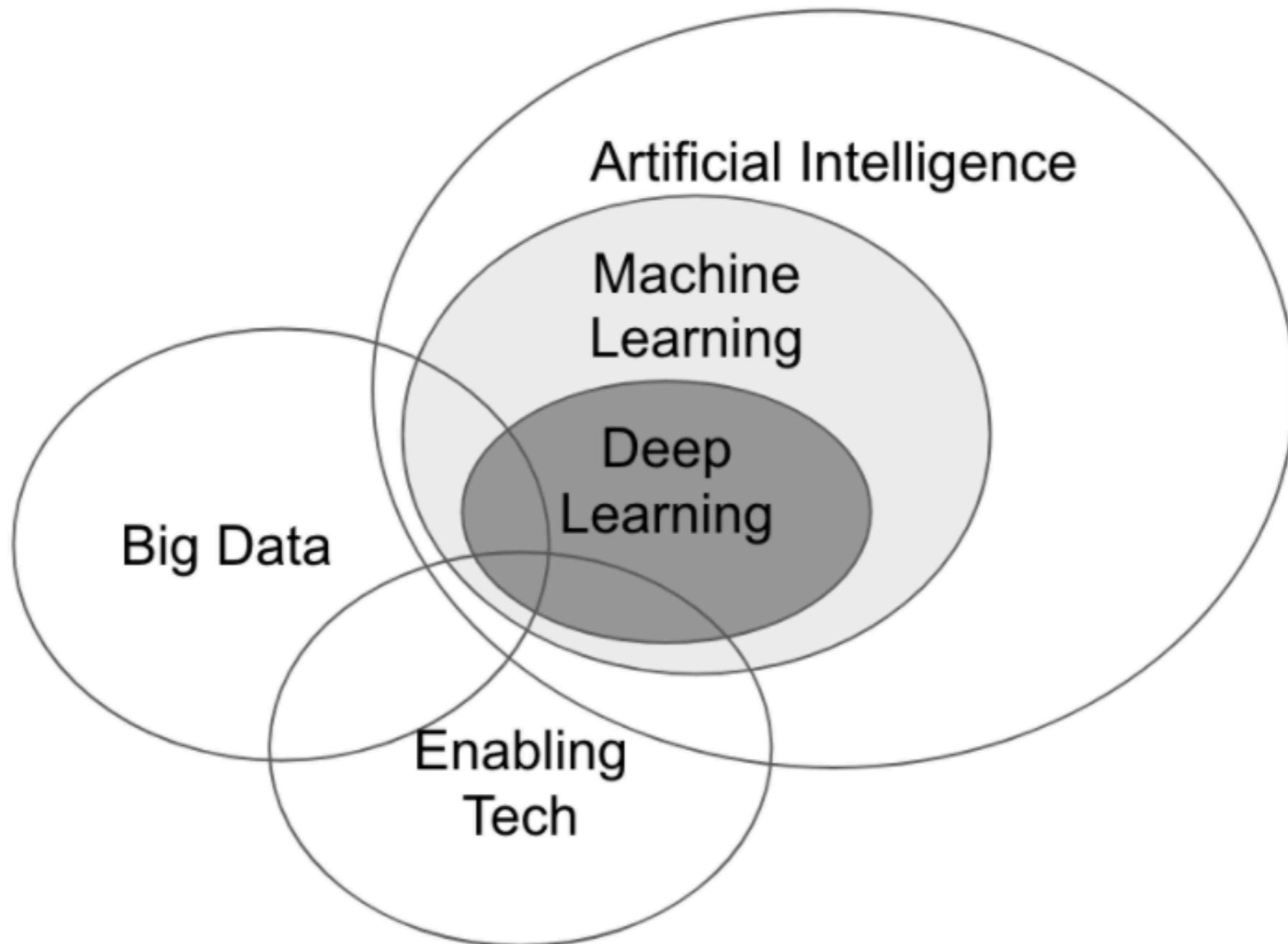


The background of the slide is a black and white aerial photograph of the Massachusetts Institute of Technology (MIT) campus. The image shows a dense cluster of buildings, including several large domes and modern structures, interspersed with green lawns and trees. The perspective is from above, looking down at the university grounds.

We used to joke that AI means  
'almost implemented'

© 2002 – Rodney Brooks  
*the director of MIT's Artificial Intelligence Laboratory*

# Место машинного обучения в области ИИ



# Определение

Машинное обучение – это процесс, в результате которого машина (компьютер) способна показывать поведение, которое в нее не было явно заложено (запрограммировано).

Артур Самуэль, 1959

Компьютерная программа обучается при решении какой-то задачи из класса Т, если ее производительность, согласно метрике Р, улучшается при накоплении опыта Е.

Том Митчелл, 1997



- Задача классификации
  - Задача регрессии
  - Задача ранжирования
  - Задача прогнозирования
- Задача кластеризации
  - Задача поиска ассоциативных правил
  - Задача фильтрации выбросов
  - Задача построения доверительной области
  - Задача сокращения размерности
  - Задача заполнения пропущенных значений

**Какие еще бывают классы задач:**

Semi-supervised learning (Частичное обучение)

Reinforcement learning (Обучение с подкреплением)

Трансдуктивное обучение

Динамическое обучение

Активное обучение

Метаобучение

# Постановка задачи

Задача: восстановить сложную зависимость по конечному числу примеров



# Обучающая выборка

## Матрица «объекты–признаки»

Датасет о задержках рейсов более 15 минут.

	Month	DayofMonth	DayOfWeek	DepTime	UniqueCarrier	Origin	Dest	Distance	dep_delayed_15min
0	c-8	c-21	c-7	1934	AA	ATL	DFW	732	N
1	c-4	c-20	c-3	1548	US	PIT	MCO	834	N
2	c-9	c-2	c-5	1422	XE	RDU	CLE	416	N
3	c-11	c-25	c-6	1015	OO	DEN	MEM	872	N
4	c-10	c-7	c-6	1828	WN	MDW	OMA	423	Y

Источник: <https://www.transtats.bts.gov>

# Обучающая выборка

## Матрица объекты–признаки

Датасет о задержках рейсов более 15 минут.

	Month	DayofMonth	DayOfWeek	DepTime	UniqueCarrier	Origin	Dest	Distance	dep_delayed_15min
0	c-8	c-21	c-7	1934	AA	ATL	DFW	732	N
1	c-4	c-20	c-3	1548	US	PIT	MCO	834	N
2	c-9	c-2	c-5	1422	XE	RDU	CLE	416	N
3	c-11	c-25	c-6	1015	OO	DEN	MEM	872	N
4	c-10	c-7	c-6	1828	WN	MDW	OMA	423	Y

Признаки

Источник: <https://www.transtats.bts.gov>

# Обучающая выборка

## Матрица «объекты–признаки»

Датасет о задержках рейсов более 15 минут.

	Month	DayofMonth	DayOfWeek	DepTime	UniqueCarrier	Origin	Dest	Distance	dep_delayed_15min
0	c-8	c-21	c-7	1934	AA	ATL	DFW	732	N
1	c-4	c-20	c-3	1548	US	PIT	MCO	834	N
2	c-9	c-2	c-5	1422	XE	RDU	CLE	416	N
3	c-11	c-25	c-6	1015	OO	DEN	MEM	872	N
4	c-10	c-7	c-6	1828	WN	MDW	OMA	423	Y

Объекты (прецеденты)

Источник: <https://www.transtats.bts.gov>

# Обучающая выборка

## Матрица «объекты–признаки»

Датасет о задержках рейсов более 15 минут.

	Month	DayofMonth	DayOfWeek	DepTime	UniqueCarrier	Origin	Dest	Distance	dep_delayed_15min
0	c-8	c-21	c-7	1934	AA	ATL	DFW	732	N
1	c-4	c-20	c-3	1548	US	PIT	MCO	834	N
2	c-9	c-2	c-5	1422	XE	RDU	CLE	416	N
3	c-11	c-25	c-6	1015	OO	DEN	MEM	872	N
4	c-10	c-7	c-6	1828	WN	MDW	OMA	423	Y

Целевая переменная

Источник: <https://www.transtats.bts.gov>

# Формальная постановка задачи

Дана обучающая выборка (объекты независимы):

$$X_m = \{ (x_1, y_1), \dots, (x_m, y_m) \}$$

Для задачи регрессии - Целевая переменная задана вещественным числом

$$(x_1, y_1) \in \mathbb{R}^m \times \mathbb{Y}, \mathbb{Y} = \mathbb{R}$$

Для задачи классификации - Целевая переменная задана конечным числом меток

$$(x_1, y_1) \in \mathbb{R}^m \times \mathbb{Y}, \mathbb{Y} = \{-1; 1\}$$

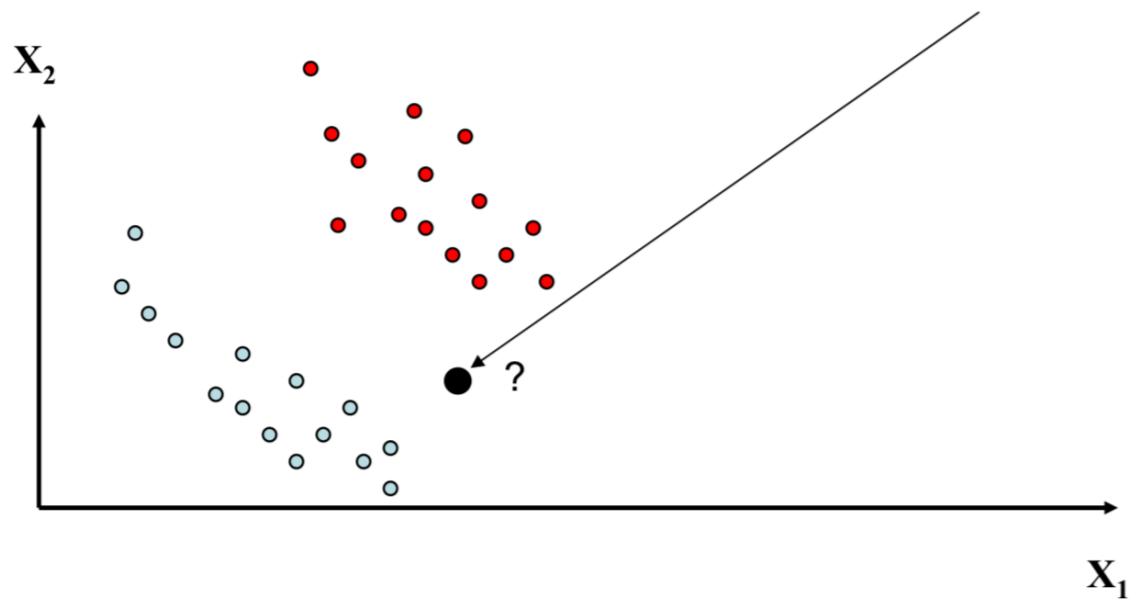
Задать такую функцию  $f(x)$  от вектора признаков  $x$ , которое выдает ответ для любого возможного наблюдения  $x$

$$f(x): \mathbb{X} \rightarrow \mathbb{Y}$$

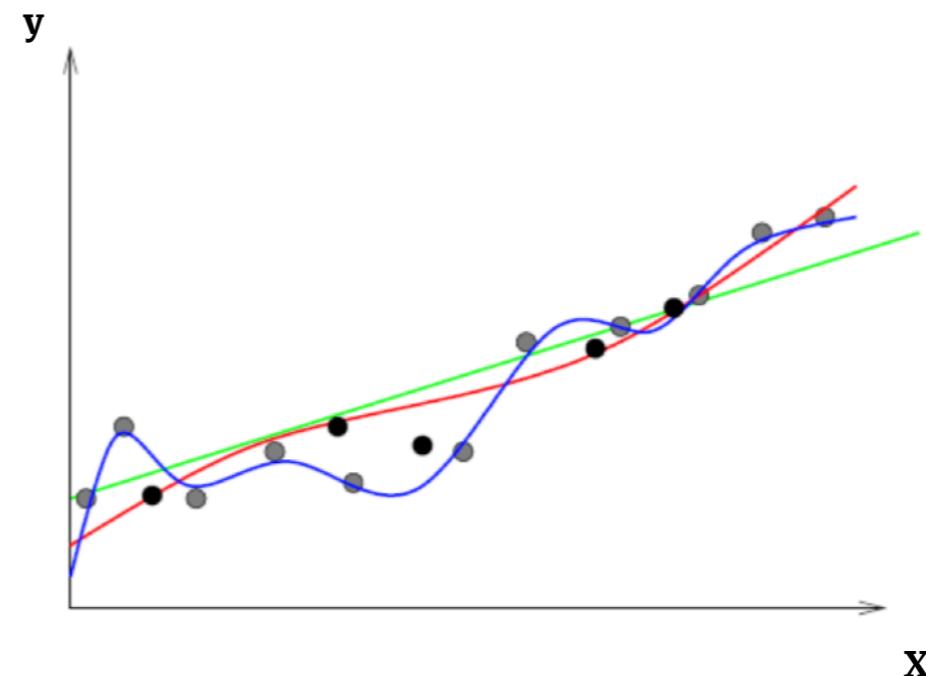
**Основная гипотеза МО:** Схожим объектам соответствуют схожие объекты

# Формальная постановка задачи

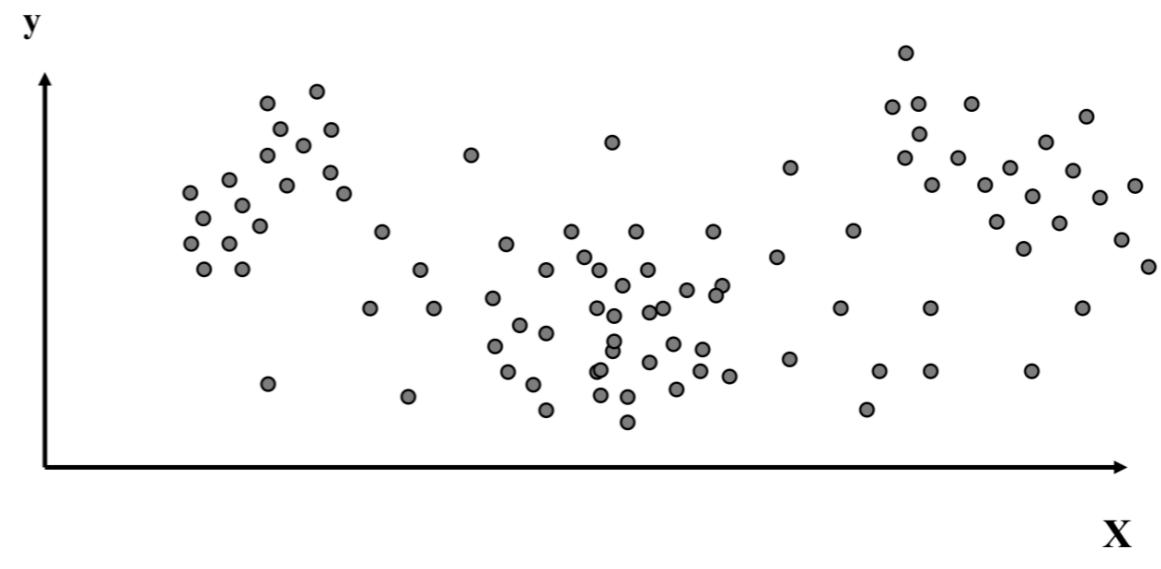
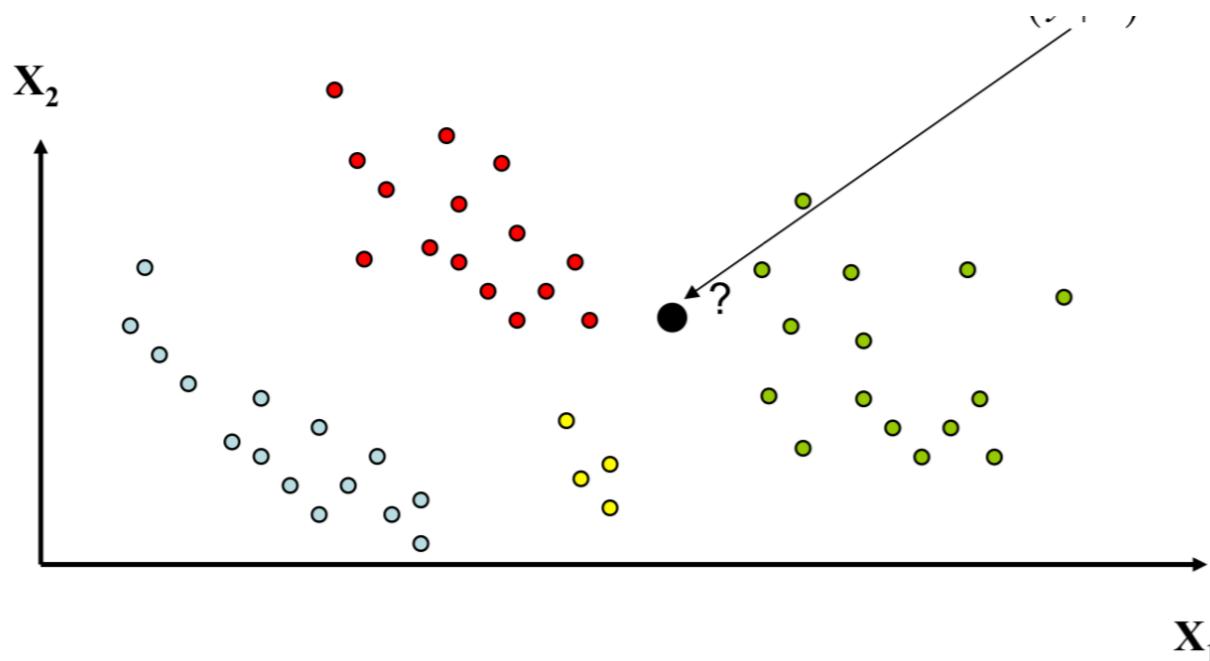
**Классификация**



**Восстановление регрессии**



**Кластеризация**



# Признаки

Признаковое описание объекта - Вектор:

$$x_i = \{d_1, d_2, d_3, \dots, d_n\}$$

Множество значений признака

$$d_j \in D_j$$

Бинарные признаки

$$D_j = \{0, 1\}$$

В нашем примере:  
Целевая переменная

Категориальные признаки

$D_j$  - упорядоченное множество

В нашем примере:  
Локация отправления  
Локация прибытия

Вещественные признаки

$$D_j = \mathbb{R}^m$$

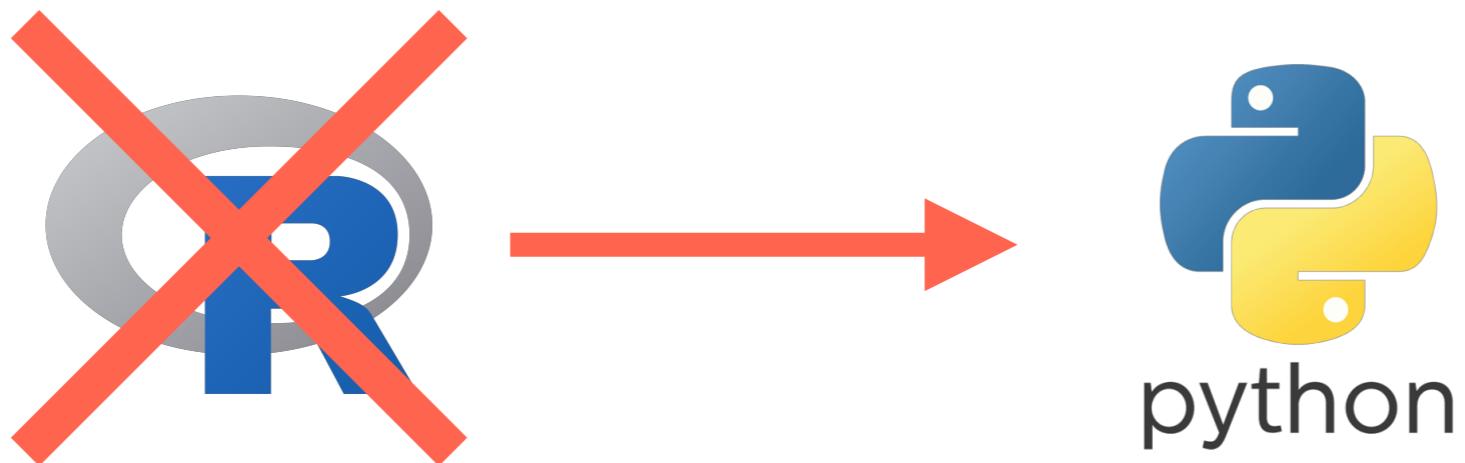
В нашем примере:  
Расстояние

# Инструменты и библиотеки



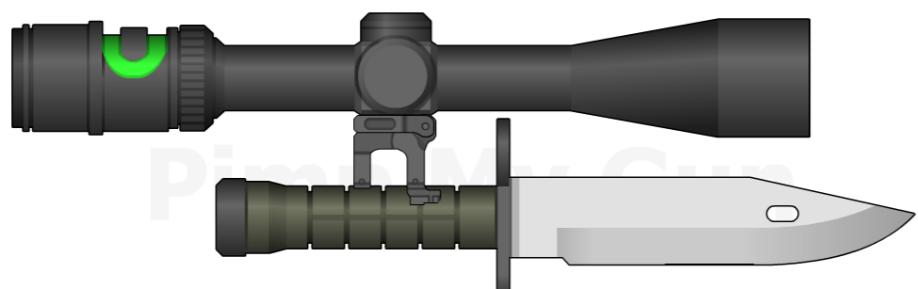
Подробнее про R vs Python: <https://habr.com/company/piter/blog/263457/>

# Инструменты и библиотеки



# Инструменты и библиотеки

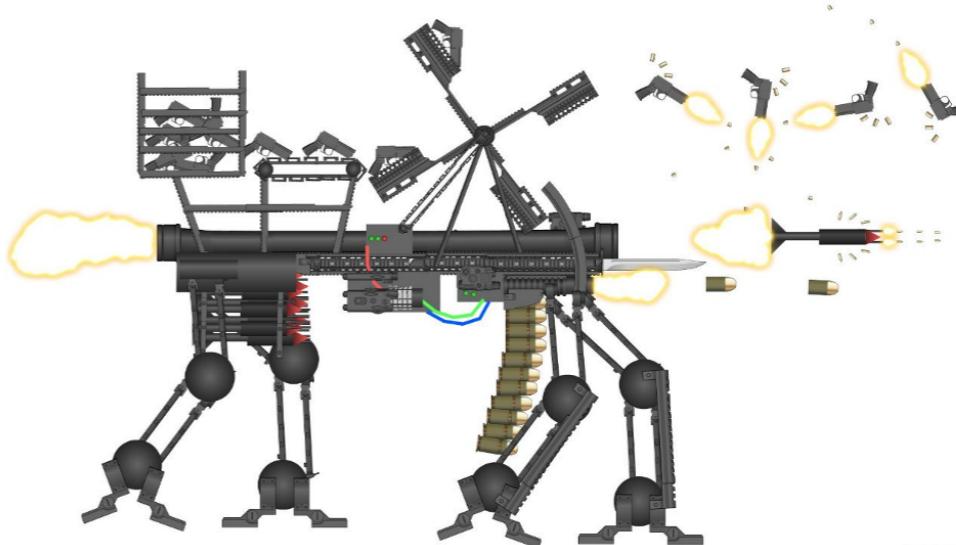
Assembly



C++



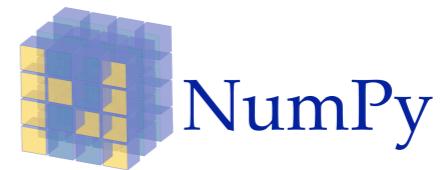
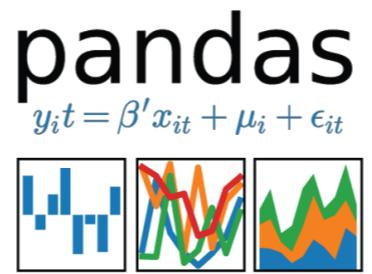
Python



C



# Инструменты и библиотеки



**XGBoost**



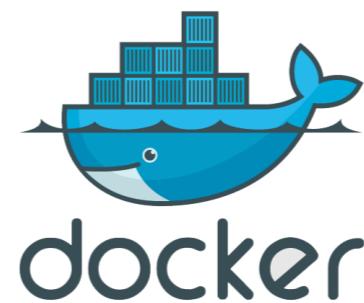
**P Y T  R C H**



# Инструменты и библиотеки



<https://www.anaconda.com/download/>



<https://www.docker.com/products/docker-desktop>

<https://hub.docker.com/r/vlasoff/ds/>

# Метод k ближайших соседей (kNN, *k Nearest Neighbours*)

**Гипотеза компактности:** если мера сходства объектов введена достаточно удачно, то схожие объекты гораздо чаще лежат в одном классе, чем в разных.

- Вычислить расстояние до каждого из объектов обучающей выборки
- Отобрать k объектов обучающей выборки, расстояние до которых минимально
- Класс классифицируемого объекта — это класс, наиболее часто встречающийся среди k ближайших соседей



## `sklearn.neighbors.KNeighborsRegressor`

`(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2,  
metric='minkowski', metric_params=None, n_jobs=None, **kwargs)`

## `sklearn.neighbors.KNeighborsClassifier`

`(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2,  
metric='minkowski', metric_params=None, n_jobs=None, **kwargs)`

# Метод k ближайших соседей (kNN, *k Nearest Neighbours*)

**Гипотеза компактности:** если мера сходства объектов введена достаточно удачно, то схожие объекты гораздо чаще лежат в одном классе, чем в разных.

- Вычислить расстояние до каждого из объектов обучающей выборки
- Отобрать k объектов обучающей выборки, расстояние до которых минимально
- Класс классифицируемого объекта — это класс, наиболее часто встречающийся среди k ближайших соседей



## `sklearn.neighbors.KNeighborsRegressor`

`(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)`

## `sklearn.neighbors.KNeighborsClassifier`

`(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)`

### Выбор числа соседей k

При  $k=1$  алгоритм ближайшего соседа неустойчив к шумовым выбросам: он даёт ошибочные классификации не только на самих объектах-выбросах, но и на ближайших к ним объектах других классов.

При  $k=l$ , наоборот, алгоритм чрезмерно устойчив и вырождается в константу. Таким образом, крайние значения нежелательны.

# Метод k ближайших соседей (kNN, *k Nearest Neighbours*)

## Выбор метрики

Евклидово расстояние (“euclidean”)

$$\sqrt{\sum (x - y)^2}$$

Расстояние городских кварталов «манхэттенское расстояние» (“manhattan”)

$$\sum |x - y|$$

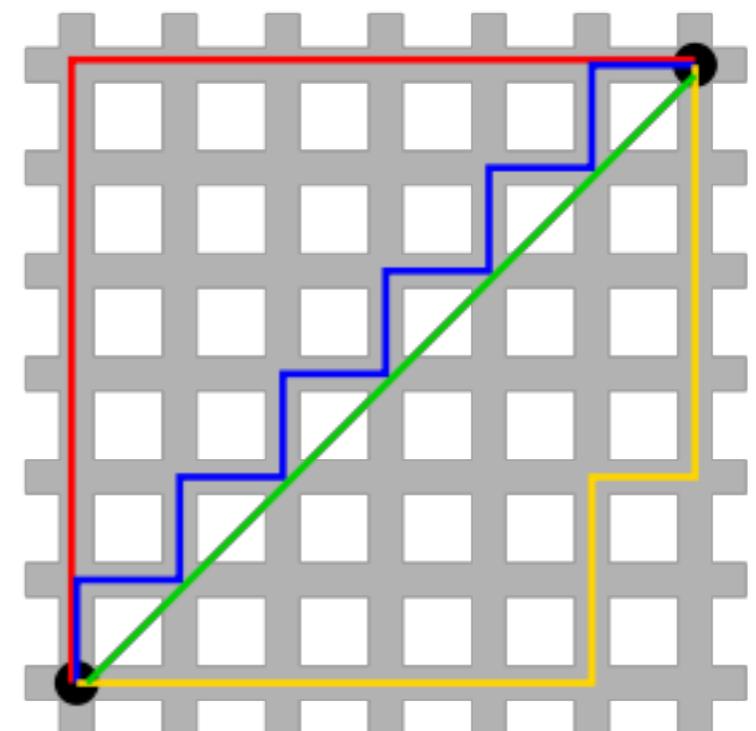
Расстояние Чебышева “chebyshev”

$$\max(x - y)$$

Расстояние Минковского “minkowski”

$$\left( \sum |x - y|^p \right)^{\frac{1}{p}}$$

расстояния с параметром  $p$  равным 1 (расстояние городских кварталов) или 2 (евклидова метрика).  
 $p = \infty$  метрика обращается в расстояние Чебышёва.



# Метод k ближайших соседей (kNN, *k Nearest Neighbours*)

## Нормирование признаков

$$z = \frac{x - \min(x)}{\max(x) - \min(x)}$$

$$z \in [0,1]$$

## Стандартизация признаков

$$z = \frac{x - \mu}{\sigma}$$

где

$\mu$  - Среднее

$\sigma$  - Стандартное отклонение

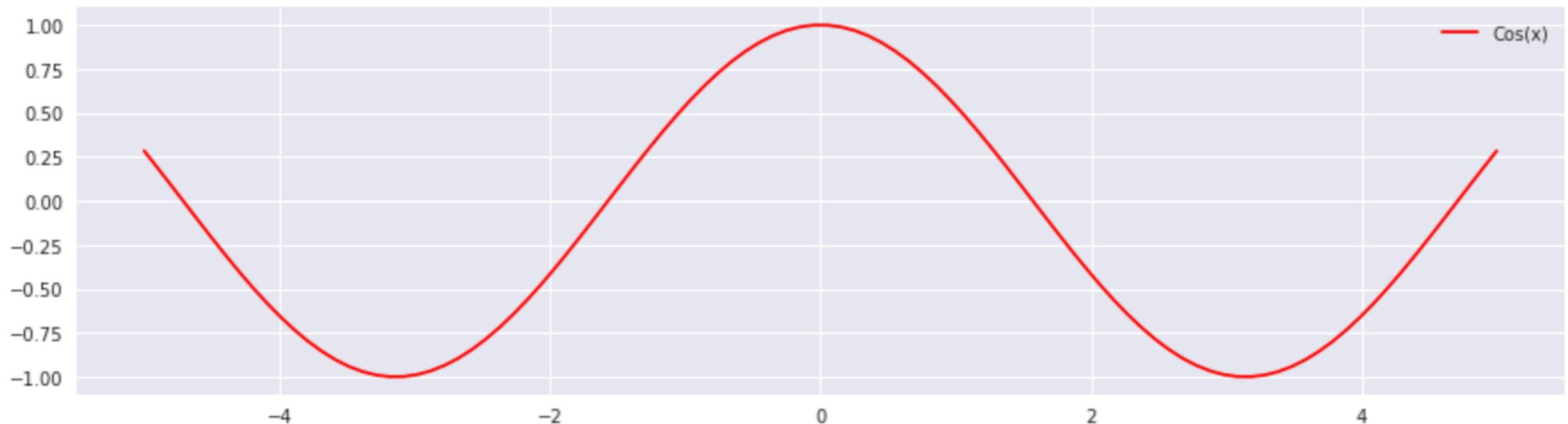
## Проклятие размерности

В пространстве высокой размерности все объекты примерно одинаково далеки друг от друга; выбор ближайших соседей становится практически произвольным.

**Рассмотрим пример на практике**

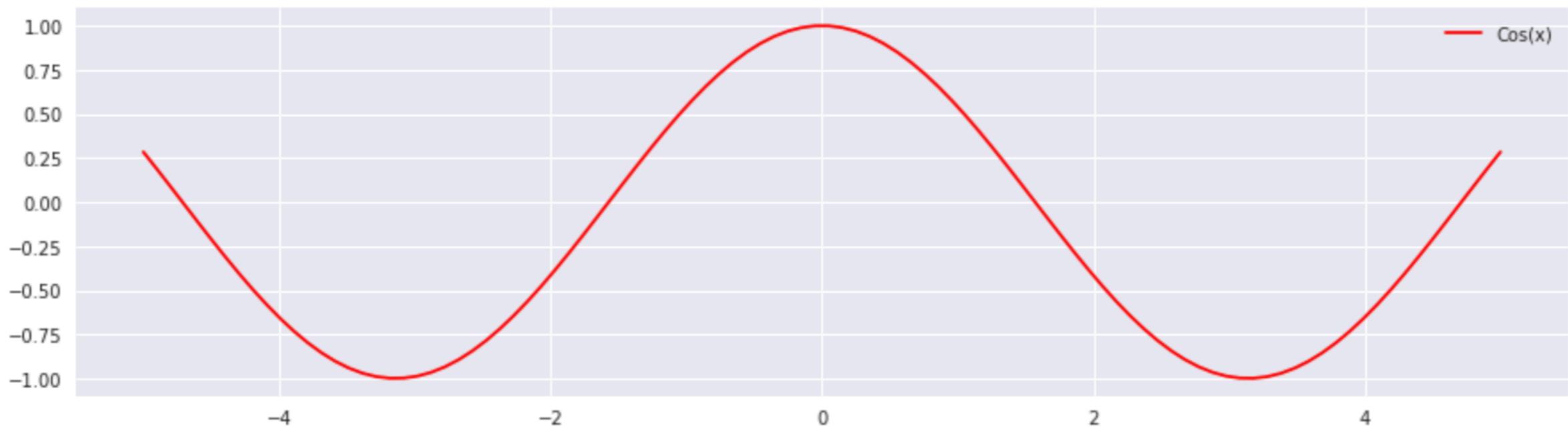
# Метрики качества

$$y = \cos(x), x \in [-5, 5]$$

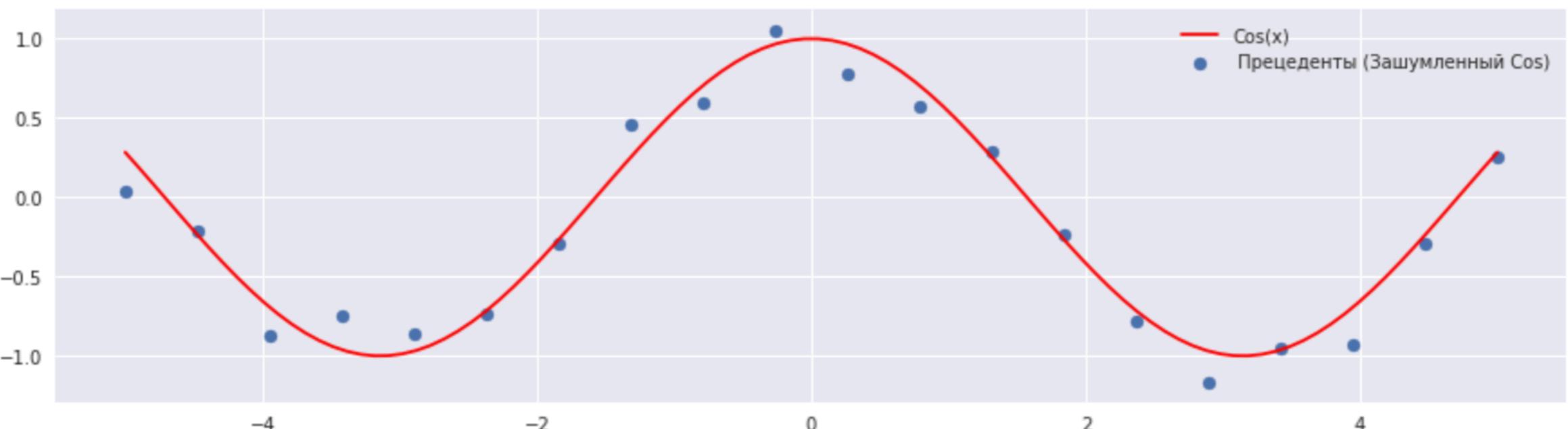


# Метрики качества

$$y = \cos(x), x \in [-5, 5]$$

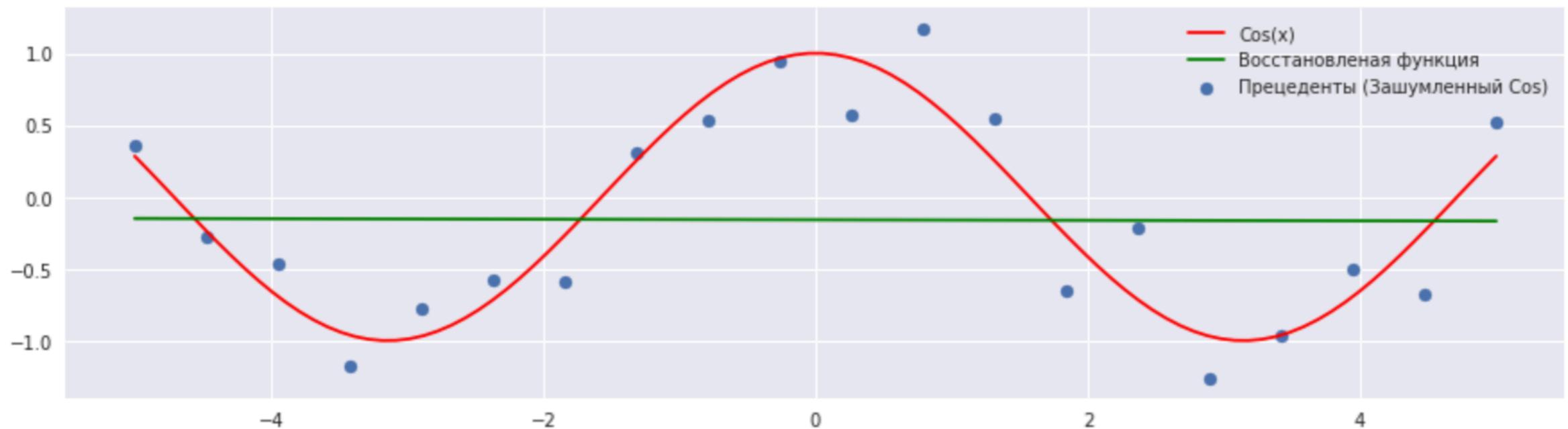


$$y = \cos(x) + \varepsilon, \text{ где } \varepsilon = \mathcal{N}\left(0, \frac{1}{2}\right), x \in [-5, 5]$$



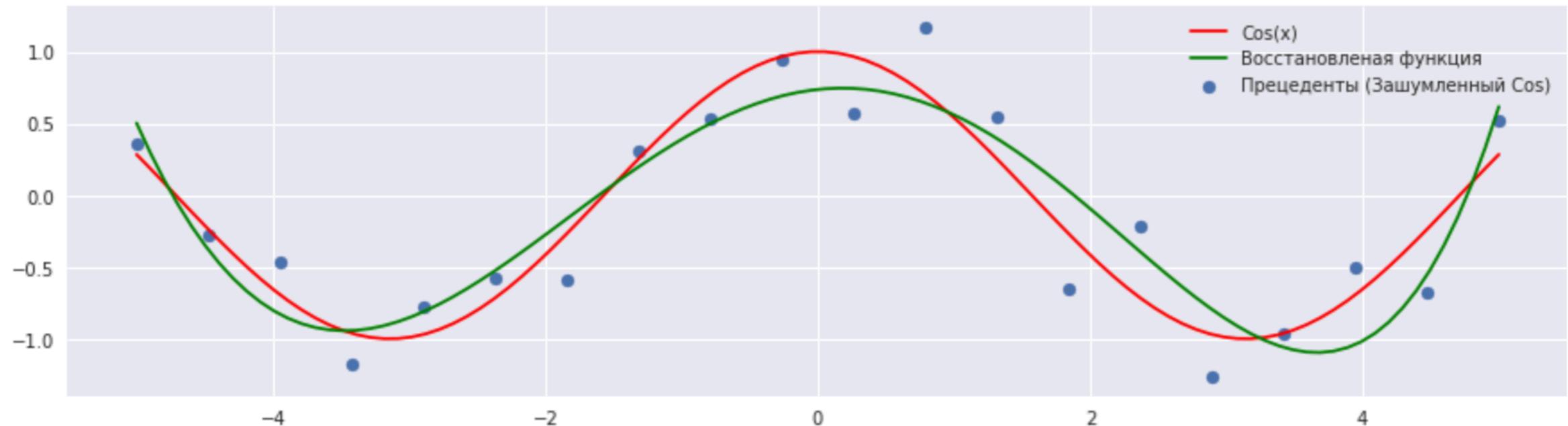
# Метрики качества

**Восстановим зависимость линейной функцией**



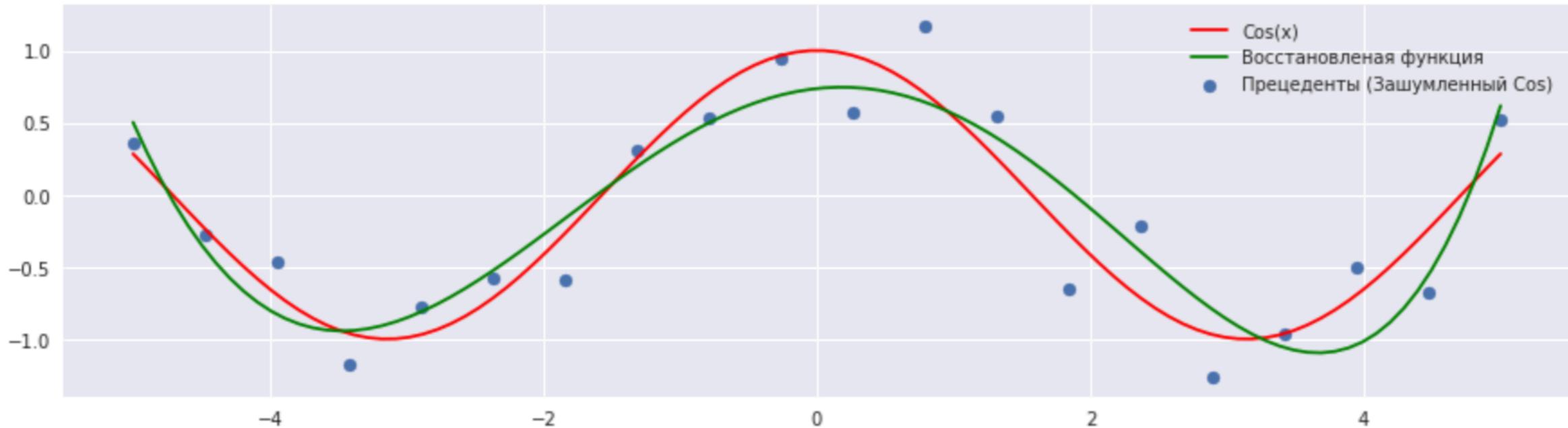
# Метрики качества

**Восстановим зависимость с помощью полинома 5-ого порядка**

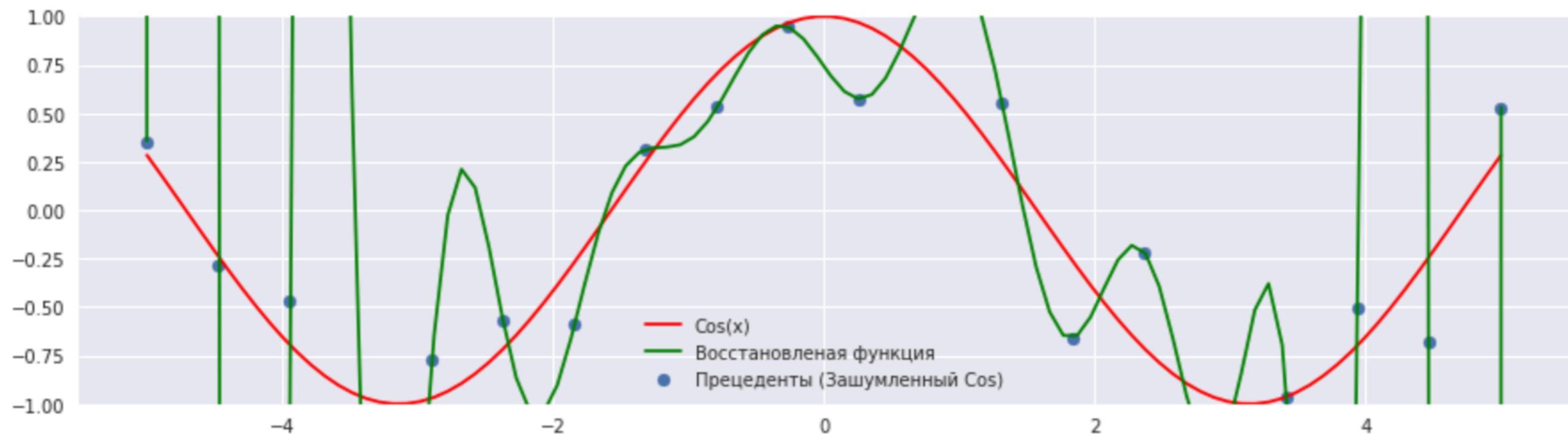


# Метрики качества

**Восстановим зависимость с помощью полинома 5-ого порядка**



**Восстановим зависимость с помощью полинома 11-ого порядка**



# Метрики качества в задачах регрессии

Средняя квадратичная (*Mean Squared Error, MSE*) ошибка:

$$MSE = \frac{1}{l} \sum_{i=1}^l (f(x_i) - y_i)^2$$

Средняя абсолютная (*Mean Absolute Error, MAE*) ошибка:

$$MAE = \frac{1}{l} \sum_{i=1}^l |f(x_i) - y_i|$$

Среднеквадратичный функционал сильнее штрафует за большие отклонения по сравнению со среднеабсолютным, и поэтому более чувствителен к выбросам.

Идеальны для сравнения моделей, но не всегда понятно как их оценивать относительно целевой переменной. Например: MSE - 10 это хорошо, если переменная лежит в пределах интервала (10000, 100000), но плохо, если целевая переменная принимает значения от 0 до 1

# Метрики качества в задачах регрессии

Коэффициент детерминации ( $R^2$ ):

$$R^2 = 1 - \frac{\sum_{i=1}^l (f(x_i) - y_i)^2}{\sum_{i=1}^l (y_i - \bar{y})^2}, \text{ где } \bar{y} = \sum_{i=1}^l y_i$$

Коэффициент детерминации измеряет долю дисперсии, объясненную моделью, в общей дисперсии целевой переменной

Если она близка к единице, то модель хорошо объясняет данные, если же она близка к нулю, то прогнозы сопоставимы по качеству с константным предсказанием.

**Другие полезные метрики:**

Квантильная регрессия

Mean Absolute Percentage Error

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	90	20
	No	10	50

Выборка: Всего 170

Положительного класса 100

Отрицательного класса 70

Прогноз:

Положительного класса 110

Отрицательного класса 60

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

Доля правильных ответов (*accuracy*):

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

Доля правильных ответов (*accuracy*):

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Сколько в нашем примере?  
О чём говорит эта цифра?

		Actual class	
		Yes	No
Predicted class	Yes	90	20
	No	10	50

# Метрики качества в задачах классификации

Еще один пример:

		Actual class	
		Yes	No
Predicted class	Yes	90	5
	No	10	5

Доля правильных ответов (accuracy):

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

# Метрики качества в задачах классификации

Еще один пример:

		Actual class	
		Yes	No
Predicted class	Yes	90	5
	No	10	5

Доля правильных ответов (accuracy):

$$\text{accuracy} = \frac{90 + 5}{90 + 5 + 10 + 5} = 86,4$$

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	90	5
	No	10	5

Доля правильных ответов (*accuracy*):

$$\text{accuracy} = \frac{90 + 5}{90 + 5 + 10 + 5} = 86,4$$

Давайте всегда будем предсказывать константным значением (110 объектов и все положительные). Посчитайте accuracy

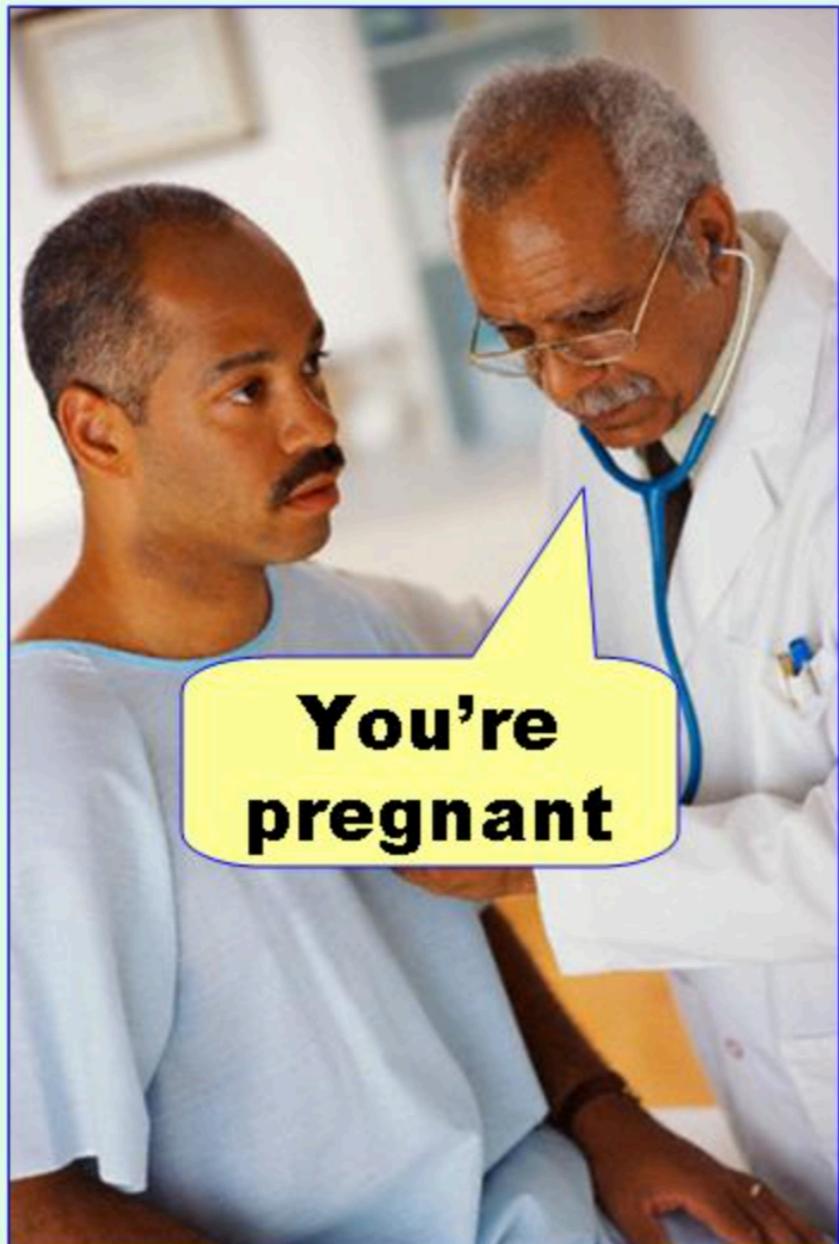
# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class		
		Yes	No	
Predicted class	Yes	True Positive (TP)	False Positive (FP)	Ошибка I-ого рода
	No	False Negative (FN)	True Negative (TN)	Ошибка II-ого рода

# Метрики качества в задачах классификации

**Type I error**  
(false positive)



**Type II error**  
(false negative)



# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

Точность (*precision*):

$$precision = \frac{TP}{TP + FP}$$

доля объектов, предсказанных как положительные, действительно является положительными.

Полнота (*recall*):

$$recall = \frac{TP}{TP + FN}$$

Доля положительных объектов, которую выделил классификатор

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

Точность (*precision*):

$$precision = \frac{TP}{TP + FP}$$

Полнота (*recall*):

$$recall = \frac{TP}{TP + FN}$$

Какая ошибку важнее оптимизировать?

Например:

1. Решаем задачу, уйдет ли от нас клиент. Какая цена ошибки?
2. Выявление фрода. Заблокировать хорошего клиента или пропустить злоумышленника?
3. Кредитный скоринг. Выдать кредит злостному неплательщику или не выдать положительному?

# Метрики качества в задачах классификации

Матрица ошибок (*confusion matrix*):

		Actual class	
		Yes	No
Predicted class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

F-мера:

$$F_\beta = (1 + \beta^2) \frac{precision \times recall}{\beta^2 precision + recall}$$

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

$\beta$  в данном случае определяет вес точности в метрике, а при  $\beta=1$  это среднее гармоническое

F-мера достигает максимума при полноте и точности, равными единице, и близка к нулю, если один из аргументов близок к нулю.

# Метрики качества в задачах классификации

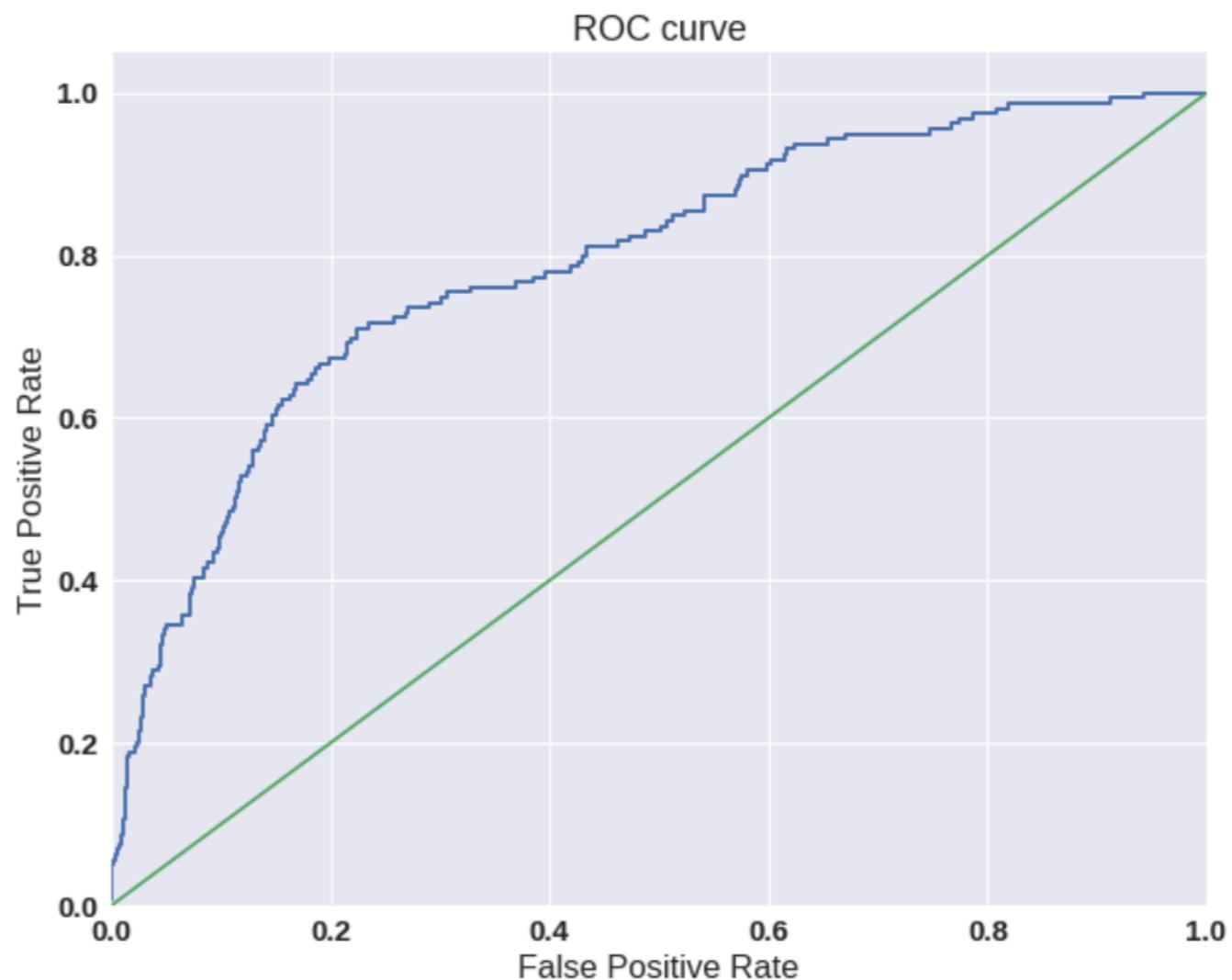
*ROC AUC*

или площадь (*Area Under Curve*) под кривой ошибок (*Receiver Operating Characteristic curve* ).

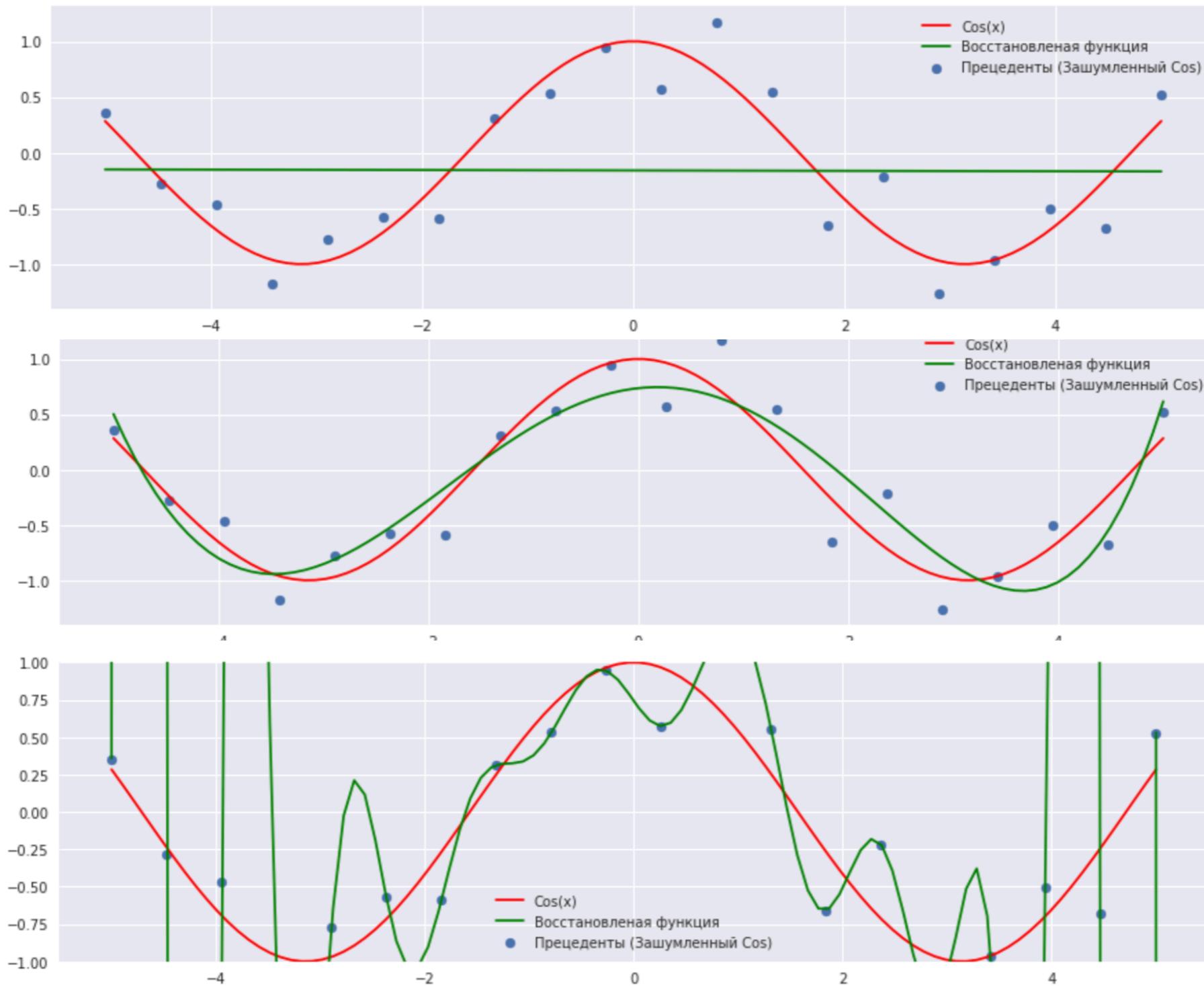
Кривая ошибок (*Receiver Operating Characteristic curve* ) представляет из себя линию от (0,0) до (1,1) в координатах True Positive Rate (TPR) и False Positive Rate (FPR):

$$TPR = \frac{TP}{TP + FN} = recall$$

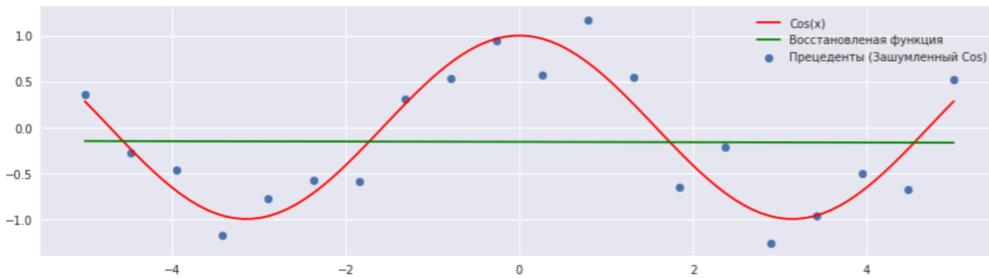
$$FPR = \frac{FP}{FP + TN}$$



# Мотивация валидации

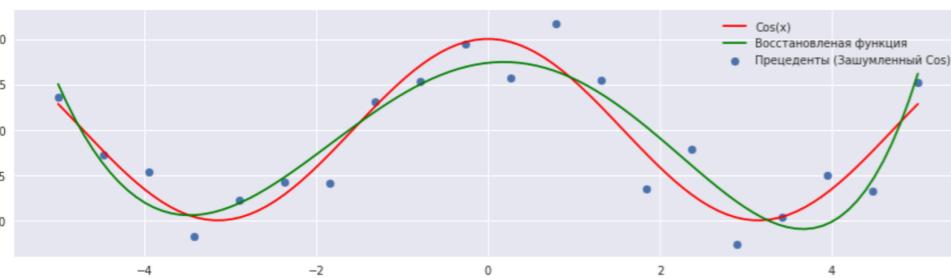


# Мотивация валидации



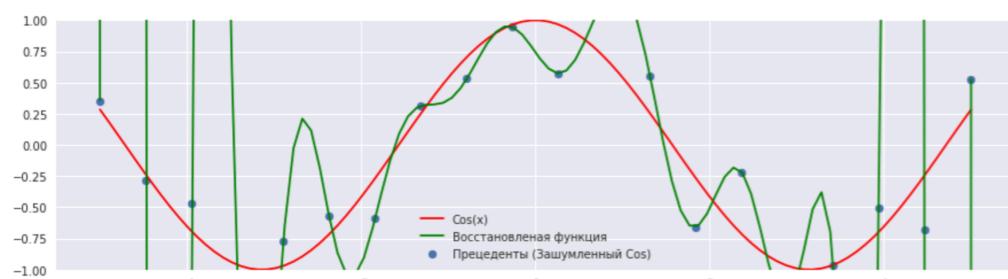
MSE	MAE	R2
-----	-----	----

Линейная модель	0.472	0.586	0.0004
-----------------	-------	-------	--------

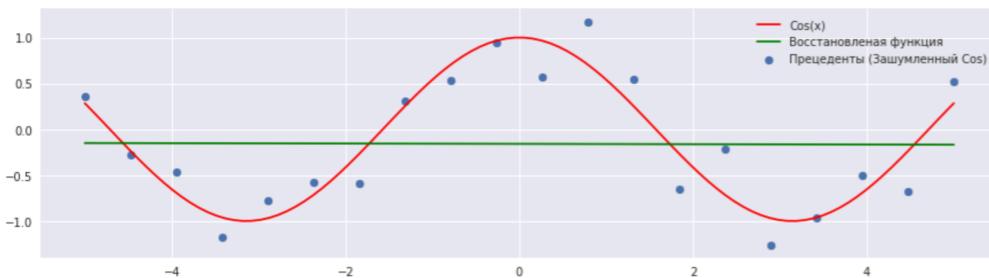


Полином 5-ой степени	0.047	0.179	0.9000
----------------------	-------	-------	--------

Полином 11-ой степени	0.000	0.000	1.0000
-----------------------	-------	-------	--------

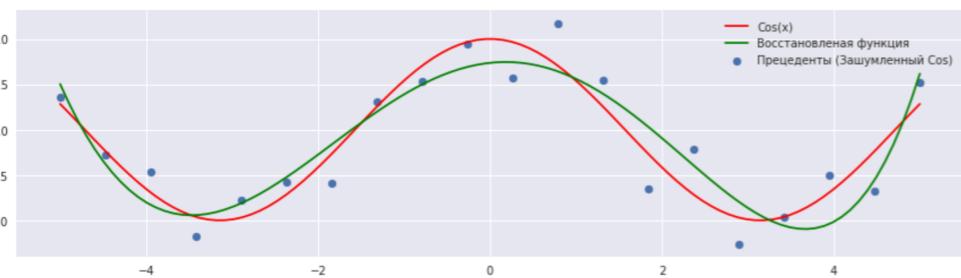


# Мотивация валидации



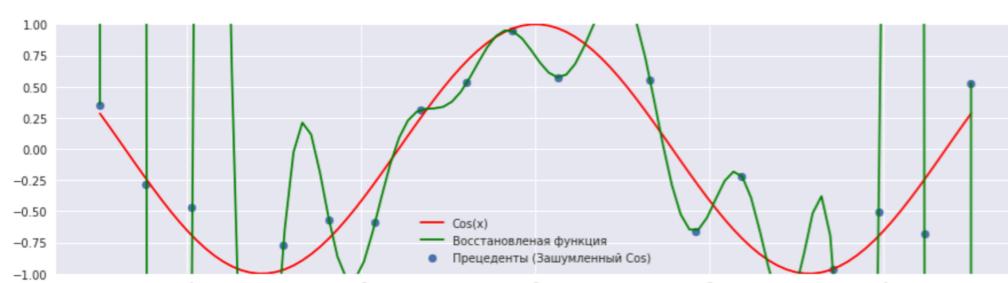
MSE	MAE	R2
-----	-----	----

Линейная модель	0.472	0.586	0.0004
-----------------	-------	-------	--------



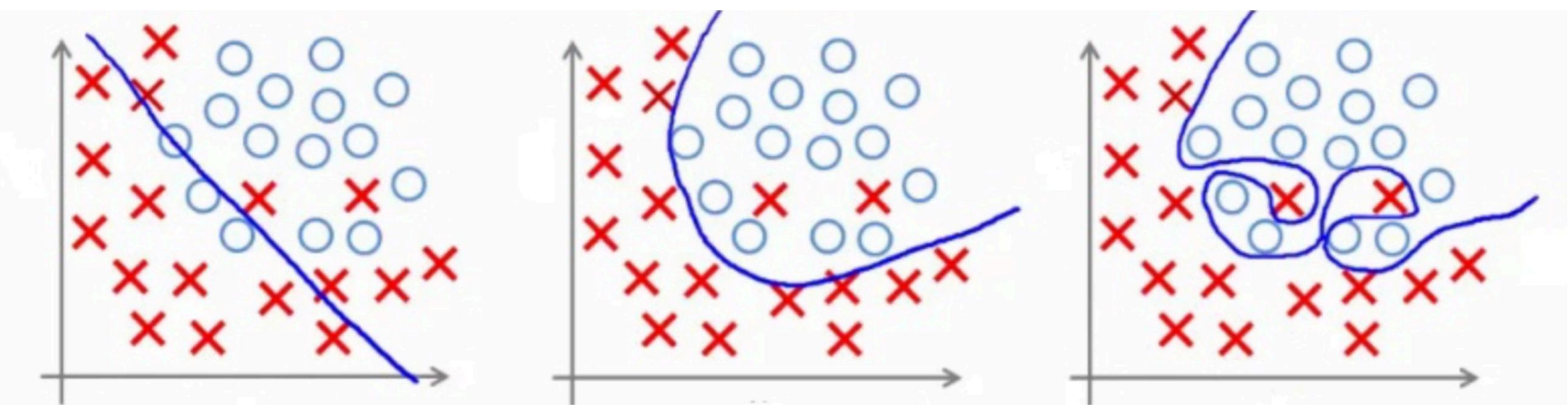
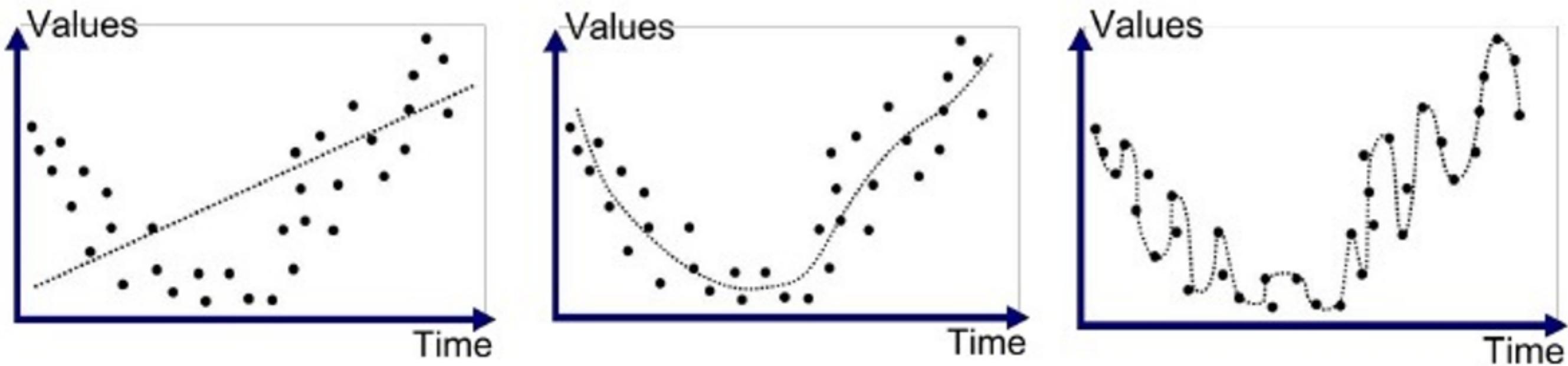
Полином 5-ой степени	0.047	0.179	0.9000
----------------------	-------	-------	--------

Полином 11-ой степени	0.000	0.000	1.0000
-----------------------	-------	-------	--------

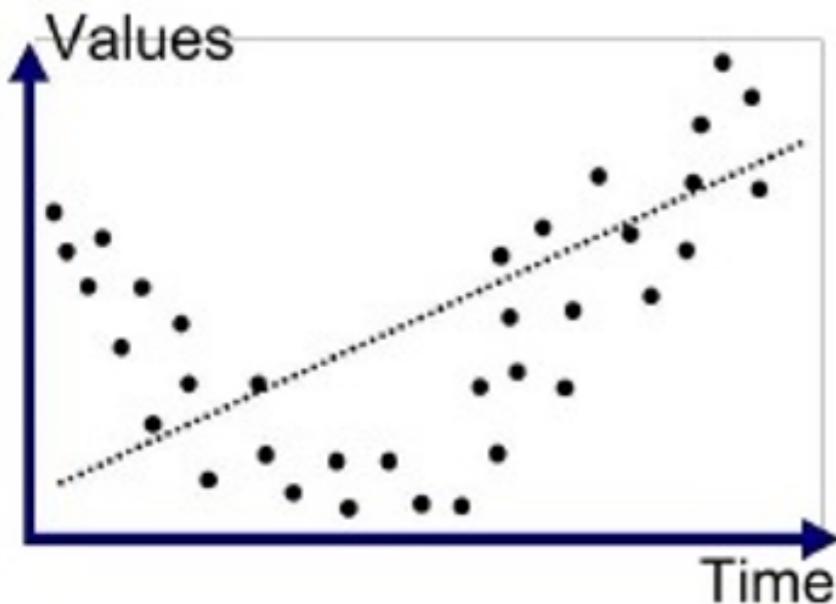


**Обобщающая способность  
Полинома 11-ой степени  
отсутствует.**

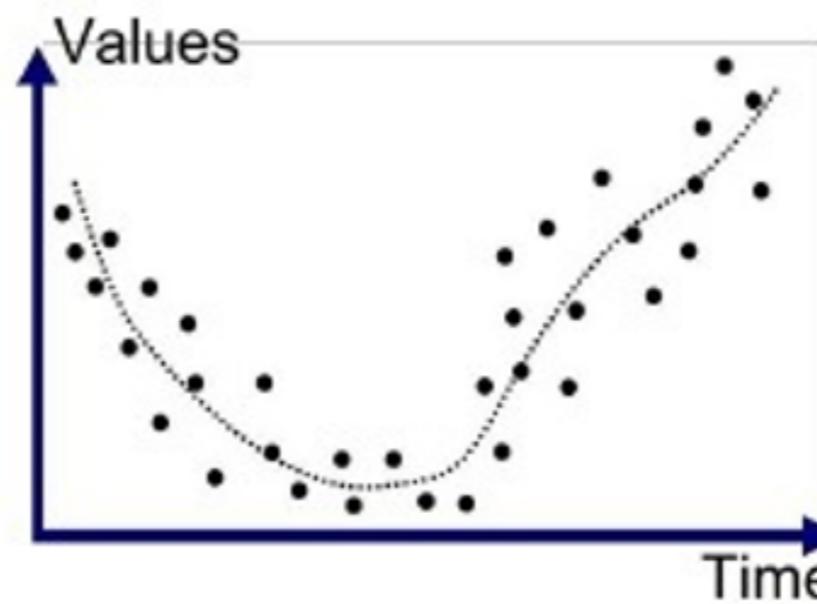
# Мотивация валидации



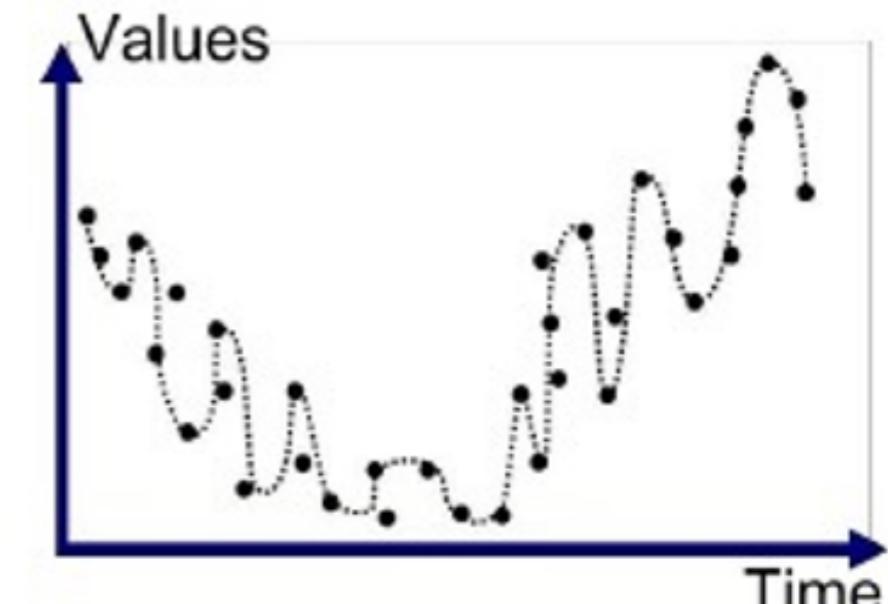
# Мотивация валидации



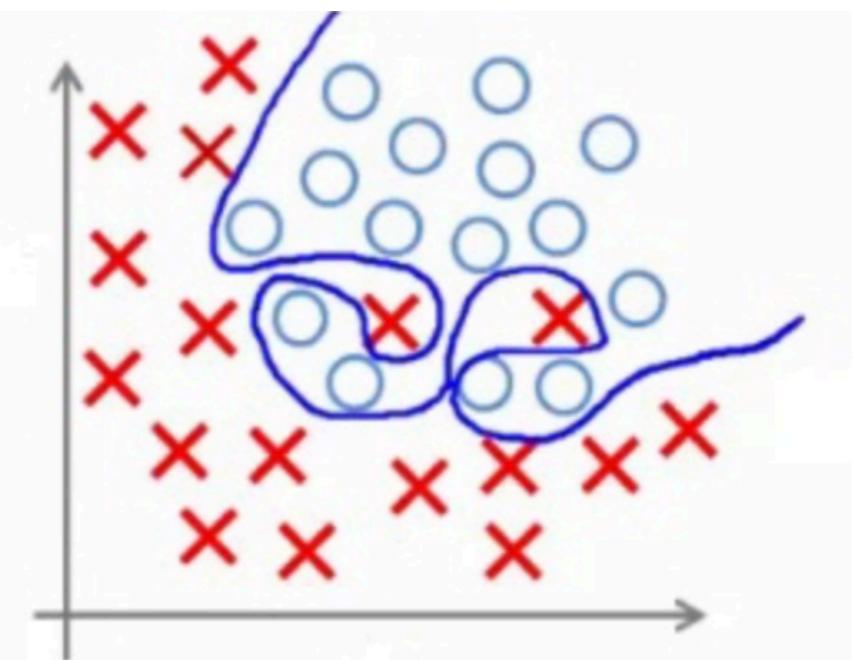
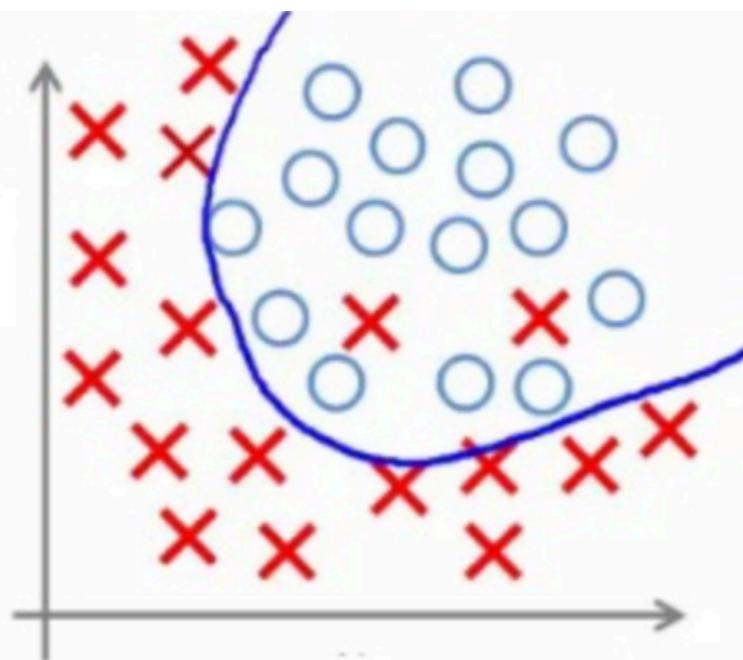
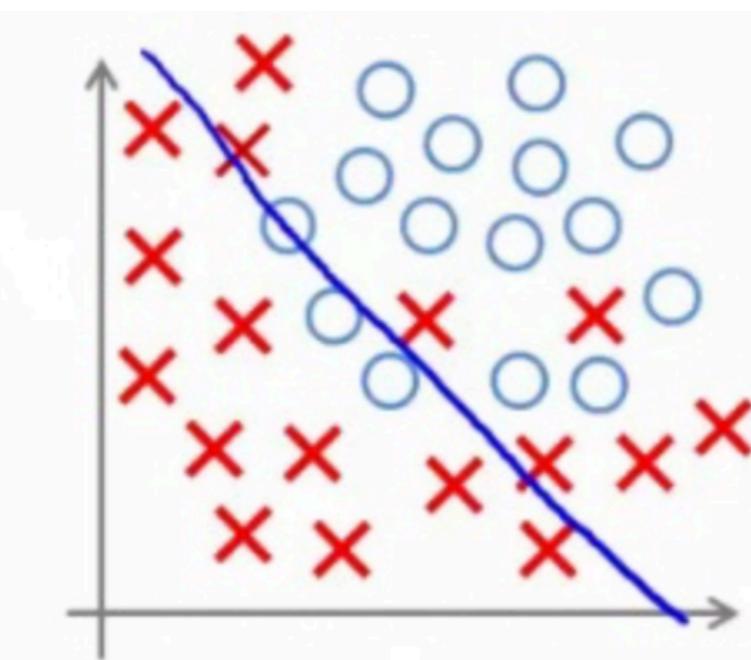
Underfitted



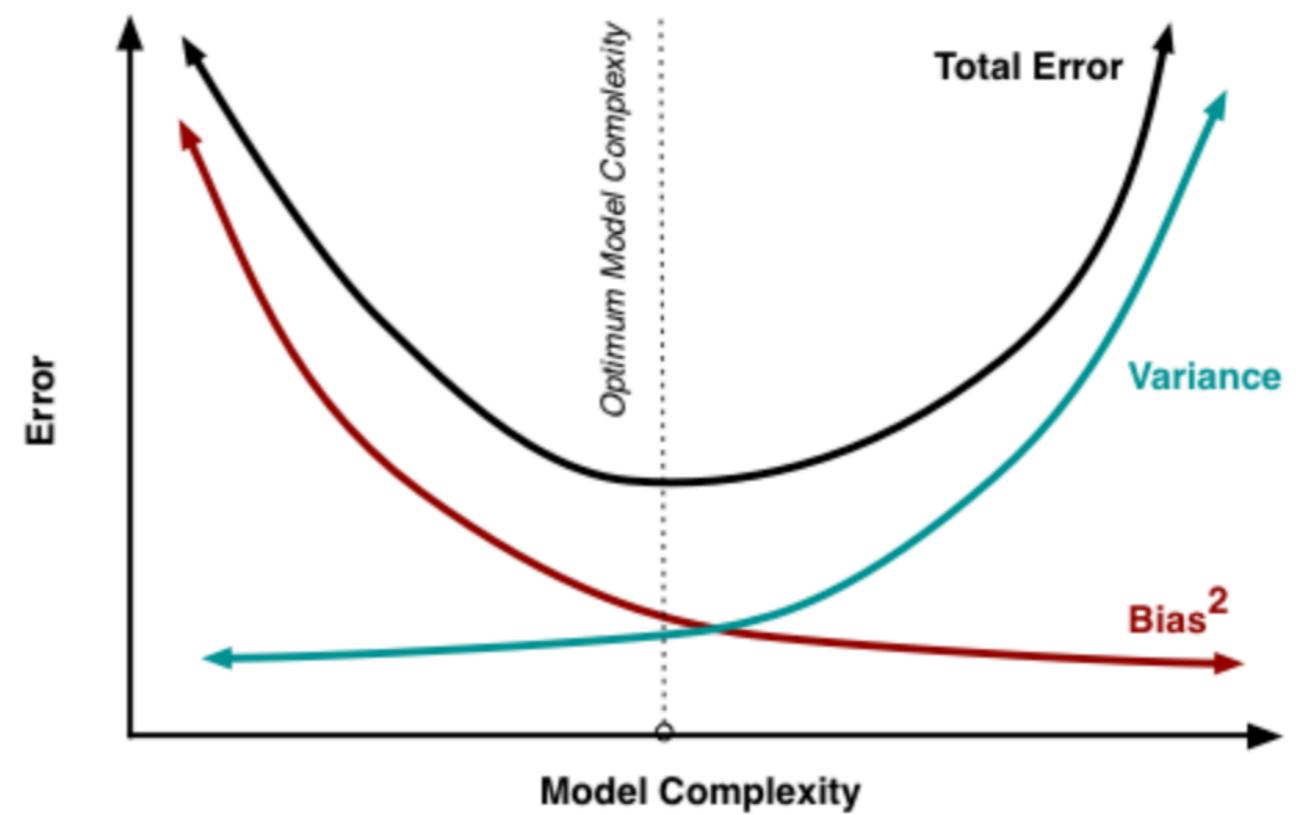
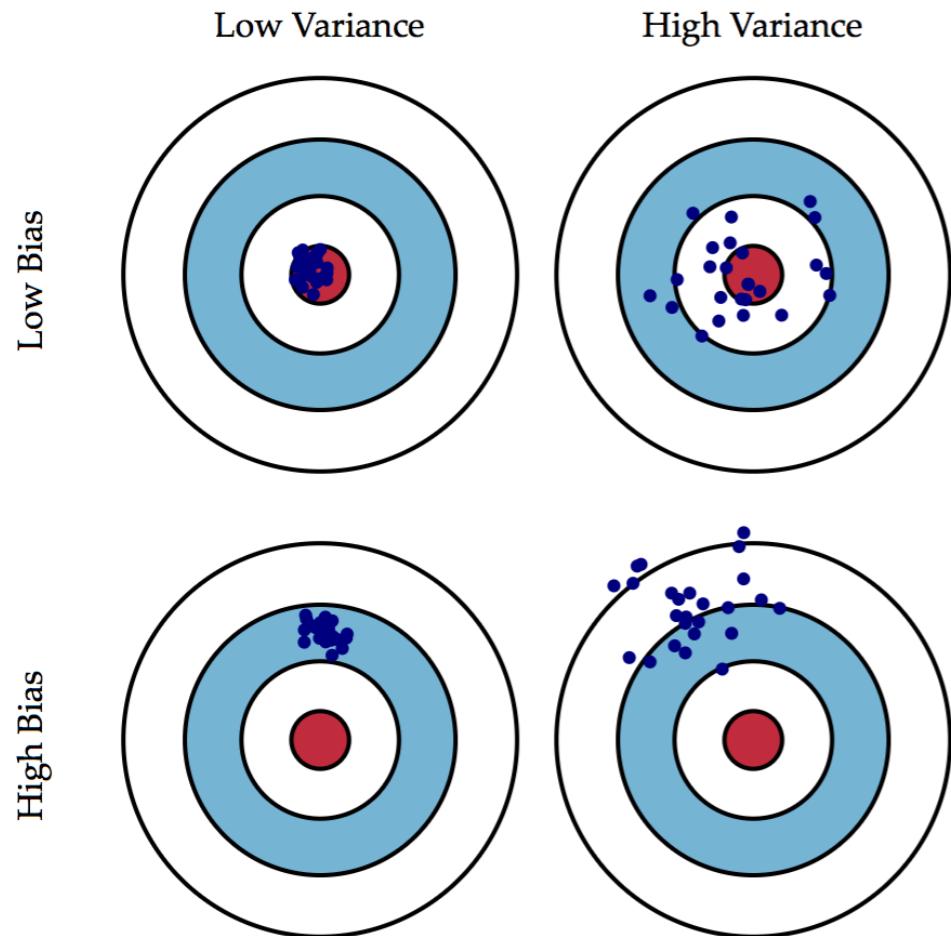
Good Fit/R robust



Overfitted



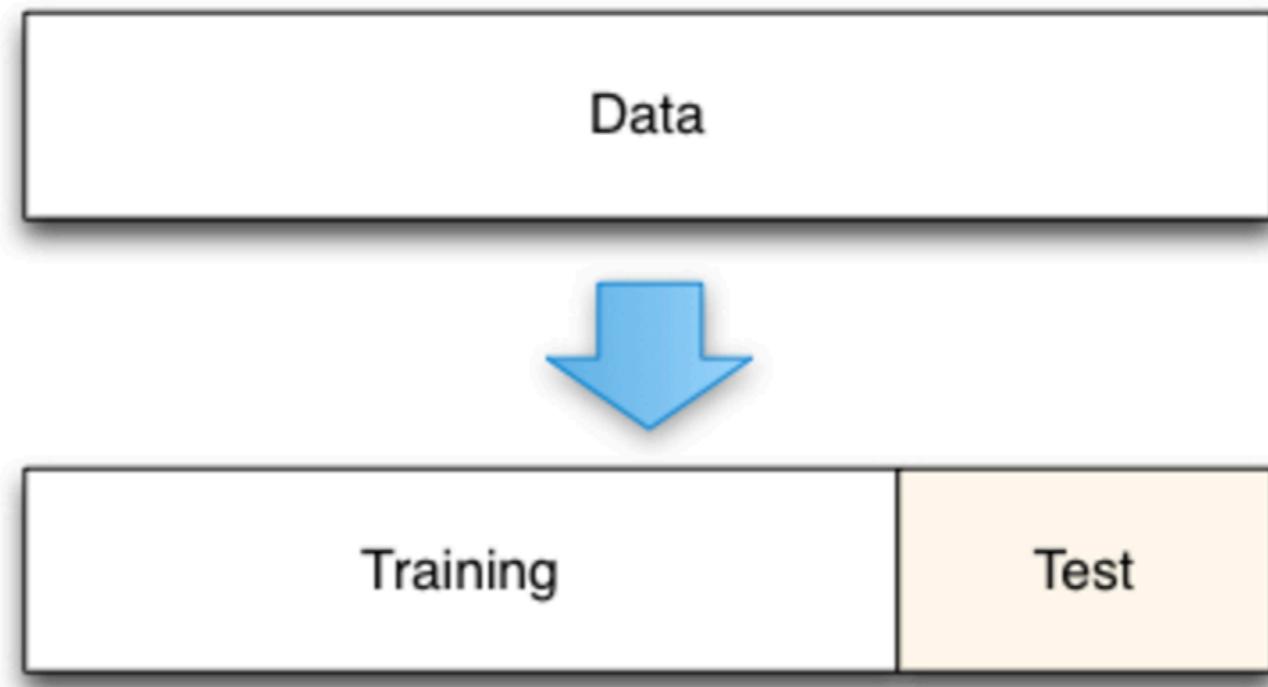
# Bias and Variance tradeoff



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

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

# Стратегии валидации



# Стратегии валидации

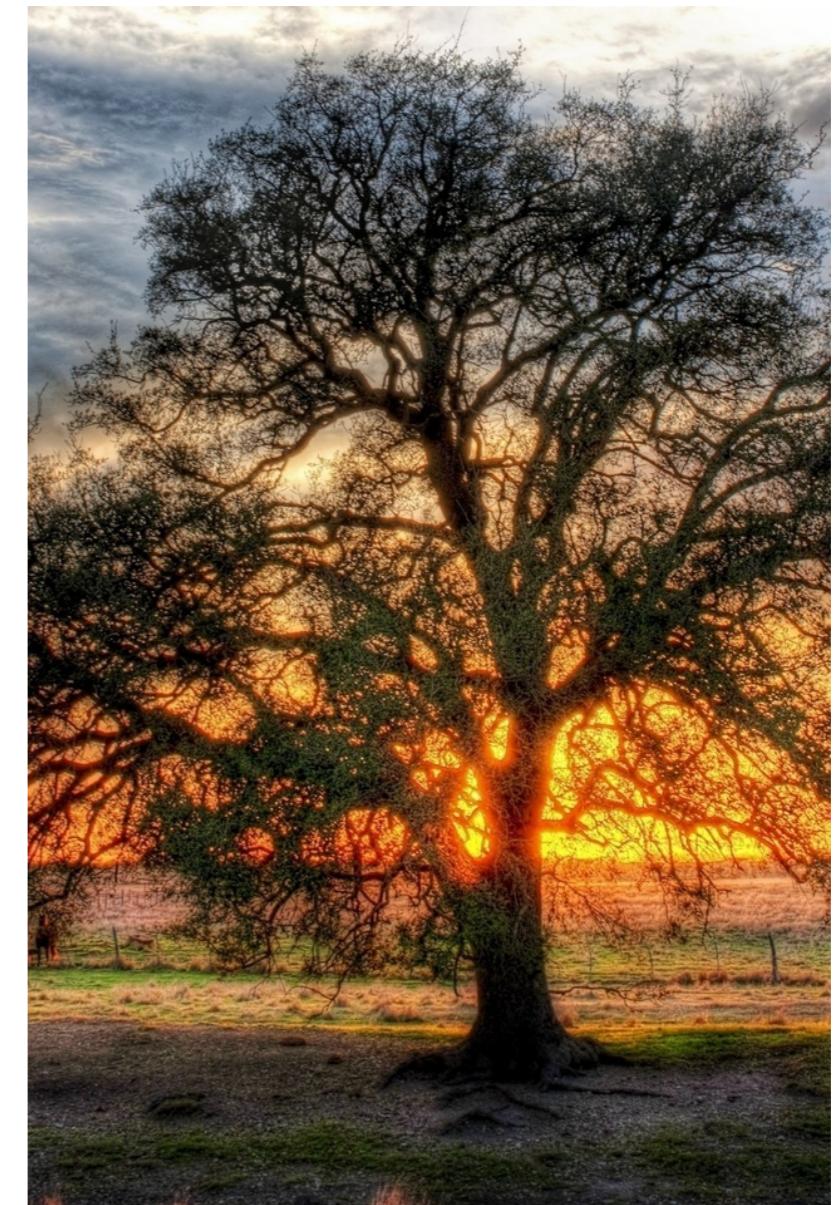
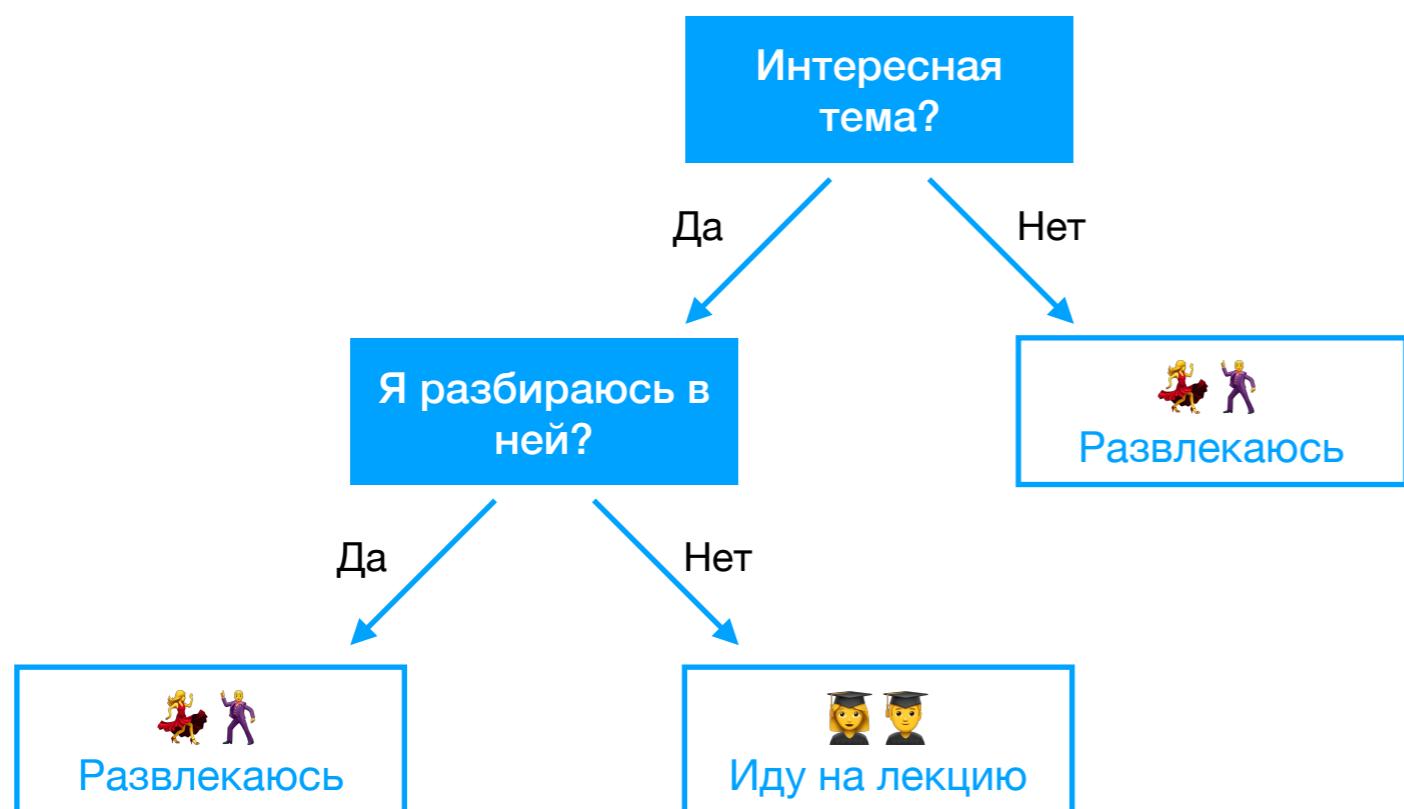


**Рассмотрим пример на практике**

# Деревья принятия решений

Логический алгоритм классификации, основанный на поиске конъюнктивных закономерностей.

Пойду ли я на МО сегодня?



# Деревья принятия решений

Дерево решений служит обобщением опыта экспертов, средством передачи знаний будущим сотрудникам или моделью бизнес-процесса компании.

Решение о выдаче кредита заемщику принималось на основе некоторых интуитивно (или по опыту) выведенных правил, которые можно представить в виде дерева решений.

## Пример: Кредитный scoring



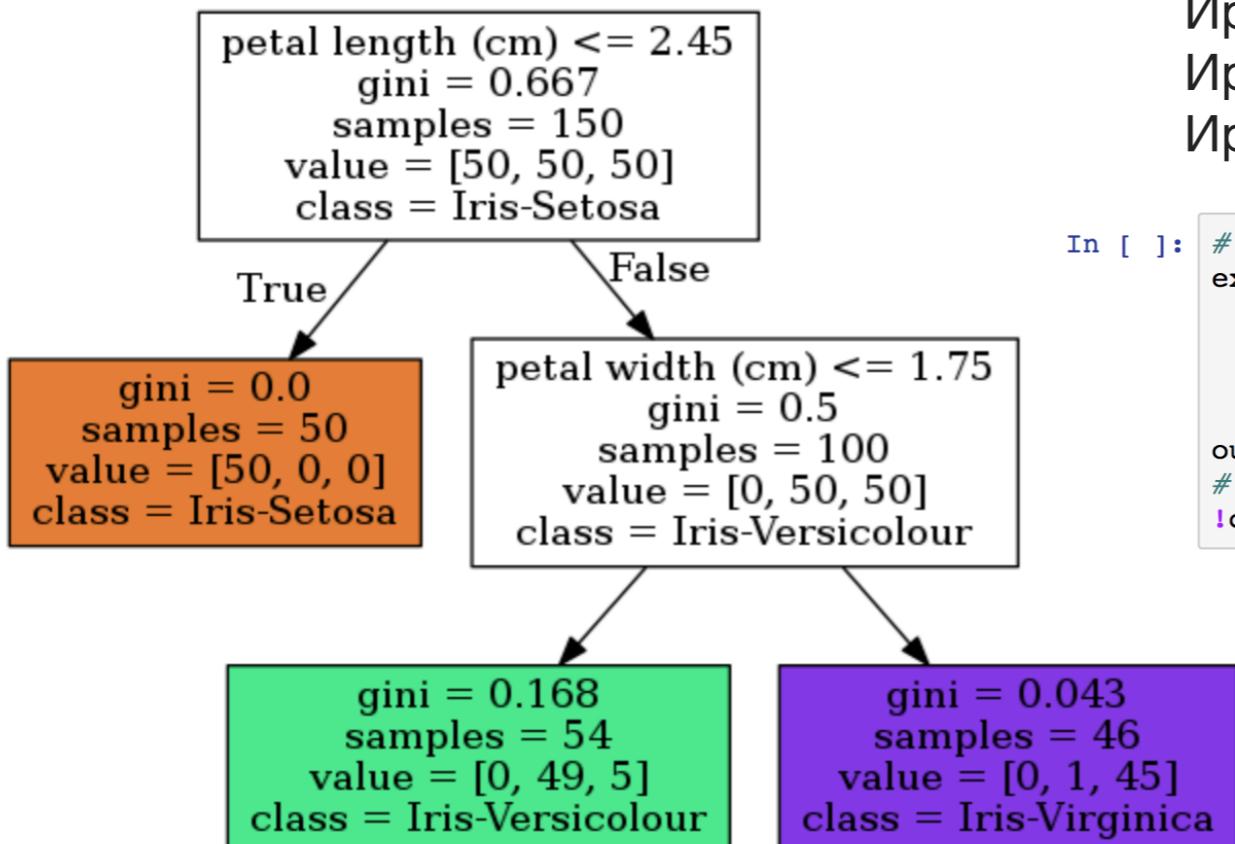
# Деревья принятия решений



`sklearn.datasets.load_iris`

`sklearn.tree.DecisionTreeClassifier`

`sklearn.tree.export_graphviz`



## Классы:

Ирис щетинистый (Iris setosa)  
Ирис виргинский (Iris virginica)  
Ирис разноцветный (Iris versicolor)

```
In [ ]: # Отрисуем дерево
export_graphviz(tree, feature_names=load_iris()['feature_names'],
                class_names=['Iris-Setosa',
                            'Iris-Versicolour',
                            'Iris-Virginica'],
                out_file='iris_tree.dot', filled=True)
# для этого понадобится библиотека pydot (pip install pydot)
!dot -Tpng 'iris_tree.dot' -o 'iris_tree.png'
```

## Признаки:

длина чашелистика (см)  
ширина чашелистика (см)  
длина лепестка (см)  
ширина лепестка (см)

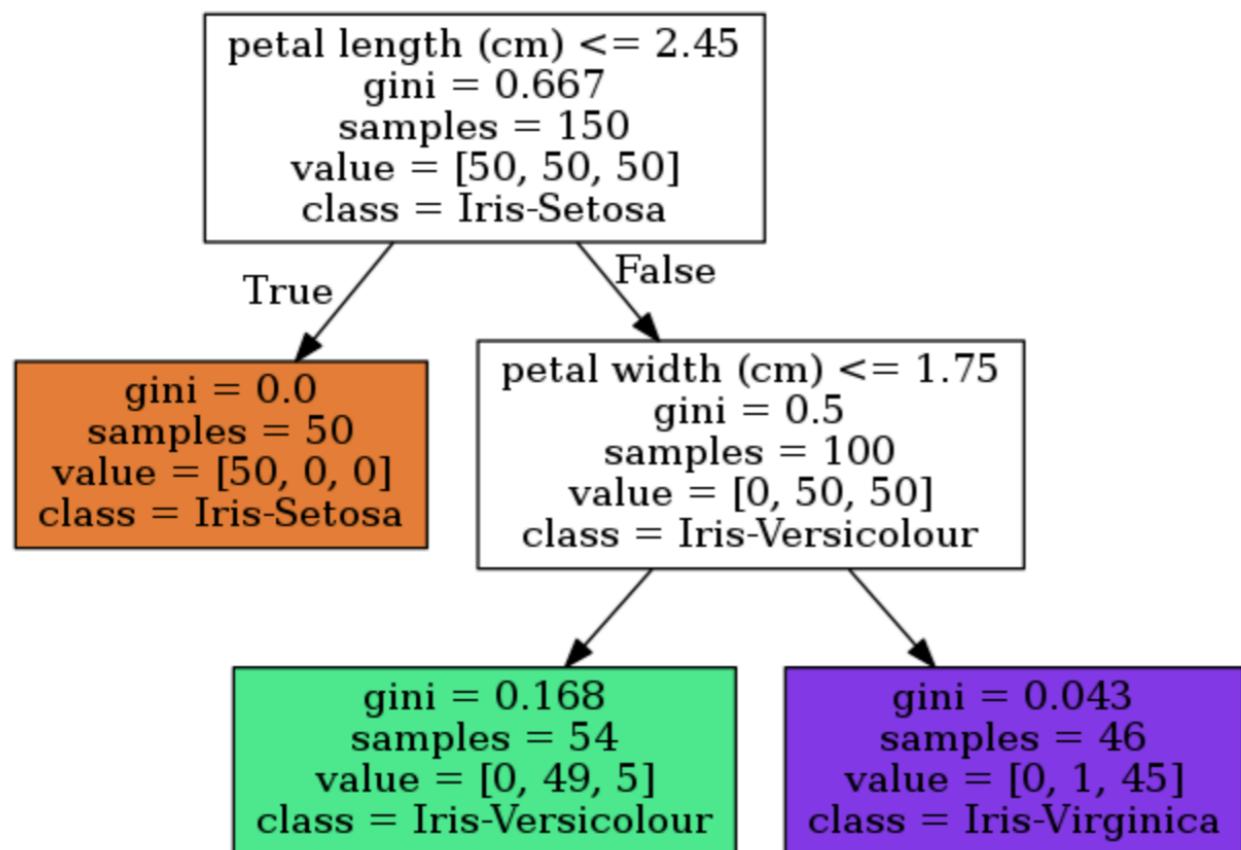
# Деревья принятия решений



`sklearn.datasets.load_iris`

`sklearn.tree.DecisionTreeClassifier`

`sklearn.tree.export_graphviz`



Неопределенность Джини (Gini impurity):

$$G_i = 1 - \sum_{k=1}^n (p_{ik})^2$$

$$G_{split} = \frac{L}{N} \times G_L + \frac{R}{N} \times G_R \rightarrow \min$$

L - Количество элементов в левой ветке

R - Количество элементов в правой ветке

N - Количество элементов в узле

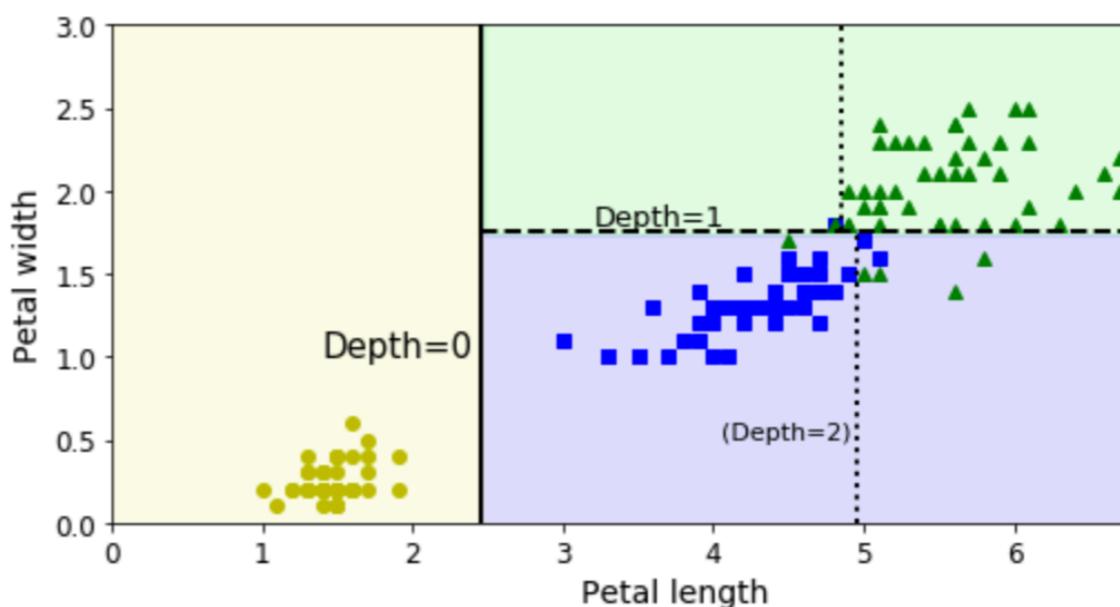
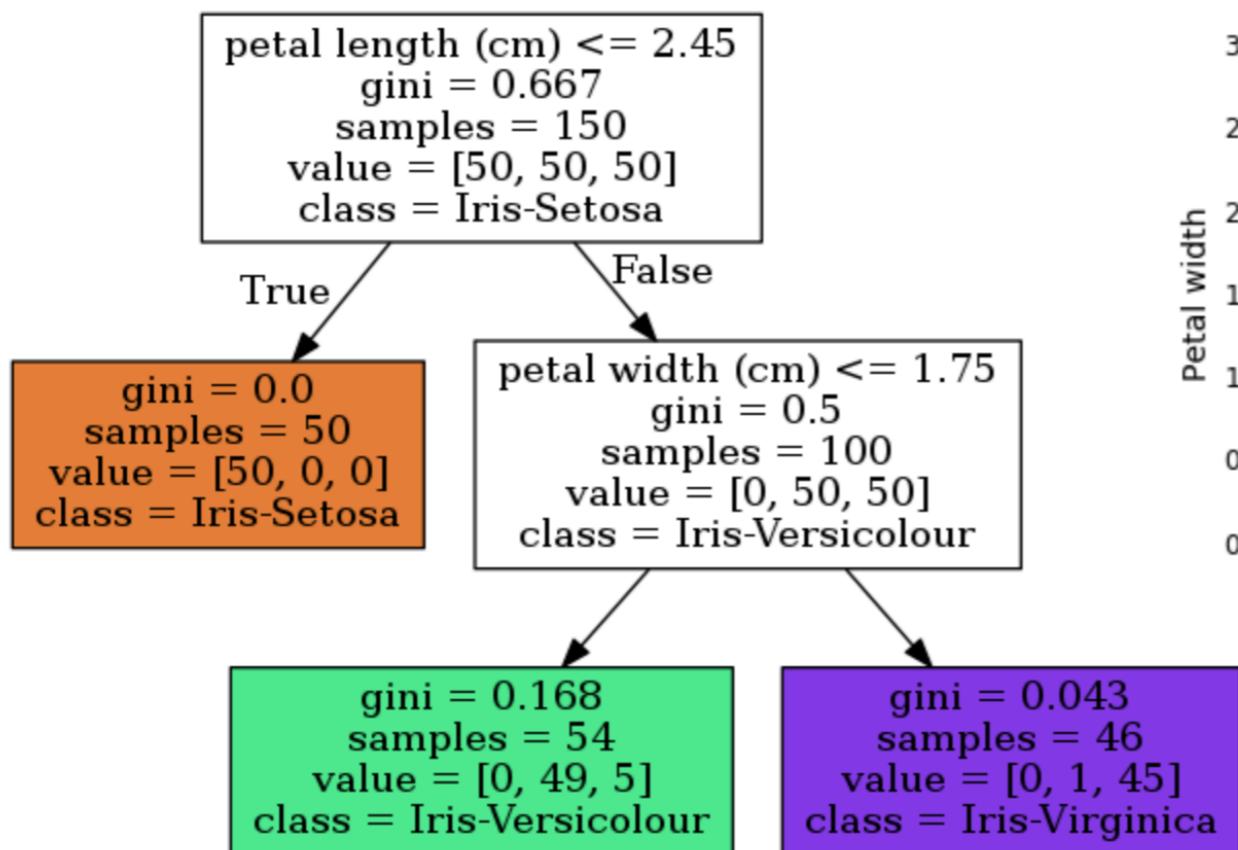
# Деревья принятия решений



`sklearn.datasets.load_iris`

`sklearn.tree.DecisionTreeClassifier`

`sklearn.tree.export_graphviz`



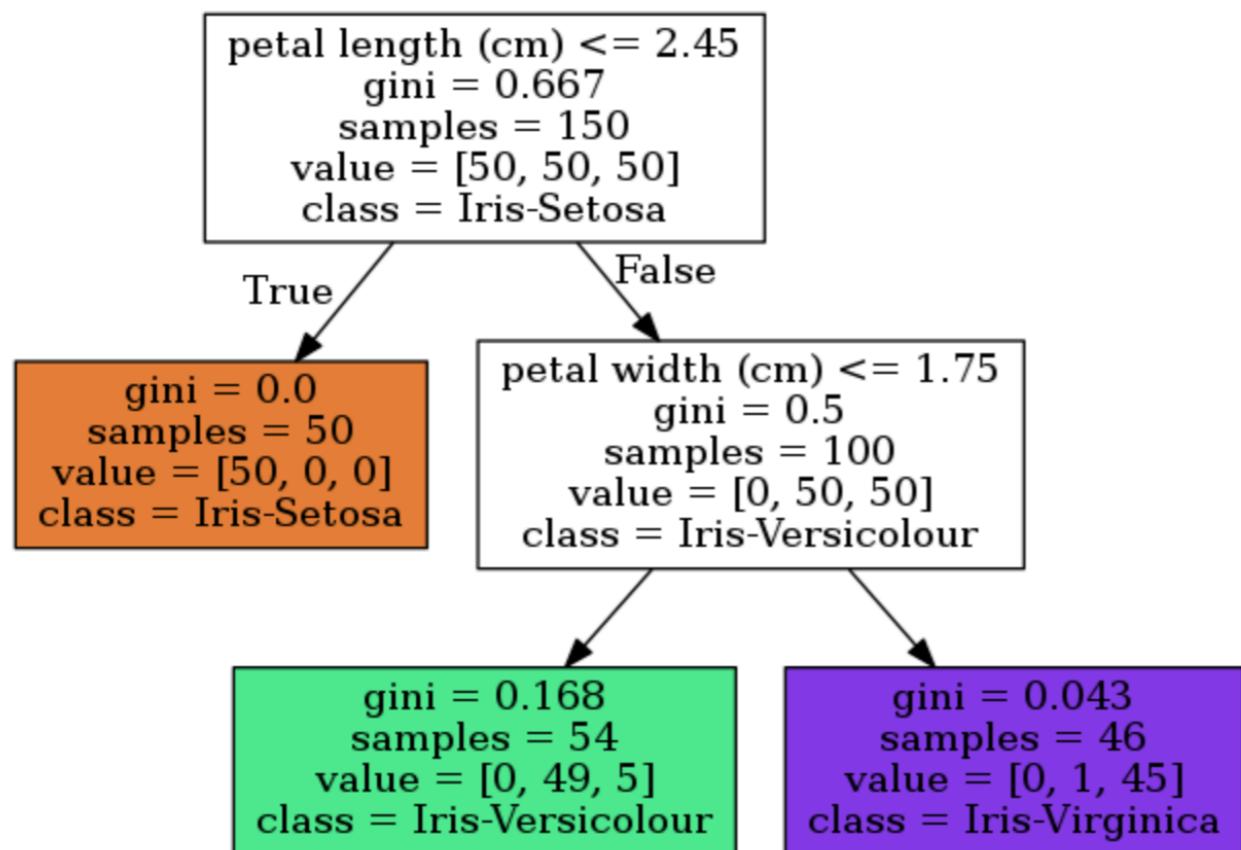
# Деревья принятия решений



`sklearn.datasets.load_iris`

`sklearn.tree.DecisionTreeClassifier`

`sklearn.tree.export_graphviz`



Энтропия Шеннона:

$$S_i = - \sum_{k=1}^n p_{ik} \log p_{ik}$$

$p_{ik}$  - Вероятность нахождения в состоянии k

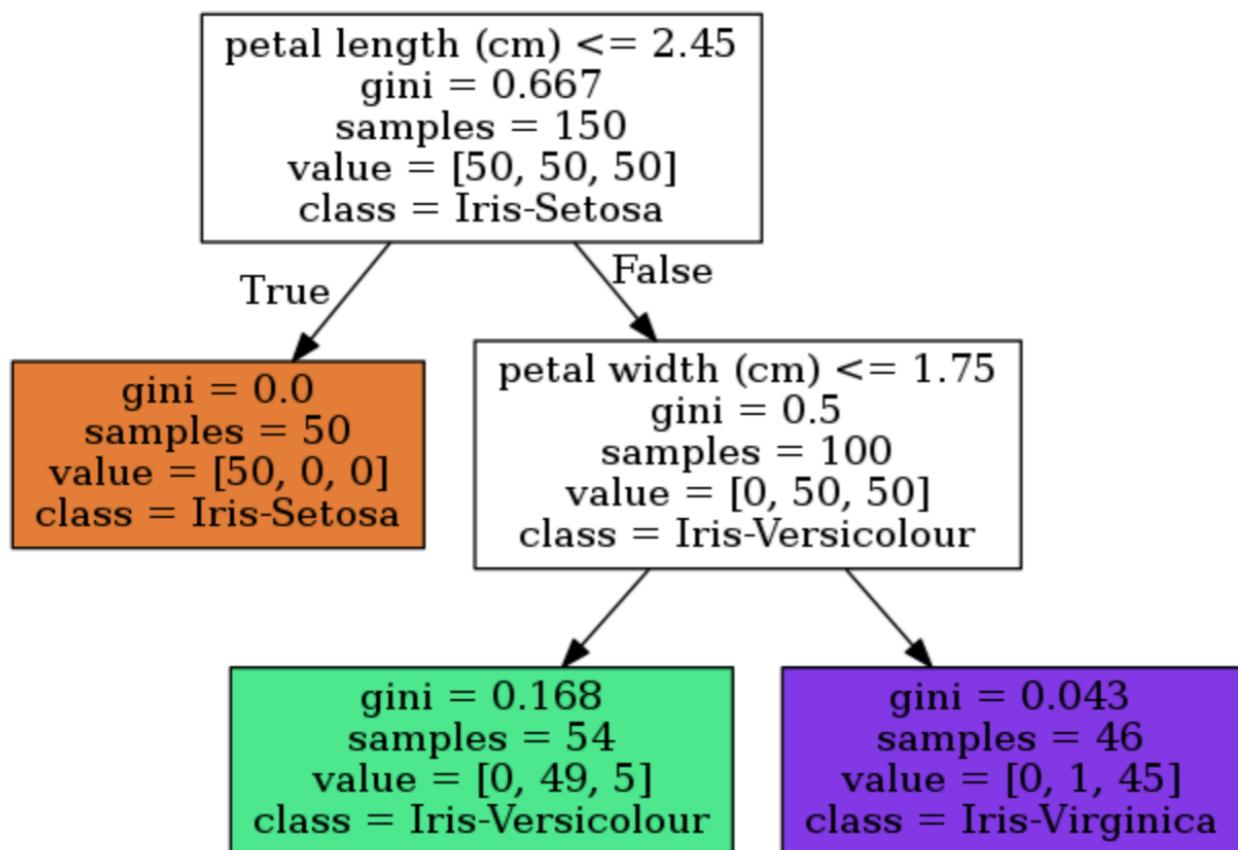
# Деревья принятия решений



`sklearn.datasets.load_iris`

`sklearn.tree.DecisionTreeClassifier`

`sklearn.tree.export_graphviz`



`tree.predict_proba( [2,3,3,1] )`

Классы:

Ирис щетинистый (Iris setosa) - 0

Ирис виргинский (Iris virginica) - 0,907

Ирис разноцветный (Iris versicolor - 0,093)

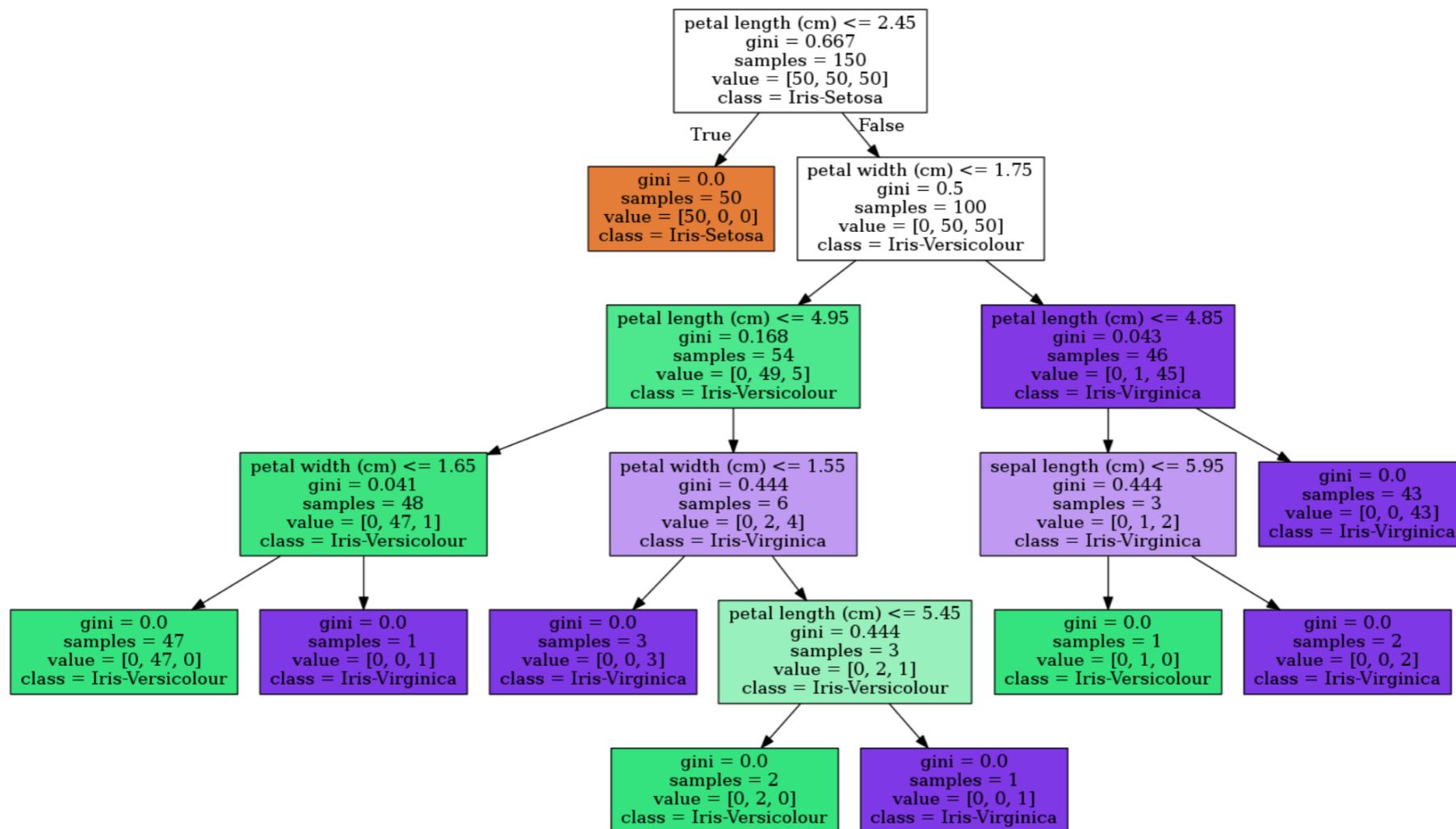
$$p_1 = \frac{0}{54} \quad p_2 = \frac{49}{54} \quad p_3 = \frac{5}{54}$$

# Деревья принятия решений



## sklearn.tree.DecisionTreeClassifier

(criterion='gini', splitter='best', max\_depth=None, min\_samples\_split=2, min\_samples\_leaf=1, min\_weight\_fraction\_leaf=0.0, max\_features=None, random\_state=None, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, class\_weight=None, presort=False)

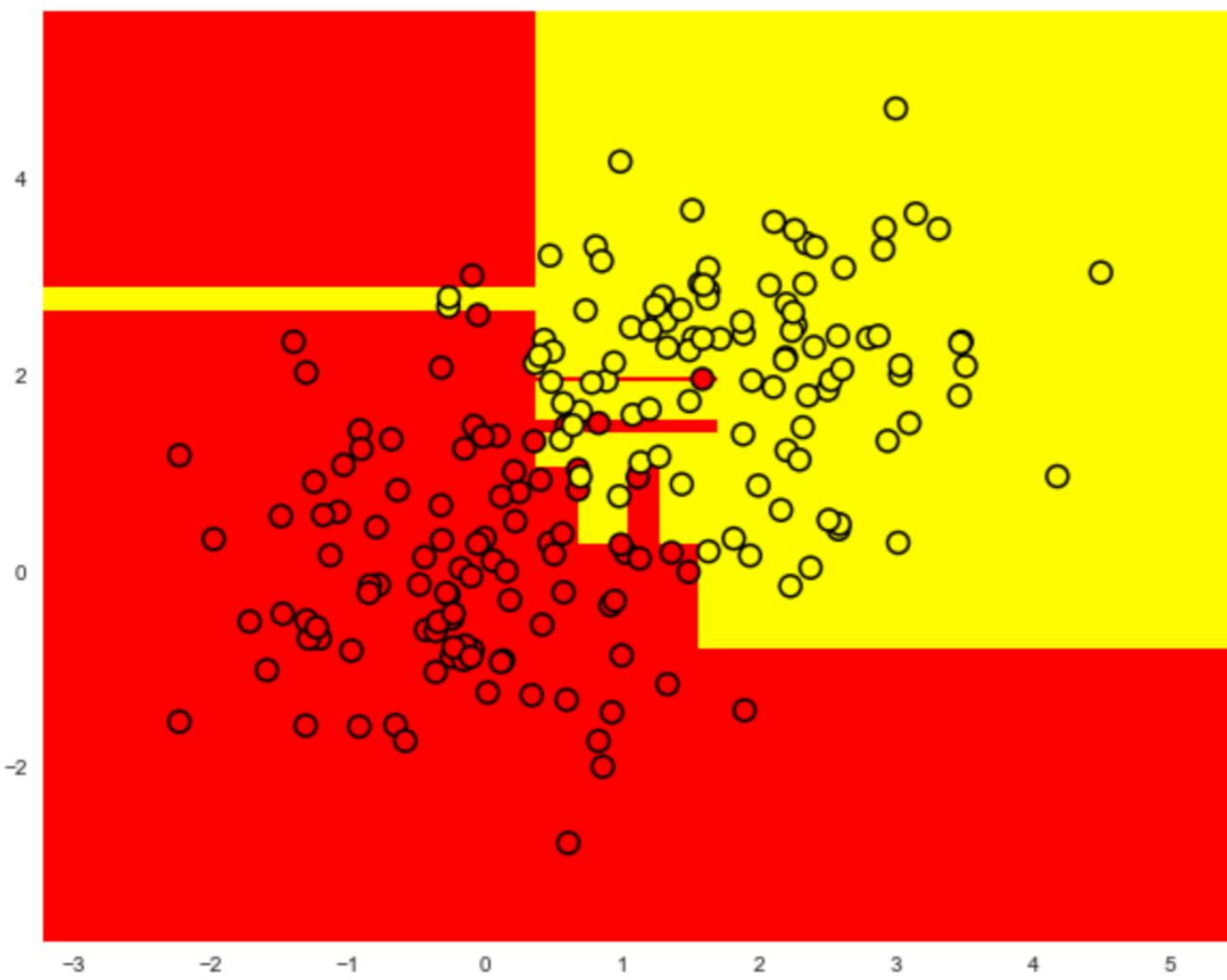


# Регуляризация деревьев



`sklearn.tree.DecisionTreeClassifier`

(`criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort=False`)



`max_depth` - глубина дерева

`min_samples_split` - Минимальное количество объектов, прежде чем можно сделать разделение

`min_samples_leaf` - Минимальное кол-во объектов в листовом узле

`max_leaf_nodes` - Максимальное количество листовых узлов

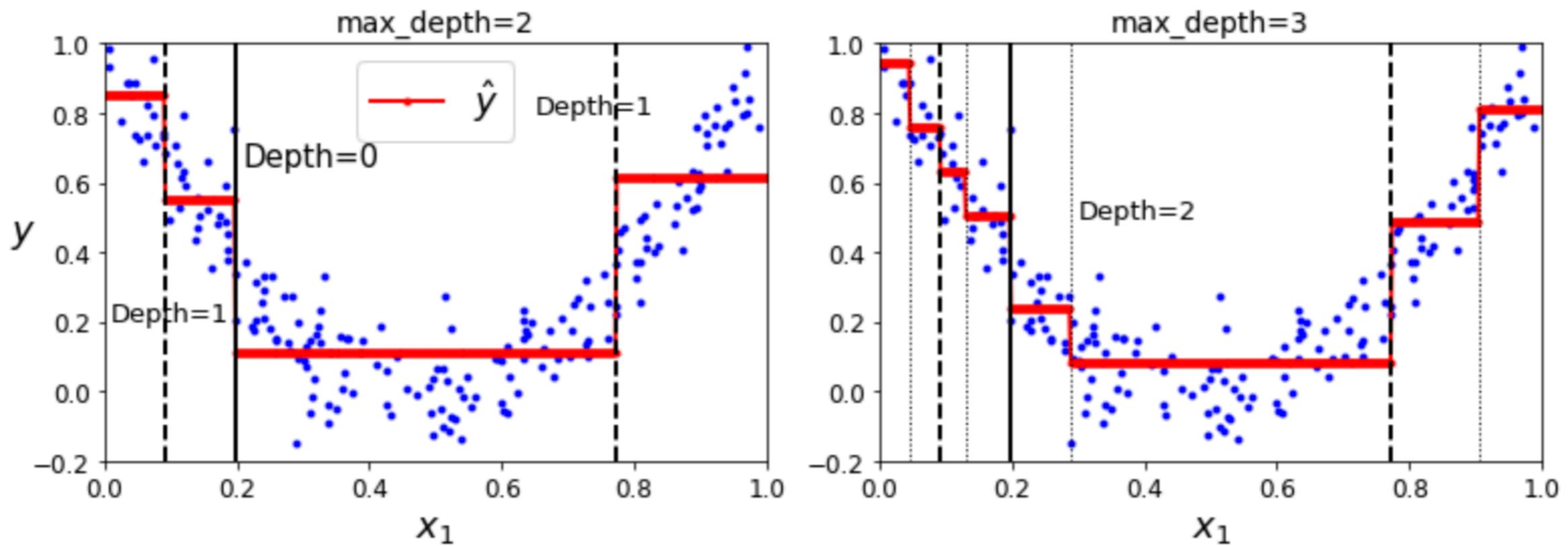
Семинар Евгения Соколова

# Деревья решений для задачи регрессии



`sklearn.tree.DecisionTreeRegressor`

(`criterion='mse'`, `splitter='best'`, `max_depth=None`, `min_samples_split=2`,  
`min_samples_leaf=1`, `min_weight_fraction_leaf=0.0`, `max_features=None`,  
`random_state=None`, `max_leaf_nodes=None`, `min_impurity_decrease=0.0`,  
`min_impurity_split=None`, `presort=False`)

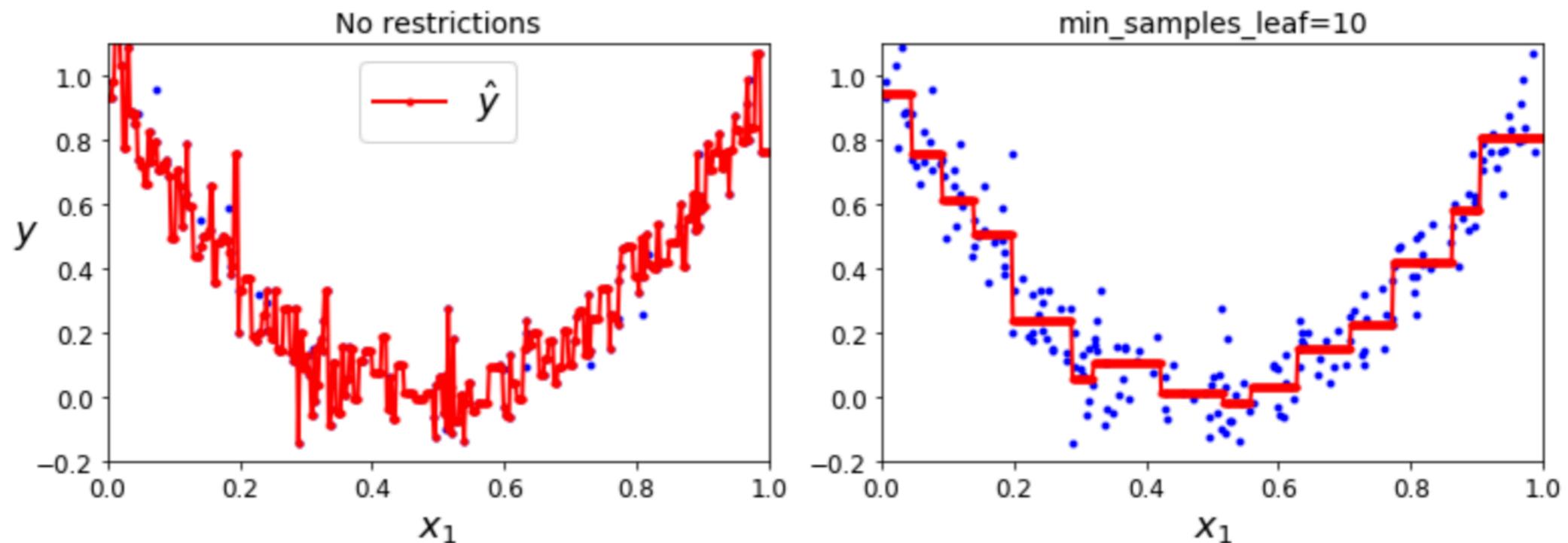


# Деревья решений для задачи регрессии



`sklearn.tree.DecisionTreeRegressor`

(`criterion='mse'`, `splitter='best'`, `max_depth=None`, `min_samples_split=2`,  
`min_samples_leaf=1`, `min_weight_fraction_leaf=0.0`, `max_features=None`,  
`random_state=None`, `max_leaf_nodes=None`, `min_impurity_decrease=0.0`,  
`min_impurity_split=None`, `presort=False`)



# Какой ответ деревьев в регрессии?

Какая стратегия поведения в листьях регрессионного дерева приводит к меньшему матожиданию ошибки по MSE: отвечать средним значением таргета на объектах обучающей выборки, попавших в лист, или отвечать таргетом для случайного объекта из листа (считая все объекты равновероятными)?

- $\hat{y} = \frac{1}{n} \sum_{i=1}^n c_i$

$$\mathbb{E}(y - \frac{1}{n} \sum_{i=1}^n c_i)^2 = \mathbb{E}y^2 + \left( \frac{1}{n} \sum_{i=1}^n c_i \right)^2 - 2 \left( \frac{1}{n} \sum_{i=1}^n c_i \right) \mathbb{E}y$$

- $\hat{y} = X$ , где  $X \sim U(c)$

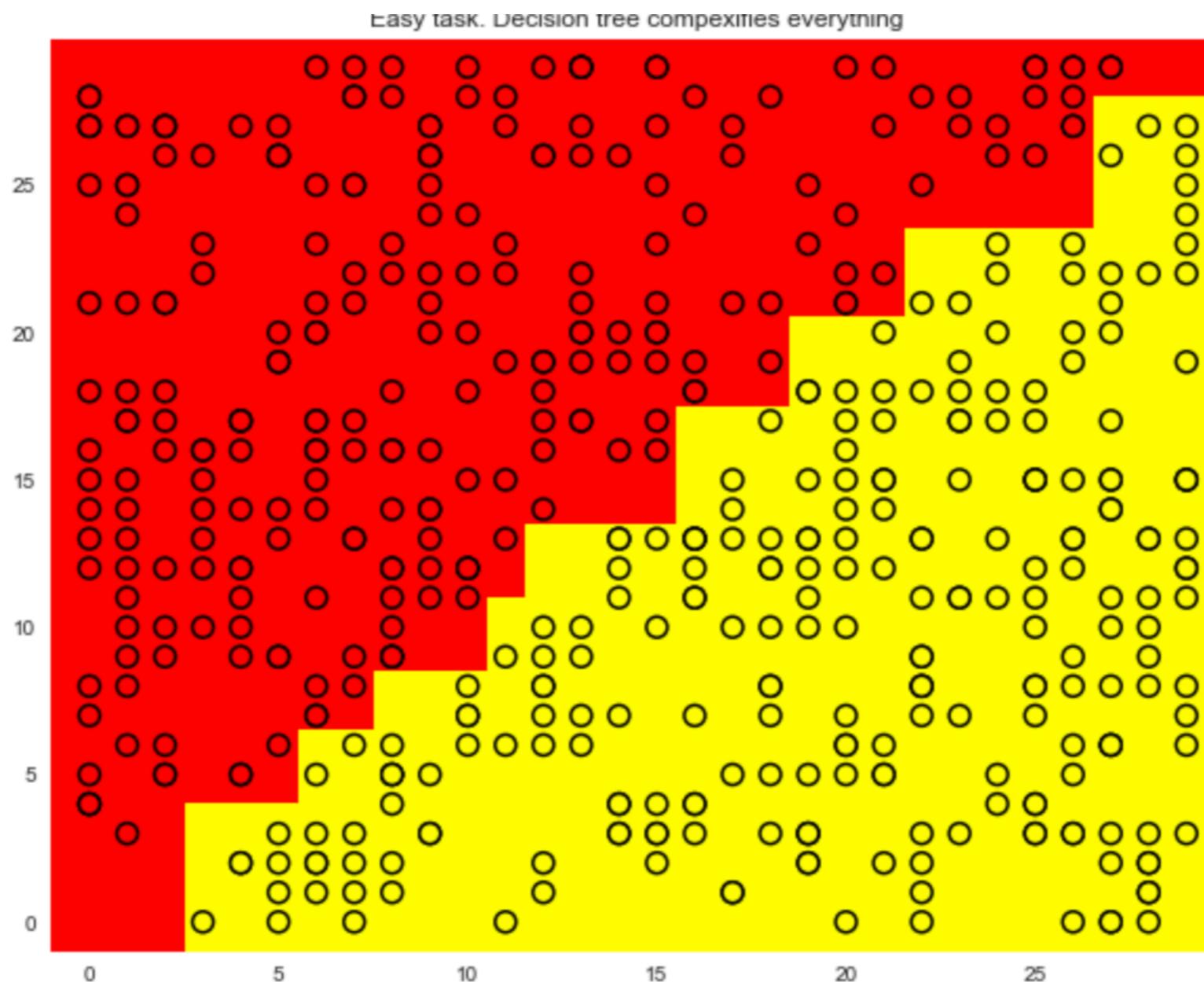
$$\mathbb{E} \frac{1}{n} \sum_{i=1}^n (y - c_i)^2 = \frac{1}{n} \sum_{i=1}^n \mathbb{E}(y - c_i)^2 = \mathbb{E}y^2 + \frac{1}{n} \sum_{i=1}^n c_i^2 - \frac{2}{n} \mathbb{E}y \sum_{i=1}^n c_i$$

Тогда выпишем их разность:

$$\mathbb{E} \frac{1}{n} \sum_{i=1}^n (y - c_i)^2 - \mathbb{E}(y - \bar{c})^2 = \frac{1}{n} \sum_{i=1}^n c_i^2 - \left( \frac{1}{n} \sum_{i=1}^n c_i \right)^2 \geq 0 \text{ (По неравенству Коши-Буняковского)}$$

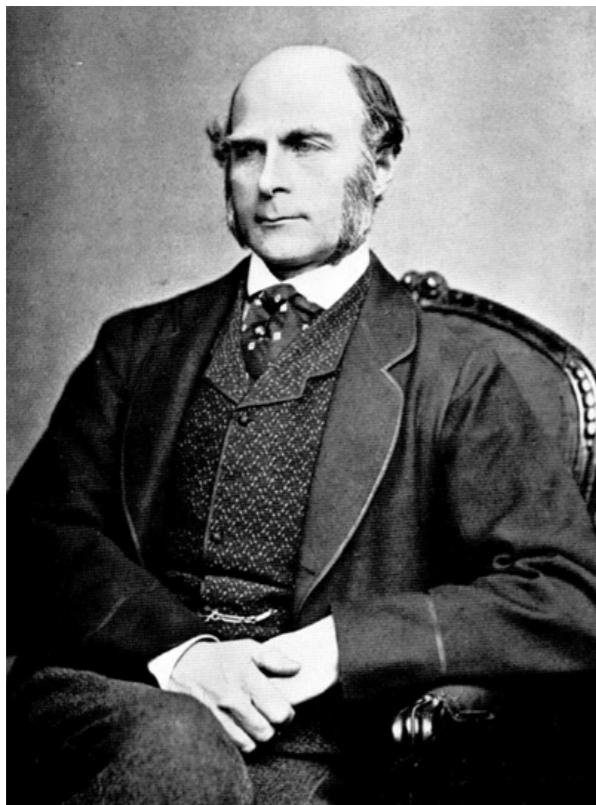
Получили, что мат. ожидание ошибки для первого поведения меньше, чем для второго.

# Сложные случаи для деревьев



**Рассмотрим пример на практике**

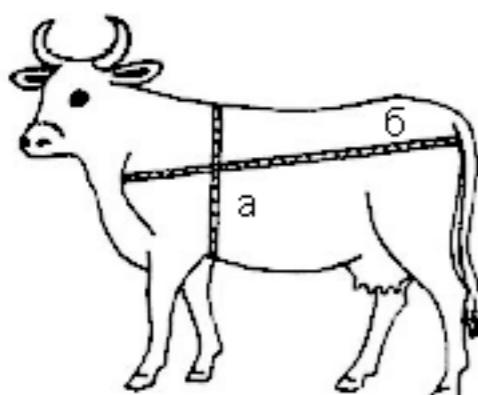
# Феномен: «Мудрость толпы»



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

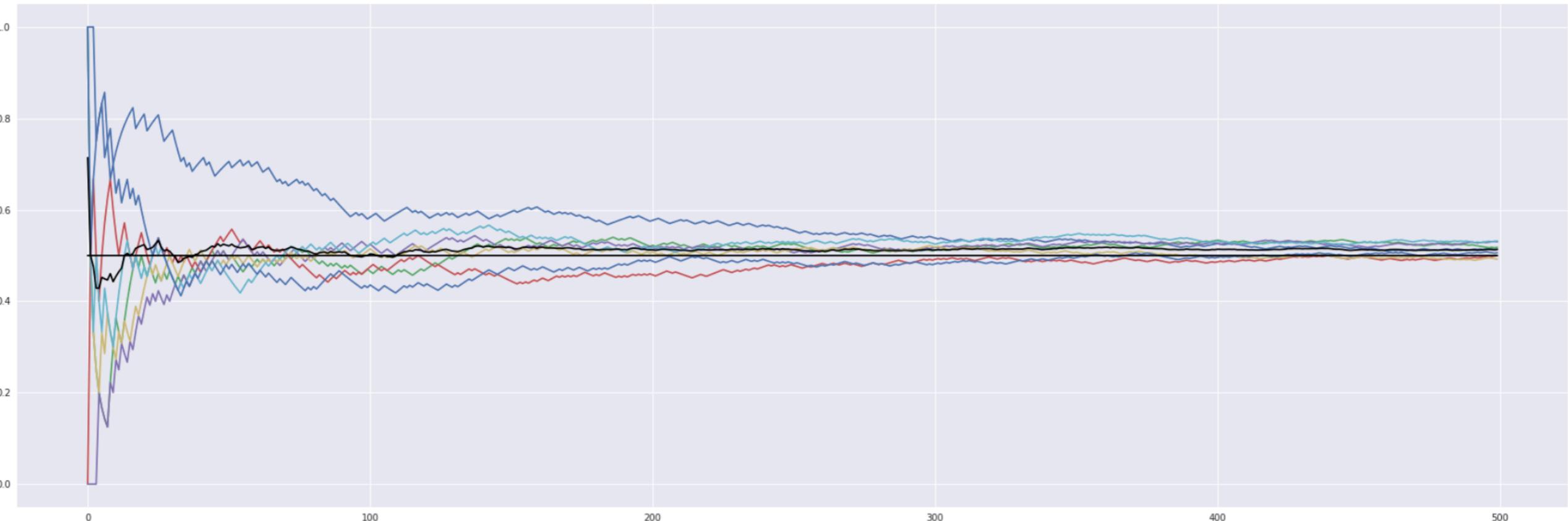
На выставке проводился конкурс, в рамках которого всем желающим предлагалось на глаз угадать точный вес забитого быка. Побеждал тот, кто называл самое близкое к истинному значение.

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»

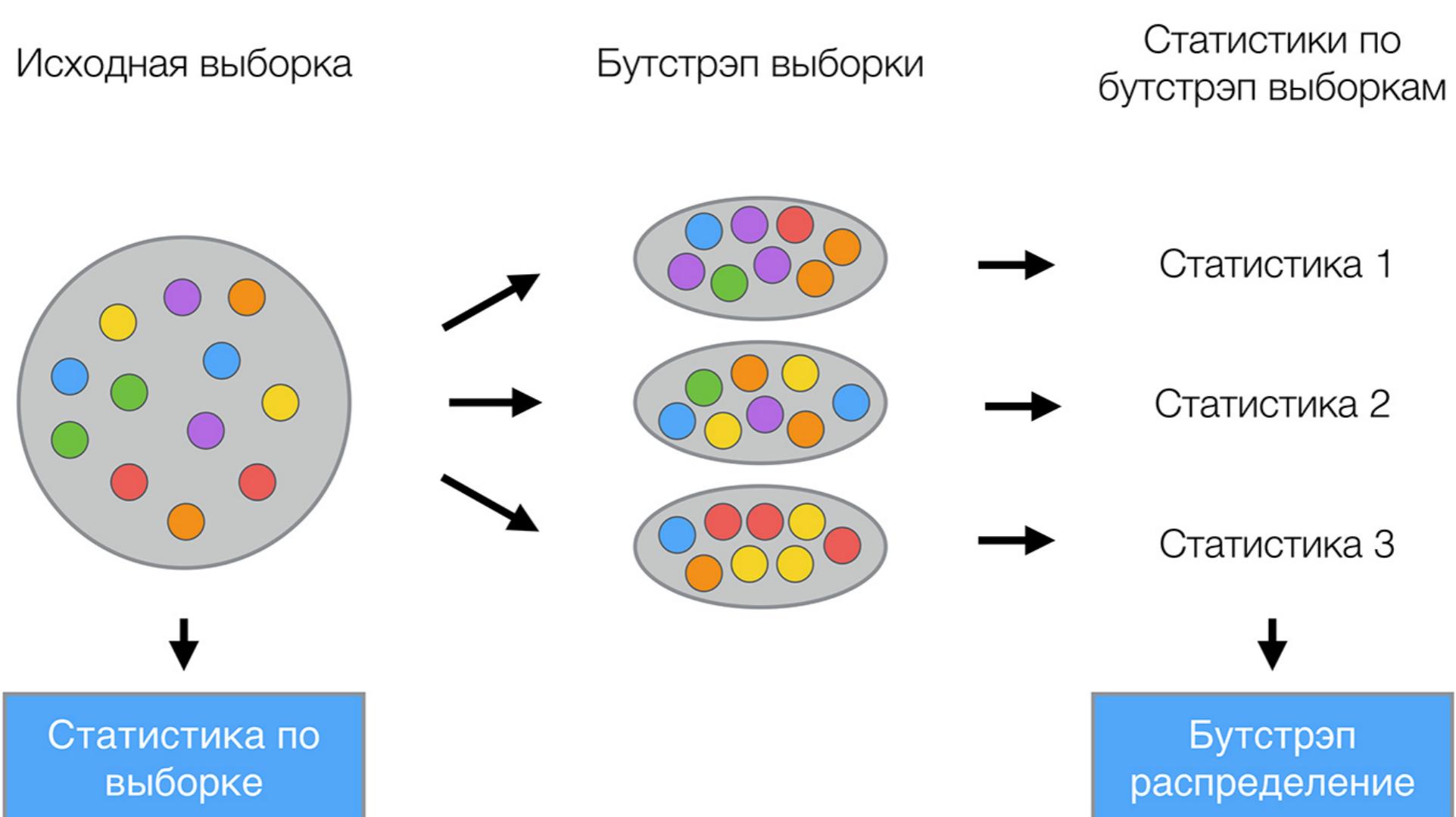


Если будем кидать монетку 500 раз, то  
соотношение «Орлов» и «Решек» с каждым  
броском будем стремиться к 1/2

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



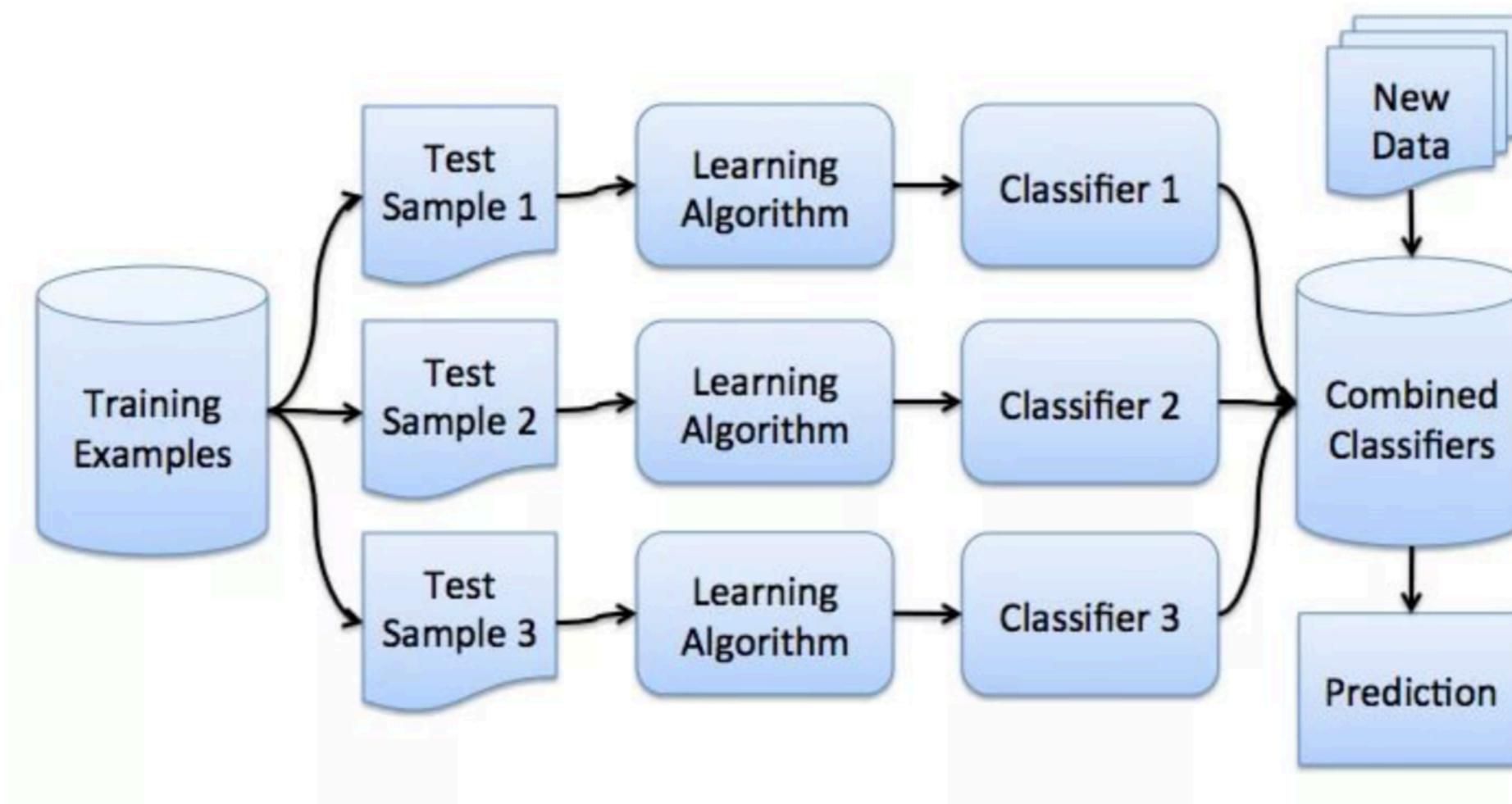
# Bootstrap



# Bootstrap

**Рассмотрим пример!**

# Bagging (Bootstrap aggregating)



# **Bagging (Bootstrap aggregating)**

**Рассмотрим пример!**

# 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

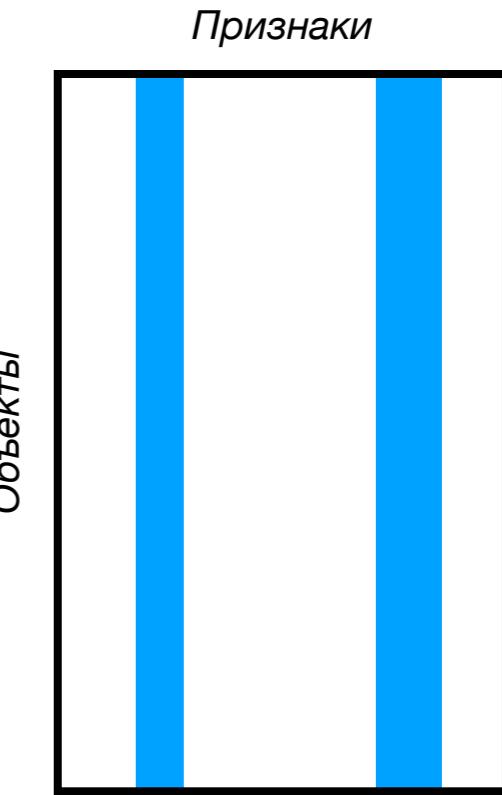
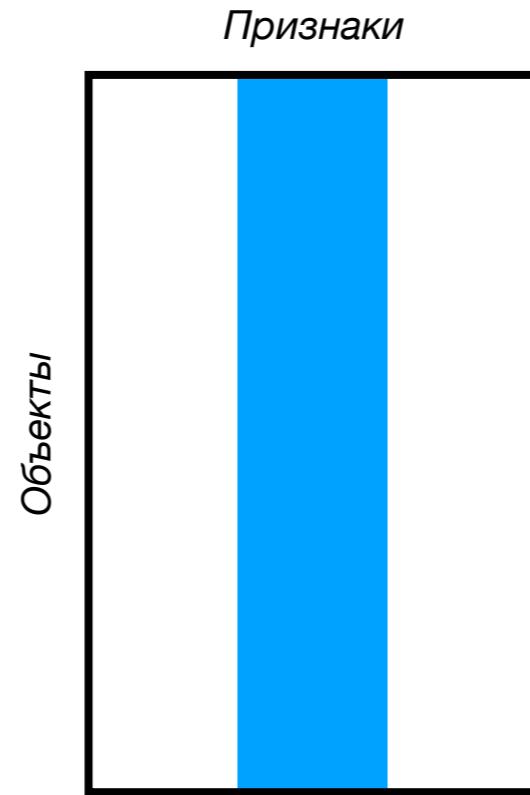
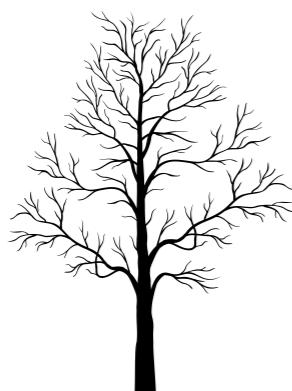
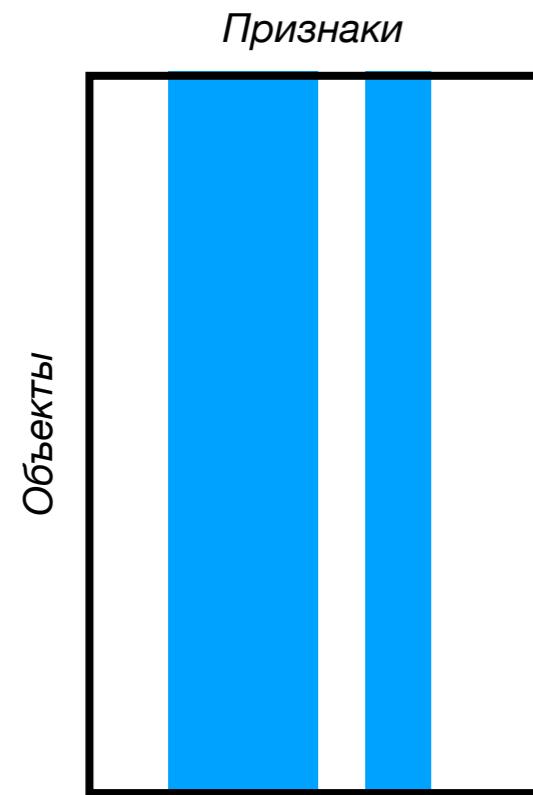
Если каждый член жюри присяжных имеет **независимое мнение**, и если вероятность правильного решения члена жюри больше 0.5, то тогда вероятность правильного решения присяжных в целом возрастает с увеличением количества членов жюри, и стремиться к единице. Если же вероятность быть правым у каждого из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

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

$$p > 0,5 \quad \mu > p$$

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

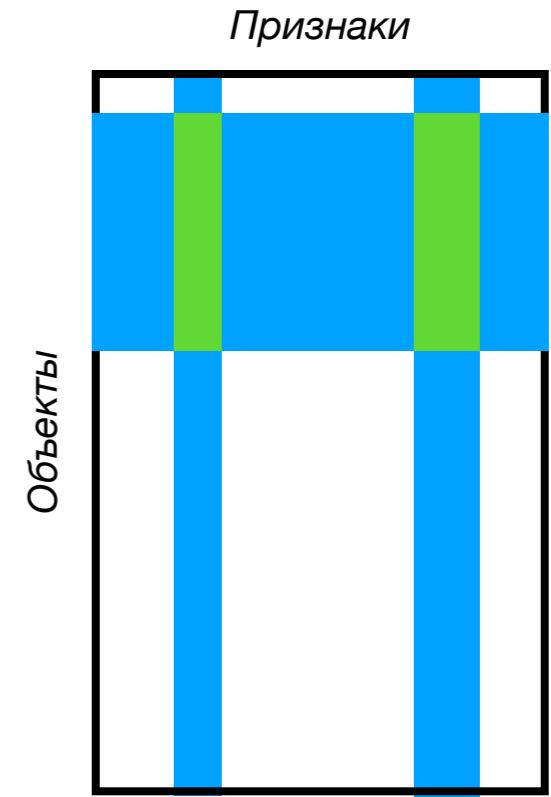
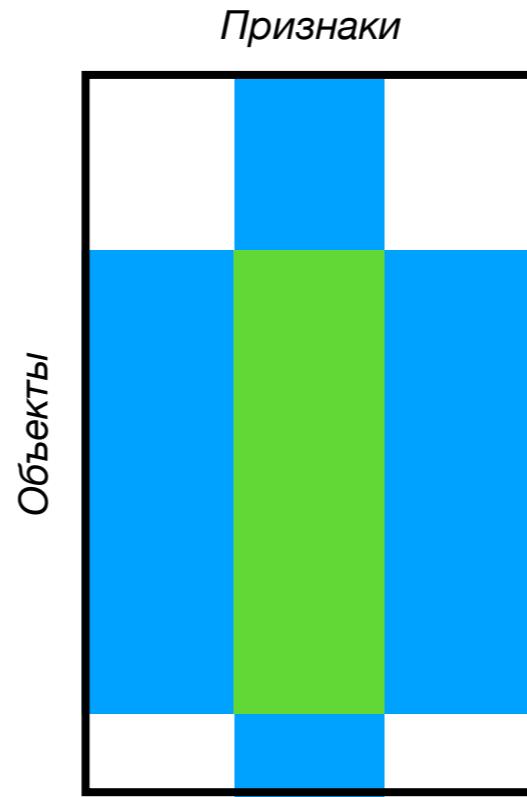
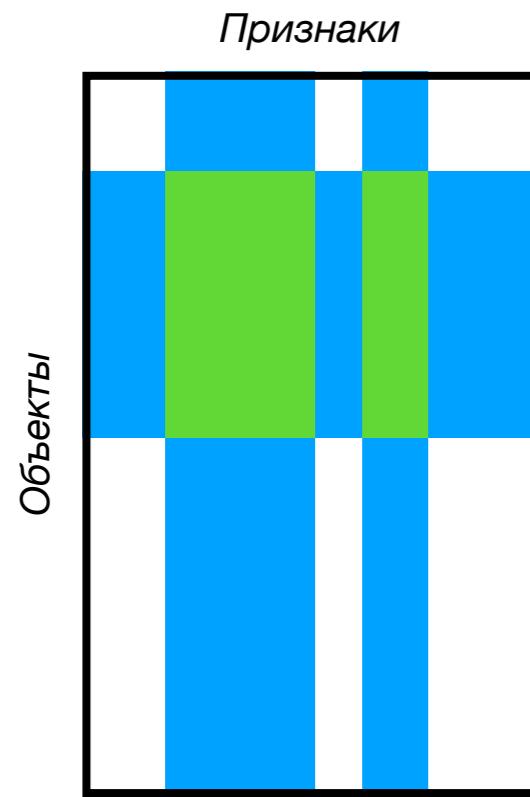
# Метод случайных подпространств



Для классификации:  $m = \sqrt{n}$

Для регрессии:  $m = \frac{n}{3}$

# 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:

$$m = \sqrt{n}$$

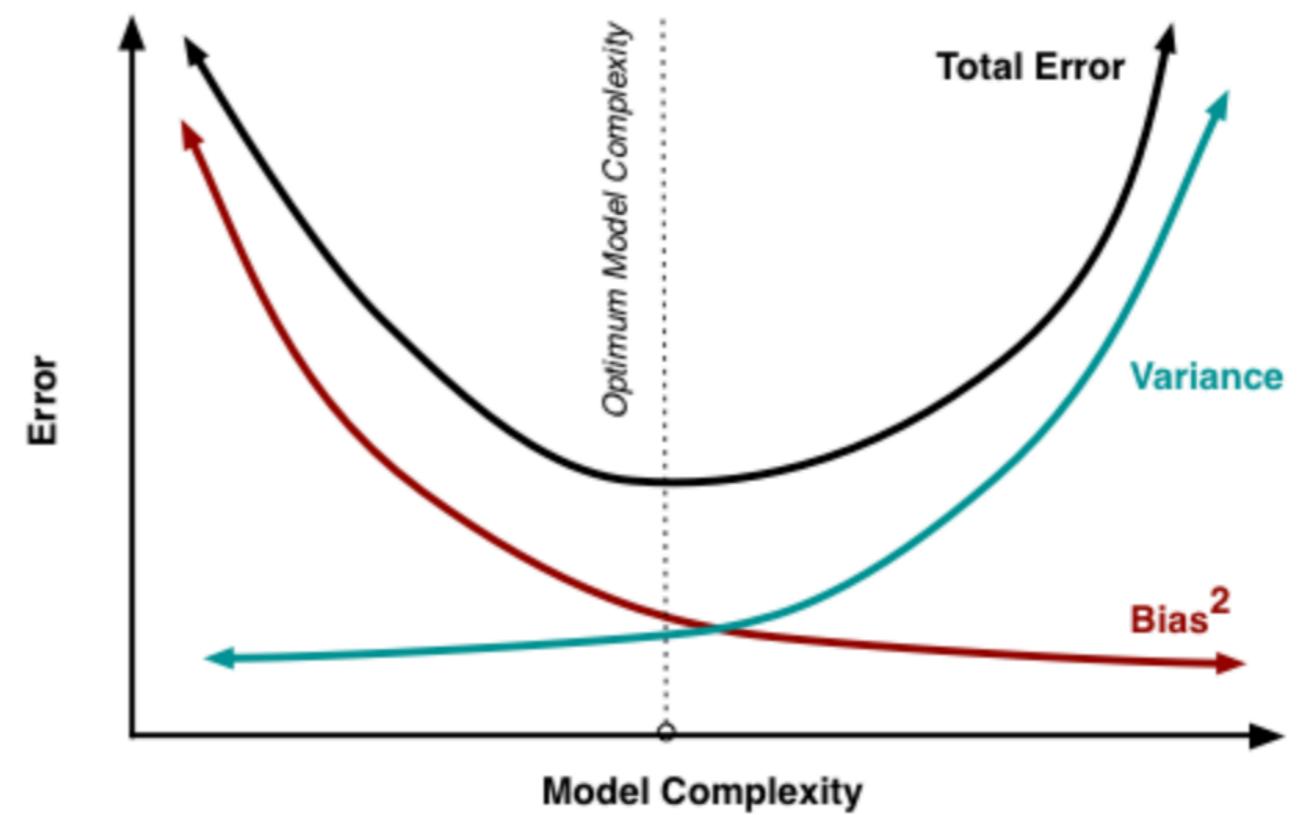
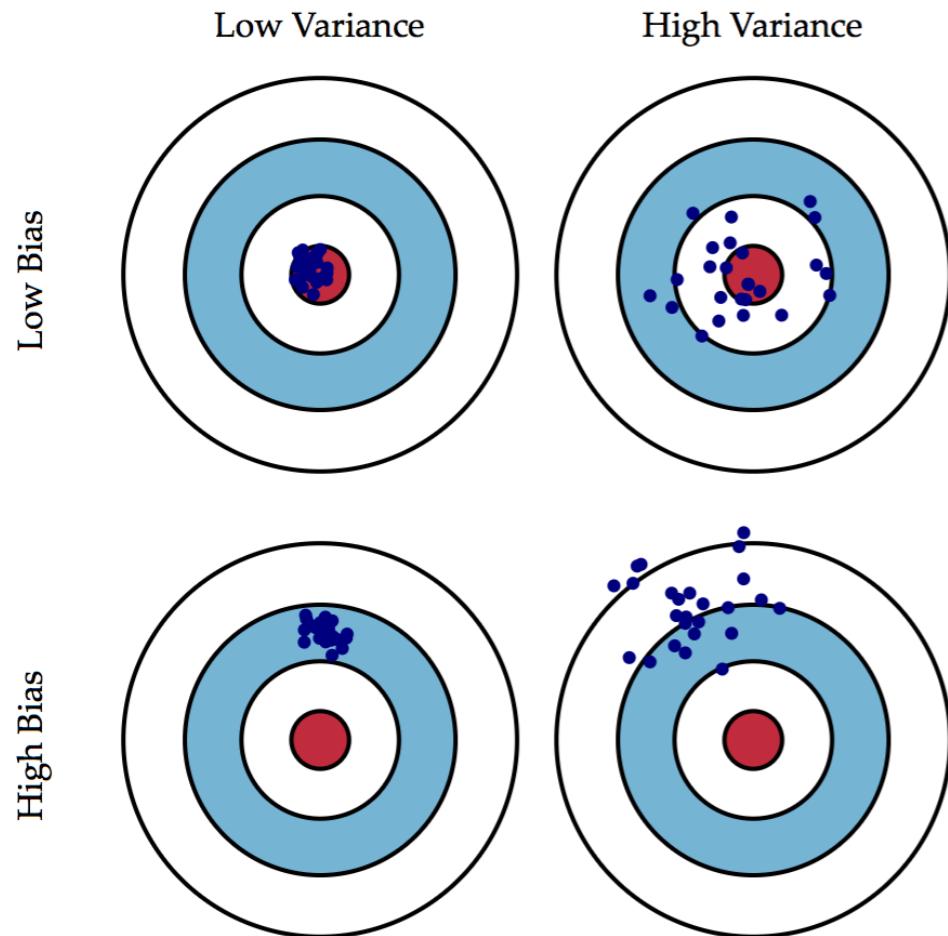
Пока один объект в листе

## Для регрессии:

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

Пока пять объектов в листе

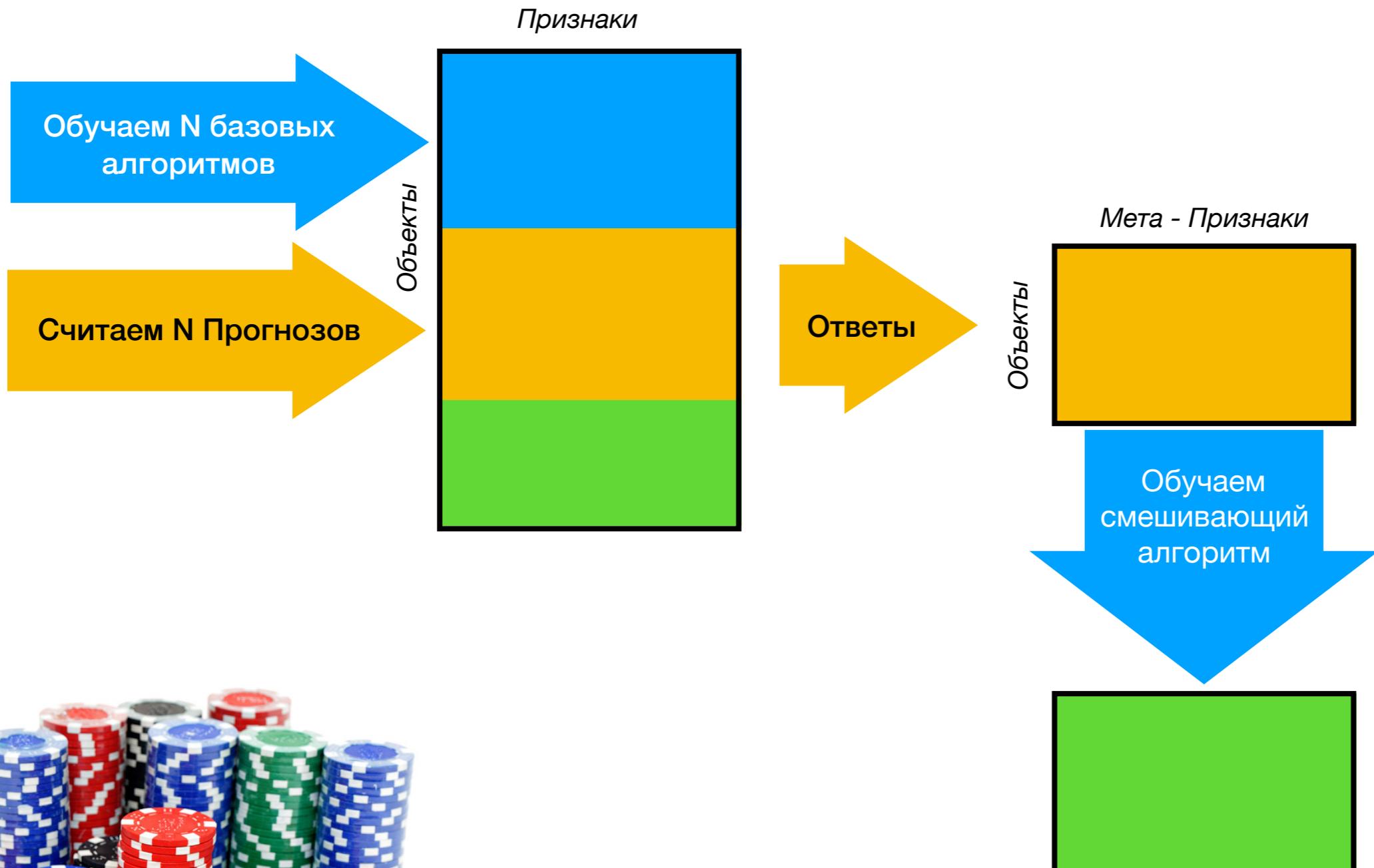
# Bias and Variance tradeoff



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

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

# Stacking



# Blending

**Blending** – частный случай Stacking'a

В качестве решающего алгоритма используется функция:

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

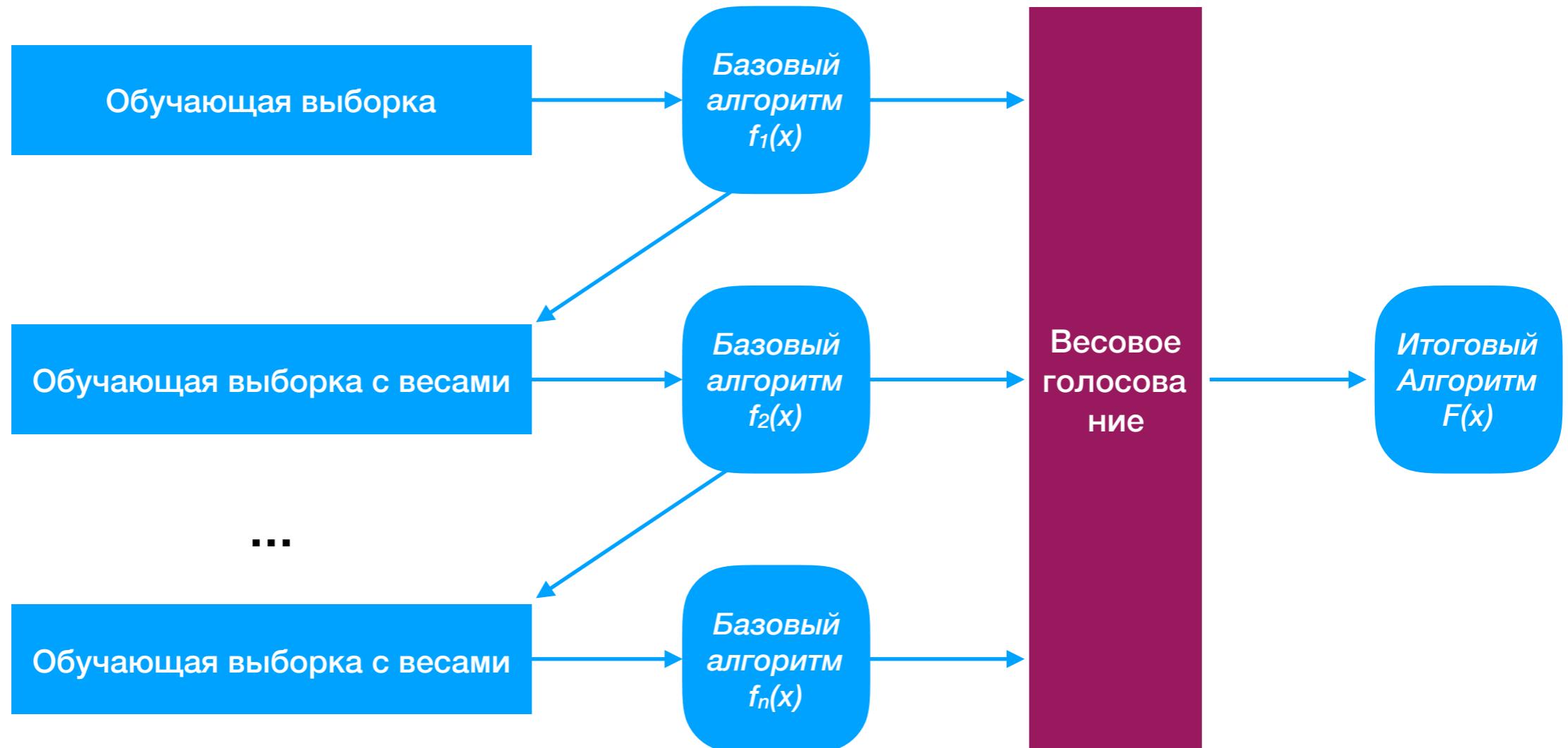
$f_i(x)$  Базовый алгоритм

$N$  Количество базовых алгоритмов



# Бустинг

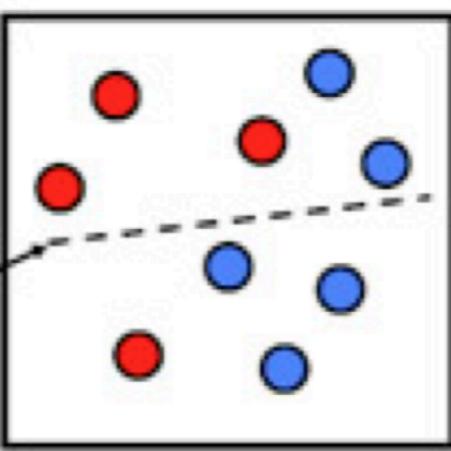
**Boosting** – последовательное добавление в ансамбль алгоритмов, каждый из которых корректирует ошибки предшественника.



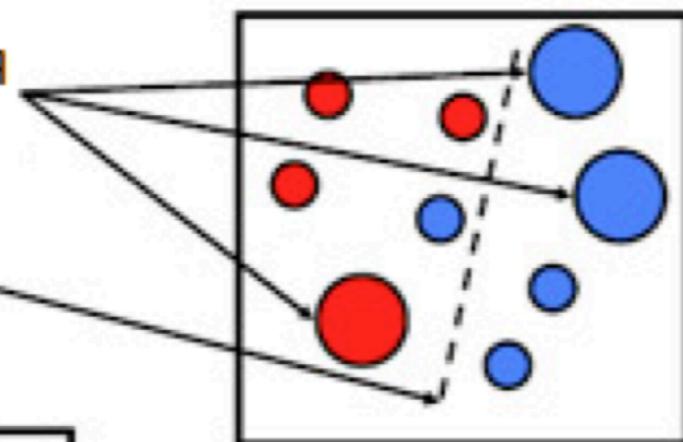
*AdaBoost (сокр. от adaptive boosting)*  
алгоритм, предложенный Йоавом Фройндом и Робертом Шапире Робертом  
Шапире

# AdaBoost

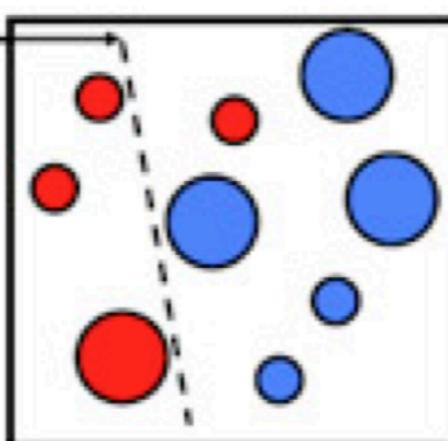
Равномерное  
распределение  
весов  
слабый  
классификатор  
1



Модификация  
весов  
слабый  
классификатор  
2



слабый  
классификатор  
3



Модификация  
весов

$$H(x) = \text{sign}(\alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x))$$

# Градиентный бустинг

1. Строим алгоритм

$$\hat{y} = f(x)$$

# Градиентный бустинг

1. Строим алгоритм

$$\hat{y} = f(x)$$

2. Получаем ответы и отклонения

$$y - \hat{y}$$

# Градиентный бустинг

1. Строим алгоритм

$$\hat{y} = f(x)$$

2. Получаем ответы и отклонения

$$y - \hat{y}$$

3. Получаем новую обучающую выборку

$$(x_1, y_1 - f(x)) \dots (x_n, y_n - f(x))$$

# Градиентный бустинг

1. Строим алгоритм

$$\hat{y} = f(x)$$

2. Получаем ответы и отклонения

$$y - \hat{y}$$

3. Получаем новую обучающую выборку

$$(x_1, y_1 - f(x)) \dots (x_n, y_n - f(x))$$

4. Обучаем классификатор

$$a_i(x)$$

# Градиентный бустинг

1. Строим алгоритм

$$\hat{y} = f(x)$$

2. Получаем ответы и отклонения

$$y - \hat{y}$$

3. Получаем новую обучающую выборку

$$(x_1, y_1 - f(x)) \dots (x_n, y_n - f(x))$$

4. Обучаем классификатор

$$a_i(x)$$

5. Объединяем

$$f(x) + a_i(x) = y$$

# Градиентный бустинг

1. Строим алгоритм

$$\hat{y} = f(x)$$

2. Получаем ответы и отклонения

$$y - \hat{y}$$

3. Получаем новую обучающую выборку

$$(x_1, y_1 - f(x)) \dots (x_n, y_n - f(x))$$

4. Обучаем классификатор

$$a_i(x)$$

5. Объединяем

$$f(x) + a_i(x) = y$$

6. Возвращаемся к п. 2

# Градиентный бустинг

$$\mathcal{Q} = \frac{1}{l} \sum_{i=1}^l \mathcal{L}(y, f(x))$$

*Рассмотрим  $f(x)$  как параметр,  
если Функция MSE*

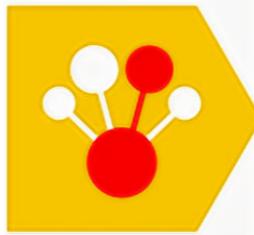
$$\frac{\partial \mathcal{Q}}{\partial f(x_i)} = f(x_i) - y$$

$y - \hat{y}$     *Отрицательный градиент!*

# Градиентный бустинг

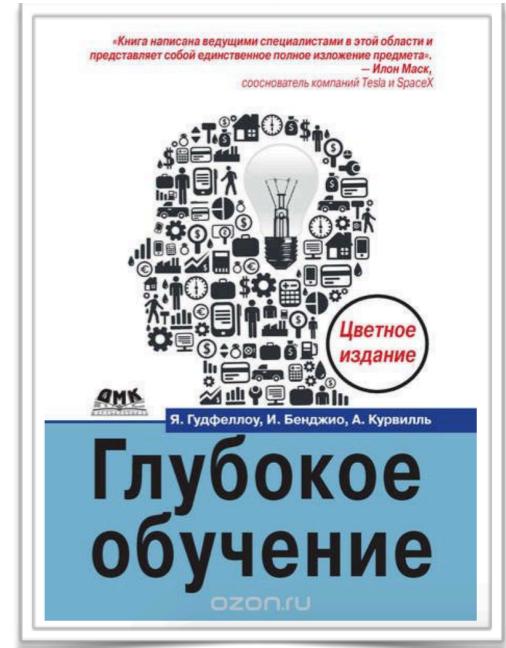
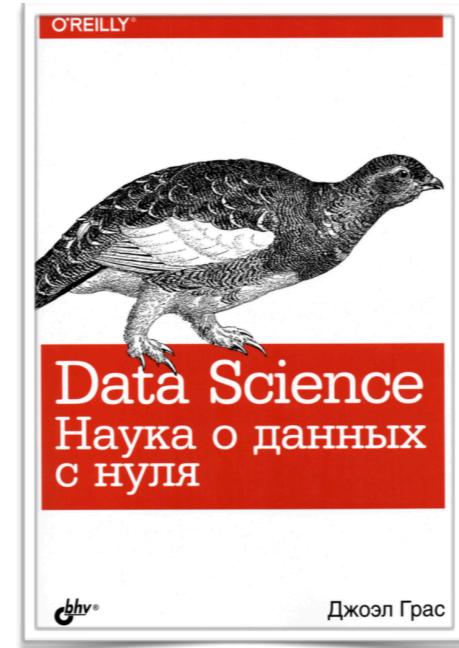
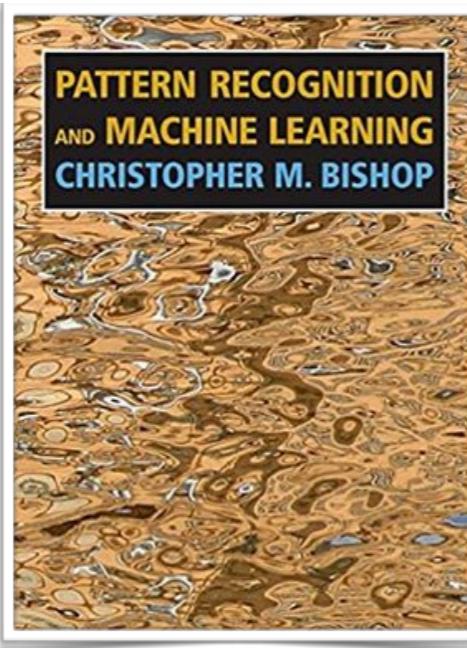
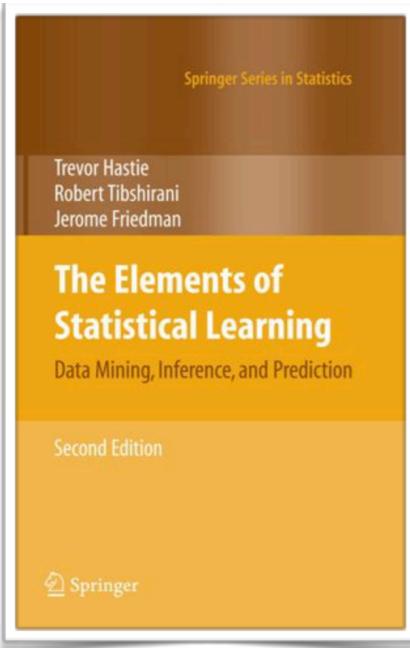


**XGBoost**



**H<sub>2</sub>O**

# Литература, курсы, ссылки



## Курсы:

- Открытый курс машинного обучения (ODS)
- Специализация МФТИ и Яндекс на Coursera
- Machine Learning от Andrew Ng
- Введение в машинное обучение от Яндекса и ВШЭ

## Ссылки:

- <https://github.com/ml-mipt>
- <https://www.openml.org>
- <https://opendatascience.slack.com>
- <https://www.kaggle.com>