Machine Learning Basics

João Campagnolo

Computational Neuroscience class

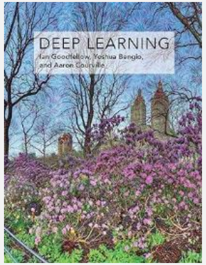
Spring 2024

UNIVERSITY OF COPENHAGEN

Overview & Credits

- Basics of Machine learning
- Types of learning
- Principles of Learning
- k-NN classifier, and k-means clustering

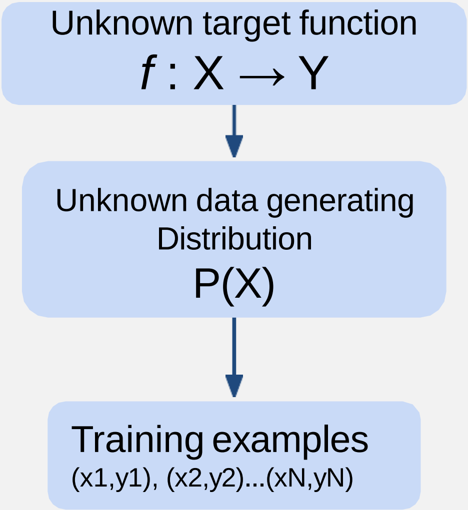
- Course adapted from MLIM lectures (Prof. Raghavendra Selvan & Prof. Erik Dam)
- Based on: @book{Goodfellow-et-al-2016, title={Deep Learning}, author={Ian Goodfellow and Yoshua Bengio and Aaron Courville}, publisher={MIT Press}, note={\url{http://www.deeplearningbook.org}}, year={2016} }



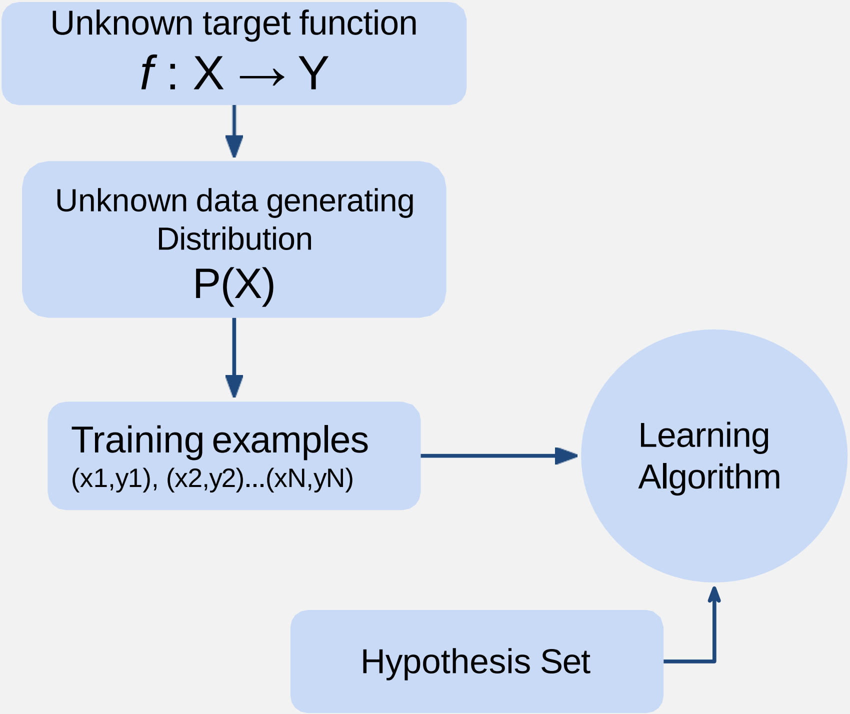
Formalising Machine Learning

Training examples
(x1,y1), (x2,y2)...(xN,yN)

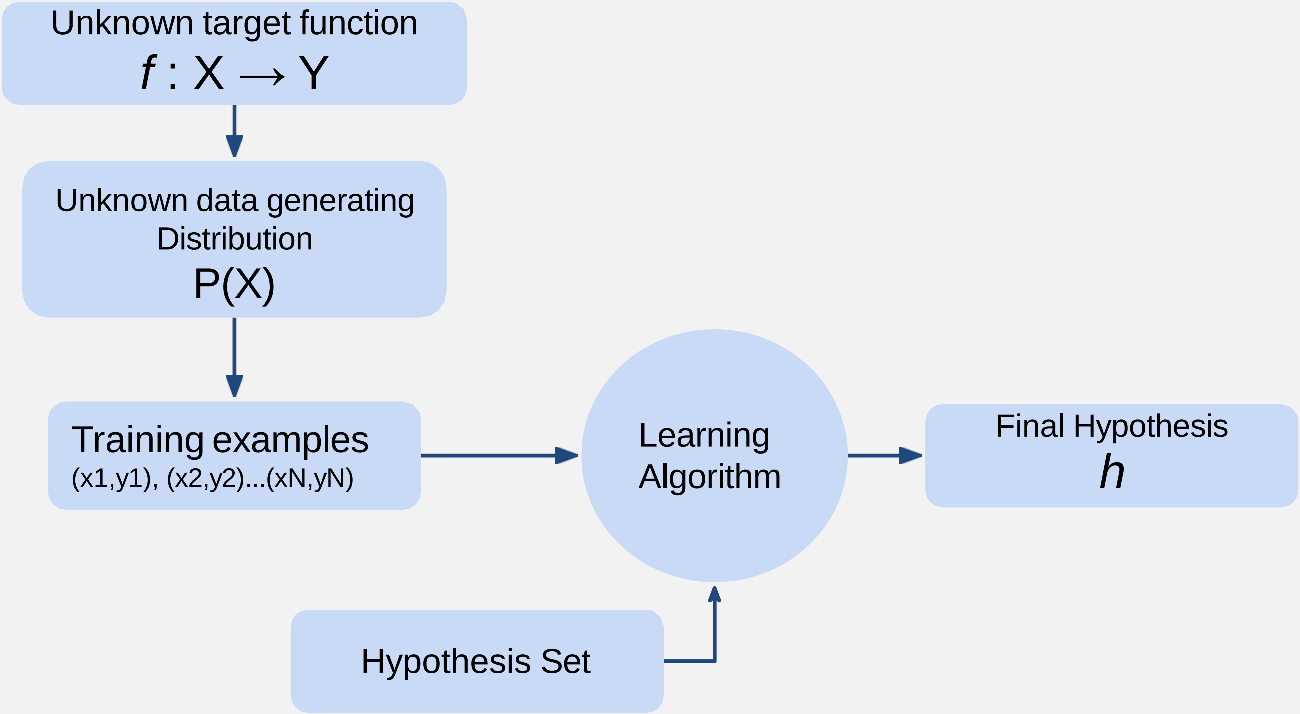
Formalising Machine Learning



Formalising Machine Learning



Formalising Machine Learning



Formalising Machine Learning

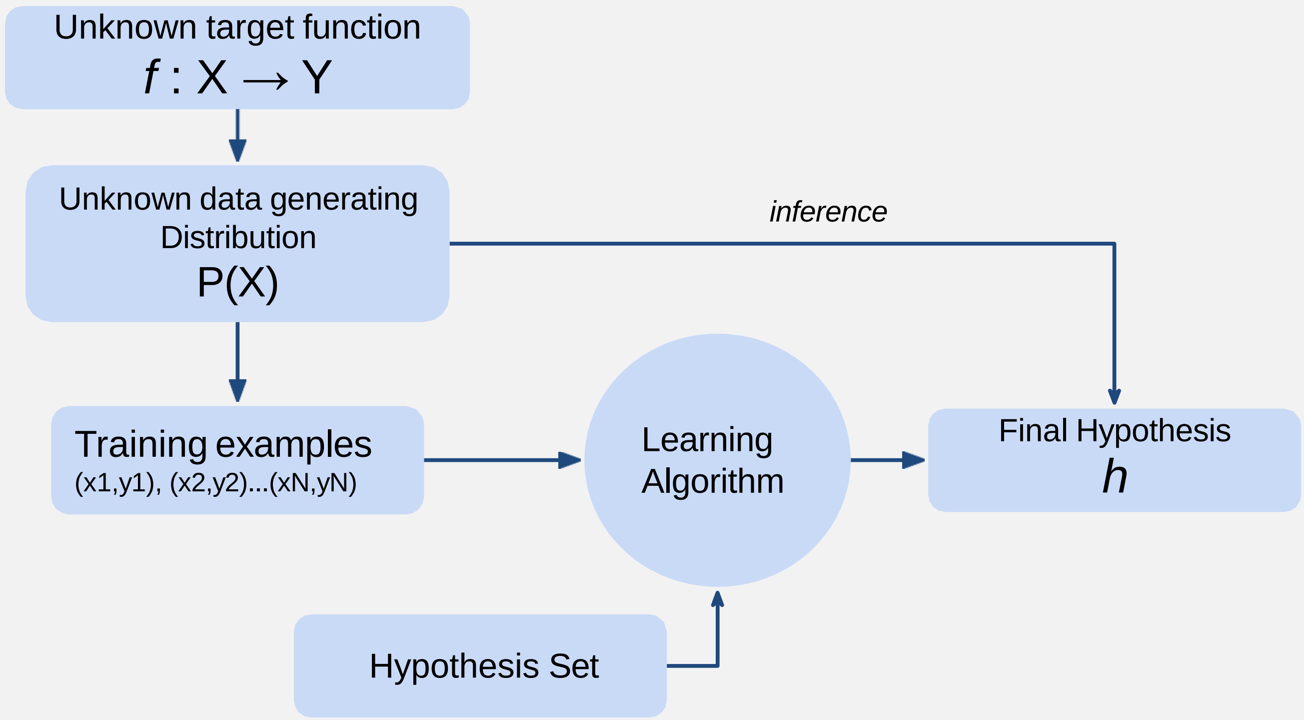
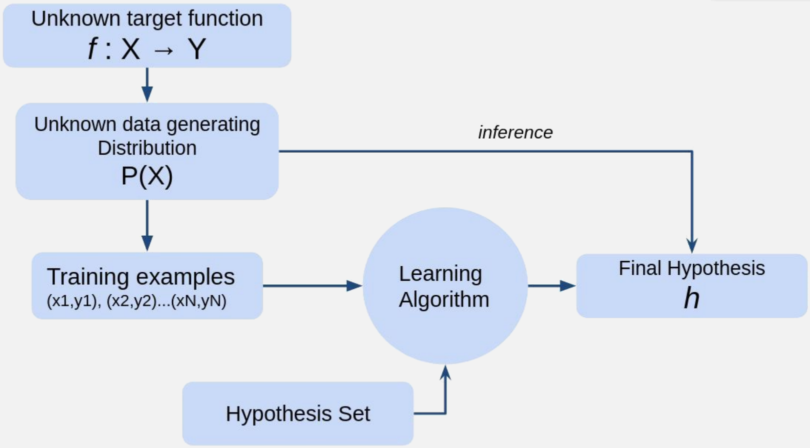


Figure based on Fig.1.9 Mostafa et al.

A learning algorithm

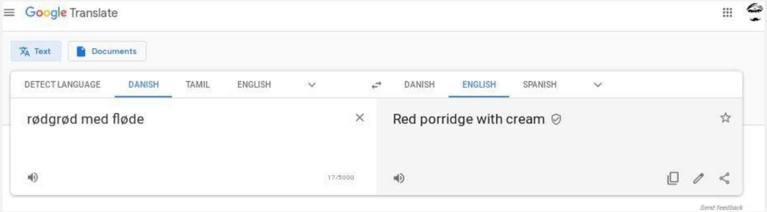
“A computer program is said to learn from experience **E** *with respect to some class of tasks* **T** *and performance measure* **P***, if its performance at tasks in* **T***, as measured by* **P***, improves with experience* **E***.”*

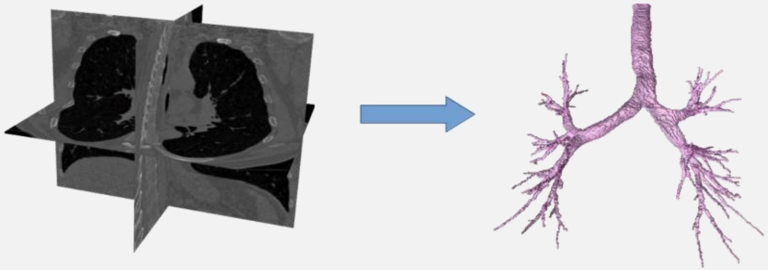
Mitchell, Tom M. Machine learning (1997)



The Task, T

- Classification
- Regression
- Transcription
- Machine translation
- Face recognition
- Anomaly detection
- Synthesis & sampling
- Denoising
- Density estimation
- Self-driving





The Performance measure, P

Not always straightforward but most common:

- Accuracy
- Error rates/ losses (0-1 loss)
- Log probability
- KL divergence

https://thispersondoesnotexist.com/

http://www.thisworddoesnotexist.com/

The Experience, **E**

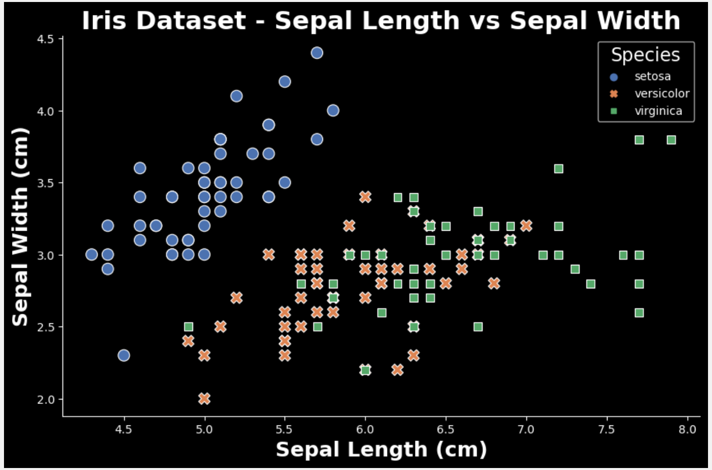
All the ways information can enter the model primarily as:

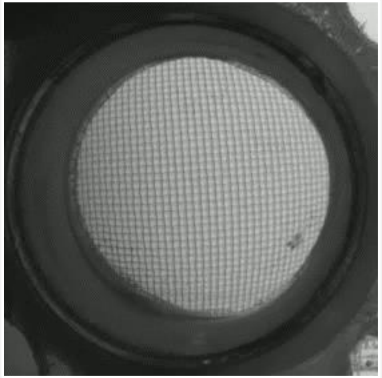
- Prior information
- Hyper-parameters
- Data/ supervision

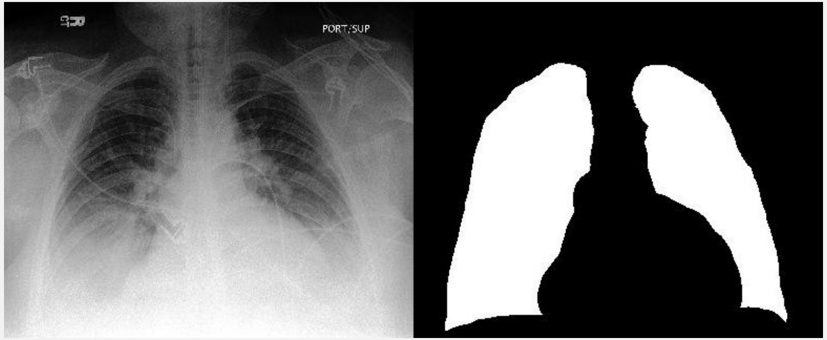
More concrete classification of ML methods is based on **E**

Supervised Learning

- Strong labels for the entire dataset
- (Relatively) Easy to train
- Hard to obtain high quality labels
- Ex: Image Segmentation

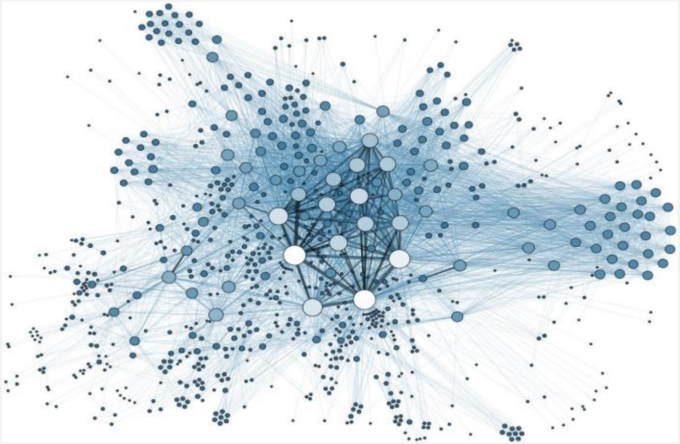


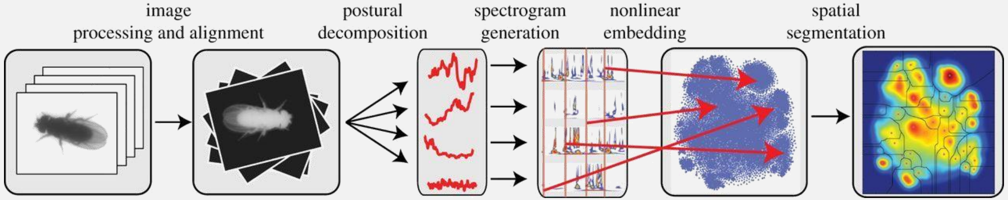


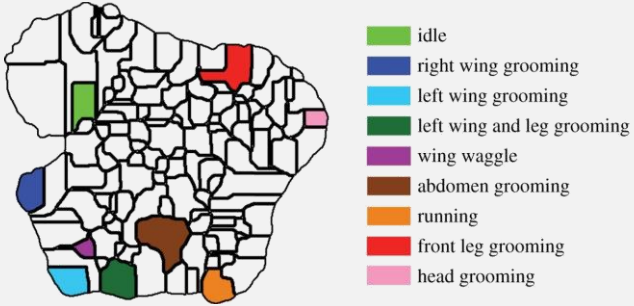


Unsupervised learning

- No labels.
- “Figure it out yourself” model
- Ex: Social networks, Gene expression networks







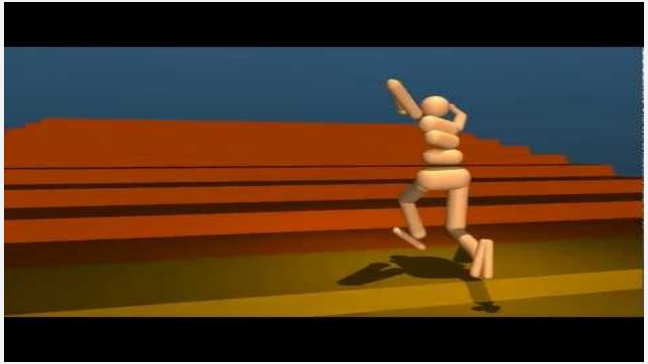
Berman et al., 2014

Semi-supervised learning

- Strong labels for some of the data
- Weak labels for all of the data
- Can be useful in cases where strong labels are hard!
- Ex: Captcha

Reinforcement learning

- Combination of strong and weak labels
- Online learning
- Constant learning
- Ex: Streaming services recommendation



More….

- Self-supervised
- Active learning
- Continual learning
- Meta-learning
- …...

We will focus on supervised and unsupervised learning methods.

Formulate your learning task

- Task **T**
- Performance **P**
- Experience **E**

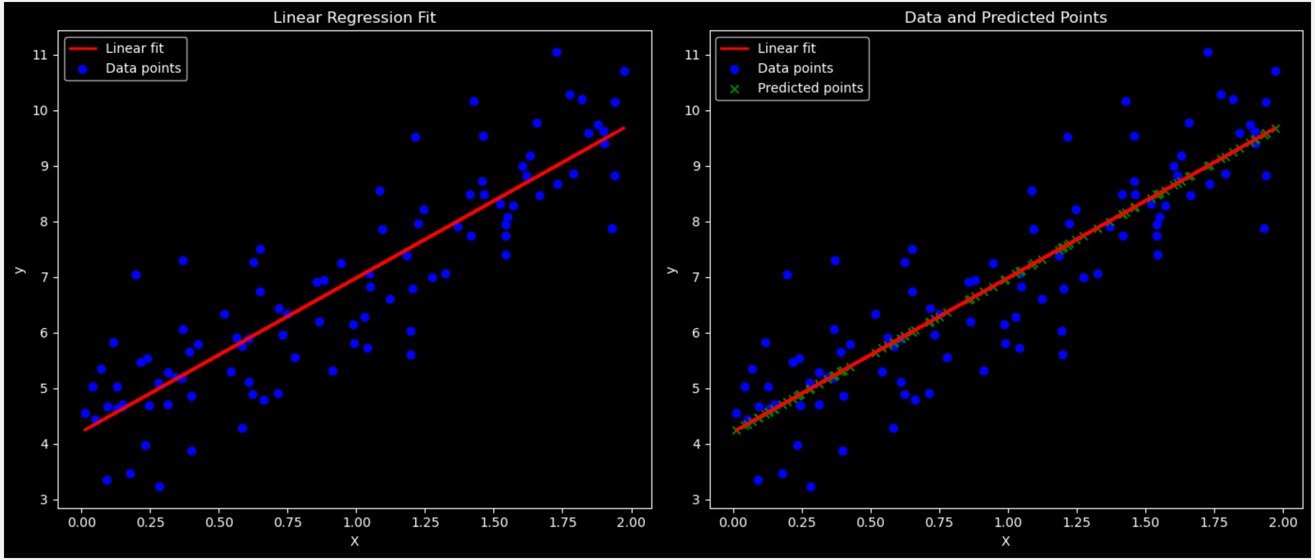
We will discuss this in the exercise session.

Example: Linear regression

- **Task (T)**: Predict a continuous value from a vector of features using a linear model.
  - $y\hat{}=\mathbf{w}T\mathbf{x}+b$, $x \in Rn$, $y \in R$
- **Performance Metric (P)**: Evaluate the model using Mean Squared Error (MSE) on a test set.
  - $\mathrm{MSE} = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}i - yi)^2$
- **Experience (E)**: Train the model using a dataset of input-output pairs to learn the optimal weights and bias.

Example: Linear regression

- **Experience (E)**: Train the model using a dataset of input-output pairs to learn the optimal weights and bias.



Principles of Learning

Generalization Error

- In-sample/ training error
- Out-of-sample/ test error
- Difference between these two is generalization

Generalization Error

$$\mathbf{E}_{in}(h) = \frac{1}{n}\sum_{i=1}^{N} l(h(X_i), Y_i)$$

$$\mathbf{E}_{out}(h) = \mathbb{E}_{p(X,Y)}[l(h(X), Y)]$$

$$\mathcal{G}_{err} = \mathbf{E}_{out}(h) - \mathbf{E}_{in}(h)$$

Not obvious to minimize the generalization error.

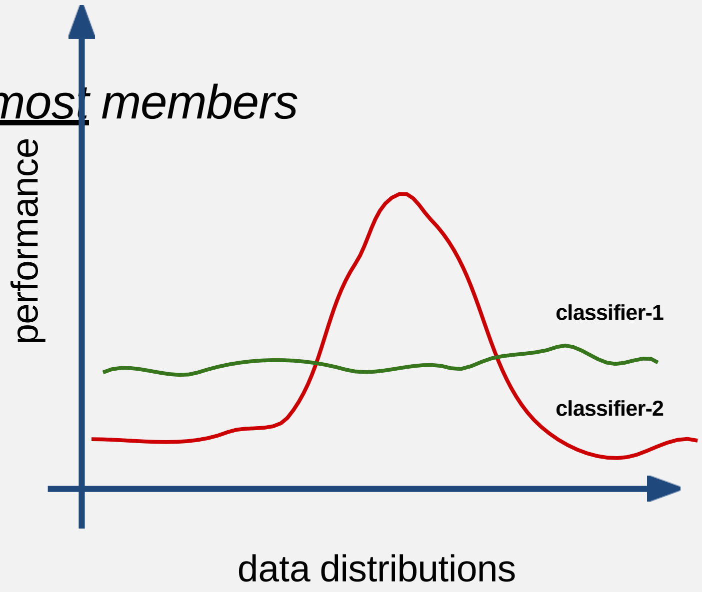
There's No Free Lunch

- ML algorithms can generalize well from finite set of examples
- Contradicts basic principles of logic!
- ML avoids this using probabilistic rules
  - ML finds rules that are <u>probably</u> correct about <u>most</u> members

There’s No Free Lunch

- ML algorithms can generalize well from finite set of examples
- Contradicts basic principles of logic!
- ML avoids this using *probabilistic rules*
- *ML finds rules that are <u>probably</u> correct about <u>most</u> members*

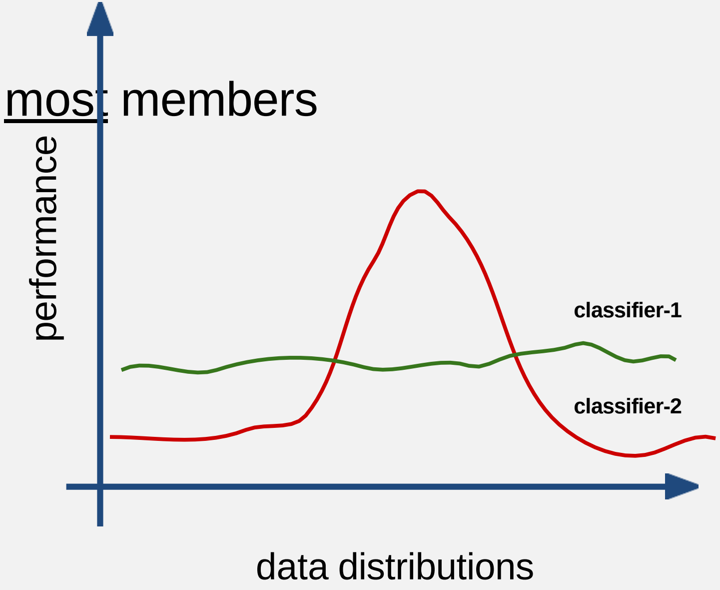
> ***No Free Lunch Theorem:*** *Averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points (Wolpert, 1996)*



There's No Free Lunch

- ML algorithms can generalize well from finite set of examples
- Contradicts basic principles of logic!
- ML avoids this using probabilistic rules
- ML finds rules that are probably correct about most members

No Free Lunch Theorem: Averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points (Wolpert, 1996)



- Goal is then not to seek a universal learning algorithm but what kinds of distributions are relevant

Four horsemen of ML failure

1. Data assumptions
2. Data snooping
3. Underfitting
4. Overfitting

Data assumptions

1. **i.i.d**
    - **Identical:** Data is drawn from the same data distribution
    - **Independent:** Data points independent from each other
2. Sampling/Selection bias

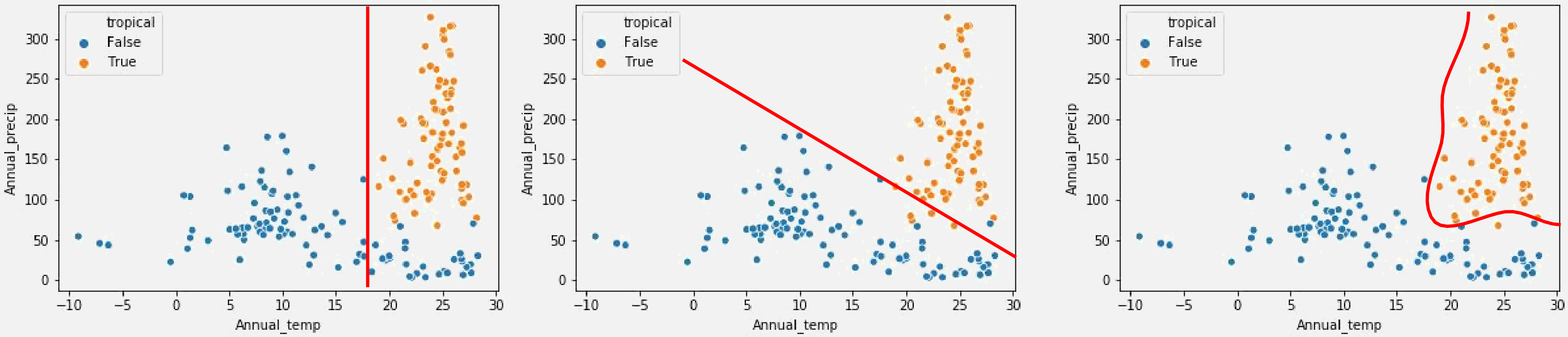
- If i.i.d assumption is violated does learning work?
- How can we overcome?

Data Snooping

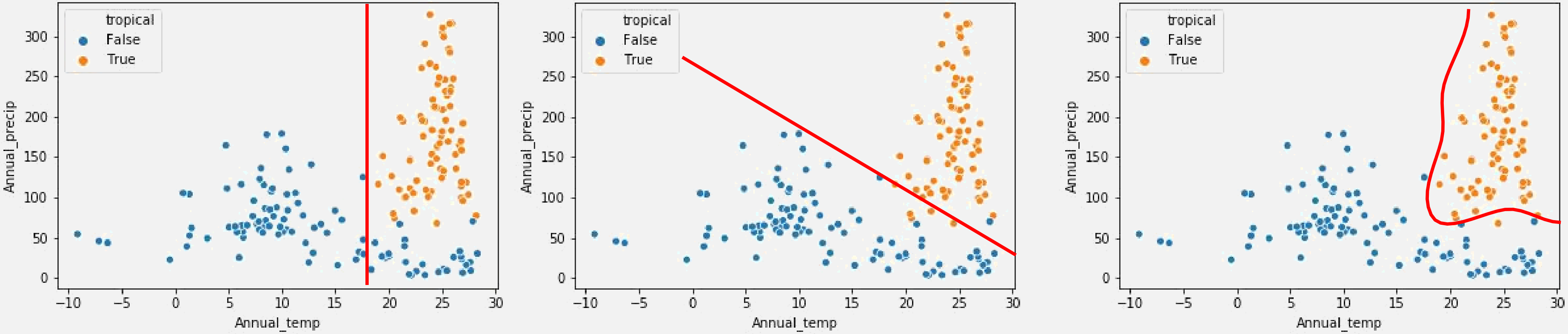
- Test data has informed the model selection
- Generalization suffers

"If you want an unbiased assessment of your learning performance, you should keep a test set in a vault and never use it for learning in any *way"*
Mostafa et al. Learning from data (book)

Underfitting & Overfitting



Underfitting & Overfitting



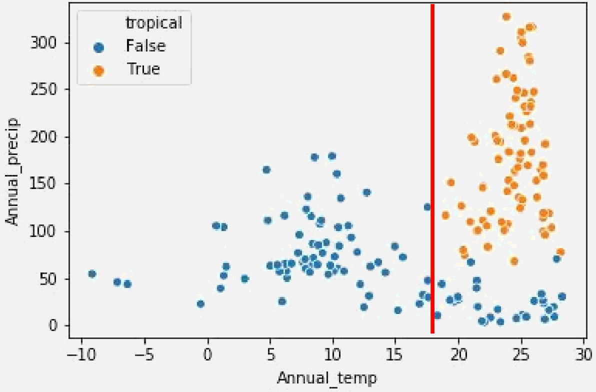
Underfitting

Appropriate capacity

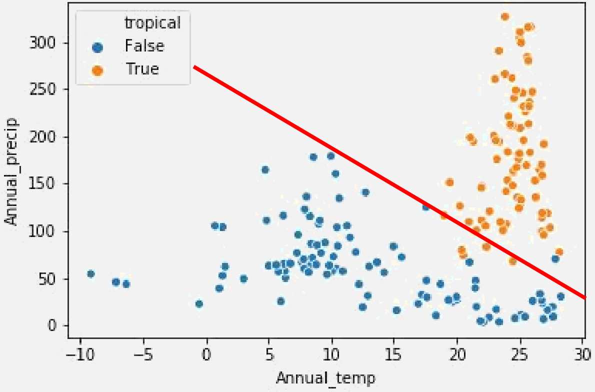
Overfitting

Underfitting & Overfitting

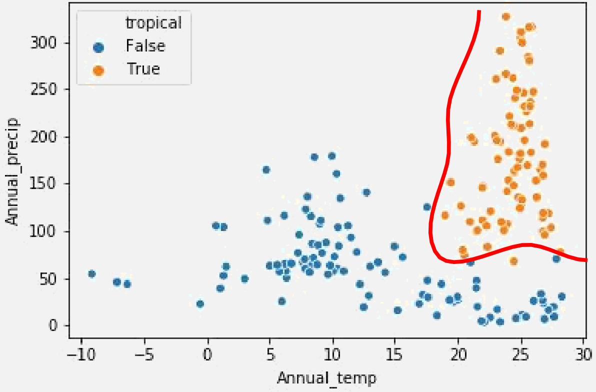
- Models are chosen based on training error
- Test error ≥ Training error



Underfitting



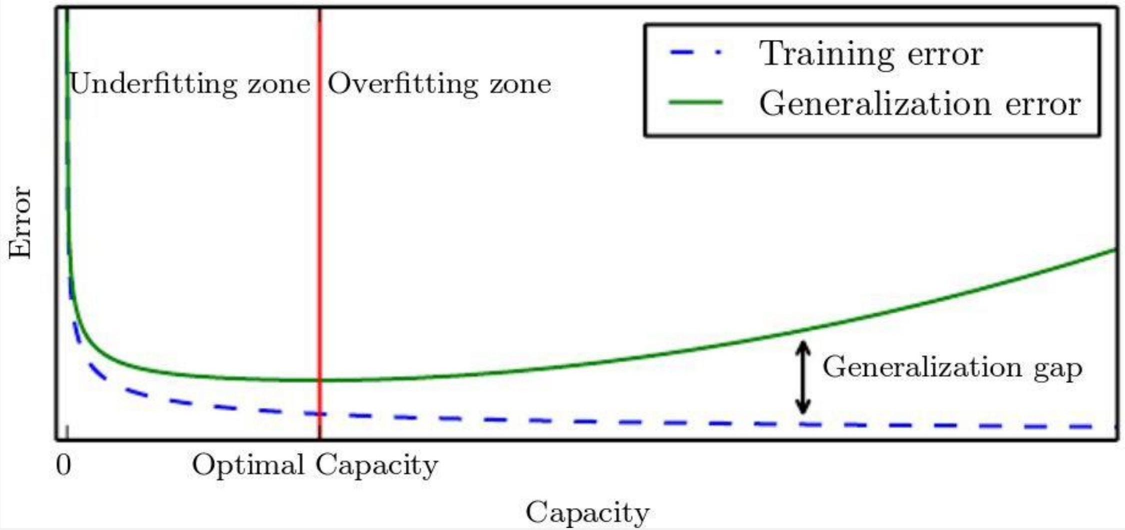
Appropriate capacity



Overfitting

Handling overfitting

- Representational capacity
    - **Occam's Razor:** *"The* simplest model that fits the data is also the most plausible."



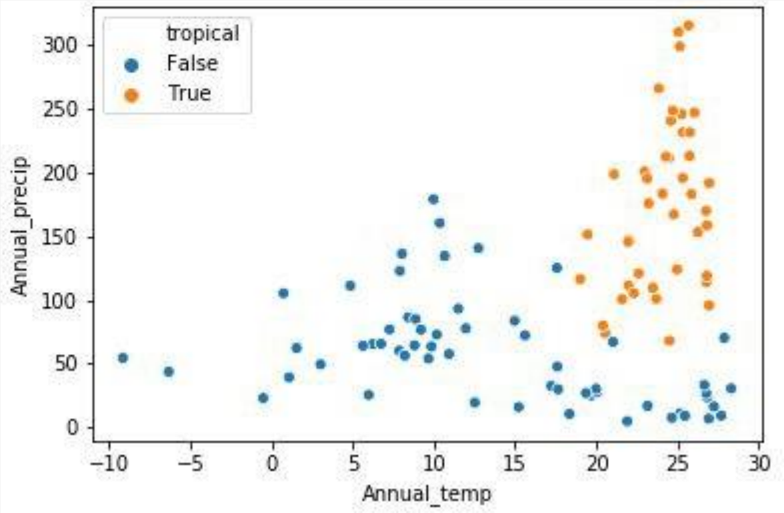
Summary of Learning Principles

- Data is not ideal
- Lock away test data
- Low generalization error is the Holy Grail of all ML
- Model capacity is hard to decide, even with Occam’s Razor
- Underfitting & Overfitting can hamper performance

A non-linear classifier

k-Nearest Neighbour Classification

- What if data is not linearly separable?
- What about multi-class classification?

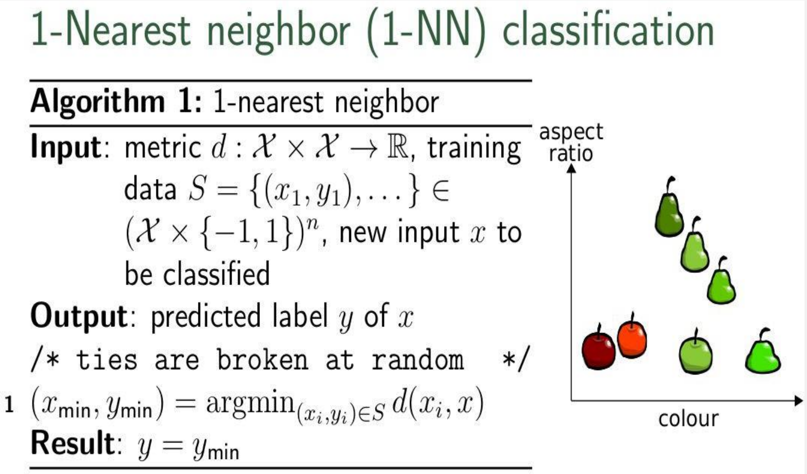


k-Nearest Neighbour Classification

- Non-parametric method
- Based on neighbourhoods
- Requires a notion of similarity
- One of the simplest, yet effective methods
- Multi-class classification

k-Nearest Neighbour Classification

- Non-parametric method
- Based on neighbourhoods
- Requires a notion of similarity
- One of the simplest, yet effective methods
- Multi-class classification



How to choose k?

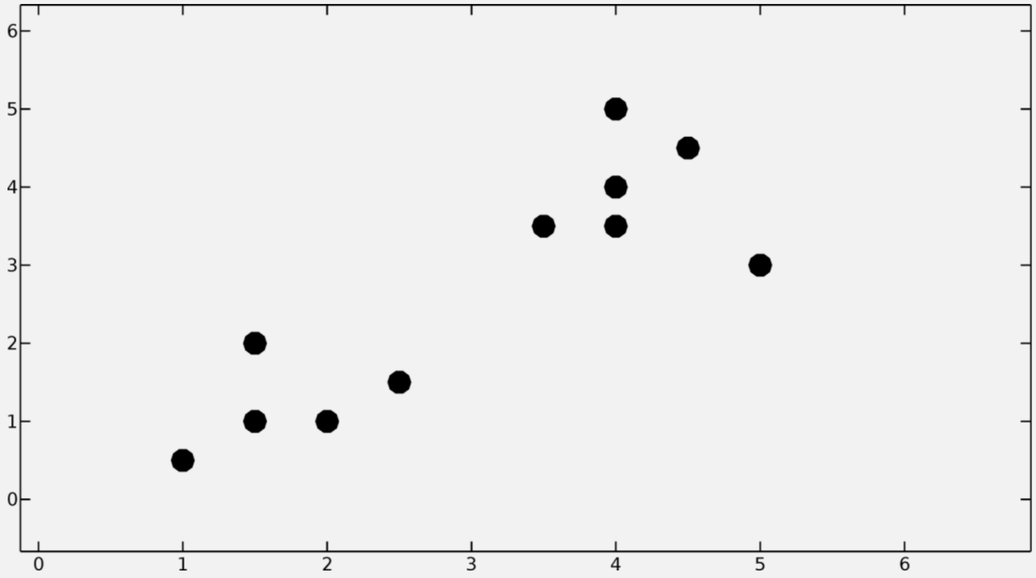
An unsupervised clustering algorithm

Unsupervised Learning: Clustering

We focus on k-Means clustering

- Process of grouping similar objects together
- Detecting patterns
- Either based on similarity or features
- Representing data at higher abstractions
- Applications like image segmentation

Toy example:

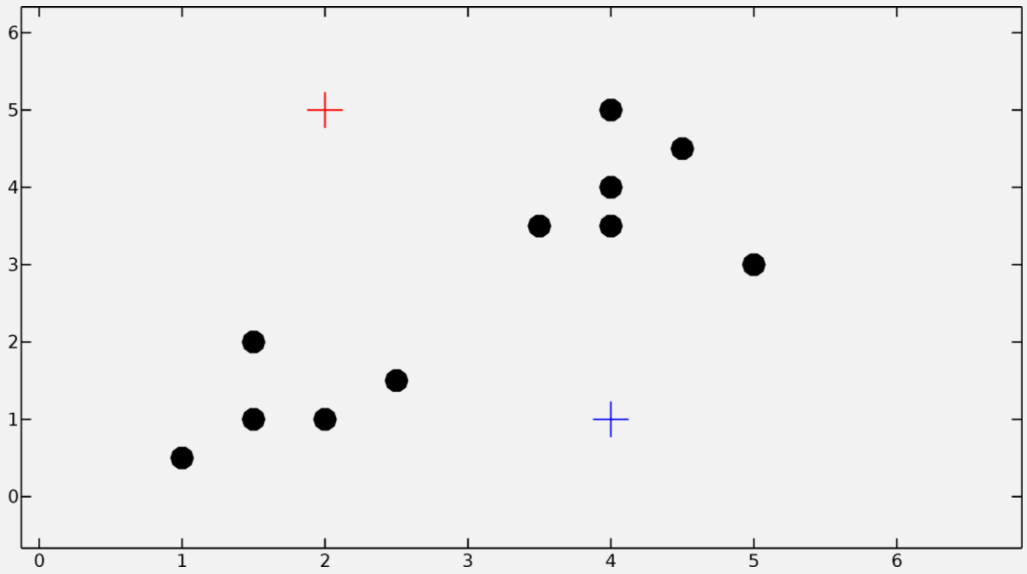


How would you cluster these points?

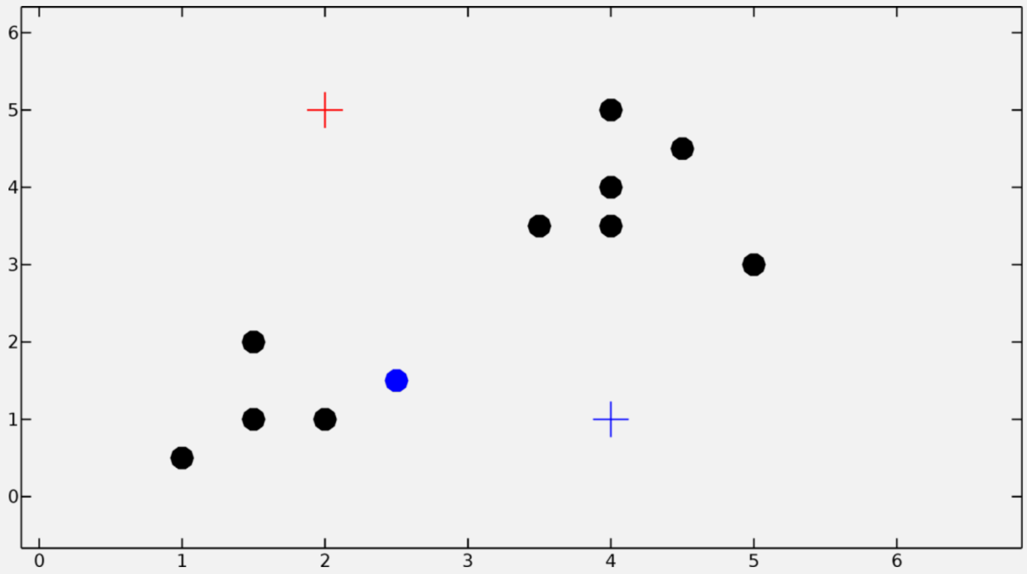
Think on these lines...

- Measure of similarity
- Number of clusters
- Complexity

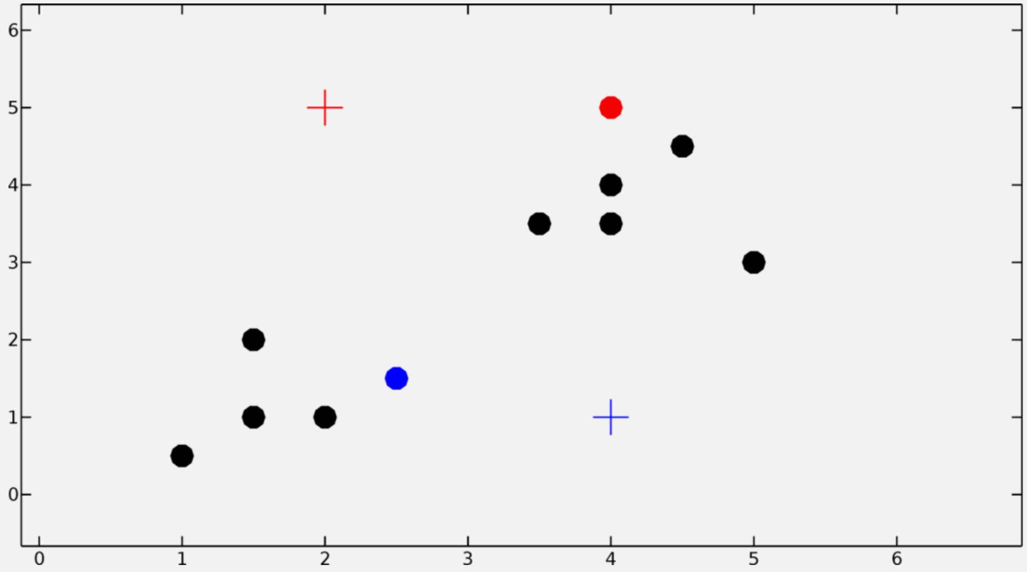
Initialize centroids, randomly!



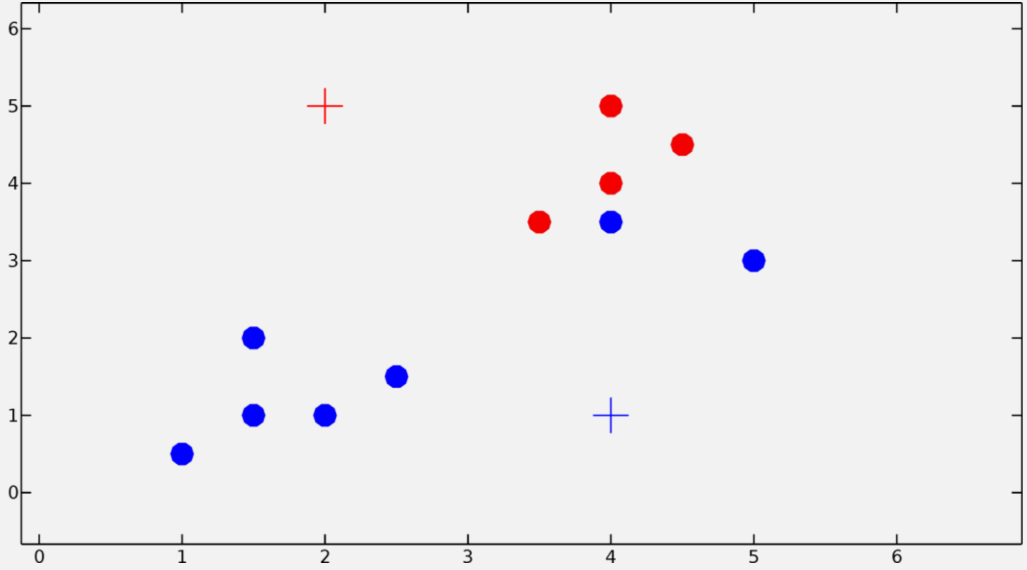
Assign points to nearest centroid!



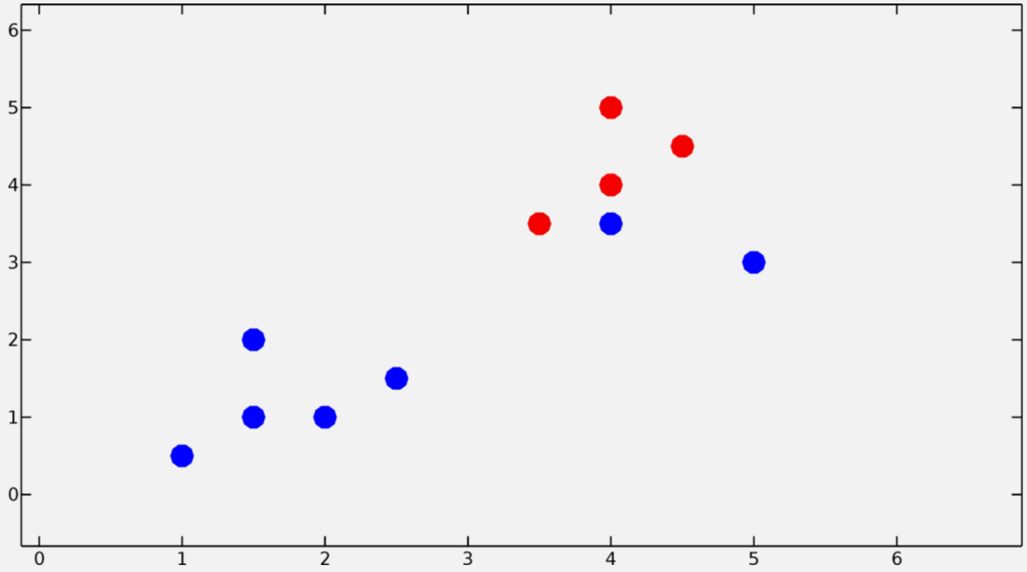
Assign points to nearest centroid!



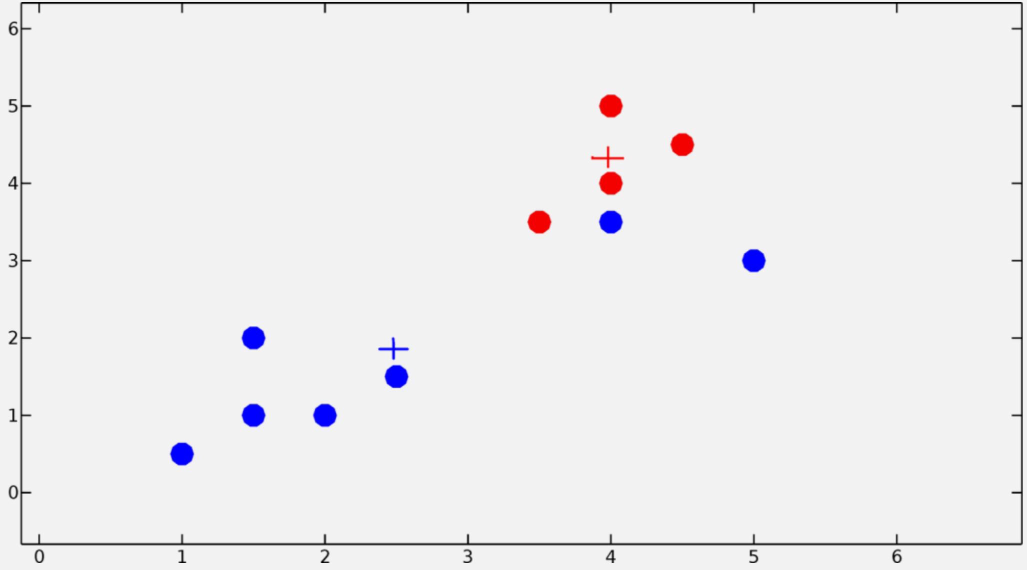
Assign points to nearest centroid!



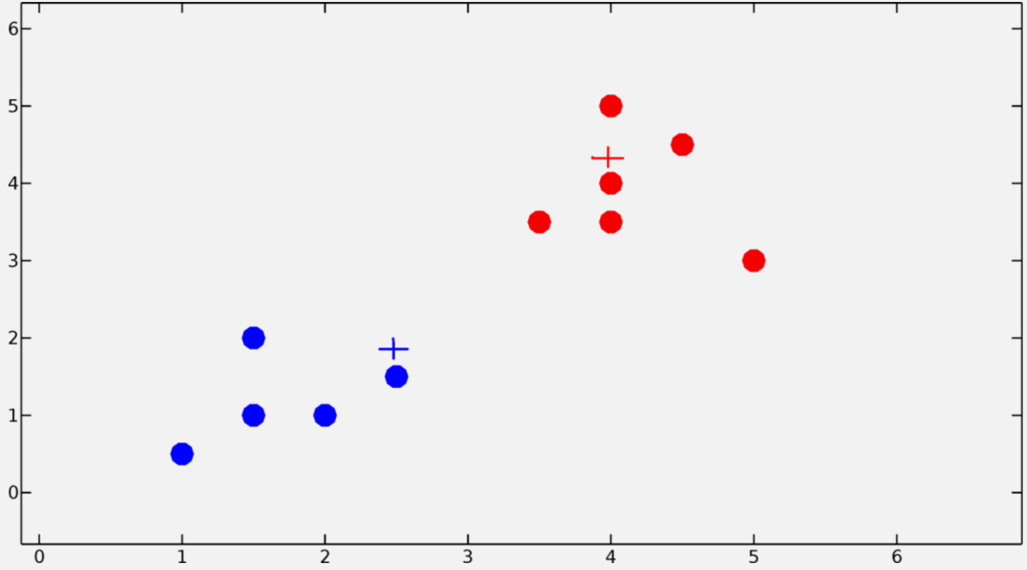
Recompute centroids!



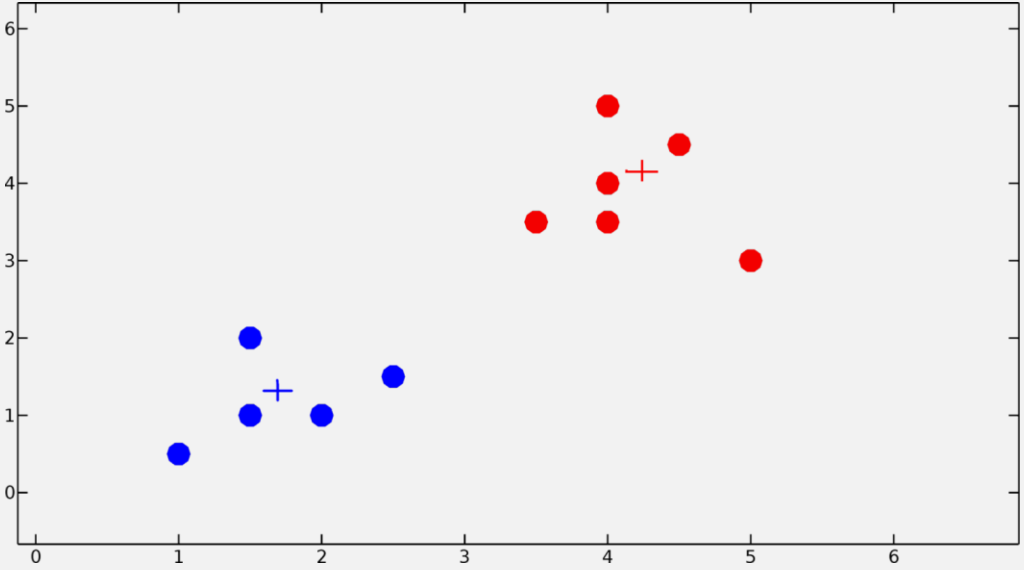
Recompute centroids!



Iterate, until convergence!



Iterate, until convergence!

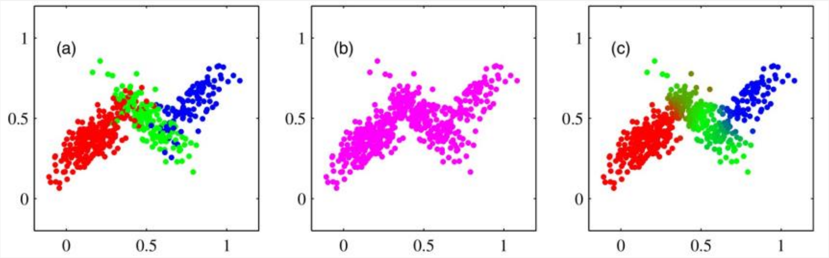


k-Means clustering based Image Segmentation

Figure 9.3 Two examples of the application of the K-means clustering algorithm to image segmentation showing the initial images together with their K-means segmentations obtained using various values of K. This also illustrates of the use of vector quantization for data compression, in which smaller values of K give higher compression at the expense of poorer image quality.

Figure from Christopher Bishop, PRML

Summary: k-Means Clustering



- + Simple with good performance
- + Single hyperparameter
- + Cross validation for parameter selection
- + Flexibile similarity measures
- + **Hard EM**: Assigns hard labels
- + Powerful unsupervised method when used with PCA
- - k has to be pre-selected; Sensitive to initialization

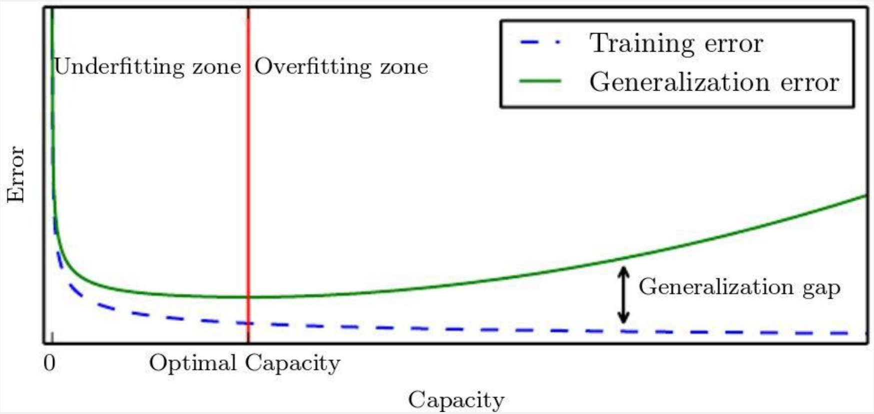
How to choose k?

[5]Fig. 9.5 from Christopher Bishop, PRML

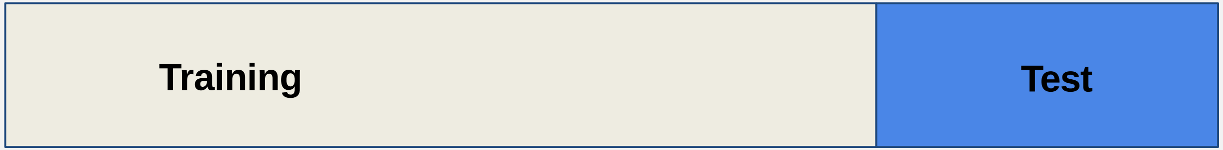
Model Selection & Validation

Model Selection & Validation

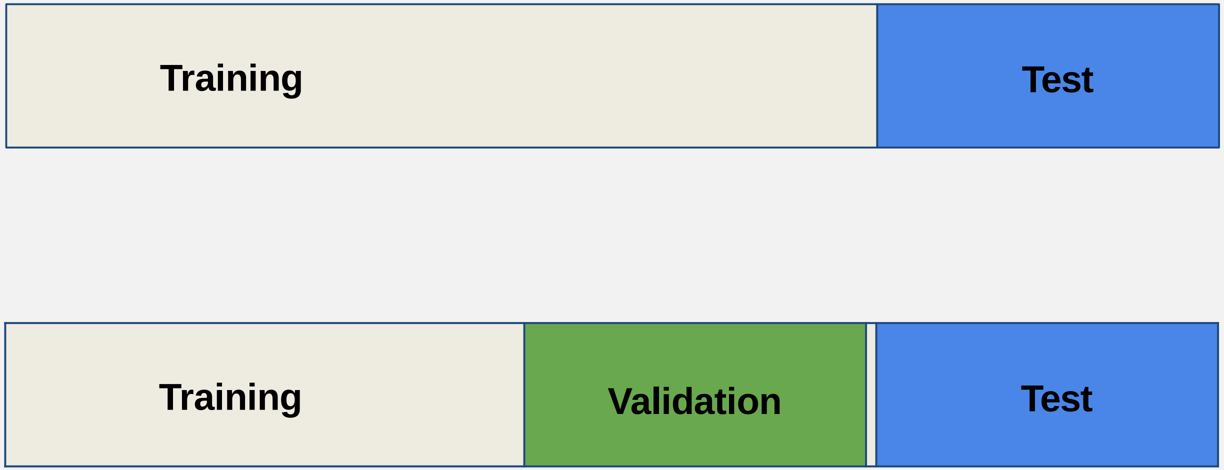
- How to avoid Overfitting
- How to pick models based on training error



Validation Set comes to the rescue



Validation Set comes to the rescue

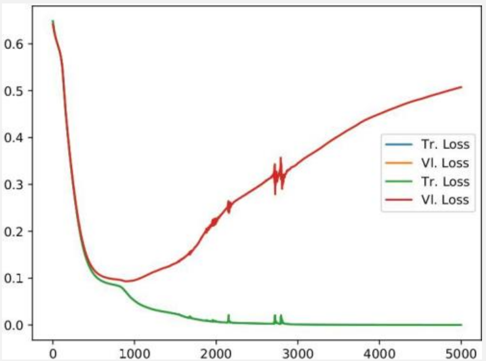


Validation Set comes to the rescue

- Training data for training
- Validation data for model selection
- Hyper-parameters can be selected with it
- Rule of thumb: 60-20-20

Consequences:

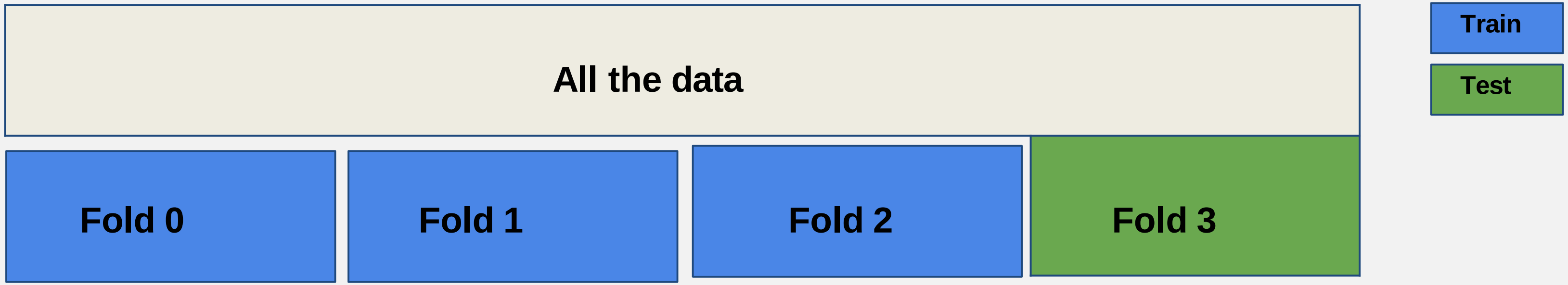
- Reduction in training data
- Computational overhead



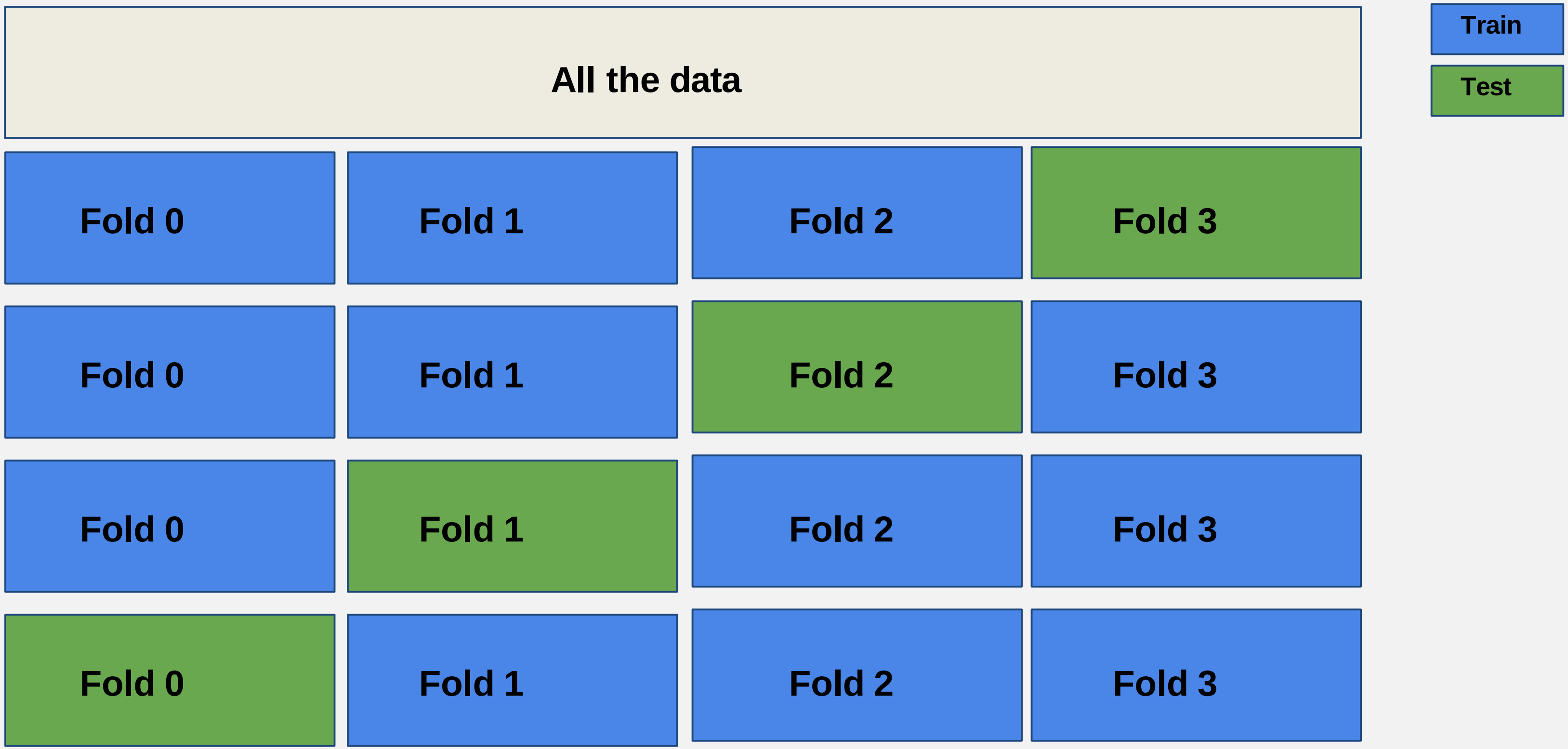
Cross-validation gives more training data

All the data

Cross-validation gives more training data



Cross-validation gives more training data



Summary

- Models selection is not straightforward
- Pick a class of models -> Tune hyper-parameters
- Training data to select models
- Generalization suffers if only based on training data
- Many (equally better/worse) models to choose from
- Cross validation to the rescue (?)
- Hard to generalize
- And, no free lunch