session 4:
Supervised learning

M. Kundegorski

13th December 2019

IAFIG-RMS - Bioimage Analysis With Python

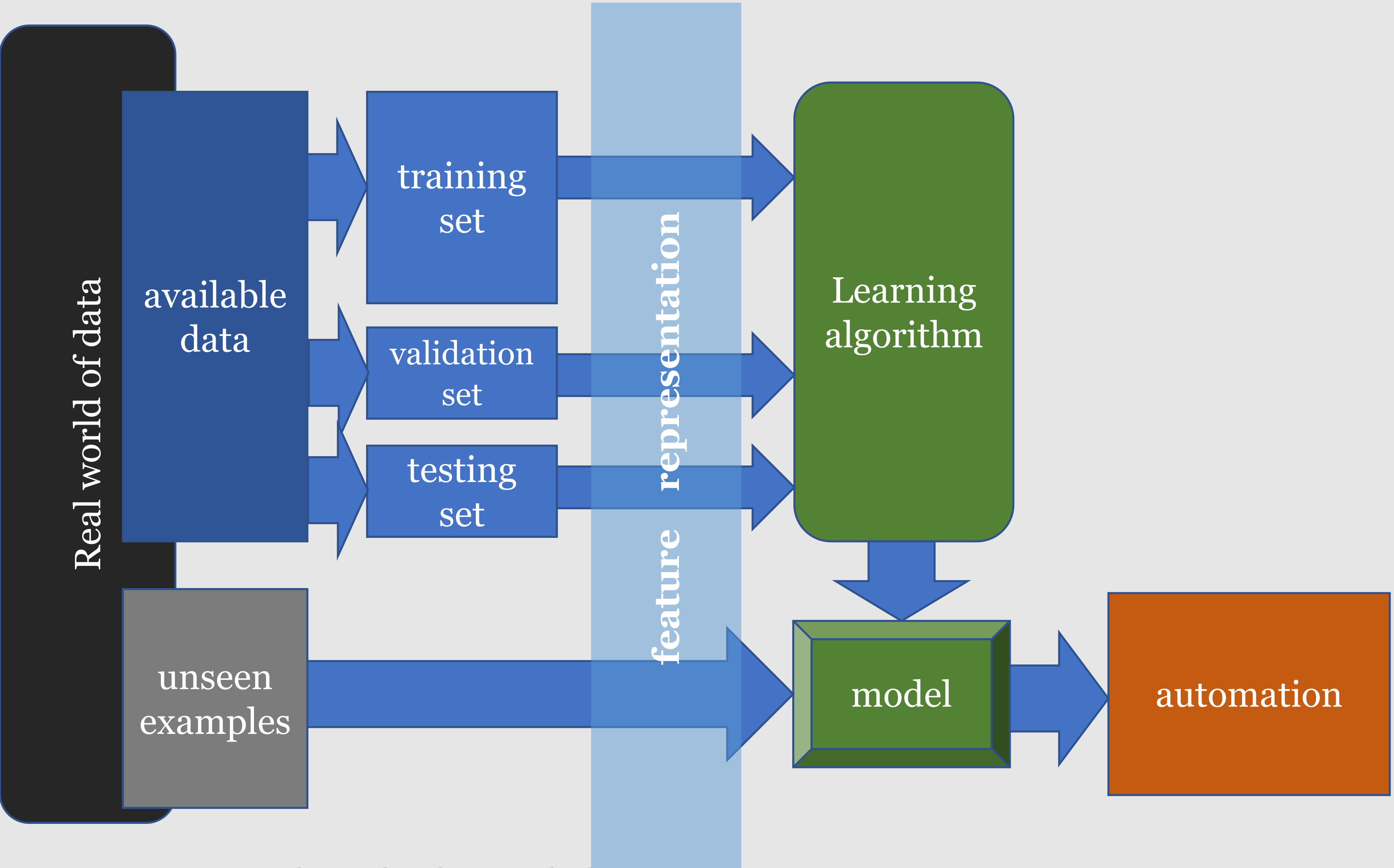
Cambridge Bioinformatics Training Centre

Royal Microscopical Society

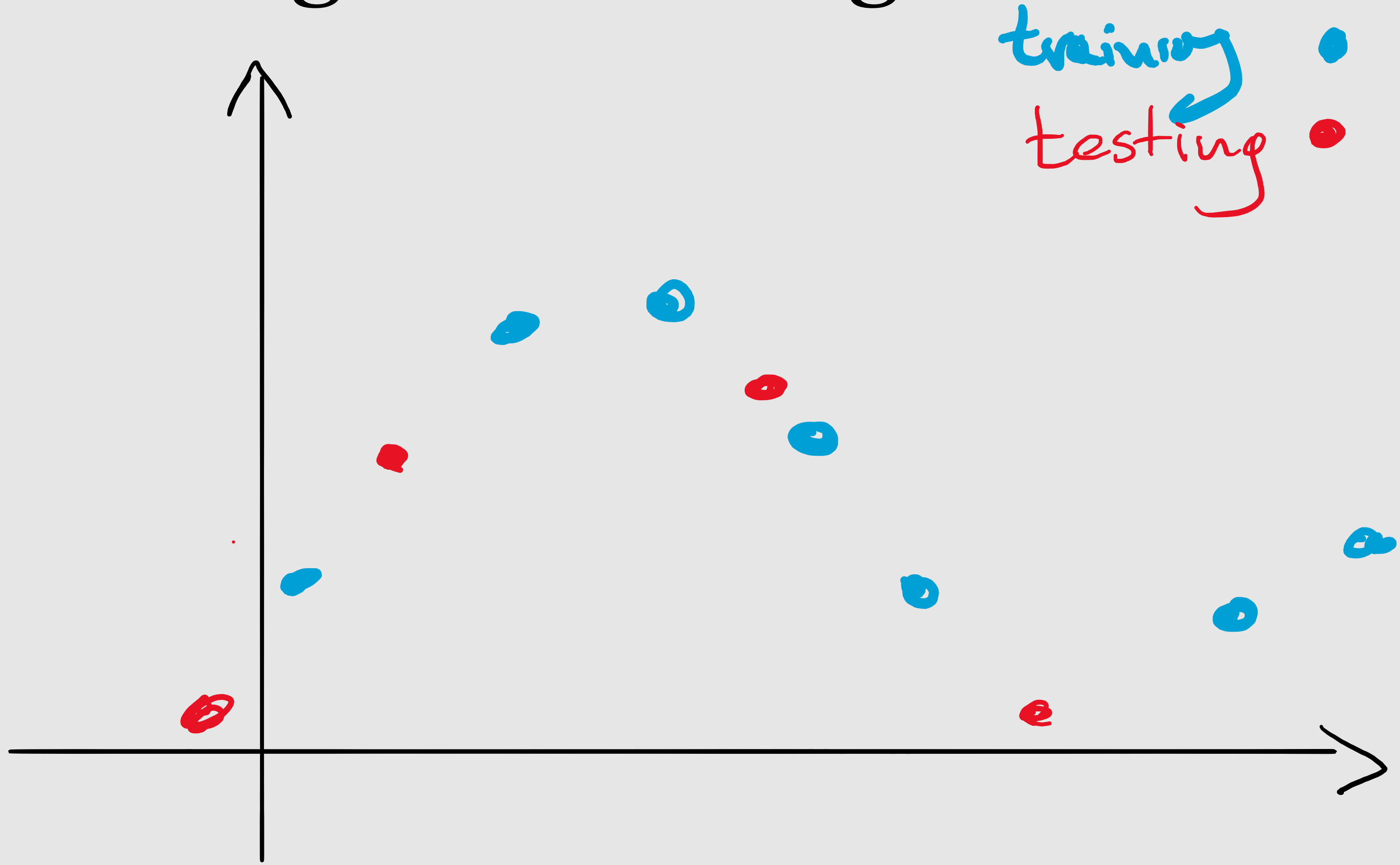
Traditional supervised learning

Traditional supervised learning still proves itself useful

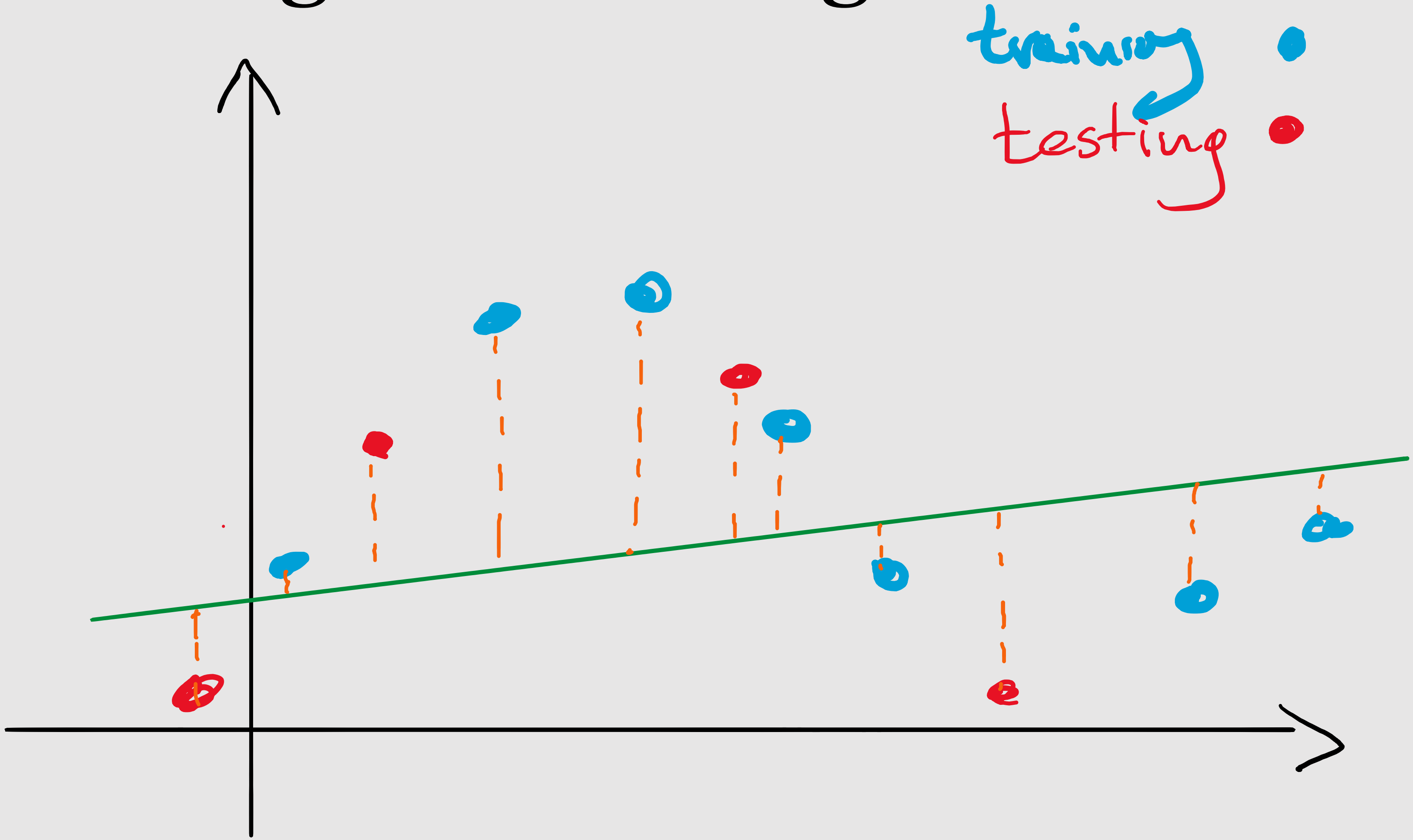
It is a framework



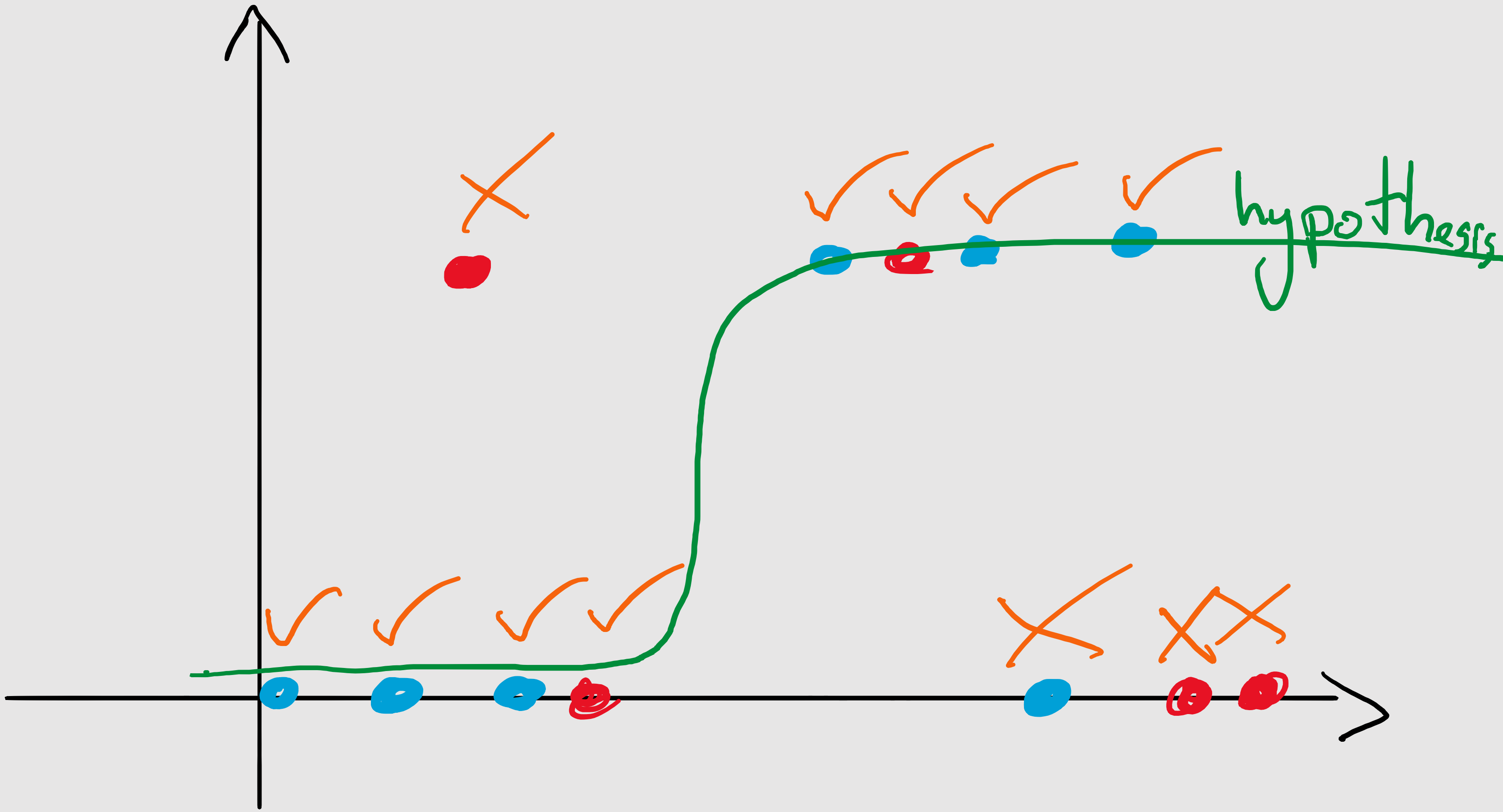
Training and Testing Errors



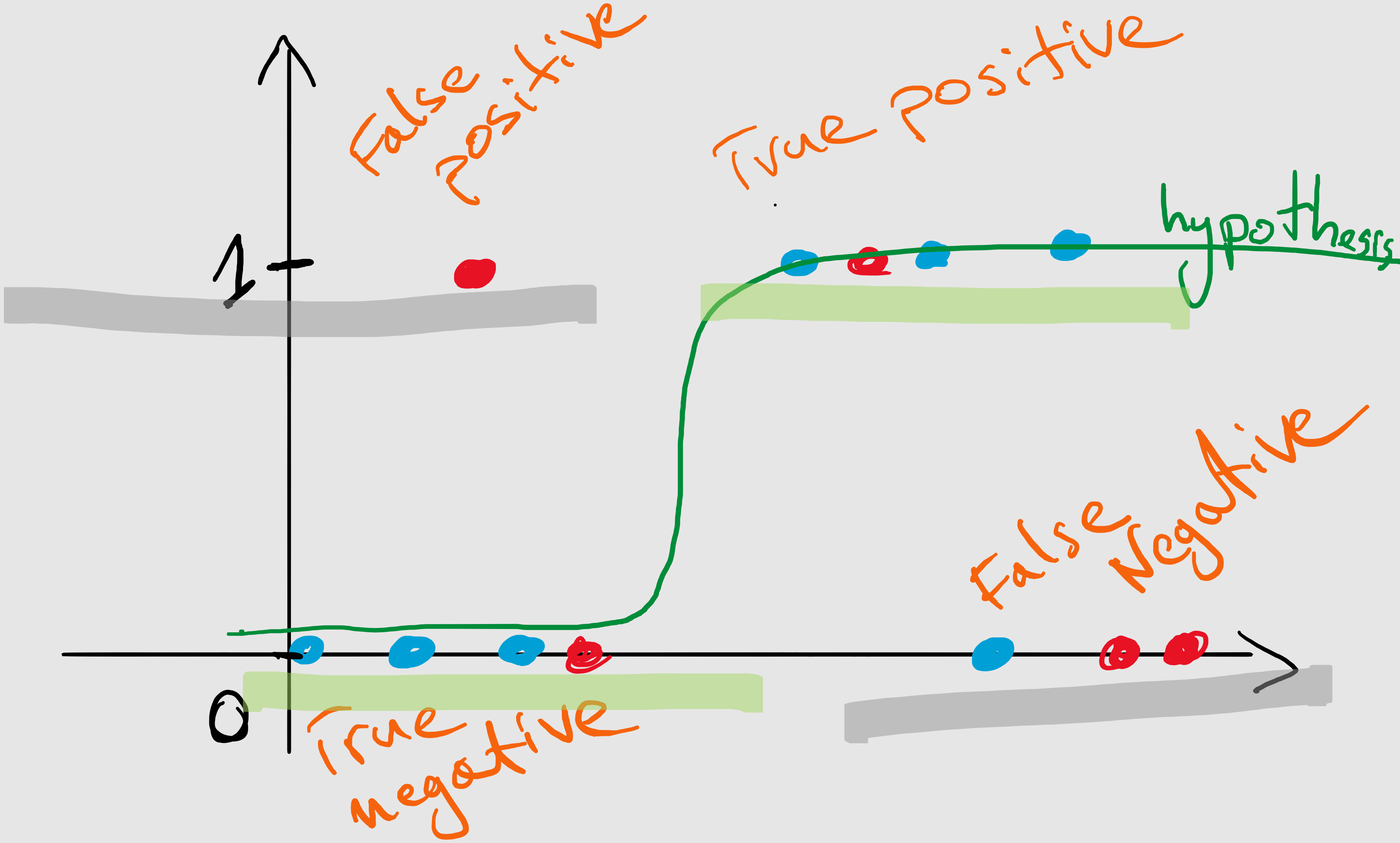
Training and Testing Errors



Training and Testing Errors



Training and Testing Errors



Performance measures

		actual	
		1	0
predicted	1	TRUE POSITIVE	FALSE POSITIVE
	0	FALSE NEGATIVE	TRUE NEGATIVE

$\text{class} \in \{1, 0\}$

Performance measures

predicted \ actual	1	0
1	TRUE POSITIVE	FALSE POSITIVE
0	FALSE NEGATIVE	TRUE NEGATIVE

Precision

$$\frac{TP}{\text{Predicted Pos}} = \frac{TP}{TP + FP}$$

Recall

$$\frac{TP}{\text{Actual Pos}} = \frac{TP}{TP + FN}$$

$\text{class} \in \{1, 0\}$

F1-score

$\frac{P+R}{2}$ $\quad\quad$ $2\frac{PR}{P+R}$

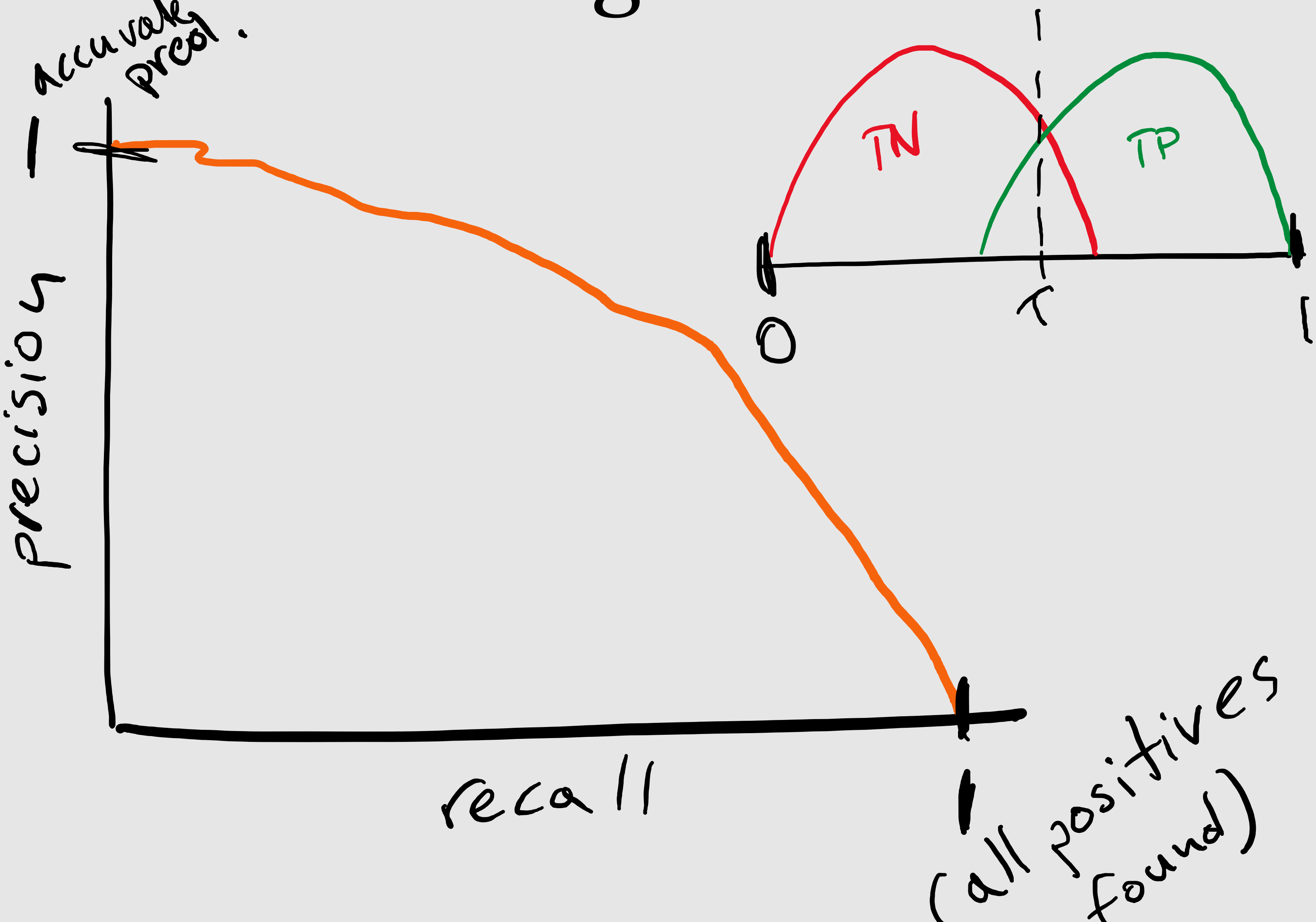
	Precision	**Recall**	**Average**	**F1-score**
Method A	0.5	0.4	0.45	0.44
Method B	0.7	0.1	0.4	0.18
Method C	0.02	1.0	0.51	0.04

Precision == Positive Pred. Power

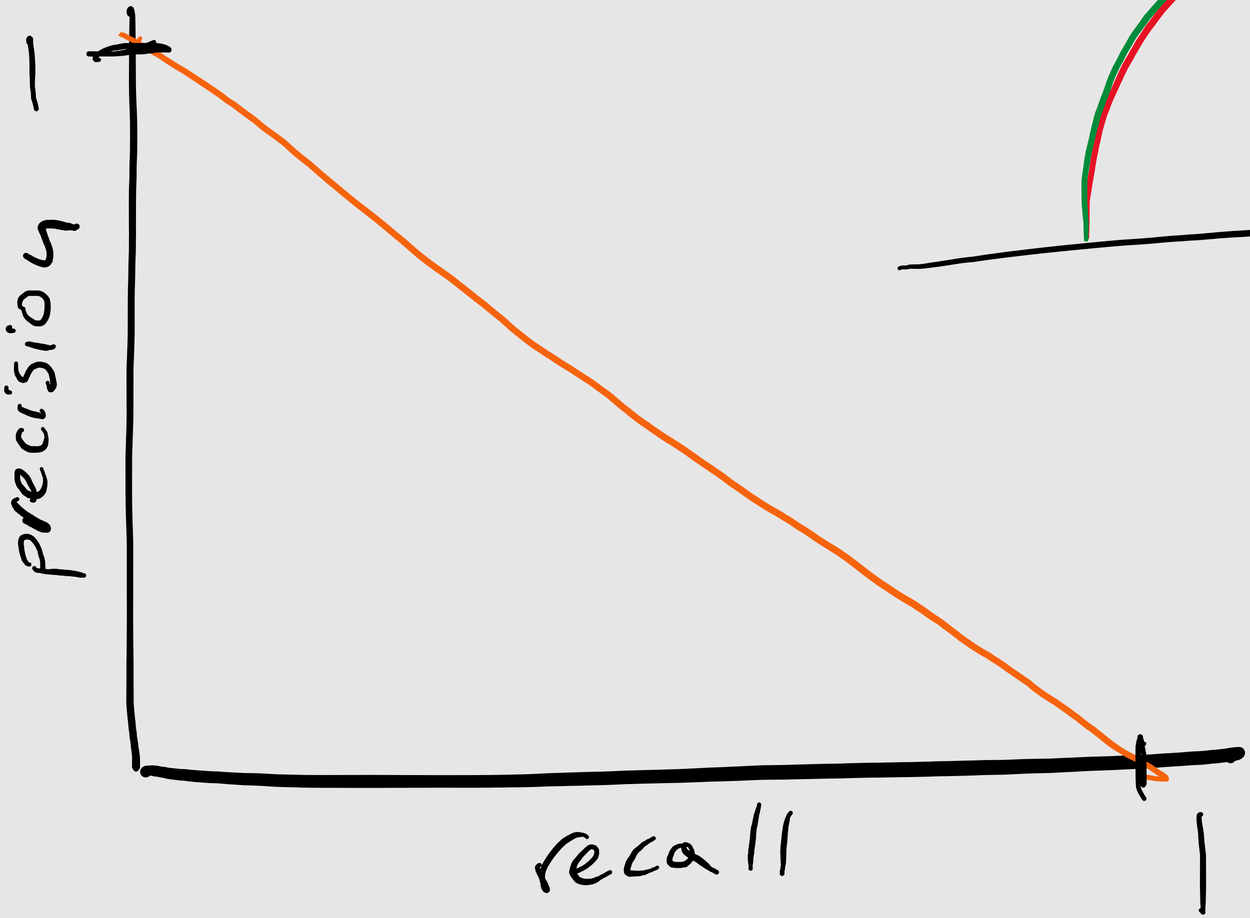
harmonic mean

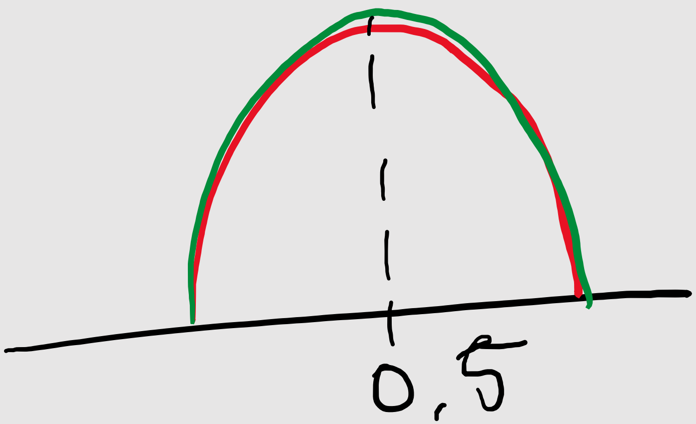
Recall == Sensitivity

mean Average Precision

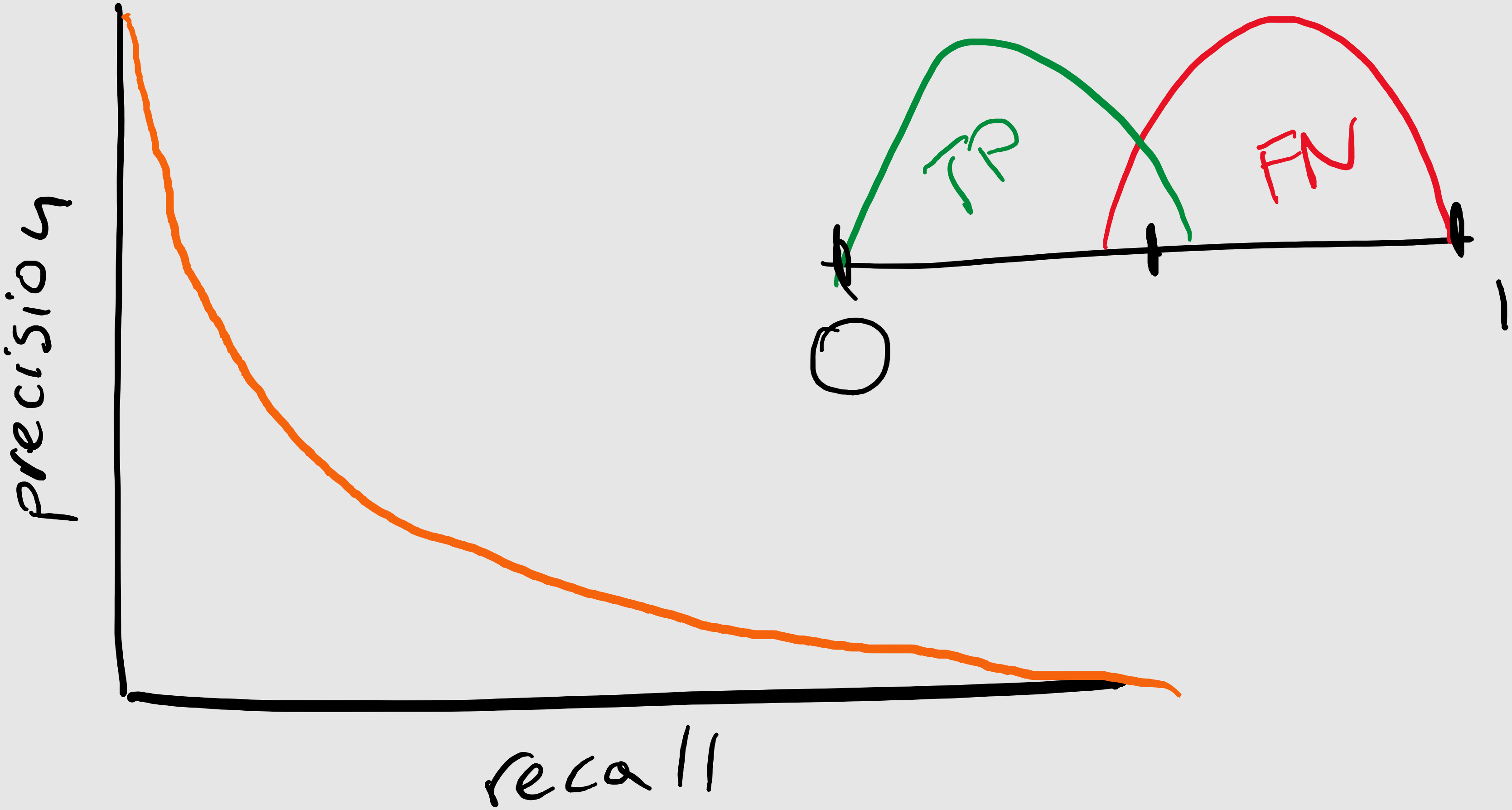


mean Average Precision

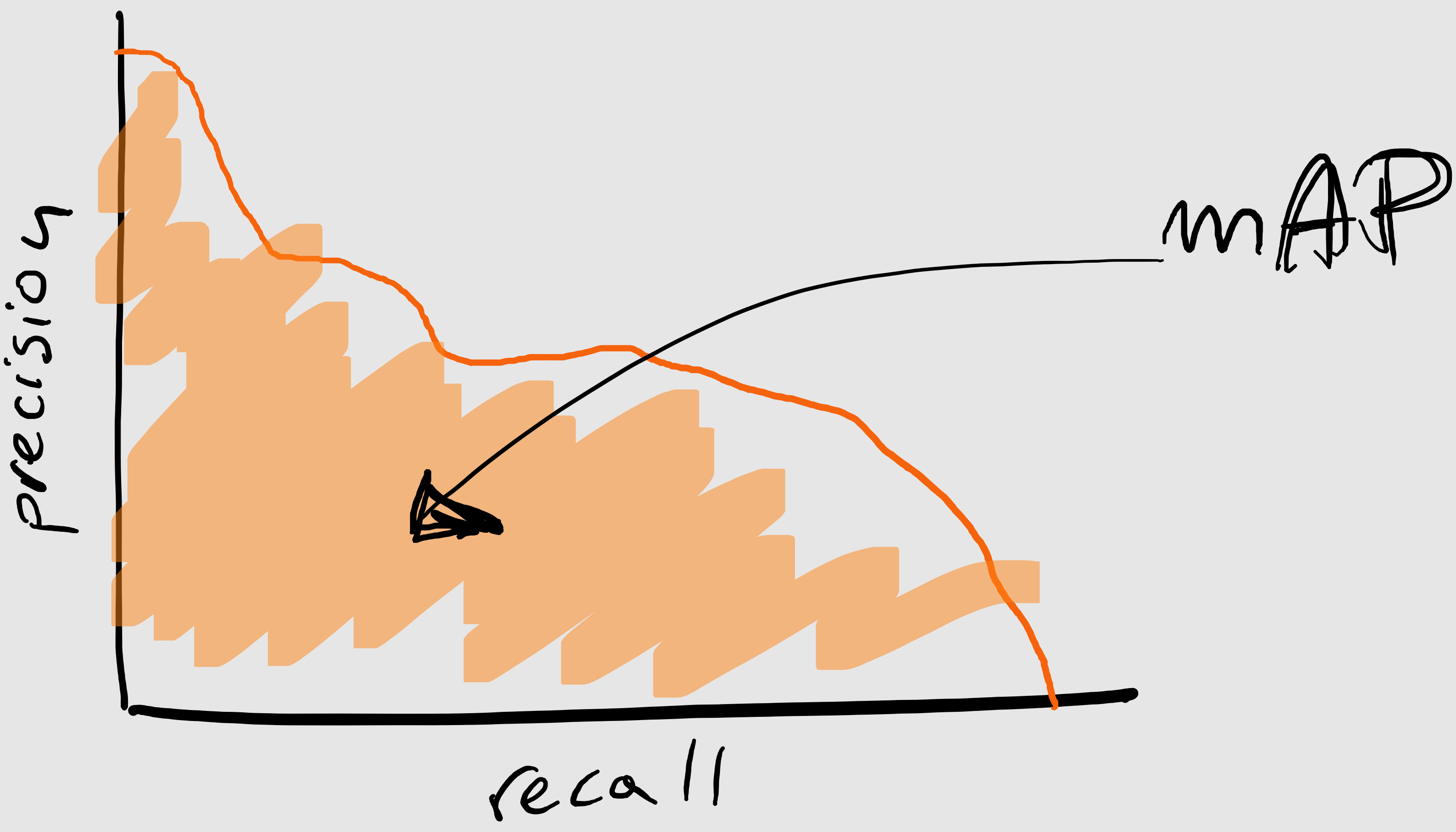




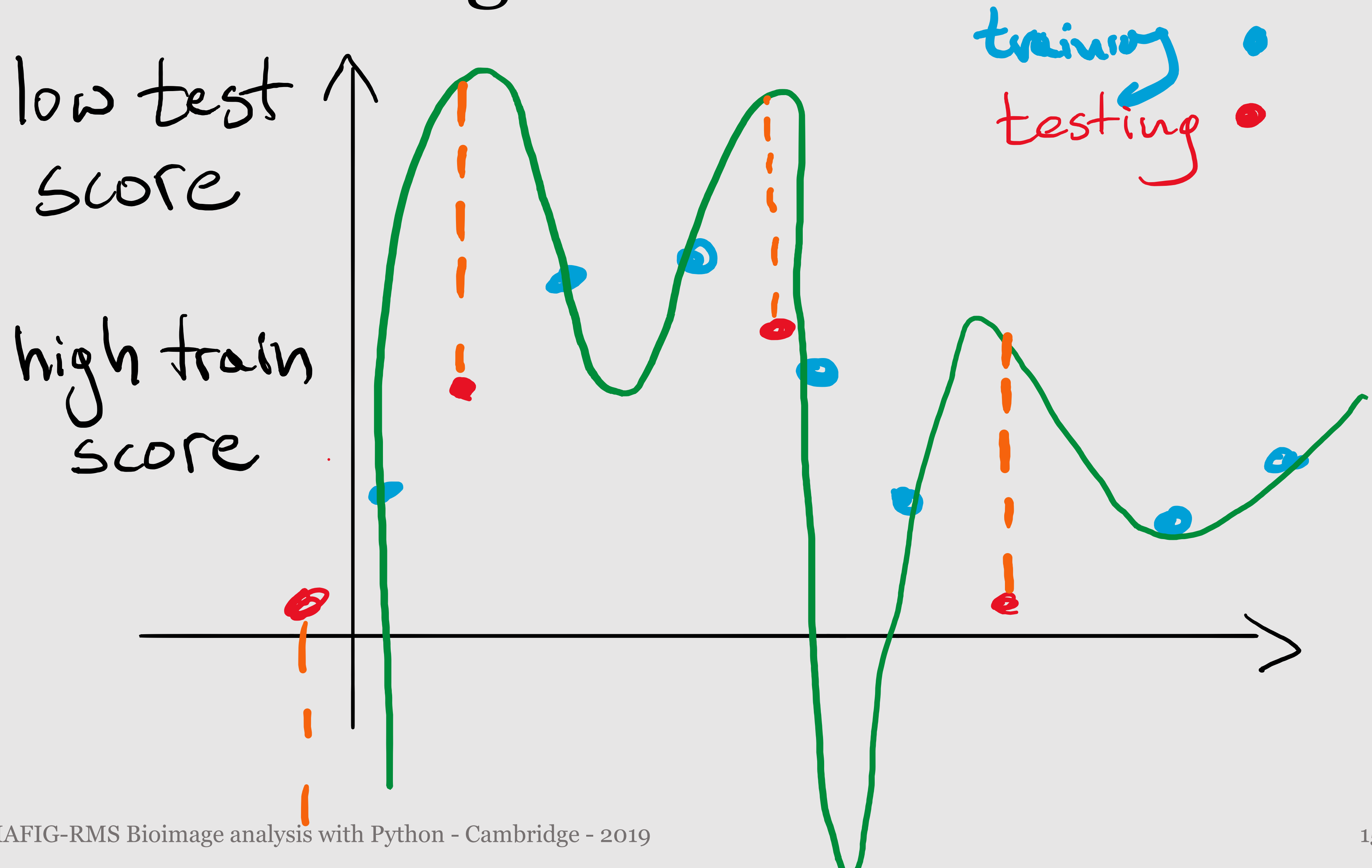
mean Average Precision



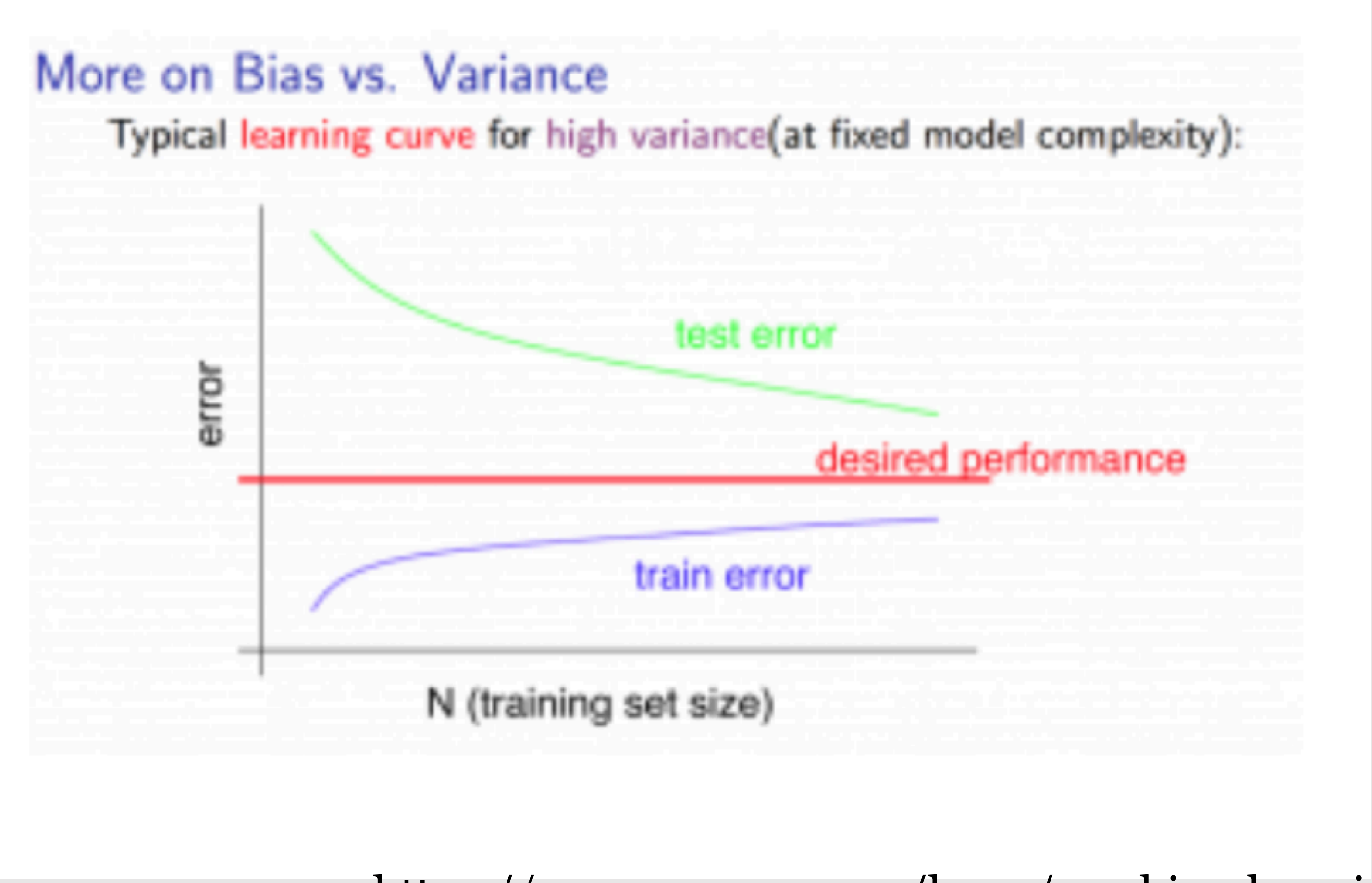
mean Average Precision



Overfitting

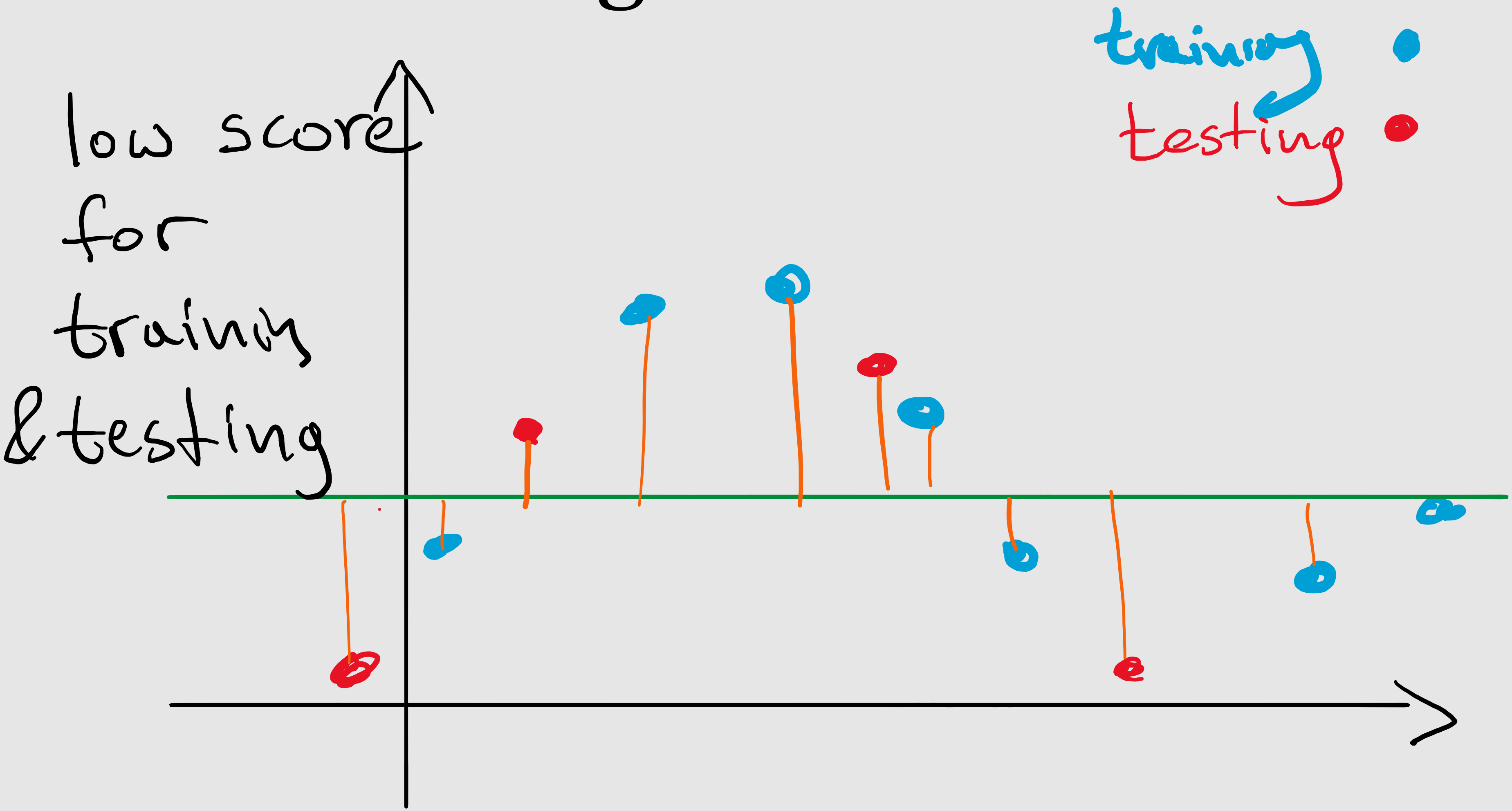


Overfitting – high variance

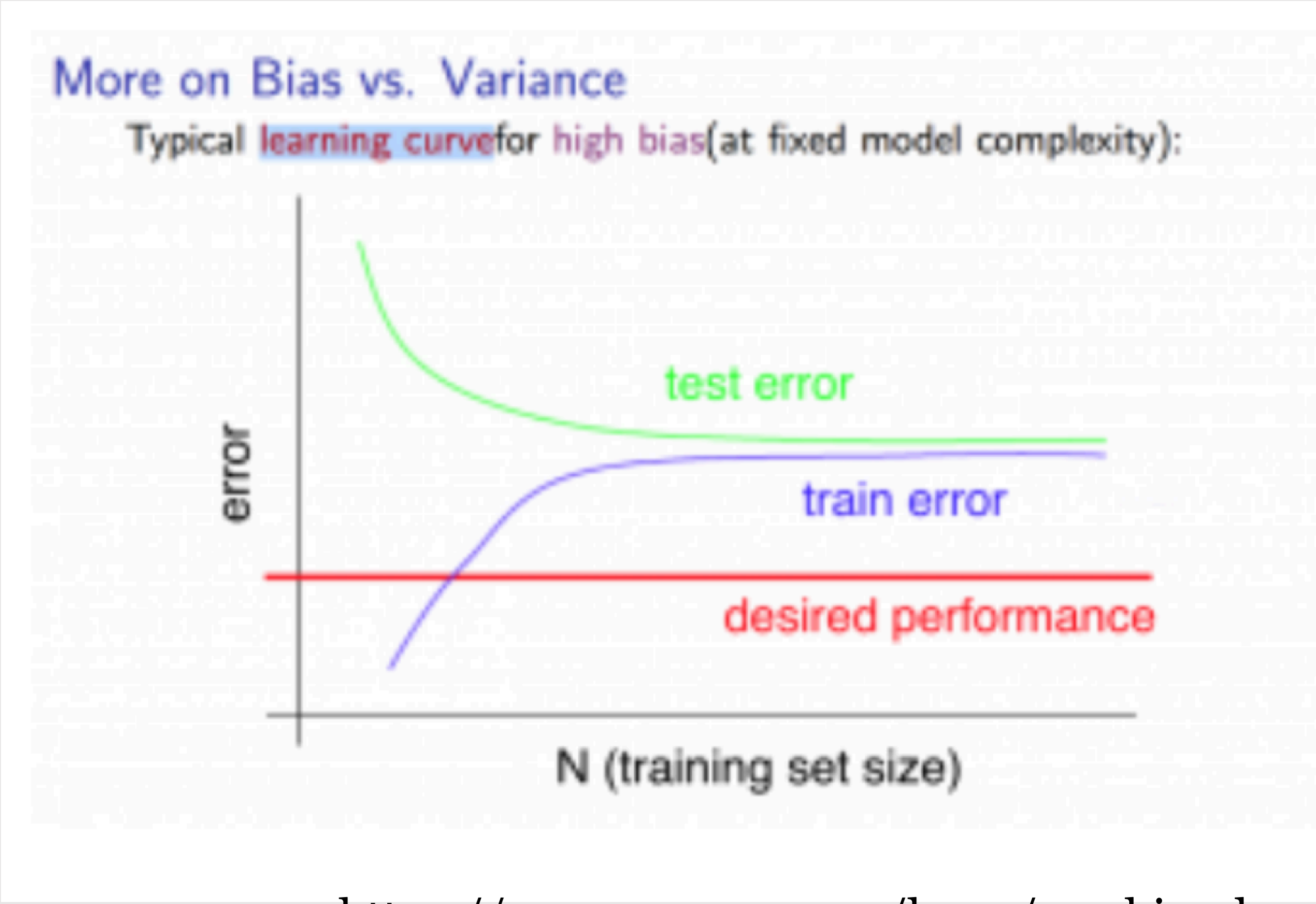


source: https://www.coursera.org/learn/machine-learning

Underfitting

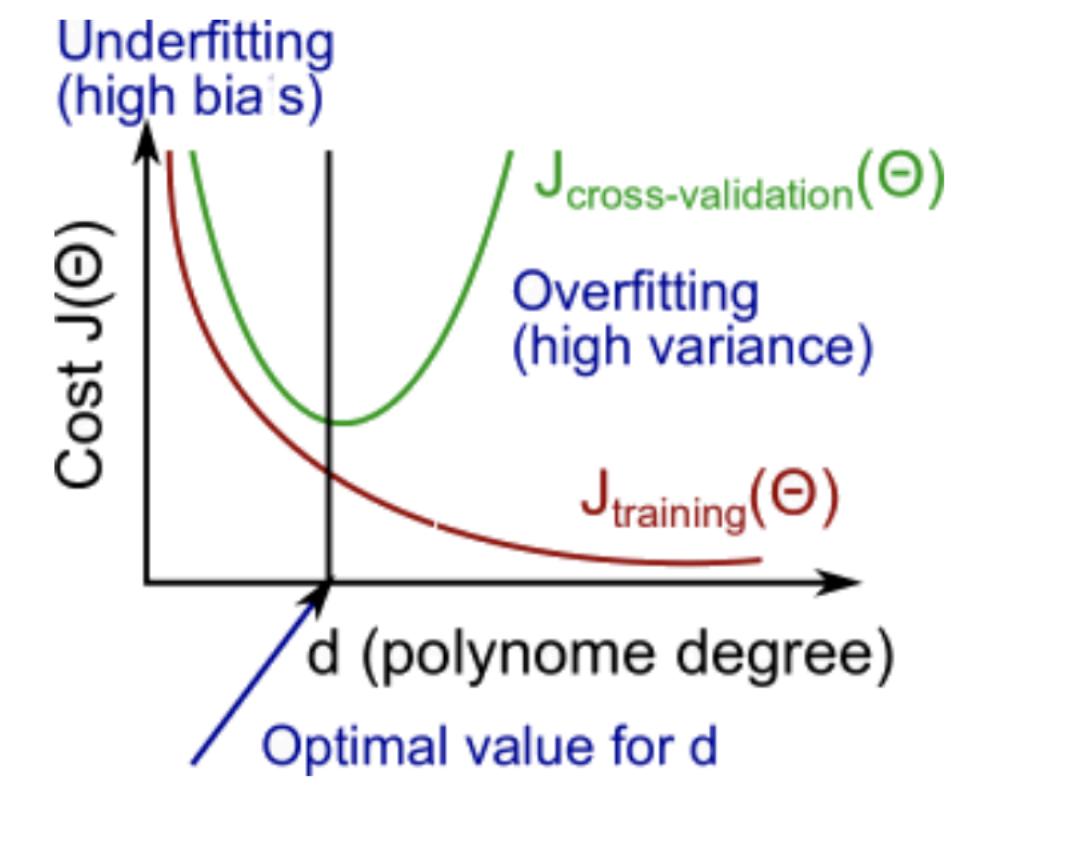


Underfitting – high bias



source: https://www.coursera.org/learn/machine-learning

Overfitting – high variance

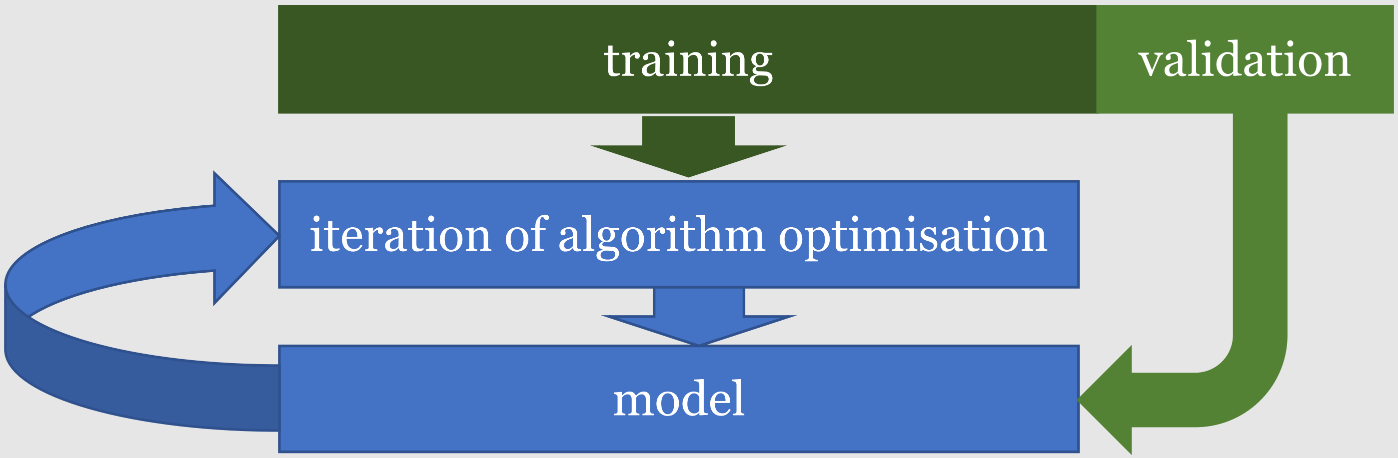


source: https://www.coursera.org/learn/machine-learning

Regularisation

$$J(\theta) = \mathrm{cost}(h_\theta(x), y) + \lambda \sum_j \theta_j^2$$

Cross-validation



k-fold CV: every iteration take different (k^{th}) part of the dataset

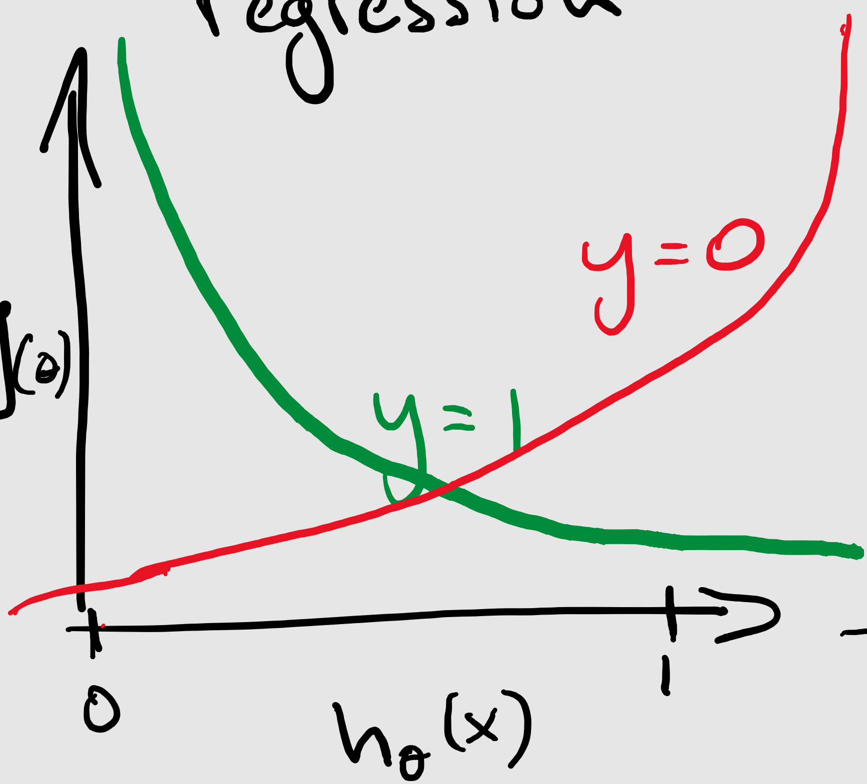
Image classification

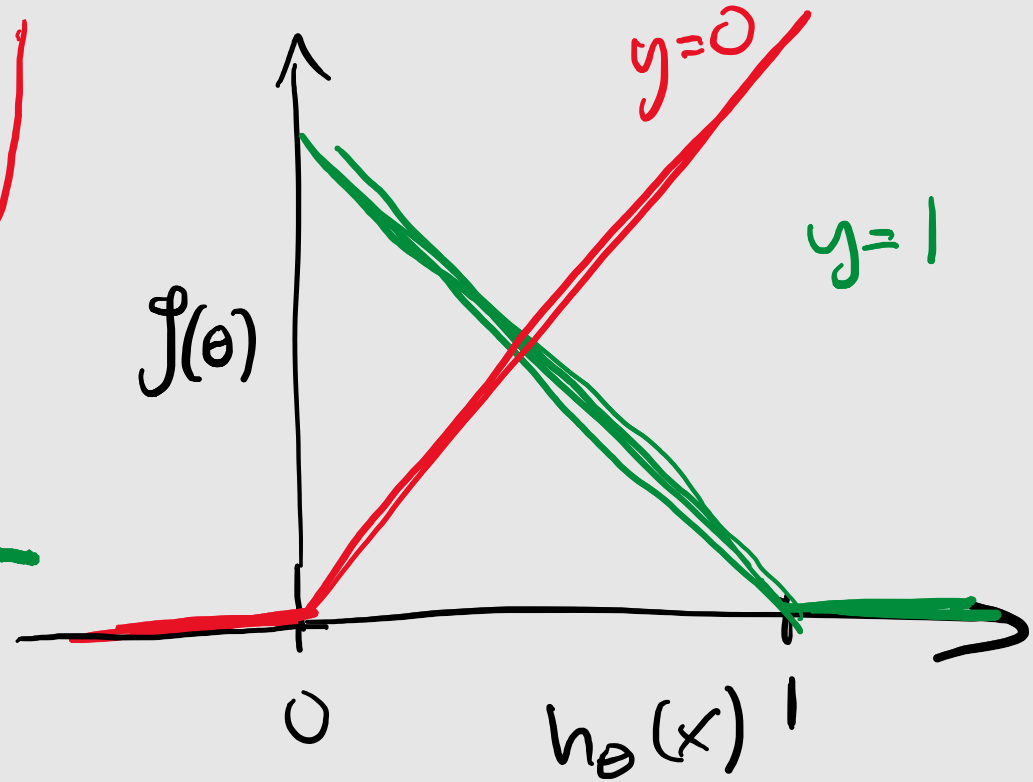
- The most common and fundamental task
- two or more classes of objects
- object is segmented and localised => presented as an image

Support Vector Machine or Large Margin Classifier

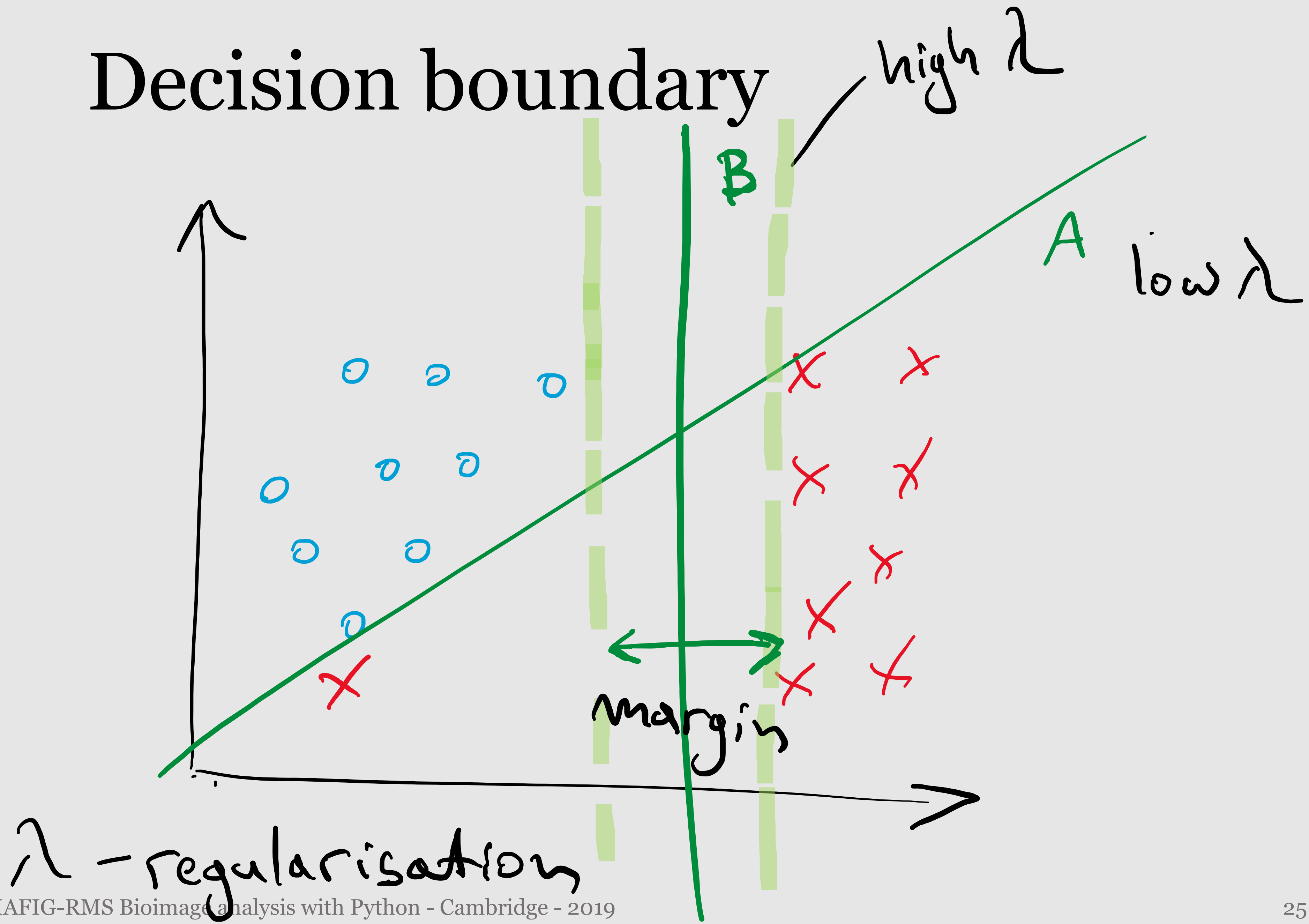
From logistic regression to kernel methods

cost in logistic regression

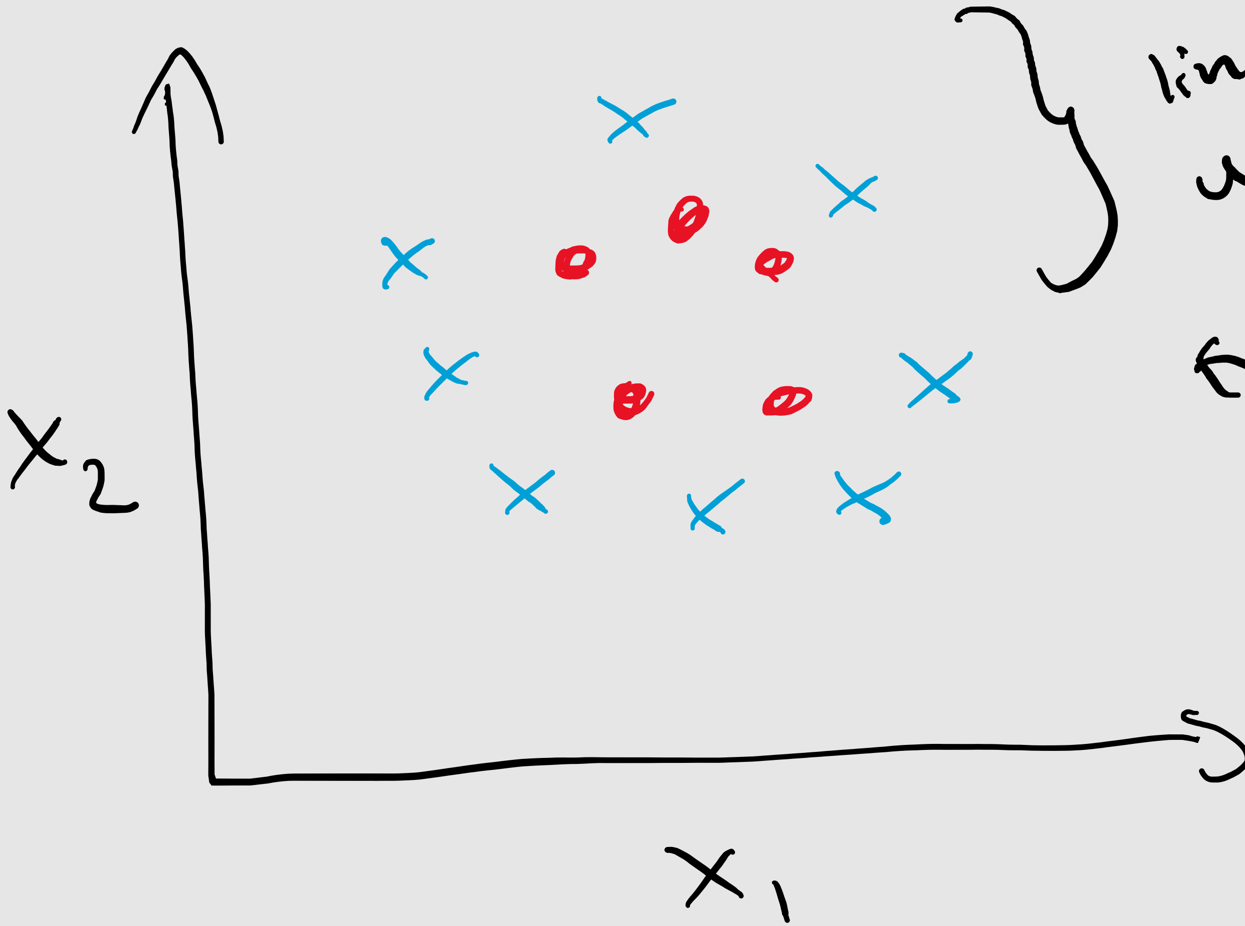




Decision boundary



Kernel “trick”

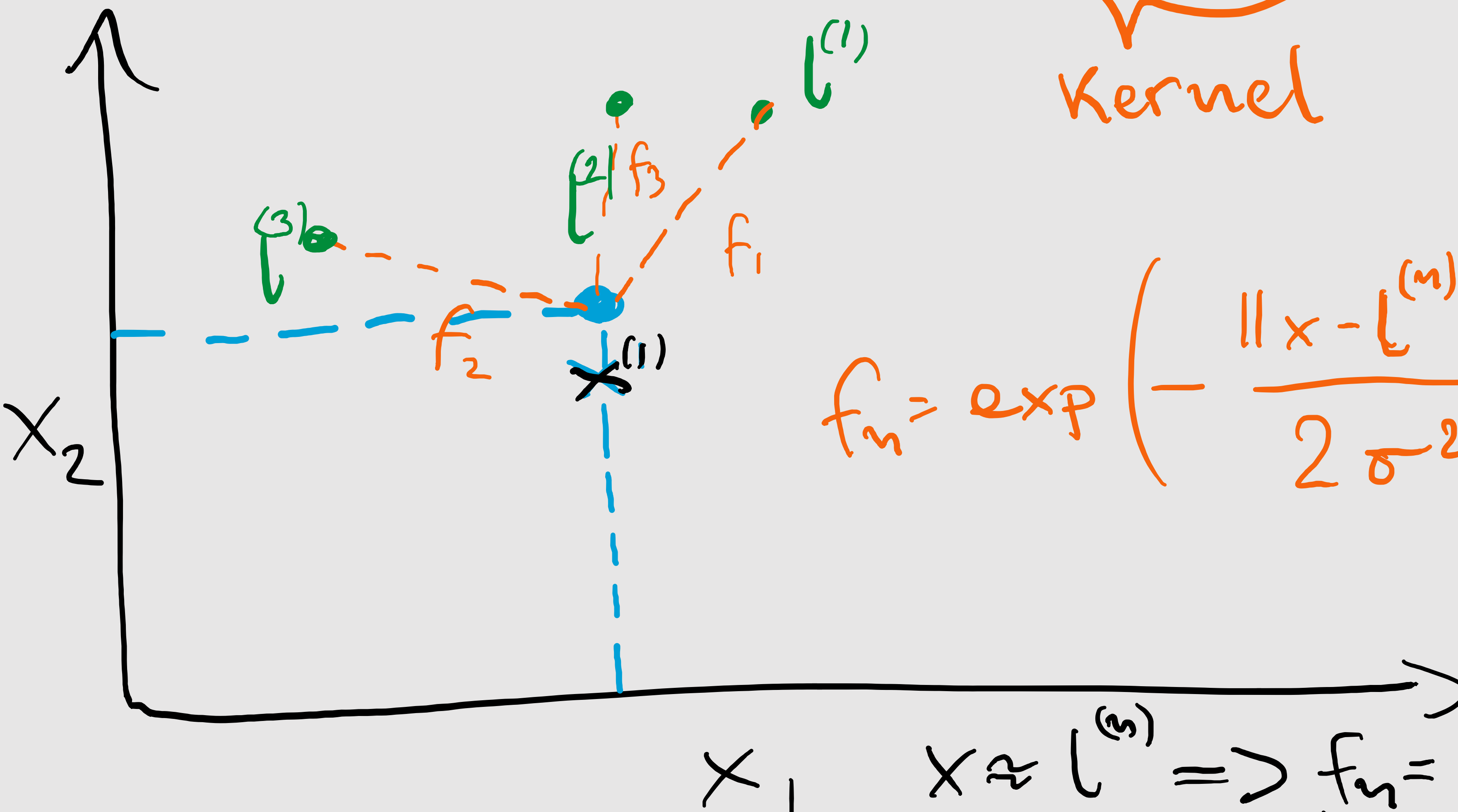


linearly unseparable

← using high polynomials risks overfitting!

Kernel "trick"

$$f_n = \underbrace{\text{similarity}(x, l^{(n)})}_{\text{Kernel}}$$



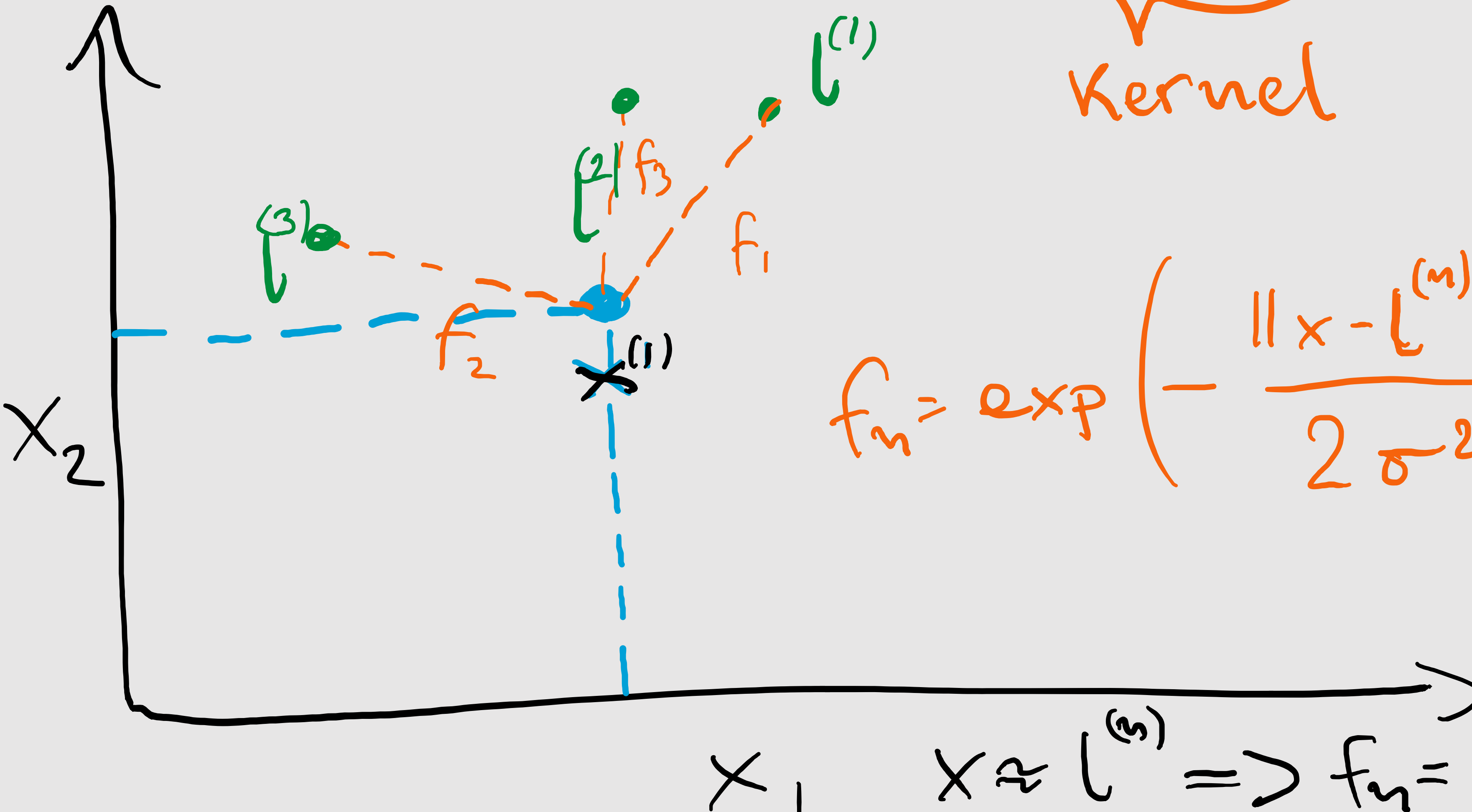
$$f_n = \exp\left(-\frac{\|x - l^{(n)}\|^2}{2\sigma^2}\right)$$

$x \approx l^{(n)} \Rightarrow f_n = 1$

x far from $l^{(n)} \Rightarrow f_n = 0$

Kernel "trick"

$$f_n = \underbrace{\text{similarity}(x, l^{(n)})}_{\text{Kernel}}$$



$$f_n = \exp\left(-\frac{\|x - l^{(n)}\|^2}{2\sigma^2}\right)$$

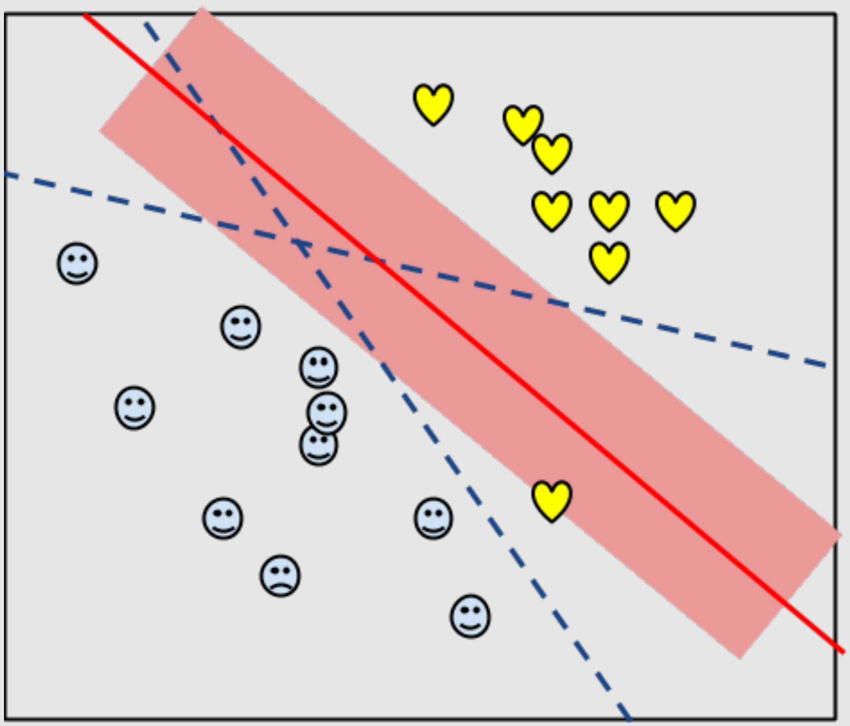
$x \approx l^{(n)} \Rightarrow f_n = 1$

x far from $l^{(n)} \Rightarrow f_n = 0$

Similarity

- If $x \approx l^{(n)} \Rightarrow f_n = 1$, x far from $l^{(n)} \Rightarrow f_n \approx 0$
- What if we have a lot of landmarks?
- What if all training examples are landmarks?
- Different kernels (Gaussian, Linear, Polynomial) depending on relationship in data

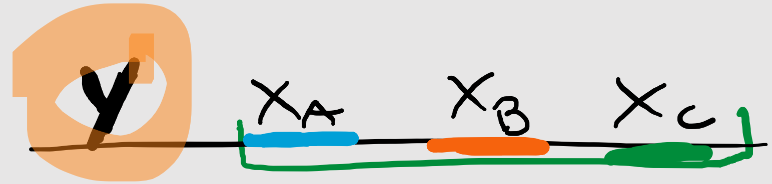
Support Vector (...machine) == Large Margin Classifier



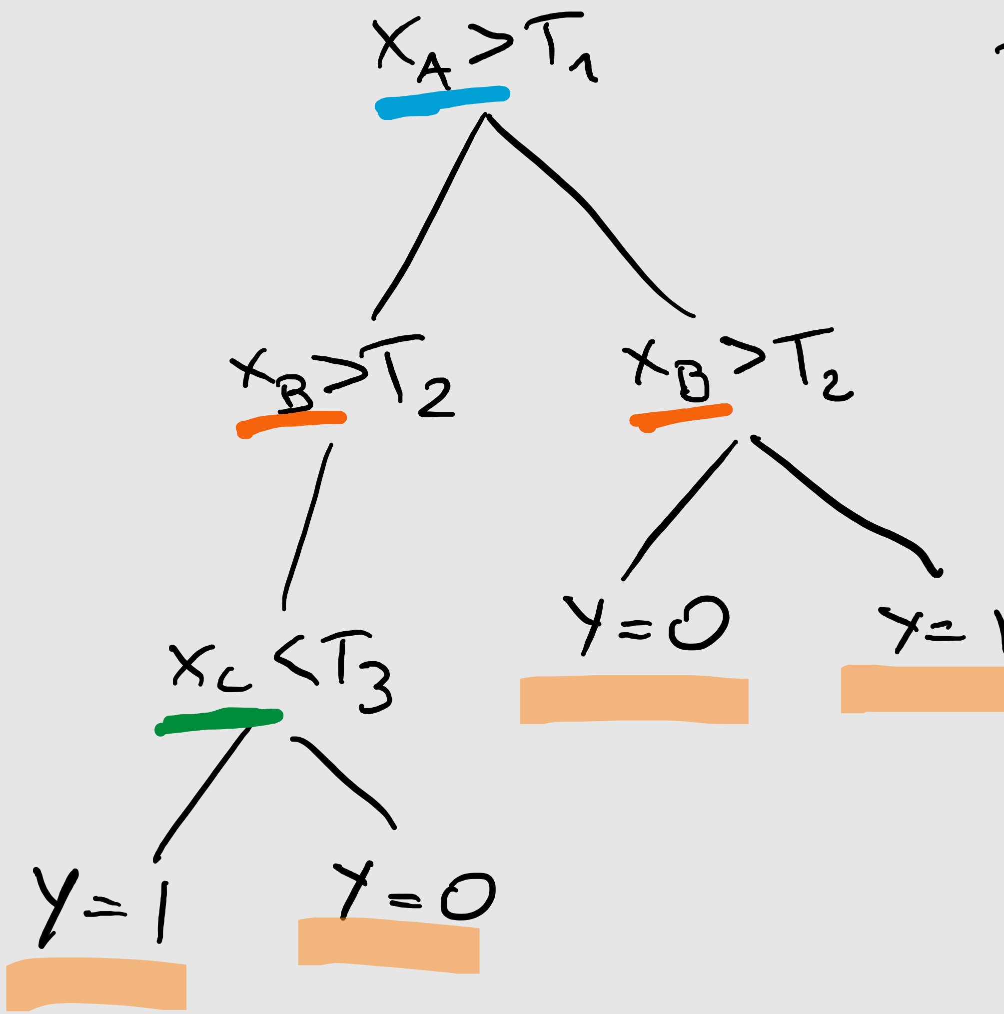
Basically, linear regression with slightly modified cost function and transformed features...

Let's make features out of data!

Decision tree

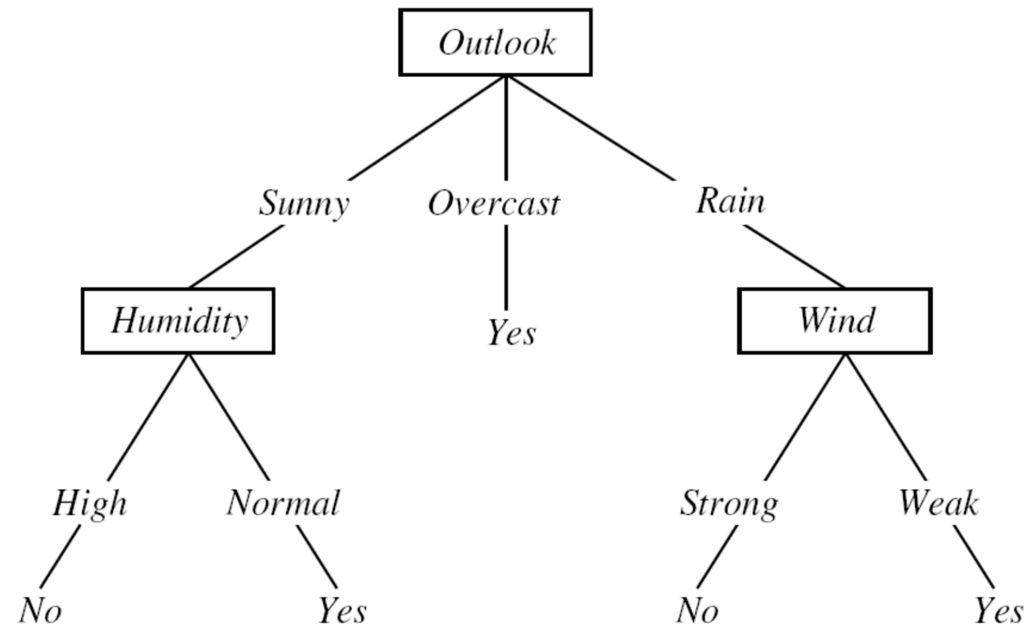


- optimise to minimize entropy with every division



Decision tree

Safe conditions to fly ?



Attributes **Classification**

Outlook	Temperature	Humidity	Windy	Fly
Sunny	85	85	False	No
Sunny	80	90	True	No
Overcast	83	86	False	Yes
Rainy	75	80	False	Yes
…	…	…	…	…

source: http://breckon.eu/toby/teaching/mltutorial/

Random Forest

- One decision tree is a basic classifier
- Train multiple small trees (each on subset of features)
- Let them vote on the final outcome
- Principles of “many wrong”
- Small trees == regularisation, no overfitting
- Routinely used to classify vast datasets from particle physics and astronomy