Лекция 1. Введение. Задачи, подходы и возможности

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

План курса

- Введение. История вопроса и современное состояние проблемы

План курса

- Введение. История вопроса и современное состояние проблемы
- Общие методы и особенности работы с данными

План курса

- Введение. История вопроса и современное состояние проблемы
- Общие методы и особенности работы с данными
- Обзор алгоритмов машинного обучения

План курса

- Введение. История вопроса и современное состояние проблемы
- Общие методы и особенности работы с данными
- Обзор алгоритмов машинного обучения
- Доклады аспирантов

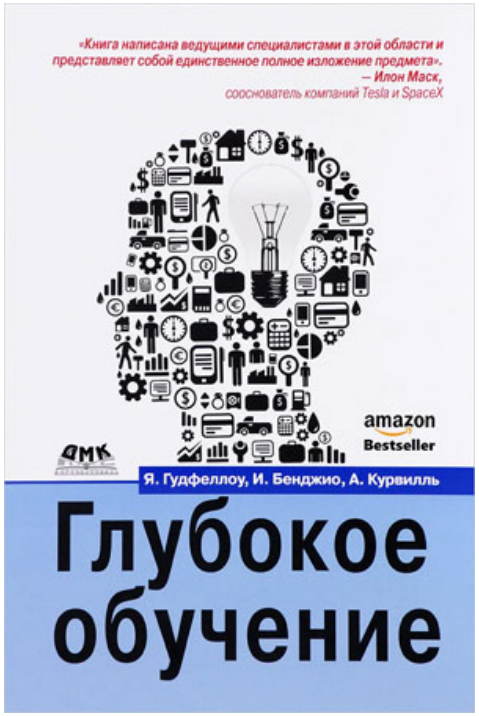
Литература и другие рекомендуемые источники

https://www.ozon.ru/product/glubokoe-obuchenie-na-python-145615583/

https://www.ozon.ru/product/obuchenie-s-podkrepleniem-28343183/

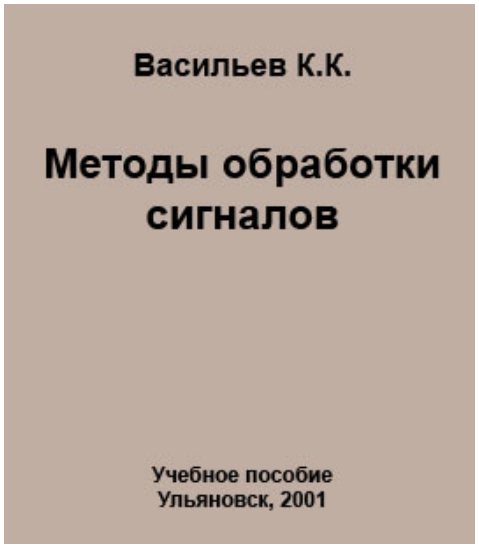
https://www.ozon.ru/product/neyronnye-seti-polnyy-kurs-135794246/

Литература и другие рекомендуемые источники



https://www.ozon.ru/product/glubokoe-obuchenie-141754606/

https://www.ozon.ru/product/vvedenie-v-matematicheskuyu-statistiku-statistika-znaet-vse-140902512/



Литература и другие рекомендуемые источники

https://ods.ai/

kaggle

https://kaggle.com/

DataRing.ru

- https://archive.ics.uci.edu/ml/
- https://www.youtube.com/c/MachineLearningPhystech
- https://www.youtube.com/c/ritvikmath
- https://www.youtube.com/c/DigitalSreeni
- https://www.youtube.com/user/SpartacanUsuals
- https://www.youtube.com/c/CompscicenterRu

Что мы понимаем под ИИ и МО?

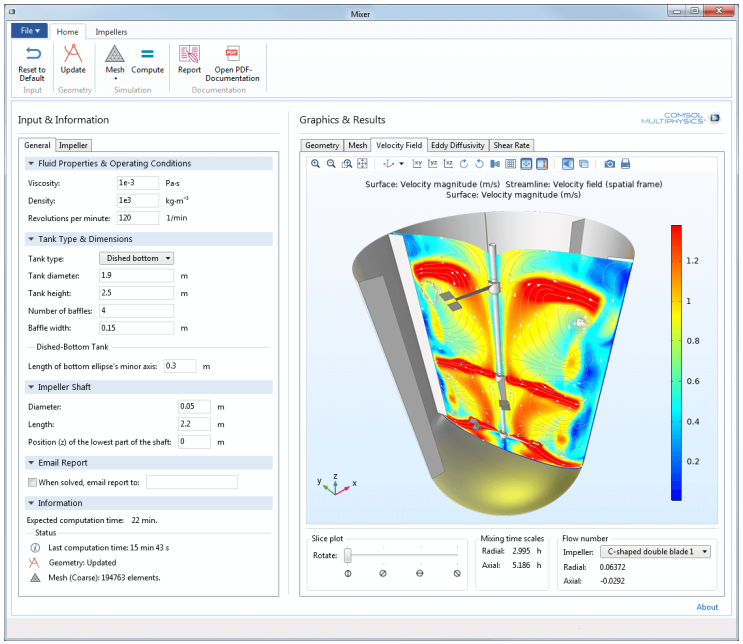
Определение 1.1

Искусственный интеллект (ИИ; англ. artificial intelligence, AI) — *свойство интеллектуальных систем выполнять творческие функции, которые традиционно считаются прерогативой человека; наука и технология создания интеллектуальных машин, особенно интеллектуальных компьютерных программ.*

Определение 1.2

Машинное обучение (англ. machine learning, ML) — *класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение за счёт применения решений множества сходных задач.*

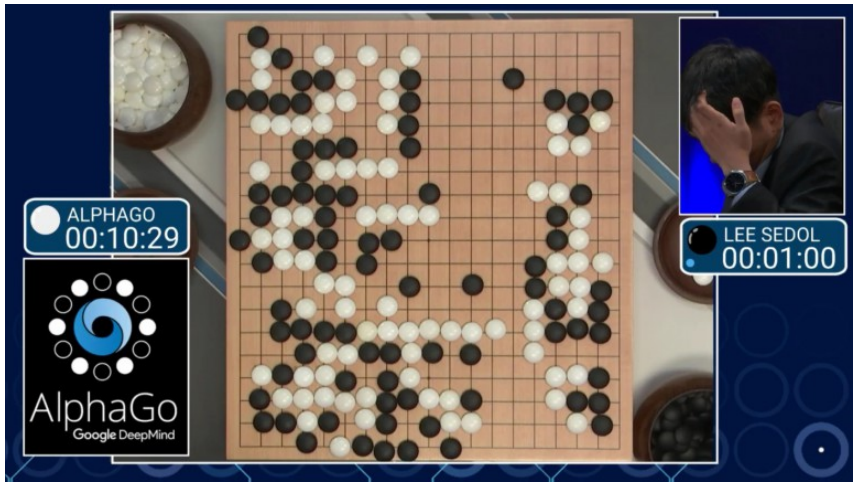
“Формальные” и “неформальные” задачи



VS

Успехи машинного обучения в играх

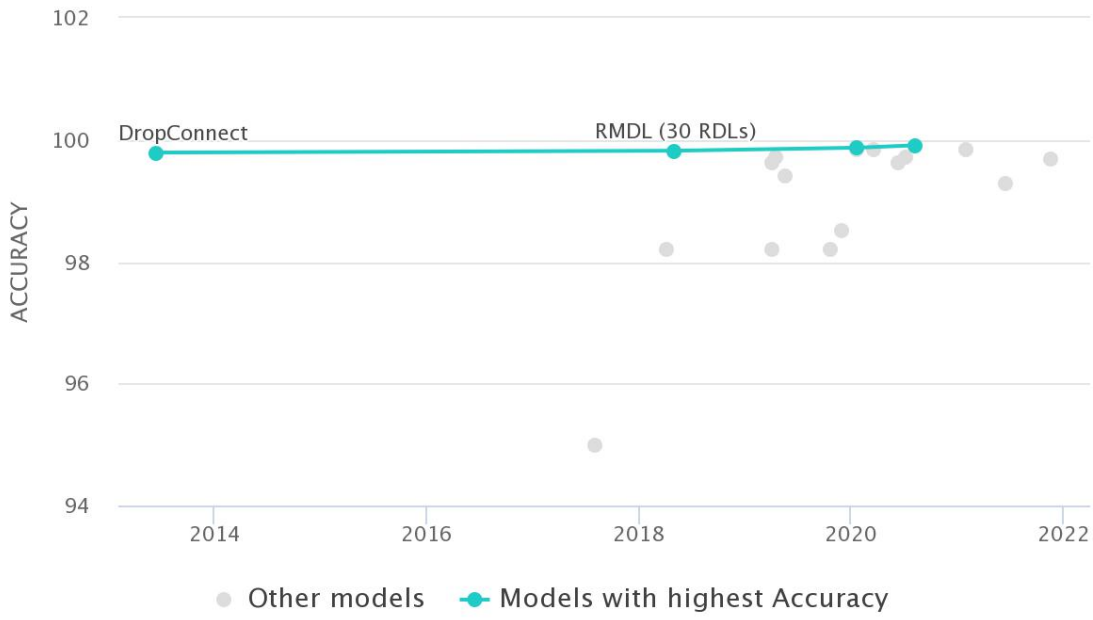
11 мая 1997 года Deep Blue выиграл матч у чемпиона мира по шахматам Гарри Каспарова



AlphaGo победил со счетом 4:1 в матче с Ли Седолем (9-й дан), который прошел с 9 по 15 марта 2016 года.

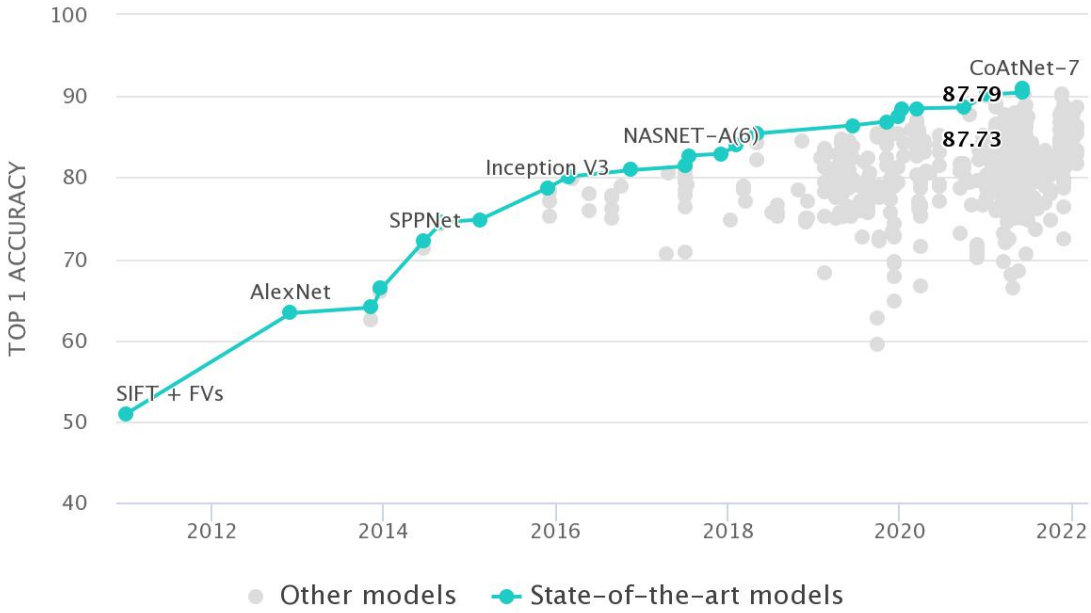
Успехи машинного обучения в распознавании изображений

Успехи машинного обучения в распознавании рукописных цифр (база данных MNIST)



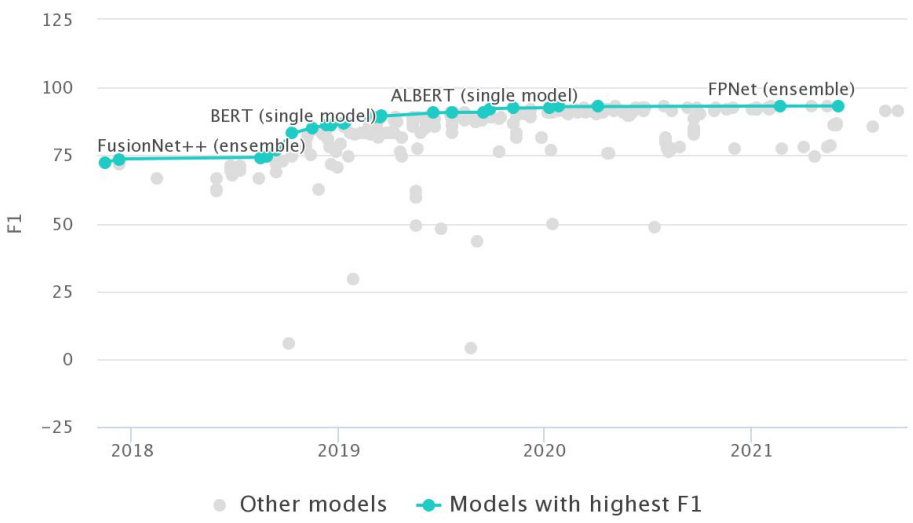
https://paperswithcode.com/sota/image-classification-on-mnist?metric=Accuracy

Успехи машинного обучения в распознавании изображений (база данных ImageNet)



https://paperswithcode.com/sota/image-classification-on-imagenet

Успехи машинного обучения в области понимания текстов (база данных SQuAD)



A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself. A natural number greater than 1 that is not a prime number is called a composite number. For example, 5 is prime because 1 and 5 are its only positive integer factors, whereas 6 is composite because it has the divisors 2 and 3 in addition to 1 and 6. The fundamental theorem of arithmetic establishes the central role of primes in number theory: any integer greater than 1 can be expressed as a product of primes that is unique up to ordering. The uniqueness in this theorem requires excluding 1 as a prime because one can include arbitrarily many instances of 1 in any factorization, e.g., 3, $1 \cdot 3$, $1 \cdot 1 \cdot 3$, etc. are all valid factorizations of 3.

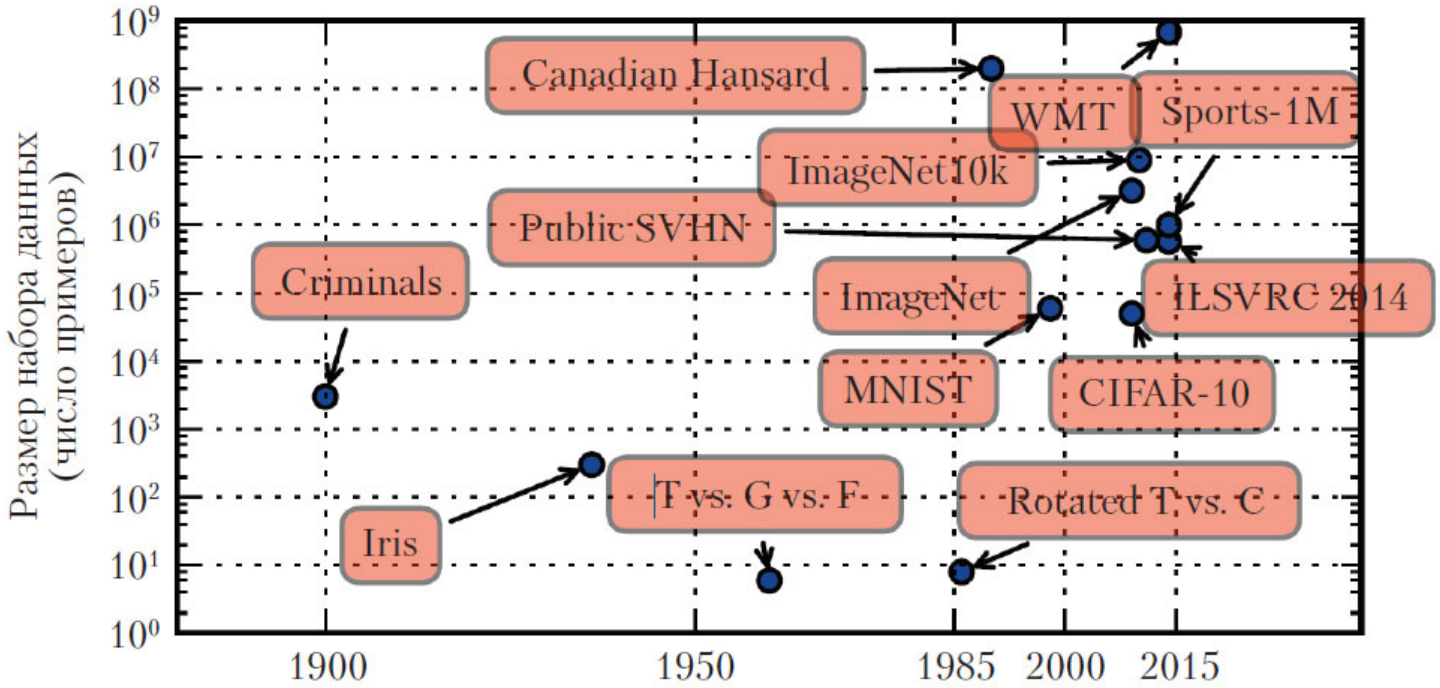
What is the only divisor besides 1 that a prime number can have?
Ground Truth Answers: itself | itself | itself | itself | itself

What are numbers greater than 1 that can be divided by 3 or more numbers called?
Ground Truth Answers: composite number | composite number | composite number | primes

What theorem defines the main role of primes in number theory?
Ground Truth Answers: The fundamental theorem of arithmetic | fundamental theorem of arithmetic | arithmetic | fundamental theorem of arithmetic | fundamental theorem of arithmetic

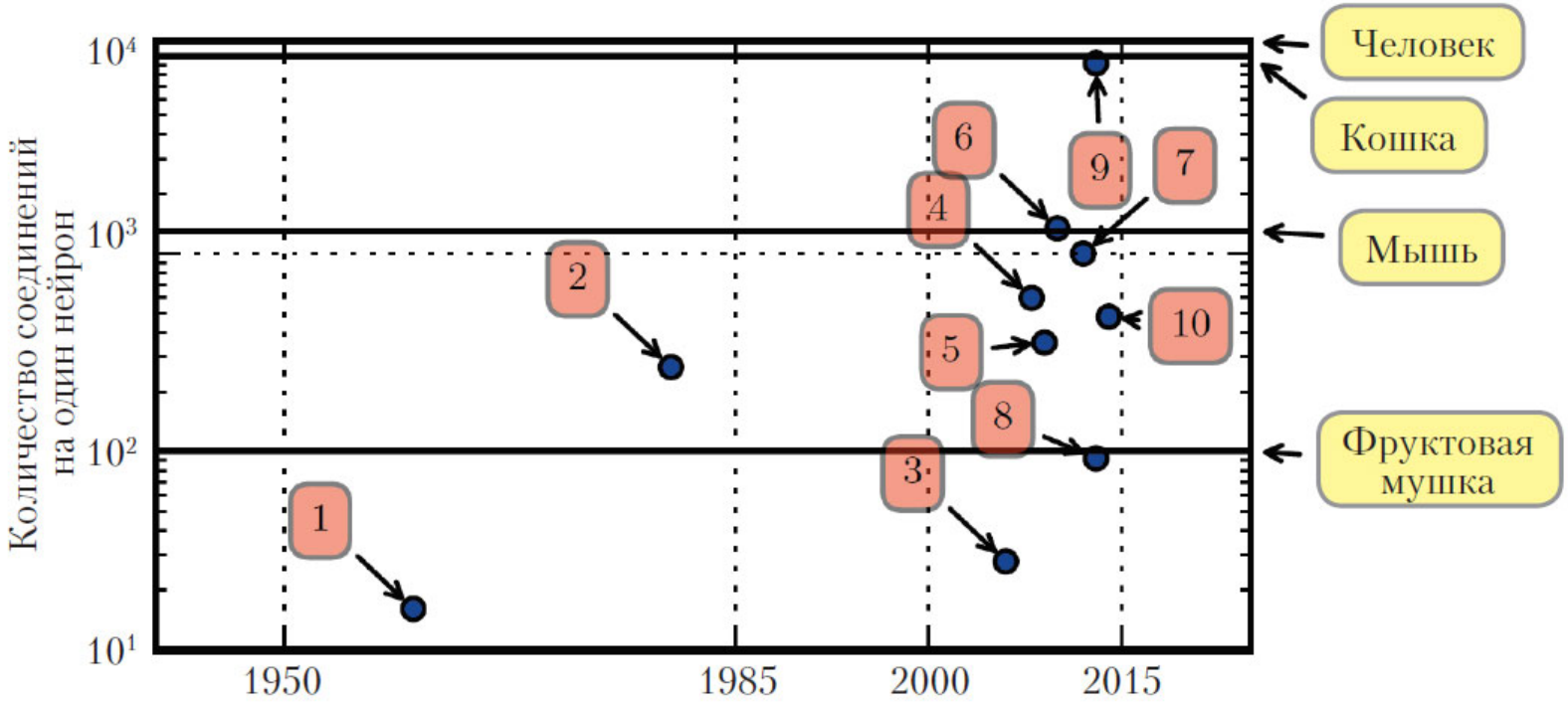
`https://paperswithcode.com/sota/question-answering-on-squad20?metric=F1`

Увеличение размера набора данных со временем



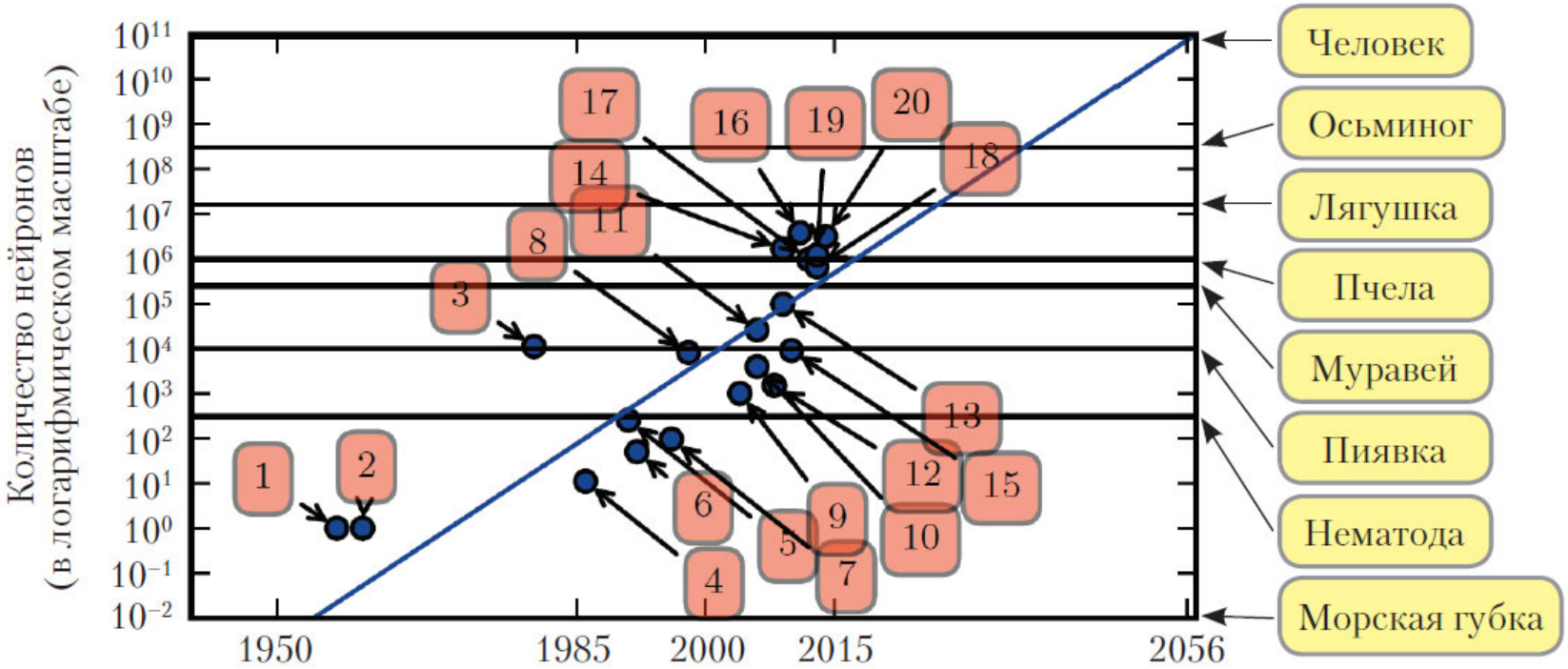
Я. Гудфеллоу, И. Бенджио, А. Курвилль “Глубокое обучение”

Рост количества соединений в расчете на один нейрон со временем



Я. Гудфеллоу, И. Бенджио, А. Курвилль “Глубокое обучение”

Рост размера нейронной сети со временем



Я. Гудфеллоу, И. Бенджио, А. Курвилль “Глубокое обучение”

Постановка задачи машинного обучения

Определение 1.3

Признак *— отображение* $f : \mathbb{X} \to \mathbb{D}_f$*, где* $\mathbb{X}$ *– множество описаний объектов;* $\mathbb{D}_f$ *– множество допустимых значений признака. Если заданы признаки* $f_1, f_2, \ldots, f_n$*, то вектор* $\vec{x} = \{f_1(x), f_2(x) \ldots, f_n(x)\}$ *называется признаковым описанием объекта* $\vec{x} \in \mathbb{X}$*. Множество* $\mathbb{D}_{f_1} \times \mathbb{D}_{f_2} \times \cdots \times \mathbb{D}_{f_n}$ *называют признаковым пространством.*

Постановка задачи машинного обучения

Определение 1.3

Признак *— отображение* $f : \mathbb{X} \to \mathbb{D}_f$*, где* $\mathbb{X}$ *– множество описаний объектов;* $\mathbb{D}_f$ *– множество допустимых значений признака. Если заданы признаки* $f_1, f_2, \ldots, f_n$*, то вектор* $\vec{x} = \{f_1(x), f_2(x) \ldots, f_n(x)\}$ *называется признаковым описанием объекта* $\vec{x} \in \mathbb{X}$*. Множество* $\mathbb{D}_{f_1} \times \mathbb{D}_{f_2} \times \cdots \times \mathbb{D}_{f_n}$ *называют признаковым пространством.*

- бинарный признак: $\mathbb{D}_f = \{0,1\}$;
- номинальный признак: $\mathbb{D}_f$ – конечное множество;
- порядковый признак: $\mathbb{D}_f$ – конечное упорядоченное множество;
- количественный признак: $\mathbb{D}_f$ – множество действительных чисел.

Постановка задачи машинного обучения

Определение 1.4

Целевой признак *— один или несколько признаков объекта y, которые представляют практический интерес с точки зрения предсказаний при решении задачи машинного обучения*

Постановка задачи машинного обучения

Определение 1.4

Целевой признак *— один или несколько признаков объекта y, которые представляют практический интерес с точки зрения предсказаний при решении задачи машинного обучения*

Задача машинного обучения – располагая выборкой $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ объектов $\vec{x}_i \in \mathbb{X}_{\text{train}}$, для которых известны значения y_i, построить алгоритм $a(\vec{x})$, максимально хорошо аппроксимирующий значения y для всех $\vec{x} \in \mathbb{X}$.

“Наивная” классификация задач по целевому признаку

- бинарный признак – классификация на 2 класса;

“Наивная” классификация задач по целевому признаку

- бинарный признак – классификация на 2 класса;
- номинальный признак – классификация на N классов;

“Наивная” классификация задач по целевому признаку

- бинарный признак – классификация на 2 класса;
- номинальный признак – классификация на N классов;
- порядковый признак – ранжирование объектов;

“Наивная” классификация задач по целевому признаку

- бинарный признак – классификация на 2 класса;
- номинальный признак – классификация на N классов;
- порядковый признак – ранжирование объектов;
- количественный признак – регрессия

“Наивная” классификация задач по целевому признаку

- бинарный признак – классификация на 2 класса;
- номинальный признак – классификация на N классов;
- порядковый признак – ранжирование объектов;
- количественный признак – регрессия

В некоторых случаях целевой признак не определен – это **задачи обучения без учителя**.

Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

- Центральная симметрия

Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

- Центральная симметрия
- Симметрия по горизонтали

Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

- Центральная симметрия
- Симметрия по горизонтали
- Симметрия по вертикали

Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

- Центральная симметрия
- Симметрия по горизонтали
- Симметрия по вертикали
- Число областей

Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

- Центральная симметрия
- Симметрия по горизонтали
- Симметрия по вертикали
- Число областей
- “Жирность” символа

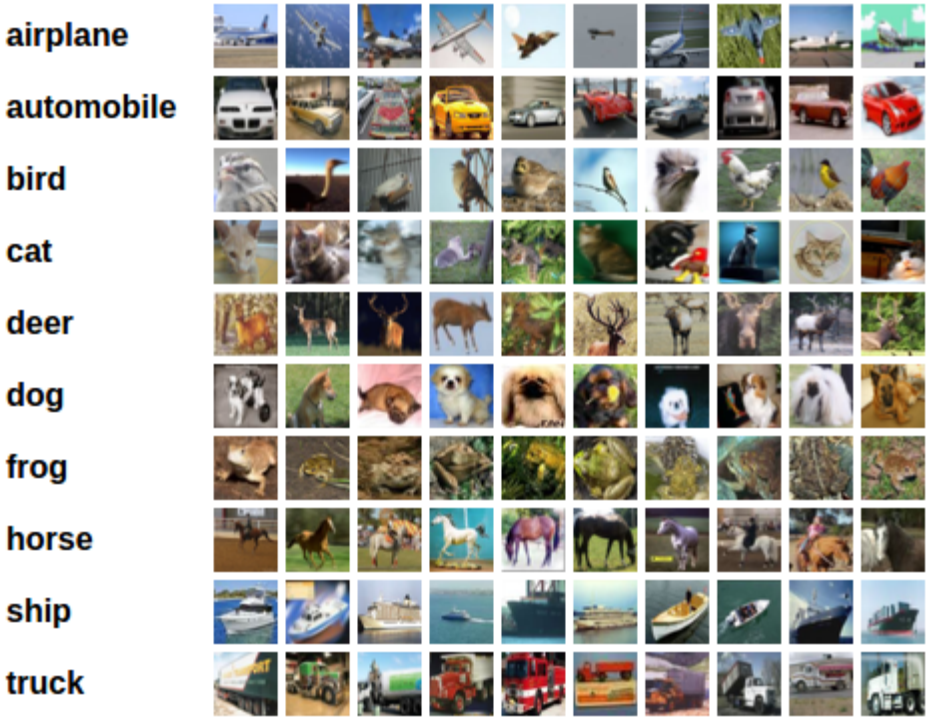
Распознавание рукописных цифр (база данных MNIST)

Придумываем признаки!

- Центральная симметрия
- Симметрия по горизонтали
- Симметрия по вертикали
- Число областей
- “Жирность” символа
- Каждый пиксель изображения

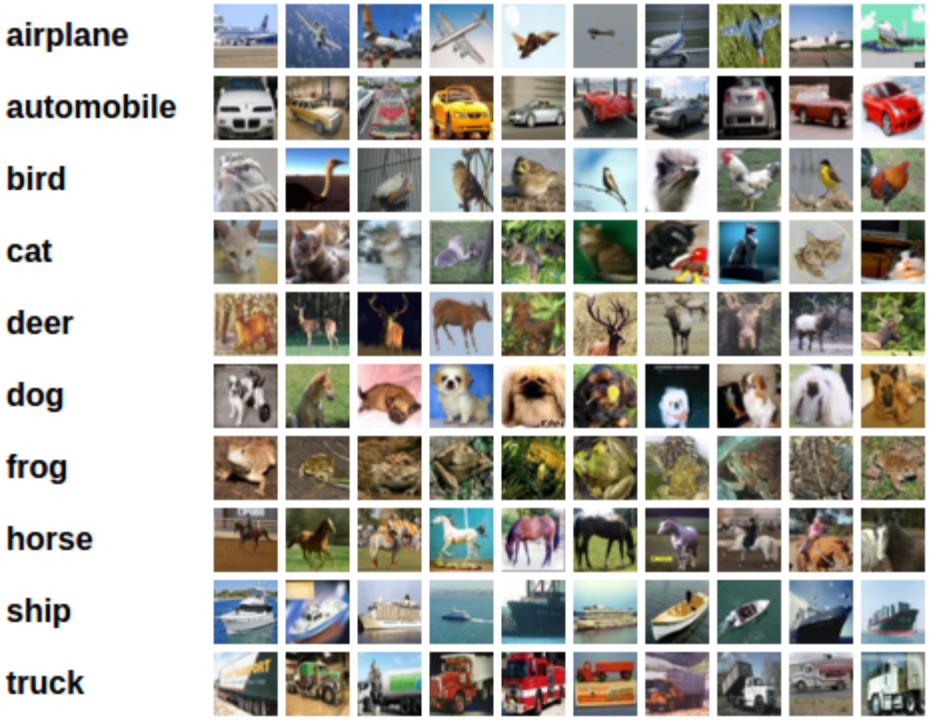
Распознавание изображений (база данных ImageNet)

Придумываем признаки!



Распознавание изображений (база данных ImageNet)

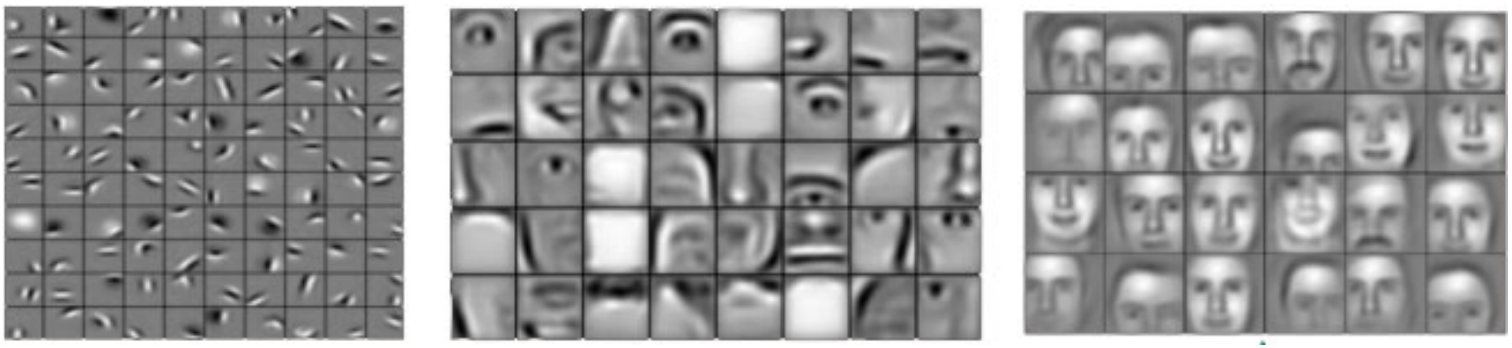
Придумываем признаки!



- Сложно что-то придумать!

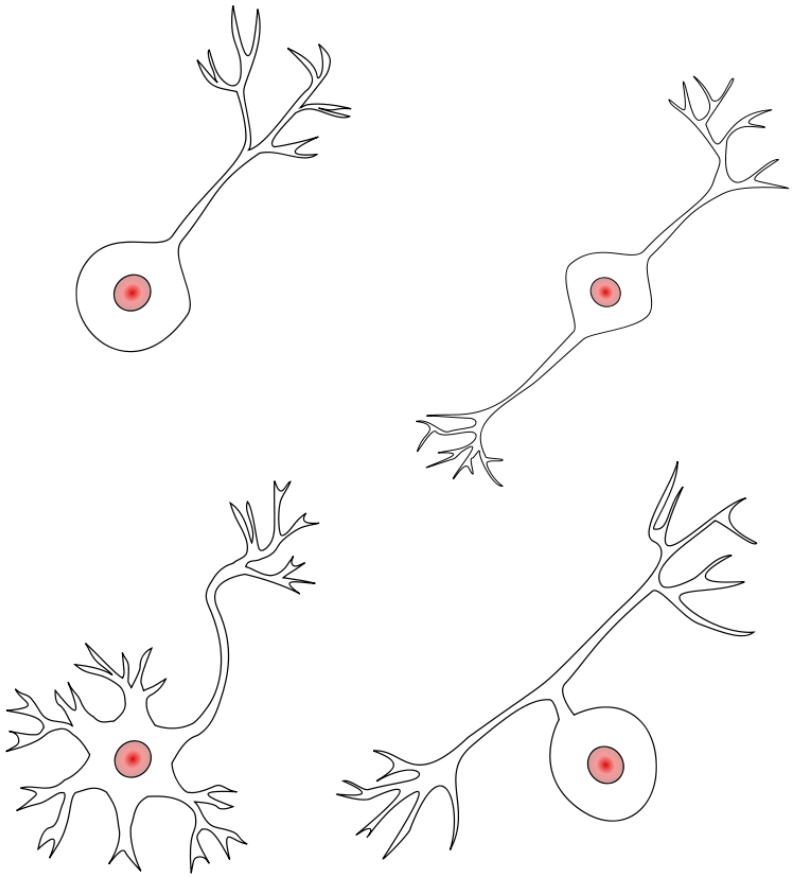
Глубокое машинное обучение – построение признаков

Детали изображения $\longrightarrow$ Фрагменты $\longrightarrow$ Объекты

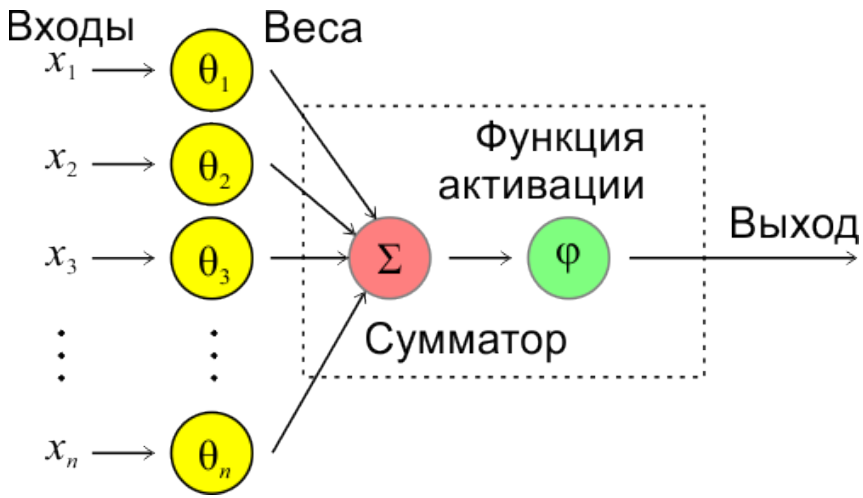


Модель нейрона Мак-Каллока–Питтса

Различные типы нейронов



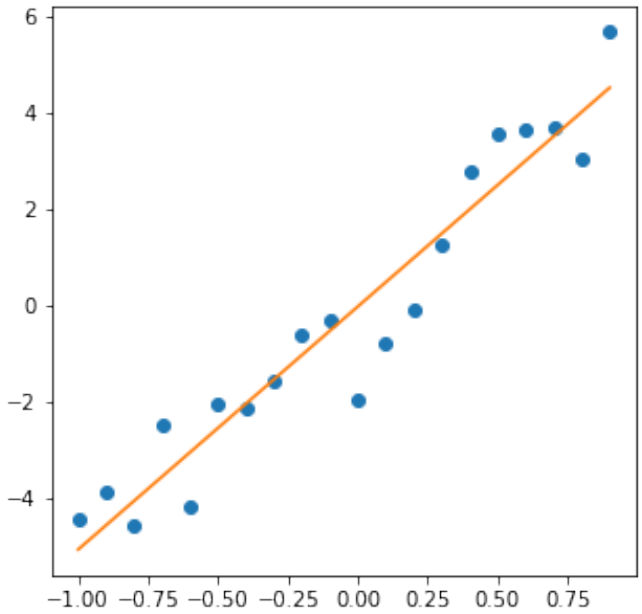
https://en.wikipedia.org/wiki/Neuron



$$a(\vec{x}) = \varphi\left(\sum x_i w_i\right)$$

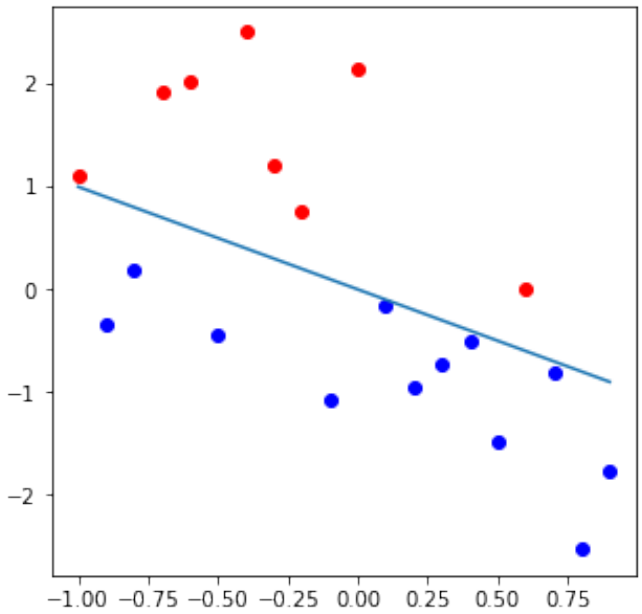
Линейная регрессия и классификация

Линейная регрессия



$$y = \sum x_i w_i$$

Линейная классификация



$$y = \operatorname{sgn}\left(\sum x_i w_i\right)$$

Для оптимизации весов w_i необходимо
указать **функционал качества** $\mathcal{L}(a, \vec{x}, y)$

Практикум. Линейная регрессия и классификация

Jupyter notebook “Визуализация IRIS”:
`https://colab.research.google.com/drive/1IJjWux4vyjo6-0SckpHRwCOUCGy2q-vy`

Jupyter notebook “Аппроксимация зависимостей”:
`https://colab.research.google.com/drive/1Rb0eJPBl72LkupbO-ZWUCaUbb9Y46MqM`

Построение моделей

Определение 1.5

Модель *— совокупность имеющихся или предполагаемых правил и параметров, которые определяют поведение объекта или характеристики исследуемых сигналов. Она позволяет свести описание сложной системы к (относительно) небольшому числу параметров и обладает предсказательной силой*

Построение моделей

Определение 1.5

Модель *— совокупность имеющихся или предполагаемых правил и параметров, которые определяют поведение объекта или характеристики исследуемых сигналов. Она позволяет свести описание сложной системы к (относительно) небольшому числу параметров и обладает предсказательной силой*

Некоторые типичные примеры моделей

- Полиномиальная модель $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots$

Построение моделей

Определение 1.5

Модель *— совокупность имеющихся или предполагаемых правил и параметров, которые определяют поведение объекта или характеристики исследуемых сигналов. Она позволяет свести описание сложной системы к (относительно) небольшому числу параметров и обладает предсказательной силой*

Некоторые типичные примеры моделей

- Полиномиальная модель $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots$
- Циклическая модель $f(x) = A_1 \sin(\omega_1 x) + A_2 \sin(\omega_2 x) + \dots$

Построение моделей

Определение 1.5

Модель *— совокупность имеющихся или предполагаемых правил и параметров, которые определяют поведение объекта или характеристики исследуемых сигналов. Она позволяет свести описание сложной системы к (относительно) небольшому числу параметров и обладает предсказательной силой*

Некоторые типичные примеры моделей

- Полиномиальная модель $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots$
- Циклическая модель $f(x) = A_1 \sin(\omega_1 x) + A_2 \sin(\omega_2 x) + \dots$
- Модель скользящего среднего $f_n = \sum_{m=1}^{M} \alpha_m g_{n-m} + \xi_n$

Построение моделей

Определение 1.5

Модель *— совокупность имеющихся или предполагаемых правил и параметров, которые определяют поведение объекта или характеристики исследуемых сигналов. Она позволяет свести описание сложной системы к (относительно) небольшому числу параметров и обладает предсказательной силой*

Некоторые типичные примеры моделей

- Полиномиальная модель $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots$
- Циклическая модель $f(x) = A_1 \sin(\omega_1 x) + A_2 \sin(\omega_2 x) + \dots$
- Модель скользящего среднего $f_n = \sum_{m=1}^{M} \alpha_m g_{n-m} + \xi_n$
- Авторегрессионная модель $f_n = \sum_{m=1}^{M} \alpha_m f_{n-m} + \xi_n$

Построение моделей

Определение 1.5

Модель *— совокупность имеющихся или предполагаемых правил и параметров, которые определяют поведение объекта или характеристики исследуемых сигналов. Она позволяет свести описание сложной системы к (относительно) небольшому числу параметров и обладает предсказательной силой*

Некоторые типичные примеры моделей

- Полиномиальная модель $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots$
- Циклическая модель $f(x) = A_1 \sin(\omega_1 x) + A_2 \sin(\omega_2 x) + \dots$
- Модель скользящего среднего $f_n = \sum\limits_{m=1}^{M} \alpha_m g_{n-m} + \xi_n$
- Авторегрессионная модель $f_n = \sum\limits_{m=1}^{M} \alpha_m f_{n-m} + \xi_n$
- Модель гауссовского процесса $N(\bar{x},\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\bar{x})^2}{2\sigma^2}}$

Прямые и обратные задачи

Определение 1.6

Прямая задача *— необходимо определить, как будет вести себя модель, все характеристики которой заданы*

Прямые и обратные задачи

Определение 1.6

Прямая задача *— необходимо определить, как будет вести себя модель, все характеристики которой заданы*

Определение 1.7

Обратная задача *— необходимо по заданному поведению объекта построить модель и определить все ее характеристики. В ряде случаев для этого объект можно подвергнуть заданному внешнему воздействию*

Общая схема постановки и решения задач машинного обучения

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$

Общая схема постановки и решения задач машинного обучения

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y

Общая схема постановки и решения задач машинного обучения

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.

Общая схема постановки и решения задач машинного обучения

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.
- Выбирается параметрическая модель $a(\vec{x}, \vec{\theta})$
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума $\mathcal{L}$.

Лекция 2. Первичная обработка и визуализация данных

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Библиотеки Python: Pandas, Matplotlib, SeaBorn

Jupyter notebook “Использование библиотек Python при анализе данных”:
`https://colab.research.google.com/drive/1tzekUR7XYY6UixcwWQwIjs4wi_FQBoIe`

Лекция 3. Проверка статистических гипотез

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Проверка статистических гипотез

- X – полученная в эксперименте выборка
- $\mathbb{F}$ – множество априори допустимых распределений X
- $F_X \in \mathbb{F}$ – истинное распределение X
- Нулевая гипотеза $H_0 : F_X \in \mathbb{F}_0 \subset \mathbb{F}$
- Альтернатива $H_1 : F_X \in \mathbb{F}_1 = \mathbb{F} \setminus \mathbb{F}_0$

Проверка статистических гипотез

- X – полученная в эксперименте выборка
- $\mathbb{F}$ – множество априори допустимых распределений X
- $F_X \in \mathbb{F}$ – истинное распределение X
- Нулевая гипотеза $H_0 : F_X \in \mathbb{F}_0 \subset \mathbb{F}$
- Альтернатива $H_1 : F_X \in \mathbb{F}_1 = \mathbb{F} \setminus \mathbb{F}_0$

Определение 3.1

Простая гипотеза (или альтернатива) *— такая, при которой множество $\mathbb{F}_0$ (или $\mathbb{F}_1$) содержит единственный элемент. Иначе гипотеза (или альтернатива) называется* ***сложной***.

Проверка статистических гипотез

- X – полученная в эксперименте выборка
- $\mathbb{F}$ – множество априори допустимых распределений X
- $F_X \in \mathbb{F}$ – истинное распределение X
- Нулевая гипотеза $H_0 : F_X \in \mathbb{F}_0 \subset \mathbb{F}$
- Альтернатива $H_1 : F_X \in \mathbb{F}_1 = \mathbb{F} \setminus \mathbb{F}_0$

Определение 3.1

Простая гипотеза (или альтернатива) *— такая, при которой множество $\mathbb{F}_0$ (или $\mathbb{F}_1$) содержит единственный элемент. Иначе гипотеза (или альтернатива) называется* ***сложной****.*

Определение 3.2

Статистический критерий *— правило, согласно которому, наблюдая X, можно принять решение об отклонении гипотезы H_0*

Проверка статистических гипотез

Ошибки 1 и 2 рода

Общий принцип – если при справедливости гипотезы H_0 наблюдаемое событие (выборка X) маловероятно, то такую гипотезу следует отклонить.

Проверка статистических гипотез

Ошибки 1 и 2 рода

Общий принцип – если при справедливости гипотезы H_0 наблюдаемое событие (выборка X) маловероятно, то такую гипотезу следует отклонить.

Определение 3.3

Критическая область — *множество выборок* $\mathbb{X}_1$, *для которых гипотеза* H_0 *отклоняется*

Проверка статистических гипотез

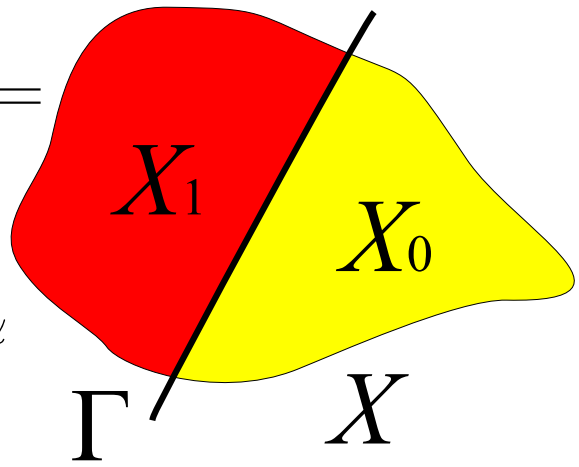
Ошибки 1 и 2 рода

Общий принцип – если при справедливости гипотезы H_0 наблюдаемое событие (выборка X) маловероятно, то такую гипотезу следует отклонить.

Определение 3.3

Критическая область — *множество выборок* $\mathbb{X}_1$, *для которых гипотеза* H_0 *отклоняется*

		Истина: Нет	Истина: Да
Отклонить?	Нет	Ошибка 2 рода β	Правильное не отклонение
	Да	Правильное отклонение	Ошибка 1 рода α



Проверка статистических гипотез

Алгоритм тестирования

1. Определить нулевую гипотезу и альтернативу;
2. Задать уровень значимости (максимальную ошибку 1 рода) α;
3. Определить правило принятия решений;
4. Вычислить статистику теста;
5. Установить результат (отклонение или не отклонение нулевой гипотезы).

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.
- Примем H_0. Тогда $w(x) \sim \mathcal{N}(m, \sigma_0) = \dfrac{1}{\sqrt{2\pi}\sigma_0} \exp\left(-\dfrac{(x-m)^2}{2\sigma_0^2}\right)$.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.
- Примем H_0. Тогда $w(x) \sim \mathcal{N}(m, \sigma_0) = \dfrac{1}{\sqrt{2\pi}\sigma_0} \exp\left(-\dfrac{(x-m)^2}{2\sigma_0^2}\right)$.

$$w(\mathcal{M}\{x\}) \sim \mathcal{N}(m, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathcal{M}\{x\} - m)^2}{2\sigma^2}\right); \sigma = \frac{\sigma_0}{\sqrt{n}}.$$

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.
- Примем H_0. Тогда $w(x) \sim \mathcal{N}(m, \sigma_0) = \dfrac{1}{\sqrt{2\pi}\sigma_0} \exp\left(-\dfrac{(x-m)^2}{2\sigma_0^2}\right)$.

  $w(\mathcal{M}\{x\}) \sim \mathcal{N}(m, \sigma) = \dfrac{1}{\sqrt{2\pi}\sigma} \exp\left(-\dfrac{(\mathcal{M}\{x\}-m)^2}{2\sigma^2}\right); \sigma = \dfrac{\sigma_0}{\sqrt{n}}$.
- Тогда $z = \dfrac{\mathcal{M}\{x\} - m}{\sigma_0}\sqrt{n} \sim \mathcal{N}(0, 1)$.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)

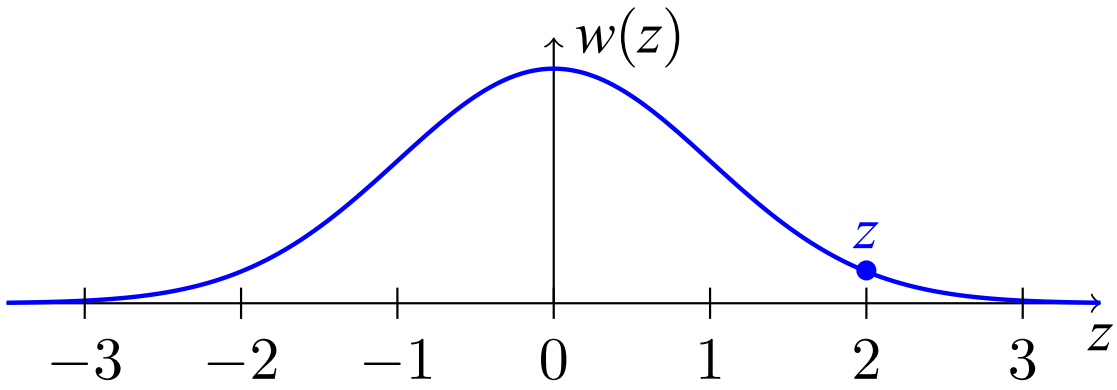
Результаты тестирования студентов распределены нормально со средним $m = 100$ и стандартным отклонением $\sigma_0 = 10$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.
- Примем H_0. Тогда $w(x) \sim \mathcal{N}(m, \sigma_0) = \dfrac{1}{\sqrt{2\pi}\sigma_0} \exp\left(-\dfrac{(x-m)^2}{2\sigma_0^2}\right)$.

  $w(\mathcal{M}\{x\}) \sim \mathcal{N}(m, \sigma) = \dfrac{1}{\sqrt{2\pi}\sigma} \exp\left(-\dfrac{(\mathcal{M}\{x\}-m)^2}{2\sigma^2}\right); \sigma = \dfrac{\sigma_0}{\sqrt{n}}$.
- Тогда $z = \dfrac{\mathcal{M}\{x\} - m}{\sigma_0}\sqrt{n} \sim \mathcal{N}(0, 1)$. При условиях задачи $z = 2$.

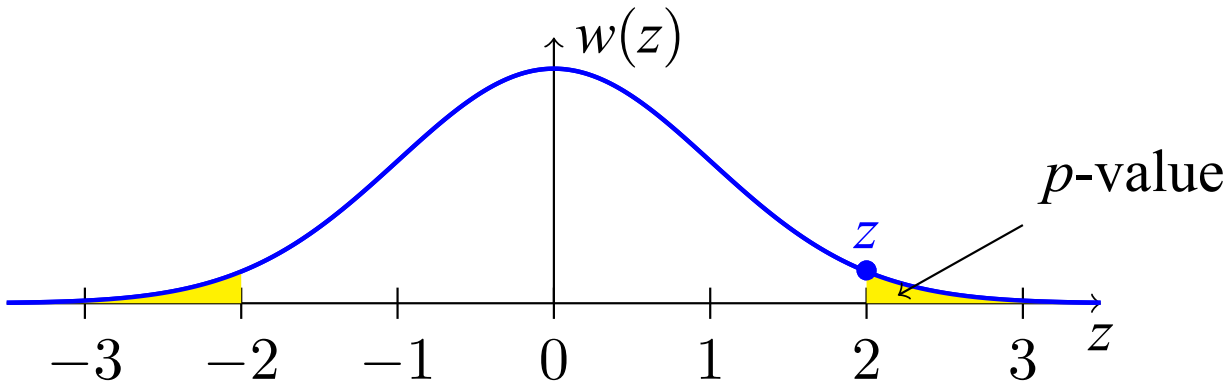
Проверка статистических гипотез

Z-тест (z-критерий Фишера)



Проверка статистических гипотез

Z-тест (z-критерий Фишера)

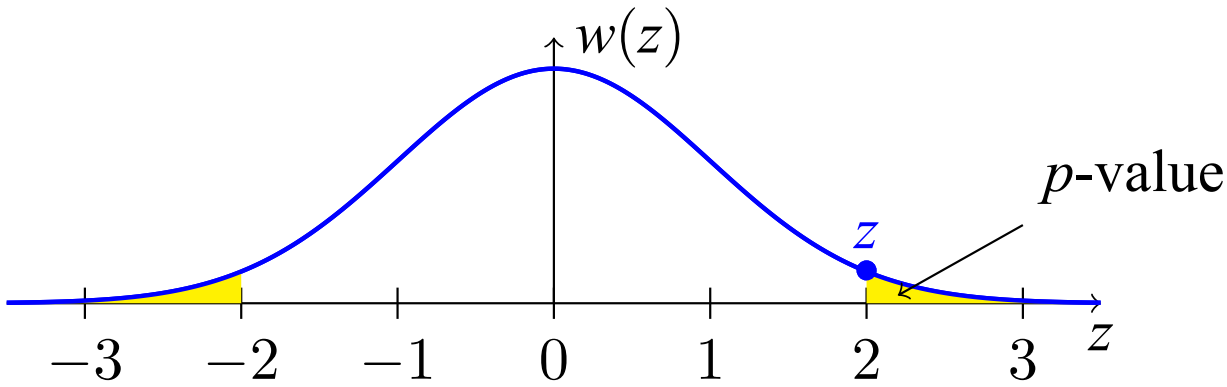


Определение 3.4

*p−**value** — вероятность получить для данной модели распределения такое же или более экстремальное значение статистики, по сравнению с наблюдаемым, при условии, что H_0 верна.*

Проверка статистических гипотез

Z-тест (z-критерий Фишера)



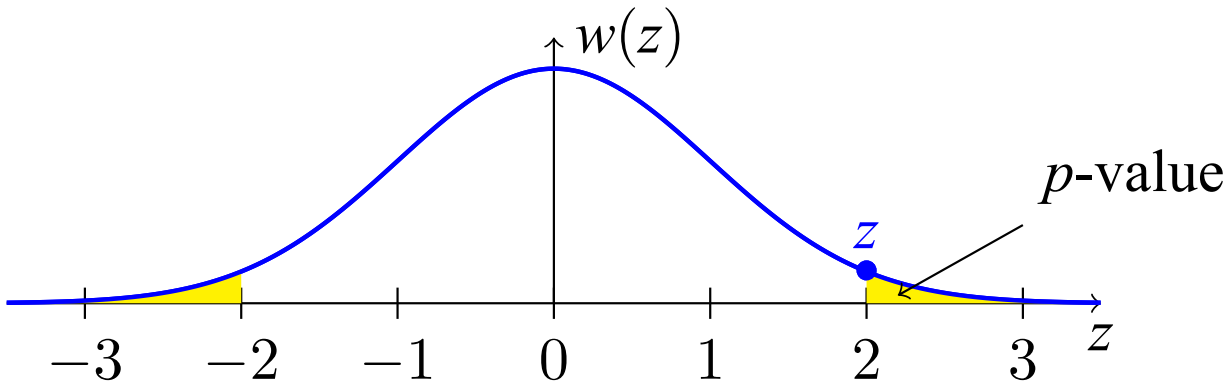
Определение 3.4

*$p-$**value** — вероятность получить для данной модели распределения такое же или более экстремальное значение статистики, по сравнению с наблюдаемым, при условии, что H_0 верна.*

- Если p-value мало ($p(z) \leq \alpha$), следует отклонить H_0.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)



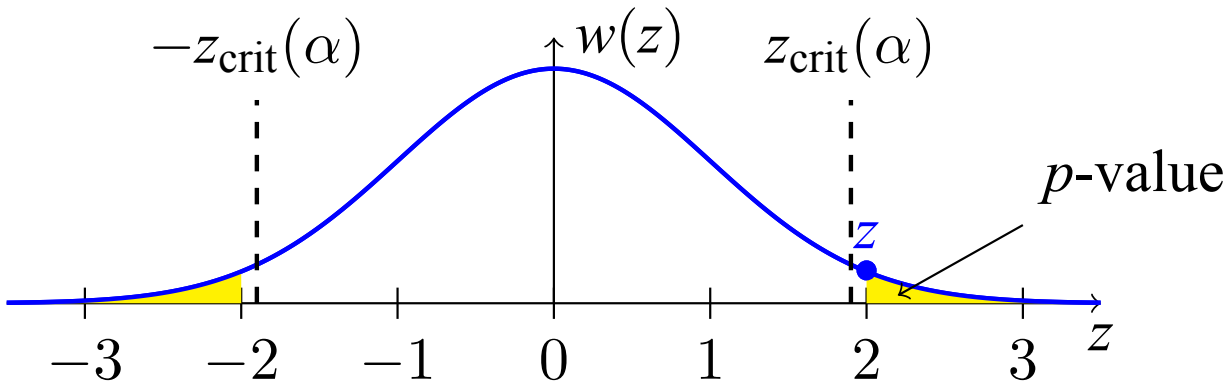
Определение 3.4

*$p-$**value** — вероятность получить для данной модели распределения такое же или более экстремальное значение статистики, по сравнению с наблюдаемым, при условии, что H_0 верна.*

- Если p-value мало ($p(z) \leq \alpha$), следует отклонить H_0.
- Если p-value велико ($p(z) > \alpha$), данных для отклонения H_0 мало.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)



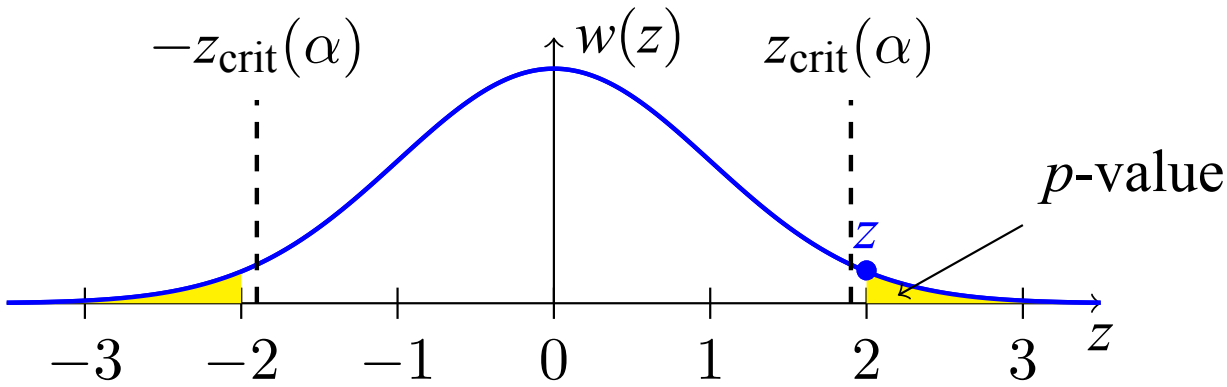
Определение 3.4

*$p-$**value** — вероятность получить для данной модели распределения такое же или более экстремальное значение статистики, по сравнению с наблюдаемым, при условии, что H_0 верна.*

- Если p-value мало ($p(z) \leq \alpha$), следует отклонить H_0.
- Если p-value велико ($p(z) > \alpha$), данных для отклонения H_0 мало.
- Другой подход – определить критическое значение z_{crit} : $p(z_{\text{crit}}) = \alpha$.

Проверка статистических гипотез

Z-тест (z-критерий Фишера)



Определение 3.4

*$p-$**value** — вероятность получить для данной модели распределения такое же или более экстремальное значение статистики, по сравнению с наблюдаемым, при условии, что H_0 верна.*

- Если p-value мало ($p(z) \leq \alpha$), следует отклонить H_0.
- Если p-value велико ($p(z) > \alpha$), данных для отклонения H_0 мало.
- Другой подход – определить критическое значение z_{crit} : $p(z_{\text{crit}}) = \alpha$.
- При условиях задачи p-value$= 0.046$ и $z_{\text{crit}}(0.05) = 1.96$

Проверка статистических гипотез

t-критерий Стьюдента

Результаты тестирования студентов распределены нормально со средним $m = 100$.

Проверка статистических гипотез

t-критерий Стьюдента

Результаты тестирования студентов распределены нормально со средним $m = 100$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$ при стандартном отклонении $\sigma_x = 10$. Следует ли пить таблетку перед тестом?

Проверка статистических гипотез

t-критерий Стьюдента

Результаты тестирования студентов распределены нормально со средним $m = 100$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$ при стандартном отклонении $\sigma_x = 10$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.

Проверка статистических гипотез

t-критерий Стьюдента

Результаты тестирования студентов распределены нормально со средним $m = 100$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$ при стандартном отклонении $\sigma_x = 10$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.

Проверка статистических гипотез

t-критерий Стьюдента

Результаты тестирования студентов распределены нормально со средним $m = 100$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$ при стандартном отклонении $\sigma_x = 10$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.
- Примем H_0. Тогда $t = \dfrac{\mathcal{M}\{x\} - m}{\sigma_x}\sqrt{n} \sim t(n-1)$, где

  $\sigma_x^2 = \sum\limits_{j=1}^{n} \dfrac{(x_j - \mathcal{M}\{x\})^2}{n-1}$ – оценка дисперсии.

Проверка статистических гипотез

t-критерий Стьюдента

Результаты тестирования студентов распределены нормально со средним $m = 100$. После приема волшебной таблетки группа из $n = 25$ студентов получила средний балл $\mathcal{M}\{x\} = 104$ при стандартном отклонении $\sigma_x = 10$. Следует ли пить таблетку перед тестом?

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Примем уровень значимости $\alpha = 0.05$.
- Примем H_0. Тогда $t = \dfrac{\mathcal{M}\{x\} - m}{\sigma_x}\sqrt{n} \sim t(n-1)$, где

  $\sigma_x^2 = \sum\limits_{j=1}^{n} \dfrac{(x_j - \mathcal{M}\{x\})^2}{n-1}$ – оценка дисперсии.
- При условиях задачи $t = 2$ и $t_{\text{crit}}(\alpha = 0.05, n - 1 = 24) = 2.064$.

Проверка статистических гипотез

Критерий согласия Колмогорова

- Пусть X – выборка объемом n из распределения с неизвестной функцией распределения $F_\xi(x)$

Проверка статистических гипотез

Критерий согласия Колмогорова

- Пусть X – выборка объемом n из распределения с неизвестной функцией распределения $F_\xi(x)$
- $\hat{F}_n(x)$ – эмпирическая функция распределения

Проверка статистических гипотез

Критерий согласия Колмогорова

- Пусть X – выборка объемом n из распределения с неизвестной функцией распределения $F_\xi(x)$
- $\hat{F}_n(x)$ – эмпирическая функция распределения
- Нулевая гипотеза $H_0 : F_\xi(x) = F(x)$

Проверка статистических гипотез

Критерий согласия Колмогорова

- Пусть X – выборка объемом n из распределения с неизвестной функцией распределения $F_\xi(x)$
- $\hat{F}_n(x)$ – эмпирическая функция распределения
- Нулевая гипотеза $H_0 : F_\xi(x) = F(x)$
- Статистика критерия: $D_n = D_n(X) = \sup\limits_{-\infty<x<\infty} |\hat{F}_n(x) - F(x)|$

Проверка статистических гипотез

Критерий согласия Колмогорова

- Пусть X – выборка объемом n из распределения с неизвестной функцией распределения $F_\xi(x)$
- $\hat{F}_n(x)$ – эмпирическая функция распределения
- Нулевая гипотеза $H_0 : F_\xi(x) = F(x)$
- Статистика критерия: $D_n = D_n(X) = \sup\limits_{-\infty<x<\infty} |\hat{F}_n(x) - F(x)|$
- Согласно теореме Колмогорова, при больших n (уже при $n \geq 20$)

$$P\left\{ D_n \geq \frac{\lambda_\alpha}{\sqrt{n}} \middle| H_0 \right\} \approx 1 - K(\lambda_\alpha) = \alpha$$

$K(t) = \sum\limits_{j=-\infty}^{\infty} (-1)^j \exp(-2j^2t^2)$ – распределение Колмогорова

Проверка статистических гипотез

В двух группах по n студентов прошло тестирование. В первой группе студенты не принимали таблетки, и средний балл оказался равен 100. Во второй группе студенты принимали таблетки, и средний балл оказался равен 104.

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Принят уровень значимости $\alpha = 0.05$. Получено p-value=0.04.

Проверка статистических гипотез

В двух группах по n студентов прошло тестирование. В первой группе студенты не принимали таблетки, и средний балл оказался равен 100. Во второй группе студенты принимали таблетки, и средний балл оказался равен 104.

- Нулевая гипотеза H_0 : таблетка не влияет на результат.
- Альтернатива: H_1: таблетка изменяет результаты теста.
- Принят уровень значимости $\alpha = 0.05$. Получено p-value=0.04.

Какие утверждения справедливы?

1. Таблетка – причина повышенных результатов с вероятностью 0.96.
2. Вероятность того, что результаты не связаны с таблеткой, равна 0.04.
3. При p-value=0.06 результаты не связаны с таблеткой.
4. Вероятность получить такие различия случайно равна 0.04.
5. Все утверждения неверные.

Лекция 4. Байесовский классификатор. Метод максимального правдоподобия. Априорная и апостериорная информация

Методы машинного обучения в анализе изображений и временных рядов

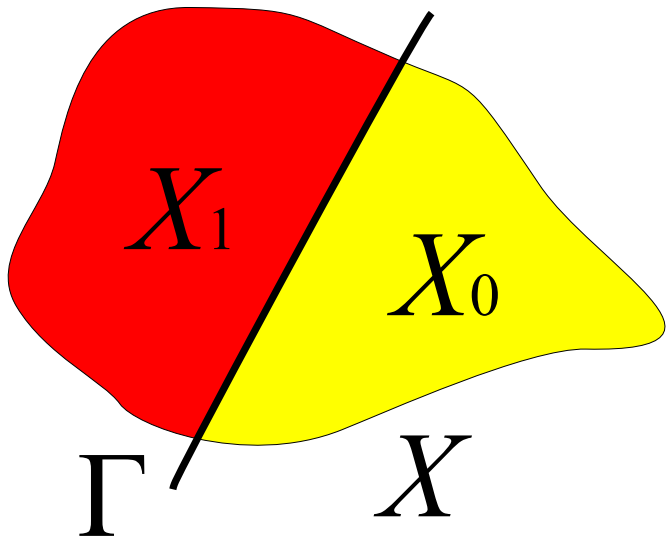
Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Принятие решений в бинарной ситуации

Метод Байеса

		Истина	
		Нет	Да
		p_0	p_1
Альтернатива	1	Ложная тревога	Правильное обнаружение
	0	Правильное необнаружение	Пропуск цели



Принятие решений в бинарной ситуации

Метод Байеса

- Известна “матрица платежей”
- Известны p_0 и p_1 – априорные вероятности событий
- Для каждого $x \in X$ известны $p(x|1)$ и $p(x|0)$

Принятие решений в бинарной ситуации

Метод Байеса

- Известна “матрица платежей”
- Известны p_0 и p_1 – априорные вероятности событий
- Для каждого $x \in X$ известны $p(x|1)$ и $p(x|0)$

$$\text{Да}(x) = \begin{cases} 1 & x \in X_1 \\ 0 & x \in X_0 \end{cases} \quad \text{Нет}(x) = \begin{cases} 1 & x \in X_0 \\ 0 & x \in X_1 \end{cases}$$

Принятие решений в бинарной ситуации

Метод Байеса

- Известна “матрица платежей”
- Известны p_0 и p_1 – априорные вероятности событий
- Для каждого $x \in X$ известны $p(x|1)$ и $p(x|0)$

$$\text{Да}(x) = \begin{cases} 1 & x \in X_1 \\ 0 & x \in X_0 \end{cases} \quad \text{Нет}(x) = \begin{cases} 1 & x \in X_0 \\ 0 & x \in X_1 \end{cases}$$

$$\text{Риск} = \text{ЦЛТ} \cdot p(x \in X_1, 0) + \text{ЦПЦ} \cdot p(x \in X_0, 1) =$$

$$= \text{ЦЛТ} \int_{X_1} p(x|0)p_0 dx + \text{ЦПЦ} \int_{X_0} p(x|1)p_1 dx =$$

$$= \int_X dx[\text{ЦЛТ} \cdot \text{Да}(x)p(x|0)p_0 + \text{ЦПЦ} \cdot \text{Нет}(x)p(x|1)p_1]$$

Принятие решений в бинарной ситуации

Метод Байеса

$$\text{Риск} = \int_X dx[\text{ЦЛТ} \cdot \text{Да}(x)p(x|0)p_0 + \text{ЦПЦ} \cdot \text{Нет}(x)p(x|1)p_1]$$

Принятие решений в бинарной ситуации

Метод Байеса

$$\text{Риск} = \int_X dx[\text{ЦЛТ} \cdot \text{Да}(x)p(x|0)p_0 + \text{ЦПЦ} \cdot \text{Нет}(x)p(x|1)p_1]$$

Минимизируем риск, выбирая для каждого x значения Да или Нет

$$\frac{\text{ЦЛТ} \cdot p(x|0)p_0}{\text{ЦПЦ} \cdot p(x|1)p_1} \underset{\text{Нет}}{\overset{\text{Да}}{\lessgtr}} 1$$

Принятие решений в бинарной ситуации

Метод Байеса

$$\text{Риск} = \int_X dx[\text{ЦЛТ} \cdot \text{Да}(x)p(x|0)p_0 + \text{ЦПЦ} \cdot \text{Нет}(x)p(x|1)p_1]$$

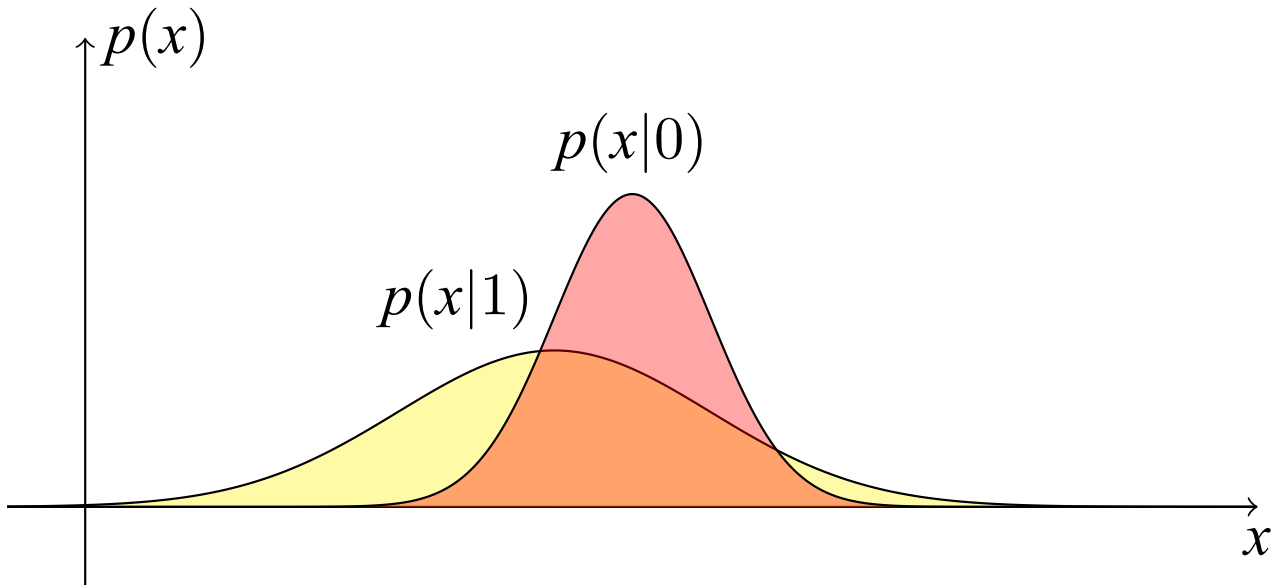
Минимизируем риск, выбирая для каждого x значения Да или Нет

$$\frac{\text{ЦЛТ} \cdot p(x|0)p_0}{\text{ЦПЦ} \cdot p(x|1)p_1} \underset{\text{Нет}}{\overset{\text{Да}}{\lessgtr}} 1$$

$$\text{Отношение правдоподобия } L = \frac{p(x|1)}{p(x|0)} \underset{\text{Нет}}{\overset{\text{Да}}{\gtrless}} \frac{\text{ЦЛТ} \cdot p_0}{\text{ЦПЦ} \cdot p_1}$$

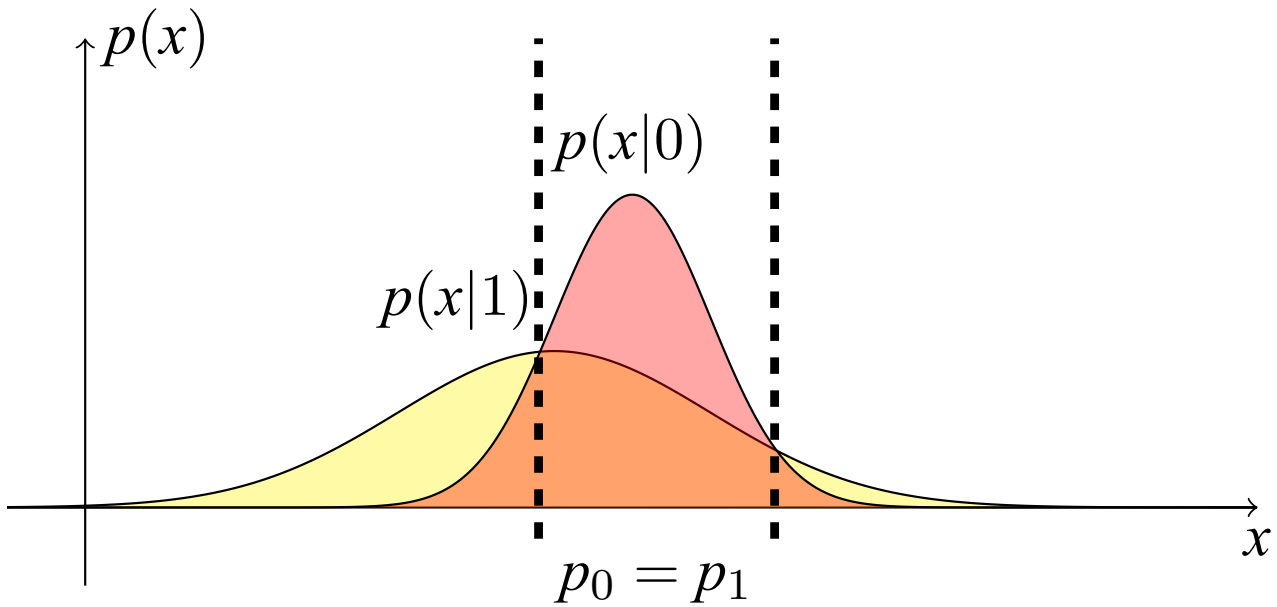
Принятие решений в бинарной ситуации

Построение разделяющих поверхностей



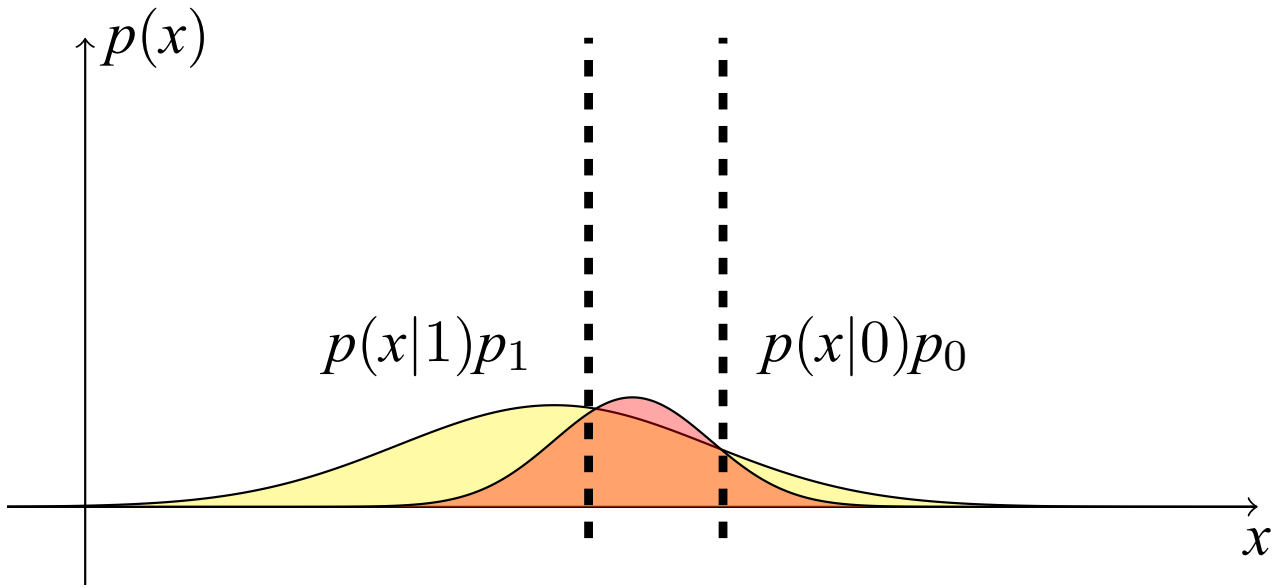
Принятие решений в бинарной ситуации

Построение разделяющих поверхностей



Принятие решений в бинарной ситуации

Построение разделяющих поверхностей



Принятие решений в бинарной ситуации

Наивный байесовский класификатор

Определение 4.1

Наивный байесовский классификатор (naïve Bayes) *— специальный частный случай байесовского классификатора, основанный на дополнительном предположении, что объекты описываются статистически независимыми признаками*

$$w(x_1, x_2, \ldots, x_n|\theta) = \prod_{j=1}^{k} w(x_j|\theta)$$

Принятие решений в бинарной ситуации

Наивный байесовский класификатор

Определение 4.1

Наивный байесовский классификатор (naïve Bayes) *— специальный частный случай байесовского классификатора, основанный на дополнительном предположении, что объекты описываются статистически независимыми признаками*

$$w(x_1, x_2, \ldots, x_n|\theta) = \prod_{j=1}^{k} w(x_j|\theta)$$

На практике это условие почти никогда не выполняется,
но позволяет упростить задачу.

Принятие решений при нескольких возможных исходах

Метод Байеса

- Известна цена ошибки λ_y для каждого события (класса) $y \in Y$
- Известны априорные вероятности событий p_y
- Для каждого $x \in X$ известны $p(x|y)$

Принятие решений при нескольких возможных исходах

Метод Байеса

- Известна цена ошибки λ_y для каждого события (класса) $y \in Y$
- Известны априорные вероятности событий p_y
- Для каждого $x \in X$ известны $p(x|y)$

$$\text{Ошибка}(x) = \begin{cases} 1 & a(x) \neq y \\ 0 & a(x) = y \end{cases}$$

Принятие решений при нескольких возможных исходах

Метод Байеса

- Известна цена ошибки λ_y для каждого события (класса) $y \in Y$
- Известны априорные вероятности событий p_y
- Для каждого $x \in X$ известны $p(x|y)$

$$\text{Ошибка}(x) = \begin{cases} 1 & a(x) \neq y \\ 0 & a(x) = y \end{cases}$$

$$\text{Риск} = \sum_{y \in Y} \lambda_y \int p(x,y)\text{Ошибка}(x)dx$$

Принятие решений при нескольких возможных исходах

Метод Байеса

- Известна цена ошибки λ_y для каждого события (класса) $y \in Y$
- Известны априорные вероятности событий p_y
- Для каждого $x \in X$ известны $p(x|y)$

$$\text{Ошибка}(x) = \begin{cases} 1 & a(x) \neq y \\ 0 & a(x) = y \end{cases}$$

$$\text{Риск} = \sum_{y \in Y} \lambda_y \int p(x,y)\text{Ошибка}(x)dx$$

Минимум риска достигается, если алгоритм $a(x)$

$$a(x) = \arg\max_{y \in Y} \lambda_y p_y p(x|y)$$

Оценка постоянных параметров сигнала

- Необходимо построить оценку $\hat{\theta}$ параметра θ

Оценка постоянных параметров сигнала

- Необходимо построить оценку $\hat{\theta}$ параметра θ
- В эксперименте получены значения $\vec{y} = y_1, y_2, \ldots y_n$, причем $w(\vec{y}|\theta)$ зависит от θ

Оценка постоянных параметров сигнала

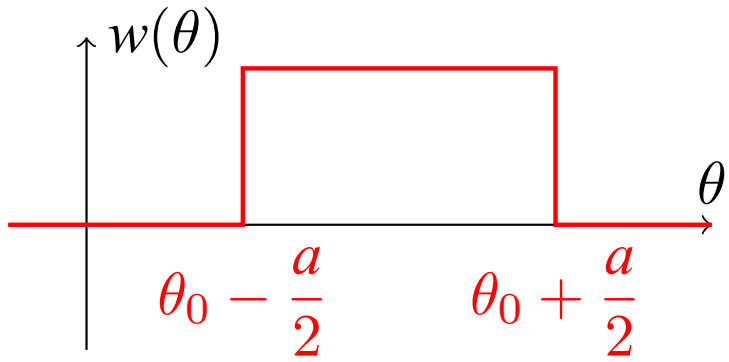
- Необходимо построить оценку $\hat{\theta}$ параметра θ
- В эксперименте получены значения $\vec{y} = y_1, y_2, \ldots y_n$, причем $w(\vec{y}|\theta)$ зависит от θ
- Априори известно распределение $w(\theta)$

Оценка постоянных параметров сигнала

- Необходимо построить оценку $\hat{\theta}$ параметра θ
- В эксперименте получены значения $\vec{y} = y_1, y_2, \ldots y_n$, причем $w(\vec{y}|\theta)$ зависит от θ
- Априори известно распределение $w(\theta)$

Пример:

$$w(\theta) = \begin{cases} 1/a & |\theta - \theta_0| \le a/2 \\ 0 & |\theta - \theta_0| > a/2 \end{cases}$$

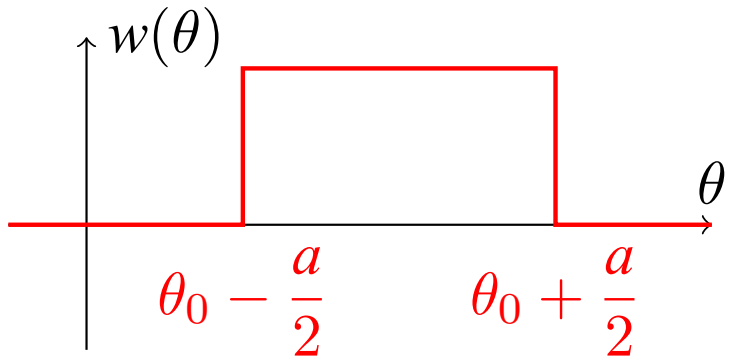


Оценка постоянных параметров сигнала

- Необходимо построить оценку $\hat{\theta}$ параметра θ
- В эксперименте получены значения $\vec{y} = y_1, y_2, \ldots y_n$, причем $w(\vec{y}|\theta)$ зависит от θ
- Априори известно распределение $w(\theta)$

Пример:

$$w(\theta) = \begin{cases} 1/a & |\theta - \theta_0| \le a/2 \\ 0 & |\theta - \theta_0| > a/2 \end{cases}$$



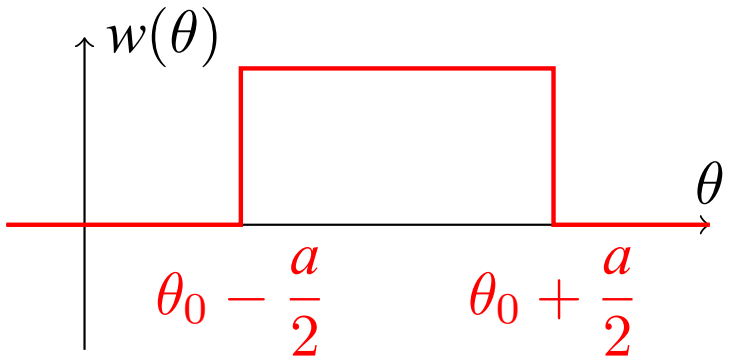
Если $a \to 0$, то играет роль априорная информация;

Оценка постоянных параметров сигнала

- Необходимо построить оценку $\hat{\theta}$ параметра θ
- В эксперименте получены значения $\vec{y} = y_1, y_2, \ldots y_n$, причем $w(\vec{y}|\theta)$ зависит от θ
- Априори известно распределение $w(\theta)$

Пример:

$$w(\theta) = \begin{cases} 1/a & |\theta - \theta_0| \le a/2 \\ 0 & |\theta - \theta_0| > a/2 \end{cases}$$



Если $a \to 0$, то играет роль априорная информация;

Если $a \to \infty$, то важны данные, полученные в эксперименте

Оценка постоянных параметров

Байесовский подход

В эксперименте получены данные $\vec{y}$

Ошибка оценки $\epsilon(\theta;\hat{\theta}(\vec{y})) = \hat{\theta}(\vec{y}) - \theta$

Характеристика потерь $\mathcal{L}(\epsilon) = \mathcal{L}(\theta;\hat{\theta}(\vec{y}))$

Нужно найти минимум средней функции потерь

$$\mathcal{M}\left\{\mathcal{L}(\theta;\hat{\theta}(\vec{y}))\right\} = \int\limits_{-\infty}^{\infty} d\theta \iint\limits_{-\infty}^{\infty} d\vec{y}\mathcal{L}(\theta;\hat{\theta}(\vec{y}))w(\theta,\vec{y})$$

Оценка постоянных параметров

Примеры функции потерь

- Квадратичная: $\mathcal{L}_{\text{quad}}(\theta;\hat{\theta}) = (\theta - \hat{\theta})^2$
- Модульная: $\mathcal{L}_{\text{abs}}(\theta;\hat{\theta}) = |\theta - \hat{\theta}|$
- Простая: $\mathcal{L}_{\text{simple}}(\theta;\hat{\theta}) = -\delta(\theta - \hat{\theta})$

Оценка постоянных параметров

$$\mathcal{M}\left\{\mathcal{L}(\theta;\hat{\theta}(\vec{y}))\right\} = \int\limits_{-\infty}^{\infty} d\theta \iint\limits_{-\infty}^{\infty} d\vec{y}\mathcal{L}(\theta;\hat{\theta}(\vec{y}))w(\theta,y) =$$

$$= \iint\limits_{-\infty}^{\infty} d\vec{y} \int\limits_{-\infty}^{\infty} d\theta\mathcal{L}(\theta;\hat{\theta}(\vec{y}))w(\theta|\vec{y})w(\vec{y}) = \iint\limits_{-\infty}^{\infty} d\vec{y}w(\vec{y}) \underbrace{\int\limits_{-\infty}^{\infty} d\theta\mathcal{L}(\theta;\hat{\theta}(\vec{y}))w(\theta|\vec{y})}_{\mathcal{L}_a}$$

$$\mathcal{L}_a \to \min \quad \frac{\partial\mathcal{L}_a}{\partial\hat{\theta}} = 0$$

Оценка постоянных параметров

Квадратичная функция потерь: $\mathcal{L}_{\text{quad}}(\theta;\hat{\theta}) = (\theta - \hat{\theta})^2$

$$\int\limits_{-\infty}^{\infty} d\theta(\theta - \hat{\theta}(\vec{y}))w(\theta|\vec{y}) = 0 \Rightarrow \quad \hat{\theta}(\vec{y}) = \frac{\int \theta w(\theta|\vec{y})d\theta}{\int w(\theta|\vec{y})d\theta} = \int \theta w(\theta|\vec{y})d\theta$$

Это математическое ожидание для апостериорного распределения $w(\theta|\vec{y})$

Оценка постоянных параметров

Квадратичная функция потерь: $\mathcal{L}_{\text{quad}}(\theta;\hat{\theta}) = (\theta - \hat{\theta})^2$

$$\int_{-\infty}^{\infty} d\theta(\theta - \hat{\theta}(\vec{y}))w(\theta|\vec{y}) = 0 \Rightarrow \quad \hat{\theta}(\vec{y}) = \frac{\int \theta w(\theta|\vec{y})d\theta}{\int w(\theta|\vec{y})d\theta} = \int \theta w(\theta|\vec{y})d\theta$$

Это математическое ожидание для апостериорного распределения $w(\theta|\vec{y})$

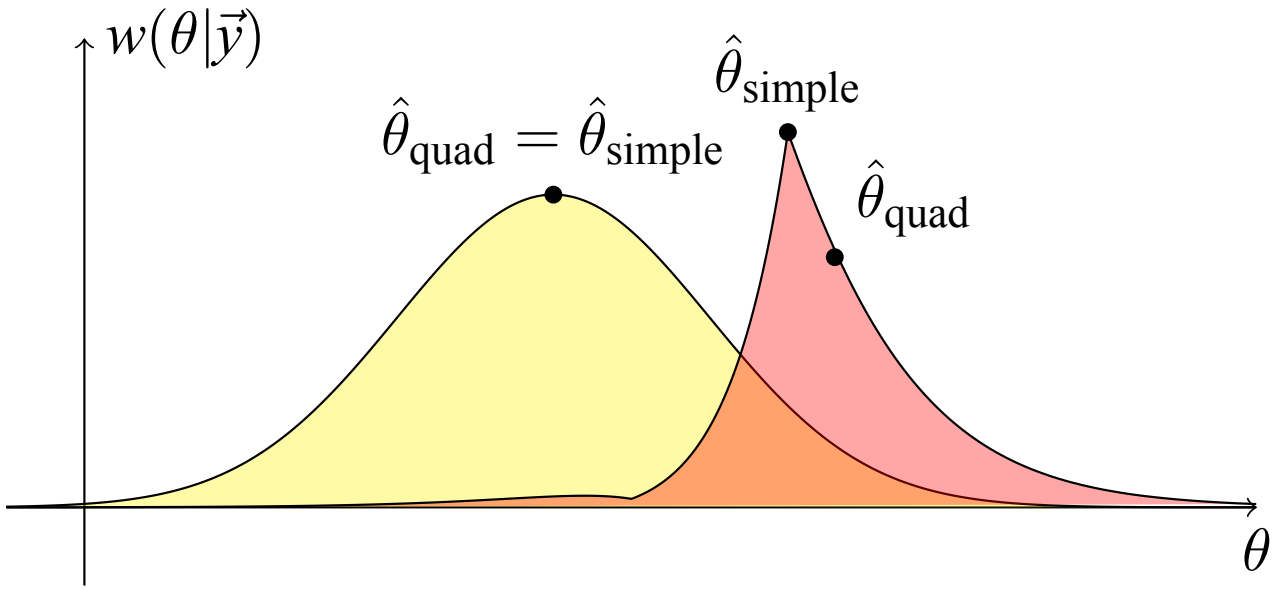
Простая функция потерь: $\mathcal{L}_{\text{simple}}(\theta;\hat{\theta}) = -\delta(\theta - \hat{\theta})$

$$\frac{\partial}{\partial\hat{\theta}} \int w(\theta|\vec{y})\delta(\hat{\theta}(\vec{y}) - \theta)d\theta = \frac{\partial w(\theta|\vec{y})}{\partial\hat{\theta}} = 0$$

Это максимум апостериорного распределения $w(\theta|\vec{y})$

Оценка постоянных параметров

Оценки для симметричных и асимметричных распределений



Оценка постоянных параметров

Функция правдоподобия

Формула Байеса: $w(\theta|\vec{y})w(\vec{y}) = w(\vec{y}|\theta)w(\theta)$

$$\Rightarrow w(\theta|\vec{y}) = \frac{1}{w(\vec{y})}w(\vec{y}|\theta)w(\theta)$$

Оценка постоянных параметров

Функция правдоподобия

Формула Байеса: $w(\theta|\vec{y})w(\vec{y}) = w(\vec{y}|\theta)w(\theta)$

$$\Rightarrow w(\theta|\vec{y}) = \frac{1}{w(\vec{y})}w(\vec{y}|\theta)w(\theta)$$

Определение 4.2

Функция правдопобия *— условная вероятность* $L(\theta) \equiv w(\vec{y}|\theta)$*, взятая при фиксированнных (полученных в эксперименте) значениях* $\vec{y}$

Оценка постоянных параметров

Функция правдоподобия

Формула Байеса: $w(\theta|\vec{y})w(\vec{y}) = w(\vec{y}|\theta)w(\theta)$

$$\Rightarrow w(\theta|\vec{y}) = \frac{1}{w(\vec{y})}w(\vec{y}|\theta)w(\theta)$$

Определение 4.2

Функция правдопобия *— условная вероятность* $L(\theta) \equiv w(\vec{y}|\theta)$*, взятая при фиксированнных (полученных в эксперименте) значениях* $\vec{y}$

- Метод максимума апостериорной функции распредления $w(\hat{\theta}|\vec{y})$
- Метод максимального правдоподобия $L(\theta)$

Наблюдения параметра на фоне гауссового шума

$\vec{y} = \{y_1, y_2, \ldots, y_k\}$ – наблюдения параметра θ
на фоне шума $\vec{n} = \{n_1, n_2, \ldots, n_k\} \quad (0; \sigma^2)$

Модель такой системы: $y_j = \theta + n_j$

Априорное распределение θ нормально:

$$w(\theta) = \frac{1}{\sqrt{2\pi}\sigma_0} \exp\left(-\frac{(\theta - \theta_0)^2}{2\sigma_0^2}\right)$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

$$L(\theta) \equiv w(\vec{y}|\theta) = \prod_j w(y_j|\theta) = \frac{1}{(2\pi)^{k/2}\sigma^k} \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{k}(y_j-\theta)^2\right)$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

$$L(\theta) \equiv w(\vec{y}|\theta) = \prod_j w(y_j|\theta) = \frac{1}{(2\pi)^{k/2}\sigma^k} \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{k}(y_j-\theta)^2\right)$$

$$\max L(\theta) \Leftrightarrow \max \ln L(\theta)$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

$$L(\theta) \equiv w(\vec{y}|\theta) = \prod_j w(y_j|\theta) = \frac{1}{(2\pi)^{k/2}\sigma^k} \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{k}(y_j-\theta)^2\right)$$

$$\max L(\theta) \Leftrightarrow \max \ln L(\theta)$$

$$\frac{\partial}{\partial\theta}\sum_{j=1}^{k}(y_j-\theta)^2 = 0$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

$$L(\theta) \equiv w(\vec{y}|\theta) = \prod_j w(y_j|\theta) = \frac{1}{(2\pi)^{k/2}\sigma^k} \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{k}(y_j-\theta)^2\right)$$

$$\max L(\theta) \Leftrightarrow \max \ln L(\theta)$$

$$\frac{\partial}{\partial\theta}\sum_{j=1}^{k}(y_j-\theta)^2 = 0 \quad \Rightarrow \quad \sum_{j=1}^{k}(y_j-\theta) = 0$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

$$L(\theta) \equiv w(\vec{y}|\theta) = \prod_j w(y_j|\theta) = \frac{1}{(2\pi)^{k/2}\sigma^k} \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{k}(y_j-\theta)^2\right)$$

$$\max L(\theta) \Leftrightarrow \max \ln L(\theta)$$

$$\frac{\partial}{\partial\theta}\sum_{j=1}^{k}(y_j-\theta)^2 = 0 \quad \Rightarrow \quad \sum_{j=1}^{k}(y_j-\theta) = 0 \quad \Rightarrow \quad \hat{\theta}_{\text{mle}} = \frac{1}{k}\sum_{j=1}^{k} y_j$$

Наблюдения параметра на фоне гауссового шума

Метод максимального правдоподобия

$$y_j = \theta + n_j \quad \Rightarrow \quad n_j = y_j - \theta$$

$$L(\theta) \equiv w(\vec{y}|\theta) = \prod_j w(y_j|\theta) = \frac{1}{(2\pi)^{k/2}\sigma^k} \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{k}(y_j-\theta)^2\right)$$

$$\max L(\theta) \Leftrightarrow \max \ln L(\theta)$$

$$\frac{\partial}{\partial\theta}\sum_{j=1}^{k}(y_j-\theta)^2 = 0 \quad \Rightarrow \quad \sum_{j=1}^{k}(y_j-\theta) = 0 \quad \Rightarrow \quad \hat{\theta}_{\text{mle}} = \frac{1}{k}\sum_{j=1}^{k} y_j$$

$$\sigma^2_{\text{mle}} = \mathcal{M}\left\{(\theta_{\text{mle}}-\theta)^2\right\} =$$

$$= \mathcal{M}\left\{\left(\frac{1}{k}\sum_{j=1}^{k} y_j - \theta\right)^2\right\} = \mathcal{M}\left\{\frac{1}{k}\sum_{j=1}^{k}\underbrace{(y_j-\theta)}_{n_j}{}^2\right\} = \frac{\sigma^2}{k}$$

Наблюдения параметра на фоне гауссового шума

Метод максимума апостериорной функции распределения

$$w(\theta|\vec{y}) = \frac{1}{w(\vec{y})} w(\vec{y}|\theta) w(\theta) =$$

$$= \frac{1}{w(\vec{y})} \frac{1}{(2\pi)^{(k+1)/2} \sigma^k \sigma_0} \exp\left(-\frac{1}{2\sigma^2} \sum_{j=1}^{k} (y_j - \theta)^2 - \frac{1}{2\sigma_0^2} (\theta - \theta_0)^2 \right)$$

Наблюдения параметра на фоне гауссового шума

Метод максимума апостериорной функции распредления

$$w(\theta|\vec{y}) = \frac{1}{w(\vec{y})} w(\vec{y}|\theta) w(\theta) =$$

$$= \frac{1}{w(\vec{y})} \frac{1}{(2\pi)^{(k+1)/2} \sigma^k \sigma_0} \exp\left(-\frac{1}{2\sigma^2} \sum_{j=1}^{k} (y_j - \theta)^2 - \frac{1}{2\sigma_0^2} (\theta - \theta_0)^2 \right)$$

$$\max w(\theta|\vec{y}) \Leftrightarrow \max \ln w(\theta|\vec{y})$$

Наблюдения параметра на фоне гауссового шума

Метод максимума апостериорной функции распределения

$$w(\theta|\vec{y}) = \frac{1}{w(\vec{y})} w(\vec{y}|\theta) w(\theta) =$$

$$= \frac{1}{w(\vec{y})} \frac{1}{(2\pi)^{(k+1)/2} \sigma^k \sigma_0} \exp\left(-\frac{1}{2\sigma^2} \sum_{j=1}^{k} (y_j - \theta)^2 - \frac{1}{2\sigma_0^2} (\theta - \theta_0)^2 \right)$$

$$\max w(\theta|\vec{y}) \Leftrightarrow \max \ln w(\theta|\vec{y})$$

$$\frac{1}{\sigma^2} \sum_{j=1}^{k} (y_j - \hat{\theta}_{\text{map}}) + \frac{1}{\sigma_0^2} (\hat{\theta}_{\text{map}} - \theta_0) = 0$$

Наблюдения параметра на фоне гауссового шума

Метод максимума апостериорной функции распределения

$$\frac{1}{\sigma^2}\sum_{j=1}^{k}(y_j-\hat{\theta}_{\text{map}})+\frac{1}{\sigma_0^2}(\hat{\theta}_{\text{map}}-\theta_0)=0$$

$$\hat{\theta}_{\text{map}}\left(\underbrace{\frac{k}{\sigma^2}}_{1/\sigma^2_{\text{mle}}}+\frac{1}{\sigma_0^2}\right)=\frac{\theta_0}{\sigma_0^2}+\underbrace{\frac{1}{k}\sum_{j=1}^{k}y_j}_{\hat{\theta}^2_{\text{mle}}}\underbrace{\frac{k}{\sigma^2}}_{1/\sigma^2_{\text{mle}}}$$

Наблюдения параметра на фоне гауссового шума

Метод максимума апостериорной функции распределения

$$\frac{1}{\sigma^2}\sum_{j=1}^{k}(y_j-\hat{\theta}_{\text{map}})+\frac{1}{\sigma_0^2}(\hat{\theta}_{\text{map}}-\theta_0)=0$$

$$\hat{\theta}_{\text{map}}\left(\underbrace{\frac{k}{\sigma^2}}_{1/\sigma^2_{\text{mle}}}+\frac{1}{\sigma_0^2}\right)=\frac{\theta_0}{\sigma_0^2}+\underbrace{\frac{1}{k}\sum_{j=1}^{k}y_j}_{\hat{\theta}^2_{\text{mle}}}\underbrace{\frac{k}{\sigma^2}}_{1/\sigma^2_{\text{mle}}}$$

$$\hat{\theta}_{\text{map}}=\frac{\sigma^2_{\text{mle}}}{\sigma_0^2+\sigma^2_{\text{mle}}}\theta_0+\frac{\sigma_0^2}{\sigma_0^2+\sigma^2_{\text{mle}}}\hat{\theta}_{\text{mle}}$$

Наблюдения параметра на фоне гауссового шума

Метод максимума апостериорной функции распределения

$$\frac{1}{\sigma^2}\sum_{j=1}^{k}(y_j-\hat{\theta}_{\text{map}})+\frac{1}{\sigma_0^2}(\hat{\theta}_{\text{map}}-\theta_0)=0$$

$$\hat{\theta}_{\text{map}}\left(\underbrace{\frac{k}{\sigma^2}}_{1/\sigma^2_{\text{mle}}}+\frac{1}{\sigma_0^2}\right)=\frac{\theta_0}{\sigma_0^2}+\underbrace{\frac{1}{k}\sum_{j=1}^{k}y_j}_{\hat{\theta}^2_{\text{mle}}}\underbrace{\frac{k}{\sigma^2}}_{1/\sigma^2_{\text{mle}}}$$

$$\hat{\theta}_{\text{map}}=\frac{\sigma^2_{\text{mle}}}{\sigma_0^2+\sigma^2_{\text{mle}}}\theta_0+\frac{\sigma_0^2}{\sigma_0^2+\sigma^2_{\text{mle}}}\hat{\theta}_{\text{mle}}$$

Если $\sigma_0^2 \gg \sigma^2_{\text{mle}}$, то $\hat{\theta}_{\text{map}} \approx \hat{\theta}_{\text{mle}}$, т.е. эксперимент имеет решающее значение

Если $\sigma_0^2 \ll \sigma^2_{\text{mle}}$, то данные эксперимента почти не существенны

Метод моментов

Метод максимального правдоподобия в ряде случаев сложен

Метод моментов

Метод максимального правдоподобия в ряде случаев сложен

Пусть $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_m\}$ – неизвестные параметры

Метод моментов

Метод максимального правдоподобия в ряде случаев сложен

Пусть $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_m\}$ – неизвестные параметры

$$m_k(\vec{\theta}) = \int\limits_{-\infty}^{\infty} y^k w(y|\vec{\theta})dy$$ – начальные моменты

$$\mu_k(\vec{\theta}) = \int\limits_{-\infty}^{\infty} (y - \mathcal{M}\{y\})^k w(y|\vec{\theta})dy$$ – центральные моменты

Метод моментов

Метод максимального правдоподобия в ряде случаев сложен

Пусть $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_m\}$ – неизвестные параметры

$$m_k(\vec{\theta}) = \int\limits_{-\infty}^{\infty} y^k w(y|\vec{\theta})dy$$ – начальные моменты

$$\mu_k(\vec{\theta}) = \int\limits_{-\infty}^{\infty} (y - \mathcal{M}\{y\})^k w(y|\vec{\theta})dy$$ – центральные моменты

$\breve{m}_k = \frac{1}{n}\sum y_j^k$ – выборочные начальные моменты $\breve{\mu}_k = \frac{1}{n}\sum (y_j - \hat{m}_1)^k$ – выборочные центральные моменты

Метод моментов

Метод максимального правдоподобия в ряде случаев сложен

Пусть $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_m\}$ – неизвестные параметры

$$m_k(\vec{\theta}) = \int\limits_{-\infty}^{\infty} y^k w(y|\vec{\theta})dy$$ – начальные моменты

$$\mu_k(\vec{\theta}) = \int\limits_{-\infty}^{\infty} (y - \mathcal{M}\{y\})^k w(y|\vec{\theta})dy$$ – центральные моменты

$\breve{m}_k = \frac{1}{n}\sum y_j^k$ – выборочные начальные моменты $\breve{\mu}_k = \frac{1}{n}\sum (y_j - \hat{m}_1)^k$ – выборочные центральные моменты

$$\breve{m}_k = m_k(\vec{\theta}_{\text{mom}}) \quad \text{или} \quad \breve{\mu}_k = \mu_k(\vec{\theta}_{\text{mom}})$$

Метод моментов

Пример: гамма-распределение

$$w(x|\alpha,\beta) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)} x^{\alpha-1} \exp(-x/\beta); \quad x \geq 0; \beta > 0$$

Метод моментов

Пример: гамма-распределение

$$w(x|\alpha,\beta) = \frac{1}{\beta^\alpha \Gamma(\alpha)} x^{\alpha-1} \exp(-x/\beta); \quad x \geq 0; \beta > 0$$

$$m_1 = \alpha\beta; \quad m_2 = \alpha(\alpha+1)\beta^2; \quad \mu_2 = \alpha\beta^2; \quad \mu_3 = 2\alpha\beta^3$$

Метод моментов

Пример: гамма-распределение

$$w(x|\alpha,\beta) = \frac{1}{\beta^\alpha \Gamma(\alpha)} x^{\alpha-1} \exp(-x/\beta); \quad x \geq 0; \beta > 0$$

$$m_1 = \alpha\beta; \quad m_2 = \alpha(\alpha+1)\beta^2; \quad \mu_2 = \alpha\beta^2; \quad \mu_3 = 2\alpha\beta^3$$

$$m_1(\alpha,\beta) = \breve{m}_1 \Rightarrow \alpha\beta = \frac{1}{n}\sum y_j$$

$$m_2(\alpha,\beta) = \breve{m}_2 \Rightarrow \alpha(\alpha+1)\beta^2 = \frac{1}{n}\sum y_j^2$$

Метод моментов

Пример: гамма-распределение

$$w(x|\alpha,\beta) = \frac{1}{\beta^\alpha \Gamma(\alpha)} x^{\alpha-1} \exp(-x/\beta); \quad x \geq 0; \beta > 0$$

$$m_1 = \alpha\beta; \quad m_2 = \alpha(\alpha+1)\beta^2; \quad \mu_2 = \alpha\beta^2; \quad \mu_3 = 2\alpha\beta^3$$

$$m_1(\alpha,\beta) = \breve{m}_1 \Rightarrow \alpha\beta = \frac{1}{n}\sum y_j$$

$$m_2(\alpha,\beta) = \breve{m}_2 \Rightarrow \alpha(\alpha+1)\beta^2 = \frac{1}{n}\sum y_j^2$$

$$\hat{\alpha}_{\text{mom}} = \frac{\breve{m}_1^2}{\breve{m}_2 - \breve{m}_1^2}; \quad \hat{\beta}_{\text{mom}} = \frac{\breve{m}_2 - \breve{m}_1^2}{\breve{m}_1}$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

$$w(y_j|\theta) = \begin{cases} 1/\theta; & 0 \le y_j \le \theta \\ 0; & y_j < 0; y_j > \theta \end{cases}$$

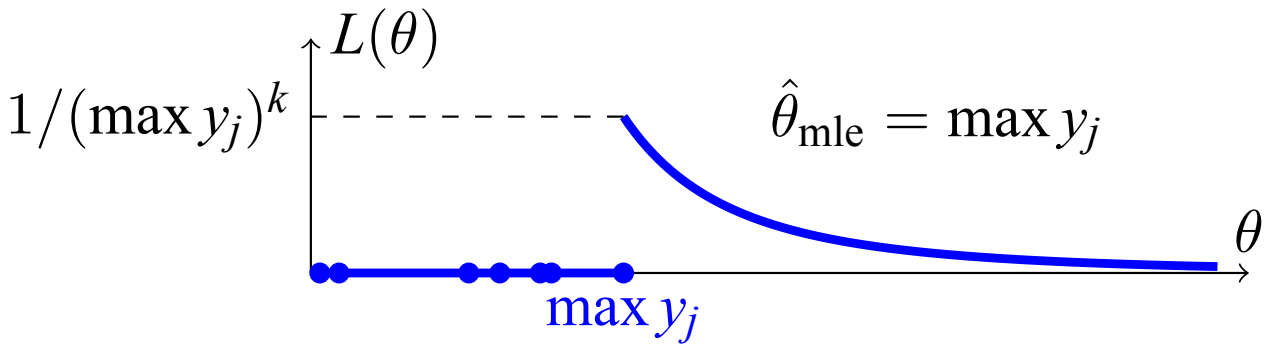
Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

$$w(y_j|\theta) = \begin{cases} 1/\theta; & 0 \le y_j \le \theta \\ 0; & y_j < 0; y_j > \theta \end{cases}$$

Метод максимального правдоподобия

$$L(\theta) = \prod w(y_j|\theta) = \begin{cases} 1/\theta^k; & 0 \le y_1, y_2, \ldots, y_k \le \theta \\ 0; & \exists t : y_t < 0; y_t > \theta \end{cases}$$



Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x)$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x)$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x) = \prod P(y_j < x)$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x) = \prod P(y_j < x) = x^k/\theta^k$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x) = \prod P(y_j < x) = x^k/\theta^k$$

$$w(x) = \frac{dF}{dx} = kx^{k-1}/\theta^k$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x) = \prod P(y_j < x) = x^k/\theta^k$$

$$w(x) = \frac{dF}{dx} = kx^{k-1}/\theta^k$$

$$\mathcal{M}\left\{\hat{\theta}_{\text{mle}}(\vec{y})\right\} = \int_0^\infty xw(x)dx = \frac{k}{k+1}\theta$$

Значит, оценка $\hat{\theta}_{\text{mle}}(\vec{y})$ – смещенная!

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x) = \prod P(y_j < x) = x^k/\theta^k$$

$$w(x) = \frac{dF}{dx} = kx^{k-1}/\theta^k$$

$$\mathcal{M}\left\{\hat{\theta}_{\text{mle}}(\vec{y})\right\} = \int_0^\infty xw(x)dx = \frac{k}{k+1}\theta$$

Значит, оценка $\hat{\theta}_{\text{mle}}(\vec{y})$ – смещенная!

$$\text{Несмещенная оценка } \hat{\theta}_{\text{mle-nobias}}(\vec{y}) = \frac{k}{k+1}\max y_j$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Функция распределения

$$F(x) = P(X < x) = P(\max y_j < x) = \prod P(y_j < x) = x^k/\theta^k$$

$$w(x) = \frac{dF}{dx} = kx^{k-1}/\theta^k$$

$$\mathcal{M}\left\{\hat{\theta}_{\text{mle}}(\vec{y})\right\} = \int_0^\infty xw(x)dx = \frac{k}{k+1}\theta$$

Значит, оценка $\hat{\theta}_{\text{mle}}(\vec{y})$ – смещенная!

$$\text{Несмещенная оценка } \hat{\theta}_{\text{mle-nobias}}(\vec{y}) = \frac{k}{k+1}\max y_j$$

$$\mathcal{M}\left\{(\hat{\theta}_{\text{mle-nobias}}(\vec{y}) - \theta)^2\right\} = \frac{\theta^2}{k(k+2)}$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Метод моментов

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Метод моментов

$$m_1(\theta) = \frac{\theta}{2}; \quad \frac{1}{k}\sum y_j = \frac{\hat{\theta}_{\text{mom}}(\vec{y})}{2}$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Метод моментов

$$m_1(\theta) = \frac{\theta}{2}; \quad \frac{1}{k}\sum y_j = \frac{\hat{\theta}_{\text{mom}}(\vec{y})}{2}$$

$$\hat{\theta}_{\text{mom}}(\vec{y}) = \frac{2}{k}\sum y_j$$

Метод моментов и метод максимального правдоподобия

Пример: шарик в ящике

Метод моментов

$$m_1(\theta) = \frac{\theta}{2}; \quad \frac{1}{k}\sum y_j = \frac{\hat{\theta}_{\text{mom}}(\vec{y})}{2}$$

$$\hat{\theta}_{\text{mom}}(\vec{y}) = \frac{2}{k}\sum y_j$$

$$\mathcal{M}\left\{(\hat{\theta}_{\text{mom}}(\vec{y}) - \theta)^2\right\} = \mathcal{M}\left\{\left(\frac{2}{k}\sum y_j - \theta\right)^2\right\} =$$

$$= \frac{\theta^2}{3k} = \frac{\theta^2}{k(k+2)} \cdot \frac{k+2}{3} > \frac{\theta^2}{k(k+2)}$$

Лекция 5. Особенности постановки и решения задач машинного обучения

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Обучение с учителем

Определение 5.1

Обучение с учителем (supervised learning) *— способ машинного обучения, в ходе которого испытуемая система принудительно обучается с помощью примеров "стимул-реакция"*

Обучение с учителем

Определение 5.1

Обучение с учителем (supervised learning) *— способ машинного обучения, в ходе которого испытуемая система принудительно обучается с помощью примеров "стимул-реакция"*

Входные данные:

- Признаковое описание объектов
- Матрица расстояний между объектами
- Временной ряд, изображение, текст

Обучение с учителем

Определение 5.1

Обучение с учителем (supervised learning) *— способ машинного обучения, в ходе которого испытуемая система принудительно обучается с помощью примеров "стимул-реакция"*

Входные данные:

- Признаковое описание объектов
- Матрица расстояний между объектами
- Временной ряд, изображение, текст

Выходные данные:

- Количественный признак – задача регрессии
- Номинальный признак – задача классификации

Обучение с подкреплением

Определение 5.2

Обучение с подкреплением (reinforcement learning) *— способ машинного обучения, в ходе которого в ходе которого испытуемая система (агент) обучается, взаимодействуя с некоторой средой.*

Обучение без учителя

Определение 5.3

Обучение без учителя (unsupervised learning) *— способ машинного обучения, в ходе которого система спонтанно обучается выполнять поставленную задачу без внешнего вмешательства*

Обучение без учителя

Определение 5.3

Обучение без учителя (unsupervised learning) *— способ машинного обучения, в ходе которого система спонтанно обучается выполнять поставленную задачу без внешнего вмешательства*

Решаемые задачи:

Обучение без учителя

Определение 5.3

Обучение без учителя (unsupervised learning) *— способ машинного обучения, в ходе которого система спонтанно обучается выполнять поставленную задачу без внешнего вмешательства*

Решаемые задачи:

- Кластеризация – разбиение выборки на непересекающиеся множества (кластеры) похожих объектов так, что объекты разных кластеров сильно отличаются.

Обучение без учителя

Определение 5.3

Обучение без учителя (unsupervised learning) *— способ машинного обучения, в ходе которого система спонтанно обучается выполнять поставленную задачу без внешнего вмешательства*

Решаемые задачи:

- Кластеризация – разбиение выборки на непересекающиеся множества (кластеры) похожих объектов так, что объекты разных кластеров сильно отличаются.
- Обнаружение аномалий – выделение данных, сильно отличающихся от типичных.

Обучение без учителя

Определение 5.3

Обучение без учителя (unsupervised learning) *— способ машинного обучения, в ходе которого система спонтанно обучается выполнять поставленную задачу без внешнего вмешательства*

Решаемые задачи:

- Кластеризация – разбиение выборки на непересекающиеся множества (кластеры) похожих объектов так, что объекты разных кластеров сильно отличаются.
- Обнаружение аномалий – выделение данных, сильно отличающихся от типичных.
- Сокращение размерности – представление исходных данных с большим количеством признаков в пространстве меньшей размерности с минимальными потерями информации.

Обучение без учителя

Определение 5.3

Обучение без учителя (unsupervised learning) *— способ машинного обучения, в ходе которого система спонтанно обучается выполнять поставленную задачу без внешнего вмешательства*

Решаемые задачи:

- Кластеризация – разбиение выборки на непересекающиеся множества (кластеры) похожих объектов так, что объекты разных кластеров сильно отличаются.
- Обнаружение аномалий – выделение данных, сильно отличающихся от типичных.
- Сокращение размерности – представление исходных данных с большим количеством признаков в пространстве меньшей размерности с минимальными потерями информации.
- Визуализация – преобразование данных для наглядного изображения их на плоскости.

Схема постановки и решения задач обучения с учителем

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.
- Выбирается параметрическая модель $a(\vec{x}, \vec{\theta})$

Недообучение и переобучение

Задача машинного обучения – задача экстраполяции данных из $\mathbb{X}_{\text{train}}$ в $\mathbb{X}$.
Алгоритм должен обладать обобщающей способностью.

Недообучение и переобучение

Задача машинного обучения – задача экстраполяции данных из $\mathbb{X}_{\text{train}}$ в $\mathbb{X}$.
Алгоритм должен обладать обобщающей способностью.

Насколько сложным должен быть алгоритм?

Недообучение и переобучение

Феномен Рунге

Пример. Пусть значения функции $y = \dfrac{1}{1+x^2}$ заданы в точках x_i, равномерно расположенных на промежутке от -5 до 5. Требуется найти полином $P_N(x) = \sum\limits_{n=0}^{N} A_n x^n$, наилучшим образом приближающий $y(x)$.

Недообучение и переобучение

Феномен Рунге

Пример. Пусть значения функции $y = \dfrac{1}{1+x^2}$ заданы в точках x_i, равномерно расположенных на промежутке от -5 до 5. Требуется найти полином $P_N(x) = \sum\limits_{n=0}^{N} A_n x^n$, наилучшим образом приближающий $y(x)$.

В качестве функционала качества можно взять

$$\mathcal{L}(N) = \text{MSE} \equiv \frac{1}{M}\sum_{i}^{M}(P_N(x_i) - y(x_i))^2$$

Недообучение и переобучение

Феномен Рунге

Пример. Пусть значения функции $y = \dfrac{1}{1+x^2}$ заданы в точках x_i, равномерно расположенных на промежутке от -5 до 5. Требуется найти полином $P_N(x) = \sum\limits_{n=0}^{N} A_n x^n$, наилучшим образом приближающий $y(x)$.

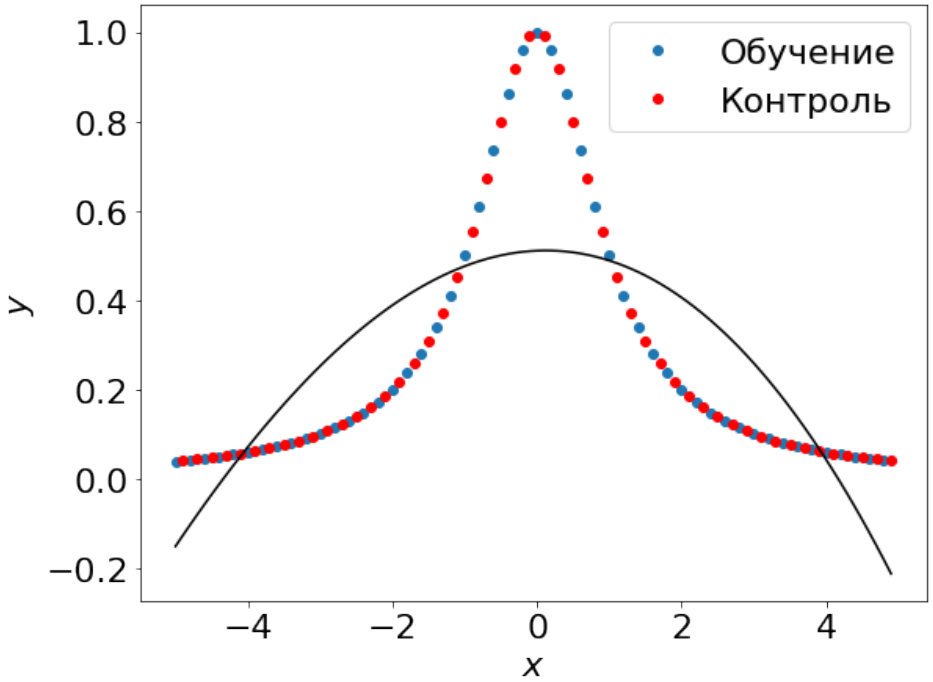
В качестве функционала качества можно взять

$$\mathcal{L}(N) = \text{MSE} \equiv \frac{1}{M}\sum_{i}^{M}(P_N(x_i) - y(x_i))^2$$

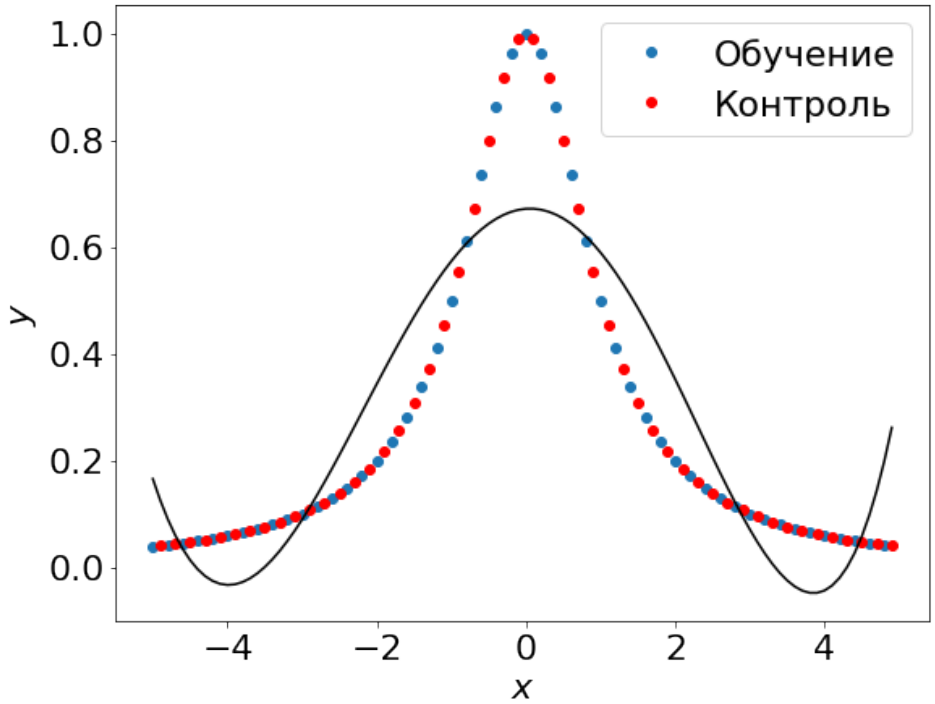
Степень полинома N определяет сложность модели и число коэффициентов.

Недообучение и переобучение

Феномен Рунге



$N = 3$

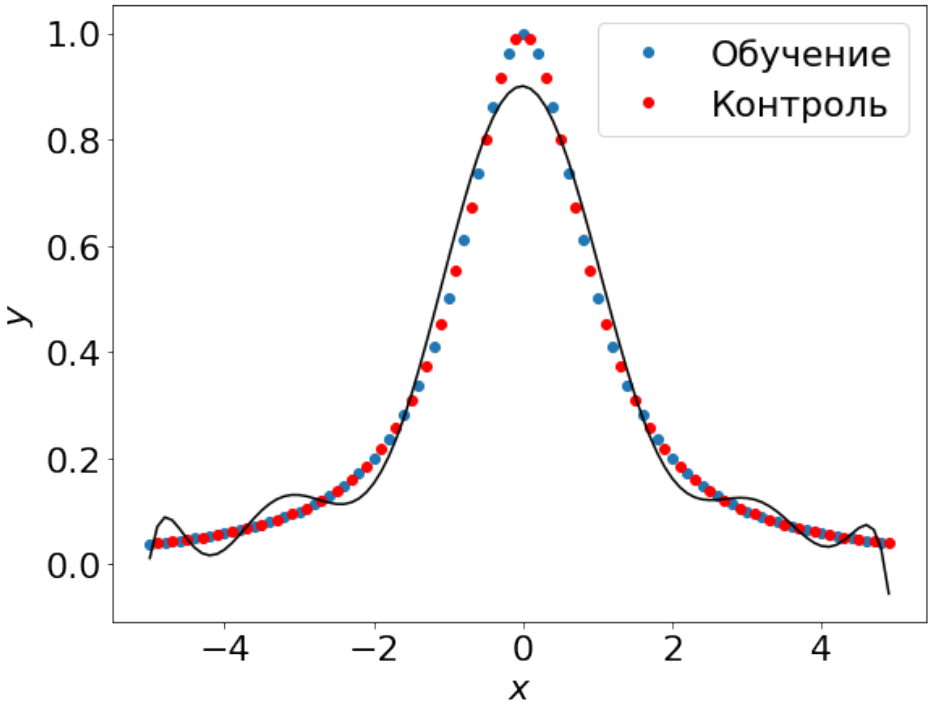


$N = 5$

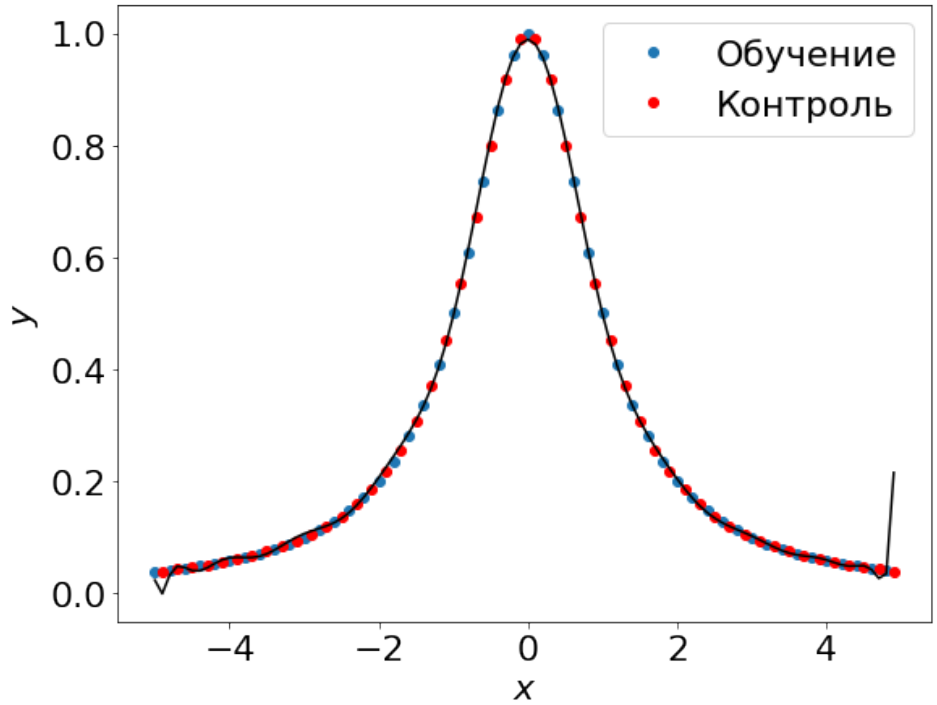
Недообучение: модель слишком простая.

Недообучение и переобучение

Феномен Рунге



$N = 10$

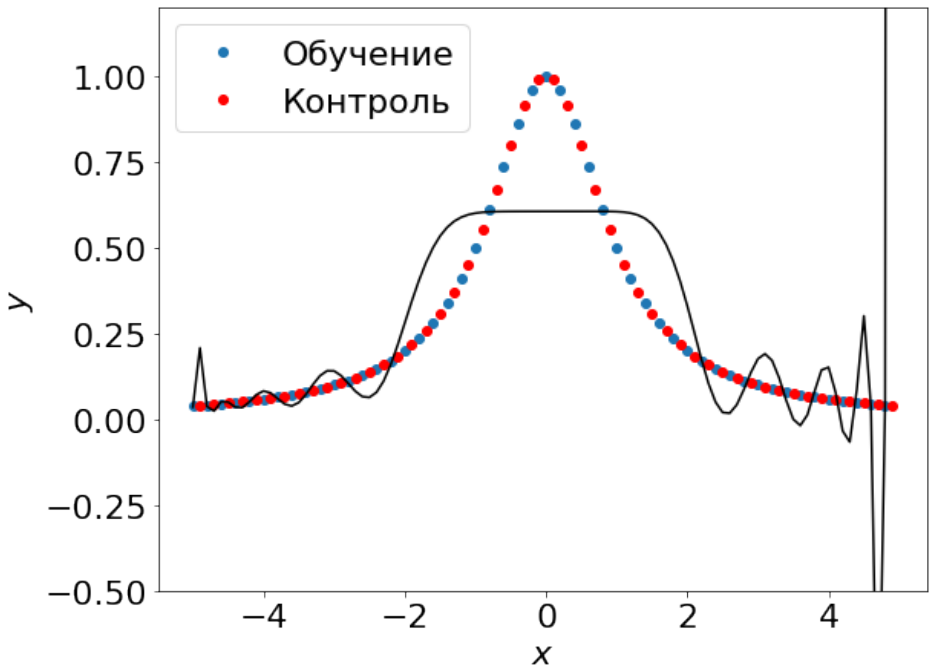


$N = 20$

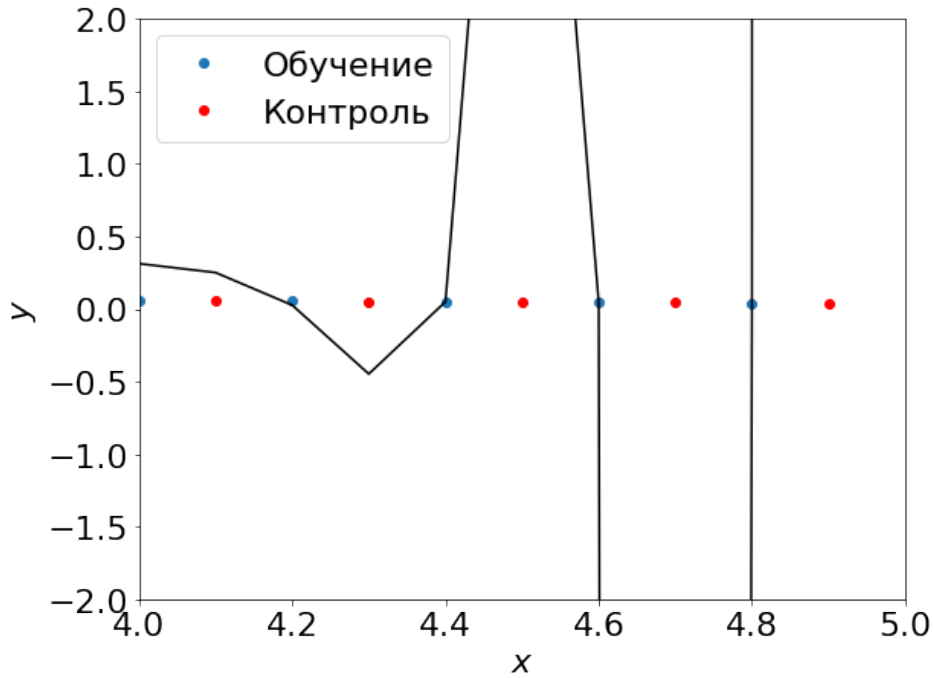
Модель удовлетворительно описывает функцию.

Недообучение и переобучение

Феномен Рунге



$N = 25$



$N = 25$

Переобучение: модель слишком сложная.

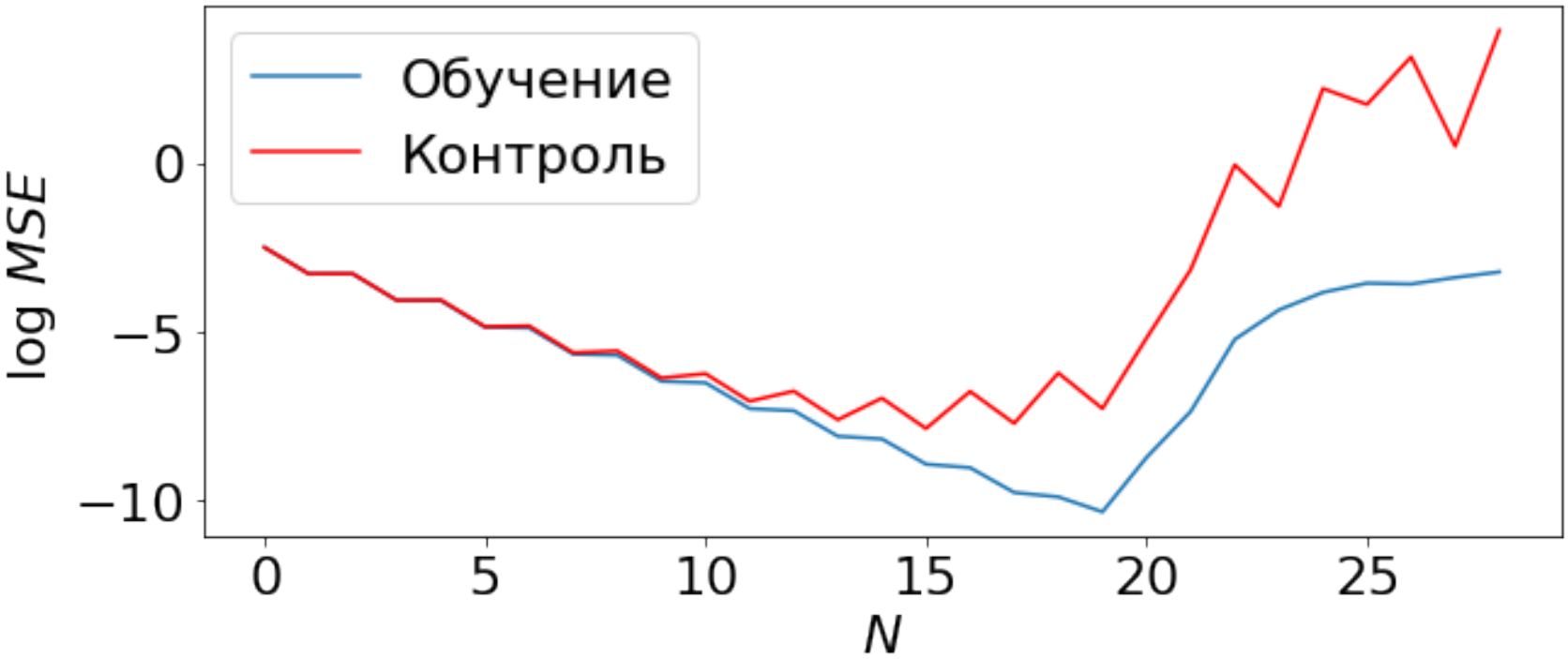
Недообучение и переобучение

Причина переобучения – слишком большое число параметров модели, подстроенных под конкретную обучающую выборку приводит к снижению обобщающей способности.

Недообучение и переобучение

Причина переобучения – слишком большое число параметров модели, подстроенных под конкретную обучающую выборку приводит к снижению обобщающей способности.

Способ выявить переобучение – разбить выборку на обучающую и контрольную и сравнить результаты.



Недообучение и переобучение

Как минимизировать вклад переобучения?

- Ввести регуляризацию в виде ограничений на параметры модели $\vec{\theta}$

Недообучение и переобучение

Как минимизировать вклад переобучения?

- Ввести регуляризацию в виде ограничений на параметры модели $\vec{\theta}$
- Разделить обучающую выборку на независимые части:
  $\mathbb{X}_{\text{train}}$ – тренировочная подвыборка и
  $\mathbb{X}_{\text{val}}$ – валидационная подвыборка, причем $\mathbb{X}_{\text{train}} \cap \mathbb{X}_{\text{val}} = \emptyset$.

Недообучение и переобучение

Как минимизировать вклад переобучения?

- Ввести регуляризацию в виде ограничений на параметры модели $\vec{\theta}$
- Разделить обучающую выборку на независимые части:

  $\mathbb{X}_{\text{train}}$ – тренировочная подвыборка и

  $\mathbb{X}_{\text{val}}$ – валидационная подвыборка, причем $\mathbb{X}_{\text{train}} \cap \mathbb{X}_{\text{val}} = \emptyset$.

  Обучать модель на $\mathbb{X}_{\text{train}}$, добиваясь минимальной ошибки на $\mathbb{X}_{\text{train}}$

Недообучение и переобучение

Как минимизировать вклад переобучения?

- Ввести регуляризацию в виде ограничений на параметры модели $\vec{\theta}$
- Разделить обучающую выборку на независимые части:
  $\mathbb{X}_{\text{train}}$ – тренировочная подвыборка и
  $\mathbb{X}_{\text{val}}$ – валидационная подвыборка, причем $\mathbb{X}_{\text{train}} \cap \mathbb{X}_{\text{val}} = \emptyset$.
  Обучать модель на $\mathbb{X}_{\text{train}}$, добиваясь минимальной ошибки на $\mathbb{X}_{\text{train}}$

  Неявно $\mathbb{X}_{\text{val}}$ всё равно влияет на результаты обучения, поэтому на новых данных, не включенных ни в $\mathbb{X}_{\text{train}}$, ни в $\mathbb{X}_{\text{val}}$, ожидается увеличение ошибки.

Недообучение и переобучение

Эмпирические способы разбиения обучающей выборки

1. **Hold-Out**
   - Статическое разбиение обучающей выборки на $\mathbb{X}_{\text{train}}$ и $\mathbb{X}_{\text{ho}}$
   - Определяются параметры $\vec{\theta}$: $\mathcal{L}(a(\vec{\theta}), \vec{x} \in \mathbb{X}_{\text{train}}) \underset{\vec{\theta}}{\rightarrow} \min$
   - Критерий качества – это значение $\mathcal{L}(a(\vec{\theta}), \vec{x} \in \mathbb{X}_{\text{ho}})$
2. **Leave-One-Out**
   - Каждый элемент обучающей выборки по разу участвует в контроле
   - Определяются параметры $\vec{\theta}_i$: $\mathcal{L}(a(\vec{\theta}_i), \vec{x} \in \mathbb{X}_{\text{train}} \setminus \{\vec{x}_i\}) \underset{\vec{\theta}_i}{\rightarrow} \min$
   - Критерий качества – это значение $\frac{1}{L}\sum_{i=1}^{L} \mathcal{L}(a(\vec{\theta}_i), \vec{x}_i)$
3. **Cross-Validation**
   - Обучающая выборка N раз разбивается на $\mathbb{X}_{\text{train},i}$ и $\mathbb{X}_{\text{val},i}$
   - Определяются параметры $\vec{\theta}_i$ $\mathcal{L}(a(\vec{\theta}_i), \vec{x} \in \mathbb{X}_{\text{train},i}) \underset{\vec{\theta}_i}{\rightarrow} \min$
   - Критерий качества – это значение $\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(a(\vec{\theta}_i), \vec{x} \in \mathbb{X}_{\text{val},i})$

Гиперпараметры

Определение 5.4

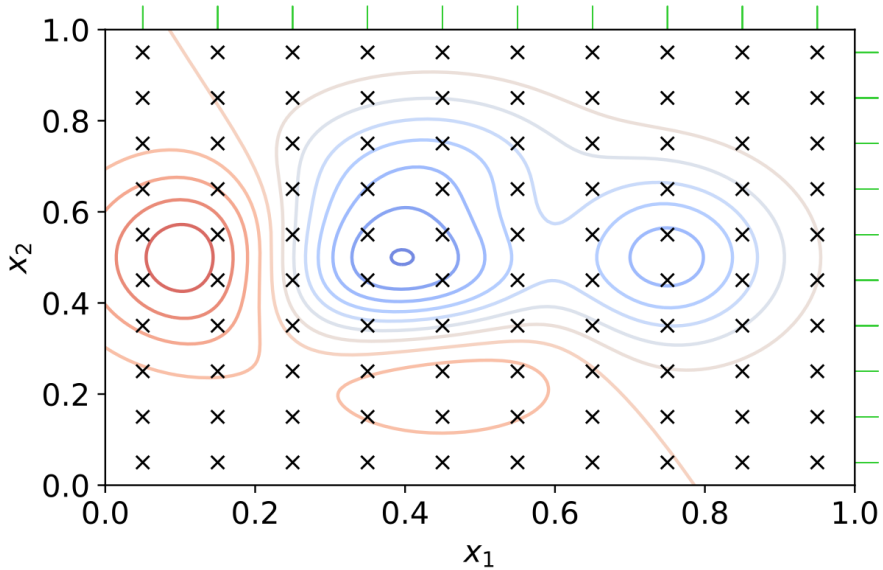
Гиперпараметры *— параметры алгоритма, значения которых устанавливаются перед запуском процесса обучения*

Гиперпараметры

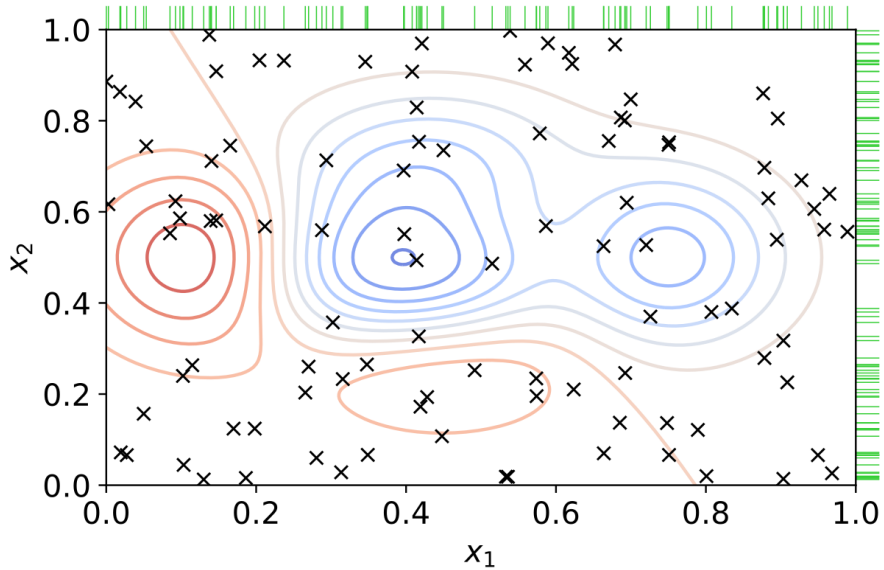
Определение 5.4

Гиперпараметры *— параметры алгоритма, значения которых устанавливаются перед запуском процесса обучения*

Поиск значения гиперпараметров



Поиск по решетке



Случайный поиск

Практикум. Недообучение и переобучение. Формирование обучающих выборок

Jupyter notebook “Недообучение и переобучение. Феномен Рунге”:
`https://colab.research.google.com/drive/1pWJEcY1sPTEct0iC4VqGe1rx3wrnBvpf`

Jupyter notebook “Обучающие выборки. Валидация. Кросс-валидация”:
`https://colab.research.google.com/drive/1_u8-gkyx1THBp8Oe9pI7vpWCTGGu6rgM`

Лекция 6. Линейные методы регрессии и классификации

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Схема постановки и решения задач обучения с учителем

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.
- Выбирается параметрическая модель $a(\vec{x}, \vec{\theta})$
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума $\mathcal{L}$.

Задача линейной регрессии

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \mathcal{R}$$

- Выбирается линейная модель: $a(\vec{x}, \vec{\theta}) = (\vec{x}; \vec{\theta}) = \sum_{j=1}^{n} x_j \theta_j$
- Квадратичная функция потерь: $\mathcal{L}(a, \vec{x}, y) = (a - y)^2$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L} \sum_{i=1}^{L} \mathcal{L}(a(\vec{x}_i, \vec{\theta}), \vec{x}_i, y_i) = \frac{1}{L} \sum_{i=1}^{L} \left((\vec{x}_i; \vec{\theta}) - y_i \right)^2 \underset{\vec{\theta}}{\to} \min$$

- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}) = \frac{1}{K} \sum_{i=1}^{K} \left((\vec{x}_i; \vec{\theta}) - y_i \right)^2$$

Задача линейной бинарной классификации

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \{-1; 1\}$$

- Выбирается линейная модель: $a(\vec{x}, \vec{\theta}) = \text{sgn}(\vec{x}; \vec{\theta}) = \text{sgn} \sum_{j=1}^{n} x_j \theta_j$
- Бинарная функция потерь: $\mathcal{L}(a, \vec{x}, y) = [ay < 0]$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L} \sum_{i=1}^{L} \left[a(\vec{x}_i; \vec{\theta}) y_i < 0 \right] = \frac{1}{L} \sum_{i=1}^{L} \left[\underbrace{(\vec{x}_i; \vec{\theta}) y_i}_{\text{отступ объекта}} < 0 \right] \underset{\vec{\theta}}{\to} \min$$

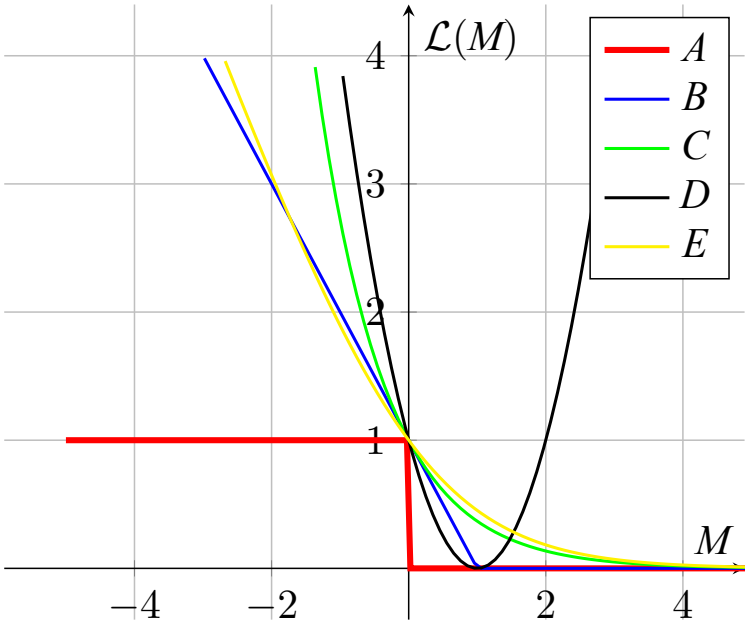
- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}) = \frac{1}{K} \sum_{i=1}^{K} \left[(\vec{x}_i; \vec{\theta}) y_i < 0 \right]$$

Задача линейной бинарной классификации

Идея: заменить пороговую функцию потерь непрерывной функцией

Отступ объекта (margin) $M = (\vec{x}; \vec{\theta}\,)y$



- $A : [M < 0]$ – пороговая
- $B : (1 - M)_+$ – SVM
- $C : e^{-M}$ – экспоненциальная
- $D : (1 - M)^2$ – квадратичная
- $E : \log_2(1 + e^{-M})$ – логарифмическая

Решение задачи линейной регрессии

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}(a(\vec{x}_i,\vec{\theta}),\vec{x}_i,y_i) = \frac{1}{L}\sum_{i=1}^{L}\left((\vec{x}_i;\vec{\theta}) - y_i\right)^2 \underset{\vec{\theta}}{\to} \min$$

Решение задачи линейной регрессии

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}(a(\vec{x}_i,\vec{\theta}),\vec{x}_i,y_i) = \frac{1}{L}\sum_{i=1}^{L}\Big((\vec{x}_i;\vec{\theta}) - y_i\Big)^2 \underset{\vec{\theta}}{\to} \min$$

Пусть $\mathcal{X} = \{\vec{x}_1,\vec{x}_2,\ldots\vec{x}_L\}$ и $\vec{y} = \{y_1,y_2,\ldots,y_L\}$, тогда

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\left\|\mathcal{X}\vec{\theta} - \vec{y}\right\|_2^2 \underset{\vec{\theta}}{\to} \min$$

Решение задачи линейной регрессии

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}(a(\vec{x}_i,\vec{\theta}),\vec{x}_i,y_i) = \frac{1}{L}\sum_{i=1}^{L}\Big((\vec{x}_i;\vec{\theta}) - y_i\Big)^2 \underset{\vec{\theta}}{\to} \min$$

Пусть $\mathcal{X} = \{\vec{x}_1,\vec{x}_2,\ldots\vec{x}_L\}$ и $\vec{y} = \{y_1,y_2,\ldots,y_L\}$, тогда

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\left\|\mathcal{X}\vec{\theta} - \vec{y}\right\|_2^2 \underset{\vec{\theta}}{\to} \min$$

$$\nabla_{\vec{\theta}} Q_{\text{train}}(\vec{\theta}) = 0 \Rightarrow \mathcal{X}^T(\mathcal{X}\vec{\theta} - \vec{y}) = 0$$

Решение задачи линейной регрессии

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}(a(\vec{x}_i,\vec{\theta}),\vec{x}_i,y_i) = \frac{1}{L}\sum_{i=1}^{L}\left((\vec{x}_i;\vec{\theta}) - y_i\right)^2 \underset{\vec{\theta}}{\to} \min$$

Пусть $\mathcal{X} = \{\vec{x}_1,\vec{x}_2,\ldots\vec{x}_L\}$ и $\vec{y} = \{y_1,y_2,\ldots,y_L\}$, тогда

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\left\|\mathcal{X}\vec{\theta} - \vec{y}\right\|_2^2 \underset{\vec{\theta}}{\to} \min$$

$$\nabla_{\vec{\theta}} Q_{\text{train}}(\vec{\theta}) = 0 \Rightarrow \mathcal{X}^T(\mathcal{X}\vec{\theta} - \vec{y}) = 0$$

Формальное решение $\vec{\theta}_0 = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T\vec{y}$

Решение задачи линейной регрессии

Проблема мультиколлинеарности

Определение 6.1

Мультиколлинеарность *— наличие линейной зависимости между объясняющими переменными (факторами) регрессионной модели*

Решение задачи линейной регрессии

Проблема мультиколлинеарности

Определение 6.1

Мультиколлинеарность *— наличие линейной зависимости между объясняющими переменными (факторами) регрессионной модели*

$$\mathcal{X} = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_L\} \text{ и } \vec{y} = \{y_1, y_2, \ldots, y_L\}$$

Решение задачи линейной регрессии

Проблема мультиколлинеарности

Определение 6.1

Мультиколлинеарность *— наличие линейной зависимости между объясняющими переменными (факторами) регрессионной модели*

$$\mathcal{X} = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_L\} \text{ и } \vec{y} = \{y_1, y_2, \ldots, y_L\}$$

Столбцы матрицы $\mathcal{X}$ линейно зависимы: $\exists \vec{\theta}_1 \neq \vec{0} : \mathcal{X}\vec{\theta}_1 = \vec{0}$

Решение задачи линейной регрессии

Проблема мультиколлинеарности

Определение 6.1

Мультиколлинеарность *— наличие линейной зависимости между объясняющими переменными (факторами) регрессионной модели*

$$\mathcal{X} = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_L\} \text{ и } \vec{y} = \{y_1, y_2, \ldots, y_L\}$$

Столбцы матрицы $\mathcal{X}$ линейно зависимы: $\exists \vec{\theta}_1 \neq \vec{0} : \mathcal{X}\vec{\theta}_1 = \vec{0}$

Найдено решение $\vec{\theta}_0 = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T\vec{y}$
$\vec{\theta} = \vec{\theta}_0 + \xi\vec{\theta}_1$ – тоже решение задачи при любом ξ

Решение задачи линейной регрессии

Регуляризация

Определение 6.2

Регуляризация *— метод добавления некоторых дополнительных ограничений к условию с целью решить некорректно поставленную задачу или предотвратить переобучение. Эта информация часто имеет вид штрафа за сложность модели. Например, это могут быть ограничения гладкости результирующей функции или ограничения по норме векторного пространства*

Решение задачи линейной регрессии

Регуляризация

Определение 6.2

Регуляризация *— метод добавления некоторых дополнительных ограничений к условию с целью решить некорректно поставленную задачу или предотвратить переобучение. Эта информация часто имеет вид штрафа за сложность модели. Например, это могут быть ограничения гладкости результирующей функции или ограничения по норме векторного пространства*

$$\text{Пусть одновременно } Q_{\text{train}}(\vec{\theta}) \underset{\vec{\theta}}{\to} \min \text{ и } \left\|\vec{\theta}\right\|_P \underset{\vec{\theta}}{\to} \min$$

$$\text{Норма вектора } L_P : \|\vec{x}\|_P = \left(\sum |x_i|^P\right)^{\frac{1}{P}}. \text{ Часто применяются } L_1 \text{ и } L_2$$

$$Q_{\text{train}}(\vec{\theta}) + \lambda \left\|\vec{\theta}\right\|_P \underset{\vec{\theta}}{\to} \min$$

Решение задачи линейной регрессии

Примеры регуляризации

- Регуляризация по Тихонову (ridge regression)

$$Q_{\text{train}}(\vec{\theta}) + \alpha|\vec{\theta}|^2 \underset{\vec{\theta}}{\to} \min$$

$$\vec{\theta}_0 = (\mathcal{X}^T\mathcal{X} + \epsilon\mathbf{I})^{-1}\mathcal{X}^T\vec{y}$$

- Регуляризация через манхэттенское расстояние (lasso regression, Least Absolute Shrinkage and Selection Operator)

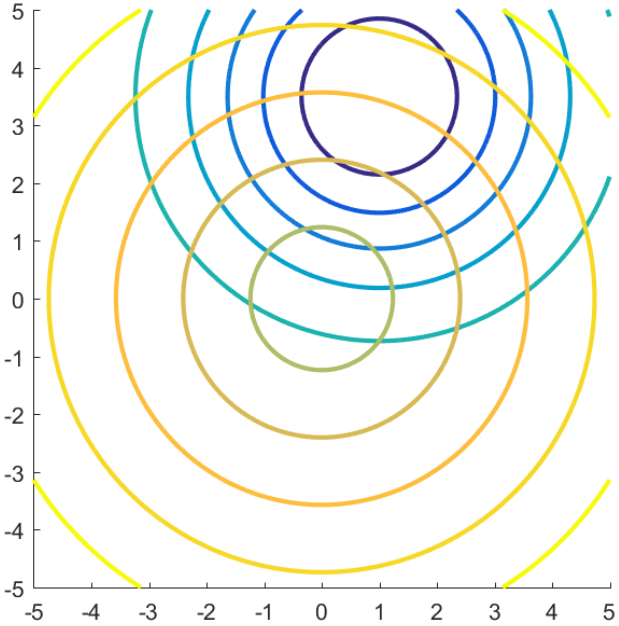
$$Q_{\text{train}}(\vec{\theta}) + \alpha\sum_{j=1}^{n}|\theta_j| \underset{\vec{\theta}}{\to} \min$$

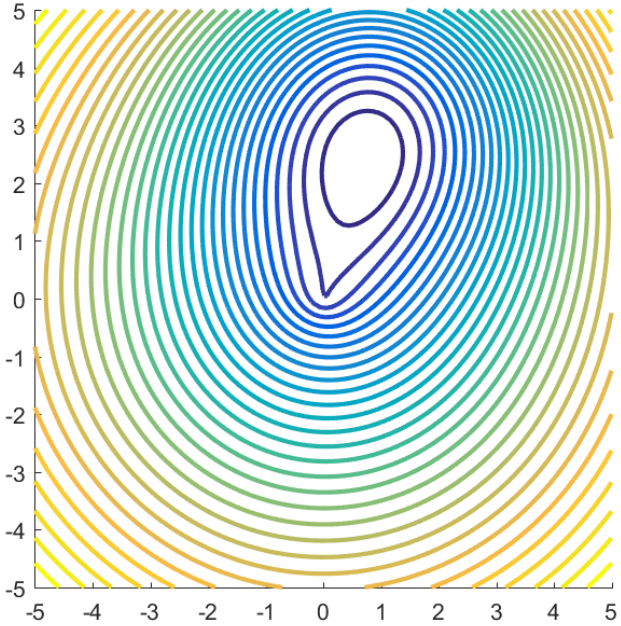
- Elasticnet regression

$$Q_{\text{train}}(\vec{\theta}) + \alpha\sum_{j=1}^{n}|\theta_j| + \beta|\vec{\theta}|^2 \underset{\vec{\theta}}{\to} \min$$

Решение задачи линейной регрессии

Регуляризация и отбор признаков

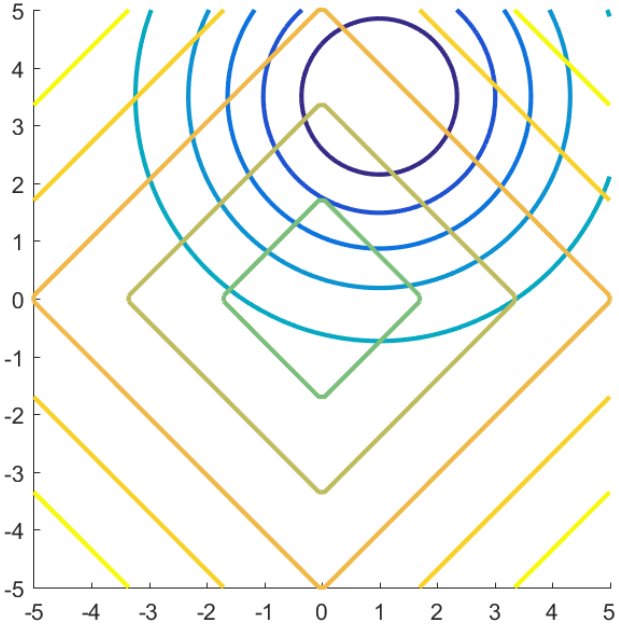


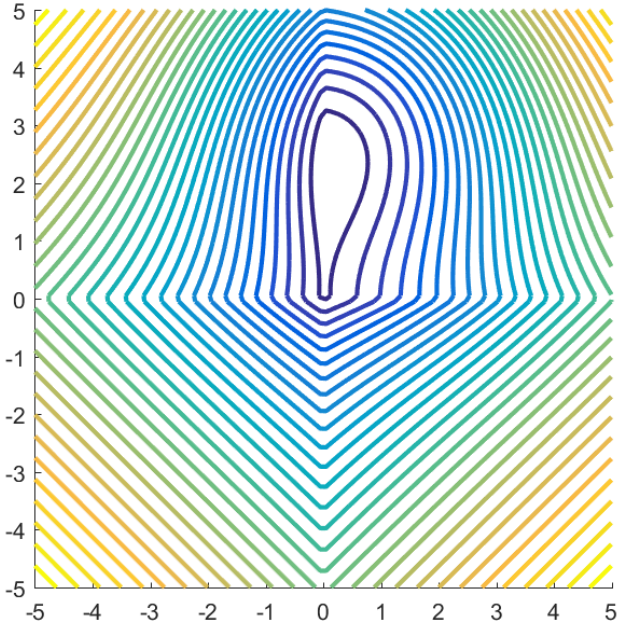


Ridge-регрессия ($P = 2$), отбор признаков не происходит.

Решение задачи линейной регрессии

Регуляризация и отбор признаков

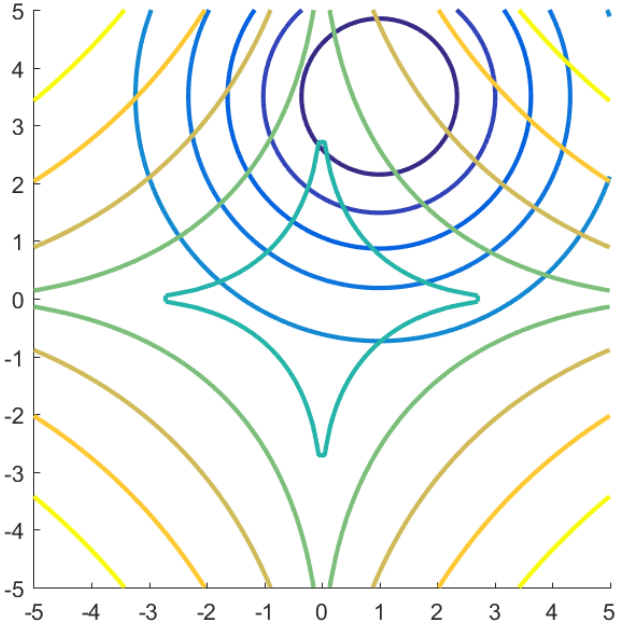


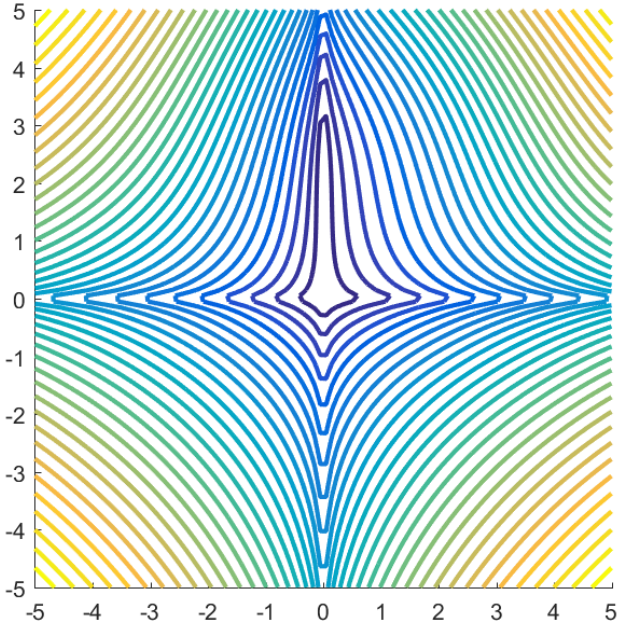


Отбор признаков при lasso-регрессии ($P = 1$).

Решение задачи линейной регрессии

Регуляризация и отбор признаков

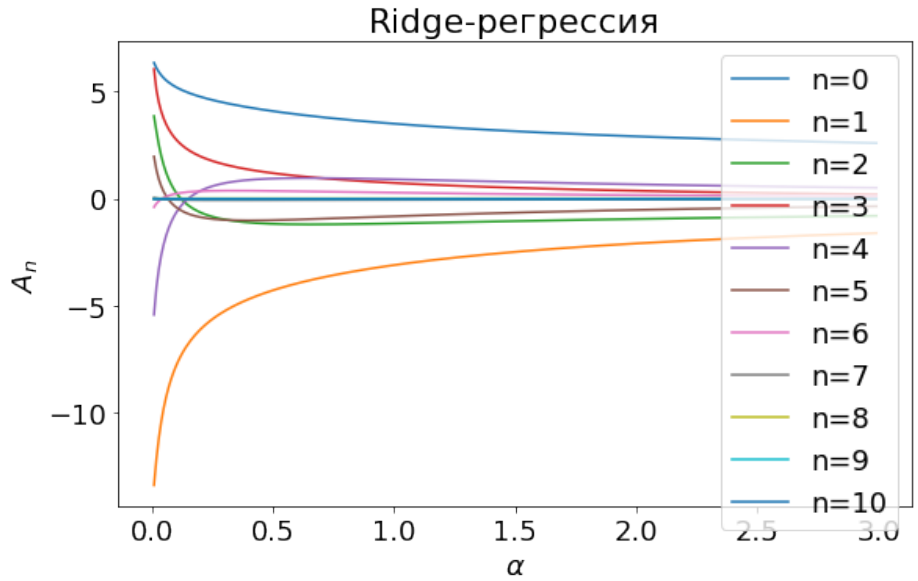


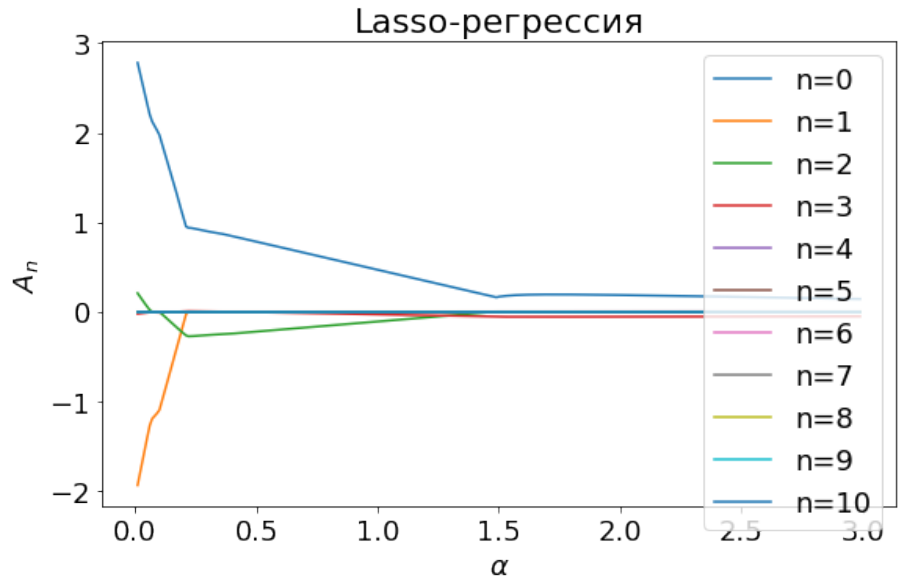


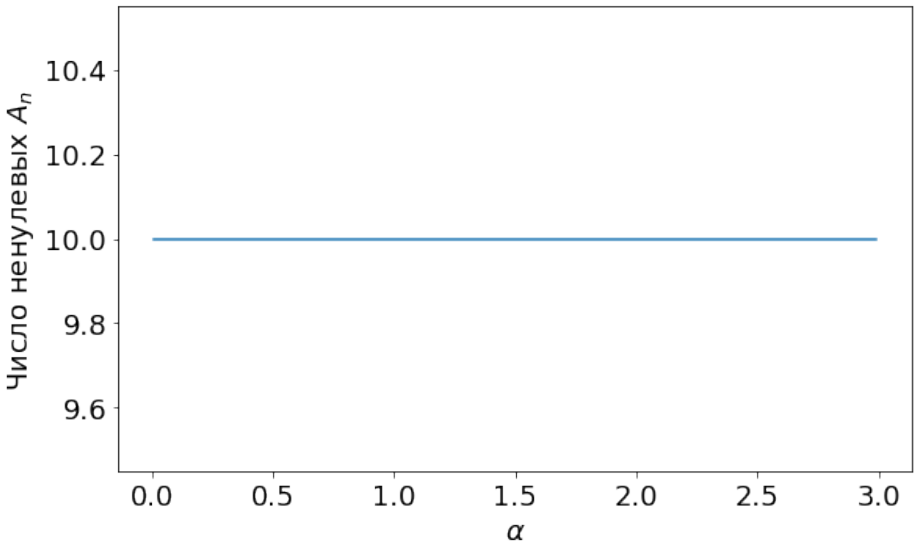
Отбор признаков при $P = 0.5$.

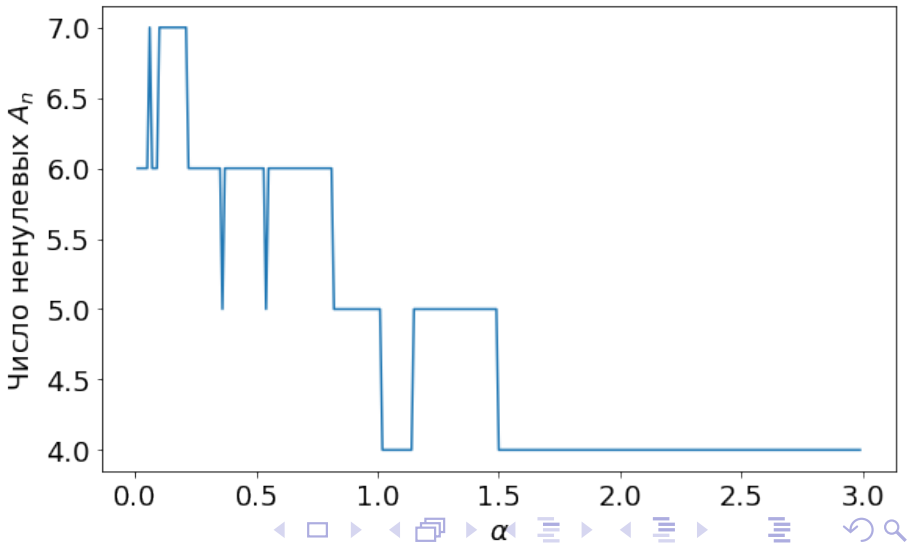
Решение задачи линейной регрессии

Регуляризация и отбор признаков









Вероятностный подход к задаче линейной регрессии

Модель: $\vec{y} = \mathcal{X}\vec{\theta} + \vec{\epsilon}; \quad \epsilon_j \sim \mathcal{N}(0, \sigma^2)$

Предполагается, что неизвестные коэффициенты $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_n\}$ распределены нормально: $w(\vec{\theta}) = \dfrac{\lambda}{(2\pi)^{n/2}} \exp\left(-\dfrac{\lambda^2}{2}|\vec{\theta}|^2\right)$

Вероятностный подход к задаче линейной регрессии

Метод максимального правдоподобия

$$\ln L(\vec{\theta}) = \ln w(\vec{y}|\mathcal{X}, \vec{\theta}) = \sum_{i=1}^{L} \left[-\frac{1}{2}\ln(2\pi\sigma^2) - \frac{\left(y_i - \sum\limits_{j=1}^{n} \mathcal{X}_{ij}\theta_j\right)^2}{2\sigma^2} \right]$$

$$\frac{1}{2\sigma^2}\sum_{i=1}^{L}\left(y_i - \sum_{j=1}^{n}\mathcal{X}_{ij}\theta_j\right)^2 \to \min$$

В матричной записи: $Q_{\text{train}}(\vec{\theta}) = \left\|\mathcal{X}\vec{\theta} - \vec{y}\right\|_2^2 \to \min$

Вероятностный подход к задаче линейной регрессии

Метод апостериорной плотности распределения

$$\ln L(\vec{\theta}) + \ln w(\vec{\theta}) =$$

$$= -\sum_{i=1}^{L} \frac{1}{2} \ln(2\pi\sigma^2) - \sum_{i=1}^{L} \frac{\left(y_i - \sum\limits_{j=1}^{n} x_{ij}\theta_j\right)^2}{2\sigma^2} + \ln \frac{\lambda}{(2\pi)^{l/2}} - \frac{\lambda^2}{2}|\vec{\theta}|^2$$

$$\underbrace{\frac{1}{2\sigma^2} \sum_{i=1}^{L} \left(y_i - \sum_{j=1}^{n} x_{ij}\theta_j\right)^2}_{\text{Квадратичная ошибка}} + \underbrace{\frac{\lambda^2}{2}|\vec{\theta}\,|^2}_{L_2\text{-регуляризация}} \to \min$$

Вероятностный подход к задаче линейной регрессии

Метод апостериорной плотности распределения

$$\ln L(\vec{\theta}) + \ln w(\vec{\theta}) =$$

$$= -\sum_{i=1}^{L} \frac{1}{2} \ln(2\pi\sigma^2) - \sum_{i=1}^{L} \frac{\left(y_i - \sum\limits_{j=1}^{n} x_{ij}\theta_j\right)^2}{2\sigma^2} + \ln \frac{\lambda}{(2\pi)^{l/2}} - \frac{\lambda^2}{2}|\vec{\theta}|^2$$

$$\underbrace{\frac{1}{2\sigma^2} \sum_{i=1}^{L} \left(y_i - \sum_{j=1}^{n} x_{ij}\theta_j\right)^2}_{\text{Квадратичная ошибка}} + \underbrace{\frac{\lambda^2}{2}|\vec{\theta}\,|^2}_{L_2\text{-регуляризация}} \to \min$$

В матричной записи: $Q_{\text{train}}(\vec{\theta}\,) = \left\| X\vec{\theta} - \vec{y} \right\|_2^2 + (\sigma\lambda)^2 \left\| \vec{\theta} \right\|_2^2 \to \min$

Вероятностный подход к задаче линейной регрессии

Метод апостериорной плотности распределения

Если неизвестные коэффициенты $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_n\}$ распределены согласно распределению Лапласа: $w(\vec{\theta_i}) = \frac{\lambda}{2} \exp(-|\theta_i|)$

Вероятностный подход к задаче линейной регрессии

Метод апостериорной плотности распределения

Если неизвестные коэффициенты $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_n\}$ распределены согласно распределению Лапласа: $w(\vec{\theta_i}) = \frac{\lambda}{2} \exp(-|\theta_i|)$

$$\underbrace{\frac{1}{2\sigma^2} \sum_{i=1}^{L} \left(y_i - \sum_{j=1}^{n} x_{ij} \theta_j \right)^2}_{\text{Квадратичная ошибка}} + \underbrace{\lambda \sum_{j=1}^{n} |\theta_j|}_{L_1\text{-регуляризация}} \to \min$$

Вероятностный подход к задаче линейной регрессии

Метод апостериорной плотности распределения

Если неизвестные коэффициенты $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_n\}$ распределены согласно распределению Лапласа: $w(\vec{\theta}_i) = \frac{\lambda}{2} \exp(-|\theta_i|)$

$$\underbrace{\frac{1}{2\sigma^2} \sum_{i=1}^{L} \left(y_i - \sum_{j=1}^{n} \mathcal{X}_{ij} \theta_j \right)^2}_{\text{Квадратичная ошибка}} + \underbrace{\lambda \sum_{j=1}^{n} |\theta_j|}_{L_1\text{-регуляризация}} \to \min$$

В матричной записи: $Q_{\text{train}}(\vec{\theta}) = \left\| \mathcal{X}\vec{\theta} - \vec{y} \right\|_2^2 + \sigma^2 \lambda \left\| \vec{\theta} \right\|_1 \to \min$

Вероятностный подход к задаче классификации

Эксперимент с монеткой: метод максимального правдоподобия

Необходимо оценить вероятность выпадения орла, имея данные эксперимента $\vec{y} = \{y_1, y_2, \ldots, y_N\}$.

Вероятностный подход к задаче классификации

Эксперимент с монеткой: метод максимального правдоподобия

Необходимо оценить вероятность выпадения орла, имея данные эксперимента $\vec{y} = \{y_1, y_2, \ldots, y_N\}$.

$$p(y_j|q) = q^{y_j}(1-q)^{1-y_j}$$

Вероятностный подход к задаче классификации

Эксперимент с монеткой: метод максимального правдоподобия

Необходимо оценить вероятность выпадения орла, имея данные эксперимента $\vec{y} = \{y_1, y_2, \ldots, y_N\}$.

$$p(y_j|q) = q^{y_j}(1-q)^{1-y_j}$$

$$\mathcal{L}_{\log} = -\frac{\ln L}{N} = -\sum_{j=1}^{N} \ln p(y_j|q) =$$

$$= \underbrace{-\sum_{j=1}^{N} (y_j \ln q + (1-y_j)\ln(1-q))}_{\text{Логистическая функция потерь}} \to \min$$

Вероятностный подход к задаче классификации

Эксперимент с монеткой: метод максимального правдоподобия

Необходимо оценить вероятность выпадения орла, имея данные эксперимента $\vec{y} = \{y_1, y_2, \ldots, y_N\}$.

$$p(y_j|q) = q^{y_j}(1-q)^{1-y_j}$$

$$\mathcal{L}_{\log} = -\frac{\ln L}{N} = -\sum_{j=1}^{N} \ln p(y_j|q) =$$

$$= \underbrace{-\sum_{j=1}^{N} (y_j \ln q + (1-y_j)\ln(1-q))}_{\text{Логистическая функция потерь}} \to \min$$

$$\hat{q}_{\text{mle}} = \frac{1}{N}\sum_{j=1}^{N} y_j$$

Вероятностный подход к задаче классификации

Перекрестная энтропия

Объект, обладающий признаками $\vec{x}$, относится к j-му классу с вероятностью $p_j; j = 1,2,\ldots,s$.

Алгоритм на основе $\vec{x}$ определяет эту вероятность как q_j.

При $N \gg 1$ испытаниях j-й класс реализовался $n_j = Np_j$ раз.

Вероятностный подход к задаче классификации

Перекрестная энтропия

Объект, обладающий признаками $\vec{x}$, относится к j-му классу с вероятностью $p_j; j = 1,2,\ldots,s$.

Алгоритм на основе $\vec{x}$ определяет эту вероятность как q_j.

При $N \gg 1$ испытаниях j-й класс реализовался $n_j = Np_j$ раз.

$$L(q_1,q_2,\ldots,q_s) = p(n_1,n_2,\ldots,n_s|q_1,q_2,\ldots,q_s) = \prod_{j=1}^{s} q_j^{Np_j}$$

Вероятностный подход к задаче классификации

Перекрестная энтропия

Объект, обладающий признаками $\vec{x}$, относится к j-му классу с вероятностью $p_j; j = 1,2,\ldots,s$.

Алгоритм на основе $\vec{x}$ определяет эту вероятность как q_j.

При $N \gg 1$ испытаниях j-й класс реализовался $n_j = Np_j$ раз.

$$L(q_1,q_2,\ldots,q_s) = p(n_1,n_2,\ldots,n_s|q_1,q_2,\ldots,q_s) = \prod_{j=1}^{s} q_j^{Np_j}$$

$$-\frac{1}{N}\ln L(q_1,q_2,\ldots,q_s) = -\sum_{j=1}^{s} p_j \ln q_j = H(p,q) \to \min$$

Вероятностный подход к задаче классификации

Перекрестная энтропия

Объект, обладающий признаками $\vec{x}$, относится к j-му классу с вероятностью $p_j; j = 1,2,\ldots,s$.
Алгоритм на основе $\vec{x}$ определяет эту вероятность как q_j.
При $N \gg 1$ испытаниях j-й класс реализовался $n_j = Np_j$ раз.

$$L(q_1,q_2,\ldots,q_s) = p(n_1,n_2,\ldots,n_s|q_1,q_2,\ldots,q_s) = \prod_{j=1}^{s} q_j^{Np_j}$$

$$-\frac{1}{N}\ln L(q_1,q_2,\ldots,q_s) = -\sum_{j=1}^{s} p_j \ln q_j = H(p,q) \to \min$$

Определение 6.3

Перекрестная энтропия (cross entropy) — *функция между двумя распределениями вероятностей* $H(p,q) = -\sum p_j \ln q_j$

Вероятностный подход к задаче классификации

Минимум перекрестной энтропии

$$H(p,q) = -\sum p_j \ln q_j$$

$$\frac{\partial H}{\partial q_j} = 0; \quad \sum q_j = 1$$

$$\xi = -\sum p_j \ln q_j + \lambda\left(1 - \sum q_j\right)$$

Вероятностный подход к задаче классификации

Минимум перекрестной энтропии

$$H(p,q) = -\sum p_j \ln q_j$$

$$\frac{\partial H}{\partial q_j} = 0; \quad \sum q_j = 1$$

$$\xi = -\sum p_j \ln q_j + \lambda\left(1 - \sum q_j\right)$$

$$\frac{\partial \xi}{\partial q_j} = 0; \quad -\frac{p_j}{q_j} - \lambda = 0; \quad q_j = -\frac{p_j}{\lambda}$$

Вероятностный подход к задаче классификации

Минимум перекрестной энтропии

$$H(p,q) = -\sum p_j \ln q_j$$

$$\frac{\partial H}{\partial q_j} = 0; \quad \sum q_j = 1$$

$$\xi = -\sum p_j \ln q_j + \lambda\left(1 - \sum q_j\right)$$

$$\frac{\partial \xi}{\partial q_j} = 0; \quad -\frac{p_j}{q_j} - \lambda = 0; \quad q_j = -\frac{p_j}{\lambda}$$

$$\sum q_j = -\sum p_j/\lambda = -1/\lambda; \quad \lambda = -1$$

Вероятностный подход к задаче классификации

Минимум перекрестной энтропии

$$H(p,q) = -\sum p_j \ln q_j$$

$$\frac{\partial H}{\partial q_j} = 0; \quad \sum q_j = 1$$

$$\xi = -\sum p_j \ln q_j + \lambda\left(1 - \sum q_j\right)$$

$$\frac{\partial \xi}{\partial q_j} = 0; \quad -\frac{p_j}{q_j} - \lambda = 0; \quad q_j = -\frac{p_j}{\lambda}$$

$$\sum q_j = -\sum p_j/\lambda = -1/\lambda; \quad \lambda = -1$$

$$q_j = p_j$$

Вероятностный подход к задаче классификации

Объект определенного класса

$$\mathcal{L} = -\sum_{j=1}^{s} p_j \ln q_j$$

Вероятностный подход к задаче классификации

Объект определенного класса

$$\mathcal{L} = -\sum_{j=1}^{s} p_j \ln q_j$$

Если точно известно, что объект принадлежит классу t,
то $p_j = \delta(j,t)$ и $\mathcal{L} = -\ln q_t$

Вероятностный подход к задаче классификации

Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1 - y) \ln(1 - q)$

Вероятностный подход к задаче классификации

Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1 - y) \ln(1 - q)$
- Модель вероятности:

$$q = \text{sigmoid}(\vec{x}; \vec{\theta}) \equiv \frac{1}{1 + e^{-(\vec{x};\vec{\theta})}}$$

Вероятностный подход к задаче классификации

Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1-y)\ln(1-q)$
- Модель вероятности:

$$q = \text{sigmoid}(\vec{x}; \vec{\theta}) \equiv \frac{1}{1+e^{-(\vec{x};\vec{\theta})}} \text{ и } 1-q = \frac{1}{1+e^{(\vec{x};\vec{\theta})}}$$

Вероятностный подход к задаче классификации

Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1-y)\ln(1-q)$
- Модель вероятности:

$$q = \text{sigmoid}(\vec{x};\vec{\theta}) \equiv \frac{1}{1+e^{-(\vec{x};\vec{\theta})}} \text{ и } 1-q = \frac{1}{1+e^{(\vec{x};\vec{\theta})}}$$

- Вместо класса $y \in \{0;1\}$ удобно взять $z = (2y-1) \in \{-1;+1\}$

Вероятностный подход к задаче классификации

Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1-y)\ln(1-q)$
- Модель вероятности:

$$q = \text{sigmoid}(\vec{x};\vec{\theta}) \equiv \frac{1}{1+e^{-(\vec{x};\vec{\theta})}} \text{ и } 1-q = \frac{1}{1+e^{(\vec{x};\vec{\theta})}}$$

- Вместо класса $y \in \{0;1\}$ удобно взять $z = (2y-1) \in \{-1;+1\}$
- $\mathcal{L}_{\log} = \frac{1+z}{2}\ln\left(1+e^{-(\vec{x};\vec{\theta})}\right) + \frac{1-z}{2}\ln\left(1+e^{(\vec{x};\vec{\theta})}\right) = \ln\left(1+e^{-z\cdot(\vec{x};\vec{\theta})}\right)$

Вероятностный подход к задаче классификации

Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1-y)\ln(1-q)$
- Модель вероятности:

$$q = \text{sigmoid}(\vec{x};\vec{\theta}) \equiv \frac{1}{1+e^{-(\vec{x};\vec{\theta})}} \text{ и } 1-q = \frac{1}{1+e^{(\vec{x};\vec{\theta})}}$$

- Вместо класса $y \in \{0;1\}$ удобно взять $z = (2y-1) \in \{-1;+1\}$
- $\mathcal{L}_{\log} = \frac{1+z}{2}\ln\left(1+e^{-(\vec{x};\vec{\theta})}\right) + \frac{1-z}{2}\ln\left(1+e^{(\vec{x};\vec{\theta})}\right) = \ln\left(1+e^{-z\cdot(\vec{x};\vec{\theta})}\right)$
- Логарифмическая функция потерь: $\mathcal{L} = \log_2\left(1+e^{-M}\right)$

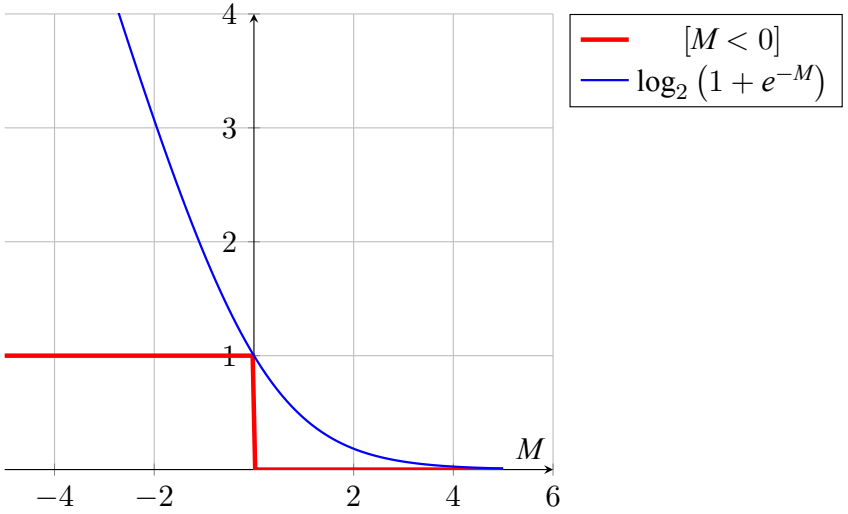
Вероятностный подход к задаче классификации

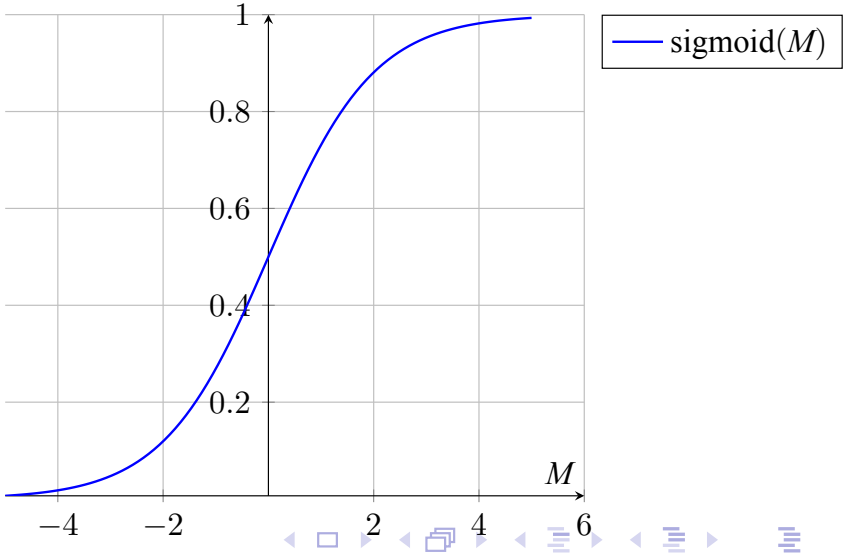
Бинарная классификация

- Логистическая функция потерь: $\mathcal{L} = -y \ln q - (1-y)\ln(1-q)$
- Модель вероятности:

$$q = \text{sigmoid}(\vec{x};\vec{\theta}) \equiv \frac{1}{1+e^{-(\vec{x};\vec{\theta})}} \text{ и } 1-q = \frac{1}{1+e^{(\vec{x};\vec{\theta})}}$$

- Вместо класса $y \in \{0;1\}$ удобно взять $z = (2y-1) \in \{-1;+1\}$
- $\mathcal{L}_{\log} = \frac{1+z}{2}\ln\left(1+e^{-(\vec{x};\vec{\theta})}\right) + \frac{1-z}{2}\ln\left(1+e^{(\vec{x};\vec{\theta})}\right) = \ln\left(1+e^{-z\cdot(\vec{x};\vec{\theta})}\right)$
- Логарифмическая функция потерь: $\mathcal{L} = \log_2\left(1+e^{-M}\right)$





Вероятностный подход к задаче классификации

Многоклассовая классификация

Многоклассовая классификация:

$$q_j = \text{softmax}\left(\vec{x}\theta_j\right) \equiv \frac{e^{(\vec{x}\theta_j)}}{\sum_{i=1}^{s} e^{(\vec{x}\theta_i)}}$$

Обобщенная линейная модель

Экспоненциальное семейство распределений

$$w(y_i|\psi_i, \phi_i) = \exp\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i, \phi_i)\right)$$

$$\begin{cases} \mathcal{M}\{y_i\} = c'(\psi_i) & \text{– среднее значение} \\ \mathcal{M}\left\{(y_i - \mathcal{M}\{y_i\})^2\right\} = \phi_i c''(\psi_i) & \text{– дисперсия} \end{cases}$$

Выбирается линейная модель параметра ψ_i : $\psi_i = (\vec{x}_i; \vec{\theta})$

$$\ln L(\vec{\theta}) = \ln w(\vec{y}|\mathcal{X}, \vec{\theta}) = \sum_{i=1}^{L}\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i, \phi_i)\right) \underset{\vec{\theta}}{\to} \max$$

Задача нелинейной регрессии: $\sum_{i=1}^{L}\frac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max$

Обобщенная линейная модель

Экспоненциальное семейство распределений

$$w(y_i|\psi_i, \phi_i) = \exp\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i, \phi_i)\right)$$

Гауссовское распределение

$$w(y_i|m_i, \sigma_i) = \frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{(y_i - m_i)^2}{2\sigma_i^2}\right) =$$

$$= \exp\left(\frac{y_i m_i - \frac{1}{2}m_i^2}{\sigma_i^2} - \frac{y_i^2}{2\sigma_i^2} - \frac{\ln(2\pi\sigma_i^2)}{2}\right)$$

$$\psi_i = m_i; \qquad c(\psi_i) = \frac{1}{2}\psi_i^2; \qquad \phi_i = \sigma_i^2; \qquad h(y_i, \phi_i) = -\frac{1}{2}\ln(2\pi\sigma_i^2)$$

$$\sum_{i=1}^{L}\frac{(y_i - (\vec{x}_i; \vec{\theta}))^2}{\sigma_i^2} \underset{\vec{\theta}}{\to} \min$$

Обобщенная линейная модель

Экспоненциальное семейство распределений

$$w(y_i|\psi_i, \phi_i) = \exp\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i, \phi_i)\right)$$

Распределение Бернулли

$$w(y_i|m_i) = m_i^{y_i}(1-m_i)^{1-y_i} = \exp\left(y_i \ln\frac{m_i}{1-m_i} + \ln(1-m_i)\right)$$

$$\psi_i = \ln\frac{m_i}{1-m_i}; \qquad c(\psi_i) = \ln(1+e^{\psi_i}); \qquad \phi_i = 1; \qquad h(y_i, \phi_i) = 0$$

$$\sum_{i=1}^{L}(1-\sigma_i)\sigma_i\left((\vec{x}_i\vec{\theta}) - \frac{z_i}{\sigma_i}\right)^2 \underset{\vec{\theta}}{\to} \min; \quad \sigma_i = \text{sigmoid}((\vec{x};\vec{\theta})z_i)$$

Лекция 7. Многомерная линейная и нелинейная регрессия. Градиентные методы

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Линейная регрессия

Метод наименьших квадратов: гомоскедастичность

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \mathcal{R}$$

- Выбирается линейная модель: $a(\vec{x}, \vec{\theta}) = (\vec{x}; \vec{\theta}) = \sum_{j=1}^{n} x_j \theta_j$
- Квадратичная функция потерь: $\mathcal{L}(a, \vec{x}, y) = (a - y)^2$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L} \sum_{i=1}^{L} \Big((\vec{x}_i; \vec{\theta}) - y_i \Big)^2 \underset{\vec{\theta}}{\to} \min$$

- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}) = \frac{1}{K} \sum_{i=1}^{K} \Big((\vec{x}_i; \vec{\theta}) - y_i \Big)^2$$

Линейная регрессия

Метод наименьших квадратов с весами: гетероскедастичность

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \mathcal{R}$$

- Выбирается линейная модель: $a(\vec{x}, \vec{\theta}) = (\vec{x}; \vec{\theta}) = \sum_{j=1}^{n} x_j \theta_j$
- Квадратичная функция потерь: $\mathcal{L}(a, \vec{x}, y) = \frac{1}{\sigma^2(\vec{x})}(a - y)^2$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L} \sum_{i=1}^{L} \frac{1}{\sigma^2(\vec{x}_i)} \left((\vec{x}_i; \vec{\theta}) - y_i \right)^2 \underset{\vec{\theta}}{\rightarrow} \min$$

- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}) = \frac{1}{K} \sum_{i=1}^{K} \frac{1}{\sigma^2(\vec{x}_i)} \left((\vec{x}_i; \vec{\theta}) - y_i \right)^2$$

Линейная регрессия

Метод наименьших квадратов с весами: гетероскедастичность

Пример: амплитуда затухающих колебаний маятника $A(t) = A_0 e^{-\gamma t}$

В эксперименте измерены значения $y(t_i) = A(t_i) + \epsilon_i$, где $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

Линейная регрессия

Метод наименьших квадратов с весами: гетероскедастичность

Пример: амплитуда затухающих колебаний маятника $A(t) = A_0 e^{-\gamma t}$
В эксперименте измерены значения $y(t_i) = A(t_i) + \epsilon_i$, где $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

Способ 1

Исходная модель:
$$y(t_i) = A_0 e^{-\gamma t_i} + \epsilon_i$$

$$A(t) = A_0 e^{-\gamma t} \Rightarrow \ln A(t) = \ln A_0 - \gamma t$$

Новая линейная модель:
$$\ln y(t_i) = \ln A_0 - \gamma t_i + \epsilon_i'$$

Это задача **линейной** регрессии
относительно A_0 и γ.

Линейная регрессия

Метод наименьших квадратов с весами: гетероскедастичность

Пример: амплитуда затухающих колебаний маятника $A(t) = A_0 e^{-\gamma t}$
В эксперименте измерены значения $y(t_i) = A(t_i) + \epsilon_i$, где $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

Способ 1

Исходная модель:

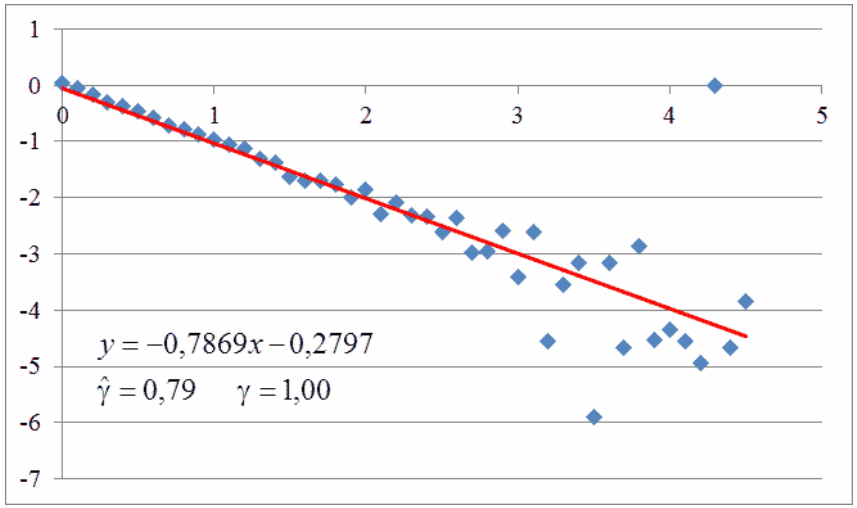
$$y(t_i) = A_0 e^{-\gamma t_i} + \epsilon_i$$

$$A(t) = A_0 e^{-\gamma t} \Rightarrow \ln A(t) = \ln A_0 - \gamma t$$

Новая линейная модель:

$$\ln y(t_i) = \ln A_0 - \gamma t_i + \epsilon_i'$$

Это задача **линейной** регрессии относительно A_0 и γ.



Гетероскедастичность

Линейная регрессия

Метод наименьших квадратов с весами: гетероскедастичность

Пример: амплитуда затухающих колебаний маятника $A(t) = A_0 e^{-\gamma t}$
В эксперименте измерены значения $y(t_i) = A(t_i) + \epsilon_i$, где $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

Способ 1

Исходная модель:
$$y(t_i) = A_0 e^{-\gamma t_i} + \epsilon_i$$

$$A(t) = A_0 e^{-\gamma t} \Rightarrow \ln A(t) = \ln A_0 - \gamma t$$

Новая линейная модель:
$$\ln y(t_i) = \ln A_0 - \gamma t_i + \epsilon_i'$$

Это задача **линейной** регрессии относительно A_0 и γ.

Способ 2

Исходная модель:
$$y(t_i) = A_0 e^{-\gamma t_i} + \epsilon_i$$

Это задача **нелинейной** регрессии относительно A_0 и γ.

Метод наименьших квадратов

Формальное решение

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L}\sum_{i=1}^{L}\Big((\vec{x}_i;\vec{\theta}) - y_i\Big)^2 \underset{\vec{\theta}}{\to} \min$$

Пусть $\mathcal{X} = \{\vec{x}_1, \vec{x}_2, \ldots \vec{x}_L\}; \vec{y} = \{y_1, y_2, \ldots, y_L\}$, тогда $\left\|\mathcal{X}\vec{\theta} - \vec{y}\right\|_2^2 \underset{\vec{\theta}}{\to} \min$

Формальное решение $\vec{\theta}_0 = \underbrace{(\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T}_{\mathcal{X}^+ \text{ – псевдообратная матрица}} \cdot \vec{y}$

Метод наименьших квадратов

Геометрический смысл

$$Q(\theta) = \left\| \mathcal{X}\vec{\theta} - \vec{y} \right\|_2^2 \underset{\vec{\theta}}{\to} \min \quad \Rightarrow \quad \vec{\theta}_0 = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T\vec{y}$$

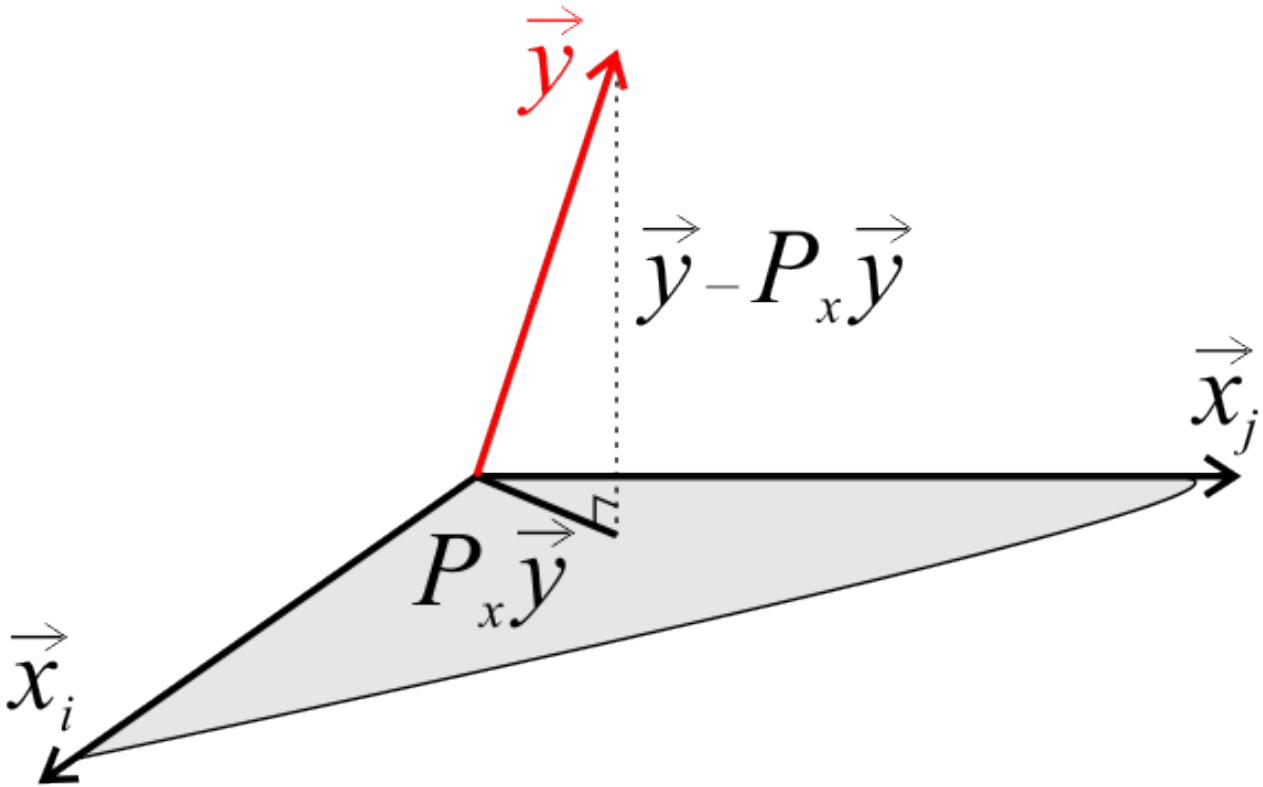
$$Q(\theta_0) = \left\| \vec{y} - \underbrace{\mathcal{X}(\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T}_{P_x}\vec{y} \right\|_2^2 = \| \vec{y} - P_x\vec{y} \|_2^2 \underset{\vec{\theta}}{\to} \min$$

- $\mathcal{X}_{[L\times n]}$ – матрица признаков
- $\vec{\theta}_{[n]}$ – вектор параметров модели
- $(\mathcal{X}\vec{\theta})_{[L]}$ – аппроксимация МНК, вектор из линейной оболочки $\vec{x}_i$
- $\vec{y}_{[L]}$ – вектор целевых переменных
- $P_{x[L\times L]}$ – проекционная матрица на линейную оболочку $\vec{x}_i$
- $(\vec{y} - P_x\vec{\theta})_{[L]}$ – вектор остатков

Метод наименьших квадратов

Геометрический смысл

$$Q(\theta_0) = \left\| \vec{y} - \underbrace{\mathcal{X}(\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T}_{P_x}\vec{y} \right\|_2^2 = \|\vec{y} - P_x\vec{y}\,\|_2^2 \underset{\vec{\theta}}{\to} \min$$



Сингулярное разложение

Произвольная матрица $\mathcal{X}_{[L\times n]}$ может быть представлена в виде сингулярного разложения:

$$\mathcal{X}_{[L\times n]} = V_{[L\times n]} D_{[n\times n]} U^T_{[n\times n]}$$

- $V_{[L\times n]}$ состоит из n собственных векторов $\vec{v}_{i[L]}$ матрицы $(\mathcal{X}\mathcal{X}^T)_{[L\times L]}$.
- $U_{[n\times n]}$ состоит из n собственных векторов $\vec{u}_{i[n]}$ матрицы $(\mathcal{X}^T\mathcal{X})_{[n\times n]}$.
- Собственные значения $\lambda_1,\ldots,\lambda_n$ матриц $\mathcal{X}\mathcal{X}^T$ и $\mathcal{X}^T\mathcal{X}$ совпадают.
- Собственные значения $\lambda_1,\ldots,\lambda_n \geq 0$.
- V и U – ортонормированные матрицы: $V^TV = I_{[n\times n]}$; $U^TU = I_{[n\times n]}$.
- Матрица $D_{[n\times n]}$ диагональна и равна $D_{[n\times n]} = \mathrm{diag}(\sqrt{\lambda_1},\ldots,\sqrt{\lambda_n})$.

Метод наименьших квадратов

Применение сингулярного разложения

- Решение задачи МНК: $\vec{\theta}_0 = \mathcal{X}^+\vec{y} = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T\vec{y}$.
- Сингулярное разложение: $\mathcal{X} = VDU^T \quad \Rightarrow \quad \mathcal{X}^T = UDV^T$.
- Псевдообратная матрица:

$$\mathcal{X}^+_{[n\times L]} = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T =$$
$$= (UDV^T \cdot VDU^T)^{-1} \cdot UDV^T = (UD \cdot DU^T)^{-1} \cdot UDV^T =$$
$$= (DU^T)^{-1} \cdot (UD)^{-1} \cdot UDV^T = (DU^T)^{-1} \cdot V^T =$$
$$= (U^T)^{-1}D^{-1}V^T = UD^{-1}V^T = \sum_{j=1}^{n} \frac{1}{\sqrt{\lambda_j}}\vec{u}_j\vec{v}_j^T.$$

- Решение задачи МНК: $\vec{\theta}_0 = \mathcal{X}^+\vec{y} = UD^{-1}V^T\vec{y} = \sum_{j=1}^{n} \frac{1}{\sqrt{\lambda_j}}\vec{u}_j(\vec{v}_j^T\vec{y})$.
- Аппроксимация: $\mathcal{X}\vec{\theta}_0 = VDU^T \cdot UD^{-1}V^T\vec{y} = VV^T\vec{y} = \sum_{j=1}^{n} \vec{v}_j(\vec{v}_j^T\vec{y})$.

Метод наименьших квадратов

Применение сингулярного разложения

- Сингулярное разложение: $\mathcal{X}_{[L\times n]} = VDU^T$.
- Псевдообратная матрица

$$\mathcal{X}^+_{[n\times L]} = UD^{-1}V^T = \sum_{j=1}^{n} \frac{1}{\sqrt{\lambda_j}} \vec{u}_j \vec{v}_j^T \qquad \text{– \textbf{неустойчива}.}$$

- Решение задачи МНК

$$\vec{\theta}_{0[n]} = UD^{-1}V^T\vec{y} = \sum_{j=1}^{n} \frac{1}{\sqrt{\lambda_j}} \vec{u}_j (\vec{v}_j^T\vec{y}) \qquad \text{– \textbf{неустойчиво}.}$$

- Аппроксимация:

$$(\mathcal{X}\vec{\theta}_0)_{[L]} = VV^T\vec{y} = \sum_{j=1}^{n} \vec{v}_j (\vec{v}_j^T\vec{y}) \qquad \text{– \textbf{устойчива}.}$$

Метод наименьших квадратов

Применение сингулярного разложения

- Сингулярное разложение: $\mathcal{X}_{[L\times n]} = VDU^T$.
- Псевдообратная матрица

$$\mathcal{X}^+_{[n\times L]} = UD^{-1}V^T = \sum_{j=1}^{n} \frac{1}{\sqrt{\lambda_j}} \vec{u}_j \vec{v}_j^T \qquad \text{– \textbf{неустойчива}.}$$

- Решение задачи МНК

$$\vec{\theta}_{0[n]} = UD^{-1}V^T\vec{y} = \sum_{j=1}^{n} \frac{1}{\sqrt{\lambda_j}} \vec{u}_j (\vec{v}_j^T\vec{y}) \qquad \text{– \textbf{неустойчиво}.}$$

- Аппроксимация:

$$(\mathcal{X}\vec{\theta}_0)_{[L]} = VV^T\vec{y} = \sum_{j=1}^{n} \vec{v}_j (\vec{v}_j^T\vec{y}) \qquad \text{– \textbf{устойчива}.}$$

Проблема мультиколлинеарности: комбинация почти линейно зависимых признаков приводит к существованию малых собственных значений λ_i.

Следствия: неустойчивость, рост $\left\|\vec{\theta}_0\right\|$, переобучение.

Метод наименьших квадратов

Методы борьбы с мультиколлинеарностью

- Регуляризация (например, ridge-регрессия).
- Отбор независимых признаков (например, LASSO-регрессия).
- Переход к новому пространству независимых признаков.

Линейное преобразование признаков

Преобразование без потери информации:

$$\mathcal{X}_{[L\times n]} = \overbrace{\underbrace{V_{[L\times n]}}_{\{\vec{v}_1,\vec{v}_2,\cdots,\vec{v}_n\}} \cdot \underbrace{D_{[n\times n]}}}^{G_{[L\times n]}\ -\ \text{новые признаки}} \cdot \overbrace{U^T_{[n\times n]}}^{\text{матрица преобразования}}$$

$$D = \begin{pmatrix} \sqrt{\lambda_1} & 0 & \cdots & 0 \\ 0 & \sqrt{\lambda_2} & \vdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{\lambda_n} \end{pmatrix}$$

Линейное преобразование признаков

Преобразование без потери информации:

$$\mathcal{X}_{[L\times n]} = \overbrace{\underbrace{V_{[L\times n]}}_{\{\vec{v}_1,\vec{v}_2,\cdots,\vec{v}_n\}} \cdot \underbrace{D_{[n\times n]}}}^{G_{[L\times n]}\ -\ \text{новые признаки}} \cdot \overbrace{U^T_{[n\times n]}}^{\text{матрица преобразования}}$$

$$D_{[n\times n]} = \begin{pmatrix} \sqrt{\lambda_1} & 0 & \cdots & 0 \\ 0 & \sqrt{\lambda_2} & \vdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{\lambda_n} \end{pmatrix}$$

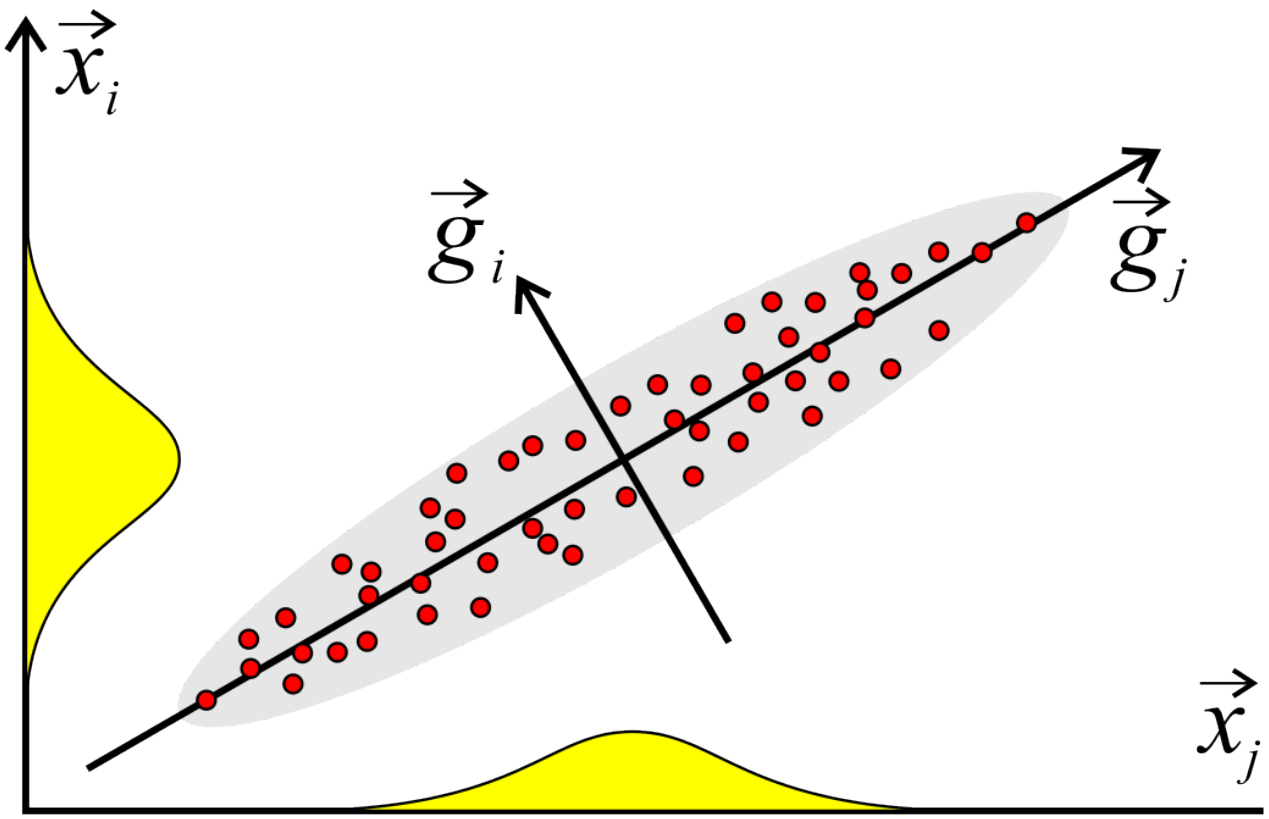
Преобразование с потерей информации:

- $\lambda'_i = \lambda_i[i \le m]$ – выбираются m наибольших собственных значений
- $D'_{[n\times n]} = \operatorname{diag}(\lambda'_1,\lambda'_2,\ldots,\lambda'_n)$; $\quad \mathcal{X}'_{[L\times n]} = V_{[L\times n]} D'_{[n\times n]} U^T_{[n\times n]}$
- $\mathcal{X}'_{[L\times n]} \approx \mathcal{X}_{[L\times n]}$

Метод главных компонент

- Для матрицы $\mathcal{X}_{[L\times n]}$ строится матрица $(\mathcal{X}^T\mathcal{X})_{[n\times n]}$
- У нее отбираются m наибольших собственных значений $\lambda_1,\ldots,\lambda_m$
- Соответствующие собственные векторы формируют матрицу $U_{[n\times m]}$
- U – матрица перехода между n-мерным и m-мерным признаковыми пространствами. Новые признаки вычисляются как $G_{[L\times m]} = \mathcal{X}U$
- Матрица U ортонормирована: $U^TU = I_{[m\times m]}$
- $\mathcal{X} \approx GU^T$

Метод главных компонент



- $\vec{x}_j$ – линейно зависимые признаки
- $\vec{g}_j$ – линейно независимые признаки с разной дисперсией

Нелинейная регрессия

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \mathcal{R}$$

- Выбирается параметрическая модель: $a(\vec{x}, \vec{\theta}) = f(\vec{x}; \vec{\theta})$
- Выбирается функция потерь: $\mathcal{L}(a, \vec{x}, y)$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L} \sum_{i=1}^{L} \mathcal{L}(f(\vec{x}_i; \vec{\theta}), \vec{x}_i, y_i) \underset{\vec{\theta}}{\to} \min$$

- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}) = \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(f(\vec{x}_i; \vec{\theta}), \vec{x}_i, y_i) \underset{\vec{\theta}}{\to} \min$$

Нелинейная регрессия

Нелинейная модель и квадратичная функция потерь

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

Нелинейная регрессия

Нелинейная модель и квадратичная функция потерь

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$

Нелинейная регрессия

Нелинейная модель и квадратичная функция потерь

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:

Нелинейная регрессия

Нелинейная модель и квадратичная функция потерь

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. Линейное приближение функции $f(\vec{x}_i; \vec{\theta})$:

$$f(\vec{x}_i; \vec{\theta}) \approx f(\vec{x}_i; \hat{\vec{\theta}}_0) + \nabla_{\vec{\theta}} f(\vec{x}_i; \hat{\vec{\theta}})(\vec{\theta} - \hat{\vec{\theta}}) =$$

$$= \underbrace{f(\vec{x}_i; \hat{\vec{\theta}}) - \nabla_{\vec{\theta}} f(\vec{x}_i; \hat{\vec{\theta}})\hat{\vec{\theta}}}_{\alpha(\vec{x}_i; \hat{\vec{\theta}})} + \underbrace{\nabla_{\vec{\theta}} f(\vec{x}_i; \hat{\vec{\theta}})}_{\beta(\vec{x}_i; \hat{\vec{\theta}})} \vec{\theta} = \alpha(\vec{x}_i; \hat{\vec{\theta}}) + \beta(\vec{x}_i; \hat{\vec{\theta}})\vec{\theta}$$

Нелинейная регрессия

Нелинейная модель и квадратичная функция потерь

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. Линейное приближение функции $f(\vec{x}_i; \vec{\theta})$:
   $$f(\vec{x}_i; \vec{\theta}) \approx f(\vec{x}_i; \hat{\vec{\theta}}_0) + \nabla_{\vec{\theta}} f(\vec{x}_i; \hat{\vec{\theta}})(\vec{\theta} - \hat{\vec{\theta}}) =$$
   $$= \underbrace{f(\vec{x}_i; \hat{\vec{\theta}}) - \nabla_{\vec{\theta}} f(\vec{x}_i; \hat{\vec{\theta}})\hat{\vec{\theta}}}_{\alpha(\vec{x}_i; \hat{\vec{\theta}})} + \underbrace{\nabla_{\vec{\theta}} f(\vec{x}_i; \hat{\vec{\theta}})}_{\beta(\vec{x}_i; \hat{\vec{\theta}})} \vec{\theta} = \alpha(\vec{x}_i; \hat{\vec{\theta}}) + \beta(\vec{x}_i; \hat{\vec{\theta}})\vec{\theta}$$
   2. Решение задачи линейной регрессии и обновление оценки $\hat{\vec{\theta}}$
   $$Q(\vec{\theta}) = \sum_{i=1}^{L} \left[\beta(\vec{x}_i; \hat{\vec{\theta}})\vec{\theta} - (y_i - \alpha(\vec{x}_i; \hat{\vec{\theta}}))\right]^2 \underset{\vec{\theta}}{\to} \min$$

Минимизации функционала качества. Методы 1 порядка

Метод градиентного спуска

$$Q(\vec{\theta}) = \sum_{i=1}^{L} \mathcal{L}(f(\vec{x}_i; \vec{\theta}), \vec{x}_i, y_i) \underset{\vec{\theta}}{\to} \min$$

Идея: поиск минимума в направлении скорейшего спуска

1. Выбирается значение скорости обучения lr
2. Инициализация начальных значений параметров $\vec{\theta}$
3. Цикл, пока $\vec{\theta}$ и $Q(\vec{\theta})$ не сошлись к определенным значениям:
   1. Вычисляется градиент $\nabla_{\vec{\theta}} Q(\vec{\theta}) = \sum_{i=1}^{L} \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
   2. Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \sum_{i=1}^{L} \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$

Минимизации функционала качества. Методы 1 порядка

Метод стохастического градиентного спуска (Stochastic Gradient Descent)

Идея: делать градиентный шаг на каждом объекте, а не на всех сразу

1. Выбирается значение скоростей обучения lr и забывания λ
2. Инициализация начальных значений параметров $\vec{\theta}$
3. Делается начальная оценка функционала $\hat{Q} := Q(\vec{\theta})$
4. Цикл, пока $\vec{\theta}$ и $\hat{Q}$ не сошлись к определенным значениям:
   1. Выбирается произвольный объект $\vec{x}_i$
   2. Вычисляется градиент $\vec{G}_i = \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
   3. Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \vec{G}_i$
   4. Обновляется значение функционала $\hat{Q} := (1 - \lambda)\hat{Q} + \lambda \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$

Минимизации функционала качества. Методы 1 порядка

Метод стохастического градиентного спуска (Stochastic Gradient Descent)

Идея: делать градиентный шаг на каждом объекте, а не на всех сразу

1. Выбирается значение скоростей обучения lr и забывания λ
2. Инициализация начальных значений параметров $\vec{\theta}$
3. Делается начальная оценка функционала $\hat{Q} := Q(\vec{\theta})$
4. Цикл, пока $\vec{\theta}$ и $\hat{Q}$ не сошлись к определенным значениям:
   1. Выбирается произвольный объект $\vec{x}_i$
   2. Вычисляется градиент $\vec{G}_i = \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
   3. Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \vec{G}_i$
   4. Обновляется значение функционала $\hat{Q} := (1-\lambda)\hat{Q} + \lambda\mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$

Используется оценка экспоненциального скользящего среднего:

$$\hat{Q} = \lambda\mathcal{L}^{(n)} + (1-\lambda)\lambda\mathcal{L}^{(n-1)} + (1-\lambda)^2\lambda\mathcal{L}^{(n-2)} + \cdots = \lambda\sum_{k=0}^{n}(1-\lambda)^k\mathcal{L}^{(n-k)}$$

Минимизации функционала качества. Методы 1 порядка

Метод стохастического усредненного градиентного спуска (Stochastic Avegare Gradient)

Идея: усреднить градиент по большому числу итераций

1. Выбирается значение скорости обучения lr и забывания λ
2. Инициализация начальных значений параметров $\vec{\theta}_0$
3. Сохраняется оценка всех градиентов $\vec{G}_i = \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
4. Делается начальная оценка функционала $\hat{Q} := Q(\vec{\theta})$
5. Цикл, пока $\vec{\theta}$ и Q не сошлись к определенным значениям:
   1. Выбирается произвольный объект $\vec{x}_i$
   2. Вычисляется и обновляется градиент $\vec{G}_i = \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
   3. Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \frac{1}{L} \sum_{i=1}^{L} \vec{G}_i$

      Из суммы вычитается старое значение $\vec{G}_i$ и добавляется новое
   4. Обновляется значение функционала $\hat{Q} := (1 - \lambda)\hat{Q} + \lambda \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$

Минимизации функционала качества. Методы 1 порядка

Метод импульса (Momentum)

Идея: применить к градиенту экспоненциальное скользящее среднее.

Аналогия – шарик, катящийся по поверхности с вязким трением.

1. Выбирается значение скоростей обучения lr и забывания λ; μ
2. Инициализация начальных значений параметров $\vec{\theta}$
3. Делается оценка функционала $\hat{Q} := Q(\vec{\theta})$ и импульса $\vec{P} = 0$
4. Цикл, пока $\vec{\theta}$ и $\hat{Q}$ не сошлись к определенным значениям:
   1. Выбирается произвольный объект $\vec{x}_i$
   2. Обновляется значение импульса $\vec{P} := (1 - \mu)\vec{P} + \mu \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
   3. Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \vec{P}$
   4. Обновляется значение функционала $\hat{Q} := (1 - \lambda)\hat{Q} + \lambda \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$

Минимизации функционала качества. Методы 1 порядка

Метод Нестерова (Nesterov Accelerated Gradient)

Идея: вычислять градиент не в “текущей”, а в “следующей” точке

1. Выбирается значение скоростей обучения lr и забывания $\lambda; \mu$
2. Инициализация начальных значений параметров $\vec{\theta}$
3. Делается оценка функционала $\hat{Q} := Q(\vec{\theta})$ и импульса $\vec{P} = 0$
4. Цикл, пока $\vec{\theta}$ и $\hat{Q}$ не сошлись к определенным значениям:
   1. Выбирается произвольный объект $\vec{x}_i$
   2. Обновляется значение импульса $\vec{P} := (1 - \mu)\vec{P} + \mu \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta} - lr \cdot P, \vec{x}_i, y_i)$
   3. Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \hat{\vec{P}}$
   4. Обновляется значение функционала $\hat{Q} := \hat{Q}(1 - \lambda) + \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)\lambda$

Выбор градиентного шага

- Если шаг слишком мал, сходимость будет долгой.
- Если шаг слишком велик, алгоритм может не сойтись.

Выбор градиентного шага

- Если шаг слишком мал, сходимость будет долгой.
- Если шаг слишком велик, алгоритм может не сойтись.
- Адаптивный шаг, зависящий от номера итерации.
  - В теории должно быть $lr^{(n)} \to 0; \sum_{n=1}^{\infty} lr^{(n)} = \infty; \sum_{n=1}^{\infty} \left(lr^{(n)}\right)^2 < \infty$

Выбор градиентного шага

- Если шаг слишком мал, сходимость будет долгой.
- Если шаг слишком велик, алгоритм может не сойтись.
- Адаптивный шаг, зависящий от номера итерации.
  - В теории должно быть $lr^{(n)} \to 0; \sum\limits_{n=1}^{\infty} lr^{(n)} = \infty; \sum\limits_{n=1}^{\infty} \left(lr^{(n)}\right)^2 < \infty$

    Например, $h^{(n)} = n^{-1}$.

Выбор градиентного шага

- Если шаг слишком мал, сходимость будет долгой.
- Если шаг слишком велик, алгоритм может не сойтись.
- Адаптивный шаг, зависящий от номера итерации.
  - В теории должно быть $lr^{(n)} \to 0; \sum\limits_{n=1}^{\infty} lr^{(n)} = \infty; \sum\limits_{n=1}^{\infty} \left(lr^{(n)}\right)^2 < \infty$

    Например, $h^{(n)} = n^{-1}$.
  - Метод наискорейшего градиентного спуска:
    $\mathcal{L}(\vec{\theta} - lr \cdot \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta})) \underset{lr}{\to} \min$.

Выбор градиентного шага

- Если шаг слишком мал, сходимость будет долгой.
- Если шаг слишком велик, алгоритм может не сойтись.
- Адаптивный шаг, зависящий от номера итерации.
  - В теории должно быть $lr^{(n)} \to 0; \sum\limits_{n=1}^{\infty} lr^{(n)} = \infty; \sum\limits_{n=1}^{\infty} \left(lr^{(n)}\right)^2 < \infty$

    Например, $h^{(n)} = n^{-1}$.
  - Метод наискорейшего градиентного спуска:
    $\mathcal{L}(\vec{\theta} - lr \cdot \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta})) \underset{lr}{\to} \min$.
  - Случайные шаги, чтобы не “застревать” в локальных минимумах.

Выбор градиентного шага

- Если шаг слишком мал, сходимость будет долгой.
- Если шаг слишком велик, алгоритм может не сойтись.
- Адаптивный шаг, зависящий от номера итерации.
  - В теории должно быть $lr^{(n)} \to 0; \sum\limits_{n=1}^{\infty} lr^{(n)} = \infty; \sum\limits_{n=1}^{\infty} \left(lr^{(n)}\right)^2 < \infty$

    Например, $h^{(n)} = n^{-1}$.
  - Метод наискорейшего градиентного спуска:
    $\mathcal{L}(\vec{\theta} - lr \cdot \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta})) \underset{lr}{\to} \min$.
  - Случайные шаги, чтобы не “застревать” в локальных минимумах.
- Использование методов второго порядка.

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего
- Предварительный быстрый поиск по малой подвыбоке объектов

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего
- Предварительный быстрый поиск по малой подвыбоке объектов
- МНК в предположении, что признаки независимы

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего
- Предварительный быстрый поиск по малой подвыбоке объектов
- МНК в предположении, что признаки независимы:
  - Матрица признаков $\mathcal{X}_{[L\times n]} = \{\vec{x}_1, \vec{x}_2, \ldots \vec{x}_L\} = \{\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_n\}^T$

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего
- Предварительный быстрый поиск по малой подвыбоке объектов
- МНК в предположении, что признаки независимы:
  - Матрица признаков $\mathcal{X}_{[L\times n]} = \{\vec{x}_1, \vec{x}_2, \ldots \vec{x}_L\} = \{\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_n\}^T$
  - Признаки считаем независимыми: $i \neq j \quad \Rightarrow \quad (\vec{f}_i; \vec{f}_j) = 0$

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего
- Предварительный быстрый поиск по малой подвыбоке объектов
- МНК в предположении, что признаки независимы:
  - Матрица признаков $\mathcal{X}_{[L\times n]} = \{\vec{x}_1, \vec{x}_2, \ldots \vec{x}_L\} = \{\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_n\}^T$
  - Признаки считаем независимыми: $i \neq j \quad \Rightarrow \quad (\vec{f}_i; \vec{f}_j) = 0$
  - Ищем решение вида $\mathcal{X}\vec{\theta}_0 = \vec{y}$

Минимизации функционала качества

Инициализация начальных значений параметров $\vec{\theta}$

- $\vec{\theta} = \vec{0}$
- $\vec{\theta}$ – небольшие случайные значения
- Запуск алгоритма с разными начальными значениями $\vec{\theta}$ и выбор лучшего
- Предварительный быстрый поиск по малой подвыбоке объектов
- МНК в предположении, что признаки независимы:
  - Матрица признаков $\mathcal{X}_{[L\times n]} = \{\vec{x}_1, \vec{x}_2, \ldots \vec{x}_L\} = \{\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_n\}^T$
  - Признаки считаем независимыми: $i \neq j \quad \Rightarrow \quad (\vec{f}_i; \vec{f}_j) = 0$
  - Ищем решение вида $\mathcal{X}\vec{\theta}_0 = \vec{y}$
  - Тогда $\vec{f}_j^T \mathcal{X}\vec{\theta}_0 = \vec{f}_j^T \vec{y} \quad \Rightarrow \quad \theta_{0,j} = (\vec{f}_j; \vec{y})/(\vec{f}_j; \vec{f}_j)$

Минимизации функционала качества

Порядок обучения на объектах выборки

- При классификации – выбирать объекты попеременно из разных классов.

Минимизации функционала качества

Порядок обучения на объектах выборки

- При классификации – выбирать объекты попеременно из разных классов.
- С большей вероятностью выбираются объекты, на которых
  - меньше отступ $M_i = a(\vec{x}_i)y_i$, т.е. алгоритм сильно ошибается;
  - меньше модуль отступа $|M_i|$, т.е. классификация неуверенная;
  - больше ошибка алгоритма $\mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$.

Минимизации функционала качества

Порядок обучения на объектах выборки

- При классификации – выбирать объекты попеременно из разных классов.
- С большей вероятностью выбираются объекты, на которых
  - меньше отступ $M_i = a(\vec{x}_i)y_i$, т.е. алгоритм сильно ошибается;
  - меньше модуль отступа $|M_i|$, т.е. классификация неуверенная;
  - больше ошибка алгоритма $\mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$.
- Не выбираются объекты, у которых
  - отступ больше порогового значения (хорошо классифицируются);
  - отступ меньше порогового значения (это выбросы);
  - ошибка алгоритма больше или меньше порогового значения.

Минимизации функционала качества. Методы 1 порядка

Методы адаптивных градиентов

Adaptive Gradient (AdaGrad)

Идея: раздельно учитывать градиент для “редких” и “частых” признаков.

- Вычисляется градиент $\vec{G}_i := \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
- Накапливается вектор суммы квадратов $\vec{\Gamma} := \vec{\Gamma} + \vec{G}_i \odot \vec{G}_i$
- Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \vec{G}_i \oslash \sqrt{\vec{\Gamma} + \vec{\epsilon}}$

Минимизации функционала качества. Методы 1 порядка

Методы адаптивных градиентов

Running Mean Square (RMSProp)

Идея: замена накопления экспоненциальным скользящим средним.

- Вычисляется градиент $\vec{G}_i := \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
- Скользящее экспоненциальное среднее $\vec{\Gamma} := \alpha\vec{\Gamma} + (1-\alpha)\vec{G}_i \odot \vec{G}_i$
- Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \vec{G}_i \oslash \sqrt{\vec{\Gamma} + \vec{\epsilon}}$

Adaptive Delta (AdaDelta)

Идея: двойная нормировка градиентов позволяет не подбирать lr.

- Вычисляется градиент $\vec{G}_i := \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
- Скользящее экспоненциальное среднее $\vec{\Gamma} := \alpha\vec{\Gamma} + (1-\alpha)\vec{G}_i \odot \vec{G}_i$
- Вычисляются “дельты”: $\begin{cases} \vec{\delta} := \vec{G}_i \odot \sqrt{\vec{\Delta} + \vec{\epsilon}} \oslash \sqrt{\vec{\Gamma} + \vec{\epsilon}}; \\ \vec{\Delta} := \alpha\vec{\Delta} + (1-\alpha)\vec{\delta} \odot \vec{\delta} \end{cases}$
- Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \vec{\delta}$

Минимизации функционала качества. Методы 1 порядка

Методы адаптивных градиентов

Adaptive Momentum (Adam)

Идея: объединение методов Momentum и RMSProp

- Вычисляется градиент $G_i := \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
- Скользящее экспоненциальное среднее $\vec{\Gamma} := \alpha\vec{\Gamma} + (1-\alpha)\vec{G}_i \odot \vec{G}_i$
- Обновляется значение импульса $\vec{P} := (1-\mu)\vec{P} + \mu\nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)$
- Корректировки: $\hat{\vec{P}} := \vec{P}/(1-\mu^n)$ и $\hat{\vec{\Gamma}} := \vec{\Gamma}/(1-\alpha^n)$.
- Обновляются значения параметров $\vec{\theta} := \vec{\theta} - lr \cdot \hat{\vec{P}} \oslash \sqrt{\hat{\vec{\Gamma}} + \vec{\epsilon}}$
- Можно использовать $\mu = 0.9; \alpha = 0.999; \epsilon = 10^{-8}$.

Nesterov Adaptive Momentum (Nadam)

Идея: объединение методов NAG и RMSProp

- Обновляются значения параметров

$$\vec{\theta} := \vec{\theta} - lr \cdot \left(\mu\hat{\vec{P}} + \frac{1-\mu}{1-\mu^n}\nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)\right) \oslash \sqrt{\hat{\vec{\Gamma}} + \vec{\epsilon}}$$

Иллюстрация скорости обучения различными алгоритмами

Alec Radford's animations for optimization algorithms:
`http://www.denizyuret.com/2015/03/alec-radfords-animations-for.html`

Минимизации функционала качества. Методы 2 порядка

Метод касательных (Ньютона-Рафсона)

Задача: найти решение уравнения $f(x) = 0$ на отрезке $[a; b]$, если оно существует, единственно, и существует непрерывная на $[a; b]$ производная $f'(x)$, не равная 0.

- Пусть $f(\hat{x}) = 0$, тогда $f(x^{(k)} + (\hat{x} - x^{(k)}) = 0$
- По формуле Лагранжа $\exists \tilde{x} : f(x^{(k)}) + f'(\tilde{x})(\hat{x} - x^{(k)}) = 0$
- Если положить $\tilde{x} := x^{(k)}$ и $\hat{x} = x^{(k+1)}$, получается формула итерационного процесса: $f(x^{(k)}) + f'(x^{(k)})(x^{(k+1)} - x^{(k)}) = 0$
- $x^{(k+1)} = x^{(k)} - \dfrac{f(x^{(k)})}{f'(x^{(k+1)})}$

Минимизации функционала качества. Методы 2 порядка

Метод касательных (Ньютона-Рафсона)

Метод Ньютона-Рафсона

$$\vec{\theta} := \vec{\theta} - lr \cdot H^{-1} \cdot \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)), \quad \text{где } H_{pq} = \frac{\partial^2 \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_p \partial \theta_q} - \text{гессиан}$$

Минимизации функционала качества. Методы 2 порядка

Метод касательных (Ньютона-Рафсона)

Метод Ньютона-Рафсона

$$\vec{\theta} := \vec{\theta} - lr \cdot H^{-1} \cdot \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)), \quad \text{где } H_{pq} = \frac{\partial^2 \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_p \partial \theta_q} - \text{гессиан}$$

Диагональный метод Левенберга-Марквардта

$$\theta_j := \theta_j - lr \cdot \left(\mu + \frac{\partial^2 \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_j^2} \right)^{-1} \cdot \frac{\partial \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_j}$$

Минимизации функционала качества. Методы 2 порядка

Метод касательных (Ньютона-Рафсона)

Метод Ньютона-Рафсона

$$\vec{\theta} := \vec{\theta} - lr \cdot H^{-1} \cdot \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)), \quad \text{где } H_{pq} = \frac{\partial^2 \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_p \partial \theta_q} - \text{гессиан}$$

Диагональный метод Левенберга-Марквардта

$$\theta_j := \theta_j - lr \cdot \left(\mu + \frac{\partial^2 \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_j^2} \right)^{-1} \cdot \frac{\partial \mathcal{L}(\vec{\theta}, \vec{x}_i, y_i)}{\partial \theta_j}$$

- Можно положить $lr = 1$ и оптимизировать только μ
- Вдали от минимума скорость обучения будет lr/μ

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p}$;

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p}$; $\qquad \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} = F_{ip}$

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p}$; $\qquad \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} = F_{ip}$

      $$H_{pq} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_q} - 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial^2 f(\vec{x}_i; \vec{\theta})}{\partial \theta_p \partial \theta_q}$$

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p}$; $\qquad \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} = F_{ip}$

$$H_{pq} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_q} - 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \overbrace{\frac{\partial^2 f(\vec{x}_i; \vec{\theta})}{\partial \theta_p \partial \theta_q}}^{\text{пренебрегается}}$$

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p}$; $\qquad \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} = F_{ip}$

      $H_{pq} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_q}$
   2. $\hat{\vec{\theta}} := \hat{\vec{\theta}} - lr \cdot (F^T F)^{-1} F^T (\vec{f} - \vec{y}) = \hat{\vec{\theta}} - lr \cdot \vec{\beta}$

Метод Ньютона-Рафсона и нелинейная регрессия

Нелинейная модель и квадратичная функция потерь. Iteratevely Reweighted Least Squares

$$Q(\vec{\theta}) = \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i)^2 \underset{\vec{\theta}}{\to} \min$$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p}$; $\qquad \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} = F_{ip}$

      $H_{pq} = 2 \sum_{i=1}^{L} (f(\vec{x}_i; \vec{\theta}) - y_i) \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_p} \frac{\partial f(\vec{x}_i; \vec{\theta})}{\partial \theta_q}$
   2. $\hat{\vec{\theta}} := \hat{\vec{\theta}} - lr \cdot (F^T F)^{-1} F^T (\vec{f} - \vec{y}) = \hat{\vec{\theta}} - lr \cdot \vec{\beta}$
   3. Решение задачи линейной регрессии и обновление оценки $\hat{\vec{\theta}}$

$$\left\| F\vec{\beta} - (\vec{f} - \vec{y}) \right\| \underset{\vec{\theta}}{\to} \min$$

Обобщенная линейная модель

Экспоненциальное семейство распределений

$$w(y_i|\psi_i, \phi_i) = \exp\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i, \phi_i)\right)$$

$$\begin{cases} \mathcal{M}\{y_i\} = c'(\psi_i) & \text{– среднее значение} \\ \mathcal{M}\left\{(y_i - \mathcal{M}\{y_i\})^2\right\} = \phi_i c''(\psi_i) & \text{– дисперсия} \end{cases}$$

Выбирается линейная модель параметра ψ_i : $\psi_i = (\vec{x}_i; \vec{\theta})$

$$\ln L(\vec{\theta}) = \ln w(\vec{y}|\mathcal{X}, \vec{\theta}) = \sum_{i=1}^{L}\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i, \phi_i)\right) \underset{\vec{\theta}}{\to} \max$$

Задача нелинейной регрессии: $\sum_{i=1}^{L}\frac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum\limits_{i=1}^{L} \dfrac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max;\quad \psi_i = (\vec{x}_i; \vec{\theta})$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum\limits_{i=1}^{L} \dfrac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max;\quad \psi_i = (\vec{x}_i; \vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum_{i=1}^{L} \frac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max;\quad \psi_i = (\vec{x}_i;\vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = \sum_{i=1}^{L} \frac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\vec{x}_p$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum\limits_{i=1}^{L} \dfrac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max;\quad \psi_i = (\vec{x}_i;\vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\dfrac{\partial Q}{\partial \theta_p} = \sum\limits_{i=1}^{L} \dfrac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\vec{x}_p;\qquad H_{pq} = -\sum\limits_{i=1}^{L} \dfrac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}(\vec{x}_p;\vec{x}_q);$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum_{i=1}^{L} \frac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max;\quad \psi_i = (\vec{x}_i;\vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = \sum_{i=1}^{L} \frac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\vec{x}_p;\qquad H_{pq} = -\sum_{i=1}^{L} \frac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}(\vec{x}_p;\vec{x}_q);$

      $D = \operatorname{diag}\left(\sqrt{\frac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}}\right)$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum\limits_{i=1}^{L} \dfrac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max;\quad \psi_i = (\vec{x}_i;\vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\dfrac{\partial Q}{\partial \theta_p} = \sum\limits_{i=1}^{L} \dfrac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\vec{x}_p;\qquad H_{pq} = -\sum\limits_{i=1}^{L} \dfrac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}(\vec{x}_p;\vec{x}_q);$

      $D = \operatorname{diag}\left(\sqrt{\dfrac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}}\right) \quad\Rightarrow\quad H = \mathcal{X}^T DD\mathcal{X}$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum_{i=1}^{L} \frac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max; \quad \psi_i = (\vec{x}_i; \vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\frac{\partial Q}{\partial \theta_p} = \sum_{i=1}^{L} \frac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i} \vec{x}_p; \qquad H_{pq} = -\sum_{i=1}^{L} \frac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i} (\vec{x}_p; \vec{x}_q);$

      $D = \operatorname{diag}\left(\sqrt{\frac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}}\right) \quad \Rightarrow \quad H = \mathcal{X}^T D D \mathcal{X}$
   2. $\hat{\vec{\theta}} := \hat{\vec{\theta}} - lr \cdot (\mathcal{X}^T D \underbrace{D\mathcal{X}}_{F})^{-1} \mathcal{X}^T D \underbrace{\left(\sqrt{\frac{\phi_i}{c''((\vec{x}_i;\vec{\theta}))}} \frac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\right)}_{\alpha} = \hat{\vec{\theta}} - lr \cdot \vec{\beta}$

Метод Ньютона-Рафсона и нелинейная регрессия

Обобщенная линейная модель. Iteratevely Reweighted Least Squares

Задача нелинейной регрессии: $Q = \sum\limits_{i=1}^{L} \dfrac{y_i\psi_i - c(\psi_i)}{\phi_i} \underset{\vec{\theta}}{\to} \max; \quad \psi_i = (\vec{x}_i; \vec{\theta})$

1. Инициализация начальных значений параметров $\hat{\vec{\theta}}$
2. Цикл, пока $\hat{\vec{\theta}}$ и Q не сошлись к определенным значениям:
   1. $\dfrac{\partial Q}{\partial \theta_p} = \sum\limits_{i=1}^{L} \dfrac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\vec{x}_p; \qquad H_{pq} = -\sum\limits_{i=1}^{L} \dfrac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}(\vec{x}_p;\vec{x}_q);$

      $D = \operatorname{diag}\left(\sqrt{\dfrac{c''((\vec{x}_i;\vec{\theta}))}{\phi_i}}\right) \quad \Rightarrow \quad H = \mathcal{X}^T D D \mathcal{X}$
   2. $\hat{\vec{\theta}} := \hat{\vec{\theta}} - lr\cdot(\mathcal{X}^T D \underbrace{D\mathcal{X}}_{F})^{-1}\mathcal{X}^T D \underbrace{\left(\sqrt{\dfrac{\phi_i}{c''((\vec{x}_i;\vec{\theta}))}}\dfrac{y_i - c'((\vec{x}_i;\vec{\theta}))}{\phi_i}\right)}_{\alpha} = \hat{\vec{\theta}} - lr\cdot\vec{\beta}$
3. Решение задачи линейной регрессии и обновление оценки $\hat{\vec{\theta}}$

$$\left\| F\vec{\beta} - \vec{\alpha} \right\| \underset{\vec{\theta}}{\to} \min$$

Обобщенная линейная модель

Экспоненциальное семейство распределений

$$w(y_i|\psi_i,\phi_i) = \exp\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i,\phi_i)\right)$$

Гауссовское распределение

$$w(y_i|m_i,\sigma_i) = \frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{(y_i-m_i)^2}{2\sigma_i^2}\right) =$$

$$= \exp\left(\frac{y_i m_i - \frac{1}{2}m_i^2}{\sigma_i^2} - \frac{y_i^2}{2\sigma_i^2} - \frac{\ln(2\pi\sigma_i^2)}{2}\right)$$

$$\psi_i = m_i; \qquad c(\psi_i) = \frac{1}{2}\psi_i^2; \qquad \phi_i = \sigma_i^2; \qquad h(y_i,\phi_i) = -\frac{1}{2}\ln(2\pi\sigma_i^2)$$

$$\sum_{i=1}^{L}\frac{(y_i - (\vec{x}_i;\vec{\theta}))^2}{\sigma_i^2} \underset{\vec{\theta}}{\to} \min$$

Обобщенная линейная модель

Экспоненциальное семейство распределений

$$w(y_i|\psi_i,\phi_i) = \exp\left(\frac{y_i\psi_i - c(\psi_i)}{\phi_i} + h(y_i,\phi_i)\right)$$

Распределение Бернулли

$$w(y_i|m_i) = m_i^{y_i}(1-m_i)^{1-y_i} = \exp\left(y_i \ln\frac{m_i}{1-m_i} + \ln(1-m_i)\right)$$

$$\psi_i = \ln\frac{m_i}{1-m_i}; \qquad c(\psi_i) = \ln(1+e^{\psi_i}); \qquad \phi_i = 1; \qquad h(y_i,\phi_i) = 0$$

$$\sum_{i=1}^{L}(1-\sigma_i)\sigma_i\left((\vec{x}_i\vec{\theta}\,) - \frac{z_i}{\sigma_i}\right)^2 \underset{\vec{\theta}}{\to} \min; \qquad \sigma_i = \text{sigmoid}((\vec{x};\vec{\theta}\,)z_i)$$

Лекция 8. Метрические методы регрессии и классификации

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Схема постановки и решения задач обучения с учителем

- Имеется множество объектов $\mathbb{X}$, между которыми определены расстояния $\rho(\vec{x}_i, \vec{x}_j)$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.
- Выбирается параметрическая модель $a(\vec{x}, \vec{\theta})$
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума $\mathcal{L}$.

Примеры задач

- Сравнение последовательностей символов (текст, ДНК и т.д.)
- Сравнение объектов по фотографиям
- Выделение групп людей на основе графа связей в социальных сетях
- Распознавание рукописного текста с помощью анализа траектории пера

Расстояние между объектами

Для объектов $\vec{x}_j \in \mathbb{X}$ вводится функция расстояния $\rho(\vec{x}_1, \vec{x}_2) \geq 0$.

Расстояние между объектами

Для объектов $\vec{x}_j \in \mathbb{X}$ вводится функция расстояния $\rho(\vec{x}_1, \vec{x}_2) \geq 0$.
Выполнение неравенства треугольника $\rho(\vec{x}_1, \vec{x}_2) + \rho(\vec{x}_2, \vec{x}_3) \geq \rho(\vec{x}_1, \vec{x}_3)$ не обязательно.

Расстояние между объектами

Для объектов $\vec{x}_j \in \mathbb{X}$ вводится функция расстояния $\rho(\vec{x}_1, \vec{x}_2) \geq 0$.
Выполнение неравенства треугольника $\rho(\vec{x}_1, \vec{x}_2) + \rho(\vec{x}_2, \vec{x}_3) \geq \rho(\vec{x}_1, \vec{x}_3)$ не обязательно.
Гипотеза: если расстояние между объектами мало, то

- в задачах классификации близкие объекты относятся к одному классу;
- в задачах регрессии значения целевого признака у этих объектов близки.

Расстояние между объектами

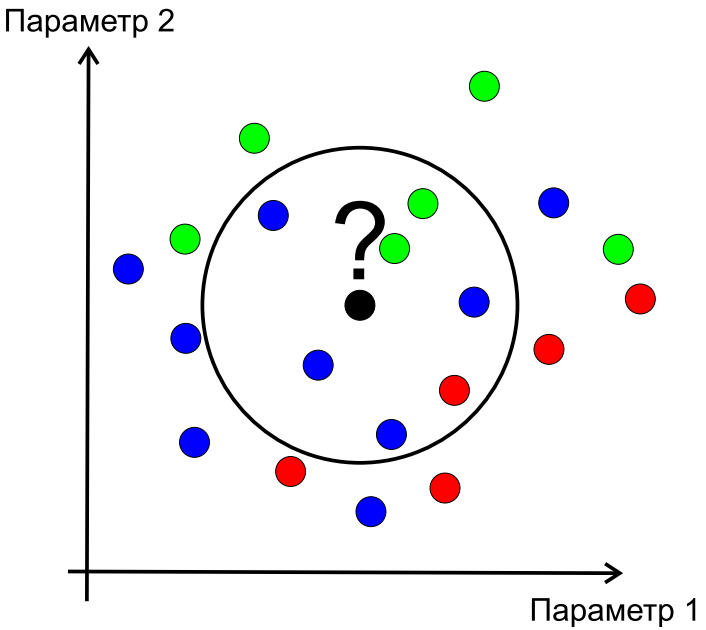
- **1 случай**
  По условию задачи известны расстояния между всеми объектами $\rho(\vec{x}_i, \vec{x}_j)$.
- **2 случай**
  По условию задачи объекты заданы своими признаками $\vec{x}$.

Расстояние между объектами

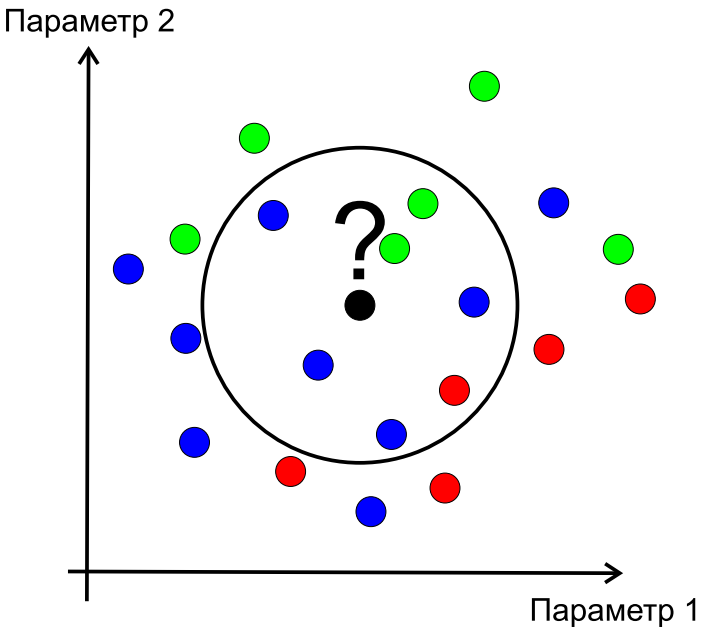
- **1 случай**
  По условию задачи известны расстояния между всеми объектами $\rho(\vec{x}_i, \vec{x}_j)$.
- **2 случай**
  По условию задачи объекты заданы своими признаками $\vec{x}$.
  Тогда расстояние определяется согласно обобщенной метрике Минковского:

$$\rho_P(\vec{x}_i, \vec{x}_j, \vec{\theta}) = \left(\sum_{k=1}^{n} \theta_k |x_{i,k} - x_{j,k}|^P \right)^{1/P}$$

Классификация объектов по соседям



Классификация объектов по соседям



1. Для выбранного $\vec{x}$ расстояния сортируются по возрастанию: $\rho(\vec{x}, \vec{x}_i) \leq \rho(\vec{x}, \vec{x}_{i+1})$
2. Определяется “близость” $\varrho(\vec{x}, y)$ объекта $\vec{x}$ к классу y
3. $a(\vec{x}) = \arg\max_y \varrho(\vec{x}, y)$

Метод k ближайших соседей (kNN)

1. Для выбранного $\vec{x}$ расстояния сортируются по возрастанию: $\rho(\vec{x}, \vec{x}_i) \leq \rho(\vec{x}, \vec{x}_{i+1})$
2. Близость объекта $\vec{x}$ к классу y: $\varrho(\vec{x}, y) = \sum_{i=1}^{k} [y_i = y]$
3. $a(\vec{x}) = \arg\max_y \varrho(\vec{x}, y)$

Метод k ближайших соседей (kNN)

1. Для выбранного $\vec{x}$ расстояния сортируются по возрастанию: $\rho(\vec{x}, \vec{x}_i) \leq \rho(\vec{x}, \vec{x}_{i+1})$
2. Близость объекта $\vec{x}$ к классу y: $\varrho(\vec{x}, y) = \sum_{i=1}^{k} [y_i = y]$
3. $a(\vec{x}) = \arg\max_y \varrho(\vec{x}, y)$

Преимущества:

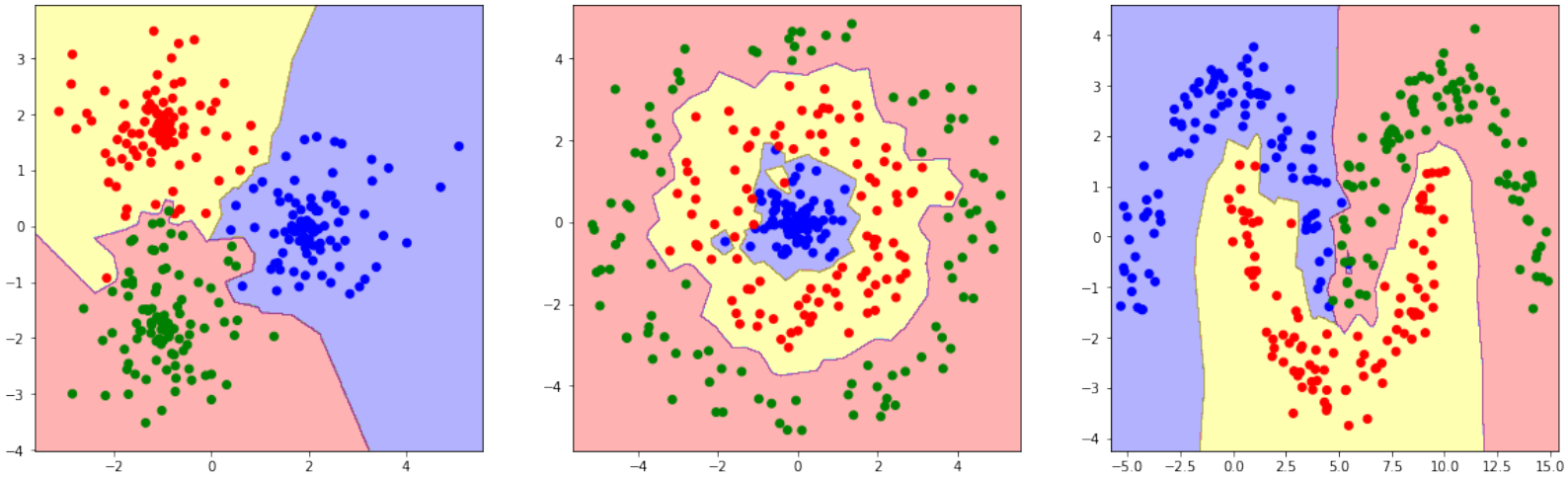
- Не нужно обучать (lazy-learning)

Недостатки:

- Не учитываются расстояния
- Возможно неоднозначное определение класса

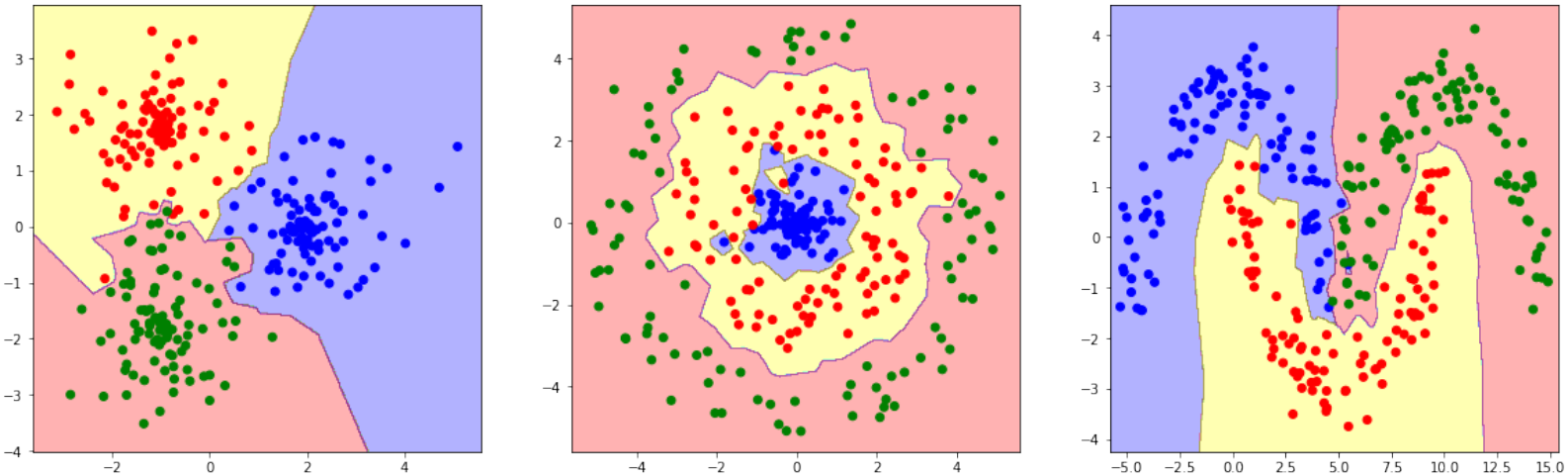
Метод k ближайших соседей (kNN)

$\mathrm{kNN}(k = 1)$

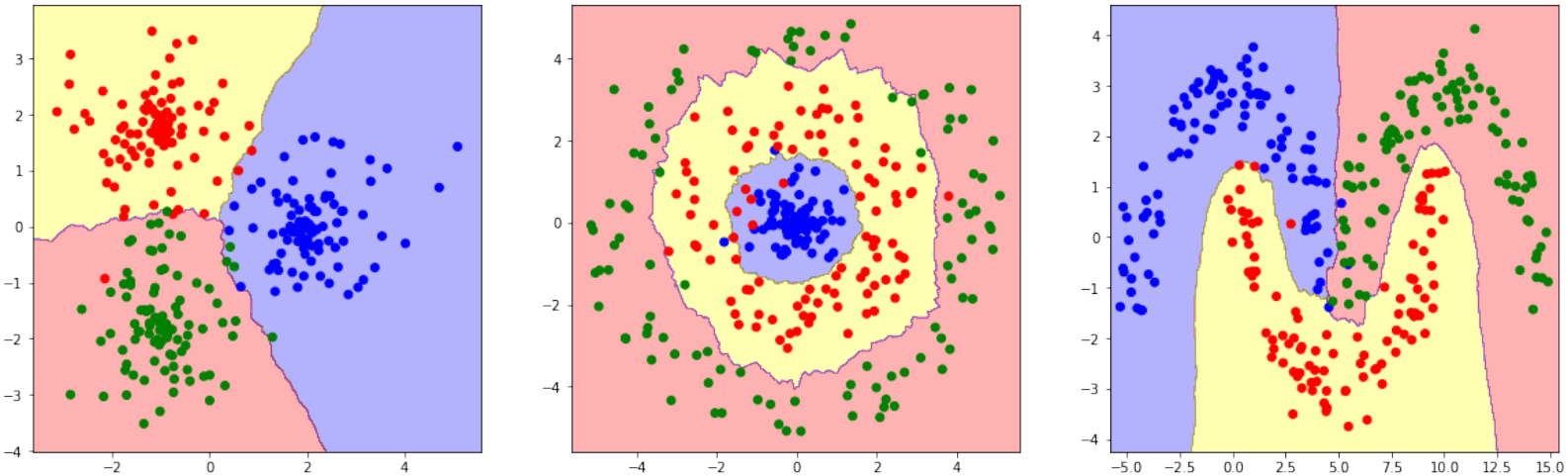


Метод k ближайших соседей (kNN)

kNN($k = 1$)

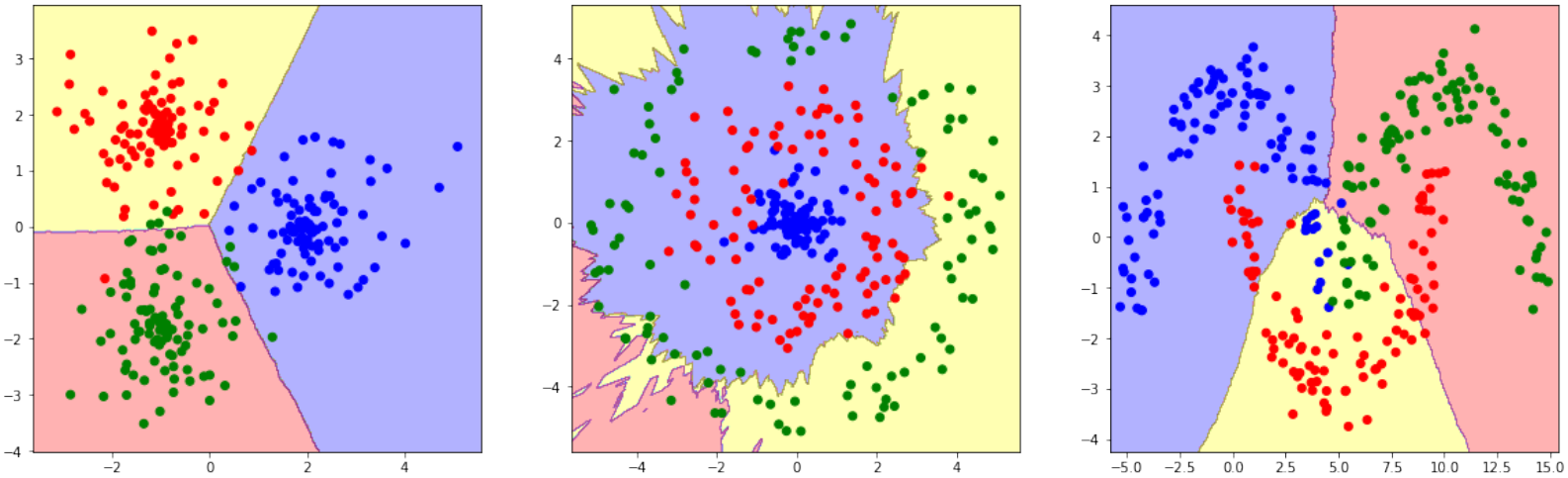


kNN($k = 20$): с ростом k границы сглаживаются

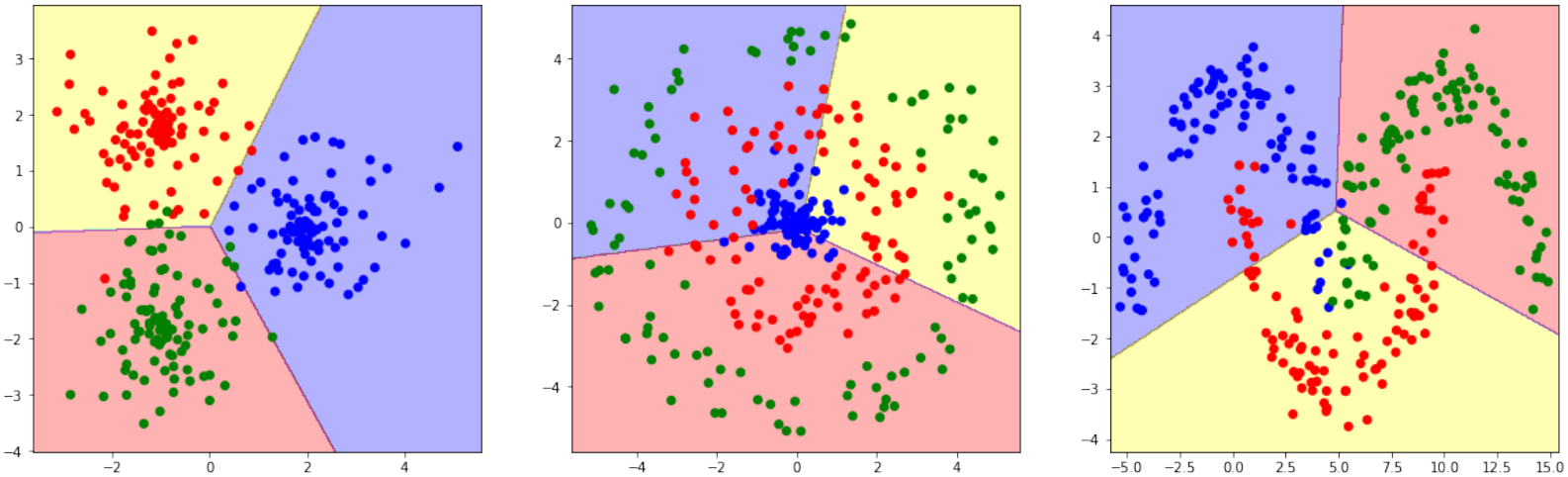


Метод k ближайших соседей (kNN)

kNN($k = 110$): с ростом k форма границ приближается к прямой



Линейный классификатор



Метод k ближайших соседей с весами

1. Для выбранного $\vec{x}$ расстояния сортируются по возрастанию: $\rho(\vec{x}, \vec{x}_i) \leq \rho(\vec{x}, \vec{x}_{i+1})$
2. Близость объекта $\vec{x}$ к классу y:
$$\varrho_{\text{linear}}(\vec{x}, y) = \sum_{i=1}^{k} [y_i = y] \frac{k+1-i}{k} \quad \text{или} \quad \varrho_{\text{exp}}(\vec{x}, y) = \sum_{i=1}^{k} [y_i = y] \alpha^i$$
3. $a(\vec{x}) = \arg\max_y \varrho(\vec{x}, y)$

Метод k ближайших соседей с весами

1. Для выбранного $\vec{x}$ расстояния сортируются по возрастанию: $\rho(\vec{x}, \vec{x}_i) \leq \rho(\vec{x}, \vec{x}_{i+1})$
2. Близость объекта $\vec{x}$ к классу y:
$$\varrho_{\text{linear}}(\vec{x}, y) = \sum_{i=1}^{k} [y_i = y] \frac{k+1-i}{k} \quad \text{или} \quad \varrho_{\exp}(\vec{x}, y) = \sum_{i=1}^{k} [y_i = y] \alpha^i$$
3. $a(\vec{x}) = \arg\max\limits_{y} \varrho(\vec{x}, y)$

Недостатки:

- Веса зависят от номера, но не от растояний $\rho(\vec{x}, \vec{x}_i)$
- Возможно неоднозначное определение класса для ϱ_{linear}

Метод парзеновского окна

1. Вводится функция-ядро $K(\rho)$, которая
   - положительна на $[0; 1]$
   - не возрастает на $[0; 1]$
2. Близость объекта $\vec{x}$ к классу y:

$$\varrho_{\text{fix}}(\vec{x}, y, h) = \sum_{i=1}^{L} [y_i = y] K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{h}\right) \text{ или}$$

$$\varrho_{\text{var}}(\vec{x}, y, k) = \sum_{i=1}^{L} [y_i = y] K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{\rho(\vec{x}, \vec{x}_{k+1})}\right)$$

3. $a(\vec{x}) = \arg\max\limits_{y} \varrho(\vec{x}, y)$

Метод парзеновского окна

1. Вводится функция-ядро $K(\rho)$, которая
   - положительна на $[0; 1]$
   - не возрастает на $[0; 1]$
2. Близость объекта $\vec{x}$ к классу y:

$$\varrho_{\text{fix}}(\vec{x}, y, h) = \sum_{i=1}^{L} [y_i = y] K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{h}\right) \text{ или}$$

$$\varrho_{\text{var}}(\vec{x}, y, k) = \sum_{i=1}^{L} [y_i = y] K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{\rho(\vec{x}, \vec{x}_{k+1})}\right)$$

3. $a(\vec{x}) = \arg\max\limits_{y} \varrho(\vec{x}, y)$

Обобщение – метод потенциальных функций

$$\varrho_{\text{pot}}(\vec{x}, y) = \sum_{i=1}^{L} [y_i = y] \underbrace{\cdot A_i \cdot}_{\text{Амплитуда } i\text{-го объекта}} K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{\underbrace{h_i}_{\text{“Радиус действия” } i\text{-го объекта}}}\right)$$

Сравнение с линейным классификатором

Алгоритм классификации методом потенциальных функций

$$a(\vec{x}, \vec{A}, \vec{h}) = \arg\max_y \sum_{i=1}^{L} [y_i = y] A_i K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{h_i}\right)$$

Алгоритм для линейного классификатора

$$a(\vec{x}, \boldsymbol{\theta}) = \arg\max_y \sum_{j=1}^{n} \theta_{y,j} f_j(\vec{x})$$

Сравнение с линейным классификатором

Алгоритм классификации методом потенциальных функций

$$a(\vec{x}, \vec{A}, \vec{h}) = \arg\max_y \sum_{i=1}^{L} [y_i = y] A_i K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{h_i}\right)$$

Алгоритм для линейного классификатора

$$a(\vec{x}, \boldsymbol{\theta}) = \arg\max_y \sum_{j=1}^{n} \theta_{y,j} f_j(\vec{x})$$

Две задачи эквивалентны при

$$\begin{cases} f_j(\vec{x}) = K\left(\dfrac{\rho(\vec{x}, \vec{x}_j)}{h_j}\right); \\ \theta_{y,j} = A_j[y_j = y]; \\ n = L. \end{cases}$$

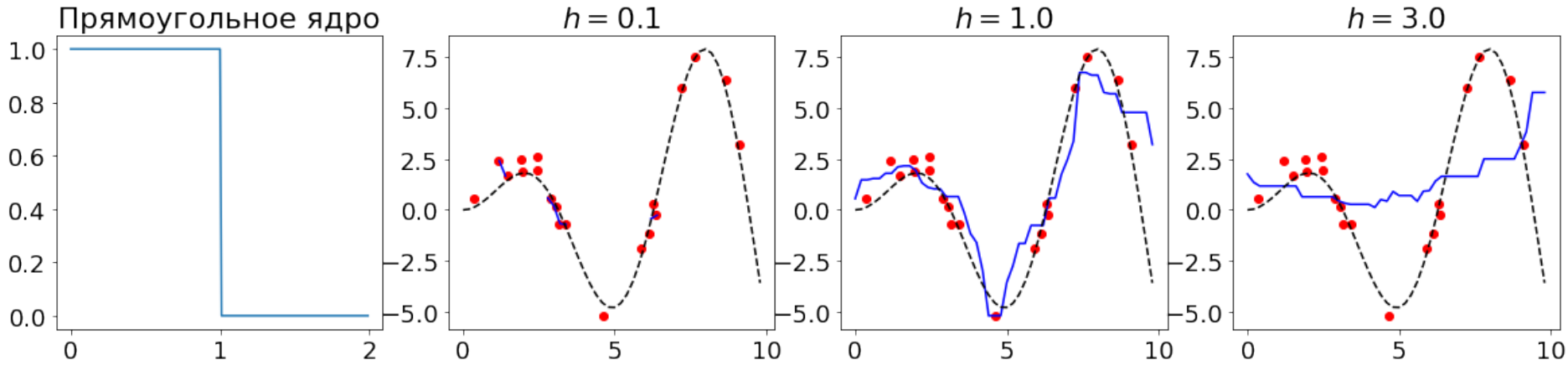
Задача регрессии. Формула Надарая-Уотсона

- Если расстояние между объектами мало, то значения их целевого признака близки
- Для вычисления значения функции следует использовать средневзвешенные значения на обучающей выборке $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

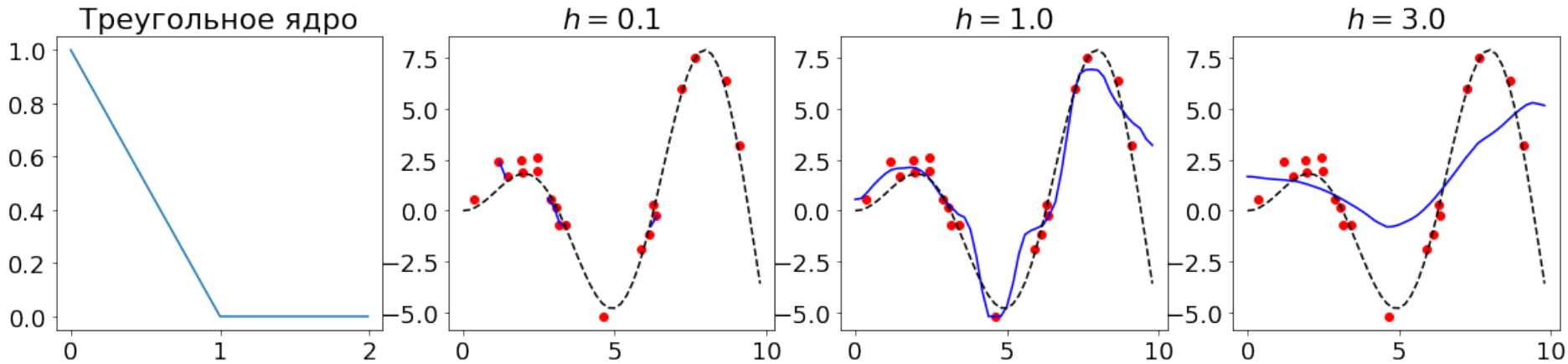
$$a(\vec{x}) = \frac{\sum\limits_{i=1}^{L} w_i y_i}{\sum\limits_{i=1}^{L} w_i}, \text{ где веса } w_i = K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{h_i}\right)$$

Формула Надарая-Уотсона. Примеры ядер

$$K_{\text{rect}}(\rho) = [\rho < 1]$$

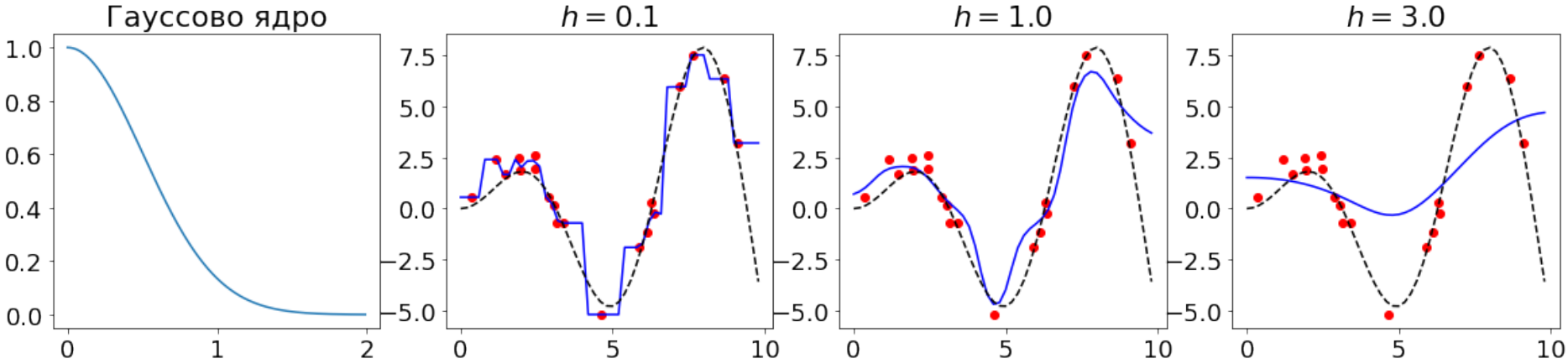


$$K_{\text{tri}}(\rho) = \max(0, 1 - \rho)$$

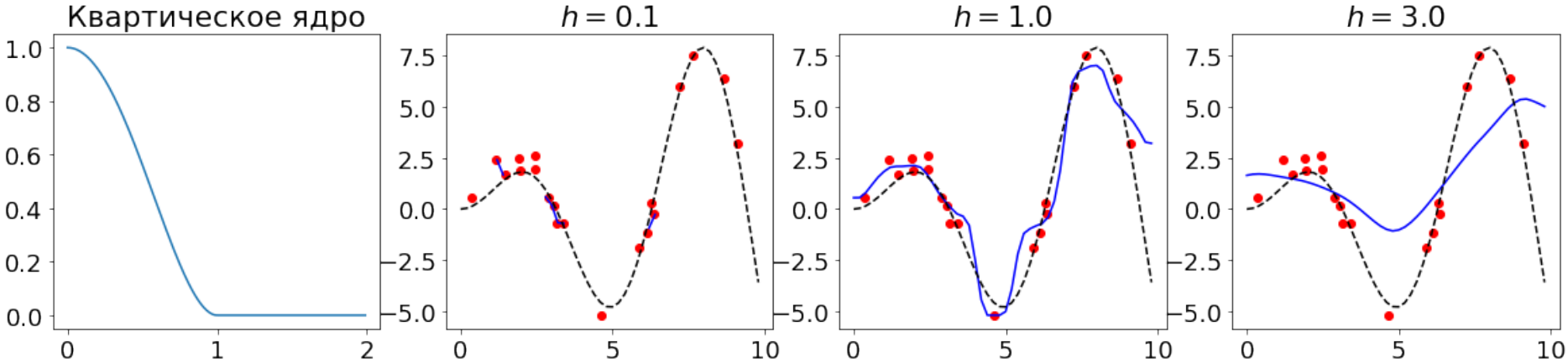


Формула Надарая-Уотсона. Примеры ядер

$$K_{\text{gauss}}(\rho) = e^{-2\rho^2}$$



$$K_{\text{quart}}(\rho) = (1-\rho^2)^2 \cdot [\rho < 1]$$



Метрические методы классификации и регрессии

Jupyter notebook “Метрические методы классификации и регрессии”:
`https://colab.research.google.com/drive/1Qou_pkTSNZZREDvxZ4_buIGy7gy7WFnZ`

Лекция 9. Метод опорных векторов

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Задача линейной бинарной классификации

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \{-1; 1\}$$

- Выбирается линейная модель: $a(\vec{x}, \vec{\theta}) = \text{sgn}(\vec{x}; \vec{\theta}) = \text{sgn} \sum_{j=1}^{n} x_j \theta_j$
- Бинарная функция потерь: $\mathcal{L}(a, \vec{x}, y) = [ay < 0]$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}) = \frac{1}{L} \sum_{i=1}^{L} \left[a(\vec{x}_i; \vec{\theta}) y_i < 0 \right] = \frac{1}{L} \sum_{i=1}^{L} \left[\underbrace{(\vec{x}_i; \vec{\theta}) y_i}_{\text{отступ объекта}} < 0 \right] \underset{\vec{\theta}}{\to} \min$$

- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}) = \frac{1}{K} \sum_{i=1}^{K} \left[(\vec{x}_i; \vec{\theta}) y_i < 0 \right]$$

Задача линейной бинарной классификации

Свободный член учитывается отдельно

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \{-1; 1\}$$

- Выбирается линейная модель: $a(\vec{x}, \vec{\theta}) = \text{sgn}\left((\vec{x}; \vec{\theta}) - \theta_0\right)$
- Бинарная функция потерь: $\mathcal{L}(a, \vec{x}, y) = [ay < 0]$
- Ставится задача оптимизации при $\vec{x}_i \in \mathbb{X}_{\text{train}}$:

$$Q_{\text{train}}(\vec{\theta}, \theta_0) = \frac{1}{L} \sum_{i=1}^{L} \left[\underbrace{((\vec{x}_i; \vec{\theta}) - \theta_0) y_i}_{\text{отступ объекта}} < 0 \right] \underset{\vec{\theta}, \theta_0}{\longrightarrow} \min$$

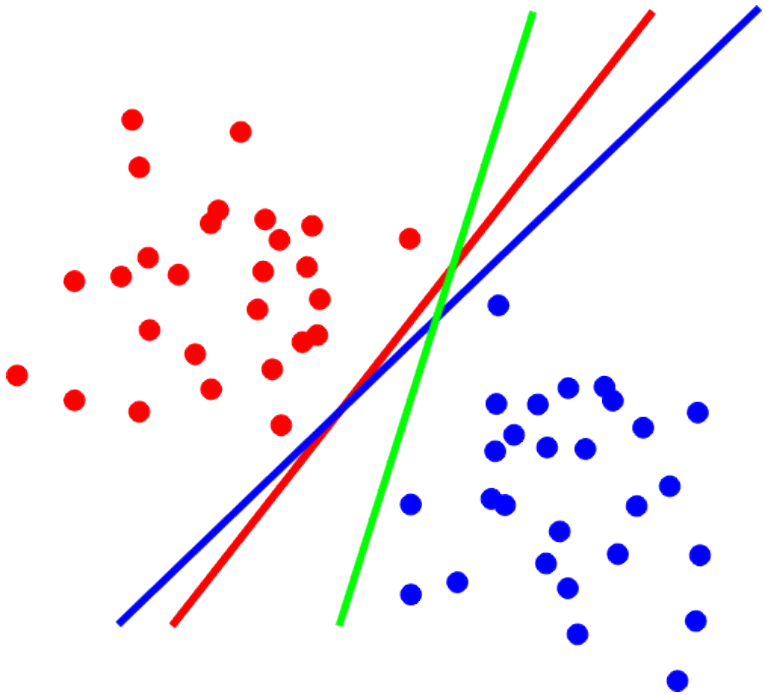
- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$:

$$Q_{\text{val}}(\vec{\theta}, \theta_0) = \frac{1}{K} \sum_{i=1}^{K} \left[((\vec{x}_i; \vec{\theta}) - \theta_0) y_i < 0 \right]$$

Задача линейной бинарной классификации

Случай линейно разделимой выборки

Как разделить выборку на классы “наилучшим” образом?

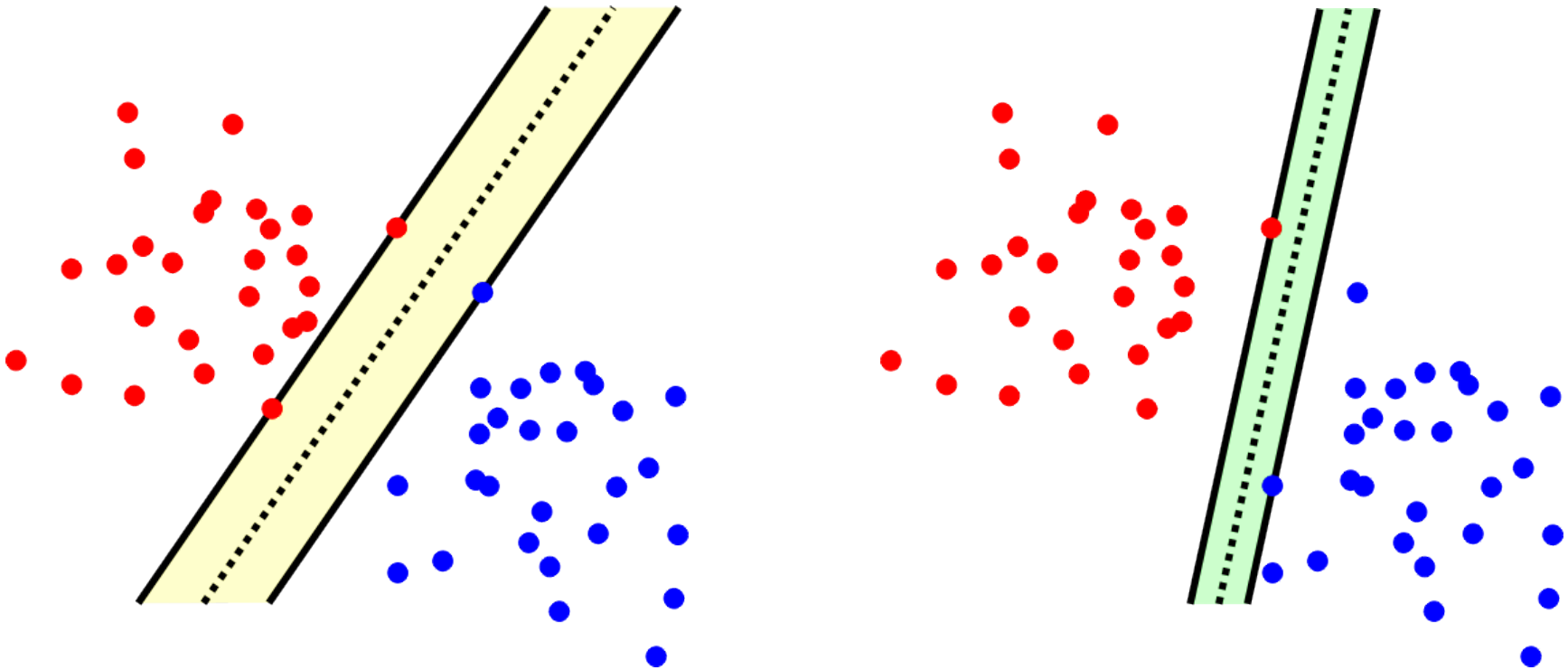


Существует неоднозначность построения гиперплоскости

Метод опорных векторов

Случай линейно разделимой выборки

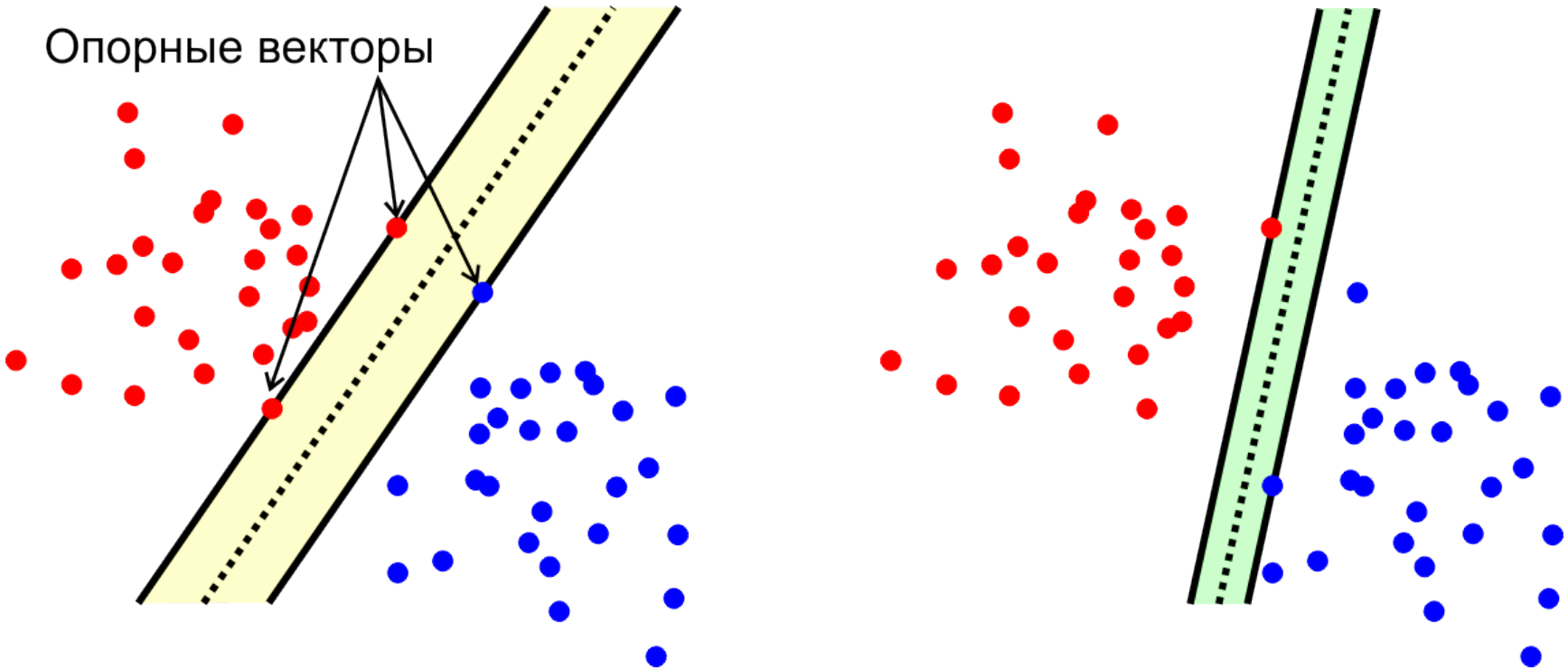
Идея: провести гиперплоскость так, чтобы расстояния от нее до ближайших объектов каждого класса были максимальны.



Метод опорных векторов

Случай линейно разделимой выборки

Идея: провести гиперплоскость так, чтобы расстояния от нее до ближайших объектов каждого класса были максимальны.



Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0)y_i$

Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0) y_i$
- Если выборка линейно разделима, то все объекты классифицируются правильно

$$\min M_i = m > 0$$

Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0) y_i$
- Если выборка линейно разделима, то все объекты классифицируются правильно

$$\min M_i = m > 0$$

- Для объектов $\vec{x}_{+1}$ и $\vec{x}_{-1}$, лежащих на границах полосы, выполнено

$$(\vec{x}_{+1}; \vec{\theta}) - \theta_0 = m; \quad (\vec{x}_{-1}; \vec{\theta}) - \theta_0 = -m$$

Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0) y_i$
- Если выборка линейно разделима, то все объекты классифицируются правильно

$$\min M_i = m > 0$$

- Для объектов $\vec{x}_{+1}$ и $\vec{x}_{-1}$, лежащих на границах полосы, выполнено

$$(\vec{x}_{+1}; \vec{\theta}) - \theta_0 = m; \quad (\vec{x}_{-1}; \vec{\theta}) - \theta_0 = -m$$

- Тогда ширина полосы равна

$$S = \frac{(\vec{\theta}(\vec{x}_{+1} - \vec{x}_{-1}))}{\left\| \vec{\theta} \right\|}$$

Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0) y_i$
- Если выборка линейно разделима, то все объекты классифицируются правильно

$$\min M_i = m > 0$$

- Для объектов $\vec{x}_{+1}$ и $\vec{x}_{-1}$, лежащих на границах полосы, выполнено

$$(\vec{x}_{+1}; \vec{\theta}) - \theta_0 = m; \quad (\vec{x}_{-1}; \vec{\theta}) - \theta_0 = -m$$

- Тогда ширина полосы равна

$$S = \frac{(\vec{\theta}(\vec{x}_{+1} - \vec{x}_{-1}))}{\left\|\vec{\theta}\right\|} = \frac{2m}{\left\|\vec{\theta}\right\|}$$

Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0) y_i$
- Если выборка линейно разделима, то все объекты классифицируются правильно

$$\min M_i = m > 0$$

- Для объектов $\vec{x}_{+1}$ и $\vec{x}_{-1}$, лежащих на границах полосы, выполнено

$$(\vec{x}_{+1}; \vec{\theta}) - \theta_0 = m; \quad (\vec{x}_{-1}; \vec{\theta}) - \theta_0 = -m$$

- Тогда ширина полосы равна

$$S = \frac{(\vec{\theta}(\vec{x}_{+1} - \vec{x}_{-1}))}{\left\|\vec{\theta}\right\|} = \frac{2m}{\left\|\vec{\theta}\right\|} \to \max$$

Метод опорных векторов

Случай линейно разделимой выборки

- Для каждого объекта вычисляется его отступ $M_i = ((\vec{x}_i; \vec{\theta}) - \theta_0) y_i$
- Если выборка линейно разделима, то все объекты классифицируются правильно

$$\min M_i = m > 0$$

- Для объектов $\vec{x}_{+1}$ и $\vec{x}_{-1}$, лежащих на границах полосы, выполнено

$$(\vec{x}_{+1}; \vec{\theta}) - \theta_0 = m; \quad (\vec{x}_{-1}; \vec{\theta}) - \theta_0 = -m$$

- Тогда ширина полосы равна

$$S = \frac{(\vec{\theta}(\vec{x}_{+1} - \vec{x}_{-1}))}{\left\|\vec{\theta}\right\|} = \frac{2m}{\left\|\vec{\theta}\right\|} \to \max \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases}$$

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\| \vec{\theta} \right\| \to \min; \\ \min M_i = 1 \end{cases}$$

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases} \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ M_i \geq 1 \end{cases}$$

- Для линейно неразделимой выборки

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases} \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ M_i \geq 1 \end{cases}$$

- Для линейно неразделимой выборки вводятся параметры ϵ_i:

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases} \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ M_i \geq 1 \end{cases}$$

- Для линейно неразделимой выборки вводятся параметры ϵ_i:

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ M_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases}$$

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases} \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ M_i \geq 1 \end{cases}$$

- Для линейно неразделимой выборки вводятся параметры ϵ_i:

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ M_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases} \Rightarrow \begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ \epsilon_i \geq \max\left(0; 1 - M_i\right) \end{cases}$$

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases} \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ M_i \geq 1 \end{cases}$$

- Для линейно неразделимой выборки вводятся параметры ϵ_i:

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ M_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases} \Rightarrow \begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ \epsilon_i \geq \max\left(0; 1 - M_i\right) = \left[1 - M_i\right]_+ \end{cases}$$

Метод опорных векторов

- Для линейно разделимой выборки:

$$\begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ \min M_i = 1 \end{cases} \Rightarrow \begin{cases} \left\|\vec{\theta}\right\| \to \min; \\ M_i \geq 1 \end{cases}$$

- Для линейно неразделимой выборки вводятся параметры ϵ_i:

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum_{i=1}^{L}\epsilon_i \to \min; \\ M_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases} \Rightarrow \begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum_{i=1}^{L}\epsilon_i \to \min; \\ \epsilon_i \geq \max(0; 1 - M_i) = [1 - M_i]_+ \end{cases}$$

$$\frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum_{i=1}^{L}[1 - M_i]_+ \to \min_{\vec{\theta},\theta_0}$$

Задача линейной бинарной классификации

Сравнение методов

- Пороговая классификация

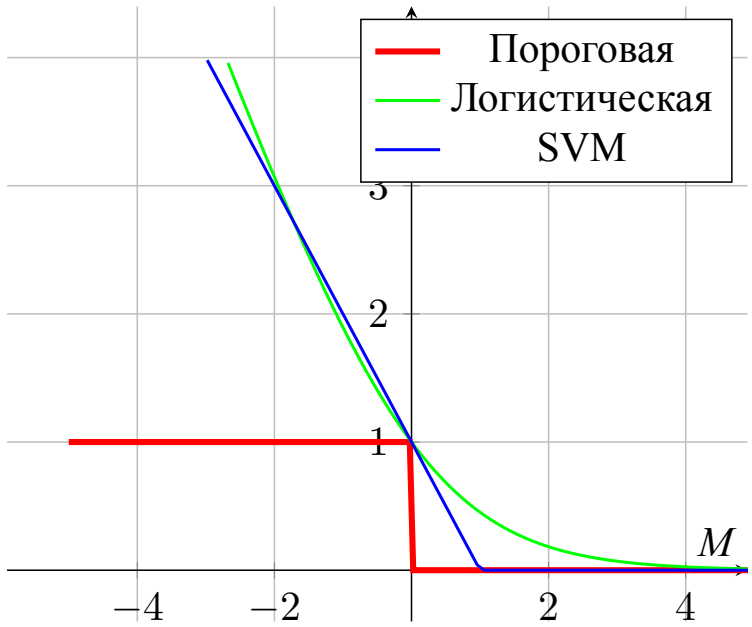
$$\frac{1}{L}\sum_{i=1}^{L}[M_i < 0] \underset{\vec{\theta}}{\to} \min$$

- Логистическая регрессия

$$\frac{1}{L}\sum_{i=1}^{L}\log_2\left(1+e^{-M_i}\right) \underset{\vec{\theta}}{\to} \min$$

- Метод опорных векторов

$$\frac{1}{L}\sum_{i=1}^{L}[1-M_i]_+ \underset{\vec{\theta},\theta_0}{\longrightarrow} \min$$



+ регуляризация

Условия Каруша-Куна-Таккера

- Задача условной минимизации

$$\begin{cases} f(x) \to \min; \\ g_i(x) \le 0, \quad i = 1, 2, \ldots, m; \\ h_j(x) = 0, \quad j = 1, 2, \ldots, k. \end{cases}$$

- Вводится функция Лагранжа

$$L(x; \vec{\mu}, \vec{\lambda}) = f(x) + \sum_{i=1}^{m} \mu_i g_i(x) + \sum_{j=1}^{k} \lambda_j h_j(x)$$

Если x – локальный минимум, то

$$\exists \mu_i; \lambda_j : \begin{cases} \frac{\partial L}{\partial x} = 0; & \\ g_i(x) \le 0; \quad h_j(x) & \text{– исходные ограничения} \\ \mu_i \ge 0; & \text{– двойственные ограничения} \\ \mu_i g_i(x) = 0. & \text{– условия дополняющей нежесткости} \end{cases}$$

Метод опорных векторов

Применение условий Каруша-Куна-Таккера

- Исходная задача минимизации

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ M_i = ((\vec{x}_i;\vec{\theta}) - \theta_0)y_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases}$$

Метод опорных векторов

Применение условий Каруша-Куна-Таккера

- Исходная задача минимизации

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ M_i = (\langle\vec{x}_i;\vec{\theta}\rangle - \theta_0)y_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases} \Rightarrow \begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{L}\epsilon_i \to \min; \\ 1 - \epsilon_i - M_i \leq 0; \\ -\epsilon_i \leq 0 \end{cases}$$

Метод опорных векторов

Применение условий Каруша-Куна-Таккера

- Исходная задача минимизации

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum_{i=1}^{L}\epsilon_i \to \min; \\ M_i = ((\vec{x}_i;\vec{\theta}) - \theta_0)y_i \geq 1 - \epsilon_i; \\ \epsilon_i \geq 0 \end{cases} \Rightarrow \begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum_{i=1}^{L}\epsilon_i \to \min; \\ 1 - \epsilon_i - M_i \leq 0; \\ -\epsilon_i \leq 0 \end{cases}$$

- Функция Лагранжа

$$L(\vec{\theta},\theta_0;\vec{\epsilon};\vec{\mu},\vec{\lambda}) = \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum_{i=1}^{L}\epsilon_i + \sum_{i=1}^{L}\mu_i(1-\epsilon_i-M_i) - \sum_{i=1}^{L}\nu_i\epsilon_i =$$

$$= \frac{1}{2}\left\|\vec{\theta}\right\|^2 - \sum_{i=1}^{L}\mu_i((\vec{x}_i;\vec{\theta})y_i - \theta_0 y_i - 1) - \sum_{i=1}^{L}\epsilon_i(\mu_i + \nu_i - C)$$

Метод опорных векторов

Применение условий Каруша-Куна-Таккера

$$L(\vec{\theta}, \theta_0; \vec{\epsilon}; \vec{\mu}, \vec{\lambda}) = \frac{1}{2}\left\|\vec{\theta}\right\|^2 - \sum_{i=1}^{L} \mu_i((\vec{x}_i; \vec{\theta})y_i - \theta_0 y_i - 1) - \sum_{i=1}^{L} \epsilon_i(\mu_i + \nu_i - C)$$

$$\begin{cases} \dfrac{\partial L}{\partial \vec{\theta}} = 0 & \Rightarrow \vec{\theta} = \sum\limits_{i=1}^{L} \mu_i y_i \vec{x}_i; \\ \dfrac{\partial L}{\partial \vec{\theta}_0} = 0 & \Rightarrow \sum\limits_{i=1}^{L} \mu_i y_i = 0; \\ \dfrac{\partial L}{\partial \epsilon_i} = 0 & \Rightarrow \mu_i + \nu_i = C; \\ \epsilon_i \geq 0; \quad \mu_i \geq 0; \quad \nu_i \geq 0; \\ \mu_i(1 - \epsilon_i - M_i) = 0; \\ \nu_i \epsilon_i = 0. \end{cases}$$

Опорные и не опорные объекты

$$\begin{cases} \vec{\theta} = \sum\limits_{i=1}^{L} \mu_i y_i \vec{x}_i; \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0; \\ \mu_i + \nu_i = C; \\ \epsilon_i \geq 0; \quad \mu_i \geq 0; \quad \nu_i \geq 0; \\ \mu_i(1 - \epsilon_i - M_i) = 0; \\ \nu_i \epsilon_i = 0. \end{cases}$$

Если $\mu_i = 0$, то

$$\begin{cases} \nu_i = C; \\ \epsilon_i = 0; \\ M_i \geq 1; \\ \vec{\theta} \text{ не зависит от } i\text{-го объекта} \end{cases}$$

Это **не опорный** объект, лежащий в глубине своего класса

Опорные и не опорные объекты

$$\begin{cases} \vec{\theta} = \sum\limits_{i=1}^{L} \mu_i y_i \vec{x}_i; \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0; \\ \mu_i + \nu_i = C; \\ \epsilon_i \geq 0; \quad \mu_i \geq 0; \quad \nu_i \geq 0; \\ \mu_i(1 - \epsilon_i - M_i) = 0; \\ \nu_i \epsilon_i = 0. \end{cases}$$

Если $0 < \mu_i < C$, то

$$\begin{cases} \nu_i = C - \mu_i > 0; \\ \epsilon_i = 0; \\ M_i = 1; \\ \vec{\theta} \text{ зависит от } i\text{-го объекта} \end{cases}$$

Это **опорный** граничный объект

Опорные и не опорные объекты

$$\begin{cases} \vec{\theta} = \sum\limits_{i=1}^{L} \mu_i y_i \vec{x}_i; \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0; \\ \mu_i + \nu_i = C; \\ \epsilon_i \geq 0; \quad \mu_i \geq 0; \quad \nu_i \geq 0; \\ \mu_i(1 - \epsilon_i - M_i) = 0; \\ \nu_i \epsilon_i = 0. \end{cases}$$

Если $\mu_i = C$, то

$$\begin{cases} \nu_i = 0; \\ \epsilon_i > 0; \\ M_i = 1 - \epsilon_i; \\ \vec{\theta} \text{ зависит от } i\text{-го объекта} \end{cases}$$

Это **опорный** объект, лежащий в глубине чужого класса или внутри разделяющей полосы

Двойственная задача

$$L(\vec{\theta}, \theta_0; \vec{\epsilon}; \vec{\mu}, \vec{\lambda}) = \frac{1}{2}\left\|\vec{\theta}\right\|^2 - \sum_{i=1}^{L} \mu_i((\vec{x}_i; \vec{\theta})y_i - \theta_0 y_i - 1) - \sum_{i=1}^{L} \epsilon_i(\mu_i + \nu_i - C)$$

$$\begin{cases} \vec{\theta} = \sum\limits_{i=1}^{L} \mu_i y_i \vec{x}_i; \qquad \sum\limits_{i=1}^{L} \mu_i y_i = 0; \qquad \mu_i + \nu_i = C; \\ \epsilon_i \geq 0; \quad \mu_i \geq 0; \quad \nu_i \geq 0; \qquad \mu_i(1 - \epsilon_i - M_i) = 0; \qquad \nu_i \epsilon_i = 0. \end{cases}$$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2} \sum\limits_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum\limits_{i=1}^{L} \mu_i \to \min \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0; \\ 0 \leq \mu_i \leq C. \end{cases}$$

Решение задачи методом опорных векторов

1. Постановка двойственной задачи относительно $\vec{\mu}$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2}\sum\limits_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum\limits_{i=1}^{L} \mu_i \to \min \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0 \\ 0 \le \mu_i \le C \end{cases}$$

Решение задачи методом опорных векторов

1. Постановка двойственной задачи относительно $\vec{\mu}$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2}\sum\limits_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum\limits_{i=1}^{L} \mu_i \to \min \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0 \quad - \vec{\mu} \text{ лежит на } L\text{-мерной гиперплоскости;} \\ 0 \le \mu_i \le C \end{cases}$$

Решение задачи методом опорных векторов

1. Постановка двойственной задачи относительно $\vec{\mu}$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2}\sum\limits_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum\limits_{i=1}^{L} \mu_i \to \min \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0 \quad - \vec{\mu} \text{ лежит на } L\text{-мерной гиперплоскости;} \\ 0 \le \mu_i \le C \quad - \vec{\mu} \text{ лежит в } L\text{-мерном кубе со стороной } C. \end{cases}$$

Решение задачи методом опорных векторов

1. Постановка двойственной задачи относительно $\vec{\mu}$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2}\sum\limits_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum\limits_{i=1}^{L} \mu_i \to \min \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0 \quad - \vec{\mu} \text{ лежит на } L\text{-мерной гиперплоскости;} \\ 0 \le \mu_i \le C \quad - \vec{\mu} \text{ лежит в } L\text{-мерном кубе со стороной } C. \end{cases}$$

2. Двойственная задача – поиск минимума квадратичной формы на выпуклом множестве – имеет единственное решение и для этого существуют эффективные численные алгоритмы.

Решение задачи методом опорных векторов

1. Постановка двойственной задачи относительно $\vec{\mu}$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2}\sum\limits_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum\limits_{i=1}^{L} \mu_i \to \min \\ \sum\limits_{i=1}^{L} \mu_i y_i = 0 \quad - \vec{\mu} \text{ лежит на } L\text{-мерной гиперплоскости;} \\ 0 \le \mu_i \le C \quad - \vec{\mu} \text{ лежит в } L\text{-мерном кубе со стороной } C. \end{cases}$$

2. Двойственная задача – поиск минимума квадратичной формы на выпуклом множестве – имеет единственное решение и для этого существуют эффективные численные алгоритмы.

3. Переход к решению исходной задачи

$$\begin{cases} \vec{\theta} = \sum\limits_{i=1}^{L} \mu_i y_i \vec{x}_i; \\ \theta_0 = (\vec{\theta}, \vec{x}_b) - y_b; \quad \vec{x}_b - \text{граничный опорный объект, т.е. } 0 < \mu_b < C. \end{cases}$$

Решение задачи методом опорных векторов

1. Постановка двойственной задачи относительно $\vec{\mu}$

$$\begin{cases} -L(\vec{\mu}) = \dfrac{1}{2}\sum\limits_{i,j=1}^{L} \mu_i\mu_j y_i y_j (\vec{x}_i;\vec{x}_j) - \sum\limits_{i=1}^{L}\mu_i \to \min \\ \sum\limits_{i=1}^{L}\mu_i y_i = 0 \quad - \vec{\mu} \text{ лежит на } L\text{-мерной гиперплоскости;} \\ 0 \le \mu_i \le C \quad - \vec{\mu} \text{ лежит в } L\text{-мерном кубе со стороной } C. \end{cases}$$

2. Двойственная задача – поиск минимума квадратичной формы на выпуклом множестве – имеет единственное решение и для этого существуют эффективные численные алгоритмы.

3. Переход к решению исходной задачи

$$\begin{cases} \vec{\theta} = \sum\limits_{i=1}^{L}\mu_i y_i \vec{x}_i; & \Rightarrow a(\vec{x}) = \operatorname{sgn}\left(\sum\limits_{i=1}^{L}\mu_i y_i (\vec{x}_i,\vec{x}) - \theta_0\right) \\ \theta_0 = (\vec{\theta},\vec{x}_b) - y_b; & \vec{x}_b - \text{граничный опорный объект, т.е. } 0 < \mu_b < C. \end{cases}$$

Kernel trick

- Двойственная задача содержит лишь скалярные произведения $(\vec{x}_i; \vec{x}_j)$, но не сами признаки объектов:

$$-L(\vec{\mu}) = \frac{1}{2}\sum_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum_{i=1}^{L} \mu_i \to \min; \sum_{i=1}^{L} \mu_i y_i = 0;\ 0 \le \mu_i \le C.$$

Kernel trick

- Двойственная задача содержит лишь скалярные произведения $(\vec{x}_i; \vec{x}_j)$, но не сами признаки объектов:

$$-L(\vec{\mu}) = \frac{1}{2}\sum_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum_{i=1}^{L} \mu_i \to \min; \sum_{i=1}^{L} \mu_i y_i = 0;\ 0 \le \mu_i \le C.$$

- Классификатор исходной задачи зависит лишь от скалярных произведений $(\vec{x}_i, \vec{x})$, но не от самих признаков объектов:

$$a(\vec{x}) = \operatorname{sgn}\left(\sum_{i=1}^{L} \mu_i y_i (\vec{x}_i, \vec{x}) - \theta_0\right)$$

Kernel trick

- Двойственная задача содержит лишь скалярные произведения $(\vec{x}_i; \vec{x}_j)$, но не сами признаки объектов:

$$-L(\vec{\mu}) = \frac{1}{2}\sum_{i,j=1}^{L} \mu_i \mu_j y_i y_j (\vec{x}_i; \vec{x}_j) - \sum_{i=1}^{L} \mu_i \to \min; \sum_{i=1}^{L} \mu_i y_i = 0;\ 0 \le \mu_i \le C.$$

- Классификатор исходной задачи зависит лишь от скалярных произведений $(\vec{x}_i, \vec{x})$, но не от самих признаков объектов:

$$a(\vec{x}) = \operatorname{sgn}\left(\sum_{i=1}^{L} \mu_i y_i (\vec{x}_i, \vec{x}) - \theta_0\right)$$

- Можно решить задачу классификации, если можно вычислять скалярные произведения объектов.

Kernel trick

- Можно решить задачу классификации, если можно вычислять скаларные произведения объектов.

Kernel trick

- Можно решить задачу классификации, если можно вычислять скалярные произведения объектов.
- Можно заменить скалярные произведения исходных признаков $(\vec{x}_i; \vec{x}_j)$ на функцию-ядро $K(\vec{x}_i; \vec{x}_j) = \left(\vec{\phi}(\vec{x}_i); \vec{\phi}(\vec{x}_j)\right)$, обладающую свойствами скалярного произведения.

$$\vec{\phi} : \mathbb{X} \to \mathbb{H}; \quad \mathbb{H} \text{ – гильбертово пространство.}$$

Kernel trick

- Можно решить задачу классификации, если можно вычислять скалярные произведения объектов.
- Можно заменить скалярные произведения исходных признаков $(\vec{x}_i;\vec{x}_j)$ на функцию-ядро $K(\vec{x}_i;\vec{x}_j) = \left(\vec{\phi}(\vec{x}_i);\vec{\phi}(\vec{x}_j)\right)$, обладающую свойствами скалярного произведения.

$$\vec{\phi} : \mathbb{X} \to \mathbb{H}; \quad \mathbb{H} \text{ – гильбертово пространство.}$$

- Функция $K(\vec{x}_i;\vec{x}_j)$ – ядро, тогда и только тогда, когда

$$\begin{cases} K(\vec{x}_i;\vec{x}_j) = K(\vec{x}_j;\vec{x}_i); \\ \iint\limits_{\mathbb{X}} K(\vec{x}_1;\vec{x}_2) f(x_1) f(x_2) dx_1 dx_2 \geq 0 \quad \forall f : \mathbb{X} \to \mathbb{R}. \end{cases}$$

Часто используемые ядра

- Линейное ядро: $K_{\text{linear}}(\vec{x}_1; \vec{x}_2) = (\vec{x}_1; \vec{x}_2)$
- Полиномиальное ядро: $K_{\text{poly}}(\vec{x}_1; \vec{x}_2) = (\gamma(\vec{x}_1; \vec{x}_2) + r)^d$
- Гауссово ядро (радиальные базисные функции): $K_{\text{rbf}}(\vec{x}_1; \vec{x}_2) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right)$
- Сигмоидное ядро: $K_{\text{sigmoid}}(\vec{x}_1; \vec{x}_2) = \tanh\left(\gamma(\vec{x}_1; \vec{x}_2) + r\right)$

Размерность спрямляющего пространства

Квадратичное ядро

Пусть $K_2(\vec{a};\vec{b}) = (\vec{a};\vec{b})^2$

Размерность спрямляющего пространства

Квадратичное ядро

Пусть $K_2(\vec{a};\vec{b}) = (\vec{a};\vec{b})^2$

$$K_2(\vec{a};\vec{b}) = ((a_1,a_2);(b_1,b_2))^2 = (a_1b_1 + a_2b_2)^2 =$$
$$= a_1^2b_1^2 + a_2^2b_2^2 + 2a_1b_1a_2b_2 = ((a_1^2, a_2^2, \sqrt{2}a_1a_2), (b_1^2, b_2^2, \sqrt{2}b_1b_2))$$

Размерность спрямляющего пространства

Квадратичное ядро

Пусть $K_2(\vec{a};\vec{b}) = (\vec{a};\vec{b})^2$

$$K_2(\vec{a};\vec{b}) = ((a_1,a_2);(b_1,b_2))^2 = (a_1b_1 + a_2b_2)^2 =$$
$$= a_1^2b_1^2 + a_2^2b_2^2 + 2a_1b_1a_2b_2 = ((a_1^2, a_2^2, \sqrt{2}a_1a_2), (b_1^2, b_2^2, \sqrt{2}b_1b_2))$$

Это скалярное произведение в 3-мерном пространстве.

Размерность спрямляющего пространства

Квадратичное ядро

Пусть $K_2(\vec{a};\vec{b}) = (\vec{a};\vec{b})^2$

$$K_2(\vec{a};\vec{b}) = ((a_1,a_2);(b_1,b_2))^2 = (a_1b_1 + a_2b_2)^2 =$$
$$= a_1^2b_1^2 + a_2^2b_2^2 + 2a_1b_1a_2b_2 = ((a_1^2, a_2^2, \sqrt{2}a_1a_2), (b_1^2, b_2^2, \sqrt{2}b_1b_2))$$

Это скалярное произведение в 3-мерном пространстве.

В общем случае для степенного ядра $K_d(\vec{a};\vec{b}) = (\vec{a};\vec{b})^d$
размерность $\dim \mathbb{H} = C_{n+d-1}^d$

Размерность спрямляющего пространства

Гауссово ядро

Пусть $K_{\text{rbf}}(\vec{x}_1; \vec{x}_2) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right)$

Размерность спрямляющего пространства

Гауссово ядро

$$\text{Пусть } K_{\text{rbf}}(\vec{x}_1; \vec{x}_2) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right)$$

$$K_{\text{rbf}}(\vec{a}; \vec{b}) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right) = \exp\left(-\gamma(\|\vec{a}\|^2 + \left\|\vec{b}\right\|^2 - 2(\vec{a}, \vec{b}))\right) =$$

$$= \exp\left(-\gamma \|\vec{a}\|^2\right) \exp\left(-\gamma \left\|\vec{b}\right\|^2\right) \exp\left(2\gamma(\vec{a}, \vec{b})\right) =$$

$$= \exp\left(-\gamma \|\vec{a}\|^2\right) \exp\left(-\gamma \left\|\vec{b}\right\|^2\right) \sum_{k=0}^{\infty} \frac{(2\gamma)^k (\vec{a}, \vec{b})^k}{k!}$$

Размерность спрямляющего пространства

Гауссово ядро

$$\text{Пусть } K_{\text{rbf}}(\vec{x}_1; \vec{x}_2) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right)$$

$$K_{\text{rbf}}(\vec{a}; \vec{b}) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right) = \exp\left(-\gamma(\|\vec{a}\|^2 + \left\|\vec{b}\right\|^2 - 2(\vec{a}, \vec{b}))\right) =$$

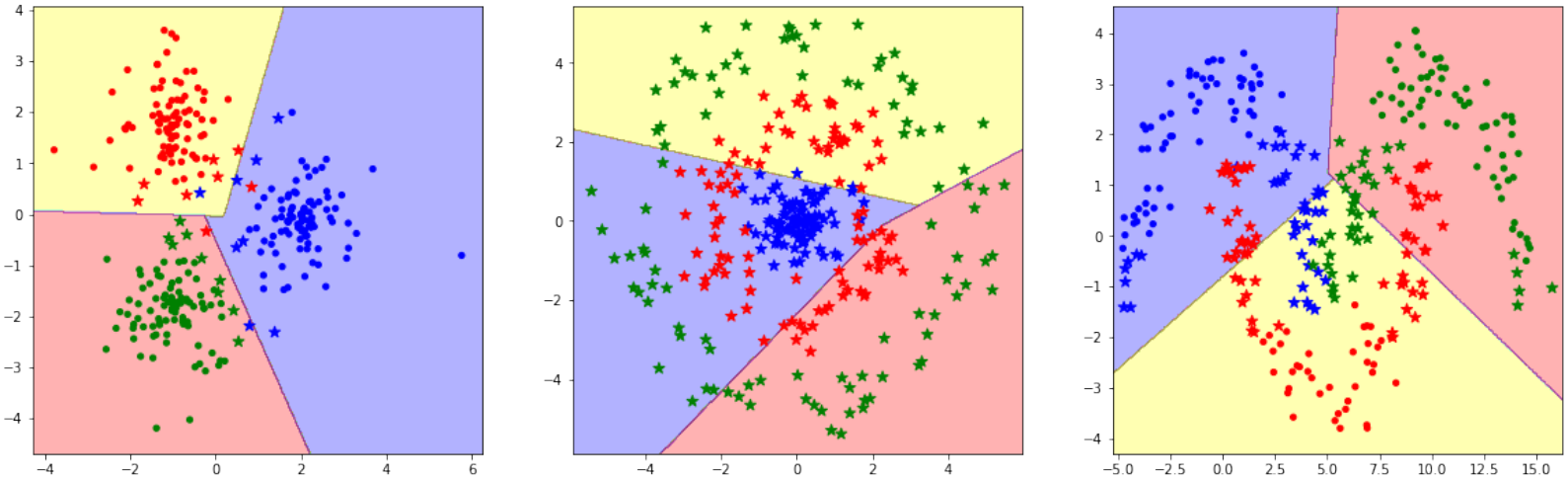
$$= \exp\left(-\gamma \|\vec{a}\|^2\right) \exp\left(-\gamma \left\|\vec{b}\right\|^2\right) \exp\left(2\gamma(\vec{a}, \vec{b})\right) =$$

$$= \exp\left(-\gamma \|\vec{a}\|^2\right) \exp\left(-\gamma \left\|\vec{b}\right\|^2\right) \sum_{k=0}^{\infty} \frac{(2\gamma)^k (\vec{a}, \vec{b})^k}{k!}$$

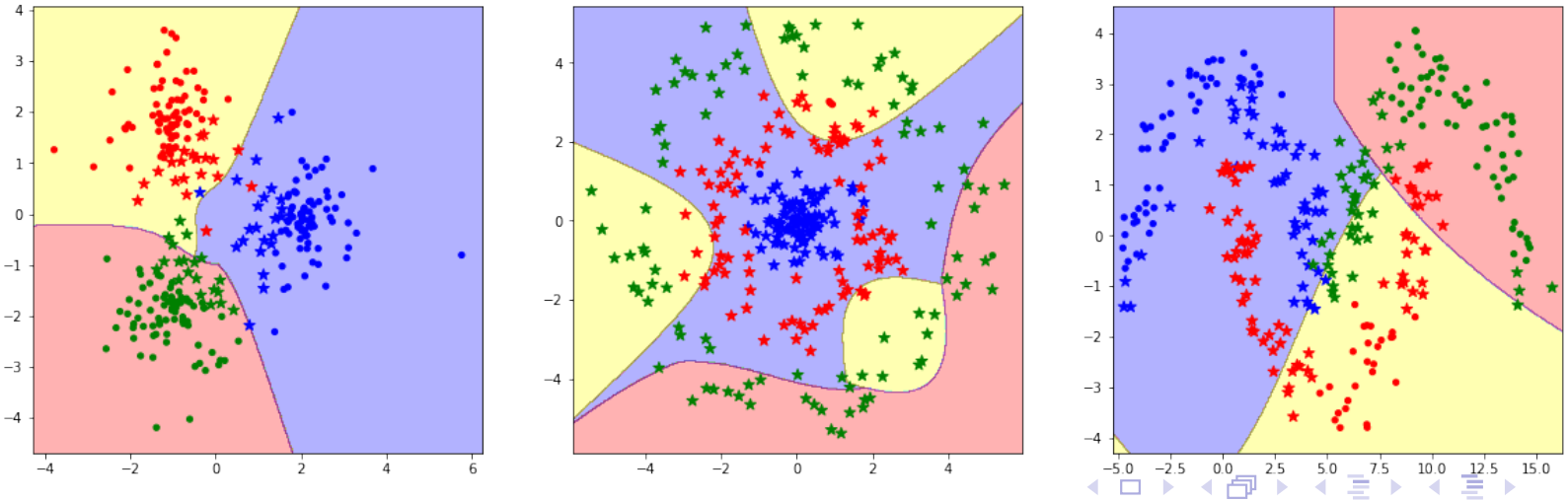
Пространство $\mathbb{H}$ имеет бесконечную размерность
из-за бесконечного числа членов ряда

Часто используемые ядра

Линейное ядро: $K_{\text{linear}}(\vec{x}_1; \vec{x}_2) = (\vec{x}_1; \vec{x}_2)$

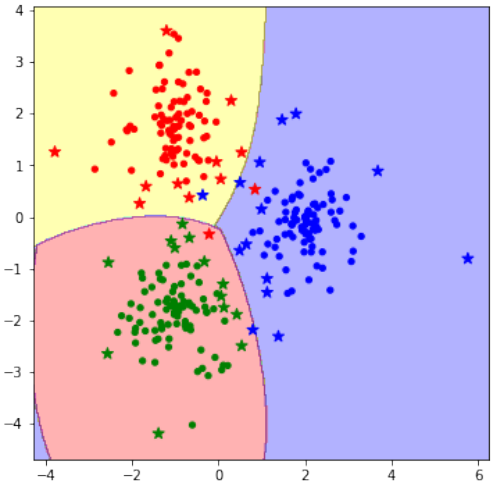
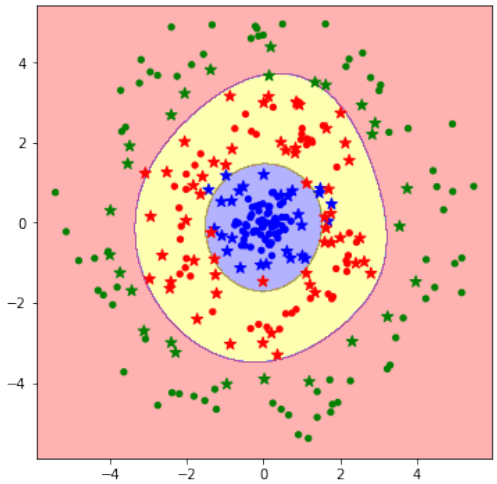
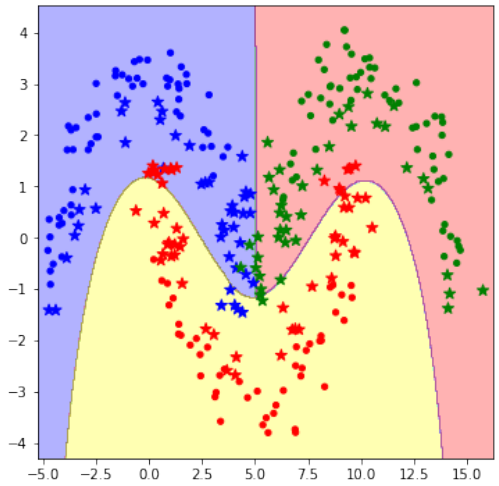


Полиномиальное ядро: $K_{\text{poly}}(\vec{x}_1; \vec{x}_2) = (\gamma(\vec{x}_1; \vec{x}_2) + r)^d$

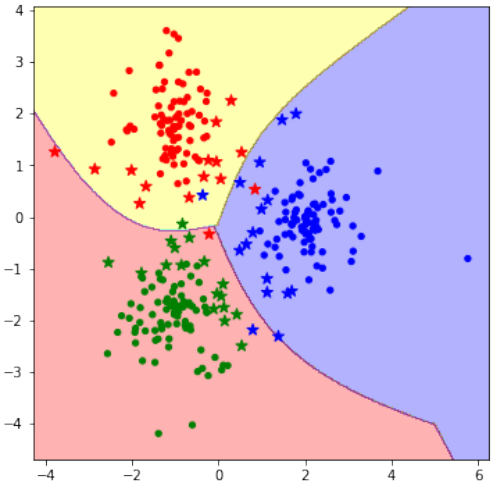
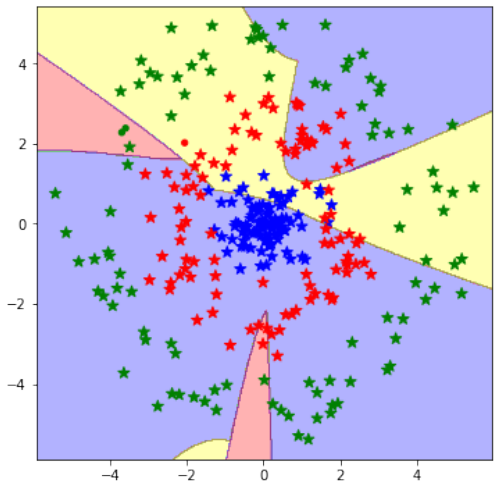
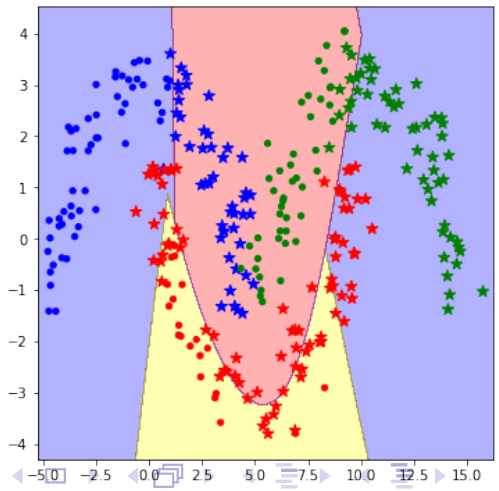


Часто используемые ядра

Гауссово ядро: $K_{\text{rbf}}(\vec{x}_1; \vec{x}_2) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right)$

Сигмоидное ядро: $K_{\text{sigmoid}}(\vec{x}_1; \vec{x}_2) = \tanh(\gamma(\vec{x}_1; \vec{x}_2) + r)$

SVM и метрические методы

Алгоритм для SVM

$$a(\vec{x}) = \operatorname{sgn}\left(\sum_{i=1}^{L} \mu_i y_i K_{\text{rbf}}(\vec{x}, \vec{x}_i) - \theta_0\right)$$

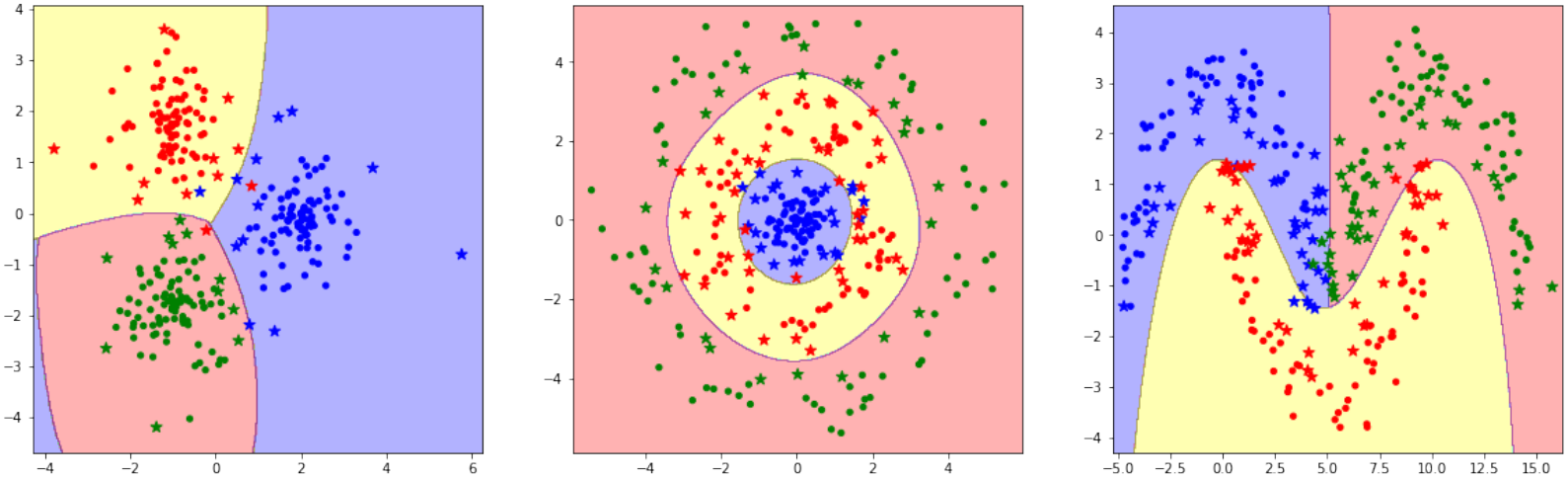
- $K_{\text{rbf}}(\vec{x}_1; \vec{x}_2) = \exp\left(-\gamma \left\|\vec{x}_1 - \vec{x}_2\right\|^2\right)$ – функция близости
- Уже определены амплитуды объектов μ_i и “радиусы” потенциальных функций
- Уже проведена селекция объектов, учитывается минимальное число “нужных” объектов

Получается алгоритм классификации методом потенциальных функций

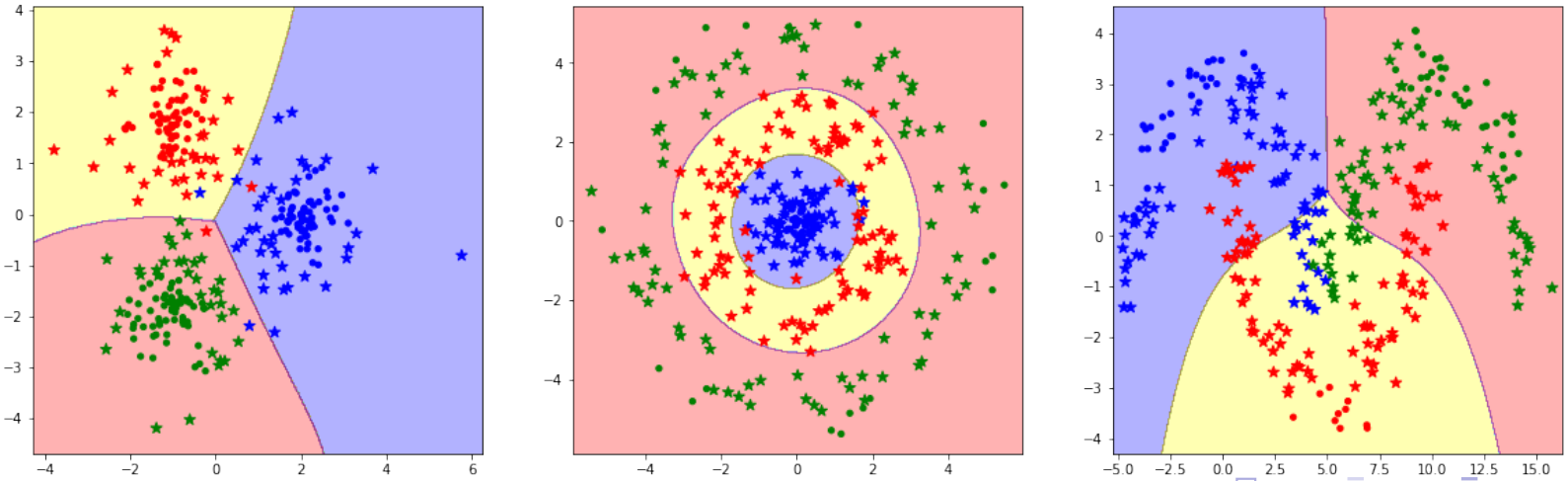
$$a(\vec{x}, \vec{A}, \vec{h}) = \arg\max_y \sum_{i=1}^{L} [y_i = y] A_i K\left(\frac{\rho(\vec{x}, \vec{x}_i)}{h_i}\right)$$

Разные значения параметра C

$C = 2.0$ – “слабая” регуляризация ($\alpha = 2/C = 0.5$), много объектов



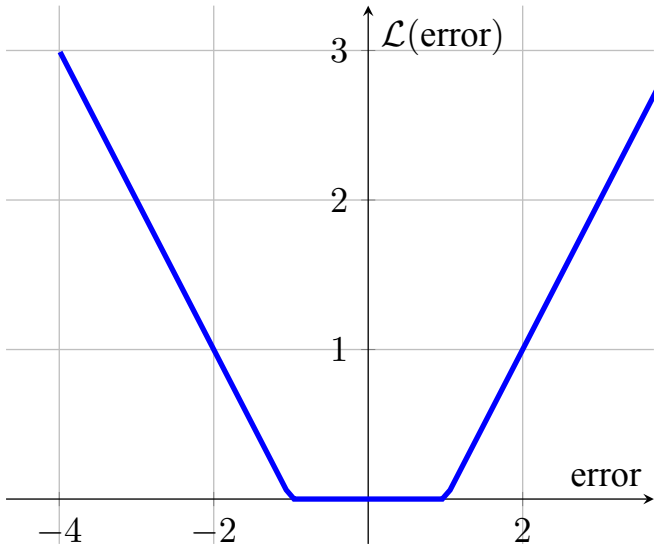
$C = 0.1$ – “сильная” регуляризация ($\alpha = 2/C = 10.0$), много объектов



SVM регрессия

$$\vec{x}_i \in \mathcal{R}^n; y_i \in \mathcal{R}$$

- Выбирается модель: $a(\vec{x}, \vec{\theta}) = (\vec{f}(\vec{x}); \vec{\theta}) - \theta_0$
- Функция потерь SVM: $\mathcal{L}(a, \vec{x}, y) = (|a - y| - \epsilon)_+$
- Ставится задача оптимизации $Q_{\text{train}}(\vec{\theta}, \theta_0)$ при $\vec{x}_i \in \mathbb{X}_{\text{train}}$
- Решение проверяется при $\vec{x}_i \in \mathbb{X}_{\text{val}}$



SVM регрессия

Исходная задача

$$\begin{cases} \frac{1}{2}\left\|\vec{\theta}\right\|^2 + C\sum\limits_{i=1}^{l}(\zeta_i + \zeta_i') \to \min \\ -\epsilon - \zeta_i' \le y_i - (\vec{\theta}, \vec{x}) - \theta_0 \le \epsilon + \zeta_i \\ \zeta_i \ge 0; \qquad \zeta_i' \ge 0. \end{cases}$$

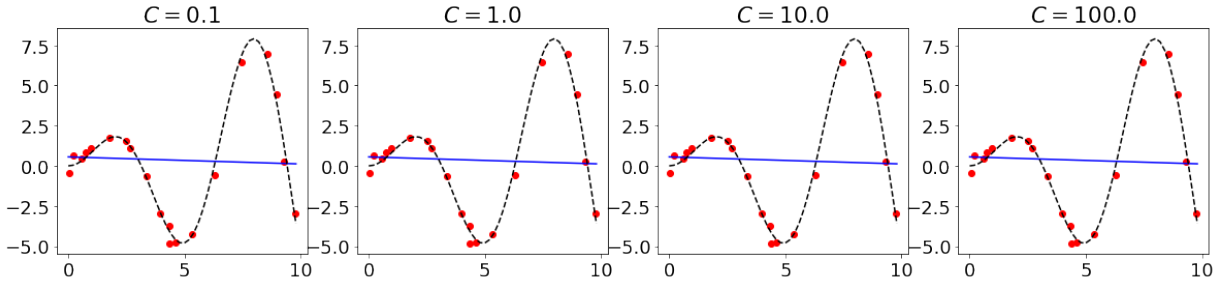
Для нее ставится и решается двойственная задача

$$\begin{cases} \frac{1}{2}\sum\limits_{i,j=1}^{L}(\mu_i - \mu_i')(\mu_j - \mu_j')y_i y_j(\vec{x}_i, \vec{x}_j) + \epsilon(\mu_i + \mu_i') - (\vec{y}, \mu_i - \mu_i') \to \min \\ \sum\limits_{i=1}^{L}(\mu_i - \mu_i') = 0; \qquad 0 \le \mu_i, \mu_i' \le C \end{cases}$$

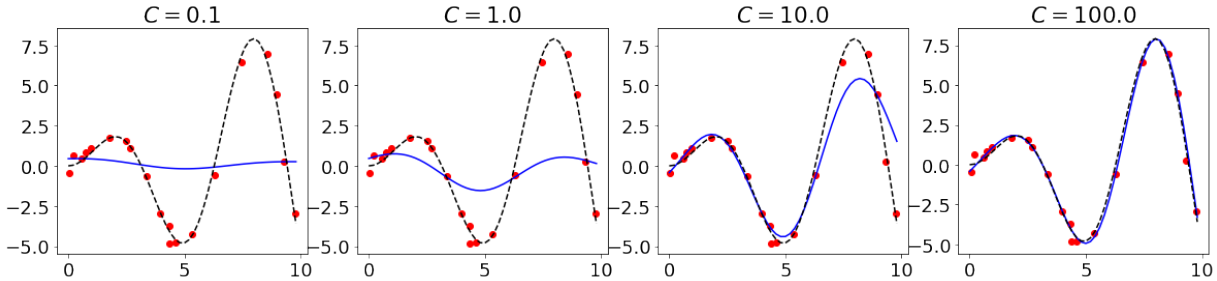
SVM регрессия

Разные типы ядер

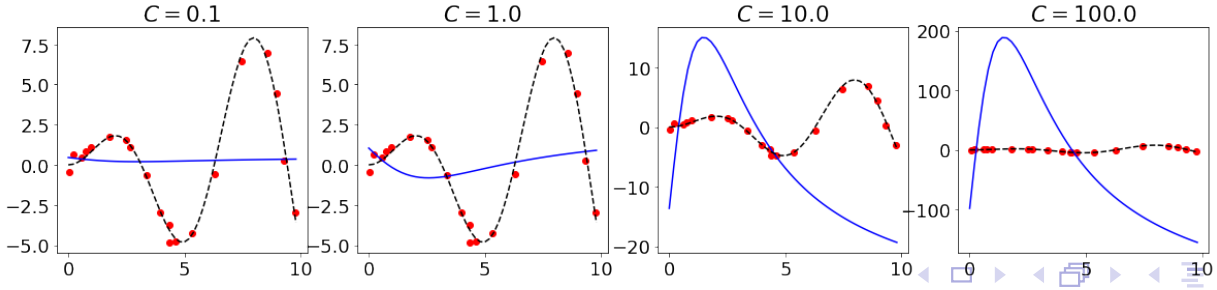
Линейное ядро



Гауссово ядро

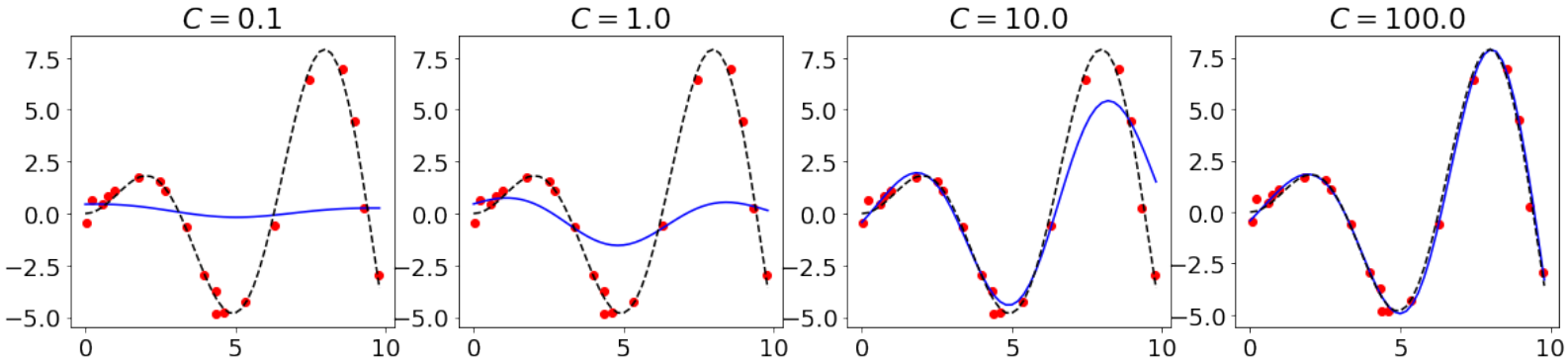


Сигмоидное ядро

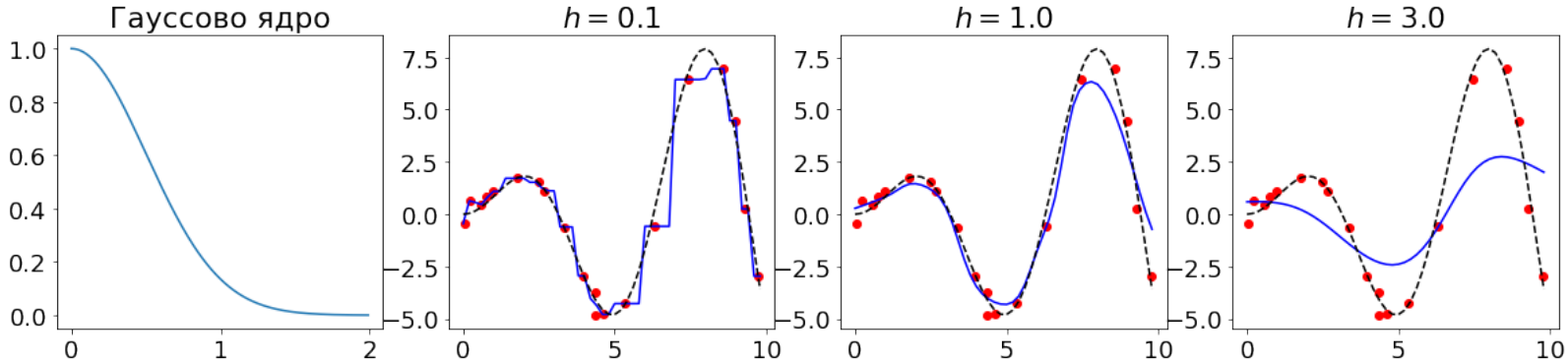


SVM регрессия и метрические методы

Гауссово ядро



Формула Надарая-Уотсона с гаусовым ядром



SVM классификация и регрессия

Jupyter notebook “SVM классификация и регрессия”:
`https://colab.research.google.com/drive/1xPxXUkcLu2OBYgOkxExBmZ7DulQjHi_S`

Лекция 10. Критерии оценки качества моделей

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Критерии качества в задачах бинарной классификации

Матрица платежей

		Истина	
		Нет	Да
		p_0	p_1
Альтернатива	1	Ложная тревога	Правильное обнаружение
	0	Правильное необнаружение	Пропуск цели

$$\text{Риск} = \text{ЦЛТ} \cdot p(\text{ЛТ}) + \text{ЦПЦ} \cdot p(\text{ПЦ})$$

Критерии качества в задачах бинарной классификации

Матрица платежей (confusion matrix)

		Predicted condition			
	Total population = P + N	**Positive (PP)**	**Negative (PN)**	Informedness, bookmaker informedness (BM) = TPR + TNR − 1	Prevalence threshold (PT) = $\frac{\sqrt{TPR \times FPR} - FPR}{TPR - FPR}$
Actual condition	**Positive (P)**	**True positive (TP),** hit	**False negative (FN),** type II error, miss, underestimation	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power = $\frac{TP}{P}$ = 1 − FNR	False negative rate (FNR), miss rate = $\frac{FN}{P}$ = 1 − TPR
	Negative (N)	**False positive (FP),** type I error, false alarm, overestimation	**True negative (TN),** correct rejection	False positive rate (FPR), probability of false alarm, fall-out = $\frac{FP}{N}$ = 1 − TNR	True negative rate (TNR), specificity (SPC), selectivity = $\frac{TN}{N}$ = 1 − FPR
	Prevalence = $\frac{P}{P+N}$	Positive predictive value (PPV), precision = $\frac{TP}{PP}$ = 1 − FDR	False omission rate (FOR) = $\frac{FN}{PN}$ = 1 − NPV	Positive likelihood ratio (LR+) = $\frac{TPR}{FPR}$	Negative likelihood ratio (LR−) = $\frac{FNR}{TNR}$
	Accuracy (ACC) = $\frac{TP + TN}{P + N}$	False discovery rate (FDR) = $\frac{FP}{PP}$ = 1 − PPV	Negative predictive value (NPV) = $\frac{TN}{PN}$ = 1 − FOR	Markedness (MK), deltaP (Δp) = PPV + NPV − 1	Diagnostic odds ratio (DOR) = $\frac{LR+}{LR-}$
	Balanced accuracy (BA) = $\frac{TPR + TNR}{2}$	F_1 score = $\frac{2PPV \times TPR}{PPV + TPR}$ = $\frac{2TP}{2TP + FP + FN}$	Fowlkes–Mallows index (FM) = $\sqrt{PPV \times TPR}$	Matthews correlation coefficient (MCC) = $\sqrt{TPR \times TNR \times PPV \times NPV} - \sqrt{FNR \times FPR \times FOR \times FDR}$	Threat score (TS), critical success index (CSI), Jaccard index = $\frac{TP}{TP + FN + FP}$

https://en.wikipedia.org/wiki/Confusion_matrix

Критерии качества в задачах бинарной классификации

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} - \text{доля верных ответов}$$

Недостатки

- не учитывается баланс классов, они входят симметрично;
- не учитываются цены ошибок.

Критерии качества в задачах бинарной классификации

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} - \text{доля верных ответов}$$

Недостатки

- не учитывается баланс классов, они входят симметрично;
- не учитываются цены ошибок.

Примеры

- Отрицательная диагностика редкого заболевания.

Критерии качества в задачах бинарной классификации

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} - \text{доля верных ответов}$$

Недостатки

- не учитывается баланс классов, они входят симметрично;
- не учитываются цены ошибок.

Примеры

- Отрицательная диагностика редкого заболевания.
- На “Титанике” выжили 19% мужчин и 75% женщин.

Критерии качества в задачах бинарной классификации

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} - \text{доля верных ответов}$$

Недостатки

- не учитывается баланс классов, они входят симметрично;
- не учитываются цены ошибок.

Примеры

- Отрицательная диагностика редкого заболевания.
- На “Титанике” выжили 19% мужчин и 75% женщин. Классификатор “выжили все женщины, и погибли все мужчины” дает $accuracy = 79\%$

Критерии качества в задачах бинарной классификации

Точность (precision)

Задача типа поиска:

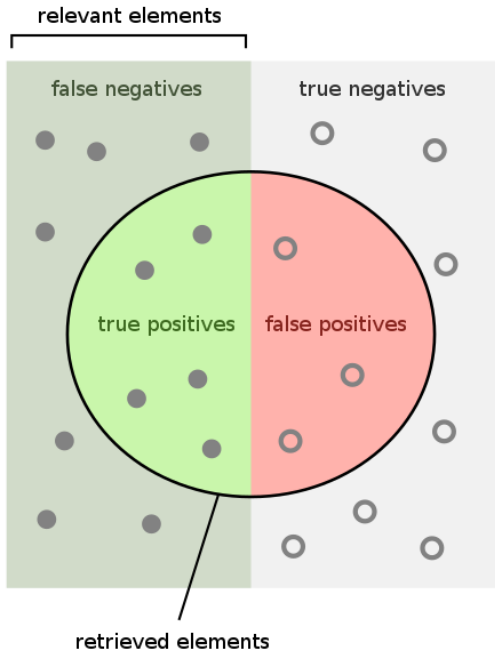
TN – заведомо большое число

Доля релевантных ответов среди выбранных

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

Недостатки

- отбирается только часть релевантных объектов



Критерии качества в задачах бинарной классификации

Полнота (recall)

Задача типа поиска:

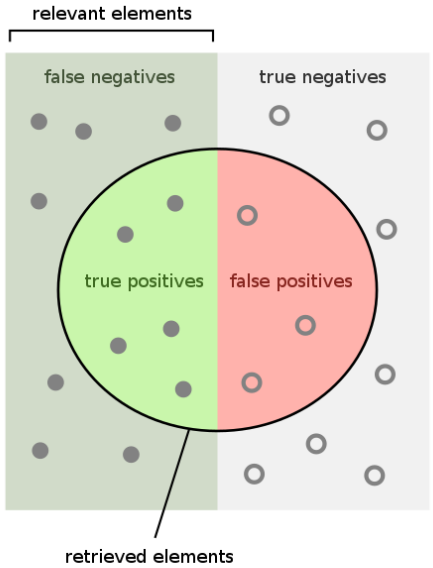
TN – заведомо большое число

Доля выбранных ответов среди релевантных

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

Недостатки

- среди отобранных объектов может быть много ошибок



Критерии качества в задачах бинарной классификации

Полнота (recall)

Задача типа поиска:

TN – заведомо большое число

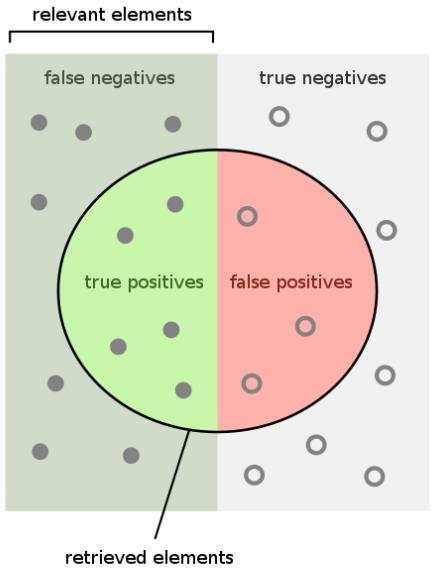
Доля выбранных ответов среди релевантных

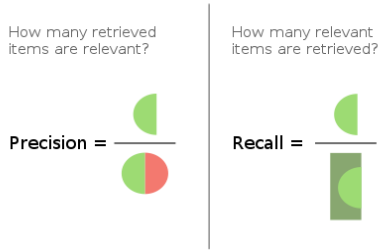
$$Recall = \frac{TP}{TP + FN}$$

Недостатки

- среди отобранных объектов может быть много ошибок

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$





Критерии качества в задачах бинарной классификации

Полнота (recall)

Задача типа поиска:

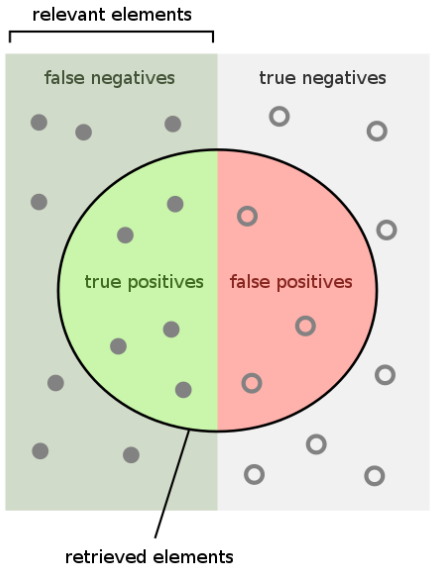
TN – заведомо большое число

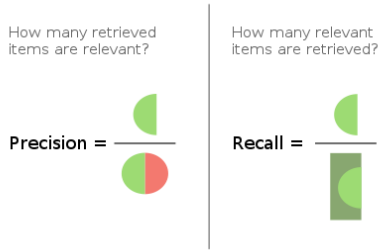
Доля выбранных ответов среди релевантных

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

Недостатки

- среди отобранных объектов может быть много ошибок





$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

$$F_\beta = \frac{(1 + \beta^2) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$

Критерии качества в задачах бинарной классификации

Чувствительность и специфичность (sensitivity and specificity)

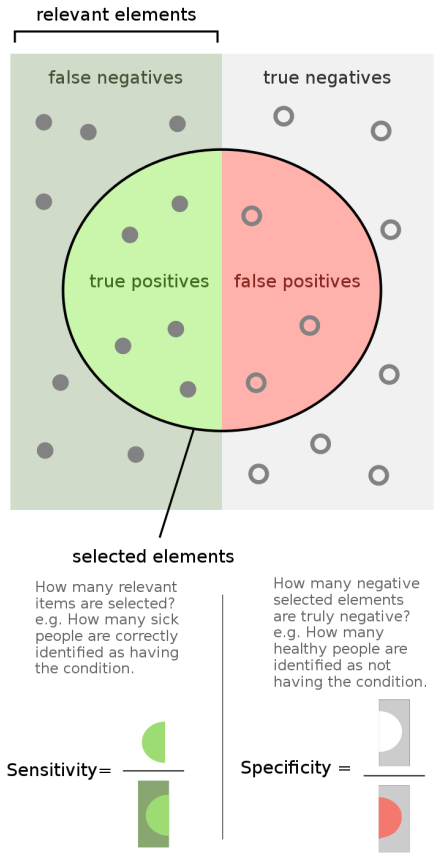
Задача с несбалансированными классами

Доля верных положительных диагнозов

$$Sensitivity = Recall = \frac{TP}{TP + FN}$$

Доля верных отрицательных диагнозов

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



Критерии качества в задачах бинарной классификации

Обобщение на мультиклассовый случай

- Усреднение 1 типа (классы сбалансированы):

$$Precision = \frac{\sum_{j=1}^{s} TP_j}{\sum_{j=1}^{s} \left(TP_j + FP_j\right)}$$

- Усреднение 2 типа (классы не сбалансированы):

$$Precision = \frac{1}{s} \sum_{j=1}^{s} \frac{TP_j}{TP_j + FP_j}$$

Критерии качества в задачах бинарной классификации

ROC-кривая (receiver operating characteristic)

- Пусть алгоритм $a(\vec{x}, \vec{\theta}) = \text{sgn}(f(\vec{x}, \vec{\theta}) - \theta_0)$

Критерии качества в задачах бинарной классификации

ROC-кривая (receiver operating characteristic)

- Пусть алгоритм $a(\vec{x}, \vec{\theta}) = \text{sgn}(f(\vec{x}, \vec{\theta}) - \theta_0)$
- Порог θ_0 изменяется в широких пределах

Критерии качества в задачах бинарной классификации

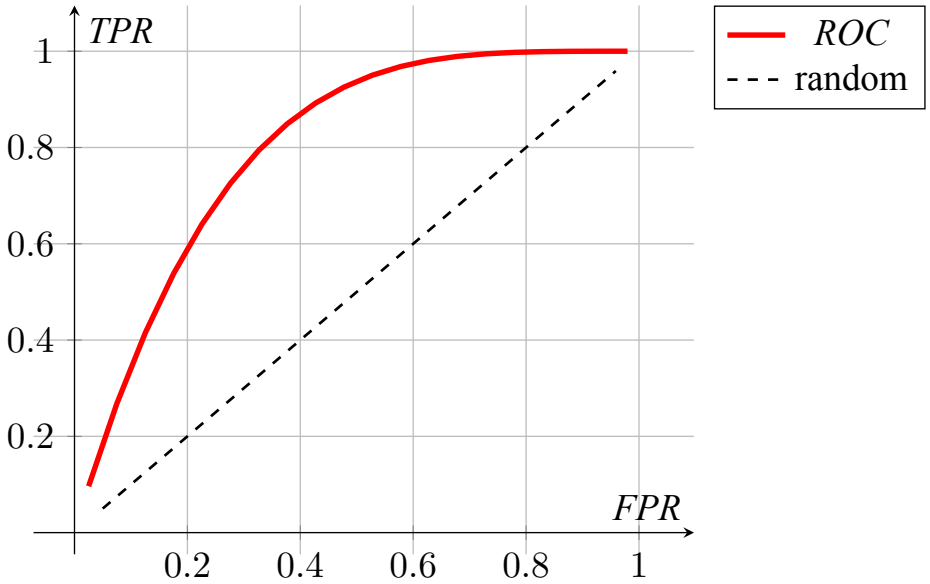
ROC-кривая (receiver operating characteristic)

- Пусть алгоритм $a(\vec{x}, \vec{\theta}) = \text{sgn}(f(\vec{x}, \vec{\theta}) - \theta_0)$
- Порог θ_0 изменяется в широких пределах
- Для каждого θ_0 определяются
  - $FPR = \frac{FP}{FP+TN} \equiv 1 - Specificity$ – доля ошибок на классе “–1”;
  - $TPR = \frac{TP}{FN+TP} \equiv Sensitivity$ – доля верных ответов на классе “+1”.

Критерии качества в задачах бинарной классификации

ROC-кривая (receiver operating characteristic)

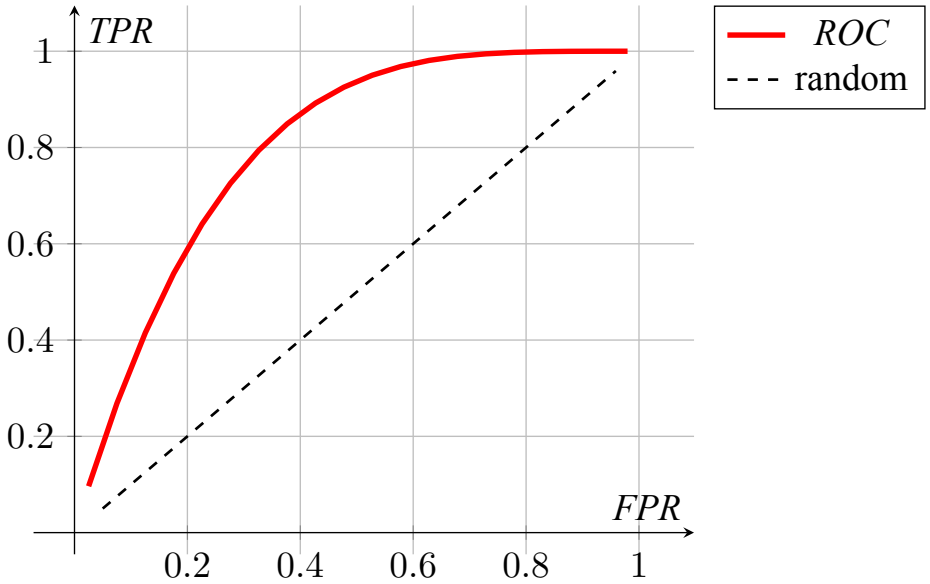
- Пусть алгоритм $a(\vec{x}, \vec{\theta}) = \text{sgn}(f(\vec{x}, \vec{\theta}) - \theta_0)$
- Порог θ_0 изменяется в широких пределах
- Для каждого θ_0 определяются
  - $FPR = \frac{FP}{FP+TN} \equiv 1 - Specificity$ – доля ошибок на классе “–1”;
  - $TPR = \frac{TP}{FN+TP} \equiv Sensitivity$ – доля верных ответов на классе “+1”.



Критерии качества в задачах бинарной классификации

ROC-кривая (receiver operating characteristic)

- Пусть алгоритм $a(\vec{x}, \vec{\theta}) = \text{sgn}(f(\vec{x}, \vec{\theta}) - \theta_0)$
- Порог θ_0 изменяется в широких пределах
- Для каждого θ_0 определяются
  - $FPR = \frac{FP}{FP+TN} \equiv 1 - Specificity$ – доля ошибок на классе “–1”;
  - $TPR = \frac{TP}{FN+TP} \equiv Sensitivity$ – доля верных ответов на классе “+1”.



- Вычисляется AUC
- ROC-кривая не зависит от числа объектов и подходит для

Критерии качества в задачах бинарной классификации

Log-loss

Модель классификации оценивает вероятность принадлежности к каждому классу.

Критерий – логарифм правдоподобия:

$$L(\vec{\theta}) = \sum_{i=1}^{L} \Big([y_i = +1] \ln f(\vec{x}, \vec{\theta}) + [y_i = -1] \ln(1 - f(\vec{x}, \vec{\theta})) \Big) \underset{\vec{\theta}}{\to} \max$$

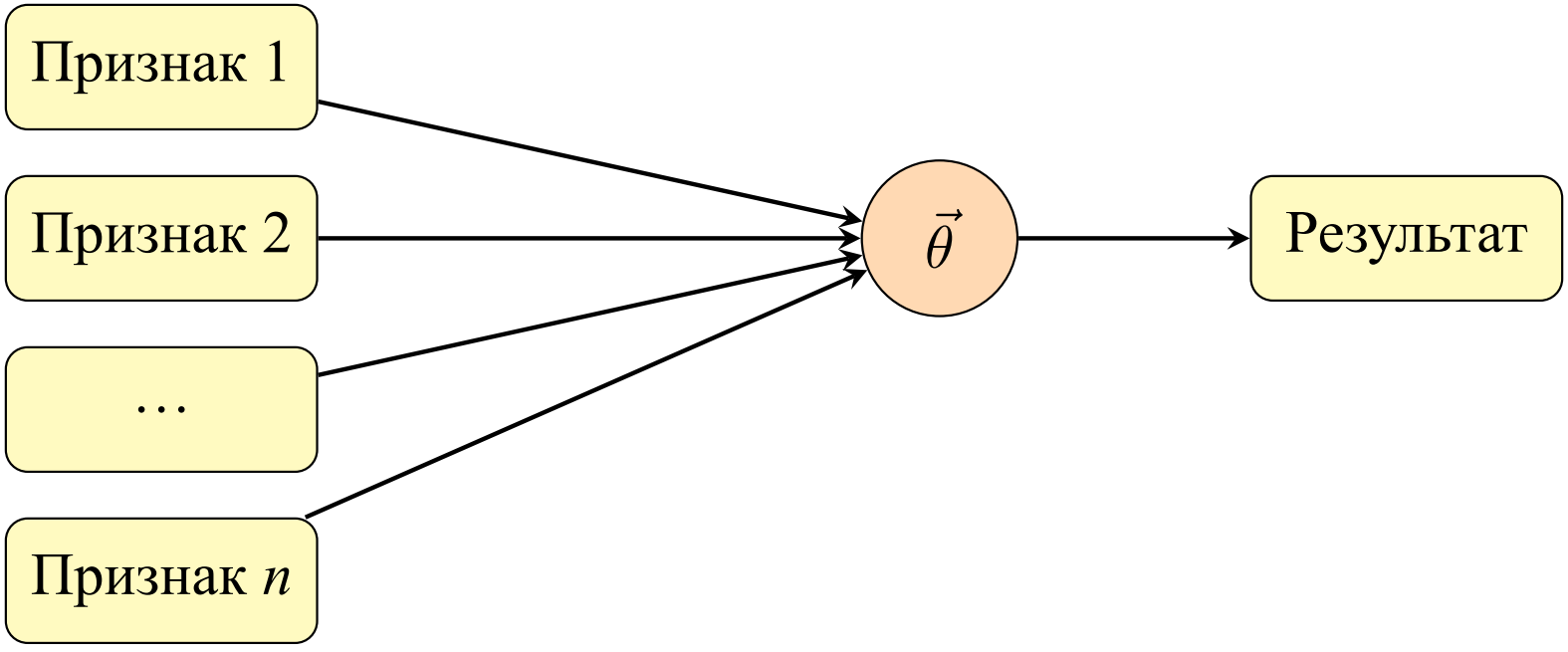
Лекция 11. Логические закономерности. Дерево принятия решений. Случайный лес

Методы машинного обучения в анализе изображений и временных рядов

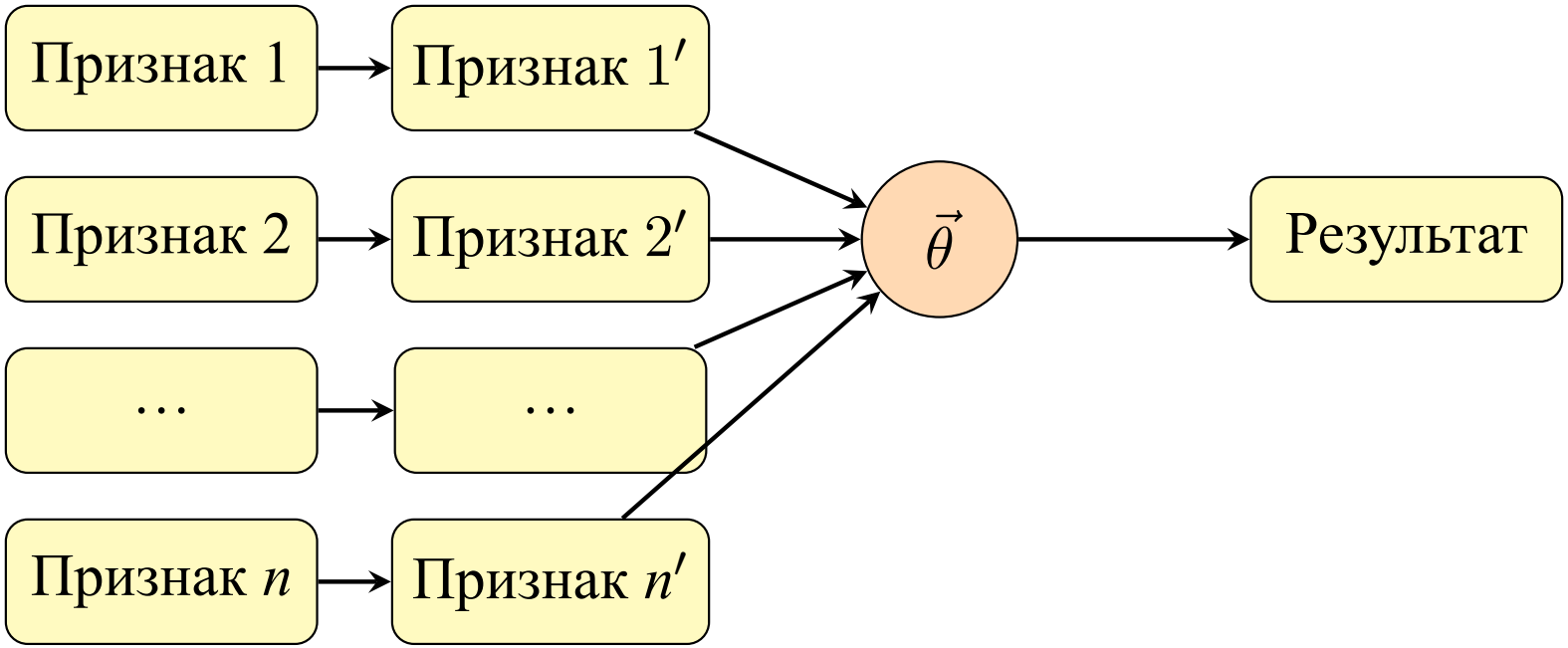
Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Задачи линейной регрессии и классификации



Задачи линейной регрессии и классификации



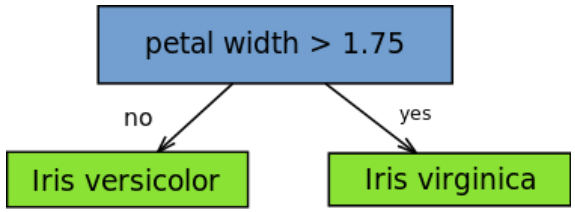
Идея: на основе признаков создать набор правил для более эффективного решения задачи; отобранные правила – новые признаки.

- Правила возвращают 1 (выделение) или 0 (отказ от классификации).
- Правила должны быть интерпретируемы.
- Правила должны быть информативны.

Генерация семейств правил

Простые правила

- Решающий пень (decision stump)
  - односторонний: $[x_j \leq a]$
  - двусторонний: $[a \leq x_j \leq b]$



- Решающий шар $[\rho(\vec{x};\vec{x}_0) \leq \theta_0]$ с применением различных функций расстояния.
- Решающая плоскость $\left[\sum_{j\in\mathbb{J}} x_j\theta_j \geq \theta_0\right]$

Генерация семейств правил

Сложные правила

- Объединение правил $\bigcap\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j \right]$
- "Синдром" $\sum\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j \right] \geq c$

Генерация семейств правил

Сложные правила

- Объединение правил $\bigcap\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right]$
- “Синдром” $\sum\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right] \geq c$
  - $c = 1 \Rightarrow \bigcup\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right]$
  - $c = |J| \Rightarrow \bigcap\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right]$

Генерация семейств правил

Сложные правила

- Объединение правил $\bigcap\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right]$
- "Синдром" $\sum\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right] \geq c$
  - $c = 1 \Rightarrow \bigcup\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right]$
  - $c = |J| \Rightarrow \bigcap\limits_{j \in \mathbb{J}} \left[a_j \leq x_j \leq b_j\right]$

Часто полагают $|\mathbb{J}|$ равным 3 для простоты подбора.

Синтез правил на основе выбранных семейств

1. Генерируется исходный набор правил.
2. Правила немного модифицируются (параметры a, b, c, вектор $\vec{x}_0$, наборы признаков $\mathbb{J}$).
3. Новый и исходный наборы правил объединяются, и при необходимости удаляются повторы.
4. Выбор наиболее информативных правил для следующего шага.

Критерии информативности правил

- $P(R)$ – число правильно отобранных правилом объектов
- $N(R)$ – число неверно отобранных правилом объектов

Критерии информативности правил

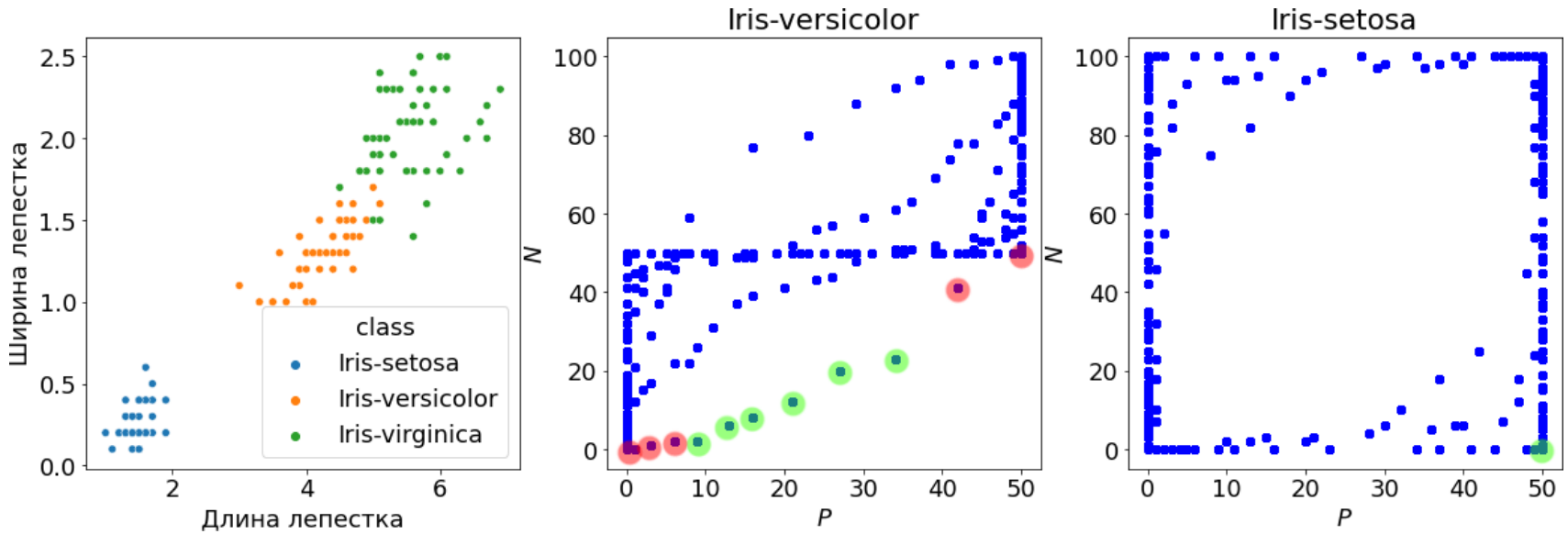
- $P(R)$ – число правильно отобранных правилом объектов
- $N(R)$ – число неверно отобранных правилом объектов

Хотелось бы, чтобы $P(R) \to \max$; $N(R) \to \min$

Критерии информативности правил

Парето-расслоение

Парето-слой – набор неулучшаемых правил



Критерии информативности правил

Проверка гипотез

Гипотеза H_0: правило $R(\vec{x})$ и результат $y(\vec{x})$ независимы.

Критерии информативности правил

Проверка гипотез

Гипотеза H_0: правило $R(\vec{x})$ и результат $y(\vec{x})$ независимы.

Точный тест Фишера:

$$\text{Информативность } (p,n) = -\frac{1}{L}\log_2 \frac{C_P^p C_N^n}{C_{P+N}^{p+n}} \to \max$$

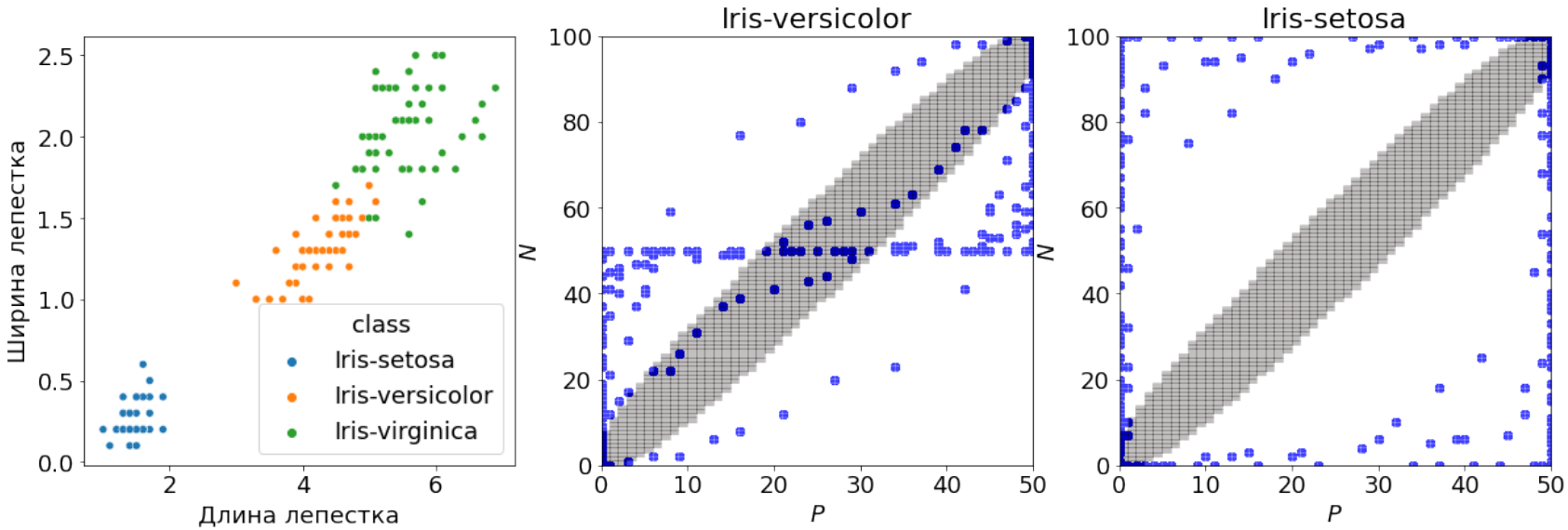
Критерии информативности правил

Проверка гипотез

Гипотеза H_0: правило $R(\vec{x})$ и результат $y(\vec{x})$ независимы.

Точный тест Фишера:

$$\text{Информативность}\ (p,n) = -\frac{1}{L}\log_2 \frac{C_P^p C_N^n}{C_{P+N}^{p+n}} \to \max$$

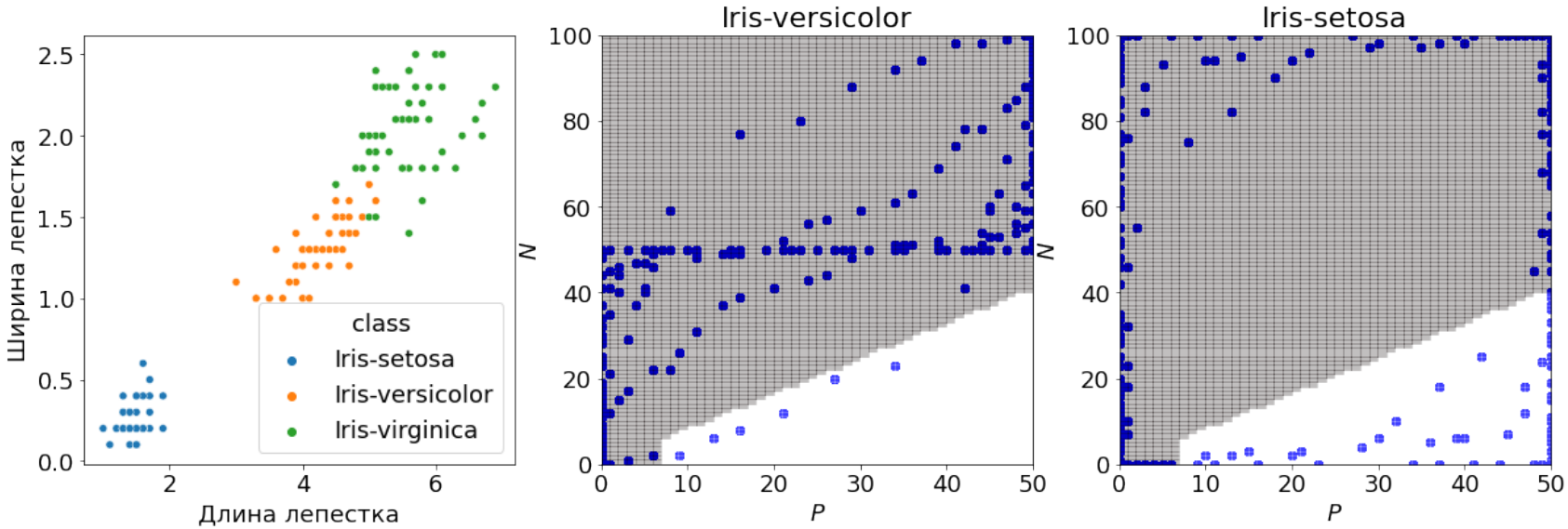


Серая область – статистически незначимые результаты

Критерии информативности правил

Условие $\epsilon - \delta$

- $\dfrac{n(R)}{p(R)+n(R)} \leq \epsilon$ – правило делает мало ошибок;
- $\dfrac{p(R)}{L} \geq \delta$ – правило отбирает много правильных объектов.



$$\epsilon = 0.45; \quad \delta = 0.05$$

Серая область – значения, где условие $\epsilon - \delta$ не выполнено

Критерии информативности правил

Информационные критерии

- Правило упорядочивает элементы выборки, выделяя один класс.

Критерии информативности правил

Информационные критерии

- Правило упорядочивает элементы выборки, выделяя один класс.
- Мера беспорядка – энтропия $H(\vec{p}) = -\sum_{j=1}^{s} p_j \log_s p_j$,

  p_j – вероятность j-го класса, s – число классов.

Критерии информативности правил

Информационные критерии

- Правило упорядочивает элементы выборки, выделяя один класс.
- Мера беспорядка – энтропия $H(\vec{p}) = -\sum_{j=1}^{s} p_j \log_s p_j$,

  p_j – вероятность j-го класса, s – число классов.
  - $p_j = \frac{1}{s} \Rightarrow H(\vec{p}) = 1$ – неупорядоченная выборка
  - $p_j = \delta(i,j) \Rightarrow H(\vec{p}) = 0$ – полностью упорядоченная выборка

Критерии информативности правил

Информационные критерии

- Правило упорядочивает элементы выборки, выделяя один класс.
- Мера беспорядка – энтропия $H(\vec{p}) = -\sum_{j=1}^{s} p_j \log_s p_j$,

  p_j – вероятность j-го класса, s – число классов.
  - $p_j = \frac{1}{s} \Rightarrow H(\vec{p}) = 1$ – неупорядоченная выборка
  - $p_j = \delta(i,j) \Rightarrow H(\vec{p}) = 0$ – полностью упорядоченная выборка
- Неопределенность до применения правила должна быть больше

Критерии информативности правил

Информационные критерии

Преобразование энтропии при разделении выборки $\mathcal{X}$ на части $\mathcal{X}_k$

$$H(\vec{p}) = -\sum_{j=1}^{s} p_j \log_s p_j = -\frac{1}{L}\sum_{j=1}^{s} N_j \log_s p_j =$$

$$= -\frac{1}{L}\sum_{i=1}^{L} [y(\vec{x}_i) = j] \log_s p_j = -\frac{1}{L}\sum_k \sum_{\vec{x}\in\mathcal{X}_k} [y(\vec{x}) = j] \log_s p_j$$

$$/p_j = p_{jk}/ \longrightarrow \sum_k \frac{L_k}{L}\left(-\frac{1}{L_k}\sum_{\vec{x}\in\mathcal{X}_k} [y(\vec{x}) = j] \log_s p_{jk}\right) = \sum_k \frac{L_k}{L} H(\vec{p}_k)$$

Критерии информативности правил

Информационные критерии

Преобразование энтропии при разделении выборки $\mathcal{X}$ на части $\mathcal{X}_k$

$$H(\vec{p}) = -\sum_{j=1}^{s} p_j \log_s p_j = -\frac{1}{L}\sum_{j=1}^{s} N_j \log_s p_j =$$

$$= -\frac{1}{L}\sum_{i=1}^{L} [y(\vec{x}_i) = j] \log_s p_j = -\frac{1}{L}\sum_{k}\sum_{\vec{x}\in\mathcal{X}_k} [y(\vec{x}) = j] \log_s p_j$$

$$/p_j = p_{jk}/ \longrightarrow \sum_k \frac{L_k}{L}\left(-\frac{1}{L_k}\sum_{\vec{x}\in\mathcal{X}_k} [y(\vec{x}) = j] \log_s p_{jk}\right) = \sum_k \frac{L_k}{L} H(\vec{p}_k)$$

$$\text{Gain} = H(\vec{p}) - \sum_k \frac{L_k}{L} H(\vec{p}_k)$$

Критерии информативности правил

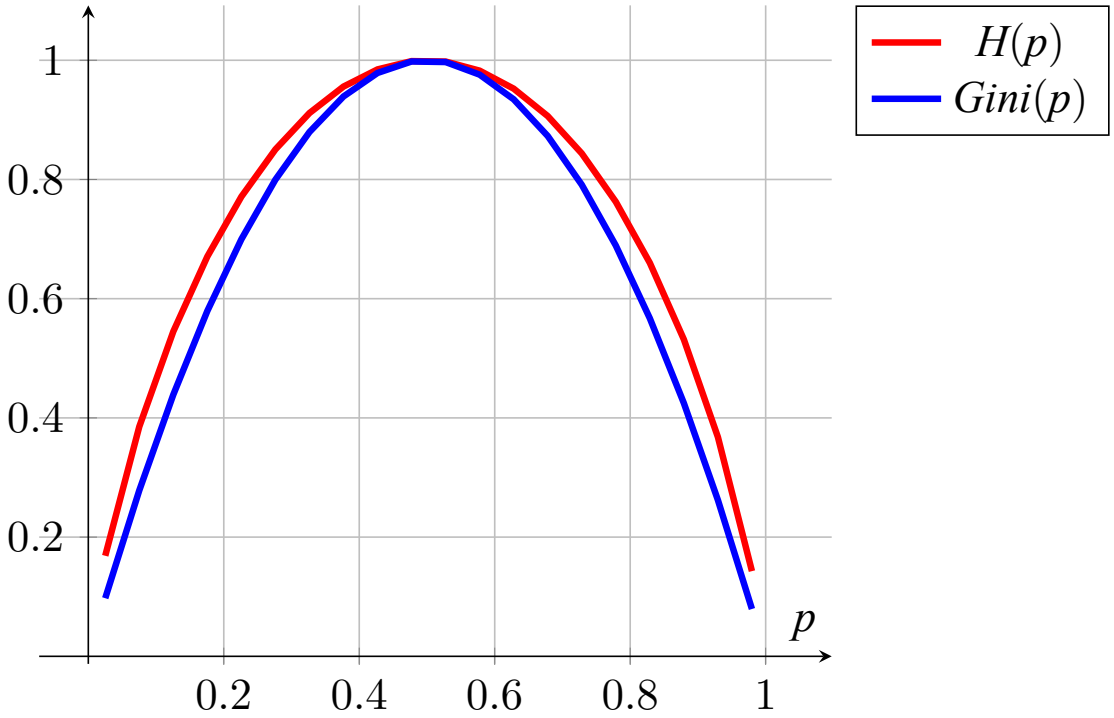
Информационные критерии. Бинарная классификация

- До разделения
  - в выборке было pL объектов класса “+1” и $(1-p)L$ – класса “-1”;
  - энтропия $H_0(p) = -p\log_2 p - (1-p)\log_2(1-p)$.
- После разделения
  - выделено P объектов класса “+1” и N – класса “-1”;
  - не выделено $pL - P$ объектов класса “+1” и $(1-p)L - N$ – класса “-1”.
  - энтропия $H(P,N) = \dfrac{P}{L}\log_2\dfrac{P}{P+N} + \dfrac{L-P-N}{L}\log_2\dfrac{(1-p)L-N}{L-P-N}$.
- Выигрыш $Gain = H(P,N) - H_0(p)$

Критерии информативности правил

Информационные критерии. Энтропия и неопределенность Джини

- Энтропия: $H(p) = -p \log_2 p - (1-p) \log_2 (1-p)$.
- Неопределенность Джини: $Gini = 4p(1-p)$



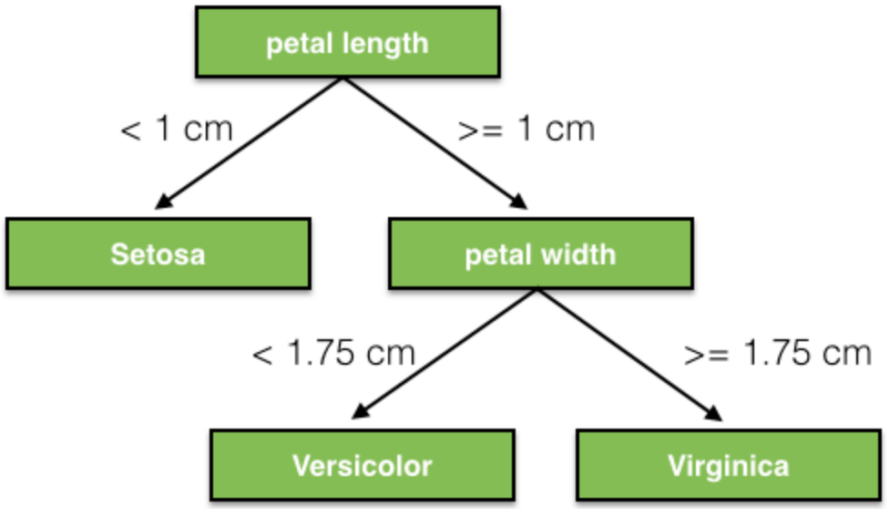
Критерии информативности правил

Сведение к однопараметрическому описанию

- $Precision = \dfrac{p(R)}{p(R) + n(R)}$
- $Accuracy = p(R) - n(R)$
- $Relative\ Accuracy = p(R)/P - n(R)/N$

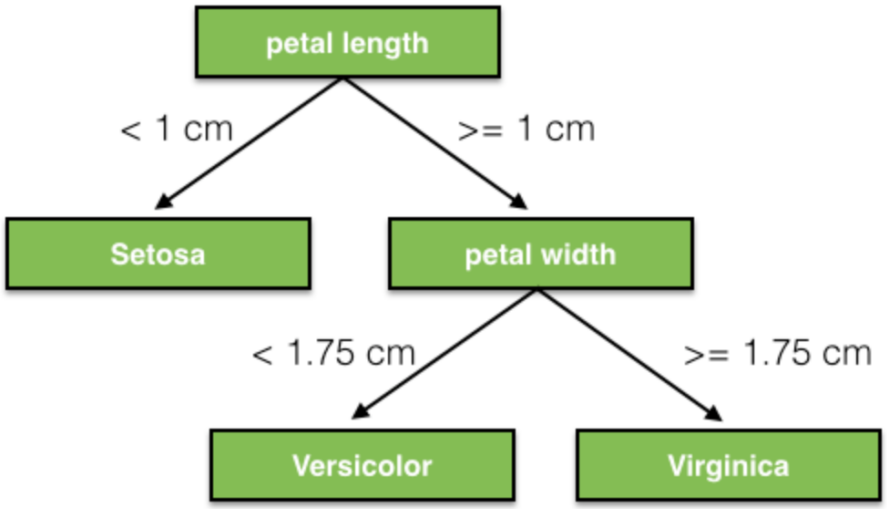
- Энтропийный критерий
- Критерий Джини
- $Boosting = \sqrt{p(R)} - \sqrt{n(R)}$

Решающие деревья



- Внутренняя вершина – правило перехода в одну из дочерних вершин. Если их всегда 2, то дерево бинарное.
- Листовая вершина содержит значение целевого признака.

Решающие деревья



- Внутренняя вершина – правило перехода в одну из дочерних вершин. Если их всегда 2, то дерево бинарное.
- Листовая вершина содержит значение целевого признака.
- Дерево – покрывающий набор конъюнкций:
  - Iris-setosa: $(PL < 1)$
  - Iris-versicolor: $(PL \geq 1) \cap (PW < 1.75)$
  - Iris-virginica: $(PL \geq 1) \cap (PW \geq 1.75)$

Решающие деревья

Алгоритм обучения дерева

1. Пусть имеется выборка объектов $\mathcal{X}$
2. Задается порог ветвления α
3. Выбирается критерий $f(r, \mathcal{X}_v)$ оценки правила r, показывающий, насколько хорошо оно делит выбоку $\mathcal{X}_v$
4. Построение дерева ведется рекурсивно, начиная с корня:

$$\text{Root}, \vec{p} = \text{Tree}(\mathcal{X})$$

`https://en.wikipedia.org/wiki/ID3_algorithm`

Решающие деревья

Алгоритм обучения дерева

Функция $v, \vec{p} = \text{Tree}(\mathcal{X}')$

1. Выбор из всех правил r наилучшего: $r_v = \arg\max_r f(r, \mathcal{X}')$
2. Если $f(r, \mathcal{X}') < \alpha$, то
   - создается листовая вершина v с целевым признаком, равным моде $\mathcal{X}$;
   - рассчитывается вектор вероятностей классов $p_j = \frac{1}{L'} \sum_{\vec{x} \in \mathcal{X}'} [y(\vec{x}) = y_j]$.
3. Если $f(r, \mathcal{X}_v) \geq \alpha$, то
   - $\mathcal{X}'$ разбивается на подвыборки $\mathcal{X}'_k$
   - для каждой из $\mathcal{X}'_k$ вычисляются $v_k, \vec{q}_k = \text{Tree}(\mathcal{X}'_k)$;
   - создается внутренняя вершина v с потомками v_k;
   - рассчитывается вектор вероятностей классов $\vec{p} = \sum_k \frac{L'_k}{L'} \vec{q}_k$
4. Возвращается вершина v и вектор $\vec{p}$

Решающие деревья

Пропущенные значения

- **При обучении**
  объекты с пропущенным значением признака игнорируются
- **При классификации**
  объектам с пропущенным значением признака в вершине v приписывается наиболее вероятный класс:

$$a(\vec{x}) = \arg\max_j p_{v,j}$$

Решающие деревья

Дерево классификации и регрессии (Classification And Regression Tree)

- Критерий $f(r, \mathcal{X}) = \frac{1}{L} \sum (y - y_j(\vec{x}))^2$
- Значения в листе усредняются: $y = \frac{1}{L} \sum y_i$

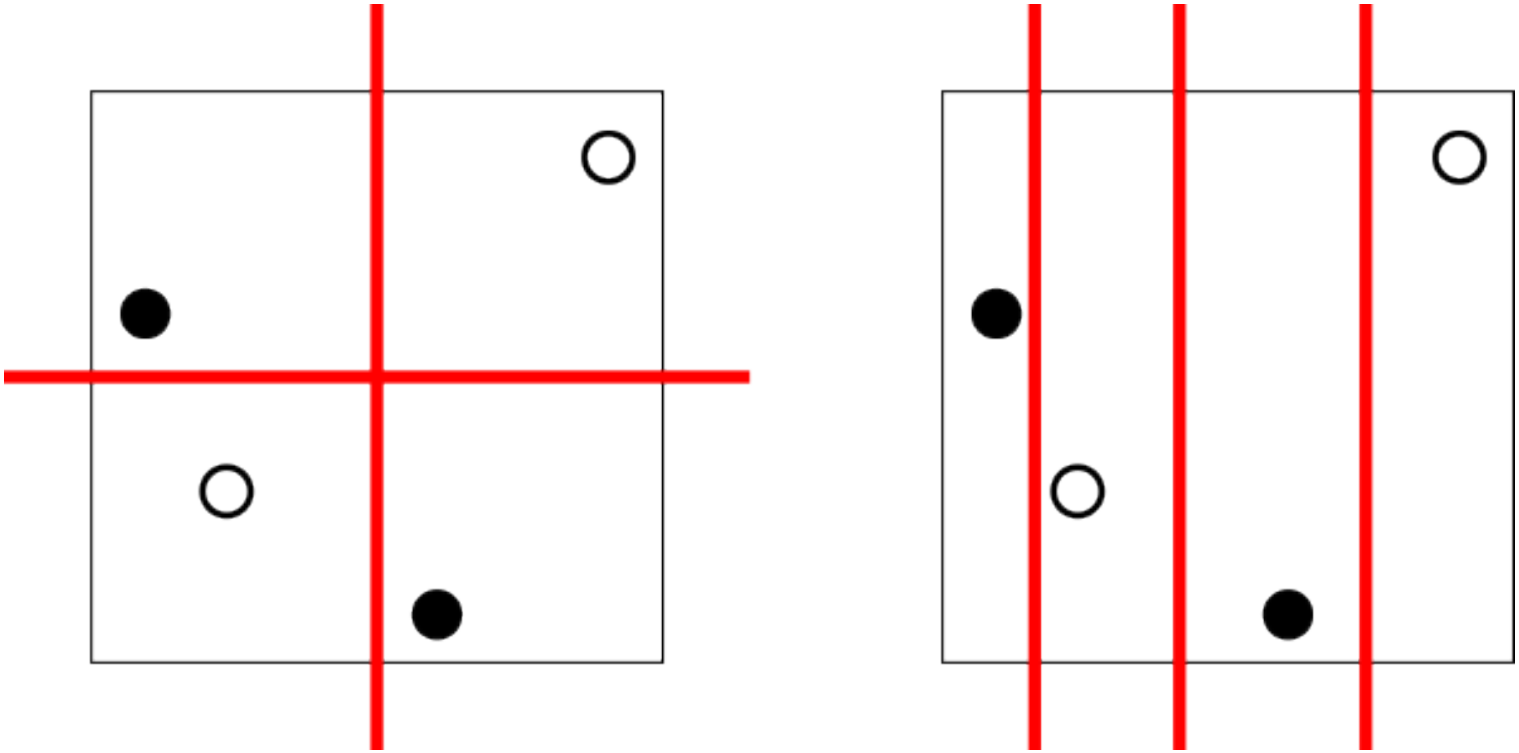
Решающие деревья

Дерево классификации и регрессии (Classification And Regression Tree)

- Критерий $f(r, \mathcal{X}) = \frac{1}{L}\sum(y - y_j(\vec{x}))^2$
- Значения в листе усредняются: $y = \frac{1}{L}\sum y_i$
- Итог – кусочно-постоянная функция

Решающие деревья

Проблемы



Решающие деревья

Проблемы

- Сильное переобучение из-за жадного алгоритма ветвления.
- Мало объектов в листьях, низкая статистическая значимость.
- Высокая неустойчивость решения.

Решающие деревья

Регуляризация: усечение дерева (pruning)

С выборкой $\mathcal{X}_{\text{val}}$ (объемом не менее $L/2$)
для каждой внутренней вершины v:

- Если объекты “не дошли” до v, заменить ее на лист $\arg\max_j p_{v,j}$

Решающие деревья

Регуляризация: усечение дерева (pruning)

С выборкой $\mathcal{X}_{\text{val}}$ (объемом не менее $L/2$)
для каждой внутренней вершины v:

- Если объекты “не дошли” до v, заменить ее на лист $\arg\max\limits_j p_{v,j}$

Вычисляются ошибки на $\mathcal{X}_{\text{val}}$ и выбирается наилучший вариант:

- Структуры вершины v сохраняется.
- Вершина v заменяется на лист с классом $\arg\max\limits_j p_{v,j}$.
- Вершина v заменяется на одно из своих поддеревьев.

Решающие деревья

Регуляризация: усечение дерева (pruning)

С выборкой $\mathcal{X}_{\text{val}}$ (объемом не менее $L/2$)
для каждой внутренней вершины v:

- Если объекты “не дошли” до v, заменить ее на лист $\arg\max_j p_{v,j}$

Вычисляются ошибки на $\mathcal{X}_{\text{val}}$ и выбирается наилучший вариант:

- Структуры вершины v сохраняется.
- Вершина v заменяется на лист с классом $\arg\max_j p_{v,j}$.
- Вершина v заменяется на одно из своих поддеревьев.

Существует множество способов перебора вершин, от которых существенно зависит результат.

Решающие деревья

Рандомный лес (random forest)

- Для классификации $a_{\text{rf}}(\vec{x}) = \text{sgn}\,\frac{1}{T}\sum_t a_{\text{t}}(\vec{x})$
- Для регрессии $a_{\text{rf}}(\vec{x}) = \frac{1}{T}\sum_t a_{\text{t}}(\vec{x})$

Рандомный лес. Плюсы и минусы

Плюсы:

- Высокая точность предсказаний (часто лучше, чем у линейных методов).
- Устойчивость к выбросам.
- Нет чувствительности к масштабированию параметров.
- Хорошо работает практически без подбора параметров.
- Хорошо работает на выборках с большим числом объектов и признаков, легко распараллеливается.
- “Встроенная” валидация и оценка важности признаков.

Рандомный лес. Плюсы и минусы

Плюсы:

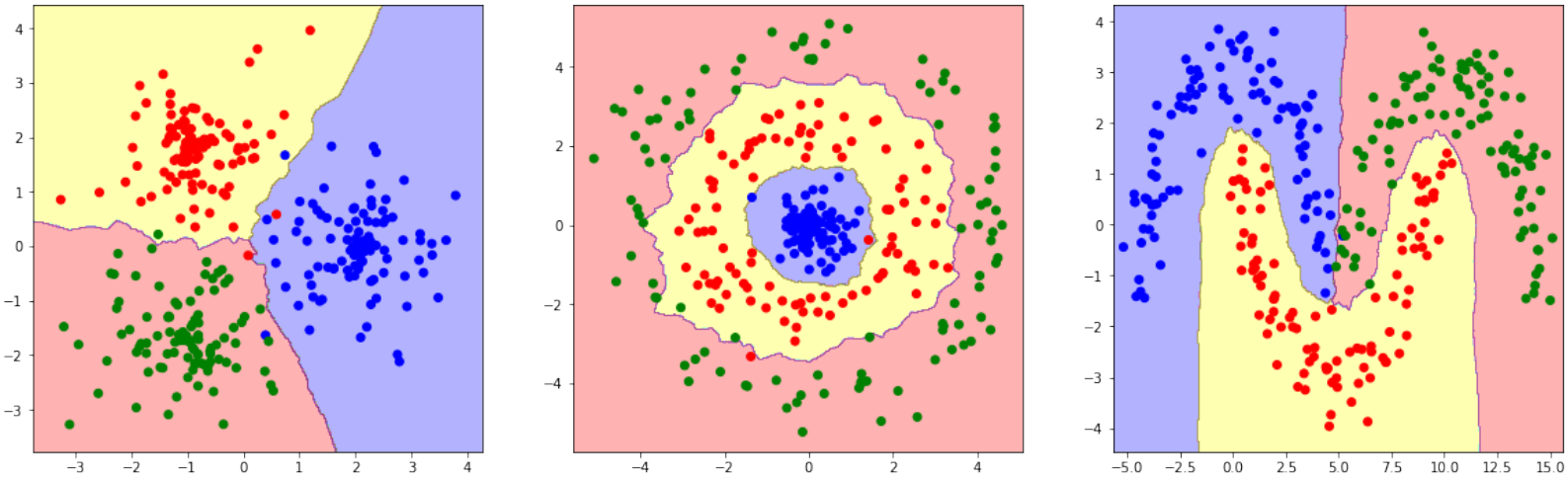
- Высокая точность предсказаний (часто лучше, чем у линейных методов).
- Устойчивость к выбросам.
- Нет чувствительности к масштабированию параметров.
- Хорошо работает практически без подбора параметров.
- Хорошо работает на выборках с большим числом объектов и признаков, легко распараллеливается.
- “Встроенная” валидация и оценка важности признаков.

Минусы:

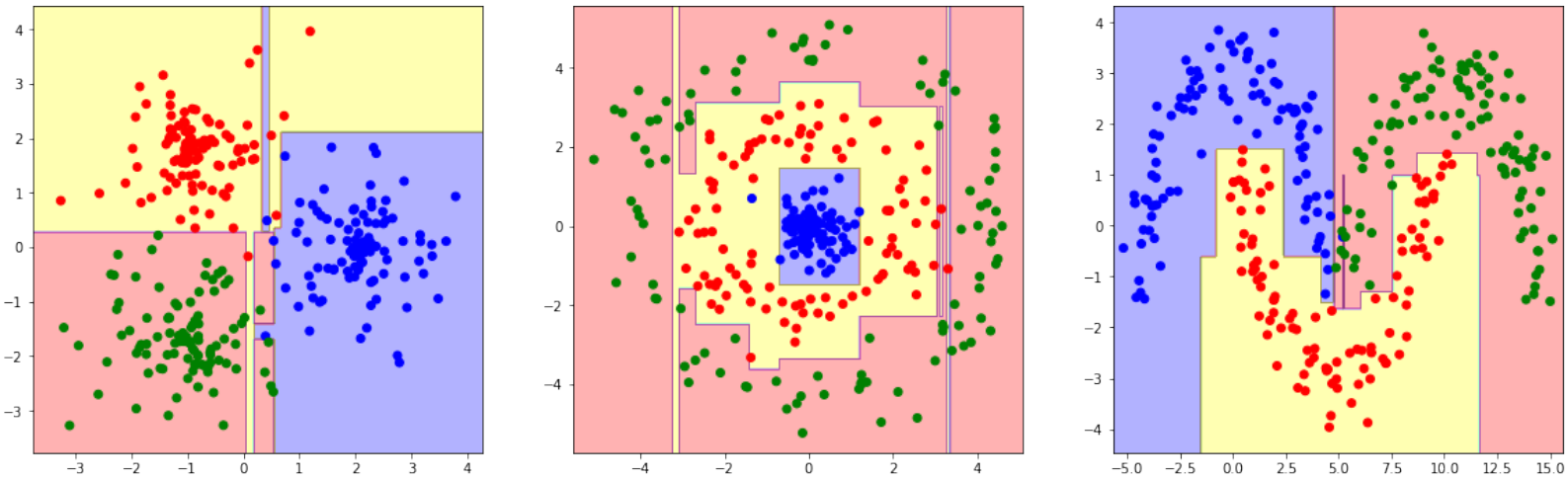
- Результат сложно интерпретировать.
- Плохо работает с разреженными данными (хуже, чем линейные методы).
- Нет возможности экстраполяции.
- Склонность к переобучению на зашумленных данных.

Сравнение решающего дерева и KNN

Решение с помощью KNN ($k = 10$)

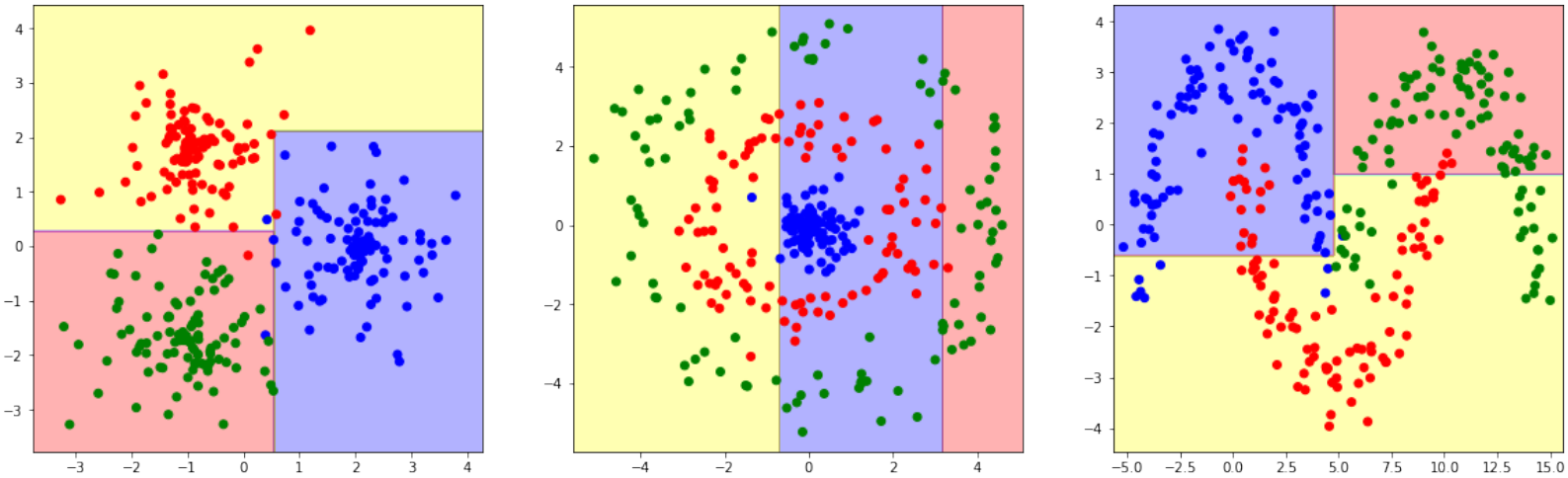


Решение с помощью решающего дерева

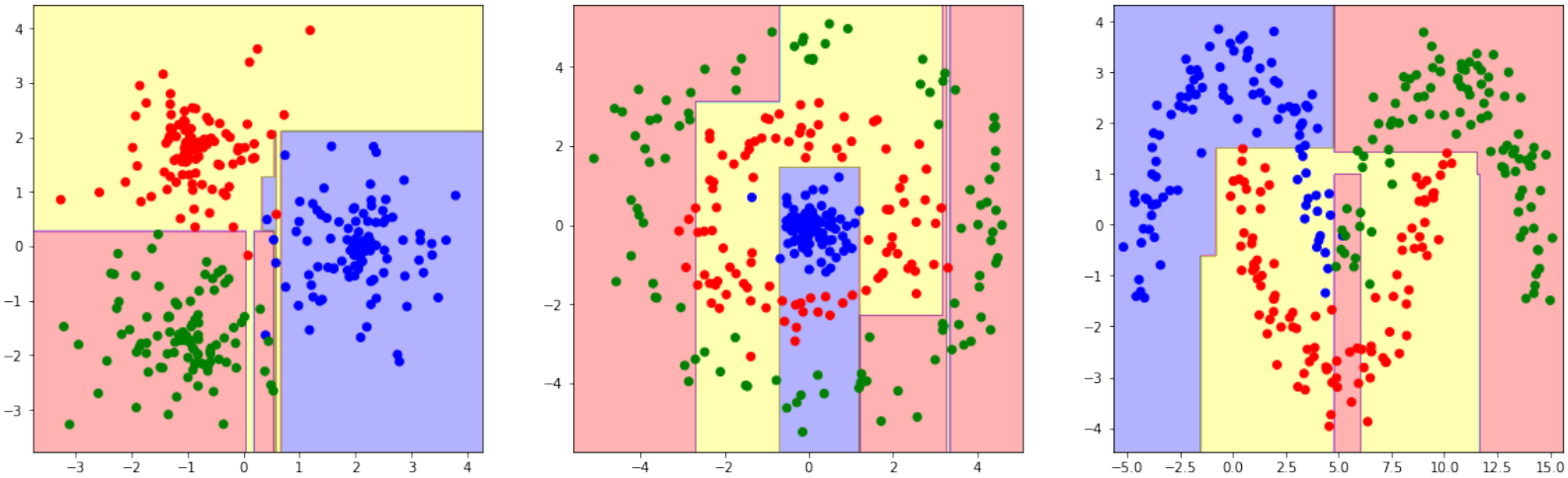


Ограничение на глубину дерева

Глубина = 2

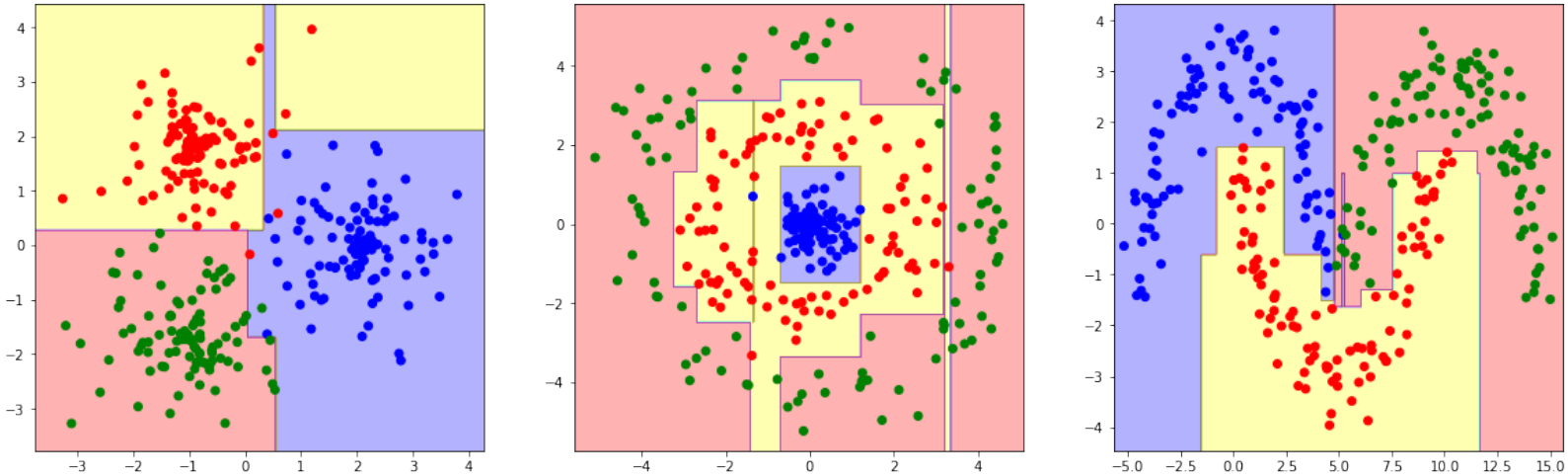


Глубина = 4

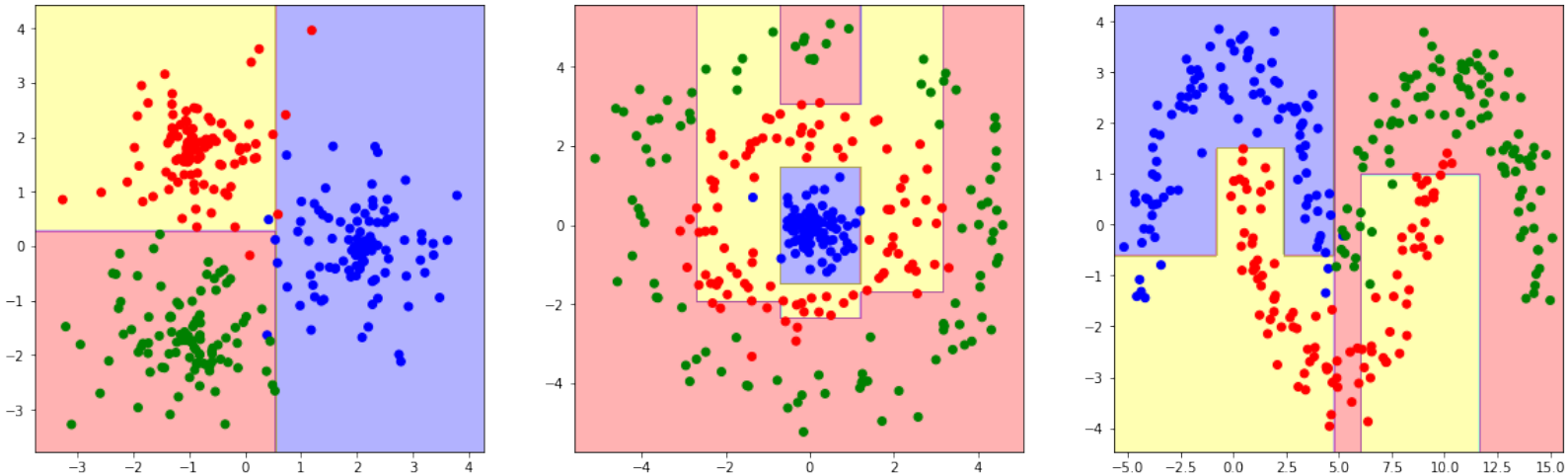


Ограничение на число элементов в листе

Число элементов в листе ≥ 2

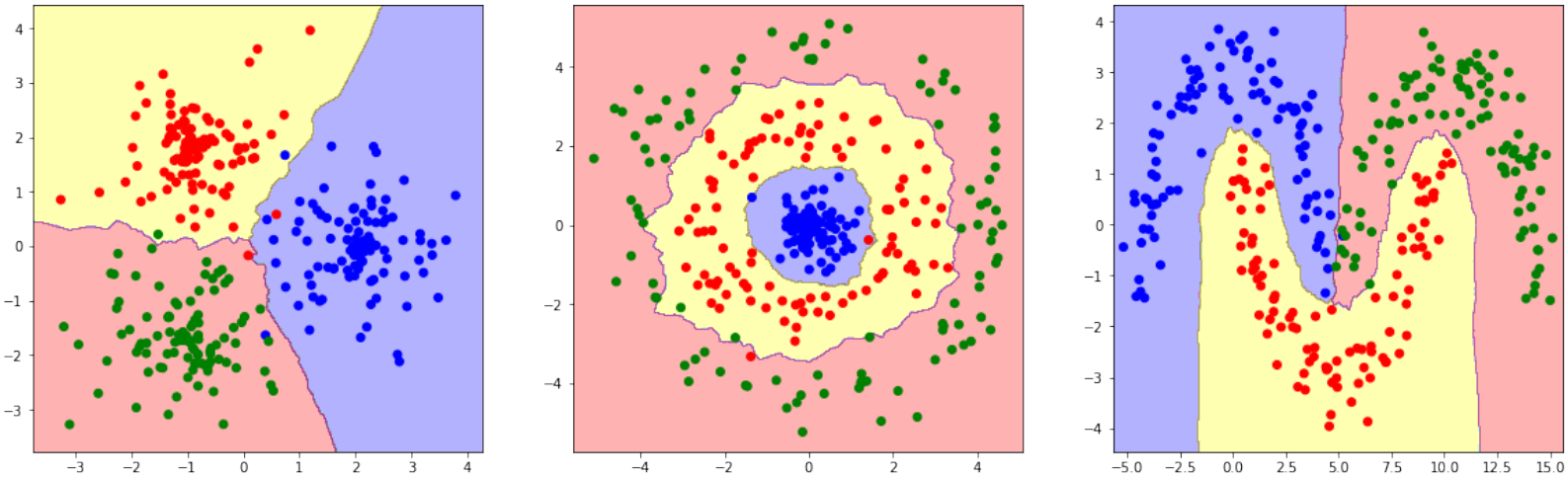


Число элементов в листе ≥ 10

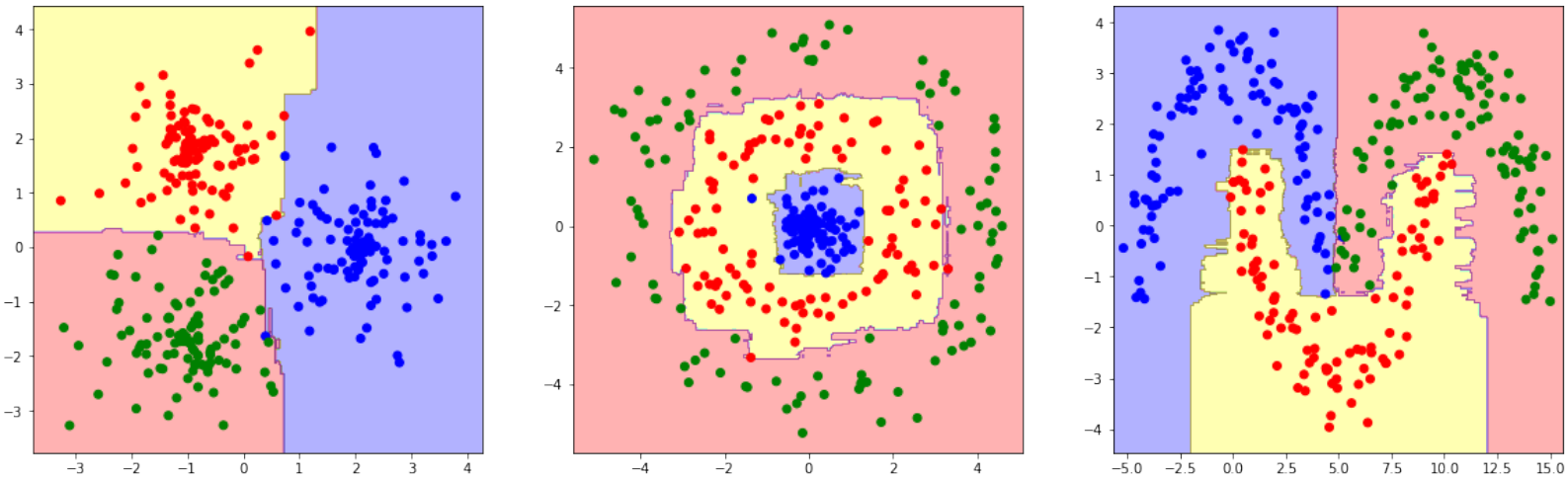


Сравнение рандомного леса и KNN

Решение с помощью KNN ($k = 10$)

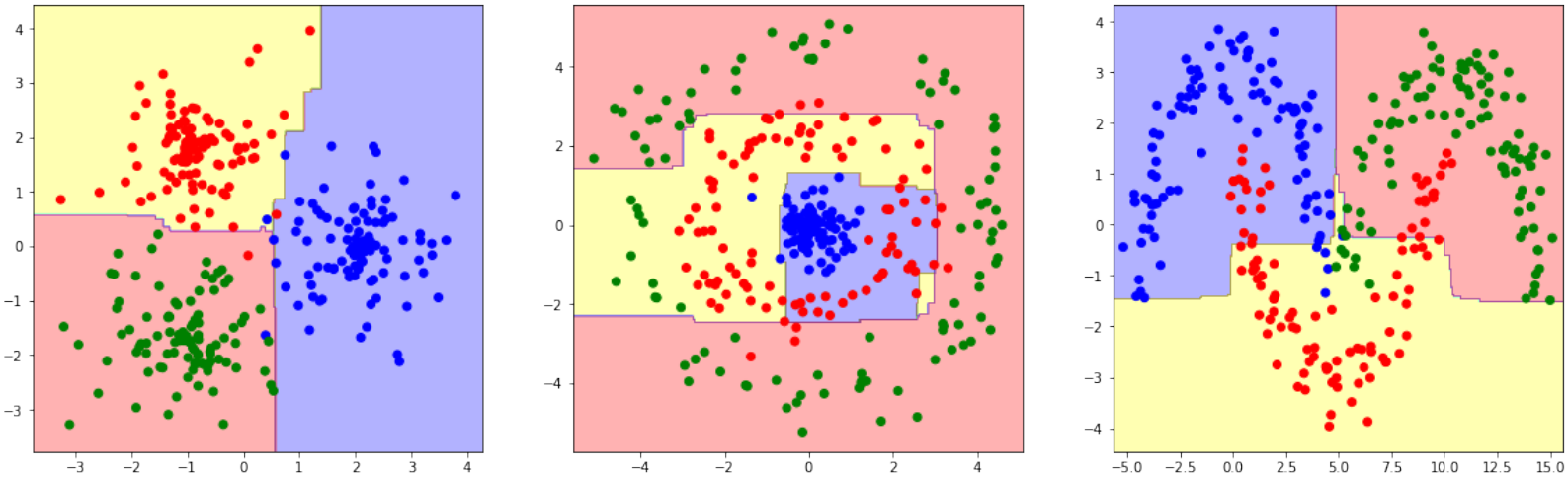


Решение с помощью рандомного леса (100 деревьев)

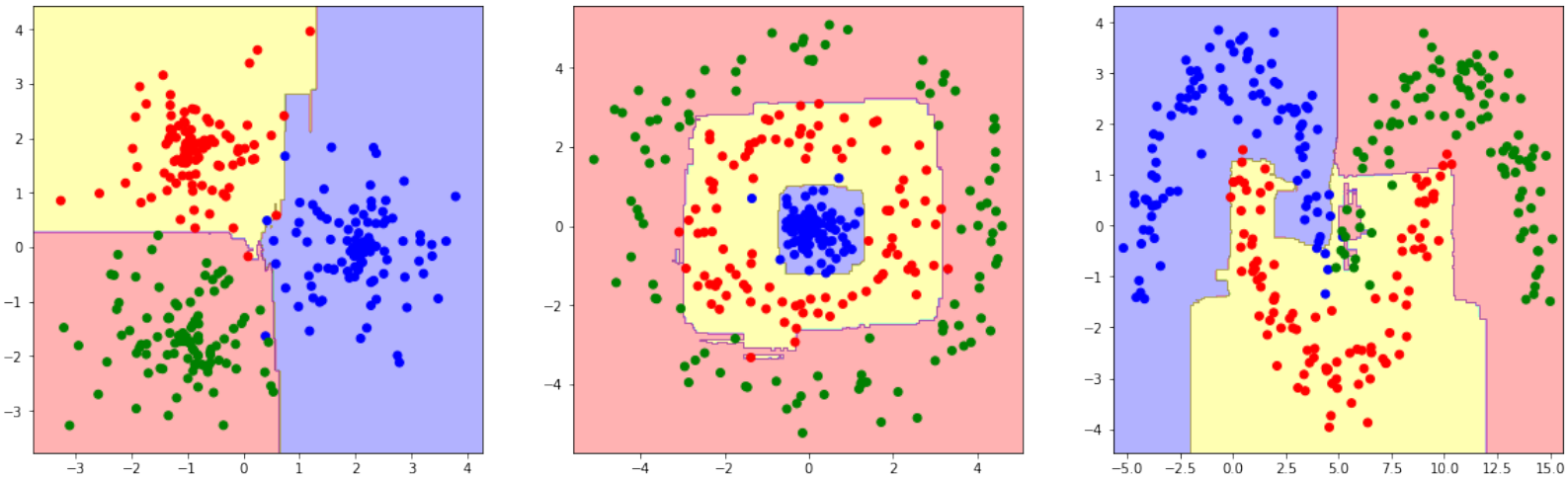


Ограничение на глубину дерева

Глубина деревьев в лесе = 2

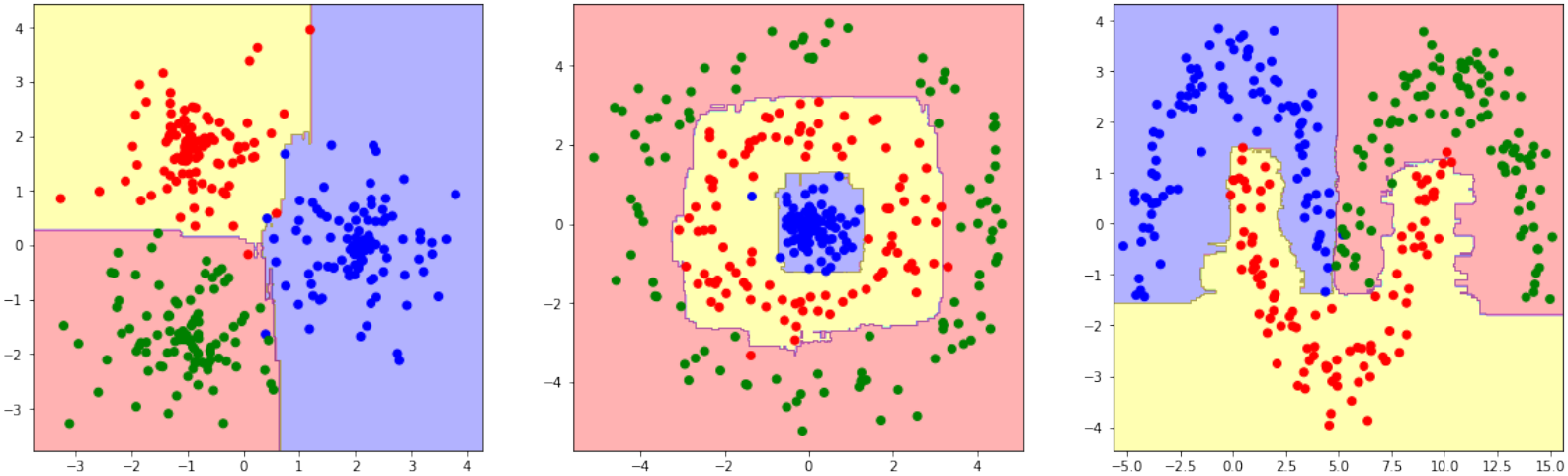


Глубина деревьев в лесе = 4

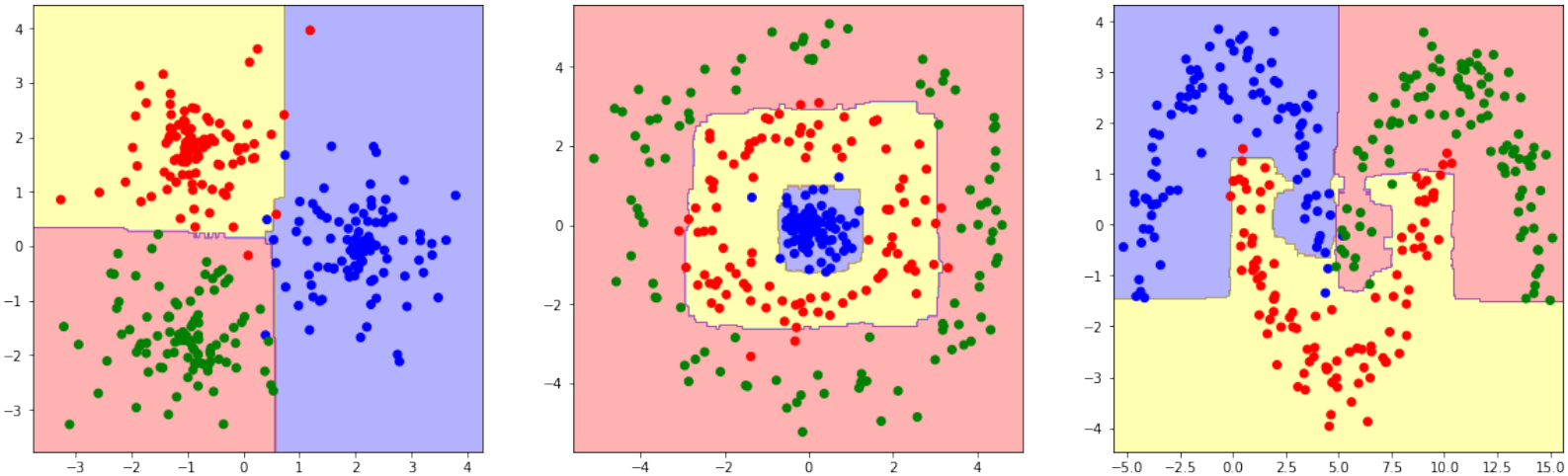


Ограничение на число элементов в листе

Число элементов в листе ≥ 2

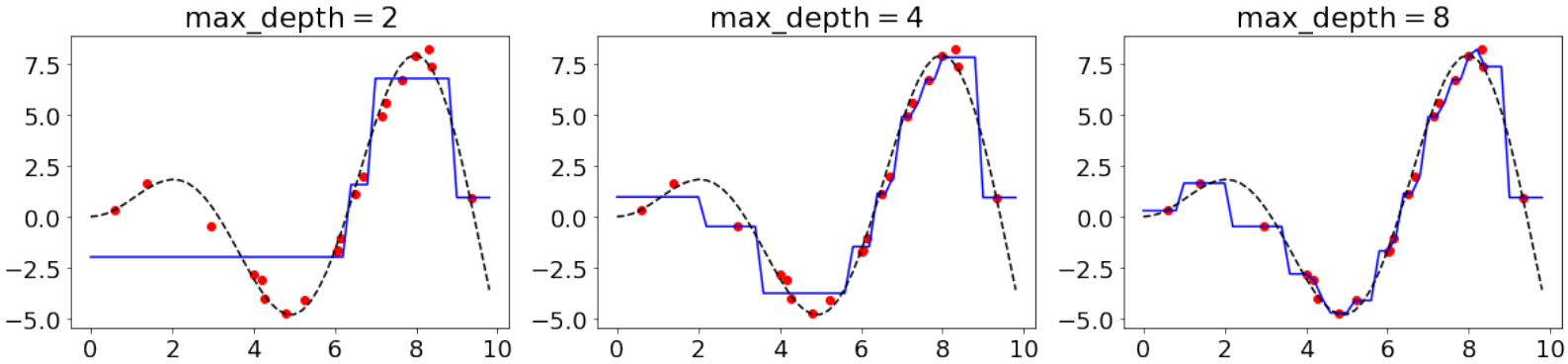


Число элементов в листе ≥ 10

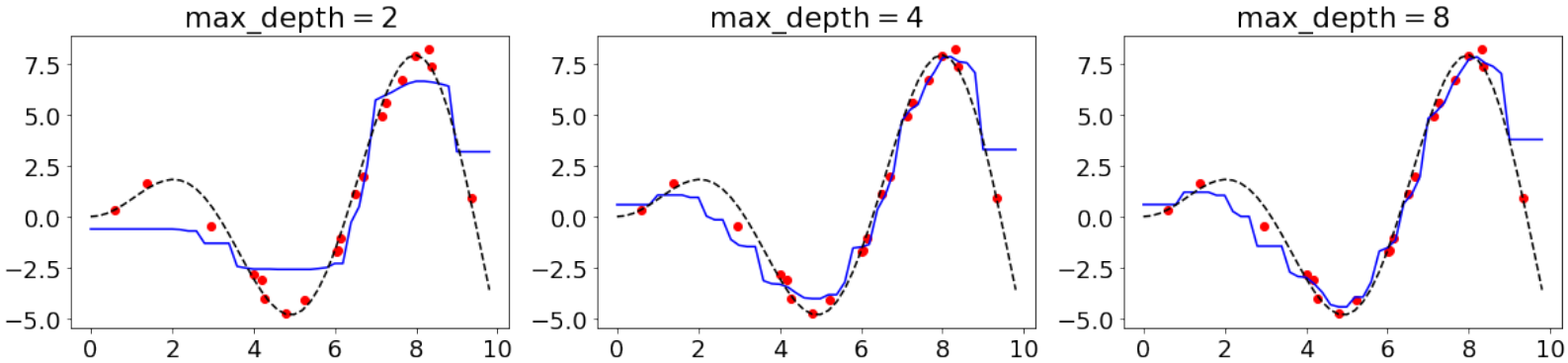


Регрессия с помощью деревьев принятия решений и рандомного леса

Дерево принятия решений



Рандомный лес



Decision Tree и RF классификация и регрессия

Jupyter notebook “Decision Tree и RF классификация и регрессия”:
`https://colab.research.google.com/drive/1oqaczrdb4U0iCDHJezvZzs3pAGP7ZAKt`

Лекция 12. Ансамбли алгоритмов. Беггинг и бустинг

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Ансамбль алгоритмов

Может ли несколько плохих алгоритмов дать вместе хороший результат?

Ансамбль алгоритмов

Может ли несколько плохих алгоритмов дать вместе хороший результат?

- Решается задача бинарной классификации.

Ансамбль алгоритмов

Может ли несколько плохих алгоритмов дать вместе хороший результат?

- Решается задача бинарной классификации.
- Пусть имеется 5 **независимых** алгоритмов $a_j(\vec{x})$, каждый из которых дает правильный ответ с вероятностью $p_0 = 0.6$.

Ансамбль алгоритмов

Может ли несколько плохих алгоритмов дать вместе хороший результат?

- Решается задача бинарной классификации.
- Пусть имеется 5 **независимых** алгоритмов $a_j(\vec{x})$, каждый из которых дает правильный ответ с вероятностью $p_0 = 0.6$.
- Итоговое решение принимается большинством голосов.

Ансамбль алгоритмов

Может ли несколько плохих алгоритмов дать вместе хороший результат?

- Решается задача бинарной классификации.
- Пусть имеется 5 **независимых** алгоритмов $a_j(\vec{x})$, каждый из которых дает правильный ответ с вероятностью $p_0 = 0.6$.
- Итоговое решение принимается большинством голосов.
- Тогда вероятность правильного решения равна

$$p = p_0^5 + C_5^4 p_0^4 (1-p) + C_5^3 p_0^3 (1-p)^2 \approx 0.68$$

.

Ансамбль алгоритмов

Может ли несколько плохих алгоритмов дать вместе хороший результат?

- Решается задача бинарной классификации.
- Пусть имеется 5 **независимых** алгоритмов $a_j(\vec{x})$, каждый из которых дает правильный ответ с вероятностью $p_0 = 0.6$.
- Итоговое решение принимается большинством голосов.
- Тогда вероятность правильного решения равна

$$p = p_0^5 + C_5^4 p_0^4 (1-p) + C_5^3 p_0^3 (1-p)^2 \approx 0.68$$

.

Задача: обучить много независимых алгоритмов

Ансамбль алгоритмов

- Простое голосование: $a(\vec{x}) = \frac{1}{T}\sum_{t=1}^{T} a_t(\vec{x})$.

Ансамбль алгоритмов

- Простое голосование: $a(\vec{x}) = \frac{1}{T}\sum_{t=1}^{T} a_t(\vec{x})$.
- Взвешенное голосование: $a(\vec{x}) = \sum_{t=1}^{T} \alpha_t a_t(\vec{x})$.

Ансамбль алгоритмов

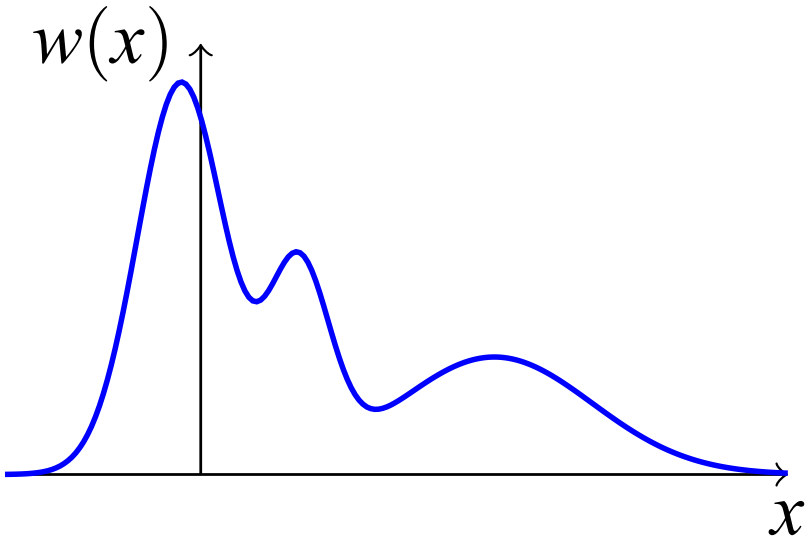
- Простое голосование: $a(\vec{x}) = \frac{1}{T}\sum_{t=1}^{T} a_t(\vec{x})$.
- Взвешенное голосование: $a(\vec{x}) = \sum_{t=1}^{T} \alpha_t a_t(\vec{x})$.
  - Значения $a_t(\vec{x})$ – это новое признаковое описание объекта.
  - Можно подбирать α_t линейными методами.
  - Нужно учитывать ограничения: $\alpha_t > 0; \sum \alpha_t = 1$.

Ансамбль алгоритмов

- Простое голосование: $a(\vec{x}) = \frac{1}{T}\sum_{t=1}^{T} a_t(\vec{x})$.
- Взвешенное голосование: $a(\vec{x}) = \sum_{t=1}^{T} \alpha_t a_t(\vec{x})$.
  - Значения $a_t(\vec{x})$ – это новое признаковое описание объекта.
  - Можно подбирать α_t линейными методами.
  - Нужно учитывать ограничения: $\alpha_t > 0; \sum \alpha_t = 1$.
- Смесь алгоритмов $a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}) a_t(\vec{x}, \vec{\theta}_t)$.
  - $g_t(\vec{x})$ – функции компетентности (gated functions).
  - $g_t(\vec{x})$ определяют области, где работает один или другой алгоритм.

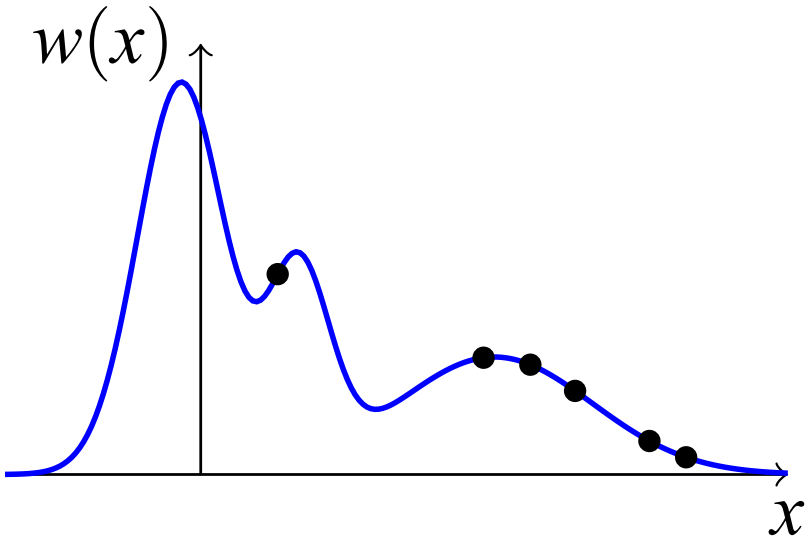
Метод bootstrap

Требуется посчитать статистику Θ распределения $w(x)$



Метод bootstrap

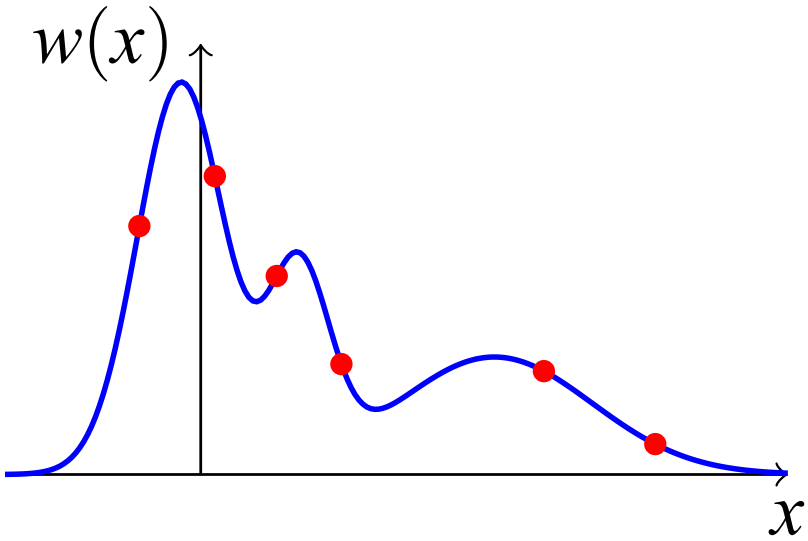
Требуется посчитать статистику Θ распределения $w(x)$



- Генерируется набор объектов $x_j^{(k)}$ из распределения $w(x)$
- Вычисляется выборочная статистика $\Theta_k = \Theta(x^{(k)})$

Метод bootstrap

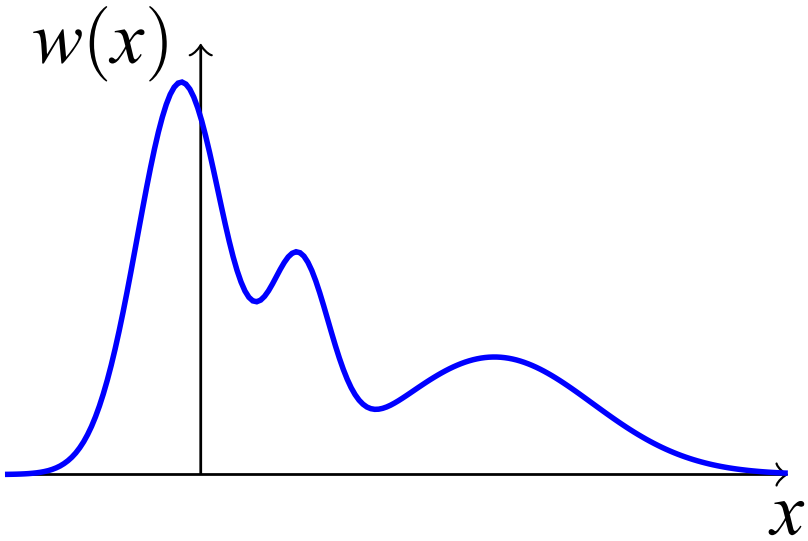
Требуется посчитать статистику Θ распределения $w(x)$



- Генерируется набор объектов $x_j^{(k)}$ из распределения $w(x)$
- Вычисляется выборочная статистика $\Theta_k = \Theta(x^{(k)})$

Метод bootstrap

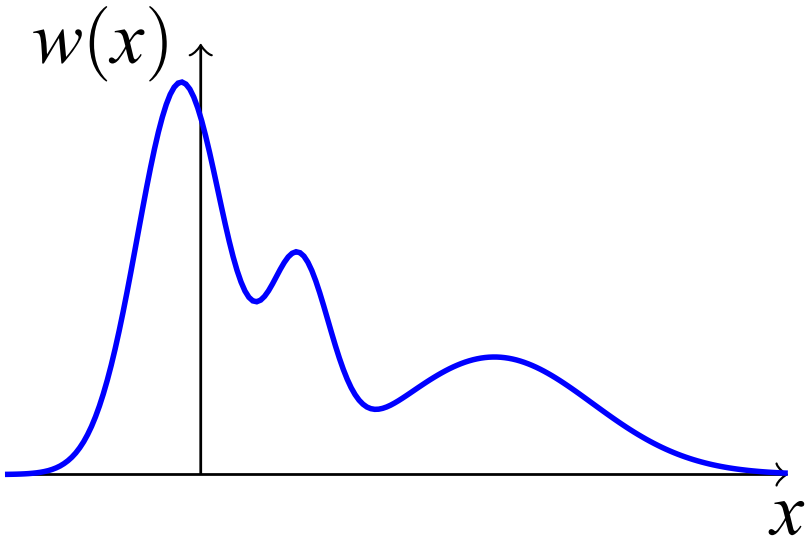
Требуется посчитать статистику Θ распределения $w(x)$



- Генерируется набор объектов $x_j^{(k)}$ из распределения $w(x)$
- Вычисляется выборочная статистика $\Theta_k = \Theta(x^{(k)})$
- Вычисляем среднюю статистику по всем наборам $\hat{\Theta} = \mathcal{M}\{\Theta_k\}$

Метод bootstrap

Требуется посчитать статистику Θ распределения $w(x)$



- Генерируется набор объектов $x_j^{(k)}$ из распределения $w(x)$
- Вычисляется выборочная статистика $\Theta_k = \Theta(x^{(k)})$
- Вычисляем среднюю статистику по всем наборам $\hat{\Theta} = \mathcal{M}\{\Theta_k\}$

Объект может быть выбран многократно.

Методы рандомизации алгоритмов

- Обучение на разных подмножествах объектов.

Методы рандомизации алгоритмов

- Обучение на разных подмножествах объектов.
- Обучение с разными подмножествами признаков.

Методы рандомизации алгоритмов

- Обучение на разных подмножествах объектов.
- Обучение с разными подмножествами признаков.
- Использование разных параметрических моделей.

Методы рандомизации алгоритмов

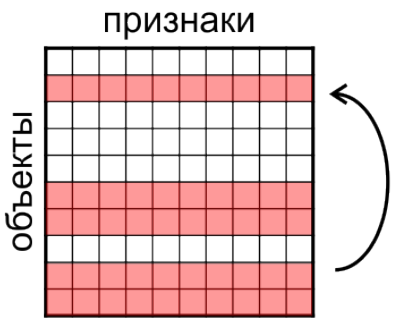
- Обучение на разных подмножествах объектов.
- Обучение с разными подмножествами признаков.
- Использование разных параметрических моделей.
- Использование разных методов обучения.

Методы рандомизации алгоритмов

- Обучение на разных подмножествах объектов.
- Обучение с разными подмножествами признаков.
- Использование разных параметрических моделей.
- Использование разных методов обучения.
- Зашумление данных.

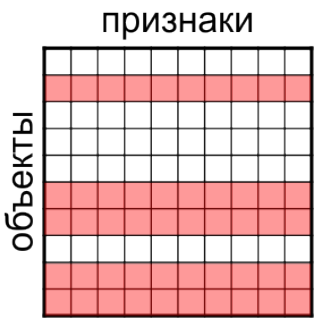
Методы рандомизации алгоритмов

Bagging



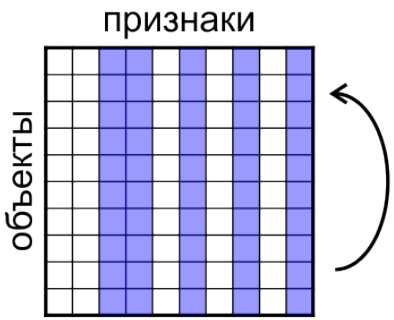
случайные выборки объектов
с возвращением

Pasting



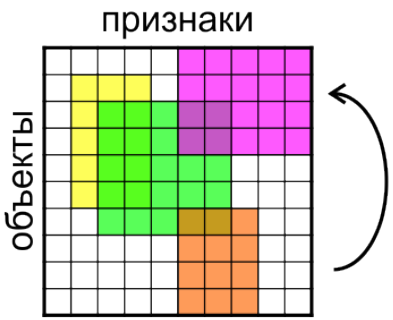
случайные выборки объектов
без возвращения

Random subspaces



случайные выборки признаков
с возвращением

Random patches

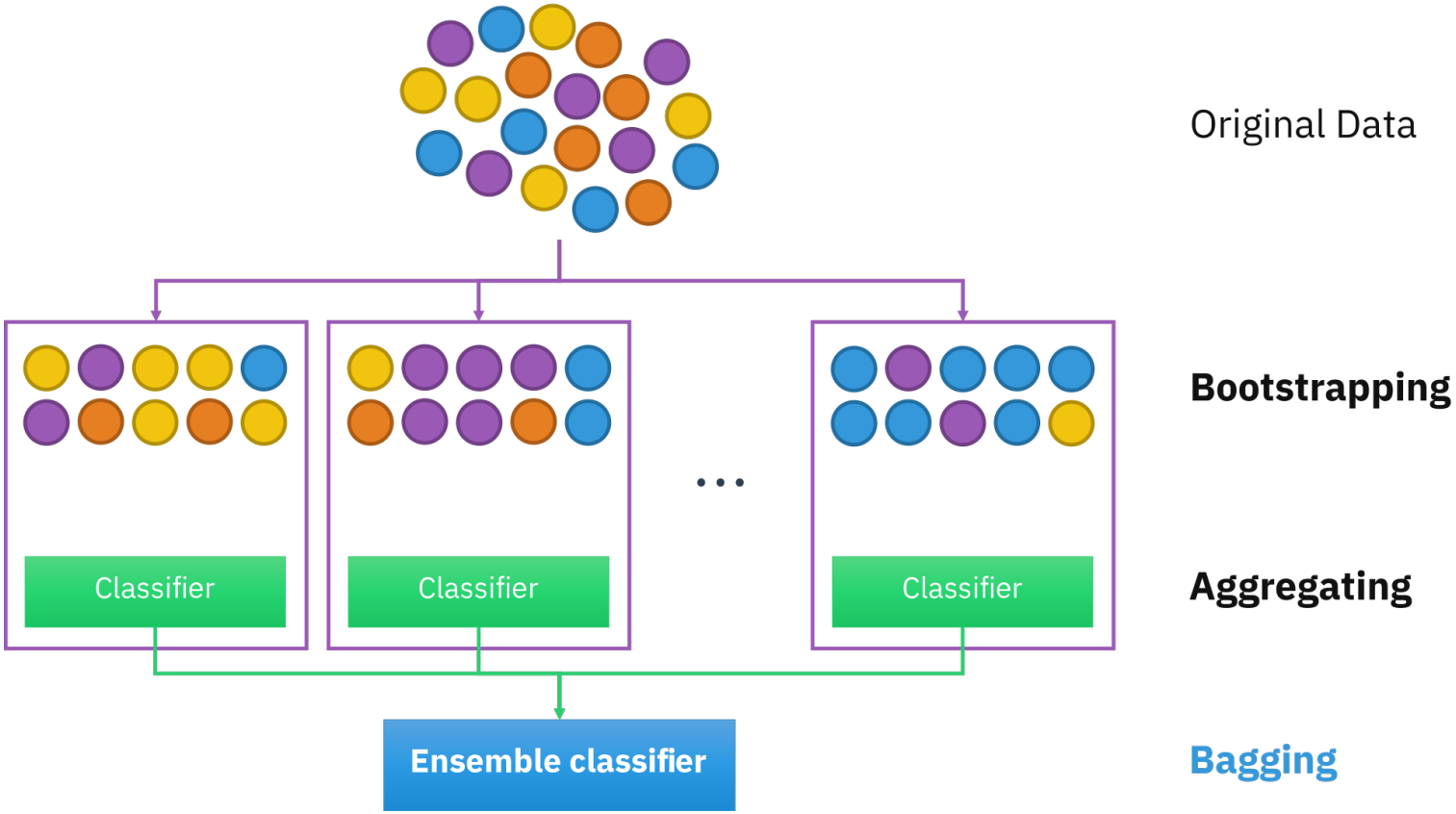


случайные выборки объектов
и признаков с возвращением

Беггинг (bagging)

Bagging = Bootstrap AGGregatING

формирование подвыборок с возвратом объектов



https://en.wikipedia.org/wiki/Bootstrap_aggregating

Беггинг (bagging)

OOB-валидация

- Пусть выборка содержит L объектов.

Беггинг (bagging)

OOB-валидация

- Пусть выборка содержит L объектов.
- Вероятность выбора каждого объекта $p_0 = \frac{1}{L}$, не выбора $1 - \frac{1}{L}$.

Беггинг (bagging)

OOB-валидация

- Пусть выборка содержит L объектов.
- Вероятность выбора каждого объекта $p_0 = \frac{1}{L}$, не выбора $1 - \frac{1}{L}$.
- Пусть процесс выбора повторяется L раз.

Беггинг (bagging)

OOB-валидация

- Пусть выборка содержит L объектов.
- Вероятность выбора каждого объекта $p_0 = \frac{1}{L}$, не выбора $1 - \frac{1}{L}$.
- Пусть процесс выбора повторяется L раз.
- Тогда вероятность не выбора объекта ни разу равна

$$\left(1 - \frac{1}{L}\right)^L \xrightarrow[L\to\infty]{} \frac{1}{e} \approx 37\%$$

.

Беггинг (bagging)

OOB-валидация

- Пусть выборка содержит L объектов.
- Вероятность выбора каждого объекта $p_0 = \frac{1}{L}$, не выбора $1 - \frac{1}{L}$.
- Пусть процесс выбора повторяется L раз.
- Тогда вероятность не выбора объекта ни разу равна

$$\left(1 - \frac{1}{L}\right)^L \xrightarrow[L\to\infty]{} \frac{1}{e} \approx 37\%$$

.

- Не выбранные объекты образуют подвыборку Out-Of-Bag. По ним можно осуществлять валидацию модели.

Беггинг (bagging)

Дополнительные возможности OOB-валидации

- Оценка OOB на каждом объекте выборки считается только по множеству алгоритмов $T(\vec{x})$ длиной $L(T(\vec{x}))$, которые на этом объекте не обучались:

$$\text{OOB}(\vec{x}) = \frac{1}{L(T(\vec{x}))} \sum_{j \in T(\vec{x})} a_j(\vec{x}).$$

- Ошибка ансамбля на $\mathcal{X}_{\text{train}}$ равна

$$\mathcal{L}_{\text{OOB}} = \sum_{\vec{x}_i \in \mathcal{X}_{\text{train}}} \mathcal{L}(\text{OOB}(\vec{x}_i), y_i)$$

- Важность признака $\text{Imp}_j = \dfrac{\mathcal{L}^{(j)}_{\text{OOB}} - \mathcal{L}_{\text{OOB}}}{\mathcal{L}_{\text{OOB}}}$

Рандомный лес – частный случай беггинга

- В качестве базовых алгоритмов используются решающие деревья.
- Такие алгоритмы неустойчивы без регуляризации – то, что нужно!
- Обычно используется $n/3$ признаков при классификации и $\sqrt{n}$ признаков при регрессии в каждой вершине.
- Могут вводиться ограничения на сложность дерева.

Бустинг (boosting)

Идея: использовать взвешенное голосование алгоритмов, добавляя новые алгоритмы и подбирая их веса по одному.

Бустинг (boosting)

Идея: использовать взвешенное голосование алгоритмов, добавляя новые алгоритмы и подбирая их веса по одному.

Бинарная классификация: каждый алгоритм $a_j(\vec{x})$ возвращает класс ± 1 или отказывается от классификации (возвращает 0).

$$a(\vec{x}) = \sum_{j=1}^{T} \alpha_j a_j(\vec{x})$$

Бустинг (boosting)

Идея: использовать взвешенное голосование алгоритмов, добавляя новые алгоритмы и подбирая их веса по одному.

Бинарная классификация: каждый алгоритм $a_j(\vec{x})$ возвращает класс ± 1 или отказывается от классификации (возвращает 0).

$$a(\vec{x}) = \sum_{j=1}^{T} \alpha_j a_j(\vec{x})$$

$$Q_T = \frac{1}{L} \sum_{i=1}^{L} [a(\vec{x}_i) y_i < 0] = \frac{1}{L} \sum_{i=1}^{L} \left[y_i \sum_{j=1}^{T} \alpha_j a_j(\vec{x}_i) < 0 \right] \to \min$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$[M < 0] \leq \exp(-M)$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$[M < 0] \leq \exp(-M)$$

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\left[y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i) < 0\right] \leq \frac{1}{L}\sum_{i=1}^{L}\exp\left[-y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i)\right] \equiv \tilde{Q}_T$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$[M < 0] \leq \exp(-M)$$

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\left[y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i) < 0\right] \leq \frac{1}{L}\sum_{i=1}^{L}\exp\left[-y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i)\right] \equiv \tilde{Q}_T$$

Пусть первые $T-1$ алгоритмов a_j и их веса α_j уже подобраны.

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$[M < 0] \leq \exp(-M)$$

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\left[y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i) < 0\right] \leq \frac{1}{L}\sum_{i=1}^{L}\exp\left[-y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i)\right] \equiv \tilde{Q}_T$$

Пусть первые $T-1$ алгоритмов a_j и их веса α_j уже подобраны.

$$\tilde{Q}_T = \sum_{i=1}^{L}\underbrace{\frac{1}{L}\exp\left[-y_i\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)\right]}_{w_i} e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)}$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$[M < 0] \leq \exp(-M)$$

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\left[y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i) < 0\right] \leq \frac{1}{L}\sum_{i=1}^{L}\exp\left[-y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i)\right] \equiv \tilde{Q}_T$$

Пусть первые $T-1$ алгоритмов a_j и их веса α_j уже подобраны.

$$\tilde{Q}_T = \sum_{i=1}^{L}\underbrace{\frac{1}{L}\exp\left[-y_i\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)\right]}_{w_i} e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)} = \sum_{i=1}^{L} w_i e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)}$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$[M < 0] \leq \exp(-M)$$

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\left[y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i) < 0\right] \leq \frac{1}{L}\sum_{i=1}^{L}\exp\left[-y_i\sum_{j=1}^{T}\alpha_j a_j(\vec{x}_i)\right] \equiv \tilde{Q}_T$$

Пусть первые $T-1$ алгоритмов a_j и их веса α_j уже подобраны.

$$\tilde{Q}_T = \sum_{i=1}^{L}\underbrace{\frac{1}{L}\exp\left[-y_i\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)\right]}_{w_i} e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)} = \sum_{i=1}^{L} w_i e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)} =$$

$$= e^{-\alpha_T}\underbrace{\sum_{i=1}^{L} w_i[y_i a_T(\vec{x}) = 1]}_{P_w} + e^{\alpha_T}\underbrace{\sum_{i=1}^{L} w_i[y_i a_T(\vec{x}) = -1]}_{N_w} + \underbrace{\sum_{i=1}^{L} w_i[y_i a_T(\vec{x}) = 0]}_{Z_w}$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right]$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right] = \tilde{Q}_{T-1}$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right] = \tilde{Q}_{T-1}$$

$$\tilde{Q}_T = P_w e^{-\alpha_T} + N_w e^{\alpha_T} + Z_w \xrightarrow[\alpha_T; a_T]{} \min$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right] = \tilde{Q}_{T-1}$$

$$\tilde{Q}_T = P_w e^{-\alpha_T} + N_w e^{\alpha_T} + Z_w \xrightarrow[\alpha_T; a_T]{} \min$$

$$\frac{\partial \tilde{Q}_T}{\partial \alpha_T} = -P_w e^{-\alpha_T} + N_w e^{\alpha_T} = 0 \quad \Rightarrow \quad \alpha_T = \frac{1}{2} \ln \frac{P_w}{N_w}$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right] = \tilde{Q}_{T-1}$$

$$\tilde{Q}_T = P_w e^{-\alpha_T} + N_w e^{\alpha_T} + Z_w \xrightarrow[\alpha_T; a_T]{} \min$$

$$\frac{\partial \tilde{Q}_T}{\partial \alpha_T} = -P_w e^{-\alpha_T} + N_w e^{\alpha_T} = 0 \quad \Rightarrow \quad \alpha_T = \frac{1}{2} \ln \frac{P_w}{N_w}$$

$$\frac{\tilde{Q}_T}{\tilde{Q}_{T-1}} = \frac{P_w}{\tilde{Q}_{T-1}} \left(\frac{P_w}{N_w}\right)^{-1/2} + \frac{N_w}{\tilde{Q}_{T-1}} \left(\frac{P_w}{N_w}\right)^{1/2} + \frac{Z_w}{\tilde{Q}_{T-1}}$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right] = \tilde{Q}_{T-1}$$

$$\tilde{Q}_T = P_w e^{-\alpha_T} + N_w e^{\alpha_T} + Z_w \xrightarrow[\alpha_T; a_T]{} \min$$

$$\frac{\partial \tilde{Q}_T}{\partial \alpha_T} = -P_w e^{-\alpha_T} + N_w e^{\alpha_T} = 0 \quad \Rightarrow \quad \alpha_T = \frac{1}{2} \ln \frac{P_w}{N_w}$$

$$\frac{\tilde{Q}_T}{\tilde{Q}_{T-1}} = \frac{P_w}{\tilde{Q}_{T-1}} \left(\frac{P_w}{N_w}\right)^{-1/2} + \frac{N_w}{\tilde{Q}_{T-1}} \left(\frac{P_w}{N_w}\right)^{1/2} + \frac{Z_w}{\tilde{Q}_{T-1}} =$$

$$= 1 - \frac{P_w}{\tilde{Q}_{T-1}} - \frac{N_w}{\tilde{Q}_{T-1}} + 2\sqrt{\frac{P_w}{\tilde{Q}_{T-1}} \frac{N_w}{\tilde{Q}_{T-1}}} = 1 - \left(\sqrt{\frac{P_w}{\tilde{Q}_{T-1}}} - \sqrt{\frac{N_w}{\tilde{Q}_{T-1}}}\right)^2 \xrightarrow[a_T]{} \min$$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$P_w + N_w + Z_w = \sum_{i=1}^{L} w_i = \sum_{i=1}^{L} \frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right] = \tilde{Q}_{T-1}$$

$$\tilde{Q}_T = P_w e^{-\alpha_T} + N_w e^{\alpha_T} + Z_w \xrightarrow[\alpha_T; a_T]{} \min$$

$$\frac{\partial \tilde{Q}_T}{\partial \alpha_T} = -P_w e^{-\alpha_T} + N_w e^{\alpha_T} = 0 \quad \Rightarrow \quad \alpha_T = \frac{1}{2} \ln \frac{P_w}{N_w}$$

$$a_T = \arg\max_{a_j} \left(\sqrt{P_w} - \sqrt{N_w}\right)$$

Если $Z_w = 0$, то $a_T = \arg\min_{a_j} N_w$

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

$$\tilde{Q}_T = \sum_{i=1}^{L} \underbrace{\frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right]}_{w_i} e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)}$$

Вес объекта w_i тем больше, чем меньше его отступ $M_i = y_i a(\vec{x}_i)$, т.е. чем больше ошибка всех предыдущих алгоритмов на этом объекте.

Бустинг (boosting)

Алгоритм адаптивного бустинга (AdaBoost)

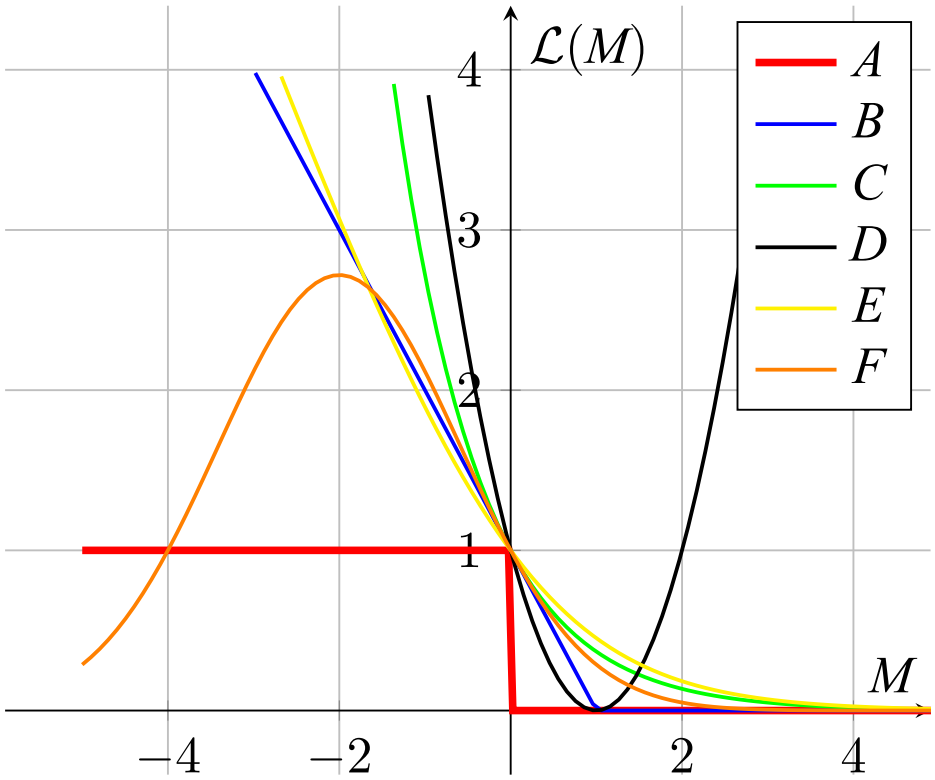
$$\tilde{Q}_T = \sum_{i=1}^{L} \underbrace{\frac{1}{L} \exp\left[-y_i \sum_{j=1}^{T-1} \alpha_j a_j(\vec{x}_i)\right]}_{w_i} e^{-\alpha_T \cdot y_i a_T(\vec{x}_i)}$$

Вес объекта w_i тем больше, чем меньше его отступ $M_i = y_i a(\vec{x}_i)$, т.е. чем больше ошибка всех предыдущих алгоритмов на этом объекте.

Нужно исключать объекты-выбросы, вес которых растет экспоненциально.

Бустинг (boosting)

Варианты функции потерь при бустинге



- $A : [M < 0]$ – пороговая
- $B : (1 - M)_+$ – SVM
- $C : e^{-M}$ – экспоненциальная (AdaBoost)
- $D : (1 - M)^2$ – квадратичная (GentleBoost)
- $E : \log_2(1 + e^{-M})$ – логарифмическая (LogitBoost)
- $F : \exp(-cM(M + s))$ – гауссовская (BrownBoost)

Градиентный бустинг (Gradient Boosting Machine)

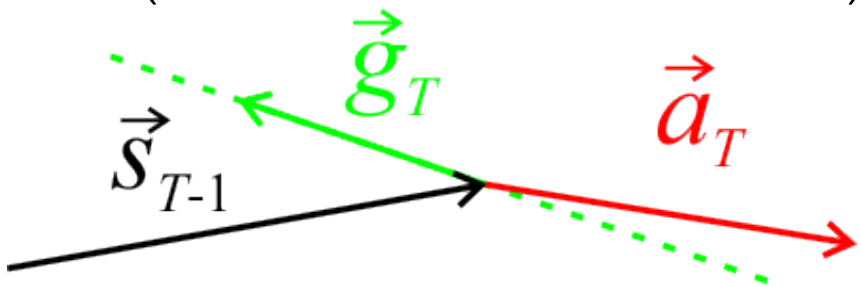
$$a(\vec{x}) = \sum_{j=1}^{T} \alpha_j a_j(\vec{x}); \quad Q_T = \frac{1}{L} \sum_{i=1}^{L} \mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$$

Градиентный бустинг (Gradient Boosting Machine)

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}\left(\underbrace{\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)}_{s_{T-1,i}} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i\right) \to \min$$

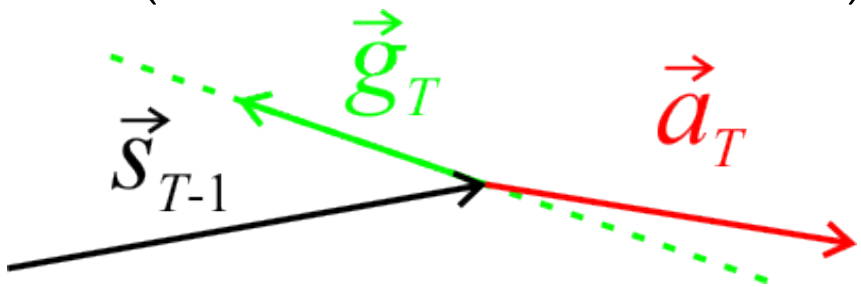
Градиентный бустинг (Gradient Boosting Machine)

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}\left(\underbrace{\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)}_{s_{T-1,i}} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i\right) \to \min$$



Градиентный бустинг (Gradient Boosting Machine)

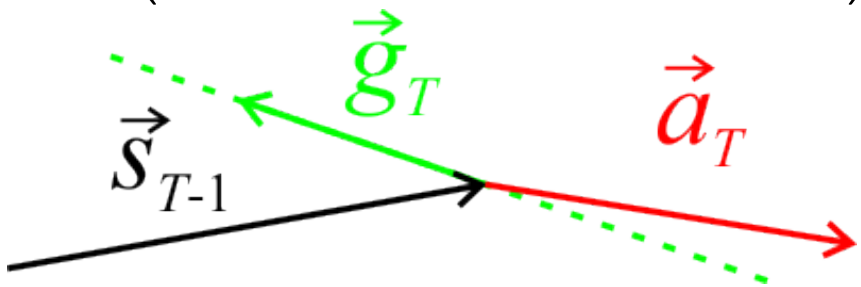
$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}\left(\underbrace{\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)}_{s_{T-1,i}} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i\right) \to \min$$



1. Вычисляется градиент $\vec{g}_T = \nabla_a \mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$

Градиентный бустинг (Gradient Boosting Machine)

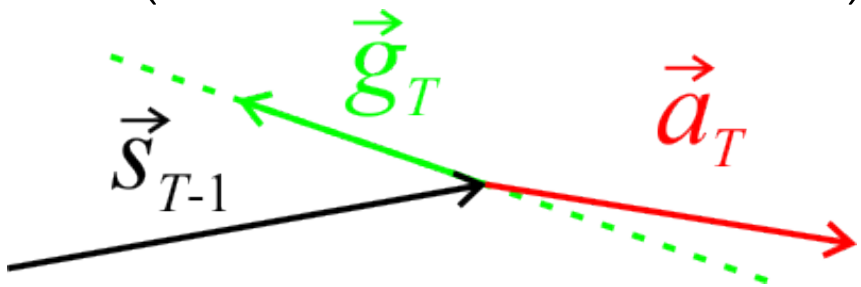
$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}\left(\underbrace{\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)}_{s_{T-1,i}} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i\right) \to \min$$



1. Вычисляется градиент $\vec{g}_T = \nabla_a \mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$
2. Подбирается алгоритм $a_T = \arg\min_a \sum_{i=1}^{L}(a(\vec{x}_i) + g_{T,i})^2$

Градиентный бустинг (Gradient Boosting Machine)

$$Q_T = \frac{1}{L}\sum_{i=1}^{L}\mathcal{L}\left(\underbrace{\sum_{j=1}^{T-1}\alpha_j a_j(\vec{x}_i)}_{s_{T-1,i}} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i\right) \to \min$$



1. Вычисляется градиент $\vec{g}_T = \nabla_a \mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$
2. Подбирается алгоритм $a_T = \arg\min_a \sum_{i=1}^{L}(a(\vec{x}_i) + g_{T,i})^2$
3. Находится $\alpha_T = \arg\min_{\alpha>0} \sum_{i=1}^{L}\mathcal{L}(s_{T-1,i} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i)$

Стохастический градиентный бустинг (SGB)

Идея: шаги алгоритма выполняются не на всей выборке $\mathcal{X}_{train}$, а на ее случайной подвыборке $\vec{x}_i \in \mathcal{X}'_{train} \subset \mathcal{X}_{train}$ (60-80% выборки).

1. Вычисляется градиент $\vec{g}_T = \nabla_a \mathcal{L}(a(\vec{x}_i), \vec{x}_i, y_i)$
2. Подбирается алгоритм $a_T = \arg\min_a \sum_{i=1}^{L} (a(\vec{x}_i) + g_{T,i})^2$
3. Находится $\alpha_T = \arg\min_{\alpha>0} \sum_{i=1}^{L} \mathcal{L}(\vec{s}_{T-1,i} + \alpha_T a_T(\vec{x}_i), \vec{x}_i, y_i)$

Плюсы:

- уменьшается время обучения;
- возможность работать с выборками большего объема;
- появляются возможности оценок *OOB*;
- в целом улучшается сходимость и обобщающая способность.

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

- Дерево решений представимо в виде $a(\vec{x},\vec{\theta}) = \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x})$,
  где v – листовая вершина; θ_v– значение в ней; $\Theta_v \Theta_{v'} = \delta(v,v')$.

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

- Дерево решений представимо в виде $a(\vec{x},\vec{\theta}) = \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x})$,
  где v – листовая вершина; θ_v– значение в ней; $\Theta_v \Theta_{v'} = \delta(v,v')$.
- $a_T^2(\vec{x},\vec{\theta}) = \left(\sum_{v=1}^{V} \theta_v \Theta_v(\vec{x}) \right)^2 = \sum_{v=1}^{V} \theta_v^2 \Theta_v(\vec{x})$

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

- Дерево решений представимо в виде $a(\vec{x},\vec{\theta}) = \sum\limits_{v=1}^{V} \theta_v \Theta_v(\vec{x})$,
  где v – листовая вершина; θ_v– значение в ней; $\Theta_v \Theta_{v'} = \delta(v,v')$.
- $a_T^2(\vec{x},\vec{\theta}) = \left(\sum\limits_{v=1}^{V} \theta_v \Theta_v(\vec{x}) \right)^2 = \sum\limits_{v=1}^{V} \theta_v^2 \Theta_v(\vec{x})$
- Критерий качества выбирается как $Q(\vec{\theta}) \to \min$

$$Q(\vec{\theta}) = \sum_{i=1}^{L} \mathcal{L}(s_{T-1,i} + a_T(\vec{x},\vec{\theta}), \vec{x}_i, y_i) + \underbrace{\xi V}_{L_0\text{-регуляризация}} + \underbrace{\lambda \sum_{v=1}^{V} \theta_v^2}_{L_2\text{-регуляризация}}$$

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

$$a(\vec{x}, \vec{\theta}) = \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x})$$

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

$$a(\vec{x}, \vec{\theta}) = \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x})$$

- Разложение в ряд до 2 порядка включительно:

$$\mathcal{L}(s_{T-1,i} + a_T(\vec{x},\vec{\theta}), \vec{x}_i, y_i) = \mathcal{L}(s_{T-1,i}, \vec{x}_i, y_i) + g_i a_T(\vec{x},\vec{\theta}) + \frac{h_i}{2} a_T^2(\vec{x},\vec{\theta})$$

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

$$a(\vec{x},\vec{\theta}) = \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x})$$

- Разложение в ряд до 2 порядка включительно:

$$\mathcal{L}(s_{T-1,i} + a_T(\vec{x},\vec{\theta}), \vec{x}_i, y_i) = \mathcal{L}(s_{T-1,i}, \vec{x}_i, y_i) + g_i a_T(\vec{x},\vec{\theta}) + \frac{h_i}{2} a_T^2(\vec{x},\vec{\theta})$$

- Тогда условие на минимум

$$Q(\vec{\theta}) = \sum_{i=1}^{L} \left[\mathcal{L}(s_{T-1,i}, \vec{x}_i, y_i) + g_i a_T(\vec{x},\vec{\theta}) + \frac{h_i}{2} a_T^2(\vec{x},\vec{\theta}) \right] + \xi V + \lambda \sum_{v=1}^{V} \theta_v^2 =$$

$$= \sum_{i=1}^{L} \left[g_i \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x}) + \frac{h_i}{2} \sum_{v=1}^{V} \theta_v^2 \Theta_v(\vec{x}) \right] + \xi V + \lambda \sum_{v=1}^{V} \theta_v^2 =$$

$$= \sum_{i=1}^{L} \left[\sum_{v=1}^{V} \left(\frac{h_i}{2} \theta_v^2 + g_i \theta_v \right) \Theta_v(\vec{x}) \right] + \xi V + \lambda \sum_{v=1}^{V} \theta_v^2 \to \min$$

Градиентный бустинг над деревьями (XGBoost, eXtreme Gradient Boosting)

$$a(\vec{x}, \vec{\theta}) = \sum_{v=1}^{V} \theta_v \Theta_v(\vec{x})$$

- Значения в листах: $\theta_v = -\dfrac{\sum\limits_{v=1}^{V} g_i \Theta_v(\vec{x}_i)}{2\lambda + \sum\limits_{v=1}^{V} h_i \Theta_v(\vec{x}_i)}$
- Условия на $\Theta_v(\vec{x})$: $-\dfrac{1}{2} \sum\limits_{v=1}^{V} \dfrac{(g_i \Theta_v(\vec{x}_i))^2}{2\lambda + h_i \Theta_v(\vec{x}_i)} + \xi V \to \min$

  Это критерий для построения дерева.

Сравнение беггинга и бустинга

Беггинг

- Лучше подходит для относительно коротких выборок.
- Легко распараллеливается по отдельным алгоритмам.
- “Встроенные” оценки *OOB*.
- Итоговый алгоритм не интерпретируем.

Бустинг

- Больше подходит для задач со сложной границей классов.
- Алгоритмы подбираются последовательно, но можно распараллеливать каждый алгоритм в отдельности.
- Итоговый алгоритм не интерпретируем.

Нелинейные ансамбли алгоритмов. Стекинг (stacking)

Идея: обучить ансамбль алгоритмов, и их ответы – метапризнаки подать на вход метаалгоритму.

Основная проблема – формирование подвыборок. Обучение базовых алгоритмов и метаалгоритма на общих данных приводит к переобучению.

Блендинг (blending)

1. Обучающая выборка разделяется на $\mathcal{X}_{\text{train}}$ и $\mathcal{X}_{\text{val}}$.
2. $\mathcal{X}_{\text{train}}$ разделяется на $\mathcal{X}_{\text{base}}$ и $\mathcal{X}_{\text{meta}}$.
3. Базовые алгоритмы обучаются на $\mathcal{X}_{\text{base}}$.
4. Метаалгоритм обучается на метапризнаках – результатах работы базовых алгоритмов на $\mathcal{X}_{\text{meta}}$.
5. Валидация происходит на $\mathcal{X}_{\text{val}}$.

Блендинг (blending)

1. Обучающая выборка разделяется на $\mathcal{X}_{\text{train}}$ и $\mathcal{X}_{\text{val}}$.
2. $\mathcal{X}_{\text{train}}$ разделяется на $\mathcal{X}_{\text{base}}$ и $\mathcal{X}_{\text{meta}}$.
3. Базовые алгоритмы обучаются на $\mathcal{X}_{\text{base}}$.
4. Метаалгоритм обучается на метапризнаках – результатах работы базовых алгоритмов на $\mathcal{X}_{\text{meta}}$.
5. Валидация происходит на $\mathcal{X}_{\text{val}}$.

Минус:
ни базовые алгоритмы, ни метаалгоритм не обучаются на всей выборке.

Блендинг (blending)

1. Обучающая выборка разделяется на $\mathcal{X}_{\text{train}}$ и $\mathcal{X}_{\text{val}}$.
2. $\mathcal{X}_{\text{train}}$ разделяется на $\mathcal{X}_{\text{base}}$ и $\mathcal{X}_{\text{meta}}$.
3. Базовые алгоритмы обучаются на $\mathcal{X}_{\text{base}}$.
4. Метаалгоритм обучается на метапризнаках – результатах работы базовых алгоритмов на $\mathcal{X}_{\text{meta}}$.
5. Валидация происходит на $\mathcal{X}_{\text{val}}$.

Минус:
ни базовые алгоритмы, ни метаалгоритм не обучаются на всей выборке.

Идея: использовать несколько различных способов разделения $\mathcal{X}_{\text{train}}$ на $\mathcal{X}_{\text{base}}/\mathcal{X}_{\text{meta}}$ и усреднить результаты. Однако этот способ не дает улучшения качества на практике.

Классический стекинг

1. Обучающая выборка разделяется на k групп объектов.
2. Последовательно выбирается каждая из k групп.
3. Базовые алгоритмы обучаются на остальных $k - 1$ группах объектов, после чего их результаты на выбранной группе подаются на вход метаалгоритма.
4. После обучения метаалгоритма базовые алгоритмы заново обучаются на всей выборке.

Классический стекинг

1. Обучающая выборка разделяется на k групп объектов.
2. Последовательно выбирается каждая из k групп.
3. Базовые алгоритмы обучаются на остальных $k - 1$ группах объектов, после чего их результаты на выбранной группе подаются на вход метаалгоритма.
4. После обучения метаалгоритма базовые алгоритмы заново обучаются на всей выборке.

Минус:
метапризнаки на обучении и на тесте оказываются разными.

Классический стекинг

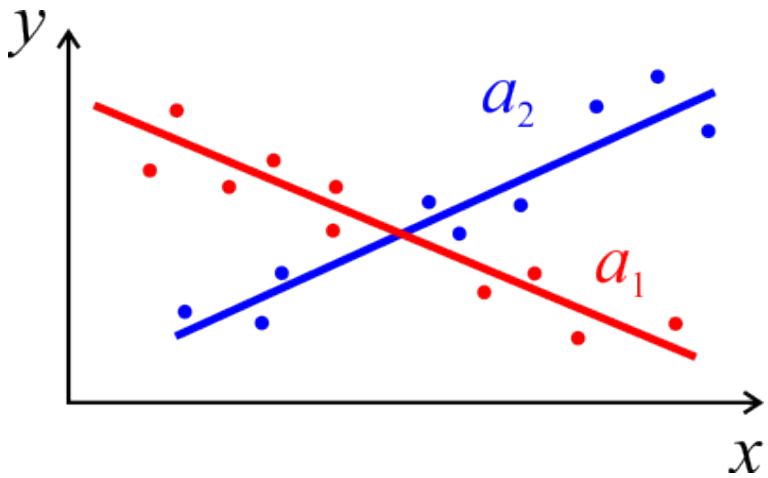
1. Обучающая выборка разделяется на k групп объектов.
2. Последовательно выбирается каждая из k групп.
3. Базовые алгоритмы обучаются на остальных $k - 1$ группах объектов, после чего их результаты на выбранной группе подаются на вход метаалгоритма.
4. После обучения метаалгоритма базовые алгоритмы заново обучаются на всей выборке.

Минус:
метапризнаки на обучении и на тесте оказываются разными.

Для компенсации этих эффектов добавляют шум в данные или регуляризацию.

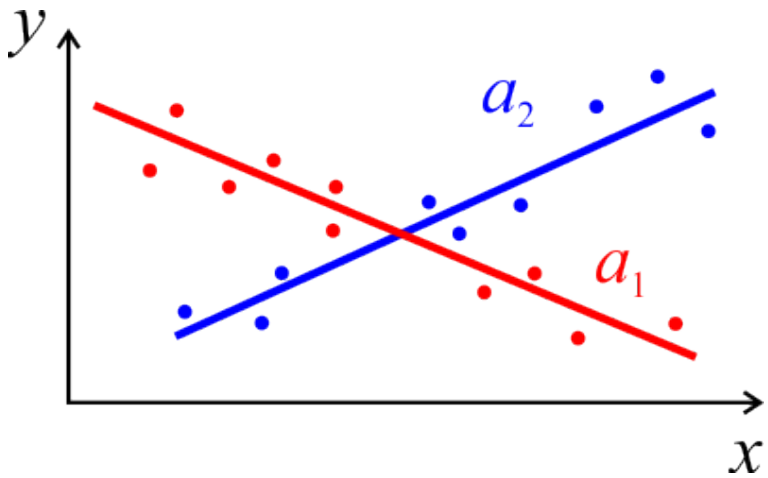
Смесь алгоритмов (Mixture of Experts)

$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}, \vec{v}_t) a_t(\vec{x}, \vec{\theta}_t)$$



Смесь алгоритмов (Mixture of Experts)

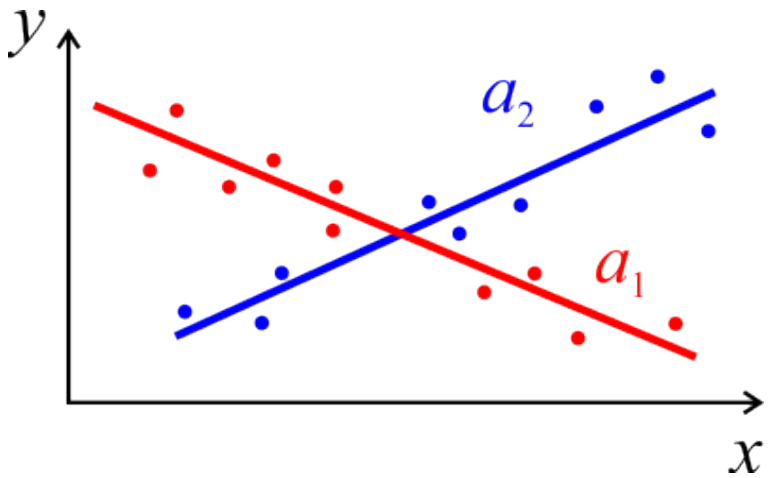
$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}, \vec{v}_t) a_t(\vec{x}, \vec{\theta}_t)$$



- Функция компетентности $g_t(\vec{x}, \vec{v}_t)$ (gated function).

Смесь алгоритмов (Mixture of Experts)

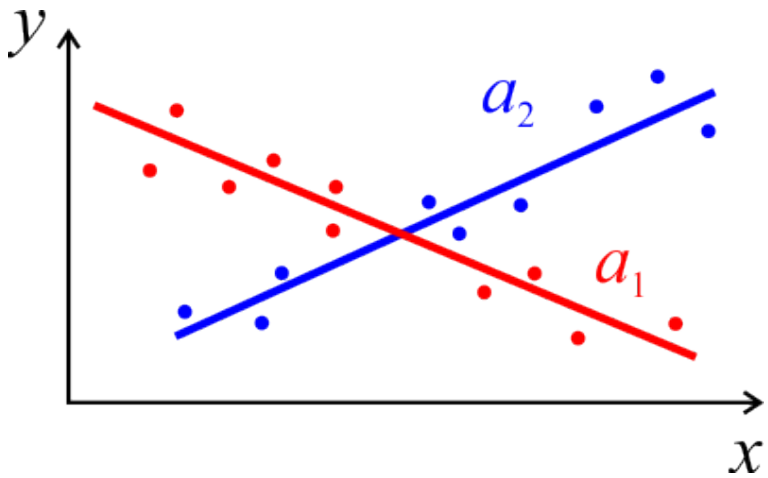
$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}, \vec{v}_t) a_t(\vec{x}, \vec{\theta}_t)$$



- Функция компетентности $g_t(\vec{x}, \vec{v}_t)$ (gated function).
- Это вероятность описания объекта x алгоритмом a_t.

Смесь алгоритмов (Mixture of Experts)

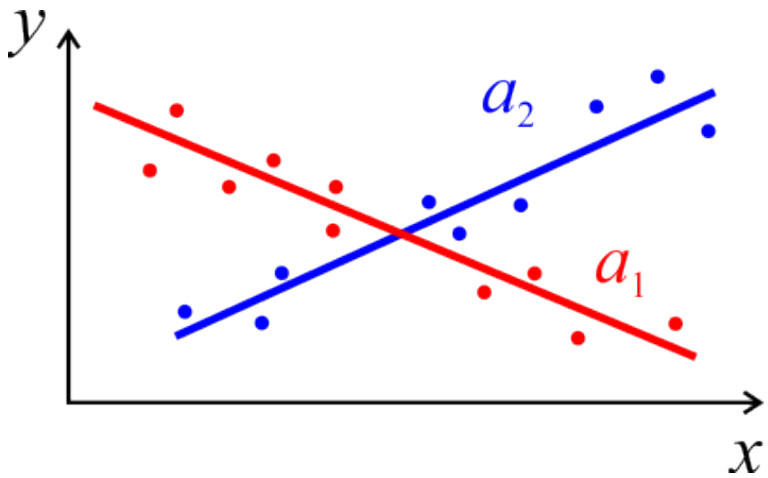
$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}, \vec{v}_t) a_t(\vec{x}, \vec{\theta}_t)$$



- Функция компетентности $g_t(\vec{x}, \vec{v}_t)$ (gated function).
- Это вероятность описания объекта x алгоритмом a_t.
- $\sum_{t=1}^{T} g_t(\vec{x}\,) = 1; \quad 0 \le g_t(\vec{x}) \le 1$

Смесь алгоритмов (Mixture of Experts)

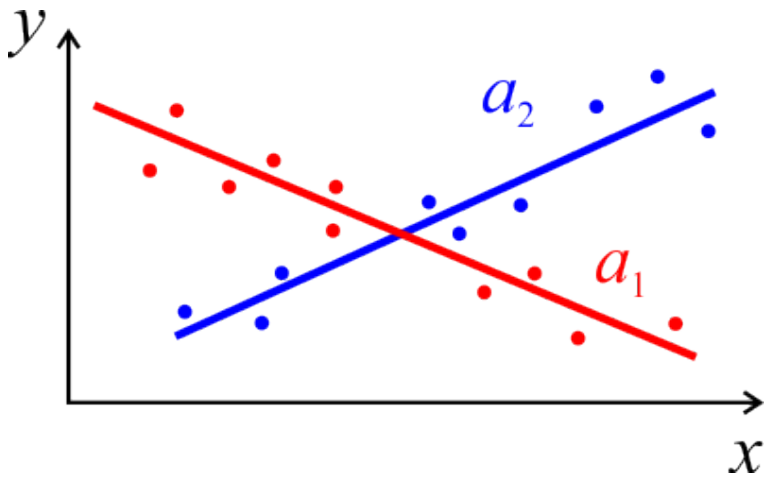
$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}, \vec{v}_t) a_t(\vec{x}, \vec{\theta}_t)$$



- Функция компетентности $g_t(\vec{x}, \vec{v}_t)$ (gated function).
- Это вероятность описания объекта x алгоритмом a_t.
- $\sum_{t=1}^{T} g_t(\vec{x}\,) = 1; \quad 0 \le g_t(\vec{x}) \le 1$
- Требуется определить число моделей T и оценить x параметры $\vec{v}_t, \vec{\theta}_t$

Смесь алгоритмов (Mixture of Experts)

$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}, \vec{v}_t) a_t(\vec{x}, \vec{\theta}_t)$$



- Функция компетентности $g_t(\vec{x}, \vec{v}_t)$ (gated function).
- Это вероятность описания объекта x алгоритмом a_t.
- $\sum_{t=1}^{T} g_t(\vec{x}\,) = 1; \quad 0 \le g_t(\vec{x}) \le 1$
- Требуется определить число моделей T и оценить x параметры $\vec{v}_t, \vec{\theta}_t$
- Expectation-Maximization итерационно уточняет оценки $\vec{v}_t$ и $\vec{\theta}$.

Сравнение методов машинного обучения

Jupyter notebook “Сравнение методов машинного обучения”:
`https://colab.research.google.com/drive/1uYZV5Fsz0y4o-GSBfsj0rlQCQRsrISKw`

Лекция 13. Искусственные нейронные сети. Базовые архитектуры и методы обучения

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Схема постановки и решения задач обучения с учителем

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.
- Выбирается параметрическая модель $a(\vec{x}, \vec{\theta})$
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума $\mathcal{L}$.

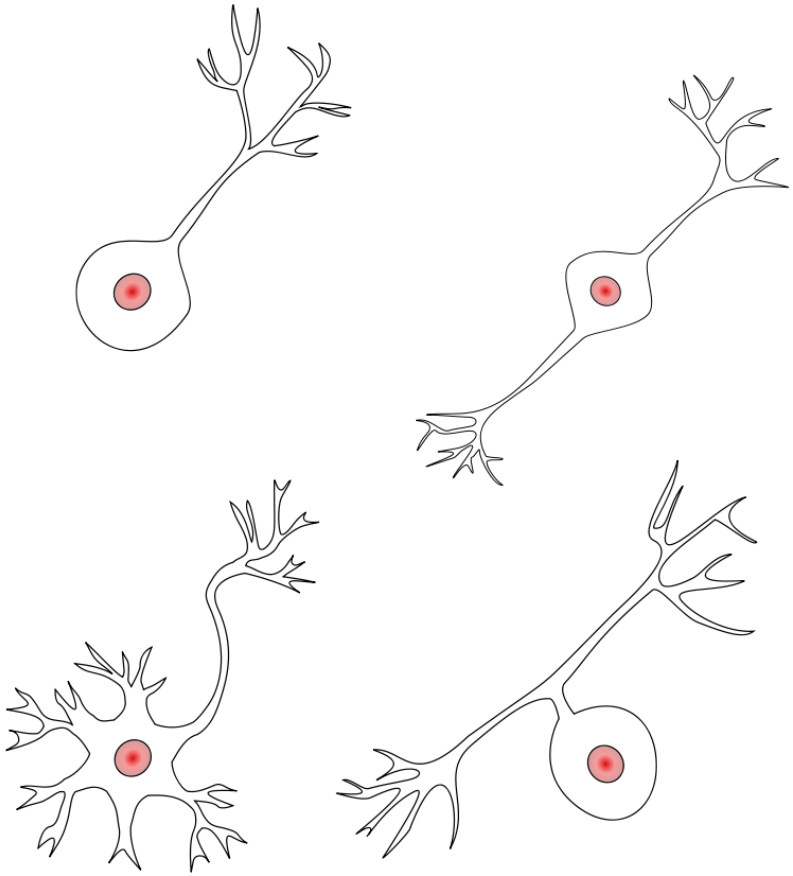
Схема постановки и решения задач обучения с учителем

- Имеется множество объектов $\mathbb{X}$, каждый из которых задан вектором признаков $\vec{x}$
- Имеется обучающая выборка объектов $\mathbb{X}_{\text{train}} \subset \mathbb{X}$ из L объектов, для которых задан целевой призак y
- Задается функционал качества $\mathcal{L}(a, \vec{x}, y)$, характеризующий, насколько хорошо алгоритм a аппроксимирует значения целевого признака для объектов из множества $\mathbb{X}$.
- Выбирается параметрическая модель $a(\vec{x}, \vec{\theta})$
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума $\mathcal{L}$.

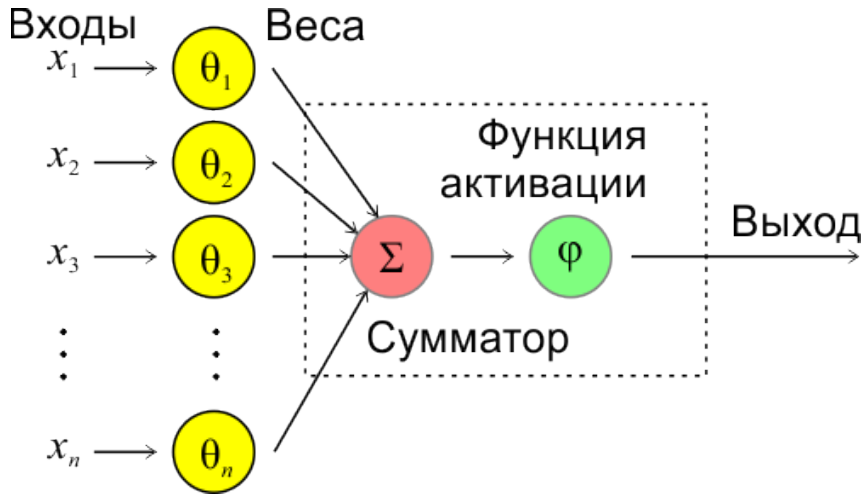
Как действовать, если параметрическая модель $a(\vec{x}, \vec{\theta})$ не определена или чрезвычайно сложна?

Модель нейрона Мак-Каллока–Питтса

Различные типы нейронов



https://en.wikipedia.org/wiki/Neuron



$$a(\vec{x}) = \varphi\left(\sum x_i \theta_i\right)$$

Ограниченность единственного нейрона

Бинарный классификатор: $a(\vec{x}) = \text{sgn}\left(\sum x_i \theta_i\right)$

- Может описать дизъюнкцию: $x \vee y = [x + y - 1/2 > 0]$.
- Может описать конъюнкцию: $x \wedge y = [x + y - 3/2 > 0]$.
- Может описать отрицание: $\neg x = [-x + 1/2 > 0]$.
- Не может описать исключающее ИЛИ.

Ограниченность единственного нейрона

Бинарный классификатор: $a(\vec{x}) = \operatorname{sgn}\left(\sum x_i \theta_i\right)$

- Может описать дизъюнкцию: $x \vee y = [x + y - 1/2 > 0]$.
- Может описать конъюнкцию: $x \wedge y = [x + y - 3/2 > 0]$.
- Может описать отрицание: $\neg x = [-x + 1/2 > 0]$.
- Не может описать исключающее ИЛИ.

Можно ввести нелинейность: $\mathrm{XOR}(x, y) = [x + y - 2xy - 1/2 > 0]$.

Ограниченность единственного нейрона

Бинарный классификатор: $a(\vec{x}) = \text{sgn}\left(\sum x_i\theta_i\right)$

- Может описать дизъюнкцию: $x \vee y = [x + y - 1/2 > 0]$.
- Может описать конъюнкцию: $x \wedge y = [x + y - 3/2 > 0]$.
- Может описать отрицание: $\neg x = [-x + 1/2 > 0]$.
- Не может описать исключающее ИЛИ.

Можно ввести нелинейность: $\text{XOR}(x, y) = [x + y - 2xy - 1/2 > 0]$.
Можно ввести 2 слой: $\text{XOR}(x, y) = [x \vee y - x \wedge y - 1/2 > 0]$.

Ограниченность единственного нейрона

Бинарный классификатор: $a(\vec{x}) = \text{sgn}\left(\sum x_i\theta_i\right)$

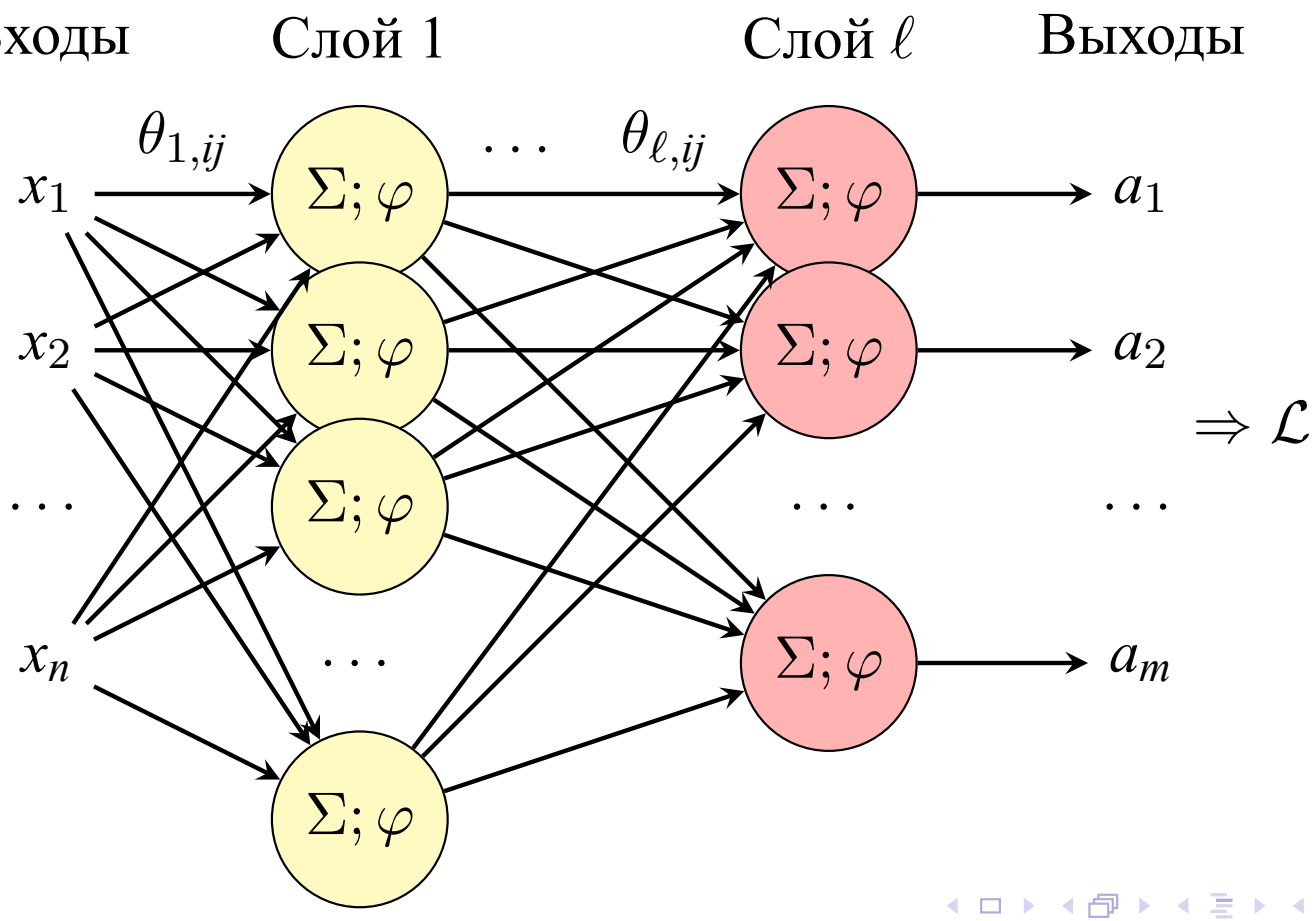
- Может описать дизъюнкцию: $x \vee y = [x + y - 1/2 > 0]$.
- Может описать конъюнкцию: $x \wedge y = [x + y - 3/2 > 0]$.
- Может описать отрицание: $\neg x = [-x + 1/2 > 0]$.
- Не может описать исключающее ИЛИ.

Можно ввести нелинейность: $\text{XOR}(x, y) = [x + y - 2xy - 1/2 > 0]$.
Можно ввести 2 слой: $\text{XOR}(x, y) = [x \vee y - x \wedge y - 1/2 > 0]$.

Любая булева функция – это дизъюнкция конъюнкций. Поэтому двуслойной сети достаточно для описания всех возможностей.

Полносвязная нейронная сеть прямого распространения

- Каждый нейрон соединен со всеми нейронами предыдущего слоя.
- Связи не образуют циклов.



Теоретические возможности не глубоких сетей

- Сеть из 2 слоев в пространстве $\{0; 1\}$ описывает любую функцию.

Теоретические возможности не глубоких сетей

- Сеть из 2 слоев в пространстве $\{0; 1\}$ описывает любую функцию.
- Сеть из 1 слоя в пространстве $\mathcal{R}^n$ описывает полупространство.

Теоретические возможности не глубоких сетей

- Сеть из 2 слоев в пространстве $\{0; 1\}$ описывает любую функцию.
- Сеть из 1 слоя в пространстве $\mathcal{R}^n$ описывает полупространство.
- Сеть из 2 слоев в пространстве $\mathcal{R}^n$ описывает выпуклый многогранник.

Теоретические возможности не глубоких сетей

- Сеть из 2 слоев в пространстве $\{0; 1\}$ описывает любую функцию.
- Сеть из 1 слоя в пространстве $\mathcal{R}^n$ описывает полупространство.
- Сеть из 2 слоев в пространстве $\mathcal{R}^n$ описывает выпуклый многогранник.
- Сеть из 3 слоев в пространстве $\mathcal{R}^n$ описывает многогранную область достаточно произвольного вида.

Теоретические возможности не глубоких сетей

- Сеть из 2 слоев в пространстве $\{0; 1\}$ описывает любую функцию.
- Сеть из 1 слоя в пространстве $\mathcal{R}^n$ описывает полупространство.
- Сеть из 2 слоев в пространстве $\mathcal{R}^n$ описывает выпуклый многогранник.
- Сеть из 3 слоев в пространстве $\mathcal{R}^n$ описывает многогранную область достаточно произвольного вида.
- Универсальная теорема аппроксимации: двуслойная нейронная сеть прямой связи может аппроксимировать любую непрерывную функцию многих переменных с любой точностью.

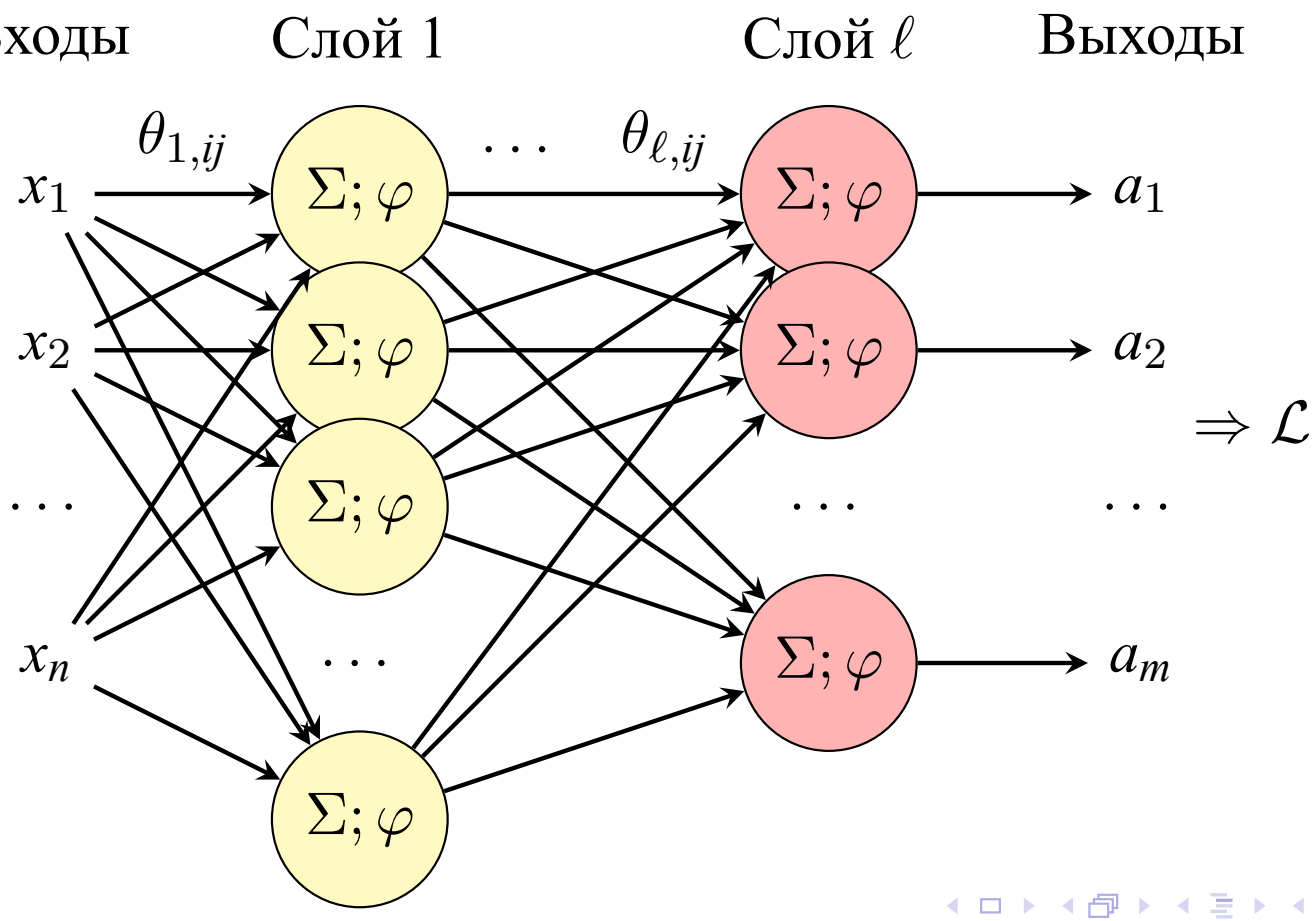
Теоретические возможности не глубоких сетей

- Сеть из 2 слоев в пространстве $\{0; 1\}$ описывает любую функцию.
- Сеть из 1 слоя в пространстве $\mathcal{R}^n$ описывает полупространство.
- Сеть из 2 слоев в пространстве $\mathcal{R}^n$ описывает выпуклый многогранник.
- Сеть из 3 слоев в пространстве $\mathcal{R}^n$ описывает многогранную область достаточно произвольного вида.
- Универсальная теорема аппроксимации: двуслойная нейронная сеть прямой связи может аппроксимировать любую непрерывную функцию многих переменных с любой точностью.

Нужны ли более глубокие нейронные сети?

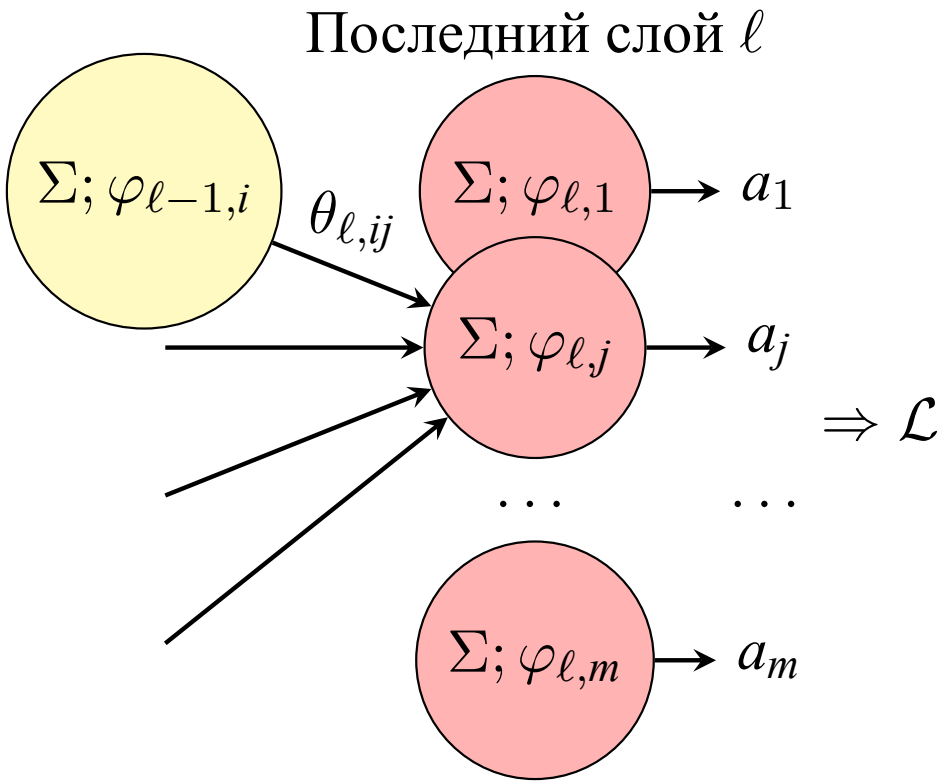
Обучение сети. Метод обратного распространения

1. Прямой шаг: на основе $\vec{x}$ рассчитывается $\vec{a}(\vec{x})$ и $\mathcal{L}(\vec{a}(\vec{x}), \vec{x}, \vec{y})$.
2. Обратный шаг: определить градиент по параметрам $\nabla_\theta \mathcal{L}(\vec{a}(\vec{x}), \vec{x}, \vec{y})$.



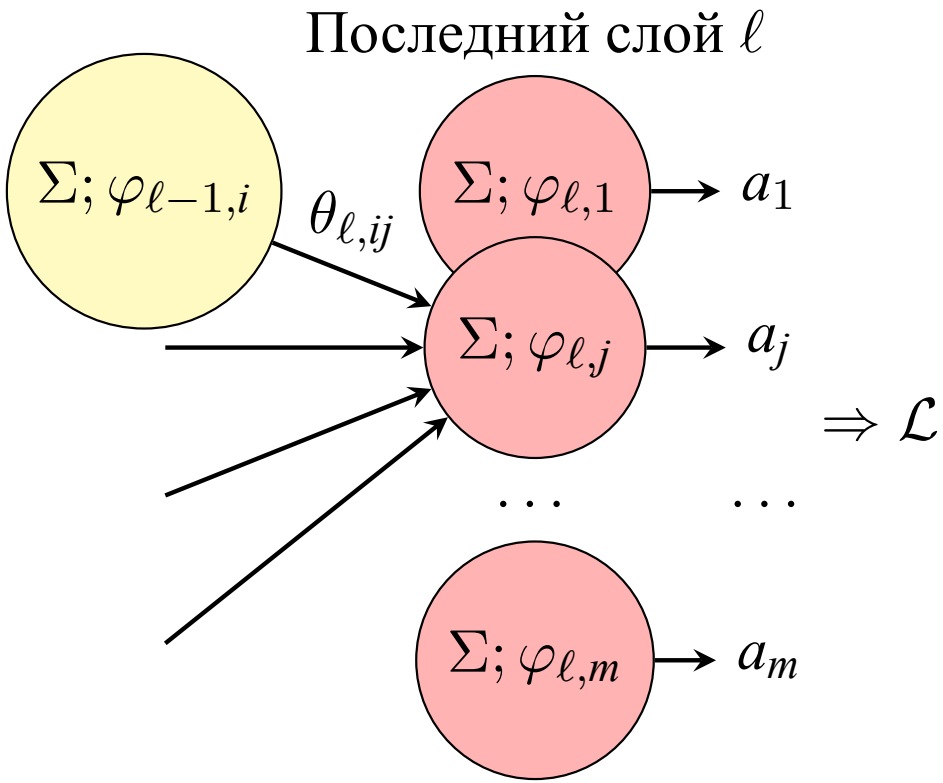
Обучение сети. Метод обратного распространения

Последний слой



Обучение сети. Метод обратного распространения

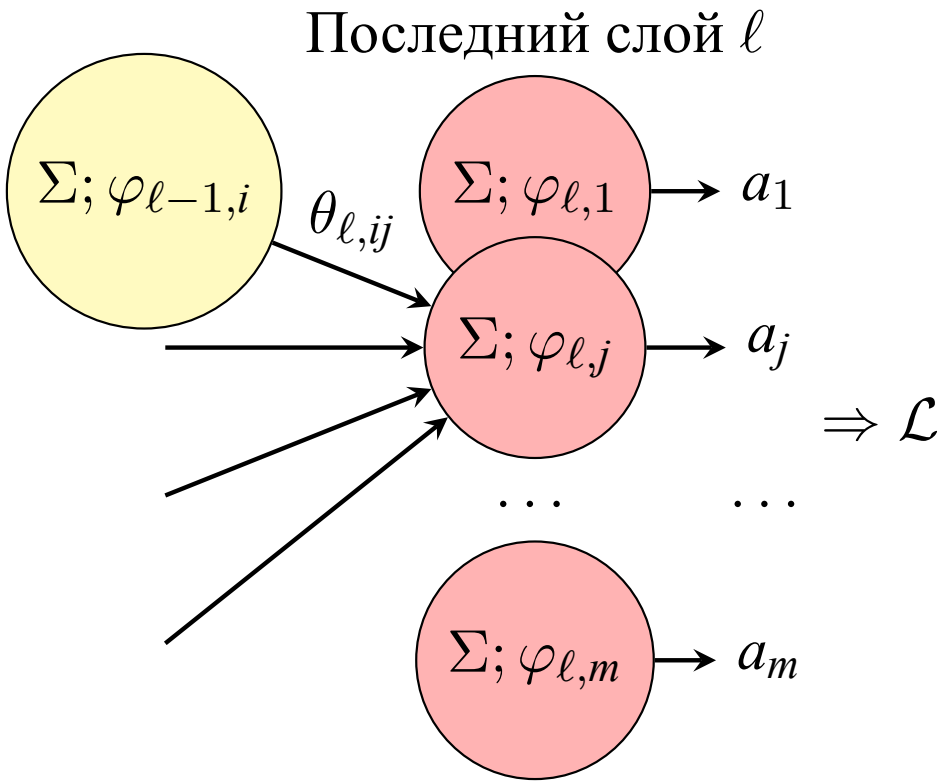
Последний слой



$$\frac{\partial \mathcal{L}}{\partial \theta_{\ell,ij}} = \frac{\partial \mathcal{L}}{\partial a_j} \frac{\partial a_j}{\partial \theta_{\ell,ij}}$$

Обучение сети. Метод обратного распространения

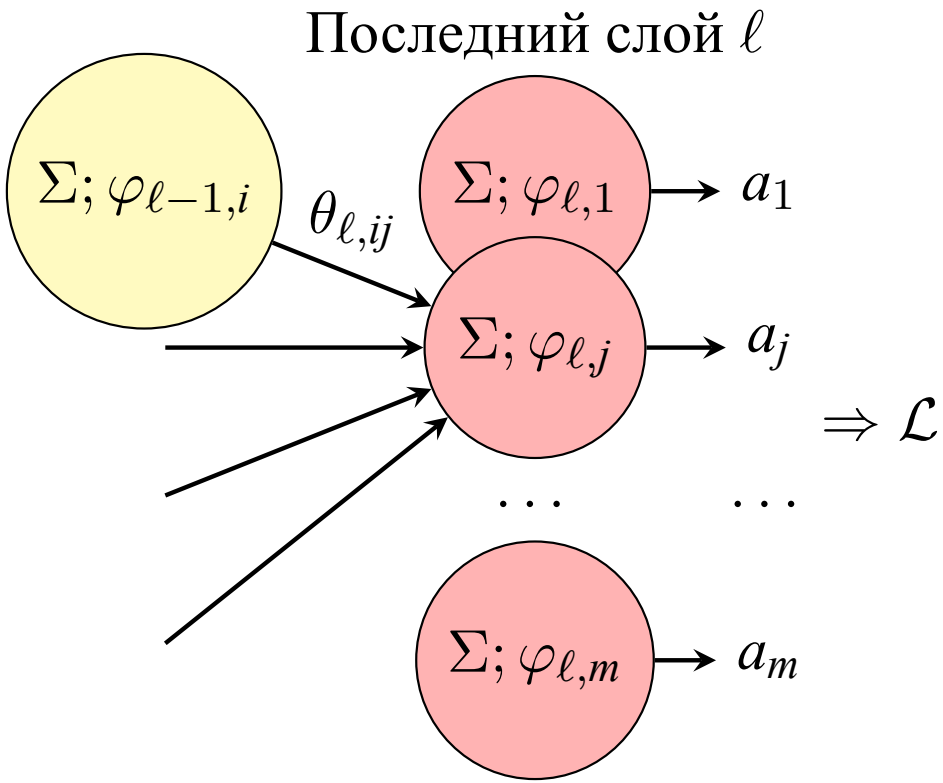
Последний слой



$$\frac{\partial \mathcal{L}}{\partial \theta_{\ell,ij}} = \frac{\partial \mathcal{L}}{\partial a_j}\frac{\partial a_j}{\partial \theta_{\ell,ij}} = \frac{\partial \mathcal{L}}{\partial \varphi_{\ell,j}}\frac{\partial \varphi_{\ell,j}}{\partial \theta_{\ell,ij}}$$

Обучение сети. Метод обратного распространения

Последний слой



$$\frac{\partial \mathcal{L}}{\partial \theta_{\ell,ij}} = \frac{\partial \mathcal{L}}{\partial a_j}\frac{\partial a_j}{\partial \theta_{\ell,ij}} = \frac{\partial \mathcal{L}}{\partial \varphi_{\ell,j}}\frac{\partial \varphi_{\ell,j}}{\partial \theta_{\ell,ij}} = \frac{\partial \mathcal{L}}{\partial \varphi_{\ell,j}}\varphi'_{\ell,j}\varphi_{\ell-1,i}$$

Обучение сети. Метод обратного распространения

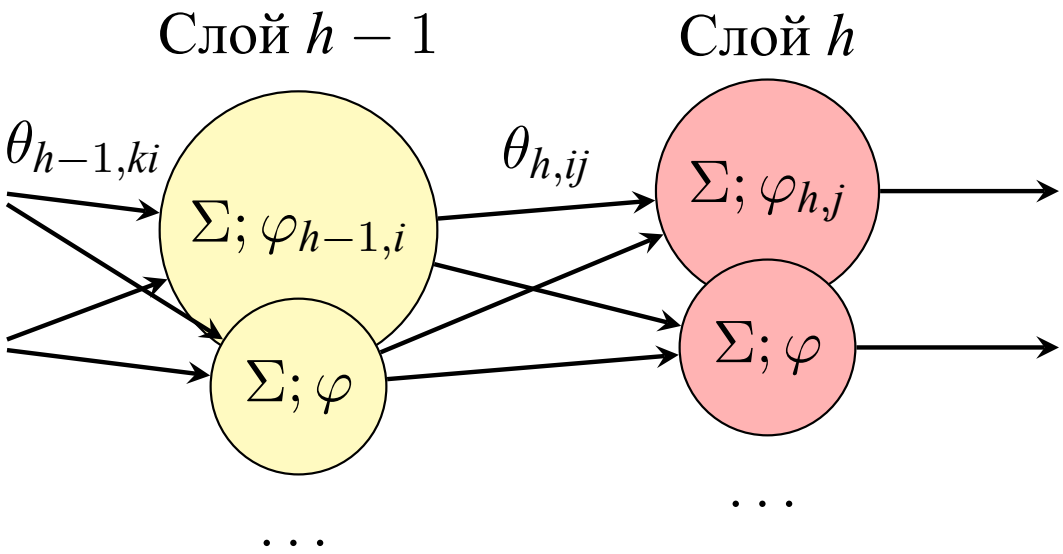
Последний слой

$$\frac{\partial \mathcal{L}}{\partial \theta_{\ell,ij}} = \underbrace{\frac{\partial \mathcal{L}}{\partial \varphi_{\ell,j}}}_{\epsilon_{\ell,j}} \varphi'_{\ell,j} \varphi_{\ell-1,i}$$

- Пусть $\mathcal{L} = \frac{1}{2}\sum_{j=1}^{m}\left(a_j - y_j\right)^2 = \frac{1}{2}\sum_{j=1}^{m}\left(\varphi_{\ell,j} - y_j\right)^2$.
  Тогда $\epsilon_{\ell,j} = a_j - y_j = \epsilon_{\ell,j}$ имеет смысл ошибки j-го нейрона ℓ-го слоя.
- $\varphi_{\ell-1,i}$ – значение, выданное предыдущим слоем.

Обучение сети. Метод обратного распространения

Произвольный слой

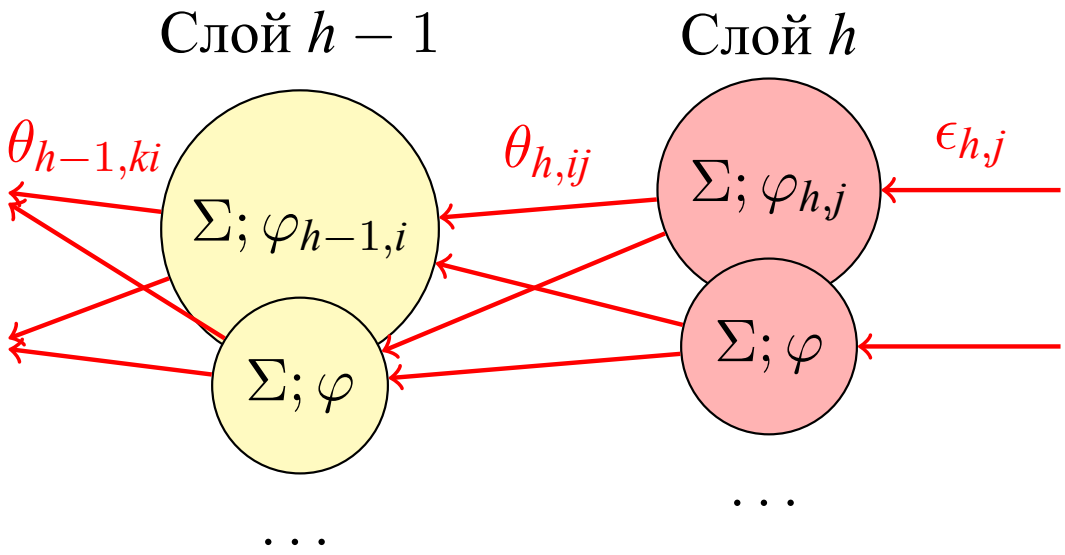


Ошибка слоя: $\epsilon_{h-1,i} = \dfrac{\partial \mathcal{L}}{\partial \varphi_{h-1,i}} = \sum\limits_j \underbrace{\dfrac{\partial \mathcal{L}}{\partial \varphi_{h,j}}}_{\epsilon_{h,j}} \underbrace{\dfrac{\partial \varphi_{h,j}}{\partial \varphi_{h-1,i}}}_{\varphi'_{h,j} \cdot \theta_{h,ij}} = \sum\limits_j \left(\epsilon_{h,j} \cdot \varphi'_{h,j}\right) \cdot \theta_{h,ij}$

Градиент: $\dfrac{\partial \mathcal{L}}{\partial \theta_{h,ij}} = \dfrac{\partial \mathcal{L}}{\partial \varphi_{h,j}} \dfrac{\partial \varphi_{h,j}}{\partial \theta_{h,ij}} = \left(\epsilon_{h,j} \cdot \varphi'_{h,j}\right) \cdot \varphi_{h-1,i}$

Обучение сети. Метод обратного распространения

Произвольный слой



Ошибка слоя: $\epsilon_{h-1,i} = \dfrac{\partial \mathcal{L}}{\partial \varphi_{h-1,i}} = \sum\limits_j \underbrace{\dfrac{\partial \mathcal{L}}{\partial \varphi_{h,j}}}_{\epsilon_{h,j}} \underbrace{\dfrac{\partial \varphi_{h,j}}{\partial \varphi_{h-1,i}}}_{\varphi'_{h,j} \cdot \theta_{h,ij}} = \sum\limits_j \left(\epsilon_{h,j} \cdot \varphi'_{h,j}\right) \cdot \theta_{h,ij}$

Градиент: $\dfrac{\partial \mathcal{L}}{\partial \theta_{h,ij}} = \dfrac{\partial \mathcal{L}}{\partial \varphi_{h,j}} \dfrac{\partial \varphi_{h,j}}{\partial \theta_{h,ij}} = \left(\epsilon_{h,j} \cdot \varphi'_{h,j}\right) \cdot \varphi_{h-1,i}$

Проблемы при обучении нейронных сетей

- Переобучение.
- Медленная сходимость.
- Локальные минимумы.

Проблемы при обучении нейронных сетей

- Переобучение.
- Медленная сходимость.
- Локальные минимумы.
- Затухание и взрыв градиента (vanishing/exploding gradients).

Проблемы при обучении нейронных сетей

- Переобучение.
- Медленная сходимость.
- Локальные минимумы.
- Затухание и взрыв градиента (vanishing/exploding gradients).
  - Пусть значения на h-м слое $x_h = \varphi_h(x_{h-1}\theta_h)$ (1 нейрон в слое).

Проблемы при обучении нейронных сетей

- Переобучение.
- Медленная сходимость.
- Локальные минимумы.
- Затухание и взрыв градиента (vanishing/exploding gradients).
  - Пусть значения на h-м слое $x_h = \varphi_h(x_{h-1}\theta_h)$ (1 нейрон в слое).
  - $$\frac{\partial \mathcal{L}}{\partial \theta_h} = \frac{\partial \mathcal{L}}{\partial \varphi_\ell}\frac{\partial \varphi_\ell}{\partial \varphi_{\ell-1}} \cdot \ldots \cdot \frac{\partial \varphi_{h+1}}{\partial \varphi_h} \cdot \varphi_h' \cdot x_{h-1}$$

Проблемы при обучении нейронных сетей

- Переобучение.
- Медленная сходимость.
- Локальные минимумы.
- Затухание и взрыв градиента (vanishing/exploding gradients).
  - Пусть значения на h-м слое $x_h = \varphi_h(x_{h-1}\theta_h)$ (1 нейрон в слое).
  - $\dfrac{\partial \mathcal{L}}{\partial \theta_h} = \dfrac{\partial \mathcal{L}}{\partial \varphi_\ell} \dfrac{\partial \varphi_\ell}{\partial \varphi_{\ell-1}} \cdot \ldots \cdot \dfrac{\partial \varphi_{h+1}}{\partial \varphi_h} \cdot \varphi'_h \cdot x_{h-1}$
  - Если $\left|\dfrac{\partial \varphi_{k+1}}{\partial \varphi_k}\right| > 1$, то $\dfrac{\partial \mathcal{L}}{\partial \theta_h} \to \infty$ – взрыв градиента.
  - Решение – клиппирование градиента (gradient clipping): если $\|\vec{g}\| > g_0$, то $\vec{g} := \vec{g} \cdot g_0 / \|\vec{g}\|$.

Проблемы при обучении нейронных сетей

- Переобучение.
- Медленная сходимость.
- Локальные минимумы.
- Затухание и взрыв градиента (vanishing/exploding gradients).
  - Пусть значения на h-м слое $x_h = \varphi_h(x_{h-1}\theta_h)$ (1 нейрон в слое).
  - $\dfrac{\partial \mathcal{L}}{\partial \theta_h} = \dfrac{\partial \mathcal{L}}{\partial \varphi_\ell} \dfrac{\partial \varphi_\ell}{\partial \varphi_{\ell-1}} \cdot \ldots \cdot \dfrac{\partial \varphi_{h+1}}{\partial \varphi_h} \cdot \varphi'_h \cdot x_{h-1}$
  - Если $\left|\dfrac{\partial \varphi_{k+1}}{\partial \varphi_k}\right| > 1$, то $\dfrac{\partial \mathcal{L}}{\partial \theta_h} \to \infty$ – взрыв градиента.
  - Решение – клиппирование градиента (gradient clipping): если $\|\vec{g}\| > g_0$, то $\vec{g} := \vec{g} \cdot g_0 / \|\vec{g}\|$.
  - Если $\left|\dfrac{\partial \varphi_{k+1}}{\partial \varphi_k}\right| < 1$, то $\dfrac{\partial \mathcal{L}}{\partial \theta_h} \to 0$ – затухание градиента (паралич сети).
  - Решение – отказ от сигмоидных функций активации.

Выбор функции активации

- Пороговая функция: $\varphi(x) = [x \geq 0]$.

Выбор функции активации

- Пороговая функция: $\varphi(x) = [x \geq 0]$.
- Линейная функция: $\varphi(x) = kx$.

Выбор функции активации

- Пороговая функция: $\varphi(x) = [x \geq 0]$.
- Линейная функция: $\varphi(x) = kx$.
- Сигмоидная функция: $\varphi(x) = \text{sigmoid}(x) \equiv \dfrac{1}{1 + e^{-x}}$.

Выбор функции активации

- Пороговая функция: $\varphi(x) = [x \geq 0]$.
- Линейная функция: $\varphi(x) = kx$.
- Сигмоидная функция: $\varphi(x) = \text{sigmoid}(x) \equiv \dfrac{1}{1+e^{-x}}$.
- Гиперболический тангенс:
  $\varphi(x) = \tanh(x) \equiv \dfrac{2}{1+e^{-2x}} - 1 = 2\text{sigmoid}(2x) - 1.$

Выбор функции активации

- Пороговая функция: $\varphi(x) = [x \geq 0]$.
- Линейная функция: $\varphi(x) = kx$.
- Сигмоидная функция: $\varphi(x) = \text{sigmoid}(x) \equiv \dfrac{1}{1+e^{-x}}$.
- Гиперболический тангенс:
  $\varphi(x) = \tanh(x) \equiv \dfrac{2}{1+e^{-2x}} - 1 = 2\text{sigmoid}(2x) - 1$.
- Rectified Linear Unit: $\varphi(x) = \text{ReLU}(x) \equiv \max(0, x)$.

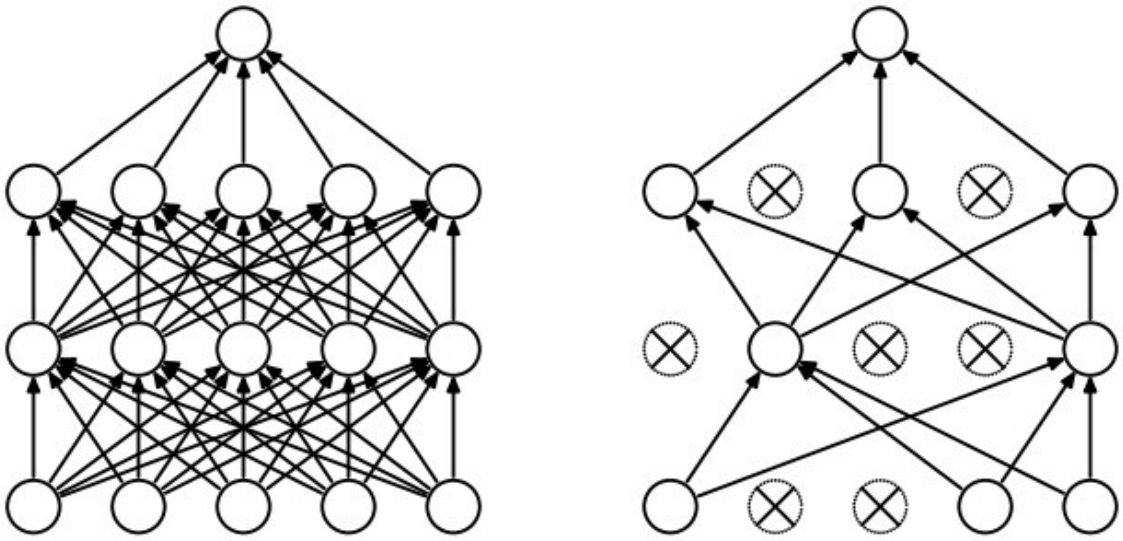
Выбор функции активации

- Пороговая функция: $\varphi(x) = [x \geq 0]$.
- Линейная функция: $\varphi(x) = kx$.
- Сигмоидная функция: $\varphi(x) = \text{sigmoid}(x) \equiv \dfrac{1}{1+e^{-x}}$.
- Гиперболический тангенс:
  $\varphi(x) = \tanh(x) \equiv \dfrac{2}{1+e^{-2x}} - 1 = 2\text{sigmoid}(2x) - 1.$
- Rectified Linear Unit: $\varphi(x) = \text{ReLU}(x) \equiv \max(0, x)$.
- Различные варианты ReLU
  - Leaky ReLU: $\varphi(x) = x; \quad x \geq 0; \quad 0.01x; \quad x < 0.$
  - Parametric ReLU: $\varphi(x) = x; \quad x \geq 0; \quad kx; \quad x < 0.$
  - Gaussian Error Linear Unit (GELU): $\varphi(x) = x\Phi(x)$
  - Sigmoid Linear Unit (SiLU): $\varphi(x) = x\,\text{sigmoid}(x)$.
  - Softplus: $\varphi(x) = \ln(1 + e^x)$.
  - Exponential Linear Unit (ELU): $\varphi(x) = x; \quad x \geq 0; \quad a(e^x - 1); \quad x < 0.$

Метод прореживания (Dropout)

Проблема: нейроны могут настраиваться друг под друга, компенсируя свои ошибки на обучающей выборке, что приводит к переобучению.

Идея: отключать часть нейронов с вероятностью p, обучая ансамбль нейронных сетей и усредняя результат.



(a) Standard Neural Net (b) After applying dropout.

https://arxiv.org/abs/1207.0580

Метод прореживания (Dropout)

Прямой Dropout

- На этапе обучения j-й нейрон h-го слоя
  - с вероятностью p выдает 0.
  - с вероятностью $1-p$ выдает результат $\varphi_{h,j}\left(\sum \varphi_{h-1,i}\theta_{h,ij}\right)$
- На этапе применения j-й нейрон h-го слоя
  - выдает результат $(1-p)\varphi_{h,j}\left(\sum \varphi_{h-1,i}\theta_{h,ij}\right)$

Обратный Dropout

- На этапе обучения j-й нейрон h-го слоя
  - с вероятностью p выдает 0.
  - с вероятностью $1-p$ выдает результат $\frac{1}{1-p}\varphi_{h,j}\left(\sum \varphi_{h-1,i}\theta_{h,ij}\right)$
- На этапе применения j-й нейрон h-го слоя
  - выдает результат $\varphi_{h,j}\left(\sum \varphi_{h-1,i}\theta_{h,ij}\right)$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta} + \Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta} + \Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.
- Лагранжиан $L_i = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta} + \lambda(\theta_i + \vec{e}_i^T \Delta\vec{\theta})$.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.
- Лагранжиан $L_i = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta} + \lambda(\theta_i + \vec{e}_i^T \Delta\vec{\theta})$.
- $\nabla_{\Delta\vec{\theta}} L_i = \Delta\vec{\theta}^T H + \lambda\vec{e}_i = 0$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.
- Лагранжиан $L_i = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta} + \lambda(\theta_i + \vec{e}_i^T \Delta\vec{\theta})$.
- $\nabla_{\Delta\vec{\theta}} L_i = \Delta\vec{\theta}^T H + \lambda\vec{e}_i = 0 \quad \Rightarrow \quad \Delta\vec{\theta} = -\lambda H^{-1}\vec{e}_i^T.$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.
- Лагранжиан $L_i = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta} + \lambda(\theta_i + \vec{e}_i^T \Delta\vec{\theta})$.
- $\nabla_{\Delta\vec{\theta}} L_i = \Delta\vec{\theta}^T H + \lambda\vec{e}_i = 0 \quad \Rightarrow \quad \Delta\vec{\theta} = -\lambda H^{-1}\vec{e}_i^T$.
- $\theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0 \quad \Rightarrow \quad \theta_i - \lambda\vec{e}_i^T H^{-1}\vec{e}_i = 0$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.
- Лагранжиан $L_i = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta} + \lambda(\theta_i + \vec{e}_i^T \Delta\vec{\theta})$.
- $\nabla_{\Delta\vec{\theta}} L_i = \Delta\vec{\theta}^T H + \lambda\vec{e}_i = 0 \quad \Rightarrow \quad \Delta\vec{\theta} = -\lambda H^{-1}\vec{e}_i^T.$
- $\theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0 \quad \Rightarrow \quad \theta_i - \lambda\vec{e}_i^T H^{-1}\vec{e}_i = 0 \quad \Rightarrow \quad \lambda = \frac{\theta_i}{\{H^{-1}\}_{ii}}.$
- $\Delta\vec{\theta} = -\frac{\theta_i}{\{H^{-1}\}_{ii}} H^{-1}\vec{e}_i^T;$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации $\mathcal{L}(\vec{\theta})$.

- В экстремуме $\mathcal{L}(\vec{\theta}+\Delta\vec{\theta}) - \mathcal{L}(\vec{\theta}) = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta}; \quad H_{ij} = \frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}.$
- Обнуление i-го коэффициента модели означает $\theta_i \to 0 \quad \Rightarrow \quad \theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0$, где $e_{ij} = \delta(i,j)$.
- Лагранжиан $L_i = \frac{1}{2}\Delta\vec{\theta}^T H \Delta\vec{\theta} + \lambda(\theta_i + \vec{e}_i^T \Delta\vec{\theta})$.
- $\nabla_{\Delta\vec{\theta}} L_i = \Delta\vec{\theta}^T H + \lambda\vec{e}_i = 0 \quad \Rightarrow \quad \Delta\vec{\theta} = -\lambda H^{-1}\vec{e}_i^T.$
- $\theta_i + \vec{e}_i^T \Delta\vec{\theta} = 0 \quad \Rightarrow \quad \theta_i - \lambda\vec{e}_i^T H^{-1}\vec{e}_i = 0 \quad \Rightarrow \quad \lambda = \frac{\theta_i}{\{H^{-1}\}_{ii}}.$
- $\Delta\vec{\theta} = -\frac{\theta_i}{\{H^{-1}\}_{ii}} H^{-1}\vec{e}_i^T; \qquad {}_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}}.$

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$L_i = \dfrac{\theta_i^2}{2\{H^{-1}\}_{ii}}$ – мера выпуклости функции $\mathcal{L}$ по параметру θ_i.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$$L_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}} - \text{мера выпуклости функции } \mathcal{L} \text{ по параметру } \theta_i.$$

- Настраивается модель регрессии, и подбираются все веса $\vec{\theta}$.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$$L_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}} - \text{мера выпуклости функции } \mathcal{L} \text{ по параметру } \theta_i.$$

- Настраивается модель регрессии, и подбираются все веса $\vec{\theta}$.
- Для каждого i находится минимальное L_i.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$$L_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}} - \text{мера выпуклости функции } \mathcal{L} \text{ по параметру } \theta_i.$$

- Настраивается модель регрессии, и подбираются все веса $\vec{\theta}$.
- Для каждого i находится минимальное L_i.
- Параметр θ_i, где $i = \arg\min_i L_i$, отсекается.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$$L_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}} - \text{мера выпуклости функции } \mathcal{L} \text{ по параметру } \theta_i.$$

- Настраивается модель регрессии, и подбираются все веса $\vec{\theta}$.
- Для каждого i находится минимальное L_i.
- Параметр θ_i, где $i = \arg\min_i L_i$, отсекается.
- Вектор параметров модели корректируется: $\vec{\theta} := \vec{\theta} - \frac{\theta_i}{\{H^{-1}\}_{ii}} H^{-1} \vec{e}_i^T$.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$$L_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}} - \text{мера выпуклости функции } \mathcal{L} \text{ по параметру } \theta_i.$$

- Настраивается модель регрессии, и подбираются все веса $\vec{\theta}$.
- Для каждого i находится минимальное L_i.
- Параметр θ_i, где $i = \arg\min_i L_i$, отсекается.
- Вектор параметров модели корректируется: $\vec{\theta} := \vec{\theta} - \frac{\theta_i}{\{H^{-1}\}_{ii}} H^{-1} \vec{e}_i^T$.
- Упрощенная модель не нуждается в обучении.

Оптимальное прореживание (optimal brain surgery)

Идея: отключать связи между нейронами, которые оказывают малое влияние на ошибку аппроксимации.

$$L_i = \frac{\theta_i^2}{2\{H^{-1}\}_{ii}} - \text{мера выпуклости функции } \mathcal{L} \text{ по параметру } \theta_i.$$

- Настраивается модель регрессии, и подбираются все веса $\vec{\theta}$.
- Для каждого i находится минимальное L_i.
- Параметр θ_i, где $i = \arg\min_i L_i$, отсекается.
- Вектор параметров модели корректируется: $\vec{\theta} := \vec{\theta} - \frac{\theta_i}{\{H^{-1}\}_{ii}} H^{-1} \vec{e}_i^T$.
- Упрощенная модель не нуждается в обучении.
- Можно повторить процедуру несколько раз, пока ошибка не превысит некоторого порогового знаяения.

Пакетная нормализация (batch normalization)

- Стохастический градиентный спуск (stochastic gradient descent) – реализация, в которой на каждой итерации алгоритма из обучающей выборки каким-то (случайным) образом выбирается только один объект.
- Пакетный градиентный спуск (batch gradient descent) – реализация градиентного спуска, когда на каждой итерации обучающая выборка просматривается целиком, и только после этого изменяются веса модели.
- “Золотая середина” – просматривается только некоторое подмножество обучающей выборки – мини-пакет (mini-batch) – фиксированного размера (batch-size).

Пакетная нормализация (batch normalization)

Идея: пронормировать вектор признаков, поступающий на вход некоторого слоя сети.

Пусть на вход подан mini-batch $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_b\} = \{x_{ij}\}$

$$\begin{cases} \mu_j = \mathcal{M}\{x_{ij}\} & \text{– математическое ожидание мини-пакета;} \\ \sigma_j^2 = \mathcal{M}\{(x_{ij} - \mu_j)^2\} & \text{– дисперсия мини-пакета;} \\ x_{\text{norm},ij} = \dfrac{x_{ij} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}} & \text{– нормализованные объекты;} \\ y_{ij} = \gamma_j x_{\text{norm},ij} + \beta_j & \text{– сжатие и сдвиг; } \beta_j \text{ и } \gamma_j \text{ – обучаемые параметры.} \end{cases}$$

Если положить $\beta_j = \mu_j$ и $\gamma_j = \sqrt{\sigma_j^2 + \epsilon}$,
то это тождественное преобразование: $y_{ij} = x_{ij}$.

Начальная инициализация параметров

- $\theta_{h,ij} = 0$
  Все нейроны одинаковы, что плохо.
- $\theta_{h,ij}$ – небольшие случайные величины.
  Дисперсия растет как корень из числа нейронов. Хорошо работает только на небольших архитектурах.
- $\theta_{h,ij} = \text{uniform}(-1/\sqrt{n}; 1/\sqrt{n})$
  Выравнивание дисперсий в разных слоях.
- Послойное обучение нейронов, как линейных моделей на случайной подвыборке объектов или признаков.
- Случайный ортогональный базис.

Лекция 14. Нейронные сети для обработки изображений

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Распознавание рукописных цифр (база данных MNIST)

Классический подход – придумывание признаков

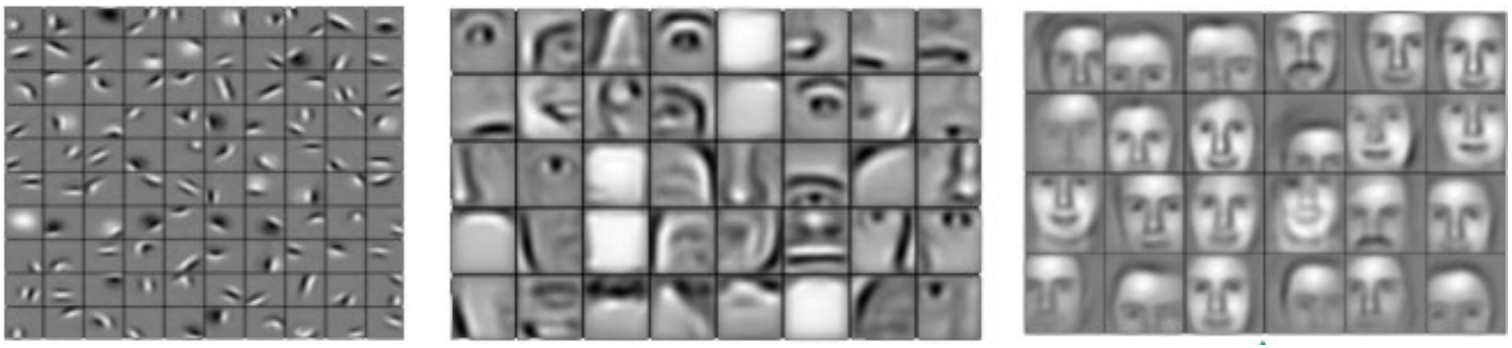
Распознавание рукописных цифр (база данных MNIST)

Классический подход – придумывание признаков

- Центральная симметрия
- Симметрия по горизонтали
- Симметрия по вертикали
- Число областей
- “Жирность” символа
- Каждый пиксель изображения

Современный подход – построение признаков

Детали изображения $\longrightarrow$ Фрагменты $\longrightarrow$ Объекты



Сверточные нейронные сети

Предыстория

A bit of history:

Hubel & Wiesel,

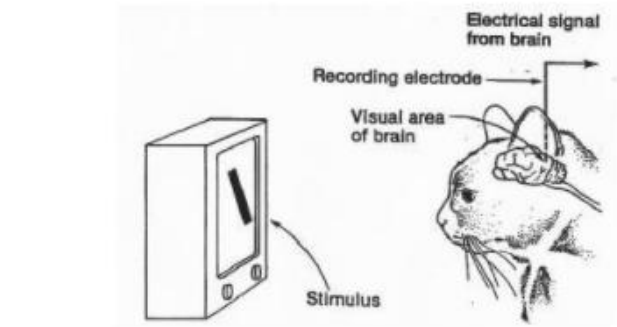
1959

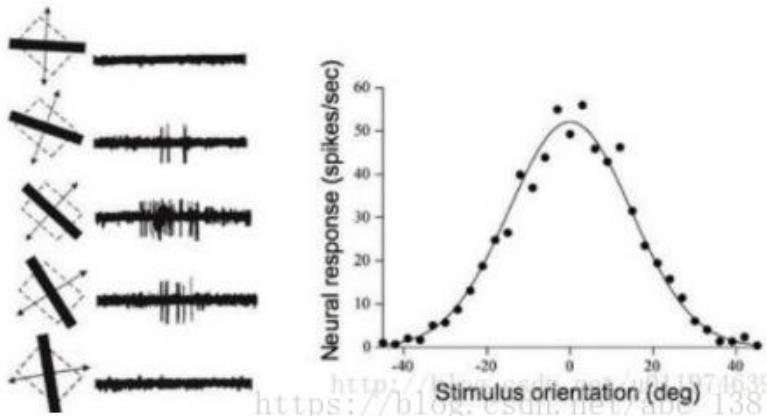
RECEPTIVE FIELDS OF SINGLE NEURONES IN THE CAT'S STRIATE CORTEX

1962

RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT'S VISUAL CORTEX

1968...





- Простые клетки реагирут на прямые линии под разными углами.
- Реакция сложных клеток связана с активацией определённого набора простых клеток.

Сверточные нейронные сети

Операция свертки

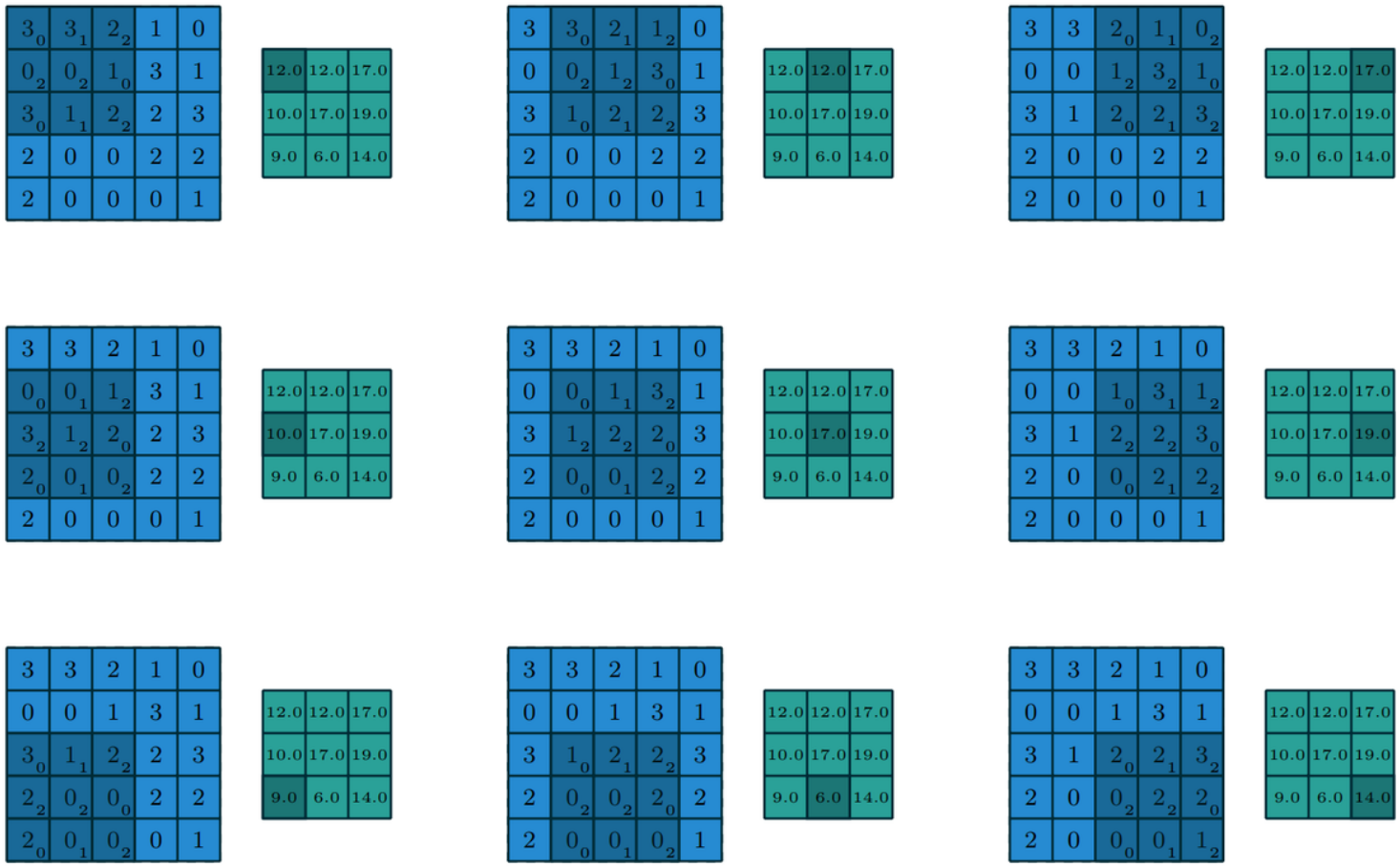
X_{ij} – исходные данные – пиксели изображения;
θ_{ab} – ядро свертки; $-A \leq a \leq A$; $-B \leq b \leq B$.

$$(\mathbf{X} * \boldsymbol{\theta})_{ij} = \sum_{a=-A}^{A} \sum_{b=-B}^{B} \theta_{ab} X_{i+a,j+b}$$

Сверточные нейронные сети

Операция свертки

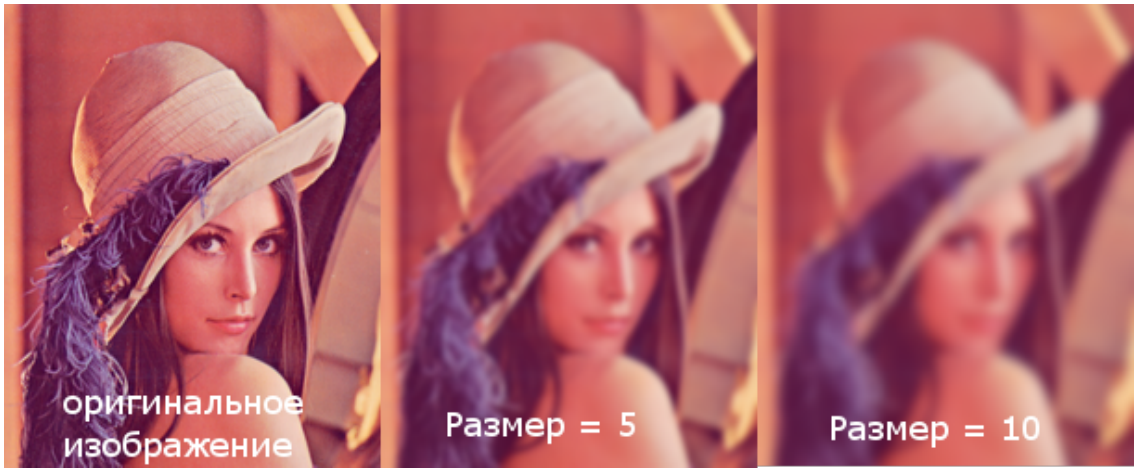
Пример свертки матриц 5×5 и 3×3



https://arxiv.org/abs/1603.07285

Сверточные нейронные сети

Фильтрующие матрицы



0,000789	0,006581	0,013347	0,006581	0,000789
0,006581	0,054901	0,111345	0,054901	0,006581
0,013347	0,111345	0,225821	0,111345	0,013347
0,006581	0,054901	0,111345	0,054901	0,006581
0,000789	0,006581	0,013347	0,006581	0,000789

https://habr.com/ru/post/142818/

Сверточные нейронные сети

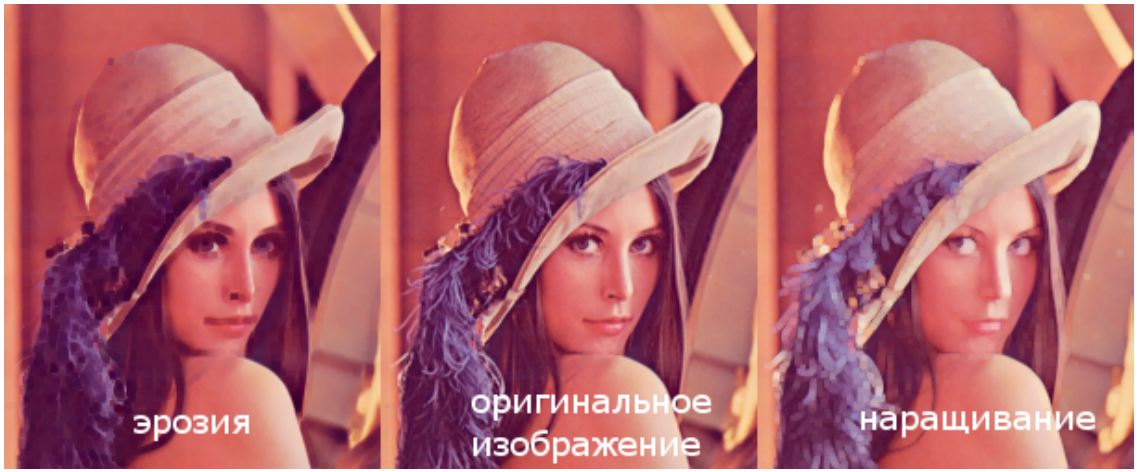
Фильтрующие матрицы

-1	-1	-1
-1	9	-1
-1	-1	-1

https://habr.com/ru/post/142818/

Сверточные нейронные сети

Фильтрующие матрицы



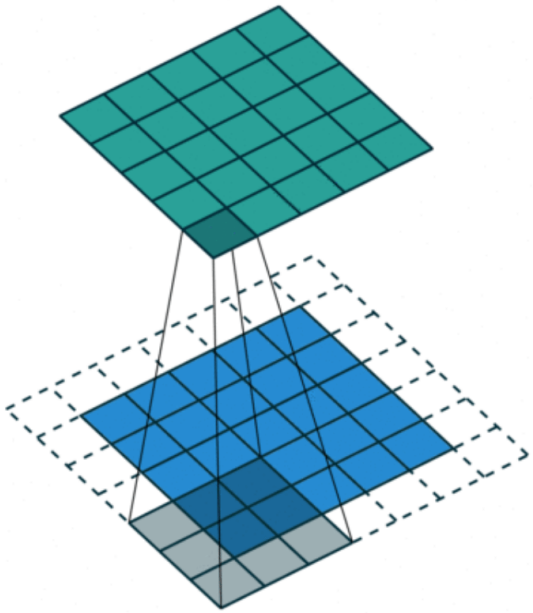
0	0	1	0	0
0	1	1	1	0
1	1	1	1	1
0	1	1	1	0
0	0	1	0	0

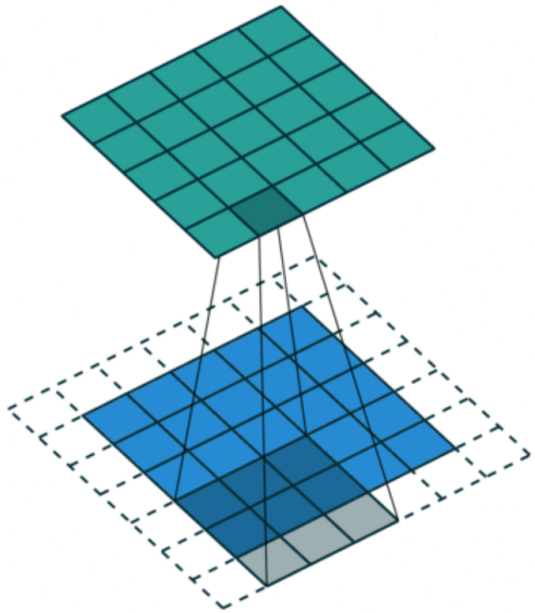
https://habr.com/ru/post/142818/

Сверточные нейронные сети

Фильтрующие матрицы

Можно сделать отступ (padding)





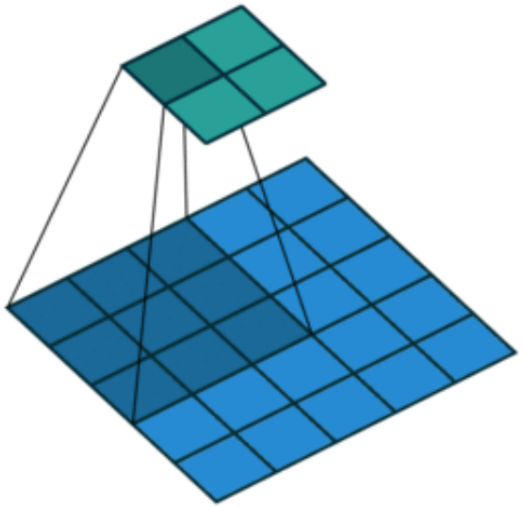
Stride=1; padding=1.

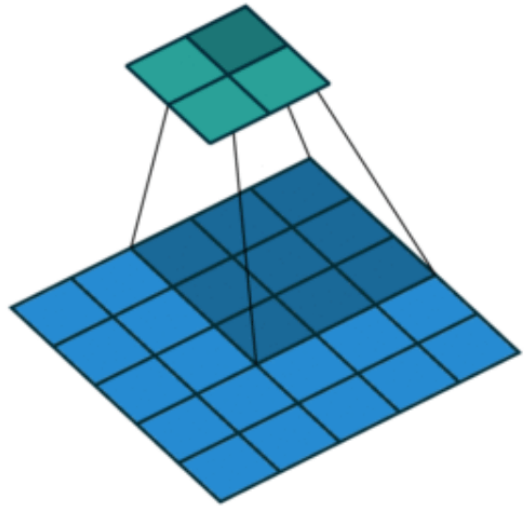
https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d

Сверточные нейронные сети

Сверточные слои (convolution layer)

Можно задать шаг (stride)





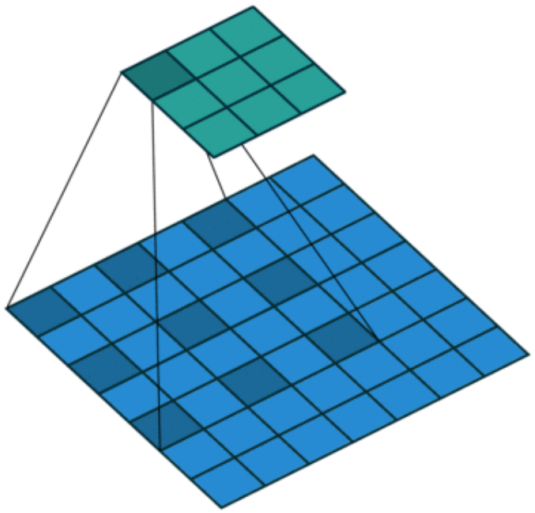
Stride=2; padding=0.

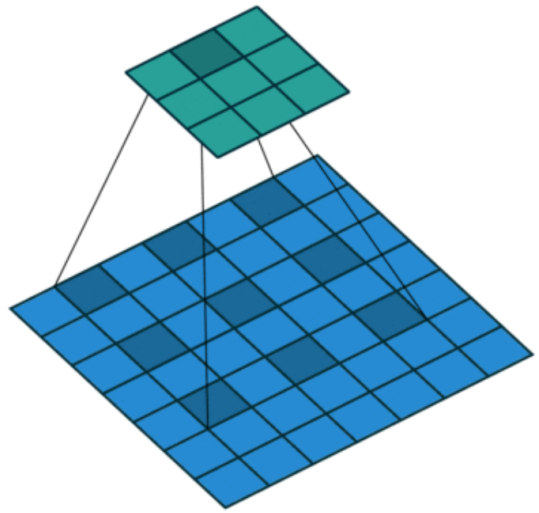
https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d

Сверточные нейронные сети

Сверточные слои (convolution layer)

Можно сделать прореживание (dilation rate)





Dilation rate=2.

https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d

Сверточные нейронные сети

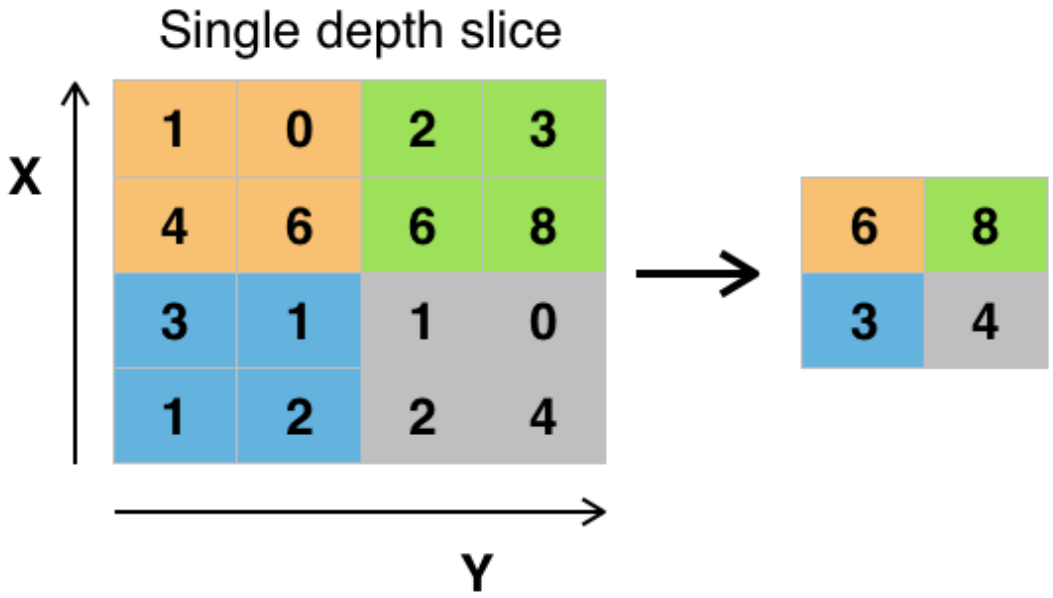
Объединяющие слои (pooling layer, ранее – subsampling layer)

- $y_{ij} = F_a(x_{hi,hj}, \ldots, x_{hi+h-1,jh+h-1})$;
- $F_a = \max(\cdot)$ – max pooling;
- $F_a = \text{avg}(\cdot)$ – average pooling.

Сверточные нейронные сети

Объединяющие слои (pooling layer, ранее – subsampling layer)

- $y_{ij} = F_a(x_{hi,hj}, \ldots, x_{hi+h-1,jh+h-1})$;
- $F_a = \max(\cdot)$ – max pooling;
- $F_a = \text{avg}(\cdot)$ – average pooling.

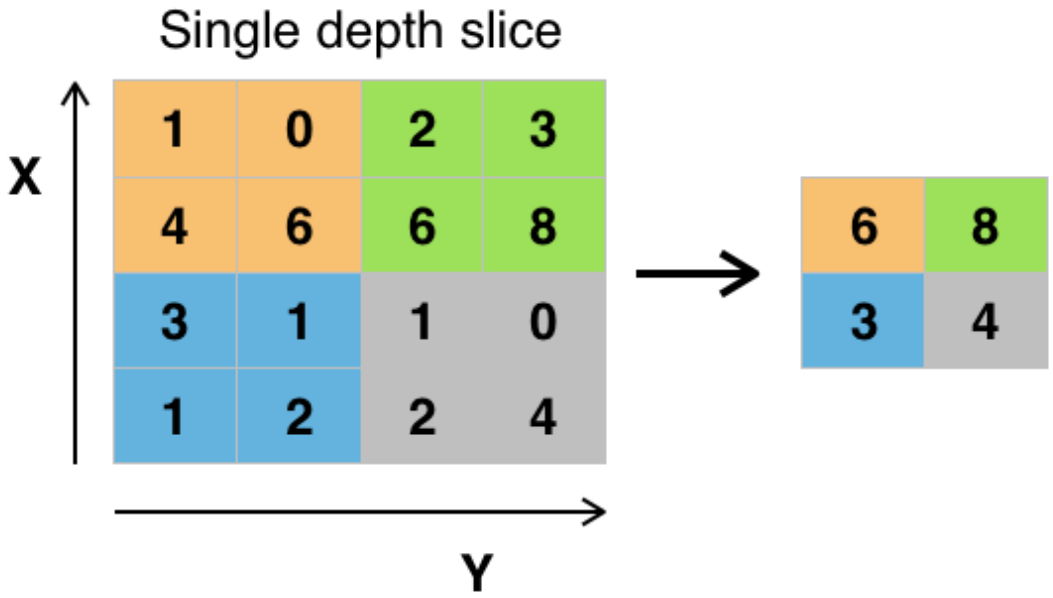


https://en.wikipedia.org/wiki/Convolutional_neural_network

Сверточные нейронные сети

Объединяющие слои (pooling layer, ранее – subsampling layer)

- $y_{ij} = F_a(x_{hi,hj}, \ldots, x_{hi+h-1,jh+h-1})$;
- $F_a = \max(\cdot)$ – max pooling;
- $F_a = \text{avg}(\cdot)$ – average pooling.

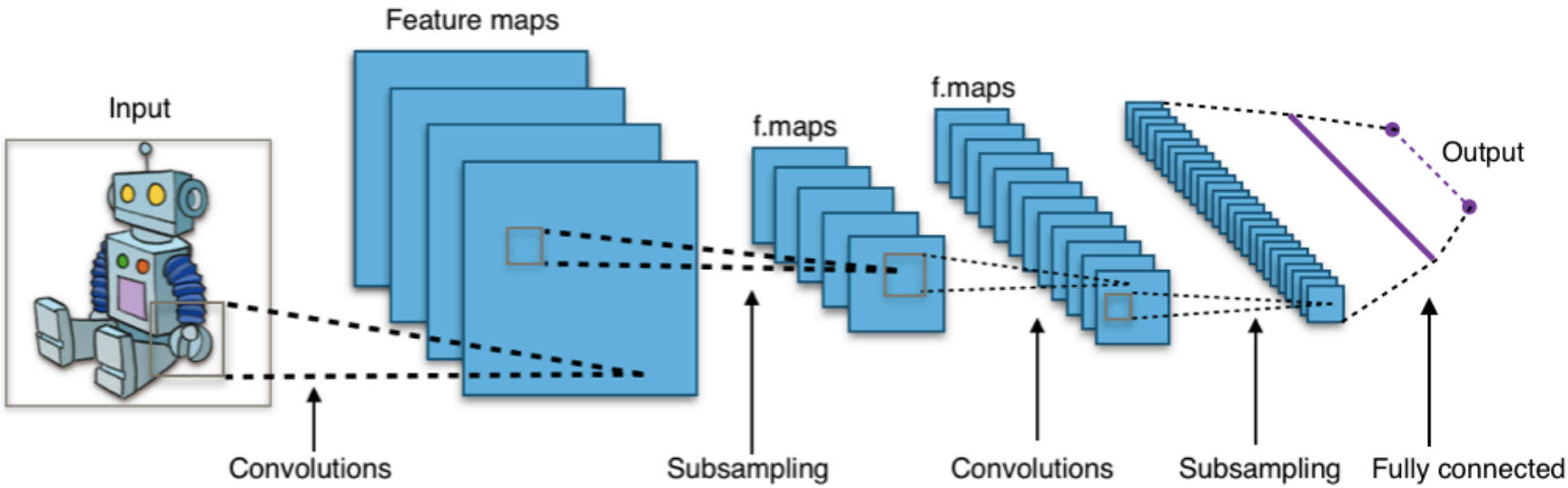


https://en.wikipedia.org/wiki/Convolutional_neural_network

Max pooling позволяет распознавать объект вне зависимости от его положения на картинке.

Сверточные нейронные сети

Типичная архитектура сверточной сети

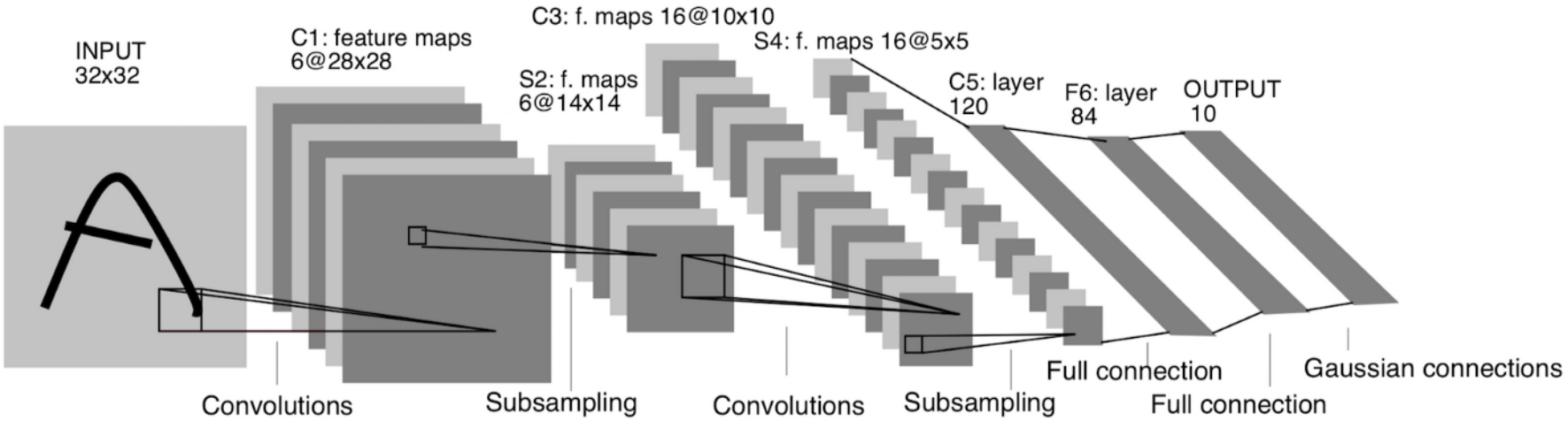


https://en.wikipedia.org/wiki/Convolutional_neural_network

Сверточные нейронные сети

LeNet (LeCun, Yann; Léon Bottou; Yoshua Bengio; Patrick Haffner, 1998)

LeNet – сеть для классификации MNIST



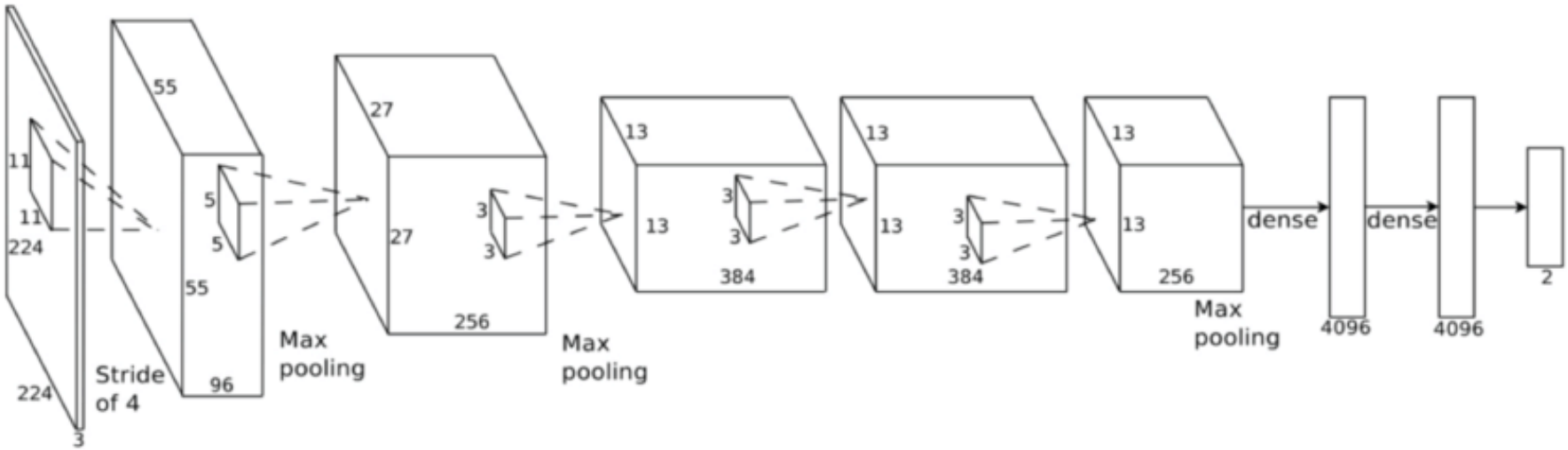
http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

Сверточные нейронные сети

AlexNet (Alex Krizhevsky; Ilya Sutskever; Geoffrey E. Hinton, 2012)

AlexNet– победитель соревнования ImageNet LSVRC-2012.

Использование ReLU и dropout в сочетании с обучением на GPU позволило обучить сеть с 62.3 млн параметрами.



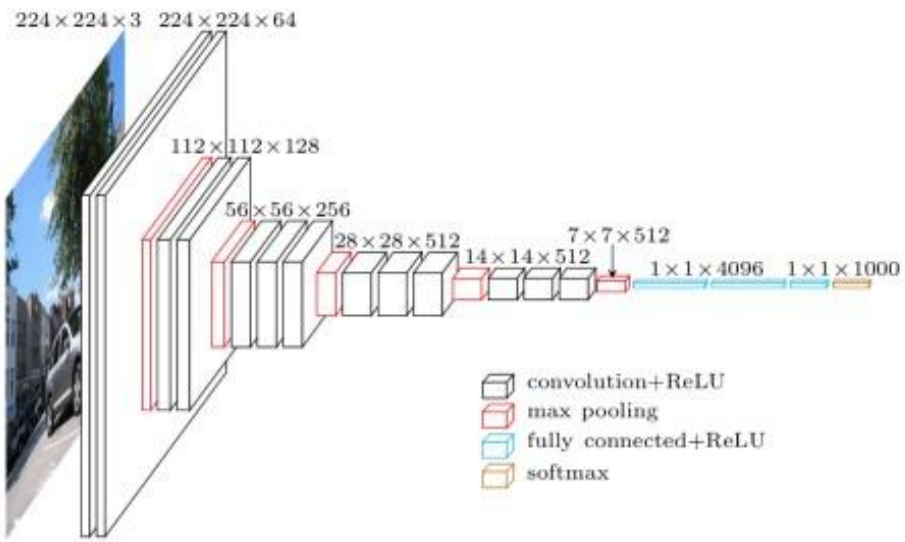
https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Сверточные нейронные сети

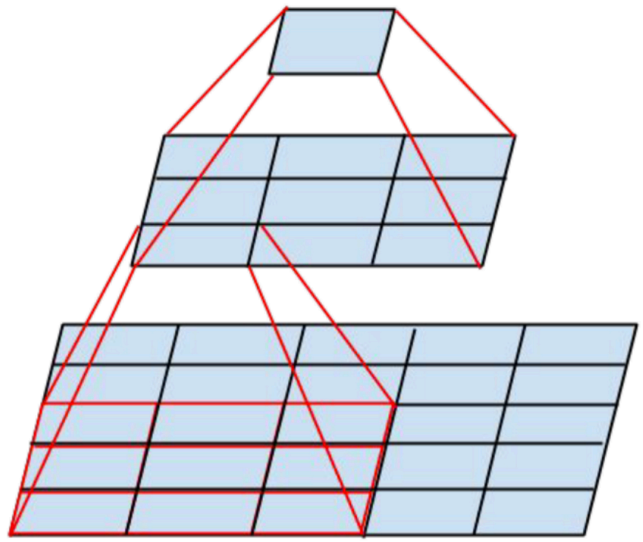
VGG (Alex Krizhevsky; Ilya Sutskever; Geoffrey E. Hinton, 2012)

VGG – второе место на соревновании ImageNet LSVRC-2014.

Идея: заменить большие свертки на многослойные свертки 3×3.
Свертка 5×5 (25 параметров) $\rightarrow$ 2 свертки 3×3 (18 параметров).



https://arxiv.org/abs/1409.1556v6

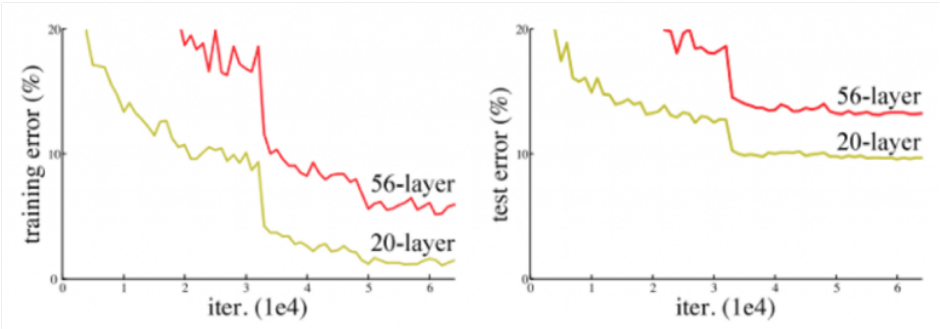


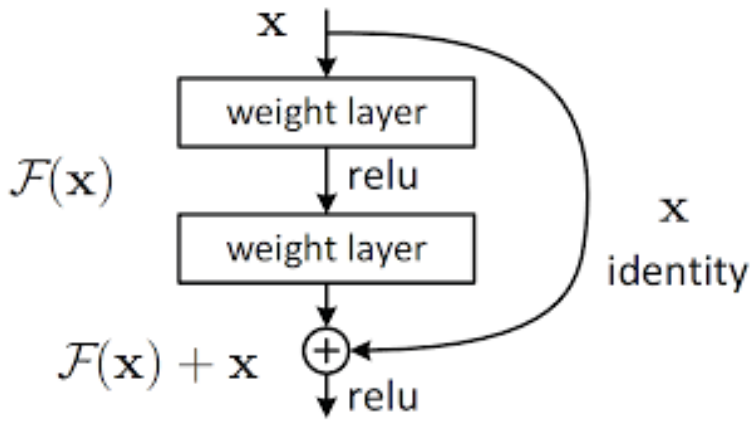
https://habr.com/ru/company/vk/blog/311706/

Остаточные нейронные сети (Residual Networks)

ResNet-152 – победитель на соревновании ImageNet LSVRC-2015.

Идея: чтобы избавиться от затухания градиента в глубокой нейронний сети, производится обучение “остаткам”: $f(x) + x$ вместо $f(x)$.

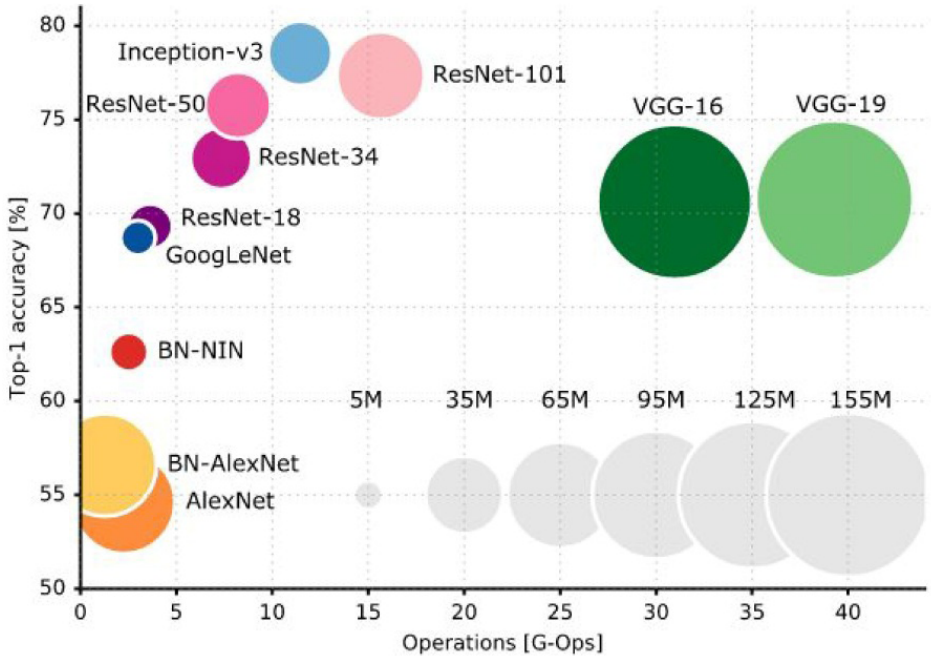




https://arxiv.org/pdf/1512.03385.pdf

Обзор нейронных сетей для обработки изображений

Точность растет, но число параметров падает – архитектура сетей изменяется по-умному!



https://culurciello.github.io/tech/2016/06/04/nets.html

Возможности для улучшения процесса обучения

- Не только уменьшение, но и увеличение lr.

Возможности для улучшения процесса обучения

- Не только уменьшение, но и увеличение lr.
- Аугментация данных для повышения разнообразия при обучении.

Возможности для улучшения процесса обучения

- Не только уменьшение, но и увеличение lr.
- Аугментация данных для повышения разнообразия при обучении.
- Аугментация данных для большей устойчивости на тесте.

Возможности для улучшения процесса обучения

- Не только уменьшение, но и увеличение lr.
- Аугментация данных для повышения разнообразия при обучении.
- Аугментация данных для большей устойчивости на тесте.
- Transfer learning

Нейронные сети для распознавания цифр

Jupyter notebook “Нейронные сети для распознавания цифр”: `https://colab.research.google.com/drive/1wq-_4AZoAyPecesseJd7d1rORsNW7dJi`

Jupyter notebook “Нейронные сети для распознавания цифр: аугментация”: `https://colab.research.google.com/drive/1_6pQfteUW2uJAPuNts4nCkFj4iD-2Qxb`

Лекция 15. Нейронные сети в задачах обучения без учителя

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

Обучение без учителя

Задачи, которые решаются без привлечения учителя:

- Кластеризация – разбиение выборки на непересекающиеся множества (кластеры) похожих объектов так, что объекты разных кластеров сильно отличаются.
- Обнаружение аномалий – выделение данных, сильно отличающихся от типичных.
- Сокращение размерности – представление исходных данных с большим количеством признаков в пространстве меньшей размерности с минимальными потерями информации.
- Визуализация – преобразование данных для наглядного изображения их на плоскости.

Схема постановки задачи кластеризации

- Имеется выборка $\mathbb{X}$ из L объектов,
  - каждый из которых задан вектором признаков $\vec{x}$,
  - или определена функция расстояния между объектами $\rho(\vec{x}_i, \vec{x}_j)$.
- Требуется определить для каждого объекта его кластер $y(\vec{x})$ так, чтобы объекты в одном кластере оказались похожими, а объекты в разных кластерах – отличались.

Схема постановки задачи кластеризации

- Имеется выборка $\mathbb{X}$ из L объектов,
  - каждый из которых задан вектором признаков $\vec{x}$,
  - или определена функция расстояния между объектами $\rho(\vec{x}_i, \vec{x}_j)$.
- Требуется определить для каждого объекта его кластер $y(\vec{x})$ так, чтобы объекты в одном кластере оказались похожими, а объекты в разных кластерах – отличались.
- Существует много разных постановок задачи для определения “сходства” и “различия” объектов.

Схема постановки задачи кластеризации

- Имеется выборка $\mathbb{X}$ из L объектов.
- Определена функция расстояния $\rho(\vec{x}_i, \vec{x}_j)$.
- Задано число кластеров.

Схема постановки задачи кластеризации

- Имеется выборка $\mathbb{X}$ из L объектов.
- Определена функция расстояния $\rho(\vec{x}_i, \vec{x}_j)$.
- Задано число кластеров.
- Требуется определить центры кластеров $\vec{\theta}_y$ и отнести каждый объект к кластеру с ближайшим центром (принцип жесткой конкуренции):

$$a(\vec{x}) = \arg\min_y \rho(\vec{x}, \vec{\theta}_y)$$

.

- Средний квадрат внутрикластерного расстояния минимален:

$$Q(\vec{\theta}) = \frac{1}{L}\sum_{i=1}^{L} \rho^2\left(\vec{x}_i, \vec{\theta}_{a(\vec{x}_i)}\right) \to \min$$

Разделительная кластеризация (k-means)

Принцип жесткой конкуренции

- Пусть векторы $\vec{x}$ и $\vec{\theta}$ – нормированные.
- Минимум расстояния в евклидовой метрике: $\rho(\vec{x}, \vec{\theta}) = (\vec{x} - \vec{\theta})^2$.
- Двуслойная нейронная сеть:
  - 1 слой – вычисление расстояний $\rho(\vec{x}, \vec{\theta})$;
  - 2 слой – $a(\vec{x}) = \arg\max\limits_{y} (\vec{x}, \vec{\theta}_y)$.
- Обучение методом градиентного спуска:
  $\vec{\theta}_y := \vec{\theta}_y + lr \cdot (\vec{x}_i - \vec{\theta}_y)[a(\vec{x}_i) = y]$

Разделительная кластеризация (k-means)

Принцип жесткой конкуренции

- Пусть векторы $\vec{x}$ и $\vec{\theta}$ – нормированные.
- Минимум расстояния в евклидовой метрике: $\rho(\vec{x}, \vec{\theta}) = (\vec{x} - \vec{\theta})^2$.
- Двуслойная нейронная сеть:
  - 1 слой – вычисление расстояний $\rho(\vec{x}, \vec{\theta})$;
  - 2 слой – $a(\vec{x}) = \arg\max\limits_{y} (\vec{x}, \vec{\theta}_y)$.
- Обучение методом градиентного спуска:
  $\vec{\theta}_y := \vec{\theta}_y + lr \cdot (\vec{x}_i - \vec{\theta}_y)[a(\vec{x}_i) = y]$
- На каждом шаге центр кластера сдвигается ближе к элементу $\vec{x}_i$ из этого кластера.

Разделительная кластеризация (k-means)

Принцип жесткой конкуренции

- Пусть векторы $\vec{x}$ и $\vec{\theta}$ – нормированные.
- Минимум расстояния в евклидовой метрике: $\rho(\vec{x}, \vec{\theta}) = (\vec{x} - \vec{\theta})^2$.
- Двуслойная нейронная сеть:
  - 1 слой – вычисление расстояний $\rho(\vec{x}, \vec{\theta})$;
  - 2 слой – $a(\vec{x}) = \arg\max\limits_{y} (\vec{x}, \vec{\theta}_y)$.
- Обучение методом градиентного спуска:
  $\vec{\theta}_y := \vec{\theta}_y + lr \cdot (\vec{x}_i - \vec{\theta}_y)[a(\vec{x}_i) = y]$
- На каждом шаге центр кластера сдвигается ближе к элементу $\vec{x}_i$ из этого кластера.
- Недостатки: обучение может происходить медленно, а некоторые кластеры могут не обновляться, т.к. в них не попали объекты.

Разделительная кластеризация (k-means)

Принцип мягкой конкуренции

Идея: ввести функцию ядра $K(\rho)$ и проводить обучение по центрам всех кластеров, а не только одного ближайшего.

Разделительная кластеризация (k-means)

Принцип мягкой конкуренции

Идея: ввести функцию ядра $K(\rho)$ и проводить обучение по центрам всех кластеров, а не только одного ближайшего.

- Пусть векторы $\vec{x}$ и $\vec{\theta}$ – нормированные.
- Минимум расстояния в евклидовой метрике: $\rho(\vec{x}, \vec{\theta}) = (\vec{x} - \vec{\theta})^2$.
- Двуслойная нейронная сеть:
  - 1 слой – вычисление расстояний $\rho(\vec{x}, \vec{\theta})$;
  - 2 слой – степени близости объекта к каждому из кластеров $a_y(\vec{x}) = K(\rho(\vec{x}, \vec{\theta}_y))$.
- Обучение методом градиентного спуска: $\vec{\theta}_y := \vec{\theta}_y + lr \cdot (\vec{x}_i - \vec{\theta}_y)K(\rho(\vec{x}_i, \vec{\theta}_y))$
- На каждом шаге центры всех кластеров сдвигаются ближе к элементу $\vec{x}_i$ в зависимости от удаления до него.

G-means

Недостаток подхода k-means – Необходимость задать число кластеров.

G-means

Недостаток подхода k-means – Необходимость задать число кластеров.

Идея: предположим, что кластеризуемые данные подчиняются некоторому унимодальному закону распределения, например гауссовскому.

G-means

Недостаток подхода k-means – Необходимость задать число кластеров.

Идея: предположим, что кластеризуемые данные подчиняются некоторому унимодальному закону распределения, например гауссовскому.

- Если исходные данные описываются унимодальным гауссовским распределением с заданными средним, то можно предположить, что все они относятся к одному кластеру.
- Если распределение данных не гауссовское, то выполняется разделение на два кластера. Если в этих кластерах распределения окажутся близки к гауссовскому, то число кластеров считается оптимальным.
- В противном случае число кластеров увеличивается, и процесс повторяется.

G-means

Недостаток подхода k-means – Необходимость задать число кластеров.

G-means

Недостаток подхода k-means – Необходимость задать число кластеров.

Идея: предположим, что кластеризуемые данные подчиняются некоторому унимодальному закону распределения, например гауссовскому.

G-means

Недостаток подхода k-means – Необходимость задать число кластеров.

Идея: предположим, что кластеризуемые данные подчиняются некоторому унимодальному закону распределения, например гауссовскому.

- Если исходные данные описываются унимодальным гауссовским распределением с заданными средним, то можно предположить, что все они относятся к одному кластеру.
- Если распределение данных не гауссовское, то выполняется разделение на два кластера. Если в этих кластерах распределения окажутся близки к гауссовскому, то число кластеров считается оптимальным.
- В противном случае число кластеров увеличивается, и процесс повторяется.

Схема постановки задачи сокращения размерности

Автокодировщик (autoencoder)

- Имеется выборка $\mathbb{X}$ из L объектов.
- Задается функционал качества $\mathcal{L}(\vec{a}, \vec{x})$, характеризующий, насколько хорошо алгоритм $\vec{a}$ аппроксимирует исходный объект $\vec{x}$.
- Выбирается параметрическая модель $\vec{a}(\vec{x}, \vec{\alpha}, \vec{\beta}) = g(f(\vec{x}, \vec{\alpha}), \vec{\beta})$
  - Функция кодировщика $f : \mathbb{X} \to \mathbb{Y}$.
  - Функция декодировщика $g : \mathbb{Y} \to \mathbb{X}$.
  - Пространство $\mathbb{Y}$ обладает полезными свойствами, например, имеет меньшую размерность по сравнению с $\mathbb{X}$.
- Параметры модели $\vec{\alpha}, \vec{\beta}$ оптимизируются для достижения минимума
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума

$$Q_{\text{AE}}(\vec{\alpha}, \vec{\beta}) = \frac{1}{L} \sum_{i=1}^{L} \mathcal{L}(g(f(\vec{x}, \vec{\alpha}), \vec{\beta}), \vec{x}).$$

Автокодировщики

Линейный автокодировщик

- Кодировщик $f(\vec{x}, \mathbf{A}) = \mathbf{A}_{[m \times n]}\vec{x}; \quad \vec{x} \in \mathcal{R}^n.$
- Декодировщик $g(\vec{y}, \mathbf{B}) = \mathbf{B}_{[n \times m]}\vec{y}; \quad \vec{y} \in \mathcal{R}^m.$
- Квадратичная функция потерь: $\mathcal{L}(\vec{a}, \vec{x}) = \|\vec{a} - \vec{x}\|^2.$
- Задача линейного автокодировщика:

$$Q_{\text{AE}}(\mathbf{A}, \mathbf{B}) = \frac{1}{L}\sum_{i=1}^{L} \|\mathbf{BA}\vec{x}_i - \vec{x}_i\|^2 \to \min$$

Автокодировщики

Метод главных компонент

- Для матрицы $\mathcal{X}_{[L\times n]}$ строится матрица $(\mathcal{X}^T\mathcal{X})_{[n\times n]}$
- У нее отбираются m наибольших собственных значений $\lambda_1,\ldots,\lambda_m$
- Соответствующие собственные векторы формируют матрицу $U_{[n\times m]}$
- U – матрица перехода между n-мерным и m-мерным признаковыми пространствами. Новые признаки вычисляются как $G_{[L\times m]} = \mathcal{X}U$
- Матрица U ортонормирована: $U^TU = I_{[m\times m]}$
- $\mathcal{X} \approx GU^T$

Автокодировщики

Линейный автокодировщик – обобщение метода главных компонент

Задача линейного автокодировщика:

$$Q_{\text{AE}}(\mathbf{A}, \mathbf{B}) = \frac{1}{L} \sum_{i=1}^{L} \left\| \mathbf{B}\mathbf{A}\vec{x}_i - \vec{x}_i \right\|^2 \to \min_{\mathbf{A},\mathbf{B}}$$

Задача метода главных компонент:

$$Q_{\text{PCA}}(\mathbf{U}) = \frac{1}{L} \left\| \mathbf{G}\mathbf{U}^T - \mathcal{X} \right\|^2 = \frac{1}{L} \sum_{i=1}^{L} \left\| \mathbf{U}\mathbf{U}^T\vec{x}_i - \vec{x}_i \right\|^2 \to \min_{\mathbf{U}}$$

Автокодировщики

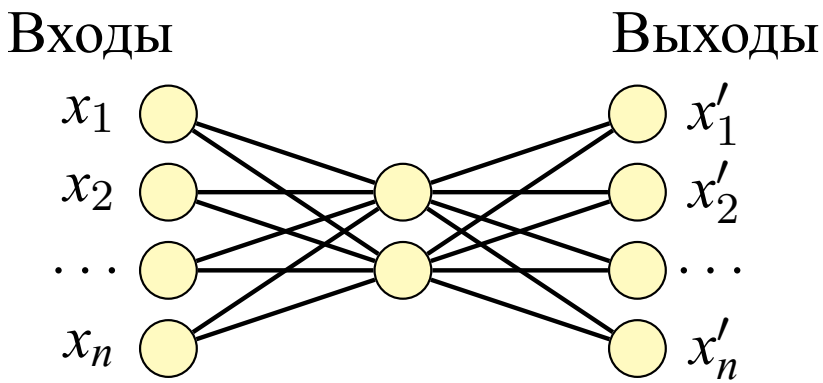
Двуслойная нейросеть

- Кодировщик $\vec{f}(\vec{x}, \mathbf{A}) = \vec{\sigma}_f(\mathbf{A}\vec{x}); \quad \vec{x} \in \mathcal{R}^n;$
- Декодировщик $\vec{g}(\vec{y}, \mathbf{B}) = \vec{\sigma}_g(\mathbf{B}\vec{y}); \quad \vec{y} \in \mathcal{R}^m;$
- $\vec{\sigma}_f, \vec{\sigma}_g$ – функции активации слоев нейросети.

$$Q_{\text{AE}}(\mathbf{A}, \mathbf{B}) = \frac{1}{L}(\vec{\sigma}_g(\mathbf{B}\vec{\sigma}_f(\mathbf{A}\vec{x}_i))\vec{x}_i - \vec{x}_i)^2 \to \min_{\mathbf{A},\mathbf{B}}$$

Автокодировщики

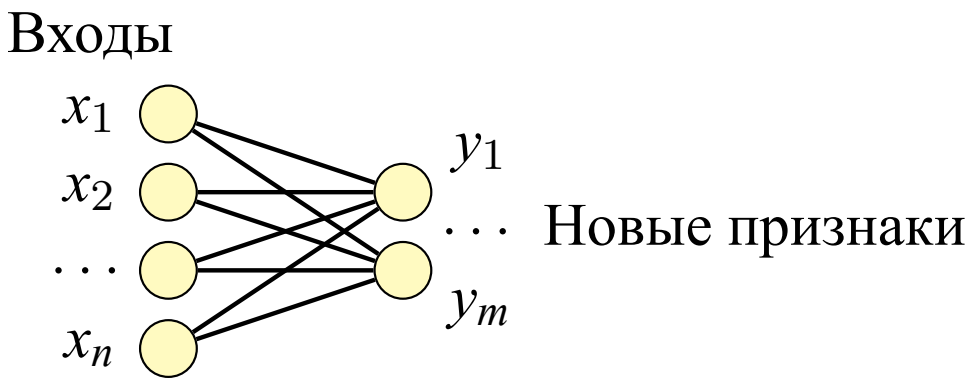
Применение автокодировщиков



- Сжатие данных при их хранении или передаче.

Автокодировщики

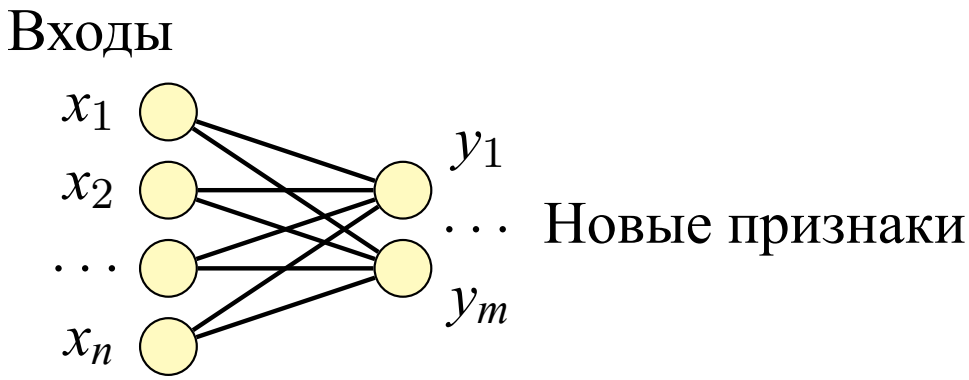
Применение автокодировщиков



- Сжатие данных при их хранении или передаче.
- Генерация признаков.
- Снижение размерности.

Автокодировщики

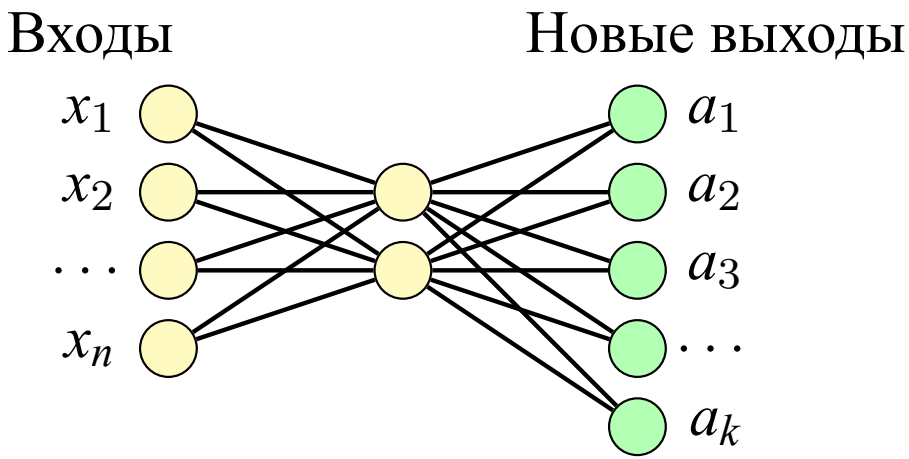
Применение автокодировщиков



- Сжатие данных при их хранении или передаче.
- Генерация признаков.
- Снижение размерности.
- Векторизация объектов.

Автокодировщики

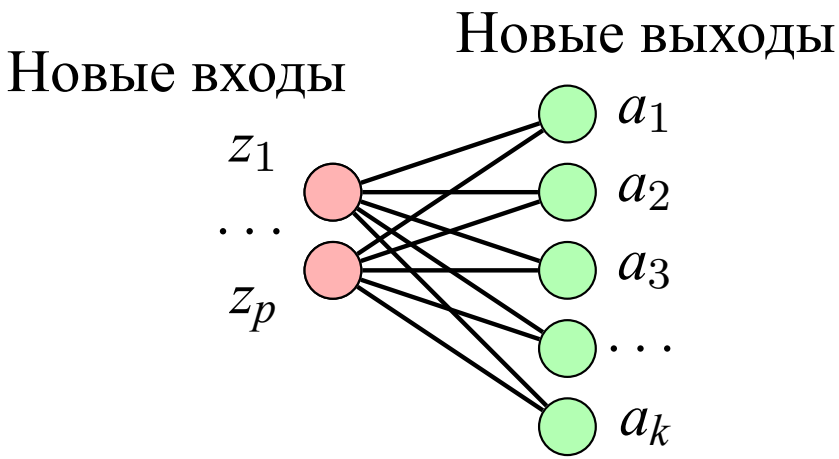
Применение автокодировщиков



- Сжатие данных при их хранении или передаче.
- Генерация признаков.
- Снижение размерности.
- Векторизация объектов.
- Подготовка для решения задач обучения с учителем.

Автокодировщики

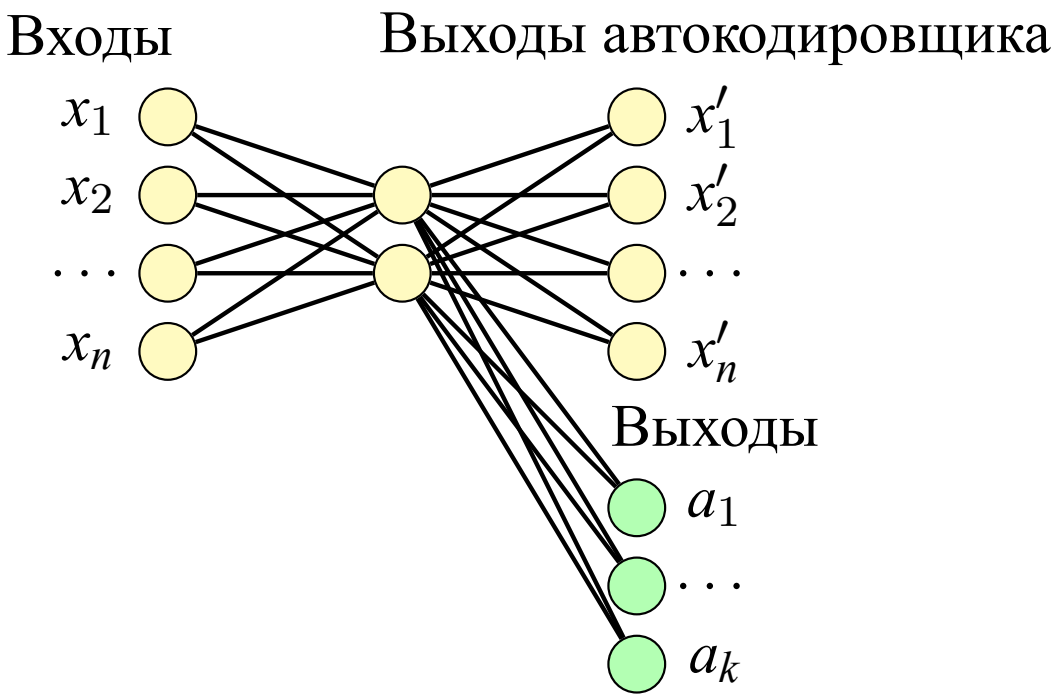
Применение автокодировщиков



- Сжатие данных при их хранении или передаче.
- Генерация признаков.
- Снижение размерности.
- Векторизация объектов.
- Подготовка для решения задач обучения с учителем.
- Генерация новых объектов.

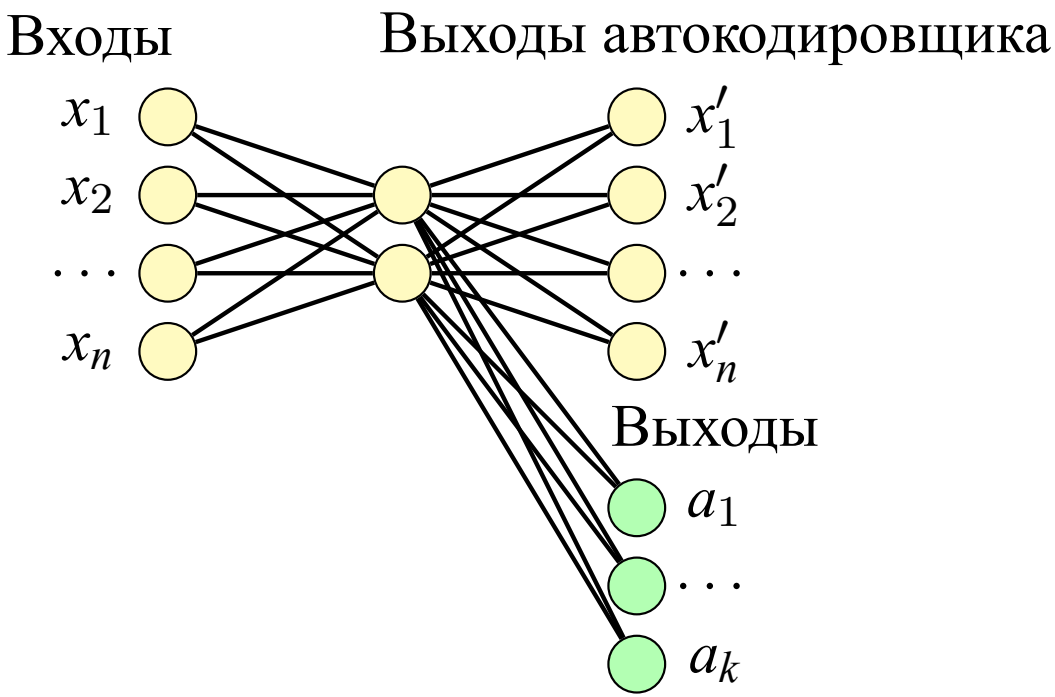
Автокодировщики

Применение автокодировщиков в задачах обучения с учителем



Автокодировщики

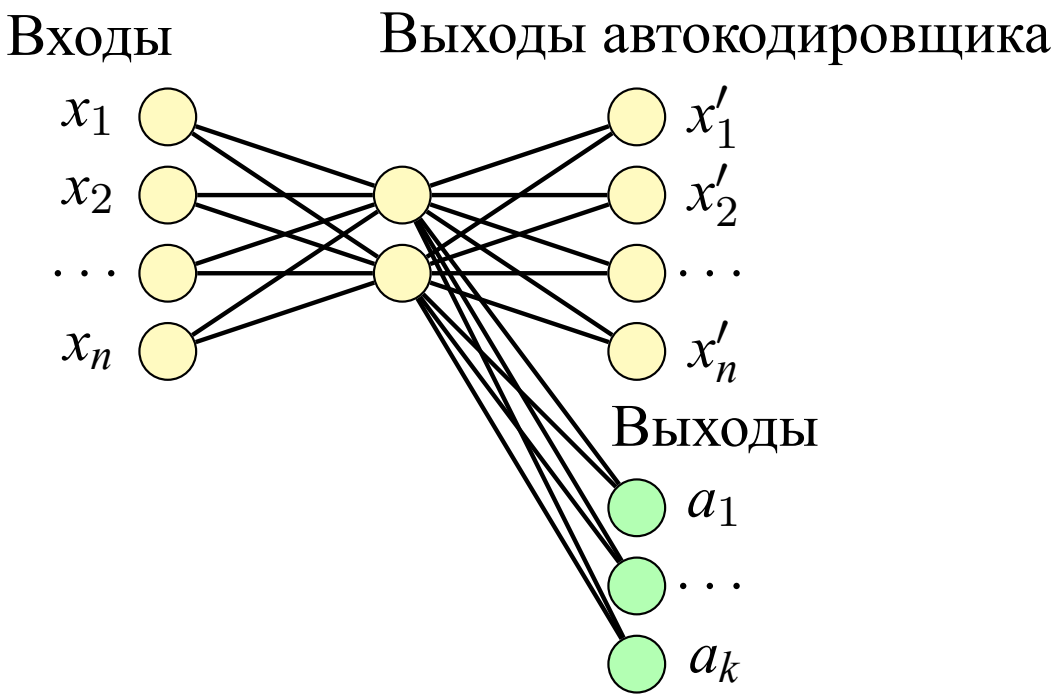
Применение автокодировщиков в задачах обучения с учителем



- Данные “от учителя” дополняют автокодировщик.

Автокодировщики

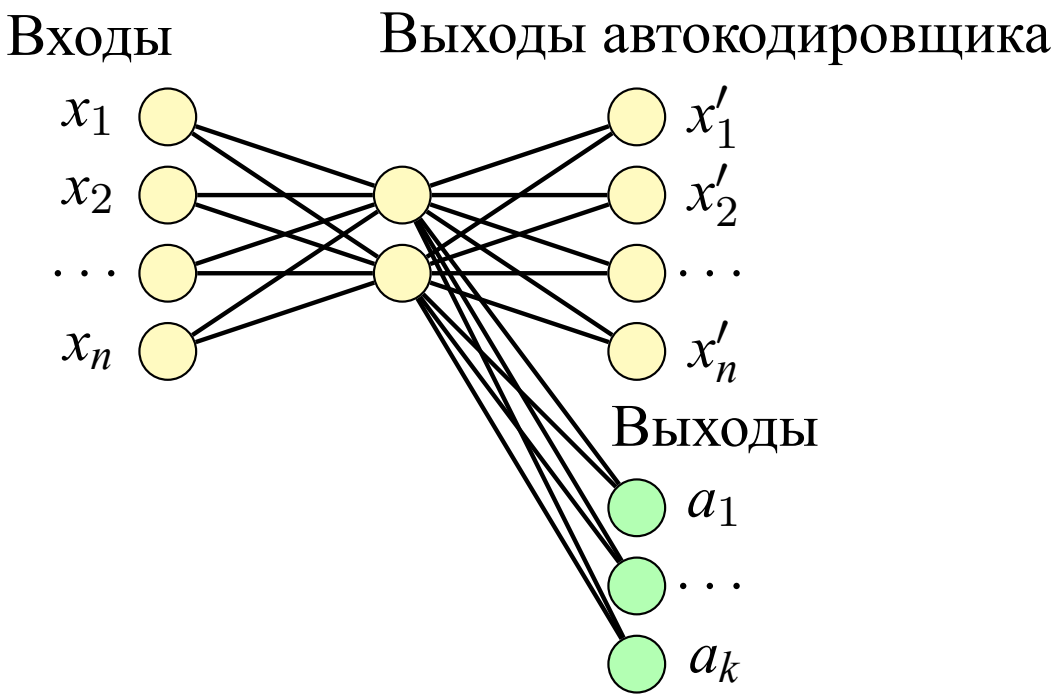
Применение автокодировщиков в задачах обучения с учителем



- Данные “от учителя” дополняют автокодировщик.
- “Обычная” нейросеть – это “скрытый автокодировщик”.

Автокодировщики

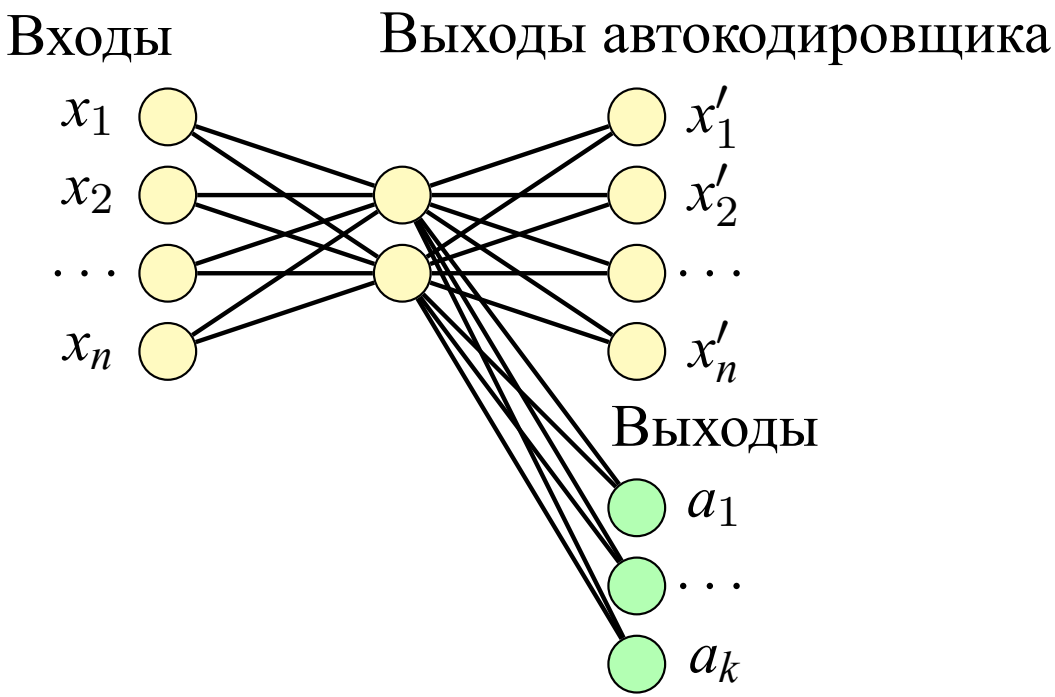
Применение автокодировщиков в задачах обучения с учителем



- Данные “от учителя” дополняют автокодировщик.
- “Обычная” нейросеть – это “скрытый автокодировщик”.
- Transfer learning – обучение “скрытого автокодировщика”.

Автокодировщики

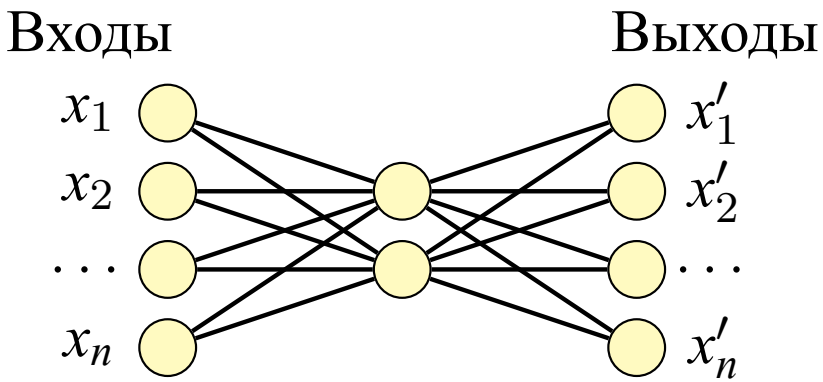
Применение автокодировщиков в задачах обучения с учителем



- Данные “от учителя” дополняют автокодировщик.
- “Обычная” нейросеть – это “скрытый автокодировщик”.
- Transfer learning – обучение “скрытого автокодировщика”.
- Многозадачное обучение (multi-task learning) – обучение общей части модели на широкой совокупности большого числа задач.

Архитектуры автокодировщиков

Автокодировщик со снижением размерности

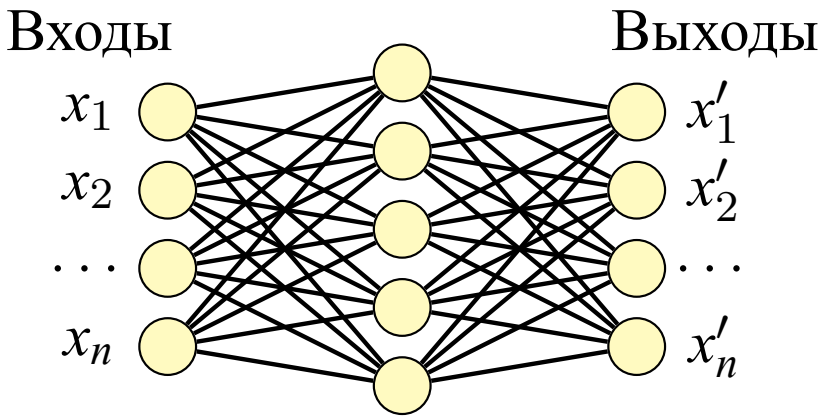


$$Q_{\text{AE}}(\vec{\alpha}, \vec{\beta}) = \frac{1}{L} \sum_{i=1}^{L} \mathcal{L}(g(f(\vec{x}_i, \vec{\alpha}), \vec{\beta}), \vec{x}_i) \to \min_{\vec{\alpha}, \vec{\beta}}$$

https://habr.com/ru/post/331382/

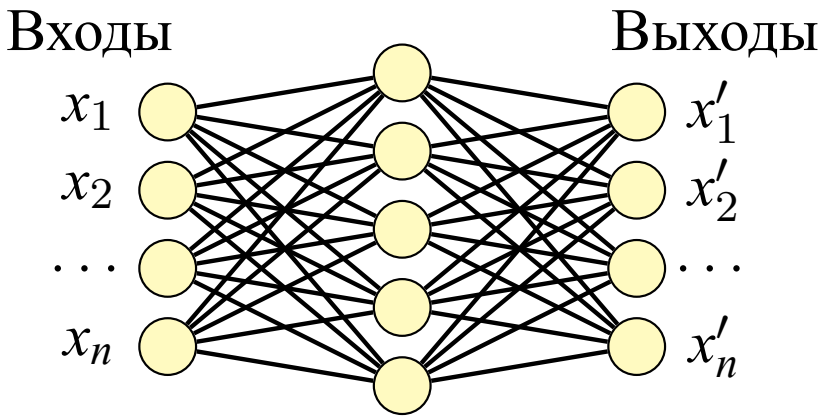
Архитектуры автокодировщиков

Разреживающий автокодировщик (Sparse autoencoder)



Архитектуры автокодировщиков

Разреживающий автокодировщик (Sparse autoencoder)

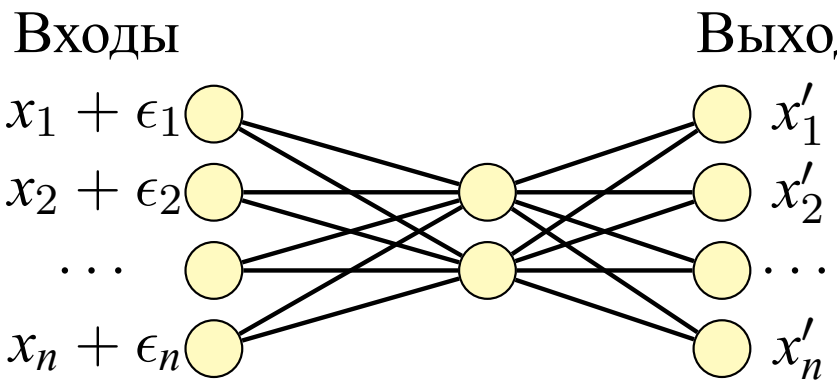


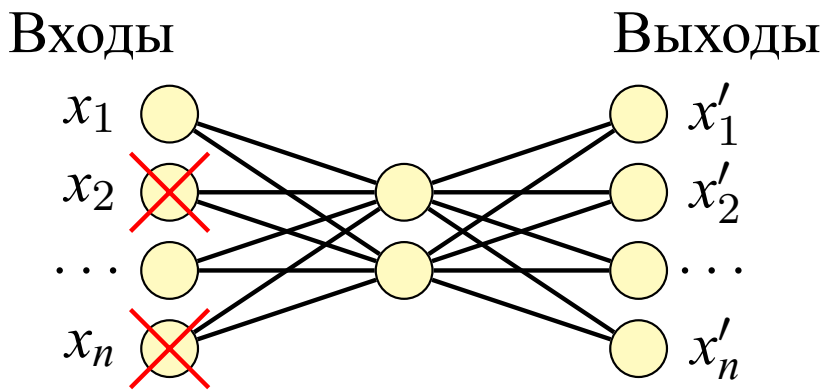
- $Q_{\text{AE}}(\vec{\alpha}, \vec{\beta}) + \underbrace{\lambda \sum_{i=1}^{L} \sum_{j=1}^{m} |f_j(\vec{x}_i, \vec{\alpha})|}_{L1-\text{регуляризация}} \to \min\limits_{\vec{\alpha}, \vec{\beta}}$
- $Q_{\text{AE}}(\vec{\alpha}, \vec{\beta}) + \sum\limits_{j=1}^{m} D_{\text{KL}} \left(\epsilon \middle|\middle| \frac{1}{L} \sum\limits_{i=1}^{L} f_j(\vec{x}_i, \vec{\alpha}) \right) \to \min\limits_{\vec{\alpha}, \vec{\beta}};$

  $D_{\text{KL}}(a, b) = a \ln \frac{a}{b} + (1-a) \ln \frac{1-a}{1-b}$ – расхождение Кульбака-Лейблера.

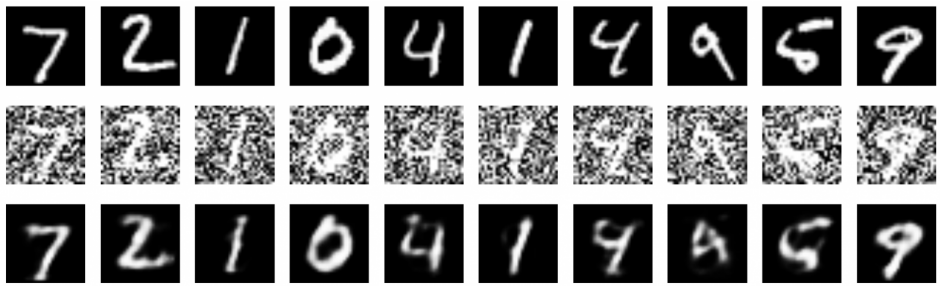
Архитектуры автокодировщиков

Шумоподавляющий автокодировщик (Denoising autoencoder)





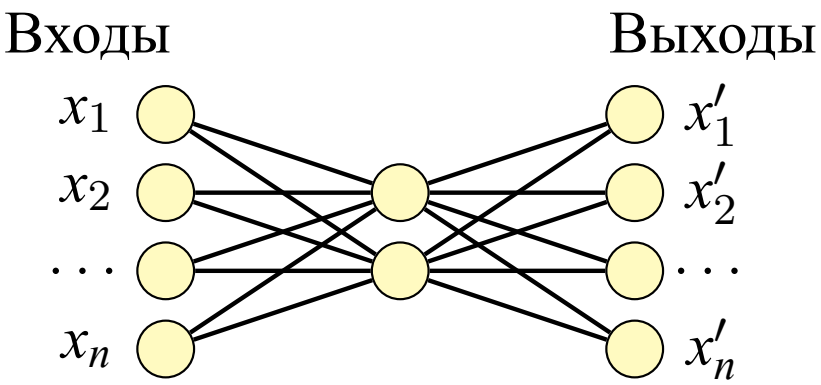
$$Q_{\text{DAE}}(\vec{\alpha}, \vec{\beta}) = \frac{1}{L} \sum_{i=1}^{L} \mathcal{M}_{\vec{\epsilon}} \Big\{ \mathcal{L}(g(f(\vec{x}_i + \vec{\epsilon}, \vec{\alpha}), \vec{\beta}), \vec{x}_i) \Big\} \to \min_{\vec{\alpha}, \vec{\beta}}$$



https://habr.com/ru/post/331382/

Архитектуры автокодировщиков

Сжимающий автокодировщик (Contractive autoencoder)

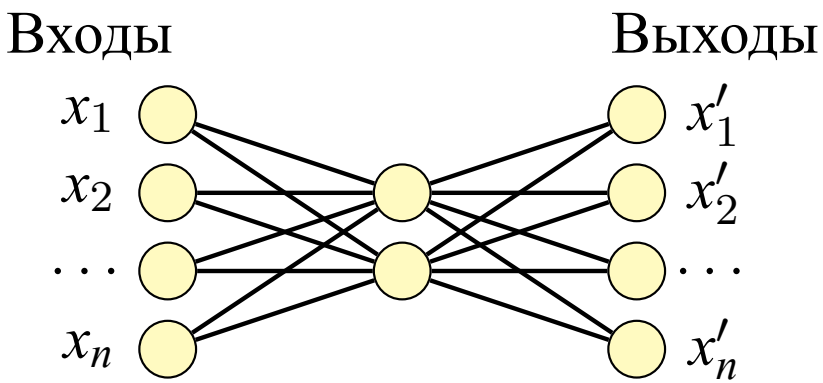


$$Q_{\text{CAE}}(\vec{\alpha},\vec{\beta}) = Q_{\text{AE}}(\vec{\alpha},\vec{\beta}) + \lambda \sum_{i=1}^{L} \underbrace{\sum_{j=1}^{n}\sum_{k=1}^{h}\left(\frac{\partial f_j\,(\vec{x},\vec{\alpha})}{\partial x_k}\bigg|_{\vec{x}=\vec{x}_i}\right)^2}_{\left\|J_f(\vec{x}_i)\right\|^2} \to \min_{\vec{\alpha},\vec{\beta}}$$

Когда норма матрицы Якоби $J_f(\vec{x})$ преобразования f минимальна, шумы на входе слабо влияют на результат кодировщика $f\,(\vec{x})$.

Архитектуры автокодировщиков

Реляционный автокодировщик (Relational autoencoder)



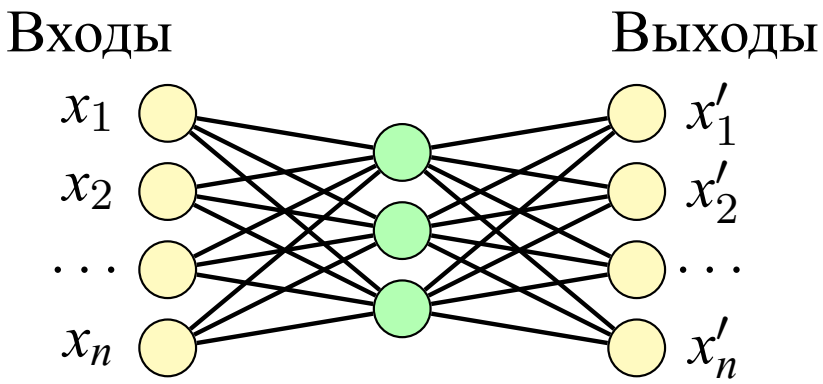
$$Q_{\text{RAE}}(\vec{\alpha},\vec{\beta}) = Q_{\text{AE}}(\vec{\alpha},\vec{\beta}) + \lambda \sum_{i'<i''} \mathcal{L}\Big(\vec{r}\{\vec{a}(\vec{x}_{i'}),\vec{a}(\vec{x}_{i''})\},\vec{r}\{\vec{x}_{i'},\vec{x}_{i''}\}\Big) \to \min_{\vec{\alpha},\vec{\beta}}$$

Наряду с самими объектами $\vec{x}_i$ требуется сохранять отношения $\vec{r}\{\vec{x}_{i'},\vec{x}_{i''}\}$ между ними.

Например, $\vec{r}\{\vec{x}_{i'},\vec{x}_{i''}\} = (\vec{x}_{i'},\vec{x}_{i''})$ – скалярное произведение.

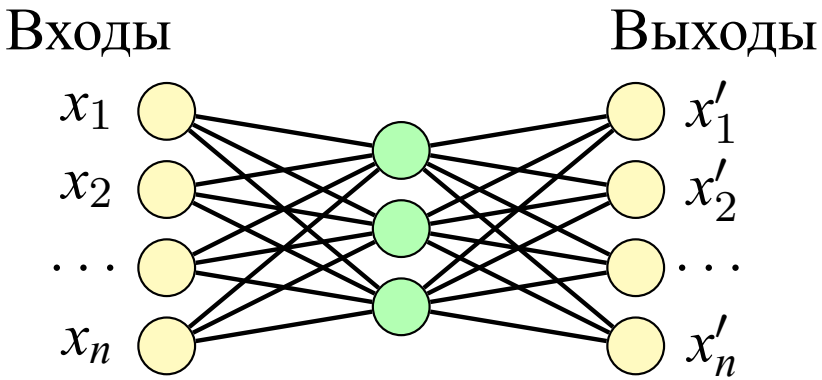
Архитектуры автокодировщиков

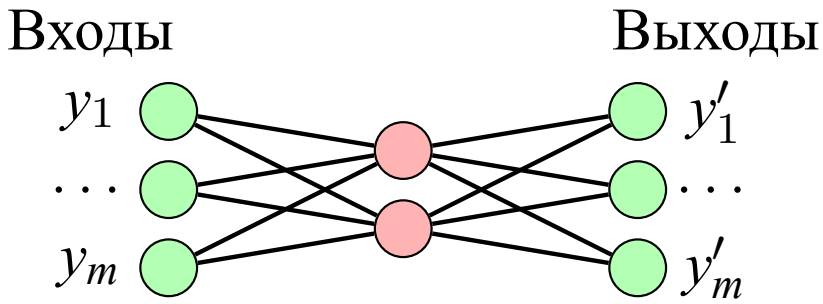
Многослойный автокодировщик (Stacked autoencoder)



Архитектуры автокодировщиков

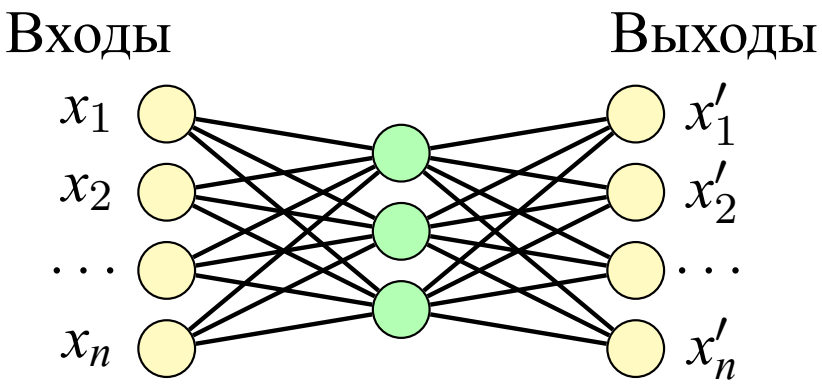
Многослойный автокодировщик (Stacked autoencoder)

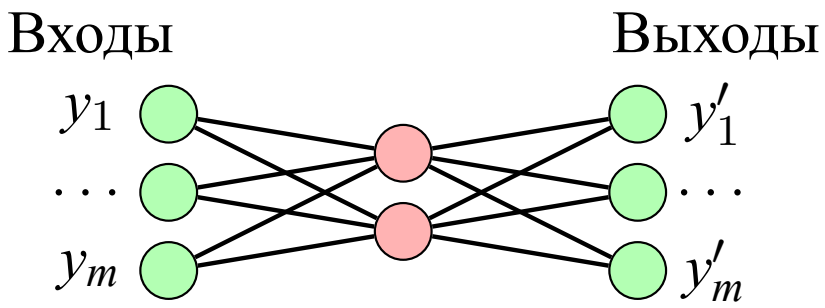


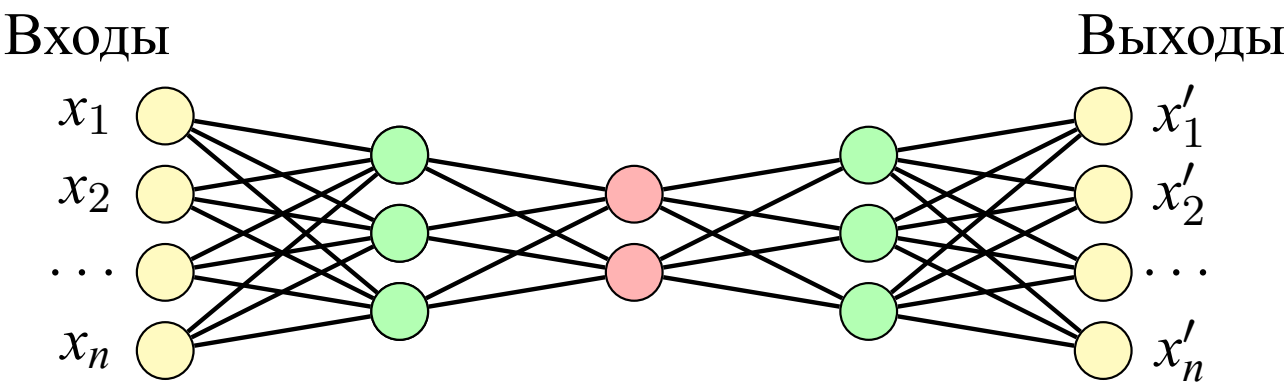


Архитектуры автокодировщиков

Многослойный автокодировщик (Stacked autoencoder)

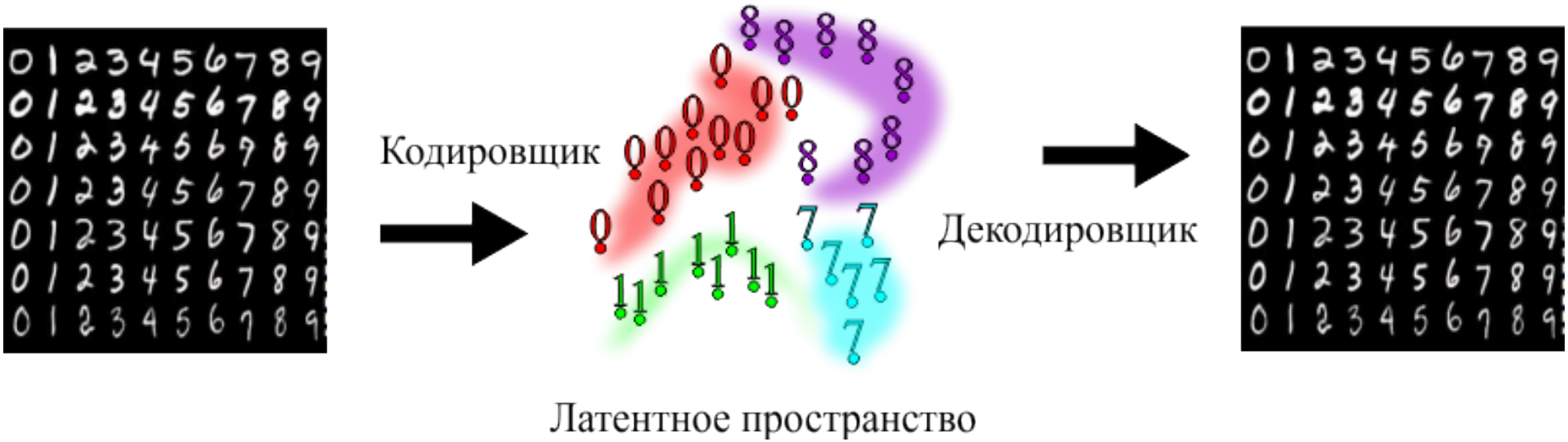






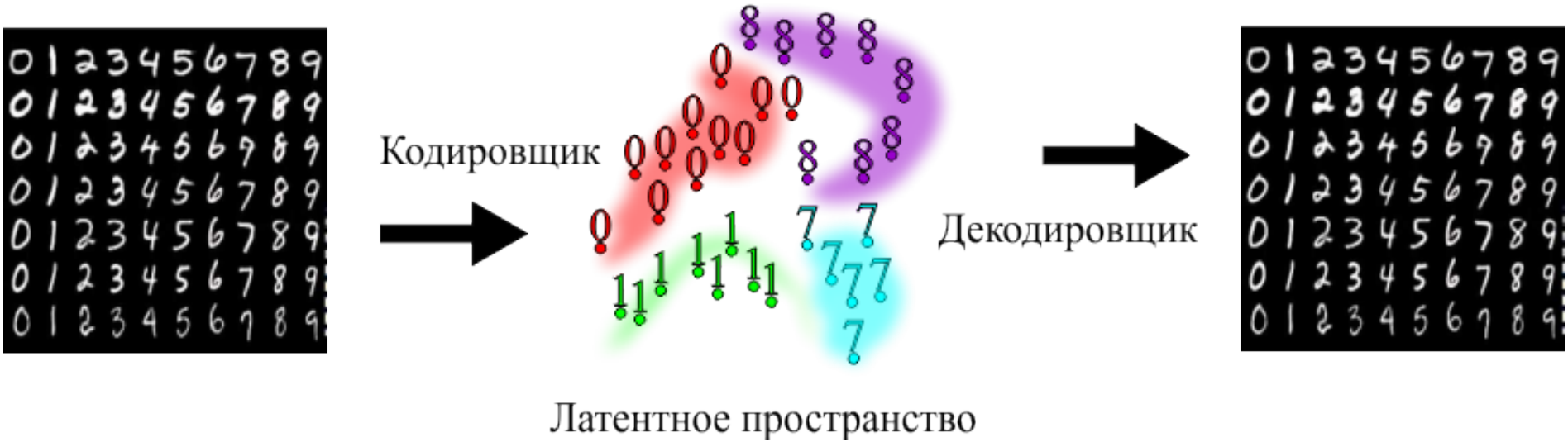
Архитектуры автокодировщиков

Латентное пространство



Архитектуры автокодировщиков

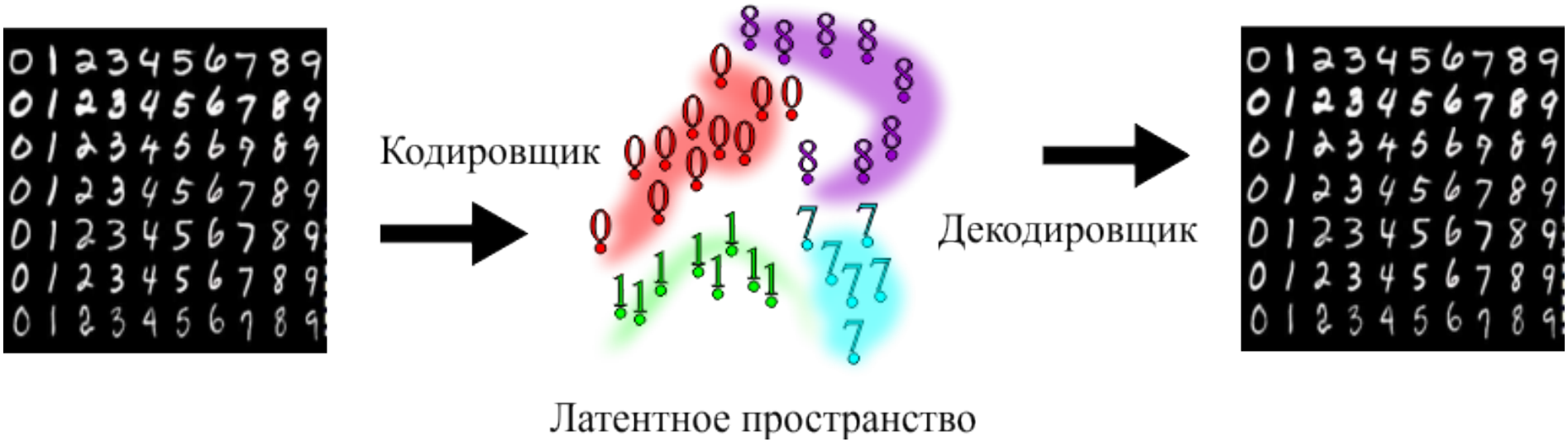
Латентное пространство



- Изображения MNIST составляют малую часть $28^2 = 724$-мерного пространства.

Архитектуры автокодировщиков

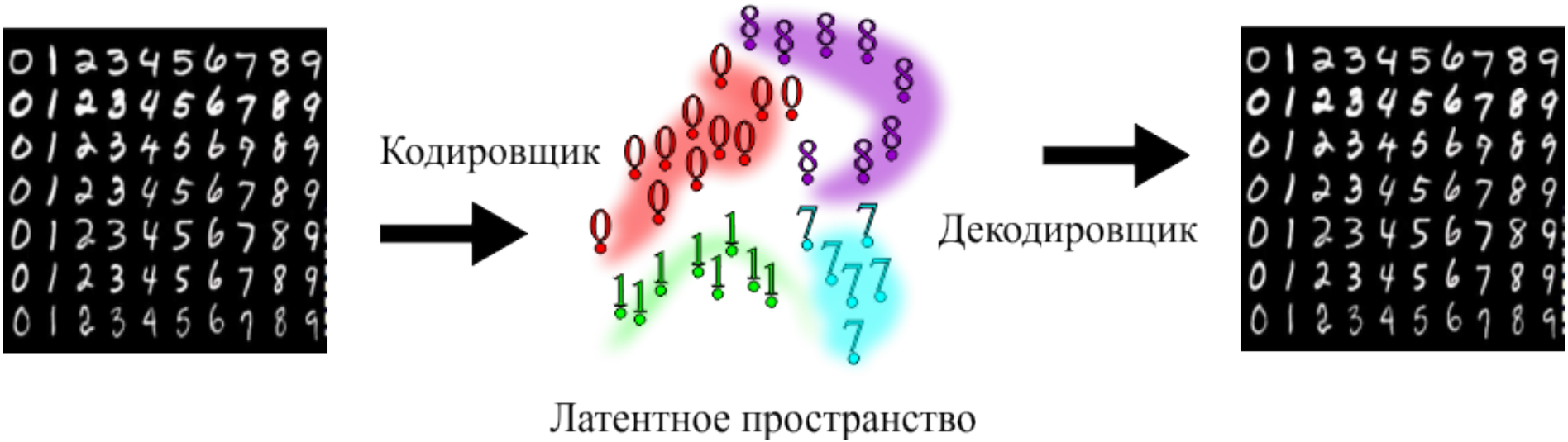
Латентное пространство



- Изображения MNIST составляют малую часть $28^2 = 724$-мерного пространства.
- Автокодировщики ищут многообразия (manifolds) с высокой вероятностью встретить на них объекты обучающей выборки.

Архитектуры автокодировщиков

Латентное пространство

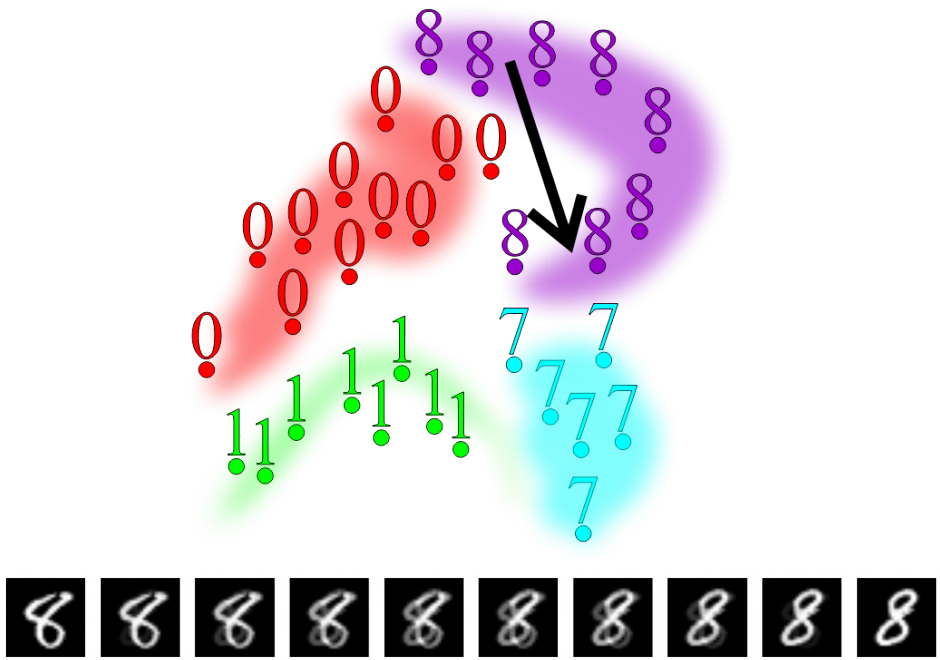


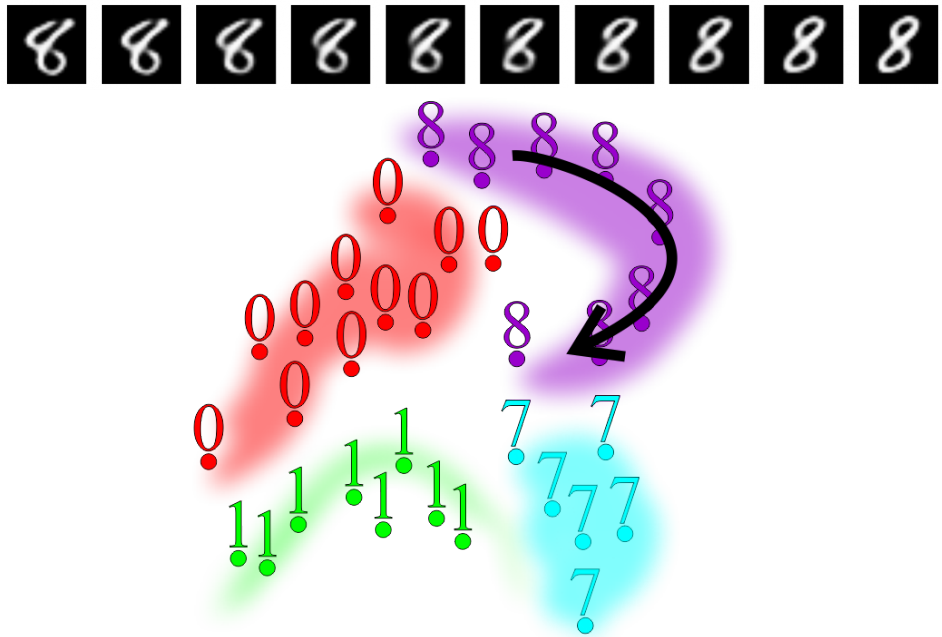
- Изображения MNIST составляют малую часть $28^2 = 724$-мерного пространства.
- Автокодировщики ищут многообразия (manifolds) с высокой вероятностью встретить на них объекты обучающей выборки.
- Размерность найденного многообразия много меньше исходной.

Архитектуры автокодировщиков

Латентное пространство

Недостаток детерминированного кодировщика –
только комбинация признаков несет разумную информацию.

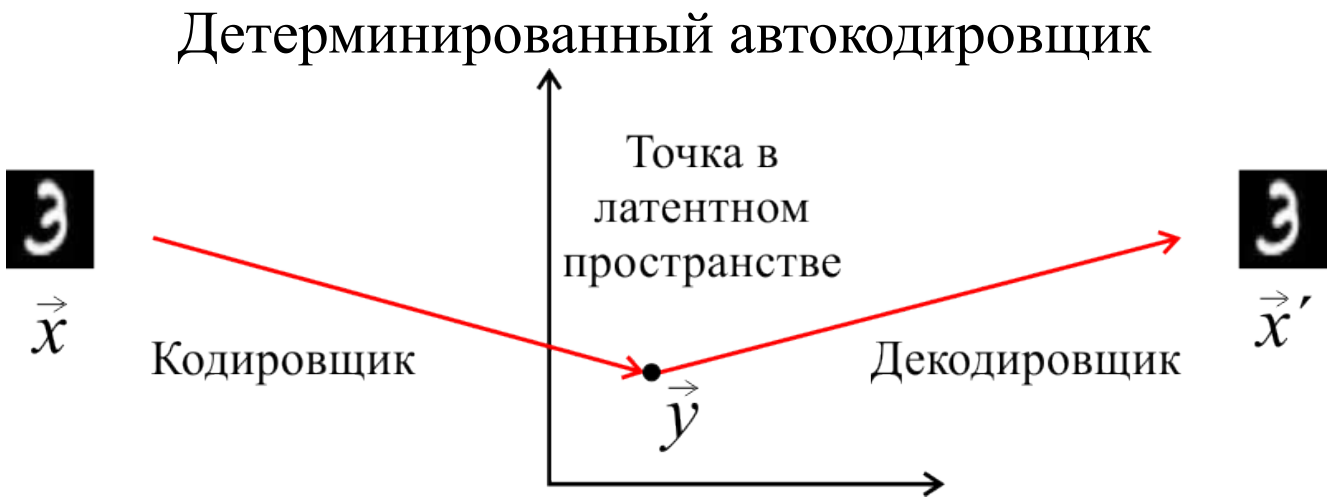




https://habr.com/ru/post/331500/

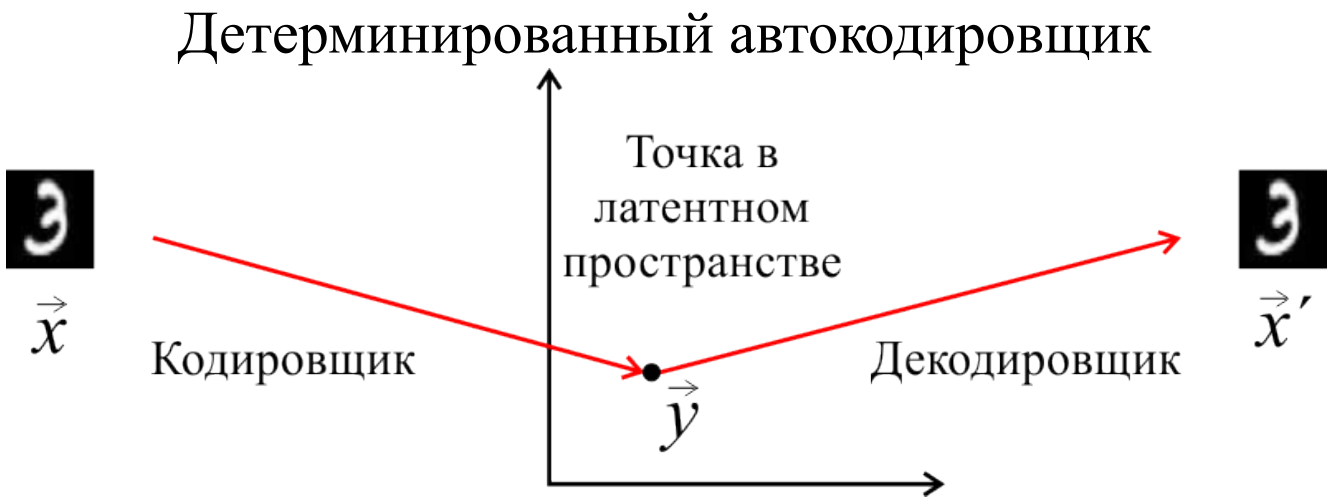
Архитектуры автокодировщиков

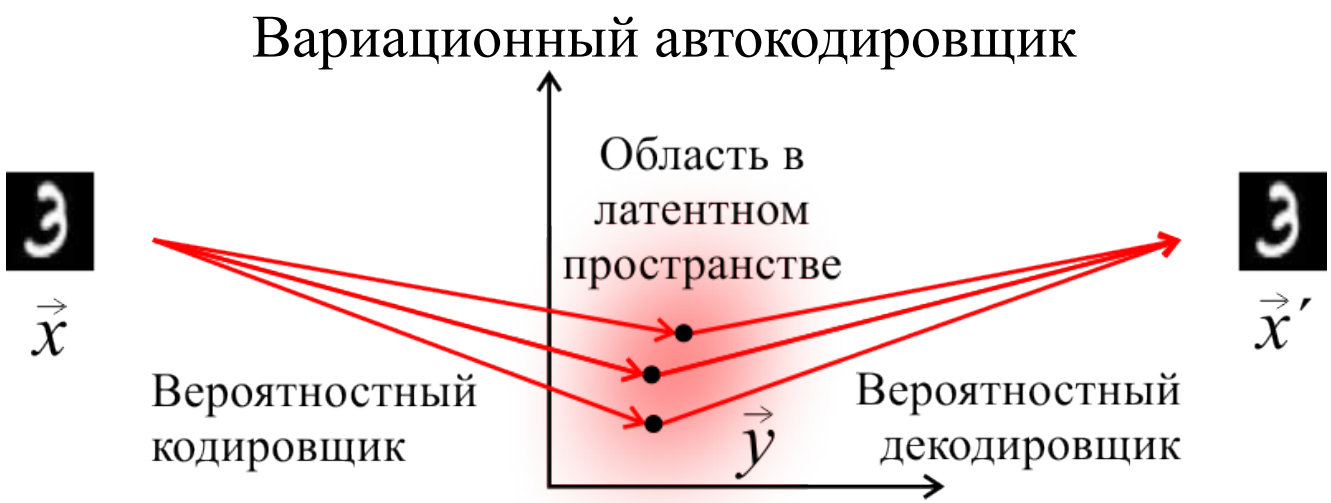
Вариационный автокодировщик (Variational autoencoder)



Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)





Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)

Вариационный кодировщик $q_{\vec{\alpha}}(\vec{y}|\vec{x})$ и декодировщик $p_{\vec{\beta}}(\vec{x}'|\vec{y})$.

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)

Вариационный кодировщик $q_{\vec{\alpha}}(\vec{y}|\vec{x})$ и декодировщик $p_{\vec{\beta}}(\vec{x}'|\vec{y})$.

$$p(\vec{x}) = \int p_{\vec{\beta}}(\vec{x}|\vec{y})p(\vec{y})d\vec{y} = \int q_{\vec{\alpha}}(\vec{y}|\vec{x})\frac{p_{\vec{\beta}}(\vec{x}|\vec{y})p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x})}d\vec{y}$$

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)

Вариационный кодировщик $q_{\vec{\alpha}}(\vec{y}|\vec{x})$ и декодировщик $p_{\vec{\beta}}(\vec{x}'|\vec{y})$.

$$p(\vec{x}) = \int p_{\vec{\beta}}(\vec{x}|\vec{y})p(\vec{y})d\vec{y} = \int q_{\vec{\alpha}}(\vec{y}|\vec{x})\frac{p_{\vec{\beta}}(\vec{x}|\vec{y})p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x})}d\vec{y}$$

Принцип максимума правдоподобия

$$\ln L = \sum_{i=1}^{L} \ln p(\vec{x}_i) = \sum_{i=1}^{L} \ln \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)\frac{p_{\vec{\beta}}(\vec{x}_i|\vec{y})p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)}d\vec{y} \geq$$

$$\geq \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p_{\vec{\beta}}(\vec{x}_i|\vec{y})p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)}d\vec{y} =$$

$$= \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)}d\vec{y} + \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y})d\vec{y} \to \max_{\vec{\alpha},\vec{\beta}}$$

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder). Репараметризация

Вариационный кодировщик $q_{\vec{\alpha}}(\vec{y}|\vec{x})$ и декодировщик $p_{\vec{\beta}}(\vec{x}'|\vec{y})$.

$$\ln L = \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} d\vec{y} + \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) d\vec{y} \to \max_{\vec{\alpha},\vec{\beta}}$$

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder). Репараметризация

Вариационный кодировщик $q_{\vec{\alpha}}(\vec{y}|\vec{x})$ и декодировщик $p_{\vec{\beta}}(\vec{x}'|\vec{y})$.

$$\ln L = \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} d\vec{y} + \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) d\vec{y} \to \max_{\vec{\alpha},\vec{\beta}}$$

- $\sum\limits_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} d\vec{y} = -D_{\mathrm{KL}}\left(q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)||p(\vec{y})\right)$
- $\sum\limits_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) d\vec{y} = \sum\limits_{i=1}^{L} \mathcal{M}_{y\sim q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} \left\{ \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) \right\}$

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder). Репараметризация

Вариационный кодировщик $q_{\vec{\alpha}}(\vec{y}|\vec{x})$ и декодировщик $p_{\vec{\beta}}(\vec{x}'|\vec{y})$.

$$\ln L = \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} d\vec{y} + \sum_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) d\vec{y} \to \max_{\vec{\alpha},\vec{\beta}}$$

- $\sum\limits_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln \frac{p(\vec{y})}{q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} d\vec{y} = -D_{\mathrm{KL}}\left(q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)||p(\vec{y})\right)$
- $\sum\limits_{i=1}^{L} \int q_{\vec{\alpha}}(\vec{y}|\vec{x}_i) \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) d\vec{y} = \sum\limits_{i=1}^{L} \mathcal{M}_{y\sim q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)} \left\{ \ln p_{\vec{\beta}}(\vec{x}_i|\vec{y}) \right\}$

Идея: вместо распределения $q_{\vec{\alpha}}(\vec{y}|\vec{x}_i)$
рассматривается $\vec{y} = \vec{y}(\vec{x}_i, \vec{\alpha}, \vec{\epsilon})$, где $\vec{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$.

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)

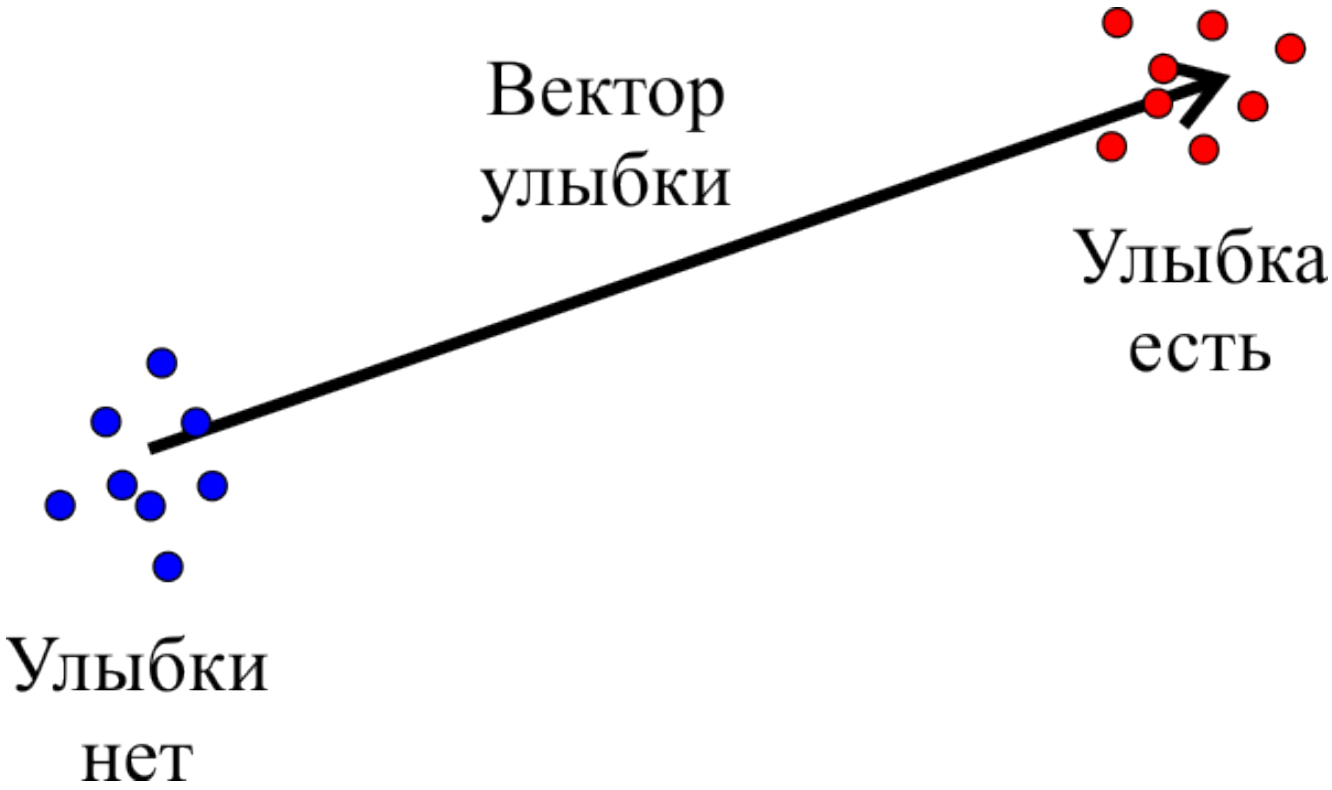
Перемещение в латентном пространстве

https://gaussian37.github.io/deep-learning-chollet-8-4/

Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)

Векторы в латентном пространстве



Архитектуры автокодировщиков

Вариационный автокодировщик (Variational autoencoder)

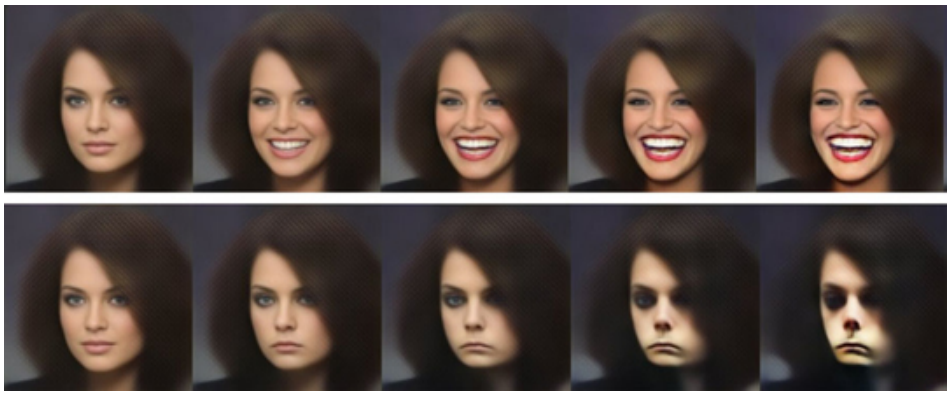
Векторы в латентном пространстве

https://arxiv.org/pdf/1609.04468.pdf

Архитектуры автокодировщиков

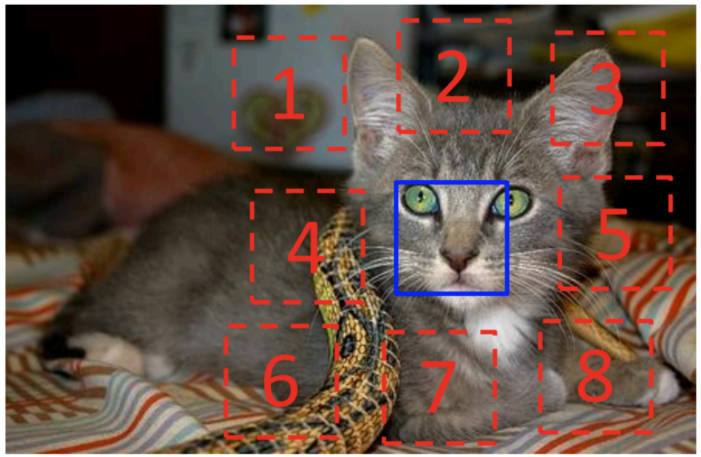
Вариационный автокодировщик (Variational autoencoder)

Векторы в латентном пространстве.

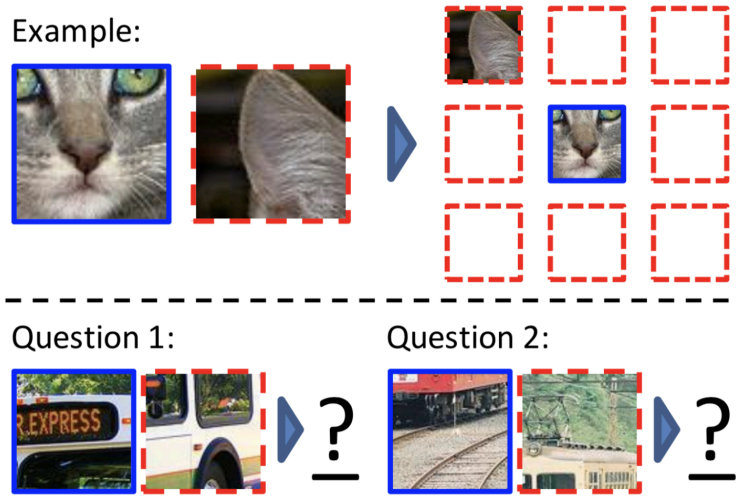


https://gaussian37.github.io/deep-learning-chollet-8-4/

Самообучение (Self-Supervised Learning)

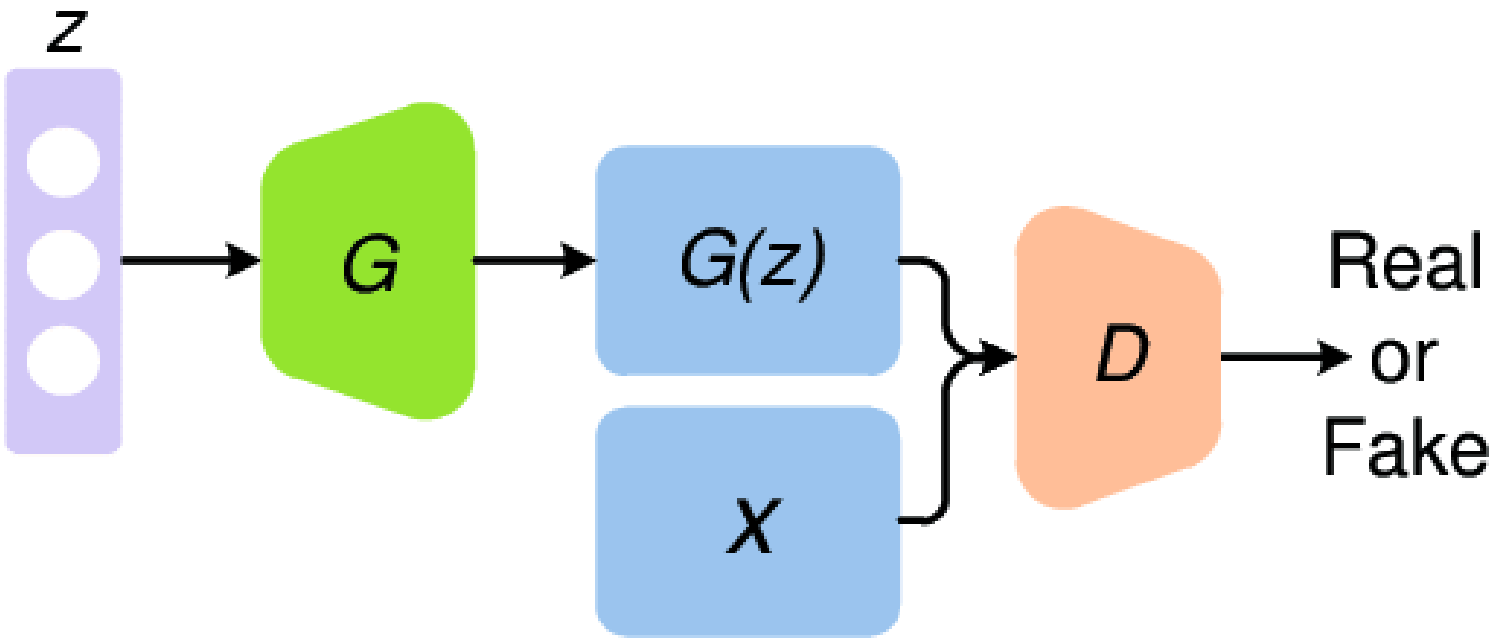


X = (,); Y = 3



https://arxiv.org/abs/1505.05192

Генеративная состязательная сеть (Generative Adversarial Net)



https://www.researchgate.net/publication/331756737_Recent_Progress_on_Generative_Adversarial_Networks_GANs_A_Survey

Генеративная состязательная сеть (Generative Adversarial Net)

Генератор случайных лиц StyleGAN



https://this-person-does-not-exist.com/ru

Векторные представления текста

Задача: на основе последовательности слов w_i (текста) сопоставить каждому слову вектор $\vec{v}_w$ так, чтобы близким по смыслу словам соответствовали близкие векторы.

Векторные представления текста

Задача: на основе последовательности слов w_i (текста) сопоставить каждому слову вектор $\vec{v}_w$ так, чтобы близким по смыслу словам соответствовали близкие векторы.

- Синтагматическая близость
  компьютер – программа, тетрадь – ручка
- Парадигматическая близость
  компьютер – ноутбук, тетрадь – блокнот.

Векторные представления текста

Задача: на основе последовательности слов w_i (текста) сопоставить каждому слову вектор $\vec{v}_w$ так, чтобы близким по смыслу словам соответствовали близкие векторы.

- Синтагматическая близость
  компьютер – программа, тетрадь – ручка
- Парадигматическая близость
  компьютер – ноутбук, тетрадь – блокнот.

$$\underbrace{\text{Съешь же ещё этих мягких французских}}_{\text{Контекст } C_i=(w_{i-k},\ldots,w_{i-1},w_{i+1},\ldots,w_{i+k})}\ \underbrace{\text{булок}}_{\text{Слово } w_i}\ ,\ \underbrace{\text{да выпей чаю}}_{\text{Контекст}}$$

Векторные представления текста

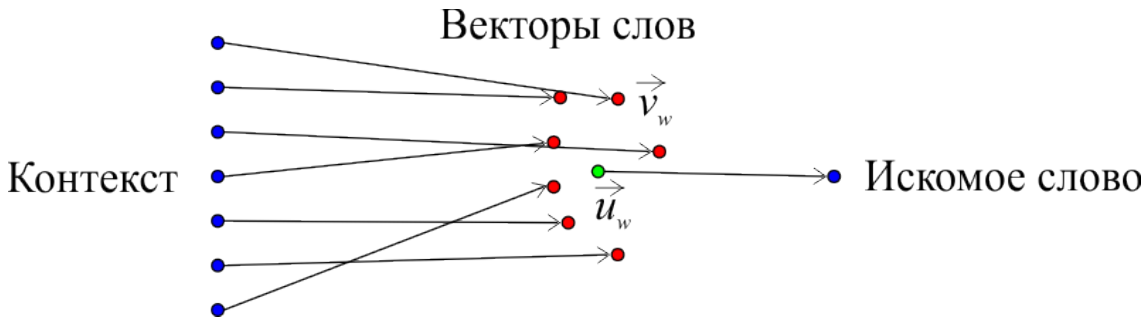
Модель Continuous Bag-Of-Words

Задача: предсказать слово w_i по его контексту C_i.

Векторные представления текста

Модель Continuous Bag-Of-Words

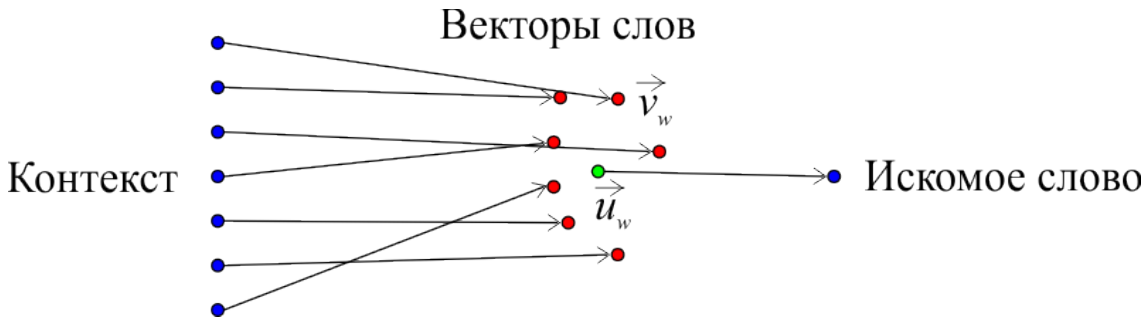
Задача: предсказать слово w_i по его контексту C_i.



Векторные представления текста

Модель Continuous Bag-Of-Words

Задача: предсказать слово w_i по его контексту C_i.

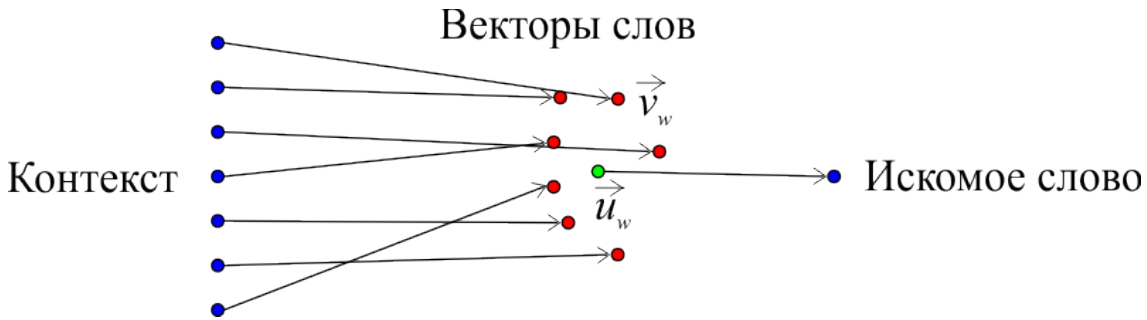


- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.

Векторные представления текста

Модель Continuous Bag-Of-Words

Задача: предсказать слово w_i по его контексту C_i.

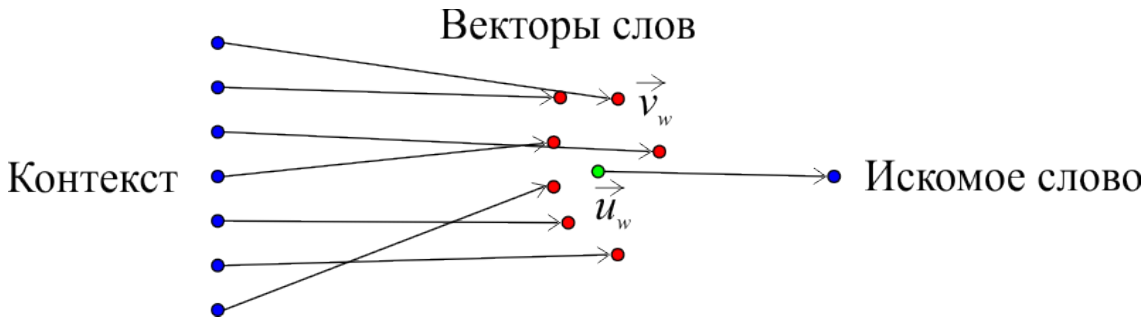


- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.
- Средний вектор предсказывающих слов $\vec{v}_{C_i} = \frac{1}{2k} \sum\limits_{w_j \in C_i} \vec{v}_{w_j}$.

Векторные представления текста

Модель Continuous Bag-Of-Words

Задача: предсказать слово w_i по его контексту C_i.

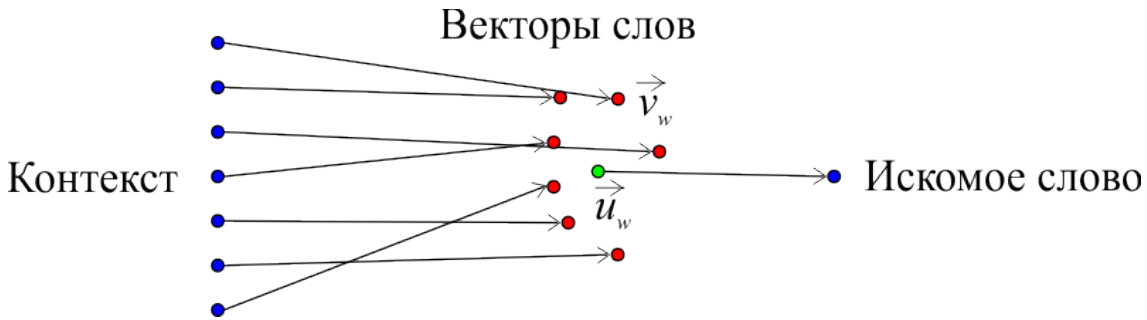


- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.
- Средний вектор предсказывающих слов $\vec{v}_{C_i} = \frac{1}{2k} \sum\limits_{w_j \in C_i} \vec{v}_{w_j}$.
- Модель $p(w|C_i) = \text{softmax}(\vec{v}_{C_i} \cdot \vec{u}_w)$.

Векторные представления текста

Модель Continuous Bag-Of-Words

Задача: предсказать слово w_i по его контексту C_i.



- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.
- Средний вектор предсказывающих слов $\vec{v}_{C_i} = \frac{1}{2k} \sum\limits_{w_j \in C_i} \vec{v}_{w_j}$.
- Модель $p(w|C_i) = \text{softmax}(\vec{v}_{C_i} \cdot \vec{u}_w)$.
- Критерий максимального log-правдоподобия:

$$\sum_{i=1}^{n} \ln p(w_i|C_i) = \sum_{i=1}^{n} \ln \text{softmax}\left(\frac{1}{2k} \sum_{w_j \in C_i} (\vec{v}_{w_j} \cdot \vec{u}_w) \right) \to \max_{\mathbf{U},\,\mathbf{V}}$$

Векторные представления текста

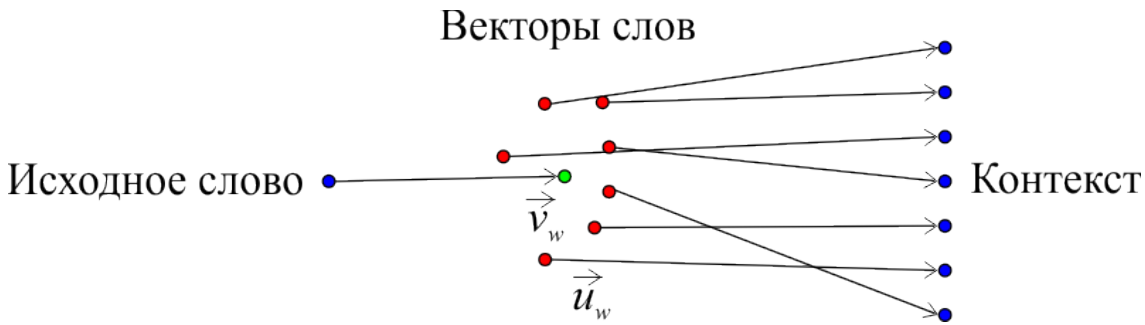
Модель Skip-gram

Задача: предсказать контекст C_i по слову w_i.

Векторные представления текста

Модель Skip-gram

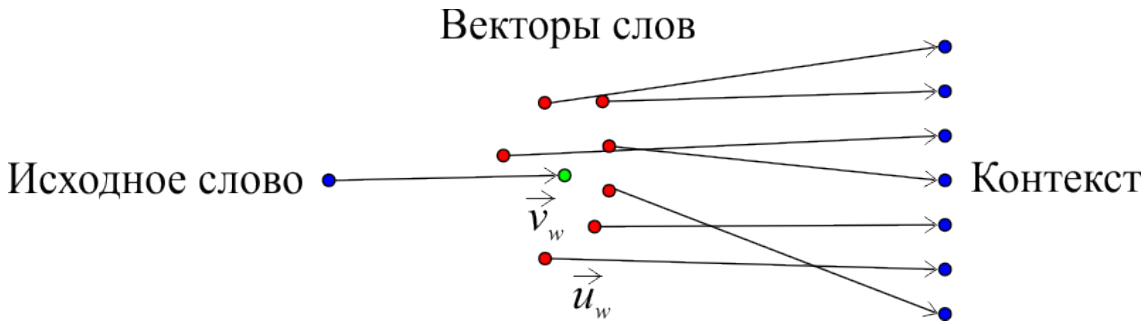
Задача: предсказать контекст C_i по слову w_i.



Векторные представления текста

Модель Skip-gram

Задача: предсказать контекст C_i по слову w_i.

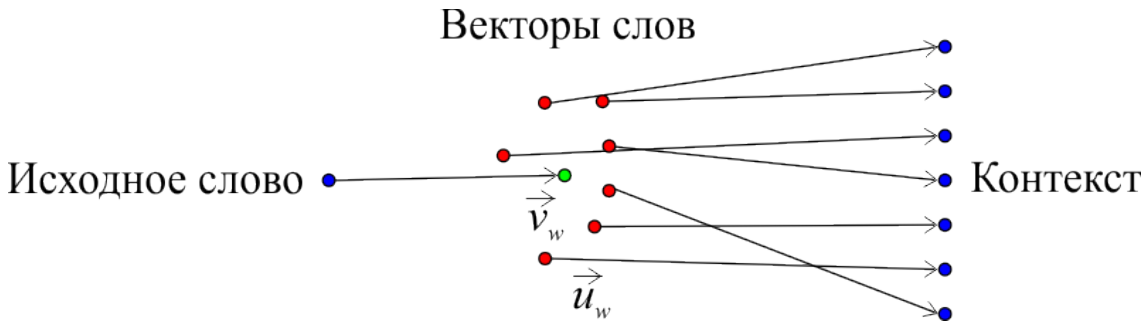


- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.

Векторные представления текста

Модель Skip-gram

Задача: предсказать контекст C_i по слову w_i.

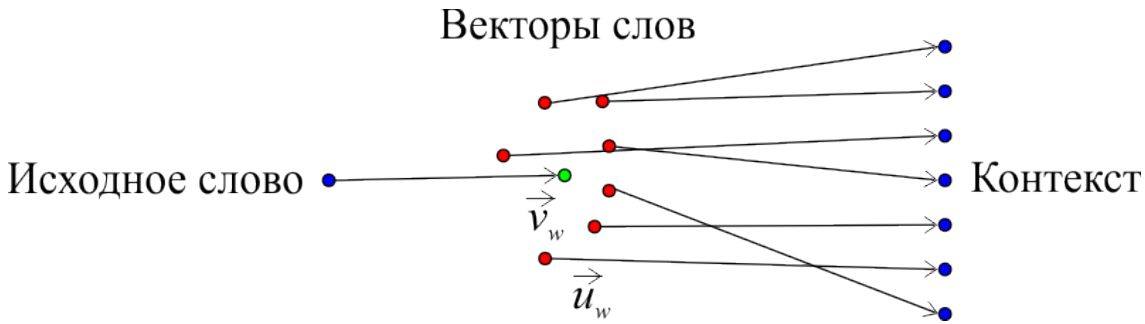


- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.
- Модель $p(w|w_i) = \text{softmax}(\vec{v}_{w_i} \cdot \vec{u}_w)$.

Векторные представления текста

Модель Skip-gram

Задача: предсказать контекст C_i по слову w_i.



- Вводятся векторы $\vec{v}_w$ для каждого предсказывающего слова.
- Вводится вектор $\vec{u}_w$ для предсказываемого слова.
- Модель $p(w|w_i) = \text{softmax}(\vec{v}_{w_i} \cdot \vec{u}_w)$.
- Критерий максимального log-правдоподобия:

$$\sum_{i=1}^{n} \sum_{w \in C_i} \ln p(w|w_i) = \sum_{i=1}^{n} \sum_{w \in C_i} \ln \text{softmax}\,(\vec{v}_{w_i} \cdot \vec{u}_w) \to \max_{\mathbf{U},\,\mathbf{V}}$$

Векторные представления текста

One-hot кодировка для текста

$$
\begin{aligned}
\text{Съешь} &\to (1,0,0,0,0,0,\dots)\\
\text{же} &\to (0,1,0,0,0,0,\dots)\\
\text{ещё} &\to (0,0,1,0,0,0,\dots)\\
\text{этих} &\to (0,0,0,1,0,0,\dots)\\
\text{мягких} &\to (0,0,0,0,1,0,\dots)\\
\text{французских} &\to (0,0,0,0,0,1,\dots)\\
&\dots
\end{aligned}
$$

$$\vec{v}_w = \mathbf{V}_{[W\times D]}\vec{w}; \quad \vec{u}_w = \mathbf{U}_{[W\times D]}\vec{w}$$

Векторные представления текста

Negative sampling

Критерий Skip-gram:

$$\sum_{i=1}^{n} \sum_{w \in C_i} \ln p(w|w_i) \to \max_{\mathbf{U},\mathbf{V}}$$

Векторные представления текста

Negative sampling

Критерий Skip-gram:

$$\sum_{i=1}^{n} \sum_{w \in C_i} \ln p(w|w_i) \to \max_{\mathbf{U},\mathbf{V}}$$

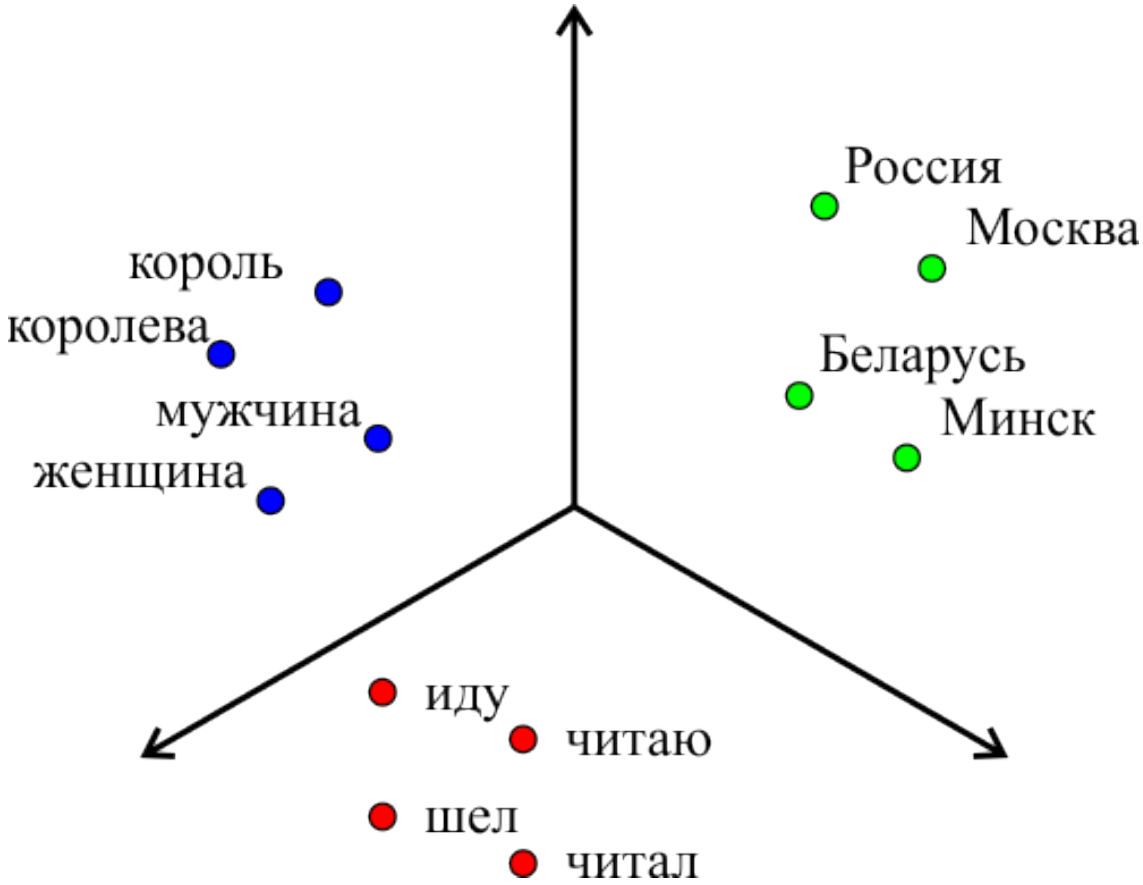
Критерий Skip-gram Negative Sampling:

$$\sum_{i=1}^{n} \sum_{w \in C_i} \left(\ln p(+1|w, w_i) + \ln p(-1|w^-, w_i)\right) \to \max_{\mathbf{U},\mathbf{V}}$$

- Модель $p(y|w, w_i) = \text{sigmoid}(y(\vec{u}_w \cdot \vec{v}_{w_i}))$.
- $y = +1$, если слова (w, w_i) находятся в контексте, и $y = -1$, если нет.
- Слово $w^- \notin C_i$ семплируется с вероятностью $p(w)^{3/4}$.

Векторные представления текста

Проверка: задачи семантической близости



Векторные представления текста

Модель FastText

Слово “хвалолюб”

Набор n-грамм: $\mathbb{G}$ ={хв, хва, вал, ало, лол, олю, люб, юб}

$$\vec{u}_w = \sum_{g \in \mathbb{G}} \vec{u}_g$$

- Решаются проблемы с опечатками и с редкими или новыми словами.
- Число n-грамм гораздо меньше числа слов.

Векторные представления графов

Многомерное шкалирование

Задача: на основе данных о расстояниях R_{ij} между вершинами графа i и j сопоставить каждой вершине вектор $\vec{v}_i$ так, чтобы близким по графу вершинам соответствовали близкие векторы.

Векторные представления графов

Многомерное шкалирование

Задача: на основе данных о расстояниях R_{ij} между вершинами графа i и j сопоставить каждой вершине вектор $\vec{v}_i$ так, чтобы близким по графу вершинам соответствовали близкие векторы.

Решение – SGD для критерия стресса (stress):

$$\sum_{i,j} K(R_{ij})\left(\left\|\vec{v}_i - \vec{v}_j\right\| - R_{ij}\right)^2 \to \min$$

Векторные представления графов

Матричные разложения графа (graph factorization)

Задача: на основе данных о близости S_{ij} вершин графа i и j сопоставить каждой вершине вектор $\vec{v}_i$ так, чтобы близким по графу вершинам соответствовали близкие векторы.

Векторные представления графов

Матричные разложения графа (graph factorization)

Задача: на основе данных о близости S_{ij} вершин графа i и j сопоставить каждой вершине вектор $\vec{v}_i$ так, чтобы близким по графу вершинам соответствовали близкие векторы.

Решение для неориентированного графа:

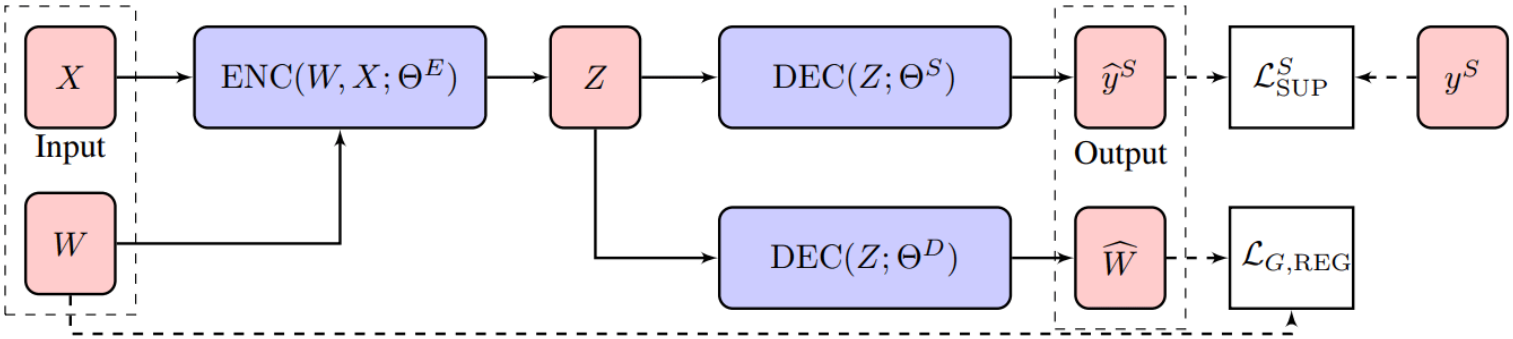
$$\sum_{i,j}((\vec{v}_i \cdot \vec{v}_j) - S_{ij})^2 \to \min_{\mathbf{V}}$$

Решение для ориентированного графа:

$$\sum_{i,j}((\vec{v}_i \cdot \vec{u}_j) - S_{ij})^2 \to \min_{\mathbf{U},\mathbf{V}}$$

Векторные представления графов

Графы и обобщенный автокодировщик. GraphEDM



- **X** – признаковые описания вершин;
- **W** – информация о ребрах графа;
- **Z** – векторное представление вершин;
- $\hat{y}^S$ – результат для задачи обучения с учителем, который сравнивается с y^S с помощью функции потерь $\mathcal{L}^S_{\mathrm{SUP}}$;
- $\hat{\mathbf{W}}$ – результат для задачи обучения без учителя, который сравнивается с **W** с помощью функции потерь $\mathcal{L}_{G,\mathrm{REG}}$

`https://arxiv.org/pdf/2005.03675.pdf`

Лекция 16. Работа с временными рядами

Методы машинного обучения в анализе изображений и временных рядов

Дмитриев Константин Вячеславович

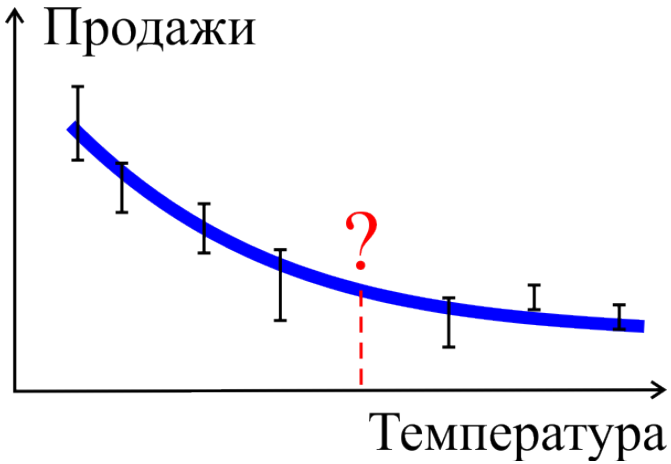
Московский государственный университет имени М.В. Ломоносова

Схема постановки и решения задачи прогнозирования

- Имеется временной ряд $x_0, x_1, \ldots, x_t$.
- Имеется целевой временной ряд $y_0, y_1, \ldots, y_t$.
- Задается функционал качества $\mathcal{L}(\vec{a}, y_{t+1}, \ldots, y_{t+d})$, характеризующий, насколько хорошо алгоритм $\vec{a}$ аппроксимирует значение ряда в течение d следующих шагов.
- Выбирается параметрическая модель $\vec{a}(x_0, \ldots, x_t; y_0, \ldots, y_t, \vec{\theta})$.
- Модель должна учитывать рост объема данных с ростом t.
- Параметры модели $\vec{\theta}$ оптимизируются для достижения минимума $\mathcal{L}$.

Особенность предсказания временных рядов

Интерполяция и экстраполяция

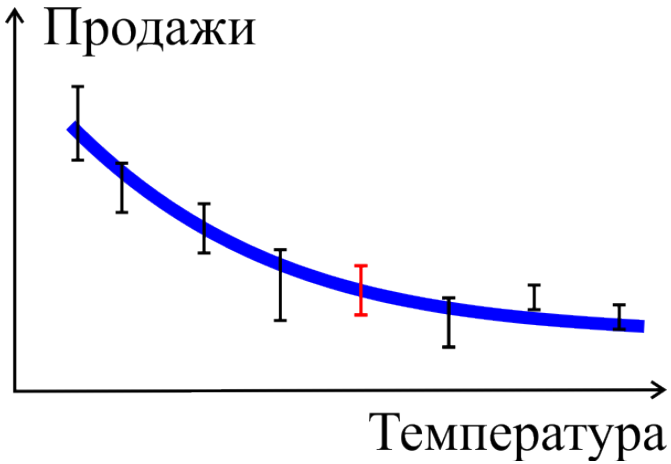


Интерполяция

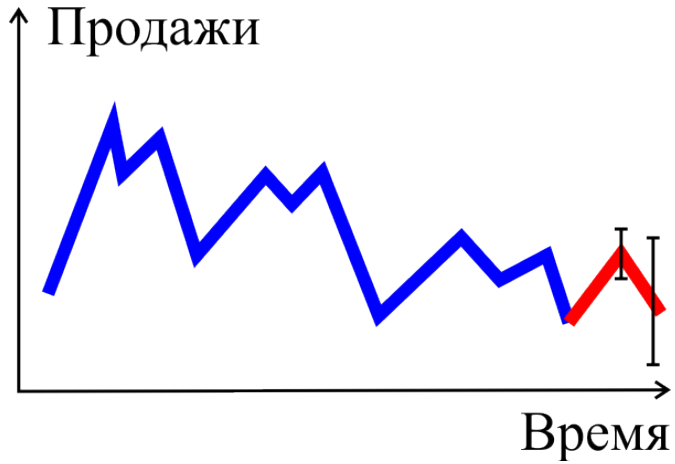
Экстраполяция

Особенность предсказания временных рядов

Изменение ошибки



Интерполяция



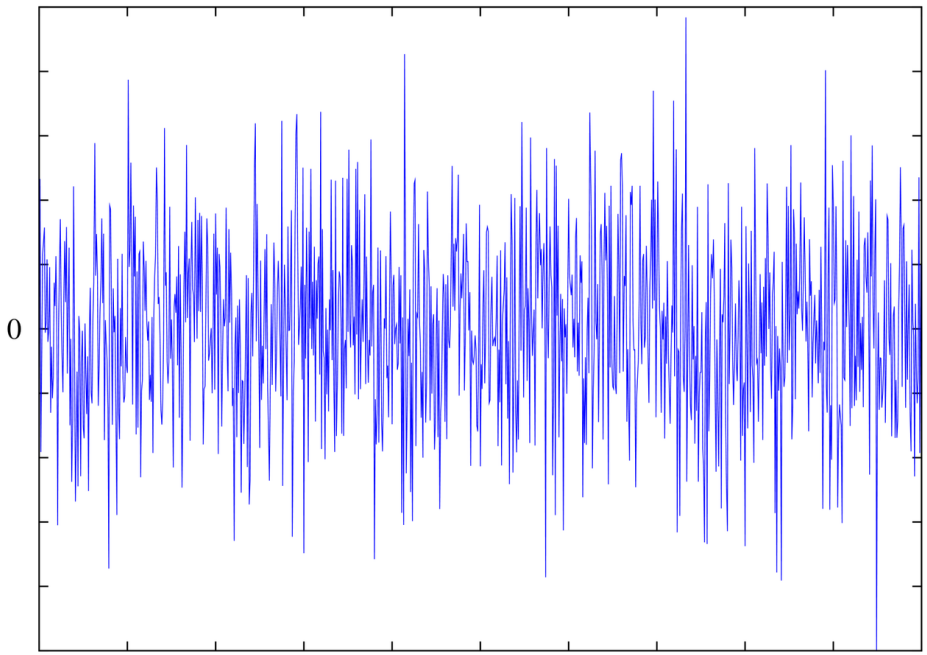
Экстраполяция

Стационарные процессы

Определение 16.1

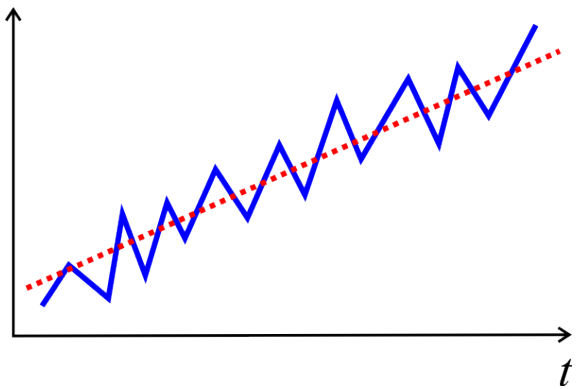
Стационарный случайный процесс (в широком смысле) — *такой процесс* ξ_t, *что*

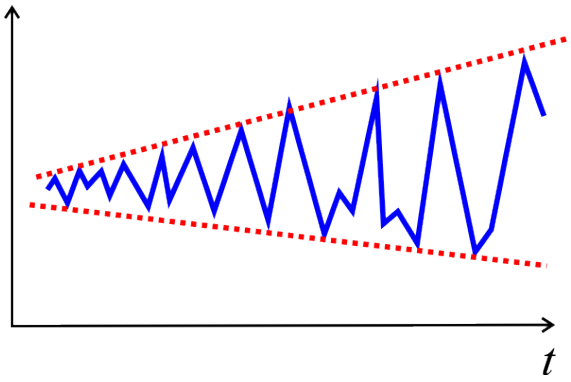
$\exists \mathcal{M}\{\xi_t\} = const;\ \exists \mathcal{D}\{\xi_t\} > 0;\ cov(\xi_t, \xi_{t'}) = cov(\xi_{t+s}, \xi_{t'+s})$

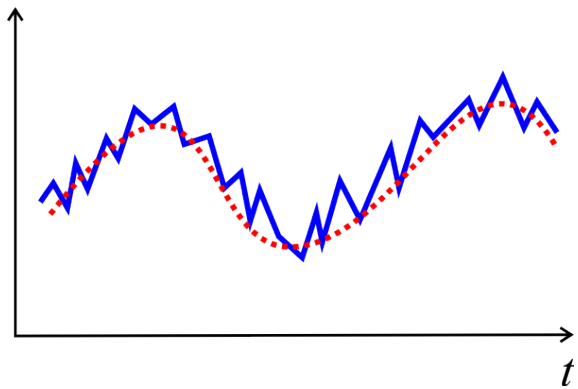


Стационарные процессы

- Среднее значение постоянно.
- Дисперсия постоянна.
- Отсутствует сезонность.







Разложение сигнала по базисным функциям

$$f(t) = \int_s \widetilde{f}(s)\Psi(s,t)ds \qquad f(t) = \sum_n \widetilde{f}_n \Psi_n(t)$$

Разложение сигнала по базисным функциям

$$f(t) = \int_s \widetilde{f}(s)\Psi(s,t)ds \qquad f(t) = \sum_n \widetilde{f}_n \Psi_n(t)$$

Выбор «удобных» базисных функций определяется *задачей*

$$\longrightarrow f(t) \quad \boxed{\hat{A}} \quad \longrightarrow g(t) = \hat{A}f(t)$$

Разложение сигнала по базисным функциям

$$f(t) = \int_s \widetilde{f}(s)\Psi(s,t)ds \qquad f(t) = \sum_n \widetilde{f}_n \Psi_n(t)$$

Выбор «удобных» базисных функций определяется *задачей*

$$\xrightarrow{f(t)} \boxed{\hat{A}} \xrightarrow{g(t) = \hat{A}f(t)}$$

$\hat{A}$ — линейный оператор; $\hat{A}\Psi_n(t) = \alpha_n \Psi_n(t)$

Разложение сигнала по базисным функциям

$$f(t) = \int_s \widetilde{f}(s)\Psi(s,t)ds \qquad f(t) = \sum_n \widetilde{f}_n \Psi_n(t)$$

Выбор «удобных» базисных функций определяется *задачей*

$$\xrightarrow{f(t)} \boxed{\hat{A}} \xrightarrow{g(t) = \hat{A}f(t)}$$

$\hat{A}$ — линейный оператор; $\hat{A}\Psi_n(t) = \alpha_n \Psi_n(t)$

$$\hat{A}f(t) = \hat{A}\sum_n \widetilde{f}_n \Psi_n(t) = \sum_n \widetilde{f}_n \hat{A}\Psi_n(t) = \sum_n \widetilde{f}_n \alpha_n \Psi_n(t) = \sum_n \widetilde{g}_n \Psi_n(t) = g(t)$$

Разложение сигнала по базисным функциям

$$f(t) = \int_s \widetilde{f}(s)\Psi(s,t)ds \qquad f(t) = \sum_n \widetilde{f}_n \Psi_n(t)$$

Выбор «удобных» базисных функций определяется *задачей*

$$\xrightarrow{f(t)} \boxed{\hat{A}} \xrightarrow{g(t) = \hat{A}f(t)}$$

$\hat{A}$ — линейный оператор; $\hat{A}\Psi_n(t) = \alpha_n \Psi_n(t)$

$$\hat{A}f(t) = \hat{A}\sum_n \widetilde{f}_n \Psi_n(t) = \sum_n \widetilde{f}_n \hat{A}\Psi_n(t) = \sum_n \widetilde{f}_n \alpha_n \Psi_n(t) = \sum_n \widetilde{g}_n \Psi_n(t) = g(t)$$

$$\widetilde{g}_n = \alpha_n \cdot \widetilde{f}_n$$

Линейные инвариантные по времени системы

Определения

Определение 16.2

Линейная система — *система, сигнал на выходе которой g связан с сигналом на ее входе f с помощью линейного оператора $\hat{A}$:*

$$g(t) = \hat{A}(\xi_1 f_1(t) + \xi_2 f_2(t)) = \xi_1 \underbrace{\hat{A} f_1(t)}_{g_1(t)} + \xi_2 \underbrace{\hat{A} f_2(t)}_{g_2(t)}$$

$$g[n] = \xi_1 \underbrace{\hat{A} f_1[n]}_{g_1[n]} + \xi_2 \underbrace{\hat{A} f_2[n]}_{g_2[n]}$$

Линейные инвариантные по времени системы

Определения

Определение 16.2

Линейная система — *система, сигнал на выходе которой g связан с сигналом на ее входе f с помощью линейного оператора $\hat{A}$:*

$$g(t) = \hat{A}(\xi_1 f_1(t) + \xi_2 f_2(t)) = \xi_1 \underbrace{\hat{A} f_1(t)}_{g_1(t)} + \xi_2 \underbrace{\hat{A} f_2(t)}_{g_2(t)}$$

$$g[n] = \xi_1 \underbrace{\hat{A} f_1[n]}_{g_1[n]} + \xi_2 \underbrace{\hat{A} f_2[n]}_{g_2[n]}$$

Определение 16.3

Инвариантная по времени система — *система, сигнал на выходе которой g связан с сигналом на ее входе f с помощью оператора $\hat{A}$, который не зависит от времени.*

Линейные инвариантные по времени системы

$$g(t) = \hat{A}f(t) =$$

$$= \hat{A} \int_{\tau=-\infty}^{\infty} f(\tau)\delta(t-\tau)d\tau =$$

Линейные инвариантные по времени системы

$$g(t) = \hat{A}f(t) =$$

$$= \hat{A} \int_{\tau=-\infty}^{\infty} f(\tau)\delta(t-\tau)d\tau =$$

$$= \int_{\tau=-\infty}^{\infty} f(\tau)\hat{A}\delta(t-\tau)d\tau =$$

Линейные инвариантные по времени системы

$$
\begin{aligned}
g(t) &= \hat{A} f(t) = \\
&= \hat{A} \int_{\tau=-\infty}^{\infty} f(\tau)\delta(t-\tau)d\tau = \\
&= \int_{\tau=-\infty}^{\infty} f(\tau)\hat{A}\delta(t-\tau)d\tau = \\
&= \int_{\tau=-\infty}^{\infty} f(\tau)h(t-\tau)d\tau =
\end{aligned}
$$

Линейные инвариантные по времени системы

$$
\begin{aligned}
g(t) &= \hat{A} f(t) = \\
&= \hat{A} \int_{\tau=-\infty}^{\infty} f(\tau)\delta(t-\tau)d\tau = \\
&= \int_{\tau=-\infty}^{\infty} f(\tau)\hat{A}\delta(t-\tau)d\tau = \\
&= \int_{\tau=-\infty}^{\infty} f(\tau)h(t-\tau)d\tau = \\
&= \int_{\tau=-\infty}^{\infty} h(\tau)f(t-\tau)d\tau
\end{aligned}
$$

Линейные инвариантные по времени системы

Определения

Определение 16.4

Сигнал g на выходе ЛИВС связан с сигналом f на ее входе с помощью ***импульсной функции отклика***

$h(\tau)$ *(для непрерывных сигналов):* $g(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau) f(t-\tau) d\tau$

или

$h[m]$ *(для дискретных сигналов):* $g[n] = \sum\limits_{m=-\infty}^{\infty} h[m] f[n-m]$

Линейные инвариантные по времени системы

Определения

Определение 16.4

Сигнал g на выходе ЛИВС связан с сигналом f на ее входе с помощью ***импульсной функции отклика***

$h(\tau)$ (для непрерывных сигналов): $g(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau) f(t-\tau) d\tau$

или

$h[m]$ (для дискретных сигналов): $g[n] = \sum\limits_{m=-\infty}^{\infty} h[m] f[n-m]$

Реакция системы на импульсное воздействие

$$g(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)\delta(t-\tau)d\tau = h(t)$$

$$g[n] = \sum\limits_{m=-\infty}^{\infty} h[m]\delta[n-m] = h[n]$$

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g(t) = \hat{A}f(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)f(t-\tau)d\tau$$

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g(t) = \hat{A}f(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)f(t-\tau)d\tau$$

Пусть $f(t) = z^t$. Тогда

$$g(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{t-\tau}d\tau = z^t \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau = f(t)\overbrace{\int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau}^{H(z)}$$

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g(t) = \hat{A}f(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)f(t-\tau)d\tau$$

Пусть $f(t) = z^t$. Тогда

$$g(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{t-\tau}d\tau = z^t \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau = f(t)\overbrace{\int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau}^{H(z)}$$

Определение 16.5

$H(z) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau$ — *непрерывная системная функция*

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g(t) = \hat{A}f(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)f(t-\tau)d\tau$$

Пусть $f(t) = z^t$. Тогда

$$g(t) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{t-\tau}d\tau = z^t \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau = f(t)\overbrace{\int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau}^{H(z)}$$

Определение 16.5

$H(z) = \int\limits_{\tau=-\infty}^{\infty} h(\tau)z^{-\tau}d\tau$ — *непрерывная системная функция*

$f(t) = z^t$ — собственные функции оператора $\hat{A}$
$H(z)$ — собственные значения оператора $\hat{A}$

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g[n] = \hat{A}f[n] = \sum_{m=-\infty}^{\infty} h[m]f[n-m]$$

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g[n] = \hat{A}f[n] = \sum_{m=-\infty}^{\infty} h[m]f[n-m]$$

Пусть $f[n] = z^n$. Тогда

$$g[n] = \sum_{m=-\infty}^{\infty} h[m]z^{n-m} = z^n \sum_{m=-\infty}^{\infty} h[m]z^{-m} = f[n] \overbrace{\sum_{m=-\infty}^{\infty} h[m]z^{-m}}^{H(z)}$$

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g[n] = \hat{A}f[n] = \sum_{m=-\infty}^{\infty} h[m]f[n-m]$$

Пусть $f[n] = z^n$. Тогда

$$g[n] = \sum_{m=-\infty}^{\infty} h[m]z^{n-m} = z^n \sum_{m=-\infty}^{\infty} h[m]z^{-m} = f[n] \overbrace{\sum_{m=-\infty}^{\infty} h[m]z^{-m}}^{H(z)}$$

Определение 16.6

$H(z) = \sum_{m=-\infty}^{\infty} h[m]z^{-m}$ — *дискретная системная функция*

Линейные инвариантные по времени системы

Собственные функции и собственные значения

$$g[n] = \hat{A}f[n] = \sum_{m=-\infty}^{\infty} h[m]f[n-m]$$

Пусть $f[n] = z^n$. Тогда

$$g[n] = \sum_{m=-\infty}^{\infty} h[m]z^{n-m} = z^n \sum_{m=-\infty}^{\infty} h[m]z^{-m} = f[n] \overbrace{\sum_{m=-\infty}^{\infty} h[m]z^{-m}}^{H(z)}$$

Определение 16.6

$H(z) = \sum_{m=-\infty}^{\infty} h[m]z^{-m}$ — *дискретная системная функция*

$f[n] = z^n$ — собственные функции оператора $\hat{A}$
$H(z)$ — собственные значения оператора $\hat{A}$

Z-преобразование

Прямое преобразование

$$\mathbb{Z}\{h[n]\} \equiv \sum_{n=-\infty}^{\infty} h[n] z^{-n}$$

Обратное преобразование

$$\mathbb{Z}^{-1}\{H(z)\} \equiv \frac{1}{2\pi i} \oint_C H(z) z^{n-1} dz$$

Контур C охватывает область сходимости $H(z)$ и содержит все полюсы $H(z)$

Z-преобразование

Прямое преобразование

$$\mathbb{Z}\{h[n]\} \equiv \sum_{n=-\infty}^{\infty} h[n]z^{-n}$$

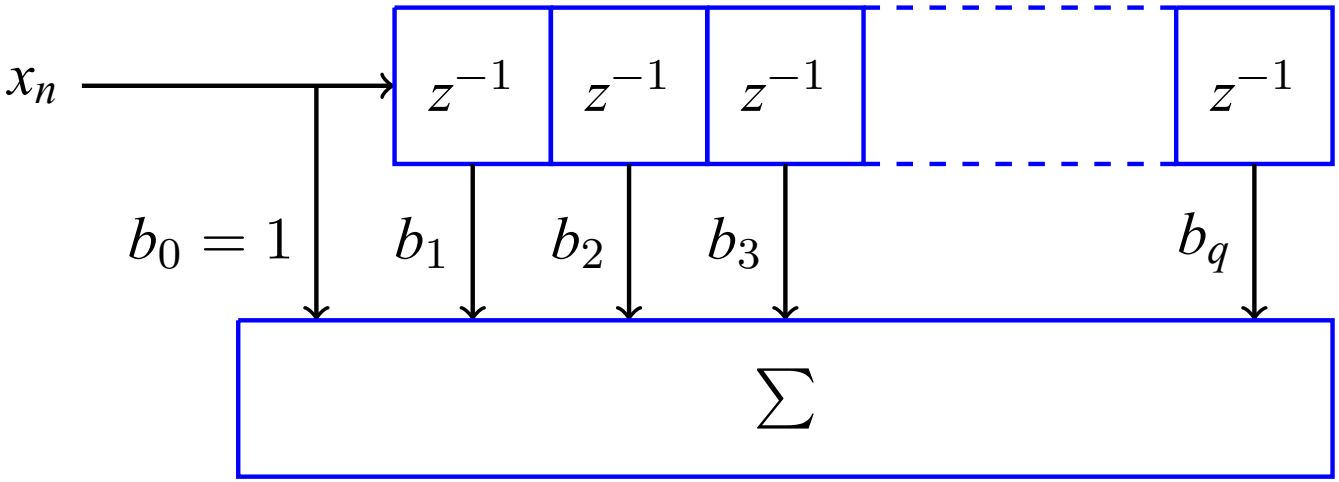
Обратное преобразование

$$\mathbb{Z}^{-1}\{H(z)\} \equiv \frac{1}{2\pi i}\oint_C H(z)z^{n-1}dz$$

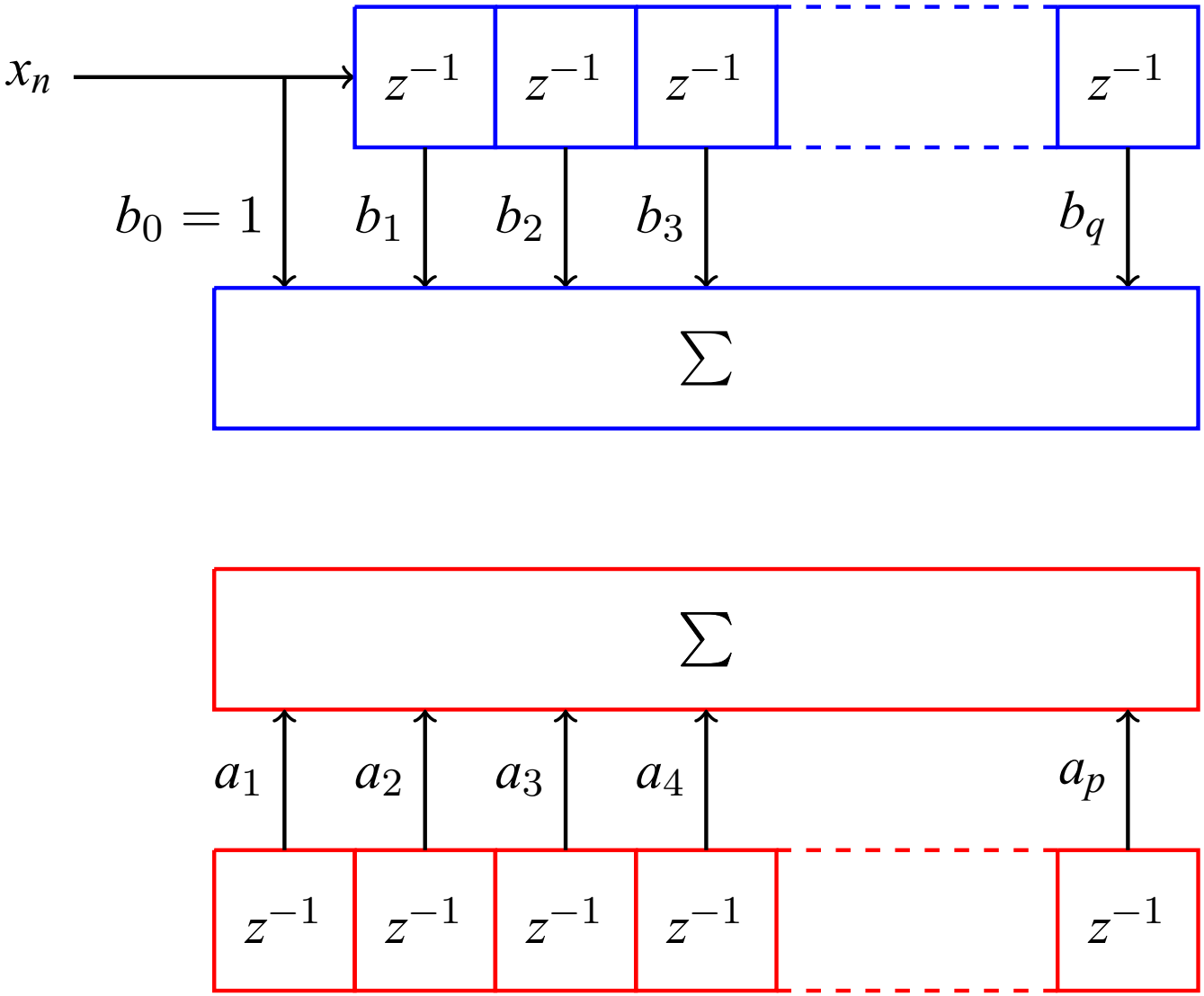
Контур C охватывает область сходимости $H(z)$ и содержит все полюсы $H(z)$

- Если положить $z = e^{i\omega t}$, то получается преобразование Фурье.
- Свойства Z-преобразования те же, что и у преобразования Фурье.
- z^{-1} – операция сдвига на 1 отсчет.

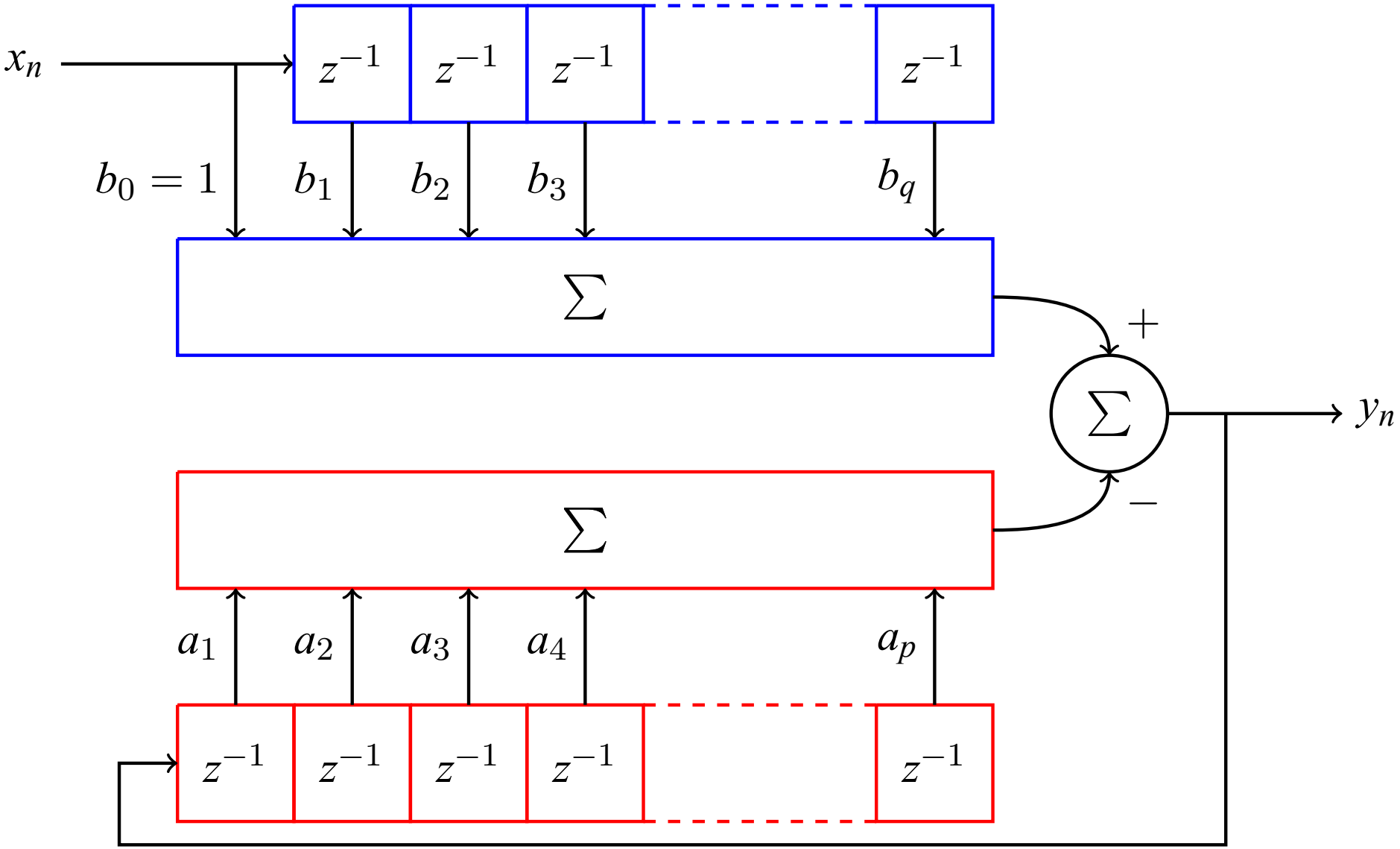
Модель APCC (ARMA)



Модель APCC (ARMA)



Модель АРСС (ARMA)



Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$\begin{cases} a[0] = 1 \\ \sum\limits_{k=0}^{q} b[k]x[n-k] = \sum\limits_{k=0}^{p} a[k]y[n-k] \end{cases}$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$\begin{cases} a[0] = 1 \\ \sum\limits_{k=0}^{q} b[k]x[n-k] = \sum\limits_{k=0}^{p} a[k]y[n-k] \end{cases}$$

$$\mathbb{Z}\left\{\sum_{k=0}^{q} b[k]x[n-k]\right\} = \sum_{n=-\infty}^{\infty} z^{-n} \sum_{k=0}^{q} b[k]x[n-k] =$$

$$= \sum_{k=0}^{q} b[k]z^{-k} \sum_{n=-\infty}^{\infty} z^{-(n-k)}x[n-k] = X(z)\sum_{k=0}^{q} b[k]z^{-k}$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$\begin{cases} a[0] = 1 \\ \sum\limits_{k=0}^{q} b[k]x[n-k] = \sum\limits_{k=0}^{p} a[k]y[n-k] \end{cases}$$

$$\mathbb{Z}\left\{\sum_{k=0}^{q} b[k]x[n-k]\right\} = \sum_{n=-\infty}^{\infty} z^{-n} \sum_{k=0}^{q} b[k]x[n-k] =$$

$$= \sum_{k=0}^{q} b[k]z^{-k} \sum_{n=-\infty}^{\infty} z^{-(n-k)}x[n-k] = X(z)\sum_{k=0}^{q} b[k]z^{-k}$$

$$\mathbb{Z}\left\{\sum_{k=0}^{p} a[k]y[n-k]\right\} = Y(z)\sum_{k=0}^{p} a[k]z^{-k}$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$X(z)\sum_{k=0}^{q} b[k]z^{-k} = Y(z)\sum_{k=0}^{p} a[k]z^{-k}$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$X(z)\sum_{k=0}^{q} b[k]z^{-k} = Y(z)\sum_{k=0}^{p} a[k]z^{-k}$$

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum_{k=0}^{q} b[k]z^{-k}}{1 + \sum_{k=1}^{p} a[k]z^{-k}}$$

Модель АРСС (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$X(z)\sum_{k=0}^{q} b[k]z^{-k} = Y(z)\sum_{k=0}^{p} a[k]z^{-k}$$

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1+\sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1-z_k/z)}{\prod\limits_{k=1}^{p}(1-p_k/z)}$$

Модель APCC (ARMA)

$$ARMA(p,q): \quad \sum_{k=0}^{q} b[k]x[n-k] - \sum_{k=1}^{p} a[k]y[n-k] = y[n]$$

$$X(z)\sum_{k=0}^{q} b[k]z^{-k} = Y(z)\sum_{k=0}^{p} a[k]z^{-k}$$

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1+\sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1-z_k/z)}{\prod\limits_{k=1}^{p}(1-p_k/z)}$$

- $b[0]$ – масштаб усиления; z_k – нули; p_k – полюсы функции $H(z)$
- $AR(p) = MA(\infty); \quad MA(q) = AR(\infty)$

Модель APCC (ARMA) для построения прогноза

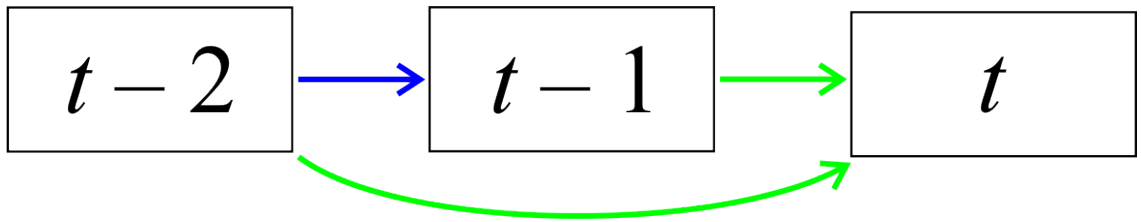
Модель $ARMA(p, q)$ позволяет получить прогноз:

$$\hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-k] - \sum_{k=1}^{p} a[k]y[n-k]; \quad \epsilon[n] = y[n] - \hat{y}[n]$$

Модель АРСС (ARMA) для построения прогноза

Влияние временных отсчетов на прогноз

- Непосредственное влияние.
- Влияние через другие отсчеты.



Автокорреляционная функция (ACF)

Корреляционная функция

- Если $y_{1,2}[n]$ имеют в момент n среднее $\mu_{1,2}[n]$ и дисперсию $\sigma^2_{1,2}[n]$,

$$Corr(y_1[n], y_2[m]) = \frac{\mathcal{M}\left\{(y_1[n] - \mu_1[n])(y_2[m] - \mu_2[m])\right\}}{\sigma_1[n]\sigma_2[m]}$$

Автокорреляционная функция (ACF)

Корреляционная функция

- Если $y_{1,2}[n]$ имеют в момент n среднее $\mu_{1,2}[n]$ и дисперсию $\sigma^2_{1,2}[n]$,

$$Corr(y_1[n], y_2[m]) = \frac{\mathcal{M}\left\{(y_1[n] - \mu_1[n])(y_2[m] - \mu_2[m])\right\}}{\sigma_1[n]\sigma_2[m]}$$

Автокорреляционная функция

- Если $y[n]$ имеет в момент n среднее $\mu[n]$ и дисперсию $\sigma^2[n]$,

$$ACF(n, m) = Corr(y[n], y[m]) = \frac{\mathcal{M}\left\{(y[n] - \mu[n])(y[m] - \mu[m])\right\}}{\sigma[n]\sigma[m]}$$

Автокорреляционная функция (ACF)

Корреляционная функция

- Если $y_{1,2}[n]$ имеют в момент n среднее $\mu_{1,2}[n]$ и дисперсию $\sigma^2_{1,2}[n]$,

$$Corr(y_1[n], y_2[m]) = \frac{\mathcal{M}\left\{(y_1[n] - \mu_1[n])(y_2[m] - \mu_2[m])\right\}}{\sigma_1[n]\sigma_2[m]}$$

Автокорреляционная функция

- Если $y[n]$ имеет в момент n среднее $\mu[n]$ и дисперсию $\sigma^2[n]$,

$$ACF(n, m) = Corr(y[n], y[m]) = \frac{\mathcal{M}\left\{(y[n] - \mu[n])(y[m] - \mu[m])\right\}}{\sigma[n]\sigma[m]}$$

- Если процесс стационарный, то

$$ACF(t) = Corr(y[n + t], y[n]) = \frac{\mathcal{M}\left\{(y[n + t] - \mu)(y[n] - \mu)\right\}}{\sigma^2}$$

Частичная автокорреляционная функция (PACF)

$$PACF(t) = \begin{cases} Corr(y[n+t], y[n]) & t = 1; \\ Corr(y[n+t] - y^{(t-1)}[n+t], y[n] - y^{(t-1)}[n]) & t > 1. \end{cases}$$

Корректировки в виде линейной регрессии на $y[n+1], y[n+2], \ldots, y[n+t-1]$:

$$\begin{cases} y^{(t-1)}[n] = \beta_1 y[n+1] + \beta_2 y[n+2] + \cdots + \beta_{t-1} y[n+t-1]; \\ y^{(t-1)}[n+t] = \beta_1 y[n+t-1] + \beta_2 y[n+t-2] + \cdots + \beta_{t-1} y[n+1]. \end{cases}$$

Модель APCC (ARMA) для построения прогноза

Сравнение ACF и PACF. Модель AR(1)

$$y[n] = ay[n-1] + \epsilon[n]$$

Модель APCC (ARMA) для построения прогноза

Сравнение ACF и PACF. Модель AR(1)

$$y[n] = ay[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

Модель APCC (ARMA) для построения прогноза

Сравнение ACF и PACF. Модель AR(1)

$$y[n] = ay[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

$$y[n] = ay[n-1] + \epsilon[n]$$

Модель APCC (ARMA) для построения прогноза

Сравнение ACF и PACF. Модель AR(1)

$$y[n] = ay[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

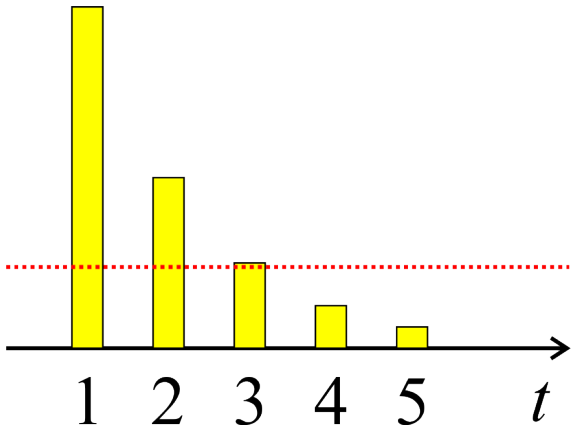
$$y[n] = ay[n-1] + \epsilon[n] = a^2 y[n-2] + a\epsilon[n-1]\epsilon[n] = \dots$$

Модель APCC (ARMA) для построения прогноза

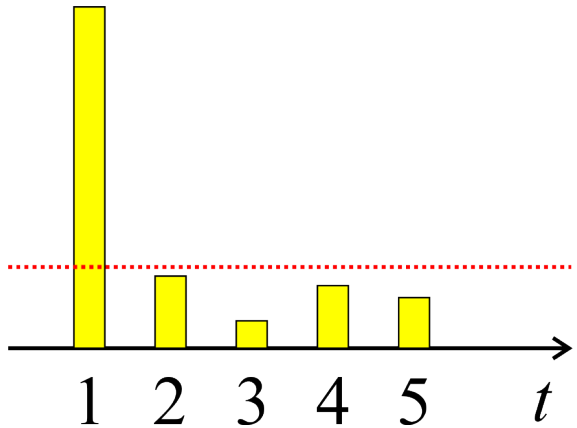
Сравнение ACF и PACF. Модель AR(1)

$$y[n] = ay[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

$$y[n] = ay[n-1] + \epsilon[n] = a^2 y[n-2] + a\epsilon[n-1]\epsilon[n] = \dots$$



ACF



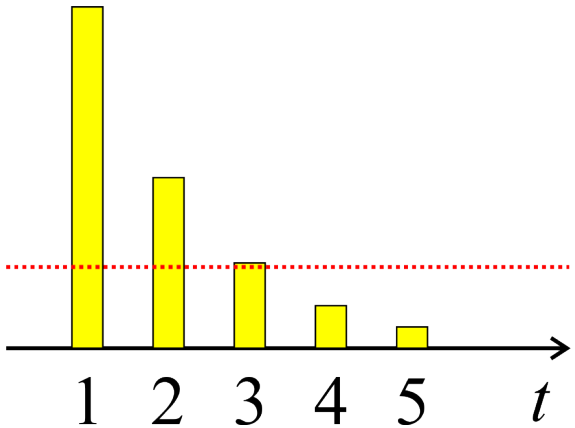
PACF

Модель APCC (ARMA) для построения прогноза

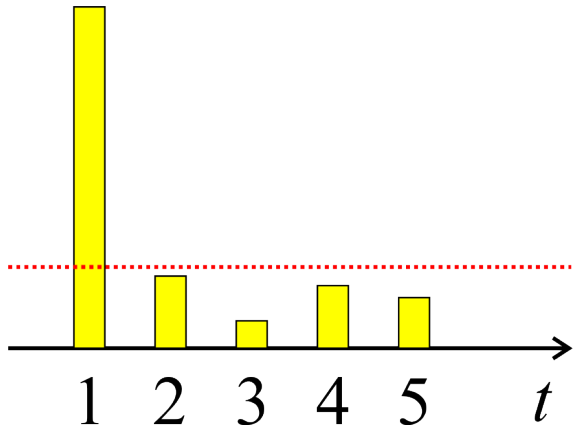
Сравнение ACF и PACF. Модель AR(1)

$$y[n] = ay[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

$$y[n] = ay[n-1] + \epsilon[n] = a^2 y[n-2] + a\epsilon[n-1]\epsilon[n] = \dots$$



ACF



PACF

Для моделей $AR(p)$ характерно
плавное спадание ACF и отсечка $PACF$ на лаге p.

Модель APCC (ARMA) для построения прогноза

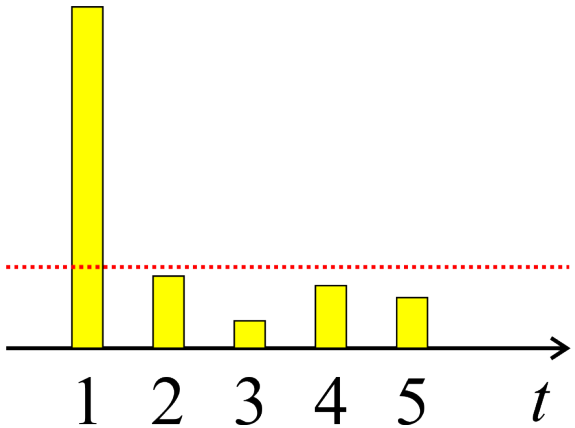
Сравнение ACF и PACF. Модель MA(1)

$$y[n] = a\epsilon[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

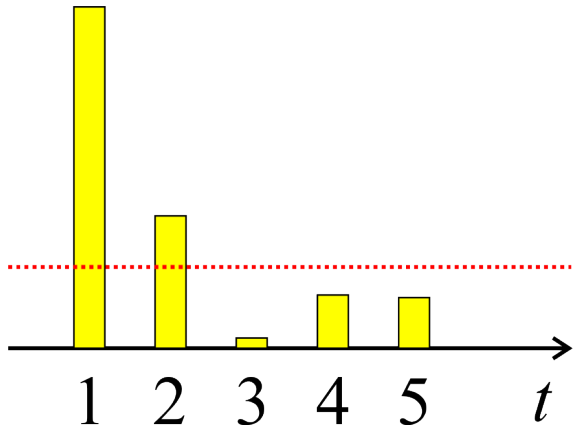
Модель APCC (ARMA) для построения прогноза

Сравнение ACF и PACF. Модель MA(1)

$$y[n] = a\epsilon[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$



ACF

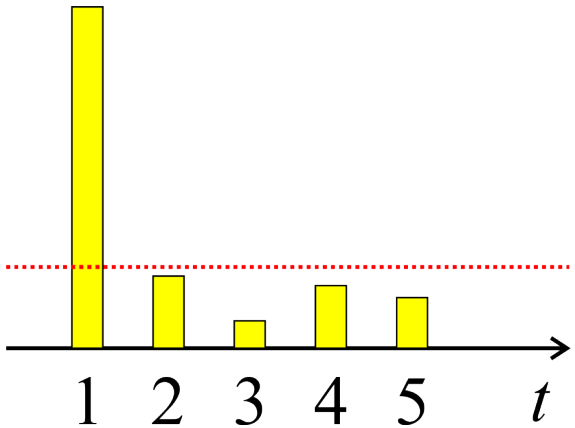


PACF

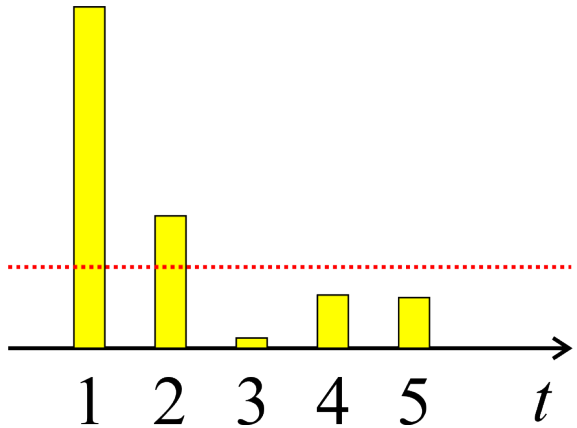
Модель APCC (ARMA) для построения прогноза

Сравнение ACF и PACF. Модель MA(1)

$$y[n] = a\epsilon[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$



ACF



PACF

Для моделей $MA(1)$ характерно
плавное спадание $PACF$ и отсечка ACF на лаге q.

Модель AR(1) и стационарность процесса

$$AR(1): \quad y[n] = ay[n-1] + \epsilon[n]$$

Модель AR(1) и стационарность процесса

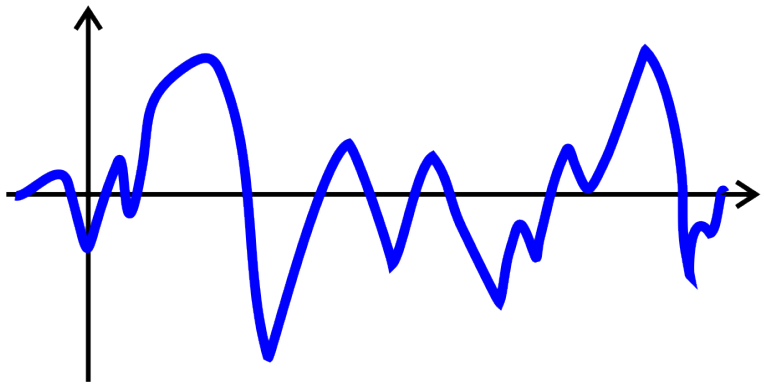
$$AR(1): \quad y[n] = ay[n-1] + \epsilon[n]; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

- $y[n] = ay[n-1] + \epsilon[n] = a^n y[0] + \sum\limits_{k=0}^{n-1} a^k \epsilon[n-k]$
- $$\begin{cases} \mathcal{M}\{y[n]\} = a\mathcal{M}\{y[n-1]\} = \cdots = a^n y[0] \\ \mathcal{D}\{y[n]\} = \sigma^2 \sum\limits_{k=0}^{n-1} a^{2k} = \sigma^2 \dfrac{1-a^{2(n-1)}}{1-a} \end{cases}$$

Модель AR(1) и стационарность процесса

$$\text{Если } |a| > 1, \text{ то } \begin{cases} \mathcal{M}\{|y[n]|\} = |a^n y[0]| \to \infty; \\ \mathcal{D}\{y[n]\} = \sigma^2 \dfrac{1 - a^{2(n-1)}}{1 - a} \to \infty \end{cases}$$

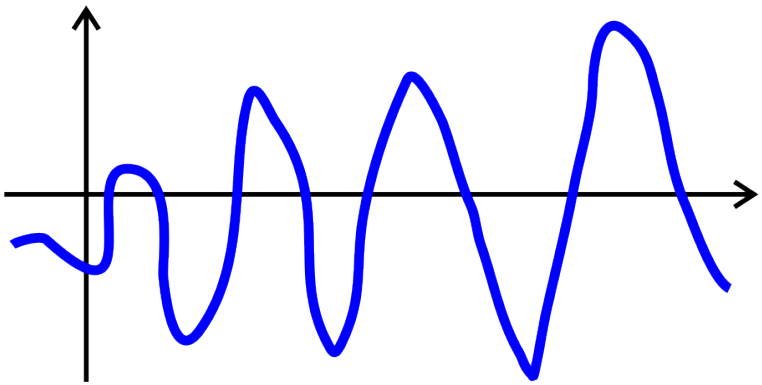
$$\Rightarrow \quad \text{процесс нестационарен}$$



Модель AR(1) и стационарность процесса

$$\text{Если } |a| < 1, \text{ то } \begin{cases} \mathcal{M}\{|y[n]|\} = |a^n y[0]| \to 0; \\ \mathcal{D}\{y[n]\} = \sigma^2 \dfrac{1 - a^{2(n-1)}}{1 - a} = \dfrac{\sigma^2}{1 - a} \end{cases}$$

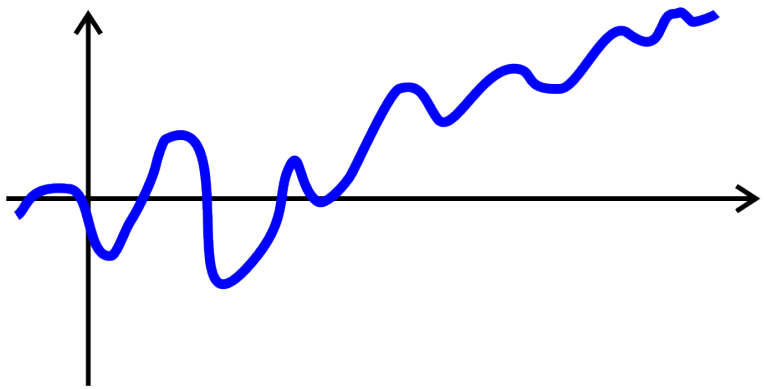
$$\Rightarrow \quad \text{процесс стационарен}$$



Модель AR(1) и стационарность процесса

$$\text{Если } |a| = 1, \text{ то } \begin{cases} \mathcal{M}\{|y[n]|\} = |a^n y[0]| = |ay[0]|; \\ \mathcal{D}\{y[n]\} = \sigma^2 \sum\limits_{k=0}^{n-1} |a|^{2k} = \sigma^2 \cdot n \to \infty \end{cases}$$

$\Rightarrow$ процесс нестационарен, существует единичный корень



Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$
- Если $|p_k| = 1$ – единичный корень, то $\exists\omega : z = p_k = e^{i\omega t}$.

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$
- Если $|p_k| = 1$ – единичный корень, то $\exists\omega : z = p_k = e^{i\omega t}$.
- Процесс нестационарный, т.к. спектральная компонента ω растет.

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$
- Если $|p_k| = 1$ – единичный корень, то $\exists\omega : z = p_k = e^{i\omega t}$.
- Процесс нестационарный, т.к. спектральная компонента ω растет.

- Пусть имеется d единичных корней, тогда

$$1 + \sum_{k=1}^{p} a[k]z^{-k} = \left(1 + \sum_{k=1}^{p-d} a'[k]z^{-k}\right)\left(1 - z^{-1}\right)^d$$

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$
- Если $|p_k| = 1$ – единичный корень, то $\exists\omega : z = p_k = e^{i\omega t}$.
- Процесс нестационарный, т.к. спектральная компонента ω растет.

- Пусть имеется d единичных корней, тогда

$$1 + \sum_{k=1}^{p} a[k]z^{-k} = \left(1 + \sum_{k=1}^{p-d} a'[k]z^{-k}\right)\left(1 - z^{-1}\right)^d = X'(z)\left(\underbrace{1 - z^{-1}}_{\Delta}\right)^d$$

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$
- Если $|p_k| = 1$ – единичный корень, то $\exists\omega : z = p_k = e^{i\omega t}$.
- Процесс нестационарный, т.к. спектральная компонента ω растет.

- Пусть имеется d единичных корней, тогда

$$1 + \sum_{k=1}^{p} a[k]z^{-k} = \left(1 + \sum_{k=1}^{p-d} a'[k]z^{-k}\right)\left(1 - z^{-1}\right)^d = X'(z)\left(\underbrace{1 - z^{-1}}_{\Delta}\right)^d$$

- $\Delta = 1 - z^{-1}$ – разностный оператор.

Единичные корни

$$H(z) = \frac{Y(x)}{X(z)} = \frac{\sum\limits_{k=0}^{q} b[k]z^{-k}}{1 + \sum\limits_{k=1}^{p} a[k]z^{-k}} = b[0]\frac{\prod\limits_{k=0}^{q}(1 - z_k/z)}{\prod\limits_{k=1}^{p}(1 - p_k/z)}$$

- При $z = p_k$ получается $X(z) = 0; H(z) = \infty$
- Если $|p_k| = 1$ – единичный корень, то $\exists\omega : z = p_k = e^{i\omega t}$.
- Процесс нестационарный, т.к. спектральная компонента ω растет.

- Пусть имеется d единичных корней, тогда

$$1 + \sum_{k=1}^{p} a[k]z^{-k} = \left(1 + \sum_{k=1}^{p-d} a'[k]z^{-k}\right)\left(1 - z^{-1}\right)^d = X'(z)\left(\underbrace{1 - z^{-1}}_{\Delta}\right)^d$$

- $\Delta = 1 - z^{-1}$ – разностный оператор.
- Дифференцирование позволяет избавиться от единичных корней.

Модель АРПСС (ARIMA) для построения прогноза

Модель $ARIMA(p, d, q)$ позволяет получить прогноз:

$$\Delta^d \hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-k] - \sum_{k=1}^{p} a[k]\Delta^d y[n-k]; \quad \epsilon[n] = y[n] - \hat{y}[n]$$

Модель АРПСС (ARIMA) для построения прогноза

Модель $ARIMA(p, d, q)$ позволяет получить прогноз:

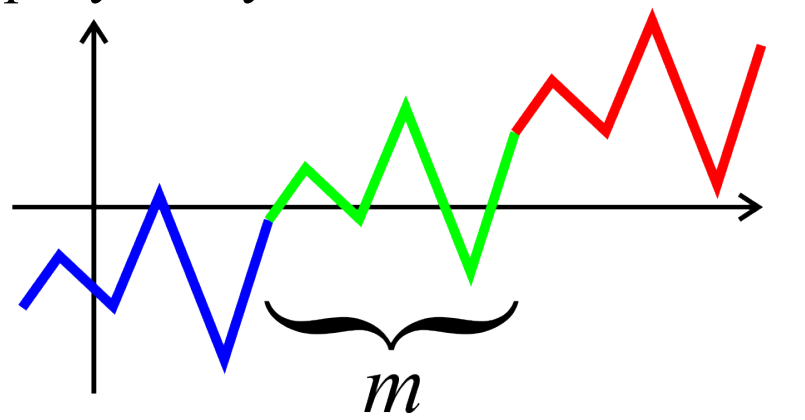
$$\Delta^d \hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-k] - \sum_{k=1}^{p} a[k]\Delta^d y[n-k]; \quad \epsilon[n] = y[n] - \hat{y}[n]$$

Модель *ARFIMA* – обобщение на случай дробного дифференцирования

$$\Delta^d = \left(1 - z^{-1}\right)^d = \sum_{k=0}^{\infty} \prod_{j=0}^{k-1} (d-j) \frac{(-1)^k}{k!} z^{-k}$$

Модель САРПСС (SARIMA) для построения прогноза

Требуется учитывать сезонность



Модель САРПСС (SARIMA) для построения прогноза

Модель $ARIMA(p, d, q)$:

$$\Delta^d \hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-k] - \sum_{k=1}^{p} a[k]\Delta^d y[n-k]$$

Учитываются лаги $1, 2, \ldots, p$ и $1, 2, \ldots, q$.

Модель САРПСС (SARIMA) для построения прогноза

Та же модель $ARIMA(p, d, q)_1$ с параметром разрежения $m = 1$:

$$\Delta^d \hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-k] - \sum_{k=1}^{p} a[k]\Delta^d y[n-k]$$

Учитываются лаги $1, 2, \ldots, p$ и $1, 2, \ldots, q$.

Модель САРПСС (SARIMA) для построения прогноза

Та же модель $ARIMA(p, d, q)_1$ с параметром разрежения $m = 1$:

$$\Delta^d \hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-k] - \sum_{k=1}^{p} a[k]\Delta^d y[n-k]$$

Учитываются лаги $1, 2, \ldots, p$ и $1, 2, \ldots, q$.

Модель $ARIMA(p, d, q)_m$ с параметром разрежения m:

$$\Delta^d \hat{y}[n] = c + \sum_{k=1}^{q} b[k]\epsilon[n-km] - \sum_{k=1}^{p} a[k]\Delta^d y[n-km]$$

Учитываются лаги $m, 2m, \ldots, pm$ и $1, 2, \ldots, qm$.

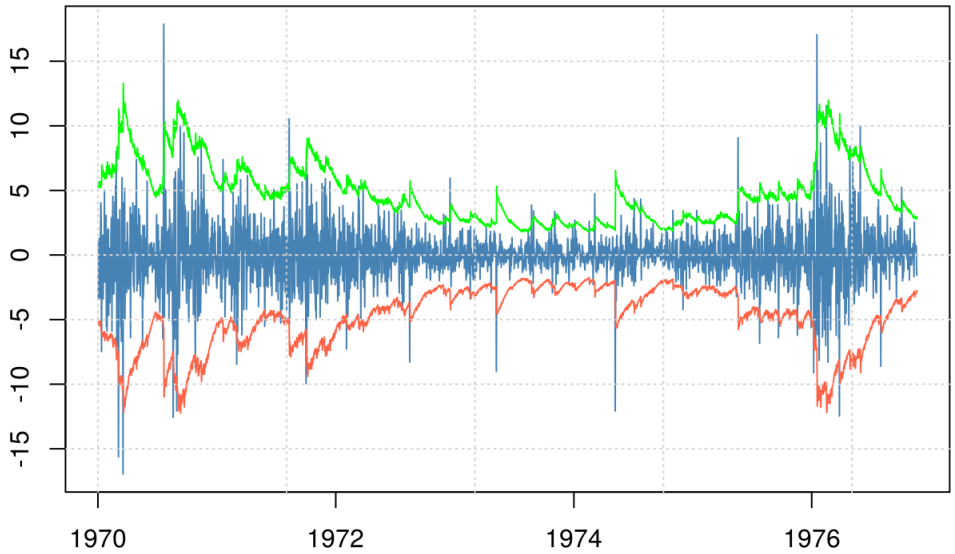
Модель САРПСС (SARIMA) для построения прогноза

Модель $SARIMA(p,d,q)\underbrace{(P,D,Q)_m}_{\text{Сезонность}}$

- Учитываются лаги $1,2,\ldots,p$ и $1,2,\ldots,q$.
- Учитываются лаги $m,2m,\ldots,Pm$ и $1,2,\ldots,Qm$.

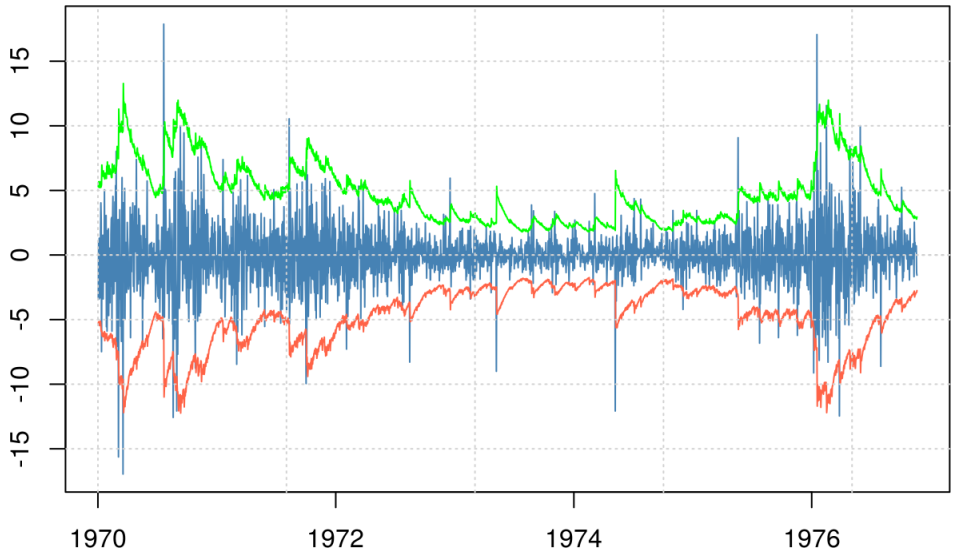
Модель АвтоРегрессионной Условной Гетероскедастичности (ARCH, GARCH)

Волатильность актива меняется с течением времени



Модель АвтоРегрессионной Условной Гетероскедастичности (ARCH, GARCH)

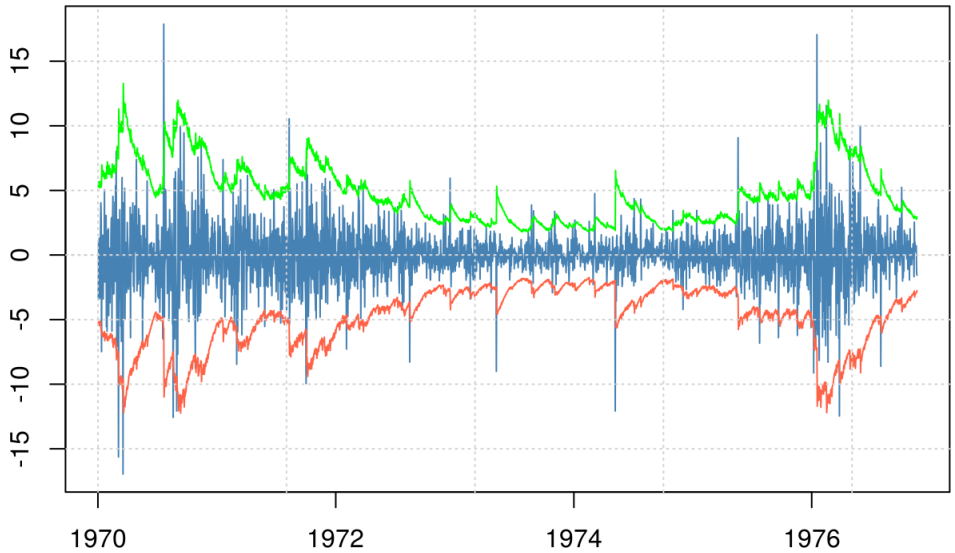
Волатильность актива меняется с течением времени



$$ARCH(q) : u[n] = \epsilon[n]\sqrt{\alpha[0] + \sum_{k=1}^{q} \alpha[k]u^2[n-k]}; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

Модель АвтоРегрессионной Условной Гетероскедастичности (ARCH, GARCH)

Волатильность актива меняется с течением времени



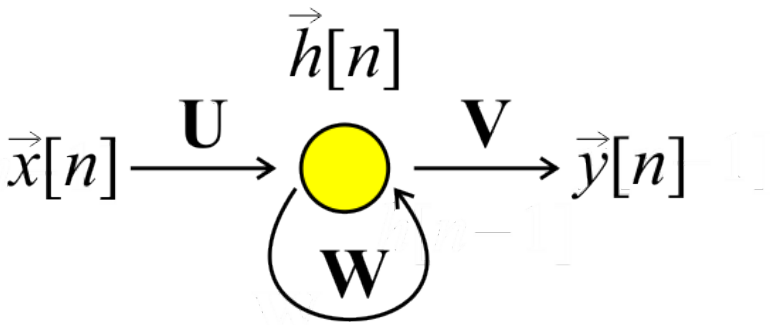
$$ARCH(q) : u[n] = \epsilon[n]\sqrt{\alpha[0] + \sum_{k=1}^{q} \alpha[k]u^2[n-k]}; \quad \epsilon[n] \sim \mathcal{N}(0, \sigma^2)$$

$$GARCH(p,q) : u[n] = \epsilon[n]\sqrt{\alpha[0] + \sum_{k=1}^{q} \alpha[k]u^2[n-k] + \sum_{k=1}^{p} \beta[k]\sigma^2[n-k]}$$

Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

Идея: сохранить некоторое состояние с предыдущего шага и передать его на следующий шаг.

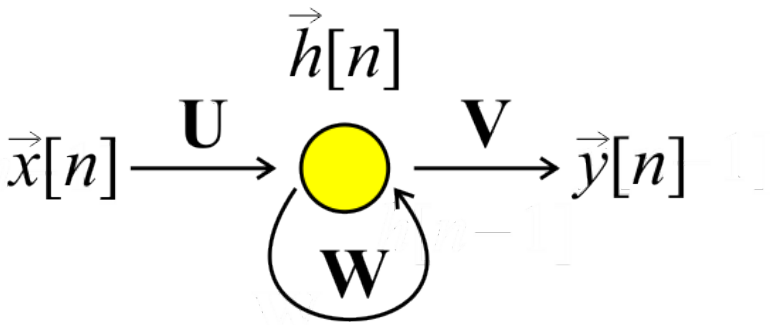


- $\vec{x}[n]$ – входные данные;
- $\vec{y}[n]$ – выходные данные;
- $\vec{h}[n]$ – внутреннее состояние.

Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

Идея: сохранить некоторое состояние с предыдущего шага и передать его на следующий шаг.



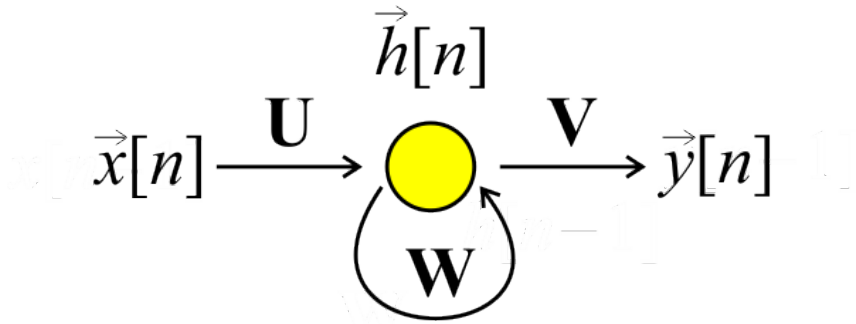
- $\vec{x}[n]$ – входные данные;
- $\vec{y}[n]$ – выходные данные;
- $\vec{h}[n]$ – внутреннее состояние.

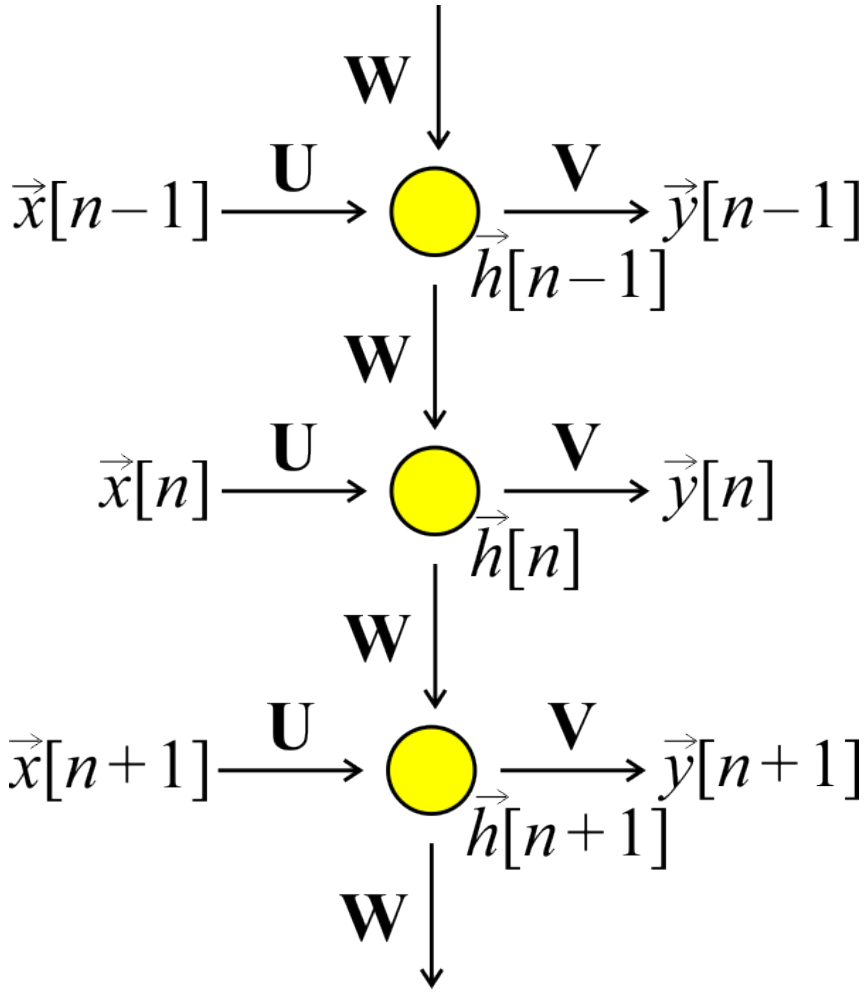
$$\begin{cases} \vec{h}[n] = \vec{\sigma}_h(\mathbf{U}\vec{x}[n] + \mathbf{W}\vec{h}[n-1]); \\ \vec{y}[n] = \vec{\sigma}_y(\mathbf{V}\vec{h}[n]) \end{cases}$$

Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

Нейронная сеть двуслойная, но она “разворачивается” в многослойную цепочку



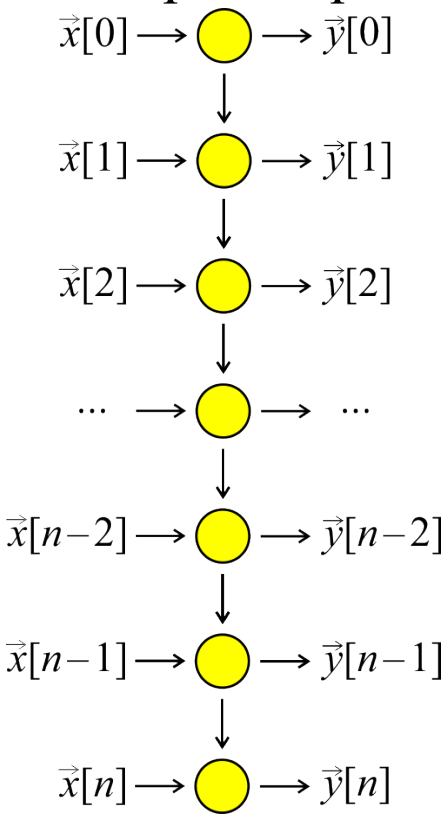


Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

Архитектуры для разных задач

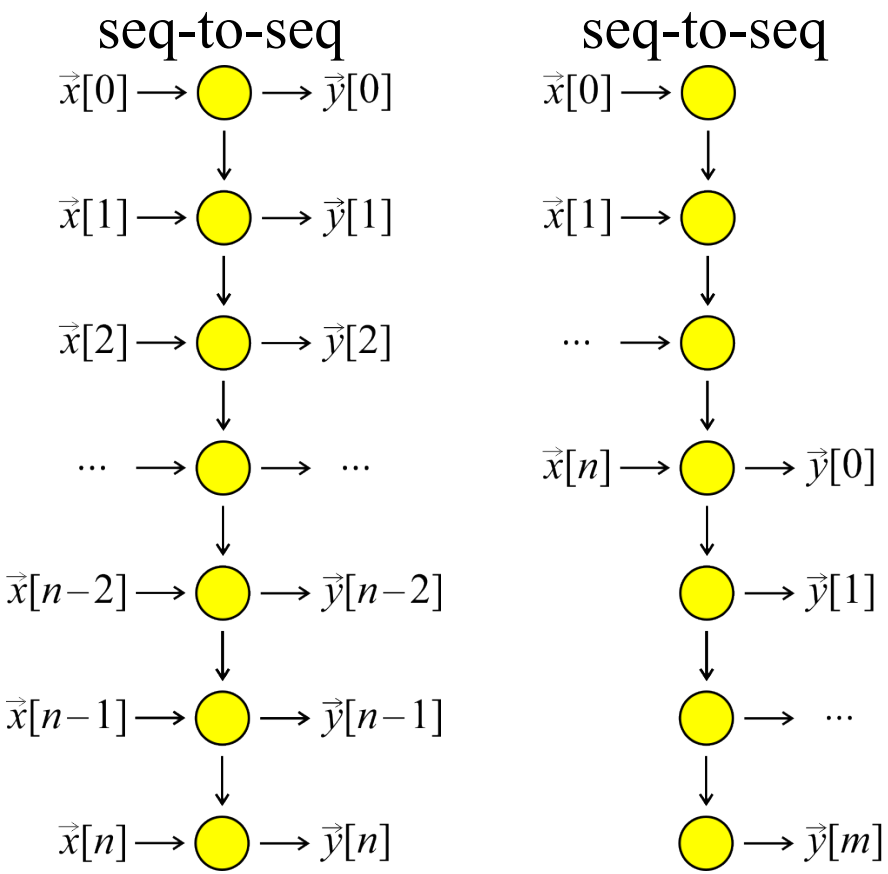
seq-to-seq



Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

Архитектуры для разных задач

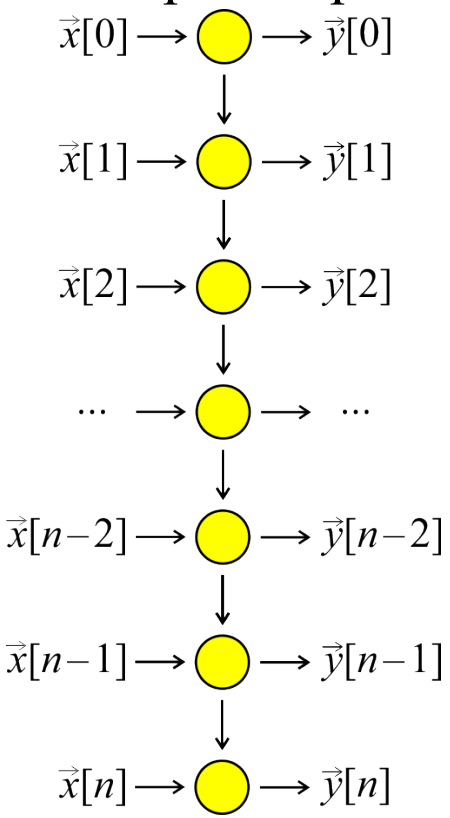


Нейросетевые методы прогнозирования

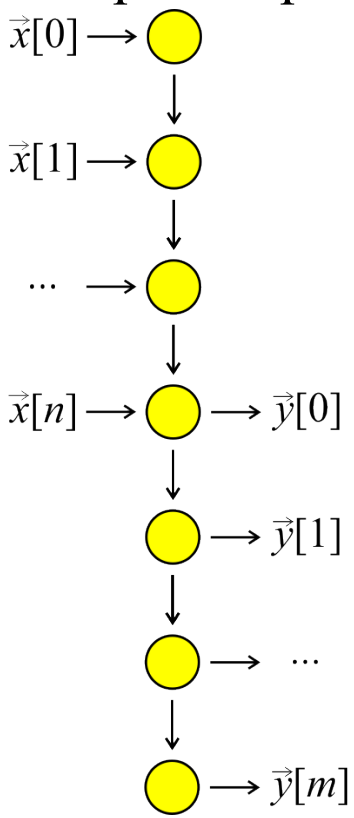
Рекуррентные нейронные сети (RNN)

Архитектуры для разных задач

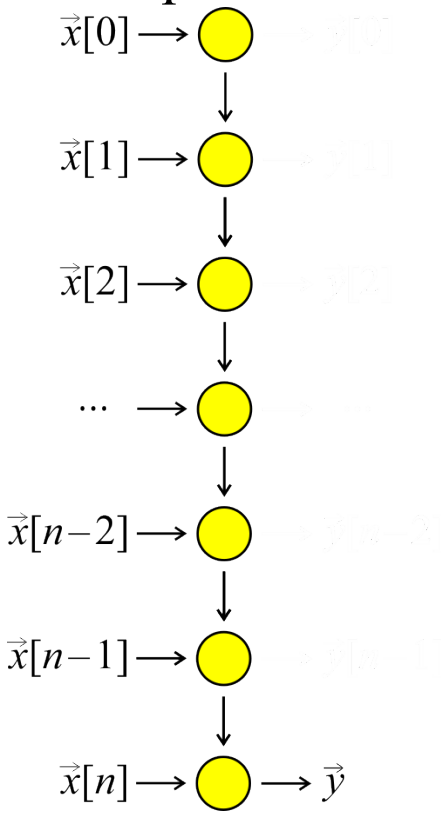
seq-to-seq



seq-to-seq



seq-to-one

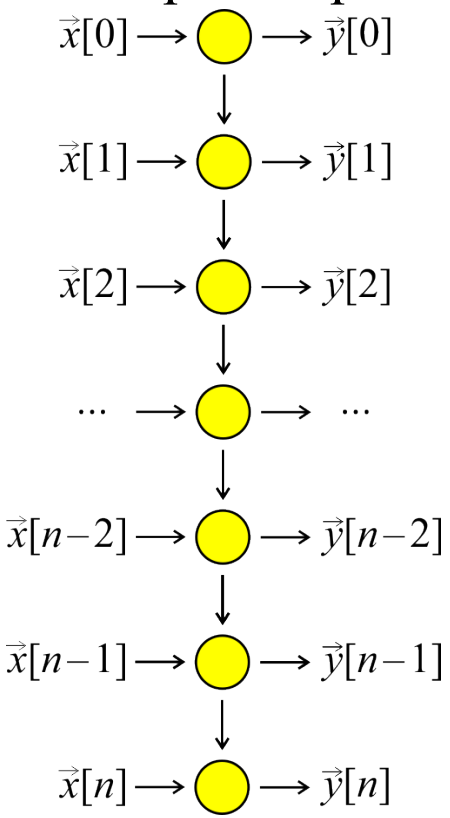


Нейросетевые методы прогнозирования

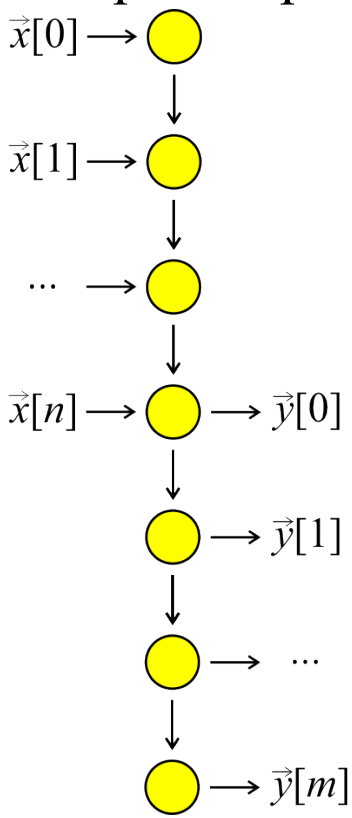
Рекуррентные нейронные сети (RNN)

Архитектуры для разных задач

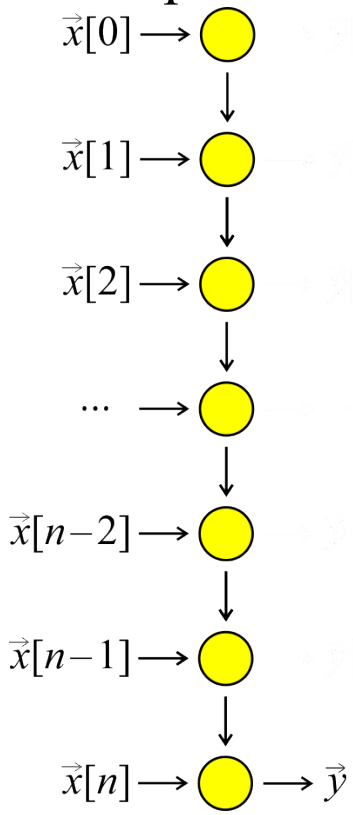
seq-to-seq



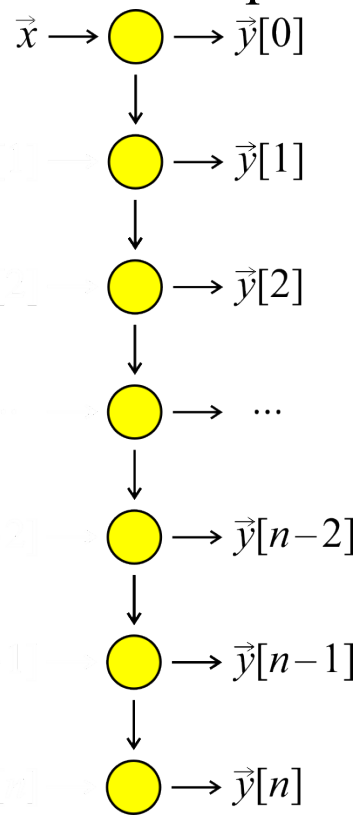
seq-to-seq



seq-to-one



one-to-seq



Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

- Методы типа SGD не работают из-за упорядоченности данных.

Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

- Методы типа SGD не работают из-за упорядоченности данных.
- Метод обратного распространения ошибки во времени (BPTT)

$$\frac{\partial \mathcal{L}[n]}{\partial W} = \frac{\partial \mathcal{L}[n]}{\partial y[n]} \frac{\partial y[n]}{\partial h[n]} \sum_{k=0}^{n} \left(\prod_{m=k+1}^{n} \frac{\partial h[m]}{\partial h[m-1]} \right) \frac{\partial h[k]}{\partial W}$$

Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

- Методы типа SGD не работают из-за упорядоченности данных.
- Метод обратного распространения ошибки во времени (BPTT)

$$\frac{\partial \mathcal{L}[n]}{\partial W} = \frac{\partial \mathcal{L}[n]}{\partial y[n]} \frac{\partial y[n]}{\partial h[n]} \sum_{k=0}^{n} \left(\prod_{m=k+1}^{n} \frac{\partial h[m]}{\partial h[m-1]} \right) \frac{\partial h[k]}{\partial W}$$

- Необходимо ввести ограничение на глубину истории.

Нейросетевые методы прогнозирования

Рекуррентные нейронные сети (RNN)

- Методы типа SGD не работают из-за упорядоченности данных.
- Метод обратного распространения ошибки во времени (BPTT)

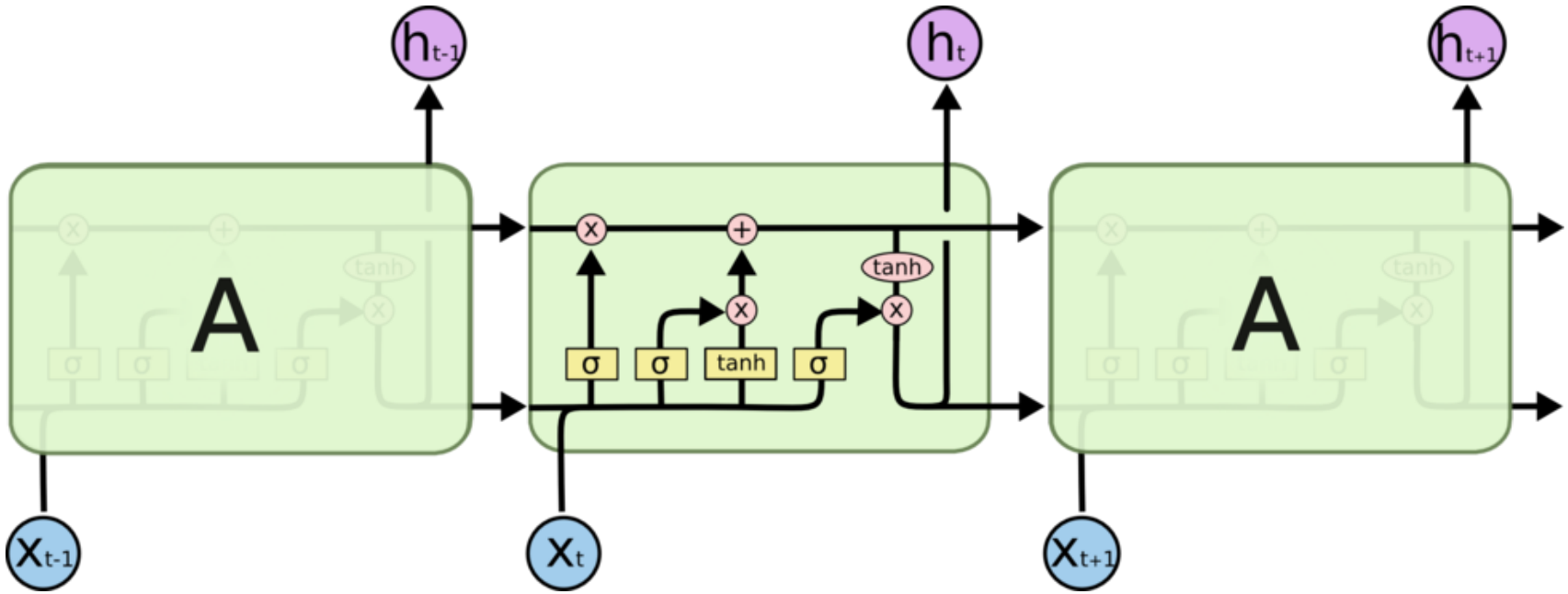
$$\frac{\partial \mathcal{L}[n]}{\partial W} = \frac{\partial \mathcal{L}[n]}{\partial y[n]} \frac{\partial y[n]}{\partial h[n]} \sum_{k=0}^{n} \left(\prod_{m=k+1}^{n} \frac{\partial h[m]}{\partial h[m-1]} \right) \frac{\partial h[k]}{\partial W}$$

- Необходимо ввести ограничение на глубину истории.
- Необходимо, чтобы $\frac{\partial h[m]}{\partial h[m-1]} \to 1$,
иначе возникает взрыв или затухание градиента.

Нейросетевые методы прогнозирования

Сеть долгой краткосрочной памяти (LSTM)

Идея: сеть должна достаточно долго помнить контекст. С этой целью функция активации внутри реккурентных блоков не используется.

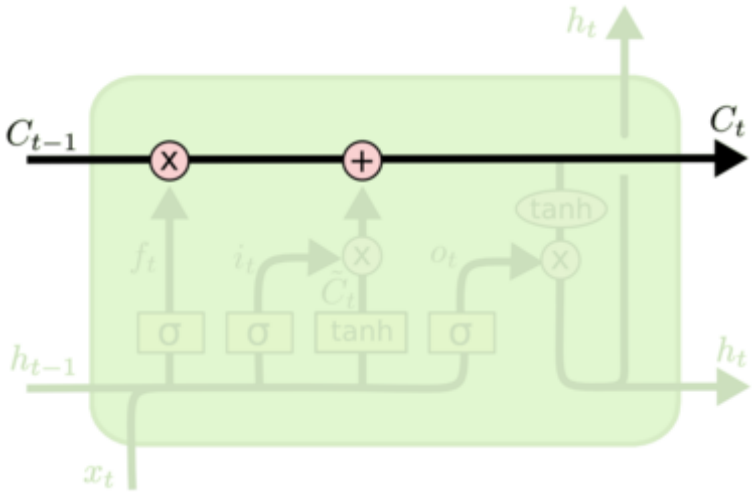


https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

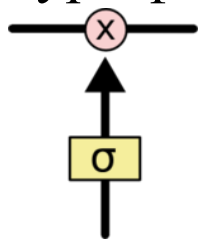
Нейросетевые методы прогнозирования

Сеть долгой краткосрочной памяти (LSTM)

Состояние ячейки изменяется с помощью фильтров.



Структура фильтра



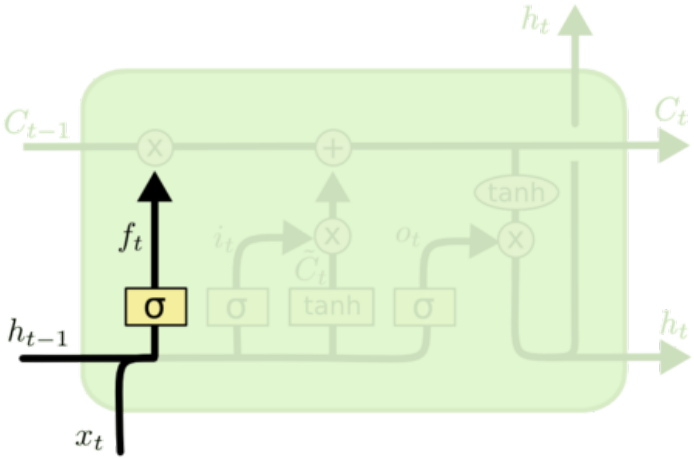
Значение сигмоиды лежит в диапазоне $[0; 1]$ и определяет долю “сигнала” на выходе фильтра.

https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

Нейросетевые методы прогнозирования

Сеть долгой краткосрочной памяти (LSTM)

Фильтр забывания (forget gate)



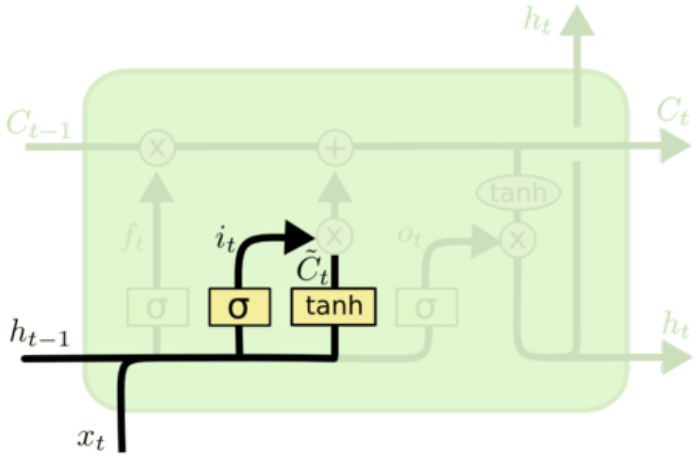
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

Нейросетевые методы прогнозирования

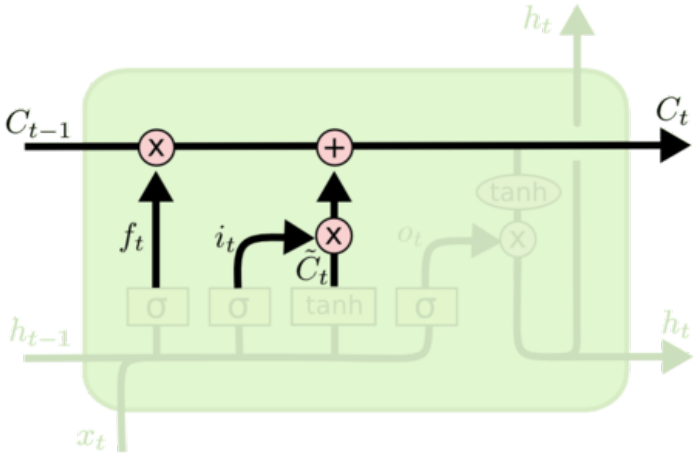
Сеть долгой краткосрочной памяти (LSTM)

Входной фильтр (input gate)



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$



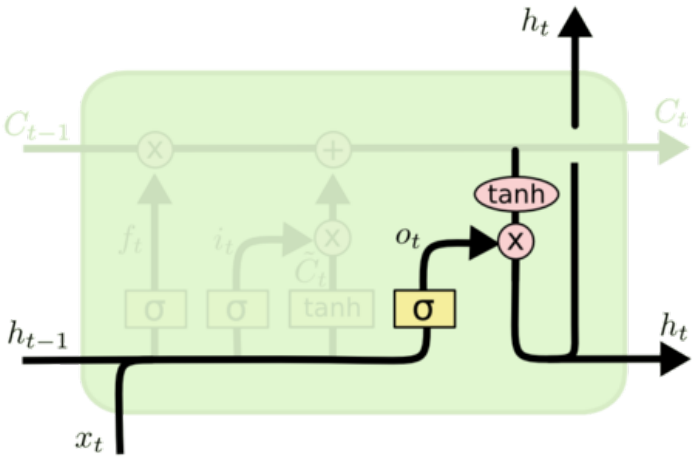
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

Нейросетевые методы прогнозирования

Сеть долгой краткосрочной памяти (LSTM)

Выходной фильтр (output gate)



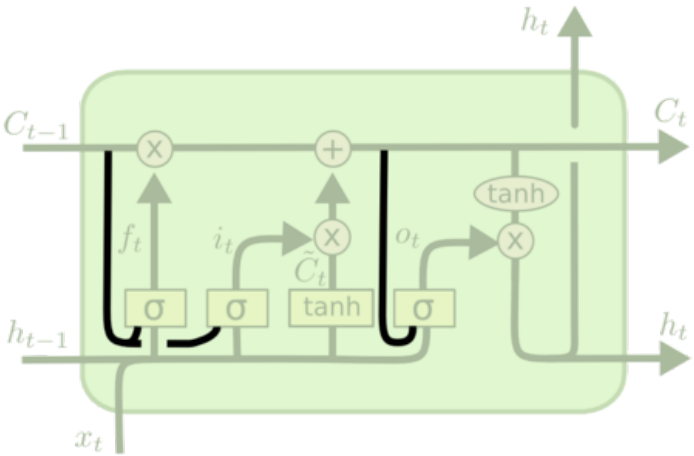
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

Нейросетевые методы прогнозирования

Сеть долгой краткосрочной памяти (LSTM) со смотровыми глазками (peephole conections)

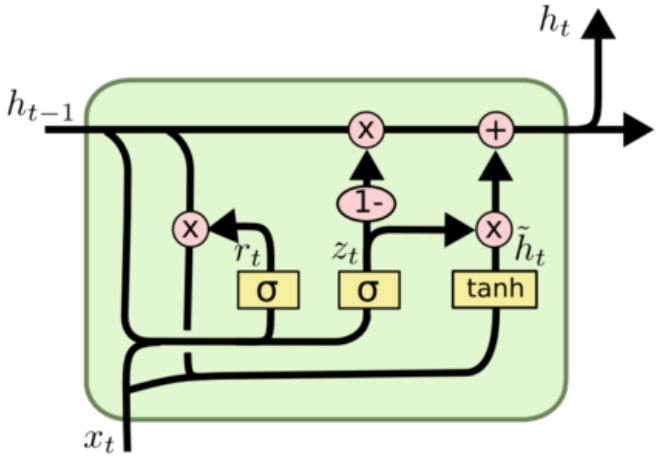


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

Нейросетевые методы прогнозирования

Управляемые рекуррентные нейроны (Gated Recurrent Units)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

https://neerc.ifmo.ru/wiki/index.php?title=Долгая_краткосрочная_память

Лекция 17. Обучение с подкреплением

Методы машинного обучения в анализе изображений и временных рядов

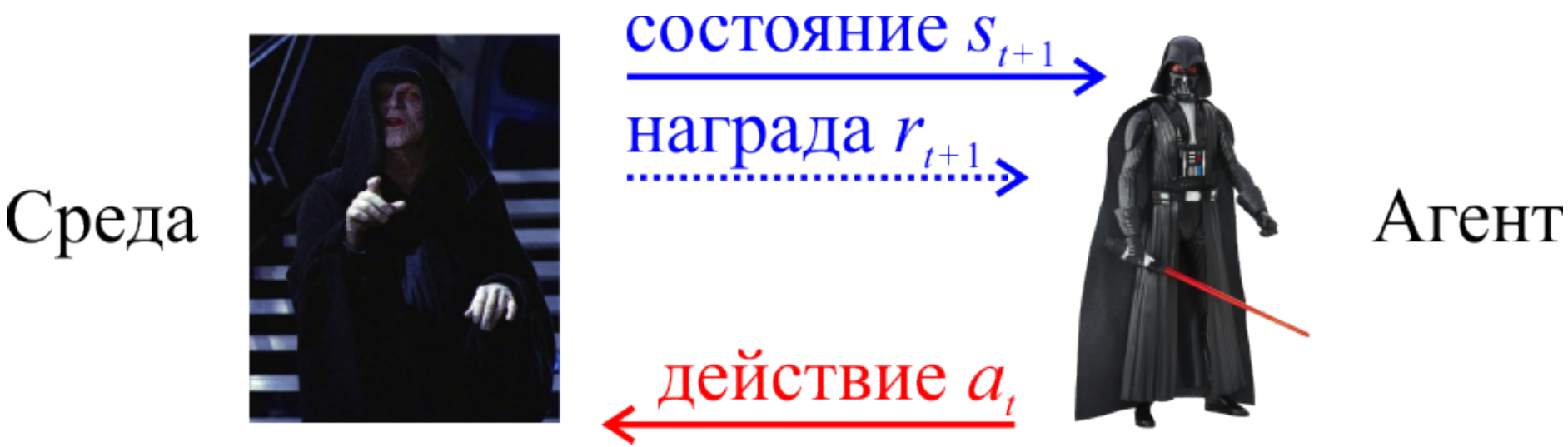
Дмитриев Константин Вячеславович

Московский государственный университет имени М.В. Ломоносова

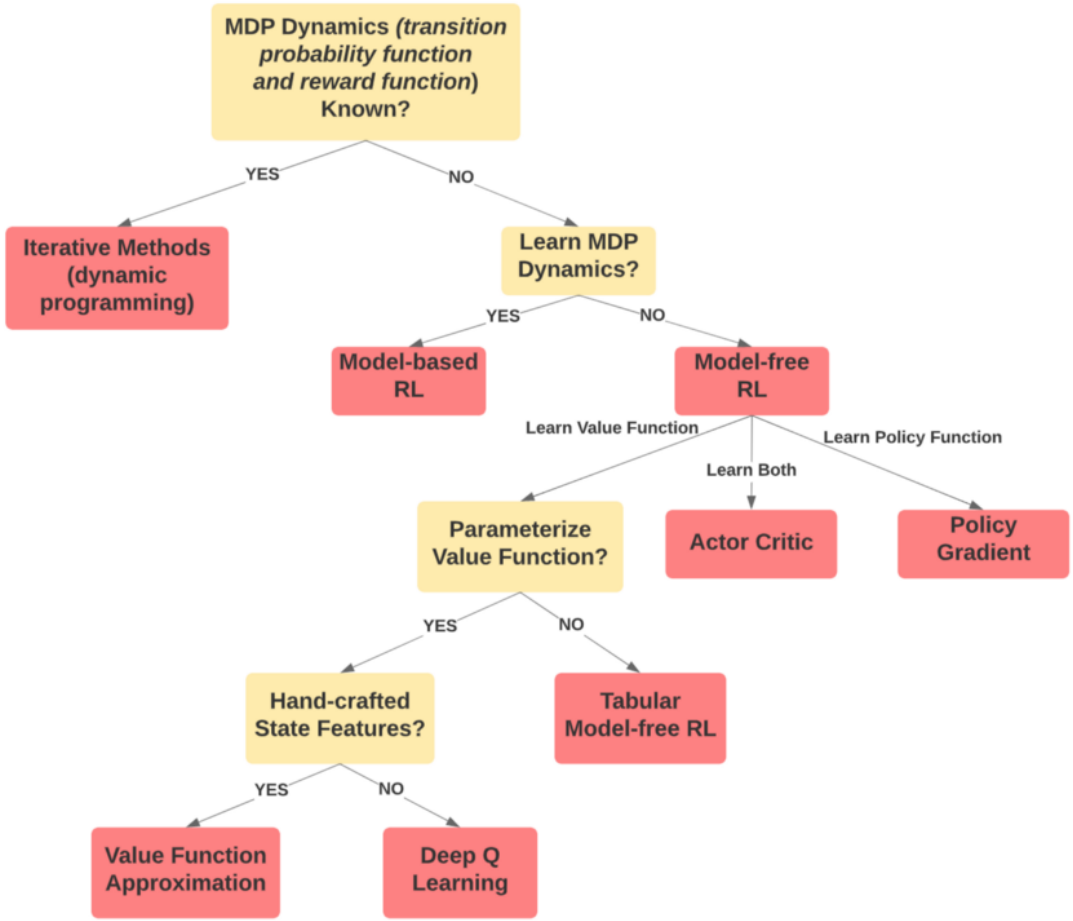
Обучение с подкреплением

Определение 17.1

Обучение с подкреплением (reinforcement learning) *— способ машинного обучения, в ходе которого в ходе которого испытуемая система (агент) обучается, взаимодействуя с некоторой средой.*

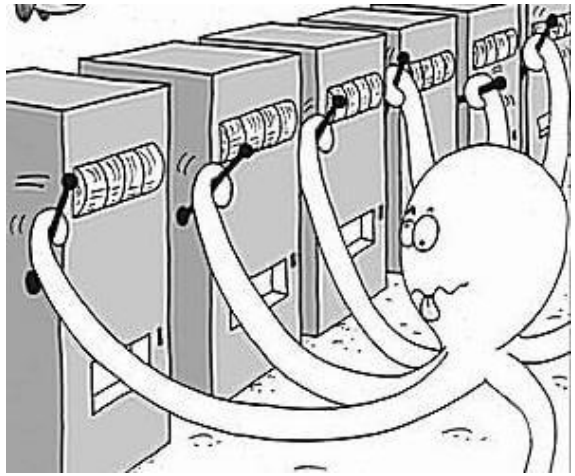


Виды обучения с подкреплением



towardsdatascience.com/an-overview-of-classic-reinforcement-learning-algorithms-part-1-f79c8b87e5af

Задача “многорукого бандита”



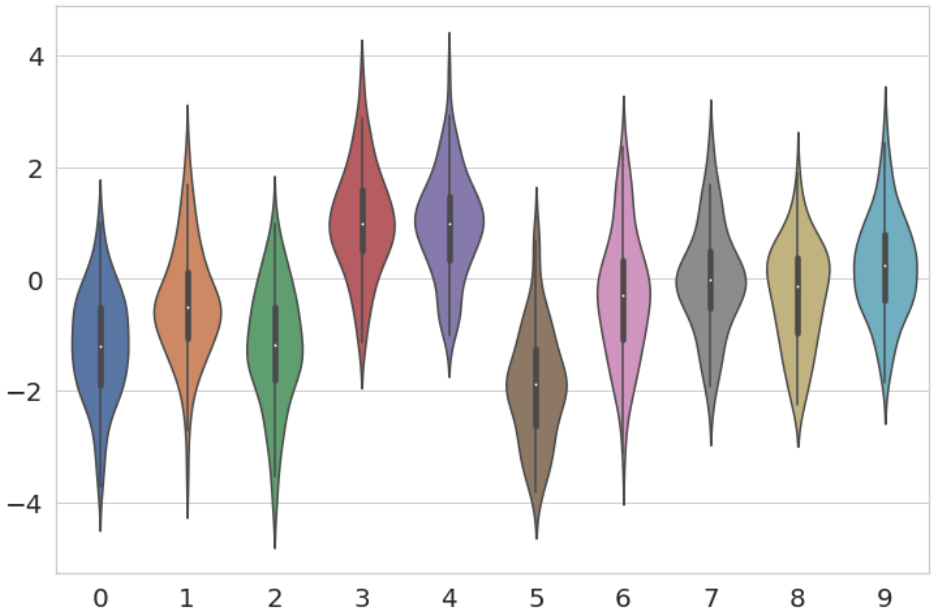
На каждом шаге t

- Агент совершает случайное действие $a_t \in \mathcal{A}$, используя для этого стретегию (policy) $\pi_t(a)$: $\quad a_t \sim \pi_t(a)$.
- Среда выдает агенту случайную премию $r_t \in \mathcal{R}$ из неизвестного распределения $p(r|a_t)$: $\quad r_t \sim p(r|a_t)$.
- Агент изменяет свою стратегию на π_{t+1}.

Задача “многорукого бандита”

Пример – десятирукий бандит

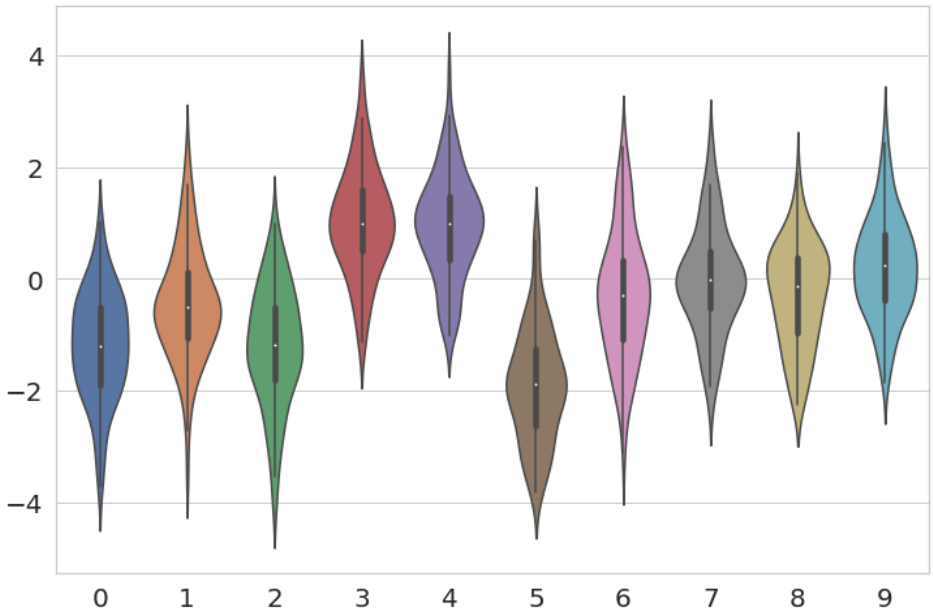
Каждая “рука” гауссовским распределением с дисперсией 1.



Задача “многорукого бандита”

Пример – десятирукий бандит

Каждая “рука” гауссовским распределением с дисперсией 1.

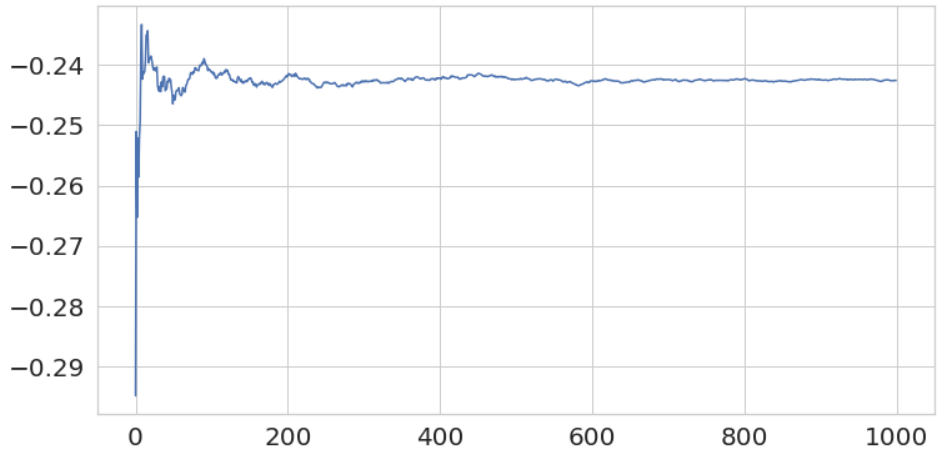


Наилучший результат при большом числе испытаний равен 1.142.

Задача “многорукого бандита”

Простейший игрок

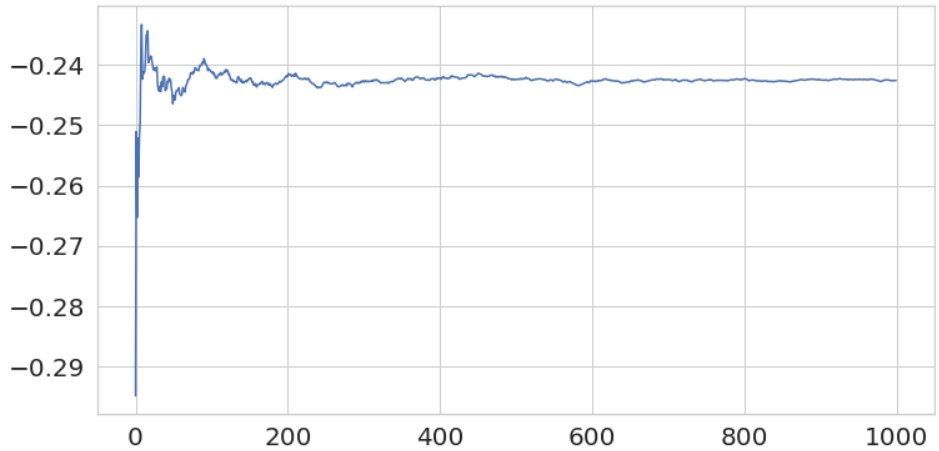
Простейший игрок делает выбор произвольно, т.е. не обучается.



Задача “многорукого бандита”

Простейший игрок

Простейший игрок делает выбор произвольно, т.е. не обучается.

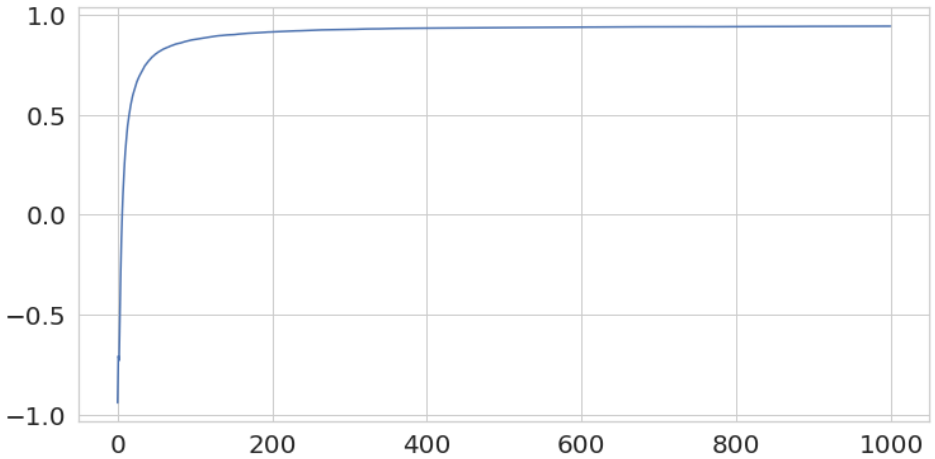


На большом числе тестов результат оказывается близок к среднему вознаграждению -0.279, что далеко от идеала, равного 1.142.

Задача “многорукого бандита”

Жадный игрок

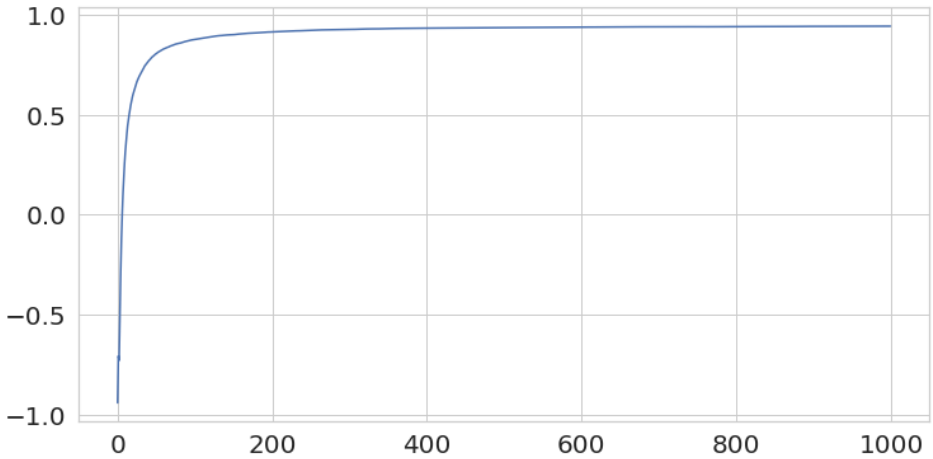
- Создаются массивы значений Q_a и счетчиков каждого действия N_a.
- На каждом шаге игры t выбираем действие $a = \arg\max Q_a$, и инкрементируем соответствующий счетчик $N_a := N_a + 1$.
- При получении награды r_t полагается $Q_a := Q_a + \frac{1}{N_a}(r_t - Q_a)$.



Задача “многорукого бандита”

Жадный игрок

- Создаются массивы значений Q_a и счетчиков каждого действия N_a.
- На каждом шаге игры t выбираем действие $a = \arg\max Q_a$, и инкрементируем соответствующий счетчик $N_a := N_a + 1$.
- При получении награды r_t полагается $Q_a := Q_a + \frac{1}{N_a}(r_t - Q_a)$.

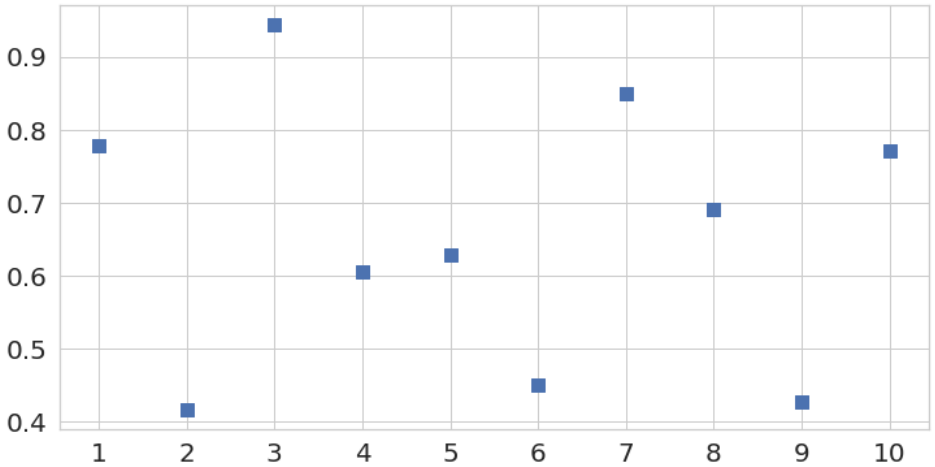


На большом числе тестов результат оказывается близок к среднему вознаграждению 0.944, что гораздо лучше простейшего игрока.

Задача “многорукого бандита”

Жадный игрок

- Создаются массивы значений Q_a и счетчиков каждого действия N_a.
- На каждом шаге игры t выбираем действие $a = \arg\max Q_a$, и инкрементируем соответствующий счетчик $N_a := N_a + 1$.
- При получении награды r_t полагается $Q_a := Q_a + \frac{1}{N_a}(r_t - Q_a)$.



Устойчивость алгоритма низкая: при 10 запусках получились сильно различные результаты.

Баланс exploration-exploitation

- Exploration – попытки исследовать задачу, выбирая, возможно, не самое лучшее на данный момент действие.
- Exploitation – попытка применить уже имеющиеся знания, чтобы получить как можно лучший результат.

Баланс exploration-exploitation

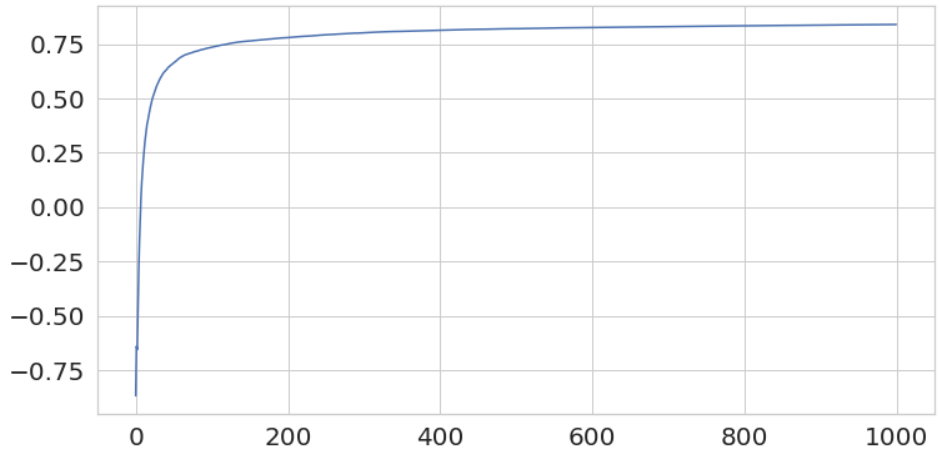
- Exploration – попытки исследовать задачу, выбирая, возможно, не самое лучшее на данный момент действие.
- Exploitation – попытка применить уже имеющиеся знания, чтобы получить как можно лучший результат.

Логично в начале отдавать предпочтение исследованию, а затем постепенно переходить к применению информации.

Задача “многорукого бандита”

Параметрически жадный игрок

- Создаются массивы значений Q_a и счетчиков каждого действия N_a.
- На каждом шаге игры t с вероятностью ϵ выбираем произвольное действие и с вероятностью $(1 - \epsilon)$ выбираем действие $a = \arg\max Q_a$. Инкрементируем счетчик $N_a := N_a + 1$.
- При получении награды r_t полагается $Q_a := Q_a + \frac{1}{N_a}(r_t - Q_a)$.

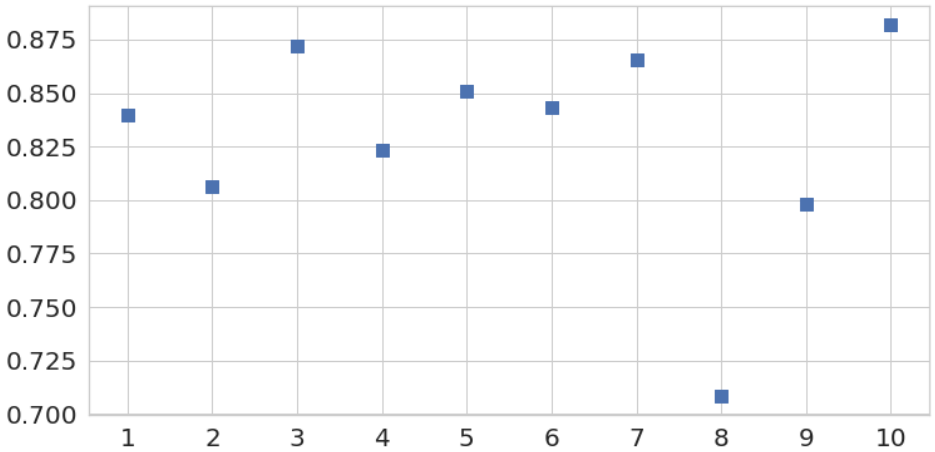


На большом числе тестов результат оказывается близок к среднему вознаграждению 0.839, что несколько хуже, чем у жадного игрока.

Задача “многорукого бандита”

Параметрически жадный игрок

- Создаются массивы значений Q_a и счетчиков каждого действия N_a.
- На каждом шаге игры t с вероятностью ϵ выбираем произвольное действие и с вероятностью $(1 - \epsilon)$ выбираем действие $a = \arg\max Q_a$. Инкрементируем счетчик $N_a := N_a + 1$.
- При получении награды r_t полагается $Q_a := Q_a + \frac{1}{N_a}(r_t - Q_a)$.

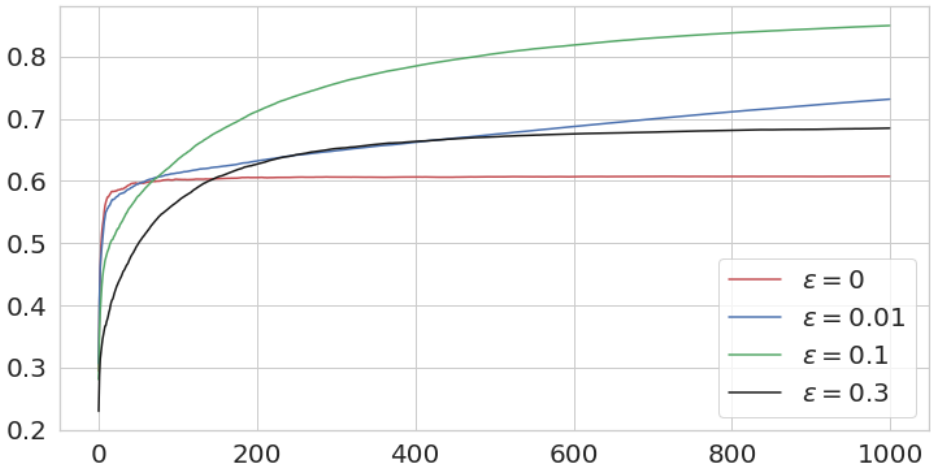


Зато устойчивость алгоритма повысилась.

Задача “многорукого бандита”

Параметрически жадный игрок

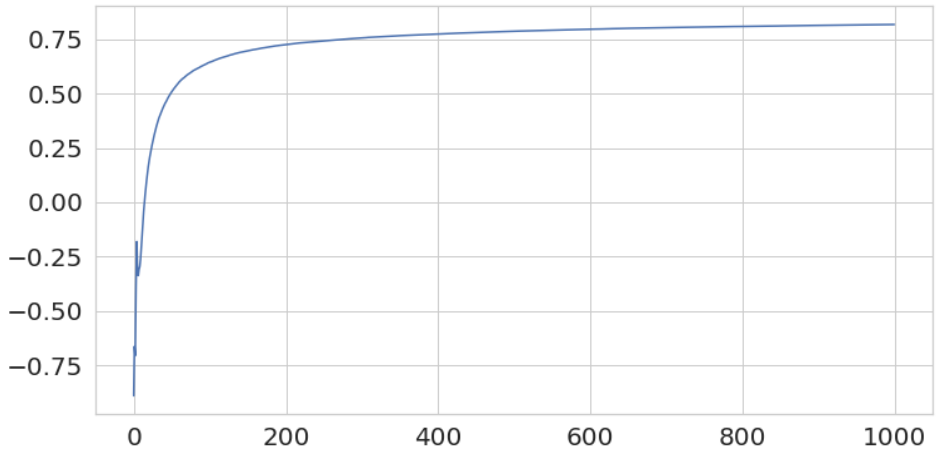
Зависимость результатов от параметра “нежадности”.



Задача “многорукого бандита”

Оптимистичный игрок

- Создаются массивы значений Q_a с относительно большими значениями и счетчиков каждого действия N_a.
- На каждом шаге игры t с вероятностью ϵ выбираем произвольное действие и с вероятностью $(1 - \epsilon)$ выбираем действие $a = \arg\max Q_a$. Инкрементируем счетчик $N_a := N_a + 1$.
- При получении награды r_t полагается $Q_a := Q_a + \frac{1}{N_a}(r_t - Q_a)$.

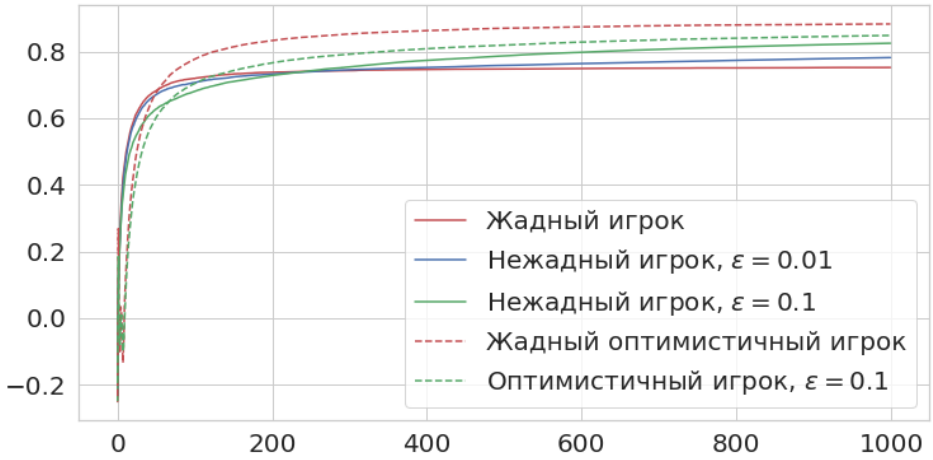


На большом числе тестов результат оказывется близок к среднему вознаграждению 0.819.

Задача “многорукого бандита”

Оптимистичный игрок

Зависимость результатов от параметров “нежадности” и “оптимистичности”.

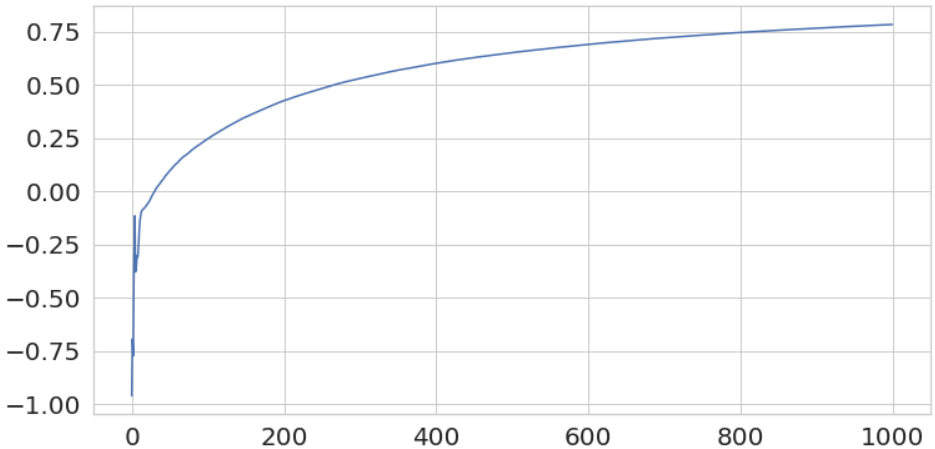


Задача “многорукого бандита”

Upper Confidence Bound игрок

Идея: стоит учитывать уровень неопределенности, существующий из-за того, что какое-то действие выбиралось редко.

- UCB-игрок выбирает действие $a = \arg\max \left\{ Q_a + C\sqrt{\frac{\ln \sum N_a}{N_a}} \right\}$.
- C – некоторая константа; $\sum N_a$ – общее число шагов.
- Если для какого-то действия $N_a = 0$, то выбирается именно оно.

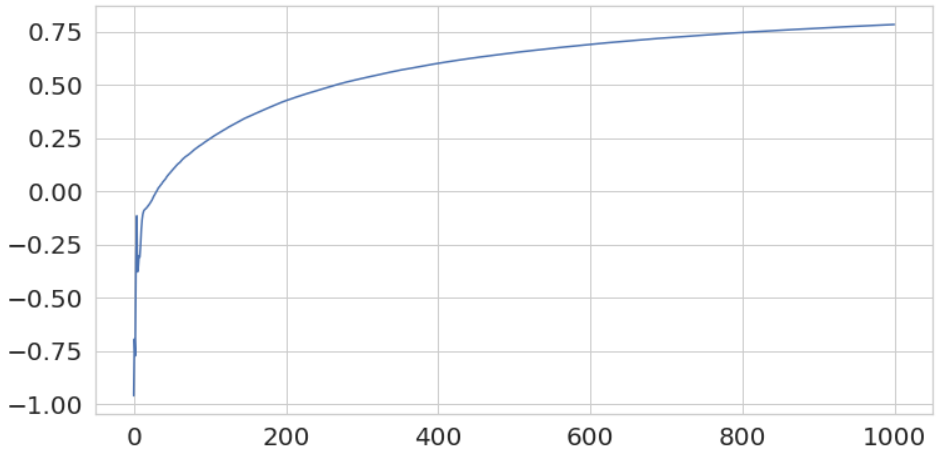


На большом числе тестов результат оказывется близок к среднему вознаграждению 0.783.

Задача “многорукого бандита”

Градиентный игрок

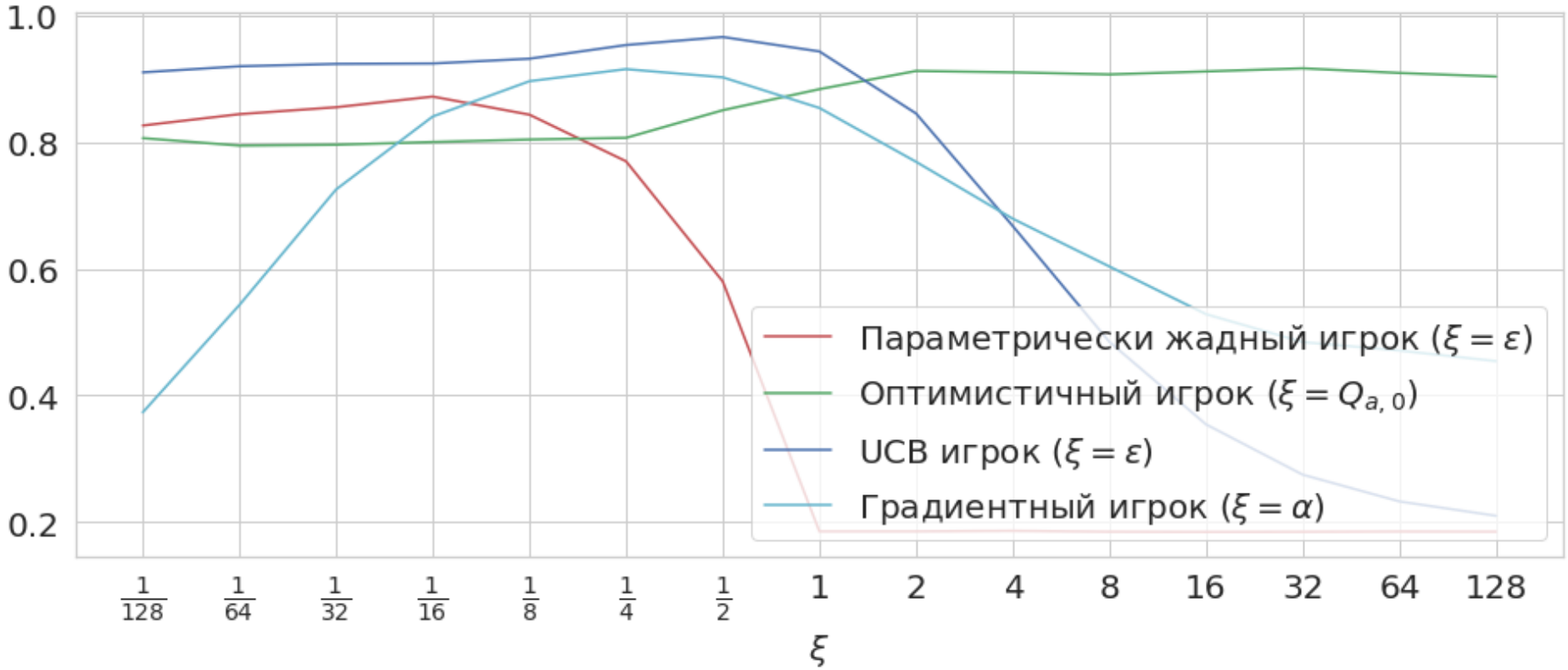
- Для каждого действия вводится функция предпочтения H_a.
- Стратегия выбора определяется как $\pi_a = e^{H_a} / \sum_{b=1}^{k} e^{H_b}$.
- Обновление после действия a и получения награды r_t:
  $H_a := H_a + \alpha(R - \overline{R})(1 - \pi(a)); \quad H_b := H_b - \alpha(R - \overline{R})\pi(b); b \neq a,$
  где $\overline{R}$ – средняя награда за всё время.



На большом числе тестов результат оказывается близок к среднему вознаграждению 0.824.

Задача “многорукого бандита”

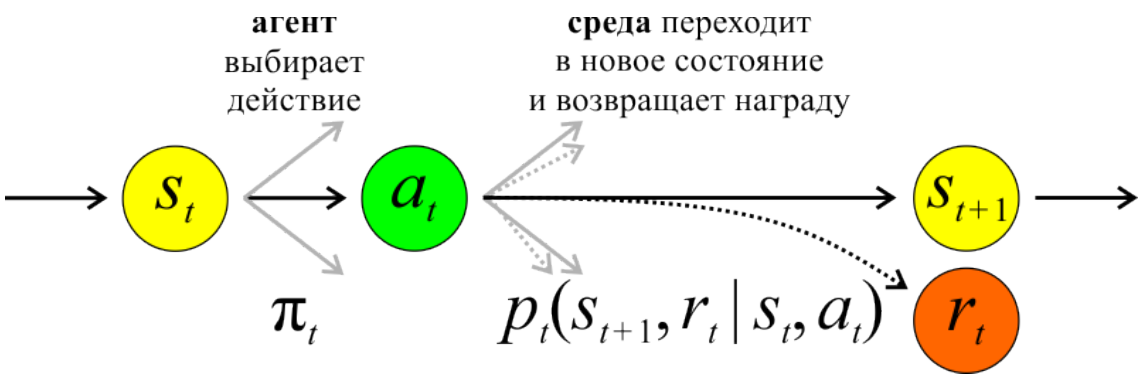
Сравнение алгоритмов при разных значениях параметров



Конечный марковский процесс принятия решений (Finite Markov Decision Process)

На каждом шаге t

- Агент получает состояние среды $s_t \in \mathcal{S}$
- Агент совершает случайное действие $a_t \in \mathcal{A}$, используя для этого стретегию (policy) $\pi_t(a|s)$: $\quad a_t \sim \pi_t(a|s)$.
- Среда переходит в новое состояние $s_{t+1} \in \mathcal{S}$ и выдает агенту случайную премию $r_t \in \mathcal{R} \subset \Re$ из распределения $p(s_{t+1}, r_t|s_t, a_t)$.
- Агент изменяет свою стратегию на π_{t+1}.



Конечный марковский процесс принятия решений (Finite Markov Decision Process)

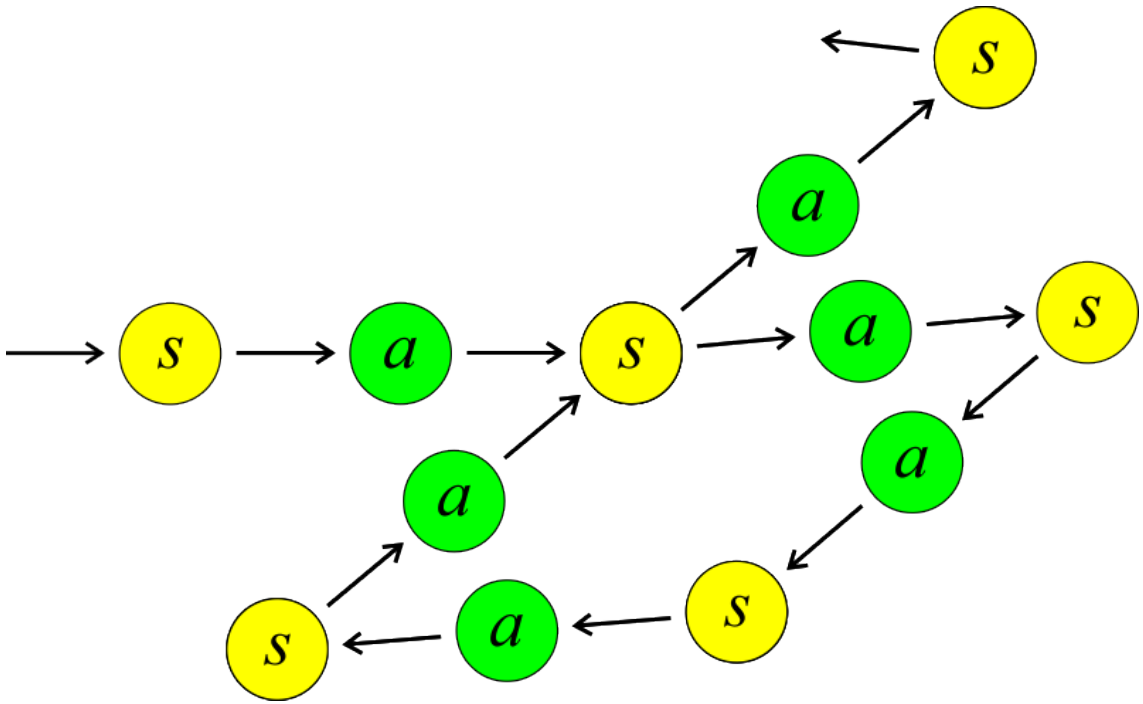
На каждом шаге t

- Агент получает состояние среды $s_t \in \mathcal{S}$
- Агент совершает случайное действие $a_t \in \mathcal{A}$, используя для этого стретегию (policy) $\pi_t(a|s)$: $\quad a_t \sim \pi_t(a|s)$.
- Среда переходит в новое состояние $s_{t+1} \in \mathcal{S}$ и выдает агенту случайную премию $r_t \in \mathcal{R} \subset \Re$ из распределения $p(s_{t+1}, r_t|s_t, a_t)$.
- Агент изменяет свою стратегию на π_{t+1}.

Особенности FMDP:

- Множества $\mathcal{S}, \mathcal{A}, \mathcal{R}$ – конечные.
- Состояние s_{t+1} и премия r_t определяются только предыдущим состоянием s_t и действием a_t – свойство марковости.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)



В зависимости от стратегии агента π требуется определять цену состояния $v_\pi(s)$ и цену действия $q_\pi(s,a)$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Вводится коэффициент дисконтирования $0 < \gamma \leq 1$.

$$v_\pi(s) = \mathcal{M}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s \right\}; \; q_\pi(a|s) = \mathcal{M}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| \begin{matrix} s_t = s, \\ a_t = a \end{matrix} \right\}$$

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Вводится коэффициент дисконтирования $0 < \gamma \leq 1$.

$$v_\pi(s) = \mathcal{M}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s \right\}; \; q_\pi(a|s) = \mathcal{M}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| \begin{matrix} s_t = s, \\ a_t = a \end{matrix} \right\}$$

$$v_\pi(s) = \mathcal{M}_\pi \left\{ r_{t+1} + \gamma \underbrace{\sum_{k=0}^{\infty} \gamma^k r_{(t+1)+k+1}}_{v_\pi(s')} \middle| s_t = s \right\}$$

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Вводится коэффициент дисконтирования $0 < \gamma \leq 1$.

$$v_\pi(s) = \mathcal{M}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s \right\}; \; q_\pi(a|s) = \mathcal{M}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| \begin{matrix} s_t = s, \\ a_t = a \end{matrix} \right\}$$

$$v_\pi(s) = \mathcal{M}_\pi \left\{ r_{t+1} + \gamma \underbrace{\sum_{k=0}^{\infty} \gamma^k r_{(t+1)+k+1}}_{v_\pi(s')} \middle| s_t = s \right\}$$

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')] \text{ – уравнение Беллмана}$$

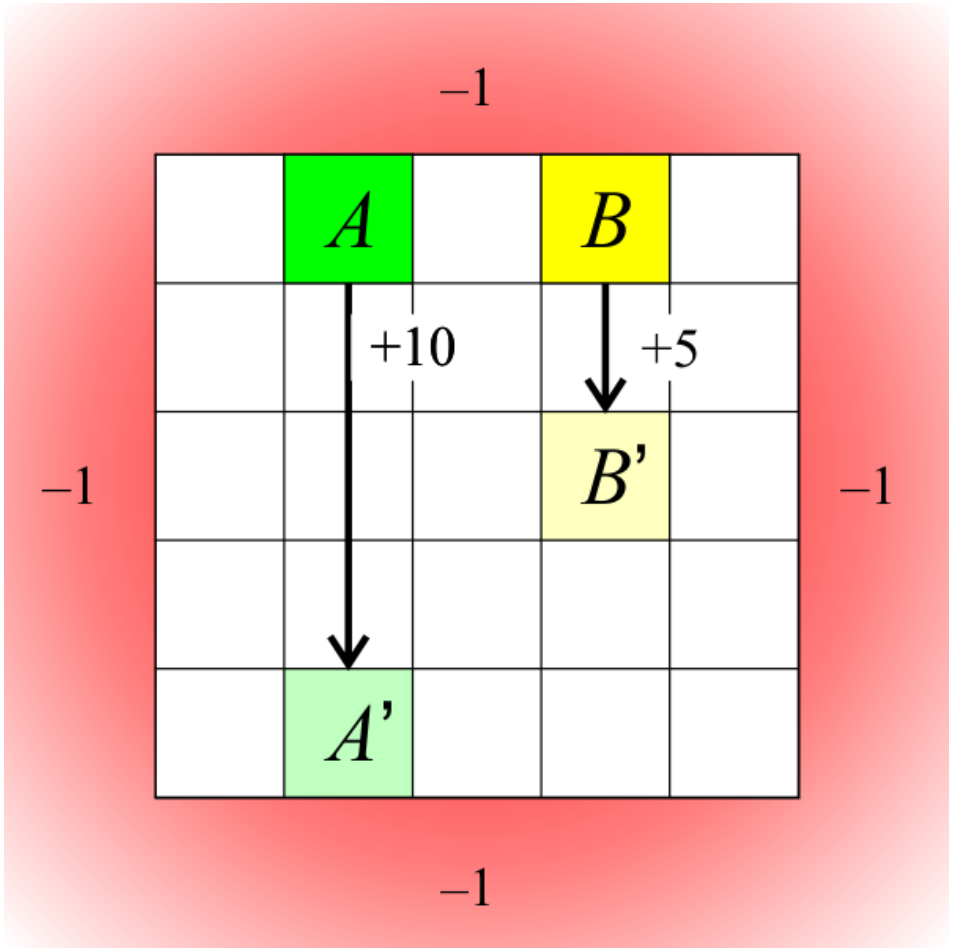
Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Алгоритм работы

- Пусть заданы начальные стратегия π и цены состояний $v_\pi(s)$.
- Обновляются цены $v_\pi(s) := \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$.
- Обновляется стратегия π': она выбирает на каждом шаге одно из действий с максимальной суммарной наградой.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

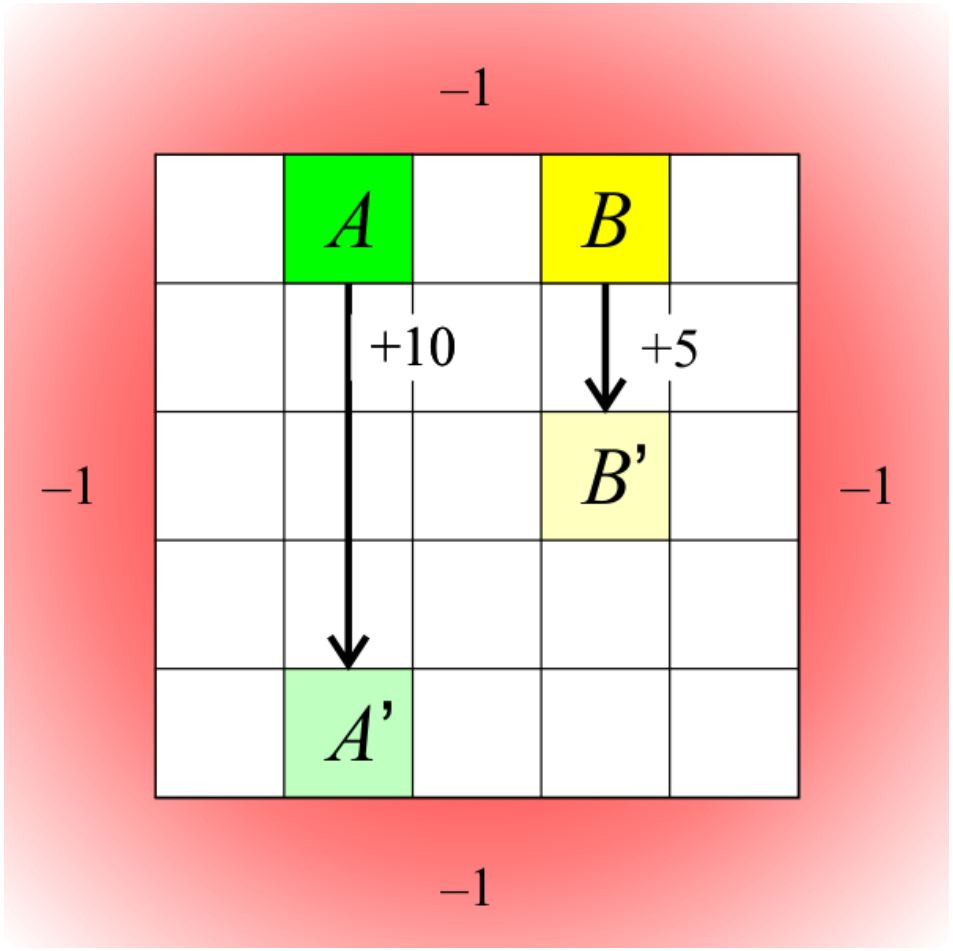
Пример: путешествие по клеткам



- Агент перемещается по клеткам на поле 5×5.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

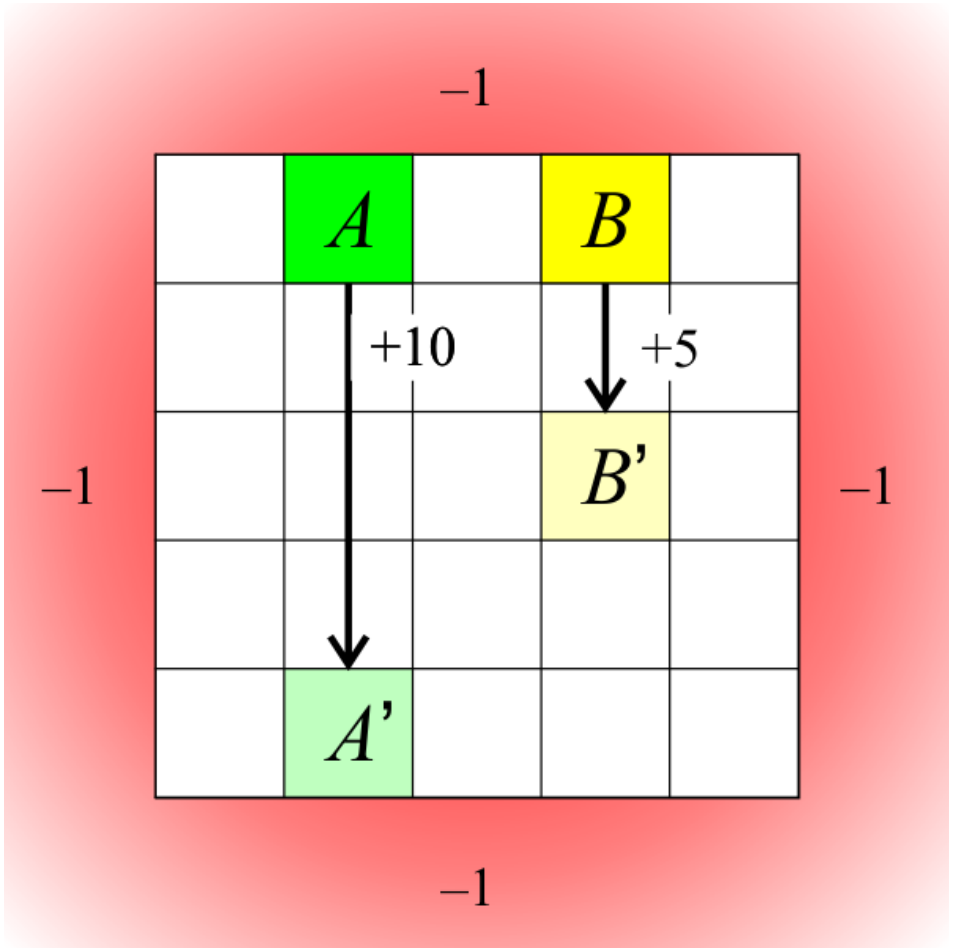
Пример: путешествие по клеткам



- Агент перемещается по клеткам на поле 5×5.
- При попытке покинуть поле назначается награда -1.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

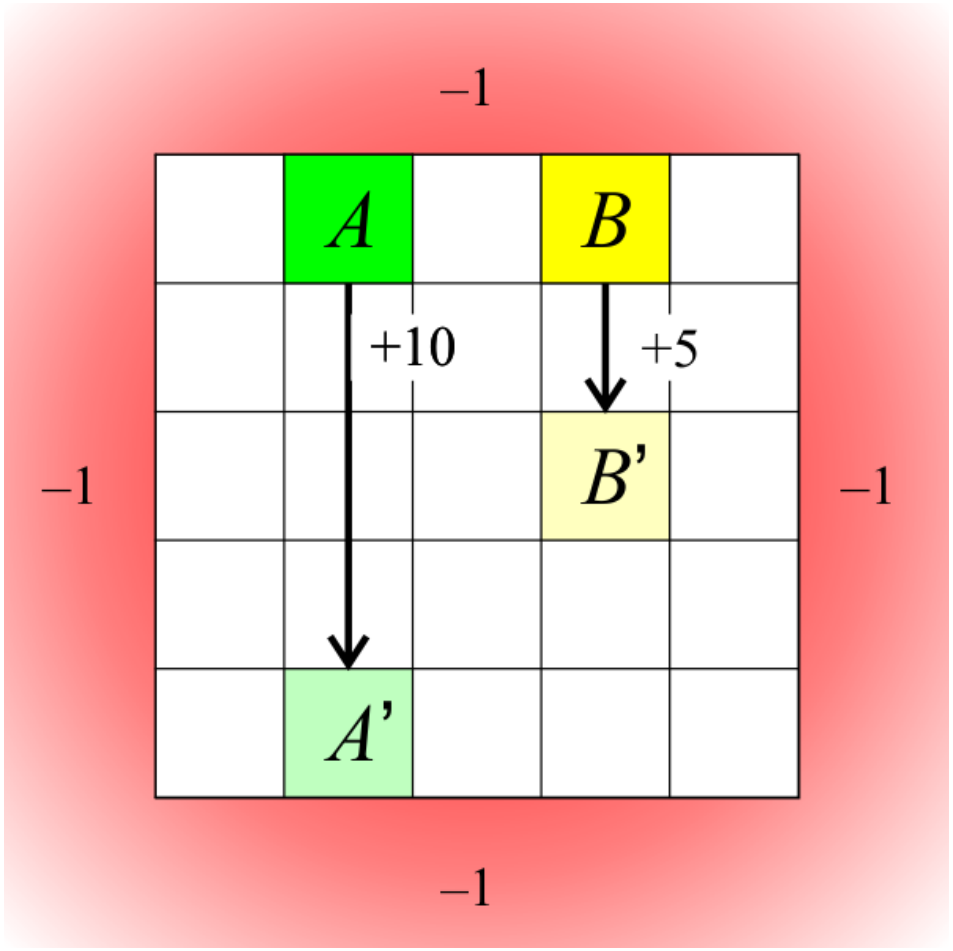
Пример: путешествие по клеткам



- Агент перемещается по клеткам на поле 5×5.
- При попытке покинуть поле назначается награда -1.
- В точке $A(1;0)$ награда $+10$ и перемещение в $A'(1;4)$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

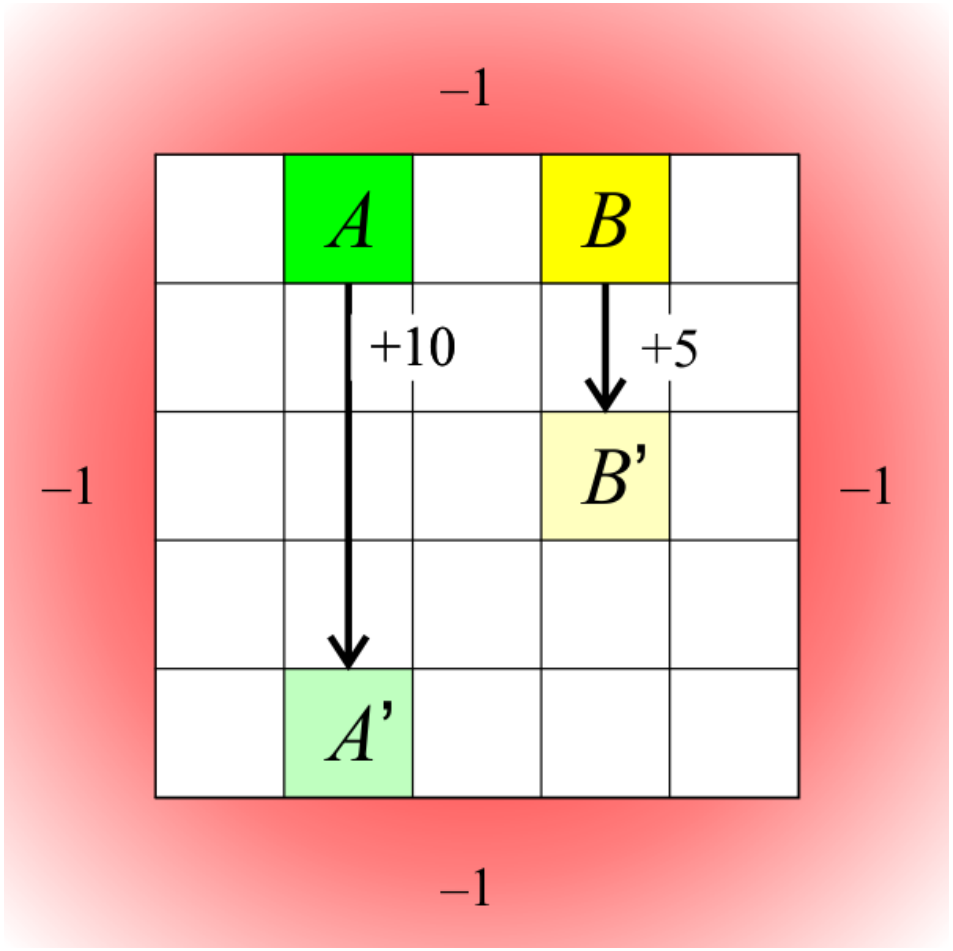
Пример: путешествие по клеткам



- Агент перемещается по клеткам на поле 5×5.
- При попытке покинуть поле назначается награда -1.
- В точке $A(1;0)$ награда $+10$ и перемещение в $A'(1;4)$.
- В точке $B(3;0)$ награда $+5$ и перемещение в $B'(3;2)$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам



- Агент перемещается по клеткам на поле 5×5.
- При попытке покинуть поле назначается награда -1.
- В точке $A(1;0)$ награда $+10$ и перемещение в $A'(1;4)$.
- В точке $B(3;0)$ награда $+5$ и перемещение в $B'(3;2)$.
- В других случаях награда 0.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

- Начальная стратегия – выбор произвольного действия $\pi(a|s) = 0.25$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

- Начальная стратегия – выбор произвольного действия $\pi(a|s) = 0.25$.
- Коэффициент дисконтирования $\gamma = 0.9$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

- Начальная стратегия – выбор произвольного действия $\pi(a|s) = 0.25$.
- Коэффициент дисконтирования $\gamma = 0.9$.
- Результат действия в данном случае однозначный, т.е. $p(s',r|s,a) = 1$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

- Начальная стратегия – выбор произвольного действия $\pi(a|s) = 0.25$.
- Коэффициент дисконтирования $\gamma = 0.9$.
- Результат действия в данном случае однозначный, т.е. $p(s',r|s,a) = 1$. Поэтому уравнение Беллмана упрощается:
  $v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] = \sum_a \pi(a|s)[r + \gamma v_\pi(s')]$.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

- Начальная стратегия – выбор произвольного действия $\pi(a|s) = 0.25$.
- Коэффициент дисконтирования $\gamma = 0.9$.
- Результат действия в данном случае однозначный, т.е. $p(s',r|s,a) = 1$. Поэтому уравнение Беллмана упрощается:
  $v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] = \sum_a \pi(a|s)[r + \gamma v_\pi(s')]$.
- Оценки $v_\pi(s)$ уточняются, пока изменение не станет меньше 10^{-7}.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

- Начальная стратегия – выбор произвольного действия $\pi(a|s) = 0.25$.
- Коэффициент дисконтирования $\gamma = 0.9$.
- Результат действия в данном случае однозначный, т.е. $p(s',r|s,a) = 1$. Поэтому уравнение Беллмана упрощается:
  $v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] = \sum_a \pi(a|s)[r + \gamma v_\pi(s')]$.
- Оценки $v_\pi(s)$ уточняются, пока изменение не станет меньше 10^{-7}.
- Задача сходится за 81 шаг:

3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	−0.4
−1.0	−0.4	−0.4	−0.6	−1.2
−1.9	−1.3	−1.2	−1.4	−2.0

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

Стратегия произвольного действия не оптимальна.
Необходимо обновить ее жадным образом.

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

Стратегия произвольного действия не оптимальна.
Необходимо обновить ее жадным образом.

Задача сходится за 5 итераций; всего 145 шагов.

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

→	↔↕	←	↔↕	←
→↑	↑	←↑	←	←
→↑	↑	←↑	←↑	←↑
→↑	↑	←↑	←↑	←↑
→↑	↑	←↑	←↑	←↑

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

Идея: для ускорения сходимости следует ограничить число шагов на каждой итерации. Например, до одного шага

Конечный марковский процесс принятия решений (Finite Markov Decision Process)

Пример: путешествие по клеткам

Идея: для ускорения сходимости следует ограничить число шагов на каждой итерации. Например, до одного шага

Задача сходится за 5 итераций; всего 5 шагов.

19.2	22.6	20.4	17.6	15.9
17.3	20.4	18.3	16.5	14.8
15.6	18.3	16.5	14.8	13.4
14.0	16.5	14.8	13.4	12.0
12.6	14.8	13.4	12.0	10.8

→	↔↕	←	↔↕	←
→	↑	←↑	←	←
→	↑	←↑	←↑	←↑
→	↑	←↑	←↑	←↑
→	↑	←↑	←↑	←↑

Обучение с подкреплением без моделирования среды

- О модели среды не известно ничего.

Обучение с подкреплением без моделирования среды

- О модели среды не известно ничего.
- Вероятности переходов неизвестны, и уравнение Беллмана применить не удается.

Обучение с подкреплением без моделирования среды

- О модели среды не известно ничего.
- Вероятности переходов неизвестны, и уравнение Беллмана применить не удается.
- Цена состояния $v_\pi(s)$ теперь не играет большой роли, поскольку нужно выбирать действие, а модель неизвестна. Гораздо важнее величина $q_\pi(s, a)$.

Обучение с подкреплением без моделирования среды

- О модели среды не известно ничего.
- Вероятности переходов неизвестны, и уравнение Беллмана применить не удается.
- Цена состояния $v_\pi(s)$ теперь не играет большой роли, поскольку нужно выбирать действие, а модель неизвестна. Гораздо важнее величина $q_\pi(s,a)$.
- Важно, чтобы все пары (s,a) были охвачены, иначе оптимальное действие может не буть выбрано никогда. Поэтому параметр жадности задается $\epsilon > 0$.

Метод Монте-Карло

- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s, a).

Метод Монте-Карло

- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s, a).
- Для каждого эпизода игры $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$

Метод Монте-Карло

- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s, a).
- Для каждого эпизода игры $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$:
  - Создается массив N, где для каждой пары состояние-действие (s, a) будет указан номер n ее первого появления в эпизоде.

Метод Монте-Карло

- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s, a).
- Для каждого эпизода игры $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$:
  - Создается массив N, где для каждой пары состояние-действие (s, a) будет указан номер n ее первого появления в эпизоде.
  - Вычисляется цена g_t пары (s_t, a_t) как $\sum_{k=t+1}^{T} r_k$.

Метод Монте-Карло

- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s, a).
- Для каждого эпизода игры $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$:
  - Создается массив N, где для каждой пары состояние-действие (s, a) будет указан номер n ее первого появления в эпизоде.
  - Вычисляется цена g_t пары (s_t,a_t) как $\sum_{k=t+1}^{T} r_k$.
  - Если $N[(s_t,a_t)] = t$, т.е. пара (s_t,a_t) встретилась в эпизоде впервые, то значение g_t добавляется в массив $R[(s_t,a_t)]$.

Метод Монте-Карло

- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s, a).
- Для каждого эпизода игры $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$:
  - Создается массив N, где для каждой пары состояние-действие (s, a) будет указан номер n ее первого появления в эпизоде.
  - Вычисляется цена g_t пары (s_t,a_t) как $\sum_{k=t+1}^{T} r_k$.
  - Если $N[(s_t,a_t)] = t$, т.е. пара (s_t,a_t) встретилась в эпизоде впервые, то значение g_t добавляется в массив $R[(s_t,a_t)]$.
  - Цена пары (s,a) заносится в массив Q: $Q[(s,a)] = \text{avg}\, R[(s,a)]$.

Метод Монте-Карло

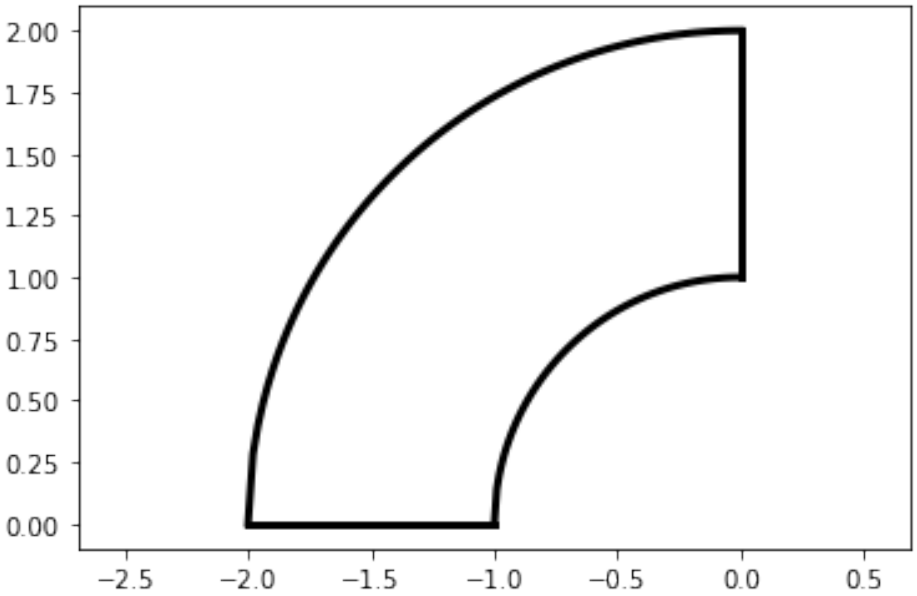
- Инициализация:
  - Задается начальная стратегия π.
  - Создается пустые массивы R и Q с индексами в виде пар (s,a).
- Для каждого эпизода игры $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$:
  - Создается массив N, где для каждой пары состояние-действие (s,a) будет указан номер n ее первого появления в эпизоде.
  - Вычисляется цена g_t пары (s_t,a_t) как $\sum_{k=t+1}^{T} r_k$.
  - Если $N[(s_t,a_t)] = t$, т.е. пара (s_t,a_t) встретилась в эпизоде впервые, то значение g_t добавляется в массив $R[(s_t,a_t)]$.
  - Цена пары (s,a) заносится в массив Q: $Q[(s,a)] = \text{avg}\, R[(s,a)]$.
  - Обновляется π: в состоянии s

    с вероятностью $(1-\epsilon)$ выбирается действие $a = \arg\max_a Q[(s,a)]$;

    с вероятностью ϵ – все остальные действия равномерно.

Метод Монте-Карло

Пример: гонки на плоскости

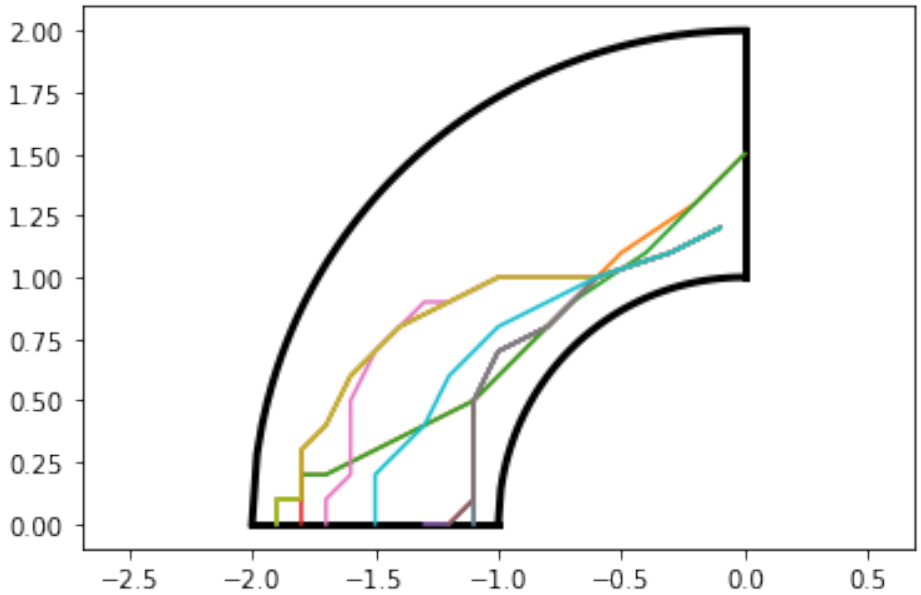


- Трасса машинки ограничена линиями старта и финиша и боковыми линиями.

- На старте $v_x = v_y = 0$.
- Далее $0 < v_x, v_y \leq 5$.
- На каждом шаге можно изменить компоненты скорости на ± 1.
- При выходе с трассы машинка помещается на старт, назначается награда -100, а эпизод продолжается.
- На каждом шаге, не приводящем к выходу с трассы, награда равна -1.

Метод Монте-Карло

Пример: гонки на плоскости



Практикум. Обучение с подкреплением

Jupyter notebook “Многорукие бандиты”:
https://colab.research.google.com/drive/1xfDBhY4ugLyTgdXSkadw569YzBCR628e

Jupyter notebook “Конечный марковский процесс принятия решений”:
https://colab.research.google.com/drive/1z3DWhRMGAAcT5al5Pxhw3jtf1H0ruM2j

Jupyter notebook “Метод Монте-Карло”:
https://colab.research.google.com/drive/1r0h9LdN2CeBv9kxVN_SBQbczKG3RqQjs

Метод SARSA (State–Action–Reward–State–Action)

Идея: использовать экспоненциальное скользящее среднее для вычисления $Q(s,a)$.

- Задается стратегия $\pi(a|s)$, например,
  с вероятностью $(1-\epsilon)$ $a_t = \arg\max_a Q[(s_t,a_t)]$;
  с вероятностью ϵ a_t – любое другое действие.
- Согласно $\pi(a|s)$ на каждом шаге t совершается действие a_t.
- В ответ среда выдает награду r_t и переходит из состояния s_t в s_{t+1}.
- Согласно $\pi(a|s)$ выбирается (но не совершается) следующее действие a'. В итоге получается набор данных $\{s_t, a_t, r_t, s_{t+1}, a'\}$.
- Обновляется $Q(s_t, a_t) := (1-\alpha)Q(s_t, a_t) + \alpha(r_t + \gamma Q(s_{t+1}, a'))$

Метод Q-обучения (Q-learning)

Идея: использовать при выборе пробного действия a' оптимальную стратегию $\arg\max_a Q[(s_t,a_t)]$ вместо обучаемой стратегии $\pi(a|s)$.

- Задается стратегия $\pi(a|s)$, например,
  с вероятностью $(1-\epsilon)$ $a_t = \arg\max_a Q[(s_t,a_t)]$;
  с вероятностью ϵ a_t – любое другое действие.
- Согласно $\pi(a|s)$ на каждом шаге t совершается действие a_t.
- В ответ среда выдает награду r_t и переходит из состояния s_t в s_{t+1}.
- Выбирается (но не совершается) действие $a' = \arg\max_{a'} Q[(s_{t+1},a')]$.
  В итоге получается набор данных $\{s_t, a_t, r_t, s_{t+1}, a'\}$.
- Обновляется $Q(s_t,a_t) := (1-\alpha)Q(s_t,a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1},a'))$

Методы on-policy и off-policy

Методы on-policy используют для формирования последовательности $\{s_t, a_t, r_t, \dots\}$ ту же стратегию, которую они обучают.
Методы off-policy используют для формирования последовательности $\{s_t, a_t, r_t, \dots\}$ стратегию, отличную от обучаемой.

Аппроксимация функции значимости (VFA)

Идея: перейти от запоминания $Q(s, a)$ к аппроксимации $\hat{Q}(s, a; \vec{\theta})$.

Аппроксимация функции значимости (VFA)

Идея: перейти от запоминания $Q(s,a)$ к аппроксимации $\hat{Q}(s,a;\vec{\theta})$.

- Метод Монте-Карло:

$$\begin{cases} \mathcal{L}(\vec{\theta}) = \mathcal{M}\left\{\left(g_t - \hat{Q}(s_t,a_t;\vec{\theta})\right)^2\right\} \\ \delta\vec{\theta} = lr \cdot \left(g_t - \hat{Q}(s_t,a_t;\vec{\theta})\right)\nabla_{\vec{\theta}}\hat{Q}(s_t,a_t;\vec{\theta}) \end{cases}$$

Аппроксимация функции значимости (VFA)

Идея: перейти от запоминания $Q(s,a)$ к аппроксимации $\hat{Q}(s,a;\vec{\theta})$.

- Метод Монте-Карло:

$$\begin{cases} \mathcal{L}(\vec{\theta}) = \mathcal{M}\left\{\left(g_t - \hat{Q}(s_t,a_t;\vec{\theta})\right)^2\right\} \\ \delta\vec{\theta} = lr \cdot \left(g_t - \hat{Q}(s_t,a_t;\vec{\theta})\right)\nabla_{\vec{\theta}}\hat{Q}(s_t,a_t;\vec{\theta}) \end{cases}$$

- Метод SARSA:

$$\begin{cases} \mathcal{L}(\vec{\theta}) = \mathcal{M}\left\{\left(r_t + \gamma\hat{Q}(s_{t+1},a_{t+1};\vec{\theta}) - \hat{Q}(s_t,a_t;\vec{\theta})\right)^2\right\} \\ \delta\vec{\theta} = lr \cdot \left(r_t + \gamma\hat{Q}(s_{t+1},a_{t+1};\vec{\theta}) - \hat{Q}(s_t,a_t;\vec{\theta})\right)\nabla_{\vec{\theta}}\hat{Q}(s_t,a_t;\vec{\theta}) \end{cases}$$

Метод полуградиента: целевая переменная зависит от модели

Аппроксимация функции значимости (VFA)

Идея: перейти от запоминания $Q(s,a)$ к аппроксимации $\hat{Q}(s,a;\vec{\theta})$.

- Метод Монте-Карло:

$$\begin{cases} \mathcal{L}(\vec{\theta}) = \mathcal{M}\left\{\left(g_t - \hat{Q}(s_t,a_t;\vec{\theta})\right)^2\right\} \\ \delta\vec{\theta} = lr \cdot \left(g_t - \hat{Q}(s_t,a_t;\vec{\theta})\right)\nabla_{\vec{\theta}}\hat{Q}(s_t,a_t;\vec{\theta}) \end{cases}$$

- Метод SARSA:

$$\begin{cases} \mathcal{L}(\vec{\theta}) = \mathcal{M}\left\{\left(r_t + \gamma\hat{Q}(s_{t+1},a_{t+1};\vec{\theta}) - \hat{Q}(s_t,a_t;\vec{\theta})\right)^2\right\} \\ \delta\vec{\theta} = lr \cdot \left(r_t + \gamma\hat{Q}(s_{t+1},a_{t+1};\vec{\theta}) - \hat{Q}(s_t,a_t;\vec{\theta})\right)\nabla_{\vec{\theta}}\hat{Q}(s_t,a_t;\vec{\theta}) \end{cases}$$

- Метод Q-обучения:

$$\begin{cases} \mathcal{L}(\vec{\theta}) = \mathcal{M}\left\{\left(r_t + \gamma\max\limits_{a'}\hat{Q}(s_{t+1},a';\vec{\theta}) - \hat{Q}(s_t,a_t;\vec{\theta})\right)^2\right\} \\ \delta\vec{\theta} = lr \cdot \left(r_t + \gamma\max\limits_{a'}\hat{Q}(s_{t+1},a';\vec{\theta}) - \hat{Q}(s_t,a_t;\vec{\theta})\right)\nabla_{\vec{\theta}}\hat{Q}(s_t,a_t;\vec{\theta}) \end{cases}$$

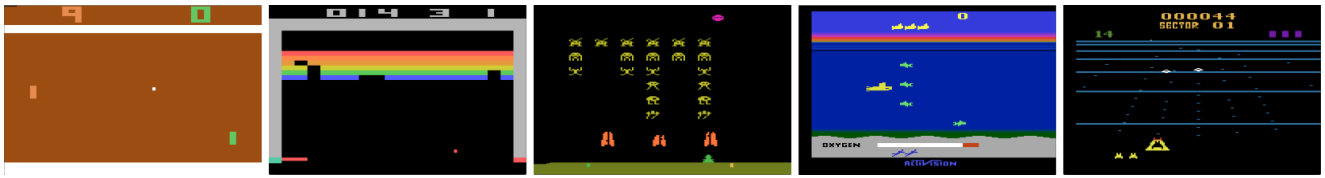
Метод Deep Q Networks (DQN)

Идея: использовать глубокие нейронные сети для аппроксимации $Q(s,a)$.

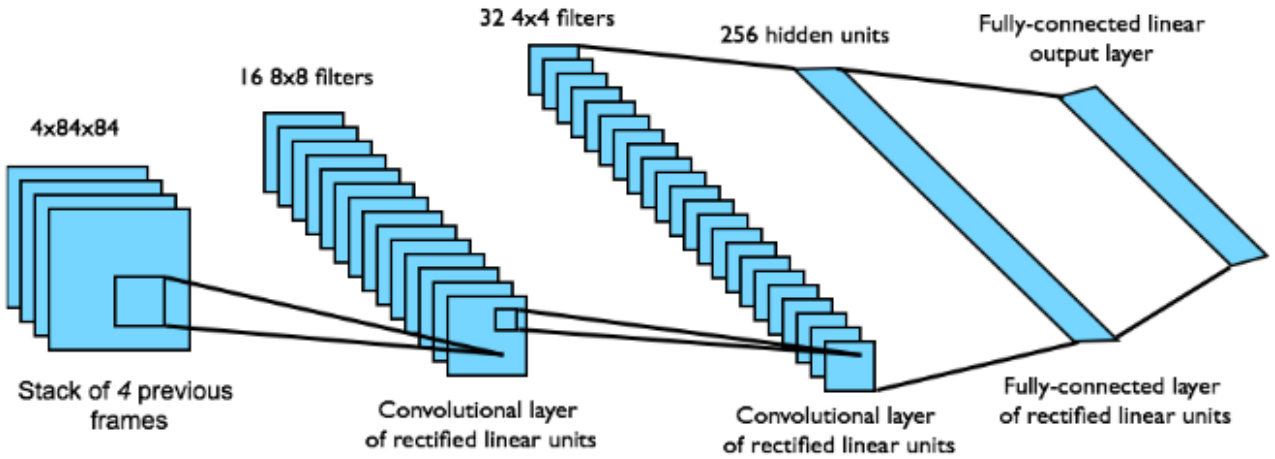
- Целевая сеть.
  Метод полуградиента приводит к нестабильным результатам.
  Поэтому используется целевая сеть с весами $\vec{\theta}^{\text{trg}}$, которые обновляются относительно нечасто:
  $$\delta\vec{\theta} = lr \cdot \left(r_t + \gamma \max_{a'} \hat{Q}(s_{t+1}, a'; \vec{\theta}^{\text{trg}}) - \hat{Q}(s_t, a_t; \vec{\theta}) \right) \nabla_{\vec{\theta}} \hat{Q}(s_t, a_t; \vec{\theta})$$
- Reply-память.
  В памяти сохраняются наборы данных $\{s_t, a_t, r_t, s_{t+1}, a'\}$.
  При получении нового набора вместо обучения на нем применяется SGD на мини-пакетах из имеющихся наборов данных.

Метод Deep Q Networks (DQN)

Игры Atari



Pong, Breakout, Space Invaders, Seaquest, Beam Rider



https://arxiv.org/pdf/1312.5602.pdf

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\,\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\,\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\} = \nabla_{\vec{\theta}}\sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s)$$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\,\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\} = \nabla_{\vec{\theta}}\sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s) = \sum_{a\in\mathcal{A}} f(s,a)\nabla_{\vec{\theta}}\,\pi_{\vec{\theta}}(a|s)$$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\} = \nabla_{\vec{\theta}}\sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s) = \sum_{a\in\mathcal{A}} f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s) =$$

$$= \sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s)\frac{\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)}{\pi_{\vec{\theta}}(a|s)}$$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\} = \nabla_{\vec{\theta}}\sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s) = \sum_{a\in\mathcal{A}} f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s) =$$

$$= \sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s)\frac{\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)}{\pi_{\vec{\theta}}(a|s)} = \mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)\}$$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\} = \nabla_{\vec{\theta}}\sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s) = \sum_{a\in\mathcal{A}} f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s) =$$

$$= \sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s)\frac{\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)}{\pi_{\vec{\theta}}(a|s)} = \mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)\}$$

- Вместо усреднения можно использовать аналог метода моментов:

$$\vec{g}_{t+1} := \vec{g}_t(1-\alpha) + \alpha f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)$$

Градиентная оптимизация стратегии (Policy gradient)

Идея: использовать параметризацию для самой стратегии: $\pi_{\vec{\theta}}(a|s)$

- Пусть функция ценности задана в виде $f(s_t, a_t)$.
- Тогда стоит задача максимизации $\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$
- Решение градиентным методом: $\vec{\theta} := \vec{\theta} + lr \cdot \nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\}$

$$\nabla_{\vec{\theta}}\mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\} = \nabla_{\vec{\theta}}\sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s) = \sum_{a\in\mathcal{A}} f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s) =$$

$$= \sum_{a\in\mathcal{A}} f(s,a)\pi_{\vec{\theta}}(a|s)\frac{\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)}{\pi_{\vec{\theta}}(a|s)} = \mathcal{M}_{a\sim\pi_{\vec{\theta}}(a|s)}\{f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)\}$$

- Вместо усреднения можно использовать аналог метода моментов:

$$\vec{g}_{t+1} := \vec{g}_t(1-\alpha) + \alpha f(s,a)\nabla_{\vec{\theta}}\pi_{\vec{\theta}}(a|s)$$

- Это аналог максимизации логарифма правдоподобия:

$$\sum_t f(s_t, a_t)\ln\pi_{\vec{\theta}}(a_t|s_t) \to \max_{\vec{\theta}}$$

Обучение с подкреплением с моделированием среды

- Вводится параметрическая модель среды $(r_t, s_{t+1}) = \mu(s_t, a_t; \vec{\theta})$.
- В ряде случаев эта модель известна.

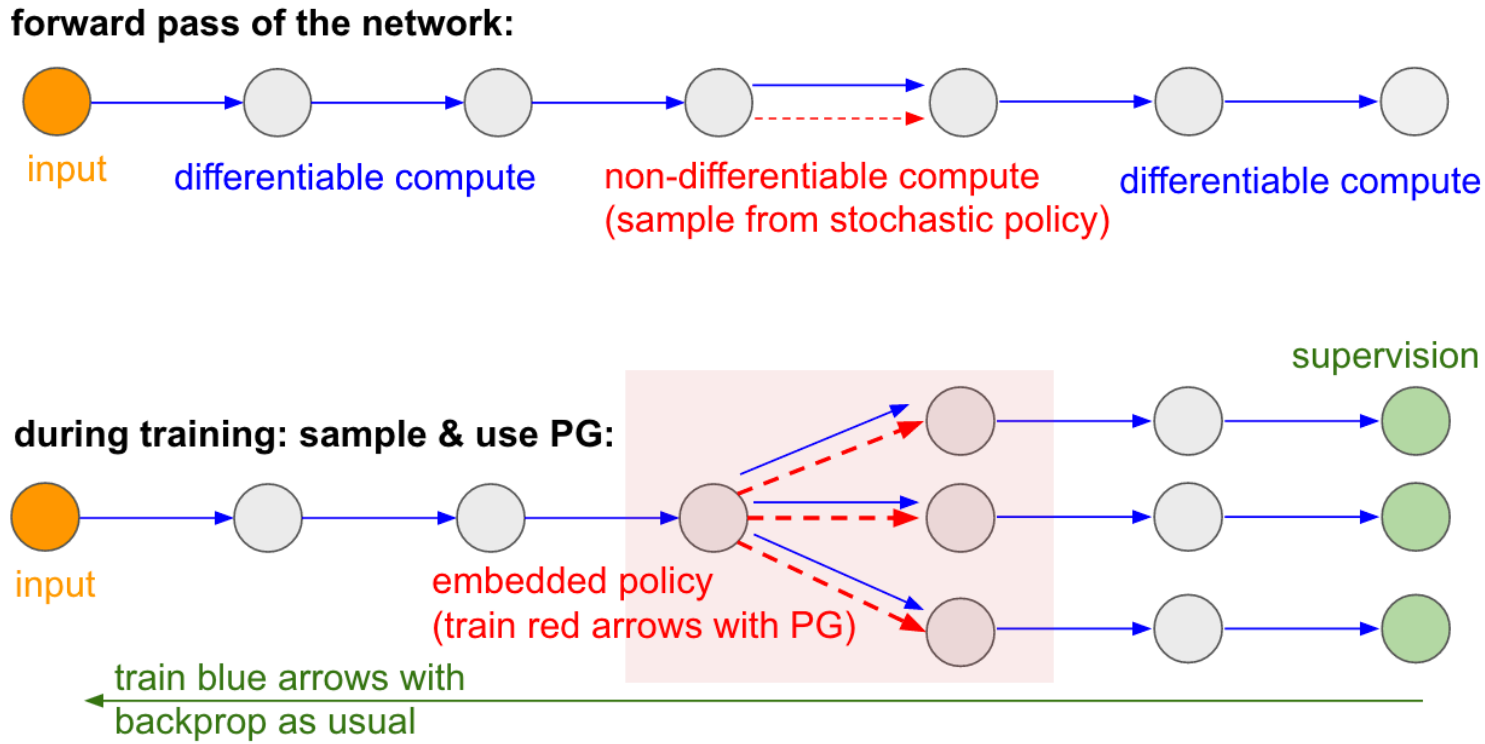
Обучение с подкреплением с моделированием среды

- Вводится параметрическая модель среды $(r_t, s_{t+1}) = \mu(s_t, a_t; \vec{\theta})$.
- В ряде случаев эта модель известна.
- Требуются большие выборки для построения моделей в случае сложных сред.

Обучение с подкреплением с моделированием среды

- Вводится параметрическая модель среды $(r_t, s_{t+1}) = \mu(s_t, a_t; \vec{\theta})$.
- В ряде случаев эта модель известна.
- Требуются большие выборки для построения моделей в случае сложных сред.
- Стратегия может оптимизироваться под модельную среду, а не под настоящую.

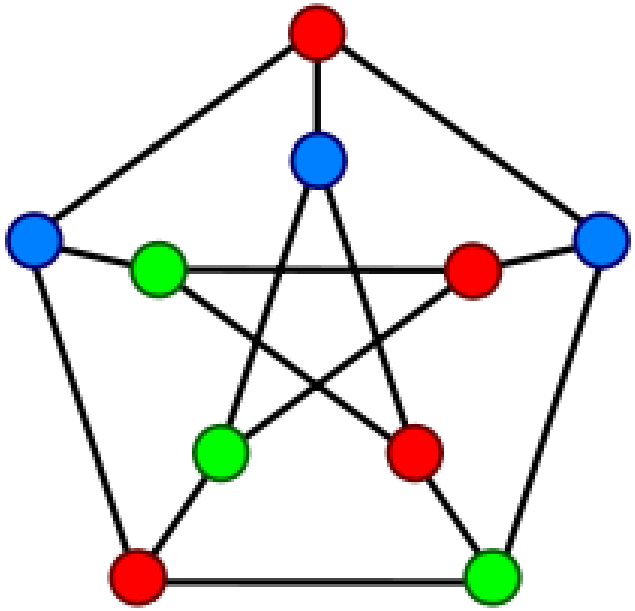
Обучение с подкреплением позволяет вводить недифференцируемые блоки



https://karpathy.github.io/2016/05/31/rl/

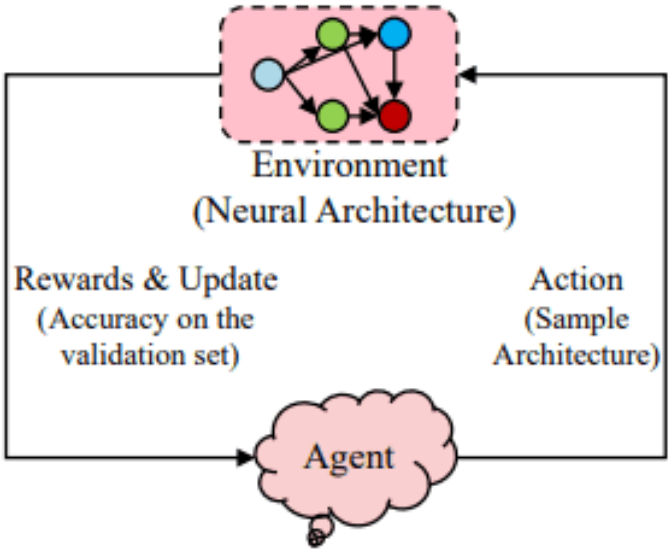
Обучение с подкреплением позволяет вводить недифференцируемые блоки

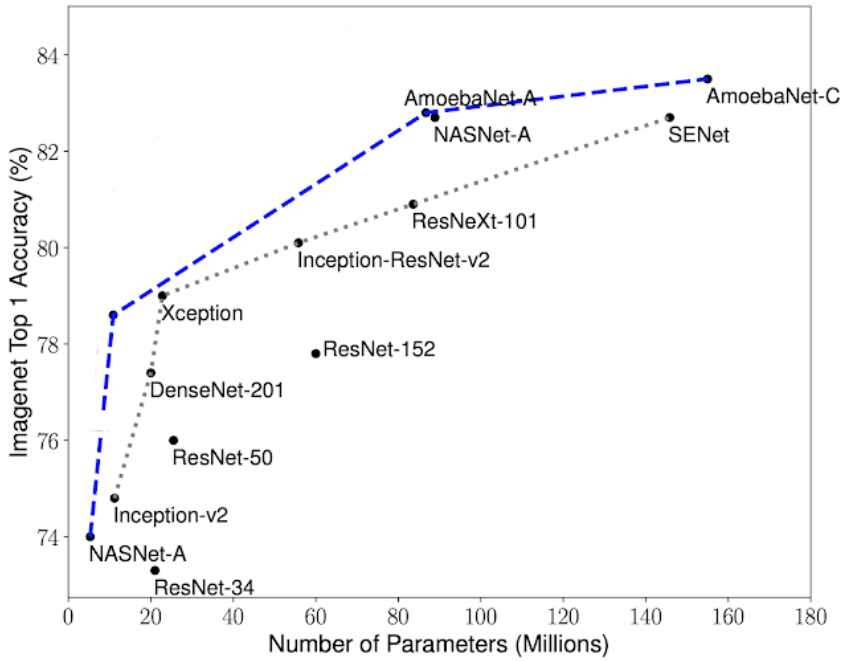
Раскраска графа



Обучение с подкреплением позволяет вводить недифференцируемые блоки

NASNet – нейросеть для проектирования нейросетей





Недостатки обучения с подкреплением

- Требуется очень много данных для обучения.

Недостатки обучения с подкреплением

- Требуется очень много данных для обучения.
- Можно тренировать только небольшие сети.

Недостатки обучения с подкреплением

- Требуется очень много данных для обучения.
- Можно тренировать только небольшие сети.
- Разреженные награды.

Недостатки обучения с подкреплением

- Требуется очень много данных для обучения.
- Можно тренировать только небольшие сети.
- Разреженные награды.
- Rewards hacking.

Недостатки обучения с подкреплением

- Требуется очень много данных для обучения.
- Можно тренировать только небольшие сети.
- Разреженные награды.
- Rewards hacking.
- В ряде случаев обычные алгоритмы лучше.