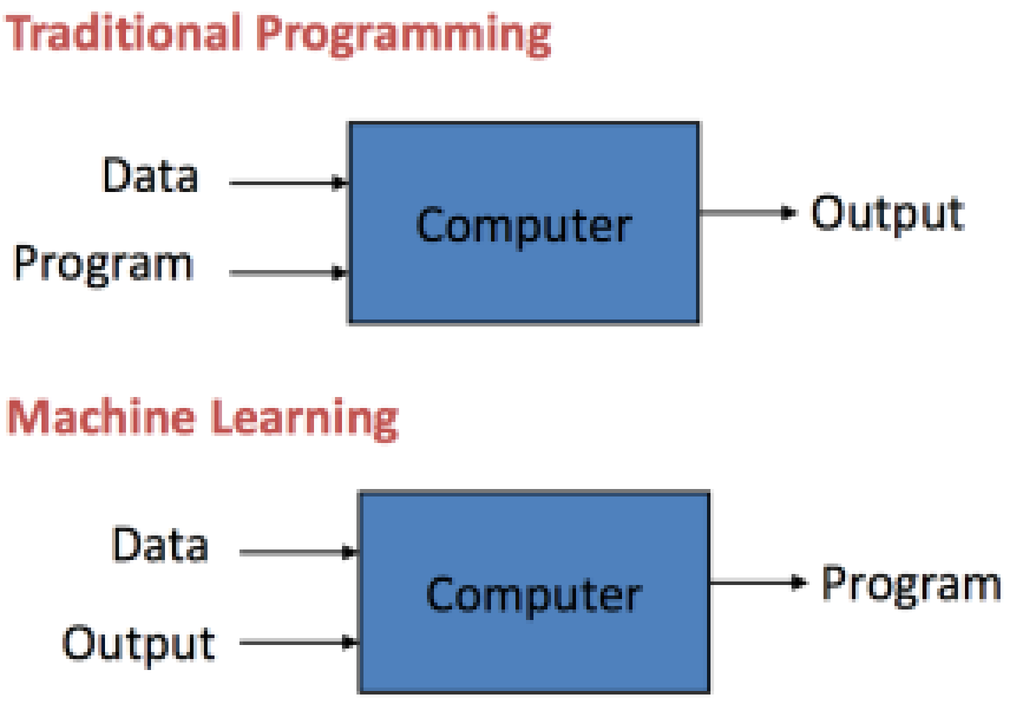
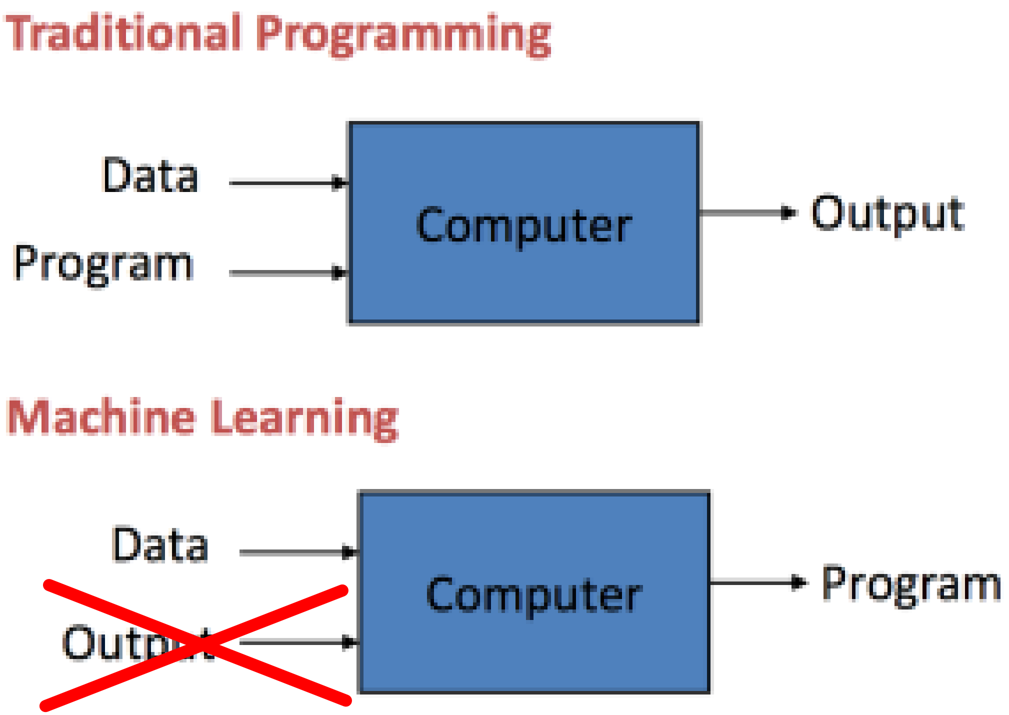
Основы машинного обучения

Программирование и машинное обучение



Программирование и машинное обучение



Обучение с учителем
Линейная регрессия

- Входные данные:

$$\vec{X} = (x_1, x_2, x_{3,}\ \dots)$$

Обучение с учителем
Линейная регрессия

- Входные данные:

$$\vec{X} = (x_1, x_2, x_{3,} \ \dots)$$

- Выходные данные:

$$\vec{y} = (y_1, y_2, y_{3,} \ \dots)$$

Обучение с учителем
Линейная регрессия

- Входные данные:

$$\vec{X} = (x_1, x_2, x_{3,}\ \dots)$$

- Выходные данные:

$$\vec{y} = (y_1, y_2, y_{3,}\ \dots)$$

- Рассмотрим случай

$$\vec{y} = (y_1\ ) = y$$

Обучение с учителем
Линейная регрессия

- Входные данные:

$$\vec{X} = (x_1, x_2, x_{3,} \dots)$$

- Выходные данные:

$$\vec{y} = (y_1, y_2, y_{3,} \dots)$$

- Рассмотрим случай

$$\vec{y} = (y_1) = y$$

- Набор данных:

$$\vec{X}_i : y_i$$

Задача машинного обучения

- На основе входных данных вида:

$$\vec{X}_i : y_i$$

- Построить функцию f:

$$y_i = f(\vec{X}_i)$$

Задача машинного обучения

- Какие требования к функции $f(\vec{X}_i)$?

Задача машинного обучения

- Какие требования к функции $f(\vec{X}_i)$?

- $f(\vec{X}_i)$ должна предсказывать ответы для всех корректных входных данных, и эти ответы должны быть как можно ближе к «правильным»

Задача машинного обучения

- Какие требования к функции $f(\vec{X}_i)$?
- $f(\vec{X}_i)$ должна предсказывать ответы для всех корректных входных данных, и эти ответы должны быть как можно ближе к «правильным»
- Что такое «правильные» ответы и как сравнивать какой из двух ответов ближе к правильному?

Функция потерь
Loss function

- Пусть мы знаем правильный ответ y_i^*

Функция потерь
Loss function

- Пусть мы знаем правильный ответ y_i^*
- Как определить какой из ответов y_i^1 или y_i^2 ближе к правильному?

Функция потерь
Loss function

- Пусть мы знаем правильный ответ y_i^*
- Как определить какой из ответов y_i^1 или y_i^2 ближе к правильному?
- Зададим критерий: функцию $L\left(y_i^k, y_i^*\right)$
  - $L(a, b) > 0$ для различных a и b
  - $L(a, a) = 0$

Функция потерь
Loss function

- Чем более похожи y_i и y_i^* тем меньше должна быть $L(y_i, y_i^*)$

Функция потерь
Loss function

- Чем более похожи y_i и y_i^* тем меньше должна быть $L(y_i, y_i^*)$
- «Более правильный» ответ – у которого меньше $L(y_i^k, y_i^*)$

Функция потерь
Loss function

- Свойства:
  - $L(a, b) > 0$ для различных a и b
  - $L(a, a) = 0$

Функция потерь
Loss function

- Свойства:
  - $L(a, b) > 0$ для различных a и b
  - $L(a, a) = 0$
- Примеры

Функция потерь
Loss function

- Свойства:
  - $L(a, b) > 0$ для различных a и b
  - $L(a, a) = 0$
- Примеры:
  - $L(a, b) = |a - b|$

Функция потерь
Loss function

- Свойства:
  - $L(a,b) > 0$ для различных a и b
  - $L(a,a) = 0$
- Примеры:
  - $L(a,b) = |a - b|$
  - $L(a,b) = (\mathrm{a} - \mathrm{b})^2$

Функция потерь
Loss function

- Свойства:
  - $L(a, b) > 0$ для различных a и b
  - $L(a, a) = 0$
- Примеры:
  - $L(a, b) = |a - b|$
  - $L(a, b) = (\mathrm{a} - \mathrm{b})^2$
  - $L(\vec{a}, \vec{b}) = |\vec{a} - \vec{b}|$

Функция потерь для набора
Loss function

- Функция потерь для множества пар:

$$L_{\text{общая}} = ?$$

Функция потерь для набора
Loss function

- Функция потерь для множества пар:

$$L_{\text{общая}} = \frac{\sum_i L(y_i, y_i^*)}{N}$$

Функция потерь для набора
Loss function

- Функция потерь для множества пар:

$$L_{\text{общая}} = \frac{\sum_i L(y_i, y_i^*)}{N} = \frac{\sum_i L\left(f(\vec{X}_i), y_i^*\right)}{N}$$

Функция потерь для набора
Loss function

- Функция потерь для множества пар:

$$L_{\text{общая}} = \frac{\sum_i L(y_i, y_i^*)}{N} = \frac{\sum_i L\left(f(\vec{X}_i), y_i^*\right)}{N}$$

- Теперь мы можем сравнивать между собой различные функции f на входных данных

Функция потерь для набора Loss function

- Если мы подберем f^* так, что значение $L_{\text{общей}}$ будет достаточно маленьким, то f будет давать ответы, близкие к правильным, на большинстве входных данных

Функция потерь для набора Loss function

- Если мы подберем f^* так, что значение $L_{\text{общей}}$ будет достаточно маленьким, то f будет давать ответы, близкие к правильным, на большинстве входных данных

- Но будет ли она давать близкие к правильным ответы на новых данных?

Построение модели

- Предположение:

Если у нас есть большой и разнообразный набор входных данных и мы найдем функцию f, которая будет давать на нем маленькую ошибку, то она будет давать «хорошие» результаты и на новых входных данных

Построение модели

- Нужны «новые» данные с известными ответами, которые мы не использовали при построении модели

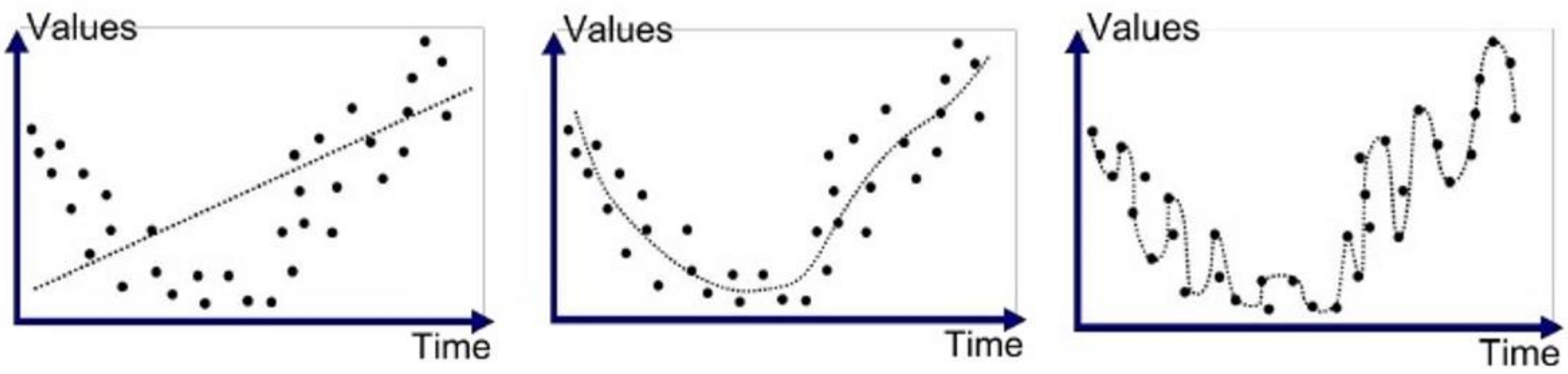
Построение модели

- Нужны «новые» данные с известными ответами, которые мы не использовали при построении модели
- Делим исходный набор на два:
  - Тренировочный
  - Тестовый

Построение модели

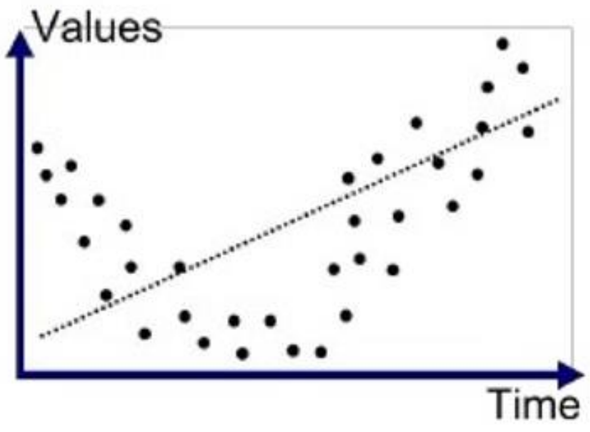
- Нужны «новые» данные с известными ответами, которые мы не использовали при построении модели
- Делим исходный набор на два:
  - Тренировочный
  - Тестовый
- Обучаем на тренировочном наборе, а затем проверяем на тестовом насколько хорошо функция обобщает зависимость

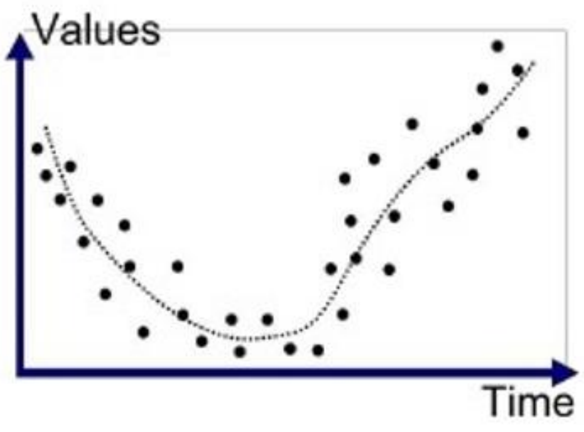
Построение модели

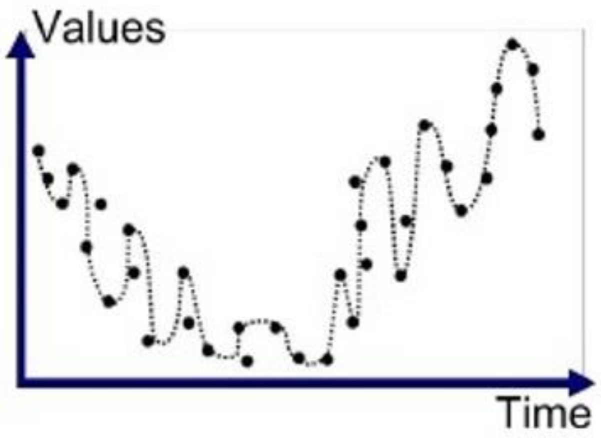


- Рисунок из https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76

Построение модели







Недообучение **Хорошая обобщающая способность** **Переобучение**

- Рисунок из https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76

Линейная регрессия

- Функция f:

$$y_i = f(\vec{X}_i)$$

Линейная регрессия

- Функция f:

$$y_i = f(\vec{X}_i)$$

- Линейная функция f:
  - $y = \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3 + \cdots b$

Линейная регрессия

- Функция f:

$$y_i = f(\vec{X}_i)$$

- Линейная функция f:
  - $y = \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3 + \cdots b$
- $L(y, y^*) = (y - y^*)^2$

Линейная регрессия

- Функция f:

$$y_i = f(\vec{X}_i)$$

- Линейная функция f:
  - $y = \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3 + \cdots b$
- $L(y, y^*) \ = (y - y^*)^2 =$
  $= (\omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3 + \cdots b \ - y^*)^2$

Линейная регрессия

- Цель:
  - Подобрать веса ω_1, ω_2 и т.д. что бы минимизировать значение $L_{общей}$

Линейная регрессия

- Цель:
  - Подобрать веса ω_1, ω_2 и т.д. что бы минимизировать значение $L_{\text{общей}}$
- $L_{\text{общая}} = \frac{\sum_i L\left(f(\overrightarrow{X}_i), y_i^*\right)}{N}$
-

Линейная регрессия

- Цель:
  - Подобрать веса ω_1, ω_2 и т.д. что бы минимизировать значение $L_{\text{общей}}$
- $L_{\text{общая}} = \frac{\sum_i L\left(f(\overrightarrow{X}_i), y_i^*\right)}{N}$
- $L_{\text{общая}} = \frac{\sum_i(\omega_1 x_{i1} + \omega_2 x_{i2} + \omega_3 x_{i3} + \cdots b - y_i^*)^2}{N}$

Минимум функции

Минимум функции

- Через производные:

Минимум функции

- Через производные:
- $\frac{df}{dx} = 0$

Минимум функции

- Через производные:
- $\frac{df}{dx} = 0$
- Случай многих переменных

Минимум функции

- Через производные:
- $$\frac{df}{dx} = 0$$
- Случай многих переменных получается система:
- $\frac{df}{d\omega_1} = 0, \frac{df}{d\omega_2} = 0, \frac{df}{d\omega_3} = 0$ и т.д.

Если параметров очень много

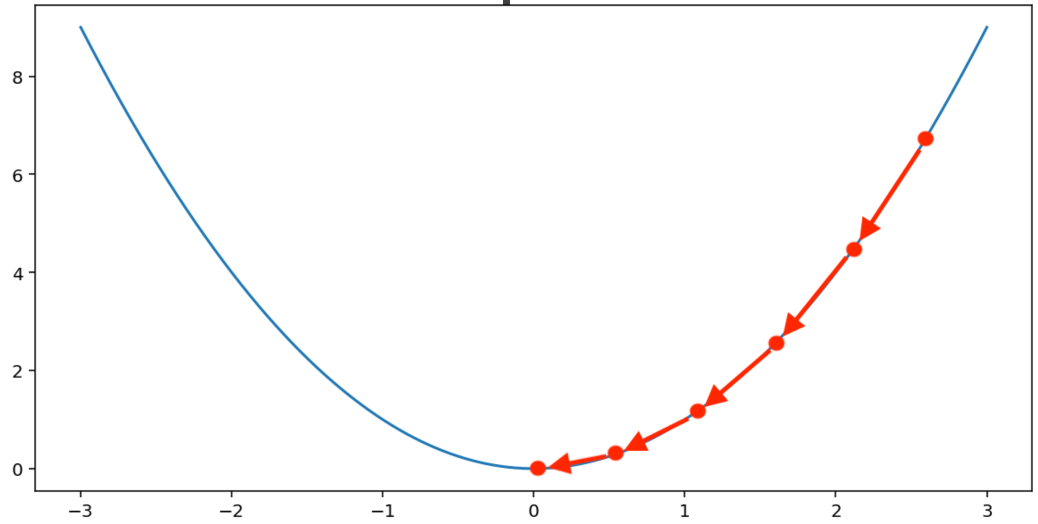
Если параметров очень много

- Решить систему слишком сложно/невозможно

Если параметров очень много

- Решить систему слишком сложно/невозможно
- Можно искать итеративно:

$L_{общая}$



ω_1

- https://medium.com/@congyuzhou/градиентный-спуск-7f9fa996e883

Если параметров очень много

- Обновление на каждой итерации:

$$w := w - \alpha \frac{\partial \mathcal{J}(w,b)}{\partial w}$$

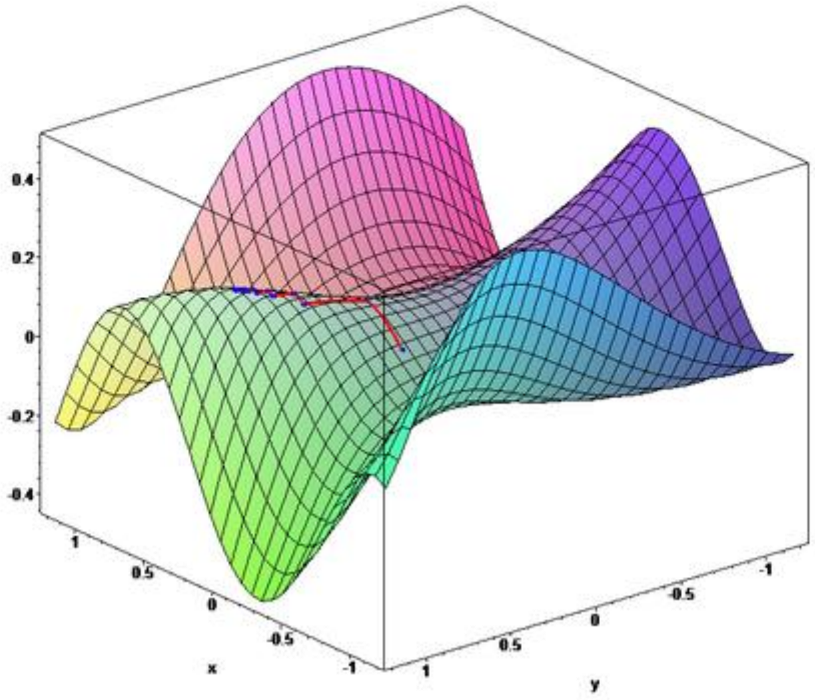
$$b := b - \alpha \frac{\partial \mathcal{J}(w,b)}{\partial b}$$

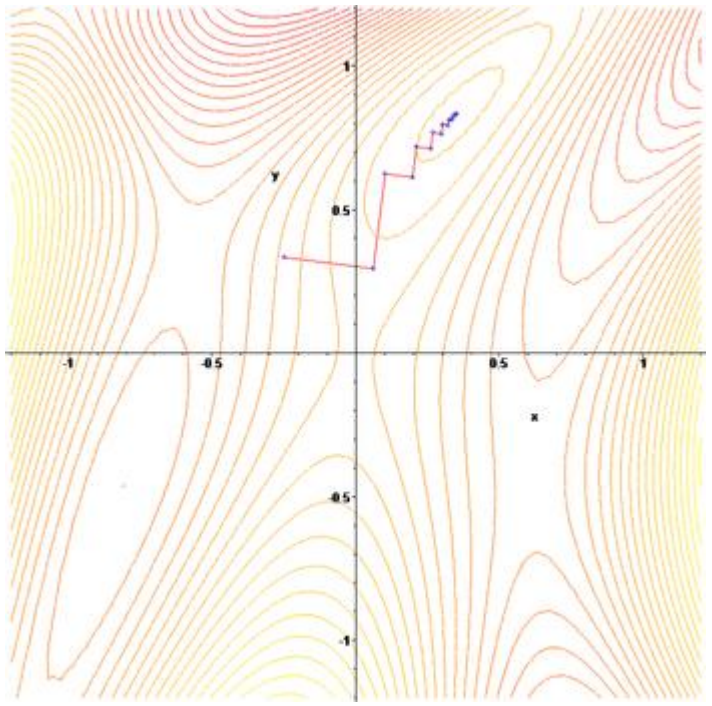
https://medium.com/@congyuzhou/градиентный-спуск-7f9fa996e883

- α – малый параметр, определяющий величину шага

Если параметров очень много

- Можно искать подбором:





- https://ru.wikipedia.org/wiki/Градиентный_спуск