Лекция 5
Нейронные сети

Бручес Елена

15.07.2025

Основные методы в NLP

- Правила/эвристики;
- Статистические методы;
- Методы машинного обучения

Машинное обучение

Машинное обучение – это область компьютерные наук, которая изучает алгоритмы, которые анализируют данные и делают предсказания, основываясь на этих данных.

Приложения машинного обучения:

- NLP;
- Computer Vision;
- Optical Character Recognition (OCR);
- Business analytics;
- etc

Machine Learning

Machine Learning tasks:

- **Supervised learning**: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs;
- **Unsupervised learning**: No labels are given to the learning algorithm, leaving it on its own to find structure in its input;
- **Reinforcement learning**: A computer program interacts with a dynamic environment in which it must perform a certain goal. The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

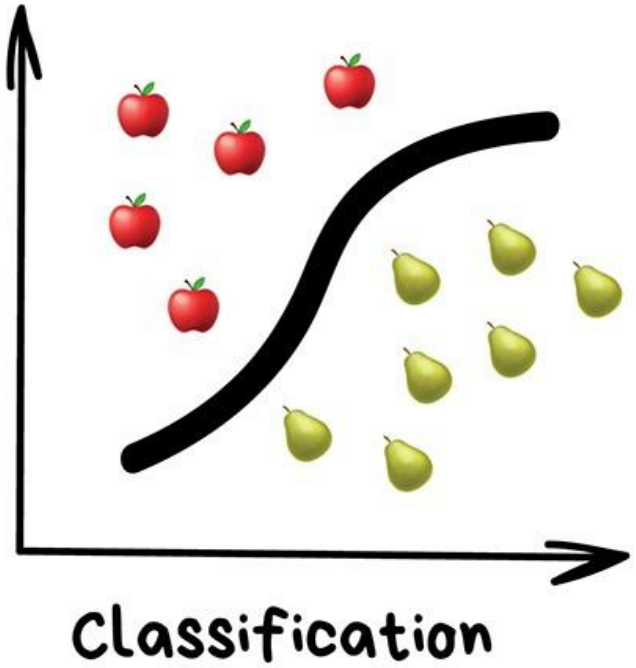
Machine Learning

Machine learning tasks:

- In **classification**, inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes;
- In **regression**, also a supervised problem, the outputs are continuous rather than discrete;
- In **clustering**, a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task;
- **Density estimation** finds the distribution of inputs in some space;
- **Dimensionality reduction** simplifies inputs by mapping them into a lower-dimensional space.

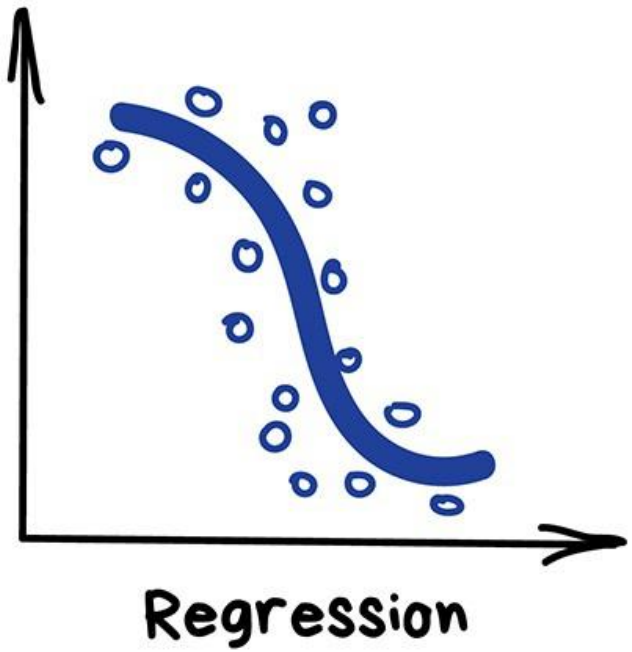
Classification

In **classification**, inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes



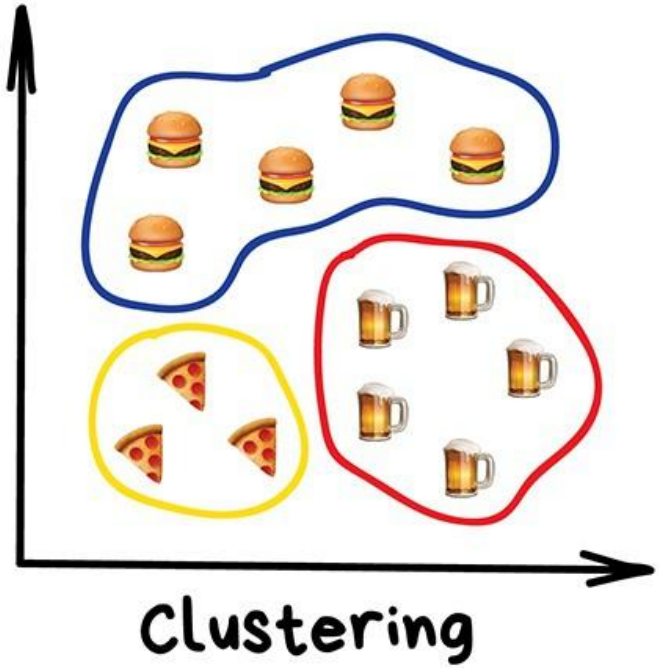
Regression

In **regression**, also a supervised problem, the outputs are continuous rather than discrete



Clustering

In **clustering**, a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task

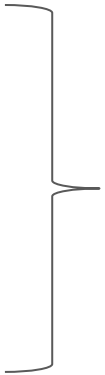


(Feature, Label) - Sample

(Feature, Label) - Sample

….

(Feature, Label) - Sample



Данные

Терминология ML

Label (метка) - это то, что мы предсказываем. Это может быть цена дома, тип животного на картинке, часть речи слова - в общем, почти всё что угодно.

Feature (признак) - это входная переменная. Самая простая модель может использовать один единственный признак; более сложные модели могут использовать миллионы признаков.

Например, в задаче определения спам/не спам признаками могут быть:

- слова в тексте письма;
- адрес отправителя;
- время дня, в которое было отправлено письмо;
- наличие в письме фразы "Ваш адрес был выбран победителем";
- и др.

Терминология ML

Sample - это конкретный пример данных. Samples бывают двух видов:

1. labeled (размеченные);
2. unlabeled (неразмеченные).

Размеченный sample включает и признак, и label:

labeled examples: *{features, label}: (x, y)*

Неразмеченный sample содержит только признак:

unlabeled examples: *{features, ?}: (x, ?)*

Итак, предположим, что Вы хотите разработать модель, которая будет предсказывать, является ли письмо спамом или нет. В качестве данных у нас есть набор писем, которые были помечены пользователями как "спам" или "не спам". Какие утверждения верны?

1. Слова в теме письма будут хорошими labels.
2. Вы будете использовать неразмеченные данные, чтобы обучить модель.
3. Те письма, которые не помечены "спам" или "не спам", - неразмеченные примеры.
4. Не все labels, которыми помечены письма, могут быть надёжными.

Итак, предположим, что Вы хотите разработать модель, которая будет предсказывать, является ли письмо спамом или нет. В качестве данных у нас есть набор писем, которые были помечены пользователями как "спам" или "не спам". Какие утверждения верны?

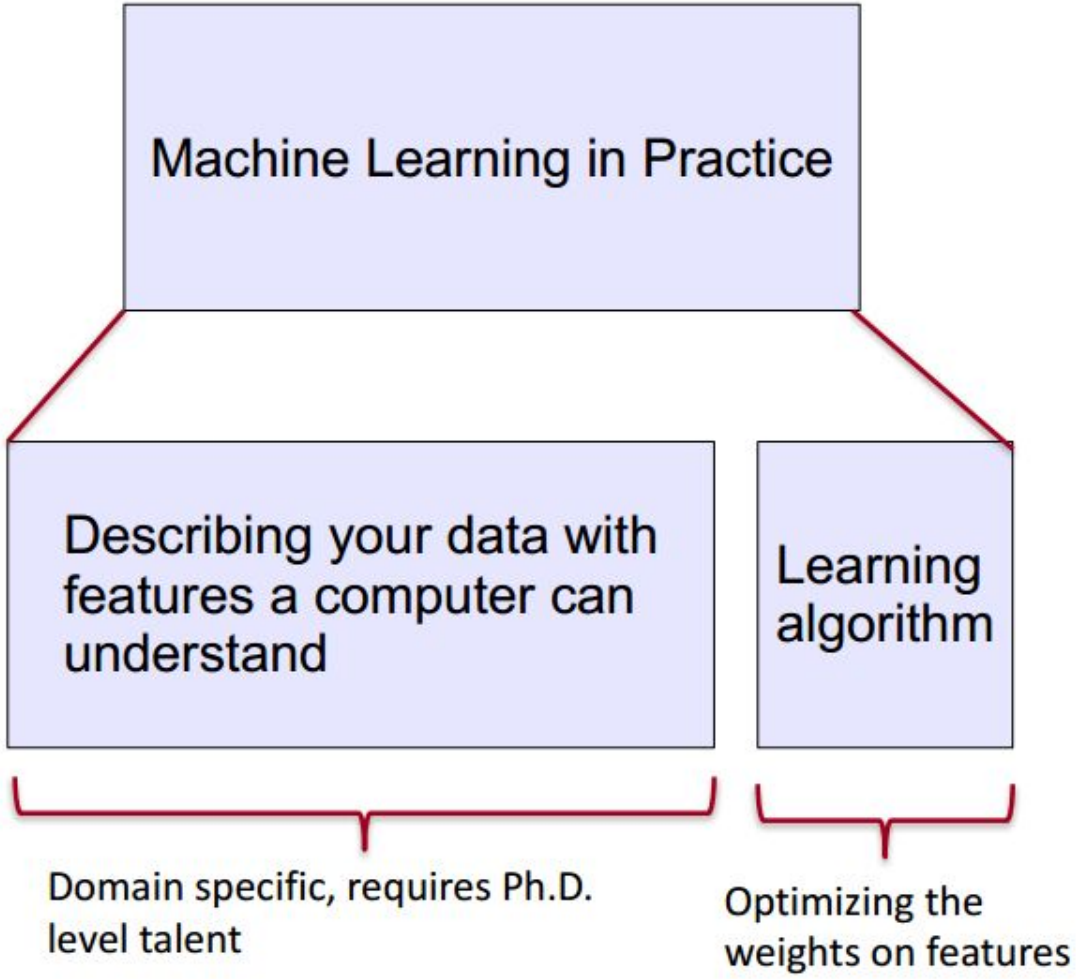
1. Слова в теме письма будут хорошими labels.
2. Вы будете использовать неразмеченные данные, чтобы обучить модель.
3. Те письма, которые не помечены "спам" или "не спам", - неразмеченные примеры.
4. **Не все labels, которыми помечены письма, могут быть надёжными.**

Теперь представим, что обувной он-лайн магазин хочет создать модель, которая будет рекомендовать пользователям обувь. То есть, модель будет рекомендовать определённые пары обуви Кейт и совершенно другие - Джону. Какие утверждения верны?

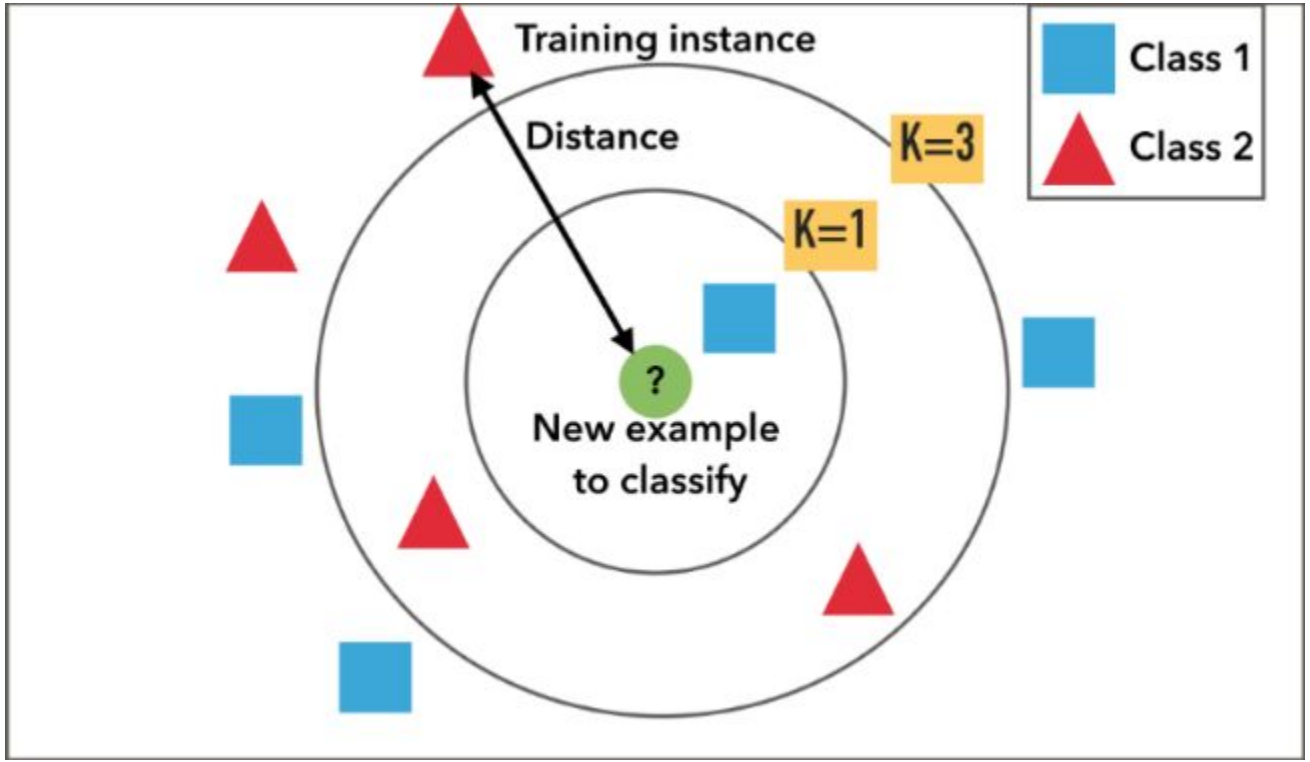
1. "Пользователь кликнул на описание обуви" - хороший label.
2. "Размер обуви" - полезный признак.
3. "Красота обуви" - полезный признак.
4. "Обувь, которую обожает пользователь" - хороший label.

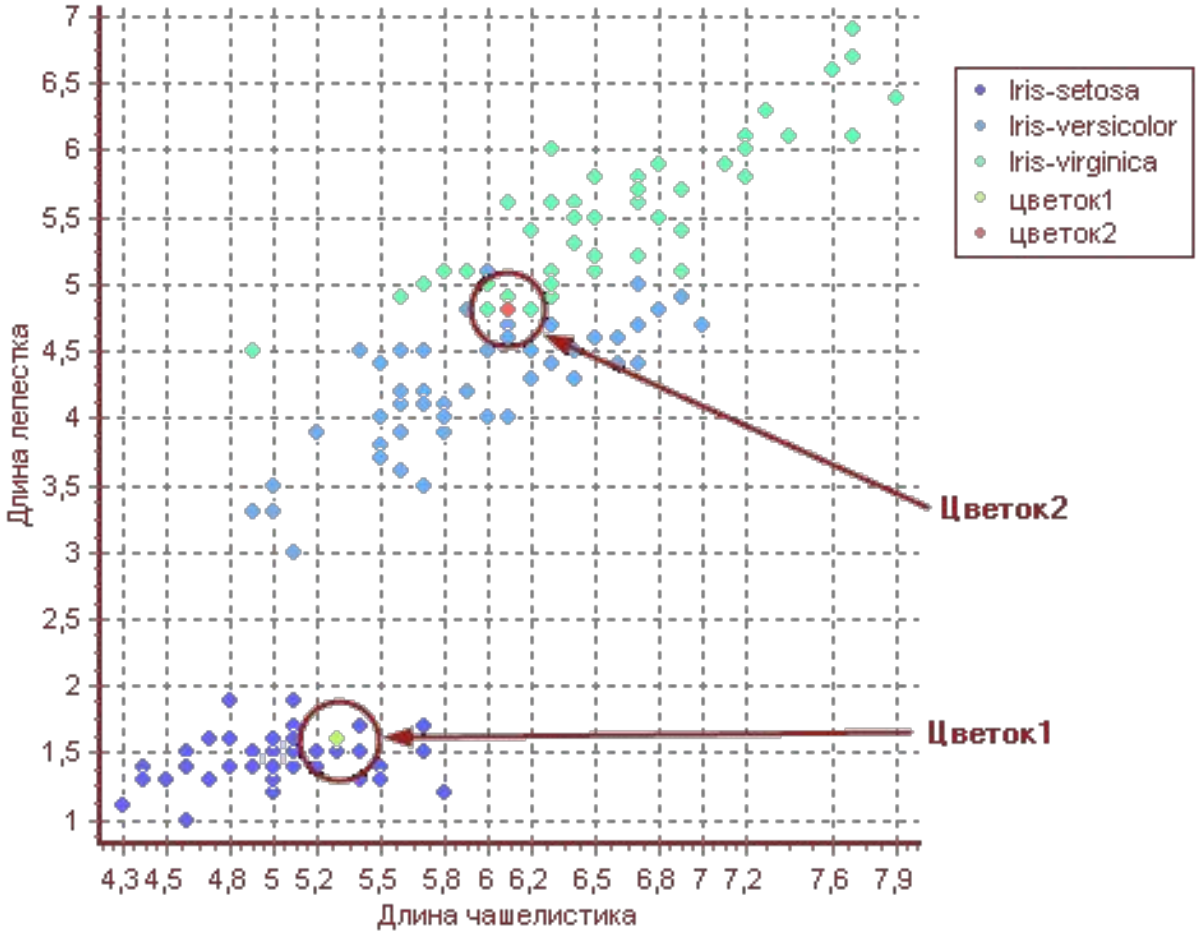
Теперь представим, что обувной он-лайн магазин хочет создать модель, которая будет рекомендовать пользователям обувь. То есть, модель будет рекомендовать определённые пары обуви Кейт и совершенно другие - Джону. Какие утверждения верны?

1. **"Пользователь кликнул на описание обуви" - хороший label.**
2. **"Размер обуви" - полезный признак.**
3. "Красота обуви" - полезный признак.
4. "Обувь, которую обожает пользователь" - хороший label.

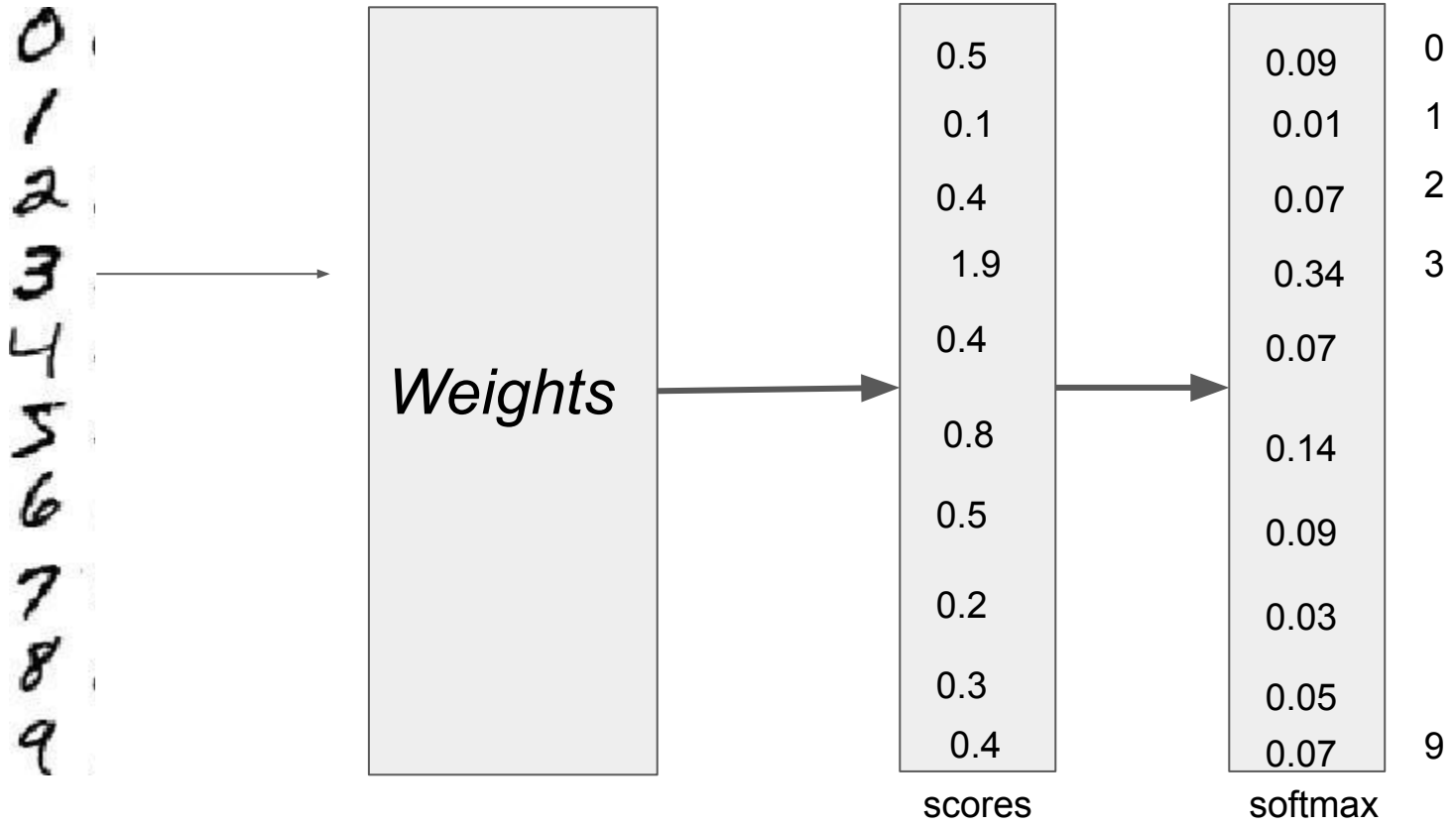

Machine Learning in Practice
Describing your data with features a computer can understand
Learning algorithm
Domain specific, requires Ph.D. level talent
Optimizing the weights on features

K-Nearest Neighbours

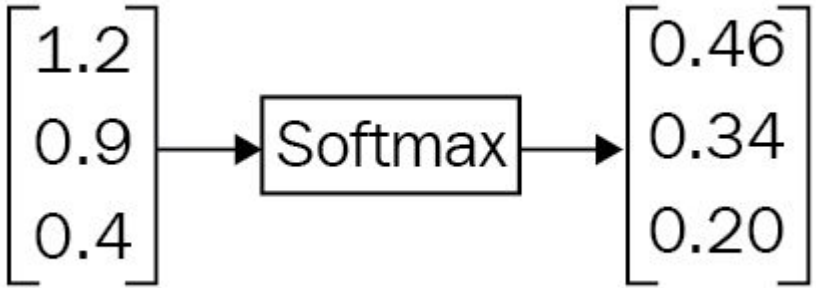



Iris-setosa
Iris-versicolor
Iris-virginica
цветок1
цветок2
Цветок2
Цветок1
Длина лепестка
Длина чашелистика
7
6,5
6
5,5
5
4,5
4
3,5
3
2,5
2
1,5
1
4,3 4,5 4,8 5 5,2 5,5 5,8 6 6,2 6,5 6,8 7 7,2 7,6 7,9

Линейный классификатор



$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$



Обучение модели

Input	Actual output	Desired output
0	0	0
1	3	2
2	6	4
3	9	6
4	12	8

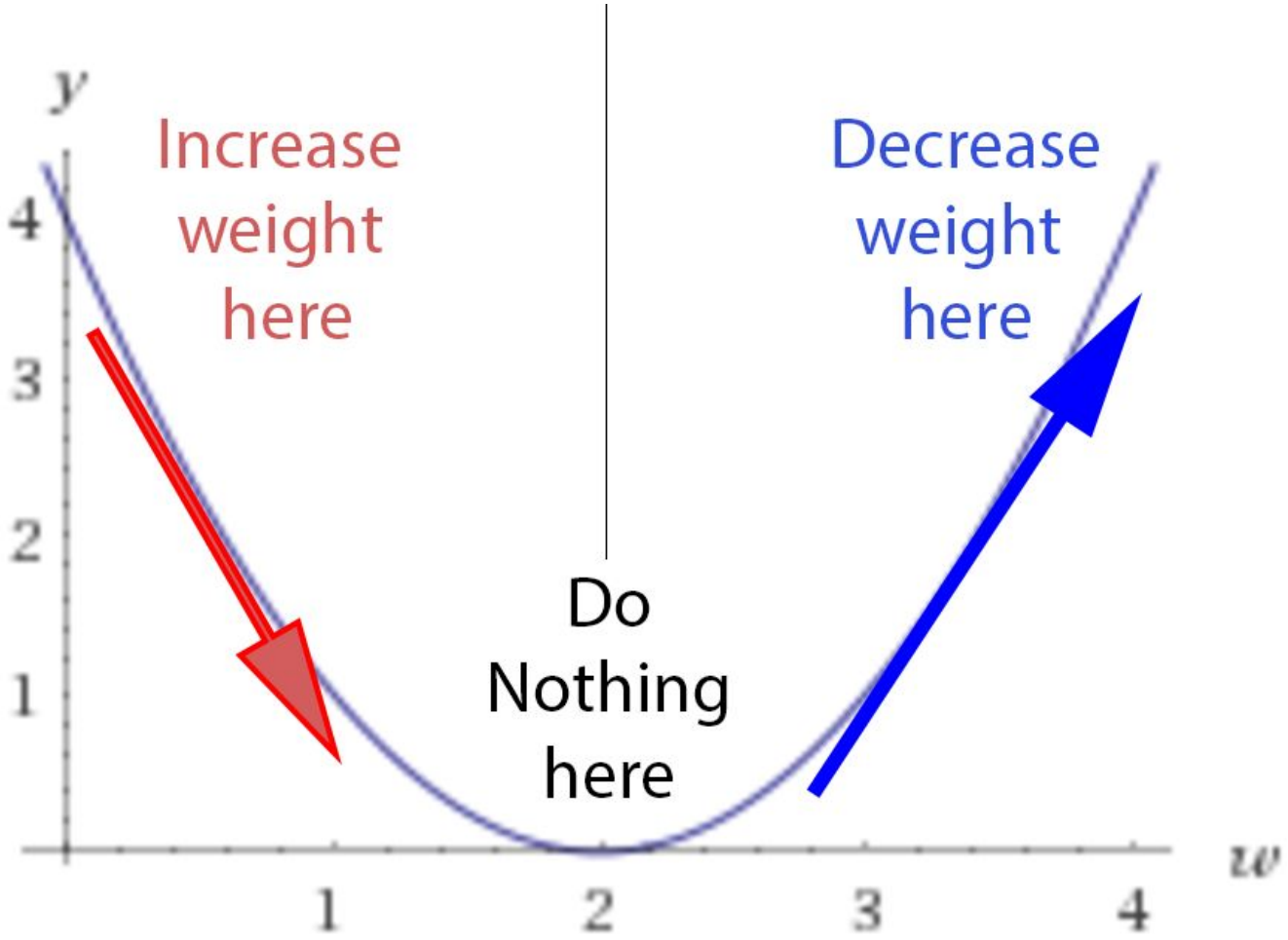
Обучение модели

Input	actual	Desired	Absolute Error	Square Error
0	0	0	0	0
1	3	2	1	1
2	6	4	2	4
3	9	6	3	9
4	12	8	4	16
Total:	-	-	10	30

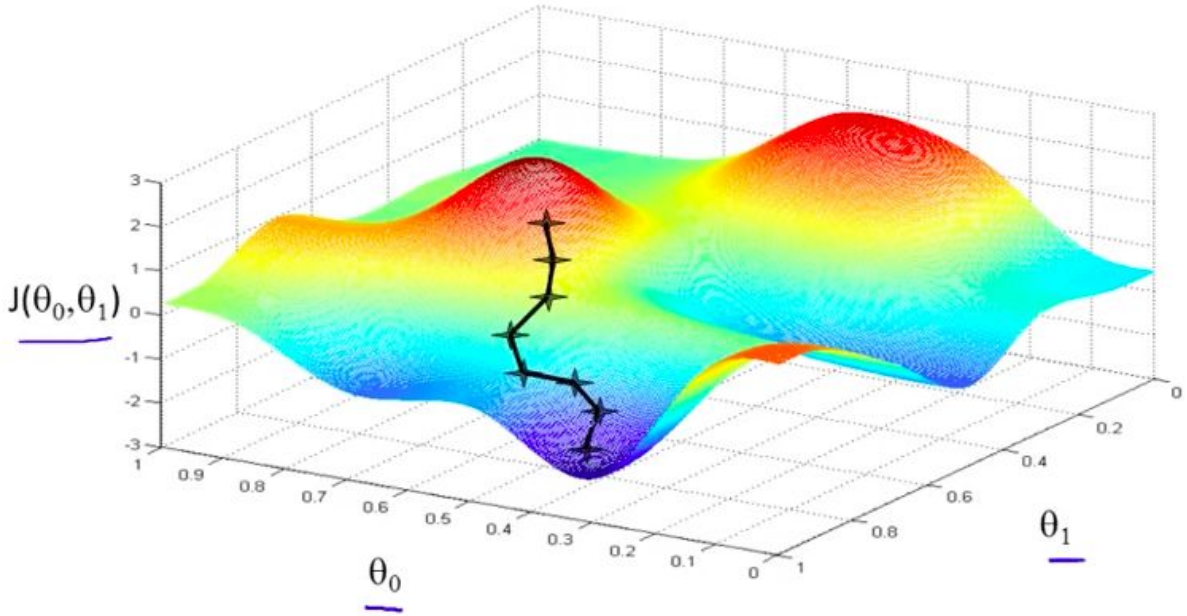
Обучение модели

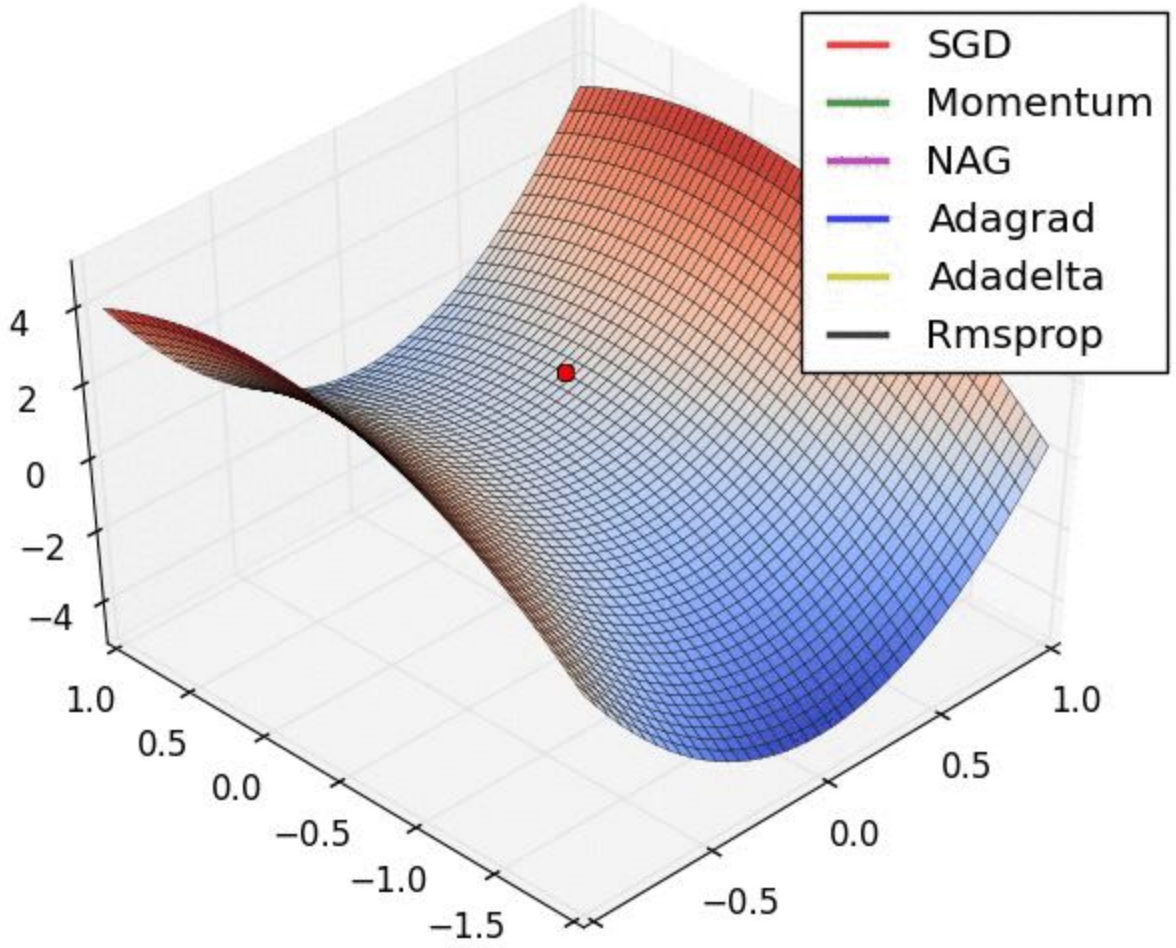
Input	Output	W=3	rmse(3)	W=3.0001	rmse
0	0	0	0	0	0
1	2	3	1	3.0001	1.0002
2	4	6	4	6.0002	4.0008
3	6	9	9	9.0003	9.0018
4	8	12	16	12.0004	16.0032
Total:	-	-	30	-	30.006

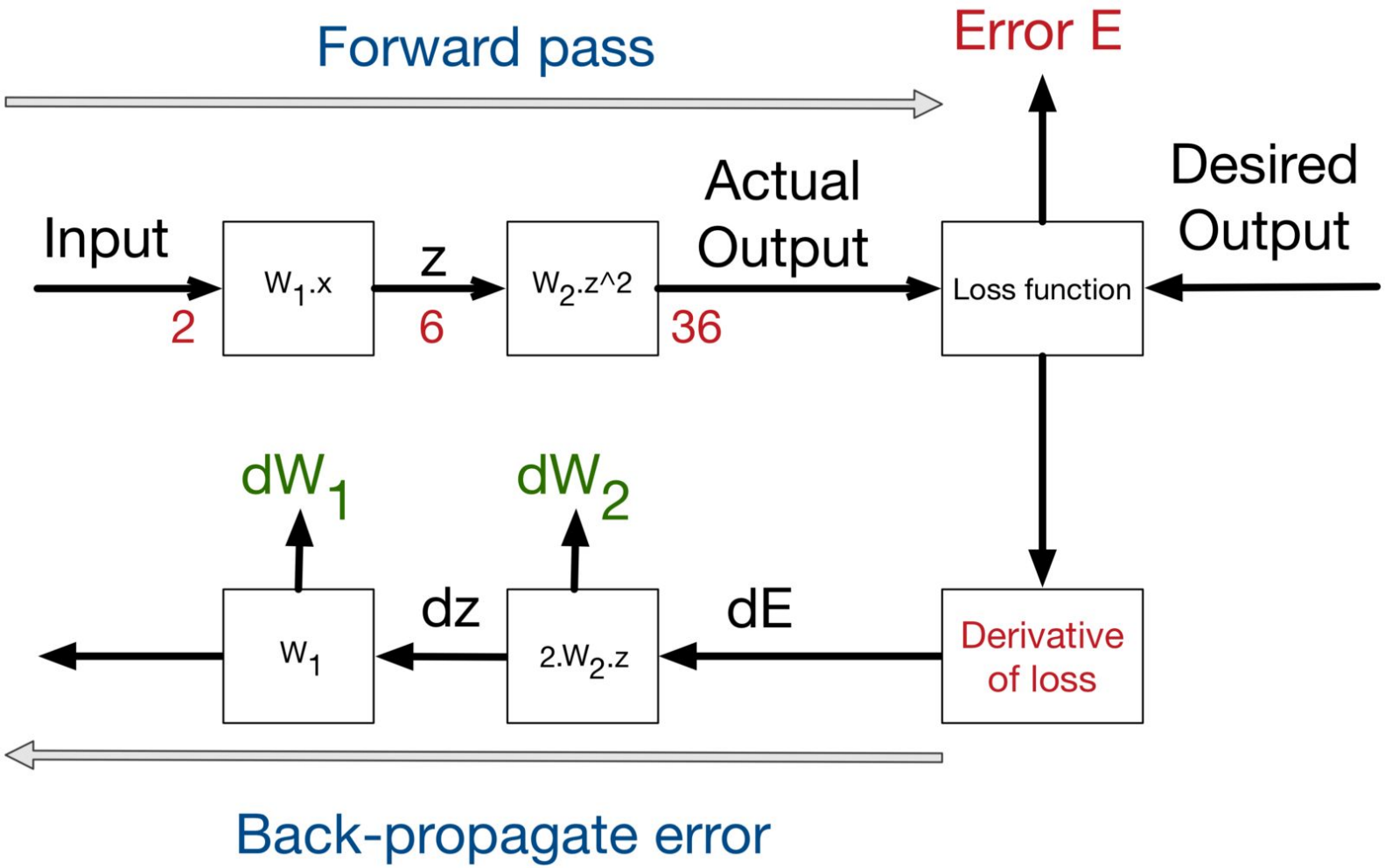
W = 2.9999, rmse - ?

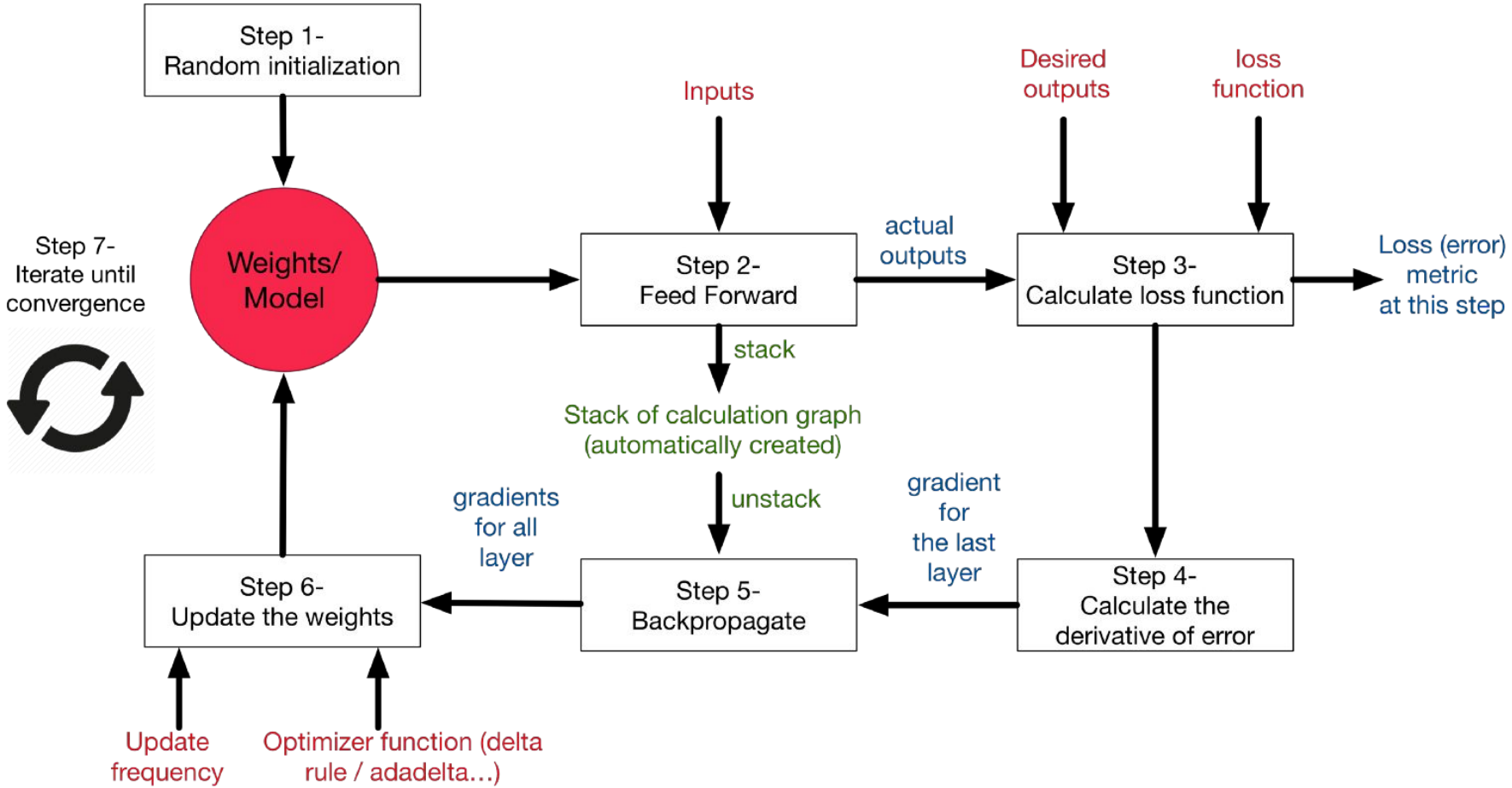

y
Increase weight here
Decrease weight here
4
3
2
1
Do Nothing here
w
1
2
3
4

Стохастический градиентный спуск

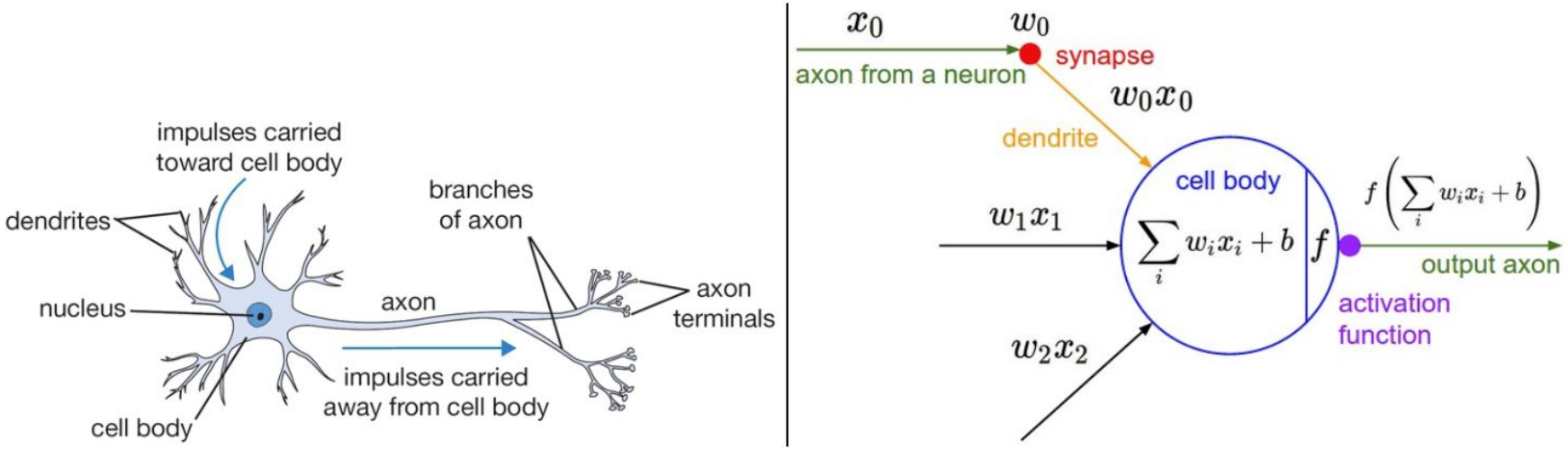



SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop
4
2
0
−2
−4
1.0
0.5
0.0
−0.5
−1.0
−1.5
1.0
0.5
0.0
−0.5

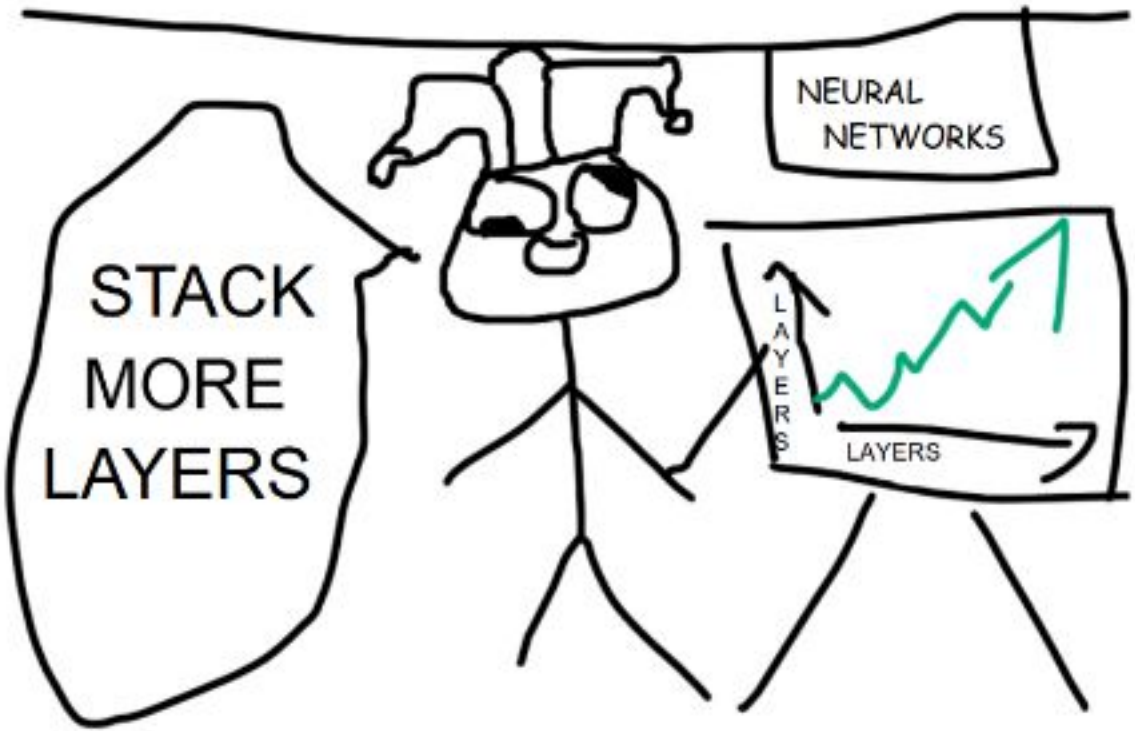

Forward pass
Error E
Input
2
$w_1.x$
z
6
$w_2.z^2$
Actual Output
36
Loss function
Desired Output
dW_1
dW_2
w_1
dz
$2.w_2.z$
dE
Derivative of loss
Back-propagate error


Step 1-
Random initialization
Weights/
Model
Step 7-
Iterate until
convergence
Inputs
Step 2-
Feed Forward
actual
outputs
Desired
outputs
loss
function
Step 3-
Calculate loss function
Loss (error)
metric
at this step
stack
Stack of calculation graph
(automatically created)
unstack
gradients
for all
layer
gradient
for
the last
layer
Step 6-
Update the weights
Step 5-
Backpropagate
Step 4-
Calculate the
derivative of error
Update
frequency
Optimizer function (delta
rule / adadelta…)

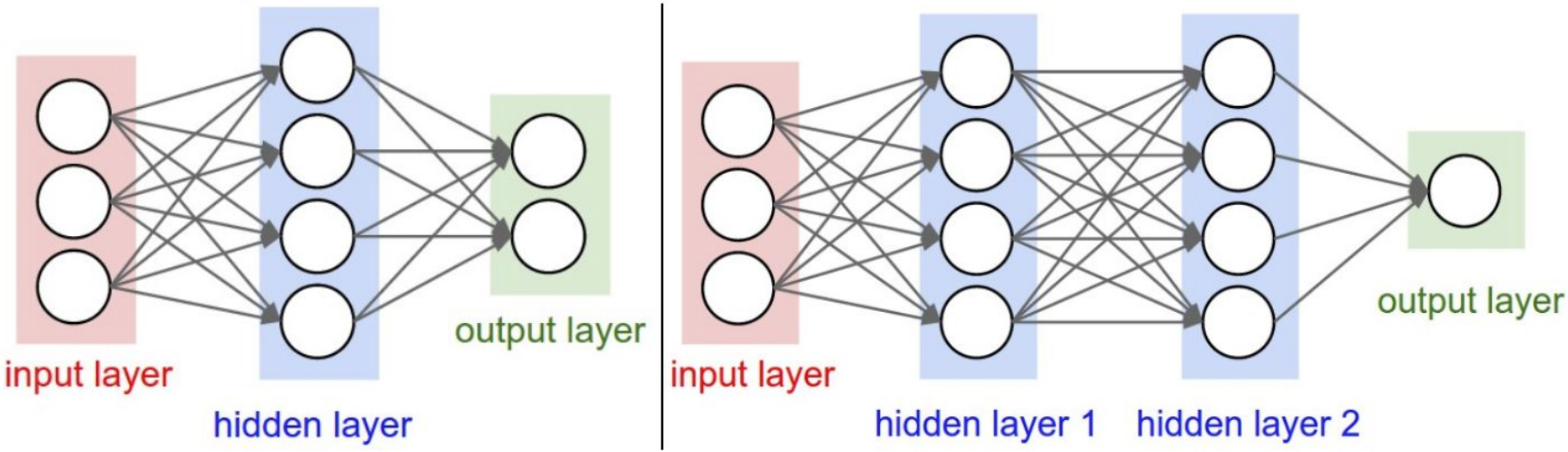
Neural Networks



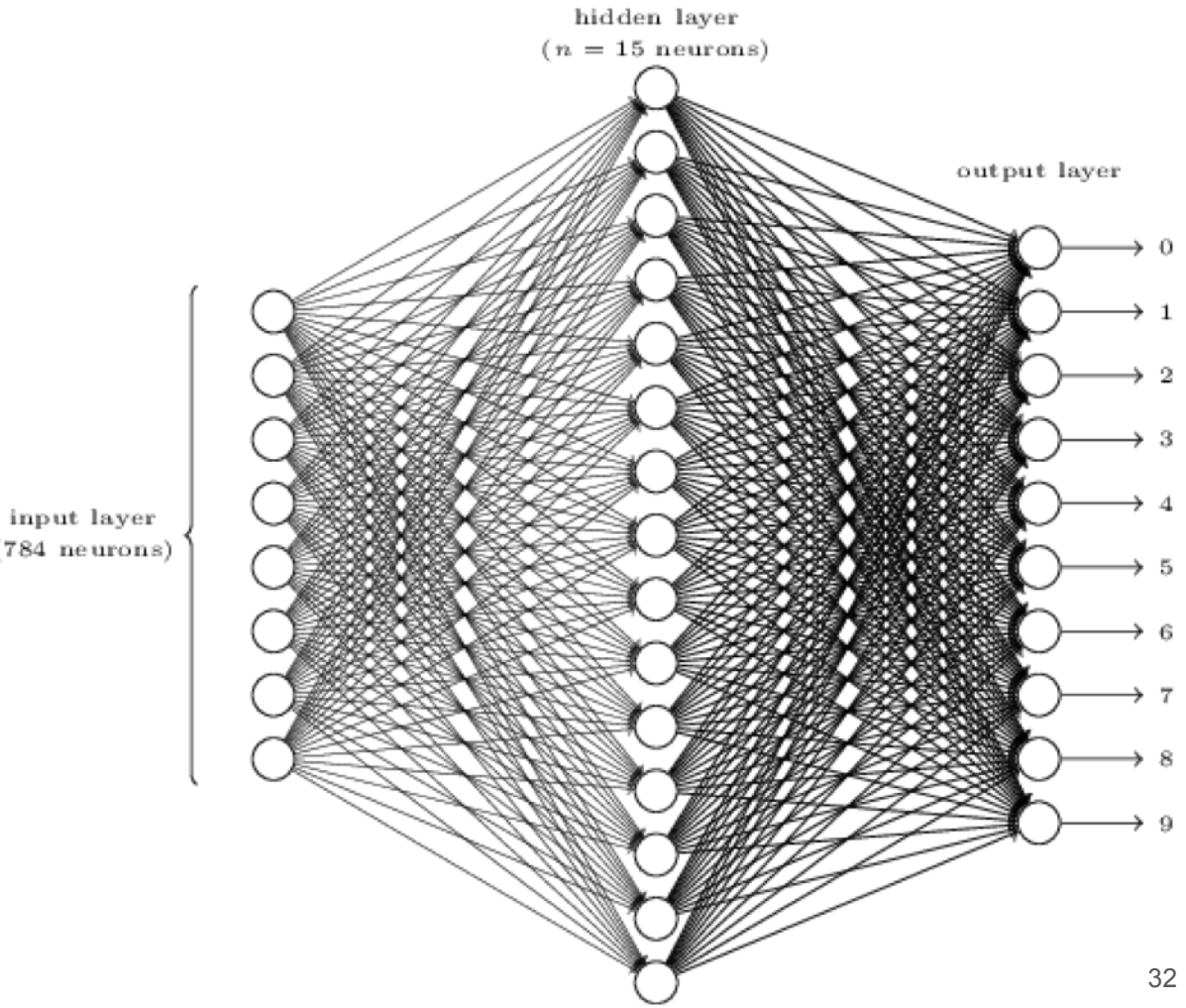
A cartoon drawing of a biological neuron (left) and its mathematical model (right).

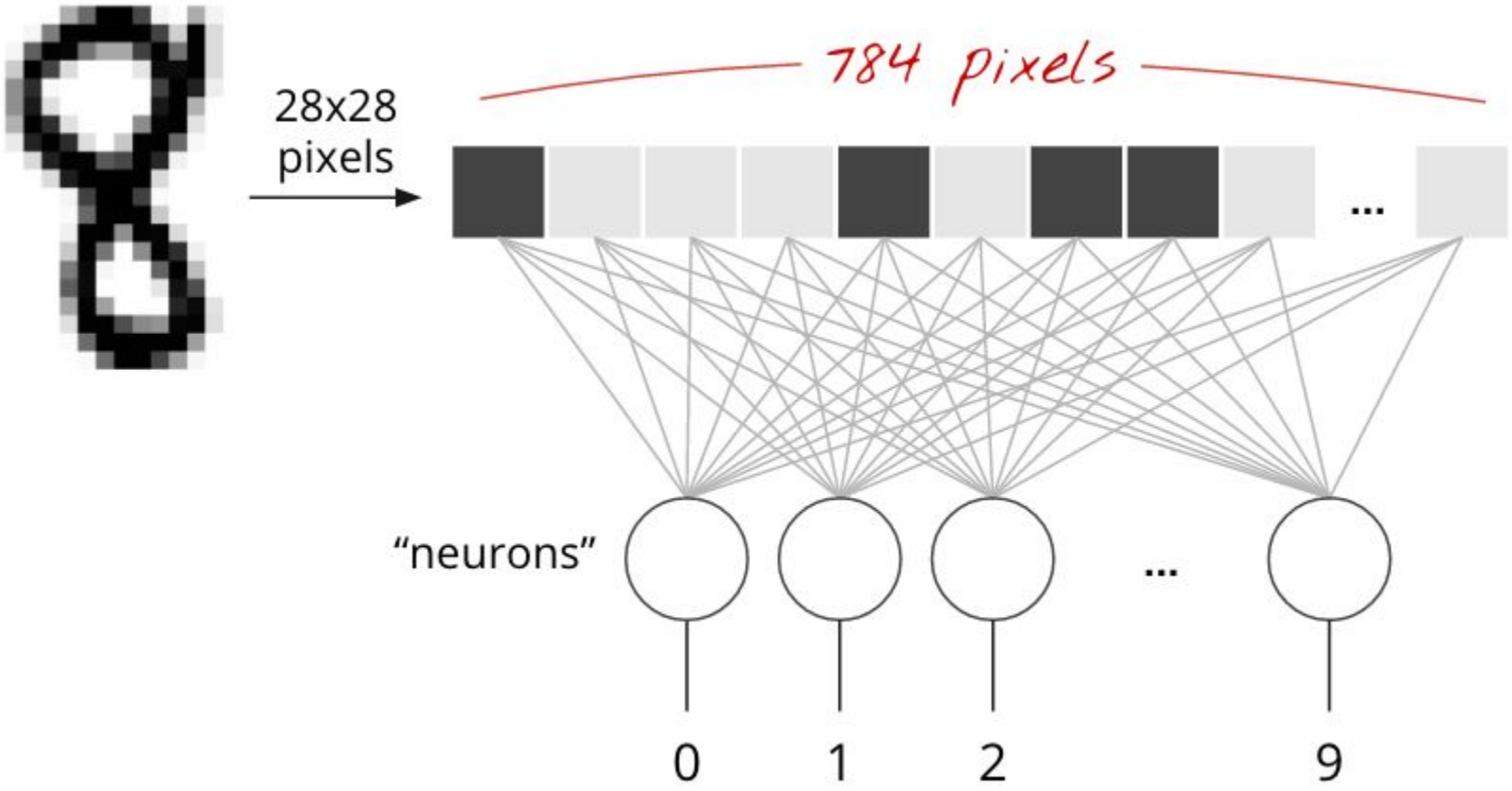

NEURAL
NETWORKS
STACK
MORE
LAYERS
LAYERS
LAYERS

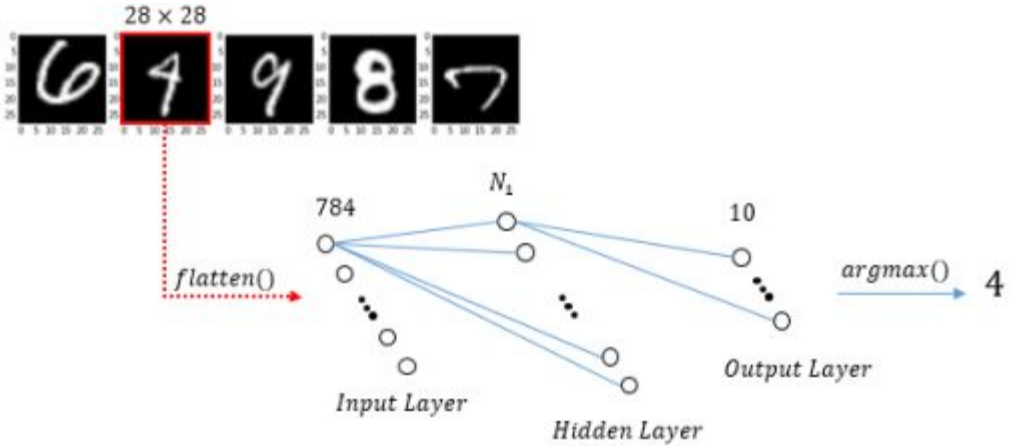
Neural Networks

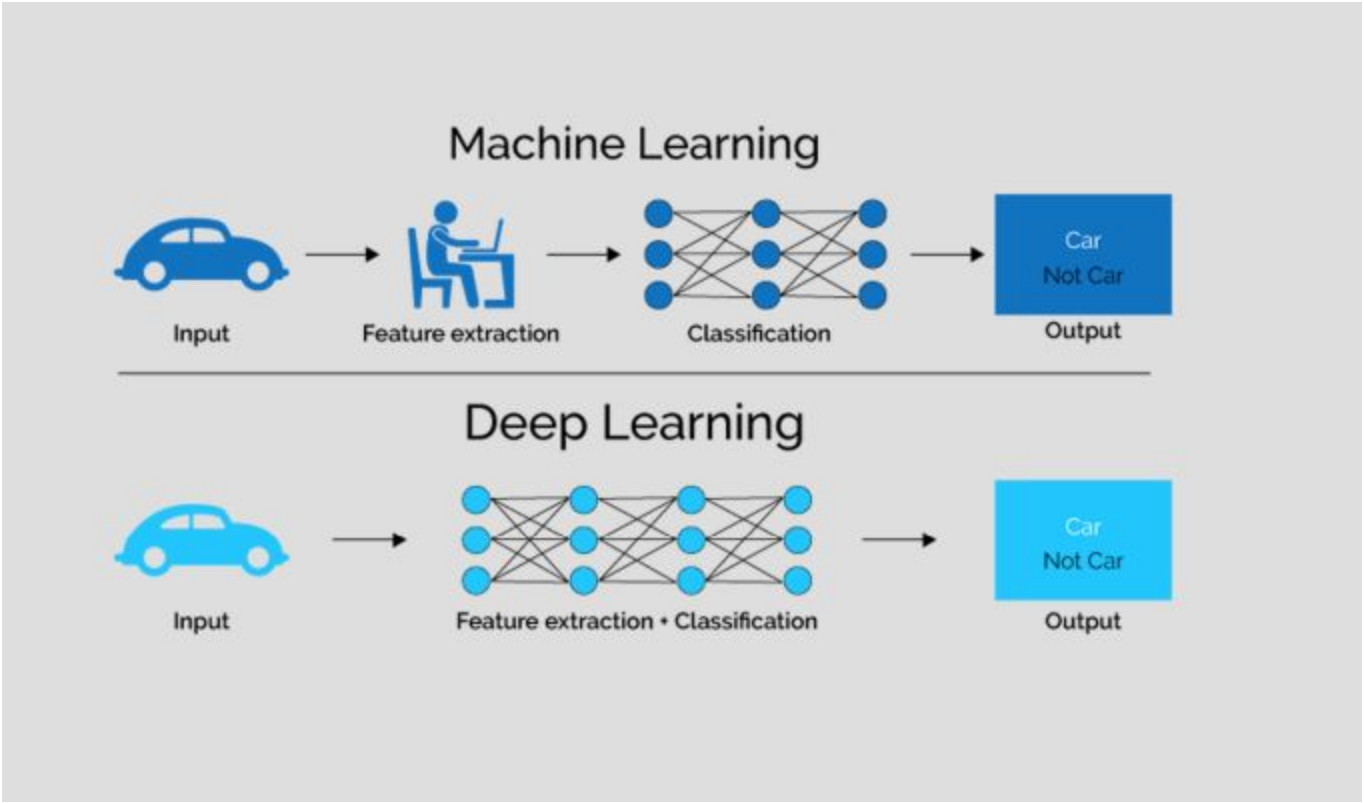


Left: A 2-layer Neural Network (one hidden layer of 4 neurons (or units) and one output layer with 2 neurons), and three inputs.
Right: A 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer. Notice that in both cases there are connections (synapses) between neurons across layers, but not within a layer.


hidden layer
(n = 15 neurons)
output layer
input layer
(784 neurons)
0
1
2
3
4
5
6
7
8
9

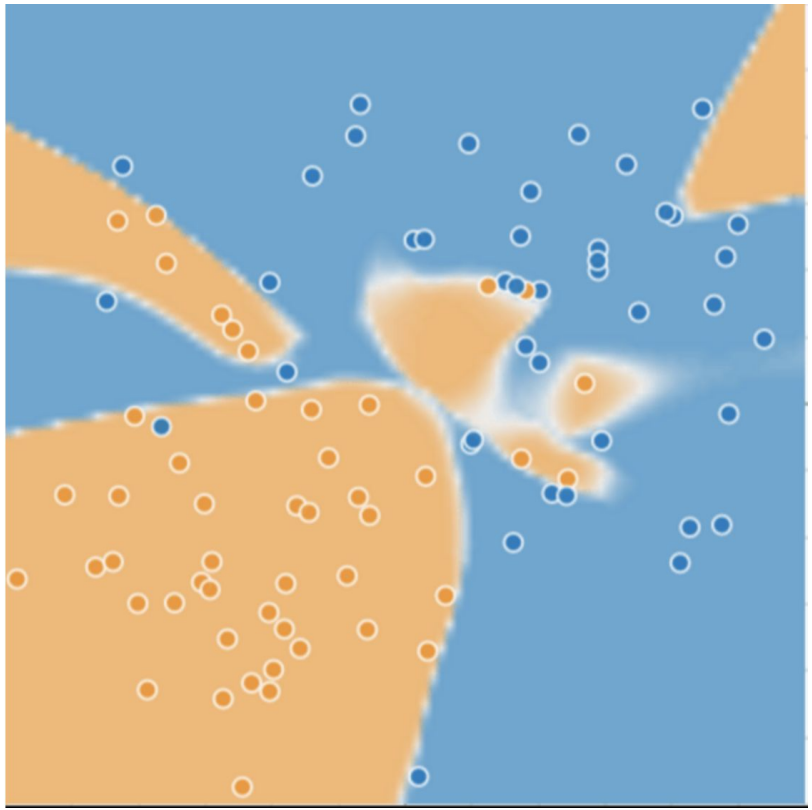

28x28
pixels
784 pixels
...
"neurons"
...
0
1
2
9


28 × 28
flatten()
784
N_1
10
argmax()
4
Input Layer
Hidden Layer
Output Layer

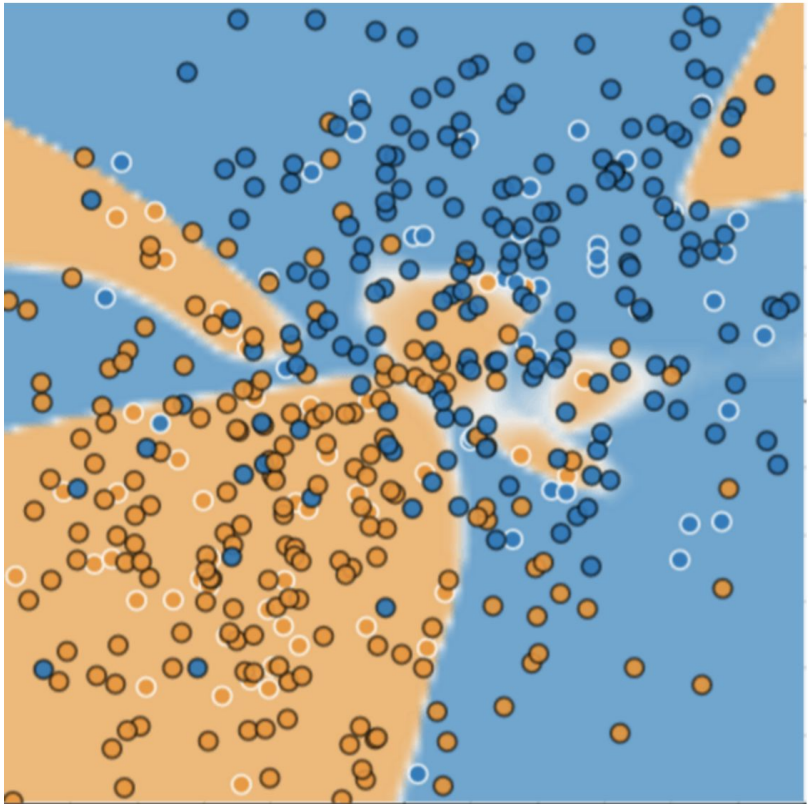

Machine Learning
Input
Feature extraction
Classification
Car
Not Car
Output
Deep Learning
Input
Feature extraction • Classification
Car
Not Car
Output

Overfitting

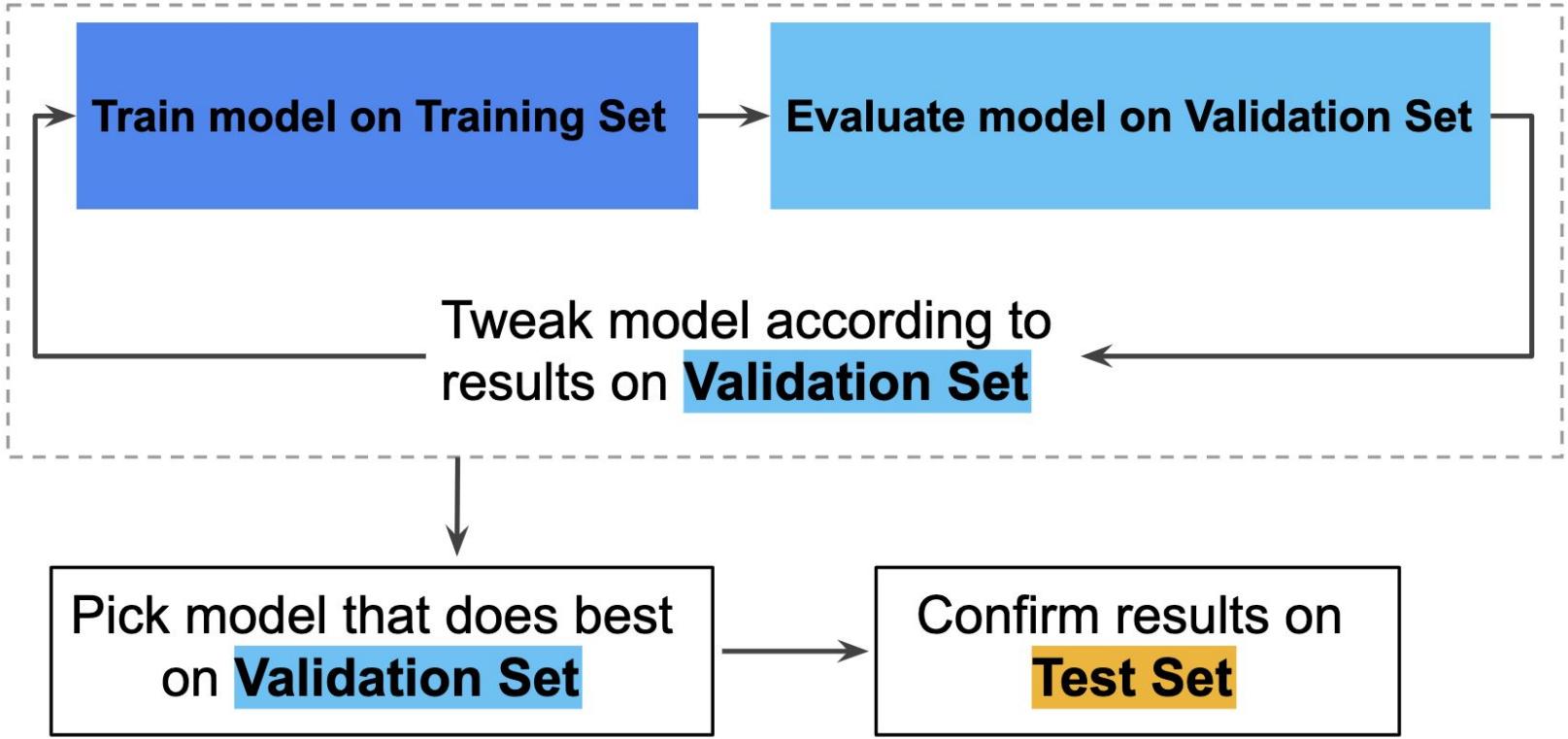
Overfitting



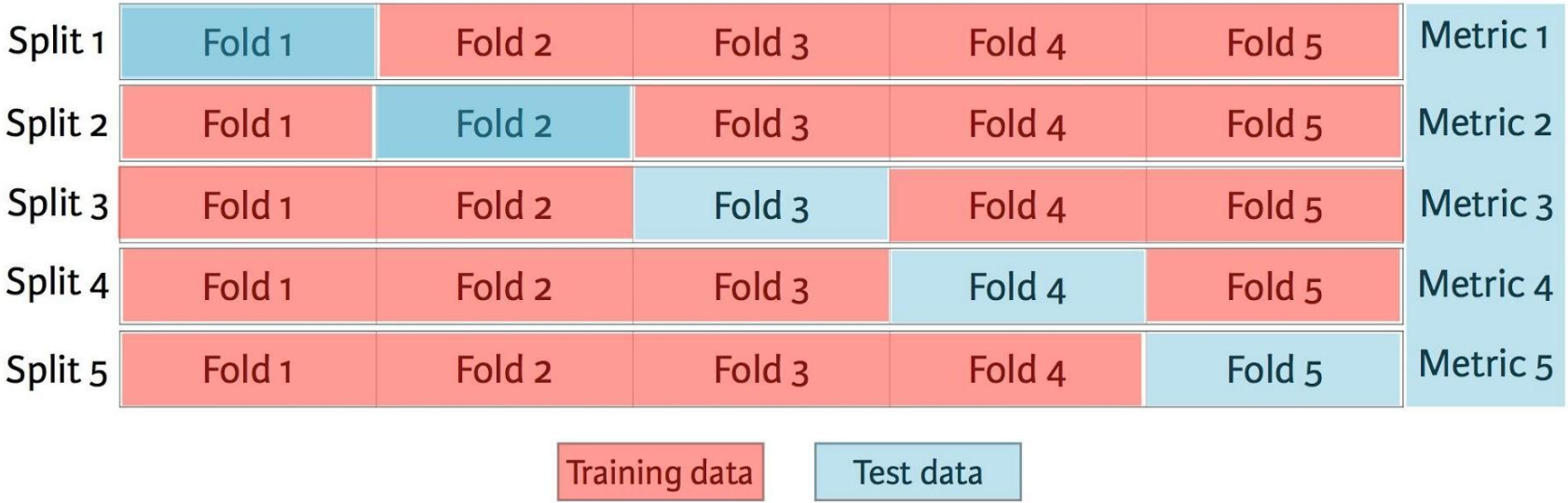
Overfitting



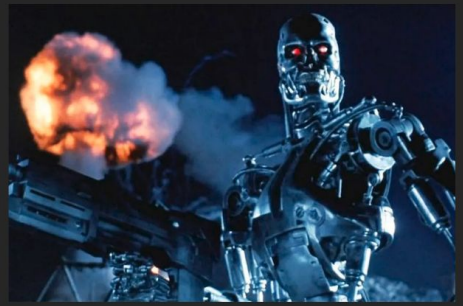
Train/val/test splitting



Cross-validation



Deep Learning



What society thinks I do

What my friends think I do

What other computer scientists think I do

What mathematicians think I do

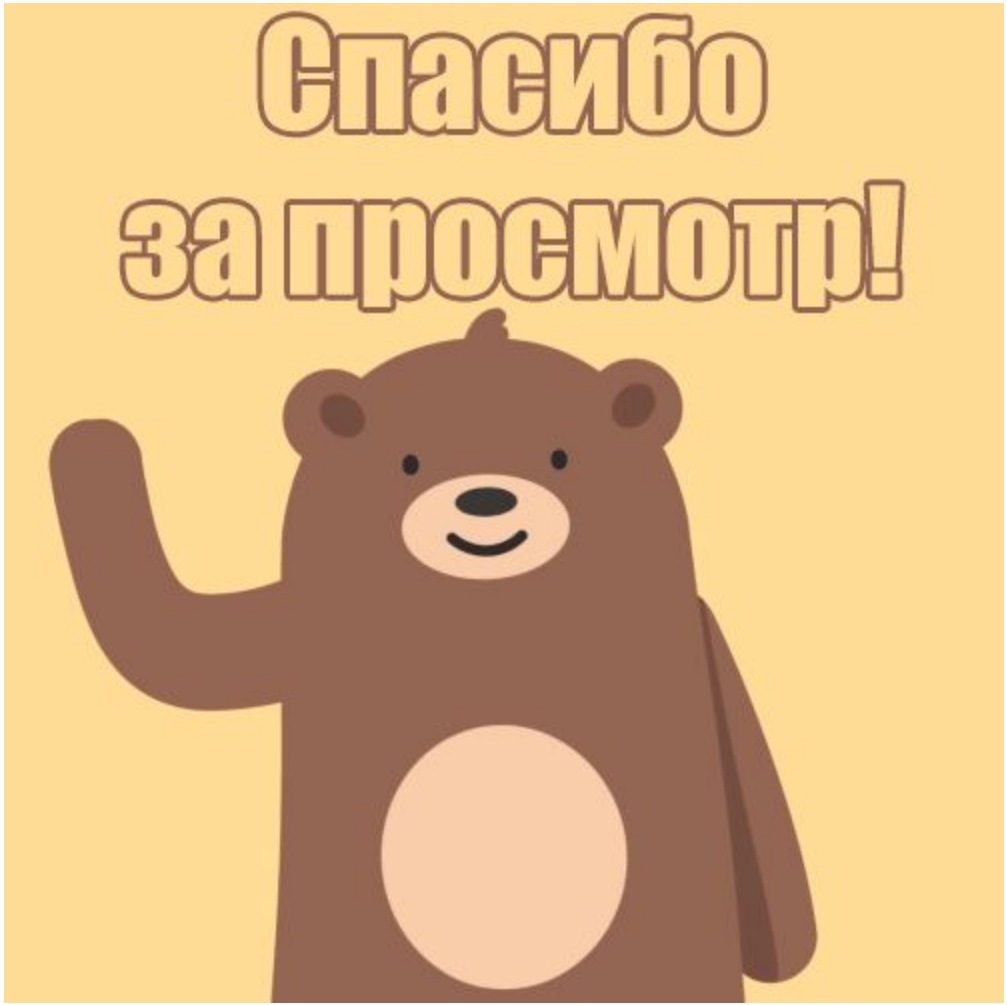
What I think I do

```
from keras import *
```

What I actually do

Neural Networks: demo

- ConvNetJS - https://cs.stanford.edu/people/karpathy/convnetjs/
- A Neural Network Playground - https://playground.tensorflow.org/


Спасибо
за просмотр!