

академия
больших
данных

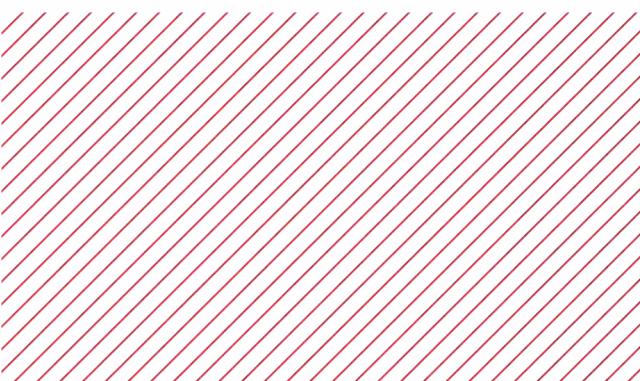


Машинное обучение

Лекция 2. Линейные модели: особенности применения

Кантор Виктор

Программный директор академии MADE





Сегодня на лекции

1. Напоминание: как работают линейные модели
2. Особенности обучения линейных моделей
3. Use-cases
4. Устойчивость линейных моделей
5. Интерпретация других моделей

1. Как работают линейные модели



Линейные модели в задаче регрессии

$$a(x) = w_0 + w_1 x_1 + \cdots + w_d x_d$$



Линейные модели в задаче регрессии

$$a(x) = w_0 + \langle w, x \rangle$$



Линейные модели в задаче регрессии

$$a(x) = w_0 + \langle w, x \rangle$$

$$Q = \sum_{i=1}^l L(y_i, a(x_i)) \rightarrow \min_w$$



Линейные модели в задаче регрессии

$$a(x) = w_0 + \langle w, x \rangle$$

$$Q = \sum_{i=1}^l L(y_i, a(x_i)) \rightarrow \min_w$$

$$L(y_i, a(x_i)) = (y_i - a(x_i))^2 \quad L(y_i, a(x_i)) = |y_i - a(x_i)|$$

Линейные модели в задаче регрессии

$$a(x) = w_0 + \langle w, x \rangle$$

$$Q = \sum_{i=1}^l L(y_i, a(x_i)) + \gamma V(w) \rightarrow \min_w$$

$$L(y_i, a(x_i)) = (y_i - a(x_i))^2 \quad L(y_i, a(x_i)) = |y_i - a(x_i)|$$

$$V(w) = \|w\|_{l2}^2 = \sum_{n=1}^d w_n^2$$

$$V(w) = \|w\|_{l1} = \sum_{n=1}^d |w_n|$$

Линейные модели в задаче регрессии

$$a(x) = w_0 + \langle w, x \rangle$$

$$Q = \sum_{i=1}^l L(y_i, a(x_i)) + \gamma V(w) \rightarrow \min_w$$

Гребневая регрессия
(Ridge regression):

$$V(w) = \|w\|_{l2}^2 = \sum_{n=1}^d w_n^2$$

LASSO (least absolute
shrinkage and selection
operator):

$$V(w) = \|w\|_{l1} = \sum_{n=1}^d |w_n|$$

Линейные модели в задаче регрессии

$$a(x) = w_0 + \langle w, x \rangle$$

$$Q = \sum_{i=1}^l L(y_i, a(x_i)) \rightarrow \min_w$$

$$L(y_i, a(x_i)) = (y_i - a(x_i))^2$$

А без регуляризатора и с квадратичными потерями
получаем привычную нам линейную регрессию

Матричная запись

$$y_i \approx \hat{y}_i = \langle w, x_i \rangle + w_0$$

Если добавить $x_{i0} = 1$:

$$y_i \approx \hat{y}_i = \langle w, x_i \rangle$$

$$y_1 \approx \hat{y}_1 = x_1^T w$$

...

$$y_i \approx \hat{y}_i = x_i^T w$$

...

$$y_l \approx \hat{y}_l = x_l^T w$$

Матричная запись

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_l \end{pmatrix} \approx \begin{pmatrix} \widehat{y}_1 \\ \widehat{y}_2 \\ \dots \\ \widehat{y}_l \end{pmatrix} = \begin{pmatrix} {x_1}^T \\ {x_2}^T \\ \dots \\ {x_l}^T \end{pmatrix} w$$

$$y \approx \hat{y} = Fw$$

$$w = \underset{w}{\operatorname{argmin}} \|y - \hat{y}\|^2$$

Аналитическое решение

$$\frac{\partial(Fw - y)^2}{\partial w} = 2F^T(Fw - y) = 0$$

$$F^T F w = F^T y$$

$$w = (F^T F)^{-1} F^T y$$

Решение в задаче с ℓ_2 регуляризатором

$$\frac{\partial(Fw - y)^2 + \gamma w^2}{\partial w} = 2F^T(Fw - y) + 2\gamma w = 0$$

$$(F^T F + \gamma I)w = F^T y$$

$$w = (F^T F + \gamma I)^{-1} F^T y$$



Линейный классификатор

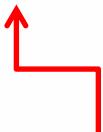
$$a(x) = \begin{cases} 1, & \text{если } f(x) > 0 \\ 0, & \text{если } f(x) \leq 0 \end{cases}$$

$$f(x) = w_0 + w_1 x_1 + \cdots + w_d x_d = w_0 + \langle w, x \rangle$$

Линейный классификатор

$$a(x) = \begin{cases} 1, & \text{если } f(x) > 0 \\ 0, & \text{если } f(x) \leq 0 \end{cases}$$

$$f(x) = w_0 + w_1 x_1 + \cdots + w_d x_d = w_0 + \langle w, x \rangle$$



Confidence score



Пример: логистическая регрессия

$$p_i = \sigma(\langle w_0, x_i \rangle) = \frac{1}{1 + e^{-(w_0 + \langle w, x_i \rangle)}}$$

Пример: логистическая регрессия

$$p_i = \sigma(\langle w_0, x_i \rangle) = \frac{1}{1 + e^{-(w_0 + \langle w, x_i \rangle)}}$$

$$Q = - \sum_{i=1}^{\ell} y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \rightarrow \min_w$$

$$y_i \in \{0, 1\}$$

Почему оцениваются вероятности

$$\mathcal{L} = \prod_{i=1}^{\ell} p_i^{y_i} (1 - p_i)^{(1-y_i)} \rightarrow \max_w$$

Почему оцениваются вероятности

$$\mathcal{L} = \prod_{i=1}^{\ell} p_i^{y_i} (1 - p_i)^{(1-y_i)} \rightarrow \max_w$$

$$\ln \mathcal{L} = \sum_{i=1}^{\ell} y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \rightarrow \max_w$$

Почему оцениваются вероятности

$$\mathcal{L} = \prod_{i=1}^{\ell} p_i^{y_i} (1 - p_i)^{(1-y_i)} \rightarrow \max_w$$

$$\ln \mathcal{L} = \sum_{i=1}^{\ell} y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \rightarrow \max_w$$

$$Q = -\ln \mathcal{L} = - \sum_{i=1}^{\ell} y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \rightarrow \min_w$$

Другие обозначения классов

$$a(x) = \begin{cases} 1, & \text{если } f(x) > 0 \\ -1, & \text{если } f(x) \leq 0 \end{cases}$$

$$f(x) = w_0 + \langle w, x \rangle$$

$$a(x) = sign(w_0 + \langle w, x \rangle)$$

Общий случай оптимизационной задачи

$$Q = \sum_{i=1}^{\ell} L(y_i, f(x_i)) + \gamma V(w) \rightarrow \min_w$$

Функция потерь

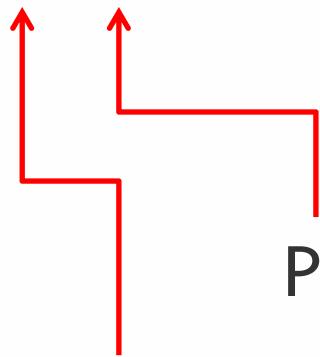
Коэффициент
регуляризации

Регуляризатор

Функция потерь как функция отступа

$$Q = \sum_{i=1}^{\ell} L(M_i) + \gamma V(w) \rightarrow \min_w$$

Функция потерь



Регуляризатор

Коэффициент
регуляризации

Отступ

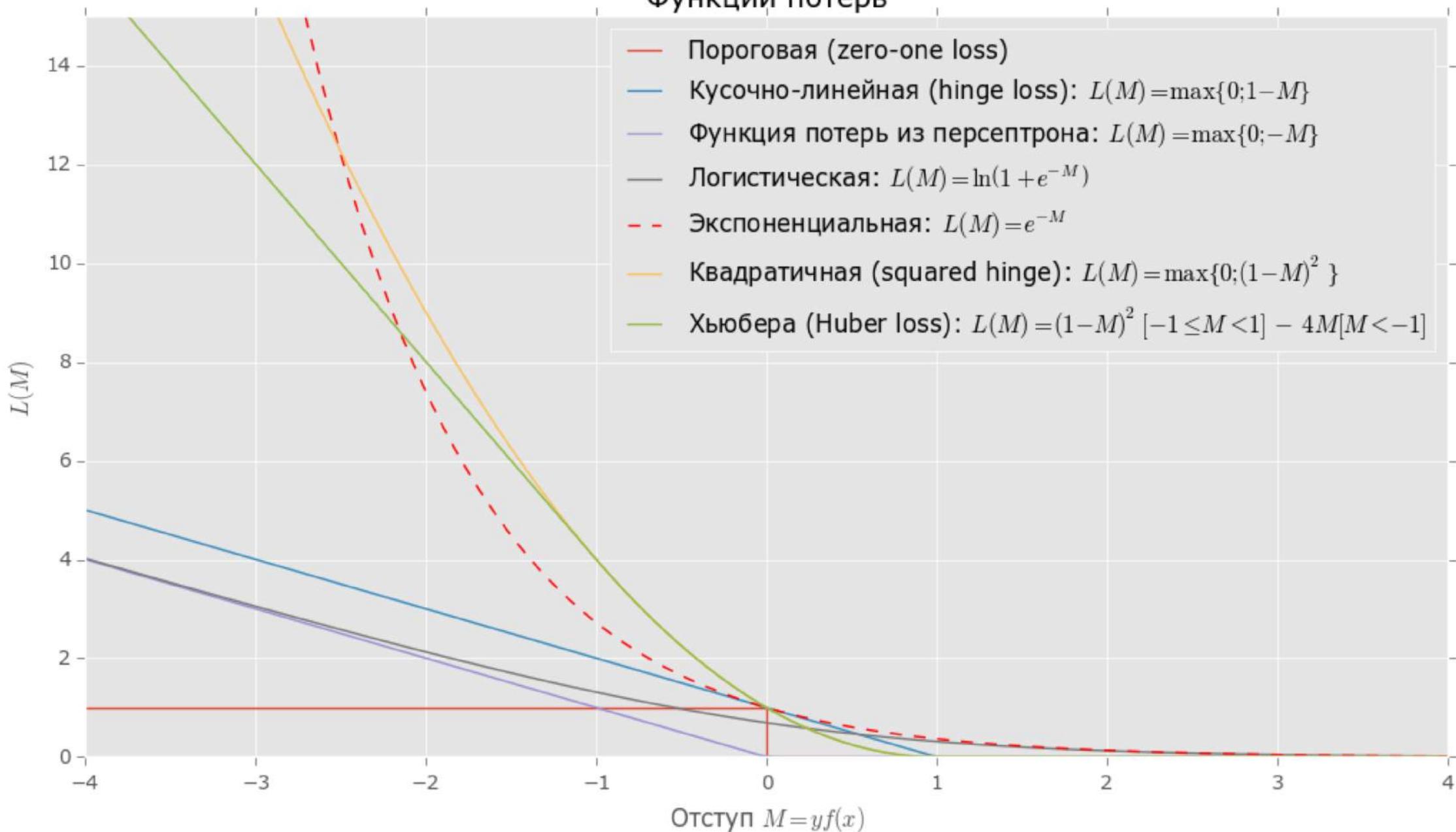
Отступом алгоритма $a(x) = \text{sign}\{f(x)\}$ на объекте x_i называется величина

$$M_i = y_i f(x_i)$$

(y_i - класс, к которому относится x_i)

$$\begin{aligned} M_i \leq 0 &\Leftrightarrow y_i \neq a(x_i) \\ M_i > 0 &\Leftrightarrow y_i = a(x_i) \end{aligned}$$

Функции потерь



Популярные линейные классификаторы

Классификатор	Функция потерь	Регуляризатор
SVM (Support vector machine, метод опорных векторов)	$L(M) = \max\{0, 1 - M\} = (1 - M)_+$	$\sum_{k=1}^m w_k^2$
Логистическая регрессия	$L(M) = \log(1 + e^{-M})$	Обычно $\sum_{k=1}^m w_k^2$ или $\sum_{k=1}^m w_k $

Градиентный спуск

$$\nabla_w \tilde{Q} = \sum_{i=1}^l \nabla L(M_i) = \sum_{i=1}^l L'(M_i) \frac{\partial M_i}{\partial w}$$

$$M_i = y_i \langle w, x_i \rangle \Rightarrow \frac{\partial M_i}{\partial w} = y_i x_i$$

$$\nabla \tilde{Q} = \sum_{i=1}^l y_i x_i L'(M_i)$$

$$w_{k+1} = w_k - \gamma_k \sum_{i=1}^l y_i x_i L'(M_i)$$

Стохастический градиентный спуск (SGD)

$$w_{k+1} = w_k - \gamma_k \sum_{i=1}^l y_i x_i L'(M_i)$$

$$w_{k+1} = w_k - \gamma_k y_i x_i L'(M_i)$$

x_i – случайный элемент обучающей выборки

Код: применение модели

```
import numpy as np

def f(x):
    return np.dot(w, x) + w0

def a(x):
    return 1 if f(x) > 0 else -1
```

Код: обучение модели

```
from random import randint
import numpy as np

def loss(x, y):
    return max([0, 1 - y * f(x)])

def der_loss(x, y):
    return -1.0 if 1 - y * f(x) > 0 else 0.0

def fit(X_train, y_train):
    w = np.random.randn(X_train.shape[1])
    w0 = np.random.randn()
    for k in range(10000):
        rand_index = randint(0, len(X_train) - 1)
        x = X_train[rand_index]
        y = y_train[rand_index]
        step = 0.01
        w -= step * y * x * der_loss(x, y)
        w0 -= step * y * der_loss(x, y)
```



Эвристики для шага

$$\gamma_k = \frac{1}{\alpha + k}$$

$$\gamma_k = \frac{1}{\sqrt{\alpha + k}}$$

$$\gamma_k = (\alpha + k)^{-\beta}$$

$\gamma_k = \tau \beta_k$, β_k - шаг наискорейшего спуска

Эвристики для шага

$$\gamma_k = \frac{1}{\alpha + k}$$

$$\gamma_k = \frac{1}{\sqrt{\alpha + k}}$$

$$\gamma_k = (\alpha + k)^{-\beta}$$

$$\gamma_k = \tau \beta_k$$

Пример: шаг в Vowpal Wabbit

$$\gamma_k = s \left(\frac{i}{i + k} \right)^p$$



Вопрос к аудитории

Как сделать более качественный критерий останова, чем на предыдущем слайде?



Упражнение

1. Вывести формулу обновления весов с учетом добавления регуляризатора
2. Попробовать воспроизвести код со слайда на любом стандартном датасете из `sklearn`, добавить квадратичный регуляризатор и добиться качества, сравнимого с качеством `LinearSVC` из `sklearn.linear_model`



Дополнительные темы: линейные модели

1. Почему $\| \cdot \|_1$ регуляризатор разреживает вектор весов
2. SVM и его вывод из максимизации ширины разделяющей полосы
3. Kernel Trick в SVM и двойственная задача
4. Стратегии построения многоклассовых линейных классификаторов
5. Кросс-энтропия и многоклассовая логистическая регрессия
6. Semi-supervised версии SVM и логистической регрессии



Semi-supervised и регуляризация

$$Q = \sum_{i=1}^{\ell} L(M_i) + \gamma V(w) \rightarrow \min_w$$

Semi-supervised и регуляризация

$$Q = \sum_{i=1}^{\ell} L(M_i) + \gamma V(w) \rightarrow \min_w$$

$$L(M_i) = (1 - M_i)_+ \quad V(w) = \sum_{i=\ell+1}^{\ell+t} (1 - a(x_i)f(x_i))_+$$

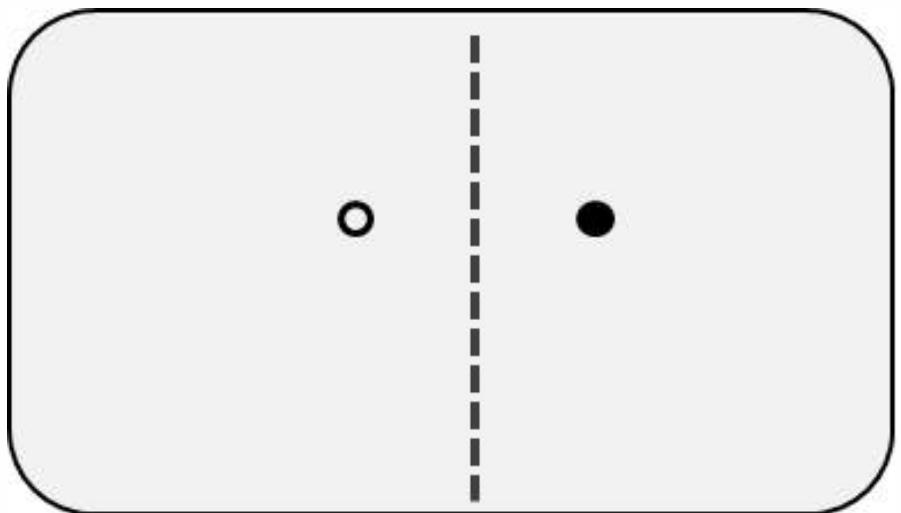
Semi-supervised и регуляризация

$$Q = \sum_{i=1}^{\ell} L(M_i) + \gamma V(w) \rightarrow \min_w$$

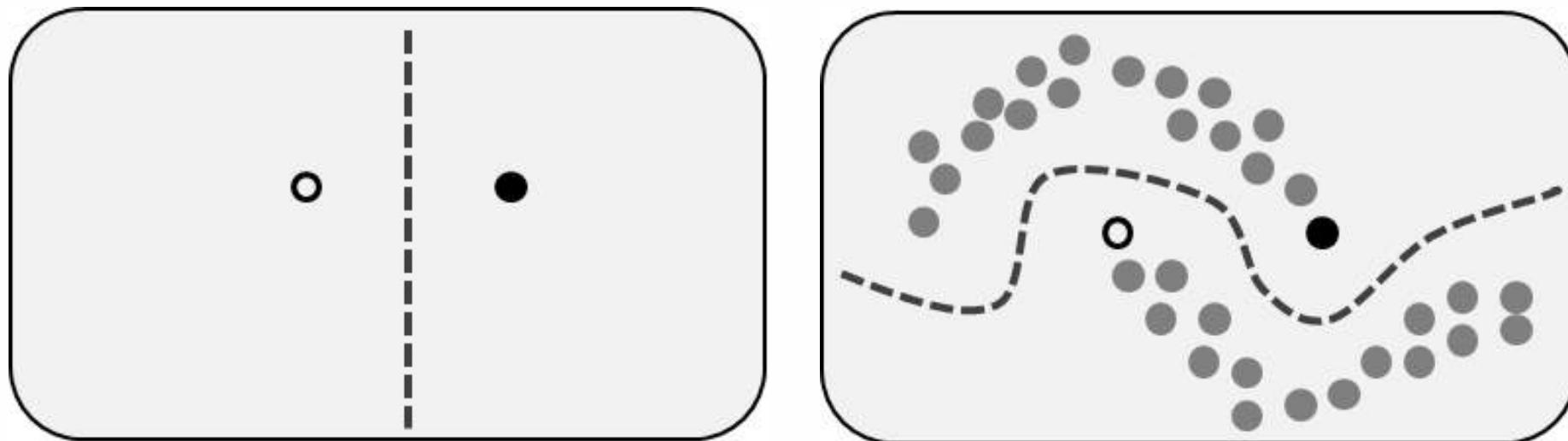
$$L(M_i) = (1 - M_i)_+ \quad V(w) = \sum_{i=\ell+1}^{\ell+t} (1 - a(x_i)f(x_i))_+$$

$$L(M_i) = \log(1 + e^{-M_i}) \quad V(w) = \sum_{i=\ell+1}^{\ell+t} p_i \ln p_i + (1 - p_i) \ln(1 - p_i)$$

Общая идея semi-supervised learning



Общая идея semi-supervised learning





Дополнительные темы: оптимизация

1. Mini-batch SGD
2. Метод Ньютона-Рафсона
3. Квазиньютоновские методы и BFGS
4. Теорема Куна-Таккера и ее геометрическое объяснение. Объяснение связи ограничений на веса и добавления штрафа в оптимизационную задачу, двойственная задача в SVM



Литература для изучения / повторения

1. Лекции К.В. Воронцова – превосходный вариант для первого знакомства
2. Лекции Е. Соколова (ФКН ВШЭ) – чуть подробнее и более актуальные
3. Elements of statistical learning, Hastie, Tibshirani, Friedman - классика

2. Особенности применения линейных моделей



Нормировка признаков

1. Интерпретируемость весов
2. Штрафы в регуляризации
3. Численная оптимизация



Интерпретируемость весов

Пример: оценка стоимости квартиры (задача регрессии)

$x_{(1)}$ — Удаленность дома от центра города (в метрах)

$x_{(2)}$ — Количество комнат в квартире

Получили веса:

$$w_{(1)} = -100$$

$$w_{(2)} = 7\ 000\ 000$$



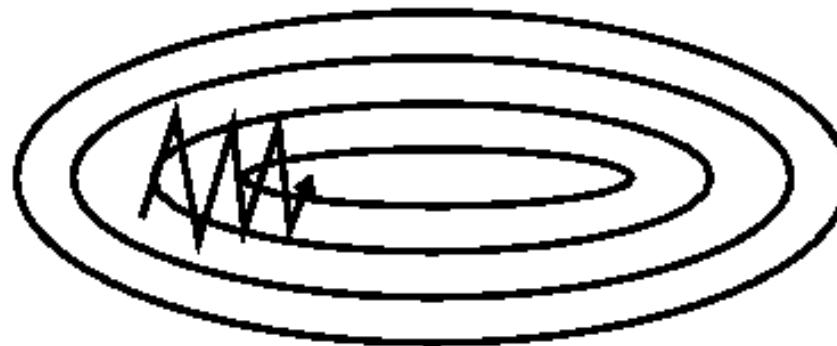
Веса в регуляризаторе

$$V(w) = \|w\|_{l2}^2 = \sum_{n=1}^d w_n^2$$

$$V(w) = \|w\|_{l1} = \sum_{n=1}^d |w_n|$$

Численная оптимизация

Траектория численной оптимизации при разном масштабе признаков:



Центрирование и нормирование

1. На обучающей выборке:

$$f' = \frac{f - a}{b}$$

Например: a – среднее значение, b – дисперсия

Другой вариант: $b = \max\{f\} - \min\{f\}$

Иногда даже так: $b = \sqrt{\max\{f\} - \min\{f\}}$

Центрирование и нормирование

1. На обучающей выборке:

$$f' = \frac{f - a}{b}$$

Например: a – среднее значение, b – дисперсия

Другой вариант: $b = \max\{f\} - \min\{f\}$

Иногда даже так: $b = \sqrt{\max\{f\} - \min\{f\}}$

2. На тестовой выборке – два варианта:

- а) Пересчитываем a и b
- б) Используем те же a и b



Пересчитывать ли a и b

На тестовой выборке – два варианта:

- a) Пересчитываем a и b
- b) Используем те же a и b

Если пересчитываем – модель работает с тем же диапазоном значений. Это уместно, если важны значения признаков **относительно выборки**

Если используем те же – возможны выходы значения признака за диапазон из обучающей выборки. Это уместно, если в этих случаях модель должна **экстраполировать прогноз** и может это делать правильно.

Взвешивание корнем и логарифмом

Когда нужно сгладить большие значения признаков, допустимо заменить признак его корнем или логарифмом:

$$f' = \sqrt{f}$$

$$f' = \ln(1 + f)$$

Например, иногда прием применяется с частотами слов в тексте

Функции потерь и распределение таргета

$$Q = \sum_{i=1}^l (y_i - a(x_i))^2 \rightarrow \min_w$$

Функции потерь и распределение таргета

$$Q = \sum_{i=1}^l (y_i - a(x_i))^2 \rightarrow \min_w$$

$$\mathcal{L} = \prod_{i=1}^l \frac{1}{\sqrt{2\sigma^2}} e^{-\frac{(y_i - a(x_i))^2}{2\sigma^2}} \rightarrow \max_w$$

Функции потерь и распределение таргета

$$Q = \sum_{i=1}^l (y_i - a(x_i))^2 \rightarrow \min_w$$

$$\mathcal{L} = \prod_{i=1}^l \frac{1}{\sqrt{2\sigma^2}} e^{-\frac{(y_i - a(x_i))^2}{2\sigma^2}} \rightarrow \max_w$$

$$a(x) \approx \mathbb{E}(y|x), \quad p(y|x) = \mathcal{N}(\mathbb{E}(y|x), Var(y|x))$$

Функции потерь и распределение таргета

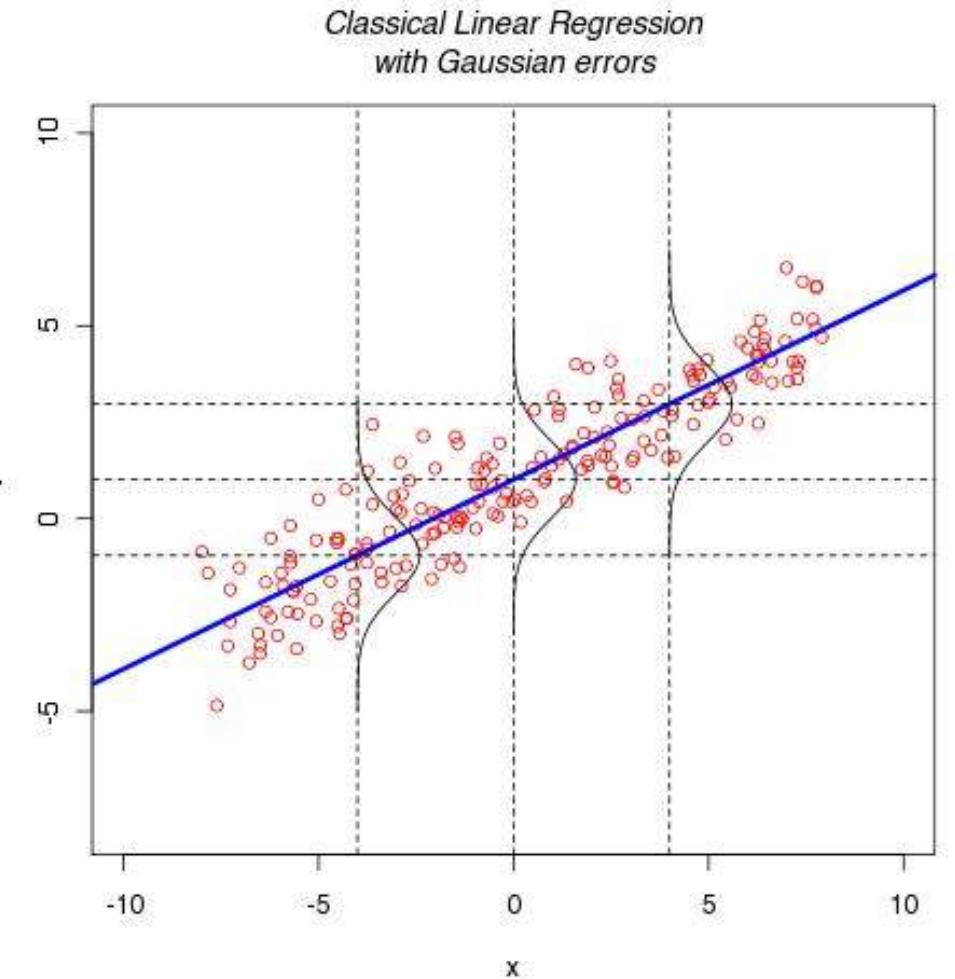
$$Q = \sum_{i=1}^l (y_i - a(x_i))^2 \rightarrow \min_w$$

$$\mathcal{L} = \prod_{i=1}^l \frac{1}{\sqrt{2\sigma^2}} e^{-\frac{(y_i - a(x_i))^2}{2\sigma^2}} \rightarrow \max_w$$

$$a(x) \approx \mathbb{E}(y|x), \quad p(y|x) = \mathcal{N}(\mathbb{E}(y|x), Var(y|x))$$

$$Var(y|x) = \sigma^2 = const(x)$$

Функции потерь и распределение таргета



$$Q = \sum_{i=1}^l (y_i - a(x_i))^2 \rightarrow \min_w$$

$$\mathcal{L} = \prod_{i=1}^l \frac{1}{\sqrt{2\sigma^2}} e^{-\frac{(y_i - a(x_i))^2}{2\sigma^2}} \rightarrow \max_w$$

$$a(x) \approx \mathbb{E}(y|x), \quad p(y|x) = \mathcal{N}(\mathbb{E}(y|x), \text{Var}(y|x))$$

$$\text{Var}(y|x) = \sigma^2 = \text{const}(x)$$

т.е. шум гомоскедастичный

Преобразование Бокса-Кокса

Однопараметрический вариант:

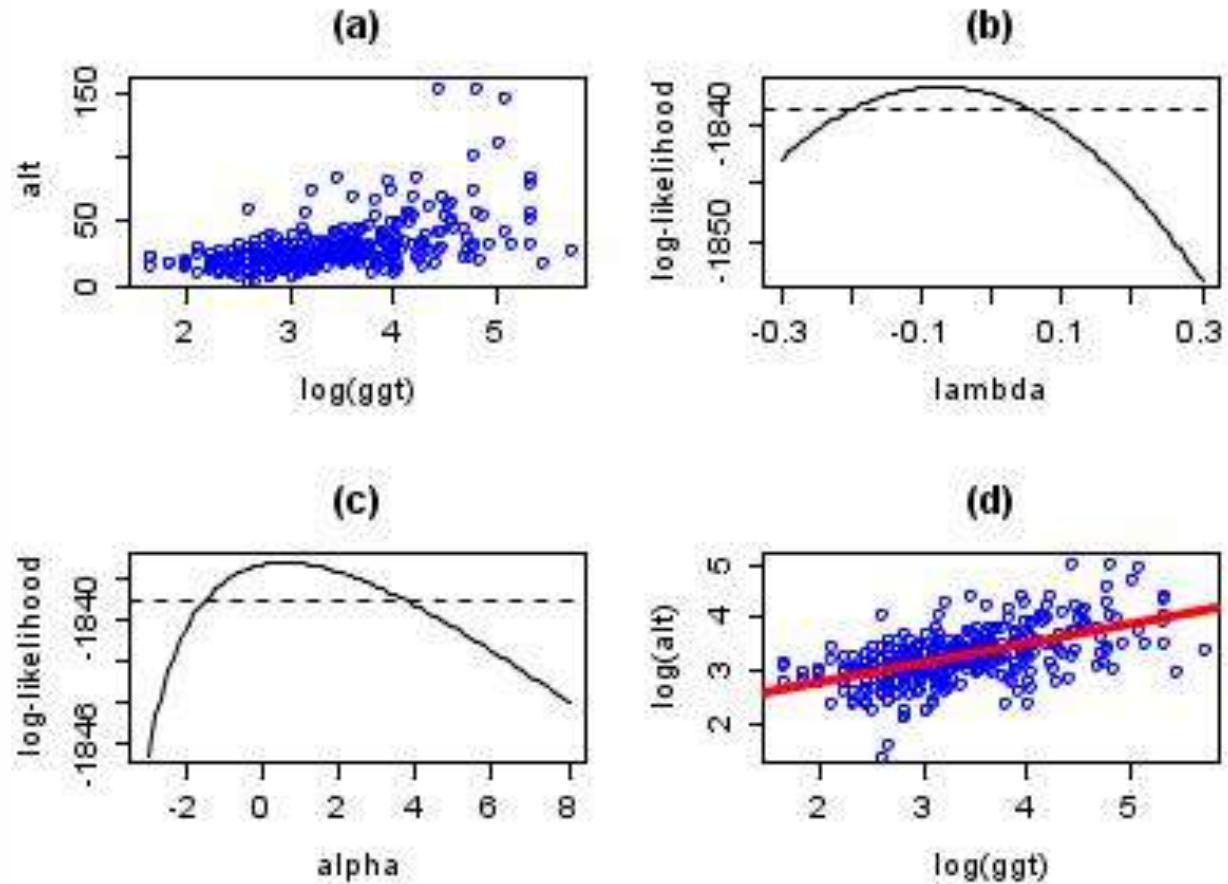
$$y_i^{(\lambda)} = \begin{cases} \frac{y_i^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0, \\ \ln y_i & \text{if } \lambda = 0, \end{cases}$$

Двухпараметрический вариант:

$$y_i^{(\lambda)} = \begin{cases} \frac{(y_i + \lambda_2)^{\lambda_1} - 1}{\lambda_1} & \text{if } \lambda_1 \neq 0, \\ \ln(y_i + \lambda_2) & \text{if } \lambda_1 = 0, \end{cases}$$

Преобразование Бокса-Кокса

Обычно – преобразование таргета, но с признаками тоже можно пробовать



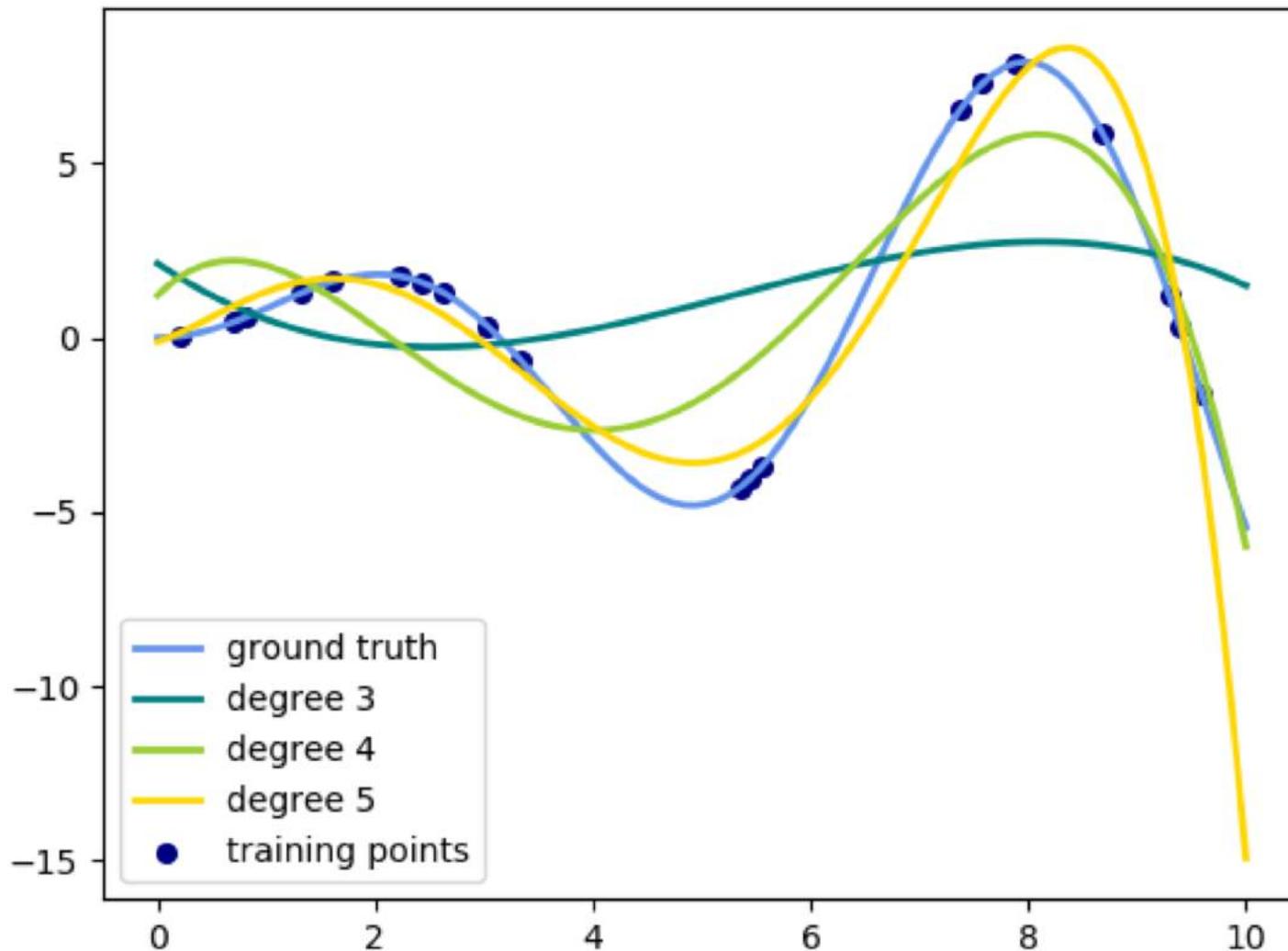


Преобразование Йео-Джонсона

Еще один вариант, который может работать с нулевыми и отрицательными значениями:

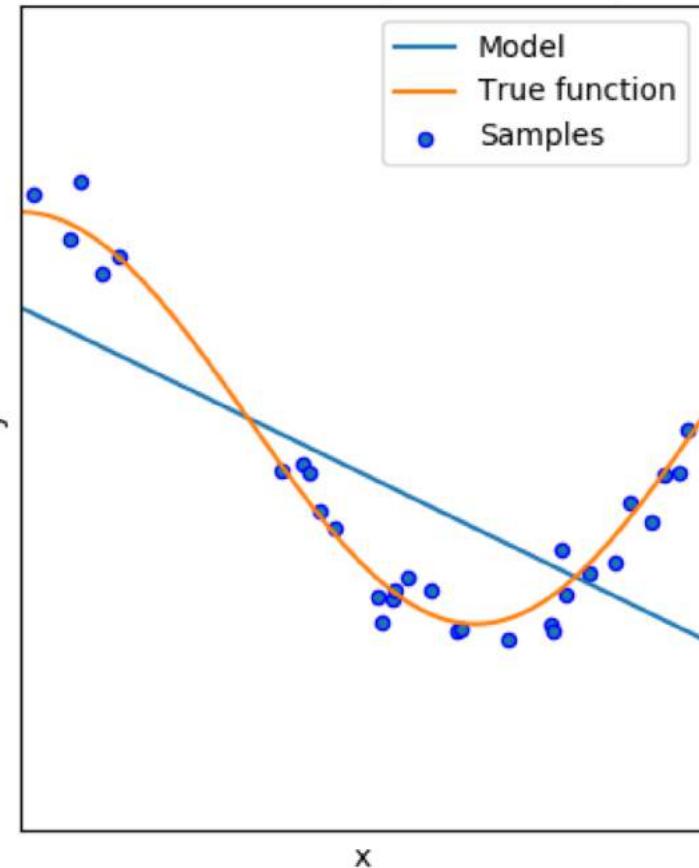
<https://www.stat.umn.edu/arc/yjpower.pdf>

Полиномиальные признаки: регрессия

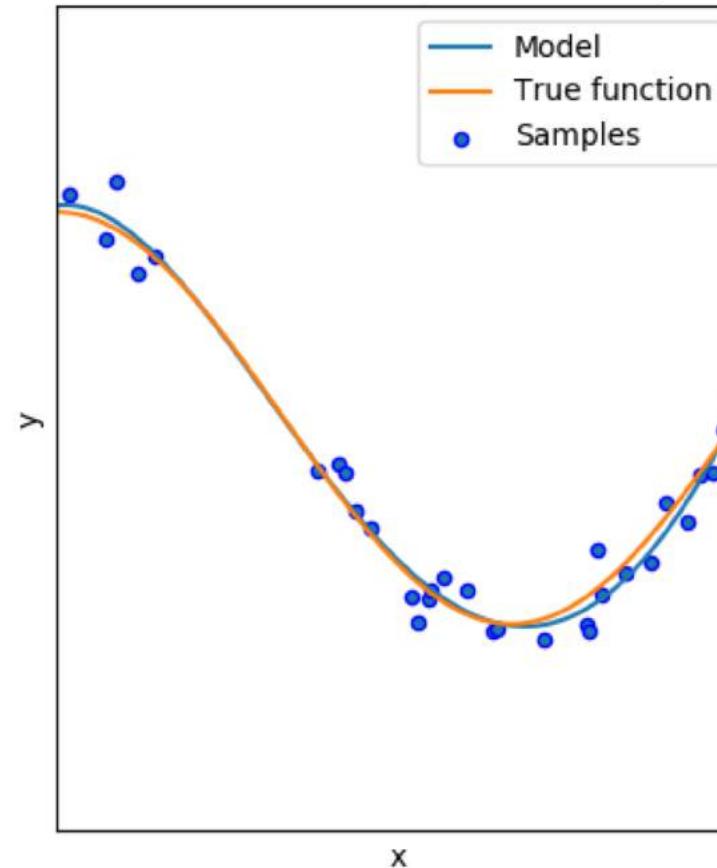


Полиномиальные признаки: регрессия

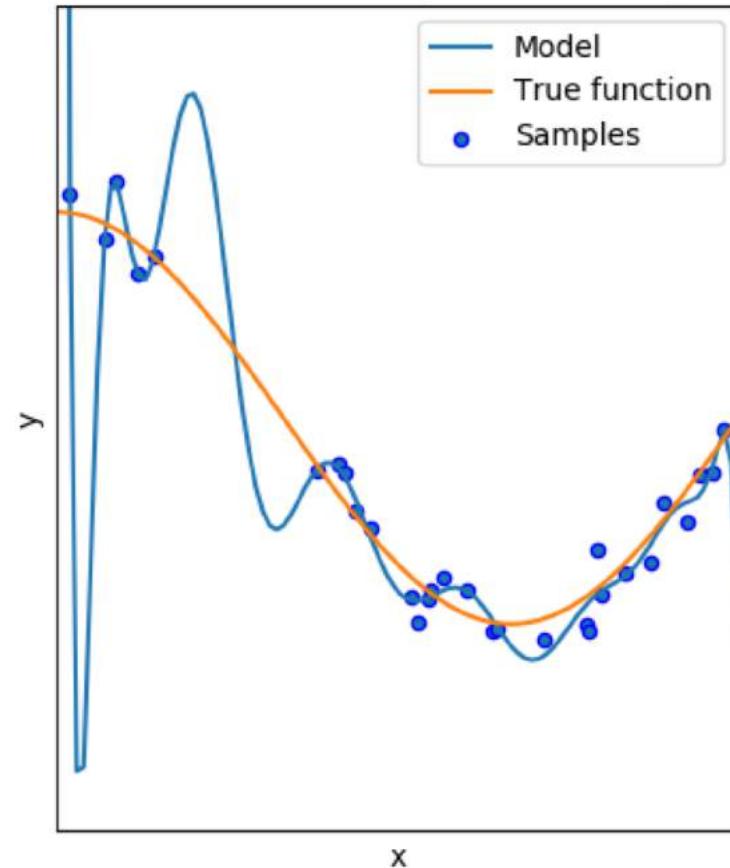
Degree 1
MSE = 4.08e-01(+/- 4.25e-01)



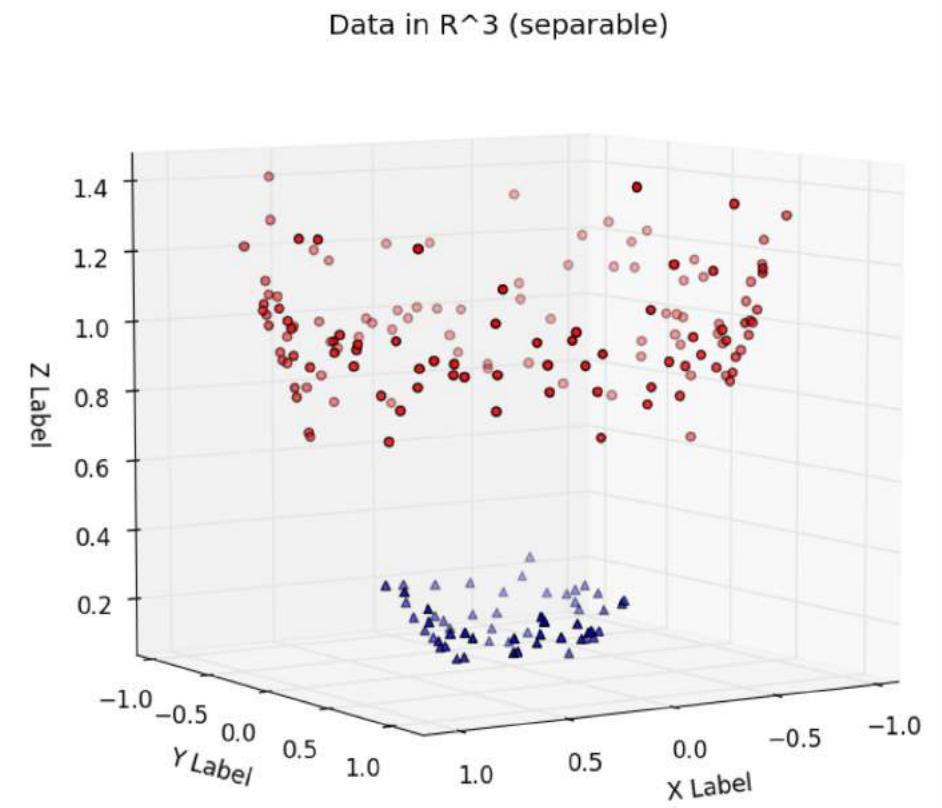
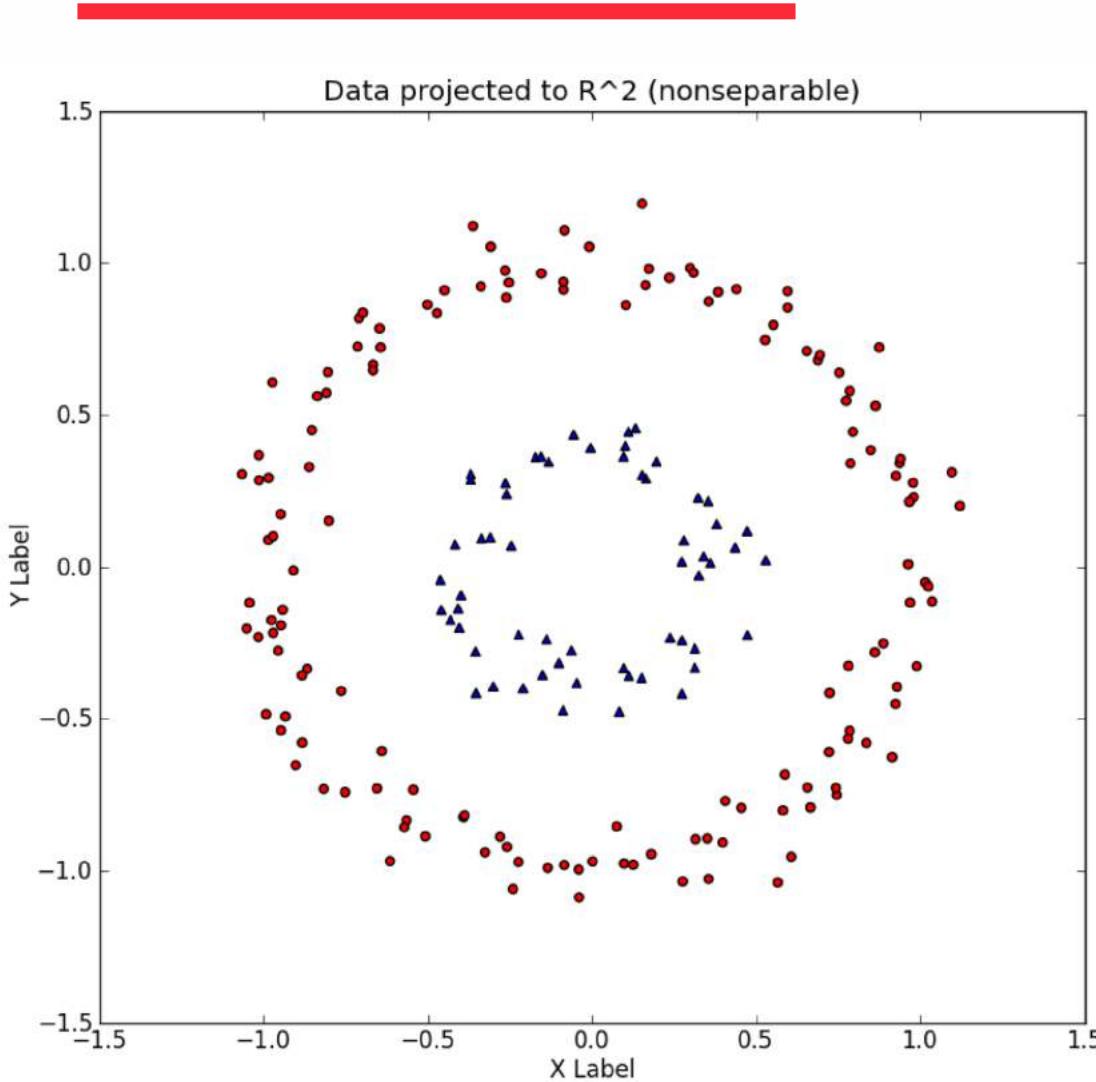
Degree 4
MSE = 4.32e-02(+/- 7.08e-02)



Degree 15
MSE = 1.83e+08(+/- 5.48e+08)



Полиномиальные признаки: классификация





Kernel trick

В SVM идея преобразования признаков была развита дальше



Kernel trick

В SVM идея преобразования признаков была развита дальше

Вместо разделяющей поверхности $w_0 + \langle w, x \rangle = 0$, можно строить $w_0 + K(w, x) = 0$, где $K(w, x) = \langle \psi(w), \psi(x) \rangle$, а ψ – отображение из исходного пространства признаков в пространство более высокой размерности



Kernel trick

В SVM идея преобразования признаков была развита дальше

Вместо разделяющей поверхности $w_0 + \langle w, x \rangle = 0$, можно строить $w_0 + K(w, x) = 0$, где $K(w, x) = \langle \psi(w), \psi(x) \rangle$, а ψ – отображение из исходного пространства признаков в пространство более высокой размерности

Задав $K(w, x)$, мы избегаем трудоемкого явного преобразования признаков. Например, для $K(w, x) = e^{-\beta \|w-x\|^2}$ (радиальное ядро) мы неявно делаем отображение в бесконечномерное (!) пространство признаков

Kernel trick

В SVM идея преобразования признаков была развита дальше

Вместо разделяющей поверхности $w_0 + \langle w, x \rangle = 0$, можно строить $w_0 + K(w, x) = 0$, где $K(w, x) = \langle \psi(w), \psi(x) \rangle$, а ψ – отображение из исходного пространства признаков в пространство более высокой размерности

Задав $K(w, x)$, мы избегаем трудоемкого явного преобразования признаков. Например, для $K(w, x) = e^{-\beta \|w-x\|^2}$ (радиальное ядро) мы неявно делаем отображение в бесконечномерное (!) пространство признаков

Но с появлением ансамблей деревьев SVM с радиальным ядром утратил популярность



Методы оптимизации

SGD и его вариации более применимы для обучения на больших выборках, чем обычный градиентный спуск или методы второго порядка



Методы оптимизации

SGD и его вариации более применимы для обучения на больших выборках, чем обычный градиентный спуск или методы второго порядка

Если выборка не очень большая – можно использовать даже методы второго порядка (например, метод Ньютона-Рафсона)



Методы оптимизации

SGD и его вариации более применимы для обучения на больших выборках, чем обычный градиентный спуск или методы второго порядка

Если выборка не очень большая – можно использовать даже методы второго порядка (например, метод Ньютона-Рафсона)

Если же выборка большая – можно сначала обучить веса с помощью SGD, а затем на батчах дообучить веса чем-то посложнее (например, L-BFGS)

3. Примеры применения



Use-cases

1. Построение бейзлайна
2. Построение интерпретируемой модели
3. Построение модели с простым внедрением
4. Обучение на разреженных признаках
5. Построение быстрой модели
6. Быстрое дообучение и онлайн-обучение
7. Обучение модели на большой выборке
8. «Упрощение» сложной правиловой модели

Построение бейзайна: пример

История Александра Дьяконова про контест Gazprom Neft SmartOil Contest:
<https://dyakonov.org/2018/12/23/как-бенчмарк-попал-в-призы/>

	name	score	place
	Viktor Yanush (MMP, MSU)	110.144	1
	Pavel Mazaev (MMP MSU)	126.403	2
	Петя Иванович	128.927	3
	Sergey Serov (MMP, MSU, Russia)	131.354	4
	Kozlovtsев Константин (MMP, MSU)	131.703	5
	Kodryan MMP MSU	133.554	6
	Hitch & Hitch Production	133.716	7
	flowMaster	139.780	8
	Ыгба	141.766	9
	Я люблю нефть	144.918	10

	name	score	place
	CheatCode	65.3493	1
	NeftGirl	65.7223	2
	c1	66.1565	3
	Бенчмарк ВМК МГУ 2018	66.4530	4
	Honey Badger	67.4968	5
	Rav1kus	67.4993	6
	Pavel Mazaev (MMP MSU)	67.6436	7
	V&V	67.6590	8
	baklajan	67.8148	9
	Ыгба	68.0727	10

Построение бейзайна: пример

История Александра Дьяконова про контест Gazprom Neft SmartOil Contest:

<https://dyakonov.org/2018/12/23/как-бенчмарк-попал-в-призы/>

$$\sum_{s,t} \left| y_t^s - \sum_{i=0}^k w_{ti} y_{-i}^s \right| \rightarrow \min, \quad w_{t0} \geq w_{t1} \geq \dots \geq 0$$

Построение интерпретируемой модели

Показатель	Значение (диапазон значений) показателя	Скоринг-балл
Возраст	До 30 лет	30
	35 - 50 лет	35
	Старше 50 лет	28
	Среднее	22
Образование	Средне специальное	29
	Высшее	35
Состоит ли в браке	Да	25
	Нет	12
Брал ли кредит ранее	Да	41
	Нет	22
Трудовой стаж	Менее 1 года	16
	От 1 до 5 лет	19
	От 5 до 10 лет	24
	Более 10 лет	31
Наличие автомобиля	Да	49
	Нет	18
Возраст автомобиля	Менее 3-х лет	45
	От 3 до 7 лет	25
	Старше 7 лет	18

Можно изучить полученные веса и объяснить их

Для усиления эффекта можно разбить признаки на сегменты, сделав их разреженными

Построение модели с простым внедрением

Показатель	Значение (диапазон значений) показателя	Скоринг-балл
Возраст	До 30 лет	30
	35 - 50 лет	35
	Старше 50 лет	28
Образование	Среднее	22
	Средне специальное	29
Брал ли кредит ранее	Высшее	35
	Да	25
Состоит ли в браке	Нет	12
	Да	41
Трудовой стаж	Нет	22
	Менее 1 года	16
Наличие автомобиля	От 1 до 5 лет	19
	От 5 до 10 лет	24
	Более 10 лет	31
Возраст автомобиля	Да	49
	Нет	18
Место жительства	Менее 3-х лет	45
	От 3 до 7 лет	25
	Старше 7 лет	18

Суммировать баллы из таблички сможет и человек без особого образования

Написать код, который это делает в продакшене сервиса тоже очень просто



Обучение на разреженных признаках

Пример: чеки в магазине

# Примера	Позиции из чека
1	iPhone 11 красный, AirPods 2, AppleWatch series 5
2	iPad Air 2019, Apple Pencil
3	iPhone Xr желтый, AppleWatch series 3
4	MacBook Air 2019
...	...

Обучение на разреженных признаках

Пример: one hot encoding

№	Значение категориального признака			
	1	Калуга		
	2	Самара		
	3	Новосибирск		
№	Белая Калитва	Калуга	Самара	...
1	0	1	0	...
2	0	0	1	...
3	0	0	0	...
4	1	0	0	...
...

Обучение на разреженных признаках

Пример: тексты в bag of words представлении

№	Текст			
1	Экран супер, батарея норм			
2	Купил вчера, все супер, но сегодня перестал работать			
3	Очень доволен покупкой			
4	Цена супер! Как продам почку обязательно возьму			
№	супер	перестал	доволен	...
1	1	0	0	...
2	1	1	0	...
3	0	0	1	...
4	1	0	0	...
...

Тексты: TF-IDF

$$w_{x,y} = tf_{x,y} \times \log \left(\frac{N}{df_x} \right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

Вариации TF

Variants of term frequency (tf) weight

weighting scheme	tf weight
binary	0, 1
raw count	$f_{t,d}$
term frequency	$f_{t,d} / \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Вариации IDF

Variants of inverse document frequency (idf) weight

weighting scheme	idf weight ($n_t = \{d \in D : t \in d\} $)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left(\frac{N}{1 + n_t} \right)$
inverse document frequency max	$\log \left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t} \right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

Natural language processing

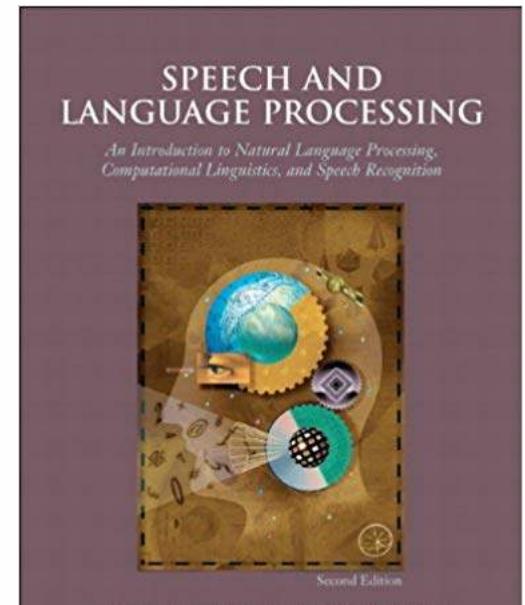
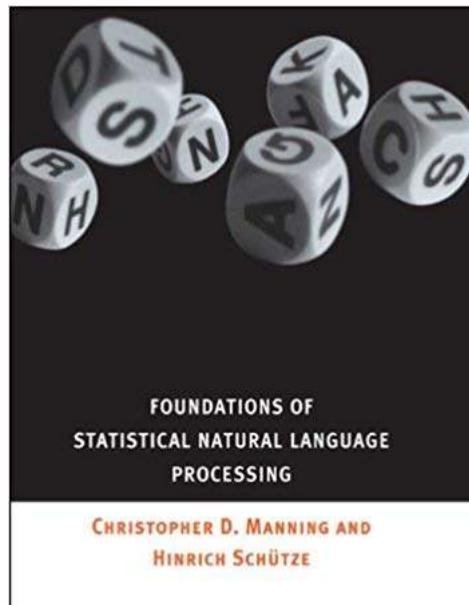
Олдскульные источники:

Foundations of statistical natural language processing, Manning

Speech and language processing, Jurafsky

Актуальное:

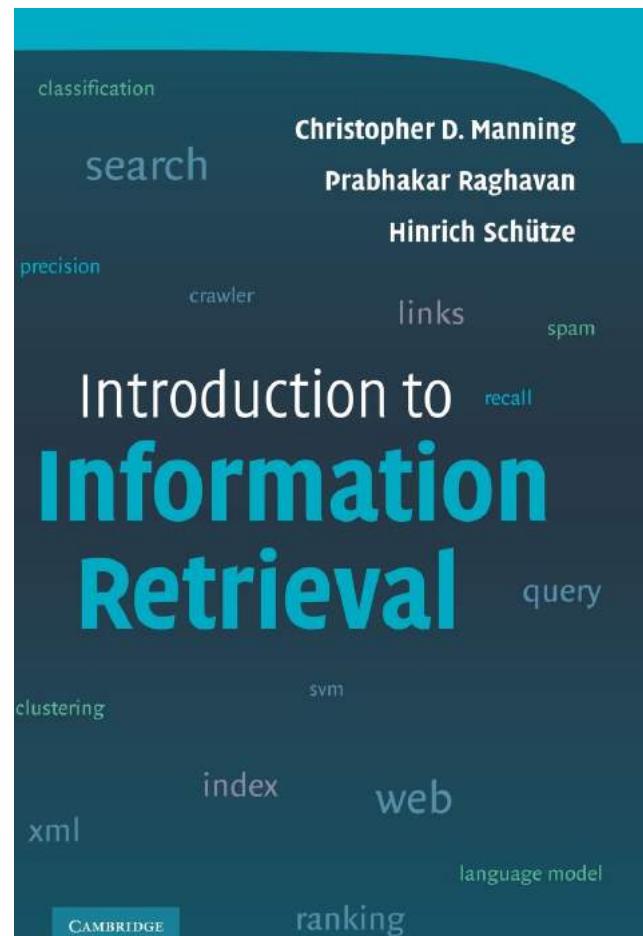
CS224n: Natural Language Processing with Deep Learning



Information retrieval

Introduction to Information Retrieval, Manning, Raghavan, Schütze

Довольно много воды, книга уже по меркам
ML «древняя», но полистать полезно



N-граммы из слов

Пример: биграммы

		№	Текст		
№	экран супер	перестал работать	доволен покупкой
1	1	0	0
2	0	1	0
3	0	0	1
4	0	0	0
...

N-граммы из букв

Пример: 4-граммы

Вопрос: чем это может быть полезно?

№	Текст			
1	Экран супер, батарея норм			
2	Купил вчера, все супер, но сегодня перестал работать			
3	Очень доволен покупкой			
4	Цена супер! Как продам почку обязательно возьму			
№	экра	кран	супе	...
1	1	1	1	...
2	0	0	1	...
3	0	0	0	...
4	0	0	1	...
...

Бейзлайн в текстовой классификации

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.pipeline import Pipeline

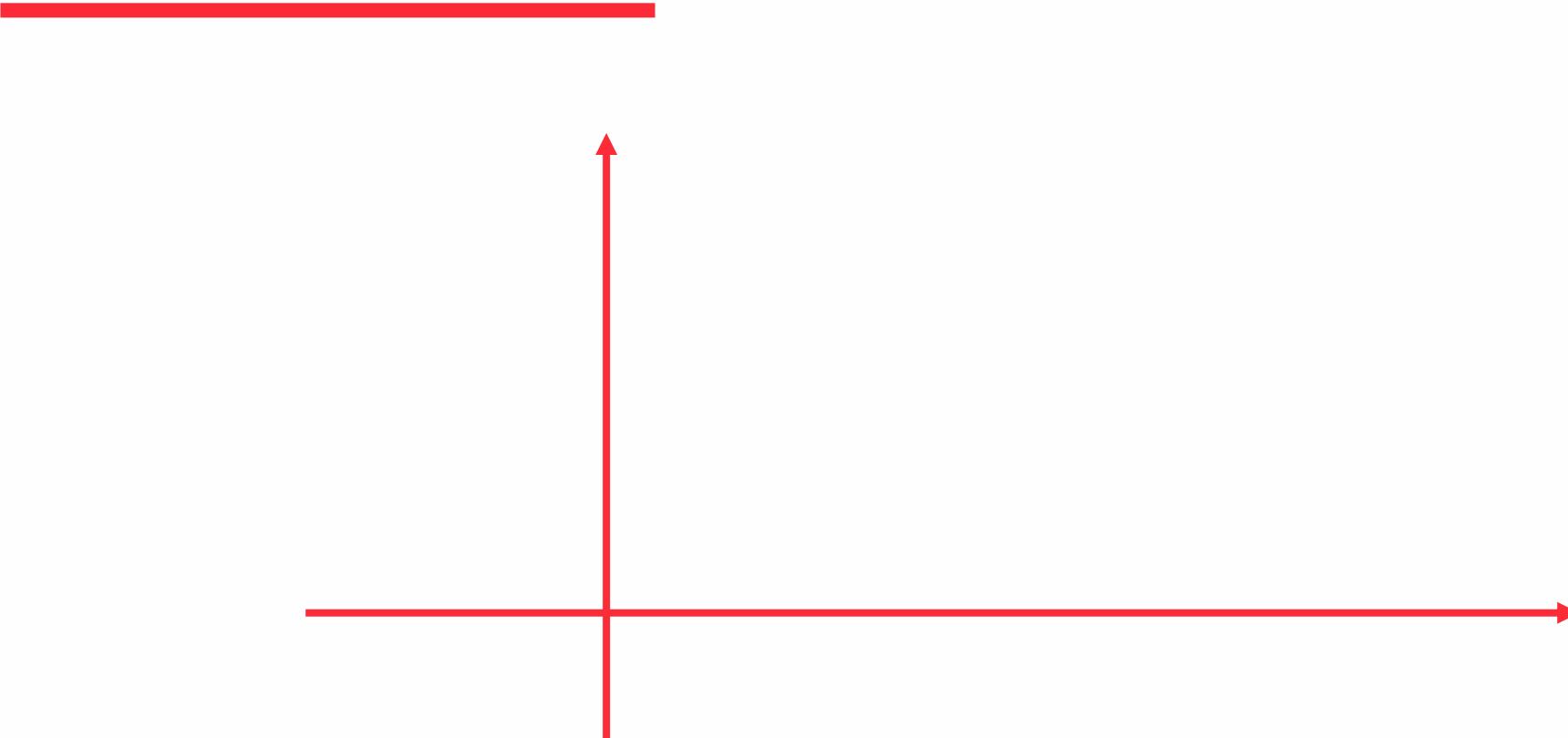
pipeline = Pipeline([
    ('vec', CountVectorizer()),
    ('tfidf', TfidfTransformer()),
    ('clf', LogisticRegression(penalty='l1')),
])
pipeline.fit(train_texts, y_train)
test_predictions = pipeline.predict(test_texts)
```



Задача про тексты

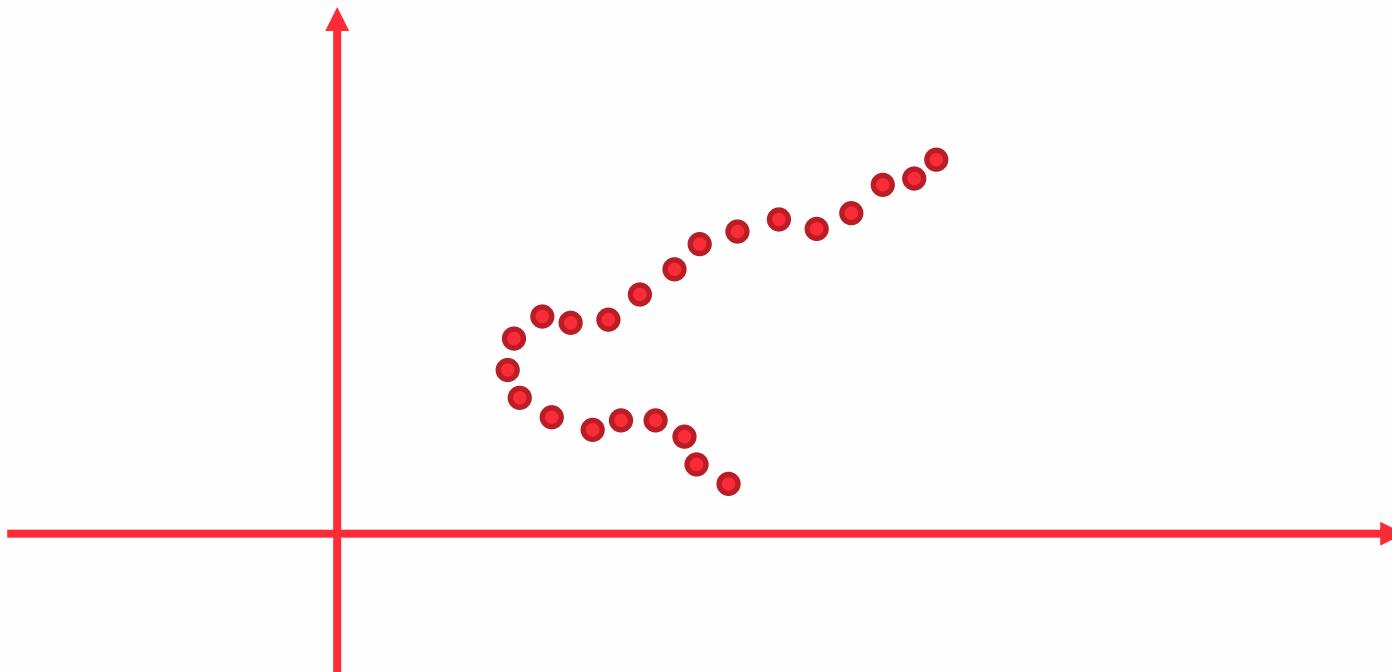


Задача про тексты





Задача про тексты





Построение быстрой модели

Умножить признаки на веса – быстрая операция, отсюда есть две надежды:

- 1) когда важна скорость работы модели, можно делать что-то разумное, но прогнозирующее быстрее, чем, например, ансамбли деревьев
- 2) когда нужно отскорить много объектов и выбрать топ, сначала можно отскорить все быстрой линейной моделью, а потом ее топ отскорить чем-то посложнее



Построение быстрой модели

Умножить признаки на веса – быстрая операция, отсюда есть две надежды:

- 1) когда важна скорость работы модели, можно делать что-то разумное, но прогнозирующее быстрее, чем, например, ансамбли деревьев
- 2) когда нужно отскорить много объектов и выбрать топ, сначала можно отскорить все быстрой линейной моделью, а потом ее топ отскорить чем-то посложнее

На практике не всегда это работает:

- LightGBM так оптимизирован и так быстро работает, что если у вас на 10-20к деревьев, тот же VowpalWabbit вычисляет прогноз примерно столько же времени по порядку величины
- Но с более «тяжеловесными» алгоритмами может сработать

Онлайн-обучение

```
from random import randint
import numpy as np

def loss(x, y):
    return max([0, 1 - y * f(x)])

def der_loss(x, y):
    return -1.0 if 1 - y * f(x) > 0 else 0.0

def fit(X_train, y_train):
    w = np.random.randn(X_train.shape[1])
    w0 = np.random.randn()
    for k in range(10000):
        rand_index = randint(0, len(X_train) - 1)
        x = X_train[rand_index]
        y = y_train[rand_index]
        step = 0.01
        w -= step * y * x * der_loss(x, y)
        w0 -= step * y * der_loss(x, y)
```

Онлайн-обучение

```
from random import randint
import numpy as np

def loss(x, y):
    return max([0, 1 - y * f(x)])

def der_loss(x, y):
    return -1.0 if 1 - y * f(x) > 0 else 0.0

def fit(X_train, y_train):
    w = np.random.randn(X_train.shape[1])
    w0 = np.random.randn()
    for k in range(10000):
        rand_index = randint(0, len(X_train) - 1)
        x = X_train[rand_index]
        y = y_train[rand_index]
        step = 0.01
        w -= step * y * x * der_loss(x, y)
        w0 -= step * y * der_loss(x, y)
```

Вместо взятия случайного объекта достаточно брать новый объект из выборки (если они приходят в случайном порядке)

Онлайн-обучение

```
from random import randint
import numpy as np

def loss(x, y):
    return max([0, 1 - y * f(x)])

def der_loss(x, y):
    return -1.0 if 1 - y * f(x) > 0 else 0.0

def fit(X_train, y_train):
    w = np.random.randn(X_train.shape[1])
    w0 = np.random.randn()
    for k in range(10000):
        rand_index = randint(0, len(X_train) - 1)
        x = X_train[rand_index]
        y = y_train[rand_index]
        step = 0.01
        w -= step * y * x * der_loss(x, y)
        w0 -= step * y * der_loss(x, y)
```

Вместо взятия случайного объекта достаточно брать новый объект из выборки (если они приходят в случайном порядке)

Также можно быстро дообучить модель на новом батче объектов



Работа с большими выборками

Идея: Использовать онлайн-обучение, чтобы единовременно держать в памяти только один объект (или сколько в нее помещается)



Работа с большими выборками

Идея: Использовать онлайн-обучение, чтобы единовременно держать в памяти только один объект (или сколько в нее помещается)

Проблема: что делать, если и один объект не помещается?



Работа с большими выборками

Идея: Использовать онлайн-обучение, чтобы единовременно держать в памяти только один объект (или сколько в нее помещается)

Проблема: что делать, если и один объект не помещается?

Решение: возможно и не нужно, чтобы помещался, если признаки разреженные и многие из них - редкие



Hashing trick с разреженными признаками

Hashing trick:

Заводим 2^N числовых признаков, нумеруем их

Проходимся по объектам выборки:

Проходимся по разреженным признакам объекта:

Если у объекта есть разреженный признак X (например в тексте - какое-то определенное слово X), считаем $id = \text{hash}(X) \% 2^N$ и записываем +1 к числовому признаку с этим id



Hashing trick с разреженными признаками

Hashing trick:

Заводим 2^N числовых признаков, нумеруем их

Проходимся по объектам выборки:

Проходимся по разреженным признакам объекта:

Если у объекта есть разреженный признак X (например в тексте - какое-то определенное слово X), считаем $id = \text{hash}(X) \% 2^N$ и записываем +1 к числовому признаку с этим id

Что делаем с коллизиями? Ничего, это будет регуляризация



Упрощение сложной правиловой модели

Пример: вы обучили простую линейную модель для Named Entity Recognition, но т.к. выборка была не очень велика, она получилась слабовата

У другой команды есть сложная лингвистическая модель для той же задачи, которую они делали 5 лет, работает она хорошо, но очень долго



Упрощение сложной правиловой модели

Пример: вы обучили простую линейную модель для Named Entity Recognition, но т.к. выборка была не очень велика, она получилась слабовата

У другой команды есть сложная лингвистическая модель для той же задачи, которую они делали 5 лет, работает она хорошо, но очень долго

Как извлечь из этого пользу: расширить выборку их тяжелой моделью и обучить уже на огромной размеченной выборке линейный классификатор. Качество будет где-то между, а скорость работы хорошая

4. Устойчивость линейных моделей



Бутстреп (bootstrap)

Выборка:

№	X
1	3.4
2	2.9
3	3.7
N	3.1

Бутстреп

Выборка:

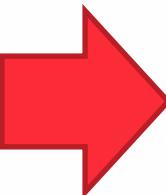
№	X
1	3.4
2	2.9
3	3.7
N	3.1

$$\mathbb{E} X = 3.3 \underline{+ ?}$$

Бутстреп

Выборка:

№	X
1	3.4
2	2.9
3	3.7
N	3.1



№	X
3	3.7
1	3.4
2	2.9
2	2.9

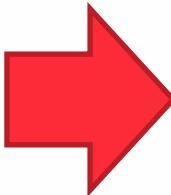
$$\mathbb{E} X = 3.3 \pm ?$$

Генерируем новую (искусственную) выборку, отобрав в нее объекты из исходной выборки по схеме выбора с возвращением

Бутстреп

Выборка:

№	X
1	3.4
2	2.9
3	3.7
N	3.1



№	X	№	X
3	3.7	1	3.4
1	3.4	3	3.7
2	2.9	2	2.9
2	2.9	M	3.0

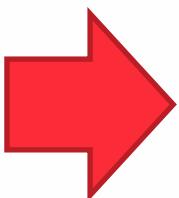
$$\mathbb{E} X = 3.3 \pm ?$$

Продолжаем
генерировать
такие выборки

Бутстреп

Выборка:

№	X
1	3.4
2	2.9
3	3.7
N	3.1



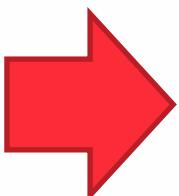
№	X	№	X	№	X
3	3.7	1	3.4	3	3.7
1	3.4	3	3.7	2	2.9
2	2.9	2	2.9
		1	3.4		
2	2.9	M	3.0	1	3.4

$$\mathbb{E} X = 3.3 \pm ?$$

Бутстреп

Выборка:

№	X
1	3.4
2	2.9
3	3.7
N	3.1



№	X	№	X	№	X
3	3.7	1	3.4	3	3.7
1	3.4	3	3.7	2	2.9
2	2.9	2	2.9
				1	3.4
2	2.9	M	3.0	1	3.4

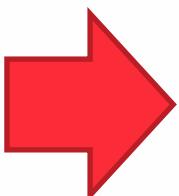
$$\mathbb{E} X = 3.3 \pm ?$$

$$\mathbb{E} X = 3.25 \quad \mathbb{E} X = 3.27 \quad \dots \quad \mathbb{E} X = 3.39$$

Бутстреп

Выборка:

№	X
1	3.4
2	2.9
3	3.7
N	3.1



№	X	№	X	№	X
3	3.7	1	3.4	3	3.7
1	3.4	3	3.7	2	2.9
2	2.9	2	2.9
				1	3.4
2	2.9	M	3.0	1	3.4

$$\mathbb{E} X = 3.3 \pm ?$$

$$\mathbb{E} X = 3.25 \quad \mathbb{E} X = 3.27 \quad \dots \quad \mathbb{E} X = 3.39$$

$$\mathbb{E} X = 3.32 \pm 0.06$$

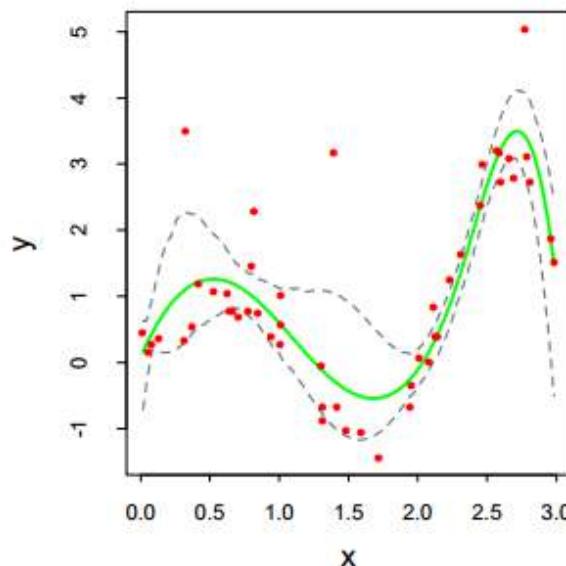
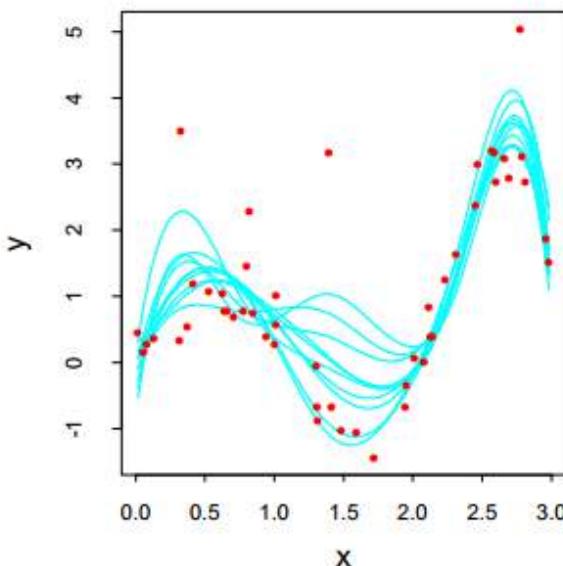
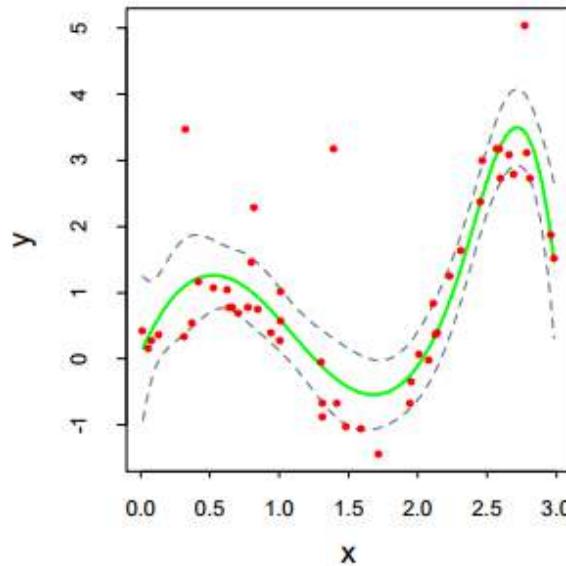
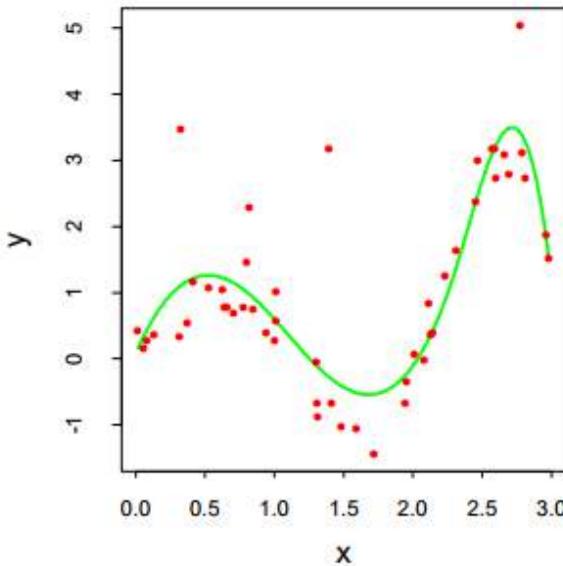


Бэггинг (bagging)

Bagging = Bootstrap aggregation

По схеме выбора с возвращением, генерируем M обучающих выборок такого же размера, обучаем на них модели и усредняем

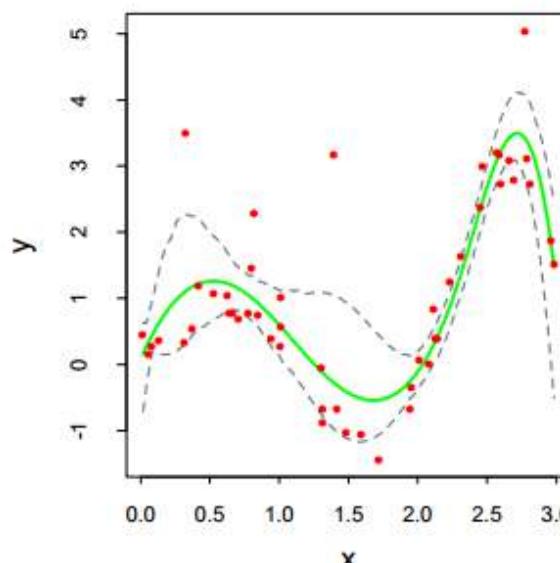
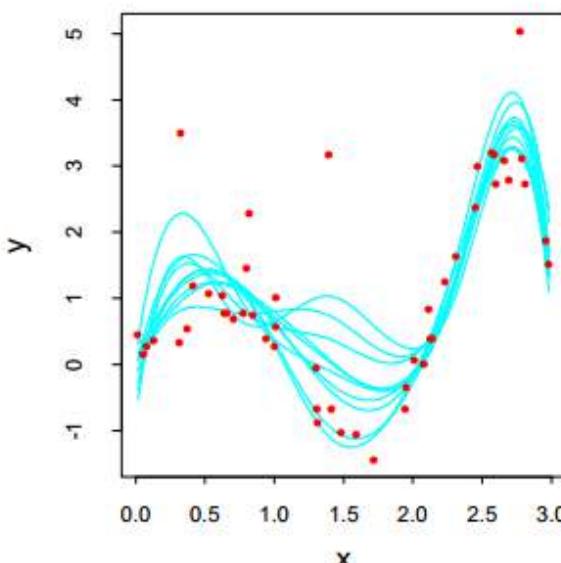
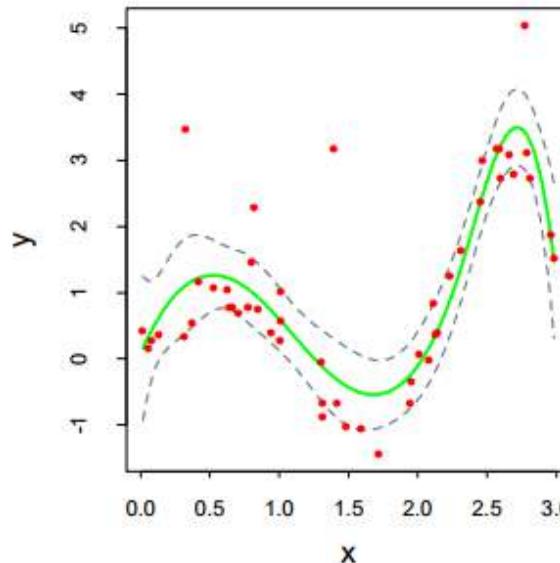
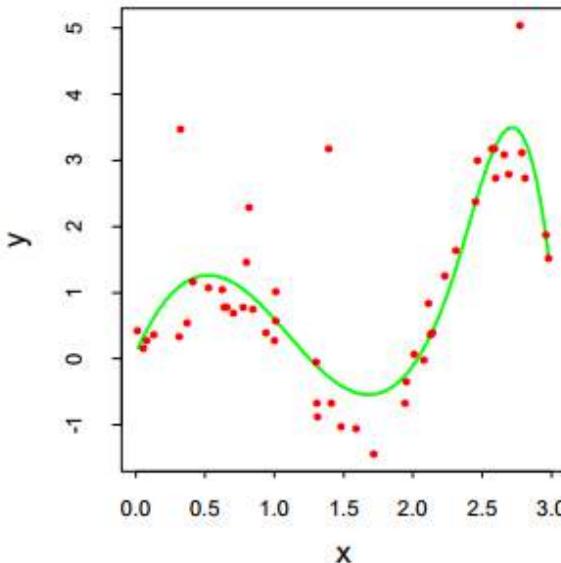
БЭГГИНГ



Вопросы:

1. Имеет ли смысл делать бэггинг над линейными классификаторами?

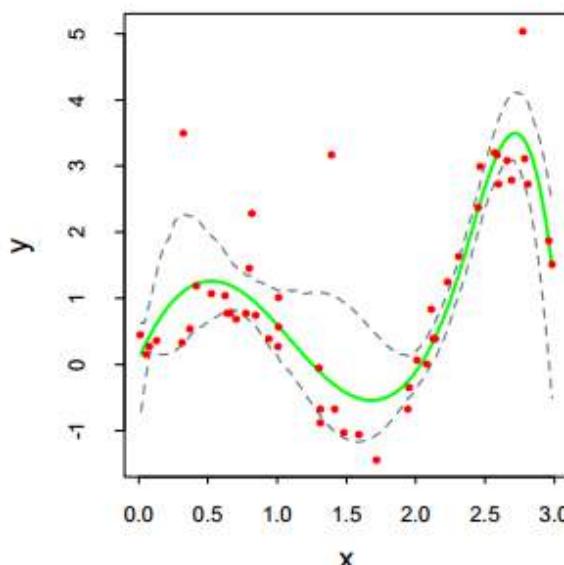
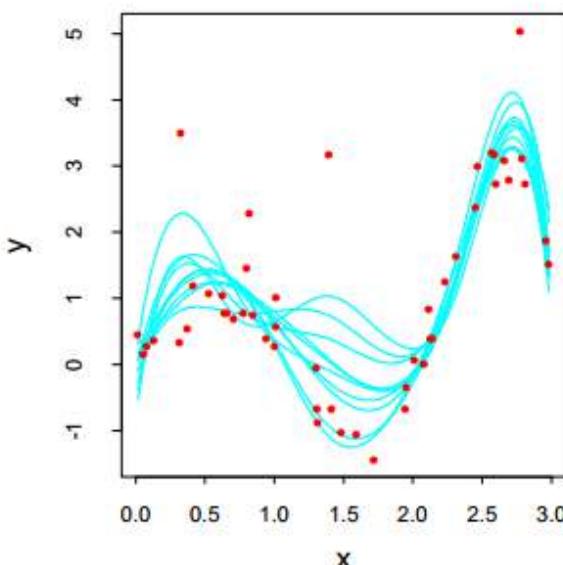
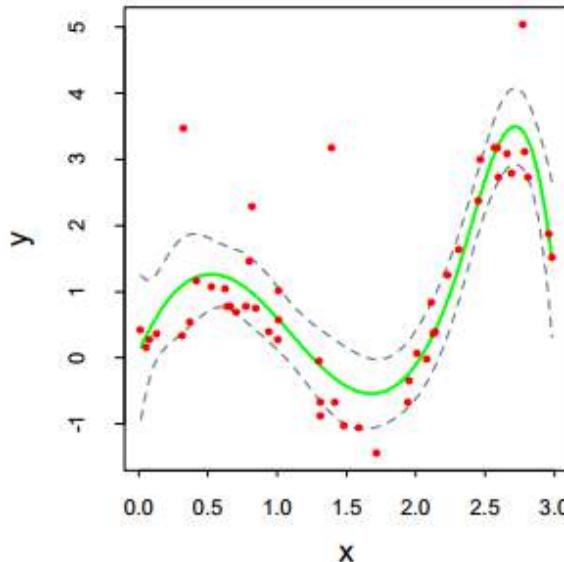
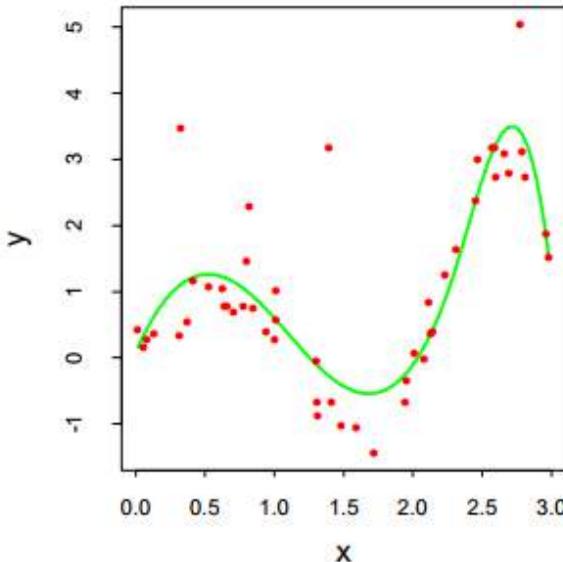
БЭГГИНГ



Вопросы:

1. Имеет ли смысл делать бэггинг над линейными классификаторами?
2. А регрессорами?

БЭГГИНГ



Вопросы:

1. Имеет ли смысл делать бэггинг над линейными классификаторами?
2. А регрессорами?

Для повышения устойчивости модели – имеет

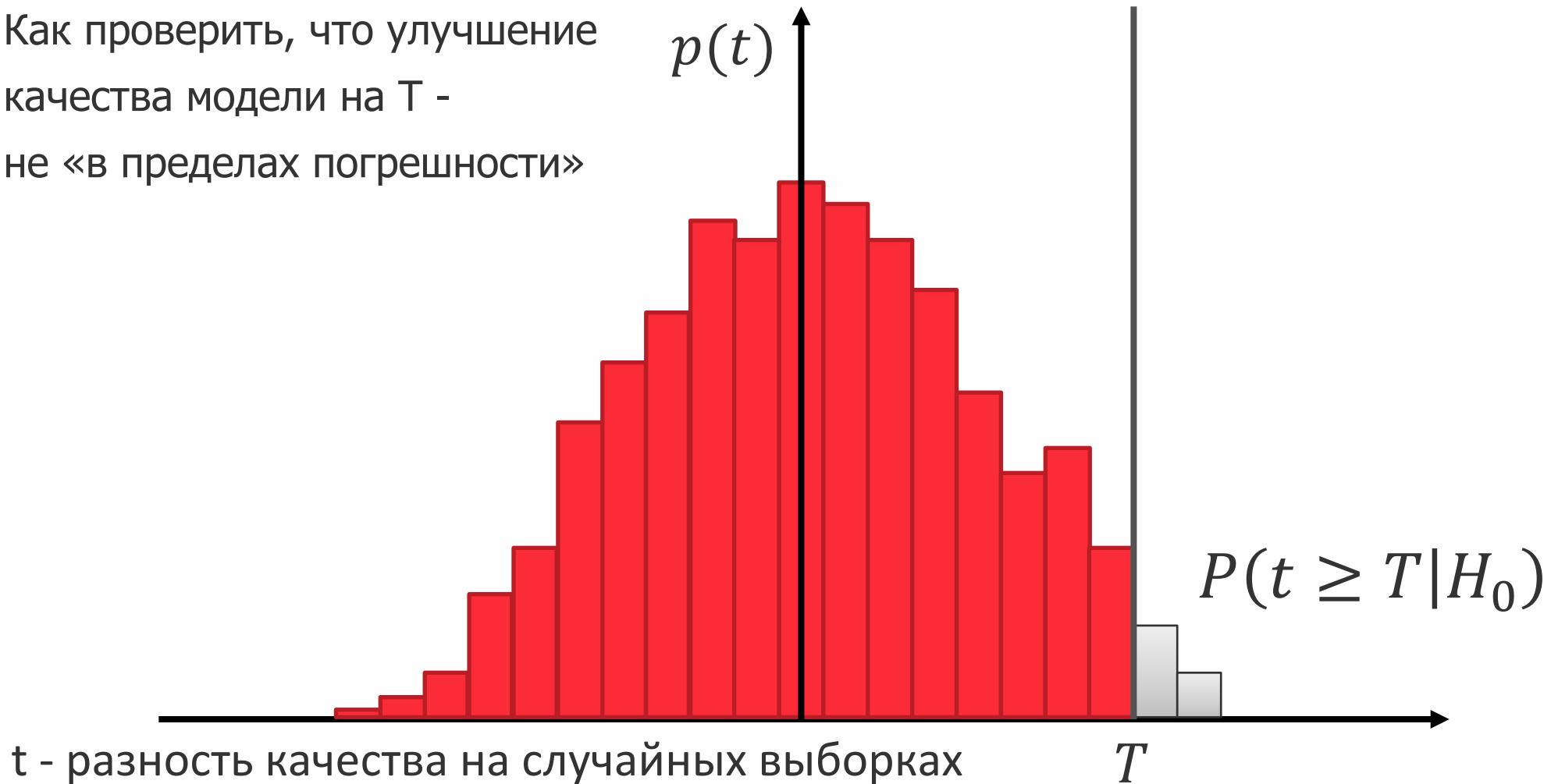


Бэггинг в sklearn.ensembles

- BaggingRegressor
- BaggingClassifier

Другое полезное применение бутстрепа

Как проверить, что улучшение
качества модели на T -
не «в пределах погрешности»

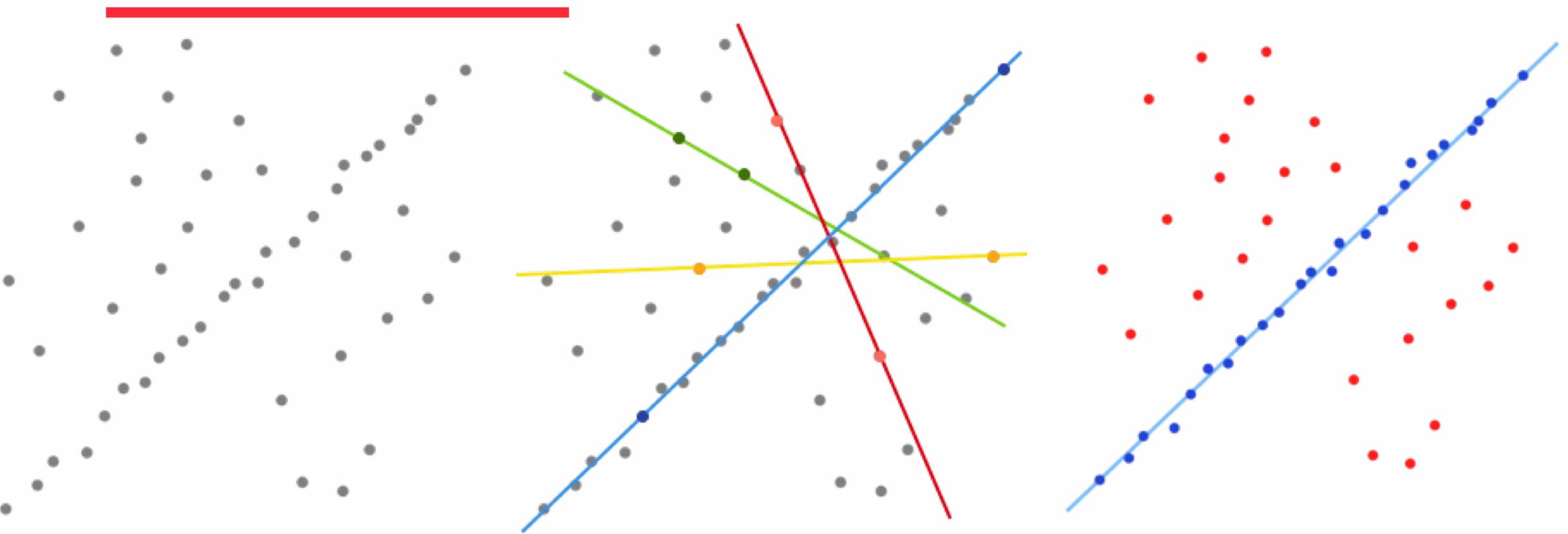




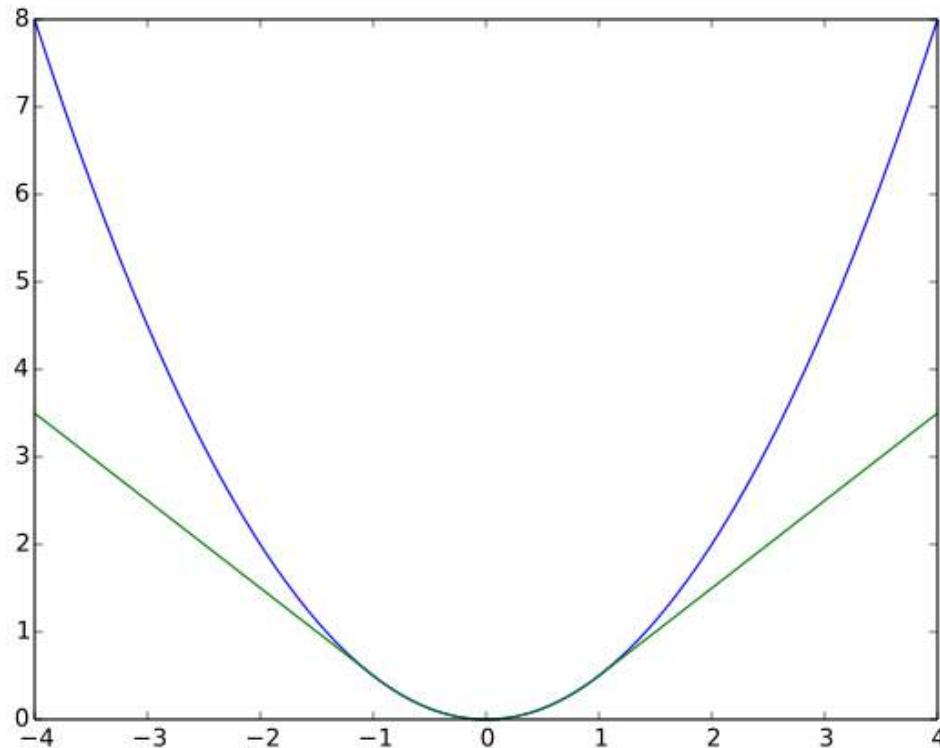
Робастные модели в sklearn.linear_model

- RANSACRegressor
- HuberRegressor
- Theil-Sen Regressor

RANSACRegressor



HuberRegressor





TheilSenRegressor

Изначально:

Угол наклона m – медиана $(y_j - y_i)/(x_j - x_i)$

Сдвиг – медиана $(y_i - mxi)$

Как обобщается на многомерный случай:

<http://home.olemiss.edu/~xdang/papers/MTSE.pdf>

5. Интерпретация других моделей

Интерпретация других моделей

“Why Should I Trust You?” Explaining the Predictions of Any Classifier

Marco Tulio Ribeiro
University of Washington
Seattle, WA 98105, USA
marcotcr@cs.uw.edu

Sameer Singh
University of Washington
Seattle, WA 98105, USA
sameer@cs.uw.edu

Carlos Guestrin
University of Washington
Seattle, WA 98105, USA
guestrin@cs.uw.edu

<https://arxiv.org/pdf/1602.04938.pdf>



Проблема и идея интерпретации

1. Не все модели машинного обучения интерпретируемы, т.е. позволяют ответить на вопрос «почему модель **на этом примере** ответила **именно так**»



Проблема и идея интерпретации

1. Не все модели машинного обучения интерпретируемы, т.е. позволяют ответить на вопрос «почему модель **на этом примере** ответила **именно так**»
2. Не всегда можно доверять «черному ящику» (вопросы медицины, финансовые вопросы и т.д.)



Проблема и идея интерпретации

1. Не все модели машинного обучения интерпретируемы, т.е. позволяют ответить на вопрос «почему модель **на этом примере** ответила **именно так**»
2. Не всегда можно доверять «черному ящику» (вопросы медицины, финансовые вопросы и т.д.)
3. Линейные модели часто вполне удачно интерпретируются



Проблема и идея интерпретации

1. Не все модели машинного обучения интерпретируемы, т.е. позволяют ответить на вопрос «почему модель **на этом примере** ответила **именно так**»
2. Не всегда можно доверять «черному ящику» (вопросы медицины, финансовые вопросы и т.д.)
3. Линейные модели часто вполне удачно интерпретируются
4. А на разреженных признаках и с небольшим количеством ненулевых коэффициентов – особенно

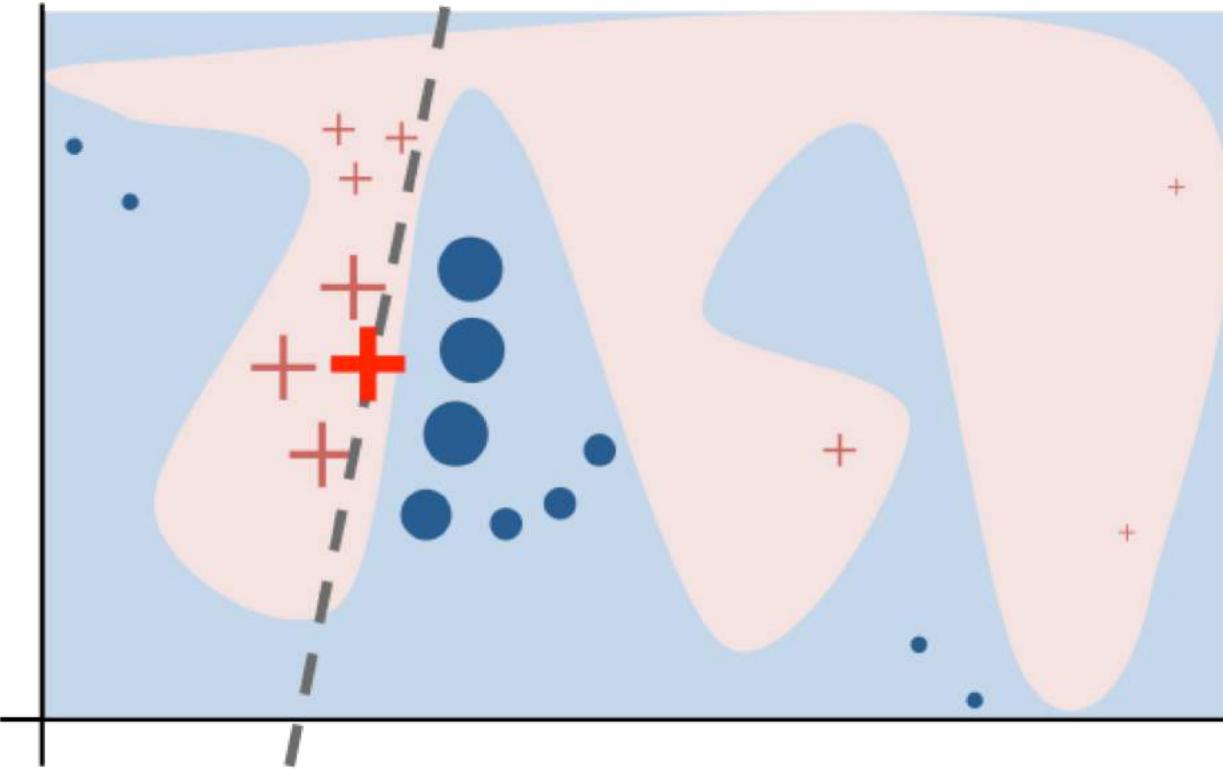


Проблема и идея интерпретации

1. Не все модели машинного обучения интерпретируемы, т.е. позволяют ответить на вопрос «почему модель **на этом примере** ответила **именно так**»
2. Не всегда можно доверять «черному ящику» (вопросы медицины, финансовые вопросы и т.д.)
3. Линейные модели часто вполне удачно интерпретируются
4. А на разреженных признаках и с небольшим количеством ненулевых коэффициентов – особенно
5. Попробуем локально приближать нашу модель линейной

Идея LIME

Local
Interpretable
Model-agnostic
Explanations

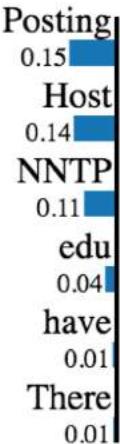


LIME на текстах

Prediction probabilities



atheism



christian

Text with highlighted words

From: johnchad@triton.unm.edu (jchadwic)

Subject: Another request for Darwin Fish

Organization: University of New Mexico, Albuquerque

Lines: 11

NNTP-Posting-Host: triton.unm.edu

Hello Gang,

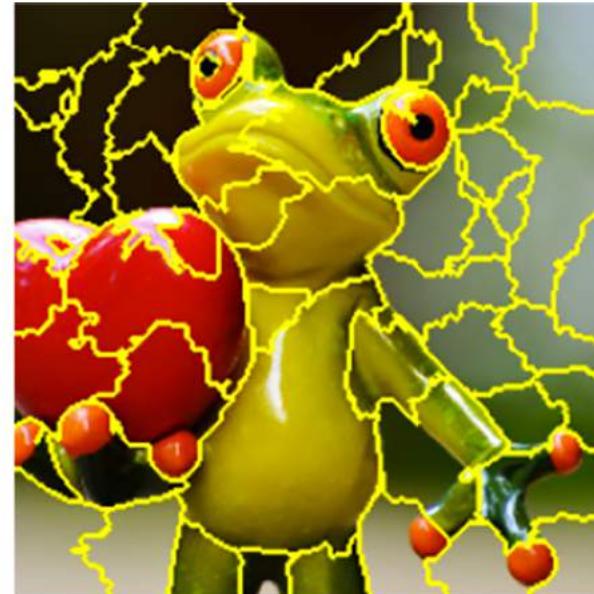
There have been some notes recently asking where to obtain the DARWIN fish.

This is the same question I have and I have not seen an answer on the net. If anyone has a contact please post on the net or email me.

LIME на изображениях

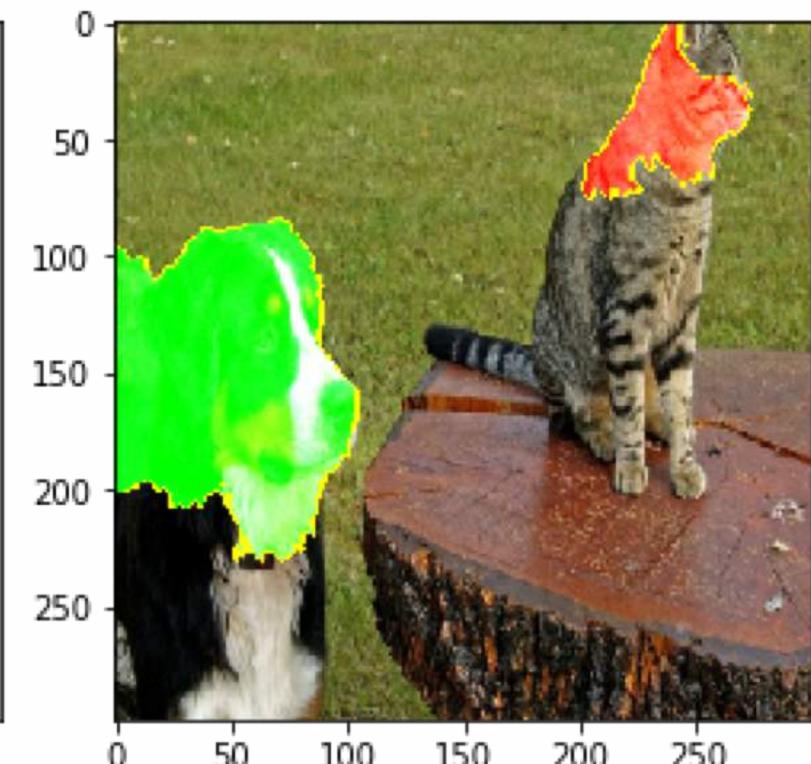
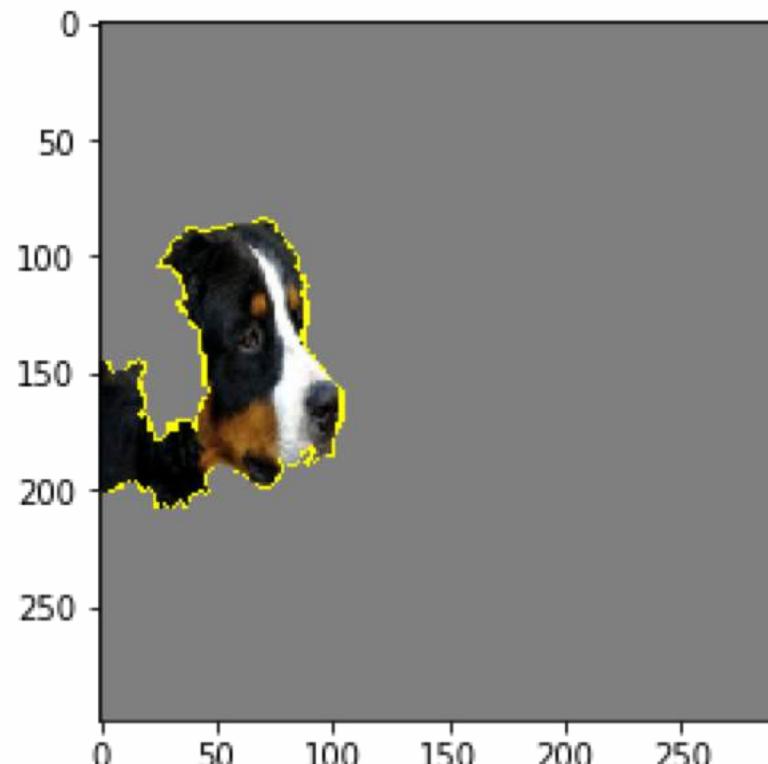
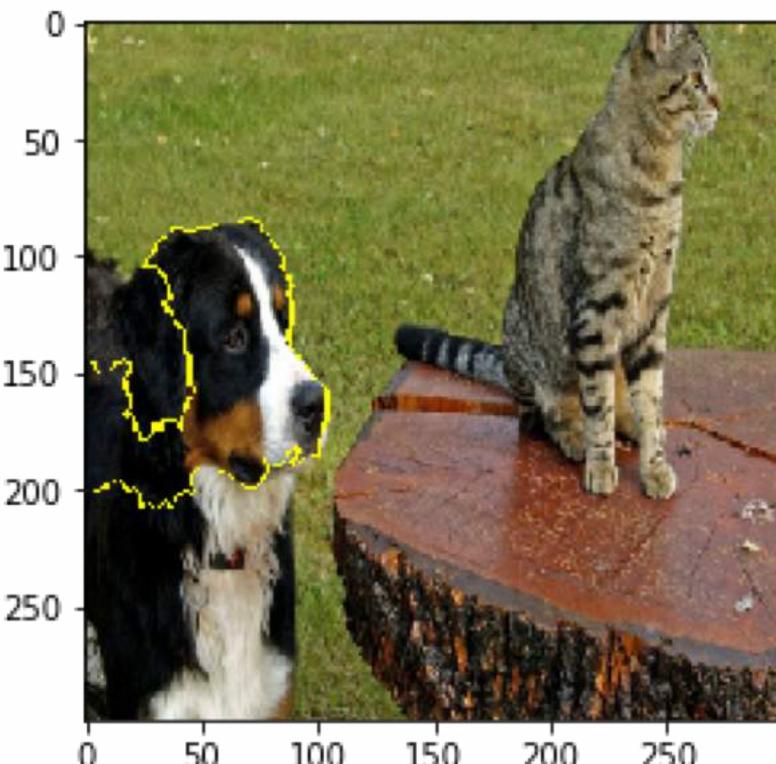


Original Image



Interpretable
Components

LIME на изображениях





Обсудили сегодня

1. Напоминание: как работают линейные модели
2. Особенности обучения линейных моделей
3. Use-cases
4. Устойчивость линейных моделей
5. Интерпретация других моделей



Что не вошло в лекцию

1. Generalized linear models
2. General linear model
3. Matrix factorization как матричное обобщение линейных моделей (когда известные нам ответы – ячейки матрицы, например рейтинги, поставленные пользователями сервиса доступным в нем товарам)
4. Factorization Machines – как развитие идеи использовать matrix factorization для обучения с учителем



Что будет через неделю

Решающие деревья и ансамбли

Как работают решающие деревья и ансамбли деревьев из популярных библиотек (XGBoost, LightGBM, CatBoost). Построение ансамблей на практике в соревнованиях и в продакшне. Блендинг и стекинг алгоритмов.



Что будет через неделю

Решающие деревья и ансамбли

Как работают решающие деревья и ансамбли деревьев из популярных библиотек (XGBoost, LightGBM, CatBoost). Построение ансамблей на практике в соревнованиях и в продакшне. Блендинг и стекинг алгоритмов.

Важно: занятие разово переносится на пятницу
25 октября (также в 19:00)