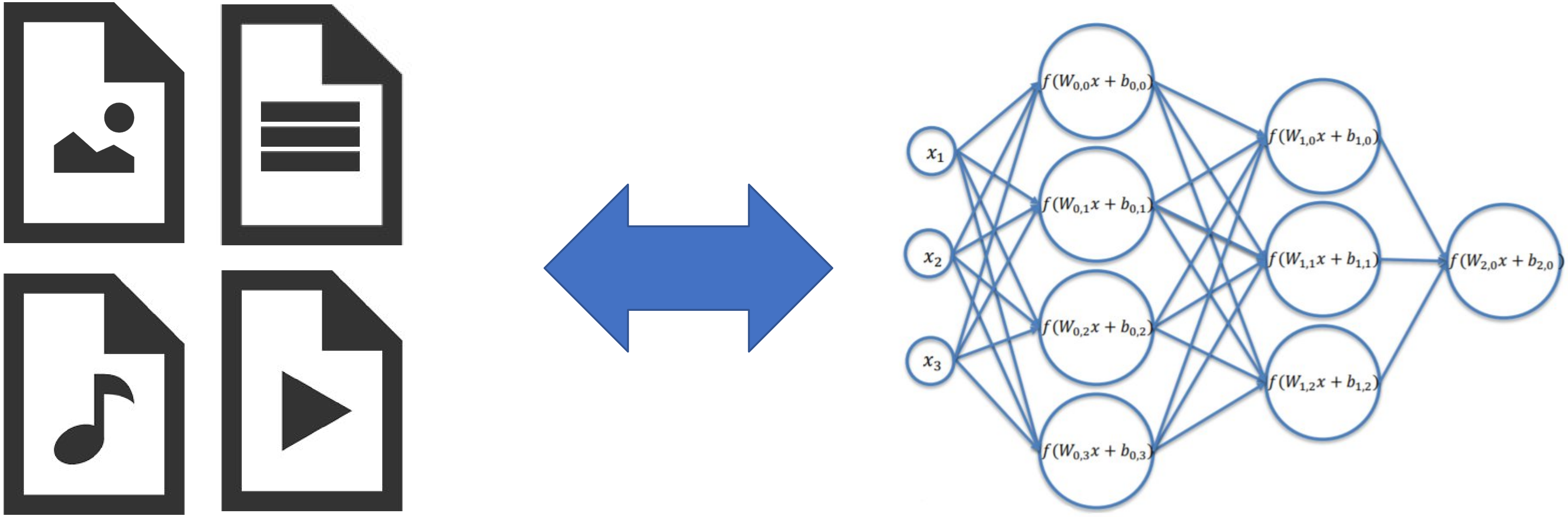
Hands-on Introduction to Deep Learning

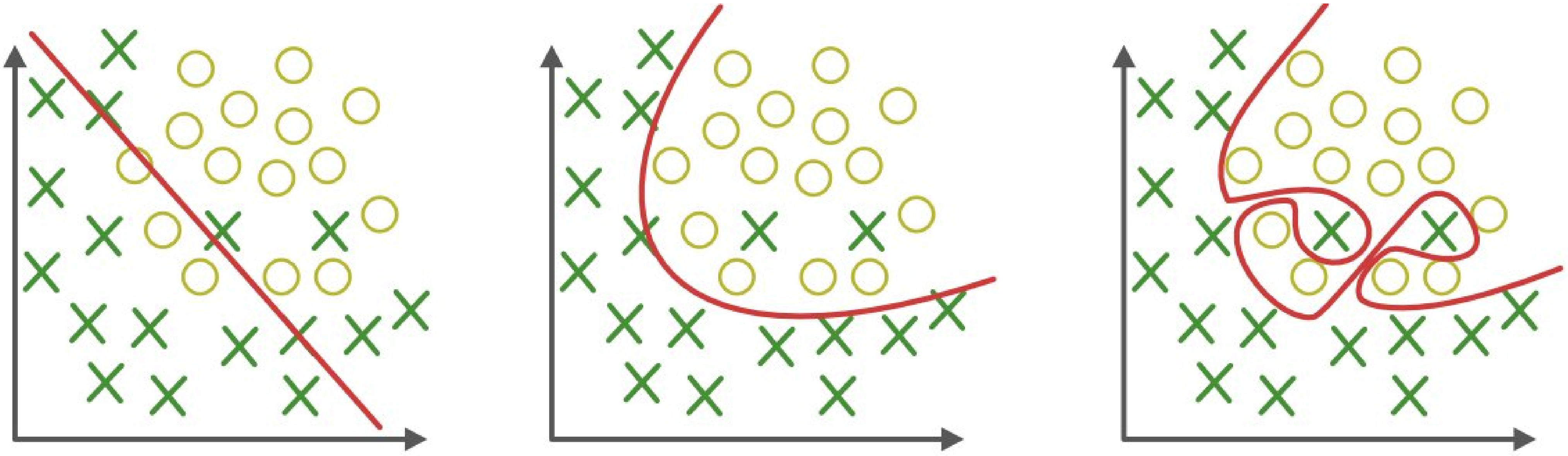
Methodology

cnrs

Institut du Développement et des Ressources en Informatique Scientifique

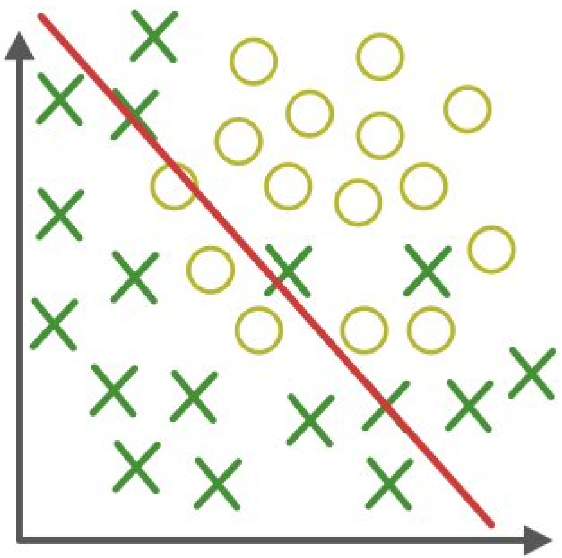


Working basis

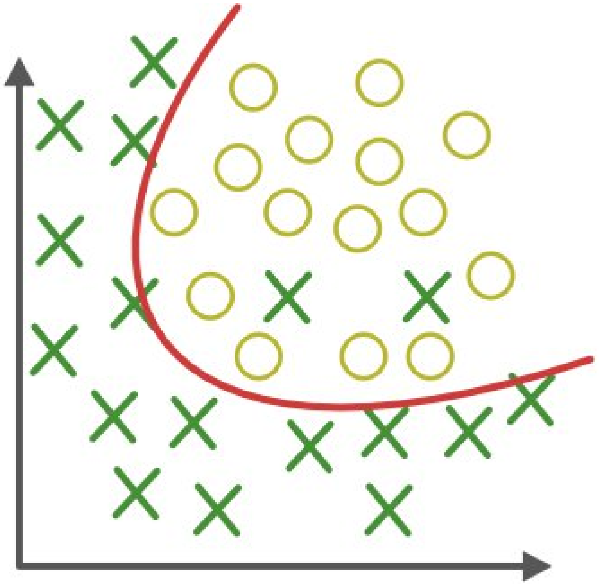


Underfitting Proper learning Overfitting

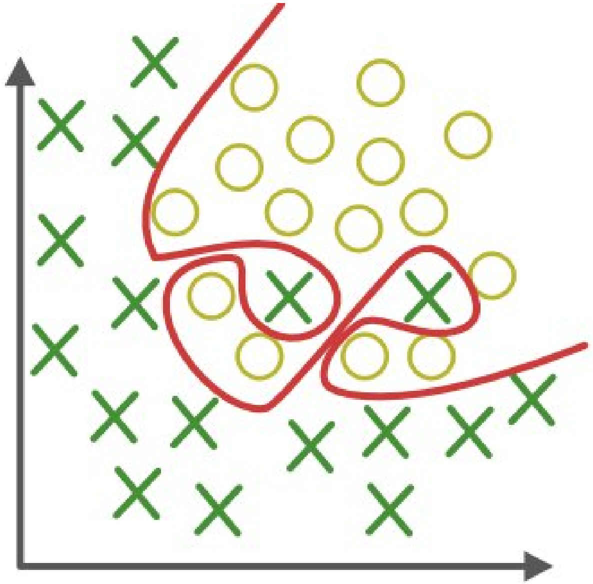
Bias variance trade-off



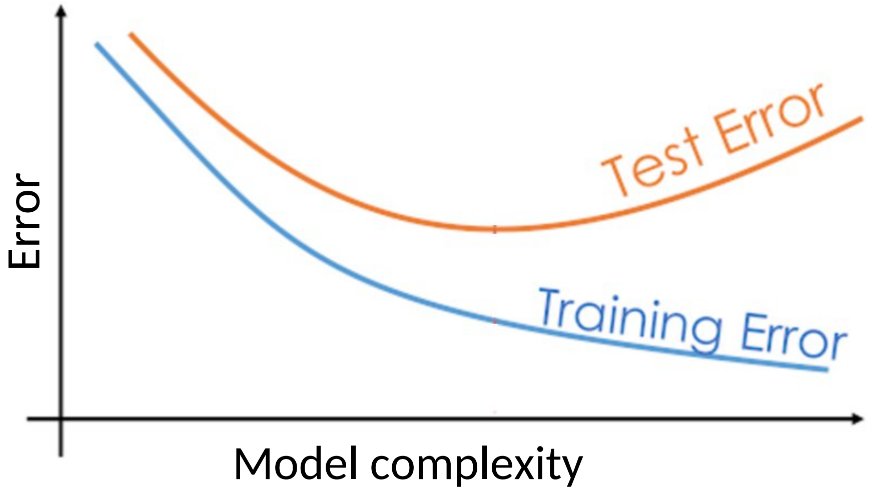
Underfitting



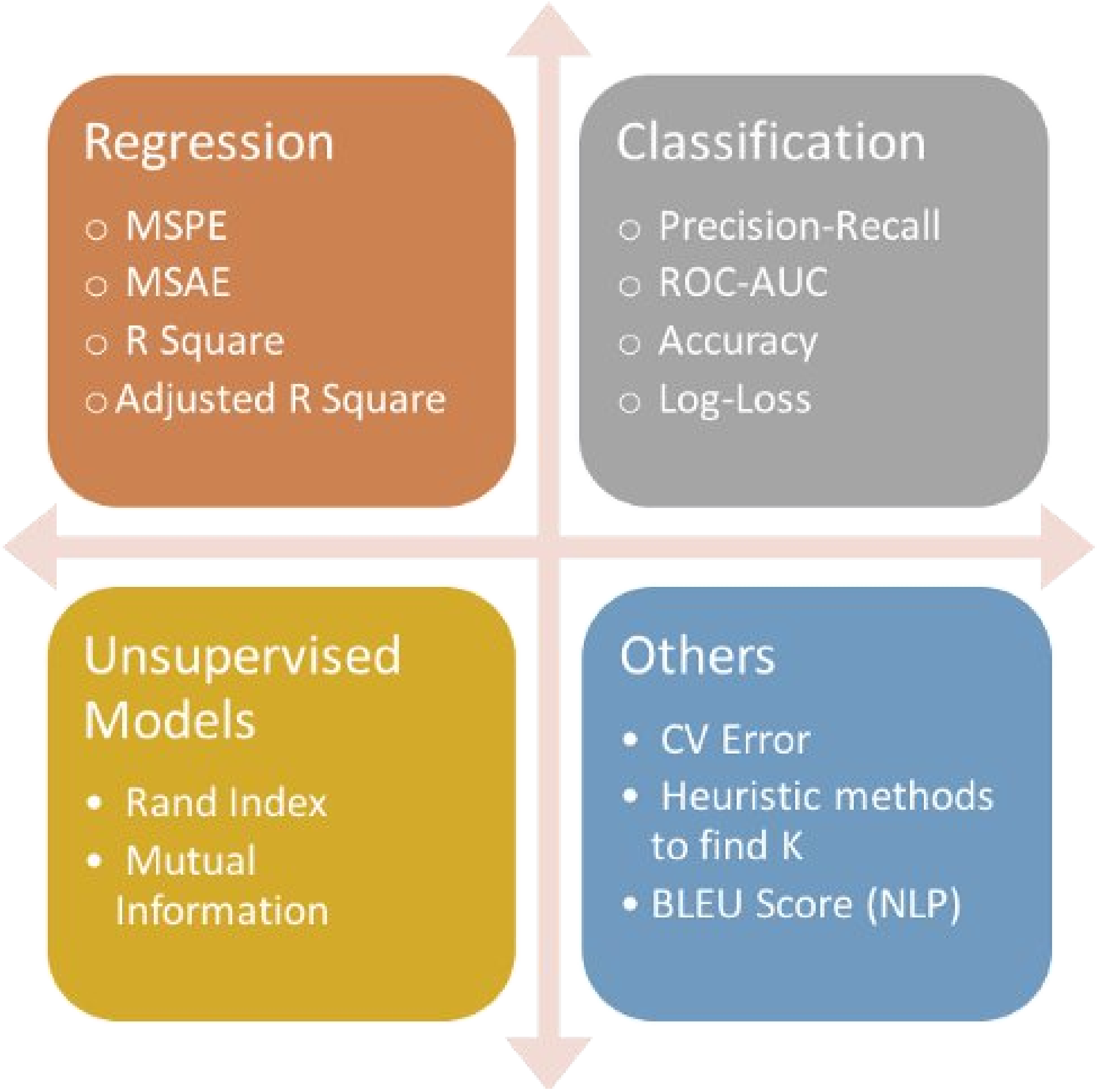
Proper learning



Overfitting

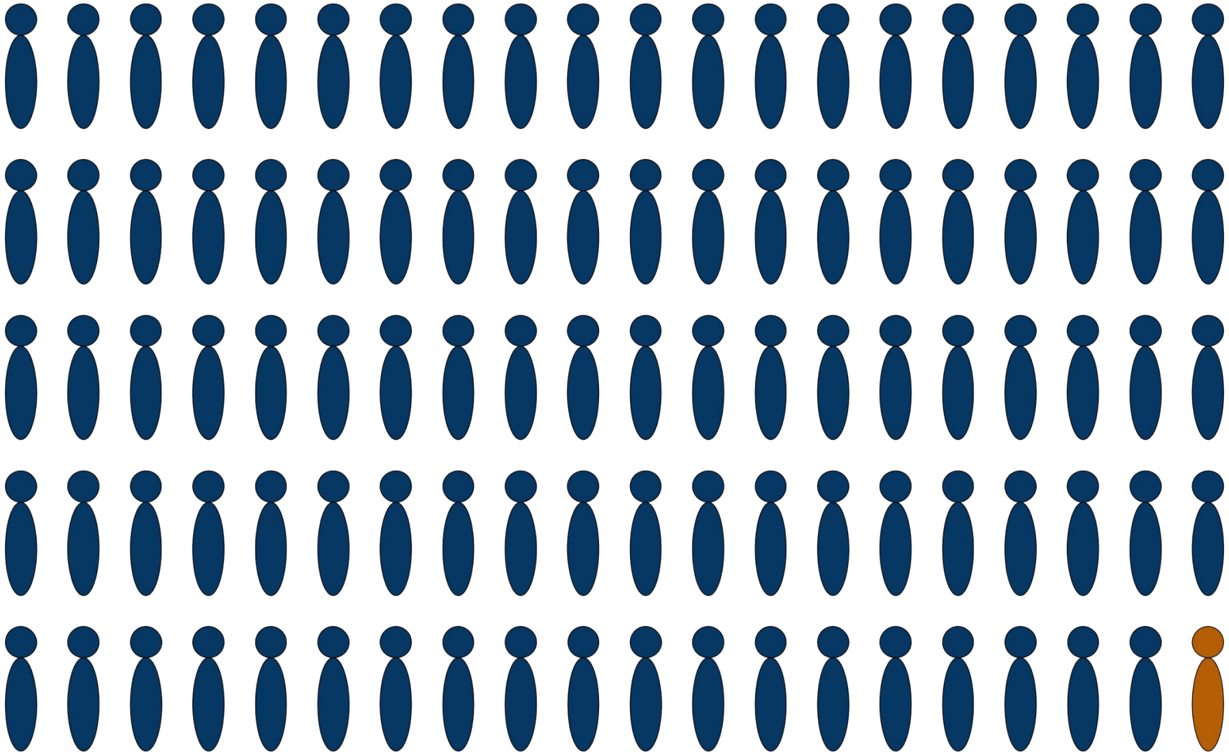


Bias variance trade-off - Detection


Regression
MSPE
MSAE
R Square
Adjusted R Square
Classification
Precision-Recall
ROC-AUC
Accuracy
Log-Loss
Unsupervised Models
Rand Index
Mutual Information
Others
CV Error
Heuristic methods to find K
BLEU Score (NLP)

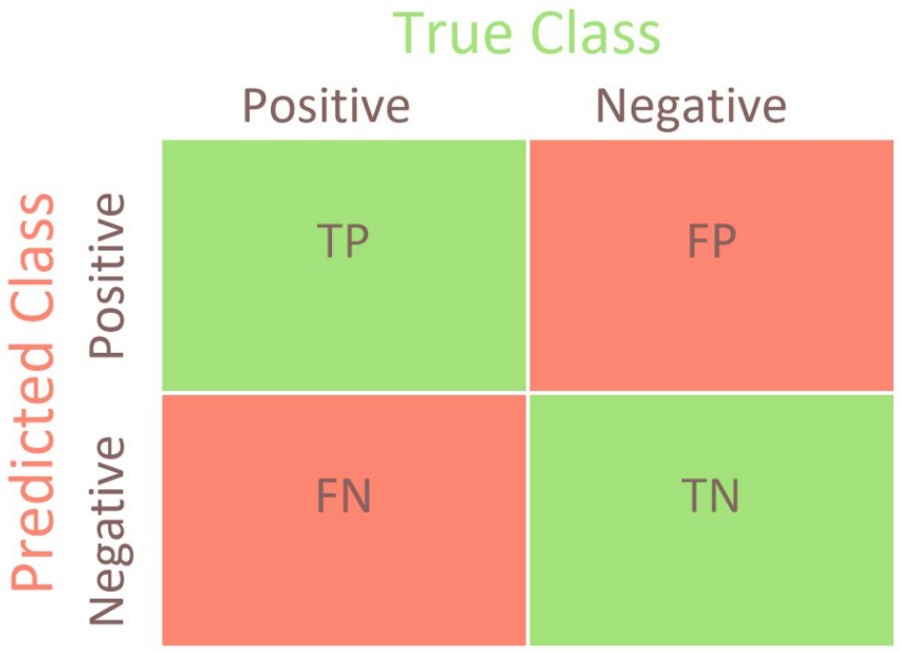
Metrics

New system for illness detection – the accuracy is not a good metric for this case



Metrics – Bad Choice

Metrics – Precision/Recall



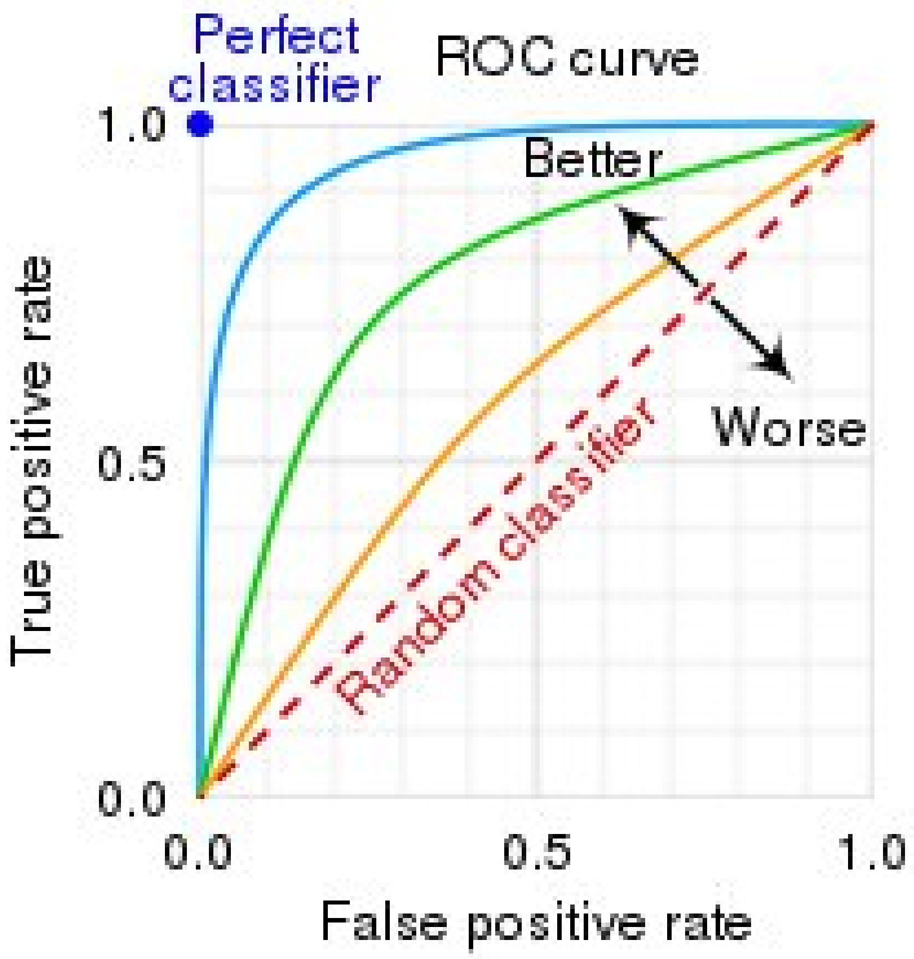
$$precision = \frac{TP}{TP + FP}$$

Above all positive prediction, how many are positive data

$$recall = \frac{TP}{TP + FN}$$

Above all positive data, how many have been predicted positive

Metrics – ROC curve



Variation of the acceptance threshold of a class to obtain the curve

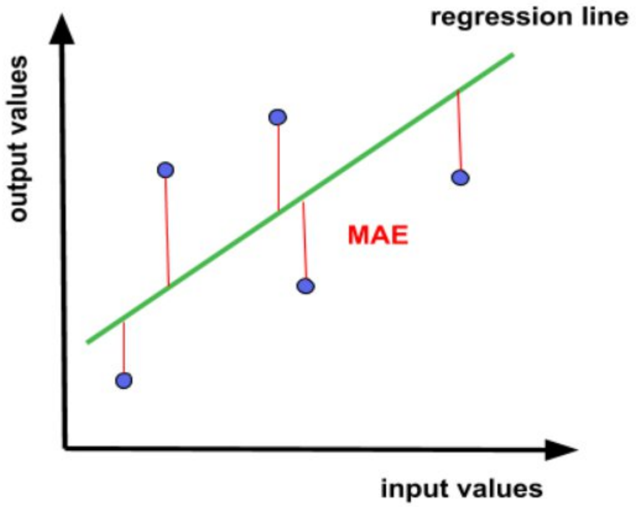
$$TPR = \frac{TP}{TP + FN}$$

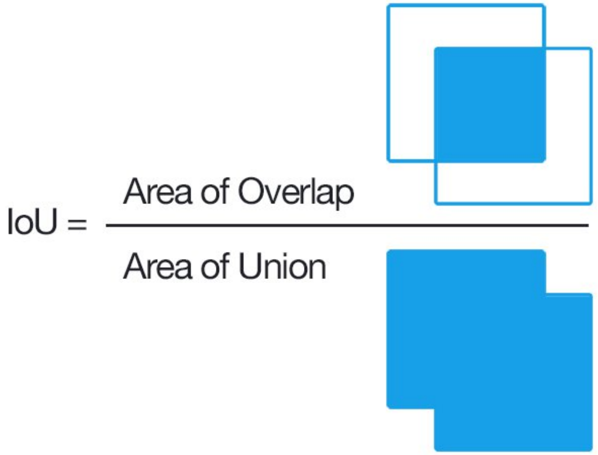
Above all positive data, how many have been predicted positive

$$FPR = \frac{FP}{FP + TN}$$

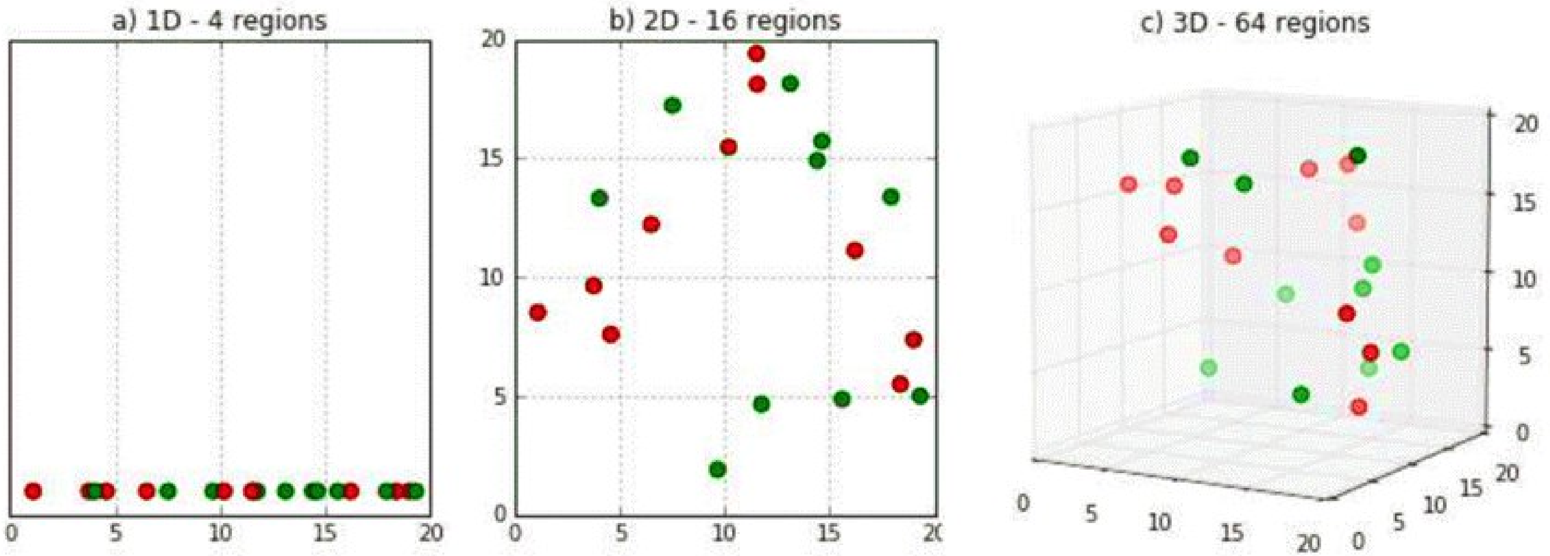
Above all negative data, how many have been predicted positive

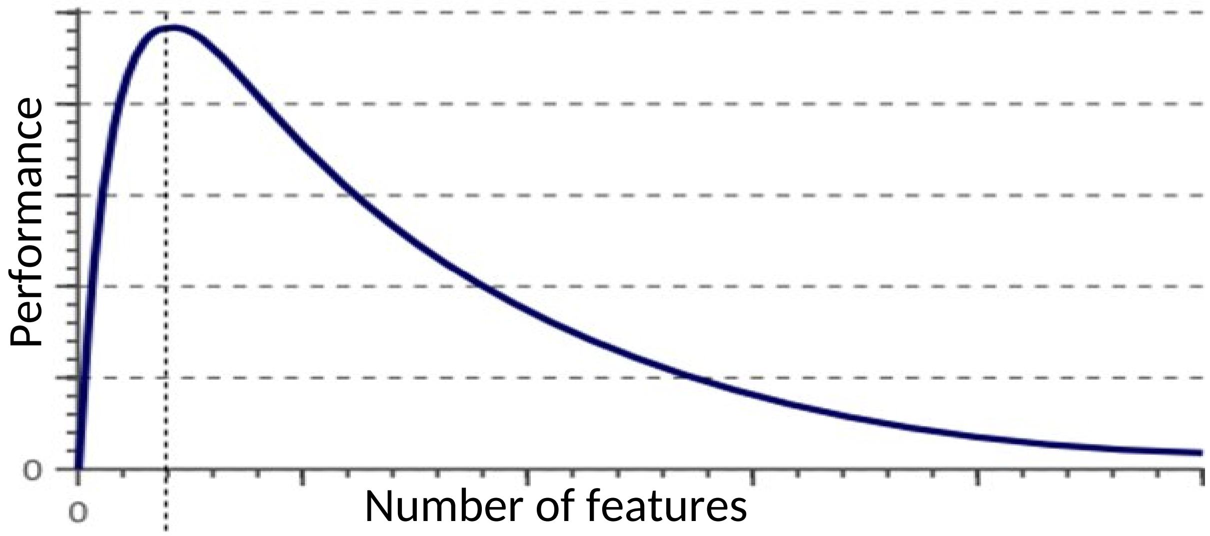
Metrics – Other examples



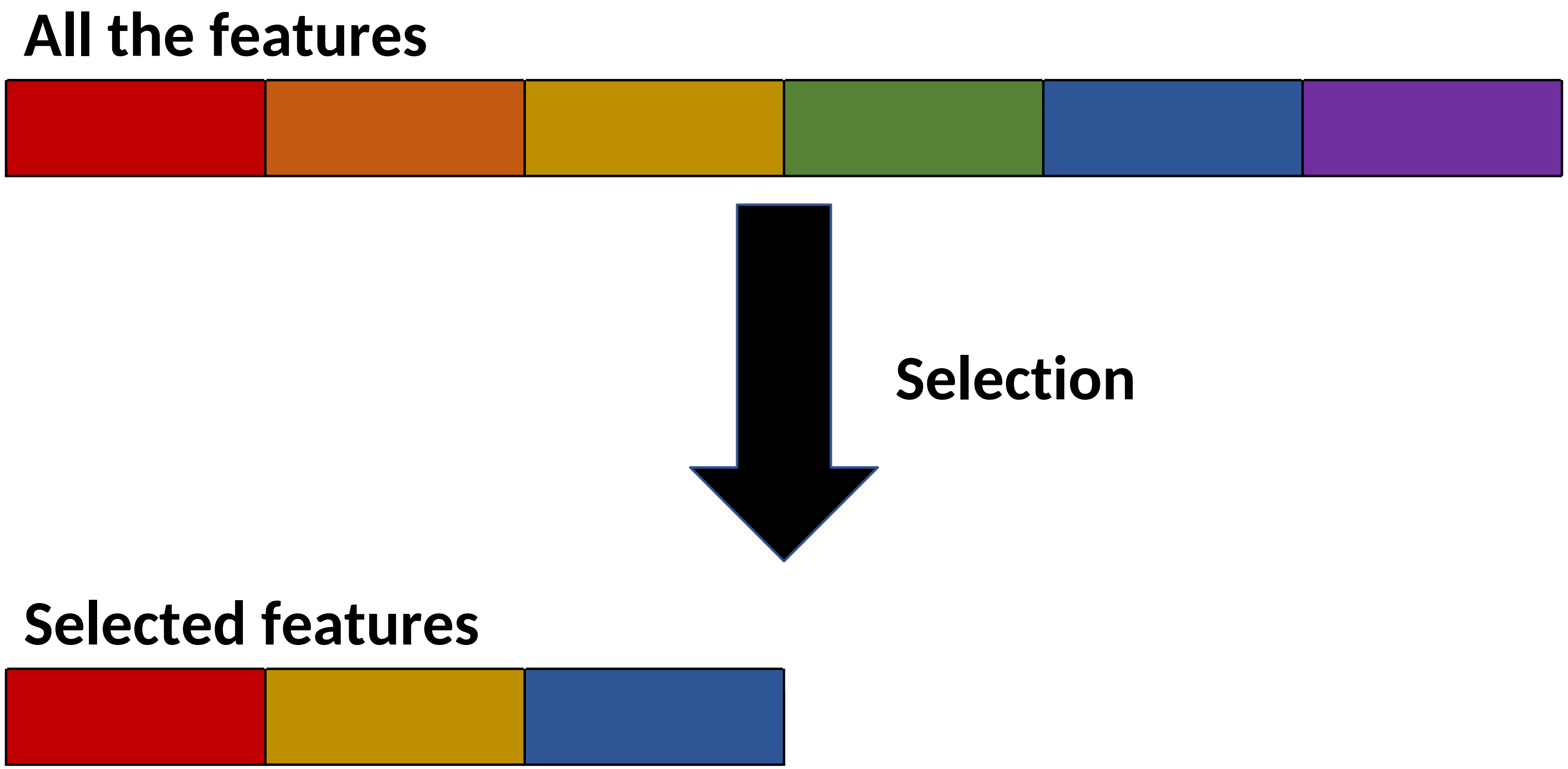


Model	CoLA 8.5k	SST-2 67k	MRPC 3.7k	STS-B 7k	QQP 364k	MNLI-m/mm 393k	QNLI 108k	RTE 2.5k	WNLI 634	AX	**Score**
BiLSTM+ELMo+Attn [1]	36.0	90.4	84.9/77.9	75.1/73.3	64.8/84.7	76.4/76.1	79.8	56.8	65.1	26.5	70.0
Singletask Pretrain Transformer [2]	45.4	91.3	82.3/75.7	82.0/80.0	70.3/88.5	82.1/81.4	87.4	56.0	53.4	29.8	72.8
GPT on STILTs [3]	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.8/80.6	-	69.1	65.1	29.4	76.9
$BERT_{LARGE}$ [4]	60.5	94.9	89.3/85.4	87.6/86.5	72.1/89.3	86.7/85.9	92.7	70.1	65.1	39.6	80.5
MT-DNN [5]	61.5	95.6	90.0/86.7	88.3/87.7	72.4/89.6	86.7/86.0	-	75.5	65.1	40.3	82.2
Snorkel MeTaL [6]	63.8	**96.2**	91.5/88.5	**90.1/89.7**	73.1/89.9	87.6/87.2	93.9	80.9	65.1	39.9	83.2
ALICE *	63.5	95.2	**91.8/89.0**	89.8/88.8	**74.0/90.4**	**87.9/87.4**	95.7	80.9	65.1	40.7	83.3
$MT\text{-}DNN_{KD}$	**65.4**	95.6	91.1/88.2	89.6/89.0	72.7/89.6	87.5/86.7	**96.0**	**85.1**	65.1	**42.8**	**83.7**
Human Performance	66.4	97.8	86.3/80.8	92.7/92.6	59.5/80.4	92.0/92.8	91.2	93.6	95.9	-	87.1



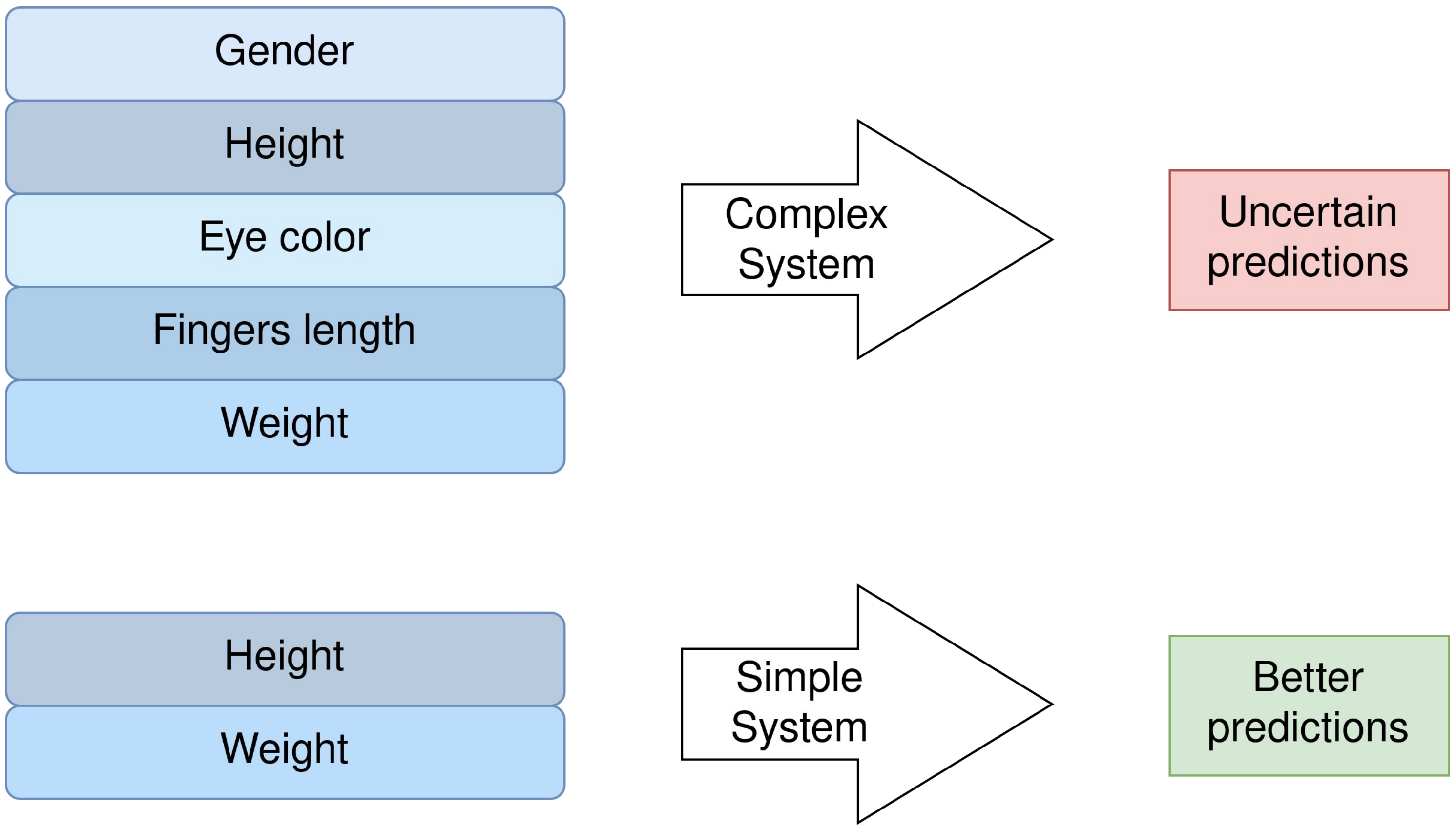


The curse of dimensionality

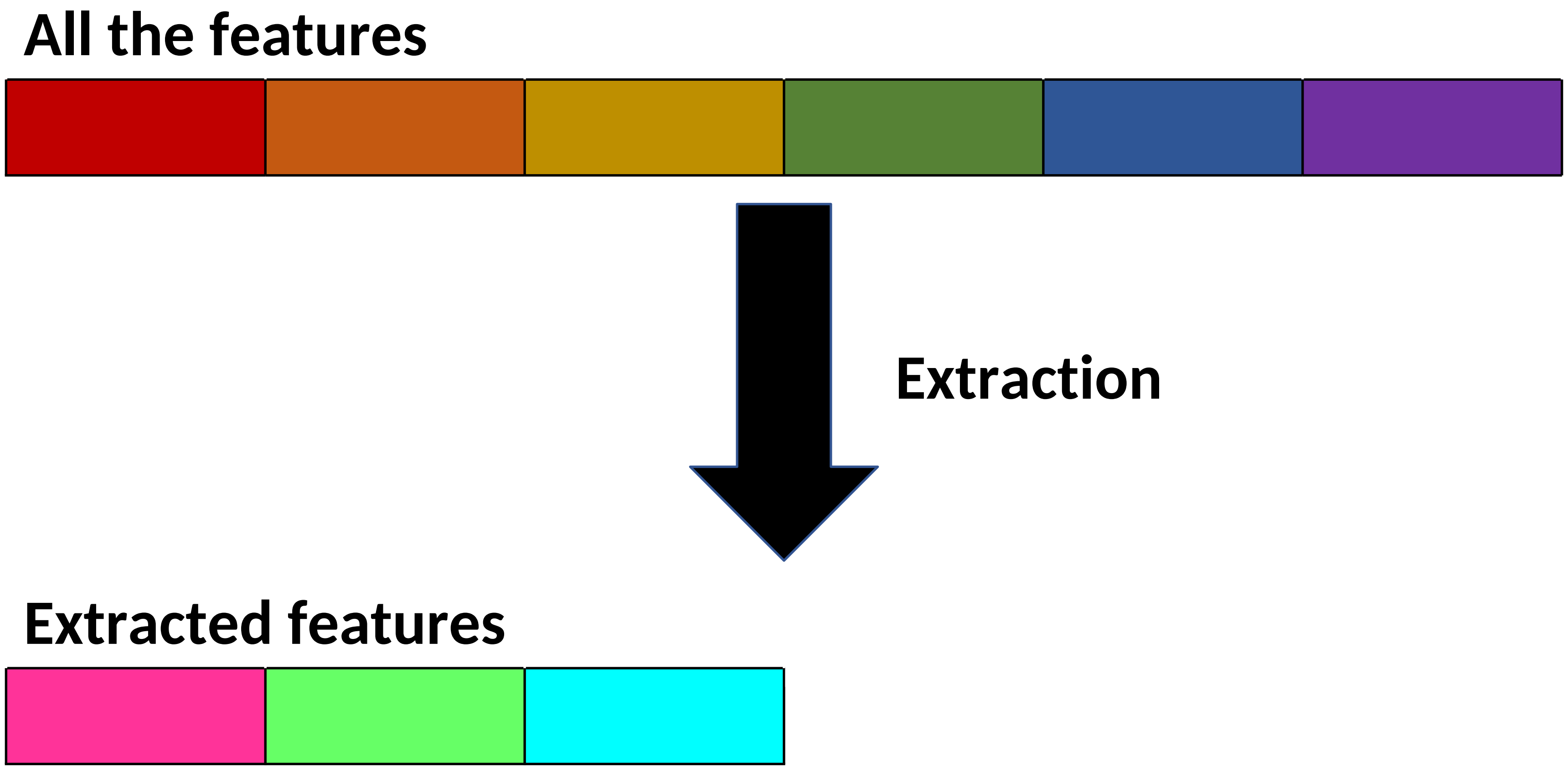


Features selection

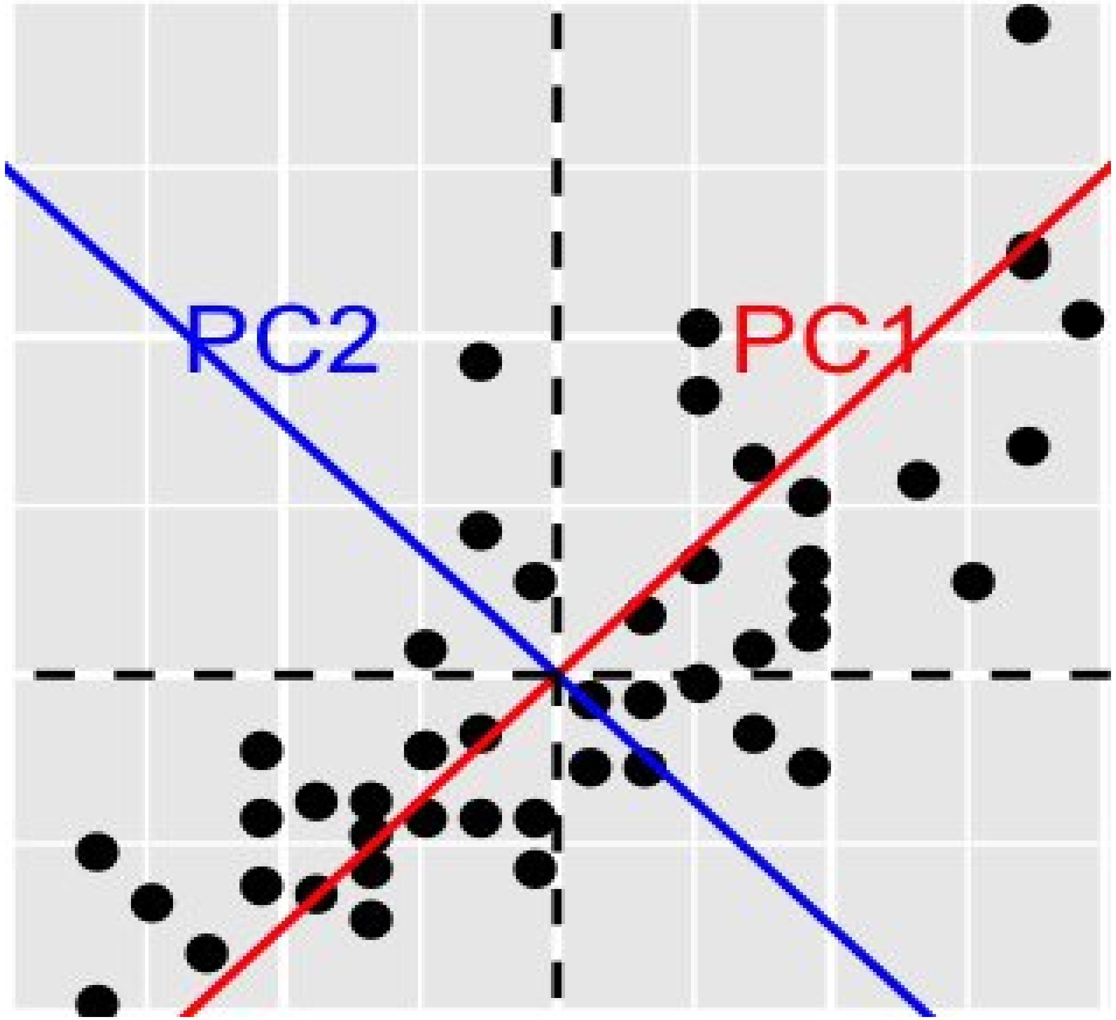
Diabetes risk prediction system



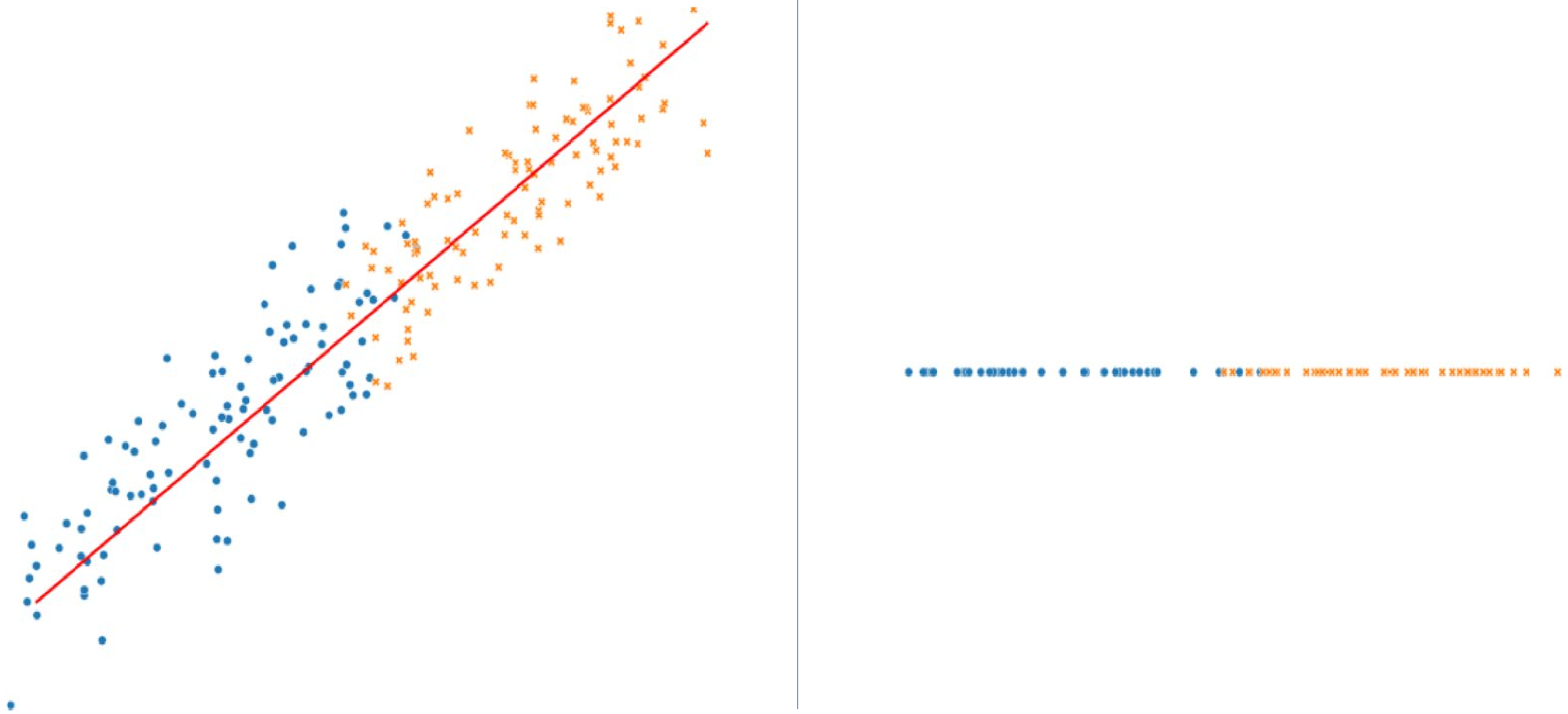
Features selection - Example



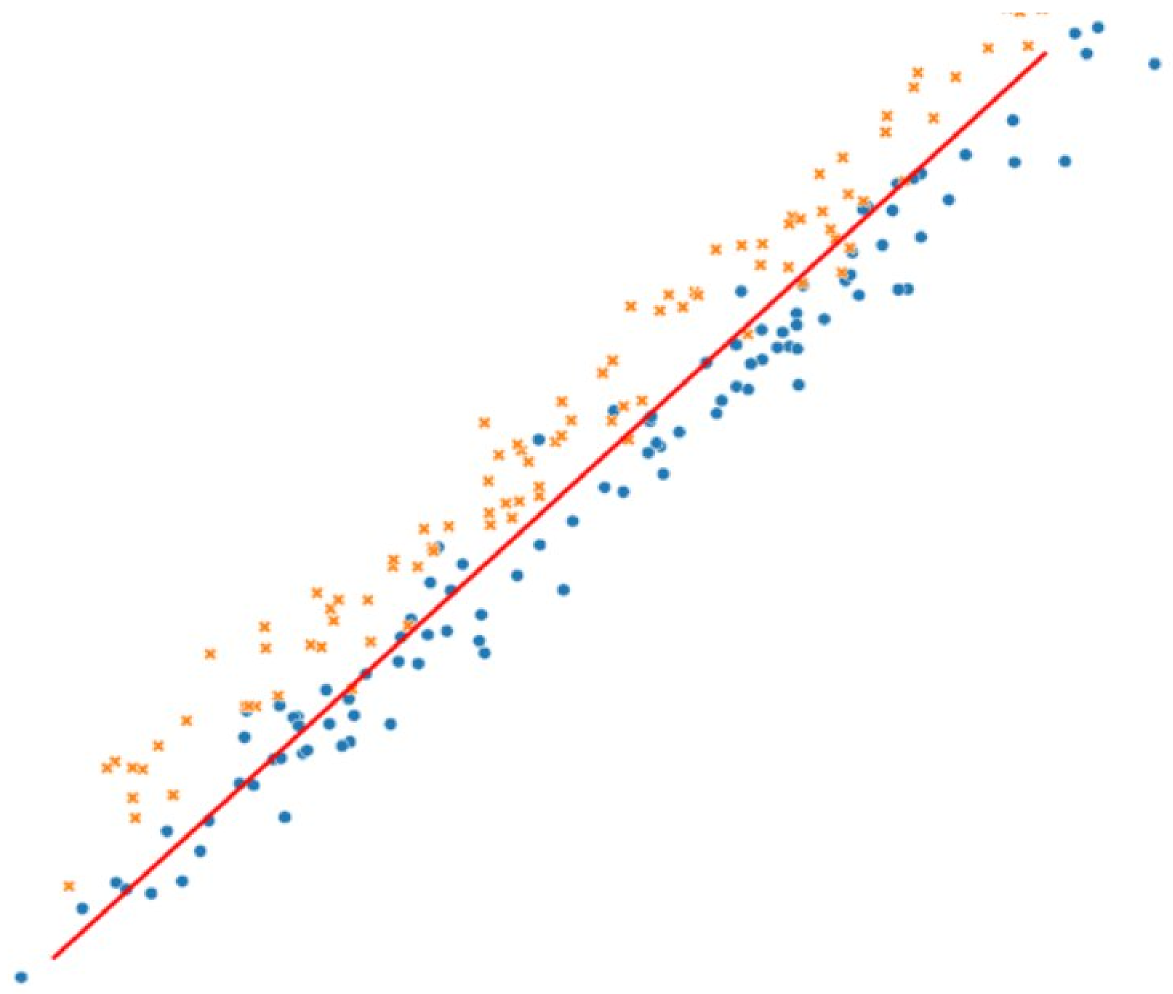
Features extraction



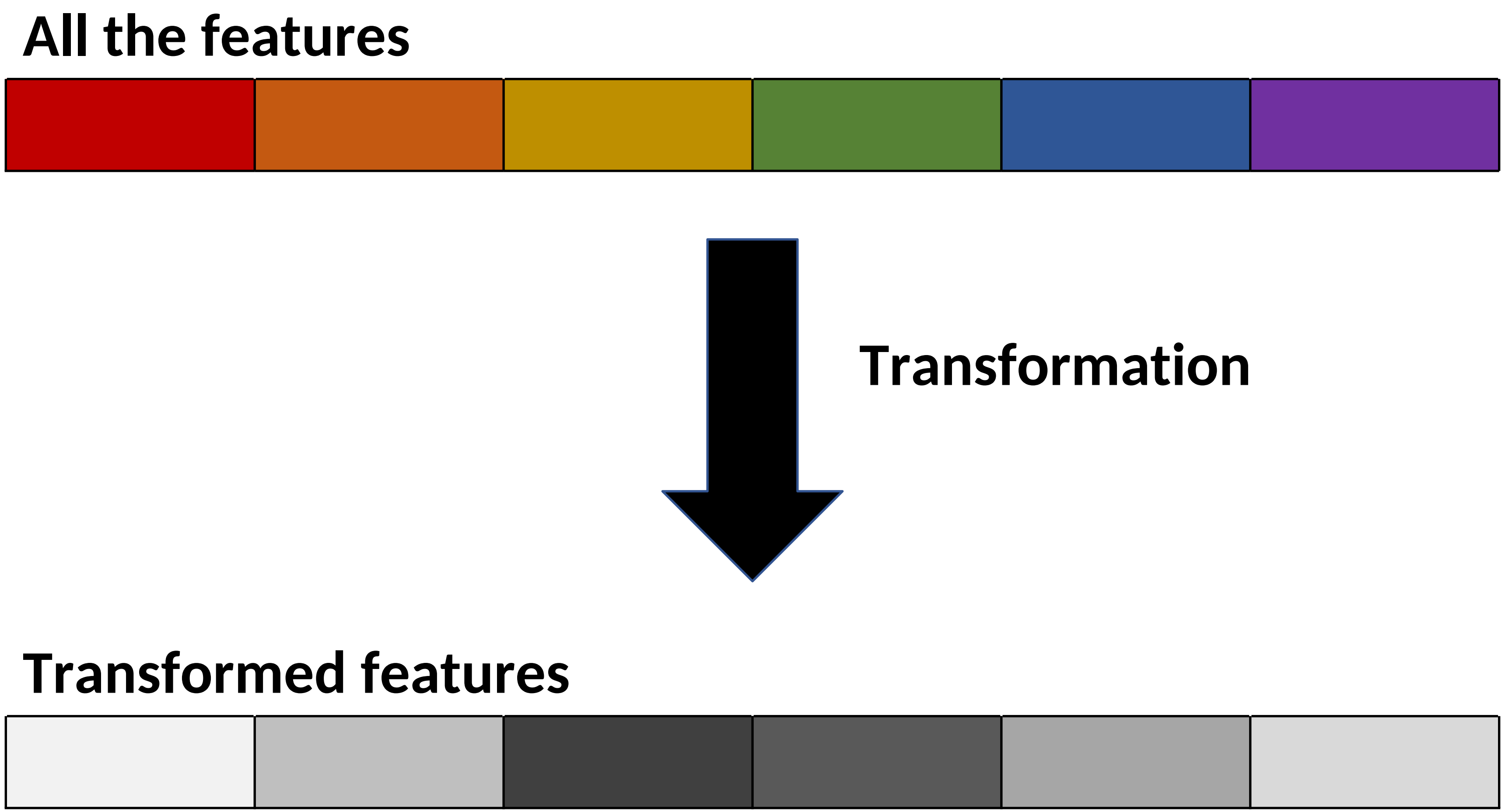
Feature extraction – PCA example



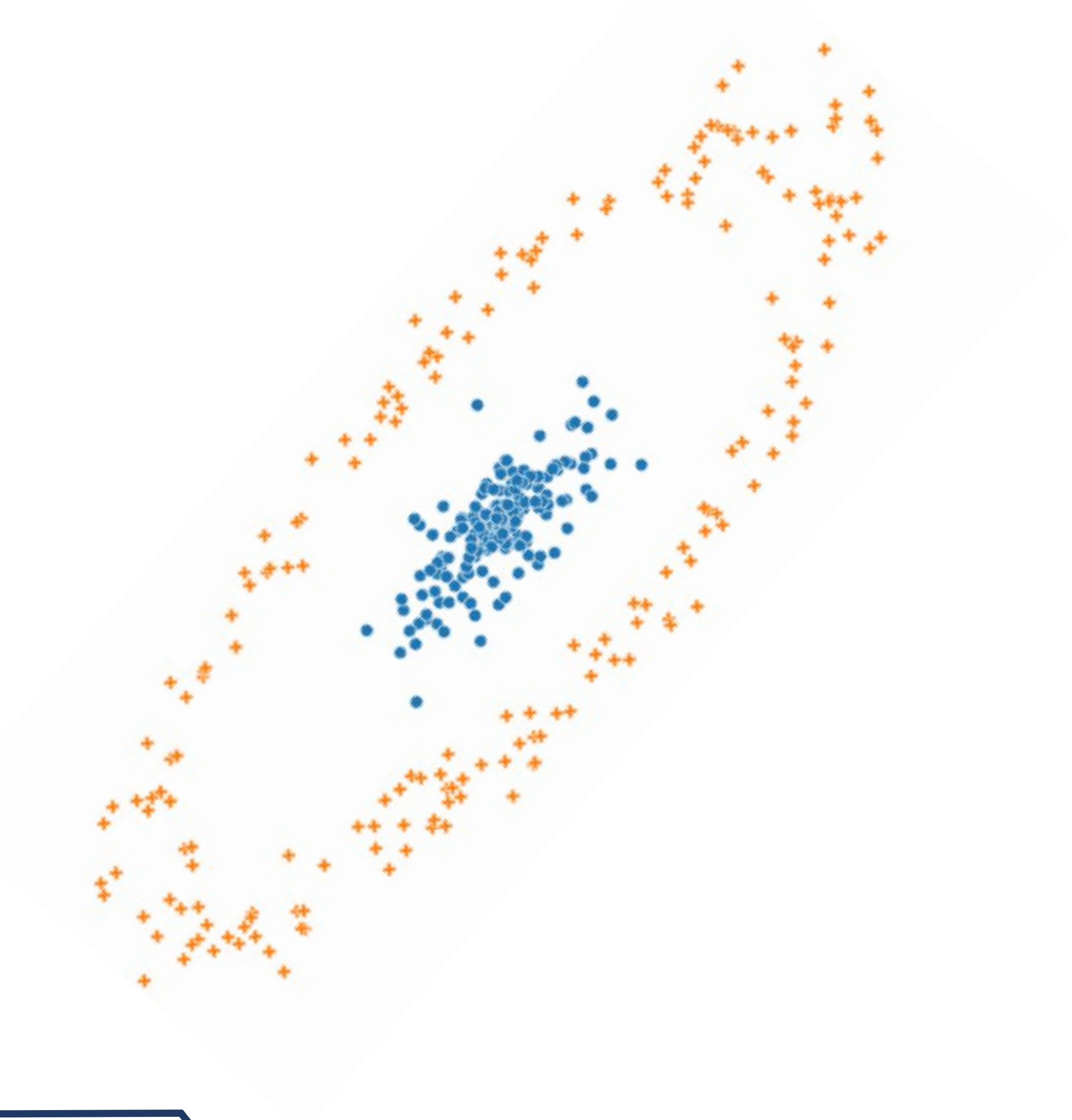
Feature extraction – PCA example



Feature extraction – PCA example



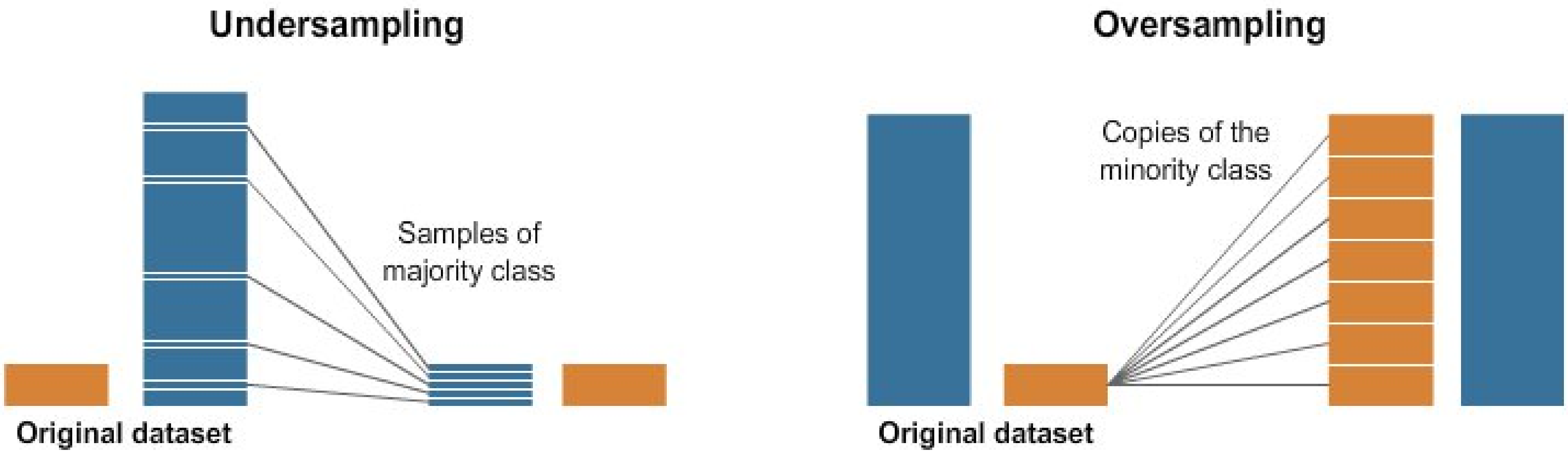
Feature transformation



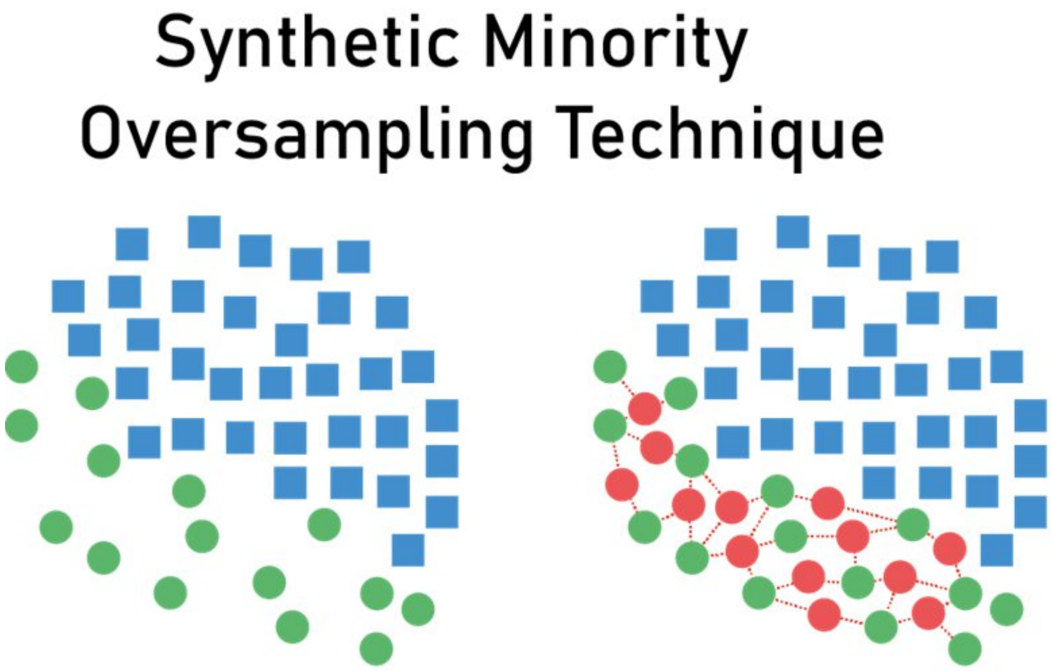
(x , y) ➡ Complex relation between x and y

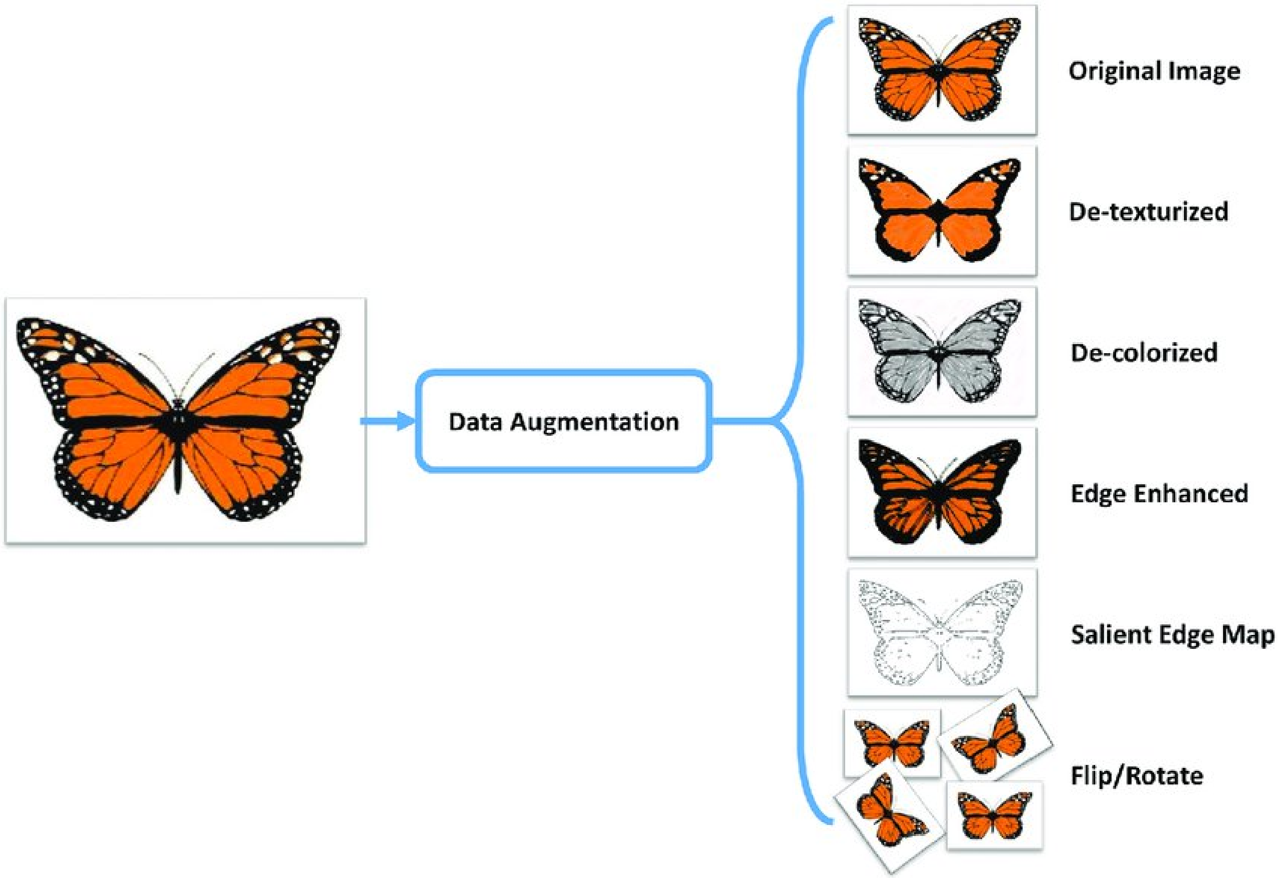
(r , θ) ➡ Simple relation with r and θ

Feature transformation - Example

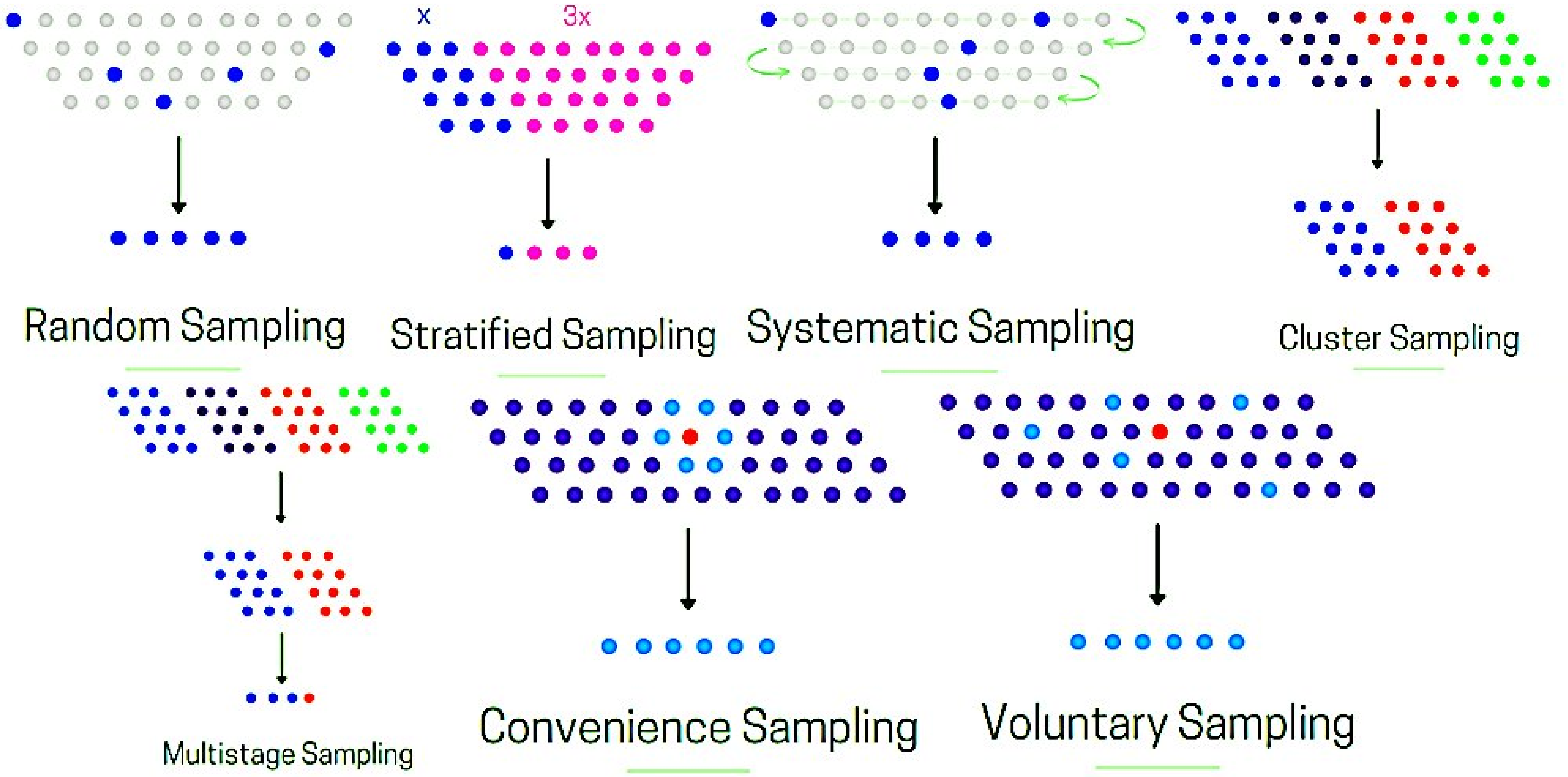


Balance the classes

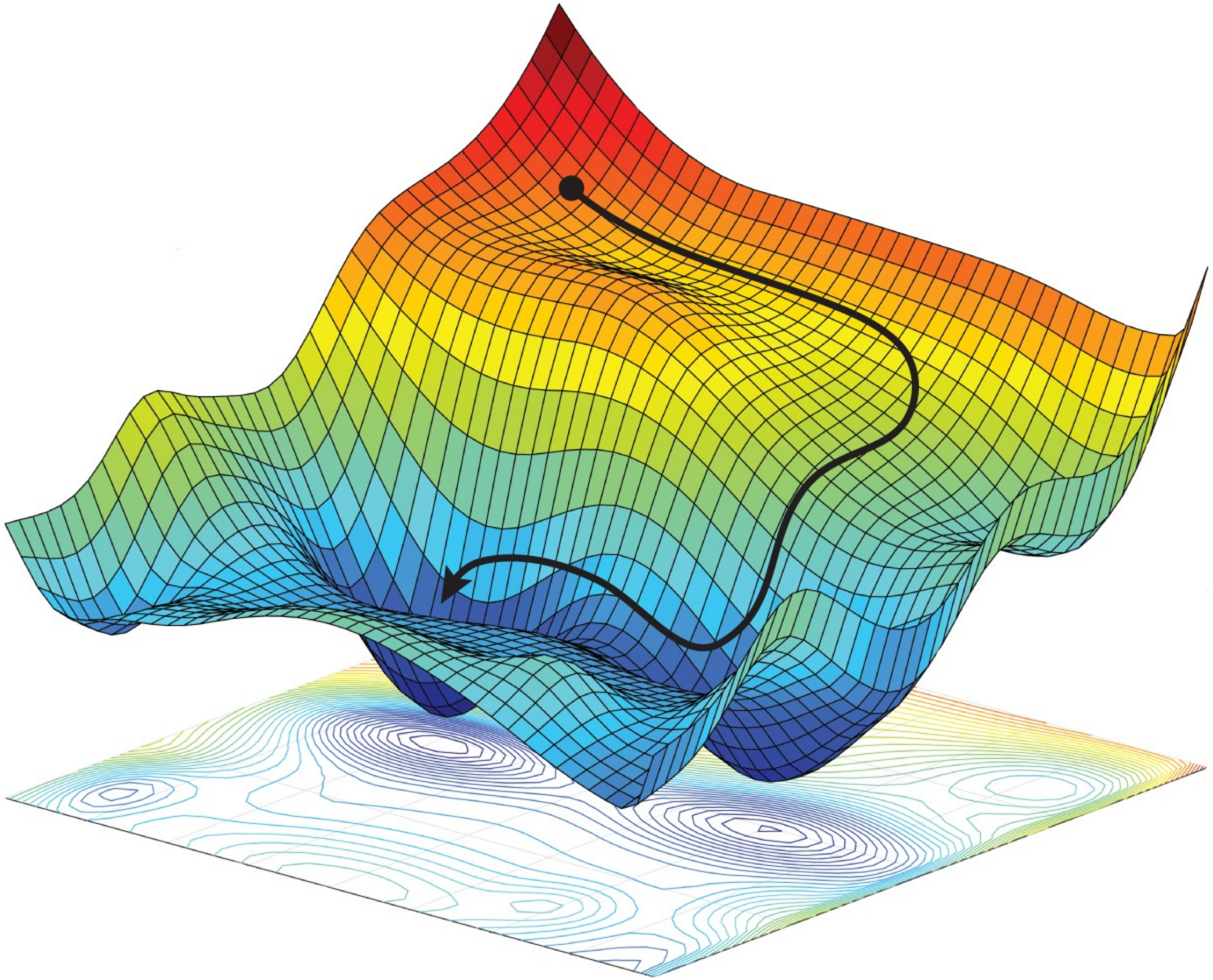




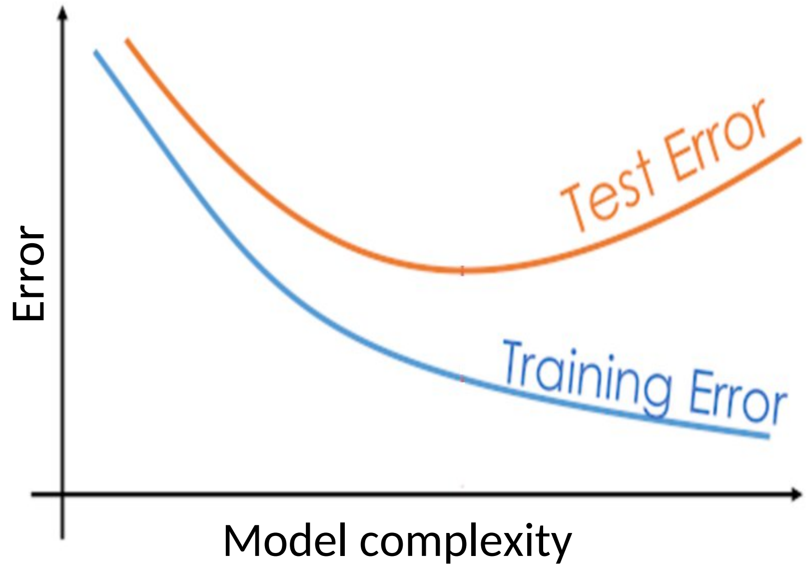
Data augmentation

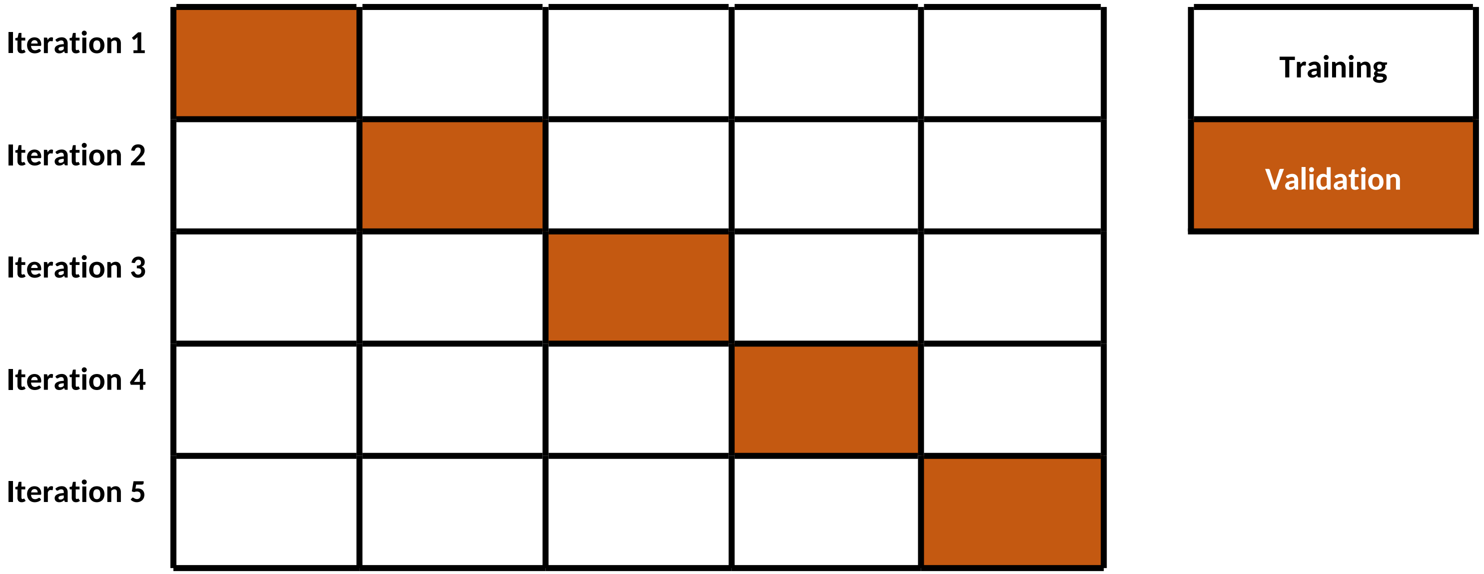


Under sampling



Reminder : training a neural network



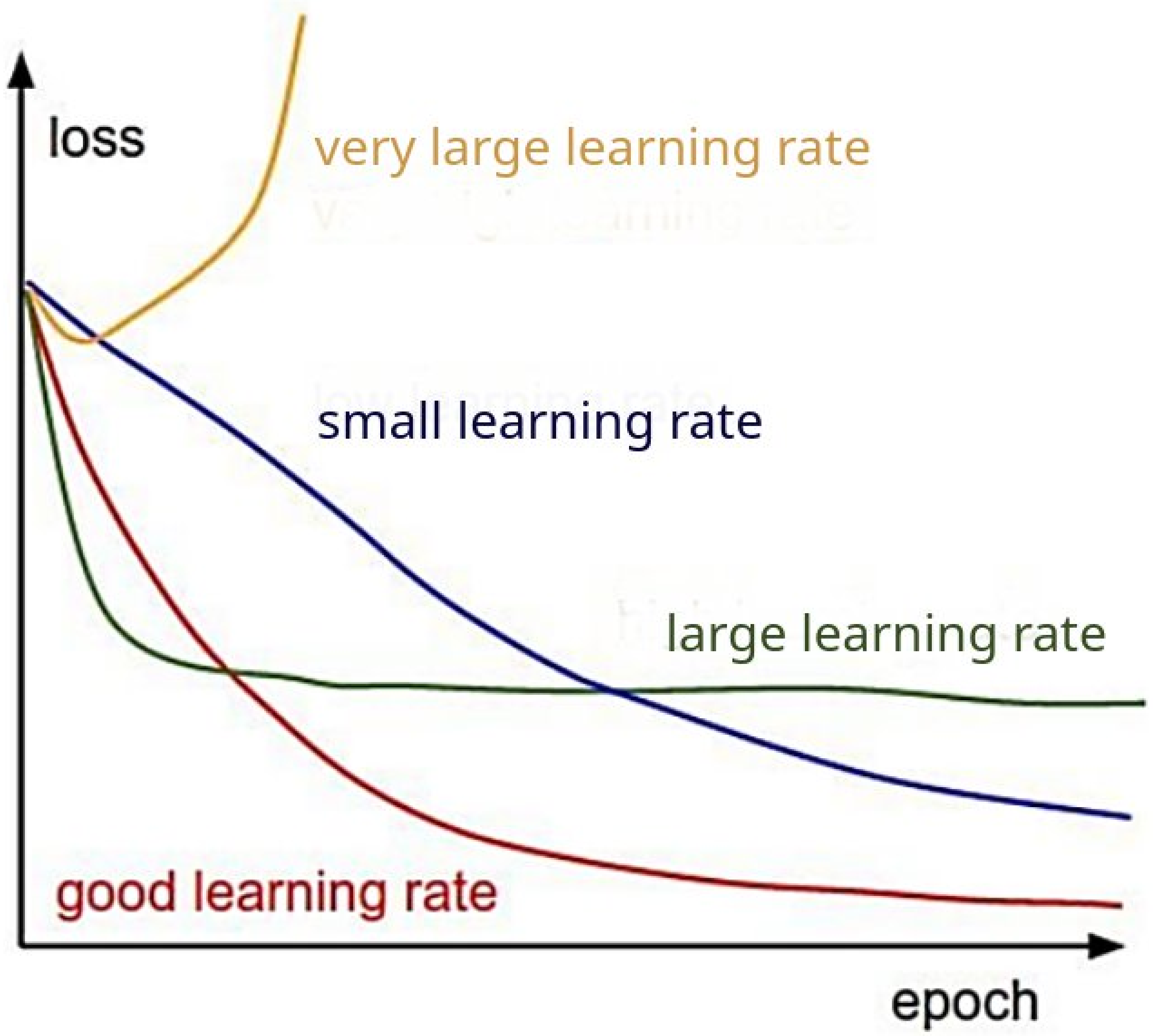


Splitting data in N parts

Data splitting

Find the hyper-parameters

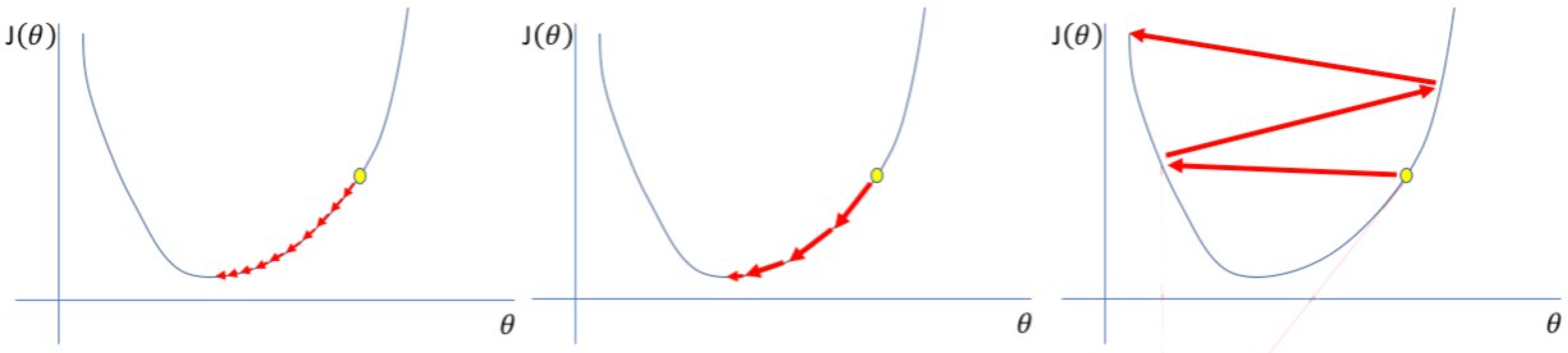
Hyper-parameters	Methods
• Learning rate • Regularization • Optimizer • Model architecture • Batch size • …	• Manual research • Grid search • Random research • Gradient • Evolutionary algorithms • …

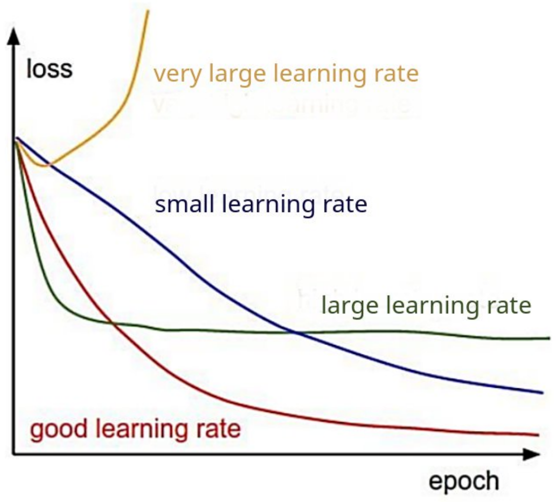


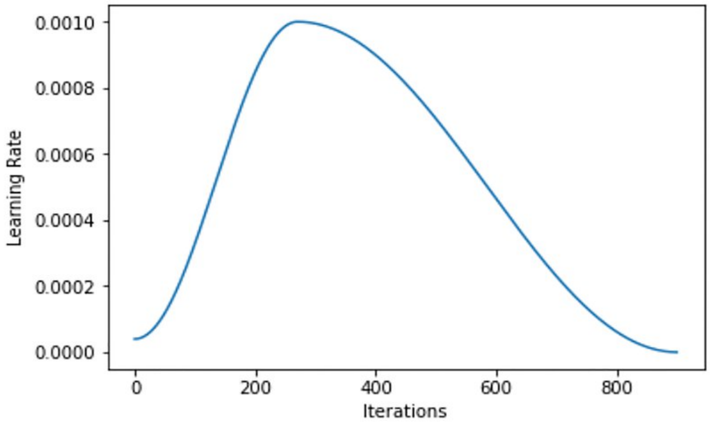
Learning rate

Learning rate

Too small | Just right | Too large



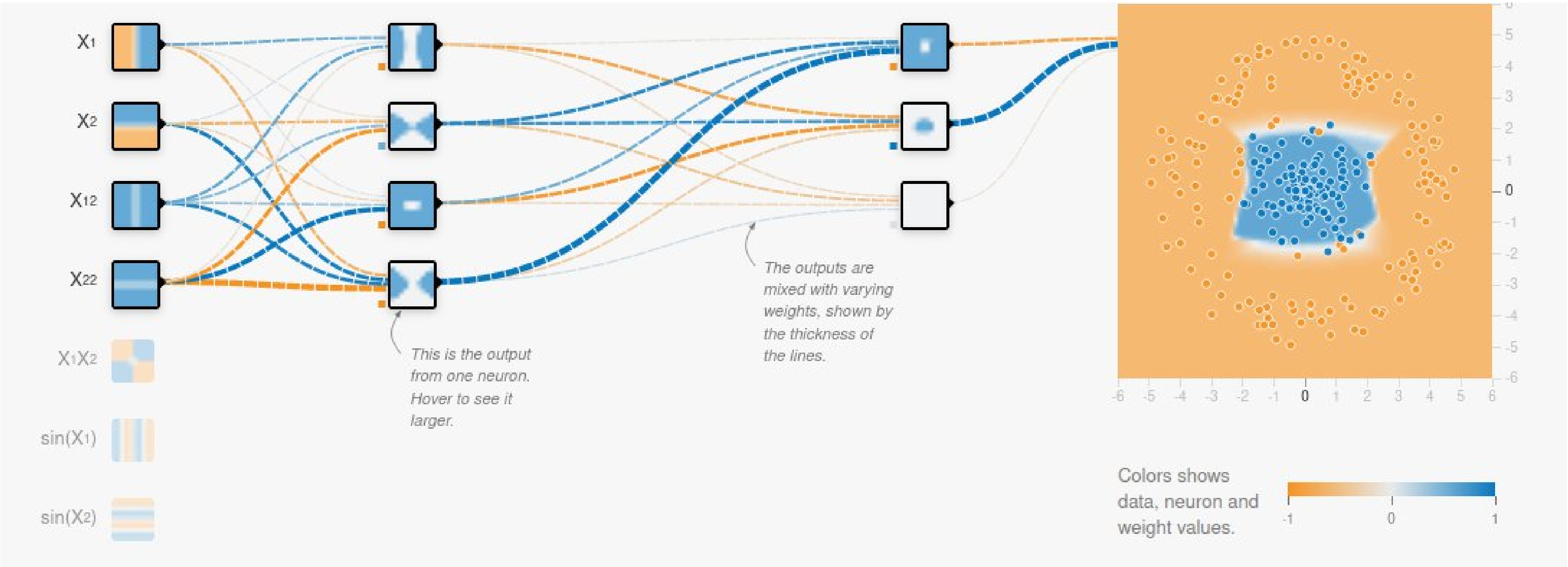




$$\Theta_{t+1} = \Theta_t - \eta \, \nabla_\Theta [\mathscr{L}(\hat{y}_i, y_i)]$$

Updated weights = Weights before update − Learning rate * Gradient [Cost function (Prediction , Label)]

Weight update equation



Regularization : motivation

$$\Theta_{t+1} = \Theta_t - \eta \nabla_\Theta [\mathscr{L}(\hat{y}_i, y_i) + \lambda R(\Theta_t)]$$

Updated weights = Weights before update − Learning rate * Gradient [Cost function (Prediction, Label) + Regularization rate * Regularization function (Weights before update)]

Weight update equation

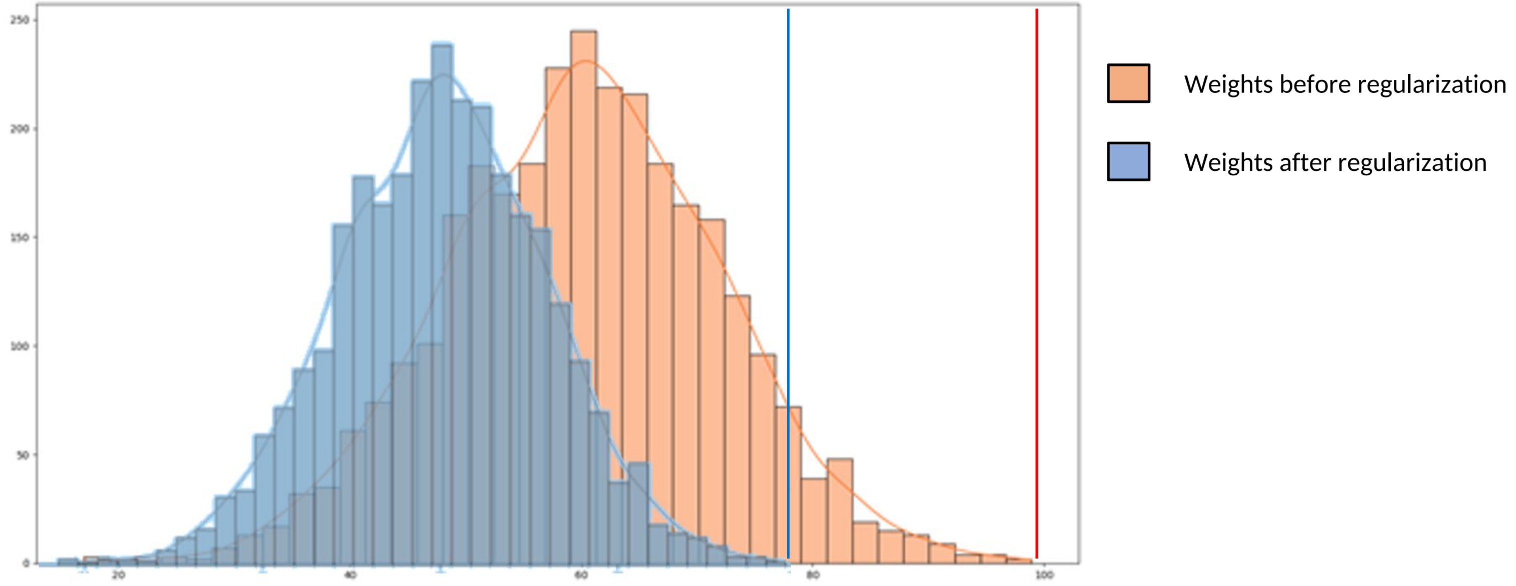
- L1 Regularization
- L2 Regularization
- Max norm Regularization
- Regularization with the cost function
- Dropout

L1 : LASSO	L2 : Ridge
$\|\Theta\|$	Θ^2

Regularization: L1 and L2

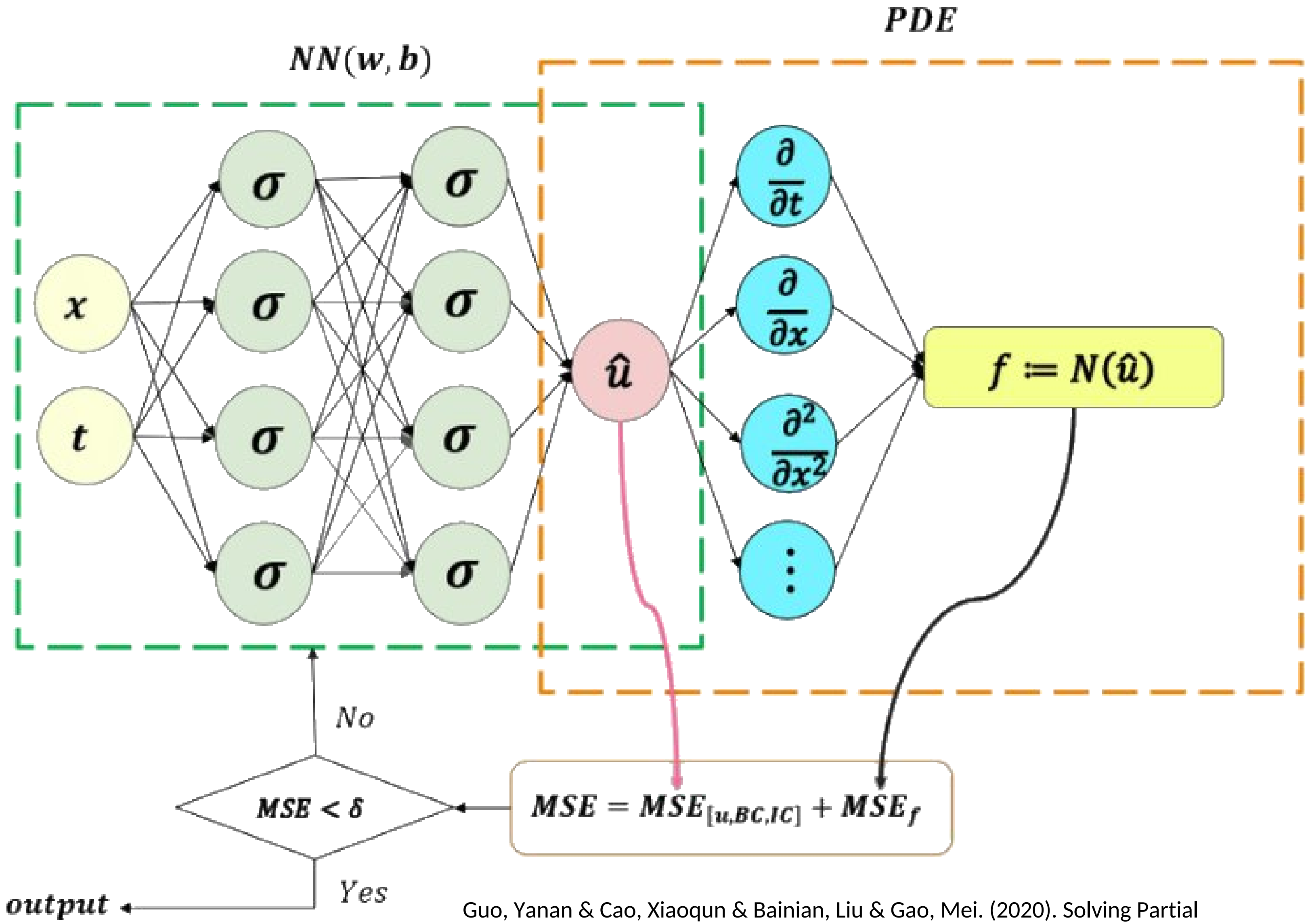
$$\Theta_{t+1} = \frac{\Theta_{t+1}}{max(\Theta_{t+1})} * scale$$

Weights distribution over a layer



$scale$ $max(\Theta_{t+1})$

Regularization: max norm



Guo, Yanan & Cao, Xiaoqun & Bainian, Liu & Gao, Mei. (2020). Solving Partial Differential Equations Using Deep Learning and Physical Constraints

Regularization: with the cost function (PINN example)

PINN : 1D heat equation

Exercise :
The heat diffusion over time along one dimension is governed by the following equation :

$$\frac{\partial u}{\partial t} - k\frac{\partial^2 u}{\partial^2 x} = 0$$

with :

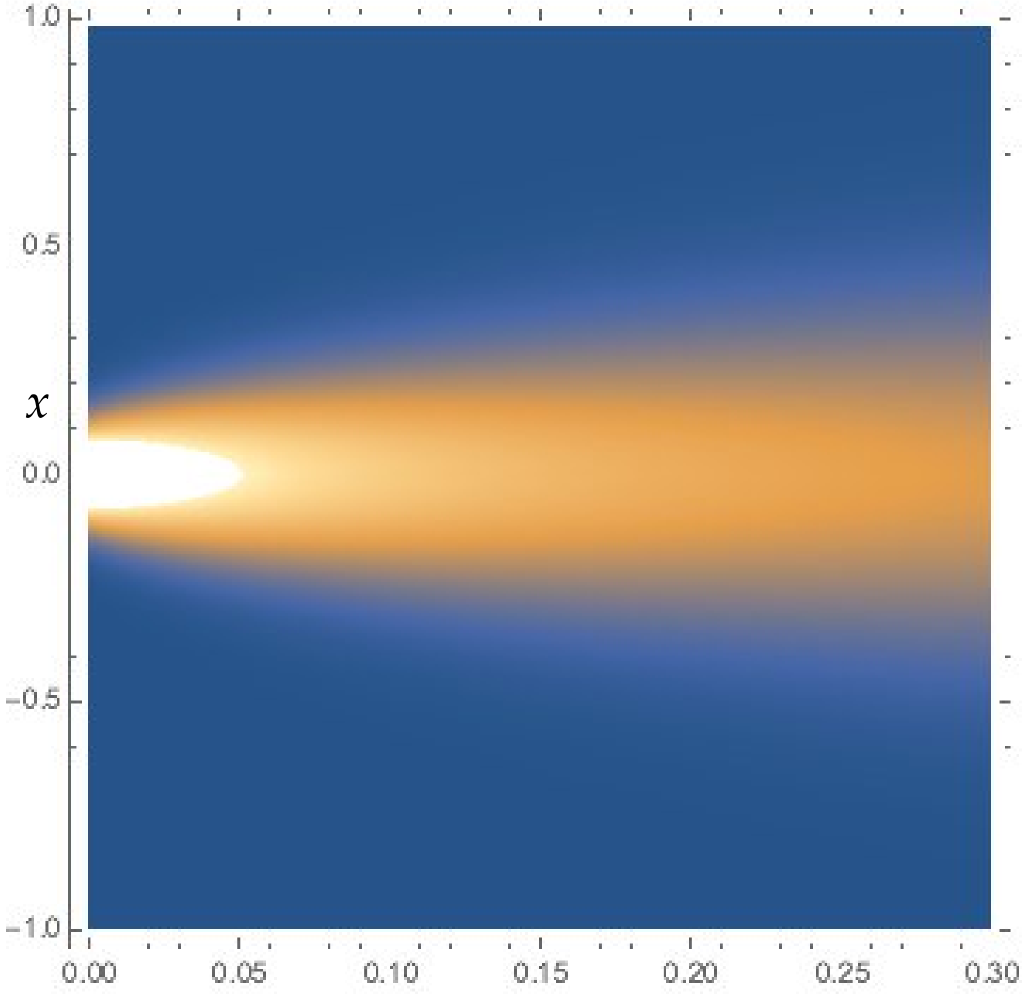
- u : the heat distribution along x,
- k : the thermal diffusivity of the material

We set theses constraints :

1. $u(x,0) = exp\left(-\left(\frac{x}{0.1}\right)^2\right)$ pour $x \in [-1,1]$
2. $u(-1,t) = u(1,t) \quad \forall\, t$

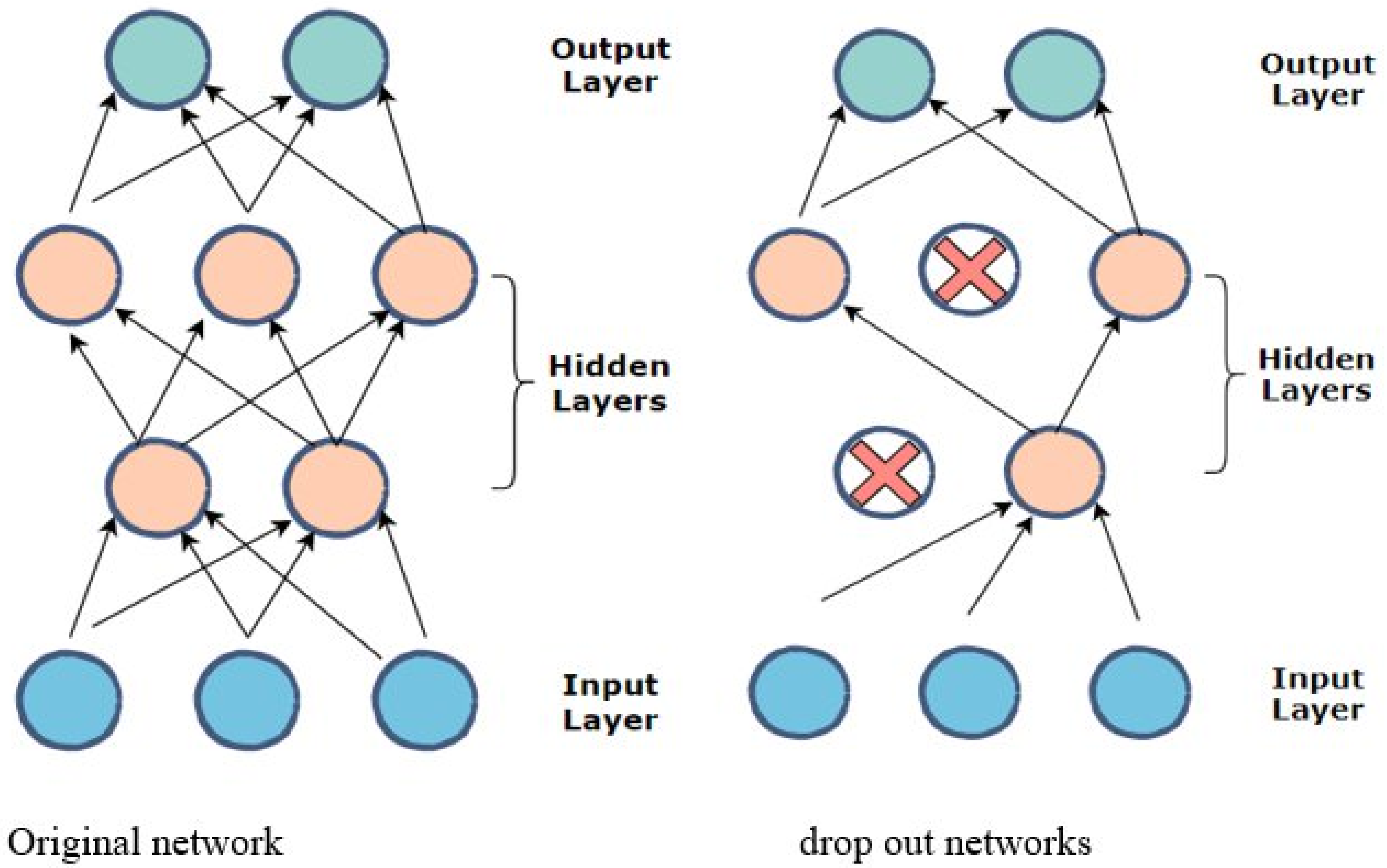
Result : The final loss is below 10^{-6}

Solution



x

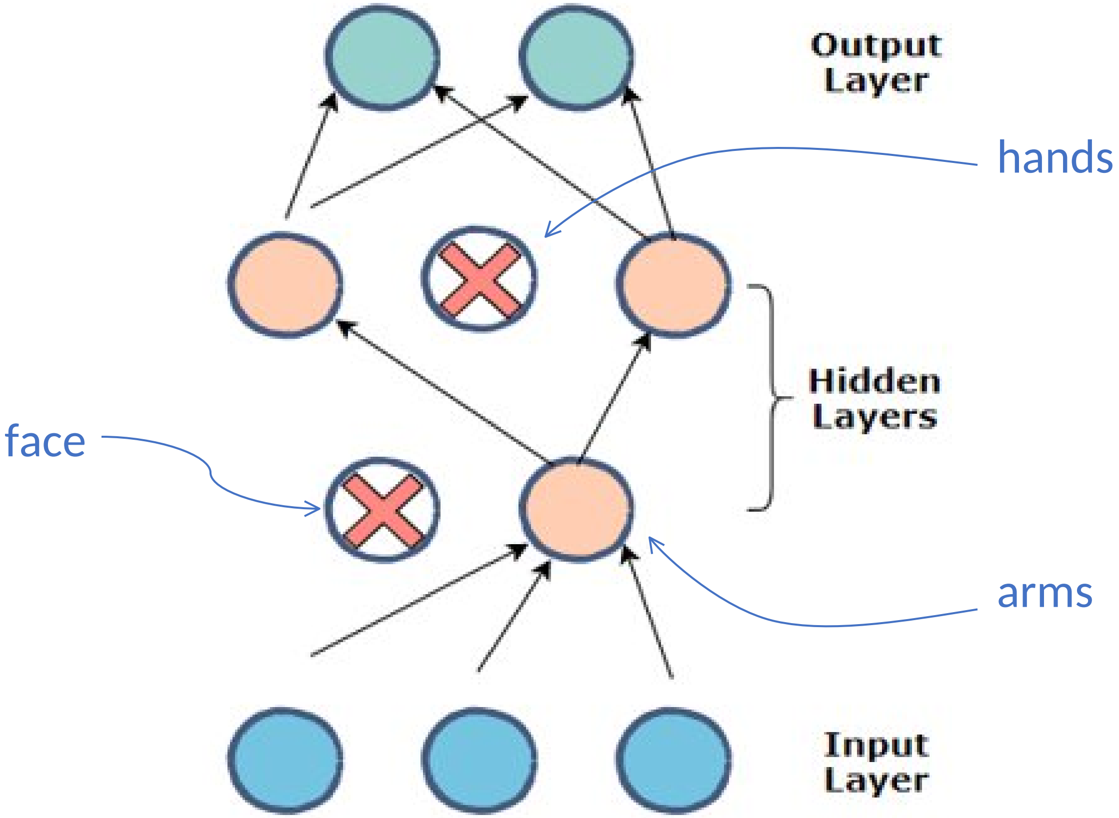
t



Regularization: Dropout

Human = face ?

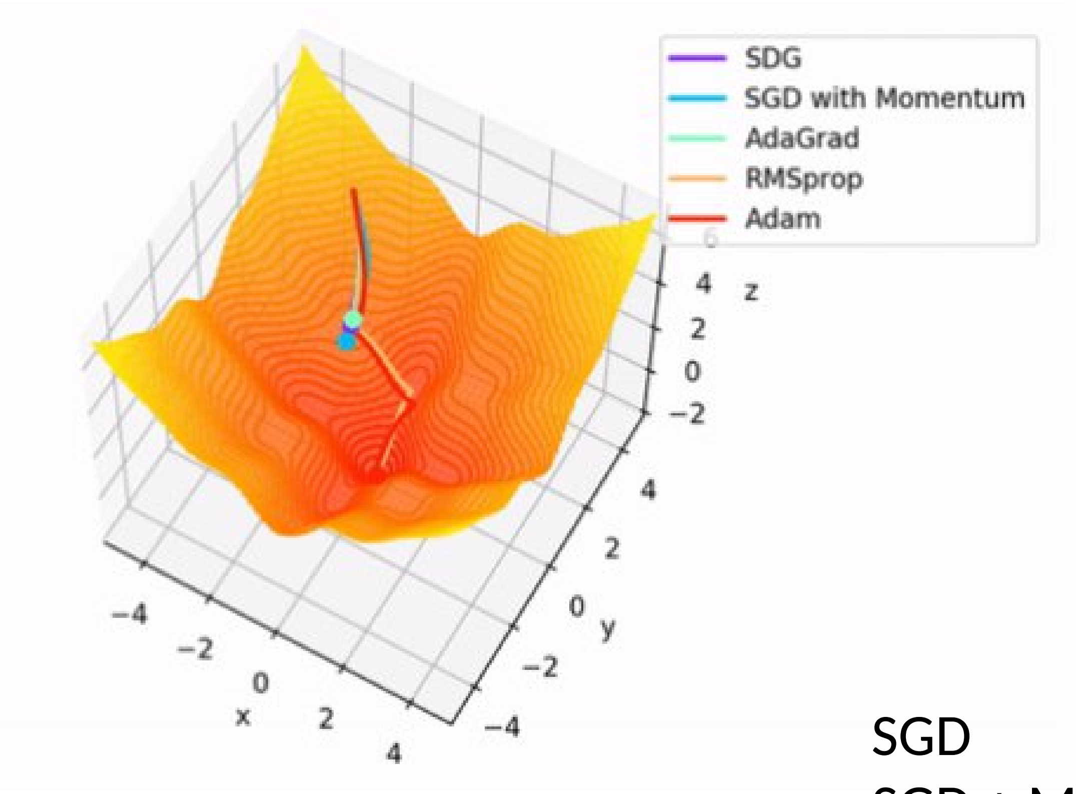
(a) Husky classified as wolf (b) Explanation

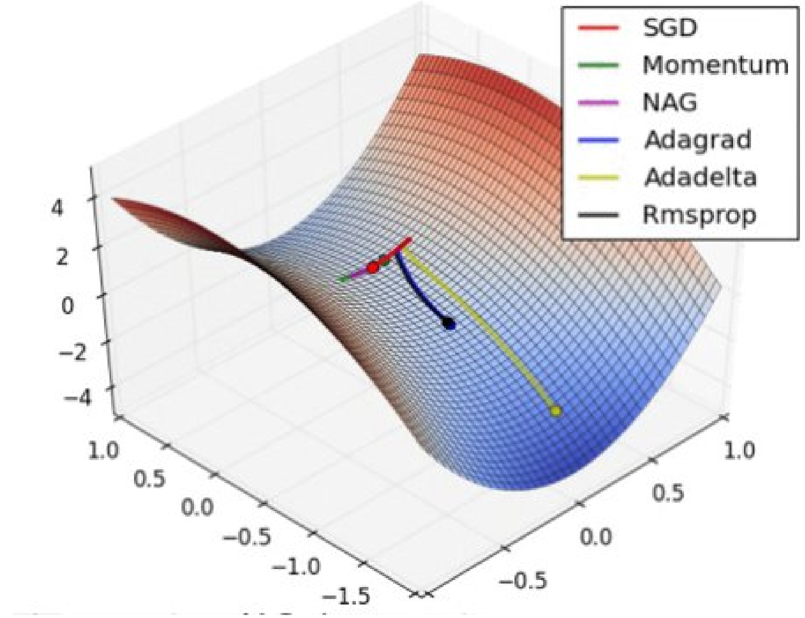


drop out networks

Source :
pytorch.org
data-flair.training
Right for the Right Reason: Making Image Classification Robust

Regularization: Dropout



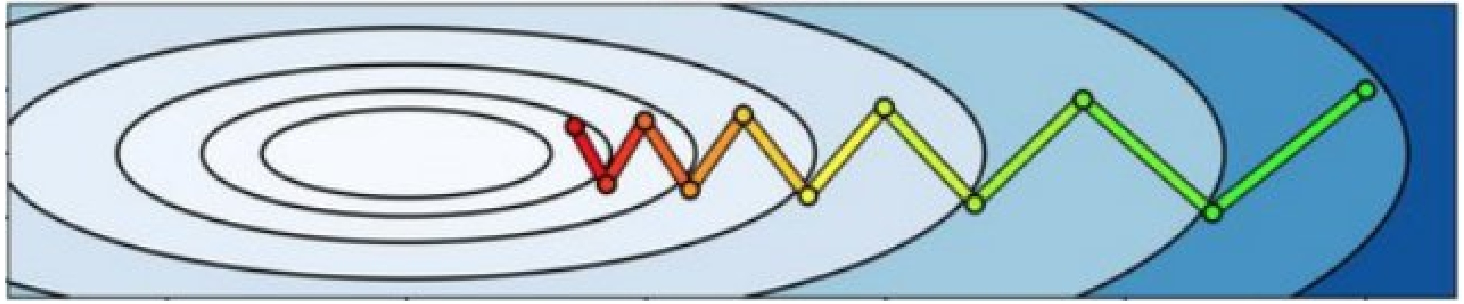


SGD
SGD + Momentum
NAG
AdaGrad
AdaDelta
RMSprop
Adam

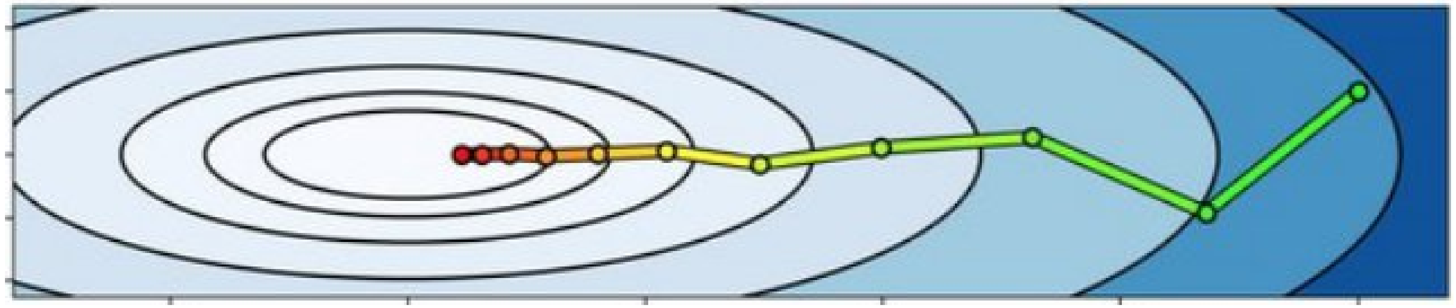
Optimizers

Optimizers: Momentum

Convergence

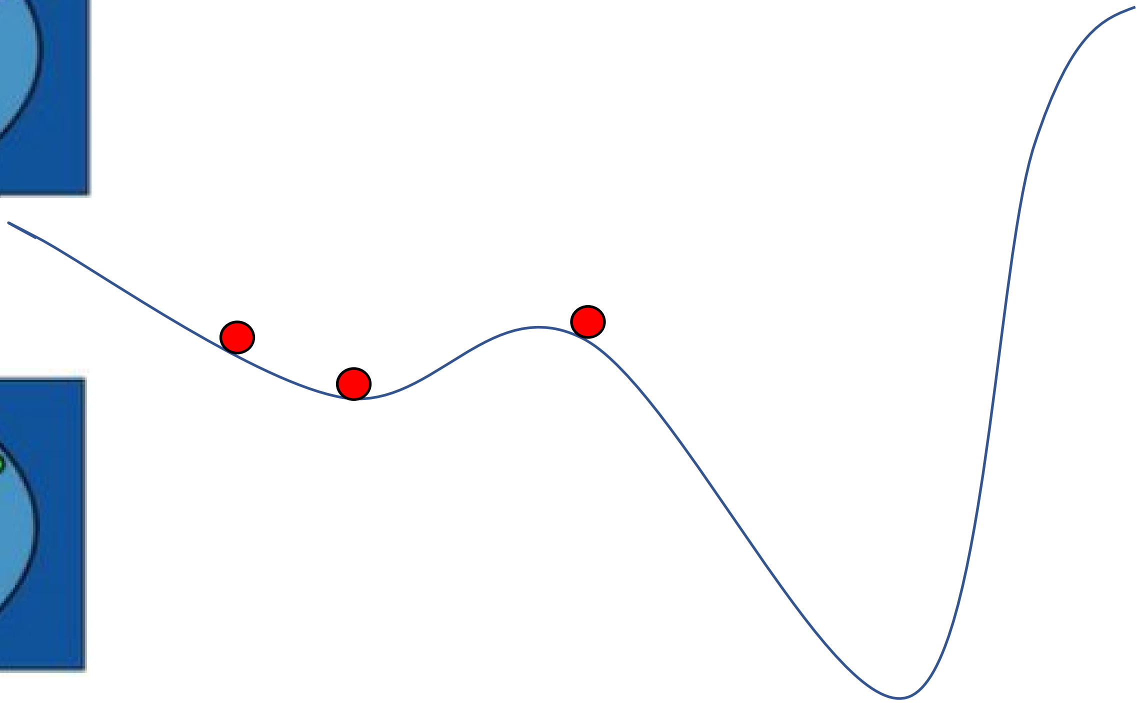


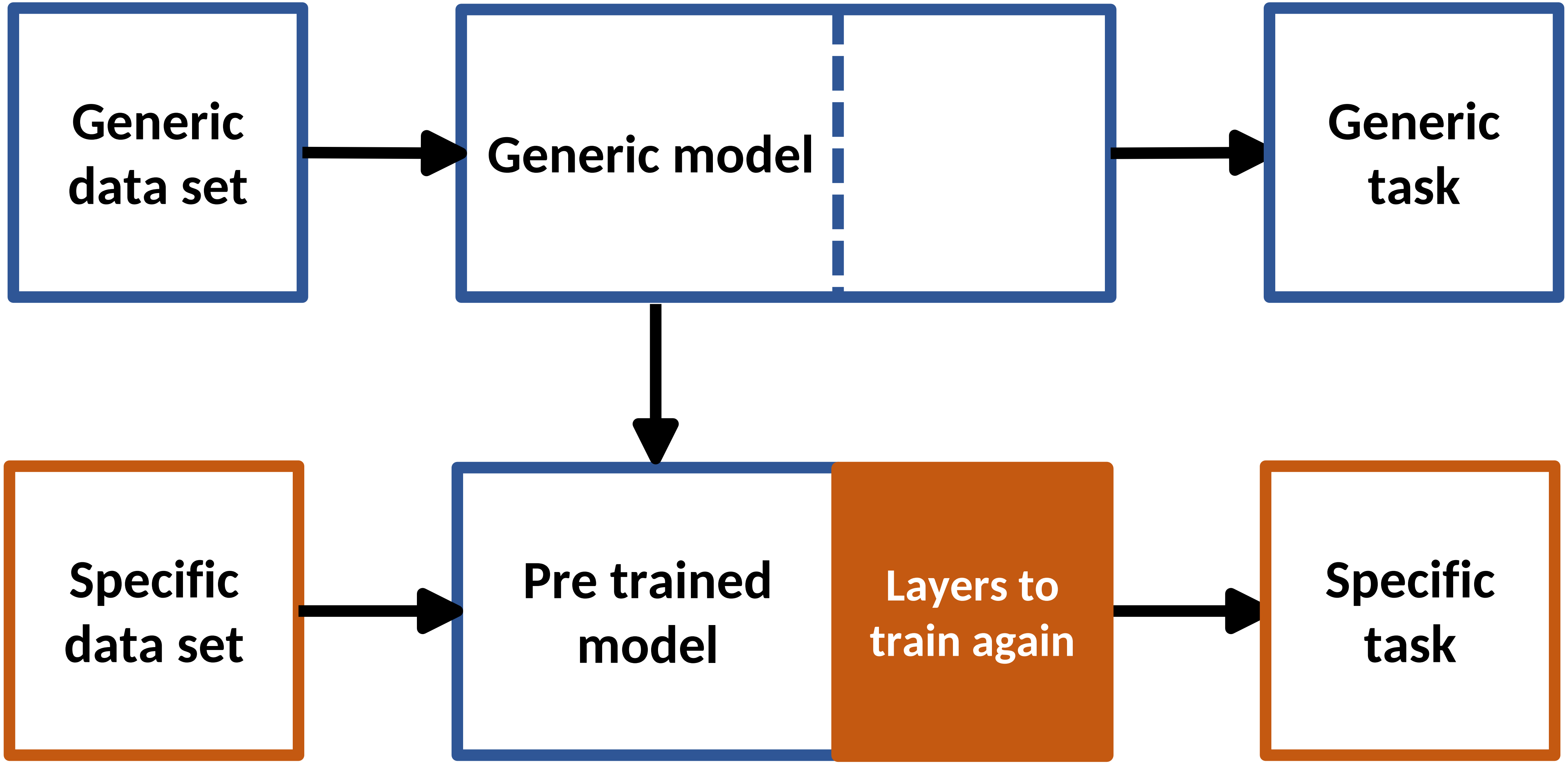
Without momentum



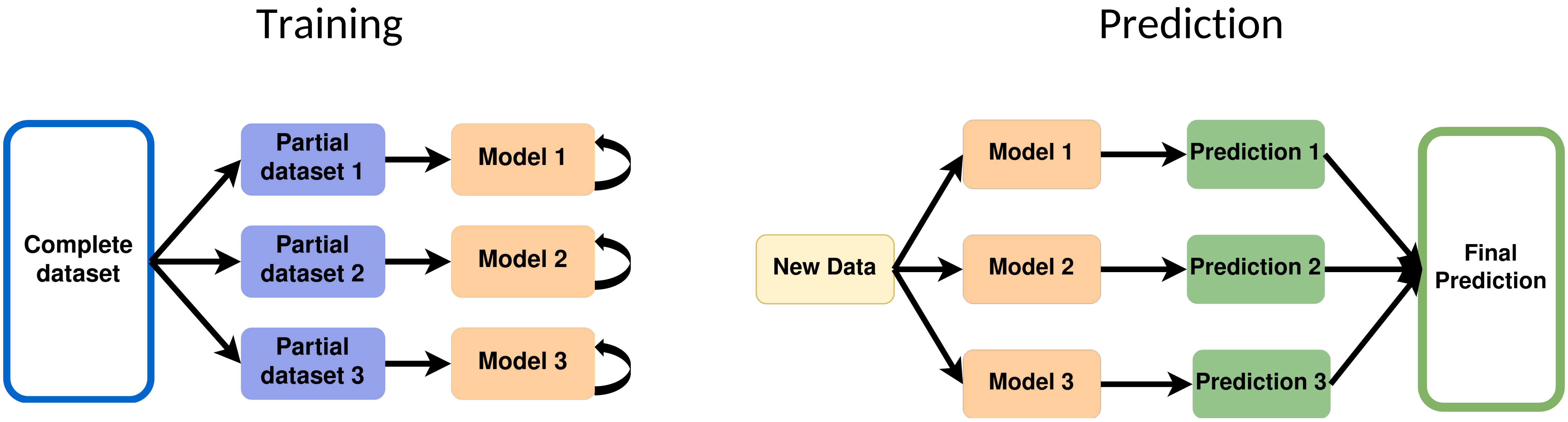
With momentum

Local minimum





Transfer Learning



Ensemble learning (ex : bagging)