ИТМО

Тренды искусственного интеллекта

Регрессия и классификация, нейронные сети





@SINECOR

Сергей Николенко

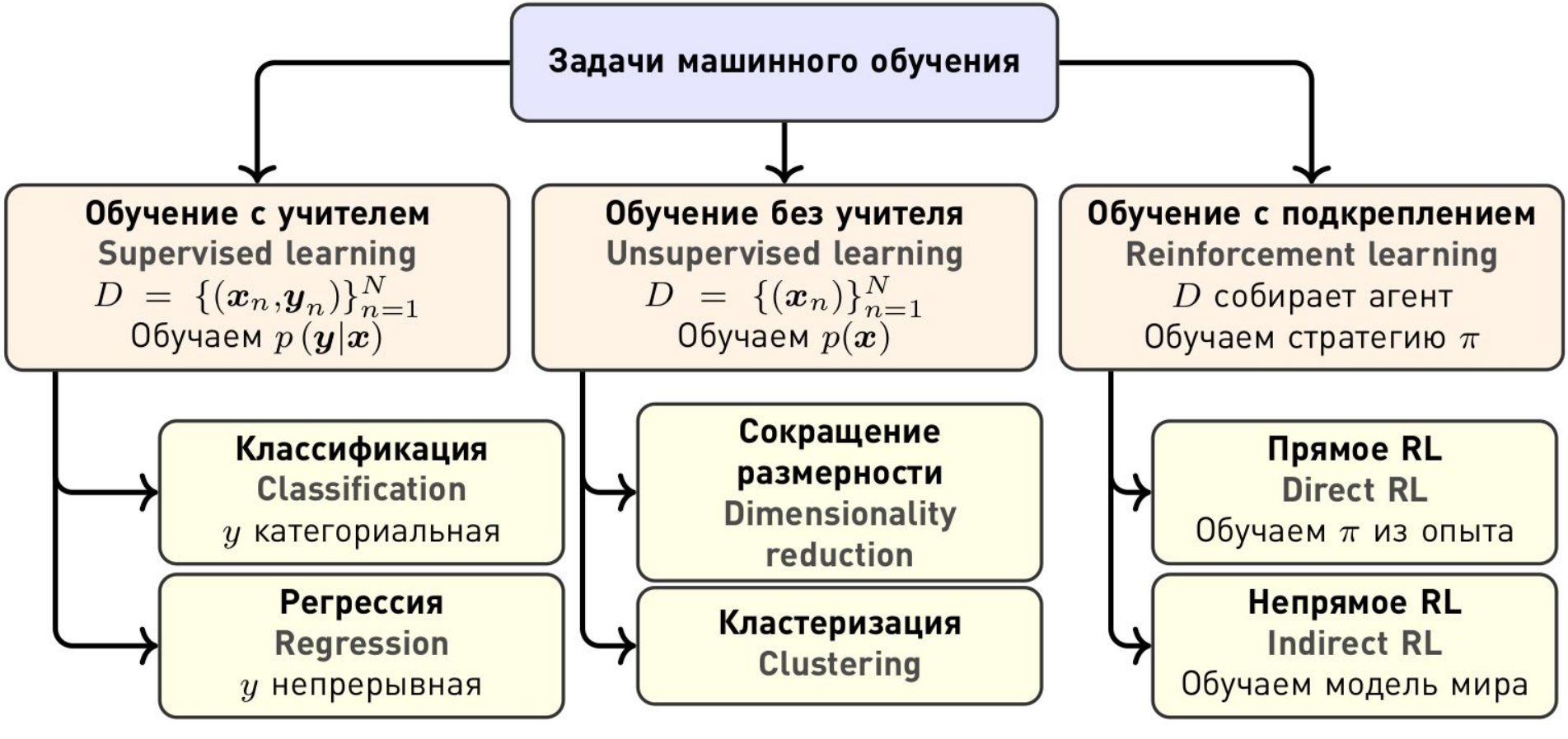
03 ноября 2026 г.

Линейная регрессия



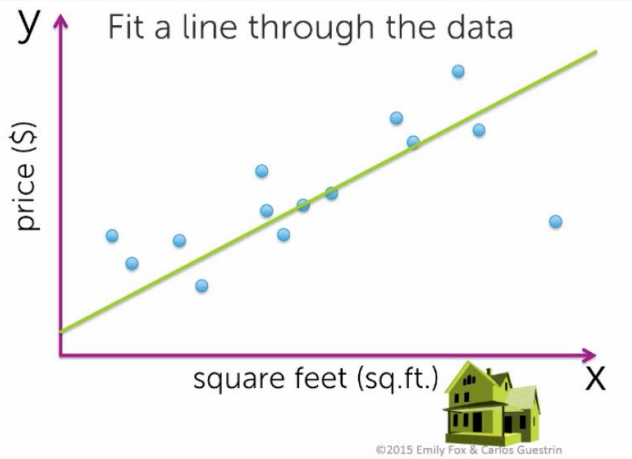
Виды постановок задач

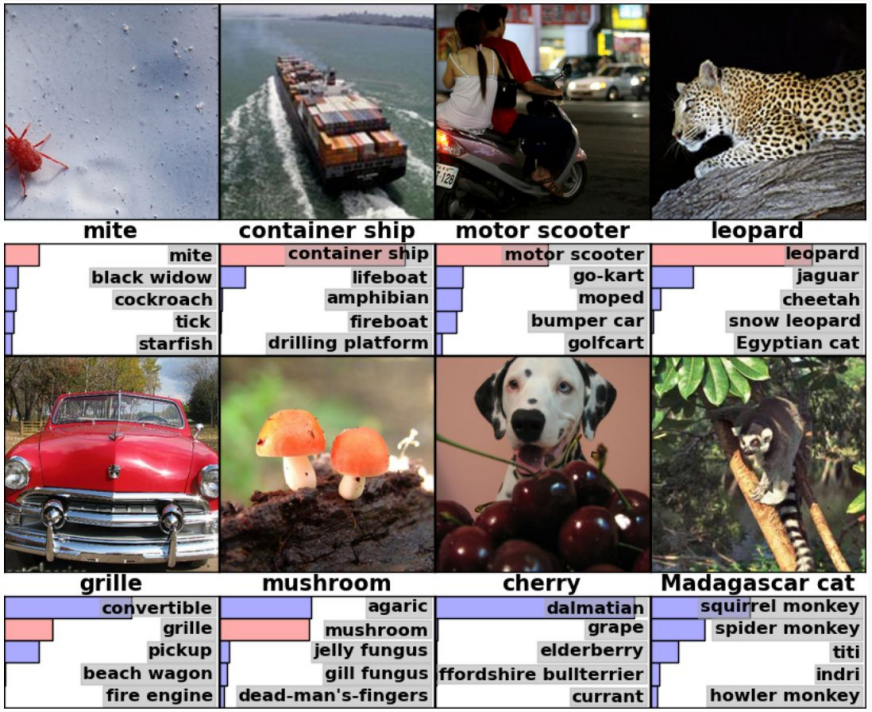
- На этой классификации давайте остановимся поподробнее...

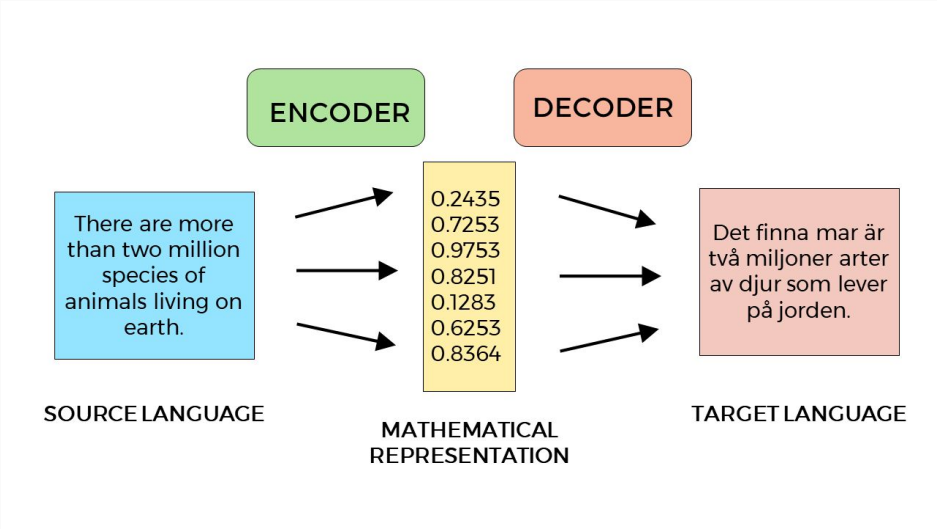


Обучение с учителем

- Примеры задач обучения с учителем:

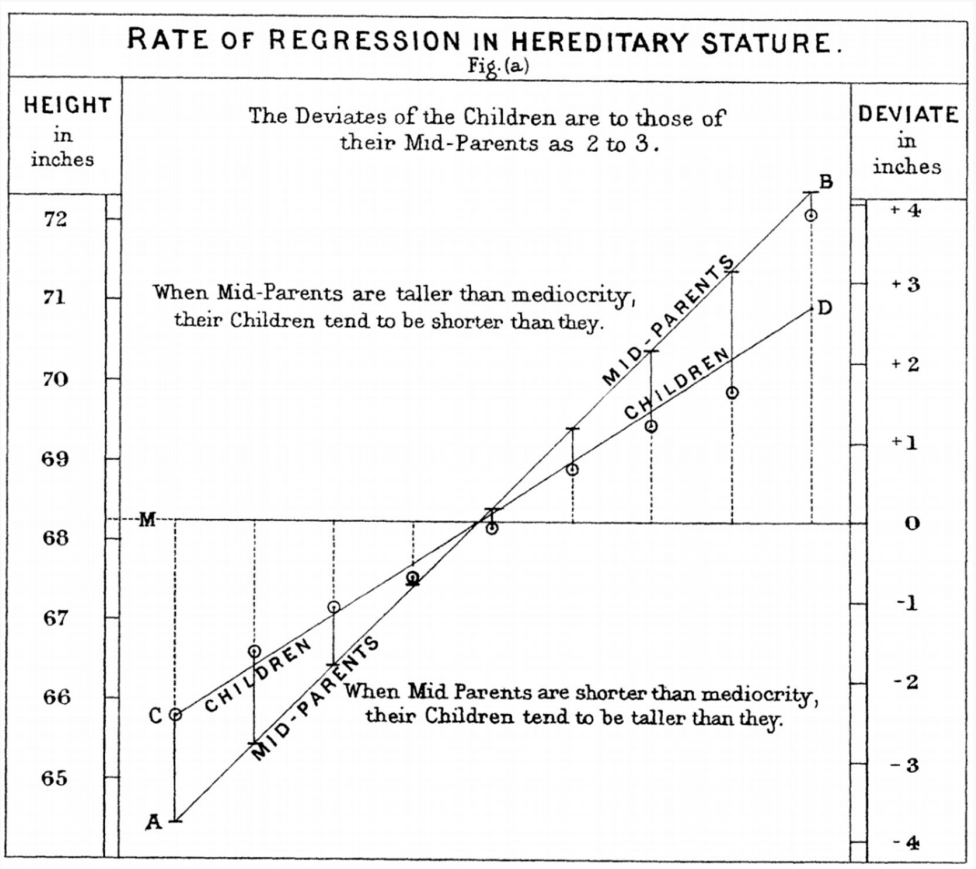


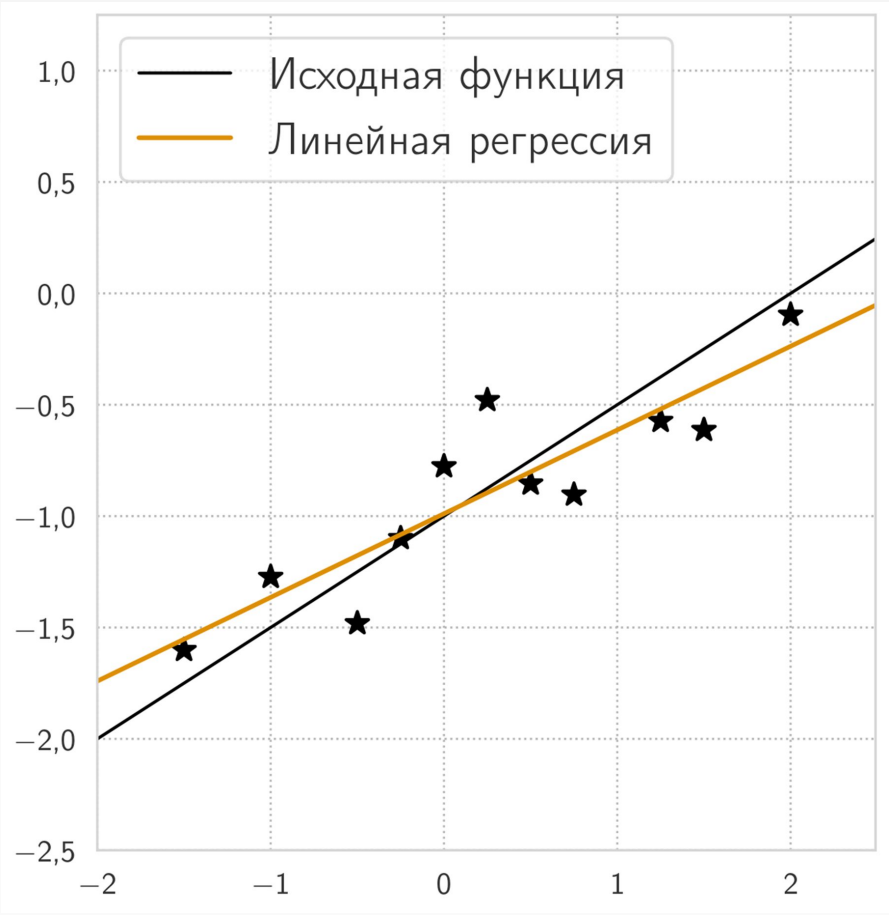




Кстати, почему “регрессия”?

- Почему регрессия называется регрессией? Странное название…





Линейная регрессия

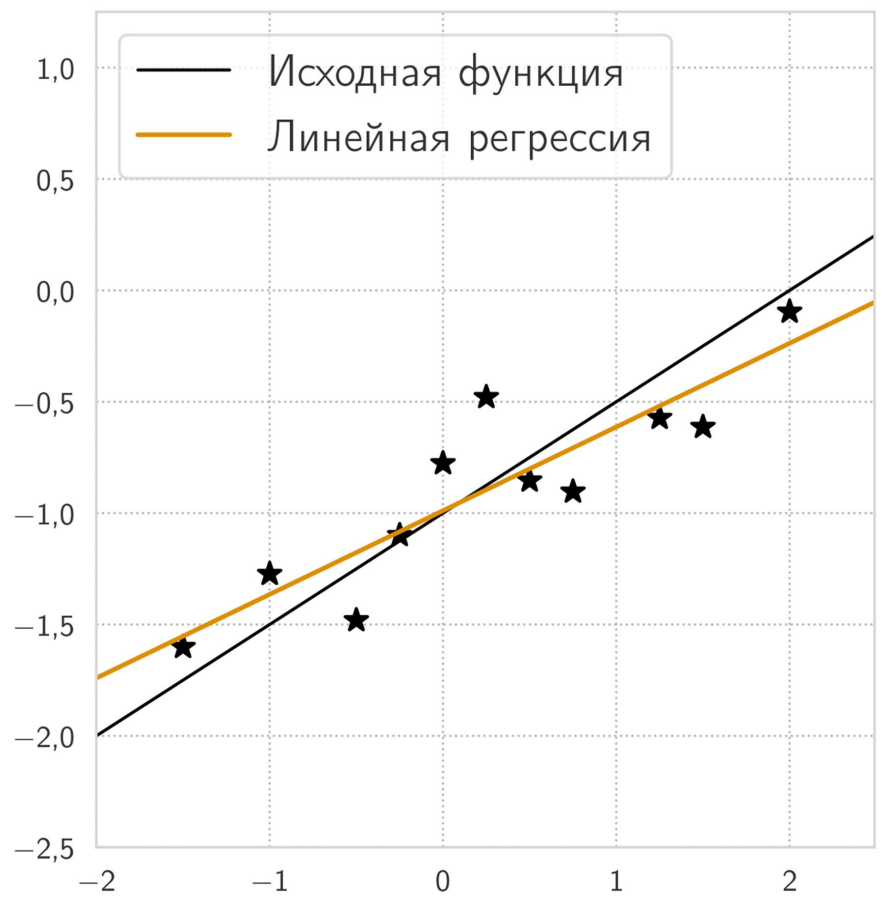
- Рассмотрим линейную функцию

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{p} x_j w_j = \mathbf{x}^\top \mathbf{w}$$

- По вектору входов **x** мы будем предсказывать выход *y* как

$$\hat{y}(\mathbf{x}) = \hat{w}_0 + \sum_{j=1}^{p} x_j \hat{w}_j = \mathbf{x}^\top \hat{\mathbf{w}}.$$

- Как найти оптимальные параметры **w** по данным?



Линейная регрессия

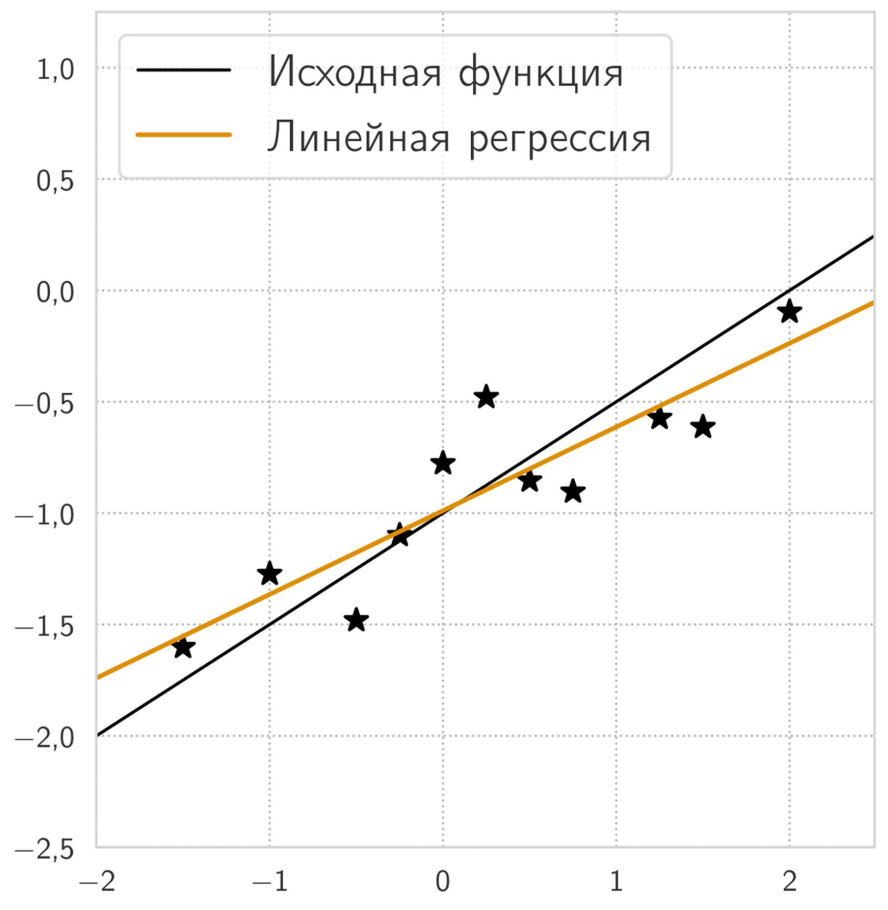
- Рассмотрим линейную функцию

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{p} x_j w_j = \mathbf{x}^\top \mathbf{w}$$

- По вектору входов **x** мы будем предсказывать выход *y* как

$$\hat{y}(\mathbf{x}) = \hat{w}_0 + \sum_{j=1}^{p} x_j \hat{w}_j = \mathbf{x}^\top \hat{\mathbf{w}}.$$

- Как найти оптимальные параметры **w** по данным?
- Можно минимизировать сумму квадратов отклонений…



Линейная регрессия

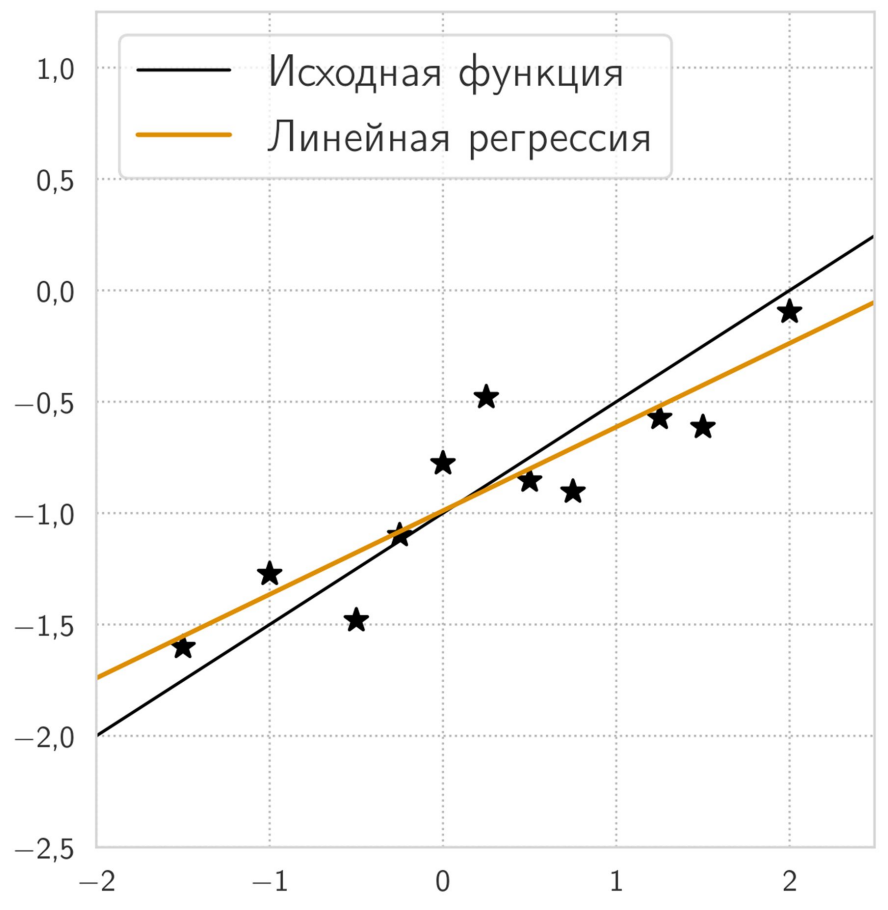
- Можно на самом деле решить задачу точно – записать как

$$\mathrm{RSS}(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w}),$$

продифференцировать, приравнять нулю, и получится

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

- Это псевдообратная Мура-Пенроуза (Moore-Penrose pseudoinverse)



Но почему квадрат?

- Но откуда берётся метод наименьших квадратов?
- Почему квадрат?



Но почему квадрат?

- Но откуда берётся метод наименьших квадратов?
- Почему квадрат?
- Основное наше предположение – в том, что шум (ошибка в данных) распределён *нормально*, т.е.

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon, \quad \epsilon \sim N(0, \sigma^2).$$

- Иначе говоря,

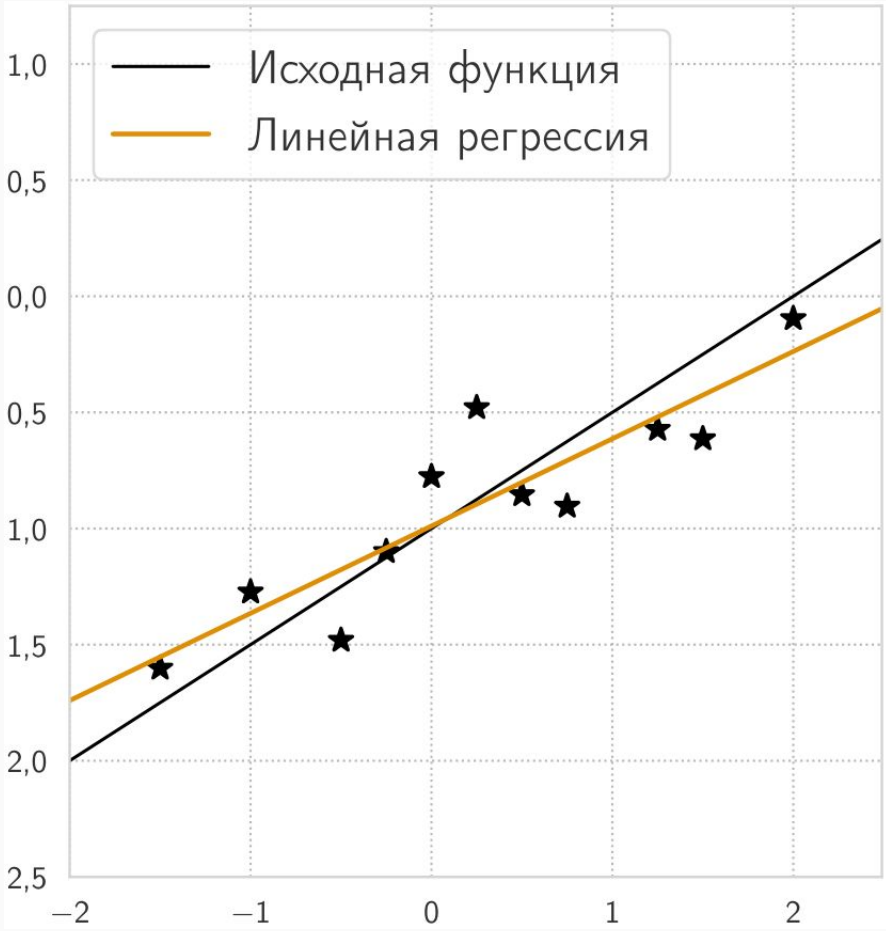
$$p(t \mid \mathbf{x}, \mathbf{w}, \sigma^2) = N(t \mid y(\mathbf{x}, \mathbf{w}), \sigma^2).$$

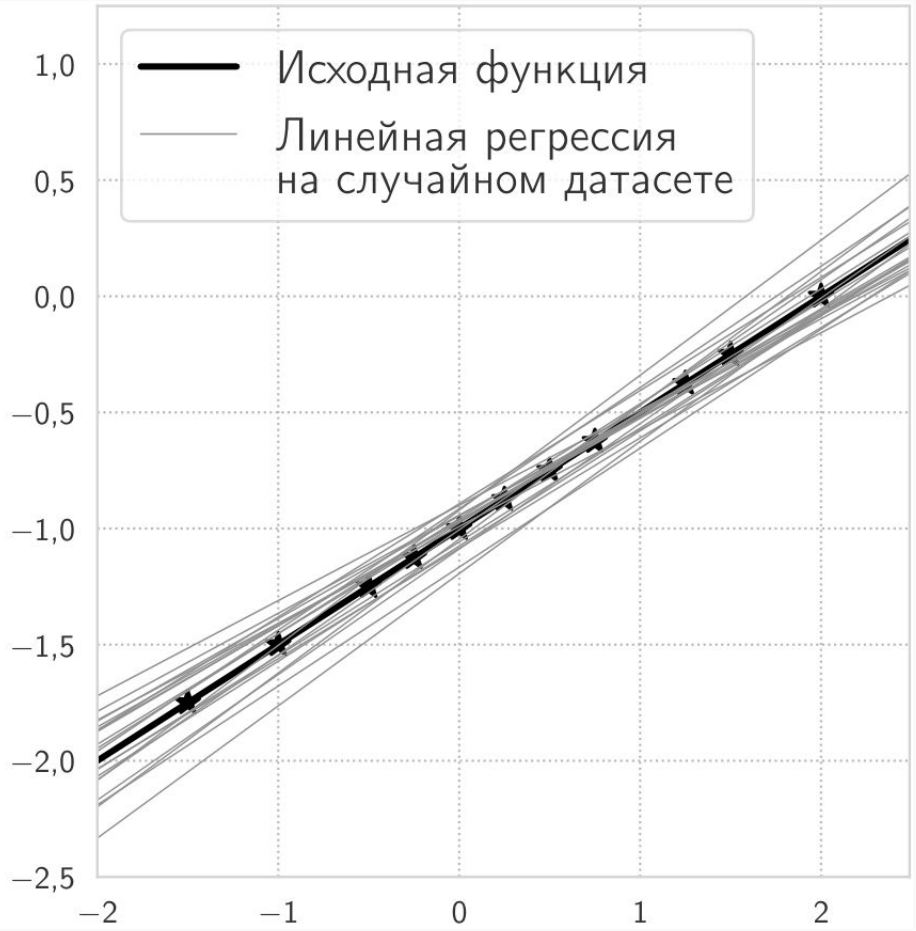
- И тогда...



Линейная регрессия по-байесовски

- Если набрасывать датасеты согласно нашим предположениям, то в среднем будет получаться то, что надо — но именно в среднем!



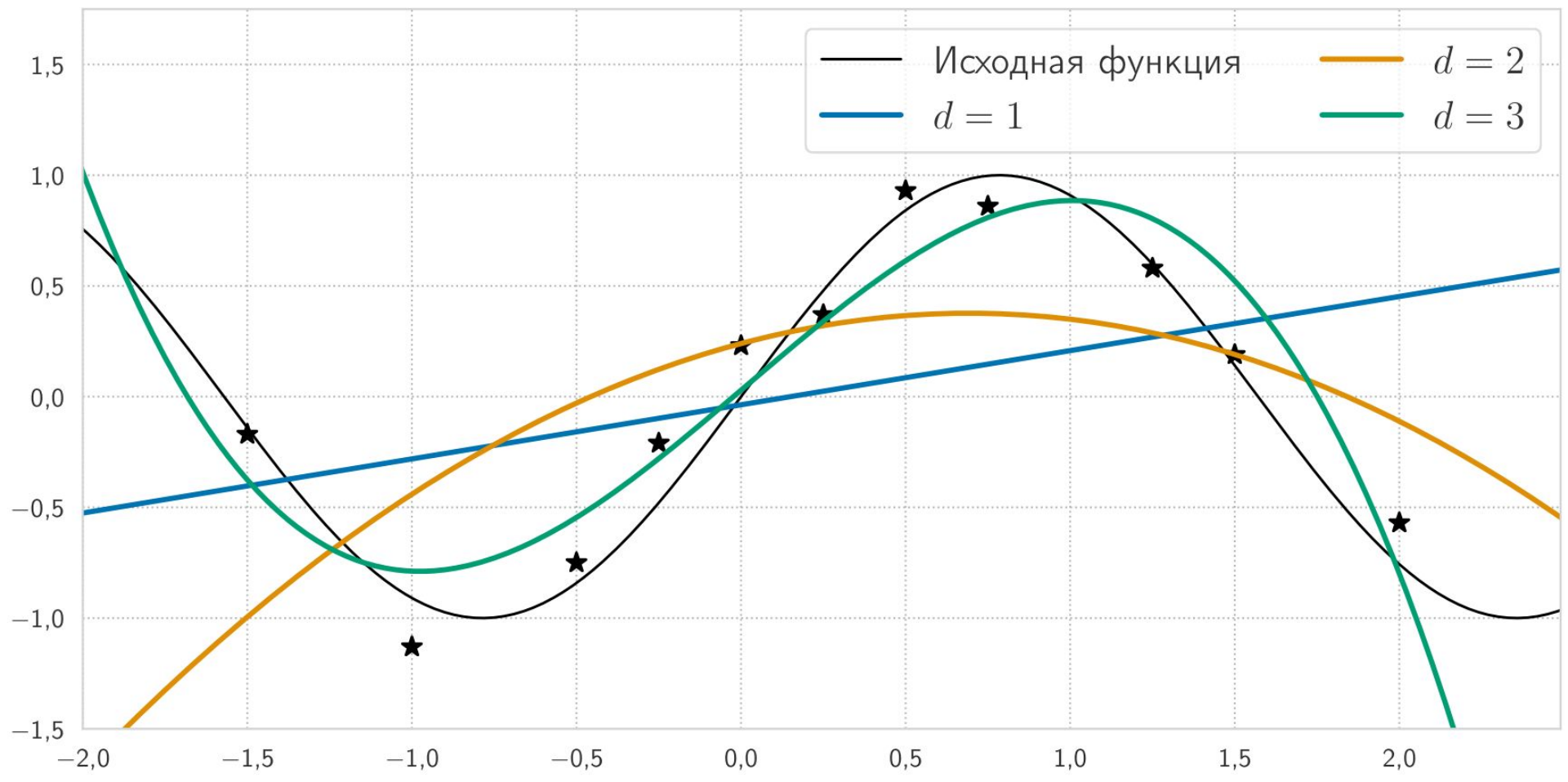


Признаки, оверфиттинг, регуляризация



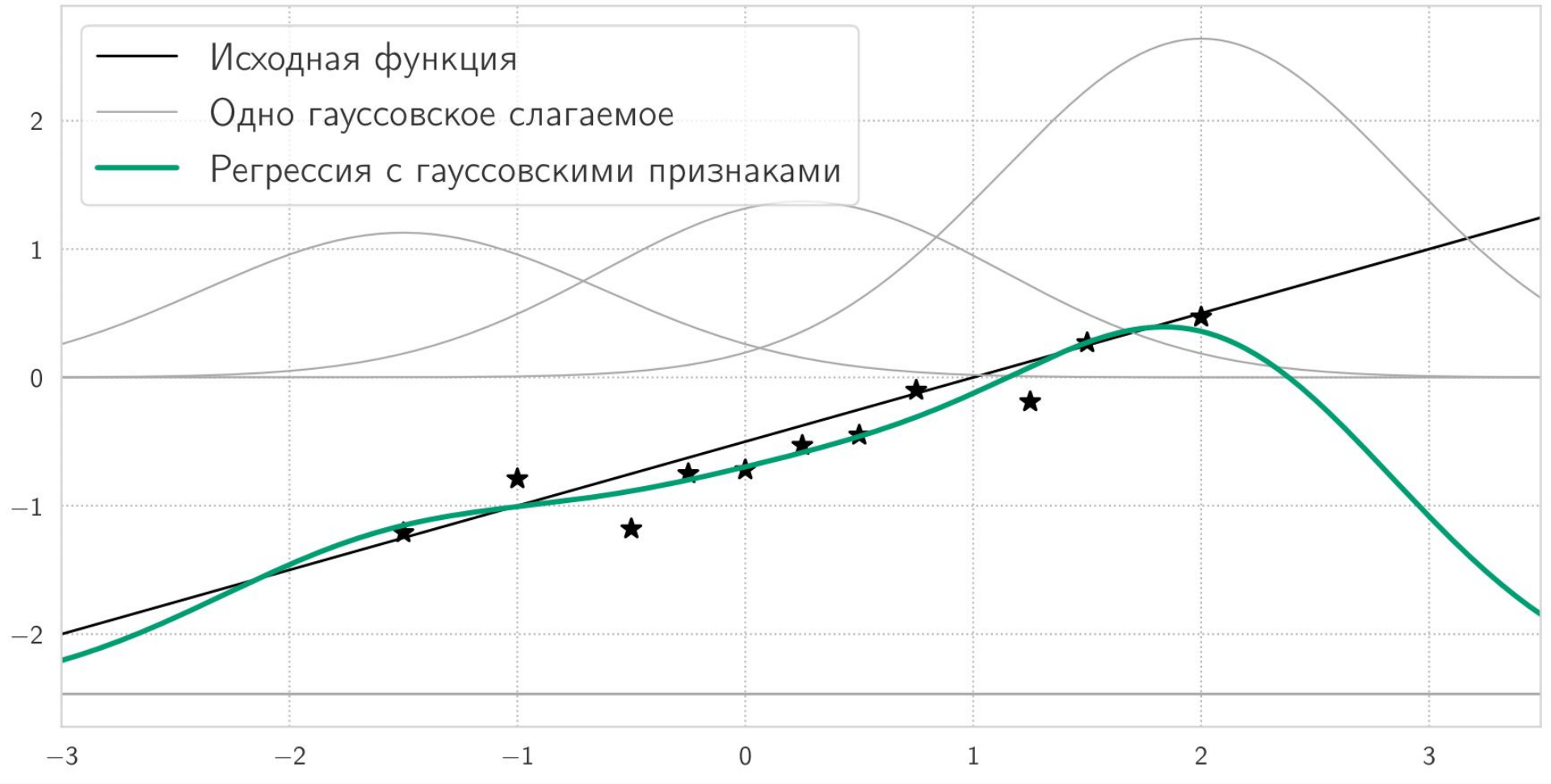
Линейная регрессия с признаками

- К линейной регрессии легко добавить какие угодно признаки:



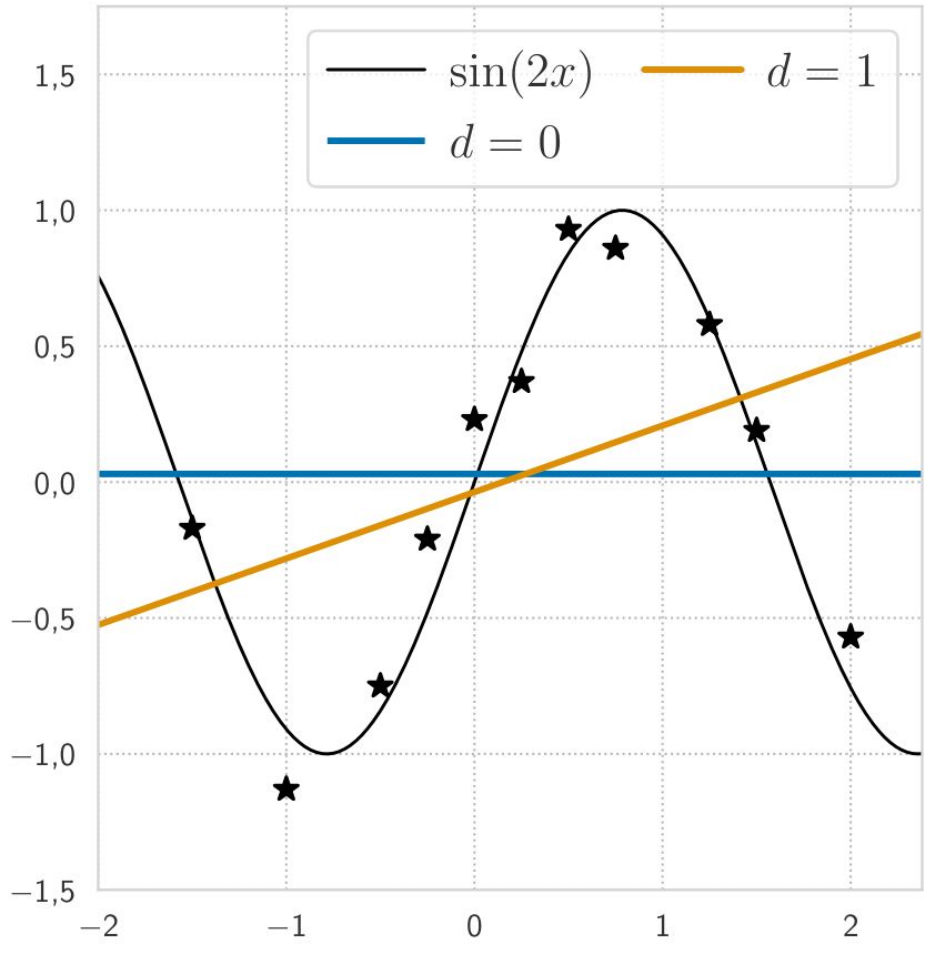
Линейная регрессия с признаками

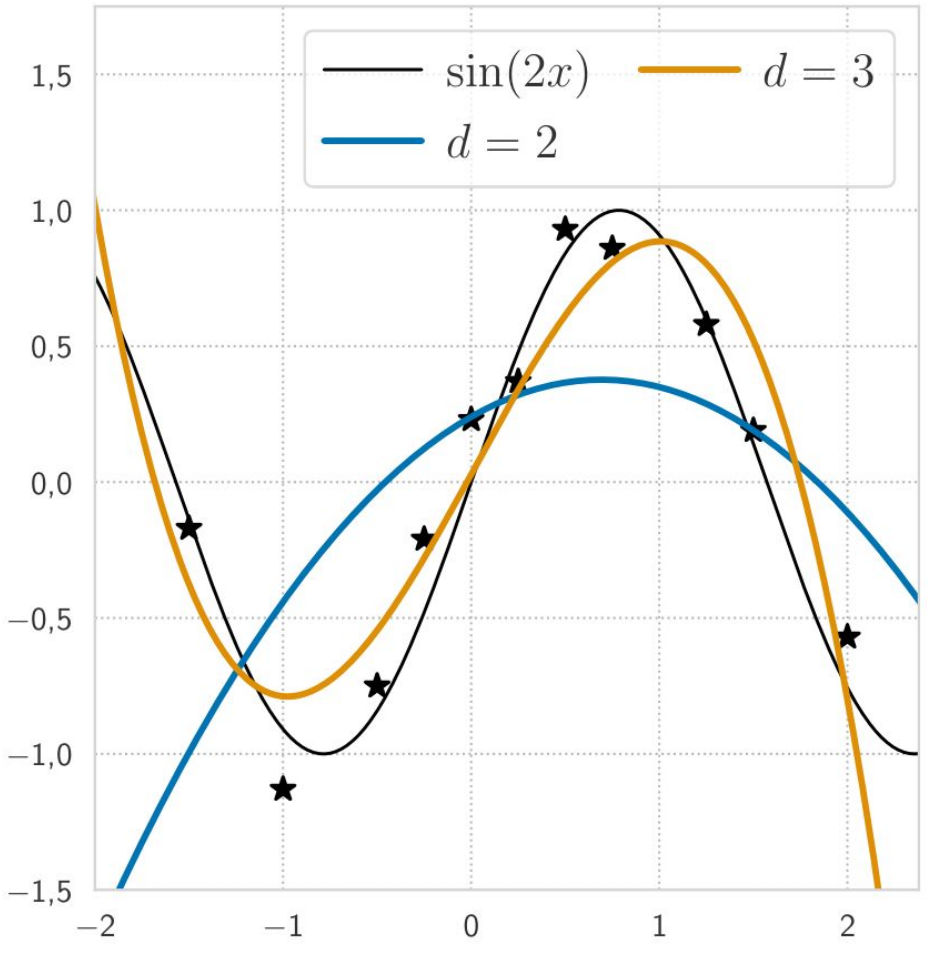
- К линейной регрессии легко добавить какие угодно признаки:



Линейная регрессия с признаками

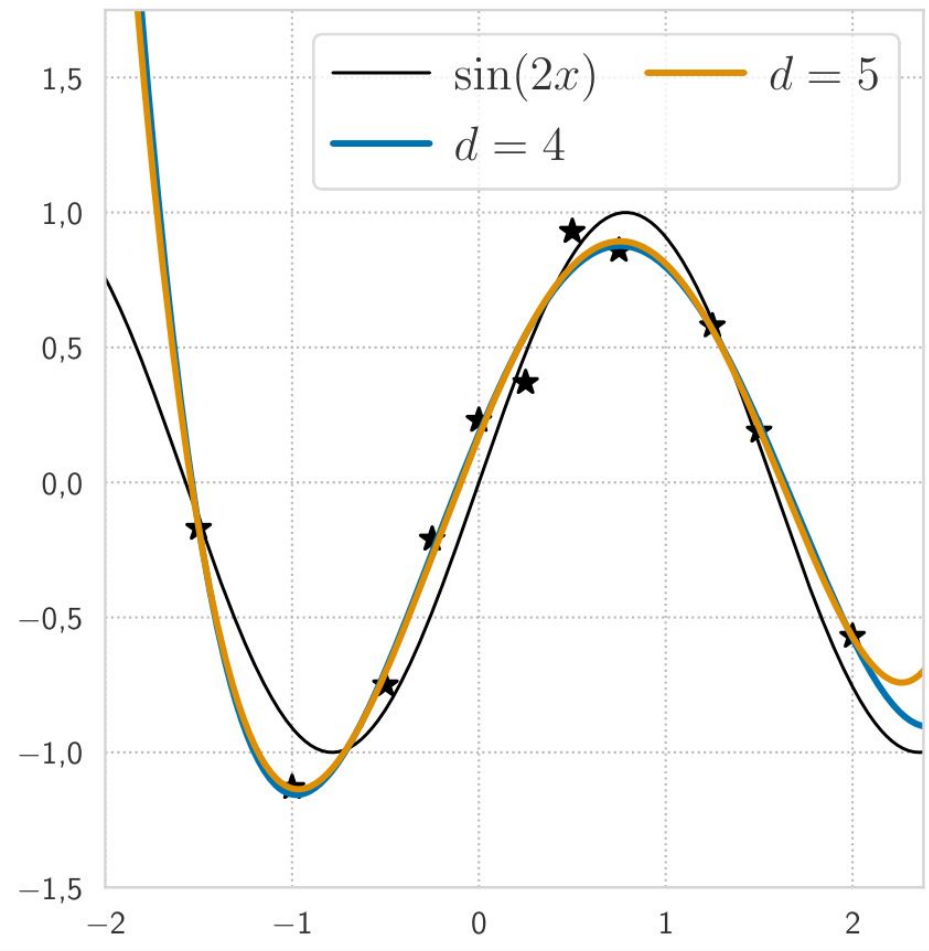
- Но новые признаки — это хорошо только до какого-то момента:

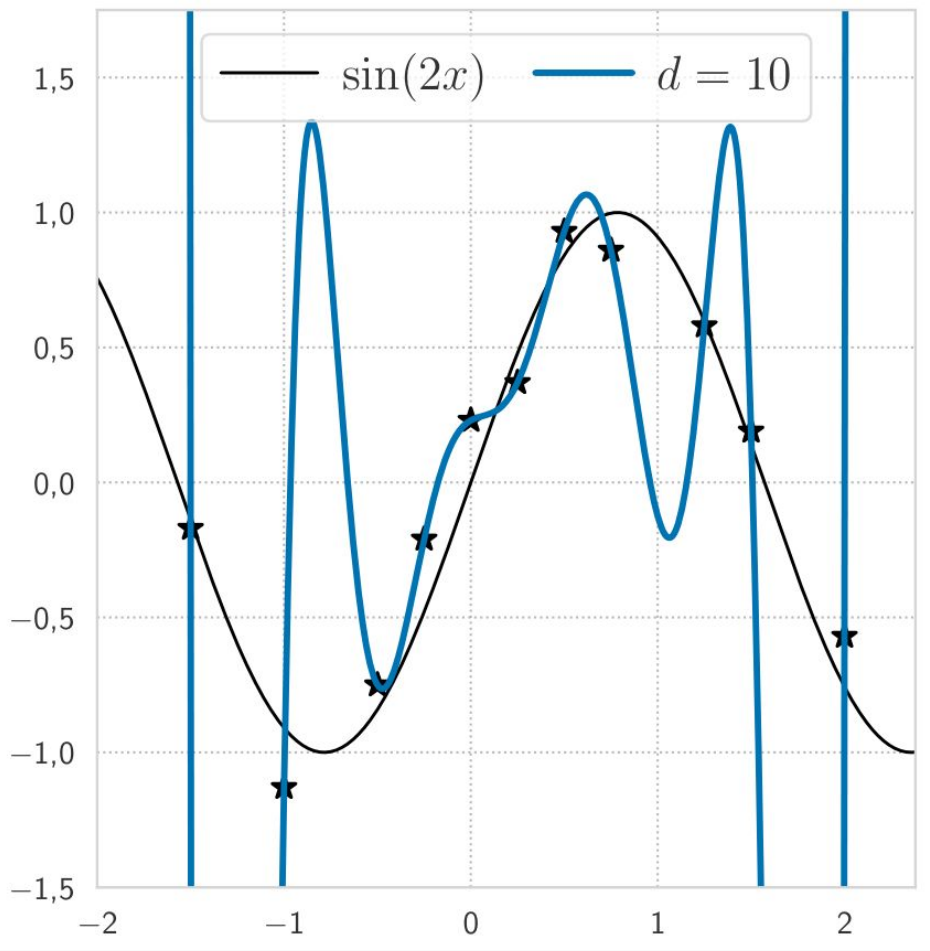




Линейная регрессия с признаками

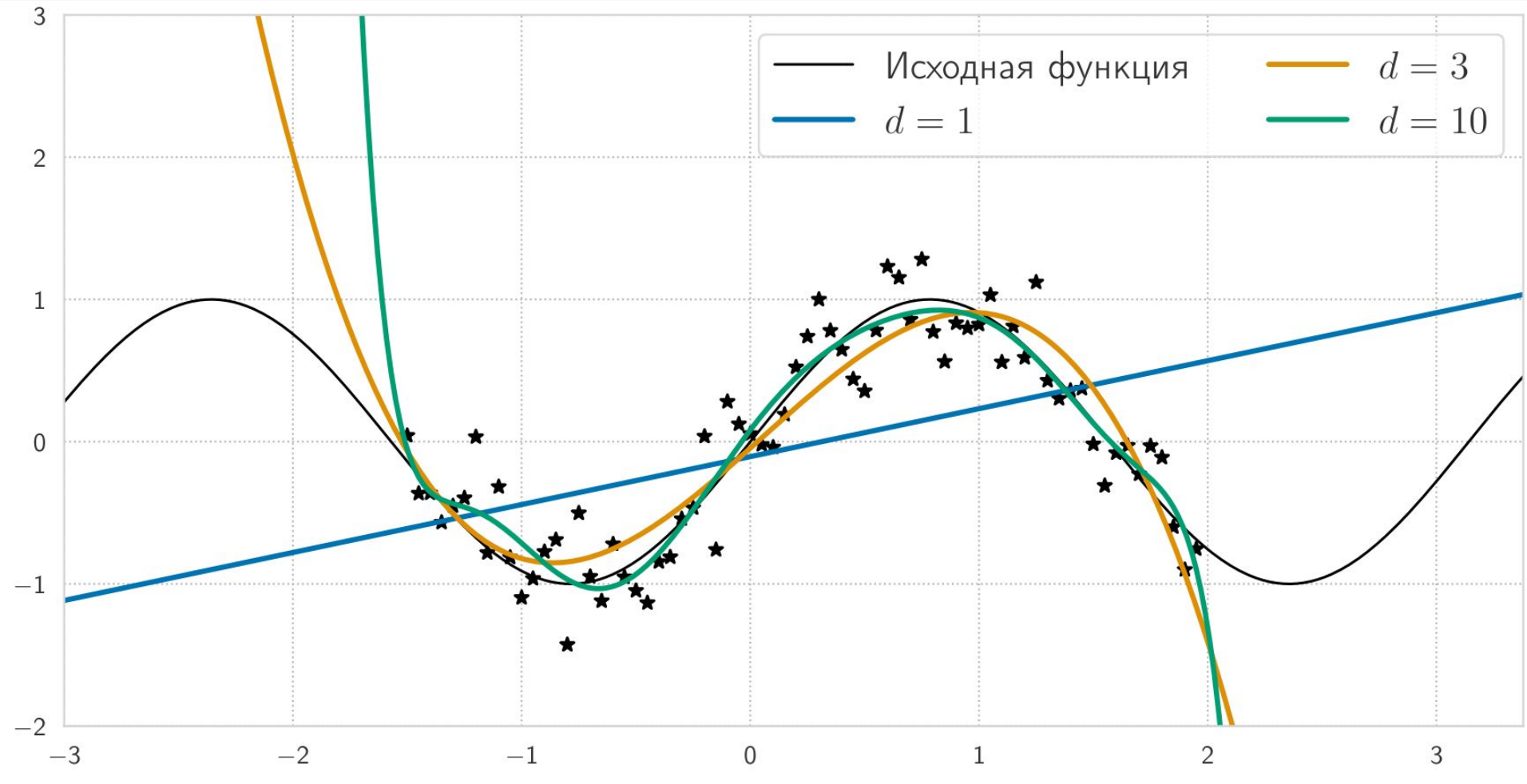
- А потом может начаться жёсткий оверфиттинг:





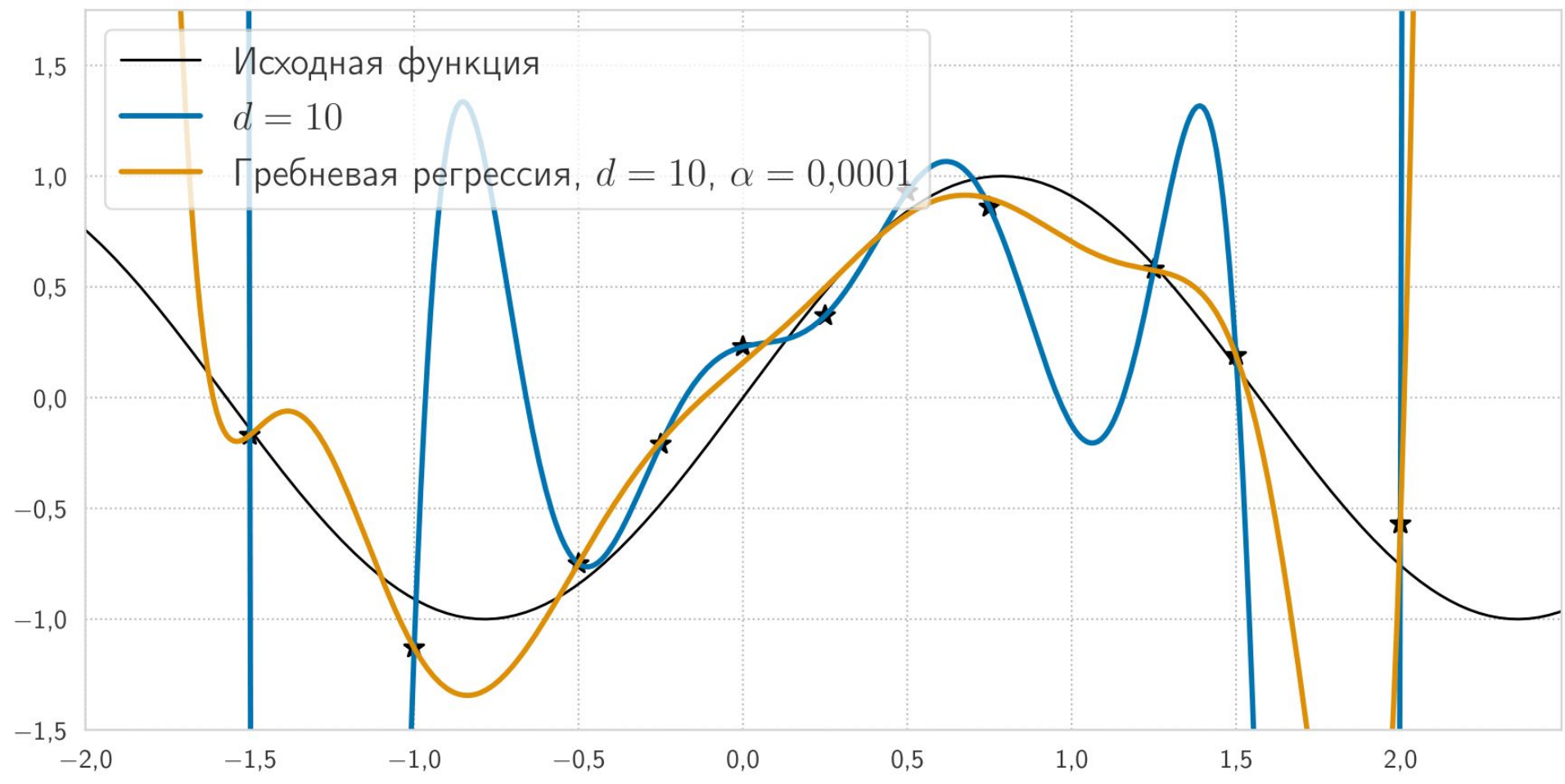
Линейная регрессия с признаками

- Это, конечно, зависит от количества имеющихся данных:



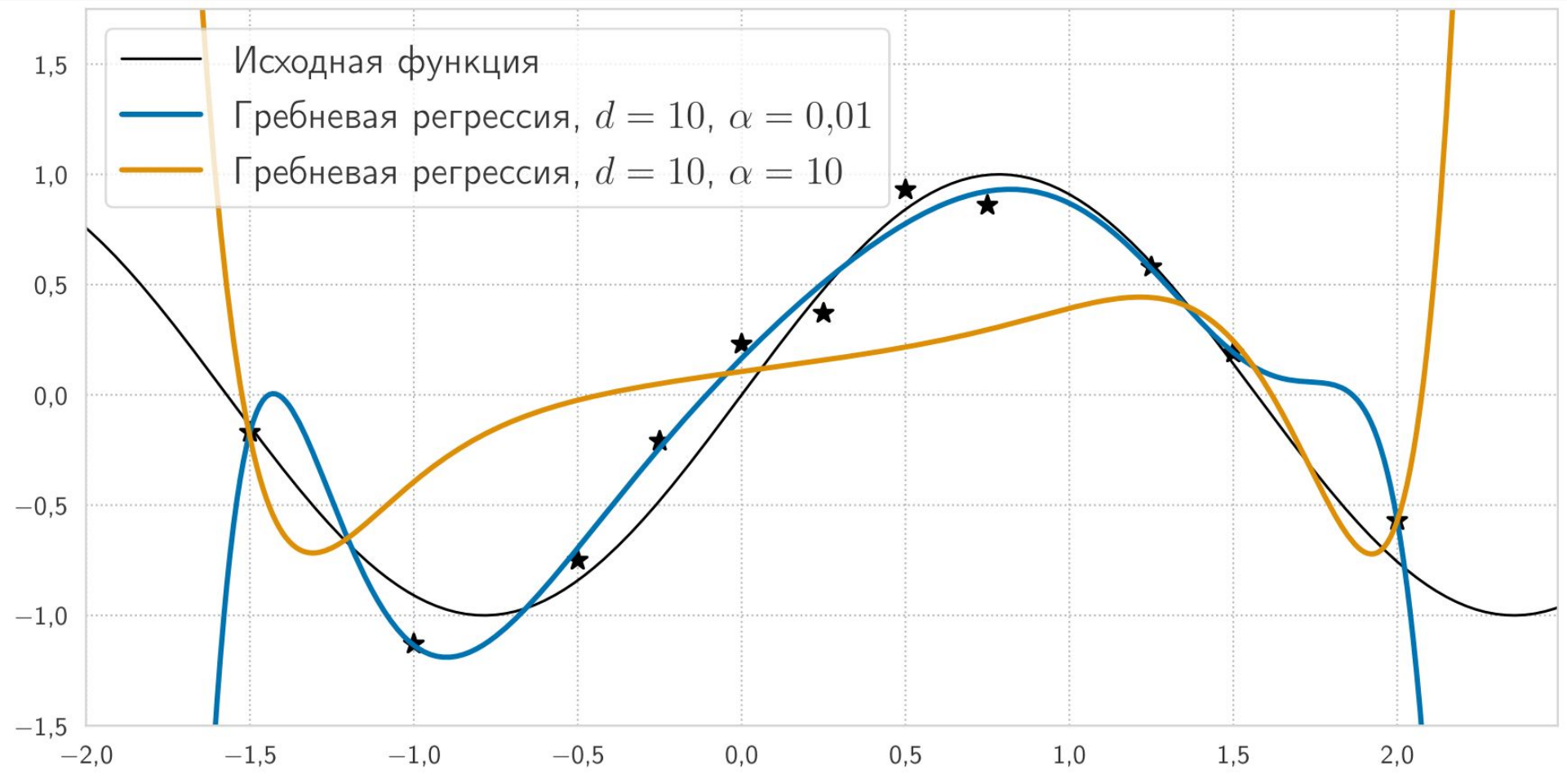
Регуляризация в линейной регрессии

- Тогда надо регуляризовать — например, гребневая регрессия:



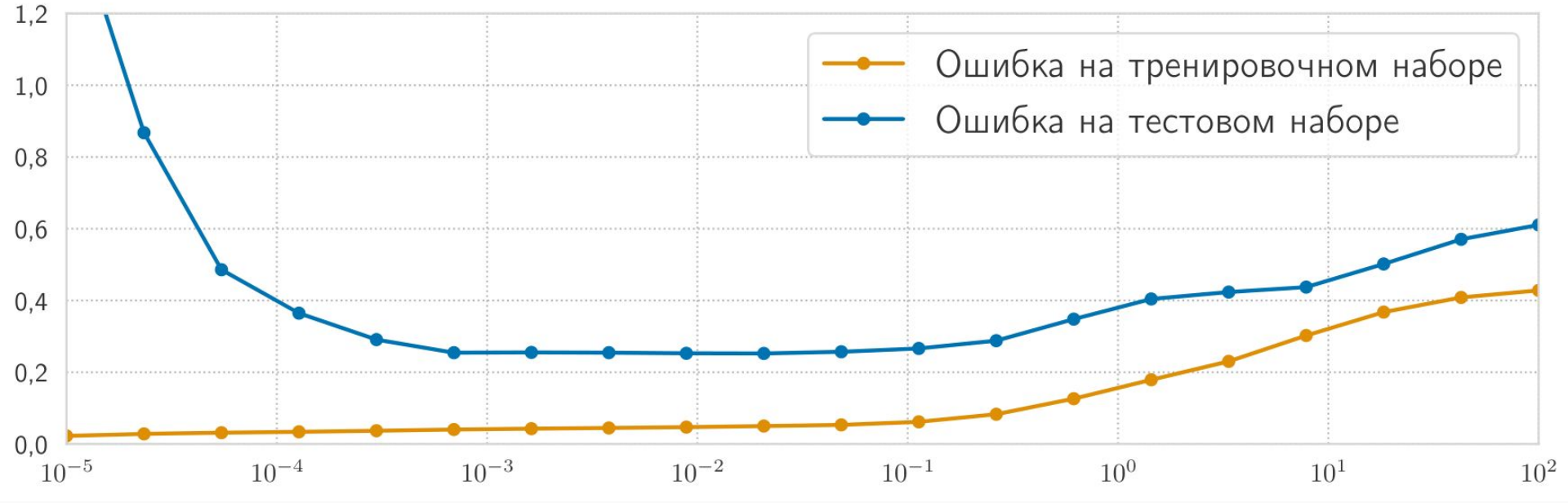
Регуляризация в линейной регрессии

- Тогда надо регуляризовать — например, гребневая регрессия:



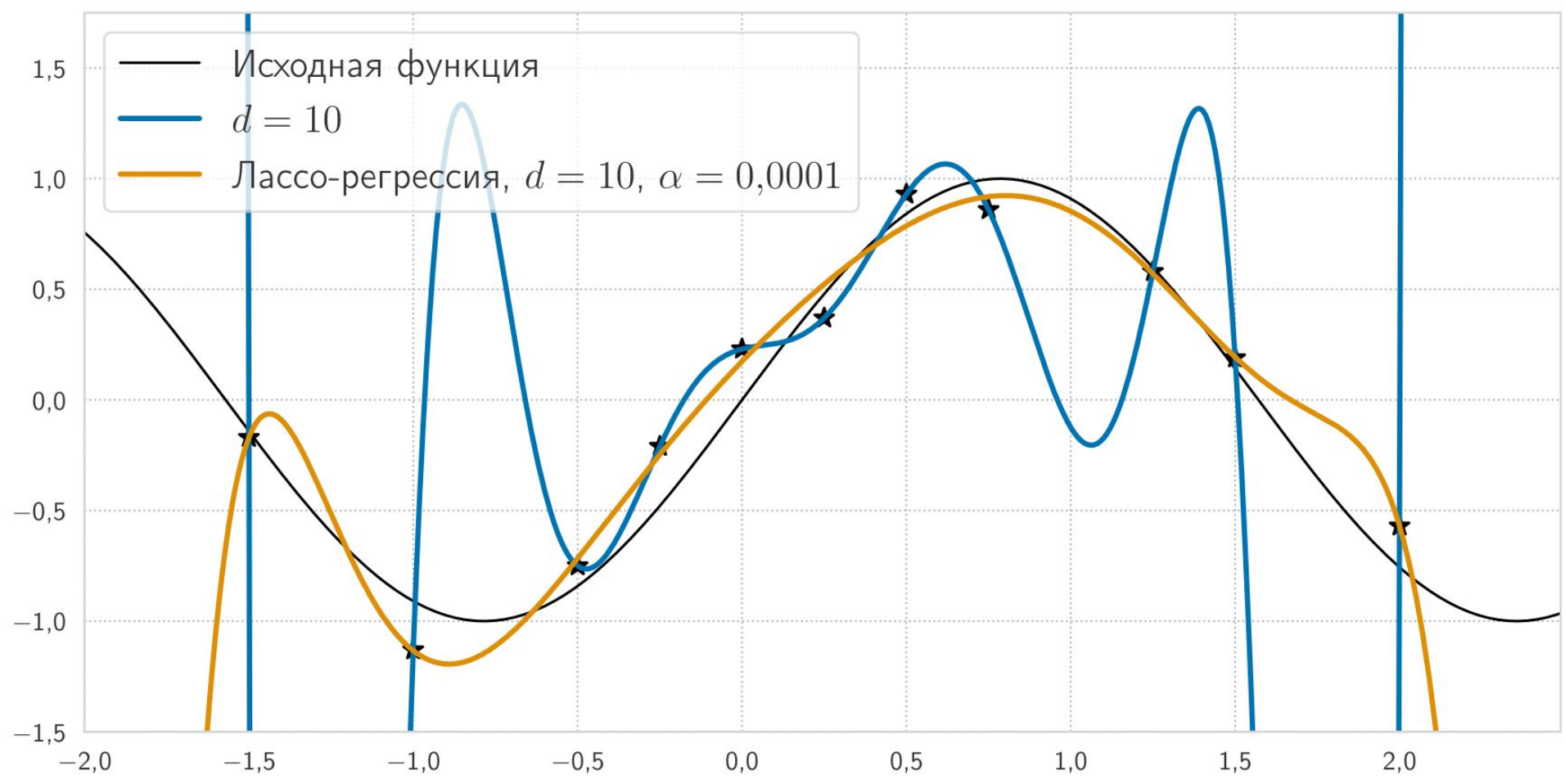
Регуляризация в линейной регрессии

- Коэффициент регуляризации надо подбирать — у него, скорее всего, есть оптимальный регион, в который желательно попасть
- Это целая наука, в которую мы не будем углубляться, но здесь и статистическая теория принятия решений (с bias-variance-noise decomposition), и байесовский выбор моделей (с BIC и AIC)...



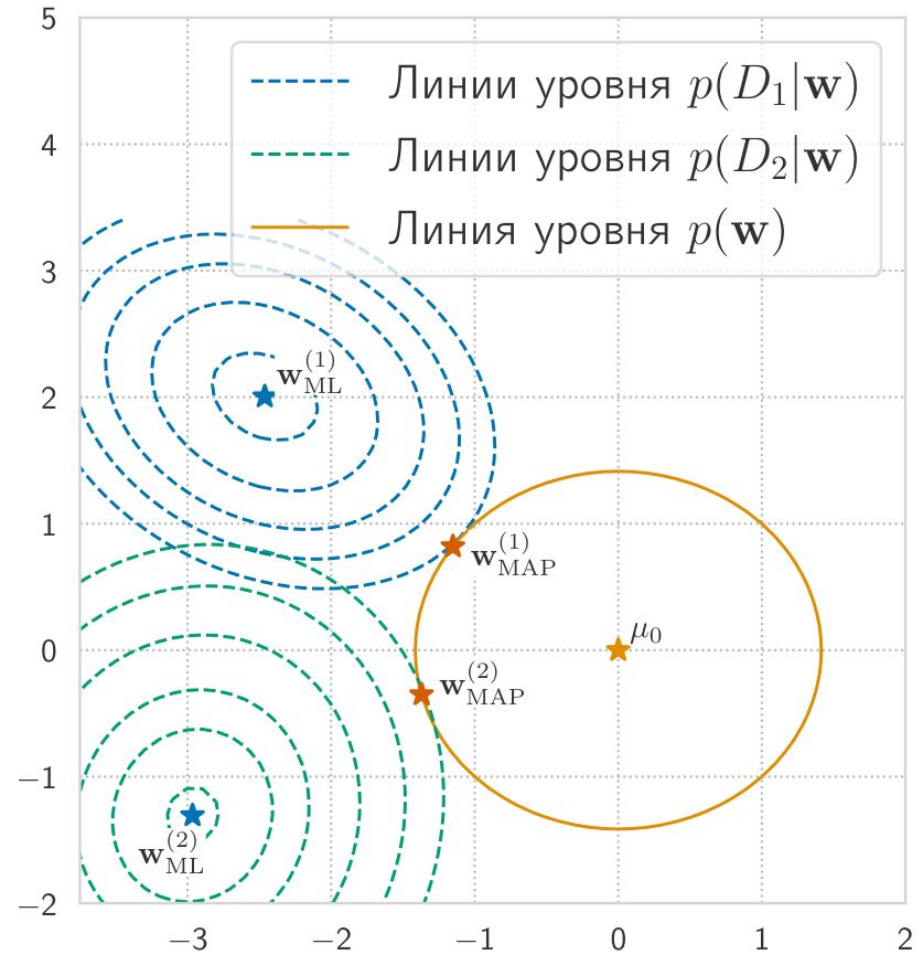
Регуляризация в линейной регрессии

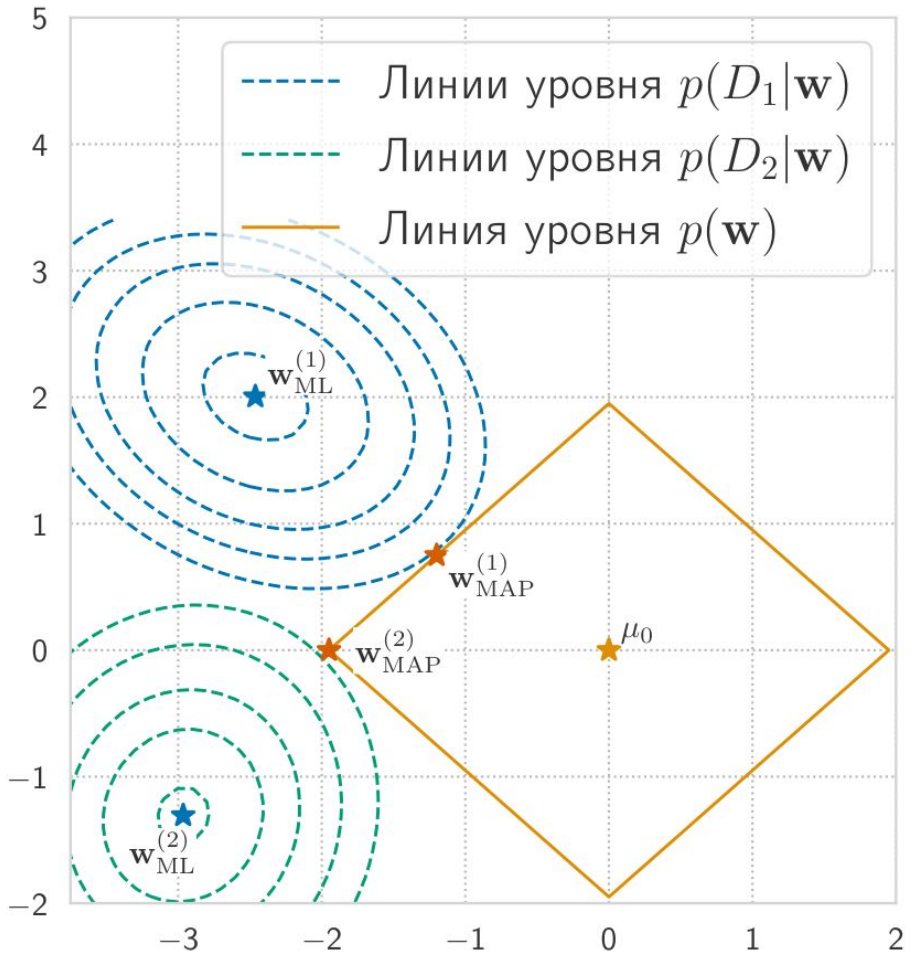
- Или лассо-регрессия. А в чём, кстати, разница?..



Регуляризация в линейной регрессии

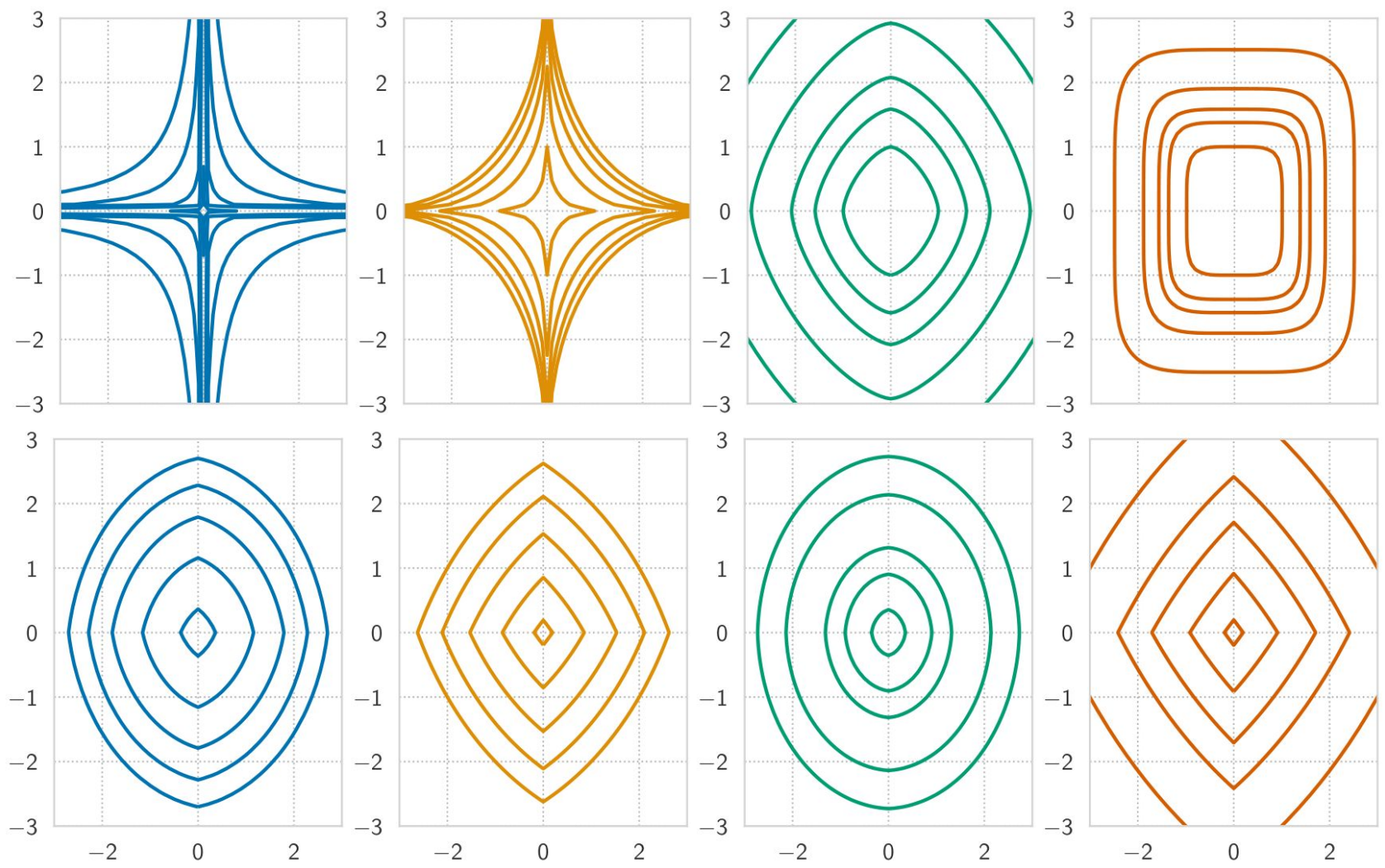
- Геометрический смысл гребневой и лассо-регрессии:





Регуляризация в линейной регрессии

- Можно, конечно, и L_p-регуляризацию рассмотреть, и elastic net:

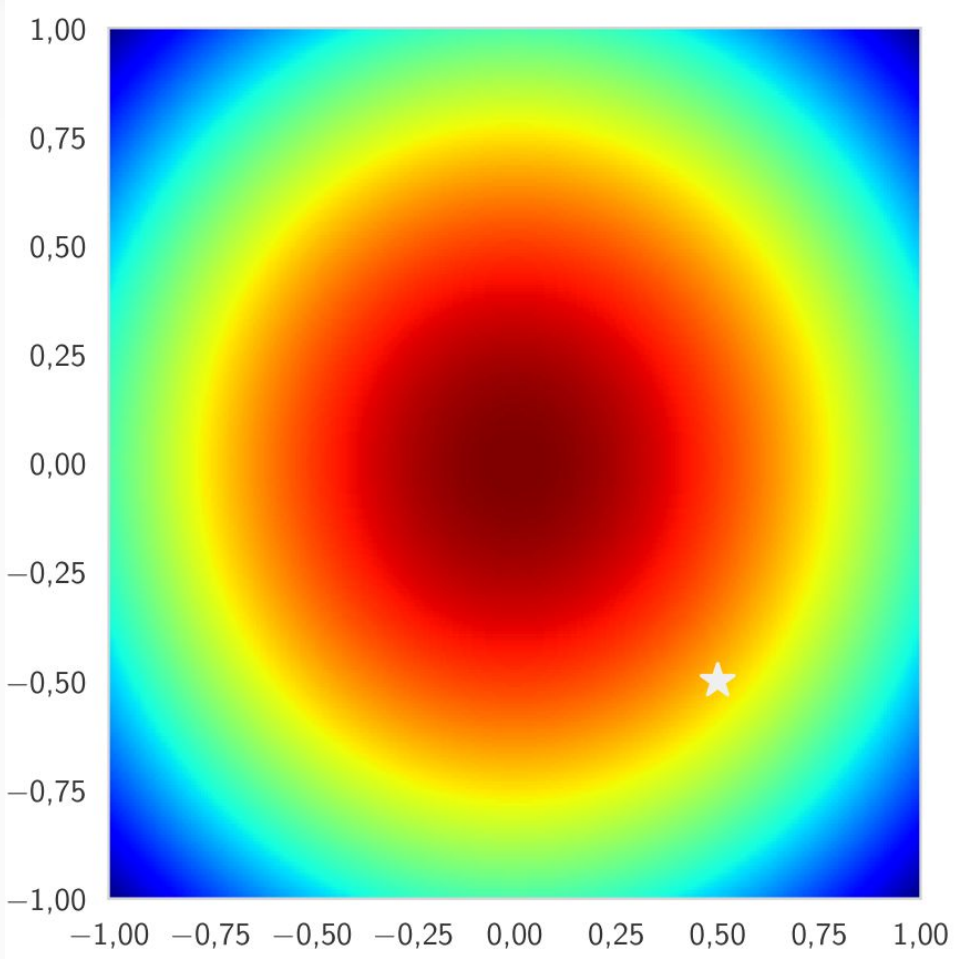


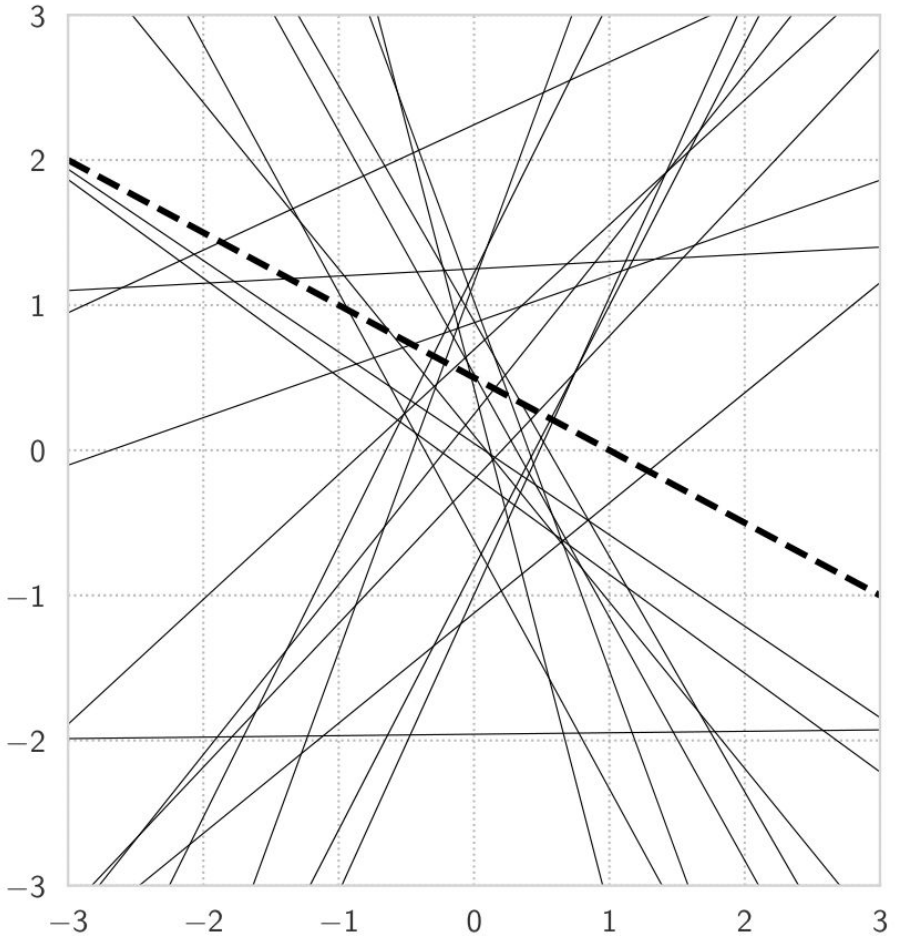
Линейная регрессия по-байесовски



Линейная регрессия по-байесовски

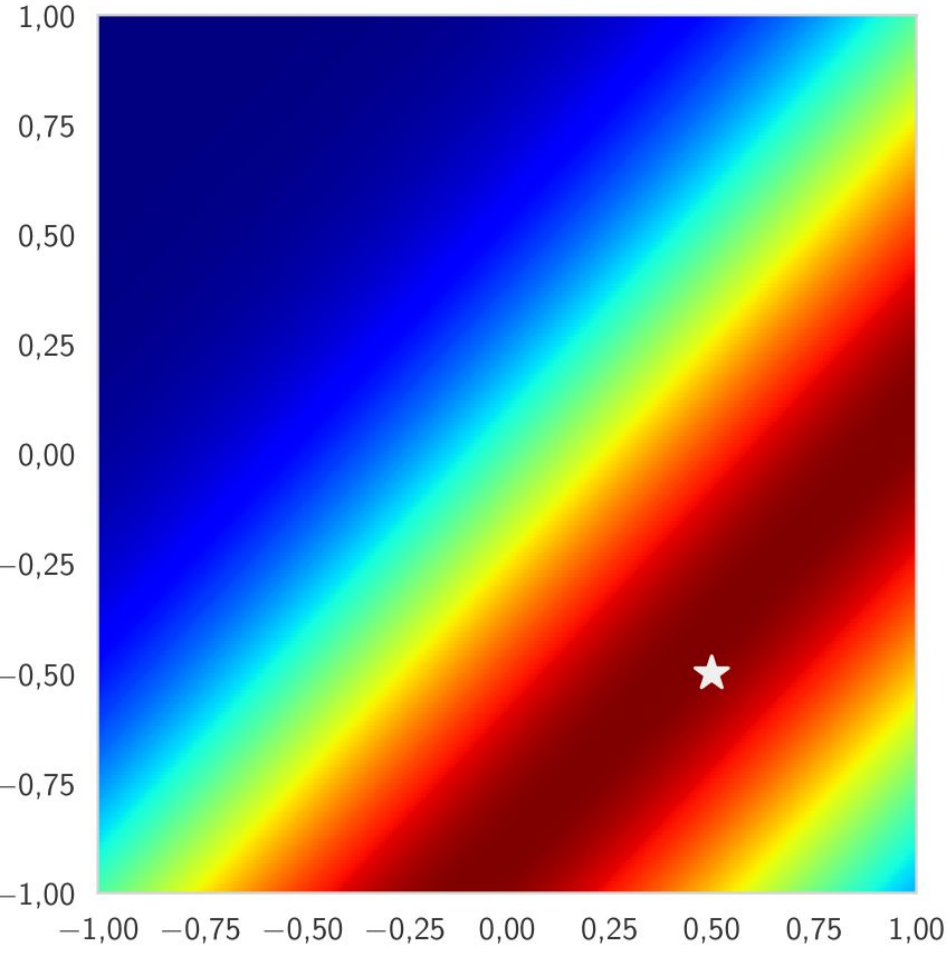
- Давайте проследим за байесовским выводом…

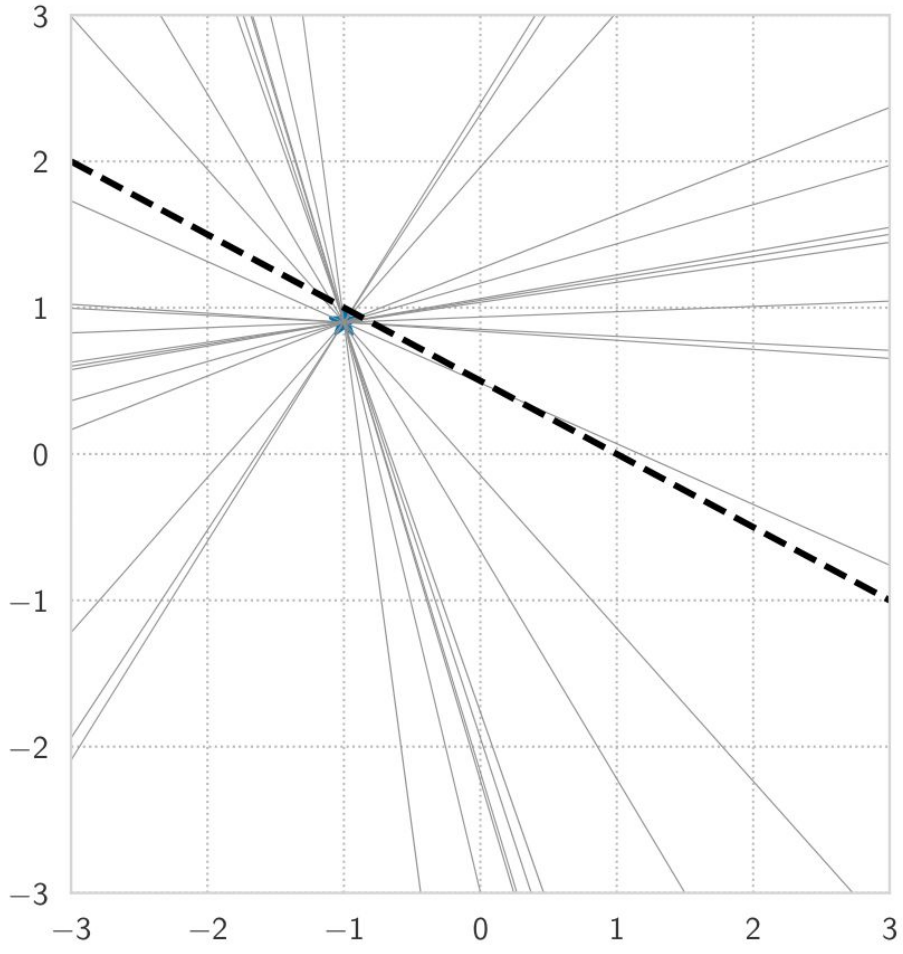




Линейная регрессия по-байесовски

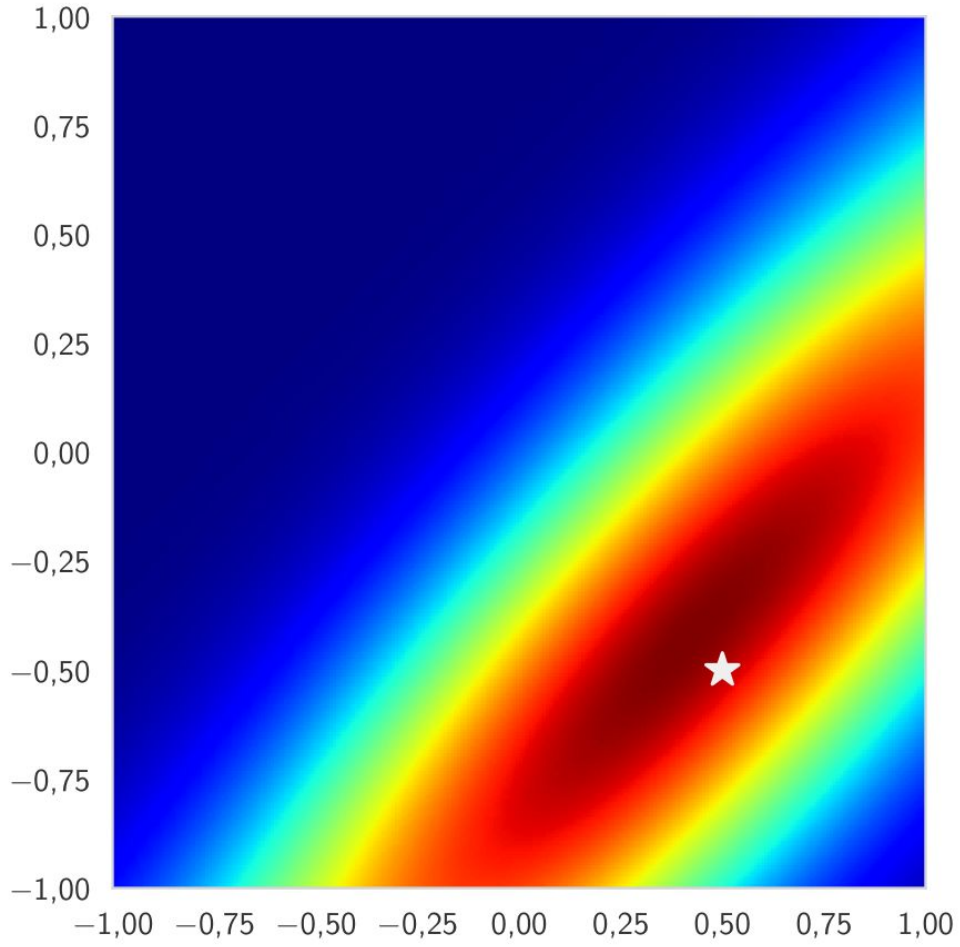
- Давайте проследим за байесовским выводом…

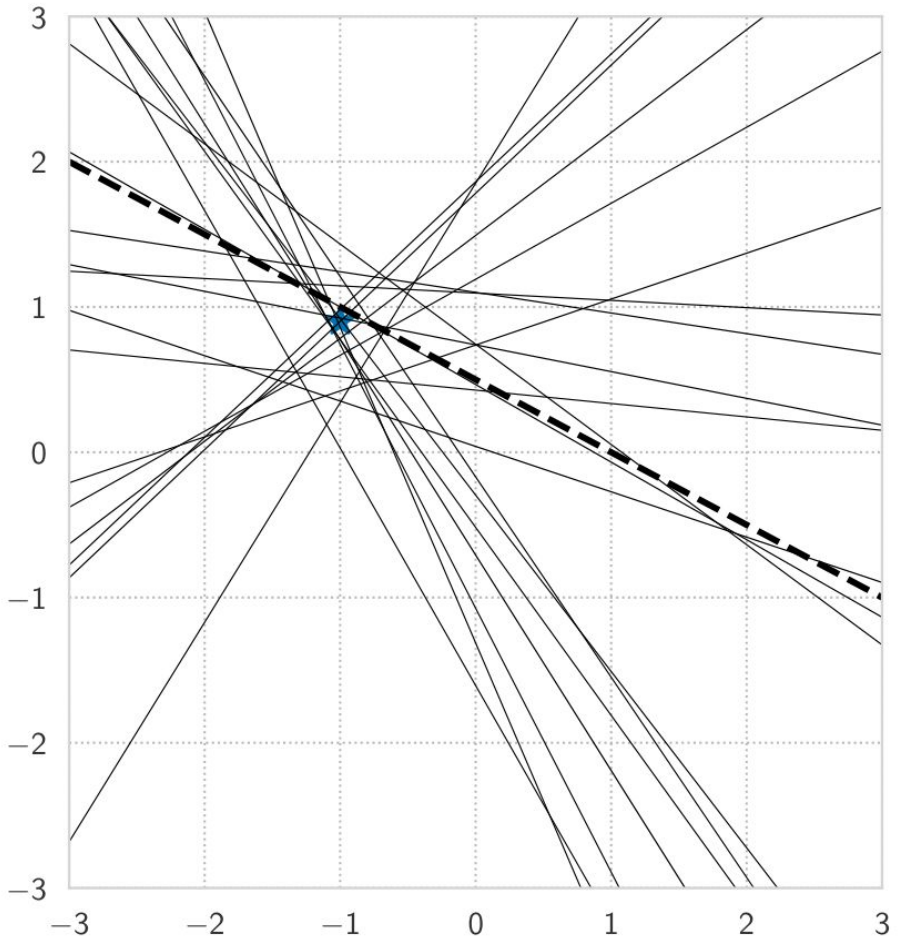




Линейная регрессия по-байесовски

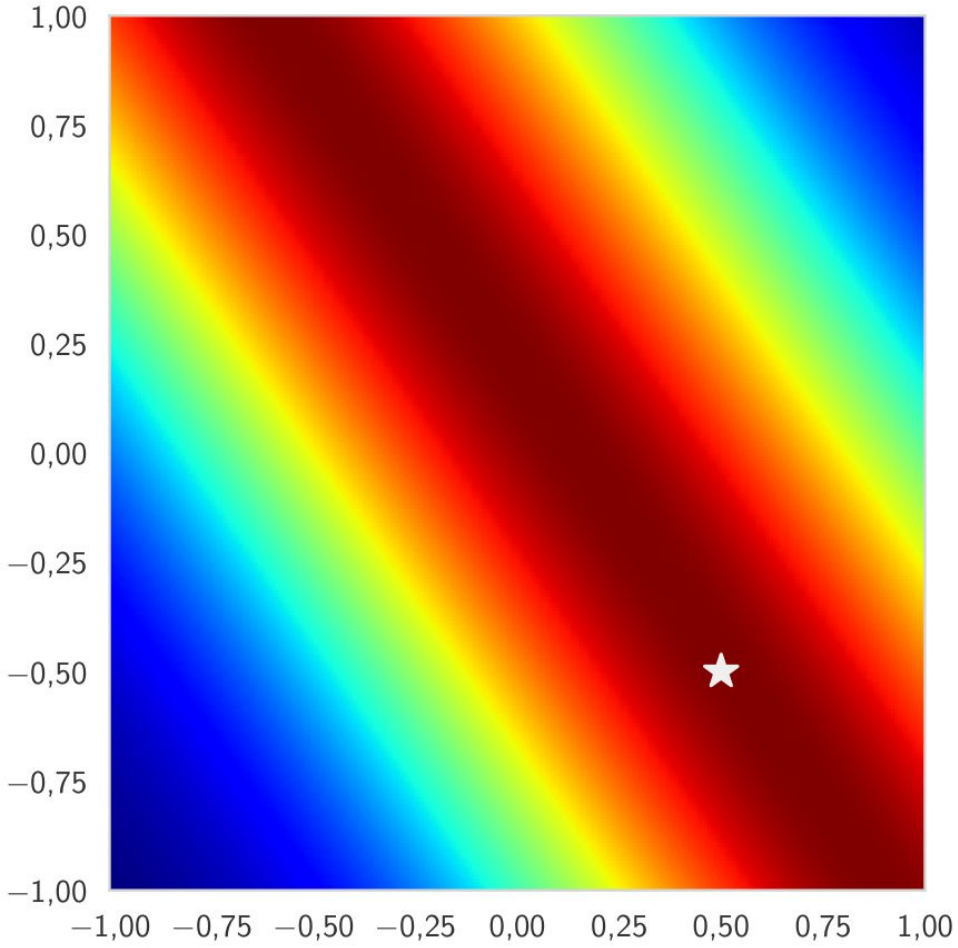
- Давайте проследим за байесовским выводом…





Линейная регрессия по-байесовски

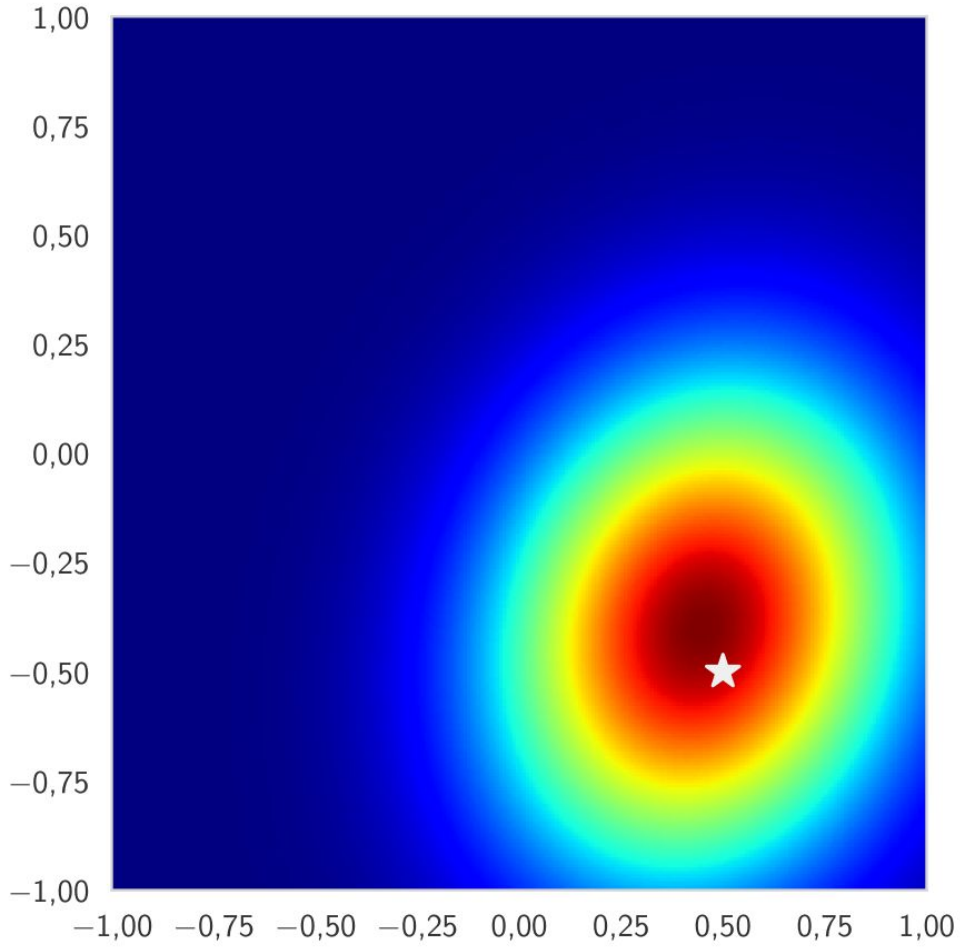
- Давайте проследим за байесовским выводом…

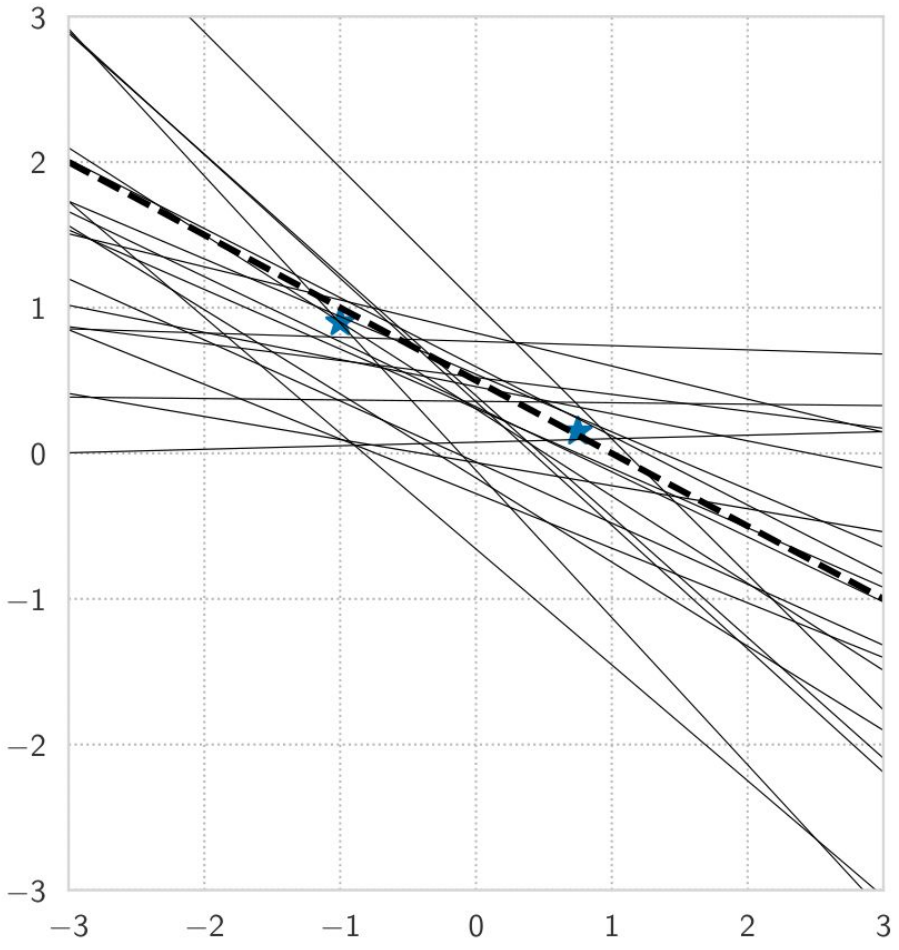




Линейная регрессия по-байесовски

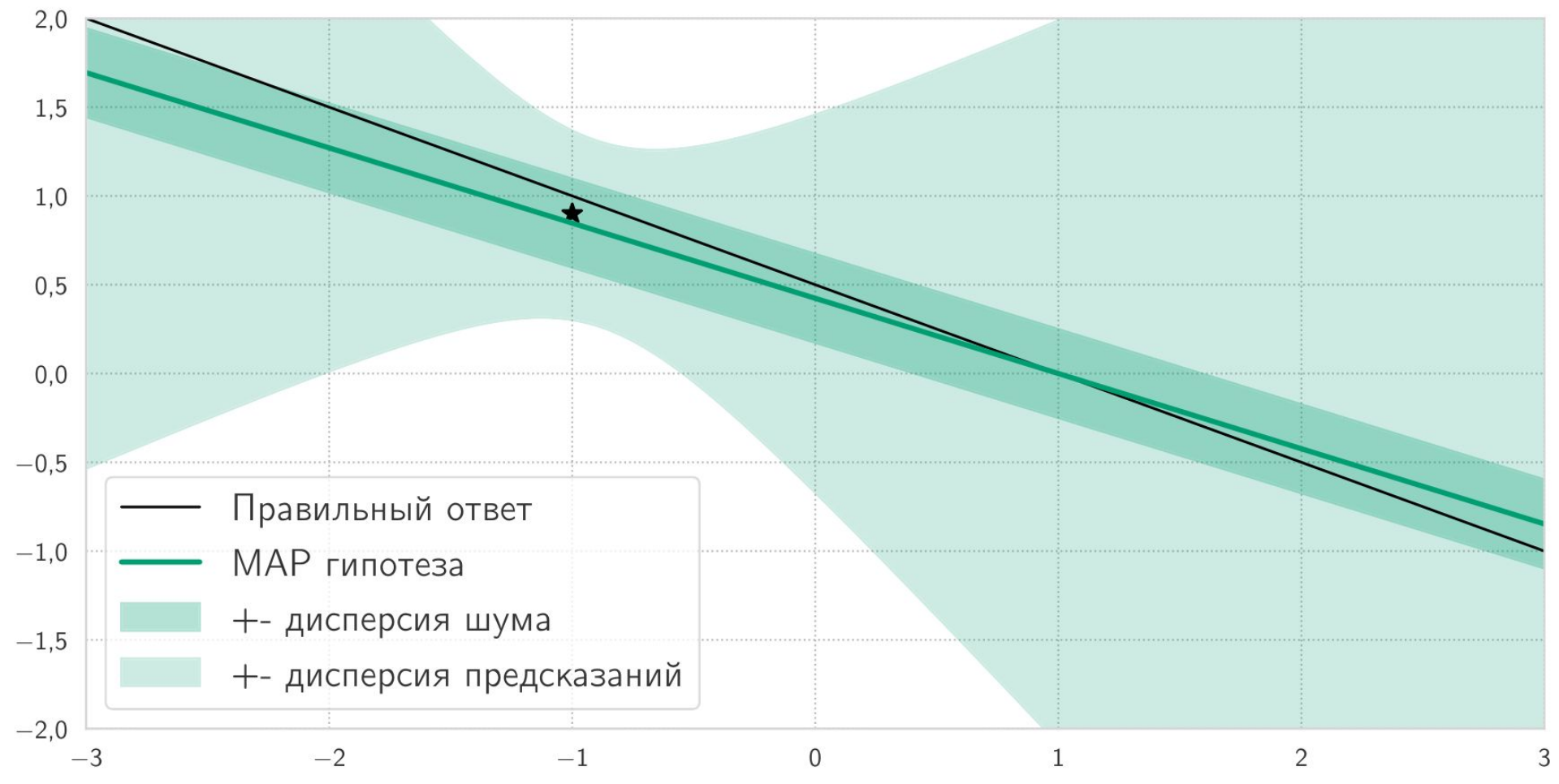
- Давайте проследим за байесовским выводом…





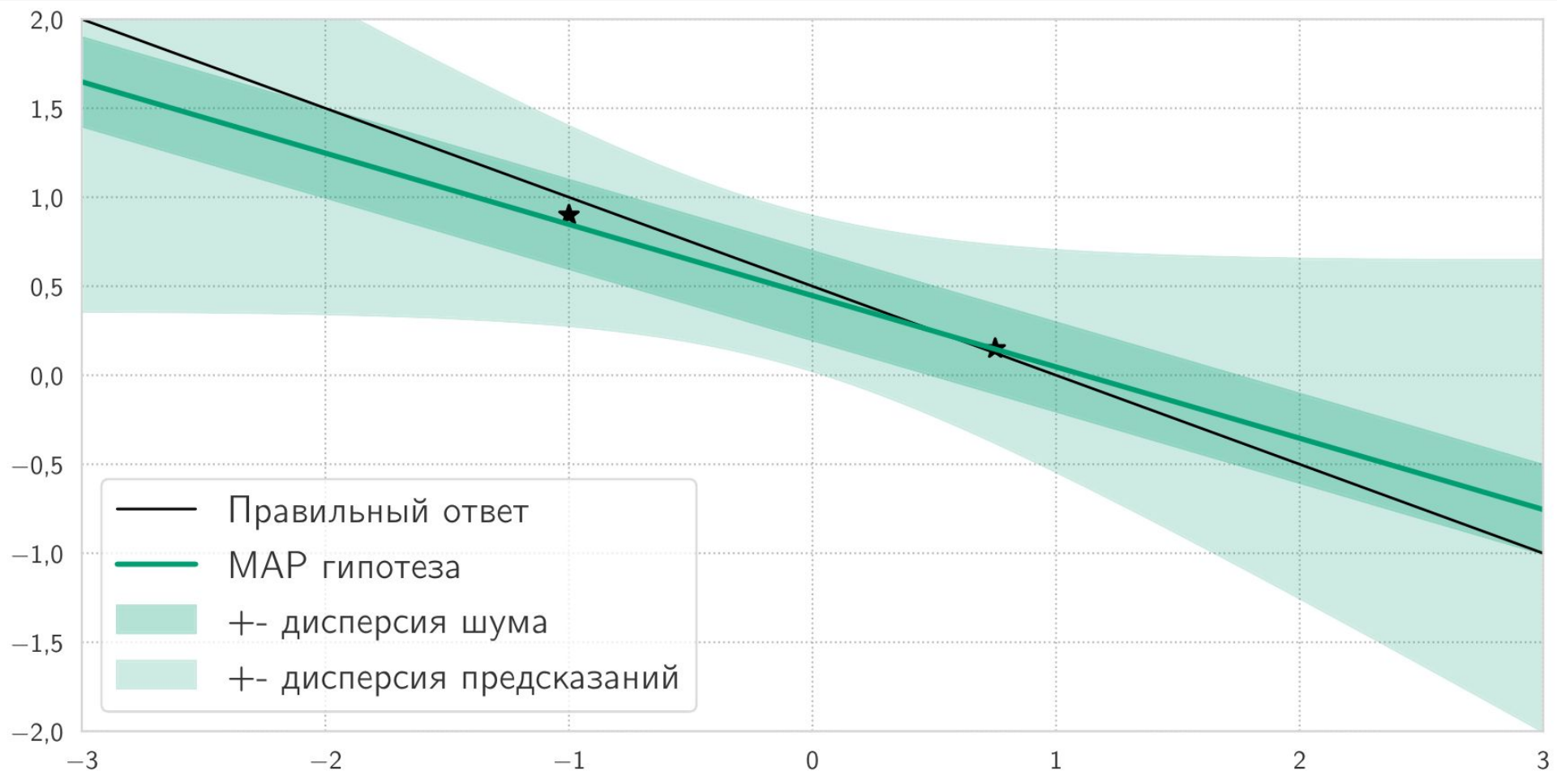
Линейная регрессия по-байесовски

- И предсказательное распределение тоже можно найти:



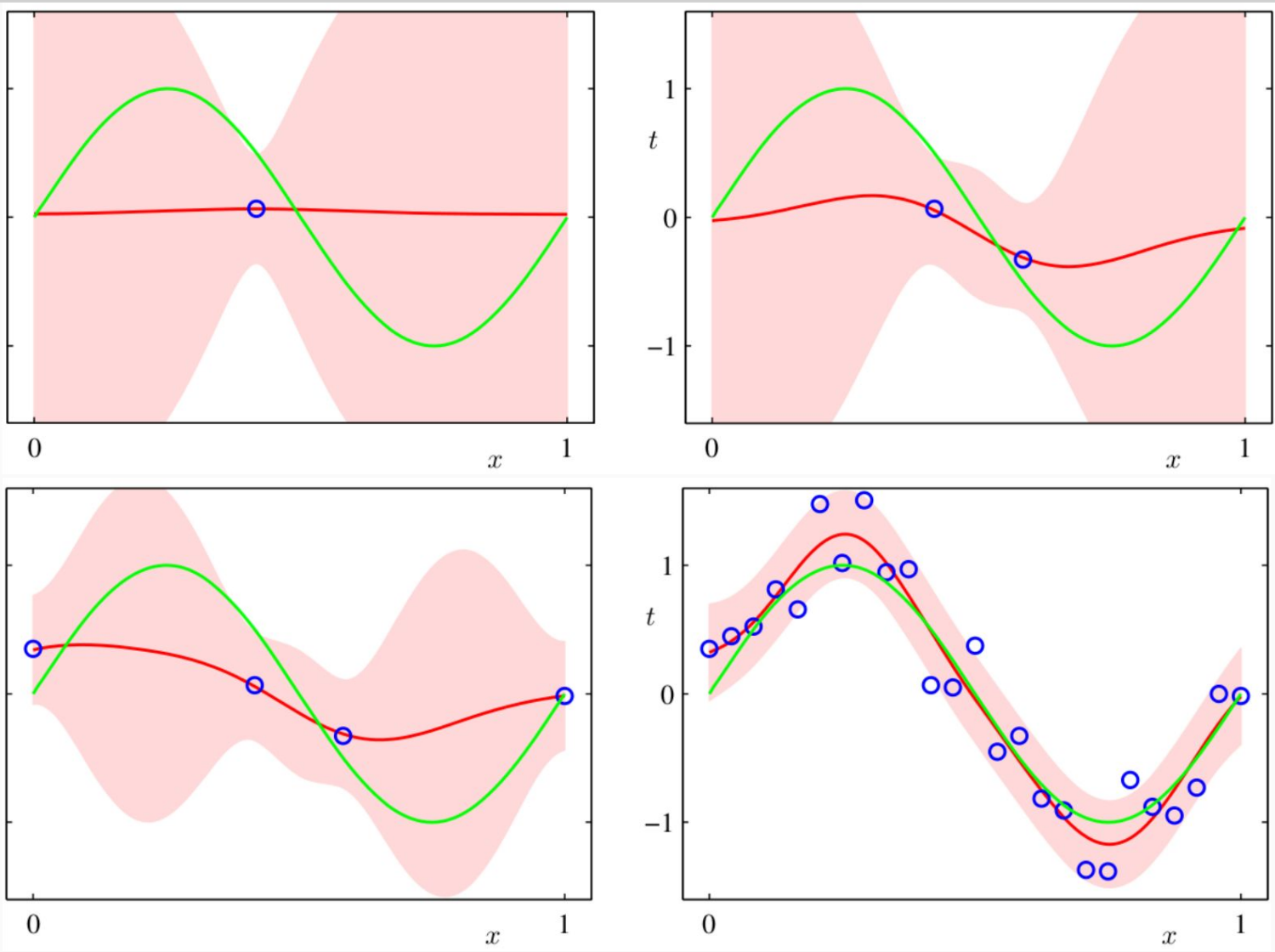
Линейная регрессия по-байесовски

- И предсказательное распределение тоже можно найти:



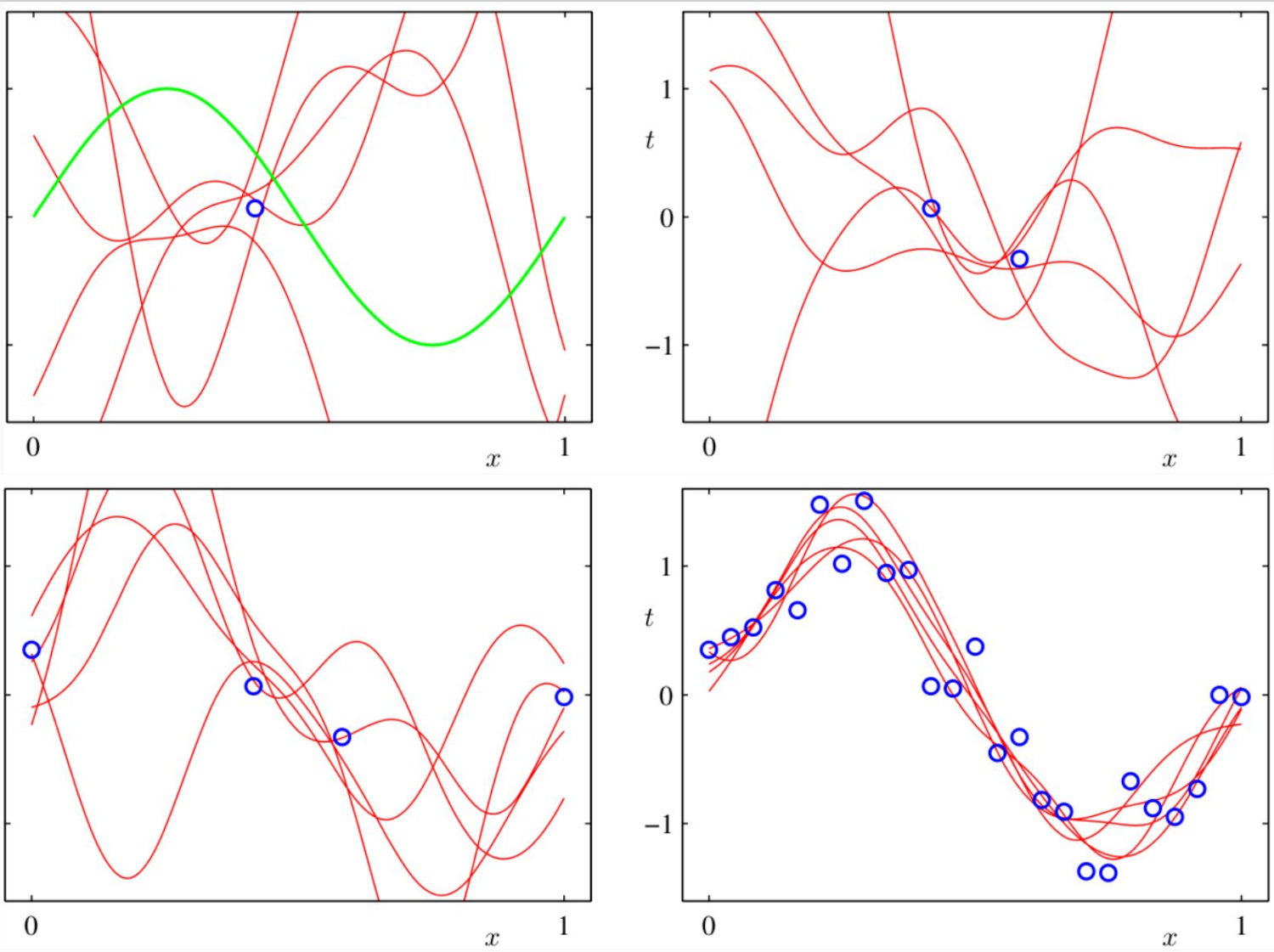
Линейная регрессия по-байесовски

- Для локальных признаков ещё нагляднее получается:



Линейная регрессия по-байесовски

- Для локальных признаков ещё нагляднее получается:

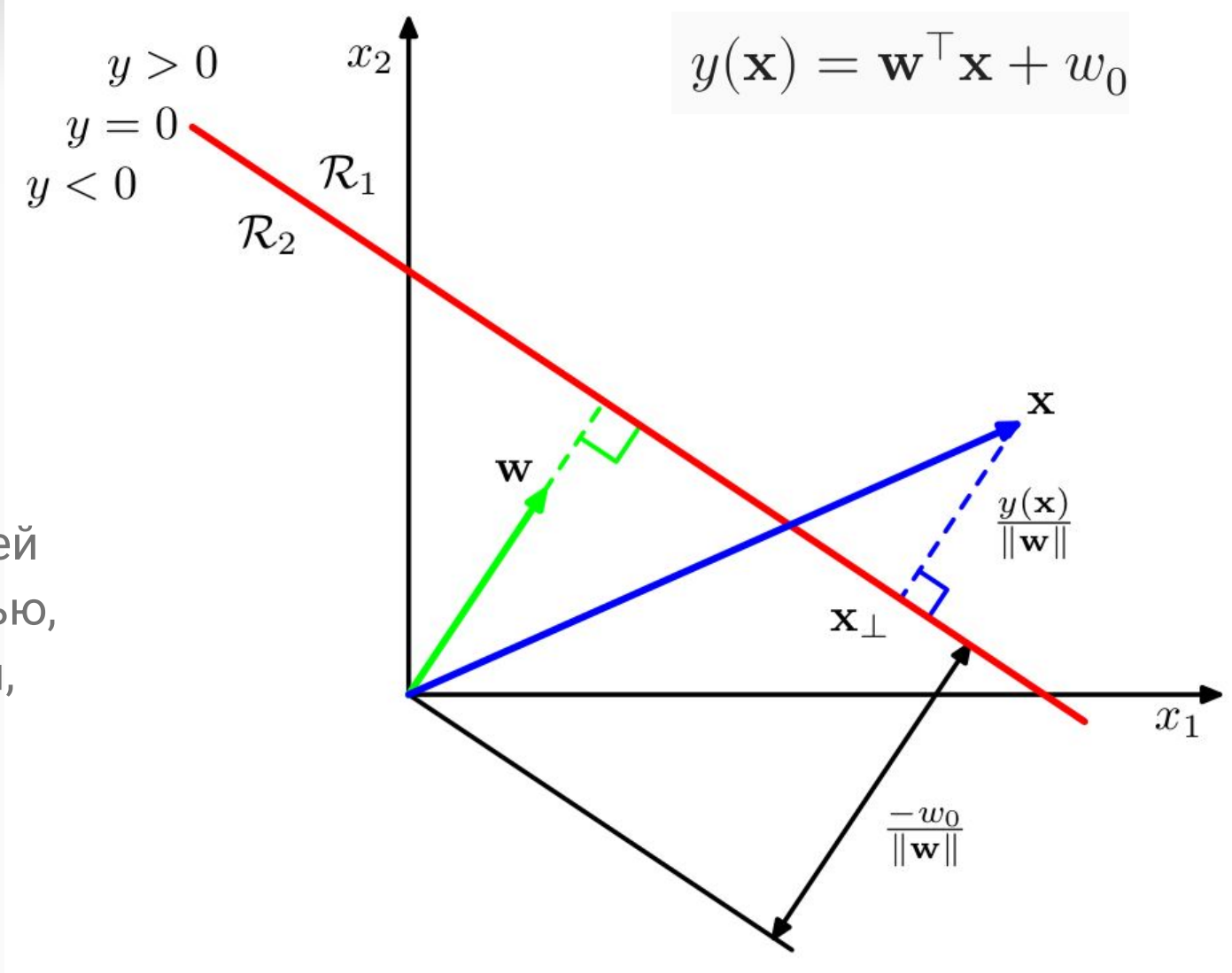


Геометрия классификации



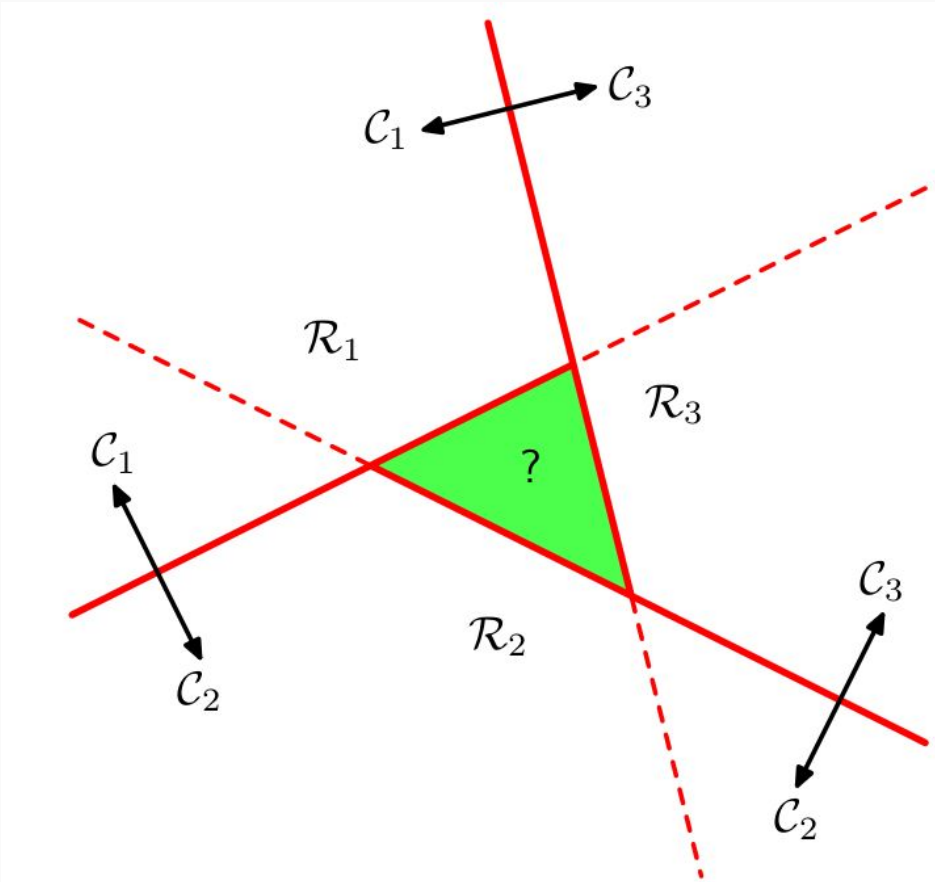
Линейная разделяющая поверхность

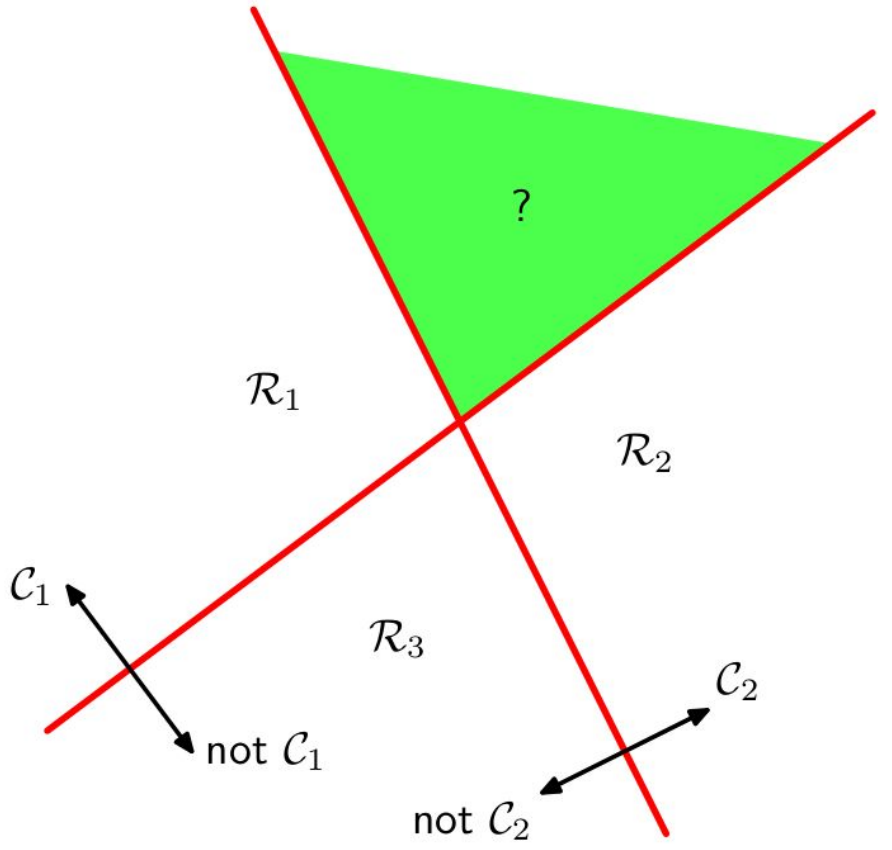
- С линейной разделяющей поверхностью, казалось бы, всё просто…



Линейная разделяющая поверхность

- …но возникают странные проблемки в случае нескольких классов
- Что делать?..



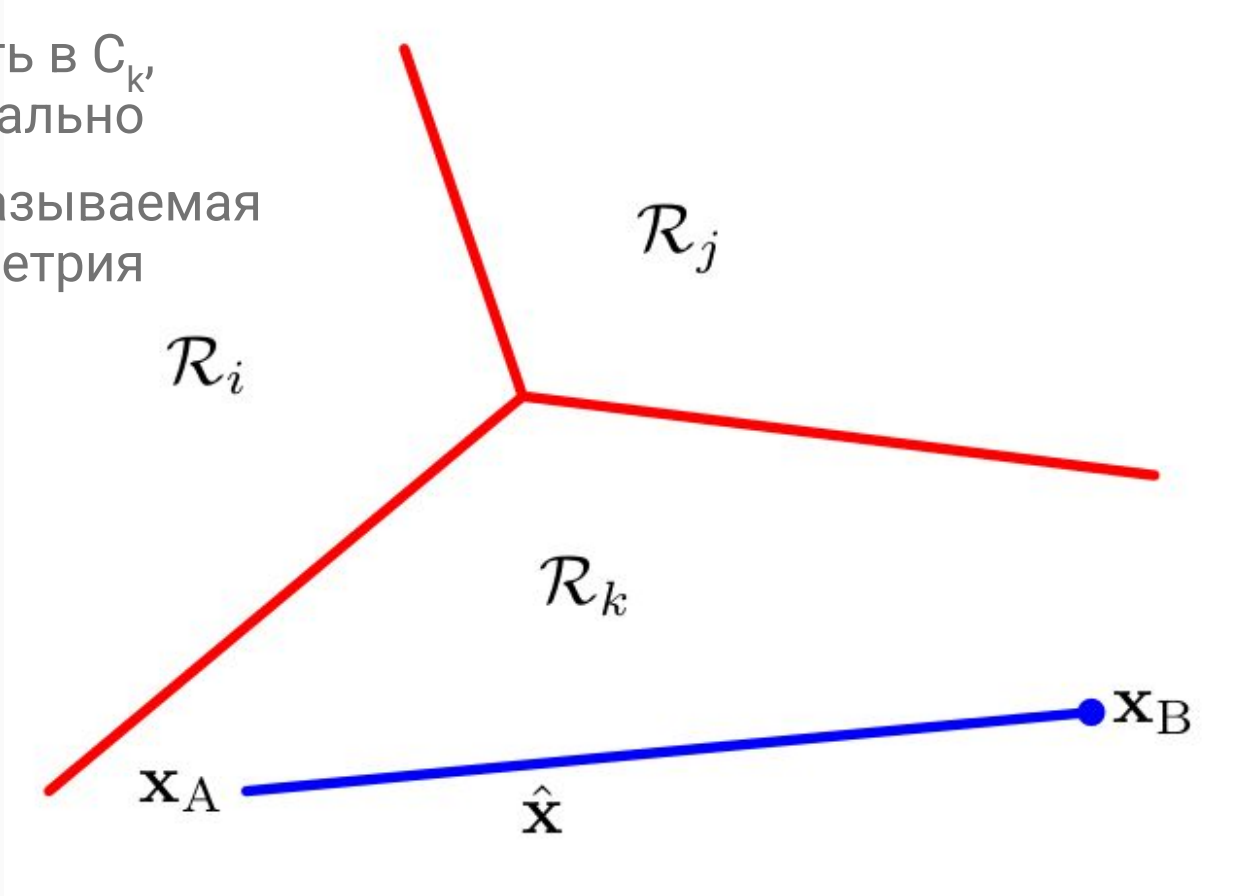


Линейная разделяющая поверхность

- Лучше рассмотреть единый дискриминант из K линейных функций:

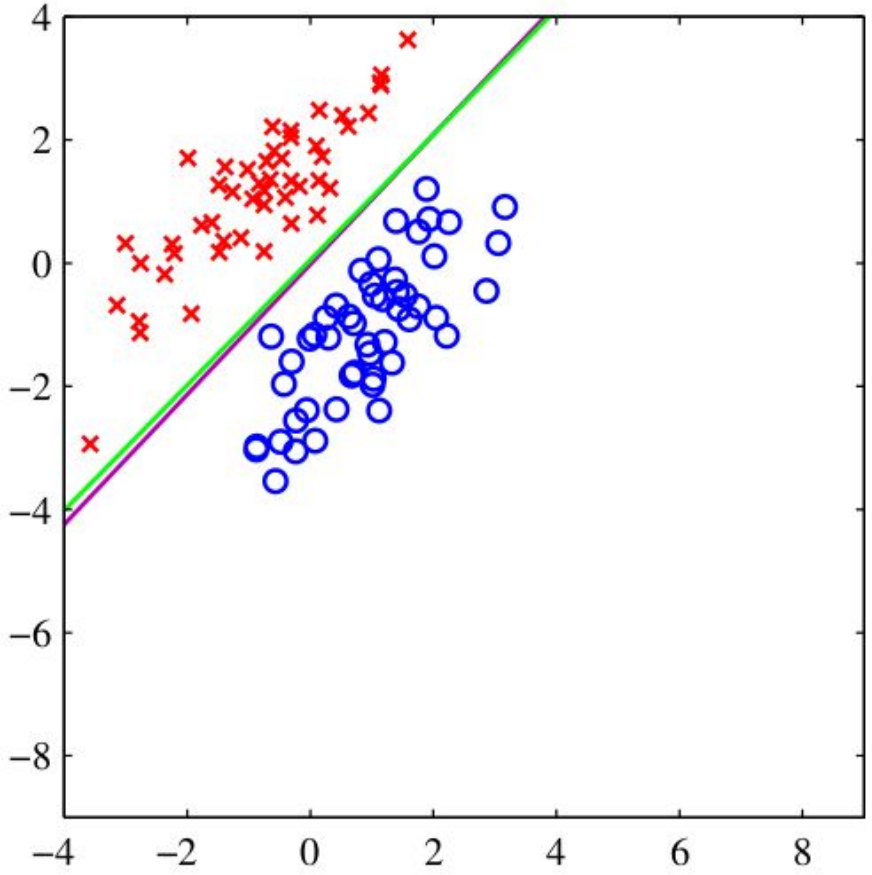
$$y_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{x} + w_{k0}.$$

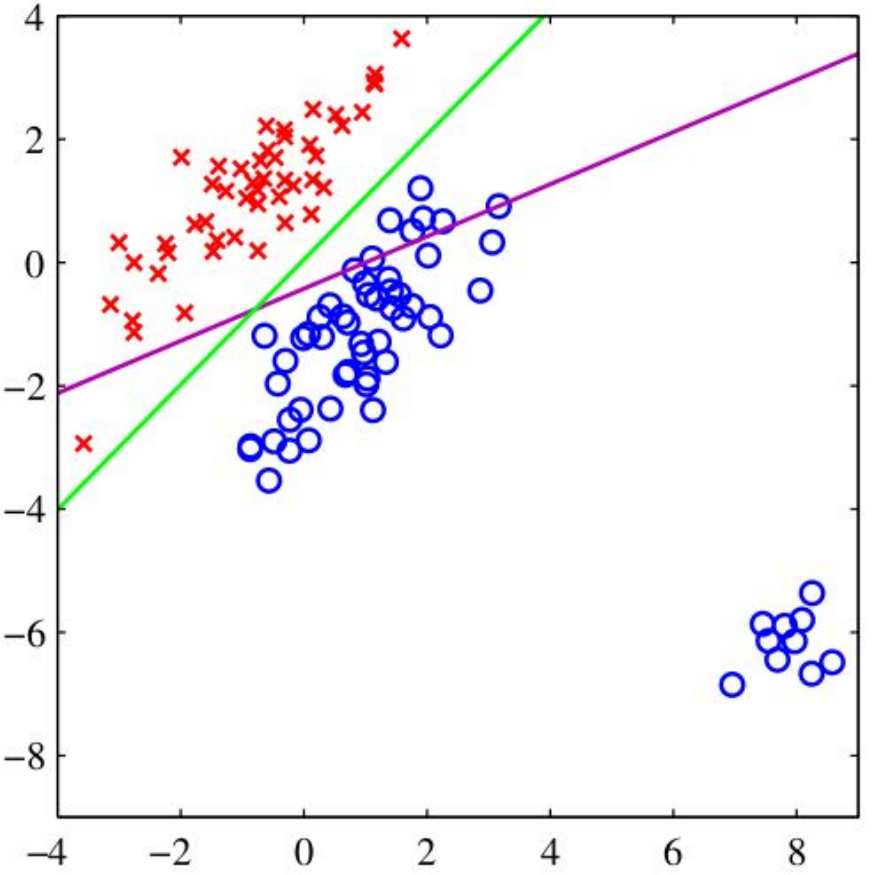
- Классифицировать в C_k, если $y_k(\mathbf{x})$ максимально
- Получается так называемая тропическая геометрия



Классификация через МНК

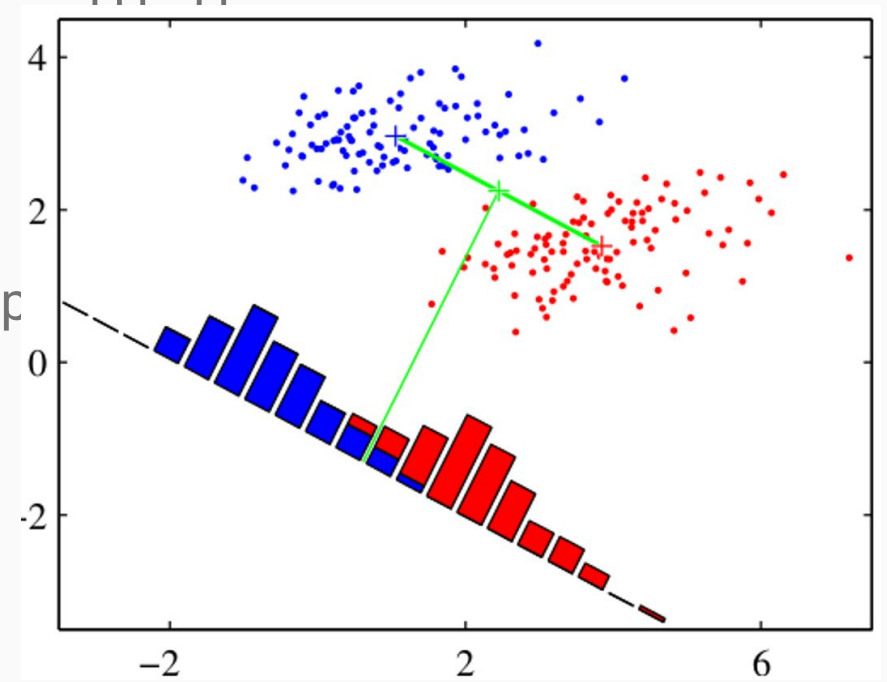
- Линейной регрессией можно классифицировать! Но не нужно :)





Линейный дискриминант Фишера

- Пример, который проясняет геометрию происходящего
- В линейном случае мы хотим спроецировать точки в размерность 1 (на нормаль разделяющей гиперплоскости) так, чтобы в этой размерности 1 они хорошо разделялись
- Т.е. классификация — это такой метод радикального сокращения размерности
- Как добиться оптимальности в этом смысле? Первая идея — найти серединный перпендикуляр между центрами наборов точек
- Но она как-то не ахти работает
- А в чём проблема?



Линейный дискриминант Фишера

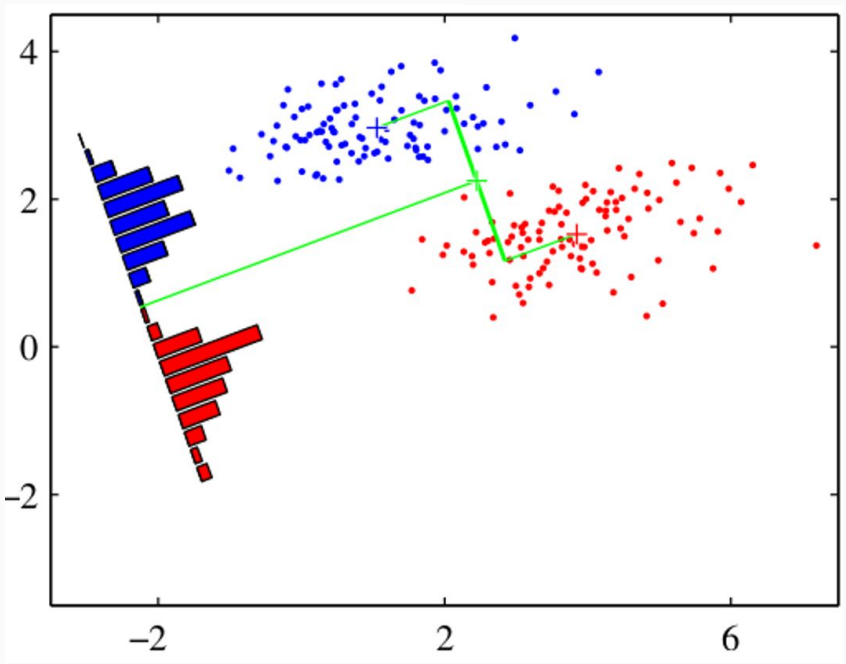
- “Хорошо разделялись” — это не только про большое расстояние, но и про маленькую дисперсию каждого класса!

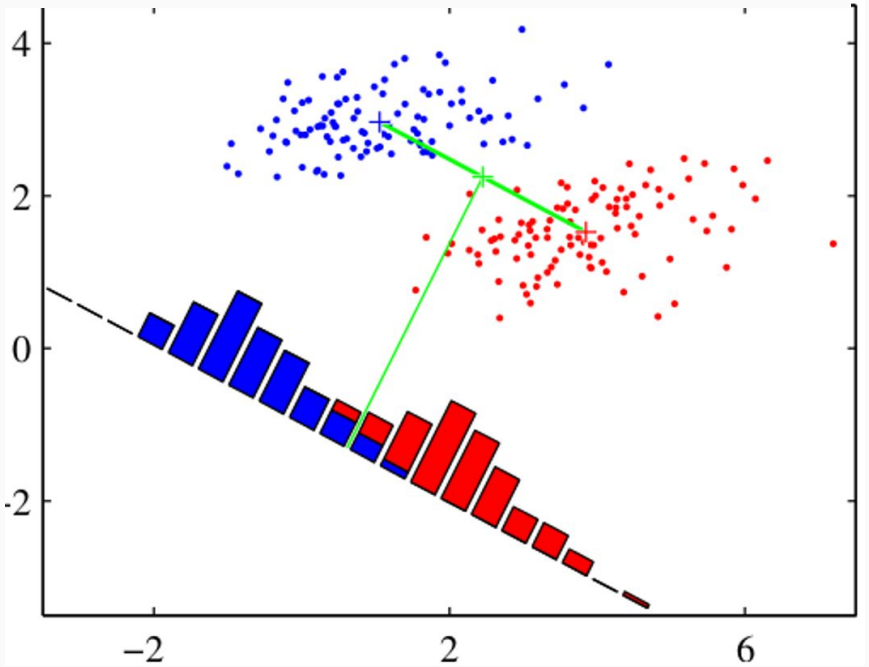
Отсюда *критерий Фишера*:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}, \text{ где}$$

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^\top,$$

$$\mathbf{S}_W = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^\top + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^\top.$$





Порождающие модели и классификация



Оптимальный байесовский классификатор

- Классификация через порождающие модели: давайте каждому классу сопоставим плотность $p(\mathbf{x}|C_k)$, найдём априорные распределения $p(C_k)$, будем искать $p(C_k|\mathbf{x})$ по теореме Байеса
- Для двух классов:

$$p(C_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_1)p(C_1) + p(\mathbf{x} \mid C_2)p(C_2)}.$$

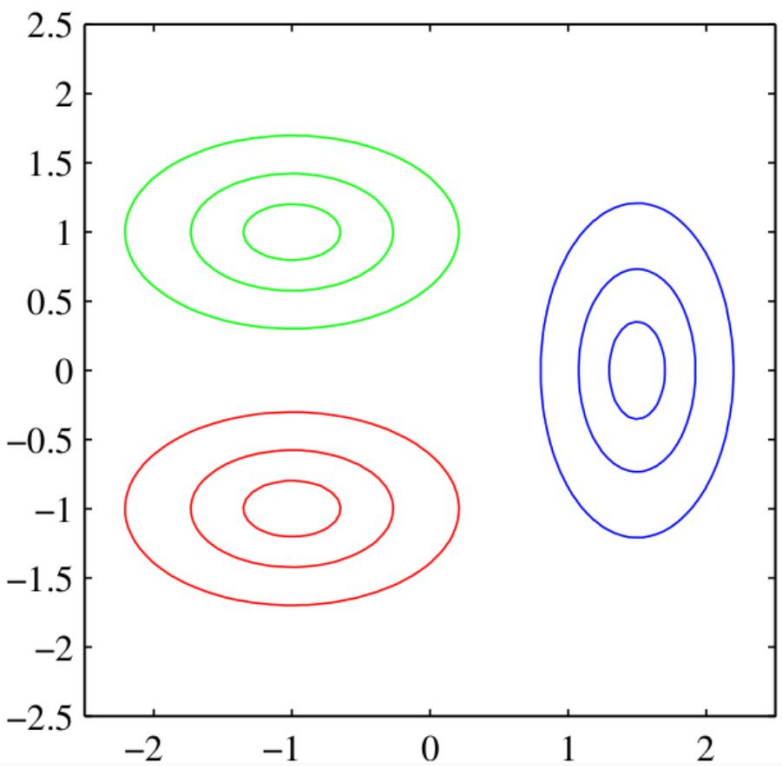
- Для нескольких классов:

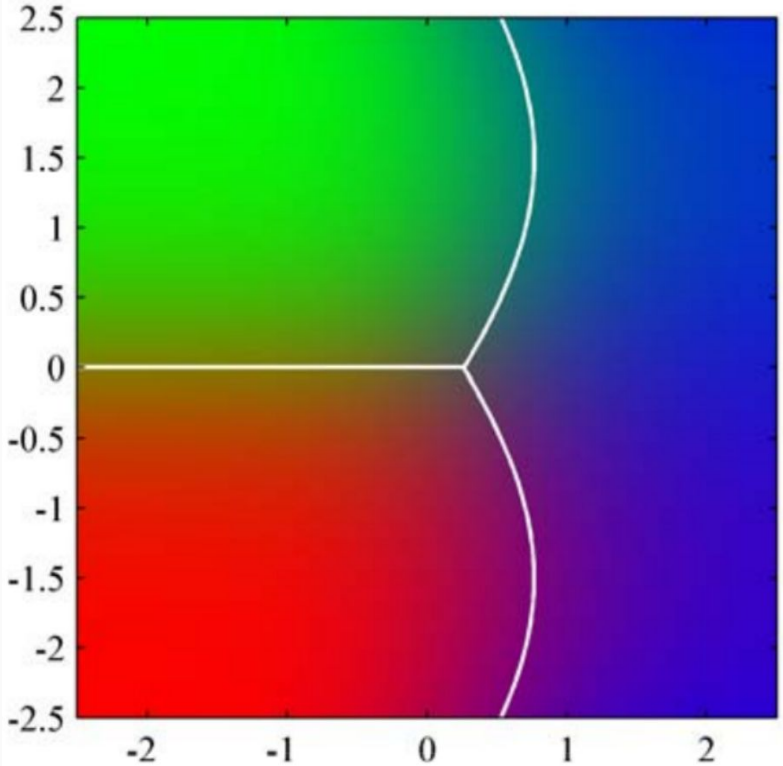
$$p(C_k \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_k)p(C_k)}{\sum_j p(\mathbf{x} \mid C_j)p(C_j)} = \frac{e^{a_k}}{\sum_j e^{a_j}}.$$

- Это и есть те самые логистический сигмоид и softmax, и к ним мы ещё вернёмся
- Это оптимальный байесовский классификатор, он будет работать всегда, для любых $p(\mathbf{x}|C_k)$

Оптимальный байесовский классификатор

- А если рассмотреть нормальные распределения в классах, то получатся LDA (linear discriminant analysis) и QDA (quadratic disc $p(\mathbf{x} \mid C_k) = N(\mathbf{x} \mid \mu_k, \Sigma)$





Логистическая регрессия



Идея логистической регрессии

- Давайте представим, что мы хотим промоделировать апостериорную вероятность $p(C|\mathbf{x})$ напрямую, с какими-нибудь весами внутри
- Как это лучше сделать?..

Идея логистической регрессии

- Давайте представим, что мы хотим промоделировать апостериорную вероятность p(C|**x**) напрямую, с какими-нибудь весами внутри
- Как это лучше сделать?..
- ...получается, что моделировать линейной функцией лучше log-odds:

$$p(C_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_1)p(C_1) + p(\mathbf{x} \mid C_2)p(C_2)} = \frac{1}{1 + e^{-a}} = \sigma(a),$$

где $a = \ln \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_2)p(C_2)}, \qquad \sigma(a) = \frac{1}{1 + e^{-a}}.$

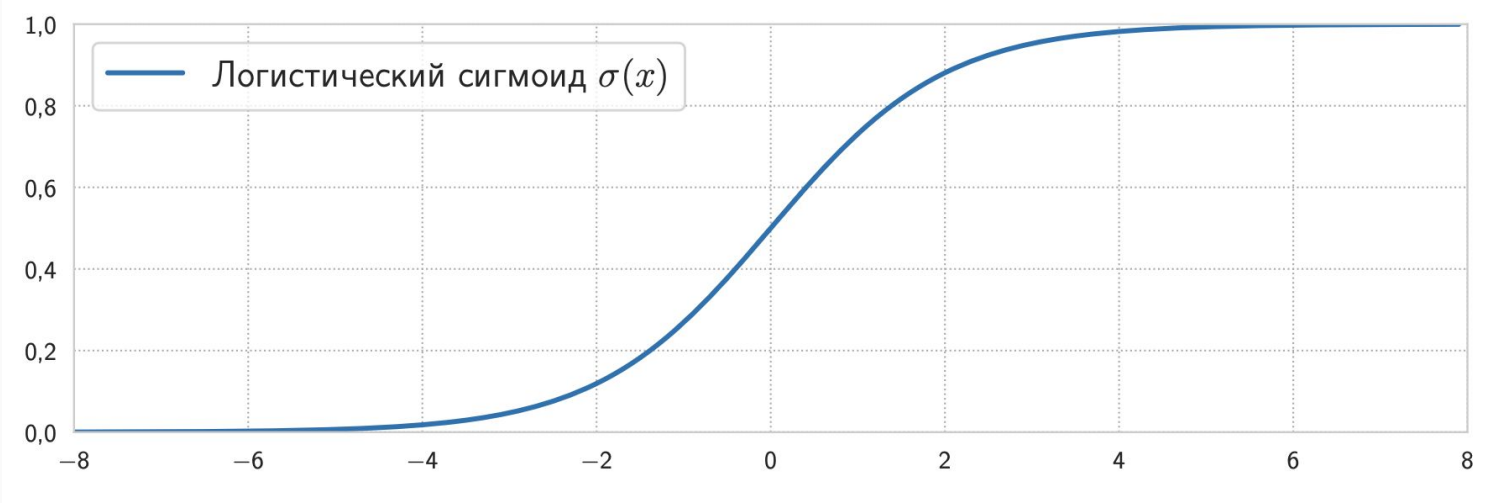
Идея логистической регрессии

- Получается, что моделировать линейной функцией лучше log-odds:

$$p(C_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_1)p(C_1) + p(\mathbf{x} \mid C_2)p(C_2)} = \frac{1}{1 + e^{-a}} = \sigma(a),$$

$$\text{где } a = \ln \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_2)p(C_2)}, \qquad \sigma(a) = \frac{1}{1 + e^{-a}}.$$

- Кстати, о логистическом сигмоиде



Правдоподобие в логистической регрессии

- Логистическая регрессия – это модель, в которой log-odds моделируются линейной функцией:

$$p\left(C_1|\boldsymbol{x}\right) = \sigma(\boldsymbol{w}^\top \boldsymbol{x}), \qquad p\left(C_2|\boldsymbol{x}\right) = 1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x}).$$

- Теперь можно записать правдоподобие и научиться его максимизировать:

$$p\left(D|\boldsymbol{w}\right) = \prod_{n=1}^{N} p\left(y_n|\boldsymbol{x}_n, \boldsymbol{w}\right) = \prod_{n=1}^{N} p\left(C_1|\boldsymbol{x}\right)^{t_n} p\left(C_2|\boldsymbol{x}\right)^{1-t_n} =$$

$$= \prod_{n=1}^{N} \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}\right)^{t_n} \left(1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}\right)\right)^{1-t_n}.$$

Правдоподобие в логистической регрессии

- Кстати, о градиенте правдоподобия:

$$\ln p\left(D|\boldsymbol{w}\right) = \sum_{n=1}^{N}\left[t_n \ln \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right) + \left(1 - t_n\right) \ln\left(1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\right)\right]$$

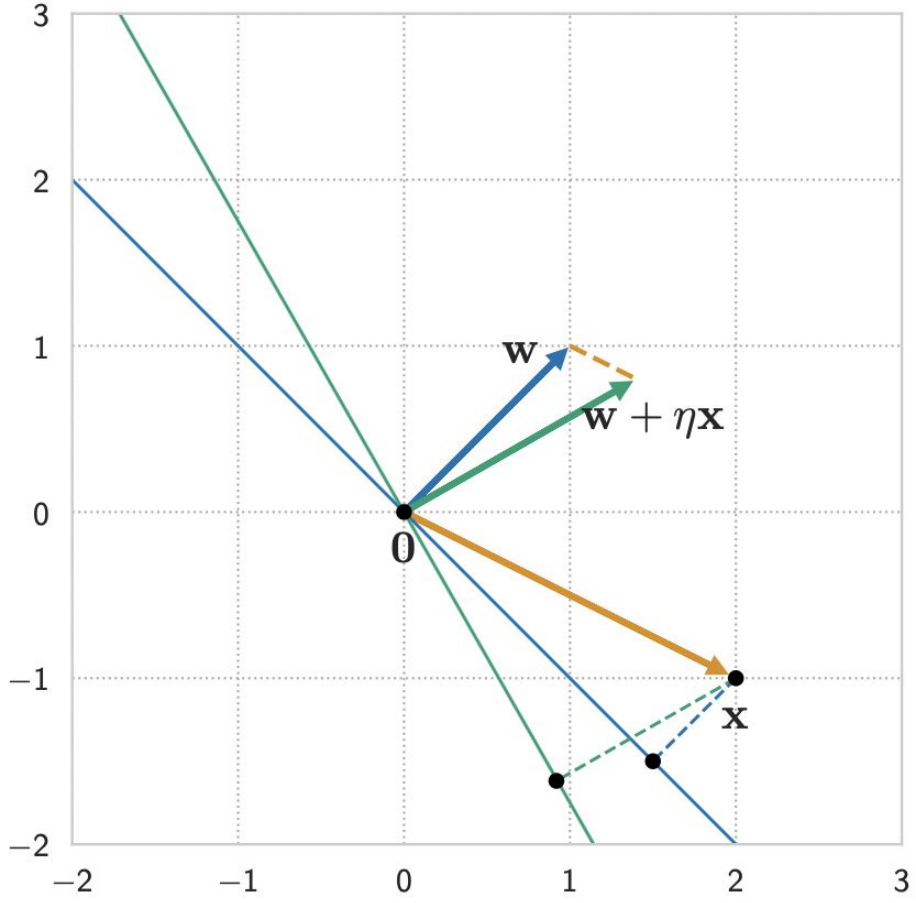
$$\nabla_{\boldsymbol{w}} \ln p\left(D|\boldsymbol{w}\right) = \sum_{n=1}^{N}\left[\frac{t_n \sigma'\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)}{\sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)}\boldsymbol{x}_n - \frac{\left(1 - t_n\right)\sigma'\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)}{1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)}\boldsymbol{x}_n\right] =$$

$$= \sum_{n=1}^{N}\left[\frac{t_n \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\left(1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\right)}{\sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)} - \frac{\left(1 - t_n\right)\sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\left(1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\right)}{1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)}\right]\boldsymbol{x}_n =$$

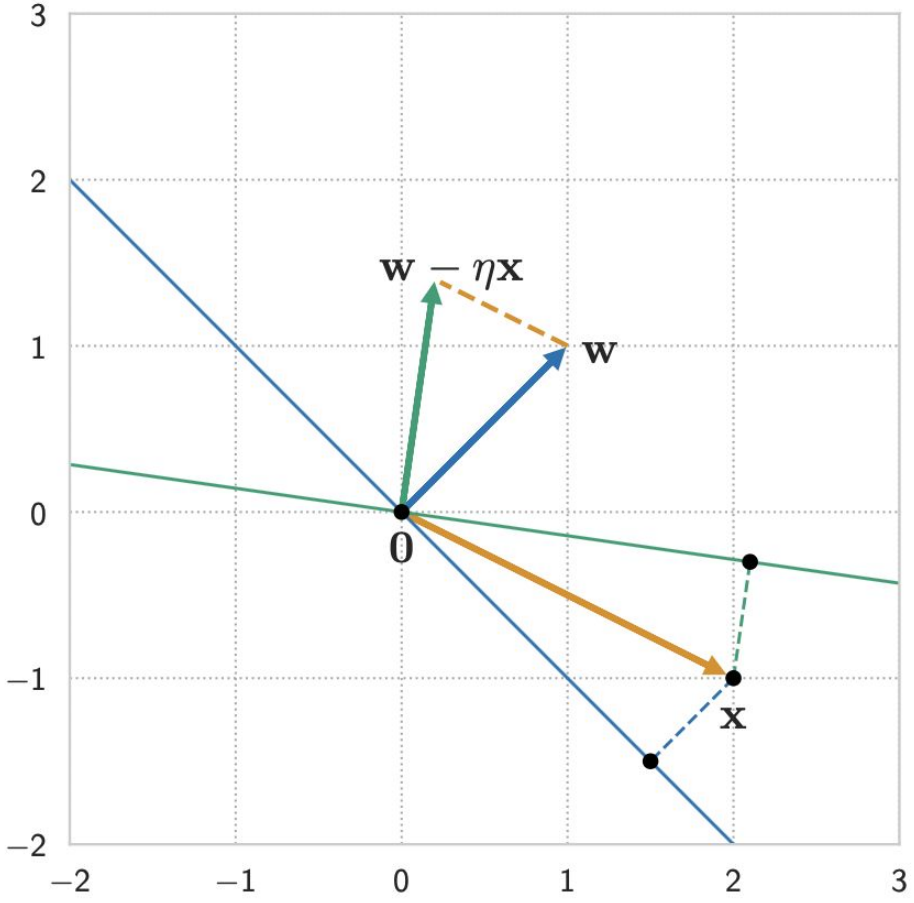
$$= \sum_{n=1}^{N}\left[t_n\left(1 - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\right) - \left(1 - t_n\right)\sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\right]\boldsymbol{x}_n = \sum_{n=1}^{N}\left(t_n - \sigma\left(\boldsymbol{w}^\top \boldsymbol{x}_n\right)\right)\boldsymbol{x}_n.$$

Правдоподобие в логистической регрессии

- Кстати, о градиенте правдоподобия:



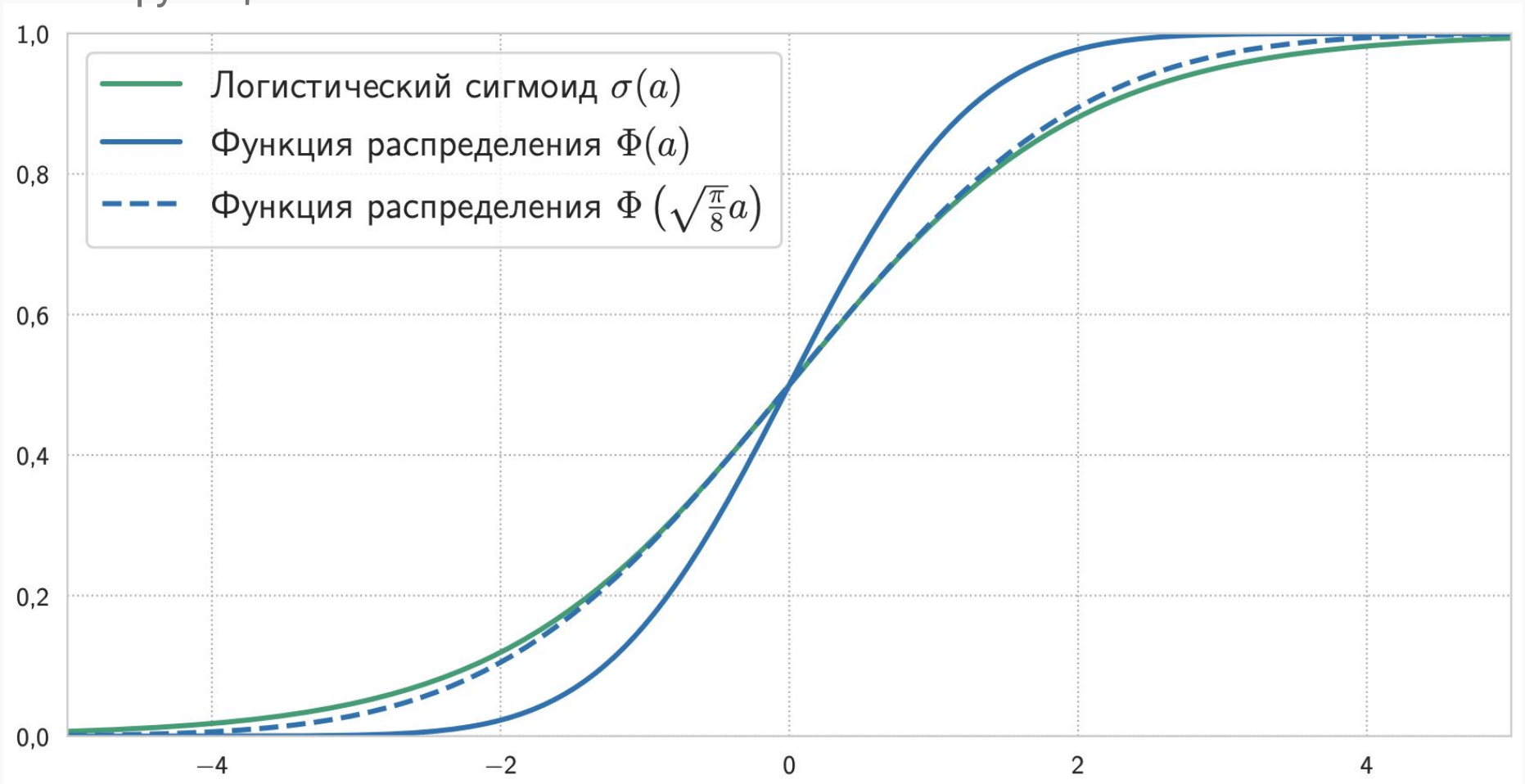
(а) x классифицирован правильно



(б) x классифицирован неправильно

Правдоподобие в логистической регрессии

- В байесовский вывод вдаваться не будем, но отметим, что сигмоидов на свете много; там как раз понадобится опять пробит-функция:

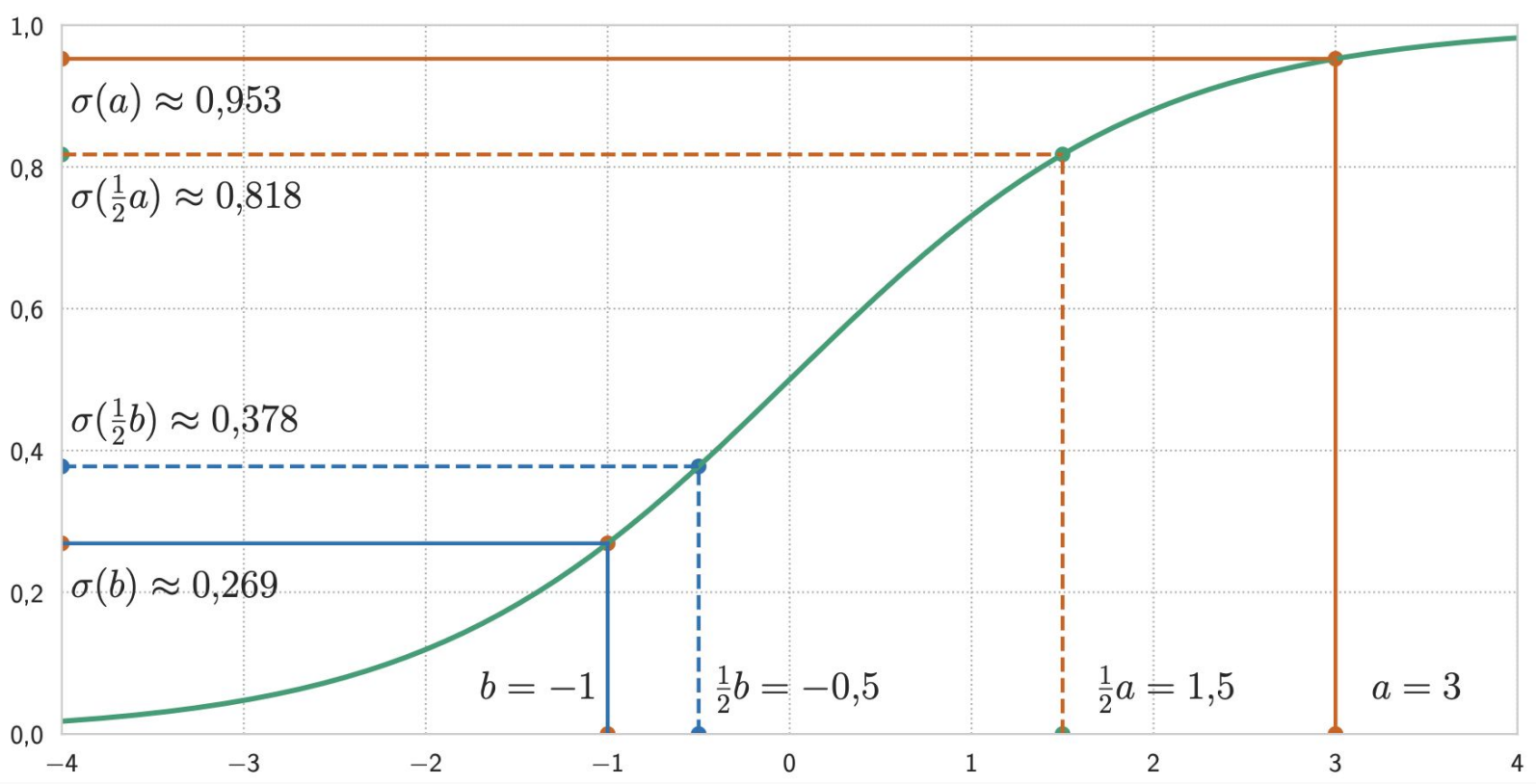


Правдоподобие в логистической регрессии

- И предсказательное распределение там получается такое:

$$p\left(\mathcal{C}_1|\boldsymbol{x},\boldsymbol{t}\right) = \sigma\left(\kappa(\sigma_a^2)\mu_a\right),$$

$$\mu_a = \boldsymbol{w}_{\text{MAP}}^\top \boldsymbol{x}, \qquad \sigma_a^2 = \boldsymbol{x}^\top \Sigma_N \boldsymbol{x}, \qquad \kappa(\sigma^2) = \frac{1}{\sqrt{1+\frac{\pi}{8}\sigma^2}}.$$

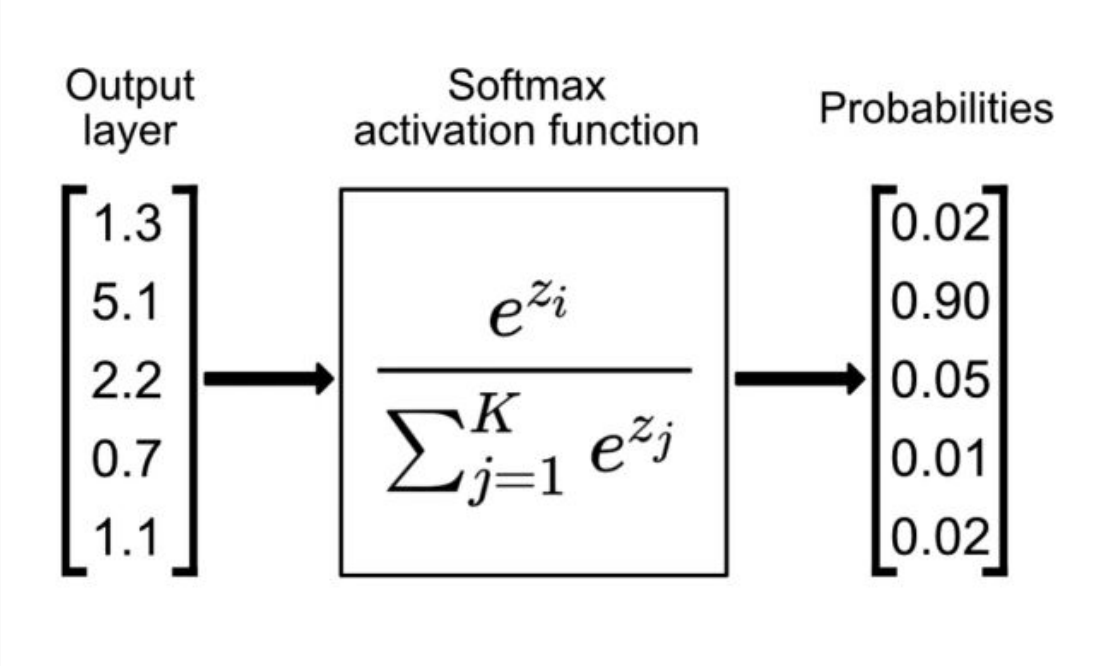


Несколько классов и softmax

- А если классов несколько, то должен быть отдельный вектор весов для каждого класса

$$p(C_k \mid \phi) = y_k(\phi) = \frac{e^{a_k}}{\sum_j e^{a_j}} \text{ для } a_k = \mathbf{w}_k^\top \phi.$$

- Это и есть пресловутая функция softmax, которой заканчивается любая нейросеть для классификации:

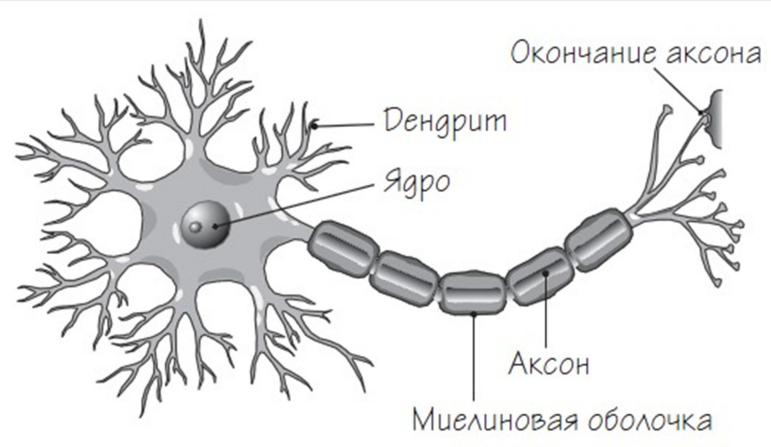


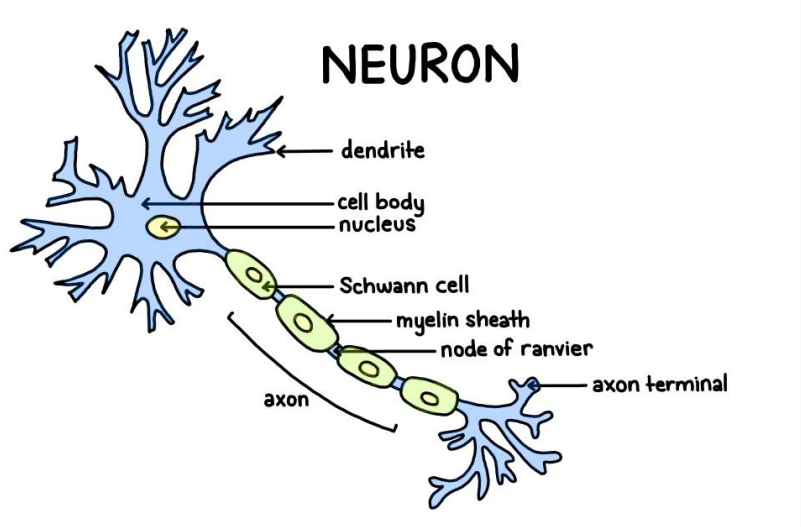
От одного нейрона к графу вычислений



От биологии к информатике

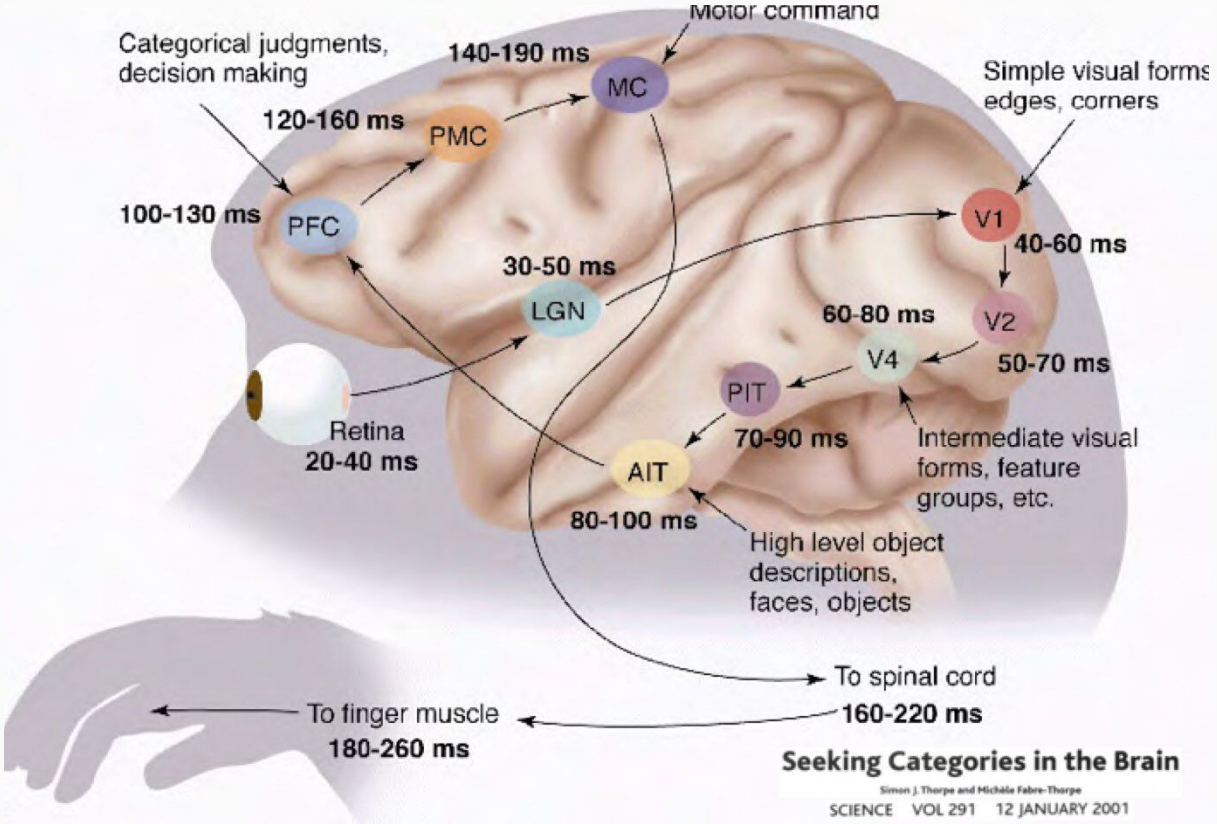
- Мозг состоит из нейронов, которые устроены примерно одинаково
- Нейрон через дендриты получает сигнал от других нейронов, время от времени запускает сигнал по аксону, и через синапсы сигнал аксона доходит до дендритов других нейронов.
- Нейронов очень много: 10^{11} нейронов, в среднем 7000 связей у каждого, то есть 10^{15} синапсов





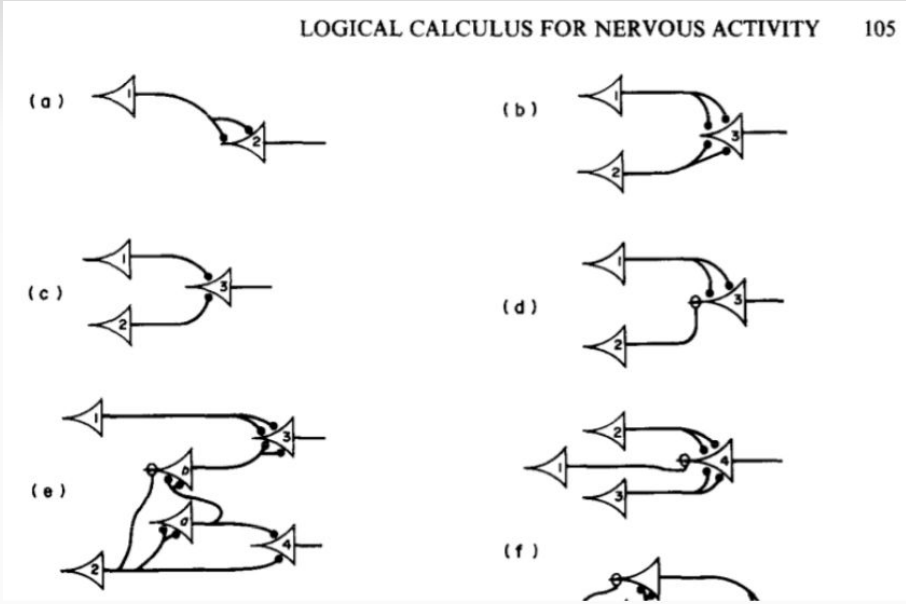
От биологии к информатике

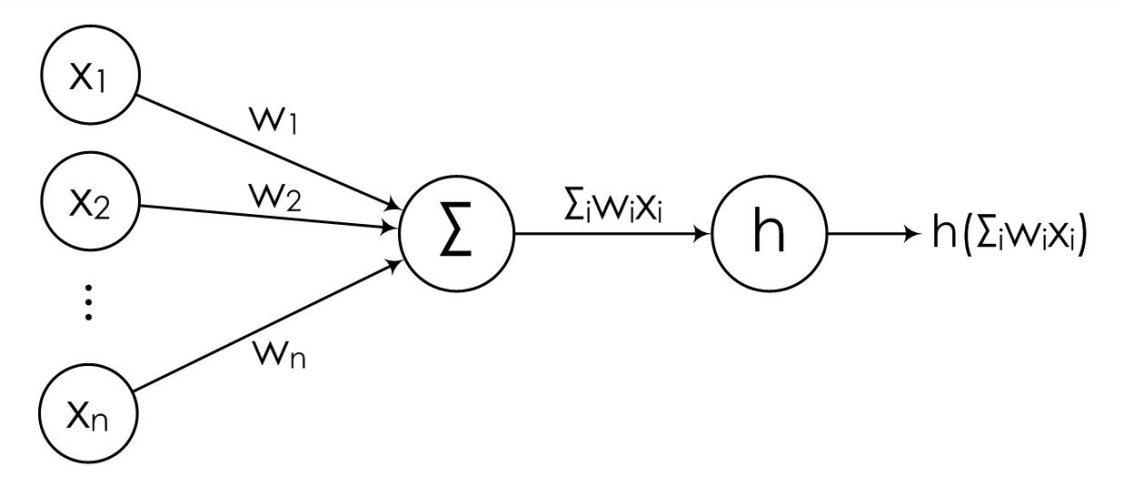
- Но «тактовая частота» этого «компьютера» очень низкая: от 10 до 200 герц.
- И при этом мы, например, распознаём лицо человека за 200-300 миллисекунд.
- Это значит, что цепочки короткие, и вычисления в мозге скорее параллельные, чем последовательные.



McCullogh, Pitts, 1943

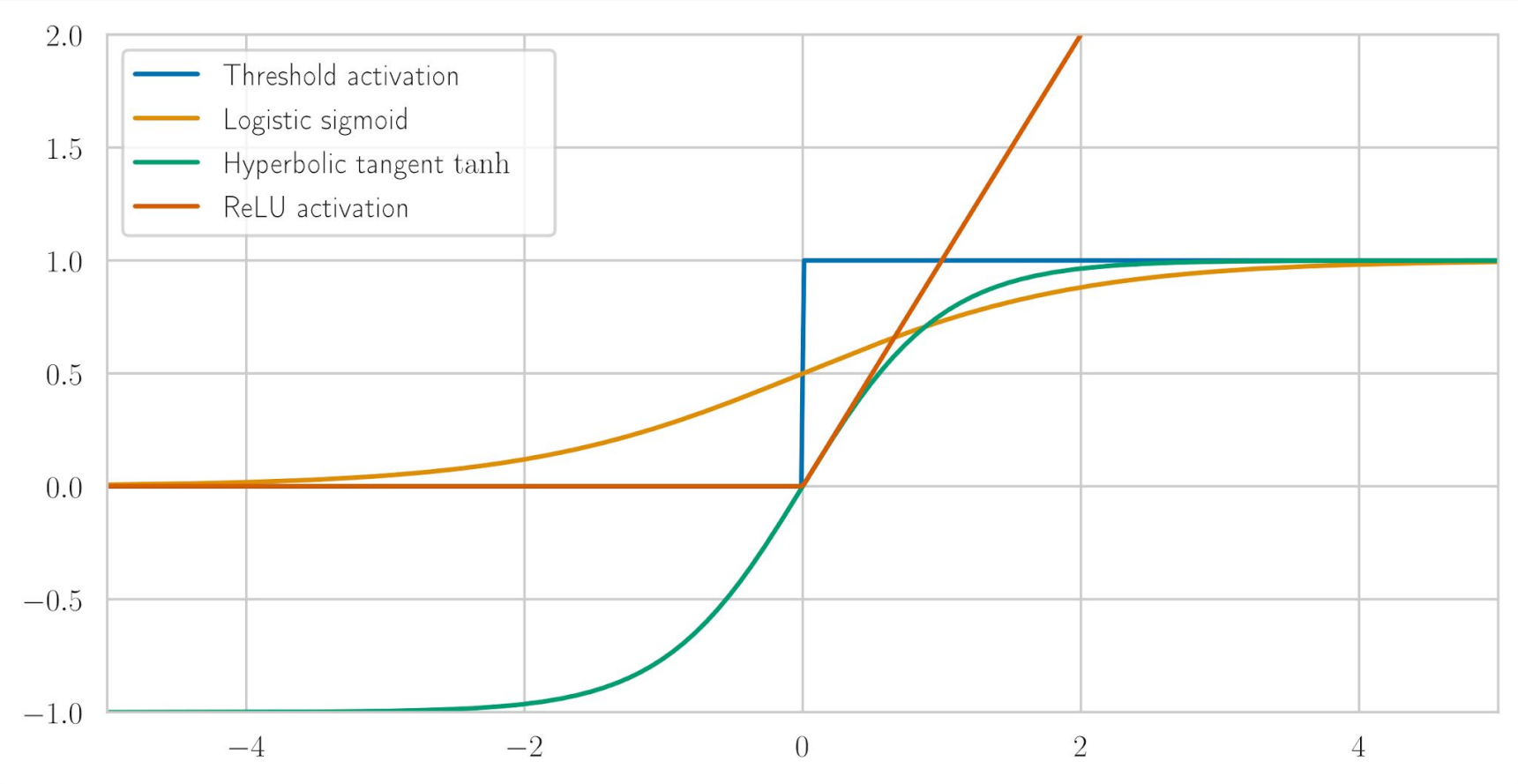
- Искусственные нейросети появились даже раньше, чем тест Тьюринга и Дартмутский семинар
- (Маккаллох, Питтс, 1943): первая модель нейрона, которая была фактически современной!





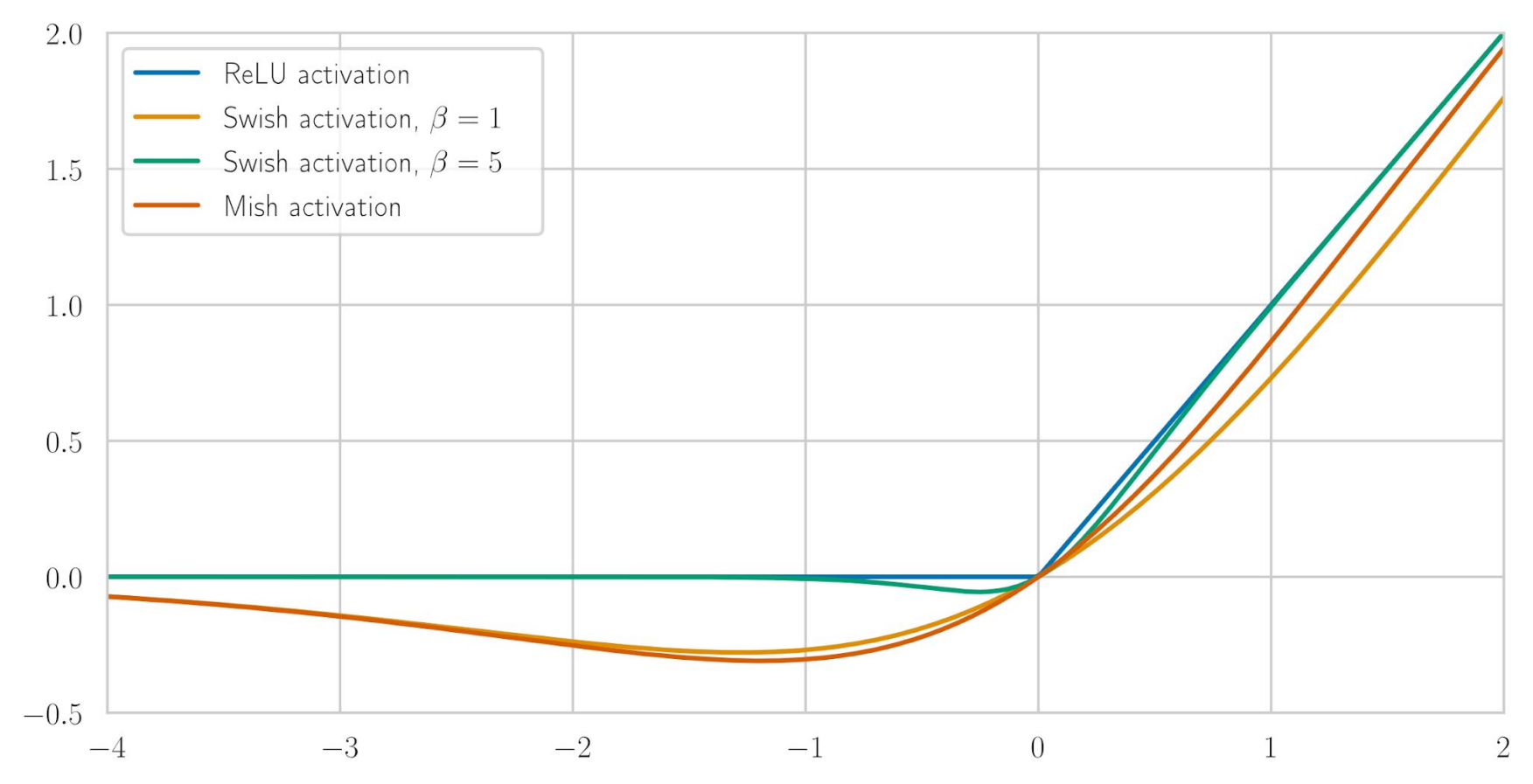
Функции активации

- В наше время появились и другие виды связей (например, механизмы внимания), но в основном сети состоят из таких же искусственных нейронов; только функции активации меняются



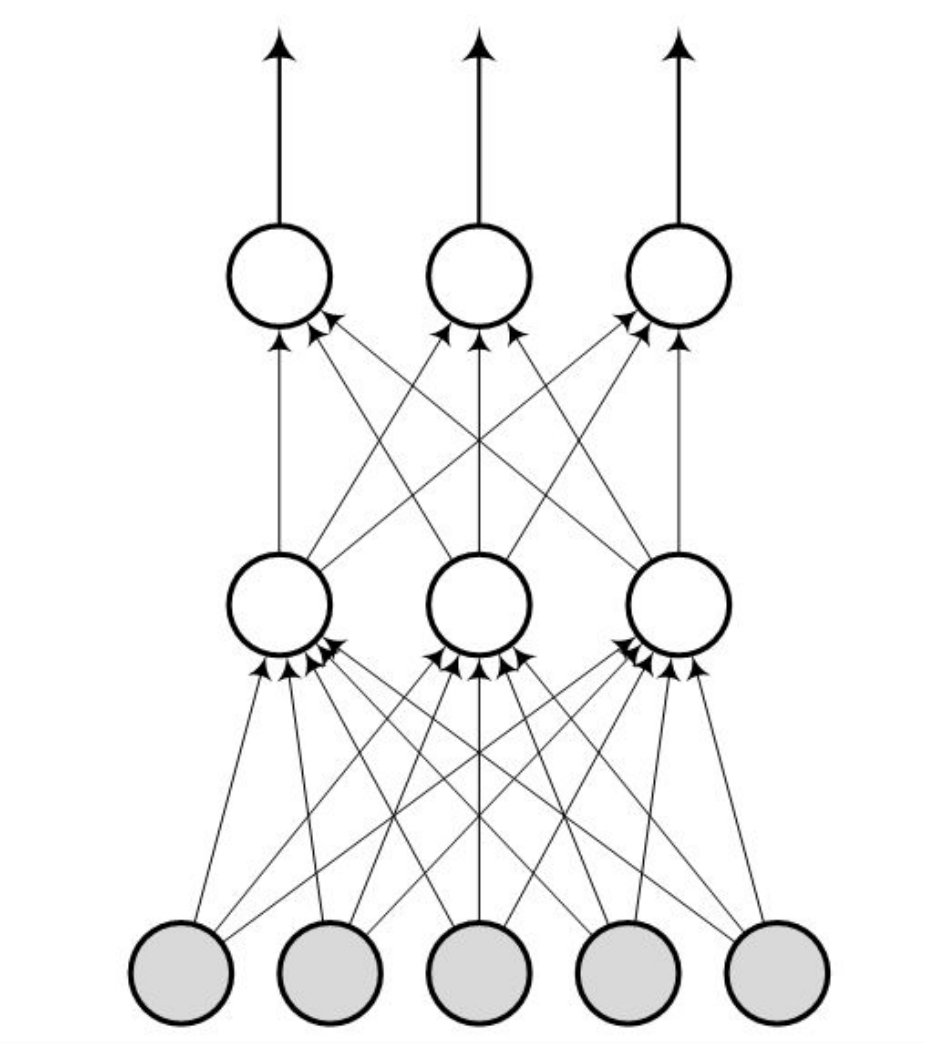
Функции активации

- В наше время появились и другие виды связей (например, механизмы внимания), но в основном сети состоят из таких же искусственных нейронов; только функции активации меняются



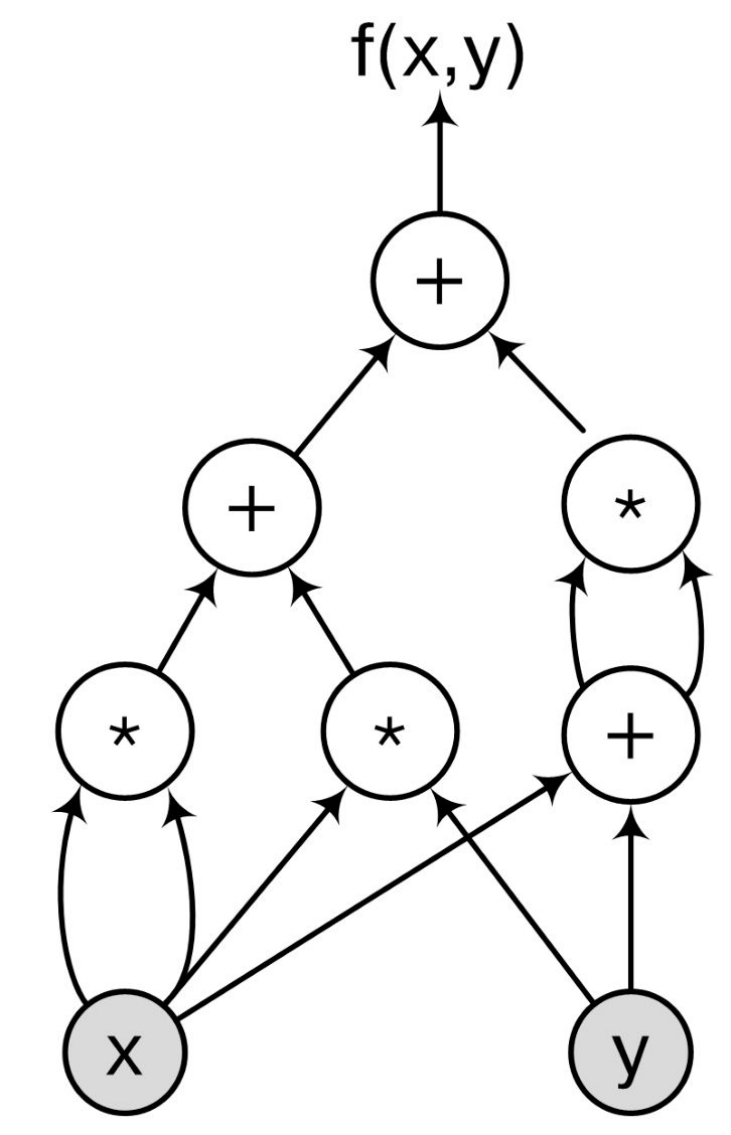
Коннекционизм

- Нейроны соединяются в сети
- Принцип коннекционизма: большая композиция простых объектов обретает новые свойства
- Нейросеть – это граф вычислений, узлы которого – элементарные функции
- Часто такие графы делятся на слои (layers); справа – пример простой сети с двумя слоями

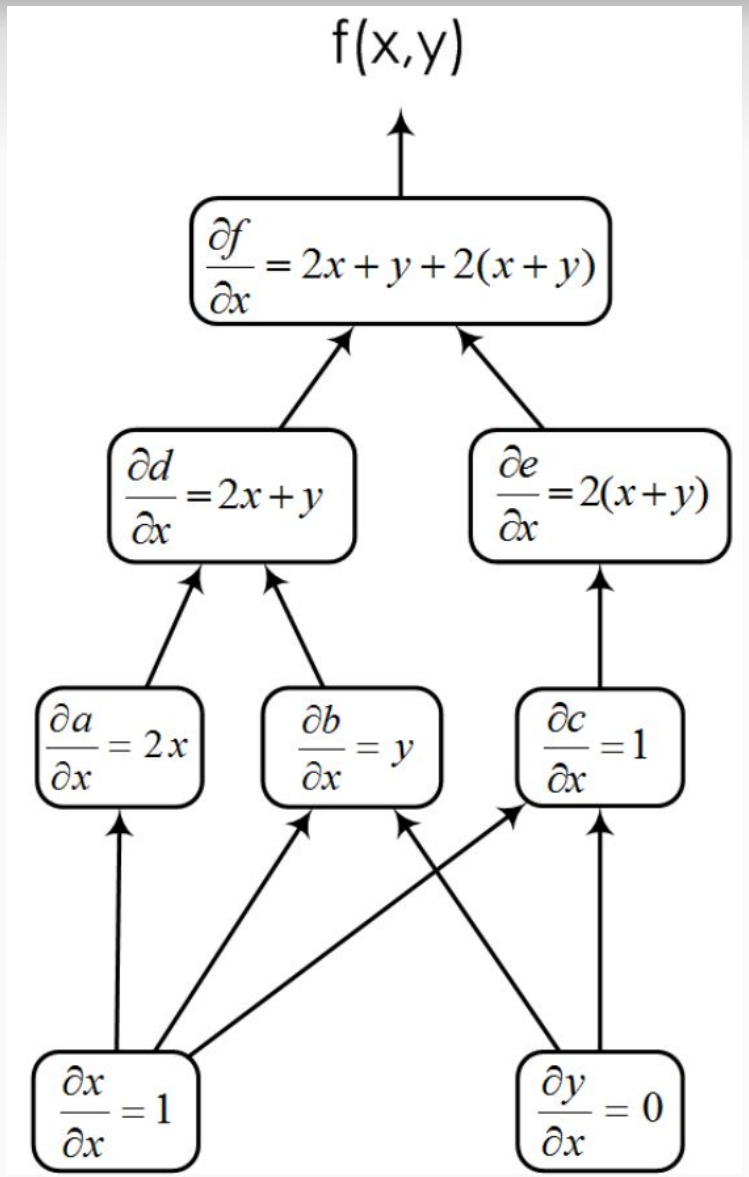


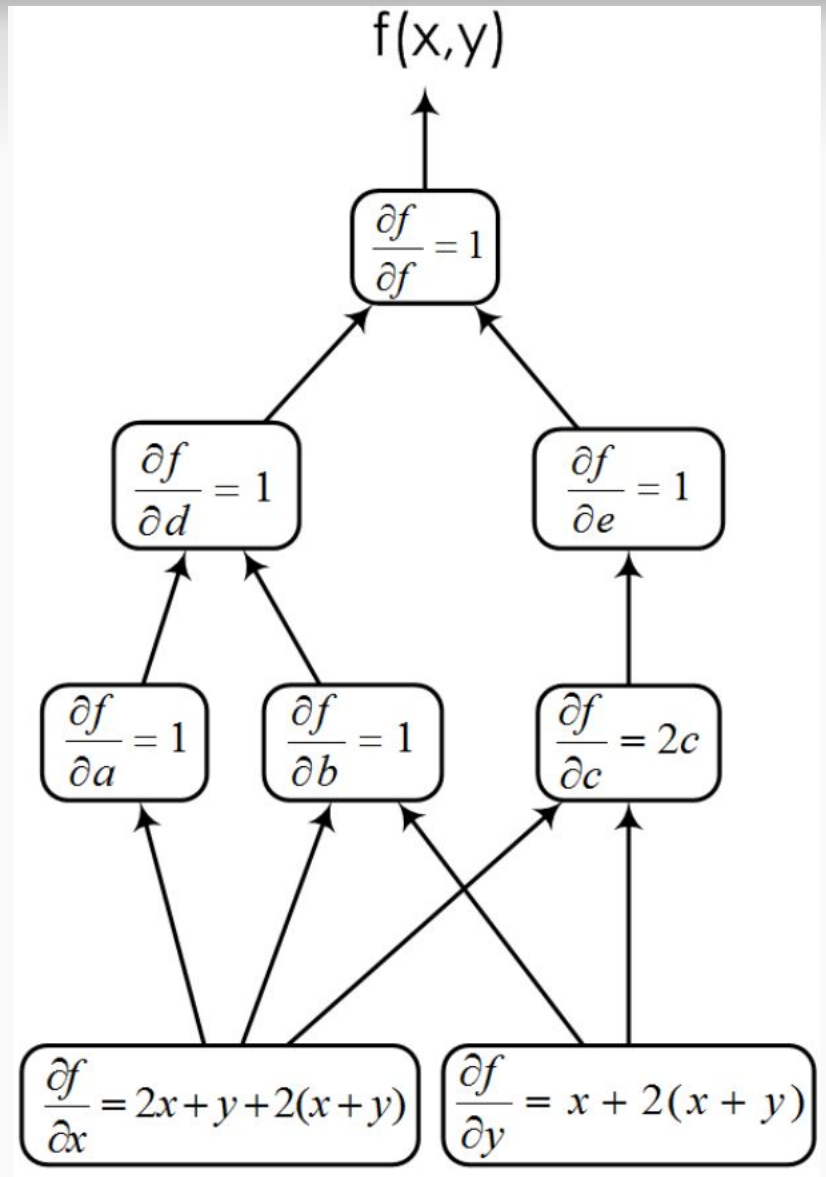
Обратное распространение ошибки

- Нейросети обучаются градиентным спуском
- Чтобы его реализовать, по сути нужно только уметь считать градиенты, т.е. брать производные по некоторым входам графа вычислений
- Давайте разберём пример и поймём, как это работает…



Обратное распространение ошибки



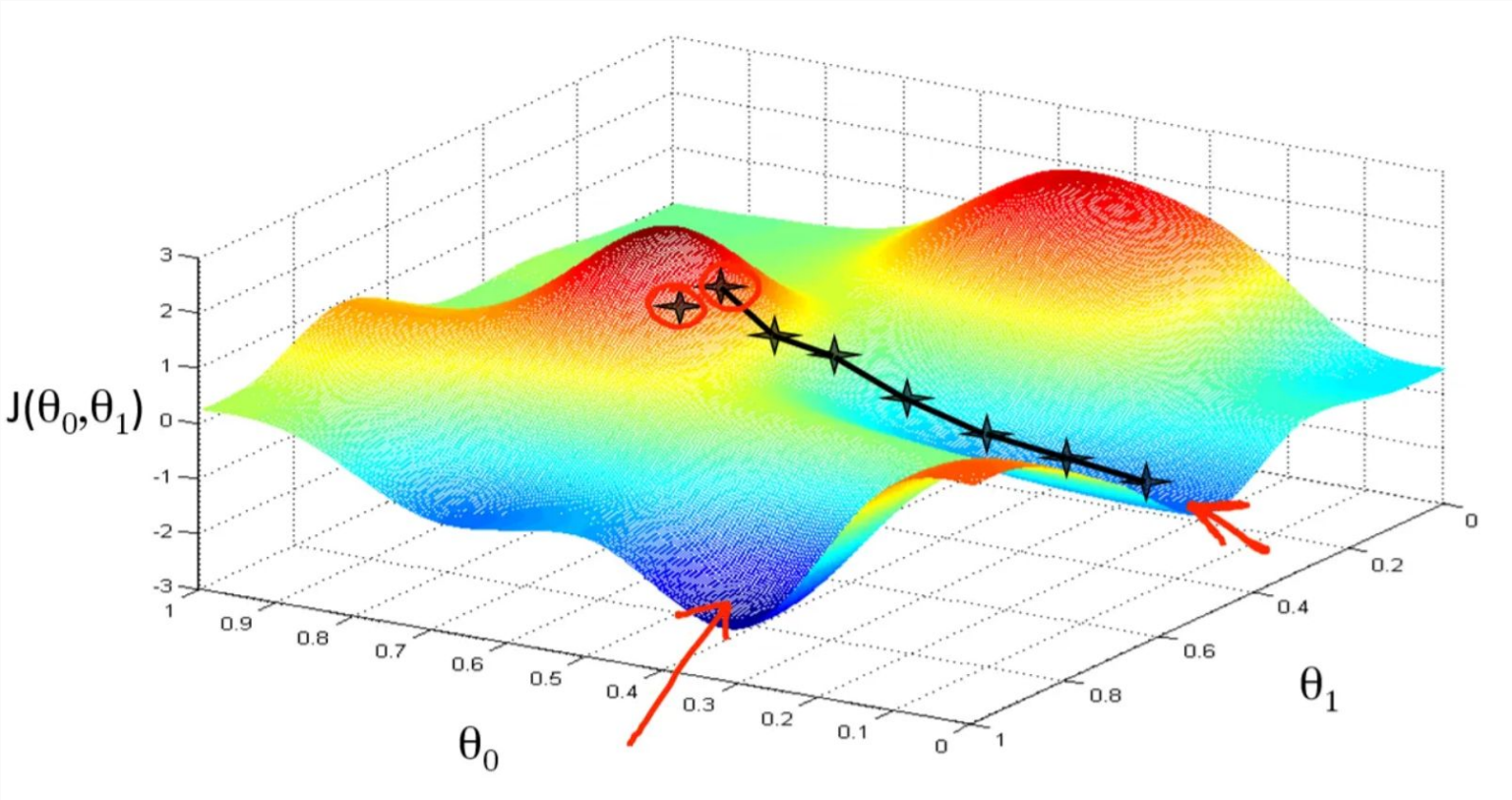


Градиентный спуск



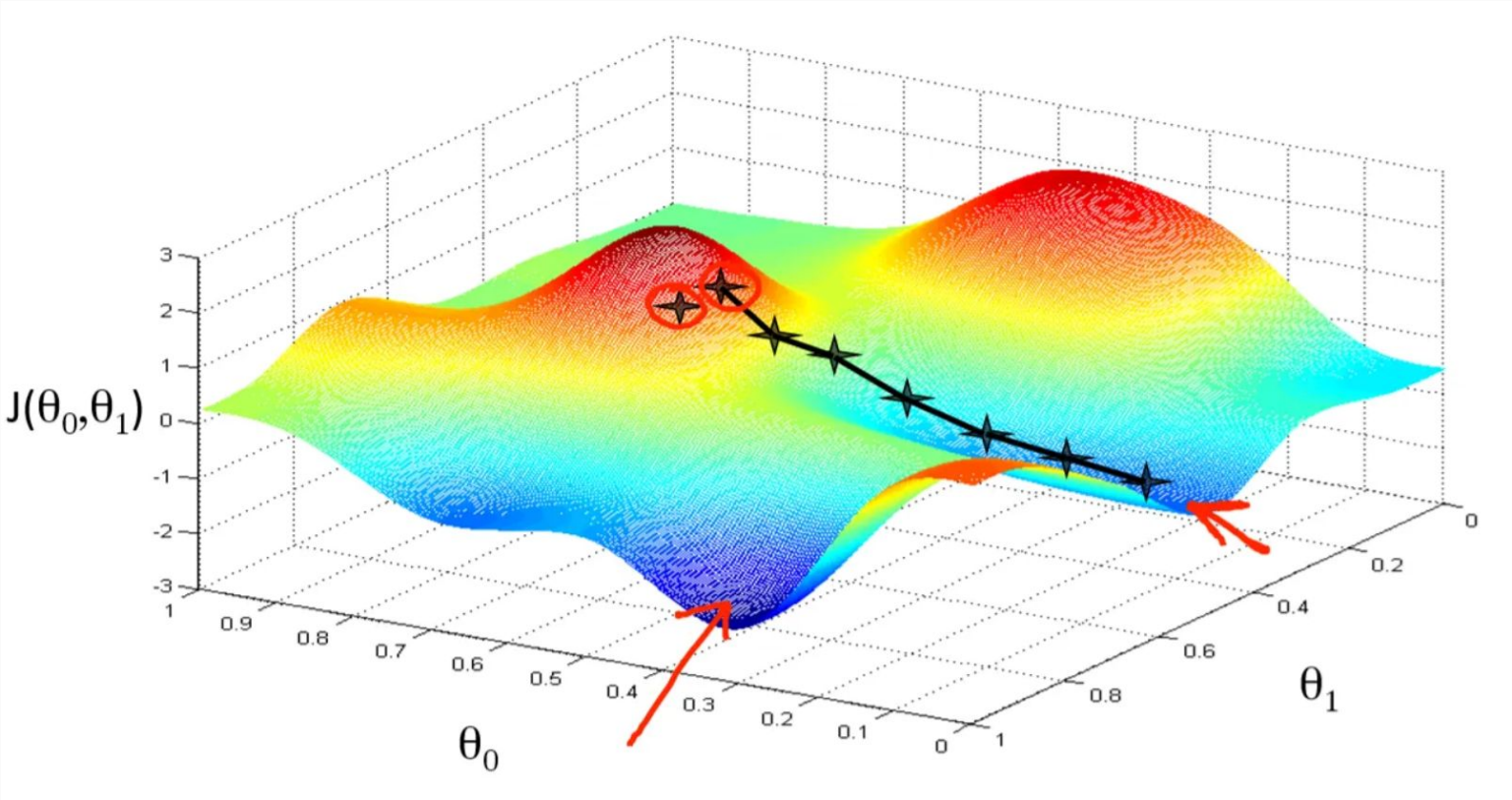
Градиентный спуск

- А потом происходит градиентный спуск, о котором мы уже говорили. Точнее…



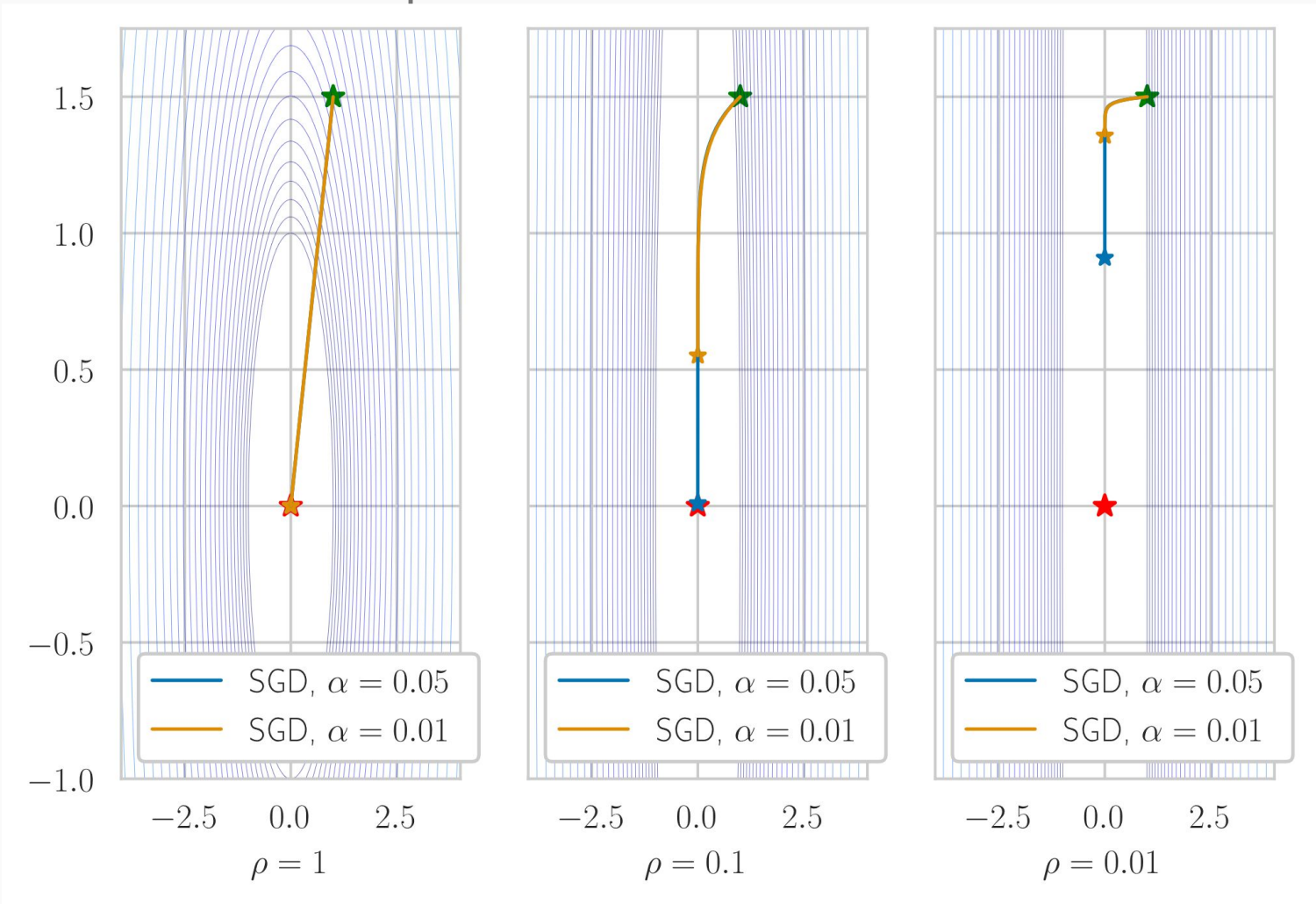
Градиентный спуск

- А потом происходит градиентный спуск, о котором мы уже говорили. Точнее, *стохастический градиентный спуск* (stochastic gradient descent, SGD), потому что вычислить всю функцию ошибки было бы слишком дорого



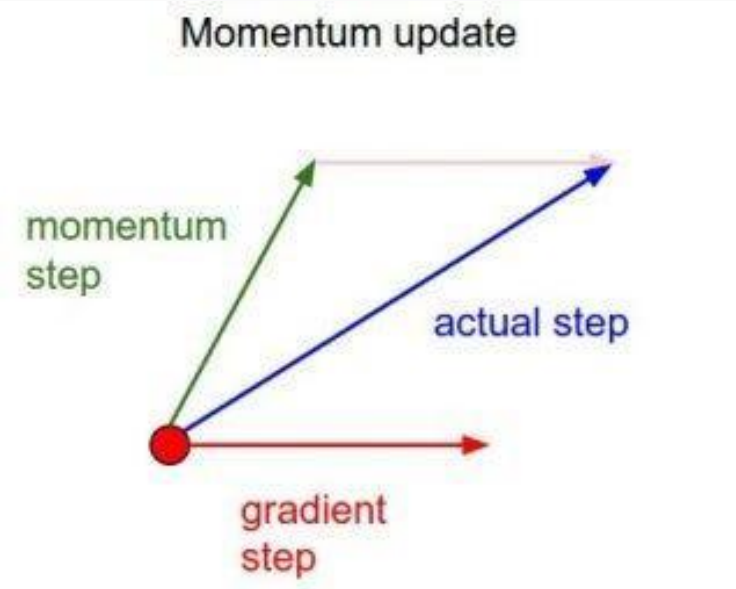
Градиентный спуск

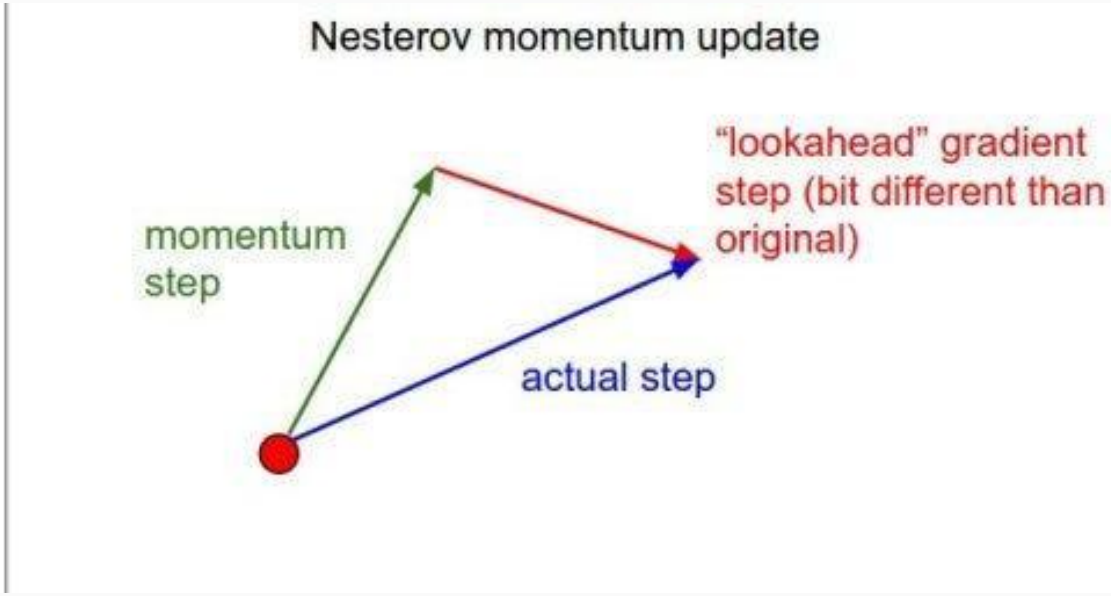
- Главные проблемы начинаются в высоких размерностях, при разном масштабе переменных:



Варианты градиентного спуска

- Варианты градиентного спуска — *метод моментов*
- И *моменты Нестерова*
- Но это только начало, в современном DL много адаптивных алгоритмов…





Варианты градиентного спуска

- От Adagrad…
- …через RMSprop и Adadelta до Adam…
- …а сейчас ещё Muon, Lion, и на этом наверняка прогресс ещё не закончился

$$\mathbf{w}_{t+1,i} = \mathbf{w}_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot g_{t,i}$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t,$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$
$$u_t = \frac{\eta}{\sqrt{v + \epsilon}} m_t.$$

```
def train(w, g, m, v, lr):
  g = clip(g, lr)
  g = arcsin(g)
  m = interp(g, v, 0.899)
  m2 = m * m
  v = interp(g, m, 1.109)
  abs_m = sqrt(m2)
  update = m / abs_m
  wd = w * 0.4602
  update = update + wd
  lr = lr * 0.0002
  m = cosh(update)
  update = update * lr
  return update, m, v
```

Algorithm 2 Muon

Require: Learning rate η, momentum μ

1: Initialize $B_0 \leftarrow 0$
2: **for** $t = 1, \ldots$ **do**
3: Compute gradient $G_t \leftarrow \nabla_\theta \mathcal{L}_t(\theta_{t-1})$
4: $B_t \leftarrow \mu B_{t-1} + G_t$
5: $O_t \leftarrow \text{NewtonSchulz5}(B_t)$
6: Update parameters $\theta_t \leftarrow \theta_{t-1} - \eta O_t$
7: **end for**
8: **return** θ_t

ИТМО



@SINECOR

Спасибо за внимание!



