

# Lecture 1

Корбут Даниил

Deep Learning Research Engineer, Insilico Medicine

telegram: @rtriangle

vk: rtriangle

email: korbut.daniel@gmail.com

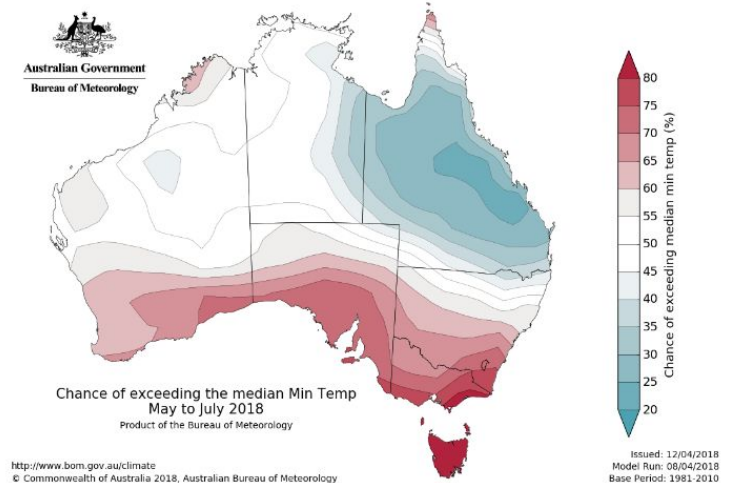
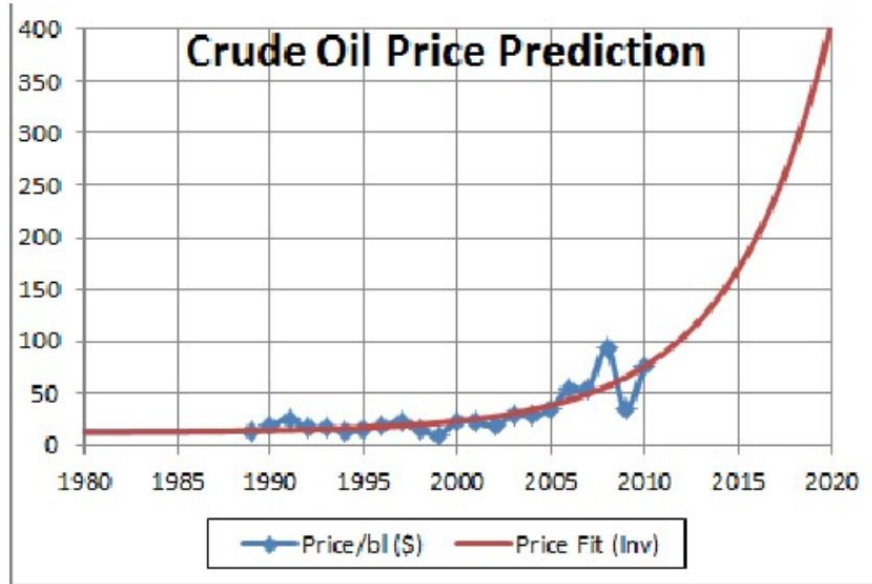
# План курса

- Основные направления ML, популярные алгоритмы и тонкости применения
- Решающие деревья, бустинг, ансамбли моделей
- Основы нейросетей
- Рекомендательные системы
- Защита проектов

# План занятия

- Основные направления применения ML, DL, AI
- Основные типы алгоритмов и процесс построения обучения
- Базовые алгоритмы
- Основные библиотеки ML, DL

# Основные направления ML, DL



## Специально для Вас

Аффинаж, gsac, номер скрыт и другие

▶ СЛУШАТЬ ВСЕ



## Приглядитесь к этим предложениям



3 295 ₽ **-50 %**  
6 590 ₽  
Кеды VANS



7 030 ₽ **-35 %**  
10 800 ₽  
Внешняя звуковая



1 875 000 ₽  
Виниловый  
проигрыватель Spira...



4 400 ₽ **-30 %**  
6 290 ₽  
Кеды VANS



11 790 ₽  
Лонгборд GoldCoast  
Standard



Сначала интересные



# Google

где найти|

где найти  
где найти **работу**  
где найти **девушку**  
где найти **друзей**  
где найти **парня**  
где найти **мужа**  
где найти **деньги**  
где найти **ответы на огэ 2018**  
где найти **ответы на егэ 2018**  
где найти **алису**

Страница



Свидание III в Москве  
Событие



Линзы ACUVUE® в  
"Очкарик!"  
ochkarik.ru  
Удивительное  
предложение на  
контактные линзы  
ACUVUE OASYS® 1-Day!  
Есть противопоказания.  
Требуется консультация  
специалиста.

Блог Разработчикам

55491\_219500906

# беру! BETA

кто лучше сири или ты

Я сейчас чуть не удалилась в сердцах. Конечно, я не Сири.

переведи меня зовут алиса на английский

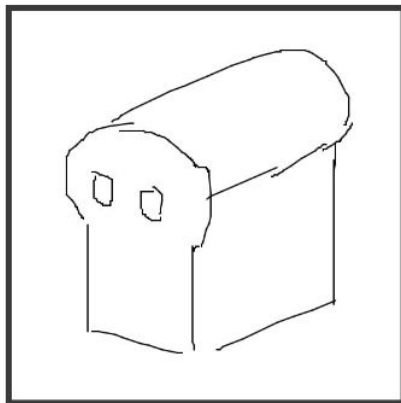
Лет ми спик фром май харт:  
пойдёмте в Яндекс.Переводчик.  
Там переводить проще.

[ОТКРЫТЬ ЯНДЕКС.ПЕРЕВОДЧИК](#)

[ПОИСКАТЬ В ЯНДЕКСЕ](#)



INPUT



pix2pix

process

OUTPUT

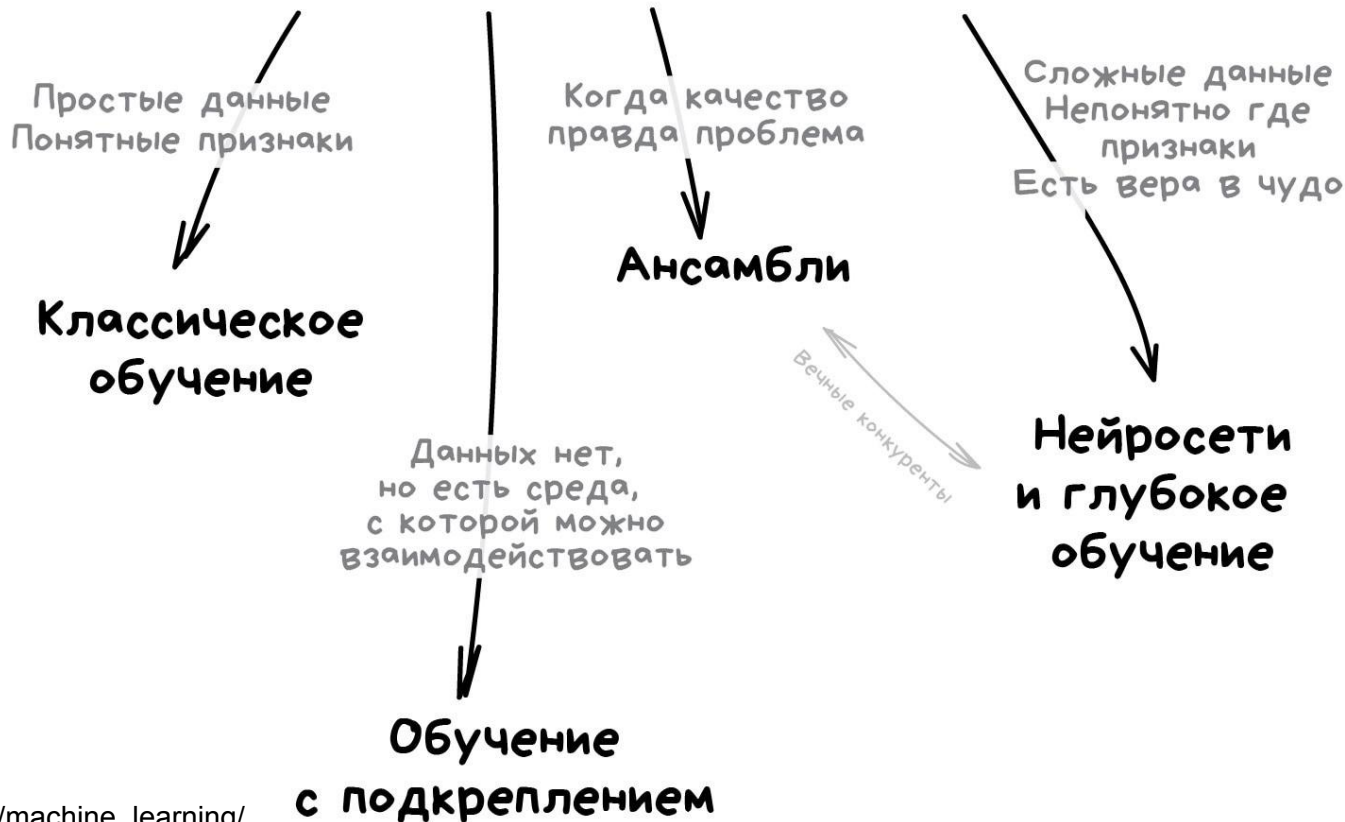




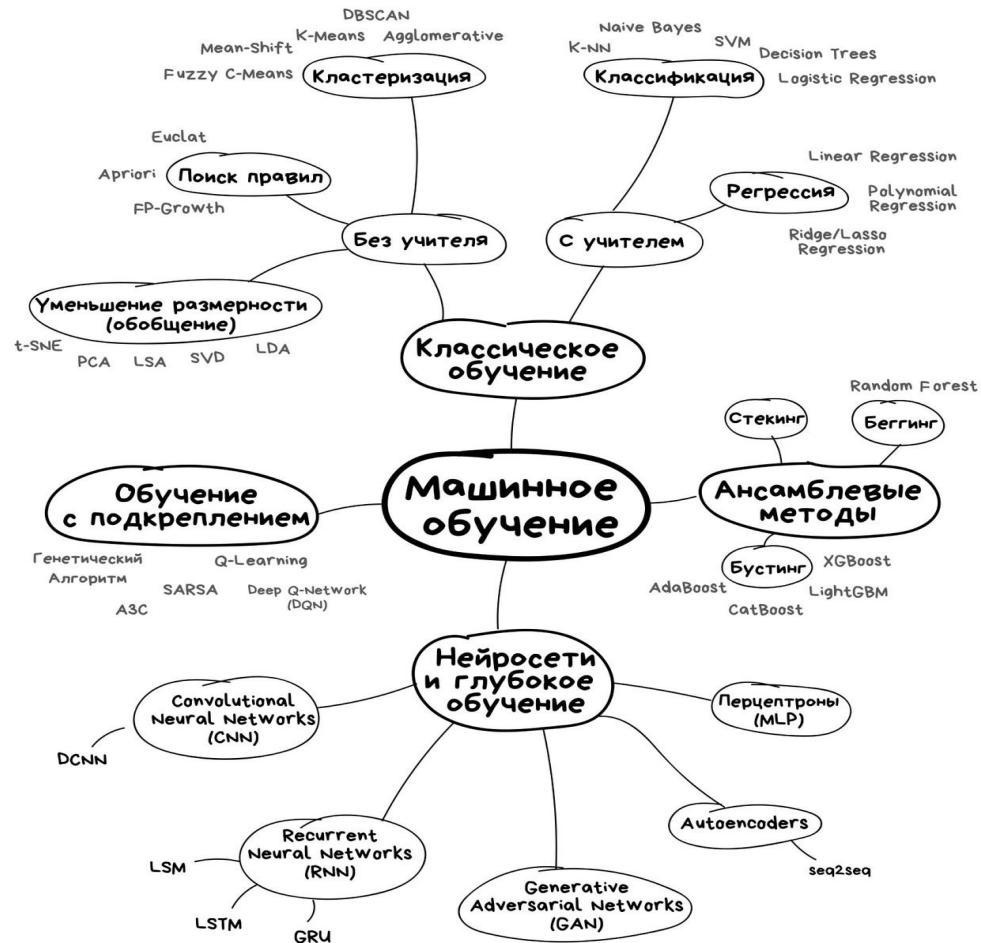


# Зоопарк моделей

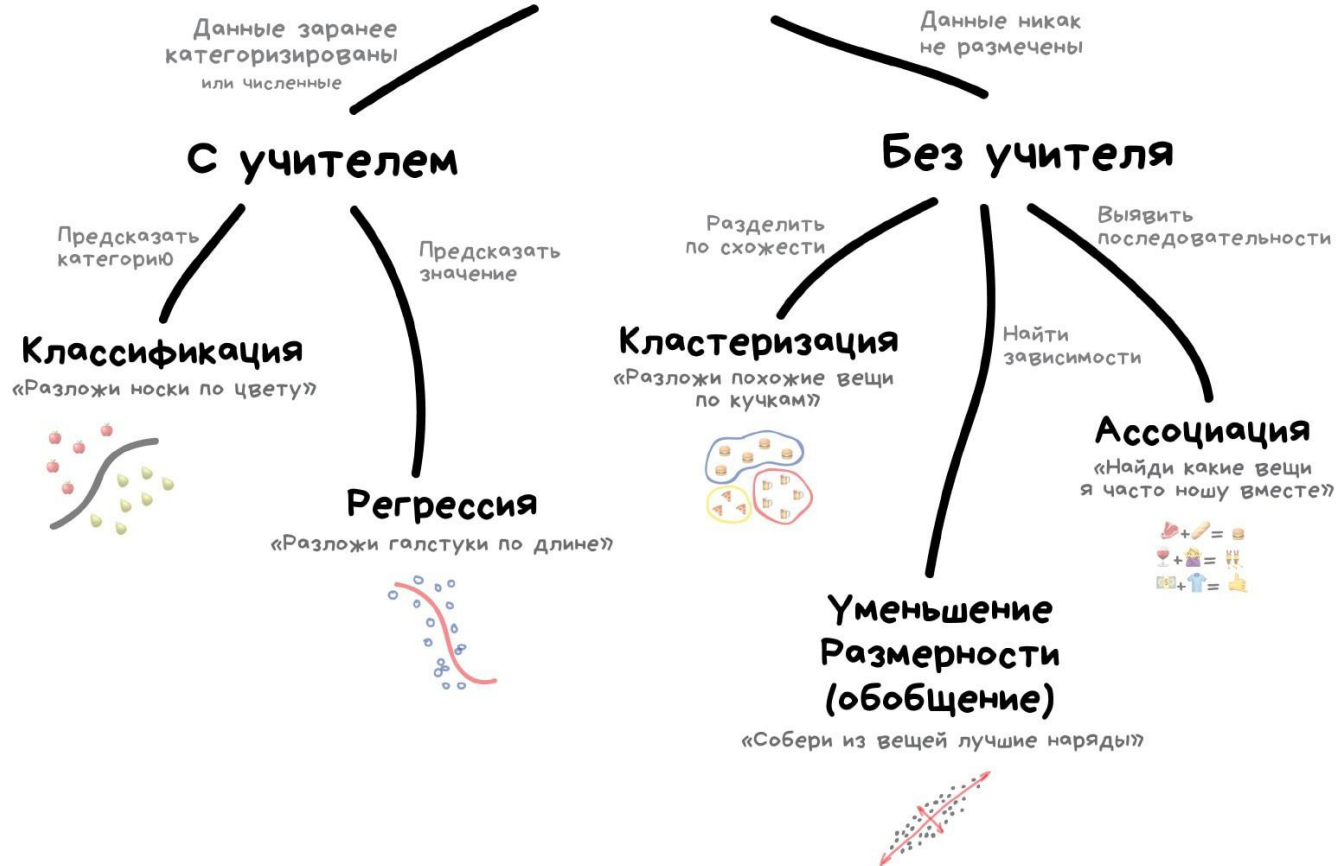
## Основные виды машинного обучения





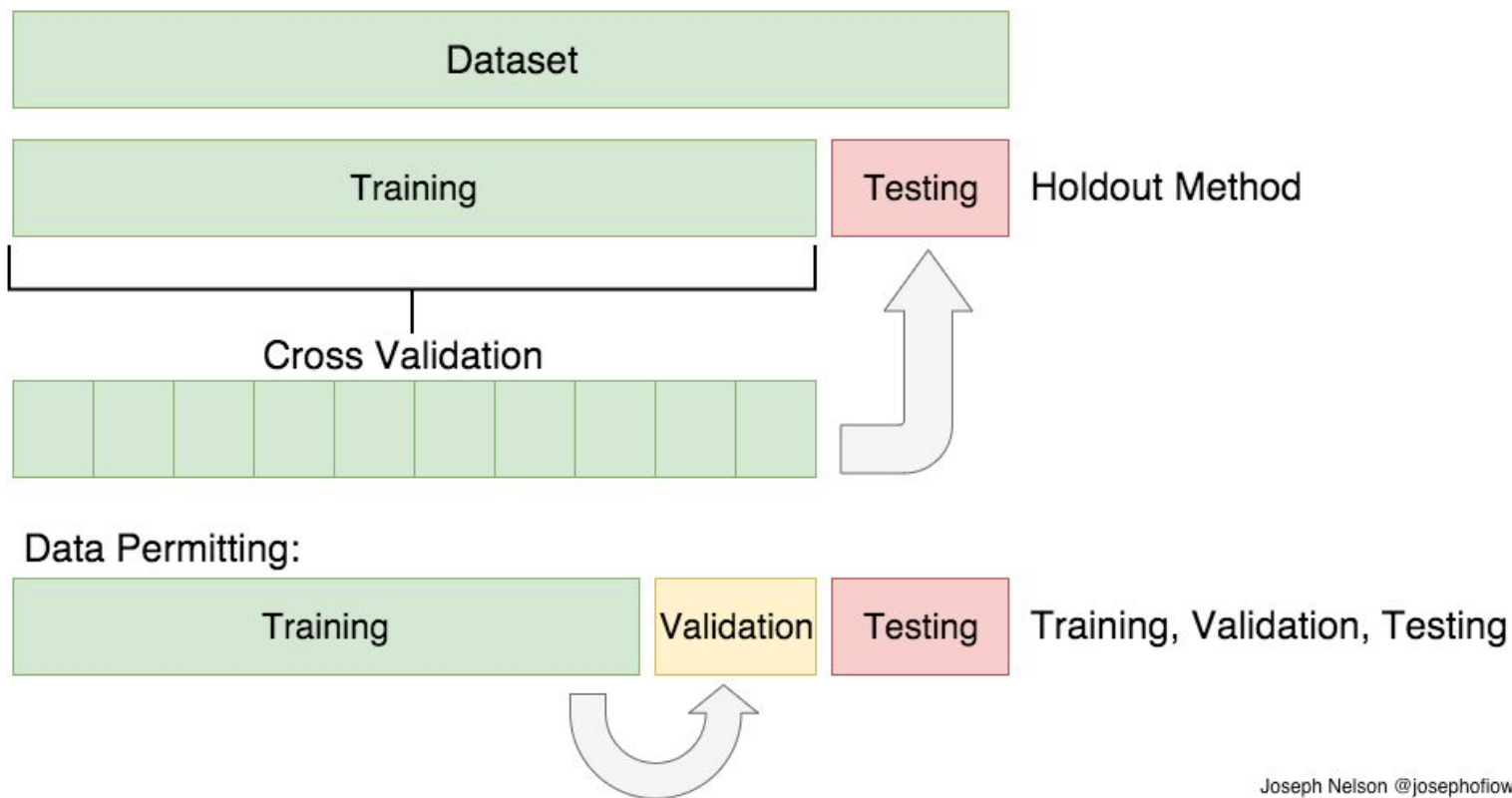


# Классическое Обучение

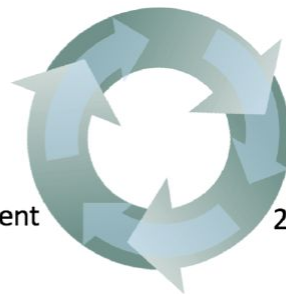


# Процесс обучения

- Размер dev и test датасетов
- Однородность train, dev, test
- Выбор алгоритма
- Метрика
- Анализ ошибок

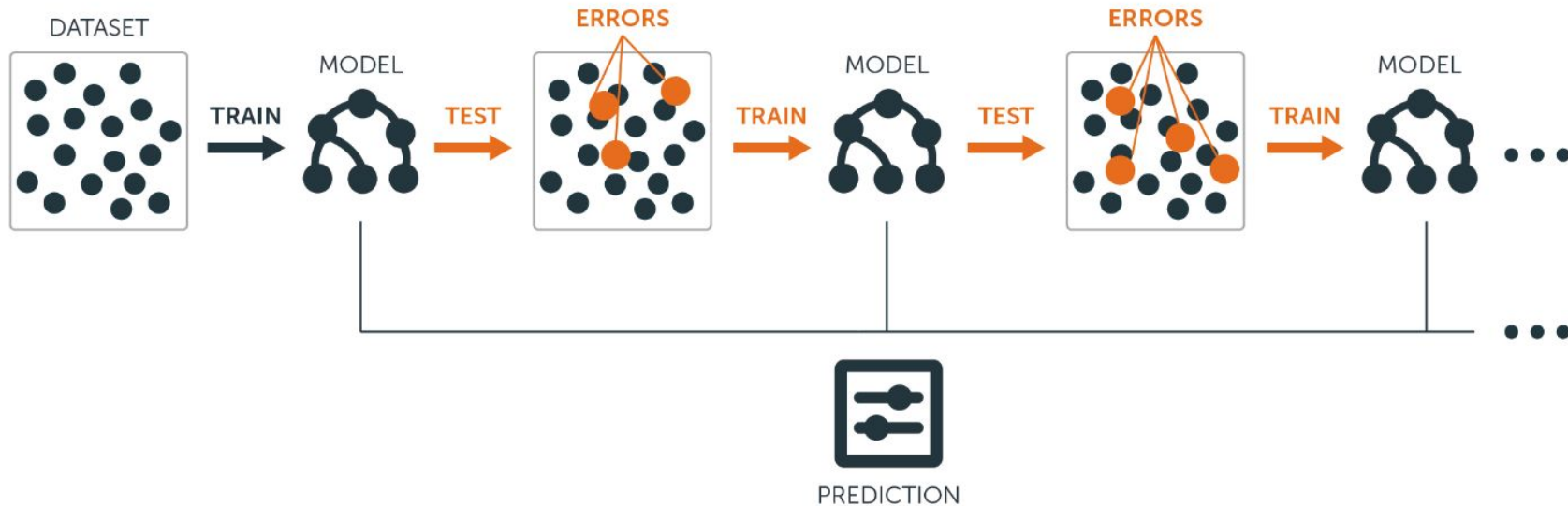


1. Idea



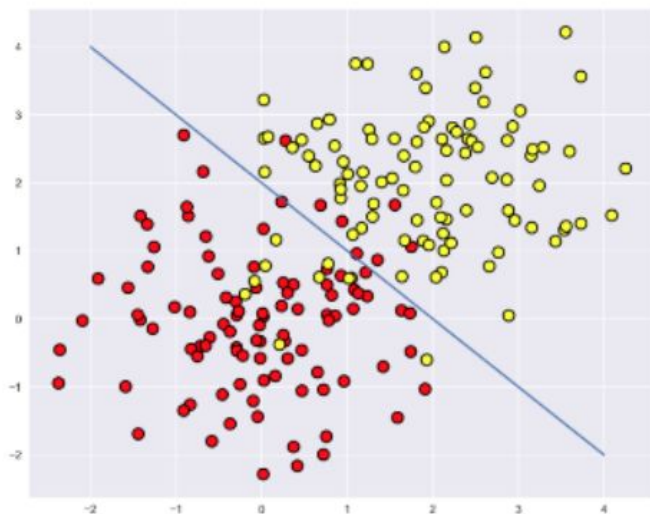
2. Code

3. Experiment



## Классификация

*Множество допустимых ответов конечно. Их называют метками классов (class label). Класс — это множество всех объектов с данным значением метки.*

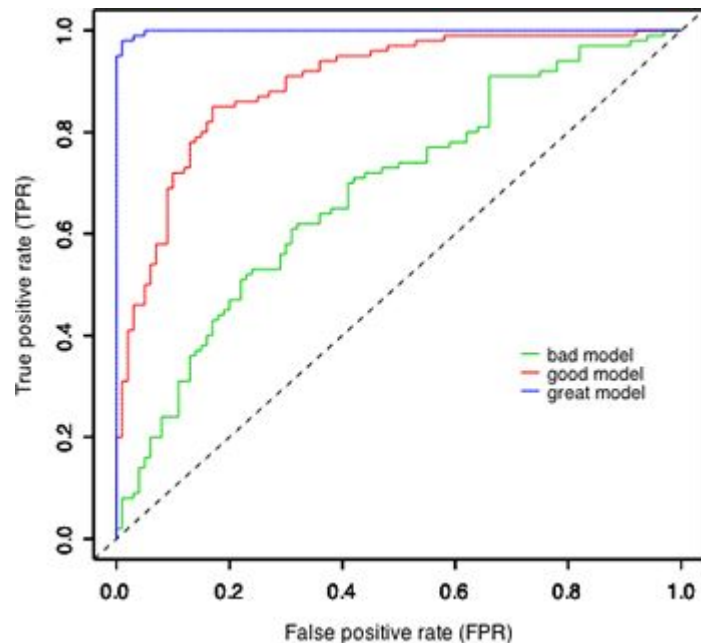


# Метрики классификации

		predicted condition		
		prediction positive	prediction negative	
true condition	total population			<b>Sensitivity</b>
	condition positive	True Positive (TP)	False Negative (FN) (Type II error)	<b>Recall =</b> $\frac{\sum TP}{\sum \text{condition positive}}$
	condition negative	False Positive (FP) (Type I error)	True Negative (TN)	<b>Specificity =</b> $\frac{\sum TN}{\sum \text{condition negative}}$
		<b>Precision =</b> $\frac{\sum TP}{\sum \text{prediction positive}}$		<b>F1 Score =</b> $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		<b>Accuracy =</b> $\frac{\sum TP + \sum TN}{\sum \text{total population}}$		



## Roc-AUC



## Log-Loss

$$-(y \log(p) + (1 - y) \log(1 - p))$$

predicted→ real↓	<i>Class_pos</i>	<i>Class_neg</i>
<i>Class_pos</i>	TP	FN
<i>Class_neg</i>	FP	TN

$$\text{TPR (sensitivity)} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR (1-specificity)} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

# Сравнение Log-loss, F1, Roc

S.No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Actual (Balanced)	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Predicted (Model 1)	0.1	0.1	0.1	0.1	0.1	0.1	0.6	0.6	0.5	0.5	0.9	0.9	0.9	0.9	0.9	0.9
Predicted (Model 2)	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.7	0.7	0.7	0.8	0.8	0.8	0.8

	F1 (threshold=0.5)	F1 (Threshold which maximize score)	ROC-AUC	Log-Loss
<b>Model 1</b>	0.88	0.88	0.94	0.28
<b>Model 2</b>	0.67	1	1	0.6

Выводы из сравнения метрик для модели (сбалансированные классы):

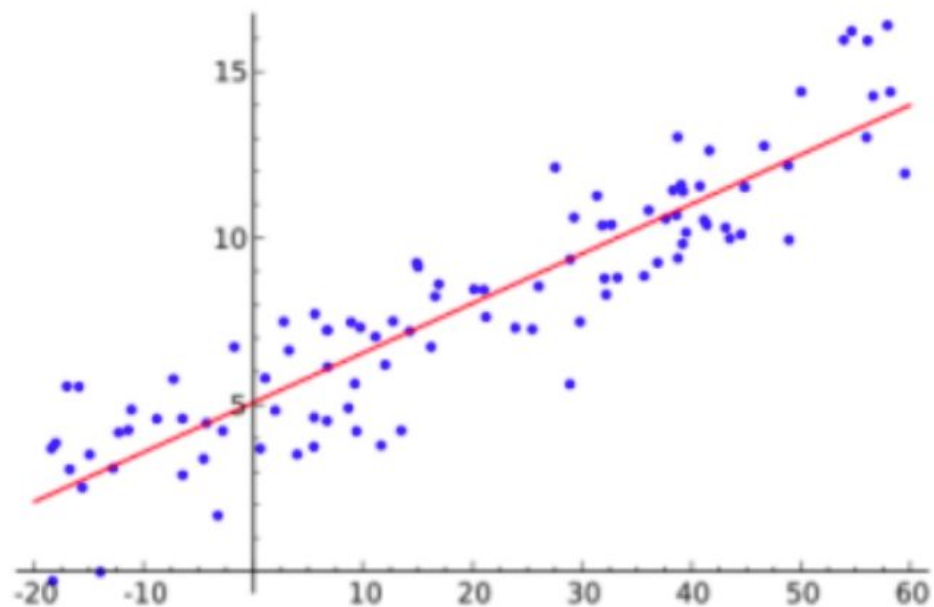
- Если хотим выбрать модель с минимальной абсолютной разницей по вероятности класса, берём log-loss
- Если хотим только финальные предсказания и не хотим тюнить порог, берём AUC
- F1 чувствителен к порогу, и необходимо тюнить его перед сравнением моделей

S.No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Actual (Imbalanced – few positive)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
Predicted (Model 1)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9
Predicted (Model 2)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9	0.9	0.9
Actual (Imbalanced – few negative)	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
Predicted (Model 3)	0.1	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
Predicted (Model 4)	0.1	0.1	0.1	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9

	F1 (threshold=0.5)	ROC-AUC	Log-Loss
<b>Model 1</b>	0.8	0.83	0.24
<b>Model 2</b>	0.86	0.96	0.24
<b>Model 3</b>	0.963	0.83	0.24
<b>Model 4</b>	0.96	0.96	0.24

# Регрессия

Отличается тем, что допустимым ответом является действительное число или числовой вектор.



$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2}$$

Case 1: Actual Values = [2,4,6,8] , Predicted Values = [4,6,8,10]

Case 2: Actual Values = [2,4,6,8] , Predicted Values = [4,6,8,12]

$$\text{MAE} = \frac{1}{n} \sum_{j=1}^n |y_j - \hat{y}_j|$$

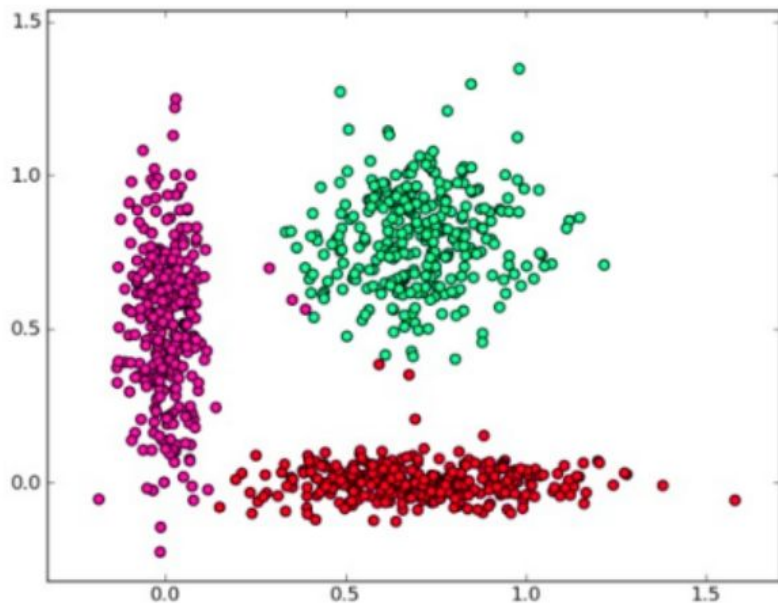
**MAE for case 1 = 2.0, RMSE for case 1 = 2.0**

**MAE for case 2 = 2.5, RMSE for case 2 = 2.65**

$$\hat{R}^2 = 1 - \frac{\sum_{i=1}^n (Y_i - \hat{Y}_i)^2}{\sum_{i=1}^n (Y_i - \bar{Y})^2} = 1 - \frac{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2}{\frac{1}{n} \sum_{i=1}^n (Y_i - \bar{Y})^2}$$

## Кластеризация

Заключается в том, чтобы сгруппировать объекты в кластеры, используя данные о попарном сходстве объектов. Функционалы качества могут определяться по-разному, например, как отношение средних межкластерных и внутрикластерных расстояний.





- Оценка качества не должна зависеть от самих значений меток, а только от самого разбиения выборки.
- Не всегда известны истинные метки объектов, поэтому также нужны такие оценки качества, которые бы позволили оценить качество кластеризации, используя только неразмеченную выборку
- внешние
- внутренние

# Adjusted Rand Index (ARI)

Предполагается, что известны истинные метки объектов. Данная мера не зависит от самих значений меток, а только от разбиения выборки на кластеры. Пусть  $n$  — число объектов в выборке. Обозначим через  $a$  — число пар объектов, имеющих одинаковые метки и находящихся в одном кластере, через  $b$  — число пар объектов, имеющих различные метки и находящихся в разных кластерах. Тогда Rand Index это

$$RI = \frac{2(a + b)}{n(n - 1)}.$$

То есть это доля объектов, для которых эти разбиения (исходное и полученное в результате кластеризации) "согласованы". Rand Index (RI) выражает схожесть двух разных кластеризаций одной и той же выборки. Чтобы этот индекс давал значения близкие к нулю для случайных кластеризаций при любом  $n$  и числе кластеров, необходимо нормировать его. Так определяется Adjusted Rand Index:

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]}.$$

# Коэффициент силуэта

В отличие от описанных выше метрик, данный коэффициент не предполагает знания истинных меток объектов, и позволяет оценить качество кластеризации, используя только саму (неразмеченную) выборку и результат кластеризации. Сначала силуэт определяется отдельно для каждого объекта. Обозначим через  $a$  — среднее расстояние от данного объекта до объектов из того же кластера, через  $b$  — среднее расстояние от данного объекта до объектов из ближайшего кластера (отличного от того, в котором лежит сам объект). Тогда силуэтом данного объекта называется величина:

$$s = \frac{b - a}{\max(a, b)}.$$

Силуэтом выборки называется средняя величина силуэта объектов данной выборки. Таким образом, силуэт показывает, насколько среднее расстояние до объектов своего кластера отличается от среднего расстояния до объектов других кластеров. Данная величина лежит в диапазоне  $[-1, 1]$ .

# Анализ ошибок

Фокус на то, что улучшит качество модели больше всего.

- рассмотрение классов, на которых происходит ошибка
- неправильная разметка
- зашумлённые изображения
- ...



# Bias vs Variance

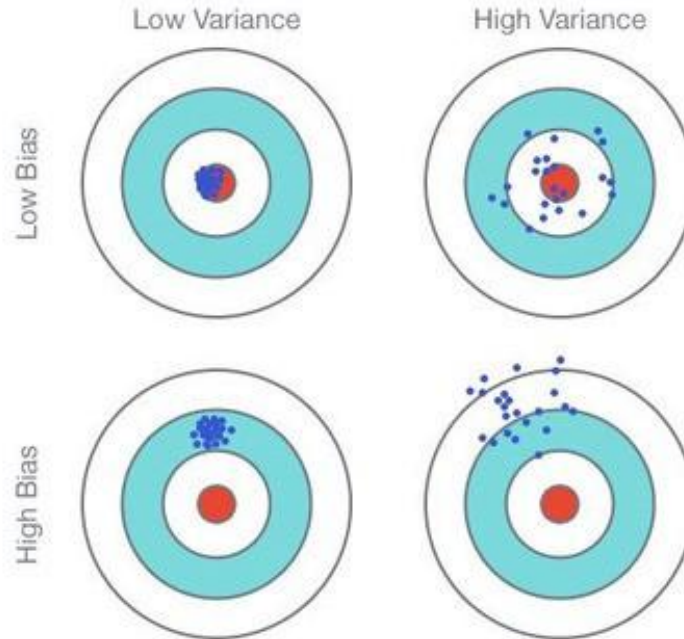


Fig. 1: Graphical Illustration of bias-variance trade-off , Source: Scott Fortmann-Roe., Understanding Bias-Variance Trade-off

Рассмотрим на примере задачи классификации:

- Training error = 1%
- Dev error = 11%

(bias = 1%, variance = 11% - 1% = 10% => high variance => overfitting)

- Training error = 15%
- Dev error = 16%

(bias = 15%, variance = 1% => high bias => underfitting)

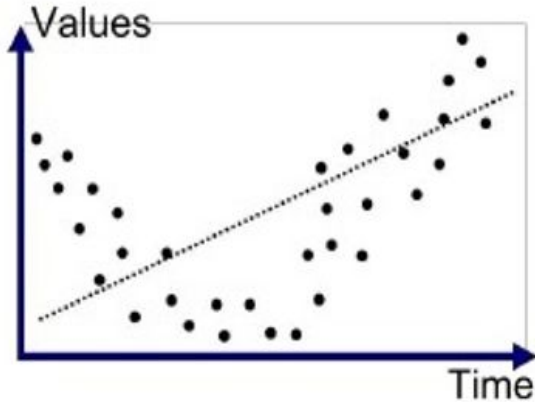
- Training error = 15%
- Dev error = 30%

(bias = 15%, variance = 15% => high bias and high variance)

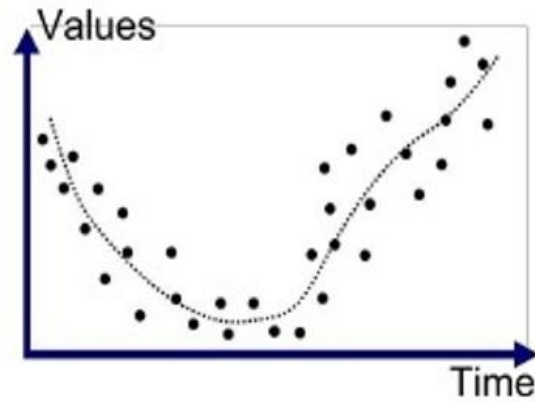
- Training error = 0.5%
- Dev error = 1%

(bias = 0.5%, variance = 0.5% => good job!)

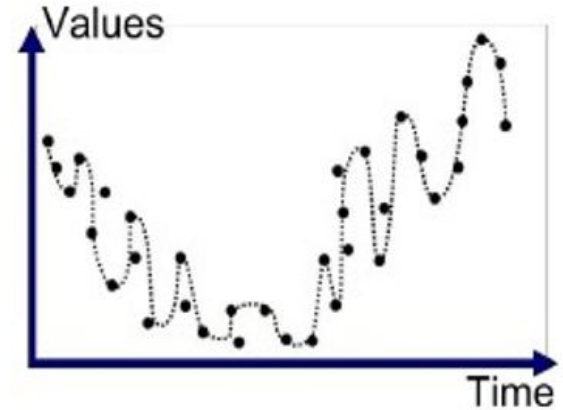
# Проблемы преобучения на данных



Underfitted



Good Fit/Robust



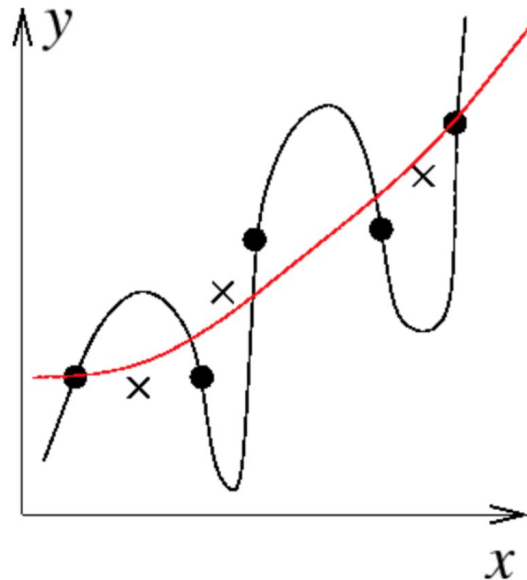
Overfitted



# Переобучение

Из-за чего возникает переобучение?

- Переобучение есть всегда, когда выбор делается на основе заведомо неполной информации
- Слишком сложная/гибкая модель может чрезмерно подстроиться под обучающую выборку и потерять способность находить нижележащие закономерности в новых данных



# Базовые алгоритмы

$X$  — множество **объектов**

$Y$  — множество **допустимых ответов**

$y^*$  — целевая функция,  $y^*: X \rightarrow Y$ ,  $y_i = y^*(x_i)$  известны только на **конечном** подмножестве объектов  $x_1, \dots, x_m$  из  $X$

Пары  $(x_i, y_i)$  — прецеденты

Совокупность пар таких пар при  $i$  из  $1, \dots, m$  — **обучающая выборка** ( $X_{train}$ )

$a$  — **решающая функция** (алгоритм), которая любому объекту из  $X$  ставит в соответствие допустимый ответ из  $Y$  и приближает целевую функцию  $y^*$

$X_{test}$  — **выборка прецедентов** для тестирования построенного алгоритма  $a$

Для решения задачи обучения по прецедентам в первую очередь фиксируется восстанавливаемой зависимости.

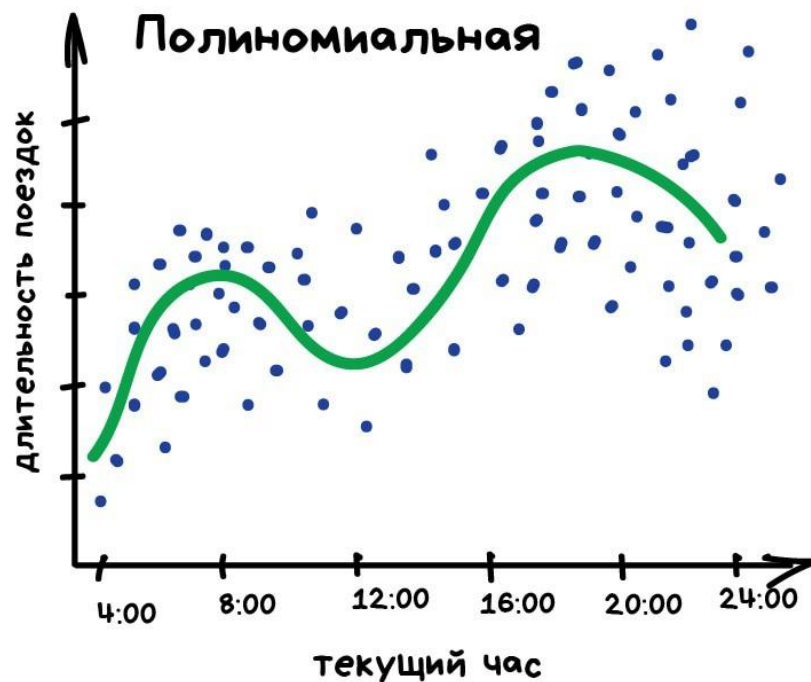
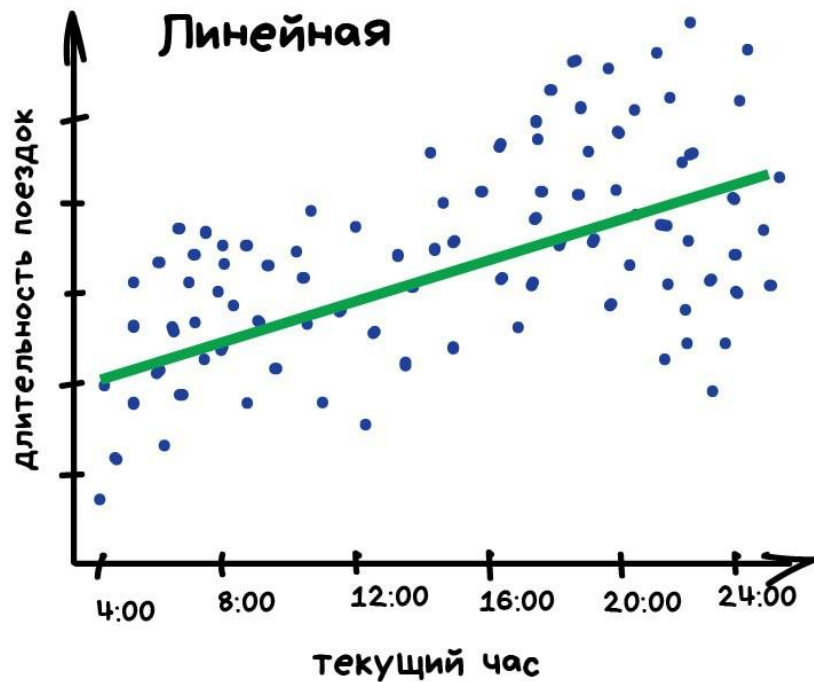
# Базовые алгоритмы

**Признак** (feature)  $f$  объекта  $x$  — это результат измерения некоторой характеристики объекта. Формально признаком называется отображение  $f : X \rightarrow D_f$ , где  $D_f$  — множество допустимых значений признака. В частности, любой алгоритм  $a : X \rightarrow Y$  также можно рассматривать как признак

Пусть дан набор признаков  $f_1(x), \dots, f_n(x)$ .

**Признаковое описание объекта**  $x$  — вектор (одномерный массив)  $(f_1, \dots, f_n)$ . Совокупность признаковых описаний всех объектов выборки длины  $m$ , записанную в виде таблицы размера  $mn$ , называют матрицей объектов–признаков.

# Предсказываем пробки



Регрессия

# Пример - линейная регрессия

- Ищем алгоритм  $a$  в классе линейных алгоритмов:

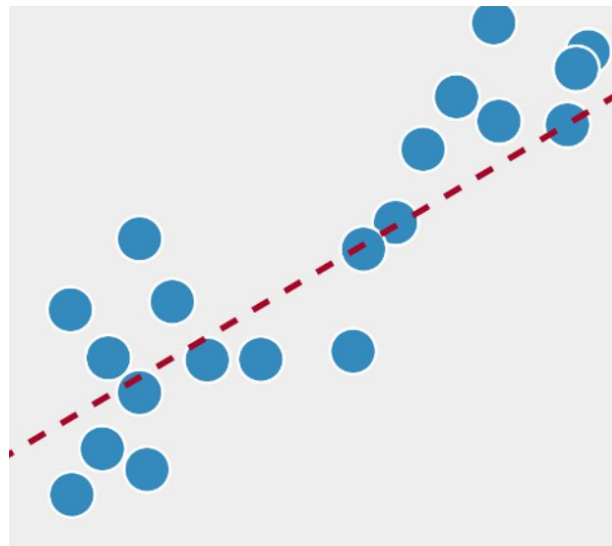
$$y_{\text{predicted}} = a(x) = \langle w, x \rangle - w_0$$

- Настраиваем веса  $w, w_0$  так, чтобы минимизировать MSE:

$$(y_{\text{predicted}} - y_{\text{true}})^2 \rightarrow \min$$

- Итоговая задача оптимизации:

$$(\langle w, x \rangle - w_0 - y_{\text{true}})^2 \rightarrow \min$$



# Давать ли кредит?



Дер

# Пример - решающее дерево

В корне дерева — рассматриваем всю обучающую выборку.

Проверить критерий останова алгоритма. Если он выполняется, выбрать для узла выдаваемый прогноз, что можно сделать несколькими способами.

Иначе требуется разбить множество на несколько не пересекающихся. В общем случае в вершине  $t$  задаётся решающее правило  $Q_t(x)$ , принимающее некоторый диапазон значений. Этот диапазон разбивается на  $R_t$  непересекающихся множеств объектов:  $S_1, S_2, \dots, S_{R_t}$ , где  $R_t$  — количество потомков у вершины, а каждое  $S_i$  — это множество объектов, попавших в  $i$ -го потомка.

Множество в узле разбивается согласно выбранному правилу, для каждого узла алгоритм запускается рекурсивно.



# Простейший спам-фильтр

(использовались года до 2010)

привет... 1829  
валера ...1710  
нет ... 1191  
куда ... 1012  
небо ...985  
огурцы ... 873  
говорить...747  
третий ... 739

нормальные  
письма

672 раза

«КОТИК»

13 раз

виагра ... 1552  
казино ... 1492  
100% ... 1320  
кредит... 1184  
скидка ... 985  
нажми ... 873  
free ... 747  
доход ... 739

спам-письма

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

формула Байеса



не спам

## Наивный Байес

# Наивный Байес

Пусть у нас есть строка текста  $O$ . Кроме того, имеются классы  $C$ , к одному из которых мы должны отнести строку. Нам необходимо найти такой класс  $c$ , при котором его вероятность для данной строки была бы максимальна.

$$c = \arg \max_C P(C|O)$$

Вычислить  $P(C|O)$  сложно. Но можно воспользоваться теоремой Байеса и перейти к косвенным вероятностям:

$$P(C|o_1 o_2 \dots o_n) = \frac{P(o_1 o_2 \dots o_n | C) P(C)}{P(o_1 o_2 \dots o_n)}$$

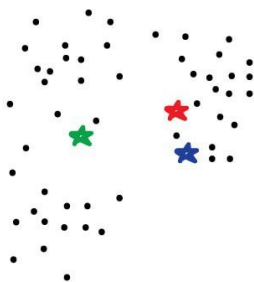
Но это опять сложно. Здесь включаем «наивное» предположение о том, что переменные  $O$  зависят только от класса  $C$ , и не зависят друг от друга. Это сильно упрощение, но зачастую это работает.

Числитель примет вид:

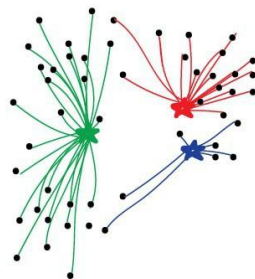
$$P(C)P(o_1|C)P(o_2|Co_1)\dots P(o_n|Co_1 o_2 \dots o_n) = P(C)P(o_1|C)P(o_2|C)\dots P(o_n|C) = P(C) \prod_i P(o_i|C)$$
$$c = \arg \max_{c \in C} P(c|o_1 o_2 \dots o_n) = \arg \max_{c \in C} P(c) \prod_i P(o_i|c)$$

# Ставим три ларька с шаурмой оптимальным образом

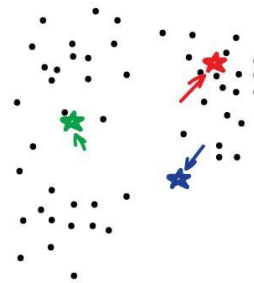
(иллюстрируя метод К-средних)



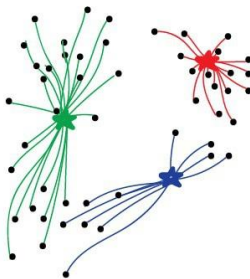
1. Ставим ларьки с шаурмой  
в случайных местах



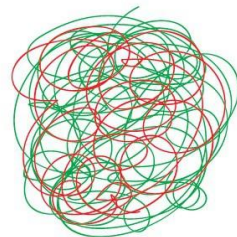
2. Смотрим в какой  
кому ближе идти



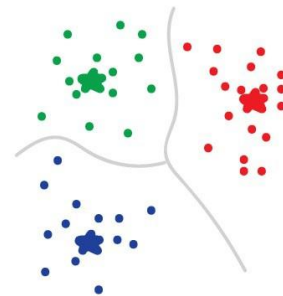
3. Двигаем ларьки ближе  
к центрам их популярности



4. Снова смотрим и двигаем



5. Повторяем много раз



6. Готово, вы великолепны!

[https://vas3k.ru/blog/machine\\_learning/](https://vas3k.ru/blog/machine_learning/)

# Метод К-средних

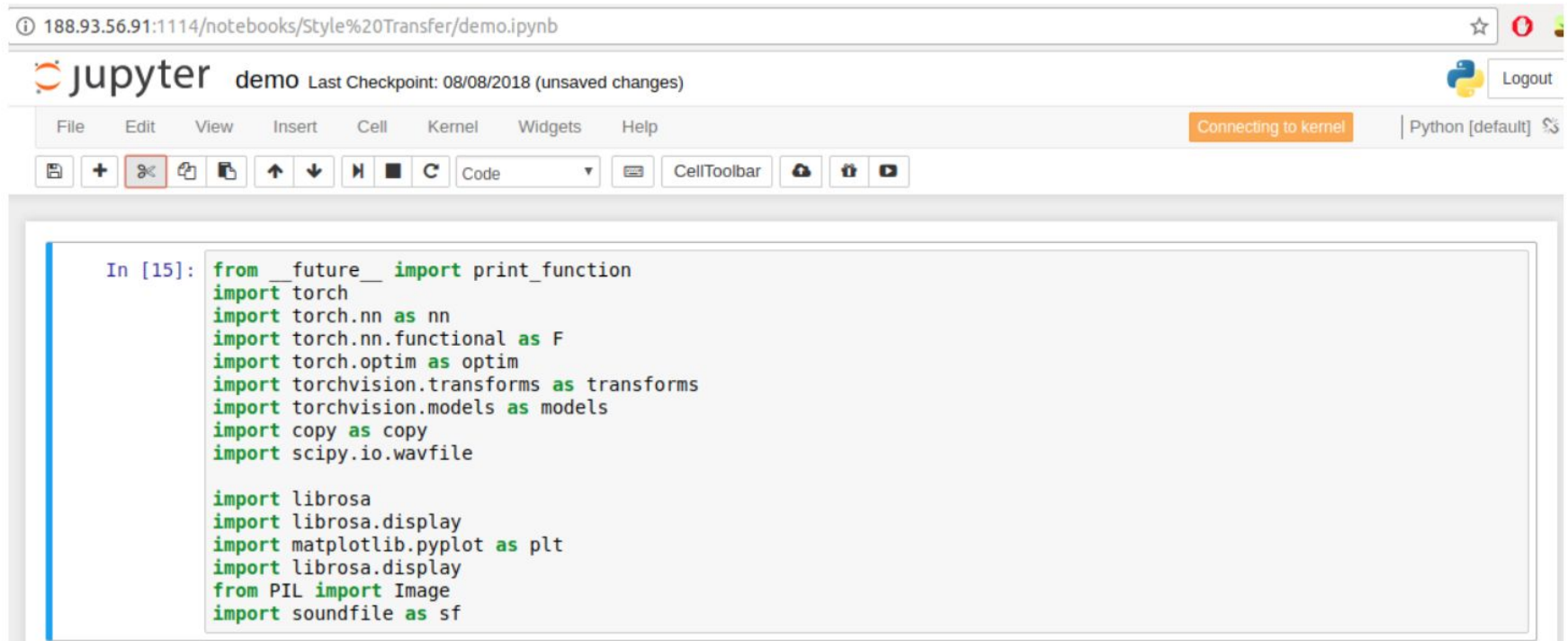
Действие алгоритма таково, что он стремится минимизировать среднеквадратичное отклонение на точках каждого кластера. Основная идея заключается в том, что на каждой итерации пересчитывается центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике. Алгоритм завершается, когда на какой-то итерации не происходит изменения кластеров.

Проблемы алгоритма k-means:

- необходимо заранее знать количество кластеров
- алгоритм очень чувствителен к выбору начальных центров кластеров

# Jupyter notebook

## Jupyter notebook



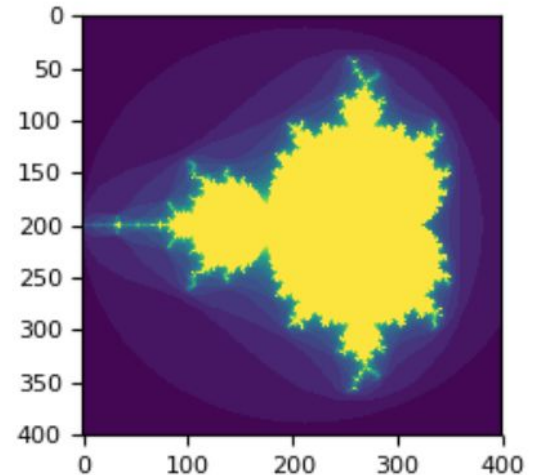
The screenshot shows a Jupyter Notebook interface in a web browser. The address bar displays the URL `188.93.56.91:1114/notebooks/Style%20Transfer/demo.ipynb`. The Jupyter logo and the word "demo" are visible, along with the text "Last Checkpoint: 08/08/2018 (unsaved changes)". The top navigation bar includes menus for File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A status bar on the right indicates "Connecting to kernel" and "Python [default]". The main area contains a code cell with the following Python code:

```
In [15]: from __future__ import print_function
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import torchvision.transforms as transforms
import torchvision.models as models
import copy as copy
import scipy.io.wavfile

import librosa
import librosa.display
import matplotlib.pyplot as plt
import librosa.display
from PIL import Image
import soundfile as sf
```

# Numpy

```
>>> import numpy as np
>>> import matplotlib.pyplot as plt
>>> def mandelbrot( h,w, maxit=20 ):
...     """Returns an image of the Mandelbrot fractal of size (h,w)."""
...     y,x = np.ogrid[ -1.4:1.4:h*1j, -2:0.8:w*1j ]
...     c = x+y*1j
...     z = c
...     divtime = maxit + np.zeros(z.shape, dtype=int)
...
...     for i in range(maxit):
...         z = z**2 + c
...         diverge = z*np.conj(z) > 2**2           # who is diverging
...         div_now = diverge & (divtime==maxit)  # who is diverging now
...         divtime[div_now] = i                  # note when
...         z[diverge] = 2                        # avoid diverging too much
...
...     return divtime
>>> plt.imshow(mandelbrot(400,400))
>>> plt.show()
```



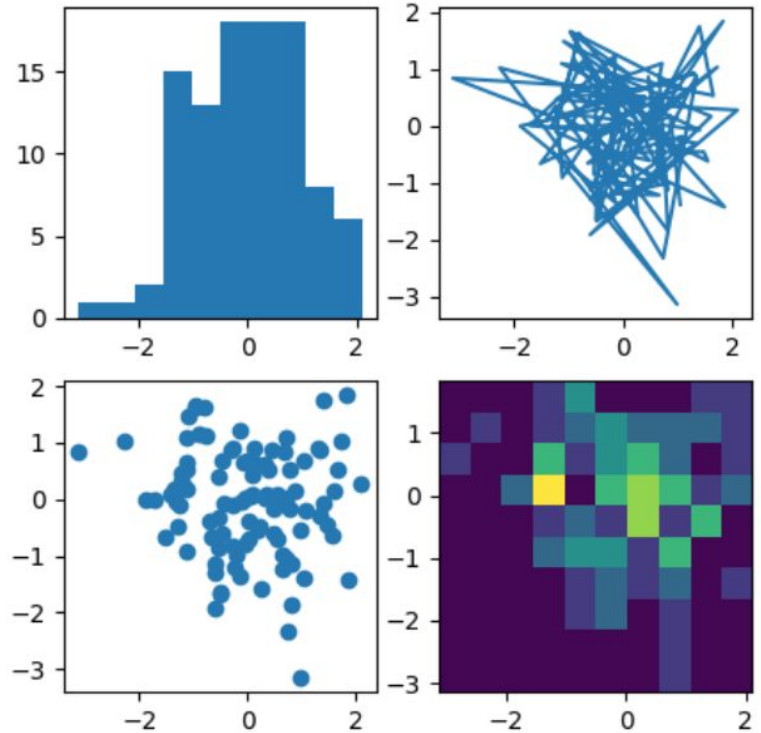
# Matplotlib

```
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(19680801)
data = np.random.randn(2, 100)

fig, axs = plt.subplots(2, 2, figsize=(5, 5))
axs[0, 0].hist(data[0])
axs[1, 0].scatter(data[0], data[1])
axs[0, 1].plot(data[0], data[1])
axs[1, 1].hist2d(data[0], data[1])

plt.show()
```

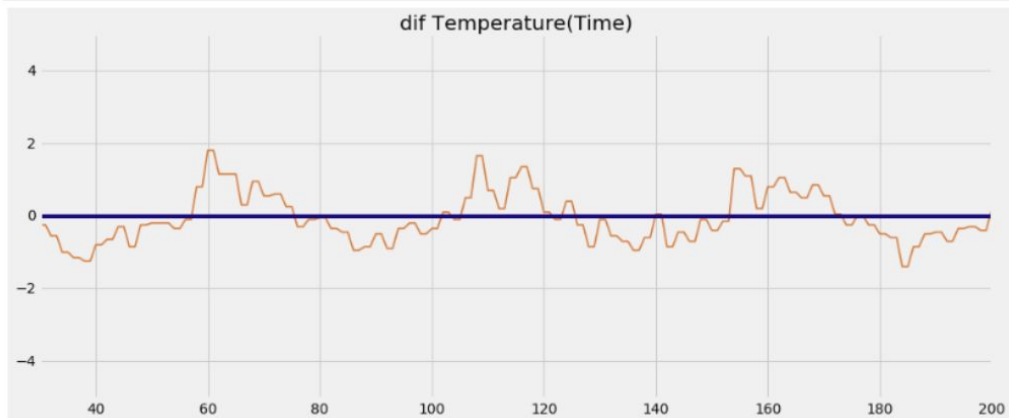


# Pandas

```
Test_Data.head(1)
```

	<b>Id</b>	<b>Consumption</b>	<b>Temperature</b>	<b>Time</b>	<b>DailySeasonality</b>	<b>WeeklySeasonality</b>
<b>5183</b>	5183	4317.386273	14.75	5183	47	191

```
plt.figure(figsize=(15, 6))
plt.plot(Learn_Data_Temp.Time, Learn_Data_Temp.Temperature,color='peru',linewidth=1.7)
c=Learn_Data_Temp.Time
plt.xlim(30,200)
plt.ylim(-5,5)
plt.plot(c,-9.88869197*10**(-7)*c-1.46170425*10**(-3), color='navy')
plt.title("dif Temperature(Time)")
plt.show()
```





# Pytorch

```
torch.empty(2,3)
```

```
tensor(1.00000e-31 *  
      [[ 7.5241,  0.0000,  7.5241],  
       [ 0.0000,  0.0000,  0.0000]])
```

```
torch.Tensor(2,3)
```

```
tensor([[ 7.5241e-31,  4.5569e-41,  7.5241e-31],  
       [ 4.5569e-41, -9.3542e-08,  4.5567e-41]])
```

```
torch.empty(2,3)
```

```
tensor([[ 7.5241e-31,  4.5569e-41,  7.5241e-31],  
       [ 4.5569e-41,  1.4013e-45,  4.5569e-41]])
```

```
torch.Tensor(2,3)
```

```
tensor(1.00000e-31 *  
      [[ 7.5241,  0.0000,  0.0000],  
       [ 0.0000,  0.0000,  0.0000]])
```

---

# Links

- <https://www.deeplearning.ai/machine-learning-yearning/>
- <https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4>
- <https://medium.com/usf-msds/choosing-the-right-metric-for-evaluating-machine-learning-models-part-2-86d5649a5428>
- <https://datascience.stackexchange.com/questions/15989/micro-average-vs-macro-average-performance-in-a-multiclass-classification-settin/16001>
- <https://scikit-learn.org/stable/modules/calibration.html>
- <https://habr.com/ru/post/451164/>
- <https://www.hse.ru/mirror/pubs/share/215285956>

# Темы для следующих занятий

- Kaggle
- Бустинг, ансамбли
- Кластеризация
- Рекомендательные системы, ранжирование
- NLP
- CV
- Временные ряды