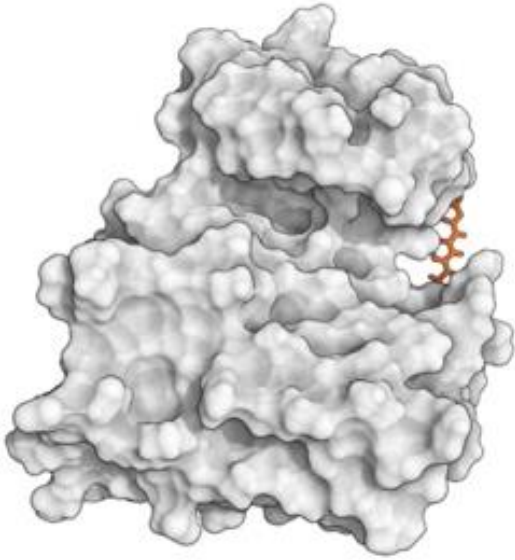
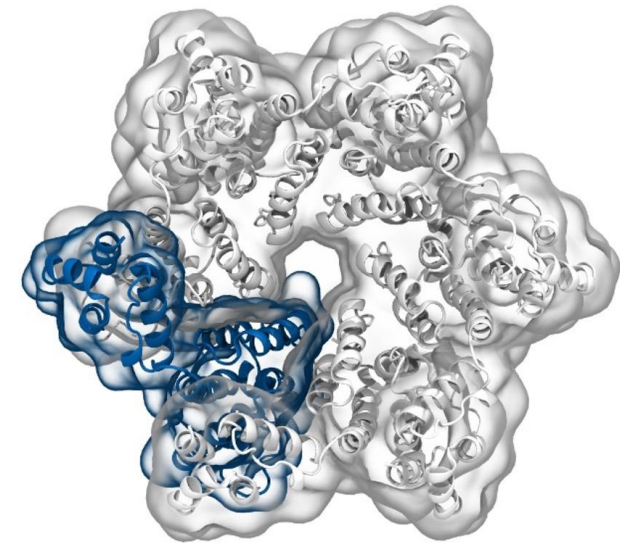


# Simulation of Biomolecules

## Classification

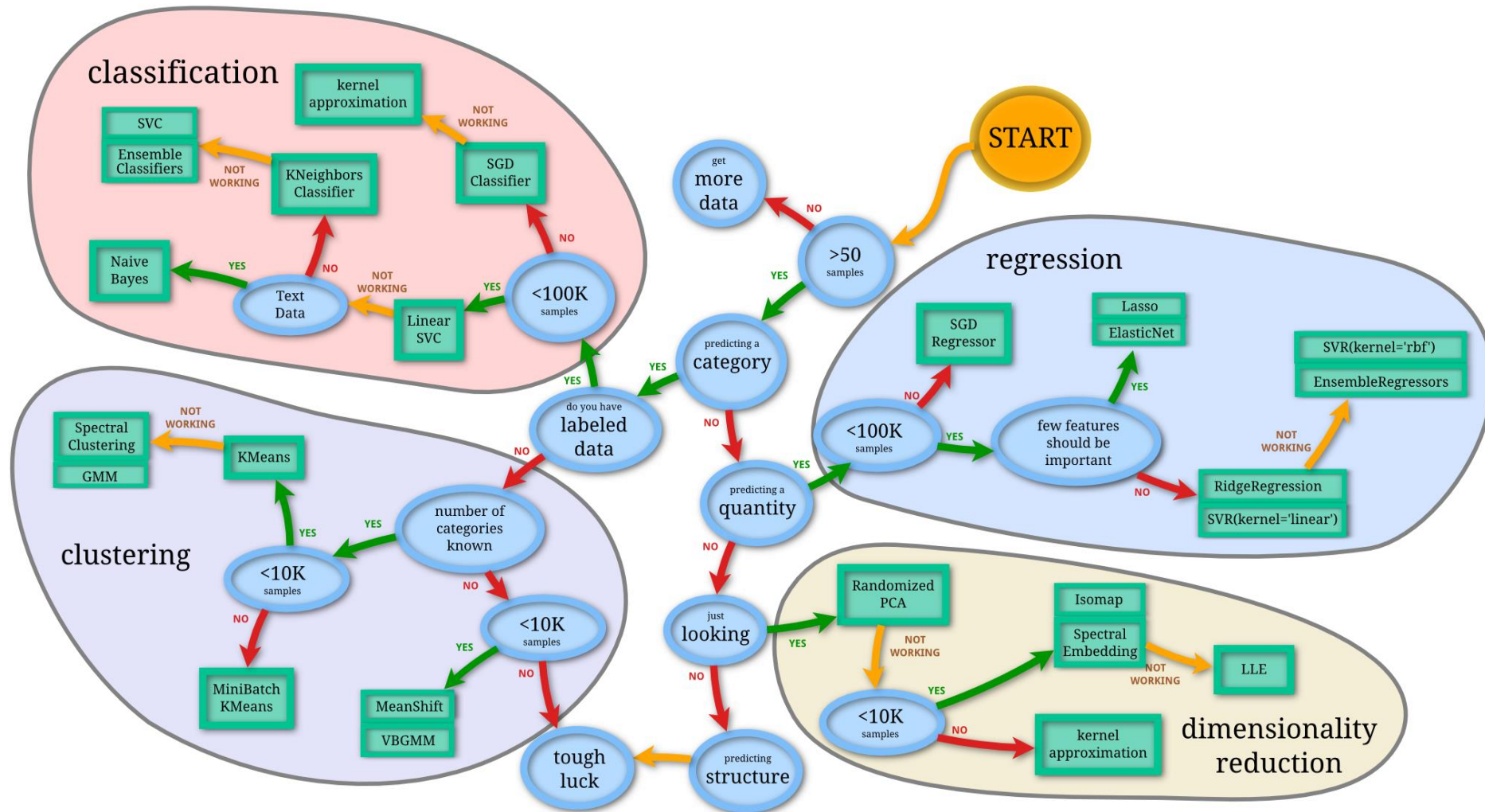


Dr Matteo Degiacomi  
University of Edinburgh  
[matteo.degiacomini@ed.ac.uk](mailto:matteo.degiacomini@ed.ac.uk)

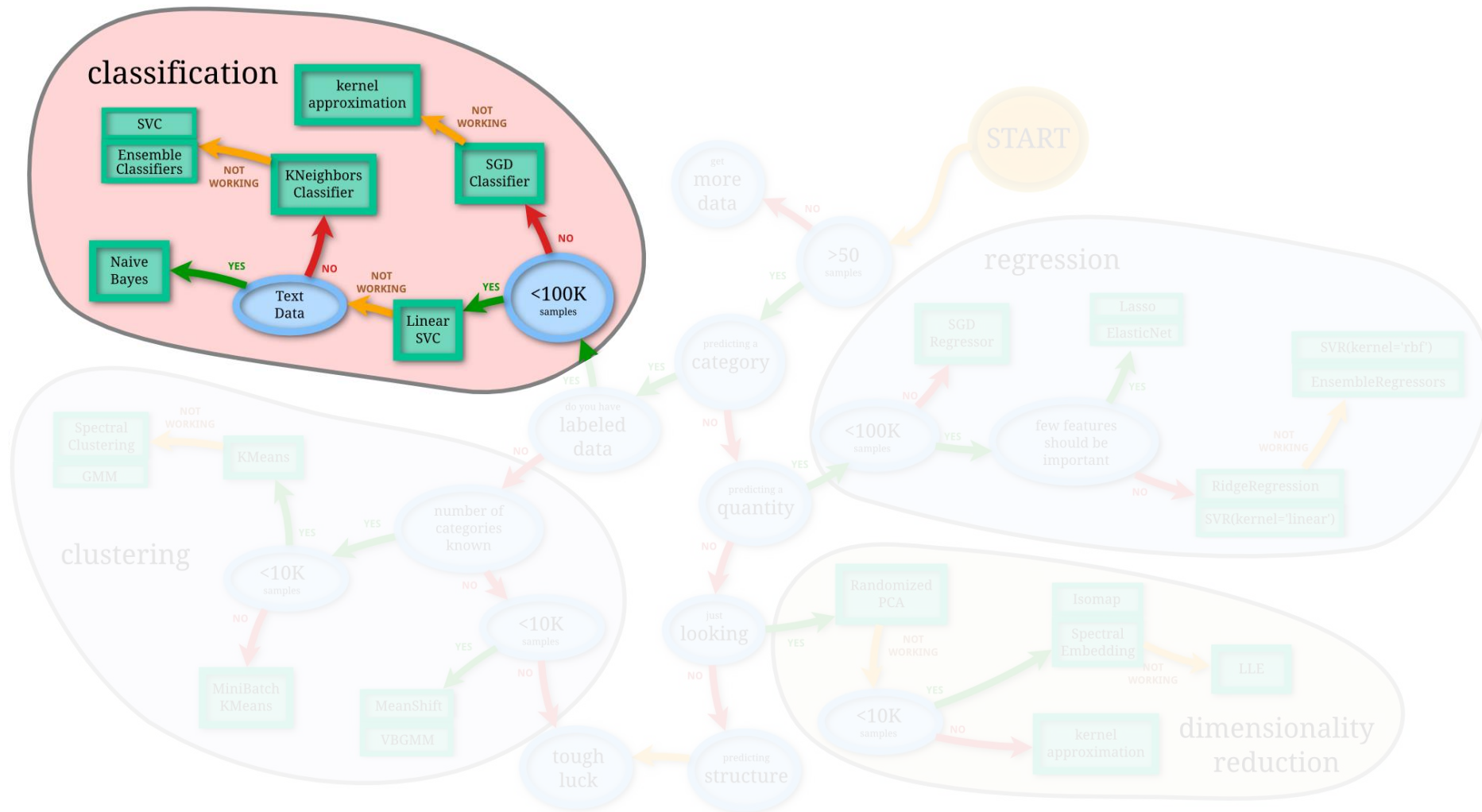


Dr Antonia Mey  
University of Edinburgh  
[antonia.mey@ed.ac.uk](mailto:antonia.mey@ed.ac.uk)

# The Data Mining world

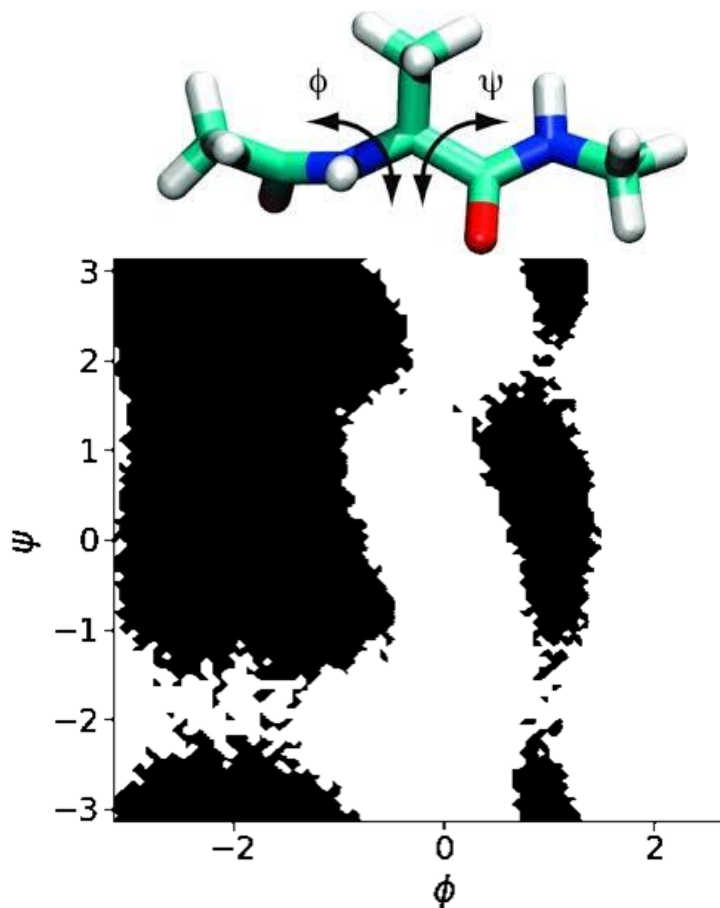
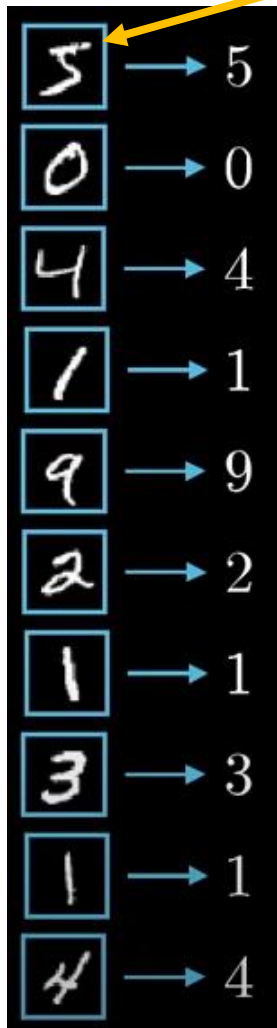


# The Data Mining world

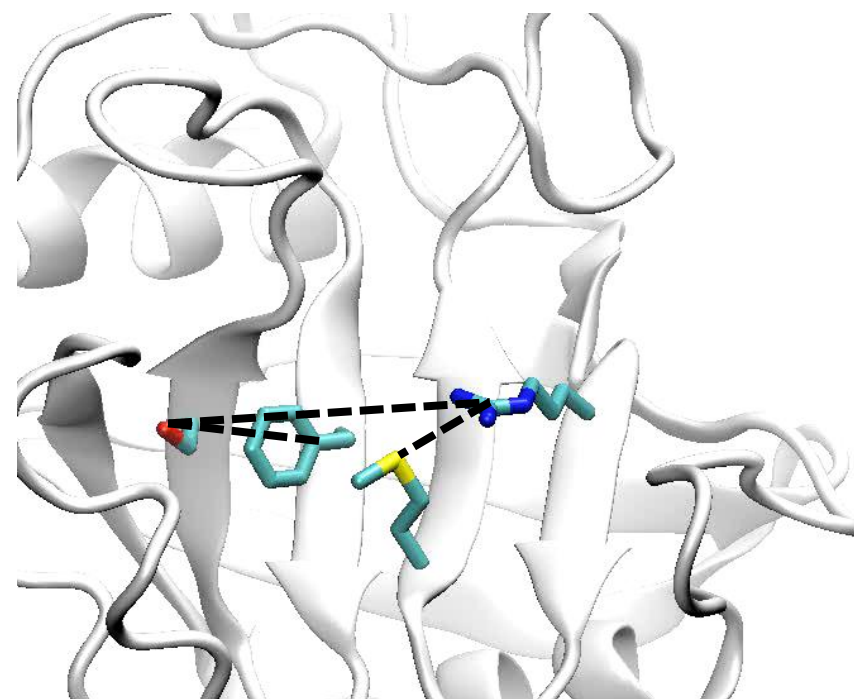


# Features are possible ways to represent data

Pixels colour



Torsional angles

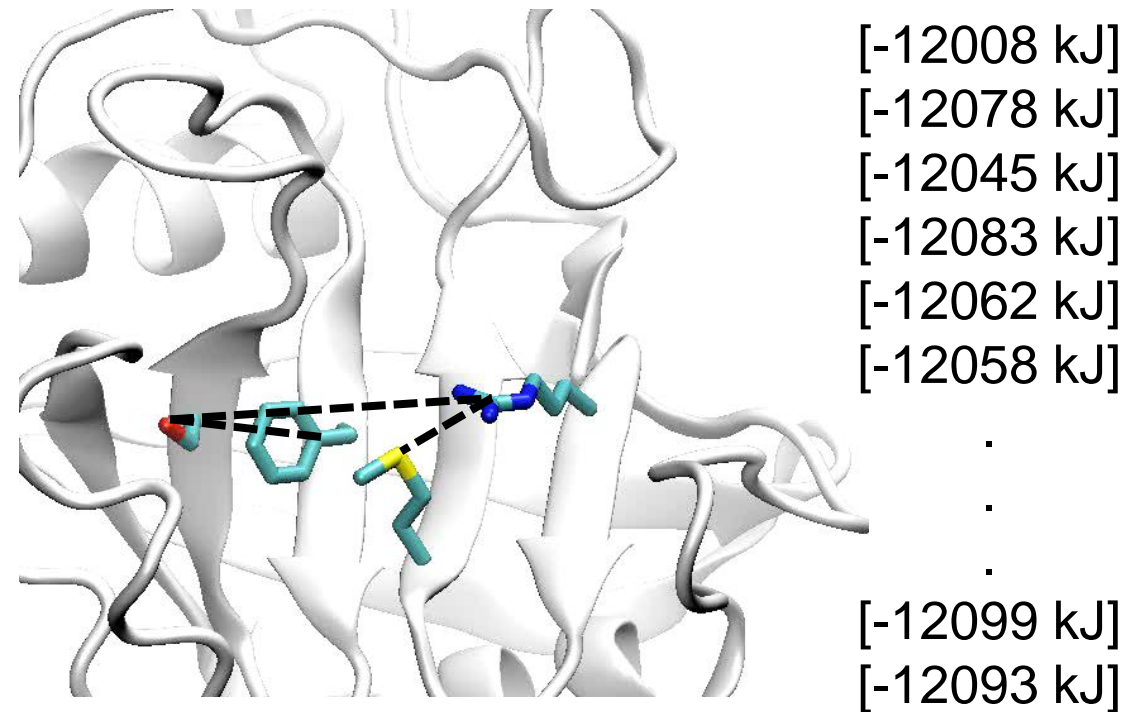
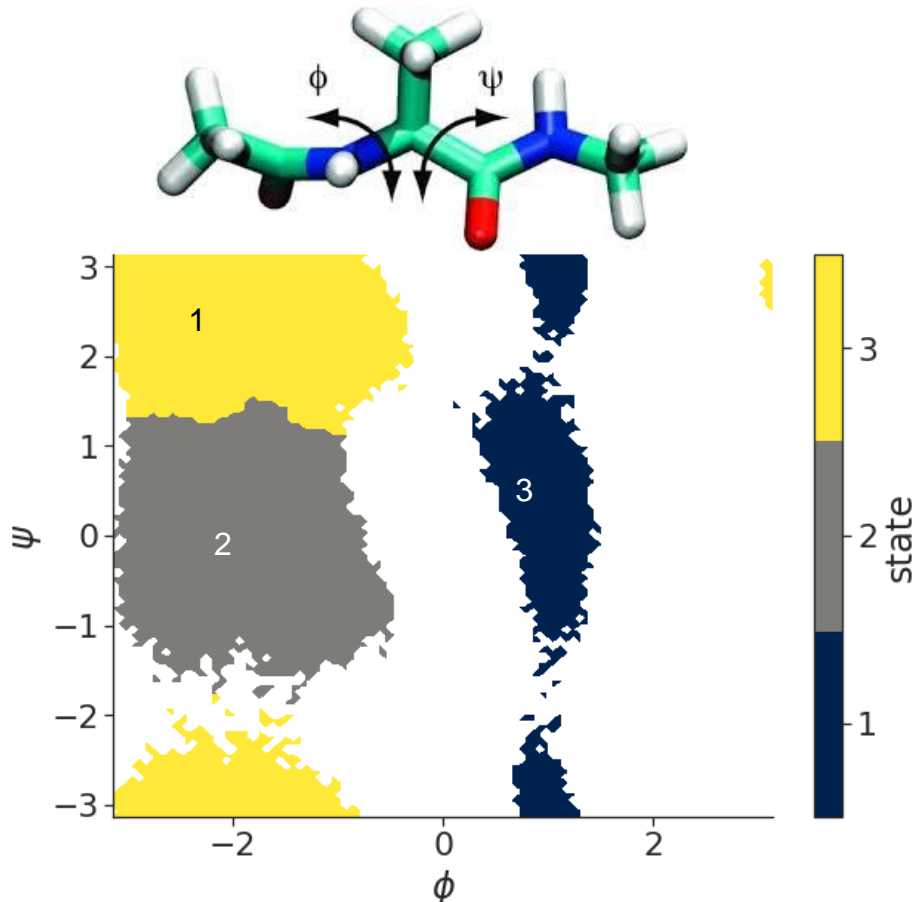
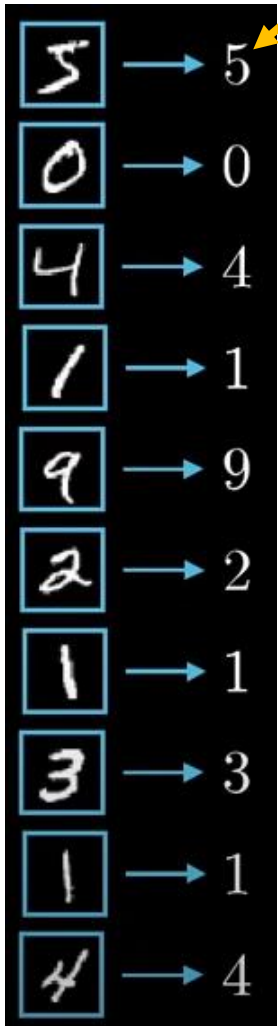


Interatomic distances



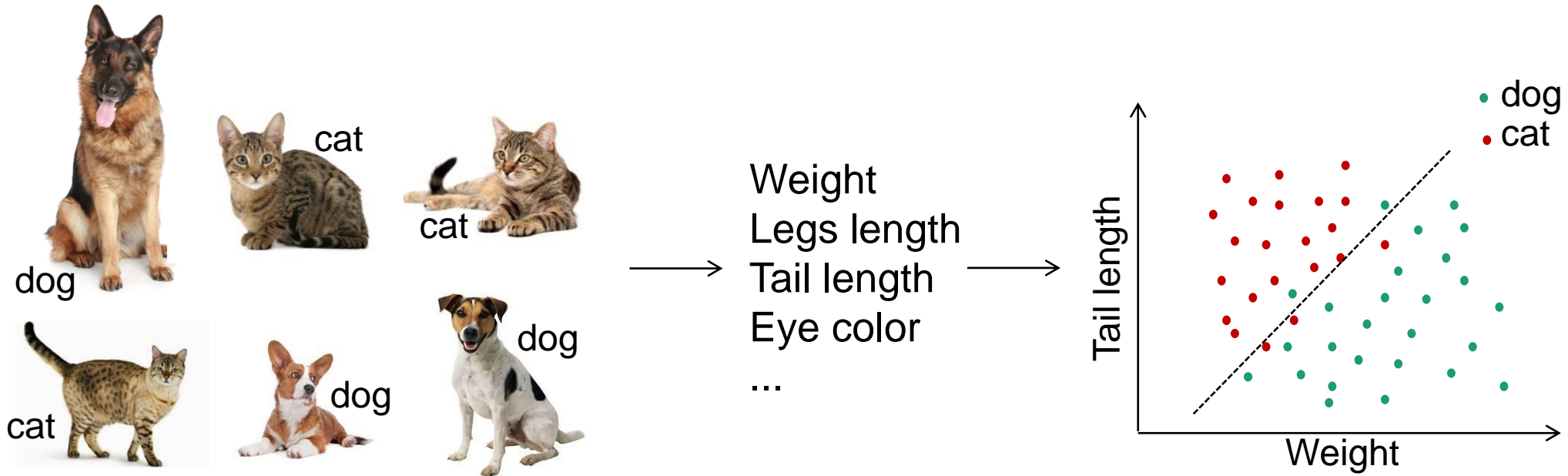
# Labels assign featurised data into categories

Labelled digits



# Data Classification via Supervised Learning

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

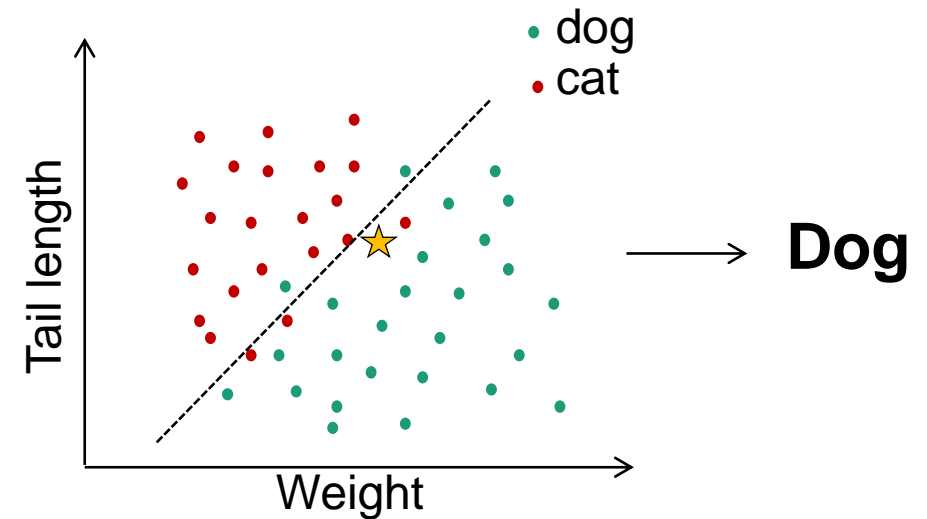


# Data Classification via Supervised Learning

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



→ Weight  
Legs length  
Tail length  
Eye color  
...

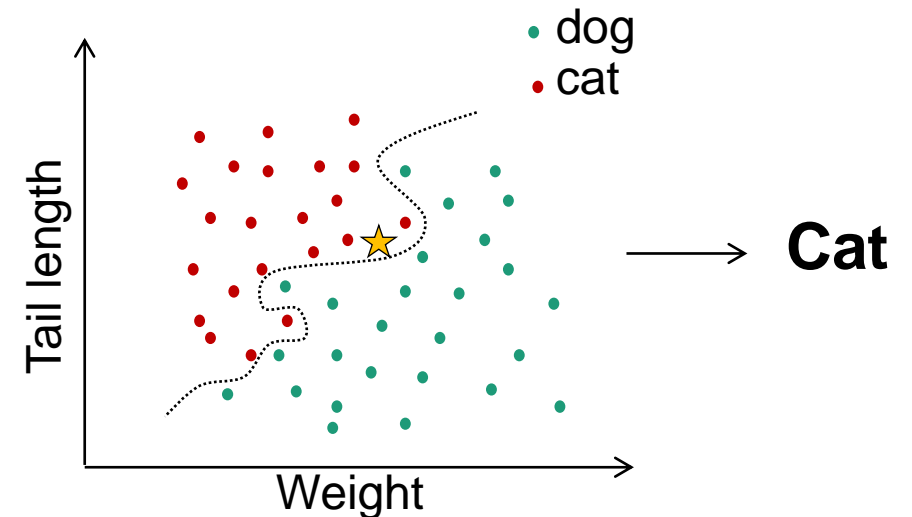


# Data Classification via Supervised Learning

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



→ Weight  
Legs length  
Tail length  
Eye color  
...



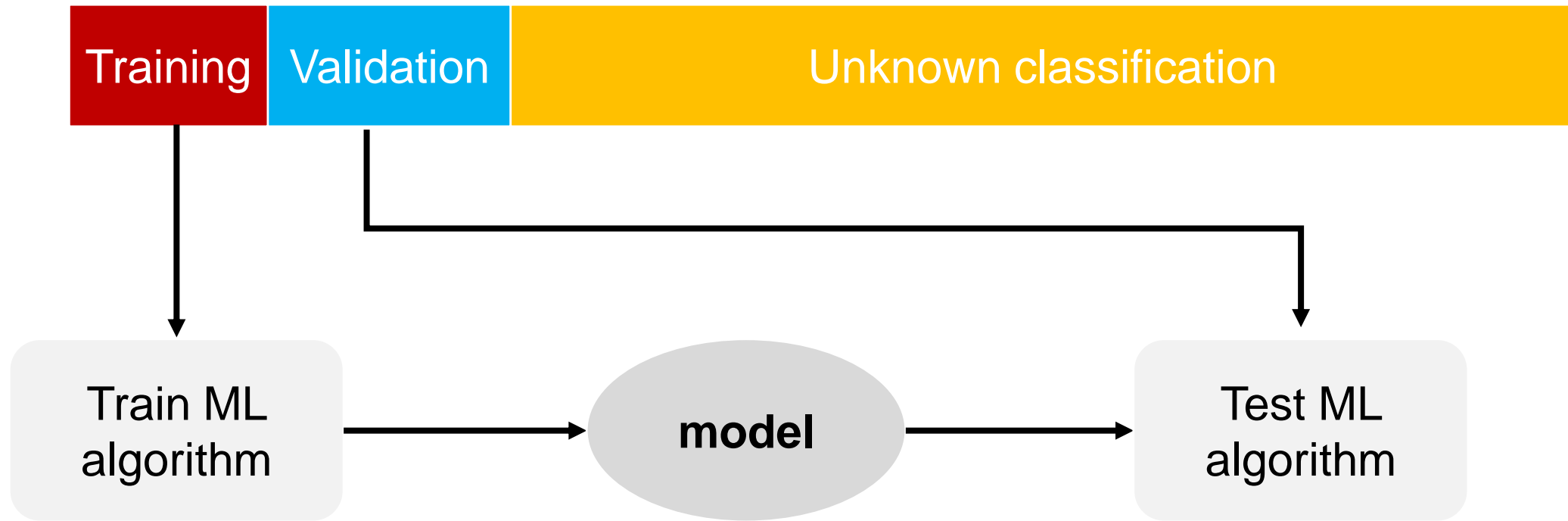


# Data Classification via Supervised Learning

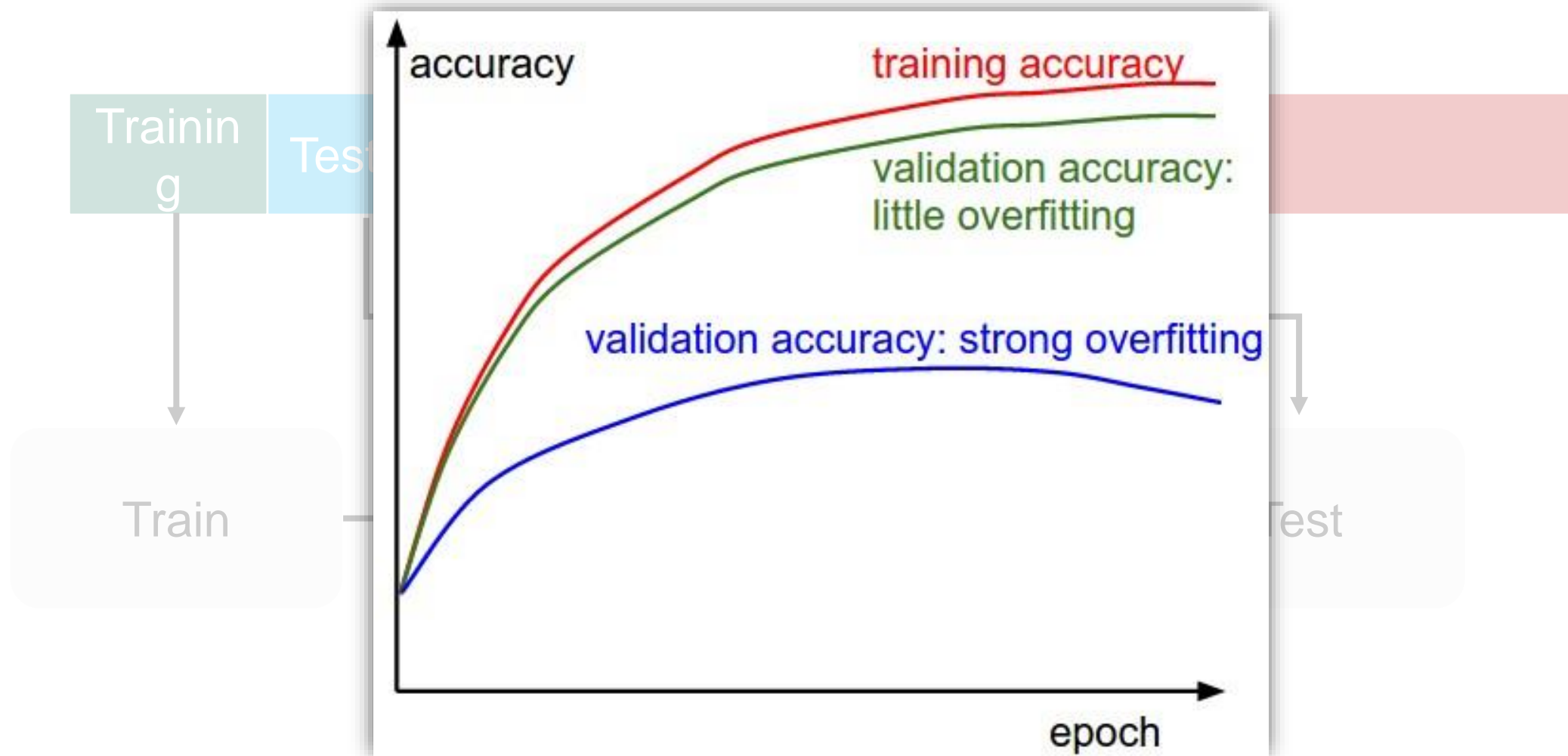


Data

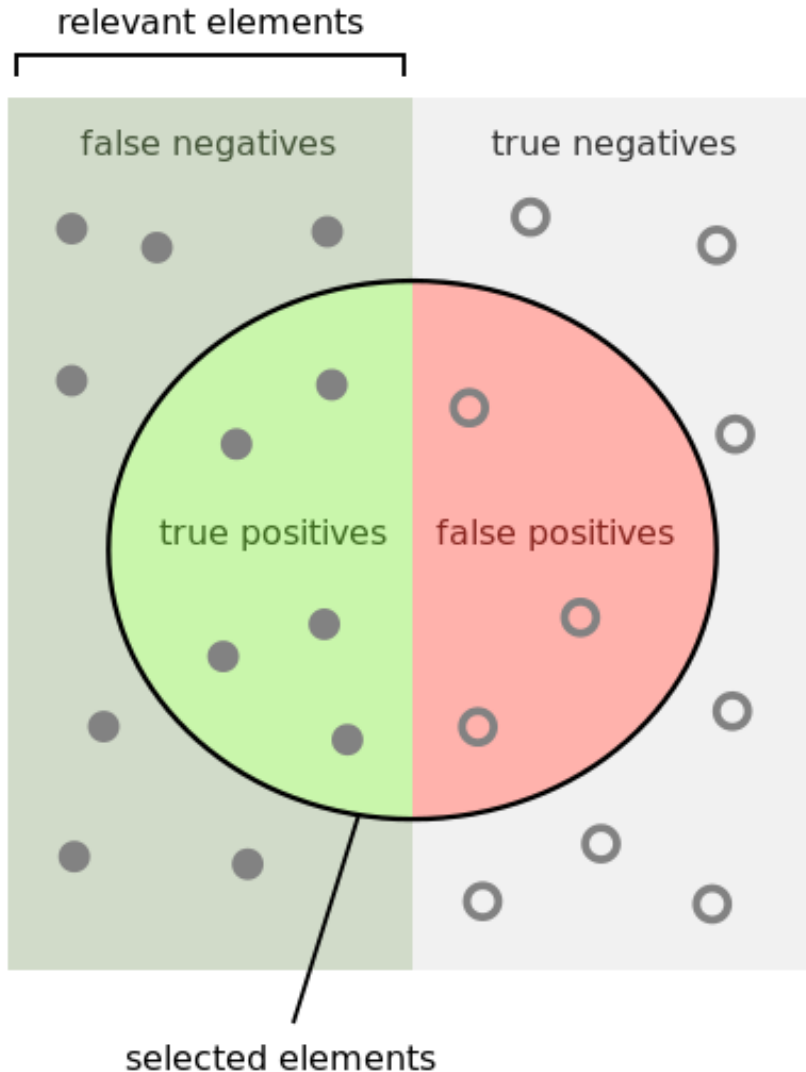
# Data Classification via Supervised Learning



# Data Classification via Supervised Learning





# Quantifying a classifier's performance



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

		real	
		Dog	Cat
result	Dog	90	10
	Cat	12	88

- **precision** =  $\frac{\text{true positives}}{\text{selected elements}}$  = 

- **sensitivity = recall** =  $\frac{\text{true positives}}{\text{relevant elements}}$  = 

- **accuracy** =  $\frac{\text{true positives} + \text{true negatives}}{\text{total population}}$

# Learning Algorithms

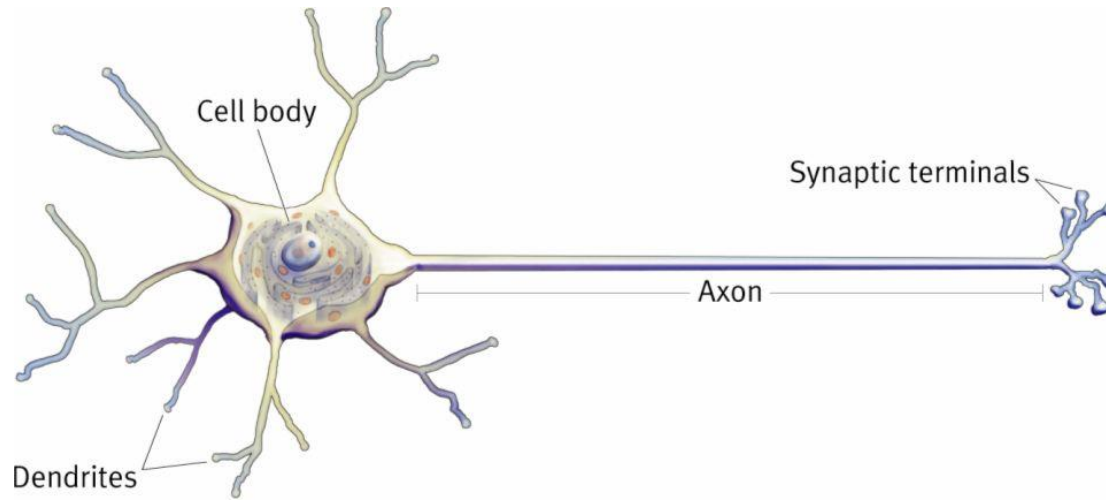
- Artificial Neural Network (ANN)
- Decision Tree (DT)
- Random Forests (RF)
- Support Vector Machine (SVM)
- Logistic Regression (LOGRES)
- Naïve Bayes (NB)
- K Nearest Neighbor (KNN)
- ...



# Learning Algorithms

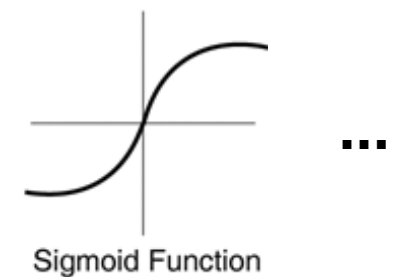
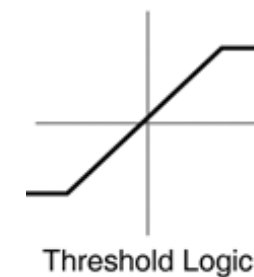
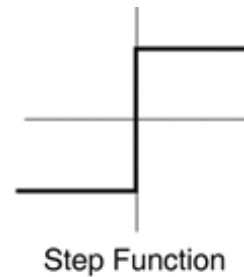
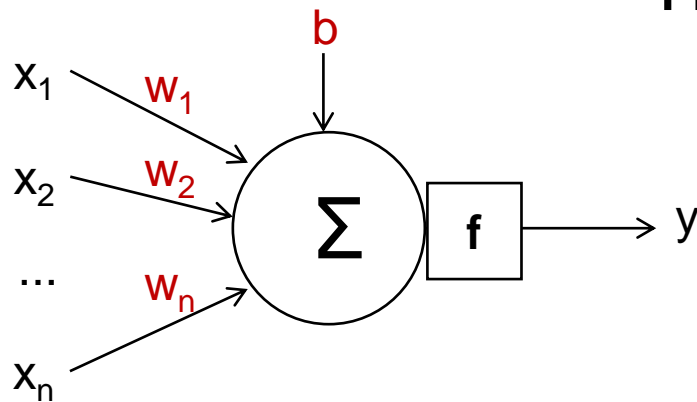
- **Artificial Neural Network (ANN)**
- **Decision Tree (DT)**
- **Random Forests (RF)**
- Support Vector Machine (SVM)
- Logistic Regression (LOGRES)
- Naïve Bayes (NB)
- K Nearest Neighbor (KNN)
- ...

# Artificial Neural Network (ANN)



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

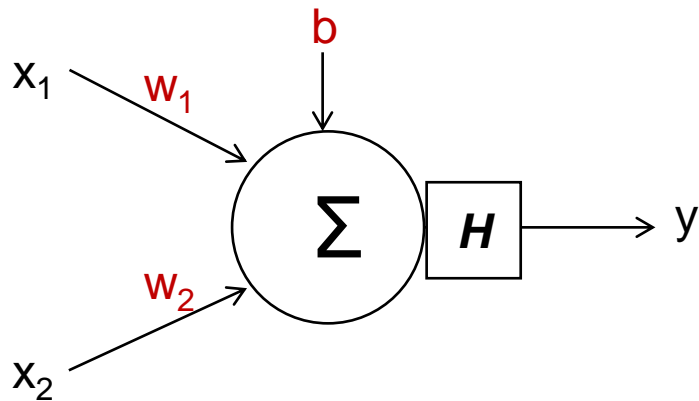
The activation function **f** can take several shapes



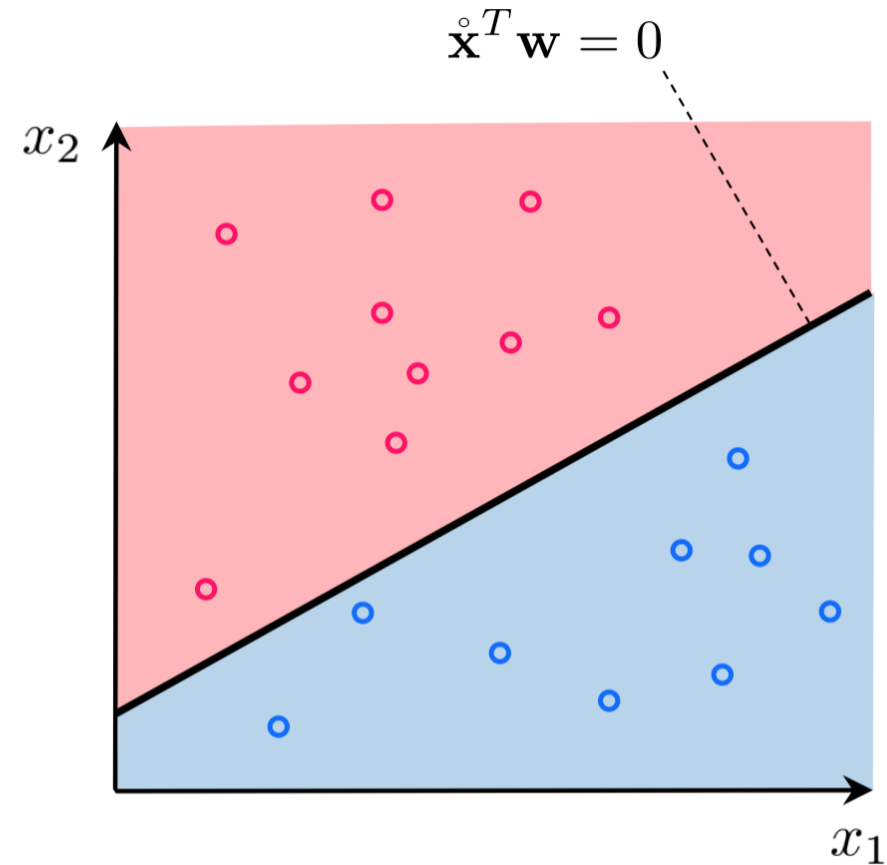
$$\mathbf{f}(w_1x_1 + w_2x_2 + \dots + w_nx_n + b) = y$$

# Artificial Neural Network (ANN)

A single neuron can be used to take simple decisions

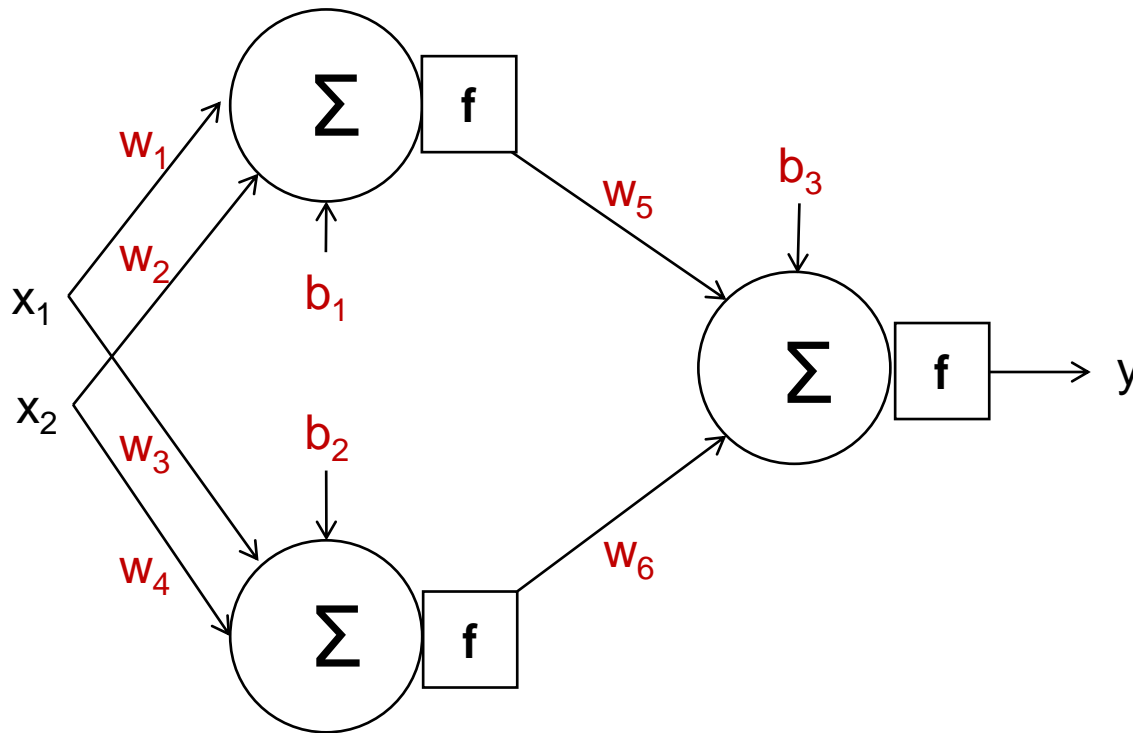


$$H(w_1x_1 + w_2x_2 + b) = y$$



# Artificial Neural Network (ANN)

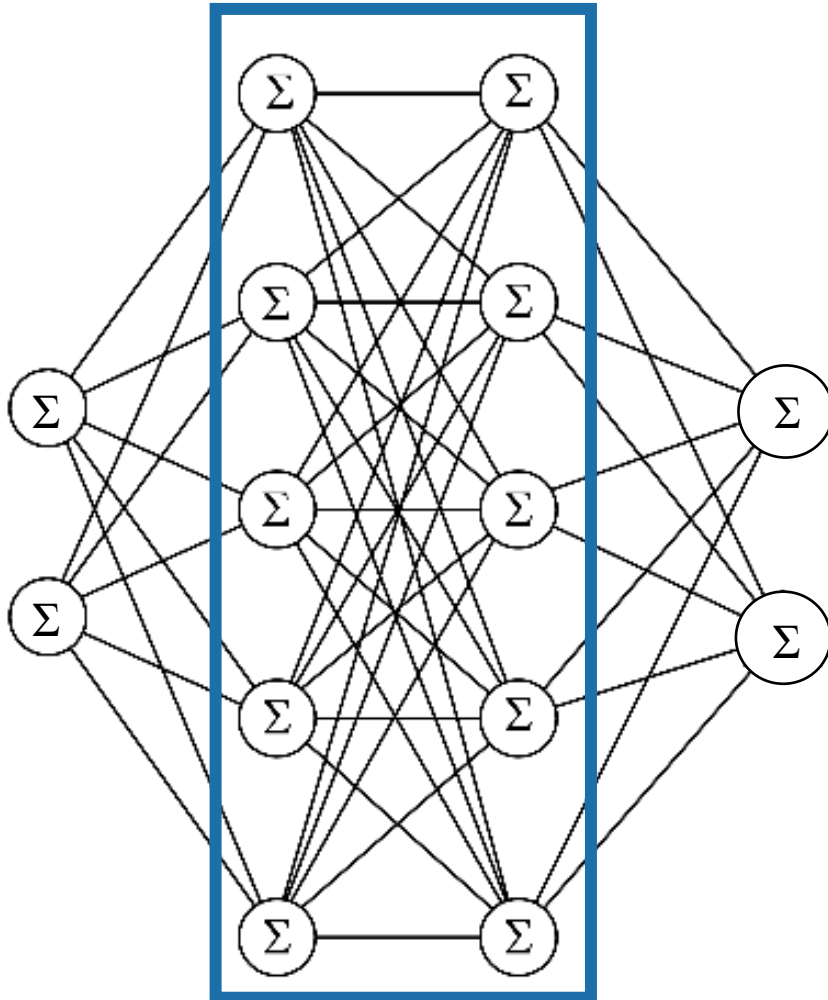
A single neuron can be used to take simple decisions



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

# Artificial Neural Network (ANN)

A single neuron can be used to take simple decisions

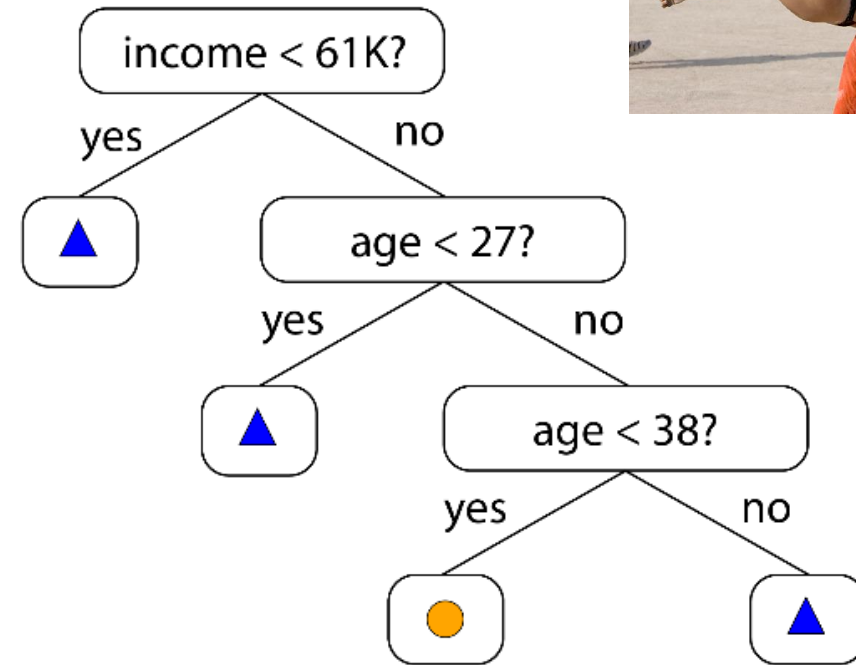
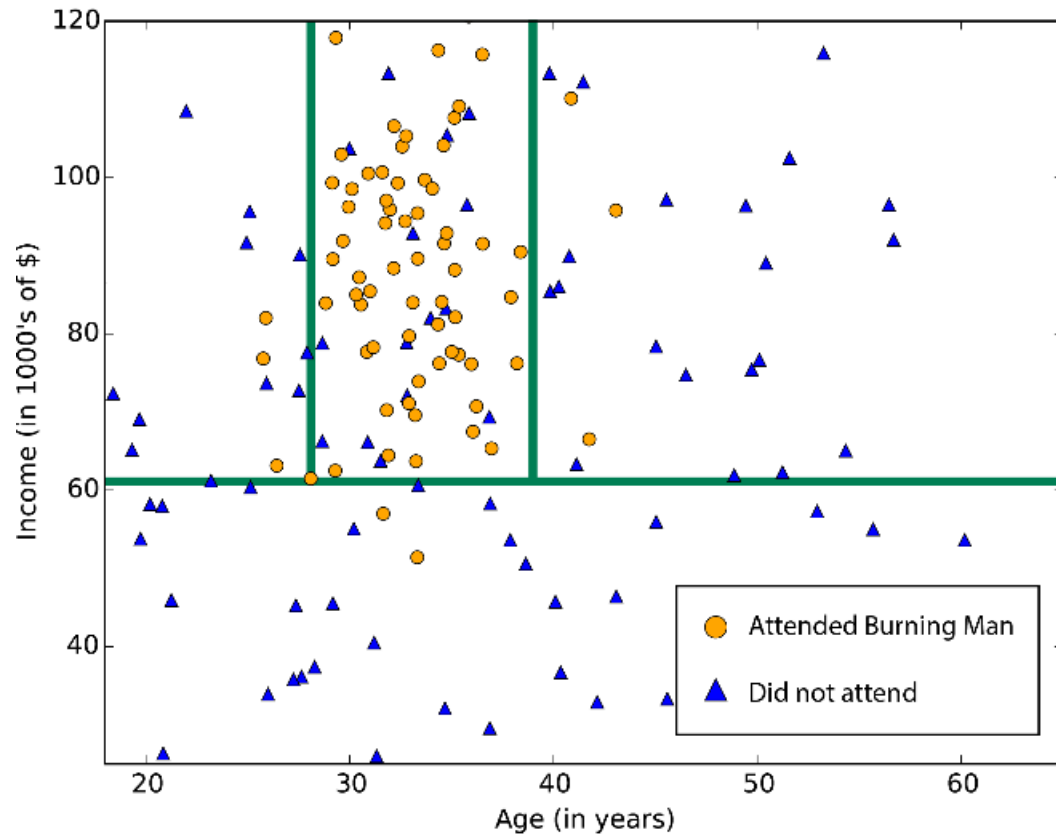


Complex decision making emerges when arranging neurons into **networks**

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



# Decision Trees (DT)



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

# 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

$D =$

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

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

$S[0] =$

5	1	7	2	7	9	2	6	5
---	---	---	---	---	---	---	---	---

$\rightarrow C[0]$

$S[1] =$

9	4	7	1	2	8	9	7	6
---	---	---	---	---	---	---	---	---

$\rightarrow C[1]$

$S[2] =$

0	8	2	0	9	7	7	0	1
---	---	---	---	---	---	---	---	---

$\rightarrow C[2]$

...

$S[N] =$

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

$\rightarrow C[N]$

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

# 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
- **Boosting:** iteratively add classifiers to a pool, tweaked to give more importance to data misclassified by previous classifiers
- Classification based on **voting** of resulting *forest*

## Advantages:

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

# How do I pick the best learning algorithm?

Choice based on what quality criterion is most important, considering the specific task at hand

- **accuracy & precision:** ability of yielding high quality predictions
- **robustness:** handling noise and missing values
- **efficiency:** time needed to construct and use the model
- **scalability:** efficiency in memory requirements
- **interpretability:** how much the model is understandable

# How do I pick the best learning algorithm?

*Example: Neural Networks (**NN**) vs Random Forests (**RF**)*

- Very large dataset? **NN**
- Small dataset? **RF**
- Highly non-linear relationship between features? **NN**
- Interpretability required? **RF**



# 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