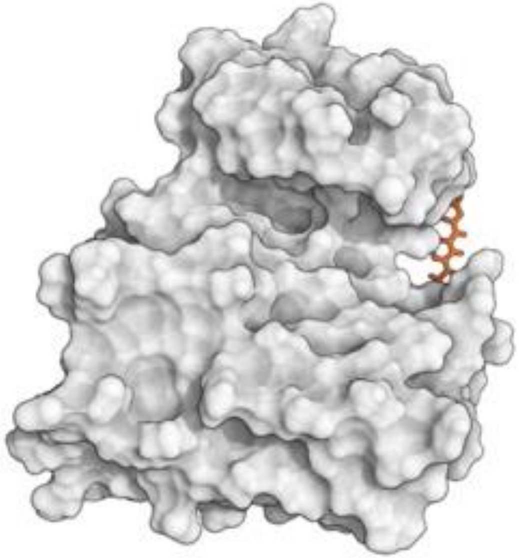
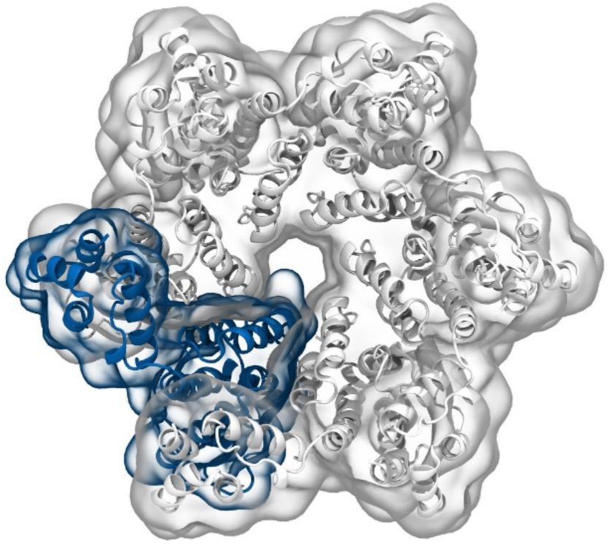
Simulation of Biomolecules



Simulation Analysis part 2

2023 CCP5 Summer School

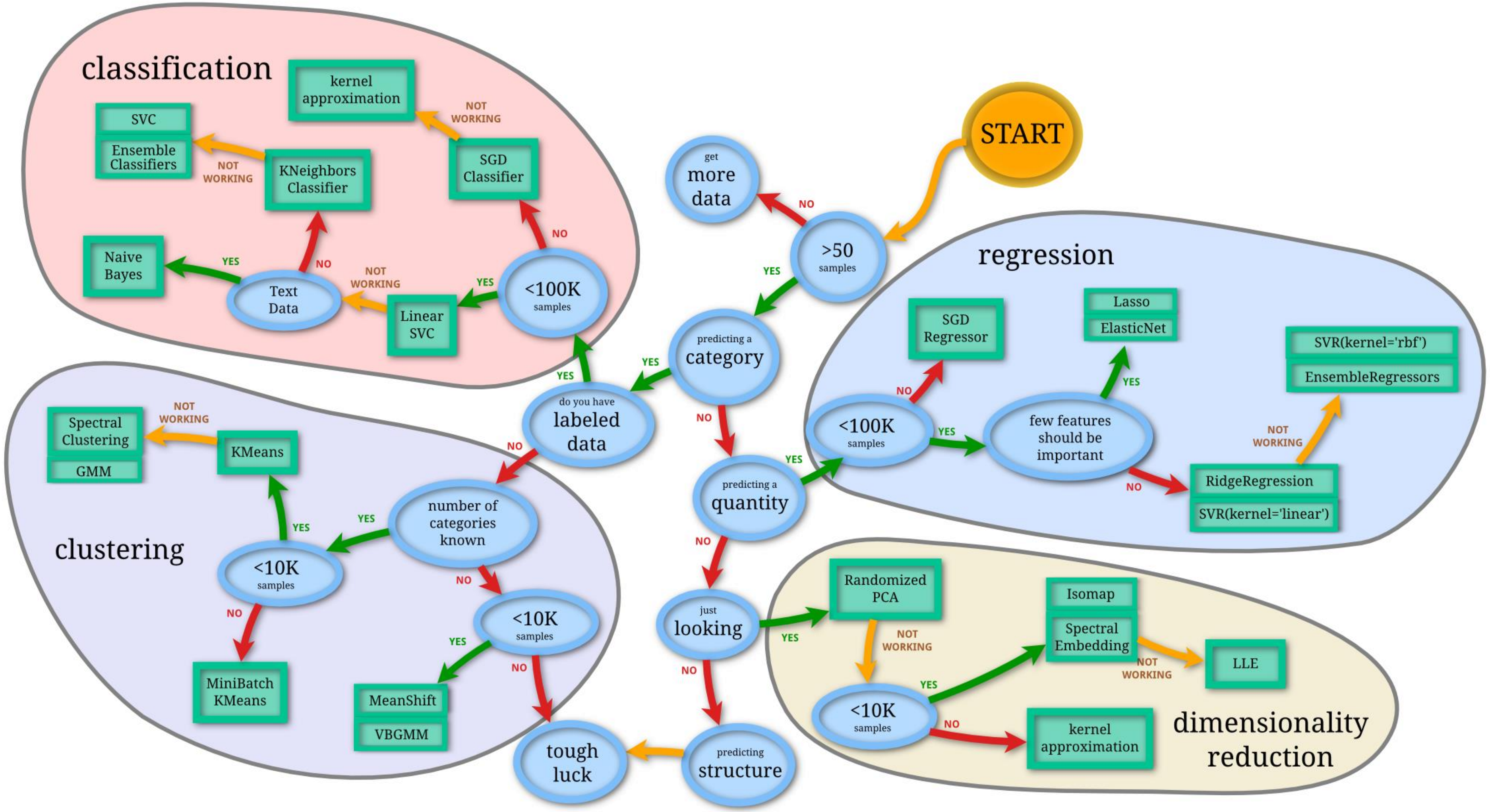


Dr Agnieszka Bronowska
Newcastle University
agnieszka.bronowska@newcastle.ac.uk

Dr Matteo Degiacomi
Durham University
matteo.t.degiacomi@durham.ac.uk

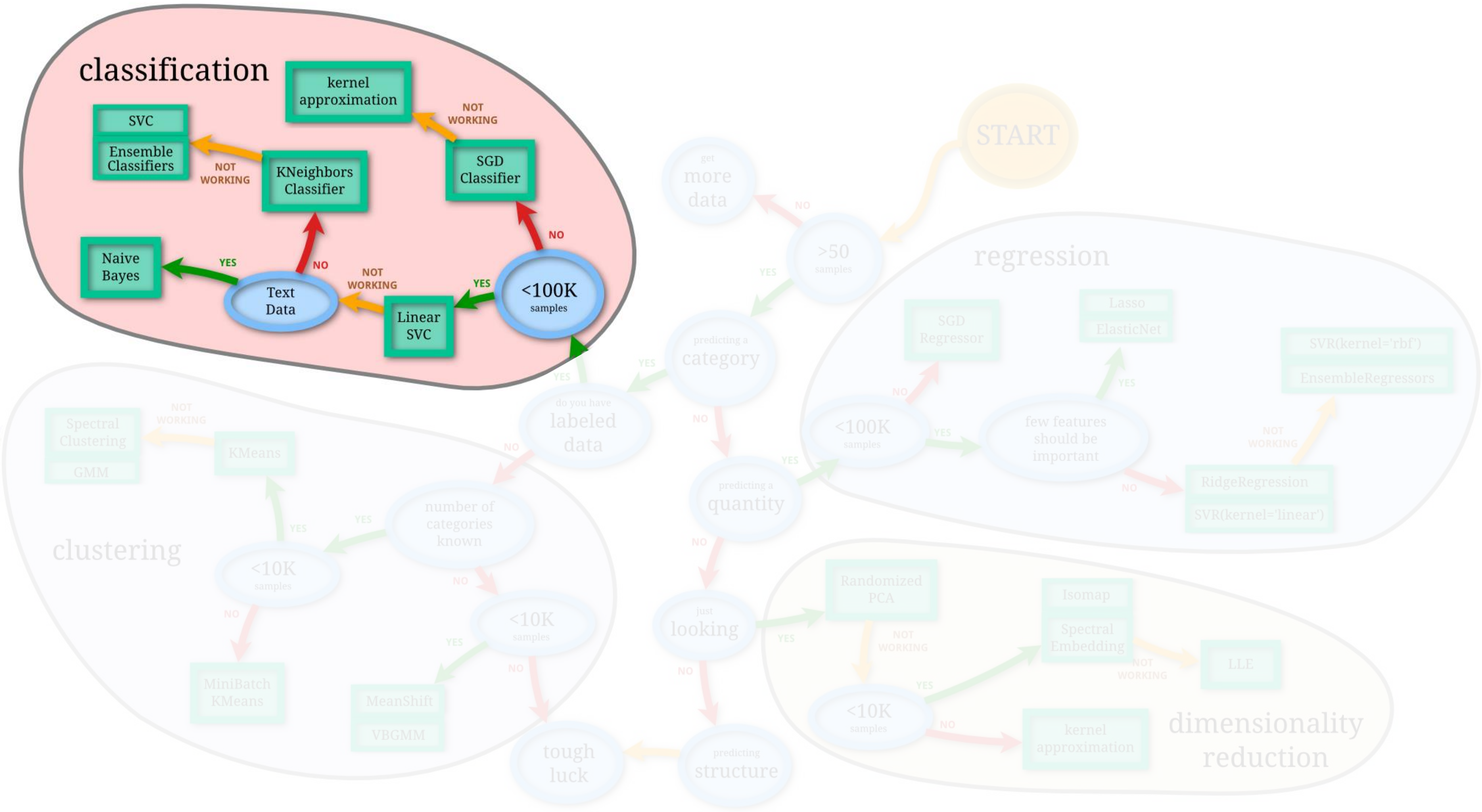
Dr Antonia Mey
University of Edinburgh
antonia.mey@ed.ac.uk

The Data Mining world



From **scikit-learn.org**

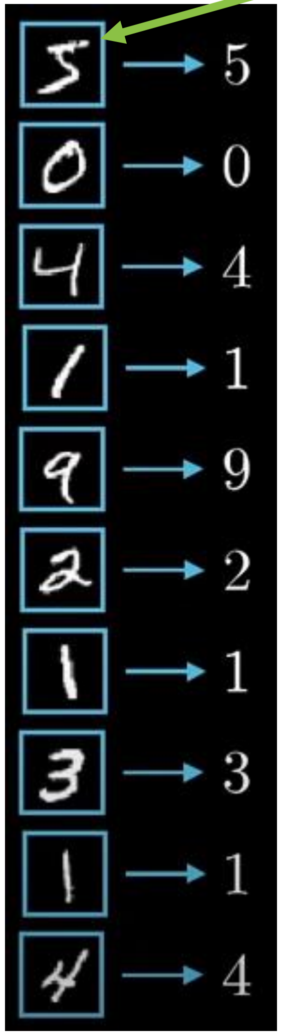
The Data Mining world

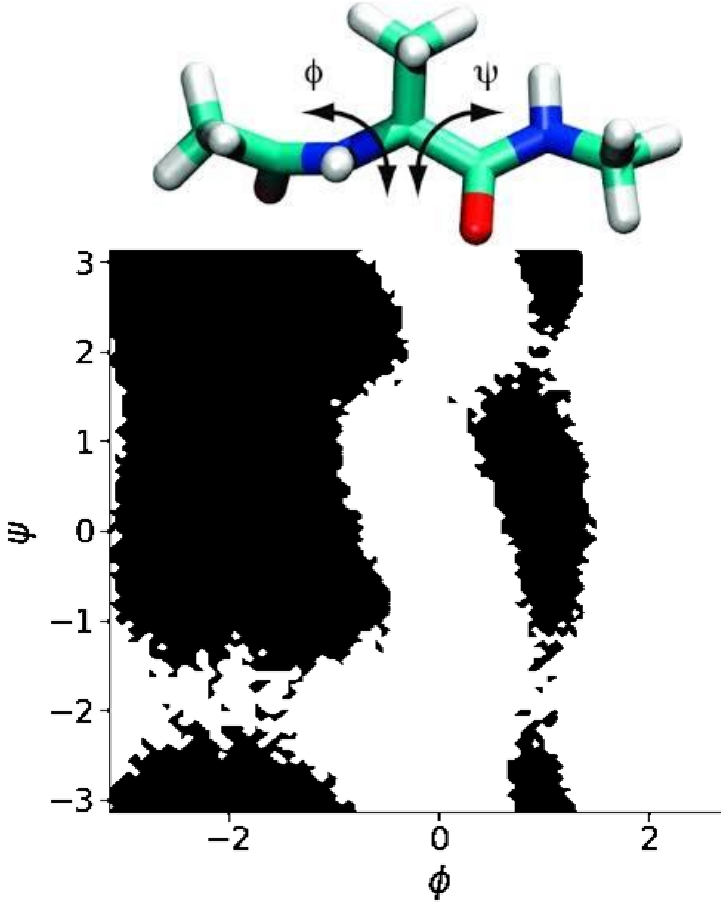


From **scikit-learn.org**

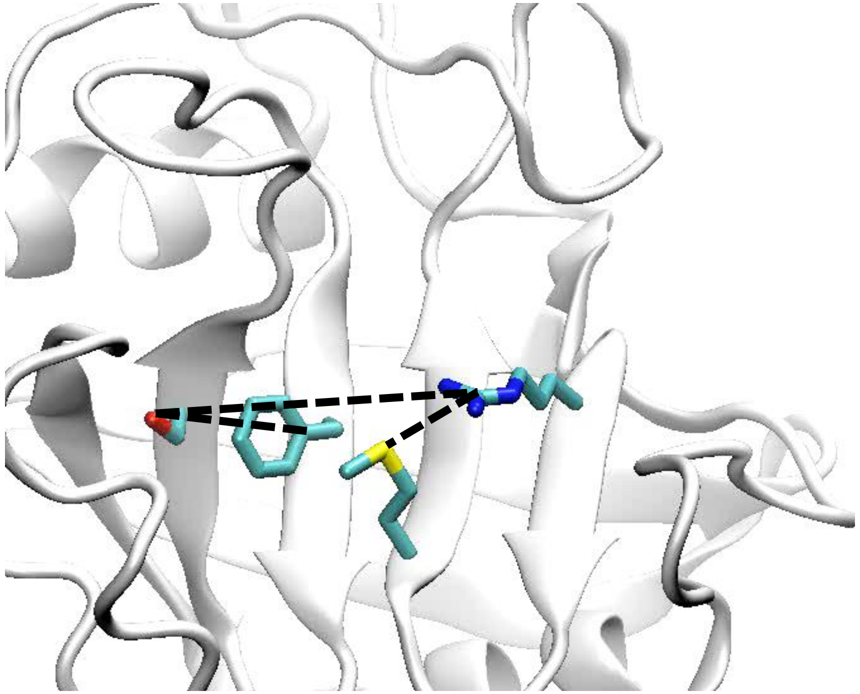
Features are possible ways to represent data

Pixels colour





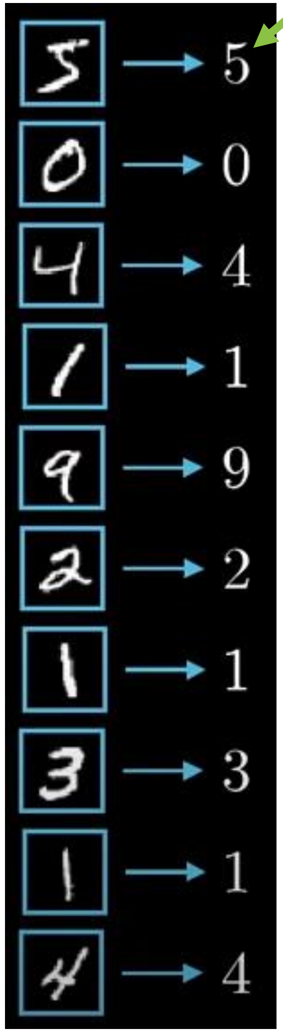
Torsional angles

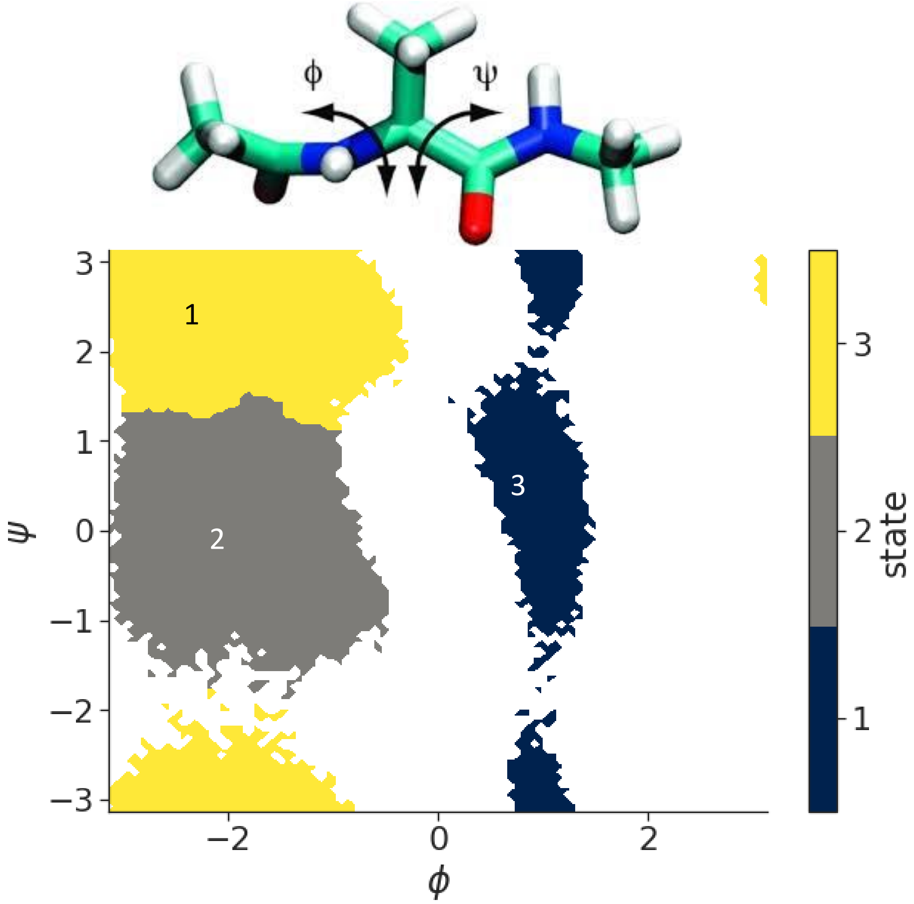


Interactomic distances

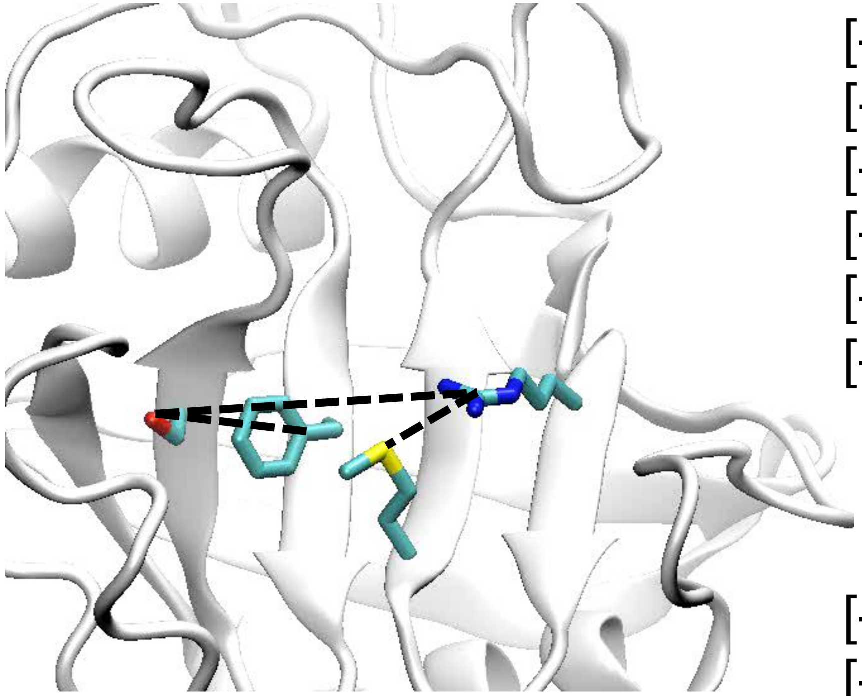
Labels assign featurised data to categories

Labelled digits





Labelled states

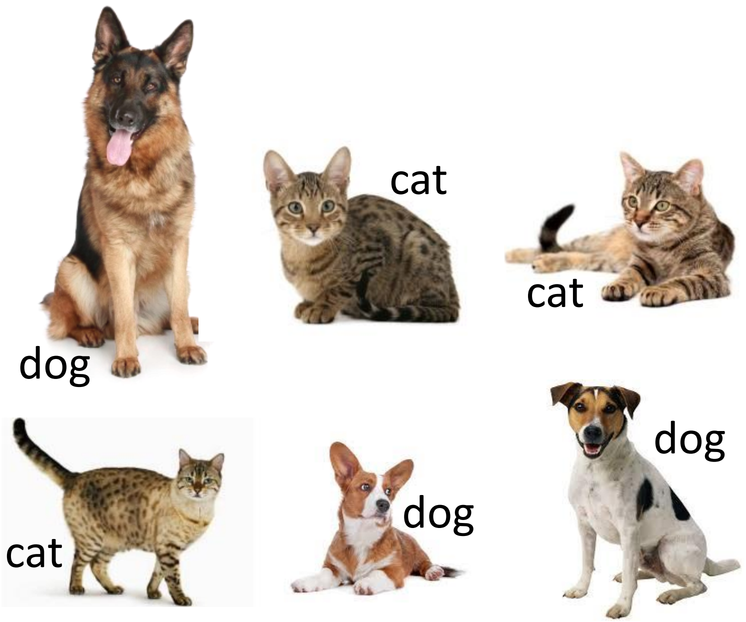


[-12008 kJ]
[-12078 kJ]
[-12045 kJ]
[-12083 kJ]
[-12062 kJ]
[-12058 kJ]
.
.
.
[-12099 kJ]
[-12093 kJ]

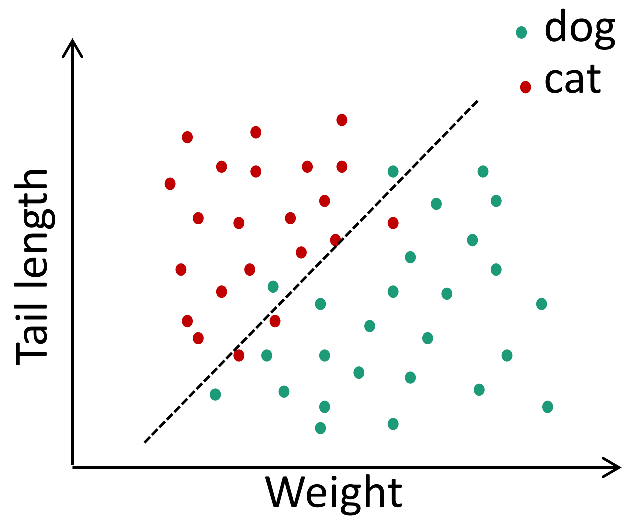
Labelled Energies

Data Classification via Supervised Learning

- take **labelled data**
- create an n-dimensional **feature vector** from data
- Separate «feature space» in different regions



Weight
Legs length
Tail length
Eye color
…



Data Classification via Supervised Learning

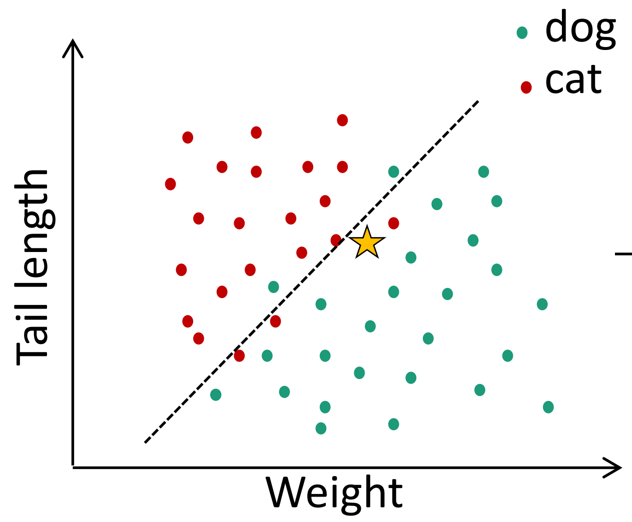
- take **labelled data**
- create an n-dimensional **feature vector** from data
- Separate «feature space» in different regions

?

→

Weight
Legs length
Tail length
Eye color
...

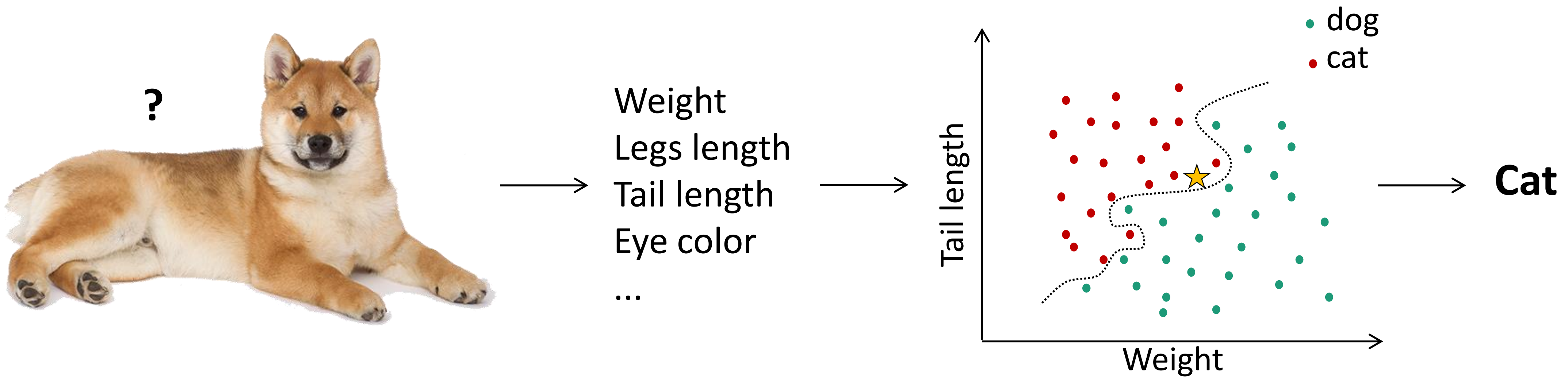
→



→ **Dog**

Data Classification via Supervised Learning

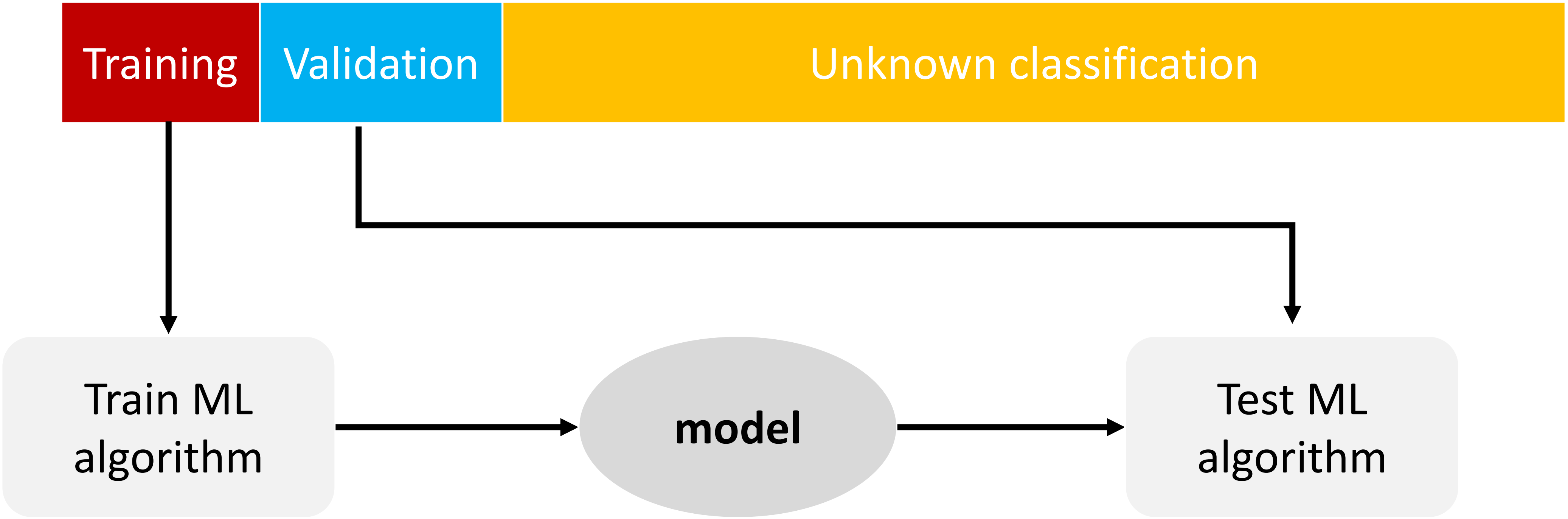
- take **labelled data**
- create an n-dimensional **feature vector** from data
- Separate «feature space» in different regions
- Warning: a too precise classification of examples might sacrifice generality (**overfitting**)



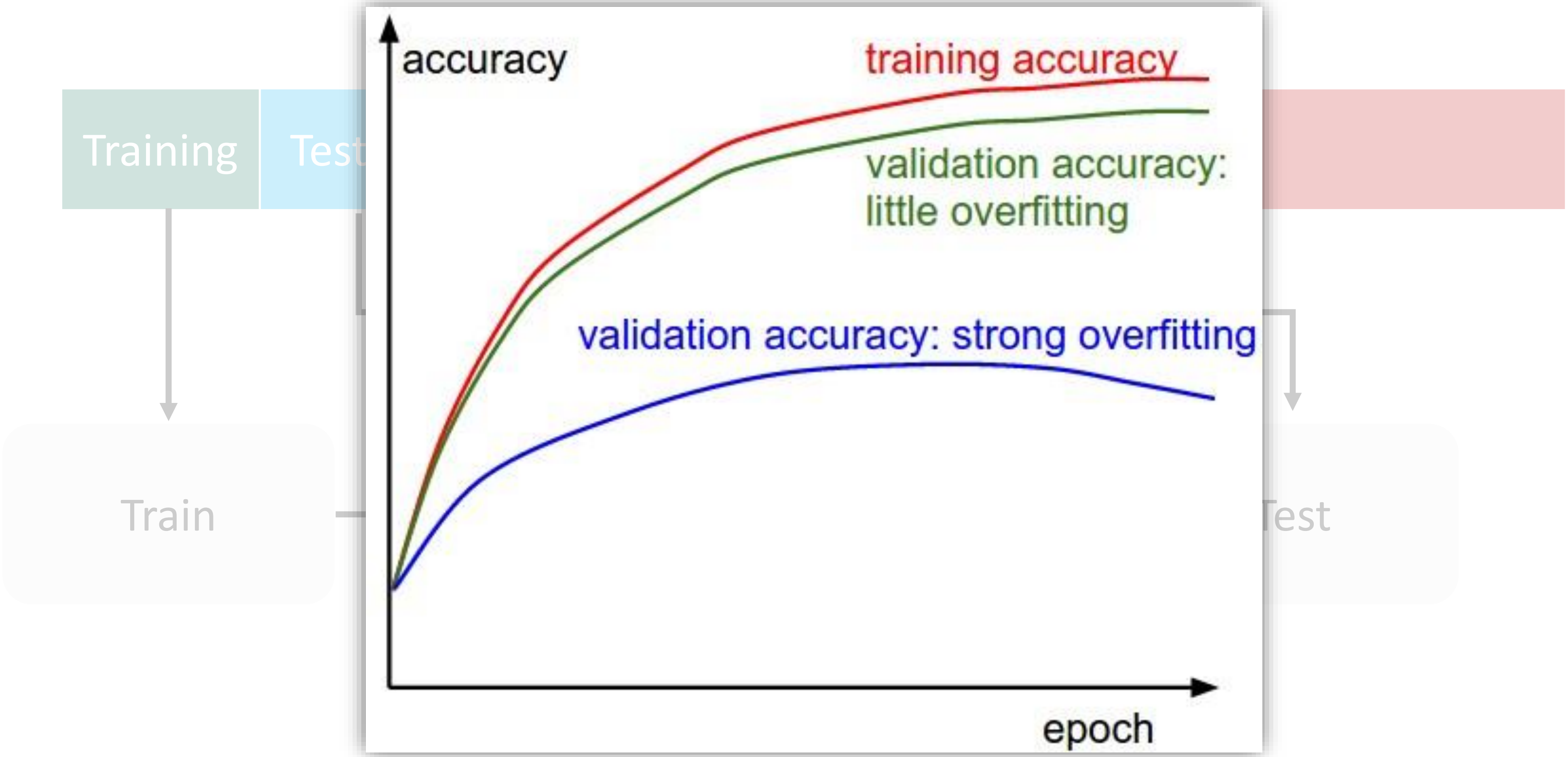
Data Classification via Supervised Learning

Data

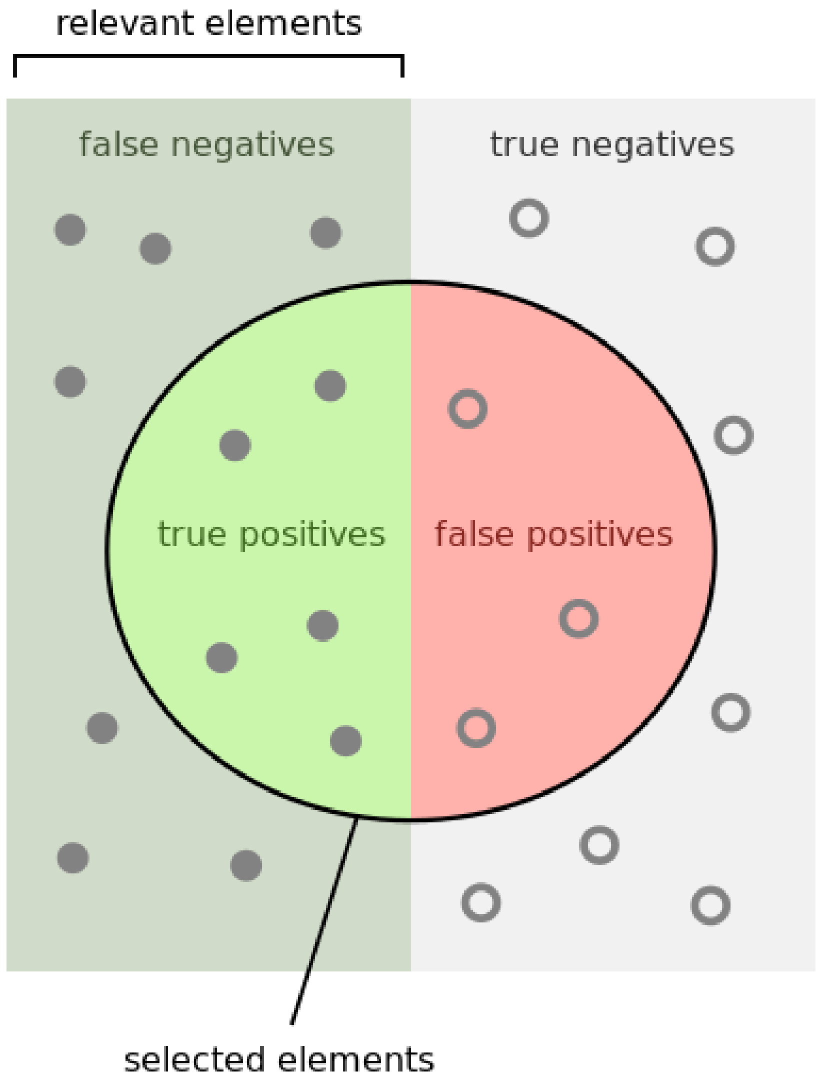
Data Classification via Supervised Learning



Data Classification via Supervised Learning

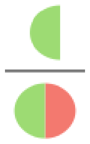


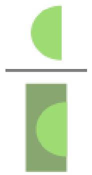
Some terminology



- **Confusion Matrix:**
  describes classification results
  can also describe n classes

result \ real	Dog	Cat
Dog	90	10
Cat	12	88

- $\textbf{precision} = \dfrac{\text{true positives}}{\text{selected elements}} =$ 

- $\textbf{sensitivity} = \textbf{recall} = \dfrac{\text{true positives}}{\text{relevant elements}} =$ 

- $\textbf{accuracy} = \dfrac{\text{true positives} + \text{true negatives}}{\text{total population}}$

en.wikipedia.org/wiki/Precision_and_recall

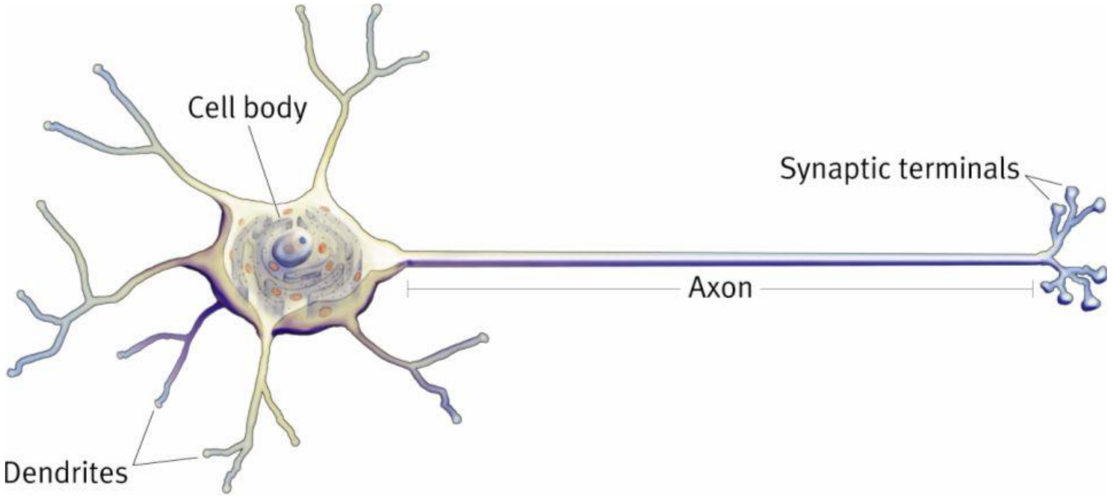
Learning Algorithms

- Artificial Neural Network (ANN)
- Decision Tree (DT)
- Random Forests (RF)
- Support Vector Machine (SVM)
- Logistic Regression (LOGRES)
- Naïve Bayes (NB)
- K Nearest Neighbor (KNN)
- …

Learning Algorithms

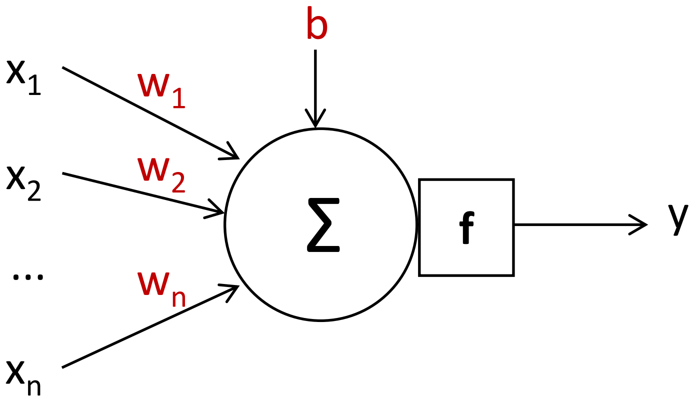
- **Artificial Neural Network (ANN)**
- Decision Tree (DT)
- **Random Forests (RF)**
- Support Vector Machine (SVM)
- Logistic Regression (LOGRES)
- Naïve Bayes (NB)
- K Nearest Neighbor (KNN)
- …

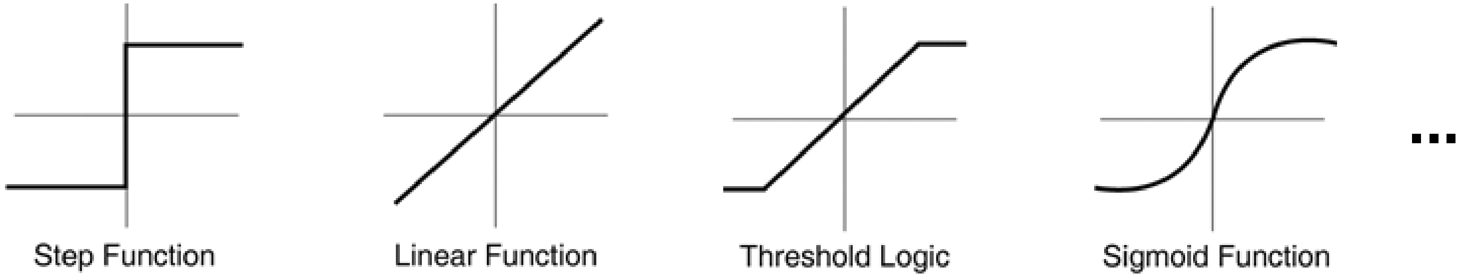
Artificial Neural Network (ANN)



A **neuron** fires if input signal is above a threshold

The activation function **f** can take several shapes

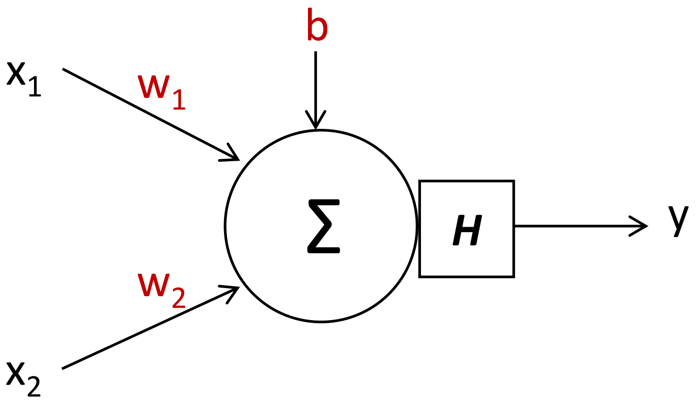




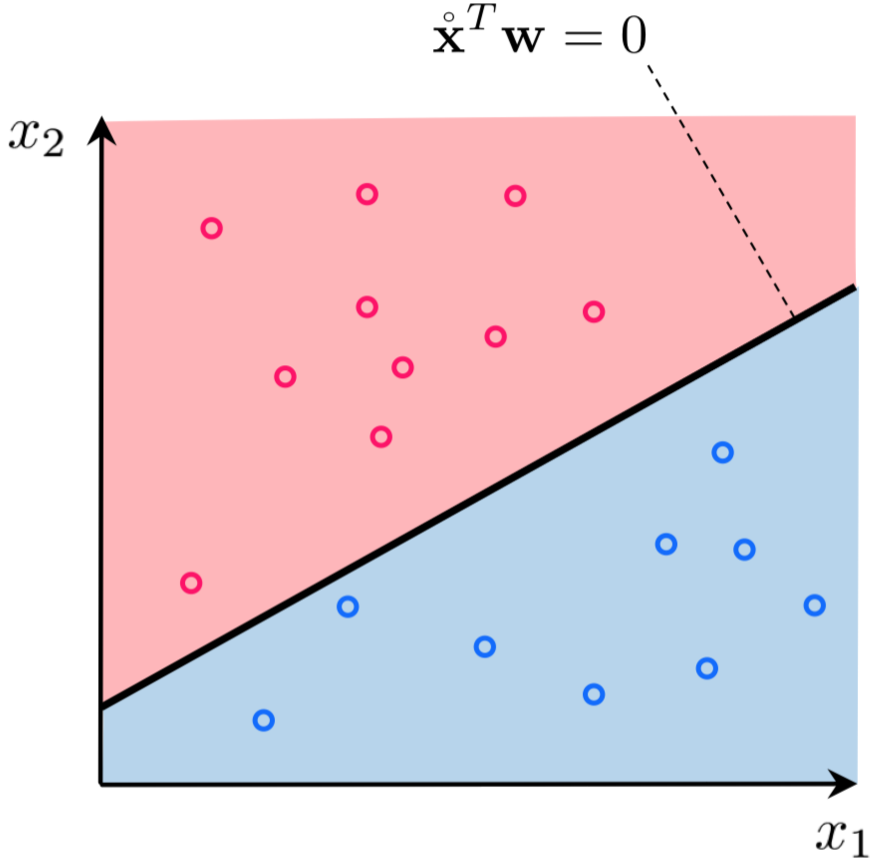
$$\mathbf{f}(w_1x_1 + w_2x_2 + \ldots + w_nx_n + b) = y$$

Artificial Neural Network (ANN)

A single neuron can be used to take simple decisions

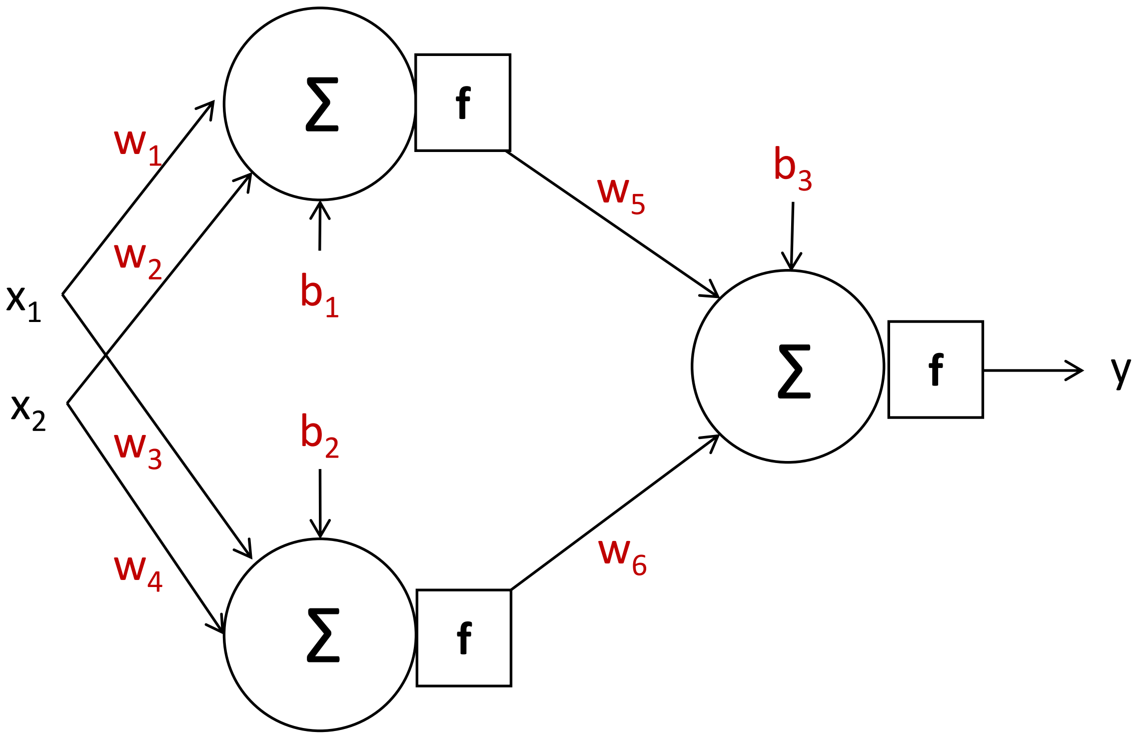


$$H(w_1x_1 + w_2x_2 + b) = y$$



Artificial Neural Network (ANN)

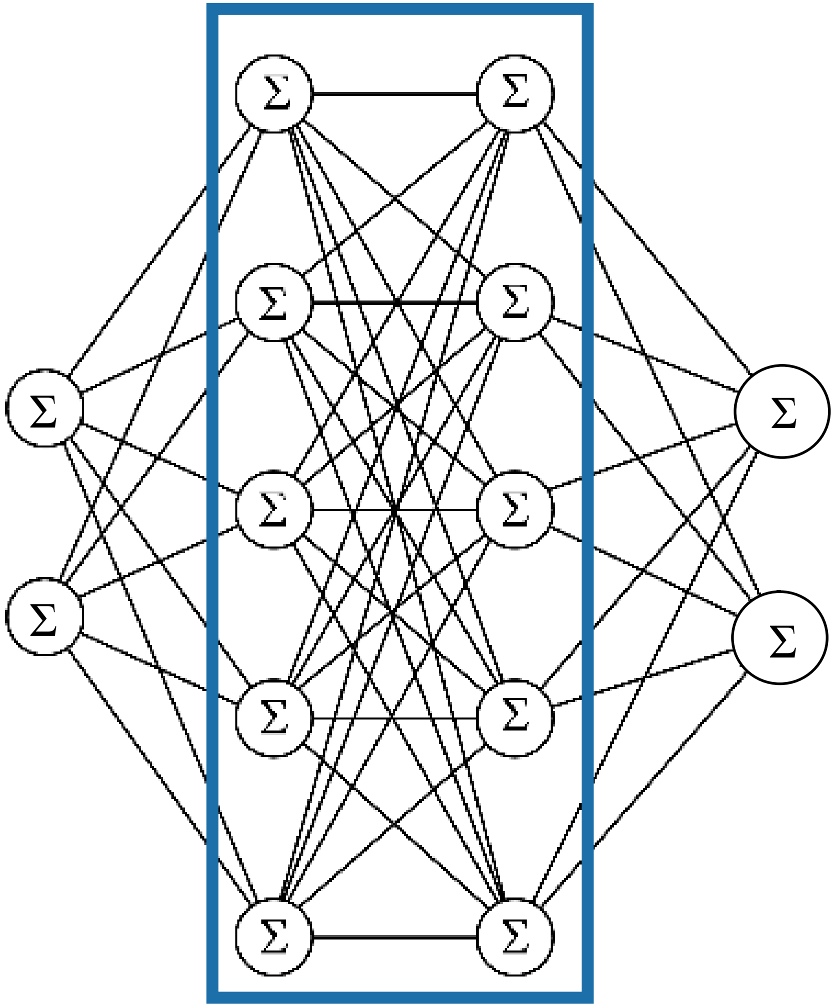
A single neuron can be used to take simple decisions



Complex decision making emerges when arranging neurons into **networks**

Artificial Neural Network (ANN)

A single neuron can be used to take simple decisions

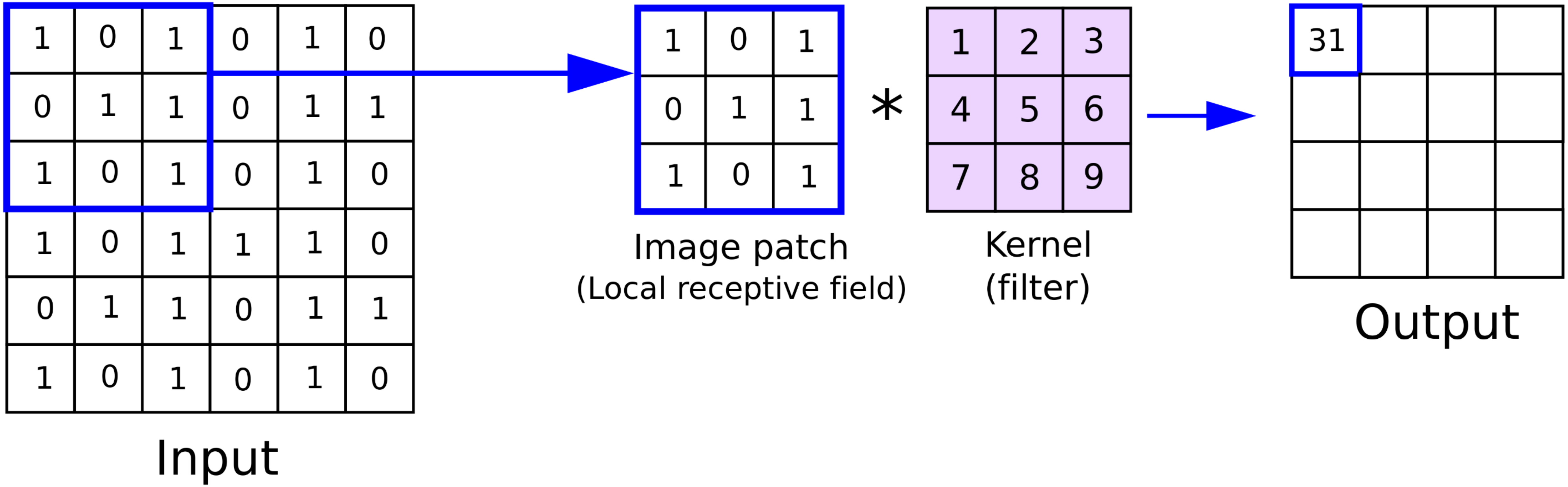


Complex decision making emerges when arranging neurons into **networks**

An ANN with one **hidden layer** can approximate any function

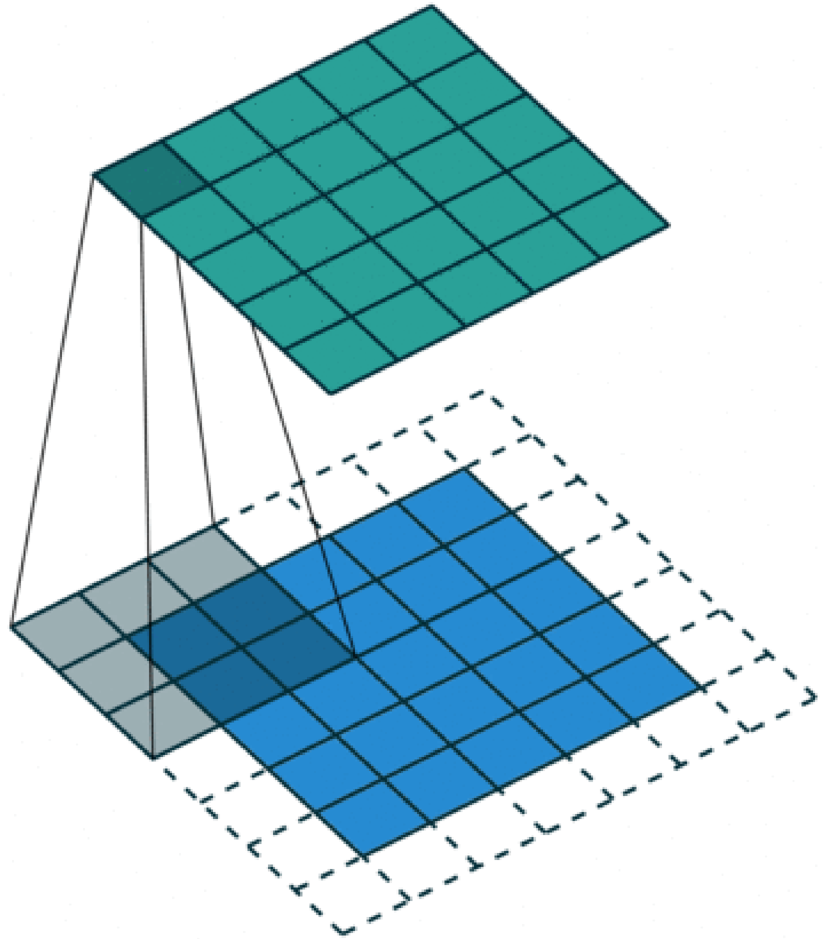
[Extra] Convolutional neural networks (CNN)

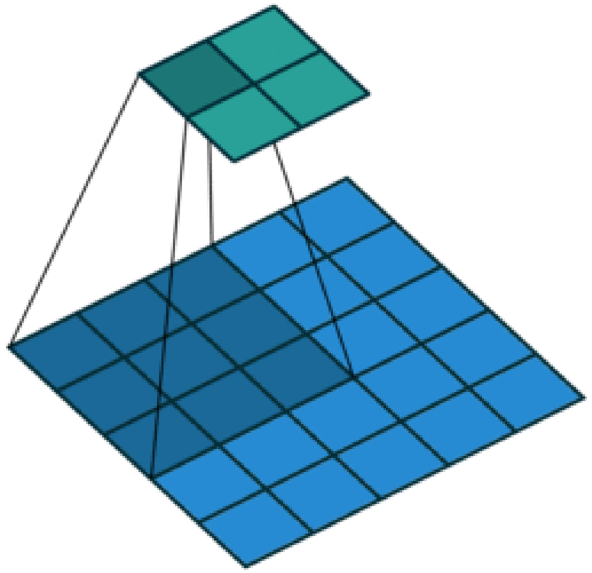
Convolution: a mathematical operation, "sliding a kernel" (filter) over the signal.



[Extra] Convolutional neural networks (CNN)

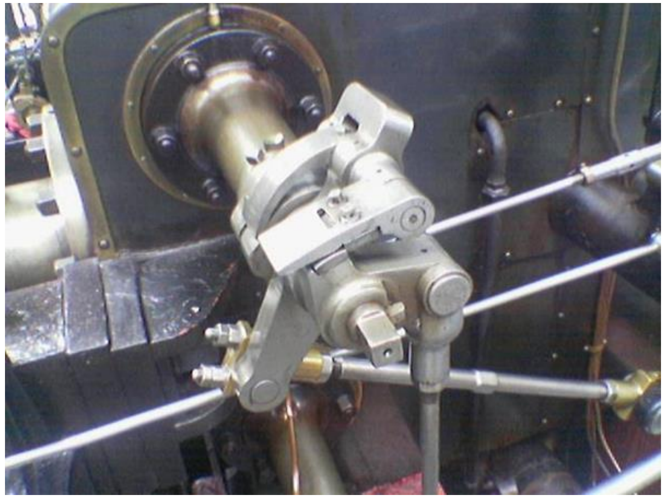
Convolution: a mathematical operation, “sliding a kernel” (filter) over the signal.





[Extra] Convolutional neural networks

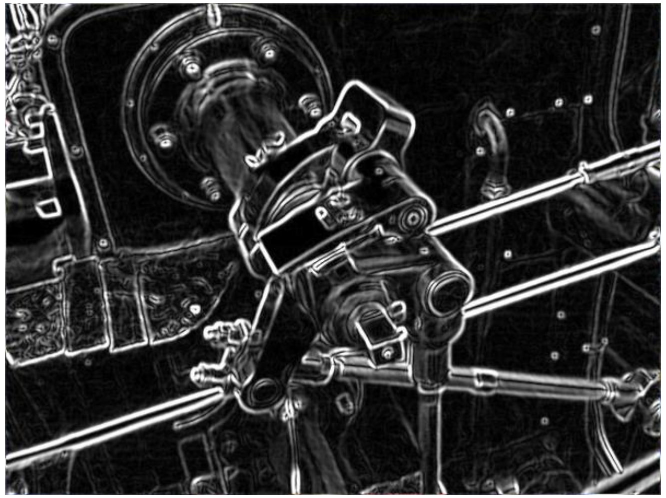
Convolution: a mathematical operation, “sliding a filter” (kernel) over the signal. *Example, edge detection:*



$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A}$$

and

$$\mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

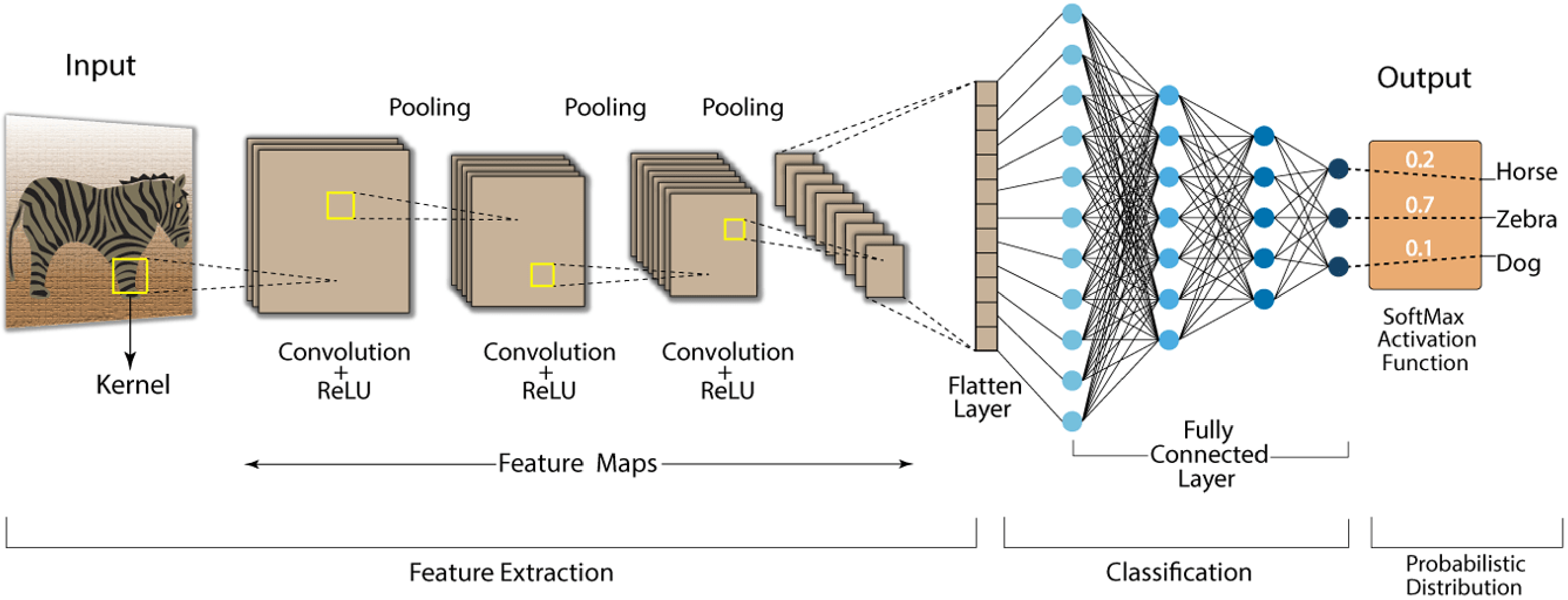


Convolutional neural network: instead of neurons, has many kernels.

Learning = optimising kernel weights

[Extra] Convolutional neural networks

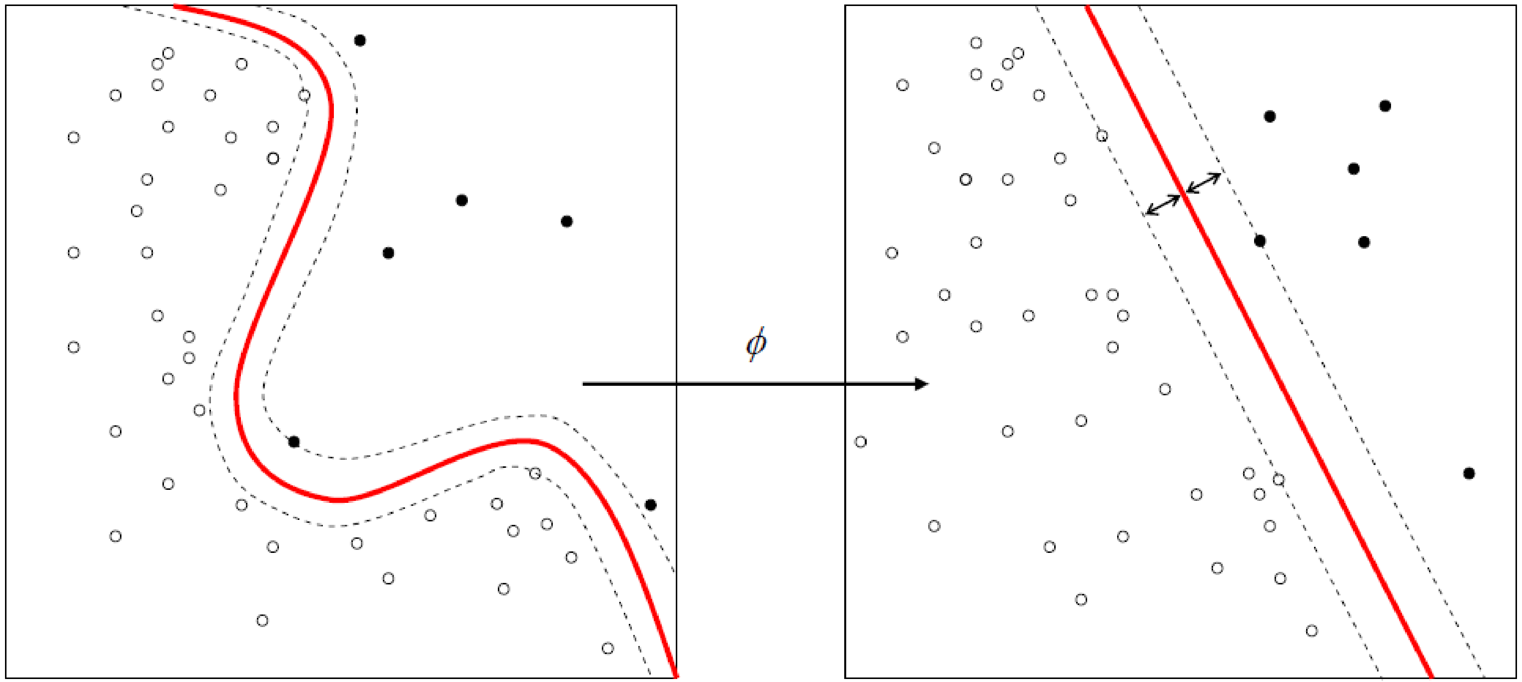
Exploit local correlation in data (e.g. Images, spectra, ...).
Can deal with inputs of arbitrary sizes with less parameters to learn



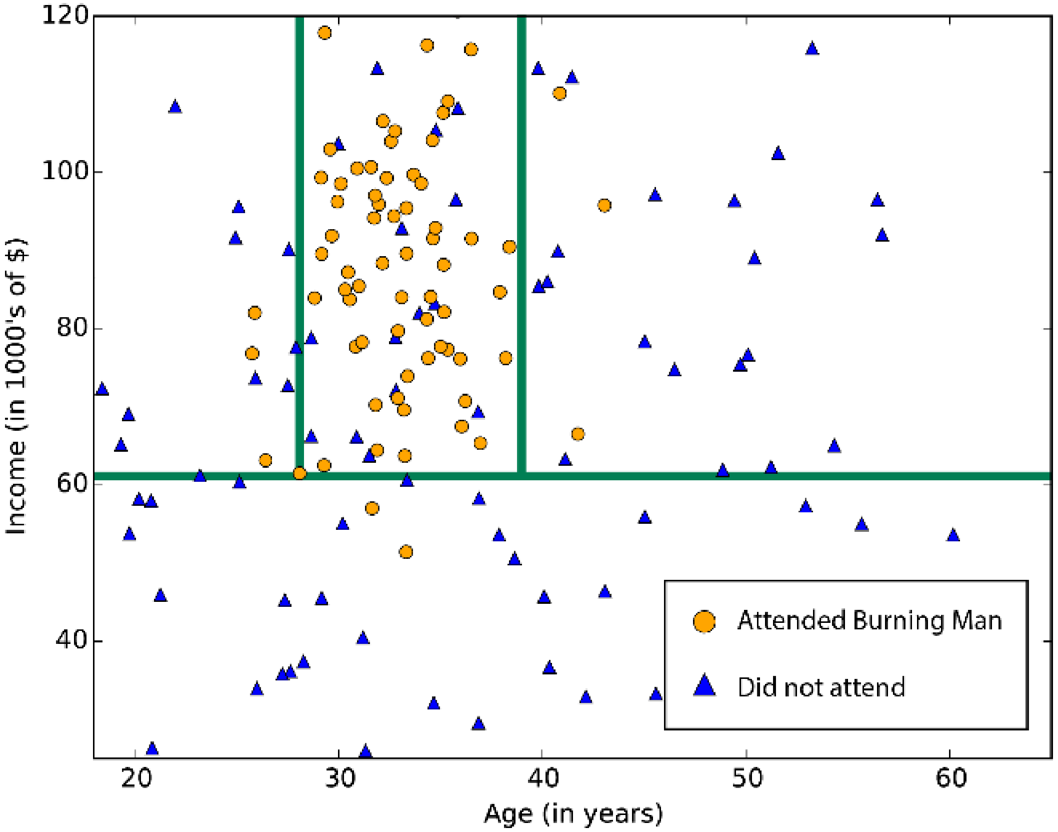
Problem with neural networks: interpretability

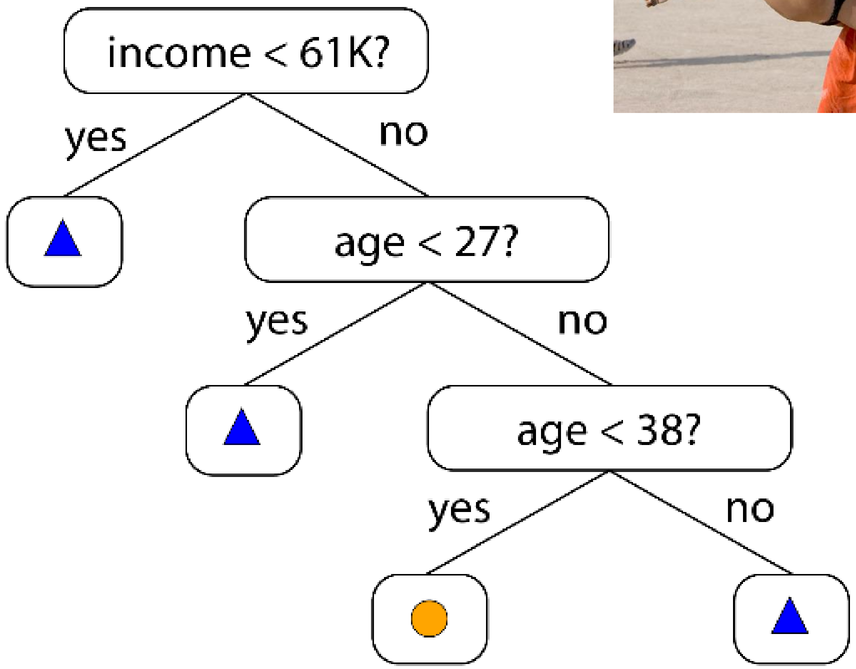
[Extra] Support Vector Machine (SVM)

- invented by V. Vapnik *et al.* in the 1970s in Russia, but only known to the West in 1992
- **linear classifier** finding a hyperplane to separate **two class** of data
- **Kernel functions** (Φ) used for nonlinear separation



Decision Trees (DT)





- Subdivides features space in sectors
- Can overfit if space subdivision becomes too fine

azure.microsoft.com/en-us/documentation/articles/machine-learning-algorithm-choice

Bootstrap Aggregating (Bagging)

a weighted sum of weak classifiers creates a single strong classifier

Useful when a small change to training set causes large change in the output classifier (“learner is unstable”)

training set *D* with *m* examples

Create *N* bootstrap samples S drawing *m* random examples from *D with replacement*

D=	1	2	3	4	5	6	7	8	9	
S[0]=	5	1	7	2	7	9	2	6	5	→C[0]
S[1]=	9	4	7	1	2	8	9	7	6	→C[1]
S[2]=	0	8	2	0	9	7	7	0	1	→C[2]
					…					
S[*N*]=	1	2	3	4	5	6	7	8	9	→C[*N*]

Training: for every S, build a distinct classifier C using the same learning algorithm

L. Breiman, *Bagging Predictors,* in *Machine* Learning, Springer, 1996

[Extra] Boosting

- **a weighted sum of weak classifiers creates a single strong classifier**

- iteratively add classifiers to a pool, tweaked to give more importance to data misclassified by previous classifiers

- Weights based on learners accuracy

Random Forests (RF)

- **Data bagging**: creates *N* decision trees trained on bagged data
- **Feature bagging**: Given *M* features, every tree learns on $m \ll M$ randomly selected features
- Classification based on **voting** of resulting *forest*

Advantages:

- does *not* overfit
- Can handle thousands of features
- estimates what variables are important for classification

L. Breiman, *Random Forests*, Machine Learning, 2001

How do I pick the best learning algorithm?

Learning algorithms quality criteria:

- **accuracy**: percentage of correct classification
- **robustness**: handling noise and missing values
- efficiency: time to construct and use the model
- scalability: efficiency in memory requirements
- interpretability: how much the model is understandable

Accuracy and robustness benchmark (1)

R. Caruana *et al.* systematically tested learning algorithms against different datasets

PROBLEM	#ATTR	TRAIN SIZE	TEST SIZE	%POZ
ADULT	14/104	5000	35222	25%
BACT	11/170	5000	34262	69%
COD	15/60	5000	14000	50%
CALHOUS	9	5000	14640	52%
COV_TYPE	54	5000	25000	36%
HS	200	5000	4366	24%
LETTER.P1	16	5000	14000	3%
LETTER.P2	16	5000	14000	53%
MEDIS	63	5000	8199	11%
MG	124	5000	12807	17%
SLAC	59	5000	25000	50%

Problem	Attr	Train	Valid	Test	%Pos
Sturn	761	10K	2K	9K	33.65
Calam	761	10K	2K	9K	34.32
Digits	780	48K	12K	10K	49.01
Tis	927	5.2K	1.3K	6.9K	25.13
Cryst	1344	2.2K	1.1K	2.2K	45.61
KDD98	3848	76.3K	19K	96.3K	5.02
R-S	20958	35K	7K	30.3K	30.82
Cite	105354	81.5K	18.4K	81.5K	0.17
Dse	195203	120K	43.2K	107K	5.46
Spam	405333	36K	9K	42.7K	44.84
Imdb	685569	84K	18.4K	84K	0.44

R. Caruana and A. Niculescu-Mizil, *An Empirical Comparison of Supervised Learning Algorithm*, Proceedings of the 23rd International Conference on Machine Learning, 2006

R. Caruana et al., *An Empirical Evaluation of Supervised Learning in High Dimensions*, Proceedings of the 25rd International Conference on Machine Learning, 2008

Accuracy and robustness benchmark (2)

Bootstrap analysis: all methods learn from of a random training subset, and get ranked by accuracy

Not high dimensional

MODEL	1ST	2ND	3RD	4TH	5TH	6TH	7TH	8TH	9TH	10TH
BST-DT	0.580	0.228	0.160	0.023	0.009	0.000	0.000	0.000	0.000	0.000
RF	0.390	0.525	0.084	0.001	0.000	0.000	0.000	0.000	0.000	0.000
BAG-DT	0.030	0.232	0.571	0.150	0.017	0.000	0.000	0.000	0.000	0.000
SVM	0.000	0.008	0.148	0.574	0.240	0.029	0.001	0.000	0.000	0.000
ANN	0.000	0.007	0.035	0.230	0.606	0.122	0.000	0.000	0.000	0.000
KNN	0.000	0.000	0.000	0.009	0.114	0.592	0.245	0.038	0.002	0.000
BST-STMP	0.000	0.000	0.002	0.013	0.014	0.257	0.710	0.004	0.000	0.000
DT	0.000	0.000	0.000	0.000	0.000	0.000	0.004	0.616	0.291	0.089
LOGREG	0.000	0.000	0.000	0.000	0.000	0.000	0.040	0.312	0.423	0.225
NB	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.030	0.284	0.686

High dimensional

AVG	1ST	2ND	3RD	4TH	5TH	6TH	7TH	8TH	9TH	10TH
RF	0.727	0.207	0.054	0.011	0.001	0	0	0	0	0
ANN	0.053	0.172	0.299	0.256	0.119	0.072	0.019	0.011	0	0
BSTDT	0.059	0.228	0.18	0.222	0.18	0.075	0.044	0.012	0.001	0
SVM	0.043	0.195	0.213	0.193	0.156	0.088	0.08	0.031	0.001	0
LR	0.089	0.132	0.073	0.075	0.108	0.177	0.263	0.081	0	0
BAGDT	0.002	0.012	0.109	0.123	0.251	0.284	0.123	0.078	0.016	0
KNN	0.023	0.045	0.051	0.057	0.085	0.172	0.122	0.177	0.258	0.01
BSTST	0.004	0.009	0.021	0.063	0.086	0.109	0.3	0.387	0.02	0
PRC	0	0	0	0	0.013	0.024	0.047	0.222	0.695	0
NB	0	0	0	0	0	0	0	0	0.01	0.99

Conclusion

- Know what algorithms do, what their limitations are, and how their parameters may affect results
- Pick your algorithm depending on the nature of your data
- Better data often beats better algorithms
- Getting started: consider Python!