CEITEC

Central European Institute of Technology

BRNO | CZECH REPUBLIC

Deep learning for biological data

Mgr. Vlastimil Martinek
Mgr. David Cechak

Goals of the Workshop

1. Machine learning **basics**
2. **Demystify** deep learning
3. **Pre-Processing** genomic data
4. Programming **neural nets** for genomics

Out of Scope

1. **Heavy math**
2. Easiest way to do/use neural networks
3. **Elaborate** architecture and training schemes
4. Interpretability
5. Able to deploy **inference server**

Agenda

1) Introduction
2) Linear regression
3) Logistic regression classifier
4) Multilayer perceptron (MLP)
5) Convolutional neural network (CNN)
6) Extras

Workshop Tools

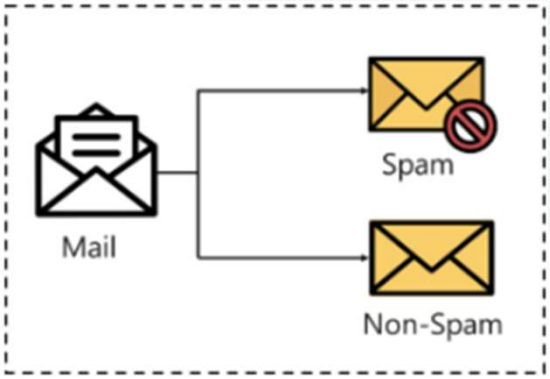
Solving a problem

f(X) = y

X = biological sequence, microscopy image, molecule, text, map as a graph…
y = contains a gene?, tumor position and volume, binding site, shortest path…

f traditionally designed by hand

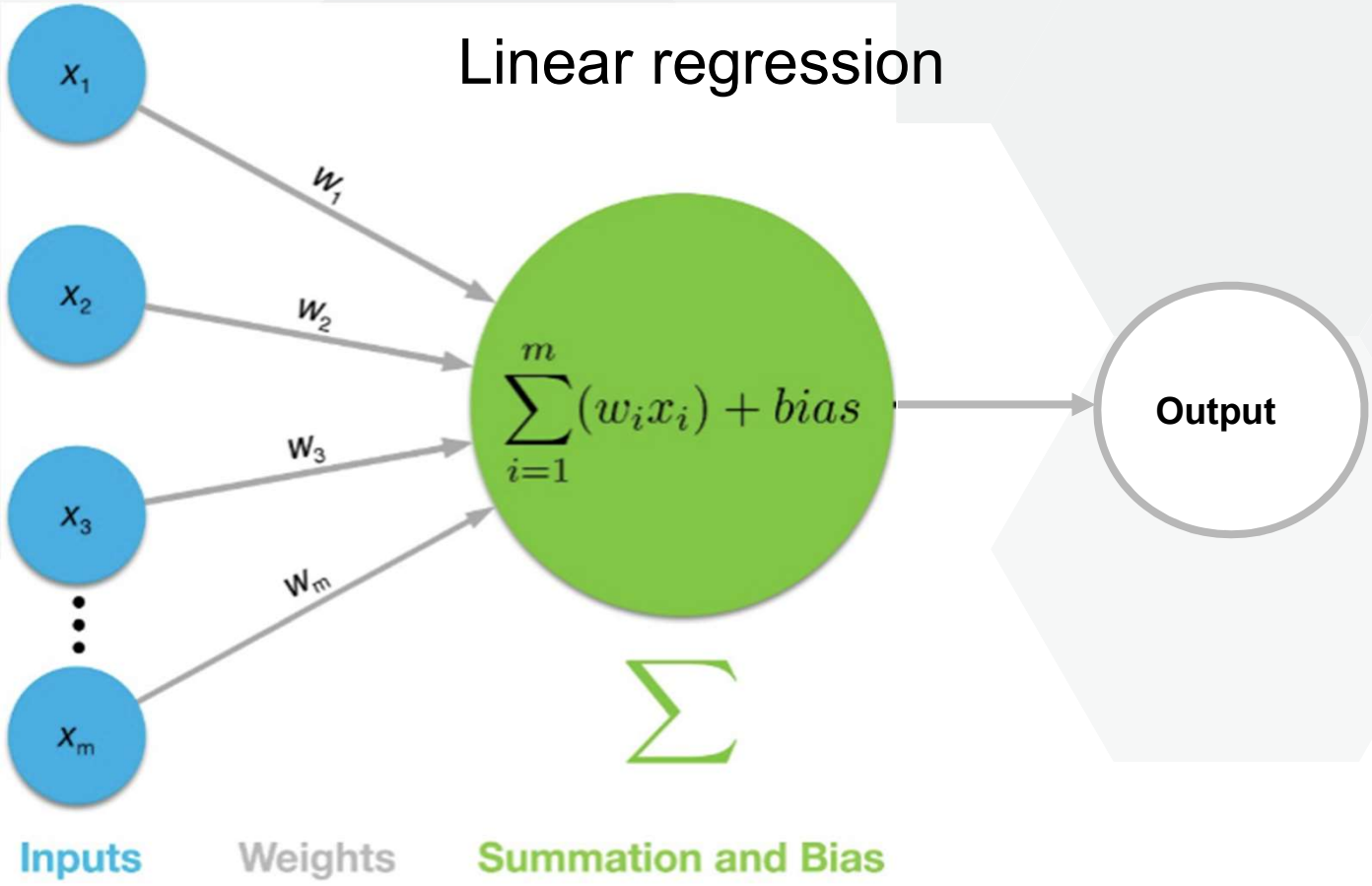
What if we let a machine design the funcion **f**?



Goal: have a useful predictor

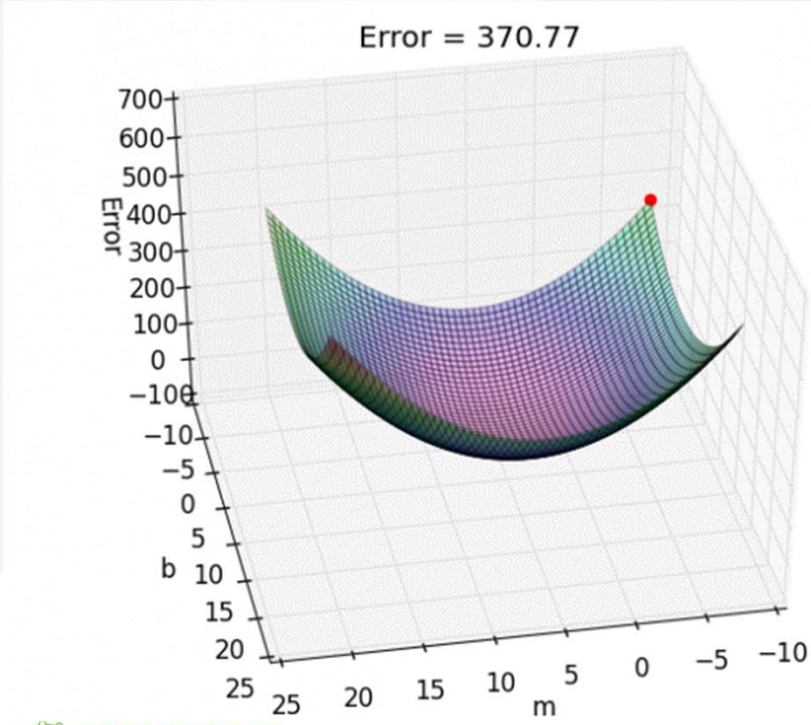

Linear
regression

Linear regression

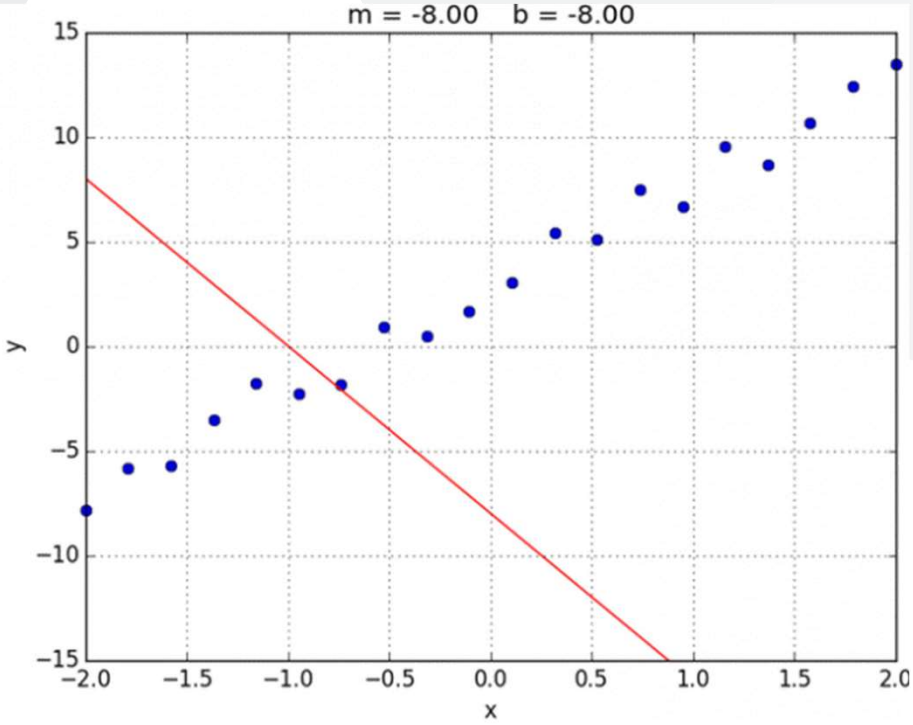


Optimization process - fitting our data

Error = MSE



Y = mX+b



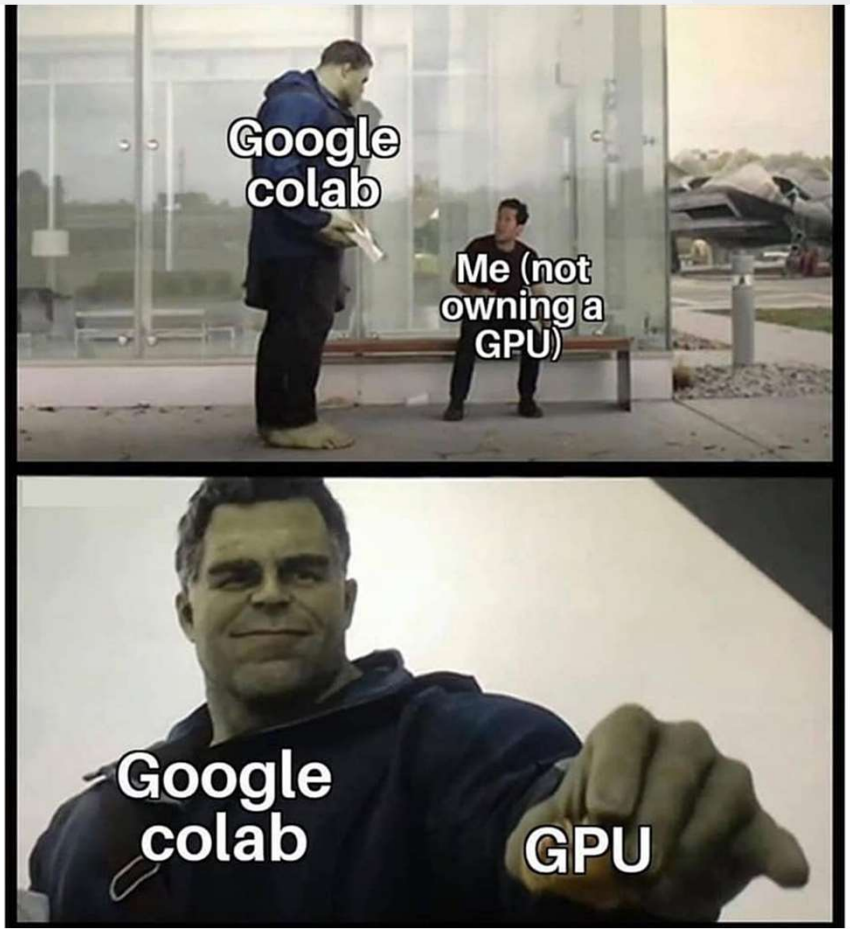
CEITEC

Practical part

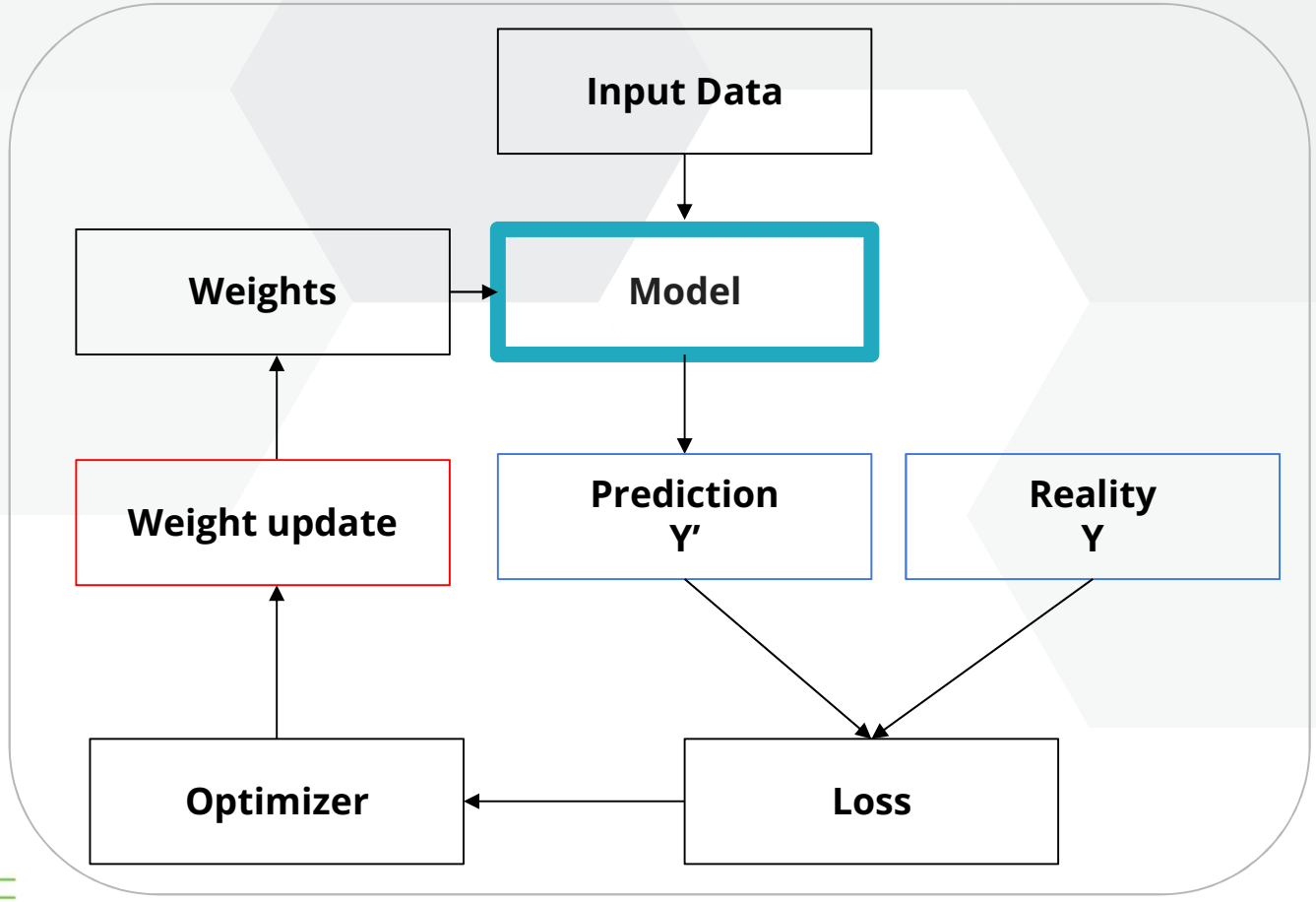
https://github.com/BioGeMT/MALTAomics-Summer-School/

Day 2 Workshop

MALTA_01 Notebook



Task: Learn Optimal Weights to Minimize Loss



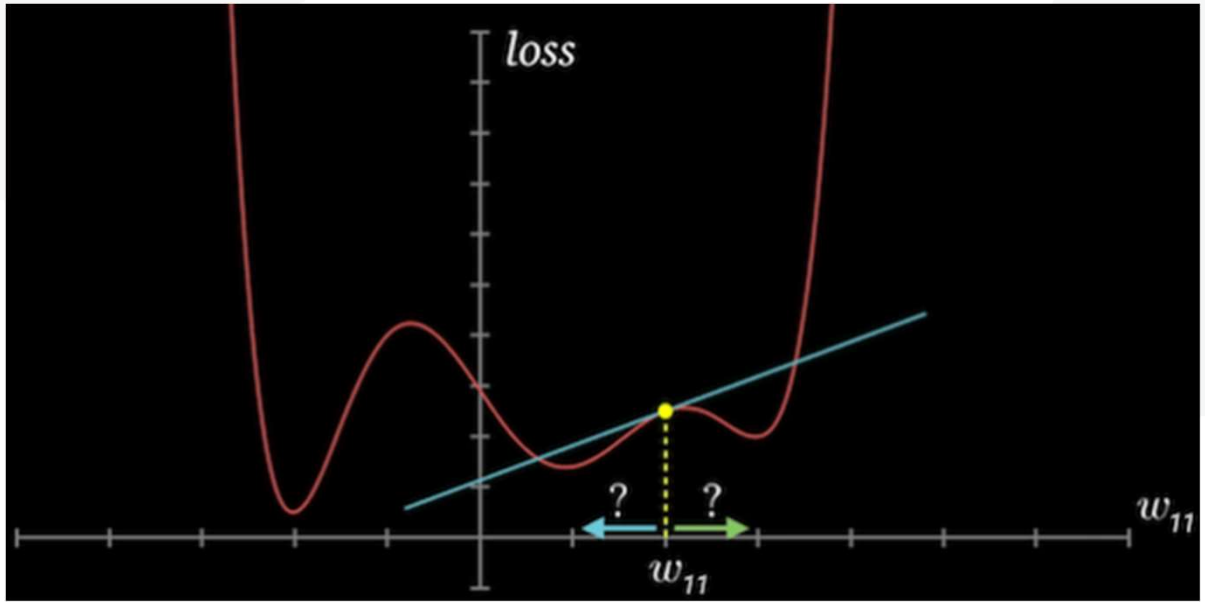
Gradient descent

Compute a gradient for a **single** weight w.r.t the loss

Average over the training data

Indicates the rate of change

```
weight = weights - weight.grad * 1e-5
bias   = bias    - bias.grad   * 1e-5
```



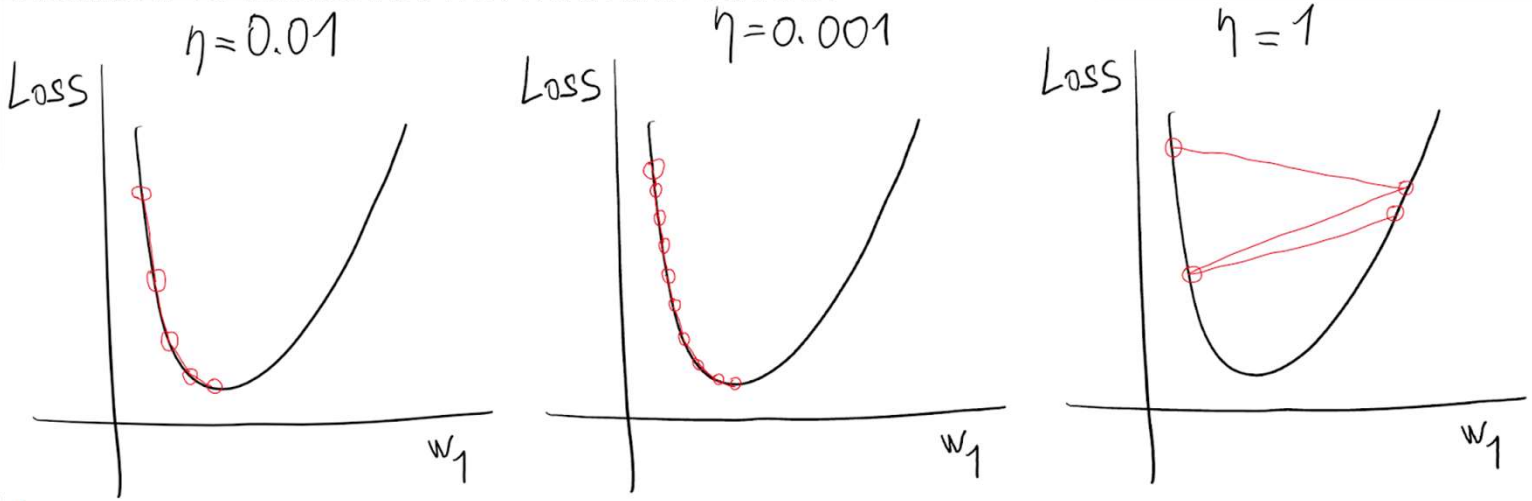
Epochs, learning rate, loss function

Learning rate - magnitude of the weight update

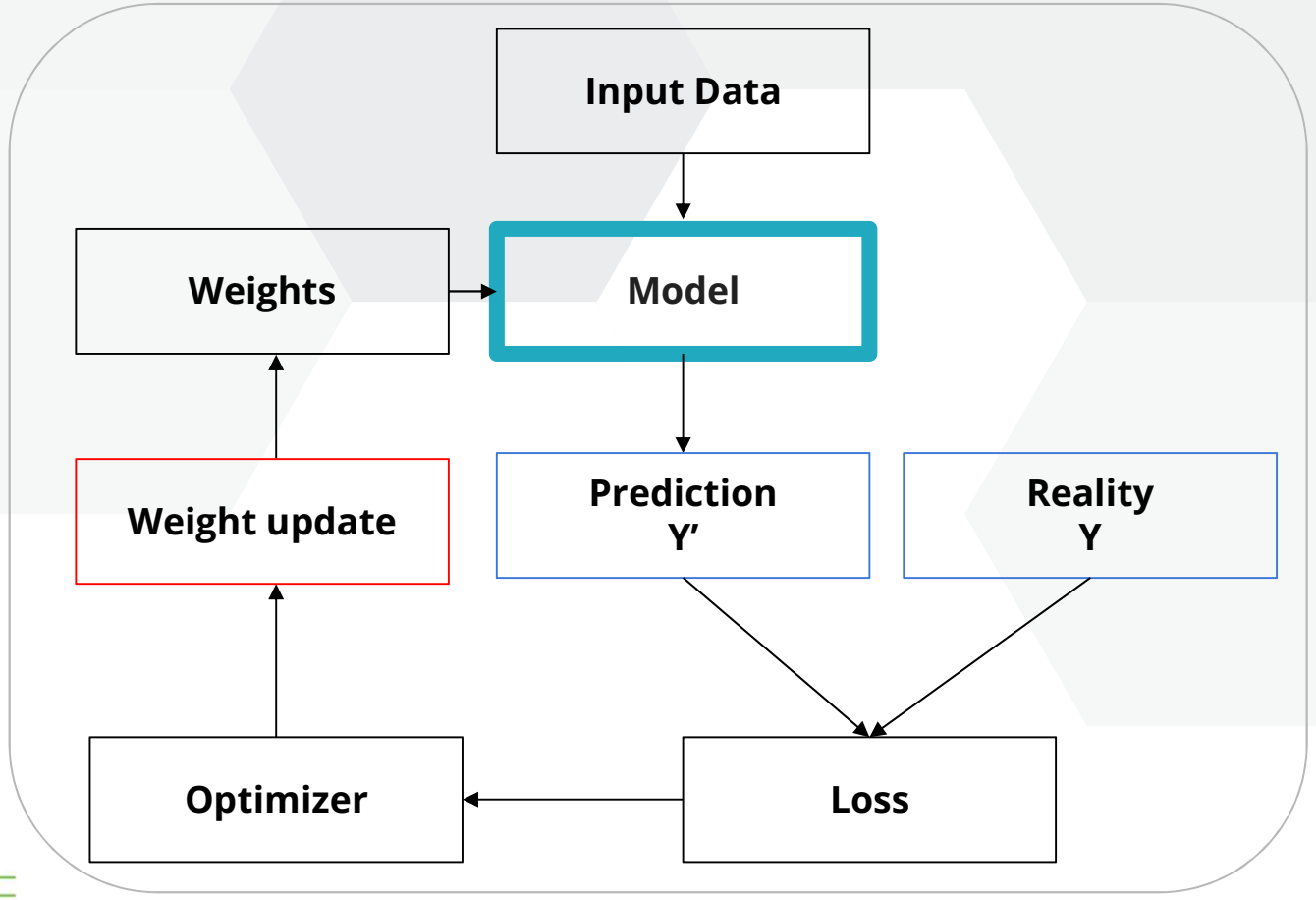
Loss function - objective to minimize

Tensor - pytorch array able to track gradient

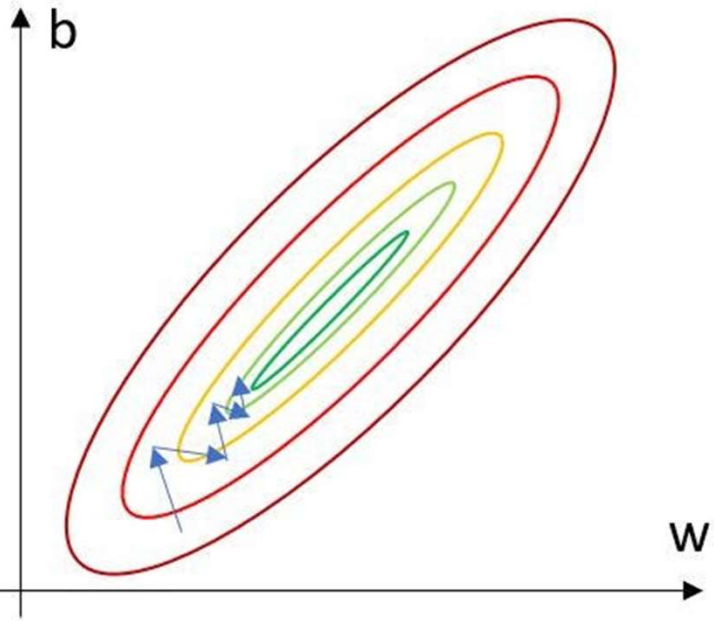
Epochs - number of cycles in the learning phase



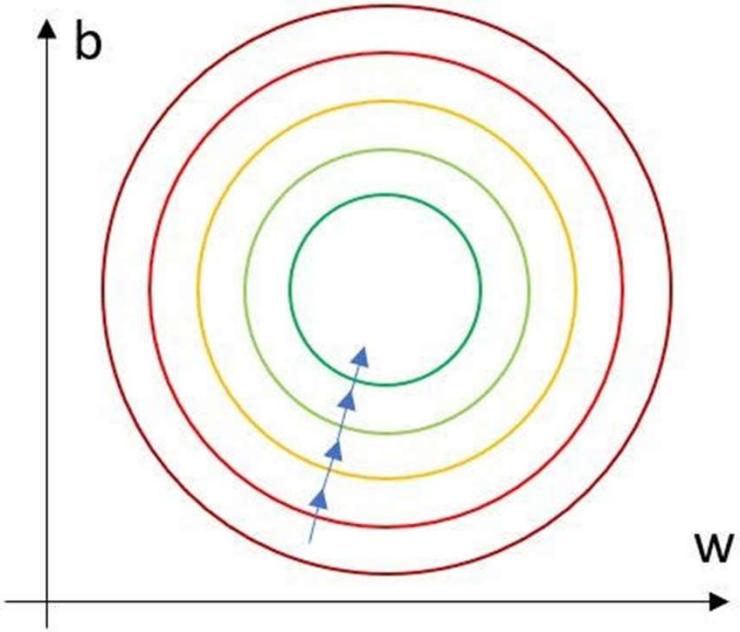
Task: Learn Optimal Weights to Minimize Loss



Unnormalized:



Normalized:



Short break.
Back at 11:12

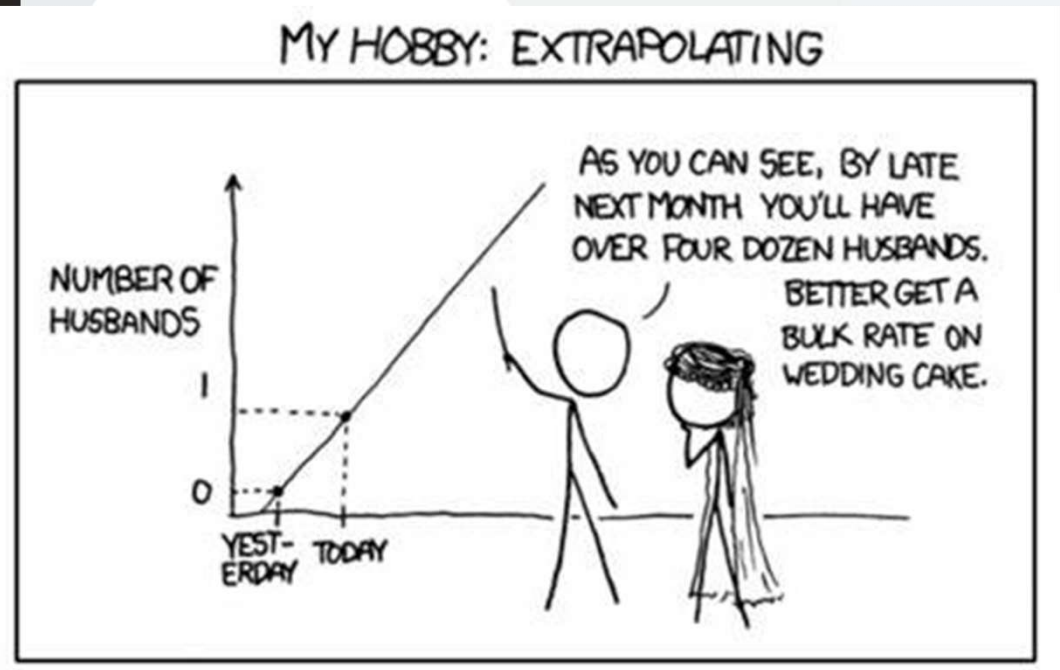
DS: How is model training going?
Intern: No so good. It's not learning anything...
DS: Show me your train loop
Intern: ...

```python
import numpy as np
import pandas as pd
import torch

def train():
    print('      o O___ _________  ')
    print('   _][__|o| |0 0 0 0|  ')
    print('  <______|-|______|  ')
    print('  /O - O O   O    O    ')


train()
```

imgflip.com



Quiz

1. What shape should be the training data for our Linear layer?
2. What is a loss function?
3. What is mean squared error?
4. What happens when you invoke the .backward function on the result of the mean squared error loss function?
5. How do you update the weights and biases of a model?

model trained for 1000 epochs

model trained for 100 epochs

model trained for 1 epoch

Quiz

6. Why do you reset gradients to zero after updating weights?
7. What is an epoch?
8. What is the benefit of training a model for multiple epochs?
9. What should you do if your model's loss doesn't decrease while training?
10. What are the inputs to a PyTorch optimizer?
11. How could we feed 2 features as input?

model trained for 1000 epochs

model trained for 100 epochs

model trained for 1 epoch

Linear regression summary

$Y = \mathbf{w}X + \mathbf{b}$

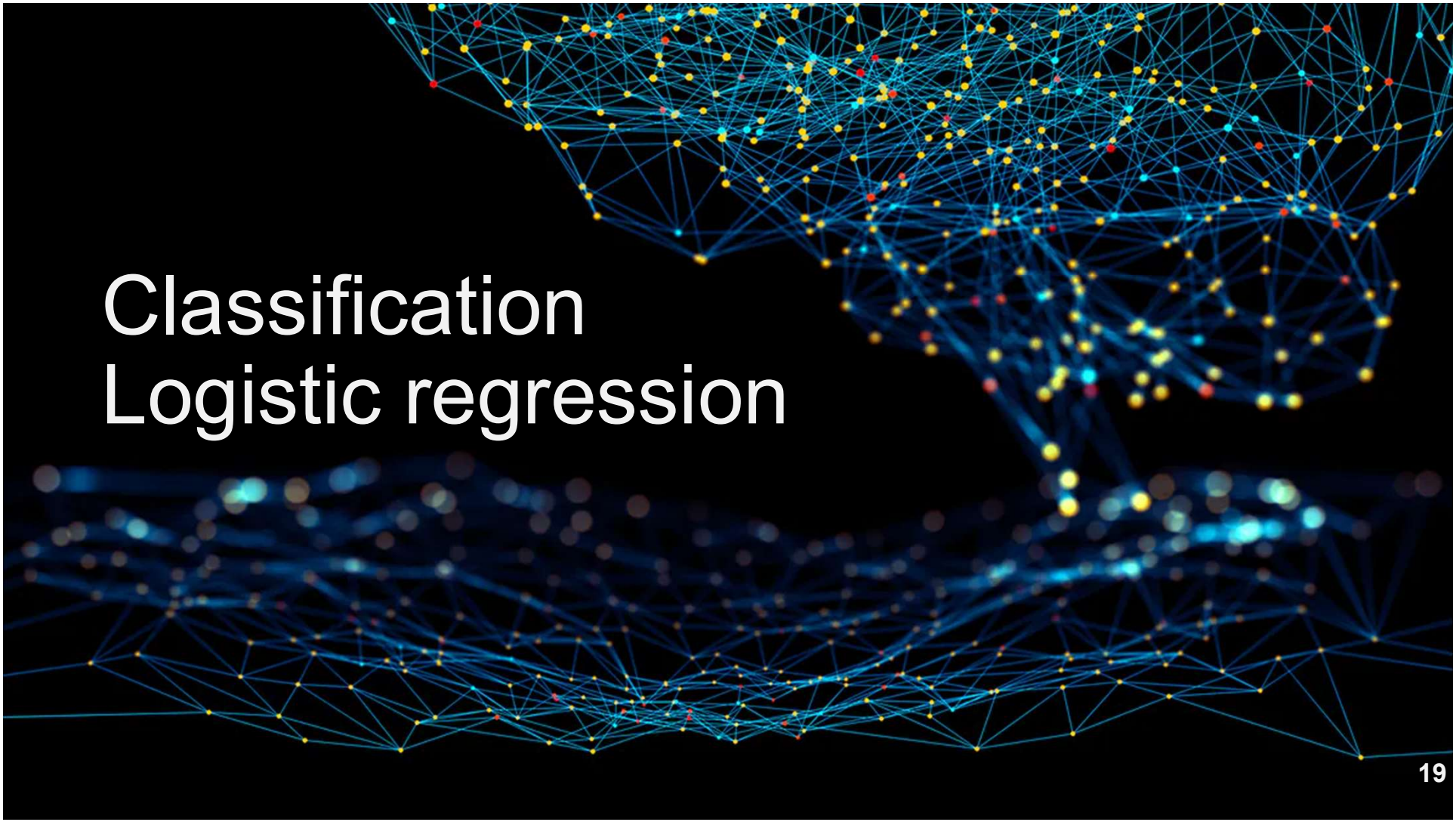
Regression problem

Common with deep learning

- Needs input data
- Starts with random weights/parameters
- Computes loss
- Repeatedly optimizes loss to improve (a tiny bit) on training data (epochs)
- Resulting model has updated weights

Differences

- Linear only

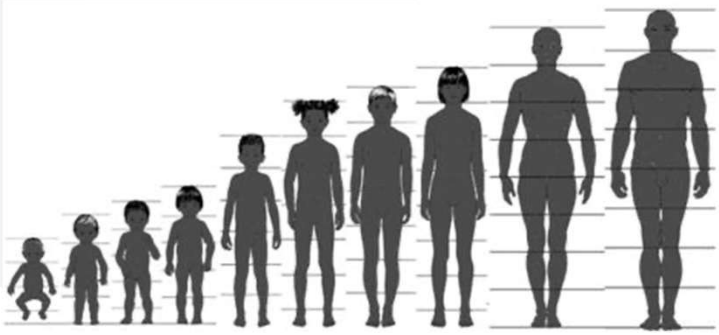

Classification
Logistic regression

Is this DNA sequence a non-TATA promoter?

f(X) = y

X = height

y = weight?



f(X) = y

X = DNA sequence

y = contains a non-T^TA promoter?

Promoter

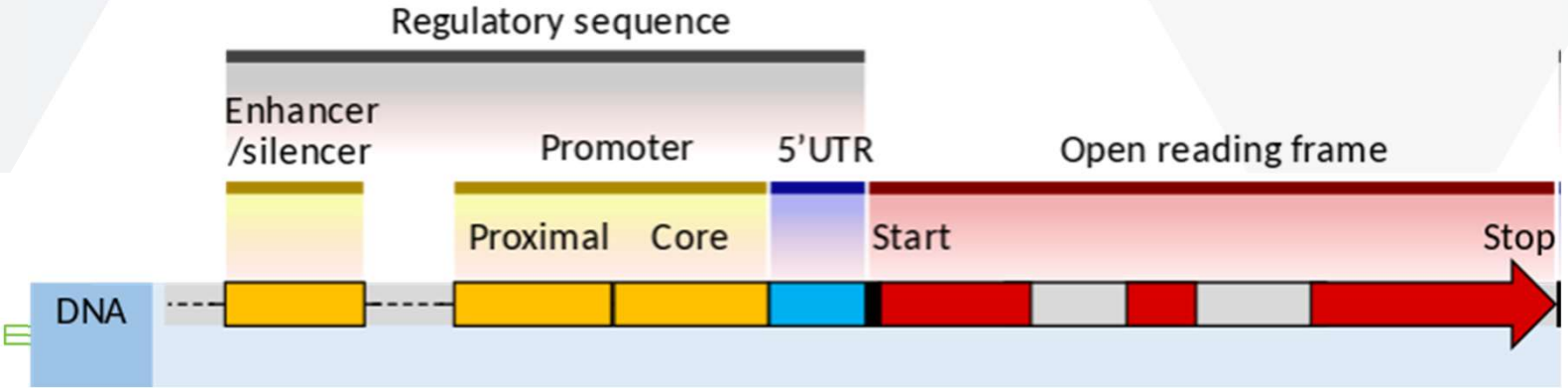
- Regulatory sequences
- Where proteins bind to initiate transcription of a gene

TATA

- drive **burst-like gene expression** (dosage)

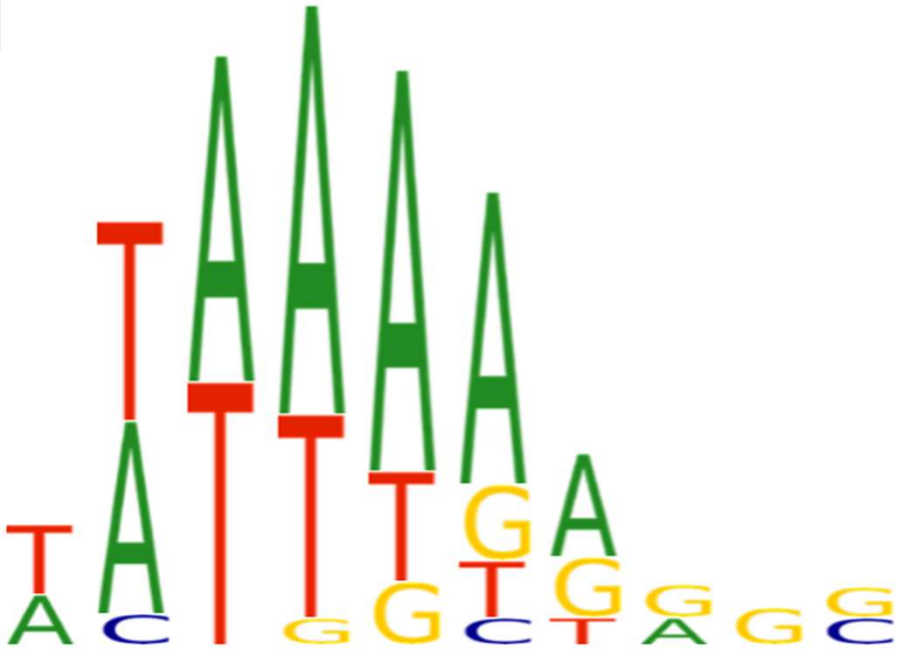
non-TATA

- govern housekeeping genes - **consistent expression** (continuous)

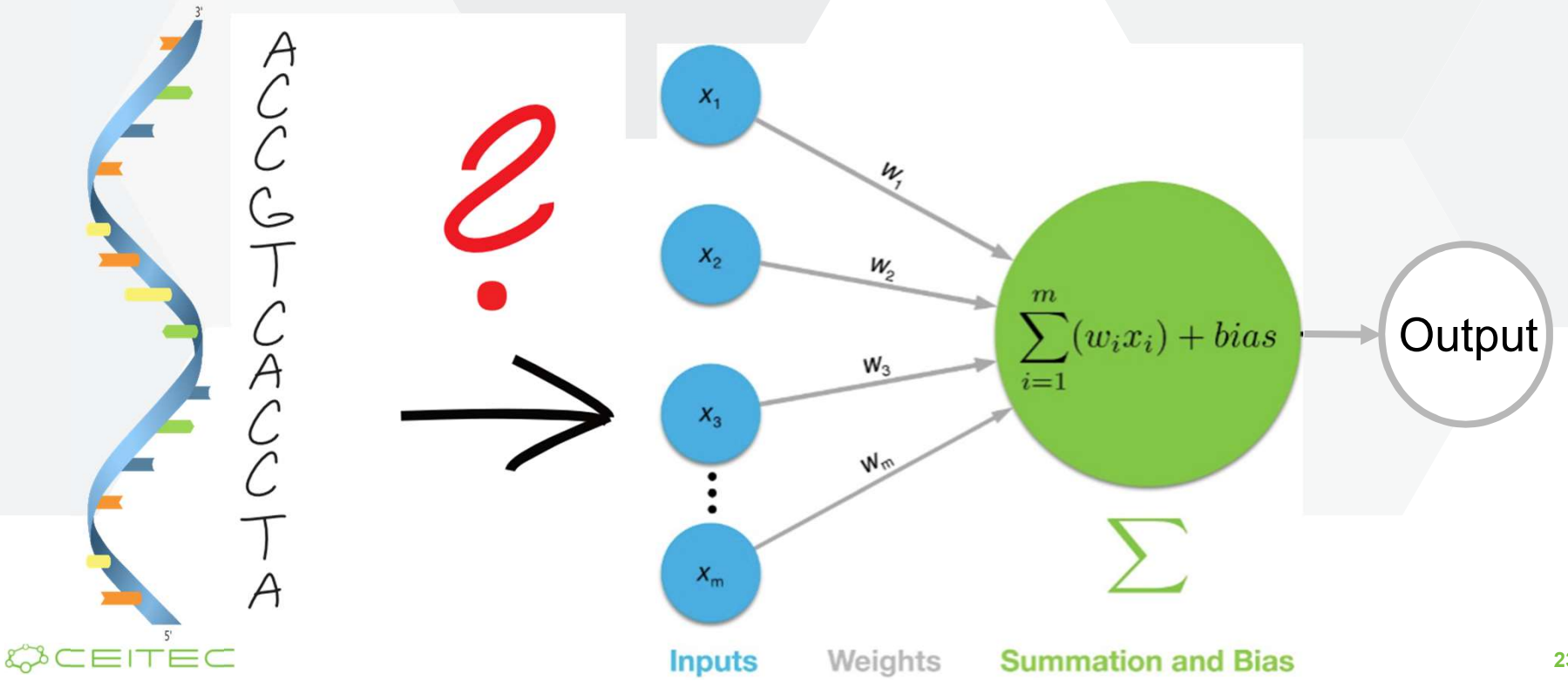


TATA-box region - motif

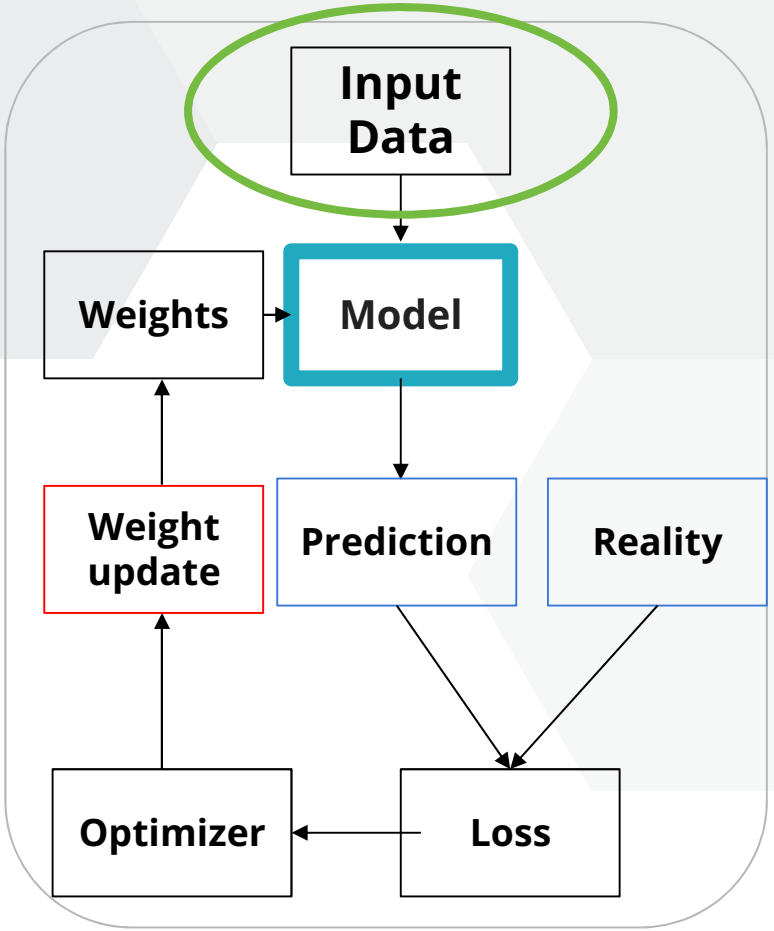
Sub sequence pattern (motif) for human TATA promoter sequences



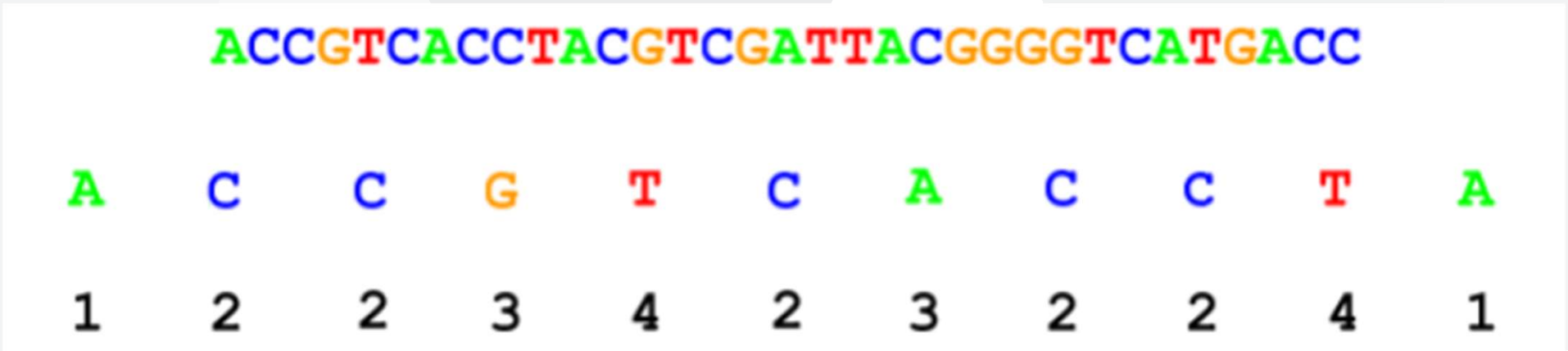
First problem: Numericalization -> Tokenization



Numericalization & Tokenization



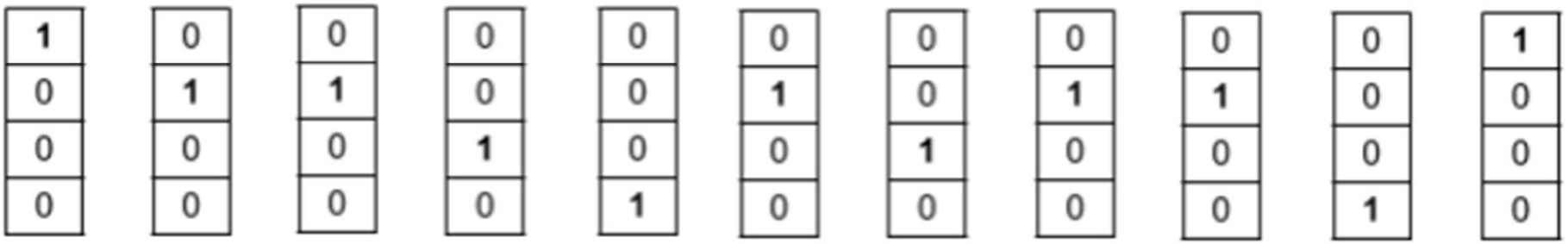
Numericalization & Tokenization

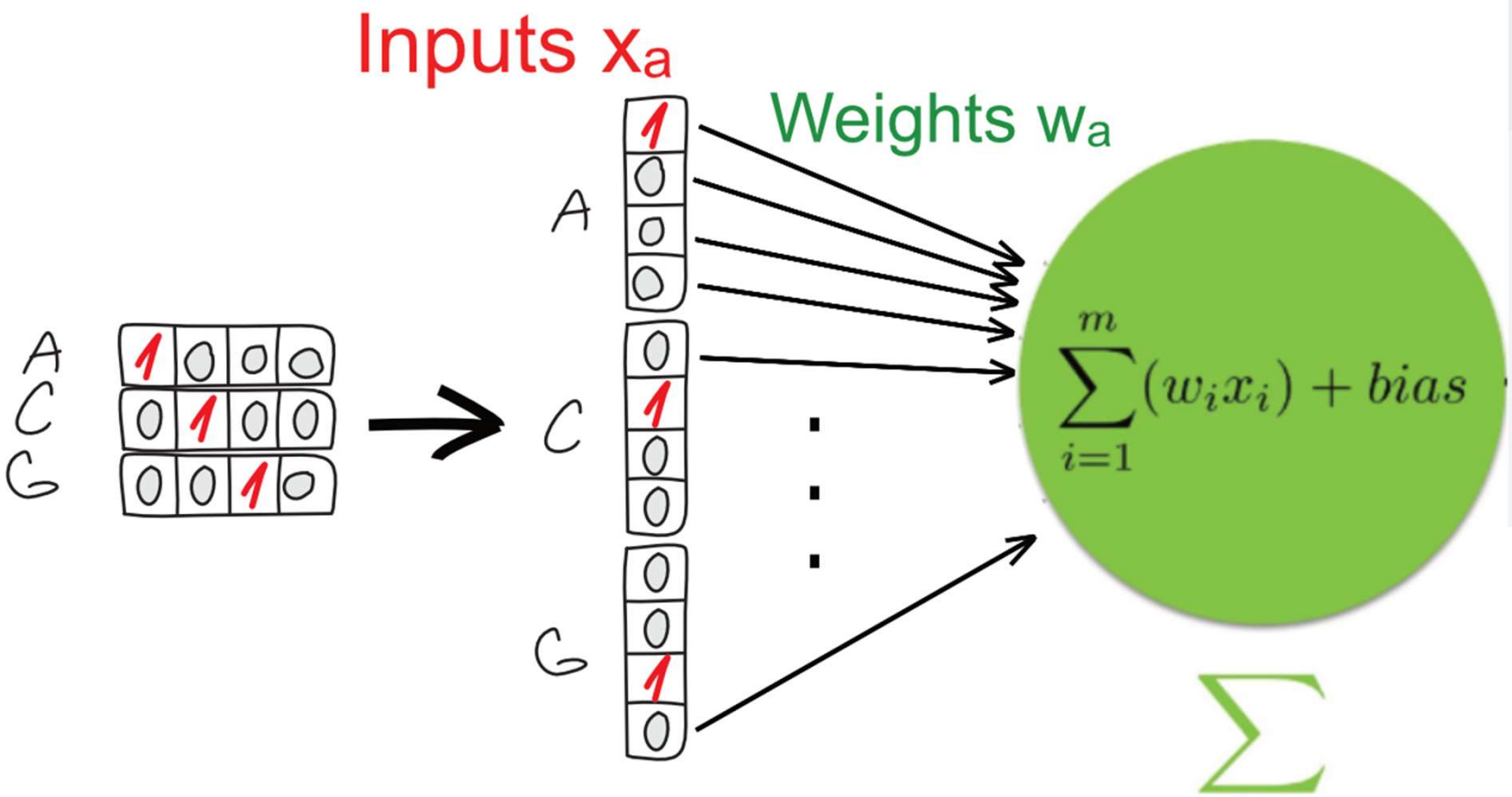


Numericalization & Tokenization

ACCGTCACCTACGTCGATTACGGGGTCATGACC

A	C	C	G	T	C	A	C	C	T	A
1	2	2	3	4	2	3	2	2	4	1



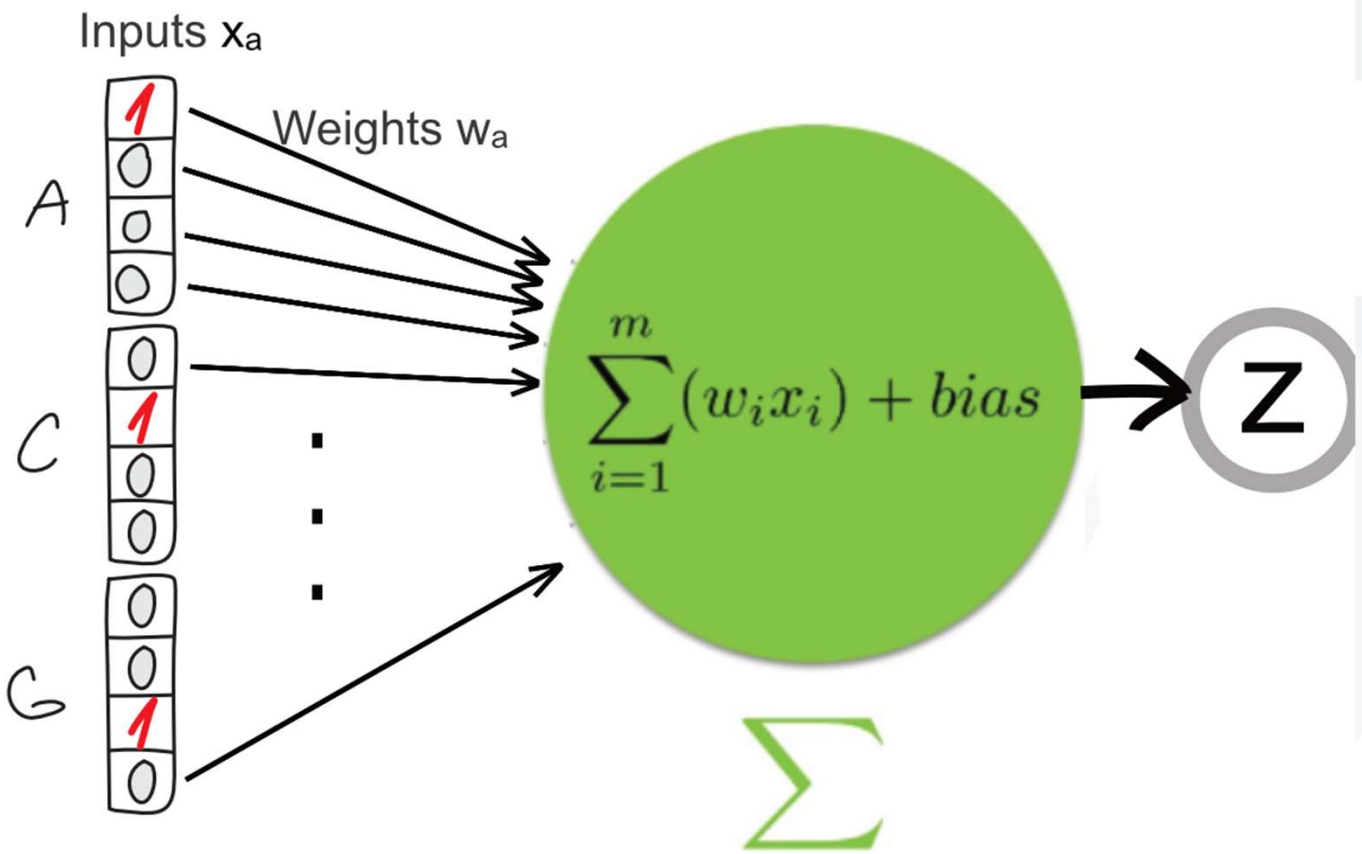

Inputs x_a
Weights w_a
A
C
G
A
C
G
$\sum_{i=1}^{m}(w_i x_i) + bias$
$\sum$

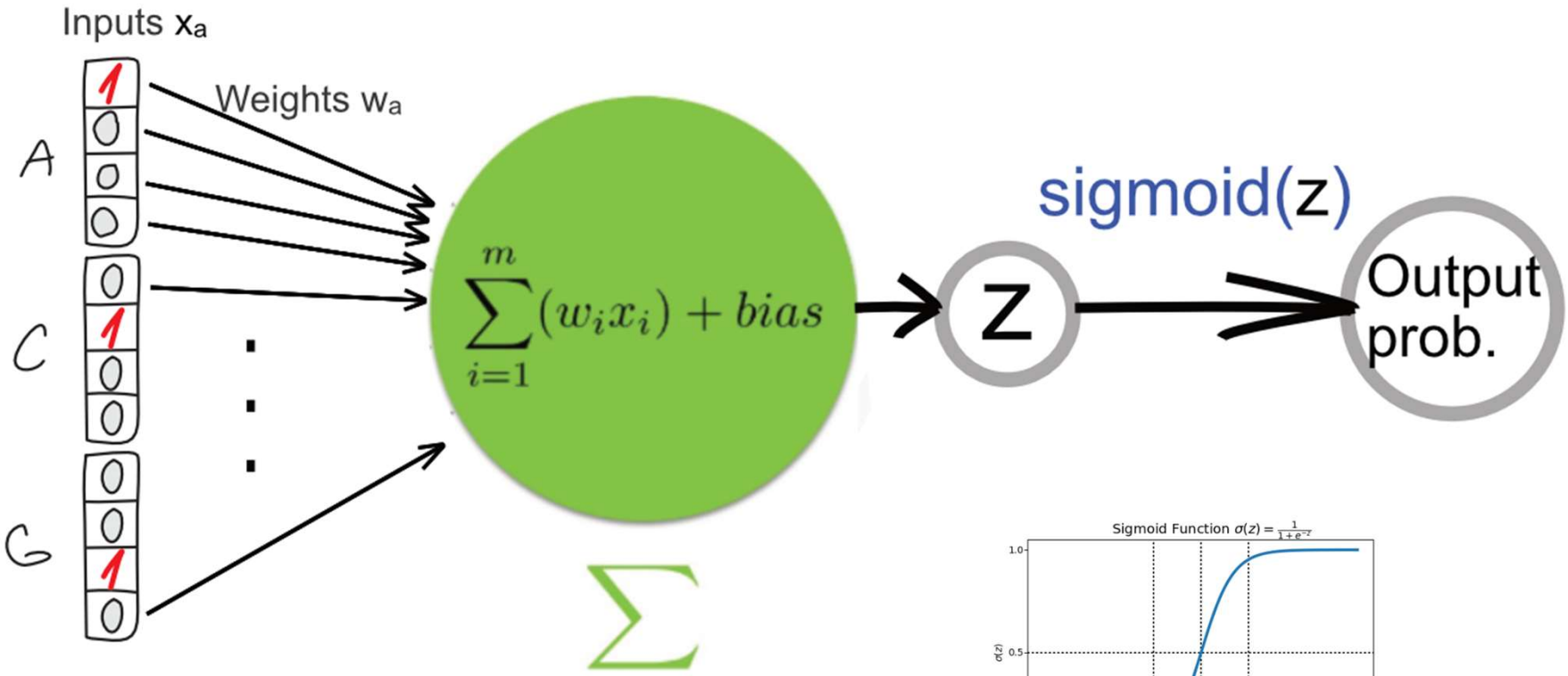
Logistic regression - DNA promoters classification

https://github.com/BioGeMT/MALTAomics-Summer-School/

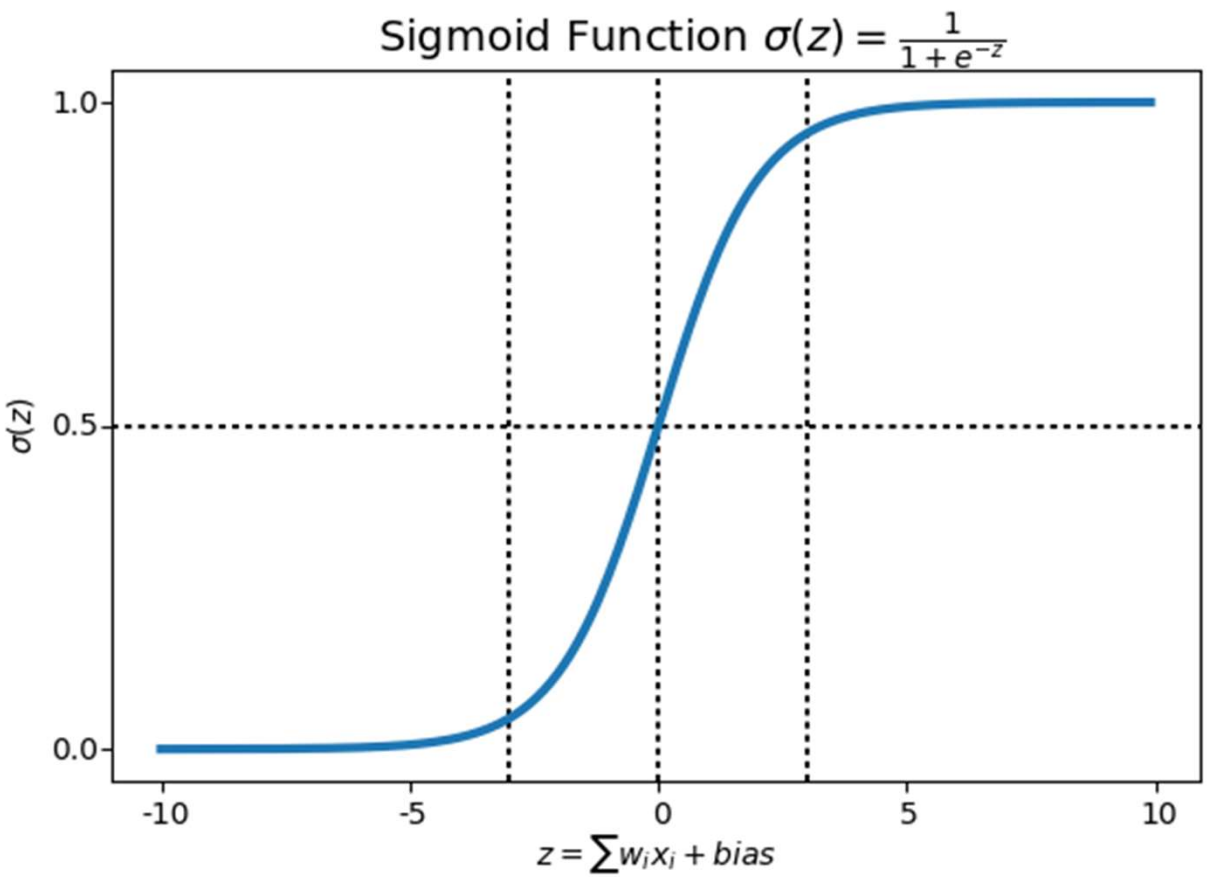
Day 2 Workshop

MALTA_02_DNA_enhancers.ipynb

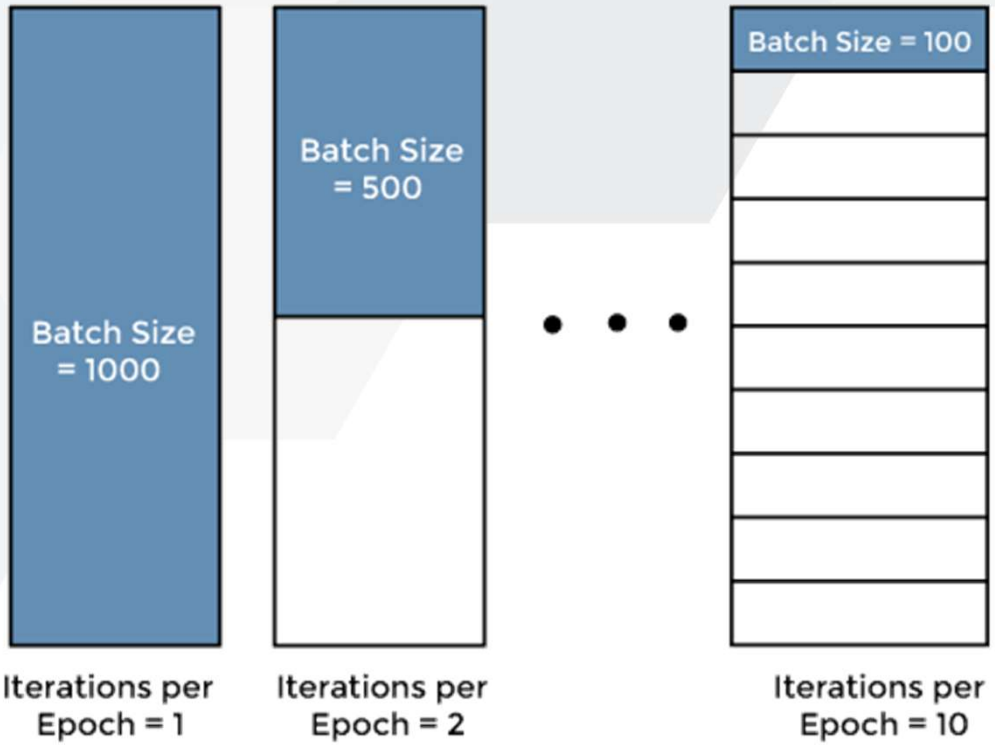

Inputs x_a
Weights w_a
A
C
G
$\sum_{i=1}^{m}(w_i x_i) + bias$
Z
$\sum$

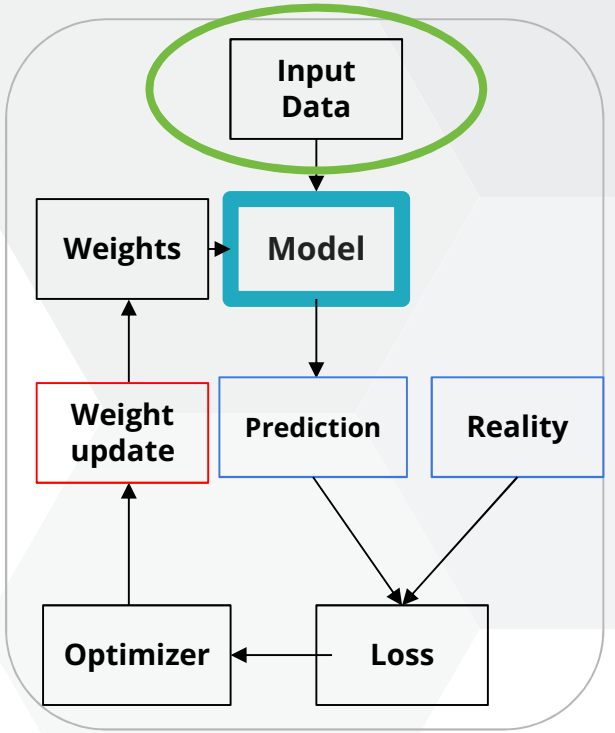

Inputs x_a
Weights w_a
A
C
G
$\sum_{i=1}^{m}(w_i x_i) + bias$
sigmoid(z)
z
Output prob.
$\sum$

Sigmoid Function $\sigma(z) = \frac{1}{1+e^{-z}}$
1.0
0.5
0.0
$\sigma(z)$
-10
-5
0
5
10
$z = \sum w_i x_i + bias$


Sigmoid Function $\sigma(z) = \frac{1}{1+e^{-z}}$
1.0
0.5
0.0
σ(z)
-10
-5
0
5
10
$z = \sum w_i x_i + bias$

Batch size and shuffling

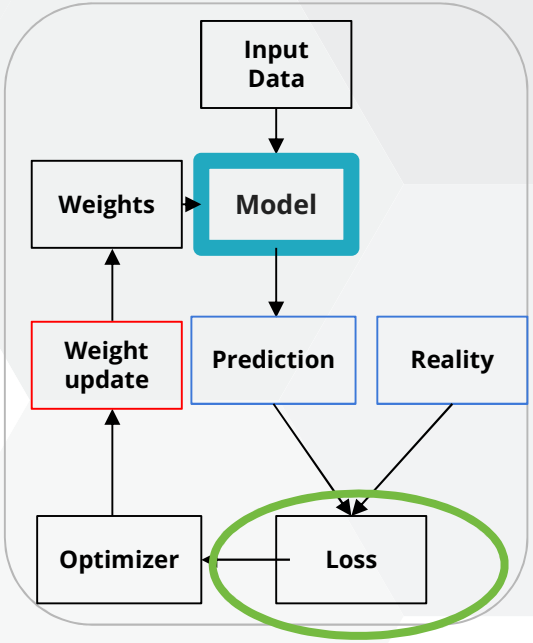




Batch size - speed VS stability of the learning process

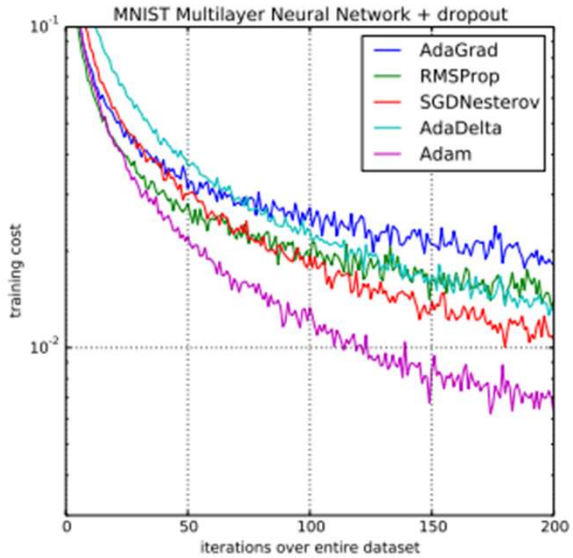
Loss functions and activations

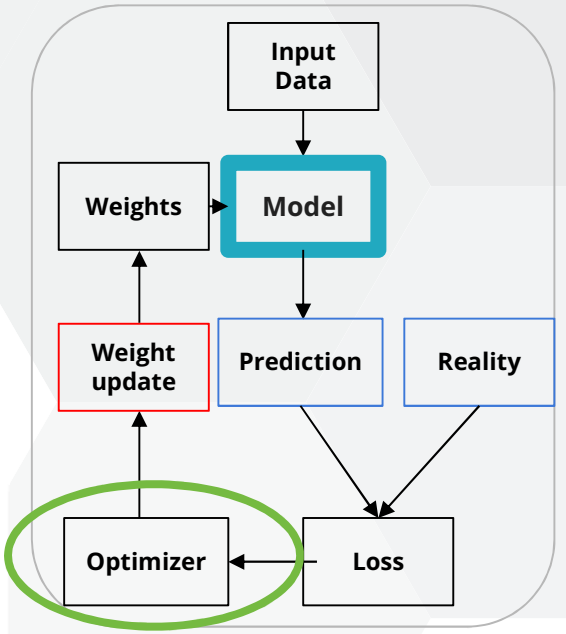
	Regression	**Binary Classification**
Loss function	MSE	Binary CE
Activation	None	Sigmoid



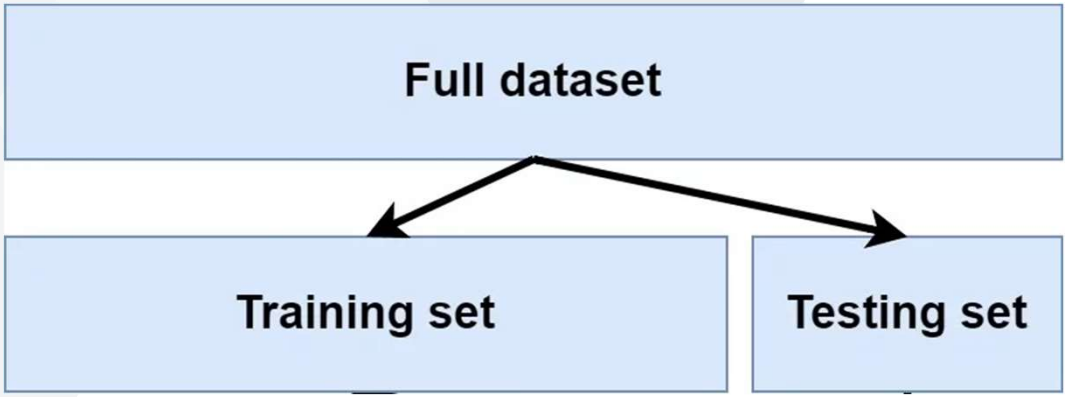
Adam optimizer

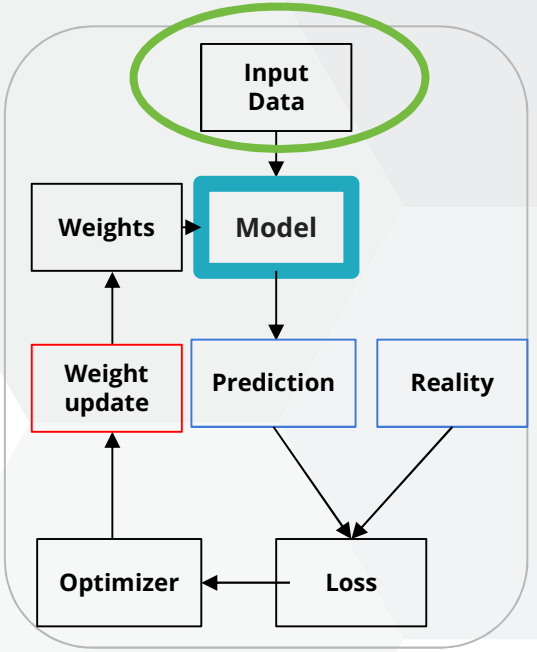
- Converges faster than SGD
- Uses momentum and moving average of the gradient
- Computer per-weight learning rates



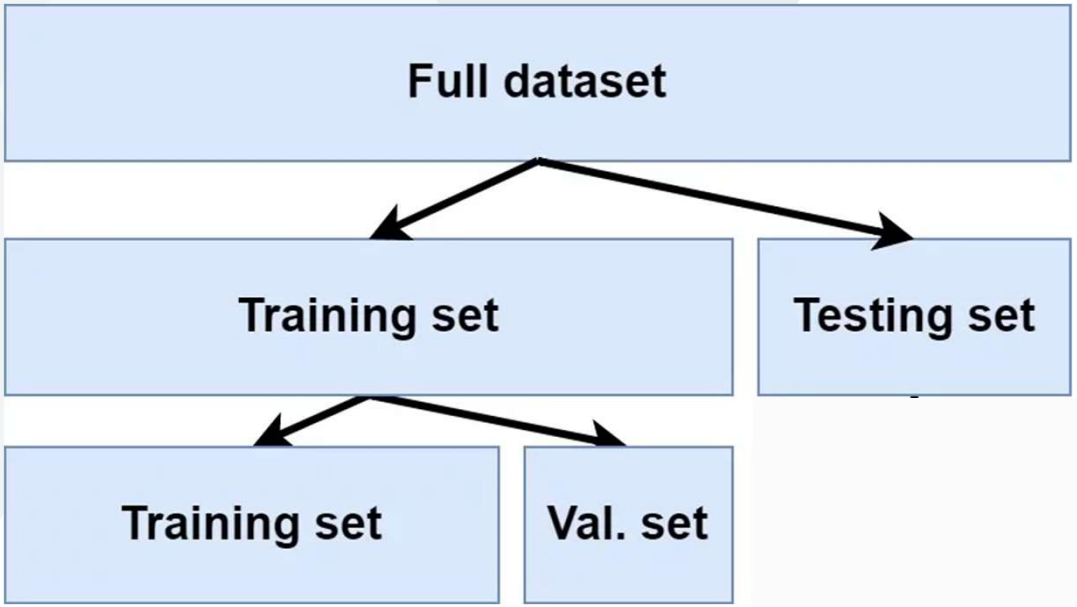


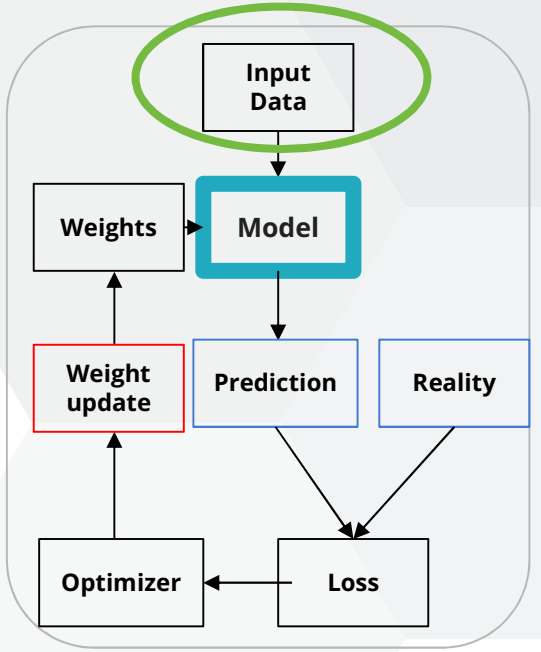
Dataset management



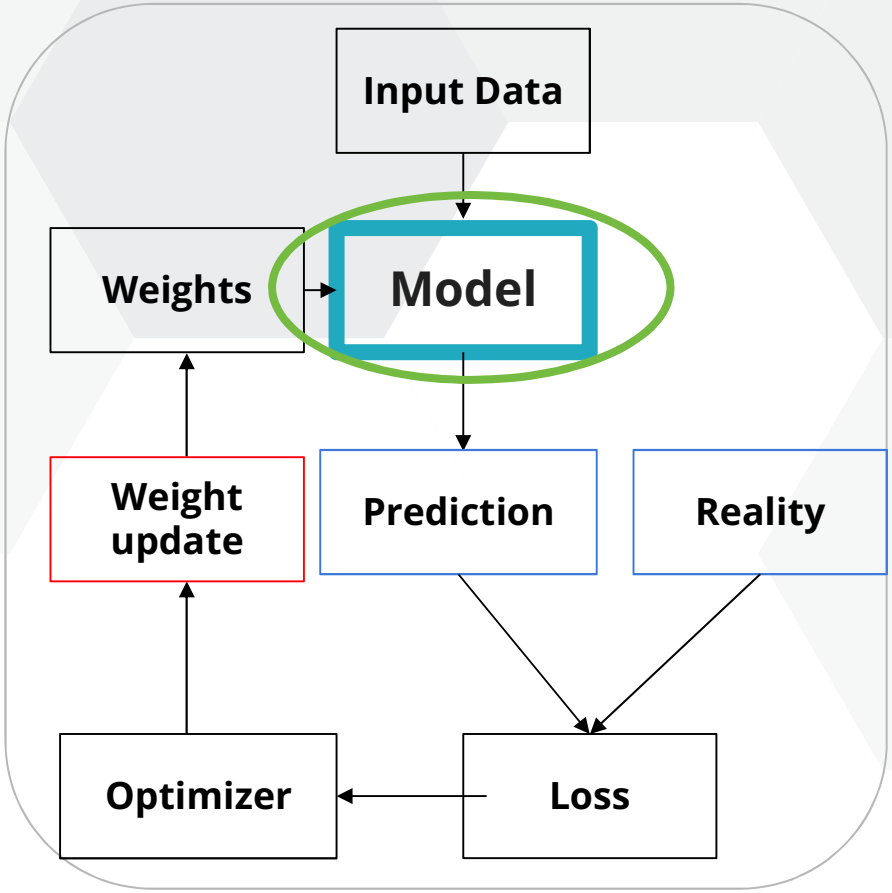


Dataset management






Multilayer Perceptron


Input Data
Weights
Model
Weight update
Prediction
Reality
Optimizer
Loss

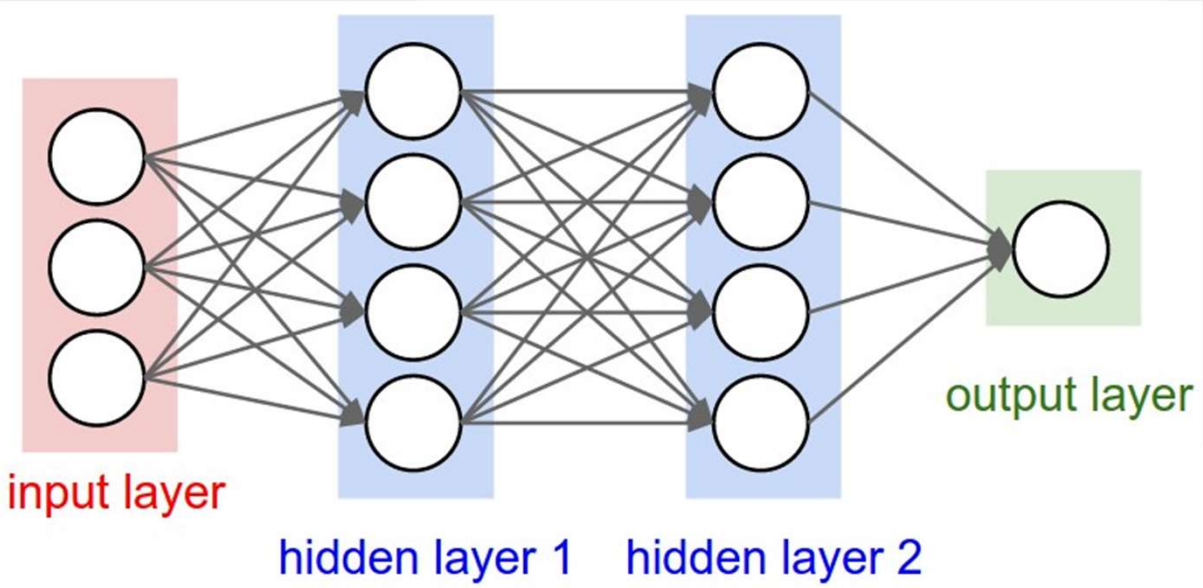
Nonlinearity and hidden layers

We could stack Linear Regression

No more powerful

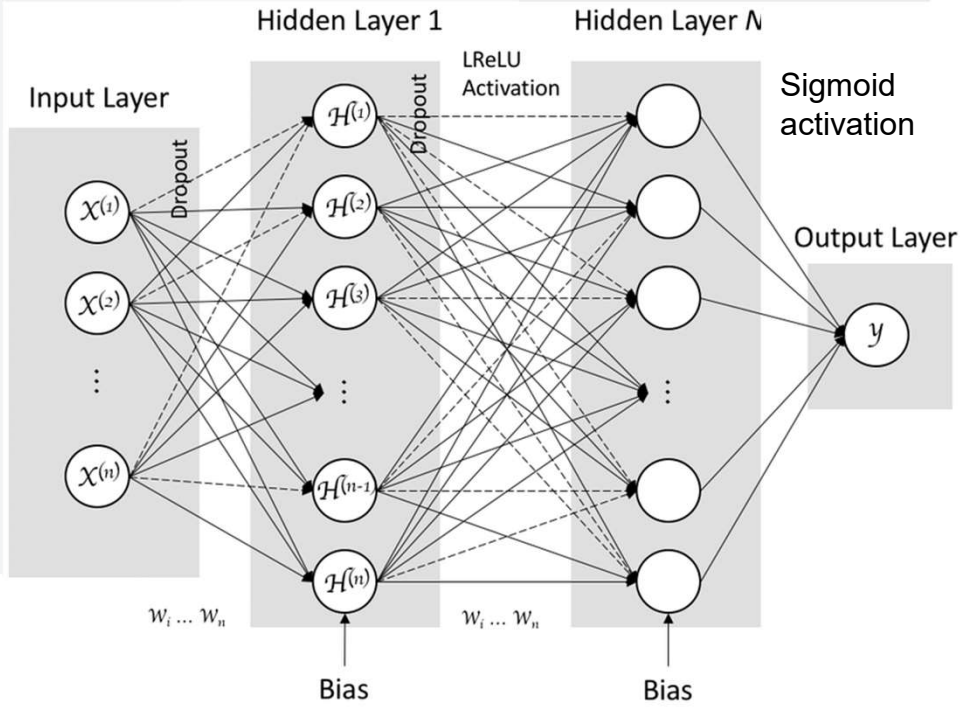
Only linear combinations

We need nonlinearity



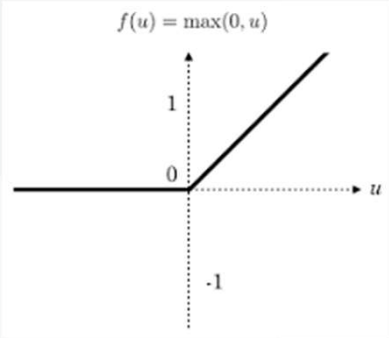
Dense layer = every neuron is connected to all neuron from the previous layer

Multilayer Perceptron (MLP)



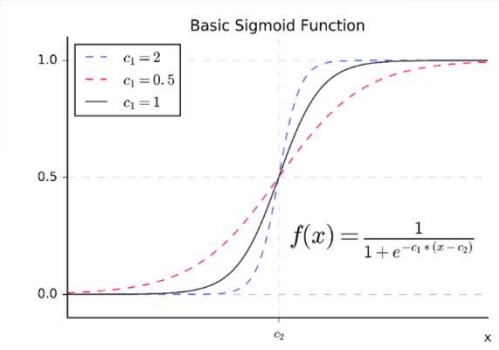
Hidden Layer

Activation:
ReLU

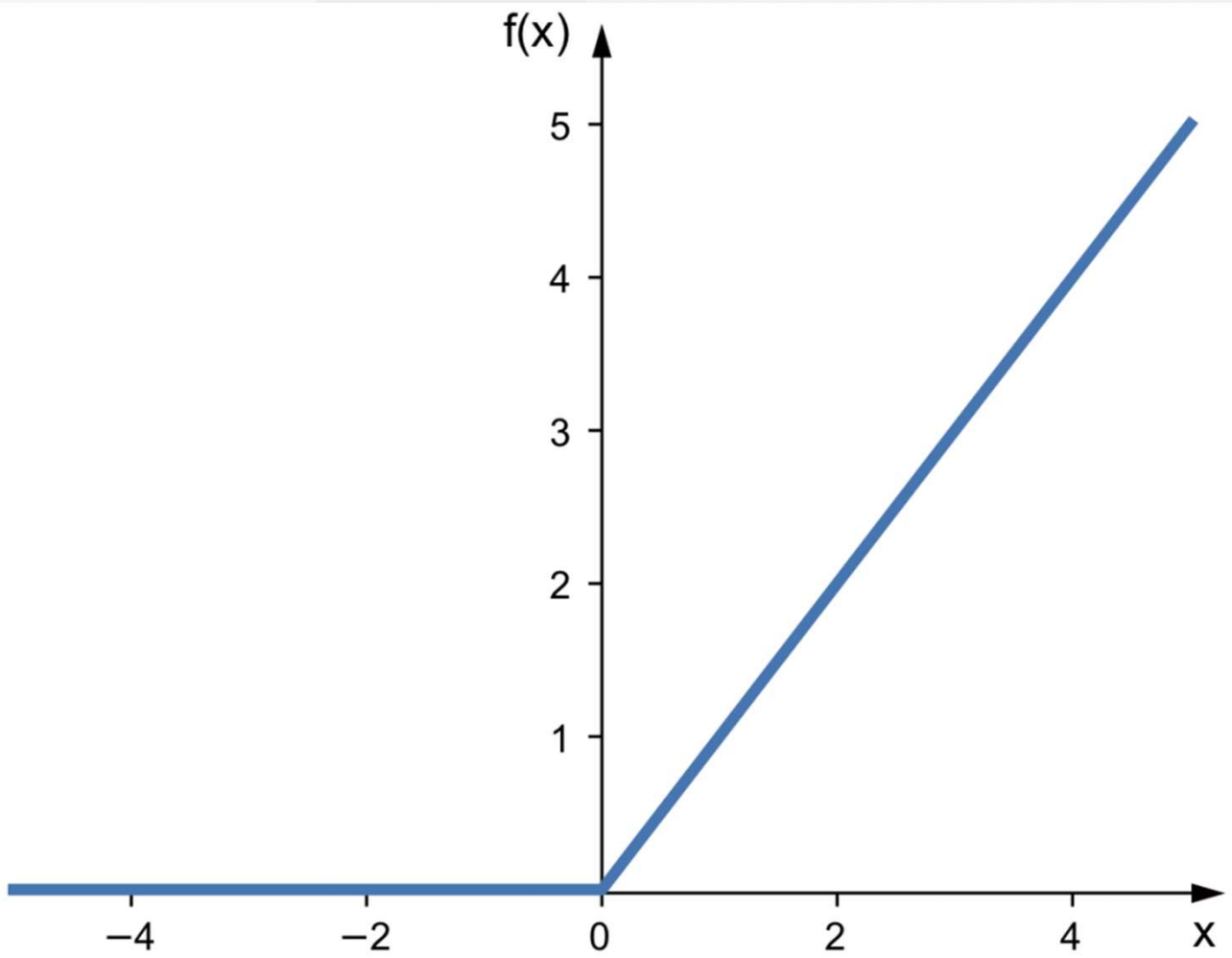


Output Layer

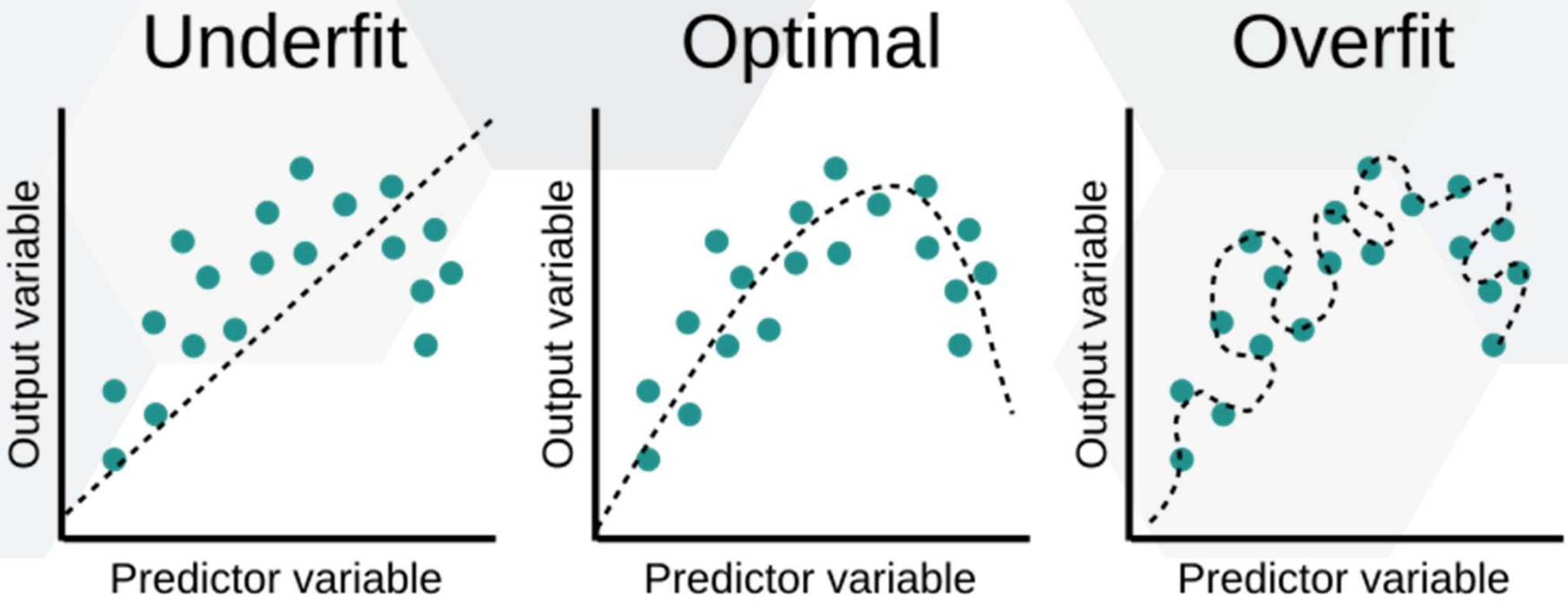
Activation:
Sigmoid



CEITEC

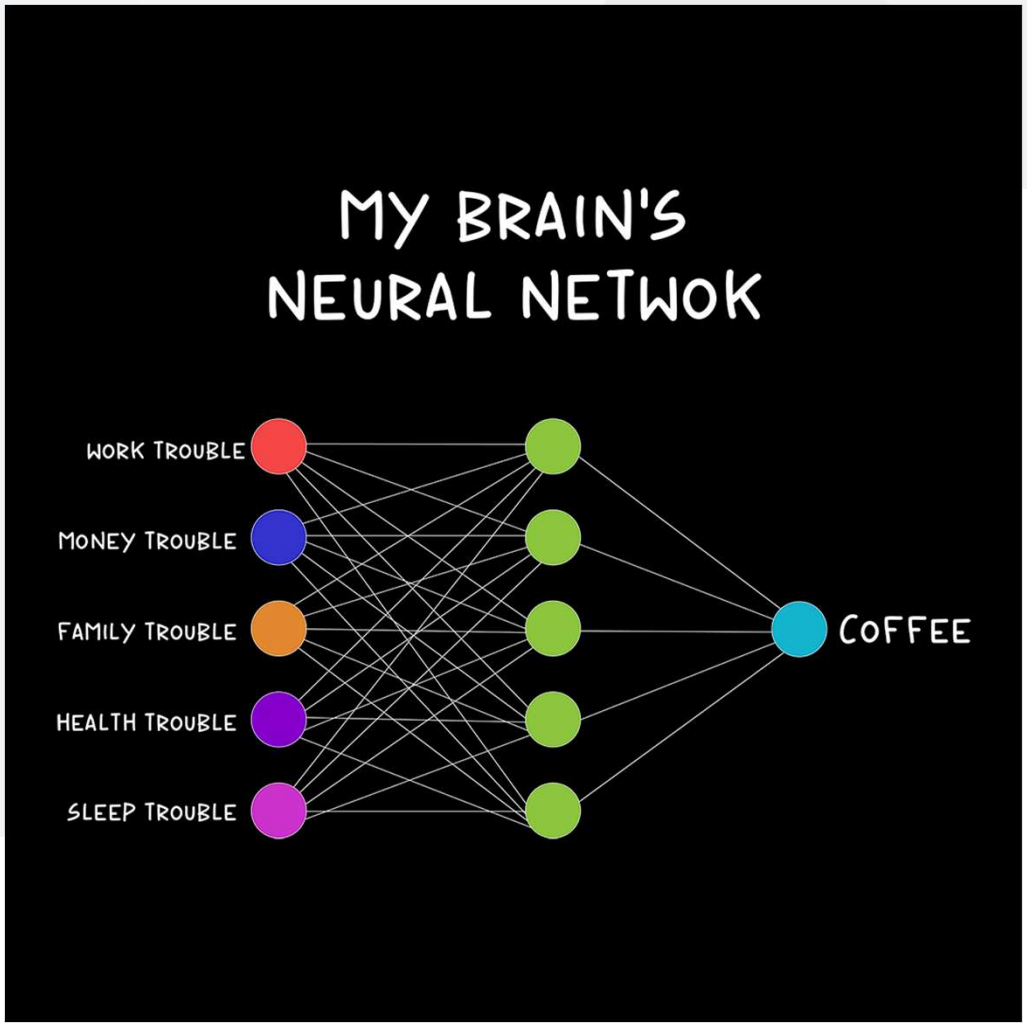

f(x)
5
4
3
2
1
−4
−2
0
2
4
x

Overfitting



Quiz

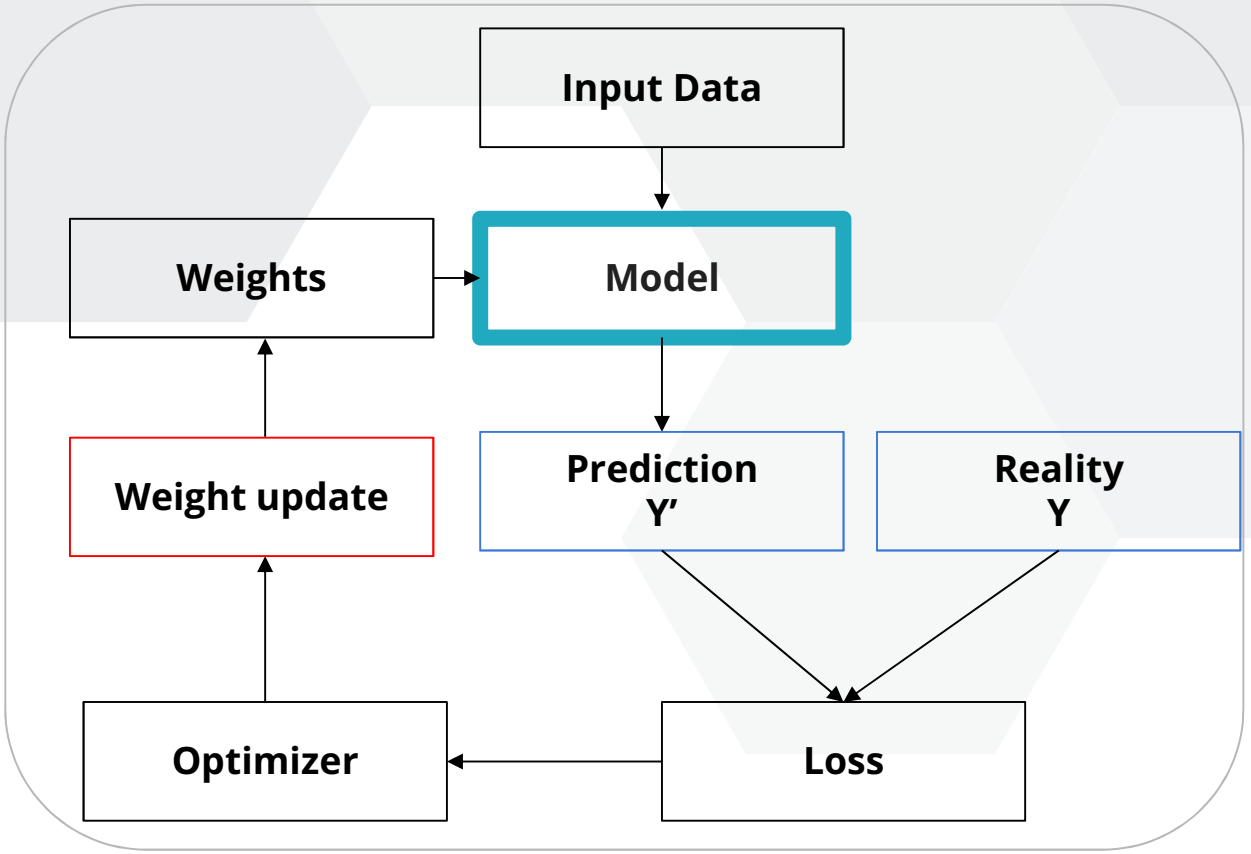
1. What are the benefits of shuffling the training data before creating batches?
2. What is the reason for adding nonlinearity?
3. What should be the output of an NN if we want to do classification?



Short break.
Back at ....

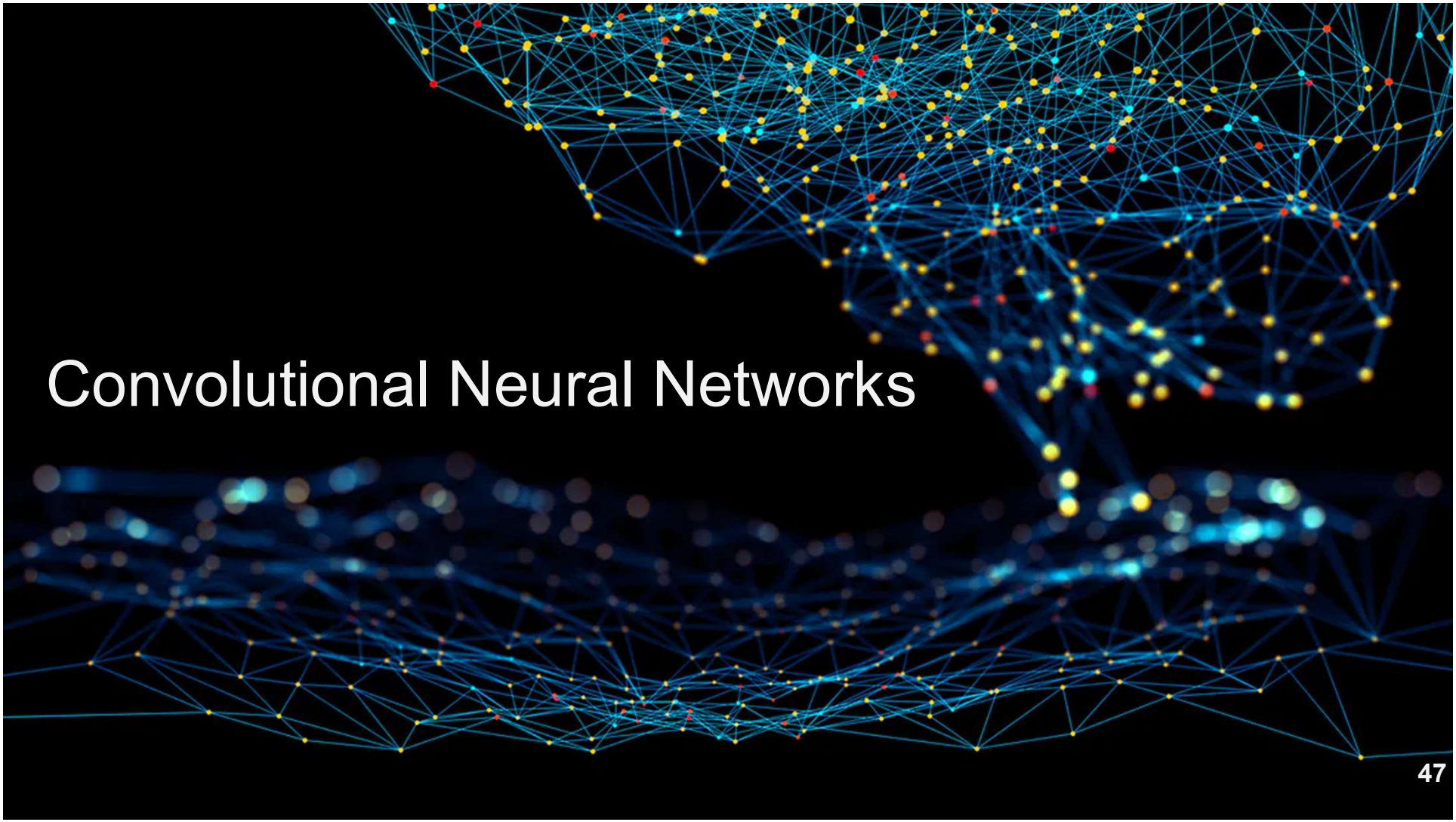
Review

- Linear regression
- Gradient
- Parameter
- Hyperparameter
- Epoch
- Batch size
- Learning rate
- Regression
- Classification
- Loss function
- Activation function
- MLP



Break

A convolutional neural network taking a coffee break
Panos x DALLE2


Convolutional Neural Networks

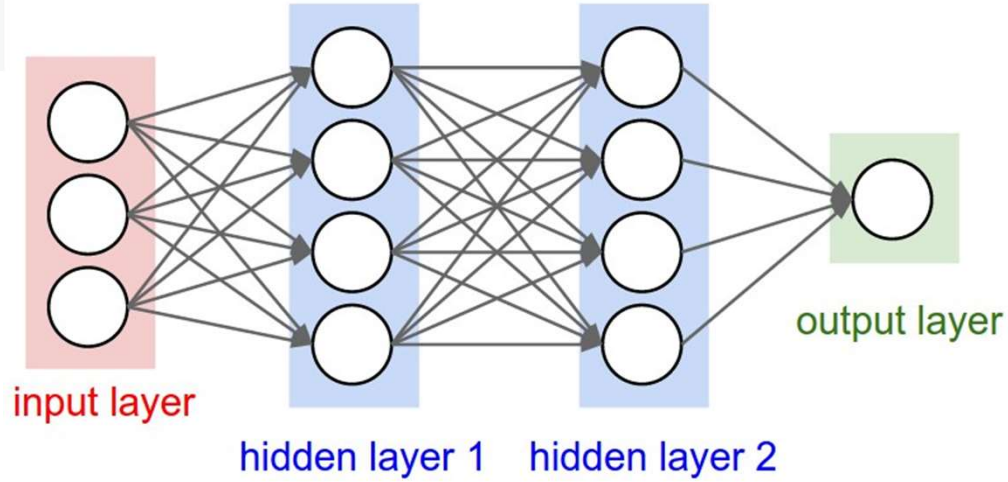
MLP drawbacks

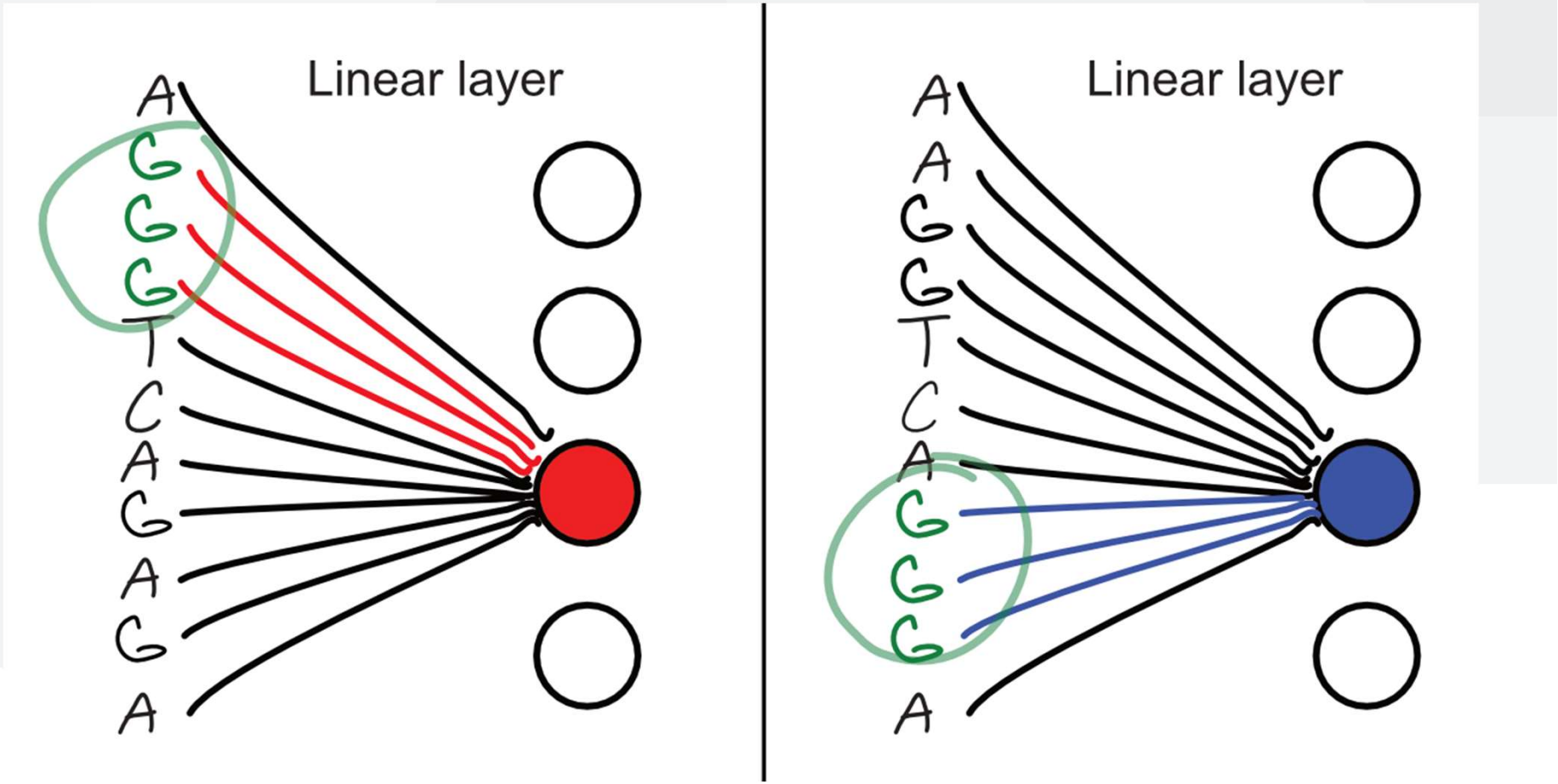
Weight for each value - too many parameters

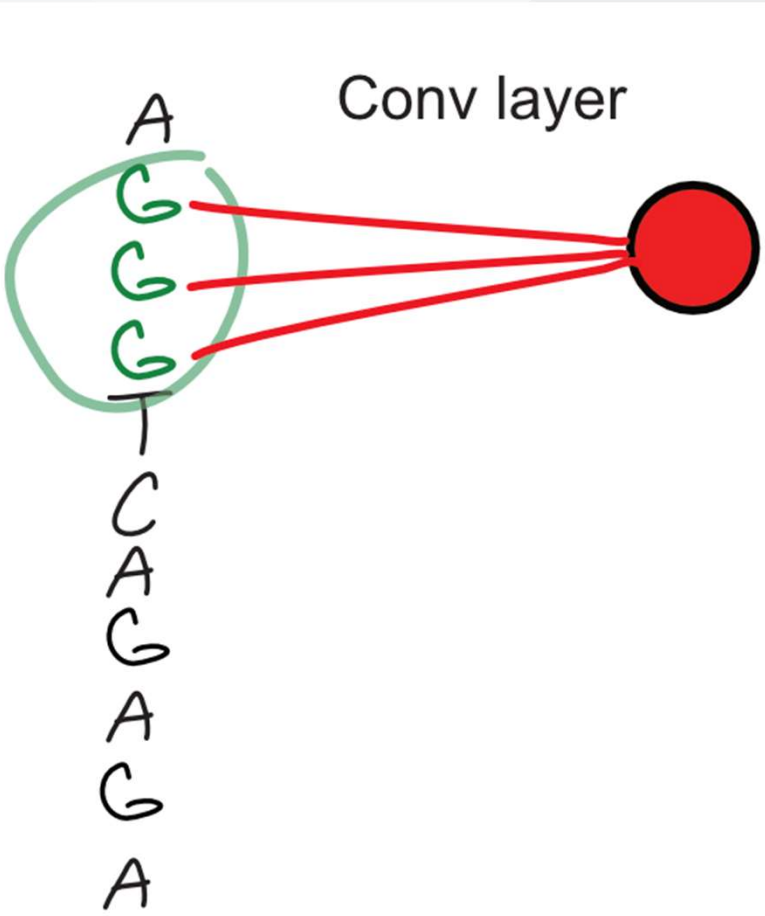
The **same pattern** in **different place** behaves differently

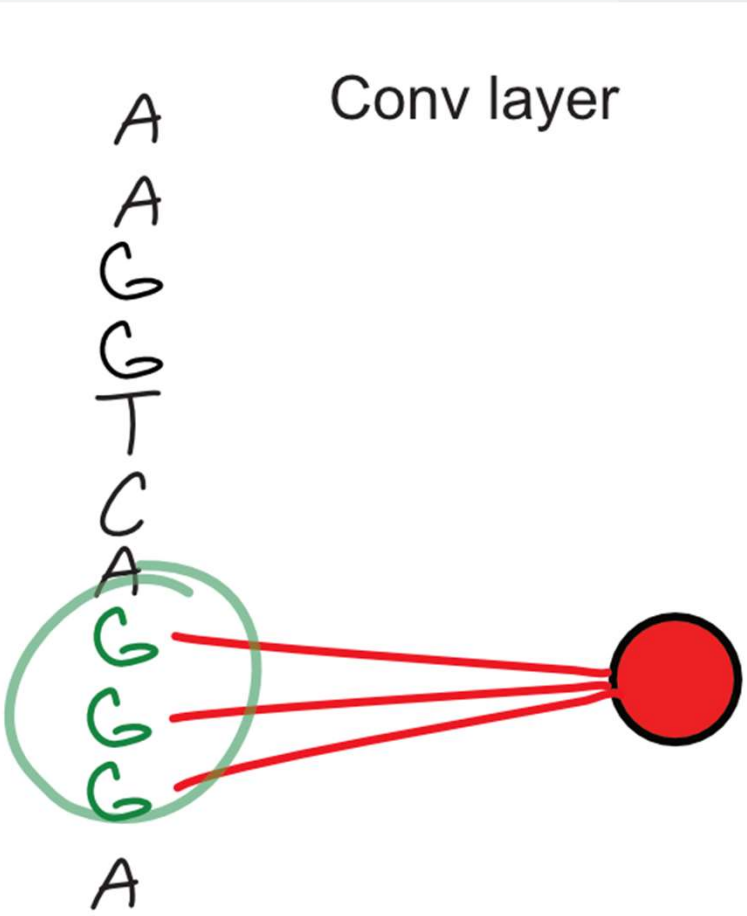
Treats the input as a row of values

- No proximity, no sense of local neighbourhood




Linear layer
A
G
G
G
T
C
A
G
A
G
A
Linear layer
A
A
G
G
T
C
A
G
G
G
A


Conv layer
A
G
G
G
T
C
A
G
A
G
A

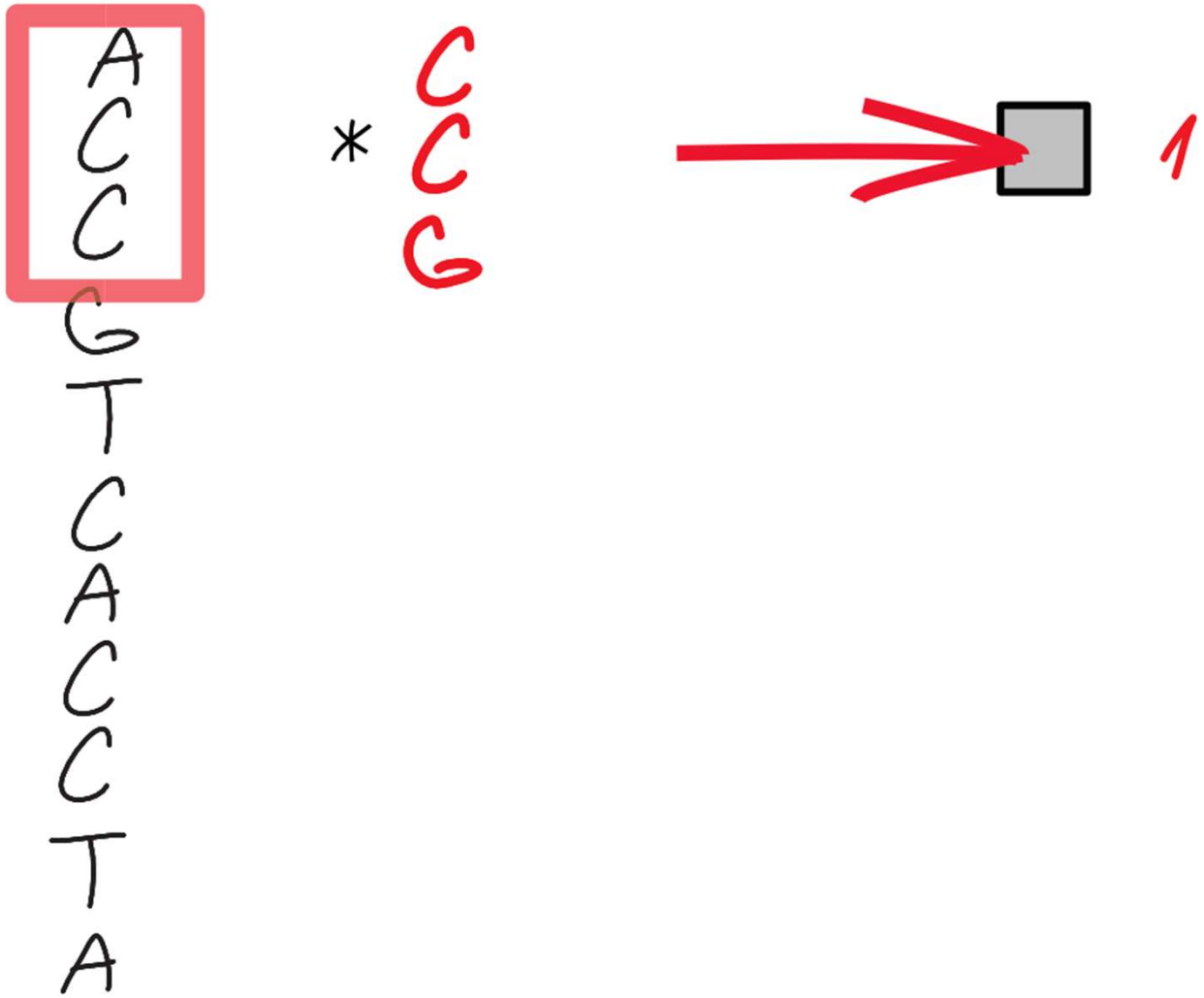

Conv layer
A
A
G
G
T
C
A
G
G
G
A

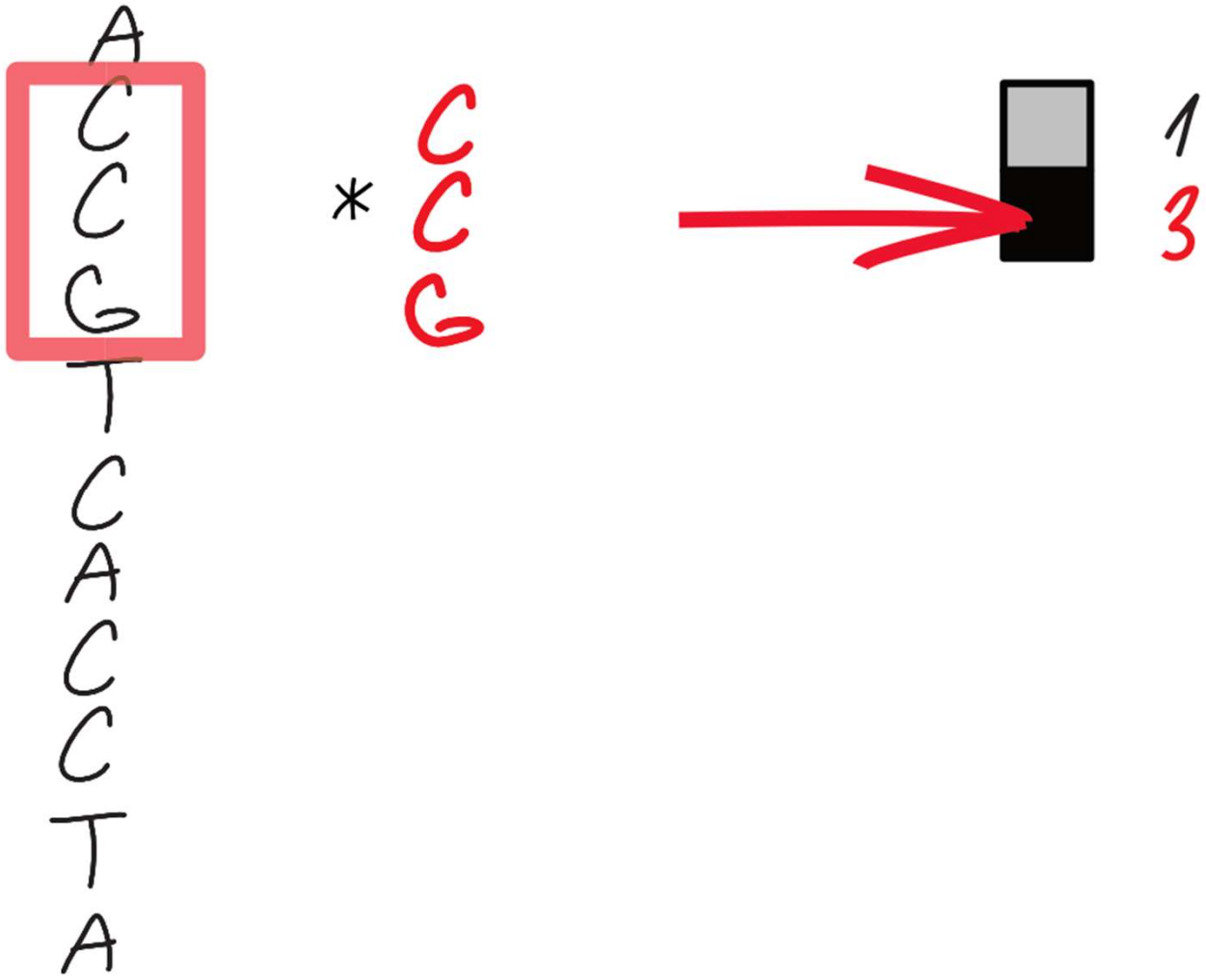
ACCGTCACCTA

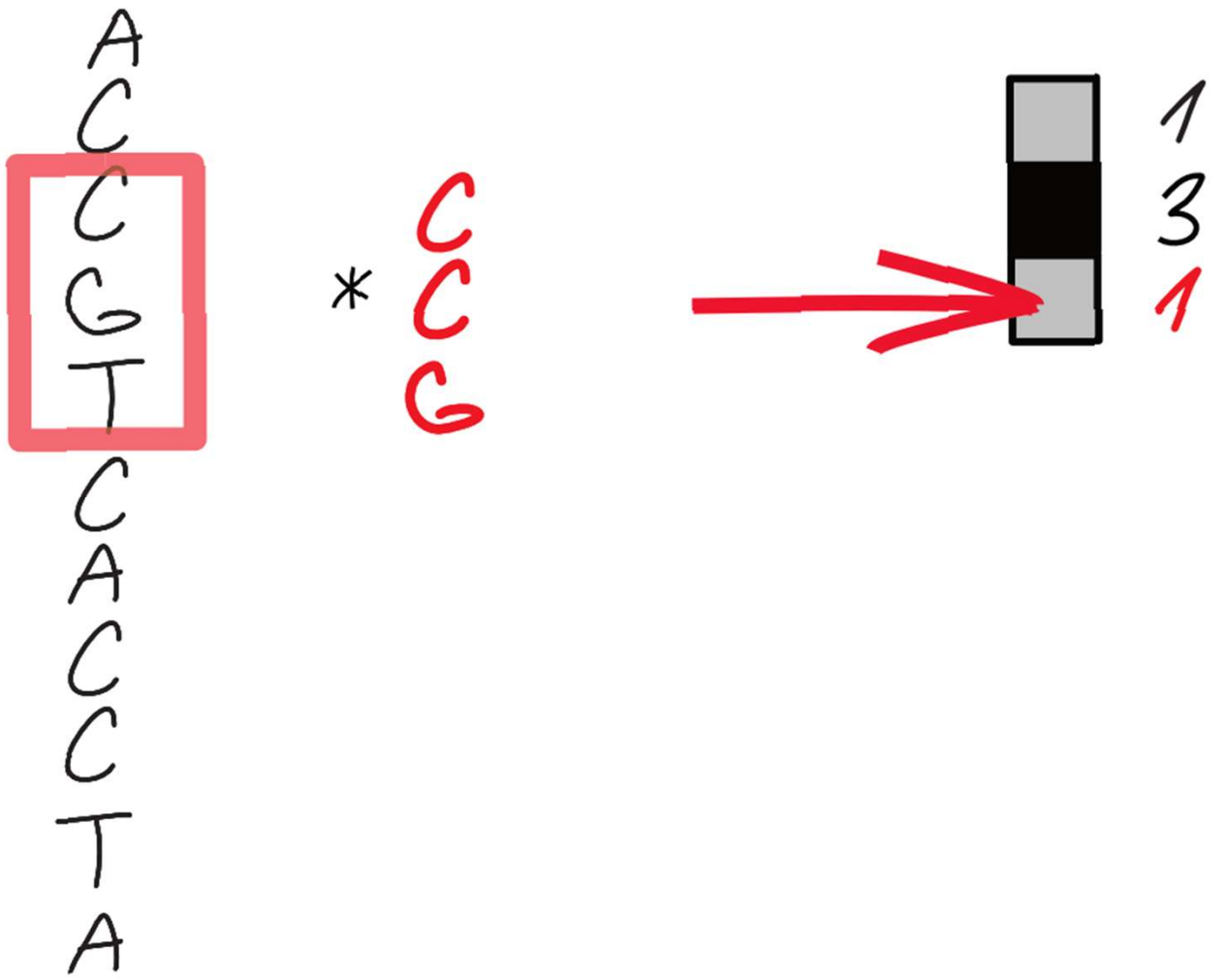
Search for the pattern

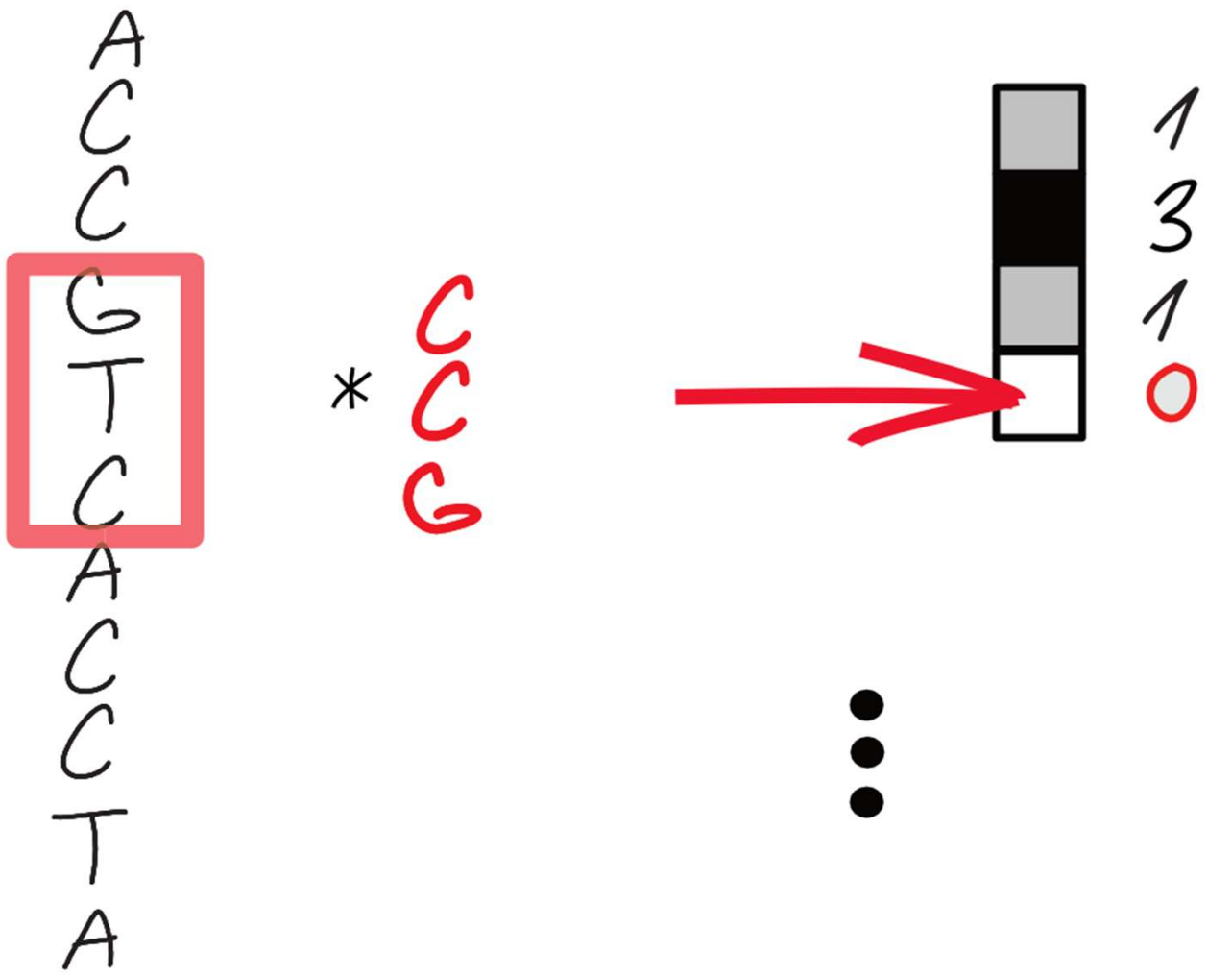
CCG

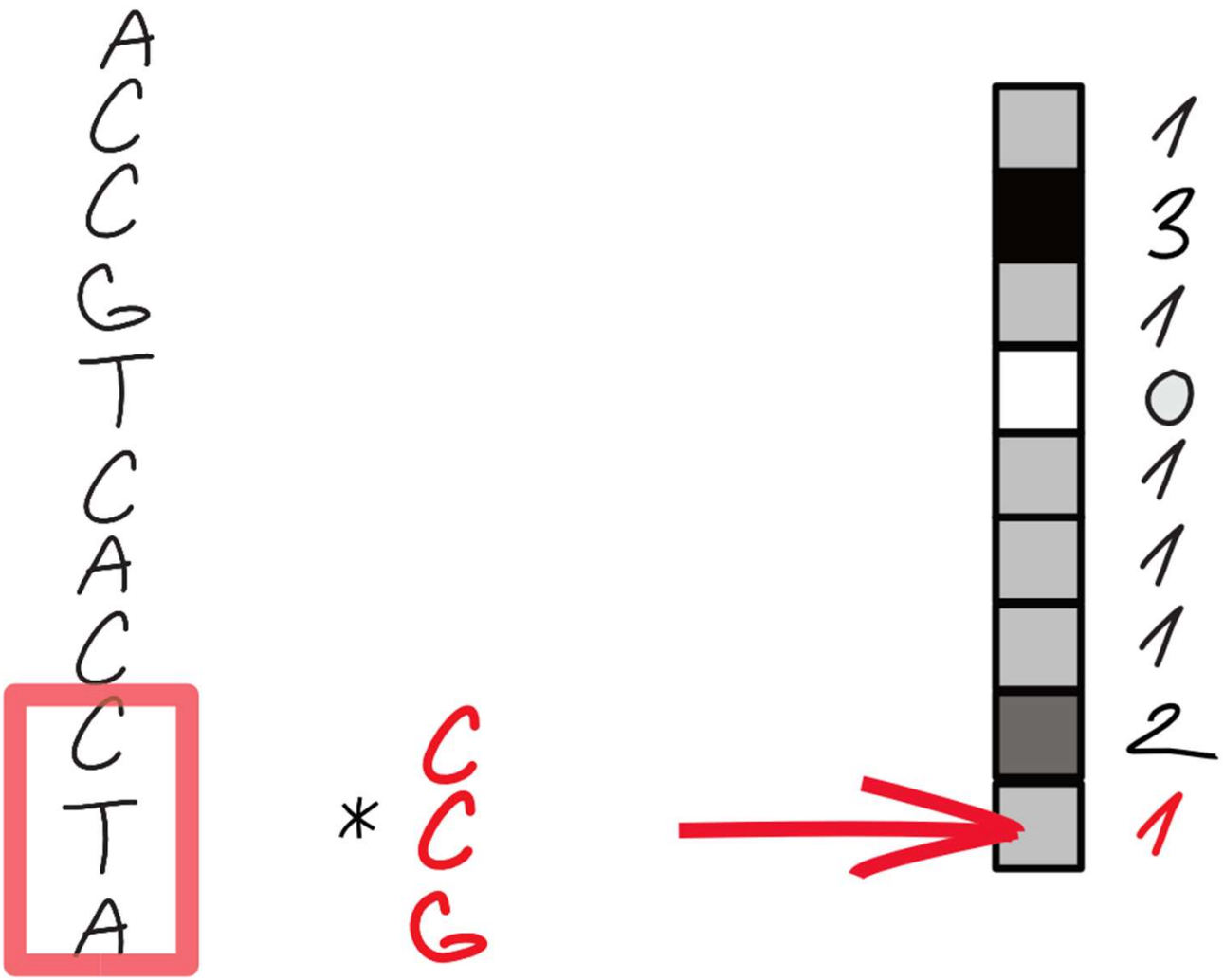
A
C
C
G
T
C
A
C
C
T
A
*
C
C
G

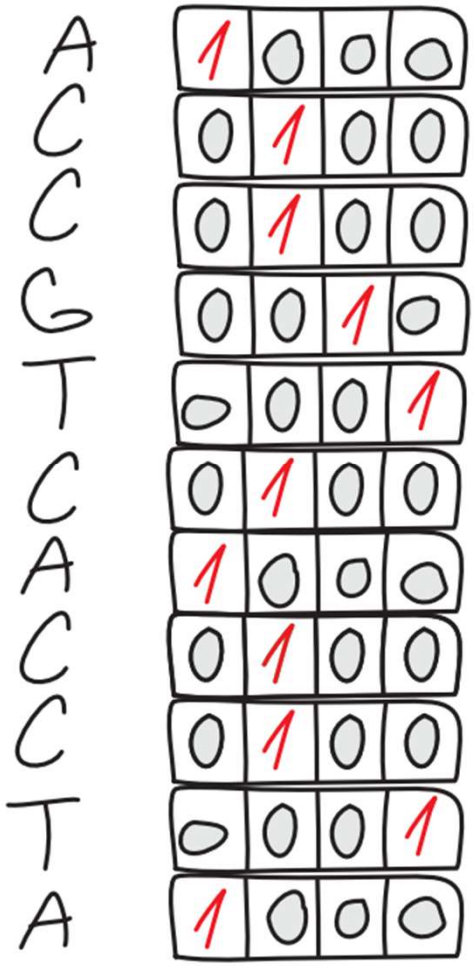

A
C
C
G
T
C
A
C
C
T
A
*
C
C
G
1

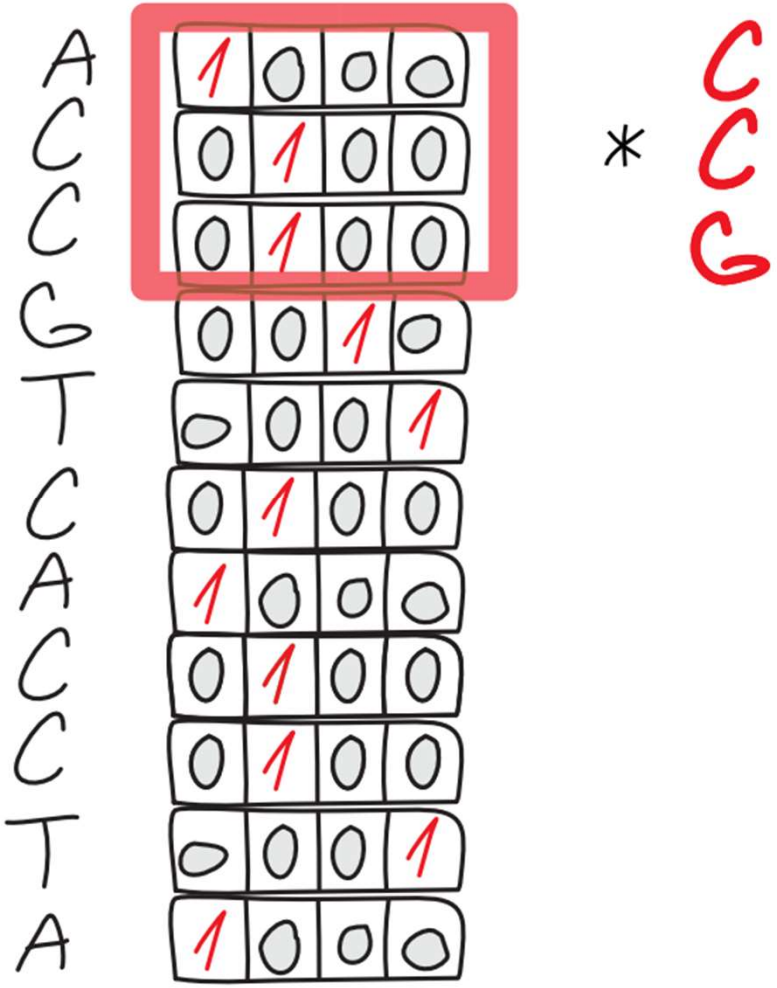

A
C
C
G
T
C
A
C
C
T
A
*
C
C
G
1
3

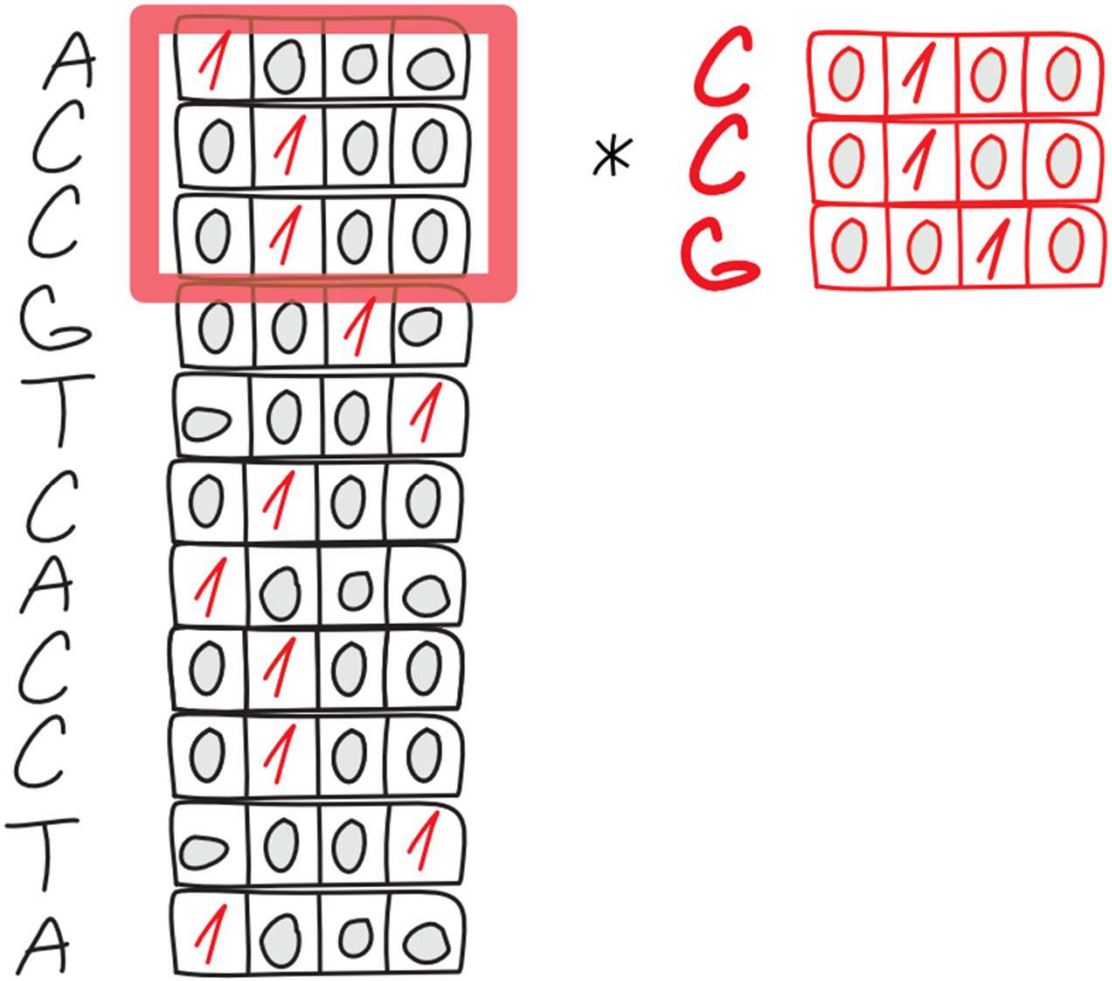

A
C
C
G
T
C
A
C
C
C
T
A
*
C
C
G
1
3
1

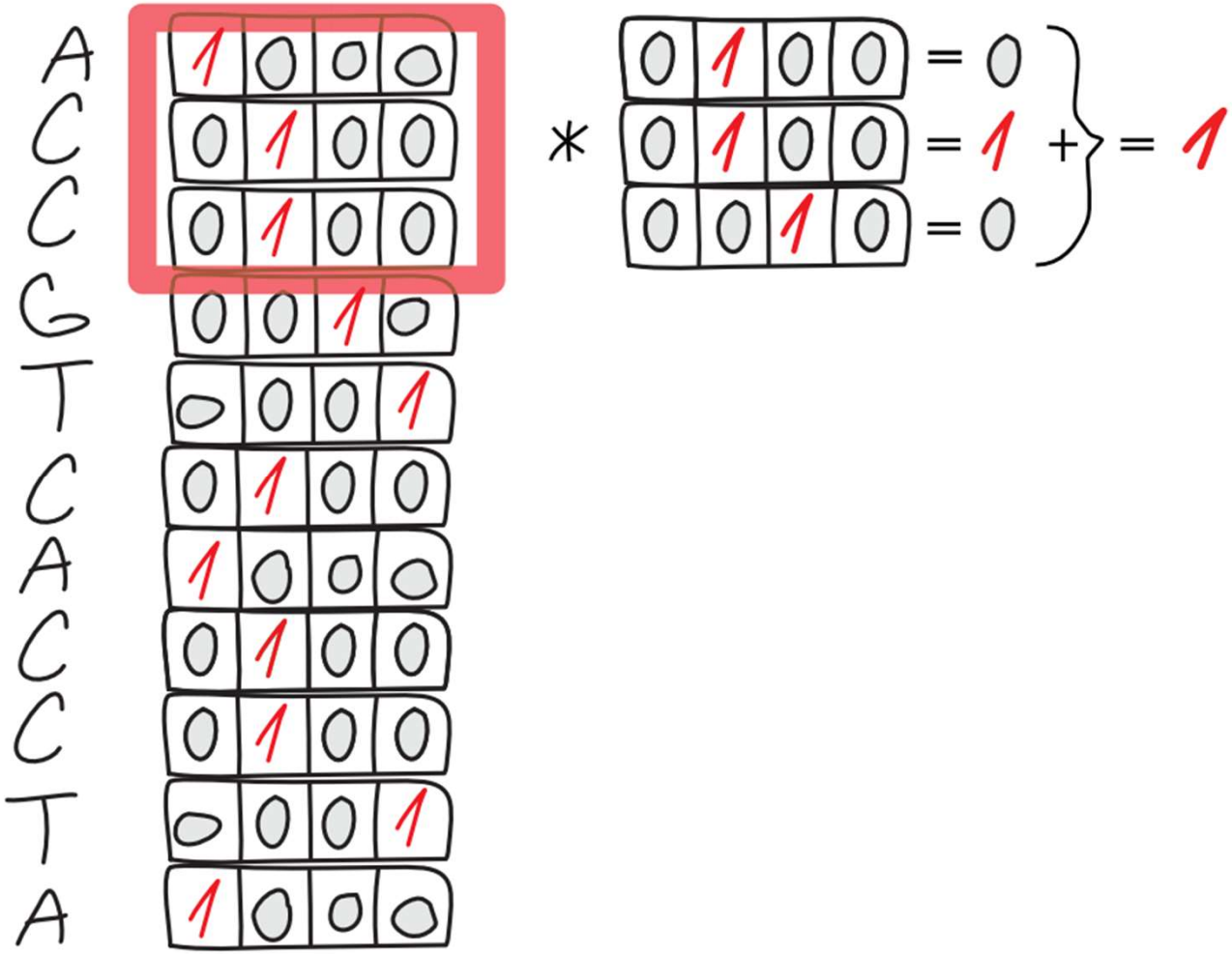

ACCGTCACCCTA
*
CCG
1310

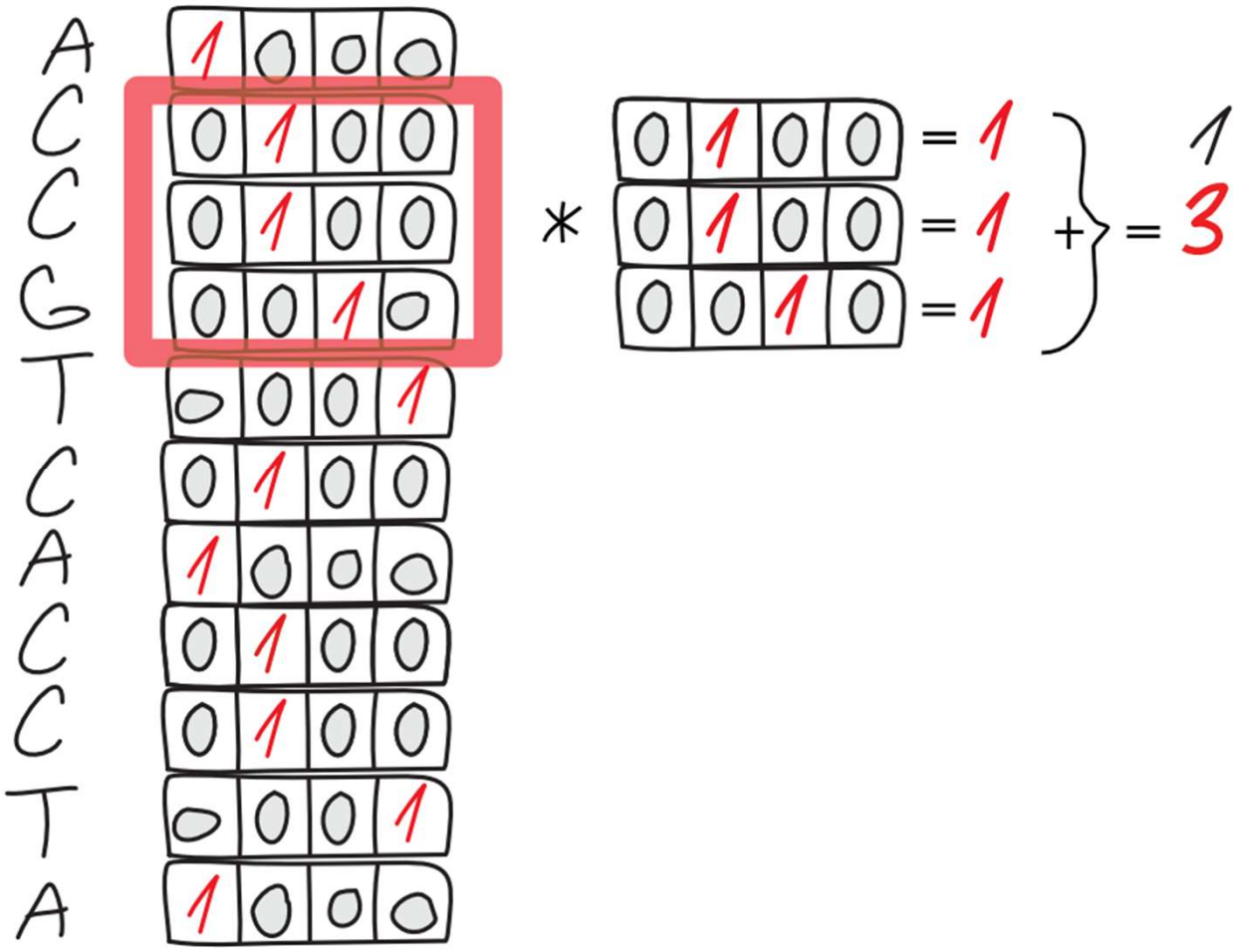

ACCGTCACCTA
*
CCG
1
3
1
0
1
1
1
2
1

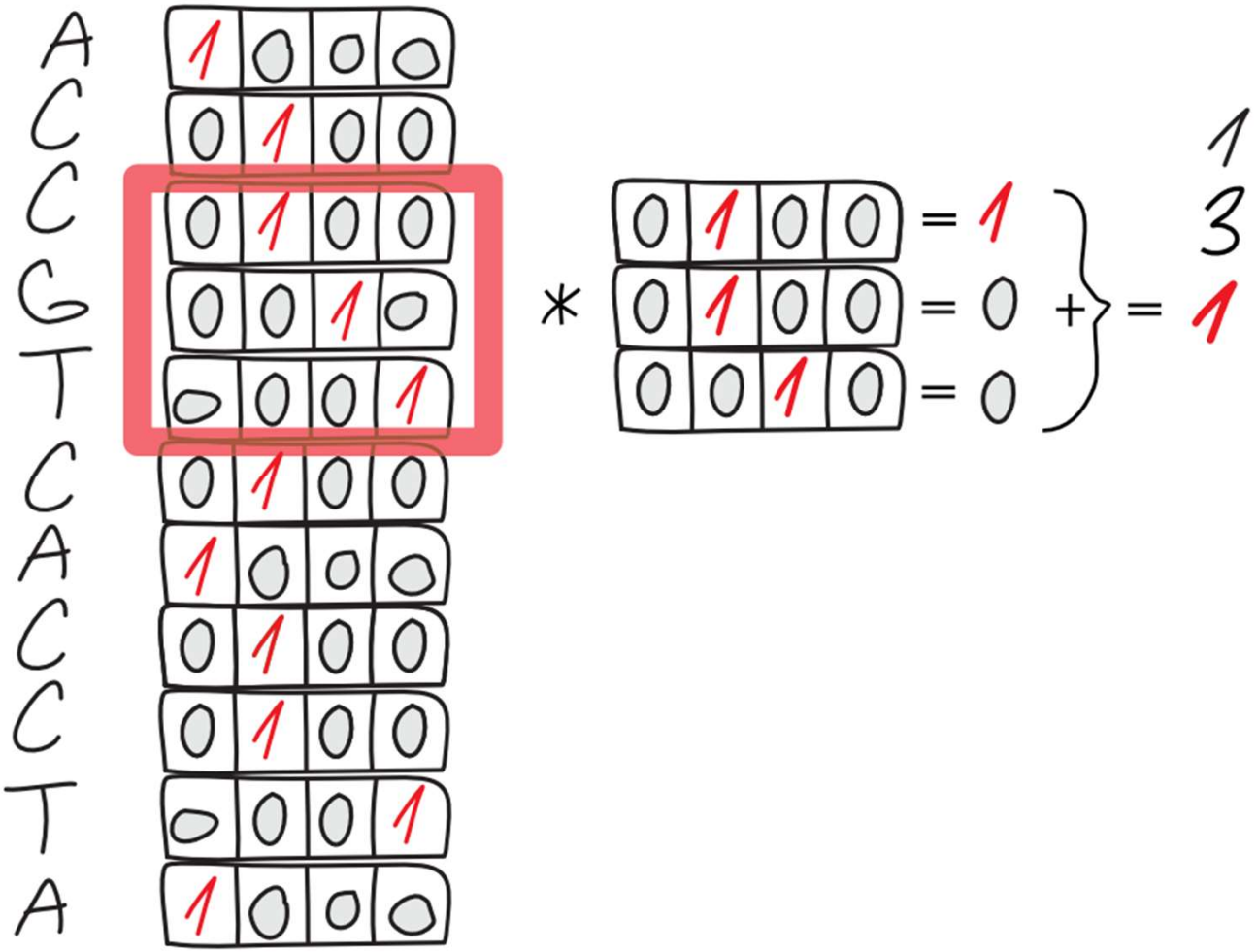

A
C
C
G
T
C
A
C
C
T
A
1 0 0 0
0 1 0 0
0 1 0 0
0 0 1 0
0 0 0 1
0 1 0 0
1 0 0 0
0 1 0 0
0 1 0 0
0 0 0 1
1 0 0 0

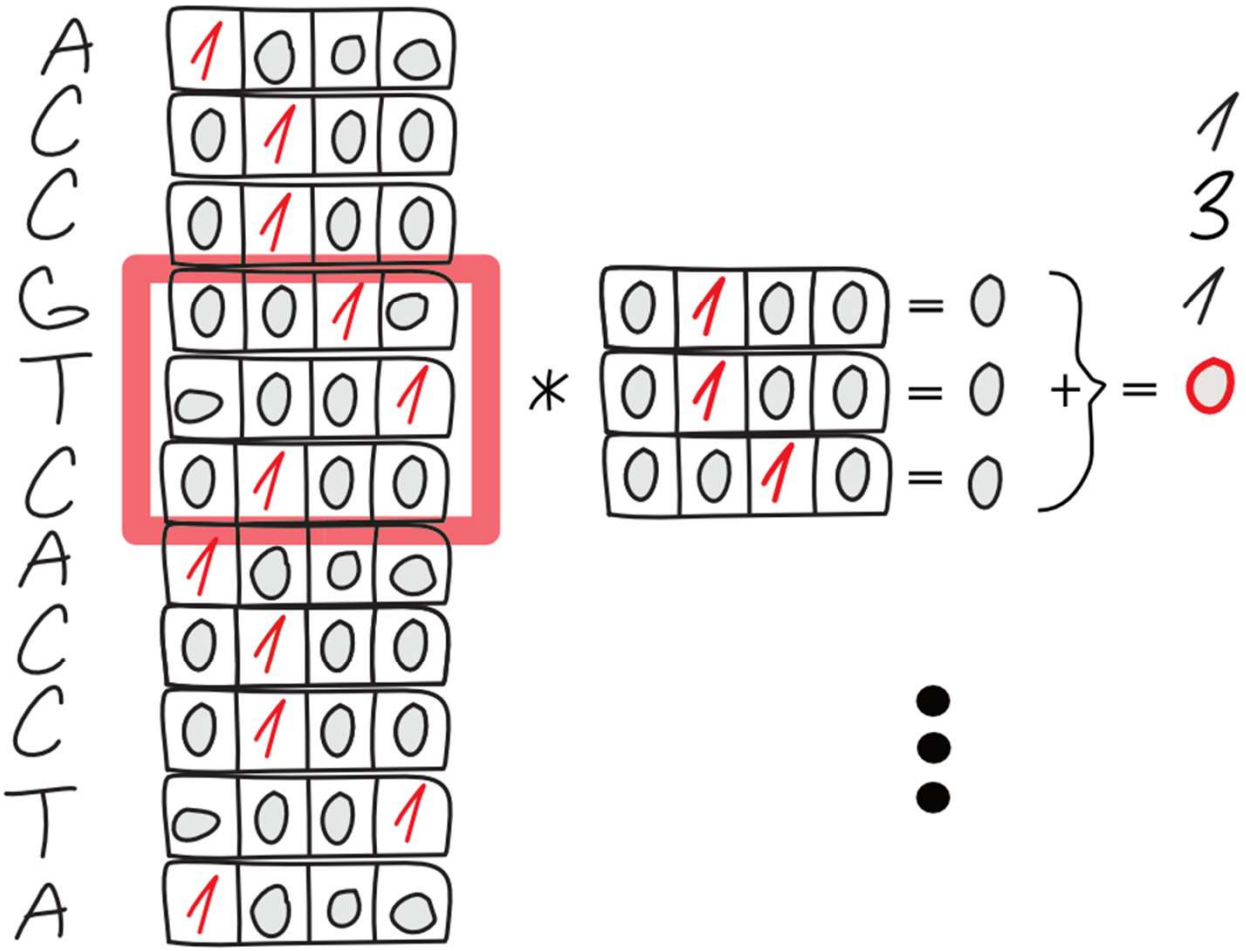

A
C
C
G
T
C
A
C
C
T
A
*
C
C
G

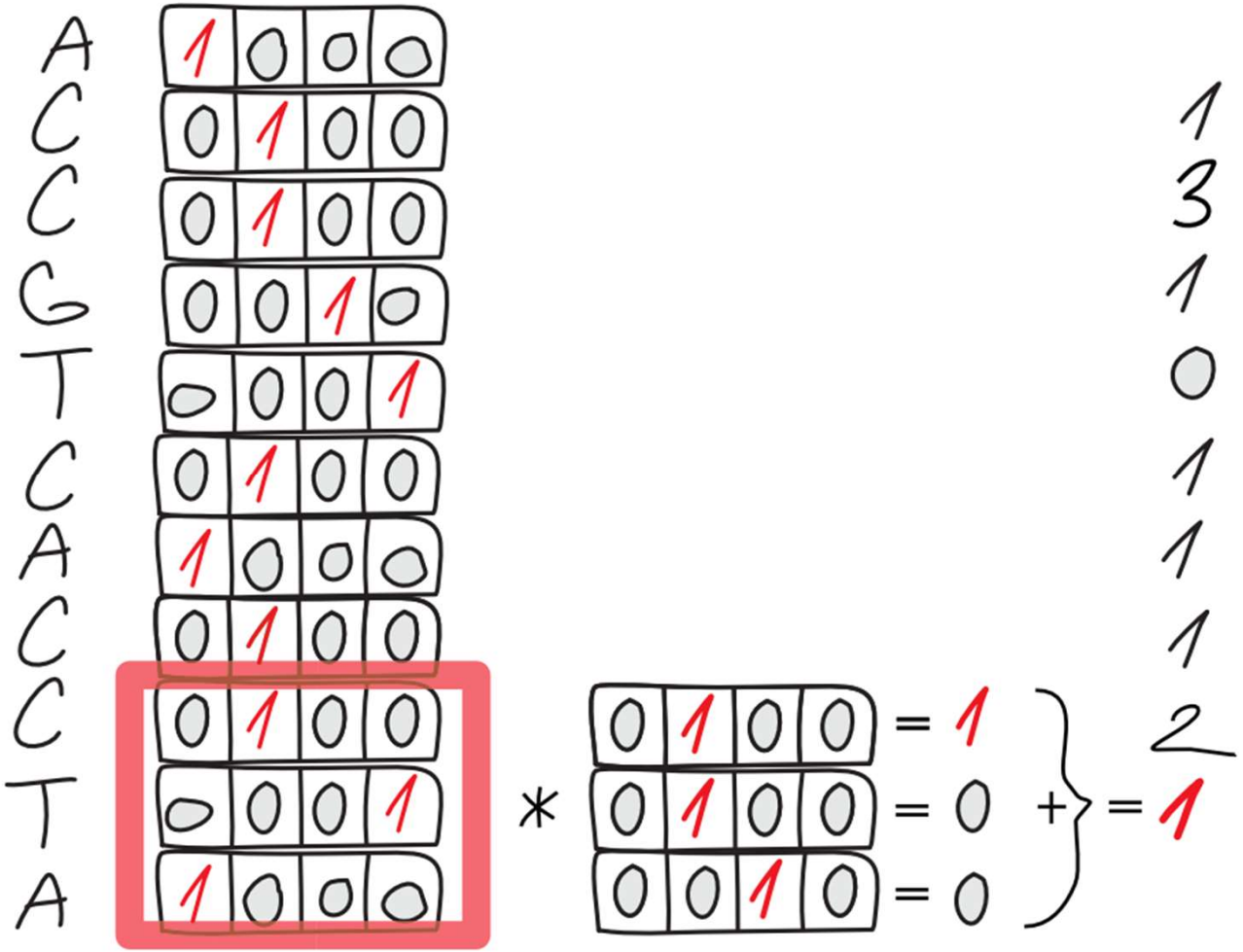

A
C
C
G
T
C
A
C
C
T
A
*
C
C
G

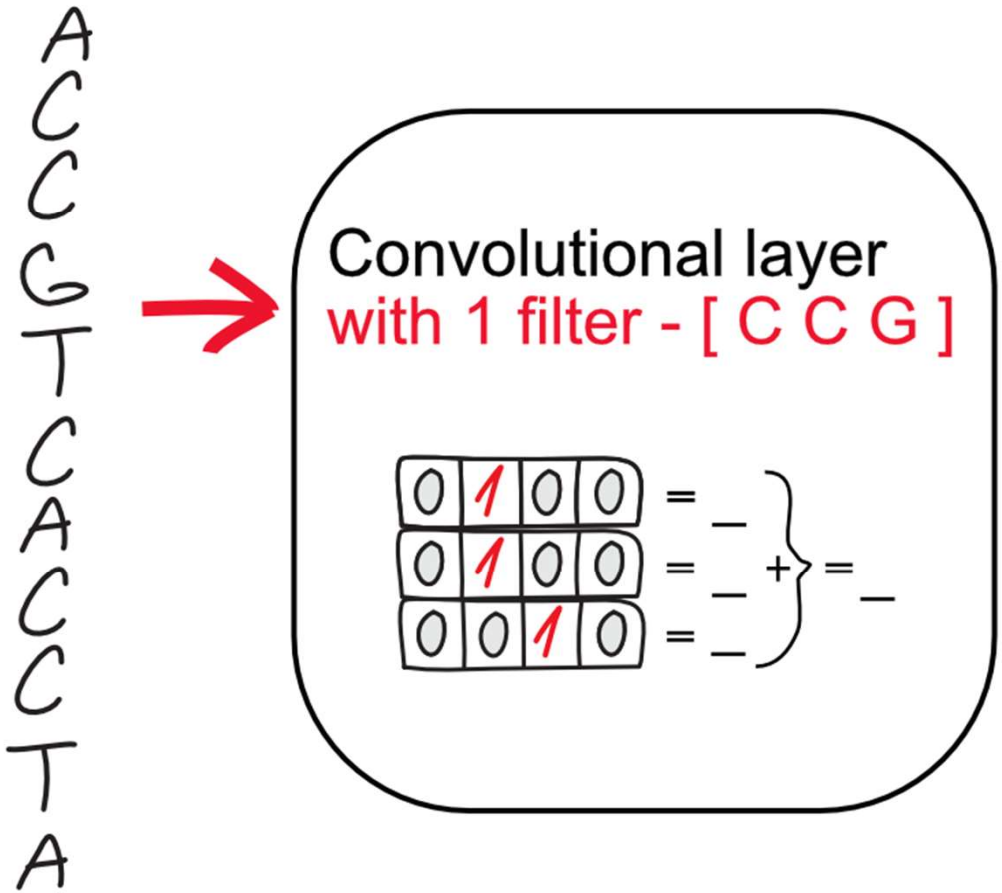

A
C
C
G
T
C
A
C
C
T
A
*
= 0
= 1
+
= 1

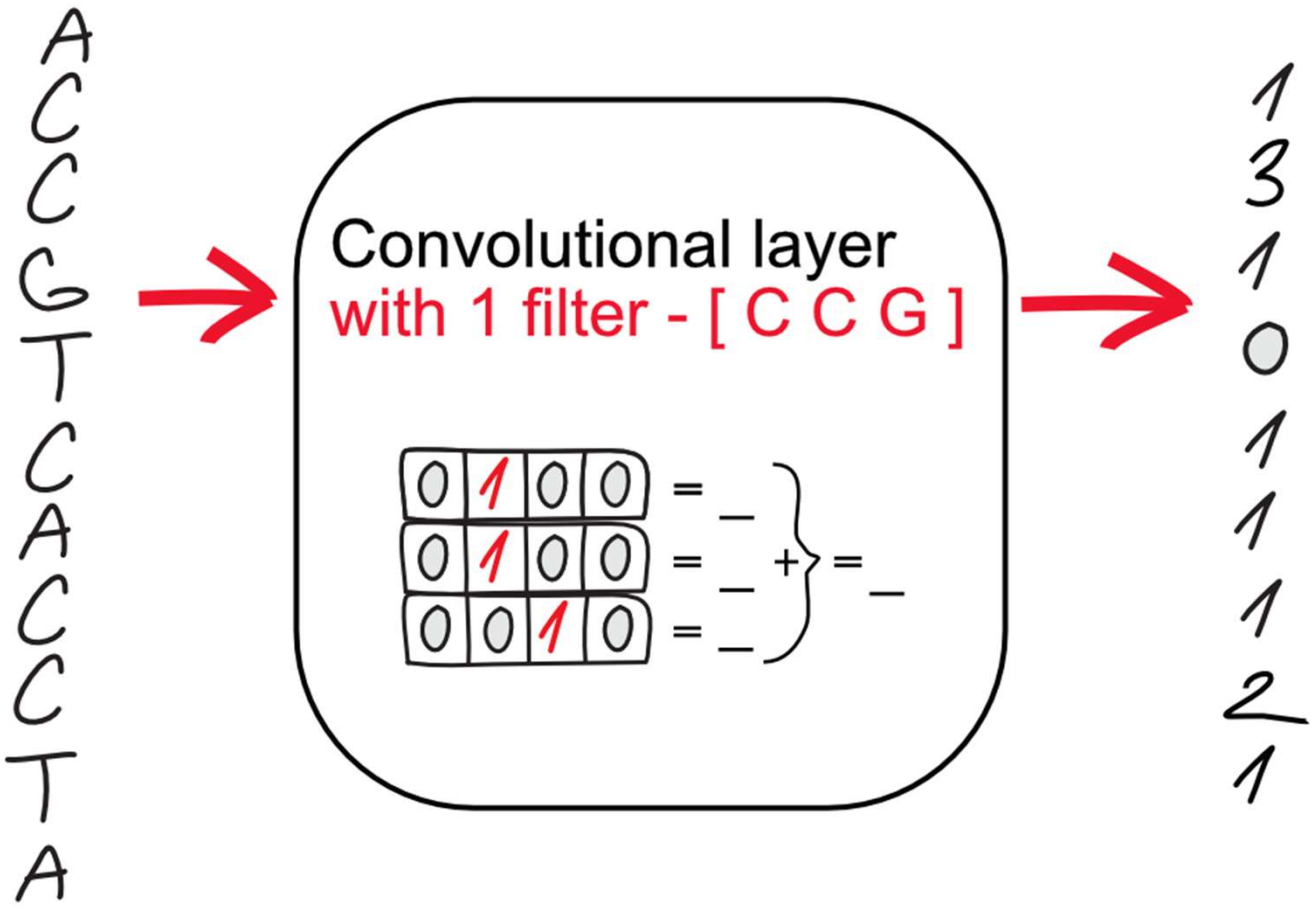

A
C
C
G
T
C
A
C
C
T
A
*
= 1
= 1
= 1
+
= 3
1

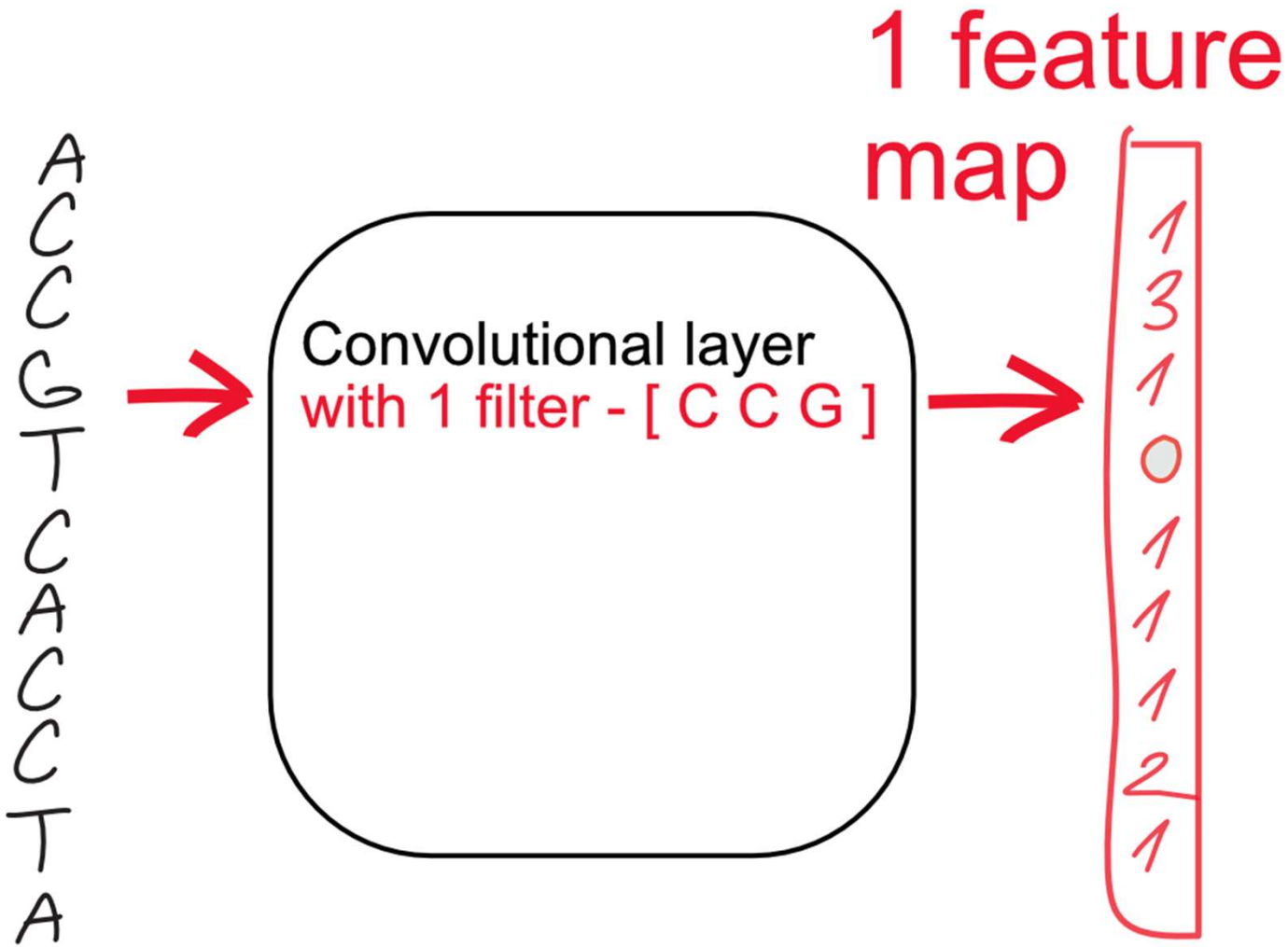

A
C
C
G
T
C
A
C
C
T
A
*
= 1
= 0
= 0
+
=
1
3
1

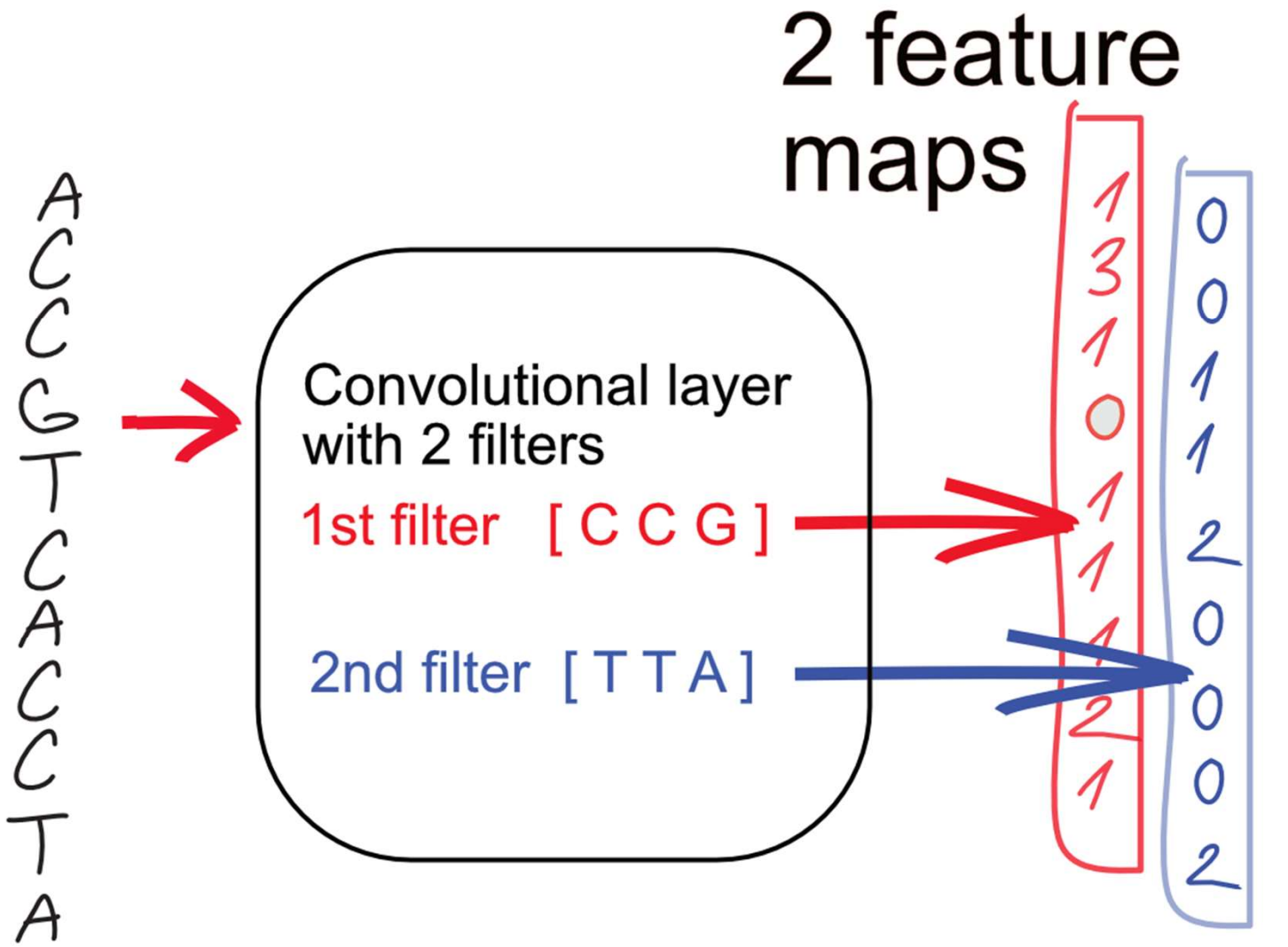

A
C
C
G
T
C
A
C
C
T
A
*
= 0
= 0
= 0
+
=
1
3
1
0


A
C
C
G
T
C
A
C
C
T
A
* = 1
= 0
= 0
+ = 1
1
3
1
0
1
1
1
2


A
C
C
G
T
C
A
C
C
T
A
Convolutional layer
with 1 filter - [C C G]
0 1 0 0 = _
0 1 0 0 = _
0 0 1 0 = _
+
= _

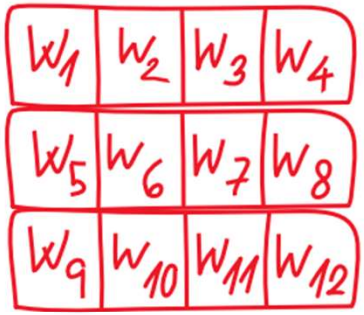

A C C G T C A C C T A
Convolutional layer
with 1 filter - [C C G]
0 1 0 0 = _
0 1 0 0 = _
0 0 1 0 = _
+ = _
1 3 1 0 1 1 1 2 1


1 feature map
A C C G T C A C C T A
Convolutional layer
with 1 filter - [C C G]
1 3 1 0 1 1 1 2 1

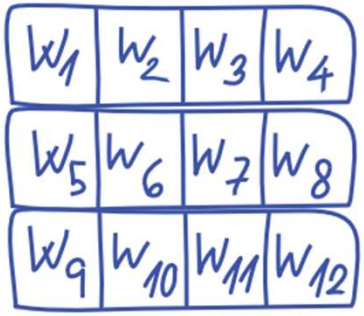

2 feature maps
A C C G T C A C C T A
Convolutional layer with 2 filters
1st filter [C C G]
2nd filter [T T A]
1 3 1 0 1 1 1 2 1
0 0 1 1 2 0 0 0 2

convolutional filters learn their weights

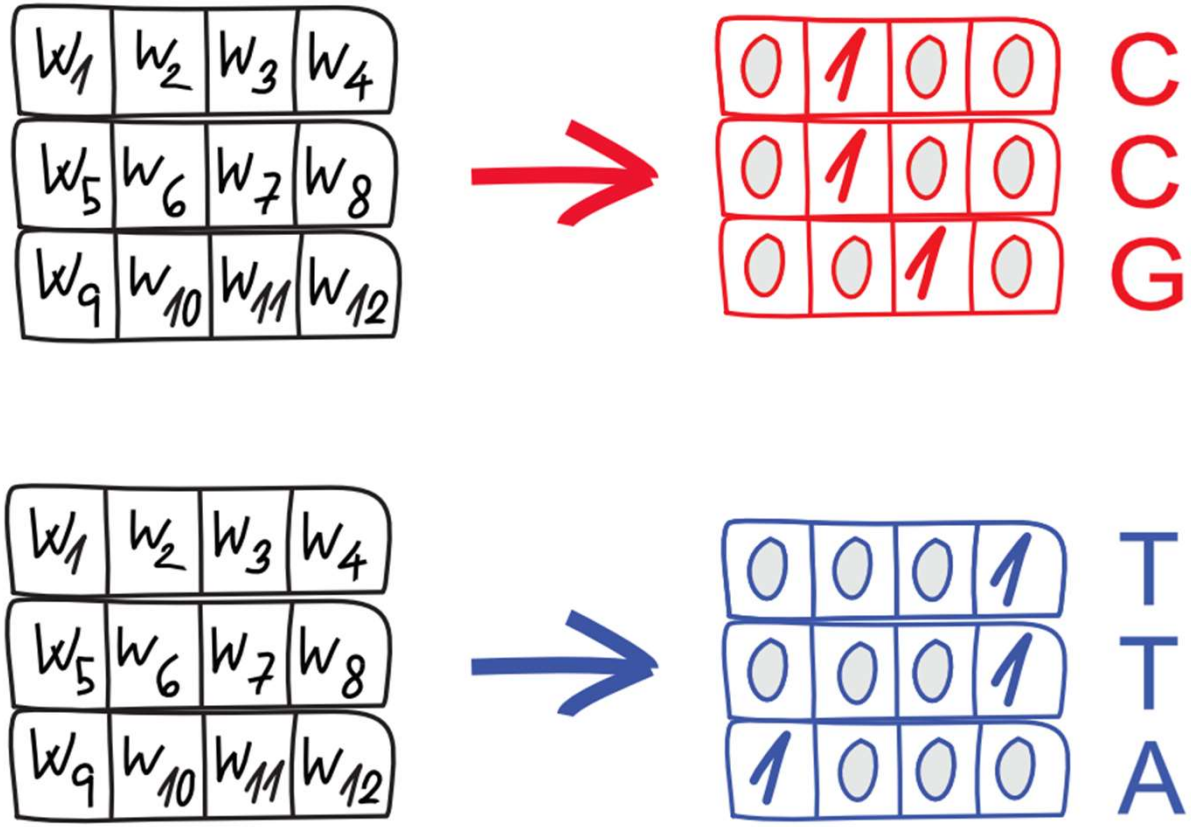
1st filter



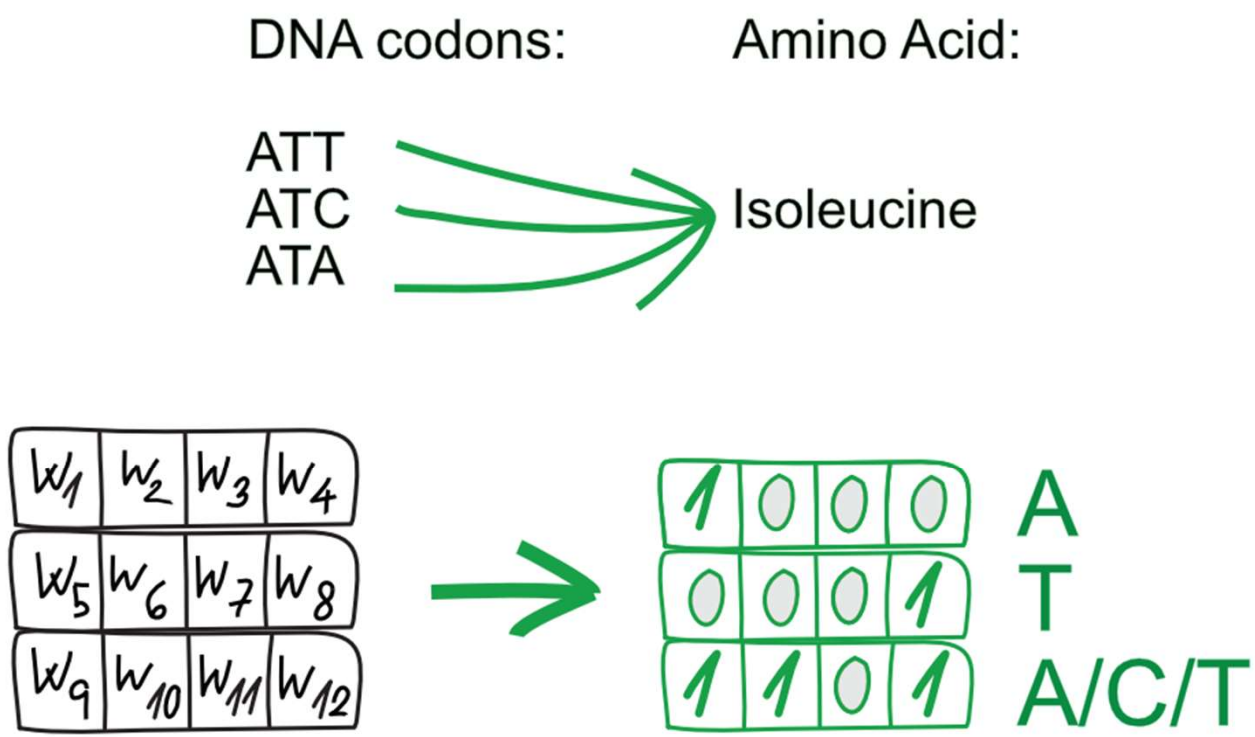
2nd filter



convolutional filters learn their weights



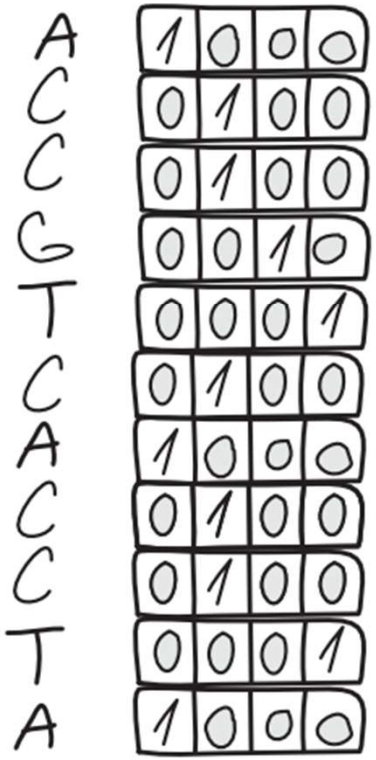
convolutional filters learn their weights



Convolutional layer

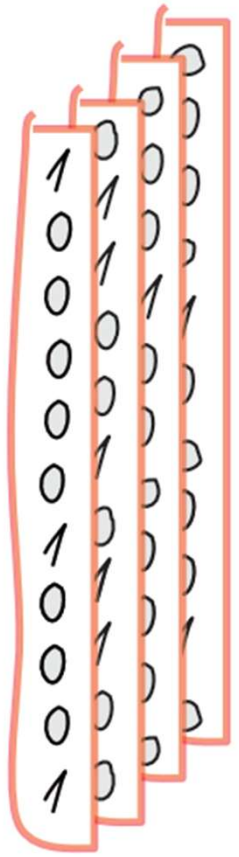
- Any number of filters
- Weights can be any number
  - 0.123
  - -0.003
  - Usually very close to 0 for “math reasons”

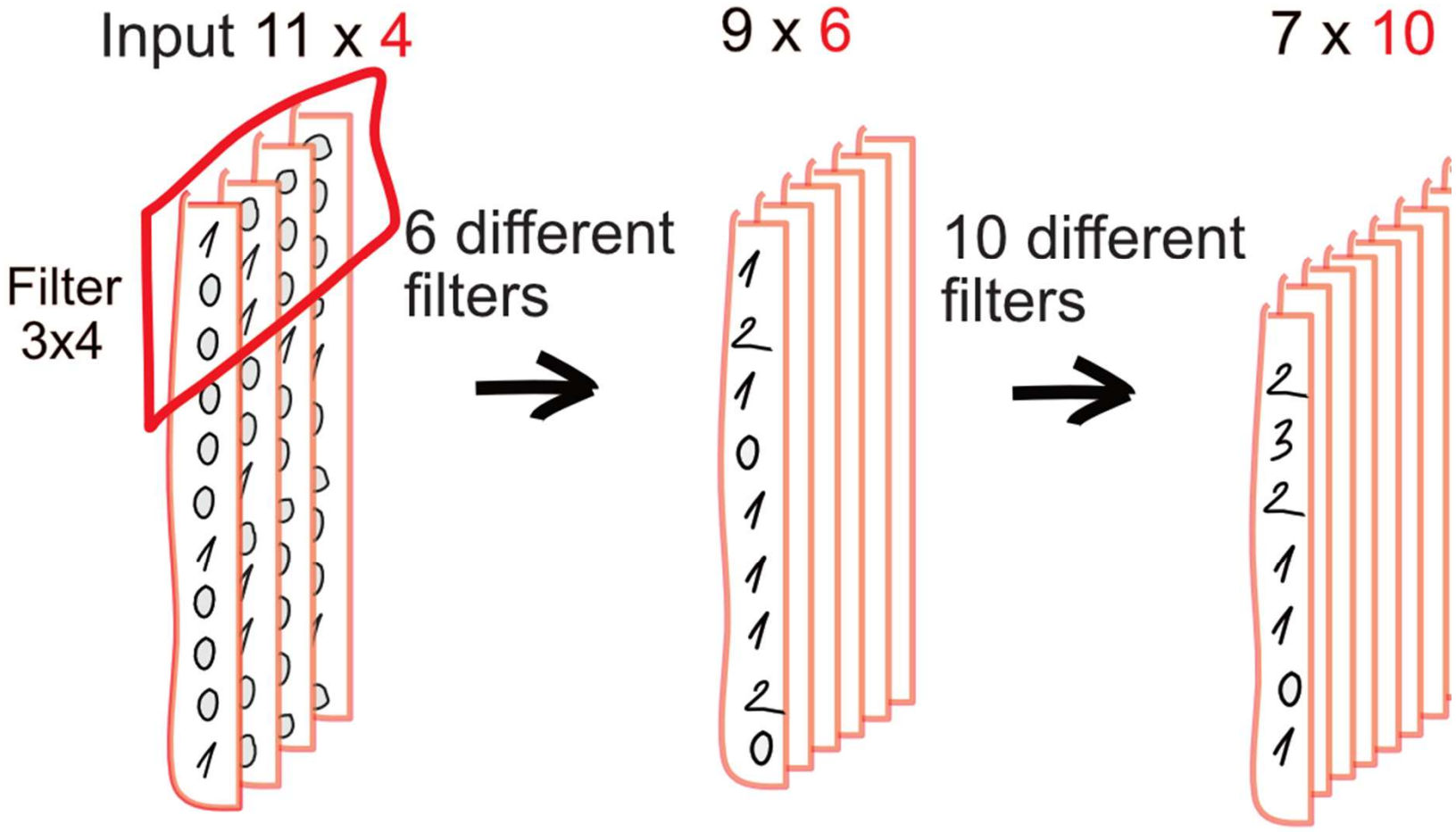
Input 11 x 4



Imagine as →

4 channels
(feature maps)




Input 11 x 4
Filter
3x4
6 different
filters
9 x 6
10 different
filters
7 x 10

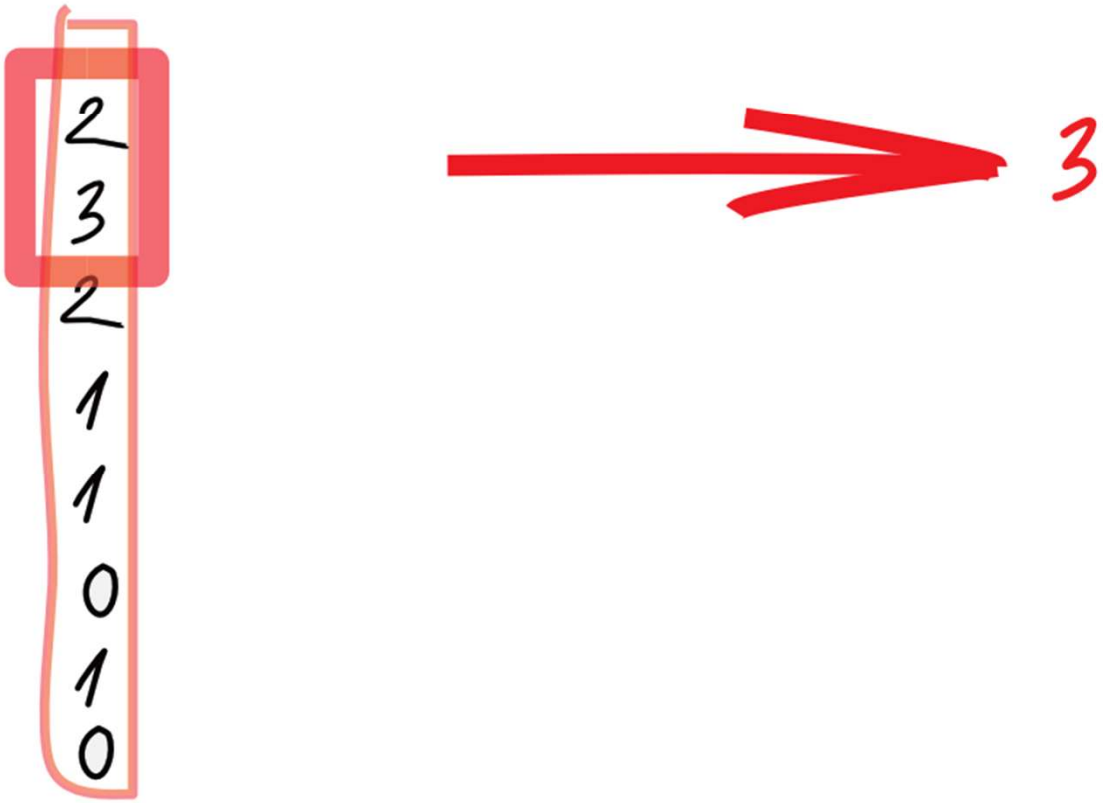
(Max) pooling layer

Decrease the spatial dimensions of the input (feature maps)

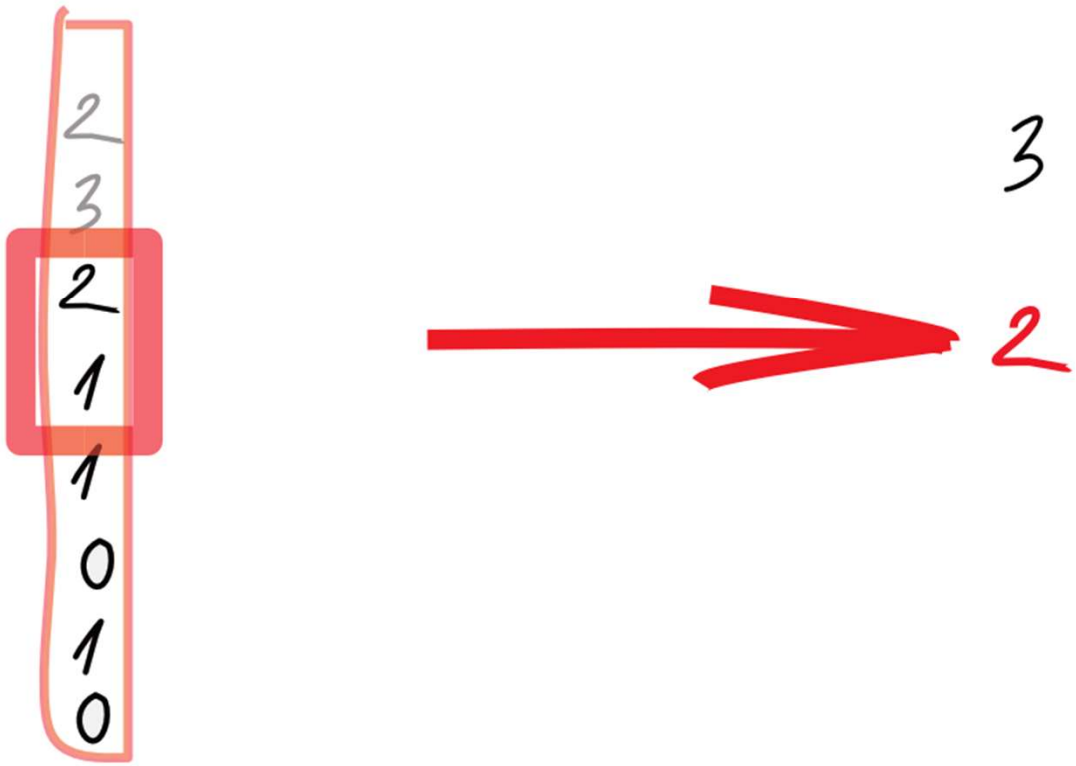
=> less computational steps

Translation invariance

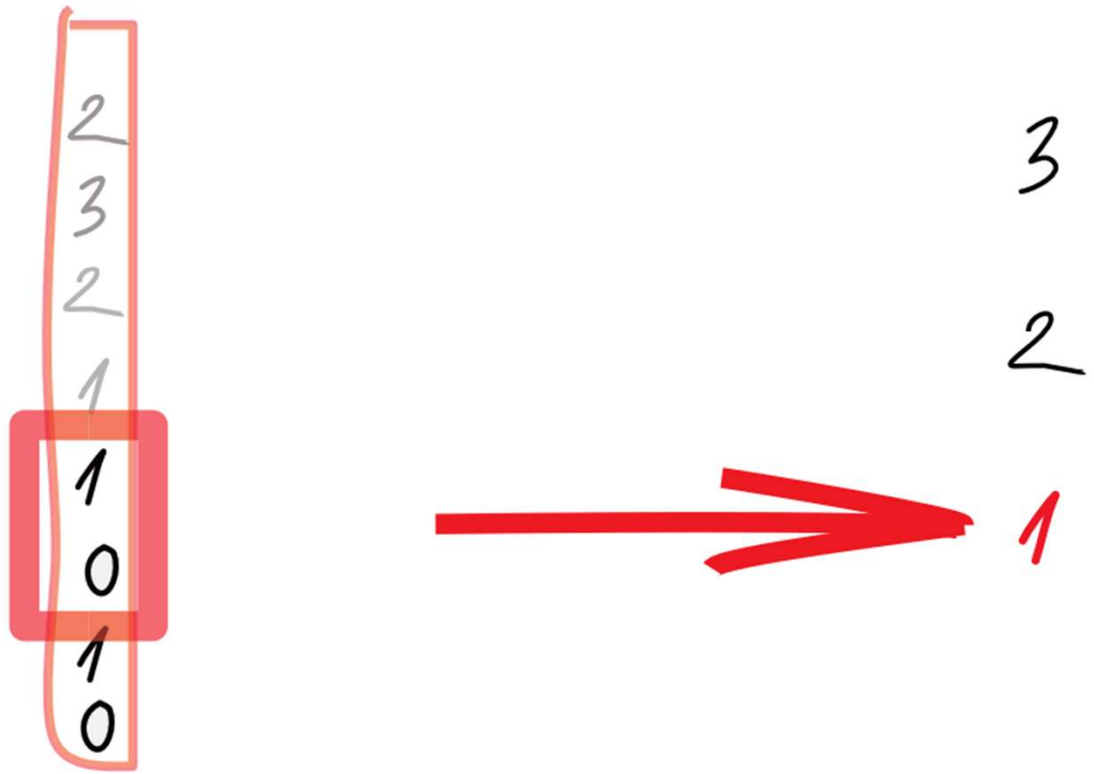
Max pooling layer of size 2



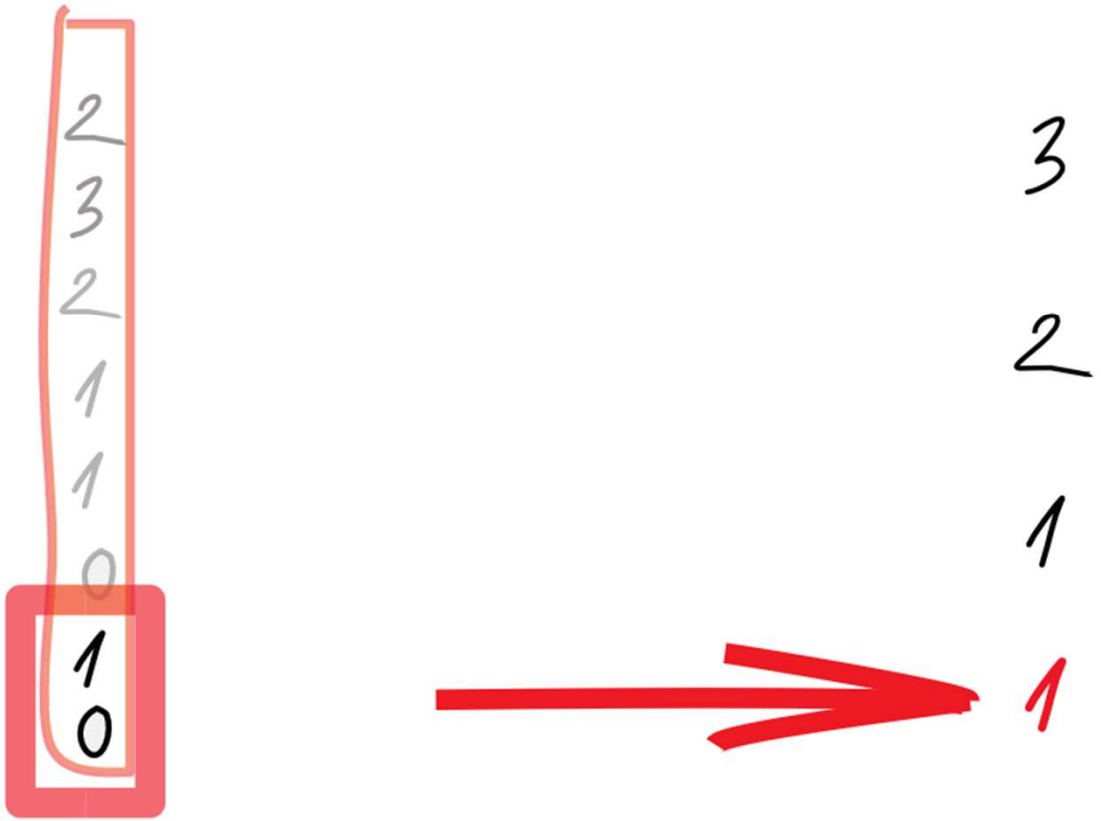
Max pooling layer of size 2



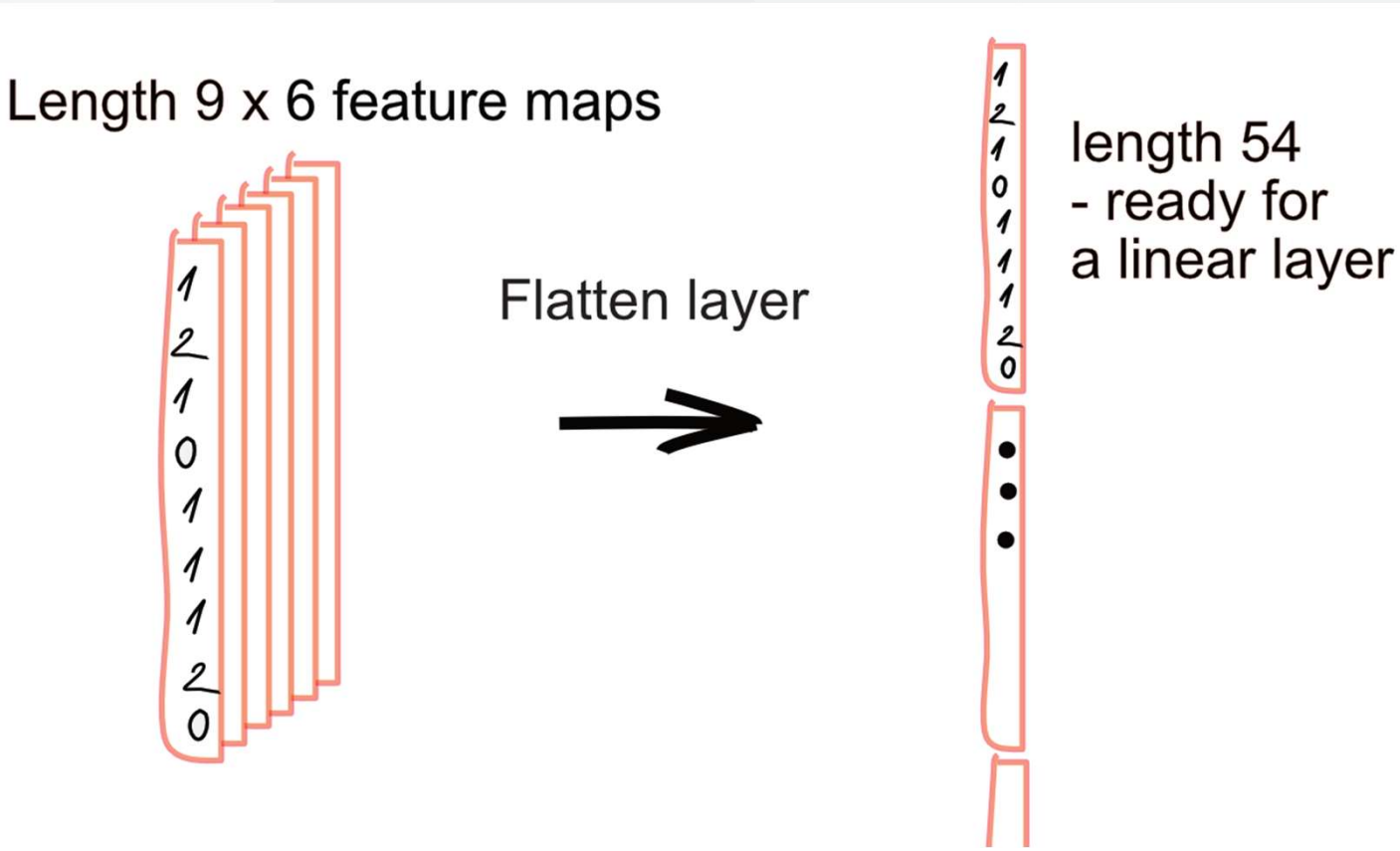
Max pooling layer of size 2



Max pooling layer of size 2



Flatten layer - spread the values



Convolutional NN - DNA enhancers classification

https://github.com/BioGeMT/MALTAomics-Summer-School/

Day 2 Workshop

MALTA_02_DNA_enhancers.ipynb

Quiz CNNs

- How many filters can a convolutional layer have?
- Did we just work with a 2-dimensional or 1-dimensional convolution layer?
- What is a kernel?

The Team

Panos
Panagiotis Alexiou

Petr
Petr Simecek

Vlasta
Vlastimil Martinek

David
David Cechak

Katarina
Katarina Gresova

martinekvlastimil95@gmail.com
Whatsapp +420 732 425 407

davidcechak@gmail.com


Thank you for your Attention
MARIE CURIE ACTIONS
metacentrum
cesnet
elixir
CZECH REPUBLIC
GAČR
GRANTOVÁ AGENTURA ČESKÉ REPUBLIKY
CEITEC
INTEG-RNA
EMBO