

Getting started with neural networks

R/keras Part 2

Lead: Prof. Mark Robinson

Deepak Tanwar

 tanward@ethz.ch

 @d_k_tanwar

18th March, 2020

Content

Introduction

Deep learning frameworks

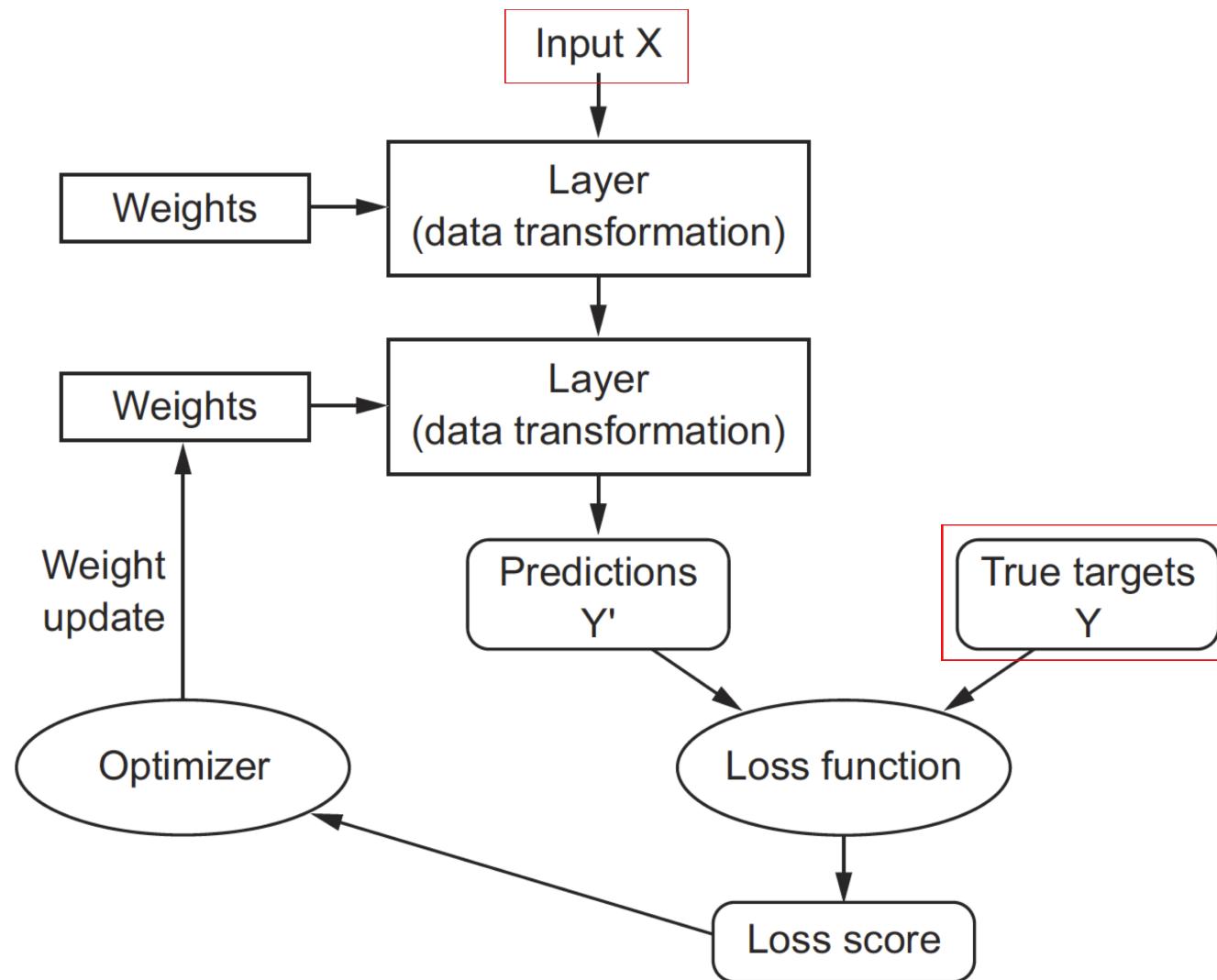
Classification

Train, validation and test sets

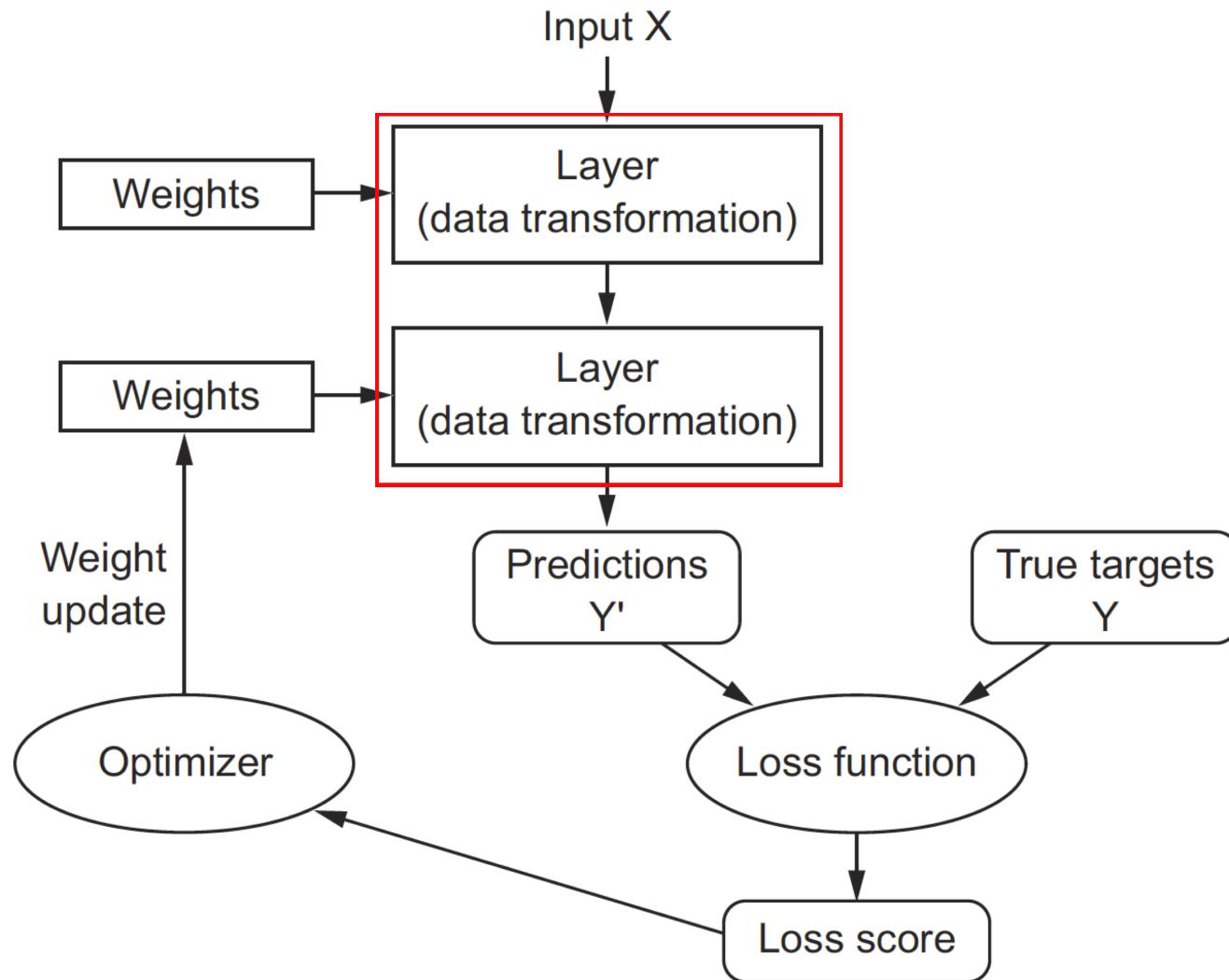
Normalization

Cross validation

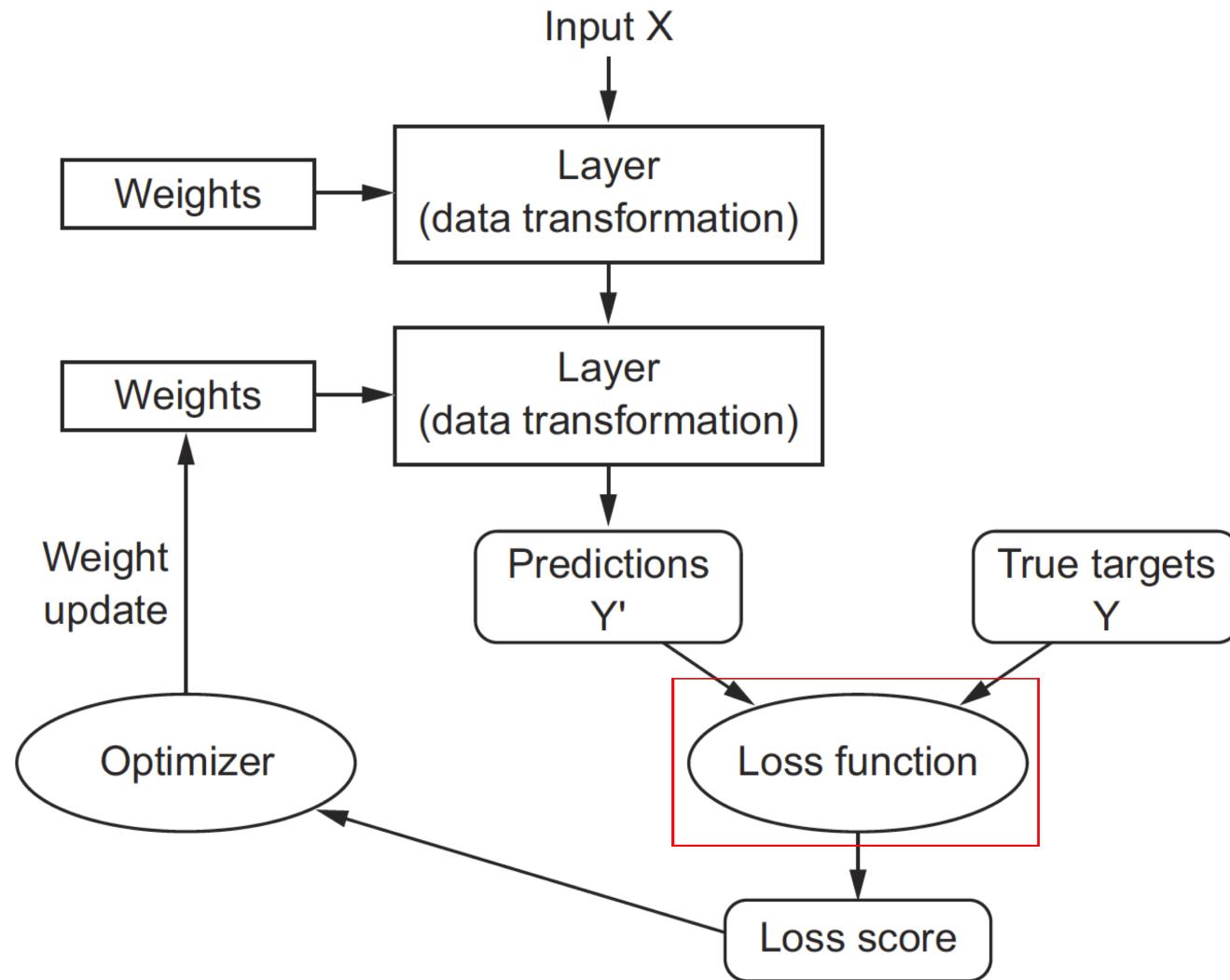
Relationship between objects of ANN



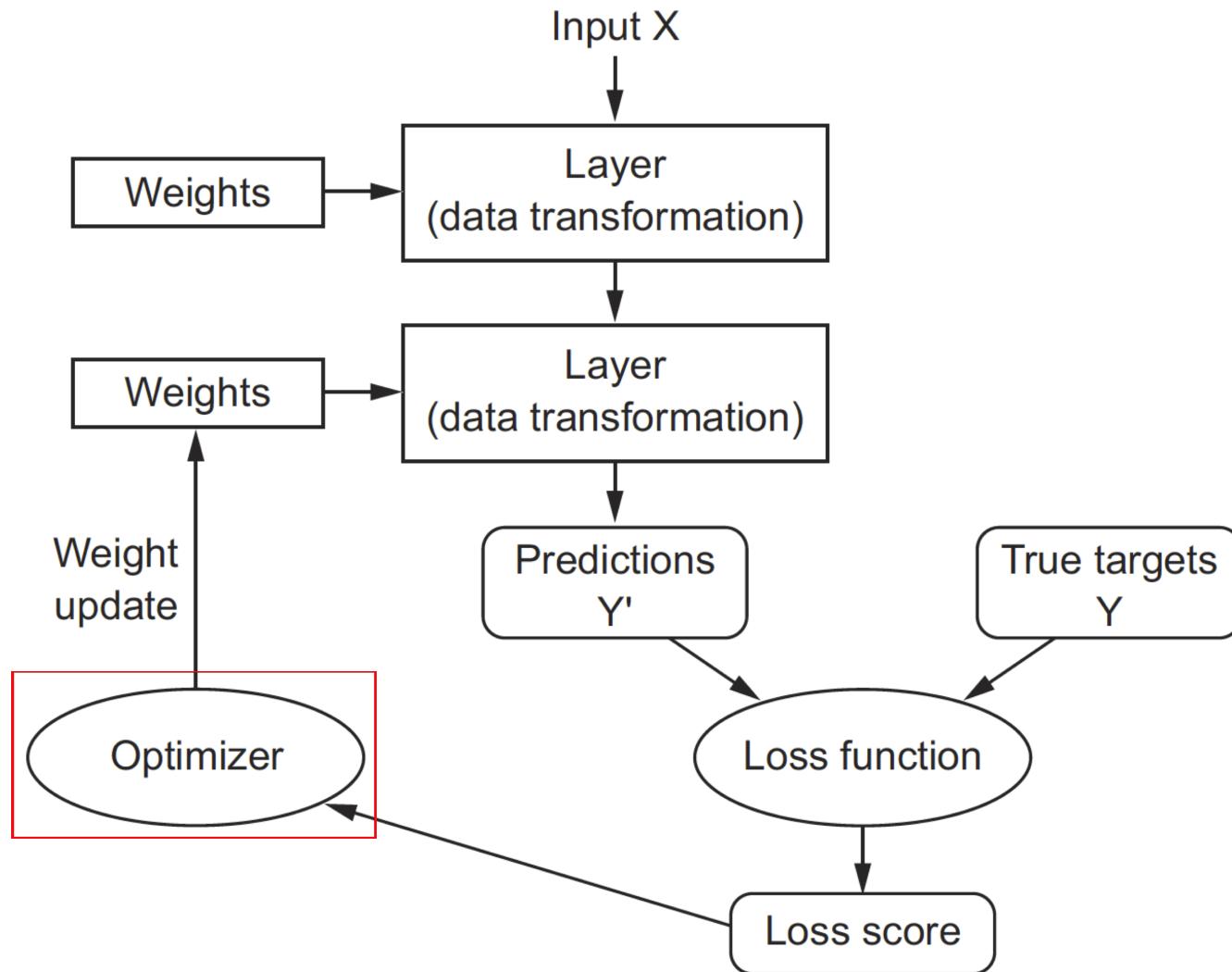
Relationship between objects of ANN



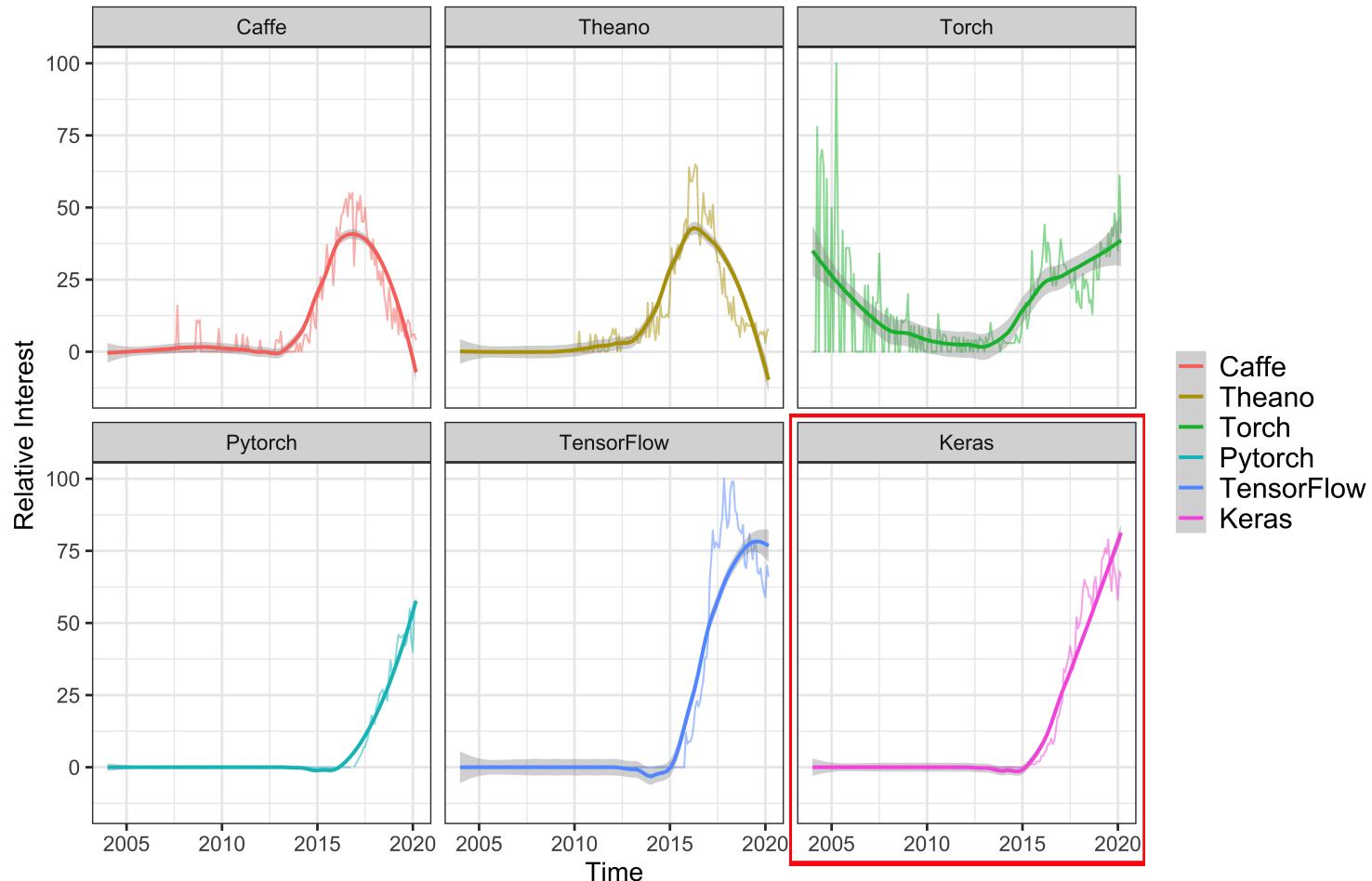
Relationship between objects of ANN



Relationship between objects of ANN



Deep learning frameworks



Introduction to Keras (κέρας)

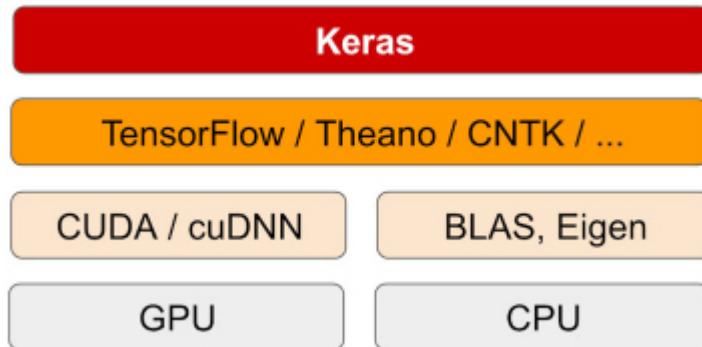
Keras is a **deep-learning framework** that provides a convenient way to **define** and **train** almost any kind of **deep-learning model**.

Features of Keras

1. Allows the same code to run seamlessly on CPU or GPU
2. User-friendly API
3. Built-in support for convolutional networks, recurrent networks, and any combination of both
4. It supports arbitrary network architectures: multi-input or multi-output models, layer sharing, model sharing, and so on.
5. Keras makes it easy to turn models into products
6. Keras development is backed by key companies in the deep learning:

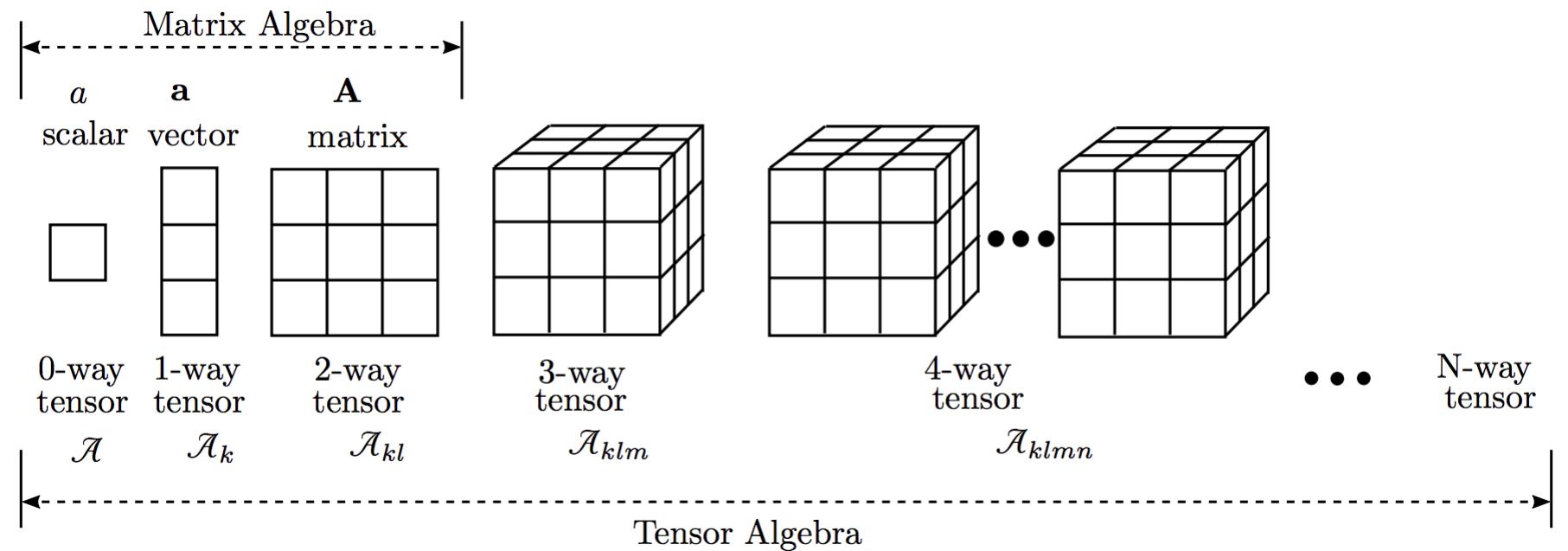


The deep-learning software and hardware stack



Keras supports **multiple backend engines** and does not lock you into one ecosystem.

Tensors



Tensors as generalizations of scalars, vectors and matrices

.... contd.

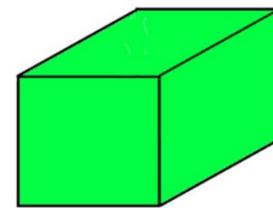
1 D TENSOR /
VECTOR

5
7
4 5
1 2
- 6
3
2 2
1
6
3
- 9

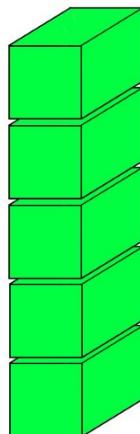
2 D TENSOR /
MATRIX

- 9	4	2	5	7
3	0	1 2	8	6 1
1	2 3	- 6	4 5	2
2 2	3	- 1	7 2	6

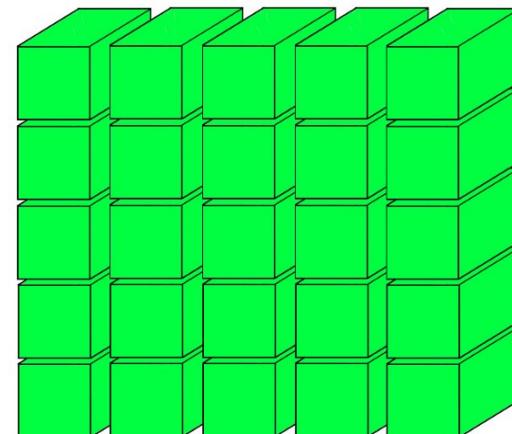
3 D TENSOR /
CUBE



- 9	4	2	5	7
3	0	1 2	8	6 1
1	2 3	- 6	4 5	2
2 2	3	- 1	7 2	6



4 D TENSOR
VECTOR OF CUBES



5 D TENSOR
MATRIX OF CUBES

Binary classification



1 (cat) vs 0 (non cat)

Binary classification



1 (cat) vs 0 (non cat)

Binary classification



1 (cat) vs 0 (non cat)

Binary classification



1 (cat) vs 0 (non cat)

y

Binary classification



1 (cat) vs 0 (non cat)

y

→ Red	→ Green	→ Blue				
			255 134 93 22			
			255 134 202 22	2		
255	231	42	22	4	30	
123	94	83	2	192	124	
34	44	187	92	34	142	
34	76	232	124	94		
67	83	194	202			

Binary classification



64



1 (cat) vs 0 (non cat)

y

64

→ Red	→ Green	→ Blue				
255	134	93	22			
255	134	202	22	2		
255	231	42	22	4	30	
123	94	83	2	192	124	
34	44	187	92	34	142	
34	76	232	124	94		
67	83	194	202			

64

64

Binary classification



1 (cat) vs 0 (non cat)

y

64

→ Red → Green → Blue

	255	134	93	22
64	255	134	202	22
	255	231	42	22
	123	94	83	2
64	34	44	187	92
	34	76	232	124
	67	83	194	202
64				

64

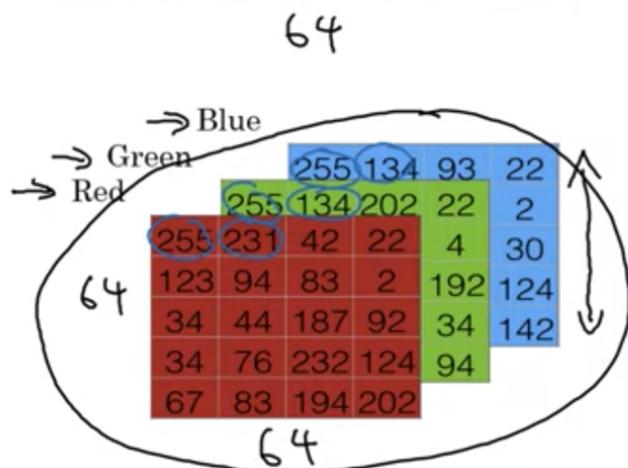
$$X = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ 255 \\ 134 \\ \vdots \end{bmatrix}$$

Binary classification



1 (cat) vs 0 (non cat)

y



$$X = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ \vdots \\ 255 \\ 134 \\ \vdots \\ \vdots \end{bmatrix}$$

$$64 \times 64 \times 3 = 12288$$

Binary classification



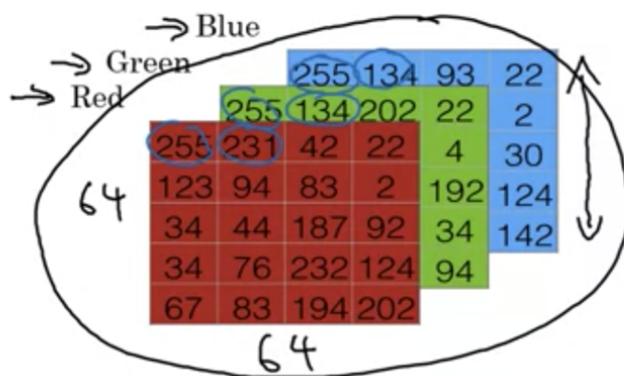
64



1 (cat) vs 0 (non cat)

y

64



$$X = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ \vdots \\ 255 \\ 134 \\ \vdots \end{bmatrix}$$

$$64 \times 64 \times 3 = 12288$$

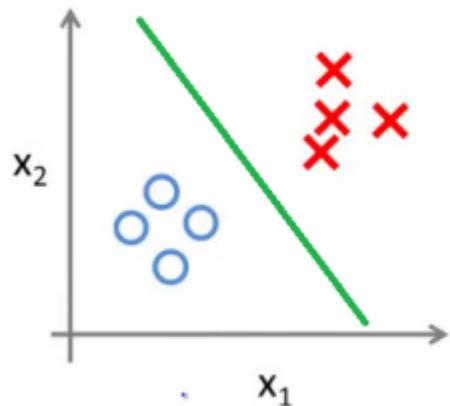
$$n = n_x = 12288$$

$$X \longrightarrow y$$

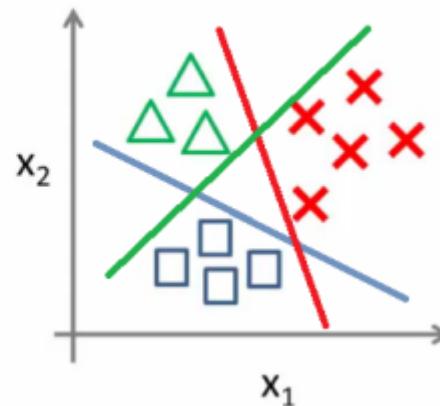
Multiclass classification

Multiclass problems involve classifying something into one of N classes (e.g. “red”, “white” or “blue”, etc).

Binary classification:



Multi-class classification:



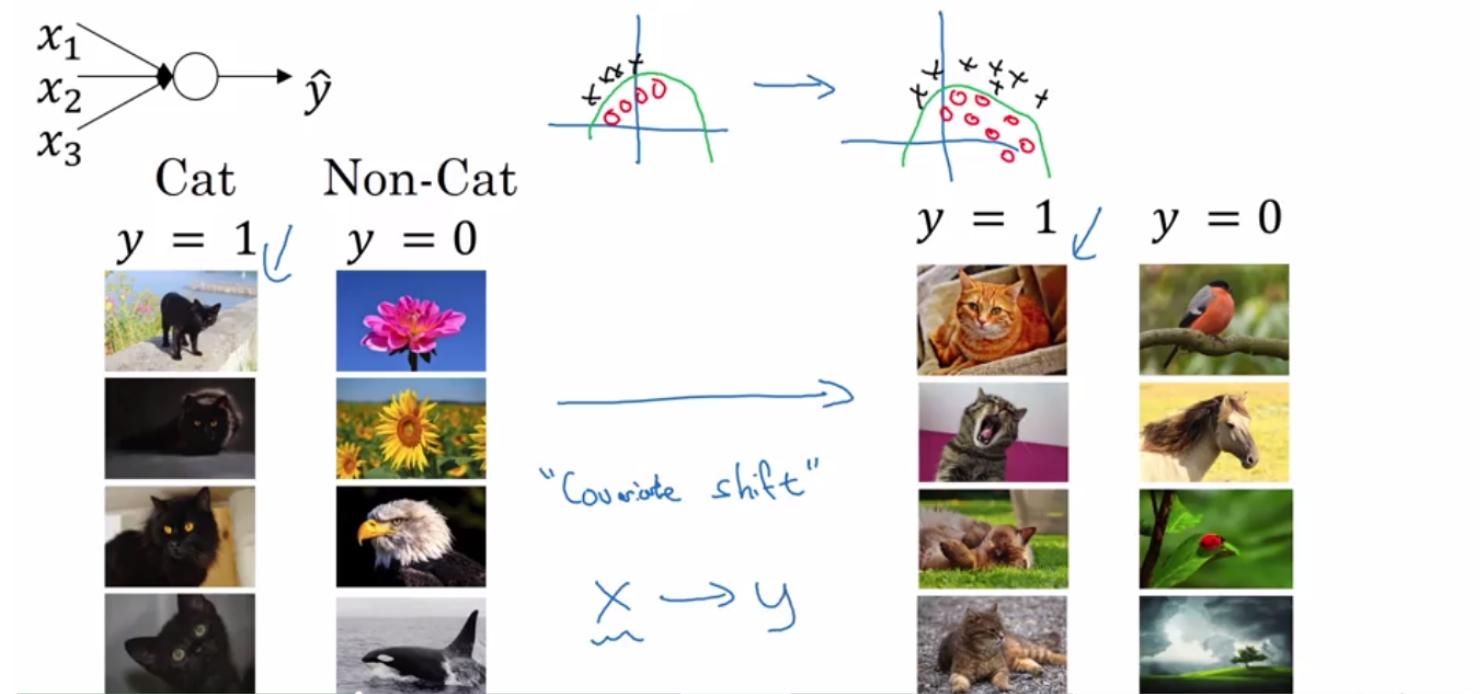
Normalization methods in Deep Learning

Problem 1: Data Distribution



we can observe that the range of income is larger than the range of age. If we do not do some preprocessing methods on these data, the effect of income feature may influence more than the one of age.

Problem 2: Internal Covariate Shift



For a cat classifier, we may face a problem with the diverse data distribution of different batches. Even though there is a function which can split the data point, it is hard for the model to distinguish the data correctly in one mini-batch. We call this phenomenon “Internal Covariate Shift”.

Problem 3: Vanishing/ Exploding Gradients

$$\frac{\partial \mathbf{h}^{(T)}}{\partial \mathbf{h}^{(1)}} = \frac{\partial \mathbf{h}^{(T)}}{\partial \mathbf{h}^{(T-1)}} \cdots \frac{\partial \mathbf{h}^{(2)}}{\partial \mathbf{h}^{(1)}}.$$

For a very deep neural network, we will multiply several gradients. If any of them is too small or too large, the neural network will face a vanishing/exploding gradients problem. Normalization can stabilize each value to reduce the vanishing/exploding gradient problem.

Batch Normalization

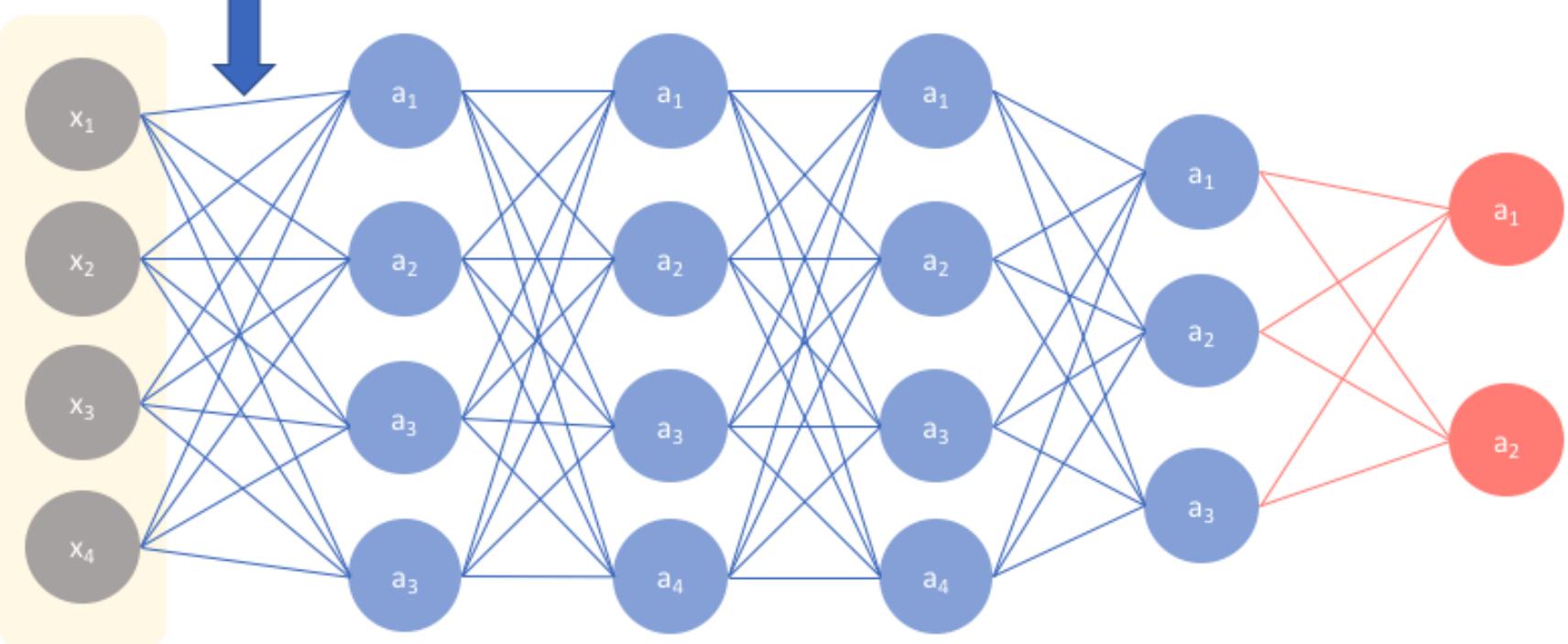
Input layer

Normalizing inputs improves loss function topology primarily in these dimensions



Hidden layers

Output layer



(1)

(2)

(3)

(4)

(5)

(6)

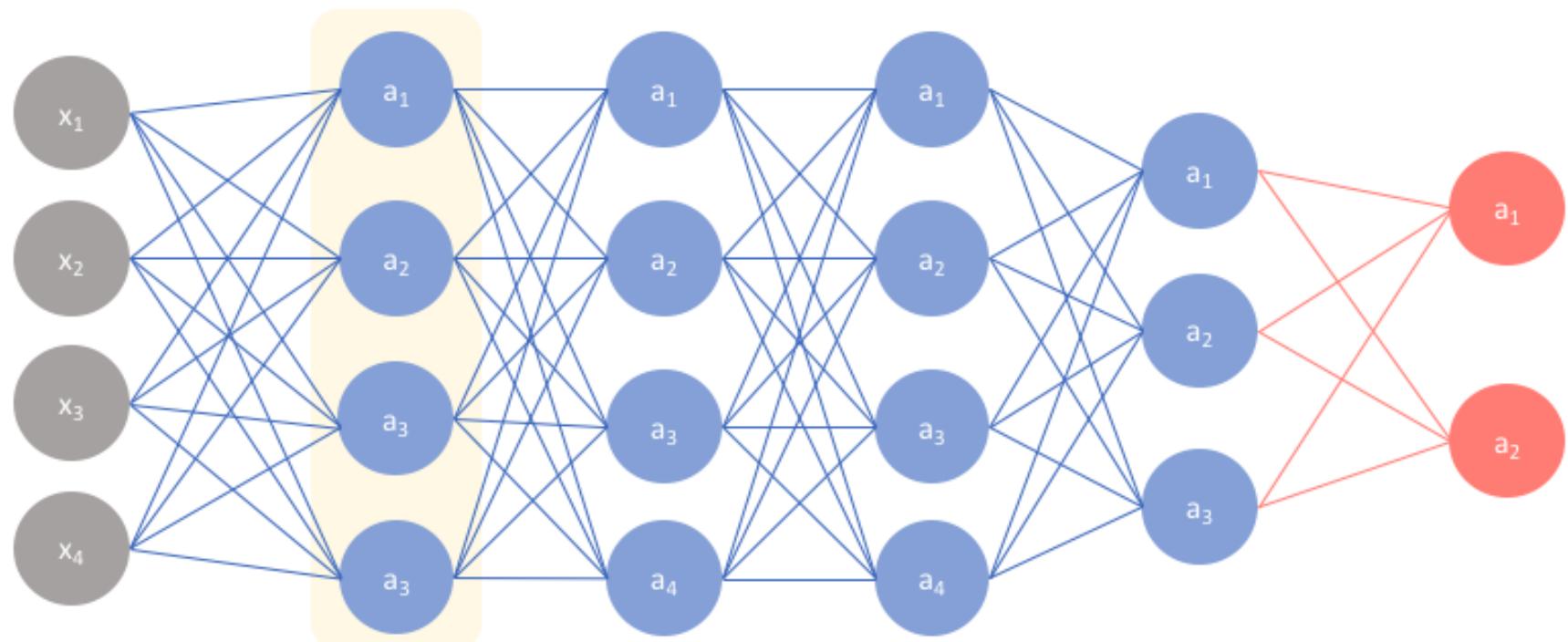
Batch Normalization

Input layer

Hidden layers

Output layer

These activations are essentially
the inputs to the following layer, so
why not normalize these values?



When not to Batch Normalize?

1. Small batch size
2. Recurrent connections in an RNN

What are the Alternatives?

1. Weight Normalization
2. Layer Normalization
3. Instance Normalization
4. Group Normalization
5. Batch Renormalization
6. Batch-Instance Normalization
7. Switchable Normalization
8. Spectral Normalization
9. ScaleNorm

Train, validation and test sets



Training Dataset: The sample of data used to fit the model.

Validation Dataset: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.

Test Dataset: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

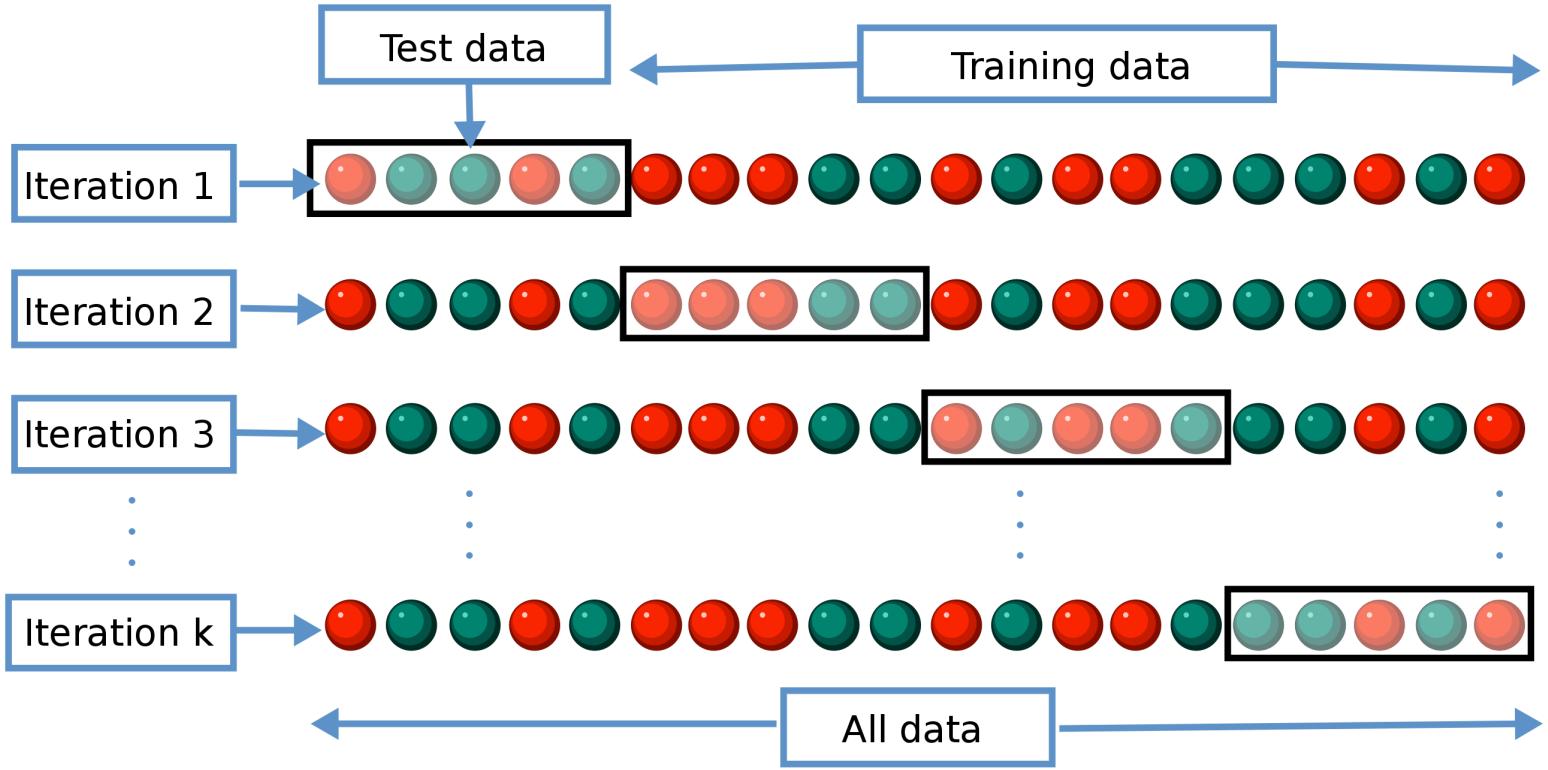
Dataset split ratio

Depends on 2 things:

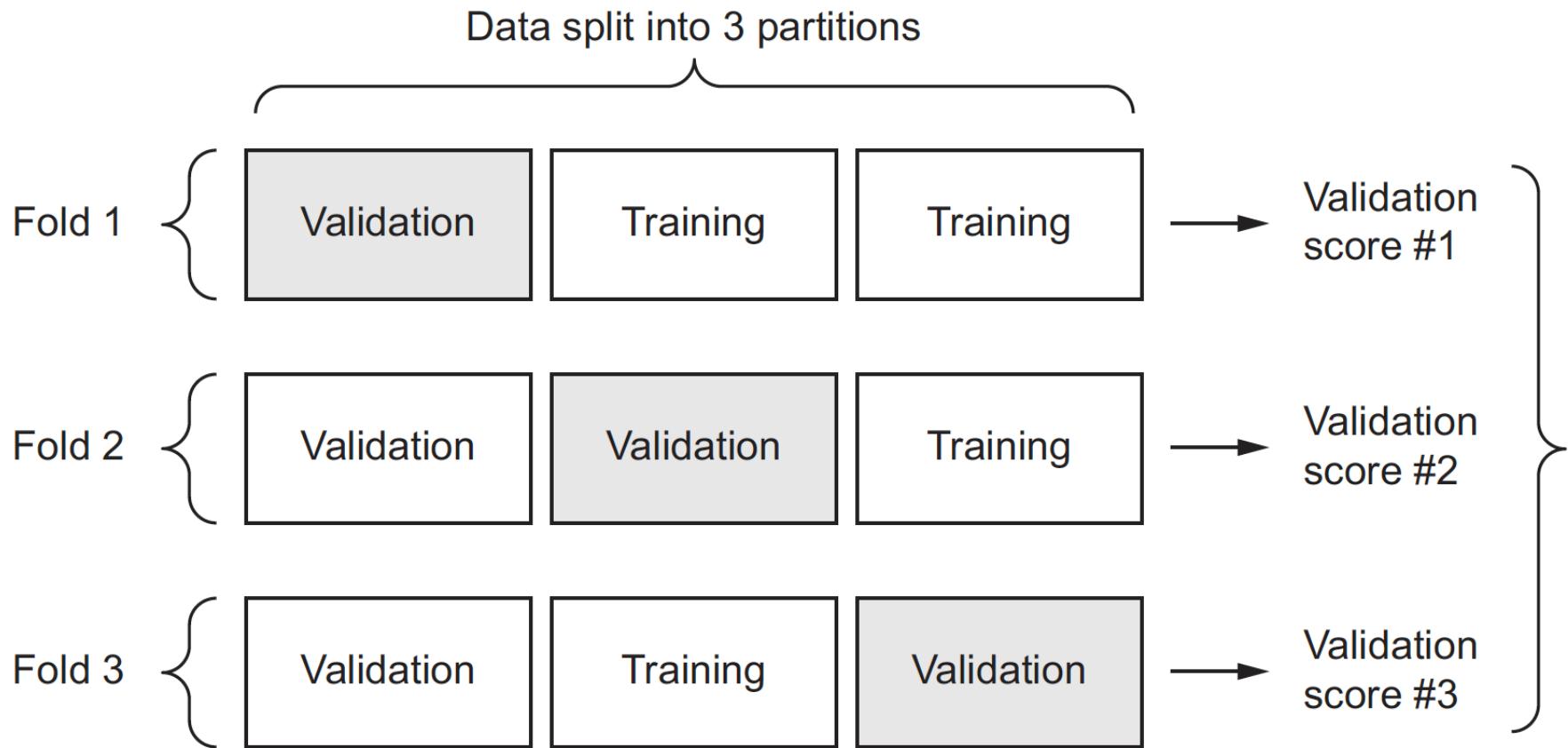
1. The total number of samples in data
2. The actual model you are training

- Models with very few hyperparameters will be easy to validate and tune, so you can probably reduce the size of your validation set
- If the model has many hyperparameters, you would want to have a large validation set as well (although you should also consider cross validation).
- If the model have no hyperparameters or ones that cannot be easily tuned, probably don't need a validation set too!

Cross validation



3-fold cross-validation



Thank you!