«Машинное обучение и анализ данных»

Линейные методы: Часть 1. – Линейная и логистическая регрессии

Александр Дьяконов

19 октября 2020 года

План

Линейная регрессия
Решение проблемы вырожденности
Регуляризация, гребневая регрессия, LASSO, Elastic Net
Устойчивая регрессия
Градиентный метод обучения
Линейные скоринговые модели в задаче бинарной классификации
Логистическая регрессия

Линейные решающие модели в задаче бинарной классификации
Идея максимального зазора
SVM
Линейный дискриминант Фишера

Линейная регрессия

Гипотеза о линейной зависимости целевой переменной

Ищем решение в виде

$$a(X_1,\ldots,X_n) = w_0 + w_1 X_1 + \ldots + w_n X_n$$

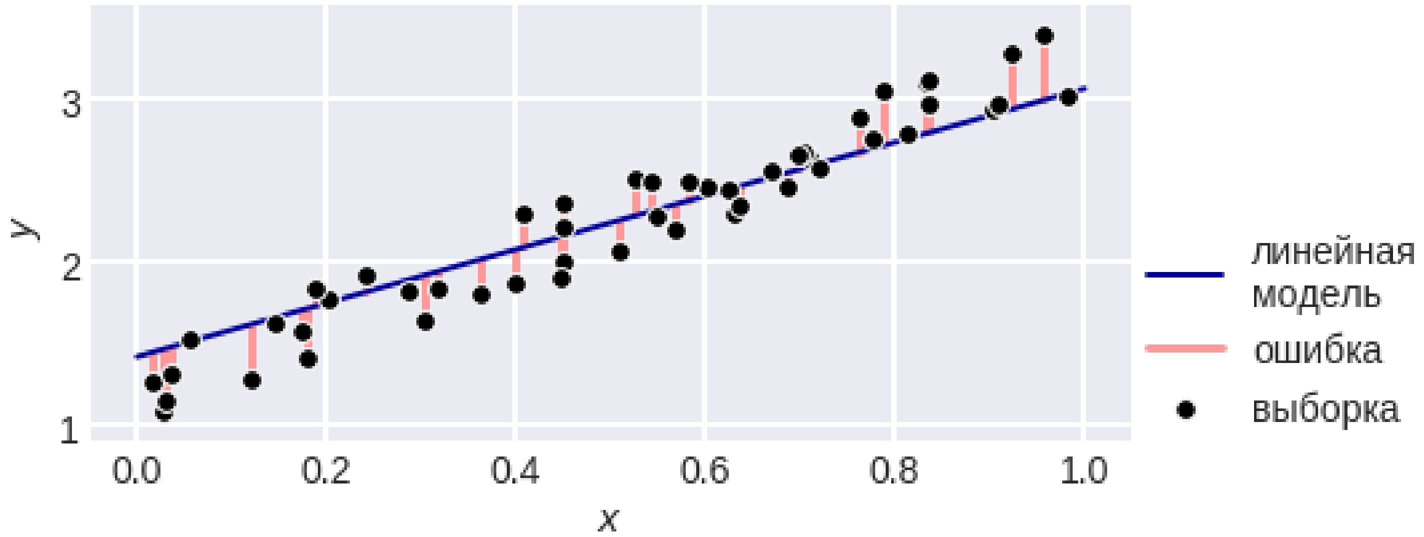
Практика:

- **часто неплохо работает и при монотонных зависимостях**
- **хорошо работает, когда есть много «однородных» зависимостей:**

цель – число продаж
признак 1 – число заходов на страницу продукта
признак 2 – число добавлений в корзину
признак 3 – число появлений продукта в поисковой выдачи
...

Линейная регрессия от одной переменной

$$a(X_1) = w_0 + w_1 X_1$$



обучение: $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, $x_i \in \mathbb{R}$,

$$\begin{cases} w_0 + w_1 x_1 = y_1 \\ \ldots \\ w_0 + w_1 x_m = y_m \end{cases}$$

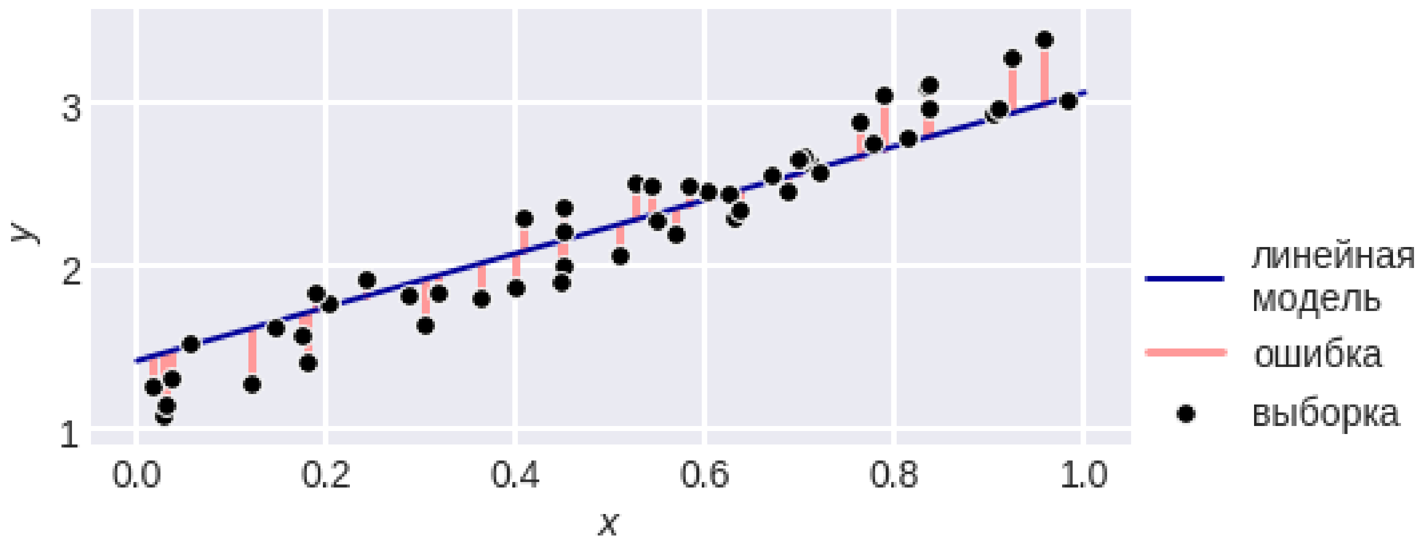
невязки/отклонения (residuals):

$$\begin{cases} e_1 = y_1 - w_0 - w_1 x_1 \\ \ldots \\ e_m = y_m - w_0 - w_1 x_m \end{cases}$$

Линейная регрессия от одной переменной

Задача минимизации суммы квадратов отклонений (residual sum of squares)

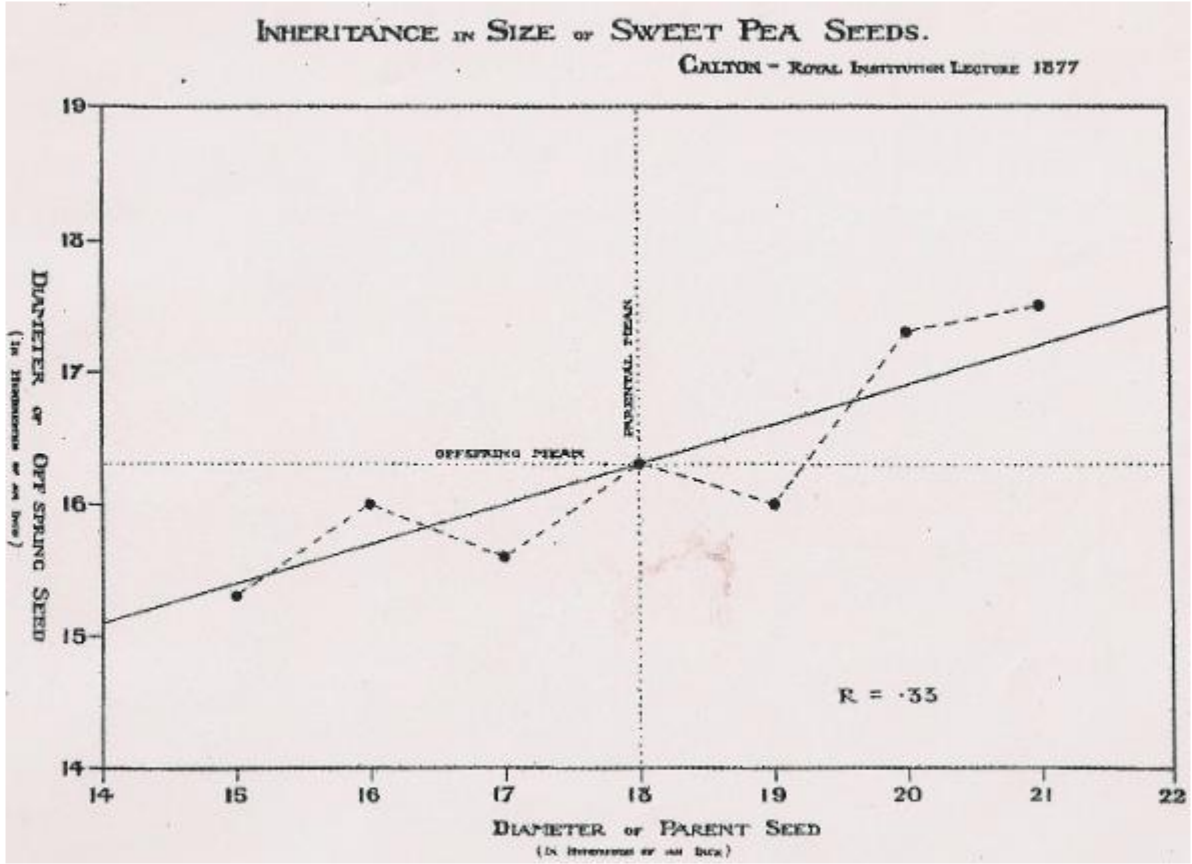
$$\mathrm{RSS} = e_1^2 + \ldots + e_m^2 \to \min$$



~ задача описания данных гиперплоскостью (но ф-л качества!)

Есть вероятностное обоснование, но пока… логично

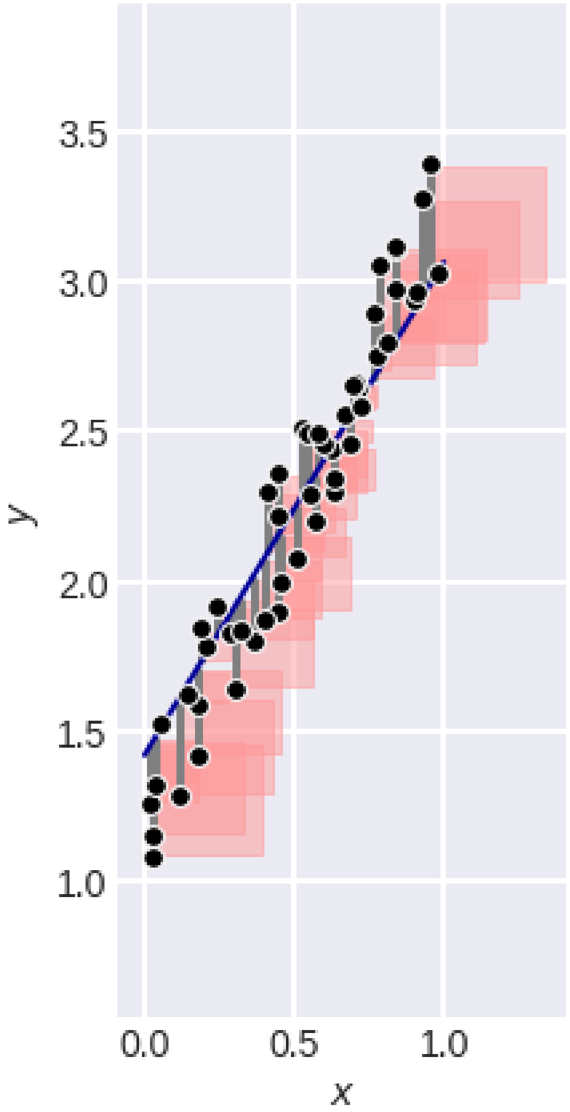
Линейная регрессия от одной переменной

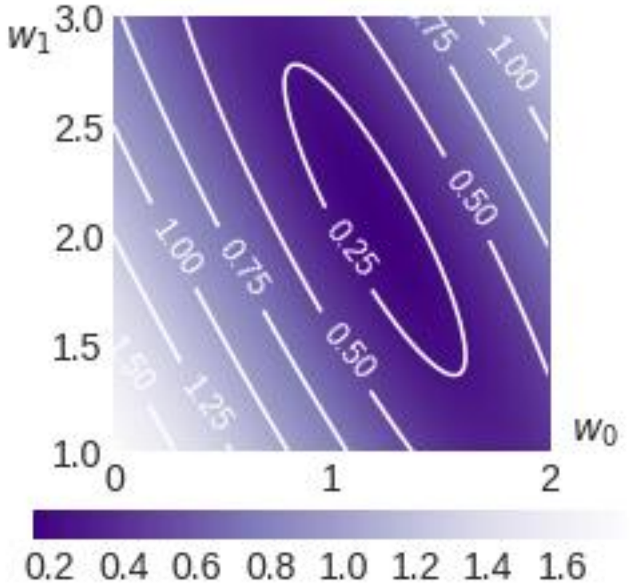


Francis Galton, 1877

Линейная регрессия от одной переменной

Геометрический смысл ошибки





Отличается от суммы расстояний до поверхности!

Линейная регрессия от одной переменной

Нетрудно показать (ДЗ):

$$w_1 = \frac{\sum_{i=1}^{m}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{m}(x_i - \overline{x})^2} = \frac{\mathrm{cov}(\{x_i\},\{y_i\})}{\mathrm{var}(\{x_i\})},$$

$$w_0 = \overline{y} - w_1\overline{x}.$$

где $\overline{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$, $\overline{y} = \frac{1}{m}\sum_{i=1}^{m} y_i$.

Общий случай (многих переменных)

$$a(X_1,\ldots,X_n) = w_0 + w_1X_1 + \cdots + w_nX_n = x^{\mathrm{т}}w$$

$$w = (w_0, w_1, \ldots, w_n)^{\mathrm{т}}$$
$$x = (X_0, X_1, \ldots, X_n)^{\mathrm{т}}$$

для удобства записи вводим фиктивный признак $X_0 \equiv 1$

обучение: $\{(x_1, y_1), \ldots, (x_m, y_m)\}$**,** $x_i \in \mathbf{R}^{n+1}$**,**

$$\begin{cases} x_1^{\mathrm{т}}w = y_1 \\ \ldots \\ x_m^{\mathrm{т}}w = y_m \end{cases}$$

$Xw = y$ **– как решать?**

Общий случай (многих переменных)

или в матричной форме

$$Xw = y$$

в матрице X по строкам записаны описания объектов,
в векторе y значения их целевого признака
(здесь есть коллизия в обозначении y)

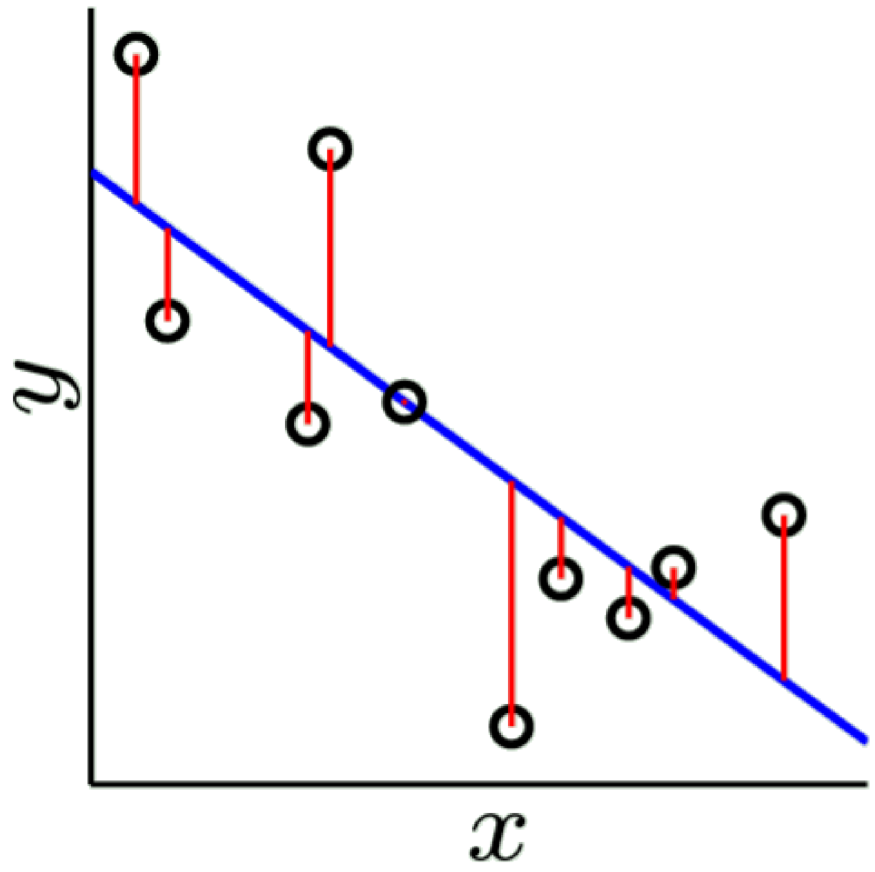
будем решать так:

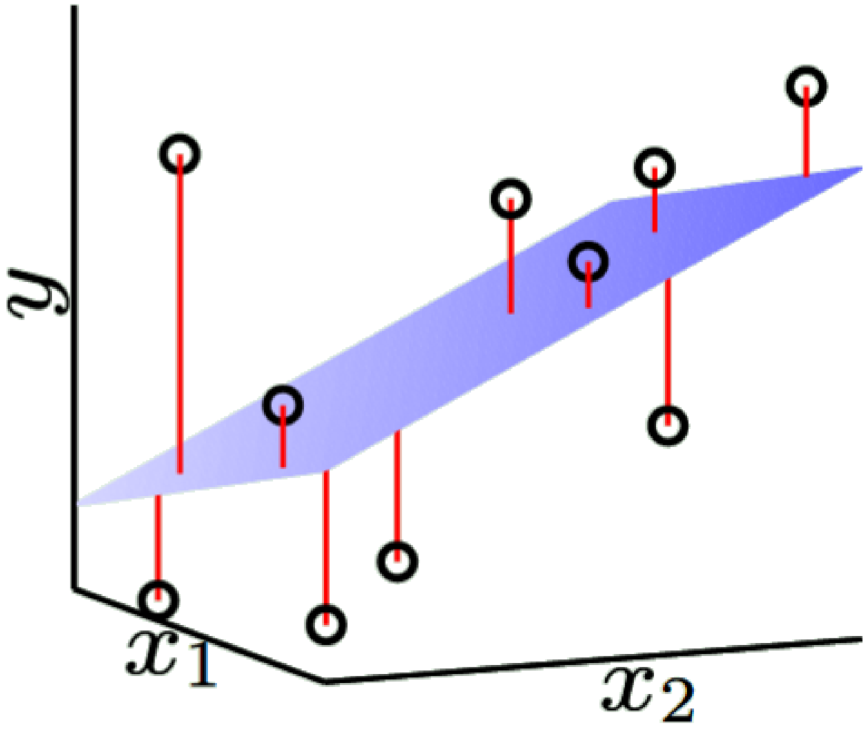
$$\| Xw - y \|_2^2 \to \min_w$$

почему?

Общий случай (многих переменных)

геометрический смысл





Решение задачи минимизации: прямой метод

$$\| Xw - y \|_2^2 \to \min_w$$

$$\| Xw - y \|_2^2 = (Xw - y)^{\mathrm{т}}(Xw - y) =$$
$$w^{\mathrm{т}}X^{\mathrm{т}}Xw - w^{\mathrm{т}}X^{\mathrm{т}}y - y^{\mathrm{т}}Xw + y^{\mathrm{т}}y$$

$$\nabla \| Xw - y \|_2^2 = 2X^{\mathrm{т}}Xw - 2X^{\mathrm{т}}y = 0$$
$$X^{\mathrm{т}}Xw = X^{\mathrm{т}}y$$
$$w = (X^{\mathrm{т}}X)^{-1}X^{\mathrm{т}}y$$

решение существует, если столбцы линейно независимые

$(X^{\mathrm{т}}X)^{-1}X^{\mathrm{т}}$ **– псевдообратная матрица Мура-Пенроуза**

обобщение обратной на неквадратные матрицы

Обобщённая линейная регрессия

вместо X – что угодно

$$a(X_1,\ldots,X_n)=w_0+w_1\varphi_1(X_1,\ldots,X_n)+\cdots+w_k\varphi_k(X_1,\ldots,X_n)$$

$$w=(w_0,w_1,\ldots,w_k)^{\mathrm{т}}$$

$$x=(X_0,X_1,\ldots,X_n)^{\mathrm{т}}$$

$$\varphi(x)=(\underset{\equiv 1}{\varphi_0(x)},\varphi_1(x),\ldots,\varphi_k(x))^{\mathrm{т}}$$

$$a(x)=\sum_{i=1}^{k}w_i\varphi_i(x)=\varphi(x)^{\mathrm{т}}w$$

базисные функции (basis functions)

они фиксированы

Подробности в нелинейных методах…

Проблема вырожденности матрицы

$$w = (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} y$$

Решения:

1. **Регуляризация – здесь и в «сложности»**
2. **Селекция (отбор) признаков – «селекция»**
3. **Уменьшение размерности (в том числе, PCA) – USL**
4. **Увеличение выборки**

если объектов много – то работать с гигантской матрицей невозможно…
но выдели как это делается в оптимизации онлайн-методами

Регуляризация

Упрощённое объяснение смысла регуляризации

$$a(X_1,\ldots,X_n) = w_0 + w_1 X_1 + \ldots + w_n X_n$$

если есть два похожих объекта, то должны быть похожи метки

пусть отличаются в j-м признаке, тогда ответы модели отличаются на

$$\varepsilon_j w_j$$

Поэтому не должно быть больших весов

(у признаков, по которым могут отличаться похожие объекты)!

П.С. Плохо, когда модель заточена на один признак!

Поэтому вместе с $\|Xw - y\|_2^2 \to \min$

Хотим $\|w\|_2^2 \to \min$

Не на все коэффициенты нужна регуляризация! Почему?

Регуляризация

Пример, пусть

$$y = X_1 = X_1 + w' X_2 - w' X_3 \text{ **при** } X_2 = X_3$$

Если теперь $X_2 \approx X_3$

тогда $\varepsilon = X_2 - X_3$

$$a = X_1 + w'\varepsilon$$

может быть сколь угодно большим при больших w'

аналогично при линейных зависимостях!

автоматически, когда объектов мало (сколько?)

Регуляризация

Иванова

$$\begin{cases} \| Xw - y \|_2^2 \to \min \\ \| w \|_2^2 \le \lambda \end{cases}$$

Тихонова

$$\| Xw - y \|_2^2 + \lambda \| w \|_2^2 \to \min$$

Удобнее: безусловная оптимизация

Всё это справедливо и для общих задач минимизации!

$$\begin{cases} L(a) \to \min \\ \mathrm{complexity}(a) \le \lambda \end{cases}$$

$$L(a) + \lambda\, \mathrm{complexity}(a) \to \min$$

Часто эти две формы эквивалентны:
решение одного можно получить как решение другого

Есть ещё регуляризация Морозова...

$$\| w \|_2^2 = w_1^2 + w_2^2 + \ldots + w_n^2$$

Регуляризация

$$\arg\min \| Xw - y \|_2^2 + \lambda \| w \|_2^2 = (X^{\mathrm{T}} X + \lambda I)^{-1} X^{\mathrm{T}} y$$

ДЗ Доказать!

– гребневая регрессия (Ridge Regression)

Другой смысл – боремся с вырожденностью матрицы!

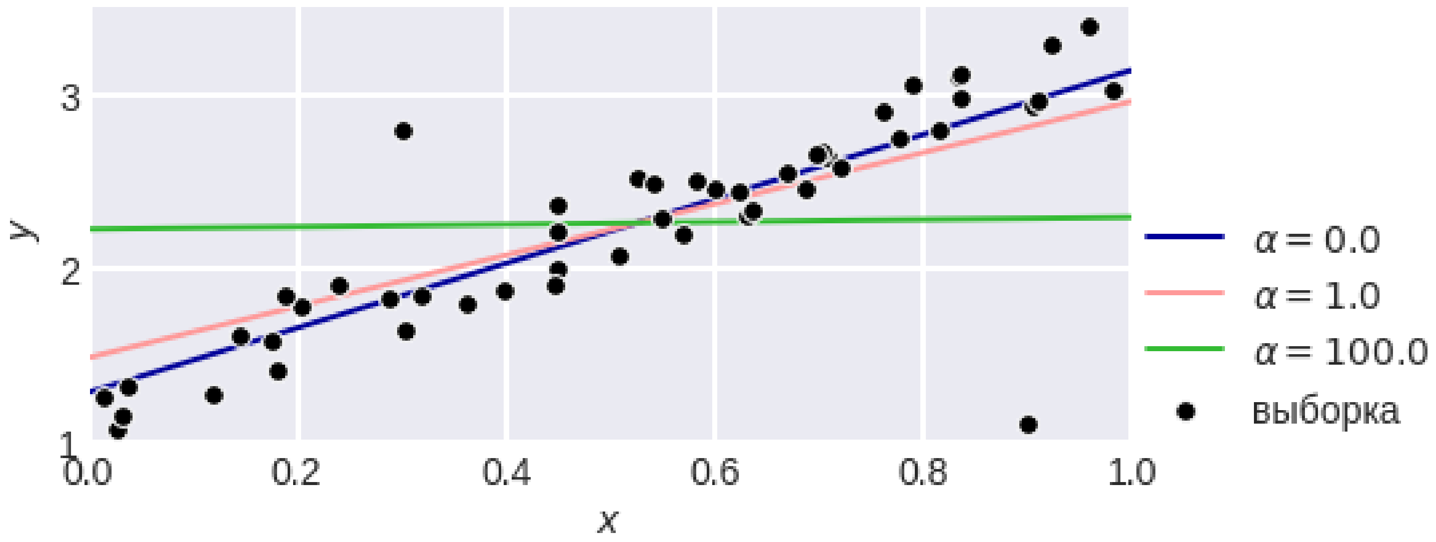
$\lambda = 0$ **– получаем классическое решение**

$\lambda \to +\infty$ **– меньше «затачиваемся на данные» и больше регуляризуем**

Матрица очевидно становится обратимой!

значение параметра регуляризации можно выбрать на скользящем контроле

Регуляризация – минутка кода




```
from sklearn.linear_model import Ridge

model = Ridge(alpha=0.0) # ридж-регрессия
# обучение
model.fit(x_train[:, np.newaxis], y_train)
# обратите внимание: np.newaxis
# контроль
a_train = model.predict(x_train[:, np.newaxis])
a_test = model.predict(x_test[:, np.newaxis])
```

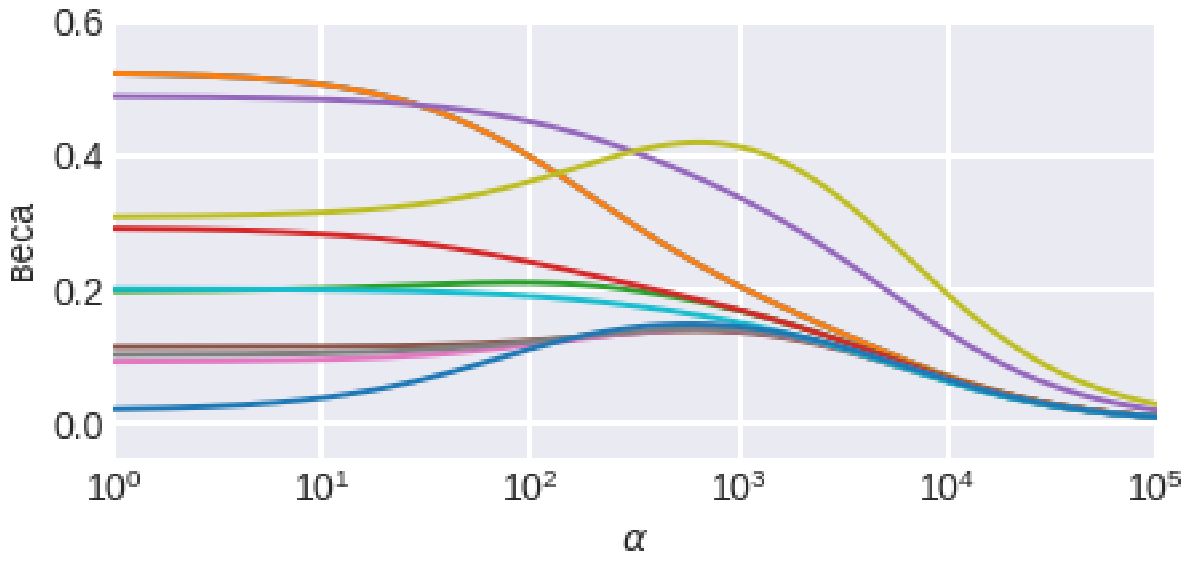
Интересно, что рисунок неудачный – получилась антиреклама регуляризации… почему?

Ridge-регрессия

$$\sum_{i=1}^{m}\left(y_i - a(x_i)\right)^2 + \lambda\sum_{j=1}^{n} w_j^2 \to \min$$

$$\lambda \geq 0$$

добавление shrinkage penalty (регуляризатора)



параметр регуляризации может подбираться с помощью скользящего контроля

Ridge-регрессия

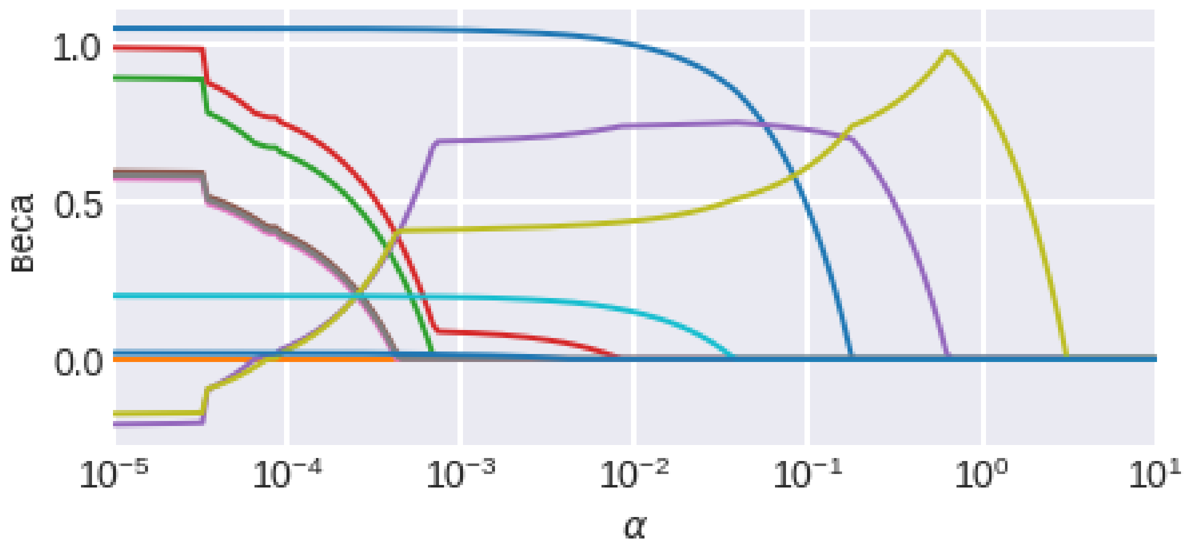
Для ridge-регрессии нужна правильная нормировка признаков!
Нет инвариантности (в отличие от линейной) от умножения признаков на скаляры

Перед регуляризацией – стандартизация!!!

LASSO

$$\sum_{i=1}^{m}\left(y_i - a(x_i)\right)^2 + \lambda\sum_{j=1}^{n}|w_j| \to \min$$

$$\lambda \geq 0$$



Здесь коэффициенты интенсивнее зануляются при увеличении $\lambda \geq 0$.

Эксперименты с одинаковыми и зависимыми признаками

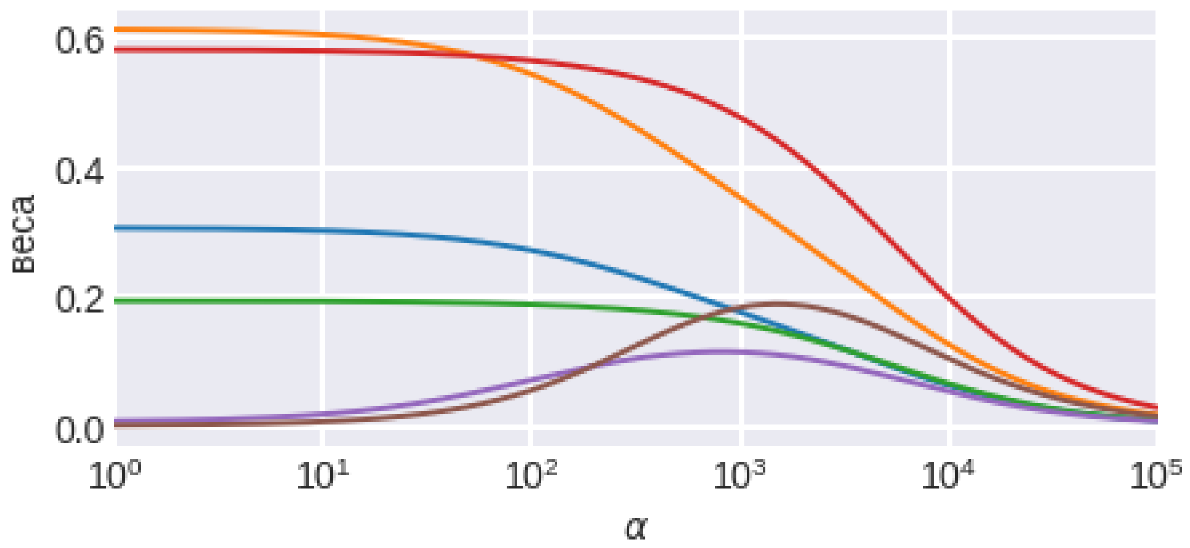
здесь была задача

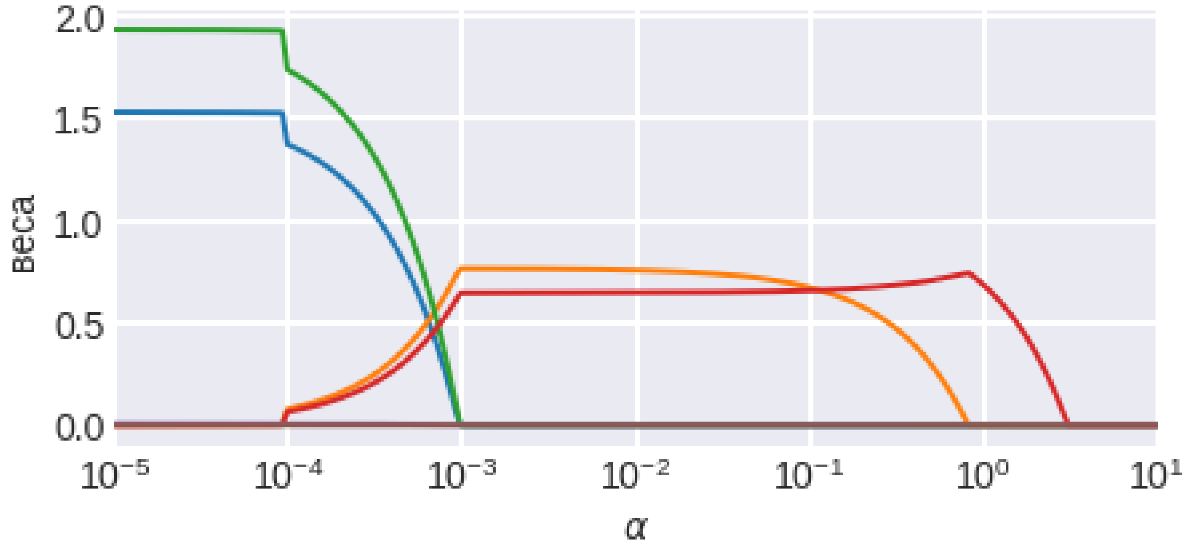
```
X = np.random.rand(1000, 11)
X[:,1] = X[:,0]
X[:,4] = X[:,2] + X[:,3]
X[:,8] = X[:,5] + X[:,6] + X[:,7]
y = X[:,0] + 0.8*X[:,4] + 0.4*X[:,8] + 0.2*X[:,9] +
             0.5*np.random.randn(1000)
```

зависит от масштаба признаков,
но из-за предварительной нормировки этот эффект не наблюдается

Масштаб очень важен! см. дальше

Эксперименты с одинаковыми и зависимыми признаками





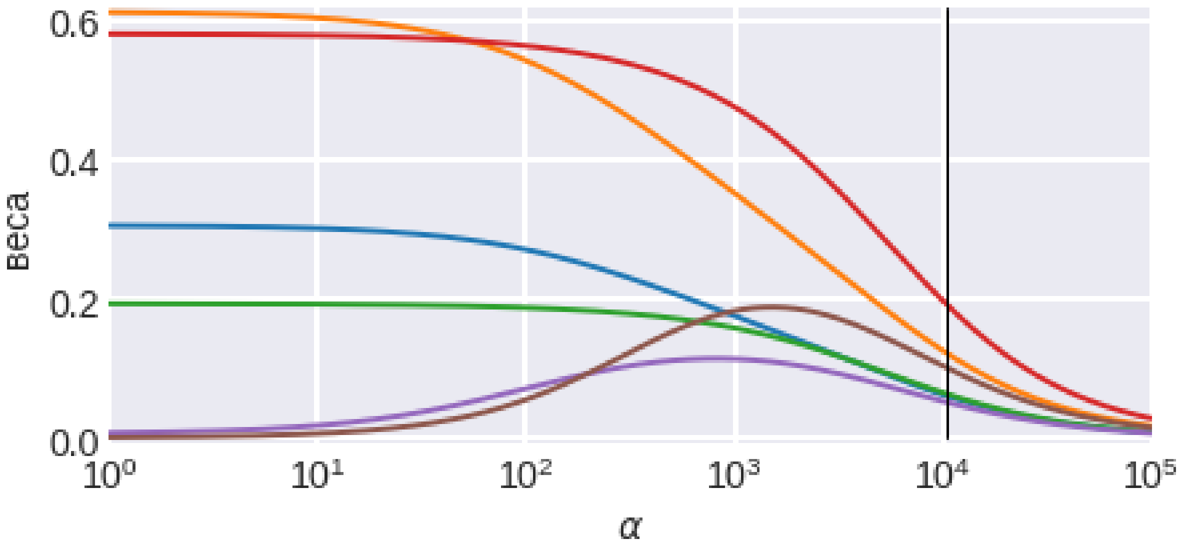

```
np.random.seed(10)
X = np.random.rand(1000, 6)
X[:,1] = X[:,0]
X[:,2] = X[:,3]

X[:,0] = 1 * X[:,0]
X[:,1] = 2 * X[:,1]
X[:,2] = 1 * X[:,2]
X[:,3] = 3 * X[:,3]
X[:,4] = 1 * X[:,4]
X[:,5] = 2 * X[:,5]

y = 1.5 * X[:,0] + 2*X[:,2] +
0.5*np.random.randn(1000)
```

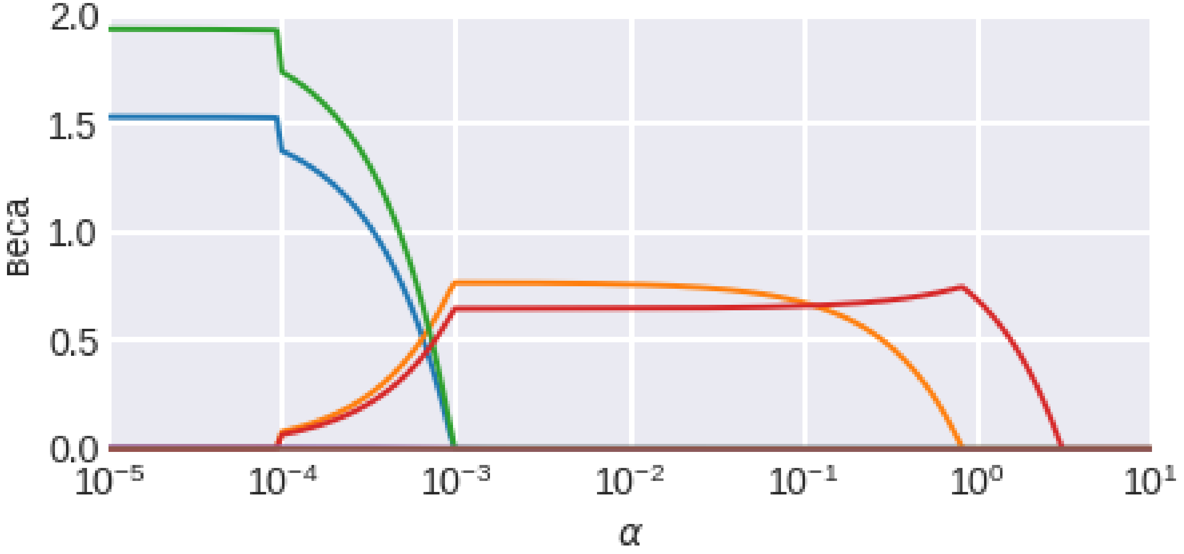
Эксперименты с одинаковыми и зависимыми признаками

$$Y = 1.5X_1 + 2X_3 = 0.75X_2 + 0.66X_4$$



$$\lambda = 1$$
$$Y = 0.31X_1 + 0.61X_2 + 0.19X_3 + \\ +0.58X_4 + 0.01X_5 + 0.0X_6$$

$$\lambda \sim 10500$$
$$Y = 0.06X_1 + 0.12X_2 + 0.06X_3 + \\ +0.19X_4 + 0.05X_5 + 0.1X_6$$



$$\lambda = 10^{-5}$$
$$Y = 1.53X_1 + 1.94X_3$$

$$\lambda \sim 0.01$$
$$Y = 0.76X_2 + 0.65X_4$$

Эксперименты с одинаковыми и зависимыми признаками

веса зависят от масштаба признаков

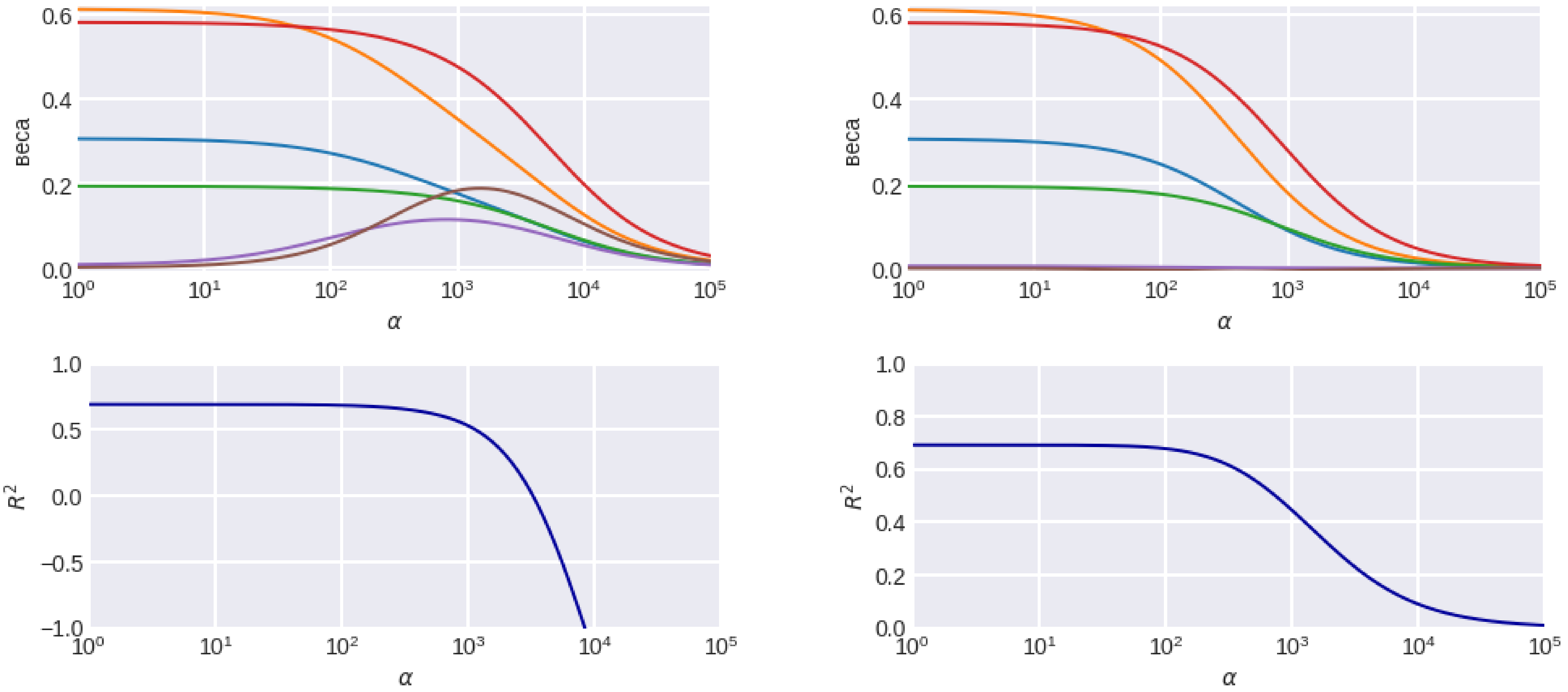
при сильной регуляризации меняется распределение весов зависимых признаков

Пусть

$$Y = 4X_1,\ X_1 = X_2$$

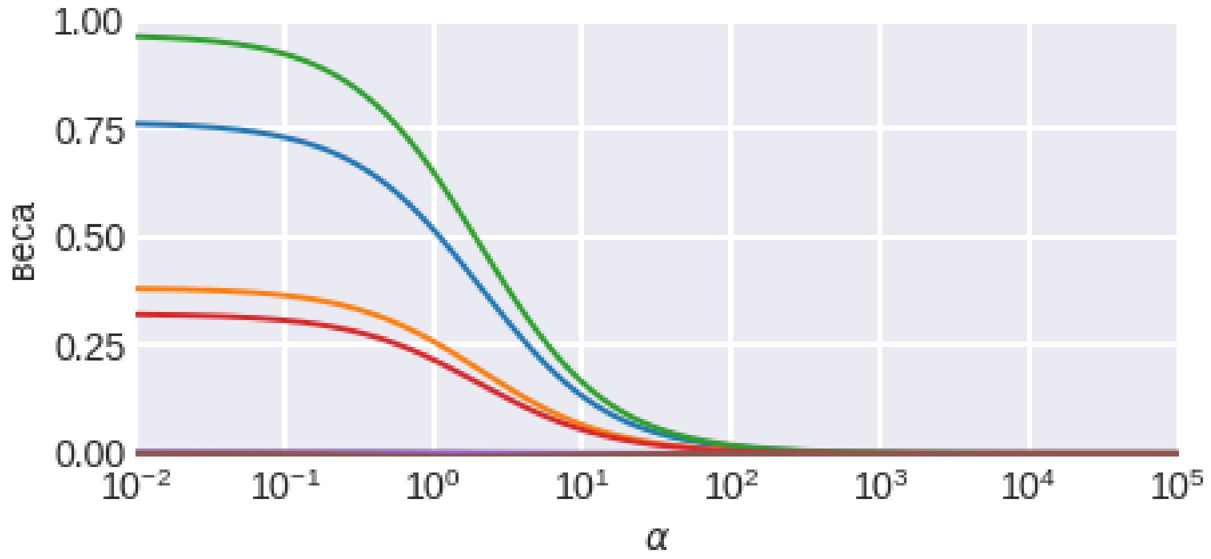
w_1	w_2	$\|\|w\|\|_1$	$\|\|w\|\|_2^2$
5	**– 1**	**6**	**26**
4	**0**	**4**	**16**
3	**1**	**4**	**10**
2	**2**	**4**	**8**

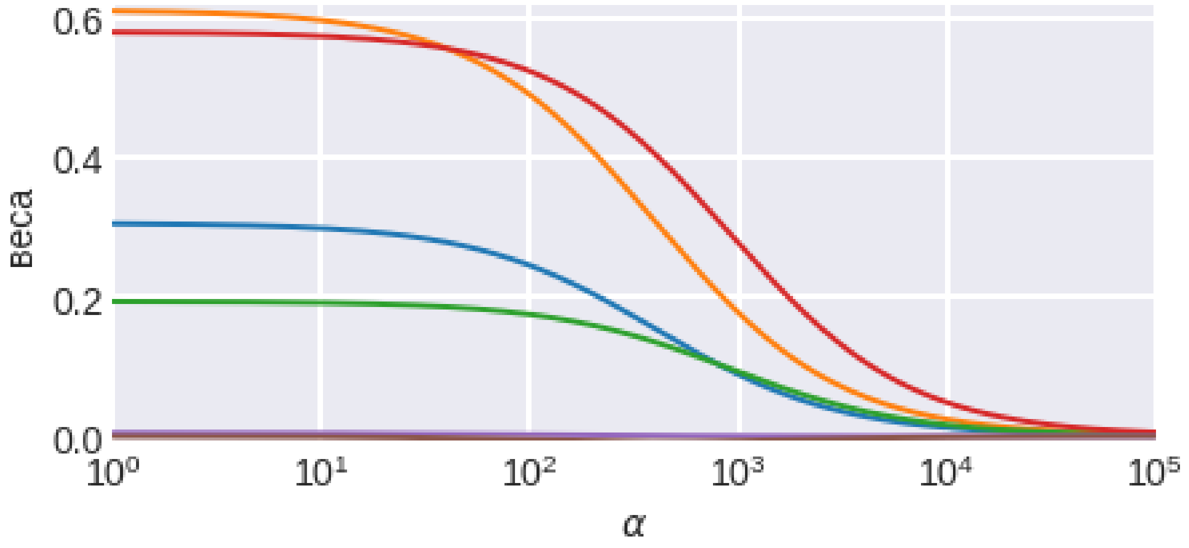
Эксперименты с одинаковыми и зависимыми признаками: L_2-регуляризация

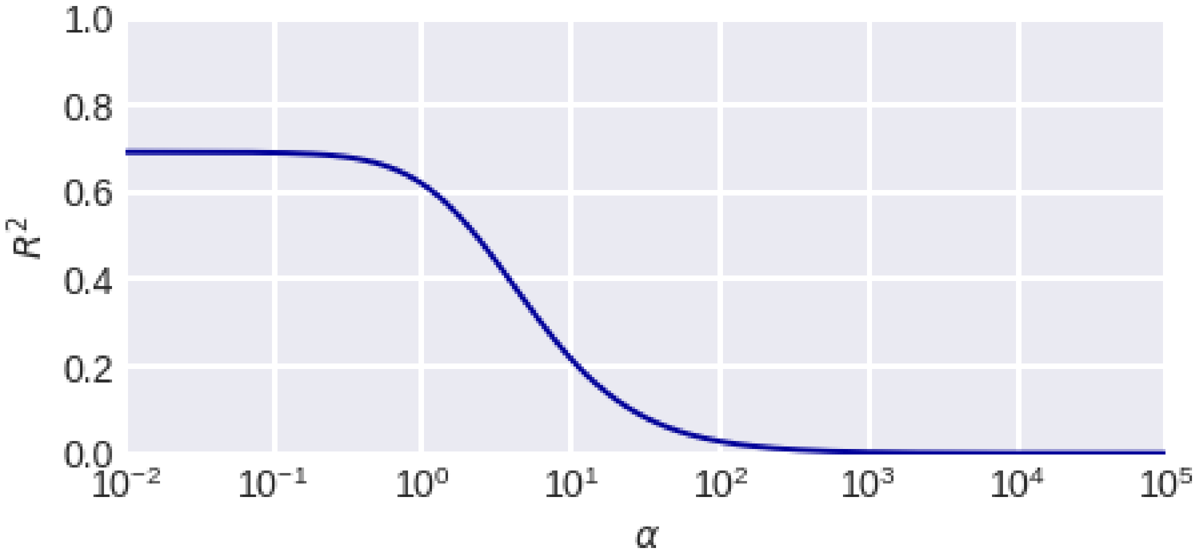


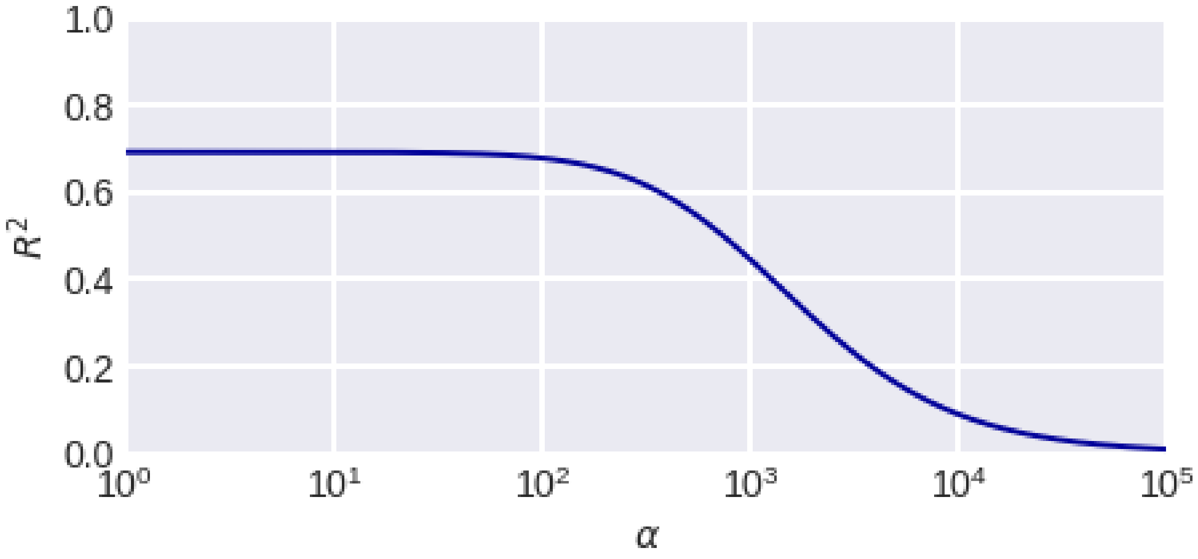
`fit_intercept=True`

Эксперименты с одинаковыми и зависимыми признаками: L_2-регуляризация





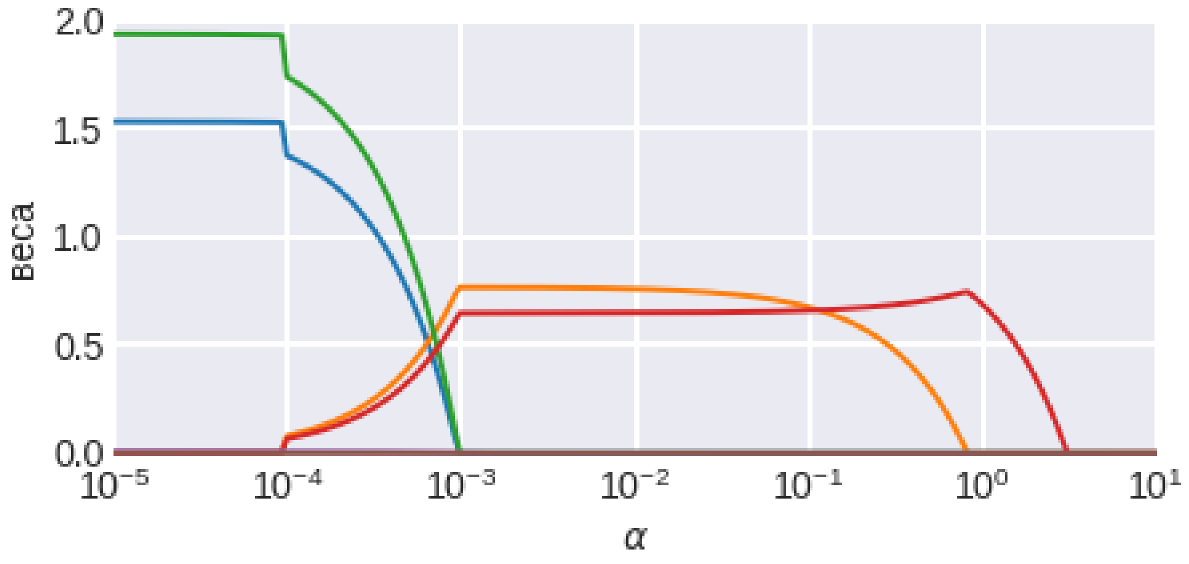


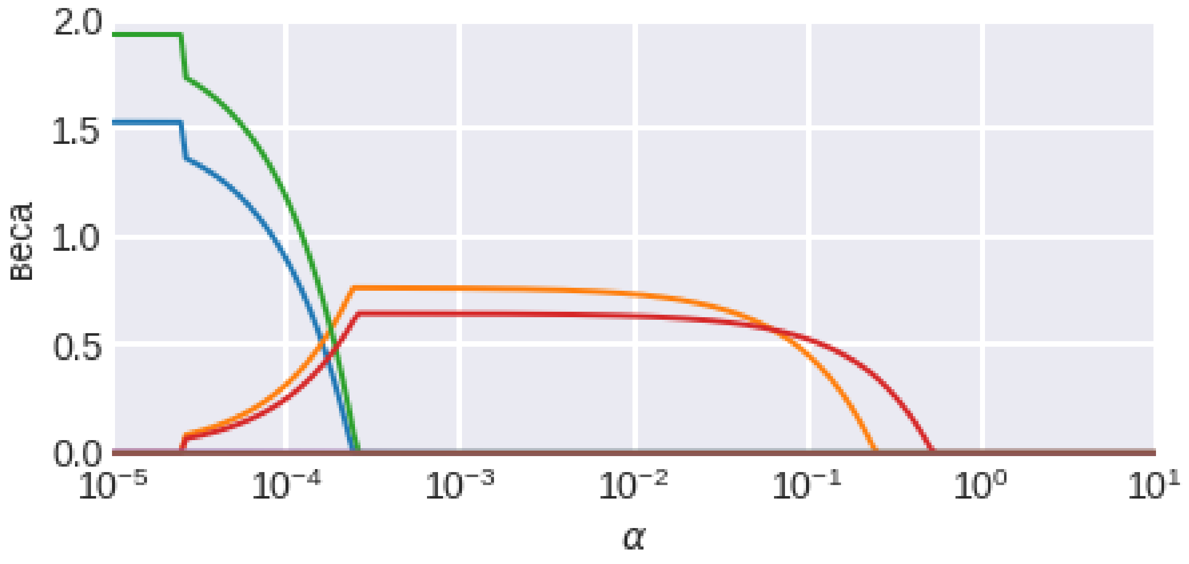


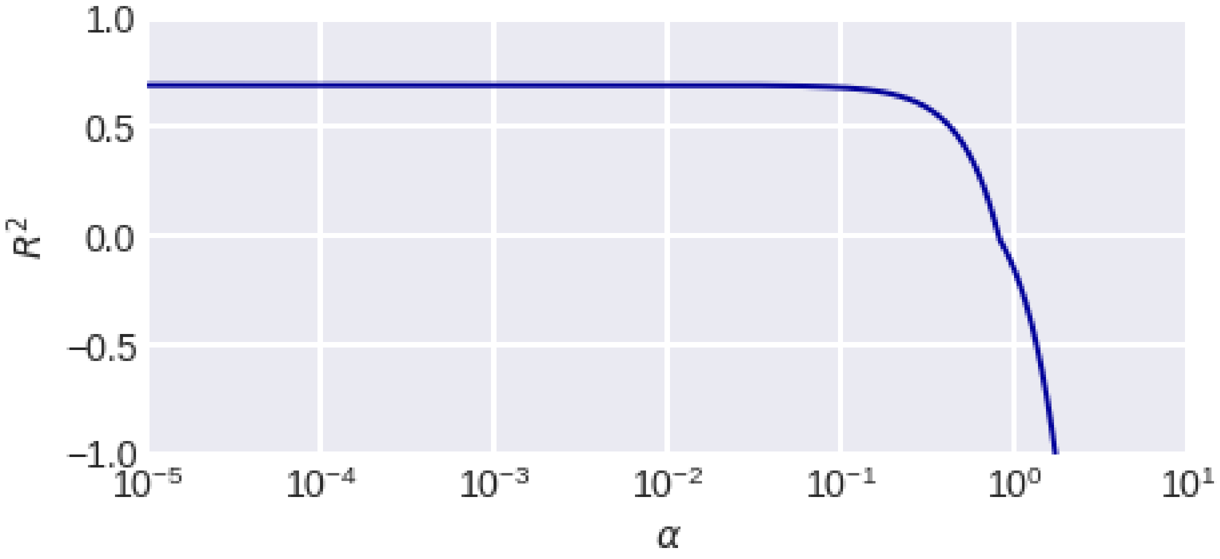
`fit_intercept=True, normalize=True`

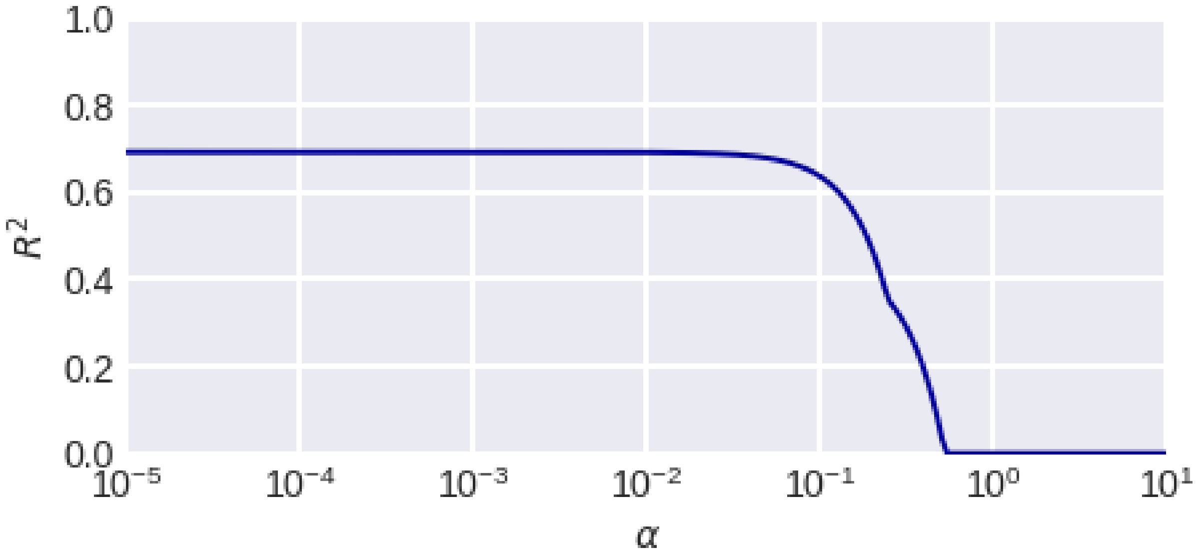
`fit_intercept=True`

Эксперименты с одинаковыми и зависимыми признаками: L_1-регуляризация



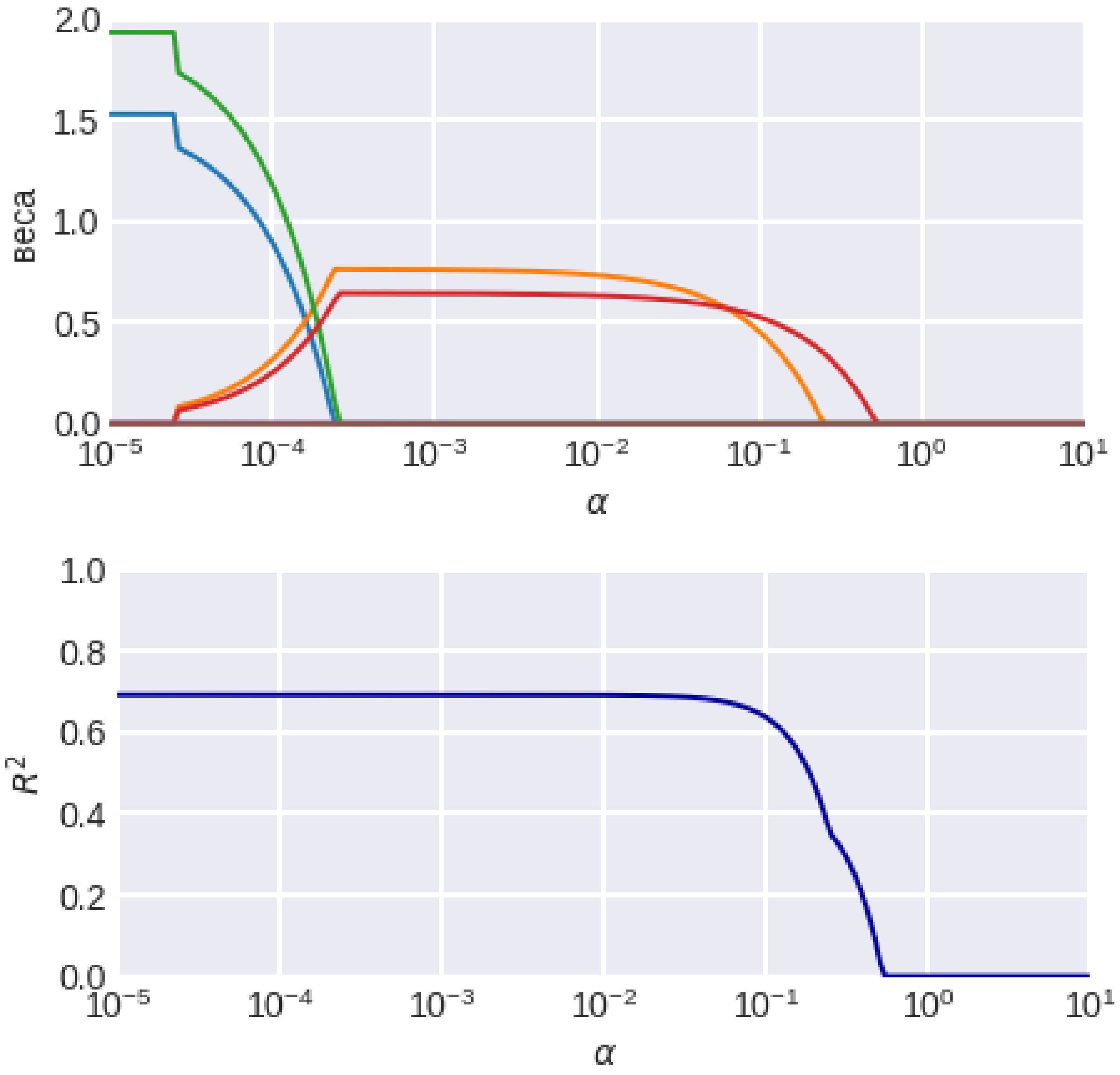




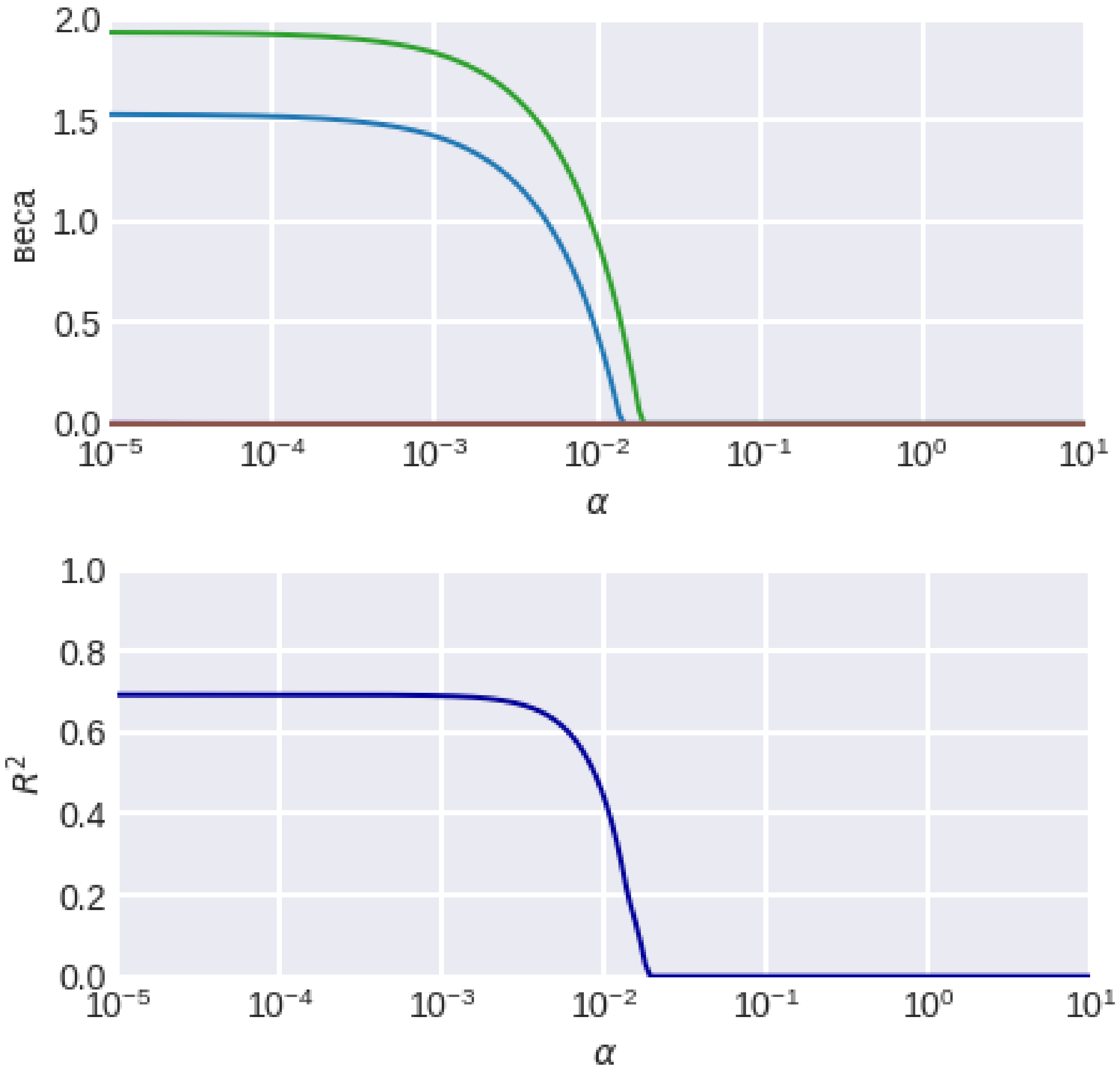


`fit_intercept=True`

Эксперименты с одинаковыми и зависимыми признаками: L_1-регуляризация



`fit_intercept=True`



`fit_intercept=True, normalize=True`

Эксперименты с одинаковыми и зависимыми признаками

Часто важно

- **использовать свободный член**
- **предварительно нормировать данные**

Семейство регуляризированных линейных методов

Ridge

$$\| y - Xw \|_2^2 + \lambda \| w \|_2^2 \to \min_w$$

LASSO (Least Absolute Selection and Shrinkage Operator)

$$\| y - Xw \|_2^2 + \lambda \| w \|_1 \to \min_w$$

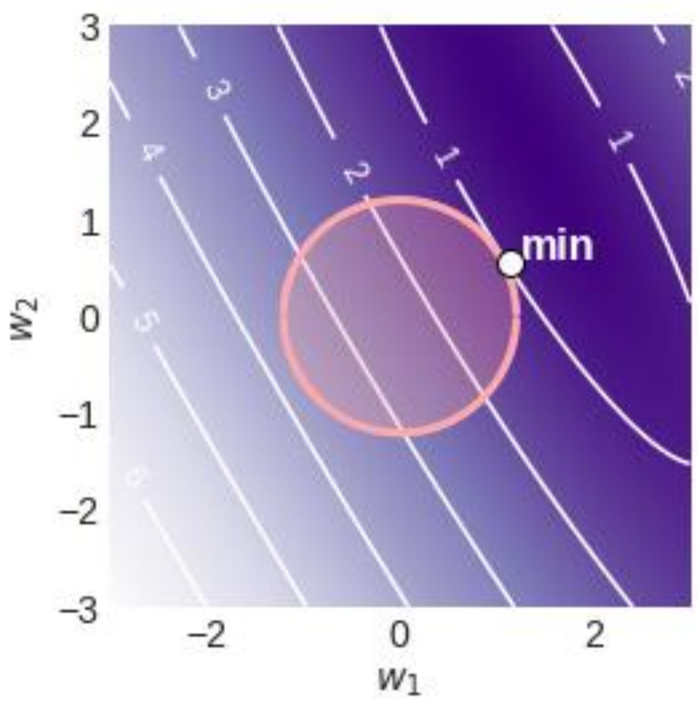
Elastic Net = LASSO + Ridge

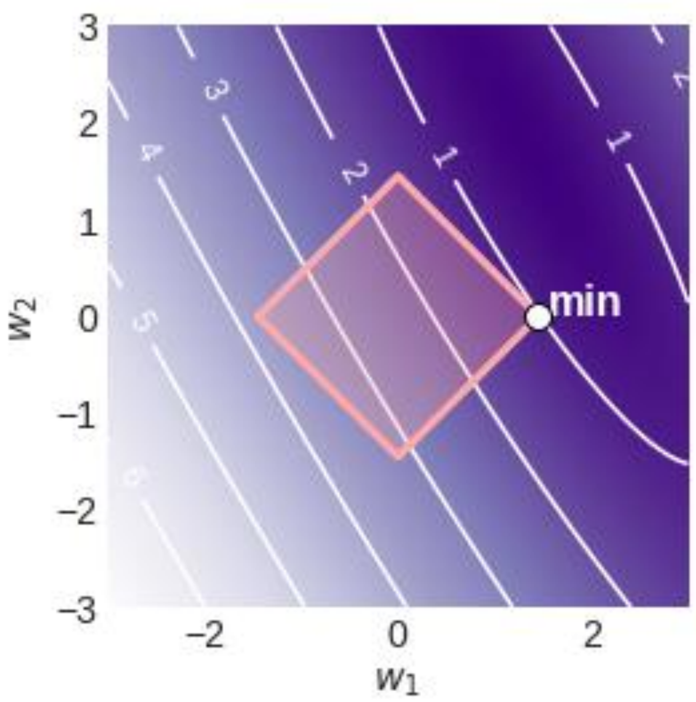
$$\| y - Xw \|_2^2 + \lambda_1 \| w \|_1 + \lambda_2 \| w \|_2^2 \to \min_w$$

Геометрический смысл Ridge, LASSO и Elastic Net

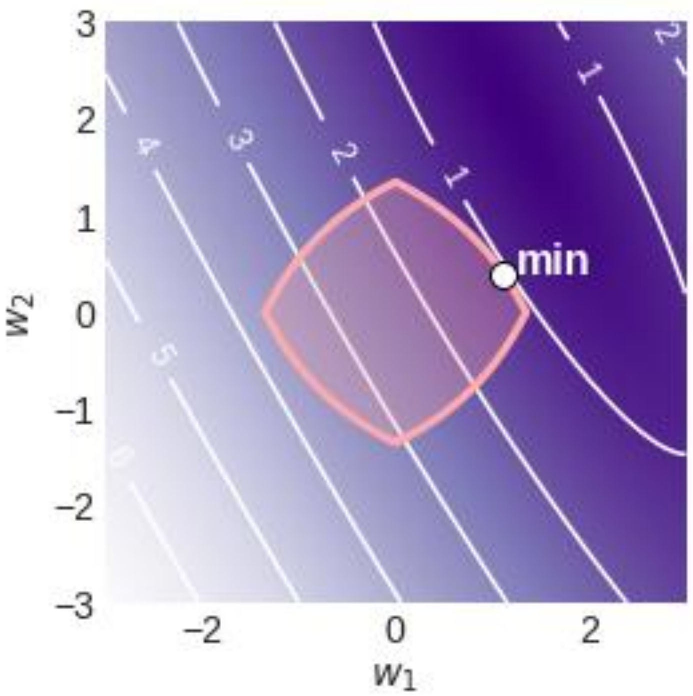
$$\sum_{i=1}^{m}\left(y_i - w_0 - \sum_{j=1}^{n} w_j x_{ij}\right)^2 \to \min_w, \quad \sum_{j=1}^{n} w_j^2 \le s$$

$$\sum_{i=1}^{m}\left(y_i - w_0 - \sum_{j=1}^{n} w_j x_{ij}\right)^2 \to \min_w, \quad \sum_{j=1}^{n} | w_j | \le s$$





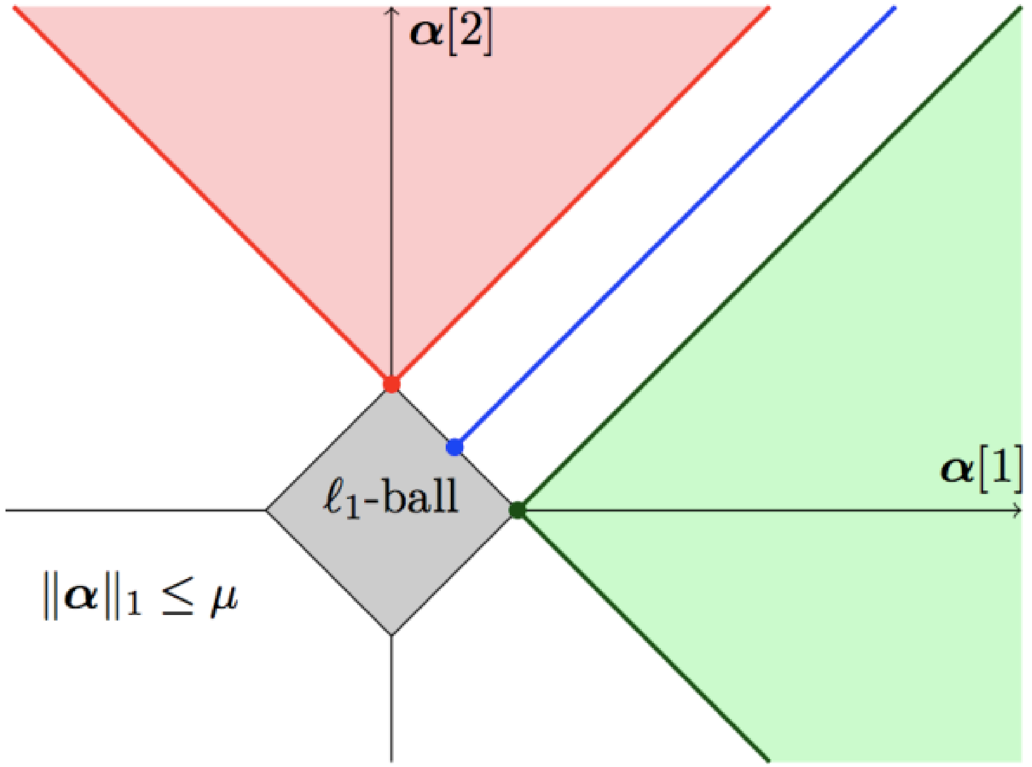
Геометрический смысл Ridge, LASSO и Elastic Net



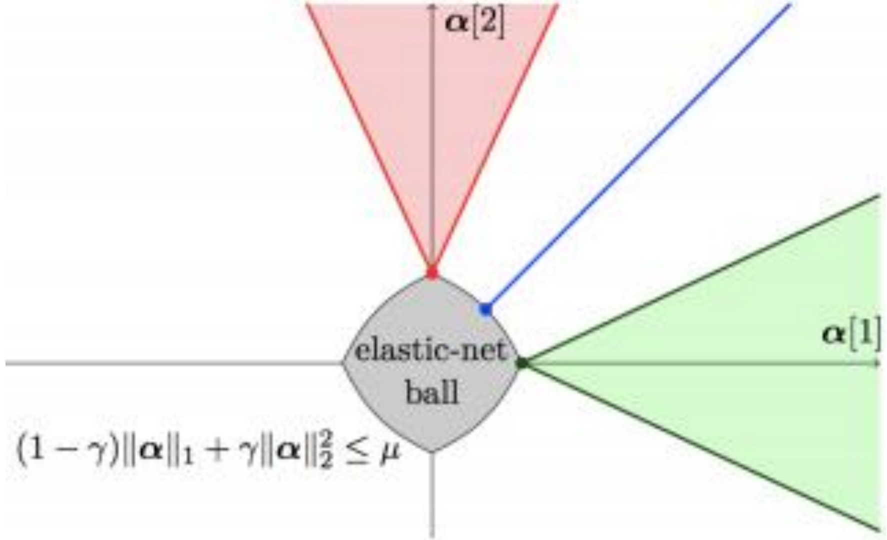
на практике часто модель и не может зависеть от небольшого числа переменных

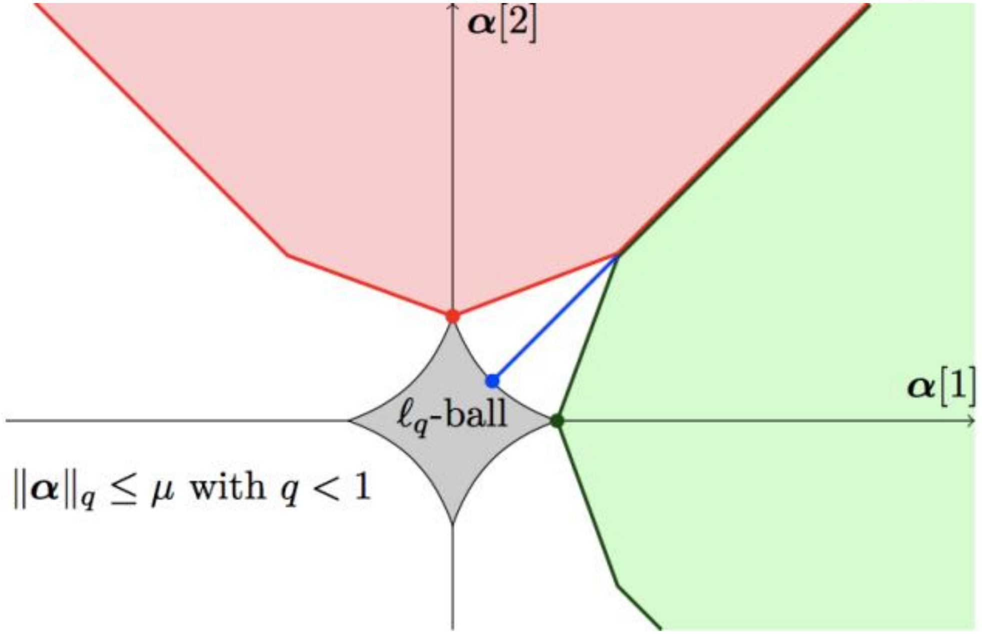
Эффект разреженности

если линии уровня оптимизируемой функции – концентрические окружности…



David S. Rosenberg «Foundations of Machine Learning»
https://bloomberg.github.io/foml/

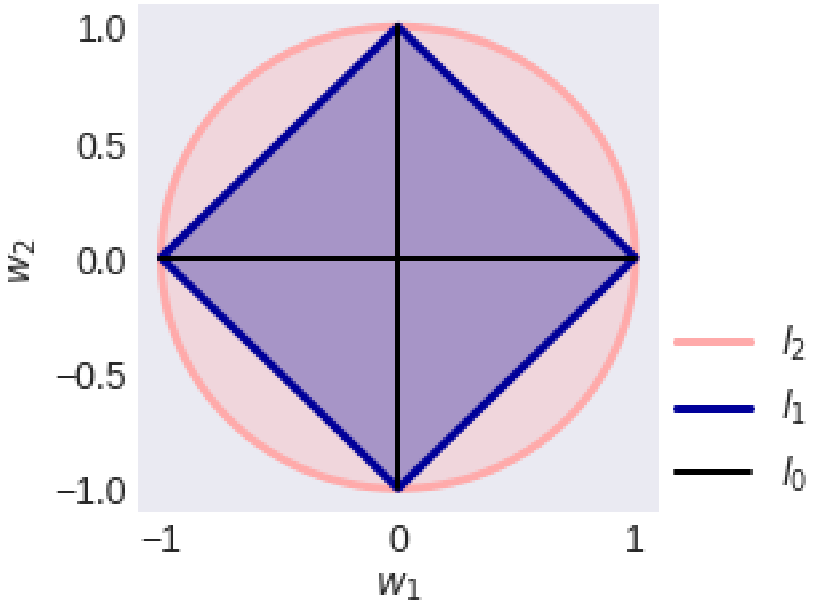




Почему L1-норма ⇒ разреженность

1. См. рис. больше вероятность, что линии уровней функции ошибки касаются области ограничений в точках с нулевыми координатами

2. L1-норма больше похожа на L0, чем L2



$$\| w \|_0 = | \{ t \mid w_t \neq 0 \} |$$

При увеличении коэффициента регуляризации веса стремятся к нулю

Обеспечивается автоматическая селекция признаков!

Регуляризация ⇒ упрощение

Соблюдение принципа Оккама

регуляризация ⇒ зануление коэффициентов ⇒ упрощение модели

В целом, неверно, что чем меньше коэффициентов, тем проще модель, но у нас линейная модель..

потом будет обоснование регуляризации с помощью вероятностных предположений

Проблема вырожденности матрицы

$$w = (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} y$$

Решения:

1. Регуляризация
2. Селекция (отбор) признаков
3. Уменьшение размерности (в том числе, PCA)
4. Увеличение выборки

Селекция признаков в линейной регрессии

~ отдельная тема

Какие признаки включить в модель:

$$a(X_1,\ldots,X_n) = w_0 + w_1 X_1 + \ldots + w_n X_n$$

пока маленький обзор стратегий:

1 стратегия – умный перебор подмножества признаков

2 стратегий – оценка качества признаков (фильтры)

3 стратегия – встроенные методы (ex: LASSO)

Обоснование необходимости селекции

1. Проблема вырожденности в линейной регрессии
2. Проблема «почти дубликатов»
3. Уменьшение модели и интерпретация
4. Уменьшение стоимости данных

Проблема вырожденности матрицы

$$w = (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} y$$

Решения:

1. Регуляризация
2. Селекция (отбор) признаков
3. **Уменьшение размерности (в том числе, PCA)**
4. **Увеличение выборки**

	x1	x2	x3	y
0	0.44	0.62	0.51	-0.25
1	0.03	0.53	0.07	-0.51
2	0.55	0.13	0.43	0.41
3	0.44	0.51	0.10	0.04
4	0.42	0.18	0.13	0.12
5	0.33	0.79	0.60	-0.45

	x1-x2	y
0	-0.18	-0.25
1	-0.50	-0.51
2	0.42	0.41
3	-0.07	0.04
4	0.24	0.12
5	-0.46	-0.45

обоснование необходимости аналогично селекции

Линейная регрессия: градиентный метод обучения

недостатки прямого…

работа с большими матрицами (тем более обращение)

Было:

$$\frac{1}{2}\sum_{i=1}^{m}(a(x_i \mid w) - y_i)^2 \to \min$$

$$w^{(t+1)} = w^{(t)} - \eta \sum_{i=1}^{m}(a(x_i \mid w^{(t)}) - y_i)\frac{\partial a(x_i \mid w^{(t)})}{\partial w}$$

Gradient Descent

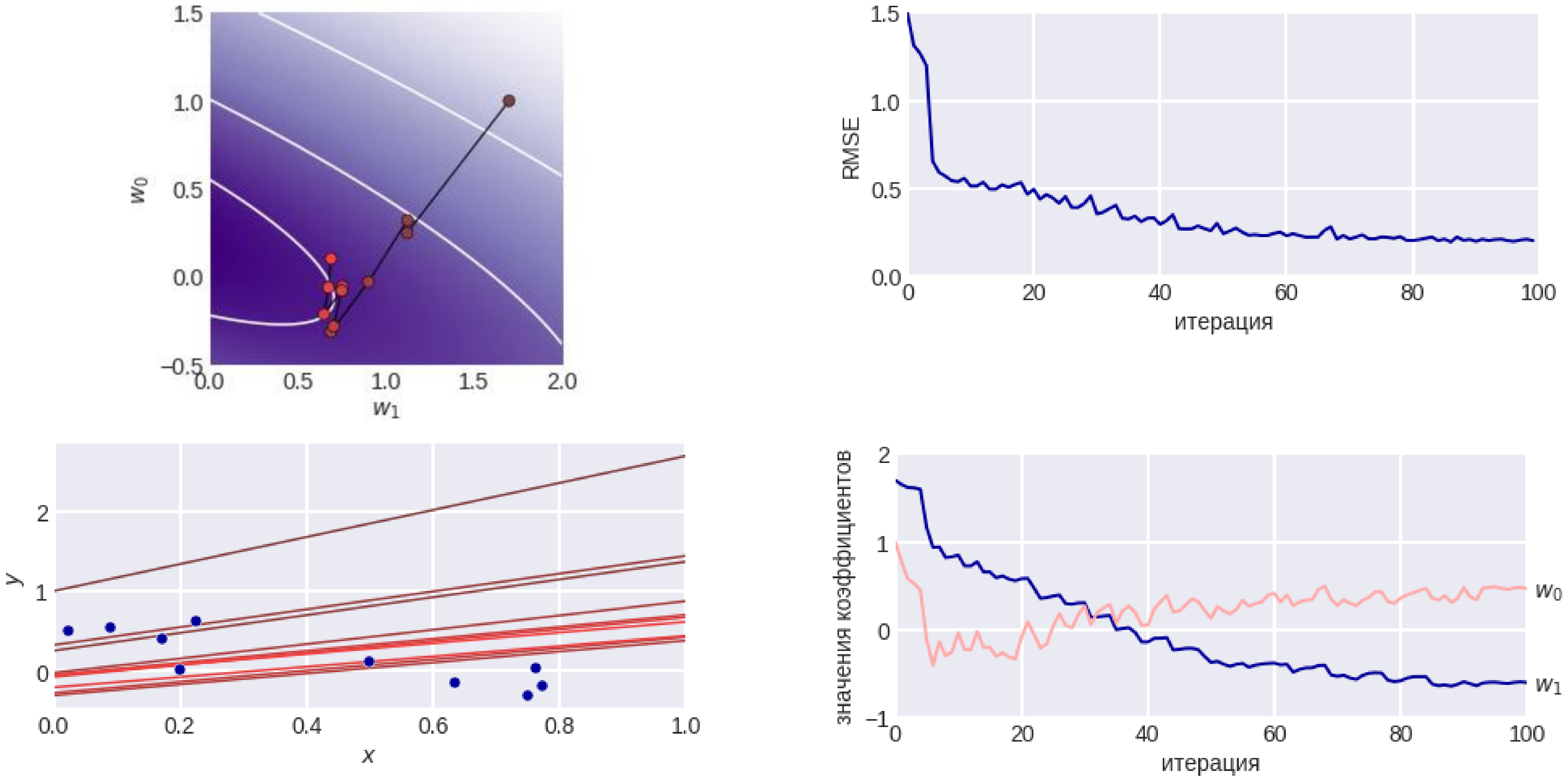
$$a(x \mid w) = w^{\mathrm{T}} x$$

$$w^{(t+1)} = w^{(t)} - \eta \sum_{i=1}^{m}(a(x_i \mid w^{(t)}) - y_i)x_i$$

Stochastic Gradient Descent

$$w^{(t+1)} = w^{(t)} - \eta_t (a(x_i \mid w^{(t)}) - y_i)x_i$$

Линейная регрессия: градиентный метод обучения



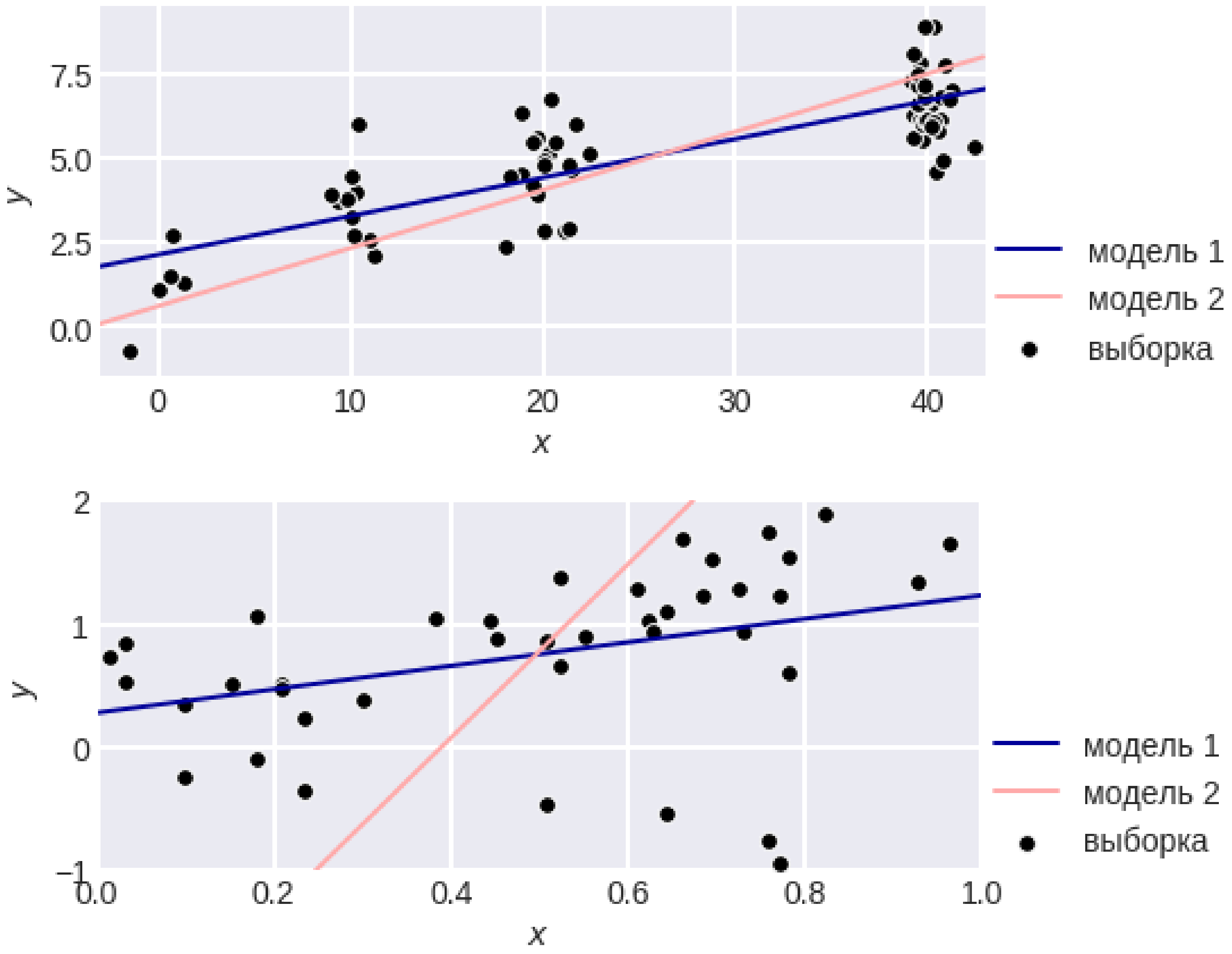
Реализация в scikit-learn

`sklearn.linear_model.Ridge`

alpha=1.0	**Коэффициент регуляризации, больше – сильнее (в отличие от других функций)**
fit_intercept=True	**Использовать ли свободный член**
normalize=False	**Нормализация данных** Игнорируется без свободного члена
solver="auto"	**Метод оптимизации** "auto", "svd", "cholesky", "lsqr", "sparse_cg", "sag", "saga"
copy_X=True, max_iter=None, tol=0.001, random_state=None	

```
sklearn.linear_model.Lasso
(alpha=1.0, fit_intercept=True, normalize=False, precompute=False,
copy_X=True, max_iter=1000, tol=0.0001, warm_start=False, positive=False,
random_state=None, selection="cyclic")
sklearn.linear_model.ElasticNet
(alpha=1.0, l1_ratio=0.5, fit_intercept=True, normalize=False,
precompute=False, max_iter=1000, copy_X=True, tol=0.0001, warm_start=False,
positive=False, random_state=None, selection="cyclic")
```

Две регрессии



Чем отличаются модели 1 и 2?

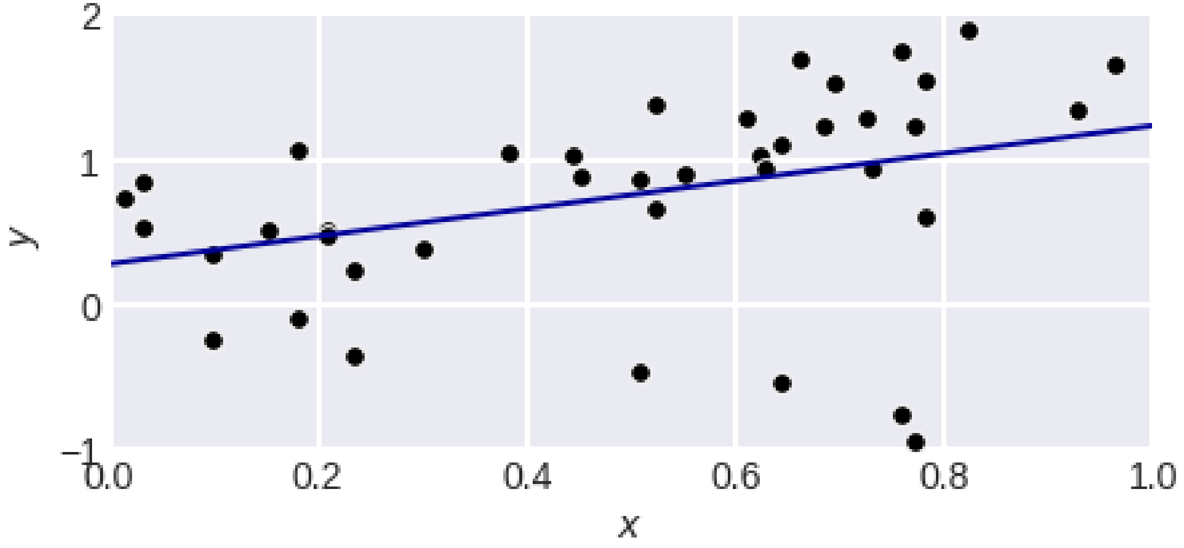
Две регрессии

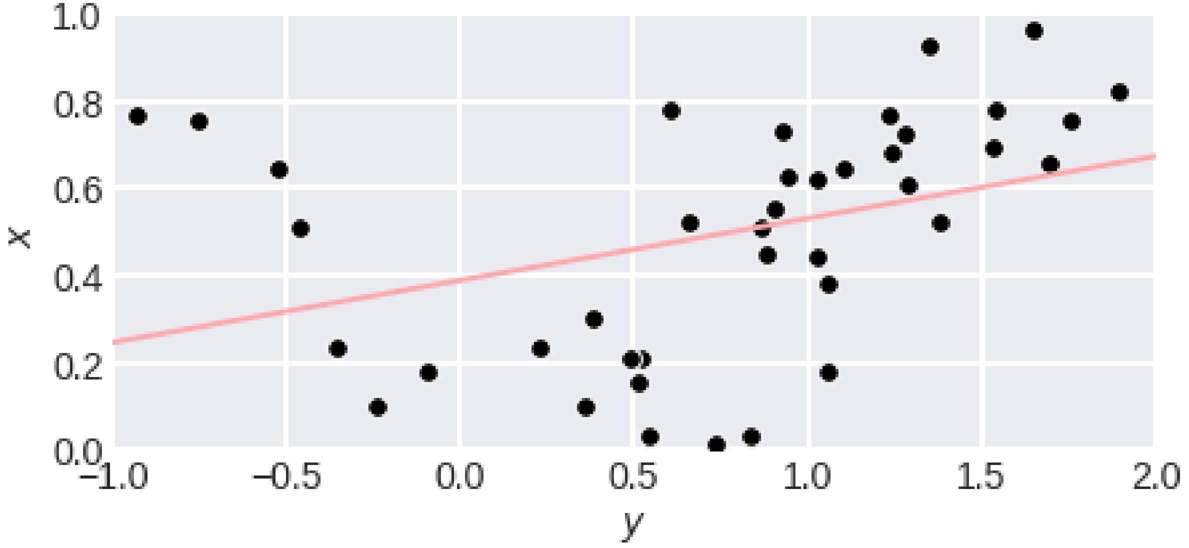
разные задачи $y(x)$ **и** $x(y)$

хотя зависимость линейная

$$Y = w_0 + w_1 X_1$$

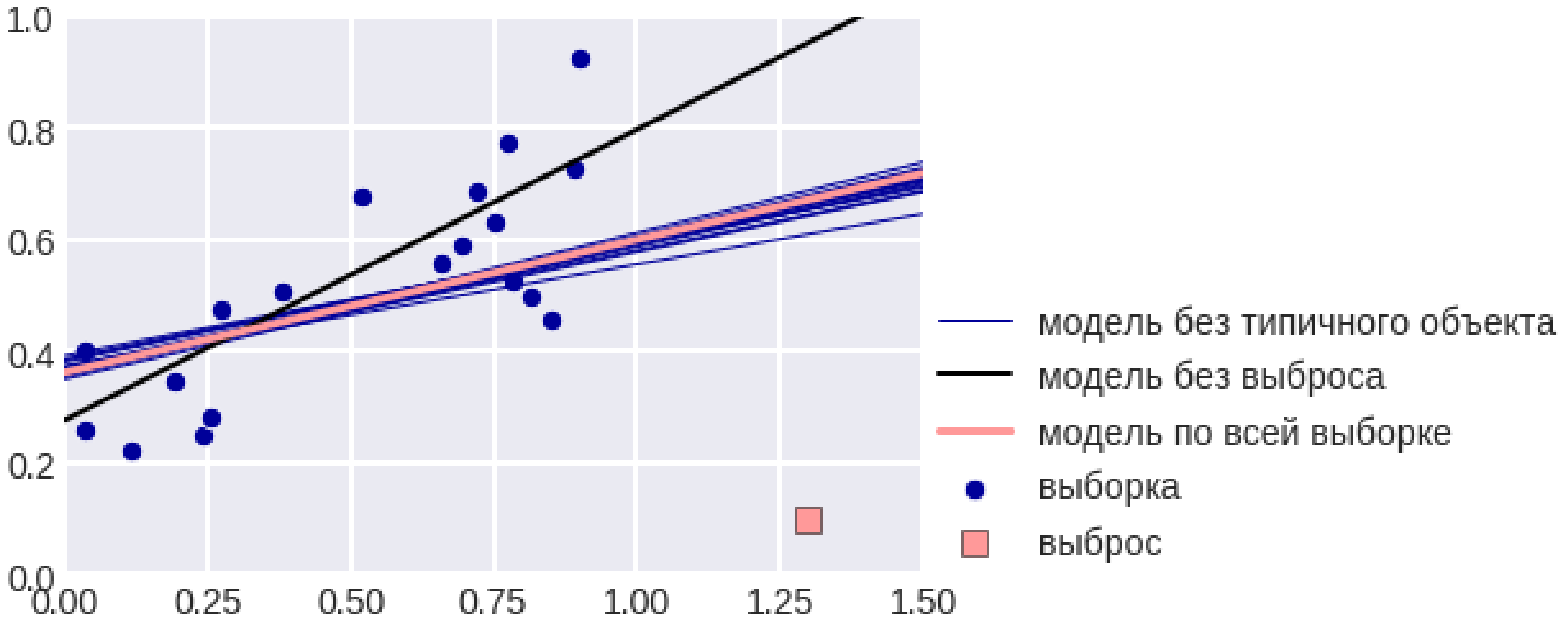
$$\left\| \begin{bmatrix} x_1 & 1 \\ \cdots & \cdots \\ x_m & 1 \end{bmatrix} \begin{pmatrix} w_1 \\ w_0 \end{pmatrix} - \begin{pmatrix} y_1 \\ \cdots \\ y_m \end{pmatrix} \right\|_2^2 \to \min \qquad \left\| \begin{bmatrix} y_1 & 1 \\ \cdots & \cdots \\ y_m & 1 \end{bmatrix} \begin{pmatrix} w_1 \\ w_0 \end{pmatrix} - \begin{pmatrix} x_1 \\ \cdots \\ x_m \end{pmatrix} \right\|_2^2 \to \min$$





есть и «промежуточная стратегия» – дальше PCA

Линейная регрессия – неустойчивость к выбросам



Ошибка с весами

Если у каждого объекта есть цена ошибки…

$$\sum_{i=1}^{m} \underset{>0}{v_i} \left(y_i - w^{\mathrm{T}} x_i \right)^2 + \ldots = \sum_{i=1}^{m} \left(\sqrt{v_i} y_i - w^{\mathrm{T}} (\sqrt{v_i} x_i) \right)^2 + \ldots \to \min$$

небольшая переформулировка задачи:

$$\{(x_1, y_1), \ldots, (x_m, y_m)\} \to \{(\sqrt{v_1} x_1, \sqrt{v_1} y_1), \ldots, (\sqrt{v_m} x_m, \sqrt{v_m} y_m)\}$$

$$(y - Xw)^{\mathrm{T}} V (y - Xw) \sim \| V^{1/2} y - V^{1/2} Xw \|_2^2 \to \min_w$$

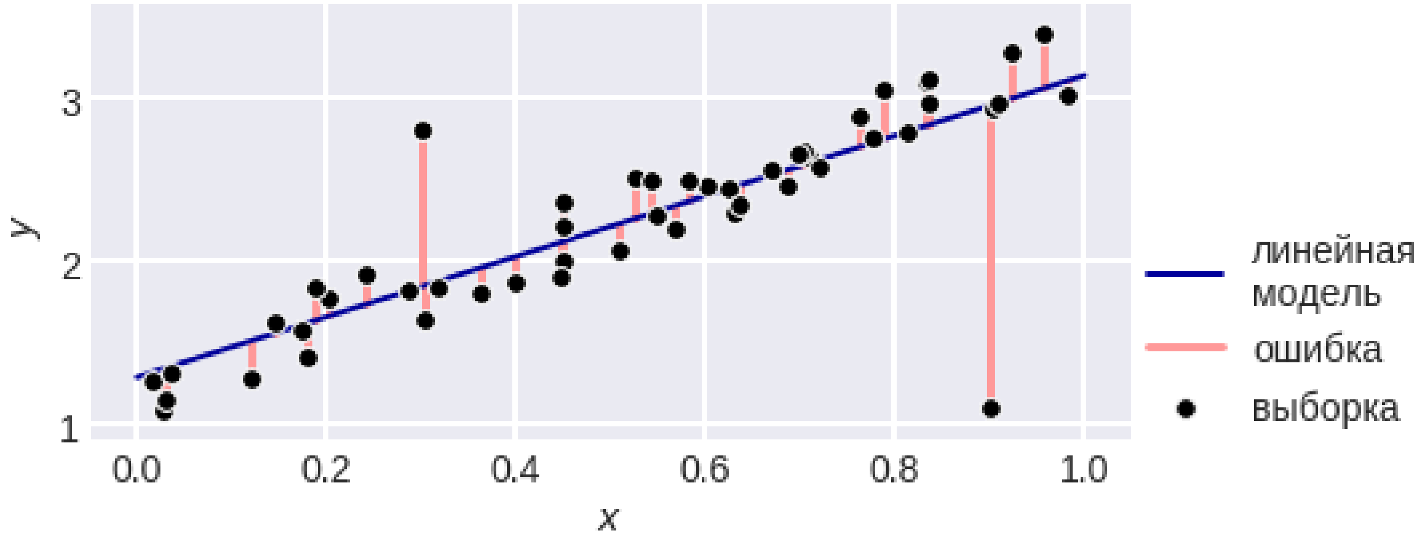
$$w = (X^{\mathrm{T}} V X)^{-1} X^{\mathrm{T}} V y$$

1) перейти к новым данным («испорченными весам»)

2) если веса целые числа – можно продублировать объекты

3) если веса из отрезка [0, 1] – при численном градиентном решении можно выбирать следующий объект с соответствующей вероятностью

Устойчивая регрессия (Robust Regression)



0. Инициализация весов объектов

$$v=(v_1,\ldots,v_m)=(1/m,\ldots,1/m)$$

1. Цикл

1.1. Настроить алгоритм, учитывая веса объектов

$$a=\mathrm{fit}(\{x_i,y_i,v_i\})$$

можно использовать любую регрессионную модель

1.2. Вычислить ошибки на обучении

$$\varepsilon_i=a(x_i)-y_i$$

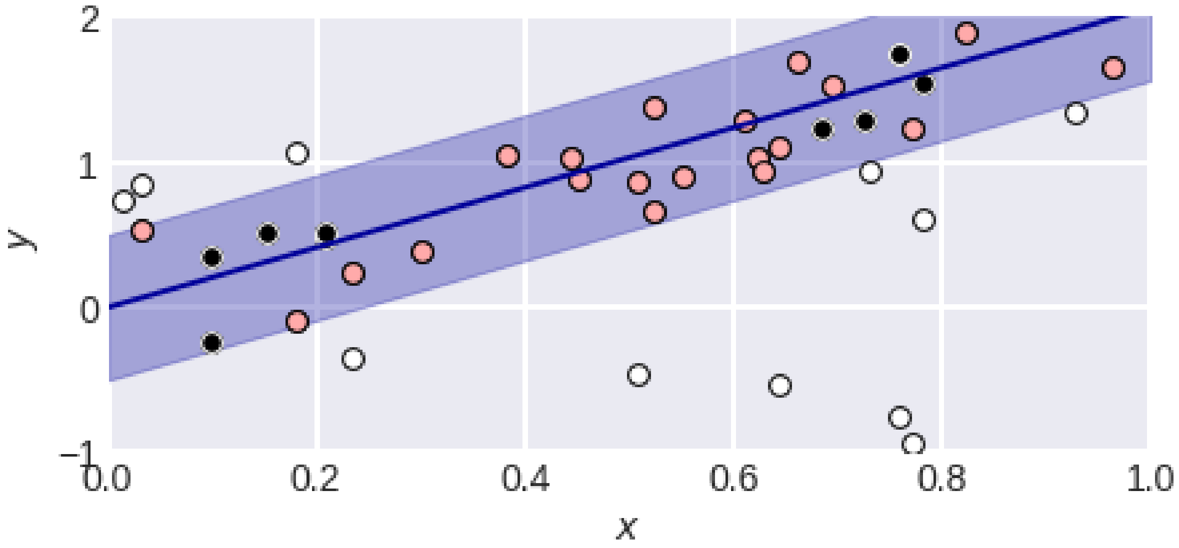
1.3. Пересчитать веса объектов $v_i=\exp(-\varepsilon_i^2)$

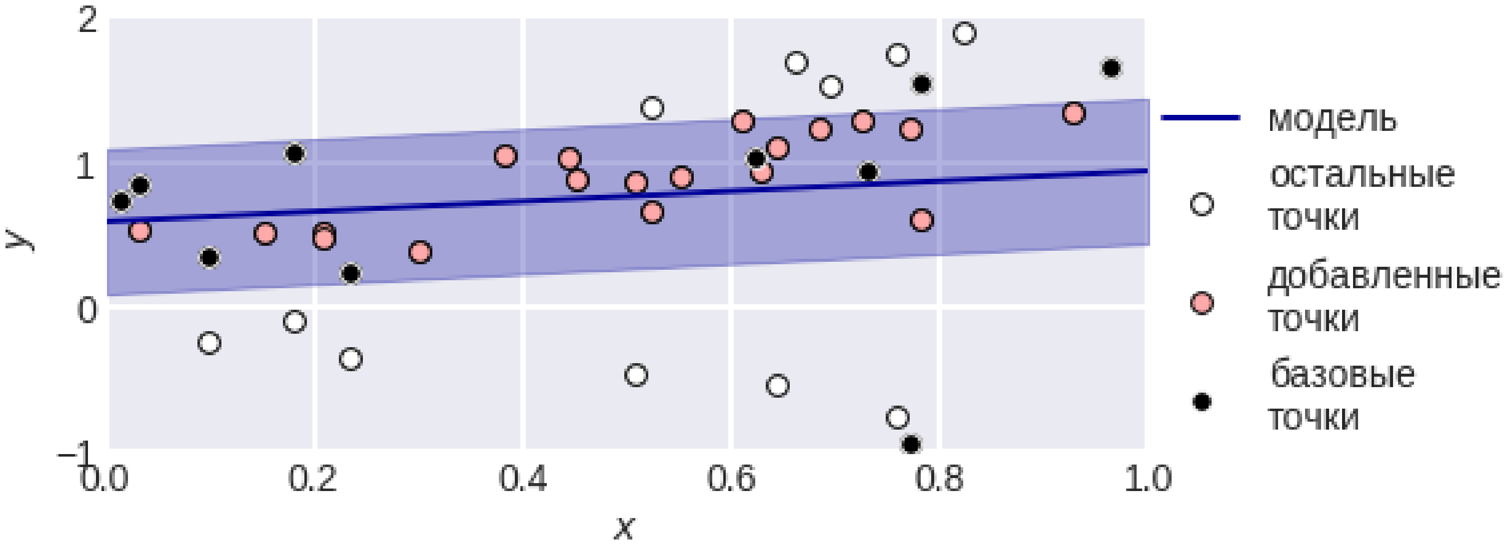
нормировать на сумму

можно использовать другую невозрастающую функцию; можно (иногда нужно) нормировать

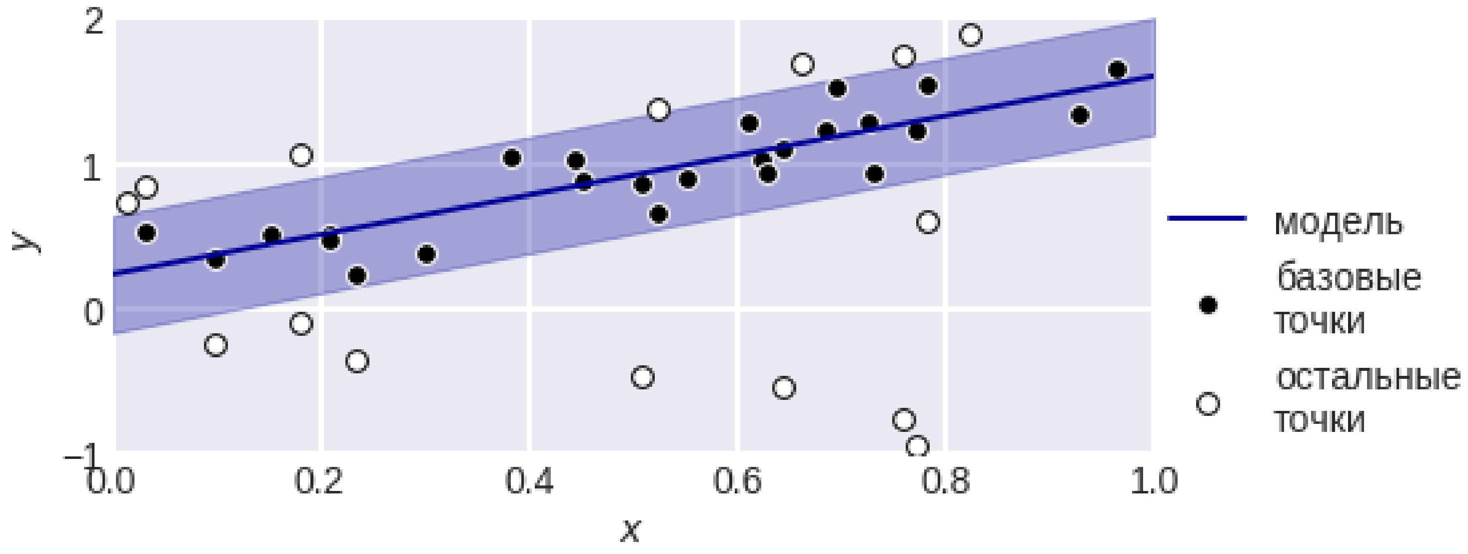
RANdom SAmple Consensus (RANSAC)

- несколько раз
  - выбрать случайное подмножество точек – базовое (inliers)
  - обучить модель на базовом подмножестве
  - найти все точки, которые хорошо предсказываются моделью
    например, ошибка не больше ε
  - пополнить ими базовое множество
  - (если добавили много) переобучить модель на новом множестве
- выбрать модель с наименьшей ошибкой





RANdom SAmple Consensus (RANSAC) в scikit-learn




```
from sklearn.linear_model import RANSACRegressor
# Robustly fit linear model with RANSAC algorithm
ransac = RANSACRegressor()
ransac.fit(x[:, np.newaxis], y)
inlier_mask = ransac.inlier_mask_
outlier_mask = np.logical_not(inlier_mask)
```

RANdom SAmple Consensus (RANSAC) в scikit-learn

```
sklearn.linear_model import RANSACRegressor
```

`base_estimator=None`	**Базовый алгоритм (по умолчанию – линейная регрессия)**
`min_samples=None`	**Число / доля базовых объектов (*n*+1)**
`residual_threshold=None`	**Порог для пополнения базового множества (MAD(y))**
`max_trials=100`	**Число итераций**
`stop_n_inliers`	**Остановить вычисления, если найдено столько базовых точек**
`loss="absolute_loss"`	**Как оценивать ошибку**

```
is_data_valid=None, is_model_valid=None, max_skips=inf,
stop_score=inf, stop_probability=0.99, lossrandom_state=None
```

Ошибка и её оценка (невязка)

решаем задачу линейной регрессии

постулируем, что есть линейная зависимость с точностью до шума

$$y = Xw^* + \varepsilon$$

потом рассмотрим такую вероятностную постановку

сами решаем так:

$$y \approx a = Xw$$

с точностью до оценки шума

$$\hat{\varepsilon} = y - a$$

нашли оптимальные коэффициенты

$$w = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}y$$

тогда оценка ошибки (невязка)

$$\hat{\varepsilon} = y - a = y - Xw = y - \underbrace{X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}}_{H}y = (I - H)y =$$

$$= (I - H)(Xw^* + \varepsilon) = \underbrace{Xw^* - HXw^*}_{Xw^* - Xw^* = 0} + (I - H)\varepsilon = (I - H)\varepsilon$$

Ошибка и её оценка (невязка)

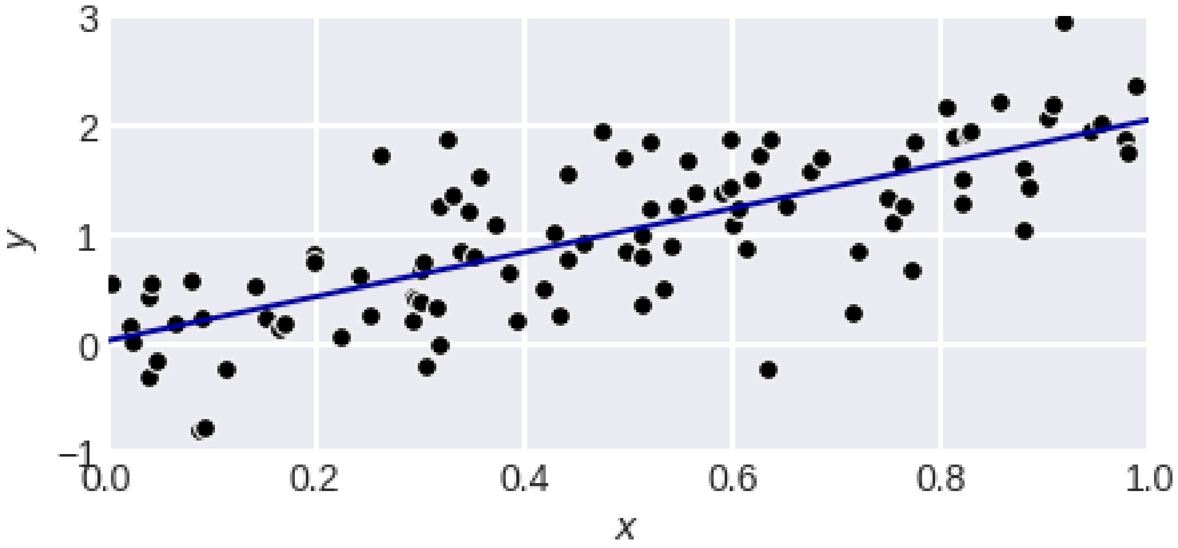
получили, что

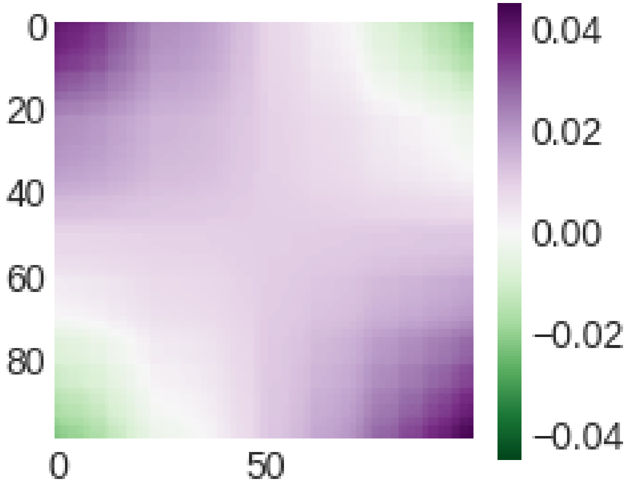
$$\hat{\varepsilon} = (I - H)\varepsilon,$$

где $H = X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$

– projection (hat) matrix

Кстати, $a = Hy$





матрица H

таким образом,

- **невязки коррелируют (даже если ошибки нет)**
- **в разном масштабе**

иногда стандартизуют (во многих пакетах, но за этим следить!)

$$\hat{\varepsilon}_{[t]} / \sqrt{1 - h_{tt}}$$

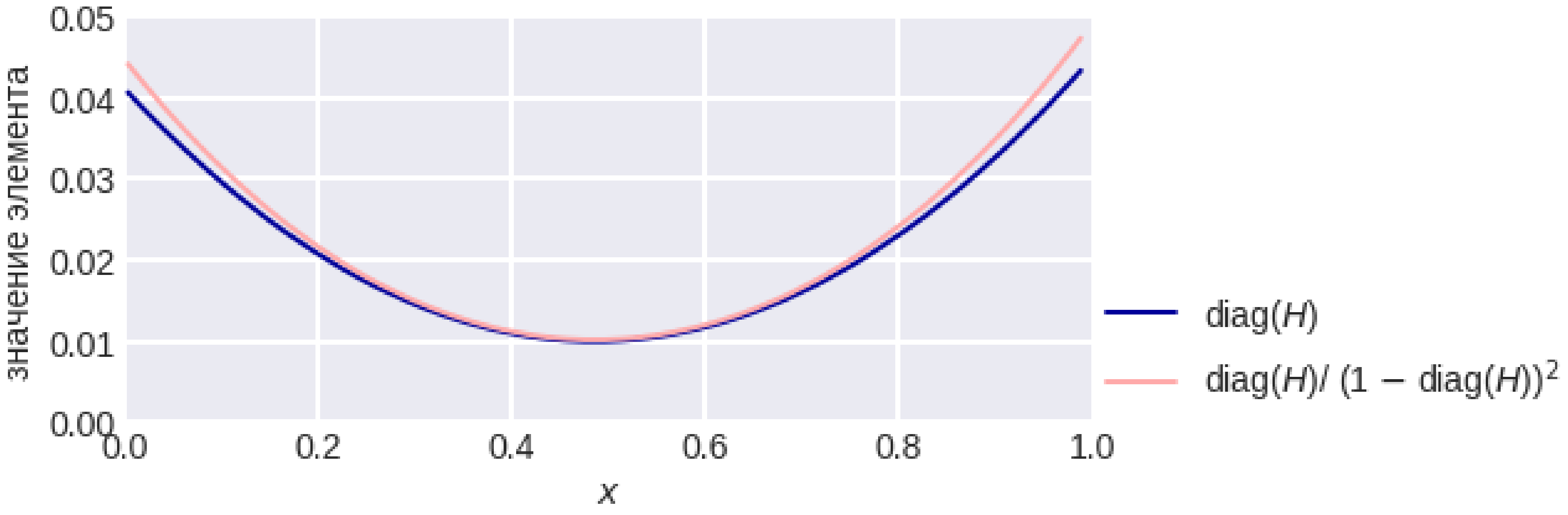
справедливости ради – м.б. не слишком заметный эффект

- **невязки коррелируют** (даже если ошибки не коррелируют)
- **но нет корреляции с целевым значением!** (если ошибка не коррелирует)

Ошибка и её оценка (невязка)

Для справки:

$$h_{tt} = \frac{1}{m} + \frac{(x_t - \overline{x})^2}{\sum_{i=1}^{m}(x_i - \overline{x})^2}$$



Если ввести Cook's distance – как сильно точка влияет на решение (зачем?)

$$D_j = \mathrm{const} \cdot \sum_{i=1}^{m} (a(x_i \mid X_{\mathrm{train}}) - a(x_i \mid X_{\mathrm{train}} \setminus \{x_j\}))^2$$

то можно вывести:

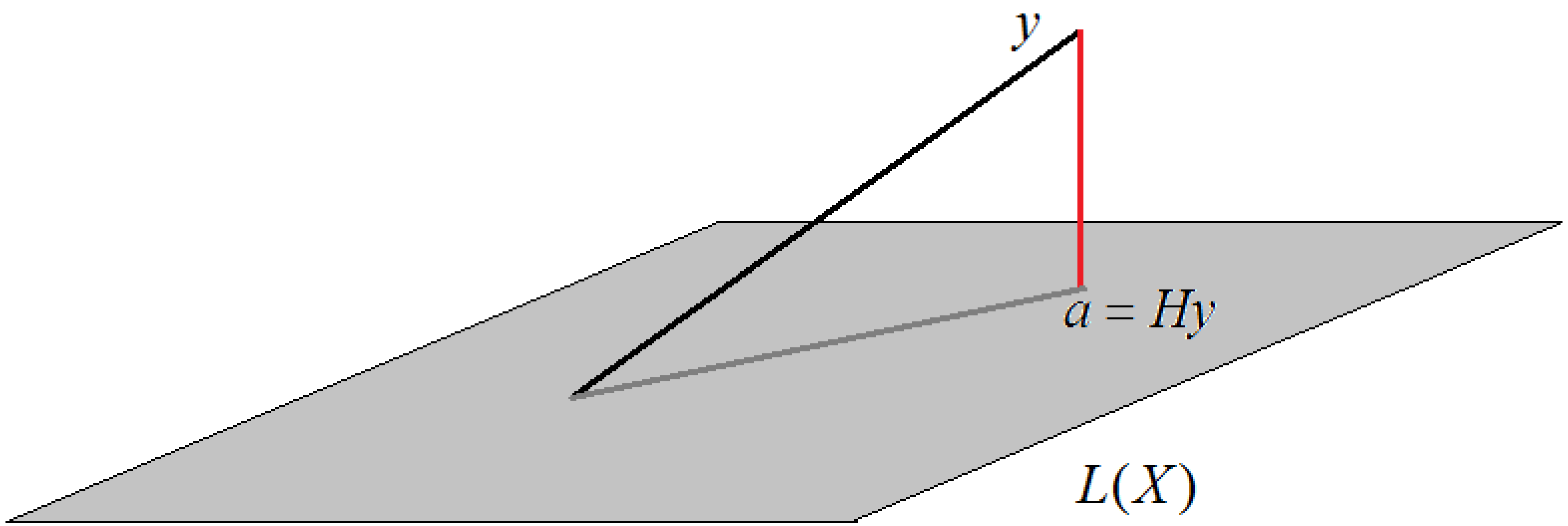
$$D_j = \mathrm{const} \cdot \frac{h_{jj}}{(1 - h_{jj})^2} \hat{\varepsilon}_{[j]}^2$$

Проекционная (hat) матрица

$$H = X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$$

$$L(X)$$

$a = Hy \in L(X)$ **– из линейной комбинация столбцов матрицы** X



Линейная регрессия: связь с SVD

Оптимальные веса $w = (X^\mathrm{T} X)^{-1} X^\mathrm{T} y$**, воспользуемся SVD:** $X = U\Lambda V^\mathrm{T}$**, тогда**

$$w = (V\Lambda^\mathrm{T} U^\mathrm{T} U \Lambda V^\mathrm{T})^{-1} V\Lambda^\mathrm{T} U^\mathrm{T} y = (V\Lambda^2 V^\mathrm{T})^{-1} V\Lambda^\mathrm{T} U^\mathrm{T} y =$$

$$= (V\Lambda^{-2} V^\mathrm{T}) V\Lambda^\mathrm{T} U^\mathrm{T} y = V\Lambda^{-1} U^\mathrm{T} y = \sum_{j=1}^{k} \frac{u_j^\mathrm{T} y}{\lambda_j} v_j$$

веса – линейные комбинации столбцов V

коэффициенты – скалярное произведение столбцов U **и целевого столбца**

ещё одна иллюстрация проблемы, когда $\lambda_j \approx 0$**, веса м.б. большими**

Теперь решение с регуляризацией: $w = (X^\mathrm{T} X + \lambda I)^{-1} X^\mathrm{T} y$

$$w = (V\Lambda^\mathrm{T} U^\mathrm{T} U \Lambda V^\mathrm{T} + \lambda I)^{-1} V\Lambda^\mathrm{T} U^\mathrm{T} y = (V(\Lambda^2 + \lambda I) V^\mathrm{T})^{-1} V\Lambda^\mathrm{T} U^\mathrm{T} y =$$

$$= V\Lambda(\Lambda^2 + \lambda I)^{-1} U^\mathrm{T} y = \sum_{j=1}^{k} \frac{\lambda_j \cdot u_j^\mathrm{T} y}{\lambda_j^2 + \lambda} v_j$$

видиен эффект от регуляризации! при больших зануляются λ **коэффициенты**

Линейные скоринговые модели в задаче бинарной классификации

Пусть $X = \mathbb{R}^n, Y = \{0, 1\}$

Как решать задачи классификации с помощью линейной модели: будем получать вероятность принадлежности к классу 1

$$a(x) \in [0, 1]$$

Любая линейная функция на $\mathbb{R}^n$ **будет получать значения в** $\mathbb{R}$**, поэтому нужна деформация (transfer function):**

$$\sigma : \mathbb{R} \to [0, 1]$$

Функции деформации

В логистической регрессии

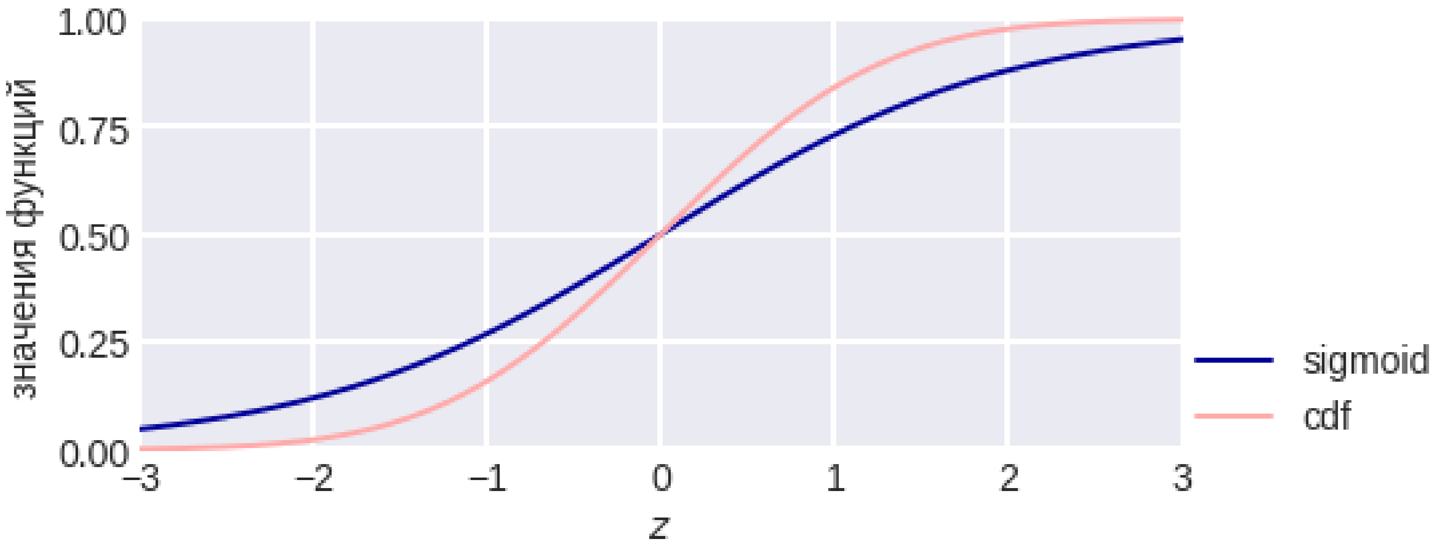
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Логистическая функция (сигмоида)

В Probit-регрессии

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp(-t^2 / 2) \partial t$$

Normal Cumulative distribution function



Логистическая регрессия

$$p(x) \equiv P(Y=1 \,|\, x) = \sigma(z) = \frac{1}{1+e^{-z}} \in (0,1),$$

$$z = w_0 + w_1 X_1 + \ldots + w_n X_n,$$

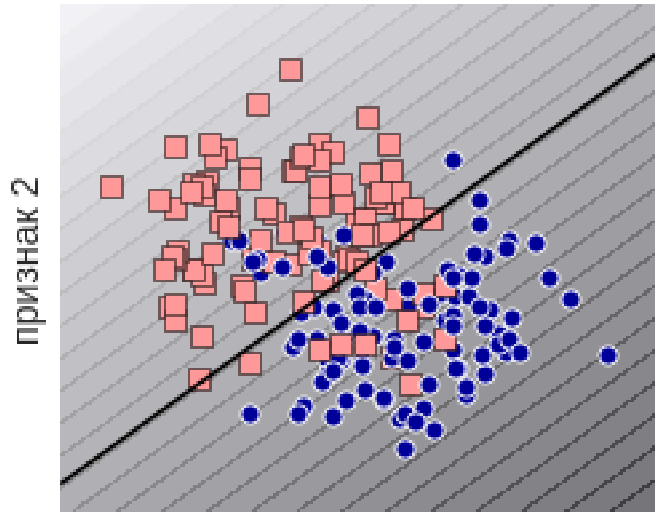
$$\log\left(\frac{p(x)}{1-p(x)}\right) = z$$

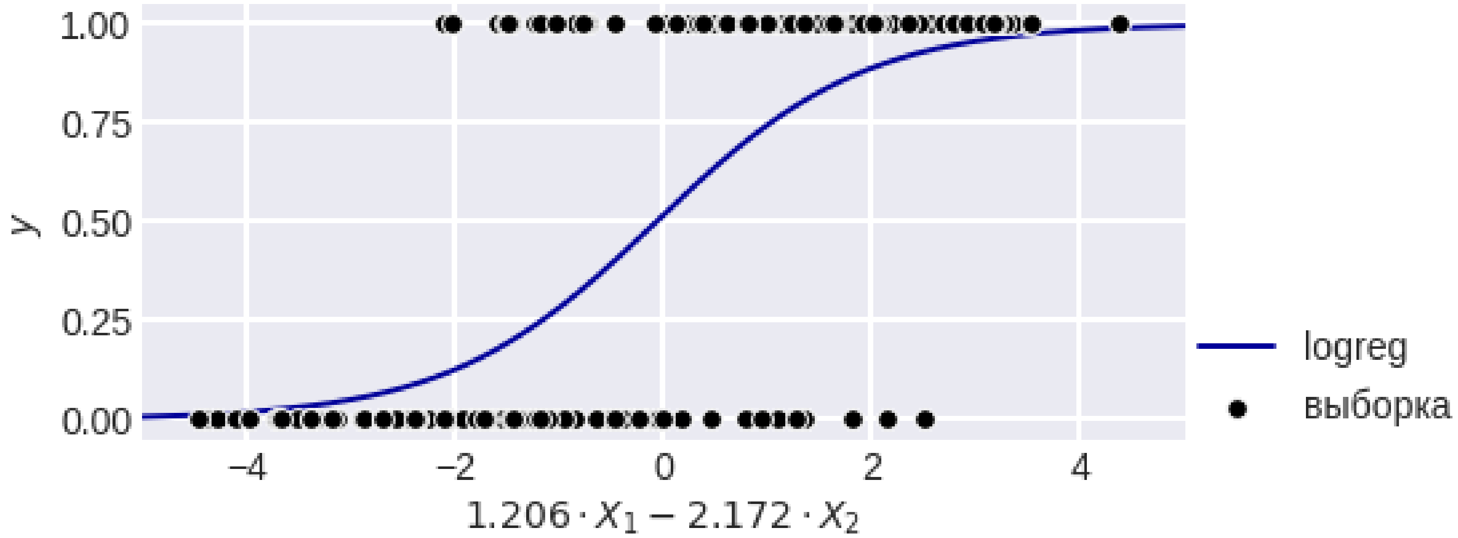
– монотонное преобразование, которое называют logit-transformation

Решаем задачу классификации, но метод называется логистическая регрессия

Геометрический смысл логистической регрессии

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X, y)
a = model.predict_proba(X_test)[:,1]
```

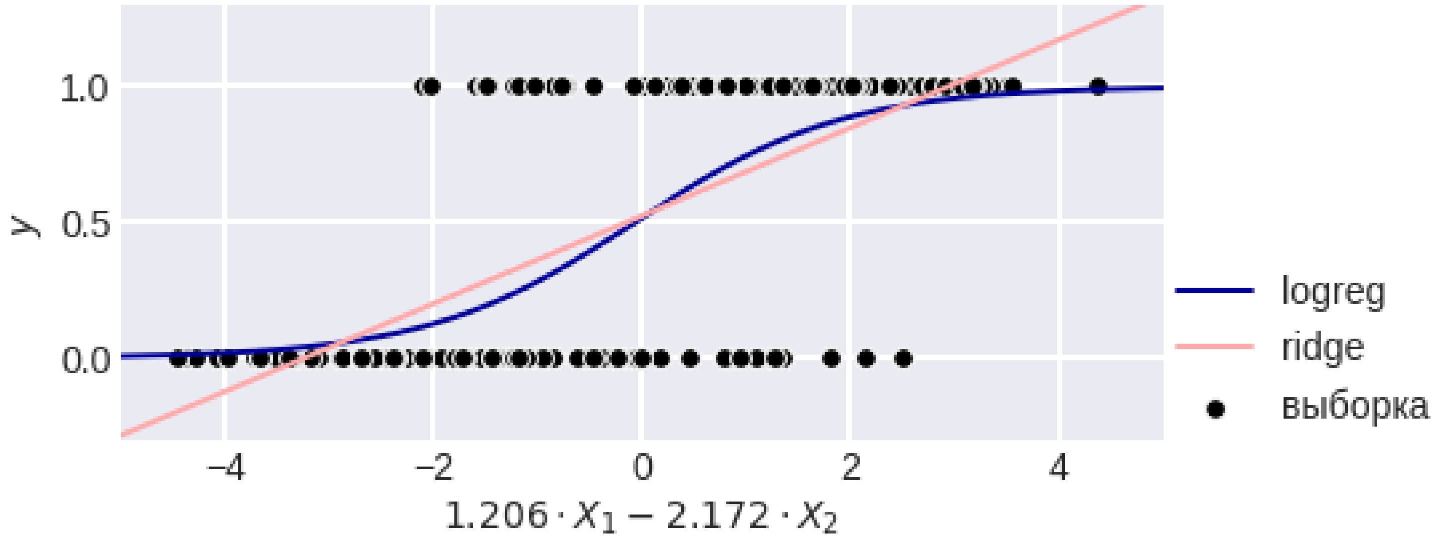




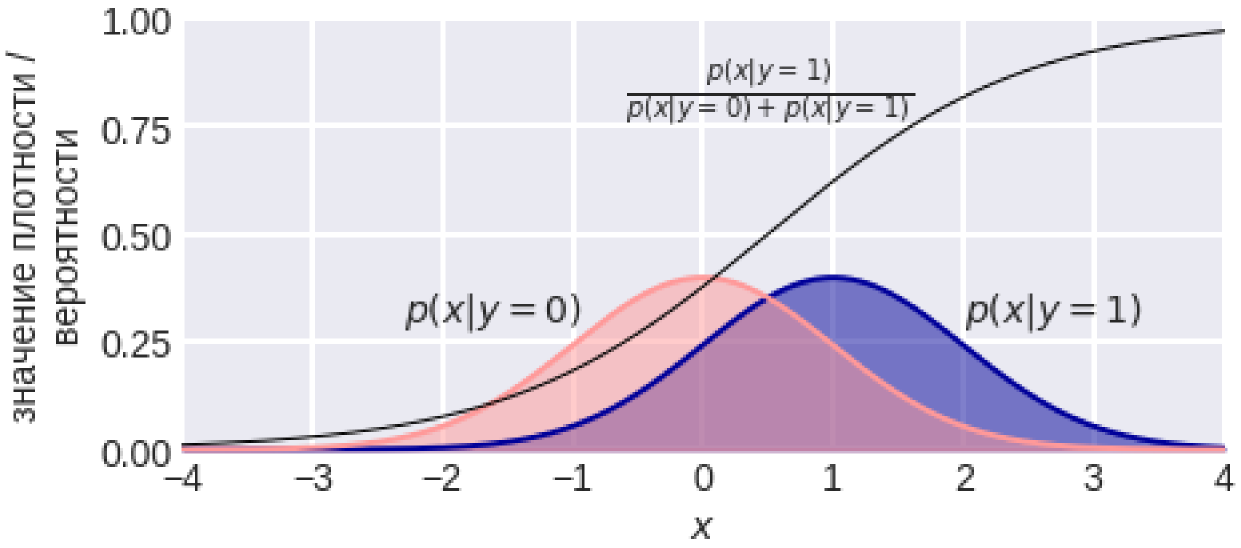
$z = w_0 + w_1 X_1 + \ldots + w_n X_n$ **– проекция на прямую (один признак)**

В однопризнаковом случае надо решить задачу классификации

Чем логистическая регрессия лучше регрессии



Откуда берётся сигмоида



$$p(x\,|\,y=t)=\frac{1}{(2\pi)^{n/2}\,|\,\Sigma\,|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu_t)^{\mathrm{T}}\Sigma^{-1}(x-\mu_t)\right)$$

нормальное распределение с одинаковыми матрицами ковариации

$$p(y=t\,|\,x)=\frac{p(x\,|\,y=t)\,p(y=t)}{p(x\,|\,y=0)\,p(y=0)+p(x\,|\,y=1)\,p(y=1)}$$

Откуда берётся сигмоида

$$p(y=t\,|\,x)=\frac{\dfrac{1}{(2\pi)^{n/2}\,|\,\Sigma\,|^{1/2}}\exp\left(-\dfrac{1}{2}(x-\mu_t)^{\mathrm{T}}\Sigma^{-1}(x-\mu_t)\right)}{\sum\limits_{t}\dfrac{1}{(2\pi)^{n/2}\,|\,\Sigma\,|^{1/2}}\exp\left(-\dfrac{1}{2}(x-\mu_t)^{\mathrm{T}}\Sigma^{-1}(x-\mu_t)\right)}=$$

$$=\frac{1}{1+\exp\left(+\dfrac{1}{2}(x-\mu_t)^{\mathrm{T}}\Sigma^{-1}(x-\mu_t)-\dfrac{1}{2}(x-\mu_{1-t})^{\mathrm{T}}\Sigma^{-1}(x-\mu_{1-t})\right)}=$$

$$=\frac{1}{1+\exp\left(-\dfrac{1}{2}\mu_t{}^{\mathrm{T}}\Sigma^{-1}x-\dfrac{1}{2}x^{\mathrm{T}}\Sigma^{-1}\mu_t+\dfrac{1}{2}\mu_t{}^{\mathrm{T}}\Sigma^{-1}\mu_t+\dfrac{1}{2}\mu_{1-t}{}^{\mathrm{T}}\Sigma^{-1}x+\dfrac{1}{2}x^{\mathrm{T}}\Sigma^{-1}\mu_{1-t}-\dfrac{1}{2}\mu_{1-t}{}^{\mathrm{T}}\Sigma^{-1}\mu_{1-t}\right)}=$$

$$=\sigma(w^{\mathrm{T}}x+w_0)$$

Обучение логистической регрессии

Метод максимального правдоподобия

$$L(w_0,\ldots,w_n) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1-p(x_i)) \to \max$$

$$\log L = -\sum_{i:y_i=1} \log(1+e^{-z_i}) - \sum_{i:y_i=0} \log(1+e^{+z_i}) \equiv -\sum_{i} \log(1+e^{-y'_i z_i})$$

$$\nabla_w \log L = \sum_{i:y_i=1} \frac{1}{1+e^{-w^{\mathrm{T}}x_i}} e^{-w^{\mathrm{T}}x_i} x_i - \sum_{i:y_i=0} \frac{1}{1+e^{+w^{\mathrm{T}}x_i}} e^{+w^{\mathrm{T}}x_i} x_i =$$

$$= \sum_{i} \frac{y'_i x_i}{1+e^{+y'_i w^{\mathrm{T}} x_i}} = \sum_{i} y'_i x_i \sigma(-y'_i w^{\mathrm{T}} x_i)$$

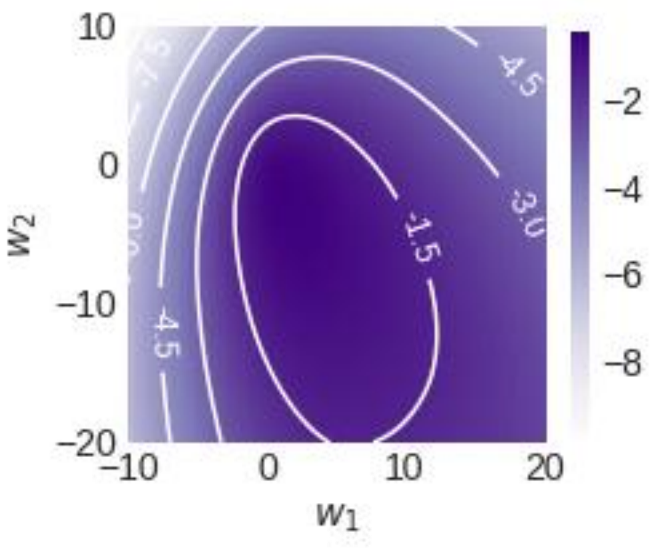
где (для удобства записи)

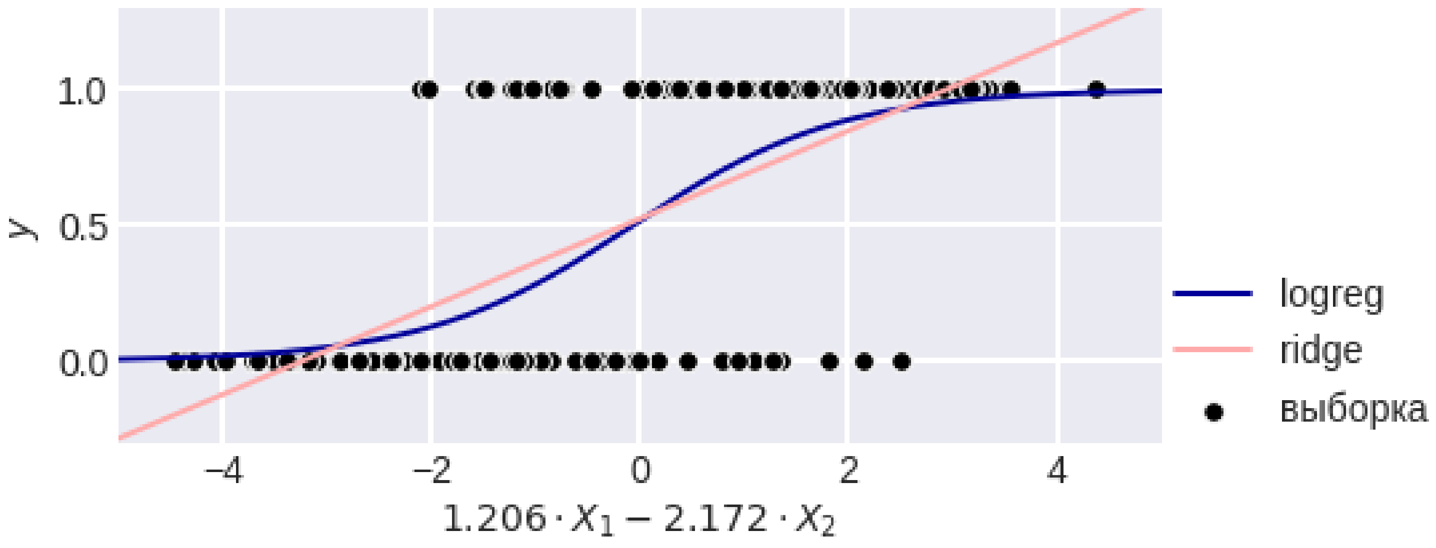
$$y'_i = 2y_i - 1$$

Качество логистической регрессии

– логарифм правдоподобия

(потом будет соответствующая функция ошибки logloss)





метод SGD

$$w \leftarrow w + \eta\sigma(-y_i' w^{\mathrm{T}} x_i) y_i' x_i$$

Запомним!

Многоклассовая логистическая регрессия
Multiclass logistic regression (multinomial regression)

в glmnet такой «симметричный вариант»

$$P(Y=k\,|\,x)=\frac{e^{w_{0k}+w_{1k}X_1+\ldots+w_{nk}X_n}}{\sum_{j=1}^{l}e^{w_{0j}+w_{1j}X_1+\ldots+w_{nj}X_n}}$$

Если

$$\mathrm{softmax}(a_1,\ldots,a_l)=\frac{1}{Z}[e^{a_1},\ldots,e^{a_l}],$$

где $Z=e^{a_1}+\ldots+e^{a_l}$

тогда

$$P(Y=k\,|\,x)=\mathrm{softmax}(w(1)^{\mathrm{т}}x,\ldots,w(l)^{\mathrm{т}}x)$$

Реализация в scikit-learn

`sklearn.linear_model.LogisticRegression`

`penalty="l2"`	**Тип регуляризации** **Не все солверы поддерживают все типы**
`dual=False`	**Переход к двойственной задачи**
`C=1.0`	**Обратная величина к коэффициенту регуляризации**
`fit_intercept=True`	**Свободный член**
`class_weight=None`	**Веса классов**
`solver="warn"`	**Солвер** `"newton-cg", "lbfgs", "liblinear", "sag", "saga"`
`warm_start=False`	**Использовать ли предыдущие начальные условия**
`l1_ratio`	**Формализация штрафов для ElasticNet**

```
tol=0.0001, intercept_scaling=1, random_state=None, max_iter=100,
multi_class='warn', verbose=0, n_jobs=None
```

Приложения

Банковский скоринг

Задачи с текстами

Бенчмарк для дебита нефти

Прогнозирование спроса

Почти любые индустриальные задачи!

Банковский скоринг

Name	Description	Type
TCS_CUSTOMER_ID	Идентификатор клиента	ID
BUREAU_CD	Код бюро, из которого получен счет	numeric
BKI_REQUEST_DATE	Дата, в которую был сделан запрос в бюро	date
CURRENCY	Валюта договора (ISO буквенный код валюты)	string
RELATIONSHIP	Тип отношения к договору	string
	1 - Физическое лицо	
	2 - Дополнительная карта/Авторизованный пользователь	
	4 - Совместный	
	5 - Поручитель	
	9 - Юридическое лицо	
OPEN_DATE	Дата открытия договора	date
FINAL_PMT_DATE	Дата финального платежа (плановая)	date
TYPE	Код типа договора	string
	1 – Кредит на автомобиль	
	4 – Лизинг. Срочные платежи за наем/пользование транспортным средством, предприятием или оборудованием и т.п.	
	6 – Ипотека – ссудные счета, имеющие отношение к домам, квартирам и прочей недвижимости. Ссуда выплачивается циклично согласно договоренности до тех пор, пока она не будет полностью выплачена или возобновлена.	
	7 – Кредитная карта	
	9 – Потребительский кредит	
	10 – Кредит на развитие бизнеса	
	11 – Кредит на пополнение оборотных средств	
	12 – Кредит на покупку оборудования	
	13 – Кредит на строительство недвижимости	
	14 – Кредит на покупку акций (например, маржинальное кредитование)	
	99 – Другой	
PMT_STRING_84M	Дисциплина (своевременность) платежей. Строка составляется из кодов состояний счета на моменты передачи банком данных по счету в бюро, первый символ - состояние на дату PMT_STRING_START, далее последовательно в порядке убывания дат.	string
	0 – Новый, оценка невозможна	
	X – Нет информации	
	1 – Оплата без просрочек	
	A – Просрочка от 1 до 29 дней	

Банковский скоринг

По описанию и истории клиента → вероятность (оценка) возврата кредита

Нужна логистическая регрессия

есть возможность получать вещественное число в виде ответа

есть более мощные методы (**на решающих деревьях**),

но здесь полезна интерпретация

Все категориальные признаки – OHE-перекодировка

Банковский скоринг

Если решение сводится к

$$a(x) = 1/(1 + \exp(-(w_0 + w_1 X_1 + \ldots + w_n X_n))$$

где все признаки бинарные, то мы составляем скоринговую карту

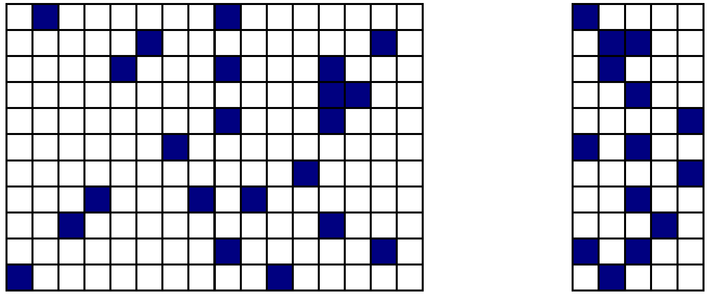
Показатель	Значение показателя	Скоринг-балл
Возраст	До 30 лет	0
	От 30 до 50 лет	35
	Старше 50 лет	28
Образование	Среднее	0
	Среднее специальное	29
	Высшее	35
Состоит ли в браке	Да	25
	Нет	0
Брал ли кредит ранее	Да	41
	Нет	0
Трудовой стаж	Менее 1 года	0
	От 1 до 5 лет	19
	От 5 до 10 лет	24
	Более 10 лет	31

https://wiki.loginom.ru/articles/scorecard.html

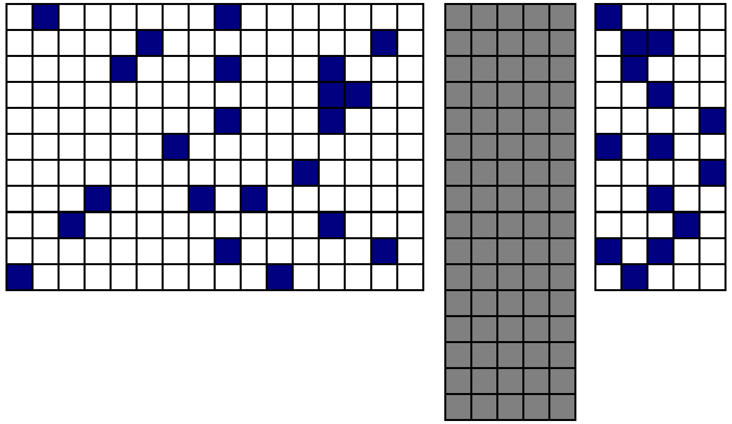
Задачи с текстами

Соревнование «Topical Classification of Biomedical Research Papers»

Данные



Логика решения

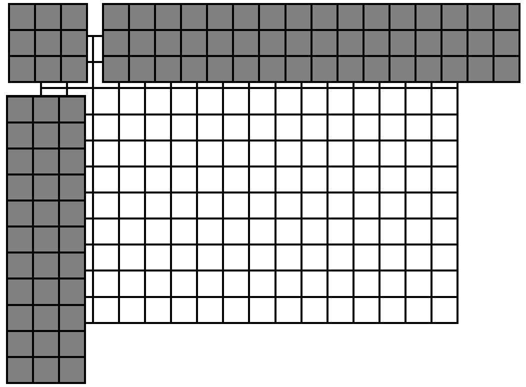


$$X_{q\times n}\cdot W_{n\times l}=Y_{q\times l}$$

$$q=10000,\ n=25000,\ l=83$$

нельзя решать напрямую

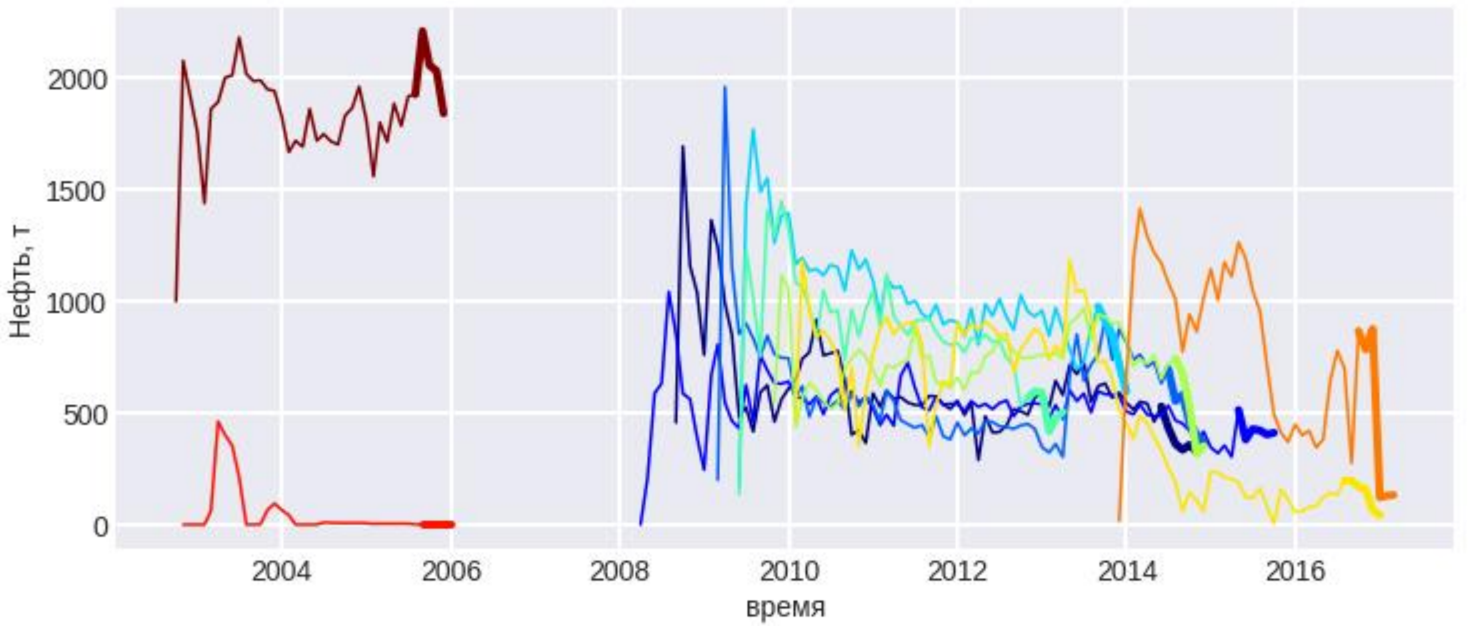
Упрощение: SVD



$$X_{q\times n}\approx U_{q\times k}L_{k\times k}V_{k\times n}$$

$$U_{q\times k}\cdot W_{n\times k}=Y_{q\times l}$$

Бенчмарк прогнозирования дебита нефти



Бенчмарк прогнозирования дебита нефти

$$y_t = \sum_{i=0}^{k} w_{ti} y_{-i}, \quad w_{t0} \geq w_{t1} \geq \ldots$$

соревнование на платформе boosters.pro

https://dyakonov.org/2018/12/23/

Прогнозирование спроса

Спрос товара конкретного id (покупок за следующую неделю)

- **# покупок за k дней**
- **# просмотров за k дней**
- **# корзин за k дней**
- **# дней без покупок**
- **изменение цены за последние k дней**
- **есть ли маркетинговая акция**

...

$$Y = \max\left[\sum_t w_t X_t, 0\right]$$

Проблемы с линейными алгоритмами

+ простой, надёжный, быстрый, популярный метод
+ интерпретируемость (⇒ нахождение закономерностей)
+ интерполяция и экстраполяция
+ может быть добавлена нелинейность, с помощью генерации новых признаков
(дальше – это можно автоматизировать)

– линейная гипотеза вряд ли верна
– в теоретическом обосновании ещё предполагается нормальность ошибок
(зависит от функции ошибок)
– «страдает» из-за выбросов
– признаки в одной шкале и однородные
– статистический вывод регрессии – много предположений
– проблема коррелированных признаков
⇒ необходимость регуляризации, селекции, PCA, data↑

Проблемы мультиколлинеарности

- **большие коэффициенты**
- **большие изменения коэффициентов при добавлении/удалении признаков**
- **нелогичности**
  (чем больше доход, меньше вероятность дать кредит)
- **большое число статистически незначимых оценок коэффициентов**

Зависимость от масштабирования

простая модель	с регуляризацией	пайплайн: нормировка + линейная
нет	есть	нет

Итог

Линейная регрессия ~ матричное уравнение
Но проблема вырожденности
Много методов решают эту проблему с разных сторон

Логистическая регрессия
– деформирование линейной
Но есть вероятностная трактовка!

Интересные ссылки

Песня о RANSAC

https://www.youtube.com/watch?v=1YNjMxxXO-E

Курс Ramesh Sridharan «Statistics for Research Projects: IAP 2015»

http://www.mit.edu/~6.s085/